Explore Jyväskylä

Final documentation

Diána Deregi  
Tamás Gálffy

Contents

[Introduction 3](#_Toc468863107)

[The final user interface 4](#_Toc468863108)

[Login 4](#_Toc468863109)

[Map 4](#_Toc468863110)

[Profile 5](#_Toc468863111)

[Search 5](#_Toc468863112)

[Place addition 6](#_Toc468863113)

[Database 7](#_Toc468863114)

[Server side 7](#_Toc468863115)

[Locations table 7](#_Toc468863116)

[Users table 7](#_Toc468863117)

[Rating table 7](#_Toc468863118)

[Client side 7](#_Toc468863119)

[Locations table 8](#_Toc468863120)

[Visits table 8](#_Toc468863121)

[Backend 9](#_Toc468863122)

[Download the locations 9](#_Toc468863123)

[Login 9](#_Toc468863124)

[Add a location 9](#_Toc468863125)

[Rate a place 9](#_Toc468863126)

[Android application 10](#_Toc468863127)

[How it starts 10](#_Toc468863128)

[Titles 10](#_Toc468863129)

[Project structure 11](#_Toc468863130)

[Timetables 12](#_Toc468863131)

[Diána Deregi 12](#_Toc468863132)

# Introduction

This documentation summarizes the final product, the workflow, the changes about the assignment at the Mobile Project course. It completes the project’s planning documentation, it tells what is different in the end.

You can find the project here: https://github.com/Zakemi/Explore-Jyvaskyla

# The final user interface

You can see a video about the working application here: https://youtu.be/Ogm8H4lb4C0

## Login

The user can sign up with a Google Account. By clicking on the button, a new Intent will start and ask the user to sign in.

## Map

This is the map page of the application. It shows a map and the place’s location. Every place icon show information about the category of the place. If the user has already visited a place, the icon will be transparent but still visible. The user is displayed with a green man icon.

With the button in the top left corner the start the place addition activity.

With the button in the top right corner the user set the map’s camera to the user’s location. If it isn’t available, the map will show the city centre.

With the buttons in the bottom the user can slide the screen to the search (left) or profile (right) screen. The use can slide between these fragments anywhere in the screen. The buttons want to make it easy because of the map.

If the user click on a place, an info window will appear with basic information. If the user click on this window, a dialog will appear where the user can rate the place.

## Profile

The profile screen shows the titles in the top. First are the gold titles, then the silver ones and after that the bronze ones.

Under them there is the user’s name which came from the Google Account.

Under that there are the icons of the category. If the user earns a title in a category, the background of the icon will change to gold, silver or bronze.

## Search

The user can select what places want to see by select the category of the place. In the list the user can pick them and with the green button do the search. It will jump to the map screen.

In the first section, the *Select all* and *Select nothing* makes it easy to select all the categories or nothing.

## Place addition

It makes possible to the user to add a place to the database.

The *PICK* *PLACE* button opens the Google’s place picker service. There the user can pick a place which is already exists in the Google’s database or just the coordinates of the new place. After the confirmation, the activity will autofill the fields.

All fields are editable, the user can correct the details after the using of the place picker or fill the fields by own.

If everything is correct, click on the green button. A notification will thank the addition.

# Database

The server is connected to a MySQL database. In this section you can read the details of the table what you should create before deploy the server.

## Server side

The goal of the server side database is to proceed the same information about the places to the clients, store the users.

In the planning documentation, there were a Titles and a Categories table too. They have been removed, because this information doesn’t change frequently. They are stored inside the application.

### Locations table

It stores information about the places which can be visible by the users.

Columns:

* ID: Identify the place. Primary key, Autoincrement. Type: int(11).
* Name: The name of the place. Type: varchar(256).
* Latitude: The location of the place (lat). Type: double.
* Longitude: The location of the place (lng). Type: double.
* Address: The address of the place. Type: varchar(512).
* GoogleId: The ID from Google. Type: varchar(256).
* Type: The type (category) of the place. Type: varchar(256).
* Phone: The phone information of the place. Type: varchar(45).
* Web: The web page of the place. Type: varchar(256).
* UserID: The ID of the user who added this place. Type: int(11).

### Users table

It stores the users to helps to identify the requests. The values came from the Google account.

Columns:

* Id: Identify the users. Generated by Google. Type: varchar(256).
* Name: The name of the user. Type: varchar(256).
* Picture: URL, points to the avatar of the user. Type: varchar(512).

### Rating table

It stores the rated what the users sent to the server.

Columns:

* UserId: Identify the user. Foreign key to Id column of users table. Type: varchar(256).
* PlaceId: Identify the place. Foreign key to ID column of locations table. Type: int(11).
* Rating: The user’s rate. Type: int(11).

## Client side

The purpose of the client side database is to store the places from the server and the visits what the user did.

The Titles and Categories are removed like in the server side. This information is stored inside the application.

Ranks table removed: The application doesn’t store the ranks in the database. If the user doesn’t connect to the internet, he/she can’t rank.

Details table removed: The application stores all visits in the Visits table. There’s no more need to this table.

### Locations table

It stores the locations from the server. The columns are the same like in the server side but the UserId.

### Visits table

It stores the places which are visited by the user.

Columns:

* Id: Identify the visit. Primary key, Auto increment. Type: Integer.
* PlaceId: Identify the place. Foreign key to ID column of locations table. Type: Integer.

# Backend

The application sends requests to the server. The following requests are available.

## Download the locations

* Path: /locations
* Method: GET
* Response body: The list of the places in JSON format.
  + Keys: Column names from *locations* table and “Rate”. Rate is the average rate based on the *ratings* table.

## Login

* Path: /login/<idtoken>
* Path variable (idtoken): Based on the Google Authentication process.
* Method: GET
* Response body: A JSON object.
  + Keys: Id, Name, Picture. They came from the idtoken.

## Add a location

* Path: /locations
* Method: POST
* Request body: The details of a place in JSON format.
  + Keys: GoogleId, Name, Address, Latitude, Longitude, Type, Phone, Web, UserID
* Response body: Information about the result.
  + Keys: success, error (if success is false)
  + Values: true, false

## Rate a place

* Path: /rate
* Method: POST
* Request body: The details about the rate.
  + Keys: UserId, PlaceId, Rating

# Android application

## How it starts

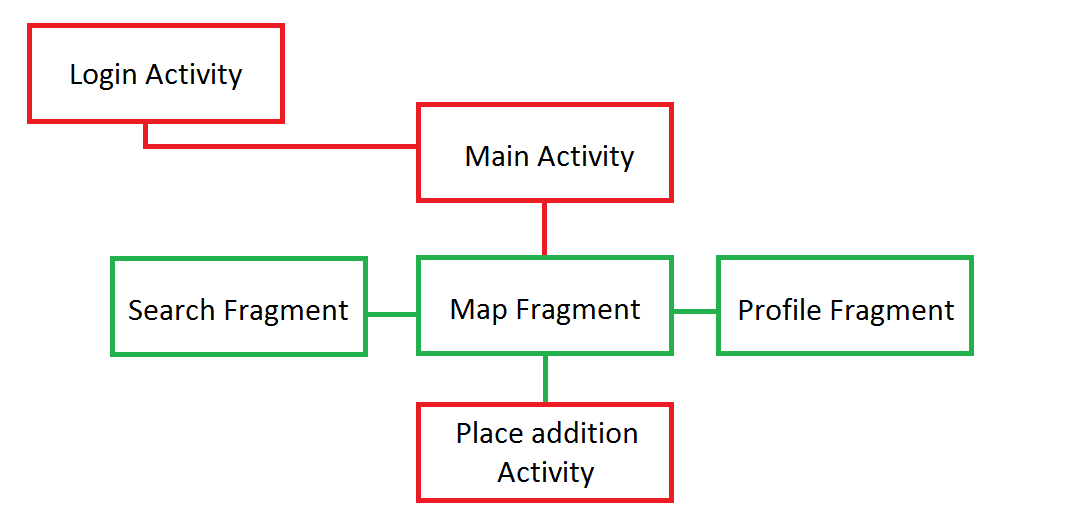
The application has 3 activities. It starts with the Login. This activity checks, is there any saved data about the use (id). If yes, it means the user have already logged in at a previous start, so it can start the Main activity. If no, it will continue to load the Login activity with the login button. It won’t load the Main activity until the user login successfully. Before the load, it saves the user’s data to the shared preferences.

The Main activity will show the three fragments. At the start, it will load the Map fragment. The user can change the screen by slide the fragments. In the Map fragment two button helps in this process. These contain the main functions of the application and the fragments are a good way to make the load of screens faster.

First, the map fragment will load the available places from the database and put the markers on the map. After it, it will try to refresh the locations table with the data from the server.

On the start of the Search fragment, it will use the Categories class to create the list of the categories. It prepares the selection functionality: make the categories clickable, setup the *Select all* and *Select nothing* buttons.

The Profile fragment collects the number of the places in each category and the number of the already visited places in the them. So, it will able to count how many percentage of the places is visited in each category. Based on this information, it will show the right title for the user. On the start, it will also set the username based on the information from the Google authentication process.

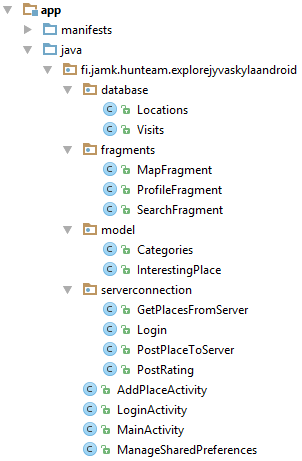


## Titles

Three levels of the titles are available in each category. The user receives gold title, if he/she visits at least 75% of the places in a category. The silver title is available after 50% place visits and the bronze is available after 25%. The following table contains the titles.

|  |  |  |  |
| --- | --- | --- | --- |
| Category name | Gold title | Silver title | Bronze title |
| Museum | Knower of the Past | Researcher of the past | Interested in the past |
| Café | Drinks Coffee like a Finnish | Coffee Lover | Knows what Coffee is |
| Church | Close to God | Seeking God | Believer |
| Entertainment | Eternal Child | On the Move | Playful |
| Bar | Just one more Beer | Makes Friends in a Bar | Bar Visitor |
| Bakery | Pulla Lover | Cookies researcher | Sweet-toothed |
| Food | Chow Hound | Gobbler | Hungry |
| Nature | Hippy | Nature Lover | Touring |
| Park | Park-ing | Sitting on bench | Park finder |
| Other | Unknown Knower | Going to Unusual Places | Dissatisfied with the Known |

## Project structure

You can see the project’s structure in the picture.

In the database package, there are the classes which connect to the application’s database. The Locations reaches the places in the databases and make queries. The Visits reaches the places what the user have already visited.

In the fragments package, there are the Fragment classes. The Main Activity will create them and show the right of them.

In the model package, there are the model of the places and the categories. The categories are created in the class.

In the serverconnection package, there are the classes which send requests to the server. All of them are Asynctask class. For the format of the requests, see the Backend section.

In the main package, there are the activities of the application and a helper class to reach the shared preferenes.

# Timetables

## Diána Deregi

|  |  |  |  |
| --- | --- | --- | --- |
| Week | Task | Estimated  Hours | Actual  Hours |
| 41 | Find the user with GPS and put a marker on map | 2 | 3 |
| 42 | Detect visits and save to database | 2 | 2 |
| 42 | GET locations from server | 1 | 2 |
| 45 | Create place addition activity | 2 | 4 |
| 46 | POST the place addition information to server | 1 | 2 |
| 46 | Make the plan of the rest of the application | 1 | 2 |
| 47 | Plan doc: Introduction, Objectives, Use cases, Mockups | 2 | 3 |
| 47 | Plan doc: Usage workflow and timetable | 2 | 2 |
| 47 | Plan doc: Databases | 1 | 1.5 |
| 47 | Plan doc: Backend | 1 | 0.5 |
| 48 | Create the three fragments | 2 | 6 |
| 48 | Refactor Locations database table and save locations there | 2 | 3 |
| 48 | List the categories in the Search fragment. Search layout. | 2 | 4 |
| 48 | Finish the category search | 3 | 5 |
| 48 | Create Visits database and make queries for Profile fragment | 2 | 3 |
| 48 | Put titles and icons in the Profile fragment with colours | 1 | 1.5 |
| 48 | Customize the markers and the info windows on the map | 2 | 2 |
| 49 | Make design, finish Map and Search functions | 4 | 4.5 |
| 49 | Work on the Google login | 3 | 3 |
| 49 | Start the rating process, AlertDialog added | 2 | 2.5 |
| 49 | Finish the rating process | 3 | 2 |
| 49 | Login process refactor, name on profile | 1 | 1.5 |
| 49 | Username added to Profile. UserID send at place adding | 1 | 1 |
| 49 | Final documentation: Introduction, Database, Backend, UI | 3 | 3.5 |
| 49 | Refactor, comments | 2 | 2 |
| 49 | Complete categories, make icons and pictures | 2 | 3 |
| 49 | Finish the final documentation, make presentation, create video | 4 | 5 |