
GitHub Username: cmuresan

Travel Transylvania

Description

Travel Transylvania makes exploring Transylvania's beautiful medieval cities along its still wild forests and mountains and its captivating stories easier than ever before by organizing all the essential info in one place, making it easy to find information about these places and also to get around.

Intended User

The intended users of the app could be travelers, but shortly it is for everyone who is interested to know a little more about Transylvania and the Carpathian Mountains in Romania and especially for those who are brave enough to visit these places.

Features

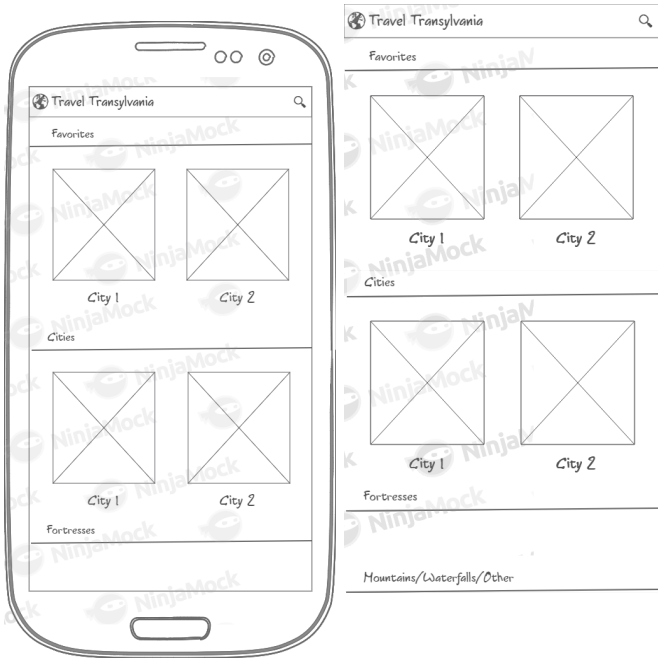
The main features of the app are:

- Provide information about places to be visited
- Mark place as favorite and create a list of favorites
- Provide directions to a specific location

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1

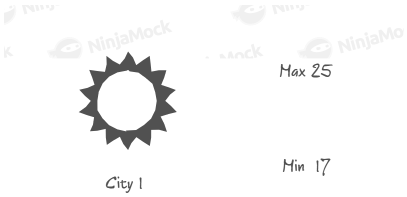


The main screen is represented by a list of locations split in different categories. First group of locations is “Favorites”, this being displayed only if the user marks a locations as favorite from inside its details screen. Following this Favorites group, there will be multiple sections like Cities, Fortresses and many others, each location being displayed as a square image with the name below. At the top of the screen there’s also a search button so the user can search for a specific place he has in mind.

Screen 2

The second screen presented is the Details screen of each location displayed on the main screen of the app. This consists in a top section displaying a cover image of the place, followed by a description and multiple actions at the bottom like checking the location on map or sharing it to a friend. Also, at the top of the screen there is a favorite button, which adds the location to user's Favorites list, and anchored bottom-right there is a floating action button for the most common action - get directions - which will show on map the route from your current location.

Screen 3



Screen 3 is actually the widget of the app, which will display an icon representing the current weather of the selected place and the maximum and minimum of the day.

Screen 4

Screen 4 represents the widget's settings screen where the user can select a place to get weather information for.

Key Considerations

How will your app handle data persistence?

The data will be locally saved using Room.

Describe any edge or corner cases in the UX.

If there's a bad internet connection, the map might load pretty slow and this might make the user think it's a bad experience.

Describe any libraries you'll be using and share your reasoning for including them.

- Picasso for better handling of images displayed
- Android Architecture Components suite of libraries for better structuring the app and aligning it to modern Android development
- Location services for providing a personalised experience
- Maps services will be used for providing useful information to the user

Describe how you will implement Google Play Services or other external services.

- Picasso library will be used for a better handling of all the images displayed in the app. For example, it will be used to dynamically load a list of items creating so the main screen.
- From the Android Architecture Components suite will be used libraries like Room, ViewModel and LiveData to create a robust architecture and this way a minimum number of database calls will be required
- Location services is needed to provide the user a personalised experience by highlighting or suggesting the user locations that are nearby so he won't miss them in case they are not in his plan, but has some spare time to visit
- Maps services are used to show each location on map and so this way, when visiting the place, the user can see his current position so it could provide useful information about the exact location and directions he can go to from that point.

Other considerations:

- The app is written solely in the Java Programming Language
- App keeps all strings in a strings.xml file and enables RTL layout switching
- The app will include support for accessibility by including content descriptors for images
- The app performs short duration search using AsyncTask
- Gradle build tools version 3.1.3
- Gradle wrapper version 4.4
- Picasso version 2.71828
- ViewModel and LiveData version 1.1.1
- Room version 1.1.1
- Location services version 15.0.1
- Maps services version 15.0.1
- Android Studio version 3.1.3

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Add gradle dependencies for libraries and services needed for the project using the latest stand organise them to use the same version, which will be declared in the project's gradle file
- Add app release keystore signing

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for PlaceDetailsActivity
- Build UI for Widget
- Build UI for WidgetSettings
- Build UI for CheckOnMapActivity

Task 3: Create Room database

Without using an API to retrieve the data, a database must be created and all the information stored in it.

- Gather information and pictures
- Use Room to create the database and all the entities needed
- Persist all the data in the database
- Provide an interface to work with the database

Task 4: Make use of Location services

Implement a location provider class that uses GoogleApiClient to retrieve last known location and also request location updates if needed

- Create LocationProvider interface and a class that implements its methods
- Build and connect a GoogleApiClient instance
- Implement ConnectionCallbacks
- Implement getLocation method which returns the last known location provider by Location services
- Implement start and stop location updates methods

Task 5: Display the MainActivity's data

Create the appropriate adapter and and view holders for creating the list of places with all the section headers.

- Create the UI for each place item

- Create the UI for headers
- User RecyclerView for the list
- Create the adapter using data types for the two different UI elements
- Bind all the data

Task 6: Create the MainActivity's search

Implement search on MainActivity so the user can look for a place by its name.

- Make the toolbar extend in a search bar
- Make the query with the input search text
- Display the proper data in the list

Task 6: Create the PlaceDetails activity

Create the UI using the Coordinator layout together with the AppBar layout for a smooth scrolling of the top section of the screen, collapsing into a toolbar.

- Display all the information regarding a particular place
- Implement click on the action buttons below text and the floating button
- Add the toolbar heart icon for marking the place as favorite

Task 7: Create the WidgetSettings screen

Display a list containing the favorite places selected by the user.

Task 8: Create the Widget

Create the UI of the widget and implement it so it will display the latest information available at all times.

Task 9: Write Espresso tests

Write Espresso tests to verify that the PlaceDetailsActivity displays the selected place's information.