IITD

DESIGN SPECIFICATION

CarPool App With Facebook Integration

Naman, Arjun, Sumit, Raunak

4/4/2016

Project Title: **PoolSquare**

Group No: **12**

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

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

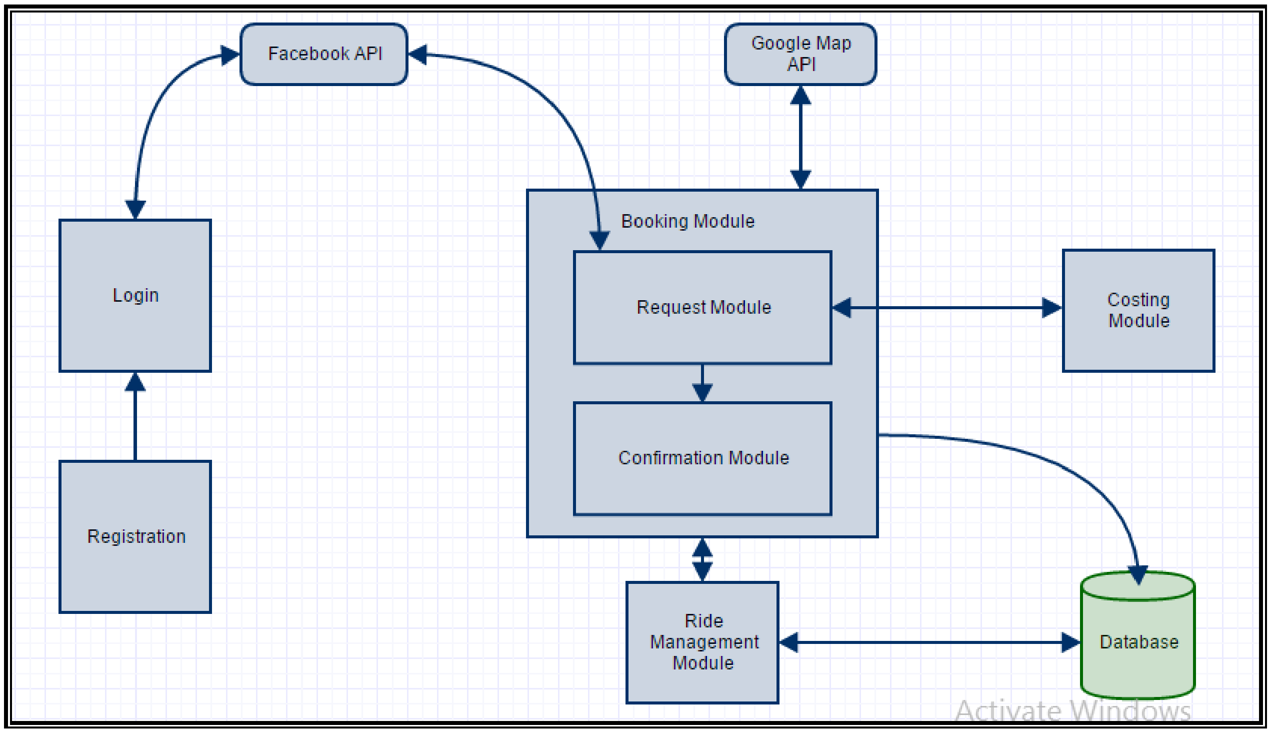
1.5. Overview

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. System Overview

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

2.1. Architectural Design



Overall Architecture

2.2. Module Definitions

2.2.1. **Registration Module**

The user on coming to the app for the first time will register into the system with necessary details such as name, address, email etc.

2.2.2. **Login Module**

Login module will communicate with Facebook API based on facebook user ID of the user and will login the user into the system.

2.2.3. **Booking Module**

Comprises of two sub-modules:

a) **Request Module** – This handles users sending ride request to ride offerers.

b) **Confirmation Module** – This allows offerers who receive ride requests to either accept or reject the request.

2.2.4. **Ride Management Module**

Allows users to create new rides that can be offered later.

2.2.5. **Costing Module**

Calculates cost of shared ride based upon route intersection as received from Google API.

2.2. Technologies/Tools Used

Android Studio to be used as standard tool for app development. Facebook API & GoogleMap API is used for getting users friends and route information respectively.

3. Detailed Design

3.1. Module APIs

Login

signIn()	**Method Description**
	A method to login into carpool system
	Input
	User_id, pasword
	Output
	Boolean
logOut()	**Method Description**
	A method to logout from the carpool system
	Input
	NA
	Output
	NA

Registration

signUp()	**Method Description**
	A method to register into carpool system
	Input
	NA
	Output
	NA
setInfo()	**Method Description**
	A method used inside signUp method
	Input
	user_id, Name, email, phone, address, govt_id
	Output
	boolean
changeInfo()	**Method Description**
	A method to change the user info
	Input
	user_id, Name, email, phon, address, govt_id
	Output
	boolean

Booking Module

bookRide()	**Method Description**
	A method to book a ride
	Input
	from, to, timing
	Output
	boolean
getRideList()	**Method Description**
	A method called from bookRide() to get the available

	rides on the basis of rout and the fb friends of user
	Input
	user_id
	Output
	ride list
getFbFriends()	**Method Description**
	A method called from bookRide() to get the facebook friends who offer the rides
	Input
	user_id
	Output
	All facebook friend list
getRideInfo()	**Method Description**
	A method called from bookRide() to get the information of a ride
	Input
	from, to, ride_id
	Output
	Ride Information and cost
getOfferedRides()	**Method Description**
	A method called from bookRide() to get the available list of rides
	Input
	friend list, timing
	Output
	List of offerd rides by each friend
sendRequest()	**Method Description**
	A method to send the request to book the selected ride
	Input
	ride_id, offerer_id
	Output
	boolean

Confirmation Module

getRequestRides()	**Method Description**
	A method to get the requested ride for a offerer
	Input
	user_id
	Output
	List of requested Rides
getRidesInfo()	**Method Description**
	A method to get the information for the requested rides
	Input
	ride_id, offerer_id
	Output
	List of requested rides' information
confirmRequest()	**Method Description**
	A method to confirm a requested ride
	Input
	ride_id, offerer_id
	Output
	boolean

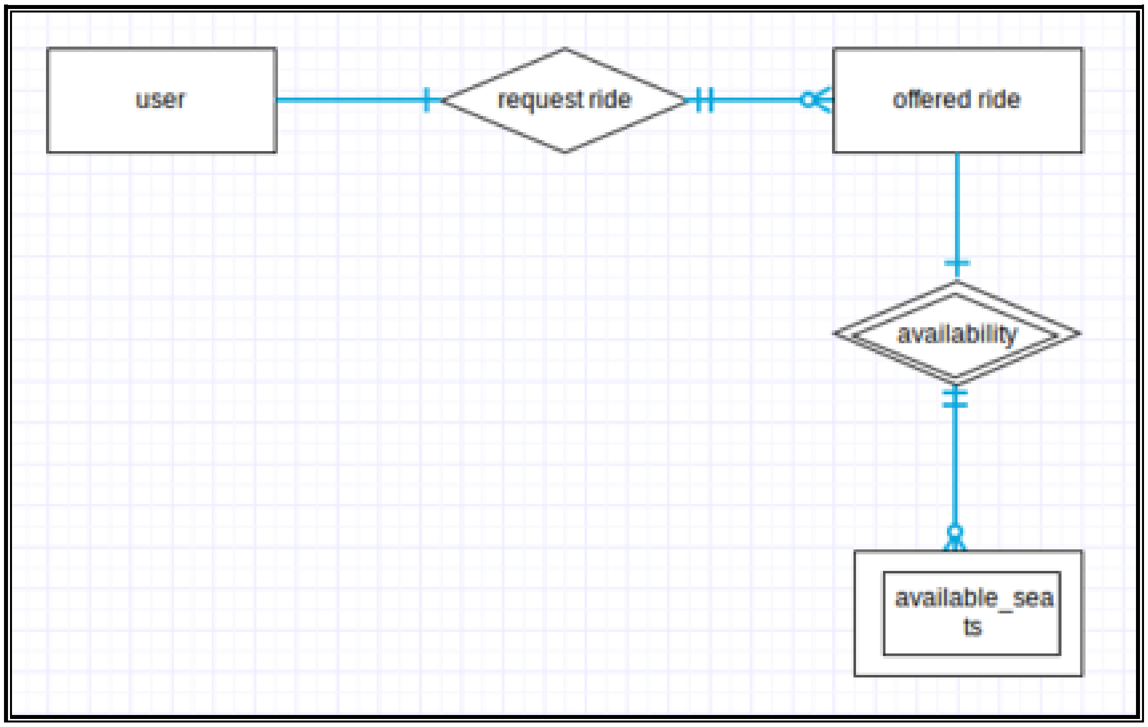
Ride Management Module

createNewRides()	**Method Description**
	A method to create new rides to be offered
	Input
	user_id, from, to , timing, vhicle_id, ride_name
	Output
	NA

Costing Module

getRouteCost()	Method Description
	A method to get the actual cost of the ride
	Input
	distance, fare
	Output
	ride cost

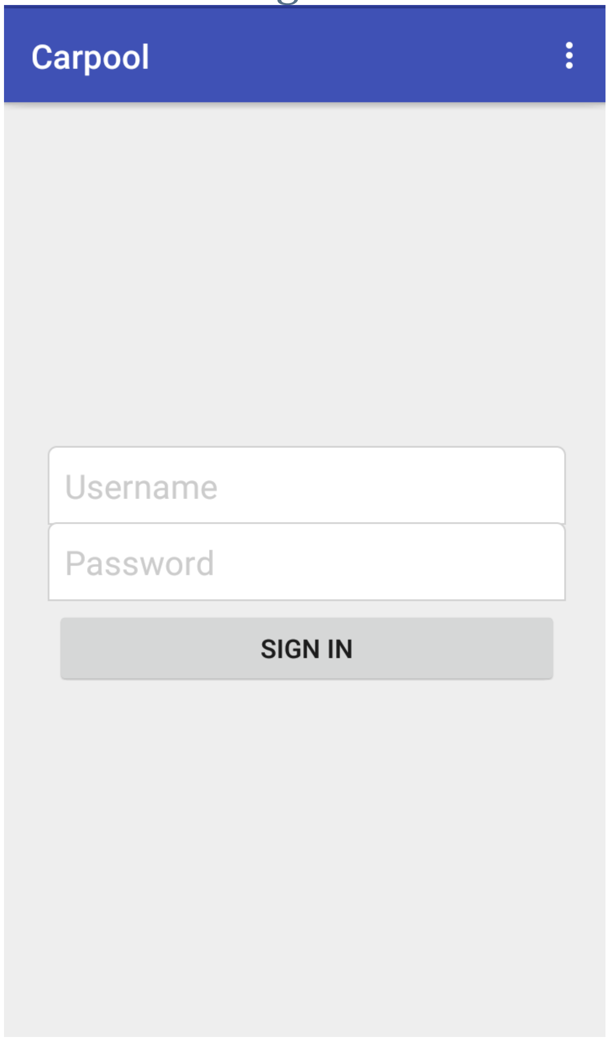
3.2. Data Base Design



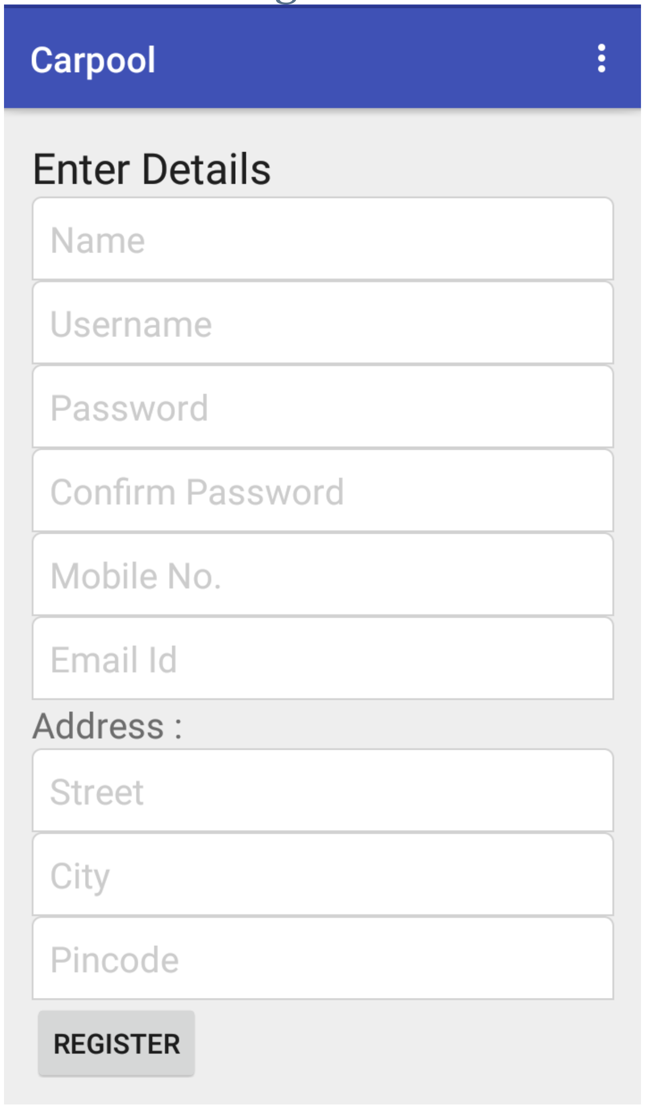
ER Diagram

3.2. Screen Layouts

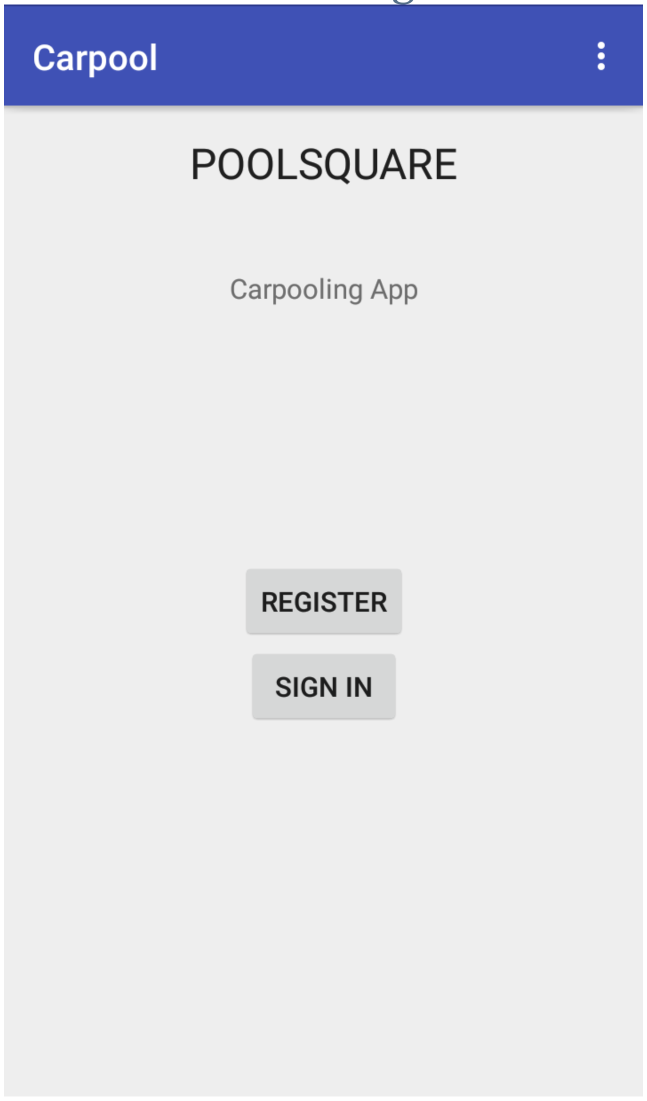
Login



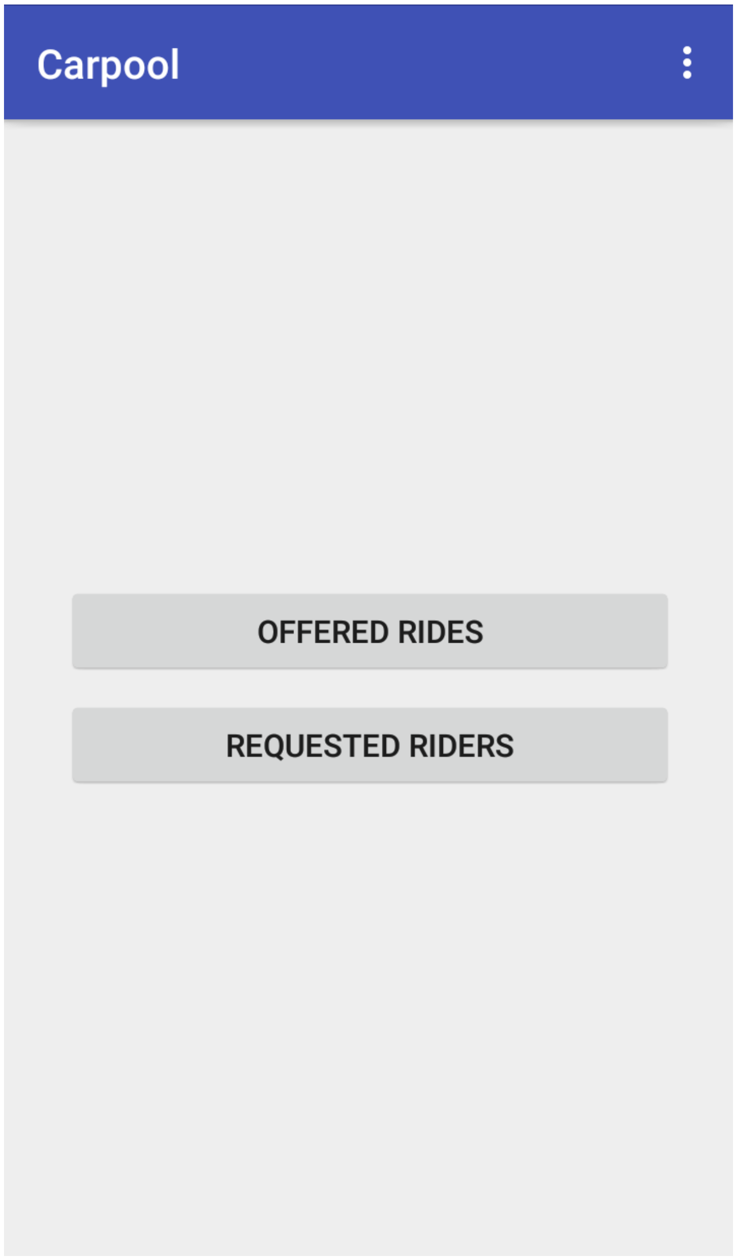
Register


Carpool
Enter Details
Name
Username
Password
Confirm Password
Mobile No.
Email Id
Address :
Street
City
Pincode
REGISTER

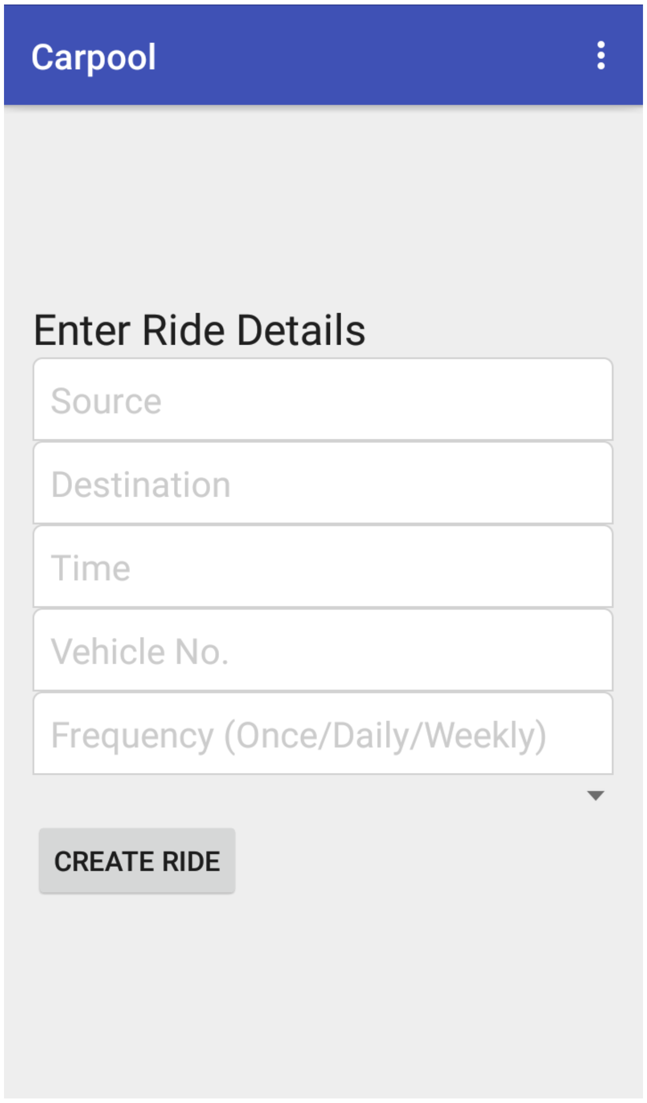
Home Page



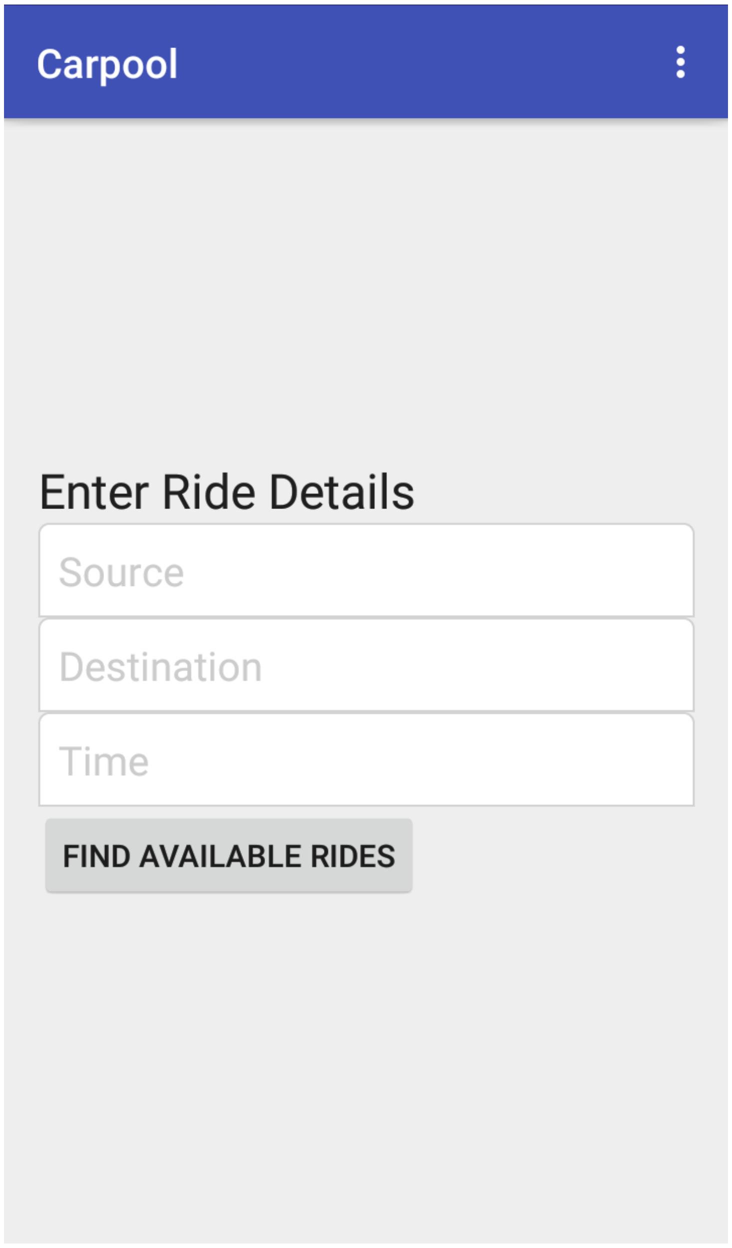
Dashboard



Create Ride


Carpool
Enter Ride Details
Source
Destination
Time
Vehicle No.
Frequency (Once/Daily/Weekly)
CREATE RIDE

Book Rides


Carpool
Enter Ride Details
Source
Destination
Time
FIND AVAILABLE RIDES

3.2. Use Cases

a) High Level Code

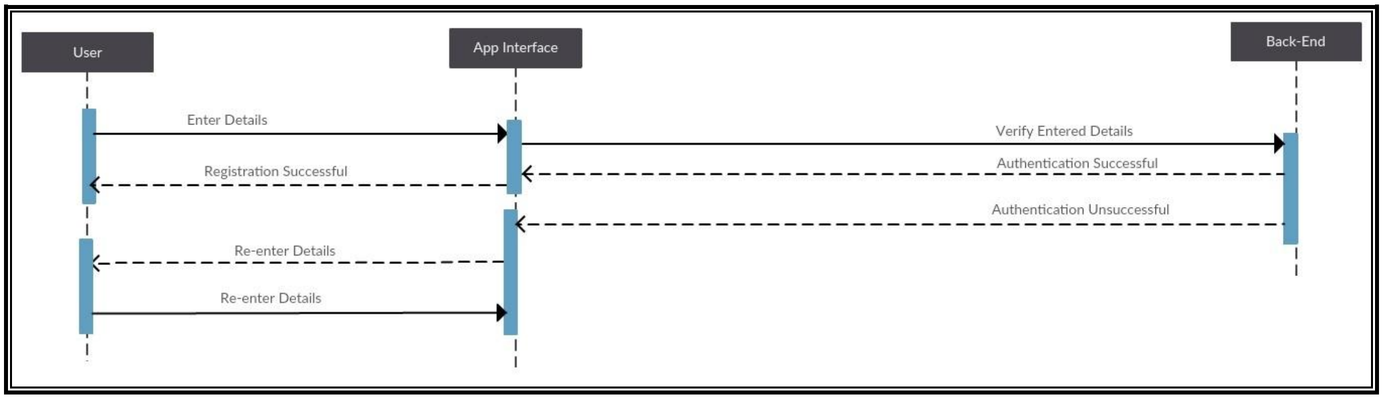
```
1. SignIn() {
   if(user_id && password)
           return success
   else
           return fail
   }


2. logOut() {
           return to login page
   }

3. signUp() {
           setInfo()
   }

4. setInfo() {
           enter the information of user into database
   }

5. changeInfo() {
           get the new Values
           setInfo()
   }

6. bookRide() {
           getFbFriends()
           getRoutInfo()
           getRideList()
           getOfferedRides()
           return sendRequest()
   }
```

```
7.  getFbFriends() {
            using Fb API get the list of facebook friends
            return friend list
    }
```

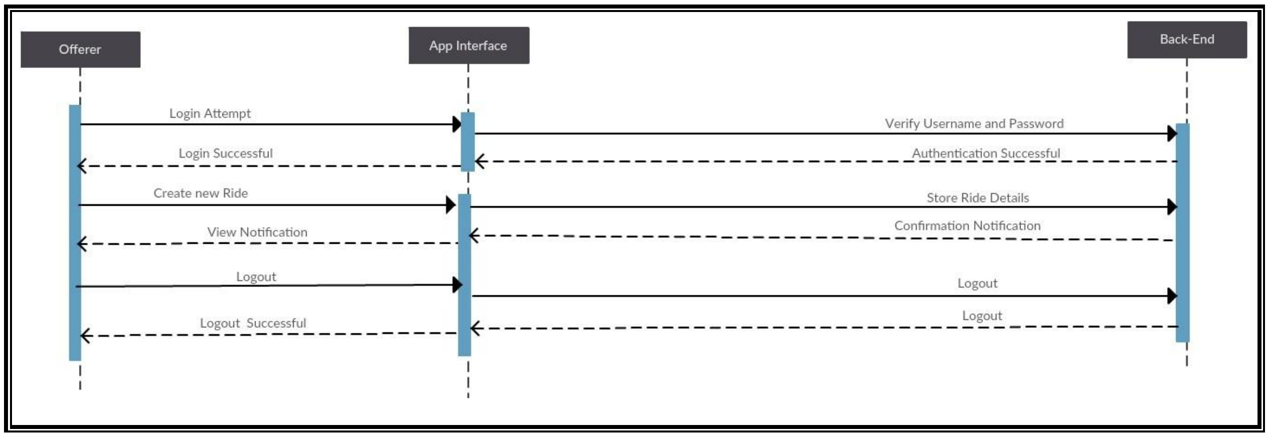
```
8.  getRoutInfo() {
            cost = rout length * fare
            return the cost of route for each Friend
    }

9.  getOfferedRides() {
            return the list of offered ride for the desired route
    }

10.         sendRequest() {
                    send the request for a selected ride
                    if(send successfully)
                            return true
                    else
                            return false
            }

11.         getRequestedRides() {
                    get the list of requested rides from database
                    return list
            }

12.         getRideInfo() {
                    select a ride
                    get the information of ride from database
                    return ride
            }
```

```
13.    confirmRequest() {
           if(success)
               return true
       else
           return false
       }

14.    createNewRide() {
           get the user data for offered ride
           insert offered ride into database
       }

15.    getRoutCost() {
           return intersect rout length * fare
       }
```

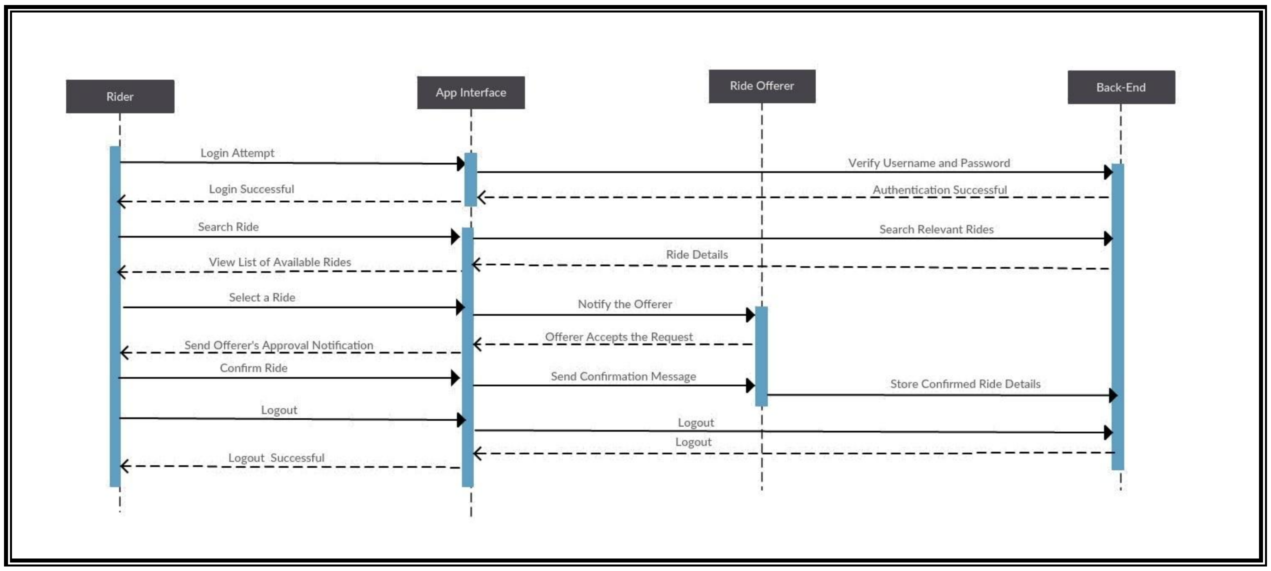
b) Sequence Diagrams



Registration Sequence

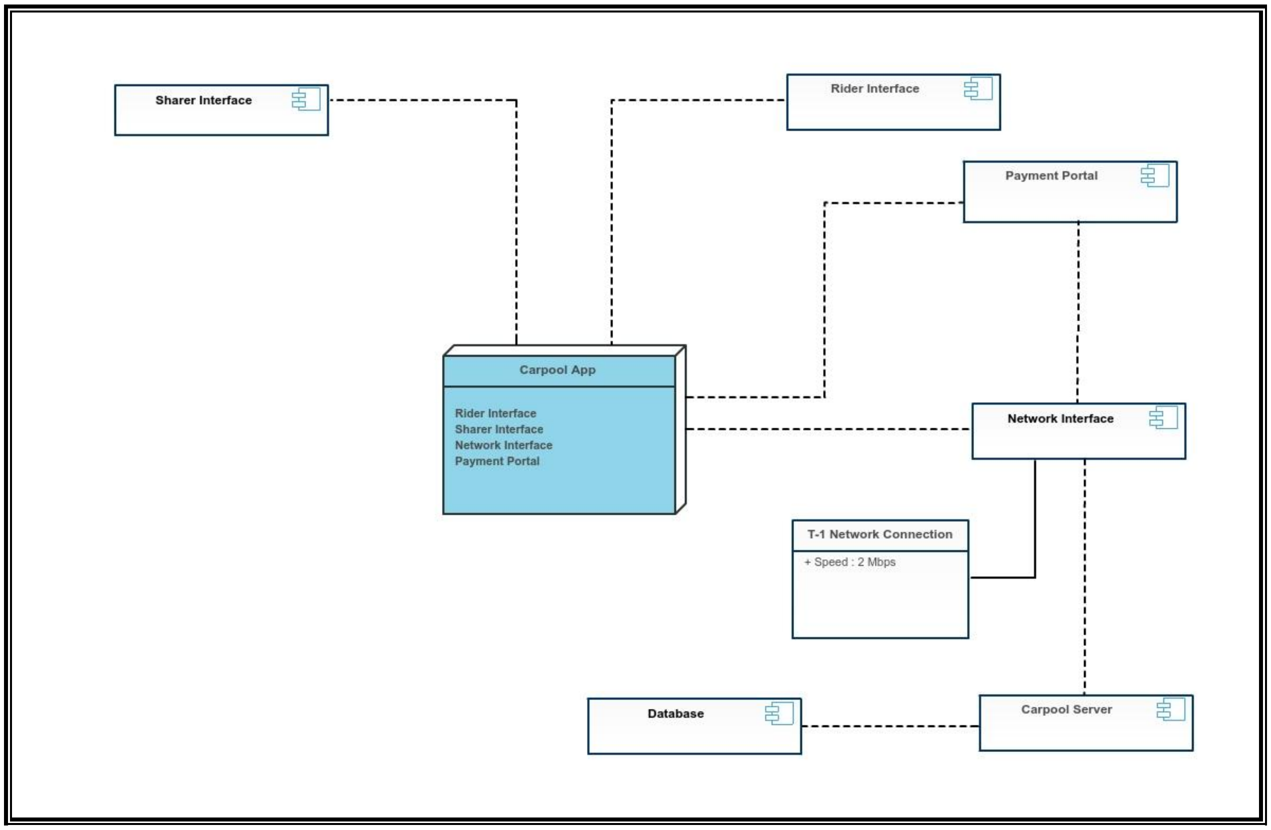


Offer Ride Sequence


Rider
App Interface
Ride Offerer
Back-End
Login Attempt
Verify Username and Password
Login Successful
Authentication Successful
Search Ride
Search Relevant Rides
View List of Available Rides
Ride Details
Select a Ride
Notify the Offerer
Send Offerer's Approval Notification
Offerer Accepts the Request
Confirm Ride
Send Confirmation Message
Store Confirmed Ride Details
Logout
Logout
Logout
Logout Successful

Rider Sequence

5. Deployment Design



Deployment Diagram