



# **MyTaxiService**

## **Requirements Analysis and Specification Documents**

**Software Engineering 2**

**Authors:**

**Hasan Aliyev  
Madat Mustafayev**

**Milan —2015**  
**Keynote**

# Contents

<b>1 Introduction</b>	<b>4</b>
1.1 Description	4
1.2 Purpose	4
1.3 Scope	4
1.4 Definition	5
1.5 References	5
1.6 Overview	5
<b>2 Overall description</b>	<b>6</b>
2.1 Product Perspective	6
2.1.1 System Interfaces	6
2.1.2 User Interfaces	6
2.1.3 Hardware Interfaces	6
2.1.4 Software Interfaces	7
2.1.5 Memory	7
2.2 Product Functions	7
2.3 Constraints	10
2.4 Assumptions	10
<b>3 Requirements</b>	<b>11</b>
3.1 External Interface Requirements	11
3.1.1 Registration screen	11
3.1.2 Login screen	12
3.1.3 Main menu screen	13
3.1.4 Edit Profile screen	14
3.1.5 Order a Taxi screen	15
3.1.6 Summary screen screen	16



# Contents

3.1.7 Payment screen . . . . .	12
3.1.8 Trip confirmation screen . . . . .	12
3.2 Functional Requirements . . . . .	19
3.2.1 Signup . . . . .	19
3.2.2 Login . . . . .	19
3.2.3 Order a Taxi . . . . .	20
3.2.4 View and Edit profile . . . . .	21
3.2.5 Taxi driver action . . . . .	22
3.2.6 Administrator . . . . .	23
3.2.7 Notify Users . . . . .	24
3.3 Sequence diagrams . . . . .	25
3.4 StateChart diagrams . . . . .	27
3.5 Non-functional Requirements . . . . .	28
3.5.1 Reliability . . . . .	28
3.5.2 Availability . . . . .	28
3.5.3 Security . . . . .	28
3.5.4 Maintainability . . . . .	28
3.5.5 Portability . . . . .	28
4 Data Model and Description . . . . .	29
4.1 User . . . . .	29
4.2 Order a Taxi . . . . .	30
4.3 View and edit Profile . . . . .	30
4.4 Taxi Queue. . . . .	30
5 Conclusion . . . . .	32
6 Alloy . . . . .	33

# 1. Introduction

## 1.1 Description

MyTaxiService is a platform where users or customers are able to order Taxi online through mobile applications or web interface and also manage their trips beforehand, calculate the estimated cost, find the best fare and gives feedback for the taxi drivers. Required information about pickup time, place and last destination should be provided while creation of the online ordering. After clicking to «order taxi» on the main menu of application, the platform offers to fill some data about trip. For instance, pickup location, destination location, share ability, urgency and etc. After filling all necessary field and pressing «order a taxi» the data will send to the server and also will be appeared on the screen of nearby taxi driver. Afterwards, taxi driver has to accept order or cancel it. In any case User will notified by instant notifications about the status of order.

This platform is relevant for users who look for to schedule their trips in advance and minimize the time finding a taxi and also do cost saving by estimating fare.

## 1.2 Purpose

The purpose of the Requirements Analysis and Specification Document is to clearly state every requirement, functions and features of the product. It is important to predict and find out about our expectation to how to use this product..RASD will provide a detailed overview of our soft, its parameters and goals, being useful both for developers and customers. For that reason, this document is written for mainly developers of the system to make sure that all team members has the same vision of requirements.

## 1.3 Scope

The project named **myTaxiService** which our team intend to develop is a mobile and web application that aims to make easy access of passengers to the taxi services and provide fair control of taxi queues. The users of our platform can easily order a taxi without losing a time in street finding taxi, explaining the exact address to the driver. As compared to other way of finding transport, our system is simply, interactive and very useful for those who care about their time. Basically, our software is capable of dealing with more cases and inputs. For example, Users could select the type of car (business or custom), choose the pick up time and also see immediately the fare and duration of trip.



# 1. Introduction

## 1.4 Definition

**Taxi Service:** The Government profit organisation which is providing Taxi service in Milano.

- **myTaxiService:** A software for the Taxi service company which intended simplify the access of passengers to the service through Hi-Tech(mobile phones,laptops,PCs) and also provide fair control of taxi queues.
- **Order a taxi:** Abbreviation in our systems.Responsible to taking request from customers and carry up to the server. We can call it also main transaction of our platform.
- **Users:** There are 3 type of users in our platform. Firstly, Our customers, let's call them «**Client**». Client is the user who intended to use our product. Secondly, **Taxi drivers**. They are drivers of taxi service. Finally, **Administrator** of the system who cares about clients and also drivers, maintaining and regulate the system.

## 1.5 References

- Start UML Guide, retrieved from [http://staruml.sourceforge.net/docs/user-guide\(en\)/toc.html](http://staruml.sourceforge.net/docs/user-guide(en)/toc.html)
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specification. IEEE Computer Society, 1998

## 1.6 Overview

There are mainly four more parts of the document. The first part after the Introduction part is the Overall Description. This section describes the general factors that affect our product and its requirements. The Specific Requirements part contains all of the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements and testers to test that the system satisfy those requirements. The fourth section is the Data Model and Description part. At this section, the domain of the software explain with the class diagrams. At the end the conclusion part and the appendix part, which contains the alloy code and corresponding diagrams, take place.

# 2. Overall description

## 2.1 Product Perspective

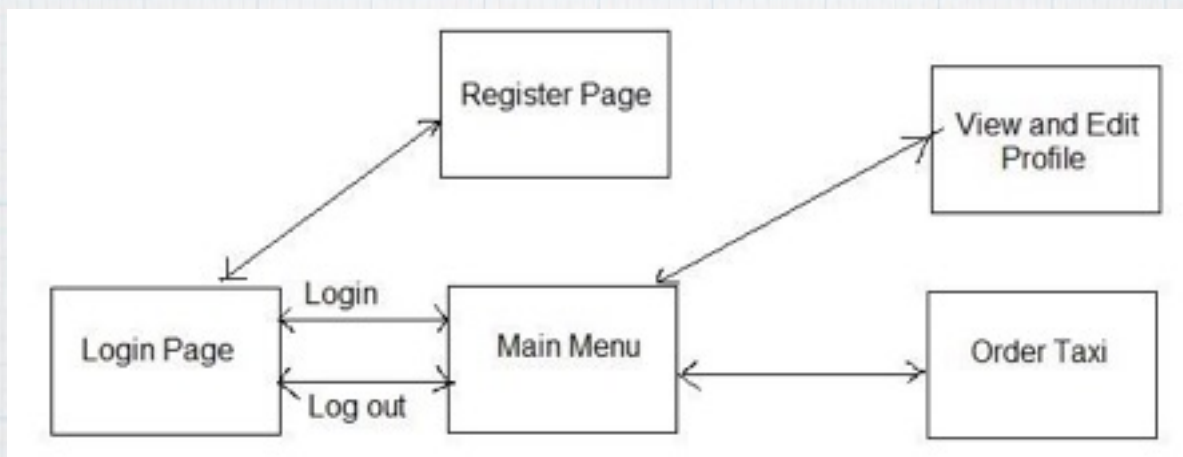
Our product is a mobile and web application mainly focuses on ordering online taxi. It is a stand-alone product that is not depend on a larger system or included in one. It mainly consists of two parts: front-end and back-end. While front-end is responsible for providing all product functionalities to the user with useful, smooth and easy interfaces, back-end is in charge to managing all functionalities and communicating with the front-end in order to provide the best experience to the users

### 2.1.1 System Interfaces

There are no system interfaces provided in our platform .

### 2.1.2 User Interfaces

There are mainly five different user interfaces. These are **Login page**, **Register**, **Main menu**, **View and Edit Profile** and the main interface from user side is **Order Taxi** interface. These pages will be explained in detail in later sections.



### 2.1.3 Hardware Interfaces

There are no hardware interfaces provided by our product.



# 2. Overall description

## 2.1.4 Software Interfaces

Our application is consists of 2 part. User Mobile application and Web application. Mobile application: For android we intend to use a java, for IOS C++, the required database system is local database SQLite, Encrypting algorithm for data transfer is RSA. When it comes to Web application: Over HTTPS, Server side scripting is Java JSP or JSF, DBS MYSQL, Client side scripting is JavaScript, jQuery. The application will work on the Glassfish Server v4.1.(Web Services on Java)

## 2.1.5 Memory

Our system memory requirements at least 1 GB in workstations (pc,laptop and etc) and also 1GB for mobile phones and tablets.

## 2.2 Product Functions

Our platform provides users an opportunity to order a taxi based on their current location. By using myTaxiService system, customers can quickly order a taxi. After completing order the system will automatically evaluate the distribution of taxis in the various zones based on the GPS information it receives from each taxi. There are mainly two kind of actors in our platform. The first actor of the project is the user. However, the users may have different roles on the system depending on their relation with the application. There are basically three different kind of users:

- **Customer:** This user can order a taxi through application
- **Driver:** This user mainly gets request from customer and if driver confirms thy system will send a confirmation to the passenger
- **Administrator:** This user basically cares about clients and also drivers, maintaining and regulate the system.

The second actor is the system. Main role of system to notify users about activities such as getting request from customers and redirecting to the drivers.It has the responsibility of notifying the customers about taxi arrival via application and e-mail.

## 2. Overall description

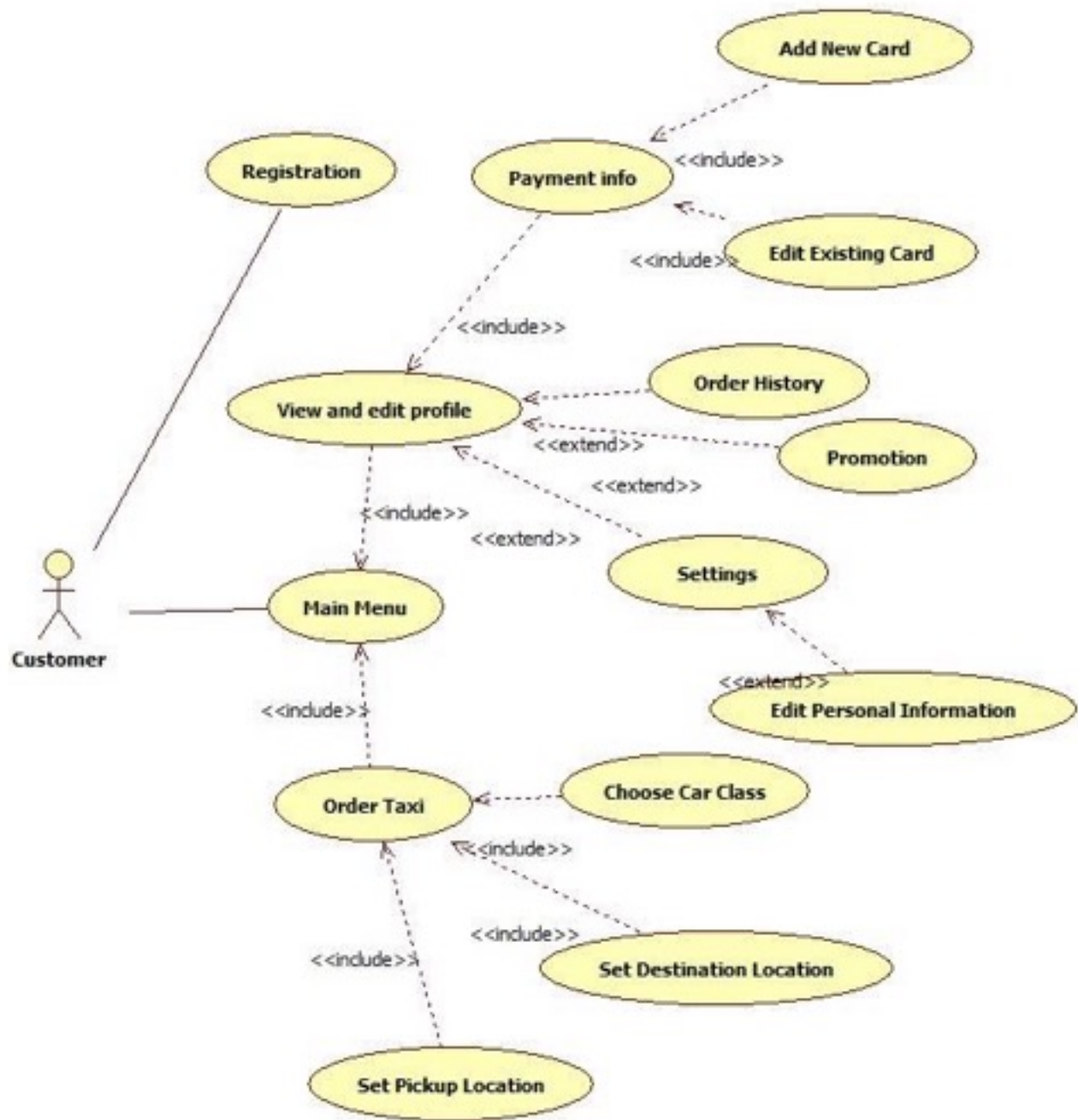


Figure 1. User «Customer» Use Cases



## 2. Overall description

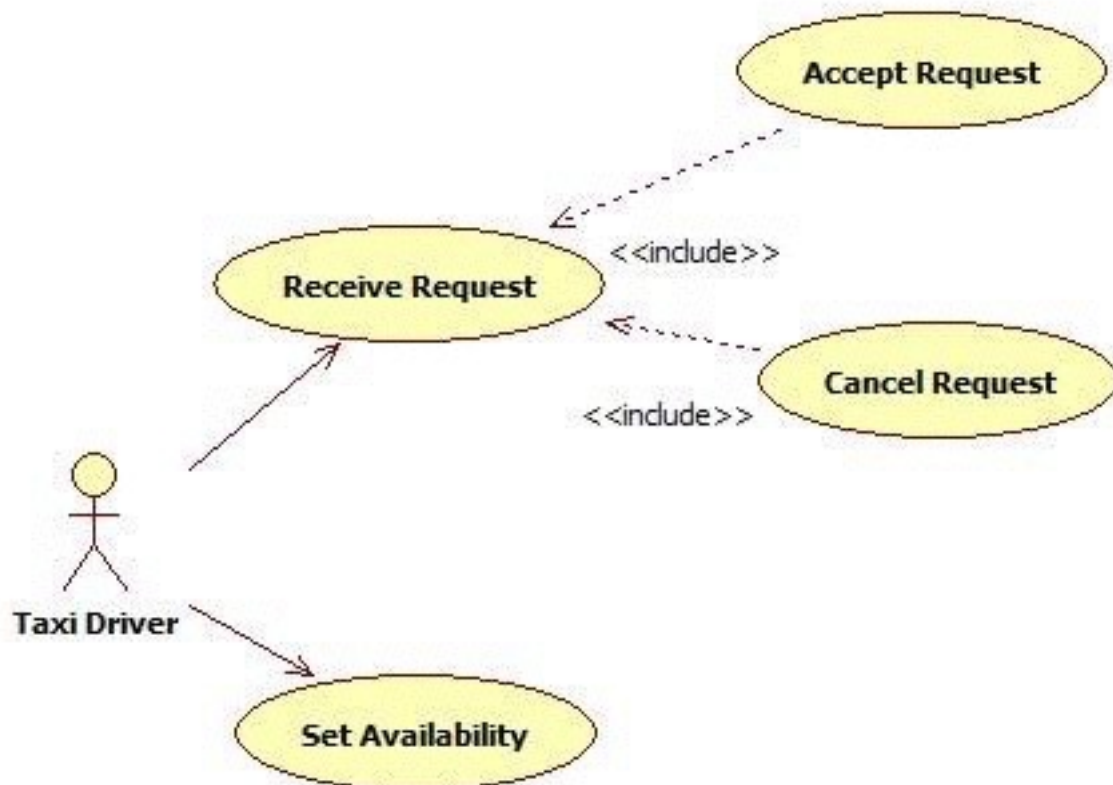


Figure 2. User «Taxi Driver» Use Cases



Figure 3. User «Administrator» Use Cases

# 2. Overall description

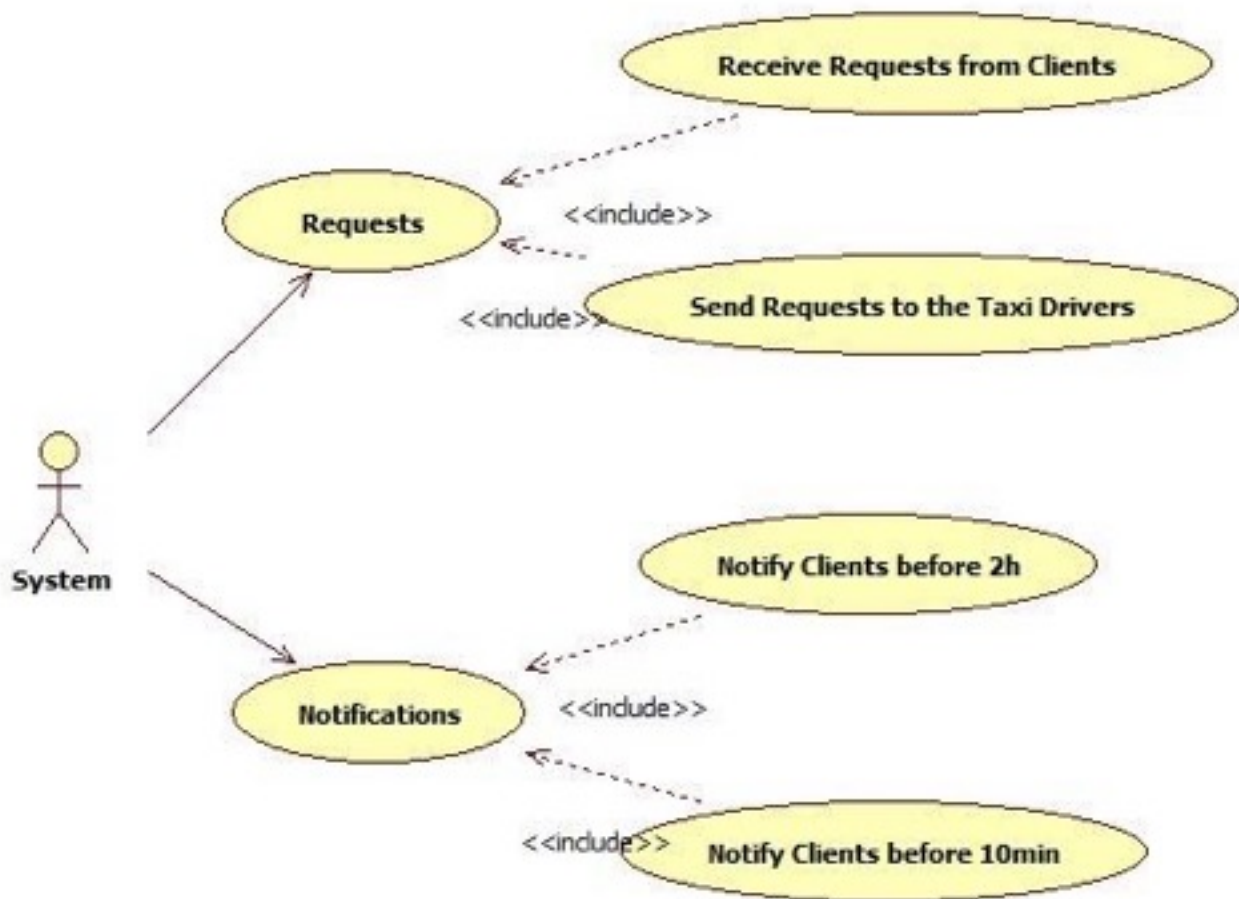


Figure 4. System Use Cases

## 2.3 Constraints

The system is constrained by the server machine on which our application works. The GPS information is constrained by the external system used for GPS tracking. Furthermore, the internet constraints the application because ordering system data is derived from the internet as the web interface is requested via HTTP Request through Internet.

## 2.4 Assumption

There are few points that are not very clear in the specification document, so we will have to assume some facts. We assume that:

- All cars have GPS system, Identical IMEI which connected to the corresponded Server. We have to assume that GPS system is working properly, otherwise, the system will not be able to recognise in which taxi zones allocate the cars.
- We also should assume that our customers are being familiar with our platform and they understand what is the online ordering taxi system and how to deal with this application.



# 3. Requirements

## 3.1 External Interface Requirements

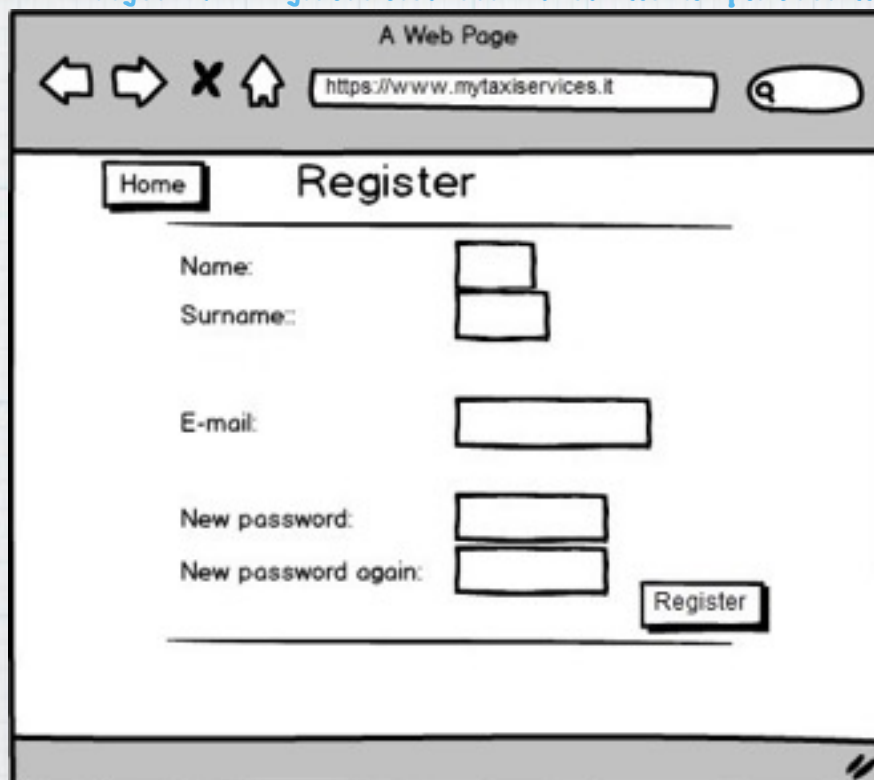
### 3.1.1 Registration screen

This mockup is related to the registration use case. It explains the registration process. To register to the system a user should fill following fields:



A mockup of a mobile application registration screen. The screen has a yellow header bar with the text "MyTaxiService". Below the header, there are five text input fields labeled "Name", "Surname", "Email address", "New Password", and "Confirm new password". At the bottom of the form is a green button with the text "Register". The entire form is displayed on a black smartphone frame.

Figure 5. Registration screen on mobile platform



A mockup of a web browser registration screen. The browser window has a title bar "A Web Page" and a address bar showing "https://www.mytaxiservices.it". Below the browser window, there is a navigation bar with a "Home" button and a "Register" button. The main content area contains five text input fields labeled "Name:", "Surname:", "E-mail:", "New password:", and "New password again:". A "Register" button is located at the bottom right of the form. The entire form is displayed within a black browser frame.

Figure 5.1 Registration screen on web platform

# 3. Requirements

## 3.1.2 Login screen

This mockup is related to the login to the system use case. It explains the login process. Registered user can login into the system filling in "e-mail" and "password" fields.



Figure 6. Login screen on mobile platform



Figure 6.1 Login screen on web platform



# 3. Requirements

## 3.1.3 Main menu screen

This mockup is related to view «Main menu» screen after successfully sing in to the system

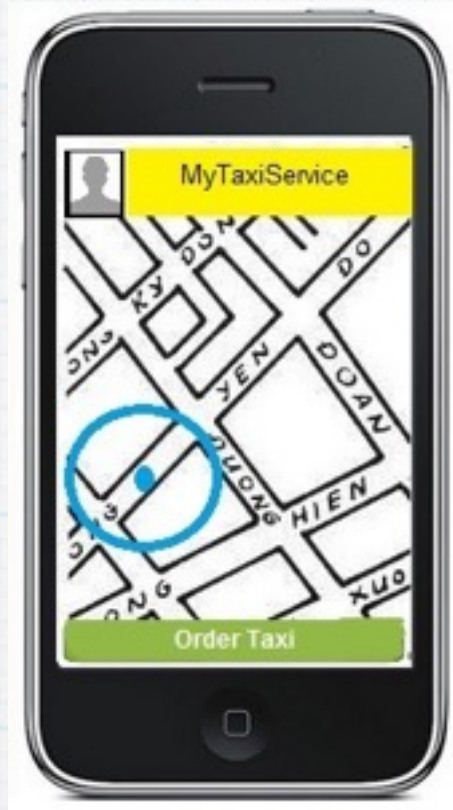


Figure 7. Main menu screen on mobile platform



Figure 7.1 Main menu screen on web platform

# 3. Requirements

## 3.1.4 Edit profile screen

This mockup is related to the edit profile use case. It explains the edit profile process. By pressing on photo (left up side) Registered customers are able to edit all the fields of their profile, add photos, payment method, promo code.



Figure 8. Edit profile screen on mobile platform

A web platform mockup for 'MyTaxiService'. The browser address bar shows 'https://www.mytaxiservices.it'. The page title is 'MyTaxiService'. The main heading is 'VIEW AND EDIT PROFILE'. There are input fields for 'Name', 'Surname', 'Email Address', and 'Add Promo Code'. A 'View Order History' button is present. On the right, there's a profile icon and an 'Add Photo' button. At the bottom right, there's a 'Save' button. The footer contains 'Copyright All Rights Reserved'.

Figure 8.1 Edit profile screen on web platform



# 3. Requirements

## 3.1.5 Order a Taxi screen

This mockup is related to the «Order a Taxi» use case. It explains how to order a taxi. Customers should set order details such as pickup location, destination location, car type.



Figure9. Order a Taxi screen on mobile platform



Figure 9.1 Order a Taxi screen on web platform

# 3. Requirements

## 3.1.6 Summary screen

This mockup is related to the «Order Summary use case. After filling all required data (show on figure 9) customer will face with new screen namely summary screen. Summary screen shows distance, visualise map and estimated fare.



Figure 10. Order Summary screen on mobile platform



Figure 10.1 Order Summary screen on web platform



# 3. Requirements

## 3.1.7 Payment screen

This mockup is related to the payment procedure use case. It is one of the main part of our software. First of all, customers should add their credit cards and save it in the our system. When they order a taxi after filling all data about trip,system redirects them to the payment window.

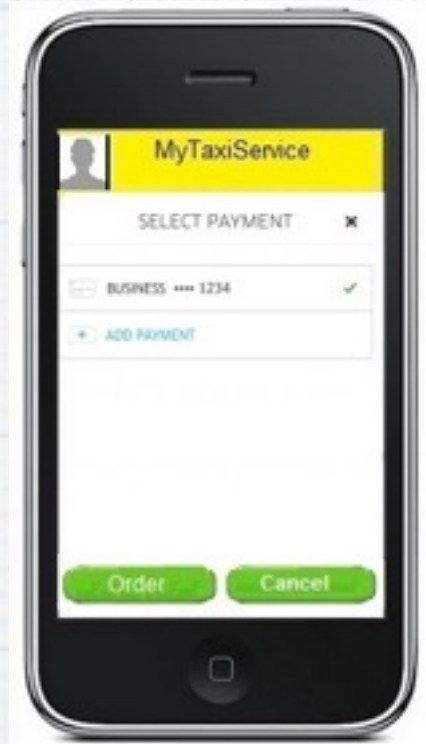


Figure 1 1. Payment screen on mobile platform



Figure 1 1.1 Payment screen on web platform

# 3. Requirements

## 3.1.8 Trip confirmation screen

This mockup is related to the last screen, trip confirmation use case. After accepting customer order from driver side, the system basically notify the customers about progress sending them a detailed information about their order.

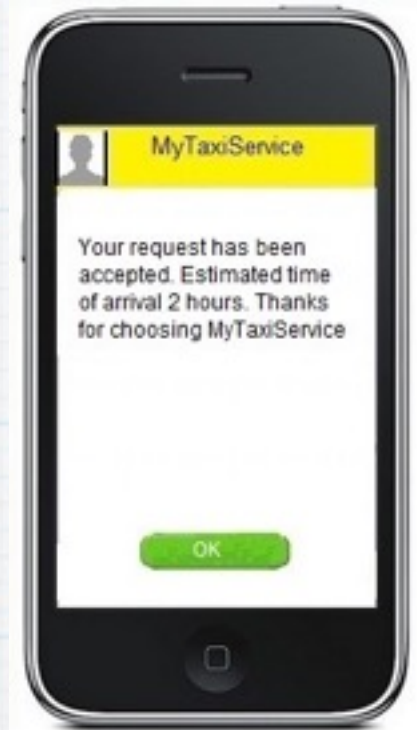


Figure 12. Trip confirmation screen on mobile platform



Figure 12.1 Trip confirmation screen on web platform



# 3. Requirements

## 3.2 Function Requirements

### 3.2.1 Sign up

Name	Signup
Actor	User-Customer
Entry condition	There are no entry conditions.
Flow of events	<ul style="list-style-type: none"><li>• The user opens the web page or application on phone</li><li>• The system shows the registration page to user.</li><li>• The user clicks on Register button.</li><li>• The system shows the form page to user.</li><li>• The user fills the form.</li><li>• The user clicks on "submit" button.</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• The system shows a success message.</li><li>• The system loads the home page.</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• The user enters wrong inputs in some fields, then an error messages is shown</li><li>• The user leaves compulsory fields empty, then an error message is shown</li><li>• The user refreshes the page without submit, then data is lost.</li></ul>

### 3.2.2 Login

Name	Login to the system
Actor	User-Customer
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• The user opens the web page or application on phone</li><li>• The system shows the registration page to user.</li><li>• The user enters his email and password in the input form.</li><li>• The user clicks on "Login" button.</li><li>• The system shows the main page.</li></ul>
Exit conditions	There are no exit conditions
Exceptions	The user enters wrong email and/or password, then an error message is shown.

# 3. Requirements

## 3.2 Function Requirements

### 3.2.3 Order a Taxi

Name	Order a Taxi
Actor	User-Customer
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• The user opens the web page or application on phone</li><li>• The system shows the main page to user.</li><li>• The user clicks on «Order Taxi» button.</li><li>• The system shows the order page to user.</li><li>• The user fills the form.(pickup,destination location,car type )</li><li>• The user clicks on "continue" button.</li><li>• The system shows the summary page and need user confirm</li><li>• the user click on «confirm» and then paying for fare or cancel the order</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• The system shows a success message.</li><li>• The system loads the home page.</li><li>• The system shows notification about taxi arrival</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• The user enters wrong inputs in some fields, then an error messages is shown</li><li>• The user leaves compulsory fields empty, then an error message is shown</li><li>• The user refreshes the page without submit, then data is lost.</li></ul>



# 3. Requirements

## 3.2 Function Requirements

### 3.2.4 View and Edit Profile

Name	View and Edit profile
Actor	User-Customer
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• The user opens the web page or application on phone</li><li>• The system shows the main page to user.</li><li>• The user clicks on «profile» button. (in the leftmost )</li><li>• The system shows the Profile page to user.</li><li>• The user can edit its personal information,add payment method ,view order history and also add promo code to get discount</li><li>• The user clicks on "save" button.</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• The system shows a success message.</li><li>• The system loads the home page.</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• The user enters wrong inputs in some fields, then an error messages is shown</li><li>• The user leaves compulsory fields empty, then an error message is shown</li><li>• The user refreshes the page without submit, then data is lost.</li></ul>

# 3. Requirements

## 3. 2 Function Requirements

### 3. 2. 5 Taxi driver actions

Name	Taxi driver action
Actor	User-Driver
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• The user opens application on tablet «Garmin»</li><li>• The system shows the main page to user.</li><li>• The user gets notification about new order</li><li>• The user has to confirm order or cancel it.</li><li>• If the user confirm order his status will be changed from available to busy</li><li>• The customer will notify about taxi driver decision</li><li>• the user click on «confirm» and then paying for fare or cancel the</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• The system shows a success message.</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• There are no exceptions</li></ul>



# 3. Requirements

## 3.2 Function Requirements

### 3.2.6 Administrator

Name	Administrator
Actor	User-Administrator
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• The user opens application on the web page</li><li>• The system shows the main page to user.</li><li>• The user can add/remove taxi drivers in the system</li><li>• The user has the right to track and find out exact location of drivers through the GPS map.</li><li>• The user also should take care of customers by analysing official complains and solving the problem via administrative tools</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• There are no any conditions</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• There are no exceptions</li></ul>

# 3. Requirements

## 3.2 Function Requirements

### 3.2.7 Notify Users

Name	Notify users
Actor	System
Entry condition	User must be registered into the system .
Flow of events	<ul style="list-style-type: none"><li>• There are 2 conditions of notifying users by the system</li><li>• After getting order from customer side and also confirmation from driver side, the system can notify the customer before 2 hours of its trip and also 10 minutes before</li><li>•</li></ul>
Exit conditions	<ul style="list-style-type: none"><li>• There are no any conditions</li></ul>
Exceptions	<ul style="list-style-type: none"><li>• There are no exceptions</li></ul>



# 3. Requirements

## 3.3 Sequence Diagram

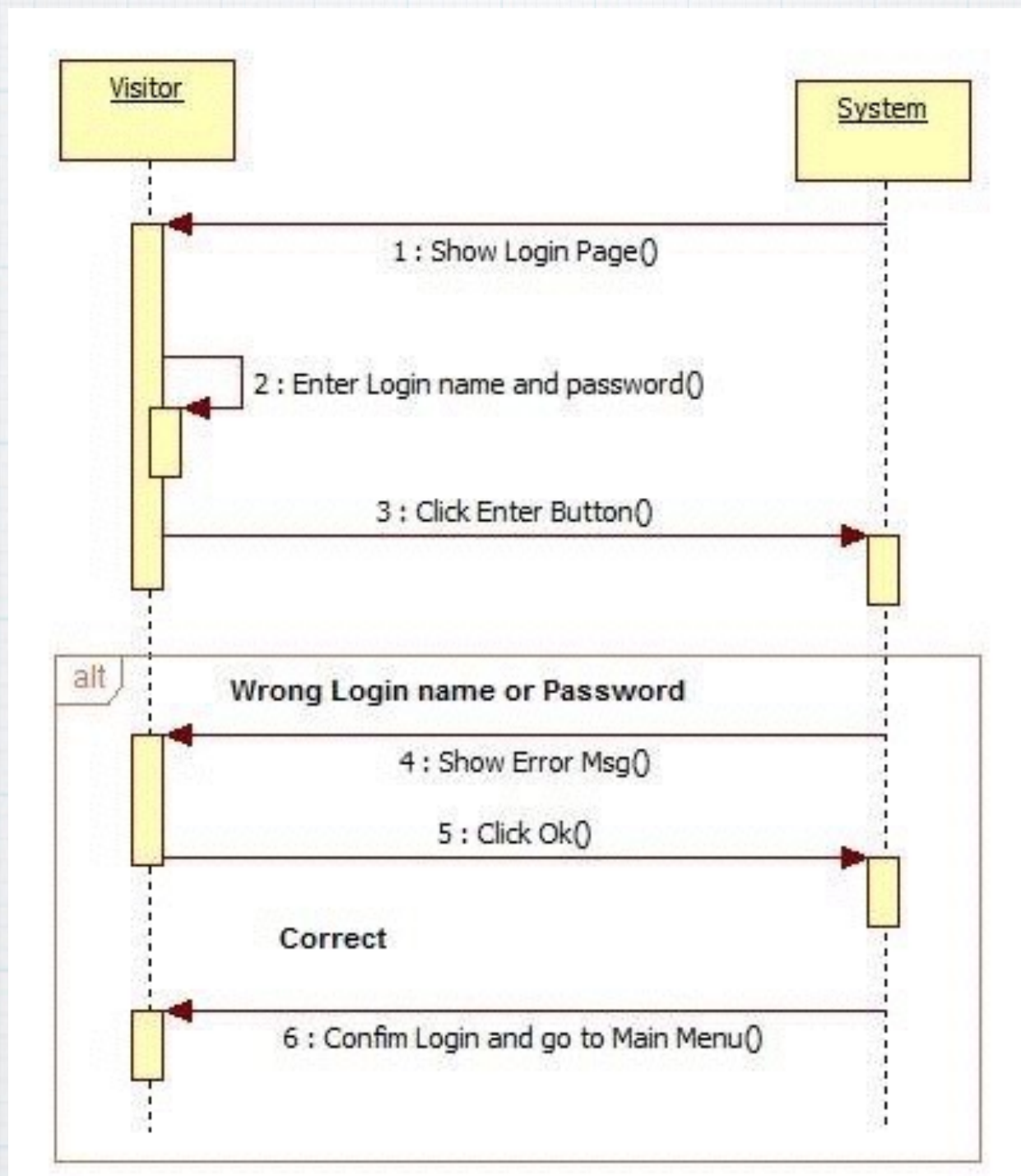


Figure 13. Login

# 3. Requirements

## 3.3 Sequence Diagram

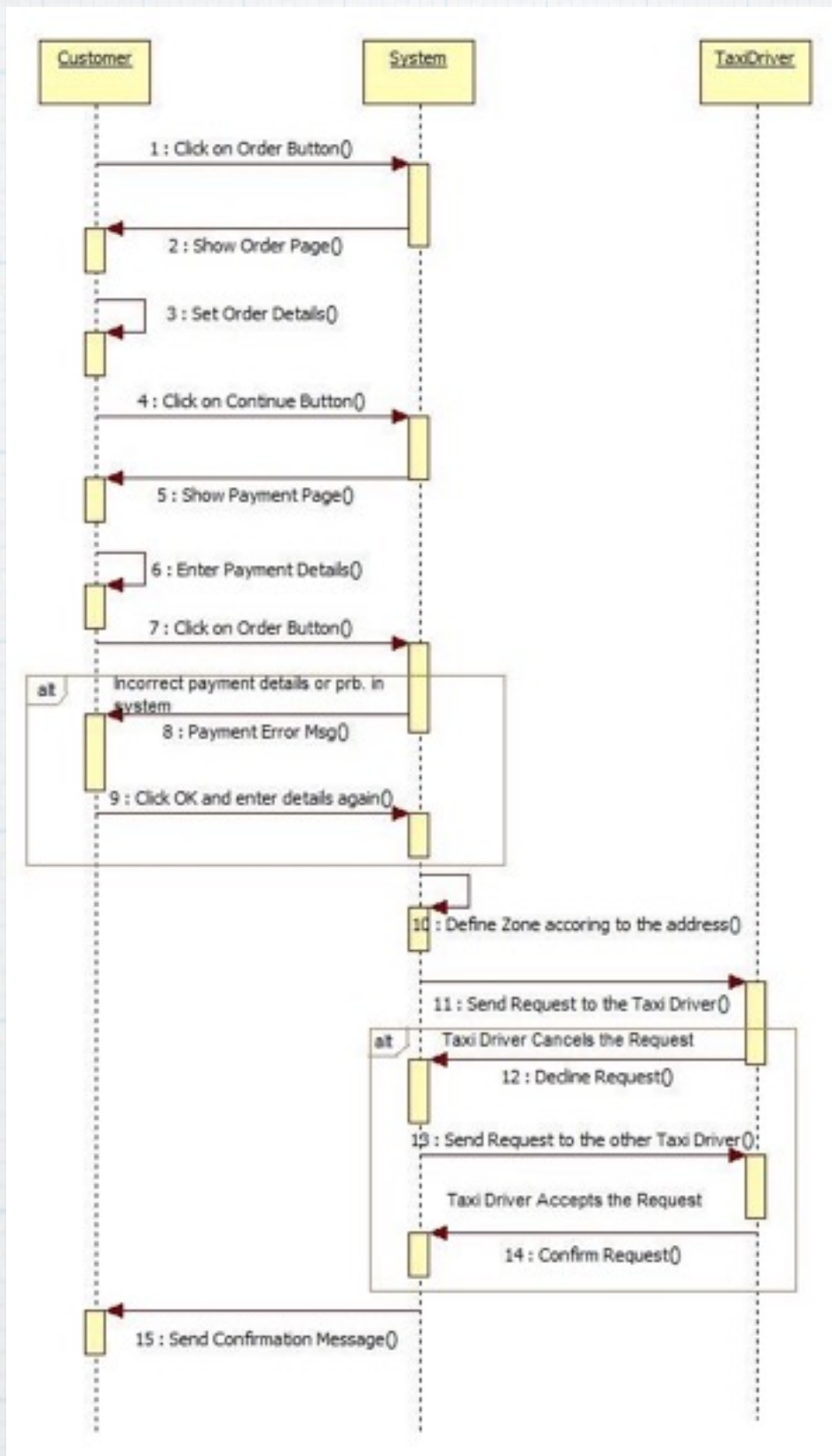


Figure 14. Order a Taxi



# 3. Requirements

## 3.4 StateChart diagrams

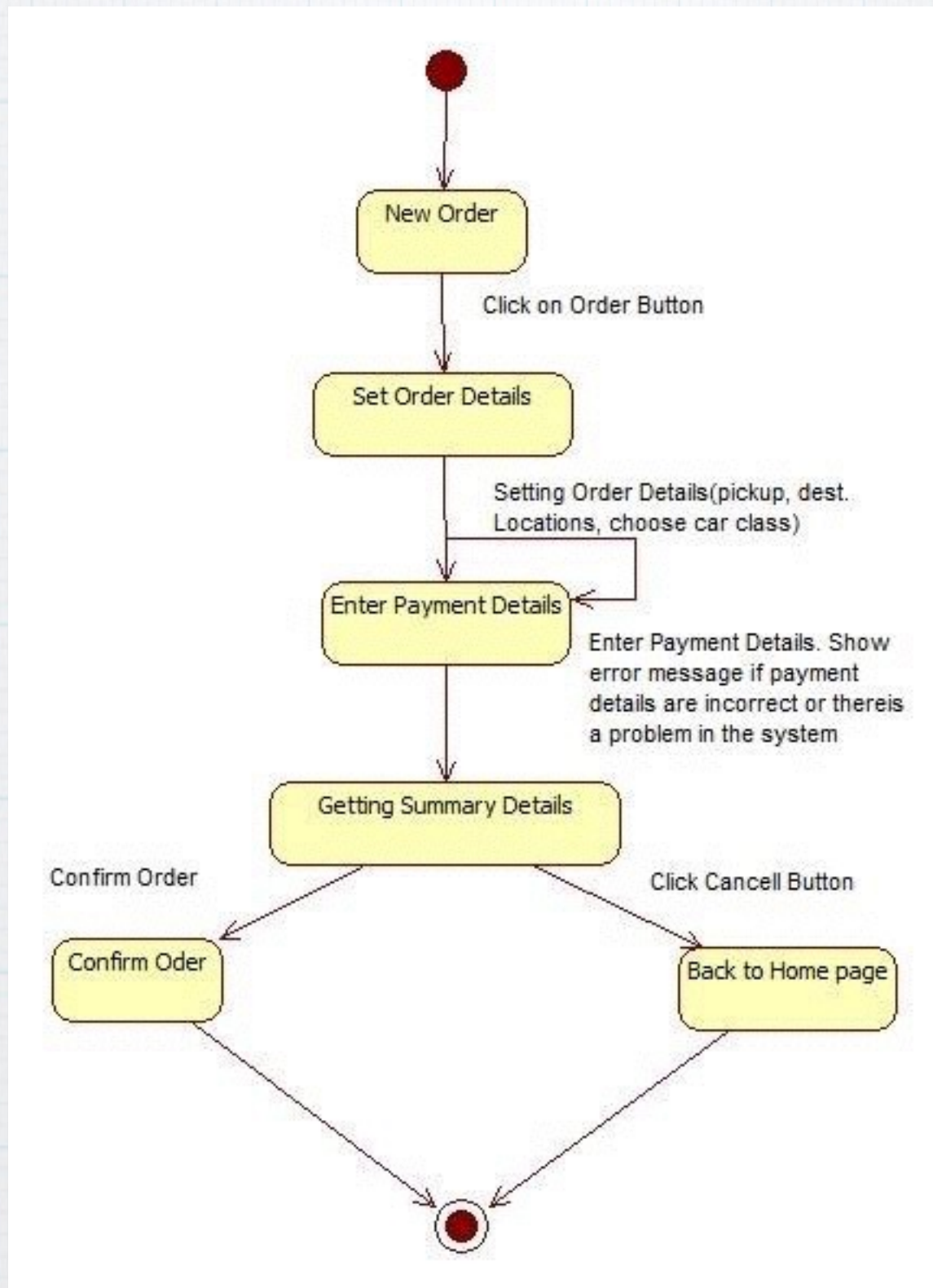


Figure 15. Order a Taxi

# 3. Requirements

## 3.5 Non-functional Requirements

### 3.5.1 Reliability

The system should be able to behave consistently in a user-acceptable manner when operating within the environment for which the system was intended.

### 3.5.2 Availability

The system should be able to deliver services when requested.

### 3.5.3 Security

The system should be able to protect itself against accidental or deliberate intrusion.

### 3.5.4 Maintainability

Maintainability is the parameter concerned with how the system in use can be restored after a failure. The system should periodically have back ups in order not to lose any information.

### 3.5.5 Portability

The application should easily be transferred from one computer environment to another.



# 4. Data Model and Description

There are four main data classes which are «Order a Taxi», «User», «View and edit Profile» and last one called «Taxi queue» classes.

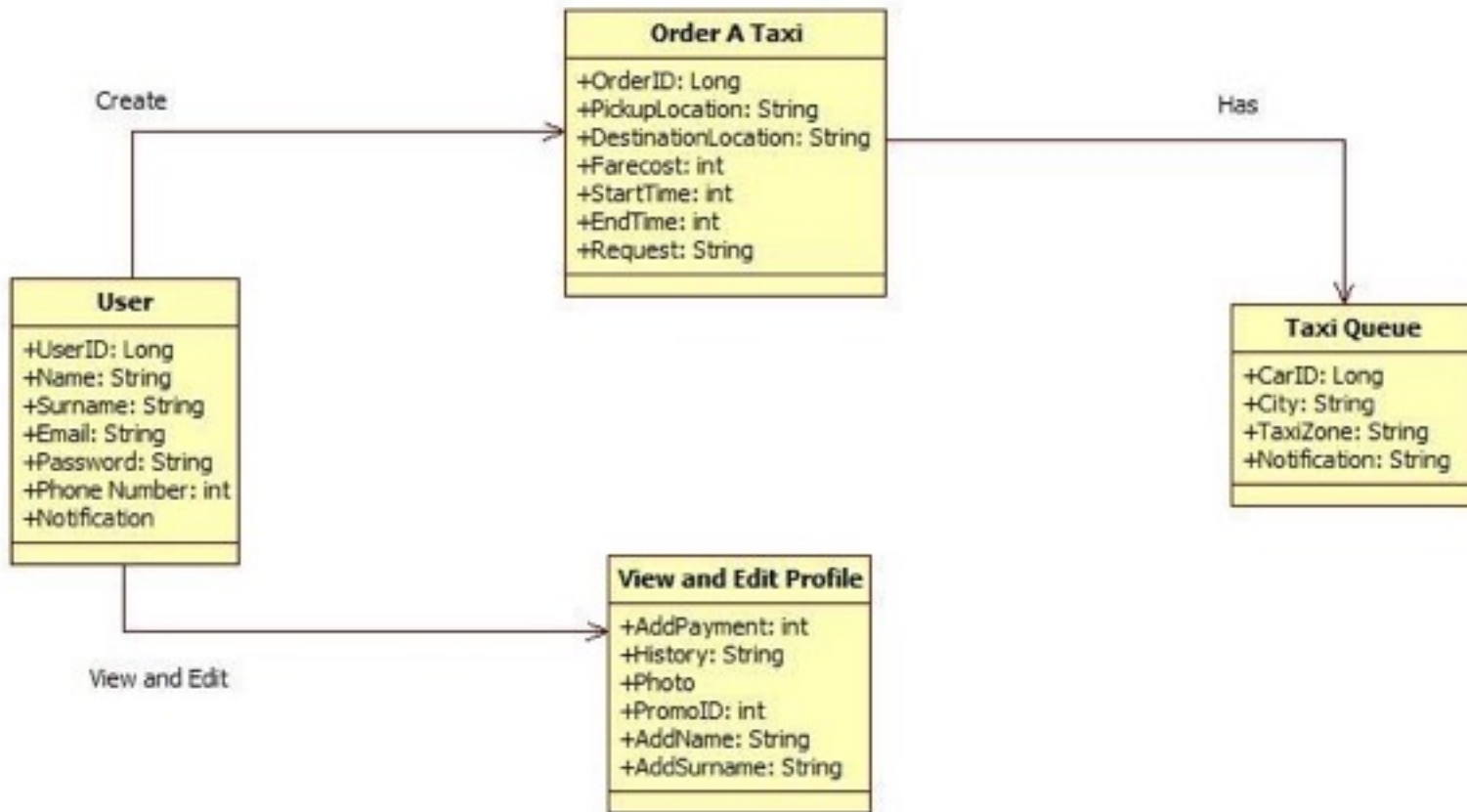


Figure 16. Class Diagram

## 4.1 User

User class stores user details and notifications received.

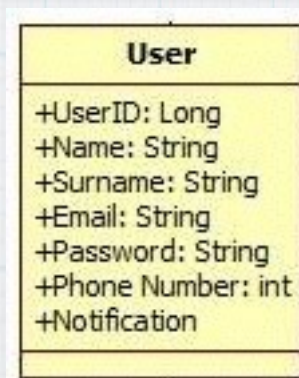


Figure 17. «User» Class Diagram

# 4. Data Model and Description

## 4.2 Order a Taxi

This class stores the order details including estimated fare cost, pickup and destination location. There is one relation between mentioned and the User class called «create».

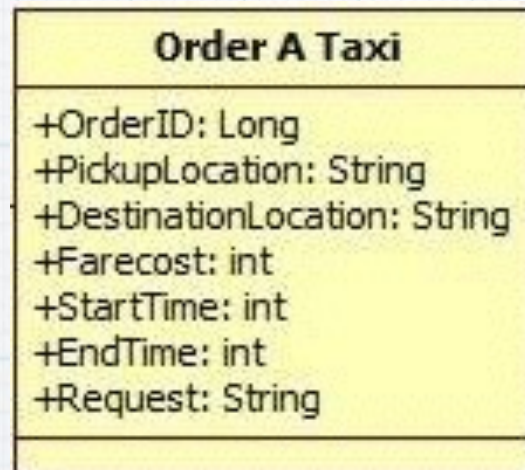


Figure 18. «Order a Taxi» Class Diagram

## 4.3 View and edit Profile

This class stores the account information of customers. Every customer of system should have its own profile and personal data.



Figure 19. «View and Edit Profile» Class Diagram



# 4. Data Model and Description

## 4.4 Taxi Queue

This class stores the Taxi queuing information, and also its location, city which it located, taxi zone and etc.

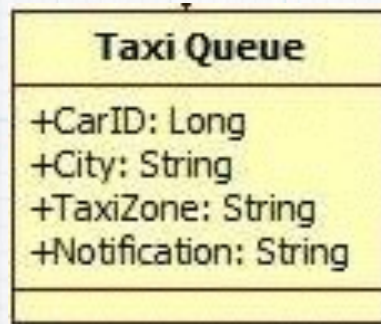


Figure 18. «Taxi Queue» Class Diagram

# 5. Conclusion

To sum up, all required information about the requirements of this project has been given. This project will provide online taxi ordering system interface in the basis of taxi queuing system with its excellent and usable interfaces to its users. Now the project is ready to go on with determining more detailed design of the product



# 6. Alloy

```
module MyTaxiService

/ Signatures
sig pickuplocation {}
sig destinationlocation {}
abstract sig carclass {}
sig taxi {}
sig creditcard {}

abstract sig request {
    sendrequest: lone system
}

abstract sig system {
    sendrequest: set taxi,
    sendnotification: one user
}

abstract sig notification {}

sig order {
    startpoint: one pickuplocation,
    endpoint: one destinationlocation,
    comfort: set carclass,
    paymentmethod: set creditcard,
    sendrequest: one request
}

sig user {}

/ Facts
fact orderproperties {
    / Order A and Order B can have the same request
    no disj o1, o2 : order | o1.sendrequest = o2.sendrequest
    / Order A and Order B can have the same carclass
    no disj o1, o2 : order | o1.comfort = o2.comfort
}

/ Assertions
assert request {
    no disj o1, o2 : order | o1.sendrequest = o2.sendrequest
}
check request

assert carclass {
    no disj o1, o2 : order | o1.comfort = o2.comfort
}
check carclass

/ Predicates
pred showorder(){
    #order = 1
}
run showorder

pred showorder(){
    #order = 2
}
run showorder
```

Figure 19. Alloy, Order a Taxi, codes, for one and two orders

# 6. Alloy

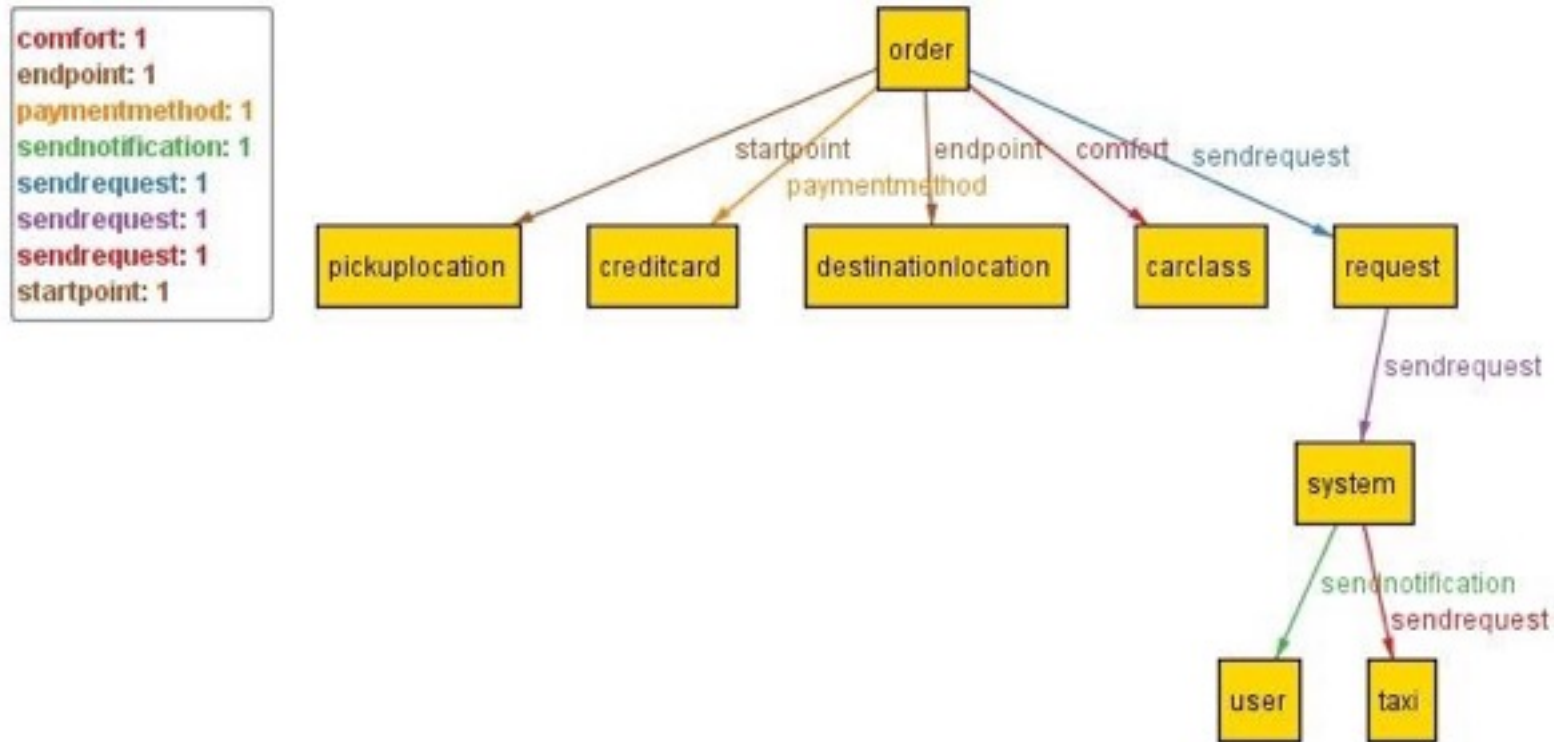


Figure 20 myTaxiService Instance (simplified) view  
1 orders

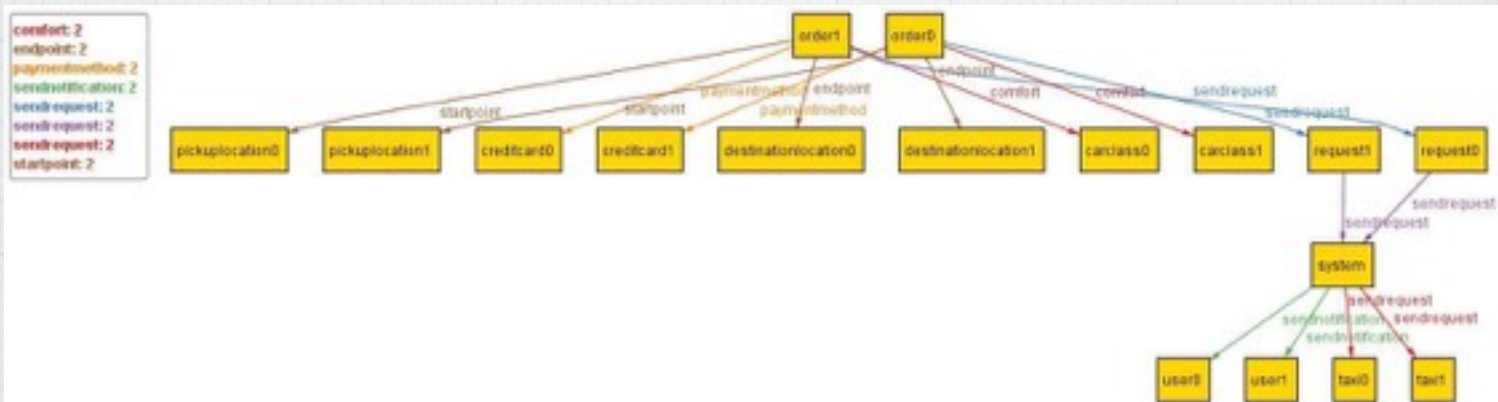


Figure 21. myTaxiService Instance (simplified) view  
2 orders