

CICLO FORMATIVO DE GRADO SUPERIOR**DESARROLLO DE APLICACIONES MULTIPLATAFORMA****ACCESO A DATOS - 2º CURSO**

AppEventos

Curso: 2021/22

AUTOR: Mario Parrilla Maroto

Introducción

Motivación y objetivos

He decidido realizar una aplicación de eventos para dar solución a las personas que quieren quedar o pedir cita para un evento o situación, como por ejemplo realizar una reunión, quedar alguien para contratarle...

Descripción de la aplicación

Esta aplicación, permitirá al usuario solicitar eventos a otros usuarios, además de aceptar eventos que le pidan otros usuarios. Los eventos podrán ser presencialmente dando la ubicación del lugar de quedada/evento, o remotamente a través de un enlace de una plataforma de videoconferencia como zoom, google meets,... Además, los eventos tendrán una duración determinada. El usuario al cual le están citando, podrá cancelar eventos.

Especificación de requisitos

Requisitos funcionales

RQ1 - Control de Acceso: Se controlará mediante una ventana con un formulario de login donde los usuarios iniciarán sesión. (Para tener cuenta de usuario en la app, se tendrá que dar de alta desde el CMS). Además desde la ventana de ajustes, se podrá cerrar la sesión del usuario actual.

RQ2 - Gestión de Eventos: En las diferentes ventanas encontraremos las diferentes funcionalidades donde podremos hacer los eventos de CRUD, además de ver mapas según el evento. Además, en esta app al trabajar con los eventos se modificará la base de datos local creada previamente por la app con los datos recibidos del CMS en la sincronización.

RQ2.1 - Ventana de Ajustes: En esta ventana, encontraremos diferentes funcionalidades como acerca de, soporte y cerrar sesión del usuario actual.

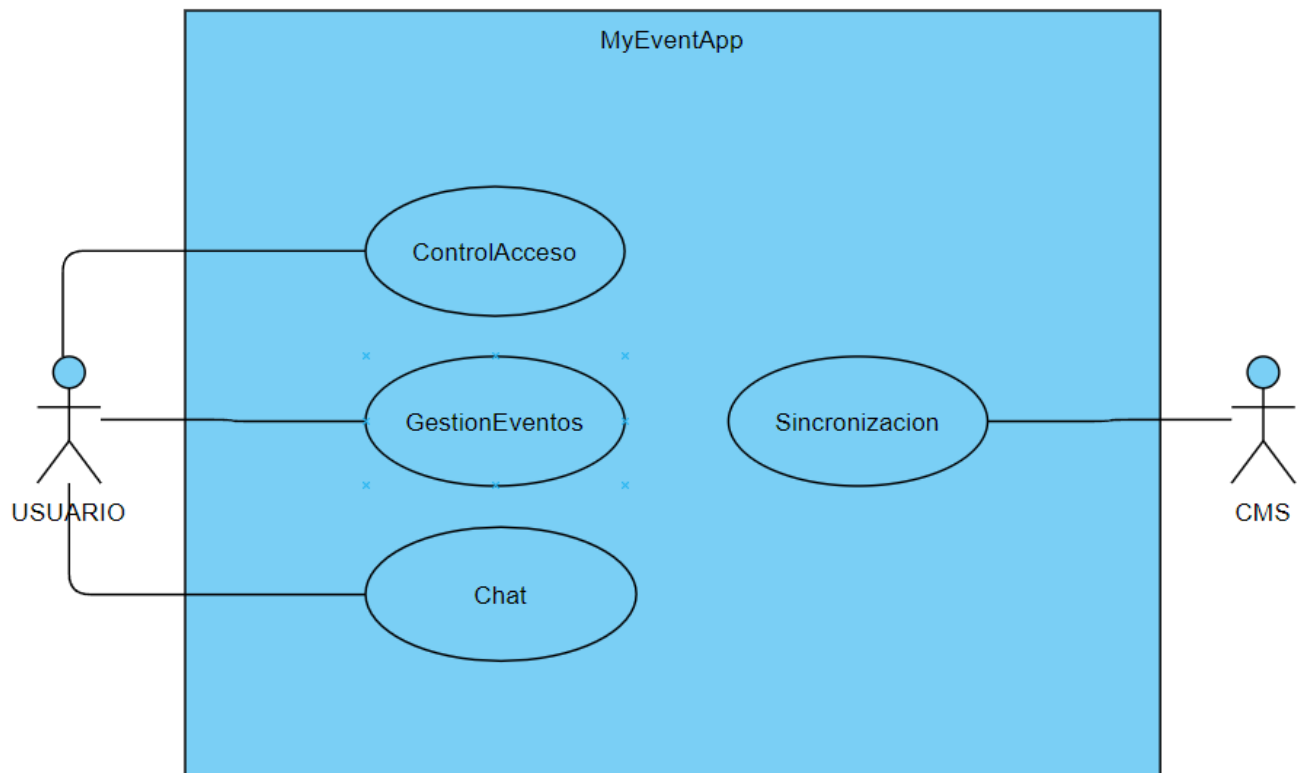
RQ2.2 - Ventana de Inicio: En esta ventana, encontraremos las cardview de los eventos del día actual con datos de importancia, donde el usuario cita a otros usuarios. Si pinchamos sobre la tarjeta de un evento de esta ventana, se nos abrirá un pop up (una nueva ventana), donde podremos ver los detalles al completo del evento.

RQ2.3 - Ventana de Búsqueda: En esta ventana, encontraremos una barra de búsqueda donde podremos filtrar las cardviews por los nombres de los usuarios de la app. Además, veremos las cardview de los usuarios con su username y su descripción. Cuando hagamos click sobre una cardview, se abrirá un pop up (una nueva ventana), donde podremos ver el perfil del usuario con su información y todos sus eventos creados. Si pinchamos sobre un evento del perfil de este usuario, se nos mostrará un pop up (una nueva ventana), con la información detallada de ese evento con la opción de citar este evento.

RQ2.4 - Ventana de Perfil: En esta ventana, encontraremos la información de nuestro usuario, además de las cardviews de nuestros eventos ya creados con la información más importante. Si pinchamos sobre estos eventos, podremos ver su información más detallada, además de poder modificar su información y poder eliminar el evento. También, podremos crear nuevos eventos a través de un fab el cual contendrá otros dos fab, donde podremos o crear un evento presencial, donde se podrá crear el evento con su información precisa o el crear un evento meeting, donde se podrá crear el evento con su información precisa. Por último, si pinchamos en el teléfono de usuario podremos llamarlo o mandar un sms.

RQ3 - Chat: Chat será bluetooth con el cual se podrá hablar con otros usuarios. A este requerimiento se podrá acceder desde una ventana de chat.

RQ4 - Sincronización: El CMS le enviará peticiones a la app para que sincronice los datos de la base de datos local de la app con la base de datos del CMS.



Requisitos no funcionales

Seguridad: El usuario necesitará un usuario con su información para poder acceder a la aplicación.

Conectividad: El dispositivo móvil, necesitará de conexión wifi, bluetooth y gps.

Tipos de dispositivos: Smartphones y tablets android en principio con versiones de android apartir de 5.0 .

Memoria: Los dispositivos necesitarán minimo 512MB de memoria RAM, aunque se recominenda 2Gb.

Procesador: Minimo se necesitará un procesador con Quad Core a 1.2GHz.

Almacenamiento: se necesitará minimo de un 1Gb de almacenamiento.

Version de Android: El dispositivo necesitará una versión minima de android 8.0 Oreo

Wireframe

LOGIN: El usuario entrará con su usuario en la aplicación pudiendo guarda la sesión. Además, podrá recordar la contraseña si no la recuerda.



LOGO

Usuario

Contraseña

☐ Recordar [Recordar Contraseña](#)

REGISTRO: El usuario se podrá registrar en la aplicación y crear su propio usuario.

LOGO

Usuario

Contraseña

Repetir Contraseña

Email

Teléfono

Registrarse

RECORDAR CONTRASEÑA: Aquí el usuario podrá recordar su contraseña recibiendo un correo a su email con un código concreto.

LOGO

Email

Usuario

Código

Reenviar Código

Recordar

NUEVA CONTRASEÑA: Aquí el usuario podrá crear una nueva contraseña para su cuenta.

LOGO

Nueva Contraseña

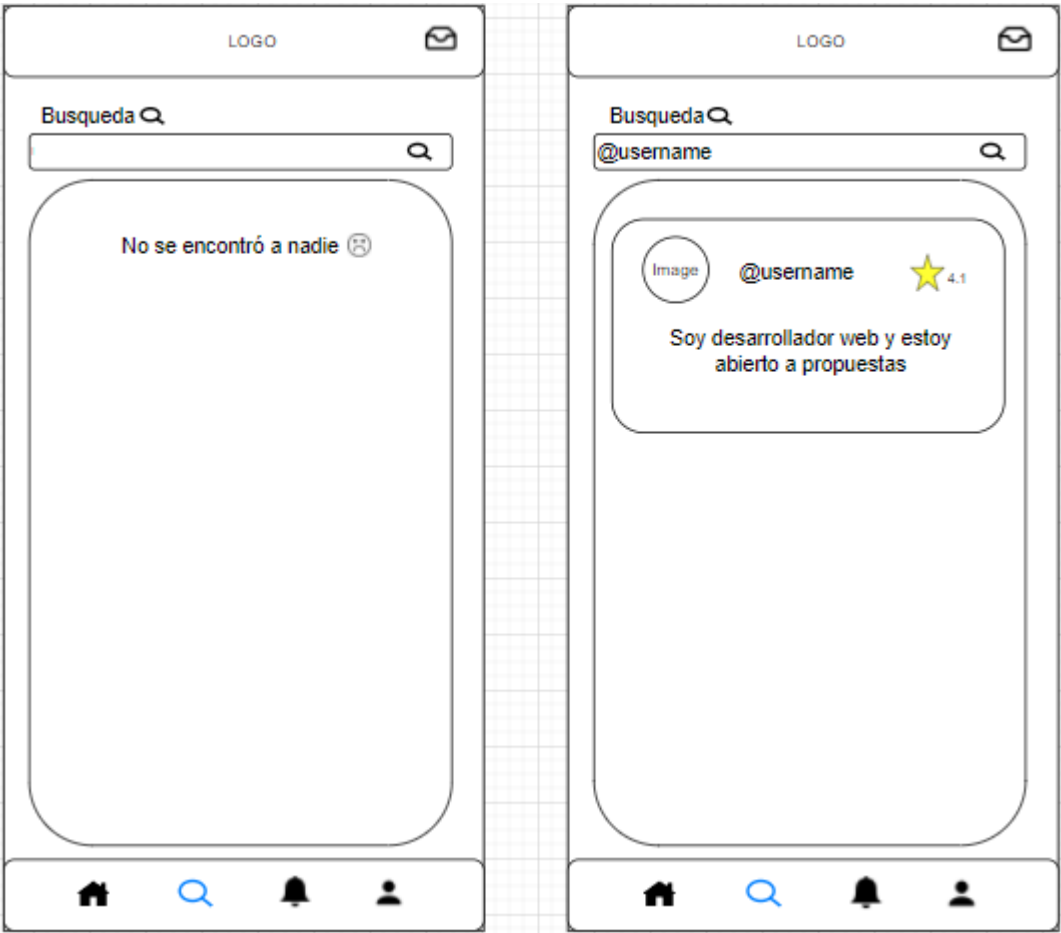
Repetir Contraseña

Login

INICIO: Este es el menu inicial donde podrá encontrar los eventos que tiene ese día, de lo contrario, podrá ir a buscar eventos.



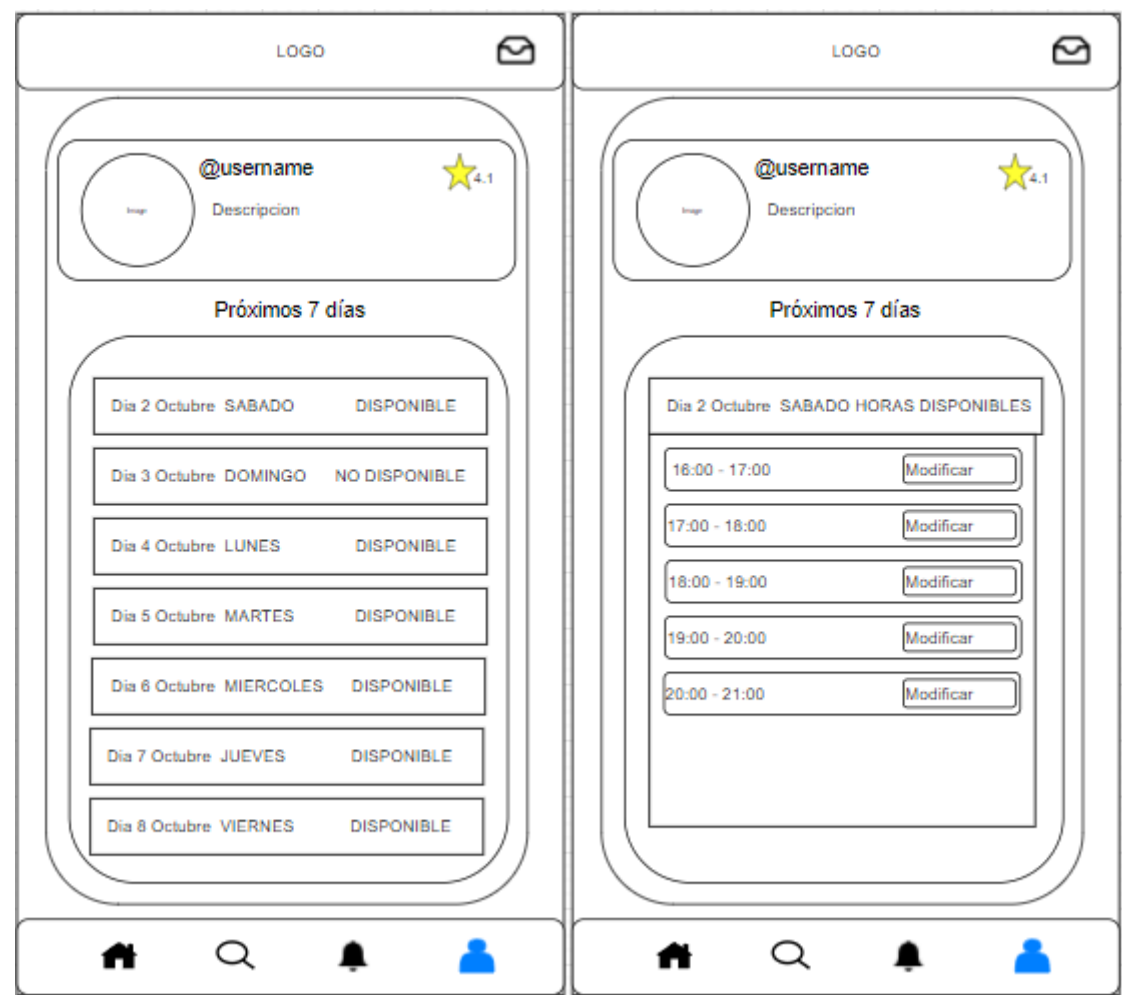
BUSQUEDA: Aquí el usuario buscará a los usuarios a los cuales desea citar un evento.



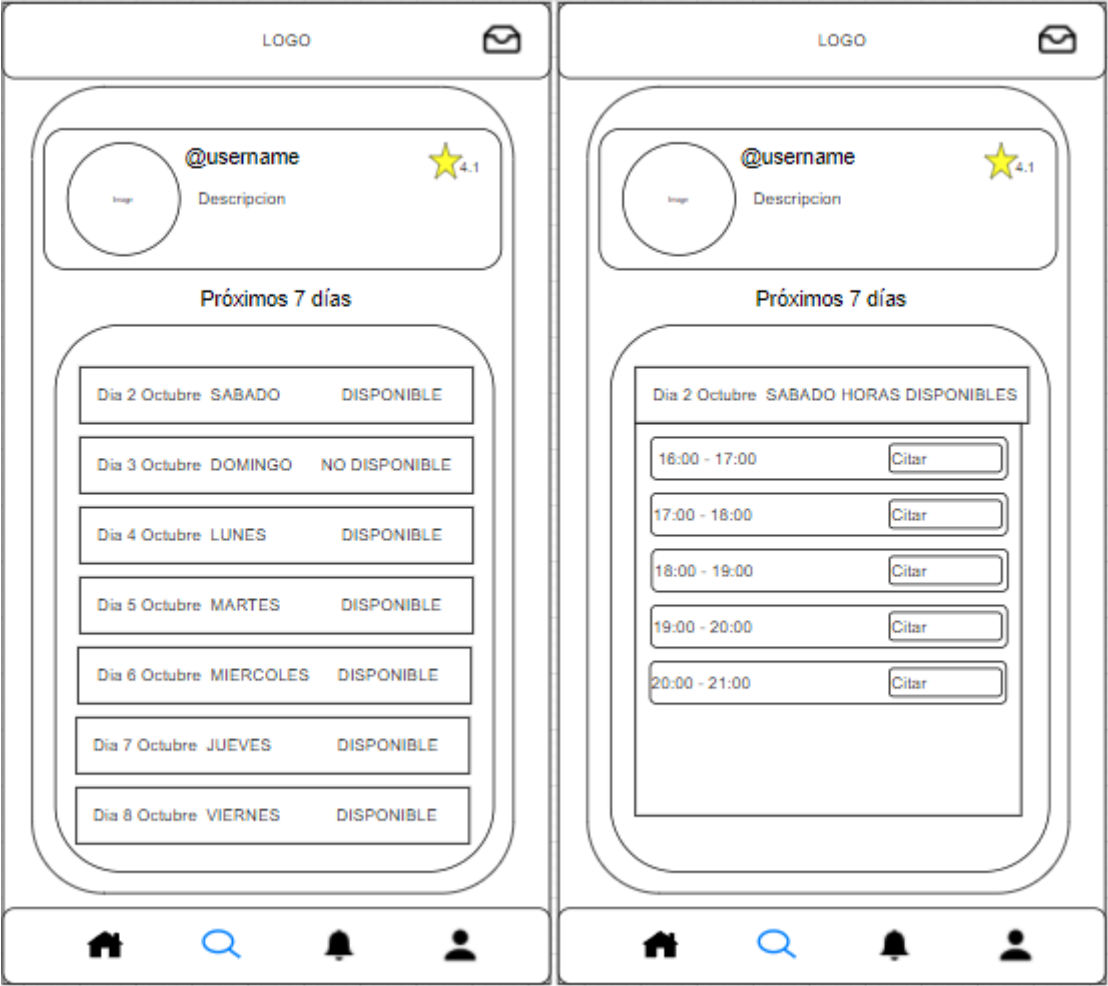
NOTIFICACIONES: Aqui se mostrarán las notificaciones de la aplicación.



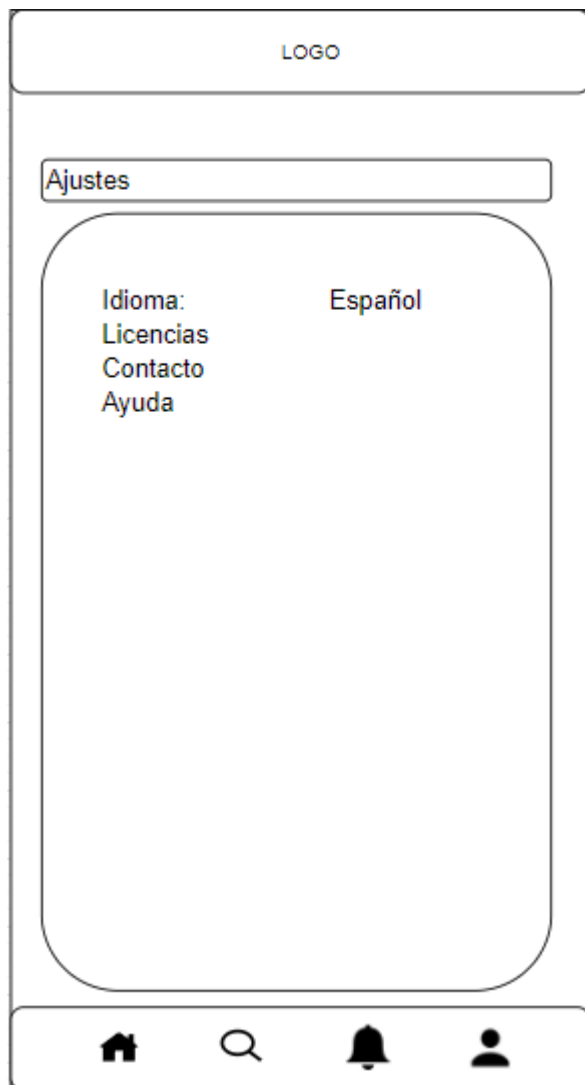
PERFIL PERSONAL: Aquí el usuario verá sus eventos pudiendo modificarlos a su elección.



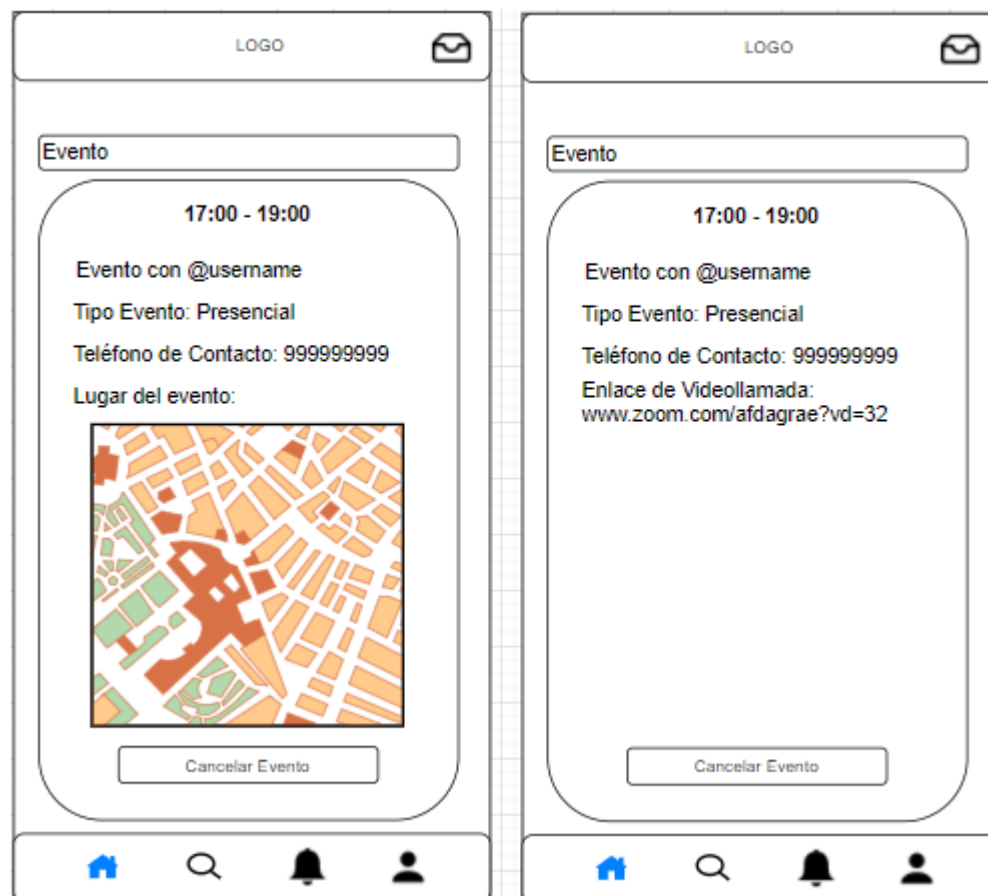
PERFIL AJENO: Aquí el usuario podrá elegir el día y la hora disponible para citar un evento a otro usuario.



AJUSTES: Aquí encontrará la selección del idioma de la aplicación, además de otros datos de interés.



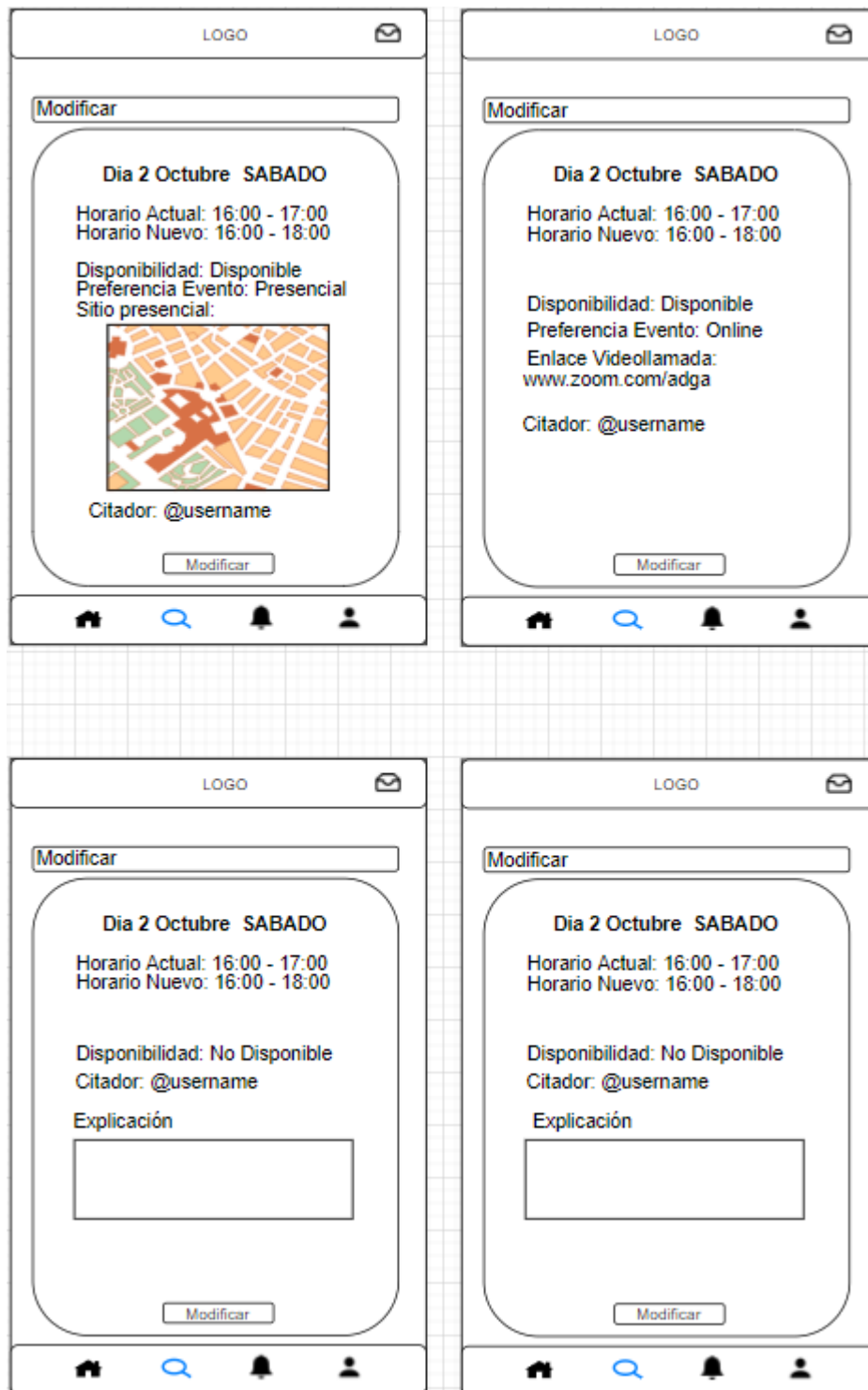
EVENTO: Aquí el usuario citador podrá ver la información de un evento pudiendo cancelar su cita.



CITAR EVENTO: El usuario podrá seleccionar los datos de la cita.



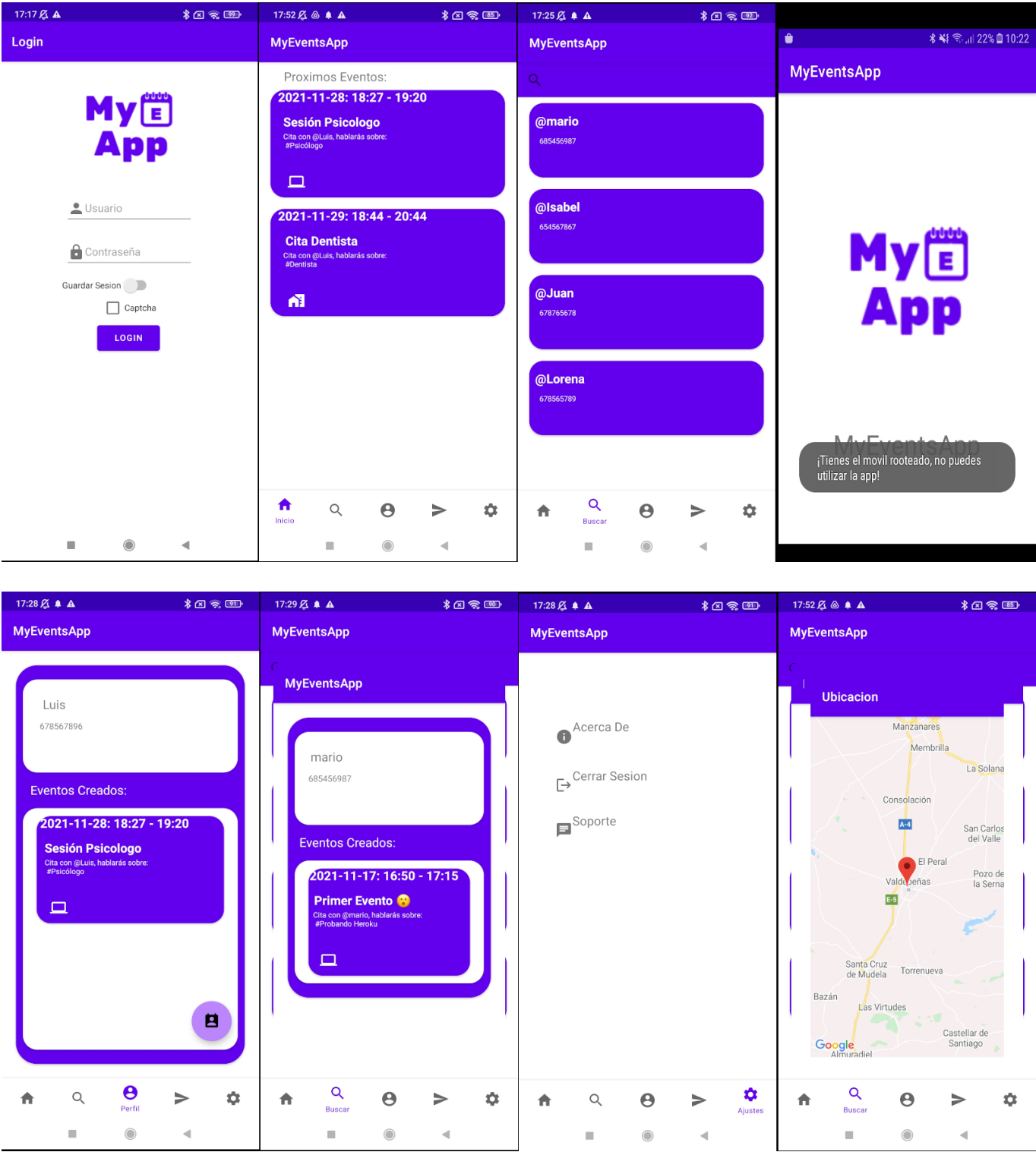
MODIFICAR EVENTO: El usuario podrá editar sus eventos e incluso cancelarlos.

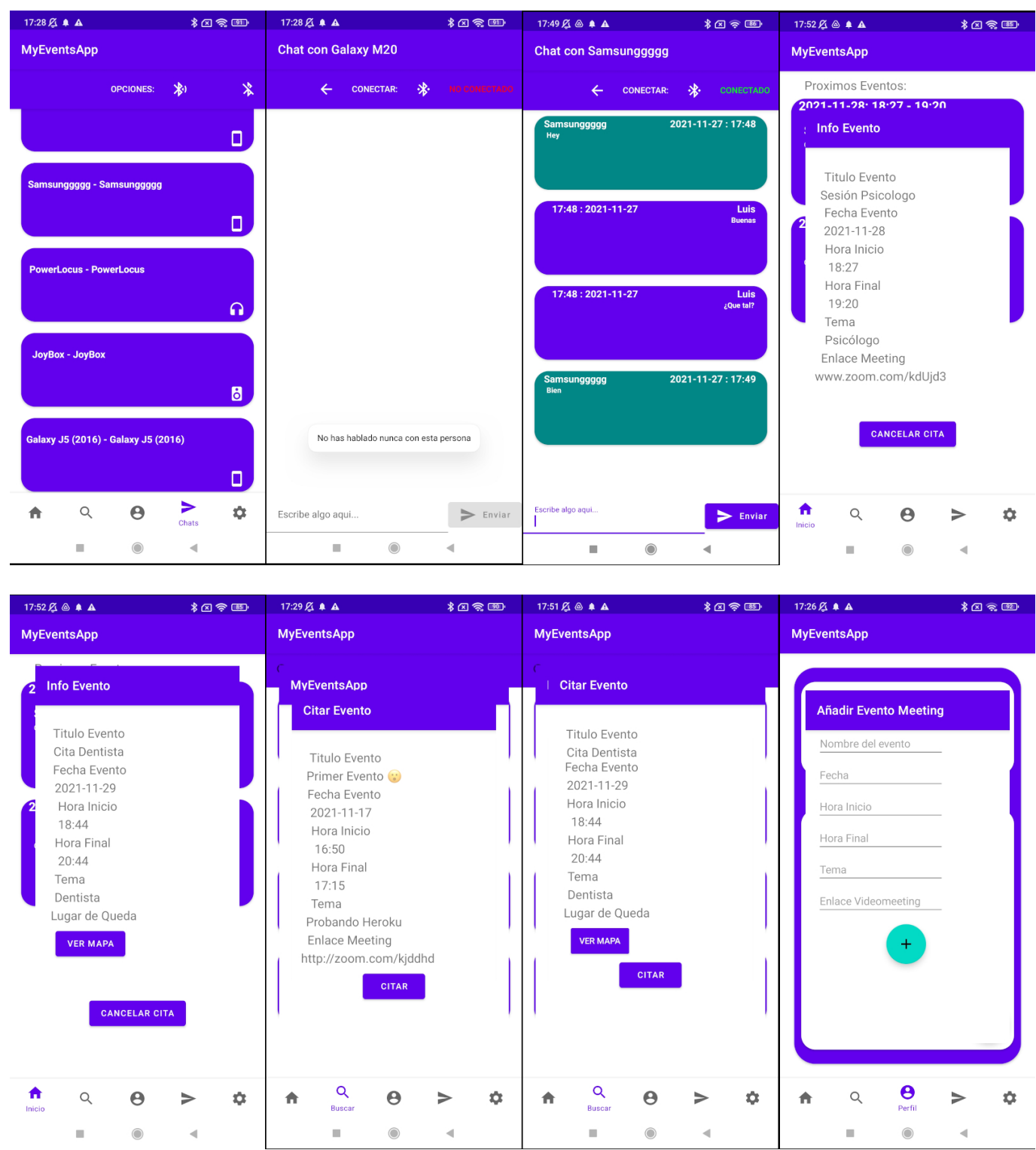


Análisis Funcional

Interfaz gráfico

App:





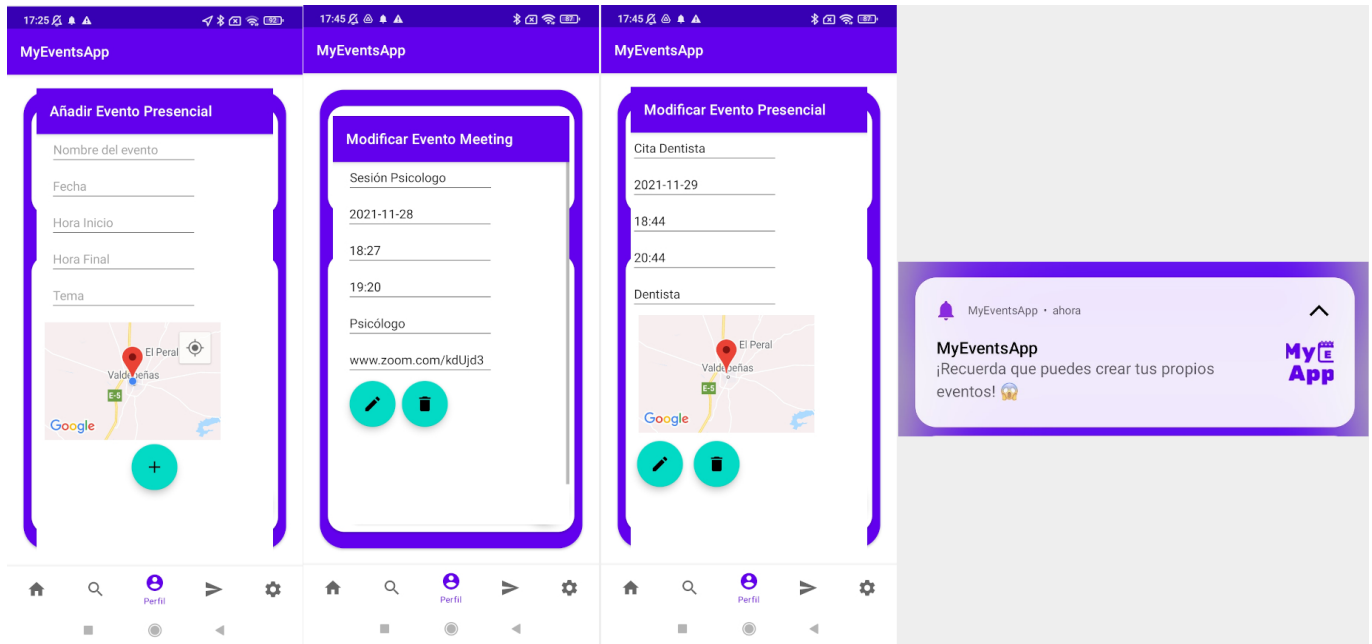
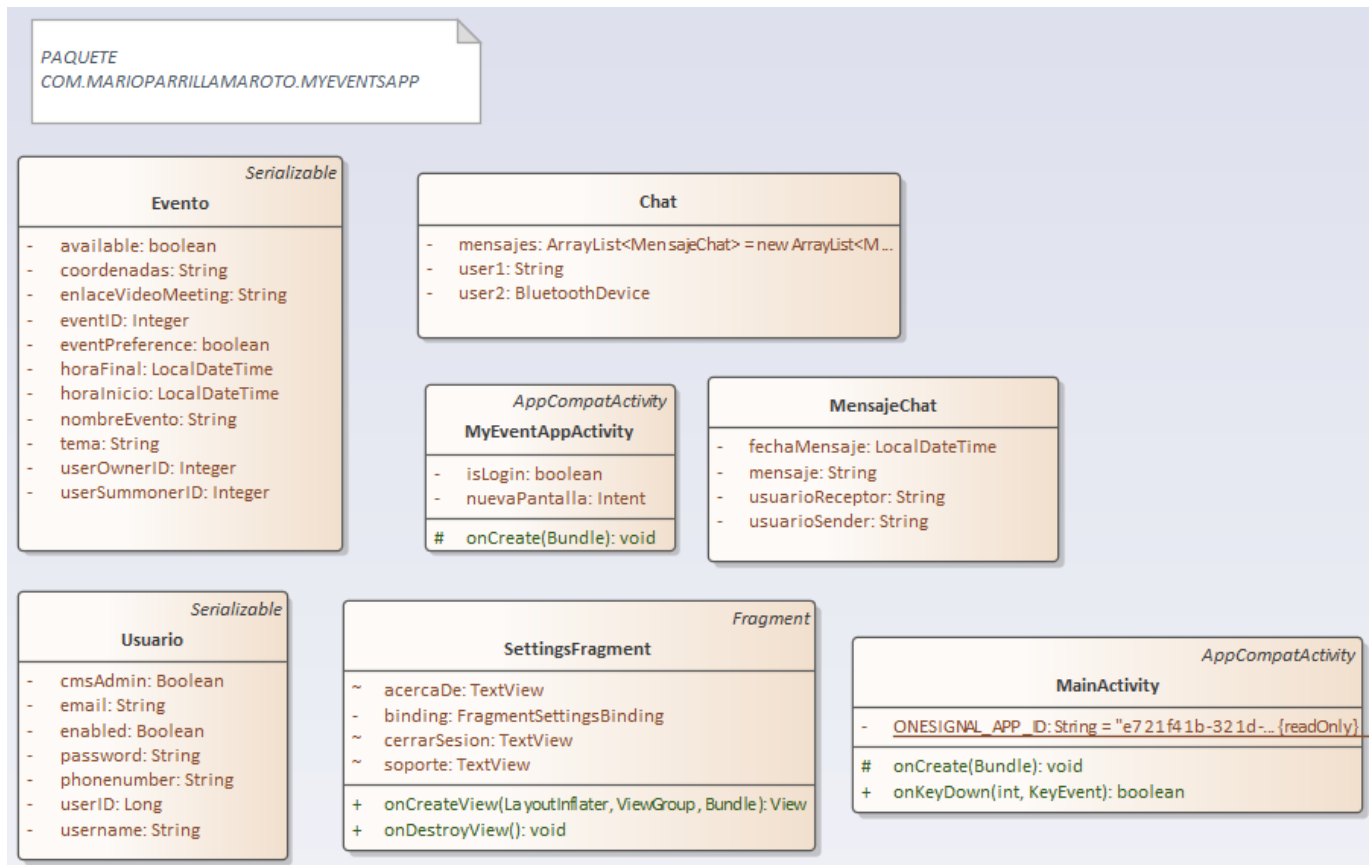


Diagrama de clases



Clase Evento: En esta clase se usará para trabajar con los eventos de la aplicación.

Clase Usuario: En esta clase se usará para trabajar con los usuarios de la aplicación.

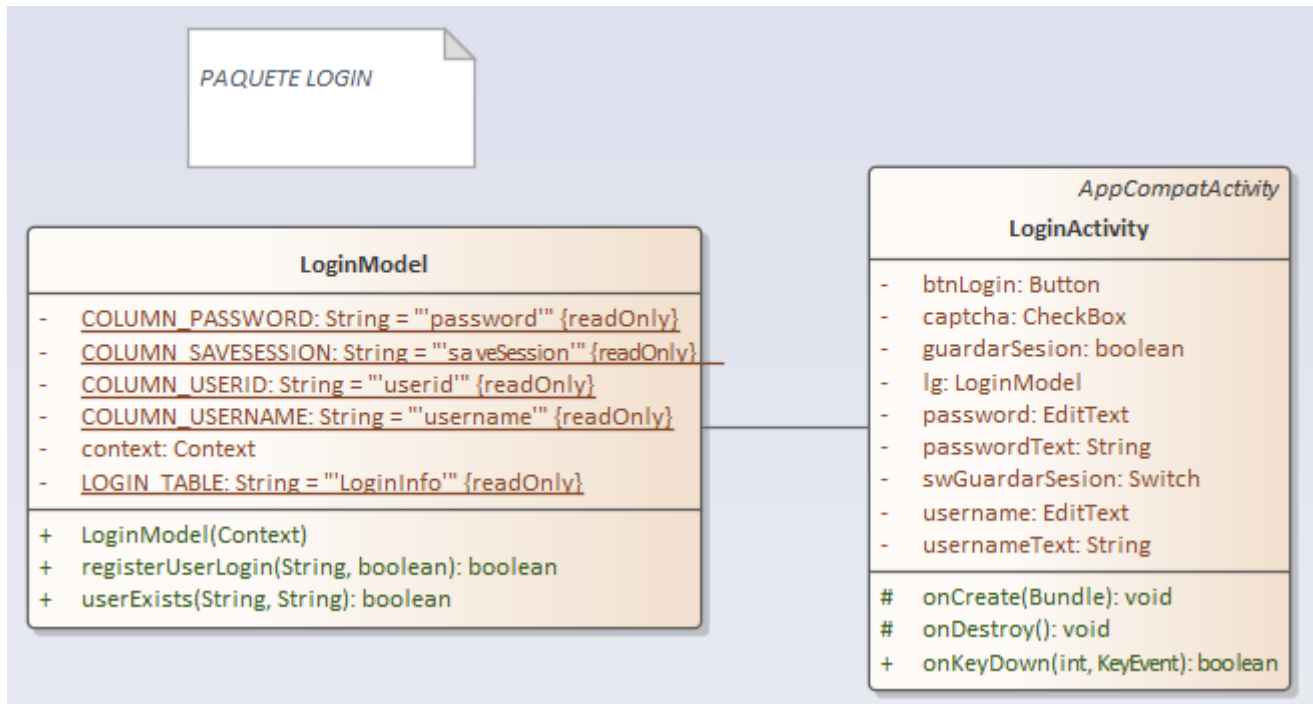
Clase Chat: Esta clase se usa para trabajar con la información de los chats en la aplicación.

Clase MensajeChat: Esta clase se utiliza para trabajar con la información de cada mensaje de los diferentes chats de la aplicación.

Clase SettingsFragment: Esta clase trabaja con la vista de ajustes de la aplicación.

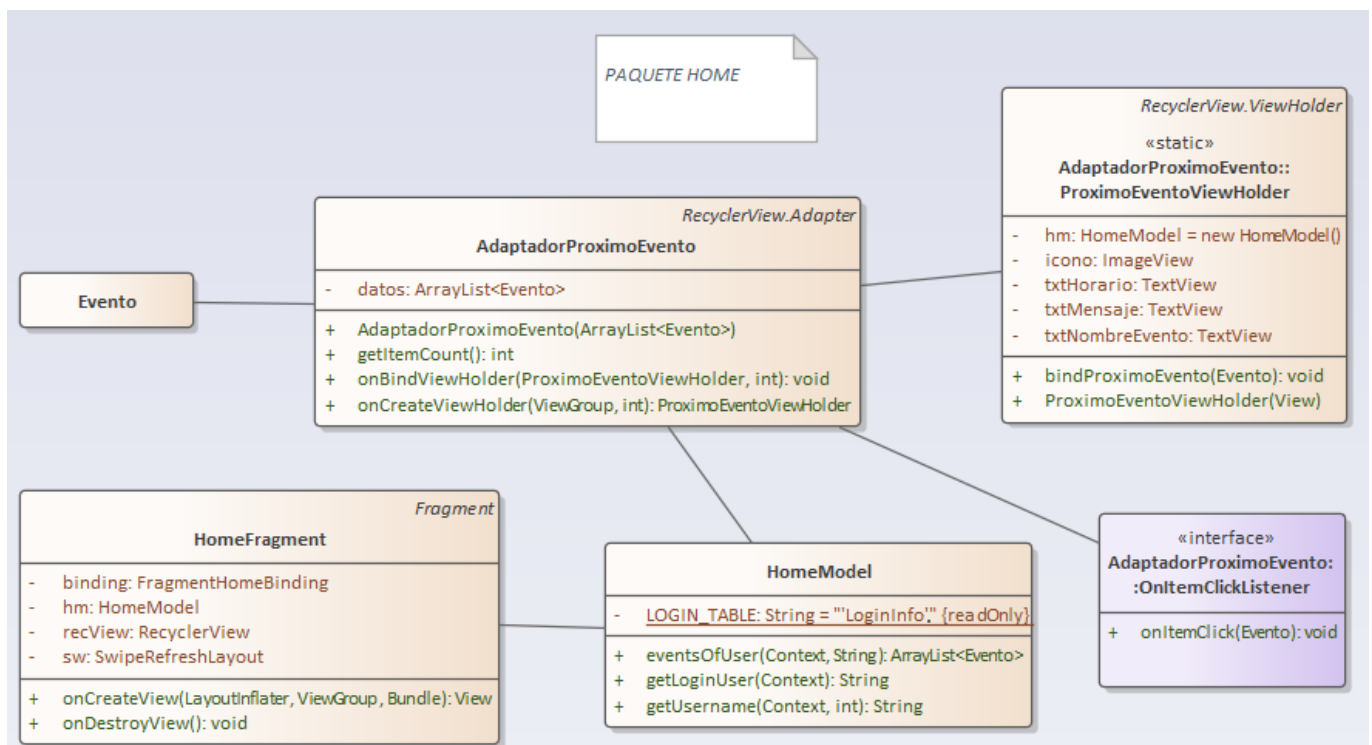
Clase MainActivity: Esta clase es la clase donde se maneja los fragmentos en la primera carga de la aplicación.

Clase MyEventAppActivity: Esta es la clase principal de la aplicación donde se manejarán la primeras cargas de datos desde el servidor cms y se realizarán varias comprobaciones como comprobar si hay un usuario logueado actualmente.



Clase LoginActivity: Esta clase se encarga de realizar las comprobaciones relacionadas con el logueo en la aplicación de los usuarios.

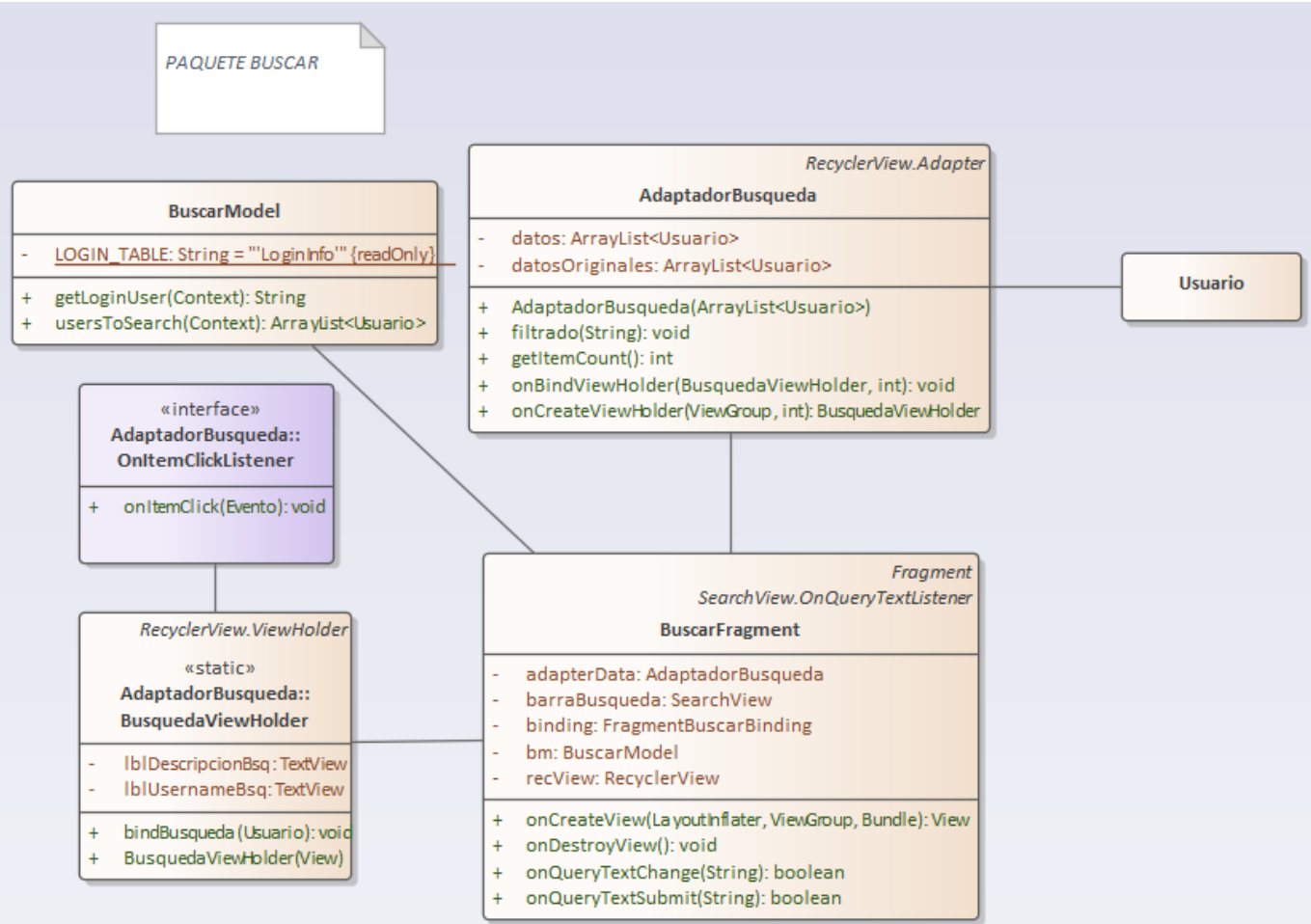
Clase LoginModel: Esta clase se encarga de recuperar datos para pasarselos al LoginActivity para que pueda realizar sus operaciones correctamente.



Clase Homefragment: Esta clase se encarga de cargar los eventos los cuales el usuario logueado tiene citados.

Clase Homemodel: Esta clase se encarga de recuperar los datos para homefragment para que trabaje correctamente.

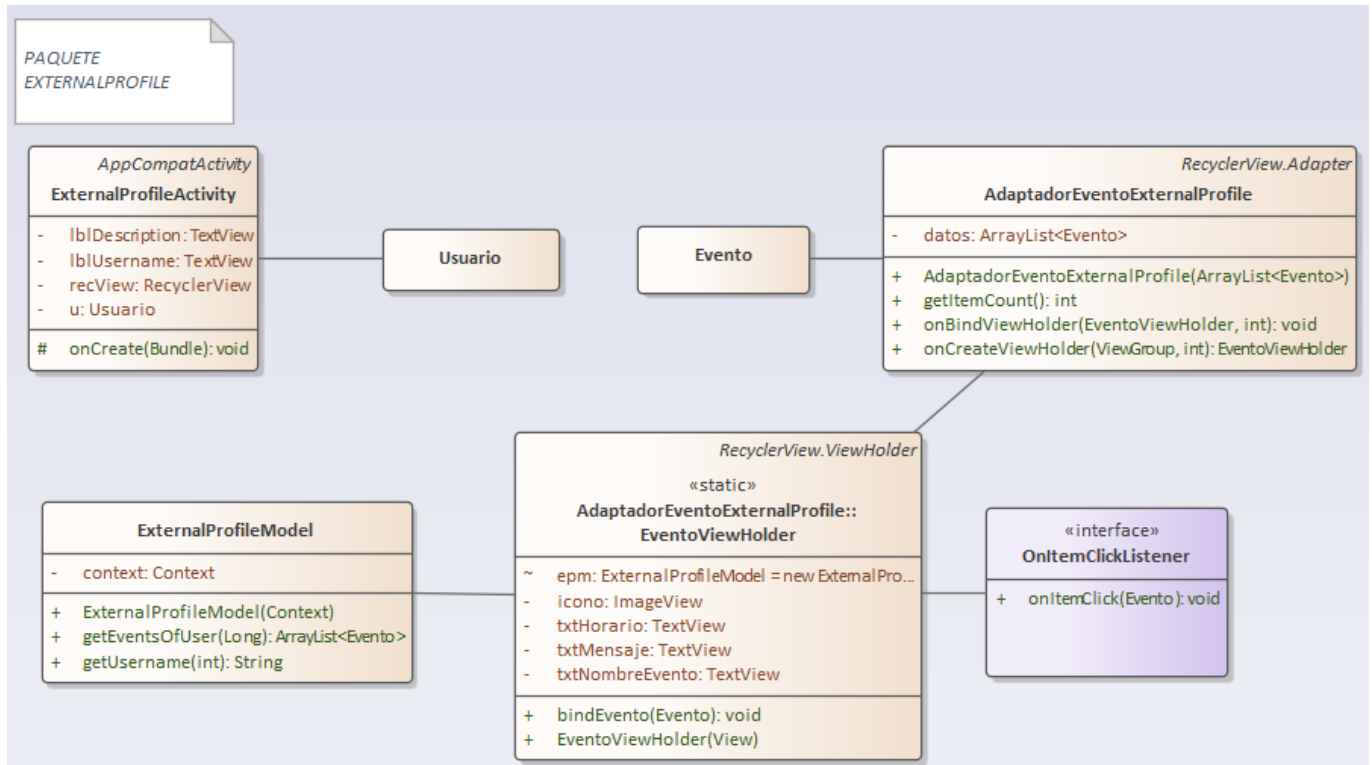
Clase Adaptadorproximoevento: Esta clase se encarga de cargar el layaout y los datos de los eventos en la vista.



Clase Buscarfragment: Esta clase se encarga de buscar los usuarios de la plataforma para poder ver sus perfiles para citar sus eventos.

Clase Buscarmodel: Esta clase proporciona los datos a buscarfragment.

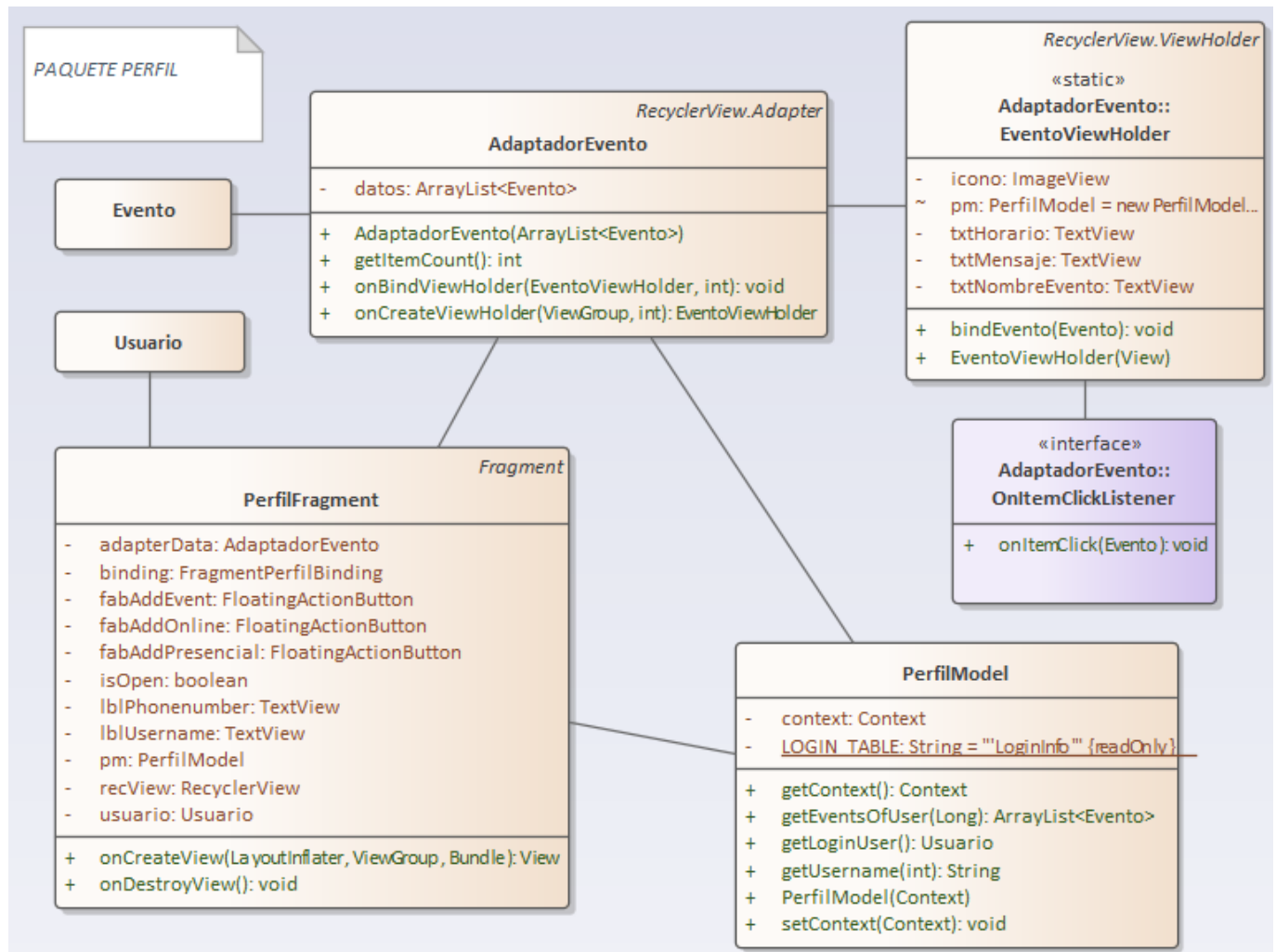
Clase Adaptadorbusqueda: Esta clase se encarga de cargar el layaout y los datos lo de los usuarios en la vista.



Clase ExternalProfileActivity: Esta clase se encarga de mostrar los datos y eventos disponibles del perfil del usuario buscado en el fragmento de buscar .

Clase ExternalProfileModel: Esta clase se encarga de pasar los datos del usuario buscado a la clase ExternalProfileActivity para su correcta funcionalidad.

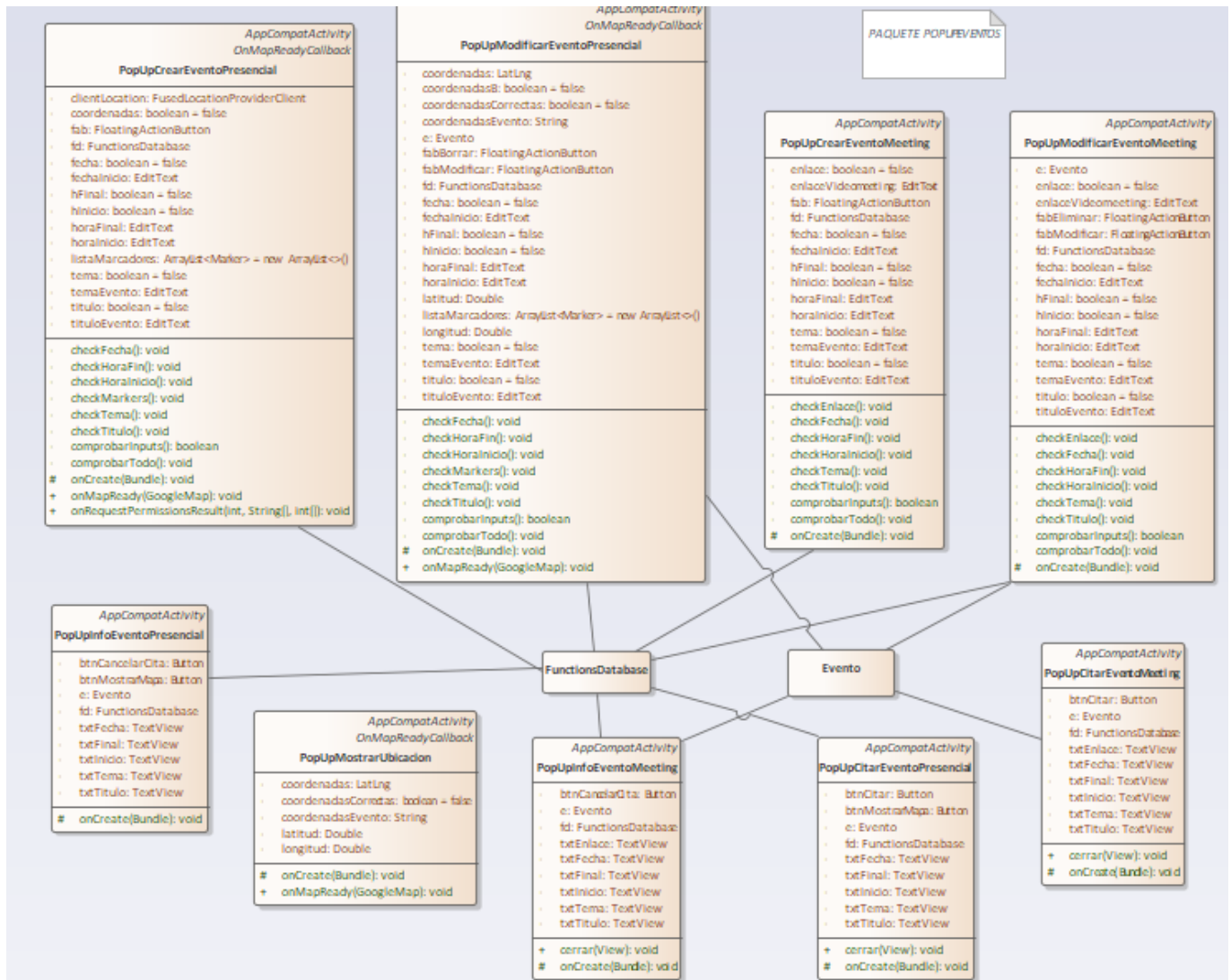
Clase AdaptadorEventoExternalProfile: Esta clase se encarga de cargar el layout y los datos del usuario y de sus eventos disponibles.



Clase Perfilfragment: Esta clase se encarga de cargar el perfil del usuario el cual esta logeado en la aplicacion y poder llamar a los popups para agregar, modificar y eliminar sus propios eventos.

Clase Perfilmodel: Esta clase proporciona los datos del usuario logeado al perfilfragment.

Clase Adaptadorevento: Esta clase se encarga de cargar el layout y los datos del usuario logeado y sus eventos.



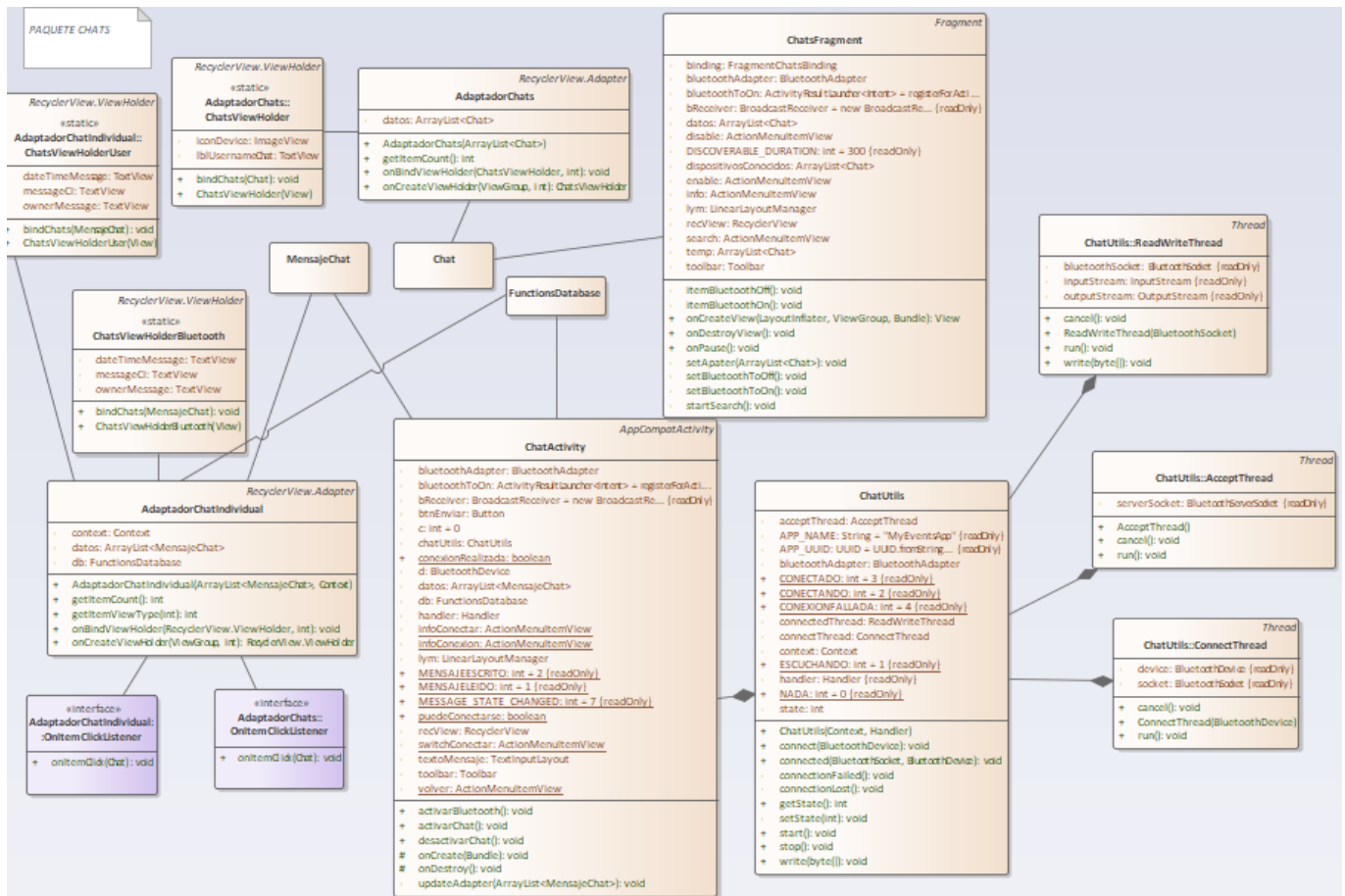
Clases PopUpCrearEventoPresencial y PopUpCrearEventoMeeting: Estas clases se encargan de crear un evento propio, verificando que los datos estén correctamente puestos.

Clases PopUpModificarEventoPresencial y PopUpModificarEventoMeeting: Estas clases se encargan de modificar un evento propio, verificando que los datos estén correctamente puestos y si este evento ya está citado, no se permitirá modificarlo, además de poder eliminar estos eventos.

Clases PopUpCitarEventoPresencial y PopUpCitarEventoMeeting: Estas clases se encargan de mostrar la información de un evento, y permitir al usuario citador citar el evento.

Clases PopUpInfoEventoPresencial y PopUpInfoEventoMeeting: Estas clases se encargan de mostrar la información de un evento.

Clase PopUpMostrarUbicación: Esta clase se encarga de mostrar un mapa en el cual se mostrará la ubicación del evento.



Clase Chatsfragment: Esta clase se encarga de utilizar el bluetooth del smartphone y poder acceder a los chats individuales con los dispositivos bluetooth.

Clase Adaptadorchat: Esta clase se encarga de cargar el layout y los datos de los dispositivos bluetooth encontrados.

Clase ChatActivity: Esta clase se encarga de cargar el historial del chat del dispositivo, realizar una conexion bluetooth con este dispositivo y poder realizar una conversación con este.

Clase AdaptadorChatIndividual: Se encarga de cargar los layouts correspondientes con sus datos de los mensajes de la conversación.

Clase Chatutils: Se encarga de las conexiones, informar de los estados de las conexiones y cambios de la conexion y enviar y recibir los mensajes del chat.

PAQUETE CORE

CoreFuntions

- SECRET: String = "MyEventsApp" {readOnly}
- SecretKey: String = "6LfdeBMdAAAAA... {readOnly}
- SiteKey: String = "6LfdeBMdAAAAA... {readOnly}
- + antiSQL(String): String
- + checkConnetionToInternet(Context): boolean
- + checkGatcha(Context, CheckBox): void
- + checksRooted(Context): boolean
- + createJWT(): String
- + handleCaptchaResult(Context, String, CheckBox): void
- + makeBackupChat(String, String, ArrayList<MensajeChat>, Context): void
- + makeBackupChatCypto(String, String, ArrayList<MensajeChat>, Context): void
- + readBackupChat(String, String, Context): ArrayList<MensajeChat>
- + readBackupChatCypto(String, String, Context): ArrayList<MensajeChat>

FunctionsDatabase

- COLUMN_AVAILABLE: String = "available" {readOnly}
- COLUMN_CMS_ADMIN: String = "cms_admin" {readOnly}
- COLUMN_COORDINATES: String = "coordinates" {readOnly}
- COLUMN_EMAIL: String = "email" {readOnly}
- COLUMN_ENABLED: String = "enabled" {readOnly}
- COLUMN_END_TIME: String = "end_time" {readOnly}
- COLUMN_EVENT_NAME: String = "event_name" {readOnly}
- COLUMN_EVENT_PREFERENCE: String = "event_preference" {readOnly}
- COLUMN_EVENTID: String = "eventid" {readOnly}
- COLUMN_PASSWORD: String = "password" {readOnly}
- COLUMN_PHONENUMBER: String = "phonenumbr" {readOnly}
- COLUMN_SAVESESSION: String = "saveSession" {readOnly}
- COLUMN_START_TIME: String = "start_time" {readOnly}
- COLUMN_TEMA: String = "tema" {readOnly}
- COLUMN_USER_OWNER_USERID: String = "user_owner_us... {readOnly}
- COLUMN_USER_SUMMONER_USERID: String = "user_summoner... {readOnly}
- COLUMN_USERID: String = "userid" {readOnly}
- COLUMN_USERNAME: String = "username" {readOnly}
- COLUMN_VIDEOMEETING: String = "videomeeting" {readOnly}
- contextRoot: Context
- db: SQLiteDatabase
- EVENTO_TABLE: String = "evento" {readOnly}
- LOGIN_TABLE: String = "LoginInfo" {readOnly}
- URLAPI: String = "https://myeven... {readOnly}
- USUARIO_TABLE: String = "usuario" {readOnly}
- + checkCloseSession(): void
- + checksLogin(): boolean
- + checkSession(): boolean
- + checkUserLoginExists(): void
- + citeEvent(Evento): void
- + closeSession(): void
- + createEvent(Evento): void
- + deleteEvent(Long): void
- + FunctionsDatabase(Context)
- + getDb(): SQLiteDatabase
- + getIDLoginUser(): Long
- getUserByID(Long): Usuario
- + getUsernameLoginUser(): String
- + getValue(int): Boolean
- + insertEventDatabase(Evento): boolean
- + insertUserDatabase(Usuario): boolean
- + modifyEvent(Evento): void
- + onCreate(SQLiteDatabase): void
- + onUpgrade(SQLiteDatabase, int, int): void
- + setDb(SQLiteDatabase): void
- + synchronizingData(): void

Clase FunctionsDatabase: Esta clase se encarga de interactuar con la base de datos en el smartphone y realizar sus respectivas acciones. Esta clase se utiliza en clases externas al paquete para que esta clase le envíe la información que necesiten de la base de datos.

Clase Corefunctions: Esta clase se encarga de varias funcionalidades importantes de la aplicación como prevenir la inyecciones SQL, comprobar si el smartphone está rooteado...

Diagrama E/R

TABLA -- USUARIO

Todos los datos de esta tabla son Not Null, no pueden ser nulos

userID: Es la clave principal de la tabla, que será el identificador del usuario, que es de tipo bigint.

username: Es el nombre de usuario, que es de tipo varchar.

email: Es el correo electrónico del usuario, que es de tipo varchar.

password: Es la contraseña del usuario, que es de tipo varchar.

phonenumbr: Es el número de teléfono del usuario, que es de tipo varchar.

enabled: Nos servirá para saber si el usuario puede utilizarse o no, que es de tipo bit que en verdad es un boolean.

TABLA -- EVENTO

eventID: Es la clave principal de la tabla, que será el identificador del evento, que es de tipo bigint. NotNull.

eventname: Es el nombre de evento, que es de tipo varchar. NotNull.

eventname: Es el tema del evento, que es de tipo varchar. NotNull.

start_time: Es la fecha y hora del inicio del evento, que es de tipo Datetime. NotNull.

end_time: Es la fecha y hora de final del evento, que es de tipo Datetime. NotNull.

event_preference: Con este dato, segun su valor, si es 0 será un evento presencial y si no, será un evento meeting y con esto trabajaremos con diferentes datos según este valor, es de tipo bit, que en realidad es un boolean. NotNull.

coordinates: Son las coordenadas de la localizacion del lugar de quedada del evento, es de tipo varchar.

videomeeting: Es el enlace de la videoconferencia del evento, es de tipo varchar.

available: Con este dato sabremos si el evento esta activado o no según su valor (0 = desahabilitado / 1 = habilitado), es de tipo bit, pero en realidad es de tipo boolean. NotNull.

user_owner_id_user: es la id del usuario que ha creado el evento, es de tipo bigint. NotNull.

user_summoner_id_user: es la id del usuario que ha citado al creador el evento, es de tipo bigint.

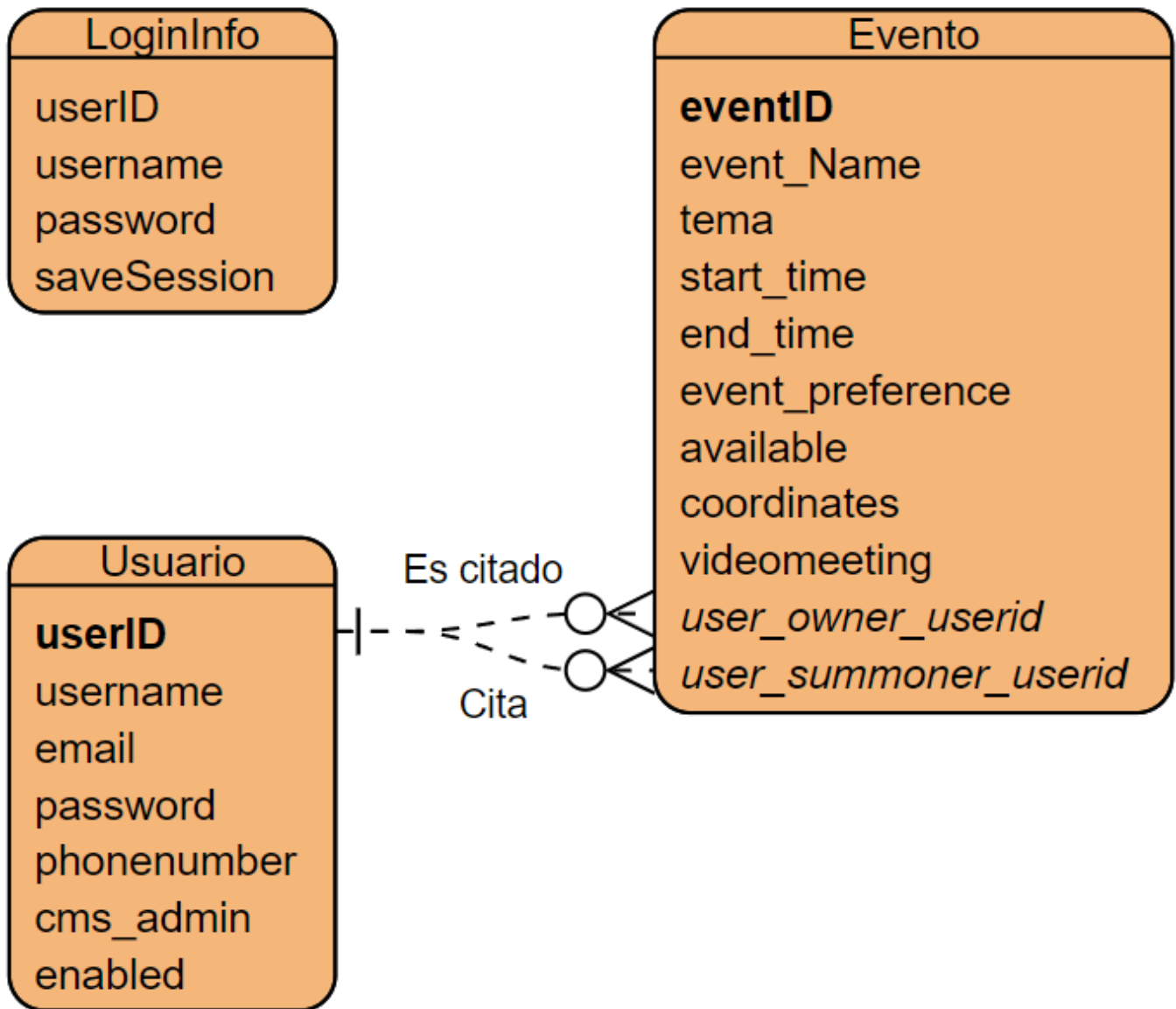
TABLA -- LOGININFO

userid: Es el identificador del usuario que esta logueado en la app, es de tipo integer.

username: Es el nombre del usuario logueado en la app, es de tipo text.

password: Es la contraseña del usuario que esta logueado en la app, es de tipo text.

saveSession: Es el estado de si ha elegido guardar sesion a la hora del logueo, es de tipo bool.



Plan de pruebas

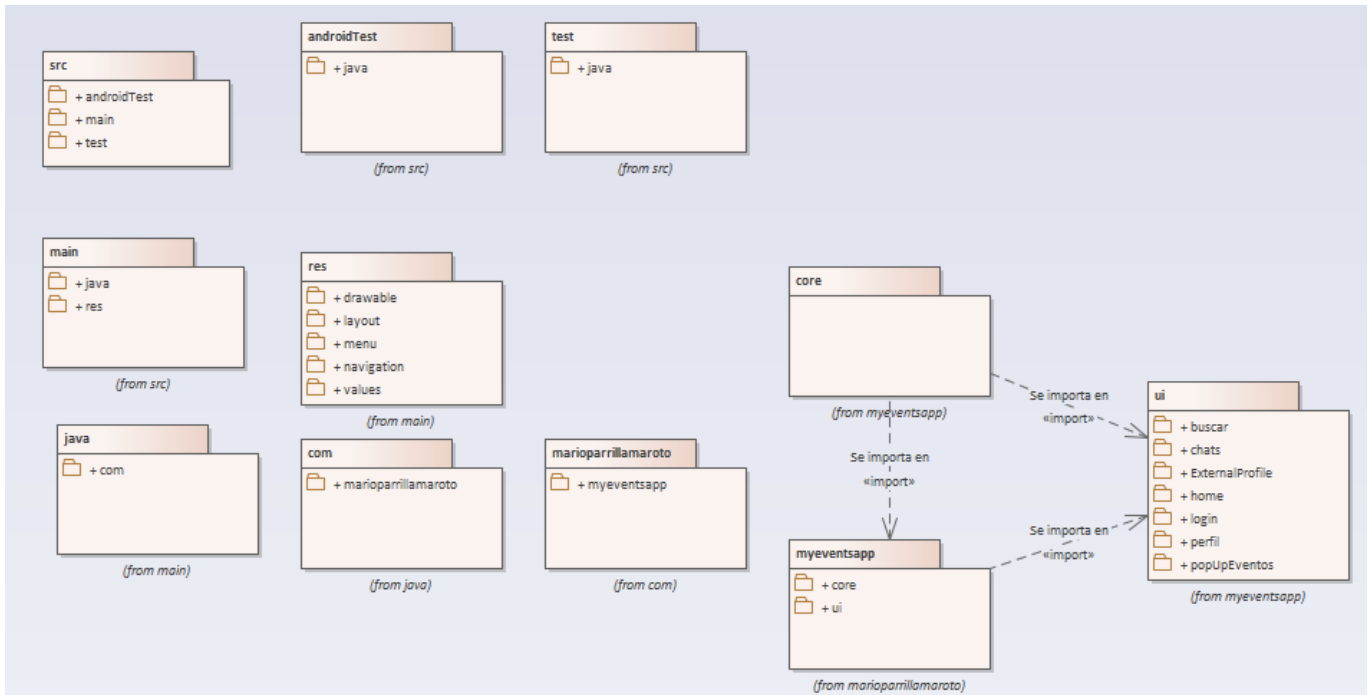
Prueba 1: Comprobar Existencia del un Usuario en el login: Se pasaran como parametros un usuario y una contraseña y si el usuario existe, nos devolverá un true, si no, nos devolverá false.

Prueba 2: Comprobar usuario agregado: Se pasará un objeto usuario y si el usuario se agrega correctamente, nos devolverá un true, si no, nos devolverá un false.

🔒 ⚠️ PLAN DE PRUEBAS EN DESARROLLO ⚠️ 🔒

Diseño Técnico

Diagrama de paquetes y de componentes



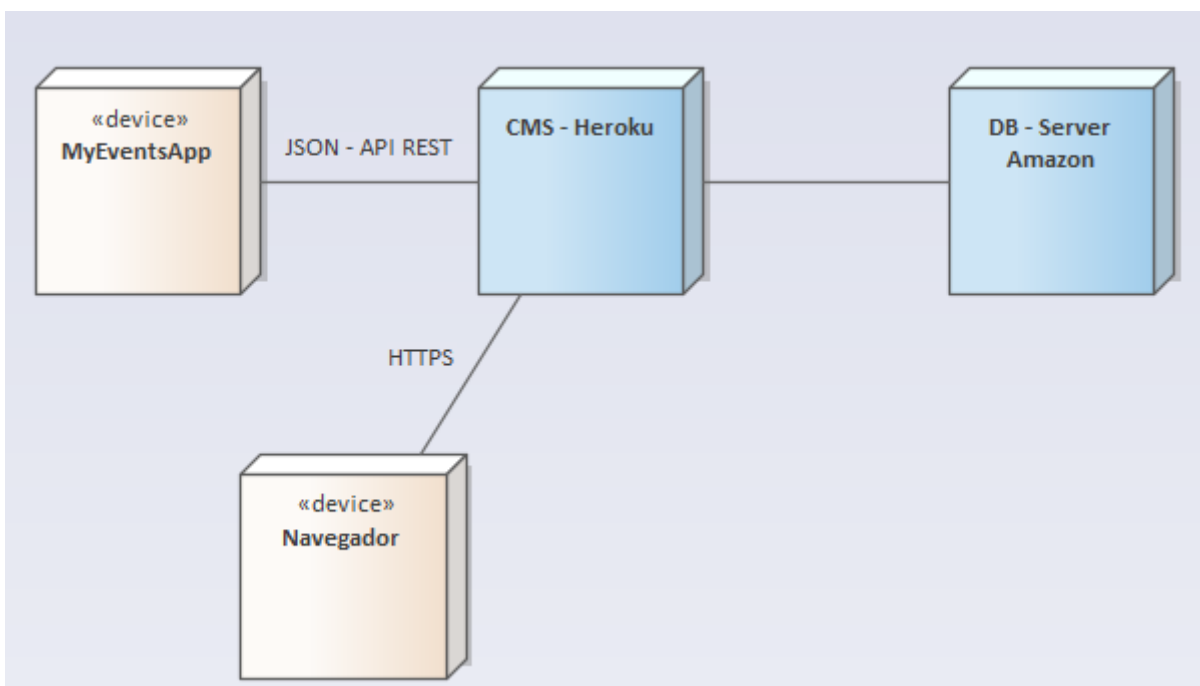
Paquete res: En este paquete se almacenan diferentes paquetes donde se almacenan archivos importantes como los layouts, los drawables o valores entre otras cosas...

Paquete MyEventsApp: En este paquete encontraremos las clases de objetos como las que se guardan en la bd, otros objetos que se usan a lo largo de la aplicación y las clases de funcionalidad en la carga principal.

Paquete Core: En este paquete se almacenan las clases las cuales contienen las funcionalidades mas importantes como conexiones con la base de datos, conectar con la api, funciones de seguridad, ...

Paquete UI: En este paquete se almacenan paquetes que contienen las clases que manejan las vistas de la aplicación.

Arquitectura del sistema



CMS: El CMS esta desplegado en la plataforma de Heroku. Este CMS esta desarrollado con Spring boot.

MyEventsApp: Esta es la aplicación android desarrollada con java.

Navegador: Se trata de un cliente web como chrome, firefox...

Base de datos: La base de datos esta hosteada en amazon ya que se utiliza un plugin en heroku que te la hostea ahí.

Entorno de desarrollo, librerías y servicios

En este punto se explicarán las diferentes tecnologías utilizadas para la realización del proyecto, así como los elementos más importantes que permitan entender el funcionamiento del sistema.

Google Maps: Esta aplicación para cargar los mapas, utiliza una api de google maps.

One Signal: Permite mandar notificaciones push a los clientes.

Bcrypt: Permite cifrar las contraseñas de los usuarios para que sus datos se almacenen seguros.

SQLite: Es el gestor de base de datos que se usa para guardar los datos de forma local.

SQL Cipher: Es una variante de MySQLite, la cual nos permite crear y almacenar los datos cifrados.

Safetynet: No permite realizar funcionalidades de seguridad en nuestra aplicación como la de agregar captchas.

Rootber: Nos permite identificar si el dispositivo esta rooteado.

Volley: Es una libreria de Google, que nos permite realizar peticiones a la api rest del CMS.

Conceal: Es una librería de facebook, la cual nos permite cifrar y descifrar archivos y cadenas de texto.

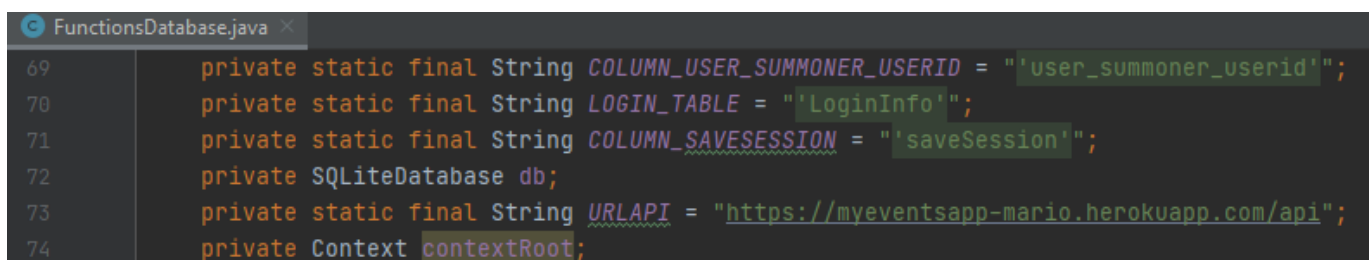
JWT: Esta tecnologia de web token, nos permite poder realizar una peticiones autorizadas a la api rest del CMS.

Instrucciones para la compilación, ejecución y despliegue de la aplicación

Para poder loguearte debes usar el usuario rogelio con la contraseña admin

En el dispositivo android se deberá tener la versión android minima Android Oreo 8.1 (SDK 26).

Si quierse cambiar la url de la api para recibir y enviar datos tendrás que hacerlo aqui:

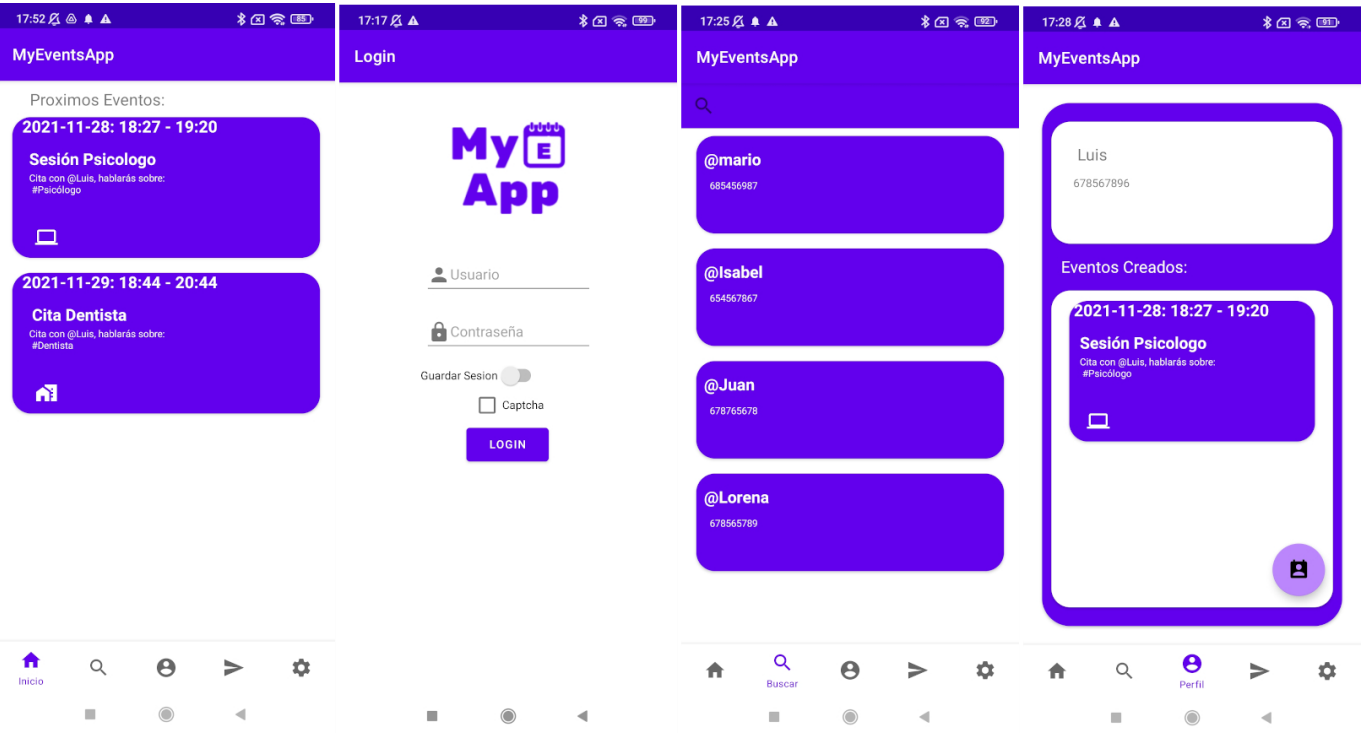


```
FunctionsDatabase.java
69 private static final String COLUMN_USER_SUMMONER_USERID = "user_summoner_userid";
70 private static final String LOGIN_TABLE = "LoginInfo";
71 private static final String COLUMN_SAVESESSION = "saveSession";
72 private SQLiteDatabase db;
73 private static final String URLAPI = "https://myeventsapp-mario.herokuapp.com/api";
74 private Context contextRoot;
```

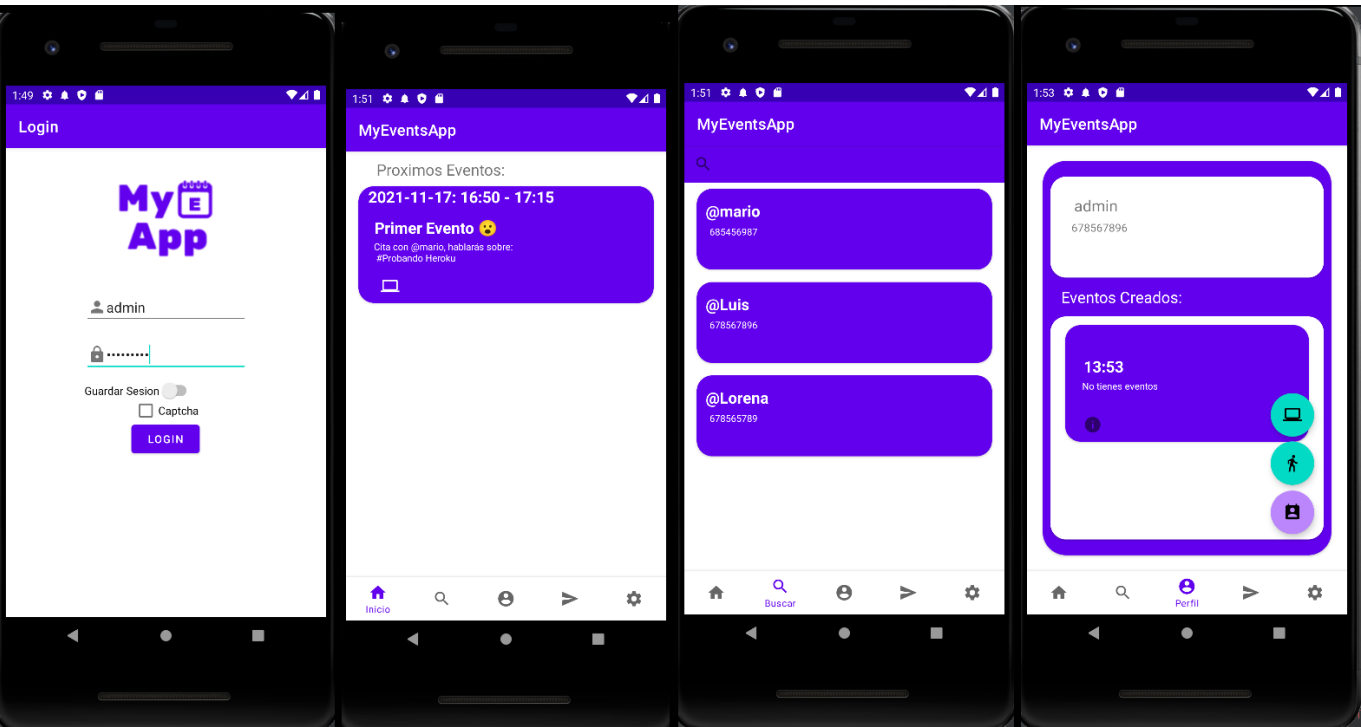
Informe de pruebas

Pruebas en emuladores y dispositivos reales

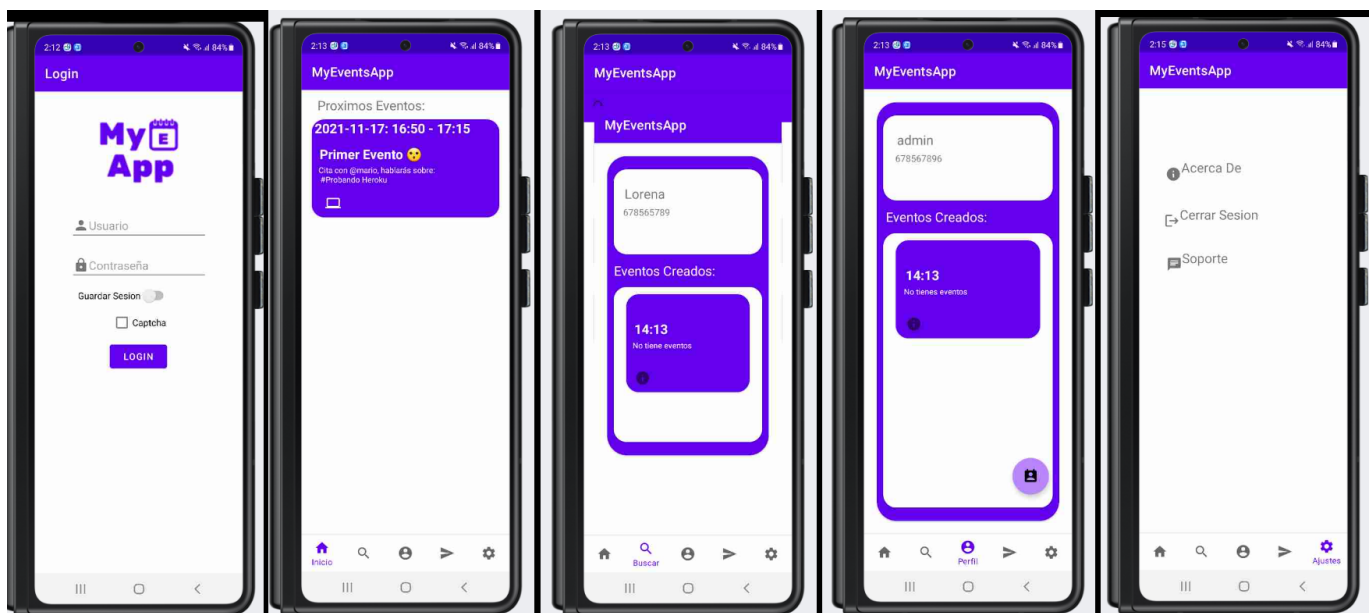
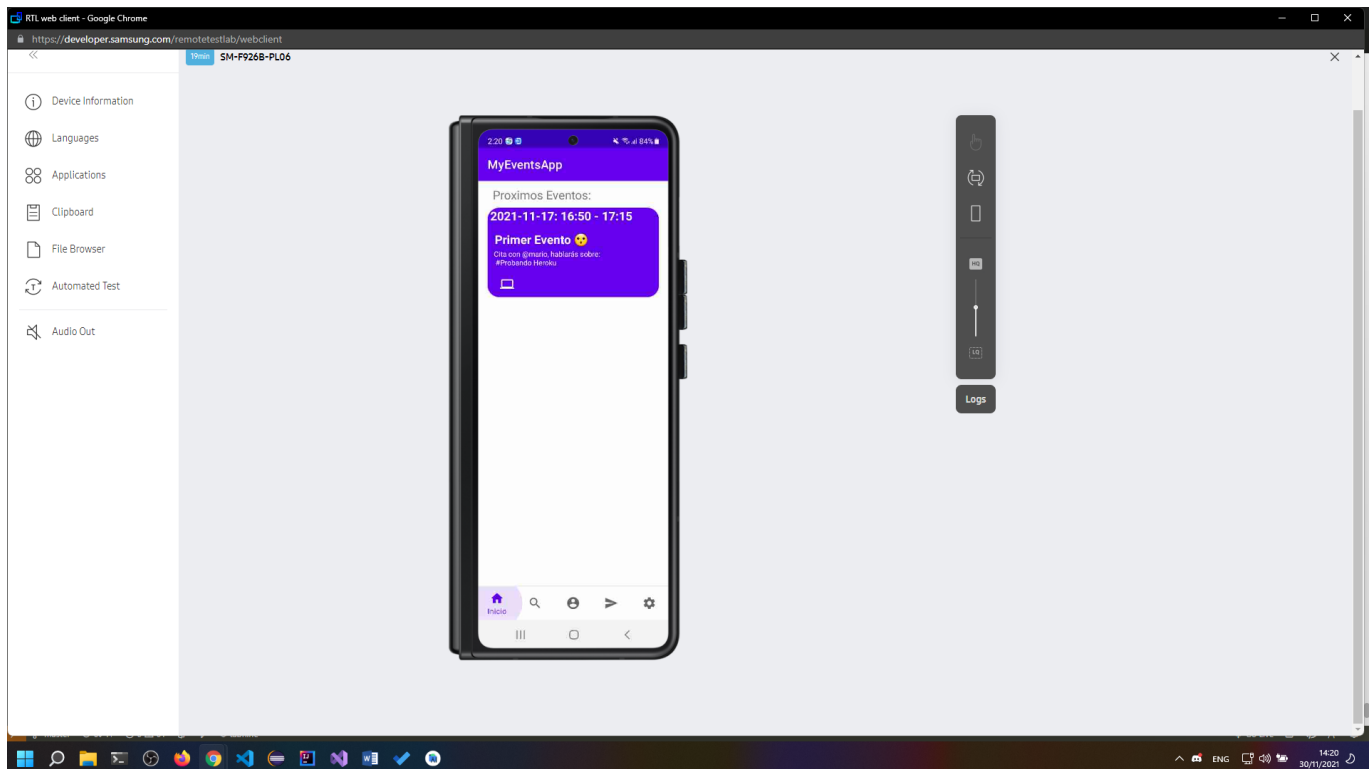
Dispositivo Real:



Emulador:



Pruebas remotas



Conclusiones

Conocimientos adquiridos

En este trabajo, he tenido que desarrollar una aplicación la cual tenía que cumplir gran variedad de competencias las cuales unas eran más fáciles que otras, pero en el transcurso del desarrollo, he aprendido bastante de cómo funcionan por dentro las aplicaciones Android, de cómo recibir datos de una API, de cómo almacenar datos en una base de datos en Android, de utilizar mapas, ... entre otras cosas. En el transcurso del desarrollo de la aplicación he tenido que ir cambiando funcionalidades como la creación de los eventos como podemos ver si comparamos el wireframe con la versión final. Uno de los problemas que me encontré a la hora de desarrollar la aplicación fue, que al encriptar la base de datos de la aplicación, la aplicación tardaba mucho en cargar las pantallas, así que tuve que quitar esta opción de la versión de producción, además otro

problema que he tenido, ha sido que a la hora de recibir datos de la API, al ser peticiones asincronas, tardan en recibirse, pero esto si se llegara a seguir el desarrollo de la aplicación se pondría un solución pero al tener un tiempo limitado cercano, he tenido que dejarlo como esta.

En conclusión, gracias a este desarrollo he aprendido bastante a manejar un desarrollo con un tiempo limitado como si fuera un proyecto real de empresa, ademas de desarrollar aplicaciones android entre otras cosas.

Mejoras futuras

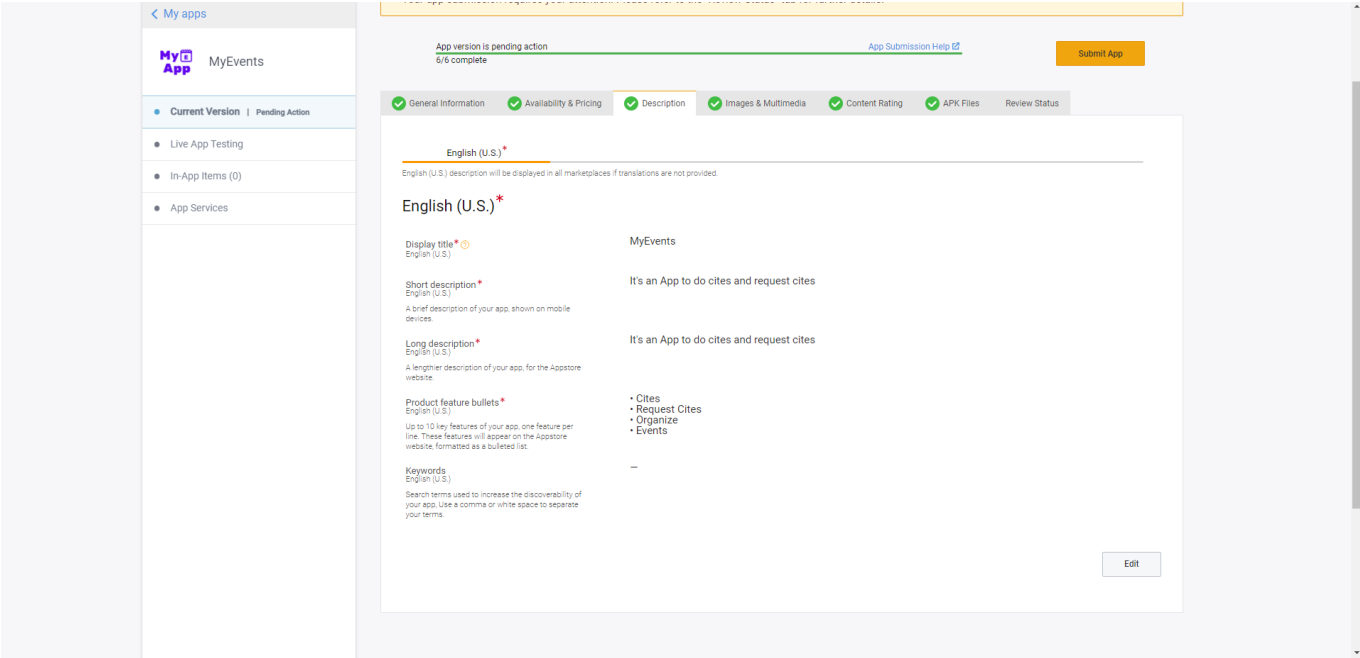
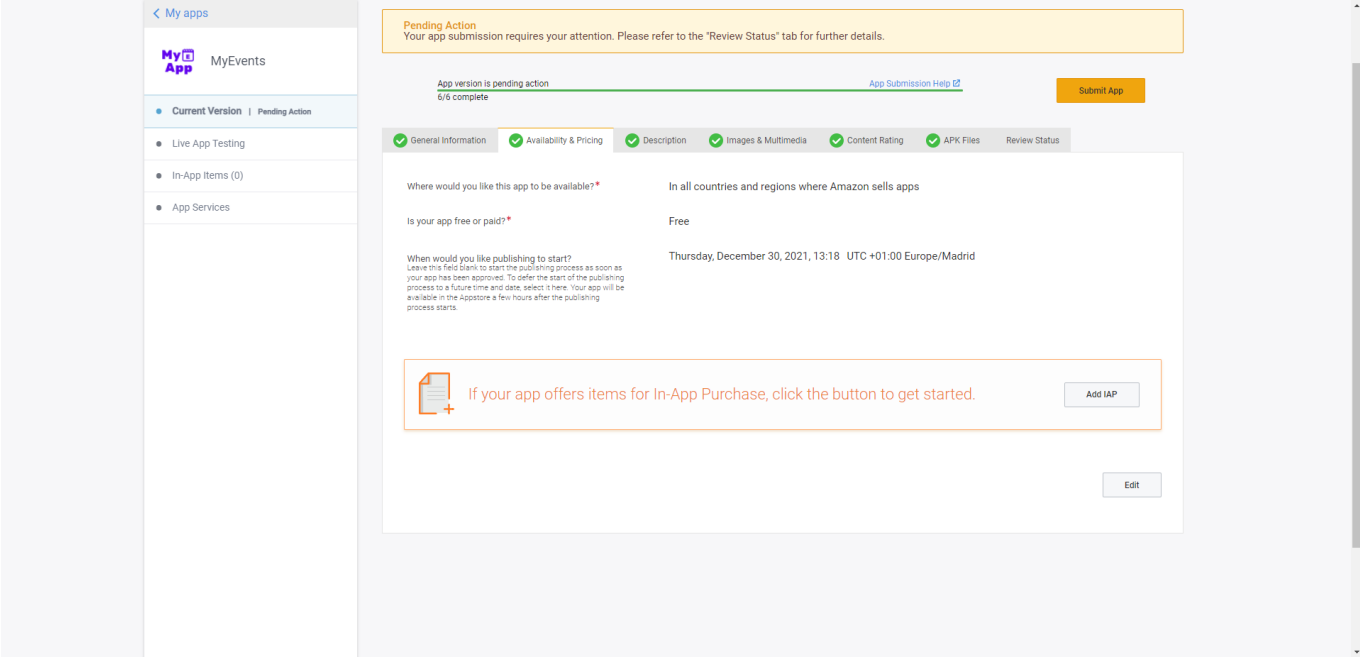
- Incluir pantallas de registro en la app.
- Incluir pantallas y la opcion de recuperar una contraseña olvidada.
- Incluir chat online y eliminar chat bluetooth.
- Incluir imagen de perfil.
- Mejorar el recibimiento de respuestas de la API

Publicacion de la APP

The screenshot shows the 'My apps' page for 'MyEvents' in the Amazon App Store. The app is in a 'Pending Action' state, indicating it is pending review. The submission progress bar shows 6/6 complete. The submission details are as follows:

- App title:** MyEvents
- App SKU:** com.marioparrillamaroto.myeventsapp
- App Submission API Keys:**
 - App ID:** amzn1.devportal.mobileapp.bf088a37545d41e09c2ab057532f0ee9
 - Release ID:** amzn1.devportal.aprelease.50166b81cc6146a7ab31bba77f65062
- App category:** Business > Meetings & Conferencing
- Customer support contact:**
 - Customer support email address:** marioparrilla2@gmail.com
 - Customer support phone:** —
 - Customer support website:** —

The interface includes a 'Submit App' button and an 'Edit' button. A 'Pending Action' banner at the top states: 'Your app submission requires your attention. Please refer to the "Review Status" tab for further details.'



Edit

Altavoces (Realtek(R) Audio): 0%

< My apps

MyApp

MyEvents

Current Version

Pending Action

Live App Testing

In-App Items (0)

App Services

Pending Action

Your app submission requires your attention. Please refer to the "Review Status" tab for further details.

App version is pending action

6/6 complete

App Submission Help

Submit App

General Information

Availability & Pricing

Description

Images & Multimedia

Content Rating

APK Files

Review Status

Your Last Activity

Successfully submitted app submission on 30 Nov 2021.

Action Required

Issues or observations were found in your app submission, please click on Details for further information.

Details

Submission History

| | | | |
|-----------------------------|---------------------------|--|---------|
| 03:01 AM PST 01 Dec 2021 | Pending Action | Issues or observations were found in your app submission, please click on Details for further information. | Details |
| 03:19 AM PST 30 Nov 2021 | App Submission Successful | Your app submission was successfully submitted for review. | |
| 08:32 PM PST 29 Nov 2021 | Pending Action | Issues or observations were found in your app submission, please click on Details for further information. | Details |
| 11:38 PM PST 28 Nov 2021 | App Submission Successful | Your app submission was successfully submitted for review. | |
| 08:53 PM PST 28 Nov 2021 | Pending Action | Issues or observations were found in your app submission, please click on Details for further information. | Details |
| 09:38 AM PST 27 Nov 2021 | App Submission Successful | Your app submission was successfully submitted for review. | |