POLITECNICO
MILANO 1863

Book a Place

Software Design Document

Pardeep Kumar, Enrico Sarneri

October 12, 2023

Contents

1 Introduction

1.1 About the Design Document

In this section we want to briefly introduce our application.

This application has been developed for the course "Design and Implementation of Mobile Application" at "Politecnico di Milano".The goal of the course is to present the main techniques and technologies to design and implement applications for mobile and wearable devices. In particular, the course will address the design of "mobile" applications by considering both the problem of designing the user experience and the problem of understanding the actual distribution of the components that constitute the application and their interactions. This document illustrates the decisions and choices we made in order to accomplish these goals.

This Software Design Document provides the documentation that will be used as a overall guidance to the architecture of the software project. We are going to provide a documentation of the software design of the project, including use case models, class and sequence diagrams.

The purpose of this document is to provide a full description of the design of *Book a Place*, showing insights into the structure and the design of each component.

1.2 Platform

We decided to develop the application with Flutter, a Google mobile app SDK. The main reasons why we decided to choose it are the following:

- It allows the development of Cross-Platform applications, offering the same UI and Business Logic in all platforms. In particular, Flutter doesn't need any platform-specific UI components to render it's UI. The only thing it needs to show the application UI is a canvas to draw onto. Its way of rendering eliminates worries about UI consistency on different platforms.
- It uses Dart language, a class-based, single-inheritance, object-oriented language with a C-style syntax, which we are already familiar with.

1.3 Choice of Application

The need of having well-organized events with a distributed visibility and a controlled access, is a necessary requirement in the nowadays society, in particular after the COVID-19 pandemic. The applications gives the possibility to the owners of different types of entertainment places to create and organize events, with the fundamental feature of checking the access effectivness through the scanning of Qr codes.

The created events can be both public and private in order to give the possibility of participation to different people. Either way, the organizer keeps the right of accepting or refusing the requests of participation, so as to guarantee a centrality in the control.

The idea for the application came from personal and third-party reported experiences. Too often we face disorganized events or locals' entrances where the management is completely random and inefficient. This could have bad consequences for the local or the setting itself in terms of feedback and possible future clients. Furthermore, with the coming of the pandemic, it has become necessary to control the accesses and the reservations guaranteeing a correct flux also with the respect of the laws.

1.4 Risk Analysis

During the problem analysis, have been identified some risks that can compromise the correct development of the project.

We had to be really careful during the requirements collection phase, since the requirements must be clear in order to avoid delays due to misunderstandings. Moreover, formal modeling of the application allows the reader not to be confused by it and to learn requirements in a clear way.

Another possible delay was learning Dart language and Flutter framework but, in this direction, hot reload feature allows seeing the applied changes almost instantly, without losing the current application state. In this way we were able to optimize the time and to increase development speed.

A last possible delay was learning concepts or techniques in advanced stages of the project, leading to possible code rewriting and inevitable loss of time. However, also in this case, the large Flutter community support and the well-organized documentations and resources have helped a lot.

1.5 Time Constraints

Differently from a project developed for real stakeholders, in this case there were not strict time constraints. However, our target was to deliver the project by the end of the first semester of A.A. 2021/2022.

We started to develop our application at the beginning of the first semester. Developing, testing and creating documentation took roughly 4 months. We started developing in mid-October 2021 and complete in February 2022.

The team is composed by two people with both solid Java and C-like languages knowledge but little experience in Android and mobile app design, thus we had to dedicate some time to learn and be familiar with the environment.

1.6 Stakeholders

The main stakeholders of the project are the professor and the other students that might be attending the presentation. The audience wants to have a clear idea about the project idea and realization. Professor's main goal instead is to check if the concepts taught during the course are clear to us and if we succeeded in implementing them inside our project.

Actually, the release of our application in the digital store is not programmed since we would prefer to implement more functionality and having a more complex and robust application. However, designed it keeping simple and intuitive, suitable for every kind of users. We used English as main language for our project, due to its spread world. If a possible future official release of *Book a Place* we will introduce other major languages to make it more usable.

2 General overview

2.1 Basic idea

Book a Place is a native Android application that provides the possibility to the users to join exclusive events in their neighborhood and also allow the owners of locals to create events that other users can easily join. The application make distinction to 2 type of users:

- Normal user which can look in the events in his neighborhood and ask to participate to that specific event or share it with their friends.
- Owner user can do everything that a normal user can do but has in addition the possibility to add new owned locals, create its own events and handle them by accepting or refusing new participants.

The application requires users to be authenticated in order to exploit the services provided. Otherwise it would be impossible for the normal users to apply for a certain event since there must be a tracking of the names in order to manage the events themselves and the bookings. The owner user likewise, wouldn't be able to proof the real ownership of a local.

2.2 Core features

Here we list the main screens that can be found in our app and their functionalities:

- Tutorial
    1. Displays a brief introduction of the application showing interactively the main functionalities and services.
- Registration

1. Allow the registration of a new user through a combination of email, password, name, surname and the specification of ownership of any locals .

- Sign in

1. Allow the user to Log in through the combination of email and password.

- Home

1. Main point of interaction of the user. A simple bottom menu shows to the user the main features and screens of the application: Google Map Screen, Share Link, Create Event/My Events and Profile. A set of icons provides the user with insights about his usage of the application.

- Google Map Screen

1. Provides both types of user the possibility to see reported events placed on a map. The events are displayed with a marker having the icon logo of the application.
2. If the user gave permission to access location information, the map is centered in the current position by default. However, after moving to a different place in the map, there's the possibility to come back to the current position through the specific button.
3. Possibility to change the texture view of the map through the specific button.
4. A filter panel, provides the user the possibility to research specific events with different criteria. The user can filter by the number of participants, by the type of setting, by the price of the event, by the distance of the places from its current position and by the date in which the events take place. By default, the events displayed on the map are the all possible starting from the current date. The user then can filter out them. All the filters can be used both singularly or in combination between each other.
5. By clicking on an event marker, user can see more detailed information about the reported event through a sliding up panel. Here are displayed the info concerning the address of the place, the name of the event, the description of it, as well as the number of participants and the price for the entrance. Furthermore, two buttons offer the possibility to share an event through a link copy and to ask for the participation.
6. A search bar is provided in order to easily find and move to a particular location, with auto-complete feature provided by Google.

- Share Link

1. Allow the users to search for an event given the event link. This is the only way to access the private events, since they are not displayed in the google map. The user will be redirected directly to the page of the detailed event where will have the possibility of asking to participate.

- Create Event as Owner User

1. Allows the owner user to create a new event. Different fields are mandatory in order to succeed: name of the event, description of the event, the name of the owned local, the type of the setting, the starting date, the ending date, the maximum number of participants and the cost of the entry. Furthermore, there's the possibility to load an image from the phone, in order to give a better exemplification and a immediate visual catch about what the event will be about.

- My Events as Normal User

1. Allows the user to see the event he's participating in. The list is presented as a carousel list with the different boxes containing the main info about each event. The name, the place and the starting - ending dates. A "More info" button will bring the user to a page with more info regarding that event.

- Profile

1. Here is presented the profile section of each user. Each section is personalized with a box containing the initial letters of the specific user as well as the corresponding email address.
2. A list of buttons is presented based on the typology of the user. The normal user will only see the "My Events" button, that will bring to the same page of "My Events" reached through the bottom menu while the owner user will also see a "Organized Events" button and a "My Locals" button.
3. For every user here is presented the possibility to log out.
4. In this section there's also the possibility to reach again the Tutorial in case of necessity as well as the possibility to send a communication/feedback message to the developers of the app.

- Organized Events
  1. Allows the owner of a local to see his created events as a carousel list of boxes containing the main info.
- My Events
  1. Allows the user see the event he's participating in as a carousel list of boxes containing the main info .
- My Locals as Owner
  1. Allows the owner to see his locals.
- Add Local
  1. Allows the owner to register a new local giving the address of the place, the name of the place and uploading the files that certificate the ownership.
- Event Screen as Owner of that event
  1. Access by clicking on the "More info" button in each box of the "Manager Events" carousel list.
  2. Allows the manager to see the details of the event: the address, the actual number of participants and the description.
  3. It also allows the manager to see the users asking to join the event with the possibility to accept or refuse each application.
  4. Allows the manager to see the actual list of the participating users.
  5. Allows the manager to scan Qr codes. The Qr codes scanning screen will appear after the click on the correspondent button and it allows to check the correct belonging of a certain Qr code to the list of the participants.
  6. Allows the manager to share the event through a unique link.
- Event Screen as Normal user
  1. Access by clicking on the "More info" button in each box of the "My Events" carousel list.
  2. Allows the user to see the details of the event : the address, the actual number of participants and the description.
  3. Allows also the user to share the event and
  4. Allows the user to ask to participate to the event
  5. If the user is already participating to the event he will have a "Show Qr button" that will display his unique Qr, necessary to be showed at the entrance of the event.

2.3 General Qualities

Book a Place offers several characteristics of accessibility and usability. The user interface is easy to use and intuitive, facilitating the process of understanding. Furthermore, the design is eye-catching and captivating to guarantee the user the best possible experience.

- Usability: the main actor in this context is the end user. We decided to develop a user interface that could be as simple as possible and at the same time engaging and compelling. The present functionalities are the basic and the required ones that can guarantee a complete usage of the application, guaranteeing the best possible user experience.
- Nice User Interface: we tried to develop our application in the nicest possible way, following Google's Material Design guidelines in order to have a user-friendly and easy-understandable design.

2.4 Functional requirements

In this section we list the general requirements necessary to the correct behavior of the application:

- General requirements
    1. The application has to be comprehensible by as many people as possible, so we decided to use English as the main language of the application.
    2. The application has to start with a splash screen while the initial settings are loading and the map is set.
    3. The application needs to provide a home screen that allows the Normal User and the Owner User to access all the functionalities.
    4. The application requires an user to be registered and logged in, in order to work.
    5. The application needs to allow new users to create an account.
    6. The application needs to allow to choose between normal and owner user during the registration of an account.
    7. The application needs to allow owner users to create new events.
    8. The application needs to allow normal users to attend the events.
- Tutorial requirements
    1. On first run, the Tutorial has to be displayed introducing the general features of the application to the user.
    2. On first run, once completed, the Tutorial has to redirect the user to the Registration and Login screen.
    3. On following runs, if re-utilized, once completed, the Tutorial has to redirect the user to the Google Map screen.
    4. The Tutorial has to allow the user to navigate through the various screen and/or to skip them.
- Log-in requirements
    1. Log-in activity has to be accessible only to users not currently logged in.
    2. Log-in activity has to guarantee the access only to the users already registered in the database application.
    3. Log-in activity has to effectuate the accessing procedure with the insert of the correct email and password.
    4. Log-in activity has to return an error notification if the credentials are wrong.
    5. Log-in activity has to return the user to the event map screen if the authentication is successful.

- Registration requirements

    1. Registration activity has to be accessible only to users not currently logged in.
    2. Registration activity has to guarantee the access only to the users not registered in the database application yet.
    3. Registration activity has to effectuate the registration procedure with the insert of all the fields: email, password, name, surname, specificity of ownership.
    4. Registration activity has to return an error notification if the fields are inserted in a wrong way.
    5. Registration activity has to return the user to the log-in screen if the registration is successful.

- Home requirements

    1. The Home menu has to allow the user to navigate through the main functionalities in a fluent and efficient way.
    2. The Home menu has to be constantly displayed in order to facilitate the interaction through the different screens.
    3. The Home menu has to redirect the user to the Google Map screen, to the Share Link screen, to the My Events screen, to the Create Event screen and to the Profile screen, based on the choice of the user itself.

- Event map requirements

    1. The event map has to display the markers of the events in a clear and distinguishable way.
    2. The event map by default has to position the user to the current location.
    3. The event map has to offer the possibility to change the view of the map itself, passing through two different levels with the apposite button: satellite mode and Google Map predefined mode.
    4. The event map has to offer the possibility to return to the current position through the apposite button.
    5. The event map has to provide a search bar in which is possible to search for a location and then update map position to the request location.
    6. On the click of markers, the event map has to display the correspondent sliding-up panel.
    7. The event map has to show the public events markers correspondingly to the filter parameters selected.
    8. The event map should display a list of boxes containing the different typologies of places used as a filter parameter.

- Filter panel requirements

    1. The filter panel has to display different sliders according to the different filter parameters required.
    2. The filter sliders have to offer the possibility to select different ranges of values.
    3. The filter panel has to display a filter calendar used for searching the events by date.
    4. The filter calendar has to offer the possibility to select one or several ranges of dates.

- Marker panel requirements

    1. The marker panel has to display the event info: photo, name, location, address, starting date and ending date, actual number of participants, description and price.
    2. The marker panel has to display the buttons that correspond to the possibility of sharing the event link and of asking to participate if the user is "normal".
    3. If the user is owner, the marker panel will not display the ask to participate button.

4. If the user is normal, the marker panel will display a notification regarding the state of his application, if performed: accepted by the owner or waiting for acceptance.

- Create event requirements

1. Create event has to offer the possibility to a owner user of creating a new event.
2. Create event requires that all the present fields has to be compiled in order to complete the creation.
3. Create event has to offer the possibility to load an image characterizing the event that will be stored and retrieved when necessary.
4. Create event has to display the following event fields: name, description, place, typology of location (based on his registered locals), privacy, starting date and hour, ending date and hour, maximum number of participants and price.
5. Create event has to inform the user about the right creation of the new event.

- Profile requirements

1. Profile has to display a personalized logo of the user with its initial letters of name and surname as well as the email address.
2. Profile has to display a button for the organized events only to the owner user.
3. Profile has to display a button for the participating events to both the types of user.
4. Profile has to display for the owned locals only to the owner user.
5. Profile has to offer the possibility of going back to the tutorial is requested, by clicking on the apposite text.
6. Profile has to offer the possibility to send a message to the developers of the application in order to report some bugs or to enter in contact with them for any necessity
7. Profile has to display a button for the log-out from the current account.

- My locals requirements

1. My locals has to show the list of the locals owned by the current owner user.
2. My locals has to offer the possibility of registering a new local through the correspondent button.

- Add local requirements

1. Add local has to offer the possibility to register a new local by the owner user.
2. Add locals has to require the address and name of the new local.
3. Add locals has to require the adding of the right documents that certificate the real ownership.

- Organized events requirements

1. Organized events has to display the list of all the events organized by the current owner user.
2. Each box of the list has to contain the main info about the correspondent event: photo, name, starting and ending date.
3. Each box has also to contain a button that will redirect to the event screen more info regarding the event.

- My events requirements

1. My events has to display the list of all the events in which the current user participates.
2. Each box of the list has to contain the main info about the correspondent event: photo, name, starting and ending date.

3. Each box has also to contain a button that will redirect to the event screen with more info regarding the event.

- Event screen requirements

1. Event screen has to display more info about the current event.
2. Event screen, in addition to the previous info, has to show to both the types of user the address of the event, the number of participants, the price and the description.
3. Event screen has to show the share link button to both the types of user.
4. Event screen has to show the ask to participate button to the normal user only if he doesn't have applied yet.
5. Event screen has to show the show qr code button to the normal user only if he's participating correctly to the current event.
6. Event screen has to show the scan qr code button only to the owner user in order to let him controlling the various qr codes.

- Qr code requirements

1. Qr code requires the access to the camera of the owner user's phone.
2. Qr code has to check the correspondence of the scanned qr code with the ones present in the list of the current event.
3. Qr code has to notify the current user both in cases of successful match or not.

2.5 Non-Functional requirements

Below we are listing some requirements that are necessary in general to guarantee a functional application.

- Portability: The application should also be extended to IOS and desktop users in order to be used by the possible largest number of user. This adding implementation will be introduced in the future.
- Stability: The application should always be available at any time it is needed. Failures of the system and server side crashes must be avoided in order to guarantee this requirement.
- Availability: The system must always be ready to be used even during a failure period. A backup facility must be present in order to do so.
- Reliability: Data must be reliable, so the remote database must be secure to satisfy this requirement.
- Efficiency: The application needs to use the least number of resources possible. Algorithms and data structures should be used as efficiently as possible to do so.
- Extensibility: The application should allow further extensions in the most simple way, without modifying the core functionalities.
- Maintainability: The application code should be well written, readable and well documented, making easier to be changed or extended by future programmers.

3 Data Design

3.1 Database design and implementation

In order to choose how to design the database first of all we created a possible Er Schema to be used. Below at figure 1 you can see the final version of that Er schema. Initially we wanted to use a SQL database but since Flutter provides a nice integration with Firebase Cloud Store we ended up using it, since this could save a lot of time, even though we hadn't a lot of experience with NO-SQL databases. The Firebase real-time database is cloud-hosted, the data is stored as JSON and synchronized in real-time to every connected client. In this case all the clients can share one real-time database instance and automatically receive updates with the newest data.

Now we want to provide a brief description of every entity we used, their attributes as well as their meaning and their relationship with other elements.

3.2 Entities and attributes

- Event
    1. EventId: The primary key of the event (String)
    2. Manager: The manager of the event (String)
    3. Description: The description of the event (String)
    4. DateBegin: The day in which the event starts (String)
    5. DateEnd: The day in which the event ends (String)
    6. Max participants: The number of maximum participants for the event (Integer)
    7. Type of Place: The type of the place associated to the event (String)
    8. Image: The link containing the image associated to the event (String)
    9. Type of event: The type of the event (String)
    10. FirstFreeQrCode: The number of the next qrCode to be associated to the next partecipant(Integer)
    11. Name: The name of the event (String)
- QrCode
    1. QrCode: The QrCode identifier(String)
- Local
    1. LocalId: The primary key of the local(String)
    2. PlaceName: The name of the local(String)
    3. LocalAddress: The address of the local(String)
    4. Latitude: The latitude of the local(Double)
    5. Longitude: The longitude of the local(Double)
- User
    1. ID: the primary key of the User (String)
    2. Name: The surname of the user (String)
    3. Surname: The surname of the user (String)
    4. Email: The email of the user (String)
    5. Password: The password of the local(String)
    6. isOwner: Indicates if the user is an owner of a local(Boolean)

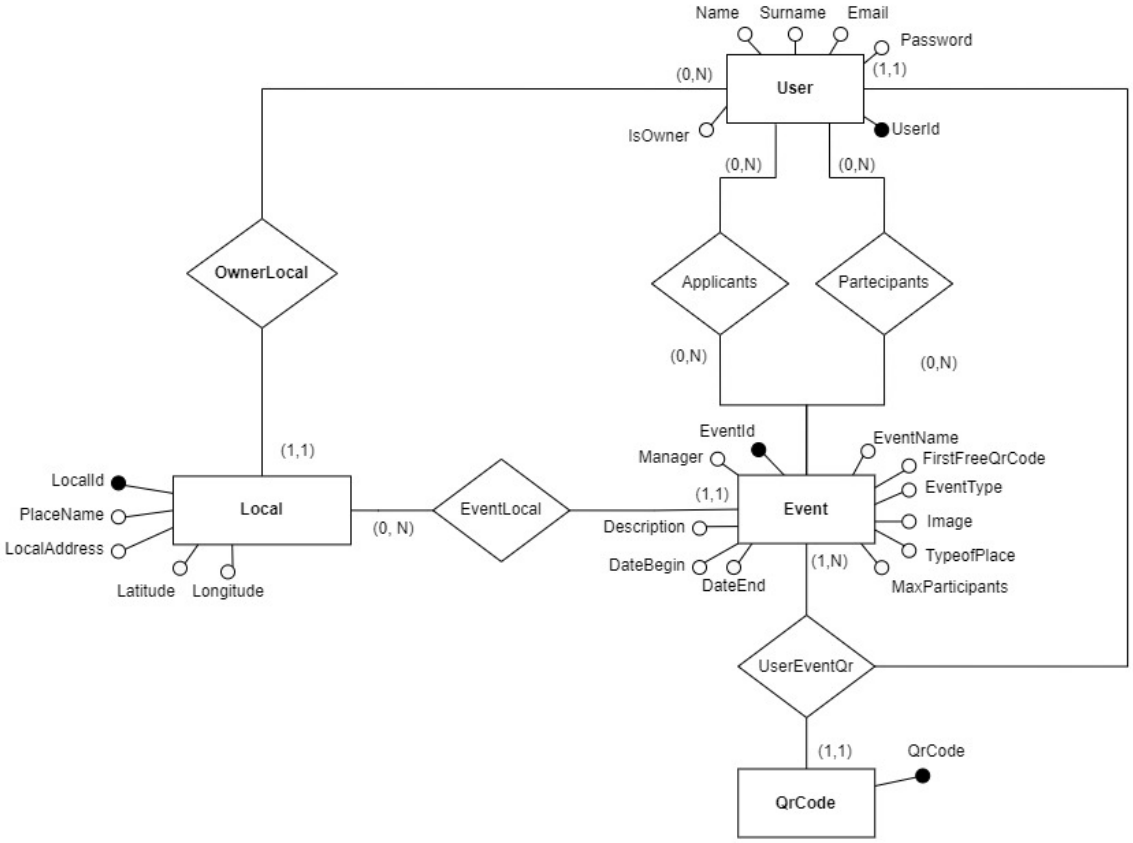

Figure 1: Initial Er Schema for the database.

4 Architectural Design

4.1 High-Level Components

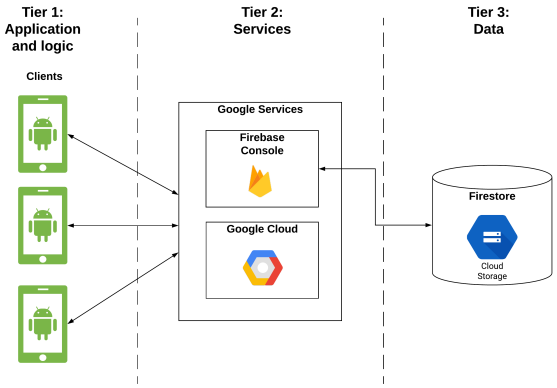

Figure 2: App Organization

Through the 3-tiers client-server architecture, the functional process logic, data access, computer data storage and user interface are developed and maintained as independent modules on separate platforms. It is particularly useful since it provides great freedom to developers who can independently update or replace only specific parts of the application without affecting the product as a whole. Thanks to

that, the application can be scaled up and out rather easily by detaching the front-end application from the databases that are selected according to the individual needs of the customer. Furthermore, it provides a higher degree of flexibility to enterprises who may want to adopt a new technology as soon as it becomes available.

4.2 Deployment view

In this section we are going to illustrate and describe the deployment view of our application. The deployment contains on major devices, the Firebase server,which communicate over a HTTP-SSL connection. In our case, the application runs on the mobile device and it has the job to communicate with Firebase services connecting them with its functionalities. Firebase server is a fundamental element of our application for data management and system architecture. It also provides the interaction with Cloud Firestore and Firebase Storage, which have the task to store all the necessaries information run-time created by the functionalities of the application. All the info about are postponed to the section of the External components, in which they will be addressed more specifically.

4.3 Flutter package organization

In the image below it is possible to see how the application was organized. We have chosen this division in order to separate the different parts of the application. We will explain below what each folder is used for.

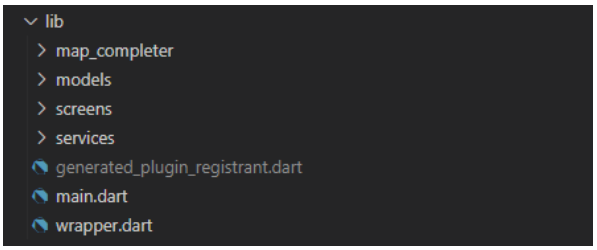

Figure 3: App Organization

In the model folder we added all the classes that represent the entities of the database

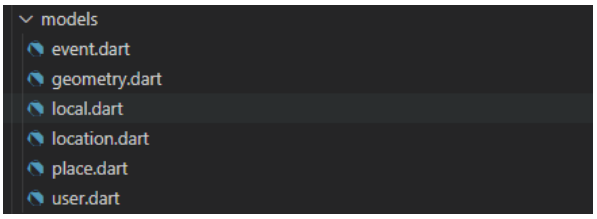

Figure 4: Model

In the Screens folder we added all the visible screens and they were also divided into sub-categories, like the one used for authentication, the ones related to events etc. etc.

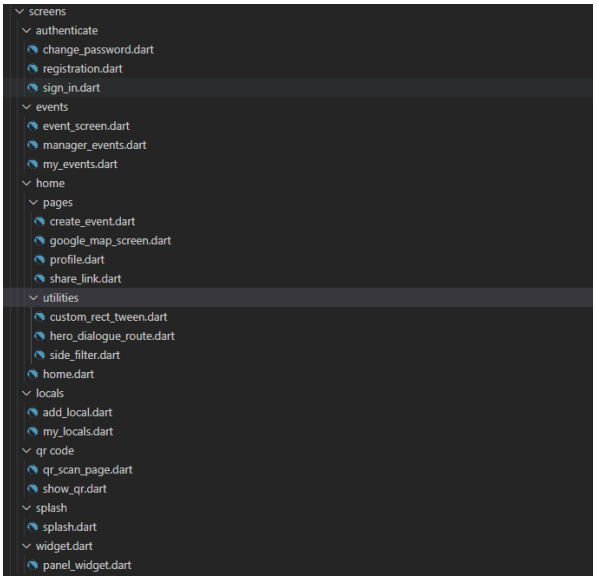

Figure 5: Screens

In the Services folder we added all the code used to handle the part of back-end with the database and also other useful classes used for localization

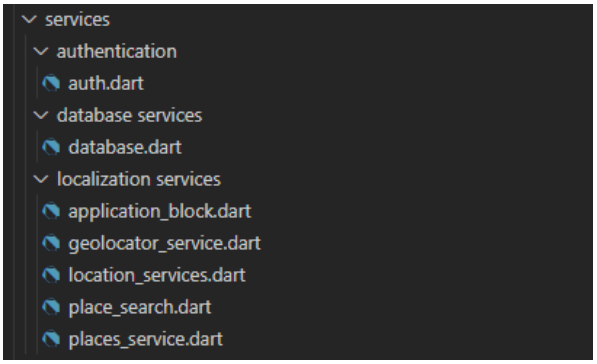

Figure 6: Services

5 User interfaces

In this section we will present the main screens of the application. The design of the application was focused mainly for mobile phones, however it's also adapted for tablet or larger screens devices. Layouts, styles, and colors have been inspired and chosen taking in consideration Android Material Design guidelines.

5.1 Splash screen

The splash screen is the first screen the user sees once started the application, while in background the application is retrieving information from the database. Once ready, the application will redirect the user to the event home map screen.

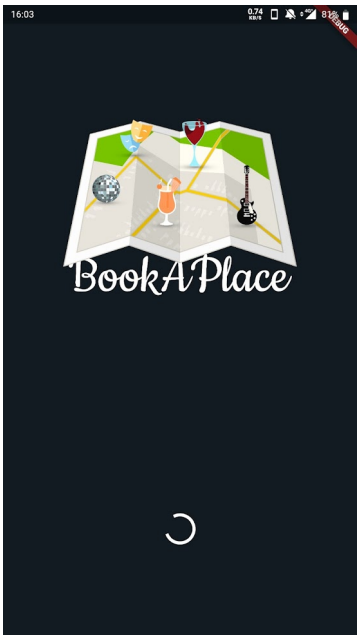

Figure 7: Splash Screen

5.2 Tutorial

The Tutorial is displayed on the first run of the application. It briefly welcomes the user to the application, explains the main functionalities and offers an introduction for the next screens. During the first run, once terminated, the Tutorial will redirect the user to the Login page. During the next runs, the user will be redirected to the Home page. The Tutorial is also reachable whenever the user wants from the Profile screen.

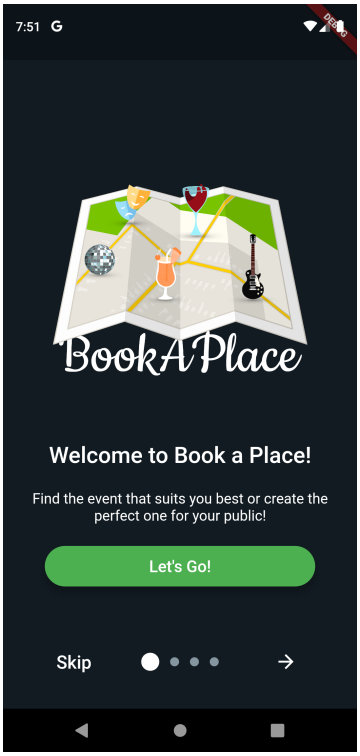

Figure 8: Tutorial first page

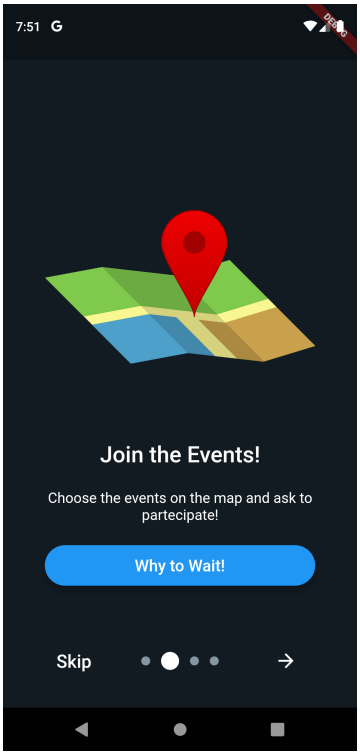

Figure 9: Tutorial second page

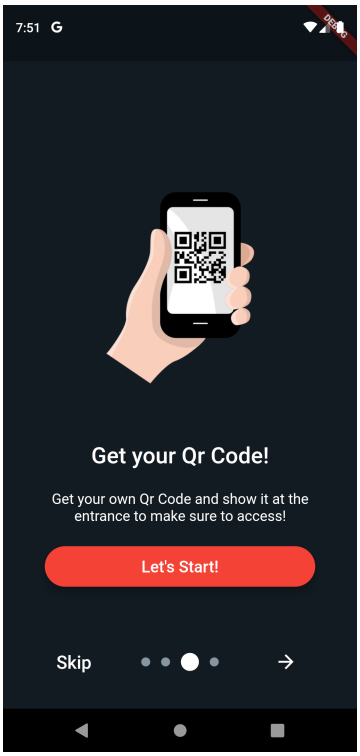

Figure 10: Tutorial third page

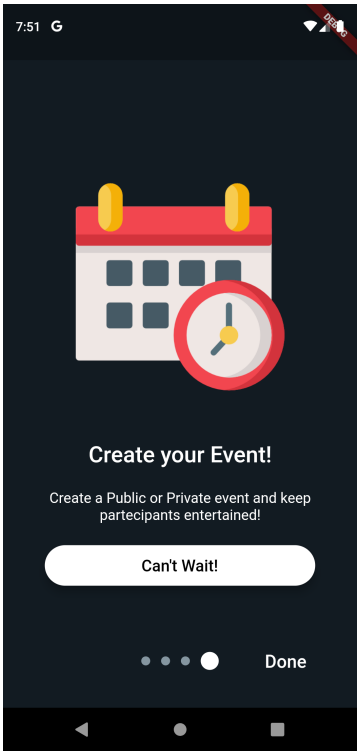

Figure 11: Tutorial fourth page

5.3 Login

The Log-in is displayed if the user is not logged in when the application is started. It is displayed automatically also during the first run after the tutorial is completed. It allows to enter the application with the own email and password, if previously registered in the Sign-up page. If the credentials are not correct, a notification error message is displayed on the screen. If the log-in is successful, the user is redirected to the home map screen page.

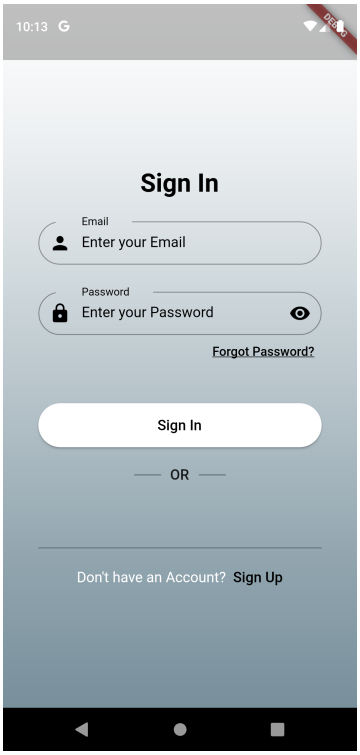

Figure 12: Tutorial fourth page

5.4 Registration

The Sign-up is displayed together with the log-in page if the user is not logged in when the application is started. It allows to register a new account in the application through the insert of name, surname, email, password and specifying the possible ownership of a local. All the fields are mandatory otherwise the account won't be created. If the fields are not correct, a notification error message is displayed on the screen. If the registration is successful, the user is redirected to the log-in page in which can insert the credentials.

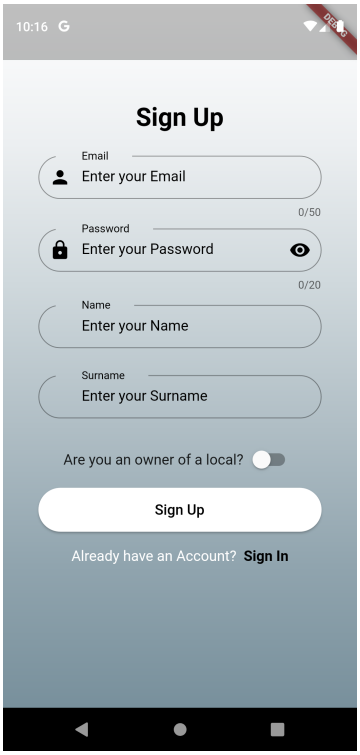

Figure 13: Tutorial fourth page

5.5 Home page

The home page screen is shown once the user has signed in or he was already signed in and starts the application. In this screen the user can navigate through the map and search for events in the area. The user can search for precise locations through the search bar the will directly move him in the desired place. The button located in the high right part allows the user to change the view of the map, passing from the default Google Map view to the satellite view and vice versa. The button in the bottom right part instead, allows the user to move directly to his current location. In this screen, under the search bar, are also present the little boxes with the different types of setting, used for filtering the preferred ones. We the bottom navigation menu will also be displayed during all the interactions with his main four functionalities, helping the switching and the communication between the different screens.

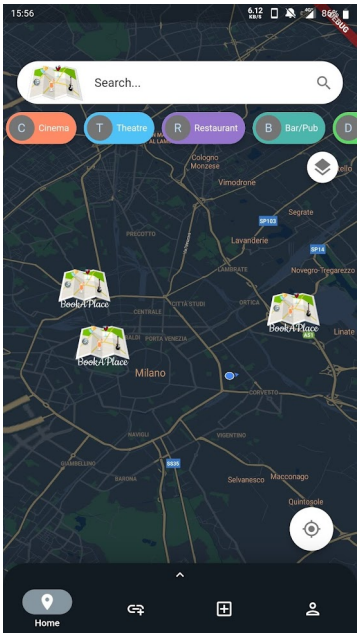

Figure 14: Home page

Figure 15: Satellite view

5.6 Filter Panel

The filter panel screen is characterized by two main pages screen. The first one displays three different sliders through which the user can select the desired range of the parameters: price, number of participants and distance in km from the current position. The second one displays a calendar in which the user can select the desired ranges of dates. All these values together with the little boxes in the Home page, once applied, will filter out the entire list of created events, displaying the correspondent markers in the map. Remember the the created private events will be never displayed on the map.

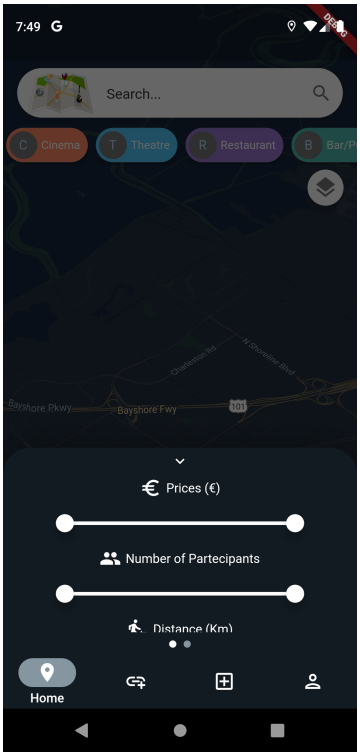

Figure 16: Filter sliders

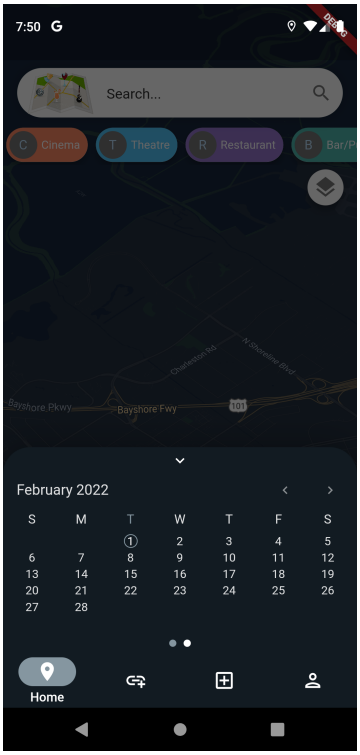

Figure 17: Filter calendar

5.7 Marker Panel

Once the user has clicked one marker on the map, this panel will come up sliding from the bottom of the screen. It shows the main details of the event such as title, address, starting and ending dates, description, and price. It offers the user the possibility to share the event through a link or the possibility to ask to participate to the event, if he has not applied yet. In those cases when the application is waiting for the owner of the event to be accepted or when the application is confirmed, the user will see a text notification and not anymore the "ask to participate" button. The owner of the events will only see the "share the link" button

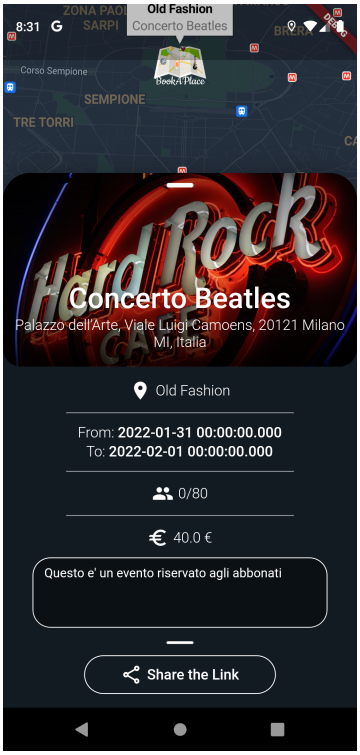

Figure 18: Marker Panel on map for the owner of an event

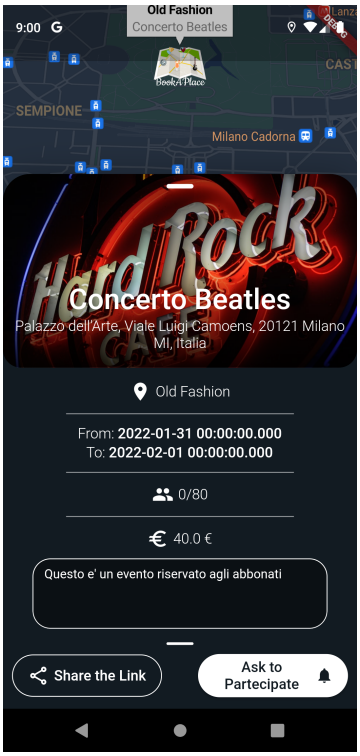

Figure 19: Marker Panel on map for the user that didn't apply yet

5.8 Search Event

This screen can be accessed by clicking on the second button in the bottom navigation menu. It allows the user to paste a link of an event in order to be redirected to the event screen. This is particularly useful when the event is private and the event can't be seen on the map but can only be accessed through the link shared by the event manager. The form will notify an error when the link is not correct.

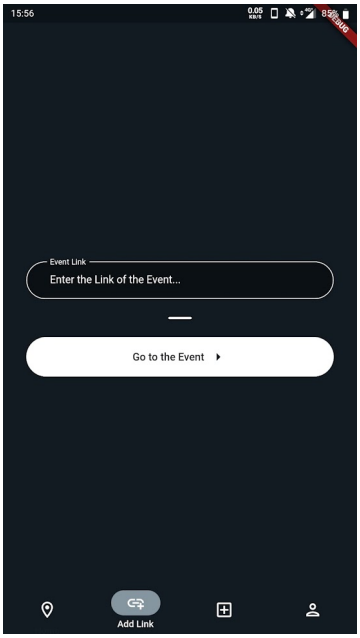

Figure 20: Search Event

5.9 Create Event

This screen can be accessed by clicking to the third button in the bottom navigation menu. It is shown only to the owner users and allows them to create a new event by compiling the form. All the different fields are mandatory otherwise an error notification will be displayed. It contains the name of the event, the description, the owned place, the typology of the location, the privacy of the event, the starting and ending dates, the price and the maximum number of participants. Moreover, there's the possibility to load a photo directly from the current device by clicking on the apposite button.

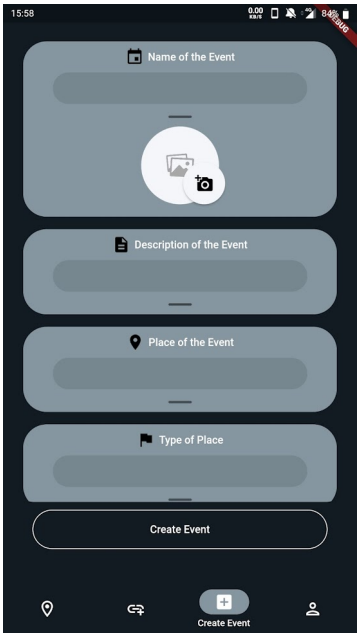

Figure 21: Create Event first part

Figure 22: Create Event second part

5.10 Profile

This screen can be accessed by clicking to the fourth button in the bottom navigation menu. It displays the private functions that a user can have. In the top part a logo with the personal initials and email is displayed to characterize a personal view. Both the types of user have the "My events" button, which will show the events a user is attending. In addition, the owner user will have two different buttons, "Organized events" and "My Locals". The first will display all the events organized by the current user, while the second will show a list of his owned locals. Moreover, both the users will have the "Log Out" button that allow the disconnection from the account, the "Go back to the Tutorial" button and the "Contact Us" button, used for communicating any bugs or problems and feedback to the developers.

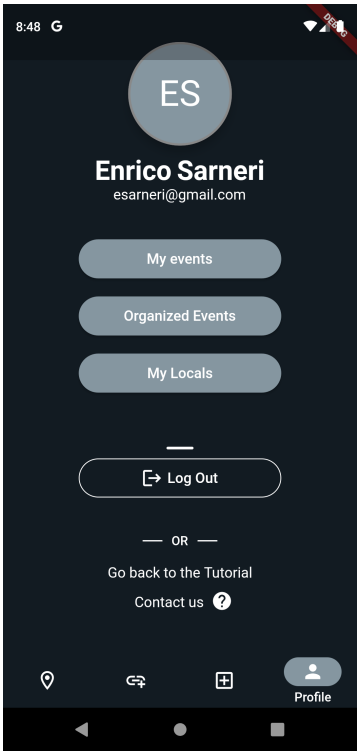

Figure 23: Owner user profile

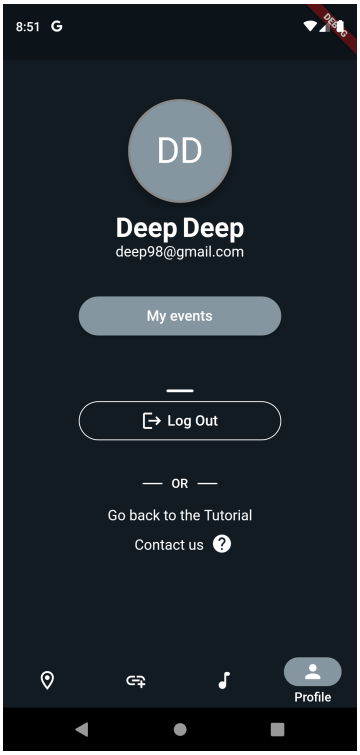

Figure 24: Normal user profile

5.11 Organized events

This page is shown only to the owner users and can be accessed by clicking on the 'Organized events' button present in "My Profile" only for owner users. The page shows all the events that are currently organized by the user. The main info are also displayed: the title, the place, the starting and ending date and the privacy. Moreover, a "More info" button is displayed in order to see more information about it.

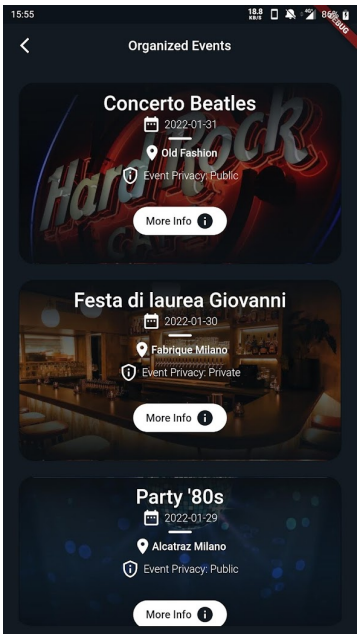

Figure 25: Organized Events

5.12 My events

This page is shown to both types of user and can be accessed by clicking on the 'My Events' button present in "My Profile". The page shows all the events the current user is participating at. The main info are also displayed: the title, the place, the starting and ending date and the privacy. Moreover, a "More info" button is displayed in order to see more information about it.

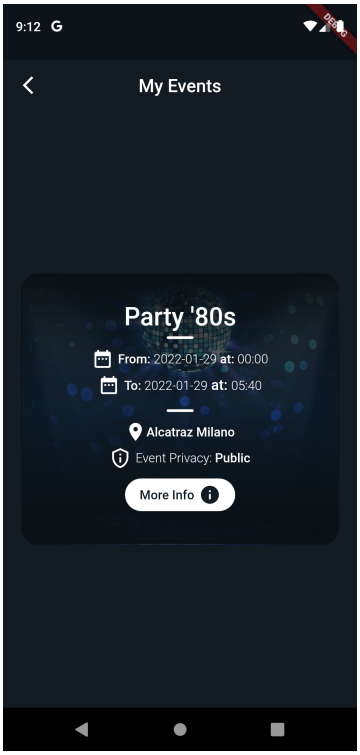

Figure 26: My Events

5.13 Event Screen

This page is displayed after clicking on the "More info" button present in the different boxes in inside "Organized Events" and "My Events". It will have a different aspect based on the fact that you're the creator of the event or not. If you're the creator, you'll see the additional info about it as the address, the description, the number of participants and the price. Furthermore it will be displayed the list of all the actual participants and the list of all the applications, ready to be accepted or refused. Finally there will be the "Share the Link" button and the "Scan Qr Code" button that will allow to copy the link of the event sharing it and to open the camera in order to check the correctness of the qr codes, respectively. If you're a participant of the event, you'll see the additional info about it as the the address, the description, the number of participants and the price. Furthermore it will be displayed the "Share the Link" button, a notification saying if you're waiting for the acceptance or already participating. In the latter case you'll also see a "Show Qr Code" button that will display the unique and correspondent Qr code for the event.

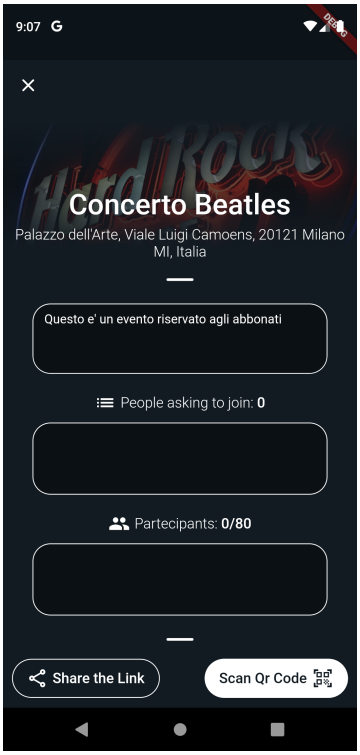

Figure 27: Event screen for the creator

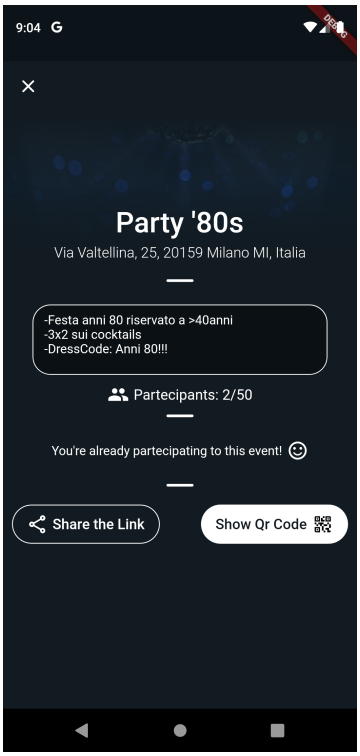

Figure 28: Event screen for participants

5.14 Scan Qr Code

This screen is displayed after clicking on the "Scan Qr Code" button from the organizer event screen. It will open the camera allowing the owner of the event to check the different qr codes at the entrance of the event. It will display positive or negative notifications according to whether the Qr code corresponds to one of those in the list of participants or not.

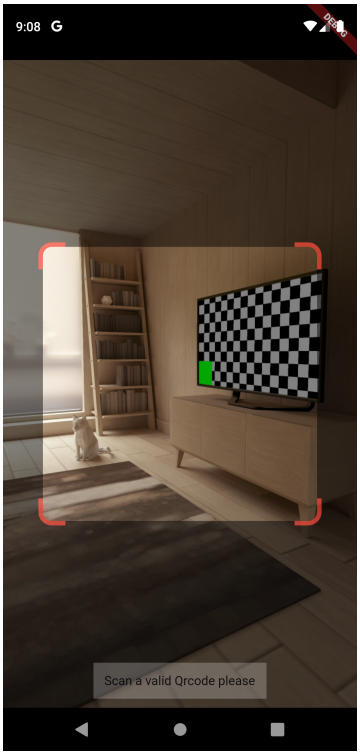

Figure 29: Scan Qr Code

5.15 Show Qr Code

This screen is displayed after clicking on the "Show Qr Code" button from the participant event screen. It will show the personal Qr code related to the current event.

Figure 30: Scan Qr Code

5.16 My locals

This page is shown only to the owner users and can be accessed through the correspondent button present in "My Profile". This screen allows the owner user to see his locals and in case to add a new one by clicking on the "Add Local" button.

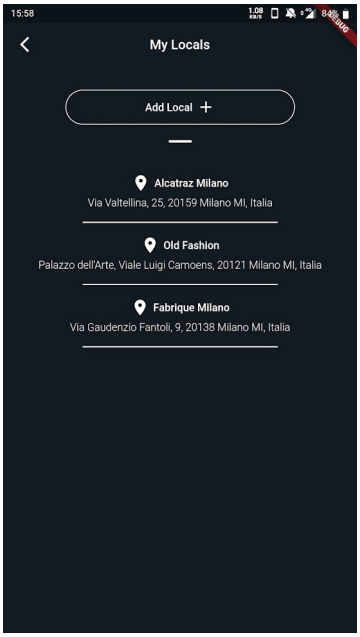

Figure 31: My Locals

5.17 Add local

This page is shown only to the owner users and can be accessed by clicking on the "Add Local" button present in the "My Locals". A new local can be added by compiling the form which requires the address of the local, the name of the local and the documentation required that guarantees the ownership of the local. In case some of these fields is not compile, a notification error will be displayed.

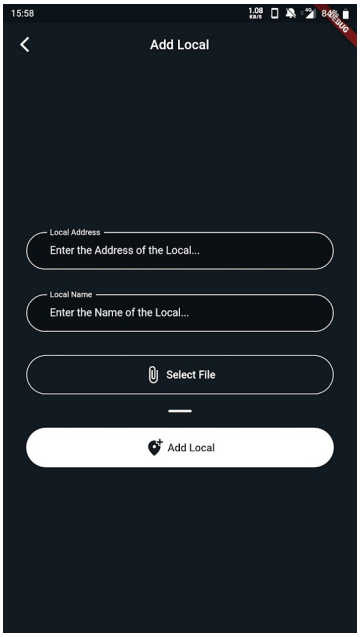

Figure 32: Add Local

6 External Services and Libraries

In our application we've used different external services to reach all the goals of our application. The main advantages of using external services is the fact that we don't have to implement specific portions of code in order to achieve the same result that these external services provide by themselves.

6.1 Google Firebase

6.1.1 Cloud Firestore

We choose to use Cloud Firestore since is a flexible No-SQL document database with high performance and specially for its ease of application development. It keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. We stored here all the models used for our app like Users, Events, Locals of users, and Qr Codes of Users for a specific event.

6.1.2 Authentication

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend. This service is of fundamental importance since it handles all the authentication procedures of the users. The user is allowed to register using a form inserting his main data such as name, surname, password, email and whether is or not an owner of a local. In this case, in order to authenticate the user, we used their email addresses and passwords through the Firebase Authentication SDK.

6.1.3 Storage

Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service. The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for your Firebase apps, regardless of network quality. In this application we used it to store the photos uploaded by the owner user in order to create the correspondent event and to retrieve them in the different screens of the application when necessary. It offers robust uploads and downloads since they restart where they stopped, saving your users time and bandwidth.

6.2 Google Play Services

6.2.1 Geocoding API

This service allow to convert an address into GPS coordinates and it is used to find a specific address in the map and also to ease the adding of a local for an owner, since the owner only needs to add the address of the local and not its coordinates.

6.2.2 Geolocation API

This service allows to retrieve the current user's location information and it is useful to provide a fast way to let the user localize himself in the map. Communication is done over HTTPS using POST. Both request and response are formatted as JSON, and the content type of both is application/json. This service requires the user to allow the application to have access to coarse or fine position.

6.2.3 Maps SDK

It is probably the main service we used, since thanks to it we can present the user the various events placed on an interactive map. The possibility to see the events placed on the map is one of the primary features of the application, since it is an immediate way for the user to catch and find the event he's looking for. With the Maps SDK for Android, we added the map to our application using Google Maps data, map displays, and map gesture responses. Trough it, we also provided additional information for map locations and supported user interaction as the adding markers and overlays.

6.2.4 Places API

The Places API is a service that returns information about places using HTTP requests. Places are defined within this API as establishments, geographic locations, or prominent points of interest. The service we used is the query auto-complete which provides a query prediction service for text-based geographic searches, returning suggested queries as users type. This is used in the search bar of the map.

6.3 Other minor libraries

Other libraries and services of secondary importance have been used in the application, in order to make the development easier and quicker, without the need to re-implement some basic functions.

6.3.1 Qr Code scanner

This service allows the owner of an event to scan the Qr code of the incoming people in the event checking if they're in the participant list.

6.3.2 Qr Flutter

This service allows to render a Qr Code associated to the correspondent user for a specific event.

6.3.3 Permission handler

This service allows the application to ask the permission at run-time to see if they're not granted since they're fundamental for the correct behaviour of the application.

6.3.4 Share

This service is used to share the link of an event through different applications like WhatsApp or Telegram or simply copying the link.

6.3.5 Uuid

This class is used to generate unique identifier for the Qr Codes to avoid 2 or more people having the same Qr Code.

6.3.6 Fake Cloud Firestore

This class is used only for the testing part of the application and allows to mock the dependencies of our app from the database. It's not essential for the correct behaviour of the application but it allowed us to find bugs and check the correct behaviour of the methods and widgets.

6.3.7 File picker and Image picker

These services have been used in order to retrieve the files and the photos from the device storage in order to proceed to the adding of a local by the owner user and to the correct creation of an event, respectively.

7 UML Diagrams

In this section we present some useful diagrams that helps the reader to better understand the interaction between the user and the application. Here we present the main ones.

7.1 Use Case Diagrams

These diagrams show the flow of operation triggered by a specific actor when it tries to perform some tasks. The actor can be the user (a human actor) or a system. The use cases we propose show the main operations that are possible to perform in the system.

7.1.1 Unlogged User interaction

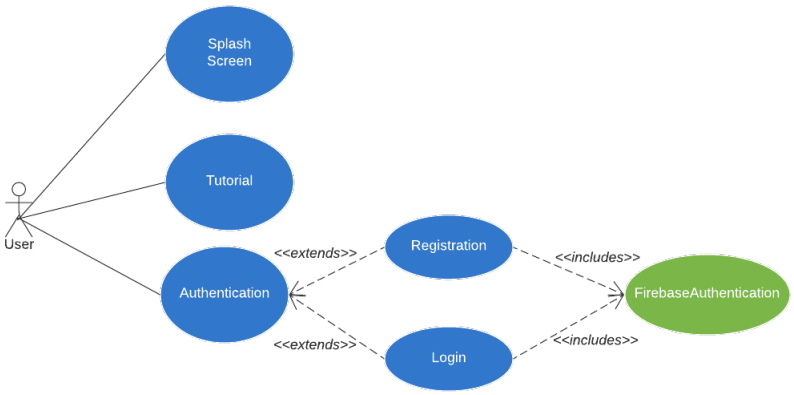

Figure 33: Unlogged User Use Case

7.1.2 Normal User interaction

Figure 34: Normal User Use Case Part 1

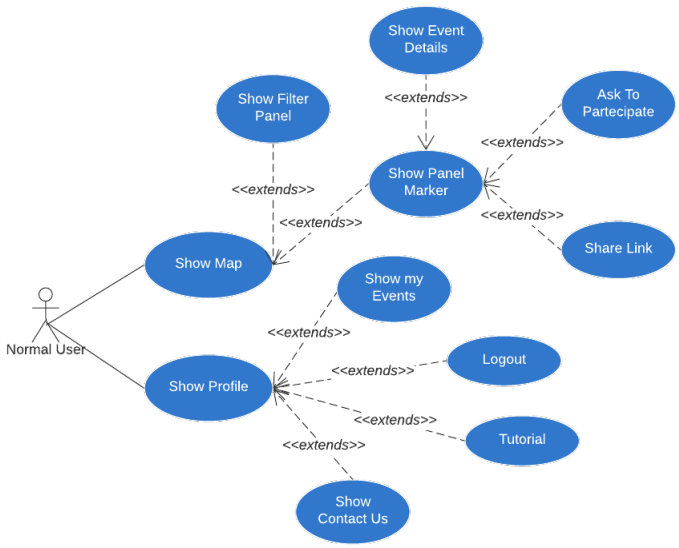

Figure 35: Normal User Use Case Part 2

7.1.3 Owner User interaction

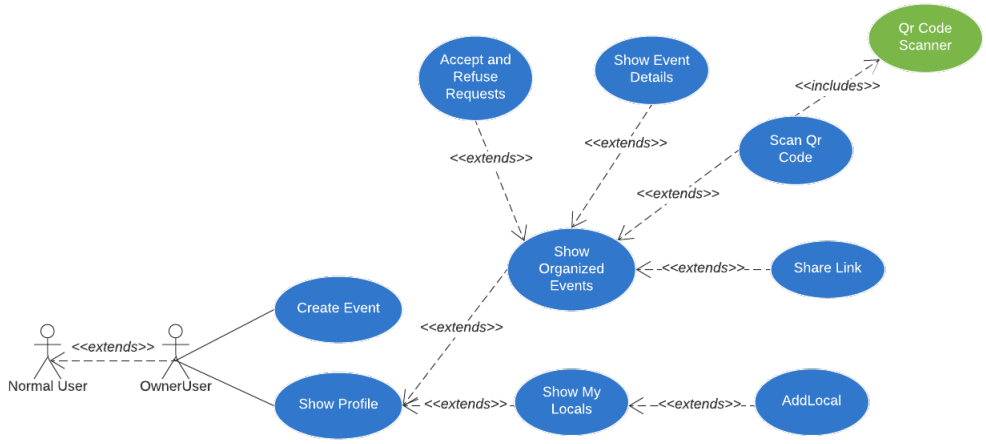

Figure 36: Owner User Use Case

7.2 Sequence Diagrams

In this section we provide few sequence diagrams concerning the flow of the logical operations performed by our application once a certain operation is executed. These diagrams will show the interaction between methods and activities, rather than user and application. We decided to include only two examples of sequence diagrams, in order to make the document not too complex and less readable. The two operation chosen are:

- Searching of the desired address on the map search bar. This diagram shows how the user, once opened the map screen, can access the search bar and write the desired address. Once the research has been completed successfully, the map will move the camera directly to the place thanks to the conversion from address to geographic coordinates provided by the services.

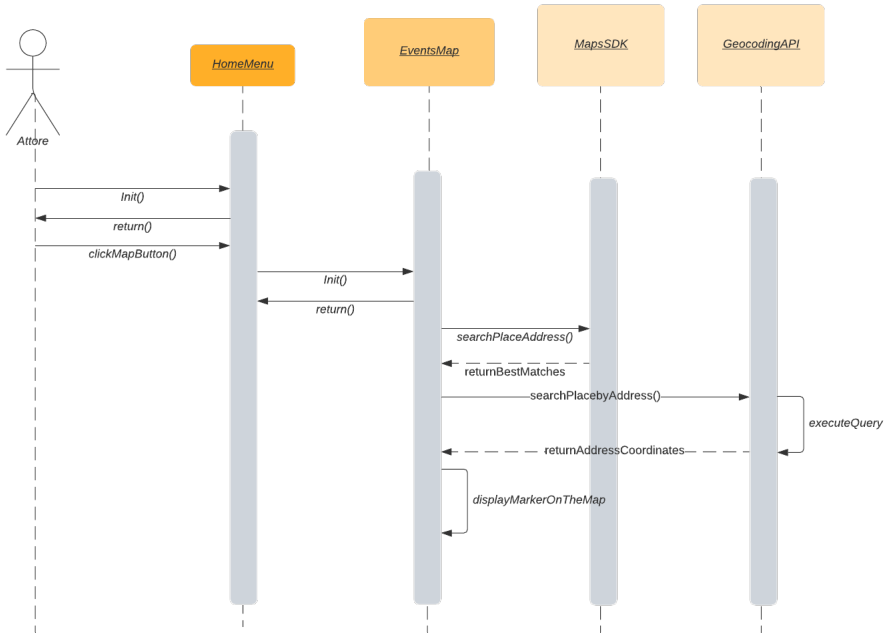

Figure 37: Address research Sequence Diagram

- Asking for a participating to a certain event and the correspondent acceptance by the owner user. In this case, the potential participant user opens the panel marker on the map and clicks on the "Ask to participate" button and waits for a response. From the other side, the owner user, when accesses the detailed page of the created event will decide to accept or not the request.

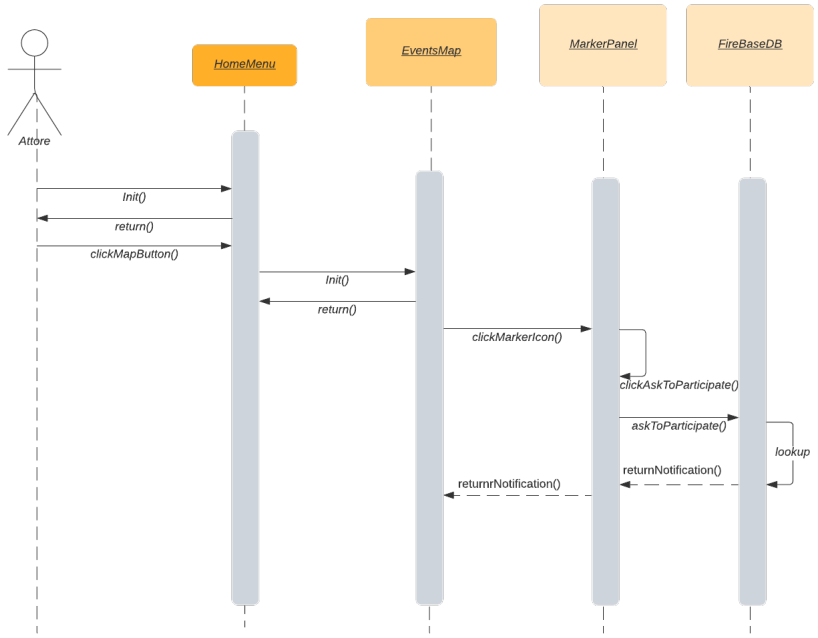

Figure 38: Ask to participate Sequence Diagram

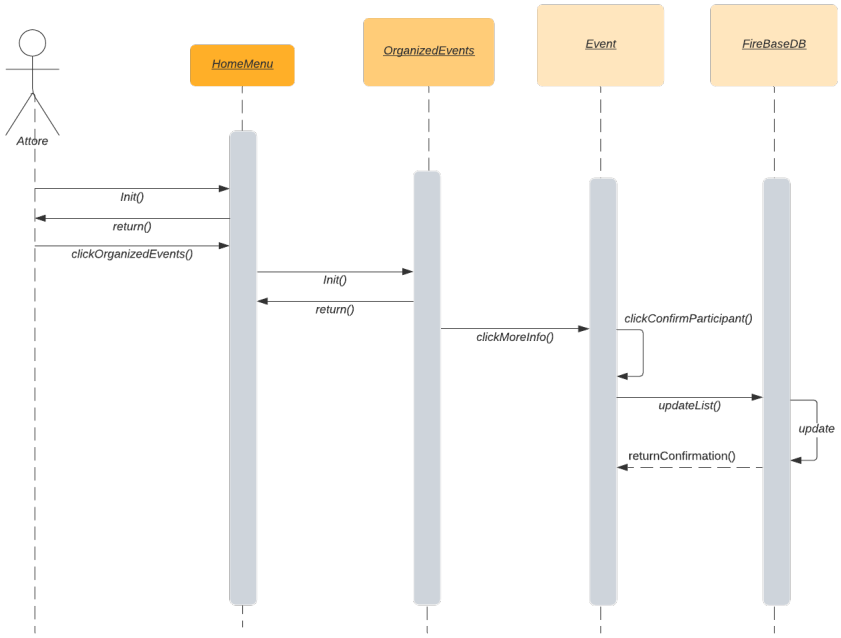

Figure 39: Acceptation of request Sequence Diagram

8 Code dependencies

In this section are listed all the code dependencies necessary for the correct development e operation on our application.

```yaml
dependencies:
  flutter:
    sdk: flutter
  google_maps_flutter: ^2.1.1
  http: ^0.13.4
  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  image_downloader: ^0.31.0
  transformer_page_view: ^0.1.6
  cupertino_icons: ^1.0.2
  firebase_core: ^1.10.0
  firebase_auth: ^3.2.0
  cloud_firestore: ^3.1.0
  provider: ^6.0.1
  fluttertoast: ^8.0.8
  geolocator: ^8.0.1
  geocoder: ^0.2.1
  rxdart: ^0.27.3
  introduction_screen: ^2.1.0
  draggable_bottom_sheet: ^0.1.2
  smooth_page_indicator: ^1.0.0+2
  uuid: 3.0.5
  animations: ^2.0.2
  sliding_up_panel: ^2.0.0+1
  flutter_svg: ^1.0.0
  draggable_scrollbar: ^0.1.0
  syncfusion_flutter_sliders: ^19.4.41
  syncfusion_flutter_datepicker: ^19.4.40
  shared_preferences: ^2.0.12
  qr_code_scanner: ^0.6.1
  qr_flutter: ^4.0.0
  file_picker: ^4.3.1
  image_picker: ^0.8.4+4
  intl: ^0.17.0
  firebase_storage: ^10.2.5
  share: ^2.0.4
  permission_handler: ^8.3.0
  fake_cloud_firestore: ^1.2.0
  firebase_auth_mocks: ^0.8.2
  network_image_mock: ^2.0.1

  numberpicker: ^2.1.1
  icon_shadow: ^1.0.1
  dropdown_button2: ^1.1.0
```

Figure 40: Code dependencies

9 Test

This section describes the tests done on Book a Place. In almost every screen we were forced to use Mock components since a lot of screen were taking data that was coming directly from the database. Thanks to Fake Cloud Firestore library we didn't need to mock everything since this library was already doing this for us and we could use it as a "fake version" of the Cloud Firestore. This allowed us to save a lot of time for the testing part.

9.0.1 Widget Tests

The widgets test were done on most of the screen and their purpose was to check that every widget components were present. In some screen such as: My Events, My Locals, Event Screen thanks to fake cloud firestore we added fake data to see if the screen was correctly showing the data coming from the fake database. We also checked that for specific screen such as Event Screen, Create Event some components were only visible to owner users such as Scan Qr and the Event screen button not to normal user

9.0.2 Unit Tests

The unit test were done to test the core functionality of the app which were all contained in the class DatabaseService. This test are very important since the behaviour of the application is based on this class. We tested almost all the main functionalities such as:

- The adding of an user to the applicant list.
- The accepting of an user in the applicant list and the add to the participating list.
- The assignment of a valid Qr Code for the user added in the participating list.
- The refuse of an user in the applicant list.
- The method to return the current user signed in.
- The method to check if the current user is an owner user.
- The adding of a local for an owner user.
- The method to retrieve an event by it's id.
- The method to retrieve a Qr Code of a specific user of a specific event.

Every time we tested 1 base/normal case which we can say is the normal behaviour of the method and also 1/2 limit cases in which the behaviour of the method a bit different. This is true both for widget and unit tests. Unluckily we didn't manage to add the integration test due to the fact that our app on every first run asks for permissions, which are required for all the functionalities of the app, and we couldn't find a way to grant this permissions when performing an integration test.