Explore Jyväskylä

Diána Deregi

Tamás Gálffy

Contents

[Introduction 3](#_Toc468019286)

[About the document 3](#_Toc468019287)

[About the idea 3](#_Toc468019288)

[Objectives 4](#_Toc468019289)

[Use cases 5](#_Toc468019290)

[Vacation in the city 5](#_Toc468019291)

[Longer stay 5](#_Toc468019292)

[Outdoor activity 5](#_Toc468019293)

[Not only with Wi-Fi 5](#_Toc468019294)

[Usage workflow with mockups 6](#_Toc468019295)

[Main activity 6](#_Toc468019296)

[Profile activity 6](#_Toc468019297)

[Search activity 7](#_Toc468019298)

[Place addition activity 7](#_Toc468019299)

[Prototypes 8](#_Toc468019300)

[Mobile Application Development 8](#_Toc468019301)

[Android Application Development 8](#_Toc468019302)

[OOP, UML 9](#_Toc468019303)

[Database planning 10](#_Toc468019304)

[Backend 11](#_Toc468019305)

[Time planning and workload 12](#_Toc468019306)

# Introduction

## About the document

This document is our project plan for the Mobile Project course. Our assignment is to find a nice application idea, plan it then realize it. We summarize here the concept of the application, how we think about it in the beginning. It’s possible it will change during the implementation so the final product may be different.

## About the idea

Our idea is an application which helps to the visitors of Jyväskylä to explore the city in a funny way. We collected some interesting places which worth to see at least once. They are displayed in our application’s map to make it easy to find them and the way to them. If the user visit a place, we detect it and use this information to score points. If the user visit enough place in one category, he/she will earn a funny title refer to the category. There are more titles to each category so visit more place to improve the title! The user can follow how many places left in the categories and see the standings of other users.

If the user think we missed an important place, it’s possible to add it manually. It’s also possible to report a place if it seems invalid. The user can rate every location in the interest of highlight the best ones.

# Objectives

During the implementation, we take care of the usability, simplicity. We tried to bearing in mind these principles while we designed the application. The goal is to realize every imagined feature in this way which are described later in this document.

What else?

# Use cases

There are some cases when the application can be useful.

## Vacation in the city

When someone want to travel somewhere and see new things, he/she may make plans, search for sights before the journey. The application can help with this process. Inside it you can see not even the locations but the ratings. These ratings come from the other users. So if you don’t have too much time to look around, you can pick the most interesting and best places if you choose the top rated ones. It’s also useful if you ran out of the ideas in the middle of the day but you want to see more. So you can use the application not only at the planning phase of the vacation.

## Longer stay

It’s always a good idea to explore the new city where you will live for a longer time not only few days. Everything is new, foreign, you know no one and nothing. The application can show a playful way to this process. You will know the city place by place.

## Outdoor activity

The application can be useful for not only the new visitors. If you live here for a long time you may picked up the love of the movement, nature, outdoor activities. Now you can find new places to visit and entertainment to compete with others to earn better titles.

## Not only with Wi-Fi

If you don’t want or can’t use Wi-Fi at all time, it isn’t problem. You can visit places without it too, you will need only GPS. The application will save the results and upload to us when you connect to the Internet.

# Usage workflow with mockups

## Main activity

The main activity’s background is a map.

On it we can see the markers of the locations with green circles. In the real application, these circles will show the category of the place. If the use click on a marker, a small info box will appear. It contains the place’s name, address, phone, e-mail and rank. The users will be able to make their own rank here.

The brown man shape means the user.

In the bottom, left corner, the yellow circle symbolizes the GPS button. The search for user’s location require much battery. So, the user will able to turn on it only when it’s necessary.

In the bottom, right corner, the red circle symbolizes the place addition button. It leads the user to the addition activity. See details at the activity’s description.

In the top, left corner, the menu button leads the user to the search activity. See details at the activity’s description.

In the top, right corner, the button leads the user to the profile activity. See details at the activity’s description.

## Profile activity

The user’s titles are in the top of the activity. First are the gold titles, then the silver ones and after that the bronze ones.

Under the titles there is the user’s avatar. It will be connected to the titles. Not designed yet.

Under the avatar there are the categories. In every circle, there will be the sign of the category. The background of the circle means the level of it: gold, silver, bronze or none. If the user click on a circle, it will change to show the name of the category and the standings: the total number of the places with the visited places in the category.

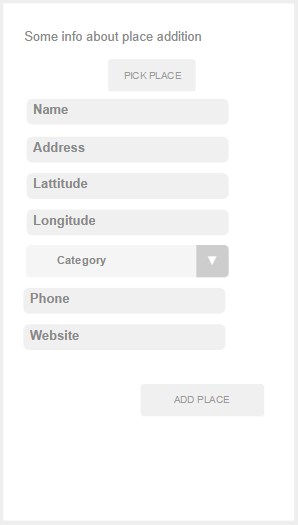
## Search activity

If the user click on the menu in the main activity, this search will slip from the left side. So it’s possible to slip it from left, no need to click on the menu button.

The user can select the categories what he/she want to see in the map. If a category selected, the sign of the category will be colourful. Otherwise it’s grey.

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

## Place addition activity

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 of the 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 *ADD PLACE* button. A notification will thank the addition.

# Prototypes

We have started the project before this course. We have already created two versions of the application.

## Mobile Application Development

Our theme was the server side so the emphasis was on the Node.js.

We have created a simple server with some example locations in the database. The clients can uses HTTP protocol to send request. The server responses in JSON format.

We have also created an Android application which can connect to the server, get the locations from it and put markers on a map. It can also detect the user’s location and show it on the map with an unique marker. If the user moves and gets too close to a place, it detects this as a visit. So the application notes these visits in a database. It’s necessary, because we want to give points only for the first visits.

You can find our presentation about this here: https://docs.google.com/presentation/d/1dFYQxMo\_qFluHhVjkhkB\_5DZOgE1dJkg39WPBwsBhQ0/edit#slide=id.g17ce424107\_0\_55

And here is a video about the Android application: https://www.youtube.com/watch?v=mwPkYpHQAjU

## Android Application Development

We made the second step during this course. We used the application what we have created during the Mobile Application Development and we improved it. Our theme was the Google Maps and GPS.

The server expanded with a new route definition. It’s able to accept a POST request which contains details about a new place. The server adds it to the database and when a new request comes to download the places, it will send the new place too.

The Android application has a new activity where the user can add the details of the new place. We used the built-in place picker widget where the users can use the Google Maps to choose easily the new location. After the addition, the application refresh the markers on the map.

You can find our presentation about this here: https://docs.google.com/presentation/d/1o7\_TSnFtUKX6fQgPI2yuPX0fYq7WFNZki\_tH4fslDfg/edit#slide=id.g194f921605\_0\_265

And here is a video about the Android application: https://www.youtube.com/watch?v=t9sUJgV6dFY

# OOP, UML

# Database planning

## Server side

The SQL database will contain the following tables.

### Places table

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

Fields:

* PlaceId: Unique field which identify the place. (Primary key)
* Google Id: If the place was added by Google place picker, this field isn’t empty.
* Name: The name of the place.
* Address: The address of the place.
* Latitude: The latitude of the location.
* Longitude: The longitude of the location.
* Category: The category of the place.
* Phone: The phone of the place.
* Web: The website of the place.
* UserId: The user who added the place.

### Ranks table

It stores the user’s ranks about a place.

Fields:

* Id: Unique field which identify the record. (Primary key)
* PlaceId: The Id of the place. (Foreign key -> Places table)
* UserId: The Id of the user. (Foreign key -> Users table)
* Rank: The rank of the place from a user. (Integer)

### Users table

It stores the user’s data.

Fields:

* UserId: Unique field which identify the user. (Primary key)
* Name: The name of the user. (Char)
* Password: The password for the user account. (Hash)

### Visits table

It stores the visits of the users.

Fields:

* Id: Unique field which identify the visit. (Primary key)
* UserId: The Id of the user. (Foreign key -> Users table)
* PlaceId: The Id of the place. (Foreign key -> Places table)
* Date: Date of the visit. (Date)

### Categories table

It stores the details of the categories.

Fields:

* CategoryId: Unique field which identify the category. (Primary key)
* Name: The name of the category. (Char)

### Titles table

It stores the titles for each category.

Fields:

* CategoryId: Identify the category. (Primary key -> Categories table)
* Level: The level of the title. (Integer, 1-3)
  + 1: Gold title
  + 2: Silver title
  + 3: Bronze title
* Title: The title with words. (Char)

## Android application side

The application stores some data because it wants to work without Wi-Fi too. It uses SQL database.

### Places table

It’s like the server’s places table. Save the places to make markers even the application can’t connect to the server.

### Ranks table

The application saves the user’s ranks. It stores these records until the application sends to the server. If it is empty, it means every rank has been uploaded.

The table is like the server side table but there isn’t User Id field. The server will add it.

### Visits table

I’s familiar with the Ranks table. It stores the user’s visits until the application upload to the server. If it is empty, it means the server knows about every visit.

The table is like the server side table but there isn’t User Id field. The server will add it.

### Details table

The application doesn’t want to store all of the visits but it has to know the number of the visits in each category.

Fields:

* CategoryId: Unique field which identify the category. (Primary key)
* Visits: The number of the user’s visits in the category. (Integer)

### Titles table

The application store the data to show the right title.

The table is like the server side table.

### Categories table

It stores the categories.

The table is like the server side table.

# Backend

## Download the locations

* Path: /locations
* Method: GET
* Response body: The list of the places in JSON format.
  + Keys: Id, Name, Address, Latitude, Longitude, Category, Phone, Web, Rate

## Add a location

* Path: /locations
* Method: POST
* Request body: The details of a place in JSON format.
  + Keys: GoogleId, Name, Address, Latitude, Longitude, Category, Phone, Web
* Response body: Information about the result.
  + Keys: Status?
  + Values: success, error

## Download the categories

* Path: /categories
* Method: GET
* Response body: The list of the categories in JSON format.
  + Keys: CategoryId, Name

## Download the titles

* Path: /titles
* Method: GET
* Response body: The list of the titles in JSON format.
  + Keys: CategoryId, Level, Title

## Download user’s details

* Path: /visits/<username>
* Method: GET
* Response body: List with the number of
  + Keys: CategoryId, Visits

## Upload the visits

* Path: /visits
* Method: POST
* Request body: The list of the visits in JSON format.
  + Keys: PlaceId, Date
* Response body: Information about the result.
  + Keys: Status
  + Values: success, error

## Rate a place

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

## Time planning and workload

* Create the planning document
* Design the categories, the titles and the icons
* Collect the places from Google
* Create the core of the Android application (with unimplemented activities)
* Create the search activity
* Create the profile activity
* Finish the main activity
* Create the final document