Politecnico di Milano

A.A. 2015-2016

Software Engineering 2: "myTaxiService"

Requirements Analysis and Specifications

Document

Version 2.0

Milica Jovanovic (mat. 835953), Pavle Vidanovic (mat. 854472)

05 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 communicate what the software should do and identify the capabilities and characteristics of the system being developed, as well as the constraints it should respect. This document is meant for everyone included in the production of the software.

1.2 Scope

The aim of this project is to develop and implement myTaxiService, an application similar to Uber, which makes the process of assigning an available taxi vehicle to possible passengers.

The developed system should allow new users to register. Users, once logged in, should be able to:

- request a taxi
- reserve a taxi
- cancel a ride
- check taxi availability around him
- receive a confirmation with information about the assigned vehicle and ETA once taxi is requested
- create/maintain user profile
- report a taxi driver

The developed system should allow new taxi drivers to register. Drivers, once logged in, should be able to:

- inform the system about their availability
- confirm/decline that they are going to take care of a certain call
- create/maintain taxi driver profile
- report a passenger

The system should keep information about new arrived requests, as well as the confirmed rides. A ride should have and id number, information about the passenger that requested the ride, as well as the code of the assigned vehicle and ETA. System should also keep information about taxi queues connected to particular zone of the city and ensure fair management of the queues. Developed system should keep information about the list of reservations made by passengers, such as id number of the reservation, information about the passenger that made the reservation and the time of reservation and time of the ride.

1.3 Glossary

The following are the definitions of some commonly used phrases throughout the document:

ETA   Estimated Time of Arrival, approximated time of arrival of taxi vehicle to destination

Reservation	Passenger request for a vehicle at least 2 hours before the ride
Request	Passenger filled form for immediate ride
Reservation Conformation	Notification sent to the user about the confirmed reservation
Ride Conformation	Notification sent to the user about the confirmed ride with information of the ride
Report	Short description of problem that user/driver stumped into
User	A person already registered and logged into the system
Guest	A person accessing a system that has either never registered or hasn't logged in yet. Guest has only two available options, to log in or to register for the first time
Taxi driver	A person already register and logged into the system as a driver
GPS	Global Positioning System
API	Application Programming Interface.

1.4   Reference Documents

- IEEE Std 830-1998 Recommender Practice for Software Requirements Specifications
- Specification Document: myTaxiService Project AA 2015-2016.pdf

1.5   Document Overview

The document is essentially structured in four parts:

- Chapter 1: Introduction, gives description of document and some basic information about the software
- Chapter 2: Overall Description, gives an overview of the main functionalities of the software to be with constraints, as well as the hardware limitations.
- Chapter 3: Specific Requirements, body of the document that describes in more detail functionalities, possible scenarios and use cases.
- Chapter 4: Appendix, Alloy simulation of the previous UML diagrams.

2 Overall Description

2.1 Product perspective

myTaxiService is mobile web application that provides user with services described in section 1.2. System will consist of two applications and server between them. 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. Considering *myTaxyService* is a mobile web application, it 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* that provided project specification and expect 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,* person working for 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, given of the ubiquitous nature of internet and social media, it is assumed that people using our software do have the basic web browsing skills.

2.4 Actors identifying

Four possible actors interacting with our system are the following:

- *Guest* person accessing a system that has either never registered of hasn't logged in yet. Guest can only access the initial page from where he has only two available options, to log in or to sign up for the first time
- *User* a person already registered and logged into the system. User can use all of the features offered by the passenger application
- *Taxi driver* a person already registered and logged into the system. User can use all of the features offered by the taxi driver application
- *Admin* a person responsible for handling reports on users 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:

- [G1] registering new user
- [G2] login to existing user's account
- [G3] managing user's profile
- [G4] requesting a taxi

- [G5] reserving a taxi
- [G6] canceling a ride
- [G7] checking taxi availability around user
- [G8] reporting a problem caused by passenger or taxi driver
- [G9] confirming/declining a ride(taxi driver)

2.6   Domain properties

It is supposed that these conditions hold in the analyzed world:

- the passenger needs a ride to specific location
- the details of the ride provided by the passenger are accurate
- money exchange between the passenger and the taxi driver is made independently from the myTaxiService system
- distinction between the zones are clearly defined

2.7   Constraints

2.7.1   Regulatory policies

myTaxiService application will not take advantage of users personal information and will respect the privacy policy. User will be notified about it.

2.7.2   Hardware limitation

User must have access to Internet and own a device with a web browser and GPS service.

2.7.3   Interfaces to other applications

myTaxiService application is integrated with Google Maps API to access their maps, Google Places API to suggest street addresses to the user and email service in order to make authentication.

2.7.4   Parallel operation

myTaxiService will support parallel access to the applications database in a transparent way.

2.8   Assumptions

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

- user have only one account
- user provides accurate information
- if users location is not available, the application will show a screen with an option of typing your current address
- there is a Terms & Conditions section to indicate clearly the usage of the application, which if not followed will result in account deactivation
- we assume that Google Maps service will calculate location used by myTaxiService accurately
- we assume that taxi driver will respect the ETA, otherwise they could be banned from the system
- if a taxi driver has an unexpected issue, the user will be automatically notified by the system and a new vehicle will be assigned to him with new ETA

2.9 Future possible implementation

- another way of payment will be added as an options for users to pay online
- an option for users to rate the drivers
- Facebook authentication could be added as a way of registering
- application could be updated so it works more efficiently
- taxi sharing options could be added to the system

3   Specific Requirements

3.1   External Interface Requirements

3.1.1   User Interfaces

In this section mockups of myTaxiService application will be enclosed which will mimic the general idea of the software system:

In figure 1. Initial page is shown, which offers two possibilities to the guest user. Guest user could create an account if he does not own one or he could sign in with his credentials.

In figure 2. Sign Up page is shown, which provides user with input fields to add his account information. After the fields are filled user can confirm by clicking the Ok button.

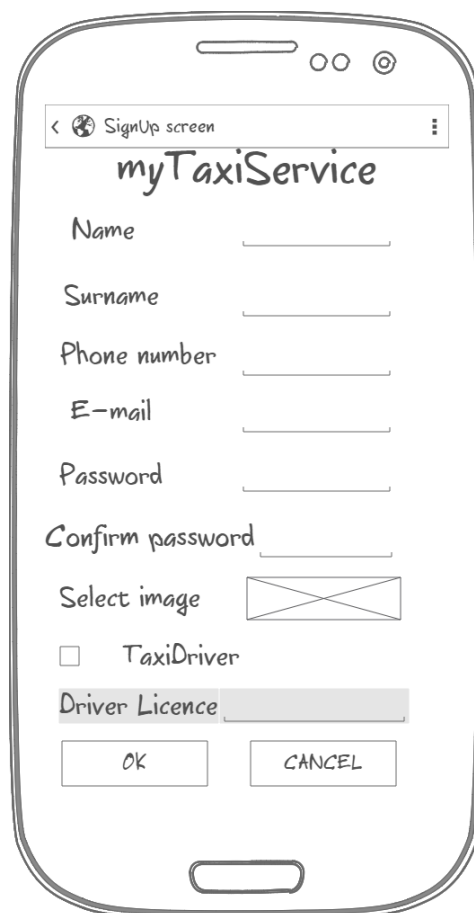Figure 1. Initial page   Figure 2. Sign Up

In figure 3. Sign In page is shown, which provides user with input fields to add credentials. After the fields are filled user can sign in by clicking the button.

In figure 4. Passenger Home page is shown, which shows user's location and a drop down menu with different options. From this page user can choose option to request a taxi, reserve a taxi, change his profile, check available taxis around him, see received notifications, go to Help section or simply log out from his account.
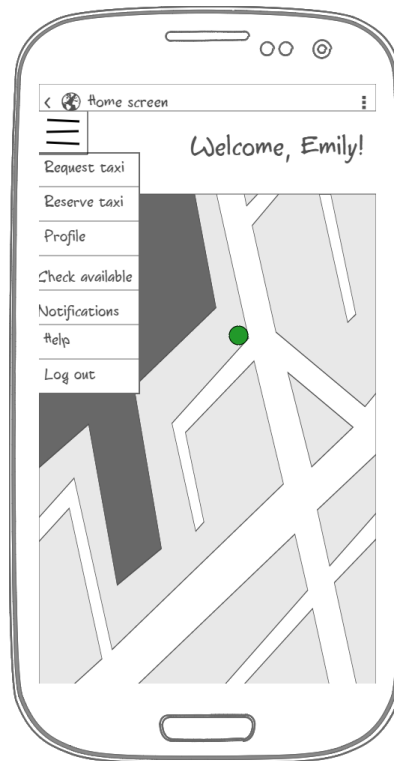

Figure 3. Sign In

Figure 4. Passenger Home page

In figure 5. Request page is shown, which provides user with possibility to request a vehicle from his current location showed on the map.

In figure 6. Reservation page is shown, which provides user with possibility to reserve a vehicle from a certain source address to certain destination address at picked time.
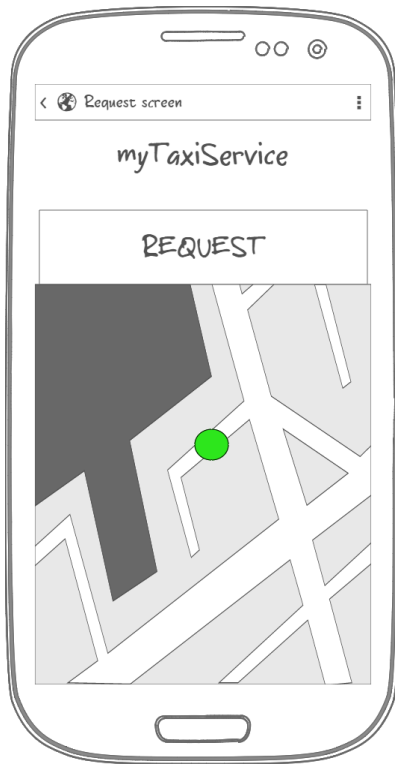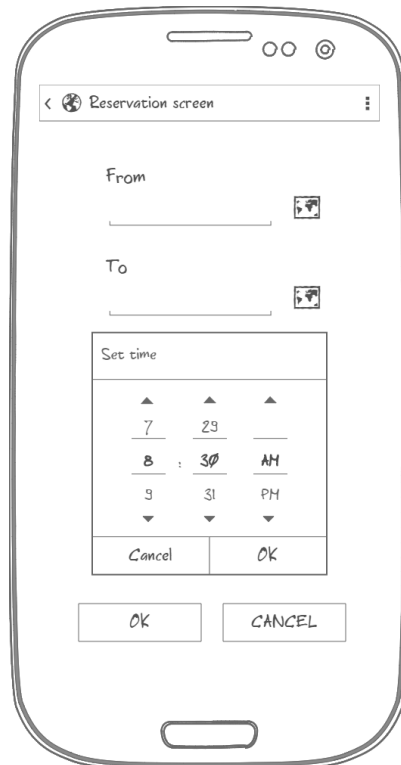

Figure 5. Request page

Figure 6. Reservation page

In figure 7. Driver Home page is shown, which provides driver with possibility to state his availability, manage his profile and check information about his current ride.

In figure 8. Pop up message is shown, it informs a driver that a certain passenger has requested his service. Driver has the possiblility to accept or decline the ride.
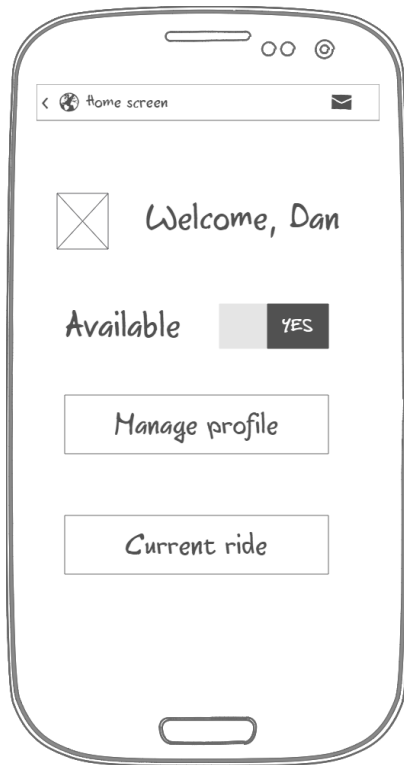

Figure 7. Driver Home page

Figure 8. Pop up message

In figure 9. Navigation page is shown, which provides the driver with the route from his location to the source address of the requested ride.

In figure 10. Taxis available screen is shown, which shows user taxi vehicles available in his zone.
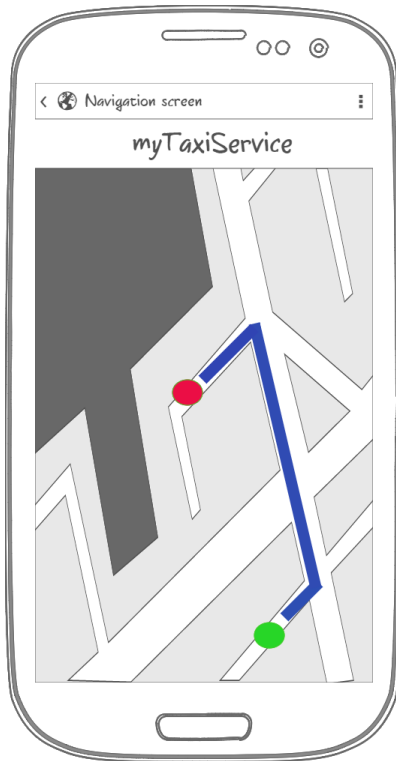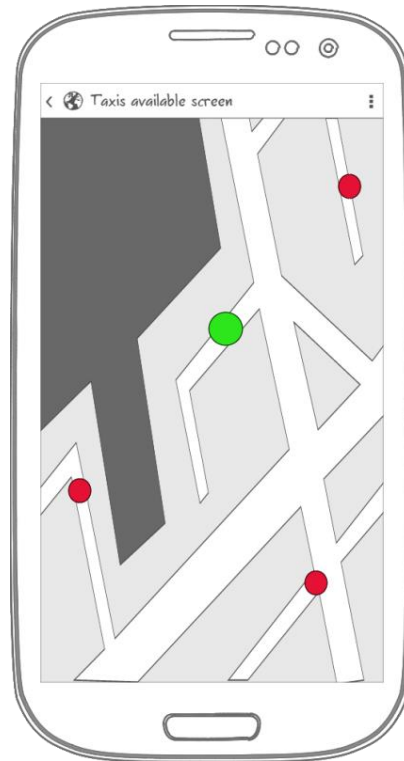

Figure 9. Navigation page

Figure 10. Taxis available screen

In figure 11. Notification screen is shown, which shows list with all requests and reservations. It shows requests that are completed, canceled and the requests that are active.

In figure 12. Reservation details screen is presented, which shows all informations about the ride and enables quick access to user's profile as well as an option of reporting a user/driver.
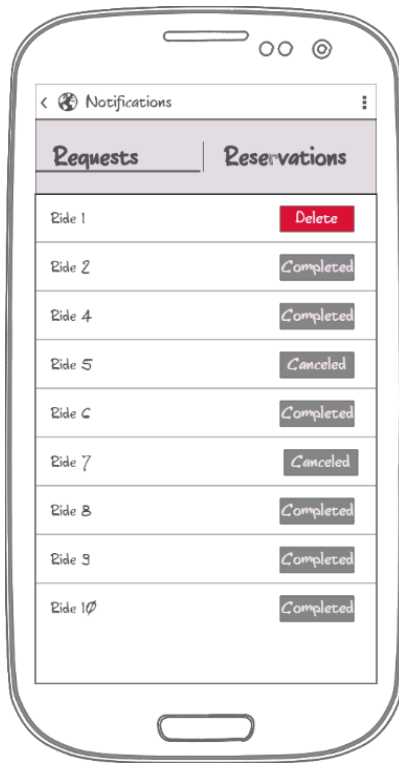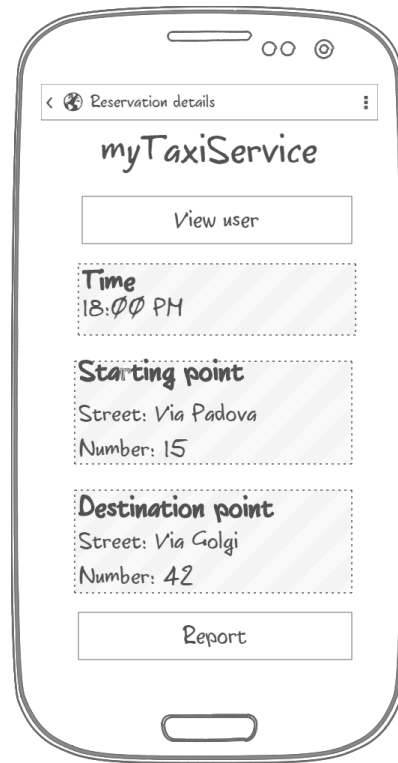
Figure 11. Notification page

Figure 12. Reservation details

In figure 13. Admin Sing In page is shown, which provides admin with input fields to add credentials. After the fields are filled admin can sign in by clicking the button.

In figure 14. Admin list reports page is presented, which shows all reports submitted by users or taxi drivers, inclusive with timestamp of the reports.

In figure 15. Admin report details page is presented, which shows details about the user/taxi driver for particular selected report, as well as the full description of the enclosed report. Admin can resolve the report by banning the user or taxi driver connected to the report.

Figure 13. Admin Sign In page

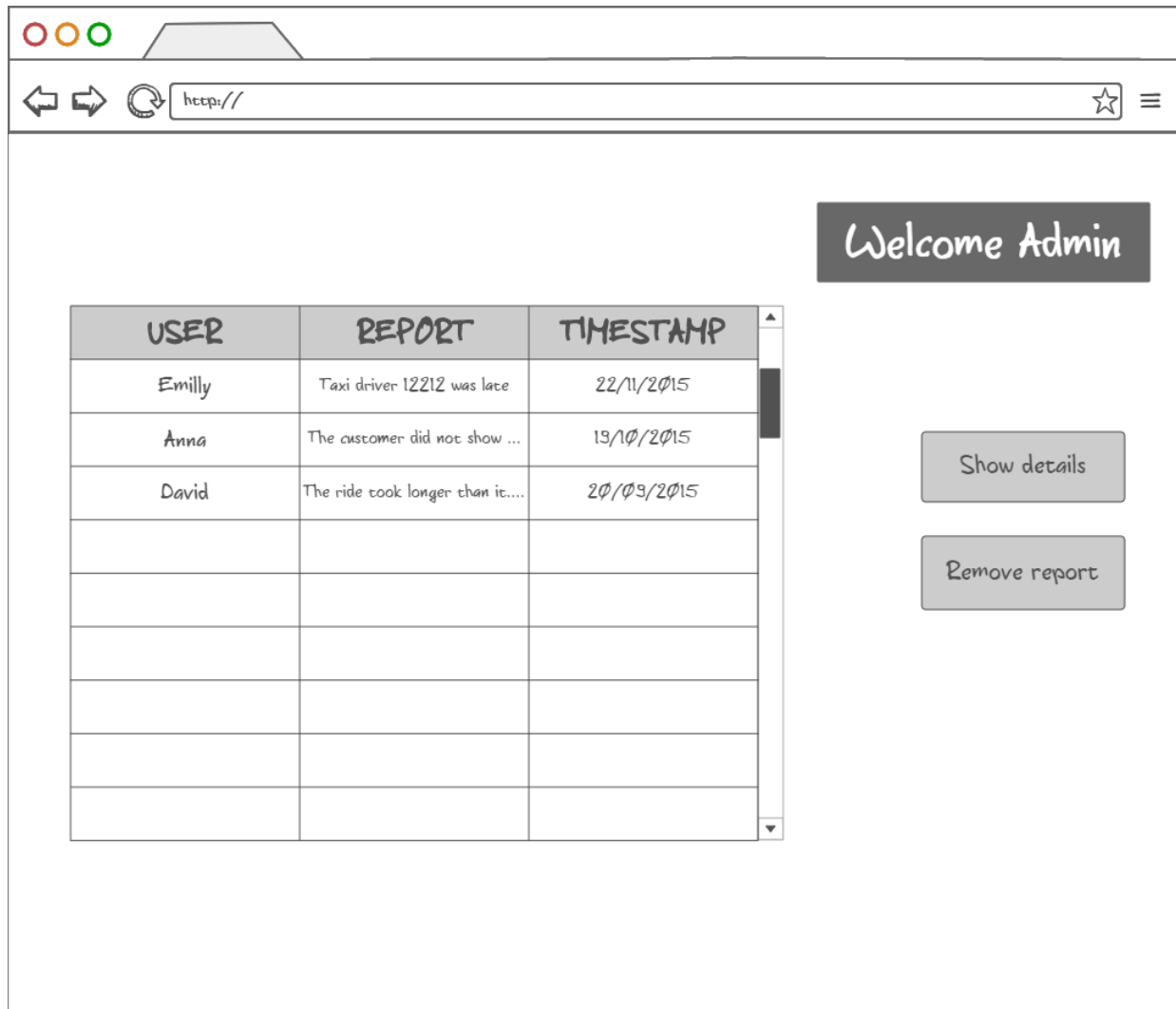| USER	REPORT	TIMESTAMP
Emilly	Taxi driver 12212 was late	22/11/2015
Anna	The customer did not show ...	19/10/2015
David	The ride took longer than it....	20/09/2015

Welcome Admin

Show details

Remove report

Figure 14.Admin list reports page
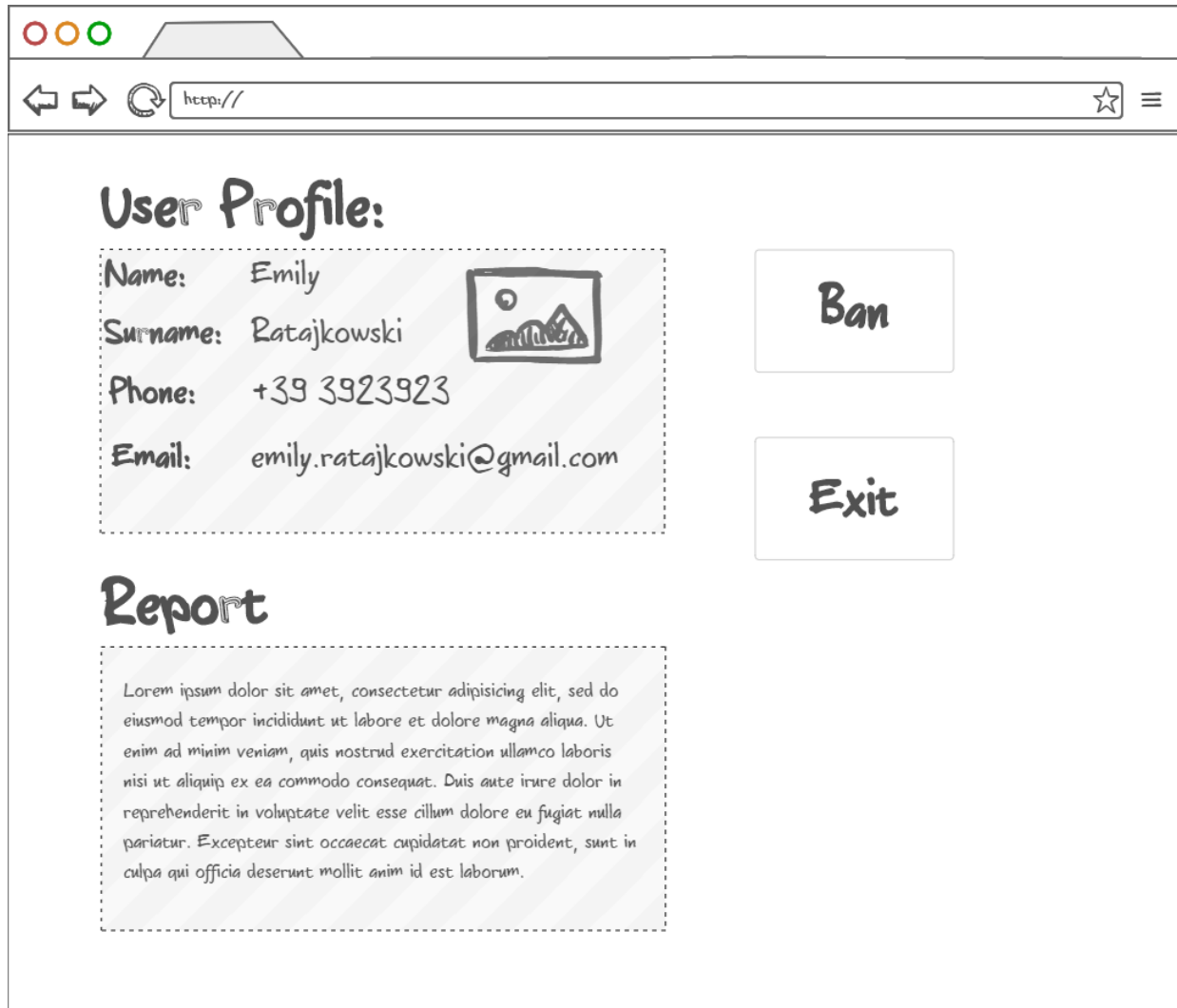
Figure 15. Admin report details page

3.1.2   API interfaces

In 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. Email service is used to allow the user to authenticate himself with his email address, a link will be sent to given email address and by clicking the link user confirms his identity. This service is used as a security measure.

3.1.3   Hardware Interfaces

This project does not support any hardware interfaces.

3.1.4   Software Interfaces

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

3.2   Functional Requirements

[G1] Allow a guest user to become registered user and choose his role in the myTaxiService system

- [R1] Guest must not be already registered to perform sign up process
- [R2] Guest must choose a username or in our case email address that is not already used by another user
- [R3] Visitor can just see Guest screen
- [R5] Guest can only access to registration form in signup screen
- [D1] Email address used for signing up must be formally correct

[G2] Allow user to sign in to application.

- [R1] User must be already registered to success to sign in process
- [R2] User must know his email address and password used during registration to success login
- [R3] Email address and password inserted during sign in process must be correct
- [R4] Wrong credentials will not grant access to user

[G3] Allow user to manage his profile

- [R1] User must be already registered and signed in the application
- [R2] User must confirm updating process

[G4] Allow user to request a taxi

- [R1] User must be already registered and logged in the application
- [R2] User must specify starting location
- [R3] User must confirm request

[G5] Allow user to make a reservation of a taxi

- [R1] User must be already registered and signed in the application
- [R2] User must specify origin of the ride
- [R3] User must specify destination of the ride
- [R4] User must specify meeting time at least 2 hours before the ride

[G6] Allow user to cancel a ride

- [R1] User must be already registered and signed in the application
- [R2] User can view list of all his rides in notification area
- [R3] User must select an active ride that he wants to cancel and confirm

[G7] Allow user to see available taxi vehicles around him

- [R1] User must be already registered and signed in the application
- [R2] User must have location settings enabled

[G8] Allow user/taxi driver to submit a report

- [R1] User/driver must be registered and signed in
- [R2] User/driver must have an active request or reservation
- [R3] User/driver must select a particular request or reservation he wants to report
- [R4] User/driver must write a description for the report

[G9] Allow driver the option to confirm or decline request for a ride

- [R1] Taxi driver must be already registered and signed in the application
- [R2] Taxi driver must set his availability option in order to receive requests

3.3   The world and the machine
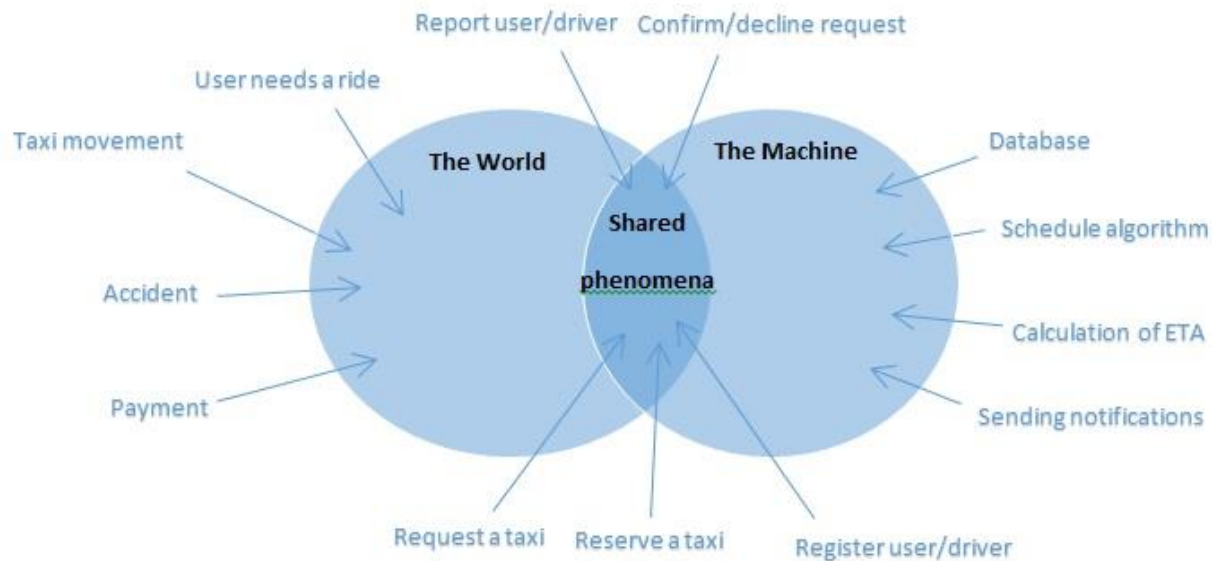
The world - these activities are initiated by the user or driver or by their environment

- *User needs a ride* – is prerequisite situation for the user to request or reserve a taxi
- *Taxi movement* – is happening in user environment
- *Accident* – can be caused another vehicles in traffic in user environment
- *Payment*- it is happening between user and driver without accessing the application

The machine – these activities are used in the system, transparent to users

- *Sending notifications* –those actions are initiated by the system, does not include user actions
- *Calculating ETA* – this calculation is done by the system, necessary information
- *Schedule algorithm* – is used to efficiently find available taxi vehicle from queue of the user's current zone
- *Database* – is located in the server side where all data is stored

Shared – these activities require both user and system actions to be completed

- *Request a taxi* –first user has to specify origin of the ride and to send to system, than system need to find available taxi driver to complete request.
- *Reserve a taxi*- user has to specify meeting time, origin and destination of the ride then the system needs to find a taxi vehicle. After that reservation is completed.
- *Register user/driver*- requires user to fill the form that will be sent to system, which creates user's profile and sends back the confirmation message.
- *Report user/driver*– User/Driver has to write a description, after that the system stores it in database.
- *Confirm decline a ride*- involves user and system. System sends proposal of the ride and then user can choose whether to confirm or decline a ride.

21

3.4   Scenarios

3.4.1   Scenario 1

Emily is running late to a Saturday evening house party. The metro is about to close, so she needs to find a ride quickly. She has heard that there is new well-functioning application called myTaxiService and she wanted to give it a try. She picked up her smart phone and typed in the web address of the app. A login page has appeared and she had to register in order to use it. Emily quickly typed in her personal information and authenticated by email. After that she requested a ride from her current location, and instantaneously received a notification with ETA and code of the incoming taxi. Taxi arrived in few minutes and her dinner hasn't got cold.

3.4.2   Scenario 2

Dan and his friends are planning to go play football in an indoor pitch court. So Dan searched and found free court today from 16 to 17 o'clock so he booked that court. His friends live close to the court or near the metro station. Dan isn't that lucky so he must find a ride that will be secure and that will respect his time schedule. His friends told him about an application myTaxiService that can solve his problems. He easily created an account and found an option of reserving a taxi for the time of the match. He made a reservation 3 hours before the meeting point and received a confirmation of the reservation. 10 minutes before the ride he received a message containing the information about the assigned vehicle. That night his team won the game.

3.4.3   Scenario 3

Economist Jack has lost his job because of the world economic crisis and he found a part-time job at city taxi service. The taxi service uses well-functional cross platform web application. So jack registered as a driver and started to work hardly. He was assigned to work in zone 1, close to the central train station. Being assigned at that area he had a lot of calls from passengers. Around noon when is the rush hour he was really busy so he had to reject some rides when his vehicle was occupied. As soon as he finishes a ride he change his status to available so he could get a contacted by the taxi system about other rides.
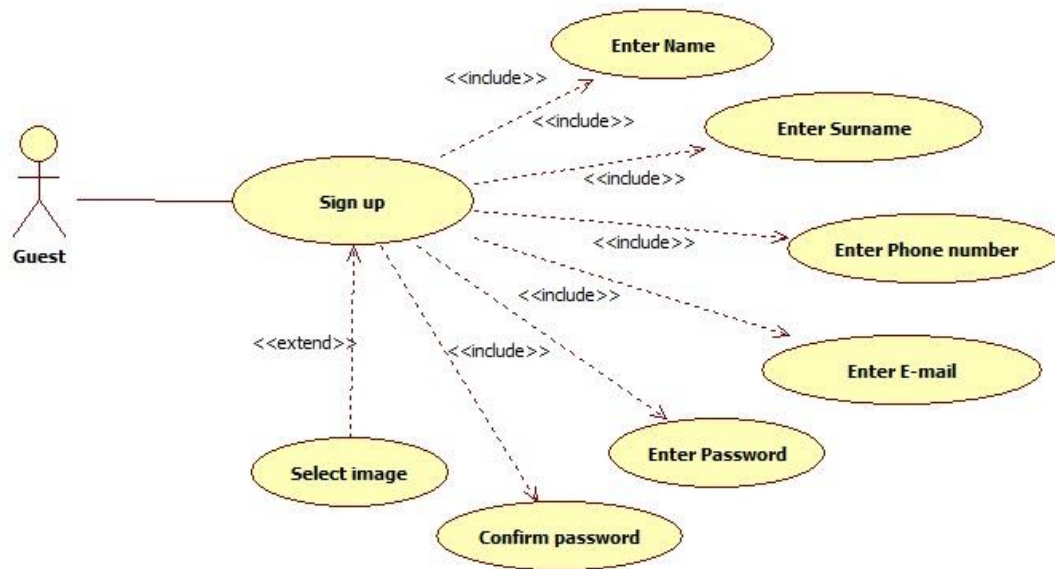
3.4.4   Scenario 4

Catherine as a well experienced F1 driver got hurt in a car accident and after the recovery her doctor told her that she has to give up that extreme sport and find a nice, less invasive job. She thought about it intensively and she figured out that she couldn't live her life without driving a car so with her resume she managed to get a job easily as a taxi driver. After a few days working as a taxi driver she still couldn't remember all the street addresses, but luckily myTaxiService application when accepting a request for a ride offers an navigation option from her current location to the passenger location and after that as well as the route from starting to destination point. When Catherine found out about this convenience she felt so relieved that she didn't have to remember all the routes around the city.
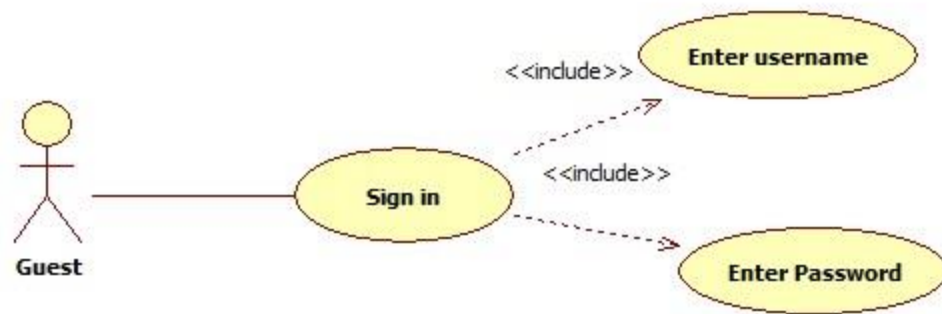
3.5   UML Models

3.5.1   Use Case

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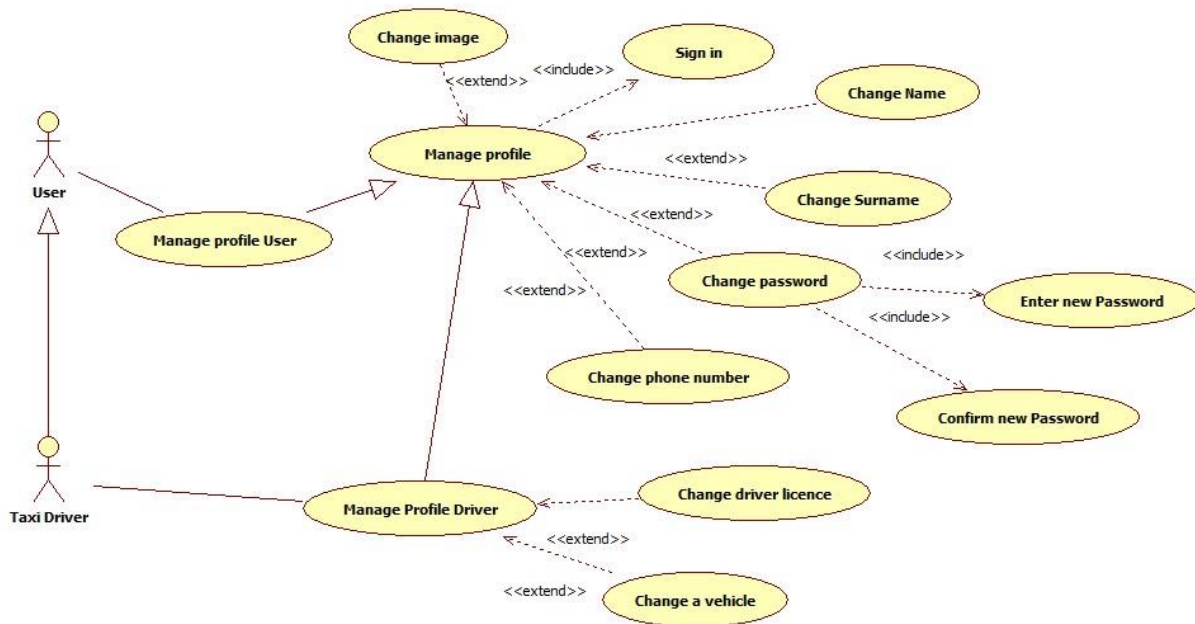
3.5.1.1   Accessing the application

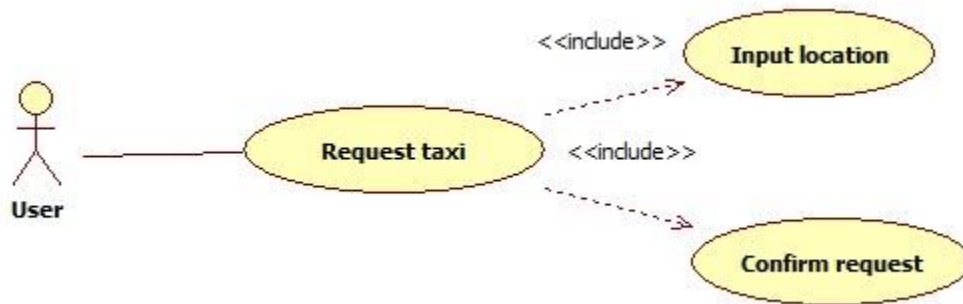| Name	SignUp
Actors	Guest
Entry conditions	The guest has never registered in the system
Event Flow	• Guest navigates to the myTaxiService homepage containing a form asking him to enter basic information, email, password, phone number and image which is optional • Upon the form submission system checks 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, an appropriate message is displayed and the guest is asked to fill the missing fields or correct specific fields

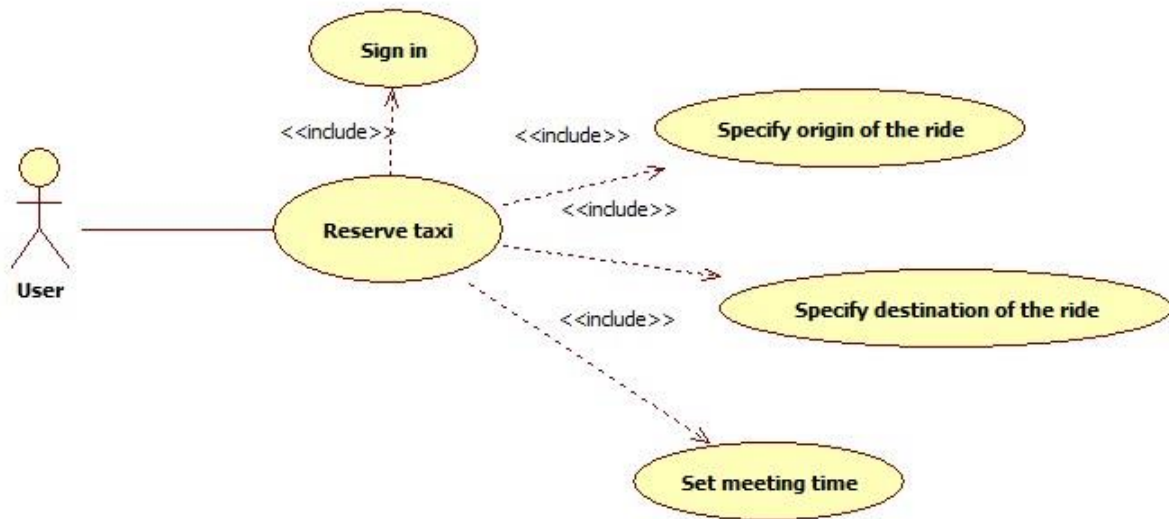| Name	SignIn
Actors	Guest
Entry conditions	The guest has already registered in the system and knows his email and password
Event Flow	- After navigating to myTaxiService homepage containing a form asking him to enter his email and password - Upon the form submission system checks if the username and password match an existing user - If the entered information is valid, the user is signed in to the system and redirected to his personal page
Exit Conditions	The user is granted access to all of the myTaxiService's functionalities
Exceptions	If he information the user entered is not valid, an appropriate message is displayed and the guest is asked to enter his username and password again

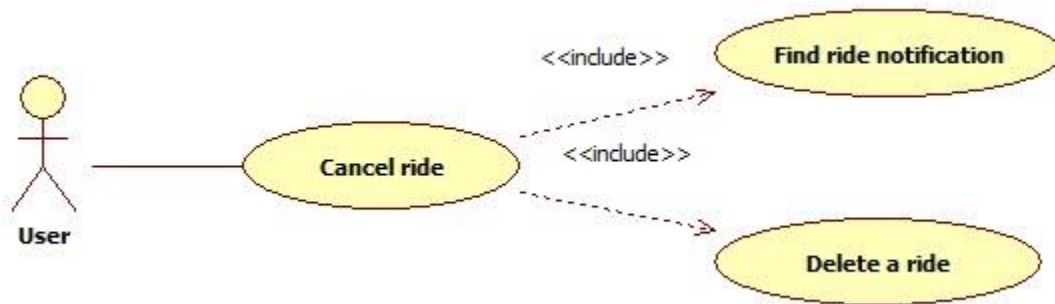| Name	Manage Profile
Actors	User
Entry conditions	The user is logged in

Event Flow	• User chooses the Manage Profile option from the drop down menu in home screen
	• User is redirected to a Manage profile screen where he/she can change the following information:
	    o name
	    o surname
	    o password
	    o phone number
	    o 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

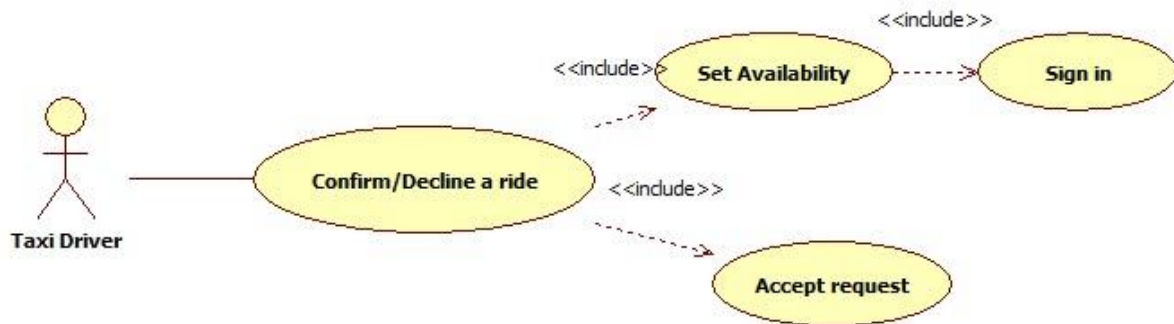| Name	Request taxi
Actors	User
Entry conditions	The user is logged in
Event Flow	• User chooses the *Request taxi* option from drop down menu of the home screen • User is redirected to a Request screen where he/she can select desired location on the map where to catch a taxi • After selecting a location user submits request simply by clicking request button
Exit Conditions	The request is sent to myTaxiService system, confirmation with code of incoming taxi and ETA is sent back to user
Exceptions	If the user doesn't select location on the map, an appropriate message will be displayed and the user will be asked to choose starting point location on the map

Name	Reserve taxi
Actors	User
Entry conditions	The user is signed in

Event Flow	• User chooses the *Reserve taxi* option from drop down menu of the home screen
	• User is redirected to a Reserve screen where he/she must:
	    o Specify Origin of the ride
	    o Specify destination of the ride
	    o Set meeting time
	• After selecting a location user submits request simply by clicking Ok 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

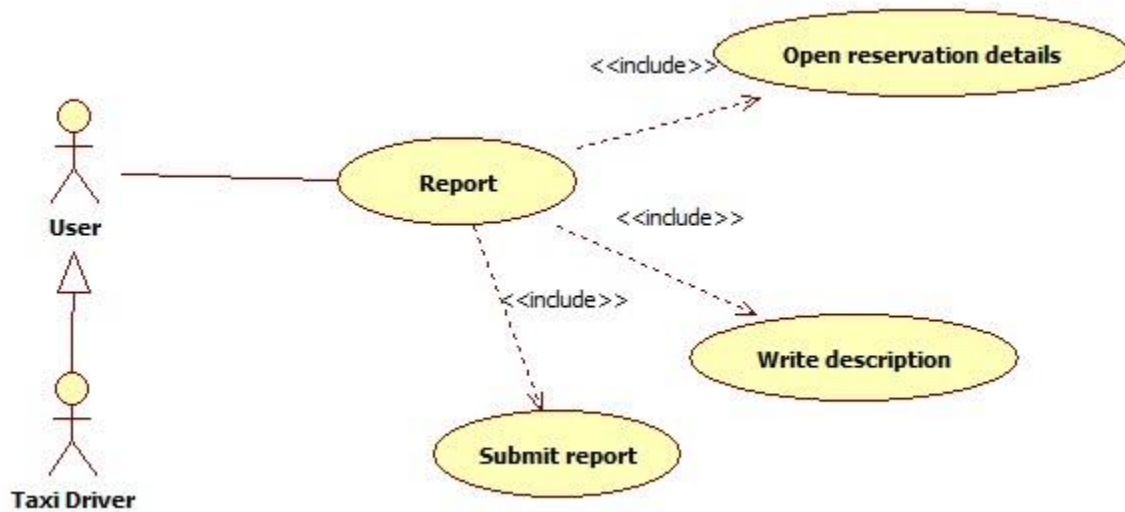| Name	Cancel ride
Actors	User
Entry conditions	The user is signed in and requested a ride
Event Flow	• User opens the *Notifications* from drop down menu of the home screen • User is redirected to a Notifications screen where he/she can delete the active requests or reservations for a ride
Exit Conditions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

Name	Confirm/Decline a ride
Actors	Taxi Driver
Entry conditions	The driver is signed in, his availability status button is set to YES and he received a request
Event Flow	• Taxi driver receive pop-up message with a request for a ride • Taxi driver accept or declines the ride by choosing appropriate option on a pop-up window
Exit Conditions	If taxi driver confirmed a ride, navigation screen is shown to him. Otherwise, his state is unchanged and he awaits for new requests
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

| Name	Check Taxis Available
Actors	User
Entry conditions	The user is signed in
Event Flow	• User chooses the Check available vehicle option from the drop down menu • User is navigated to Taxis available screen
Exit Conditions	-
Exceptions	• If user's GPS service is not functioning, he will be notified by a pop-up message stating that his GPS is not working

| Name	Report
Actors	User
Entry conditions	The user is signed in
Event Flow	• User opens reservation details page • User chooses report option and 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

3.5.2   Class Diagram

Here is presented the class diagram. This diagram will be updated during the developing process.
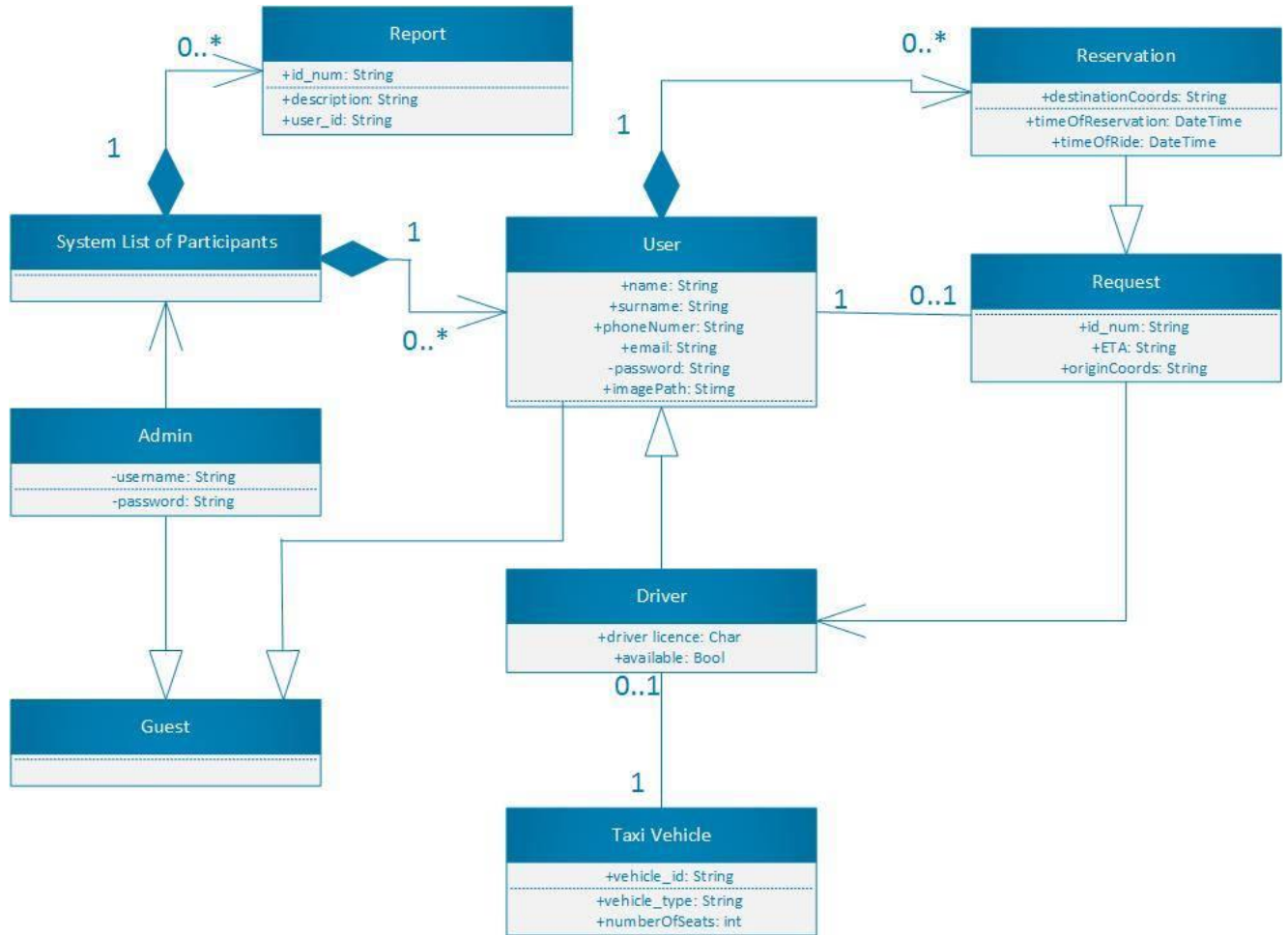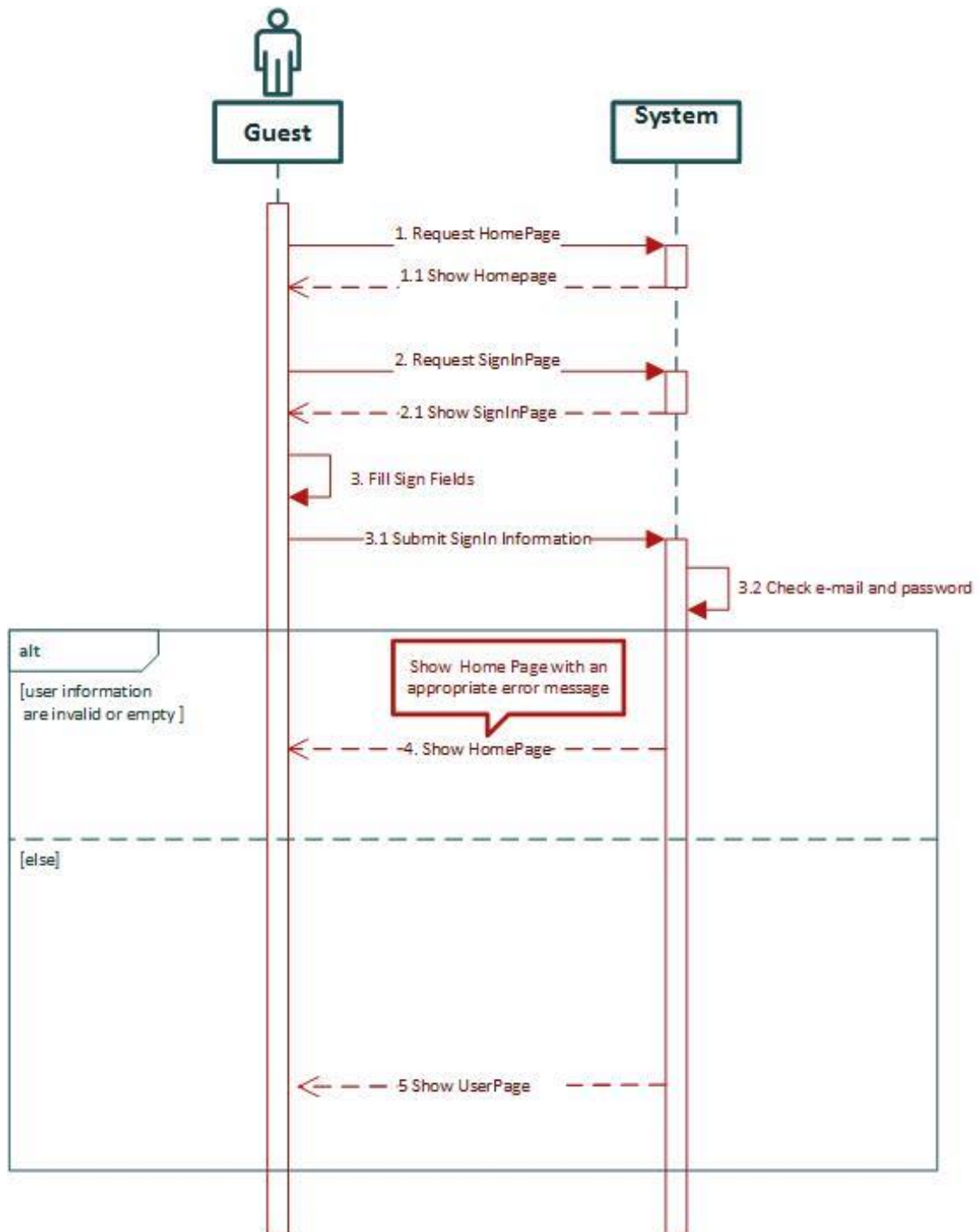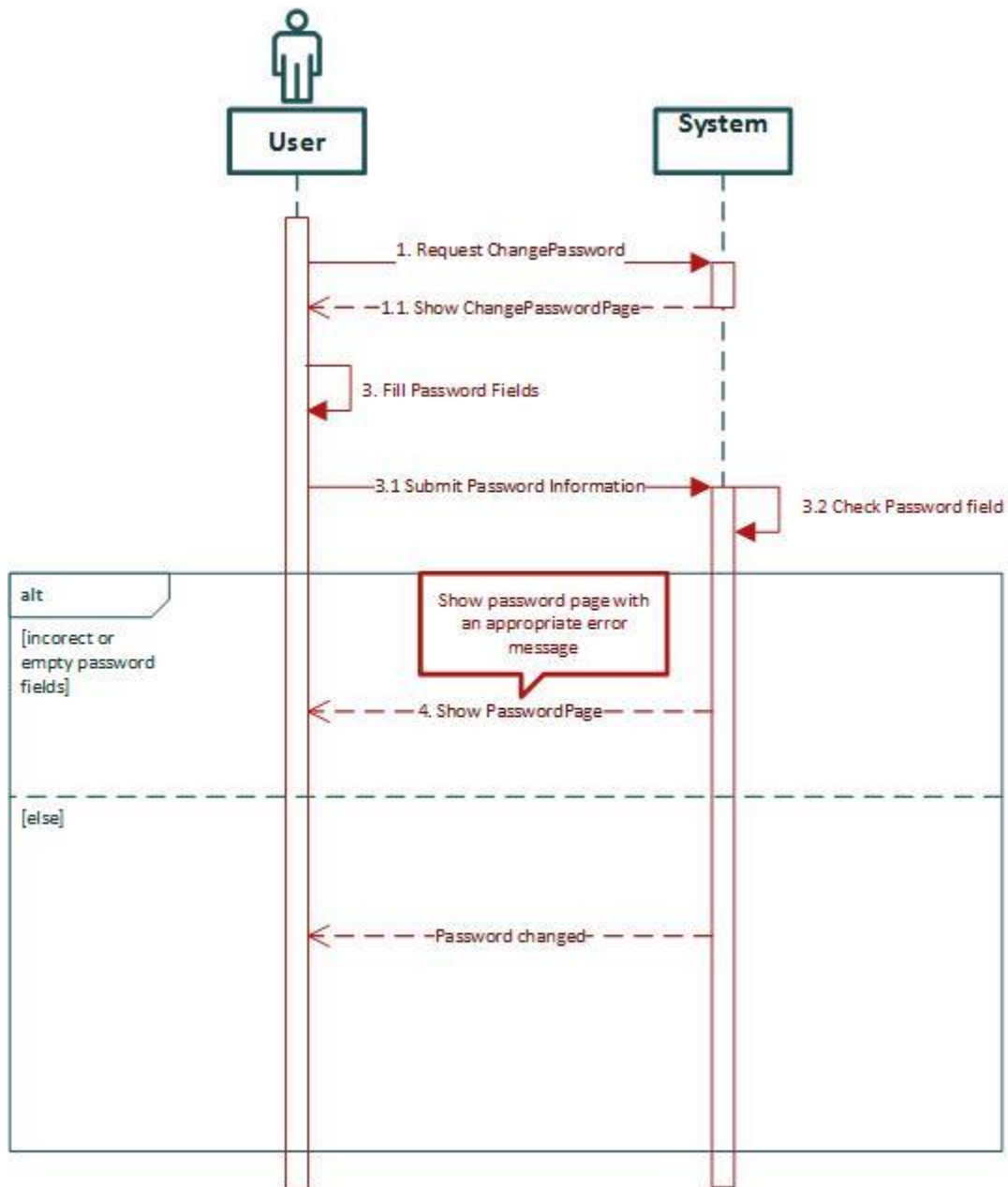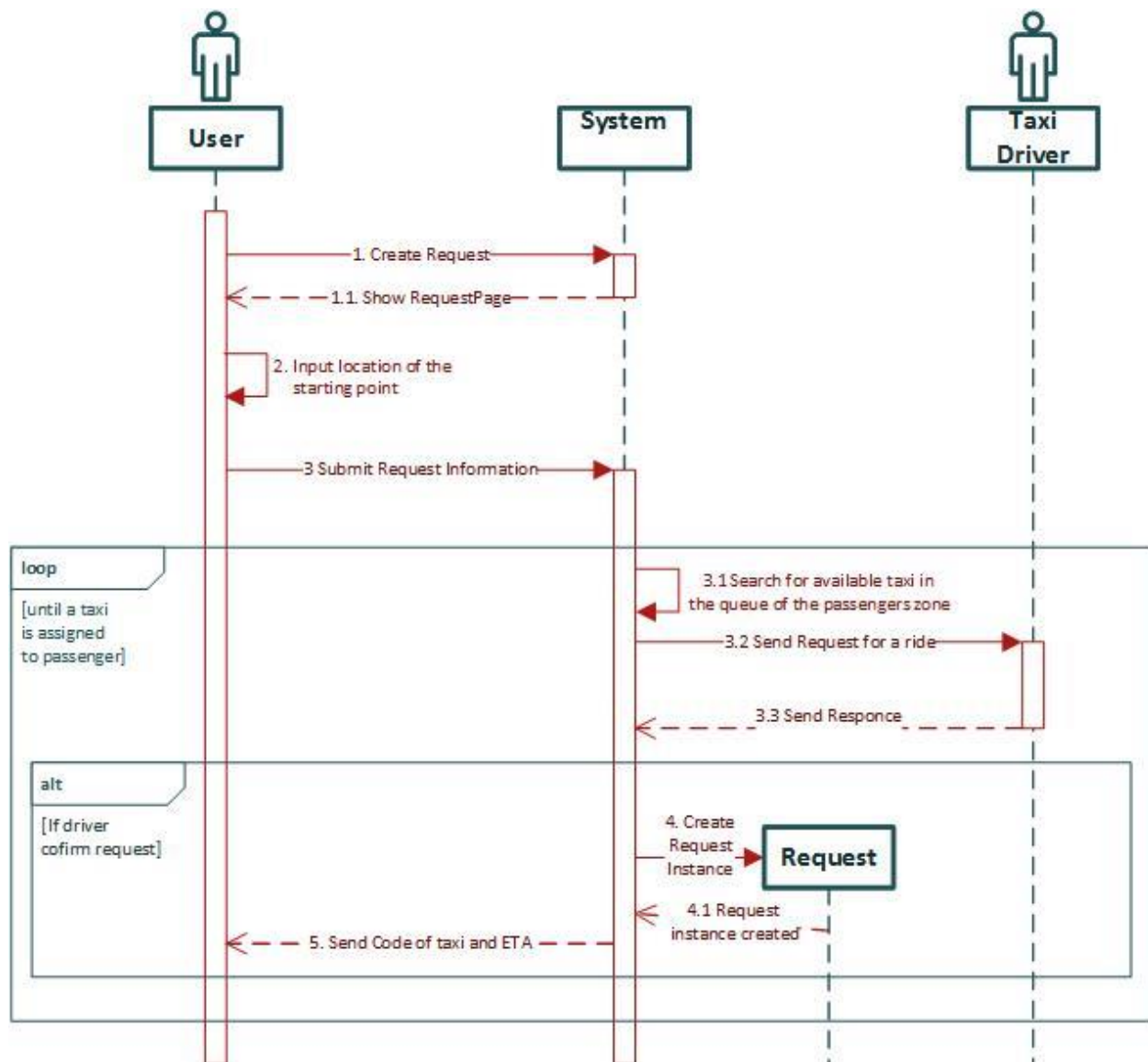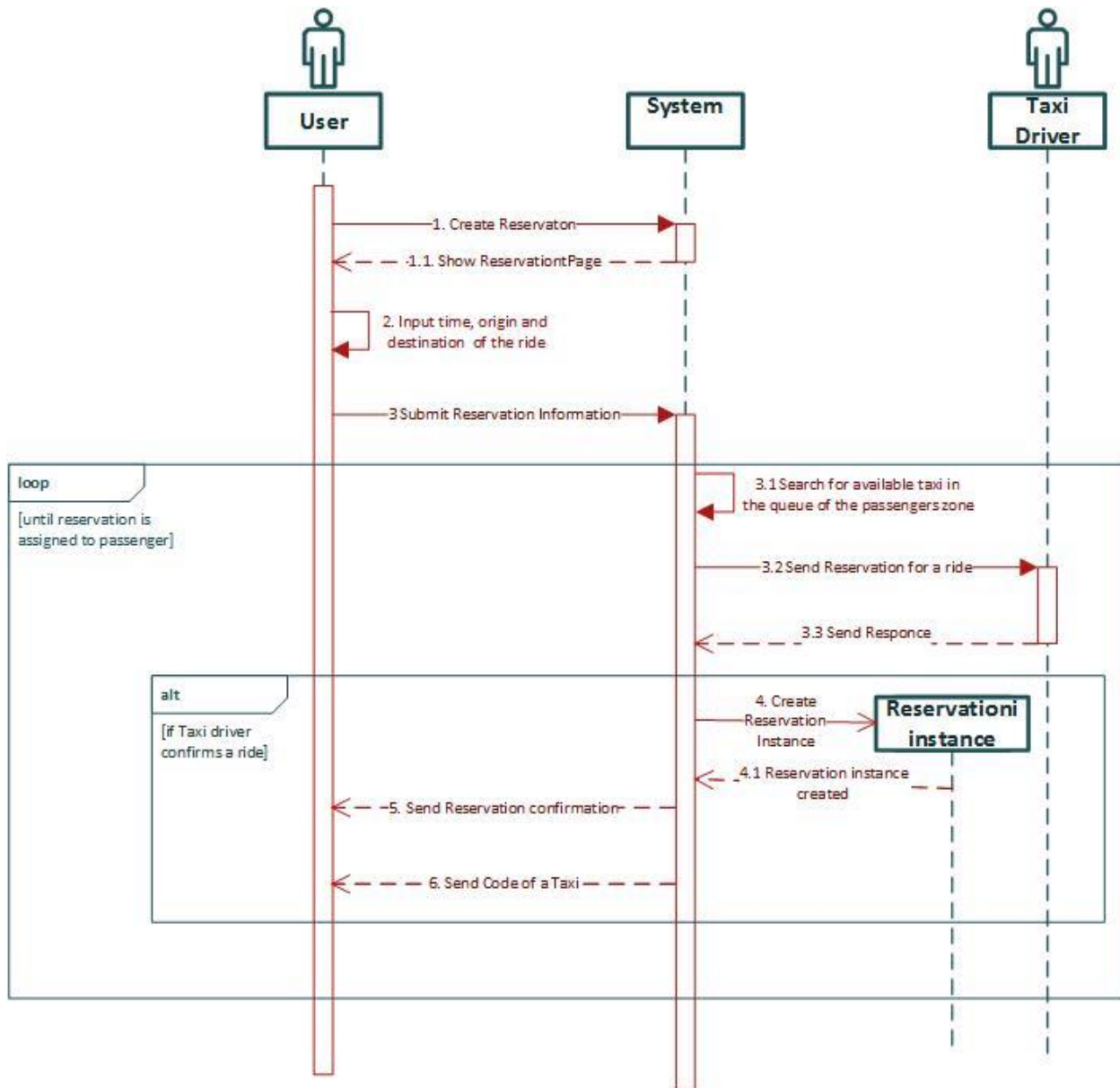
3.5.3   Sequence Diagram

3.5.3.1   SignUp
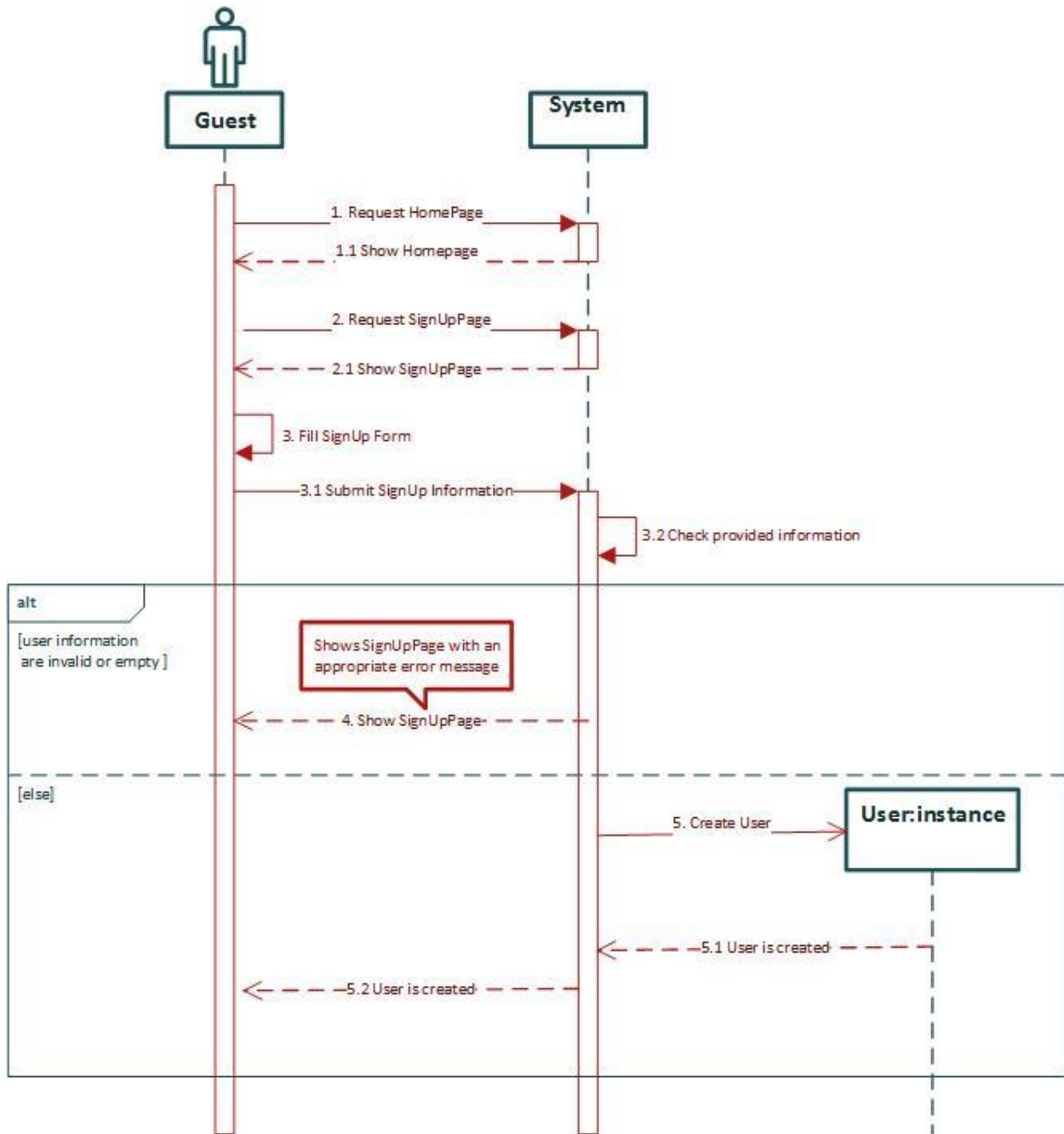
In this sequence diagram as example of managing user profile, Change password sequence of actions is shown.

3.5.3.5 Reserve a ride
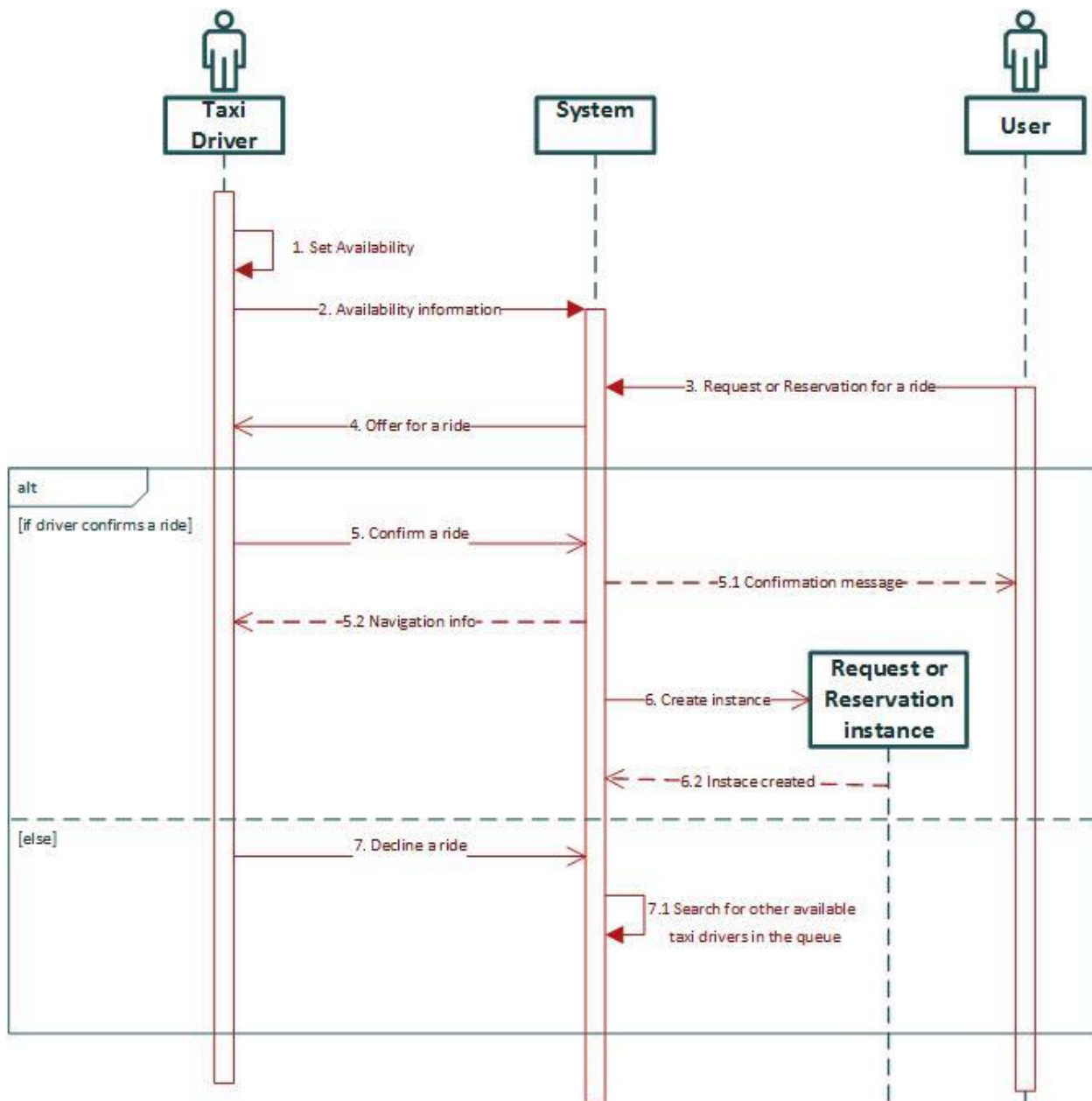
3.5.4 State Chart diagrams
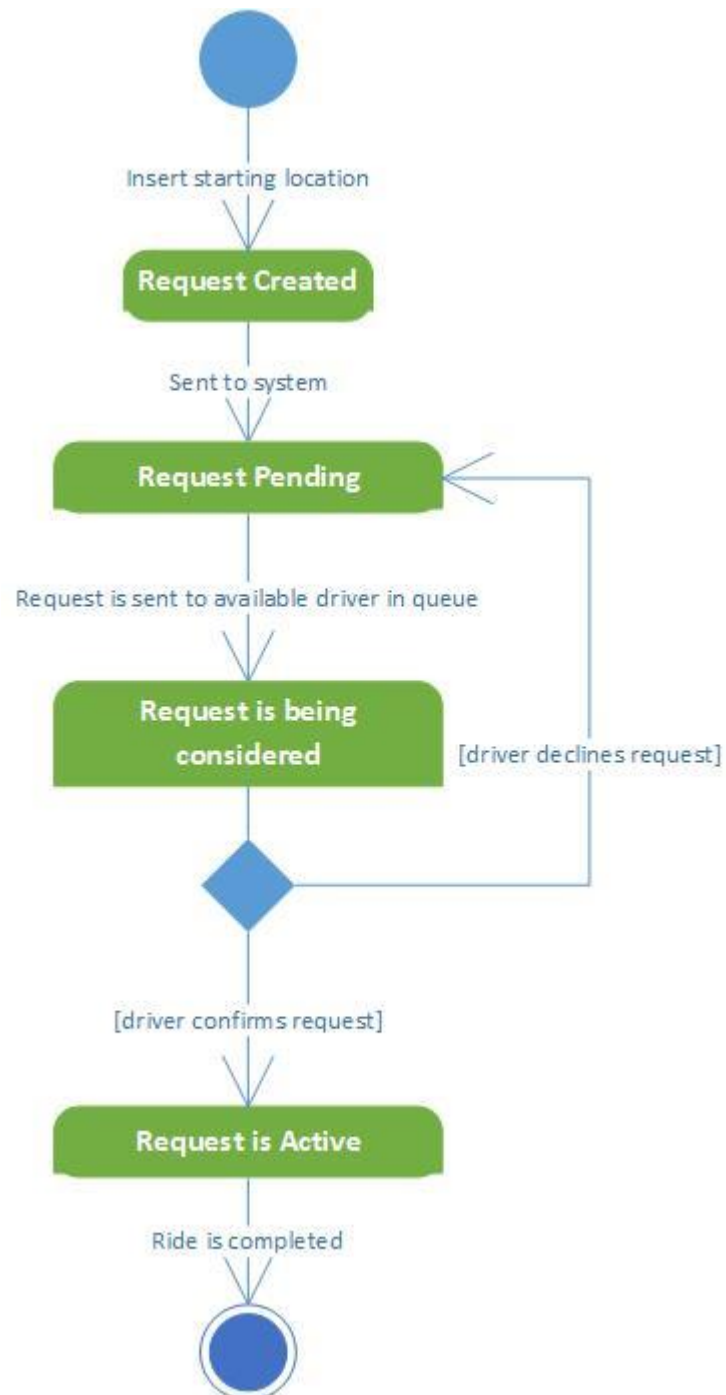
3.5.4.1 Request lifecycle

Insert starting location

Request Created

Sent to system

Request Pending

Request is sent to available driver in queue

Request is being considered

[driver declines request]

[driver confirms request]

Request is Active

Ride is completed
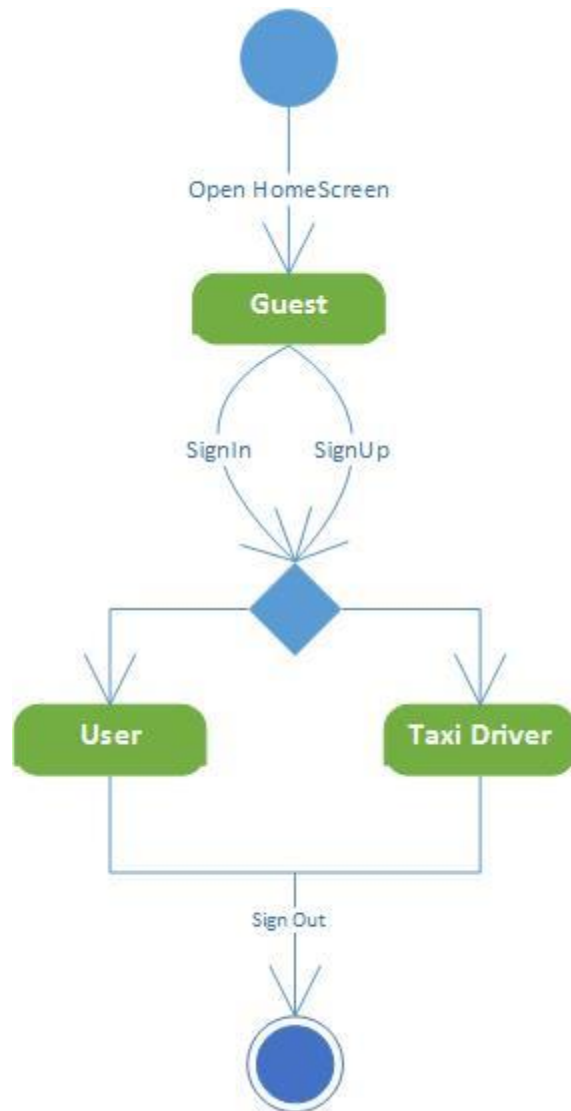
3.6 Non Functional Requirements

3.6.1 Performance Requirements

Performance of the system should be good enough to provide the user with fast responding software system. Response time should be small enough to enable good user experience.

3.6.2 Software System Attributes

- *Availability,* the application should be available to handle user's request at all times using any device with an installed web browser.
- *Maintainability,* The software system provide specific API for enabling future developers with option to add more services or fix bugs in the system.
- *Portability,* the application could be run on device with any OS that has access to Internet and has a web browser.
- *User Interface,* the web application should be intuitive so even the nontechnical users can use the system as simply and efficiently as possible.

4   Appendix

4.1   Alloy

```
module language/myTaxiService
//SIGNATURES
sig Guest{}

sig User extends Guest{
    request: lone Request,
    reservations: set Reservation
}

sig Driver extends User{
    car: one TaxiVehicle
}

sig TaxiVehicle{
    currentDriver: lone Driver
}

sig Report{}

sig SystemListOfParticipants{
    participants: set User,
    reports: set Report
}

sig Request{
    passenger: one User,
    driver: one Driver
}

sig Reservation extends Request{}

sig Admin extends Guest{
    list: one SystemListOfParticipants
}
```

```
//FACTS
fact differetDriversPerVehicle{
    no d:Driver | some t1,t2: TaxiVehicle |
    t1!=t2 and d in t1.currentDriver and d in t2.currentDriver
}
fact carDriverRelation{
    all t:TaxiVehicle | all d:Driver| t in d.car => t.currentDriver=d
}
fact noSameCarTwoDrivers{
    no t:TaxiVehicle| some d1,d2: Driver | d1!=d2 and d1.car=t and d2.car=t
}
fact noSameCarTwoDrivers1{
    all d:Driver| all t:TaxiVehicle|  d in t.currentDriver => t in d.car
}

fact DriverCantBePassenger{
    no u:Driver| some r:Request| u in r.passenger
}

fact singleSystemListOfParticipants{
    no disj l1,l2: SystemListOfParticipants| l1!=l2
}
fact allParticipantsMustBeInList{
    all p: User | one l1:SystemListOfParticipants | p in l1.participants
}
fact reportsAreInSystemList{
    all r:Report| one l:SystemListOfParticipants| r in l.reports
}

fact diffDriversPerRequest{
    all r1,r2:Request | some d1,d2:Driver| (d1 in r1.driver and d2 in r2.driver) => (d1!=d2)
}

fact diffUsersPerRequest{
    all r1,r2:Request | some u1,u2:User| (u1 in r1.passenger and u2 in r2.passenger) => (u1!=u2)
}



//PREDICATES
pred show{|
#Guest=10
#User=8
#Driver= 4
#Admin=1
#Request= 4
#Reservation=2
#Report=2
}
run show for 15
```

4.2   Worlds generated

In this section some worlds generated by Alloy Analyzer will be display in order to prove the model's consistency.

4.2.1   General world

A world is shown in figure 4.1 in which

- the taxi driver cannot drive two cars in the same moment
- the user cannot request two rides
- the request cannot have driver as a passenger and as a taxi driver
- the request cannot have two taxi drivers assigned to it
- there is only one list of system's participants
- all reports are stored in the unique list
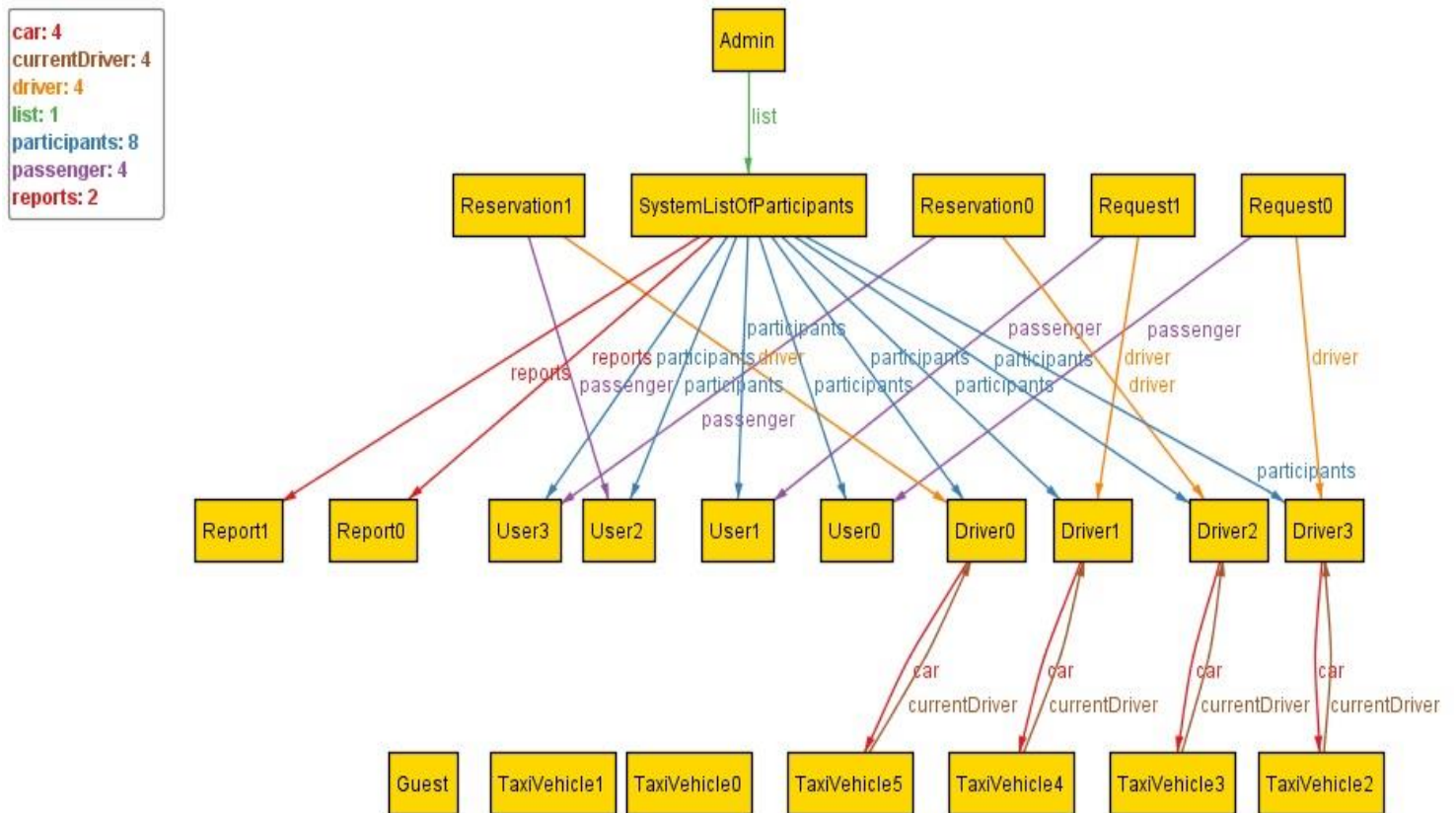- only Admin can access the list of reports

Figure 4.1