CICLO FORMATIVO DE GRADO SUPERIOR
DESARROLLO DE APLICACIONES MULTIPLATAFORMA

PROYECTO FIN DE CICLO

Move Your School

Raúl Ordás Fernández, Andrés Pérez Gómez y Cristina Díez Sobrino

CURSO 2018-19

TÍTULO: Move Your School

AUTORES: RAÚL ORDÁS FERNÁNDEZ

ANDRÉS PÉREZ GÓMEZ

CRISTINA DÍEZ SOBRINO

TUTOR DEL PROYECTO: Ernesto Ramiro Córdoba

FECHA DE LECTURA: Junio 2019

En Madrid

Ernesto Ramiro Córdoba
Tutor del PFC

RESUMEN:

Hemos observado que muchas instituciones educativas consideran tedioso encontrar diferentes actividades para sus alumnos debido a que deben buscar en distintas plataformas información sobre las excursiones. Queremos facilitar esta búsqueda con una aplicación que contenga todos los datos sobre varias actividades para grupos educativos. Mediante nuestra aplicación se podrán subir detalles de las distintas actividades que las empresas u organizaciones ofrecen para centros educativos. Gracias a esta aplicación el profesorado podrá organizar de manera sencilla las actividades más convenientes para sus alumnos.

Por tanto, nuestra aplicación está dirigida tanto como a profesores en busca de actividades para sus alumnos, como para empresas u organizaciones que ofrezcan eventos educativos, excursiones orientadas para jóvenes, y actividades infantiles. De esta manera los centros escolares podrán encontrar fácilmente actividades que les interesen y las empresas podrán ofrecer información sobre sus servicios de una manera sencilla y llegar a un público mayor.

Mediante las valoraciones de los usuarios se podrá obtener información sobre las actividades más interesantes para los docentes que podrán filtrar por distintas áreas para facilitar la búsqueda.

Nuestro objetivo principal es proporcionar sencillez a la hora de organizar actividades curriculares y extracurriculares para los colegios, institutos, academias, universidades… Y promover la comunicación entre empresas e instituciones con objetivo educacional.

Adicionalmente, creemos que es una buena herramienta para pequeñas empresas y organizaciones que pueden ofrecer sus ofertas y servicios para grupos escolares y ser recomendadas por los usuarios. Esta puede ser una forma eficaz de hacer publicidad que llegue al tipo de cliente que buscan.

ABSTRACT:

We have observed that many educational institutions consider tedious to find different activities for their students because they must search in different information-platforms about the excursions. We want to facilitate this search with an application that contains all the data about various activities for educational groups. Through our application you can upload details of the different activities that companies or organizations offer for educational centers. Thanks to this application teachers can easily organize the most convenient activities for their students.

Therefore, our application is aimed both at teachers in search of activities for their students and at companies or organizations that offer educational events, guided excursions for young people, and children's activities. In this way schools can easily find activities that interest them and companies can offer information about their services in a simple way and reach a larger audience.

Through user ratings, information about the most interesting activities can be obtained for teachers who can filter through different areas to facilitate the search.

Our main objective is to provide simplicity when organizing curricular and extracurricular activities for schools, institutes, academies, universities ... And promote communication between companies and institutions for educational purposes.

Additionally, we believe it is a good tool for small businesses and organizations that can offer their offers and services to school groups and be recommended by users. This can be an effective way of advertising that reaches the type of customer they are looking for.

AGRADECIMIENTOS

Nos gustaría agradecer a los padres de Cristina, ambos maestros, por ofrecernos la idea que hizo que este proyecto surgiera.

También agradecer a nuestro profesor Ernesto el seguimiento de nuestra actividad para su correcta elaboración.

Por supuesto, agradecer a los compañeros de proyecto que han puesto todo su esfuerzo en esta aplicación y han trabajado como un magnífico equipo ayudándose los unos a los otros.

ÍNDICE

1. INTRODUCCION

El principal objetivo de este PFC es realizar una aplicación útil, sencilla y manejable para que los centros docentes y otras instituciones educativas puedan organizar excursiones de la manera más cómoda posible. Además debe abastecer las necesidades de búsqueda de los usuarios con filtros y otras herramientas, que ayuden a encontrar eventos acordes con la edad de los alumnos o el tipo de asignatura.

Pensamos que esta herramienta puede despertar el interés de las empresas organizadoras de eventos ya que pueden ofertar sus tarifas y servicios llegando a un gran número de gente. Además, es una herramienta que facilita el contacto con universidades, colegios, institutos, academias… que llevan grandes números de alumnos a estas actividades.

Con estos objetivos hemos desarrollado *Move Your School,* una aplicación para dispositivos Android que ofrece información sobre excursiones y eventos organizados por empresas u organizaciones. Tiene un método de búsqueda para el usuario y varios filtros para facilitar el manejo de la aplicación. Los usuarios se pueden registrar como institución educativa o empresa organizadora de eventos.

A nivel gráfico queremos que sea sencilla y muy visual para que el usuario se encuentre a gusto utilizándola y no haya ningún tipo de confusión.

Las herramientas usadas para este proyecto han sido principalmente *Android Studio y Firebase*, las cuales pueden ser consultadas para mayor información en el apartado de herramientas tecnológicas.

1.1. Objetivos

Los objetivos de *Move Your School* son:

- Crear una aplicación Android funcional y manejable.
- Crear una base de datos Firebase que guarde los datos de la aplicación.
- Desarrollar una interfaz limpia y sencilla.
- Promover la diversidad de eventos y excursiones

1.2. Motivación

Este proyecto surgió por nuestra inquietud cultural y por la detección de un problema a la hora de encontrar actividades en los centros educativos.

La idea surgió después de que Cristina, una integrante del proyecto, escuchara a sus padres (ambos maestros) hablando sobre las dificultades que les suponía indagar en internet y otras fuentes para obtener información de actividades adecuadas para sus alumnos.

Nos pareció una idea brillante para nuestro PFC porque cubre nuestros conocimientos y podría incluso ser utilizada por nuestro centro de educación, la Universidad Europea.

1.3. Antecedentes

Hemos elegido la plataforma de aplicación móvil para nuestro proyecto. Esto se debe al gran crecimiento que se prevé para el futuro para las aplicaciones Android.

En la *Figura 1* se presenta el número de descargas por todo el mundo en 2017, 2018 y el pronóstico de descargas para 2022. En 2017 los consumidores descargaron 178.1 billones de aplicaciones móviles en sus dispositivos. En 2022, se predice un aumento de 258.2 billones de descargas.

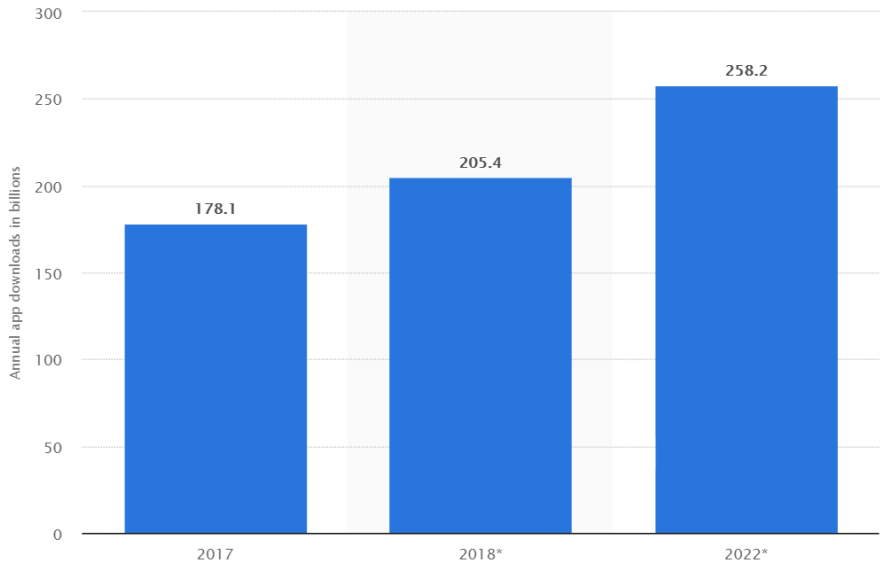


Figura 1: Número mundial de descargas de aplicaciones en 2017,2018 y 2022(en billones)

Se espera un aumento de los beneficios de hasta 139 billones de dólares anualmente. Además, China seguirá encabezará el mercado, seguida por Estados Unidos, Japón y Europa. En el siguiente vídeo de la *Figura 2* se muestra el crecimiento económico y geográfico del desarrollo de aplicaciones móviles para 2021.

Figura 2: Ctrl + Click para ver el vídeo

El lenguaje que hemos utilizado para programar *Move Your School* es ***Java***, un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por *Sun Microsystems*. Este lenguaje de programación es uno de los más utilizados en todo el mundo, como se puede observar en la *Figura 3*.

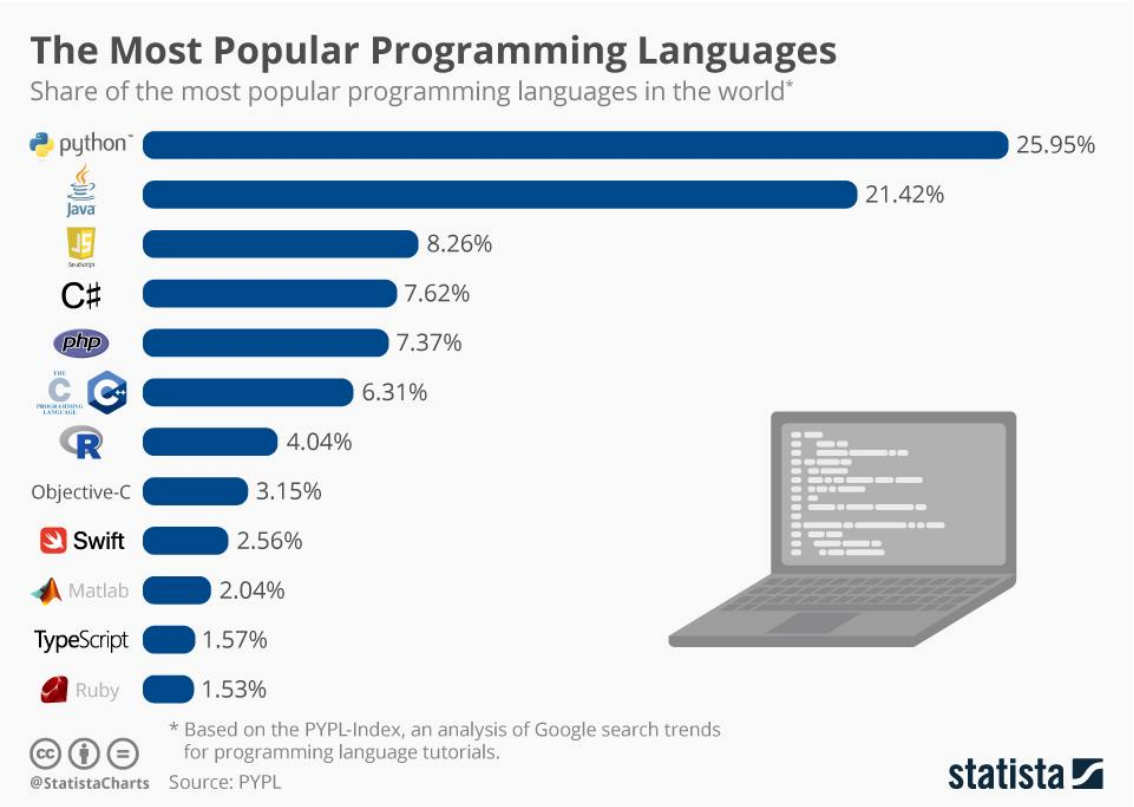


Figura 3

Principales competidores:

Eventbrite es una aplicación que sirve para organizar eventos y vender las entradas. También sirve de buscador de eventos.
Lo que diferencia a *Eventbrite* de *Move Your School* es que no está orientada para grupos ni para colegios o instituciones educativas.

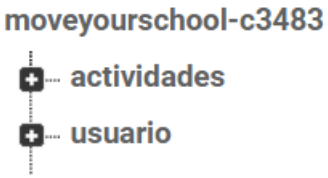
Es similar a la app anterior, *All Events in City* te muestra eventos en ciudades concretas.
No proporciona tampoco datos para actividades colegiales y está más centrado en la fiesta y la ciudad.

Fever tiene un diseño moderno e interactivo y la app permite seleccionar todos los *hashtags* que sean de interés. Es muy completa a la hora de detallar los eventos. Tiene muchas funcionalidades pero está más orientada a uso personal, no para actividades educativas en grupos.

2. DESARROLLO DEL PROYECTO

Una parte esencial del proyecto es la base de datos, la cual hemos desarrollado con tecnología *Firebase* de *Google*.

Se ha escogido un tipo de base de datos no relacional, que se basa en una estructura de nodos donde hay pares de clave y valor.
Un ejemplo de nuestra aplicación:
Podemos observar que se divide en actividades, usuarios... etc.

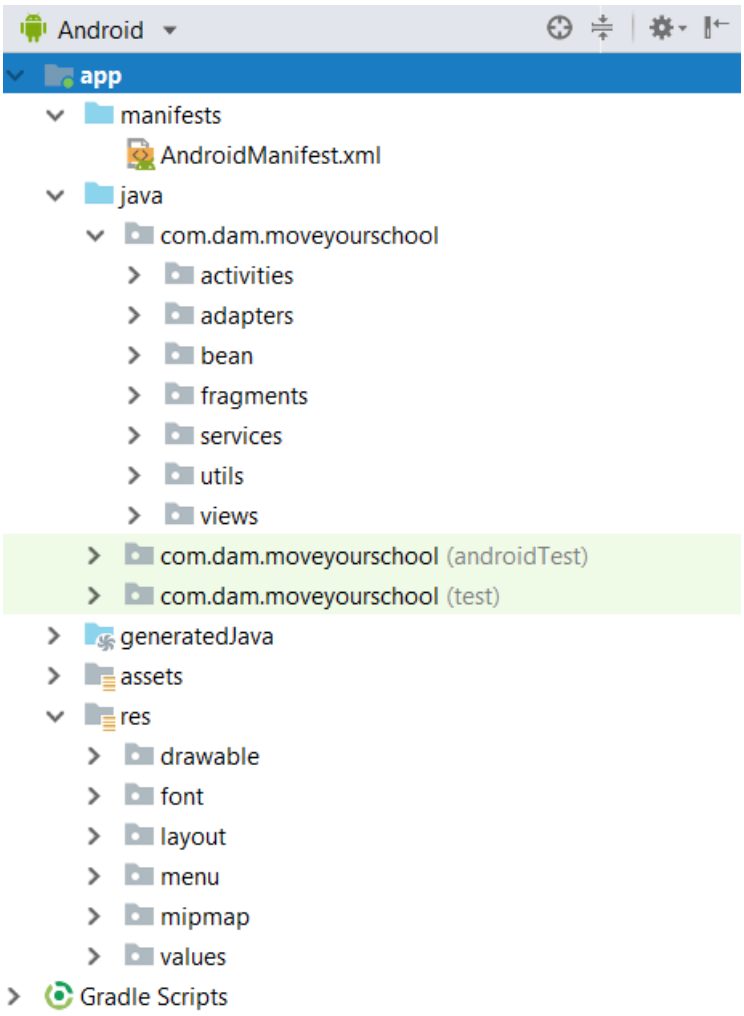


Las actividades tienen una clave y esta corresponde a la de un usuario, y el valor es un objeto actividad.

Respecto al *Backend*, se corresponde a una arquitectura de la aplicación *MVC*, se ha escogido una arquitectura con servicios desacoplados, utilizando clases abstractas para intentar separar la vista de la lógica de negocio.

```java
public abstract class FireDBUsuarios {
    private ChildEventListener cel;
    private DatabaseReference dbr;
    private static final String NODO_USUARIOS = "usuario";

    public FireDBUsuarios() {

        dbr = FirebaseDatabase.getInstance().getReference().child(NODO_USUARIOS);
        if (cel==null) {
            cel = new ChildEventListener() {
                @Override
                public void onChildAdded(@NonNull DataSnapshot dataSnapshot, @Nullable String s) {
                    nodoAgregado(dataSnapshot, s);
                }

                @Override
                public void onChildChanged(@NonNull DataSnapshot dataSnapshot, @Nullable String s)
                    nodoModificado(dataSnapshot, s);
                }
```

La arquitectura de los paquetes se divide en funcionalidades:



2.1 Herramientas tecnológicas

A continuación ofrecemos un resumen informativo sobre las tecnologías que utilizamos para nuestro proyecto:

Firebase:
La base de datos de *Move Your School* está desarrollada en *Firebase*.

Ctrl + Click para acceder a sitio web

Firebase es la plataforma de desarrollo móvil en la nube de *Google.* Fue comprado por *Google* en 2014 y luego la continuó mejorando con la compra del equipo de *Divshot.*

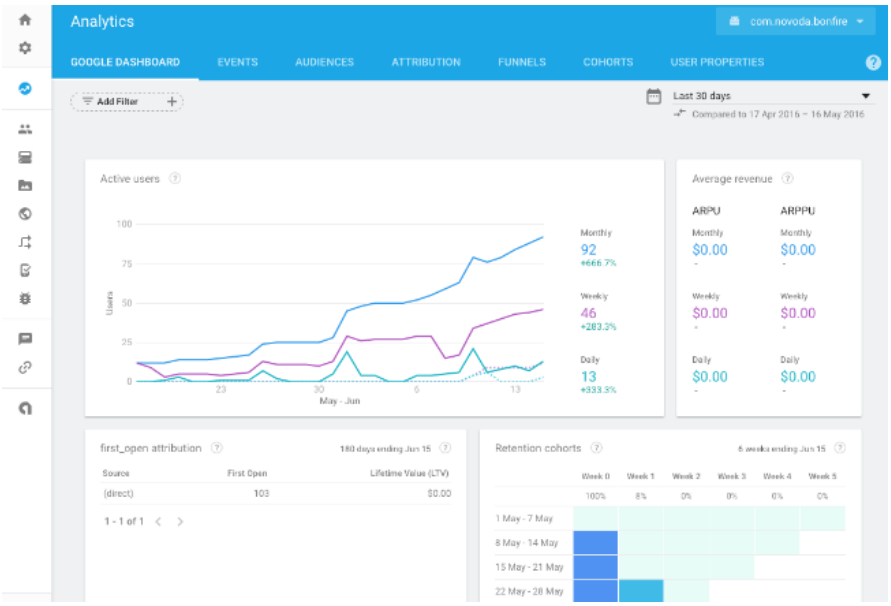
Firebase surgió para proveer una API para guardar y sincronizar datos en la nube en tiempo real.

Firebase se trata de una plataforma móvil creada por Google, cuya principal función es desarrollar y facilitar la creación de apps de elevada calidad de una forma rápida. La plataforma está subida en la nube y está disponible para diferentes plataformas como IOS, Android y web. Contiene diversas funciones para que cualquier desarrollador pueda combinar y adaptar la plataforma a medida de sus necesidades.

Principales características de *Firebase*:

- **Desarrollo**: *Firebase* permite la creación de mejores apps, minimizando el tiempo de optimización y desarrollo, mediante diferentes funciones, entre las que destacan la detección de errores y de testeo, que supone poder dar un salto de calidad a la app. Poder almacenar todo en la nube, testear la app o poder configurarla de manera remota, son características destacables de la plataforma.

- **Analítica**: Tener un control máximo del rendimiento de la app mediante métricas analíticas, todo desde un único panel y de forma gratuita, es una de las ventajas que ofrece *Firebase* respecto a la analítica web.



- **Poder de crecimiento**: Permite gestionar de manera fácil todos los usuarios de las aplicaciones, con el añadido de que se pueden captar nuevos usuarios, mediante invitaciones o notificaciones.

- **Monetización**: Mediante *AdMob*, *Firebase* permite que puedas ganar dinero.

- **Rapidez**: Implementar *Firebase* puede ser fácil y rápido, gracias a su API que es muy intuitiva, sostenida en un solo *SDK*. Con *Firebase* puedes centrar tus esfuerzos en resolver los problemas de tus clientes y así poder evitar la pérdida de tiempo en la creación de una infraestructura compleja.

- **Agilidad**: *Firebase* ofrece apps multiplataforma con unas *Apis* integradas a *SDK* individuales para *IOS*, *Android* y *Javascript*, de tal forma que se puede gestionar diferentes apps sin necesidad de salir de la propia plataforma.

Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en *IntelliJ IDEA.*

Ctrl + Click para acceder a sitio web

Android Studio ofrece muchas funciones que aumentan la productividad durante la compilación de apps para Android, como las siguientes:

- Un sistema de compilación basado en Gradle flexible
- Un emulador rápido con varias funciones
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos *Android*
- *Instant Run* para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo *APK*
- Integración de plantillas de código y *GitHub* para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código
- Gran cantidad de herramientas y frameworks de prueba
- Herramientas *Lint* para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con *C++* y *NDK*
- Soporte incorporado para *Google Cloud Platform*, lo que facilita la integración de *Google Cloud Messaging* y *App Engine*

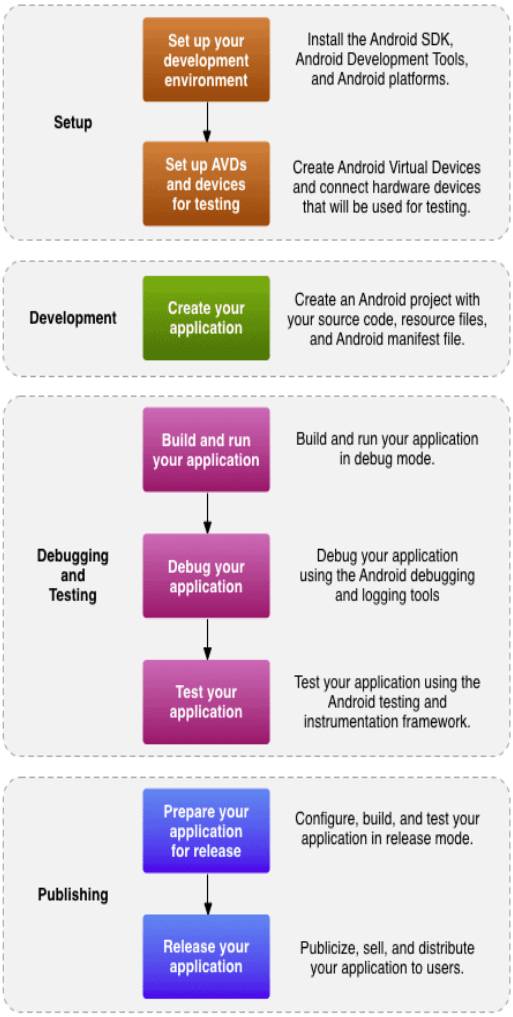
Entre las fases de desarrollo que abarcan la realización de aplicaciones en Android Studio encontramos cuatro etapas.

1. La primera es la **configuración de entorno**; durante esta fase se instala y configura el entorno de desarrollo. Además se realiza la conexión a los elementos en donde se pueden realizar la instalación de las app, y se crean dispositivos virtuales Android (AVDS).

2. La segunda fase abarca la **Configuración del Proyecto y Desarrollo**; durante esta se realiza la configuración del proyecto y el desarrollo del mismo. Hablamos de la creación de módulos que contengan recursos para la aplicación y archivos de código fuente.

3. La tercera fase comprende las **pruebas, depuración y construcción de la aplicación**; A esta altura se construye el proyecto en un paquete (s) depurable .apk que se puede instalar y ejecutar en el emulador o en un dispositivo con Android.

4. Ya como última fase se haya la **publicación de la aplicación**; en esta etapa se realiza la configuración y se arma la solicitud para el uso y libre distribución de la aplicación a los usuarios. Durante la etapa de preparación se construye una versión de la aplicación, que los usuarios pueden descargar e instalar en sus dispositivos de modo que se pueda vender y distribuir la versión de esta.

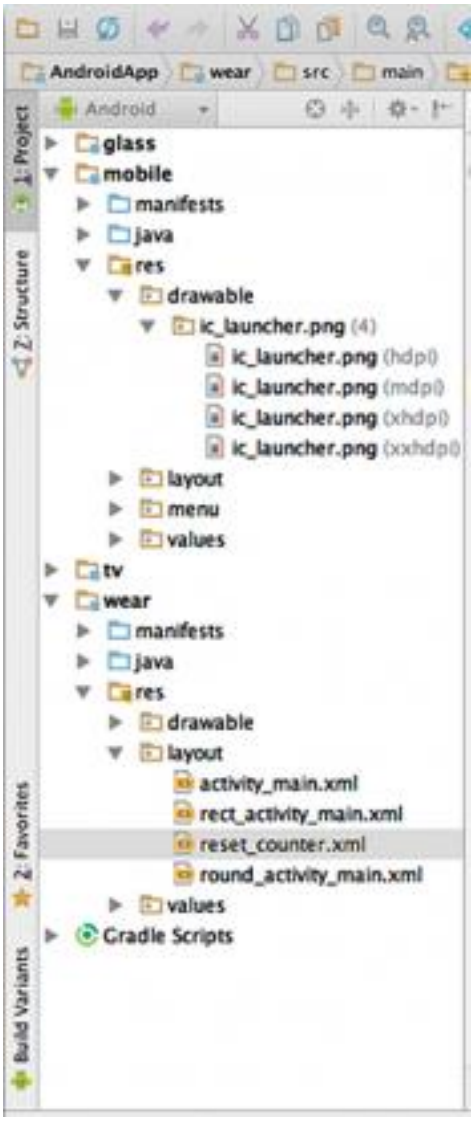


En esta imagen podemos apreciar el diagrama de las etapas para la realización de aplicaciones en Android Studio.

En el caso de cada proyecto, en referencia con la base modular, la aplicación contiene uno o más módulos con archivos de código fuente y archivos de recursos.

De forma predeterminada, Android Studio muestra los archivos del proyecto en la vista del proyecto Android. En este punto se aprecia de forma organizada los módulos para proporcionar un acceso rápido a los archivos de código fuente clave.

En *Android Studio* se utiliza *Gradle* como la base del sistema de construcción de aplicaciones. Este sistema de creación, se ejecuta como una herramienta integrada en el menú *Android Studio*, y a su vez es independiente de la línea de comandos.



Organización de los módulos en Android Studio

En resumen *Android Studio* es entorno dedicado en exclusiva a la programación de aplicaciones para dispositivos Android, proporcionando a *Google* un mayor control sobre el proceso de producción.

Saas, Paas e Iaas

MoveYourSchool ofrece una aplicación de funcionalidades básicas para poder consultar, desde el lado del administrador, el estado de toda la información de la aplicación móvil.

Esta aplicación es *MoveYourSchool Administrator*, creada mediante la plataforma Heroku. Esta plataforma es un servicio (**PaaS**) para desarrollar aplicaciones en la nube. Estos ofrecen entornos abstractos en los que los desarrolladores pueden simplemente dejar su código y dejar que la plataforma se ocupe de los detalles de aprovisionamiento.

Cuando nos referimos a **desarrollar aplicaciones en la nube** tenemos que puntualizar de qué manera lo vamos a hacer, ya que dentro del concepto nube existen distintas formas de hacerlo que nos permiten una mayor flexibilidad o sencillez a la hora de desplegar nuestras aplicaciones o mantenerlas. Entre estas distintas formas que puede adoptar la nube se encuentran: **Software-as-a-Service** (**SaaS**), **Platform-as-a-Service (PaaS)** y **Infraestructure-as-a-Service (IaaS)**.

Software-as-a-Service (SaaS)

Básicamente se trata de cualquier servicio basado en la web. Tenemos ejemplos claros como el *Webmail* de *Gmail*, los *CRMonlines*. En este tipo de servicios nosotros accedemos normalmente a través del navegador sin atender al software. Todo el desarrollo, mantenimiento, actualizaciones, copias de seguridad es responsabilidad del proveedor.

En este caso tenemos poco control, nosotros nos situamos en la parte más arriba de la capa del servicio. Si el servicio se cae es responsabilidad de proveedor hacer que vuelva a funcionar.

Platform-as-a-Service (PaaS)

En este caso nuestra única preocupación es la construcción de nuestra aplicación, ya que la infraestructura nos la da la plataforma.

Es un modelo que reduce bastante la complejidad a la hora de desplegar y mantener aplicaciones ya que las soluciones PaaS gestionan automáticamente la escalabilidad usando más recursos si fuera necesario. Los desarrolladores aun así tienen que preocuparse de que sus aplicaciones estén lo mejor optimizadas posibles para consumir menos recursos posibles (número de peticiones, escrituras en disco, espacio requerido, tiempo de proceso, etc...) Pero todo ello sin entrar al nivel de máquinas.

Para los desarrolladores que ignoran la infraestructura que deben montar y sólo quieren preocuparse de escribir software, esta es la alternativa a seguir.

Heroku es un ejemplo de este modelo.

Infraestructure-as-a-Service (IaaS)

En este caso con **IaaS** se tiene mucho más control que con PaaS, aunque a cambio hay que de encargarse de la gestión de infraestructura.

El ejemplo perfecto es el proporcionado por ***Amazon Web Service (AWS)***. Se puede elegir qué tipo de instancias queremos usar LInux o Windows, así como la capacidad de memoria o procesador de cada una de nuestras maquinas. El hardware resulta transparente, todo lo que se maneja es de forma virtual.

La principal diferencia es que hay que encargarse de escalar nuestras aplicaciones según las necesidades, además de preparar todo el entorno en las máquinas.

Además de AWS nos encontramos ejemplos como Rackspace Cloud o vCloud de VMWare.

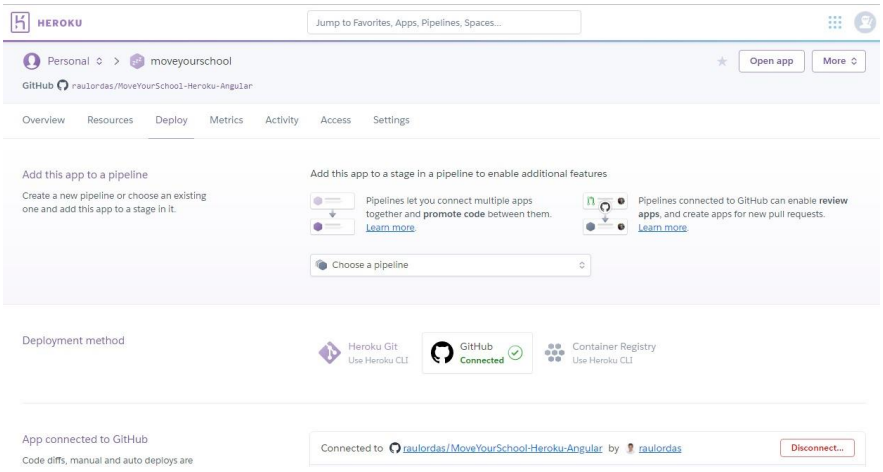
Heroku

Utiliza Git (un sistema de control de versiones distribuido para la administración de códigos) para administrar implementaciones de aplicaciones.

Algunas características de Heroku son:

- Ejecuta su aplicación a través de un número predeterminado de servidores virtuales.
- Gestiona los lanzamientos desplegando su aplicación a diferentes entornos.
- Asegura que su aplicación se recupere automáticamente de las fallas del servidor.
- Maneja el equilibrio de carga en muchas instancias de aplicaciones, lo que le permite escalar instantáneamente su aplicación para que sea compatible con millones de usuarios.
- Le permite agregar y eliminar rápidamente bloques de infraestructura, como servidores de almacenamiento en caché y servidores de bases de datos.

Heroku es compatible con los lenguajes de programación Ruby, Node.js, Python, Java, Go, PHP y Scala. Esto proporciona una implementación fácil de las tecnologías existentes en Heroku con las modificaciones mínimas necesarias.

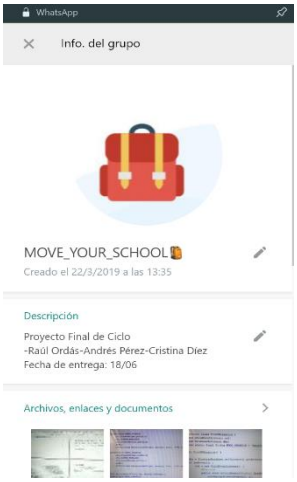


2.2 Planificación

Grupo de whatssap:

Mediante *Whatssap* enviamos apuntes sencillos, archivos, Fotos y vídeos explicatorios para los demás integrantes del grupo.

Será la vía de comunicación más directa pero con menos carácter formal.

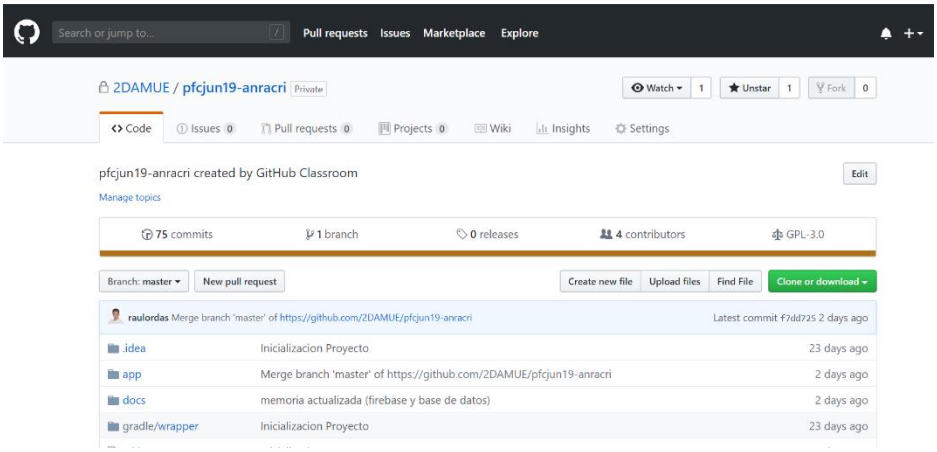


Slack:

Slack es la plataforma mediante la que nos comunicamos con nuestro tutor, Ernesto, para detallar los avances del proyecto y ultimar las decisiones del desarrollo.

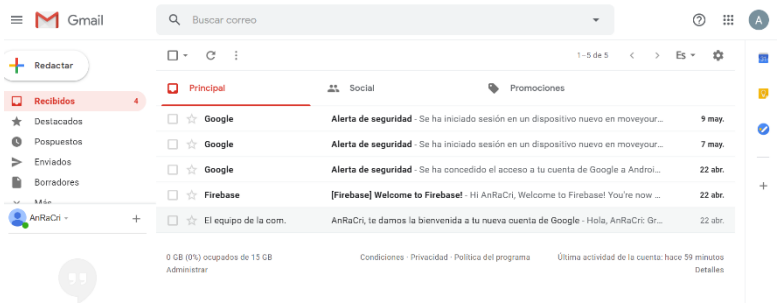
GitHub

La plataforma mediante la cual compartimos el código es *GitHub.*



Gmail

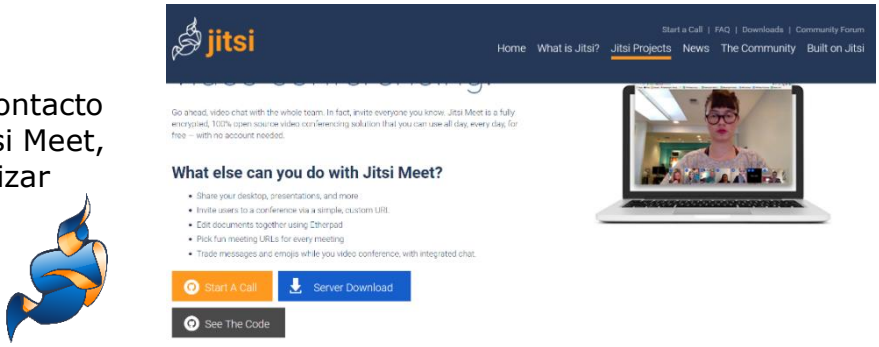
Hemos creado una cuenta de *Gmail* para administrar las herramientas de *Google* y que los usuarios puedan contactar con nosotros mediante éste.



AnRaCri School
moveyourschoolapp@gmail.com

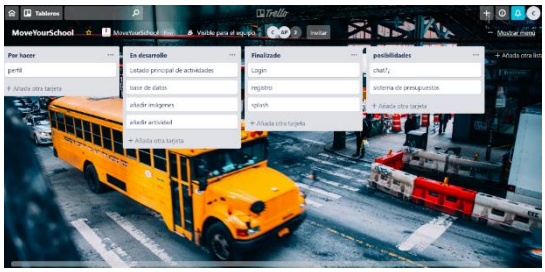
Jitsi Meet

El equipo se pondrá en contacto con el tutor mediante Jitsi Meet, una plataforma para realizar videoconferencias online.



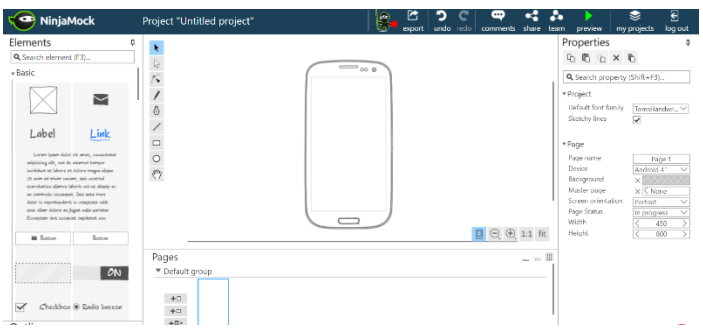
Trello

Trello es una herramienta de tecnología *Kanban* online. Con esta aplicación podemos definir las tareas realizadas, en curso y sin empezar.
Podemos añadir y eliminar tareas y plantear posibilidades. Cada usuario podrá administrar sus tareas e indicar así a los demás si ha conseguido finalizarlas o no.



NinjaMock

Es la herraienta que hemos elegido Para realizar el mockup inicial del Proyecto.



Integrantes y rol en el proyecto:

Cloud Master: Administrar y actualizar los sitios web del equipo y mantener el correcto funcionamiento de las aplicaciones usadas para el desarrollo del proyecto.

Design Master: Cuida la estética de la aplicación, decide las fuentes, elige la paleta de colores… Administra todo el aspecto visual.

Scrum Master: Dirige el grupo y revisa que se cumplan los plazos. También se encarga de mantener el contacto entre los integrantes del equipo para seguir el desarrollo del proyecto y comprobar los cambios realizados.

Temporalización:

El proyecto tendrá 3 fases: Fase inicial en la que se desarrollará la idea, se realizarán esquemas y bocetos de la aplicación. Fase intermedia de desarrollo del proyecto. Y fase final en la cual se defenderá el proyecto ante un tribunal de la Universidad y se subirá al *Play Store* la app.

Cada semana el equipo se pondrá en contacto para explicar y avanzar el proyecto. En cualquier momento cada uno de los integrantes del grupo podrá contactar con los otros mediante *Whatssap* para cualquier duda o aclaración.

Una vez cada dos semanas, el equipo se pondrá en contacto con el tutor mediante *Jitsi Meet* para mostrar el desarrollo del proyecto. La vida de desarrollo del proyecto es entre el 1 de abril de 2019 hasta el 18 de junio de 2019, cuando se realizará la defensa del mismo.

2.3 Descripción del trabajo realizado

Antes de comenzar con la realización del proyecto, realizamos un diagrama de casos de uso (*Figura 5*) para plantear los servicios de la aplicación e identificar los tipos de usuarios que manejarán la aplicación:

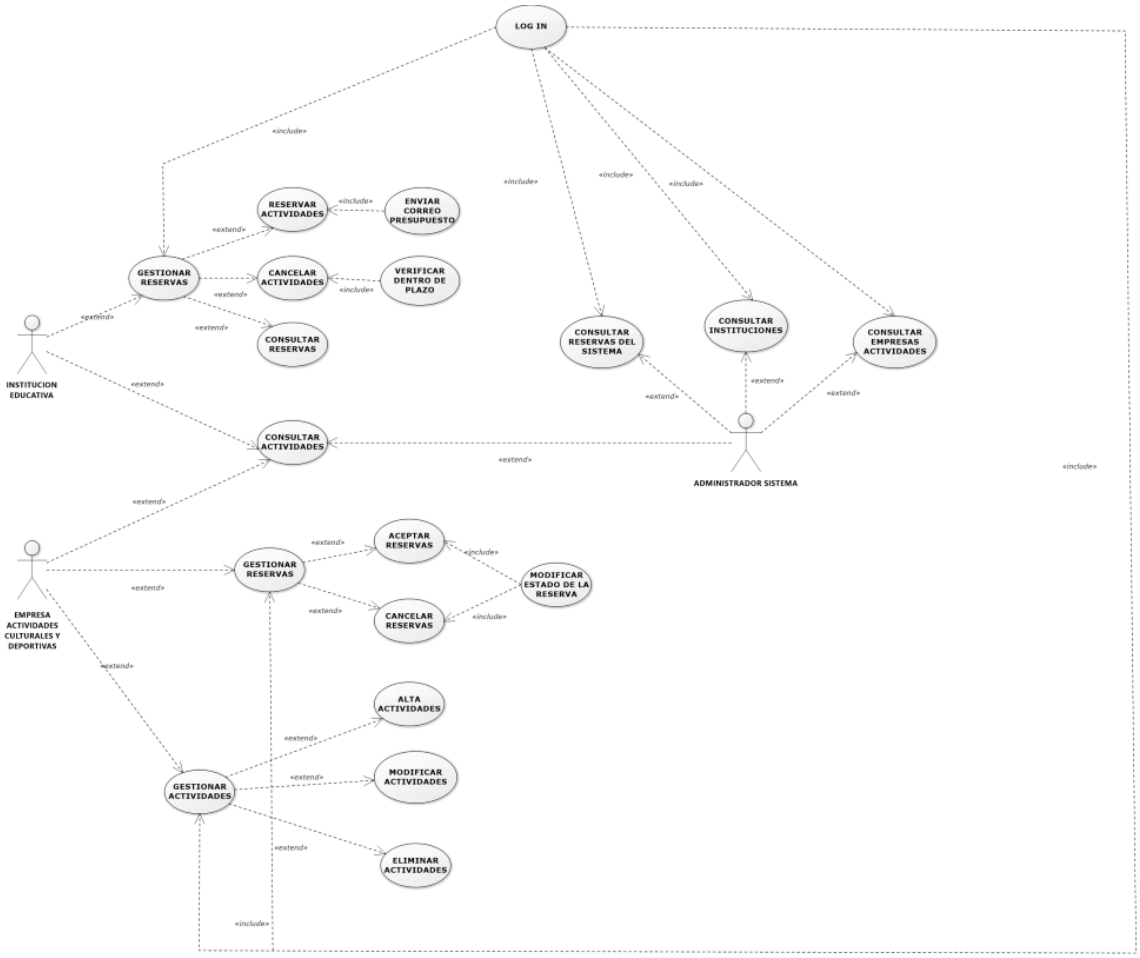
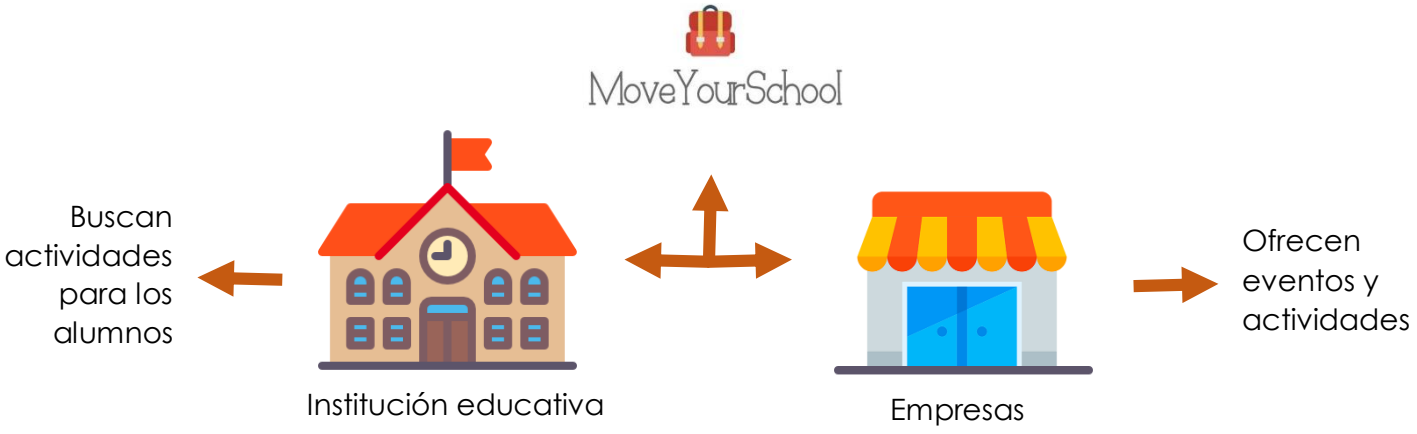


Figura 5

Mediante este proceso obtuvimos la idea de realizar un sistema de usuarios tipo empresa y otros usuarios tipo institución educativa; dependiendo de sus necesidades principales.



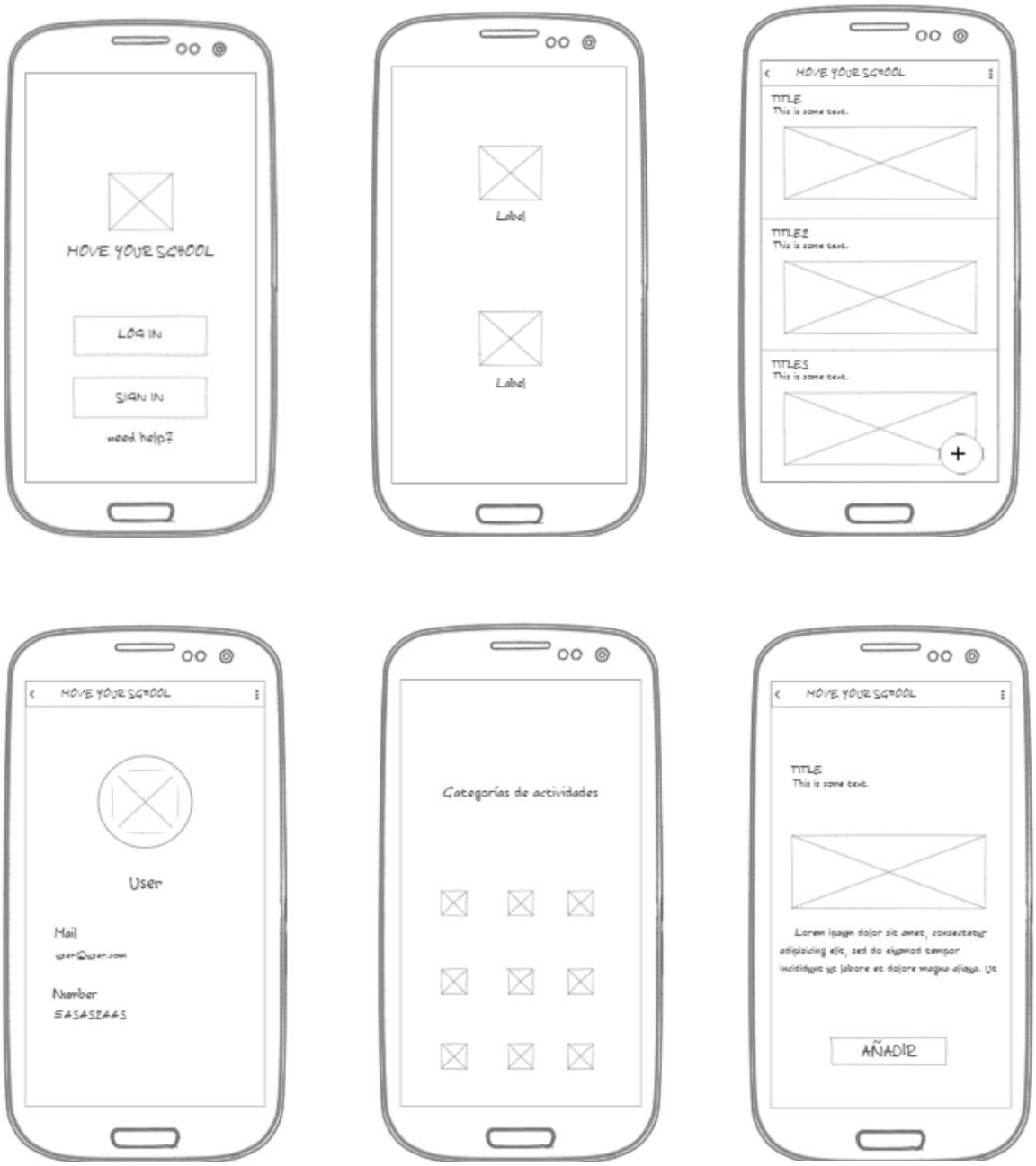


Figura 4

En la *Figura 4* está representado el *Mockup* de la app inicialmente. Este "boceto" del funcionamiento de la aplicación muestra como debe ser la estructura del proyecto. En las dos primeras maquetas se retrata el registro de un nuevo usuario y acceso con una cuenta ya existente.
Las siguientes representaciones corresponden con el contenido de la aplicación, como la zona del usuario, el listado de actividades, los filtros posibles y la opción de añadir un evento.
Este *Mockup* es solo una referencia para el desarrollo del proyecto. Por lo que el resultado final del proyecto podría variar.

Realizamos un esquema conceptual de cómo debemos montar la base de datos:

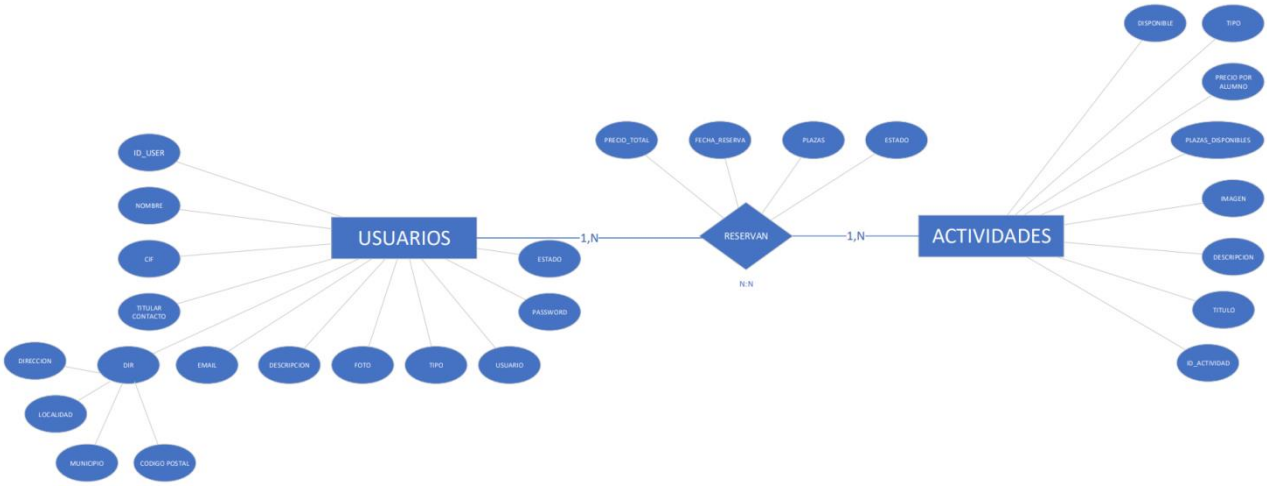


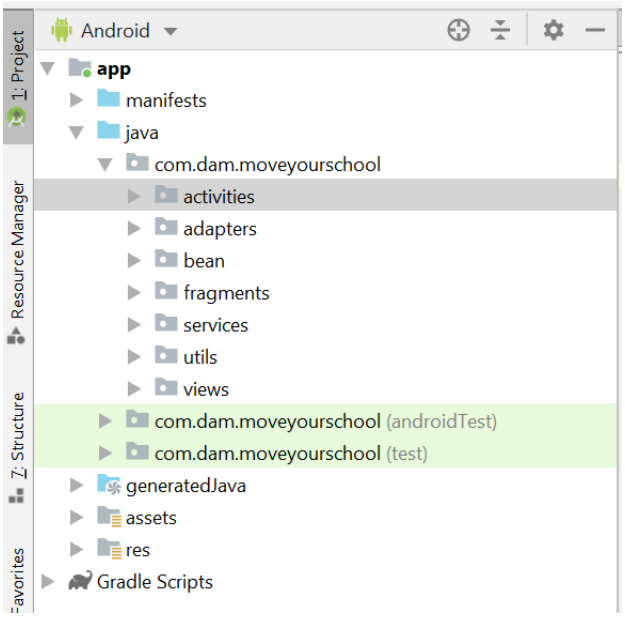
Figura 6

En el esquema de la Figura 6 se pueden observar las relaciones y cardinalidad existentes entre las tres entidades que forman parte de la persistencia y los atributos de los usuarios, de las reservas y de las actividades.

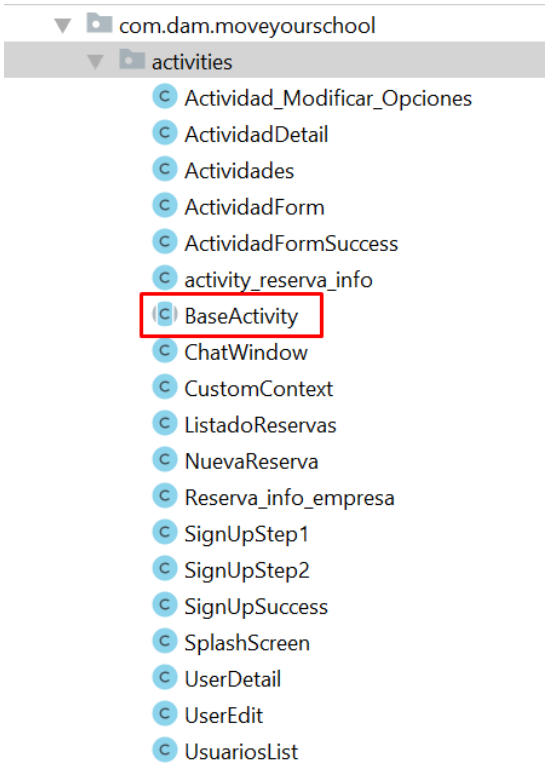
Detalles del código

A continuación presentamos un análisis básico de algunas de las partes del código de MoveYourSchool y su arquitectura.

La estructura del proyecto en *Android Studio* está organizada por carpetas. En el **paquete de java** encontramos estas carpetas relacionadas con las clases y las *activities* del proyecto:



- En la carpeta *activities* podemos encontrar las clases que representan a cada pantalla con la que el usuario puede interactuar y contiene las acciones, métodos y referencias para que esta funcione.

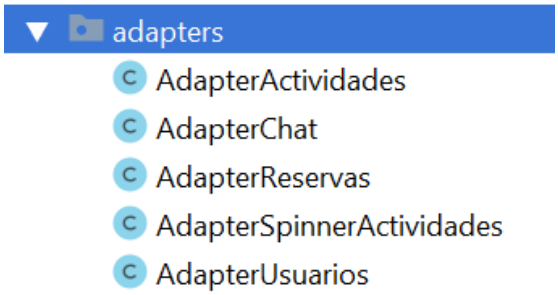


Cabe destacar el *activity* ***Base Activity*** ***(Anexo 2*)*** del cual extenderán la mayoría del resto, como por ejemplo:

```
public class Actividades extends BaseActivity {

public class ListadoReservas extends BaseActivity {

public class Reserva_info_empresa extends BaseActivity {
```

Listado 1: Ejemplos de clases que extienden de BaseActivity

- La carpeta de *adapters* contiene todas las clases *adaptador* para usar *recyclers* y *spinners*.



```
public class AdapterActividades extends
RecyclerView.Adapter<AdapterActividades.HolderActividades> implements
View.OnClickListener {
```

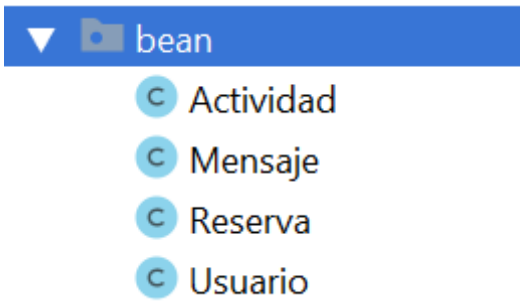
Listado 2: Ejemplo Adapter

Llevan las clases internas "*onCreateViewHolder*" y "*onBindViewHolder*"

```
@NonNull
@Override
public HolderActividades onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
    View view =
LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.item_actividad,
viewGroup, false);
    view.setOnClickListener(this);
    HolderActividades holderActividades = new HolderActividades(view);
    return holderActividades;
}
----------------------------------------------------------------------------------

@Override
public void onBindViewHolder(@NonNull HolderActividades holderActividades, int i)
{
    if (listaActividades.get(i).getUrlFoto() != null &&
!listaActividades.get(i).getUrlFoto().equals("")) {

glide.load(listaActividades.get(i).getUrlFoto()).into(holderActividades.imagen);
    } else {
        glide.load(R.drawable.democactivity).into(holderActividades.imagen);
    }
```

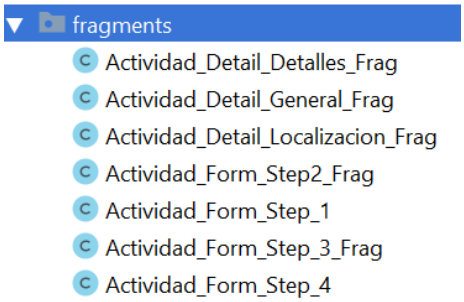
Listado 3: Ejemplo onCreateViewHolder y onBindViewHolder

- En el paquete *bean* se crean las clases *objetos*, o *POJO*:



Estas clases representan a los objetos que vamos a tratar y sus atributos. Se componen de los atributos, el constructor , getters, setters y el método toString.

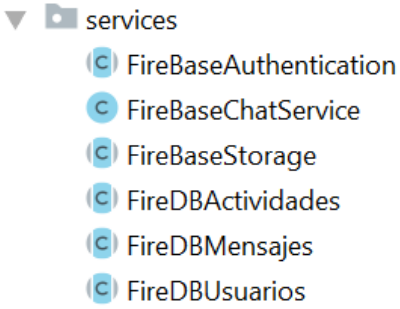
- En la carpeta *Fragments* guardamos los *fragmentos* de la aplicación, los cuales representan un comportamiento de la interfaz de usuario en una Activity. Extienden de Fragment.




```
public class Actividad_Detail_Localizacion_Frag extends Fragment {
```

Listado 4: Fragment

- *Services* se refiere a todas las clases-conexión con la base de datos de *FireBase*.

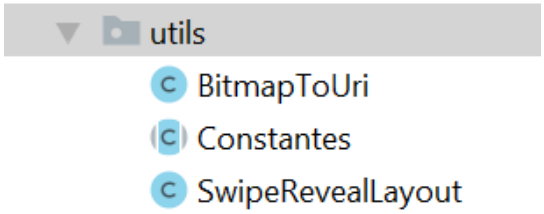


Hacen referencia a los a la base de datos de *Firebase*:

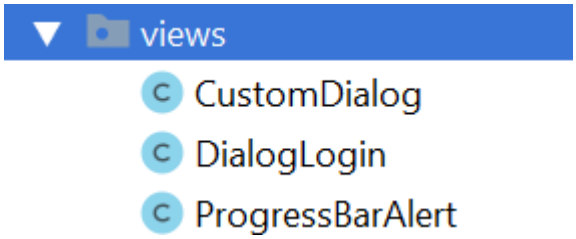
```
public abstract class FireDBActividades {
    private ChildEventListener cel;
    private DatabaseReference dbr;
    private ChildEventListener auxListener;
    private ValueEventListener valListener;
    private static final String NODO_ACTIVIDADES = "actividades";

    public FireDBActividades() {

        dbr =
FirebaseDatabase.getInstance().getReference().child(NODO_ACTIVIDADES);
```

Listado 5: Ejemplo de service FireBase

- El paquete *utils* sirve para almacenar clases auxiliares. Estas clases son las que se muestran en la siguiente imagen:



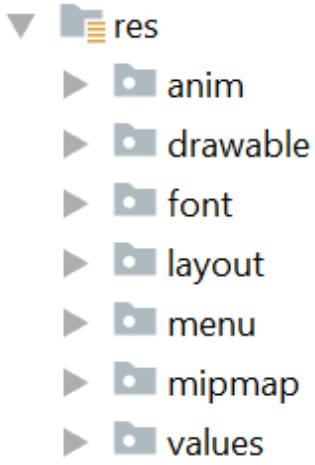
- El paquete *view* guarda las clases de los cuadros de diálogo y las alertas:



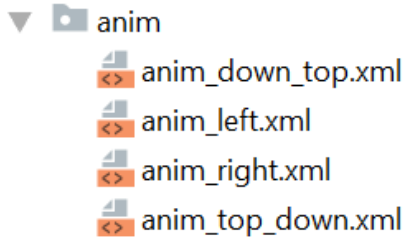

```
public class ProgressBarAlert extends Dialog {

    public ProgressBarAlert(Context context) {
        super(context,  R.style.Theme_AppCompat_Dialog);
        setContentView(R.layout.progressbar);
    }
}
```

Listado 6: Clase ProgressBarAlert

Respecto a la **carpeta res** contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, layouts, cadenas de texto, etc.



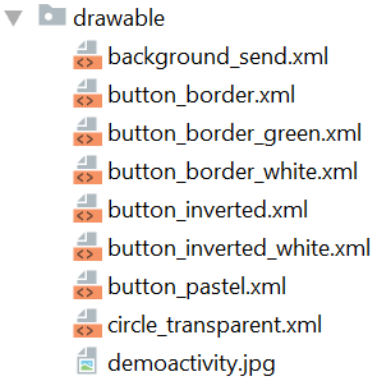
- La carpeta *anim* contiene los *xml* que definen de las animaciones utilizadas por la aplicación.




```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android" >
    <translate
        android:duration="500"
        android:fromXDelta="100%"
        android:interpolator="@android:interpolator/accelerate_decelerate"
        android:toXDelta="0%" >
    </translate>
</set>
```

Listado 7: anim_left.xml (animación izquierda)

- La carpeta *drawable* contiene las imágenes y otros elementos gráficos usados por la aplicación:

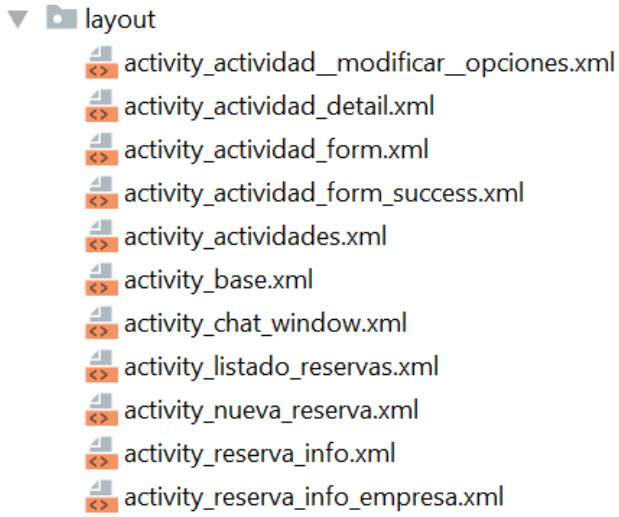



```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle"
    android:tint="@color/blue_pastel">
    <corners
        android:bottomLeftRadius="20dp"
        android:bottomRightRadius="20dp"
        android:topLeftRadius="20dp"
        android:topRightRadius="20dp">
    </corners>
</shape>
```

Listado 8: Estilo botón color blue_pastel

- En *Font* se guardan las distintas fuentes que se usan en la aplicación:

- El fichero *layout* contiene los ficheros de definición XML de las diferentes pantallas de la interfaz gráfica.

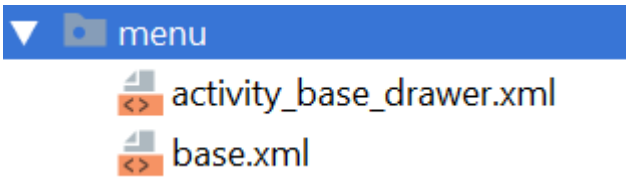


Los ficheros archivos xml "activity" referencian a una clase Activity.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.ActividadDetail">
```

Listado 9: Inicio de activity_actividad_detail.xml, contexto ActividadDetail

- La carpeta *menu* contiene la definición XML de los menús de la aplicación.

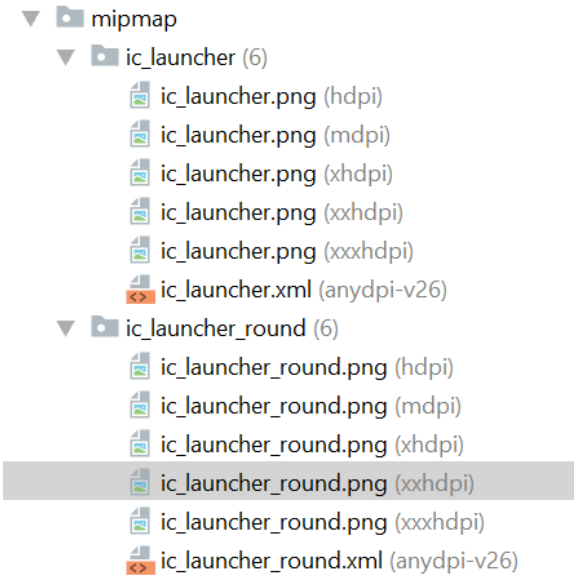


Dentro de estos xml se pueden distinguir distintos ítems que corresponden a iconos de drawable:

```
<item
    android:id="@+id/nav_MisReservas"
    android:icon="@drawable/ic_cards"
    android:title="Mis Reservas" />
```

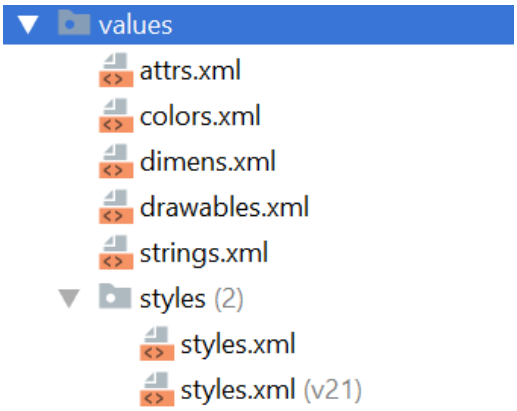
Listado 10: ítem de activity_base_drawer

- *Mipmap* contiene los iconos de lanzamiento de la aplicación (el icono que aparecerá en el menú de aplicaciones del dispositivo) para las distintas densidades de pantalla existentes.



Ic_launcher_round.png

- La carpeta *values* contiene otros ficheros XML de recursos de la aplicación, como por ejemplo cadenas de texto (*strings.xml*), estilos (*styles.xml*), colores (*colors.xml*), tamaños (*dimens.xml*), etc.




```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="SwipeRevealLayout">
        <attr name="dragFromEdge">
            <flag name="left" value="1" />
            <flag name="right" value="2" />
        </attr>
    </declare-styleable>
</resources>
```

Listado 11: attrs.xml

```
<color name="blue_pastel">#AEC6CF</color>
```

Listado 12: Ejemplo "azul pastel" de colors.xml

```
<dimen name="activity_vertical_margin">16dp</dimen>
```

Listado 13: Ejemplo dimesion 16dp's de dimens.xml

```
<item name="ic_menu_gallery"
type="drawable">@android:drawable/ic_menu_gallery</item>
```

Listado 14: Ejemplo de ítem de drawables.xml

```
<string name="app_name">MoveYourSchool</string>
```

Listado 15: Ejemplo de string de strings.xml

```
<style name="DialogTheme" parent="Theme.AppCompat.Light.Dialog">
    <item name="colorAccent">@color/rojo_icons</item>
</style>
```

Listado 16: Ejemplo de estilo en styles.xml

Move Your School Administrator

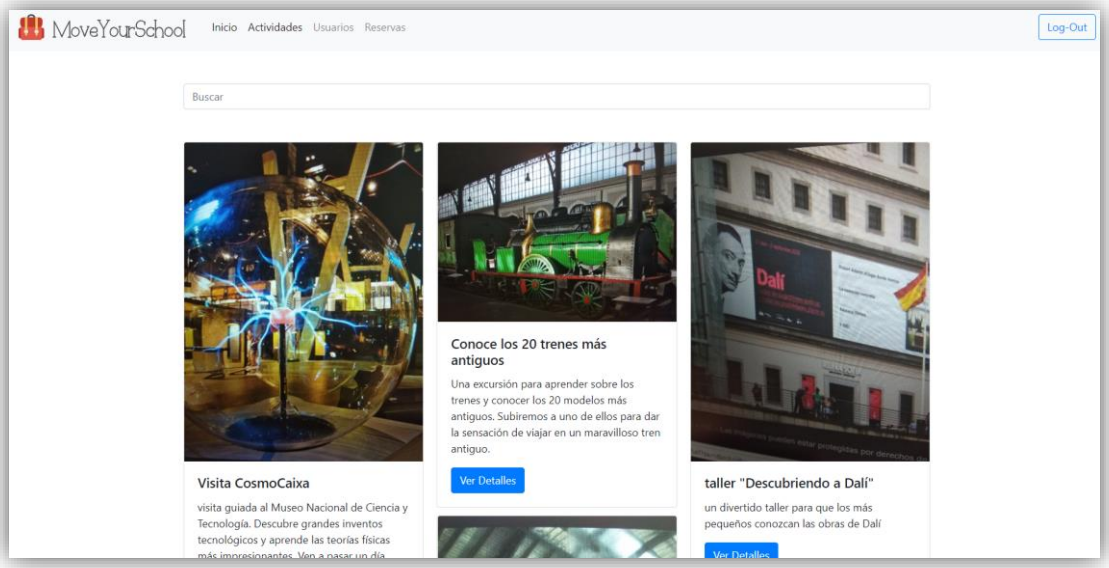
MYS Administrator es un desarrollo adicional a la aplicación Android de Move Your school.

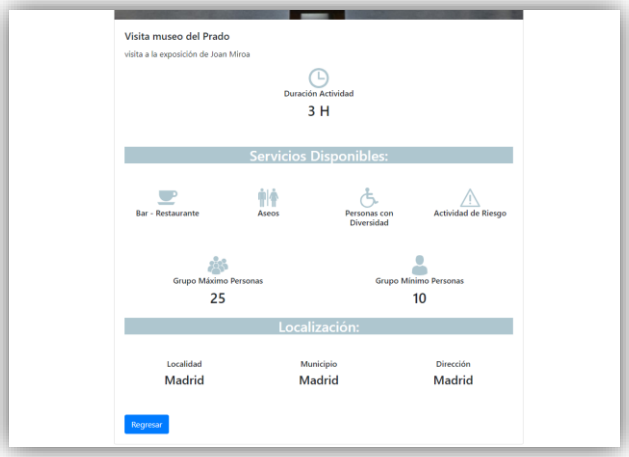
Trata de una aplicación paralela desarrollada en *Angular* con funcionalidad básica diseñado para gestionar y administrar los datos de la aplicación móvil.

Al ser un componente complementario, obtiene los datos de la misma base de datos de *Firebase*. En cambio la plataforma para desplegar la aplicación ha sido desde los servidores de Heroku.

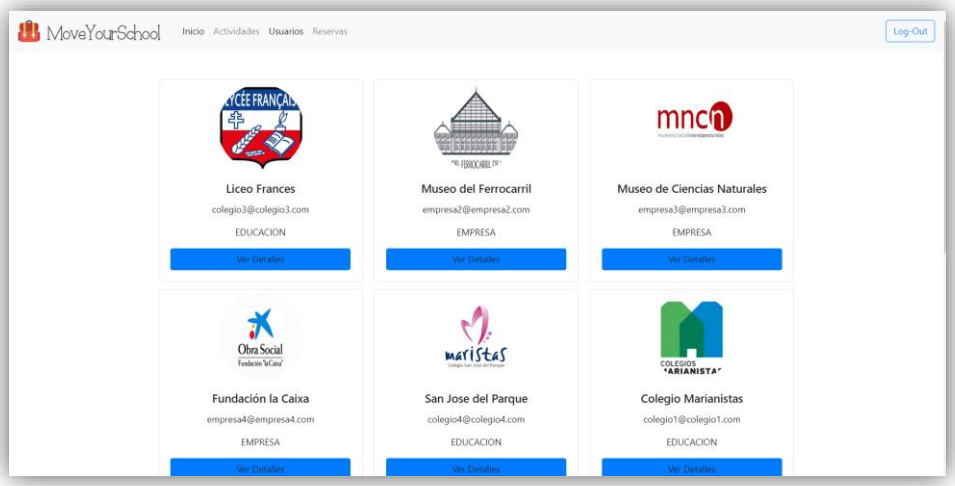
Las funcionalidades son más reducidas que las de la aplicación móvil. Tiene tres pestañas principales:

- *Actividades:* Muestra las actividades disponibles en la aplicación. Se pueden consultar los detalles y servicios ofrecidos en cada excursión pulsando sobre el botón "*Ver detalles*" de las distintas actividades.





- *Usuarios*: Muestra los usuarios registrados en *Move Your School*. Se pueden observar los detalles de los usuarios pulsado en "*Ver detalles*".



- *Reservas*: En esta pestaña se pueden observar las reservas de actividades con sus códigos identificadores.

Los detalles del código de Move Your School Administrator están especificados en el Anexo 1.*

2.4 Resultados y validación

Move Your School no presentó problemas iniciales de funcionamiento o errores bloqueantes. El desarrollo y proceso de elaboración de la aplicación ha sido fluido y acorde con los sprints acordados por el equipo. Además han demostrado eficacia ante los problemas de diseño o de código planteados, resolviéndolos casi inmediatamente.

Aun así, a lo largo del desarrollo de Move Your School hemos afrontado pequeñas dudas y complicaciones, las cuales se proceden a detallar a continuación:

Ilustración 1

Al inicio de la aplicación aparece esta pantalla, *Ilustración 1*, en la que las nubes y el autobús están animados con sensación de movimiento. Pulsando en el botón *"SING IN"* podremos ingresar nuestras credenciales para iniciar la aplicación con nuestro usuario, clicando en *"COMENZAR"* tenemos la opción de abrir la aplicación con funciones básicas y en el enlace *"¿No tienes cuenta? Date de alta"* saltaremos a la pantalla de la *Ilustración 2*.

La complicación principal de esta *activity* es que, al ser la principal, tiene que ser llamativa y concordar con la temática de la aplicación. Todo esto lo hemos conseguido añadiendo una animación de un autobús escolar y una colocación adecuada y limpia de los distintos componentes como botones y logotipos.

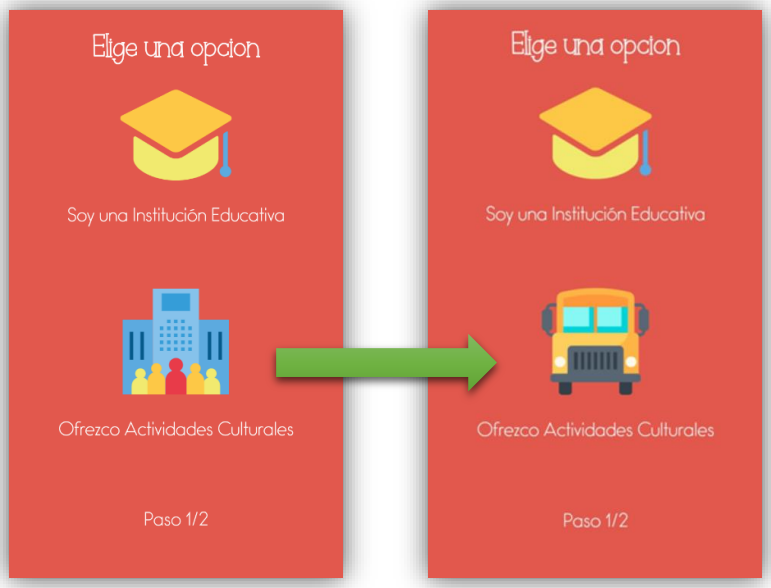


Ilustración 2

La *Ilustración 2* es la pantalla de elección de tipo de perfil. Surgió un problema estético con el logotipo de las empresas ya que parecía poco desenfadado para el estilo de la aplicación. Decidimos cambiarlo por la imagen de un autobús más animado y acorde con la aplicación. Dependiendo de la opción elegida, aparecerá una pantalla de registro con los datos correspondientes al tipo de perfil.

Ilustración 3

La *Ilustración 3* representa la pantalla principal de la aplicación. En ella aparece un listado con las actividades disponibles. Se puede acceder al menú desde la parte superior izquierda y a los filtros en la pestaña de *"ORDENAR" y "FILTRAR POR CATEGORÍA".* Dichos filtros fueron el reto de esta pantalla, pero no resultó ningún problema para el correcto desarrollo del proyecto.

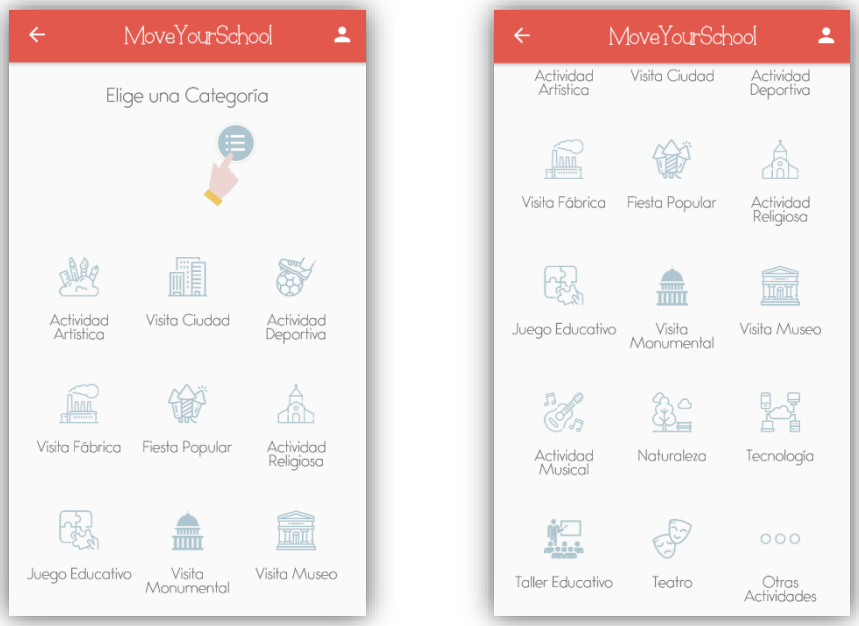


Ilustración 4

Cuando se cree una nueva actividad, se abrirá el *activity* *Ilustración 4*, en el cual aparecen todas las categorías para las excursiones. En la parte superior aparece una animación de una mano pulsando una opción indicando al usuario que debe elegir entre las distintas categorías. Decidimos poner una opción final de "Otras actividades" para cubrir todas las opciones posibles. Estos filtros dan mucha flexibilidad a *Move Your School* a la hora de buscar excursiones concretas.

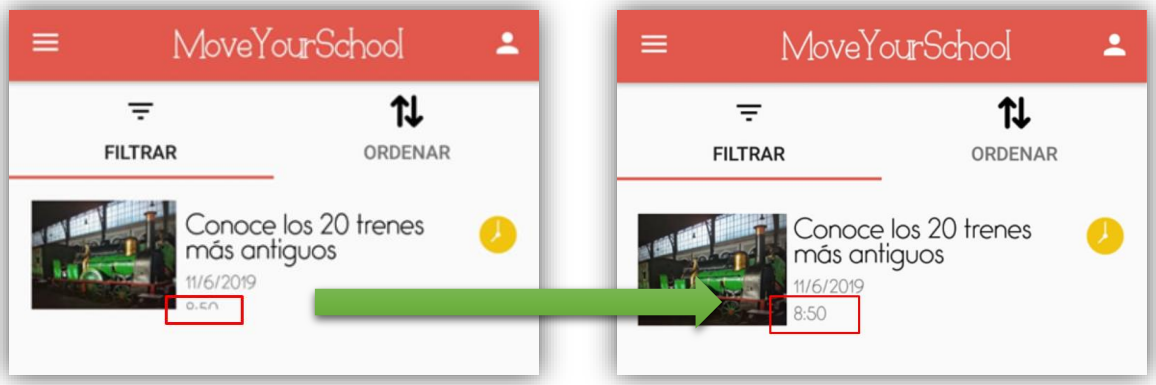


Ilustración 5

En la *Ilustración 5* se puede observar un error que tuvimos con uno de los adapters. En este caso se trata del adapter del listado de las reservas. Como se puede observar, la información se cortaba. Aumentamos el valor del alto de los elementos del adapter, solucionando así el problema.

3 CONCLUSIONES

Move Your School tenía como principal objetivo implementar una interfaz sencilla y llamativa y promover la variedad de excursiones.
Estos objetivos han sido cumplidos con creces, ya que hemos obtenido un magnífico trabajo gracias a todo el equipo.

Respecto al aspecto visual, gracias al diseño del Design Master, hemos conseguido desarrollar una interfaz con colores llamativos y alegres, acordes con la temática de nuestra aplicación (el aprendizaje, las excursiones, niños…), y a su vez sencilla de usar e intuitiva.

Nuestros esfuerzos también se han centrado en ofrecer una gran cantidad de filtros y de variedad a la hora de ofrecer una excursión. Éstas pueden ser encontradas mediante filtros específicos de la aplicación y, a la hora de subir una nueva actividad, se puede proporcionar información como servicios que se ofrecen, fotos, descripciones…

Todo este conjunto de objetivos aporta a la aplicación una gran funcionalidad y comodidad, cumpliendo así nuestras principales inquietudes.

El tiempo de desarrollo ha sido el correcto, por lo que no ha habido ningún atraso en el proyecto. Los miembros del equipo han cumplido correctamente con los "sprints" acordados.

Los principales problemas que se nos han presentado han sido:

- La complejidad de crear un chat en vivo para que los usuarios puedan comunicarse.
- El manejo de la base de datos para guardar imágenes y aplicarlas para las excursiones.
- El cierre de la página "Waffle" ha provocado que tengamos que usar una alternativa, la web de "Trello". Al final, no ha resultado ningún inconveniente ya que tienen una funcionalidad similar.
- El uso de colores llamativos sin resultar sobre-cargado.
- Insertar animaciones para dar un aspecto más juvenil y dinámico a la aplicación.

Hemos seguido la idea principal del mockup(Figura 4) agregando funcionalidades y "activities" conformando la aplicación final. Hemos conservado el recycler view para mostrar las actividades disponibles y el sistema de filtrado. El mayor cambio se ha realizado en la pantalla de registro o acceso a la aplicación, ya que hemos cambiado la disposición de los botones.

En conclusión, el proyecto se ha desarrollado correctamente, cumpliendo las expectativas esperadas y sin retrasos.

3.1 Innovación

Move Your School es una aplicación innovadora ya que plantea un nuevo formato de publicación de eventos. Move Your School no es el encargado de gestionar las actividades, son los propios usuarios los que pueden editar la información de sus excursiones, dando así mayor libertad de cambios en los precios y datos.
También es una aplicación pionera en comunicar empresas con colegios de manera sencilla y permitiendo la comunicación entre ambos.

3.2 Trabajo futuro

Move Your School es un proyecto con carácter real, es decir, que para el futuro podría ser potenciado como herramienta útil para colegios y empresas reales. Si así fuera, sería recomendable pagar el coste de una base de datos más potente de Firebase, en caso de que fuera necesario.

Move Your School Administrator es un proyecto que podría ser mejorado en un futuro, ofreciendo más funcionalidades y mayor control sobre las excursiones y actividades que se ofrezcan en Move Your School.

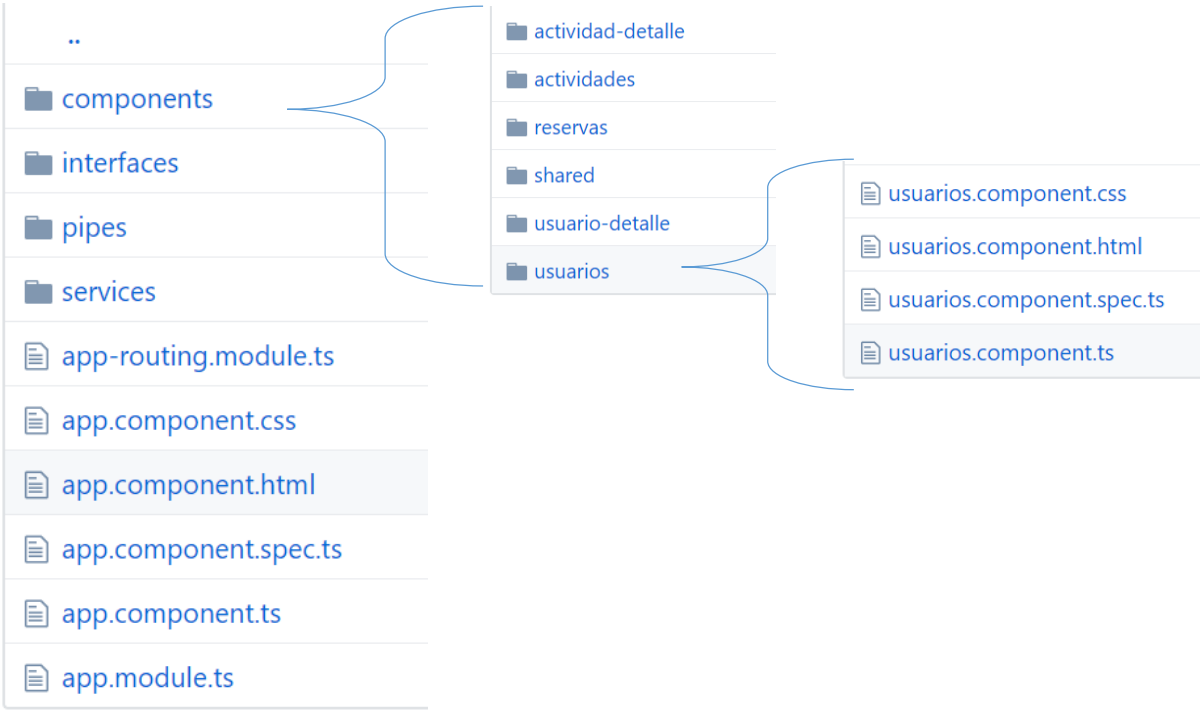
4 BIBLIOGRAFÍA Y WEBGRAFÍA

- FLATICON (2013) Consulta de todos los iconos de la aplicación. https://www.flaticon.com/
- Manuel Pérez Cardona. (14 OCT 2016). Firebase, qué es y para qué sirve la plataforma de Google. 7 MAY 2019, de IEBS Sitio web: https://www.iebschool.com/blog/firebase-que-es-para-que-sirve-la-plataforma-desarroladores-google-seo-sem/
- Google Developers.(2019) Conoce Android Studio. 08 MAY 2019, de User guide Android Sitio web: https://developer.android.com/studio/intro/?hl=ES
- pedrini210. (2016). Carácteristicas y cualidades de Android Studio. 08 MAY 2019, de DESDE LINUX Sitio web: https://blog.desdelinux.net/caracteristicas-y-cualidades-de-android-studio/
- LAURA MARION GOEVERT, EXPERTA EN CHATBOTS Y NATURAL LANGUAGE PROCESSING. (2017). Evolución del mercado de las apps hasta 2021. 14 MAY 2019, de Vanessa Estorach Sitio web: https://www.vanessaestorach.com/evolucion-mercado-de-las-apps-2021/
- Heroku. (2019). The Heroku Platform. 24/05/2019, de Heroku Sitio web: https://www.heroku.com
- Android Developers. (2019). Componentes. 28 MAY 2019, de Android Sitio web: https://developer.android.com/guide/components/activities
- Sgoliver. (2014). Estructura de un proyecto Android (Android Studio). 29 MAY 2019, de Sgoilver.net Sitio web: http://www.sgoliver.net/blog/estructura-de-un-proyecto-android-android-studio/

5 ANEXOS

5.1 MYS ADMINISTRATOR

El código de Move Your School Administrator se organiza en varios paquetes (Components, interfaces, pipes y services). Vamos a centrarnos en ejemplo del código de los usuarios:



Ejemplo de código Usuarios:

- usuarios.component.css

```
.card-img-top {
  width: 150px;
  height: 150px;
  border-radius: 50%;
}
/*

.btn {
  border-width: 0px;
  background-color: #AEC6CF;
}
 */
```

- usuarios.component.html

```
<div class="card-columns mt-5">
  <div class="card text-center" *ngFor="let usuario of usuarios; let i = index;">
    <img class="card-img-top" *ngIf="usuario.urlFoto != null && usuario.urlFoto != ''" [src]="usuario.urlFoto" class="card-img-top mt-2" alt="...">
    <img class="card-img-top" *ngIf="usuario.urlFoto == '' || usuario.urlFoto == null" src="assets/images/idea.png" class="card-img-top mt-2" alt="...">
    <div class="card-body">
      <h5 class="card-title">{{usuario.nombre}}</h5>
      <p class="card-text">{{usuario.email}}</p>
      <p class="card-text">{{usuario.tipo}}</p>
      <a class="btn btn-primary btn-block" (click)="verDetalle(i)">Ver Detalles</a>
    </div>
  </div>
</div>
```

- usuarios.component.spec.ts

```
import { async, ComponentFixture, TestBed } from '@angular/core/testing';

import { UsuariosComponent } from './usuarios.component';

describe('UsuariosComponent', () => {
  let component: UsuariosComponent;
  let fixture: ComponentFixture<UsuariosComponent>;

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [ UsuariosComponent ]
    })
    .compileComponents();
  }));

  beforeEach(() => {
    fixture = TestBed.createComponent(UsuariosComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

- usuarios.component.ts

```
import { Component, OnInit } from '@angular/core';
import {FirebaseusuariosService} from '../../services/firebaseusuarios.service';
import {Usuario} from '../../interfaces/usuario';
import {Router} from '@angular/router';

@Component({
  selector: 'app-usuarios',
  templateUrl: './usuarios.component.html',
  styleUrls: ['./usuarios.component.css']
})
export class UsuariosComponent implements OnInit {
  usuarios: Usuario[] = [];

  constructor(private serviceUsuarios: FirebaseusuariosService,
              private router: Router) {
    this.serviceUsuarios.getAllUsers().subscribe(value => {
      for (let clave in value) {
        console.log(value[clave]);
        this.usuarios.push(value[clave]);
      }

    });
  }

  ngOnInit() {
  }

  verDetalle(userIndex: number) {
    this.serviceUsuarios.setUsuarioSeleccionado(this.usuarios[userIndex]);
    this.router.navigate(['/usuarios', this.usuarios[userIndex].uid]);
  }


}
```

5.2 BaseActivity código

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_base);

    showCart = false;

    navigationView = (NavigationView) findViewById(R.id.nav_view);
    navigationView.setNavigationItemSelectedListener(this);

    //Inicialización de los atributos del Nav Header
    final View hView =  navigationView.getHeaderView(0);
    imgMenuImgUser = hView.findViewById(R.id.imgMenuUsuario);
    tvMenuInstitucion = hView.findViewById(R.id.tvMenuInstitucion);

    final FirebaseUser userAuth = FirebaseAuth.getInstance().getCurrentUser();

    //Inicialmente los menús que requieren login no estan accesibles

navigationView.getMenu().findItem(R.id.nav_MisActividades).setVisible(false);
    navigationView.getMenu().findItem(R.id.nav_MisReservas).setVisible(false);
    navigationView.getMenu().findItem(R.id.nav_Usuarios).setVisible(false);
    navigationView.getMenu().findItem(R.id.nav_MisChats).setVisible(false);

    if (userAuth != null) {
        navigationView.getMenu().findItem(R.id.nav_Usuarios).setVisible(true);
        navigationView.getMenu().findItem(R.id.nav_MisChats).setVisible(true);
        navigationView.getMenu().findItem(R.id.nav_logout).setVisible(true);
        navigationView.getMenu().findItem(R.id.nav_login).setVisible(false);
```

[...]

```
  Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);
    getSupportActionBar().setHomeButtonEnabled(true);
    getSupportActionBar().setDisplayHomeAsUpEnabled(true);

    //Carga el Drawer
    final DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
            this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);

    //Decide si estamos en modo Menu o Modo Back en función del parámetro que
recibe el método
    //desde la implementación en el activity
    toggle.setDrawerIndicatorEnabled(setDrawer());
    drawer.addDrawerListener(toggle);
    toggle.syncState();

    //Añadimos un listener para que cuando se haga click en el boton back de
navegación
    //invoquemos el metodo onBackpressed que nos devolvera al activity anterior.
    toggle.setToolbarNavigationClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            onBackPressed();
```

```
        }
    });

    View container = findViewById(R.id.contenedorMain);
    ConstraintLayout rel = (ConstraintLayout) container;
    getLayoutInflater().inflate(cargarLayout(), rel);

}

//Metodo abstracto que utilizamos desde la clase que extienda para cargar el
layout.
public abstract int cargarLayout();

//Metodo abstracto para establecer la modalidad del activity (menu o back)
public abstract boolean setDrawer();

@Override
public void onBackPressed() {
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    if (drawer.isDrawerOpen(GravityCompat.START)) {
        drawer.closeDrawer(GravityCompat.START);
    } else {
        super.onBackPressed();
    }
}

@Override
public boolean onPrepareOptionsMenu(Menu menu) {

    if (FirebaseAuth.getInstance().getCurrentUser() == null) {
        menu.findItem(R.id.menu_Perfil).setVisible(false);
    }

    if (showCart) {
        menu.findItem(R.id.icoCart).setVisible(true);
    }

    return super.onPrepareOptionsMenu(menu);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.base, menu);

    //Buscamos el Elemento de menu que corresponde a la lupa
    MenuItem myActionMenuItem = menu.findItem( R.id.action_search);

    //Inicializamos el SearchView correspondiente a la lupa como invisible
    action_search= (SearchView) myActionMenuItem.getActionView();
    action_search.setVisibility(View.INVISIBLE);

    //Si el activity es una instancia de Lista de Actividades activamos la
SearchView
    if (this instanceof Actividades) {
        myActionMenuItem.setVisible(true);
        action_search.setVisibility(View.VISIBLE);
        action_search.setOnQueryTextListener(new
SearchView.OnQueryTextListener() {
```

[...]

MoveYourSchool

Gracias por su atención.