POLITECNICO
MILANO 1863

Design Document

Raffaele Cicellini(10626081)
Francesca Forbicini (10628756)

Design and Implementation of Mobile Application
Professor Baresi Luciano

A.Y. 2022/2023

Contents

1 Introduction

1.1 Purpose

This design document provides necessary informations about the mobile application "Spotify Music Quiz" realised for the "Design and Implementation of Mobile Applications" course held at Politecnico di Milano. The document includes definitions, design principles, functionalities and interfaces used and provided by the app.
The purpose of the design document is to present the design details about the implementation of the application. It serves as a reference during development and as a complete description of the project once completed.

1.2 Definitions, acronyms, abbreviations

User = registered user of the application, he plays the quizzes

Player = used as synonym of User, it is a user of the application

Quiz = an infinite set of questions regarding a specific topic

Artist = an artist that the user follows on Spotify, it can be the topic of a quiz

Playlist = a Spotify playlist of the user, it can be the topic of a quiz

Location = the country corresponding to the user gps coordinates

API = Application Programming Interface, it is a common way to communicate with another systems

OS = Operating System, it is the main software that manages the device hardware and software resources, it provides common services to other programs and applications

UX = User eXperience, it defines how a user interacts with a product, system or application

UI = User Interface, it is a set of components where the interaction between user and application occurs

App = Application

Bio = Biography

1.3 Document structure

- Chapter 1: introduction describing the design document and its purpose
- Chapter 2: detailed description of the mobile app
- Chapter 3: description of architectural and design choices, sequence diagrams for the main implemented functionalities
- Chapter 4: UX diagrams and images representing the user interface and the interactions of the user with it
- Chapter 5: testing campaign summary
- Chapter 6: references used to write the document and to realize the mobile app

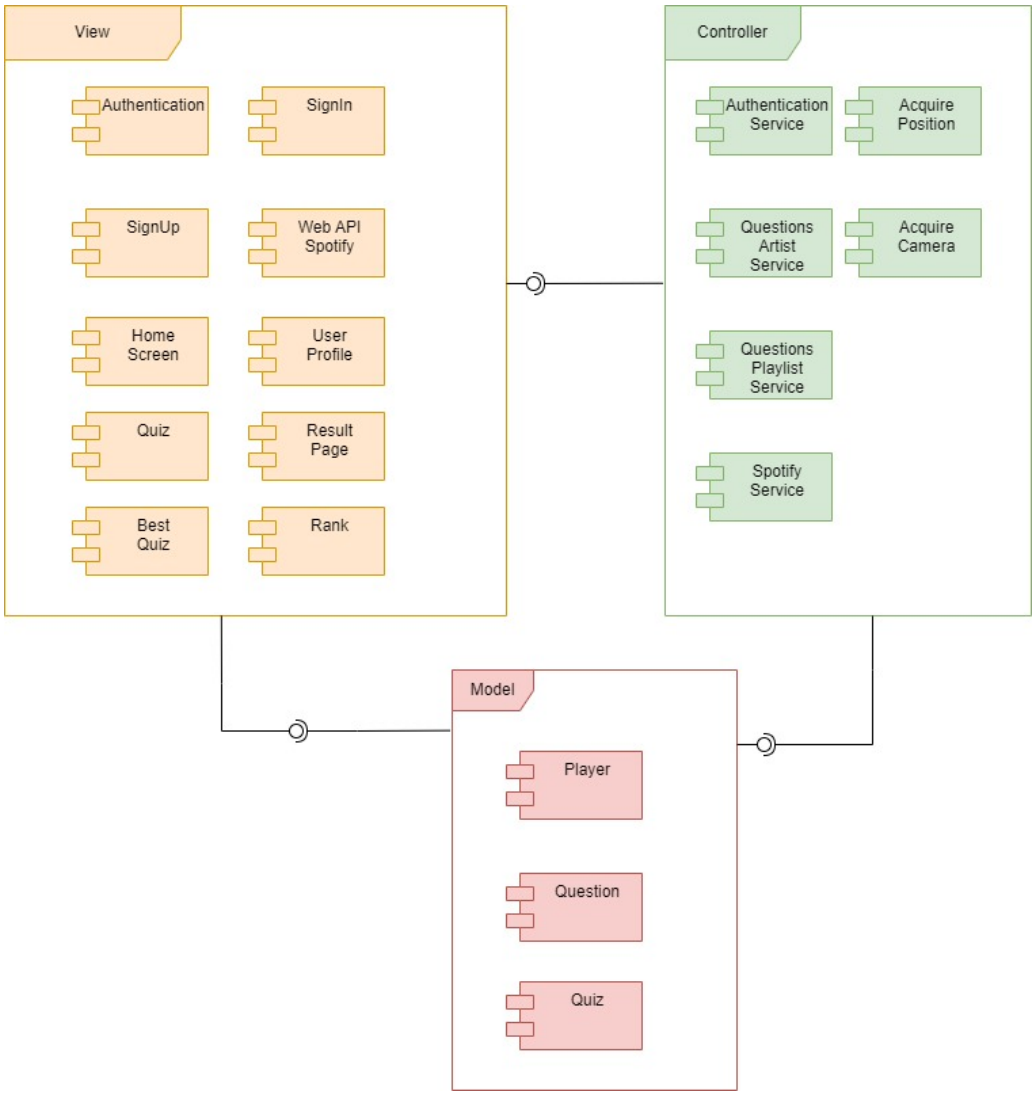
2 Application functionalities and implementation

The main objective of this mobile app is to make music-addicted test their knowledge about their favourite artists and playlists. They can also see how far other users of the app went with their quizzes.
Once the user opens the app he must allow it to access the position, since it is needed for the leaderboards. Then, after he signs up and logs in, he must allow the app to access his private Spotify account by logging into it from the app itself.
After these preliminary steps, a user can start a new quiz from the home page of the app by tapping on a playlist or on an artist. By doing it, the app retrieves all needed informations about the chosen topic (artist or playlist) and creates an initial list of questions. The questions can be of different types, such as "When album X was released?", "Who sing song X?", "Which album contains song X?". Some of the questions exploit the 30 seconds preview feature offered by Spotify Web API, so that the user must listen to the song and guess the title. If the user answers correctly to all the questions it goes to the next level, where there can be questions also about related artists to make them more difficult. The quiz ends when the user gives a wrong answer to a question or if it reaches the end of a level and decides to stop. In both cases, the app checks the current score and compares it to the user's best score: if it is greater, the user can save the quiz by taking a picture and saving the location to appear in the leaderboard.
From the home page, the user can also access its user profile page where he can modify his account photo, see his best score and the photo and location of the corresponding quiz, and access the leaderboard. The leaderboard page is divided in two tabs: in the first one, there is the global one with users from all over the world (the position of each user is the position where he did his best score quiz); the second one, instead, there is the local leaderboard based on the current gps position of the user (he will appear in the local leaderboard only if he did his best quiz in the country where he is currently located). In both cases, the user will appear in the standings only if he did and saved at least a quiz.

3 Architectural design

3.1 Baseline

We implemented a MVC architecture, where the model is composed of classes without any logic related to the logic of the quiz; the view is composed of all widgets that allows to let the user made a quiz, to show the playlists and the artists of the user, his/her profile with his/her best score and the ranking; the controller takes care of the logic of the quiz and to retrieve the playlists and the artists of a user from the Spotify API.



3.2 Overview

We decided to build Spotify Music Quiz using Flutter. Flutter is an open source framework made by Google to build in an easy way cross-platform applications from a single codebase. Each platform version of the app is natively compiled. The framework is powered by Dart, an object-oriented programming language designed by Google and optimized to build fast apps on any platform. Spotify Music Quiz uses different backend services, such as Firebase Authenticator, Cloud Firestore and Spotify Web API in order to deliver the desired functionalities. Firebase is an online platform made by Google to help in the creation of applications for mobile devices and for the web, since it offers a variety of backend services. In order to use it, we needed to register the app on the Firebase website.

3.3 Firebase Authenticator

In order to manage the login and registration of users for our app, we decided to use Firebase Authenticator since it provides easy-to-use backend functionalities. We decided to give the user the possibility to register with email and password or to sign in with his Google account, since we know that it can be annoying from the user point of view creating a new account and remembering a new password for yet another application. Either way, once the user signs in for the first time, a new user account is created with its unique user ID that identifies the user across our app.

3.4 Cloud Firestore

For the backend data layer we opted for Cloud Firestore, which is provided by Firebase. It is Firebase's newest database for mobile app development. It is a No-SQL database based on collections and documents: collections corresponds to tables in a SQL database, whereas a document corresponds to a row in a table. Each attribute is organized in a set of key-value pairs. The data stored by our app are organized in two different collections, whose structure is described below.

Users is the collection used to contain all needed informations for each user. Since the e-mail is unique, the id of each user document is the e-mail of the registered user. Each document in this collection contains:

- *username*: username of the user, it appears in the leaderboard;
- *bestScore*: best score of a quiz played by the user (at the beginning it is set to 0);
- *accessToken*: access token needed to use Spotify Web API to retrieve user's data;
- *refreshToken*: refresh access token, it is automatically used by the Spotify Web API once the access token is expired. The api also creates a new refresh token;
- *expiration*: expiration date and time of the access token;
- *clientId*: client id, needed by the Spotify Web API to verify which app is trying to use the API;
- *clientSecret*: client secret, needed by the Spotify Web API to verify that the app which is trying to use the API is the real one.

Quiz is the collection used to contain all needed informations for the best quiz of each user. Also for this collection, the id of each document is the e-mail of the user who made the quiz. Each document in this collection contains:

- *username*: username of the user, it appears in the leaderboard;
- *score*: the score of the quiz;
- *country*: the country code corresponding to the GPS location of the quiz;
- *position*: the position of the quiz saved in (region, country) format;
- *image*: the picture taken by the user when saving the quiz.

3.5 Spotify Web API

In order to easily use the Spotify Web API in Flutter we decided to exploit a Dart package called "*spotify*" (version 0.8.0): this package implements a lot of useful functions and classes in order in order to read data from the API in a meaningful way. The API offers a variety of different endpoints (which are URIs): to each of these endpoints we can perform different web GET methods in order to retrieve all the data that we need, such as the songs in a playlist, a spotify user's favourite artists, top tracks for an artist or related artists, all the metadata of each song and album (i.e. album's year release, song's artist and album, etc) and for each song a link to a 30 seconds preview that we use in the quiz to make the user guess the title based on the preview.
In order to use the Spotify Web API, we needed to register the app on the Spotify for Developers website: this provided us with the clientId and clientSecret needed to make queries to the API. Then, we used the package's functions to perform the needed queries to the API endpoints: we exploited its functionalities to manage in a quite simple way the process of granting the app and the user access to a set of public and private data of the Spotify user (by defining a list of scopes), to retrieve the user's playlists and favourite artists in the home page and to retrieve the metadata of each track in order to build the questions for the quizzes.

3.6 Multi-Language Support

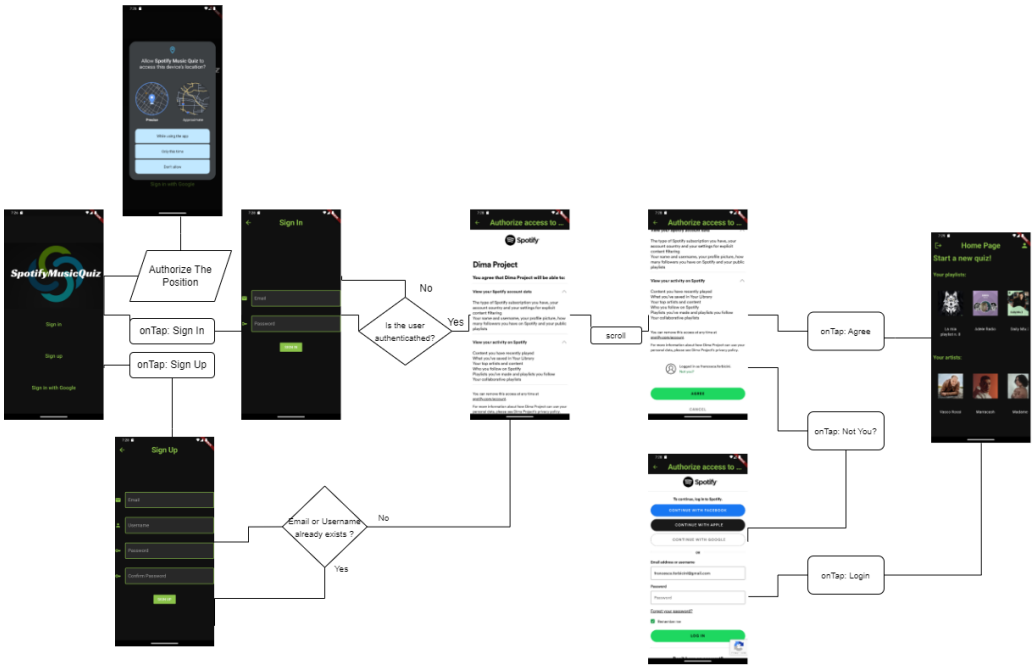
To further personalize the user experience and to make the user feel comfortable using the app, we decided to provide multi-language support via Flutter Intl. Flutter Intl it is an Android Studio plugin that creates a binding between the app and the translations in different languages contained in *.arb* files. It helps the developers in providing multi-language support because it creates boilerplate code for the official Dart Intl library. The only

thing that the developer needs to do is to fill the *.arb* files, one for each supported language, with key-value pairs where the key is used in the code to specify a certain text and the value is the corresponding text that needs to be inserted into the app.

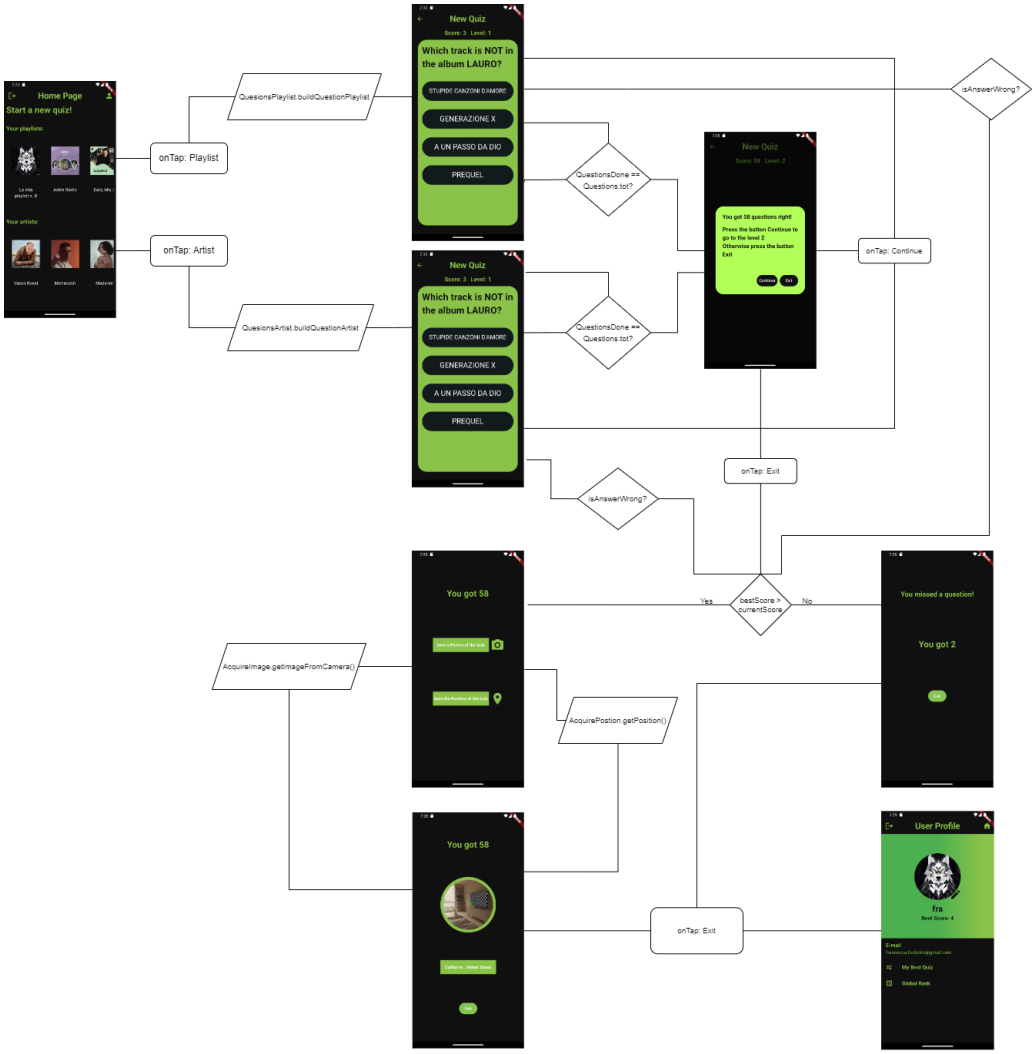
4 User interface design

4.1 UX diagrams

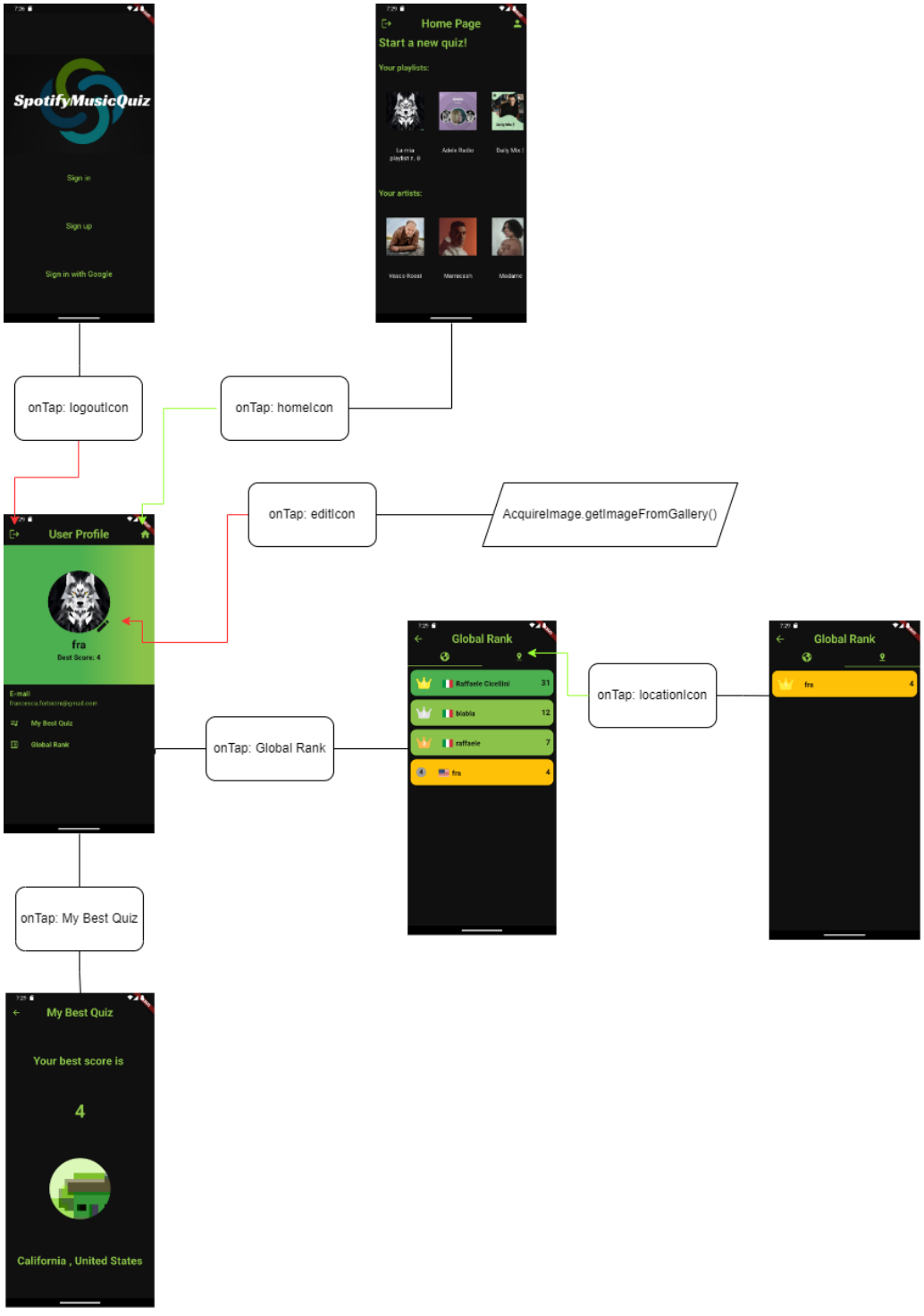
4.1.1 Login and Authorization



4.1.2 Quiz Screen and Result Screen



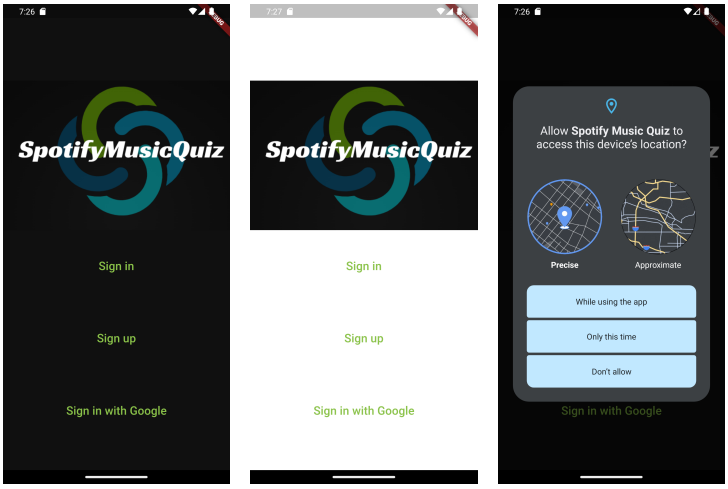
4.1.3 User Profile Screen

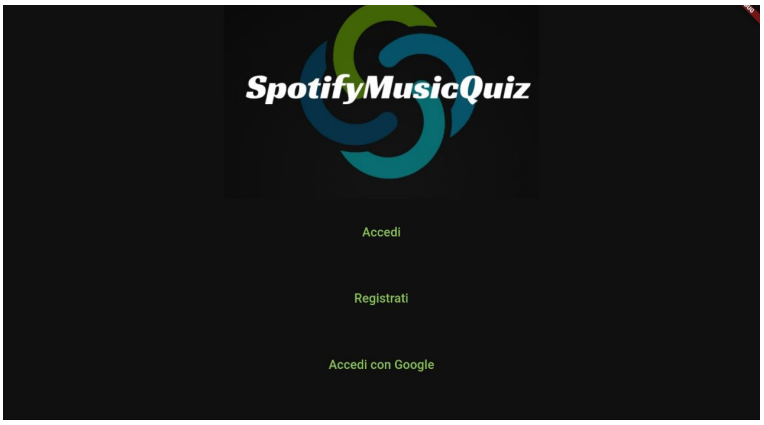


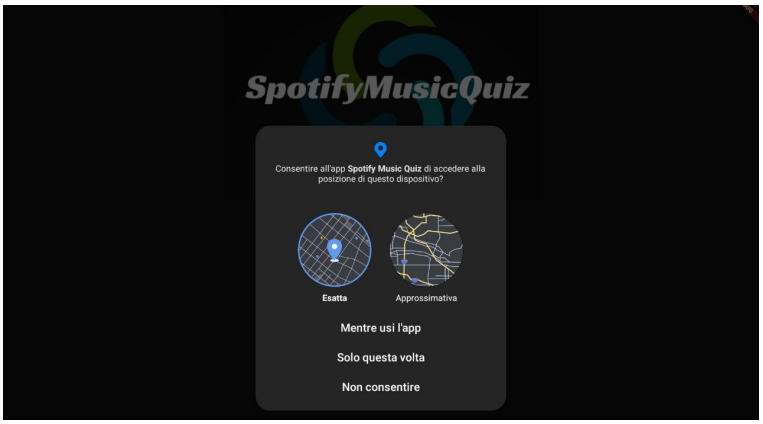
4.2 User interface

In the following paragraphs the screens of the "Spotify Music Quiz" will be shown. In particular, it will be illustrated both dark and light modes for the smartphone with the app in English and tablet screens will be shown in Italian and for simplicity only in dark mode.

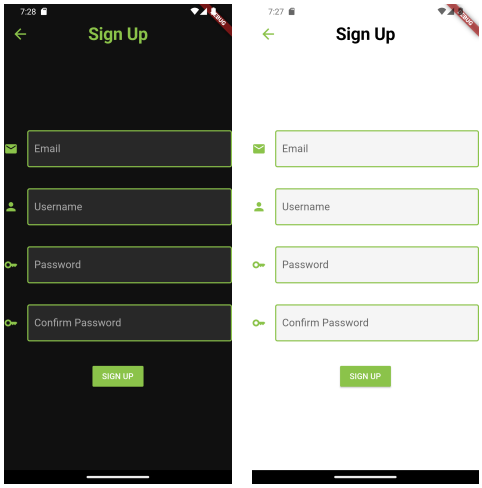
4.2.1 Unauthenticated Screen

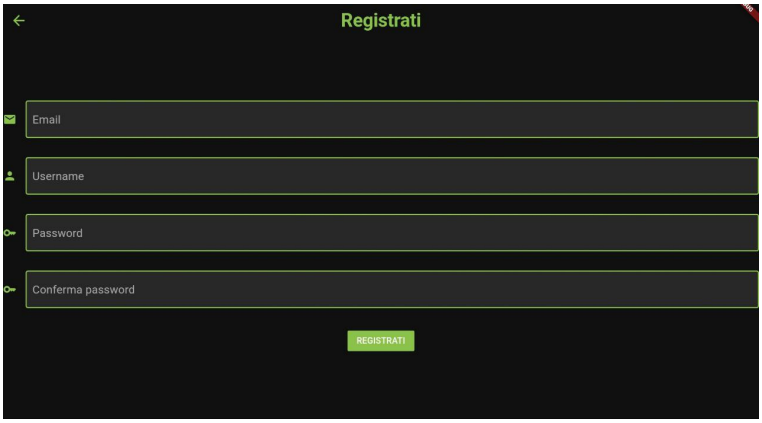



SpotifyMusicQuiz
Accedi
Registrati
Accedi con Google

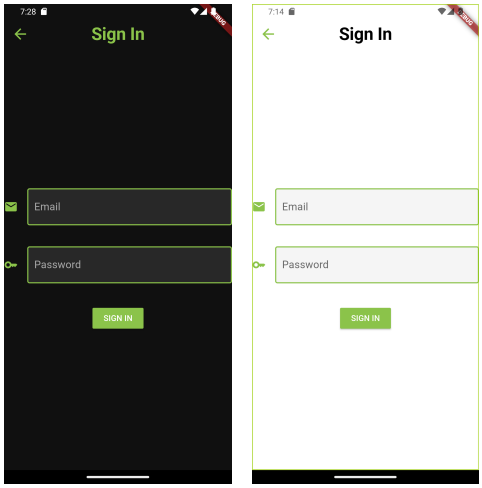

SpotifyMusicQuiz
Consentire all'app Spotify Music Quiz di accedere alla posizione di questo dispositivo?
Esatta
Approssimativa
Mentre usi l'app
Solo questa volta
Non consentire

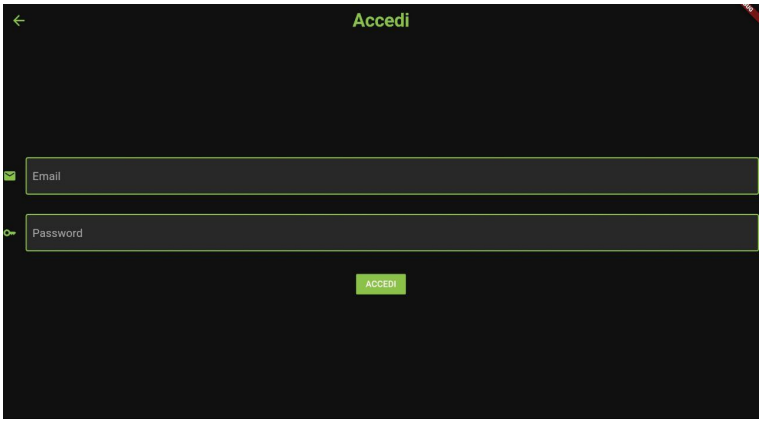
4.2.2 Sign Up Screen



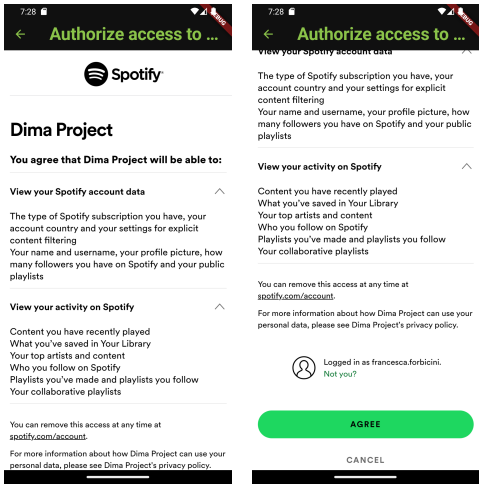


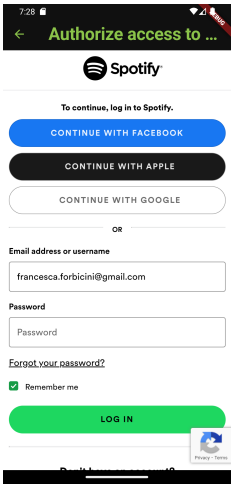
4.2.3 Sign In Screen

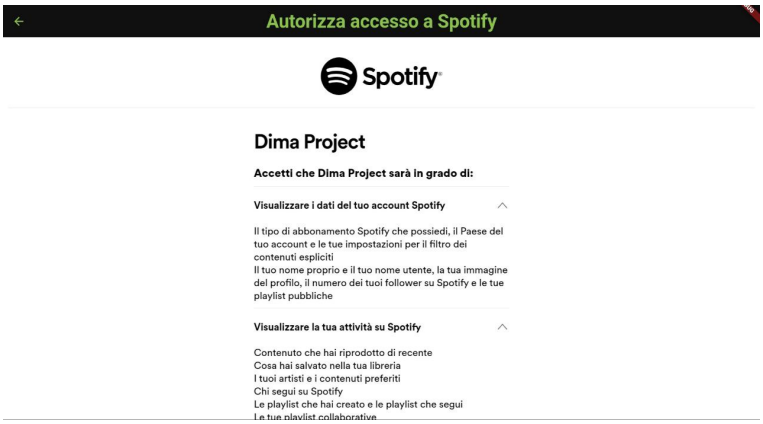


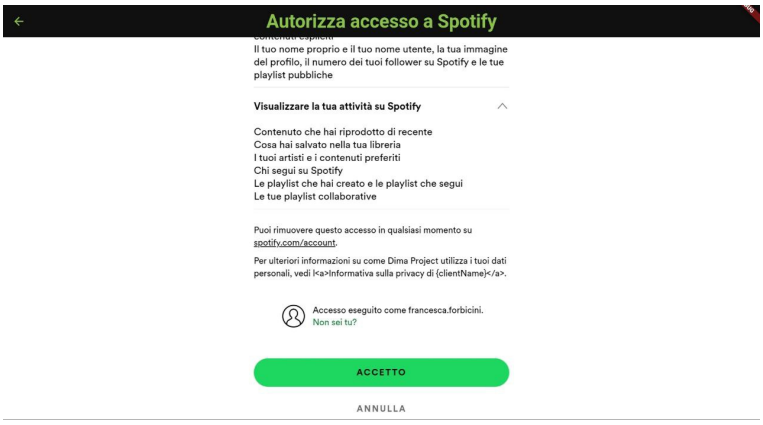


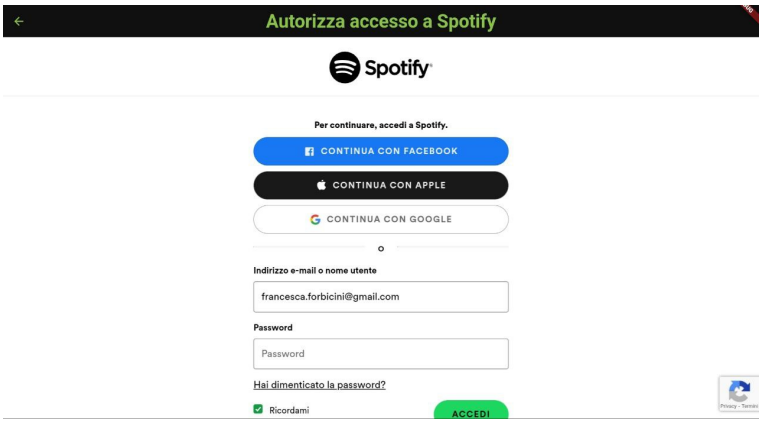
4.2.4 Authorization Spotify Screen



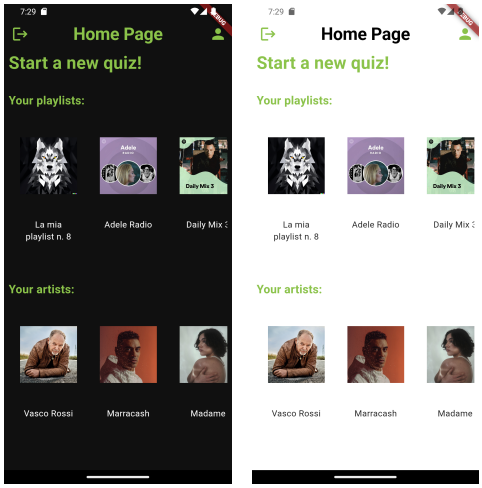


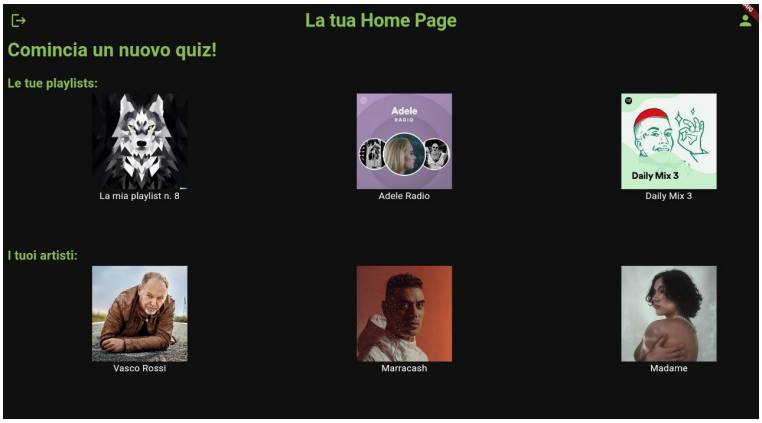

Autorizza accesso a Spotify
Spotify
Dima Project
Accetti che Dima Project sarà in grado di:
Visualizzare i dati del tuo account Spotify
Il tipo di abbonamento Spotify che possiedi, il Paese del tuo account e le tue impostazioni per il filtro dei contenuti espliciti
Il tuo nome proprio e il tuo nome utente, la tua immagine del profilo, il numero dei tuoi follower su Spotify e le tue playlist pubbliche
Visualizzare la tua attività su Spotify
Contenuto che hai riprodotto di recente
Cosa hai salvato nella tua libreria
I tuoi artisti e i contenuti preferiti
Chi segui su Spotify
Le playlist che hai creato e le playlist che segui
Le tue playlist collaborative


Autorizza accesso a Spotify
Il tuo nome proprio e il tuo nome utente, la tua immagine del profilo, il numero dei tuoi follower su Spotify e le tue playlist pubbliche
Visualizzare la tua attività su Spotify
Contenuto che hai riprodotto di recente
Cosa hai salvato nella tua libreria
I tuoi artisti e i contenuti preferiti
Chi segui su Spotify
Le playlist che hai creato e le playlist che segui
Le tue playlist collaborative
Puoi rimuovere questo accesso in qualsiasi momento su spotify.com/account.
Per ulteriori informazioni su come Dima Project utilizza i tuoi dati personali, vedi l<a>Informativa sulla privacy di {clientName}.
Accesso eseguito come francesca.forbicini.
Non sei tu?
ACCETTO
ANNULLA


Autorizza accesso a Spotify
Spotify
Per continuare, accedi a Spotify.
CONTINUA CON FACEBOOK
CONTINUA CON APPLE
CONTINUA CON GOOGLE
o
Indirizzo e-mail o nome utente
francesca.forbicini@gmail.com
Password
Password
Hai dimenticato la password?
Ricordami
ACCEDI

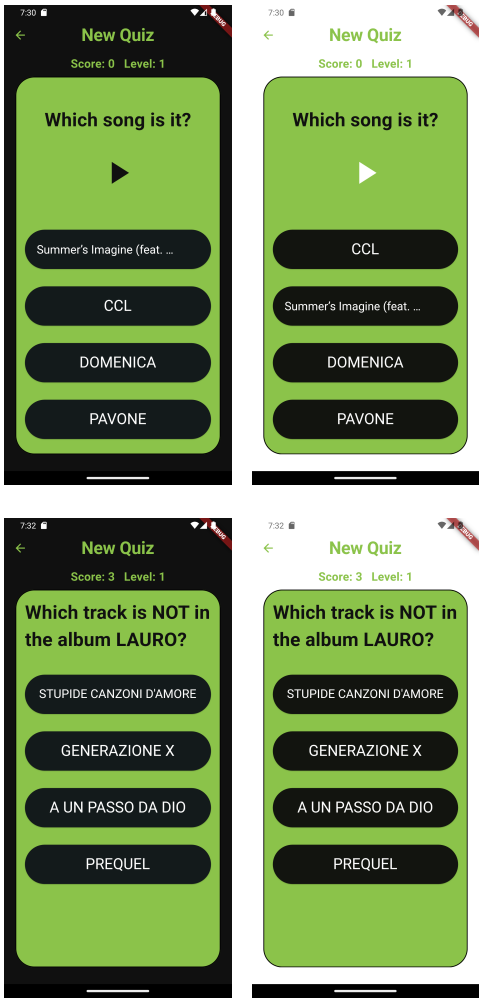
4.2.5 Home Screen

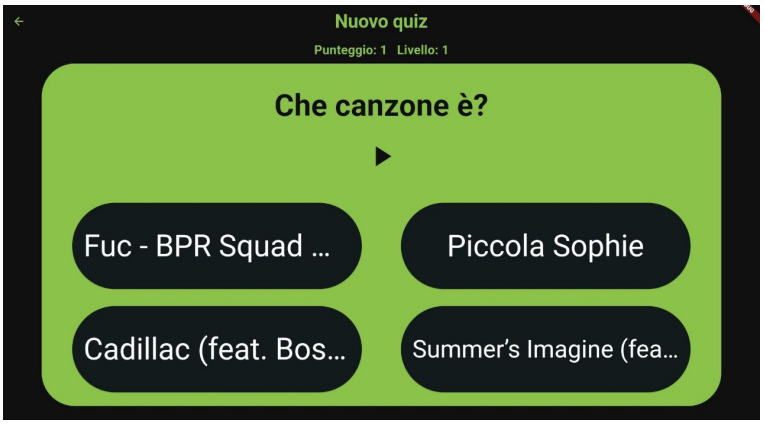


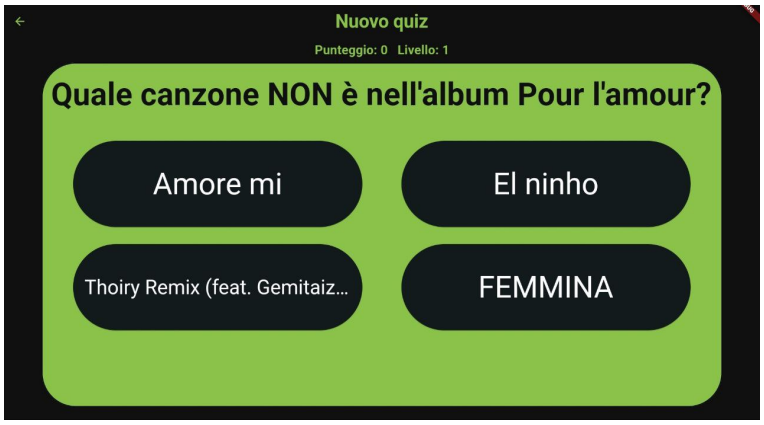


4.2.6 Quiz Screen

There are different types of questions: the first two screens are the dark and the light mode for the type of question where the user by pressing on the 'play button' can produce 30 seconds of a song and guess it.
The seconds two screens are an example of a standard question where the user must select the song that does not belong to a specific album of an artist.

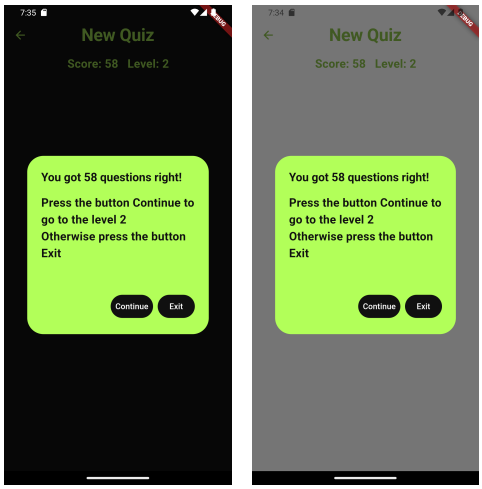


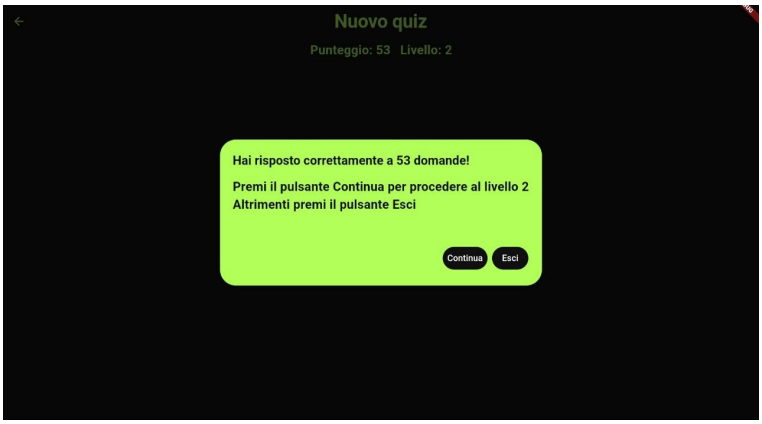

Nuovo quiz
Punteggio: 1 Livello: 1
Che canzone è?
Fuc - BPR Squad ...
Piccola Sophie
Cadillac (feat. Bos...
Summer's Imagine (fea...


Nuovo quiz
Punteggio: 0 Livello: 1
Quale canzone NON è nell'album Pour l'amour?
Amore mi
El ninho
Thoiry Remix (feat. Gemitaiz...
FEMMINA

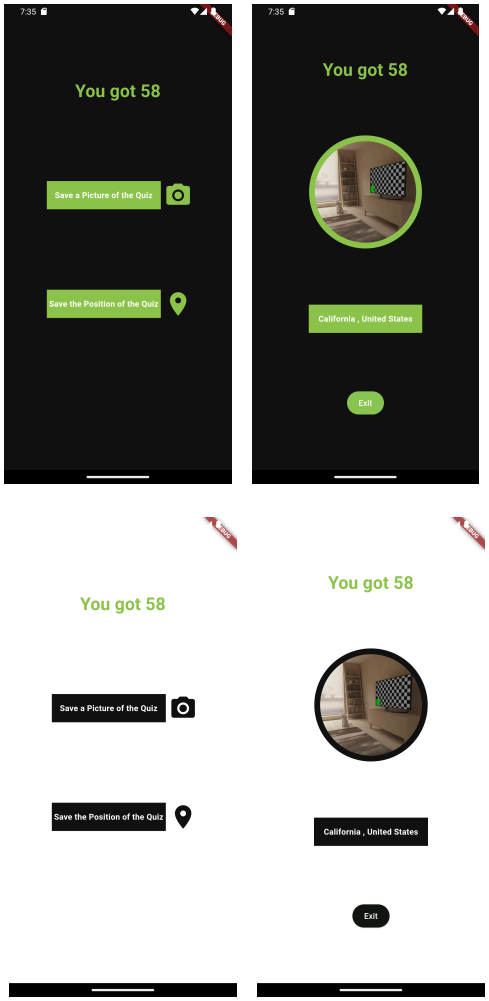
4.2.7 Dialog Screen

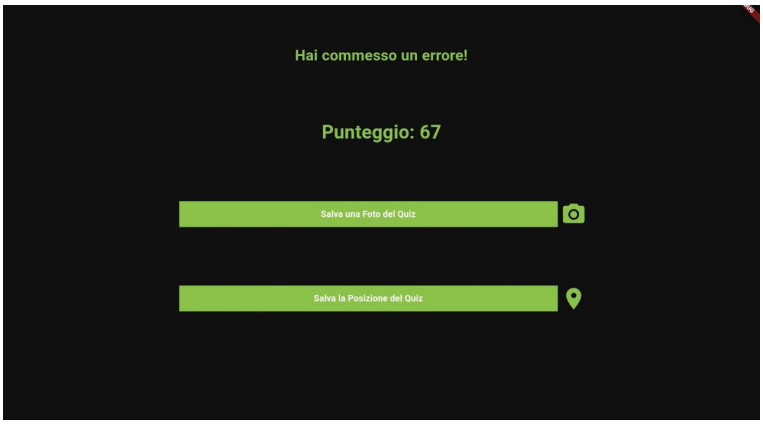
After a certain number of questions, a user can decide to continue with another level of the quiz or to stop the quiz.

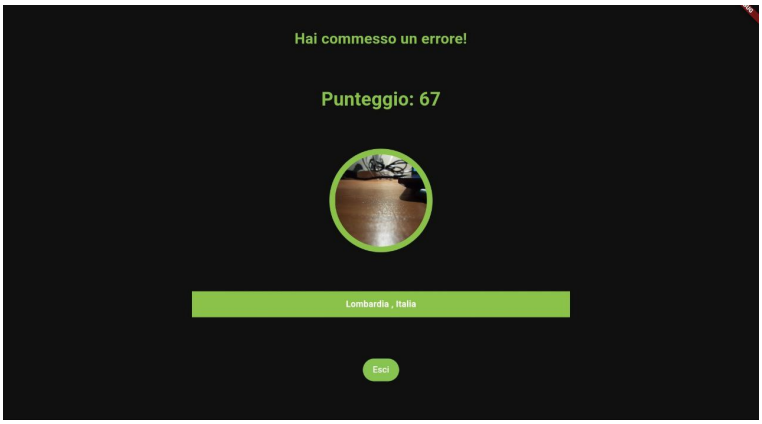




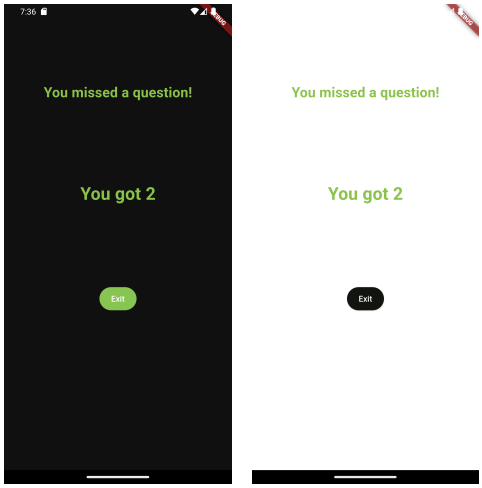
4.2.8 Result Screen for the Best Score

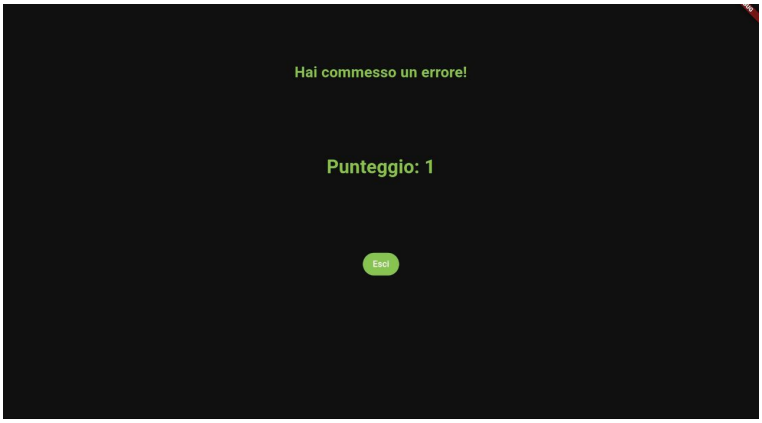



Hai commesso un errore!
Punteggio: 67
Salva una Foto del Quiz
Salva la Posizione del Quiz

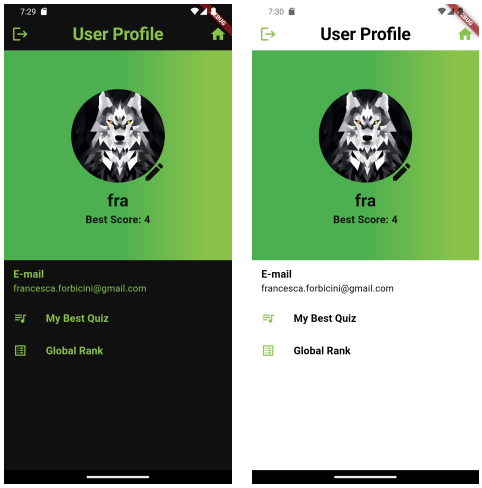

Hai commesso un errore!
Punteggio: 67
Lombardia , Italia
Esci

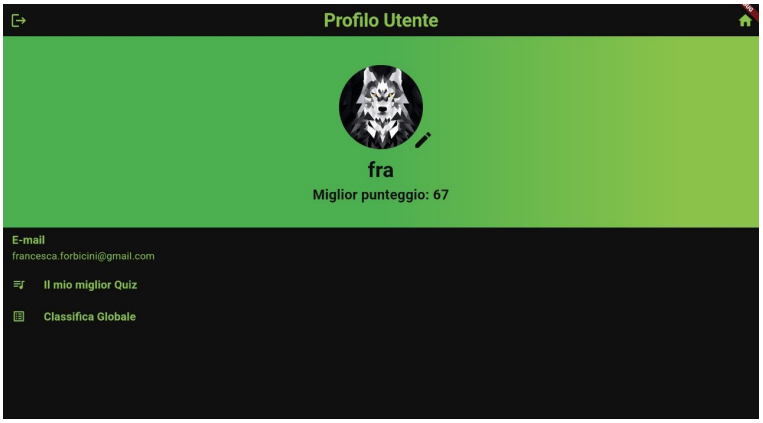
4.2.9 Result Screen



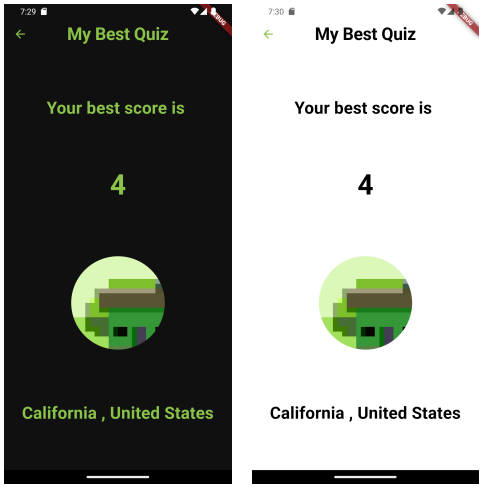


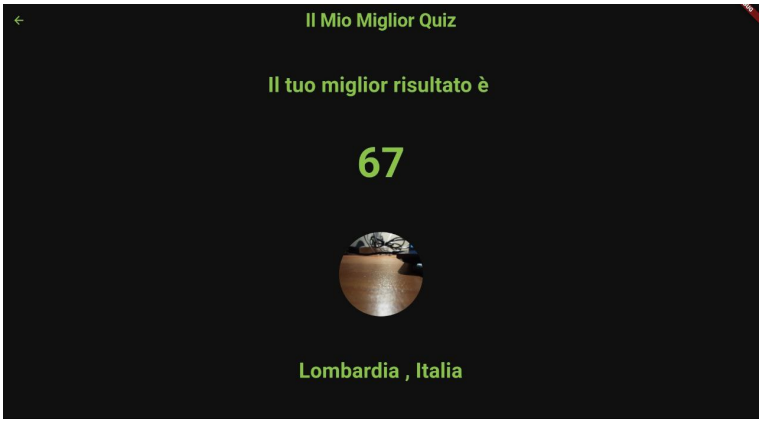
4.2.10 User Profile Screen



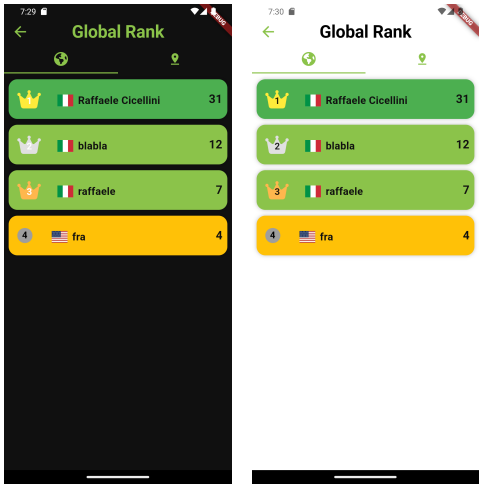


4.2.11 Best Quiz Screen

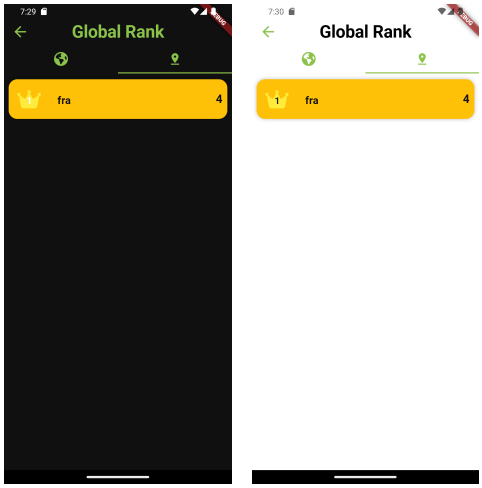




4.2.12 Global Rank Screen



4.2.13 Local Rank Screen



5 Testing

Since the main focus of the app is to make a user play a quiz and view the leaderboard, there is no complex computation to implement through dedicated functions. Thus, we decided to perform no unit testing, but only integration testing.

5.1 Integration testing

The focus of the integration testing campaign was to check whether the widgets behaved as intended during development when used together. We focused also on checking if the navigation routes were doing as intended, so if going back and forth between different screens created problems. The tests were very useful in spotting bugs in some of the screens, for example we found a bug in building questions because sometimes a question didn't have a preview url to play the track, thus raising an exception. We implemented 12 integration tests (divided in different groups) of different functionalities, obtaining an overall line coverage of ca. 80%. Each test is concluded with the sign out procedure in order to execute them all together.

- ***Sign In test***: from the authentication screen, click on "Sign In" button, click on "Username" field and enter it, click on "Password" field and enter it, then click on "Sign In" button. Check if we go in home page (NB: we don't check the authorization phase on Spotify since it is a webview and we assume that the Spotify webpage works. With an if statement, if we are in testing we bypass this process).

- ***Sign In Error test***: from the authentication screen, click on "Sign In" button, click on "Username" field and enter it, click on "Password" field and enter a wrong password, then click on "Sign In" button. Check if there is an alert dialog saying that the username or password were wrong.

- ***Sign Out test***: after login, from the home page click on the "Sign Out" icon. Check whether we are back in authentication screen.

- ***Sign Up Error test***: from the authentication screen, click on "Sign Up" button, click on "Username" field and enter an already present username, click on "Password" field and enter it, enter the same password in "Confirm password" field, click on "E-mail" field and enter a randomically generated one, then click on "Sign Up" button. Check if there is an alert dialog saying that the username is already in use.

- ***Sign Up test***: from the authentication screen, click on "Sign Up" button, click on "Username" field and enter a random username, click

on "Password" field and enter it, enter the same password in "Confirm password" field, click on "E-mail" field and enter a randomically generated one, then click on "Sign Up" button. Check if there is an alert dialog saying that the sign up procedure went well.

- ***Question Playlist test***: after login, from home screen click on the image of one of the playlist to check that it correctly creates a playlist quiz, then click a wrong answer, click on "Exit" button in result page to go back to user profile page.

- ***Question Artist test***: after login, from home screen click on the image of one of the artist to check that it correctly creates an artist quiz, then click on a wrong answer, click on "Exit" button in result page to go back to user profile page.

- ***Bad Result test***: after login, from home screen click on the image of one of the playlist, click on a wrong answer in order to check if it correctly creates the result page since the score was not the best one, so we don't need to save the result. Click on "Exit" button in result page to go back to user profile page.

- ***Best Result test***: after login, from home screen click on the image of one of the playlist, click on a right answer, click on a wrong answer in order to check if it correctly creates the result page since the score was the best one, so we need to save the result. Click on the camera button (NB: we don't test the camera, there is an if statement checking if we are in test, in this case we upload a predefined image), click on the position button. Then, click on "Exit" button in result page to go back to user profile page.

- ***Back to Home test***: after login, from the home screen click on "User Profile" icon. After a while, click on the "Home" icon and check if we correctly go back to the home page screen.

- ***Best Quiz test***: after login, from the home screen click on "User Profile" icon, click on "Best Quiz" tile, check if we are in the best quiz page. Then, click on the "Back" icon and check if we correctly go back to the user profile page.

- ***Rank test***: after login, from the home screen click on "User Profile" icon, click on "Rank" tile, check if we are in the rank page. Click on "Local Rank" tab icon and check it is the correct page, click on "Global Rank" tab icon and check it is the correct page. Then, click on the "Back" icon and check if we correctly go back to the user profile page.

6 References

- Flutter documentation
- Firebase documentation
- Spotify Web API documentation
- Android Studio documentation
- Stack Overflow
- draw.io (for the diagrams)