SYSTEM REQUIREMENT SPECIFICATION

CARPOOL SYSTEM WITH FACEBOOK INTEGRATION

(PoolSquare)

Project Title: **PoolSquare**

Group No: **12**

2014MCS2802	Raunak
2014MCS2121	Arjun Singh
2014JCA2466	Naman Agarwal
2014JCA2471	Sumit Singh

TABLE OF CONTENT

TABLE OF CONTENT

1. Introduction

1.1. Purpose

Aim of this software specification requirements document is to provide a complete description of all of the features that are planned to implement to system and define the expectations from the Carpool project. It also describes how the system operates and how users interact with the application. Besides external systems and interfaces which the application depends, are specified in this SRS document

The potential audiences for this document are design and development team of the Carpool Project in order to specify software designs.

1.2. Scope

There have been many applications that avail the users to pool care rides with other people. This project aims at bringing some sort of reliability with whom users pool with by integrating users' facebook account with the application.

PoolSquare is an android based application. It is going to provide communication environment for its users (rider and offerer). The user can set his/her preference of which facebook friends he/she wants to pool with e.g friends, friends of friends and so on.

The offerer can offer rides between two points. This ride will be visible to those of his facebook friends as his setting. A rider can search for rides between two points. He will get a list of all intersecting rides from his facebook friends list. Intersection of rides is decided based upon shortest route taken from Google Map API.

1.3. Definition, Acronyms and Abbreviations

The definitions of the terms, which are used in this SRS document, are shown below:

Terms	**Definitions**
User	Offerer and Rider
GUI	Graphical User Interface

DBMS	Database Management System
IEEE	Institute of Electrical and Electronics Engineers

SRS	System Requirements Specification
API	Application Programming Interface
PHP	Hypertext Preprocessor
Rider	The user requiring to ride in some vehicle
Offerer	User who owns the car and is offering the ride
Route	Transportation path
SMS	Short Message Service
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart

1.4. References

[1] IEEE STD 1233-1998, IEEE Guide for Developing System Requirements Specifications

[2] IEEE STD 830-1998, IEEE Recommended Practice for Software Requirements Specifications

[3] senior.ceng.metu.edu.tr/2014/such/documents/SRS.pdf

1.5. Overview

The rest of the document contains overall description of the system which includes interface properties, use cases, context diagrams, system design, software attributes, product functions and dependencies. It also contains functional and non-functional requirements of the system. Various data and description models of the system have been documented.

---

2. Overall Description

This section will give an overall viewpoint of the carpool application.

2.1. Product Perspective

2.1.1. System Interfaces

The system uses Google Map API to fetch the route between endpoints as specified by the user.
The system also uses Facebook API to fetch friends' list of the user.

2.1.2. User Interfaces

The android based system will have a login/register page. The user will have a screen to book a ride between two places and to offer a ride between two places. The offerer can create a one off ride or a recurring ride. The rider upon requesting a ride will get a screen showing the list of available rides from his friends that intersect his route. He can then select a ride depending upon his choice. On selecting, the corresponding offerer will get a notification. The offerer can respond to requests by riders confirming the ride with its amount. The offerer may decide to change the default amount. The rider upon viewing the response will finally confirm the ride.

The User can view the history of rides he has finished so far. He can rate the rides based upon experience. He has the option to change his account settings, view all the rides he currently offers.

2.1.3. Hardware Interfaces

PoolSquare will be compatible with all the hardware persistent in the market that allows Android 2.3 and above to run.

2.1.4. Software Interfaces

Amazon cloud service will be used to host the app.
MySQL database management system is used as the DB application.
Google Map API and Facebook API will be used to fetch information about the route and friends of the user respectively.

2.1.5. Communication Interfaces

The system will use Push notification services provided by Android as means of communication between devices to show requests and responses.

2.1.6. Memory

Smartphone with 2GB RAM or more.

2.1.7. Operations

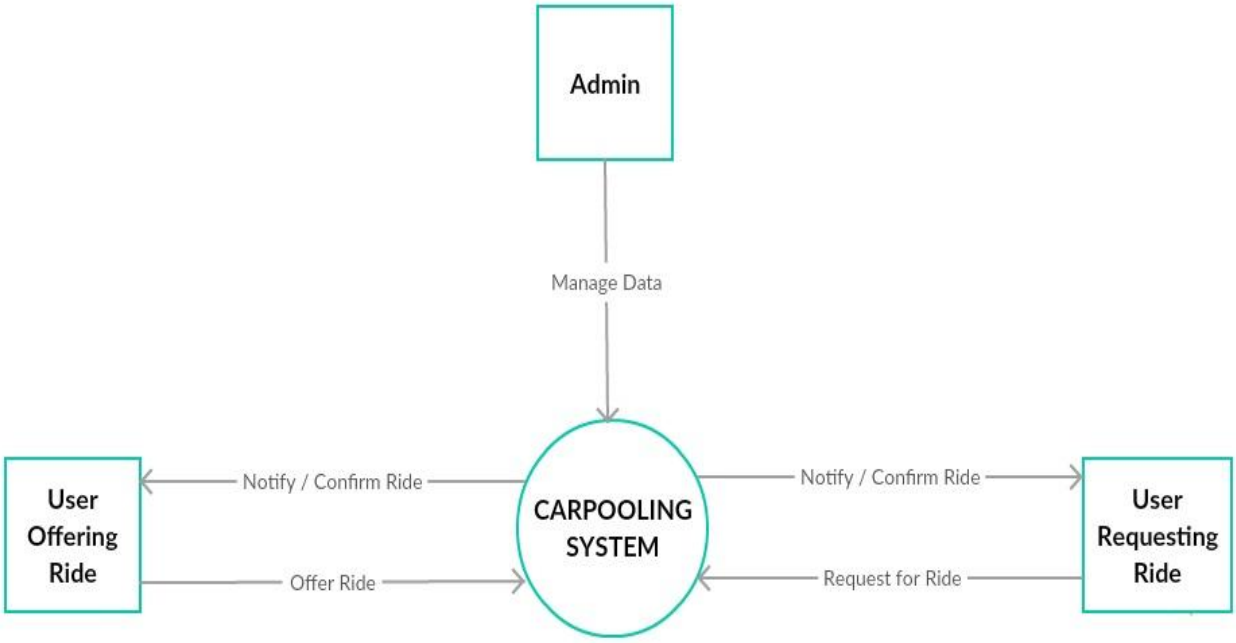
N/A

2.1.8. Site Adaptation Requirements

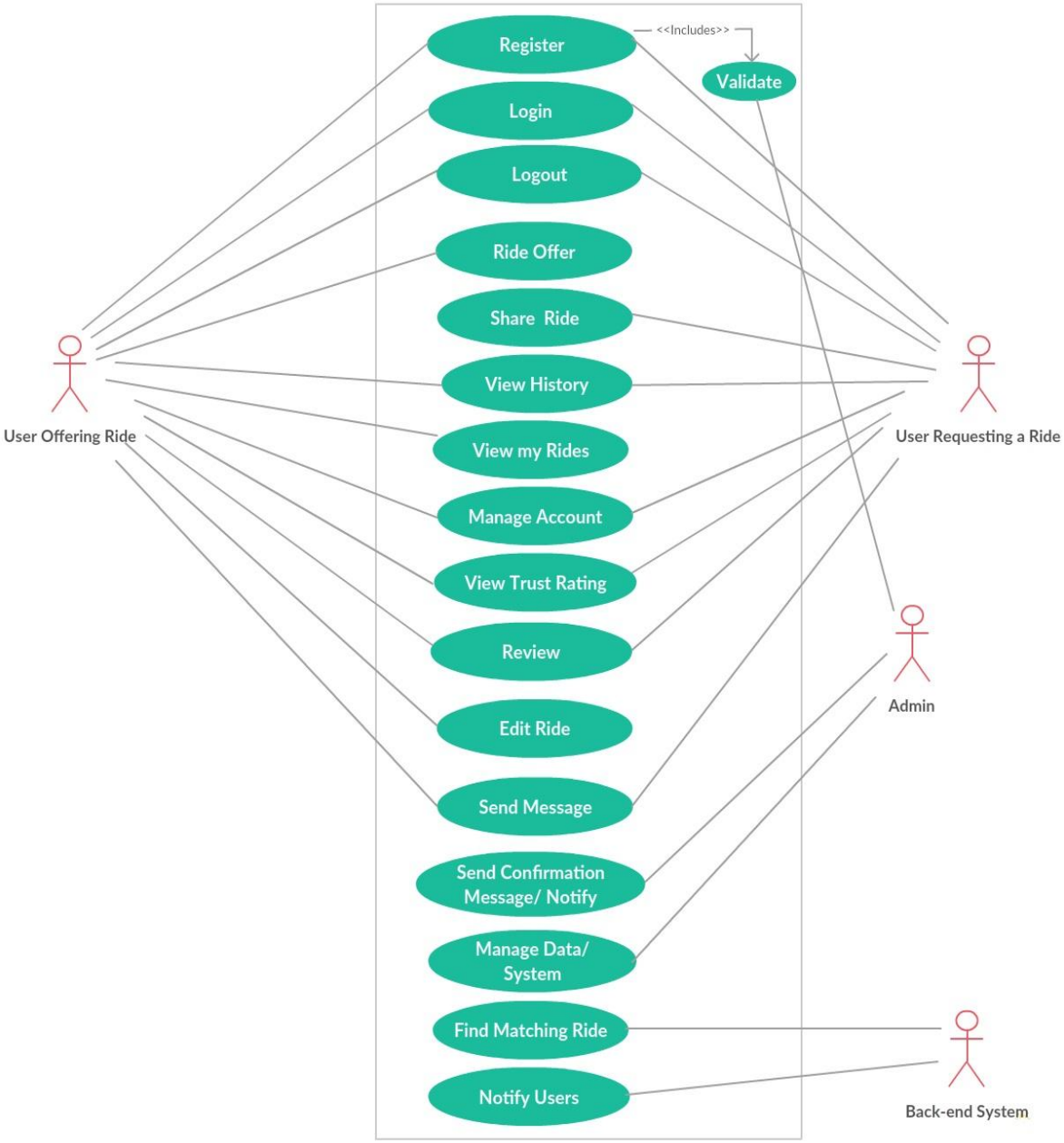
PoolSquare should be able to run on any smartphone that supports Android 2.3 or above.

2.2. Product Functions

2.2.1 Context Diagram



2.2.2 Use Case Diagram



2.2.3 Use Case Descriptions / Introductions

2.2.3.1. Sign Up

Before using the application user will have to register by providing the basic information like – Name, Phone, Address, Govt Id and password.

2.2.3.2. Sign In

The user shall be able to sign in to the system. User will have to provide his/her mobile number and password, provided at the

time of registration. User shall be redirected to application main page on successful sign in.

2.2.3.3. Sign Out

User shall be able to leave the system at any point of time. Application login page will be loaded.

2.2.3.4. Book Ride

User shall have the option to book a ride by providing basic information like – From, To and Timing. User shall be redirected to 'Select Ride' page. On 'Select Ride' page user shall be shown a list of available options (List of users who own the car and want to share the ride on same route). User shall be able to select multiple options from this list. Once the response come from any of the selected option, User will have to confirm the ride.

2.2.3.5. Offer Ride

User shall be able to offer his/her car via this option. User will have the option to offer the ride on the regular and one time basis. Once user has created a ride,it shall be automatically shared.

2.2.3.6. Notifications

User shall get the notification whenever some other user wish to share ride with him/her. User will be able to accept/reject (do nothing) to offer the requested ride. Once the ride request is accepted, a notification will be sent to user who have requested for the ride.

2.2.3.7. Account Setting

User shall be able to manage his/her account. User will be able to change its password, address, add/remove vehicle (If offer rides).

2.2.3.8. View History

User will be able to see his/her rides history at any point of time.

2.2.3.9. Dashboard

On login the user shall be able to see the dashboard. On dashboard user will be shown two lists –

- Offered Rides – User will be shown the list of rides which he has offered along with a number indicating the number of requests for this ride. User shall be able to confirm/reject these requests.
- Requested Rides – User will be shown the list of rides which he has requested along with a number indicating the number of responses received for this ride. User shall be able to

confirm/reject these offers. User will be able to confirm only one response per ride.

2.2.3.10. Send Message

On completion of handshake between offerer and rider, a message will be exchanged between them, showing contact information.

2.2.3.11. Review User

Rider and Offerer shall be able to rate each other.

2.2.3.12. Track Ride

User shall be able to track a confirmed ride.

2.2.4 User Characteristics

The user is expected to understand English language and should be able to use an android app.

2.2.5 Constraints

Carpool system is required mutual trust for example user's security of life must be protected by the government's law system but there is no legal infrastructure about this driver and hitchhiker relation in our country. So, this is an important constraint for Carpool system. Another constraint is that the system requires remote server which enables the system functionality and data storage. Because of this situation, when the server crashes the system will not be able to its operations until the server become available to respond system requests. In addition to these, since the user information is stored in a database and this database can be hacked and user information will be no longer private to the user. To sum up, Carpool system has constraints in terms of regulatory, reliability, safety and security but these constraints can be manageable.

2.2.6 Assumption and Dependencies

User interface and some functionalities can change during the development process of project. And also new functionalities can be added which is able to change the dependent system requirements.

---

3. Specific Requirements

3.1 Interface Requirements

The system shall consist of user friendly interfaces which are explained in Product Functions' section. Also, visualized version of the interfaces are explained in Description for Software Behavior section.

3.2 Functional Requirements

3.2.1 Use Cases

3.2.1.1 Share Ride

Introduction: Allow User with a vehicle to share a ride with other users.

Actors: User offering ride

Pre-Conditions: User must be logged onto the system and should have a registered vehicle before this use case begins.

Post-condition: If use case is successful, the user's request to share ride and its details is stored in the database. Otherwise, the information is discarded.

Basic Flow: Starts when User wishes to share a ride. The system requests the User to specify the source, destination, timing, vehicle, frequency and vacancy of the ride. The System checks information entered by the user and validates the ride by adding the relevant details to the database.

Alternative flows: None

Special requirements: None

Use case relationships: None

3.2.1.2 Book a ride

Introduction: Allow User with a vehicle to book a ride shared by other users.

Actors: User requesting for ride

Pre-Conditions: User must be logged onto the system before this use case begins.

Post-conditions: If use case is successful, the user's request to book ride and its details is stored in the database and a list of rides offered by other users is displayed to the user.. Otherwise, the information is discarded.

Basic Flow: Starts when User wishes to book a ride. The system requests the User to specify the source, destination and timing of the ride. The System checks information entered by the user and validates the ride by adding the relevant details to the database. The system finds the rides being offered by other users which match with the source, destination and timing of the ride requested by the current user. The user clicks on the rides which he wants to book and confirms the ride.

Alternative flows:

Matching Ride Not Found:
If a matching ride is not found then the system does not display the details of any ride to the user.

Special requirements: None

Use case relationships: None

3.2.1.3 Register

Introduction: Allow User to register himself on the app.

Actors: User

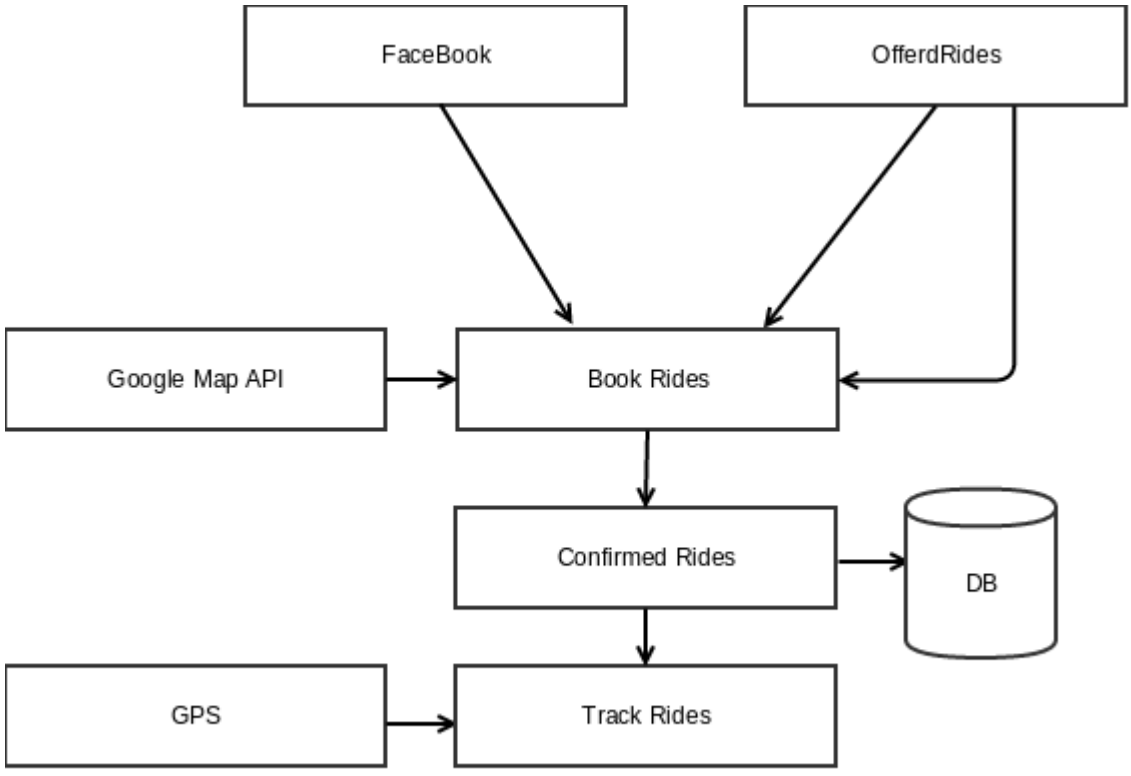
Pre-Conditions: User should not be already registered on the app.

Post-conditions: If use case is successful, the user is registered successfully and his information is stored in the database.

Basic Flow: The user opens the app and clicks the register button. Then the user enters his/her name, address, phone, emailed, ID proof, vehicle information, password and clicks on submit button.

Alternative flows: If any of the information entered by the user is found out to be improper, then a warning message is shown to the user.

3.2.2 Data Flow Diagram



3.3. Performance Requirements

The user shall be able to enter a page of the system in less than 1 second. The system shall be able to respond more than one thousand users simultaneously. The system shall be able to keep user information of more than one hundred thousand users.

3.4. Logical Database Requirements

There are five main data entities following table describe logical classification of those data entities as well as their attributes.

Data Entity	Attributes	Use
User_Info	• User_Id • Name • Gender • Age • Email_Id • Password • Govt_Id_Proof • Avg_Rating • No_of_Ratings • No_of_Fb_Friends • Friendship_Level	This data is used to store user information and to view user profile.
Regular_Ride_Info	• Ride_Name • User_Id • Source • Destination • Departure_Time • Arrival_Time • Seats_Available • Ride_Cost • Repetition	This Data entity is used to store information of regular rides corresponding to each user.
Offered_Ride	• Ride_Id • User_Id • Source • Destination • Date	This data stores information about the rides offered by all the users.

	• Departure_Time • Arrival_Time • Available_Seats • State	
Requested_Ride	• Ride_Id • User_Id • Source • Destination • Date • State	It stores search results of all rides queried by users.
Live_Ride	• Offered_Ride_Id • Requested_Ride_Id	This joins rides being offered and requested

3.5. Design Constraints

3.5.1. User must give access to his Facebook account friends list.

3.5.2. Hopping points of the ride are decided on the basis of Google Map data.

3.5.3. Cost of the ride is decided on a pro rata basis.

3.6. Software System Attributes

3.6.1. Availability

Application must be in available 24X7.

3.6.2. Usability

Application must be easy to navigate and easy to use. Any user without any prior knowledge about system must be able to use it.

3.6.3. Extensibility

Application must be easily extensible. Any Extension should be easily implementable at any point of time. New extension should never effect the functionality of previous version of application.

3.6.4. Interoperability

Application could be used different version of Androids.

3.6.5. Scalability

Application should be scalable.

3.6.6. Response Time

Response time for any request must be feasible.

3.6.7. System Security

User's data must be protected.

3.7. Organizing Specific Requirements

N/A

3.8. Additional Comments

N/A

---

4. Supporting Information

4.1 Process Model

The project will be processed by Waterfall Model. Waterfall Model is plan driven process which is listed below consecutively.

- Cleared requirements,
- System and software design,
- Implementation and unit testing,
- Integration and system testing,
- Operation and Maintenance.

5. Conclusion

In this document, the functional and other requirements of the system are described. Furthermore, the needs of the user are stated through the document. However, all requirements are not defined and some of the requirements needs to be clarified in this document.