Car Sharing

Project Functional Specification

version 1.0

15.04.2015

Prepared by: Martin Brehhov

Project Supervisor: Joseph Kehoe

TABLE OF CONTENTS

Contents

Introduction 3

Purpose 3

Description 3

Target Readers 3

Scenario 4

Scope 4

Simple architecture overview 5

Core Functionalities of the System 5

1. Login and Registration 5
2. Show Map 6
3. Posting messages 6
4. User Types 6
5. Profile 6
6. Notifications 6
7. GPS coordinates 6
8. Sessions 6

Brief Use Cases 7

1. ShowMap 7
2. Register 7
3. Login 7
4. LiftOffer 7
5. LiftWanted 8
6. ContactUser 8
7. StartProfile 8
8. Exit 8

User Interface 9

Suggested Screenflow overview 9

Screen by screen 10

Home activity 10

“See the Map” 11
Login 12
Register 12
Profile 13
Lift Offered 14
MapClosest 14
Contact Users 14
Lift Wanted 15
Non functional Requirements 15

Introduction

Purpose

This document states the functionality of the App from user perspective. Specs won't go into details of implementation (ex. algorithms, data structures). Document will cover briefly likely hardware solution for given App.

Description

Car Sharing App is a 4th year main project intent to create “human-friendly” application for people who want to get lift (hitchhikers) or for people who are willing to give a lift (drivers) . Also project is aiming meet new people.

Target Readers

Document main target group are software developers and could contain IT related terminology.

Scenario

Some informal scenario :

Hitchhiker is standing on the road aiming to travel from point A to point B. He starts the App , specifies his location and destination. Waits until someone will contact with him.

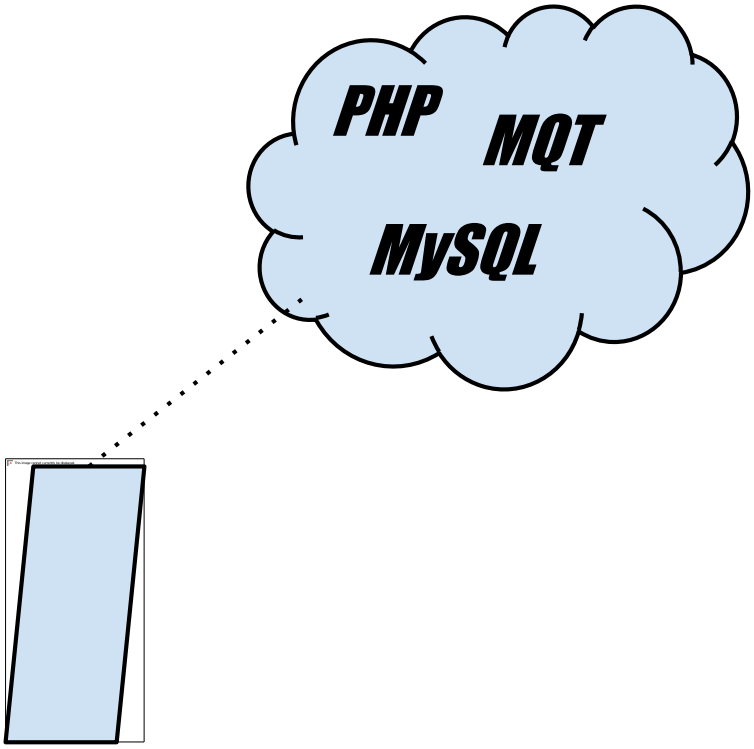
Driver is system-user whose typical scenario might look as following: Person starts App, specifies his type as Driver and observe all Hitchhikers around 30km from his GPS coordinates. Then he selects appropriate user and contact with him.

NB: user of the App can specify their type during runtime changing from Hitchhiker to Driver and vise versa.

Scope

As this idea suggest we are dealing with client side application or Android device and server side services which is crucial part to store data or retrieve data.
The Document scope would be description of client and server side functionalities.

Simple architecture overview



Android is client device. Client device is in constant connection with Mosquitto (MQTT) server, which is implemented for notify clients with events. Additionally the “cloud” side uses PHP for data flow between clients and MySQL database.
More details in research manual and Design document.

Core Functionalities of the System

Before describing use cases its good to set some critical or core functionalities of the system.

1. **Login and Registration**
    - **a.** Allows user to login into “Car Sharing” App
    - **b.** Allow user to register an account and become a member

2. Show Map

- **a.** System should be able to use functionality of the google map API and populate the map with data from 3th party web server databases.
- **b.** Provide a way for members and non-members with function to observe map and users on map.

3. Posting messages

- **a.** Store received or sent message objects within the system .
- **b.** Deprecate all the data when User logs out.
- **c.** Allow actively logged in member to send messages to each other.

4. User Types

- **a.** System should be able to deal with 2 type of users
- **b.** Driver user
- **c.** Hitchhiker user

5. Profile

- **a.** Automatically directed logged in users. Profile should provide basic functions for Driver user and Hitchhiker user.

6. Notifications

- **a.** When User send message to someone, message has to appear on recipients device over network.
- **b.** Enable messages to come asynchronously in background.

7. GPS coordinates

- **a.** Hitchhiker user should be able to Store his current location and his destination gps coordinates.
- **b.** driver user should be able to see hitchhikers that are close to his own gps coordinates.

8. Sessions

- **a.** Logged in users should be able to navigate through different activities.

Those are App major features.When member (Android side) is logged in he should be in constant connection with Server. That allows communication via messages. Note that messages are not stored. There are no real need to store message objects, so if messages are sent to non connected person - message will be lost.

Brief Use Cases

Taking an account of system functionalities following uses cases main scenarios are proposed. More detail implementation of the use cases are describe in Design document with alternative paths.

1. **ShowMap**
    - **a.** Use case begins when anonymous user want to see the map.
    - **b.** User clicks the “Show the map”
    - **c.** System returns google map to user. with all Hitchhikers (names hidden) on it.

2. **Register**
    - **a.** Use case begins when anonymous user intents to become a member of the “Car Sharing” system. Click the register button from main screen.
    - **b.** Inputs name, password and email which will be verified upon click at server side.

3. **Login**
    - **a.** Login use case starts when member clicks on Login button from main menu.
    - **b.** User input details of username and password which upon click of another button verifies that data with cloud.

4. **LiftOffer**
    - **a.** This use case starts when system member as Driver user decide to click “LiftOffer” button from Profile.
    - **b.** This use case wraps several activities with it - Like map activity and contact users activity.
    - **c.** Drivers GPS coordinates are determined and closest Hitchhikers brought to him in google map activity.

d. Driver selects user and starts ContactUser Use case.

5. LiftWanted

a. This use case starts when system member as Hitchhiker user decide to click “LiftWanted” button within Profile.

b. User should be able to save his GPS location and determine his destination point on google map.

c. Data is saved and in cloud.

6. ContactUser

a. This use case allows users to check their messages.

b. use case can starts when User click on mail icon in Profile activity

c. use case can start when “LiftOffer” use case is used.

d. new screen should contain “select - option” type of function that provides user way to select contact user. Up on each selection List-View should be populated with messages from that user.

e. Each time System user selects different contact , messages are repopulated.

7. StartProfile

a. Use case starts when User logs into the system

b. upon login session variable has to be set up.

c. constant connection with external 3th party server should be established for messaging.

8. Exit

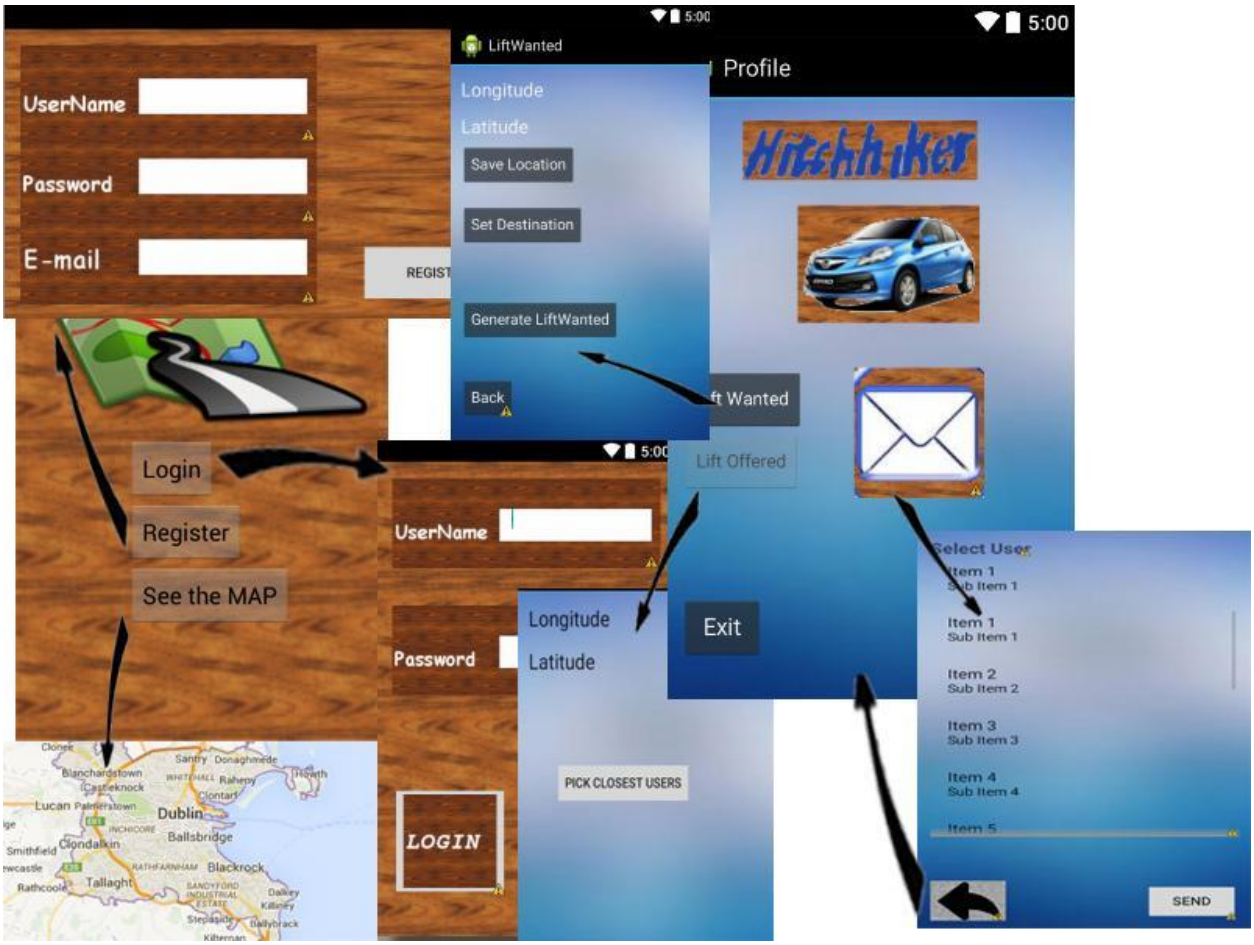
a. Use case starts when system user decide to exit the system

User Interface

Version 1 of “Car Sharing” App is more concentrating on core functionality than user Interface Design. Nevertheless some basic navigation should be provided (Allowing user to navigate towards one activity and back).

Suggested Screenflow overview

Initial screen design:



Car sharing App should have multiple activities connected with each other. Centristic activity is Profile where registered users can choose their action. Note that App has basic design and more concentrating on functionality.

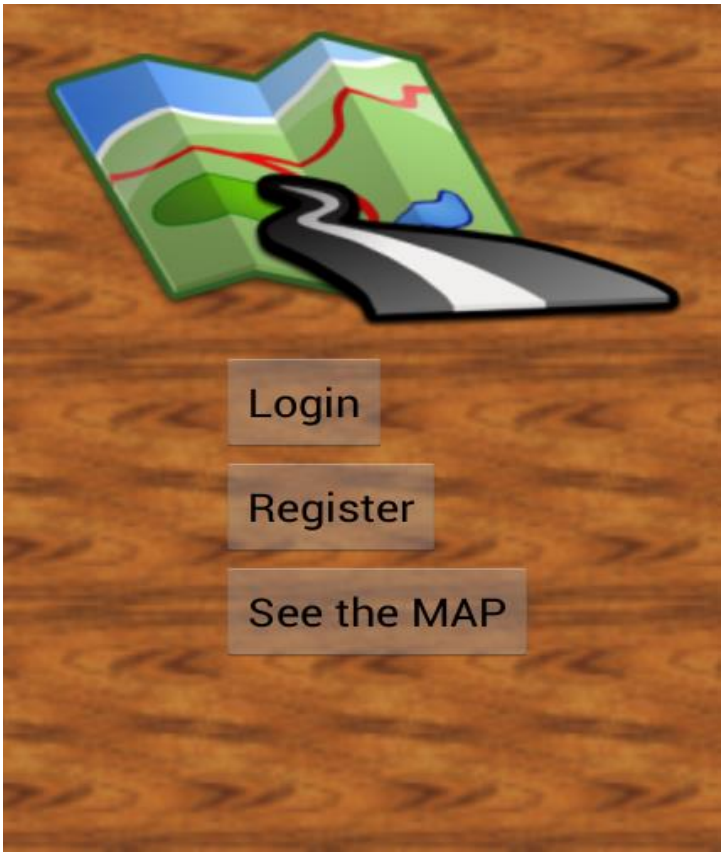
In this section each activity will be described from user point of view. Name of the activity will be fully qualified project name and placed underneath of the mock screen.

Screen by screen

Home activity

Displayed when user launches the APP. Screen has a 3 basic clickable Buttons.
Main purpose of the screen is to give access for members of the system. Additionally you can register or “see the map”.

1. Button “SEE THE MAP” allow people to see world map with all users on it. On clicking that “map_result” activity starts.
2. Button “LOGIN” Allows system members to log in. On clicking that button activity ”activity_login” starts.
3. Button “REGISTER” . Allows to become a member of the system. On clicking that button activity ”activity_register” starts.

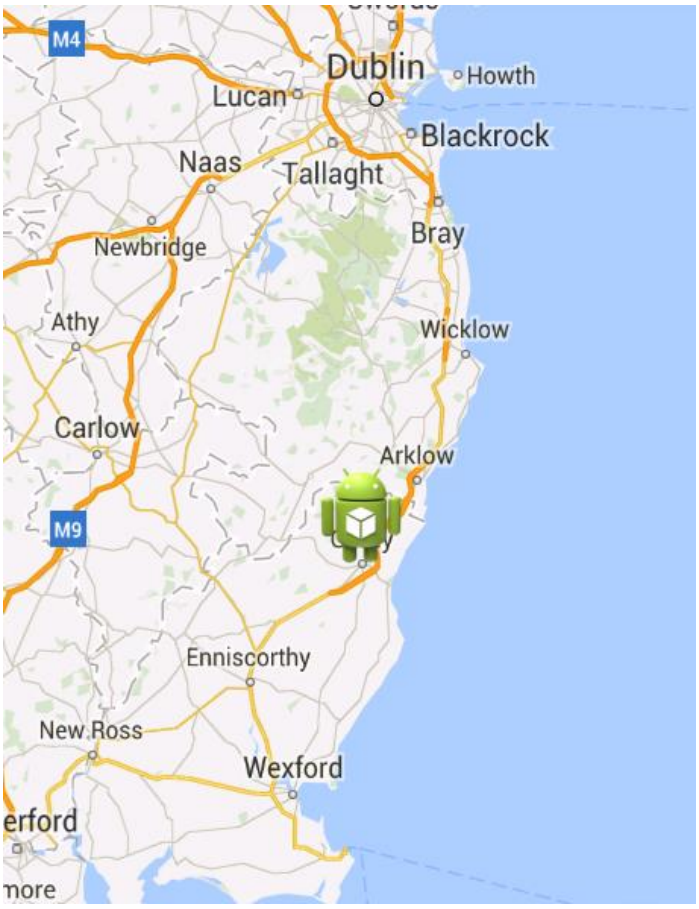


“See the Map”

Map activity shows google map with all users actively looking for lift at the moment. It wont give their names or destinations. It also doesn't have options for contacting these people.

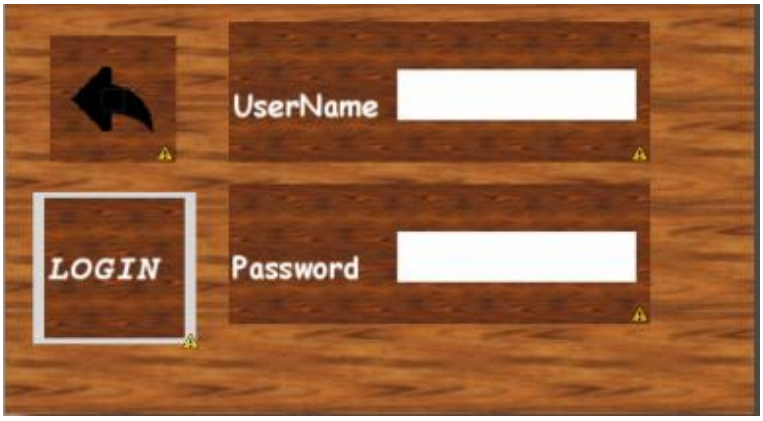
The purpose of the activity is to gather information about “what's happening outside”.

Example of map activity:



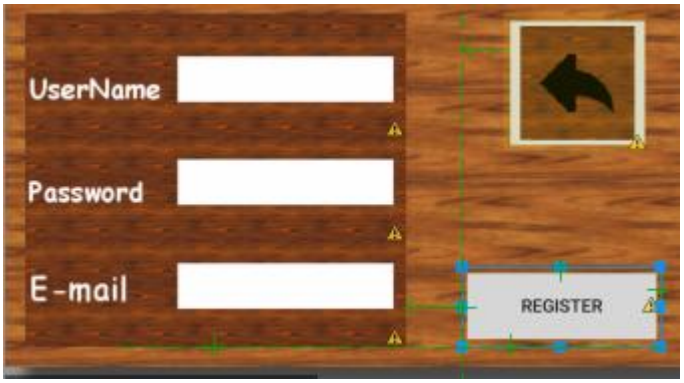
Login

This activity allows system members to log in.
User enters his name and password into the EditText fields and press button “Login”
When user clicks Login button , In case successful login - Profile(Profile.class, activity_profile.xml) activity is started.



Register

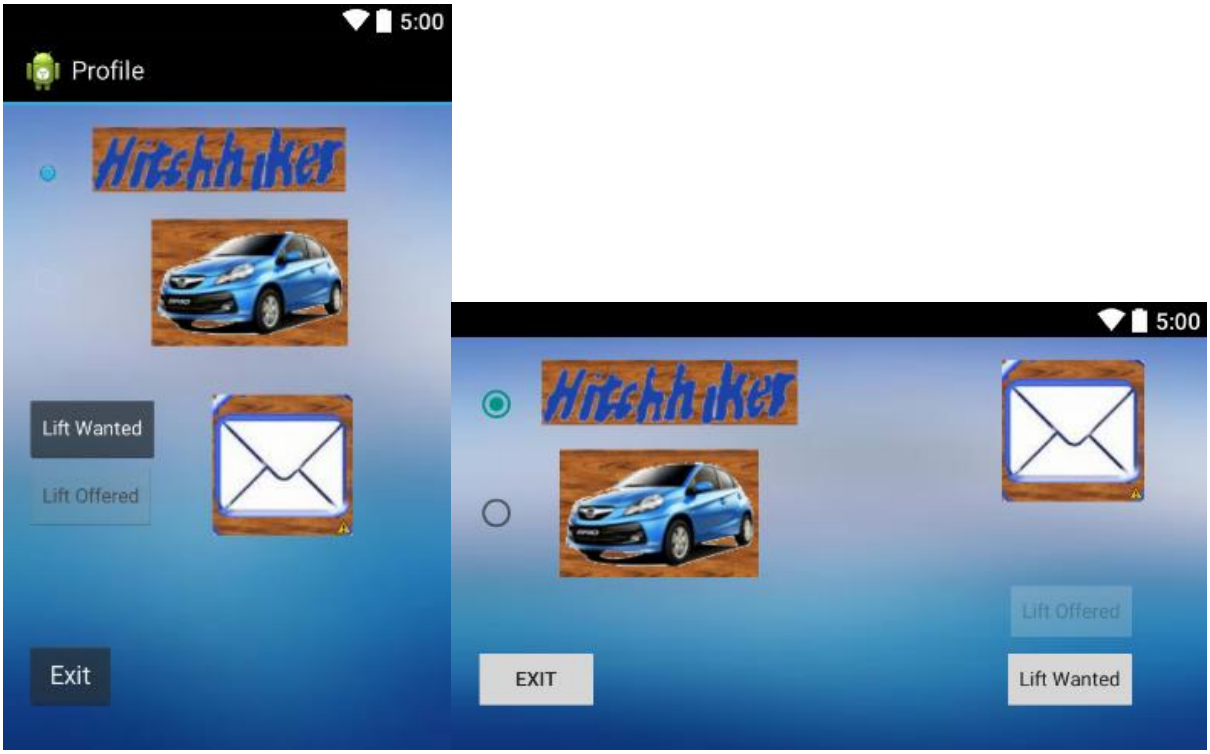
Displayed when Button “Register” was clicked from Home Activity.
Main purpose of the screen is to give anonymous user to register into the system. Screen has a 3 EditText fields placed above 3 different ImageViews and 1 button .



Profile

Profile is important activity, where person can make several options, based on needs. Profile activity should be accessible only by members of the system.
Profile activity should have option for Hitchhiker who can Generate Lift Wanted record in database.
Profile activity should have also option for Driver ,that want to see Hitchhikers around his own location. Activity should contain option for checking the messages.
In designed layout you can see

- 2 radio buttons - one for Hitchiker and one for Driver. If User of the system decide to be Hitchhiker he won't be able generate Lift Offered option. And vice versa if user of the system changes to Driver he can't generate “Lift Wanted”.
- image button for checking messages
- two “normal” button . “Lift Wanted” for Hitchiker and “Lift Offered” for Driver
- “Exit” button.



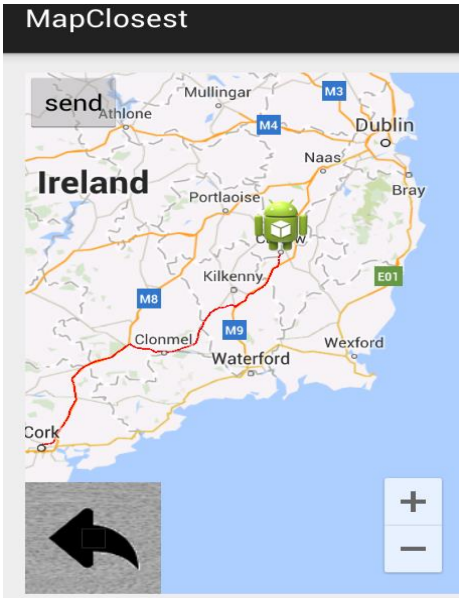
portrait and landscape view.

Lift Offered

This activity should allow Driver to determine his location. Required for next activity if he want to see Hitchhikers around him within some radius.
Activity has Image button to go back into the Profile and regular button named “Pick Closest users” to start next activity. Also there are two text fields for gps coordinates.

MapClosest

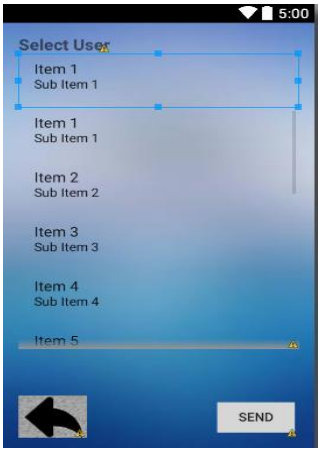
Button from (“Pick closest users”) Screenshot above should bring to Driver google map with Hitchhikers on it. Each Hitchhiker would have red line Route showing destination point.



Contact Users

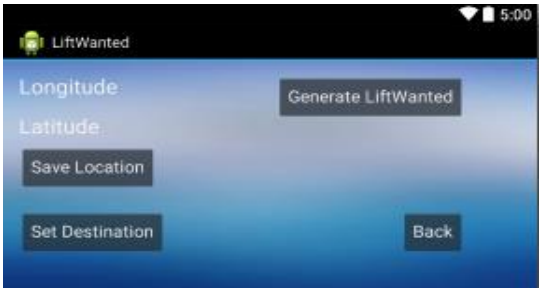
This activity should allow Driver to write message to selected user from previous activity.

It can also be accessed anytime from Profile by Hitchhiker or Driver - and in that case Selected users were previously stored into the system and you have replay option to any of them.
Layout looks like that:
Screen contains “Spinner” (see the Item1 inside the square) and List view (next Selection of Items). Spinner is going to have usernames to contact and List messages from those users.
Screen have also 2 buttons. 1 image button back to profile and second regular button to send text to selected user.



Lift Wanted

This activity should allow Hitchhiker to determine his location. Activity has 2 text field for GPS coordinates and 4 buttons . Buttons with self explainable names: “Save Location” , “Set Destination”, “Generate LiftWanted” and “Back” User click “save Location” to save his GPS coordinates on server. “Set Destination” should open google map where user can point destination. “Generate LiftWanted” should store this user query into cloud Database for Drivers to see him. “Back” button brings back into profile.



Non functional Requirements

- Delete private data on user request.

- security checks must be done on server side
- external Interfaces:
    - Google play services
    - web cloud services
    - MQTT