POLITECNICO DI MILANO

A.A. 2015-2016

SOFTWARE ENGINEERING 2: “myTaxiService”

Requirements Analysis and Specifications Document

Andrijana Mirchevska (838622)

Marija Mavcheva(838647)

09 November 2015

Contents

1. Introduction

1.1 Purpose

The purpose of this document is to provide a comprehensive description of the myTaxiService system. It's aim is to describe what the software should do and identify the capabilities and characteristics of the system being developed. This document is meant for everyone included in the production of the software.

1.2 Scope

The project is about developing an application that will enable fast and optimized taxi services in the city. The application will allow users to register and then sign in into the app for using its services. Also taxi drivers can register and sign into the myTaxiService application with the purpose to manage their availability and duties.

Once registered into the app, the user can do the following actions:

- request a taxi
- make reservation
- cancel reservation
- find available taxi near
- receive confirmation about the vehicle, its code and estimated waiting time
- manage user profile
- report driver

Also the registered taxi driver once signed in can do these actions:

- inform the system about availability
- confirm/decline a request for taxi call
- manage driver profile
- report user

1.3 Glossary

MakeReservation	Passenger request a vehicle at least 2 hours before the ride
Request	Passenger request an immediate ride
RideConfirmation	Receive a confirmation about the confirmed ride with all the information about the particular ride
ReservationConfirmation	Notification that the reservation is successfully completed
ReportDriver	Passenger reports driver in case of any irregularities during the ride

ReportUser	Taxi driver reports user (particular passenger) in case of any irregularities during the ride
Guest	Not registered person that visits the app
User	A person that is already registered and signed in as user
TaxiDriver	A person that is already registered and signed in as driver
API	Application Programming Interface
GPS	Global Positioning System

2. Overall Description

2.1 Product perspective

Our mobile web application will provide the user with services described in section 1.2. The software will be developed using a client-server model. The server side contains the application logic and is used to interact with permanent storage, serve pages to the client and process user input. The web client consists of dynamic web pages which provides user friendly graphical interface and the web browser through which they are accessed. The application is platform independent. The only requirement is users having a web browser installed on a device of their choice.

2.2 Identifying stakeholders

There are four distinct interest groups of people regarding this project:

- *Company* - which required project specification and expects it to be delivered in a way that satisfies given specification while respecting the set deadlines and budget
- *Developer group* - in this case group of two people
- *Taxi driver* - worker at the company that ordered the software product
- *Passenger* - person who need a ride to specific location

2.3 User characteristics

myTaxiService is expected to have users across a wide range of demographics, meaning users of any age, gender and educational background. Still it is assumed that people using our software do have the basic web browsing skills.

2.4 Actors identifying

There are four possible actors interacting with our system:

- **Guest -** is a person that access the system but has never registered on it or still hasn`t logged in. The only possibilities of the guest, who can access just the initial page, are to register into it or to log in.
- **User -** is a person that is already registered and logged into the system. The user can access and manage with all the services that the application offers.
- **Taxidriver -** a person that is already registered and logged into the system. The user can access and manage with all the services that the taxi driver application offers.
- **Admin** - is the person that is responsible for handling reports on users and drivers of my taxi service. Admin can ban a user or a driver from the system.

2.5 Goals

Having possible users in mind, myTaxyService should have these features:

- registering a new user
- sending notification about taxi availability
- confirming about the reserved vehicle, its code and waiting time
- managing user profile
- ban user/driver

- for each **user** it should provide:
  - logging in to his/her profile
  - making reservation
  - requesting taxi
  - canceling reservation
  - reporting driver

- for each **taxi driver** it should provide:
  - logging in to his/her profile
  - confirming/declining a request for taxi call
  - reporting user

2.6 Domain properties

- user making reservation from a specific location to a specific destination
- taxis are organized into taxi zones
- the payment process is irrelevant to the system, it`s done between the passenger and driver

2.7 Assumptions

Considering that there were some ambiguities in the specification document, the following facts are assumed:

- user registers with email and password
- user can change the email and password
- user can only have one account
- there is a Terms & Conditions section that indicates clearly the usage of the application, which if not followed will result with deactivation of the account
- we assume that Google Maps service will calculate location used by myTaxiService accurately
- if Google Maps does not provide automatically the location, new screen is showed to the user where he/she types the address
- if the taxi driver does not respect the estimated waiting time, he is banned from the system
- is there are any irregularities with the taxi driver, new vehicle is sent to the passenger

3. Requirements

3.1 Functional requirements

Functional requirements are defined for the system and for each actor defined in section 2.4.

For a **Guest** the system should provide two functionalities:

- **Register**: the guest is on the initial page of the application and should provide some information (name, surname, email, password) in order to register to the app
- **Log in**: after entering the correct identification information the guest will access the homepage of the app

For a **User** the system should provide the following functionalities:

- Option to sign out
- Manage profile setting:
  - change email
  - change password
- Request taxi
- Make reservation
- Cancel reservation
- Report driver

For a **TaxiDriver** the system should provide the following functionalities:

- Option to sign out
- Manage profile setting:
  - change email
  - change password
- Confirming/declining a taxi call
- Managing the requests
- Report user

The system should:

- send notifications about taxi availability
- confirm about the reserved vehicle, its code and waiting time
- ban user/driver

3.2 Non-functional requirements

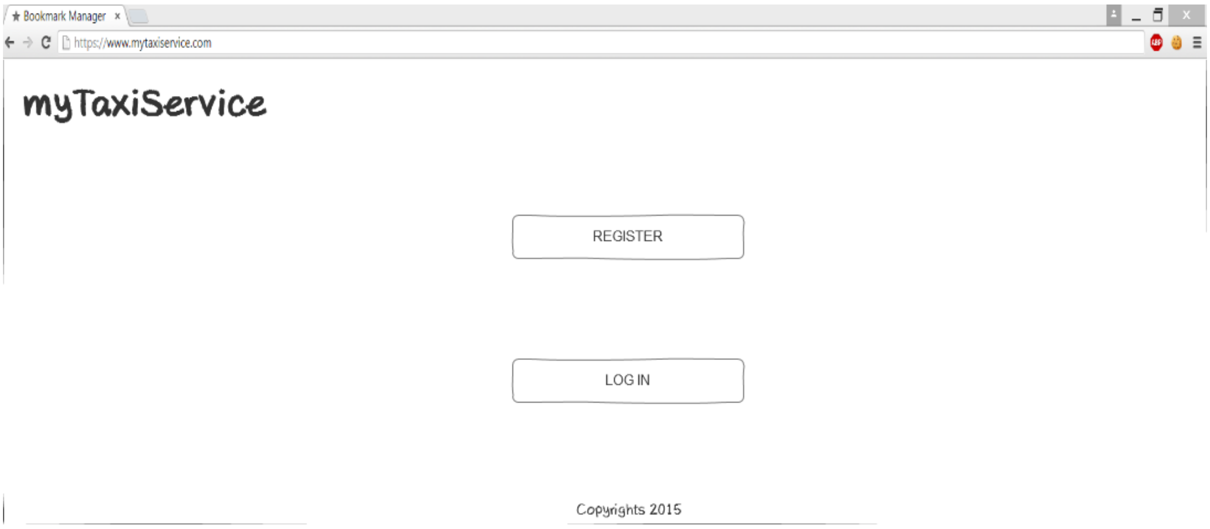
3.2.1 User-interface

The user interface of the application should be intuitive, simple and easy to use even for non-technically savvy users. The design should be clean and user friendly.

Availability - the application should be easily accessible at any time by the user who is using any kind of device with web browser installed on it.

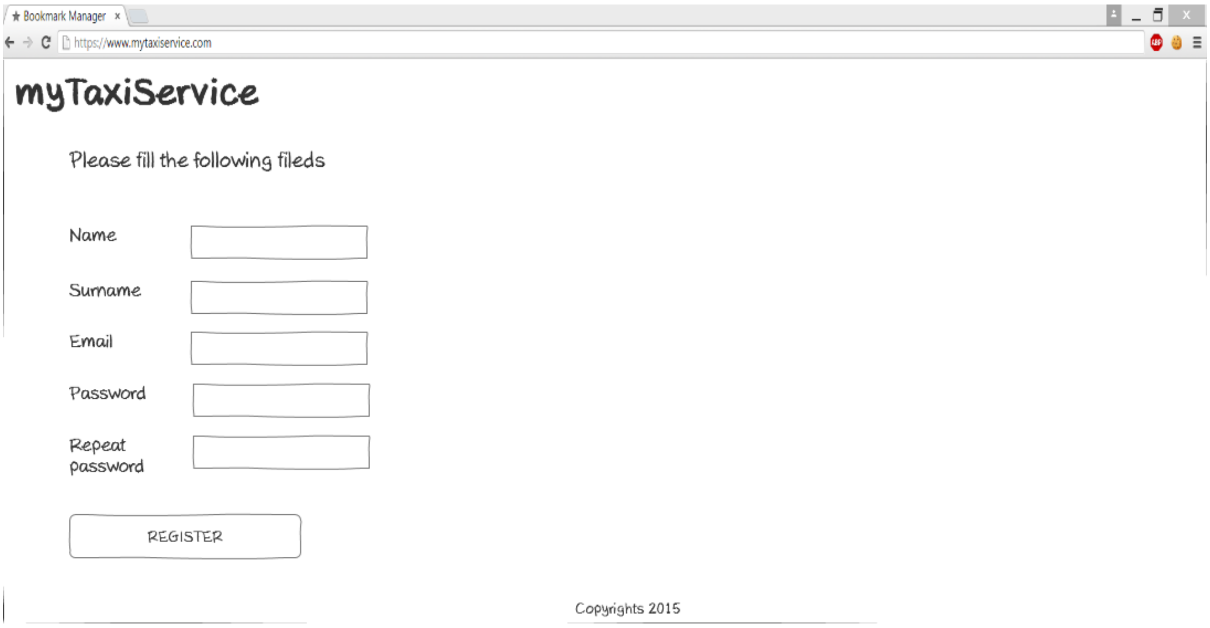
In order how we imagine the interface of our application we introduce a brief hand sketching of the main pages that the app will have.

- ➢ The following hand sketching represent the actions that the guest and user can do.



Sketch 1. Initial page of the app

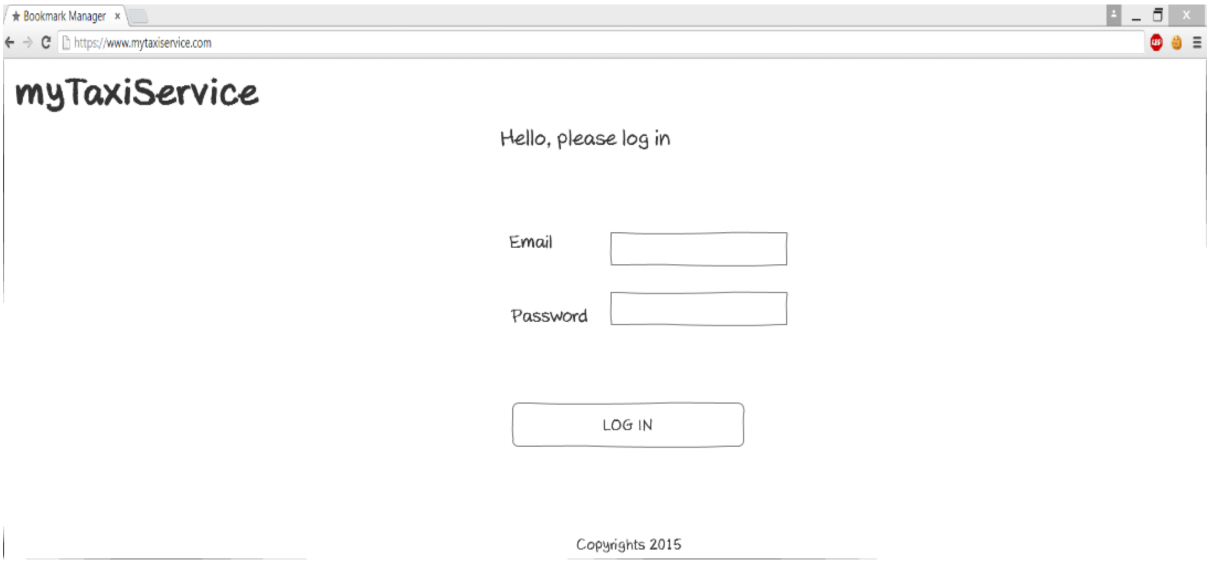
Sketch 1: The initial page is accessible by the guest, any person that is not registered or logged in into the app. There are only two options in this page either to register to the app or to log in into an existing profile.



Please fill the following fileds

Sketch 2. Register page

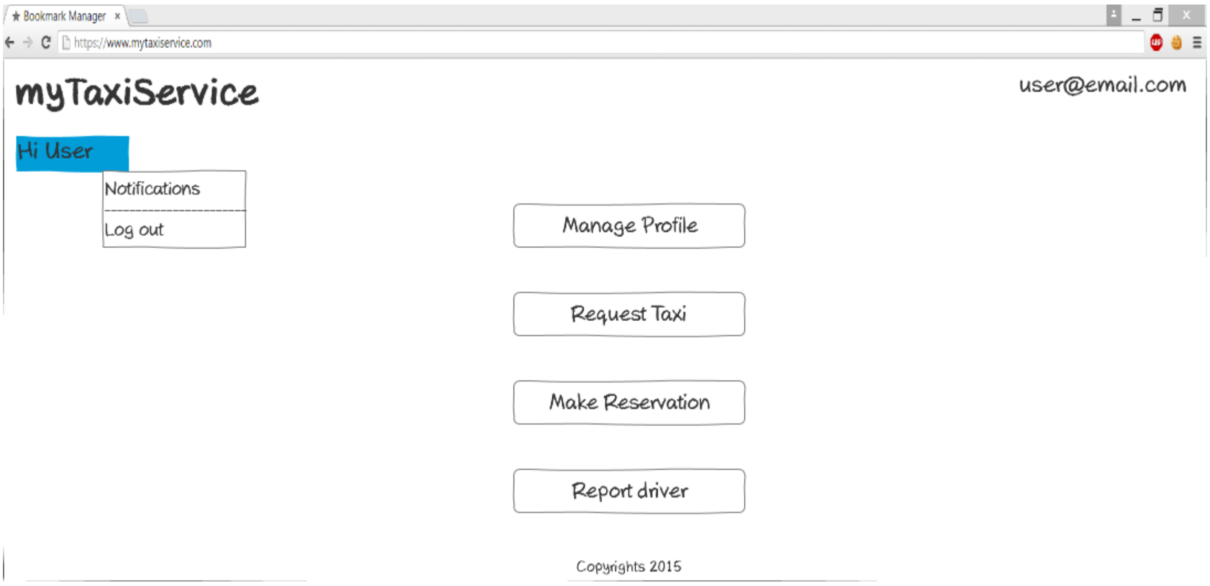
Sketch 2: If the person, who is initially a guest, chooses the register button it redirects him/her to the registration page. This is the page where the user is obligated to type his name, surname, email, and password into the given fields. Then he clicks on the register button and if the data is correct, he/she is redirected to the homepage of the app.



Hello, please log in

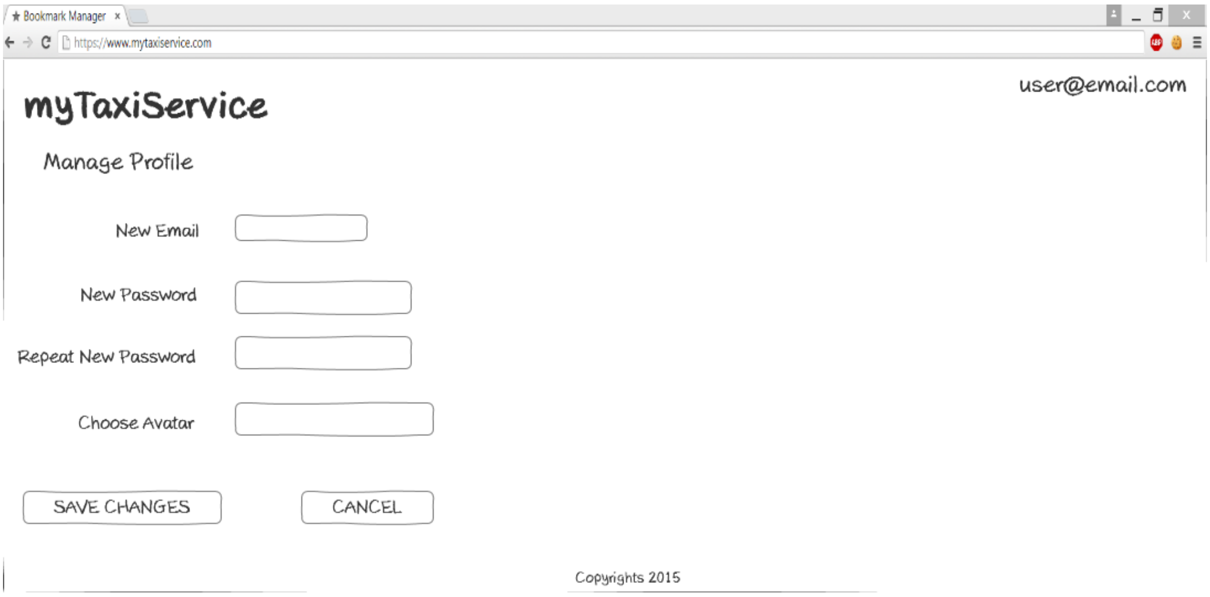
Sketch 3. Login page

Sketch 3: In the log in page the user should provide his/her email and password in order to enter to his/her profile and access the services that the app provides.



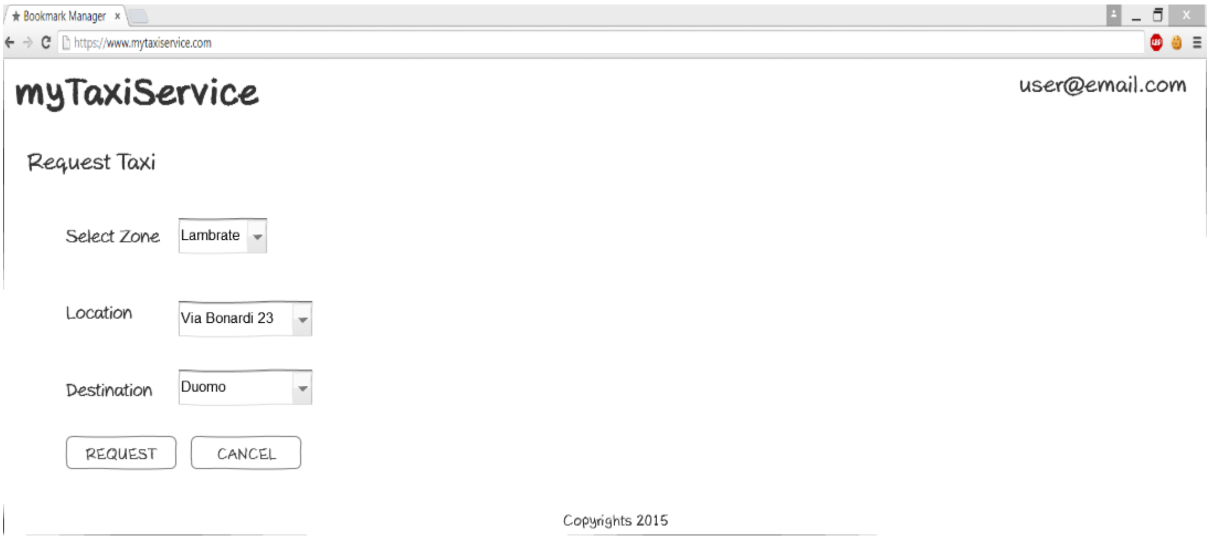
Sketch 4. Homepage

Sketch 4: Once logged in, the user is lead to the homepage of the app where he can access the services that this app provides. He can choose either to manage the profile, request a taxi, or make a reservation.



Sketch 5. Manage profile page

Sketch 5: When the user clicks on the manage profile button, the app will bring him/her to the manage profile page where he/she can modify the email and password that were previously used.



myTaxiService

user@email.com

Request Taxi

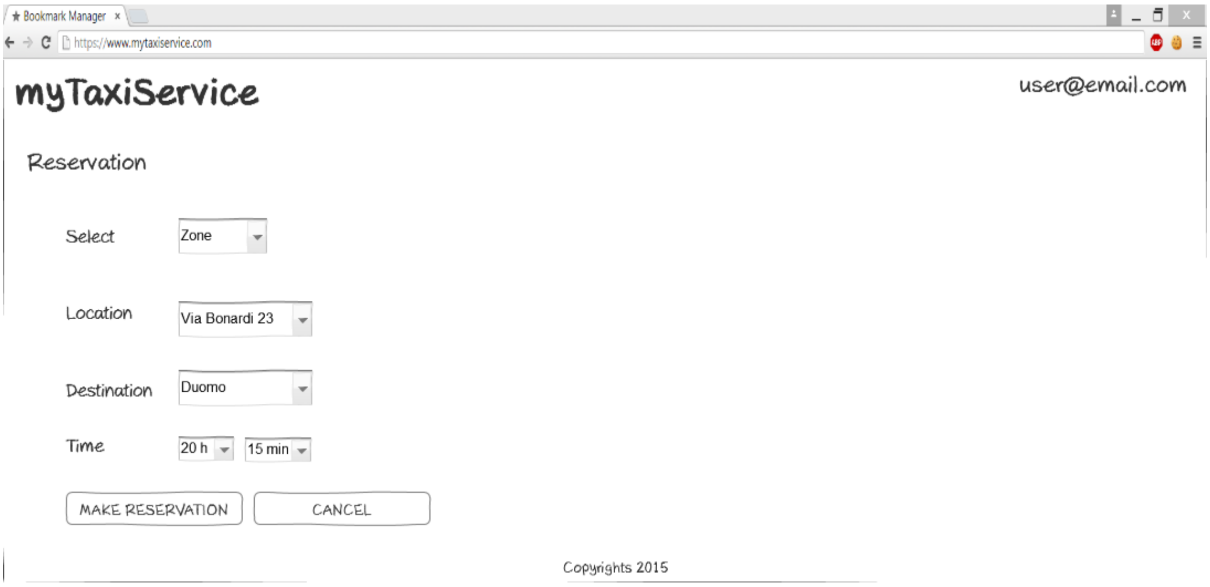
Select Zone Lambrate

Location Via Bonardi 23

Destination Duomo

Sketch 6. Request taxi page

Sketch 6: If the user wants to request a taxi immediately, he will click on the Request taxi button. In this page he should specify the following information: zone, current location, and destination of the ride. Then he clicks request and the request is sent to the system.



myTaxiService

user@email.com

Reservation

Select Zone

Location Via Bonardi 23

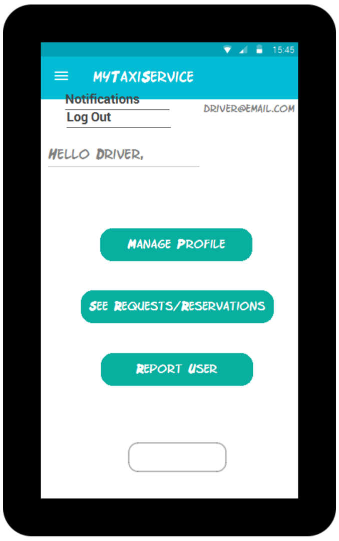
Destination Duomo

Time 20 h 15 min

Sketch 7. Make reservation page

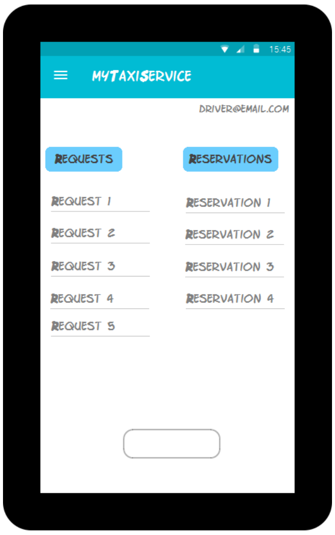
Sketch 7: If the user wants to make a reservation for a tax, he will click on the Make reservation button. In this page he should specify the following information: zone, current location, destination of the ride, and the time when he wants use the service, which should be at least 2 hours before the ride. Then he clicks make reservation and the reservation is sent to the system.

- ➢ Next hand sketchings represent the actions that the driver can do.



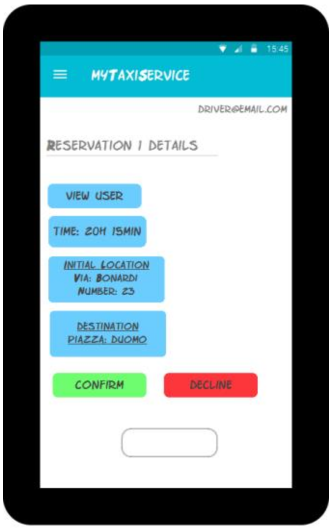
Sketch 8. Driver homepage

Sketch 8: Once logged in, the driver is lead to the homepage of the app where he can access the services that this app provides. He can choose either to manage the profile, see the requests or the reservations on which he should respond.



Sketch 9. Notification page

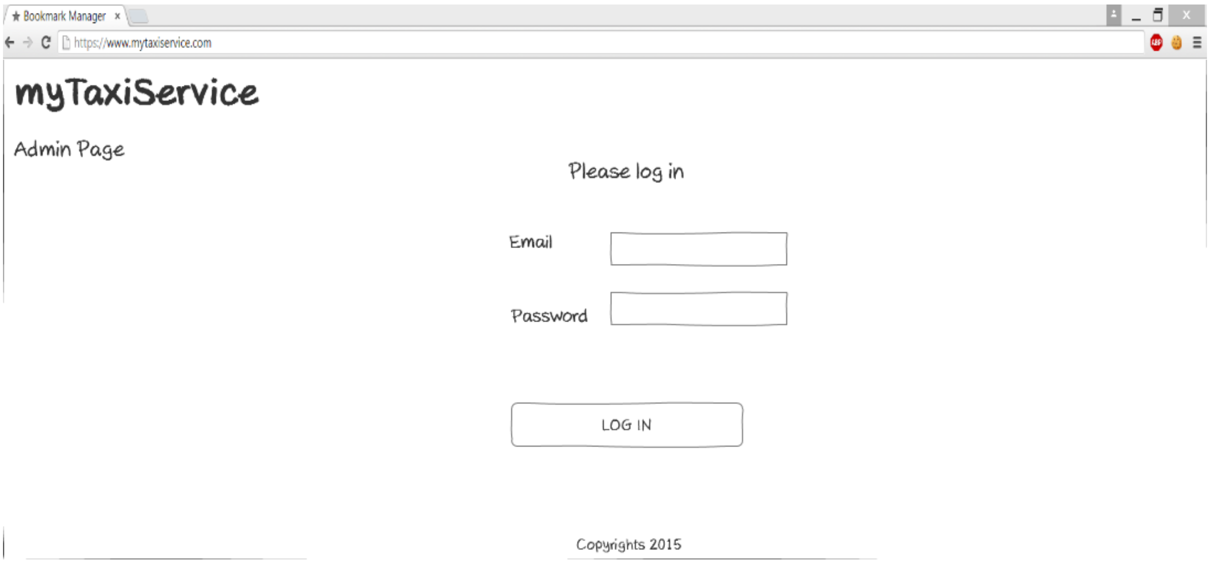
Sketch 9: When the driver chooses See requests/reservations, the page where all the requests and reservations are, is loaded. If he wants to see the details and to respond to a particular request/reservation he should click on it.



Sketch 10. Reservation details page

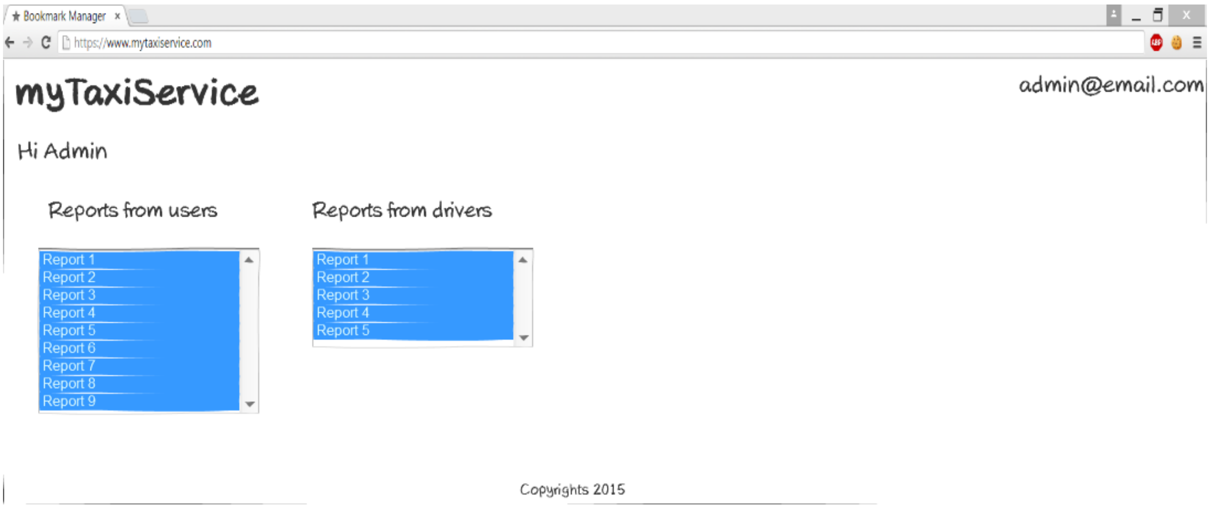
Sketch 10: When driver chooses some reservation from the list, its details are represented in a new page. Here he can see who is making the reservation (link to the user profile is provided), the time that the vehicle should be on the starting location, the address of the location (starting point), and the final destination. Further he can confirm or decline the reservation.

- ➢ And last are the hand sketching for the actions that the admin can do.



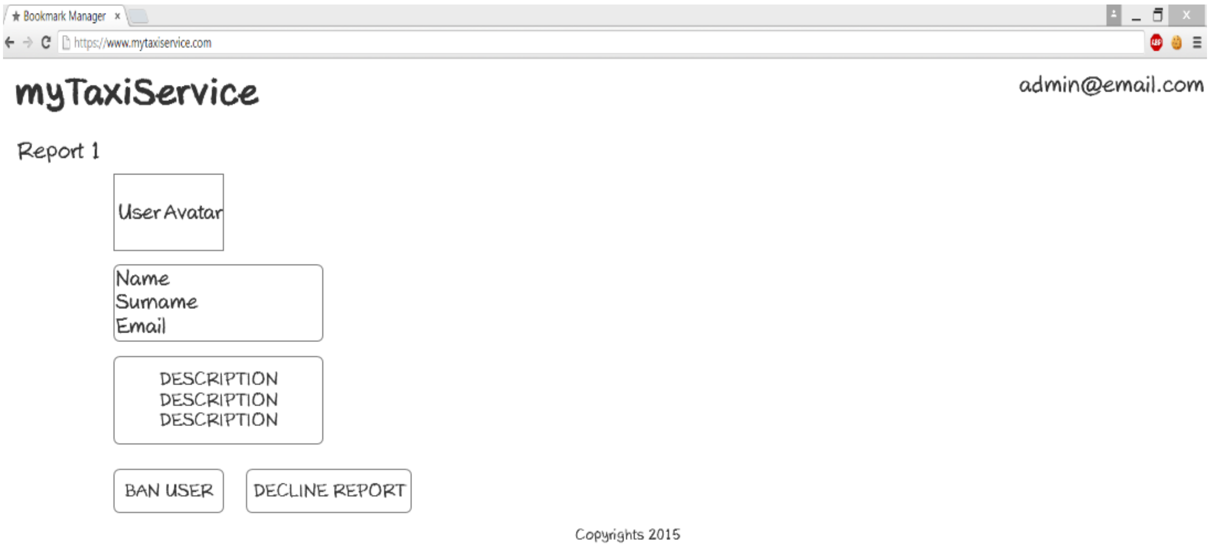
Sketch 11. Admin log in page

Sketch 11: In the admin log in page the admin should provide his/her email and password in order to enter to his/her profile where he can see the reports from the users and taxi drivers, and make further actions, banning or not a user or taxi driver.



Sketch 12 Admin Home Page

Sketch 12: After the admin logs in the app, he/she has access to all the reports that a user or a driver has given. By selecting a particular report (both from the user or the driver) admin will be lead to a new page with the details of the report.



Sketch 13 Ban user/decline report Page

Sketch 13: After clicking on a particular report (example report 1) the admin gain access to its details, such as user's profile or driver's profile, name, surname and email. Also he can see a description of the problem that happened. Therefore, he decides whether the report makes sense and bans the user/driver, or decline it.

3.2.2 API Interfaces

For the purpose of myTaxiService application Google Maps API (https://developers.google.com/maps) is used to represent the map around user's current location, as well as to provide routes between two given addresses. Google Places API (https://developers.google.com/places) is used to provide the user with autocompleting addresses and to suggest addresses.

3.2.3 Hardware interfaces

There are no hardware interfaces that are used with this application.

3.2.4 Software interfaces

- Database Management System (DBMS): MySQL
- Programming technologies: HTML5, CSS3, jQuery, PHP
- Application server: Apache
- Operating System (OS): Cross platform

4. Scenarios

Scenario 1**:** Mike is a layer and he travels a lot, because his company has two main offices, one in Milan, and other in Rome. He has a flight on Wednesday night from Milan to Rome, and his colleague tells him about a new application called myTaxiService, that he has already used and was very satisfied from it. Mike decides to use the service of the app and he makes a profile. After that, he makes reservation for Wednesday night in order to go to the airport. His flight is at 10 pm so he makes reservation for 8 pm the same day, giving all the necessary information the app is requiring.

On Wednesday at 7:50 pm the taxi is in front of Mike's home and drives him to the destination.

Scenario 2: Lina is preparing for a birthday party, but she realized that she is late. She remembers that she has installed on her smart phone an app that she saw on Facebook. The app name was myTaxiService. She already has created a profile but this is her first time using the service of the app. She requests a taxi given her current location, and ending destination. After a minute she receives a notification that the taxi with 005 code will be there in 10 minutes.

Lina is in front of her building waiting for the car, and already has past 20 minutes. The taxi driver is late. After 5 minute the taxi car arrives, and drove Lina to the birthday party. Lina is not satisfied with the service and she decides to report the taxi driver for the delay. Using the code of the taxi she sends report to the system.

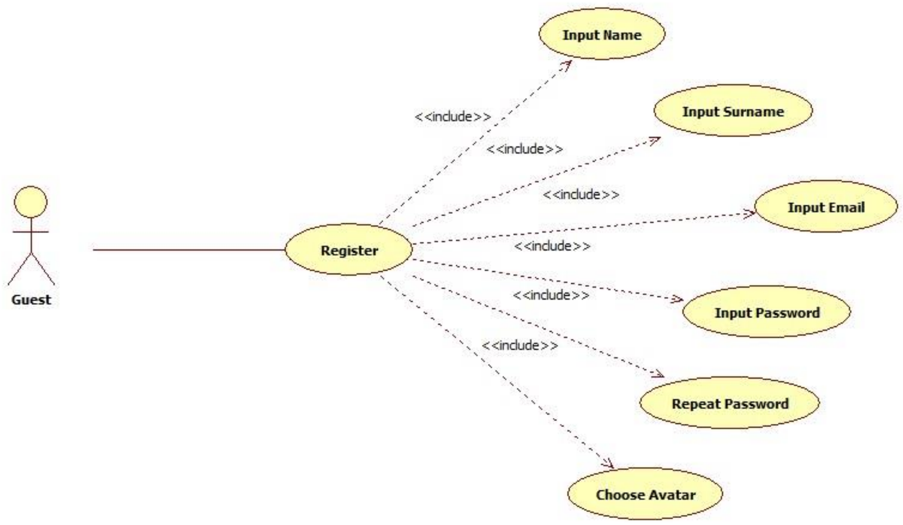
Scenario 3**:** Tom is a taxi driver and he uses myTaxiService app for better performance of his job. Right now, Tom has received 2 requests for a taxi in the zone that he is working, and one reservation 3 hours from now. He accepts the reservation because he is free at that time, and he also accepts just one of the requests, while he rejects the other. After accepting, the system sends notifications to the users.

5. UML Models

5.1 Use Case Diagrams

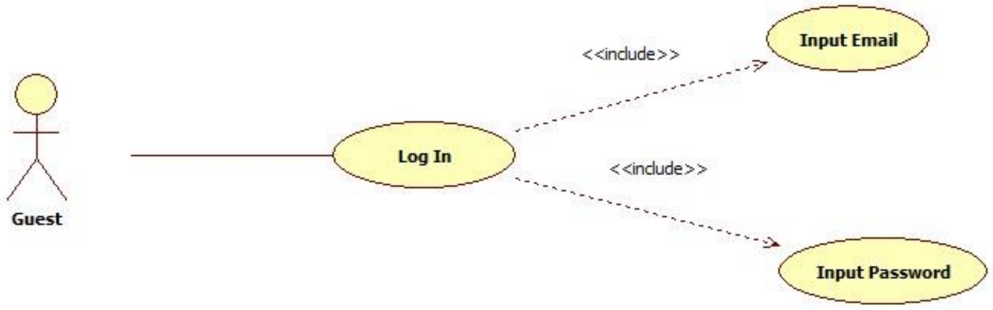
After stating all the desired features, goals and requirements, and describing some possible scenarios we can identify some use cases. The diagrams are shown and described below.

5.1.1 Register (accessing the application)



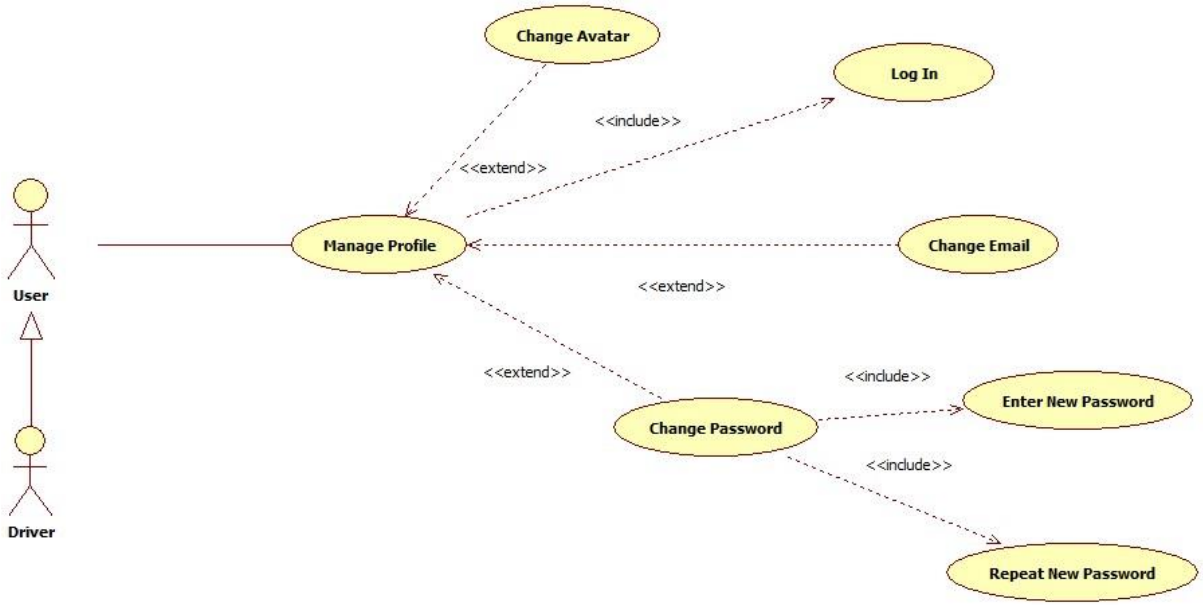
Name	Register
Actors	Guest
Entry Conditions	The guest has never registered to the app
Event Flow	• Guest navigates to the myTaxiService homepage containing a form asking him to enter basic information, name, surname, email, password and avatar which is optional • Upon the form submission system checks if the entered data is valid i.e. if all the fields are filled and email is not already in use. If the entered data is valid, the user will get confirmation email with which he will be automatically logged in to the system and redirected to his personal home page
Exit Conditions	The information about the new user is correctly stored, a welcome email is sent and the user can use all functions of myTaxiService
Exceptions	If the data provided by the user is not valid, or all the required information are not filled an appropriate message is displayed and the guest is asked to fill the missing fields or correct specific fields

5.1.2 Log In



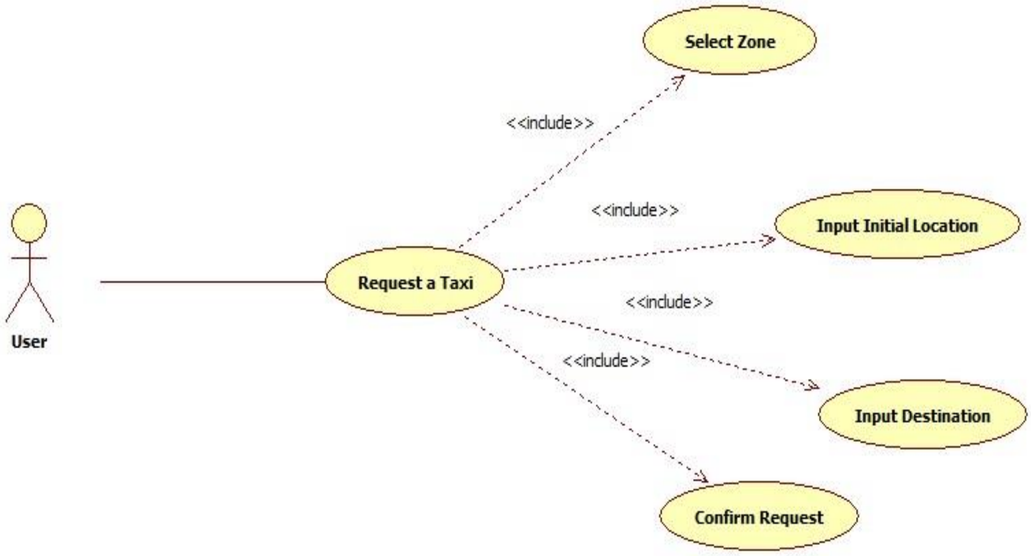
Name	Log In
Actors	Guest
Entry Conditions	The guest has already registered in the system and knows his email and password
Event Flow	• After navigating to myTaxiService homepage containing a form asking him to enter his email and password • Upon the form submission system checks if the username and password match an existing user If the entered information is valid, the user is logged in to the system and redirected to his personal page
Exit Conditions	The user is granted access to all of the application`s services
Exceptions	If he information the user entered wrong password or email, an appropriate message is displayed and the guest is asked to enter his email and password again

5.1.3 Manage Profile



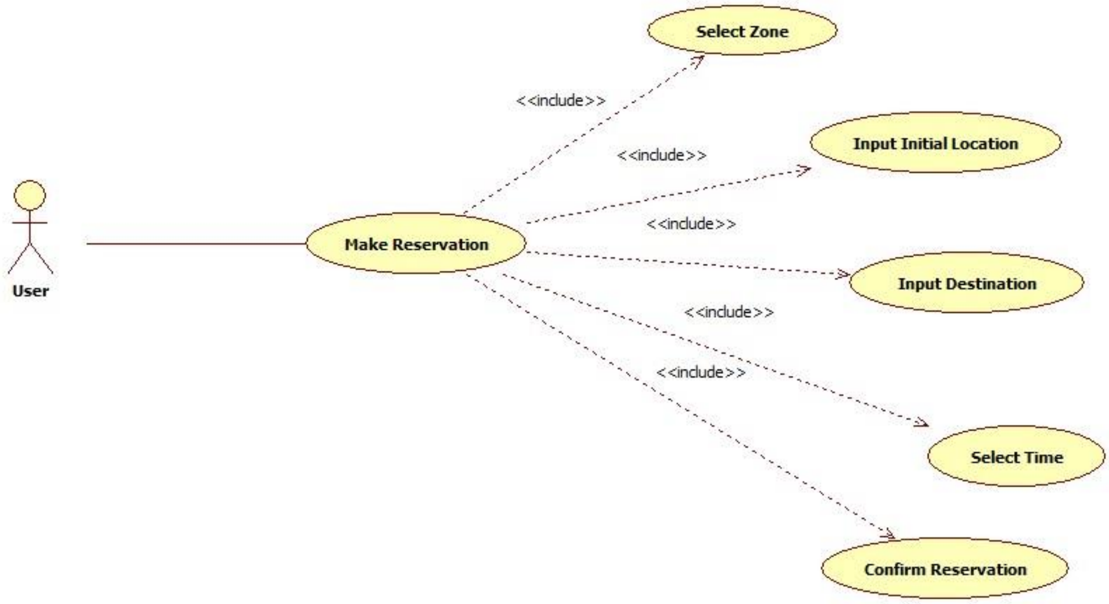
Name	Manage Profile
Actors	User
Entry Conditions	The user is logged in
Event Flow	• User chooses the Manage Profile option from the home screen • User is redirected to a Manage profile screen where he/she can change the following information: ○ email ○ password ○ image (optional) User submits the filled form
Exit Conditions	The new user’s data are correctly stored and the account is successfully edited
Exceptions	If the user submits the form leaving any field empty, an appropriate message will be displayed and the user will be asked to fill out the missing information

5.1.4 Request a Taxi



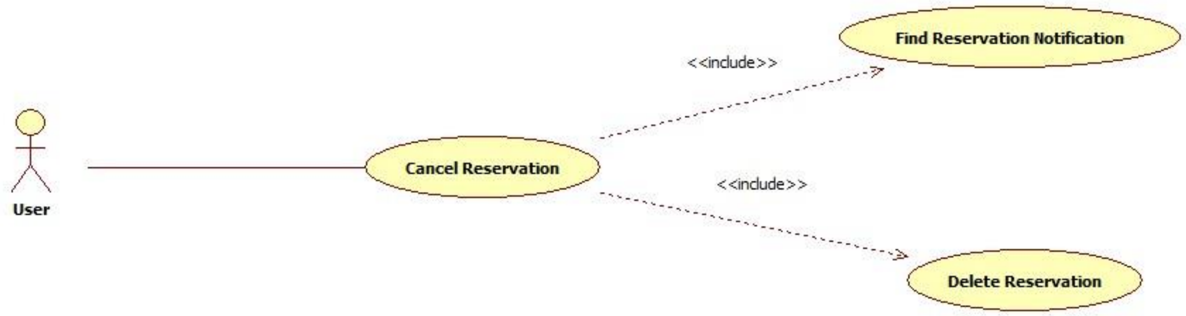
Name	Request a Taxi
Actors	User
Entry Conditions	The user is logged in
Event Flow	• User chooses the *Request taxi* option from the home screen • User is redirected to a Request screen where he/she can select desired location where to catch a taxi, and the final destination, where he/she wants to go After selecting locations user submits request simply by clicking request button
Exit Conditions	The request is sent to myTaxiService system, confirmation with code of incoming taxi and waiting time is sent back to user
Exceptions	If the user doesn't select location, an appropriate message will be displayed and the user will be asked to choose starting and ending point of the ride

5.1.5 Make Reservation



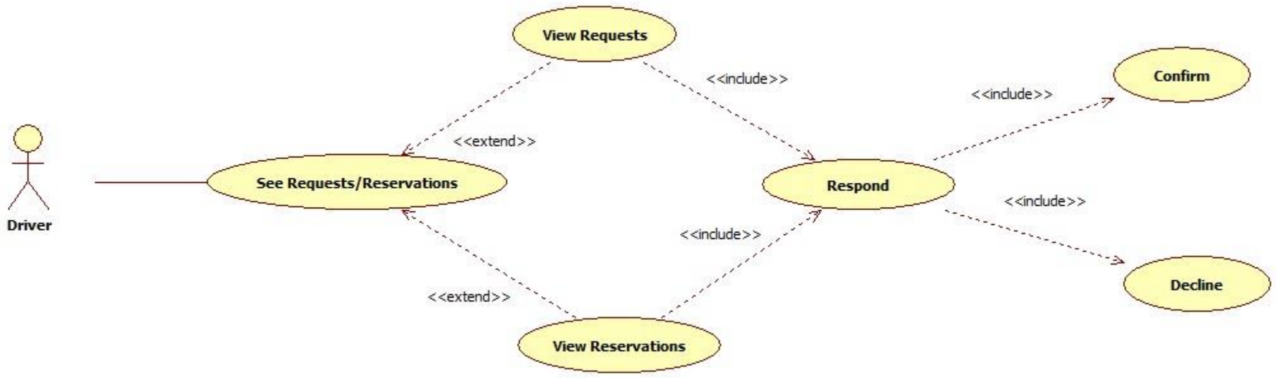
Name	**Make Reservation**
Actors	User
Entry Conditions	The user is logged in
Event Flow	• User chooses the *Make Reservation taxi* option from the home screen • User is redirected to a Make Reservation screen where he/she can select desired location where to catch a taxi, the final destination where he/she wants to go, and the time of the ride After selecting locations and time user submits reservation simply by clicking Make Reservation button
Exit Conditions	The request of reservation is send to myTaxiService system. User receives confirmation of the reservation immediately. And 10 minutes before the ride he/she receives message with the information of the assigned vehicle
Exceptions	• If user doesn't fill all the fields the warning message will be displayed and the user will be asked to fill the missing information If the doesn't reserve a taxi at least 2 hours before the ride the appropriate message will be displayed and the user will be asked to set new time

5.1.6 Cancel Reservation



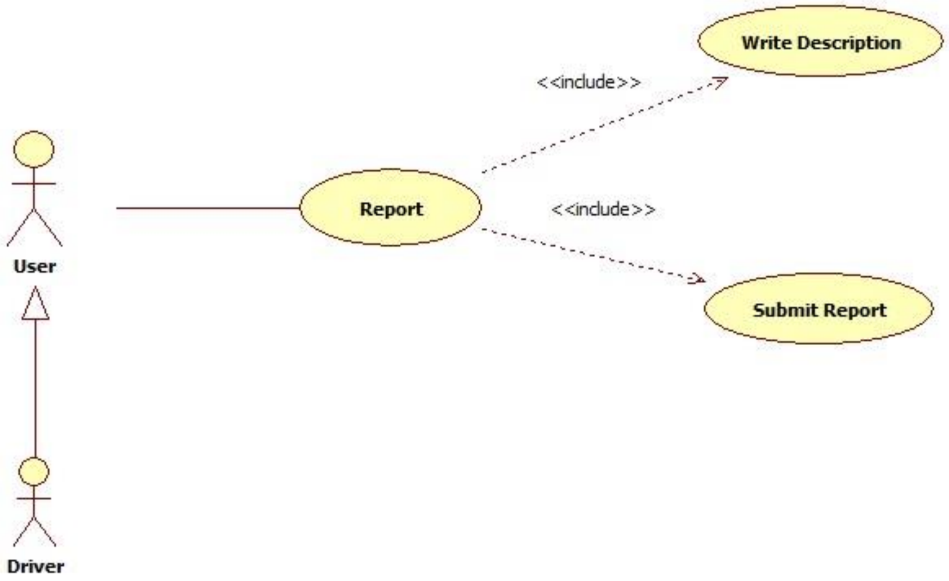
Name	Cancel Reservation
Actors	User
Entry Conditions	The user is signed in and reserved a ride
Event Flow	• User opens the *Notifications* from drop down menu of the home screen User is redirected to a Notifications screen where he/she can delete the active requests or reservations for a ride
Exit Conditions	The cancel reservation is sent to myTaxiService system, after that the system responds with confirmation pop-up
Exceptions	User cancels reservation while riding in a taxi. The appropriate error message will be displayed on users screen

5.1.7 Driver`s Respond to a Request or Reservation



Name	Driver`s Respond to a Request or Reservation
Actors	Driver
Entry Conditions	Driver is logged in, and reservation/request is already assigned to the driver
Event Flow	• Taxi driver receives notification for a request/reservation for a ride Taxi driver accepts or declines the ride by choosing appropriate option, confirm or decline
Exit Conditions	If taxi driver confirmed a ride, details of the ride are showed to him. Otherwise, his state is unchanged and he awaits for new requests/reservations
Exceptions	If driver has some problem with the car and can’t reach destination point in the agreed time he/she can cancel the ride. The system sends a notification to the passenger about the issue and new assigned vehicle

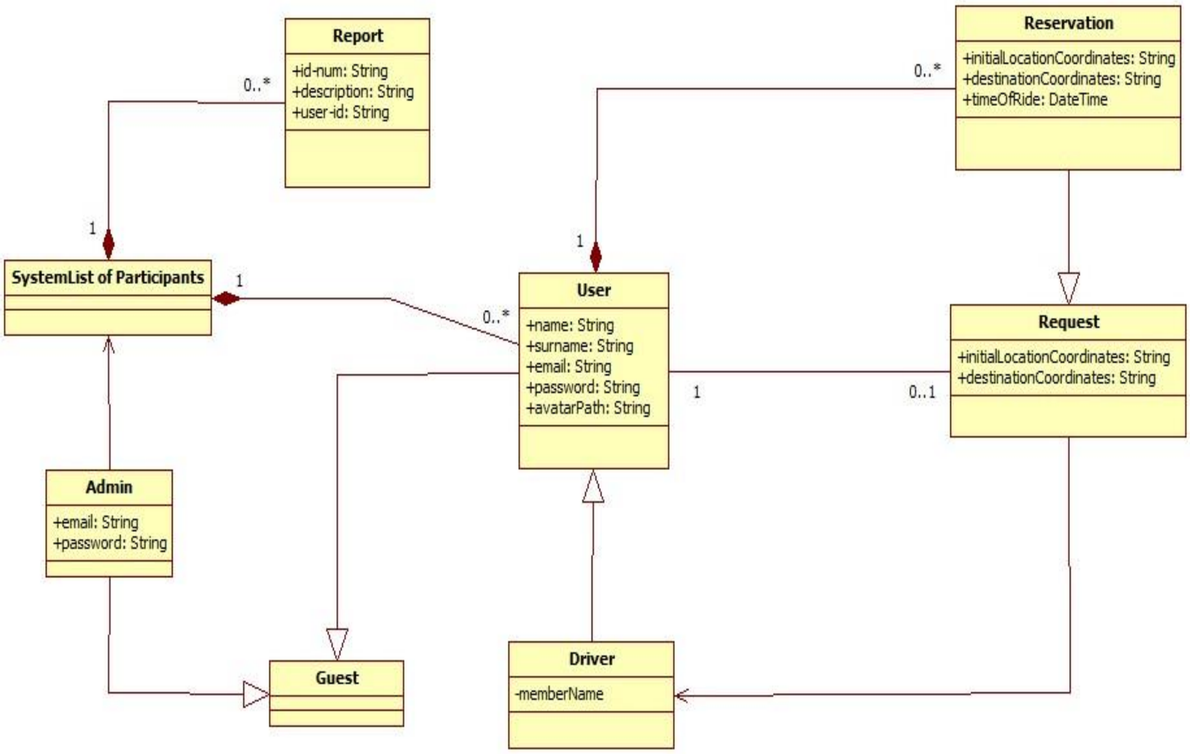
5.1.8 Report a User/Driver



Name	**Report a User/Driver**
Actors	User
Entry Conditions	The user is logged in
Event Flow	- User chooses the *Report* option from the home screen - User is redirected to report screen User writes a report and then submits it to the system
Exit Conditions	The user receives confirmation of sent report
Exceptions	If user did not write the report description, he will receive an error message pointing out about the missing description

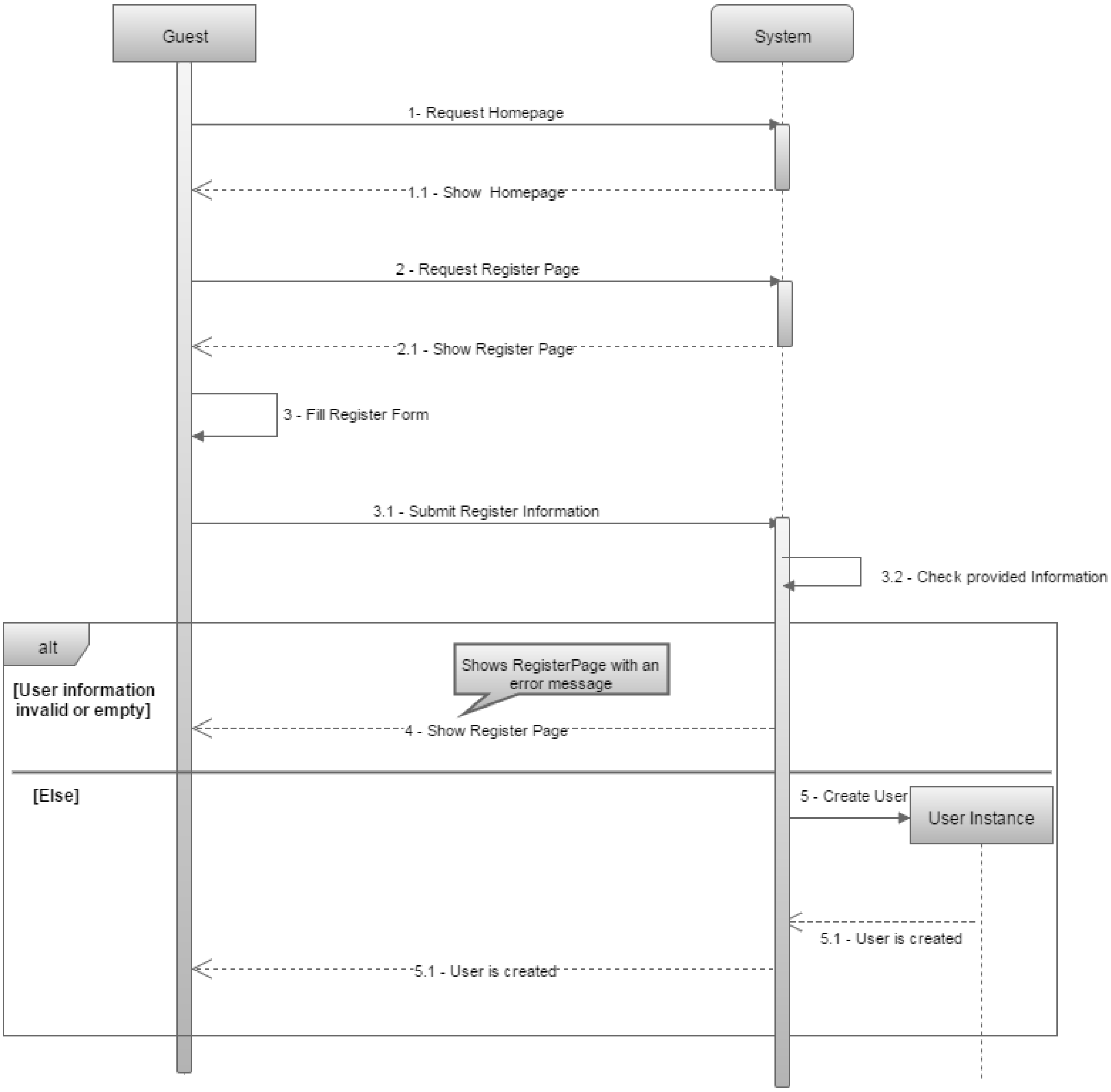
5.2 Class Diagram

Here is presented the class diagram which will be updated during the development process.

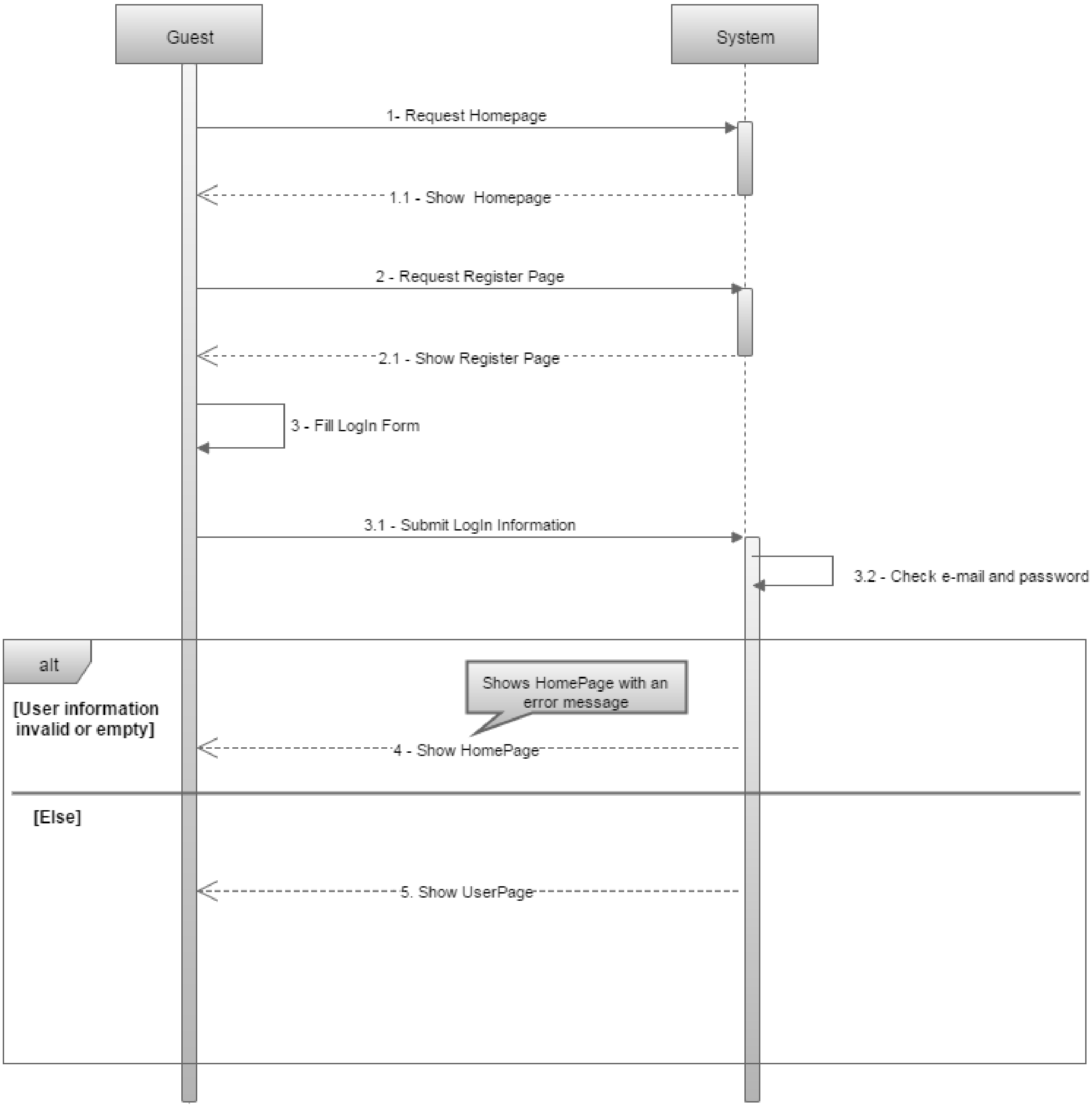


5.3 Sequence Diagrams

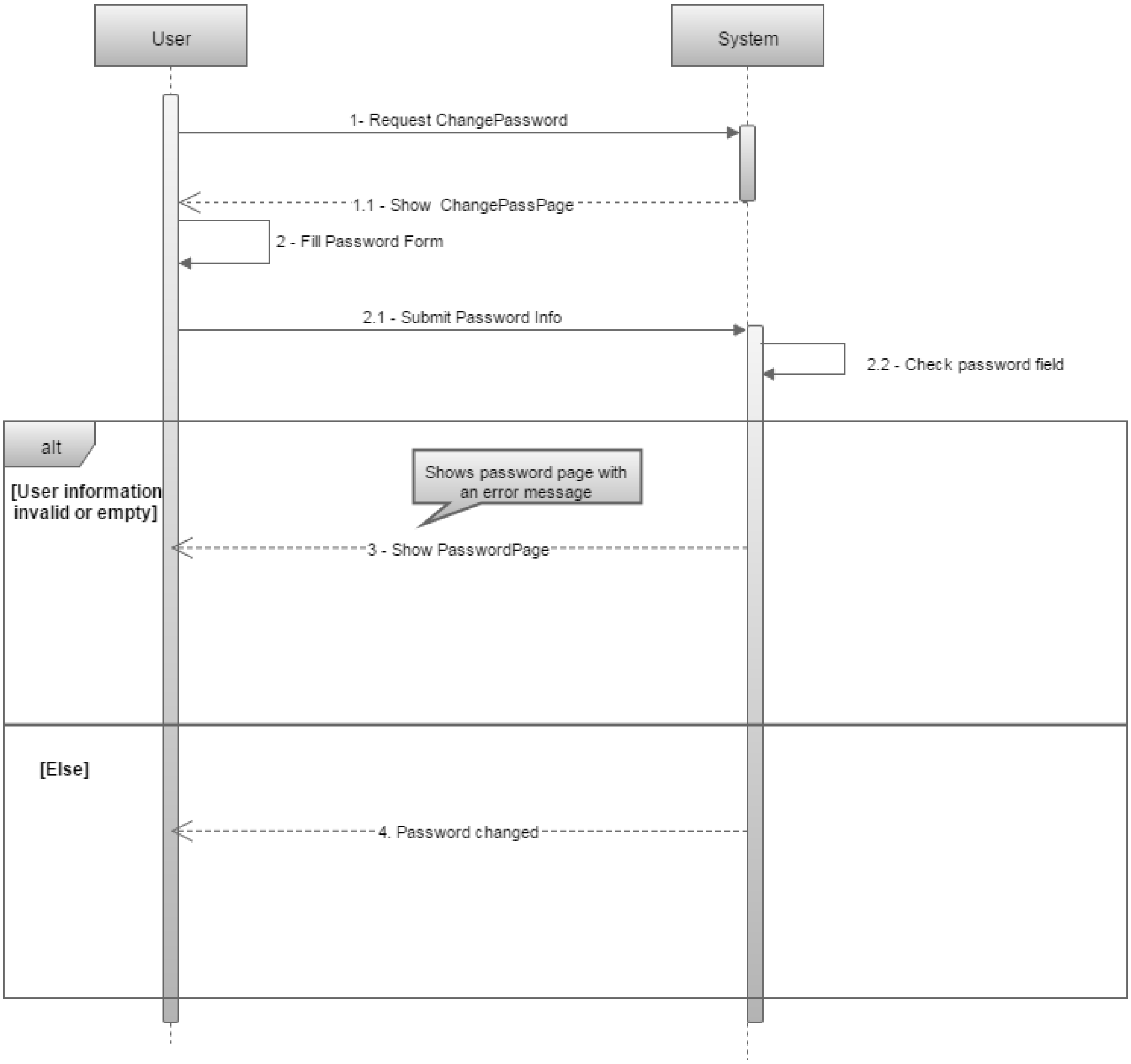
5.3.1 Register



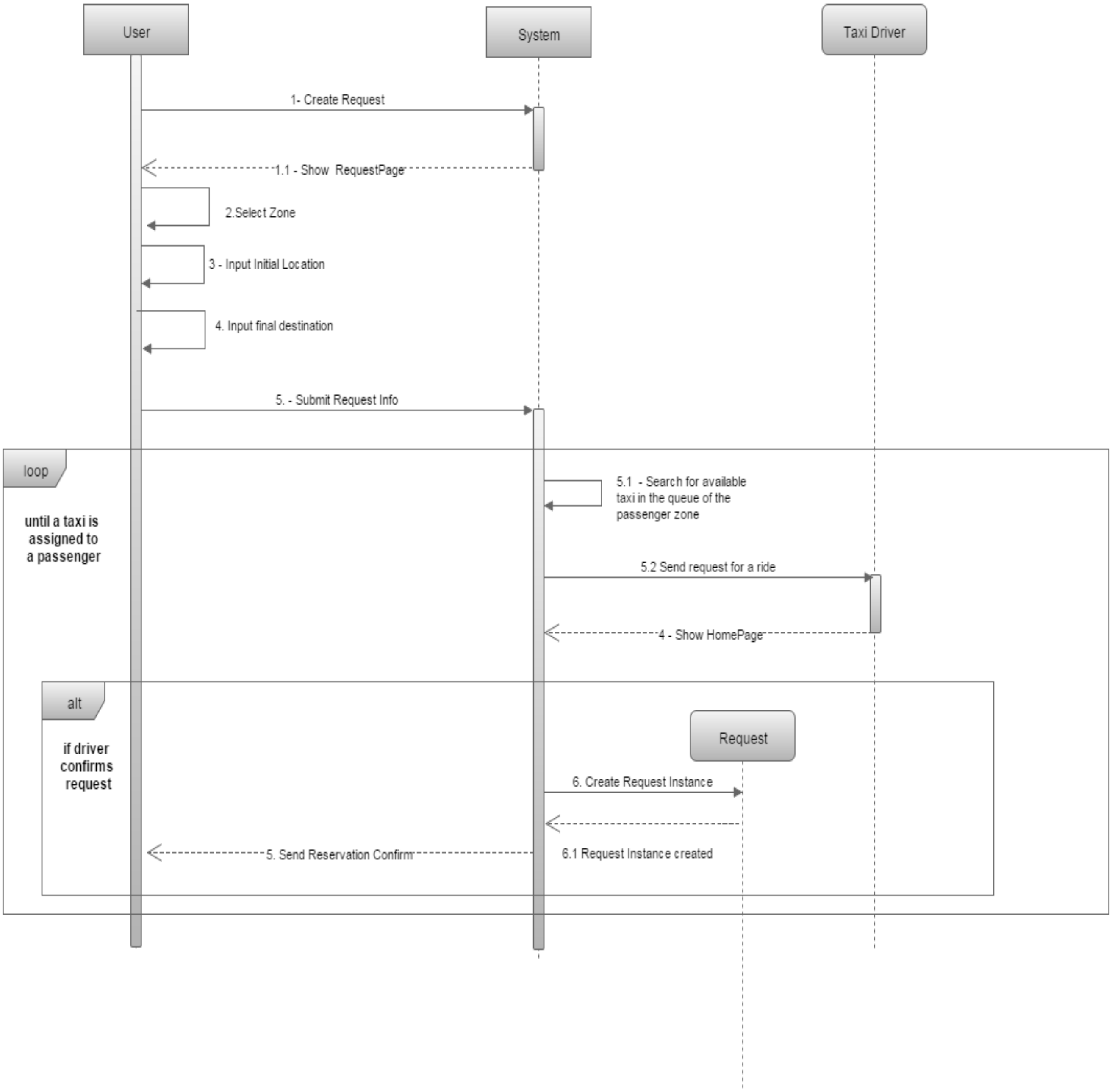
5.3.2 Log In


Guest
System
1- Request Homepage
1.1 - Show Homepage
2 - Request Register Page
2.1 - Show Register Page
3 - Fill LogIn Form
3.1 - Submit LogIn Information
3.2 - Check e-mail and password
alt
[User information invalid or empty]
Shows HomePage with an error message
4 - Show HomePage
[Else]
5. Show UserPage

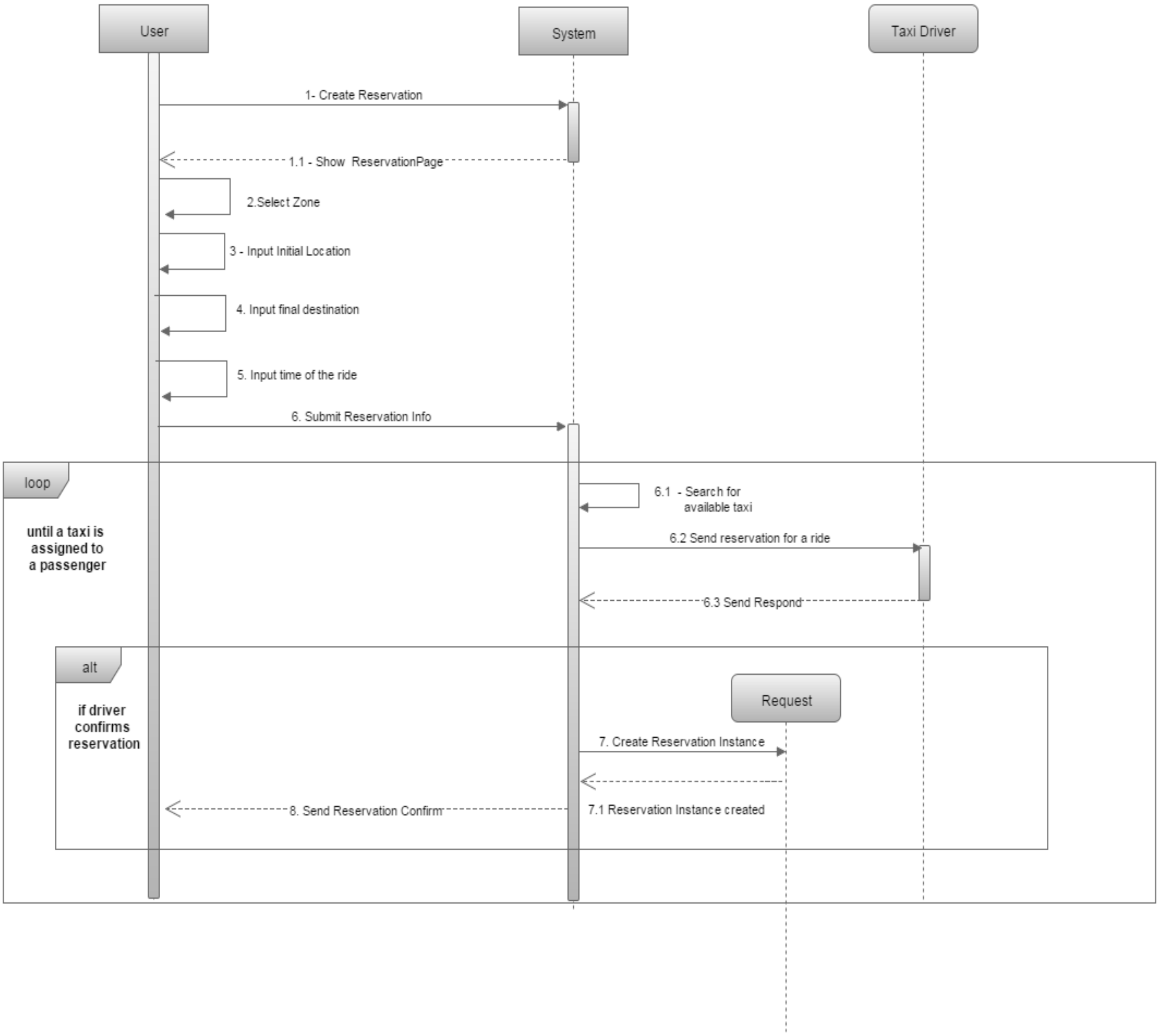
5.3.3 Manage Profile


User
System
1- Request ChangePassword
1.1 - Show ChangePassPage
2 - Fill Password Form
2.1 - Submit Password Info
2.2 - Check password field
alt
[User information invalid or empty]
Shows password page with an error message
3 - Show PasswordPage
[Else]
4. Password changed

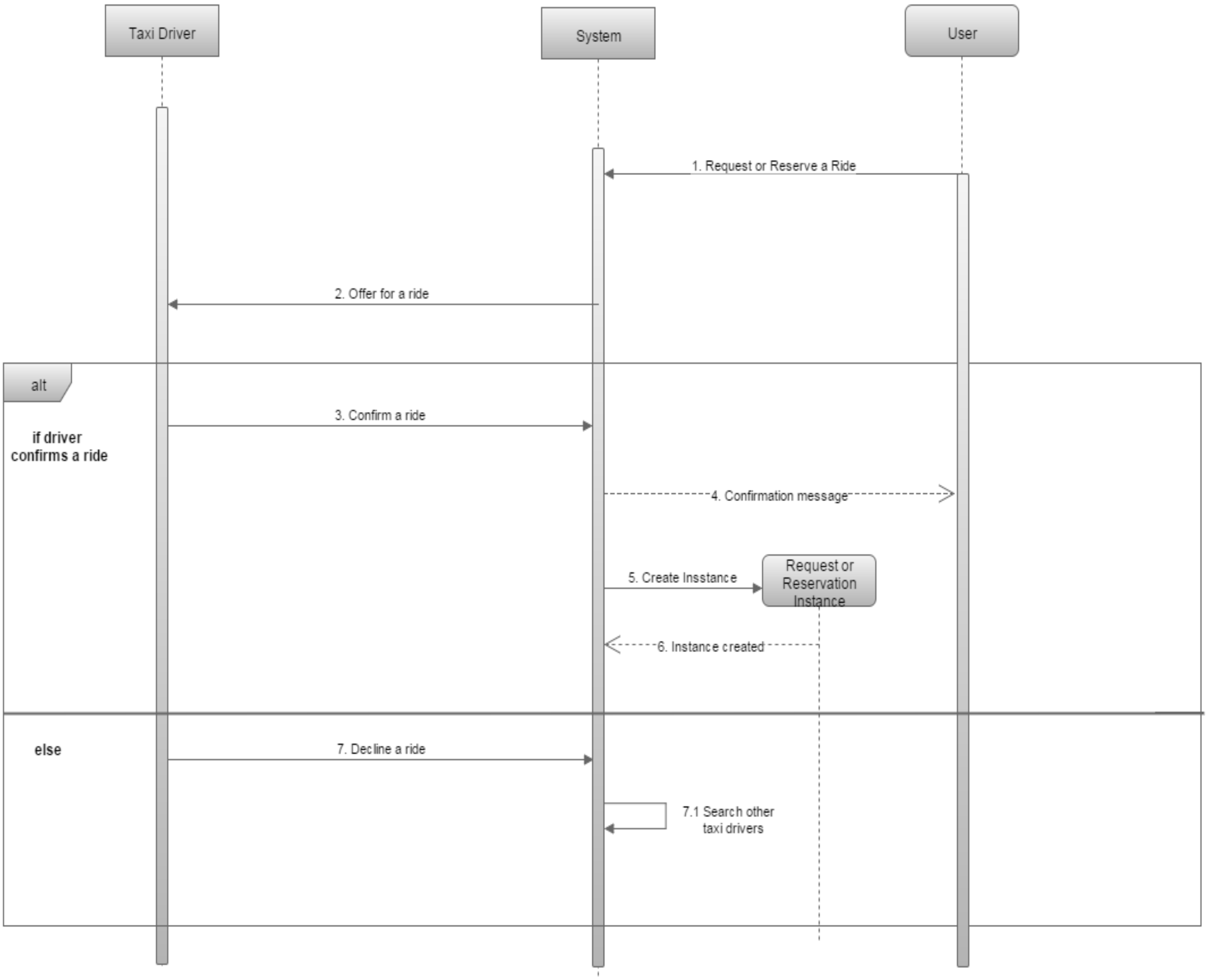
5.3.4 Request a taxi


User
System
Taxi Driver
1- Create Request
1.1 - Show RequestPage
2.Select Zone
3 - Input Initial Location
4. Input final destination
5. - Submit Request Info
loop
until a taxi is assigned to a passenger
5.1 - Search for available taxi in the queue of the passenger zone
5.2 Send request for a ride
4 - Show HomePage
alt
if driver confirms request
Request
6. Create Request Instance
6.1 Request Instance created
5. Send Reservation Confirm

5.3.5 Reserve a taxi

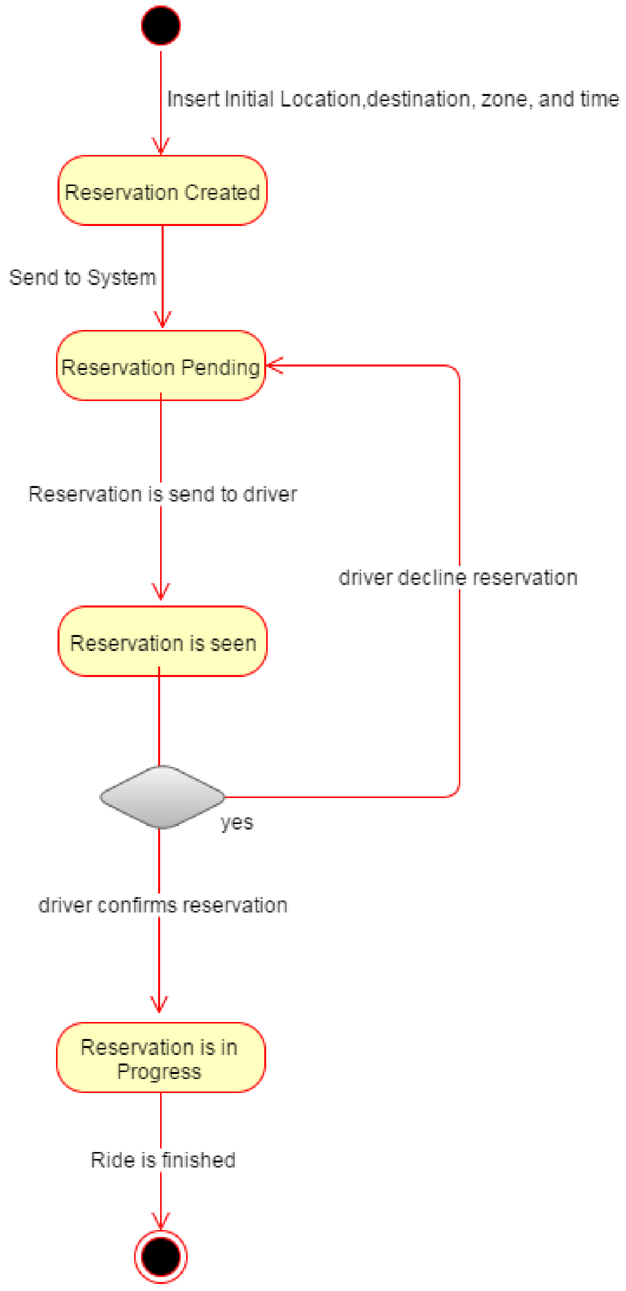

User
System
Taxi Driver
1- Create Reservation
1.1 - Show ReservationPage
2.Select Zone
3 - Input Initial Location
4. Input final destination
5. Input time of the ride
6. Submit Reservation Info
loop
until a taxi is assigned to a passenger
6.1 - Search for available taxi
6.2 Send reservation for a ride
6.3 Send Respond
alt
if driver confirms reservation
Request
7. Create Reservation Instance
7.1 Reservation Instance created
8. Send Reservation Confirm

5.3.6 Confirm - Decline a ride

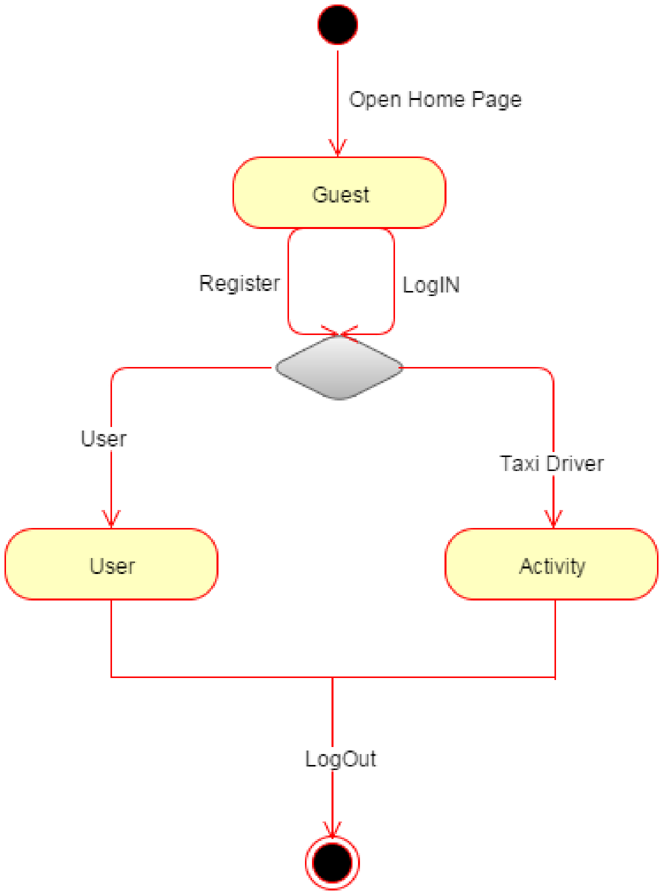


5.4 State Chart Diagrams

5.4.1 Reservation Lifecycle


Insert Initial Location,destination, zone, and time
Reservation Created
Send to System
Reservation Pending
Reservation is send to driver
driver decline reservation
Reservation is seen
yes
driver confirms reservation
Reservation is in Progress
Ride is finished

5.4.2 Visitor Lifecycle



6. Alloy Modeling

6.1 Alloy code

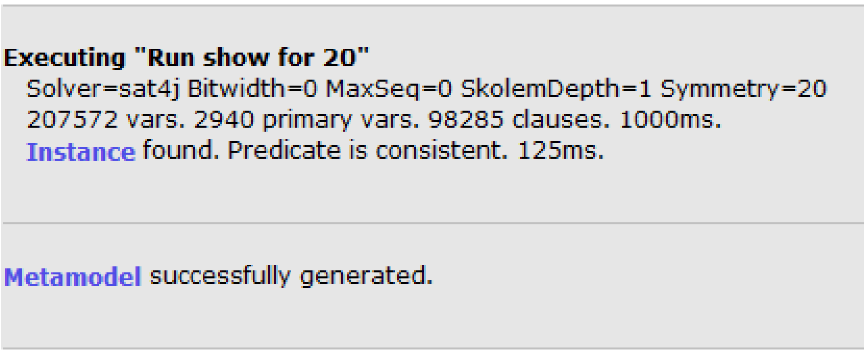
```
module language/myTaxiService

//SIGNATURES

sig Guest{}

sig User extends Guest {

        reservation: lone Reservation,

        requests: set Request

}

sig Driver extends User{

        car: one TaxiCar

}

sig TaxiCar{

        currentDriver: lone Driver

}

sig Reservation{

        driver: one Driver,

        passenger: one User

}

sig Request extends Reservation{}

sig Zone{

        drivers: set Driver

}
```

//FACTS

```
fact noSameDriverPerCar{

        no d:Driver | some t1,t2: TaxiCar |

        t1!=t2 and d in t1.currentDriver and d in t2.currentDriver

}

fact connectionCarDriver{

        all t:TaxiCar | all d:Driver | t in d.car => t.currentDriver=d

}

fact differentCarsTwoDrivers{

        no t:TaxiCar | some d1,d2: Driver | d1!=d2 and d1.car=t and d2.car=t

}

fact differentCarsTwoDrivers1{

        all d:Driver | all t:TaxiCar | d in t.currentDriver => t in d.car

}

fact NoDriverAPassenger{

        no u:Driver | some res:Reservation | u in res.passenger

}

fact diffDriversPerReservation{

        all res1, res2:Reservation | some d1,d2:Driver | (d1 in res1.driver and d2 in res2.driver) => (d1!=d2)

}

fact diffUsersPerReservation{

        all res1,res2:Reservation | some u1,u2:User | (u1 in res1.passenger and u2 in res2.passenger) =>
(u1!=u2)

}

fact oneZonePerDriver{

        all d:Driver | one z:Zone | d in z.drivers

}
```

```
//PREDICATES

pred show{

#Guest=8

#User=7

#Driver=6

#Request=2

#Reservation=3

#Zone=2

#TaxiCar=7

}

run show for 20
```

6.1.1 Final result of the analysis:

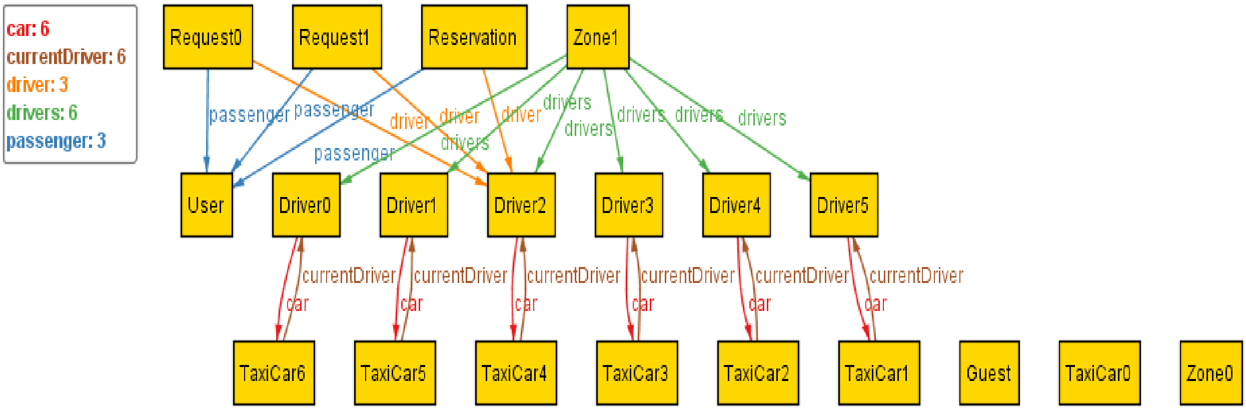


```
Executing "Run show for 20"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   207572 vars. 2940 primary vars. 98285 clauses. 1000ms.
   Instance found. Predicate is consistent. 125ms.

Metamodel successfully generated.
```

6.2 Alloy Worlds

6.2.1 General World

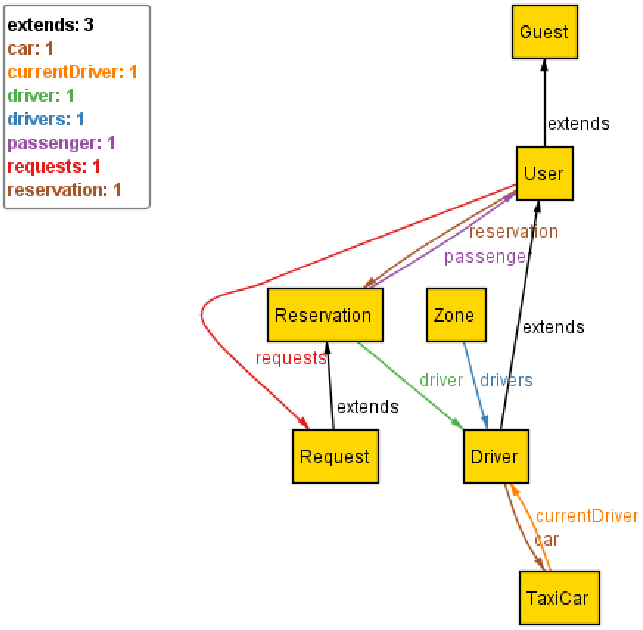
The following world is a general world generated with the analyzer. It has been generated using the predicate show.



6.2.2 Alloy Metamodel

The following metamodel describes that User is an extension of Guest, also Driver is an extension of User, meaning extension of Guest. Further it also shows that Request is an extension of Reservation. In this model are illustrated the main constraints of the application such as:

- A driver belongs only to one zone
- A request belongs only to one user
- A reservation belongs only to one user
- A user belongs only to one reservation
- A reservation belongs to one driver
- A taxi car belongs to one driver
- A driver have only one taxi car



7. Used tools

The tools we used to create the RASD document are:

- ***Microsoft Office Word***: to redact and to format this document

- ***Axure RP Pro 7.0***: to create the sketches for the interface of the web version of the application

- ***Justinmind Prototyper 6.9.1***: to create the sketches for the interface of the mobile version of the application

- ***Draw IO*** and ***StarUML***: to create the State Charts, the Class Diagram, the Sequence Diagrams and the Use Case Diagram

- ***Alloy Analizer 4.2***: to prove the consistency of our model