School of Science and Engineering

EGR 4402: Capstone Design

Mobile Application for GPS-Locator of Shops: E-Picerie

Yassir Lemallem

Supervisor: Pr. Omar Iraqi Houssaini
Mobile Application for Online Shopping: E-Picerie

Capstone Report

Student Statement:

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

Yassir Lemallem

Approved by the Supervisor(s)

Pr. Omar Iraqi Houssaini
ACKNOWLEDGEMENTS

I would like to acknowledge Pr. Omar Iraqi Houssaini for his continuous assistance and relentless hard work. Without his contribution, this project would have never been what it is today. Your priceless advices have been of a great help to this work. I also would like to thank my parents without whom, I could never have become the man I am today. I would like to thank all my friends with whom I have shared these wonderful years, and these same people who encouraged me and helped during my last semester. Finally, I would like to thank Al Akhawayn University as a whole, because without my experience here, I would have never opened my eyes on so many things.
Table of Contents

I. Introduction ...................................................................................................................... 6
II. Feasibility Study ............................................................................................................. 7
III. STEEPLE Analysis ...................................................................................................... 10
IV. Methodology .............................................................................................................. 11
V. Requirements Specification .......................................................................................... 13
    Functional Requirements ............................................................................................ 15
    Use Case Diagram ....................................................................................................... 20
    Non-Functional Requirements .................................................................................... 21
    Schedule ...................................................................................................................... 23
VI. System Design and Architecture .................................................................................. 24
    Entity Relationship Diagram ..................................................................................... 26
    Sequence Diagram ...................................................................................................... 30
    Class Diagram ............................................................................................................ 32
VII. Technologies Enablers ............................................................................................... 33
VIII. Development and Implementation ............................................................................ 36
IX. Results .......................................................................................................................... 39
X. References ..................................................................................................................... 42
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Someone who interacts with the mobile phone application.</td>
</tr>
<tr>
<td>Admin/Administrator</td>
<td>System administrator who is given specific permission for managing and controlling the system.</td>
</tr>
<tr>
<td>Shop Owner</td>
<td>Someone who has a shop and wants his shop to be a part the application.</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GPS-Navigator</td>
<td>An installed software on mobile phone which could provide GPS connection and data, show locations on map and find paths from current position to defined destination</td>
</tr>
<tr>
<td>Application Store</td>
<td>An installed application on mobile phone which helps user to find new compatible applications with mobile phone platform and download them from Internet</td>
</tr>
<tr>
<td>DEP</td>
<td>Dependency</td>
</tr>
</tbody>
</table>
Abstract

The goal of this capstone is to create an application that will help people shop from their nearest shops by just using their smartphones. It aims to make the process faster, and show the nearest shops that surrounds the application users.

I had the idea once when I really needed to find a shop in a city I wasn’t familiar with its neighbourhoods. I thought that an application that would show me all the shops surrounding my location would be very useful. It would save me a lot of time driving around trying to find one. I shared the idea with some of my friends who found it very innovative and encouraged me to work on it.

I am very pleased to work on this idea as my capstone project, because I never thought that those random ideas you have when walking in the streets, or while sitting with your friends, would have such an impact on my academic life.

This report shows all the different steps I followed in order to create this application.
I. Introduction

The application, which goes by the name of “E-Picerie”, will not only help people find the stores, it will also help their owners make their stores more visible by listing them into the application. The application will use a third-party web application to display the stores, near the user’s location, on an interactive map allowing him to see the details about the stores, such as its exact address, how far it is located, and other information. It will also allow the users to buy groceries online if the store accepts this kind of transaction, and also ask for a home delivery. The stores are listed at the moment by collecting their locations in the city of Ifrane to launch the application. The additional information such as, the opening hours, the products offered to online shopping, etc. were gathered through meetings with the shop owners. But later, in order to expand the application in other regions of Morocco, a feature allowing the shop owners to register their stores will be provided in order

EPicerie is a GPS-based mobile application which helps people to find the closest shops based on the user’s current position. The application should be free to download from either a mobile phone application store or similar services. After registering, the shop owners will be able to provide information about their stores, and manage them using a different interface from what a casual user would have. This information will act as the bases for the search results displayed to the user, if he wishes to perform a detailed search. An administrator also uses the web-portal/application in order to administer the system and keep the information accurate. Furthermore, the software needs both Internet and GPS connection to fetch and display the results. All the system information is maintained in a database, which is located on a web-server.
II. Feasibility Study

1. Project Context

*E-Picerie* is a project that aims to locate shops that are close to the user’s location. The concept is to help the users find the shops surrounding them, and scale down the results according to their needs. This application will be implemented as a mobile application since it will be more useful outdoors when the user is looking for the shop.

The application will offer services to three different types of users:

- **Random Users who want to find a shop:**
  - They will be able to create a profile for login and location purposes.
  - Find shops according to their needs (General nutrition, drinks, tobacco shops…)
  - Add/Check ratings

- **Shop Owners who want their shops displayed in the application:**
  - They will also be able to create their profiles for verification purposes.
  - Add a store by specifying its location and the kind of items it offers.

- **Administrators\(^1\)** who will be able to check the shop’s data submitted by the shop owner, via a web portal that will be implemented as a future work.

\(^1\) Since I won’t work on the web portal for the moment, I won’t talk about it nor the administrator’s functionalities.
2. Client Analysis

This project targets all users owning an Android phone for now. It can be very useful for those who need to find a shop in a city they’re just visiting or where they have just moved in. It can be also very useful when most of the shop are closed, it will help them find those who are still open by verifying their opening hours. It will also help the shop owners have more visibility and therefore increase their profits.

3. Market Analysis

The services offered by E-Picerie are not offered in any other application here in Morocco. Such services are offered by Google Maps but in the case of Morocco it doesn’t show all the shops with their details. So this application will be an innovative concept and may seduce a large segment of Android users.

4. Economic Analysis

For now the only costs are the time and human efforts. The application will be implemented using open source tools. There may be some costs generated later on when the application is deployed, since some APIs, Google Maps for instance, require the developer to pay some fees when the application is used by a certain number of users.

5. Technical Analysis

The project is feasible, since all the required tools are available online along with their documentations and tutorials.
6. **Constraints**

The mobile application is constrained by the system interface to the GPS navigation system within the mobile phone. Since it requires the location of both the user and stores, it will be imperative that the user’s phone supports a GPS service.

The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function.

7. **Assumptions and dependencies**

One assumption about the product is that it will always be used on mobile phones that have enough performance. If the phone does not have enough hardware resources available for the application, for example the users might have allocated them with other applications, there may be scenarios where the application does not work as intended or even at all.

**Conclusion**

Even if there are some constraints, *E-Picerie* is worth implementing since it’s a new concept here in Morocco. This tool can be used on a daily basis by people who need help locate the kind of shop they want. It will also change the way customers shop their daily groceries in Morocco. Instead of driving from a store to another asking for a specific item, they could find it just by using this application.
III. STEEPLE Analysis

STEEPLE is an acronym for "social, technological, economic, environmental, political, and legal." The purpose of a STEEPLE analysis is to identify all of the various external factors mentioned before, that might affect my application.

- Social: This application helps people find nearby shops quickly. It facilitates information access about the shops.
- Technology: Nowadays people are more likely to use their mobiles to search for any information. This application is designed to locate the shops near one’s location and view details about them.
- Environmental: This application does not harm the environment; however, it decreases the use of papers which helps solving environmental issues.
- Economic: This application will help the local shops get more organized and more popular through this application.
- Political: This application doesn’t affect any political entity in any way.
- Legal: The implementation of the application will be legal, it will respect copyrights and use open source software’s.
- Ethical: The application will ensure the privacy of user’s information.
IV. Methodology

In order to implement this project, I had to choose one methodology between a variety of methodologies. That chosen methodology would assist me in almost every aspect of this project. Since the project was an idea of mine, along with its functional and nonfunctional requirements, I chose the Agile Scrum methodology since it allows the “Product Owner” to identify and prioritize the system’s functionalities in form of a “Product Backlog”. The Product Backlog consists of features that the project should implement as well as their priorities. It may also have a section that handles the system bugs, but it’s not mandatory.

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between the software developers and the clients. Working with Agile offers the developers many advantages such as:

- Increase the quality of the deliverables
- Cope better with change (and expect the changes)
- Provide better estimates while spending less time creating them
- Be more in control of the project schedule and state
Working with this methodology allowed me to break down the project into small sub-projects, relatively easier to implement, and also to debug.

The Scrum Product Owner uses the Scrum Product Backlog during the Sprint Planning Meeting to describe the top entries to the team. The Scrum Team then determines which items they can complete during the coming sprint.

Each Scrum Product Backlog has certain properties that differentiate it from a simple to-do list:

- The entries in the Scrum Product Backlog are prioritized and ordered accordingly
- The level of detail depends on the position of the entry within the Scrum Product Backlog
- The Scrum Product Backlog is a living document
- There are no action-items or low-level tasks in the Scrum Product Backlog
V. Requirements Specification

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

1. Product Perspective

This system will consist of two parts: one mobile application and one web portal. The mobile application will be used to find shops, view information about them, to display the ratings submitted by the application’s users. While the web portal will be used for managing the accuracy of the information submitted by the shop owners about their stores.

Since the database will be populated by myself in the beginning for testing purposes, I won’t need a web portal for the moment. Later on, when the application goes on a larger scale, the shop owners will have to submit information about their stores, and a manager will have to approve the store if the information submitted are accurate.

The mobile application will need to communicate to a GPS application within the mobile phone. The GPS will provide the mobile application with locations of both the user and the stores, but it will also provide maps and the functionality to display the application’s data on the map.
2. **Product Functions**

   With the mobile application, the users will be able to search for shopping stores. The result will be based on their location. The result of the search will be viewed in a map view, depending on what criteria is included in the search or on the user’s location. The map view will show each shop location as a pin on the map as well as the user’s own location. The users will be able to either select a shop as target destination, get information on how to get there, or view its information like the opening days and hours, the kind of items available, etc.

3. **User Characteristics**

   There are two types of users that interact with the mobile application; users or potential customers, and shop owners. Each of these users has different use of the system so each of them has their own requirements.

   The regular mobile application users can only use the application to find a shop. This means that the regular user have to be able to search for shops, choose a store from that search and then navigate to it. In order for the users to get a relevant search result there are multiple criteria the users can specify and all results matches all of those. The shop owners manage the information about their stores, for example a description of the shop, contact information and the type of merchandise they sell.
4. **Functional requirements**

This section includes the requirements that specify all the fundamental actions of the software system.

### 4.1. Functional requirement 1

ID: FR1  
TITLE: User registration  
DESC: Given that a user has downloaded the mobile application, then the user should be able to register through the mobile application. The user must provide a valid email and a password.  
RAT: In order for a user to register on the mobile application.  
DEP: FR1

### 4.2. Functional requirement 2

ID: FR2  
TITLE: Shop registration  
DESC: Given that a shop owner has registered, then the user should be able to enter the information about his store. The data will be stored in the database and can be retrieved for later display in the map.  
RAT: In order for a user to log in the mobile application.  
DEP: FR1
4.3. Functional requirement 3

ID: FR3

TITLE: User login

DESC: Given that a user has registered, then the user should be able to log in to the mobile application. The login information will be stored on the phone and in the future the user should be logged in automatically.

DEP: FR1

4.4. Functional requirement 4

ID: FR4

TITLE: View map

DESC: Given that a user is logged in to the mobile application, then the first page that is shown should be the map page. The user should be able to browse the different shops, according to his/her location.

- Search results can be viewed on a map. On the map, the relevant and closest shops according to the user’s position are shown.

- A specific pin will represent a specific store location. On each pin there should be an information link.

DEP: FR3
4.5. Functional requirement 5

ID: FR5

TITLE: Navigation to shop

DESC: A user should be able to select a pin on a map. When a selection is made, the location of the shop should be sent to the mobile phone’s GPS-navigation program. The user should then be navigated to the destination. When the destination is reached, a user should be able to go back to the map page on the mobile application.

RAT: To navigate a user to a chosen shop.

DEP: FR4

4.6. Functional requirement 6

ID: FR6

TITLE: Selecting the shop’s information link

DESC: A user should be able to select the information link, which is included on all result items. The link will direct the user to an information page, which includes a picture of the shop, the shop’s name, address, phone number, e-mail address, type of items sold, shop’s description and a menu of available items, description and price of the different items.

DEP: FR3, FR4
4.7. Functional requirement 7

ID: FR7
TITLE: Manage Store Information
DESC: A shop owner should be able to edit the information he filled about his store whenever he/she intends to.
DEP: FR2

4.8. Functional requirement 8

ID: FR8
TITLE: Search by specific item
DESC: A user should be able to select a specific item in a given list as input. The result is displayed in a map view by default.
RAT: In order for a user to search by specific criteria.
DEP: FR3
4.9. Functional requirement 9

ID: FR9

TITLE: Profile page

DESC: On the mobile application, a user should have a profile page. On the profile page a user can edit his/her information, which includes the password, e-mail address and phone number.

RAT: In order for a user to have a profile page on the mobile application.

DEP: FR1, FR3

4.10. Functional requirement 10

ID: FR10

TITLE: Logout

DESC: On the mobile application, a user should be able to log out from the application whenever he/she wants

DEP: FR3
5. Use Case Diagram

This use case diagram shows all the different actions a client or a store owner can perform. The shop owner inherits from the users some of his actions since a shop owner can also be a user in other circumstances. All the features require the user login.

Figure 1: Use Case Diagram
6. Non-Functional Requirements

6.1 Product Requirements

6.1.1. Usability Requirement

The application is user-friendly and provides a simple interface for the user (users and shop owners). It is very easy to understand thanks to the simple and clear user interface.

6.1.2. Efficiency Requirements

6.1.2.1. Performance Requirement

The start time of this application shall be quick. Also, the response time shall not exceed in average 3 seconds since the throughput of operations performed per second is not high.

6.1.2.2. Space Requirement

The system shall not occupy more than 50 Mb.

6.1.2.3. Reliability Requirement

The system shall be reliable and all of its functions shall perform as required.

6.1.1.4. Portability Requirement

The system shall be compatible with Android Operating System.
6.2 Organizational Requirement

6.2.1. Delivery Requirement

_E-Picerie_ System should be delivered by the end of November.

6.2.2. Development Requirement

The application should be developed using Android technologies.

6.3. Legislative Requirements

6.3.1. Privacy Requirement

The application must keep the privacy of its users.

6.3.2. Safety Requirement

The application should not harm the users
7. Project Scheduling

7.1 Schedule

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Start Date</th>
<th>End Date</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Specification</td>
<td>28-Aug</td>
<td>8-Sep</td>
<td>12</td>
</tr>
<tr>
<td>Feasibility Study and Analysis</td>
<td>9-Sep</td>
<td>14-Sep</td>
<td>6</td>
</tr>
<tr>
<td>Diary #1</td>
<td>15-Sep</td>
<td>18-Sep</td>
<td>4</td>
</tr>
<tr>
<td>Diary #2</td>
<td>19-Sep</td>
<td>25-Sep</td>
<td>7</td>
</tr>
<tr>
<td>Diary #3</td>
<td>26-Sep</td>
<td>2-Oct</td>
<td>7</td>
</tr>
<tr>
<td>Diary #4</td>
<td>3-Oct</td>
<td>9-Oct</td>
<td>7</td>
</tr>
<tr>
<td>Diary #5</td>
<td>10-Oct</td>
<td>10-Oct</td>
<td>7</td>
</tr>
<tr>
<td>Interim Report</td>
<td>17-Oct</td>
<td>20-Oct</td>
<td>4</td>
</tr>
<tr>
<td>Diary #6</td>
<td>21-Oct</td>
<td>23-Oct</td>
<td>3</td>
</tr>
<tr>
<td>Diary #7</td>
<td>24-Oct</td>
<td>30-Oct</td>
<td>7</td>
</tr>
<tr>
<td>Diary #8</td>
<td>31-Oct</td>
<td>6-Nov</td>
<td>7</td>
</tr>
<tr>
<td>Diary #9</td>
<td>7-Nov</td>
<td>13-Nov</td>
<td>7</td>
</tr>
<tr>
<td>Final Report</td>
<td>14-Nov</td>
<td>4-Dec</td>
<td>20</td>
</tr>
<tr>
<td>Updated Final Report</td>
<td>5-Dec</td>
<td>12-Dec</td>
<td>8</td>
</tr>
</tbody>
</table>

7.2 Gantt Chart
VI. System Design and Architecture

After the requirement part, I took a long time to think about the kind of architecture I would apply to my project. I wanted to avoid a heavy client and separate the business logic from the UI and these two from the data retrieved from the database. In other words, I wanted to create a three-tiers application.

After intensive reading about the three-tier architecture and its documentation, I concluded that it would make my code:

- Independent of Frameworks
- Testable.
- Independent of UI.
- Independent of Database.
- Independent of any external agency.
After customizing this architecture for my project, I was able to come up with the following structure of layers:

- **Client Side** will include normal Activities and Fragments, which will only handle rendering views with the necessary code to establish connections with the server’s application and send HTTP requests.

- **Server Side** will include all business logic and interact between Data and Client side. The objective is to make this tier independent of anything, so the business logic can be tested without any dependency to external components. It will also handle the call and the logic received by the APIs like Google Maps API to get the location of the user and the shops nearby. When retrieved from the database, it is converted in entity classes using ORM and accessed/persisted using Java Persistence API.

- **Data Side** All the data is stored in the database, MySQL Workbench for instance.
1. Database Design

Since this application follows the three-tier architecture, it will need somewhere to store the data. For that, a database will be used. The mobile application will communicate with the database via Internet by using different tools to fetch or store the data about the user and the store. The database was implemented using Mysql Workbench, following the necessary logic and requirements I designed the tables needed to store data as well as the relationships that exists between some of them.

The following is the ERD of my project.

Figure 2: Entity Relationship Diagram
<table>
<thead>
<tr>
<th>Table</th>
<th>Attribute</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>UserID</td>
<td>Integer</td>
<td>Auto incremented ID for each user in the database.</td>
</tr>
<tr>
<td></td>
<td>FirstName</td>
<td>VarChar(30)</td>
<td>The user’s first name.</td>
</tr>
<tr>
<td></td>
<td>LastName</td>
<td>VarChar(30)</td>
<td>The user’s last name.</td>
</tr>
<tr>
<td></td>
<td>PhoneNumber</td>
<td>Long</td>
<td>The user’s phone number.</td>
</tr>
<tr>
<td></td>
<td>Email</td>
<td>VarChar(50)</td>
<td>The user’s email.</td>
</tr>
<tr>
<td></td>
<td>Password</td>
<td>Text</td>
<td>The user’s encrypted password.</td>
</tr>
<tr>
<td></td>
<td>User_Type</td>
<td>String</td>
<td>The user’s type to differentiate a simple user from a shop owner.</td>
</tr>
<tr>
<td></td>
<td>PhotoURL</td>
<td>Text</td>
<td>The user’s profile picture.</td>
</tr>
<tr>
<td>Shop</td>
<td>ShopID</td>
<td>Int</td>
<td>Auto incremented ID for each shop in the database.</td>
</tr>
<tr>
<td></td>
<td>ShopName</td>
<td>VarChar(30)</td>
<td>The shop’s name.</td>
</tr>
<tr>
<td></td>
<td>Opening Time</td>
<td>Time</td>
<td>The shop’s opening time.</td>
</tr>
<tr>
<td></td>
<td>Closing Time</td>
<td>Time</td>
<td>The shop’s closing time.</td>
</tr>
<tr>
<td></td>
<td>Lat</td>
<td>Double</td>
<td>The shop’s latitude.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>-------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Lng</td>
<td>Double</td>
<td>The shop’s longitude.</td>
<td></td>
</tr>
<tr>
<td>ImageURL</td>
<td>Text</td>
<td>The shop’s picture.</td>
<td></td>
</tr>
<tr>
<td>ShopOwner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OwnerID</td>
<td>Int</td>
<td>Auto incremented ID for each shop owner in the database.</td>
<td></td>
</tr>
<tr>
<td>FirstName</td>
<td>VarChar(30)</td>
<td>The shop owner’s name.</td>
<td></td>
</tr>
<tr>
<td>Email</td>
<td>VarChar(50)</td>
<td>The shop owner’s email.</td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td>Text</td>
<td>The shop owner’s password.</td>
<td></td>
</tr>
<tr>
<td>ShopID</td>
<td>Int</td>
<td>Foreign key, links owner to his shop.</td>
<td></td>
</tr>
<tr>
<td>OpeningDays</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DayID</td>
<td>Int</td>
<td>Auto incremented ID for each day.</td>
<td></td>
</tr>
<tr>
<td>ShopID</td>
<td>Int</td>
<td>Foreign key, links its shop with its opening days.</td>
<td></td>
</tr>
<tr>
<td>Days</td>
<td>VarChar(3)</td>
<td>Abbreviation of the days.</td>
<td></td>
</tr>
<tr>
<td>Rating</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ShopID</td>
<td>Int</td>
<td>Foreign key, links the shop with the rating table.</td>
<td></td>
</tr>
<tr>
<td>UserID</td>
<td>Int</td>
<td>Foreign key, links the user with the rating table.</td>
<td></td>
</tr>
<tr>
<td>Rating</td>
<td>Double</td>
<td>The value of rating set by the user for a shop</td>
<td></td>
</tr>
<tr>
<td>Item</td>
<td>ItemID</td>
<td>Int</td>
<td>The ID of the item offered.</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>ShopID</td>
<td>Int</td>
<td></td>
<td>Foreign key, links the shop with the items table.</td>
</tr>
<tr>
<td>ItemName</td>
<td>VarChar(10)</td>
<td></td>
<td>The name of the item sold.</td>
</tr>
</tbody>
</table>
2. Sequence Diagrams

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It shows in details the flow of events that happen in each use case.

The following figure shows the sequence diagram of the user login use case.

![Login Sequence Diagram](image_url)

Figure: Login Sequence Diagram
The next diagram shows the flow of event when the shop owner chooses to add a store to the application.

Figure: Add Store Sequence Diagram
3. Class Diagram

![Class Diagram Image]

Figure: Class Diagram
VII. Technologies Enablers

**Presentation Layer**

<table>
<thead>
<tr>
<th><img src="android-studio.png" alt="Android Studio" /></th>
<th>Android Studio, Standard IDE for android development</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="xml.png" alt="XML" /></td>
<td>Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts.</td>
</tr>
</tbody>
</table>

**Domain Layer**

<table>
<thead>
<tr>
<th><img src="java.png" alt="Java" /></th>
<th>The Java programming language is used to implement the business logic and the controller on server side.</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="glassfish.png" alt="GlassFish" /></td>
<td>GlassFish Server delivers a flexible, lightweight, and production-ready Java EE 6 application server</td>
</tr>
</tbody>
</table>
MySQL is used to create and populate the tables needed in the database. It also used to retrieve, store, and update data.

The NetBeans Platform allows applications to be developed from a set of modular software components called modules.

Data Layer

MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system.
Other Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Postman" /></td>
<td>Postman is a powerful HTTP client for testing web services.</td>
</tr>
<tr>
<td><img src="image" alt="Adobe Photoshop" /></td>
<td>Adobe Photoshop is a raster graphics editor developed and published by Adobe Systems for macOS and Windows.</td>
</tr>
<tr>
<td><img src="image" alt="Creately" /></td>
<td>Creately is an online tool that makes it easy for you to create, share, and collaborate with data-rich diagrams.</td>
</tr>
</tbody>
</table>
VIII. Implementation and Development

I implemented the business logic by using POJO classes that are wrapped using session beans. JPA is used to persist data in the database and connects to the Google maps server as it uses Google Maps API, which sends as parameters the location as Latitude and Longitude shop address. The service, on the other hand, returns a JSON file which I parsed in my code to retrieve the data I needed.

For connecting to the Client side, in my Web controller, using Restful Api, I send and receive HTTP requests that are provided as JSON files to retrieve the type of request and what kind of method I use.

I used some APIs such as Google Maps and Google Place Locator by signing in the Google Services Platform. I got an API Key for each service I wanted to import into my application.

After adding the keys in the file called “AndroidManifest” and implemented the dependencies in the Gradle file, I was able to add and modify the different widgets offered by Google Services. I also had to grant some permissions such as the location and the Internet.

```xml
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Figure: Permissions granted in the “AndroidManifest” file.
Also, I imported a library called Volley that handles the HTTP requests. It is an HTTP library that makes networking for Android apps easier and most importantly, faster, according to the Android documentation.

The Volley library offers the following benefits:

- Automatic scheduling of network requests.
- Multiple concurrent network connections.
- Transparent disk and memory response caching with standard HTTP cache coherence.
- Support for request prioritization.
- Cancellation request API. You can cancel a single request, or you can set blocks or scopes of requests to cancel.
- Ease of customization, for example, for retry and backoff.
- Strong ordering that makes it easy to correctly populate your UI with data fetched asynchronously from the network.

- Debugging and tracing tools.

Figure: Volley library in the user login.
IX. Results

The following screenshots present the different activities that my application implements.

Figure: Splash Screen
Figure: Welcome Screen  Figure: Login Screen  Figure: Map Activity
Figure: Register Shop  Figure: Place Selector  Figure: Map View of shop
X. References

- The PHP documentation