



Bilkent University
Department of Computer Engineering

Senior Design Project

CarBuds

ANALYSIS REPORT

Ahmet Emre Nas	21402357
Doğukan Altay	21400627
Ali Osman Çetin	21302483
Aras Heper	21302248

Supervisor: Uğur Doğrusöz

Jury Members: Ercüment Çicek, H. Altay Güvenir

Innovation Expert: Doğukan Şengül

March 19, 2018

Table of Contents

1. Introduction	1
2. Current System.....	1
3. Proposed System	2
3.1. Overview	2
3.2. Functional Requirements	2
3.3. Nonfunctional Requirements	3
3.4. Pseudo Requirements	4
3.5. System Models	5
3.5.1. Scenarios	5
3.5.1.1. User Signup	5
3.5.1.2. User Login.....	6
3.5.1.3. User Landing Page.....	7
3.5.1.4. User Setup Hitchhiker Profile.....	7
3.5.1.5. User Setup Driver Profile	8
3.5.1.6. User MatchMaking Page.....	9
3.5.1.7. User Set Trip As Driver	9
3.5.1.8. User Set Trip As Hitchhiker	10
3.5.1.9. User Notification	10
3.5.2. Use Case Model.....	11
3.5.2.1. Register & Login Use Case.....	11
3.5.2.2. Registered User System Use Case.....	12
3.5.1. Object and Class Model	14
3.5.2. Dynamic Models	16
3.5.2.1. Register And Log In Activity Diagram.....	16

3.5.2.2.	Driver General Usage Activity Diagram	17
3.5.2.3.	Passenger General Usage Activity Diagram	18
3.5.2.4.	Register And Login Sequence Diagram	19
3.5.2.5.	Being Hitchhiker Sequence Diagram.....	20
3.5.2.6.	Being Driver Sequence Diagram	20
3.5.2.7.	Update Profile	21
3.5.2.8.	User Interface –Screen Mock-Ups	22

1. INTRODUCTION

In Universities, students create networks in order to solve their problems collectively. The problems can vary in terms of buying used/old books from other students or exchange stuff and most of the time sharing a ride. Most of these networks established in internet environment which not mean to use for a specific problem of students. Thus, Student can have a hard time to find help or another student fast and precise. Finding a ride or sharing your car with another person rises several issues, the most important one is the security. Inside the university campus most of the drivers can be sure that the hitchhiker most probably another student of the same school. However, outside the campus no one can assure that the hitchhiker even a student. To solve this problem students reach each other through un-specified networks and this arises another issue, time.

CarBuds is a mobile application that any university student can register to reach hitchhikers or can be hitchhiker to find a ride to a specific location. The application offers to find a car or a hitchhiker very quickly while assuring the validity of the other person's whether he or she a student of the proposed university. Therefore, with the help of the Carbuds students can reach each other for sharing a ride in a fast and secure way.

2. CURRENT SYSTEM

In the recent years usage of car sharing apps increased significantly. All these applications aimed difference purposed. For instance, in USA only cars with at least 4 passengers can use left line and therefore most of the apps used in USA aims to find enough people for using left line. Lyft Line [1] is one of the example that used in America for this purpose.

In Turkey apps like BlaBlaCar [2] used as an alternative to travel between cities. With apps like BlaBlaCar people find drivers with same destination city and same start city and split the price of fuel.

Both of these applications do not aim a specific target user and everyone can use them as long as passengers accepted to pay some kind of payments or split fuel expense.

3. PROPOSED SYSTEM

CarBuds is a mobile application which find driver or passenger with same destination and close to each other. This application has target audience of University students and do not permit any kind of payment to driver or the application itself. The only purpose of the application is finding people with same school and help students to go school easier.

The users required to sign up with their school email addresses in order to increase security. People can match with people near to them and have same destination. In order to be matched bot driver and passengers need to agree before they are chatting because there can be cases like people may not want to go with opposite genders.

3.1. OVERVIEW

As discussed, CarBuds will match people who are going to same place. The application will be use by students. If a driver starts a trip system will show people who wants to go to same place with the driver and match them if both driver and passengers agreed to go with each other. After they agreed to drive together they can start chatting. Passengers will not pay anything to driver. Bilkent University Students use Facebook pages for helping students without car to take to school or take to home. But the chance of finding passenger or driver on Facebook is extremely low since posts on Facebook goes down as new post comes. Carbuds, will guarantee to find driver or passenger if there are people who is going to same place with each other. Passengers or drivers can trust each other since the system will be closed to non-students.

3.2. FUNCTIONAL REQUIREMENTS

- The system should hold the users data(schedule, personal details, ratings) .

- The system should be defining the hubs, or sub networks where users should be grouped up with the other users sharing the similar paths, according to users wishes.
- The system should request an authentication process before initiating with users account.
- The system should provide a further authentication service to sign in before the users use the system, if they log off.
- The system should provide search and matching tools to search from the all users using the system.
- The system should allow users to rate and give a written feedback to other users.
- The system matches a driver with a hitchhiker and give both appropriate notifications.
- Matched users can talk to each other via using built-in messaging system.
- Both hitchhikers and drivers can specify their preferences for the trip in order to increase the satisfaction of both parties.
- Users will be choose their location and destination point using Google Maps API.
- The system offers each user a set of drivers/hitchhikers that generated according to their preferences, by swiping right or left users can show their willingness or not.
- All users will be authenticated with their school email addresses in order to ensure all users are university students.
- The system will give brief information about the traffic rules inside the university's campus in order to assure that driver is aware of the restrictions in the respective university.

3.3. NONFUNCTIONAL REQUIREMENTS

Usability

- The interface should be user-friendly, simple and straightforward.
- The user interface should be intractable.
- Internet connection is required for using the application.

Efficiency

- The matching algorithm should be fast and efficient in order to give responses to multiple users.
- The route finder algorithm should be fast for calculating routes for users.

Extensibility

- The system should allow addition of future feature developments and should be up to date for each new Android.

Reliability

- The matching of the users should be precise with their preferences.

Compatibility

- The mobile application must be compatible with the most of the android phones in the market.

3.4. PSEUDO REQUIREMENTS

Implementation Constraints

- The application will be a mobile application.
- Github will be used for version control.
- The application will be implemented on both android and iOS.
- For android application Java will be used and for iOS swift will be used for implementation.

Economic Constraints

- The application will not charge users for its service
- Users will not charge each other for sharing a ride.
- The application need a server for both database solutions and logic solutions.
- The application show ads to users in some pages for expenses.

Ethical Constraints

- None of the personal data of the users will be used or shared with third parties.
- The matching algorithm should not be affected with the users' race, ethnicity and/or social status.
- The statistics that extracted from the application will not be sold or shared with third parties.

Time Constraints

- The project should be completed before December 2018.

Professional and Ethical Issues

- The application relies on the users' preferences in order to match them for a shared ride. However, system should be aware in order to prevent any actions related the race, ethnicity, social status and gender discrimination but we offer our users to choose their matched user's gender in order to increase satisfaction due to social culture in Turkey [1].

3.5. SYSTEM MODELS

3.5.1. SCENARIOS

3.5.1.1. USER SIGNUP

Actors: User

Entry Conditions:

- User opens the application.
- User chooses 'Create a User' to register.

Exit Conditions:

- User successfully registers as a new user.
- User closes the application.

Main Flow of Events :

- User opens the application.

- CarBuds checks whether the user has logged in before or not. If already logged in, system opens the landing page.
- User presses ‘Create a User’ button.
- User enters his/her university e-mail and phone number for verifying whether he/she is an ongoing student in the specified university and has a working cell phone.
- User enters his/her credentials.
- User clicks ‘Proceed’ and enters the validation codes which have been sent by CarBuds to the university email and to his/her phone number after a successful validation.
- If verification fails, system raises an error message to the user.
- If verification is successful, User continues to the set profile page.

3.5.1.2. USER LOGIN

Actors: User

Entry Conditions:

- User opens the application.
- User chooses to login or already logged in.

Exit Conditions:

- User successfully logins.
- User closes the application.

Main Flow of Events :

- CarBuds checks whether the user has already logged in before. If already logged in, user lands on the landing page.
- If user has not logged in before, user chooses ‘Login’ option.
- User enters his/her credentials.
- CarBuds checks the authenticity of his/her credentials and if the credentials are correct, user successfully logs in to the application.

3.5.1.3. USER LANDING PAGE

Actors: User

Entry Conditions:

- User opens the application.
- User manually logs in or should already logged in.

Exit Conditions:

- User changes the page.
- User chooses the purpose.
- User closes the application.

Main Flow of Events :

- After a successful login, user welcomed with landing page.
- User decides whether he/she uses as hitchhiker or driver and user will be directed to the appropriate profile set up page if he/she has never set a profile as his/her preference before.
- If user has already chosen, CarBuds redirect the user to the finding match page.

3.5.1.4. USER SETUP HITCHHIKER PROFILE

Actors: User

Entry Conditions:

- User opens the application.
- User opens the profile page.
- User chooses 'Edit Profile' to set up his/her profile information.
- User chooses him/herself as hitchhiker or driver and no active profiles set up yet.

Exit Conditions:

- User leaves the profile page.
- User closes the application.

Main Flow of Events:

- If user already logged in, CarBuds provides a profile setup page.

- User enters his/her personal information either manually or can link his/her Facebook account to retrieve personal information.
- User can define his/her preferences for the trips and preferences about hitchhikers or drivers.
- Whenever user leaves the profile page last changes/updates will be saved to the database.

3.5.1.5. USER SETUP DRIVER PROFILE

Actors: User

Entry Conditions:

- User opens the application.
- User opens the profile page.
- User chooses 'Edit Profile' to set up his/her profile information.
- User chooses him/herself as hitchhiker or driver and no active profiles set up yet.

Exit Conditions:

- User leaves the profile page.
- User closes the application.

Main Flow of Events:

- If user already logged in, CarBuds provides a profile setup page.
- User enters his/her personal informations either manually or can link his/her Facebook account to retrieve personal information.
- User can define his/her preferences for the trips and preferences about hitchhikers or drivers.
- Whenever user leaves the profile page last changes/updates will be saved to the database.

3.5.1.6. USER MATCHMAKING PAGE

Actors: User

Entry Conditions:

- User opens the application.
- User manually logs in or should already logged in.
- User should already decided his/her role and set up a profile with preferences.
- User should already decided where he/she is and what is his/her destination.

Exit Conditions:

- User changes the page.
- User closes the application.

Main Flow of Events :

- CarBuds will found the appropriate buddy candidates for the user by filtering with its preferences in order to show the available drivers/hitchhikers that goes the same location or a location which is close enough for both parties.
- User chooses between the shown buddies by swiping left as a decline action while swiping right as accept action.
- CarBuds will save your accepted buddy candidates in order to matchmaking possible.

3.5.1.7. USER SET TRIP AS DRIVER

Actors: User

Entry Conditions:

- User opens the application.
- User should already be logged in.
- User should already decided his/her role as driver and set up a profile with preferences.

Exit Conditions:

- User changes the page.
- User closes the application.

Main Flow of Events :

- User chooses his destination and his/her point of departure from the map API of Google Maps.
- User specifies available seats for the hitchhikers, departure time as well.
- User presses 'Create Trip' button to finish the trip setup.
- CarBuds will take the credentials and save the trip information to the database.

3.5.1.8. USER SET TRIP AS HITCHHIKER

Actors: User

Entry Conditions:

- User opens the application.
- User should already be logged in.
- User should already decided his/her role as hitchhiker and set up a profile with preferences.

Exit Conditions:

- User changes the page.
- User closes the application.

Main Flow of Events :

- User chooses his targeted destination, potential alternative stops to make progress as hitchhiker and his/her point of departure from the map API of Google Maps.
- User specifies available departure time.
- User presses 'Create Trip' button to finish the trip setup.
- CarBuds will take the credentials and save the trip information to the database.

3.5.1.9. USER NOTIFICATION

Actors: User

Entry Conditions:

- A notification should have been generated from the server.
- User should already be logged in before.

Exit Conditions:

- User closes the application.

Main Flow of Events :

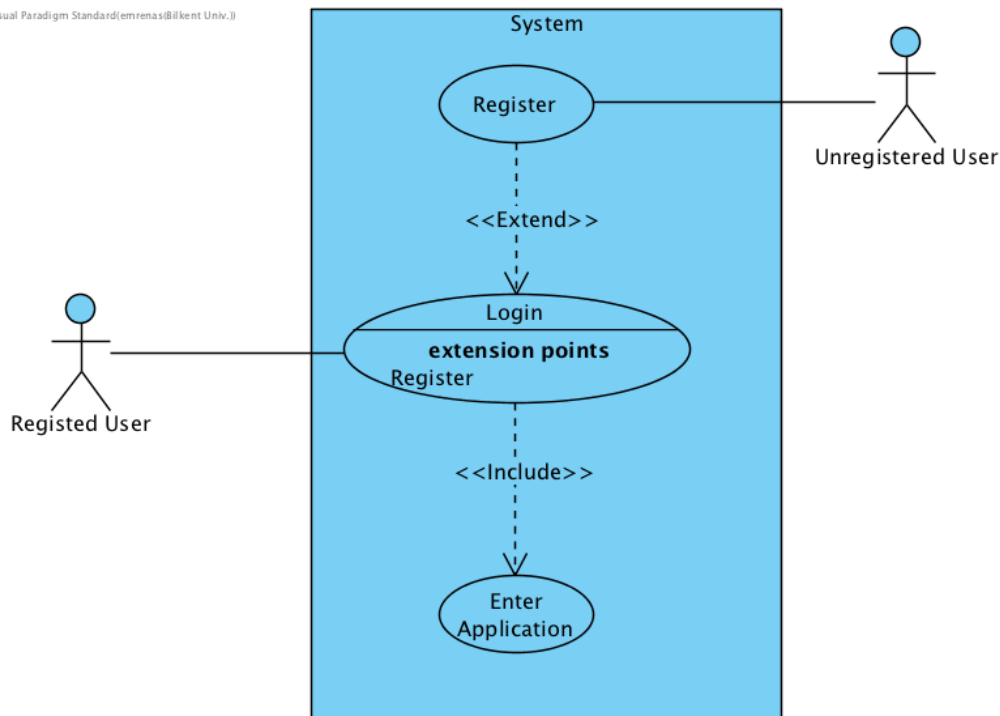
- CarBuds generates a notification (new message, new match found) and sends the appropriate notification to the corresponding user.
- User will be able to see the notification from the outside of the application and clicks it to open the application.
- Application will be opened with the appropriate page about the notification.

3.5.2. USE CASE MODEL

Use case diagrams of CarBuds application are described in this section

3.5.2.1. REGISTER & LOGIN USE CASE

Visual Paradigm Standard(emrenasi@ilkent Univ.)



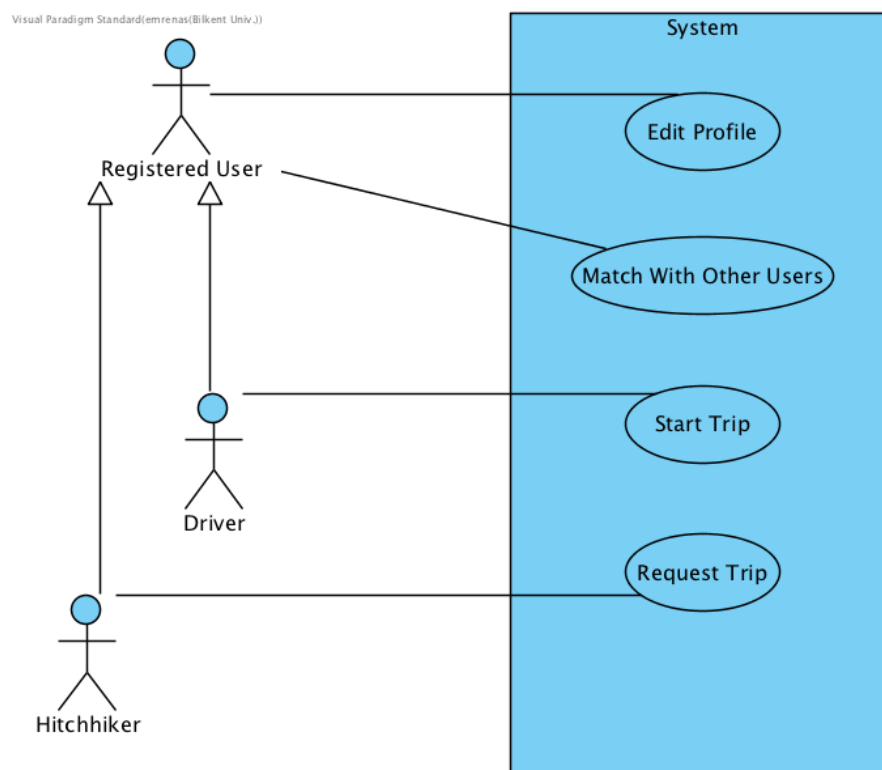
Description:

This use case show relation of entering the application of registered users and unregistered users.

There is no way unregistered user to enter application. Since unregistered user cannot use application they must register through application in order to use the app. This is necessary for understanding if person who is trying to enter system is student.

After an unregistered user registered to system they automatically become registered user. When a registered user wants to enter the system, they must provide their school email address and their password for authentication. If the authentication is successful they can use the application.

3.5.2.2. REGISTERED USER SYSTEM USE CASE



Description:

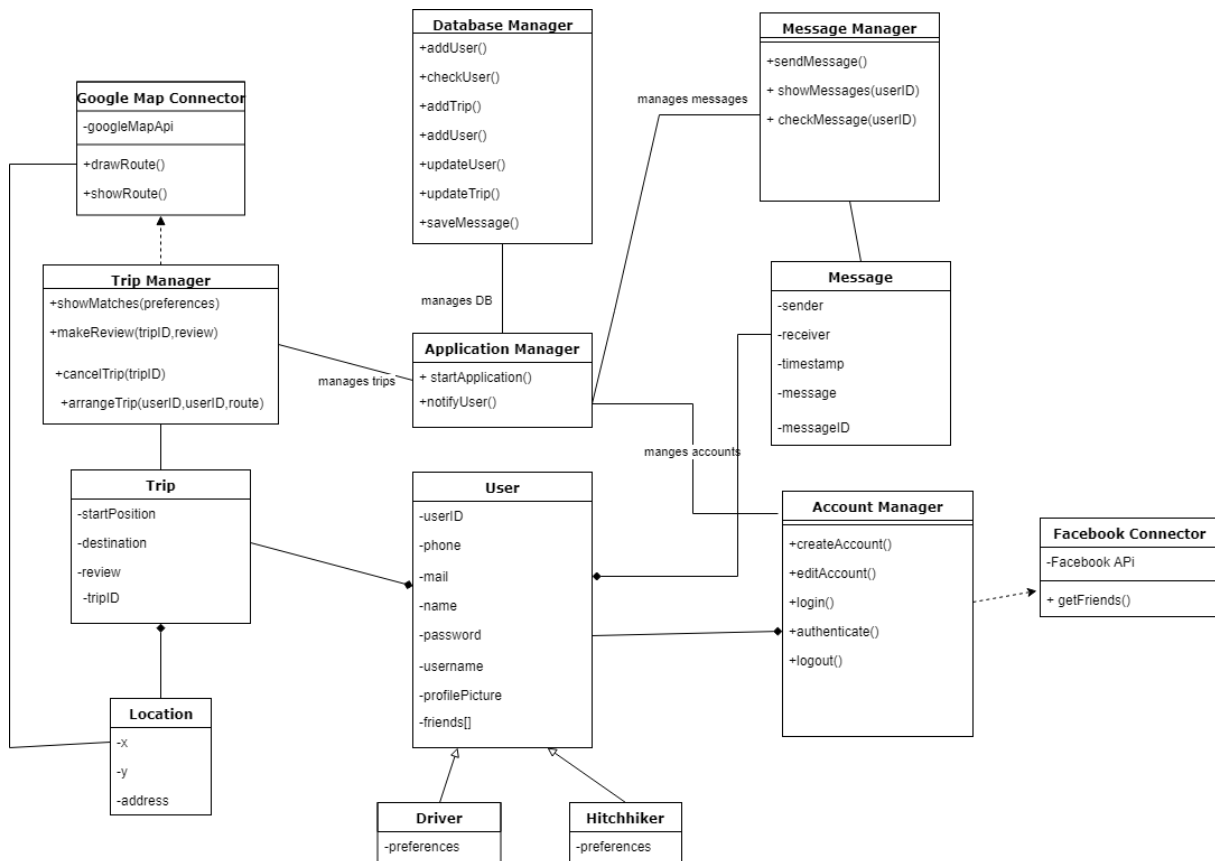
This system model show what a registered user can do in the application. A registered user divide into two user model Driver and Hitchhiker. Both of these users can edit their profile for mentioning their preferences like if they accept to travel with a pet or their road trip mate's gender. They can also change their password or add additional information like their bio or music preferences. Both of them can match with each other either as a driver or hitchhiker.

Driver actor can start a trip which mean they are looking for a hitchhiker to match with same route and similar preferences.

Hitchhiker actor can request a trip and search for a driver with same route as theirs.

3.5.1. OBJECT AND CLASS MODEL

Object and Class model is demonstrated in this section.



4. **APPLICATIONMANAGER** : This class will manage the application. It will create main user interface and will create connections between other manager classes
4. **MessageManager**: This class will manages sending and receiving messages. It will create messaging user interface, create and send messages and will notify users for incoming messages.

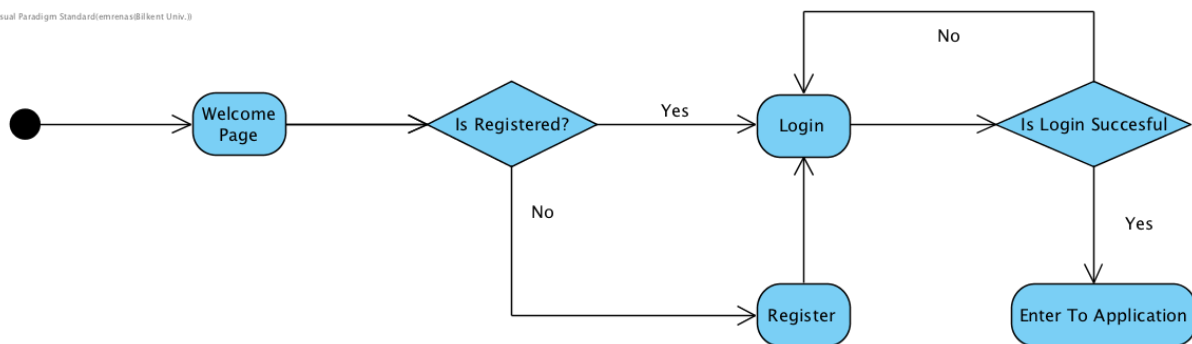
4. **TripManager:** This class will be responsible for managing trips. It will create matching user interface, creates and updates trips.
4. **AccountManager:** This class is responsible for managing accounts. It will create and update accounts. Also, it will responsible for authentication and login processes.
4. **DatabaseManager:** This class will be responsible for database connection of the project. It will store all accouts, users and trips.
4. **Message:** This class is the message object. It will keeps sender, receiver, message and messageID.
4. **User:** This class keeps common user information. Hence, there are two type users this class is a parent class for that user types.
4. **Driver:** This is a child class of User. It is a type of user and it stores driver information.
4. **Hitchhiker:** This is a child class of User. It is a type of user and it stores hitchhiker's information.
4. **Trip:** This class keeps information about trip. It stores startPosition, destination, review and tripID.
4. **Location:** This class basically stores a location with address.
4. **Facebook Connector:** This class will be responsible for creating connection account manager and facebook. With facebook connection account manager will find friends of users.
4. **GoogleMapConnector:** This class will responsible for creating connection with Google Maps API. Gooogle Maps API will be used for selecting starting location and destination. Also it will create route and show it to users.

3.5.2. DYNAMIC MODELS

Activity and sequence diagrams and graphical user interface mockups are presented in this section

3.5.2.1. REGISTER AND LOG IN ACTIVITY DIAGRAM

Visual Paradigm Standard (emrenas@kent Univ.)

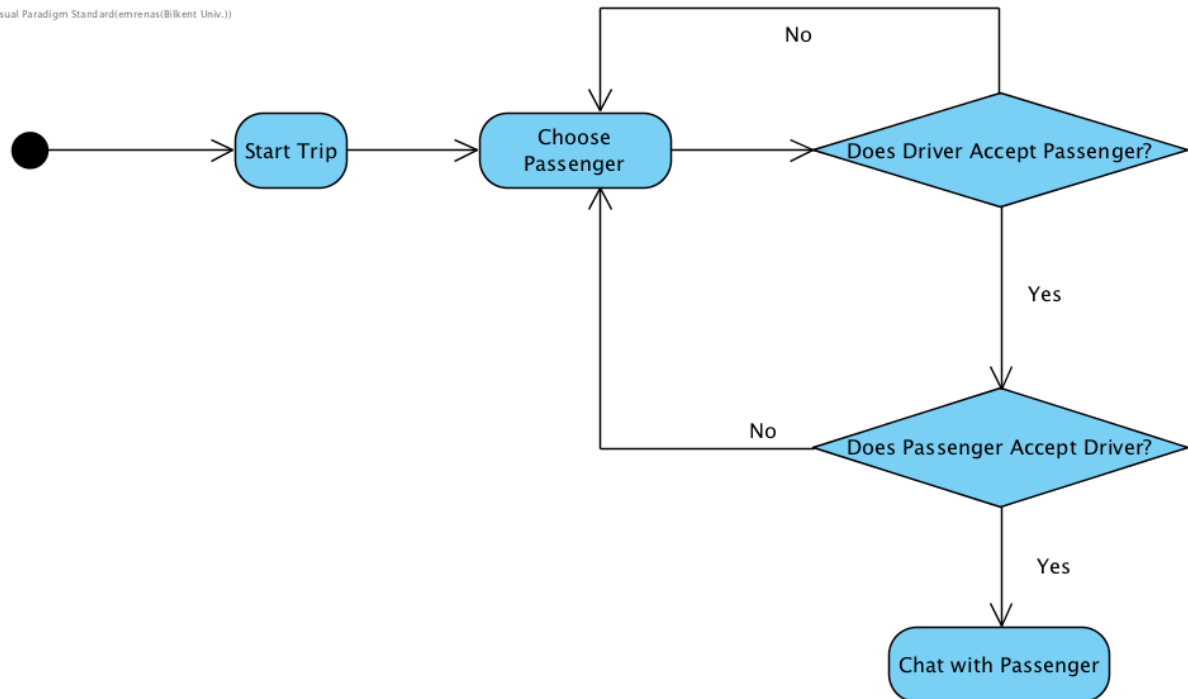


Description:

The activity diagram shown above is register and log in activity diagram. When a person opens the application, a welcome page will welcome them. Since in order to enter to application person must be registered. If he/she is not registered the user will be directed to register activity where they create a new account. After they register they will be directed to login activity. Registered users will bypass the register activity and directly go to login activity. When they enter their e-mail and password system will check if the given information by user is valid. If they are not valid they will go back to login page. After they successfully logged in they will enter to application.

3.5.2.2. DRIVER GENERAL USAGE ACTIVITY DIAGRAM

Visual Paradigm Standard(emrenas(Bilkent Univ.))

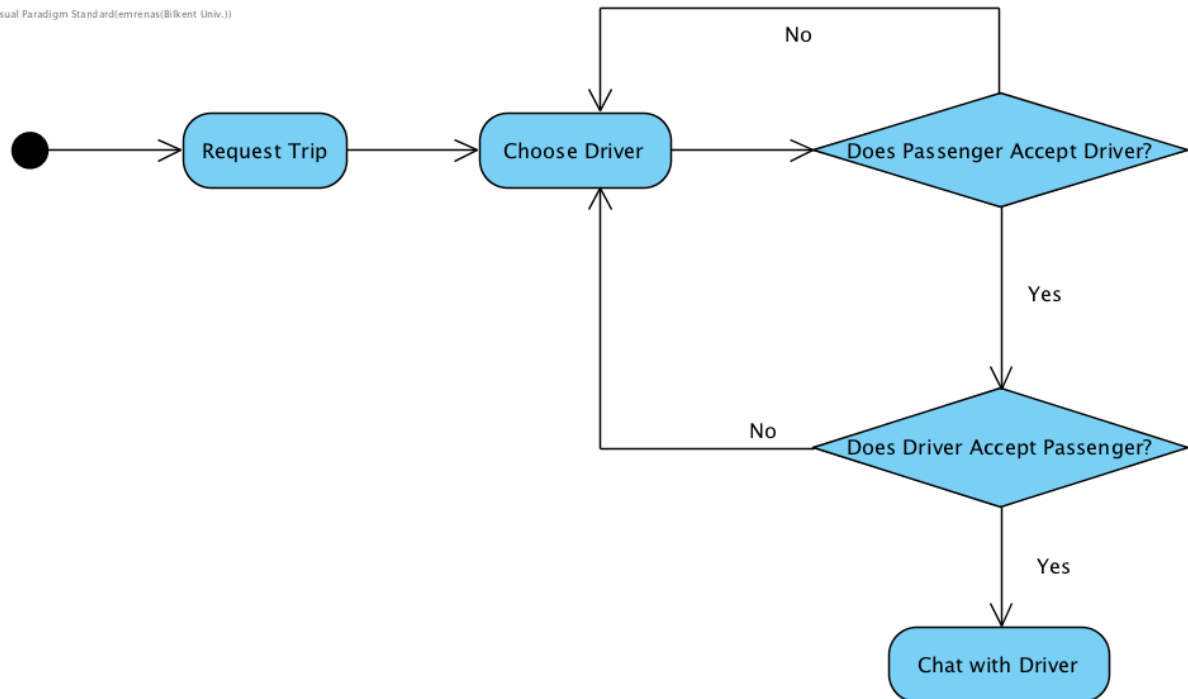


Description:

The activity diagram shown above is general activity diagram of a driver. When a driver starts a trip by entering the route possible candidates of hitchhikers will be shown to him/her by their preferences and by distance proximity. If they do not choose the shown candidate they can always find new passenger. If they accept a passenger and the passenger accept the driver they can start chatting for more information. If passenger do not accept the driver they need to find new passenger so they will be directed to choose passenger activity.

3.5.2.3. PASSENGER GENERAL USAGE ACTIVITY DIAGRAM

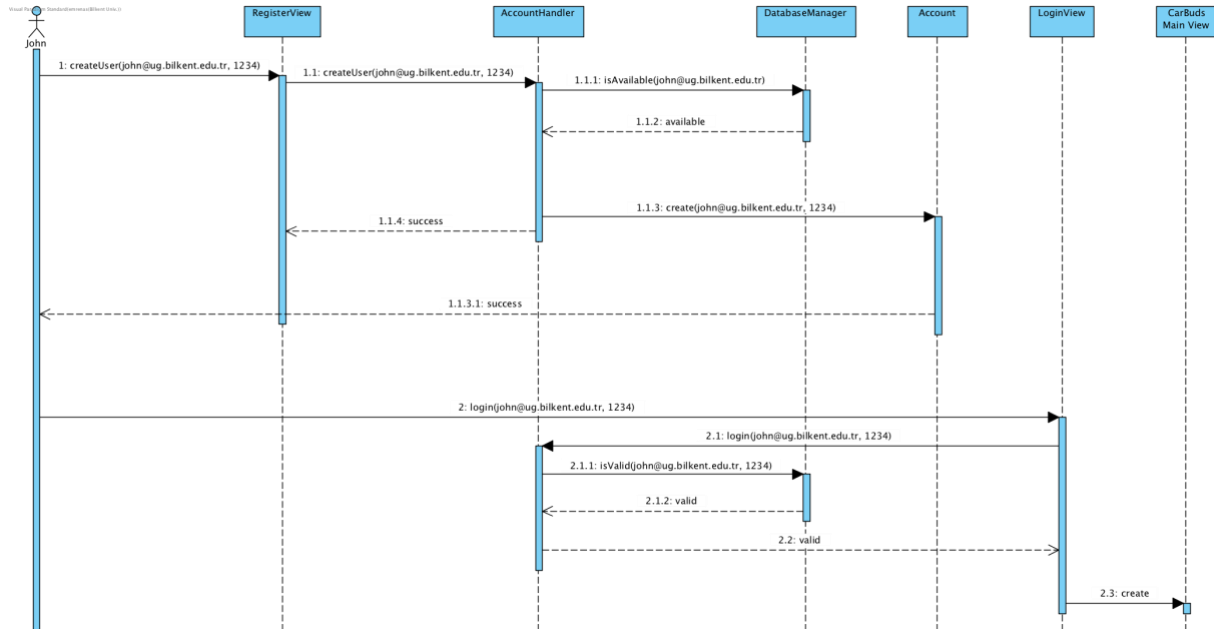
Visual Paradigm Standard(emrenas(Bilkent Univ.))



Description:

The activity diagram shown above is general activity diagram of a passenger. This activity diagram is similar with Driver General Usage Activity Diagram. A passenger requests a trip instead of starting a trip. When they request a trip, possible candidates will be shown to them by their preferences and by distance proximity. When they do not accept a driver, they will be directed to choose driver activity. If they accept the driver and the driver accept the passenger they can start chatting but if driver, which passenger accepted, does not accept the passenger, passenger will be directed to choose driver activity again.

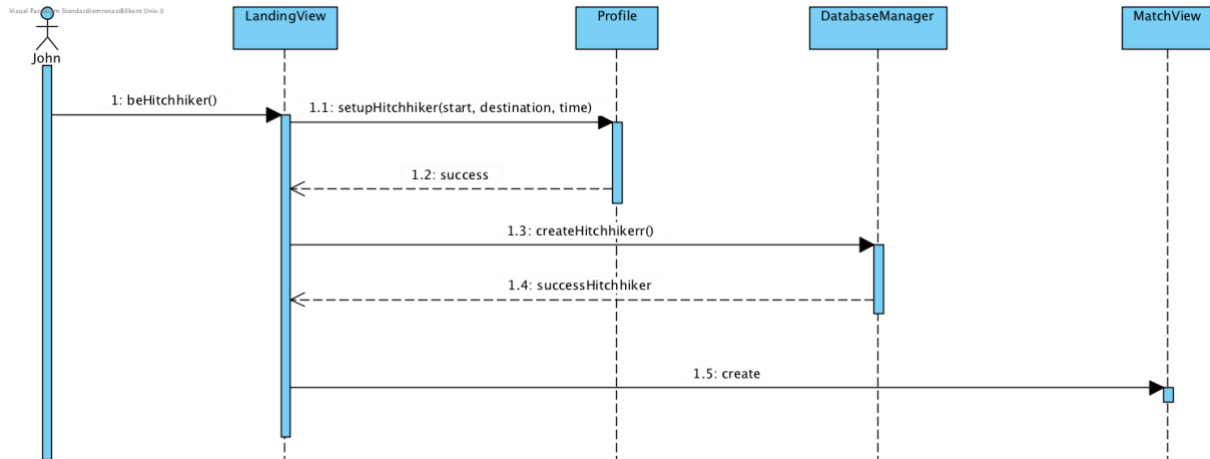
3.5.2.4. REGISTER AND LOGIN SEQUENCE DIAGRAM



Description:

In this diagram, user John try to create a new account from RegisterView. AccountHandler call method createUser, then AccountHandler send message to DatabaseManager to check if given email address is available. When DatabaseManager return success message a new account object created. After that John goes to loginView and try to login by his email and password by invoking login method in AccountHandler. AccountHandler sends message to DatabaseManager to check if given informations are correct. When AccountHandler receives valid signal John successfully goes to CarbudsMainView.

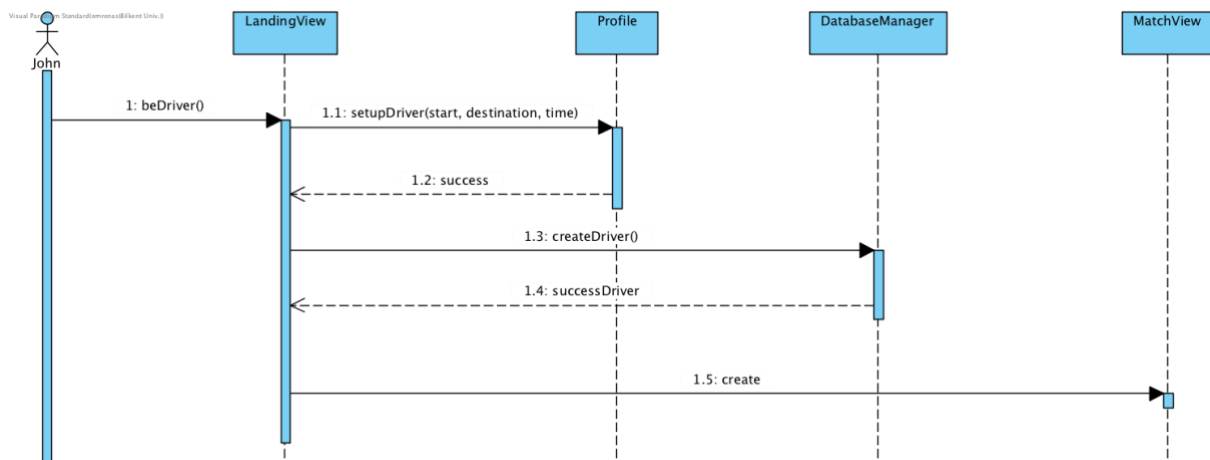
3.5.2.5. BEING HITCHHIKER SEQUENCE DIAGRAM



Description:

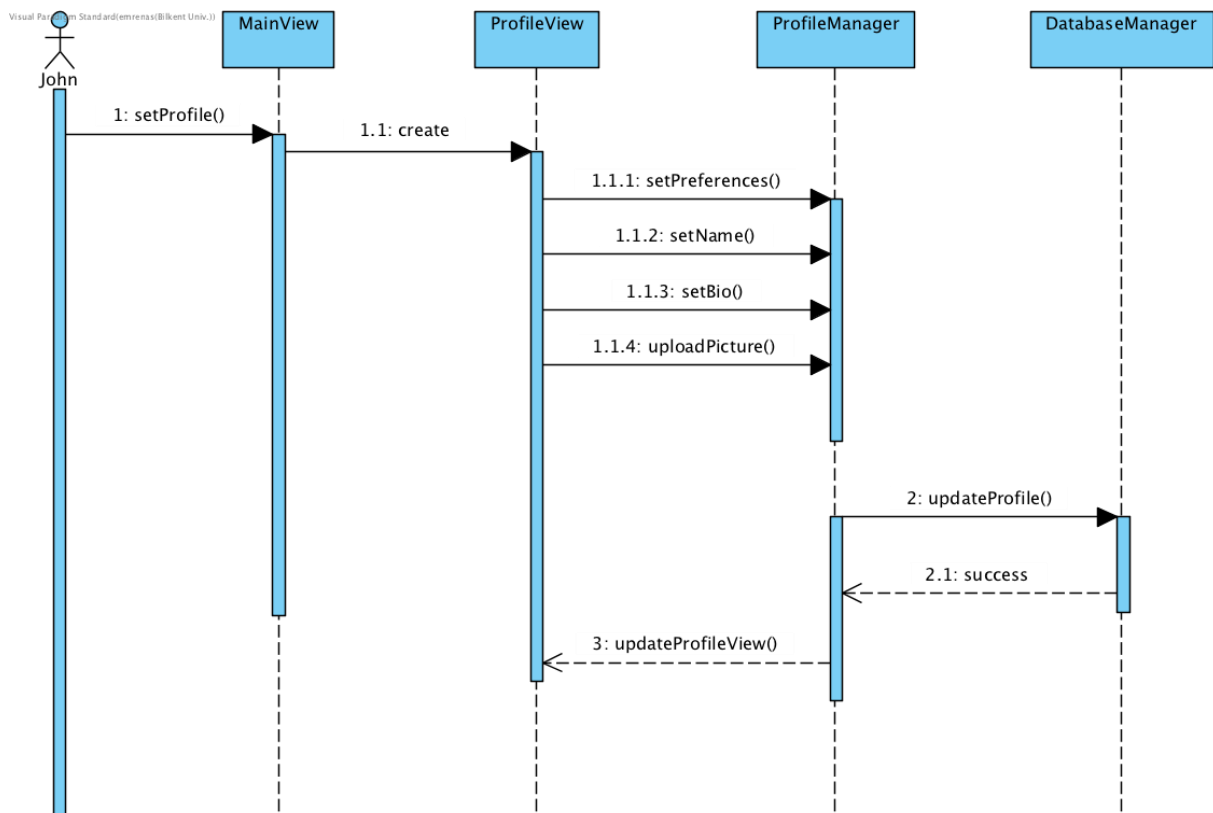
In this diagram, user John wants to be a hitchhiker. If he successfully logged in to application a landing page will welcome him. He will decide whether he wants to be driver or hitchhiker. When he chooses he wants to be a hitchhiker he will create hitchhiker profile by invoking method `setupHitchhiker(start, destination, time)`. When success message returns from Profile a database entry will be created by DatabaseManager. If success message returns from DatabaseManager he will be directed to MatchView

3.5.2.6. BEING DRIVER SEQUENCE DIAGRAM



Description:

In this diagram user John wants to be a driver. After he successfully entered the application he goes to LandingView in order to choose whether he wants to be hitchhiker or driver. After John decide he wants to be driver, he setup his profile as a driver. If success message returns from profile an entry is created in database by DatabaseManager. After DatabaseManager return success user John will be directed to MatchView to see possible matches.

3.5.2.7. UPDATE PROFILE**Description:**

In this diagram user John update his profile. When he is in main view he decide to change his profile or directed after he decided to be hitchhiker or driver. MainView create ProfileView for John where he can update his profile. He may change his/her preferences,

name, biography or uploadPicture. After he successfully done his editing ProfileManager will be invoked according to what John changed. ProfileManager will invoke DatabaseManager by calling method updateProfile(). If success message returns from DatabaseManager his profileView will be updated with new data.

3.5.2.8. USER INTERFACE –SCREEN MOCK-UPS



Figure 1: First landing page of the CarBuds application.

This is the first landing page of the application. New users will be welcomed with this page. If they have an active account, they can login or click sign up in order to create a new account.

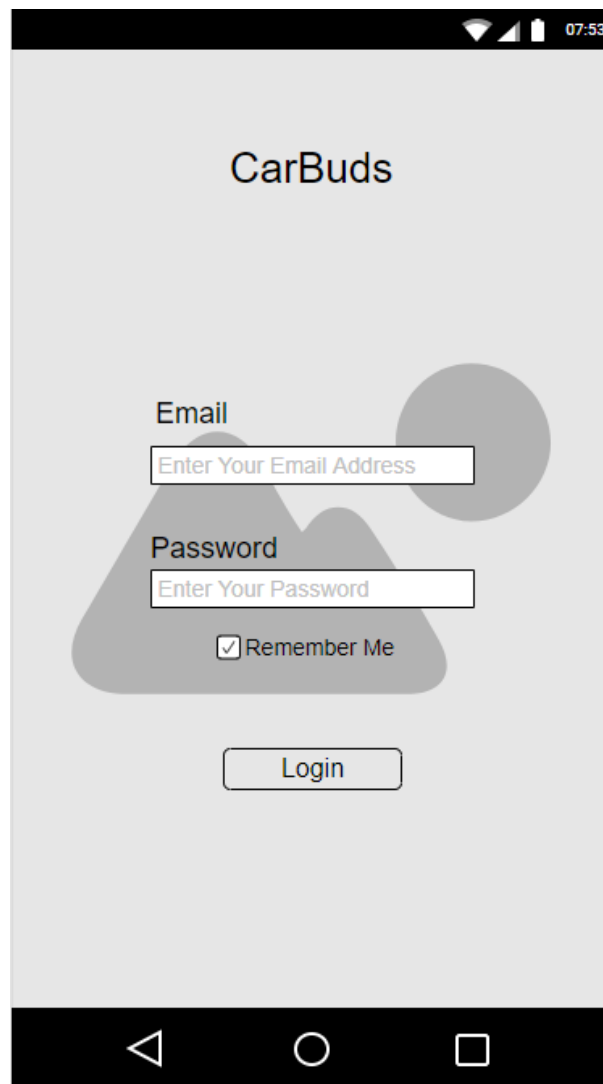
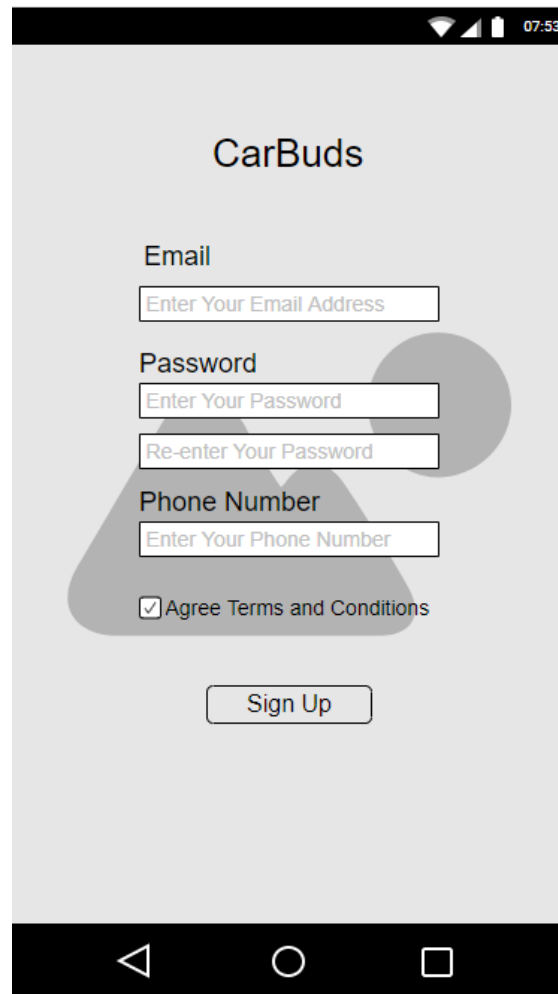


Figure 2: Login page

If user has an active account, he/she can login to his/her account from this page. Also can check the 'Remember Me' button in order to pass this action after a successful login.



CarBuds

Email

Enter Your Email Address

Password

Enter Your Password

Re-enter Your Password

Phone Number

Enter Your Phone Number

☒ Agree Terms and Conditions

Sign Up

Figure 3: Sign up page

A new user can create a new account from this page. User need to enter his/her university e-mail in order to get validated as a university student and user needed to provide his/her personal phone number in order to receive account validation code.

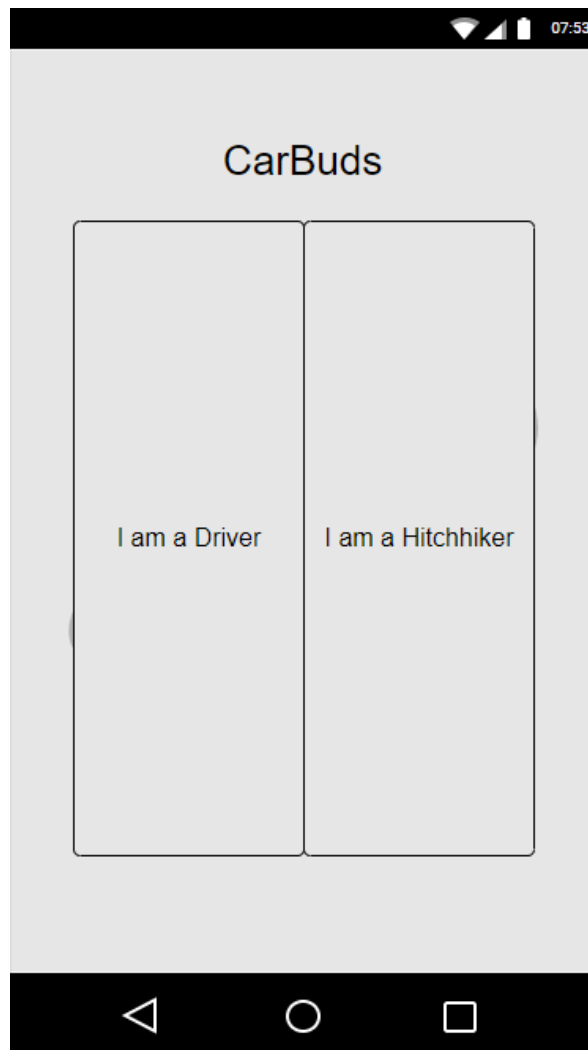
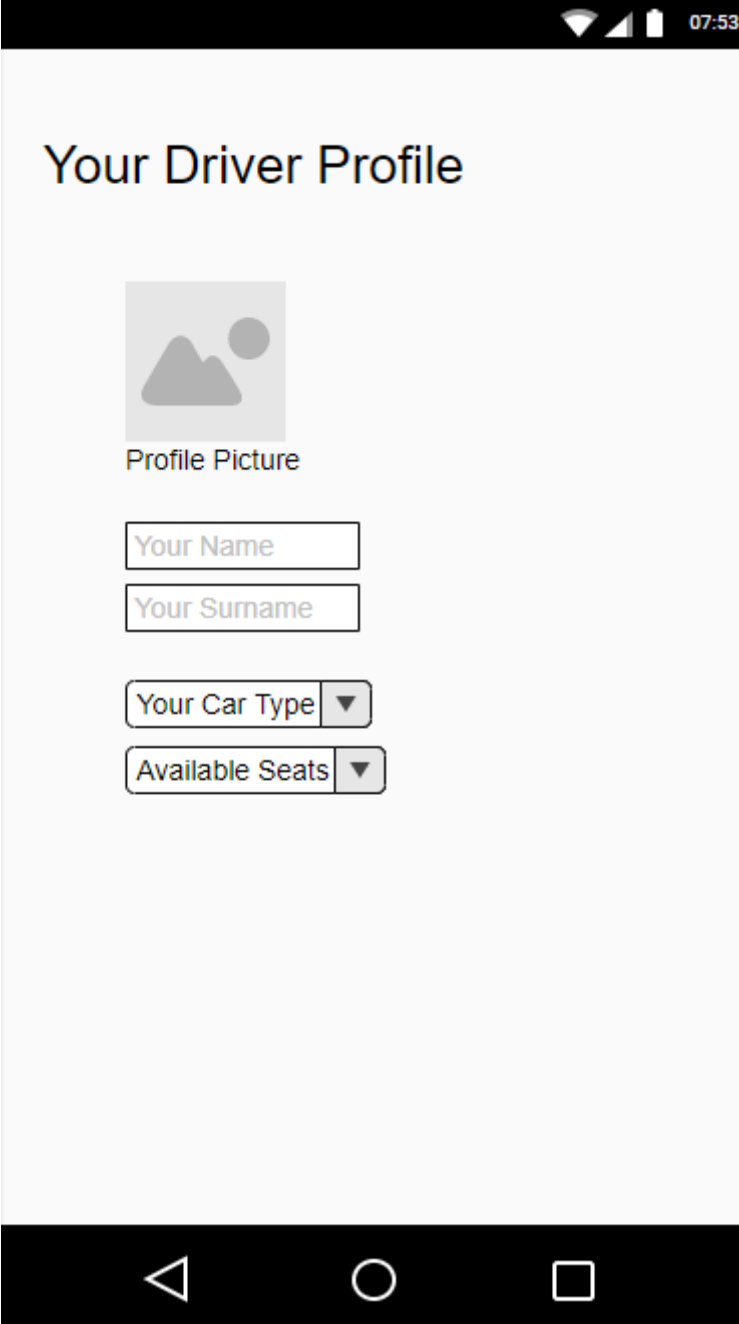


Figure 4: Role Selection Page

A new user asked by his/her role for the first time. User can decide whether his/her role as hitchhiker or driver. After the selection, application will direct user to the appropriate role's profile page in order to set up a profile.



07:53

Your Driver Profile

Profile Picture

Your Name

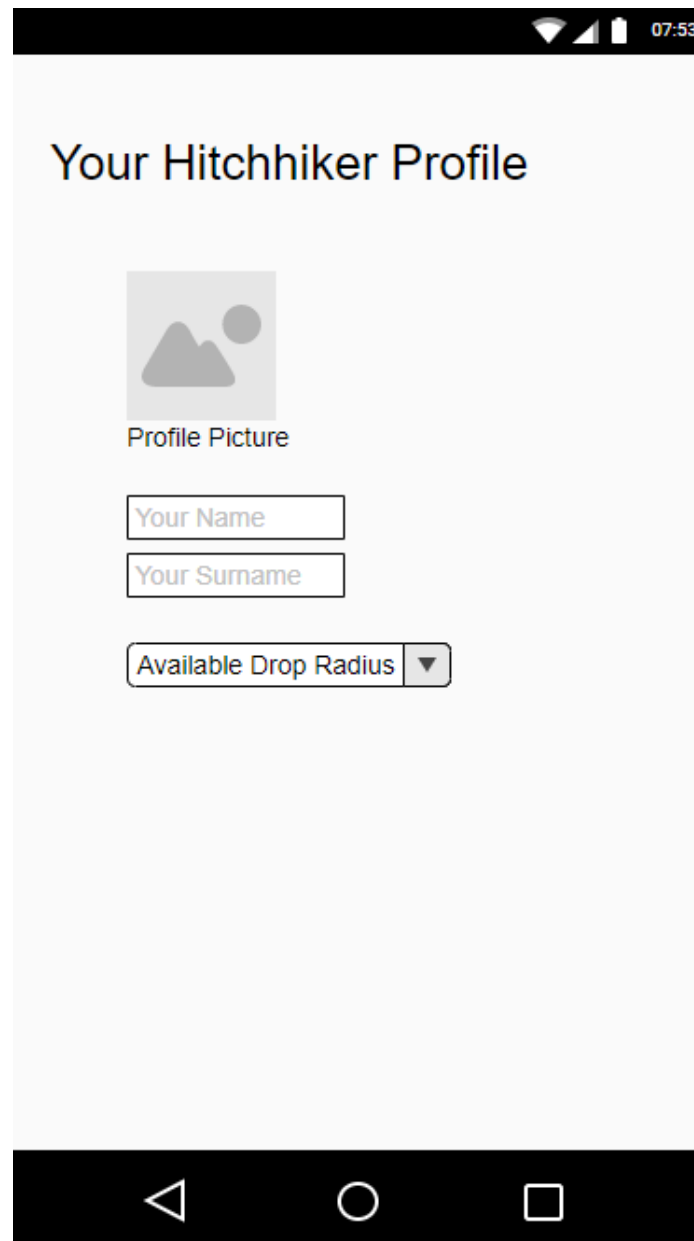
Your Surname

Your Car Type ▼

Available Seats ▼

Figure 5: Driver Profile Page

User can edit his/her preferences as driver and his/her information that other users can see.



07:53

Your Hitchhiker Profile

Profile Picture

Your Name

Your Surname

Available Drop Radius ▼

Figure 6: Hitchhiker Profile Page

User can edit his/her preferences as hitchhiker and his/her information that other users can see.

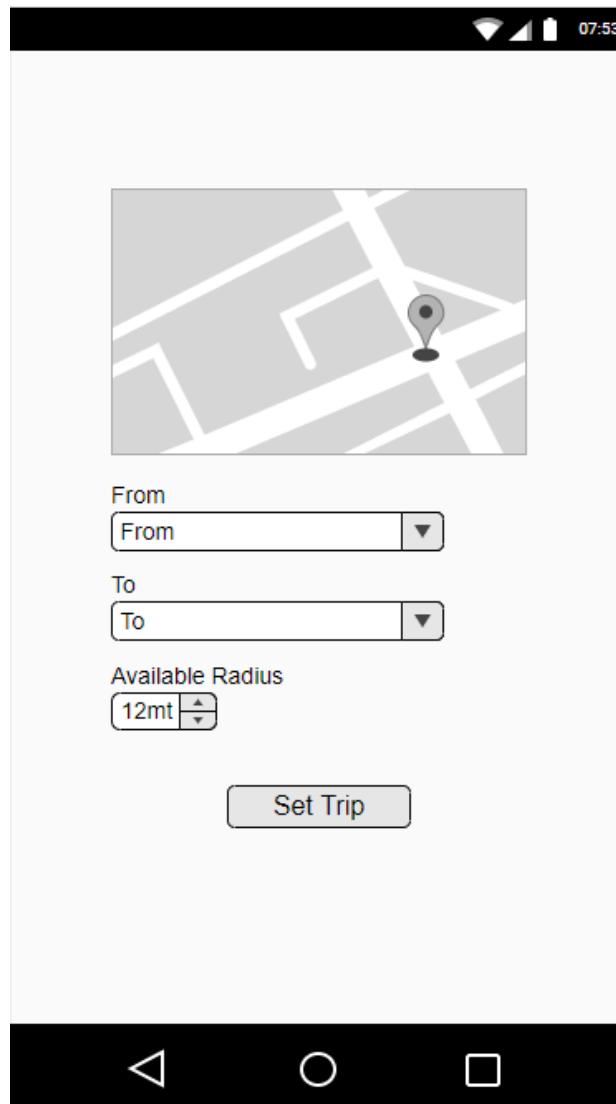


Figure 7: Trip Setup Page

Users of both roles can setup a trip from this page by choosing their departure location and their target location. Also both users can choose their available drop by radius.

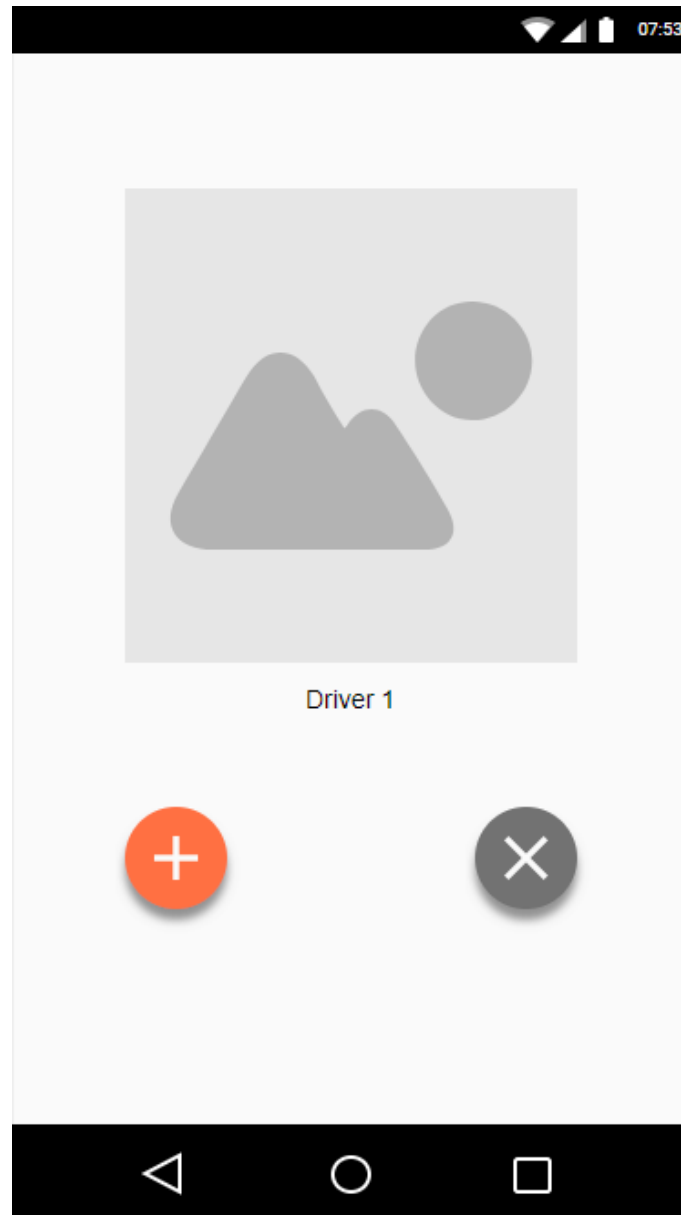


Figure 8: MatchMaking Decision Page

After setting up a trip, users can see this page until they cancel their trips. Users can see available candidates in this page and they can make their decisions either doing a swipe motion or clicking the appropriate buttons on the screen.

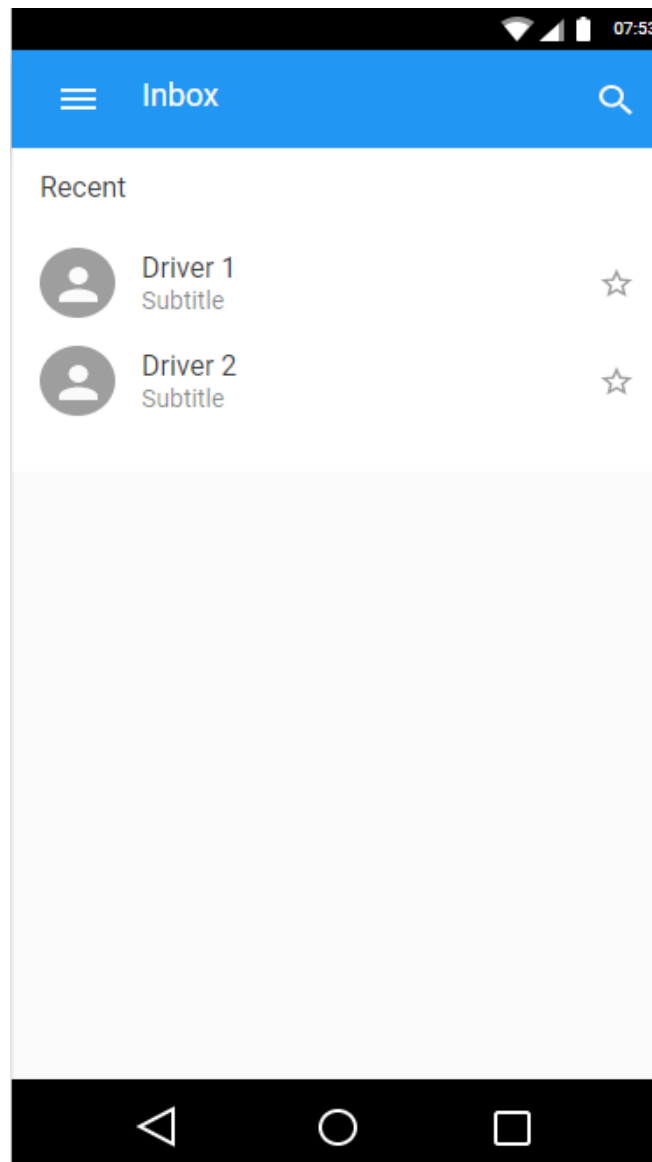


Figure 9: User Inbox Page

Users can see their matched car buddies from this page and able to see new messages that came from other users or they can choose a chat in order to send a message.

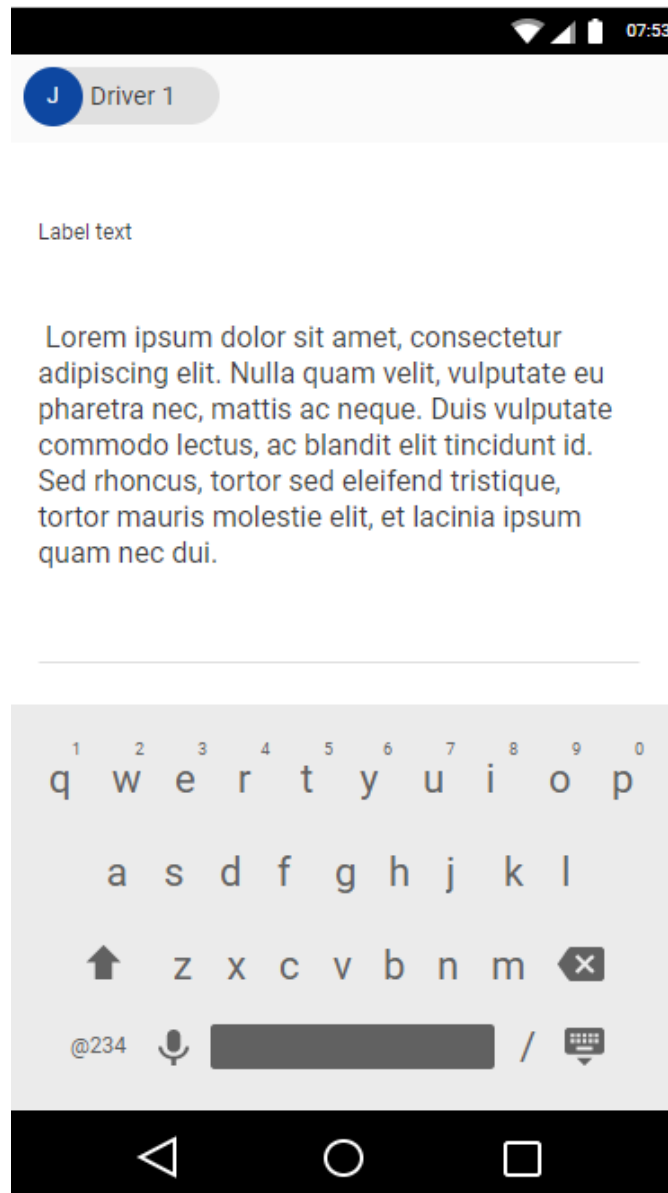


Figure 10: In-App Chat Page

Users can send text messages to their car buddies from this page.

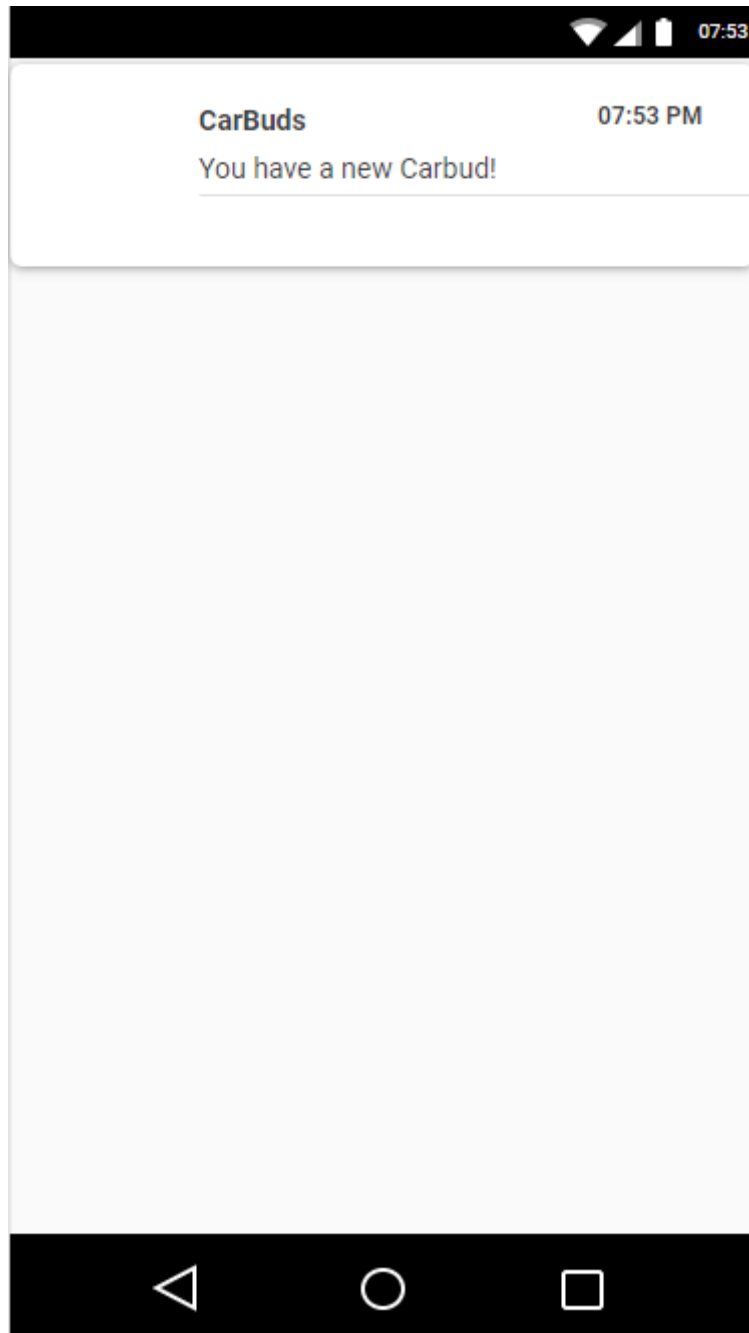


Figure 11: Application Notification

Users will see this notification both in-app or outside the application.

4. REFERENCES

[1] lyft.com. (2018). *HomePage*. [online] Available at: <https://www.lyft.com/line>
[Accessed 19 Mar. 2018].

[2] blablacar.com. (2018). *HomePage*. [online] Available at: <https://blablacar.com>
[Accessed 19 Mar. 2018].