First Step

Mobile Application for Donation Management

FINAL REPORT

School of Science and Engineering

Capstone Design

Spring 2017

Done by:

Kaoutar El Khroufi

Supervised by:

Dr. Omar Houssaini Iraqi
Capstone Project Report:

Approved by the supervisor

Dr. Omar Hussain Iraql
# Table of Contents

Acknowledgement: ........................................................................................................4

Abstract: ..........................................................................................................................5

Introduction: ....................................................................................................................6

Feasibility Study: ............................................................................................................7

User & System Requirements: .......................................................................................9

Steeple Analysis: ............................................................................................................13

Technology Enablers: ....................................................................................................14

System Architecture & Design: .....................................................................................16

Implementation: ..............................................................................................................24

Results: ..........................................................................................................................27

References: ......................................................................................................................31
Acknowledgement:

First, I’d like to thank whomever is reading this project for their interest and time. It was a great pleasure going through a hard yet necessary experience.

Most importantly, I’d like to express my deepest gratitude to my supervisor Dr. Omar Houssaini Iraqi for his support and patience throughout the semester, and also for sharing his knowledge and helping me in every stage of the project development.

Also, I had a great amount of support from my friends who were also working on their capstone, especially for their emotional support. We all faced problems and we all managed to get through them by supporting each other.
Abstract:

*First Step* capstone project objective is to handle donations and connect the donators with the nearest/appropriate needy person through a social mobile application. The application handles different services such as the reviews of the searched needy person and displays their contacts and their location.

The application allows users to create a profile for themselves and the information about a certain family that needs help. This profile will be shown to donators who are looking for someone to donate to. Mainly only information is exchanged. The donators are able to post/see reviews about other people’s profiles.

This application is aimed to help the Moroccan society and create a sense of solidarity through modern techniques. The social implications it will have after being launched will certainly aim at closing the gap in terms of quality of life for the less fortunate citizens.
Introduction:

First Step Mobile Application, as its name might suggest, is a social application designed to provide the necessary information for people to take the initiative and support each other. This application allows their users to search based on their location for the nearest people in need according to the type of help they could provide. On the other hand, it also allows other people to share information about their needs and ask for help in a modern and less embarrassing way.

This software will make it easier for anyone to donate anything they want in a simple way and the least troubling way, because it shows the nearest possible people who need help.

Some might ask, if a person is actually in need, they won’t have access to the modern technologies. My answer to that is that anyone can create a profile for these people, by simply provide their location and their contacts. Again, this might not be the case for some families; however, this project is designed to help those who can’t help themselves, and also get people to take the initiative, either donating or by creating profiles for people they know who might need help.

Note: through this report, I’ll be referring to less unfortunate people type of users as LUP
Feasibility Study:

Project General Context:

*First Step* is a project that mainly manages donations. The concept of this project is to make it easy to get information about families who are in need. It would provide a link to the nearest family based on the criteria the user chooses. The idea is to either enter data about a family or to get information. The application will be a mobile application due to the current atmosphere where it is common to use specified apps for most activities.

Global Objectives of the project:

Services that this application will provide are the following:

- There are two types of users
  1. Users who want to provide information about families
     - They will be able to create a profile with the necessary information (contact, location, etc.)
     - Add/check reviews
  2. Users who want to donate
     - They will be able to access profiles based on the criteria they specify
     - Add/check reviews

Client Analysis:

This project targets all citizens. Yet, do they really need this application?

Usually people donate to the people they actually know that they are in need or go directly to an association. However, it is not always the case. When someone moves to a new city, or they actually do not know many people in need, or they would like to help people who are in a harder situation. This new concept will help the unfortunate people to get a high chance of getting donations through a modern system.
**Market Analysis:**

Doing my research, I could not find a similar application already existing. The applications that are available belong to certain organizations and their goal is to collect donations.

My application focuses mainly on helping citizens accessing information and also identifying cases that need help through profiles and reviews.

**Economic feasibility:**

This first part of the project does not generate any financial costs because the tools that will be used to develop this application are all open-source tools. The only cost will be time and human efforts. However, the second part requires many costs. For submitting the app to a certain app store, the submission fees is required. Also, maintenance of the app, managing the raising amount of users and storing data will generate costs.

**Technical feasibility:**

This project is feasible in the technical sense. All tools necessary are available and only requires a fair amount of time to acquire a good knowledge about how to use them.

**Conclusion:**

The concept of this project was actually a personal project that is not aimed to generate profit. However, the profit that is most important in a capstone project is the learning experience and the outcome of developing such an application. The scope of my application targets two types of users while generating the appropriate information based on the criteria specified by the users. This projects has social implications in term of creating a sense of solidarity along with closing the gap for the less fortunate people in terms of quality of life.
User & System Requirements:

Functional Requirements:

User account management:

1. User registration:
   - The application shall allow users to register and create a profile
2. User authentication
   - When registering, the application shall allow users to authenticate using their user name and password.
3. User Log in
   - Users shall be able deactivate their account
   - Users shall be able log out

Profile Management:

The profiles of users shall contain the following:

- First name
- Last name
- Phone number
- Location
- Whether they are a Donator or an LUP

2. User profile modification:

- Users shall be able to edit/delete data on their profiles.

Donations Management:

LUP profile creation:

- The application shall enable users registered under a LUP profile to add the following information about themselves or about the ones concerned:
  - The state of the family.
  - General information about the help needed.
  - Location.
- Phone number.
  - The system shall enable donators to evaluate the profiles using reviews and comments.

LUP profiles modification:
  - Users under this category shall be able to change information about their profiles.

LUP profiles deletion:
  - LUP profiles’ owners shall be able to deactivate their accounts.

LUP profiles display:
  - Users of the application shall be able to search LUP profiles based on their category and sub-category and it will be sorted based on their location.

**Donators Management:**

Donation request creation:
  - Users shall be able to browse LUP profiles using:
    - Donation category
    - Location.
  - Notifications shall be sent to appropriate LUP users when there is a possible donor
    - Notifications shall be sent to LUP chosen by the users.
    - Search results for LUP shall enable location filters and reviews/comments filters.
Use Case Diagram:

Figure 1: Use Case Diagram
Non-Functional Requirements:

Performance:

- The application performance should be optimised, and response time should be minimized.

Scalability:

- The application should be highly scalable; since it is meant to be used by people around the world.

Extensibility:

- The application should be extensible to allow adding other services in the future, such as using profiles from Facebook. Also, allowing integration with other APIs such as Google maps and OpenID.

Integration:

- The system should have the ability to extend its requirements. For example, having the ability to extend with the minimum development and deployment costs.

Security:

- The system should be highly secure since only authenticated users can have access to the server. It should respect the 3 aspects:
  - Confidentiality:
    - The user shall ensure who gets to see their information
  - Integrity:
    - Editing data shall be authorized only to owners of the accounts
  - Availability:
    - The application shall be available at all times.

Maintainability:

- The system should be easily maintainable to allow for additional upgrades that can be implemented in the future.
Steeple Analysis:

Social:
This application is designed to be a social app. Its purpose is to enable a sense of solidarity in the community. The app has a societal impact since its aim is to close the gap between the less fortunate people and the more unfortunate in terms of quality of life.

Technology:
Nowadays people are more likely to use their mobiles to search for their daily needs. This app is designed to leverage this dependency on technologies to provide a service.

Economic:
This app has no economic implications.

Environmental:
This application does not harm the environment nor helps solving environmental issues.

Political:
The application does not target any governmental issues nor political. The donations are done physically and only information is transmitted through this app. Also, it respects the rules of academic university.

Legal:
The application will be fully legal, it will respect copyright and use open source softwares.

Ethical:
Concerning ethical issues, this application will only post information that are published by the users and any personal information will not be disclosed.
## Technology Enablers:

### Server Side:

<table>
<thead>
<tr>
<th>Technology</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Java</strong> programming language</td>
<td>The <strong>Java programming language</strong> is used to implement the business logic and the controller on server side.</td>
</tr>
<tr>
<td><strong>GlassFish</strong></td>
<td>According to Oracle: “<strong>GlassFish Server</strong> delivers a flexible, lightweight, and production-ready Java EE 6 application server”</td>
</tr>
<tr>
<td><strong>MySQL</strong></td>
<td><strong>MySQL</strong> is used to populate the tables needed in the database.</td>
</tr>
<tr>
<td><strong>MySQL Workbench</strong></td>
<td>IDE used to implement the database.</td>
</tr>
<tr>
<td><strong>NetBeans</strong> IDE</td>
<td><strong>NetBeans</strong> IDE is used to implement the business logic and the controller</td>
</tr>
</tbody>
</table>

**Table1:** Server-Side technology enablers
Client Side:

| Android Studio | Android Studio, Standard IDE for android development |

Table 2: Client-Side technology enablers

Other Tools:

<table>
<thead>
<tr>
<th>Ngrok</th>
<th>Ngrok was used to test the request and responses for my server side</th>
</tr>
</thead>
<tbody>
<tr>
<td>Postman</td>
<td>Postman is available as Chrome extension and is used to manage REST calls.</td>
</tr>
<tr>
<td>Adobe Photoshop</td>
<td>Adobe Photoshop was used for the design of the client side images and interfaces</td>
</tr>
<tr>
<td>Draw.io</td>
<td>Draw.io was used to design the architecture and diagrams I used in this report</td>
</tr>
</tbody>
</table>

Table 3: Other tools used
System Architecture & Design:

The System architecture follows the MVC model which enables the concept of separation of concerns. It has a standard approach for the 3 tiers architecture: Client tier, Server Tier, and Data Tier.

The Server Tier contains the business logic, which also make use of a web service: Google Maps Web Services by using Google Maps Distance Matrix API. It is exposed to the client side using RESTful web service. And using ORM, data that is stored to Mysql database is converted into classes, and it is accessed/persisted using JPA.

The Data tier stores in Mysql database all data I am using in the application.

For the client Side, it is implemented using Android Studio. It had the necessary requirements to establish a connection with the server side using HTTP requests. Also, following the requirements, it is responsible for getting the location of the device either using GPS or internet.

Figure 2: System Architecture
Entity Relation Diagram:

Figure3: ERD
**Entities and their attributes description:**

This following table is a description for the database tables and their attributes along with the data type of each attribute.

<table>
<thead>
<tr>
<th>Table</th>
<th>Attribute</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Table</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>idUser</td>
<td>Integer</td>
<td>Auto incremented id for each user that is added to the table in the database</td>
</tr>
<tr>
<td></td>
<td>userName</td>
<td>String</td>
<td>The user name of the user, it is an email address</td>
</tr>
<tr>
<td></td>
<td>password</td>
<td>String</td>
<td>The password that is set by the user to be able to access his/her profile.</td>
</tr>
<tr>
<td></td>
<td>firstName</td>
<td>String</td>
<td>First name of the user</td>
</tr>
<tr>
<td></td>
<td>lastName</td>
<td>String</td>
<td>Last name of the user</td>
</tr>
<tr>
<td></td>
<td>phoneNumber</td>
<td>String</td>
<td>Phone number of the user</td>
</tr>
<tr>
<td></td>
<td>latitude</td>
<td>Double</td>
<td>The latitude of the user’s location and it is generated by in the client side using GPS on their device</td>
</tr>
<tr>
<td></td>
<td>longitude</td>
<td>Double</td>
<td>The longitude of the user’s location, and is also generated in the client side using GPS on the user’s device</td>
</tr>
<tr>
<td>Table 4: Entities and their attributes’ description</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Donator Table</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>donatorId</td>
<td>Integer</td>
<td>The donator id (primary key) and it is auto generated/incremented</td>
<td></td>
</tr>
<tr>
<td>Score</td>
<td>Integer</td>
<td>The score of the donator that is edited each time they make a donation</td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>Integer</td>
<td>The user id (foreign key) that links the user to its type.</td>
<td></td>
</tr>
<tr>
<td><strong>LUP Table</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lupId</td>
<td>Integer</td>
<td>The lup id (primary key)</td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>Integer</td>
<td>The user id (foreign key) that links the user to this type.</td>
<td></td>
</tr>
<tr>
<td><strong>Review Table</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>idReview</td>
<td>Integer</td>
<td>The id of the review (primary key)</td>
<td></td>
</tr>
<tr>
<td>lupId</td>
<td>Integer</td>
<td>The id of the lup (foreign key)</td>
<td></td>
</tr>
<tr>
<td>donatorId</td>
<td>Integer</td>
<td>The id of the donator (foreign key)</td>
<td></td>
</tr>
<tr>
<td>Review</td>
<td>String</td>
<td>The review of the lup that is put by the donator</td>
<td></td>
</tr>
</tbody>
</table>
Description of the table relations:

<table>
<thead>
<tr>
<th>Tables</th>
<th>Relationship type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User and Donator</td>
<td>One-to-one</td>
<td>Mainly the User table is the parent where additional data about the donator is stored.</td>
</tr>
<tr>
<td>User and LUP</td>
<td>One-to-one</td>
<td>A user can only be of one type and is the parent table of the LUP.</td>
</tr>
<tr>
<td>Donator and Review</td>
<td>One-to-many</td>
<td>A donator can have many reviews, but a review can be written by one Donator</td>
</tr>
<tr>
<td>LUP and Review</td>
<td>One-to-many</td>
<td>A lup user can have many reviews but a review can be for only one lup.</td>
</tr>
</tbody>
</table>

*Table 5: Entities and their type of relationships*
Class Diagram:

FirstStepController

- CreateUser(String username, String password, String name, String iname, String phone, double latitude, double longitude)
- Authenticate(String username, String password)
- getDistance(double source_longitude, double source_latitude, double destination_longitude, double destination_latitude)
- getProfile(String username)
- search(String category, double latitude, double longitude)
- addReview(String lup_username, String donator_username, String review)
- updateRating(String username)
- editProfile(String email, String field, String value)
- changePassword(String username, String oldpass, String newpass)

FirstStepBean

- CreateUser(String username, String password, String name, String iname, String phone, double latitude, double longitude)
- Authenticate(String username, String password)
- getDistance(double source_longitude, double source_latitude, double destination_longitude, double destination_latitude)
- getProfile(String username)
- search(String category, double latitude, double longitude)
- addReview(String lup_username, String donator_username, String review)
- updateRating(String username)
- editProfile(String email, String field, String value)
- changePassword(String username, String oldpass, String newpass)

FirstStepImplementation

- CreateUser(String username, String password, String name, String iname, String phone, double latitude, double longitude)
- Authenticate(String username, String password)
- getDistance(double source_longitude, double source_latitude, double destination_longitude, double destination_latitude)
- getProfile(String username)
- search(String category, double latitude, double longitude)
- addReview(String lup_username, String donator_username, String review)
- updateRating(String username)
- editProfile(String email, String field, String value)
- changePassword(String username, String oldpass, String newpass)

Figure 4: Class Diagram
Application Activity Diagram:

Figure 5: Activity Diagram
The following table provides the description of each activity in the diagram in Figure 5.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>The user can register and save his/her information into the system</td>
</tr>
<tr>
<td>Login</td>
<td>The user can login using his/her credentials</td>
</tr>
<tr>
<td>Type of user</td>
<td>Upon registering, the user specifies which type of user they are</td>
</tr>
<tr>
<td>Manage profiles</td>
<td>The user can edit/disable their accounts</td>
</tr>
<tr>
<td>Browse Categories</td>
<td>The donators can browse the categories of lup they want to donate to.</td>
</tr>
<tr>
<td>Search Profiles</td>
<td>The donators can search profiles based on the nearest</td>
</tr>
<tr>
<td>Call LUP</td>
<td>The user can call the phone number of the lup</td>
</tr>
<tr>
<td>Send Notification</td>
<td>The donator can send notification for possible donation</td>
</tr>
<tr>
<td>Add Reviews</td>
<td>The donators can add reviews concerning lups</td>
</tr>
<tr>
<td>Confirm Donation</td>
<td>The lup can confirm receiving donations</td>
</tr>
<tr>
<td>Check Notification</td>
<td>The lup can check notifications for possible donations</td>
</tr>
<tr>
<td>Log out</td>
<td>The user can log out</td>
</tr>
</tbody>
</table>

Table 6: Mobile app Activities and description
Implementation:

Database Implementation:

Database was implemented using Mysql Workbench following the necessary logic and requirements I designed the tables needed to store data.

The following is the tables as they exist in Mysql Workbench

**Figure 6:** User Table  
**Figure 7:** Donator Table  
**Figure 8:** LUP Table  
**Figure 9:** Review Table

Server Side Implementation:

The business logic was implemented in a POJO class and wrapped using session beans. It uses JPA to persist data from database and connects to the Google maps server as it uses Google Maps Distance Matrix API, which sends as parameters the location and destination as Latitude and Longitude address, and the service returns a JSON file which I parse in my code to retrieve the distance and the time needed (I used time needed if using a vehicle – other ways are available).
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.

As for the Database connection, I use Java Persistence API to persist data.

The following is the screenshot of the classes in business logic in different folders, classes, and libraries of the Server Side.

![Figure 10: First Step app EJB](image)

Figure 10: First Step app EJB
Client Side Implementation:

The client side implementation was done using Android Studio, it is java based and uses XML for the layouts.

The following are the activities I used:

![Image of Android Studio Java classes]

**Figure 11:** Android Studio Java classes

Some of the activities I used were provided by Android Studio with basic layouts that I managed to edit and set it the way I wanted, and personalized the design.

![Image of Android Studio Layouts]

**Figure 12:** Android Studio Layouts
Results:

The following screenshots are from the working application using my personal android phone. The first page of the application:

Figure 13: First Page
Figure 14: Sign Up Process
Figure 15: Process for searching a profile
References:


Websites:

http://docs.oracle.com/javae/6/tutorial/doc/bnbpz.html
https://developers.google.com/maps/documentation/distance-matrix/intro#Audience
https://apexapps.oracle.com
https://netbeans.org
https://Stackoverflow.com
https://Youtube.com
https://Tutorialspoint.com