Favorite Places Documentation

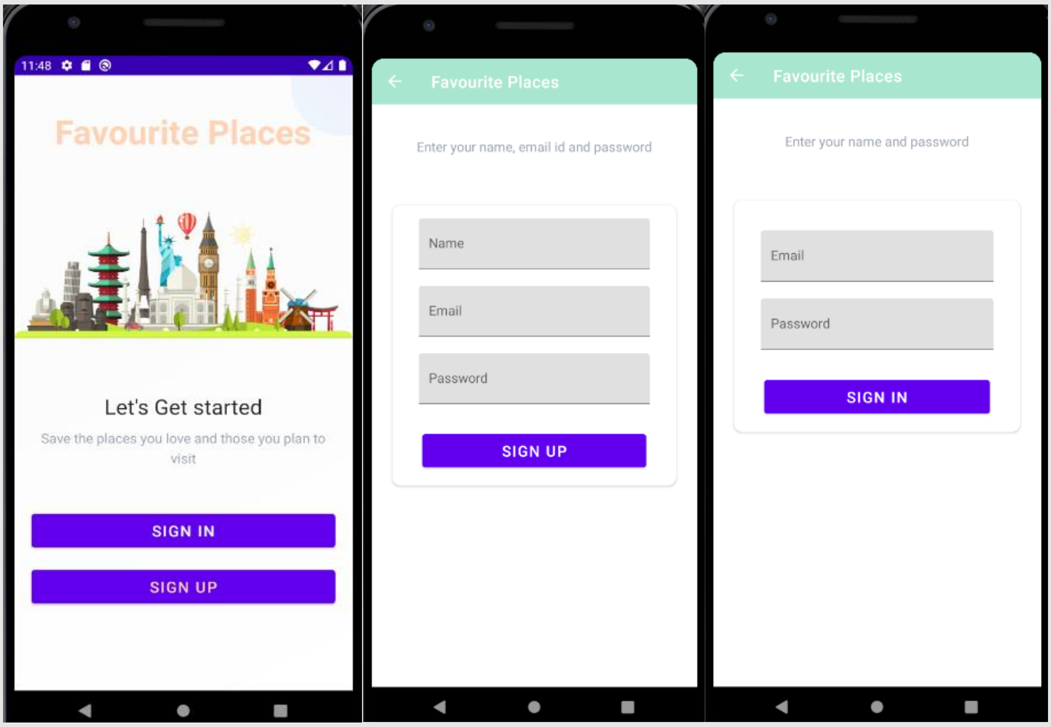
Introduction

The application's purpose is to keep track of user's favorite places. Often people want to remember a nice place that they visit so that later they can share it with friends or revisit. This app serves that purpose. A user can choose a place they visited earlier and save it after adding details and comments as well as a picture from their gallery, or alternatively they could take a new picture. Moreover, users can choose the option of getting their current location while adding a new place. Functionality for managing the places as well as user authentication is also in place.

Features

- **Authentication**

Once the app starts the user is met with a welcome screen where they can choose to sign in if they have an account or sign up if they do not. Each option opens a new screen with input fields for introducing data. The input is checked and only if it is accurate it will be accepted. Firebase is used for authentication. As soon as a user signs up their login details are saved in firebase and they can sign up with them later on.



Save the places you love and those you plan to visit

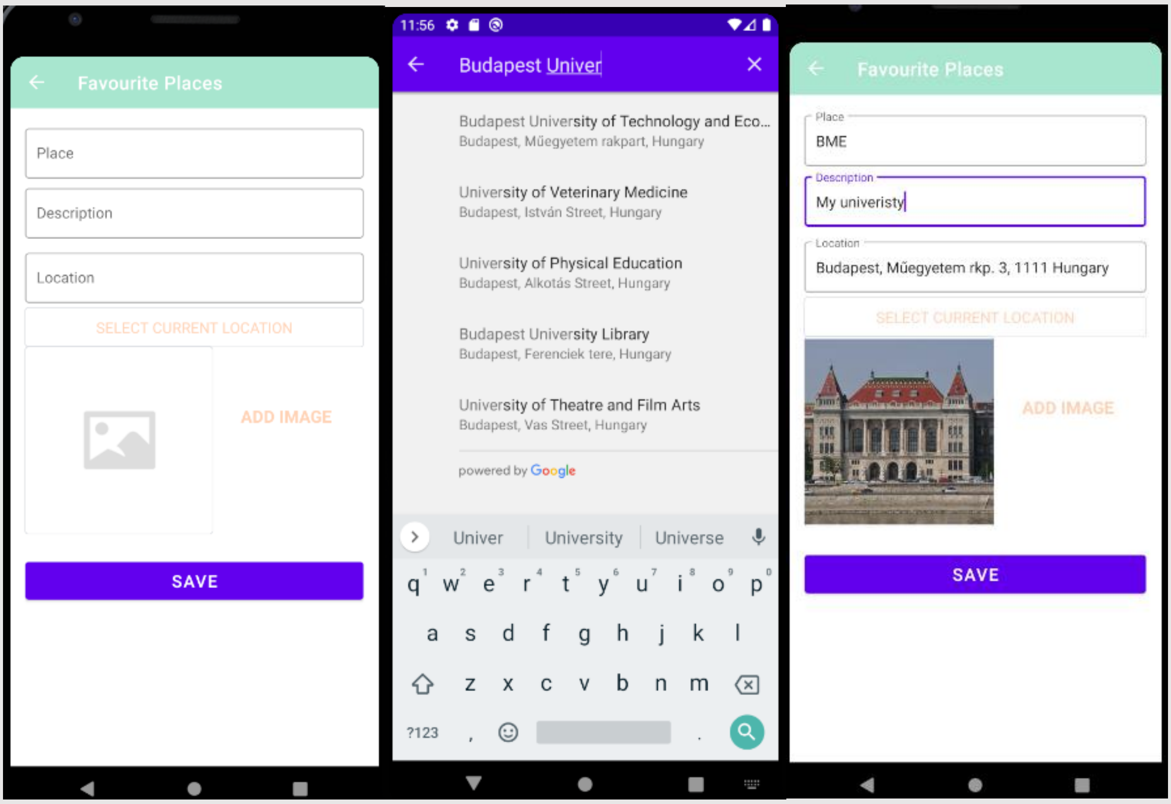
Enter your name, email id and password

Enter your name and password

- **Add new place**

Once the user chooses to add a new place they will have to pick a name for the place, a description and the location. For location picking I used google places API, which contains millions of places and thus the user could find almost any place they would want to save.

When searching for a place they will get suggestions based on what they are typing, like in the screenshot below.



11:56

Budapest Univer

Budapest University of Technology and Eco...
Budapest, Műegyetem rakpart, Hungary

University of Veterinary Medicine
Budapest, István Street, Hungary

University of Physical Education
Budapest, Alkotás Street, Hungary

Budapest University Library
Budapest, Ferenciek tere, Hungary

University of Theatre and Film Arts
Budapest, Vas Street, Hungary

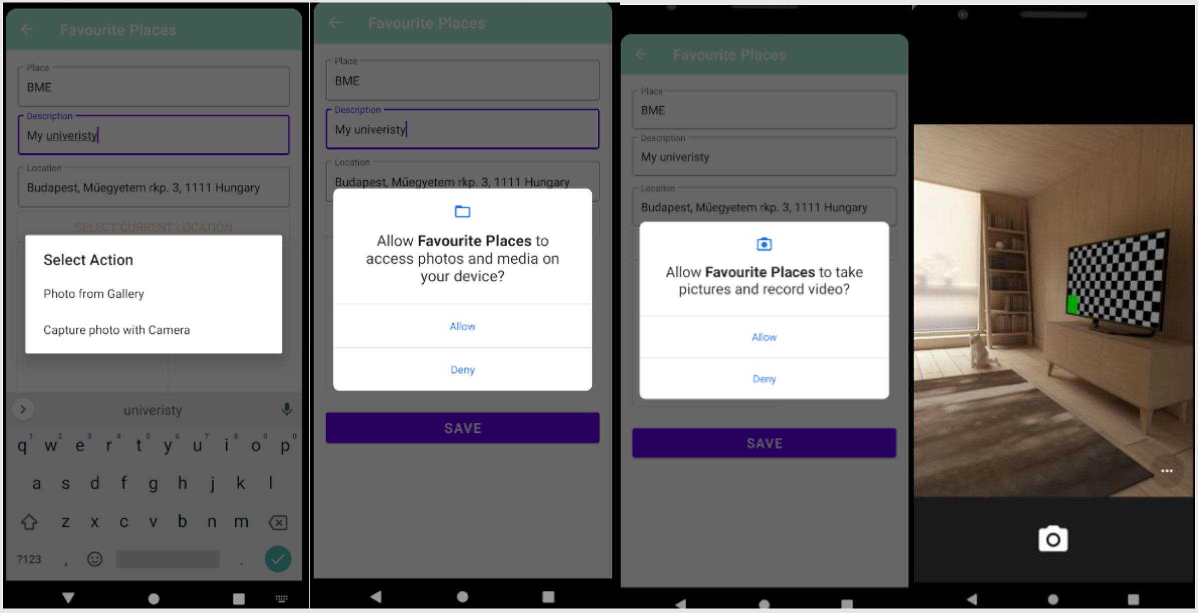
powered by Google

Place: BME

Description: My univeristy

Location: Budapest, Műegyetem rkp. 3, 1111 Hungary

As a last step a picture for the place needs to be picked, when clicking “Add image” a popup will appear asking whether the place should be picked from the gallery or whether a new picture needs to be taken, in which case the camera will open. In order to handle camera and gallery permissions I used the library ‘dexter’.



Place: BME

Description: My univeristy

Location: Budapest, Műegyetem rkp. 3, 1111 Hungary

Select Action

Photo from Gallery

Capture photo with Camera

Allow **Favourite Places** to access photos and media on your device?

Allow

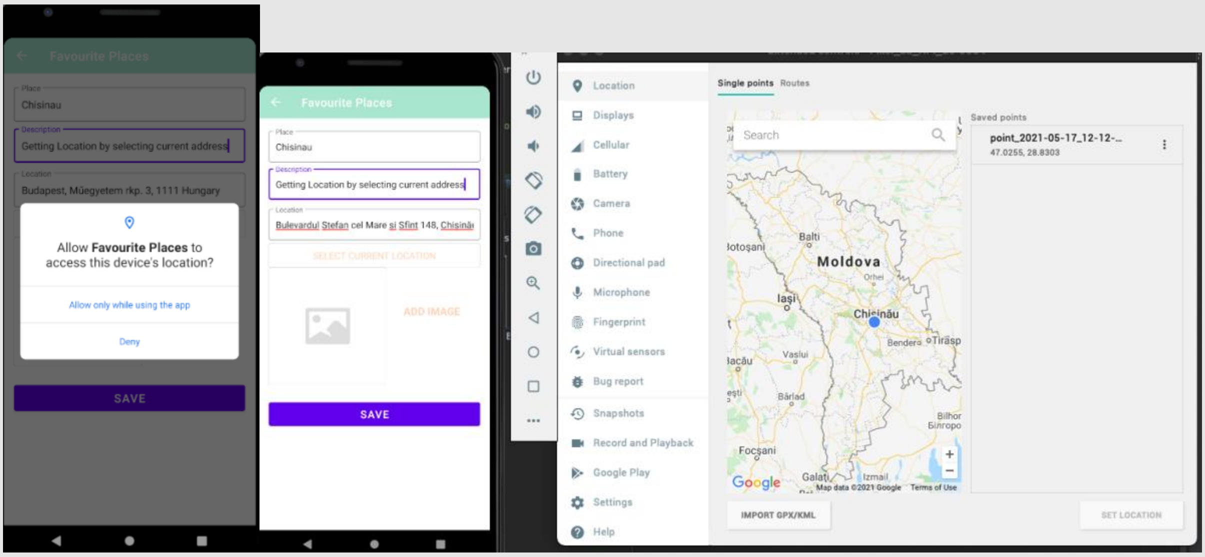
Deny

Allow **Favourite Places** to take pictures and record video?

Allow

Deny

The user has the option to choose current location, in which case the user will be asked to give permission to the up for their current location. If done their current location will be saved for the place.



Place: Chisinau
Description: Getting Location by selecting current address
Location: Budapest, Műegyetem rkp. 3, 1111 Hungary

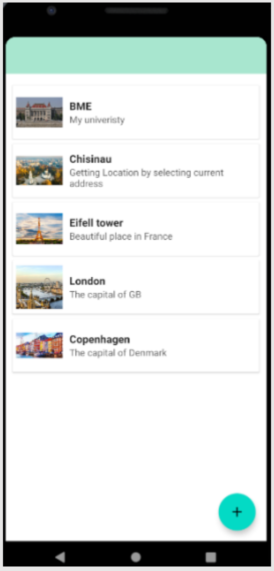
Allow **Favourite Places** to access this device's location?

Place: Chisinau
Description: Getting Location by selecting current address
Location: Bulevardul Stefan cel Mare si Sfint 148, Chisinău

point_2021-05-17_12-12-...
47.0255, 28.8303

- **View existing places**

All saved places will be displayed for the user in a RecyclerView where they can browse through them or click any individual one for more details.



BME
My univeristy

Chisinau
Getting Location by selecting current address

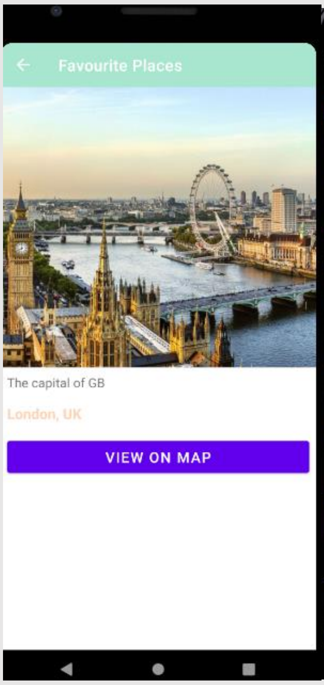
Eifell tower
Beautiful place in France

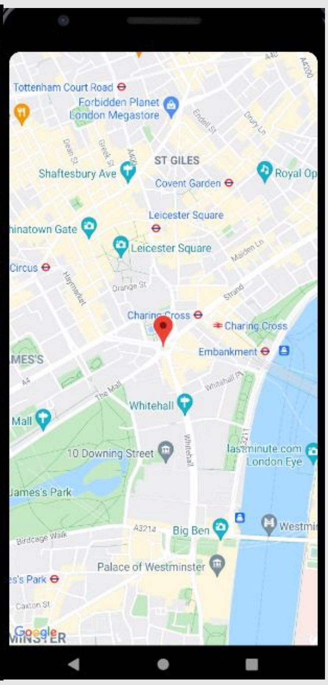
London
The capital of GB

Copenhagen
The capital of Denmark

- **Taping any place for details**

Once a place is picked it will open in a new screen where the user will also have the option to view it on a map. If picked a nice animation will zoom into the place on a google map.

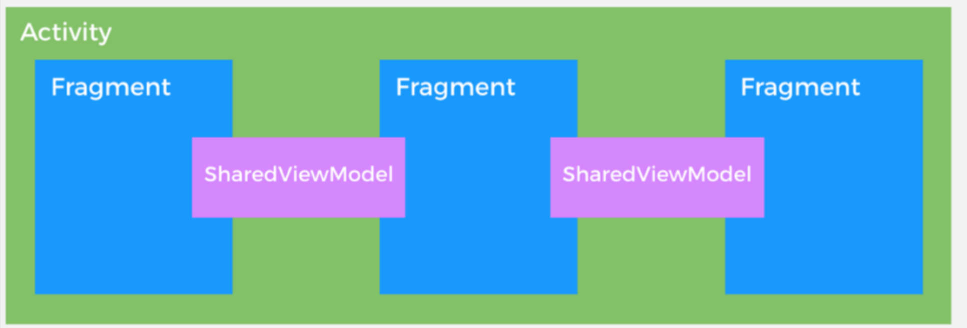




Technical details

- **Architecture**

I opted for a Single – Activity architecture. There is only one activity that initializes the first fragment and holds the logic for changing between fragments. The main benefit of this approach is speed since switching between fragments is less resource intensive than creating and switching between activities.



Activity

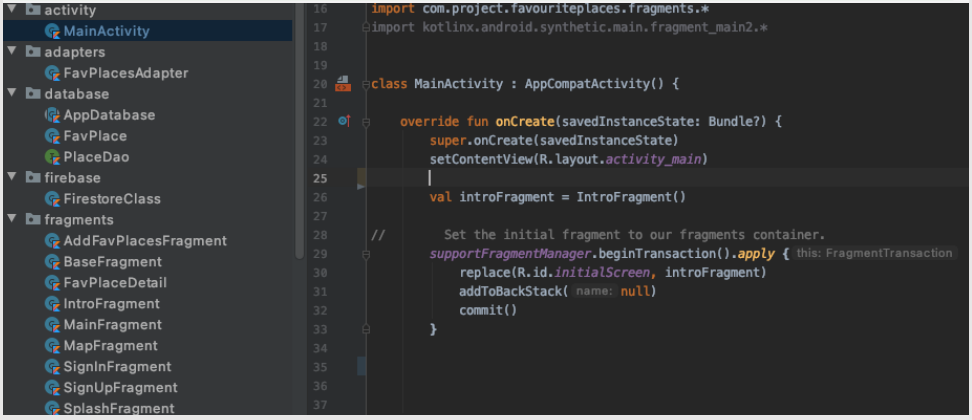
Fragment

Fragment

Fragment

SharedViewModel

SharedViewModel



```
▼ activity
    MainActivity
▼ adapters
    FavPlacesAdapter
▼ database
    AppDatabase
    FavPlace
    PlaceDao
▼ firebase
    FirestoreClass
▼ fragments
    AddFavPlacesFragment
    BaseFragment
    FavPlaceDetail
    IntroFragment
    MainFragment
    MapFragment
    SignInFragment
    SignUpFragment
    SplashFragment
```

```
import com.project.favouriteplaces.fragments.*
import kotlinx.android.synthetic.main.fragment_main2.*


class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val introFragment = IntroFragment()

//        Set the initial fragment to our fragments container.
        supportFragmentManager.beginTransaction().apply { this: FragmentTransaction
            replace(R.id.initialScreen, introFragment)
            addToBackStack( name: null)
            commit()
        }
```

- **Dexter**

Dexter is an Android library that simplifies the process of requesting permissions at runtime. Android Marshmallow and above includesa new functionality to let users grant or deny permissions when running an app instead of granting them all when installing it. This approach gives the user more control over applications but requires developers to add lots of code to support it. GitHub: https://github.com/Karumi/Dexter

- **Firebase Authentication**

Most apps need to know the identity of a user. Knowing the user's identity allows an app to personalize their experience and save their data. Firebase Authentication provides SDKs and ready-made UI libraries to help in the process of authenticating users.
https://firebase.google.com/docs/auth

- **Database**

Room was used to store data on the device. The Room persistence library provides an abstraction layer over SQLite to allow fluent database access while harnessing the full power of SQLite.
https://developer.android.com/training/data-storage/room

- **Google places API**

This API from google contains millions of places and I am using it to find places of interest. The Places API uses a place ID to uniquely identify a place.
https://developers.google.com/maps/documentation/places/web-service/overview