Segunda Entrega DAS: UniStars

Joel M. García Escribano

Marzo 2023

Índice

1. Introducción

Partiendo del primer proyecto de la asignatura de *Desarrollo Avanzado de Software* en el que se ha creado una aplicación con nombre ***UniStars***, se pide complementarla con nuevas funcionalidades para la segunda entrega.

En este documento se encuentran en este orden:

1. Requisitos de valoración
2. Explicación de cada clase Java
3. Manual de Usuario
4. Enlace a repositorio Github

2. Requisitos de valoración

A continuación se describen los requisitos establecidos para la aplicación que se han conseguido cumplir:

2.1. Base de datos remota con Usuarios

- Se ha implementado una base de datos remota **MySQL** en el servidor proporcionado. Esta base de datos se utiliza para la gestión de los usuarios de la aplicación. Se han implementado 3 funcionalidades para los usuarios en un mismo archivo PHP. Para elegir la funcionalidad que desea la aplicación, la aplicación manda un parámetro que puede ser *Identificación*, *Registro* o *existeEmail* en una petición HTML con método "POST". Luego el servidor transmite la información al archivo PHP *usuarios.php* y según el parámetro realiza una operación u otra.

  Es importante remarcar que los usuarios no están ligados a un dispositivo; una persona puede tener tantos usuarios registrados como quiera sin límite. A continuación se detallan las 3 funcionalidades:

  1. **Identificación**:
     - Para mayor seguridad, la contraseña de cada usuario se guarda en el servidor después de cifrarla. De esta forma las contraseñas no se guardan en texto plano y en el caso de un ataque a la base de datos remota estarían mucho más protegidas.
     - Dado un email y una contraseña que escribe el usuario desde la aplicación, se mandan a un servidor Apache que redirecciona la petición a un archivo PHP local. El archivo PHP comprueba primero que el email existe y obtiene la clave cifrada de ese email para luego compararla con la escrita por el usuario desde la aplicación.
     - Si las credenciales coinciden, se inicia sesión con el usuario. Si no coinciden se muestra un mensaje en la aplicación diciendo que ha fallado el inicio de sesión.

  En la Fig. 1 se puede ver la actividad de Identificación

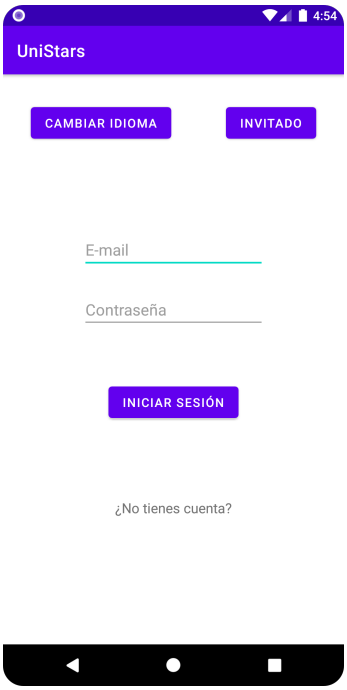

Figura 1: Actividad de Identificación de UniStars.

2. **ExisteEmail**
   - Antes de registrar un nuevo usuario se llama a esta función para comprobar que el email introducido no existe ya en la base de datos, ya que el email es un identificador único y por lo tanto no se puede repetir.
   - Dependiendo de si el email existe o no, devuelve a la aplicación el mensaje *NoExiste* o *Existe*, que la aplicación interpretará para notificarle al usuario que introduzca otro email con un mensaje.
3. **Registro**
   - Después de comprobar que el email introducido no está registrado actualmente, la aplicación manda la petición de registrar al usuario con el email y la contraseña.
   - El archivo PHP guarda el email junto con la contraseña en la tabla *t_usuarios*. La contraseña se cifra antes de guardarla para una mayor seguridad en caso de ataque a la base de datos.
   - Tras registrar al usuario en la base de datos, se manda a la aplicación el mensaje *Bien* para notificar que se ha registrado correctamente. Este es el comportamiento que debería seguir siempre, pero si surge algún error al registrar al usuario, se manda a la aplicación el mensaje *Mal*.

   En la Fig. 2 se puede ver la actividad de Registro

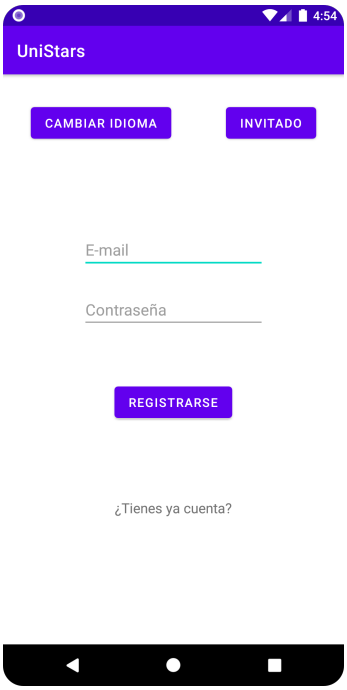

Figura 2: Actividad de Registro de UniStars.

2.2. Mensajeria Firebase Cloud Messaging

- Se pide realizar la implementación del servicio de Firebase Cloud Messaging o también llamado FCM. FCM es un servicio de mensajería en la nube que permite a los desarrolladores enviar notificaciones a los usuarios de sus aplicaciones.
- En este proyecto se ha utilizado de forma que cuando se añade una imagen a una universidad el usuario recibe una notificación de Firebase agradeciéndole que haya subido una imagen de la universidad a la base de datos de imágenes alojada en Firebase. En la Fig. 3 se puede ver el mensaje recibido cuando se añade una imagen a una universidad.

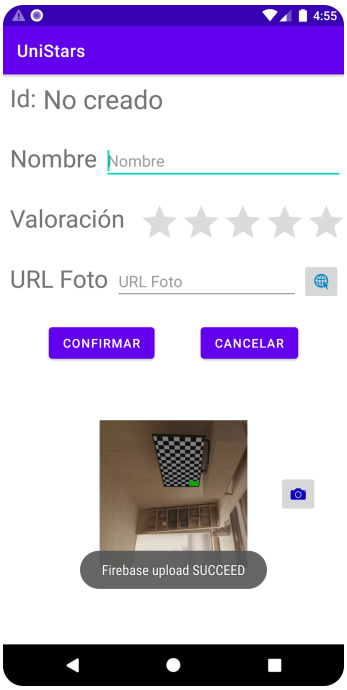

Figura 3: Mensaje que indica que se ha recibido un mensaje de Firebase.

- Además de este mensaje, aparece una notificación en el dispositivo que se puede ver en la Fig 4

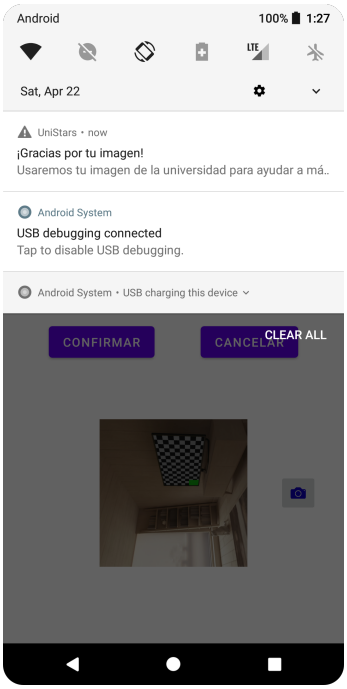

Figura 4: Notificación de Firebase al subir una imagen.

- Para implementar todo, primero se ha tenido que registrar la aplicación con Firebase. Además de modificar las dependencias del proyecto para incluir las dependencias de mensajería de Firebase, se han creado dos clases nuevas para realizar las operaciones necesarias para las operaciones y se ha creado un nuevo fichero PHP *notificacionFirebase.php*.

- Se han tenido que añadir las dependencias de Firebase a la aplicación. Para esto hacía falta modificar el manifiesto y crear una clase específica para gestionar los mensajes recibidos desde Firebase.

- Para poder comprobar el funcionamiento de la implementación del servicio de Firebase en la aplicación de forma externa tal y como se pide, se ha creado un archivo PHP que, cuando se accede a él desde un navegador, manda una notificación a todos los dispositivos. Esto se ha conseguido haciendo que cada vez que se genera un token para un dispositivo, se añade el dispositivo a un tema llamado *ALERTS* en Firebase. Cuando se accede al fichero PHP desde un navegador se trata de un método GET a la página, lo que manda una petición a Firebase para que se mande una notificación a todos los dispositivos que tengan el tema *ALERTS*.

  El enlace al fichero es: http://ec2-54-93-62-124.eu-central-1.compute.amazonaws.com/jgarcia424/WEB/notificacionFirebase.php.

- En la Fig. 5 se puede ver la notificación recibida en todos los dispositivos después de clicar el link.

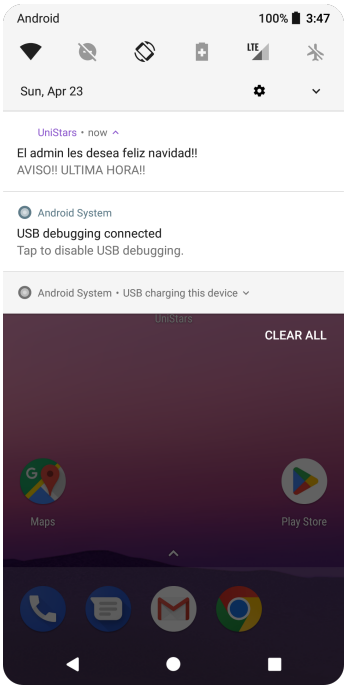

Figura 5: Notificación de Firebase al abrir el enlace desde el navegador.

- Por defecto, si el usuario está dentro de la aplicación cuando se recibe un mensaje, no se le muestra la notificación pertinente. Para poder demostrar el correcto funcionamiento de la mensajería Firebase se ha hecho que cuando llegue un mensaje siempre se muestre la notificación, aunque el usuario esté en la aplicación

2.3. Almacenamiento de imágenes en la nube

- Se pide la posibilidad de subir imágenes sacadas desde el dispositivo a una base de datos remota para luego descargarlas y mostrarlas en la aplicación.
- En esta aplicación se ha implementado de forma que cuando se añade una nueva universidad o se edita una ya existente, se puede sacar una foto del logo de una universidad para guardarlo en una base de datos en la nube.
- En la Fig. 6 se puede ver con un círculo rojo el botón de la cámara que abre la cámara del dispositivo para poder subir la imagen a la nube.

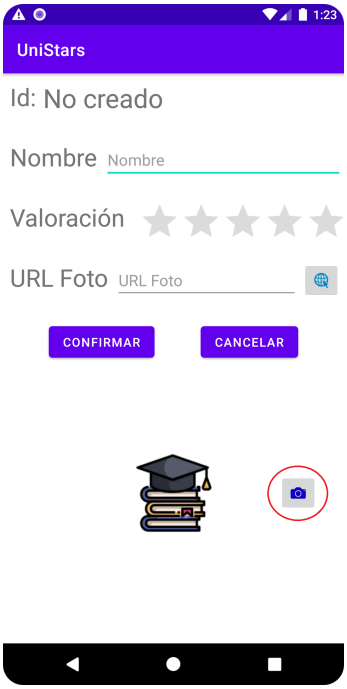

Figura 6: Actividad en la que se puede subir una imagen a Firebase.

- Tras hacer la foto, se asocia la universidad creada con la dirección de la imagen de Firebase, por lo que es persistente.

3. Explicación de cada clase Java

Solo se detallan las clases nuevas del segundo proyecto, no las ya creadas para el primer proyecto.

3.1. DbUsuarios y NotificationFirebase

- Ambas clases tienen el propósito de realizar una conexión asíncrona a un fichero php del servidor. Extienden de la clase Worker para ello
- DbUsuarios se comunica con el fichero *usuarios.php* para identficiar un usuario, registrarlo y comprobar si existe. Para ello toma los parámetros del email y la contraseña además de qué funcionalidad de las 3 se va a realizar.
- NotificationFirebase utiliza el archivo *notificacionFirebase.php* para comunicarle al archivo que mande una orden a Firebase para mandar una notificación al dispositivo. Esto lo hace mandándole el *token* o identificador único del dispositivo.
- Las dos clases imprimen logs de la información que mandan y reciben para poder arreglar problemas en caso de que ocurran errores en la conexión

3.2. ServicioFirebase

- Es la clase responsable de recibir las notificaciones de Firebase y poder tratarlas. Tiene sólo dos métodos:
  - **onNewToken:** Cuando se crea el token del dispositivo añade el dispositivo al tema de Firebase *ALERTS* para poder mandar un mensaje a todos los dispositivos con ese tema haciendo click en un link.
  - **onMessageReceived:** Por defecto, cuando se recibe una notificación de Firebase, si el usuario está dentro de la aplicación no se muestra la notificación. Para poder mostrarla cuando el usuario está dentro de la aplicación en este método se fuerza la creación de la notificación con el título y el mensaje recibido de Firebase.

3.3. AnadirOtro

- Esta actividad ya estaba presente en la anterior entrega, pero se ha modificado bastante para poder implementar las funcionalidades requeridas.
- Se ha añadido la posibilidad de hacer una foto a la universidad sin tener que poner la url de una imagen de internet.
- Aunque se ha mantenido la posibilidad de añadir una imagen de la universidad poniendo la url de su imagen, si se ha añadido una imagen desde la cámara, esta ultima sobreescribirá siempre la imagen de la url. En la Fig 7 se puede ver la actividad *AnadirOtro* cuando utiliza el url de la imagen y no se ha utilizado una foto.

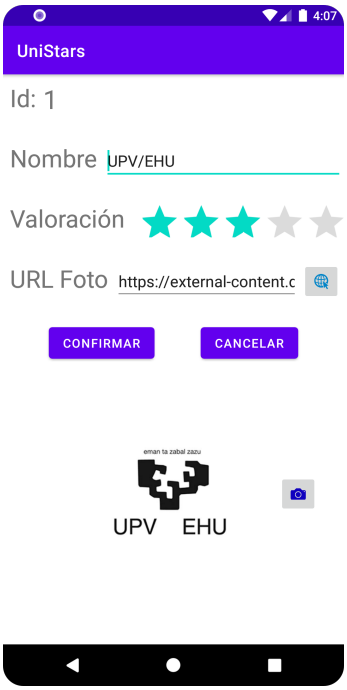

Figura 7: Actividad AnadirOtro con la url de la imagen.

En la Fig 8 se ve la misma actividad pero después de haber guardado una imagen sacada desde el dispositivo

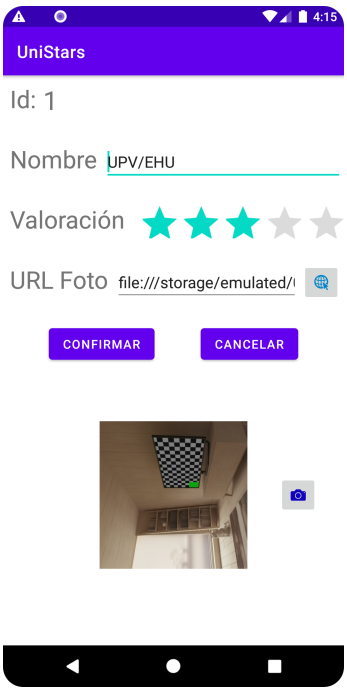

Figura 8: Actividad AnadirOtro con la foto.

4. Explicación de archivos PHP

4.1. usuarios.php

Se utiliza para realizar una de 3 funcionalidades distintas relacionadas con la base de datos que controla los usuarios registrados. Permite identificar a un usuario, registrarlo y comprobar que no existe ya en la base de datos. Se accede a él mediante un POST junto con los datos necesarios

- Dado un parámetro se elige la funcionalidad de las 3 que va a ejecutar.
- Si se identifica un usuario, simplemente se comprueba que su email y contraseña dados después de cifrarlos coinciden con los registrados ya en la base de datos. De ser así devuelve que se ha realizado correctamente.
- Si se pide registrar un nuevo usuario en la base de datos, primero se cifra la contraseña dada y luego se registra el usuario. Si no se puede registrar es porque ya existe ese email en la base de datos, por lo que se le notifica al usuario.
- La aplicación puede preguntarle a la base de datos si existe un email ya para desde la aplicación no permitir un registro de un usuario ya existente. Es un paso que se mantiene para evitar posibles errores.

4.2. notificacionFirebase.php

Para acceder a Firebase Cloud Messaging es necesario realizar consultas desde este fichero PHP. Desde la aplicación se le llama con un *token* para que Firebase envíe una notificación al dispositivo con ese *token* agradeciéndole que haya añadido una imagen a la base de datos.

- Cuando la aplicación llama a este fichero para lo anteriormente descrito, lo hace con un método POST, por lo que activa esa funcionalidad
- Sin embargo, cuando se accede por navegador al fichero, este detecta que se ha accedido a él por un método GET y tiene una funcionalidad distinta, que es la de mandar una notificación a todos los dispositivos con la aplicación. El link para acceder a esta funcionalidad es: http://ec2-54-93-62-124.eu-central-1.compute.amazonaws.com/jgarcia424/WEB/notificacionFirebase.php.
- En la Fig. 9 se puede ver la página resultante después de haber accedido al link. Primero muestra el código del mensaje enviado y luego se avisa de que se ha mandado un mensaje a todos los móviles de forma satisfactoria.

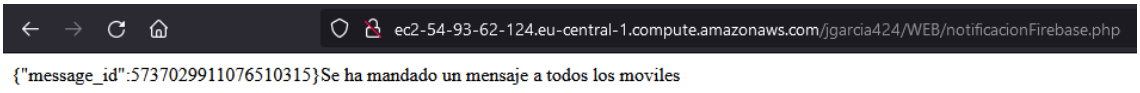

Figura 9: Página en la que se avisa de que se ha mandado una notificación a todos los dispositivos.

5. Manual de Usuario

Al iniciar la aplicación lo primero que verá el usuario es la actividad de identificación. Ver Fig.10

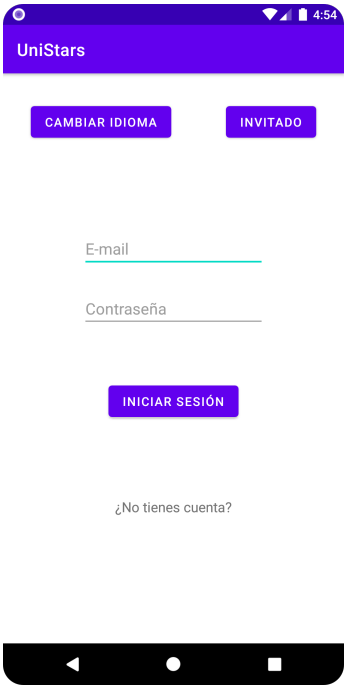

Figura 10: Actividad de Identificación de UniStars.

Se puede crear una nueva cuenta pulsando ¿No tienes cuenta?. Por defecto se puede acceder con las siguientes credenciales:

- E-mail: invitado@gmail.com
- Contraseña: 123

Tras iniciar sesión se presenta la lista de universidades, en este momento vacía. Se pueden añadir universidades por defecto pulsando el botón marcado con el círculo rojo y se puede añadir una nueva universidad. Ver Fig. 11

Figura 11: Menú principal de UniStars.

En este caso, se pulsará el botón *Añadir Otro*. Se mostrará entonces la siguiente ventana. Ver Fig. 12

Figura 12: Actividad de creación de una nueva universidad.

Para la funcionalidades pedidas solo hace falta pulsar el botón de la cámara ubicado abajo a la derecha. Después el usuario tomará una foto de la universidad y aceptará. Se mostrará entonces la imagen 13 junto con una notificación agradeciendo al usuario la nueva imagen guardada en la base de datos remota.

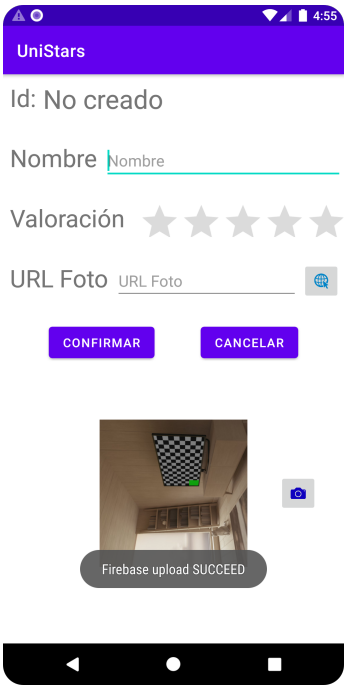

Figura 13: Se ha subido satisfactoriamente la imagen.

Si ahora se pulsa confirmar, se guardará la universidad junto con el enlace de la nueva foto y permanecerá en el dispositivo de forma local.

6. Enlace a repositorio Github

El repositorio en Github se encuentra en la dirección korah91/SegundoProyectoAndroid