Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

# HOME FAR FROM HOME

## INDEX



**Link to the application code:**

https://drive.google.com/file/d/1x12Jirrg1_991wl58fcTXGDNZsHdc7gG/view?usp=share_link

**Link to the youtube code:**

https://youtu.be/qiyqdYfw_Zs

Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

The aim of our application is to join different things of interest for people who move to Madrid form other cities or countries.

In the first view of our app, we find the main view. Here you can choose between different options such as log in, sign up or enter as an invited. These functionalities will be later explained. Moreover, you can navigate through all the proposed events, view them in a map, save them as favourite or check the directions to arrive there.

Finally, for users who are logged in we have available a profile view, where they can check for their favourite events, remove those events from favourites or even log out.

## THE COMPONENTS

For analysing the different components of our application, we are going to navigate through the application to get to all the details.

However, we wanted to put attention in some general aspects of our application:

- We have created all the layouts with a clean and proper format, making use of linear layouts and their different properties.
- All the strings have been stored in the values folder in order of having the possibility of translating them and having a clean and proper code.
- We have connected our application with Google Cloud Platform and obtained an API token for using google maps services. Moreover, we have added the google play services dependency to the Gradle. Finally, we have declared the API in the manifest and ask for permissions for using the location (ACCESS_FINE_LOCATION and ACCESS_COARSE_LOCATION).

  Thanks to this process we managed to add maps and directions into our application.
- Every event has the possibility to be added to favourites (if there is a user logged in and that event is not already in his favourites) will be notify to the user so that he does not forget the event he wants to go. In addition to that our app if full of toast messages to inform the user when he is doing something wrong (for example when inserting the name and information in the sign up activity) or just helpful messages.
- We offer the possibility of having the whole app in Spanish and English so that everyone can use it.

Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

- We have created a local database. A model has been created so that we have organized and well-managed data. Our database will be a preloaded one (SQLite) of this form:

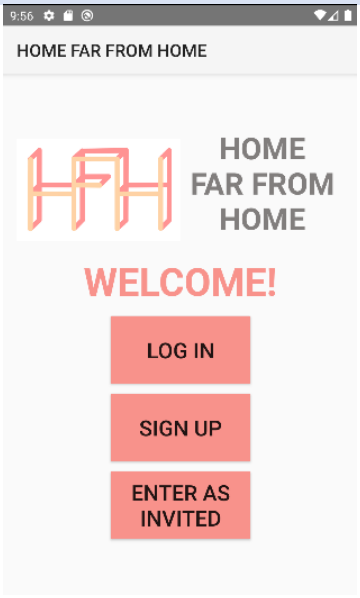
| Name | Type | Schema |
|---|---|---|
| Tables (4) | | |
| Event | | CREATE TABLE "Event" ( "ID" INTEGER NOT NULL UNIQUE, "TITLE" TEXT, "LONGITUD" REAL, "LAT |
| ID | INTEGER | "ID" INTEGER NOT NULL UNIQUE |
| TITLE | TEXT | "TITLE" TEXT |
| LONGITUD | REAL | "LONGITUD" REAL |
| LATITUD | REAL | "LATITUD" REAL |
| DESCRIPCION | TEXT | "DESCRIPCION" TEXT |
| IMAGE | TEXT | "IMAGE" TEXT |
| PREFERENCE | TEXT | "PREFERENCE" TEXT |
| DATE | INTEGER | "DATE" INTEGER NOT NULL |
| Favourite | | CREATE TABLE "Favourite" ( "ID" INTEGER NOT NULL UNIQUE, "EVENT_ID" INTEGER NOT NULL, "L |
| ID | INTEGER | "ID" INTEGER NOT NULL UNIQUE |
| EVENT_ID | INTEGER | "EVENT_ID" INTEGER NOT NULL |
| USER_ID | INTEGER | "USER_ID" INTEGER NOT NULL |
| PREFERENCE | TEXT | "PREFERENCE" TEXT |
| User | | CREATE TABLE "User" ( "ID" INTEGER NOT NULL UNIQUE, "NAME" TEXT, "EMAIL" TEXT, "PASSWOI |
| ID | INTEGER | "ID" INTEGER NOT NULL UNIQUE |
| NAME | TEXT | "NAME" TEXT |
| EMAIL | TEXT | "EMAIL" TEXT |
| PASSWORD | TEXT | "PASSWORD" TEXT |
| LANGUAGE | TEXT | "LANGUAGE" TEXT |
| PREFERENCE | TEXT | "PREFERENCE" TEXT |
| TELEFONO | TEXT | "TELEFONO" TEXT |

Where we could observe that Favourite table had two foreign keys, the event id and the user id, that way every favourite that the user keeps is linked to him and the event he is referring to. Every column of each table ha a function, for example *latitude* and *longitude* was to store the coordinates of each event so then it would be easier to show them in the map. As well as the *preference* in all of the tables which was just to link those users with its preferences when looking for events to go.

Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

## MAIN VIEW

This layout created in xml and provided with its correspondent java activity, has been created with the aim of directing users to our functionalities. Here we can find three buttons created on the xml and provided of an ID. This is important since we use this ID to identify which button is clicked in the java class. One we identify which button is clicked, with the help of *Listeners* and *Intents* we navigate to the selected functionality.

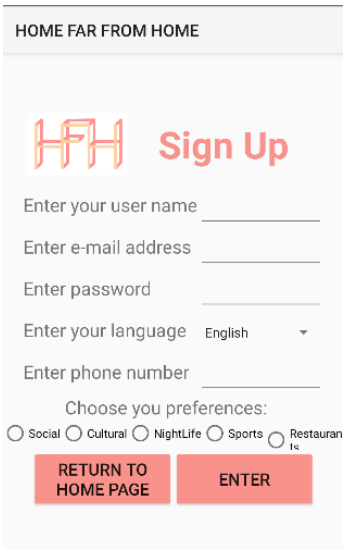



## SIGN UP VIEW

This activity is designed to allow new users to register in our system. Firstly, we create a database instance in the activity so we can add new users to our database. Secondly, we provided all text views with the proper IDs to recover the information inserted by the user. As well, we created a radio button group for selecting the preferences and a spinner for choosing the language.

Finally, we created the logic for the sign-up process. We add several constraints for being sure that when the user fills the different fields, things have the proper format (e.g., emails, have an email format, phone number has 6 digits. Moreover, we add more constraints with the help of queries in order of assuring that there are not repeated mails or usernames. In case some of these things aren't accomplished, a toast message is sent advising the user of what things should change.

To finish with, when all conditions are accomplished, user is registered in a database with the help of a query. A toast message is sent to the user, and it is redirected to the events view.

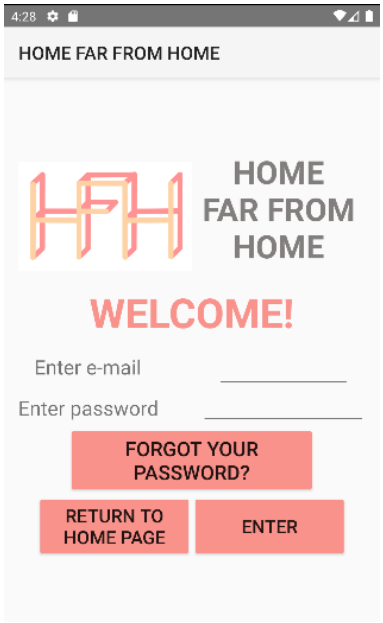


Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

## LOG IN

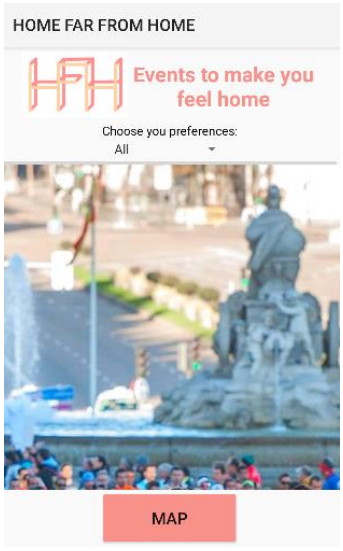
This section of the application is created with the goal of allowing users to get into the app as a registered user. As in previous layouts, all text views are created with the proper IDs so we can recover the information of the user. When we get the information, we perform a query for checking if there is a user registered with that email in our database. On the one hand, if there is not, a toast message is shown. On the other hand, if it is, we recover that user and check if the password coincides. In case it does, user is redirected to the events view.

A remarkable aspect is that we have created a shared preferences object with the aim of checking if there is a logged in user. By defaults it is created as false and when the log in is performed we change it to true.




## EVENTS VIEW

In this view, we have created this view with two perspectives depending on if you are logged in or not.




Enter as an invited

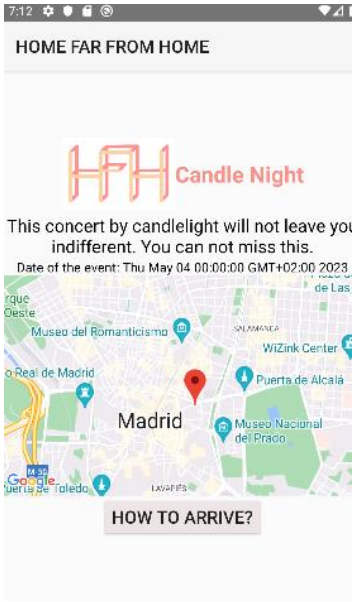

User is logged in

Marta Almagro Fuello 100451979
Gracia Estrán Buyo 100452014
Marta Balairón García 100451724

With the help of the shared preferences object, we detect if the user enter as invited or logged in. In case it is logged in, we recover the user from the database (the username is sent from the log in or sign up as an extra of the intent). On the one hand, in case the user is logged in, the profile button will be shown only in this case at it requires a user to get the user's name later for the profile view, where the user will be able to navigate to the profile view later explained or add events to favourites. On the other hand, if the user is not logged in, the app will find the "profile" button and set its visibility to "GONE", so that it is not visible on the screen if there is not a user logged in.

In this page we can find a scroll down view in which we can find image buttons for all the events. As a first step, we recover all the events from the database with a query and with the help of a loop we add all of them in button image format. When we click on them, the event view is displayed with help of an intent. We pass through the intent as an extra the event title and in the event view, we recover it and obtain the event object view querying it in the database. Moreover, we can find a spinner for choosing a filter for selecting a preference. When it is selected, just the events matching with that category are shown.

In the event view once we recover the event from the database, we post the event description and its location in the map. Moreover, if the user is logged in, a button for adding to favourite the events is added.




Enter as an invited


User is logged in

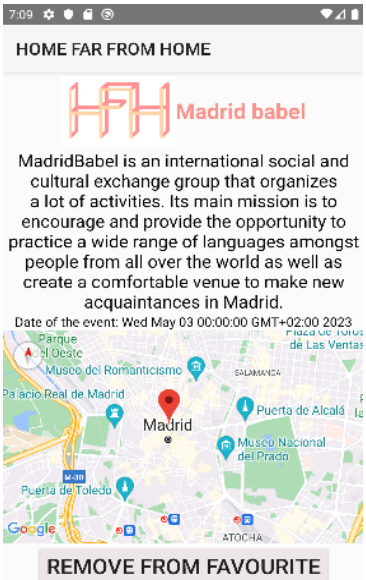
When the user clicks in the button that says *How to arrive?* It shows him the best route of how to get to that event.


## MAP VIEW

As its name says, this view will be where the map is displayed, with all the events represented where they are. This is thanks to the fact that before we have stored the latitude and longitude of each event in the database. We used the Google Cloud services and created and an API of Google Maps. In the code we used the library android.location we achieve to make the map and show all the different events that are around the city. In addition, we also import the library google.android.gms for all the directions and paths to go from our location to the event marked. As well as in the events activity you can filter in case you want to change your preferences. When you click in each pin, a tag with the name of the event appears.

## PROFILE VIEW

The profile view was created with the intention of creating a space where the user could se his favorites events and manage the language or if he wanted to log out of his account. From this view the user can click in the events and delete them. In case he did not have any event marked as favorite gives the option to go to the main view and see some events to add. Which can be observe in the following image:




This view was using the event view too, but with help of a *sharedpreferences* object we could check where the user was coming from, so when he came from the profile it

meant that the event was in his favorites list and that we should show the delete button instead of the add button. Then when that button is clicked it will just call a function from our DAO for that table and delete it.

## CONSIDERATIONS TO EXECUTE THE APP

This app has been tested during all the project with the android emulator, whose device definition was pixel 2 size 5.0 and a system image version Android 10.0. In order to use successfully the location service with configured the location of the device in Madrid so we could see the path correctly.

As our mobile application is Multilanguage the language can be changed in the whole app from the setting of the phone too. We tried to also add the option to change it from the app, but at the end we saw with our teacher that it was not possible. So what we did was storing all text displayed in variables in *strings* and after we translated them all to Spanish, with the tools offered by Android Studio. This way, if you change the language in the phone, it is changed also in the app.

Finally, as we have used an API, there must be in the local.properties file a line that says: MAPS_API_KEY=AIzaSyDP6RrHegrC-trOyTajve7V-QqZewS9_F0 which is the API used to connect with google maps services.

## DESCRIPTION OF THE RESPONSIBILITIES OF EACH GROUP MEMBER.

During the executing of the project we have observed that even though every part was related with each other, it was easy to divide different tasks so each of us could focus on learning and, with help of the teacher examples, understanding the code and investigate for possible implementations. At the beginning we were the three of us planning all the ideas we had in our head and that we wanted to include in the mobile application. This part was crucial because we wanted to do many things, but we did not know what was really the order, so we figured out that before starting we must have our data model, so we take it from there.

Afterwards each of us focused on different parts, Marta A started creating the database and implement it to the code. Marta B started to coding the user logging in and signing up, so when the database was loaded we could start testing the user experience. Meanwhile Gracia started the layout configuration and the general schema to the application. Once this parts

where done, we all worked it to join them and fix all the bugs that started appearing when we put it together. Then we repeated the process of dividing the tasks again, but this time Marta B was taking care of the maps and location situation, Gracia of the Multilanguage stuff and Marta A the notifications. Although each of us had a “task” we worked together and made sure that the other two understood how it worked. Finally, it came the moment of joining all parts and repairing bugs, there were sometimes that we would get stuck because it is the first time we code with Java, but thanks to the teacher he would guide us into getting with the answer and learning more about how java and Android Studio work.

We started this project using Github and updating the project every time someone changed something, however when two people were working in the same file and one of them push it and commit it, the changes of the other one would be overwritten. So we changed this strategy and just send every last version, so the next one would paste her code in the good version.

To sum up, we have enjoyed this project more than we could imagine because it let us get creative and learn the main things that are behind a mobile application. It was really satisfying when after many tries we finally see what we wanted to achieve in our simulator and understand what we were doing wrong for next steps in the project. Also, learning how the programming language Java works and the different ways it uses when creating objects or calling variables. We have also enjoyed working with Android Studio as it has many tools that are really helpful, as being able to previsualize a layout before running the emulator or the facilities it offers to see exactly where an error is.