uc3m | Universidad Carlos III de Madrid

Dual Bachelor in Data Science and Telecommunications Technologies Engineering 2020-2025

Final Delivery

Mobile Applications

Alejo González García (100454351)

Daniel Toribio Bruna (100454242)

Andrés Navarro Pedregal (100451730)

Table of Contents

1. Introduction

This project consists of a tracking app for runners and cyclers. The main focus is to allow users to track their runs and paths that they do when exercising. And also have the ability to see the current weather to be able to gear up correctly for the weather.

2. Work Distribution

This project was divided into 3 different big areas:

Login, database handling, and main view

We had to be able to sign in users, load and retreive the data from Firestore and also the proper handling of models for the different types of data saved. And also the main page with the dynamic load of runs.

For this the layouts developed were the splash screen, the login, screen, and the main view screen. And for the functionality, it was the ability to sign in with Google and annonymously, proper handing of the database and modeling of the Data classes to properly load the data for each user. And also the development of the RecyclerView to dynamically load the different runs and update accordinly when new runs were added.

Tracking

For the tracking, the main point was to create a background service that was able to log the different GPS locations and calculate the distance traveled. Also a chronometer had to be created in order to track the time.

Moreover a map view was developed to see the route that the user took for a given run. For this the proper handling of the GoogleMaps API was to be establied and the ability to draw paths on the map.

Therefore the main layouts were the tracking layout and the map view layout.

Weather and permissions

For the weather, the main point was to be able to retreive information from external services, in our case the “OpenWeatherAPI”. For this we made use of third-party libraries to retreive the information from the internet.

Also as we wanted to the the current location and city, we used the GoogleMaps API to get the city from a set of coordinates and also the proper handling of the permissions to access the location and the internet.

Workload

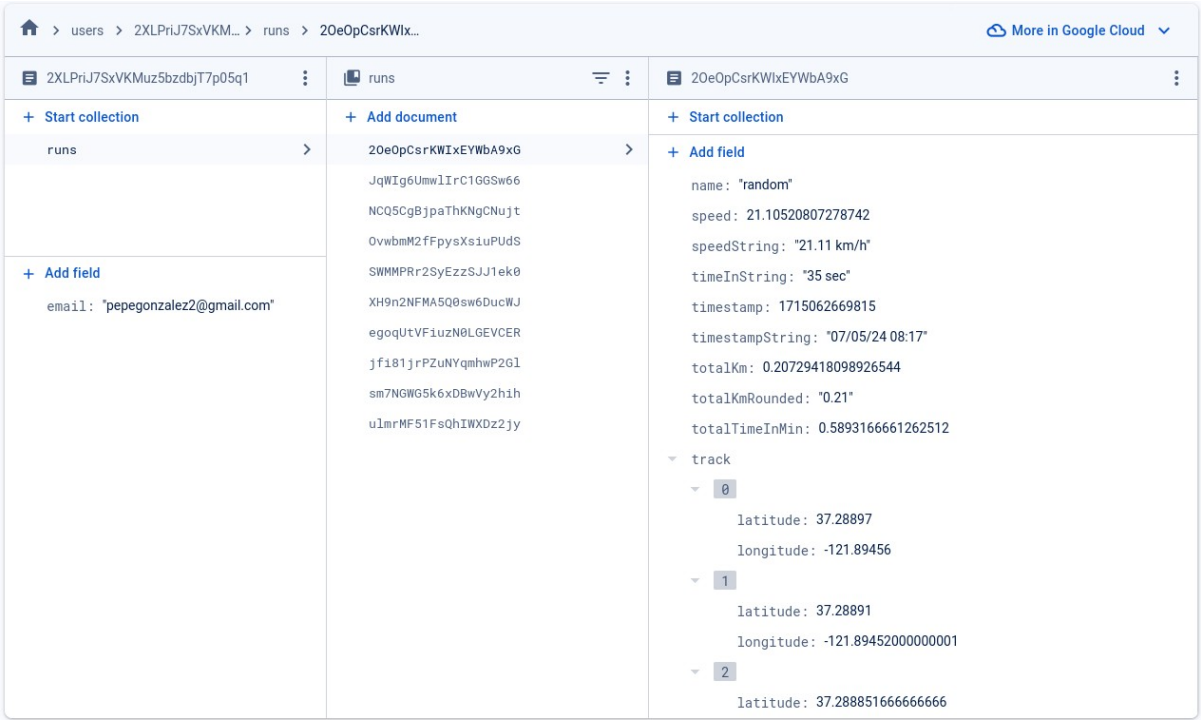
As we had 3 main areas to work on and we were 3 people in the team. Each one of use did one main area and then we got together to integrate the work done. In the end we found out that with the use of collaborating tools such as GitHub we were able to work in parallel and successfully develop a Android app.

All in all, this approach ended up to be successfully as the workload was fairly well balanced. Moreover, we were able to help each other out as debug in parallel with the use of git branches.

3. Main Features

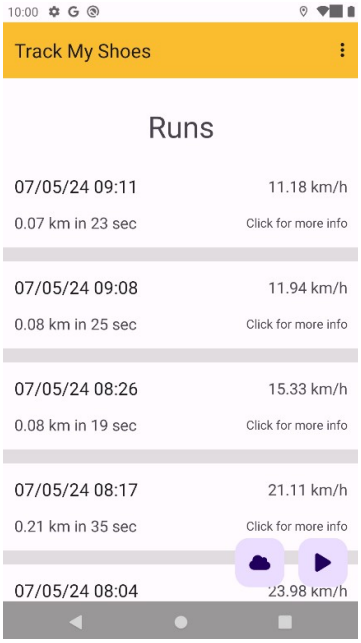
Persistent storage and user authentication

In order to be able for users to use the app we used Firebase Auth with the Google login. And to store the data we used the Firestore Database from Firebase. In order to link the two together we use the identifier that Firebase Auth assigned for every account and link to a document collection in the Firestore Database. Here you can see a screenshot of the data saved for each user.



To load and retreive the data, we made use of Java Classes and the built in feature to map data from firestore to a Java Class. The main focus was on the retreival and save of the location data used to recreate the route of the users.

In order to show each run, we used a recylcer layout with a custom adapter to be able to load the data. You can see the main view of the app and the different runs of the user

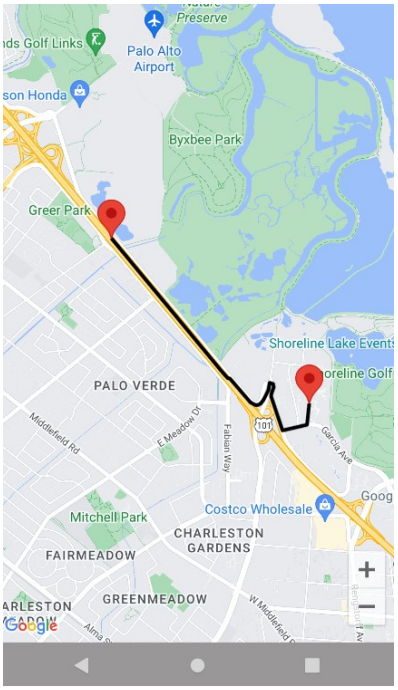


Maps, location services, and background services

For the tracking, the main point was to create a background service that was able to log the different GPS locations and calculate the distance traveled. Moreover, we use the GoogleMaps API to create a map with the route the user took for a run.

The main effort here was to develop the ability to get the live location of the user and create the background service to do so.

Here you can see a route a user took and the path drawn in the map:



External Services

For the weather, the main point was to be able to retreive information from external services, in our case the “OpenWeatherAPI”. For this we made use of third-party libraries to retreive the information from the internet. Here you can see a screenshot of the weather view:

4. Secondary Features

For the secondary features, we used different third-party libraries such as the Retrofit or the Picasso libraries to get infromation from external APIs, parse JSON data and retreive images from the internet.

Moreover, we had to create a stopwatch in order to track the time it takes for each run. For this we used the system time to calculate the difference between the times.

Also, we used collaboartion tools such as GitHub to be able to work in parallel and track our progress along the development phase.

Finally, we use extensivly custom callbacks in order to interact with the different custom services inside our application and to properly handle the interactions. And example can be seen in the image below:

```
private static void getWeatherFromCity(String city, final WeatherCallback callback) {
    if (city.isEmpty()) {
        callback.onFailed( errorMessage: "Cannot detect the city");
    }

    Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(OPEN_WEATHER)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

    WeatherApi weatherApi= retrofit.create(WeatherApi.class);
    Call<WeatherModel> weather = weatherApi.getWeather(API_KEY, city);

    Andres Navarro
    weather.enqueue(new Callback<WeatherModel>() {
        Andres Navarro
        @Override
        public void onResponse(Call<WeatherModel> call, Response<WeatherModel> response) {
            callback.onSuccess(response.body());
        }

        Andres Navarro
        @Override
        public void onFailure(Call<WeatherModel> call, Throwable t) {
            callback.onFailed( errorMessage: "Error fetching weather");
        }
    });

}
```

5. Links

Link for the Goole Drive Folder: https://drive.google.com/drive/folders/13xg2-5PFV-lJ56Or4S2h2b_LM7hSEL-u?usp=sharing

Link for the Youtube Video: https://youtu.be/fUjG3rfh0Tk

6. Summary And Outlook

All in all, we were able to successfully create an android app from scratch that people can use. With a focus on user authentication, database management, real-time tracking, and weather integration, the app provides a comprehensive experience for fitness enthusiasts. Through collaboration and effective use of tools like Firebase, Google Maps API, and third-party libraries, we were able to achieve a balanced workload and smooth integration of features. The app enables users to track their routes, view real-time weather updates, and analyze their workout sessions efficiently.

Looking ahead, we think of expanding the app's capabilities to enhance user experience further. Future plans involve refining the user interface for better accessibility, optimizing performance to reduce battery consumption during tracking, and adding more features like social sharing, leaderboards, and personalized training insights.

7. References

- Firebase Documentation: https://firebase.google.com/docs
- Google Maps Platform Documentation: https://developers.google.com/maps/documentation
- OpenWeatherAPI Documentation: https://openweathermap.org/api
- Retrofit Documentation: https://square.github.io/retrofit/
- Picasso Library Documentation: https://square.github.io/picasso/
- Class notes