ESAT INNOVATION SCHOOL | ESCUELA SUPERIOR DE ARTE Y TECNOLOGÍA

Programación Mobile 2020-2021

KIBO

Nombre Asignatura: Programación Mobile
Profesor: Rubén Blanco
Curso: 2020/2021
Autor/es: Alexandre Calabuig Langa

Índice/Index

1.- INTRODUCCIÓN

La APP desarrollada se llama KIBO, la cual es una aplicación companion que se ejecuta en dispositivos Android y que permite al usuario acceder con un correo y contraseña, para poder consultar su ranking, historial de partidas e ítems del personaje que teóricamente tendría un usuario que usaría la aplicación frecuentemente.

El objetivo de la aplicación es la retención de usuarios del hipotético juego, ampliando su mercado a las plataformas móviles.

Imagen 1: Splash screen con el logo rotatorio

2.- CARACTERÍSTICAS

2.1 Estructura

A nivel de código, cada sección es una Activity predeterminada, y en alguna de ellas, se añade un container donde se introduce un FragmentList, en el cual se podrá visualizar el contenido de la sección deseada.
En el proyecto se ha tenido que acceder a JSON ubicados en internet, por lo tanto, se han tenido que realizar clases para poder manejar los datos obtenidos del JSON.

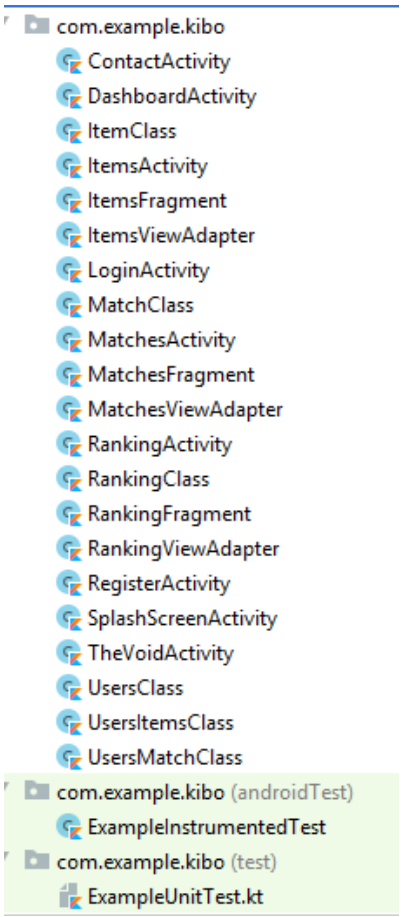


Imagen 2: Clases y Activity de la aplicación

2.2 Patrones de desarrollo

Al comienzo del desarrollo, se han realizado tests para saber cómo recoger los datos del JSON, ya sea vía archivos locales o por archivos ubicados en internet, así como otro tipo de acciones como el uso de los fragment list y cómo se pueden rellenar.

Una vez queda claro cómo se van a realizar todas las cosas, se empieza a diseñar cuáles van a ser la clase de JSON y qué elementos se van a recoger de ellos, para poder crearlos y subirlos a un repositorio en internet.

A continuación, se crean las actividades que van a ser necesarias y se empieza a trabajar por el login del usuario. Es importante empezar por aquí porque se va a necesitar los datos del usuario para acceder a los datos del mismo durante la aplicación.

Una vez empezando a programar, se separa en funciones lo que se desea empaquetar, como, por ejemplo, las funciones de cambiado de fondo de pantalla, o la creación de un Thread para poder acceder a datos online.
Después, se han hecho las Activity en las que no son necesarias el acceso a internet y recogida de datos a través de threads, para tener esas partes ya acabadas y funcionando de una Activity a otra.

Una vez terminadas las Actividades más básicas, se acaba con la realización una por una de las Actividades con acceso a internet y recogida de datos JSON.
Al hacer todas estas Actividades seguidas, se hacen más rápidas que de forma separadas, ya que al acostumbrarse a saber qué tipo de arrays tocar y a qué parámetros acceder, esta parte que puede llegar a ser un poco liosa, se realice de forma más fácil.

Al terminar toda la funcionalidad, se revisa todo el código y el diseño del mismo, ajustando los posibles problemas y se comenta todo adecuadamente (al menos a las funciones necesarias) para que cualquier desarrollador que tenga que acceder al código, sepa qué es cada elemento.

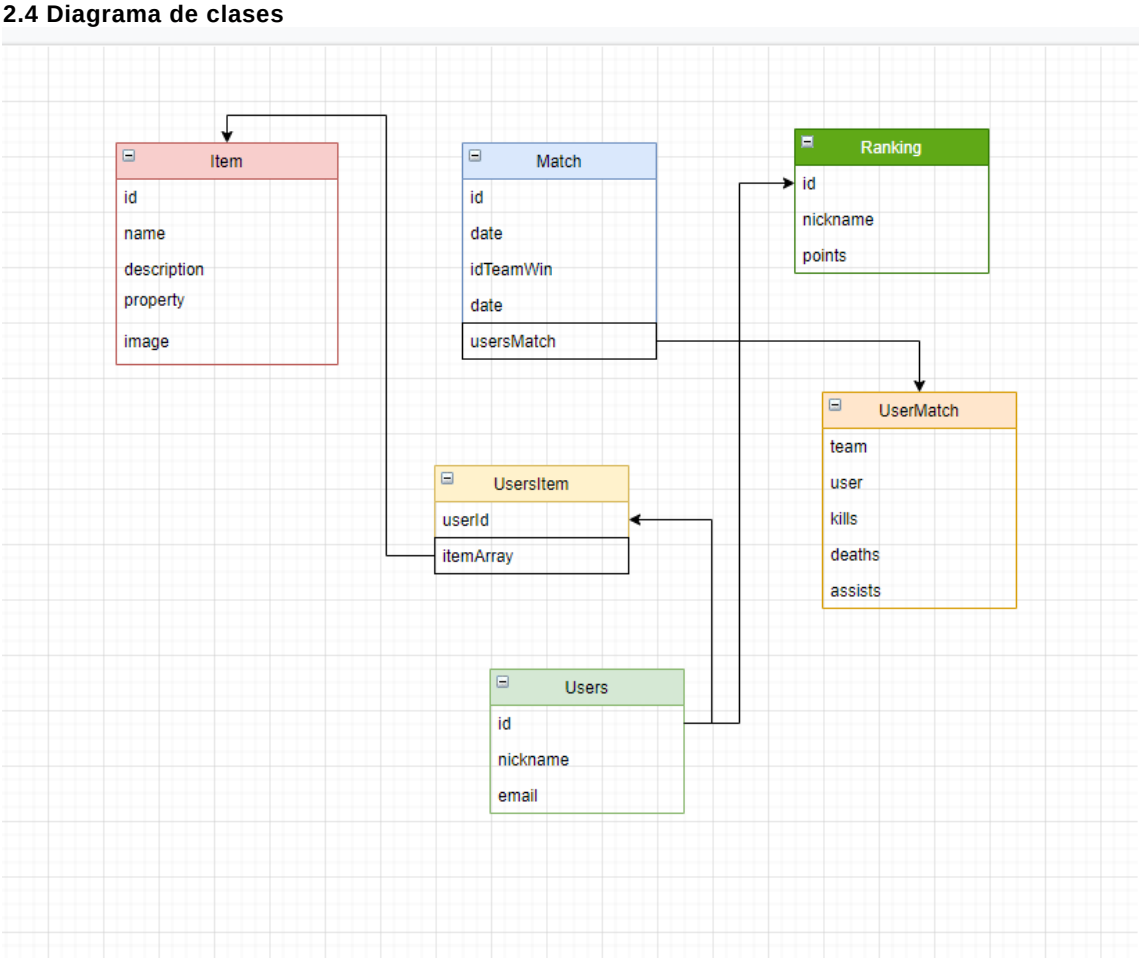
2.3 Tests

En el Unit Test se ha realizado tests, principalmente del acceso a JSONs como se ha comentado con anterioridad. Para demostrar que las clases están correctamente formateadas, así como el JSON, se ponen a una dirección concreta del array para demostrar que la información que se busca es la que desde el primer momento se ha buscado.

```
var userItemsArray: ArrayList<UsersItemsClass> = ArrayList()
var itemsFromUser: ArrayList<ItemClass> = arrayListOf()
@Test
fun getId_isCorrect() {

    var userArray = ("[{'id': 0,'nickname': 'rEkeD', 'email': 'reked@esat.es'}, "
        + "{'id': 1,'nickname': 'Soriano','email': 'soriano@esat.es'}, "
        + "{'id': 2,'nickname': 'BadBoy','email': 'badboy@esat.es'}]")

    val userListType: Type = object : TypeToken<ArrayList<UserClass?>?>() {}.type
    val usy: ArrayList<UserClass> = Gson().fromJson(userArray, userListType)

    assertEquals( expected: 0, usy[0].id)
}

@Test
fun getNickname_isCorrect(){
    var userArray = ("[{'id': 0,'nickname': 'rEkeD', 'email': 'reked@esat.es'}, "
            + "{'id': 1,'nickname': 'Soriano','email': 'soriano@esat.es'}, "
            + "{'id': 2,'nickname': 'BadBoy','email': 'badboy@esat.es'}]")

    val userListType: Type = object : TypeToken<ArrayList<UserClass?>?>() {}.type
    val usy: ArrayList<UserClass> = Gson().fromJson(userArray, userListType)

    assertEquals( expected: "Soriano",usy[1].nickname)
}
```

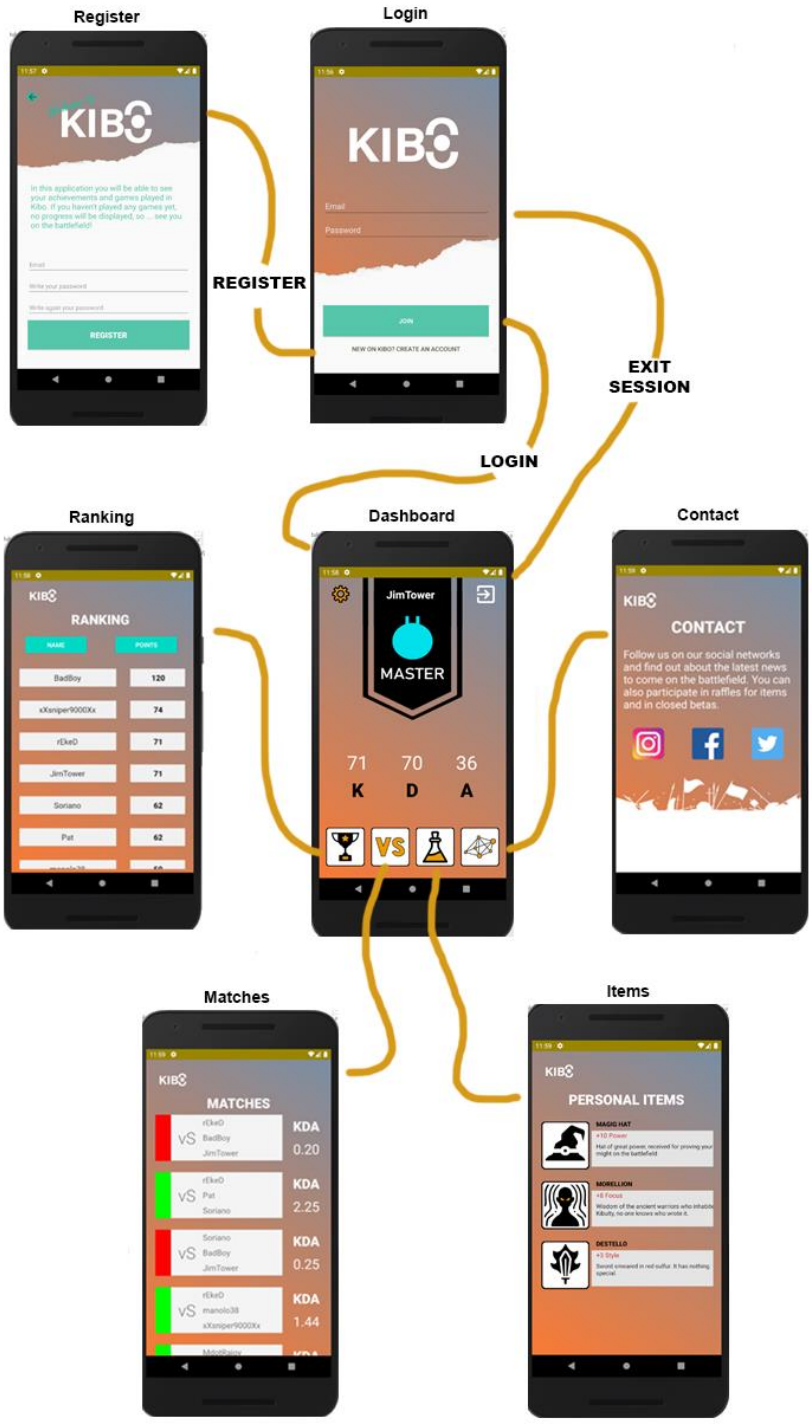
Imagen 3: Ejemplo de algunas funciones en el test

2.4 Diagrama de clases



3.- UI/UX FLOWCHART

(Mostrando estilo moqups, como se interactúa y como se va de una pantalla a otra.



4.- SOBRE EL DESARROLLO

4.1 Problemas durante la práctica

Durante el transcurso de la práctica, ha habido problemas en varias ocasiones:

- La recogida de datos de internet por JSON: Para poder usar datos de internet, se tiene que usar un Thread para poder acceder a esos datos, por lo que se intentaba acceder a esos datos dentro del Fragment creado para cada sección de la aplicación. Pero no se puede poner un Thread como siempre dentro del fragment, y cuando lo intentaba, salían errores y fallos varios.

- Creación de un menú para la selección del background: En la aplicación, el usuario puede elegir el color del background, por ello se pensó en hacer un menú desplegable que mostrara las opciones disponibles, pero tras varios intentos, el menú no salía como esperaba.

4.2 Solución de los problemas

Para los anteriores problemas se han hecho las siguientes soluciones:

- La recogida de datos de internet por JSON: Como daba problemas al hacerlo desde un fragment, descubrí que al fragment se pueden pasar variables y arrays por parámetro, por lo que la recogida de datos se hace en la Activity, y una vez procesados los datos necesarios que necesito en el fragment, paso esos datos a través de un commit en el container introducido en la Activity. Al pasar esos datos, también se puedn pasar al RecyclerView, por lo que podermos mostrar los datos que hagan falta.

- Creación de un menú para la selección del background: Como finalmente no salió correctamente la creación del menú, se me ocurrió una idea mejor; que no tuviera menú. Simplemente al presionar una vez encima del botón de opciones, se cambia de color. Así, no hace falta entrar a un menú y puedes pulsar de manera rápida en ese botón, por lo que finalmente al no haber un menú desplegable, la utilización de esa función es más rápida y con menos código que realizar.

4.3 Posibles mejoras

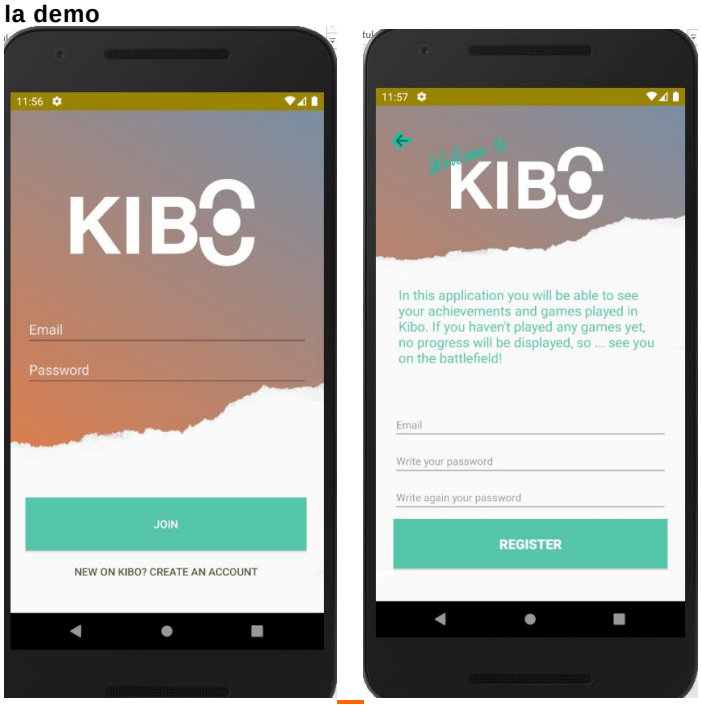
Conforme se ha ido avanzando en la realización del código, se ha visto que al final hay cosas que se podrían haber mejorado, como recoger todos los datos necesarios al principio de la aplicación y guardarlos de forma local hasta el cierre de sesión, en lugar de cargarlos cada vez que entras en la sección que se necesita. Se podría haber hecho eso para que sólo lo cargase una vez y la aplicación fuera rápida durante todo el rato.
También se podría haber puesto el código de recoger el background en cada activity si se hubiera puesto en una clase aparte y sólo se necesitará acceder a través de una llamada a función.
A nivel de estructura o de funcionalidad, en la sección de ítems se podría haber puesto una segunda vista de ítems con todos los que existen, para que el usuario sepa cuántos hay y cuántos tiene.
También me hubiera gustado mejorar la sección de Contacto, para que fuera más interactiva y tuviera más contenido, además de que se pudiera registrar y ya acceder a unos datos predeterminados y se cargara en un JSON los datos nuevos.

5.- MANUAL DE USUARIO

Hay dos formas de poder ejecutar la aplicación:

- Si se quiere instalar el APK en el teléfono móvil, vaya a la sección APP→APK y coja el archivo para descargarlo en el dispositivo móvil. Una vez tengas el archivo en el dispositivo, podrás ejecutarlo e instalarlo.
- Si se desea ejecutar desde Android Studio, abra la aplicación de Android y desde “Open”, seleccione el archivo de ejecución de APP→CODE. Una vez abierto, espere a que se haga la build y una vez finalizada, en la parte superior derecha se hace clic en el botón de “Play” para que se abra un dispositivo virtual, y ya podrá disfrutar de la aplicación.

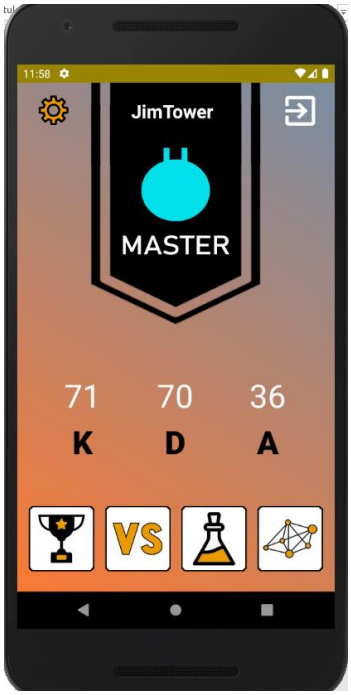
5.1 Uso de la demo



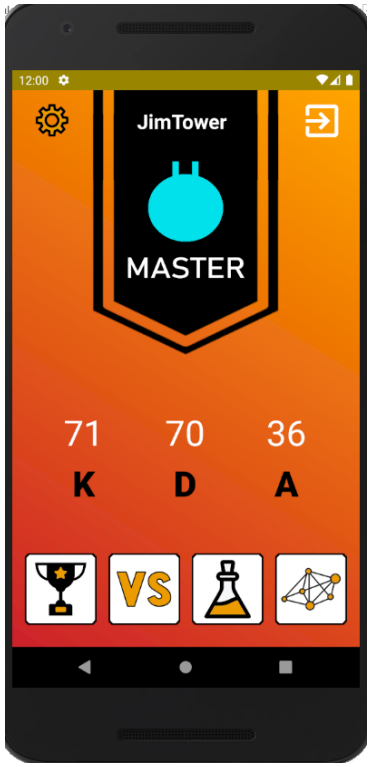
Al inicio de la aplicación podrás hacer login o registrarse. Como el registro actualmente llevaría a una pantalla en la que te dice que tendrías que tener partidas jugadas, se inicia la sesión con alguno de los siguientes emails y contraseñas:

reked@esat.es
sniper@esat.es
emedot@esat.es
jimtower@esat.es
badboy@esat.es
rita@esat.es
manolo38@esat.es
pat@esat.es
soriano@esat.es

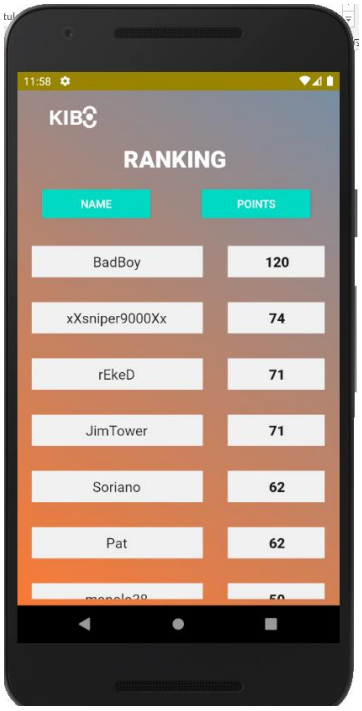
Email de todos: 123456



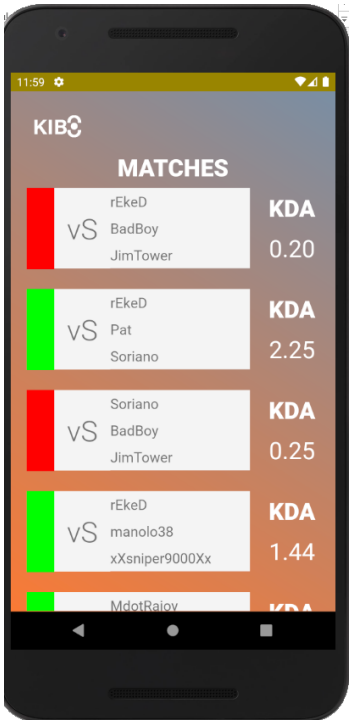
Este es el dashboard en el que podrás ver el total de muertes y asistencias, además de poder cambiar el color de fondo de pantalla al gusto del usuario.



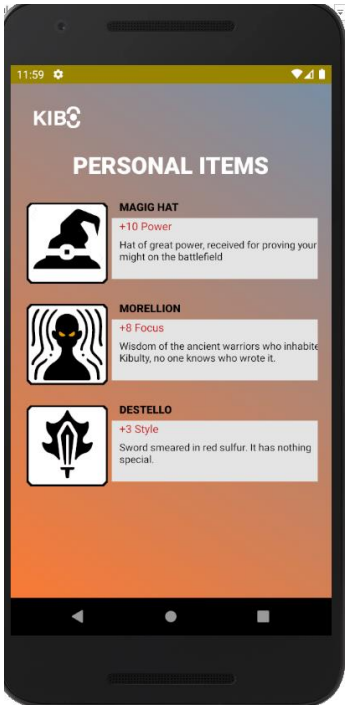
En la parte superior izquierda podrás cambiar el color haciendo clic una vez, y en la parte superior derecha cerrarás sesión y volverás al login. En la parte inferior podrás acceder al Ranking, Historial de partidas, Items y Contacto.



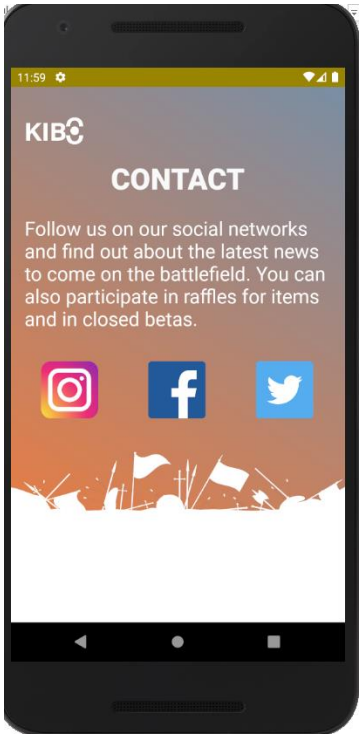
En el ranking verás los puntos de los usuarios, y podrás organizarlo por puntos o por nombre de usuario.



En el historial de matches, podrás ver tus resultados, contra quién has jugado y el KDA.



En la sección de personal ítems podrás ver los ítems propios del jugador, con su descripción y atributos.



Finalmente en la sección de contacto podrás acceder a las cuentas del desarrollador, si haces clic en las imágenes, te llevará fuera de la aplicación.