

AUTOR: VÍCTOR SÁENZ MARTÍNEZ

TUTORA: Mª VICTORIA SÁEZ-BENITO

PROYECTO FIN DE CLICLO – DAM

CURSO 2021-2022

**VS MOTORS, GESTIÓN DE ALQUILER DE COCHES**

# **Índice**

**1) Introducción 2**

**2) Planificación 2**

**3) Análisis 4**

a) Alcance 4

b) Objetivos 4

**4) Diseño 5**

a) Descripción componentes software 5

b) Esquema funcional 5

c) Diseño de base de datos 6

d) Diseño de clases. 9

**5) Desarrollo e Implementación 10**

**6) Pruebas 25**

**7) Ampliación y posibles mejoras 27**

**8) Conclusión 28**

**9) Bibliografía 28**

**1 Introducción**

Mi proyecto consiste en una aplicación para la gestión de una hipotética empresa de alquiler de coches.

Su finalidad es ofrecer un catálogo de automóviles y una herramienta desde la que poder reservar un coche en apenas unos segundos, agilizando tareas como la planificación de viajes o el desarrollo de actividades laborales.

Mi principal motivación a la hora de elegir esta temática ha sido el impacto que tiene el automóvil en el día a día de la mayor parte de la población. Ya sea para llegar a tu centro de trabajo o para disfrutar de unos días de vacaciones, el primer paso suele ser girar la llave del contacto de tu coche.

Esta aplicación pretende ser intuitiva y fácil de usar, de forma que los usuarios encuentren en su interfaz un sitio cómodo donde reservar un coche o explorar los distintos modelos de los que dispone. Un lugar al que recurrir cuando necesitas una solución lo más rápida posible.

La hipotética empresa a la que pretende gestionar dispone de una sede, donde se encuentran los vehículos publicados en el stock de la aplicación, y a la que se deberá acudir para recoger el vehículo reservado en su caso.

**2 Planificación**

Para agilizar la planificación a la hora de desarrollar la aplicación he utilizado un modelo de metodología ágil, SCRUM. Esto facilita la tarea dividir el proyecto en fases o sprints y ponerme plazos para la realización de los mismos. Cada fase será planificada con anterioridad, se evaluará el resultado que se quiere obtener en cada una y un tiempo estimado para su realización. Asimismo, al finalizar el desarrollo de una fase se hará una pequeña prueba para encontrar posibles fallos o mejoras.

**Fases en la creación de la aplicación:**

**1 Diseño de las principales interfaces:**

* Boceto de varias pantallas para ir construyendo la app
* Plazo de 4 horas para su realización

**2 Desarrollo de los menús y enlaces entre pantallas**

* Plazo de 2 horas para su realización

**3 Desarrollo y conexión de la BD (Firebase):**

* Plazo de 3 horas para su realización

**4 Creación del login y registro de usuarios:**

* Desarrollo de la interfaz y el código necesario para registrar usuarios en la BD
* Plazo de 3 horas para su realización

**5 Creación de la interfaz de la pantalla “catálogo”:**

* Plazo de 5 horas para su realización

**6 Desarrollo del diálogo que permite reservar un coche:**

* Plazo de 4 horas para su realización

**7 Desarrollo de la funcionalidad de reserva de coches:**

* Plazo de 6 horas para su realización

**8 Creación de la pantalla de información sobre el coche:**

* Plazo de 2 horas para su realización

**9 Desarrollo de la pantalla relacionada con la información del usuario**

* Plazo de 3 horas para su realización

**3 Análisis**

**3a Alcance**

Dentro del alcance de la aplicación vamos a definir para que usuarios o dispositivos está orientada la app:

**En cuanto al dispositivo para el que está desarrollada:**

Al ser una aplicación móvil, desarrollada en Android Studio, solo estará disponible para tablets, móviles, o cualquier otro dispositivo que funcione con el SO Android.  
Dentro de este grupo cabe especificar que no es compatible con todas las versiones de Android, concretamente funciona entre las versiones de SDK de la 27 a la 31, que corresponden a las versiones de Android entre la 8.1 y la 12.

La aplicación solo va a estar disponible en castellano, lo que limita a los usuarios que no hablen este idioma.

**En lo respectivo a la funcionalidad:**

Al tratarse del proceso de alquiler de vehículos, esta aplicación no está enfocada para personas menores de 18 años, o aquellas que no tengan el permiso de circulación vigente.

**3b Objetivos**

El objetivo principal es constituir una herramienta que permita reservar coches y consultar el catálogo para comparar especificaciones de los mismos.

Dado que los medios informáticos están siendo la plataforma más usada a la hora de contratar un servicio o adquirir un producto, debido en gran parte a la comodidad de la gestión, muchos usuarios pueden estar interesados en reservar su coche a través de esta aplicación.

Como mencioné en la introducción, la hipotética empresa que pretende gestionar solo entrega los coches en su sede, por lo que esta herramienta está enfocada a la reserva y consulta de los coches disponibles.

El sistema de usuarios facilita en gran medida el trabajo a la empresa, ya que con solo introducir la id del usuario puede acceder a los datos de alquiler del mismo, mejorando así la calidad del servicio.

**4 Diseño**

**4a Descripción de los componentes de software**

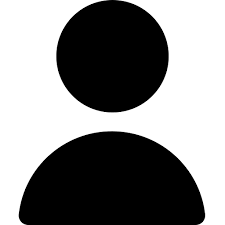
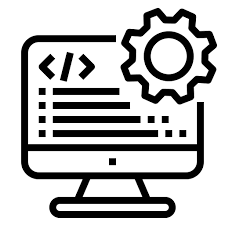
Para realizar esta aplicación móvil, he usado el IDE “Android Studio”, con los plugins necesarios para la implementación del servicio de bases de datos “Firebase”.

El lenguaje en el que se ha realizado es Java, la alternativa más usada sobre Android Studio.

Otro punto a destacar es la versión en Android en la que está construido el proyecto, o el nivel de API. Yo he elegido el nivel de api 31, correspondiente a la versión 12 de Android. Esta versión cuenta con las últimas actualizaciones de los componentes del SDK. Como desventaja, la aplicación solo permitirá ejecutarla a los dispositivos con un Android 8.1 o mayor, dejando atrás a aquellos que funcionen con versiones anteriores, que representan aproximadamente el 16.5% del total.

Tanto Android Studio como Firebase son gratuitas, aunque esta última, gestionada por Google, impone tasas para proyectos de nivel empresarial.

**4b Esquema funcional**



Usuario

Programa

Introduce credenciales

Inicio Sesión/

Registro

Valida las credenciales

Elegir tipo y modelo en el catálogo

Envía los datos del coche elegido al dialogo

Consultar Coche

Consultar los datos del perfil activo

Acceder a la pantalla ‘mi perfil

Acceder a la pantalla ‘mi coche’

Consultar los datos de cochesPrestados para ese registro en concreto

Consultar el coche en alquiler

Añade un registro a la colección cochesPrestados

Elegir el modelo y el periodo de alquiler

Alquilar Coche

Cargar los datos de la bd del usuario activo con su id

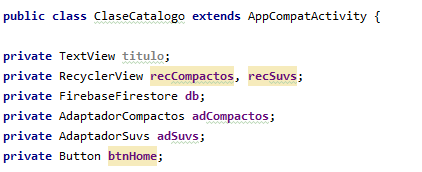
**4c Diseño de la base de datos**

Para el almacenamiento y la gestión de los datos de la aplicación he usado la base de datos no relacional “Firebase”, administrada por Google.

Este sistema consiste en una serie de colecciones, sustituyendo a las tablas de un sistema relacional. Dentro de cada colección se puede crear un número indefinido de documentos. En mi caso, cada documento corresponde a un objeto, cuyos elementos serán sus propiedades, denominadas “campos” por la base de datos.

Para acceder a los datos de un documento en concreto, se llama a la colección que lo contiene y dentro de ella al propio documento, simplemente con su nombre. En este punto podremos seleccionar sus campos.

Para la consulta de los datos, en primer lugar se tendrá que crear el objeto que va a generar la conexión con el servidor de Firebase: FirebaseFirestore.

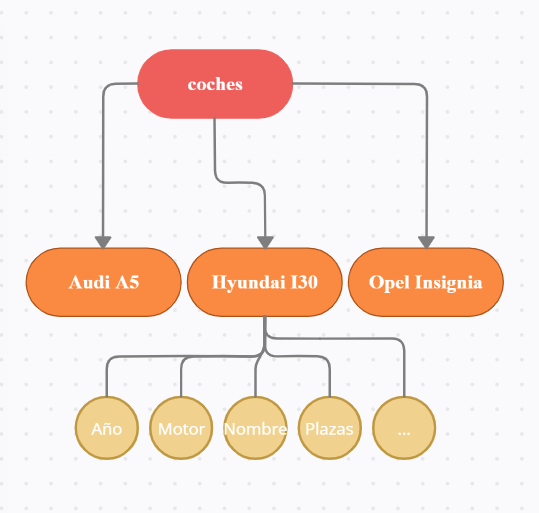


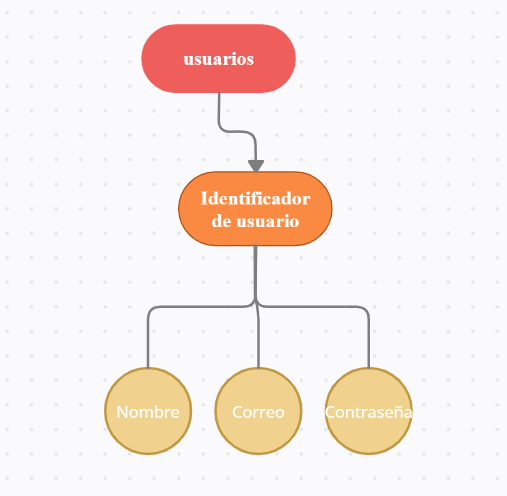
Después de crea la consulta, a partir del complemento anterior. Dentro de la consulta se especifica la colección en la que queremos buscar los datos, y la restricción que queremos aplicar:

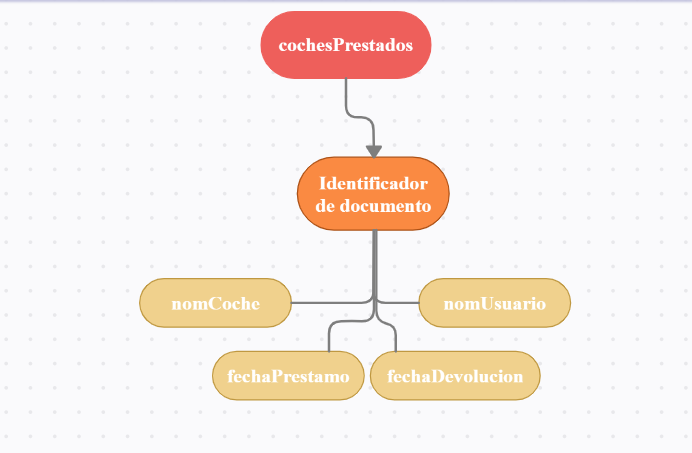


Explicado esto, podemos definir la estructura de este sistema como un diagrama de árbol, donde cada colección formara un bloque separado de las demás.

Para abarcar todos los datos del proyecto han sido necesarias tres colecciones, una que representa a los usuarios, otra a los coches, y una tercera para registrar cada reserva realizada por los usuarios.

**Esquema de las colecciones:**

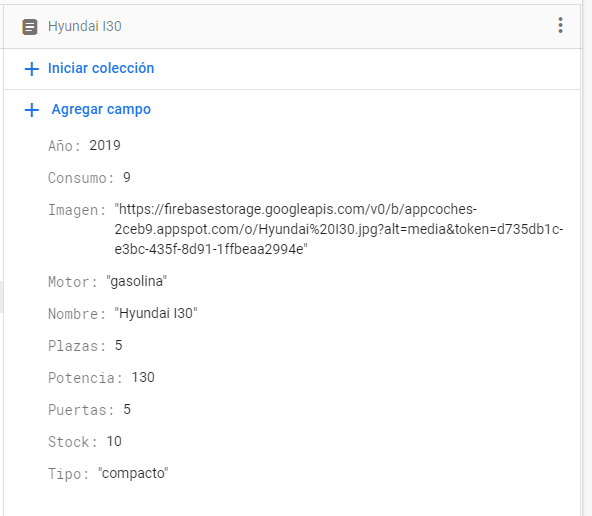




**Almacenamiento de las imágenes**

Para gestionar las imágenes que estarán asociadas a los coches, he usado el almacenamiento en la nube de Firebase, guardando todos estos recursos en una carpeta virtual.

Para asociar a cada coche su respectiva imagen he añadido un campo a cada documento “coche”, que contiene la URL en la que Firebase ha almacenado esta imagen:



Para obtener esta imagen y asociarla a un elemento visual en Android Studio, he usado la librería “Glide”. Esta clase permite cargar y mostrar recursos de muchas fuentes diferentes, como servidores remotos o el sistema local de archivos.

En mi caso le he proporcionado el objeto que representa al control “imageView” (1) de Android, donde se mostará la foto, y la url del recurso (2), extraida de la consulta a la base de datos con el campo “Imagen”:



**(2)**

**(1)**

**4d Diseño de clases**

|  |
| --- |
| CocheCompacto |
| + Nombre: String  + Año: long  + Consumo: long  + Motor: String  + Puertas: long  + Plazas: long  + Stock: long  + Tipo: String  + Imagen: String  + Potencia: long |

|  |
| --- |
| CocheSuv |
| + Nombre: String  + Año: long  + Consumo: long  + Motor: String  + Puertas: long  + Plazas: long  + Stock: long  + Tipo: String  + Imagen: String  + Potencia: long |

|  |
| --- |
| CocheBerlina |
| + Nombre: String  + Año: long  + Consumo: long  + Motor: String  + Puertas: long  + Plazas: long  + Stock: long  + Tipo: String  + Imagen: String  + Potencia: long |

|  |
| --- |
| Furgoneta |
| + Nombre: String  + Año: long  + Consumo: long  + Motor: String  + Puertas: long  + Plazas: long  + Stock: long  + Tipo: String  + Imagen: String  + Potencia: long |

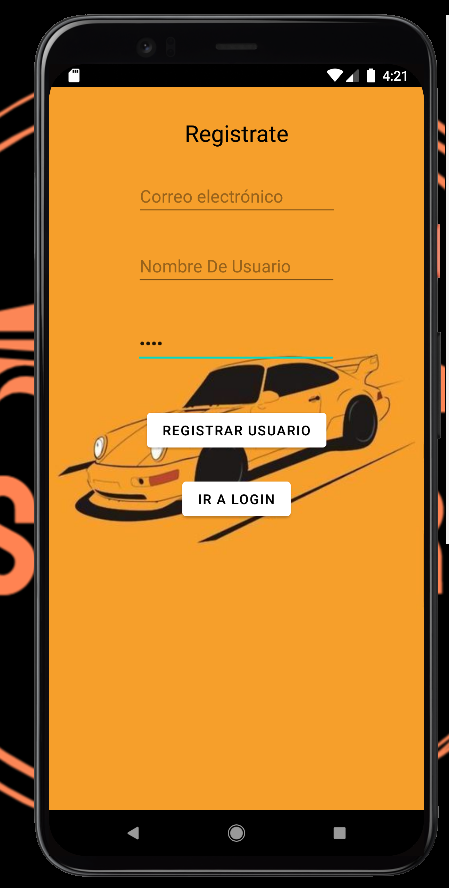
|  |
| --- |
| Prestamo |
| + NomUsuario: String  + NomCoche: String  + fechaDevol: date  + fechaPrestamo: date |

**5 Desarrollo e Implementación**

**Pantalla de Registro**

Su función es recoger datos de los nuevos usuarios para que estos puedan empezar a usar la aplicación.

**Interfaz:**



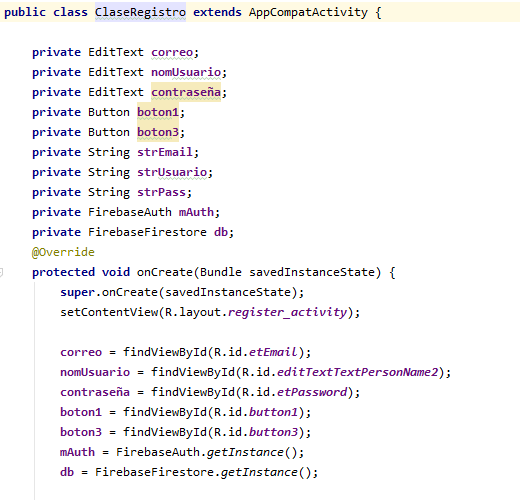
**Funcionamiento:**

Uno de los botones te redirecciona al login, en caso de que ya tengas cuenta y quieras iniciar sesión.

Al introducir los datos en los editText y pulsar el botón “registrar usuario”, añade las credenciales a la base de datos, y abre la pantalla principal.

Para gestionar el correo y la contraseña, valores que usaremos cada vez que queramos iniciar sesión en la appc función de Firebase dedicada a este fin: FireBaseAuthentication. Esta clase se comunica con un gestor de credenciales de la base de datos, proporcionando una forma segura de acceder a tu cuenta y simplificando en gran medida el código necesario para registrarse e iniciar sesión.

Creamos la instancia a la clase FirebaseAuth.



llamamos a su método createUserWithEmailAndPassword, y le pasamos el valor de los campos de texto de “correo” y “contraseña”.

El propio metodo tiene por defecto una función de prevención de errores al estilo Try-Catch.

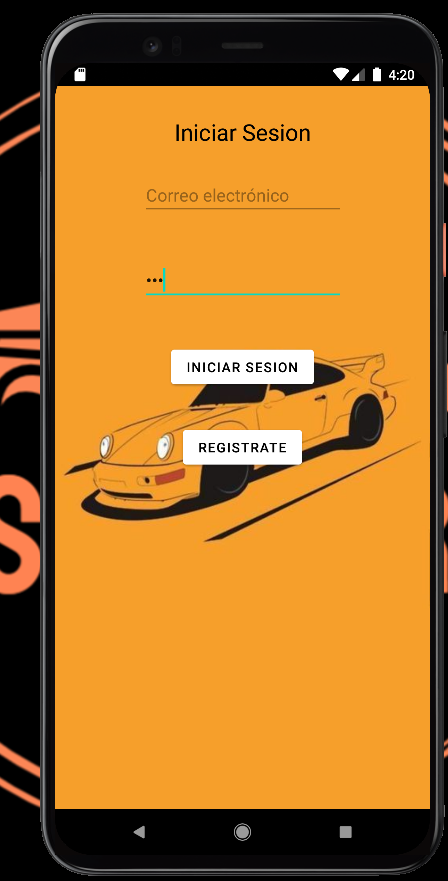
Si la transacción sale bien, almacena los datos del usuario en una colección de la base de datos, y en el gestor de credenciales de Firebase.



**Pantalla de login**

Permite al usuario iniciar sesión con sus credenciales, o ir al registro para crear una cuenta.

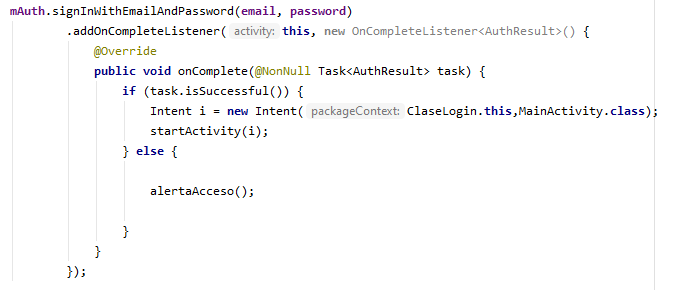
**Interfaz:**



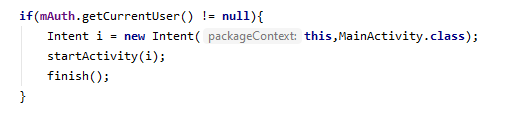
**Funcionamiento:**

Al introducir las credenciales correctamente y pulsar en iniciar sesión, accede a la pantalla main de la aplicación con el usuario especificado.

Al igual que en el registro, las credenciales son administradas por Firebase authentication. En este caso se usa el método “singnInWithEmailAndPassword”, al que se le administran las credenciales introucidas, y él se encarga de gestionar la validación.



Firebase también proporciona una función que permite recordar el último usuario que inició sesión en la aplicación y saltarse el login, ya que valida las credenciales automáticamente. Está gestionado por Firebase Authentication.



**Pantalla principal**

Pantalla principal de la aplicación, muestra un resumen de los datos del alquiler del coche, en caso de que el usuario esté alquilando alguno. En caso contrario, invita a reservar uno. En la parte de abajo tiene un menú de navegación que permite desplazarse entre las pantallas de la aplicación. Hay uno similar en cada pantalla.

**Interfaz:**





**Funcionamiento:**

El cartel de información está cargado con los datos de la colección ‘cochesPrestados’, que almacena los datos referentes a los préstamos. Se envía una consulta a dicha colección, restringiendo por nombre de usuario. Si el usuario no tiene datos en cochesPrestados, significa que no tiene ningún coche en alquiler. Entonces se muestran los datos de la segunda captura.

El menú de navegación está formado por tres botones enlazados con cada una de las pantallas, con un icono representativo.

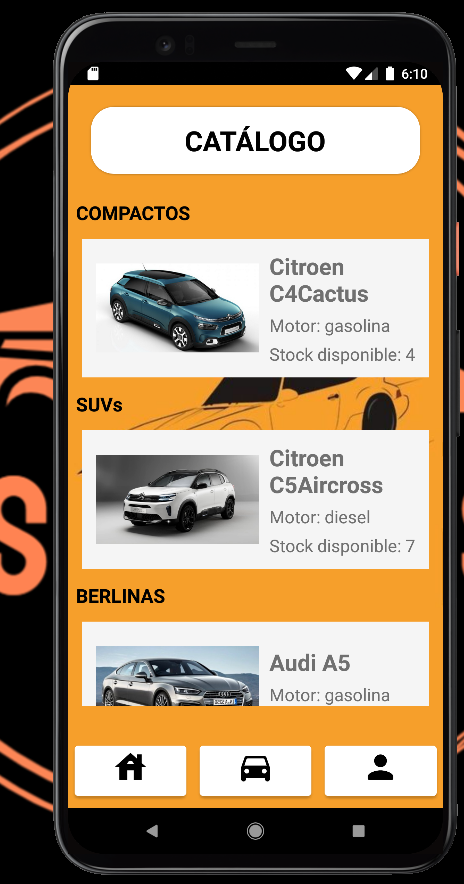
También dispone de un botón que permite cerrar sesión con el usuario activo y volver al login.

**Catálogo**

Es la interfaz donde se encuentran todos los coches disponibles para alquilar, volcados desde la base de datos. Al Pulsar en el elemento de cualquier coche se muestra un diálogo con la información del mismo, y un pequeño formulario para su alquiler.

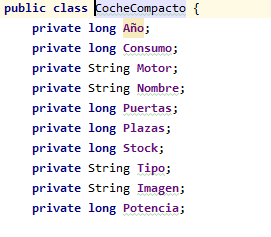
**Interfaz:**

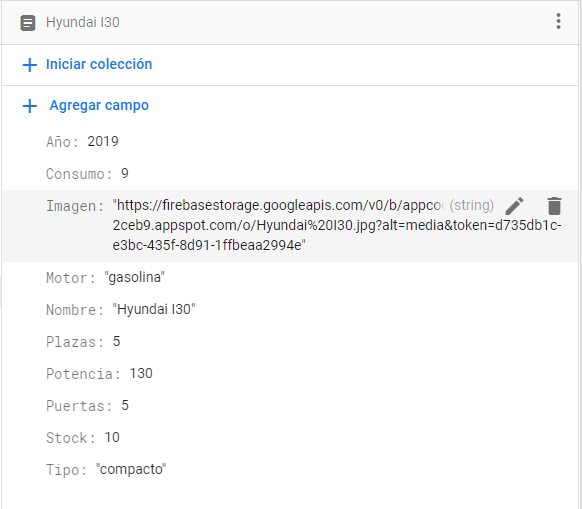
Cada categoría de coches se muestra en un recycler view horizontalmente, los ítems del recycler view llaman a un diálogo, interfaz que explicaré en el siguiente apartado.



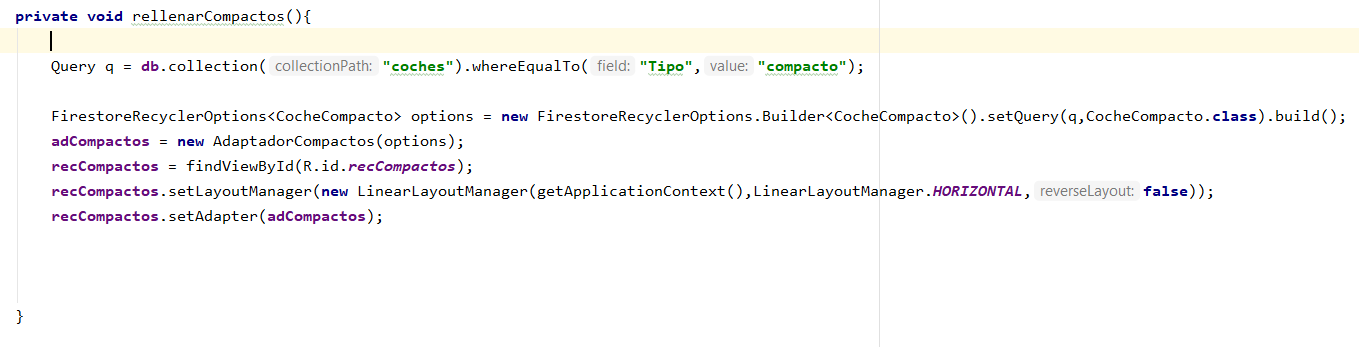
**Funcionamiento:**

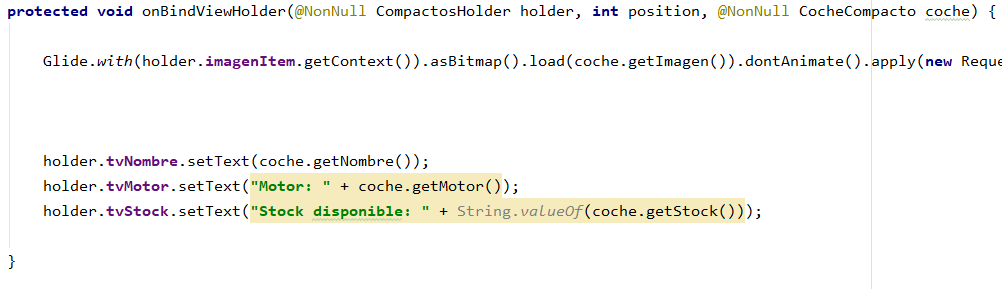
Los datos de cada categoría se muestran en un recycler view. Firebase aporta un modo de implementación para los recyclerViews, simplificando la carga de datos desde la colección determinada. Cada documento que queremos recuperar de la bd se mapea en un objeto de java, previamente construido con los campos del documento que nos interesa obtener.





Una vez ejecutada la consulta, se genera un objeto con los valores de los campos del documento, que son recuperados a través de los getters del mismo.





**Diálogo Alquiler**

Esta interfaz se muestra al pulsar en un ítem de los recyclerView del catálogo, muestra toda la información del coche elegido, y permite su reserva, eligiendo la fecha en la que se debe devolver el coche.

**Interfaz:**

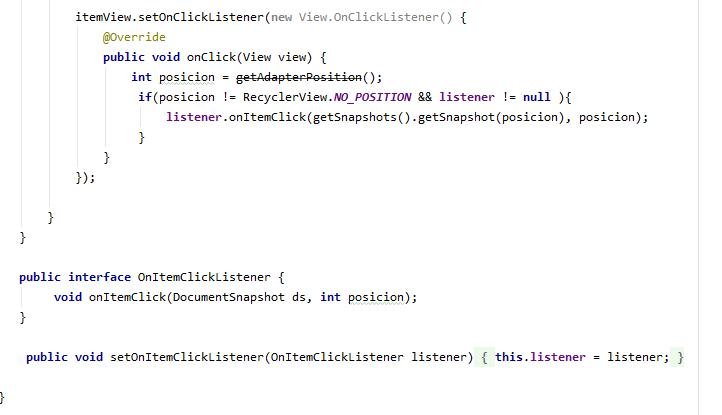


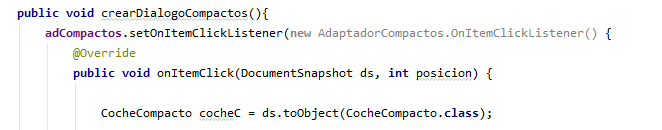
**Funcionamiento:**

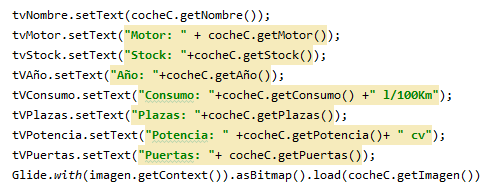
El diálogo se genera a partir de los datos del RecyclerViewItem pulsado, ya que la captura del evento está en el adaptador de dicho reciclerView.

Este método listener enviará al dialogo abierto una instancia de la clase CocheCompato en este caso, al pertenecer al tipo “compactos”. Esta transferencia se realiza gracias a un DocumentSnapshot.

Con este objeto recibido se obtienen todas las propiedades del coche a partir de sus getters.







La imagen, como expliqué anteriormente, se genera a partir de la url almacenada en el campo de la base de datos, utilizando la librería “Glide”

Esta interfaz cuenta con un selector de fecha, que muestra días desde la fecha actual. Con este control el usuario decide cuando tiempo va a tener el coche en alquiler, ya que es la fecha de devolución del coche.

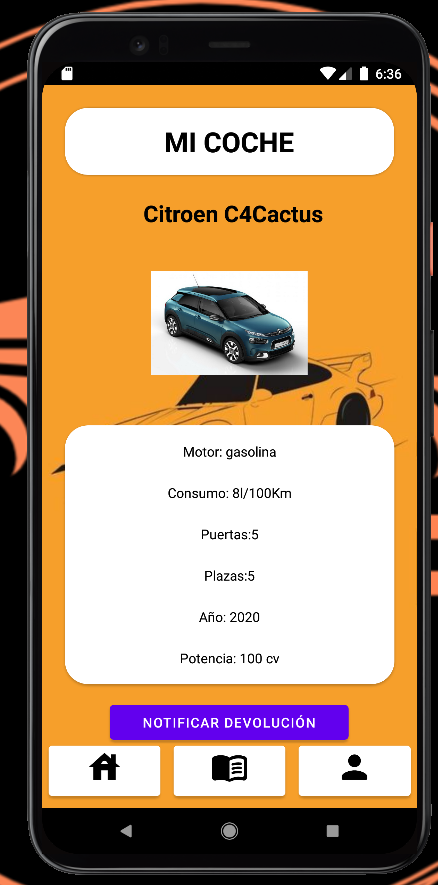
Para acabar, si el usuario decide reservar el coche, hará click en el botón reservar coche, lo que creará un nuevo documento en la colección cochesPrestados de la bd, con los datos del alquiler.

**Mi Coche**

Interfaz en la que se visualizarán los datos relacionados con el coche alquilado por el usuario, en caso de tenerlo.

**Interfaz:**





**Funcionamiento:**

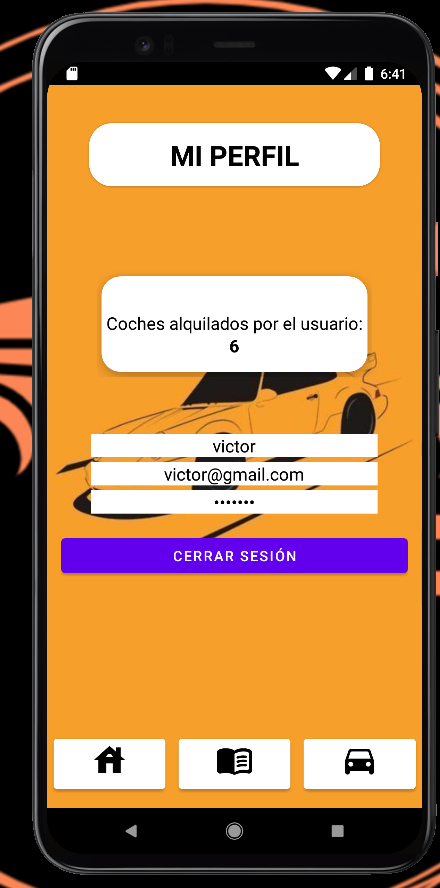
En caso de no tener ningún coche en alquiler, se mostrará la interfaz de la primera captura, al alquilar un coche, se cargan los datos del mismo desde el documento “Coches”, filtrando por el campo común entre las colecciones coches y cochesPrestados: nomCoche.

Esta interfaz permite notificar la devolución del coche a la empresa, esta acción desactivará el alquiler del coche por parte del usuario.

**Mi perfil**

En esta pantalla se mostrarán los datos del usuario activo, y un registro de los coches que ha alquilado en total.

**Interfaz:**



**Funcionamiento:**

El usuario activo se identifica con la función FirebaseAuthentication, que recupera los datos de la sesión actual. El número de coches alquilados totales es un valor acumulativo cargado desde la base de datos

La interfaz también proporciona la opción de cerrar la sesión.

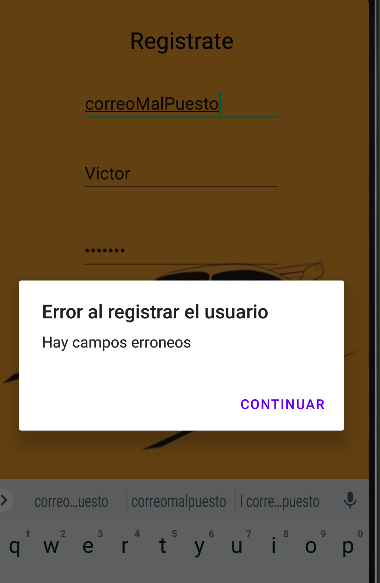
**6 Pruebas**

En esta sección explicare las comprobaciones que he ido realizando durante y después de la realización del desarrollo de la aplicación.

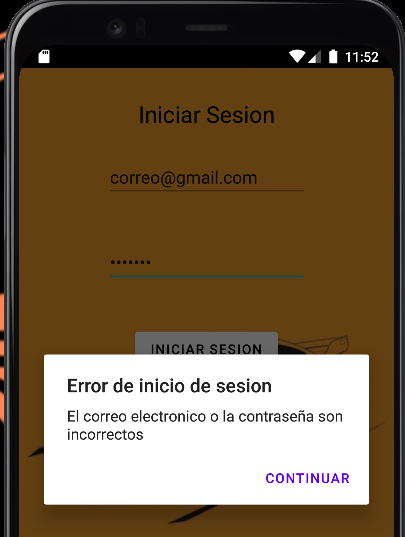
Para estas pruebas, he utilizado el dispositivo virtual de Android Studio. He elegido un dispositivo móvil virtual de tamaño medio (5,7’), que representa la experiencia media que tendría un usuario utilizando la aplicación en su dispositivo.

En primer lugar comprobamos que el registro no permita introducir campos erróneos, especialmente en el EditText que corresponde al correo electrónico.

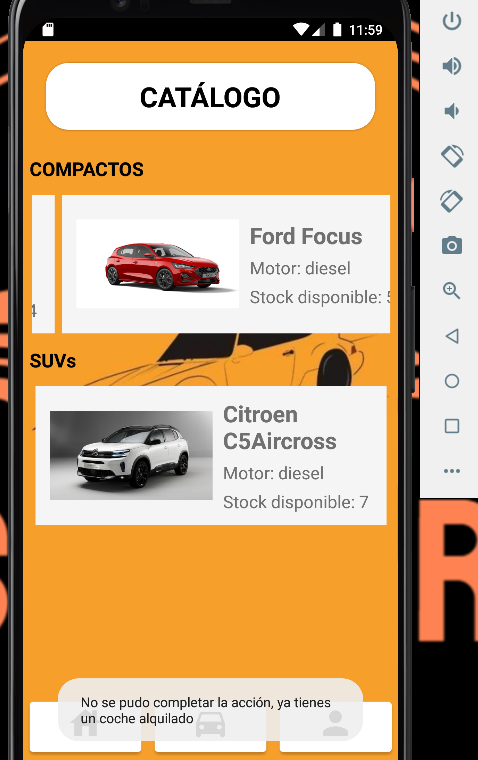
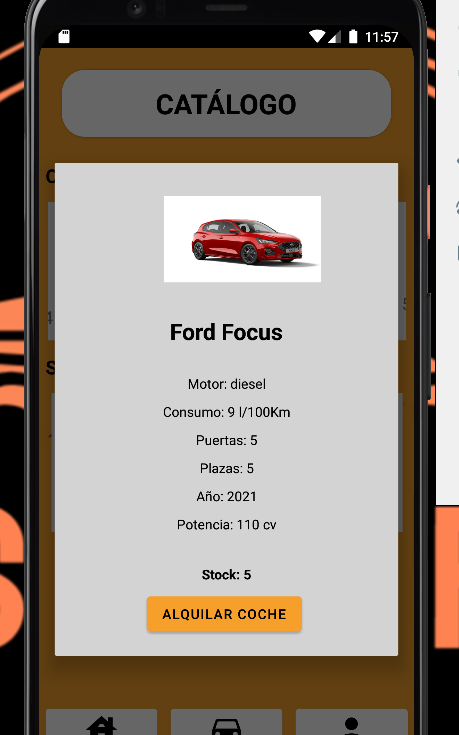
Vemos que si no se introduce el formato de correo adecuado, el programa no permite registrar al usuario.



Siguiendo el recorrido que realizaría el usuario, comprobaremos el comportamiento del login, cuando no se introducen las credenciales correctamente.



Ahora intentaremos alquilar un coche teniendo ya un alquiler activo, el programa no lo permite

**7 Ampliación y posibles mejoras**

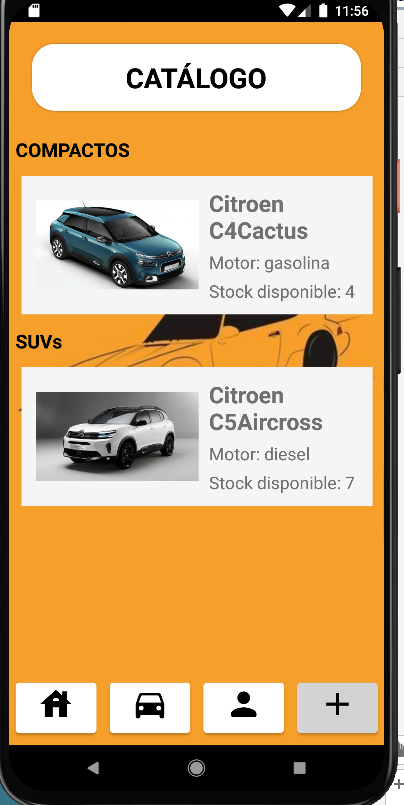
La aplicación cuenta con las funciones básicas necesarias para el alquiler de un coche y el volcado de datos de los diferentes apartados de una empresa de este tipo, pero hay una gran cantidad de funcionalidades que se podrían implementar y mejorarían la experiencia del usuario a la hora de usar esta app en su teléfono.

En esta sección hare un resumen de las que me resultan más viables.

**Creación de usuarios administradores:** a la hora de expandir el catálogo añadiendo nuevos automóviles a la base de datos, la única opción es hacerlo directamente desde la base de datos. Añadiendo esta función, un representante de la supuesta empresa podría añadir coches fácilmente desde un formulario solo visible para este tipo de usuarios.

Darse de alta como administrador requeriría una clave especial, que al introducirla en el registro daría la opción al usuario de ser administrador de la aplicación.

Este formulario se podría asociar a un botón en la barra de navegación de la pantalla “catálogo”, solo visible para administradores.



**Notificación de devolución:** la aplicación podría lanzar una notificación al usuario cuando queden pocos días para la devolución del coche en alquiler.

**Información sobre el combustible:** el combustible consumido en la utilización de un coche de alquiler se tiene que reponer antes de devolverlo. Por eso una pestaña informativa sobre el consumo y precio de este en diferentes estaciones de servicio sería de gran utilidad para el usuario.

Por una parte, el precio del combustible se obtendría de los datos en abierto que publican ciertos organismos de forma informativa, sería un valor a tiempo real.

Por otra parte se implementaría una calculadora del consumo, eligiendo el precio al que se vende el combustible en la gasolinera elegida por el usuario, introduciendo los kilómetros recorridos, se generaría el coste del recorrido realizado.   
Esta función puede ser útil a la hora de organizar un viaje, pues haría parte del trabajo a la hora de calcular el coste total.

**8 Conclusión**

Esta aplicación puede resultar muy útil a empresas que trabajan el sector del alquiler de coches, y quieren dar a conocer su negocio en las redes.

Tener disponible en el móvil, una herramienta que todos nosotros usamos diariamente, la posibilidad de informarse sobre los coches de los que dispone la empresa, y la posibilidad de reservar uno desde cualquier lugar, darían gran ventaja a la empresa en cuanto a visibilidad y marketing digital.

Con las mejoras que he expuesto en el apartado de ampliación, la aplicación sería mas completa y sería de utilidad incluso a la hora de planificar viajes y negocios, aumentando así los usuarios potenciales.

**9 Bibliografía**

* <https://www.youtube.com/c/pildorasinformaticas>
* <https://es.stackoverflow.com/>
* <https://developer.android.com>
* https://firebase.google.com/docs/android/setup