Lost-and-Found Android Application

User Requirements Document

version 2.0

Develop Team: MobileAppDirectory
Members: Baiyang Wang, Peng Tong, Ruoxiao Wang
Date: 05/01/2015

User Requirements Document

Table of Contents

Introduction

The objective of this User Requirements Document (URD) is to obtain agreement with all technical advisers & developers in MobileAppDirectory team regarding the qualitative and quantitative characteristics of a proposed system. This URD is intended to unambiguously communicate our understanding and expectations for the "Lost-and-Found" (LaF) Android Application to technical advisers, as well as among ourselves. To this end, this URD avoids technical implementation language, and restricts notations used to those that express functionality from users' viewpoints.

Document Scope

This URD provides a comprehensive description of all intended functions of the *"Lost-and-Found" Android Application*. Also, it outlines the sequence of events for each user experience, including when and how that user would utilize other systems and interact with the backend database, as well as other users. Requirements for the operational and development environments also are described.

Intended Audience

Technical Advisors:

Professor: Bob Singh

Instructor: Mary Catherine Bishop

Teaching Assistant: Pushkar Joglekar

Users:

All potential users who works or studies in NASA campus, Moffet Blvd, Mountain View, CA, 94043

Developers:

MobileAppDirectory Team

System Overview

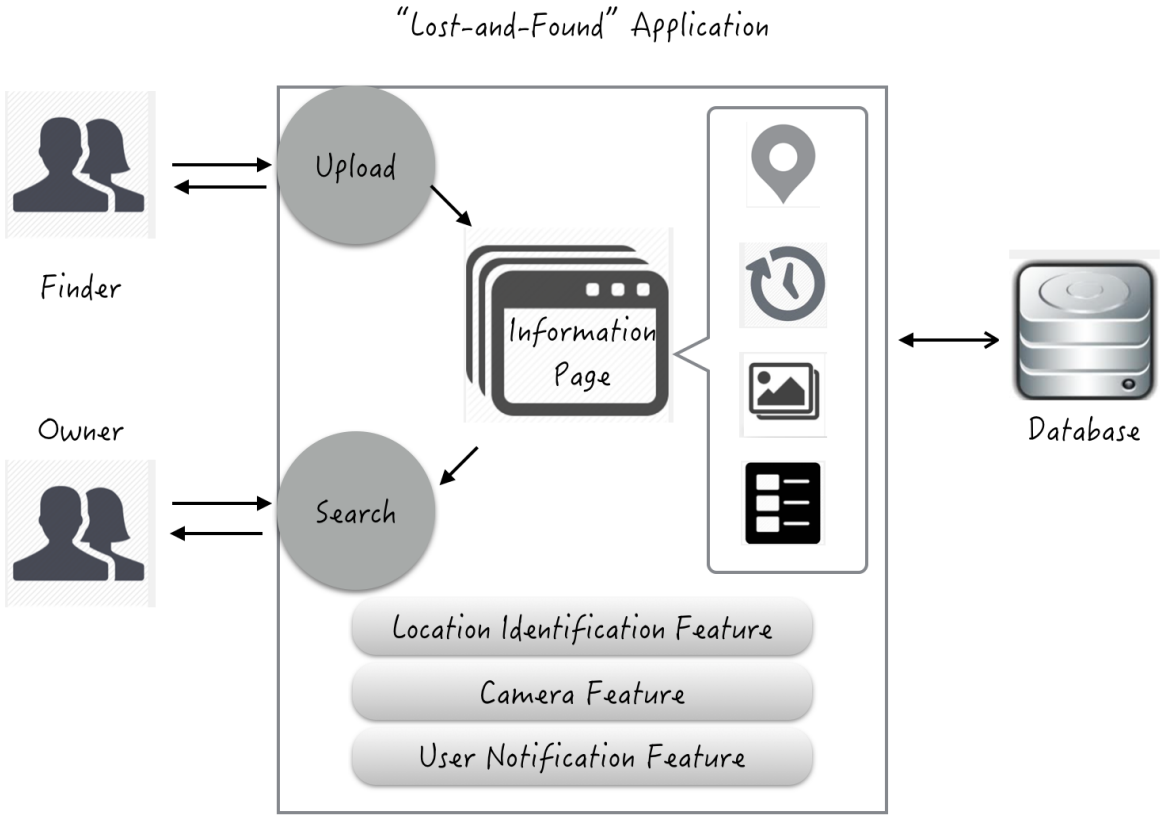


Figure 1 An overview of the LaF system

Purpose

The Android application "Lost-and-Found" system provides a platforms for both users who lose properties searching the finder and users who find properties searching the owner. If a match is confirmed by the system, a link between the owner and the finder will be created by the system after a mutual identification authentication is passed.

Scope

Currently, "Lost-and-Found" system will serve every potential user who works or studies in NASA campus. This system allows user to create an account, modify profiles, and create an event either as a "property finder" or "property loser".

The module allows user to upload information into the backend database with the format of text, location, time and graph. It can maintain the lists of properties categorized in different tags. Not all information about a property will be available to first-stage user, access constraint is applied in case of false claiming and an identification and authentication process is mandatory when necessary.

The system serves as a man-in-middle between a finder and a possible owner. It handles sending any corresponding notifications to both sides before the mutual identification and authentication being passes and providing the finder's contact information to the owner.

Also, It is assumed the module supports concurrent user access and requests from the users are synchronized.

General Requirements

In this project, LaF requires an local instance serving as a database to store all backend data. A Servlet would be created and maintained stably and efficiently to guarantee multiple users logging-in, uploading data and searching information. Also, the system should support information interacting between users via sending notifications to and screening feedbacks from different users.

User Characteristics

Everyone may be the user of this system with different education and cultural background. We assume all users are non-technical individuals and beginner to user intermediate system. Also, we assume all users are busy workers or students who have limited time and hate tedious and complex UI design.

Constraints

A. Consistent and secure login screens

B. Secure data storage and information access control

C. Neat, simple and user-friendly user interface

D. Precise and simple instructions

E. Ease of use to system menus, pull-down lists, photo-taking and location-sharing buttons

F. Instant feedback responding and notification sending

System-Wide Requirements

Actors

The LaF system recognizes 2 types of users.

The first type is a property finder, who is willing to search the owner and return it back. The finder can create a new item for the property and then upload basic information. If the property is valuable, a finder-driven identification authentication process is required between the finder and the owner through the system.

The second type is a property owner, who is searching for his or her belongings. The owner can use the smart search engine to check whether there is a match with his or her lost belongings in the database. If the match is a valuable property, the system will launch a finder-driven identification authentication process, only through passing which the owner can gain more information on the property and finally access the finder.

Events

LaF is a tool serves as an intermediary to assist to build a secure bridge between the property finder and a possible owner. The most critical events are: 1) Fetching location and time 2) Taking photos 3) Categorizing tags 4) Storing, updating and retrieving information 5) Providing smart searching 6) Controlling information access 7) Sending notifications 8) Managing memory and deleting events

Detailed Requirements

Functional Requirements

The system allows the finder to choose (if it has already been created) or create a tag(the category of the property) and create a new item using the location and time of the event. Also, he or she will be asked to upload 1-2 photos of the property. Finally, if the property is valuable- that means it is in the predefined category of 'valuable property' by the system, the finder will be asked to design 1-3 questions for identification authentication. When some answers are sent back from a possible owner, the finder can check whether it is the right person and inform the system if his or her contact information can be shared or not.

The module allows the owner to choose a tag and see all available found properties in the format of the location and time it is found. When one match is detected, the owner can request to contact the finder. If it is a valuable property, he or she has to answer the questions designed by the finder for identification authentication. Then the owner will receive a notification from the system informing the result after the finder checks the answers. If passes, the owner will receive the photos of the property and then be asked to confirm whether it is his or her belongings. If yes, the finder's contact information will be shared.

To maintain the efficiency of backend database, the system should have the capability of memory self-management. First, an item and all corresponding data should be deleted when an event is completed, that is - the finder's contact information has been notified to the owner. Then, all information of this event will only stored in the owner's profile (as a 'history event') in the format of text. Second, as for the unmatched items, they should be cleaned by the system in a certain period of time, 2 months for valuable property and 1 month for common property, when permission is obtained from the finder.

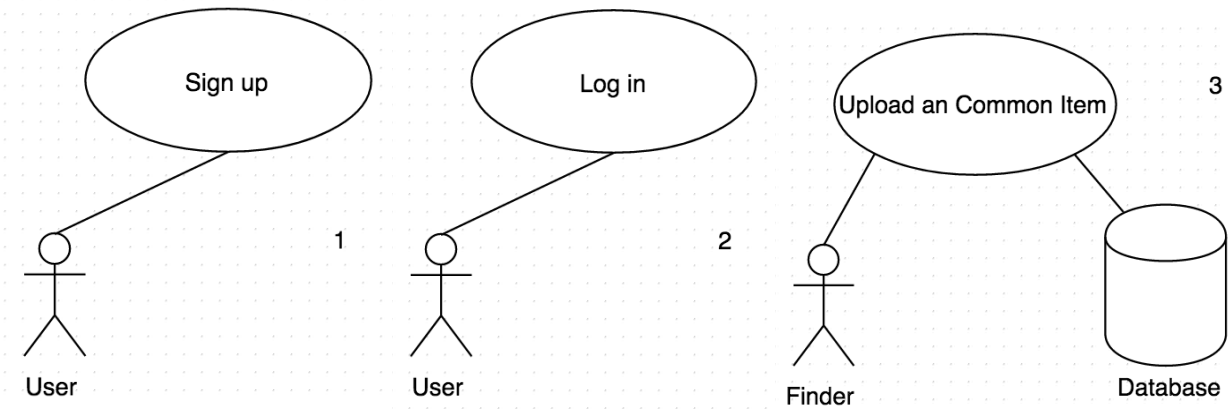
User Requirements Model

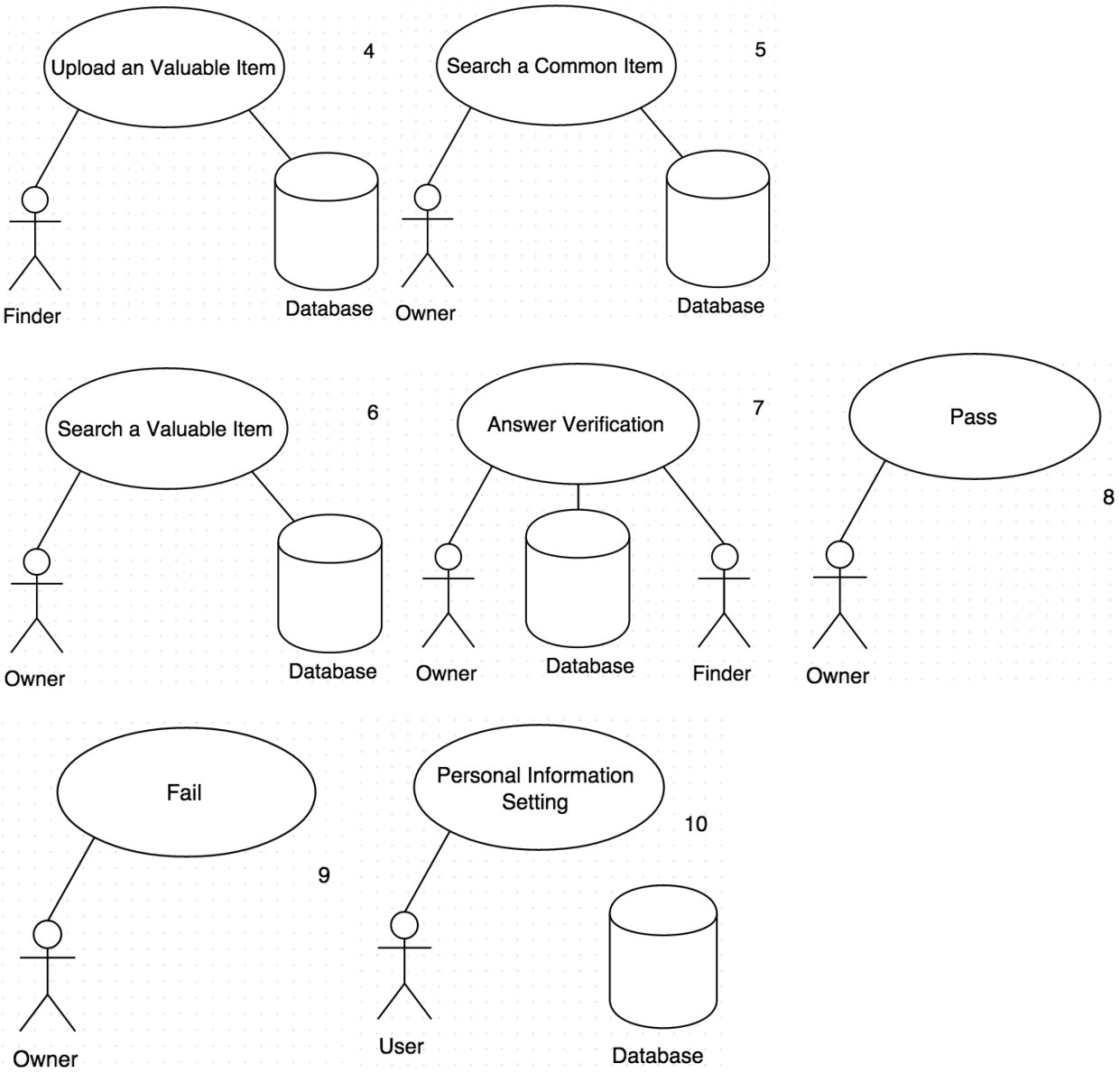
Use Case Overview

The table below offers a list of use cases. Detailed descriptions of the use cases are given in the Specifications section following this table.

Table 1 List of use cases

No.	Use Case ID	Use Case Name	Priority
1	SIGN-UP	Sign Up To The LaF App	High
2	LOG-IN	Log In To The LaF App	High
3	UPL-COM-ITM	Upload A Common Lost Item To The Database	High
4	UPL-VAL-ITM	Upload A Valuable Lost Item To The Database	Medium
5.	SER-COM-ITM	Search A Common Lost Item	High
6.	SER-VAL-ITM	Search A Valuable Lost Item	Medium
7.	VEF-ANS	Answers Verification	Medium
8.	PAS-VEF	Owner Pass The Verification	Medium
9.	FAL-VEF	The Verification Fail	Low
10.	USR-SET	User Personal Information Setting	Low





Detailed Use Case Specifications

Table 2 Use Case Details – SIGN-UP: Sign Up To The LaF App

Use Case ID: SIGN-UP	**Use Case Name:** Sign Up To The LaF App
Primary Actor(s):	All users of LoF App
Secondary Actor(s):	N/A
Description:	Entering username, password, contact information to register to the

	App.
Preconditions:	User choose “Sign up” and Sign up page is available and waiting for user input.
Normal Flow of Events:	1. User enters username, password, contact information and re-enter password in corresponding text boxes. 2. User click “submit”.
Postconditions:	After step 2, the main user page appears.
Frequency of Use:	High
Alternative Flows:	User can click “back” within the Sign Up page to cancel the sign up.
Exceptions:	User not entering data in any text box can not enter the main user page.
Assumptions:	LoF is up and running.
Issues:	Database has the same username.
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 3 Use Case Details – LOG-IN: Log In To The LaF App

Use Case ID: LOG-IN	**Use Case Name:** Log In To The LaF App
Primary Actor(s):	All users of LoF App
Secondary Actor(s):	N/A
Description:	Requirements username and password and authenticates user into LoF App.
Preconditions:	User choose “Log in” and Log in page is available and waiting for user input.
Normal Flow of Events:	1. User enters username, password in corresponding text boxes.

	2. User click “submit”.
Postconditions:	After step 2, the main user page appears.
Frequency of Use:	High
Alternative Flows:	User can click “back” within the Sign Up page to cancel the sign up.
Exceptions:	User not entering data in any text box or entering wrong information can not enter the main user page.
Assumptions:	Lof is up and running, user has already registered the LoF App.
Issues:	User never has a username and password
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 4 Use Case Details – UPL-COM-ITM: Upload A Common Lost Item To The Database

Use Case ID: UPL-COM-ITM	**Use Case Name:** Upload A Common Lost Item To The Database
Primary Actor(s):	All common lost item finders.
Secondary Actor(s):	N/A
Description:	User who find the common lost item can use this function to upload the information about this item.
Preconditions:	User chooses “I Find” in the main user page and Upload page is available and waiting for user input.
Normal Flow of Events:	1. User clicks “I Find” button in the main user page. 2. User selects category name the item belongs to and select “Common” and click “Next” button. 3. User sets the location and time of finding the lost item and click “Next” button. 4. User take two photos of the lost item and click “Submit” button.

Postconditions:	After step 4, the data will be sent and stored in the database and main user page appears.
Frequency of Use:	High
Alternative Flows:	User can click “back” within the upload process to go back to the front page.
Exceptions:	User not entering data in any text box or entering wrong information can not enter the next upload page.
Assumptions:	Lof is up and running, user has not uploaded this item before.
Issues:	User selects the wrong category.
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 5 Use Case Details – UPL-VAL-ITM: Upload A Valuable Lost Item To The Database

Use Case ID: UPL-VAL-ITM	**Use Case Name:** Upload A Valuable Lost Item To The Database
Primary Actor(s):	All valuable lost item finders.
Secondary Actor(s):	N/A
Description:	User who find the valuable lost item can use this function to upload the information about this item.
Preconditions:	User chooses “I Find” in the main user page and Upload page is available and waiting for user input.
Normal Flow of Events:	1. User clicks “I Find” button in the main user page. 2. User selects category name the item belongs to and select “Valuable” and click “Next” button. 3. User sets the location and time of finding the lost item and click “Next” button. 4. User take two photos of the lost item and click “Next” button.

	5. User designs three questions about this item and click “Submit” button.
Postconditions:	After step 4, the data will be sent and stored in the database and main user page appears.
Frequency of Use:	Medium
Alternative Flows:	User can click “back” within the upload process to go back to the front page.
Exceptions:	User not entering data in any text box or entering wrong information can not enter the next upload page.
Assumptions:	Lof is up and running, user has not uploaded this item before.
Issues:	User selects the wrong category.
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 6 Use Case Details – SER-COM-ITM: Search A Common Lost Item

Use Case ID: SER-COM-ITM	**Use Case Name:** Search A Common Lost Item
Primary Actor(s):	Who lost common items.
Secondary Actor(s):	N/A
Description:	User who lost common item can use this function to find the information about this item.
Preconditions:	User chooses “I Lost” in the main user page and Search page is available and waiting for user search.
Normal Flow of Events:	1. User clicks “I Lost” button in the main user page. 2. User enter category name the item belongs to in the search text box and click “Search” button. 3. A list of lost items’ location and time will appear. Selecting a location and time.

Postconditions:	After step 3, the finder contact information of this lost item will appear.
Frequency of Use:	High
Alternative Flows:	User can click “back” in the Search Page to go back to the main user page.
Exceptions:	User not entering data in search text box or the entered category do not have lost items now.
Assumptions:	The database has this category.
Issues:	User enters the wrong category.
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 7 Use Case Details – SER-VAL-ITM: Search A Valuable Lost Item

Use Case ID: SER-VAL-ITM	**Use Case Name:** Search A Valuable Lost Item
Primary Actor(s):	Who lost valuable items.
Secondary Actor(s):	N/A
Description:	User who lost valuable item can use this function to find the information about this item.
Preconditions:	User chooses “I Lost” in the main user page and Search page is available and waiting for user search.
Normal Flow of Events:	1. User clicks “I Lost” button in the main user page. 2. User enter category name the item belongs to in the search text box and click “Search” button. 3. A list of lost items’ location and time will appear. Selecting a location and time. 4. Three questions will appear in the screen. Answering them and press “Submit”.

Postconditions:	After step 4, the answers will be sent and stored in the database. And finder can check these answers.
Frequency of Use:	Medium
Alternative Flows:	User can click “back” in the Search Page to go back to the front page.
Exceptions:	User not entering data in search or answer text box, or the entered category do not have lost items now.
Assumptions:	The database has this category.
Issues:	User enters the wrong category or answers.
Source:	LoF UI design
Includes:	N/A
Associated Requirements:	N/A

Table 8 Use Case Details – VEF-ANS: Answers Verification

Use Case ID: VEF-ANS	**Use Case Name:** Answers Verification
Primary Actor(s):	Who find this lost item.
Secondary Actor(s):	Who answer these answers.
Description:	User who find this lost item will check the other user’s answers. Then the user who answers these question will receive the result.
Preconditions:	User check the information center.
Normal Flow of Events:	1. User check the information center. 2. User check the other user’s answers. If all answers are right, pressing “Pass”. If answer is wrong, click “Fail” button. 3. User who answer these questions will receive the result.
Postconditions:	After step 3, the user who answer these questions will do the next step based on the result.
Frequency of Use:	Medium

Alternative Flows:	N/A
Exceptions:	User device is not connect to the server.
Assumptions:	N/A
Issues:	User do wrong operations.
Source:	LoF UI design.
Includes:	N/A
Associated Requirements:	N/A

Table 9 Use Case Details – PAS-VEF: Owner Pass the Verification

Use Case ID: PAS-VEF	**Use Case Name:** Owner Pass the Verification
Primary Actor(s):	Who answer these answers.
Secondary Actor(s):	N/A
Description:	If owner pass the verification, he/she will receive the notification.
Preconditions:	User check the information center.
Normal Flow of Events:	1. User check the information center. 2. Photos will appear in the screen to let the user do the final check. 3. If user press “Yes, it is mine!”, the contact information of the finder will appear in the screen.
Postconditions:	After step 3, the user can call the finder.
Frequency of Use:	Medium
Alternative Flows:	If the item in the photos are not the lost item, the user can press “Sorry, it is not mine”.
Exceptions:	User device is not connect to the server.
Assumptions:	N/A
Issues:	User do wrong operations.

Source:	LoF UI design.
Includes:	N/A
Associated Requirements:	N/A

Table 10 Use Case Details – FAL-VEF: The Verification Fail

Use Case ID: FAL-VEF	**Use Case Name:** The Verification Fail
Primary Actor(s):	Who answer these answers.
Secondary Actor(s):	N/A
Description:	If the verification failed, he/she will receive the notification.
Preconditions:	User check the information center.
Normal Flow of Events:	1. User check the information center. 2. Fail page will appear.
Postconditions:	N/A
Frequency of Use:	Low
Alternative Flows:	N/A
Exceptions:	User device is not connect to the server.
Assumptions:	N/A
Issues:	User do wrong operations.
Source:	LoF UI design.
Includes:	N/A
Associated Requirements:	N/A

Table 11 Use Case Details – USR-SET: User Personal Information Setting

Use Case ID: USR-SET	**Use Case Name:** User Personal Information

	Setting
Primary Actor(s):	All registered users.
Secondary Actor(s):	N/A
Description:	Users can change the information in the “MyProfile”.
Preconditions:	User presses the “MyProfile” in the user main page.
Normal Flow of Events:	1. User presses the “MyProfile”. 2. User changes the information. 3. User presses the “Submit” button.
Postconditions:	New data will be sent and stored in the database.
Frequency of Use:	Low
Alternative Flows:	User can click “back” to go back to the main user page.
Exceptions:	User device is not connect to the server. User not entering data in any text box.
Assumptions:	N/A
Issues:	User do wrong operations.
Source:	LoF UI design.
Includes:	N/A
Associated Requirements:	N/A

Technical Requirements

The purpose of this section is to obtain agreement regarding the platforms to be used for deploying the working systems and for developing the working system.

Operational Environment

The Laf system backend database will be built on a MacBook Pro (Retina, 13-inch) with 8 GB memory, 2.4 GHz Inter Core i5 processor and OS X Yosemite v10.10.1. The Android application can be utilized on any Android phone and tablet device with aversion not earlier than Android 2.3. Communication between database and mobile devices is done bluetooth and wifi.

Development Environment

Developers will use the platform of SDK API 21: Android 5.0 (Lollipop). Developers should have access to GUI development tools, Google map API as well as some Android Hardware Features, like camera, GPS, bluetooth, etc.Developers may use Android Studio as an develop IDE and an Android tablet for simulating and testing.