AirQualityPollen

Description:

AirQualityPollen is an App for getting Air-quality- and pollen-data of specified locations. Locations can be saved as favorites for fast access to the data of often-used locations.

Intended User:

People interested in Air-quality data. Also useful for users with pollen allergy.

Features:

The app gets the current position of the device. With one-click, the air-quality and pollen data can be shown for this current location. For this, the Ambee-API is used.

The user can select a location on a map-view and save this location in the favorites list. Then, the air-quality and pollen-data can be requested for these locations.
The data is stored in local database and is updated, when new data is available.

Finally, on the main screen, a 'nature picture of the day' is shown (fetched from unsplash.com)

Architecture

The architecture is structured in domain driven design, the packages are structured in the BCE (Boundary-Control-Entity) form. Therefore, the app is divided into the main business area ("business"-packages) and the more technical "presentation"-packages, where the activity/fragment/viewmodel/adapters are hold.

In addition, the best practices and Android architecture components are used continuously. The MVVM pattern is implemented for every fragment. The data is hold in ViewModels in LiveData structures with observers.
Furthermore, data-binding and view-binding from Android Architecture components are used throughout the app.

Retrofit2 and Moshi are used to implement the HTTP-REST communication and JSON-parsing of the data. This is used for both API-integrations (AmbeeAPI for the air-quality and poll data as well as the unsplash-API for getting the 'nature picture of the day').
The image data is fetched by the Picasso library and is also cached to minimize API-calls.

The favorites and their air-quality and pollen data are stored in a Room-database, also to minimize API-calls. If more current data is available, the new data is fetched and updated in the database.

For choosing a favorite location, the device location is used. The user is asked for permission and the permission is checked before making a call to update the current location.

When selecting a new favorite, the map is zoomed to the current location. From there, the user can navigate around the map, and with a long click, a marker is set to this location. When the user is satisfied with the location, he can “Add” this position to the favorites (which is stored in the Room database). In the background, the air-quality and pollen data are fetched and the dataset for this location is updated in the database.
These favorite locations are displayed in a RecyclerView in the main screen for fast access.

Some fields in the UI use binding-adapters for formatting or controlling, if elements are visible or not (like spinner). The air-quality and pollen-data are formatted to a unified format.

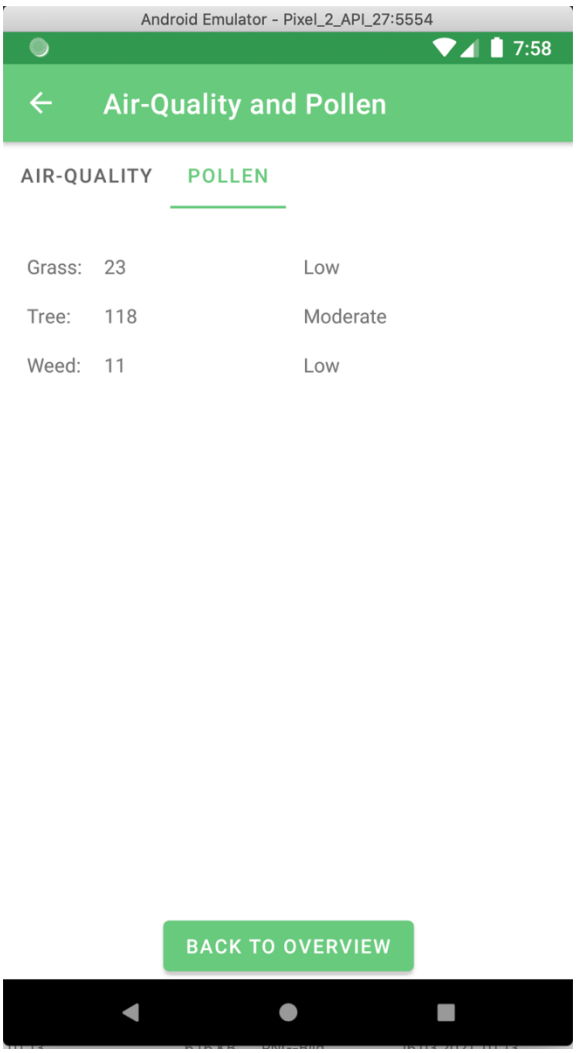
User-Interface:

- The color-schema is chosen from the Material Design color tool (https://material.io/resources/color)

The Main-Screen:

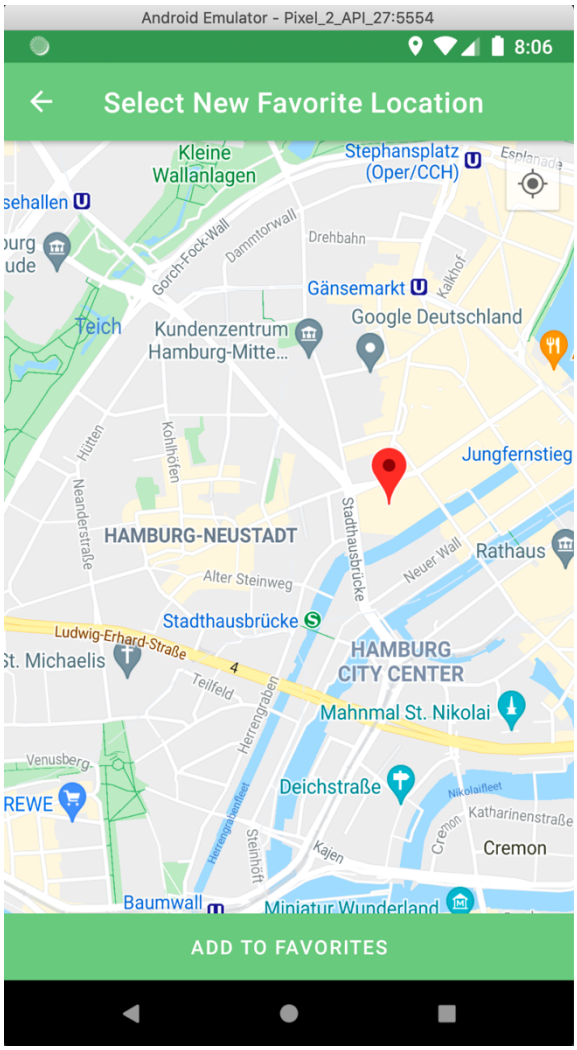
Details-Screen:

In the details screen, the user can switch between the air-quality and the pollen via Tab-Layout and ViewPager2-implementation.



Select favorite location:

The user can choose a location by long-clicking on a position. When a position is marked, the button “Add to favorite” made visible and displayed at the bottom of the screen.



Libraries and dependencies:

- Basic Android architecture components (ViewModel. LiveData,)
- Material design
- RecyclerView
- Retrofit2
- Moshi
- Picasso
- Room
- Google play services

Project-planning and Milestones

The project is divided into the following milestones:

Milestone 1:

- Basic project setup and with version control with GitHub
- Configure the basic libraries/plugins/dependencies for the project

Milestone 2:

- Integrate the Ambee-API (https://www.getambee.com)
- Construct the JSON-data objects for parsing the air-quality result data
- Implement the API-REST-call with Retrofit2 and Moshi to get the air-quality data
- Construct the JSON-data objects for parsing the pollen result data
- Implement the API-REST-call with Retrofit2 and Moshi to get the pollen data
- Implement the basic app structure with the main activity and fragments

Milestone 3:

- Integrate the TabLayout with ViewPager2 to separate air-quality and pollen data
- Integrate Google-Maps for selecting a location as favorite
- Integration of Room-database for storing the favorites with the latest air-quality and pollen data

Milestone 4:

- Integrate the RecyclerView to display the favorite locations and add the possibility to swipe-delete entries
- Integration of MotionLayout for animating the GoogleMaps

Milestone 5:

- Implement the ‘nature picture of the day’ feature from Unsplash.com (https://api.unsplash.com) using Picasso as image downloader and image cache
- Code cleanup