Trabajo Fin de Grado
Ingeniería de Telecomunicación

Aplicación de gestión de entrenamiento con notificaciones de enventos mediante cola kafka

Autor: Pablo Gallegos Jiménez
Tutor: Mª Teresa Ariza Gómez

Departamento de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2015

Trabajo Fin de Grado
Ingeniería de Telecomunicación

Aplicación de gestión de entrenamiento con notificaciones de enventos mediante cola kafka

Autor:
Pablo Gallegos Jiménez

Tutor:
Mª Teresa Ariza Gómez
Profesora titular

Departamento de Ingeniería Telemática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2015

Trabajo Fin de Grado: Aplicación de gestión de entrenamiento con notificaciones de enventos mediante cola kafka

Autor: Pablo Gallegos Jiménez

Tutora: Mª Teresa Ariza Gómez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2015

El Secretario del Tribunal

Resumen

Los avances tecnológicos han permitido que hoy en día prácticamente todo el mundo posea un Smartphone. Esto permite desarrollar un mayor número de aplicaciones para los Smartphone sabiendo que sus posibilidades de uso son casi infinitas.

En nuestro caso, se ha desarrollado una herramienta que permite gestionar el entrenamiento de un alto número de usuarios a distancia, cuya única condición es que dispongan de un Smartphone o Tablet con el sistema operativo Android y una conexión a internet. Con ello podemos ayudar a evitar el actual estilo de vida sedentario.

El objetivo principal del proyecto consiste en realizar la herramienta con la cual el entrenador podrá gestionar el entrenamiento de los diversos usuarios desde una interfaz sencilla y cómoda y conseguir una comunicación rápida, directa y fluida entre los implicados.

Abstract

The technological improvements have allowed that nowadays everyone has a smartphone. This enable develop a lot of smartphones applications knowing that their possible uses are almost endless.

In this project, we have developed a tool that let you manage the workout of a lot of users at distance who only need an android smartphone or tablet and Internet connection. This can help prevent current sedentary lifestyle.

The main objective of the project consists of making the tool with which the coach can manage workout of the different users a simple interface and useful and get a fast communication direct and fluent among those involved.

Índice

Índice de Figuras

CAPÍTULO 1 INTRODUCCIÓN

Los avances tecnológicos han permitido que hoy en día prácticamente todo el mundo posea un Smartphone. Esto permite desarrollar un mayor número de aplicaciones para los Smartphone sabiendo que sus posibilidades de uso son casi infinitas.

En nuestro caso se ha desarrollado una herramienta que permite gestionar el entrenamiento de un alto número de usuarios a distancia, y así evitar en lo posible, que el estilo de vida actual impida a un gran número de personas realizar una actividad física necesaria para llevar una vida saludable.

El entrenador desde su ordenador asigna diferentes ejercicios o rutinas a un usuario. Éste desde su Smartphone podrá visualizarlos y realizarlos cuando desee, permitiéndole así una mayor autonomía. Además de esto, para una mayor supervisión por parte del entrenador, éste recibirá diversas notificaciones automáticas.

1.1 Antecedentes

Este proyecto continua la línea de trabajo que comenzaron Juan García Piosa y Antonio José Díaz Lora. El objetivo de estos trabajos era realizar una herramienta de entrenamiento a distancia que permitiera a un fisioterapeuta o a un entrenador gestionar los ejercicios realizados por los usuarios.

El proyecto realizado por Juan García Piosa abarcaba tanto el software del usuario como el del entrenador. Debido a su gran extensión, no se hizo hincapié en la interfaz de usuario, por ello se podía mejorar para conseguir un software más fácil de usar por el usuario. De este proyecto se hereda parte de la arquitectura y el código no ha sido reutilizado.

El proyecto realizado por Antonio José Díaz Lora se centraba en desarrollar una herramienta Android cuyo objetivo era que el usuario pudiera desde su Smartphone realizar un entrenamiento guiado. Este software ha sido usado, aunque con diversas

modificaciones que se detallarán más adelante, en este proyecto.

1.2 Objetivos

Esta herramienta intenta ser un apoyo importante para la realización de ejercicios, ya sea a distancia o incluso para realizarlos en el mismo gimnasio, permitiendo a los fisioterapeutas o entrenadores dejar sus tareas programadas y encargarse exclusivamente de que todo vaya correctamente.

El objetivo de este trabajo es crear una nueva herramienta para facilitar la gestión del entrenamiento por parte del personal especializado, realizando interacciones correctas con la herramienta Android. Además de esto, también tiene como objetivo la instalación de una cola de mensajes distribuidos, en este caso se ha elegido Kafka, que permita una mejor comunicación entre las partes implicadas.

Por lo tanto podemos distinguir en el trabajo realizado tres partes claramente diferenciadas, de distinta extensión y complejidad, las cuales serán detalladas en sus respectivos capítulos:

- Herramienta java de gestión que permite al entrenador gestionar los usuarios/las rutinas y los ejercicios de la base de datos. Además desde esta herramienta también podemos visualizar los diferentes eventos.
- Cola de mensajes Kafka, encargada de almacenar los diferentes eventos producidos por el usuario.
- Modificaciones en la herramienta Android, la cual permite al usuario realizar los entrenamientos asignados. En cuando a las modificaciones realizadas podemos destacar:
  - Exportar el proyecto de Eclipse a Android Studio.
  - Desarrollo de nueva clase encargada de realizar la petición de envió de mensaje a la cola Kafka.

- Creación de código PHP que atienda esta petición y envié el mensaje a la cola.
- Creación de un botón en la pantalla principal que permite modificar la IP del servidor sin necesidad de modificar el código.
- Modificación en el código PHP encargado de recordar la contraseña.

1.3 Agentes del sistema y arquitectura

- **Servidor**: Se sitúa en el gimnasio, en el equipo donde esté ubicada la base de datos. Se instalará la cola Kafka y el servidor PHP.
- **Aplicación Monitor:** Denominaremos de esta manera a la aplicación Java desde la cual el monitor podrá gestionar usuarios, rutinas y ejercicios, además de ver los mensajes enviados a la cola Kafka.

  En este caso ha sido configurado para estar situado en el mismo equipo que el servidor, no obstante con unos sencillos cambios de IP en la configuración podríamos situarlo en otro equipo diferente. La aplicación Monitor se encarga de conectarse al servidor y modificar la base de datos, creando nuevos elementos o modificándolos según vea oportuno el entrenador. También mostrará la información actualizada que se encuentre en la cola Kafka del servidor.
- **Aplicación Android:** Es la aplicación desde la cual los usuarios pueden ver los ejercicios que tienen asignados y realizarlos de una manera cómoda. Además manda eventos al servidor.

  La aplicación Android realiza peticiones al servidor que mediante PHP accede a la base de datos y devuelve la información a la aplicación. Además de esto al hacer click en algunos botones se producirán eventos que serán enviados al servidor para que sean agregados a la cola Kafka.

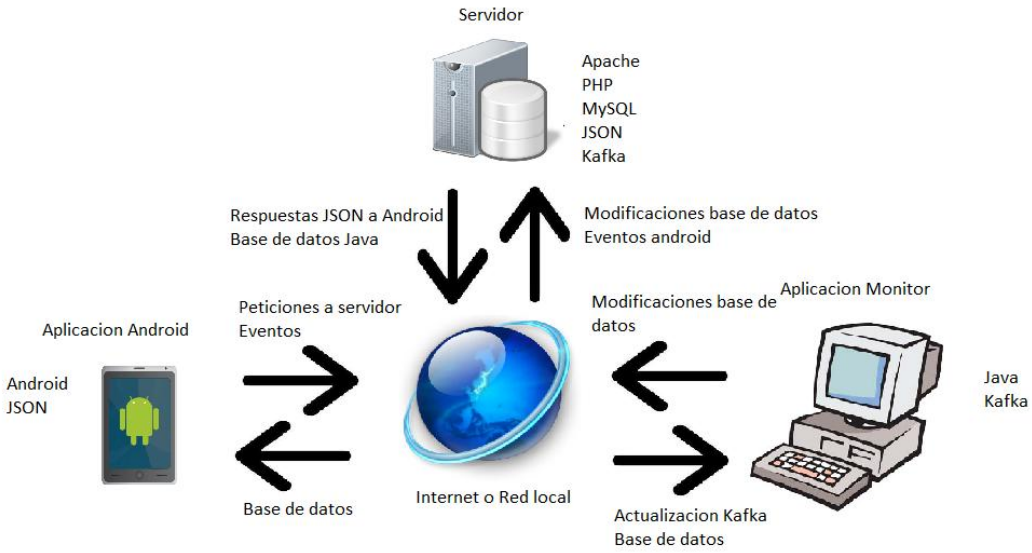

Ilustración 1: Arquitectura de Red

1.4 Estructura de la memoria

En cuanto a la memoria, dispondrá de 6 capítulos y 2 anexos con la siguiente estructura:

- Capítulo 1: Introducción, donde se explicará de manera clara y concisa los antecedentes y los objetivos, la arquitectura del sistema y la estructuración de la memoria.
- Capítulo 2: Tecnologías y entornos de trabajo, en el que se explicarán las tecnologías más importantes usadas para la realización de este proyecto y el entorno utilizado para el desarrollo del mismo.
- Capítulo 3: Herramienta monitor java, donde se detallará la herramienta Java encargada de la gestión de la base de datos y de la recepción de los mensajes, explicando sus diferentes funciones y realizando pruebas de uso.
- Capítulo 4: Herramienta usuario android. En este capítulo se detallará la herramienta Android encargada de mostrar al usuario los entrenamientos asignados, comenzando explicando los cambios más importantes realizados y terminando con una demostración de uso.

- Capítulo 5: Apache Kafka, donde se explicará con mayor detalle en qué consiste la tecnología para la comunicación entren usuario y entrenador, Kafka.
- Capítulo 6: Conclusiones, donde se resumen las conclusiones del proyecto y se dan pautas para posibles líneas de continuación.
- Anexo A: Manual de instalación
- Anexo B: Códigos del proyecto.

CAPÍTULO 2 TECNOLOGÍAS Y ENTORNOS DE TRABAJO

2.1 XAMPP

XAMPP es una distribución de Apache completamente gratuita y fácil de instalar que contiene **MySQL**, **PHP** y **Perl**. En pocas palabras Xampp es un servidor web que podremos instalar de forma local en nuestras máquinas, y con ello implementar un entorno de desarrollo para realizar nuestras pruebas antes de lanzarlo a un entorno de producción. XAMPP ofrece el ambiente ideal para el desarrollo de aplicaciones basadas en PHP.

En nuestro caso, el trabajo se ha realizado usando LAMP.

LAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- **L**inux, el sistema operativo. En algunos casos también se refiere a LDAP.
- **A**pache, el servidor web.
- **M**ySQL/**M**ariaDB, el gestor de base de datos, en este caso MySQL.
- **P**erl, **P**HP o **P**hyton, los lenguajes de programación, en este caso PHP.

La distribución usada también contiene algún modulo útil como phpMyAdmin, la cual de una forma intuitiva nos permite realizar una fácil gestión de la base de datos.

Ilustración 2: Logo de LAMP

2.2 Ubuntu

Ilustración 3: Logo de Ubuntu

Ubuntu es la distribución Linux usada usado como base para desarrollar este proyecto.

Ubuntu es un sistema operativo basado en GNU/Linux y que se distribuye como software libre, incluyendo su propio entorno de escritorio denominado Unity.

Está orientado al usuario novel y promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia de usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto.

2.3 Servidor HTTP Apache

Ilustración 4: Logo de Apache

Apache es un servidor web HTTP de código abierto multiplataforma que implementa el protocolo HTTP/1.1. El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server (httpd).

Apache presenta entre otras características altamente configurables, bases de datos de

autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Jugó un papel fundamental en el desarrollo fundamental de la World Wide Web y alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años. En 2009 se convirtió en el primer servidor web que alojó más de 100 millones de sitios web.

La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas, tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas se pueden accionar remotamente en ciertas situaciones, o explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache.

2.4 PHP

Ilustración 5: Logo de PHP

PHP (PHP Hypertext Pre-processor) es un lenguaje de programación interpretado de alto nivel, diseñado para el desarrollo web de contenido dinámico y el cual puede ser embebido en páginas HTML. Lo que distingue a PHP frente a otros lenguajes de programación como JavaScript, el cual se ejecuta en la máquina cliente, es, que el código PHP es ejecutado en el servidor.

Aunque el desarrollo de PHP está concentrado en la programación de scripts en el lado del servidor, también tiene la posibilidad de manejar imágenes, archivos PDF, películas

Flash, manejo de archivos XHTML y XML. PHP tiene soporte para una gran cantidad de Bases de Datos, así como ODBC (el estándar abierto de Conexión con Bases de Datos).

PHP cuenta también con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, POP3, HTTP, COM, etc. PHP soporta WDDX para el intercambio de datos entre lenguajes de programación web. PHP puede utilizar objetos Java de forma transparente como objetos PHP y la extensión CORBA puede ser utilizada para acceder a objetos remotos.

2.5 MySQL

Ilustración 6: Logo de MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso.

2.6 PhpMyAdmin.

Ilustración 7: Logo de phpMyAdmin

PhpMyAdmin es una herramienta de software libre escrito en PHP. Esta herramienta se desarrolló para permitir manejar MySQL (y MariaDB) de una manera más sencilla e intuitiva.

2.7 Eclipse

Ilustración 8: Logo de Eclipse

Eclipse es una plataforma de desarrollo, diseñada para ser extendida de forma indefinida a través de *plug-ins*. Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo. No tiene en mente un lenguaje específico,

sino que es un IDE genérico, aunque goza de mucha popularidad entre la comunidad de desarrolladores del lenguaje **Java** usando el *plug-in* JDT que viene incluido en la distribución estándar del IDE.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. A día de hoy Eclipse es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés *plug-in*) para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Phyton, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos

La definición que da el proyecto Eclipse acerca de su software es: "*una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular*".

2.8 Android Studio

Ilustración 9: Logo de Android Studio

Android Studio es un entorno de desarrollo integrado para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse

como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014.

Está basado en el software IntelliJ IDEA de JetBrains, y es publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux.

Algunas de sus características más importantes son:

- Renderización en tiempo real
- Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
- Soporte para construcción basada en Gradle.
- Refactorización especifica de Android y arreglos rápidos.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones, y otros problemas.
- Posibilita el **control de versiones** accediendo a un repositorio desde el que poder descargar como por ejemplo Mercurial, Git, Github o Subversion.
- **Vista previa** en diferentes dispositivos y resoluciones.
- **Integración con Google Cloud Platform**, para el acceso a los diferentes servicios que proporciona Google en la nube.
- Editor de diseño que muestra una vista previa de los cambios realizados directamente en el archivo xml.

2.9 Lenguaje de formato de datos JSON

Ilustración 10: Logo de JSON

JSON (JavaScript Object Notation) es un formato bastante ligero empleado para el intercambio de datos, siendo un subconjunto de la notación para objetos empleada en JavaScript. Pese a que JSON se basa en la notación Javascript, está considerado como un lenguaje independiente de formato de datos cuya especificación es descrita en RFC4627. La sencillez de este formato le ha dado ventaja, permitiendo una gran difusión de la tecnología como alternativa a XML. Una de las grandes ventajas de JSON sobre XML como formato de intercambio de datos se observa a la hora de escribir un analizador sintáctico (parser) para JSON es mucho más sencillo que para XML, además de procesarse más rápido el primero.

La elección de JSON se debe a que resulta muy simple tanto para humanos como para máquinas, y a que está orientado a las estructuras de datos de los lenguajes de programación modernos.

2.10 Gedit

Editor de texto de propósito general que se caracteriza por su simplicidad cuyas principales funciones extras son el coloreado de sintaxis para diversos lenguajes de programación y marcado, además de multipestaña para editar varios archivos a la vez.

En este proyecto se ha usado para crear scripts para manejar Kafka con mayor simplicidad y para pequeñas modificaciones en el código PHP heredado del proyecto anterior.

2.11 Kafka

Ilustración 11: Logo de kafka

Apache Kafka es un sistema de almacenamiento publicador/subscriptor distribuido, particionado y replicado. Estas características, añadidas a que es muy rápido en lecturas y escrituras lo convierten en una herramienta excelente para comunicar streams de información que se generan a gran velocidad y que deben ser gestionados por una o varias aplicaciones.

CAPÍTULO 3 HERRAMIENTA MONITOR JAVA

3.1 Funcionamiento

La aplicación java del entrenador funciona a través de una interfaz gráfica, donde se le presentan las opciones necesarias para poder realizar una gestión personalizada de los usuarios inscritos o crear nuevos usuarios.

Las principales funciones de la aplicación son:

- Recordar la contraseña de usuario mediante el envío de un correo electrónico.
- Crear, modificar y eliminar usuarios.
- Crear, modificar y eliminar rutinas.
- Crear, modificar y eliminar ejercicios.
- Modificar las rutinas asignadas a cada usuario, permitiendo añadir nuevas o eliminar las antiguas.
- Modificar los ejercicios que contiene cada rutina, permitiendo añadir nuevos o eliminar los antiguos.
- Modificar las fotos y los videos que se mostrarán de cada ejercicio en la aplicación Android.

3.2 Inicio de la aplicación

El programa se inicia con esta ventana, la cual permite escribir el usuario y la contraseña para iniciar sesión en el sistema o si se ha olvidado la contraseña, te permite recordarla haciendo click en ***¿Olvidó su contraseña?.***

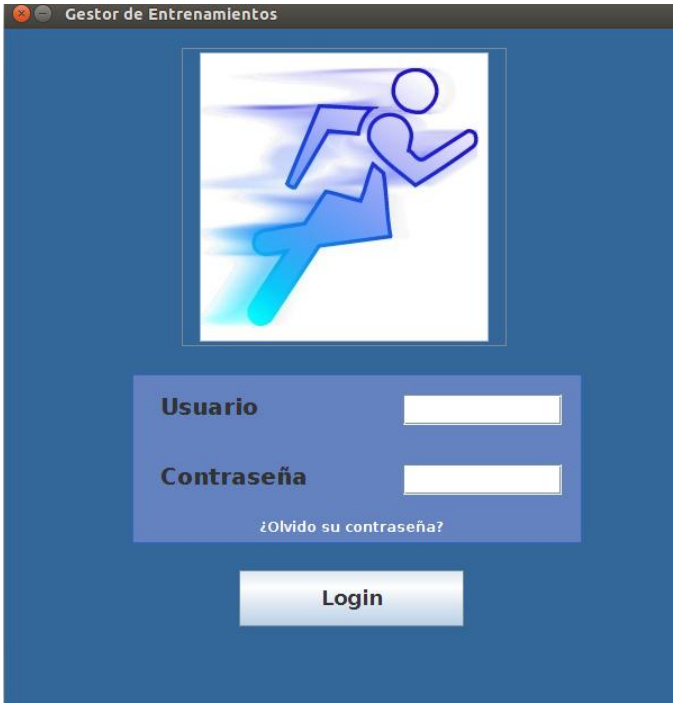

Ilustración 12: Herramienta Java Inicio

Al hacer click en el botón de olvidó su contraseña, se le solicita el usuario y el email mediante una ventana emergente.

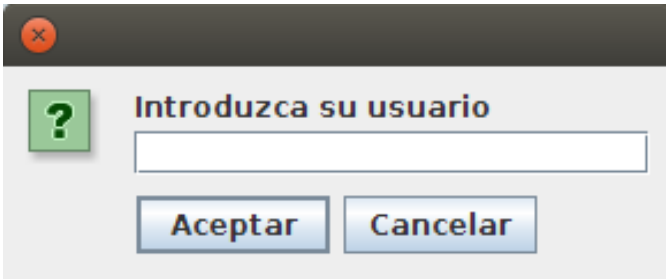

Ilustración 13: Herramienta Java-Ventana emergente-Introduzca su usuario

Durante todo el programa si dejamos en blanco una ventana emergente nos lo indicará mediante un mensaje de aviso en una nueva ventana emergente:

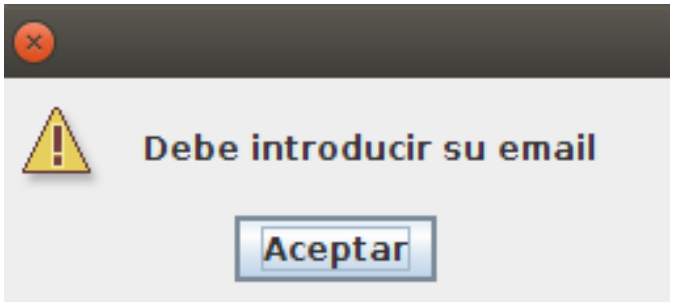

Ilustración 14: Herramienta Java-Ventana emergente-Debe introducir su email

Si se ha rellenado usuario y email pero estos no existen en la base de datos, aparecerá una ventana emergente de alerta con el siguiente mensaje: "No existe usuario con ese nombre de usuario y ese email".

Por contra si no ha habido error se te muestra un mensaje: " Se ha enviado un email con su contraseña".El contenido de ese email es el siguiente:

Ilustración 15: Herramienta Java-Contenido email

Si el registro es incorrecto se nos mostrará el siguiente mensaje de aviso: "Usuario/contraseña incorrecta"

Tras registrarse correctamente se nos muestra una ventana con cuatro pestañas, cada una con diferentes campos de texto.

Ilustración 16: Herramienta Java-Pestaña usuario

3.3 Usuario

El botón ***Limpiar*** únicamente borra todos los campos para volver a escribir lo que se desee.

El botón ***Crear*** obtiene toda la información escrita en los campos de textos y crea un nuevo usuario en la base de datos, comprobando que las contraseñas sean iguales, que no exista ningún campo vacio y que el usuario no exista en la base de datos, mostrando los respectivos mensajes de aviso en caso de que algo no sea correcto:

- Si algún campo está en blanco se nos mostrar el mensaje de alerta: "Algún campo está vacío".
- Si el texto introducido en los dos campos de contraseñas no son iguales, se nos mostrará el texto: "Las contraseñas no coinciden".
- Si el nombre de usuario elegido ya está siendo usado por algún otro usuario se nos mostrara el mensaje: " Usuario en uso".

Si todo está relleno correctamente se crea el usuario en la base de datos y se nos muestra este mensaje:

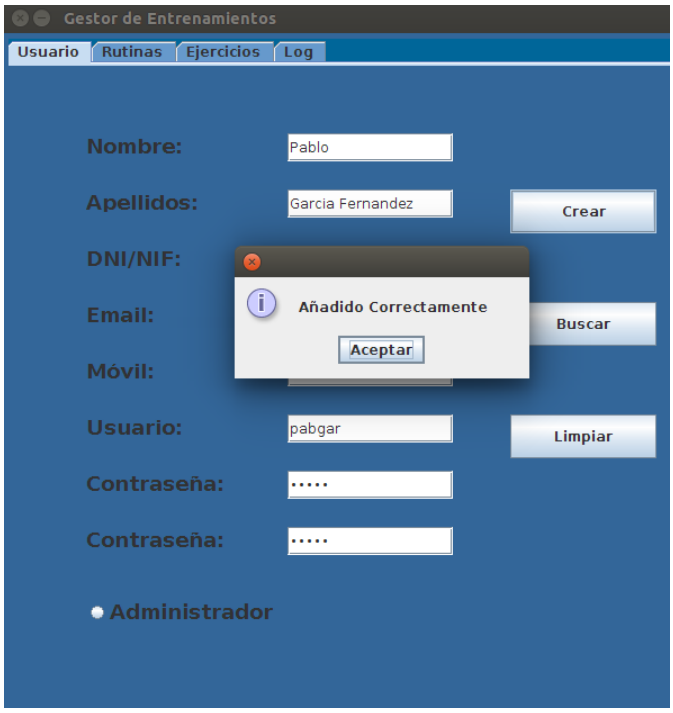

Ilustración 17: Herramienta Java-Pestaña Usuario-Usuario creado

El botón ***Buscar***, busca toda la información escrita en los campos de textos y muestra todos los usuarios que en los campos escritos tienen ese valor. En caso de estar totalmente vacío muestra todos los usuarios de la base de datos.

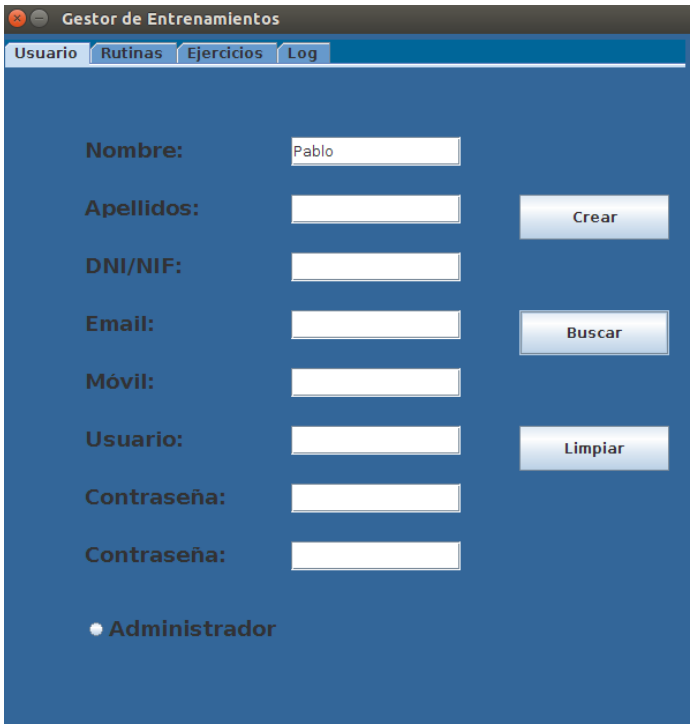

Ilustración 18: Herramienta Java-Pestaña Usuario-Buscando Pablo

En este caso hemos realizado una búsqueda usando como único parámetro de búsqueda el nombre Pablo, mostrándonos los dos usuarios cuyo nombre es Pablo. Al hacer click en uno, éste cambia de color y se nos muestra un mensaje pidiéndonos confirmación sobre si es ese el usuario que queremos ver.

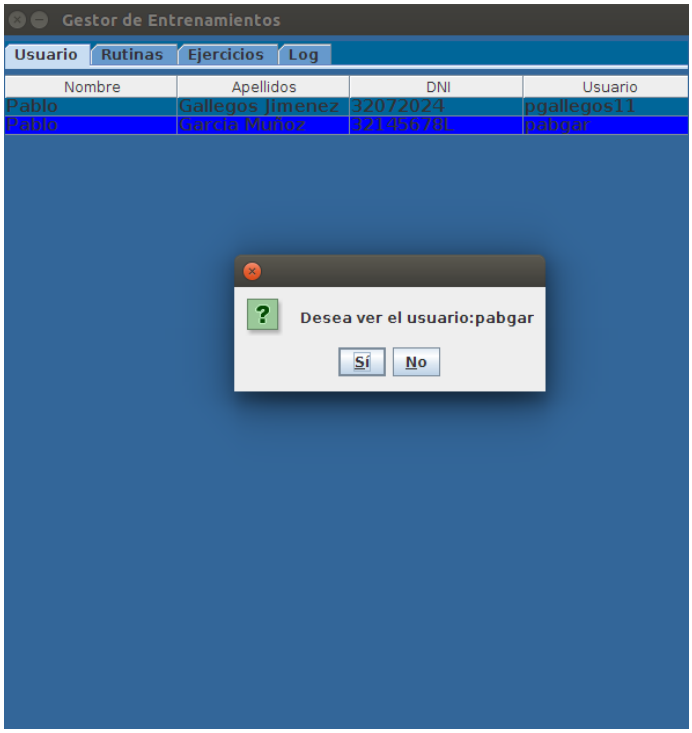

Ilustración 19: Herramienta Java-Pestaña Usuario-Usuarios encontrados

Tras aceptar, se nos muestra esta pantalla donde además de mostrarnos toda la información de dicho usuario hay 5 botones con sus diferentes funciones:

Ilustración 20: Herramienta Java-Pestaña Usuario- Usuario Pablo

El botón ***Guardar***, realiza en la base de datos los cambios que se han realizado en los diferentes campos, mostrando el mensaje de confirmación.

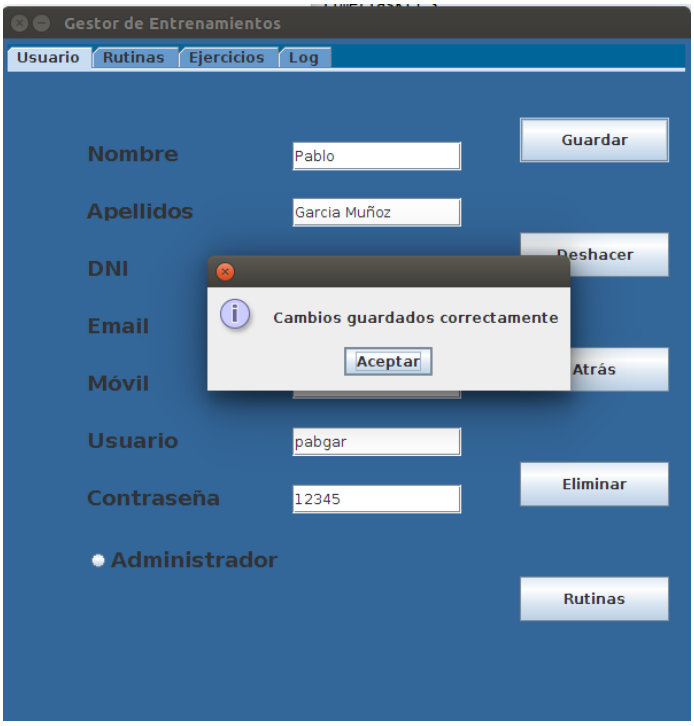

Ilustración 21: Herramienta Java-Pestaña Usuario-Modificado

El botón ***Deshacer*** , tras solicitar confirmación obtiene los datos de la base de datos y recarga los campos guardados en la misma.

El botón ***Atrás*** vuelve a la pantalla inicial desde la cual buscamos el usuario Pablo.

El botón ***Eliminar*** tras recibir confirmación elimina de la base de datos el usuario que estamos viendo y vuelve a la misma pantalla que el botón de atrás, mostrándonos un mensaje informativo:"Usuario eliminado correctamente".

El botón ***Rutinas*** es el más importante. Tras hacer click en él, se nos muestran todas las rutinas de la base de datos, divididas en las asignadas y las no asignadas al usuario:

Ilustración 22: Herramienta Java-Pestaña Usuario-Botón rutinas

Al hacer click en una rutina que no esté asignada al usuario, permite asignarla, pudiéndose así gestionar las rutinas asignadas a cada usuario:

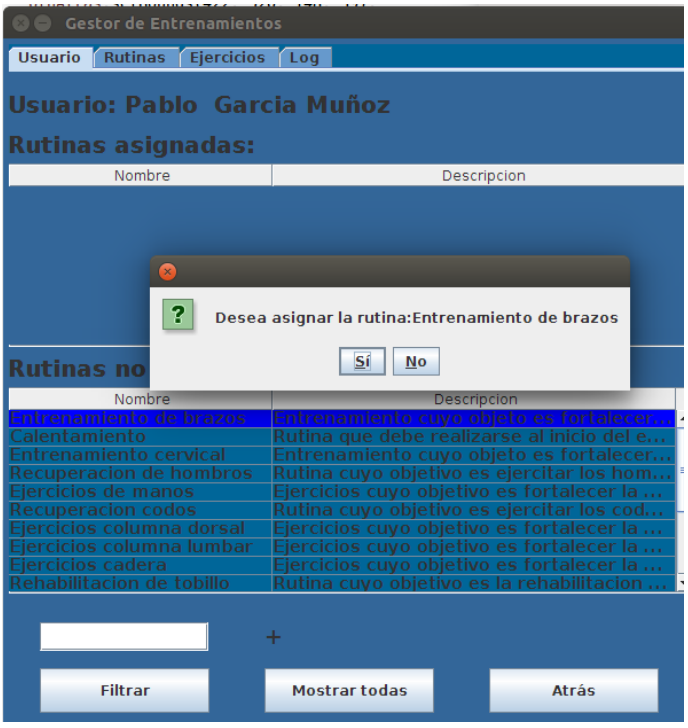

Ilustración 23: Herramienta Java-Pestaña Usuario-Asignar rutinas

Por el contrario si la rutina en la que hacemos click ya está asignada al usuario, nos permite desasignarla:

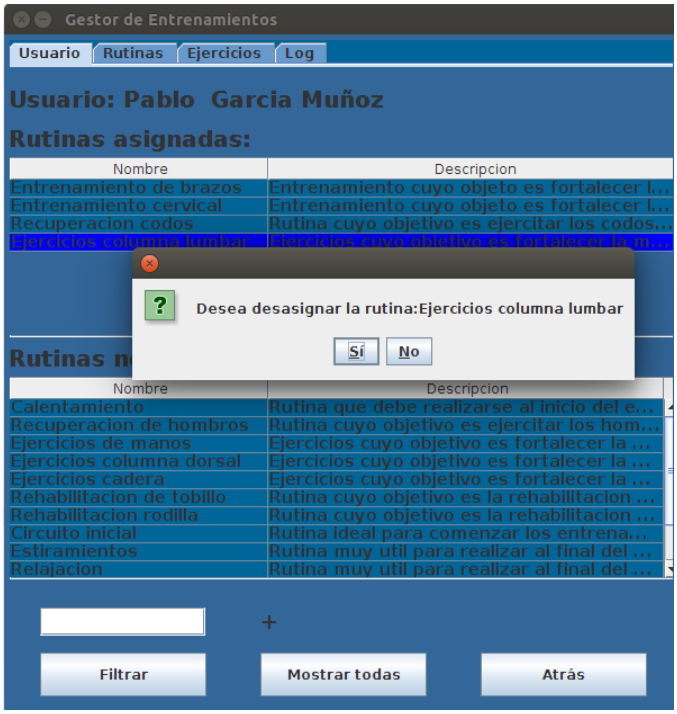

Ilustración 24: Herramienta Java-Pestaña Usuario-Desasignar rutinas

Además de esto si hacemos click en "**+**":

Ilustración 25: Herramienta Java-Pestaña Usuario-Filtros

Si se vuelve a hacer click en el nuevo "**+**" nos aparece otro cuadro de texto. Estos cuadros de textos nos permiten filtrar las rutinas no asignadas al usuario, haciendo click en el botón.

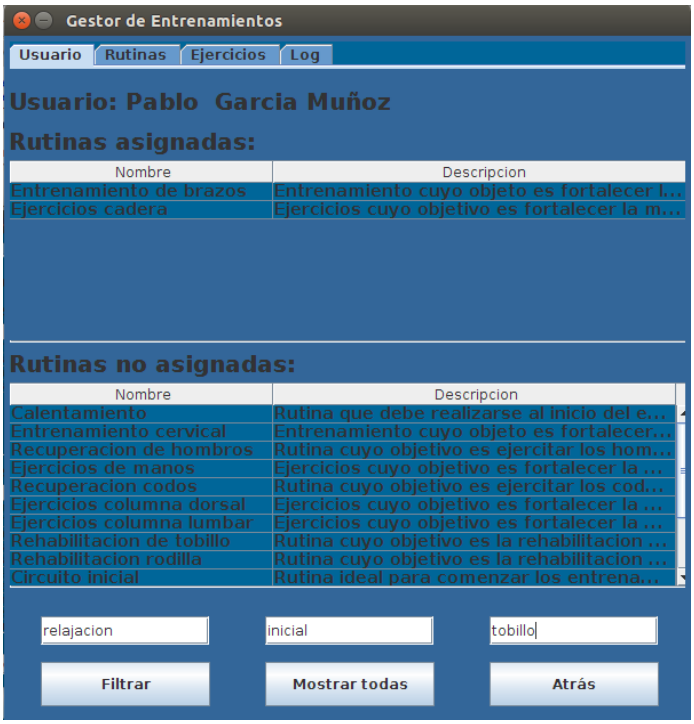

Ilustración 26: Herramienta Java-Pestaña Usuario-Filtrando

Ilustración 27: Herramienta Java-Pestaña Usuario-Filtrado

El botón ***Atrás*** nos devuelve a la pantalla del usuario concreto y por último el botón ***Mostrar todas***, elimina el filtro y muestra todas las rutinas no asignadas.

3.4 Rutinas

Hay muchas cosas que van a tener un funcionamiento similar, por lo tanto únicamente analizaremos los aspectos que más varían.

En el caso de Rutinas las diferencias principales son los campos:

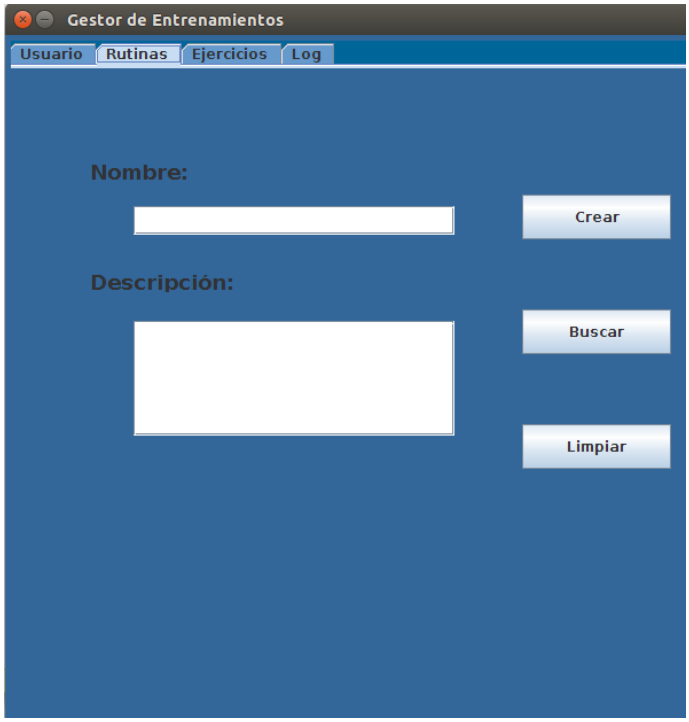

Ilustración 28: Herramienta Java-Pestaña Rutina

Además de esto, la función ***Buscar*** es algo diferente, ya que no busca la coincidencia exacta de todo el texto sino que basta con que la palabra escrita esté en la base de datos. Por ejemplo, rellenando la descripción solo con entrenamiento nos saldrán varias rutinas:

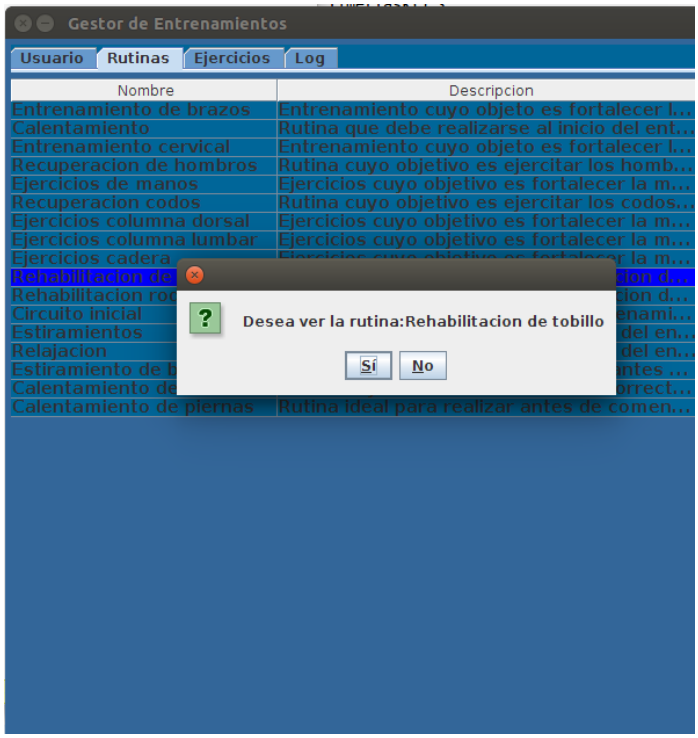

Ilustración 29: Herramienta Java-Pestaña Rutina-Buscando rutinas

Seleccionamos una rutina y vemos que en este caso el último botón es ***Ejercicios***, el cual nos permitirá gestionar los ejercicios incluidos en la rutina.

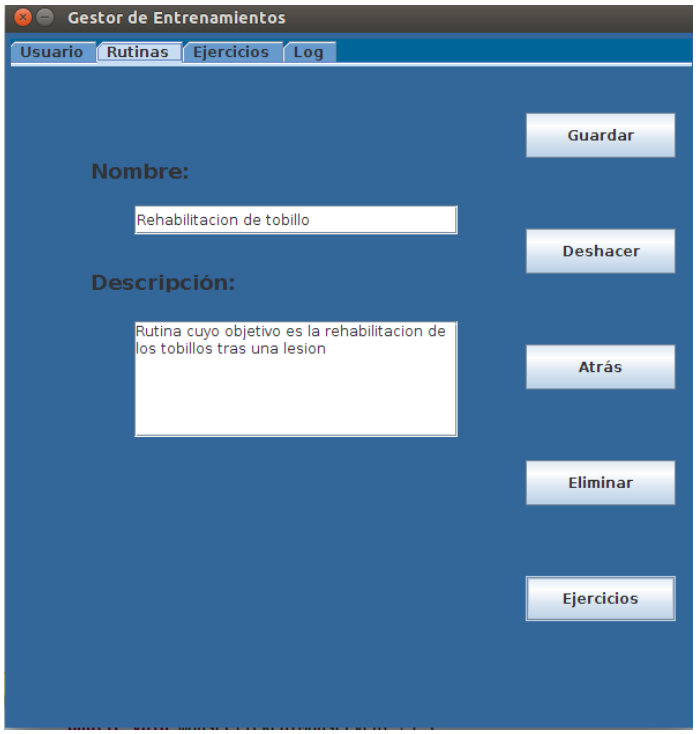

Ilustración 30: Herramienta Java-Pestaña Rutina-Rutina rehabilitación de tobillo

Al hacer click en él, vemos los diferentes ejercicios, asignados ó no, a esta rutina.

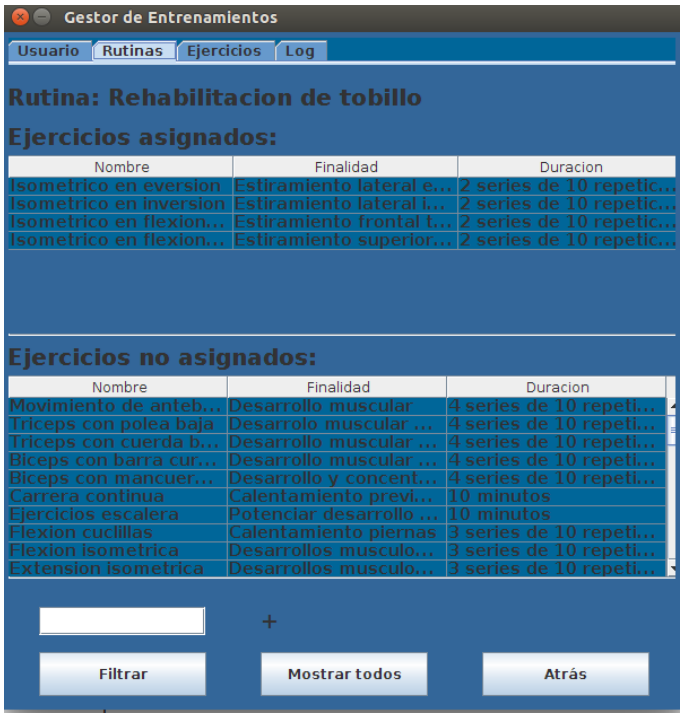

Ilustración 31: Herramienta Java-Pestaña Rutina-Botón ejercicios

3.5 Ejercicios

La pestaña Ejercicios es la más completa. En ella nos encontramos 6 cuadros de textos, pero sólo los 3 primeros son obligatorios de rellenar.

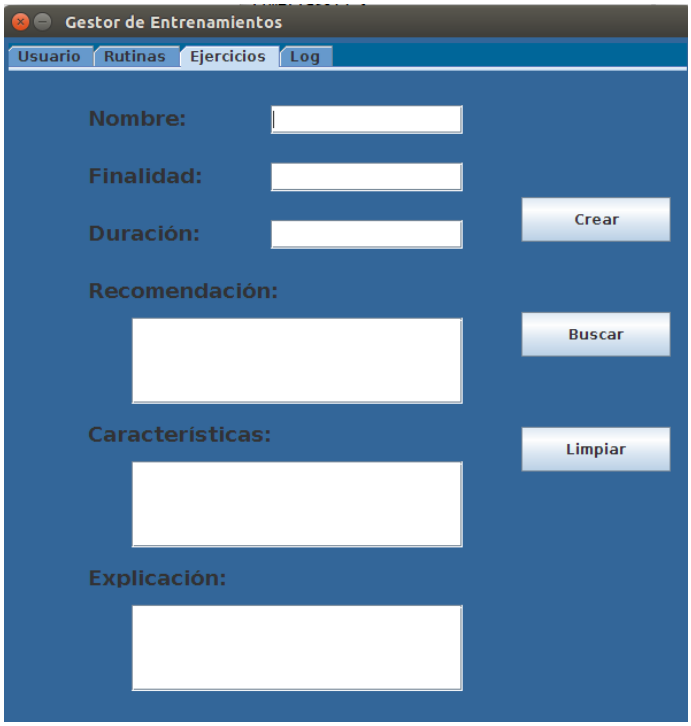

Ilustración 32: Herramienta Java-Pestaña ejercicios

Si alguno de esos cuadros tal y como pasaba en las otras pestañas está vacío, se nos mostrará el aviso:

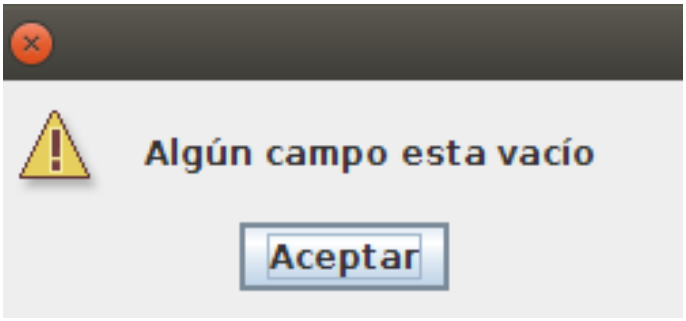

Ilustración 33: Herramienta Java-Ventana emergente-Algún campo está vació

En cambio si los campos vacíos son alguno de los inferiores se nos preguntará:

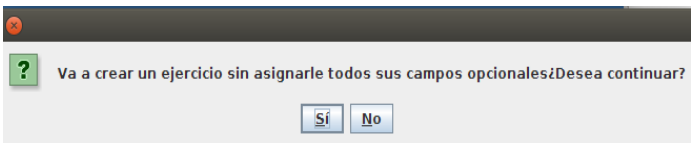

Ilustración 34: Herramienta Java-Ventana emergente-Campos opcionales

En caso de aceptar, se creará el ejercicio y podremos buscarlo de una manera similar a como lo hacíamos en el caso de las rutinas.

El botón ***Buscar*** es parecido al de la pestaña Rutina. No busca la coincidencia exacta de todo el texto sino que basta con que la palabra escrita esté en la base de datos.

Tras hacer click en un ejercicio, se nos muestra su información como anteriormente, pero con algunos botones diferentes o con algunos cambios:

Ilustración 35: Herramienta Java-Pestaña ejercicios-Ejercicio estiramiento de tobillo

El botón ***Guardar*** permite crear campos que antes no existían o modificar los que estuvieran ya creados.

Ilustración 36: Herramienta Java-Pestaña ejercicios-Ejercicio modificado

Por último, vemos que aparecen los botones ***Útiles, Músculos*** y ***Recursos***. Sus funcionamientos son similares:

Ilustración 37: Herramienta Java-Pestaña ejercicios-Útiles asignados

Como se puede ver, es parecido al funcionamiento de asignar ejercicios a rutinas o rutinas a usuarios, salvo los botones de ***Crear*** y ***Eliminar*** y la ausencia de filtro.

Al pulsar el botón ***Crear*** nos salen ventanas donde se nos solicitan todos los campos de cada elemento uno a uno:

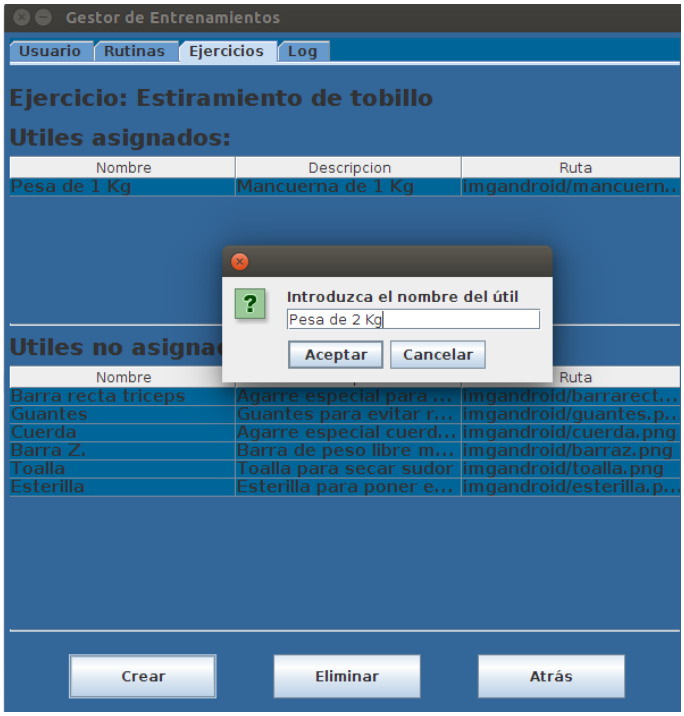

Ilustración 38: Herramienta Java-Pestaña ejercicios-Crear útil

El botón ***Eliminar*** nos solicita el nombre del elemento que se desea eliminar, y si el nombre es correcto, nos solicita confirmación tras lo cual nos muestra un mensaje indicando que se ha eliminado correctamente.

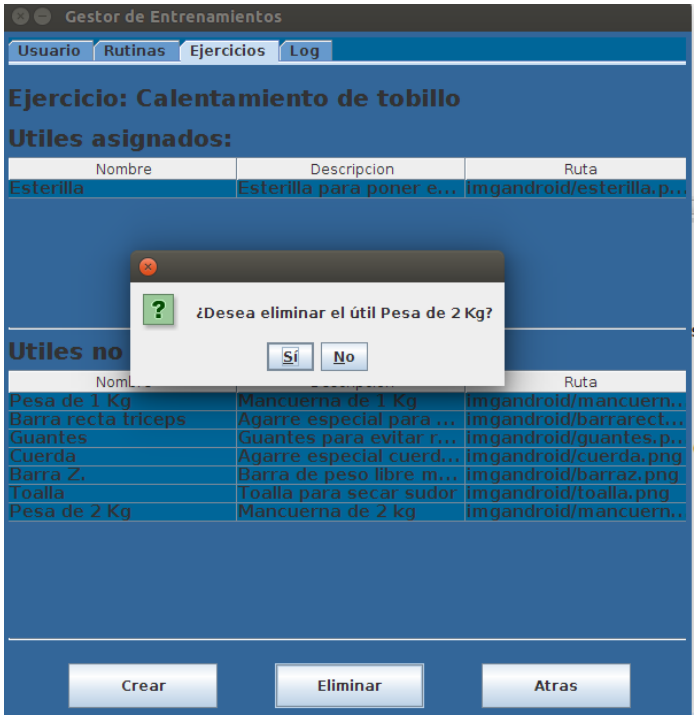

Ilustración 39: Herramienta Java-Pestaña ejercicios-Eliminar útil

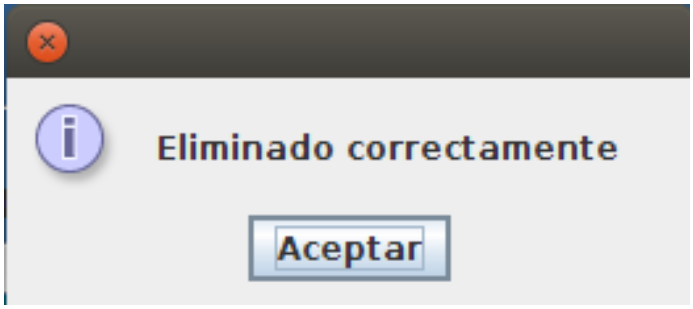

Ilustración 32: Herramienta Java-Ventana emergente-Eliminado correctamente

3.6 Log

La última pestaña nos muestra un registro de las notificaciones recibidas desde la herramienta Android. En esta pantalla se muestran, además de dos campos de texto y 3 botones los últimos 40 eventos, si aun no han ocurrido 40 eventos se mostrarían todos. Los botones "**<**" y "**>**" permiten moverse para ver los mensajes anteriores a estos 40 y para volver a los últimos mensajes.

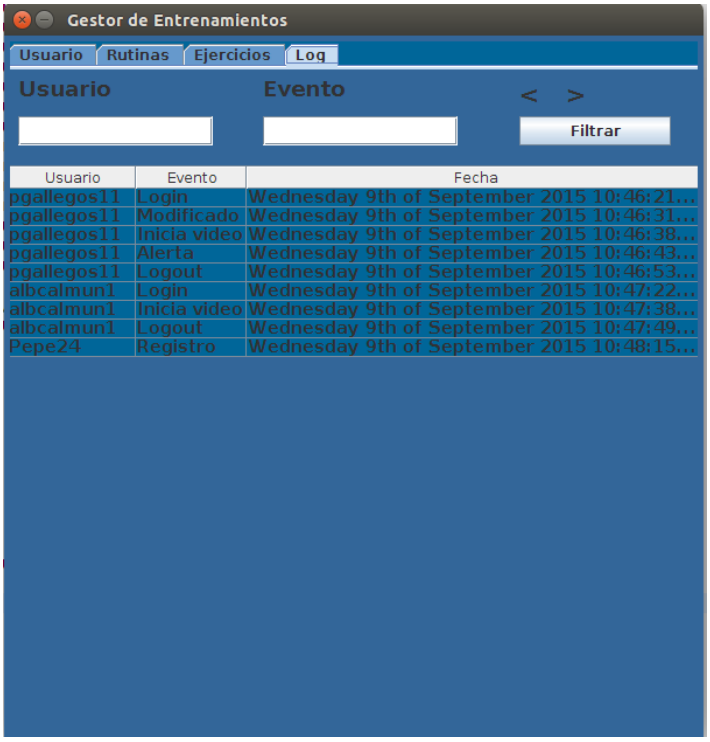

Ilustración 41: Herramienta Java-Ventana log

Los campos de texto permiten filtrar los eventos según pertenezcan a un usuario o según el tipo de evento, en esta imagen hemos realizado un filtro para ver solo los eventos de Login:

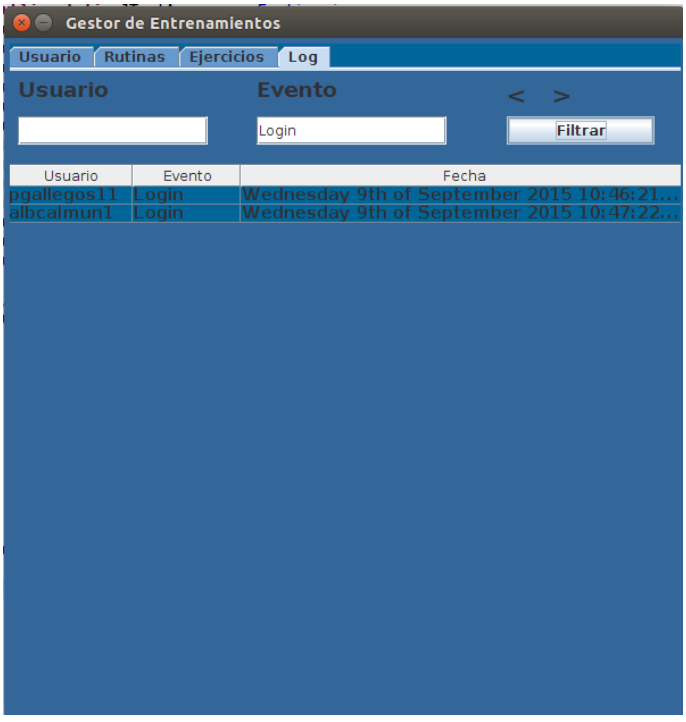

Ilustración 42: Herramienta Java-Ventana log-Filtrado

CAPÍTULO 4 HERRAMIENTA USUARIO ANDROID

4.1 Resumen del funcionamiento y cambios realizados

Esta aplicación tiene como objetivo permitir que un usuario pueda recibir en su Smartphone los ejercicios físicos que el monitor ó personal especializado considera que debe realizar.

Además de esto, y como uno de los cambios más importantes respecto al software anterior, permite enviar información al monitor, pudiendo así realizar una mejor elección de ejercicios a asignar.

Como otros cambios significativos podemos destacar:

- Posibilidad de cambiar la IP del servidor mediante un botón de configuración.
- Cambio en el recordatorio de contraseña permitiendo recibir un e-mail cuando se solicite.
- Exportar el proyecto de Eclipse a Android Studio dado que Eclipse deja de tener soporte para Android.

4.2 Funcionamiento completo

La aplicación comienza con una pantalla en la que nos encontramos diferentes opciones:

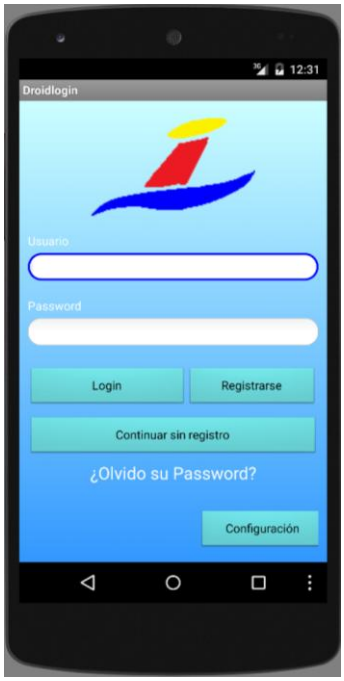

Ilustración 43: Herramienta Android-Ventana inicial

Iniciar sesión, registrarnos, cambiar la configuración, continuar sin registro y recordar la contraseña.

Como ya hemos adelantado, el botón de configuración nos permite cambiar la IP del servidor de una manera más sencilla.

Ilustración 44: Herramienta Android-Cambiando IP

Si pulsamos en ¿Olvido su Password? nos muestra la siguiente ventana, la cual nos solicita el usuario y el email, enviándonos un correo electrónico recordando nuestra contraseña. Una vez enviado el correo, se notifica automáticamente al monitor.

Ilustración 45: Herramienta Android-Recordar contraseña

Como última opción de configuración, nos permite crear una cuenta de usuario en la base de datos, permitiendo así que en un futuro un entrenador nos pueda asignar las rutinas que considere oportunas.

La creación de una cuenta también será notificada al entrenador para asignarle rutinas o incluso poder eliminarlo si así lo considera oportuno.

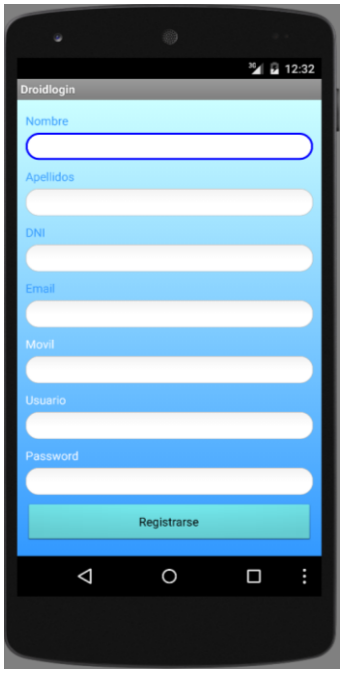

Ilustración 43: Herramienta Android-Creando Usuario

En cuanto a las opciones de continuar sin registro y Login son muy parecidas, ambas son notificadas al entrenador dado que permiten comenzar el entrenamiento.

En caso de optar por continuar sin registro, nos advierte de que nuestro acceso será limitado aunque nos permite continuar.

Ilustración 47: Herramienta Android-Ventana inicial

Como vemos en la siguiente imagen, podemos diferenciar el entrenamiento en dos modelos: el entrenamiento personalizado y el entrenamiento libre. La diferencia entre ambos consiste en que el usuario registrado tiene acceso a ambos mientras que el no registrado únicamente puede acceder a Entrenamiento Personalizado donde se le han asignado previamente unas rutinas básicas.

Ilustración 48: Herramienta Android-Comenzar entrenamiento

Si optamos por pulsar Login e introducimos el usuario y la contraseña de forma correcta, nos muestra una ventana desde la cual además de entrenar y cerrar sesión podemos modificar los datos personales.

Ilustración 49: Herramienta Android-Bienvenido Usuario

Al pulsar en Consulta/Modificación de datos nos muestra nuestros datos permitiendo realizar los cambios que deseemos. Estos cambios serán notificados también al entrenador.

Ilustración 50: Herramienta Android-Modificando datos

Si optamos por pulsar el botón de entrenar, se nos da la opción de elegir entre Entrenamiento Personalizado y Entrenamiento Libre. El entrenamiento libre únicamente muestra todas las rutinas de la base de datos y te permite elegir la que desees mientras que el entrenamiento personalizado esta supervisado por un monitor que decide que rutinas realizar.

Si pulsamos en el botón de Entrenamiento Personalizado se nos muestran todas las rutinas que el entrenador nos recomienda realizar. En este ejemplo únicamente tenemos una rutina asignada.

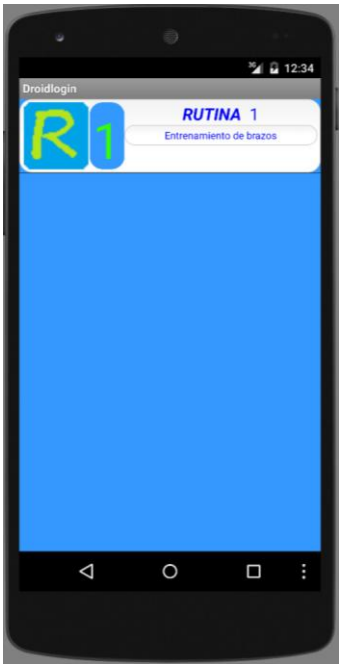

Ilustración 51: Herramienta Android-Lista de rutinas Ilustración 52: Herramienta Android-Opciones de rutina

Al pulsar en ella vemos tres botones, ver la rutina mediante videos, ejercicio paso a paso y avisar a entrenador. Este último botón al ser pulsado mandará un mensaje de alerta al monitor.

Si elegimos la opción de rutina mediante videos se nos muestra una lista de todos los videos asignados a esa rutina, permitiéndonos visualizarlos con un solo click, evento que también se notificará al monitor.

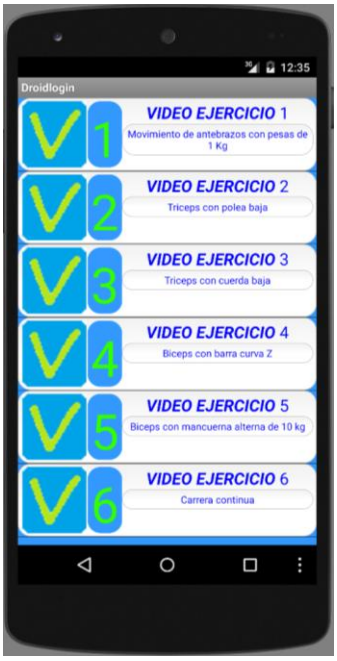

Ilustración 53: Herramienta Android-Lista de vídeos Ilustración 54: Herramienta Android-Vídeo

Si por el contrario elegimos la opción de ejercicio paso a paso se nos mostrara un listado de los ejercicios asignados a la rutina. Al hacer click en cualquiera de ellos se no muestra toda la información existente en la base de datos.

Ilustración 55: Herramienta Android-Lista de Ejercicios Ilustración 56: Herramienta Android-Ejercicios

Desde esta pantalla podemos pulsar en las lupas para acceder a más información. Los tres primeros (Explicación, Características y Recomendaciones) son campos de textos cuya interfaz es similar:

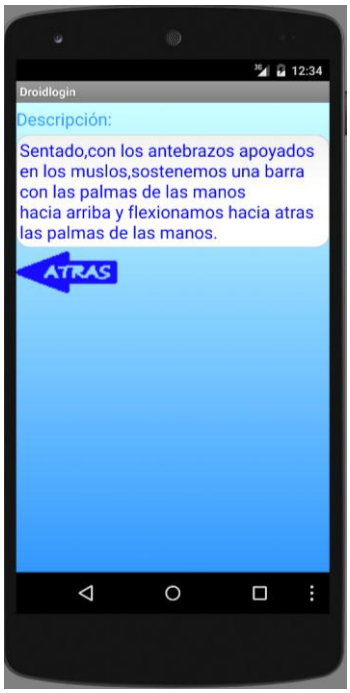

Ilustración 43: Herramienta Android-Descripción ejercicio

En cambio, si pulsamos sobre las dos siguientes (útiles y músculos), nos muestra un listado de todos los útiles y músculos asignados al ejercicio, permitiendo consultar imágenes de cada uno de ellos.

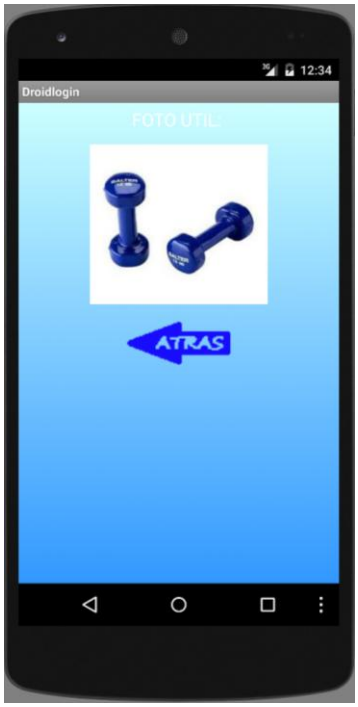

Ilustración 58: Herramienta Android-Lista de útiles Ilustración 59: Herramienta Android-Útil

Finalmente, si pulsamos la última lupa (que el simulador de Android Studio no permite ver correctamente) mostrará las imágenes asignadas a este ejercicio, pudiendo así realizar el ejercicio paso a paso, tal y como vemos en las imágenes siguientes.

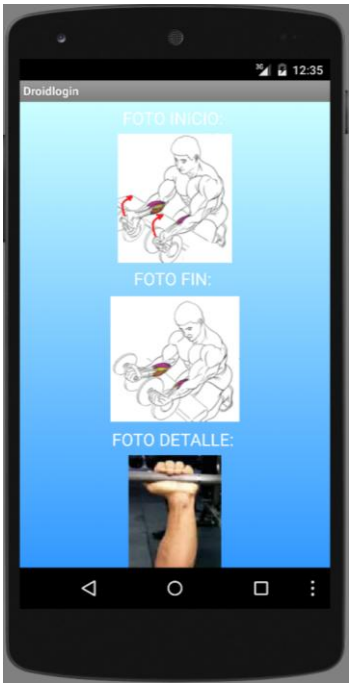

Ilustración 60: Herramienta Android-Imágenes

CAPÍTULO 5 APACHE KAFKA

5.1 Introducción

Apache Kafka es un sistema de almacenamiento publicador/subscriptor distribuido, particionado y replicado. Estas características, añadidas a que es muy rápido en lecturas y escrituras, lo convierten en una herramienta excelente para comunicar streams de información que se generan a gran velocidad y que deben ser gestionados por una o varias aplicaciones.

Podemos destacar las siguientes características:

- Mensajería persistente a estructuras de disco O1 que proporcionan un rendimiento constante en el tiempo, incluso con varios TB de mensajes almacenados.
- Alto rendimiento: incluso con hardware muy modesto Kafka puede soportar cientos de miles de mensajes por segundo.
- Soporte para la partición de mensajes a través de los servidores de Kafka y consumo distribuido en un clúster de máquinas consumidores, manteniendo la ordenación por partición
- Kafka pretende unificar el procesamiento online y offline, proporcionando un mecanismo para la carga paralela en Hadoop, así como la capacidad de partición en tiempo real del consumo en un clúster.
- Funciona como un servicio de mensajería, categoriza los mensajes en tópics.
- Los procesos que publican se denominan brokers y los subscriptores son los consumidores de los tópics.
- Utiliza un protocolo propio basado en TCP y Apache Zookeeper para almacenar el estado de los brokers. Cada broker mantiene un conjunto de particiones (primaria y secundaria) de cada tópic.
- Se pueden programar productores/consumidores en diferentes lenguajes: Java, Scala, Python, Ruby, C++ …

- Escalable y tolerante a fallos.
- Se puede utilizar para servicios de mensajería (tipo ActiveMQ o RabbitMQ), procesamiento de streams, web tracking, trazas operacionales, etc.
- Escrito en Scala.
- Creado por LinkedIn.

La Arquitectura de Kafka no pretende que un único clúster gestione varios data centers, sino para soportar una topología multi-datacenter.

Para esto permite el **mirroring** o "sincronización" entre los clúster de una forma muy sencilla: el clúster espejo simplemente actúa como un consumidor de la fuente de clúster. Esto significa que es posible que un solo grupo pueda unir datos de muchos centros de datos en un solo lugar. Un ejemplo sería:

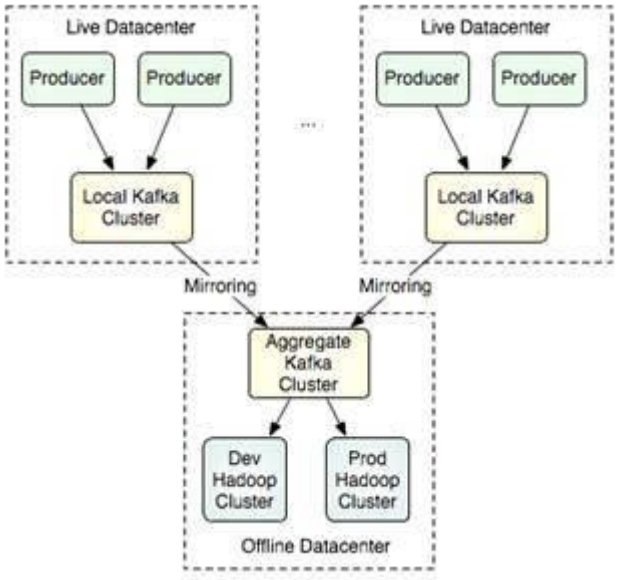

Ilustración 61: Kafka-Estructura

Actualmente Kafka es un software en pleno desarrollo y que está comenzando a extenders. A día de hoy la última versión (en fase beta) es la 0.8.2. En este proyecto se ha optado por la versión 0.7.2.

5.2 Motivación

Desde el primer momento se pensó que este proyecto necesitaba una comunicación más

directa entre usuario y monitor, permitiendo un entrenamiento más supervisado. Tras valorar varias opciones, se optó por la cola de mensajes distribuidos Kafka debido sobre todo a su velocidad.

5.3 Problemas encontrados

Uno de los principales problemas encontrados en todo el proyecto ha sido incluir la cola Kafka en Android. Esto se debe a que Kafka está construido usando Scala y ésta no tiene soporte para Android.

Tras buscar diferentes alternativas se optó por usar un servidor PHP como intermediario, realizando desde la aplicación Android una conexión al servidor y éste escribiendo los mensajes en la cola Kafka.

Esto obligó a que la versión usada en el proyecto fuese la 0.7.2, la antepenúltima versión definitiva, dado que para la versión 0.8.0, la posterior a la usada, las librerías usadas PHP no funcionan correctamente y aún están en desarrollo.

5.4 Funcionamiento

Apache Kafka se **ejecuta en modo clúster para asegurar el servicio** compuesto por varios servicios. Cada **servicio** se le denomina **broker**. Cada partición está configurada sobre un servicio y éste, a su vez, está replicado en otros servicios para aumentar la tolerancia a fallos.

De todos los servicios, uno de ellos actúa como líder controlando las operaciones de entrada-salida replicando todo la información que maneja sobre los servidores seguidores. Si el servidor líder falla, uno de los servidores seguidores asume el rol de líder. Para realizar el balanceo de servidores, Apache Kafka se ejecuta sobre ZooKeeper.

La visión arquitectónica de Apache Kafka desde un punto de vista gráfico es la siguiente:

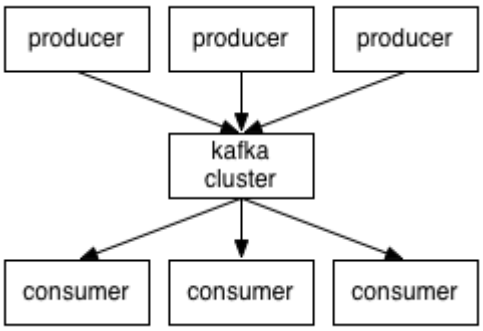

Ilustración 62: Kafka-Funcionamiento

Lo más llamativo de Apache Kafka es cómo se realiza la lectura y borrado de los mensajes:

- La lectura de los mensajes por parte de los consumidores se basa en un offset desde el inicio para leer el mensaje correspondiente en cada momento. Esto implica que una lectura no implica el borrado del mensaje leído por lo que se mejora el rendimiento de manera considerable.

- Por otro lado, el borrado de los mensajes se realiza mediante una especie de proceso desatendido que se encarga de hacer una especie de *purge* de base de datos en los ficheros de logs (particiones) en el tiempo que se configure. Esto es posible ya que la estructuración de las particiones se fundamenta en la jerarquía de ficheros con indirección.

5.5 Demostración

Los scripts usados y como configurarlos estarán en el anexo.

Como se ha dicho anteriormente, Kafka se ejecuta sobre ZooKeeper por lo tanto lo primero que se debe hacer es arrancar este servidor.

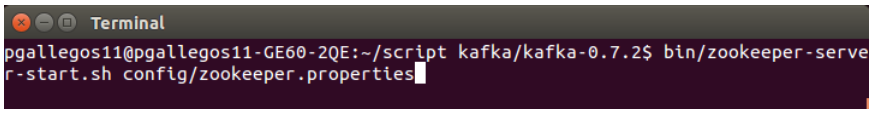

Ilustración 63: Kafka-Arrancando Zookeeper

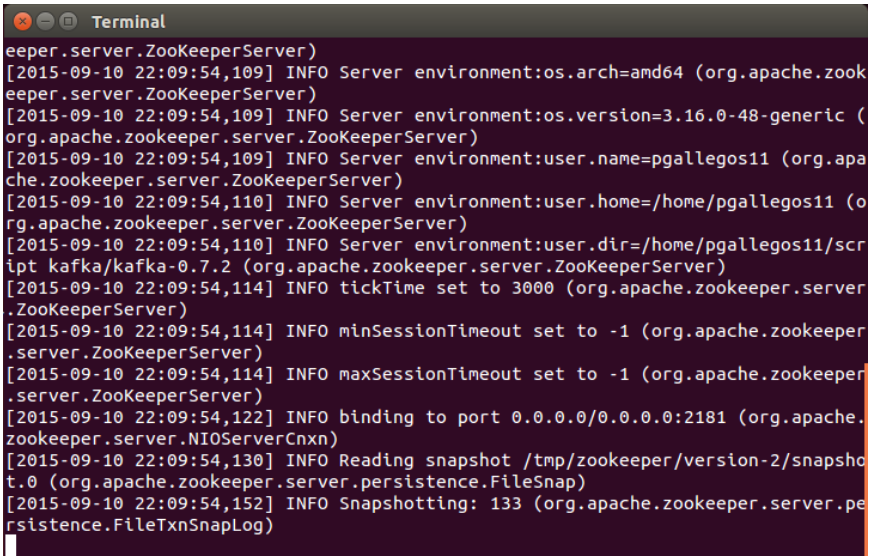

Ilustración 64: Kafka-Zookeeper arrancado

Tras eso debemos arrancar el servidor Kafka.

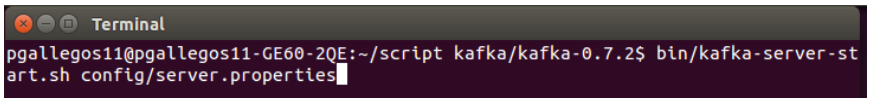

Ilustración 65: Kafka-Arrancando Kafka

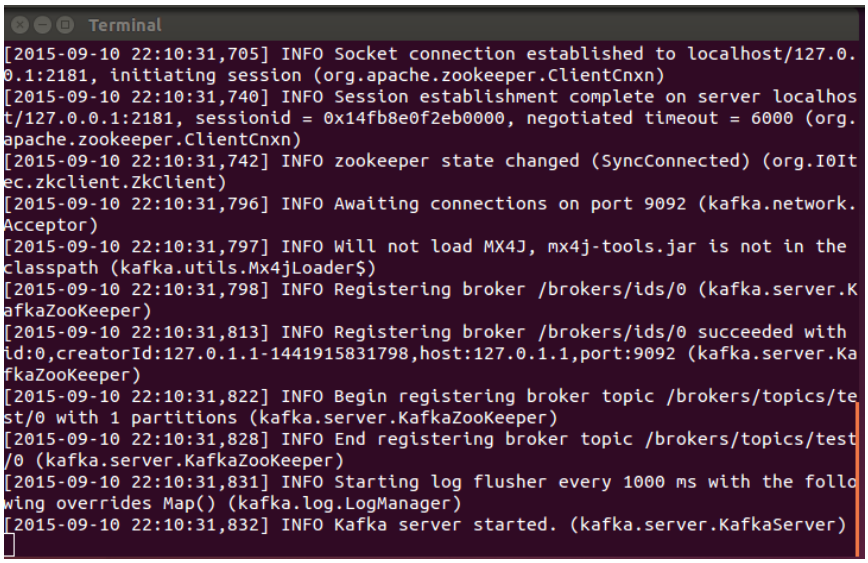

Ilustración 66: Kafka-Kafka arrancado

En este caso hemos arrancado un consumidor que leerá todos los mensajes desde el inicio de la cola cuyo topic sea test.

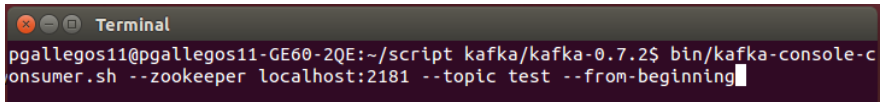

Ilustración 67: Kafka-Arrancando Consumidor

Como vemos en la siguiente imagen la cola contiene dos mensajes iguales.

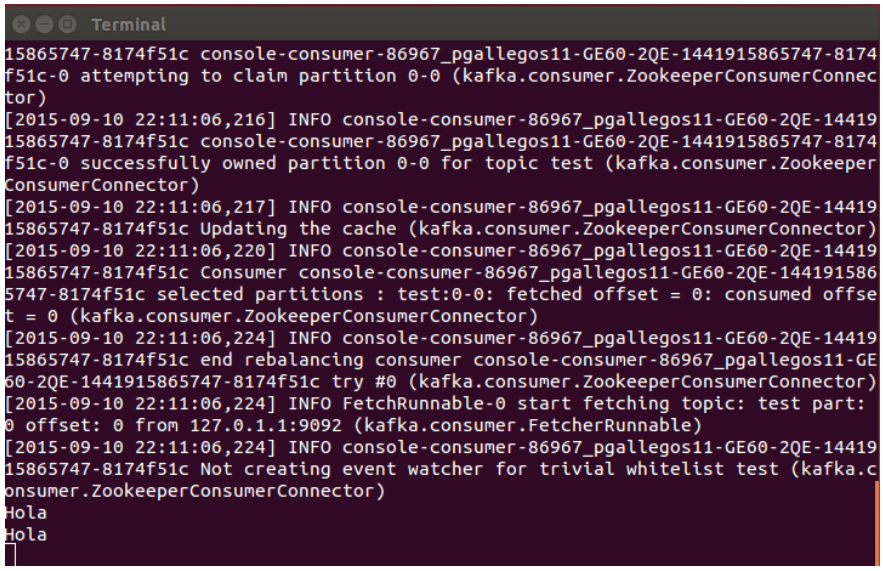

Ilustración 68: Kafka-Consumidor arrancado

También arrancamos un productor para producir nuevos mensajes en el mismo topic

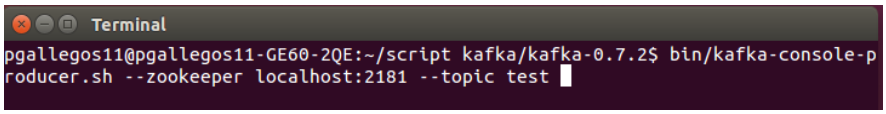

Ilustración 69: Kafka-Arrancando Productor

Al escribir un mensaje nos muestra un mensaje de información de conexión, algo que no volverá a pasar cuando volvamos a escribir otros mensajes.

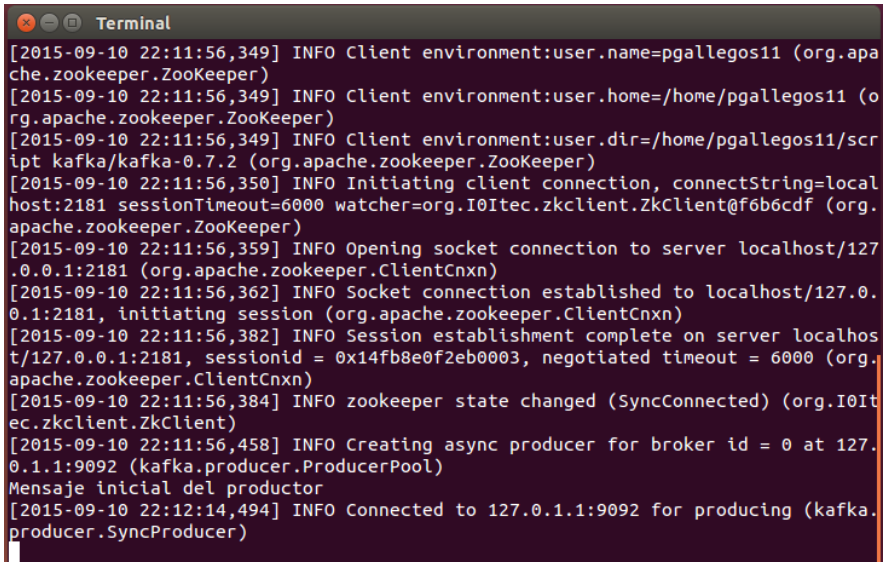

Ilustración 70: Kafka-Mensaje enviado

Este mensaje es recibido normalmente por el consumidor casi inmediatamente

Ilustración 71: Kafka-Nuevo mensaje recibido

5.6 Uso en el proyecto

En este proyecto el uso que se le da a la cola Kafka es mantener un registro de las acciones realizadas por los usuarios, esto ocurre de la siguiente manera:

Cada vez que un usuario pulse un botón considerado de interés, este realiza una petición al servidor PHP para que este sea el encargado de escribir el mensaje en la cola.

Estos mensajes son leídos automáticamente por la herramienta del monitor permitiendo de esta manera controlar en tiempo real y mantener un registro de las acciones más importantes realizadas por los usuarios.

Para este proyecto las acciones de interés han sido: registrarse, iniciar sesión, cerrar sesión, modificación de datos, iniciar una rutina e iniciar la visualización de un video explicativo.

El formato del mensaje es usuario-evento-fecha.

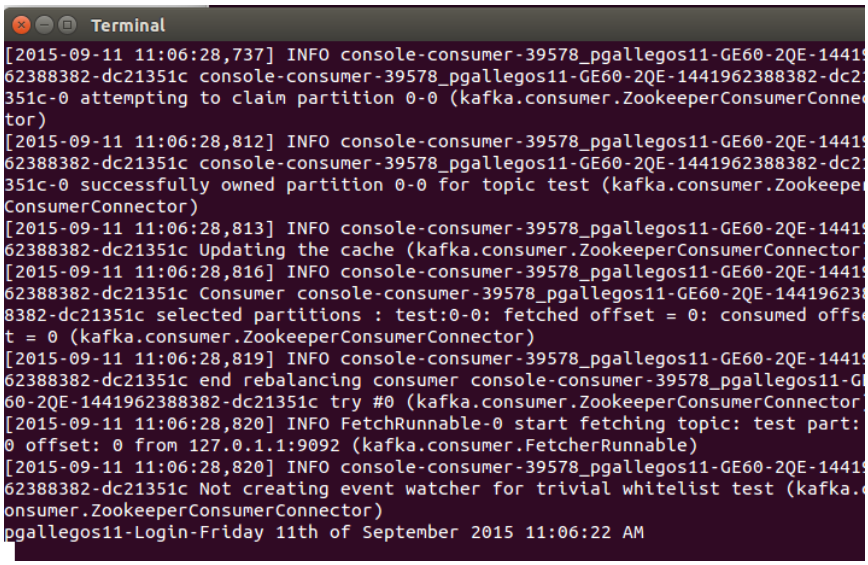

Ilustración 72: Kafka-Mensaje recibido de aplicación

CAPÍTULO 6 CONCLUSIONES

La realización de este proyecto ha supuesto un reto muy grande debido tanto al hecho de tener que combinar varias tecnologías como al hecho de tener que adaptar el trabajo realizado por otros alumnos, obligando previamente a un estudio en profundidad de dichos trabajos.

Este proyecto, como hemos dicho anteriormente, combina desarrollo de aplicaciones Android y Java, y el uso de muchas tecnologías diferentes como PHP, JSON o MySQL. Pero sin duda la tecnología más costosa de manejar ha sido Kafka. En primer lugar porque Kafka es un tecnología que aún está en desarrollo y además porque Kafka está escrito en Scala, un tipo de objeto no compatible con Android, lo cual ha obligado a usar un servidor Apache, con código PHP, como intermediario en la comunicación mediante la cola Kafka.

6.1 Objetivos logrados

Una vez terminado el proyecto, podemos ver con satisfacción que los objetivos que nos propusimos al comienzo del mismo se han cumplido:

- Herramienta Java sencilla e intuitiva desde la cual se puede:
  - Añadir, modificar y eliminar usuarios a la base de datos.
  - Gestionar las rutinas asignadas a cada usuario.
  - Añadir, modificar y eliminar rutinas a la base de datos.
  - Gestionar los ejercicios pertenecientes a cada rutina
  - Añadir, modificar y eliminar ejercicios a la base de datos.

- Recibir una actualización de los eventos que sucedan en cada usuario de la herramienta Android.

- Modificaciones en la herramienta Android:
  - Exportar la herramienta Android ya desarrollada de Ecplise a Android, dado que Eclipse deja de tener soporte para Android.
  - Botón de configuración para poder modificar la IP a la cual se conecta la herramienta desde el propio terminal.
  - Pequeñas modificaciones de estilo y código para permitir la comunicación perfecta con la base de datos y la herramienta Java.
  - Añadir clase para enviar eventos a la cola Kafka.

- Modificaciones en los archivos PHP:
  - Creación de archivo de configuración para modificación de parámetros de una manera más cómoda.
  - Modificación completa del encargado de mandar e-mail de recordatorio de contraseña.
  - Creación de archivo para enviar mensajes a la cola Kafka.

6.2 **Líneas de mejora**

En función de las necesidades que puedan ir surgiendo, hay elementos que pueden añadirse a la aplicación e incluso mejorar los ya instalados si así fuera preciso. Como ejemplo, podemos señalar algunas de ellas:

- Añadir otras opciones para la comunicación entre el monitor y el usuario. Actualmente la única forma que tiene para responder el entrenador es mediante e-mail o yendo a ver al usuario, lo cual impediría un entrenamiento a distancia.
- Añadir nuevos eventos a la aplicación Android.

- Mejorar la forma de crear y eliminar útiles, fotos y músculos en la herramienta Java.
- Mantener un historial de los usuarios en la base de datos.
- Añadir mayor contenido audiovisual al servidor.
- Desarrollar una versión del software de usuario en otros sistemas operativos como por ejemplo iOs.

BIBLIOGRAFÍA

1. Lampp. https://www.apachefriends.org/es/faq_linux.html. [En línea]
2. Lampp. Configurar. http://www.jonijnm.es/web/manual-joomla/218-18-instalar-y-configurar-lampp.html. [En línea]
3. MySQL. http://www.mysql.com/. [En línea]
4. Apache. Servidor. http://httpd.apache.org/. [En línea]
5. Java. Conector para MYSQL. http://dev.mysql.com/downloads/connector/j/. [En línea]
6. Java. Programar tareas. http://snippetea.blogspot.com.es/2012/10/timer-y-timertask-programar-tareas-twf.html. [En línea]
7. Java. Enviar email. http://www.chuidiang.com/java/herramientas/javamail/enviar-correo-javamail.php. [En línea]
8. Java. Enviar email. http://codehero.co/java-desde-cero-correos-electronicos/. [En línea]
9. Java. Introducción. http://www3.uji.es/~belfern/pdidoc/IX26/Documentos/introJava.pdf. [En línea]
10. Java. Interfaz grafica. https://www.fdi.ucm.es/profesor/jpavon/poo/Tema6resumido.pdf. [En línea]
11. Java. Interfaz grafica. http://codejavu.blogspot.com.es/2014/03/como-crear-interfaces-graficas-en.html. [En línea]
12. Java. Interfaz grafica. http://www.redribera.es/formacion/tutoriales/viewfile.html?file=javaintro0108-3.xml. [En línea]
13. PHP. www.php.net. [En línea]
14. Manual PHP. http://us3.php.net/manual/es/index.php. [En línea]
15. Json. http://www.json.org/. [En línea]
16. Google. https://developers.google.com/youtube/android/player/. [En línea]
17. Kafka. https://en.wikipedia.org/wiki/Apache_Kafka. [En línea]
18. Kafka. http://www.adictosaltrabajo.com/tutoriales/kafka-logs/. [En línea]
19. Kafka. http://kafka.apache.org/. [En línea]
20. Kafka. http://www.pc-100.com/es/article-apache-kafka-next-generation-

distributed-messaging-system/. [En línea]

21. Kafka. http://blog.cloudera.com/blog/2014/09/apache-kafka-for-beginners/. [En línea]
22. Kafka. http://wp-alvaromonsalve.rhcloud.com/2015/04/10/introduccion-a-apache-kafka/. [En línea]
23. Kafka. http://www.infoq.com/articles/apache-kafka. [En línea]
24. Kafka. http://blog.gfi.es/flume-kafka-spark-y-storm-un-nuevo-ejercito-apache/. [En línea]
25. Kafka. https://unpocodejava.wordpress.com/2012/12/21/un-poco-de-kafka/. [En línea]

ANEXO A: MANUAL DE INSTALACIÓN

Este manual de instalación ha sido probado en Ubuntu 14.04, no obstante debería ser compatible con cualquier sistema operativo en base Linux reciente y escalable a otros sistemas.

A.1 XAMPP

Recordemos que XAMPP es una distribución de Apache completamente gratuita y fácil de instalar que contiene **MySQL**, **PHP** y **Perl**. En pocas palabras Xampp es un servidor web que podremos instalar de forma local en nuestras máquinas, y con ello implementar un entorno de desarrollo para realizar nuestras pruebas antes de lanzarlo a un entorno de producción. XAMPP ofrece el ambiente ideal para el desarrollo de aplicaciones basadas en PHP.

PASO 1: Descarga de XAMPP

Aunque dentro de la documentación del proyecto se incluye el ejecutable para poder instalar XAMPP en Linux para el ordenador, se van a indicar los pasos por si el lector desease descargar el ejecutable para otro sistema operativo como Windows o Apple, o bien porque pasado un tiempo quizás la versión incluida en la documentación quedase obsoleta y en la URL que se indica a continuación se puede encontrar la versión más reciente.

Así pues, desde la siguiente URL [https://www.apachefriends.org/index.html], se puede encontrar la versión más reciente y para todos los sistemas operativos:

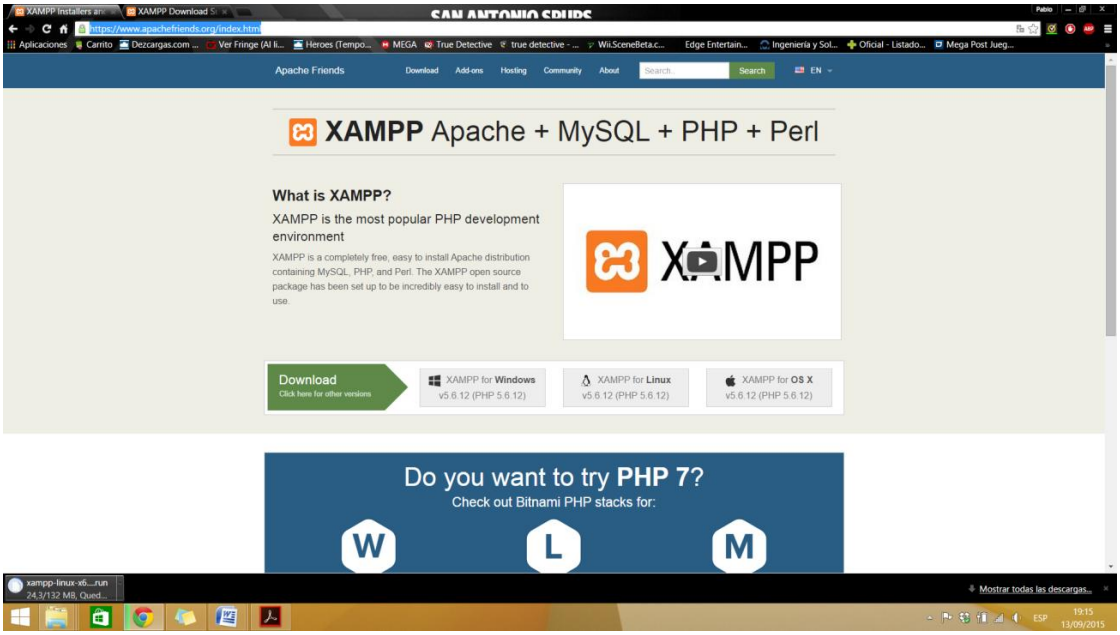

Ilustración 73: Pagina descarga XAMPP para S.O. deseado

Al pulsar sobre la opción deseada, nos saldrá la siguiente página:

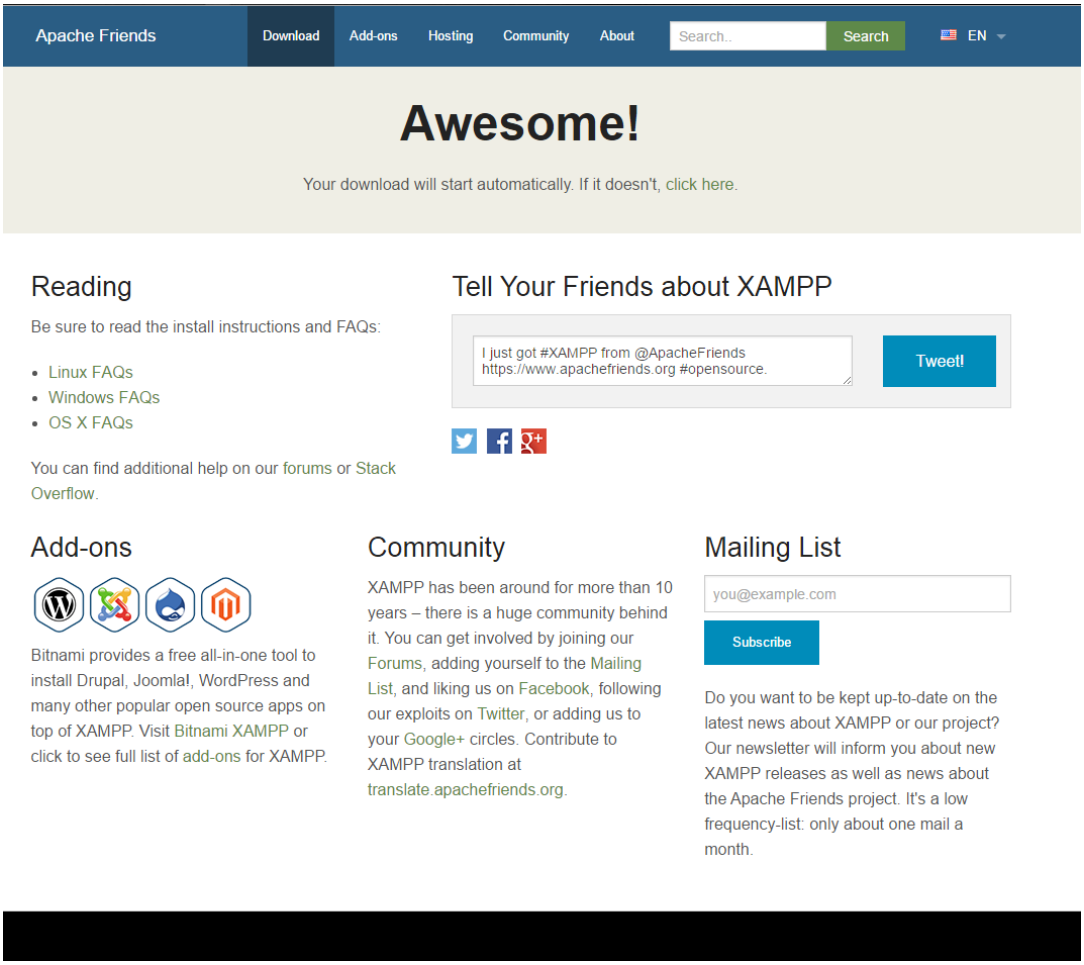

Ilustración 74: Inicio descarga de XAMPP

Inmediatamente comenzará la descarga del ejecutable de XAMPP; si esto no ocurriese, pulsar sobre “click here” de la página anterior.

Una vez descargado el ejecutable, buscarlo en la carpeta de descargas.

PASO 2: Instalación de XAMPP

Al ejecutarlo mediante la siguiente orden desde el terminal

```
./ xampp-linux-x64-5.6.12-0-installer.run
```

Nos aparecerá la siguiente pantalla, en la cual únicamente debemos pulsar el botón next:

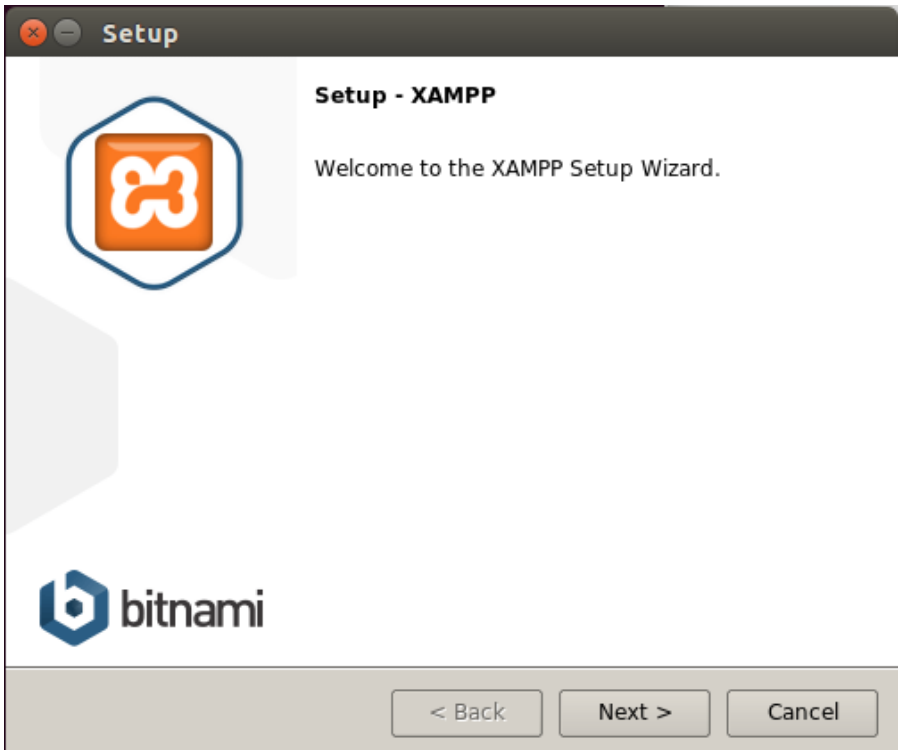

Ilustración 75: Instalación XAMPP-1

Antes de pulsar next en la siguiente pantalla debemos asegurarnos de que los dos ticks están marcados en XAMPP Core Files y XAMPP Developer Files:

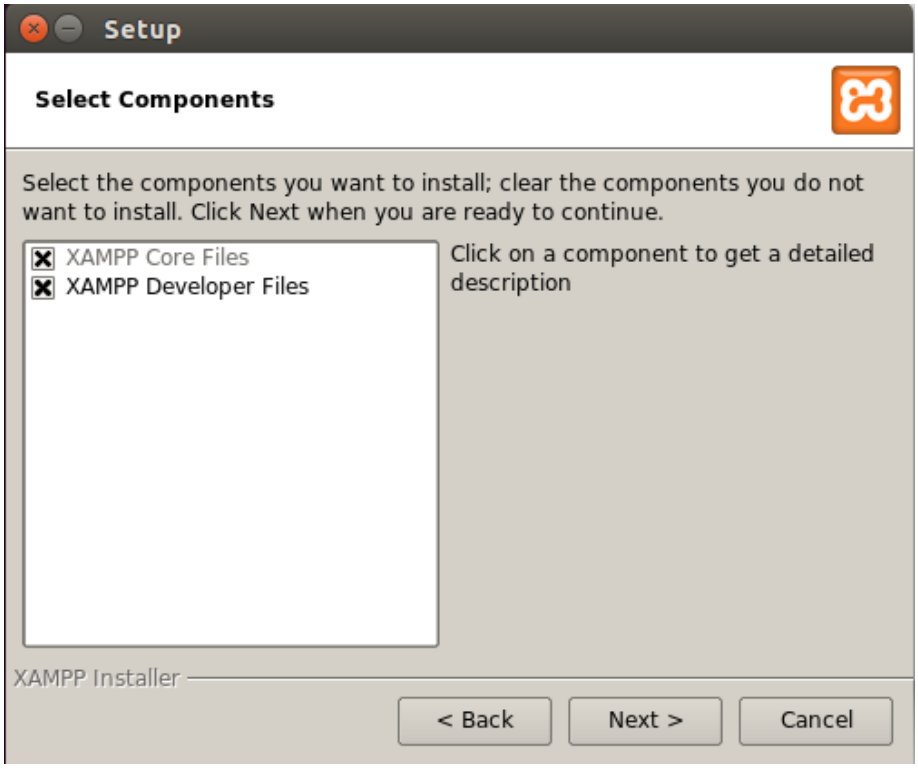

Ilustración 76: Instalación XAMPP-2

En la siguiente pantalla se nos muestra donde se instalara XAMPP lo cual nos resultara muy útil posteriormente si queremos modificar alguna configuración o añadir nuevos archivos al servidor.

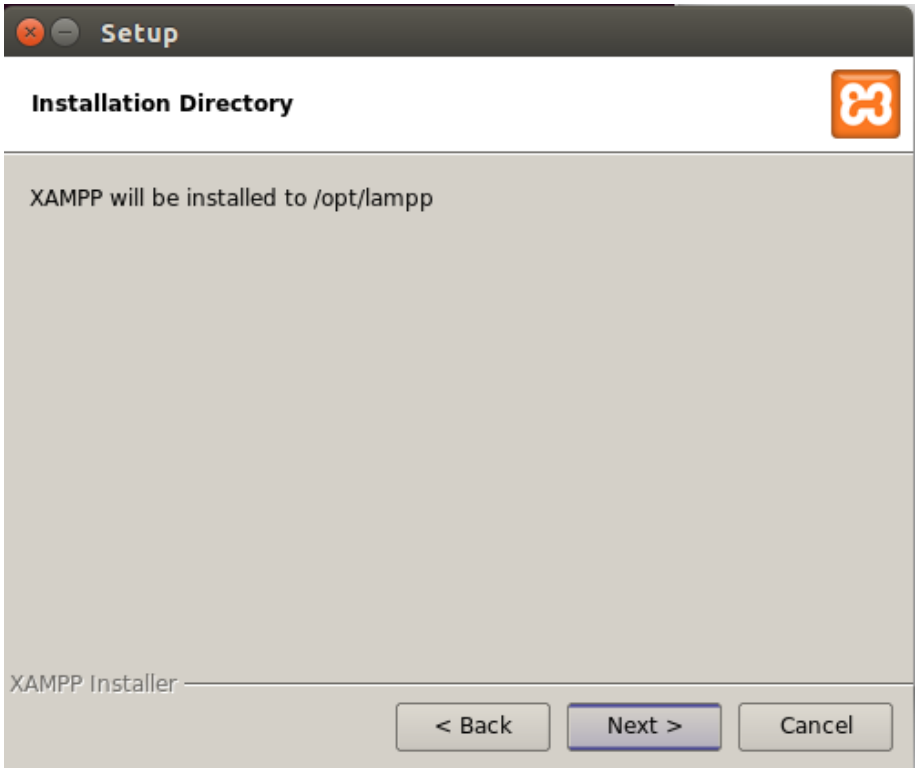

Ilustración 77: Instalación XAMPP-3

En la siguiente pantalla se nos da la posibilidad de aprender más sobre Bitnami para XAMPP esto puede ser muy útil si queremos montar un servidor en condiciones, ya que te permite añadir diferentes funcionalidad, no obstante para el caso que nos atañe recomiendo desmarcar esa opción antes de pulsar Next

Ilustración 78: Instalación XAMPP-4

Una vez pulsado next nos permitirá por última vez volver hacia atrás, en caso que todo vaya según lo previsto pulsamos Next y comenzamos con la instalación.

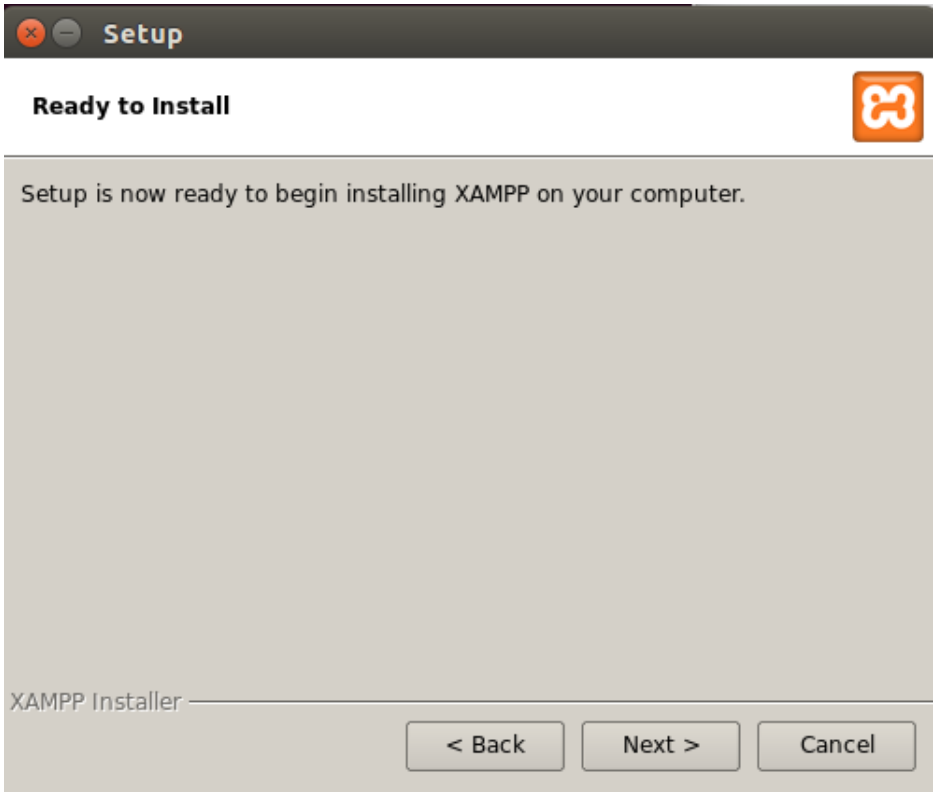

Ilustración 79: Instalación XAMPP-5

Esta instalación es bastante rápida, en un ordenador medio no tardará más de dos minutos.

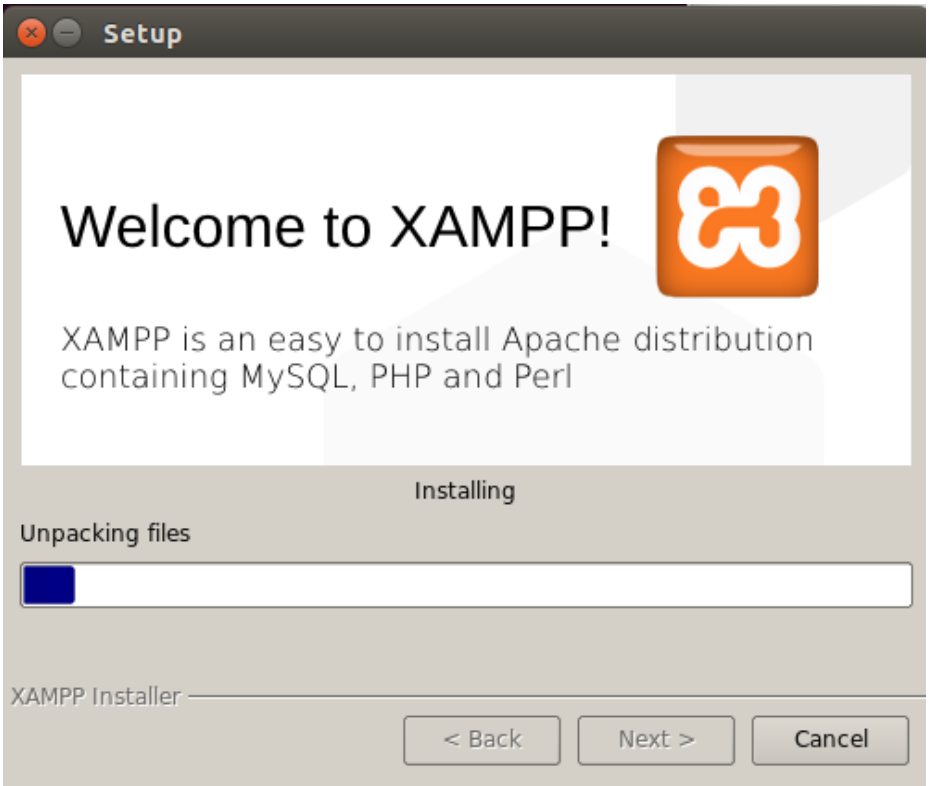

Ilustración 80: Instalación XAMPP-6

Una vez esté instalado se nos permitirá arrancar XAMPP

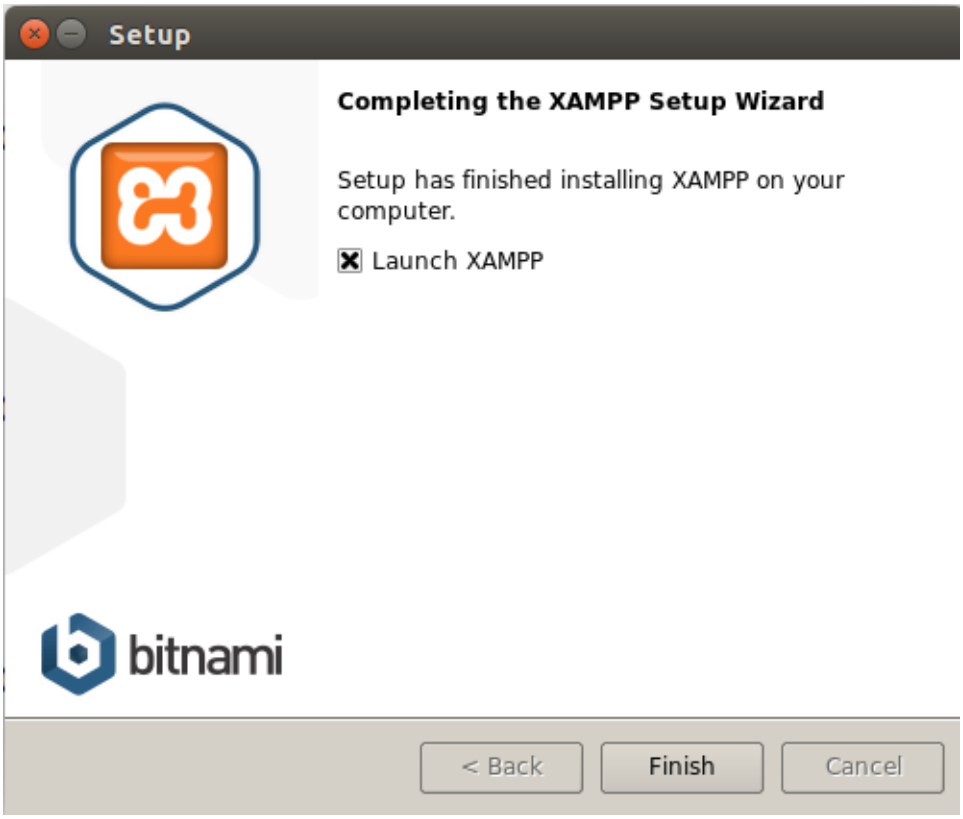

Ilustración 81: Instalación XAMPP-7

PASO 3: Arranque de XAMPP

Una vez XAMPP esté arrancado nos mostrará esta venta:

Ilustración 81: Arrancando XAMPP

Si pulsamos en la pestaña de en medio "Manage Servers" nos mostrará esta pantalla:

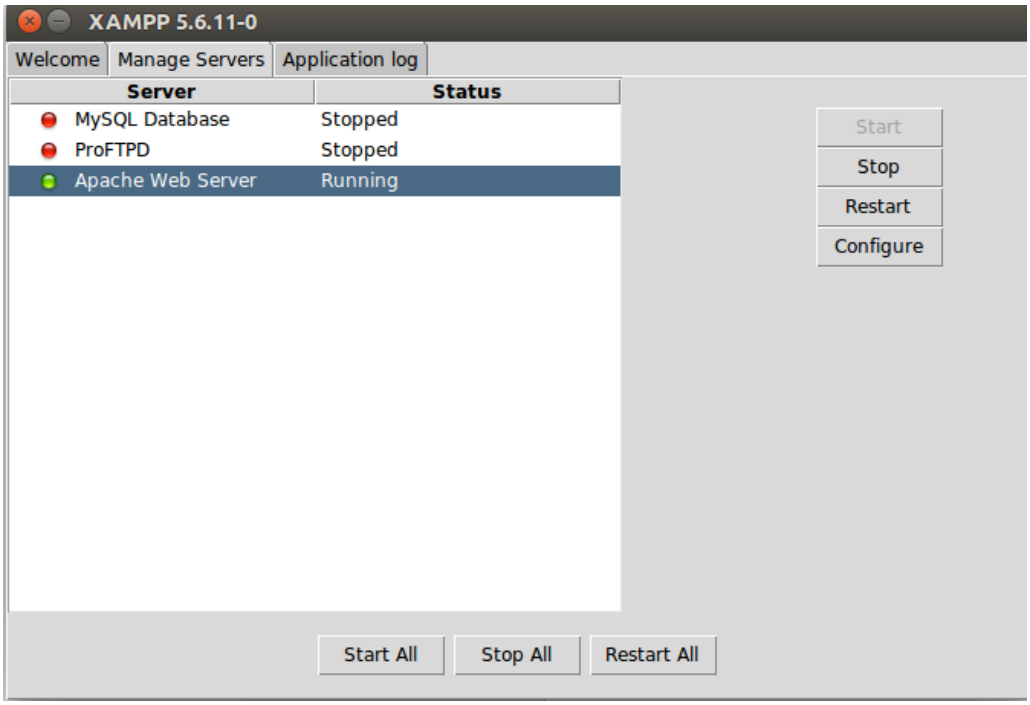

Ilustración 83: Servidores XAMPP

Simplemente pulsamos en el botón inferior "Start All" y ya tenemos nuestro servidor corriendo correctamente.

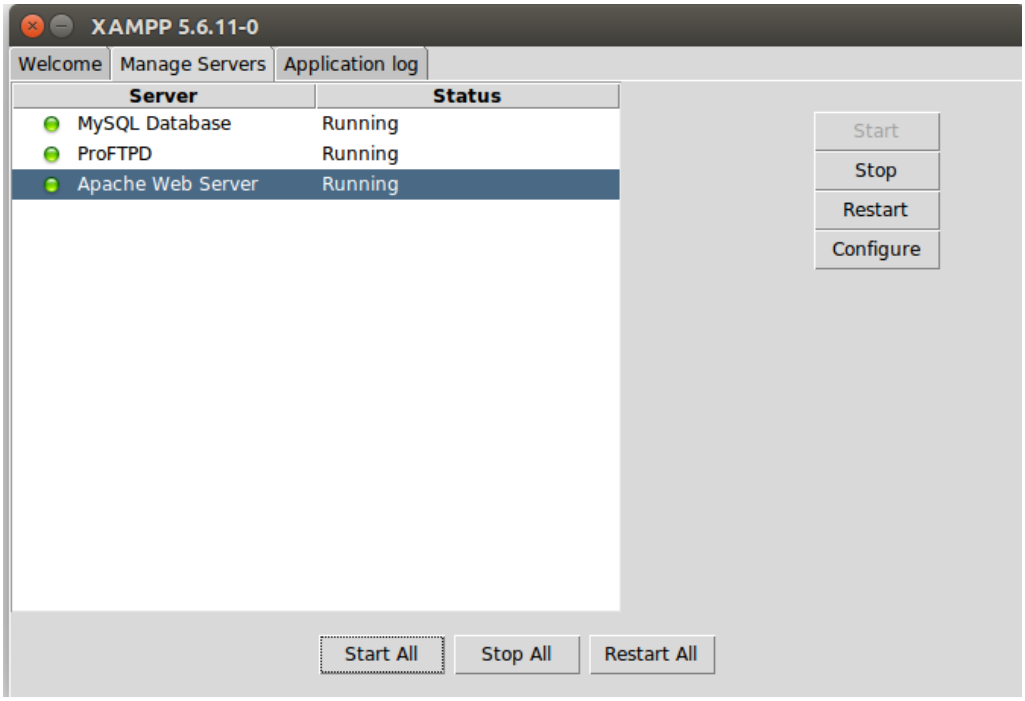

Ilustración 84: Servidores XAMPP instalados

PASO 4: Configurando XAMPP y añadiendo proyecto

Para configurar XAMPP lo primero que tenemos que hacer es añadir el contenido a nuestro servidor, tanto los archivos PHP como la información a la base de datos.

Para ello lo más sencillo es:

Borrar todos los archivos actuales ejecutando lo siguiente en el terminal:

```
> sudo su

> cd /opt/lampp/htdocs

> rm -Rf *
```

Colocarnos en la carpeta donde se encuentre htdocs con todo el código PHP del proyecto y ejecutar lo siguiente en el terminal:

```
> cp -r htdocs/* /opt/lampp/htdocs/

> chmod 777 /opt/lampp/htdocs/*

> chmod 777 /opt/lampp/htdocs/*/*

> chmod 777 /opt/lampp/htdocs/*/*/*

> chmod 777 /opt/lampp/htdocs/*/*/*/*
```

Los últimos cambios de permiso son necesario para la poder ejecutar la cola Kafka.

Con esto tendríamos configurada los archivos PHP, no obstante faltaría la base de datos, para ello lo más sencillo es:

Entrar en el navegador en la dirección: http://localhost/phpmyadmin/ tras lo cual nos aparecerá esta pantalla:

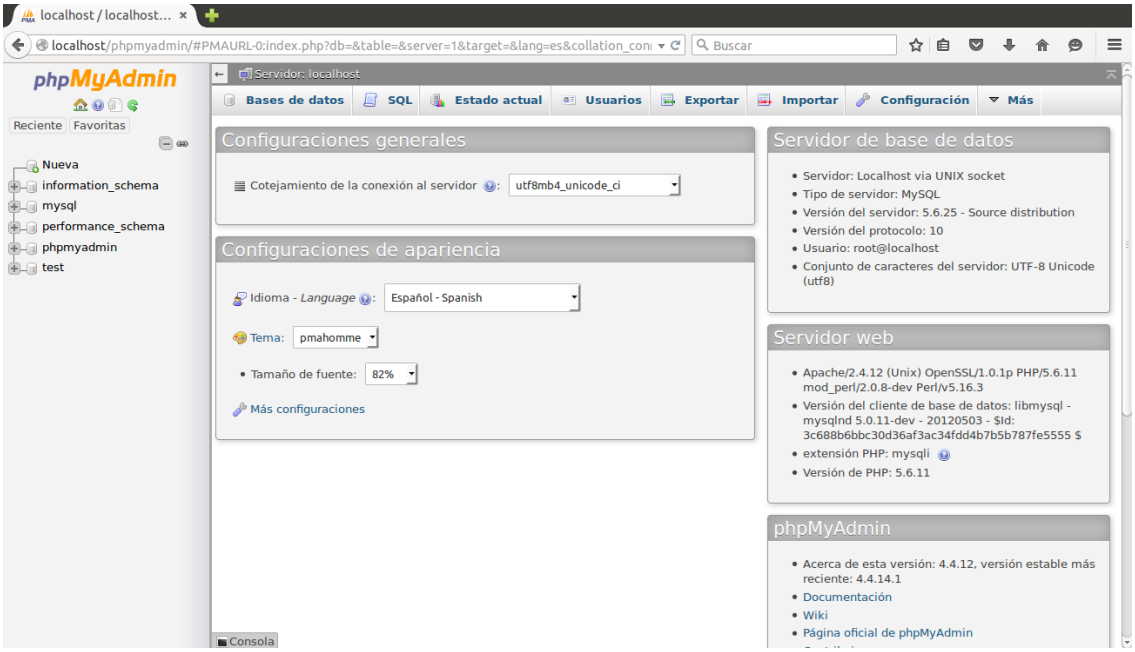

Ilustración 85: PhpMyAdmin

Pulsamos el botón importar y nos aparecerá la siguiente pantalla, en la cual únicamente debemos darle a Examinar y buscar en nuestro ordenador donde está el archivo con la información de base de datos y pulsamos en continuar.

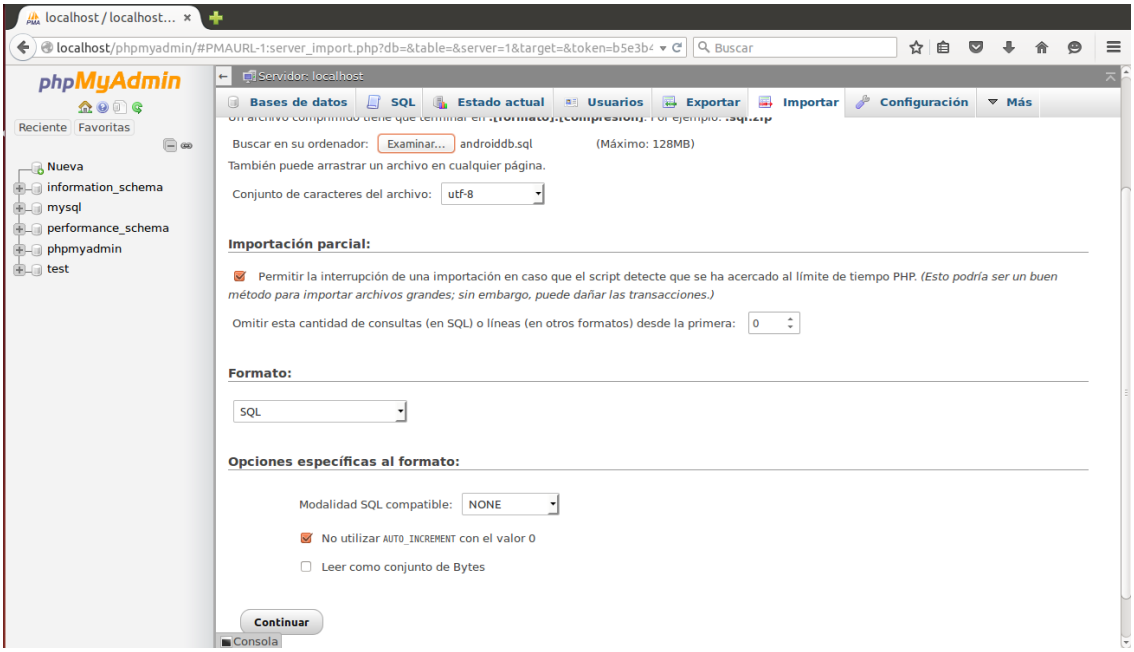

Ilustración 86: PhpMyAdmin importado

Tras un par de minutos la base de datos estará completamente cargada.

PASO 5: Configurando servidor de correos

Para terminar con XAMPP, si queremos que se pueda recuperar la contraseña mediante email, debemos configurar el servidor de correos. Para ello abrimos un terminal y ejecutamos lo siguiente:

> sudo apt-get install ssmtp

> sudo apt-get install mailutils

Posiblemente en alguna instalación se nos pregunte si debemos continuar, únicamente escribir S

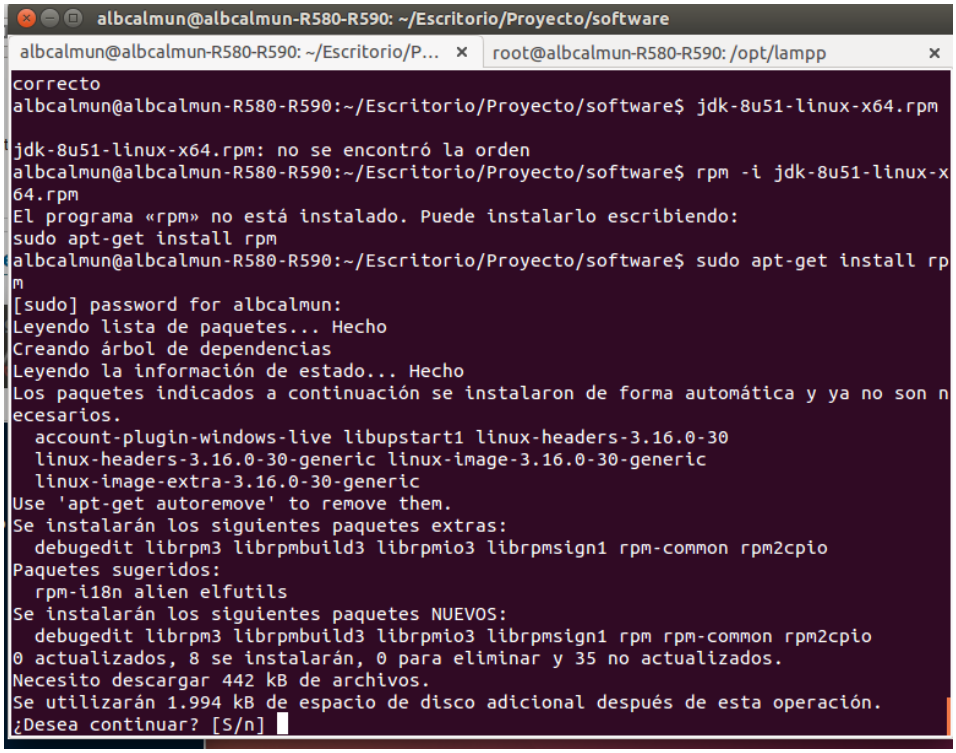

Ilustración 87: Terminal Linux

> gedit /opt/lampp/etc/php.ini

Este archivo debemos de dejarlo con la siguiente configuración en la zona donde pone Mail function:

```
[mail function]
; For Win32 only.
; http://php.net/smtp
SMTP=localhost
; http://php.net/smtp-port
smtp_port=25

; For Win32 only.
; http://php.net/sendmail-from
;sendmail_from = me@example.com

; For Unix only.  You may supply arguments as well (default:
"sendmail -t -i").
; http://php.net/sendmail-path
sendmail_path =/usr/sbin/ssmtp -t

; Force the addition of the specified parameters to be passed as
extra parameters
; to the sendmail binary. These parameters will always replace the
value of
```

Ilustración 88: PHP.ini

> gedit /etc/ssmtp/ssmtp.conf

Este archivo debemos de dejarlo con la siguiente configuración

```
#
# Config file for sSMTP sendmail
#
# The person who gets all mail for userids < 1000
# Make this empty to disable rewriting.
root=droidlogingestor@gmail.com

# The place where the mail goes. The actual machine name is required no
# MX records are consulted. Commonly mailhosts are named mail.domain.com
mailhub=smtp.gmail.com:587

# Where will the mail seem to come from?
#rewriteDomain=

# The full hostname
hostname=droidlogingestor@gmail.com

# Are users allowed to set their own From: address?
# YES - Allow the user to specify their own From: address
# NO - Use the system generated From: address
FromLineOverride=YES
UseTLS=YES
UseSTARTTLS=YES
AuthUser=droidlogingestor@gmail.com
AuthPass=Pruebafinal1!
```

Ilustración 89: Ssmtp.conf

> gedit /etc/ssmtp/revaliases

Por último este archivo debe tener la siguiente configuración, cambiando albcalmun, por el nombre de usuario:

```
# sSMTP aliases
#
# Format:       local_account:outgoing_address:mailhub
#
# Example: root:your_login@your.domain:mailhub.your.domain[:port]
# where [:port] is an optional port number that defaults to 25.
root:droidlogingestor@gmail.com:smtp.gmail.com:587
albcalmun:droidlogingestor@gmail.com:smtp.gmail.com:587
```

Ilustración 90: Revaliases

> cd /opt/lampp

> ./lampp restart

Ya tenemos configurado nuestro servidor.

A.2 Eclipse

Eclipse es el IDE que se ha usado para modificar la herramienta java del monitor por lo que recomendamos su instalación si se desea realizar algún cambio.

PASO 1: Descarga de Eclipse

Aunque dentro de la documentación del proyecto se incluye el ejecutable para poder instalar Eclipse en Linux para el ordenador, se van a indicar los pasos por si el lector desease descargar el ejecutable para otro sistema operativo como Windows o Apple, o bien porque pasado un tiempo quizás la versión incluida en la documentación quedase obsoleta y en la URL que se indica a continuación se puede encontrar la versión más reciente.

Así pues, desde la siguiente URL [https://eclipse.org/downloads/]se puede encontrar la versión más reciente y para todos los sistemas operativos, se recomienda la instalación de Eclipse IDE for Java Developers.

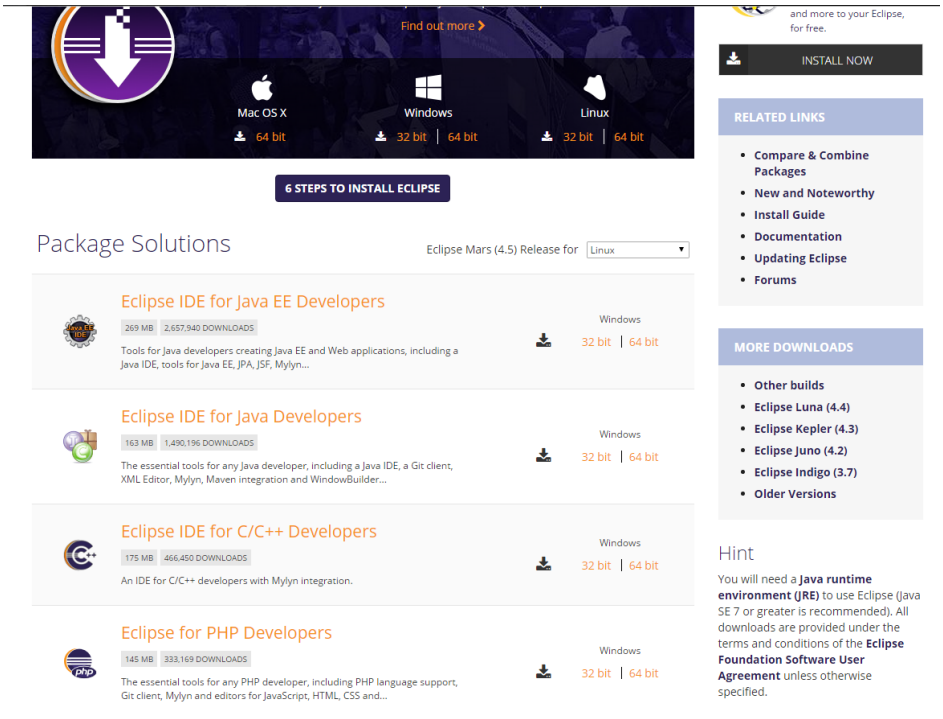

Ilustración 91: Página descarga Eclipse para S.O. deseado

PASO 2: Instalando Eclipse

La instalación de eclipse es muy sencilla únicamente debemos de descomprimir el archivo descargado y desde el terminal colocándonos en su carpeta ejecutar:

```
> chmod 777 ./eclipse
```

> ./eclipse

Con esto ya tendríamos eclipse arrancado.

PASO 3: Añadiendo proyecto

Para importar el código es muy sencillo únicamente debemos pulsar en File->Import->General->Existing Projects into Workspace y buscar el proyecto en la opción Browse:

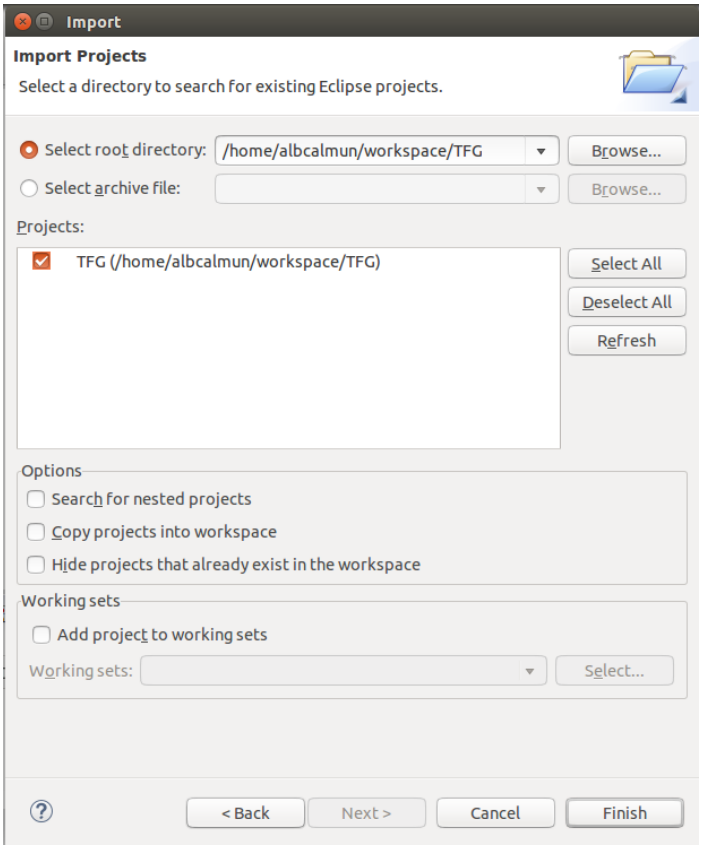

Ilustración 92: Importando proyecto Eclipse

A.3 Android Studio

Android Studio es el IDE que se ha usado para crear la herramienta android del usuario por lo que recomendamos su instalación si se desea realizar algún cambio.

PASO 1: Descarga de Android Studio

Aunque dentro de la documentación del proyecto se incluye el ejecutable para poder instalar Android Studio en Linux para el ordenador, se van a indicar los pasos por si el lector desease descargar el ejecutable para otro sistema operativo como Windows o Apple, o bien porque pasado un tiempo quizás la versión incluida en la documentación quedase obsoleta y en la URL que se indica a continuación se puede encontrar la versión más reciente.

Así pues, desde la siguiente URL [https://developer.android.com/sdk/index.html#Other] se puede encontrar la versión más reciente y para todos los sistemas operativos, se debe descargar tanto el android-sdk como el android-studio-ide:

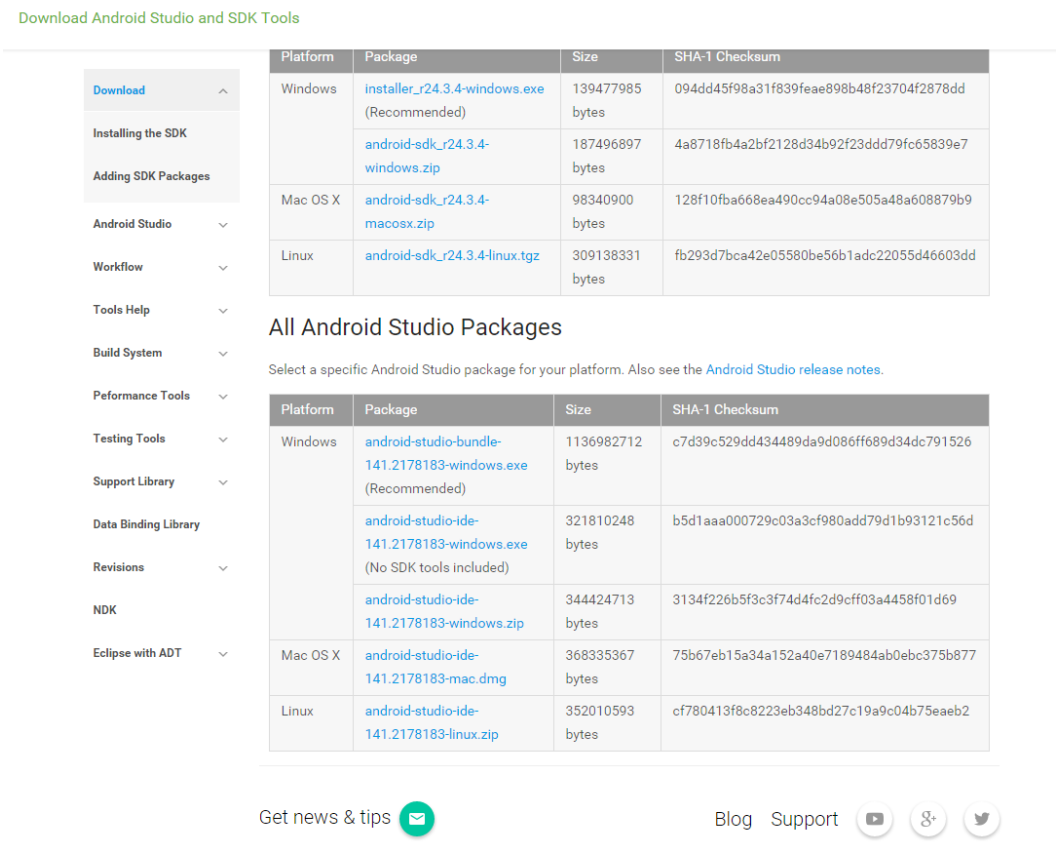

Ilustración 93: Página descarga Android-Studio para S.O. deseado

PASO 2: Instalando Android Studio

Tras descomprimir los dos archivos descargados, entremos en la carpeta android-studio y desde el terminal ejecutamos:

./studio.sh

Tras ejecutarlo nos aparecerá esta ventana en la cual debemos seleccionar la segunda opción si nunca hemos tenido instalado Android Studio:

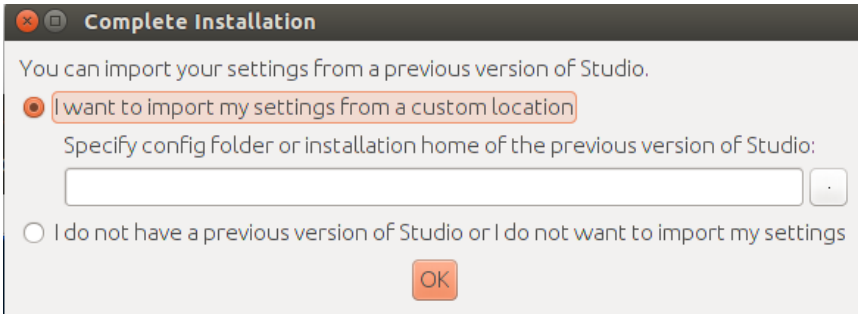

Ilustración 94: Importar configuración Android-Studio

Tras hacer nuestra elección nos aparecerá una ventana de bienvenida en la que debemos pulsar next:

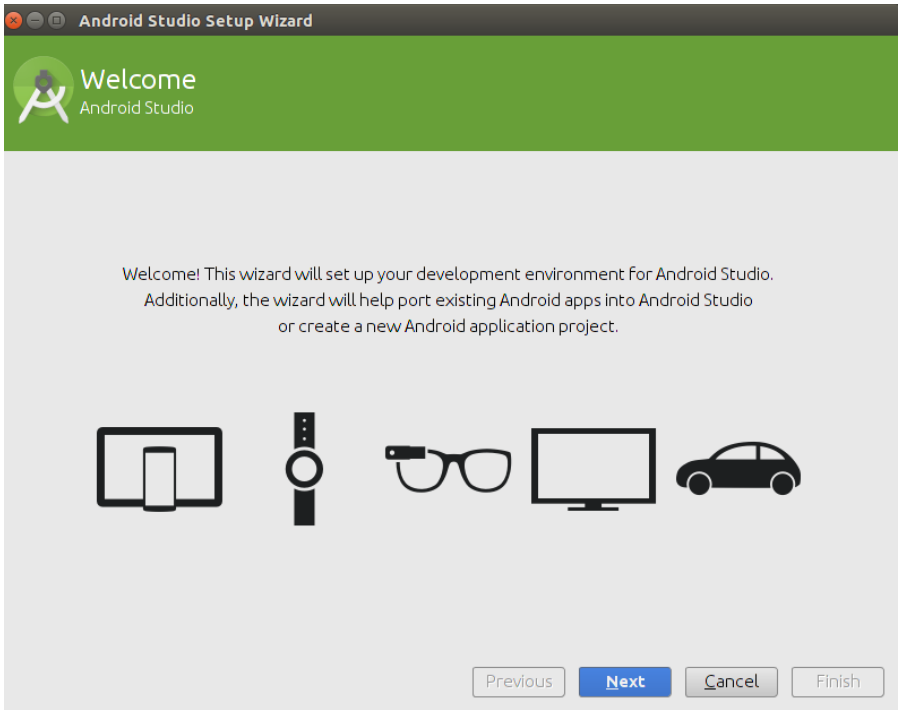

Ilustración 95: Instalación Android-Estudio-1

En la siguiente ventana nos permite hacer una instalación estándar o elegir que elementos deseamos instalar, para personas no expertas en la materia, recomiendo la instalación por defecto:

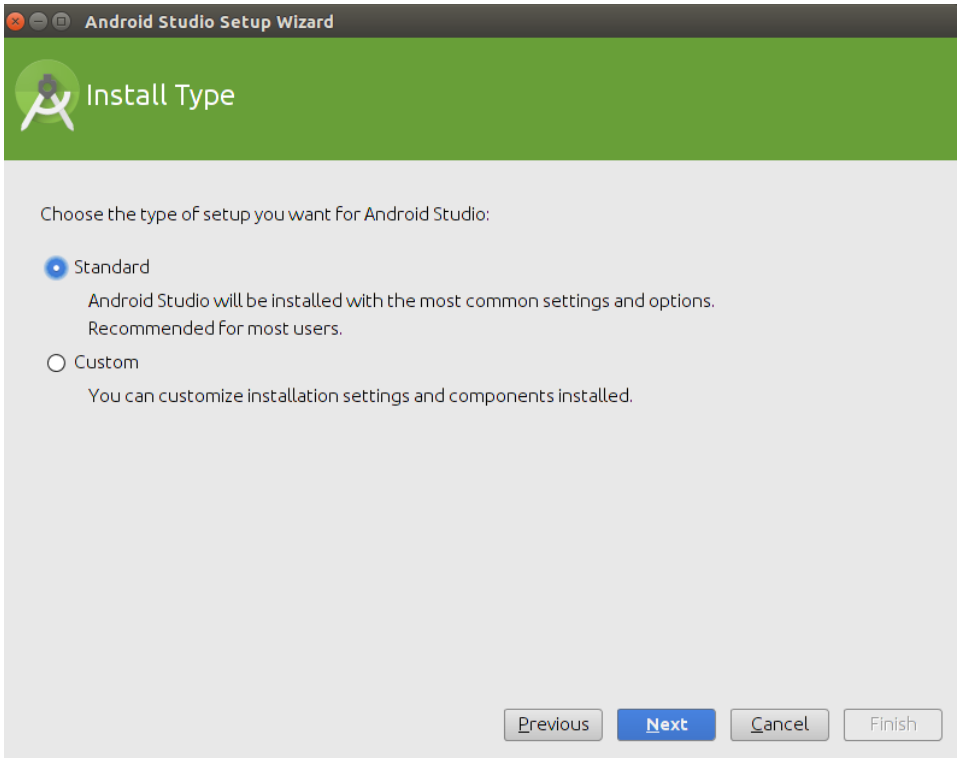

Ilustración 96: Instalación Android-Estudio-2

En la siguiente ventana nos habla de virtualización que permite que Android Studio pueda ir a una mayor velocidad, al igual que anteriormente, para personas no expertas recomendamos obviar este punto y pulsar Next

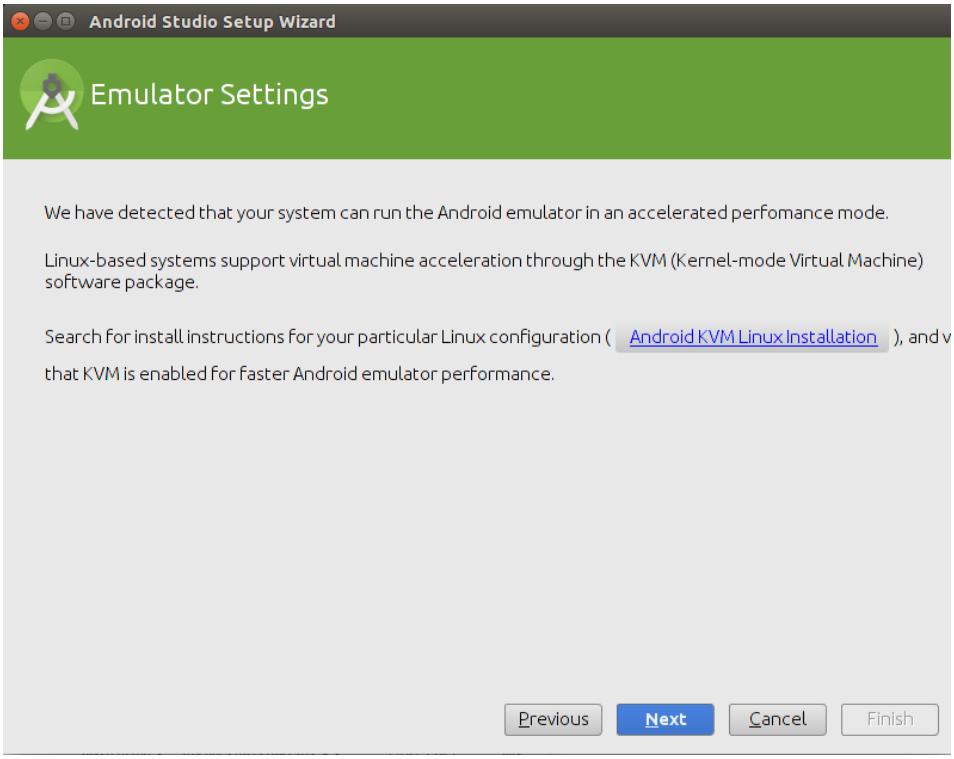

Ilustración 97: Instalación Android-Estudio-3

Para terminar se nos mostrara una ventana indicándonos el espacio requerido y los componentes que se van a instalar, tras pulsar next comenzara la instalación:

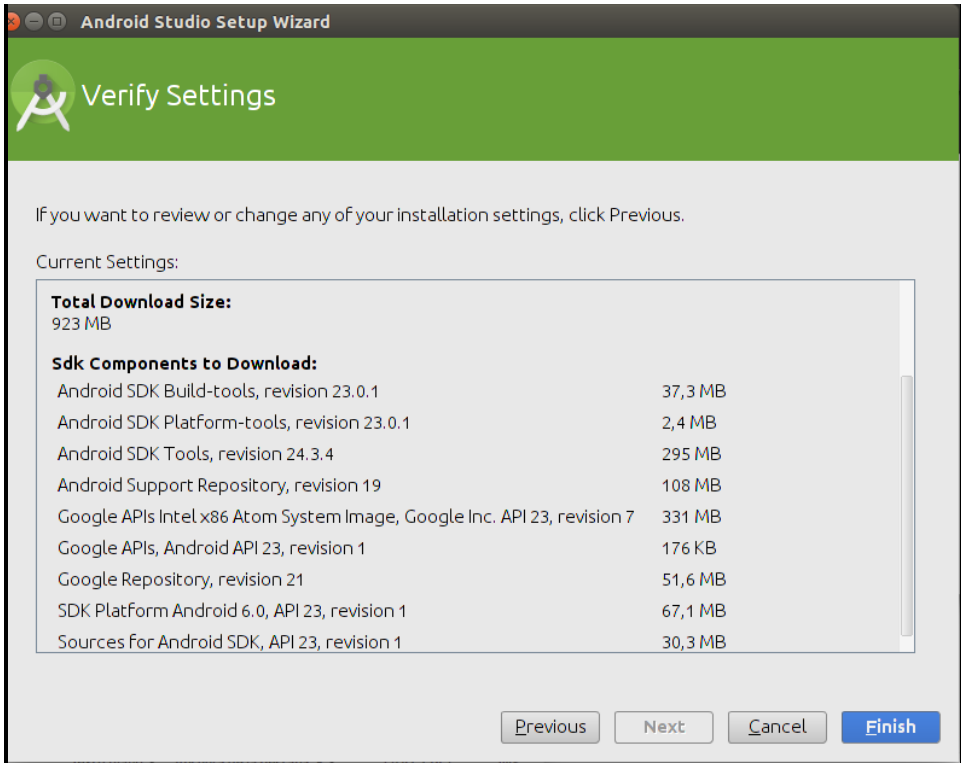

Ilustración 98: Instalación Android-Estudio-4

Cuando la instalación haya finalizado, pulsamos finish

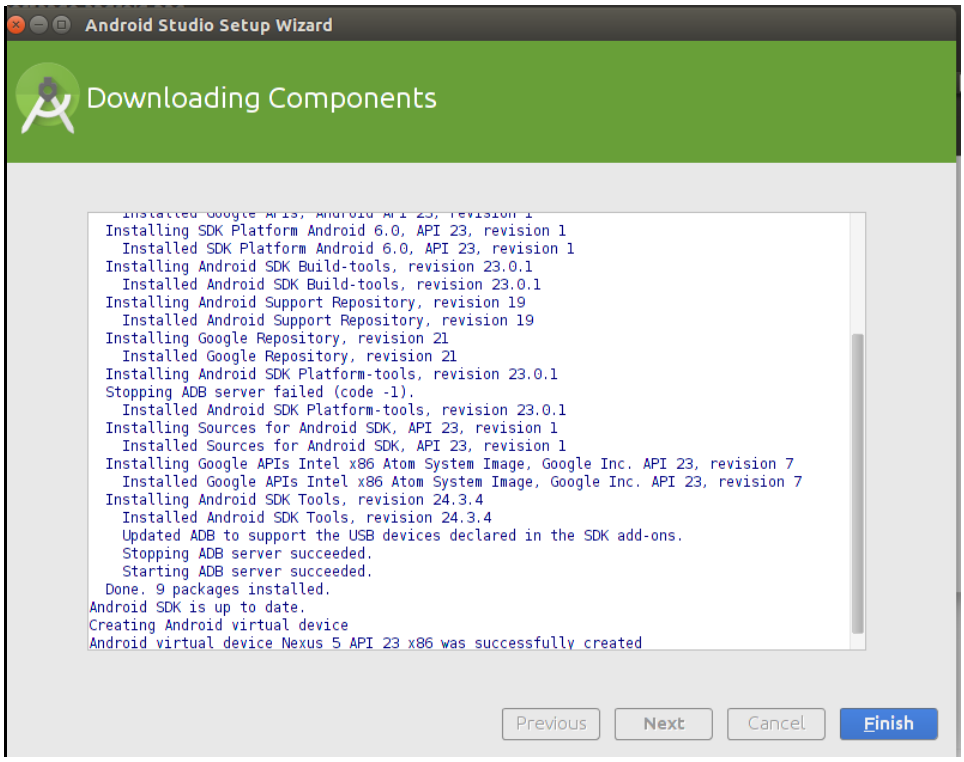

Ilustración 99: Instalación Android-Estudio-5

PASO 3: Importando proyecto Android Studio

Para importar un proyecto únicamente debemos seleccionar la opción marcada en la siguiente imagen:

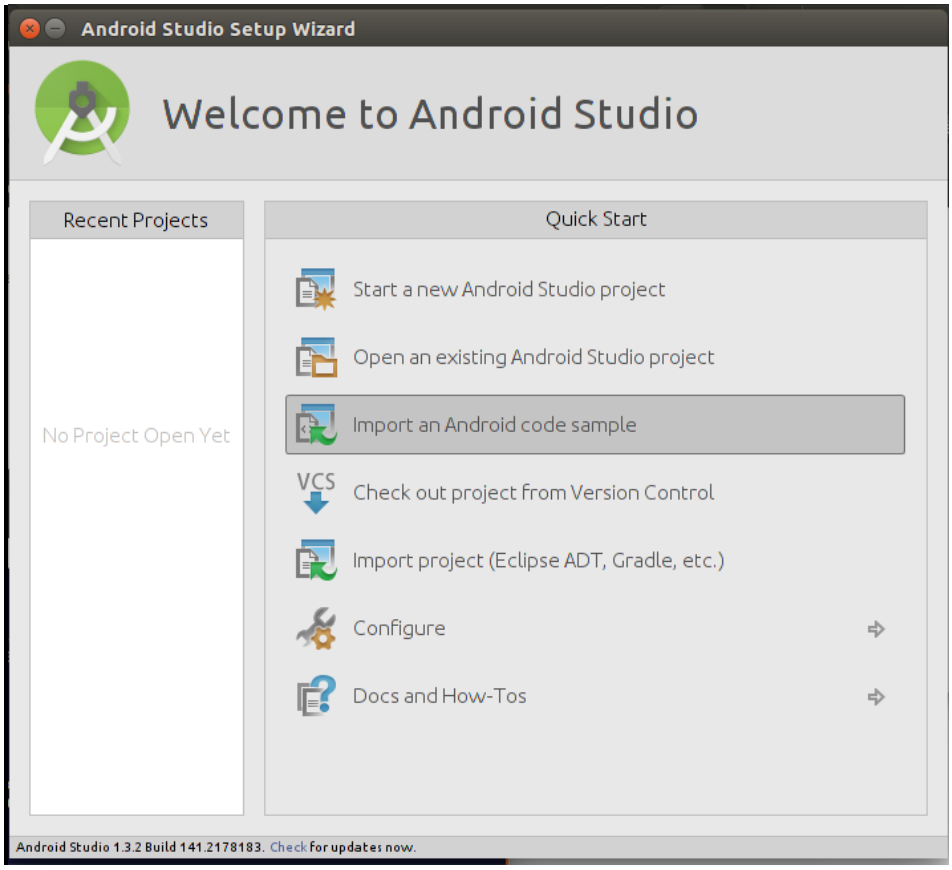

Ilustración 100: Android-Estudio

Tras esto nos cargara Android Studio con el proyecto importado.

PASO 4: Faltan algunos paquetes

Es posible que al importar el proyecto nos aparezca una ventana con el siguiente mensaje en su parte inferior:

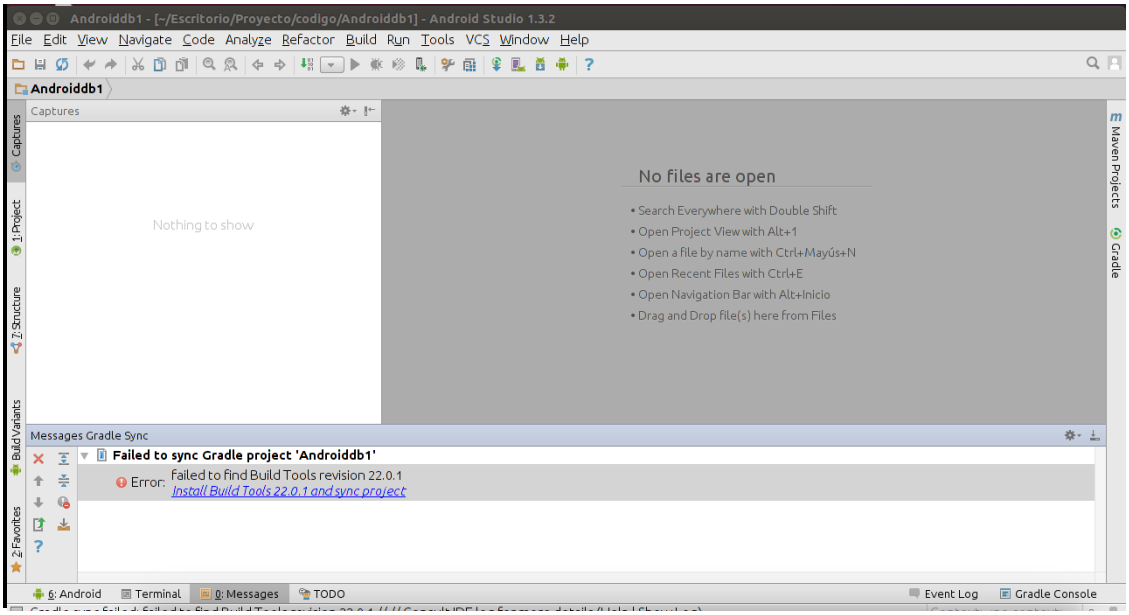

Ilustración 101: Problemas Android-Estudio

Si esto ocurriese únicamente debemos hacer click en el enlace en azul para instalar el paquete que falta, cuando esté instalado nos mostrara la siguiente ventana:

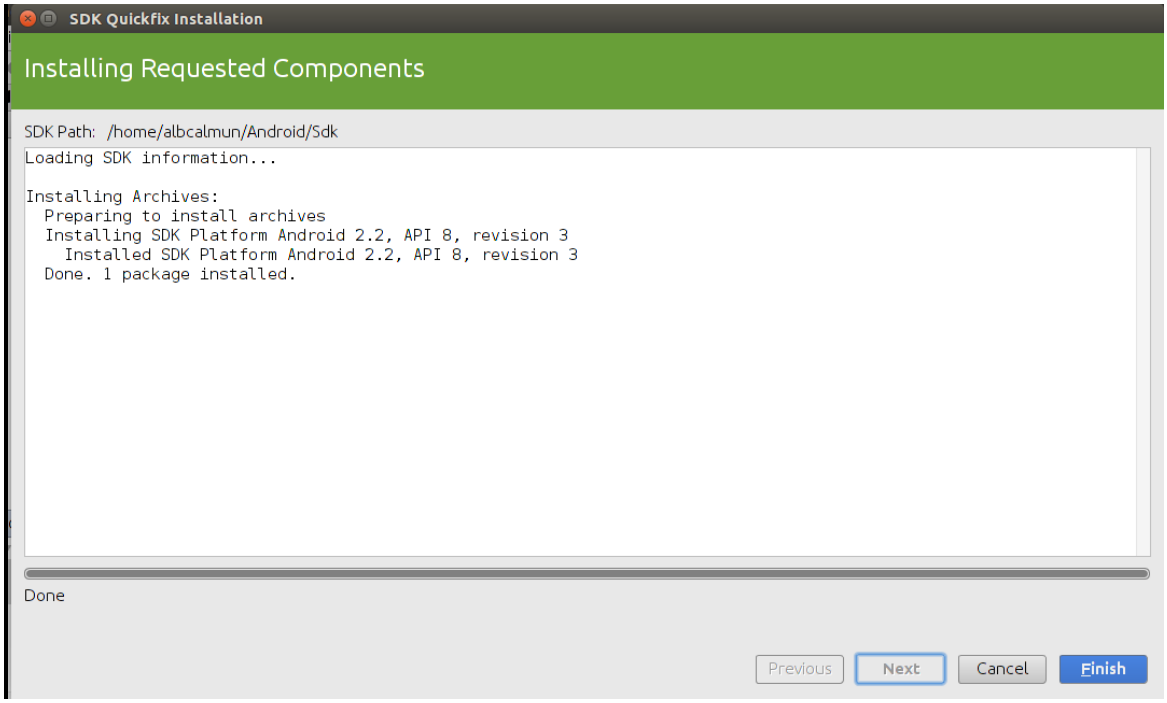

Ilustración 102: Solución Android-Estudio

Si el mensaje sigue apareciendo repetir el proceso hasta que todos los paquetes faltantes se instalen.

A.4 Kafka

Kafka es la herramienta utilizada para la cola de mensajes distribuidos.

PASO 1: Descarga Kafka

Aunque dentro de la documentación del proyecto se incluye el archivo necesario para poder instalar Kafka en Linux para el ordenador, se van a indicar los pasos por si el lector desease descargarlo directamente.

Así pues, desde la siguiente URL [http://kafka.apache.org/downloads.html] se puede encontrar la versión más reciente no obstante se recomienda la instalación de la versión 0.7.2 dado que es con la que se ha realizado el proyecto:

Tras descargar el archivo ejecutamos lo siguiente en el terminal

tar xzf kafka-<VERSION>.tgz

> cd kafka-<VERSION>

> ./sbt

Nos aparecerá una ventana parecida a ésta en la que debemos ejecutar:

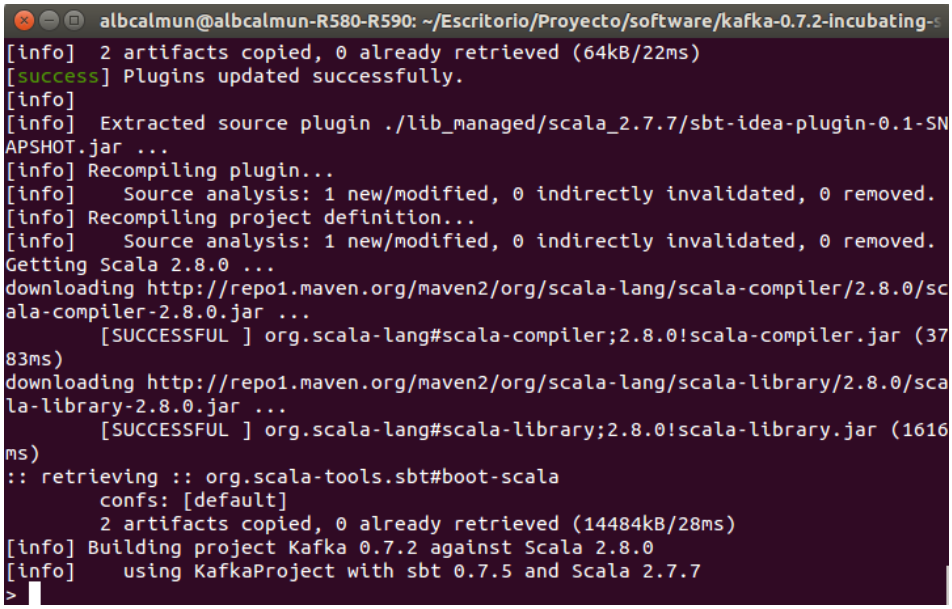

Ilustración 103: Sbt Kafka

> update

> package

> exit

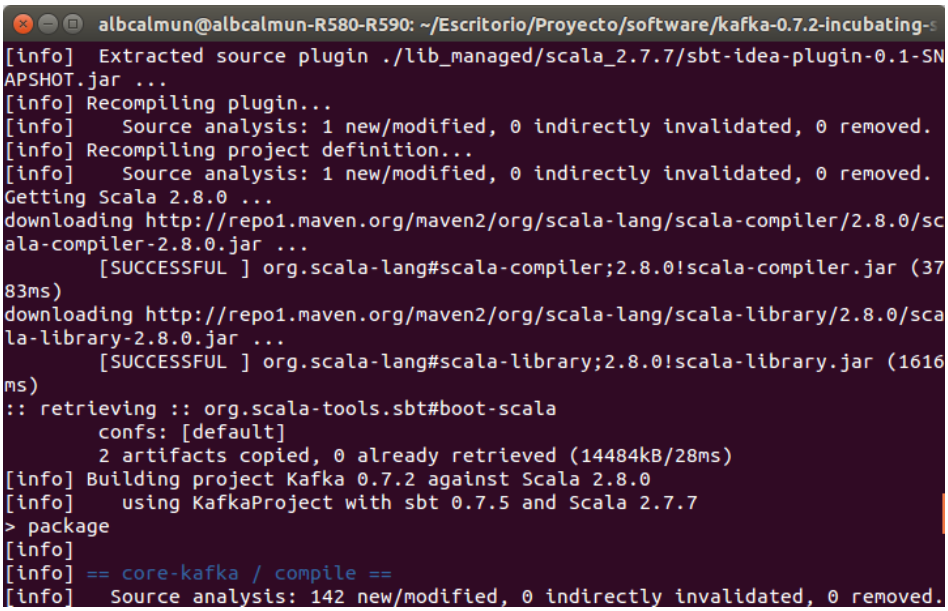

Ilustración 104: Sbt completado

Con esto ya estaría instalada la cola Kafka.

A.5 JDK y JRE

En algún punto de la instalación puede indicarnos que no tenemos instalado JDK o JRE para ello se recomienda ejecutar en el terminal:

```
> sudo apt-get install default-jre
```

```
> sudo apt-get install default-jdk
```

ANEXO B: CÓDIGOS DEL PROYECTO

En este anexo vamos a detallar completamente el código de la herramienta del entrenador y alguna de las partes más importantes del código del servidor y de la herramienta android.

B.1 Código java

B.1.1 Presentacion.java

```
package Presentacion;

import java.awt.BorderLayout;
import java.awt.Component;
import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JOptionPane;
import javax.swing.UIManager;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import java.awt.Rectangle;
import javax.swing.JTextField;
import java.awt.Font;
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.border.MatteBorder;
import javax.swing.JPasswordField;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.*;
import javax.mail.*;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
/**
 * @author pgallegos11
 *
 */
class Presentacion extends JFrame {
	/**
	 *
	 */
```

```
	private static final long serialVersionUID = 1L;
	private JPanel PanelPresentacion;
	public JTextField CampoUsuario;
	public JPasswordField CampoContraseña;
	public static Presentacion frame;
	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					frame = new Presentacion();
					frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}
	/**
	 * Create the frame.
	 */
	public Presentacion() {
		super("Gestor de Entrenamientos");
this.setResizable(false);
		setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
		setBackground(new Color(51, 102, 153));
		setBounds(new Rectangle(650, 275, 600, 600));
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		PanelPresentacion = new JPanel();
		PanelPresentacion.setBackground(new Color(51, 102, 153));
		PanelPresentacion.setBorder(new EmptyBorder(5, 5, 5, 5));
		PanelPresentacion.setLayout(new BorderLayout(0, 0));
		setContentPane(PanelPresentacion);

		JPanel panel = new JPanel();
		panel.setBackground(new Color(51, 102, 153));
		PanelPresentacion.add(panel, BorderLayout.CENTER);
		panel.setLayout(null);

		JLabel lblNewLabel = new JLabel("¿Olvido su contraseña?");
		lblNewLabel.addMouseListener(new MouseAdapter() {
			@Override
			public void mouseClicked(MouseEvent e) {
	//		JOptionPane.showInputDialog(parentComponent, message)
			String usuario=JOptionPane.showInputDialog(null,"Introduzca su
usuario","",JOptionPane.QUESTION_MESSAGE );
			String email;
				String para;
				String de;
				// String servidor;

				Properties propiedades;
				Session session;
				Connection con;
				ResultSet rs;
```

```
                        String password;
                        boolean correcto=false;
                if(usuario!=null)
                {
                        if(usuario.equals(""))
                                JOptionPane.showMessageDialog(null,"Debe introducir su nombre de
usuario","",JOptionPane.WARNING_MESSAGE);
                        else
                                {
                                email=JOptionPane.showInputDialog(null,"Introduzca su
email","",JOptionPane.QUESTION_MESSAGE);
                                if(email!=null)
                                        {
                                        if(email.equals(""))
                                                JOptionPane.showMessageDialog(null,"Debe
introducir su email","",JOptionPane.WARNING_MESSAGE);
                                        else
                                                {
                                                try {

                                                    Class.forName("com.mysql.jdbc.Driver");
                                                    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb",
                                                    "root", "");

                                                     // Creamos un Statement para poder hacer
peticiones a la bd
                                                     Statement stat = con.createStatement();

                                                    // Seleccionar todos los datos
                                                    String seleccionar = "SELECT * FROM
usuarios WHERE username='"+usuario+"' and direccion='"+email+"' ";

                                                    rs = stat.executeQuery(seleccionar);
                                                    if(rs.next())
                                                    {

                                                        password=rs.getString("passw");


                                                        // La dirección de envío (to)
                                                        para = email;

                                                        // La dirección de la cuenta de envío (from)
                                                        de = "droidlogingestor@gmail.com";

                                                    // El servidor (host). En este caso usamos
localhost
                                                      //  servidor = "localhost";

                                                        // Obtenemos las propiedades del sistema
                                                        propiedades = new Properties();

                                                        // Nombre del host de correo, es
smtp.gmail.com
                                                        propiedades.setProperty("mail.smtp.host",
"smtp.gmail.com");
```

```
                                        // TLS si está disponible
propiedades.setProperty("mail.smtp.starttls.enable", "true");
                                        // Puerto de gmail para envio de correos
propiedades.setProperty("mail.smtp.port","587");

                                        // Nombre del usuario
                                        propiedades.setProperty("mail.smtp.user",
"ejemplo@gmail.com");
                                        // Si requiere o no usuario y password para
conectarse.
                                        propiedades.setProperty("mail.smtp.auth",
"true");
                                        // Obtenemos la sesión por defecto
                                        session =
Session.getDefaultInstance(propiedades);
                                        session.setDebug(true);
                                        try{
                                                // Creamos un objeto mensaje tipo
MimeMessage por defecto.
                                                MimeMessage mensaje = new
MimeMessage(session);
                                                // Asignamos el "de o from" al
header del correo.
                                                mensaje.setFrom(new
InternetAddress(de));
                                                // Asignamos el "para o to" al
header del correo.
        mensaje.addRecipient(Message.RecipientType.TO, new InternetAddress(para));
                                                // Asignamos el asunto
                                                mensaje.setSubject("Recordatorio
de contraseña");
                                                // Asignamos el mensaje como tal
                                                mensaje.setText("Mensaje
recordatorio de contraseña"
                                                        + "\nEste
mensaje es enviado para el usuario: " + usuario +
                                                        "\nCon el
siguiente e-mail: " + email +
                                                        "\nSu contraseña
es: " + password);
                                                // Enviamos el correo
                                                Transport t =
session.getTransport("smtp");
        t.connect("droidlogingestor@gmail.com","Pruebafinal1!");
```

```
t.sendMessage(mensaje,mensaje.getAllRecipients());
                                                        t.close();
                                                        correcto=true;
                                                            con.close();

                                                    } catch (MessagingException e1) {
                                                        e1.printStackTrace();
                                                    }
                                                }
                                                else
                                                    JOptionPane.showMessageDialog(null,"No existe usuario con ese nombre de usuario y ese email","",JOptionPane.WARNING_MESSAGE);

                                                    } catch (ClassNotFoundException | SQLException e1) {
                                                        System.out.println("Error: " + e1.getMessage());
                                                    }

                                            }

                                    }
                                    else
                                            JOptionPane.showMessageDialog(null,"Debe introducir su email","",JOptionPane.WARNING_MESSAGE);

                                }
                        }
                        else
                                JOptionPane.showMessageDialog(null,"Debe introducir su nombre de usuario","",JOptionPane.WARNING_MESSAGE);
                        if(correcto)
                                JOptionPane.showMessageDialog(null,"Se ha enviado un email con su contraseña","",JOptionPane.INFORMATION_MESSAGE);
                                else
                                JOptionPane.showMessageDialog(null,"No se ha podido enviar el email","",JOptionPane.ERROR_MESSAGE);

                                }
                        });
                        lblNewLabel.setForeground(Color.WHITE);
                        lblNewLabel.setBounds(222, 430, 200, 15);
                        panel.add(lblNewLabel);

                        JButton Imagen = new JButton("");
                        Imagen.setBounds(153, 12, 289, 265);
                        Imagen.setContentAreaFilled(false);
                        Imagen.setIcon(new ImageIcon("/home/pgallegos11/Escritorio/tfg/enough-pro-blue_running_man_.jpg"));
                        panel.add(Imagen);

                        JLabel Usuario = new JLabel("Usuario");
                        Usuario.setFont(new Font("Dialog", Font.BOLD, 20));
                        Usuario.setBounds(134, 306, 200, 50);
                        panel.add(Usuario);
```

```
                JLabel Contraseña = new JLabel("Contraseña");
                Contraseña.setFont(new Font("Dialog", Font.BOLD, 20));
                Contraseña.setBounds(134, 368, 200, 50);
                panel.add(Contraseña);

                CampoUsuario = new JTextField();
                CampoUsuario.setBounds(350, 320, 141, 27);
                panel.add(CampoUsuario);
                CampoUsuario.setColumns(10);

                JButton BotonLog = new JButton("Login");
                BotonLog.setFont(new Font("Dialog", Font.BOLD, 18));
                BotonLog.setBounds(204, 476, 200, 50);
                panel.add(BotonLog);
        BotonLog.addActionListener((ActionListener) new click(PanelPresentacion));

                CampoContraseña = new JPasswordField();
                CampoContraseña.setBounds(350, 382, 141, 27);
                panel.add(CampoContraseña);

                JPanel panelRegistro = new JPanel();
                panelRegistro.setBorder(new MatteBorder(1, 1, 1, 1, (Color) new Color(51, 102,
204)));

            panelRegistro.setBackground(UIManager.getColor("CheckBoxMenuItem.acceleratorForeground
"));
                panelRegistro.setBounds(109, 302, 400, 150);
                panelRegistro.add(panelRegistro);
            }


/**
 * @author pgallegos11
 *
 */
public class click implements ActionListener{

            JPanel panelPresentacion=null;

    public click(JPanel panelPresentacion) {
            this.panelPresentacion=panelPresentacion;
    }

            @Override
    public void actionPerformed(ActionEvent e) {
    Component[] componentes=PanelPresentacion.getComponents();
    String password = new String(CampoContraseña.getPassword());
    new clickLogin(CampoUsuario.getText(),password,frame,(JPanel)componentes[0]);

        }
    }
}
```

B.1.2 clickLogin.java

```
package Presentacion;

import java.sql.*;

import javax.swing.JOptionPane;
import javax.swing.JPanel;

//Clase que se encarga de comprobar si el usuario y la contraseña existen y tiene permiso de administrado
public class clickLogin {


	private Connection con;
	private ResultSet rs;

	public clickLogin(String usuario,String contraseña, Presentacion frame, JPanel panelPresentacion){

		try {

			Class.forName("com.mysql.jdbc.Driver");
			con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");

			// Creamos un Statement para poder hacer peticiones a la bd
			Statement stat = con.createStatement();

			// Seleccionar los usuarios administradores con ese nombre y contraseña
			String seleccionar = "SELECT * FROM usuarios WHERE admin=1 and username='"+usuario+"' and passw='"+contraseña+"'";
			rs = stat.executeQuery(seleccionar);

			// Comprobamos que hemos obtenido algo
			if (rs.next())
				new Principal(frame,panelPresentacion);
			else
				JOptionPane.showMessageDialog(null,"Usuario/contraseña incorrecto","",JOptionPane.WARNING_MESSAGE);

		    con.close();

			//En caso contrario mostramos un mensaje de error
		} catch (ClassNotFoundException | SQLException e) {
			System.out.println("Error: " + e.getMessage());
		}

	}
}
```

B.1.3 Principal.java

```
package Presentacion;


import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JRadioButton;
import javax.swing.JScrollPane;
import javax.swing.JTabbedPane;
import javax.swing.JTextArea;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.util.Timer;
import java.util.TimerTask;
import java.util.Vector;

import javax.swing.JTextField;

import kafka.javaapi.consumer.SimpleConsumer;

//Clase que crea la interfaz grafica principal del programa
public class Principal extends JFrame {


	private static final long serialVersionUID = 1L;
	static Presentacion frame;
	public static JPanel contentPane;
	public static JTextField campoNombre;
	public static JTextField campoApellidos;
	public static JTextField campoDni;
	public static JTextField campoEmail;
	public static JTextField campoMovil;
	public static JTextField campoUsuario;
	public static JPasswordField campoContraseña;
	public static JPasswordField campoContraseña2;
	public static JRadioButton rdbtnAdministrador;
	public static JTextField campoTituloRutina;
	public static JTextArea campoDescripcionRutina;
	public JTabbedPane tabbedPane;
	public static JTextField  campoNombreEjercicios;
	public static JTextField  campoFinalidad;
	public static JTextField  campoDuracion;
	public static JTextArea  campoRecomendaciones;
	public static JTextArea  campoCaracteristicas;
	public static JTextArea campoExplicacion;
	public static JTextField campoUsuarioKafka;
	public static JTextField campoEventoKafka;
	public  long offset;
```

```
public long maxoffset;
public static Vector<Mensaje> v;
public int i;
private boolean recarga;
protected boolean primvez=true;
protected int pulsadoatras=0;
protected int vez;
public static int nummensajesmax=36;
public static int tamañomensajes=132;
public static int mensajesporcarga=7;

//Constructores
public Principal(Presentacion frame, JPanel panelPresentacion){
		iniciar();
		if(panelPresentacion!=null)
		{
			panelPresentacion.setVisible(false);
			frame.remove(panelPresentacion);
		}
		frame.add(contentPane);
		frame.setVisible(true);
}


public void iniciar() {
		i=0;
		 final SimpleConsumer simpleConsumer = new SimpleConsumer("localhost",
  9092,
  10000,
  1024);
		 offset = 0;
		//Crea el panel principal donde situamos el panel de pestañas
		this.setResizable(false);
		setBounds(new Rectangle(650, 275, 600, 600));
		setBackground(new Color(51, 102, 153));
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setBounds(650, 275, 600, 600);
		contentPane = new JPanel();
		contentPane.setBounds(new Rectangle(650, 275, 600, 600));
		contentPane.setBackground(new Color(0, 102, 153));
		contentPane.setBorder(null);
		setContentPane(contentPane);
		contentPane.setLayout(null);

		tabbedPane = new JTabbedPane(JTabbedPane.TOP);
		tabbedPane.setBounds(-5, 0, 655, 609);
		tabbedPane.setBorder(null);
		tabbedPane.setBackground(new Color(102, 153, 204));
		contentPane.add(tabbedPane);

		//Comenzamos con la pestaña usuario
		JPanel panelUsuario = new JPanel();
		panelUsuario.setBackground(new Color(51, 102, 15));
		tabbedPane.addTab("Usuario", null, panelUsuario, null);
		panelUsuario.setLayout(null);

		JPanel Usuario = new JPanel();
```

```
Usuario.setBounds(0, 0, 650, 582);
panelUsuario.add(Usuario);
Usuario.setBackground(new Color(51, 102, 153));
Usuario.setLayout(null);

campoNombre = new JTextField();
campoNombre.setBounds(251, 60, 147, 25);
Usuario.add(campoNombre);
campoNombre.setColumns(10);

campoApellidos = new JTextField();
campoApellidos.setColumns(10);
campoApellidos.setBounds(251, 110, 147, 25);
Usuario.add(campoApellidos);

campoDni = new JTextField();
campoDni.setColumns(10);
campoDni.setBounds(251, 160, 147, 25);
Usuario.add(campoDni);

campoEmail = new JTextField();
campoEmail.setColumns(10);
campoEmail.setBounds(251, 210, 147, 25);
Usuario.add(campoEmail);

campoMovil = new JTextField();
campoMovil.setColumns(10);
campoMovil.setBounds(251, 260, 147, 25);
Usuario.add(campoMovil);

campoUsuario = new JTextField();
campoUsuario.setColumns(10);
campoUsuario.setBounds(251, 310, 147, 25);
Usuario.add(campoUsuario);

campoContraseña = new JPasswordField();
campoContraseña.setColumns(10);
campoContraseña.setBounds(251, 360, 147, 25);
Usuario.add(campoContraseña);

campoContraseña2 = new JPasswordField();
campoContraseña2.setColumns(10);
campoContraseña2.setBounds(251, 410, 147, 25);
Usuario.add(campoContraseña2);

JLabel txtNombre = new JLabel("Nombre:");
txtNombre.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombre.setBounds(72, 65, 94, 15);
Usuario.add(txtNombre);

JLabel txtApellido = new JLabel("Apellidos:");
txtApellido.setFont(new Font("Dialog", Font.BOLD, 18));
txtApellido.setBounds(72, 115, 120, 15);
Usuario.add(txtApellido);

JLabel txtDNI = new JLabel("DNI/NIF:");
```

```
txtDNI.setFont(new Font("Dialog", Font.BOLD, 18));
txtDNI.setBounds(72, 165, 94, 15);
Usuario.add(txtDNI);

JLabel txtEmail = new JLabel("Email:");
txtEmail.setFont(new Font("Dialog", Font.BOLD, 18));
txtEmail.setBounds(72, 215, 94, 15);
Usuario.add(txtEmail);

JLabel txtMovil = new JLabel("Móvil:");
txtMovil.setFont(new Font("Dialog", Font.BOLD, 18));
txtMovil.setBounds(72, 265, 94, 15);
Usuario.add(txtMovil);

JLabel txtUsuario = new JLabel("Usuario:");
txtUsuario.setFont(new Font("Dialog", Font.BOLD, 18));
txtUsuario.setBounds(72, 315, 94, 15);
Usuario.add(txtUsuario);

JLabel txtContraseña = new JLabel("Contraseña:");
txtContraseña.setFont(new Font("Dialog", Font.BOLD, 18));
txtContraseña.setBounds(72, 365, 123, 15);
Usuario.add(txtContraseña);

JLabel txtContraseña2 = new JLabel("Contraseña:");
txtContraseña2.setFont(new Font("Dialog", Font.BOLD, 18));
txtContraseña2.setBounds(72, 415, 123, 15);
Usuario.add(txtContraseña2);

JButton btnCrear = new JButton("Crear");
btnCrear.setBounds(449, 110, 130, 39);
Usuario.add(btnCrear);

JButton btnBuscar = new JButton("Buscar");
btnBuscar.setBounds(449, 210, 130, 39);
Usuario.add(btnBuscar);

JButton btnBorrar = new JButton("Limpiar");
btnBorrar.setBounds(449, 310, 130, 39);
Usuario.add(btnBorrar);

rdbtnAdministrador = new JRadioButton("Administrador");
rdbtnAdministrador.setFont(new Font("Dialog", Font.BOLD, 18));
rdbtnAdministrador.setBackground(new Color(51, 102, 153));
rdbtnAdministrador.setBounds(72, 475, 188, 23);
Usuario.add(rdbtnAdministrador);

btnCrear.addActionListener((ActionListener) new clickCrearUsuario());
btnBuscar.addActionListener((ActionListener) new clickBuscarUsuario(panelUsuario,
Usuario, contentPane));
btnBorrar.addActionListener((ActionListener) new clickBorrarUsuario());

//Comenzamos con la pestaña rutina
JPanel PanelRutinas = new JPanel();
PanelRutinas.setBackground(new Color(51, 102, 153));
tabbedPane.addTab("Rutinas", null, PanelRutinas, null);
```

```
PanelRutinas.setLayout(null);


JPanel Rutinas = new JPanel();
Rutinas.setBounds(0, 0, 650, 582);
PanelRutinas.add(Rutinas);
Rutinas.setBackground(new Color(51, 102, 153));
Rutinas.setLayout(null);

campoTituloRutina = new JTextField();
campoTituloRutina.setBounds(110, 120, 280, 25);
Rutinas.add(campoTituloRutina);

campoDescripcionRutina = new JTextArea();
campoDescripcionRutina.setColumns(10);
campoDescripcionRutina.setLineWrap(true);
JScrollPane scroll = new JScrollPane(campoDescripcionRutina);
scroll.setBounds(110, 220, 280, 100);
campoDescripcionRutina.setWrapStyleWord(true);
Rutinas.add(scroll);

JLabel txtTituloRutina = new JLabel("Nombre:");
txtTituloRutina.setFont(new Font("Dialog", Font.BOLD, 18));
txtTituloRutina.setBounds(72, 84, 94, 15);
Rutinas.add(txtTituloRutina);

JLabel txtDescripcionRutina = new JLabel("Descripción:");
txtDescripcionRutina.setFont(new Font("Dialog", Font.BOLD, 18));
txtDescripcionRutina.setBounds(72, 180, 130, 15);
Rutinas.add(txtDescripcionRutina);

JButton btnCrearRutina = new JButton("Crear");
btnCrearRutina.setBounds(449, 110, 130, 39);
Rutinas.add(btnCrearRutina);

JButton btnBuscarRutina = new JButton("Buscar");
btnBuscarRutina.setBounds(449, 210, 130, 39);
Rutinas.add(btnBuscarRutina);

JButton btnBorrarRutina = new JButton("Limpiar");
btnBorrarRutina.setBounds(449, 310, 130, 39);
Rutinas.add(btnBorrarRutina);


btnCrearRutina.addActionListener((ActionListener) new clickCrearRutina());
btnBuscarRutina.addActionListener((ActionListener) new
clickBuscarRutina(PanelRutinas, Rutinas, contentPane));
btnBorrarRutina.addActionListener((ActionListener) new clickBorrarRutina());

//Comenzamos con la pestaña ejercicio
JPanel panelEjercicios = new JPanel();
panelEjercicios.setBackground(new Color(51, 102, 153));
tabbedPane.addTab("Ejercicios", null, panelEjercicios, null);
panelEjercicios.setLayout(null);

JPanel Ejercicios = new JPanel();
```

```
Ejercicios.setBounds(0, 0, 650, 582);
panelEjercicios.add(Ejercicios);
Ejercicios.setBackground(new Color(51, 102, 153));
Ejercicios.setLayout(null);

campoNombreEjercicios = new JTextField();
campoNombreEjercicios.setBounds(231, 30, 167, 25);
Ejercicios.add(campoNombreEjercicios);
campoNombreEjercicios.setColumns(10);

campoFinalidad = new JTextField();
campoFinalidad.setColumns(10);
campoFinalidad.setBounds(231, 80, 167, 25);
Ejercicios.add(campoFinalidad);

campoDuracion = new JTextField();
campoDuracion.setColumns(10);
campoDuracion.setBounds(231, 130, 167, 25);
Ejercicios.add(campoDuracion);

campoRecomendaciones = new JTextArea();
campoRecomendaciones.setColumns(10);
campoRecomendaciones.setLineWrap(true);
JScrollPane scroll1 = new JScrollPane(campoRecomendaciones);
scroll1.setBounds(110, 215, 288, 75);
campoRecomendaciones.setWrapStyleWord(true);
Ejercicios.add(scroll1);

campoCaracteristicas = new JTextArea();
campoCaracteristicas.setColumns(10);
campoCaracteristicas.setLineWrap(true);
JScrollPane scroll2 = new JScrollPane(campoCaracteristicas);
scroll2.setBounds(110, 340, 288, 75);
campoCaracteristicas.setWrapStyleWord(true);
Ejercicios.add(scroll2);

campoExplicacion = new JTextArea();
campoExplicacion.setColumns(10);
campoExplicacion.setLineWrap(true);
JScrollPane scroll3 = new JScrollPane(campoExplicacion);
scroll3.setBounds(110, 465, 288, 75);
campoExplicacion.setWrapStyleWord(true);
Ejercicios.add(scroll3);

JLabel txtNombreEjercicios = new JLabel("Nombre:");
txtNombreEjercicios.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombreEjercicios.setBounds(72, 35, 94, 15);
Ejercicios.add(txtNombreEjercicios);

JLabel txtFinalidad = new JLabel("Finalidad:");
txtFinalidad.setFont(new Font("Dialog", Font.BOLD, 18));
txtFinalidad.setBounds(72, 85, 120, 15);
Ejercicios.add(txtFinalidad);

JLabel txtDuracion = new JLabel("Duración:");
txtDuracion.setFont(new Font("Dialog", Font.BOLD, 18));
```

```
                txtDuracion.setBounds(72, 135, 120, 15);
                Ejercicios.add(txtDuracion);

                JLabel txtRecomendaciones = new JLabel("Recomendación:");
                txtRecomendaciones.setFont(new Font("Dialog", Font.BOLD, 18));
                txtRecomendaciones.setBounds(72, 185, 200, 15);
                Ejercicios.add(txtRecomendaciones);

                JLabel txtCaracteristicas = new JLabel("Características:");
                txtCaracteristicas.setFont(new Font("Dialog", Font.BOLD, 18));
                txtCaracteristicas.setBounds(72, 310, 200, 15);
                Ejercicios.add(txtCaracteristicas);

                JLabel txtExplicacion = new JLabel("Explicación:");
                txtExplicacion.setFont(new Font("Dialog", Font.BOLD, 18));
                txtExplicacion.setBounds(72, 435, 200, 15);
                Ejercicios.add(txtExplicacion);

                JButton btnCrearEjercicios = new JButton("Crear");
                btnCrearEjercicios.setBounds(449, 110, 130, 39);
                Ejercicios.add(btnCrearEjercicios);

                JButton btnBuscarEjercicios = new JButton("Buscar");
                btnBuscarEjercicios.setBounds(449, 210, 130, 39);
                Ejercicios.add(btnBuscarEjercicios);

                JButton btnBorrarEjercicios = new JButton("Limpiar");
                btnBorrarEjercicios.setBounds(449, 310, 130, 39);
                Ejercicios.add(btnBorrarEjercicios);

                btnCrearEjercicios.addActionListener((ActionListener) new clickCrearEjercicio());
                btnBuscarEjercicios.addActionListener((ActionListener) new
clickBuscarEjercicio(panelEjercicios, Ejercicios, contentPane));
                btnBorrarEjercicios.addActionListener((ActionListener) new clickBorrarEjercicio());

                 final JPanel panelKafka = new JPanel();
                 panelKafka.setBackground(new Color(51, 102, 153));
                tabbedPane.addTab("Log", null, panelKafka, null);
                panelKafka.setLayout(null);

                final JPanel Kafka = new JPanel();
                Kafka.setBackground(new Color(51, 102, 153));
                Kafka.setLayout(null);
                Kafka.setBounds(0, 80, 650, 550);
                panelKafka.add(Kafka);
            v = new Vector<Mensaje>();


                JLabel txtUsuarioKafka= new JLabel("Usuario");
                txtUsuarioKafka.setFont(new Font("Dialog", Font.BOLD, 18));
                txtUsuarioKafka.setBounds(10, 10, 94, 15);
                panelKafka.add(txtUsuarioKafka);

                campoUsuarioKafka = new JTextField();
                campoUsuarioKafka.setBounds(10, 40, 167, 25);
                panelKafka.add(campoUsuarioKafka);
```

```
                JLabel txtEventoKafka= new JLabel("Evento");
                txtEventoKafka.setFont(new Font("Dialog", Font.BOLD, 18));
                txtEventoKafka.setBounds(220, 10, 94, 15);
                panelKafka.add(txtEventoKafka);

                campoEventoKafka = new JTextField();
                campoEventoKafka.setBounds(220, 40, 167, 25);
                panelKafka.add(campoEventoKafka);
        final MuestraKafka b =new MuestraKafka(v,simpleConsumer,i,Kafka);
                JButton btnFiltrar = new JButton("Filtrar");
                btnFiltrar.setBounds(440, 40, 130, 25);
                panelKafka.add(btnFiltrar);

                recarga = true;
                final TimerTask timerTask = new TimerTask() {
                  public void run() {
                        if(recarga==true)
                                if(pulsadoatras==0 || pulsadoatras!=0 &&
vez<(nummensajesmax/mensajesporcarga))
                                {
                                        offset=b.MuestraMensajes(offset);
                                        vez++;
                                }

                  }
                };
                final Timer timer = new Timer();
                timer.scheduleAtFixedRate(timerTask, 0, 2000);

                btnFiltrar.addMouseListener(new MouseAdapter() {
                        @Override
                        public void mouseClicked(MouseEvent e) {
                                if(campoUsuarioKafka.getText().isEmpty() &&
campoEventoKafka.getText().isEmpty())
                                {
                                        recarga=true;
                                        v.removeAllElements();

                                }
                                else
                                {

        b.filtraMensaje(campoUsuarioKafka.getText(),campoEventoKafka.getText(),offset);
                                        v.removeAllElements();

                                        recarga=false;
                                }

                        }
                });
                final JLabel lblNewLabel = new JLabel("<");
                lblNewLabel.setFont(new Font("Dialog", Font.BOLD, 20));
                lblNewLabel.setBounds(440, 10, 20, 25);
                panelKafka.add(lblNewLabel);
                lblNewLabel.addMouseListener(new MouseAdapter() {
                        @Override
```

```
		public void mouseClicked(MouseEvent e) {
			vez=0;
			System.out.println("entra < "+offset);
			if(offset>2*nummensajesmax*tamañomensajes)
			{
				maxoffset=offset;
				offset=offset-2*nummensajesmax*tamañomensajes;
				v.removeAllElements();
				offset=b.MuestraMensajes(offset);

			}
			else
			{
				v.removeAllElements();
				pulsadoatras++;

				offset=0;
			}
		}
	});

	final JLabel lblNewLabel1 = new JLabel(">");
	lblNewLabel1.setFont(new Font("Dialog", Font.BOLD, 20));
	lblNewLabel1.setBounds(480, 10, 20, 25);
	panelKafka.add(lblNewLabel1);
	lblNewLabel1.addMouseListener(new MouseAdapter() {
		@Override
		public void mouseClicked(MouseEvent e) {

			if(offset+nummensajesmax*tamañomensajes<maxoffset){
			primvez=true;
			offset=offset+nummensajesmax*tamañomensajes;
			recarga=true;
			System.out.println("entra > "+offset);
			}
			if(pulsadoatras>0)
			{
				pulsadoatras--;
				vez=0;
			}
		}
	});




}

public static void Recargar(JPanel panelPadre, JPanel panelAnterior,int i) {
	panelPadre.remove(panelAnterior);
	if(i==0)
	{
		JPanel Usuario = new JPanel();
		Usuario.setBounds(0, 0, 650, 582);
```

```
panelPadre.add(Usuario);
Usuario.setBackground(new Color(51, 102, 153));
Usuario.setLayout(null);

campoNombre = new JTextField();
campoNombre.setBounds(251, 60, 147, 25);
Usuario.add(campoNombre);
campoNombre.setColumns(10);

campoApellidos = new JTextField();
campoApellidos.setColumns(10);
campoApellidos.setBounds(251, 110, 147, 25);
Usuario.add(campoApellidos);

campoDni = new JTextField();
campoDni.setColumns(10);
campoDni.setBounds(251, 160, 147, 25);
Usuario.add(campoDni);

campoEmail = new JTextField();
campoEmail.setColumns(10);
campoEmail.setBounds(251, 210, 147, 25);
Usuario.add(campoEmail);

campoMovil = new JTextField();
campoMovil.setColumns(10);
campoMovil.setBounds(251, 260, 147, 25);
Usuario.add(campoMovil);

campoUsuario = new JTextField();
campoUsuario.setColumns(10);
campoUsuario.setBounds(251, 310, 147, 25);
Usuario.add(campoUsuario);

campoContraseña = new JPasswordField();
campoContraseña.setColumns(10);
campoContraseña.setBounds(251, 360, 147, 25);
Usuario.add(campoContraseña);

campoContraseña2 = new JPasswordField();
campoContraseña2.setColumns(10);
campoContraseña2.setBounds(251, 410, 147, 25);
Usuario.add(campoContraseña2);

JLabel txtNombre = new JLabel("Nombre:");
txtNombre.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombre.setBounds(72, 65, 94, 15);
Usuario.add(txtNombre);

JLabel txtApellido = new JLabel("Apellidos:");
txtApellido.setFont(new Font("Dialog", Font.BOLD, 18));
txtApellido.setBounds(72, 115, 120, 15);
Usuario.add(txtApellido);

JLabel txtDNI = new JLabel("DNI/NIF:");
txtDNI.setFont(new Font("Dialog", Font.BOLD, 18));
```

```
                txtDNI.setBounds(72, 165, 94, 15);
                Usuario.add(txtDNI);

                JLabel txtEmail = new JLabel("Email:");
                txtEmail.setFont(new Font("Dialog", Font.BOLD, 18));
                txtEmail.setBounds(72, 215, 94, 15);
                Usuario.add(txtEmail);

                JLabel txtMovil = new JLabel("Móvil:");
                txtMovil.setFont(new Font("Dialog", Font.BOLD, 18));
                txtMovil.setBounds(72, 265, 94, 15);
                Usuario.add(txtMovil);

                JLabel txtUsuario = new JLabel("Usuario:");
                txtUsuario.setFont(new Font("Dialog", Font.BOLD, 18));
                txtUsuario.setBounds(72, 315, 94, 15);
                Usuario.add(txtUsuario);

                JLabel txtContraseña = new JLabel("Contraseña:");
                txtContraseña.setFont(new Font("Dialog", Font.BOLD, 18));
                txtContraseña.setBounds(72, 365, 123, 15);
                Usuario.add(txtContraseña);

                JLabel txtContraseña2 = new JLabel("Contraseña:");
                txtContraseña2.setFont(new Font("Dialog", Font.BOLD, 18));
                txtContraseña2.setBounds(72, 415, 123, 15);
                Usuario.add(txtContraseña2);

                JButton btnCrear = new JButton("Crear");
                btnCrear.setBounds(449, 110, 130, 39);
                Usuario.add(btnCrear);

                JButton btnBuscar = new JButton("Buscar");
                btnBuscar.setBounds(449, 210, 130, 39);
                Usuario.add(btnBuscar);

                JButton btnBorrar = new JButton("Limpiar");
                btnBorrar.setBounds(449, 310, 130, 39);
                Usuario.add(btnBorrar);

                rdbtnAdministrador = new JRadioButton("Administrador");
                rdbtnAdministrador.setFont(new Font("Dialog", Font.BOLD, 18));
                rdbtnAdministrador.setBackground(new Color(51, 102, 153));
                rdbtnAdministrador.setBounds(72, 475, 188, 23);
                Usuario.add(rdbtnAdministrador);

                btnCrear.addActionListener((ActionListener) new clickCrearUsuario());
                btnBuscar.addActionListener((ActionListener) new
clickBuscarUsuario(panelPadre, Usuario, contentPane));
                btnBorrar.addActionListener((ActionListener) new clickBorrarUsuario());
        }
        if(i==1)
        {
                JPanel Rutinas = new JPanel();
                Rutinas.setBounds(0, 0, 650, 582);
                panelPadre.add(Rutinas);
```

```
                    Rutinas.setBackground(new Color(51, 102, 153));
                    Rutinas.setLayout(null);

                    campoTituloRutina = new JTextField();
                    campoTituloRutina.setBounds(110, 120, 280, 25);
                    Rutinas.add(campoTituloRutina);

                    campoDescripcionRutina = new JTextArea();
                    campoDescripcionRutina.setColumns(10);
                    campoDescripcionRutina.setLineWrap(true);
                    JScrollPane scroll = new JScrollPane(campoDescripcionRutina);
                    scroll.setBounds(110, 220, 280, 100);
                    campoDescripcionRutina.setWrapStyleWord(true);
                    Rutinas.add(scroll);

                    JLabel txtTituloRutina = new JLabel("Nombre:");
                    txtTituloRutina.setFont(new Font("Dialog", Font.BOLD, 18));
                    txtTituloRutina.setBounds(72, 84, 94, 15);
                    Rutinas.add(txtTituloRutina);

                    JLabel txtDescripcionRutina = new JLabel("Descripción:");
                    txtDescripcionRutina.setFont(new Font("Dialog", Font.BOLD, 18));
                    txtDescripcionRutina.setBounds(72, 180, 130, 15);
                    Rutinas.add(txtDescripcionRutina);

                    JButton btnCrearRutina = new JButton("Crear");
                    btnCrearRutina.setBounds(449, 110, 130, 39);
                    Rutinas.add(btnCrearRutina);

                    JButton btnBuscarRutina = new JButton("Buscar");
                    btnBuscarRutina.setBounds(449, 210, 130, 39);
                    Rutinas.add(btnBuscarRutina);

                    JButton btnBorrarRutina = new JButton("Limpiar");
                    btnBorrarRutina.setBounds(449, 310, 130, 39);
                    Rutinas.add(btnBorrarRutina);


                    btnCrearRutina.addActionListener((ActionListener) new clickCrearRutina());
                    btnBuscarRutina.addActionListener((ActionListener) new
clickBuscarRutina(panelPadre, Rutinas, contentPane));
                    btnBorrarRutina.addActionListener((ActionListener) new
clickBorrarRutina());
                }
                if(i==2)
                {
                    JPanel Ejercicios = new JPanel();
                    Ejercicios.setBounds(0, 0, 650, 582);
                    panelPadre.add(Ejercicios);
                    Ejercicios.setBackground(new Color(51, 102, 153));
                    Ejercicios.setLayout(null);

                    campoNombreEjercicios = new JTextField();
                    campoNombreEjercicios.setBounds(231, 30, 167, 25);
                    Ejercicios.add(campoNombreEjercicios);
                    campoNombreEjercicios.setColumns(10);
```

```
campoFinalidad = new JTextField();
campoFinalidad.setColumns(10);
campoFinalidad.setBounds(231, 80, 167, 25);
Ejercicios.add(campoFinalidad);

campoDuracion = new JTextField();
campoDuracion.setColumns(10);
campoDuracion.setBounds(231, 130, 167, 25);
Ejercicios.add(campoDuracion);

campoRecomendaciones = new JTextArea();
campoRecomendaciones.setColumns(10);
campoRecomendaciones.setLineWrap(true);
JScrollPane scroll1 = new JScrollPane(campoRecomendaciones);
scroll1.setBounds(110, 215, 288, 75);
campoRecomendaciones.setWrapStyleWord(true);
Ejercicios.add(scroll1);

campoCaracteristicas = new JTextArea();
campoCaracteristicas.setColumns(10);
campoCaracteristicas.setLineWrap(true);
JScrollPane scroll2 = new JScrollPane(campoCaracteristicas);
scroll2.setBounds(110, 340, 288, 75);
campoCaracteristicas.setWrapStyleWord(true);
Ejercicios.add(scroll2);

campoExplicacion = new JTextArea();
campoExplicacion.setColumns(10);
campoExplicacion.setLineWrap(true);
JScrollPane scroll3 = new JScrollPane(campoExplicacion);
scroll3.setBounds(110, 465, 288, 75);
campoExplicacion.setWrapStyleWord(true);
Ejercicios.add(scroll3);

JLabel txtNombreEjercicios = new JLabel("Nombre:");
txtNombreEjercicios.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombreEjercicios.setBounds(72, 35, 94, 15);
Ejercicios.add(txtNombreEjercicios);

JLabel txtFinalidad = new JLabel("Finalidad:");
txtFinalidad.setFont(new Font("Dialog", Font.BOLD, 18));
txtFinalidad.setBounds(72, 85, 120, 15);
Ejercicios.add(txtFinalidad);

JLabel txtDuracion = new JLabel("Duración:");
txtDuracion.setFont(new Font("Dialog", Font.BOLD, 18));
txtDuracion.setBounds(72, 135, 120, 15);
Ejercicios.add(txtDuracion);

JLabel txtRecomendaciones = new JLabel("Recomendación:");
txtRecomendaciones.setFont(new Font("Dialog", Font.BOLD, 18));
txtRecomendaciones.setBounds(72, 185, 200, 15);
Ejercicios.add(txtRecomendaciones);

JLabel txtCaracteristicas = new JLabel("Características:");
```

```
                        txtCaracteristicas.setFont(new Font("Dialog", Font.BOLD, 18));
                        txtCaracteristicas.setBounds(72, 310, 200, 15);
                        Ejercicios.add(txtCaracteristicas);

                        JLabel txtExplicacion = new JLabel("Explicación:");
                        txtExplicacion.setFont(new Font("Dialog", Font.BOLD, 18));
                        txtExplicacion.setBounds(72, 435, 200, 15);
                        Ejercicios.add(txtExplicacion);

                        JButton btnCrearEjercicios = new JButton("Crear");
                        btnCrearEjercicios.setBounds(449, 110, 130, 39);
                        Ejercicios.add(btnCrearEjercicios);

                        JButton btnBuscarEjercicios = new JButton("Buscar");
                        btnBuscarEjercicios.setBounds(449, 210, 130, 39);
                        Ejercicios.add(btnBuscarEjercicios);

                        JButton btnBorrarEjercicios = new JButton("Limpiar");
                        btnBorrarEjercicios.setBounds(449, 310, 130, 39);
                        Ejercicios.add(btnBorrarEjercicios);

                        btnCrearEjercicios.addActionListener((ActionListener) new
clickCrearEjercicio());
                        btnBuscarEjercicios.addActionListener((ActionListener) new
clickBuscarEjercicio(panelPadre, Ejercicios, contentPane));
                        btnBorrarEjercicios.addActionListener((ActionListener) new
clickBorrarEjercicio());


                }
        }

}
```

B.1.4 clickCrearUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;
import java.sql.*;

import javax.swing.JOptionPane;

//Clase encargada de crear los nuevos usuarios
public class clickCrearUsuario implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private ResultSet rs2;
	private int id=0;
	private String admin;
	private Boolean completo=false;
	private Boolean sal=false;

	 public  clickCrearUsuario() {

	  }

	@Override
	public void actionPerformed(ActionEvent e) {

		String password1 = new String(Principal.campoContraseña.getPassword());
		String password2 = new String(Principal.campoContraseña2.getPassword());
		completo=false;
		//Comprobamos que todos los campos esten completos
		if(Principal.campoNombre.getText().equals("")==false &&
Principal.campoApellidos.getText().equals("")==false
				&& Principal.campoDni.getText().equals("") ==false   &&
Principal.campoEmail.getText().equals("") ==false
				&& Principal.campoMovil.getText().equals("") ==false &&
Principal.campoUsuario.getText().equals("") ==false
				&& password1.equals("")==false && password2.equals("")==false)
			completo=true;
		//Comprobamos si el boton administardo estaba pulsado
		if(Principal.rdbtnAdministrador.isSelected())
			admin ="1";
		else
			admin ="0";

		//Comprobamos que las dos contraseñas sean iguales
		if(password1.equals(password2) && completo==true)
		{
			try {
				Class.forName("com.mysql.jdbc.Driver");
				con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			// Creamos un Statement para poder hacer peticiones a la bd
				Statement stat = con.createStatement();
```

```
                                        String comprobar= "SELECT * FROM usuarios WHERE
username='"+Principal.campoUsuario.getText()+"'";
                                        rs = stat.executeQuery(comprobar);
                                        //Comprobamos que el usuario este libre
                                        if (rs.next())
                                                JOptionPane.showMessageDialog(null,"Usuario en
uso","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                rs.close();
                                                System.out.println("entra else");
                                                String seleccionar = "SELECT * FROM usuarios";
                                                rs2 = stat.executeQuery(seleccionar);
                                                //Obtenemos el menor id que este libre
                                                for( id=1;sal==false;id++)
                                                {
                                                        if(rs2.next()){

        if(!rs2.getString("Id_Usuario").equals(""+id))
                                                                        sal=true;
                                                        }
                                                        else
                                                                sal=true;
                                                }

                                                id--;

                                                String insertar= "INSERT INTO usuarios VALUES
('"+id+"','" + admin + "','" + Principal.campoNombre.getText() + "','"
                                                        + Principal.campoApellidos.getText() + "','"
+ Principal.campoDni.getText() + "','" + Principal.campoEmail.getText() + "','"
                                                        + Principal.campoMovil.getText() + "','" +
Principal.campoUsuario.getText() + "','"+ password1 + "')";
                                                stat.executeUpdate(insertar);
                                                //Añadimos en la base de datos mostramos el mensaje por
pantalla y vaciamos los campos
                                                JOptionPane.showMessageDialog(null,"Añadido
Correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                                Principal.campoNombre.setText(null);
                                                Principal.campoApellidos.setText(null);
                                                Principal.campoDni.setText(null);
                                                Principal.campoEmail.setText(null);
                                                Principal.campoMovil.setText(null);
                                                Principal.campoUsuario.setText(null);
                                                Principal.campoContraseña.setText(null);
                                                Principal.campoContraseña2.setText(null);
                                                Principal.rdbtnAdministrador.setSelected(false);


                                        }
                                    con.close();

                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }  } else {
                                        //En caso de no estar completo o ser las contraseñas diferentes
mostramos sus respectivos avisos por pantalla
```

```
                                        if(completo==false)
                                                JOptionPane.showMessageDialog(null,"Algun campo esta
vacio","",JOptionPane.WARNING_MESSAGE);
                                        else
                                                JOptionPane.showMessageDialog(null,"Las contraseñas no
coinciden","",JOptionPane.WARNING_MESSAGE);
                                }
                }
}
```

B.1.5 clickBorrarUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

//Clase encargada de vaciar todos los campos del formulario Usuario
public class clickBorrarUsuario implements ActionListener {

	@Override
	public void actionPerformed(ActionEvent e) {

		Principal.campoNombre.setText(null);
		Principal.campoApellidos.setText(null);
		Principal.campoDni.setText(null);
		Principal.campoEmail.setText(null);
		Principal.campoMovil.setText(null);
		Principal.campoUsuario.setText(null);
		Principal.campoContraseña.setText(null);
		Principal.campoContraseña2.setText(null);
		Principal.rdbtnAdministrador.setSelected(false);

	}
}
```

B.1.6 clickBuscarUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.DriverManager;

import javax.swing.JOptionPane;
import javax.swing.JPanel;

import java.sql.*;

//Clase encargada de buscar los usuarios que cumplan las codiciones establecidas
//Primero comprueba si el campo esta vacio y despues en el caso de no estar vacio
//añade la condicion de ese campo al string de busqueda
public class clickBuscarUsuario implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private String seleccionar;
	private JPanel panelUsuario;
	private JPanel Usuario;
	private JPanel panel;

	public clickBuscarUsuario(JPanel panelUsuario, JPanel Usuario, JPanel contentPane){

		this.panelUsuario=panelUsuario;
		this.Usuario=Usuario;
		this.panel=contentPane;

	}



	@Override
	public void actionPerformed(ActionEvent e) {

		String busqueda="";
		String admin;
		//Comprobamos si esta seleccionado el boton de administrador
		if(Principal.rdbtnAdministrador.isSelected())
		{
			admin ="1";
			busqueda=busqueda+"admin="+admin;
		}

		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");

			Statement stat = con.createStatement();
			boolean primero=true;
			//para cada campo comprobamos si esta vacio
			if(!Principal.campoNombre.getText().equals(""))
```

```
                    if(primero==true)
                    {       //En caso de no estarlo comprobamos si es es el primero que
no lo esta, en cuyo caso no añadimos and al string y ponemos primero a false
                            busqueda=busqueda+"
nombre='"+Principal.campoNombre.getText()+"'";
                            primero=false;
                    }
                    else
                            busqueda=busqueda+" and
nombre='"+Principal.campoNombre.getText()+"'";

                if(!Principal.campoApellidos.getText().equals(""))
                    if(primero==true)
                    {
                            busqueda=busqueda+"
apellidos='"+Principal.campoApellidos.getText()+"'";
                            primero=false;
                    }
                    else
                            busqueda=busqueda+" and
apellidos='"+Principal.campoApellidos.getText()+"'";

                if(!Principal.campoDni.getText().equals(""))
                    if(primero==true)
                    {
                            busqueda=busqueda+"
dni='"+Principal.campoDni.getText()+"'";
                            primero=false;
                    }
                    else
                            busqueda=busqueda+" and
dni='"+Principal.campoDni.getText()+"'";

                if(!Principal.campoEmail.getText().equals(""))
                    if(primero==true)
                    {
                            busqueda=busqueda+"
direccion='"+Principal.campoEmail.getText()+"'";
                            primero=false;
                    }
                    else
                            busqueda=busqueda+" and
direccion='"+Principal.campoEmail.getText()+"'";

                if(!Principal.campoMovil.getText().equals(""))
                    if(primero==true)
                    {
                            busqueda=busqueda+"
movil='"+Principal.campoMovil.getText()+"'";
                            primero=false;
                    }
                    else
                            busqueda=busqueda+" and
movil='"+Principal.campoMovil.getText()+"'";

                if(!Principal.campoUsuario.getText().equals(""))
                    if(primero==true)
```

```
                                        {
                                                busqueda=busqueda+"
username='"+Principal.campoUsuario.getText()+"'";
                                                primero=false;
                                        }
                                        else
                                                busqueda=busqueda+" and
username='"+Principal.campoUsuario.getText()+"'";

                        //Si la sentencia de busqueda se ha modificado realizamos esa busqueda, en caso
contrario obtenemos todos los usuarios
                                if(!busqueda.equals(""))
                                        seleccionar = "SELECT * FROM usuarios WHERE"+busqueda;
                                else
                                        seleccionar = "SELECT * FROM usuarios";


                                rs = stat.executeQuery(seleccionar);
                                //Si ningun usuario cumple las conidciones lo mostramos por pantalla
                                if (!rs.next())
                                        JOptionPane.showMessageDialog(null,"Ningún usuario cumple las
condiciones escritas","",JOptionPane.WARNING_MESSAGE);
                                else{
                                        //En caso contrario comprobamos cuantos hay y creamos un objetct
que rellenamos
                                        rs.last();
                                        int cuantos = rs.getRow();
                                        rs.beforeFirst();
                                        Object [][]ob=new Object[cuantos][4];
                                        int i=0;
                                        while (rs.next())
                                        {
                                                ob[i][0]=rs.getString("nombre");
                                                ob[i][1]=rs.getString("apellidos");
                                                ob[i][2]=rs.getString("dni");
                                                ob[i][3]=rs.getString("username");
                                                i++;
                                        }
                                        final String[] columnNames = {"Nombre",
                                                        "Apellidos",
                                                        "DNI",
                                        "Usuario"};

                                        new
UsuariosEncontrados(panelUsuario,Usuario,ob,panel,columnNames);

                                }

                            con.close();
                        } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                        }


                }
}
```

B.1.7 UsuariosEncontrados.java

```
ackage Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
//Clase que muestra los usuarios que cumplen los parametros de busqueda y permite acceder a el
public class UsuariosEncontrados extends JFrame {


        private static final long serialVersionUID = 1L;
        public JPanel usuario;
   public JPanel panelUsuarios;
   public JPanel panelEncontrados;

        public UsuariosEncontrados(JPanel panelUsuario, JPanel usuario, final Object[][] ob, final
JPanel tabbedPane,final String[] columnNames) {
        this.panelUsuarios=panelUsuario;
      this.usuario=usuario;
      usuario.setVisible(false);
      panelEncontrados = new JPanel();
      panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      final MyTableModel myModel = new MyTableModel(ob,columnNames);
                  final JTable table = new JTable(myModel);
                  table.setSelectionBackground(new Color(0, 0, 255));
                  table.setBackground(new Color(0, 102, 153));
                  table.setBorder(null);
                  table.setFont(new Font("Dialog", Font.BOLD, 15));
                  table.setOpaque(true);

      //Creatamos un contenedor para la Tabla
      JScrollPane scrollPane = new JScrollPane(table);
      scrollPane.setBounds(new Rectangle(2, 5, 600, 582));
      scrollPane.getViewport().setBackground(new Color(51, 102, 153));
      scrollPane.getViewport().setBorder(null);

      //Añadimos el Listener del click del raton
      table.addMouseListener(new MouseAdapter()
      {
        public void mouseClicked(MouseEvent e)
        {
            int modificar=10;
            int fila = table.rowAtPoint(e.getPoint());
            int columna = table.columnAtPoint(e.getPoint());
```

```
            if ((fila > -1) && (columna > -1))
            {
                    modificar=JOptionPane.showConfirmDialog(null,"Desea ver el
usuario:"+ob[fila][3],"", JOptionPane.YES_NO_OPTION);
                    if(modificar==0)
                            new
UsuarioConcreto(panelUsuarios,panelEncontrados,ob[fila][3],tabbedPane);
            }
        }
      });

      panelEncontrados.add(scrollPane);
      panelUsuarios.add(panelEncontrados);
   }
```

B.1.8 UsuarioConcreto.java

```
package Presentacion;

import java.awt.Font;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JTextField;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
//Clase que muestra la unformacion del usuario concreto y que permite interactuar con el
public class UsuarioConcreto extends JFrame {

	private static final long serialVersionUID = 1L;
	public String usuariostring;
	private Connection con;
	private ResultSet rs;
	public static JTextField campoNombre;
	public static JTextField campoApellidos;
	public static JTextField campoEmail;
	public static JTextField campoDni;
	public static JTextField campoMovil;
	public static JTextField campoUsuario;
	public static JTextField campoContraseña;
	public static JRadioButton rdbtnAdministrador;

	public UsuarioConcreto(JPanel panelUsuario2, JPanel panelEncontrados, Object ob, JPanel
tabbedPane) {
		usuariostring=ob.toString();

		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String comprobar= "SELECT * FROM usuarios WHERE
username='"+usuariostring+"'";
			rs = stat.executeQuery(comprobar);
			rs.next();
			usuariostring=rs.getString("Id_Usuario");
			panelUsuario2.removeAll();

			JPanel panelUsuario = new JPanel();
			panelUsuario.setBackground(new Color(51, 102, 153));
			panelUsuario.setBounds(0, 0, 650, 582);
			panelUsuario.setLayout(null);
			panelUsuario2.add(panelUsuario);

			JPanel Usuario = new JPanel();
```

```
Usuario.setBounds(0, 0, 650, 582);
panelUsuario.add(Usuario);
Usuario.setBackground(new Color(51, 102, 153));
Usuario.setLayout(null);

campoNombre = new JTextField();
campoNombre.setBounds(251, 61, 147, 25);
Usuario.add(campoNombre);
campoNombre.setColumns(10);
campoNombre.setText(rs.getString("nombre"));

campoApellidos = new JTextField();
campoApellidos.setColumns(10);
campoApellidos.setBounds(251, 110, 147, 25);
Usuario.add(campoApellidos);
campoApellidos.setText(rs.getString("apellidos"));

campoDni = new JTextField();
campoDni.setColumns(10);
campoDni.setBounds(251, 160, 147, 25);
Usuario.add(campoDni);
campoDni.setText(rs.getString("dni"));

campoEmail = new JTextField();
campoEmail.setColumns(10);
campoEmail.setBounds(251, 210, 147, 25);
Usuario.add(campoEmail);
campoEmail.setText(rs.getString("direccion"));

campoMovil = new JTextField();
campoMovil.setColumns(10);
campoMovil.setBounds(251, 260, 147, 25);
Usuario.add(campoMovil);
campoMovil.setText(rs.getString("movil"));

campoUsuario = new JTextField();
campoUsuario.setColumns(10);
campoUsuario.setBounds(251, 310, 147, 25);
Usuario.add(campoUsuario);
campoUsuario.setText(rs.getString("username"));

campoContraseña = new JTextField();
campoContraseña.setColumns(10);
campoContraseña.setBounds(251, 360, 147, 25);
Usuario.add(campoContraseña);
campoContraseña.setText(rs.getString("passw"));

JLabel txtNombre = new JLabel("Nombre");
txtNombre.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombre.setBounds(72, 65, 94, 15);
Usuario.add(txtNombre);

JLabel txtApellido = new JLabel("Apellidos");
txtApellido.setFont(new Font("Dialog", Font.BOLD, 18));
txtApellido.setBounds(72, 115, 94, 15);
Usuario.add(txtApellido);
```

```
JLabel txtDNI = new JLabel("DNI");
txtDNI.setFont(new Font("Dialog", Font.BOLD, 18));
txtDNI.setBounds(72, 165, 94, 15);
Usuario.add(txtDNI);

JLabel txtEmail = new JLabel("Email");
txtEmail.setFont(new Font("Dialog", Font.BOLD, 18));
txtEmail.setBounds(72, 215, 94, 15);
Usuario.add(txtEmail);

JLabel txtMovil = new JLabel("Móvil");
txtMovil.setFont(new Font("Dialog", Font.BOLD, 18));
txtMovil.setBounds(72, 265, 94, 15);
Usuario.add(txtMovil);

JLabel txtUsuario = new JLabel("Usuario");
txtUsuario.setFont(new Font("Dialog", Font.BOLD, 18));
txtUsuario.setBounds(72, 315, 94, 15);
Usuario.add(txtUsuario);

JLabel txtContraseña = new JLabel("Contraseña");
txtContraseña.setFont(new Font("Dialog", Font.BOLD, 18));
txtContraseña.setBounds(72, 365, 123, 15);
Usuario.add(txtContraseña);

rdbtnAdministrador = new JRadioButton("Administrador");
rdbtnAdministrador.setFont(new Font("Dialog", Font.BOLD, 18));
rdbtnAdministrador.setBackground(new Color(51, 102, 153));
rdbtnAdministrador.setBounds(72, 415, 188, 23);
if(rs.getString("admin").equals(""+1))
        rdbtnAdministrador.setSelected(true);
Usuario.add(rdbtnAdministrador);

JButton btnGuardar = new JButton("Guardar");
btnGuardar.setBounds(449, 40, 130, 39);
Usuario.add(btnGuardar);

JButton btnDeshacer = new JButton("Deshacer");
btnDeshacer.setBounds(449, 140, 130, 39);
Usuario.add(btnDeshacer);

JButton btnAtras = new JButton("Atrás");
btnAtras.setBounds(449, 240, 130, 39);
Usuario.add(btnAtras);

JButton btnRutinas = new JButton("Rutinas");
btnRutinas.setBounds(449, 440, 130, 39);
Usuario.add(btnRutinas);

JButton btnEliminar = new JButton("Eliminar");
btnEliminar.setBounds(449, 340, 130, 39);
Usuario.add(btnEliminar);

btnGuardar.addActionListener((ActionListener) new
clickGuardarUsuario(usuariostring));
```

```
                        btnDeshacer.addActionListener((ActionListener) new
clickDeshacerUsuario(usuariostring));
                        btnAtras.addActionListener((ActionListener) new
clickAtrasUsuario(panelUsuario,Usuario));
                        btnEliminar.addActionListener((ActionListener) new
clickEliminarUsuario(panelUsuario,Usuario,usuariostring));
                        btnRutinas.addActionListener((ActionListener) new
clickRutinasUsuario(panelUsuario,Usuario,usuariostring,tabbedPane));

                    con.close();


                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

        }
}
```

B.1.9 clickGuardarUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;

import javax.swing.JOptionPane;

//Clase encargada de guardar los cambios en la base de datos
public class clickGuardarUsuario implements ActionListener {


	private Connection con;
	private ResultSet rs;
	private String usuariostring;
	private String id;

   public clickGuardarUsuario(String id) {

	this.id=id;

   }


   @Override
   public void actionPerformed(ActionEvent e) {
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea guardar los
cambios?","", JOptionPane.YES_NO_OPTION);
	//Tras confirmar que se desea guardar los cambios, diferenciamos dos casos
	if(modificar==0)
	{

		String admin;
		if(UsuarioConcreto.rdbtnAdministrador.isSelected())
			admin ="1";
		else
			admin ="0";
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String comprobar= "SELECT * FROM usuarios WHERE
Id_Usuario='"+id+"'";
			rs = stat.executeQuery(comprobar);
			rs.next();
			usuariostring=rs.getString("username");
			//Caso uno , no se desea cambiar el nombre de usuario, se procede a guardar
los demas cambios
			if(usuariostring.equals(UsuarioConcreto.campoUsuario.getText())){
				String cambio="UPDATE usuarios SET admin='"+ admin+"', nombre
='"+ UsuarioConcreto.campoNombre.getText()+"', apellidos='"+
UsuarioConcreto.campoApellidos.getText()
+ "', dni='"+UsuarioConcreto.campoDni.getText()+"'
, direccion='"+ UsuarioConcreto.campoEmail.getText()+"' , movil='" +
UsuarioConcreto.campoMovil.getText()
```

```
                                                    + "', username='" +
UsuarioConcreto.campoUsuario.getText()+"', passw='" +UsuarioConcreto.campoContraseña.getText()+
"' WHERE username='"+usuariostring+"'";
                                        stat.executeUpdate(cambio);
                                                JOptionPane.showMessageDialog(null,"Cambios guardados
correctamente","",JOptionPane.INFORMATION_MESSAGE);

                                }
                                else
                                {
                                        //Caso dos, se desea cambiar el nombre de usuario, antes de guardar se
comprueba que ese nombre este libre mostrando un mensaje en caso contrario
                                        comprobar= "SELECT * FROM usuarios WHERE
username='"+UsuarioConcreto.campoUsuario.getText()+"'";
                                        rs = stat.executeQuery(comprobar);
                                        if (rs.next())
                                                JOptionPane.showMessageDialog(null,"Si desea cambiar el
nombre de usuario debe utilizar uno libre","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                String cambio="UPDATE usuarios SET nombre ='"+
UsuarioConcreto.campoNombre.getText()+"', apellidos='"+ UsuarioConcreto.campoApellidos.getText()
                                                                + "',
dni='"+UsuarioConcreto.campoDni.getText()+"' , direccion='"+
UsuarioConcreto.campoEmail.getText()+"' , movil='" + UsuarioConcreto.campoMovil.getText()
                                                                + "', username='" +
UsuarioConcreto.campoUsuario.getText()+"', passw='" +UsuarioConcreto.campoContraseña.getText()+
"' WHERE username='"+usuariostring+"'";
                                                stat.executeUpdate(cambio);
                                                JOptionPane.showMessageDialog(null,"Cambios guardados
correctamente","",JOptionPane.INFORMATION_MESSAGE);

                                        }

                                }
                        con.close();

                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }
        }
    }
}
```

B.1.10 clickDeshacerUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;

//Clase encargada de volver a rellenar el formulario con los datos de la base de datos
public class clickDeshacerUsuario implements ActionListener {

	private  Connection con;
	private ResultSet rs;
	private  String id;
   public clickDeshacerUsuario(String id) {
		this.id=id;
		}
   @Override
   public void actionPerformed(ActionEvent e) {
//Pedimos confirmacion de deshacer los cambios y realizamos una nueva consulta a la base de datos
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea cancelar los
cambios?","", JOptionPane.YES_NO_OPTION);
	if(modificar==0)
	{
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String cargar= "SELECT * FROM usuarios WHERE Id_Usuario="'+ id +"'";
			rs = stat.executeQuery(cargar);
			if (rs.next())
			{
		//tras obtener los datos los colocamos de nuevo en los campos correspondientes
				UsuarioConcreto.campoNombre.setText(rs.getString("nombre"));
				UsuarioConcreto.campoApellidos.setText(rs.getString("apellidos"));
				UsuarioConcreto.campoDni.setText(rs.getString("dni"));
				UsuarioConcreto.campoEmail.setText(rs.getString("direccion"));
				UsuarioConcreto.campoMovil.setText(rs.getString("movil"));
				UsuarioConcreto.campoUsuario.setText(rs.getString("username"));
				UsuarioConcreto.campoContraseña.setText(rs.getString("passw"));
				if(rs.getString("admin").equals(""+1))
					UsuarioConcreto.rdbtnAdministrador.setSelected(true);
				else
					UsuarioConcreto.rdbtnAdministrador.setSelected(false);
			}
		    con.close();
		} catch (ClassNotFoundException | SQLException i) {
			System.out.println("Error: " + i.getMessage());
		}
	}
   }
}
```

B.1.11 clickAtrasaUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JPanel;

//Clase encargada de volver a la ventana inicial de usuario
public class clickAtrasUsuario implements ActionListener {

	private JPanel panelPadre=null;
	private JPanel usuario=null;

	public clickAtrasUsuario( JPanel panelPadre, JPanel usuario) {

		this.panelPadre=panelPadre;
		this.usuario=usuario;

	}

	@Override
	public void actionPerformed(ActionEvent e) {

		Principal.Recargar(panelPadre,usuario,0);

	}

}
```

B.1.12 clickEliminarUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase encargada de eliminar un usuario de la base de datos
public class clickEliminarUsuario implements ActionListener {

	private JPanel panelPadre=null;
	private Connection con;
	private int rs;
	private String id;
	private JPanel usuario=null;

   public clickEliminarUsuario(JPanel panelPadre, JPanel usuario, String id) {
	this.panelPadre=panelPadre;
	this.usuario=usuario;
	this.id=id;
   }

   @Override
   public void actionPerformed(ActionEvent e) {
	int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar el usuario?","",
JOptionPane.YES_NO_OPTION);
			if(modificar==0)
			{
	try {
		//Elimina el usuario y comprueba si  se ha eliminado correctamente mostrando un mensaje de
error en caso contrario
			Class.forName("com.mysql.jdbc.Driver");
			con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();

			String comprobar= "DELETE FROM usuarios WHERE Id_Usuario="'+id+"'";
			rs = stat.executeUpdate(comprobar);
			if(rs==1)
				JOptionPane.showMessageDialog(null,"Usuario eliminado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
			else
				JOptionPane.showMessageDialog(null,"Usuario no eliminado
correctamente","",JOptionPane.ERROR_MESSAGE);
			    con.close();

	} catch (ClassNotFoundException | SQLException i) {
			System.out.println("Error: " + i.getMessage());
	}
	//Vuelve a la ventana principal, a la pestaña por defecto(0)
			Principal.Recargar(panelPadre,usuario,0);
			}
   }
}
```

B.1.13 clickRutinasUsuario.java

```
package Presentacion;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;
//Clase cuya funcion es cargar las rutinas en dos grupos
//Estos grupos son las que estan asignados a este usuario y los que no
public class clickRutinasUsuario implements ActionListener {

	private JPanel panelUsuario=null;
	private JPanel Usuario=null;
	private JPanel tabbedPane=null;
	private String id=null;
	private Connection con;
	private ResultSet rs;
	private int i;

	public clickRutinasUsuario(JPanel panelUsuario, JPanel Usuario, String id, JPanel tabbedPane)
{

		this.panelUsuario=panelUsuario;
		this.Usuario=Usuario;
		this.id=id;
		this.tabbedPane=tabbedPane;
	}

	public clickRutinasUsuario() {
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			//Consulta a la base de datos que identifica loas rutinas que estan asignados al
usuario

			String seleccionar = "SELECT r.Id_Rutina Id_Rutina,r.descripcion
descripcion,r.descripcion_completa descripcion_completa "
					+ " FROM rutinasdeusuarios rde INNER JOIN rutinas r ON
r.Id_Rutina=rde.Id_Rutina WHERE rde.Id_Usuario="'+id+"'";

			rs = stat.executeQuery(seleccionar);
			rs.last();
			i = rs.getRow();
			rs.beforeFirst();
			//Object donde almacenamos las rutinas que si estan asignados
```

```
                        Object [][]si=new Object[i][2];
                        i=0;
                        while (rs.next())
                        {

                                si[i][0]=rs.getString("descripcion");
                                si[i][1]=rs.getString("descripcion_completa");
                                i++;

                        }
                        //Consulta a la base de datos que identifica loas rutinas que no estan asignados
al usuario

                        seleccionar = "SELECT ru.Id_Rutina Id_Rutina,ru.descripcion
descripcion,ru.descripcion_completa descripcion_completa FROM rutinas ru WHERE ru.Id_Rutina NOT
IN "
                                        + "(SELECT rde.Id_Rutina FROM rutinasdeusuarios rde
WHERE rde.Id_Usuario='"+id+"')";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        //Object donde almacenamos las rutinas que no estan asignados

                        Object [][]no=new Object[i][2];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("descripcion");
                                no[i][1]=rs.getString("descripcion_completa");
                                i++;
                        }

                        final String[] columnNames = {"Nombre",
                        "Descripcion"};
                    con.close();

                    con.close();

                        new RutinasdeUsuarioEncontradas( panelUsuario,  Usuario,  si,
columnNames,no,tabbedPane,id,true);


                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

        }

        //Metodo que se encarga de realizar una actualizacion de estos grupos
        public  void actualiza(JPanel panelrutina, JPanel rutina, JPanel tabbedPane2, String id2,boolean
filtro) {

                try {
```

```
                    Class.forName("com.mysql.jdbc.Driver");
                    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                    Statement stat = con.createStatement();


                    String seleccionar = "SELECT r.Id_Rutina Id_Rutina,r.descripcion
descripcion,r.descripcion_completa descripcion_completa "
                                        + " FROM rutinasdeusuarios rde INNER JOIN rutinas r ON
r.Id_Rutina=rde.Id_Rutina WHERE rde.Id_Usuario="'+id2+"'";

                    rs = stat.executeQuery(seleccionar);
                    rs.last();
                    i = rs.getRow();
                    rs.beforeFirst();
                    Object [][]si=new Object[i][2];
                    i=0;
                    while (rs.next())
                    {

                            si[i][0]=rs.getString("descripcion");
                            si[i][1]=rs.getString("descripcion_completa");
                            i++;

                    }

                    seleccionar = "SELECT ru.Id_Rutina Id_Rutina,ru.descripcion
descripcion,ru.descripcion_completa descripcion_completa FROM rutinas ru WHERE ru.Id_Rutina NOT
IN "
                                        + "(SELECT rde.Id_Rutina FROM rutinasdeusuarios rde
WHERE rde.Id_Usuario="'+id2+"')";

                    rs = stat.executeQuery(seleccionar);
                    rs.last();
                    i = rs.getRow();
                    rs.beforeFirst();
                    Object [][]no=new Object[i][2];
                    i=0;
                    while (rs.next())
                    {

                            no[i][0]=rs.getString("descripcion");
                            no[i][1]=rs.getString("descripcion_completa");
                            i++;

                    }

                    final String[] columnNames = {"Nombre",
                    "Descripcion"};
                con.close();

                    new RutinasdeUsuarioEncontradas( panelrutina,  rutina,  si,
columnNames,no,tabbedPane2,id2,filtro);
                con.close();

            } catch (ClassNotFoundException | SQLException i2) {
                    System.out.println("Error: " + i2.getMessage());
```

```
                }
        }
        //Metodo que se encarga de realizar una actualizacion con filtros
        public void actualizafiltra(JPanel panelrutina, JPanel rutina, JPanel tabbedPane2, String id2,
String text, String text2,
                        String text3) {
                boolean esta=false;
                i=0;
                try {
                        Class.forName("com.mysql.jdbc.Driver");
                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                        Statement stat = con.createStatement();


                        String seleccionar = "SELECT r.Id_Rutina Id_Rutina,r.descripcion
descripcion,r.descripcion_completa descripcion_completa "
                                        + " FROM rutinasdeusuarios rde INNER JOIN rutinas r ON
r.Id_Rutina=rde.Id_Rutina WHERE rde.Id_Usuario='"+id2+"'";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        Object [][]si=new Object[i][2];
                        i=0;
                        while (rs.next())
                        {
                                si[i][0]=rs.getString("descripcion");
                                si[i][1]=rs.getString("descripcion_completa");
                                i++;
                        }

                        seleccionar = "SELECT ru.Id_Rutina Id_Rutina,ru.descripcion
descripcion,ru.descripcion_completa descripcion_completa FROM rutinas ru WHERE ru.Id_Rutina NOT
IN "
                                        + "(SELECT rde.Id_Rutina FROM rutinasdeusuarios rde
WHERE rde.Id_Usuario='"+id2+"')";

                        rs = stat.executeQuery(seleccionar);
                        i=0;
        while (rs.next())
        {
         esta=false;

         if(!text.equals(""))
         {
                if(rs.getString("descripcion").indexOf(text)==-1)
                {
                        if(!text.equals(""))
                                if(!(rs.getString("descripcion_completa").indexOf(text)==-1))
                                        esta=true;
                }
                else
                        esta=true;
         }
         if(!text2.equals(""))
```

```
{
	if(rs.getString("descripcion").indexOf(text2)==-1)
	{
		if(!text2.equals(""))
			if(!(rs.getString("descripcion_completa").indexOf(text2)==-1))
				esta=true;
	}
	else
		esta=true;
}
if(!text3.equals(""))
{
	if(rs.getString("descripcion").indexOf(text3)==-1)
	{
		if(!text3.equals(""))
			if(!(rs.getString("descripcion_completa").indexOf(text3)==-1))
				esta=true;
	}
	else
		esta=true;
}
if(esta==true)
	i++;

		}
		rs = stat.executeQuery(seleccionar);
		Object [][]no=new Object[i][2];
		i=0;
		while (rs.next())
		{
esta=false;

if(!text.equals(""))
{
	if(rs.getString("descripcion").indexOf(text)==-1)
	{
		if(!text.equals(""))
			if(!(rs.getString("descripcion_completa").indexOf(text)==-1))
				esta=true;
	}
	else
		esta=true;
}
if(!text2.equals(""))
{
	if(rs.getString("descripcion").indexOf(text2)==-1)
	{
		if(!text2.equals(""))
			if(!(rs.getString("descripcion_completa").indexOf(text2)==-1))
				esta=true;
	}
	else
		esta=true;
}
if(!text3.equals(""))
{
	if(rs.getString("descripcion").indexOf(text3)==-1)
	{
```

```
                        if(!text3.equals(""))
                                if(!(rs.getString("descripcion_completa").indexOf(text3)==-1))
                                        esta=true;
                }
                else
                        esta=true;
        }

        if(esta==true)
        {
                                no[i][0]=rs.getString("descripcion");
                                no[i][1]=rs.getString("descripcion_completa");
                                i++;
        }
                        }

                        final String[] columnNames = {"Nombre",
                        "Descripcion"};
                    con.close();

                        new RutinasdeUsuarioEncontradas( panelrutina,  rutina,  si,
columnNames,no,tabbedPane2,id2,false);

                    con.close();


                } catch (ClassNotFoundException | SQLException i2) {
                        System.out.println("Error: " + i2.getMessage());
                }
        }

}
```

B.1.14 RutinasEncontradas.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.TableColumn;
//Clase que muestra las rutinas que cumplen los parametros de busqueda y permite acceder a ella
public class RutinasEncontradas extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel rutina;
   public JPanel panelrutinas;
   public JPanel panelEncontrados;
   private Connection con;
   private ResultSet rs;

	public RutinasEncontradas(JPanel panelrutina, JPanel rutina, final Object[][] ob, final JPanel
tabbedPane,final String[] columnNames) {
		this.panelrutinas=panelrutina;
      this.rutina=rutina;
      rutina.setVisible(false);
      panelEncontrados = new JPanel();
      panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      final MyTableModel myModel = new MyTableModel(ob,columnNames);
			final JTable table = new JTable(myModel);
			TableColumn columna;
			columna=table.getColumnModel().getColumn(0);
			columna.setPreferredWidth(230);
			columna.setMaxWidth(230);



			table.setSelectionBackground(new Color(0, 0, 255));
			table.setBackground(new Color(0, 102, 153));
			table.setBorder(null);
			table.setFont(new Font("Dialog", Font.BOLD, 15));
      //Creatamos un contenedor para la Tabla
      final JScrollPane scrollPane = new JScrollPane(table);
```

```
        scrollPane.setBounds(new Rectangle(2, 5, 600, 582));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);

        //Añadimos el Listener del click del raton
        table.addMouseListener(new MouseAdapter()
        {
          public void mouseClicked(MouseEvent e)
          {
              int modificar;
              int fila = table.rowAtPoint(e.getPoint());
              int columna = table.columnAtPoint(e.getPoint());
              if ((fila > -1) && (columna > -1))
              {
                        modificar=JOptionPane.showConfirmDialog(null,"Desea ver la
rutina:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                        if(modificar==0)
                        {
                                  try {
                                            Class.forName("com.mysql.jdbc.Driver");
                                            con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                            Statement stat = con.createStatement();


                                            String comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+ob[fila][0]+"'";
                                            rs = stat.executeQuery(comprobar);
                                            rs.next();
                                            String idrut=rs.getString("Id_Rutina");
                                  new RutinaConcreta(panelrutinas,panelEncontrados,idrut,tabbedPane);
                                    con.close();

                                  } catch (ClassNotFoundException | SQLException i) {
                                            System.out.println("Error: " + i.getMessage());
                                        }

                        }
              }
          }
        });

        panelEncontrados.add(scrollPane);
        panelrutinas.add(panelEncontrados);
    }

}
```

B.1.15 RutinasdeUsuarioEncontradas.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.TableColumn;
//Clase que muestra los dos grupos de rutinas segun esten asignados o no y permite cambiarlo
public class RutinasdeUsuarioEncontradas extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel rutina;
   public JPanel panelrutinas;
   public JPanel panelEncontrados;
   public JPanel panelPadre;
   public String id;
   private Connection con;
   private ResultSet rs;
   private int rs1;

	public RutinasdeUsuarioEncontradas(JPanel panelrutina, JPanel rutina, final Object[][] ob, final
String[] columnNames,final Object[][] ob2,final JPanel panelPadre, final String id,boolean
muestra_filtro) {

		this.panelrutinas=panelrutina;
		String usuariostring=null;
      this.rutina=rutina;
      this.panelPadre=panelPadre;
      this.id=id;
      rutina.setVisible(false);
      panelEncontrados = new JPanel();
      panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
```

```
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                Statement stat = con.createStatement();


                                String comprobar= "SELECT * FROM usuarios WHERE
Id_Usuario="'+id+"'";
                                rs = stat.executeQuery(comprobar);
                                rs.next();
                                usuariostring = "Usuario: "+rs.getString("Nombre")+"
"+rs.getString("Apellidos");
            } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                        }
        JLabel usuariotxt = new JLabel(usuariostring);
        usuariotxt.setFont(new Font("Dialog", Font.BOLD, 20));
        usuariotxt.setBounds(2, 15, 500, 28);
                        panelEncontrados.add(usuariotxt);

        JLabel asignadas = new JLabel("Rutinas asignadas:");
        asignadas.setFont(new Font("Dialog", Font.BOLD, 20));
        asignadas.setBounds(2, 50, 500, 28);
                        panelEncontrados.add(asignadas);
        final MyTableModel myModel = new MyTableModel(ob,columnNames);
                        final JTable table = new JTable(myModel);
                        TableColumn columna;
                        columna=table.getColumnModel().getColumn(0);
                        columna.setPreferredWidth(230);
                        columna.setMaxWidth(230);

                        table.setSelectionBackground(new Color(0, 0, 255));
                        table.setBackground(new Color(0, 102, 153));
                        table.setBorder(null);
                        table.setFont(new Font("Dialog", Font.BOLD, 15));
        //Creatamos un contenedor para la Tabla
        final JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setBounds(new Rectangle(2, 80, 600, 160));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);

        table.addMouseListener(new MouseAdapter()
        {
          public void mouseClicked(MouseEvent e)
          {
              int modificar;
              int fila = table.rowAtPoint(e.getPoint());
              int columna = table.columnAtPoint(e.getPoint());
              if ((fila > -1) && (columna > -1))
              {
                        modificar=JOptionPane.showConfirmDialog(null,"Desea desasignar la
rutina:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                       if(modificar==0)
                                try {
                                        Class.forName("com.mysql.jdbc.Driver");
                                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                        Statement stat = con.createStatement();
```

```
                                        String comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+ob[fila][0]+"'";
                                        rs = stat.executeQuery(comprobar);
                                        rs.next();
                                        String idrut = rs.getString("Id_Rutina");
                                    comprobar= "DELETE FROM rutinasdeusuarios WHERE
Id_Usuario='"+id+"' and Id_Rutina='"+idrut+"'";
                                    rs1 = stat.executeUpdate(comprobar);
                                    if(rs1==1)
                                            JOptionPane.showMessageDialog(null,"Desasignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                    else
                                            JOptionPane.showMessageDialog(null,"No desasignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                                      con.close();

                                    clickRutinasUsuario  b=new clickRutinasUsuario();
                                    b.actualiza(panelrutinas,panelEncontrados, panelPadre ,id,true);
                            } catch (ClassNotFoundException | SQLException i) {
                                    System.out.println("Error: " + i.getMessage());
                            }
                    }
                }
            });
            JLabel noasignadas = new JLabel("Rutinas no asignadas:");
            noasignadas.setFont(new Font("Dialog", Font.BOLD, 20));
            noasignadas.setBounds(2, 245, 500, 28);
                    panelEncontrados.add(noasignadas);


            final MyTableModel myModel2 = new MyTableModel(ob2,columnNames);
                    final JTable table2 = new JTable(myModel2);

                    columna=table2.getColumnModel().getColumn(0);
                    columna.setPreferredWidth(230);
                    columna.setMaxWidth(230);

                    table2.setSelectionBackground(new Color(0, 0, 255));
                    table2.setBackground(new Color(0, 102, 153));
                    table2.setBorder(null);
                    table2.setFont(new Font("Dialog", Font.BOLD, 15));
            //Creatamos un contenedor para la Tabla
            final JScrollPane scrollPane2 = new JScrollPane(table2);
            scrollPane2.setBounds(new Rectangle(2, 275, 600, 180));
            scrollPane2.getViewport().setBackground(new Color(51, 102, 153));
            scrollPane2.getViewport().setBorder(null);



            //Agregamos nuestra tabla al contenedor
            table2.addMouseListener(new MouseAdapter()
            {
              public void mouseClicked(MouseEvent e)
              {
                  int modificar;
                  int fila = table2.rowAtPoint(e.getPoint());
```

```
            int columna = table2.columnAtPoint(e.getPoint());
            if ((fila > -1) && (columna > -1))
            {
                    modificar=JOptionPane.showConfirmDialog(null,"Desea asignar la
rutina:"+ob2[fila][0],"", JOptionPane.YES_NO_OPTION);
                    if(modificar==0)
                            try {
                                    Class.forName("com.mysql.jdbc.Driver");
                                    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                    Statement stat = con.createStatement();
                                    String comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+ob2[fila][0]+"'";
                                    rs = stat.executeQuery(comprobar);
                                    rs.next();
                                    String idrut = rs.getString("Id_Rutina");
                                     comprobar= "INSERT INTO rutinasdeusuarios VALUES
('"+id+"','"+idrut+"')";
                                    rs1 = stat.executeUpdate(comprobar);
                                    if(rs1==1)
                                            JOptionPane.showMessageDialog(null,"Asignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                    else
                                            JOptionPane.showMessageDialog(null,"No asignado
correctamente","",JOptionPane.ERROR_MESSAGE);
                                                    con.close();

                                    clickRutinasUsuario  b=new clickRutinasUsuario();
                                    b.actualiza(panelrutinas,panelEncontrados, panelPadre ,id,true);
                            } catch (ClassNotFoundException | SQLException i) {
                                    System.out.println("Error: " + i.getMessage());
                            }
            }
      }
    });
    panelEncontrados.add(scrollPane);

        final JButton btnAtras = new JButton("Atrás");
                btnAtras.setBounds(422, 520, 148, 39);
                panelEncontrados.add(btnAtras);

                btnAtras.addActionListener((ActionListener) new
clickAtrasaUsuario(panelrutina,panelEncontrados,id,panelPadre));

                        final JTextField campo;
                        campo = new JTextField();
                        campo.setBounds(30, 480,  147, 25);
                        panelEncontrados.add(campo);
                        campo.setColumns(10);

                        final JTextField campo1;
                        campo1 = new JTextField();
                        campo1.setBounds(226, 480,  147, 25);
                        campo.setColumns(10);

                        final JTextField campo2;
                        campo2 = new JTextField();
```

```
                campo2.setBounds(422, 480,  147, 25);
                campo2.setColumns(10);

                final JLabel lblNewLabel1 = new JLabel("+");
                final JLabel lblNewLabel = new JLabel("+");
                lblNewLabel.setFont(new Font("Dialog", Font.BOLD, 20));
                lblNewLabel.setBounds(226, 480, 20, 25);
                panelEncontrados.add(lblNewLabel);
                lblNewLabel.addMouseListener(new MouseAdapter() {
                        @Override
                        public void mouseClicked(MouseEvent e) {
                                panelEncontrados.add(campo1);
                                panelEncontrados.add(lblNewLabel1);
                                panelEncontrados.remove(lblNewLabel);
                                panelEncontrados.updateUI();
                                panelrutinas.updateUI();
                        }
                });
                lblNewLabel1.setFont(new Font("Dialog", Font.BOLD, 20));
                lblNewLabel1.setBounds(422, 480, 40, 25);
                lblNewLabel1.addMouseListener(new MouseAdapter() {
                        @Override
                        public void mouseClicked(MouseEvent e) {
                                panelEncontrados.add(campo2);
                                panelEncontrados.remove(lblNewLabel1);
                                panelEncontrados.updateUI();
                                panelrutinas.updateUI();
                        }
                });
                final JButton btnFiltrar = new JButton("Filtrar");
                btnFiltrar.setBounds(30, 520, 148, 39);
                panelEncontrados.add(btnFiltrar);
                btnFiltrar.addMouseListener(new MouseAdapter() {
                        @Override
                        public void mouseClicked(MouseEvent e) {
                                clickRutinasUsuario  b=new clickRutinasUsuario();
                                b.actualizafiltra(panelrutinas,panelEncontrados, panelPadre
,id,campo.getText(),campo1.getText(),campo2.getText());
                        }
                });

        panelEncontrados.add(scrollPane2);

        final JButton btnTodas = new JButton("Mostrar todas");
        btnTodas.setBounds(226, 520, 148, 39);
        panelEncontrados.add(btnTodas);
        btnTodas.addMouseListener(new MouseAdapter() {
                @Override
                public void mouseClicked(MouseEvent e) {

                        clickRutinasUsuario  b=new clickRutinasUsuario();
                        b.actualiza(panelrutinas,panelEncontrados, panelPadre ,id,false);
                }
        });
        panelrutinas.add(panelEncontrados);
    }
}
```

B.1.16 clickAtrasaUsuario.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;

public class clickAtrasaUsuario implements ActionListener {

	private JPanel panelPadre;
	private JPanel panelUsuario;
	private JPanel panelEncontrados;
	private String id;

	public clickAtrasaUsuario(JPanel panelUsuario, JPanel panelEncontrados,String id, JPanel
panelPadre) {

		this.id=id;
		this.panelPadre=panelPadre;
		this.panelUsuario=panelUsuario;
		this.panelEncontrados=panelEncontrados;

		}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Conectamos a la base de datos para acceder al nombre del ejercicio
		try {
			Class.forName("com.mysql.jdbc.Driver");
			Connection con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			// Creamos un Statement para poder hacer peticiones a la bd
			Statement stat = con.createStatement();
			String comprobar= "SELECT * FROM usuarios WHERE
Id_Usuario='"+id+"'";
			ResultSet rs = stat.executeQuery(comprobar);
			rs.next();
			String id=rs.getString("username");
			new UsuarioConcreto(panelUsuario,panelEncontrados,id,panelPadre);
		    con.close();


		} catch (ClassNotFoundException | SQLException i) {
			System.out.println("Error: " + i.getMessage());
		}

	}

}
```

B.1.17 clickCrearRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;

//Clase encargada de crear las nuevas rutinas
public class clickCrearRutina implements ActionListener{

	private Connection con;
	private ResultSet rs;
	private ResultSet rs2;
	private int id=0;
	private Boolean sal=false;


	public  clickCrearRutina() {
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Comprobamos que los dos campos tengan algo escrito
		if(Principal.campoTituloRutina.getText().equals("")==false &&
Principal.campoDescripcionRutina.getText().equals("")==false)
		{

			try {
				Class.forName("com.mysql.jdbc.Driver");
				con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
				Statement stat = con.createStatement();

				//Comprobamos si ya existe una rutina con ese nombre
				String comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+Principal.campoTituloRutina.getText()+"'";
				rs = stat.executeQuery(comprobar);
				//En caso de estar vacio mostramos el mensaje de nombre de rutina en
uso
				if (rs.next())
					JOptionPane.showMessageDialog(null,"Nombre de rutina en
uso","",JOptionPane.WARNING_MESSAGE);
				else
				{
					//En caso contrario hallamos el id mas bajo libre
					rs.close();
					String seleccionar = "SELECT * FROM rutinas";
					rs2 = stat.executeQuery(seleccionar);

					for( id=1;sal==false;id++)
```

```
                                        {
                                                if(rs2.next()){

            if(!rs2.getString("Id_Rutina").equals(""+id))
                                                                sal=true;
                                                }
                                                else
                                                        sal=true;


                                        }
                    //Debemos restar uno por la ultima ejecucion del id++ del bucle for
                                        id--;
                                        //Añadimos la rutina , mostramos un mensaje y limpiamos
los campos
                                        String insertar= "INSERT INTO rutinas VALUES ('"+id+"','"
+ Principal.campoTituloRutina.getText() + "','"
                                                        +
Principal.campoDescripcionRutina.getText() + "')";
                                        stat.executeUpdate(insertar);
                                        JOptionPane.showMessageDialog(null,"Añadida
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                        Principal.campoTituloRutina.setText(null);
                                        Principal.campoDescripcionRutina.setText(null);

                                    }
                                con.close();

                            } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                            }
                    }
                    else
                            JOptionPane.showMessageDialog(null,"Algún campo esta
vacio","",JOptionPane.WARNING_MESSAGE);

            }
}
```

B.1.18 clickBorrarRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

//Clase encargada de vaciar todos los campos del formulario Rutina
public class clickBorrarRutina implements ActionListener {

	@Override
	public void actionPerformed(ActionEvent e) {

		Principal.campoTituloRutina.setText(null);
		Principal.campoDescripcionRutina.setText(null);

	}
}
```

B.1.19 clickBuscarRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;

//Clase encargada de buscar las rutinas que cumplan las codiciones establecidas
//Primero comprueba si el campo esta vacio y despues en el caso de no estar vacio
//comprueba si las palabras escritas estan en alguna rutina de la base de datos
public class clickBuscarRutina implements ActionListener{

		private Connection con;
		private ResultSet rs;
		private String seleccionar;
	public JPanel panelRutina;
	private JPanel Rutina;
	private JPanel panelPadre;
	private boolean esta=false;
	private int i=0;

	public clickBuscarRutina(JPanel panelRutina, JPanel Rutina, JPanel panelPadre){

		this.panelRutina=panelRutina;
	   this.Rutina=Rutina;
	   this.panelPadre=panelPadre;

		}


	@Override
		public void actionPerformed(ActionEvent e) {


	    try {
	    Class.forName("com.mysql.jdbc.Driver");
	    con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
	    seleccionar = "SELECT * FROM rutinas";
			Statement stat = con.createStatement();
	    rs = stat.executeQuery(seleccionar);


	   // Primero calculamos cuantas rutinas cumplen las condiciones
	    while (rs.next())
	    {
	     esta=false;

	     if(!Principal.campoTituloRutina.getText().equals(""))
	     {
```

```
        if(rs.getString("descripcion").indexOf(Principal.campoTituloRutina.getText())==-1)
                        {
                                if(!Principal.campoDescripcionRutina.getText().equals(""))
                                {

        if(rs.getString("descripcion_completa").indexOf(Principal.campoDescripcionRutina.getText())=
=-1)
                                                esta=false;
                                        else
                                                esta=true;
                                }
                        }
                        else
                                esta=true;
                }
                else
                        if(!Principal.campoDescripcionRutina.getText().equals(""))

        if(rs.getString("descripcion_completa").indexOf(Principal.campoDescripcionRutina.getText())=
=-1)
                                                esta=false;
                                        else
                                                esta=true;
                                else
                                        esta=true;



                if(esta==true)
                        i++;

                        }
                //Si el numero es 0 mostramos que ninguna rutina cumple las condiciones
                //En caso de ser distinto de cero realizamos el mismo algoritmo que antes pero guardando
                //datos en un object
                if (i==0)
                  JOptionPane.showMessageDialog(null,"Ninguna rutina contiene esas
palabras","",JOptionPane.WARNING_MESSAGE);
                else{
                  Object [][]ob=new Object[i][3];
                  i=0;
                  rs.beforeFirst();
                  while (rs.next())
                  {
                        esta=false;
                        if(!Principal.campoTituloRutina.getText().equals(""))
                        {

        if(rs.getString("descripcion").indexOf(Principal.campoTituloRutina.getText())==-1)
                                {
                                        if(!Principal.campoDescripcionRutina.getText().equals(""))
                                        {

        if(rs.getString("descripcion_completa").indexOf(Principal.campoDescripcionRutina.getText())=
=-1)
                                                                esta=false;
                                                        else
```

```
                                                    esta=true;
                                            }
                                    }
                                    else
                                            esta=true;
                            }
                            else
                                    if(!Principal.campoDescripcionRutina.getText().equals(""))

        if(rs.getString("descripcion_completa").indexOf(Principal.campoDescripcionRutina.getText())=
=-1)
                                                    esta=false;
                                            else
                                                    esta=true;
                                    else
                                            esta=true;


                            if(esta==true)
                            {
                                    ob[i][0]=rs.getString("descripcion");
                                    ob[i][1]=rs.getString("descripcion_completa");
                                    i++;
                            }

                                    final String[] columnNames = {"Nombre",
                     "Descripcion"};
                                    new RutinasEncontradas(panelRutina,Rutina,ob,panelPadre,columnNames);

                             }
                        }
                            con.close();
                        } catch (ClassNotFoundException | SQLException i) {
                          System.out.println("Error: " + i.getMessage());
                        }


                }
}
```

B.1.20 RutinasEncontradas.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.TableColumn;
//Clase que muestra las rutinas que cumplen los parametros de busqueda y permite acceder a ella
public class RutinasEncontradas extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel rutina;
   public JPanel panelrutinas;
   public JPanel panelEncontrados;
   private Connection con;
   private ResultSet rs;

	public RutinasEncontradas(JPanel panelrutina, JPanel rutina, final Object[][] ob, final JPanel
tabbedPane,final String[] columnNames) {
		this.panelrutinas=panelrutina;
      this.rutina=rutina;
      rutina.setVisible(false);
      panelEncontrados = new JPanel();
      panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      final MyTableModel myModel = new MyTableModel(ob,columnNames);
			final JTable table = new JTable(myModel);
			TableColumn columna;
			columna=table.getColumnModel().getColumn(0);
			columna.setPreferredWidth(230);
			columna.setMaxWidth(230);



			table.setSelectionBackground(new Color(0, 0, 255));
			table.setBackground(new Color(0, 102, 153));
			table.setBorder(null);
			table.setFont(new Font("Dialog", Font.BOLD, 15));
      //Creatamos un contenedor para la Tabla
      final JScrollPane scrollPane = new JScrollPane(table);
```

```
        scrollPane.setBounds(new Rectangle(2, 5, 600, 582));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);

        //Añadimos el Listener del click del raton
        table.addMouseListener(new MouseAdapter()
        {
          public void mouseClicked(MouseEvent e)
          {
              int modificar;
              int fila = table.rowAtPoint(e.getPoint());
              int columna = table.columnAtPoint(e.getPoint());
              if ((fila > -1) && (columna > -1))
              {
                        modificar=JOptionPane.showConfirmDialog(null,"Desea ver la
rutina:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                        if(modificar==0)
                        {
                                  try {
                                            Class.forName("com.mysql.jdbc.Driver");
                                            con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                            Statement stat = con.createStatement();


                                            String comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+ob[fila][0]+"'";
                                            rs = stat.executeQuery(comprobar);
                                            rs.next();
                                            String idrut=rs.getString("Id_Rutina");
                                  new RutinaConcreta(panelrutinas,panelEncontrados,idrut,tabbedPane);
                                    con.close();

                                  } catch (ClassNotFoundException | SQLException i) {
                                            System.out.println("Error: " + i.getMessage());
                                        }

                        }
              }
          }
        });

        panelEncontrados.add(scrollPane);
        panelrutinas.add(panelEncontrados);
    }

}
```

B.1.21 RutinaConcreta.java

```
package Presentacion;

import java.awt.Font;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

//Clase que muestra la unformacion de la rutina concreta y que permite interactuar con ella
public class RutinaConcreta extends JFrame {

	private static final long serialVersionUID = 1L;
	public String id;
	private Connection con;
	private ResultSet rs;
	public static JTextField campoTituloRutina;
	public static JTextArea campoDescripcionRutina;
	public RutinaConcreta(JPanel panelRutinas2, JPanel panelEncontrados, Object ob, JPanel
tabbedPane) {
		id=ob.toString();
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String comprobar= "SELECT * FROM rutinas WHERE Id_Rutina='"+id+"'";
			rs = stat.executeQuery(comprobar);
			rs.next();
			panelRutinas2.removeAll();

			JPanel PanelRutinas = new JPanel();
			PanelRutinas.setBackground(new Color(51, 102, 153));
			PanelRutinas.setBounds(0, 0, 650, 582);
			PanelRutinas.setLayout(null);
			panelRutinas2.add(PanelRutinas);

			JPanel Rutinas = new JPanel();
			Rutinas.setBounds(0, 0, 650, 582);
			PanelRutinas.add(Rutinas);
			Rutinas.setBackground(new Color(51, 102, 153));
			Rutinas.setLayout(null);

			campoTituloRutina = new JTextField();
			campoTituloRutina.setBounds(110, 120, 280, 25);
			Rutinas.add(campoTituloRutina);
			campoTituloRutina.setText(rs.getString("descripcion"));
```

```
                campoDescripcionRutina = new JTextArea();
                campoDescripcionRutina.setColumns(10);
                campoDescripcionRutina.setLineWrap(true);
                JScrollPane scroll = new JScrollPane(campoDescripcionRutina);
                scroll.setBounds(110, 220, 280, 100);
                campoDescripcionRutina.setWrapStyleWord(true);
                campoDescripcionRutina.setText(rs.getString("descripcion_completa"));
                Rutinas.add(scroll);

                JLabel txtTituloRutina = new JLabel("Nombre:");
                txtTituloRutina.setFont(new Font("Dialog", Font.BOLD, 18));
                txtTituloRutina.setBounds(72, 84, 94, 15);
                Rutinas.add(txtTituloRutina);

                JLabel txtDescripcionRutina = new JLabel("Descripción:");
                txtDescripcionRutina.setFont(new Font("Dialog", Font.BOLD, 18));
                txtDescripcionRutina.setBounds(72, 180, 130, 15);
                Rutinas.add(txtDescripcionRutina);

                JButton btnGuardar = new JButton("Guardar");
                btnGuardar.setBounds(449, 40, 130, 39);
                Rutinas.add(btnGuardar);

                JButton btnDeshacer = new JButton("Deshacer");
                btnDeshacer.setBounds(449, 140, 130, 39);
                Rutinas.add(btnDeshacer);

                JButton btnAtras = new JButton("Atrás");
                btnAtras.setBounds(449, 240, 130, 39);
                Rutinas.add(btnAtras);

                JButton btnRutinas = new JButton("Ejercicios");
                btnRutinas.setBounds(449, 440, 130, 39);
                Rutinas.add(btnRutinas);

                JButton btnEliminar = new JButton("Eliminar");
                btnEliminar.setBounds(449, 340, 130, 39);
                Rutinas.add(btnEliminar);

                btnGuardar.addActionListener((ActionListener) new clickGuardarRutina(id));
                btnDeshacer.addActionListener((ActionListener) new
clickDeshacerRutina(id));
                btnAtras.addActionListener((ActionListener) new
clickAtrasRutina(PanelRutinas, Rutinas));
                btnEliminar.addActionListener((ActionListener) new
clickEliminarRutina(PanelRutinas, Rutinas,id));
                btnRutinas.addActionListener((ActionListener) new
clickEjerciciosRutina(PanelRutinas, Rutinas, id,tabbedPane));

                    con.close();
                } catch (ClassNotFoundException | SQLException i) {
                    System.out.println("Error: " + i.getMessage());
                }

        }
}
```

B.1.22 clickGuardarRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;

import javax.swing.JOptionPane;
//Clase encargada de guardar los cambios en la base de datos
public class clickGuardarRutina implements ActionListener {


	private Connection con;
	private ResultSet rs;
	private String rutinastring;
	private String id;

	public clickGuardarRutina(String id) {

		this.id=id;

	}


	@Override
	public void actionPerformed(ActionEvent e) {
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea guardar los
cambios?","", JOptionPane.YES_NO_OPTION);
	//Tras confirmar que se desea guardar los cambios, diferenciamos dos casos

		if(modificar==0)
		{
			try {
				Class.forName("com.mysql.jdbc.Driver");
				con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
				Statement stat = con.createStatement();
				String comprobar= "SELECT * FROM rutinas WHERE
Id_Rutina='"+id+"'";
				rs = stat.executeQuery(comprobar);
				rs.next();
				rutinastring=rs.getString("descripcion");

			//Caso uno , no se desea cambiar el nombre de la rutina, se procede a guardar
los demas cambios

				if(rutinastring.equals(RutinaConcreta.campoTituloRutina.getText())){
					String cambio="UPDATE rutinas SET descripcion ='"+
RutinaConcreta.campoTituloRutina.getText()+"', descripcion_completa='"+
RutinaConcreta.campoDescripcionRutina.getText()+"' WHERE descripcion='"+rutinastring+"'";
					stat.executeUpdate(cambio);
					JOptionPane.showMessageDialog(null,"Cambios guardados
correctamente","",JOptionPane.INFORMATION_MESSAGE);
				}
				else
				{
				//Caso dos, se desea cambiar el nombre de rutina, antes de guardar se
```

comprueba que ese nombre este libre mostrando un mensaje en caso contrario

```
                                        comprobar= "SELECT * FROM rutinas WHERE
descripcion='"+RutinaConcreta.campoTituloRutina.getText()+"'";
                                        rs = stat.executeQuery(comprobar);
                                        if (rs.next())
                                                JOptionPane.showMessageDialog(null,"Si desea
cambiar el nombre de la rutina debe usar uno libre","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                String cambio="UPDATE rutinas SET descripcion
='"+ RutinaConcreta.campoTituloRutina.getText()+"', descripcion_completa='"+
RutinaConcreta.campoDescripcionRutina.getText()+"' WHERE descripcion='"+rutinastring+"'";
                                                stat.executeUpdate(cambio);
                                                JOptionPane.showMessageDialog(null,"Cambios
guardados correctamente","",JOptionPane.INFORMATION_MESSAGE);

                                        }
                                }

                        } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                        }
        }
    }
}
```

B.1.23 clickDeshacerRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
//Clase encargada de volver a rellenar el formulario con los datos de la base de datos
public class clickDeshacerRutina implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private String id;

   public clickDeshacerRutina(String id) {

	this.id=id;

   }

   @Override
   public void actionPerformed(ActionEvent e) {
	//Pedimos confirmacion de deshacer los cambios y realizamos una nueva consulta a la base de datos
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea cancelar los cambios?","", JOptionPane.YES_NO_OPTION);
	if(modificar==0)
	{
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String cargar= "SELECT * FROM rutinas WHERE Id_Rutina="'+ id +"'";
			rs = stat.executeQuery(cargar);
			if (rs.next())
			{
				//tras obtener los datos los colocamos de nuevo en los campos correspondientes

	RutinaConcreta.campoTituloRutina.setText(rs.getString("descripcion"));

	RutinaConcreta.campoDescripcionRutina.setText(rs.getString("descripcion_completa"));
			}
			    con.close();

			} catch (ClassNotFoundException | SQLException i) {
				System.out.println("Error: " + i.getMessage());
			}
	}
   }
}
```

B.1.24 clickAtrasaRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


import javax.swing.JPanel;

public class clickAtrasaRutina implements ActionListener {

	private JPanel panelPadre;
	private JPanel panelRutinas;
	private JPanel panelEncontrados;
	private String id;

	public clickAtrasaRutina(JPanel panelRutinas, JPanel panelEncontrados,String id, JPanel
panelPadre) {

		this.id=id;
		this.panelPadre=panelPadre;
		this.panelRutinas=panelRutinas;
		this.panelEncontrados=panelEncontrados;

		}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Conectamos a la base de datos para acceder al nombre del ejercicio


			new RutinaConcreta(panelRutinas,panelEncontrados,id,panelPadre);

	}

}
```

B.1.25 clickEliminarRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase encargada de eliminar una rutina de la base de datos

public class clickEliminarRutina implements ActionListener {

	private JPanel panelPadre=null;
	private JPanel rutinas=null;
	private Connection con;
	private int rs;
	private String id;

	public clickEliminarRutina(JPanel panelPadre, JPanel rutinas, String id) {
		this.panelPadre=panelPadre;
		this.rutinas=rutinas;
		this.id=id;
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar la rutina?","",
JOptionPane.YES_NO_OPTION);
		if(modificar==0)
		{
		try {
//Elimina la rutina y comprueba si  se ha eliminado correctamente mostrando un mensaje de error en caso
contrario
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			String comprobar= "DELETE FROM rutinas WHERE Id_Rutina='"+id+"'";
			rs = stat.executeUpdate(comprobar);
			if(rs==1)
				JOptionPane.showMessageDialog(null,"Rutina eliminada
correctamente","",JOptionPane.INFORMATION_MESSAGE);
			else
				JOptionPane.showMessageDialog(null,"Rutina no eliminada
correctamente","",JOptionPane.ERROR_MESSAGE);
			    con.close();
		} catch (ClassNotFoundException | SQLException i) {
			System.out.println("Error: " + i.getMessage());
		}
	//Vuelve a la ventana principal, a la pestaña 1
		Principal.Recargar(panelPadre,rutinas,1);
		}
	}
}
```

B.1.26 clickEjerciciosRutina.java

```
package Presentacion;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;
//Clase cuya funcion es cargar los ejercicios en dos grupos
//Estos grupos son los  que estan asignados a esta rutina y los que no
public class clickEjerciciosRutina implements ActionListener {

	private JPanel panelRutinas=null;
	private JPanel Rutina=null;
	private JPanel tabbedPane=null;
	private String id=null;
	private Connection con;
	private ResultSet rs;
	private int i;

	public clickEjerciciosRutina(JPanel panelRutinas, JPanel Rutina, String id, JPanel tabbedPane)
{

		this.panelRutinas=panelRutinas;
		this.Rutina=Rutina;
		this.id=id;
		this.tabbedPane=tabbedPane;


	}

	public clickEjerciciosRutina() {
	}


	@Override
	public void actionPerformed(ActionEvent e) {
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			//Consulta a la base de datos que identifica los ejercicios que estan asignados a
la rutina

			String seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio, e.Descripcion
Descripcion,e.Finalidad Finalidad, e.Duracion Duracion "
					+ " FROM ejerciciosderutinas er  INNER JOIN ejercicios e
ON e.Id_Ejercicio=er.Id_Ejercicio WHERE er.Id_Rutina="'"+id+"'";

			rs = stat.executeQuery(seleccionar);
			rs.last();
			i = rs.getRow();
```

```
                        rs.beforeFirst();
                        //Object donde almacenamos los ejercicios que si estan asignados
                        Object [][]si=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {
                                si[i][0]=rs.getString("Descripcion");
                                si[i][1]=rs.getString("Finalidad");
                                si[i][2]=rs.getString("Duracion");
                                i++;
                        }
                        //Consulta a la base de datos que identifica los ejercicios que no estan
asignados a la rutina
                        seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio, e.Descripcion
Descripcion,e.Finalidad Finalidad, e.Duracion Duracion"
                                        + " FROM ejercicios e WHERE e.Id_Ejercicio NOT IN "
                                        + "(SELECT rde.Id_Ejercicio FROM ejerciciosderutinas rde
WHERE rde.Id_Rutina='"+id+"')";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        //Object donde almacenamos los ejercicios que no estan asignados
                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("Descripcion");
                                no[i][1]=rs.getString("Finalidad");
                                no[i][2]=rs.getString("Duracion");
                                i++;

                        }
                        final String[] columnNames = {"Nombre",
                                        "Finalidad",
                        "Duracion"};

                        new EjerciciosdeRutinasEncontradas( panelRutinas,  Rutina,  si,
columnNames,no,tabbedPane,id,true);

                    con.close();

        } catch (ClassNotFoundException | SQLException i) {
                System.out.println("Error: " + i.getMessage());
        }

}
        //Metodo que se encarga de realizar una actualizacion de estos grupos
        public  void actualiza(JPanel panelRutinas, JPanel Rutina, JPanel tabbedPane, String
id2,boolean filtro) {


                try {
                        Class.forName("com.mysql.jdbc.Driver");
                        con =
```

```
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                        Statement stat = con.createStatement();

                        String seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio, e.Descripcion
Descripcion,e.Finalidad Finalidad, e.Duracion Duracion "
                                        + " FROM ejerciciosderutinas er  INNER JOIN ejercicios e
ON e.Id_Ejercicio=er.Id_Ejercicio WHERE er.Id_Rutina='"+id2+"'";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        Object [][]si=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                si[i][0]=rs.getString("Descripcion");
                                si[i][1]=rs.getString("Finalidad");
                                si[i][2]=rs.getString("Duracion");
                                i++;

                        }
                        seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio, e.Descripcion
Descripcion,e.Finalidad Finalidad, e.Duracion Duracion"
                                        + " FROM ejercicios e WHERE e.Id_Ejercicio NOT IN "
                                        + "(SELECT rde.Id_Ejercicio FROM ejerciciosderutinas rde
WHERE rde.Id_Rutina='"+id2+"')";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("Descripcion");
                                no[i][1]=rs.getString("Finalidad");
                                no[i][2]=rs.getString("Duracion");
                                i++;
                        }

                        final String[] columnNames = {"Nombre",
                                        "Finalidad",
                        "Duracion"};

                        new EjerciciosdeRutinasEncontradas( panelRutinas,  Rutina,  si,
columnNames,no,tabbedPane,id2,filtro);

                    con.close();

                } catch (ClassNotFoundException | SQLException i2) {
                        System.out.println("Error: " + i2.getMessage());
                }
        }
```

```
        //Metodo que se encarga de realizar una actualizacion con filtros
                public void actualizafiltra(JPanel panelrutina, JPanel rutina, JPanel tabbedPane2, String
id2, String text, String text2,
                                String text3) {
                        boolean esta=false;
                        i=0;
                        try {
                                Class.forName("com.mysql.jdbc.Driver");
                                con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                Statement stat = con.createStatement();

                                String seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio,
e.Descripcion Descripcion,e.Finalidad Finalidad, e.Duracion Duracion "
                                                + " FROM ejerciciosderutinas er  INNER JOIN
ejercicios e ON e.Id_Ejercicio=er.Id_Ejercicio WHERE er.Id_Rutina="'+id2+"'";

                                rs = stat.executeQuery(seleccionar);
                                rs.last();
                                i = rs.getRow();
                                rs.beforeFirst();
                                Object [][]si=new Object[i][3];
                                i=0;
                                while (rs.next())
                                {

                                        si[i][0]=rs.getString("Descripcion");
                                        si[i][1]=rs.getString("Finalidad");
                                        si[i][2]=rs.getString("Duracion");
                                        i++;

                                }

                                seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicio, e.Descripcion
Descripcion,e.Finalidad Finalidad, e.Duracion Duracion"
                                                + " FROM ejercicios e WHERE e.Id_Ejercicio NOT
IN "
                                                + "(SELECT rde.Id_Ejercicio FROM
ejerciciosderutinas rde WHERE rde.Id_Rutina="'+id2+"')";

                                rs = stat.executeQuery(seleccionar);
                                i=0;
                        while (rs.next())
                        {
                          esta=false;

                          if(!text.equals(""))
                          {
                                if(rs.getString("Descripcion").indexOf(text)==-1)
                                {
                                        if(!text.equals(""))
                                                if(!(rs.getString("Finalidad").indexOf(text)==-1))
                                                        esta=true;
                                                else
                                                        if(!text.equals(""))
                                                        if(!(rs.getString("Duracion").indexOf(text)==-1))
```

```
                                                esta=true;
                }
                else
                        esta=true;
}
if(!text2.equals(""))
{
                if(rs.getString("Descripcion").indexOf(text2)==-1)
                {
                        if(!text2.equals(""))
                                if(!(rs.getString("Finalidad").indexOf(text2)==-1))
                                        esta=true;
                                else
                                        if(!text2.equals(""))
                                        if(!(rs.getString("Duracion").indexOf(text2)==-1))
                                                esta=true;
                }
                else
                        esta=true;
}
if(!text3.equals(""))
{
                if(rs.getString("Descripcion").indexOf(text3)==-1)
                {
                        if(!text3.equals(""))
                                if(!(rs.getString("Finalidad").indexOf(text3)==-1))
                                        esta=true;
                                else
                                        if(!text.equals(""))
                                        if(!(rs.getString("Duracion").indexOf(text3)==-1))
                                                esta=true;
                }
                else
                        esta=true;
}

if(esta==true)
                i++;

                        }
                        rs = stat.executeQuery(seleccionar);
                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {
esta=false;

if(!text.equals(""))
{
                if(rs.getString("Descripcion").indexOf(text)==-1)
                {
                        if(!text.equals(""))
                                if(!(rs.getString("Finalidad").indexOf(text)==-1))
                                        esta=true;
                                else
                                        if(!text.equals(""))
                                        if(!(rs.getString("Duracion").indexOf(text)==-1))
```

```
                                                    esta=true;
                        }
                        else
                                esta=true;
                }
                if(!text2.equals(""))
                {
                        if(rs.getString("Descripcion").indexOf(text2)==-1)
                        {
                                if(!text2.equals(""))
                                        if(!(rs.getString("Finalidad").indexOf(text2)==-1))
                                                esta=true;
                                        else
                                                if(!text2.equals(""))
                                                if(!(rs.getString("Duracion").indexOf(text2)==-1))
                                                        esta=true;
                        }
                        else
                                esta=true;
                }
                if(!text3.equals(""))
                {
                        if(rs.getString("Descripcion").indexOf(text3)==-1)
                        {
                                if(!text3.equals(""))
                                        if(!(rs.getString("Finalidad").indexOf(text3)==-1))
                                                esta=true;
                                        else
                                                if(!text3.equals(""))
                                                if(!(rs.getString("Duracion").indexOf(text3)==-1))
                                                        esta=true;
                        }
                        else
                                esta=true;
                }
                if(esta==true)
                {
                        no[i][0]=rs.getString("Descripcion");
                                        no[i][1]=rs.getString("Finalidad");
                                        no[i][2]=rs.getString("Duracion");
                                        i++;
                }
                                }

                                final String[] columnNames = {"Nombre",
                                                "Finalidad",
                                "Duracion"};
                            con.close();

                                new EjerciciosdeRutinasEncontradas( panelrutina,  rutina,  si,
columnNames,no,tabbedPane2,id2,false);

                            con.close();
                        } catch (ClassNotFoundException | SQLException i2) {
                                System.out.println("Error: " + i2.getMessage());
                        }
                }
}
```

B.1.27 EjerciciosdeRutinasEncontradas.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
//Clase que muestra los dos grupos de ejercicios segun esten asignados o no y permite cambiarlo
public class EjerciciosdeRutinasEncontradas extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel Rutina;
	public JPanel panelRutinas;
	public JPanel panelEncontrados;
	public JPanel panelPadre;
	public String id;
	private Connection con;
	private ResultSet rs;
	private int prueba;


	public EjerciciosdeRutinasEncontradas(final JPanel panelRutina, JPanel Rutina, final Object[][]
ob, final String[] columnNames,final Object[][] ob2,final JPanel panelPadre, final String id,boolean
muestra_filtro) {
		this.panelRutinas=panelRutina;
		this.Rutina=Rutina;
		this.panelPadre=panelPadre;
		this.id=id;
		Rutina.setVisible(false);
		panelEncontrados = new JPanel();
		panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
		panelEncontrados.setBackground(new Color(51, 102, 153));
		panelEncontrados.setBorder(null);
		panelEncontrados.setLayout(null);
		String rutinastring=null;

		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
```

```
                        Statement stat = con.createStatement();


                        String comprobar= "SELECT * FROM rutinas WHERE Id_Rutina='"+id+"'";
                        rs = stat.executeQuery(comprobar);
                        rs.next();
                        rutinastring = "Rutina: "+rs.getString("descripcion");
        } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }
                JLabel rutinatxt = new JLabel(rutinastring);
                rutinatxt.setFont(new Font("Dialog", Font.BOLD, 20));
                rutinatxt.setBounds(2, 15, 500, 28);
                panelEncontrados.add(rutinatxt);

                JLabel asignadas = new JLabel("Ejercicios asignados:");
                asignadas.setFont(new Font("Dialog", Font.BOLD, 20));
                asignadas.setBounds(2, 50, 500, 28);
                panelEncontrados.add(asignadas);

                final MyTableModel myModel = new MyTableModel(ob,columnNames);
                final JTable table = new JTable(myModel);



                table.setSelectionBackground(new Color(0, 0, 255));
                table.setBackground(new Color(0, 102, 153));
                table.setBorder(null);
                table.setFont(new Font("Dialog", Font.BOLD, 15));

        //Creatamos un contenedor para la Tabla
                final JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setBounds(new Rectangle(2, 80, 600, 160));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);



        //Agregamos nuestra tabla al contenedor
        table.addMouseListener(new MouseAdapter()
        {
            public void mouseClicked(MouseEvent e)
            {
                int modificar;
                int fila = table.rowAtPoint(e.getPoint());
                int columna = table.columnAtPoint(e.getPoint());
                if ((fila > -1) && (columna > -1))
                {
                        modificar=JOptionPane.showConfirmDialog(null,"Desea desasignar el
ejercicio:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                        if(modificar==0)
                                try {
                                        Class.forName("com.mysql.jdbc.Driver");
                                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                        Statement stat = con.createStatement();
```

```
                                        String comprobar= "SELECT * FROM ejercicios WHERE
Descripcion='"+ob[fila][0]+"'";
                                        rs = stat.executeQuery(comprobar);
                                        rs.next();
                                        String idejer=rs.getString("Id_Ejercicio");

                                        comprobar= "DELETE FROM ejerciciosderutinas WHERE
Id_Ejercicio='"+idejer+"' and Id_Rutina='"+id+"'";
                                        prueba = stat.executeUpdate(comprobar);
                                        if(prueba==1)

            JOptionPane.showMessageDialog(null,"Desasignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                        else
                                                JOptionPane.showMessageDialog(null,"No
desasignado correctamente","",JOptionPane.ERROR_MESSAGE);
                                    con.close();

                                        clickEjerciciosRutina  b=new clickEjerciciosRutina();
                                        b.actualiza(panelRutinas,panelEncontrados, panelPadre
,id,true);
                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }
                        }
                }
        });


    JLabel noasignadas = new JLabel("Ejercicios no asignados:");
    noasignadas.setFont(new Font("Dialog", Font.BOLD, 20));
    noasignadas.setBounds(2, 245, 500, 28);
                panelEncontrados.add(noasignadas);


    final MyTableModel myModel2 = new MyTableModel(ob2,columnNames);
                final JTable table2 = new JTable(myModel2);

                table2.setSelectionBackground(new Color(0, 0, 255));
                table2.setBackground(new Color(0, 102, 153));
                table2.setBorder(null);
                table2.setFont(new Font("Dialog", Font.BOLD, 15));

    //Creatamos un contenedor para la Tabla
    final JScrollPane scrollPane2 = new JScrollPane(table2);
    scrollPane2.setBounds(new Rectangle(2, 275, 600, 180));
    scrollPane2.getViewport().setBackground(new Color(51, 102, 153));
    scrollPane2.getViewport().setBorder(null);

    //Agregamos nuestra tabla al contenedor
    table2.addMouseListener(new MouseAdapter()
    {
        public void mouseClicked(MouseEvent e)
        {
                int modificar;
                int fila = table2.rowAtPoint(e.getPoint());
                int columna = table2.columnAtPoint(e.getPoint());
```

```
                    if ((fila > -1) && (columna > -1))
                    {
                            modificar=JOptionPane.showConfirmDialog(null,"Desea añadir el
ejercicio:"+ob2[fila][0],"", JOptionPane.YES_NO_OPTION);
                            if(modificar==0)
                                    try {
                                            Class.forName("com.mysql.jdbc.Driver");
                con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                Statement stat = con.createStatement();
                String comprobar= "SELECT * FROM ejercicios WHERE Descripcion="'+ob2[fila][0]+"'";
                rs = stat.executeQuery(comprobar);
                rs.next();
                String idejer=rs.getString("Id_Ejercicio");

                comprobar= "INSERT INTO ejerciciosderutinas VALUES ("'+id+'","'+idejer+"')";
                prueba = stat.executeUpdate(comprobar);
                if(prueba==1)
                     JOptionPane.showMessageDialog(null,"Asignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                else
                     JOptionPane.showMessageDialog(null,"No asignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                clickEjerciciosRutina  b=new clickEjerciciosRutina();
                b.actualiza(panelRutinas,panelEncontrados, panelPadre ,id,true);
                        con.close();

                                    } catch (ClassNotFoundException | SQLException i) {
                                            System.out.println("Error: " + i.getMessage());
                                    }
                    }
            }
        });
        panelEncontrados.add(scrollPane);

            final JButton btnAtras = new JButton("Atrás");
                    btnAtras.setBounds(422, 520, 148, 39);
                    btnAtras.addActionListener((ActionListener) new
clickAtrasaRutina(panelRutinas,panelEncontrados,id,panelPadre));

                    panelEncontrados.add(btnAtras);

                            final JTextField campo;
                            campo = new JTextField();
                            campo.setBounds(30, 480,  147, 25);
                            panelEncontrados.add(campo);
                            campo.setColumns(10);

                            final JTextField campo1;
                            campo1 = new JTextField();
                            campo1.setBounds(226, 480,  147, 25);
                            campo.setColumns(10);
                            final JTextField campo2;
                            campo2 = new JTextField();
                            campo2.setBounds(422, 480,  147, 25);
                            campo2.setColumns(10);
```

```
                        final JLabel lblNewLabel1 = new JLabel("+");


                        final JLabel lblNewLabel = new JLabel("+");
                        lblNewLabel.setFont(new Font("Dialog", Font.BOLD, 20));
                        lblNewLabel.setBounds(226, 480, 20, 25);
                        panelEncontrados.add(lblNewLabel);
                        lblNewLabel.addMouseListener(new MouseAdapter() {
                                @Override
                                public void mouseClicked(MouseEvent e) {
                                        panelEncontrados.add(campo1);
                                        panelEncontrados.add(lblNewLabel1);
                                        panelEncontrados.remove(lblNewLabel);
                                        panelEncontrados.updateUI();
                                        panelRutina.updateUI();
                                }
                        });
                        lblNewLabel1.setFont(new Font("Dialog", Font.BOLD, 20));
                        lblNewLabel1.setBounds(422, 480, 40, 25);
                        lblNewLabel1.addMouseListener(new MouseAdapter() {
                                @Override
                                public void mouseClicked(MouseEvent e) {
                                        panelEncontrados.add(campo2);
                                        panelEncontrados.remove(lblNewLabel1);
                                        panelEncontrados.updateUI();
                                        panelRutina.updateUI();
                                }
                        });
                        final JButton btnFiltrar = new JButton("Filtrar");
                        btnFiltrar.setBounds(30, 520, 148, 39);
                        panelEncontrados.add(btnFiltrar);
                        btnFiltrar.addMouseListener(new MouseAdapter() {
                                @Override
                                public void mouseClicked(MouseEvent e) {
                                        clickEjerciciosRutina  b=new clickEjerciciosRutina();
                                        b.actualizafiltra(panelRutina,panelEncontrados, panelPadre
,id,campo.getText(),campo1.getText(),campo2.getText());
                                }
                        });
                panelEncontrados.add(scrollPane2);

                final JButton btnTodas = new JButton("Mostrar todos");
                btnTodas.setBounds(226, 520, 148, 39);
                panelEncontrados.add(btnTodas);
                btnTodas.addMouseListener(new MouseAdapter() {
                        @Override
                        public void mouseClicked(MouseEvent e) {
                                clickEjerciciosRutina  b=new clickEjerciciosRutina();
                                b.actualiza(panelRutina,panelEncontrados, panelPadre ,id,false);
                        }
                });

                panelRutina.add(panelEncontrados);

        }

}
```

B.1.28 clickAtrasaRutina.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


import javax.swing.JPanel;

public class clickAtrasaRutina implements ActionListener {

	private JPanel panelPadre;
	private JPanel panelRutinas;
	private JPanel panelEncontrados;
	private String id;

	public clickAtrasaRutina(JPanel panelRutinas, JPanel panelEncontrados,String id, JPanel
panelPadre) {

		this.id=id;
		this.panelPadre=panelPadre;
		this.panelRutinas=panelRutinas;
		this.panelEncontrados=panelEncontrados;

		}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Conectamos a la base de datos para acceder al nombre del ejercicio


			new RutinaConcreta(panelRutinas,panelEncontrados,id,panelPadre);

	}

}
```

B.1.29 clickCrearEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;

import javax.swing.JOptionPane;

//Clase encargada de crear los ejercicios añadiendo los campos opcionales si estos existen
public class clickCrearEjercicio implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private ResultSet rs2;
	private int id=0;
	private Boolean completo=false;
	private Boolean sal=false;
	private int continuar=0;

	public  clickCrearEjercicio() {
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		continuar=0;
		completo=false;
		//Comprobamos que los campos obligatorios estan completos
		if(Principal.campoNombreEjercicios.getText().equals("")==false &&
Principal.campoFinalidad.getText().equals("")==false
				&& Principal.campoDuracion.getText().equals("") ==false )
			completo=true;

		if(completo==false)
			JOptionPane.showMessageDialog(null,"Algún campo esta
vacío","",JOptionPane.WARNING_MESSAGE);
		else
		{
			//Comprobamos si los campos opcionales esta completos y en caso negativo
mostramos un aviso
			if(!(Principal.campoRecomendaciones.getText().equals("")==false &&
Principal.campoCaracteristicas.getText().equals("")==false
					& Principal.campoDuracion.getText().equals("") ==false ))
				continuar=JOptionPane.showConfirmDialog(null,"Va a crear un
ejercicio sin asignarle todos sus campos opcionales"
						+ "¿Desea continuar?","",
JOptionPane.YES_NO_OPTION);
			if(continuar==0)
			{
				try {

					Class.forName("com.mysql.jdbc.Driver");
					con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
					Statement stat = con.createStatement();
					String comprobar= "SELECT * FROM ejercicios WHERE
Descripcion='"+Principal.campoNombreEjercicios.getText()+"'";
```

```
                                        rs = stat.executeQuery(comprobar);
                                        //Comprobamos si existe un ejercicio con esa descripcion
basica/nombre
                                        if (rs.next())
                                                JOptionPane.showMessageDialog(null,"Nombre de
ejercicio en uso","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                rs.close();
                                                String seleccionar = "SELECT * FROM ejercicios";
                                                rs2 = stat.executeQuery(seleccionar);
                                                //buscamos el id libre mas bajo que este libre
                                                for( id=1;sal==false;id++)
                                                {

                                                        if(rs2.next())
                                                        {

        if(!rs2.getString("Id_Ejercicio").equals(""+id))
                                                                        sal=true;
                                                        }
                                                        else
                                                                sal=true;

                                                }

                                                id--;
                                                //Realizamos la insercion normal en la tabla
ejercicios
                                                String insertar= "INSERT INTO ejercicios VALUES
("'+id+"','"  + Principal.campoNombreEjercicios.getText() + "','"
                                                                        +
Principal.campoFinalidad.getText() + "','" + Principal.campoDuracion.getText() + "')";
                                                stat.executeUpdate(insertar);

                                                //Comprobamos cuales de los 3 campos opcionales
estan completos para añadir la entrada a sus tablas

        if(Principal.campoRecomendaciones.getText().equals("") ==false)
                                                {
                                                        insertar= "INSERT INTO
recomendacionesdeejercicios VALUES ("'+id+"','"  + id + "','"
                                                                        +
Principal.campoRecomendaciones.getText()  + "')";
                                                        stat.executeUpdate(insertar);
                                                }


        if(Principal.campoCaracteristicas.getText().equals("") ==false)
                                                {
                                                        insertar= "INSERT INTO
carasteristicasdeejercicios VALUES  ("'+id+"','"  + id + "','"
                                                                        +
Principal.campoCaracteristicas.getText()  + "')";
                                                        stat.executeUpdate(insertar);
                                                }
```

```
                                        if(Principal.campoExplicacion.getText().equals("")
==false)
                                        {
                                                insertar= "INSERT INTO
explicacionesdeejercicios VALUES ('"+id+"','"  + id + "','"
                                                        +
Principal.campoExplicacion.getText()  + "')";
                                                stat.executeUpdate(insertar);
                                        }
                                        //mostramos mensaje de añadido correctamente y
ponemos los campos vacios de nuevo
                                        JOptionPane.showMessageDialog(null,"Añadido
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                        Principal.campoNombreEjercicios.setText(null);
                                        Principal.campoFinalidad.setText(null);
                                        Principal.campoDuracion.setText(null);
                                        Principal.campoRecomendaciones.setText(null);
                                        Principal.campoCaracteristicas.setText(null);
                                        Principal.campoExplicacion.setText(null);

                                    }
                                con.close();

                            } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                            }
                        }
                    }
                }
            }
```

B.1.30 clickBorrarEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

//Clase encargada de vaciar todos los campos del formulario Ejercicio
public class clickBorrarEjercicio implements ActionListener {

	@Override
	public void actionPerformed(ActionEvent e) {

		Principal.campoNombreEjercicios.setText(null);
		Principal.campoFinalidad.setText(null);
		Principal.campoDuracion.setText(null);
		Principal.campoRecomendaciones.setText(null);
		Principal.campoCaracteristicas.setText(null);
		Principal.campoExplicacion.setText(null);

	}

}
```

B.1.31 clickBuscarEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;

//Clase encargada de buscar los ejercicios que cumplan las codiciones establecidas
//Primero comprueba si el campo esta vacio y despues en el caso de no estar vacio
//comprueba si existe un ejercicio con todo un campo con las palabras escritas en los campos
public class clickBuscarEjercicio implements ActionListener{

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicio;
	private JPanel Ejercicio;
	private JPanel panelPadre;
	private boolean esta=false;
	private boolean vacio=false;
	private String seleccionar;
	private int i=0;

	public clickBuscarEjercicio(JPanel panelEjercicio, JPanel Ejercicio, JPanel panelPadre){

		this.panelEjercicio=panelEjercicio;
	   this.Ejercicio=Ejercicio;
	   this.panelPadre=panelPadre;
		}
	  @Override
	  public void actionPerformed(ActionEvent e) {

		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			//Realizamos una consulta en la que realizamos join para unir hasta 4 tablas de
la base de datos
			seleccionar = "SELECT e.Id_Ejercicio Id_Ejercicios,e.Descripcion
Descripcion,e.Finalidad Finalidad,e.Duracion Duracion"
					+ ",r.descripcion recomendaciones,c.descripcion
caracteristicas, ex.descripcion explicacion "
					+ " FROM ejercicios e LEFT JOIN
recomendacionesdeejercicios r ON e.Id_Ejercicio=r.Id_Ejercicio "
					+ "LEFT JOIN carasteristicasdeejercicios c ON
e.Id_Ejercicio=c.Id_Ejercicio LEFT JOIN explicacionesdeejercicios ex "
					+ "ON e.Id_Ejercicio=ex.Id_Ejercicio";
			Statement stat = con.createStatement();
			rs = stat.executeQuery(seleccionar);
			vacio=false;
```

```
                        //Comprobamos si estan todos los campos vacios
                        if(!(Principal.campoNombreEjercicios.getText().equals("")==true &&
Principal.campoFinalidad.getText().equals("")==true
                                && Principal.campoDuracion.getText().equals("") ==true
&& Principal.campoRecomendaciones.getText().equals("")==true
                                &&
Principal.campoCaracteristicas.getText().equals("")==true &&
Principal.campoDuracion.getText().equals("") ==true))
                                while (rs.next())
                                {
//En caso de no estarlo comprobamos uno a uno todos los campos, para cada campo comprobamos que
cumpla todo
//los anteriores, y en caso de no estar vacio comprobamos si cumple los requisitos escritos
                                esta=true;
                                if(esta==true &&
!Principal.campoNombreEjercicios.getText().equals(""))

        if(rs.getString("Descripcion").indexOf(Principal.campoNombreEjercicios.getText())==-1)
                                        esta=false;

                                if(esta==true &&
!Principal.campoFinalidad.getText().equals(""))

        if(rs.getString("Finalidad").indexOf(Principal.campoFinalidad.getText())==-1)
                                        esta=false;

                                if(esta==true &&
!Principal.campoDuracion.getText().equals(""))

        if(rs.getString("Duracion").indexOf(Principal.campoDuracion.getText())==-1)
                                        esta=false;

                                if(esta==true &&
!Principal.campoRecomendaciones.getText().equals(""))

        if(rs.getString("recomendaciones").indexOf(Principal.campoRecomendaciones.getText())==-1)
                                        esta=false;

                                if(esta==true &&
!Principal.campoCaracteristicas.getText().equals(""))

        if(rs.getString("caracteristicas").indexOf(Principal.campoCaracteristicas.getText())==-1)
                                        esta=false;

                                if(esta==true &&
!Principal.campoDuracion.getText().equals(""))

        if(rs.getString("explicacion").indexOf(Principal.campoDuracion.getText())==-1)
                                        esta=false;

                                if(esta==true)
                                        i++;
                                }
                        else
                        {
        //En caso de no estar vacio damos por hecho que no sabemos que buscar por lo que mostramos
todo
```

```
                                vacio=true;
                                rs.last();
                                i = rs.getRow()+1;
                                rs.beforeFirst();
                        }
                        //Si no hay ningun ejercicio que cumple las condiciones mostramos ese
mensaje
                        if (i==0)
                                JOptionPane.showMessageDialog(null,"Ningún ejercicio contiene
esas palabras","",JOptionPane.WARNING_MESSAGE);
                        else{
                                //En caso contrario los almacenamos en un object
                                Object [][]ob=new Object[i][3];
                                i=0;
                                rs.beforeFirst();
                                while (rs.next())
                                {
                                        esta=false;
                                        if(vacio==false)
                                        {
                                                esta=true;
                                                if(esta==true &&
!Principal.campoNombreEjercicios.getText().equals(""))

        if(rs.getString("Descripcion").indexOf(Principal.campoNombreEjercicios.getText())==-1)
                                                                esta=false;

                                                if(esta==true &&
!Principal.campoFinalidad.getText().equals(""))

        if(rs.getString("Finalidad").indexOf(Principal.campoFinalidad.getText())==-1)
                                                                esta=false;

                                                if(esta==true &&
!Principal.campoDuracion.getText().equals(""))

        if(rs.getString("Duracion").indexOf(Principal.campoDuracion.getText())==-1)
                                                                esta=false;

                                                if(esta==true &&
!Principal.campoRecomendaciones.getText().equals(""))

        if(rs.getString("recomendaciones").indexOf(Principal.campoRecomendaciones.getText())==-1)
                                                                esta=false;

                                                if(esta==true &&
!Principal.campoCaracteristicas.getText().equals(""))

        if(rs.getString("caracteristicas").indexOf(Principal.campoCaracteristicas.getText())==-1)
                                                                esta=false;

                                                if(esta==true &&
!Principal.campoDuracion.getText().equals(""))

        if(rs.getString("explicacion").indexOf(Principal.campoDuracion.getText())==-1)
                                                                esta=false;
                                        }
```

```
                                        else
                                                esta=true;
                                        if(esta==true)
                                        {
                                                ob[i][0]=rs.getString("Descripcion");
                                                ob[i][1]=rs.getString("Finalidad");
                                                ob[i][2]=rs.getString("Duracion");
                                                i++;
                                        }
                                        final String[] columnNames = {"Nombre",
                                                        "Finalidad",
                                        "Duracion"};
                                        new
EjerciciosEncontrados(panelEjercicio,Ejercicio,ob,panelPadre,columnNames);

                                }
                        }
                            con.close();

                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

        }
}
```

B.1.32 EjercicioConcreto.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
//Clase que muestra la unformacion del ejecericio concreto y que permite interactuar con el
public class EjercicioConcreto extends JFrame {

	private static final long serialVersionUID = 1L;
	private String ejerciciostring;
	private Connection con;
	private ResultSet rs;
	private ResultSet rs1;
	private ResultSet rs2;
	private ResultSet rs3;
	public static JTextField  campoNombreEjercicios;
	public static JTextField  campoFinalidad;
	public static JTextField  campoDuracion;
	public static JTextArea  campoRecomendaciones;
	public static JTextArea  campoCaracteristicas;
	public static JTextArea campoExplicacion;


	public EjercicioConcreto(JPanel panelEjerciciospadre, JPanel panelEncontrados, Object ob,
JPanel panelPadre) {
		ejerciciostring=ob.toString();
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();

			String comprobar= "SELECT * FROM ejercicios WHERE
Descripcion='"+ejerciciostring+"'";
			String seleccionar;
			rs = stat.executeQuery(comprobar);
			rs.next();
			String id=rs.getString("Id_Ejercicio");


			panelEjerciciospadre.removeAll();

			JPanel panelEjercicios = new JPanel();
```

```
panelEjercicios.setBackground(new Color(51, 102, 153));
panelEjercicios.setBounds(0, 0, 650, 582);
panelEjerciciospadre.add(panelEjercicios);
panelEjercicios.setLayout(null);

JPanel Ejercicios = new JPanel();
Ejercicios.setBounds(0, 0, 650, 582);
panelEjercicios.add(Ejercicios);
Ejercicios.setBackground(new Color(51, 102, 153));
Ejercicios.setLayout(null);

campoNombreEjercicios = new JTextField();
campoNombreEjercicios.setBounds(231, 30, 167, 25);
Ejercicios.add(campoNombreEjercicios);
campoNombreEjercicios.setColumns(10);
campoNombreEjercicios.setText(rs.getString("Descripcion"));

campoFinalidad = new JTextField();
campoFinalidad.setColumns(10);
campoFinalidad.setBounds(231, 80, 167, 25);
Ejercicios.add(campoFinalidad);
campoFinalidad.setText(rs.getString("Finalidad"));


campoDuracion = new JTextField();
campoDuracion.setColumns(10);
campoDuracion.setBounds(231, 130, 167, 25);
Ejercicios.add(campoDuracion);
campoDuracion.setText(rs.getString("Duracion"));

campoRecomendaciones = new JTextArea();
campoRecomendaciones.setColumns(10);
campoRecomendaciones.setLineWrap(true);
JScrollPane scroll1 = new JScrollPane(campoRecomendaciones);
scroll1.setBounds(110, 215, 288, 75);
campoRecomendaciones.setWrapStyleWord(true);
Ejercicios.add(scroll1);
seleccionar = "SELECT * FROM recomendacionesdeejercicios WHERE
Id_Ejercicio="'+id +"'";
rs3 = stat.executeQuery(seleccionar);
if(rs3.next())
        campoRecomendaciones.setText(rs3.getString("descripcion"));


campoCaracteristicas = new JTextArea();
campoCaracteristicas.setColumns(10);
campoCaracteristicas.setLineWrap(true);
JScrollPane scroll2 = new JScrollPane(campoCaracteristicas);
scroll2.setBounds(110, 340, 288, 75);
campoCaracteristicas.setWrapStyleWord(true);
Ejercicios.add(scroll2);
seleccionar = "SELECT * FROM carasteristicasdeejercicios WHERE
Id_Ejercicio="'+id +"'";
rs1 = stat.executeQuery(seleccionar);
if(rs1.next())
        campoCaracteristicas.setText(rs1.getString("descripcion"));
```

```
campoExplicacion = new JTextArea();
campoExplicacion.setColumns(10);
campoExplicacion.setLineWrap(true);
JScrollPane scroll3 = new JScrollPane(campoExplicacion);
scroll3.setBounds(110, 465, 288, 75);
campoExplicacion.setWrapStyleWord(true);
seleccionar = "SELECT * FROM explicacionesdeejercicios WHERE
Id_Ejercicio="'+id+"'";

Ejercicios.add(scroll3);
rs2 = stat.executeQuery(seleccionar);
if(rs2.next())
        campoExplicacion.setText(rs2.getString("descripcion"));


JLabel txtNombreEjercicios = new JLabel("Nombre:");
txtNombreEjercicios.setFont(new Font("Dialog", Font.BOLD, 18));
txtNombreEjercicios.setBounds(72, 35, 94, 15);
Ejercicios.add(txtNombreEjercicios);

JLabel txtFinalidad = new JLabel("Finalidad:");
txtFinalidad.setFont(new Font("Dialog", Font.BOLD, 18));
txtFinalidad.setBounds(72, 85, 120, 15);
Ejercicios.add(txtFinalidad);

JLabel txtDuracion = new JLabel("Duración:");
txtDuracion.setFont(new Font("Dialog", Font.BOLD, 18));
txtDuracion.setBounds(72, 135, 120, 15);
Ejercicios.add(txtDuracion);

JLabel txtRecomendaciones = new JLabel("Recomendación:");
txtRecomendaciones.setFont(new Font("Dialog", Font.BOLD, 18));
txtRecomendaciones.setBounds(72, 185, 200, 15);
Ejercicios.add(txtRecomendaciones);

JLabel txtCaracteristicas = new JLabel("Características:");
txtCaracteristicas.setFont(new Font("Dialog", Font.BOLD, 18));
txtCaracteristicas.setBounds(72, 310, 200, 15);
Ejercicios.add(txtCaracteristicas);

JLabel txtExplicacion = new JLabel("Explicación:");
txtExplicacion.setFont(new Font("Dialog", Font.BOLD, 18));
txtExplicacion.setBounds(72, 435, 200, 15);
Ejercicios.add(txtExplicacion);


JButton btnGuardarEjercicios = new JButton("Guardar");
btnGuardarEjercicios.setBounds(449, 20, 130, 39);
Ejercicios.add(btnGuardarEjercicios);

JButton btnDeshacerEjercicio = new JButton("Deshacer");
btnDeshacerEjercicio.setBounds(449, 100, 130, 39);
Ejercicios.add(btnDeshacerEjercicio);
```

```
                        JButton btnAtrasEjercicio = new JButton("Atrás");
                        btnAtrasEjercicio.setBounds(449, 180, 130, 39);
                        Ejercicios.add(btnAtrasEjercicio);

                        JButton btnUtiles = new JButton("Útiles");
                        btnUtiles.setBounds(449, 340, 130, 39);
                        Ejercicios.add(btnUtiles);

                        JButton btnEliminarEjercicio = new JButton("Eliminar");
                        btnEliminarEjercicio.setBounds(449, 260, 130, 39);
                        Ejercicios.add(btnEliminarEjercicio);

                        JButton btnMusculos = new JButton("Músculos");
                        btnMusculos.setBounds(449, 420, 130, 39);
                        Ejercicios.add(btnMusculos);

                        JButton btnRecursos = new JButton("Recursos");
                        btnRecursos.setBounds(449, 500, 130, 39);
                        Ejercicios.add(btnRecursos);

                        btnGuardarEjercicios.addActionListener((ActionListener) new
clickGuardarEjercicio(id));
                        btnDeshacerEjercicio.addActionListener((ActionListener) new
clickDeshacerEjercicio(id));
                        btnAtrasEjercicio.addActionListener((ActionListener) new
clickAtrasEjercicio(panelEjercicios, Ejercicios));
                        btnEliminarEjercicio.addActionListener((ActionListener) new
clickEliminarEjercicio(panelEjercicios, Ejercicios,id));
                        btnUtiles.addActionListener((ActionListener) new clickUtiles(panelEjercicios,
Ejercicios, id,panelPadre));
                        btnMusculos.addActionListener((ActionListener) new
clickMusculos(panelEjercicios, Ejercicios, id,panelPadre));
                        btnRecursos.addActionListener((ActionListener) new
clickRecursos(panelEjercicios, Ejercicios, id,panelPadre));

                    con.close();

                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

        }
}
```

B.1.33 clickGuardarEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;

import javax.swing.JOptionPane;
//Clase encargada de guardar los cambios en la base de datos
public class clickGuardarEjercicio implements ActionListener {


	private Connection con;
	private ResultSet rs;
	private String id;
	private String ejerciciostring;
	private String insertar;

	public clickGuardarEjercicio(String id) {

		this.id=id;

	}


	@Override
	public void actionPerformed(ActionEvent e) {
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea guardar los
cambios?","", JOptionPane.YES_NO_OPTION);
	//Tras confirmar que se desea guardar los cambios, diferenciamos dos casos
		if(modificar==0)
		{

			try {
				Class.forName("com.mysql.jdbc.Driver");
				con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
				Statement stat = con.createStatement();
				String comprobar= "SELECT * FROM ejercicios WHERE
Id_Ejercicio='"+id+"'";
				rs = stat.executeQuery(comprobar);
				rs.next();
				ejerciciostring=rs.getString("Descripcion");
			//Caso uno , no se desea cambiar el nombre del ejercicio, se procede a guardar
los demas cambios

	if(ejerciciostring.equals(EjercicioConcreto.campoNombreEjercicios.getText()))
					{

					String cambio="UPDATE ejercicios SET Descripcion ='"+
EjercicioConcreto.campoNombreEjercicios.getText()+"', Finalidad='"+
EjercicioConcreto.campoFinalidad.getText()+"', Duracion='"
							+
EjercicioConcreto.campoDuracion.getText()+"' WHERE Id_Ejercicio='"+id+"'";
					stat.executeUpdate(cambio);

					//Se comprueba cada campo opcional si tenia algo escrito en
```

cuyo caso

```
        if(EjercicioConcreto.campoRecomendaciones.getText().equals("") ==false)
                                        {
                                                //Se comprueba si ya existia para
                                                comprobar= "SELECT * FROM
recomendacionesdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                rs = stat.executeQuery(comprobar);
                                                if (rs.next())
                                                {
                                                        //actualizar
                                                        cambio= "UPDATE
recomendacionesdeejercicios SET descripcion ='"+
EjercicioConcreto.campoRecomendaciones.getText()+"'  WHERE Id_Ejercicio='"+id+"'";
                                                        stat.executeUpdate(cambio);
                                                }
                                                else
                                                {
                                                        //o crear segun sea
                                                        insertar= "INSERT INTO
recomendacionesdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoRecomendaciones.getText()  + "')";
                                                        stat.executeUpdate(insertar);
                                                }
                                        }


        if(EjercicioConcreto.campoCaracteristicas.getText().equals("") ==false)
                                        {
                                                comprobar= "SELECT * FROM
carasteristicasdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                rs = stat.executeQuery(comprobar);

                                                if (rs.next())
                                                {
                                                        cambio= "UPDATE
carasteristicasdeejercicios SET descripcion ='"+ EjercicioConcreto.campoCaracteristicas.getText()+"'
WHERE Id_Ejercicio='"+id+"'";
                                                        stat.executeUpdate(cambio);

                                                }
                                                else
                                                {
                                                        System.out.println("else");
                                                        insertar= "INSERT INTO
carasteristicasdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoCaracteristicas.getText()  + "')";
                                                        stat.executeUpdate(insertar);
                                                }
                                        }

                                        if(EjercicioConcreto.campoExplicacion.getText().equals("")
==false)
                                        {
                                                comprobar= "SELECT * FROM
explicacionesdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                rs = stat.executeQuery(comprobar);
                                                if (rs.next())
```

```
                                                {
                                                        cambio= "UPDATE
explicacionesdeejercicios SET descripcion ='"+ EjercicioConcreto.campoExplicacion.getText()+"'
WHERE Id_Ejercicio='"+id+"'";
                                                        stat.executeUpdate(cambio);
                                                }
                                                else
                                                {
                                                        insertar= "INSERT INTO
explicacionesdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoExplicacion.getText()  + "')";
                                                        stat.executeUpdate(insertar);
                                                }
                                        }

                                        JOptionPane.showMessageDialog(null,"Cambios guardados
correctamente","",JOptionPane.INFORMATION_MESSAGE);

                                        }
                                else
                                {
                                //Caso dos, se desea cambiar el nombre del ejercicio, antes de guardar
se comprueba que ese nombre este libre mostrando un mensaje en caso contrario
                                        comprobar= "SELECT * FROM ejercicios WHERE
Descripcion='"+EjercicioConcreto.campoNombreEjercicios.getText()+"'";
                                        rs = stat.executeQuery(comprobar);
                                        if (rs.next())
                                                JOptionPane.showMessageDialog(null,"Si desea
cambiar el nombre del ejercicio debe usar uno libre","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                String cambio="UPDATE ejercicios SET
Descripcion ='"+ EjercicioConcreto.campoNombreEjercicios.getText()+"', Finalidad='"+
EjercicioConcreto.campoFinalidad.getText()+"', Finalidad='"
                                                                                        +
EjercicioConcreto.campoDuracion.getText()+"' WHERE Id_Ejercicio='"+id+"'";
                                                stat.executeUpdate(cambio);

        if(EjercicioConcreto.campoRecomendaciones.getText().equals("") ==false)
                                                {
                                                        comprobar= "SELECT * FROM
recomendacionesdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                        rs = stat.executeQuery(comprobar);
                                                        if (rs.next())
                                                        {
                                                                cambio= "UPDATE
recomendacionesdeejercicios SET descripcion ='"+
EjercicioConcreto.campoRecomendaciones.getText()+"'  WHERE Id_Ejercicio='"+id+"'";
                                                                stat.executeUpdate(cambio);
                                                        }
                                                        else
                                                        {
                                                                insertar= "INSERT INTO
recomendacionesdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoRecomendaciones.getText()  + "')";
                                                                stat.executeUpdate(insertar);
                                                        }
                                                }
```

```
            if(EjercicioConcreto.campoCaracteristicas.getText().equals("") ==false)
                                                {
                                                        comprobar= "SELECT * FROM
caracteristicasdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                        rs = stat.executeQuery(comprobar);
                                                        if (rs.next())
                                                        {
                                                                cambio= "UPDATE
caracteristicasdeejercicios SET descripcion ='"+ EjercicioConcreto.campoCaracteristicas.getText()+"'
WHERE Id_Ejercicio='"+id+"'";
                                                                stat.executeUpdate(cambio);
                                                        }
                                                        else
                                                        {
                                                                insertar= "INSERT INTO
caracteristicasdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoCaracteristicas.getText()  + "')";
                                                                stat.executeUpdate(insertar);
                                        }
                                                }
                                        if(EjercicioConcreto.campoExplicacion.getText().equals("") ==false)
                                                {
                                                        comprobar= "SELECT * FROM
explicacionesdeejercicios WHERE Id_Ejercicio='"+id+"'";
                                                        rs = stat.executeQuery(comprobar);
                                                        if (rs.next())
                                                        {
                                                                cambio= "UPDATE
explicacionesdeejercicios SET descripcion ='"+ EjercicioConcreto.campoExplicacion.getText()+"'
WHERE Id_Ejercicio='"+id+"'";
                                                                stat.executeUpdate(cambio);
                                                        }
                                                        else
                                                        {
                                                                insertar= "INSERT INTO
explicacionesdeejercicios VALUES ('"+id+"','"  + id + "','" +
EjercicioConcreto.campoExplicacion.getText()  + "')";
                                                                stat.executeUpdate(insertar);
                                                        }
                                                }

                                                        JOptionPane.showMessageDialog(null,"Cambios
guardados correctamente","",JOptionPane.INFORMATION_MESSAGE);

                                                }
                                        }
                                    con.close();

                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }
                        }
                }
}
```

B.1.34 clickDeshacerEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
//Clase encargada de volver a rellenar el formulario con los datos de la base de datos
public class clickDeshacerEjercicio implements ActionListener {


	private Connection con;
	private ResultSet rs;
	private ResultSet rs1;
	private ResultSet rs2;
	private ResultSet rs3;
	private String id;
	private String seleccionar;


	public clickDeshacerEjercicio(String id) {

		this.id=id;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
	//Pedimos confirmacion de deshacer los cambios y realizamos una nueva consulta a la base de datos
		int modificar=JOptionPane.showConfirmDialog(null,"¿Desea cancelar los cambios?","", JOptionPane.YES_NO_OPTION);
		if(modificar==0)
		{
			try {
				Class.forName("com.mysql.jdbc.Driver");
				con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
				Statement stat = con.createStatement();
				String cargar= "SELECT * FROM ejercicios WHERE Id_Ejercicio="'+ id +"'";

				rs = stat.executeQuery(cargar);
				if (rs.next())
				{
				//tras obtener los datos los colocamos de nuevo en los campos correspondientes

	EjercicioConcreto.campoNombreEjercicios.setText(rs.getString("Descripcion"));

	EjercicioConcreto.campoFinalidad.setText(rs.getString("Finalidad"));
```

```
            EjercicioConcreto.campoDuracion.setText(rs.getString("Duracion"));
                                    //Comprobamos si este ejercicio tenia campos opcionales y si
es asi los colocamos tambien
                                    seleccionar = "SELECT * FROM
recomendacionesdeejercicios WHERE Id_Ejercicio="'+id +"'";
                                    rs3 = stat.executeQuery(seleccionar);
                                    if(rs3.next())

            EjercicioConcreto.campoRecomendaciones.setText(rs3.getString("descripcion"));
                            else
                                    EjercicioConcreto.campoRecomendaciones.setText("");
                                    seleccionar = "SELECT * FROM carasteristicasdeejercicios
WHERE Id_Ejercicio="'+id +"'";

                                    rs2 = stat.executeQuery(seleccionar);
                                    if(rs2.next())

            EjercicioConcreto.campoCaracteristicas.setText(rs2.getString("descripcion"));
                                    else

            EjercicioConcreto.campoCaracteristicas.setText("");
                                    seleccionar = "SELECT * FROM explicacionesdeejercicios
WHERE Id_Ejercicio="'+id+"'";

                                    rs1 = stat.executeQuery(seleccionar);
                                    if(rs1.next())

            EjercicioConcreto.campoExplicacion.setText(rs1.getString("descripcion"));
                                    else
                                            EjercicioConcreto.campoExplicacion.setText("");

                                }
                            con.close();

                        } catch (ClassNotFoundException | SQLException i) {
                            System.out.println("Error: " + i.getMessage());
                        }
                    }
            }
}
```

B.1.35 clickAtrasEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JPanel;

//Clase encargada de volver a la ventana inicial de Ejercicio
public class clickAtrasEjercicio implements ActionListener {

	private JPanel panelPadre=null;
	private JPanel ejercicios=null;

	public clickAtrasEjercicio(JPanel panelPadre, JPanel ejercicios) {

		this.panelPadre=panelPadre;
		this.ejercicios=ejercicios;

	}

	@Override
	public void actionPerformed(ActionEvent e) {

		Principal.Recargar(panelPadre,ejercicios,2);

	}
}
```

B.1.36 clickEliminarEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;

import java.sql.ResultSet;

//Clase encargada de eliminar un ejercicio de la base de datos
public class clickEliminarEjercicio implements ActionListener {

	private JPanel panelPadre=null;
	private JPanel ejercicios=null;

	private Connection con;
	private int res=0;
	private ResultSet rs;
	private String id;
	private String eliminar;
	private String seleccionar;
	int resv=0;



   public clickEliminarEjercicio(JPanel panelPadre, JPanel ejercicios, String id) {

	this.panelPadre=panelPadre;
		this.id=id;
		this.ejercicios=ejercicios;

   }

   @Override
   public void actionPerformed(ActionEvent e) {
	int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar el ejercicio?","",
JOptionPane.YES_NO_OPTION);
		if(modificar==0)
		{
	try {
		Class.forName("com.mysql.jdbc.Driver");
		con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
		Statement stat = con.createStatement();

		//Para cada campo opcional comprueba si existe y en caso de existir lo elimina
		seleccionar = "SELECT * FROM recomendacionesdeejercicios WHERE
Id_Ejercicio='"+id +"'";
		rs = stat.executeQuery(seleccionar);
		if(rs.next())
		{
			eliminar="DELETE FROM recomendacionesdeejercicios WHERE
```

```
Id_Ejercicio="'+id+"'";
                                res = stat.executeUpdate(eliminar)+res;
                                //Contaviliza las respuestas del executeUpdate y lo que deberia ser como
respuesta
                                resv++;
                        }

                        seleccionar = "SELECT * FROM carasteristicasdeejercicios WHERE
Id_Ejercicio="'+id +"'";
                        rs = stat.executeQuery(seleccionar);
                        if(rs.next())
                        {
                                eliminar="DELETE FROM carasteristicasdeejercicios WHERE
Id_Ejercicio="'+id+"'";
                                res = stat.executeUpdate(eliminar)+res;
                                resv++;
                }

                        seleccionar = "SELECT * FROM explicacionesdeejercicios WHERE Id_Ejercicio="'+id
+"'";
                        rs = stat.executeQuery(seleccionar);
                        if(rs.next())
                        {
                                eliminar="DELETE FROM explicacionesdeejercicios WHERE
Id_Ejercicio="'+id+"'";
                   res = stat.executeUpdate(eliminar)+res;
                   resv++;
                        }
                //Elimina los campos obligatorios y comprueba si todo se ha eliminado correctamente
                        resv++;
                        eliminar= "DELETE FROM ejercicios WHERE Id_Ejercicio="'+id+"'";
                        res = stat.executeUpdate(eliminar)+res;
                        if(res==resv)
                                JOptionPane.showMessageDialog(null,"Ejercicio eliminado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                        else
                                JOptionPane.showMessageDialog(null,"Ejercicio no eliminado
correctamente","",JOptionPane.ERROR_MESSAGE);

                            con.close();

                   } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                   }
                   //Vuelve a la ventana principal, a la pestaña 2
                        Principal.Recargar(panelPadre,ejercicios,2);
                        }
           }
}
```

B.1.37 clickUtiles.java

```
package Presentacion;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;

//Clase cuya funcion principal es cargar los utiles en dos grupos
//Estos grupos son los utiles que estan asignados al ejercicio y los que no
public class clickUtiles implements ActionListener {

	private JPanel panelEjercicios=null;
	private JPanel Ejercicios=null;
	private JPanel panelPadre=null;
	private String id=null;
	private Connection con;
	private ResultSet rs;
	private int i;

   public clickUtiles(JPanel panelEjercicios, JPanel Ejercicios, String id, JPanel panelPadre) {

	this.panelEjercicios=panelEjercicios;
		this.Ejercicios=Ejercicios;
		this.id=id;
		this.panelPadre=panelPadre;


   }


	public clickUtiles() {
	}


	@Override
	public void actionPerformed(ActionEvent e) {
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			//Consulta a la base de datos que identifica los utiles que estan asignados al
ejercicio
			String seleccionar = "SELECT u.Id_Util Id_Util, u.nombre nombre
,u.descripcion descripcion , u.urlfoto urlfoto "
					+ " FROM utilesdeejercicios ue  INNER JOIN utiles u ON
ue.Id_Util=u.Id_Util WHERE ue.Id_Ejercicio='"+id+"'";

			rs = stat.executeQuery(seleccionar);
			rs.last();
			i = rs.getRow();
```

```
                        rs.beforeFirst();
                        //Object donde almacenamos los utiles que si estan asignados
                        Object [][]si=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                si[i][0]=rs.getString("nombre");
                                si[i][1]=rs.getString("descripcion");
                                si[i][2]=rs.getString("urlfoto");
                                i++;
                        }
                        //Consulta a la base de datos que identifica los utiles que no estan asignados al
ejercicio

                        seleccionar = "SELECT u.Id_Util Id_Util, u.nombre nombre ,u.descripcion
descripcion , u.urlfoto urlfoto "
                                        + " FROM utiles u WHERE u.Id_Util NOT IN "
                                        + "(SELECT ue.Id_Util FROM utilesdeejercicios ue
WHERE ue.Id_Ejercicio='"+id+"')";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        //Object donde almacenamos los utiles que no estan asignados
                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("nombre");
                                no[i][1]=rs.getString("descripcion");
                                no[i][2]=rs.getString("urlfoto");
                                i++;

                        }
                        //Nombres de las columnas que queremos que se muestren
                        final String[] columnNames = {"Nombre",
                                        "Descripcion",
                        "Ruta"};

                        new UtilesEjercicioEncontrado( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id);
                    con.close();


                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }
        }

//Metodo que se encarga de realizar una actualizacion de estos grupos
        public  void actualiza(JPanel panelEjercicios, JPanel Ejercicios, JPanel panelPadre, String id2) {


                try {
```

```
                        Class.forName("com.mysql.jdbc.Driver");
                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
        Statement stat = con.createStatement();

        String seleccionar = "SELECT u.Id_Util Id_Util, u.nombre nombre ,u.descripcion descripcion ,
u.urlfoto urlfoto "
                        + " FROM utilesdeejercicios ue  INNER JOIN utiles u ON ue.Id_Util=u.Id_Util
WHERE ue.Id_Ejercicio='"+id2+"'";

        rs = stat.executeQuery(seleccionar);
        rs.last();
        i = rs.getRow();
        rs.beforeFirst();
        Object [][]si=new Object[i][3];
        i=0;
        while (rs.next())
        {
          si[i][0]=rs.getString("nombre");
          si[i][1]=rs.getString("descripcion");
          si[i][2]=rs.getString("urlfoto");
          i++;
        }

        seleccionar = "SELECT u.Id_Util Id_Util, u.nombre nombre ,u.descripcion descripcion , u.urlfoto
urlfoto "
                        + " FROM utiles u WHERE u.Id_Util NOT IN "
                        + "(SELECT ue.Id_Util FROM utilesdeejercicios ue  WHERE
ue.Id_Ejercicio='"+id2+"')";
        rs = stat.executeQuery(seleccionar);
        rs.last();
        i = rs.getRow();
        rs.beforeFirst();
        Object [][]no=new Object[i][3];
        i=0;
        while (rs.next())
        {

          no[i][0]=rs.getString("nombre");
          no[i][1]=rs.getString("descripcion");
          no[i][2]=rs.getString("urlfoto");
          i++;
        }
        final String[] columnNames = {"Nombre",
                        "Descripcion",
        "Ruta"};

        new UtilesEjercicioEncontrado( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id2);
                        con.close();


                } catch (ClassNotFoundException | SQLException i2) {
                        System.out.println("Error: " + i2.getMessage());
                }
          }
}
```

B.1.38 UtilesEjercicioEncontrado.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
//Clase que muestra los dos grupos de utiles segun esten asignados o no y permite cambiarlo
//Ademas de crear y eliminarlos

public class UtilesEjercicioEncontrado extends JFrame {

	private static final long serialVersionUID = 1L;
	public JPanel rutina;
   public JPanel panelrutinas;
   public JPanel panelEncontrados;
   public JPanel tabbedPane;
   public String id;
   private Connection con;
   private ResultSet rs;
   private int prueba;

	public UtilesEjercicioEncontrado(JPanel panelrutina, JPanel rutina, final Object[][] ob, final
String[] columnNames,final Object[][] ob2,final JPanel tabbedPane, final String id) {
		this.panelrutinas=panelrutina;
      this.rutina=rutina;
      this.tabbedPane=tabbedPane;
      this.id=id;
      rutina.setVisible(false);
      panelEncontrados = new JPanel();
      panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      String ejerciciostring=null;
			try {
				Class.forName("com.mysql.jdbc.Driver");
				con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
				Statement stat = con.createStatement();
```

```
                        String comprobar= "SELECT * FROM ejercicios WHERE
Id_Ejercicio='"+id+"'";
                        rs = stat.executeQuery(comprobar);
                        rs.next();
                        ejerciciostring = "Ejercicio: "+rs.getString("Descripcion");
            } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                    }

                    JLabel ejerciciotxt = new JLabel(ejerciciostring);
                    ejerciciotxt.setFont(new Font("Dialog", Font.BOLD, 20));
                    ejerciciotxt.setBounds(2, 15, 500, 28);
                    panelEncontrados.add(ejerciciotxt);

        JLabel asignadas = new JLabel("Utiles asignados:");
        asignadas.setFont(new Font("Dialog", Font.BOLD, 20));
        asignadas.setBounds(2, 50, 500, 28);
                    panelEncontrados.add(asignadas);

        final MyTableModel myModel = new MyTableModel(ob,columnNames);
                    final JTable table = new JTable(myModel);



                    table.setSelectionBackground(new Color(0, 0, 255));
                    table.setBackground(new Color(0, 102, 153));
                    table.setBorder(null);
                    table.setFont(new Font("Dialog", Font.BOLD, 15));

        //Creatamos un contenedor para la Tabla
        final JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setBounds(new Rectangle(2, 80, 600, 150));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);
        table.addMouseListener(new MouseAdapter()
        {
          public void mouseClicked(MouseEvent e)
          {
              int modificar;
            int fila = table.rowAtPoint(e.getPoint());
            int columna = table.columnAtPoint(e.getPoint());
            if ((fila > -1) && (columna > -1))
            {
              modificar=JOptionPane.showConfirmDialog(null,"Desea desasignar el útil:"+ob[fila][0],"",
JOptionPane.YES_NO_OPTION);
             if(modificar==0)
                        try {
                Class.forName("com.mysql.jdbc.Driver");
                con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                Statement stat = con.createStatement();



                String comprobar= "SELECT * FROM utiles WHERE nombre='"+ob[fila][0]+"'";
                rs = stat.executeQuery(comprobar);
                rs.next();
                String idutil=rs.getString("Id_Util");
```

```
                comprobar= "DELETE FROM utilesdeejercicios WHERE Id_Ejercicio='"+id+"' and
Id_Util='"+idutil+"'";
                prueba = stat.executeUpdate(comprobar);
                if(prueba==1)
                    JOptionPane.showMessageDialog(null,"Desasignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                else
                    JOptionPane.showMessageDialog(null,"No desasignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                clickUtiles  b=new clickUtiles();
                b.actualiza(panelrutinas,panelEncontrados, tabbedPane ,id);
                    con.close();

                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }
          }
        }
      });
      JLabel noasignadas = new JLabel("Utiles no asignados:");
      noasignadas.setFont(new Font("Dialog", Font.BOLD, 20));
      noasignadas.setBounds(2, 235, 500, 28);
                panelEncontrados.add(noasignadas);


      final MyTableModel myModel2 = new MyTableModel(ob2,columnNames);
                final JTable table2 = new JTable(myModel2);


                table2.setSelectionBackground(new Color(0, 0, 255));
                table2.setBackground(new Color(0, 102, 153));
                table2.setBorder(null);
                table2.setFont(new Font("Dialog", Font.BOLD, 15));

      //Creatamos un contenedor para la Tabla
      final JScrollPane scrollPane2 = new JScrollPane(table2);
      scrollPane2.setBounds(new Rectangle(2, 265, 600, 235));
      scrollPane2.getViewport().setBackground(new Color(51, 102, 153));
      scrollPane2.getViewport().setBorder(null);



      //Agregamos nuestra tabla al contenedor
      table2.addMouseListener(new MouseAdapter()
      {
        public void mouseClicked(MouseEvent e)
        {
           int modificar;
          int fila = table2.rowAtPoint(e.getPoint());
          int columna = table2.columnAtPoint(e.getPoint());
          if ((fila > -1) && (columna > -1))
          {
            modificar=JOptionPane.showConfirmDialog(null,"Desea añadir el útil:"+ob2[fila][0],"",
JOptionPane.YES_NO_OPTION);
            if(modificar==0)
                    try {
```

```
                                Class.forName("com.mysql.jdbc.Driver");
                                con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                Statement stat = con.createStatement();
                                String comprobar= "SELECT * FROM utiles WHERE
nombre="'+ob2[fila][0]+"'";
                                rs = stat.executeQuery(comprobar);
                                rs.next();
                                String idutil=rs.getString("Id_Util");


                                comprobar= "INSERT INTO utilesdeejercicios VALUES
("'+id+"','"+idutil+"')";
                                prueba = stat.executeUpdate(comprobar);
                                if(prueba==1)
                                        JOptionPane.showMessageDialog(null,"Asignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                else
                                        JOptionPane.showMessageDialog(null,"No asignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                                clickUtiles  b=new clickUtiles();
                                b.actualiza(panelrutinas,panelEncontrados, tabbedPane ,id);
                           con.close();

                        } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                        }
                }
              }
            });
                JButton btnAtras = new JButton("Atrás");
                        btnAtras.setBounds(415, 520, 130, 39);
                        panelEncontrados.add(btnAtras);

                JButton btnCrear = new JButton("Crear");
                btnCrear.setBounds(55, 520, 130, 39);
                        panelEncontrados.add(btnCrear);

                        JButton btnEliminar = new JButton("Eliminar");
                        btnEliminar.setBounds(235, 520, 130, 39);
                        panelEncontrados.add(btnEliminar);
                        btnCrear.addActionListener((ActionListener) new
CrearUtil(panelrutinas,panelEncontrados, tabbedPane ,id));
                        btnEliminar.addActionListener((ActionListener) new
EliminarUtil(panelrutinas,panelEncontrados, tabbedPane ,id));

                        btnAtras.addActionListener((ActionListener) new
clickAtrasaEjercicio(panelrutina,panelEncontrados,id,tabbedPane));

            panelEncontrados.add(scrollPane);
            panelEncontrados.add(scrollPane2);
            panelrutinas.add(panelEncontrados);
        }


}
```

B.1.39 CrearUtil.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase encargada de crear nuevos utiles
public class CrearUtil implements ActionListener{

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id2=null;

	public CrearUtil(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id2) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id2=id2;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Vamos mostrando mensajes solicitando cada uno de los campos de los utiles , si este esta vacio mostramos un mensaje de error
		String nombre=JOptionPane.showInputDialog(null,"Introduzca el nombre del útil","",JOptionPane.QUESTION_MESSAGE);
		String descripcion;
		String url;
		String insertar;
		int id;
		boolean sal=false;
		if(nombre.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir  el nombre del útil","",JOptionPane.WARNING_MESSAGE);
		else
		{
			descripcion=JOptionPane.showInputDialog(null,"Introduzca la descripción","",JOptionPane.QUESTION_MESSAGE);
			if(descripcion!=null)
				if(descripcion.equals(""))
					JOptionPane.showMessageDialog(null,"Debe introducir la descripción del útil","",JOptionPane.WARNING_MESSAGE);
				else
```

```
                                        {
                                                url=JOptionPane.showInputDialog(null,"Introduzca la
url","",JOptionPane.QUESTION_MESSAGE );
                                                if(url.equals(""))
                                                        JOptionPane.showMessageDialog(null,"Debe
introducir la url del útil","",JOptionPane.WARNING_MESSAGE);
                                                else
                                                {
                                                        try {
                                                                Class.forName("com.mysql.jdbc.Driver");
                                                                con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                                                Statement stat = con.createStatement();
                                                                String seleccionar = "SELECT * FROM
utiles";

                                                                rs = stat.executeQuery(seleccionar);
                                                                //Buscamos el id mas bajo libre

                                                                for( id=1;sal==false;id++)
                                                                {

                                                                        if(rs.next())
                                                                        {
                                        if(!rs.getString("Id_Util").equals(""+id))
                                                sal=true;
                                                                        }
                                                                        else
                                                                                sal=true;

                                                                }


                                                                id--;
                                                                //Insertamos el nuevo util en la base de
datos

                                                                insertar= "INSERT INTO utiles VALUES
("'+id+"','"+nombre+"','"  + descripcion + "','" + url  + "')";
                                                                stat.executeUpdate(insertar);
                                                                clickUtiles  b=new clickUtiles();
                                                                b.actualiza(panelEjercicios,Ejercicio,
panelPadre ,id2);
                                                        con.close();

                                                        } catch (ClassNotFoundException | SQLException
i) {
                                                                System.out.println("Error: " +
i.getMessage());
                                                        }

                                                }


                                        }
                                }
                        }
}
```

B.1.40 EliminarUtil.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase que elimina utiles de la base de datos

public class EliminarUtil implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id=null;

	public EliminarUtil(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id=id;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Solicita la peticion del nombre del util
		String nombre=JOptionPane.showInputDialog(null,"Introduzca el nombre del
útil","",JOptionPane.QUESTION_MESSAGE);
		String eliminar;
		int res;
		String comprobar;
		String idutil;

		//Comprueba si es cadena vacia
		if(nombre.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir el nombre del
útil","",JOptionPane.WARNING_MESSAGE);
		else
		{
			//En caso de no estar vacia solicita confirmacion
			int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar el útil
"+nombre+"?","", JOptionPane.YES_NO_OPTION);
			if(modificar==0)
			{
				try {
					Class.forName("com.mysql.jdbc.Driver");
					con =
```

```
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                        Statement stat = con.createStatement();
                                        comprobar= "SELECT * FROM utiles WHERE
nombre='"+nombre+"'";
                                        rs = stat.executeQuery(comprobar);
                                        //Comprueba que exista
                                        if (rs.next())
                                        {
                                                idutil=rs.getString("Id_Util");
                                                eliminar="DELETE FROM utiles WHERE
nombre='"+nombre+"'";
                                                res = stat.executeUpdate(eliminar);
                                                if(res==1)

        JOptionPane.showMessageDialog(null,"Eliminado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                                else
                                                        JOptionPane.showMessageDialog(null,"No
eliminado correctamente","",JOptionPane.ERROR_MESSAGE);
                                //Elimina el util y desasigna de todos los ejercicios
                                                comprobar= "SELECT * FROM utilesdeejercicios
WHERE Id_Util='"+idutil+"'";
                                                rs = stat.executeQuery(comprobar);
                                                if (rs.next())
                                                {
                                                        eliminar="DELETE FROM
utilesdeejercicios WHERE Id_Util='"+idutil+"'";
                                                        res = stat.executeUpdate(eliminar);
                                                }

                                        }
                                        else
                                                JOptionPane.showMessageDialog(null,"No existe
un util con ese nombre","",JOptionPane.WARNING_MESSAGE);
                                        //Sino existe muestra el error
                                    con.close();

                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }

                        //Actualizas las tablas
                                clickUtiles  b=new clickUtiles();
                                b.actualiza(panelEjercicios,Ejercicio, panelPadre ,id);
                            }
                    }
            }
}
```

B.1.41 clickAtrasaEjercicio.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;
//Clase encargada de volver a la pantalla de el ejercicio que se esta modificando
public class clickAtrasaEjercicio implements ActionListener {

	private JPanel panelPadre;
	private JPanel panelEjercicios;
	private JPanel panelEncontrados;
	private String id;

	public clickAtrasaEjercicio(JPanel panelEjercicios, JPanel panelEncontrados,String id, JPanel panelPadre) {

		this.id=id;
		this.panelPadre=panelPadre;
		this.panelEjercicios=panelEjercicios;
		this.panelEncontrados=panelEncontrados;

		}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Conectamos a la base de datos para acceder al nombre del ejercicio
		try {
			Class.forName("com.mysql.jdbc.Driver");
			Connection con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			// Creamos un Statement para poder hacer peticiones a la bd
			Statement stat = con.createStatement();

			String comprobar= "SELECT * FROM ejercicios WHERE
Id_Ejercicio='"+id+"'";
			ResultSet rs = stat.executeQuery(comprobar);
			rs.next();
			String id=rs.getString("Descripcion");

			new EjercicioConcreto(panelEjercicios,panelEncontrados,id,panelPadre);
		    con.close();


		} catch (ClassNotFoundException | SQLException i) {
			System.out.println("Error: " + i.getMessage());
		}

	}

}
```

B.1.42 clickMusculos.java

```
package Presentacion;


import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JPanel;

//Clase cuya funcion principal es cargar los utiles en dos grupos
//Estos grupos son los musculos que estan asignados al ejercicio y los que no
public class clickMusculos implements ActionListener {

	private JPanel panelEjercicios=null;
	private JPanel Ejercicios=null;
	private JPanel panelPadre=null;
	private String id=null;
	private Connection con;
	private ResultSet rs;
	private int i;

	public clickMusculos(JPanel panelEjercicios, JPanel Ejercicios, String id, JPanel panelPadre) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicios=Ejercicios;
		this.id=id;
		this.panelPadre=panelPadre;


	}

	public clickMusculos() {
	}


	@Override
	public void actionPerformed(ActionEvent e) {
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			//Consulta a la base de datos que identifica los musculos que estan asignados al
ejercicio

			String seleccionar = "SELECT m.Id_Musculo Id_Musculo, m.Nombre Nombre
,m.Descripcion Descripcion , m.urlfoto urlfoto "
					+ " FROM musculosdeejercicios me  INNER JOIN musculos
m ON me.Id_Musculo=m.Id_Musculo WHERE me.Id_Ejercicio="'+id+"'";

			rs = stat.executeQuery(seleccionar);
			rs.last();
			i = rs.getRow();
```

```
                        rs.beforeFirst();
                        //Object donde almacenamos los musculos que si estan asignados
                        Object [][]si=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                si[i][0]=rs.getString("Nombre");
                                si[i][1]=rs.getString("Descripcion");
                                si[i][2]=rs.getString("urlfoto");
                                i++;
                        }
                        //Consulta a la base de datos que identifica los musculos que no estan
asignados al ejercicio

                        seleccionar = "SELECT m.Id_Musculo Id_Musculo, m.Nombre Nombre
,m.Descripcion Descripcion , m.urlfoto urlfoto "
                                        + " FROM musculos m WHERE m.Id_Musculo NOT IN "
                                        + "(SELECT me.Id_Musculo FROM musculosdeejercicios
me WHERE me.Id_Ejercicio='"+id+"')";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        //Object donde almacenamos los musculos que no estan asignados

                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("Nombre");
                                no[i][1]=rs.getString("Descripcion");
                                no[i][2]=rs.getString("urlfoto");
                                i++;
                        }
                        //Nombres de las columnas que queremos que se muestren
                        final String[] columnNames = {"Nombre",
                                        "Descripcion",
                        "Ruta"};
                        new MusculosEjercicioEncontrados( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id);


                } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

        }

        //Metodo que se encarga de realizar una actualizacion de estos grupos
        public  void actualiza(JPanel panelEjercicios, JPanel Ejercicios, JPanel panelPadre, String id2) {


                try {
                        Class.forName("com.mysql.jdbc.Driver");
```

```
                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                        Statement stat = con.createStatement();


                        String seleccionar = "SELECT m.Id_Musculo Id_Musculo, m.Nombre Nombre
,m.Descripcion Descripcion , m.urlfoto urlfoto "
                                        + " FROM musculosdeejercicios me  INNER JOIN musculos
m ON me.Id_Musculo=m.Id_Musculo WHERE me.Id_Ejercicio='"+id2+"'";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        Object [][]si=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                si[i][0]=rs.getString("Nombre");
                                si[i][1]=rs.getString("Descripcion");
                                si[i][2]=rs.getString("urlfoto");
                                i++;
                        }

                        seleccionar = "SELECT m.Id_Musculo Id_Musculo, m.Nombre Nombre
,m.Descripcion Descripcion , m.urlfoto urlfoto "
                                        + " FROM musculos m WHERE m.Id_Musculo NOT IN "
                                        + "(SELECT me.Id_Musculo FROM musculosdeejercicios
me WHERE me.Id_Ejercicio='"+id2+"')";
                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        Object [][]no=new Object[i][3];
                        i=0;
                        while (rs.next())
                        {

                                no[i][0]=rs.getString("Nombre");
                                no[i][1]=rs.getString("Descripcion");
                                no[i][2]=rs.getString("urlfoto");
                                i++;
                        }
                        final String[] columnNames = {"Nombre",
                                        "Descripcion",
                        "Ruta"};
                        new MusculosEjercicioEncontrados( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id2);


                } catch (ClassNotFoundException | SQLException i2) {
                        System.out.println("Error: " + i2.getMessage());
                }
        }
}
```

B.1.43 MusculosEjercicioEncontrados.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
//Clase que muestra los dos grupos de musculos segun esten asignados o no y permite cambiarlo
//Ademas de crear y eliminarlos
public class MusculosEjercicioEncontrados extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel Rutina;
	public JPanel panelRutinas;
	public JPanel panelEncontrados;
	public JPanel panelPadre;
	public String id;
	private Connection con;
	private ResultSet rs;
	private int prueba;


	public MusculosEjercicioEncontrados(JPanel panelRutina, JPanel Rutina, final Object[][] ob,
final String[] columnNames,final Object[][] ob2,final JPanel panelPadre, final String id) {

		this.panelRutinas=panelRutina;
		this.Rutina=Rutina;
		this.panelPadre=panelPadre;
		this.id=id;
		Rutina.setVisible(false);
		panelEncontrados = new JPanel();
		panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
		panelEncontrados.setBackground(new Color(51, 102, 153));
		panelEncontrados.setBorder(null);
		panelEncontrados.setLayout(null);
	String ejerciciostring=null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
```

```
                        String comprobar= "SELECT * FROM ejercicios WHERE
Id_Ejercicio="'+id+"'";
                        rs = stat.executeQuery(comprobar);
                        rs.next();
                        ejerciciostring = "Ejercicio: "+rs.getString("Descripcion");
                    con.close();

            } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                    }

                    JLabel ejerciciotxt = new JLabel(ejerciciostring);
                    ejerciciotxt.setFont(new Font("Dialog", Font.BOLD, 20));
                    ejerciciotxt.setBounds(2, 15, 500, 28);
                    panelEncontrados.add(ejerciciotxt);

                    JLabel asignadas = new JLabel("Musculos asignados:");
                    asignadas.setFont(new Font("Dialog", Font.BOLD, 20));
                    asignadas.setBounds(2, 50, 500, 28);
                    panelEncontrados.add(asignadas);

                    final MyTableModel myModel = new MyTableModel(ob,columnNames);
                    final JTable table = new JTable(myModel);



                    table.setSelectionBackground(new Color(0, 0, 255));
                    table.setBackground(new Color(0, 102, 153));
                    table.setBorder(null);
                    table.setFont(new Font("Dialog", Font.BOLD, 15));
        //Creatamos un contenedor para la Tabla
        final JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setBounds(new Rectangle(2, 80, 600, 150));
        scrollPane.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane.getViewport().setBorder(null);


        table.addMouseListener(new MouseAdapter()
        {
            public void mouseClicked(MouseEvent e)
            {
                    int modificar;
                    int fila = table.rowAtPoint(e.getPoint());
                    int columna = table.columnAtPoint(e.getPoint());
                    if ((fila > -1) && (columna > -1))
                    {
                        modificar=JOptionPane.showConfirmDialog(null,"Desea desasignar el
músculo:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                        if(modificar==0)
                            try {
                                Class.forName("com.mysql.jdbc.Driver");
                                con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                Statement stat = con.createStatement();
                String comprobar= "SELECT * FROM musculos WHERE Nombre="'+ob[fila][0]+"';
```

```
                System.out.println(comprobar);
                rs = stat.executeQuery(comprobar);
                rs.next();
                String idmus=rs.getString("Id_Musculo");

                comprobar= "DELETE FROM musculosdeejercicios WHERE Id_Ejercicio='"+id+"' and
Id_Musculo='"+idmus+"'";
                prueba = stat.executeUpdate(comprobar);
                if(prueba==1)
                        JOptionPane.showMessageDialog(null,"Desasignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                else
                        JOptionPane.showMessageDialog(null,"No desasignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                clickMusculos  b=new clickMusculos();
                b.actualiza(panelRutinas,panelEncontrados, panelPadre ,id);
            } catch (ClassNotFoundException | SQLException i) {
                System.out.println("Error: " + i.getMessage());
             }
                    }
            }
        });

        JLabel noasignadas = new JLabel("Musculos no asignados:");
        noasignadas.setFont(new Font("Dialog", Font.BOLD, 20));
        noasignadas.setBounds(2, 235, 500, 28);
        panelEncontrados.add(noasignadas);

        final MyTableModel myModel2 = new MyTableModel(ob2,columnNames);
        final JTable table2 = new JTable(myModel2);


                    table2.setSelectionBackground(new Color(0, 0, 255));
                    table2.setBackground(new Color(0, 102, 153));
                    table2.setBorder(null);
                    table2.setFont(new Font("Dialog", Font.BOLD, 15));

        //Creatamos un contenedor para la Tabla
        final JScrollPane scrollPane2 = new JScrollPane(table2);
        scrollPane2.setBounds(new Rectangle(2, 265, 600, 235));
        scrollPane2.getViewport().setBackground(new Color(51, 102, 153));
        scrollPane2.getViewport().setBorder(null);
         table2.addMouseListener(new MouseAdapter()

        {
            public void mouseClicked(MouseEvent e)
            {
                    int modificar;
                    int fila = table2.rowAtPoint(e.getPoint());
                    int columna = table2.columnAtPoint(e.getPoint());
                    if ((fila > 0) && (columna > -1))
                    {
                            modificar=JOptionPane.showConfirmDialog(null,"Desea añadir el
músculo:"+ob2[fila][0],"", JOptionPane.YES_NO_OPTION);
                            if(modificar==0)
                                    try {
```

```
                                        Class.forName("com.mysql.jdbc.Driver");
                                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                        Statement stat = con.createStatement();
                                        String comprobar= "SELECT * FROM musculos WHERE
Nombre='"+ob2[fila][0]+"'";
                                        rs = stat.executeQuery(comprobar);
                                        rs.next();
                                        String idmus=rs.getString("Id_Musculo");

                                        comprobar= "INSERT INTO musculosdeejercicios VALUES
('"+id+"','"+idmus+"')";
                                        prueba = stat.executeUpdate(comprobar);
                                        if(prueba==1)
                                                JOptionPane.showMessageDialog(null,"Asignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                        else
                                                JOptionPane.showMessageDialog(null,"No
asignado correctamente","",JOptionPane.ERROR_MESSAGE);

                                        clickMusculos  b=new clickMusculos();
                                        b.actualiza(panelRutinas,panelEncontrados, panelPadre ,id);
                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }
                        }
                }
        });

        JButton btnAtras = new JButton("Atrás");
                        btnAtras.setBounds(415, 520, 130, 39);
                        panelEncontrados.add(btnAtras);

                JButton btnCrear = new JButton("Crear");
                btnCrear.setBounds(55, 520, 130, 39);
                        panelEncontrados.add(btnCrear);

                        JButton btnEliminar = new JButton("Eliminar");
                        btnEliminar.setBounds(235, 520, 130, 39);
                        panelEncontrados.add(btnEliminar);


                        btnCrear.addActionListener((ActionListener) new
CrearMusculo(panelRutinas,panelEncontrados, panelPadre ,id));
                        btnEliminar.addActionListener((ActionListener) new
EliminarMusculo(panelRutinas,panelEncontrados, panelPadre ,id));
                        btnAtras.addActionListener((ActionListener) new
clickAtrasaEjercicio(panelRutina,panelEncontrados,id,panelPadre));

        panelEncontrados.add(scrollPane);
        panelEncontrados.add(scrollPane2);

        panelRutinas.add(panelEncontrados);
    }

}
```

B.1.44 CrearMusculo.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase encargada de crear nuevos musculos
public class CrearMusculo implements ActionListener{

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id2=null;

	public CrearMusculo(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id2) {
		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id2=id2;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Vamos mostrando mensajes solicitando cada uno de los campos de los musculos , si
este esta vacio mostramos un mensaje de error
		String nombre=JOptionPane.showInputDialog(null, "Introduzca el nombre del
músculo", "",JOptionPane.QUESTION_MESSAGE);
		String descripcion;
		String url;
		String insertar;
		int id;
		boolean sal=false;
		if(nombre.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir  el nombre del
músculo","",JOptionPane.WARNING_MESSAGE);
		else
		{
			descripcion=JOptionPane.showInputDialog(null,"Introduzca la descripción",
"",JOptionPane.QUESTION_MESSAGE);
			if(descripcion!=null)
				if(descripcion.equals(""))
					JOptionPane.showMessageDialog(null,"Debe introducir la
descripción del músculo","",JOptionPane.WARNING_MESSAGE);
				else
				{
					url=JOptionPane.showInputDialog(null, "Introduzca la url",
"",JOptionPane.QUESTION_MESSAGE);
```

```
                                        if(url.equals(""))
                                                JOptionPane.showMessageDialog(null,"Debe
introducir la url del músculo","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {
                                                try {
                                                        Class.forName("com.mysql.jdbc.Driver");
                                                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                                        Statement stat = con.createStatement();
                                                        String seleccionar = "SELECT * FROM
musculos";

                                                        rs = stat.executeQuery(seleccionar);
                                                        //Buscamos el id mas bajo libre
                                                        for( id=1;sal==false;id++)
                                                        {

                                                                if(rs.next())
                                                                {

        if(!rs.getString("Id_musculo").equals(""+id))
                                                                                sal=true;
                                                                }
                                                                else
                                                                        sal=true;

                                                        }

                                                        id--;
                                                        //Insertamos el nuevo musculo en la base de
datos
                                                        insertar= "INSERT INTO musculos
VALUES ("'+id+"','"+nombre+"','"  + descripcion + "','" + url  + "')";
                                                        stat.executeUpdate(insertar);
                                                        clickMusculos  b=new clickMusculos();
                                                        b.actualiza(panelEjercicios,Ejercicio,
panelPadre ,id2);
                                                    con.close();

                                                } catch (ClassNotFoundException | SQLException
i) {
                                                        System.out.println("Error: " +
i.getMessage());
                                                }

                                        }
                                }
                        }
                }
}
```

B.1.45 EliminarMusculo.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase que elimina musculos de la base de datos
public class EliminarMusculo implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id=null;

	public EliminarMusculo(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id=id;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Solicita la peticion del nombre del musculo
		String nombre=JOptionPane.showInputDialog(null,"Introduzca el nombre del
músculo","",JOptionPane.QUESTION_MESSAGE );
		String eliminar;
		int res;
		String comprobar;
		String idmusc;

		//Comprueba si es cadena vacia
		if(nombre.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir  el nombre del
músculo","",JOptionPane.WARNING_MESSAGE);
		else
		{
			//En caso de no estar vacia solicita confirmacion
			int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar el
músculo "+nombre+"?","", JOptionPane.YES_NO_OPTION);
			if(modificar==0)
			{
				try {
					Class.forName("com.mysql.jdbc.Driver");
					con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
```

```
                                        Statement stat = con.createStatement();
                                        comprobar= "SELECT * FROM musculos WHERE
Nombre='"+nombre+"'";
                                        rs = stat.executeQuery(comprobar);
                                        //Comprueba que exista
                                        if (rs.next())
                                        {
                                                idmusc=rs.getString("Id_Musculo");
                                                eliminar="DELETE FROM musculos WHERE
Nombre='"+nombre+"'";
                                                res = stat.executeUpdate(eliminar);
                                                if(res==1)

        JOptionPane.showMessageDialog(null,"Eliminado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                                else
                                                        JOptionPane.showMessageDialog(null,"No
eliminado correctamente","",JOptionPane.ERROR_MESSAGE);
                                        //Elimina el musculo y desasigna de todos los ejercicios
                                                comprobar= "SELECT * FROM
musculosdeejercicios WHERE Id_Musculo='"+idmusc+"'";
                                                rs = stat.executeQuery(comprobar);
                                                if (rs.next())
                                                {
                                                        eliminar="DELETE FROM
musculosdeejercicios WHERE Id_Musculo='"+idmusc+"'";
                                                        res = stat.executeUpdate(eliminar);
                                                }

                                        }
                                        else
                                                JOptionPane.showMessageDialog(null,"No existe
un músculo con ese nombre","",JOptionPane.WARNING_MESSAGE);
                                        //Sino existe muestra el error
                                    con.close();

                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }

                        //Actualizas las tablas
                                clickMusculos  b=new clickMusculos();
                                b.actualiza(panelEjercicios,Ejercicio, panelPadre ,id);
                            }
                        }
                }
        }
}
```

B.1.46 clickRecursos.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JPanel;

//Clase cuya funcion principal es cargar los utiles en dos grupos
//Estos grupos son los recursos que estan asignados al ejercicio y los que no
public class clickRecursos implements ActionListener {

	private JPanel panelEjercicios=null;
	private JPanel Ejercicios=null;
	private JPanel panelPadre=null;
	private String id=null;
	private Connection con;
	private ResultSet rs;
	private int i;

   public clickRecursos(JPanel panelEjercicios, JPanel Ejercicios, String id, JPanel panelPadre) {

	this.panelEjercicios=panelEjercicios;
	this.Ejercicios=Ejercicios;
		this.id=id;
		this.panelPadre=panelPadre;
  }

	public clickRecursos() {
	}
	@Override
	public void actionPerformed(ActionEvent e) {
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
			Statement stat = con.createStatement();
			//Consulta a la base de datos que identifica los recursos que estan asignados al
ejercicio

			String seleccionar = "SELECT r.Id_Recurso Id_Recurso, r.titulo titulo
,r.descripcion descripcion , r.posicion_foto posicion_foto, r.url url, r.tipo_recurso tipo_recurso"
					+ " FROM recursosdeejercicios re  INNER JOIN recursos r
ON re.Id_Recurso=r.Id_Recurso WHERE re.Id_Ejercicio='"+id+"'";

			rs = stat.executeQuery(seleccionar);
			rs.last();
			i = rs.getRow();
			rs.beforeFirst();
			//Object donde almacenamos los recursos que si estan asignados

			Object [][]si=new Object[i][5];
			i=0;
```

```
                        while (rs.next())
                        {
                                si[i][0]=rs.getString("titulo");
                                si[i][1]=rs.getString("descripcion");
                                si[i][2]=rs.getString("posicion_foto");
                                si[i][3]=rs.getString("url");
                                si[i][4]=rs.getString("tipo_recurso");
                                i++;
                        }
        //Consulta a la base de datos que identifica los recursos que no estan asignados al ejercicio
                        seleccionar = "SELECT r.Id_Recurso Id_Recurso, r.titulo titulo ,r.descripcion
descripcion , r.posicion_foto posicion_foto, r.url url, r.tipo_recurso tipo_recurso"
                                        + " FROM recursos r WHERE r.Id_Recurso NOT IN "
                                        + "(SELECT re.Id_Recurso FROM recursosdeejercicios re
WHERE re.Id_Ejercicio="'+id+'")";

                        rs = stat.executeQuery(seleccionar);
                        rs.last();
                        i = rs.getRow();
                        rs.beforeFirst();
                        //Object donde almacenamos los recursos que no estan asignados
                        Object [][]no=new Object[i][5];
                        i=0;
                        while (rs.next())
                        {
                                no[i][0]=rs.getString("titulo");
                                no[i][1]=rs.getString("descripcion");
                                no[i][2]=rs.getString("posicion_foto");
                                no[i][3]=rs.getString("url");
                                no[i][4]=rs.getString("tipo_recurso");
                                i++;
                        }
                        //Nombres de las columnas que queremos que se muestren
                        final String[] columnNames = {"Titulo",
                                        "Descripcion",
                                        "Posicion_foto",
                                        "Ruta",
                        "Tipo de recurso"};

                        new RecursosEjercicioEncontrados( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id);

                            con.close();
                        } catch (ClassNotFoundException | SQLException i) {
                                System.out.println("Error: " + i.getMessage());
                        }
                }
                //Metodo que se encarga de realizar una actualizacion de estos grupos
                public  void actualiza(JPanel panelEjercicios, JPanel Ejercicios, JPanel panelPadre, String id2) {


                        try {
                Class.forName("com.mysql.jdbc.Driver");
                con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                Statement stat = con.createStatement();

                String seleccionar = "SELECT r.Id_Recurso Id_Recurso, r.titulo titulo ,r.descripcion descripcion ,
```

```
r.posicion_foto posicion_foto, r.url url, r.tipo_recurso tipo_recurso"
                    + " FROM recursosdeejercicios re  INNER JOIN recursos r ON
re.Id_Recurso=r.Id_Recurso WHERE re.Id_Ejercicio='"+id2+"'";

          rs = stat.executeQuery(seleccionar);
          rs.last();
          i = rs.getRow();
          rs.beforeFirst();
          Object [][]si=new Object[i][5];
          i=0;
          while (rs.next())
          {
            si[i][0]=rs.getString("titulo");
            si[i][1]=rs.getString("descripcion");
            si[i][2]=rs.getString("posicion_foto");
            si[i][3]=rs.getString("url");
            si[i][4]=rs.getString("tipo_recurso");
            i++;
          }

          seleccionar = "SELECT r.Id_Recurso Id_Recurso, r.titulo titulo ,r.descripcion descripcion ,
r.posicion_foto posicion_foto, r.url url, r.tipo_recurso tipo_recurso"
                    + " FROM recursos r WHERE r.Id_Recurso NOT IN "
                    + "(SELECT re.Id_Recurso FROM recursosdeejercicios re WHERE
re.Id_Ejercicio='"+id2+"')";
          rs = stat.executeQuery(seleccionar);
          rs.last();
          i = rs.getRow();
          rs.beforeFirst();
          Object [][]no=new Object[i][5];
          i=0;
          while (rs.next())
          {
            no[i][0]=rs.getString("titulo");
            no[i][1]=rs.getString("descripcion");
            no[i][2]=rs.getString("posicion_foto");
            no[i][3]=rs.getString("url");
            no[i][4]=rs.getString("tipo_recurso");
            i++;
          }

          final String[] columnNames = {"Titulo",
                    "Descripcion",
                    "Posicion_foto",
                    "Ruta",
          "Tipo de recurso"};
                         con.close();

          new RecursosEjercicioEncontrados( panelEjercicios,  Ejercicios,  si,
columnNames,no,panelPadre,id2);

            } catch (ClassNotFoundException | SQLException i2) {
                    System.out.println("Error: " + i2.getMessage());
            }
            }
}
```

B.1.47 RecursosEjercicioEncontrados.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
//Clase que muestra los dos grupos de recursos segun esten asignados o no y permite cambiarlo
//Ademas de crear y eliminarlos

public class RecursosEjercicioEncontrados extends JFrame {


	private static final long serialVersionUID = 1L;
	public JPanel Ejercicios;
   public JPanel panelEjercicioss;
   public JPanel panelEncontrados;
   public JPanel panelPadre;
   public String id;
   private Connection con;
   private ResultSet rs;
   private int prueba;


	public RecursosEjercicioEncontrados(JPanel panelEjercicios, JPanel Ejercicios, final Object[][]
ob, final String[] columnNames,final Object[][] ob2,final JPanel panelPadre, final String id) {

		this.panelEjercicioss=panelEjercicios;
		this.Ejercicios=Ejercicios;
		this.panelPadre=panelPadre;
		this.id=id;
		Ejercicios.setVisible(false);
		panelEncontrados = new JPanel();
		panelEncontrados.setBounds(new Rectangle(0, 0, 650, 582));
      panelEncontrados.setBackground(new Color(51, 102, 153));
      panelEncontrados.setBorder(null);
      panelEncontrados.setLayout(null);

      String ejerciciostring=null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			con =
```

```
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                        Statement stat = con.createStatement();


                        String comprobar= "SELECT * FROM ejercicios WHERE
Id_Ejercicio="'+id+"'";
                        rs = stat.executeQuery(comprobar);
                        rs.next();
                        ejerciciostring = "Ejercicio: "+rs.getString("Descripcion");
        } catch (ClassNotFoundException | SQLException i) {
                        System.out.println("Error: " + i.getMessage());
                }

                JLabel ejerciciotxt = new JLabel(ejerciciostring);
                ejerciciotxt.setFont(new Font("Dialog", Font.BOLD, 20));
                ejerciciotxt.setBounds(2, 15, 500, 28);
                panelEncontrados.add(ejerciciotxt);

    JLabel asignadas = new JLabel("Recursos asignados:");
    asignadas.setFont(new Font("Dialog", Font.BOLD, 20));
    asignadas.setBounds(2, 50, 500, 28);
                panelEncontrados.add(asignadas);

    final MyTableModel myModel = new MyTableModel(ob,columnNames);
                final JTable table = new JTable(myModel);



                table.setSelectionBackground(new Color(0, 0, 255));
                table.setBackground(new Color(0, 102, 153));
                table.setBorder(null);
                table.setFont(new Font("Dialog", Font.BOLD, 15));

                //Creatamos un contenedor para la Tabla
    final JScrollPane scrollPane = new JScrollPane(table);
    scrollPane.setBounds(new Rectangle(2, 80, 600, 150));
    scrollPane.getViewport().setBackground(new Color(51, 102, 153));
    scrollPane.getViewport().setBorder(null);


    table.addMouseListener(new MouseAdapter()
    {
      public void mouseClicked(MouseEvent e)
      {
          int modificar;
          int fila = table.rowAtPoint(e.getPoint());
          int columna = table.columnAtPoint(e.getPoint());
          if ((fila > -1) && (columna > -1))
        {

                    modificar=JOptionPane.showConfirmDialog(null,"Desea desasignar el
recurso:"+ob[fila][0],"", JOptionPane.YES_NO_OPTION);
                    if(modificar==0)
                            try {
                                    Class.forName("com.mysql.jdbc.Driver");
                                    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
```

```
                                        Statement stat = con.createStatement();


                                        String comprobar= "SELECT * FROM recursos WHERE
titulo='"+ob[fila][0]+"'";
                                        rs = stat.executeQuery(comprobar);
                                        rs.next();
                                        String idrec=rs.getString("Id_Recurso");

                                        comprobar= "DELETE FROM recursosdeejercicios WHERE
Id_Ejercicio='"+id+"' and Id_Recurso='"+idrec+"'";
                                        prueba = stat.executeUpdate(comprobar);
                                        if(prueba==1)
                                                JOptionPane.showMessageDialog(null,"Desasignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                        else
                                                JOptionPane.showMessageDialog(null,"No desasignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                                        clickRecursos  b=new clickRecursos();
                                        b.actualiza(panelEjercicioss,panelEncontrados, panelPadre ,id);
                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }
                        }
                    }
                });

                JLabel noasignadas = new JLabel("Recursos no asignados:");
                noasignadas.setFont(new Font("Dialog", Font.BOLD, 20));
                noasignadas.setBounds(2, 235, 500, 28);
                        panelEncontrados.add(noasignadas);


                final MyTableModel myModel2 = new MyTableModel(ob2,columnNames);
                        final JTable table2 = new JTable(myModel2);

                        table2.setSelectionBackground(new Color(0, 0, 255));
                        table2.setBackground(new Color(0, 102, 153));
                        table2.setBorder(null);
                        table2.setFont(new Font("Dialog", Font.BOLD, 15));

                //Creatamos un contenedor para la Tabla
                final JScrollPane scrollPane2 = new JScrollPane(table2);
                scrollPane2.setBounds(new Rectangle(2, 265, 600, 235));
                scrollPane2.getViewport().setBackground(new Color(51, 102, 153));
                scrollPane2.getViewport().setBorder(null);

                table2.addMouseListener(new MouseAdapter()
                {
                  public void mouseClicked(MouseEvent e)
                  {
                        int modificar;
                        int fila = table2.rowAtPoint(e.getPoint());
                        int columna = table2.columnAtPoint(e.getPoint());
                        if ((fila > -1) && (columna > -1))
                        {
```

```
                    modificar=JOptionPane.showConfirmDialog(null,"Desea añadir el
recurso:"+ob2[fila][0],"", JOptionPane.YES_NO_OPTION);
                    if(modificar==0)
                            try {
                                    Class.forName("com.mysql.jdbc.Driver");
                                    con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                    Statement stat = con.createStatement();
                                    String comprobar= "SELECT * FROM recursos WHERE
titulo='"+ob2[fila][0]+"'";
                                    rs = stat.executeQuery(comprobar);
                                    rs.next();
                                    String idrec=rs.getString("Id_Recurso");
                                    comprobar= "INSERT INTO recursosdeejercicios VALUES
('"+id+"','"+idrec+"')";
                                    prueba = stat.executeUpdate(comprobar);
                                    if(prueba==1)
                                            JOptionPane.showMessageDialog(null,"Asignado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                    else
                                            JOptionPane.showMessageDialog(null,"No asignado
correctamente","",JOptionPane.ERROR_MESSAGE);

                                    clickRecursos  b=new clickRecursos();
                                    b.actualiza(panelEjercicioss,panelEncontrados, panelPadre ,id);
                            } catch (ClassNotFoundException | SQLException i) {
                                    System.out.println("Error: " + i.getMessage());
                            }
                    }
            }
        });
            JButton btnAtras = new JButton("Atrás");
                    btnAtras.setBounds(415, 520, 130, 39);
                    panelEncontrados.add(btnAtras);

            JButton btnCrear = new JButton("Crear");
            btnCrear.setBounds(55, 520, 130, 39);
                    panelEncontrados.add(btnCrear);

                    JButton btnEliminar = new JButton("Eliminar");
                    btnEliminar.setBounds(235, 520, 130, 39);
                    panelEncontrados.add(btnEliminar);

                    btnAtras.addActionListener((ActionListener) new
clickAtrasaEjercicio(panelEjercicios,panelEncontrados,id,panelPadre));
                    btnCrear.addActionListener((ActionListener) new
CrearRecursos(panelEjercicioss,panelEncontrados, panelPadre ,id));
                    btnEliminar.addActionListener((ActionListener) new
EliminarRecurso(panelEjercicioss,panelEncontrados, panelPadre ,id));

        panelEncontrados.add(scrollPane);
        panelEncontrados.add(scrollPane2);

        panelEjercicioss.add(panelEncontrados);
    }

}
```

B.1.48 CrearRecursos.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase encargada de crear nuevos recursos
public class CrearRecursos implements ActionListener{

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id2=null;

	public CrearRecursos(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id2) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id2=id2;
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Vamos mostrando mensajes solicitando cada uno de los campos de los recursos , si este esta vacio mostramos un mensaje de error
		String titulo=JOptionPane.showInputDialog(null,"Introduzca el nombre del recurso","",JOptionPane.QUESTION_MESSAGE);
		String descripcion;
		String posicion_foto;
		String url;
		String tipo_recurso;
		String insertar;
		int id;
		boolean sal=false;
		if(titulo.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir el nombre del recurso","",JOptionPane.WARNING_MESSAGE);
		else
		{
			descripcion=JOptionPane.showInputDialog(null,"Introduzca la descripción","",JOptionPane.QUESTION_MESSAGE );
			if(descripcion.equals(""))
				JOptionPane.showMessageDialog(null,"Debe introducir la descripción del recurso","",JOptionPane.WARNING_MESSAGE);
			else
			{
				url=JOptionPane.showInputDialog(null,"Introduzca la url","",JOptionPane.QUESTION_MESSAGE);
```

```
                                        if(url.equals(""))
                                                JOptionPane.showMessageDialog(null,"Debe introducir la
url del recurso","",JOptionPane.WARNING_MESSAGE);
                                        else
                                        {

        posicion_foto=JOptionPane.showInputDialog(null,"Introduzca la posición de la
foto","",JOptionPane.QUESTION_MESSAGE );
                                                if(posicion_foto.equals(""))
                                                        JOptionPane.showMessageDialog(null,"Debe
introducir la posición de la foto del recurso","",JOptionPane.WARNING_MESSAGE);
                                                else
                                                {

        tipo_recurso=JOptionPane.showInputDialog(null,"Introduzca el tipo de
recurso","",JOptionPane.QUESTION_MESSAGE );
                                                        if(tipo_recurso.equals(""))

        JOptionPane.showMessageDialog(null,"Debe introducir el tipo de
recurso","",JOptionPane.WARNING_MESSAGE);
                                                        else
                                                        {
                                                                try {

        Class.forName("com.mysql.jdbc.Driver");
                                                                        con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
                                                                        Statement stat =
con.createStatement();
                                                                        String seleccionar = "SELECT *
FROM recursos";
                                                                        rs =
stat.executeQuery(seleccionar);
                                                                        //Buscamos el id mas bajo libre

                                                                        for( id=1;sal==false;id++)
                                                                        {

                                                                                if(rs.next())
                                                                                {

                        if(!rs.getString("Id_Recurso").equals(""+id))

        sal=true;

                                                                                }
                                                                                else
                                                                                        sal=true;

                                                                        }

                                                                        id--;
                                                                        //Insertamos el nuevo recurso en la
base de datos


                                                                        insertar= "INSERT INTO recursos
VALUES ("'+id+"','"+titulo+"','"  + descripcion + "','"
                                                                                        + posicion_foto
+ "','"+url+"','" + tipo_recurso  +"')";
```

```
                                                    stat.executeUpdate(insertar);
                                                    clickRecursos  b=new
clickRecursos();

            b.actualiza(panelEjercicios,Ejercicio, panelPadre ,id2);
                                                        con.close();

                                                    } catch (ClassNotFoundException |
SQLException i) {
                                                            System.out.println("Error: " +
i.getMessage());
                                                    }

                                                }
                                            }
                                        }
                                    }
                                }
                            }
                        }
```

B.1.49 EliminarRecurso.java

```
package Presentacion;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;
import javax.swing.JPanel;
//Clase que elimina recursos de la base de datos

public class EliminarRecurso implements ActionListener {

	private Connection con;
	private ResultSet rs;
	private JPanel panelEjercicios=null;
	private JPanel Ejercicio=null;
	private JPanel panelPadre=null;
	private String id=null;

	public EliminarRecurso(JPanel panelEjercicios, JPanel Ejercicio,JPanel panelPadre, String id) {

		this.panelEjercicios=panelEjercicios;
		this.Ejercicio=Ejercicio;
		this.panelPadre=panelPadre;
		this.id=id;

	}

	@Override
	public void actionPerformed(ActionEvent e) {
		//Solicita la peticion del nombre del recurso
		String nombre=JOptionPane.showInputDialog(null,"Introduzca el nombre del
recurso","",JOptionPane.QUESTION_MESSAGE);
		String eliminar;
		int res;
		String comprobar;
		String idrec;
		//Comprueba si es cadena vacia
		if(nombre.equals(""))
			JOptionPane.showMessageDialog(null,"Debe introducir el nombre del
recurso","",JOptionPane.WARNING_MESSAGE);
		else
		{
			//En caso de no estar vacia solicita confirmacion
			int modificar=JOptionPane.showConfirmDialog(null,"¿Desea eliminar el
recurso "+nombre+"?","", JOptionPane.YES_NO_OPTION);
			if(modificar==0)
			{
				try {
					Class.forName("com.mysql.jdbc.Driver");
					con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1/androiddb","root", "");
```

```
                                        Statement stat = con.createStatement();
                                        comprobar= "SELECT * FROM recursos WHERE
titulo='"+nombre+"'";
                                        rs = stat.executeQuery(comprobar);
                                        //Comprueba que exista
                                        if (rs.next())
                                        {
                                                idrec=rs.getString("Id_Recurso");
                                                eliminar="DELETE FROM recursos WHERE
titulo='"+nombre+"'";
                                                res = stat.executeUpdate(eliminar);
                                                if(res==1)

        JOptionPane.showMessageDialog(null,"Eliminado
correctamente","",JOptionPane.INFORMATION_MESSAGE);
                                                else
                                                        JOptionPane.showMessageDialog(null,"No
eliminado correctamente","",JOptionPane.ERROR_MESSAGE);
                                        //Elimina el recurso y desasigna de todos los ejercicios
                                                comprobar= "SELECT * FROM
recursosdeejercicios WHERE Id_Recurso='"+idrec+"'";
                                                rs = stat.executeQuery(comprobar);
                                                if (rs.next())
                                                {
                                                        eliminar="DELETE FROM
recursosdeejercicios WHERE Id_Recurso='"+idrec+"'";
                                                        res = stat.executeUpdate(eliminar);
                                                }

                                        }
                                        else
                                                JOptionPane.showMessageDialog(null,"No existe
un util con ese nombre","",JOptionPane.WARNING_MESSAGE);
                                        //Sino existe muestra el error
                                    con.close();

                                } catch (ClassNotFoundException | SQLException i) {
                                        System.out.println("Error: " + i.getMessage());
                                }

                        //Actualizas las tablas
                                clickRecursos  b=new clickRecursos();
                                b.actualiza(panelEjercicios,Ejercicio, panelPadre ,id);
                            }
                    }
            }
}
```

B.1.50 MuestraKafka.java

```
package Presentacion;

import java.awt.Color;
import java.awt.Font;
import java.awt.Rectangle;
import java.util.Vector;

import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.TableColumn;

import kafka.api.FetchRequest;
import kafka.javaapi.consumer.SimpleConsumer;
import kafka.javaapi.message.ByteBufferMessageSet;
import kafka.message.MessageAndOffset;

public class MuestraKafka {

	private Vector<Mensaje> v;
	private SimpleConsumer simpleConsumer;
	private int i;
	private JPanel Kafka;
	private JScrollPane scrollPane;
	private boolean filtrado=false;
	private Vector<Mensaje> n;
	private boolean primvez=true;


	public MuestraKafka(Vector<Mensaje> v,SimpleConsumer simpleConsumer, int i, JPanel
Kafka) {
		this.v=v;
		this.simpleConsumer=simpleConsumer;
		this.i=i;
		this.Kafka=Kafka;
	n = new Vector<Mensaje>();

		}

	public long MuestraMensajes(long offset) {
		if(filtrado==true)
		{
			if(offset>Principal.nummensajesmax*Principal.tamañomensajes)
			offset=offset-Principal.nummensajesmax*Principal.tamañomensajes;
			else
				offset=0;

			n.removeAllElements();
		}
	FetchRequest req = new FetchRequest("test", 0, offset, 1000);
	ByteBufferMessageSet messageSet = simpleConsumer.fetch(req);
	for (MessageAndOffset messageAndOffset : messageSet)
	{
		String mensaje=ExampleUtils.getMessage(messageAndOffset.message());
	String [] campos = mensaje.split("-");
	if(campos.length>2)
```

```
		{
			Mensaje mes=new
Mensaje(campos[0].replaceAll("\\s*$",""),campos[1].replaceAll("\\s*$",""),campos[2].replaceAll("\\s*$",
""));
			if(n.size()>Principal.nummensajesmax)
				n.remove(0);
			n.add(mes);
		}
		offset=messageAndOffset.offset();
		System.out.println(offset/Principal.tamañomensajes);
		}
		if(primvez==true || (n.get(0).getFecha()!=v.get(0).getFecha() || filtrado==true||
n.size()<Principal.nummensajesmax))
		{
			primvez=false;
			v=n;
		i=v.size();
		if(i>0)
		{
		Object [][]ob=new Object[i][3];
		int j=0;
		while (j<i)
		{

			ob[j][0]=v.get(j).getUsuario();
			ob[j][1]=v.get(j).getEvento();
			ob[j][2]=v.get(j).getFecha();
			j++;

		}

		final String[] columnNames = {"Usuario",
				"Evento",
		"Fecha"};

		final MyTableModel myModel = new MyTableModel(ob,columnNames);
		final JTable table = new JTable(myModel);

		table.setSelectionBackground(new Color(0, 0, 255));
		table.setBackground(new Color(0, 102, 153));
		table.setBorder(null);

		table.setFont(new Font("Dialog", Font.BOLD, 15));
		TableColumn columna;
		columna=table.getColumnModel().getColumn(0);
		columna.setPreferredWidth(110);
		columna.setMaxWidth(110);
		columna=table.getColumnModel().getColumn(1);
		columna.setPreferredWidth(95);
		columna.setMaxWidth(95);

		if(scrollPane!=null)
			Kafka.remove(scrollPane);
	//Creatamos un contenedor para la Tabla
		scrollPane = new JScrollPane(table);
	scrollPane.setBounds(new Rectangle(0, 2, 600, 500));
	scrollPane.getViewport().setBackground(new Color(51, 102, 153));
```

```
    scrollPane.getViewport().setBorder(null);
            Kafka.add(scrollPane);
            filtrado=false;
            }
            }
            return offset;
            }

            public void filtraMensaje(String usuario, String evento,long offset) {
                    //Funcion para filtrar
                    filtrado=true;
                                    if(!(usuario.isEmpty()&&evento.isEmpty()))
                                    {
                                            FetchRequest req = new FetchRequest("test", 0, offset,
1000);
                                            ByteBufferMessageSet messageSet =
simpleConsumer.fetch(req);
                                            for (MessageAndOffset messageAndOffset : messageSet)
                                            {
                                                    String
mensaje=ExampleUtils.getMessage(messageAndOffset.message());
                                            String [] campos = mensaje.split("-");
                                            if(campos.length>2)
                                            {
                                                    Mensaje mes=new
Mensaje(campos[0].replaceAll("\\s*$",""),campos[1].replaceAll("\\s*$",""),campos[2].replaceAll("\\s*$",
""));
                                                    if(v.size()>Principal.nummensajesmax)
                                                            v.remove(0);
                                                    v.add(mes);
                                            }
                                            offset=messageAndOffset.offset();
                                            System.out.println(offset);
                                            }
                                            i=v.size();
                                            if(i>0)
                                            {
                                                    int j=0;
                                                    int h=0;
                                                    while (j<i)
                                                    {
                                                    if(usuario.equals(v.get(j).getUsuario())){

                                                            j++;
                                                            h++;
                                                    }
                                                    else
                                                            if(evento.equals(v.get(j).getEvento())){

                                                                    h++;
                                                                    j++;
                                                            }
                                                            else
                                                                    j++;
                                                    }
                                            Object [][]ob=new Object[h][3];
                                             j=0;
                                             h=0;
```

```
                                        while (j<i)
                                        {
                                        if(usuario.equals(v.get(j).getUsuario())){
                                                ob[h][0]=v.get(j).getUsuario();
                                                ob[h][1]=v.get(j).getEvento();
                                                ob[h][2]=v.get(j).getFecha();
                                                j++;
                                                h++;
                                        }
                                        else
                                                if(evento.equals(v.get(j).getEvento())){
                                                        ob[h][0]=v.get(j).getUsuario();
                                                        ob[h][1]=v.get(j).getEvento();
                                                        ob[h][2]=v.get(j).getFecha();
                                                        h++;
                                                        j++;
                                                }
                                                else
                                                        j++;
                                        }

                                        final String[] columnNames = {"Usuario",
                                                        "Evento",
                                        "Fecha"};

                                        final MyTableModel myModel = new
MyTableModel(ob,columnNames);
                                        final JTable table = new JTable(myModel);

                                        table.setSelectionBackground(new Color(0, 0, 255));
                                        table.setBackground(new Color(0, 102, 153));
                                        table.setBorder(null);

                                        table.setFont(new Font("Dialog", Font.BOLD, 15));
                                        TableColumn columna;
                                        columna=table.getColumnModel().getColumn(0);
                                        columna.setPreferredWidth(110);
                                        columna.setMaxWidth(110);
                                        columna=table.getColumnModel().getColumn(1);
                                        columna.setPreferredWidth(95);
                                        columna.setMaxWidth(95);

                                        if(scrollPane!=null)
                                                Kafka.remove(scrollPane);
                                    //Creatamos un contenedor para la Tabla
                                        scrollPane = new JScrollPane(table);
                                    scrollPane.setBounds(new Rectangle(0, 2, 600, 500));
                                    scrollPane.getViewport().setBackground(new Color(51, 102, 153));
                                    scrollPane.getViewport().setBorder(null);
                                        Kafka.add(scrollPane);
                                        }
                                }
        }
}
```

B.1.51 Mensaje.java

```
package Presentacion;

public class Mensaje {
	public String Usuario;
	public String Evento;
	public String Fecha;



	public Mensaje(String Usuario, String Evento, String Fecha) {
	    this.Usuario = Usuario;
	    this.Evento = Evento;
	    this.Fecha = Fecha;
	    }



	  public String getUsuario(){
		return Usuario;
	  }
	  public String getEvento(){
		return Evento;
	  }
	  public String getFecha(){
		return Fecha;
	  }

	  public String toString() {
	    return "Mensaje-> Usuario: "+Usuario+" Evento: "+Evento+" Fecha: "+Fecha+"\n";
	  }
}
```

B.1.52 MyTableModel.java

```
package Presentacion;

import javax.swing.table.AbstractTableModel;

public class MyTableModel extends AbstractTableModel  {

	private static final long serialVersionUID = 1L;
	public String[] columnNames = null;
	public Object[][] data=null;

	public MyTableModel(Object[][] ob, String[] colum) {
		data = ob;
		columnNames=colum;
	}

	public int getColumnCount() {
		return columnNames.length;
	}

	public int getRowCount() {
		return data.length;
	}

	public String getColumnName(int col) {
		return columnNames[col];
	}

	public Object getValueAt(int row, int col) {
		return data[row][col];
	}

}
```

B.2 Código php

Como se indico al inicio del anexo no se va a incluir todo el codigo php, unicamente los más importantes.

B.2.1 funciones_bd.php

```php
<?php

class funciones_BD {
    private $db;
    // constructor
    function __construct() {
        require_once 'connectbd.php';
        //Establece conexion a la base de datos
        $this->db = new DB_Connect();
        $this->db->connect();
    }

    // destructor
    function __destruct() {
    }

        /**
     * Para agregar a un nuevo nuevo usuario a la base de datos
     */
    public function adduser2($Nomb, $Apell, $DNI, $Direc, $Movil, $usuario, $passw) {
    $result = mysql_query("INSERT INTO usuarios (nombre, apellidos, dni, direccion, movil, username, passw)
                                                VALUES('$Nomb', '$Apell', '$DNI', '$Direc', '$Movil', '$usuario', '$passw')");
        //Si se guardo el usuario de manera correcta
        if ($result) {
                    //Devuelve true
            return true;
                //si no fue de manera correcta
        } else {
                    //Devuelve false
            return false;
        }
    }

        /**
     * Para modificar los datos de un usuario en la base de datos
         */
        public function modifica($Nomb, $Apell, $DNI, $Direc, $Movil, $usuario, $passw) {
        $result = mysql_query("UPDATE `usuarios` SET `nombre`='$Nomb',`apellidos`='$Apell',
            `dni`='$DNI',`direccion`='$Direc',`movil`='$Movil',`passw`='$passw' WHERE username='$usuario'");
                    //Si se modifica el usuario de manera correcta
            if ($result) {
                        //Devuelve true
                return true;
                    //si no fue de manera correcta
            } else {
                        //Devuelve false
                return false;
            }
        }


            /**
         * Para enviar mail si se olvido contraseña
         */

                public function enviomail($user, $direccion){
                $result = mysql_query("SELECT * FROM usuarios
                WHERE username = '$user' and direccion = '$direccion'");
                //Comprueba que la dupla usuario-mail es correcta
                if (!$result){
                        //Si no existe dicha dupla, devuelve error
                        return false;
                //Si existe, se envia mail.
                }else{

                        $row = mysql_fetch_array ( $result );

                        //envio de mail
                        $header = 'From:  droidlogingestor@gmail.com '. " \r\n";
                        $header .= "X-Mailer: PHP/" . phpversion() . " \r\n";
                        $header .= "Mime-Version: 1.0 \r\n";
                        $header .= "Content-Type: text/plain";
```

```
                        $mensaje = "Este mensaje es enviado para el usuario: " . $user . " \r\n";
                        $mensaje .= "Con el siguiente e-mail: " . $direccion . " \r\n";
                        $mensaje .= "La contraseña del usuario es: " . $row [ "passw" ] . " \r\n";
                        $mensaje .= "Enviado el " . date('d/m/Y', time());

                        $para = $direccion;
                        $asunto = 'Recordatorio password';
                        //funcion que envia mail
                        mail($para, $asunto, utf8_decode($mensaje), $header);
                        //mail($para, utf8_decode($asunto), utf8_decode($mensaje));
                        return true;
                }
        }

    /**
    * Comprobacion de si el usuario ya existe por el username
    */

                public function isuserexist($username) {
        $result = mysql_query("SELECT username from usuarios WHERE username = '$username'");
        $num_rows = mysql_num_rows($result); //numero de filas devueltas
                //Si devuelve filas, el usuario existe
        if ($num_rows > 0) {
            return true;
        } else {
            //no existe
            return false;
        }
    }


        /**
    * Comprobacion de si el usuario ya existe por el username para nuevo usuario
    */
        public function usexist($user){
                $result=mysql_query("SELECT COUNT(*) FROM usuarios WHERE username='$user' ");
                $count = mysql_fetch_row($result);
                //usuario unico
                if ($count[0]==0){
                        return true;
                }else{
                        return false;
                }
        }

>
```

B.2.2 produce.php

```php
<?php
set_include_path(
        implode(PATH_SEPARATOR, array(
                realpath(__DIR__ . '/../lib'),
                get_include_path(),
        ))
);
require 'autoloader.php';
//$usuario=$_POST['usuario'];
//$evento=$_POST['evento'];
$usuario =str_pad($_POST['usuario'], 30);
$evento =str_pad($_POST['evento'], 30);
$host = 'localhost';
$port = 9092;
$topic = 'test';
$fecha =date('l jS \of F Y h:i:s A');
$fecha = str_pad($fecha,60);
$producer = new Kafka_Producer($host, $port, Kafka_Encoder::COMPRESSION_NONE);

$mensaje= $usuario . "-" . $evento . "-" . $fecha;
        $messages = explode(',', $mensaje);

        $bytes = $producer->send($messages, $topic);
$resultado[]=array("logstatus"=>"1");
//Codifica resultado en JSON
echo json_encode($resultado);
?>
```

B.3 Código android

Como se indico al inicio del anexo no se va a incluir todo el codigo android, unicamente los más importantes.

B.3.1 familia login-main

B.3.1.1 Login.java

```
package test.Droidlogin;
import java.util.ArrayList;
import org.apache.http.NameValuePair;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import test.Droidlogin.library.Httppostaux;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.SystemClock;
import android.os.Vibrator;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.graphics.PorterDuff;
public class Login extends Activity {
    /** Called when the activity is first created. */

    EditText user;
    EditText pass;
    Button BConfig;
   Button blogin;
    Button registrar;
    Button noregistrar;
    TextView forgetpssw;
    Httppostaux post;
    // String URL_connect="http://www.scandroidtest.site90.com/acces.php";
    String IP_Server;//IP DE NUESTRO PC
    String URL_connect;//ruta en donde estan nuestros archivos

    boolean result_back;
    private ProgressDialog pDialog;

    @Override
    public void onCreate(Bundle savedInstanceState) {
      final Context context = this;
        Global g = Global.getInstance();
        IP_Server=g.getData();
```

```
        URL_connect="http://"+IP_Server+"/droidlogin/acces.php";
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        post=new Httppostaux();

        user= (EditText) findViewById(R.id.edusuario);
        pass= (EditText) findViewById(R.id.edpassword);
        blogin= (Button) findViewById(R.id.Blogin);
      BConfig= (Button) findViewById(R.id.BConfig);
        registrar= (Button) findViewById(R.id.Bregistro);
        noregistrar= (Button) findViewById(R.id.BNoregistro);
        forgetpssw= (TextView) findViewById(R.id.forgotpassw);

        blogin.getBackground().setColorFilter(0xFF80FFFF, PorterDuff.Mode.MULTIPLY);
        registrar.getBackground().setColorFilter(0xFF80FFFF, PorterDuff.Mode.MULTIPLY);
        noregistrar.getBackground().setColorFilter(0xFF80FFFF, PorterDuff.Mode.MULTIPLY);
      BConfig.getBackground().setColorFilter(0xFF80FFFF, PorterDuff.Mode.MULTIPLY);
      //Login button action
        blogin.setOnClickListener(new View.OnClickListener(){

            public void onClick(View view){

                //Extreamos datos de los EditText
                String usuario=user.getText().toString();
                String passw=pass.getText().toString();
              Global g = Global.getInstance();
              g.setUsuario(usuario);          //verificamos si estan en blanco
                if( checklogindata( usuario , passw )==true){
                    //si pasamos esa validacion ejecutamos el asynctask pasando el usuario y clave
como parametros

                new asynclogin().execute(usuario,passw);


                }else{
                    //si detecto un error en la primera validacion vibrar y mostrar un Toast con un
mensaje de error.
                    err_login();
                }

            }
                                                    });

        registrar.setOnClickListener(new View.OnClickListener(){

            public void onClick(View view){

                Intent i=new Intent(Login.this, regscreen.class);
              startActivity(i);
                                    }
                                                        });


        noregistrar.setOnClickListener(new View.OnClickListener(){

            public void onClick(View view){
```

```
                Intent i=new Intent(Login.this, noregscreen.class);
                startActivity(i);
                                }
                                                        });

        forgetpssw.setOnClickListener(new View.OnClickListener(){

            public void onClick(View view){

                //Aqui saltaria a la pantalla del envio del e-mail
                Log.e("Olvido de contrasena", "Si");
                Intent i=new Intent(Login.this, forgetscreen.class);
                startActivity(i);
                                }
                                                        });
        BConfig.setOnClickListener(new View.OnClickListener(){
            public void onClick(View view) {
                Log.e("Configuracion", "Si");
                LayoutInflater layoutInflater = LayoutInflater.from(context);
                View promptView = layoutInflater.inflate(R.layout.prompts, null);
                AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(context);
                alertDialogBuilder.setView(promptView);
                final EditText input = (EditText) promptView.findViewById(R.id.userInput);
                alertDialogBuilder
                        .setCancelable(false)
                        .setPositiveButton("OK", new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog, int id) {
                                Global g = Global.getInstance();
                                Log.e("Edita G", "Si");
                                g.setData(input.getText().toString());
                                Log.e("Edita G", "Si");
                            }
                        })
                        .setNegativeButton("Cancel",
                                new DialogInterface.OnClickListener() {
                                    public void onClick(DialogInterface dialog, int id) {
                                        dialog.cancel();
                                    }
                                });
                AlertDialog alertD = alertDialogBuilder.create();
                alertD.show();
            }
        });

    }

    //vibra y muestra un Toast
    public void err_login(){
        Vibrator vibrator =(Vibrator) getSystemService(Context.VIBRATOR_SERVICE);
        vibrator.vibrate(200);
        Toast toast1 = Toast.makeText(getApplicationContext(),"Error:Nombre de usuario o password
incorrectos", Toast.LENGTH_SHORT);
        toast1.show();
    }

    //vibra y muestra un Toast si usuario correcto
```

```
    public void login_ok(){
        Vibrator vibrator =(Vibrator) getSystemService(Context.VIBRATOR_SERVICE);
        vibrator.vibrate(200);
        Toast toast1 = Toast.makeText(getApplicationContext(),"El usuario y la contraseña son
correctos", Toast.LENGTH_LONG);
        toast1.show();
    }

    /*Valida el estado del logueo solamente necesita como parametros el usuario y passw*/
    public boolean loginstatus(String username ,String password ) {
        int logstatus=-1;

        /*Creamos un ArrayList del tipo nombre valor para agregar los datos recibidos por los
parametros anteriores
         * y enviarlo mediante POST a nuestro sistema para relizar la validacion*/
        ArrayList<NameValuePair> postparameters2send= new ArrayList<NameValuePair>();

                    postparameters2send.add(new BasicNameValuePair("usuario",username));
                    postparameters2send.add(new BasicNameValuePair("password",password));
        Global g = Global.getInstance();
        IP_Server=g.getData();
        URL_connect="http://"+IP_Server+"/droidlogin/acces.php";
          //realizamos una peticion y como respuesta obtenes un array JSON
          JSONArray jdata=post.getserverdata(postparameters2send, URL_connect);
          /*como estamos trabajando de manera local el ida y vuelta sera casi inmediato
           * para darle un poco realismo decimos que el proceso se pare por unos segundos para
poder
           * observar el progressdialog
           * la podemos eliminar si queremos
           */
          SystemClock.sleep(950);

          //si lo que obtuvimos no es null
            if (jdata!=null && jdata.length() > 0){
              JSONObject json_data; //creamos un objeto JSON
              try {
                  json_data = jdata.getJSONObject(0); //leemos el primer segmento en nuestro caso
el unico
                    logstatus=json_data.getInt("logstatus");//accedemos al valor
                    Log.e("loginstatus","logstatus= "+logstatus);//muestro por log que obtuvimos
              } catch (JSONException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
              }

              //validamos el valor obtenido
                if (logstatus==0){// [{"logstatus":"0"}]
                    Log.e("loginstatus ", "invalido");
                    return false;
                }
                else{// [{"logstatus":"1"}]
                    Log.e("loginstatus ", "valido");
                    return true;
                }

          }else{   //json obtenido invalido verificar parte WEB.
                    Log.e("JSON  ", "ERROR");
```

```
                    return false;
                }

        }


        //validamos si no hay ningun campo en blanco
        public boolean checklogindata(String username ,String password ){

        if    (username.equals("") || password.equals("")){
            Log.e("Login ui", "checklogindata user or pass error");
        return false;

        }else{

            return true;
        }

}

/*    CLASE ASYNCTASK
  *
  * usaremos esta para poder mostrar el dialogo de progreso mientras enviamos y obtenemos los datos
  * podria hacerse lo mismo sin usar esto pero si el tiempo de respuesta es demasiado lo que podria
ocurrir
  * si la conexion es lenta o el servidor tarda en responder la aplicacion sera inestable.
  * ademas observariamos el mensaje de que la app no responde.
  */
        class asynclogin extends AsyncTask< String, String, String > {

            String user,pass;
              protected void onPreExecute() {
                  //para el progress dialog
                    pDialog = new ProgressDialog(Login.this);
                    pDialog.setMessage("Autenticando....");
                    pDialog.setIndeterminate(false);
                    pDialog.setCancelable(false);
                    pDialog.show();
              }

           protected String doInBackground(String... params) {
                //obtnemos usr y pass
                user=params[0];
                pass=params[1];

                //enviamos y recibimos y analizamos los datos en segundo plano.
                 if (loginstatus(user,pass)==true){
                      return "ok"; //login valido
                 }else{
                      return "err"; //login invalido
                 }

           }

           /*Una vez terminado doInBackground segun lo que halla ocurrido
           pasamos a la sig. activity
```

```
        o mostramos error*/
            protected void onPostExecute(String result) {
                pDialog.dismiss();//ocultamos progess dialog.
                Log.e("onPostExecute=","+result);

                if (result.equals("ok")){
                  Enviaevento envia = new Enviaevento();
                  Global g = Global.getInstance();
                  IP_Server=g.getData();
                  String usuario=g.getUsuario();
                  envia.enviamensaje(usuario,"Login");
                  Intent i=new Intent(Login.this, HiScreen.class);
                  i.putExtra("user",user);
                  login_ok();     //Aqui llamamos a una funcion que muestra un toast con un mensaje de
login correcto
                  startActivity(i);

                  }else{
                      err_login();
                  }

                                                          }

            }

        }
```

B.3.1.2 Main.xml

```
<LinearLayout
            xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="vertical"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:id="@+id/layout1"
            android:background="#3399FF">
            <!--  Usuario Label -->

            <TextView android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:textColor="#FFFFFF"
                    android:layout_marginTop="5dip"
                   android:text="Usuario"
              />

            <EditText android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dip"
                    android:layout_marginBottom="5dip"
                    android:singleLine="true"
                    android:id="@+id/edusuario"
                    android:background="@drawable/edittext_rounded_corners"

             />

            <!--  Password Label -->

            <TextView android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:textColor="#FFFFFF"
                    android:text="Password"/>

            <EditText android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="5dip"
                    android:singleLine="true"
                    android:password="true"
                    android:id="@+id/edpassword"
                    android:background="@drawable/edittext_rounded_corners"

                    />


            <!-- Layaout Login-Registro -->
           <LinearLayout
                 android:layout_width="match_parent"
                 android:layout_height="wrap_content"
                 android:orientation="horizontal">

            <!-- Login button -->
                 <Button
                      android:id="@+id/Blogin"
                      android:layout_width="256dp"
                      android:layout_height="wrap_content"
```

```
                android:layout_marginTop="15dip"
                android:text="Login" />

        <!-- Registro button -->
            <Button
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="15dip"
                    android:text="Registrarse"
                    android:id="@+id/Bregistro"/>

        </LinearLayout>

    <!-- Sin registro button -->
    <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="5dip"
            android:text="Continuar sin regsitro"
            android:id="@+id/BNoregistro"/>

        <!-- Link to forgot passw -->

        <TextView android:id="@+id/forgotpassw"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="¿Olvido su Password?"
                android:gravity="center"
                android:textSize="20dip"
                android:textColor="#FFFFFF"/>
      <Button
      android:layout_marginLeft="420dip"
      android:layout_width="150dp"
      android:layout_height="wrap_content"
      android:layout_marginTop="10dip"
      android:text="Configuración"
      android:id="@+id/BConfig"
       />
</LinearLayout>
```

B.3.1.3 main.xml

```
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:padding="10dip"
        android:id="@+id/layout1"
        android:background="@drawable/degradado"

        >
      <!-- Logo -->

        <ImageView android:layout_height="wrap_content"
                android:src="@drawable/android"
        android:id="@+id/img"
        android:layout_width="fill_parent"
        android:layout_marginTop="10dip"
        android:gravity="center" />


        <!-- Email Label -->

        <TextView android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:textColor="#FFFFFF"
                android:text="Usuario"
                android:layout_marginTop="20dip"
         />

        <EditText android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dip"
                android:layout_marginBottom="20dip"
                android:singleLine="true"
                android:id="@+id/edusuario"
                android:background="@drawable/edittext_rounded_corners"
         />

        <!-- Password Label -->

        <TextView android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:textColor="#FFFFFF"
                android:text="Password"/>

        <EditText android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dip"
                android:singleLine="true"
                android:password="true"
                android:id="@+id/edpassword"
                android:background="@drawable/edittext_rounded_corners"

                />
```

```
<!-- Layaout Login-Registro -->

<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:paddingRight="4px" >
      <!-- Login button -->
      <Button
           android:id="@+id/Blogin"
           android:layout_width="wrap_content"
           android:layout_height="fill_parent"
           android:layout_marginTop="25dip"
           android:layout_weight="0.56"
           android:text="Login" />
      <!-- Registro button -->
      <Button
           android:id="@+id/Bregistro"
           android:layout_width="134dp"
           android:layout_height="fill_parent"
           android:layout_marginTop="25dip"
           android:layout_weight="0.08"
           android:text="Registrarse" />
</LinearLayout>

<!-- Sin Registro button -->
  <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dip"
        android:text="Continuar sin registro"
        android:id="@+id/BNoregistro"
        />
 <!-- Link to forgot passw -->

 <TextView android:id="@+id/forgotpassw"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dip"
        android:layout_marginBottom="20dip"
        android:text="¿Olvido su Password?"
        android:gravity="center"
        android:textSize="20dip"
        android:textColor="#FFFFFF"/>
<Button
     android:layout_marginLeft="200dip"
     android:layout_width="150dp"
     android:layout_height="wrap_content"
     android:layout_marginTop="10dip"
     android:text="Configuración"
     android:id="@+id/BConfig"
/>



</LinearLayout>
```

B.3.2 Enviaenvento.java

```
package test.Droidlogin;
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.SystemClock;
import android.util.Log;
import org.apache.http.NameValuePair;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.ArrayList;
import test.Droidlogin.library.Httppostaux;
public class Enviaevento extends Activity{
    /** Called when the activity is first created. */
    Httppostaux post;
    // String URL_connect="http://www.scandroidtest.site90.com/acces.php";
    String IP_Server;//IP DE NUESTRO PC
    String URL_connect;//ruta en donde estan nuestros archivos
    @Override
    public void onCreate(Bundle savedInstanceState) {
        Global g = Global.getInstance();
        IP_Server = g.getData();
        URL_connect = "http://" + IP_Server + "/droidlogin/acces.php";
        super.onCreate(savedInstanceState);
        post = new Httppostaux();
    }
            public void onClick(String user, String evento){
                        //verificamos si estan en blanco
                    new asynclogin().execute(user, evento);
            }
    /*Envia el mensaje, solamente necesita como parametros el usuario y el evento*/
    public boolean enviamensaje(String username ,String evento ) {
        int envio = 0;
        URL_connect = "http://" + IP_Server + "/droidlogin/acces.php";
        new asynclogin().execute(username, evento);
        post = new Httppostaux();
       /*Creamos un ArrayList del tipo nombre valor para agregar los datos recibidos por los
parametros anteriores
        * y enviarlo mediante POST a nuestro sistema para relizar la validacion*/
        ArrayList<NameValuePair> postparameters2send = new ArrayList<NameValuePair>();
        postparameters2send.add(new BasicNameValuePair("usuario", username));
        postparameters2send.add(new BasicNameValuePair("evento", evento));
        Global g = Global.getInstance();
        IP_Server = g.getData();
        URL_connect = "http://" + IP_Server + "/kafka-php-master/src/examples/produce.php";
        //realizamos una peticion y como respuesta obtenes un array JSON
        JSONArray jdata = post.getserverdata(postparameters2send, URL_connect);
          /*como estamos trabajando de manera local el ida y vuelta sera casi inmediato
            * para darle un poco realismo decimos que el proceso se pare por unos segundos para
poder
            * observar el progressdialog
            * la podemos eliminar si queremos
            */
        SystemClock.sleep(50);
        //si lo que obtuvimos no es null
```

```
            if (jdata != null && jdata.length() > 0) {
                JSONObject json_data; //creamos un objeto JSON
                try {
                    json_data = jdata.getJSONObject(0); //leemos el primer segmento en nuestro caso el
unico
                    envio = json_data.getInt("envio");//accedemos al valor
                    Log.e("envio", "envio= " + envio);//muestro por log que obtuvimos
                } catch (JSONException e) {
                    e.printStackTrace();
                }
                //validamos el valor obtenido
                if (envio == 1) {// [{"logstatus":"0"}]
                    Log.e("envio ", "valido");
                    return true;
                }else
                    return false;
            } else {    //json obtenido invalido verificar parte WEB.
                Log.e("JSON  ", "ERROR");
                return false;
            }
        }
/*    CLASE ASYNCTASK
 *
 * usaremos esta para poder mostrar el dialogo de progreso mientras enviamos y obtenemos los datos
 * podria hacerse lo mismo sin usar esto pero si el tiempo de respuesta es demasiado lo que podria
ocurrir
 * si la conexion es lenta o el servidor tarda en responder la aplicacion sera inestable.
 * ademas observariamos el mensaje de que la app no responde.
 */
        class asynclogin extends AsyncTask< String, String, String > {
            String user,pass;
            protected void onPreExecute() {
                //para el progress dialog
            }
            protected String doInBackground(String... params) {
                //obtnemos usr y pass
                user=params[0];
                pass=params[1];
                    return "ok"; //login valido
            }
            /*Una vez terminado doInBackground segun lo que halla ocurrido
            pasamos a la sig. activity
            o mostramos error*/
            protected void onPostExecute(String result) {
                Log.e("onPostExecute=","+result);
                }
            }
        }
```

B.3.3 Global.java

```
package test.Droidlogin;
/**
 * Created by pgallegos11 on 5/08/15.
 */
public class Global {
        private static Global instance;
        // Global variable
        private String data="192.168.1.103";
        private String usuario;
        // Restrict the constructor from being instantiated
        private Global(){}
        public void setData(String d){
            this.data=d;
        }
        public String getData(){
            return this.data;
        }
    public void setUsuario(String usuario){
        this.usuario=usuario;
    }
    public String getUsuario(){
        return this.usuario;
    }
        public static synchronized Global getInstance(){
            if(instance==null){
                instance=new Global();
            }
            return instance;
        }
    }
```