Project Map'Trip

Technical report
ITUe project
FIT BUT, 2020

Team name
Team PSL

Authors
Pavlo Vilkhovyi, xvilkh00
Simge Cimbilaz, xcimbi00
Louis Rives-Lehtinen, xrives00

Notes / instructions:

- It may happen that the questions or recommendations in the individual chapters will not fit for some assignments. Edit the chapter to contain reasonable information if convenient (for example, an explanation of why there is nothing to write in the chapter).
- Content in the shared parts of the technical report (TR) must be the consensus of all team members. Therefore, it is necessary to agree on it in the team.
- Follow the prepared formatting and structuring of the text (unless the logic of the matter requires otherwise).
- Images must contain a title and must be referenced from the text.
- Unless explicitly stated, you can write text together (shared parts) and the text is a collective work.
- Although the references are at the end of the document, you should add references especially in the first phase of the project, when you are conducting research and study.

Content

First part

---

1. Assignment and organization of the team

1.1 Objective

The goal of the project is to create a phone application designed for all aged travelers (good for Erasmus students like us). The app would be a map on which you could create notes of every country, the goal of these notes would be to write about your trips, when you went there, what did you do, etc… and add photos about it. The app would act like a souvenir book of all your trips so you will never forget them. The resulting GUI should be very simple since not all travelers are eased with technology, so a simple minimalist clickable map with a list of all the notes on another page should do the trick. A simple menu on the bottom of the app will allow to move between the map and the list, and the user should be able to add notes either from the map or the list.

1.2 Team

The work was at first split between Simge, Pablo, Thomas and Louis. Pablo and Simge would deal with the database, Thomas the list panel and Louis the Map panel. But after few weeks we figured out that each people had to work on a GUI part and Thomas had to cancel

his Erasmus to go back to France. Thus Louis remained on the implementation of the map and all features about it (adding notes while clicking a country and the search bar). Pablo started working on the menus and the list panel while Simge implemented the notes with the photos through list panel.

1.3 Roadmap

The goal of the first meeting was to organize the whole project, we talked about all the potential features of the app and the guidelines, which technologies & IDE we would use, imagined personas of the potential users. Louis made a Figma of the application to have an idea of how the GUI should look like, and finally we made a shared GitHub project for an easier access/share of files through AndroidStudio. Then on the next meetings, we started the actual “programming” with little tasks in Java that all of us would to and we would talk about it around a weekly meeting on Tuesday evening on the application “Discord”. After few weeks when the project started to get advanced, everyone started to work on their own to finalize the important tasks making few alpha versions of the app. And during the last week of the project Pablo merge all our work to create the final app.

1.4 Risks and measures

-Organizing and splitting tasks as a team:

- Risk: Difficulty in task organization and allocation.
- Precaution: Establish regular weekly meetings to discuss progress, challenges, and task assignments. Foster trust among team members through open communication and encourage transparency about individual workloads.

-Merging the project:

- Risk: Potential conflicts and overlapping in code due to independent work.
- Precaution: Assign a team member, in this case, Pablo, as the point person for merging tasks. We as a team helped him with the conflicts and explained him certain part of our codes.

-Thomas leaving the project mid-semester:

- Risk: Disruption in team dynamics.
- Precaution: Staying focus and rallied as a team to overcome the challenges.

-Unforeseen technical issues or SDK malfunctions:

- Risk: Technical glitches or SDK malfunctions.
- Precaution: Maintain clear channels of communication for team members to report and address technical challenges promptly when testing team-mates features.

2. Research and experience

2.1 Existing solutions

Get to know existing applications or services that solve a similar problem as your project. Ideally gain your own experience (install, log in, try it out). Briefly describe the benefits and limitations of these solutions. Each team member should study/test at least 2 solutions. In this section, list the authors of specific subchapters (meaning team members who have researched and tested existing solutions, not the authors of those applications :).

First existing solution (Louis Rives-Lehtinen)

-Google Maps: Google maps can also be used to save locations but its primary goal nowadays is mobility (how to travel from point A to point B) so it is not very instinctive for all users and the GUI is not the most adapted for our goal.

Second existing solution (Louis Rives-Lehtinen)

-Day One: is a popular journaling app that allows users to create entries with text, photos, and location information. User-friendly interface for easy journaling. Seamless integration of photos and maps to capture the travel experience. Automatic weather and location tagging for entries.
Although some advanced features might require a premium subscription, no direct implementation from the map.

Google maps: (Pavlo)

Inside of them users also can create notes

OneNote: (Pavlo)

Microsoft OneNote is a powerful note-taking app that can be adapted for journaling. It allows multimedia entries, organization, and syncing across devices.

2.2 User needs

In the interview with an Erasmus student who travels frequently, several valuable insights were gathered to inform the development of the travel diary app. The student expressed the need for an app that allows easy consolidation of photos and thoughts, with the ability to categorize entries based on countries and trips. A user-friendly interface is crucial for seamless navigation. After testing our application, he the lack of social aspect, expressing interest in sharing the travel diary with others, connecting with fellow Erasmus students. Although he expressed a genuine interest in trying out our “MapTrip” application.

2.3 Summary

After doing our researches, the design of our “MapTrip” app should prioritize a user-friendly interface and consolidation of travel memories. While we currently lack social connectivity and collaborative features, these aspects have been recognized as potential enhancements for future updates. By staying attuned to user preferences and evolving needs, our app can evolve to provide a more comprehensive and engaging experience for travelers.

3. Solution architecture

3.1 System architecture

The entire system is divided into two parts: the map panel and the list panel. The map panel comprises a clickable map with a search bar. When clicking on a country in the map panel, you can add notes related to that specific country. On the other hand, the list panel consists of a comprehensive list of all countries. When selecting a country from the list panel, you should be able to view the notes you've added about that country.

Initially, there was intended to be seamless communication between the two panels through the shared notes functionality. However, due to time constraints, we were unable to implement this feature as planned.

3.2 Application Architecture

1) We are processing user actions through the touchscreen with the maps or buttons.
2) The data is processed directly in the device through multiple java classes.
3) The visualization is processed in the “layout” files (.xml).

3.3 Data model

The Data is processed through different methods, on the map panel: the data is processed through a “note.txt” file that stocks every note but on the list panel, the data is stored in an SQL database. We are also using google maps API data.

3.4 Selected technologies and implementations

Technologies used during this project were Android Studio as the IDE, google maps APIs : Maps SDK for Android, Places API (new), and Maps Platform Datasets API. The whole code is in java. And we are using a lot of libraries (specified in the question 5.2).

4. GUI design - XY application (or XY part of the application) [Louis Rives-Lehtinen xrives00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

4.1 GUI Requirements

1. **Map Display:**

    - The main screen contains a **SupportMapFragment** to display a Google Map.
    - Users can click on the map, and the app will determine the country based on the clicked coordinates.

2. **Search Functionality:**

    - The app includes a **SearchView** at the top to allow users to search for a specific location or country.

    - The search bar is set to be slightly transparent (**android:background="#80FFFFFF"**) for a visually appealing effect.

3. **Information Window (CardView):**

    - When a country is clicked on the map or searched, an information window (CardView) appears at the bottom of the screen.

    - The information window displays the country name, an empty text box, a button to add notes, and a section to display existing notes.

4. **Adding Notes:**

    - Users can add notes for a selected country by entering text into the **EditText** and clicking the "Add Note" button.

    - The notes are stored in a **Map<String, String>** (**countryNotesMap**) and are displayed in the **TextView** within the information window. Then they are saved in the “notes.txt” file.

4.2 Models

Here is our figma.

https://www.figma.com/file/GOn68m0VTUgmnmmKShK98q/ITUe-Project?type=design&node-id=0-1&mode=design

4.3 Pilot test

I ran the application on my personal phone so my friends could test it easily.

4.4 Test evaluation and design review

The testers were able to test all features without any explanations so the GUI is quite intuitive. Adding colors to the countries with notes would be a good feature if we have time.

Second part

---

5. GUI implementation - XY application (or XY part of the application) [Louis Rives-Lehtinen xrives00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

5.1 Implementation

- Google Maps Integration (MainActivity and associated layout):
    - Purpose: Display an interactive map and handle user interactions related to mapping.
    - Key Implementation:
        - Utilizes SupportMapFragment for embedding the map.
        - Implements OnMapReadyCallback to perform actions once the map is ready.
        - Uses GoogleMap.OnMapClickListener to detect map clicks.
    - Connection with Functions:
        - onMapReady: Initializes the map and sets up click listeners.
        - onMapClick: Determines the country based on clicked coordinates.
- SearchView (searchView in MainActivity layout):
    - Purpose: Enable users to search for specific locations.
    - Key Implementation:
        - Configured as an AndroidX SearchView widget.
    - Connection with Functions:
        - onQueryTextSubmit: Triggers the move to the searched country on submission.
- CardView (infoCardView in MainActivity layout):
    - Purpose: Display information about the selected country.
    - Key Implementation:
        - Utilizes CardView for a stylized container.
        - Contains child views like TextView and EditText for displaying and interacting with data.
    - Connection with Functions:
        - addNoteForCountry: Adds notes for the selected country.
        - updateNotesTextView: Updates the notes display.
- Geocoding (moveMapToSearchedCountry in MainActivity):
    - Purpose: Convert country names to coordinates for map movement.
    - Key Implementation:
        - Uses Geocoder to get coordinates from the country name.
    - Connection with Functions:

        - moveMapToSearchedCountry: Moves the map to the location of the searched country.
- File I/O and Notes (loadNotesFromFile, saveNotesToFile, updateNotesTextView in MainActivity):
    - Purpose: Persist and display notes related to each country.
    - Key Implementation:
        - Reads and writes notes to a local file.
    - Connection with Functions:
        - addNoteForCountry: Adds notes to the data model.
        - updateNotesTextView: Updates the displayed notes.
- Auxiliary Data Structures:
    - countryNotesMap: A Map<String, String> to store notes for each country.

5.2 Tools and libraries used

My part is using all of those libraries:

```
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.location.Address;
import android.location.Geocoder;
import androidx.appcompat.widget.SearchView;
import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
```

I did not have any particular issues with them, especially that I already worked almost all of those java libraries. “Google Maps APIs” offers powerful mapping features with easy integration but it requires a thorough understanding of the API documentation for efficient use and the free trial lasts only 3 months. “SearchView and CardView” (part of AndroidX) provid consistent UI components across different devices.

5.3 Final testing

I ran the final application on my personal phone so my friends could test it easily.

5.4 Test evaluation

The testers were able to test all features of the final application without any explanations so the GUI is still quite intuitive.

6. Conclusion

Users successfully navigated the map by clicking on countries. Users found the map responsiveness to be swift and accurate. Clicking on a country effectively triggered the display of relevant information in the CardView. The search functionality was effective; the map smoothly transitioned to the selected country upon search submission. Adding notes was intuitive, and users liked the seamless integration of notes with the geographical data. Users confirmed that their notes were saved even when the app was closed. This added to the overall positive user experience. Since the concept “MapTrip” of the app was my idea, I kind off was the leader of the team, even though there was no real hierarchy, we would all listen to each other. Working in a team allows bigger project, and the teamwork is more motivating since you are not only working for yourself. I think that utilizing a project management tool (such as “Jira”) would be more effective for next projects.

Reference (Louis)

Write here especially important sources from which you drew information (books, articles, magazines, blogs, etc.).
Links to software, manuals, forums, etc. should be provided in the form of a url in a footnote.

https://developer.android.com/studio
Implementing Google Maps on Android:
https://www.youtube.com/watch?v=WouAQmqJI_I

Step by Step Google Maps Implementation in Android App | Google Maps in Android: Step-by-Step Guide:
https://www.youtube.com/watch?v=pOKPQ8rYe6g

https://developer.android.com/develop/ui/views/components/button
https://developer.android.com/develop/sensors-and-location/location/maps-and-places

https://developer.android.com/reference/android/widget/SearchView

https://chat.openai.com/

https://stackoverflow.com/questions/16808191/how-to-change-the-color-of-a-particular-area-on-google-map-api-v2-in-android

https://www.figma.com/file/GOn68m0VTUgmnmmKShK98q/ITUe-Project?type=design&node-id=0-1&mode=design

https://developer.android.com/studio/intro/update?hl=en

4. GUI design - MapTrip part of the application [xcimbi00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

4.1 GUI Requirements

- Display a RecyclerView (`recyclerViewComments`) to show a list of comments.
- Each item in the RecyclerView should use the `comment_item.xml` layout.
- Buttons (`buttonEditComment` and `buttonDeleteComment`) to edit and delete comments.
- Display the details of a selected comment.
- Allow the user to navigate back to the list of all comments.
- No direct GUI requirements, as it's a data model.
- Inflate the `comment_item.xml` layout for each item in the RecyclerView.
- Handle interactions for editing and deleting comments.
- Display details about the selected country.
- Include a RecyclerView (`recyclerViewComments`) to show comments related to the country.
- Buttons to add new comments or view all comments.
- CheckBox (`checkBoxComment`) to select comments.
- TextView (`textViewComment`) to display the comment text.
- Buttons (`buttonEditComment` and `buttonDeleteComment`) for actions.
- EditText (`editTextUpdatedComment`) for entering updated comments.
- Button (`buttonSave`) to save changes.
- Display details about the selected country.
- Include a RecyclerView (`recyclerViewComments`) for comments related to the country.
- Buttons for adding new comments or viewing all comments.
- Display a RecyclerView (`recyclerViewComments`) to show a list of all comments.

- Buttons for editing and deleting comments.

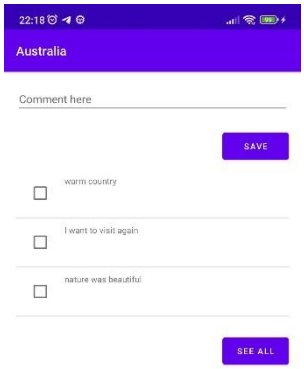
These GUI requirements aim to provide a clear user interface for managing comments related to countries. Users should be able to view, edit, and delete comments seamlessly through the provided activities and layouts.

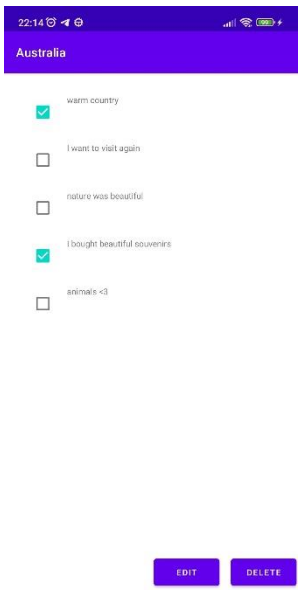
4.2 Models

Here is our first design.

https://www.figma.com/file/GOn68m0VTUgmnmmKShK98q/ITUe-Project?type=design&node-id=0-1&mode=design

And last design for my part.





4.3 Pilot test

I used my personal phone as a physical device to show it to my friends that I would like to hear their opinions.

.4.4 Test evaluation and design review

The testers were able to test all features without any explanations so the GUI is quite intuitive. Adding colors to the countries with notes would be a good feature if we have time.

Second part

5. GUI implementation - MapTrip part of the application [cximbi00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

5.1 Implementation

AllCommentsActivity.java:

GUI Implementation:

- Layout: Uses the activity_all_comments.xml layout, which likely includes a RecyclerView for displaying all comments.
- Navigation: May include navigation options to move to other activities or perform specific actions.

Connection with Functions and Data Model:

- RecyclerView: Connects with a CommentsAdapter to populate and display comments.
- Interaction: Implements logic for user interactions, such as clicking on comments.

AllCommentsResultActivity.java:

GUI Implementation:

- Layout: Could use a layout similar to activity_all_comments.xml.
- Display Results: Likely displays the result of all comments retrieved.

Connection with Functions and Data Model:

- Data Retrieval: Fetches data, possibly from a database or another source.
- Display: Populates the UI with the retrieved data.

CommentsAdapter.java:

GUI Implementation:

- RecyclerView Adapter: Binds the data from a list of comments to the RecyclerView in the associated layout (comment_item.xml).
- Item View: Inflates comment_item.xml for each item in the list.
- Connection with Functions and Data Model:
- ViewHolder: Holds references to UI elements in each item view.
- Data Binding: Binds comment data to the UI elements.

CountryDetailActivity.java:

GUI Implementation:

- Layout: Uses activity_country_detail.xml.
- Display Details: Displays details of a specific country, possibly with comments.
- Connection with Functions and Data Model:
- Data Retrieval: Retrieves details of the selected country.
- Comments: Connects with the comment system for the specific country.

DatabaseHelper.java:

- Data Model and Storage:
- SQLite Database: Manages the local storage of comments or country details.
- CRUD Operations: Implements methods for creating, reading, updating, and deleting data.

comment_item.xml:

GUI Implementation:

- Item Layout: Represents the layout for each item in the RecyclerView.
- Components: Includes components like CheckBox, TextView for comment text, and buttons for editing and deleting.

edit_comments_dialog.xml:

GUI Implementation:

- Dialog Layout: Represents the layout for editing comments.
- Components: Includes an EditText for entering updated comments and a Button for saving changes.

activity_country_detail.xml:

GUI Implementation:

- Layout: Represents the layout for displaying details of a specific country.
- Components: May include text views, images, or other elements for country details.

activity_all_comments.xml:

GUI Implementation:

- Layout: Represents the layout for displaying all comments.
- Components: Likely includes a RecyclerView for listing comments.

5.2 Tools and libraries used

In the GUI implementation, the following tools and libraries were utilized:

```
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Button;
import android.content.ContentValues;
import android.widget.Toast;
import android.util.Log;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.database.sqlite.SQLiteDatabase;
import com.example.petproject.Comment;
import com.example.petproject.CommentsAdapter;
import com.example.petproject.DatabaseHelper;
import java.util.Arrays;
import java.util.ArrayList;
import java.util.List;
```

5.3 Final testing

I used my personal phone as a physical device to show it to my friends that I would like to hear their opinions.

5.4 Test evaluation

To discuss the results, no revision of the design is necessary, just a description of what and how it is necessary to do differently in the future.

Reference (Simge)

Write here especially important sources from which you drew information (books, articles, magazines, blogs, etc.).
Links to software, manuals, forums, etc. should be provided in the form of a url in a footnote.

https://developer.android.com/studio/intro/update?hl=en

https://www.figma.com/file/GOn68m0VTUgmnmmKShK98q/ITUe-Project?type=design&node-id=0-1&mode=design

https://developer.android.com/develop/ui/views/components/button

https://youtu.be/BBWyXo-3JGQ?si=NYeOHXrgkZGRtmxZ

https://chat.openai.com/

https://www.codebrainer.com/blog/android-studio-layout-editor-for-beginners

https://developer.android.com/studio/write/layout-editor
https://developer.android.com/studio/run/managing-avds

4. GUI design - XY application (or XY part of the application) [xvilkh00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

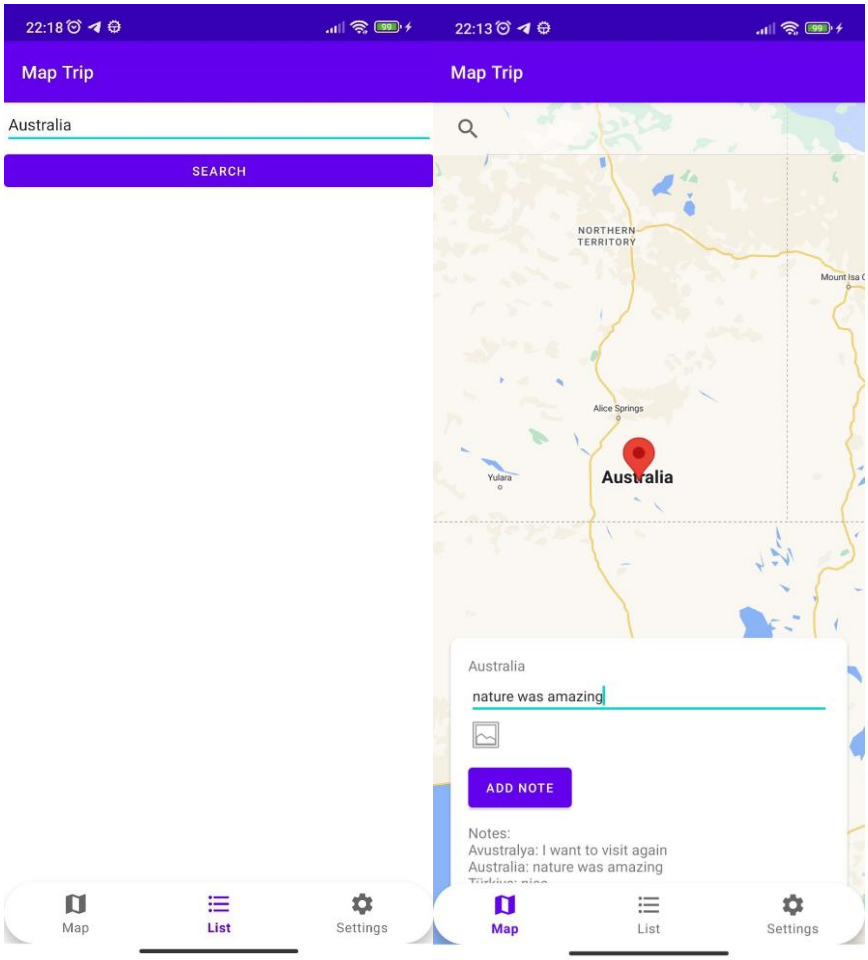
4.1 GUI Requirements

User requires a mobile phone in order to launch the application with android operation system. Also the phone needs to have google play installed in order to open the map properly.

4.2 Models

Here is our figma.

https://www.figma.com/file/GOn68m0VTUgmnmmKShK98q/ITUe-Project?type=design&node-id=0-1&mode=design



4.3 Pilot test

I have run this application on my laptop on a virtual device in android studio. The users were not told about the project and they managed to manipulate between screens properly and searched for specific country

4.4 Test evaluation and design review

The testers were able to test all features without any explanations so the GUI is quite intuitive. Adding colors to the countries with notes would be a good feature if we have time.

Second part

---

5. GUI implementation - XY application (or XY part of the application) [xvilk00]

This chapter, including subchapters, must be prepared by each member of the team separately, for his own part of the GUI application, which he will solve as an author.

5.1 Implementation

AutoCompleteTextView implementation which allows for the user to get the country name by entering 1 symbol inside of serchbar.

```
<AutoCompleteTextView
    android:id="@+id/autoCompleteTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:completionThreshold="1"
    android:hint="Tap to select a country"></AutoCompleteTextView>
```

Implementation of the bottom navigation bar:

```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/bottomAppBar"
    android:layout_gravity="bottom"
    app:fabCradleMargin="10dp"
    app:fabCradleRoundedCornerRadius="50dp"
    android:background="@drawable/bottom_background"
    app:menu="@menu/bottom_menu"
    >
</com.google.android.material.bottomnavigation.BottomNavigationView>
```

```
BottomNavigationView bottomNavigationView =
findViewById(R.id.bottomAppBar);
bottomNavigationView.setSelectedItemId(R.id.list);
bottomNavigationView.setOnItemSelectedListener(item -> {
    switch (item.getItemId()){
        case R.id.settings:
            startActivity(new Intent(getApplicationContext(),
SettingActivity.class));
            overridePendingTransition(R.anim.slide_right,
R.anim.slide_left);

            finish();
            return true;

        case R.id.map:
            startActivity(new Intent(getApplicationContext(),
MainActivity.class));
            overridePendingTransition(R.anim.slide_right,
R.anim.slide_left);

            finish();
            return true;
        case R.id.list:

            return true;
    }
    return false;
});
```

It allows to move between the activities (Intents) and is always placed at the bottom of the screen. Whenever user press the button on the navigation bar the icons change the color into different from original, so user can understand in which screen user is located and what can be done there.

Animation:

```
overridePendingTransition(R.anim.slide_right, R.anim.slide_left);
```

Animation allows to move between screens more smoothly and opens them from left to right slide

5.2 Tools and libraries used

My part is using all of those libraries:

```
import android.os.Bundle;
import android.view.View;
import android.view.Menu;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.TextView;

import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.navigation.NavigationView;

import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
```

From time to time I have experienced issues with android studio itself. When I have tried to create the activity, android studio automatically added kotlin file instead of java. Through the project creating I had to update android studio to newer versions in order to launch the application. Sometimes it took too much time to launch the project.

5.3 Final testing

I ran the application on my laptop on a virtual device so others can take a look at it and evaluate it.

5.4 Test evaluation

The GUI is quite intuitive. Project runs pretty fast on physical devices.

6. Conclusion

Implementation

Used AutoCompleteTextView for efficient country name retrieval.

Implemented BottomNavigationView for seamless navigation and visual cues.

Employed animation effects (overridePendingTransition) for smoother transitions.

Tools and Libraries

Utilized essential tools, including NavigationView and NavController, for enhanced functionality.

Challenges

Faced occasional Android Studio issues, resolved through updates and adjustments.

The search feature proved efficient country selection upon submission. Adding notes proved to be an intuitive process, and users liked the saving of their notes which contribued to an overall positive user experience.

Reference (Pavlo)

https://chat.openai.com/

https://www.youtube.com/watch?v=bgIUdb-7Rqo&ab_channel=CodinginFlow

https://stackoverflow.com/questions/14779688/put-buttons-at-bottom-of-screen-with-linearlayout

https://www.geeksforgeeks.org/autocompletetextview-in-android/

https://stackoverflow.com/questions/17526533/moving-from-one-activity-to-another-activity-in-android

https://www.youtube.com/watch?v=DX8PZTpnudg&ab_channel=AndroidKnowledge

https://stackoverflow.com/questions/23602393/unique-id-for-rows-in-sqlite-across-all-devices-in-android

https://developer.android.com/build/releases/gradle-plugin#kts