LOCAL TIPS

Capstone Design

Oumaïma Benaboud

Supervised by:

Dr. Nasser Assem

Fall 2018
LOCAL TIPS

Capstone Report

Student Statement:

I, Oumâïma Benaboud, assert that I have applied ethics to the design process and in the selection of the final proposed design. I also affirm that I have held the safety of the public to be paramount and have addressed this in the presented design wherever may be applicable.

_______________________  Oumâïma Benaboud_______________________

Approved by the Supervisor

Dr. Nasser Assem
ACKNOWLEDGEMENTS

I would like to express my sincere gratitude towards my supervisor Dr. Assem Nasser, for accepting to be my capstone supervisor. Dr. Nasser is one of the best professors at AUI and I am extremely thankful for his support, availability, and constructive advices. This capstone report would not have been possible without the contribution and help of these important factors. My deepest gratitude goes to:

- Almighty Allah for giving me the strength to complete my capstone and prepare this report.
- Al Akhawayn University, the School of Science and Engineering, and the capstone coordinator Dr. Yassine Salih Alj for his precious advices.
- My parents, who always believed in me, supported me, and helped me achieve my goals.
- My aunt Fadila, and my brothers Zakaria and Mahdi for their love and affection.
- My friends for their moral support and friendship.
- Last but not least, my dear friends Sara and Lamyae, who have always been by my side during my journey at AUI.
# TABLE OF CONTENTS

1. **INTRODUCTION** ..................................................................................................................... 1

2. **PROJECT SCOPE** ................................................................................................................... 2

3. **STEEPLE ANALYSIS** ............................................................................................................. 2
   3.1. Socio-cultural .................................................................................................................. 2
   3.2. Technological ................................................................................................................... 2
   3.3. Economical ....................................................................................................................... 3
   3.4. Environmental .................................................................................................................. 3
   3.5. Political ............................................................................................................................. 3
   3.6. Legal .................................................................................................................................. 3
   3.7. Ethical ................................................................................................................................ 3

4. **PROJECT METHODOLOGY** .................................................................................................... 4

5. **FEASIBILITY STUDY** .......................................................................................................... 5
   5.1. Time feasibility ................................................................................................................ 5
   5.2. Technological feasibility .................................................................................................. 6
   5.3. Economical feasibility ...................................................................................................... 6
   5.4. Market feasibility .............................................................................................................. 6

6. **SOFTWARE SPECIFICATIONS** .......................................................................................... 7
   6.1. Software specifications requirement .................................................................................. 7
       6.1.1. *Functional requirements* ....................................................................................... 7
6.1.2. Non-Functional requirements ................................................................. 8

6.2. Software specifications analysis ............................................................... 11

6.2.1. Use case diagram .............................................................................. 11

6.2.2. Use case specifications ....................................................................... 11

7. SOFTWARE DEVELOPMENT ...................................................................... 14

7.1. Software design ..................................................................................... 14

7.1.1. Architectural design .......................................................................... 14

7.1.2. Detailed design .................................................................................. 17

7.2. Software implementation ........................................................................ 19

7.2.1. Dependencies used ............................................................................ 19

7.2.2. Screenshots of the application ......................................................... 21

8. SOFTWARE VALIDATION ............................................................................. 25

9. TECHNOLOGY ENABLERS ......................................................................... 26

10. CONCLUSION ............................................................................................ 27

11. FUTURE WORK .......................................................................................... 28

12. REFERENCES ............................................................................................. 29

13. APPENDIX A: SCREENSHOTS OF THE CODE ........................................... 30
LIST OF FIGURES

Figure 4.1 The Incremental Model [1] .............................................................................................. 4
Figure 6.1. Use Case Diagram ........................................................................................................ 11
Figure 7.1. Architecture of The Application .................................................................................... 14
Figure 7.2 Json Object For Service .................................................................................................. 15
Figure 7.3 Json Object for Service Owner ........................................................................................ 15
Figure 7.4 Sequence Diagram: Login Use Case ............................................................................... 16
Figure 7.5 Sequence Diagram: Adding a Service Use Case ............................................................. 16
Figure 7.6 Class Diagram ................................................................................................................ 17
Figure 7.7 Activity Diagram .............................................................................................................. 18
Figure 7.9 Firebase SDK Dependencies .......................................................................................... 19
Figure 7.8. Firebase Project Main Page ............................................................................................ 19
Figure 7.10 Firebase Authentication Methods ................................................................................ 20
Figure 7.11 Google Maps and Places API Dependencies ................................................................ 20
Figure 7.12 Splash Activity .............................................................................................................. 21
Figure 7.13 List View Activity with Bottom Navigation Menu ........................................................ 21
Figure 7.14 List View of Near Hospitals .......................................................................................... 22
Figure 7.15 Map View of Near Restaurants .................................................................................... 22
Figure 7.16 Registration Activity ..................................................................................................... 23
Figure 7.17 Login Activity ................................................................................................................. 23
Figure 7.18 Adding or Updating Services ......................................................................................... 24
Figure 7.19 Add Service Activity .................................................................................................... 24
Figure 13.1 Screenshot of The Source Code of The Splash Activity .................................................. 30
Figure 13.2 Screenshot of The Source Code of The List View Activity .......................................... 30
Figure 13.3 Screenshot of The Source Code of The Map Activity ................................................... 31
Figure 13.4 Screenshot of The Source Code of The Service Class ................................................... 31
Figure 13.5 Screenshot of The Source Code of The Login Activity .................................................. 32
Figure 13.6 Screenshot of The Source Code of The Registration Activity ....................................... 32
Figure 13.7 Screenshot of The Source Code of The Add Service Activity ....................................... 33
Figure 13.8 Screenshot of The Source Code of The Update Service Activity .................................... 33
LIST OF TABLES

Table 5.1.1 Time Management........................................................................................................ 5
Table 6.2.1 First Use Case Specification....................................................................................... 11
Table 6.2.2 Second Use Case Specification ................................................................................... 12
Table 6.2.3 Third Use Case Specification ....................................................................................... 12
Table 6.2.4 Fourth Use Case Specification ..................................................................................... 12
Table 6.2.5 Fifth Use Case Specification ......................................................................................... 13
Table 6.2.6 Sixth Use Case Specification ......................................................................................... 13
Table 6.2.7 Seventh Use Case Specification ................................................................................... 13
RÉSUMÉ

Le fruit de ce projet de fin d'études est une application mobile dédiée à la recherche d'établissements à proximité. Local Tips va servir ses utilisateurs d'une plateforme pour chercher les services aux alentours et les classifier selon leur contigüïté. L’application mobile aura également des filtres de recherches qui permettront de restreindre les choix en fonction du type de service. En phase de conception, nous identifierons les composants de l’application tels que les acteurs et les classes. Différents diagrammes UML seront utilisés pour la modélisation de l’application. Comme nous allons utiliser une méthode agile de développement, nous allons d'abord développer et tester chaque unité indépendamment. Ensuite, nous effectuerons un test d’intégration, au cours duquel nous testerons toutes les unités sous forme de groupes. Nous exécuterons par la suite un test du système global pour vérifier si l’application répond à nos attentes. L'application sera principalement destinée aux utilisateurs d'Android. Elle respectera les normes d’éthiques en protégeant la confidentialité des informations des utilisateurs et en vérifiant la conformité des informations de chaque service.
ABSTRACT

The aim of this capstone is creating a mobile application that will serve as a local city guide allowing users to search for nearby hospitality services. The mobile application will also have search tools that ensure you to narrow down choices according to the type of service. In the design phase, we will be more concerned with the identification of the application components, such as actors and classes. We will also use different UML diagrams to illustrate this phase. In it last part; we will determine the appropriate useful technology enablers. As we will be using an agile methodology, we will implement and test each unit independently. Then, we will perform an integration testing where we will be testing all units as groups. We will then execute an overall system testing to check if the mobile application meets the requirements. The application will be mainly for Android users. It will also satisfy ethical standards by protecting user information for confidentiality, and by verifying the validity of the information of each service.
1. INTRODUCTION

Are you tired of looking for a hotel? Do you feel hungry and in need of restaurant suggestions? Are you a tourist who wants to explore the city like a local? Local Tips is definitely the solution you did not know you needed. This mobile application will serve as a local city guide that will provide you with a list of services based on your location. Local Tips will not be used as a search application for nearest services only, but it will also help local service owners to feature their businesses and attract more customers.

We live in a world where the search for information happens in a blink of an eye. Hence, an application that not only makes use of existing solution like Google Maps, but also filters these solutions to come up with accurate and complete information makes the process easier and more satisfactory.

The mobile application provides its users with different types of services like restaurants, hospitals, hotels and shops. Depending on the chosen type of service, a display of the nearest services will appear. To help service owners reach out to more customers, Local Tips provides a functionality to feature their services in the application.

Local Tips is a GPS based mobile application that requires both location permission and an internet connection. All information used within the application is stored in a secured real time database.
2. PROJECT SCOPE

Local Tips is a mobile application designed especially for android devices. This project is a local search application that allows its users to search for the nearest services like restaurants, hospitals, hotels, and banks. The project will be implemented during the fall 2018 semester for the Capstone Design course.

3. STEEPLE ANALYSIS

The STEEPLE analysis is needed in order to evaluate the influence of our application on external factors, i.e., socio-cultural, technological, economical, ecological, political, legal, and ethical factors.

3.1. Socio-cultural

With the improvement of the Smartphone industry, people are more and more addicted to their phones and the services they provide. Phone users are relying on Online Services even when performing basic daily tasks like choosing a restaurant to eat at. Here comes the utility of Local Tips, as it provides its users with up to date information about nearest services and provide them with a map to reach their desired destination.

3.2. Technological

New technologies gave us the opportunity to improve the quality of our lives. Using this application will not only ease the search for nearby services; it will also help the services to reach out to more customers.
3.3. Economical

This application’s development was possible thanks to open source tools like Android Studio, it will be free and available for everyone. However, Local Tips will ensure the growth of the local services’ business and attraction of new customers.

3.4. Environmental

Local Tips does not have any impact on the environment since it does not make use of any environmental resource.

3.5. Political

This project does not have any political implication and does not influence any political entity.

3.6. Legal

The application follows all legal regulations, as it is designed using open source tools, respects copyrights, and secures the confidentiality of its users.

3.7. Ethical

Local Tips will not violate the privacy of its users, as its main objective is to provide them with navigation assistance in order to explore any city like locals.
4. PROJECT METHODOLOGY

The Agile Incremental Model is the methodology I followed in the development of the project. It gives priority to the final product and to the involvement of the customer during the development process. This model allowed me to deliver individual features and speed up the software’s process. The first module is integrated with the next features and is released as a new increment.

![Incremental Model Diagram]

Figure 4.1 The Incremental Model [1]

The first increment I delivered was the splash activity of the application. This activity should be simple: include texts to welcome the users. The next activity prompts the users to choose the type of service desired. The second increment was the activity that shows the near services to the user’s location in a list or a map fragment. The following increment allows the users to choose their destination and provides them with directions to reach it. The last one is allowing service owners to register in the application and add their services in the database.
5. FEASIBILITY STUDY

As I learned in the Software engineering course, every software development should follow a well-defined process. Before the start of the process, it is necessary to conduct a feasibility study to determine if the project is practical or not. This study will take into consideration different factors, i.e. time, technological and economical factors as well as market.

5.1. Time feasibility

One of the main constraints of this project is time. To ensure delivering a ready to use product by the due date, I tried managing my time following a precise plan. The time management of my capstone project was as follows:

<table>
<thead>
<tr>
<th>Week</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Project Selection</td>
</tr>
<tr>
<td>2</td>
<td>Initial Specification</td>
</tr>
<tr>
<td>3</td>
<td>Feasibility Study and Analysis</td>
</tr>
<tr>
<td>4-7</td>
<td>Interim Report</td>
</tr>
<tr>
<td>8-12</td>
<td>Software Design and Implementation</td>
</tr>
<tr>
<td>13</td>
<td>Software Validation, Final Report and Professional CV</td>
</tr>
<tr>
<td>14</td>
<td>Project Defense</td>
</tr>
<tr>
<td>15</td>
<td>Updated Final Report</td>
</tr>
</tbody>
</table>
5.2. Technological feasibility
After the evaluation of all the available tools for mobile application development, I chose to build Local Tips as a native mobile application on Android as an operating system. The IDE used to implement the application is Android Studio. I implemented the application using Java as a programming language for the front end. For the back end, I opted for Firebase Database as it provides us with a real time database. This project is technically feasible as the required tools for design are available with tutorials and documentations for help [2].

5.3. Economical feasibility
As we are using open source tools for the design of the application, it will be free. However, we will be using Ad Mob to generate revenues to pay Google APIs fees. Google APIs are free only for a limited number of users with a daily quota of use, so after the deployment of the application, we will need to pay additional fees to use these APIs. [4]

5.4. Market feasibility
Google Maps offer the services offered by Local Tips; however, in Morocco it does not feature all near establishments. This application will allow service owners to register and be featured in the application. Therefore, Local Tips will help them advertise their services and flourish their local market.
6. SOFTWARE SPECIFICATIONS

6.1. Software specifications requirement

6.1.1. Functional requirements

6.1.1.1. Choose a service type

- **User requirement**
  The users shall **choose** the type of service they are searching for.

- **System requirement**
  The users shall **click** on the icon of the type of service they are searching for.

6.1.1.2. Choose a service

- **User requirement**
  The users shall **choose** their destination.

- **System requirement**
  The users shall **click** on their desired service from the list of nearest services given by the application.

6.1.1.3. Choose the map view

- **User requirement**
  The user shall **choose** the map view to see the nearest services according to the map.

- **System requirement**
  The user shall **click** on the button of map view to see the nearest services in a map fragment.

6.1.1.4. Check the desired service’s information

- **User requirement**
  The users shall **check** the information about their desired service.

- **System requirement**
  The user shall **click** on the desired service to see more information about it.
6.1.1.5. Login for service owners

- **User requirement**
  The service owners shall **login** to the system by entering their emails and corresponding passwords.

- **System requirement**
  The service owners shall **login** to be able to register a new service or to edit information about it.

6.1.1.6. Register a service

- **User requirement**
  The service owners shall **register** to add a new service to the database.

- **System requirement**
  The service owners shall **enter** information like their name, location and type of their services.

6.1.1.7. Edit service information

- **User requirement**
  The service owners shall **edit** the information about his/her service.

- **System requirement**
  The service owners shall **have** access to modify information about their services.

6.1.2. Non-Functional requirements

6.1.2.1. Usability requirements

- **User requirement**
  All users shall **complete** any task offered by the application without assistance.

- **System requirement**
  The interface of the application shall **have** clear buttons, be simple and easy to operate for any user and have a simple design.
6.1.2.2. Efficiency requirements

6.1.2.2.1. Performance requirements

- **User requirement**
The system shall *execute* in the minimum possible time.

- **System requirement**
The system shall *allow* the users to find the nearest services, information about each service and directions to reach them in the minimum amount of time possible.

6.1.2.2.2. Space requirements

- **User requirement**
The application shall *use* 15 megabytes of memory.

- **System requirement**
The system is a mobile application that shall *have* a maximum size of 15 megabytes.

6.1.2.3. Dependability requirements

- **User requirement**
The product shall not *depend* on any factors other than internet connection and GPS location.

- **System requirement**
The user shall use the application without a need for any factor other than internet access and GPS location.

6.1.2.4. Organizational requirements

6.1.2.4.1. Operational requirements

- **User requirement**
The application shall usually *operate* effortlessly.

- **System requirement**
The customer shall use the application without assistance. The software shall *operate* successfully 99% of the time.
6.1.2.4.2. Development requirements

- **User requirement**
  The software shall be developed using Android Studio as IDE.

- **System requirement**
  The developer shall use Android Studio as IDE. There are no other limitations for the implementation.

6.1.2.5. External requirements

6.1.2.5.1. Regulation requirements

- **User requirement**
  The application shall notice the user of the updates that will happen.

- **System requirement**
  The application shall send a notice to the users each time a new version is developed and ask them to update.

6.1.2.5.2. Ethical requirements

- **User requirement**
  The application shall be acceptable to the user and the public.

- **System requirement**
  The application shall respect all the ethical and professional code given by the Association for Computing Machinery.
6.2. Software specifications analysis

6.2.1. Use case diagram

![Use Case Diagram]

Figure 6.1. Use Case Diagram

6.2.2. Use case specifications

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Choose service type</td>
</tr>
<tr>
<td>Description</td>
<td>The user clicks on the icon of the desired service type.</td>
</tr>
<tr>
<td>Actor</td>
<td>User</td>
</tr>
<tr>
<td>Precondition</td>
<td>Launch the mobile application</td>
</tr>
<tr>
<td>Main flow</td>
<td>Press on the launching icon.</td>
</tr>
<tr>
<td></td>
<td>The mobile application will start</td>
</tr>
<tr>
<td></td>
<td>The icons will show up.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The list activity will be displayed for the user</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Go back to the home page of your phone</td>
</tr>
</tbody>
</table>

Table 6.2.1 First Use Case Specification
### Table 6.2.2 Second Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Choose service from list</td>
</tr>
<tr>
<td>Description</td>
<td>The user clicks on a service from list of near services.</td>
</tr>
<tr>
<td>Actor</td>
<td>User</td>
</tr>
<tr>
<td>Precondition</td>
<td>The user clicked on a service type</td>
</tr>
<tr>
<td>Main flow</td>
<td>Select a service</td>
</tr>
<tr>
<td></td>
<td>New activity will start.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The information activity is launched</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Go back to the main screen</td>
</tr>
</tbody>
</table>

### Table 6.2.3 Third Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>003</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Choose the map view</td>
</tr>
<tr>
<td>Description</td>
<td>The user clicks on the button to switch to the map view.</td>
</tr>
<tr>
<td>Actor</td>
<td>User</td>
</tr>
<tr>
<td>Precondition</td>
<td>The user clicked on a service type</td>
</tr>
<tr>
<td>Main flow</td>
<td>Press on the button.</td>
</tr>
<tr>
<td></td>
<td>New activity will start.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The map view is loaded</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Go back to the list view</td>
</tr>
</tbody>
</table>

### Table 6.2.4 Fourth Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>004</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Check service information</td>
</tr>
<tr>
<td>Description</td>
<td>The user clicks on the desired service to have more information about it.</td>
</tr>
<tr>
<td>Actor</td>
<td>User</td>
</tr>
<tr>
<td>Precondition</td>
<td>The user clicked on a service from the list view</td>
</tr>
<tr>
<td>Main flow</td>
<td>Select item.</td>
</tr>
<tr>
<td></td>
<td>Information activity will start.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The information about the service are shown in the screen</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Go back to the list view</td>
</tr>
</tbody>
</table>
### Table 6.2.5 Fifth Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>005</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Login</td>
</tr>
<tr>
<td>Description</td>
<td>The service owners log in the application using their email and password.</td>
</tr>
<tr>
<td>Actor</td>
<td>Service owner</td>
</tr>
<tr>
<td>Precondition</td>
<td>The service owner clicks on the registration and management link</td>
</tr>
<tr>
<td>Main flow</td>
<td>Click on the registration link. Log in to the application.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The service owners can register a new service, manage existing one, or log out.</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Go back to the main screen</td>
</tr>
</tbody>
</table>

### Table 6.2.6 Sixth Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>005</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Register Service</td>
</tr>
<tr>
<td>Description</td>
<td>The service owner fills a chart with different information about the service such as name and location.</td>
</tr>
<tr>
<td>Actor</td>
<td>Service owner</td>
</tr>
<tr>
<td>Precondition</td>
<td>Log in to the application</td>
</tr>
<tr>
<td>Main flow</td>
<td>Click on the registration and management link. Log in to the application. Information activity will start.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The information about the service are stored in the database</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Log out</td>
</tr>
</tbody>
</table>

### Table 6.2.7 Seventh Use Case Specification

<table>
<thead>
<tr>
<th>Use Case Id</th>
<th>007</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name</td>
<td>Manage service information</td>
</tr>
<tr>
<td>Description</td>
<td>The service owners edit information about their services.</td>
</tr>
<tr>
<td>Actor</td>
<td>The service owner</td>
</tr>
<tr>
<td>Precondition</td>
<td>Log in to the application</td>
</tr>
<tr>
<td>Main flow</td>
<td>Click on the registration link. Log in to the application. Editing activity will start.</td>
</tr>
<tr>
<td>Post Condition</td>
<td>The new information are saved in the database</td>
</tr>
<tr>
<td>Alternatives</td>
<td>Log out</td>
</tr>
</tbody>
</table>
7. SOFTWARE DEVELOPMENT

7.1. Software design

7.1.1. Architectural design

For the system architecture, Local Tips requires a backend database. I chose Firebase database, which is a backend as a service database (BAAS) that provides us with different services like authentication and cloud storage [5]. These services are easily integrated in the application using different APIs. As Firebase is a real time database, it displays the real time processed data to the client in a short time. To ensure communication between the different tiers of this architecture, we use two APIs: REST API and HTTPS. Firebase processes the HTTPS requests from the client’s side, returns, and displays the stored data to the client’s side.
7.1.1.1. Data Modeling

Data in Firebase real time database is stored as JSON object. Firebase is a No SQL database where added data is represented as: a new node in a JSON tree with a related key. To integrate Firebase in our mobile application, we need to provide a SHA-1 fingerprint of the application to authenticate. These measures ensure a secure connection between the database and the client side. Examples of JSON object used in the database are as follows:

![JSON Object for Service](https://console.firebase.google.com/project/androidnearbyplaces-222421/database/androidnearbyplaces-222421)

Figure 7.2 JSON Object for Service

![JSON Object for Service Owner](https://console.firebase.google.com/project/androidnearbyplaces-222421/database/androidnearbyplaces-222421)

Figure 7.3 JSON Object for Service Owner

The services are identified using four characteristics: name, address, type of the service and the location using two parameters i.e. the latitude and longitude.

Service owners are identified with a user ID provided automatically by Firebase. This user ID is created once the service owners create their account.
7.1.1.2. Sequence diagram

![Sequence Diagram of Login Use Case](image1)

**Figure 7.4 Sequence Diagram of Login Use Case**

![Sequence Diagram of Adding a Service Use Case](image2)

**Figure 7.5 Sequence Diagram of Adding a Service Use Case**
7.1.2. Detailed design

7.1.2.1. Class diagram

Figure 7.6 Class Diagram
7.1.2.2. Activity diagram

Figure 7.7 Activity Diagram
7.2. Software implementation

7.2.1. Dependencies used

To start implementing, I had to integrate Firebase in my application. The first step is adding the project to the Firebase console and secure this connection by providing the SHA-1 fingerprint of the application.

Figure 7.8. Firebase Project Main Page

We will then add a Google service JSON file to the application package and add the dependencies for Firebase SDK into the Gradle File.

```java
implementation 'com.google.firebase:firebase-core:16.0.5'
implementation 'com.google.firebase:firebase-database:16.0.5'
implementation 'com.google.firebase:firebase-auth:16.0.5'
```

Figure 7.9 Firebase SDK Dependencies
As you can see, Firebase provides its users with different registration modes. For a first release, I chose email address and password authentication. Other methods are to be included in the upcoming releases.

```
implementation 'com.google.android.gms:play-services-maps:16.0.0'
implementation 'com.google.android.gms:play-services-places:16.0.0'
implementation 'com.google.android.gms:play-services-location:16.0.0'
```

I used Google Maps API and Google Places API by signing in the Google API Console. I added the dependencies in the figure above to the Gradle file. The API console gives us an API key that we use to send HTTP requests; the retrieved results are in the form of JSON file.
7.2.2. Screenshots of the application

This splash activity starts once the users launch the application. Its duration is five seconds. During this time, the users can wait or click on the “Join as a service owner” button if they want to register in the application.

This is the list view activity, where the users have access to a list of nearby services depending on its type. In this screenshot, the user clicked on the “Hotel Icon”. The users can also click on the button go to map, to see near services on a map fragment.
This is the list view activity, where the users have access to a list of nearby hospitals.

Figure 7.14 List View of Nearby Hospitals

As we mentioned before, the “Go to Map” button allows the users to see the service in a map fragment.

Figure 7.15 Map View of Nearby Restaurants
This registration activity will be displayed for new service owners that want to join the platform. The users need to provide their email and a password to register.

![Registration Activity](image)

**Figure 7.16 Registration Activity**

This Login Activity is for service owners who already have an account. They just have to log in using the same email and password used for the registration.

![Login Activity](image)

**Figure 7.17 Login Activity**
After logging in to the application, the service owners can either add a new service to our database or update information about the services they already added.

Figure 7.18 Adding or Updating Services

The service owners need to fill the following information in order to add their services in the database, and be featured in the application.

Figure 7.19 Add Service Activity
8. SOFTWARE VALIDATION

Testing is one of the most important activities of the software development process; it helps us identify the bugs and faults in our project. Unit testing is the first step in the software testing process, in which I tested the different activities independently. I started by performing this testing on the map activity to see if the retrieved results change if we alter the location. The following step is the integration testing, where we integrate all units and test them in groups. In this step, I mainly focused on checking how the activities interact when we click on buttons to move from one activity to another. The last step is system testing, in which I tested the application as a whole to see if it meets the requirements.

One of the numerous advantages of using the incremental model is optimizing the time spent in the testing phase. As the project is implemented and tested incrementally until the last increment.
9. TECHNOLOGY ENABLERS

<table>
<thead>
<tr>
<th>Image</th>
<th>Technology</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="android_sdk.png" alt="Android SDK" /></td>
<td>Android SDK</td>
<td>Android Software Development Kit was used to develop the mobile application</td>
</tr>
<tr>
<td><img src="android_studio.png" alt="Android Studio" /></td>
<td>Android Studio</td>
<td>Android Studio was used as an integrated development environment</td>
</tr>
<tr>
<td><img src="java.png" alt="Java" /></td>
<td>JAVA</td>
<td>Java was used as programming language for the client side of the application</td>
</tr>
<tr>
<td><img src="xml.png" alt="XML" /></td>
<td>XML</td>
<td>XML was used as markup language to design the user interface and different activity layouts</td>
</tr>
<tr>
<td><img src="firebase.png" alt="Firebase" /></td>
<td>Firebase</td>
<td>Firebase database was used for the backend client</td>
</tr>
<tr>
<td><img src="google_maps.png" alt="Google Maps Platform" /></td>
<td>Google Maps Platform</td>
<td>Google Maps API and Google Places APIs were used in the implementation of Local Tips. These APIs are provided by Google Maps Platform.</td>
</tr>
<tr>
<td><img src="creately.png" alt="Creately" /></td>
<td>Creately</td>
<td>Creately is an online tool used to draw the different UML diagrams</td>
</tr>
</tbody>
</table>
10. CONCLUSION

This capstone project served as a great opportunity to put in practice all the skills and concepts I learned during my journey here at AUI. Thanks to Dr. Assem’s suggestion, I was able to develop an interesting and useful mobile application that helps its users to find the nearest services in their location. Conducting a feasibility study was the first step in the development of this project, followed by the elicitation and prioritization of the requirements. Designing and implementing the application was my next step by first gathering enough information about the different technology enablers and deciding on which technology to use. As I followed the agile methodology to develop this project, testing was performed simultaneously with the implementation.
11. FUTURE WORK

For future work, numerous functionalities could be added to the application. Due to time constraints, these features were not fully developed. For starters, we need to expand our application and database to cover different type of services in the whole country. Another feature is including the different sign up methods offered by Firebase such as Phone and Facebook. We should also allow users to write reviews about the services, rate them and react to reviews of other users. Eventually, we will need to develop an administrator platform to manage the services, and remove them for non-compliant information, among other reasons. In the long term, AdMob will be used to generate revenues needed to pay for the utility of Google Maps API and Firebase services.
12. REFERENCES


Google Maps Platform Documentation, Google Maps Platform retrieved from https://developers.google.com/maps/documentation/ 10 November 2018


13. APPENDIX A: SCREENSHOTS OF THE CODE

Figure 13.1 Screenshot of the source code of the Splash Activity

Figure 13.2 Screenshot of the source code of the List View Activity
Figure 13.3 Screenshot of the source code of the Map Activity

Figure 13.4 Screenshot of the source code of the Service Class

31
Figure 13.5 Screenshot of the source code of the Login Activity

Figure 13.6 Screenshot of the source code of the Registration Activity
Figure 13.7 Screenshot of the source code of the Add Service Activity

```java
package com.example.hp.androidnearbyplaces;

import ...

public class AddServiceActivity extends AppCompatActivity {

    EditText newserviceName, newserviceAddress, newserviceLog, newserviceLat;
    button btnadd;
    EditText newserviceType;
    FirebaseDatabase database;
    DatabaseReference ref;
    service newservice;
    FirebaseAuth auth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_service);

        newserviceName = (EditText) findViewById(R.id.newserviceName);
        newserviceAddress = (EditText) findViewById(R.id.newserviceAddress);
        newserviceLog = (EditText) findViewById(R.id.newserviceLog);
        newserviceLat = (EditText) findViewById(R.id.newserviceLat);
        btnadd = (Button) findViewById(R.id.btnadd);
        database = FirebaseDatabase.getInstance();
    }
}
```

Figure 13.8 Screenshot of the source code of the Update Service Activity

```java
package com.example.hp.androidnearbyplaces;

public class UpdateServiceActivity extends AppCompatActivity {

    EditText updserviceName, updserviceAddress, updserviceLog, updserviceLat;
    Button btnupdate;
    EditText updserviceType;
    FirebaseDatabase database;
    DatabaseReference reference;
    service updservice;
    FirebaseAuth auth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_update_service);

        updserviceName = (EditText) findViewById(R.id.updserviceName);
        updserviceAddress = (EditText) findViewById(R.id.updserviceAddress);
        updserviceLog = (EditText) findViewById(R.id.updserviceLog);
        updserviceLat = (EditText) findViewById(R.id.updserviceLat);
        btnupdate = (Button) findViewById(R.id.btnupdate);
        database = FirebaseDatabase.getInstance();
    }
}
```