



# **Politecnico di Milano**

Master in Computer Science and Engineering

## *Software Engineering 2 Project: myTaxiService*

**Requirements analysis and specification document  
(RASD)**

**Authors:**

Filippo Leporati  
Danilo Fusi

**Professor:**

Elisabetta Di Nitto

# 1 TABLE OF CONTENTS

---

2	Introduction.....	3
2.1	Purpose.....	3
2.2	Scope .....	3
2.3	Acronyms and definitions .....	4
2.3.1	Definitions .....	4
2.3.2	Acronyms and abbreviations .....	5
2.4	References .....	5
2.5	Overview .....	5
3	Overall Description .....	6
3.1	Product Perspective.....	6
3.1.1	System interfaces .....	6
3.1.2	User interfaces.....	6
3.1.3	Hardware interfaces .....	10
3.1.4	Software interfaces.....	10
3.1.5	Communications interfaces .....	11
3.1.6	Memory .....	11
3.1.7	Operations .....	11
3.2	Product functions .....	11
3.2.1	General Requirements.....	11
3.3	User characteristics .....	14
3.4	Constraints .....	14
3.4.1	Regulatory policies.....	14
3.4.2	Hardware limitation.....	14
3.4.3	Interfaces to other applications.....	14
3.4.4	Parallel operation .....	14
3.4.5	Audit functions .....	14
3.4.6	Control functions .....	14
3.4.7	Higher-order language requirements .....	14
3.4.8	Criticality of application .....	15
3.5	Assumptions and dependencies .....	15
3.6	Apportioning of requirements .....	15
4	Specific requirements.....	16
4.1	External interface requirements.....	16
4.1.1	User interfaces.....	16

4.1.2	Hardware interfaces .....	16
4.1.3	Software interfaces.....	16
4.1.4	Communication interfaces.....	17
4.2	Functional Requirements.....	17
4.2.1	Scenarios.....	17
4.2.2	Analysis diagram and state charts .....	23
4.2.3	Use Cases .....	26
4.3	Performance requirements .....	63
4.4	Logical database requirements.....	63
4.5	Design constraints .....	63
4.6	Standard compliance .....	63
4.7	Software system attributes .....	63
4.7.1	Reliability .....	63
4.7.2	Availability .....	63
4.7.3	Security .....	64
4.7.4	Maintainability.....	64
4.7.5	Portability .....	64
4.8	Other requirements.....	64
5	Appendixes.....	65
5.1	Alloy.....	65
5.1.1	Constraints.....	67
5.1.2	Worlds generated .....	68

## 2 INTRODUCTION

---

### 2.1 PURPOSE

This document represents the Software Requirements Specification (SRS) for all the component and sub-component of the *myTaxiService* application, requested by the Milan's government. It is designed and written for the stakeholders, such as the citizen, taxi drivers and developers involved in the project. Its purpose is to describe the scope, both the functional and non-functional software requirements, as well as the design constraints of the whole service. Furthermore, this document shows how the system's interface are designed in detail.

### 2.2 SCOPE

We are about to project and implement *myTaxiService*, which is an information service, aiming at optimizing the taxi service of a large city. In particular, it wants to:

- Business Goal 01)** Simplify the access of passengers to the service.
- Business Goal 02)** Guarantee a fair management of taxi queues.

Passengers can request a taxi either through a web application or through a mobile app. The system answers to the request by informing the passenger about the code of the incoming taxi and the waiting time.

Taxi drivers use a mobile application to inform the system about their availability and to confirm that they are going to take care of a certain call.

*myTaxiService* will provide general functionalities for managing:

- **Profile**  
*myTaxiService* will manage personal data of the different types of users. Users must be registered to use the service, both for daily users and administrators.
- **Connections**  
*myTaxiService* will manage the network of connections between users and taxi drivers.
- **Taxis distribution**  
*myTaxiService* system guarantees a fair management of taxi queues and distribution in the city.
- **Users**  
*myTaxiService* will manage registering, logging in/out users.

*myTaxiService* will have the following limitations and probably they will develop in future versions:

1. **Taxi sharing option.**  
This means that the user is ready to share a taxi with others if possible, thus sharing the cost of the ride.
2. **User and taxi statistics**  
It will not offer any kind of statistics based on user and taxi interaction.

### 3. Payment

It will not offer any form of payment information for the help offered or any kind of payment gateway.

### 4. External interfaces to other Social Networks

It will not offer external interfaces to Facebook, Twitter and LinkedIn...

myTaxiService will have the following fundamental goals:

- G1** Put in relationship a customer and a taxi driver (from a taxi company).
- G2** Improve the Quality of Service by managing taxi queues and their distributions.
- G3** Facilitate the relationship by detecting proximity between the taxi and the customer
- G4** Manage a system of taxis reservations and request.

## 2.3 ACRONYMS AND DEFINITIONS

### 2.3.1 Definitions

Keyword	Definitions
Administrator	The administrator of the system is the person allowed to manage the entire system.
User	Citizen or taxi driver who interact with the application.
Server	Refers to the myTaxiService server.
LOGIC	A sub-component of the myTaxiService that is responsible for maintaining the taxi distribution logic.
NET	A sub-component of the myTaxiService that is responsible for sending and receiving messages between the other devices.
Taxis' queue	Data structure to manage taxis in the city.
Request	A passenger makes a request when he wants immediately a taxi.
Reservation	A passenger makes a reservation when he does not want taxi immediately, but he wants to select a different time or day.

### 2.3.2 Acronyms and abbreviations

Acronym or abbreviation	Definition
RASD	Requirements Analysis and Specification Document
APP	myTaxiService
NFR	Non-functional Requirements
QA	Quality Attributes
FR	Functional Requirement
DBMS	Database Management System
AS	Application Server
JEE	Java Enterprise Edition
QoS	Quality of Service
GUI	Graphical User Interface

## 2.4 REFERENCES

1. IEEE Recommended Practice for Software Requirements Specifications:  
<http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5841>
2. Alloy model file: 2015\_project\_Alloy\_myTaxiService.als

## 2.5 OVERVIEW

1. Introduction: provides a summary of the software product to be developed.
2. Overall Description: describes the general factors that affect the software product and its requirements.
3. Specific Requirements: contains the information obtained by the analysis. It describes all of the software requirements to a more deep level of detail.
4. Appendixes: provides supporting information about how the alloy model contributed to the requirement analysis and analysis model.

### 3 OVERALL DESCRIPTION

---

This section does not provide specific requirements, but simply illustrates the perspective and the background for specifying concrete requirements in the next section of this document.

#### 3.1 PRODUCT PERSPECTIVE

The software product is a complete self-contained and it is not part of any other larger system or legacy system. However in the future it is possible to cooperate with other service providers, which can offer a much more pervasive experience for the users.

##### 3.1.1 System interfaces


The software does not provide any external interface. The administrator can manage the LOGIC of the system using his account via Web.

##### 3.1.2 User interfaces

The software will present two different page layouts for taxi driver and customers. These following sketches up offers a simplified idea of the design and navigation.



User Interface 1: Default Home Page




**myTaxiService.it**

### Login

[Did you forget your password?](#)

User Interface 2: Login Page




**myTaxiService.it**

### Registration

<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>
<input type="password"/>	<input type="button" value="Register"/>
<input type="password"/>	

User Interface 3: Registration Page






myTaxiService.it

Welcome Mario Rossi!


[Profile](#)
[New Reservation](#)
[Reservation Chronology](#)
[Logout](#)

### Taxi Reservation

Origin:   
 Destination:   
 Date:  /  /    
 Hour and Minute:

Reserve it!

User Interface 4: Taxi Reservation Page



myTaxiService.it

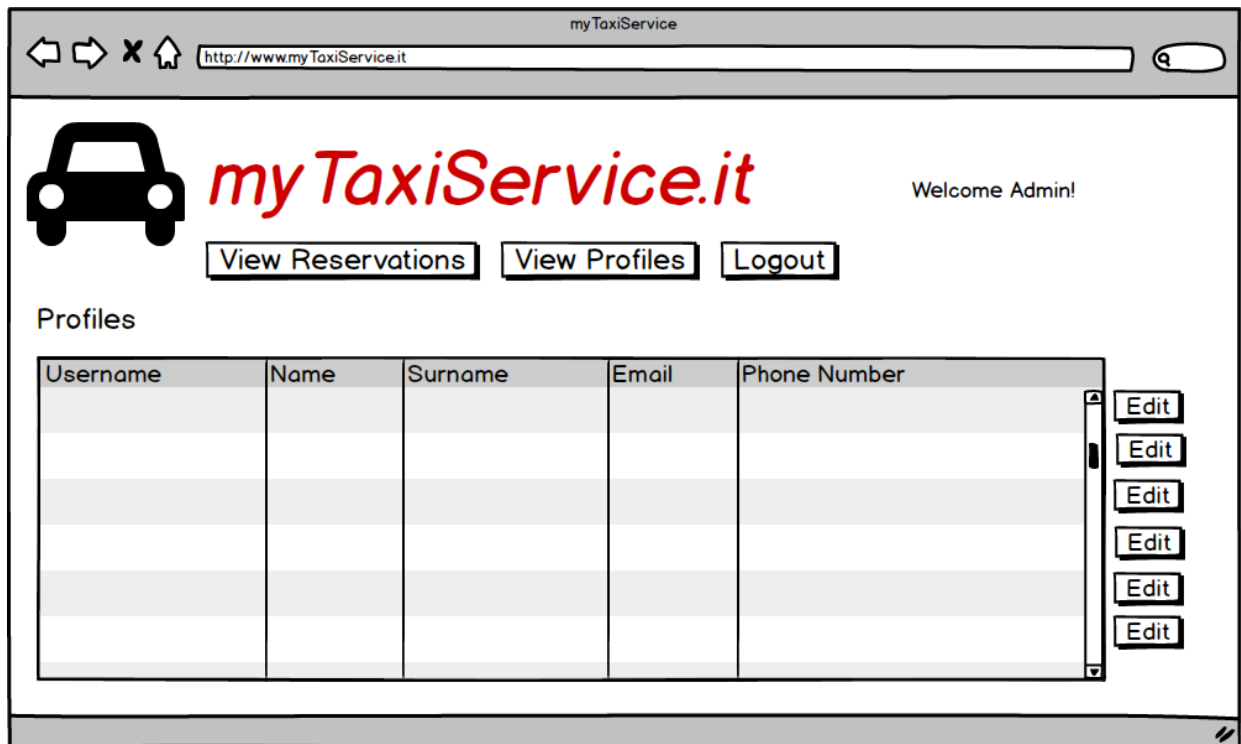
Welcome Mario Rossi!

[Profile](#)
[New Reservation](#)
[Reservations History](#)
[Logout](#)

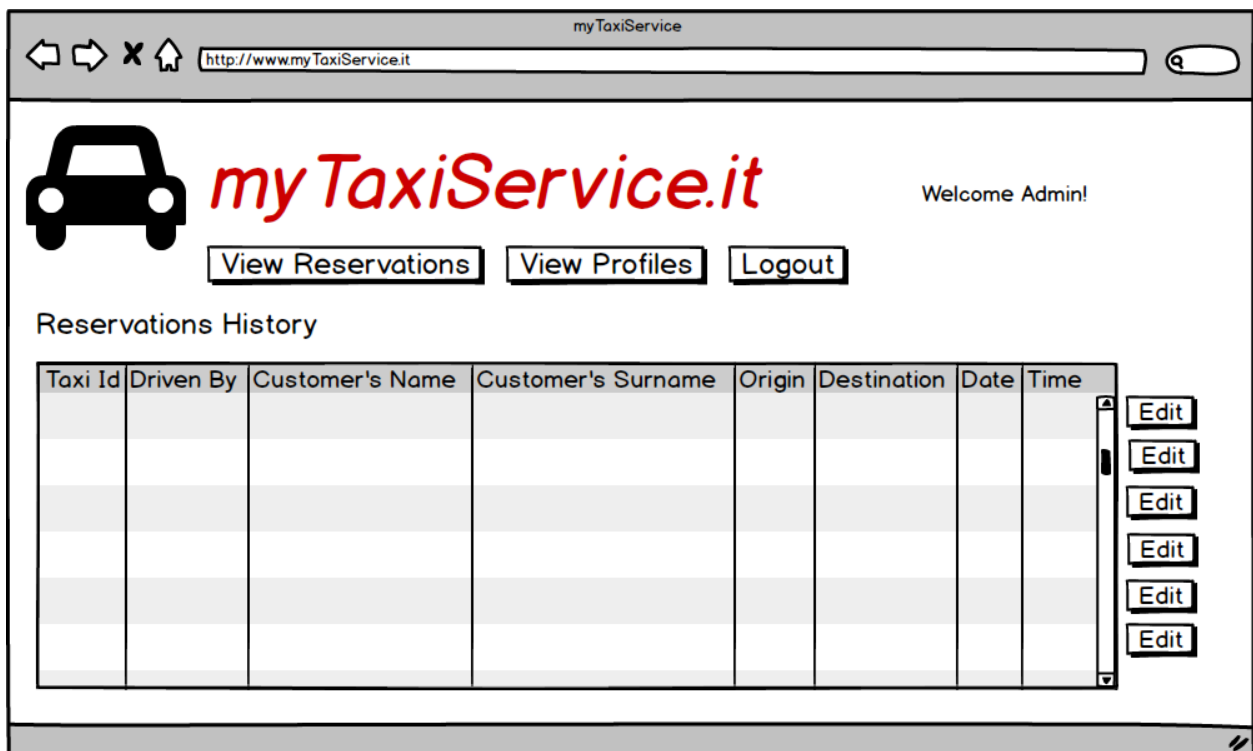
### Reservations History

Origin	Destination	Date	Time

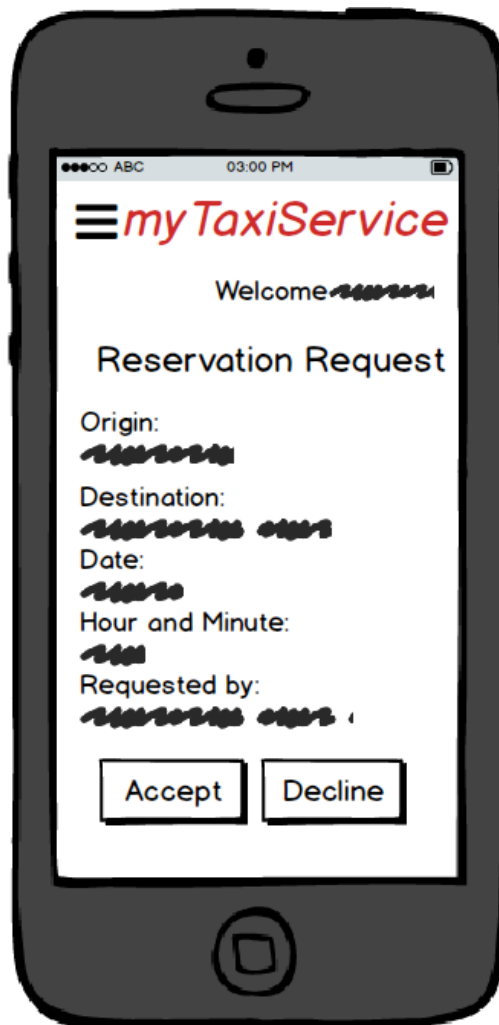
User Interface 5: Reservation History Page



User Interface 6: Registered Users Page - Admin



User Interface 7: Reservation History – Admin



*User Interface 8: Reservation Request – Taxi Driver*

### 3.1.3 Hardware interfaces

The software product will not provide any hardware interface, because the application will work only on mobile phone devices and web application.

### 3.1.4 Software interfaces

Database managements systems	
<b>Name</b>	MySQL
<b>Mnemonic</b>	MySQL
<b>Specification number</b>	Community edition
<b>Version number</b>	5.6.27
<b>Source</b>	<a href="http://www.mysql.com/downloads">http://www.mysql.com/downloads</a>

In order to use the web application, users are required to have installed any of the following web browsers  
i: IE 6.0+, FF 10+ or Chrome 20+.

### Operating System

The software product will run in any mobile operating system like Android, iOS and Windows Phone. The taxi driver can use as well their mobile phones, simply they have to log in with a different type of username, given by the system. If the passenger prefers to use a web application, they must have a computer, on which is installed any operating system that support a modern browser seen before.

#### 3.1.5 Communications interfaces

Protocol	Port	Service
TCP	80	HTTP/HTML connection
TCP	3306	MySQL

For the first release of the project, we will assume that the DBMS and the Application Server run both on the same physical server. All the mobile devices, to work with the entire system, must be connected to internet network, through high-speed connection (3G/4G).

The connection must be encrypted via HTTPS or SSL to ensure the total privacy of account.

#### 3.1.6 Memory

The minimum memory requirements for the mobile devices are:

- Main memory: 32 Gigabytes or more.
- Secondary memory: at least 1000 Gigabytes.

It is important to remark that secondary memory is recommended to be physically in a different server form the software is installed, because it will probably increase rapidly and can effect on system performance. For the first part of the project, we will assume that the memory is installed on the same server.

#### 3.1.7 Operations

The customers and the taxi drivers will interact with their devices as functional registered user. For these kind of users, the product functions section below describes the operations, which they can perform.

## 3.2 PRODUCT FUNCTIONS

This subsection provides a summary of the major functions of the application.

#### 3.2.1 General Requirements

We have pointed out four main general requirements:

- Managing profiles
- Managing users
- Managing taxi request and reservation
- Managing distribution and reallocation

The following subsections will discuss, in detail, the FR and NFR of points above.

#### **3.2.1.1 Managing profiles**

##### **Functional requirements**

1. View personal information
2. Modify personal information
3. View personal reservation

#### **3.2.1.2 Managing users**

##### **Functional requirements**

1. Register to the system
2. Login
3. Logout
4. Modify password
5. Recover password
6. Delete an user from the system
7. Support, at the same time, a user who is both a taxi driver as passenger.
  - a. The taxi driver will login with his driver Code and his password.
  - b. The same taxi driver can use his personal username and password to login as passenger.

##### **Non-functional requirements**

- a. User password must be stored securely, with certified cryptography.
- b. User password must be at least 8 characters and it must be changed every 3 months.
- c. System must support a high number of users (at least 500.000).

#### **3.2.1.3 Managing taxis request and reservation**

##### **Functional requirements**

1. Users must log in order to use the application.
2. All users' devices must connect to the Network in order to use the application.
3. Passenger can request a taxi through either a web application or a mobile app.
4. The system answer to the request informing the passenger about the code of the incoming taxi and the waiting time.
5. Taxi drivers inform the system about their availability and confirm that they are going to take care of a certain call.
6. A user can reserve a taxi by specifying the origin and the destination of the ride.
  - 6.1. The reservation has to occur at least two hours before the ride.
  - 6.2. In the case of correct reservation, the system confirms the reservation to the user and allocates a taxi to the request 10 minutes before the meeting time with the user.
  - 6.3. In the case the user wants to cancel the reservation, he must do it two hours before the meeting time.
  - 6.4. The system will treat a reservation as a future request, so it simply calls taxi before the reservation appointment in the preselected zone.
7. If a user reserves or require a taxi and he does not use it for more than 3 times, the administrator block his account for a month.

## Non-functional requirements

- a. The system must support tens of request arriving in seconds.
- b. The system must be responsive( maximum 1-3 seconds to react to user's input)
- c. The system must be user-friendly. A passenger must be able to make a request in at maximum 10 seconds.

### 3.2.1.4 Managing taxi distribution and reallocation

## Functional Requirements

1. The system guarantees a fair management of taxi queues.
2. The service needs to put in relationship customer and driver finding the closest driver.
3. The system must localized (using GPS or Internet connection) the user's device in order to manage the distribution of the taxis.
4. The city should be divided in taxi zones (approximately two km<sup>2</sup> each, with a minimum of one km<sup>2</sup> and a maximum of three km<sup>2</sup>).
  - a. Each zone is associated to a queue of taxis.
  - b. It is possible to cancel just the most external areas, but before the administrator must move all the taxis in another zone.
  - c. The system automatically computes the distribution of the taxis in the various zones based on the localization information it receives from the taxi.
  - d. When a taxi is available, its identifier is stored in the queue of taxis in the corresponding zone.
  - e. When a request arrives from a certain zone, the system forwards it to the first taxi queuing in that zone.
  - f. Every zone should have a minimum number of taxis, according to certain decisional parameters (density of people, number of possible calling, exposition building...).
  - g. The sum of all minimum queues values and the number of zones must be the same as the number of taxi of the service (available and not).
5. If the taxi confirms the request, then the system will send a confirmation to the passenger.
6. If the taxi does not confirm the request, then the system will forward it to the second in the queue and, at the same time, it will move the first taxi in the last position in the queue.
7. If a certain zone reach the minimum number of taxis, then the system explore the adjacent zones to find the queue with the largest number of taxis available and move one of them to that zone (it will put at the end of the new queue). This algorithm is applied recursively.
  - a. In the case all queues in the adjacent zones are below the minimum threshold, the system does nothing and wait until a taxi becomes free.
8. If a customer deletes a request before the taxi arrives or a taxi is arrived to destination, the system stops the taxi driver and put him in the queue of the current zone or the adjacent ones. The system adds the taxi to the queue in which the difference between the minimum number of taxi and the current number of them is the biggest (if there are more possibilities, the choice is random).
9. A customer can cancel the taxi request only if it is elapsed less time than the half of arrival time given by the taxi driver. For instance if the arrival time indicated is four minutes, it is possible to cancel the request within two minutes from the demand.
10. The customer who requires a taxi can specify the number of passengers with him. So that the system can choose the correct car, if available. If it is not possible to fulfil the request, the system will advise the customer.

11. If no taxi in a certain zone are available, i.e. all refuses the request, the system tells the customer that there are not free taxi in his area so it will take more time to have a cab. The customer can decline or accept. If he accepts, a taxi in adjacent areas is used, until the system can find a taxi. After that, the customer can accept or decline the choice.
12. In the rare case, no taxis are available in whole city, i.e. all the taxis refuse a request, the system tells to the customer that he has to try in another moment to require a taxi, because no one is free.
13. The system administrator decides the minimum number of taxi in each area, depending on how much taxis are request for that day in a certain zone. For instance, if it will be a football match in the evening, the admin will increase the minimum number of taxi in the areas near the stadium, so that a certain amount of cars will move towards that zone.

### 3.3 USER CHARACTERISTICS

User should meet the following characteristics:

- High school education level.
- Knowledge in using a browser.
- Knowledge in using a mobile device.

### 3.4 CONSTRAINTS

The following constraints apply to the myTaxiService:

#### 3.4.1 Regulatory policies

The myTaxiService will face with a certain number of regulatory policies. For example, policies related with taxi usage, personal privacy and internet connection (cookies, private details...).

These policies will be analysed by a team of lawyers, during the whole project's development.

#### 3.4.2 Hardware limitation

The myTaxiService has strict hardware limitations, because the application runs on mobile devices, which as limited resource and power computation.

#### 3.4.3 Interfaces to other applications

The software product must interface with the geo-localization system of the mobile devices, in order to identify the taxi's position.

#### 3.4.4 Parallel operation

The software product must support the operation of simultaneous users, especially when a certain number of customers makes a request to have a taxi.

#### 3.4.5 Audit functions

The software product does not perform any audit.

#### 3.4.6 Control functions

The software product does not control any other devices or system.

#### 3.4.7 Higher-order language requirements

The software product requires knowledge of different kind of programming languages. In fact, the team has to work with Web application, which uses HTML, PHP, MySQL.... It is necessary also a good ability in using Java, Objective-C and other languages to create the mobile application.

#### 3.4.8 Criticality of application

The application requires proper support of concurrent users and operations. Another critical aspect is the correct localization of taxi drivers.

### 3.5 ASSUMPTIONS AND DEPENDENCIES

The requirements in this document are based on these following assumptions:

1. Android, iOS or Windows Phone compatible devices or a browser to use the Web Application.
2. Users have a decent and acceptable Internet connection.
3. Users must be localized by one of these technologies, 3G/4G connection or GPS, in order to manage their position.
4. The software product provides one administrator user by default.
5. The software product does not support more than one administrator.

### 3.6 APPORTIONING OF REQUIREMENTS

Future releases of the software product may provide:

1. Connectivity to other social networks.
2. Many administrators.
3. Statistics and auditing operations.
4. Taxi sharing between more users.
5. Path optimization for taxi driver.

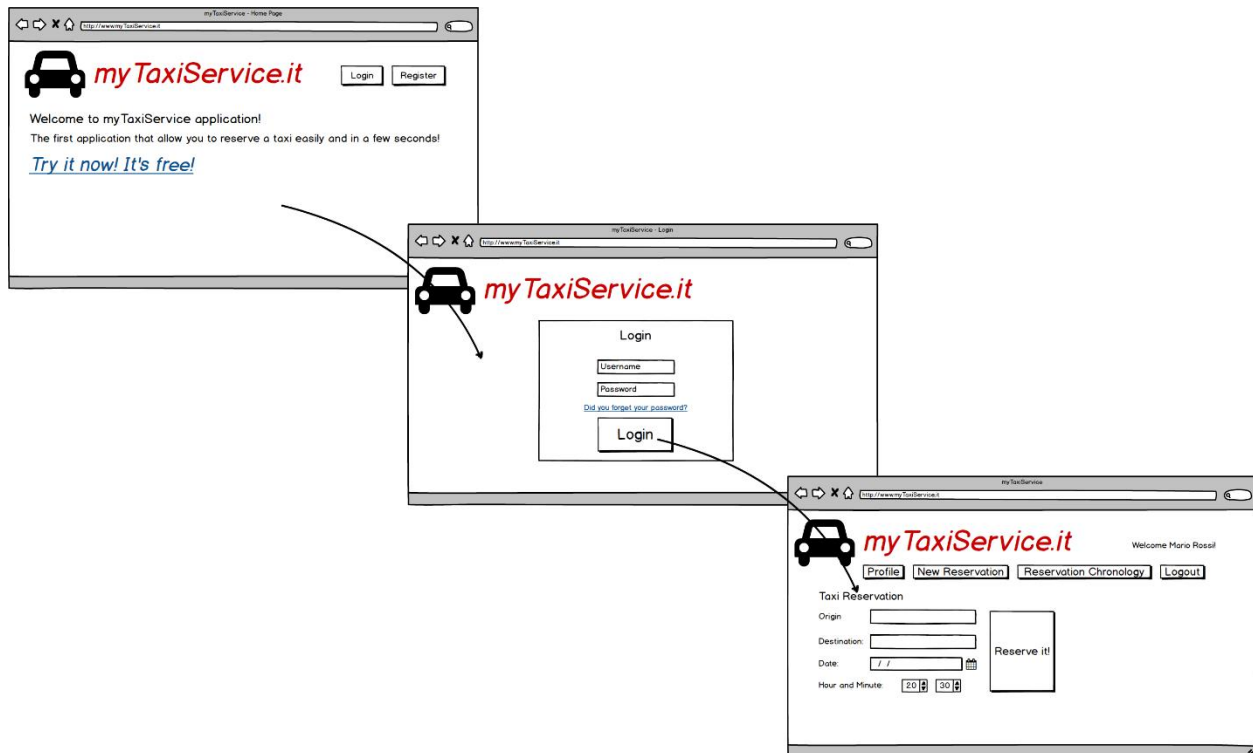


## 4 SPECIFIC REQUIREMENTS

### 4.1 EXTERNAL INTERFACE REQUIREMENTS

#### 4.1.1 User interfaces

The following storyboard represents the workflow to access to the home page of a registered user and reserve a taxi.



Storyboard 1

#### 4.1.2 Hardware interfaces

The software product will not provide any hardware interface, because the application will work only on mobile phone devices and web application.

#### 4.1.3 Software interfaces

Database managements systems	
Name	MySQL
Mnemonic	MySQL
Specification number	Community edition
Version number	5.6.27
Source	<a href="http://www.mysql.com/downloads">http://www.mysql.com/downloads</a>

In order to use the web application, users are required to have installed any of the following web browsers  
i: IE 6.0+, FF 10+ or Chrome 20+.

#### 4.1.4 Communication interfaces

For the first release of the project, we will assume that the DBMS and the Application Server run both on the same physical server. All the mobile devices, to work with the entire system, must be connected to internet network, through high-speed connection (3G/4G).

The connection must be encrypted via HTTPS or SSL to ensure the total privacy of account.

## 4.2 FUNCTIONAL REQUIREMENTS

### 4.2.1 Scenarios

User registration to the system	
<b>Code</b>	SC-PASSENGER-001
<b>Description</b>	Describe how a customer registers to the system
<b>Goal</b>	[G1] Put in relationship a customer and a taxi driver
<b>Assumptions</b>	The customer is not register to myTaxiService
<b>Scenario</b>	
<p>John is a foreign student living in Milan. He does not like public transport and he prefers to move using taxi in daytime. John is looking forward to a service, which can help him. John discovers myTaxiService.</p> <p>John connects to the internet and goes to the internet address of the system using a browser installed in his computer, when the system load the first page John notice that there is a register option, so he clicks it and fill out the register form. Therefore, he introduces his name, last name, email, password and his telephone number.</p> <p>When John filled out everything, he sends the form and receives a successful registration message. In this way John want to check if he can login into the system with his credential, so he choose the “go to login page” link in the successful registration message and then login and John can see the functionalities that he can do in the system.</p>	

Scenario 1: User registration

Making / consulting a taxi request	
<b>Code</b>	SC-PASSENGER-002
<b>Description</b>	Describe how a customer can make a taxi request
<b>Goal</b>	[G4] Manage a system of taxis' reservations and requests
<b>Assumptions</b>	1. The customer is registered to the system 2. Internet connection available
<b>Scenario</b>	
<p>It's Saturday evening and John would like to take a taxi to go to the restaurant. He takes out his phone and opens the myTaxiService mobile application.</p> <p>He waits that the application localize his position and he clicks on the button “require a taxi”.</p> <p>The application shows that his request is forwarded correctly and asks to wait for just few seconds.</p> <p>After a while, the application notifies to John that a taxi has accepted his request. Moreover, it displays the taxi's code and the minute that he has to wait before the car arrives.</p>	

John would like to see again the request just made. He clicks in the section “Reservation Chronology” where he finds out all the information given by the application when he made the request (taxi’s code, minutes to wait...).

Scenario 2: Making/ consulting a taxi request.

Update a pending request (delay)	
<b>Code</b>	SC-PASSENGER-003
<b>Description</b>	Describe how a pending request is updated
<b>Goal</b>	[G4] Manage a system of taxis’ reservations and requests
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. The customer is registered to the system</li> <li>2. Internet connection available</li> <li>3. The customer must have a taxi request.</li> <li>4. Taxi driver is registered to the system</li> </ol>
<b>Scenario</b>	
John is waiting for the taxi but he is worried because it has not arrived yet. In that moment, the myTaxiService application notifies him that his pending request has changed. John opens the application and he sees the new arrival time, because of a taxi problem.	

Scenario 3: Update a pending request(delay)

Update a pending request (car problem)	
<b>Code</b>	SC-PASSENGER-004
<b>Description</b>	Describe how a pending request is updated
<b>Goal</b>	[G4] Manage a system of taxis’ reservations and requests
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. The customer is registered to the system</li> <li>2. Internet connection available</li> <li>3. The customer must have a taxi request.</li> <li>4. The taxi driver is registered to the system</li> </ol>
<b>Scenario</b>	
John is waiting for his taxi but it is worried because it has not arrived yet. In that moment, the myTaxiService application notifies him that his pending request has changed. John opens the application and he sees that a new car is going to arrive, because the other one had a problem. The taxi code and arrival time are changed.	

Scenario 4: Update a pending request (car problem)

Cancel a pending request	
<b>Code</b>	SC-PASSENGER- 005
<b>Description</b>	Describe how a pending request is cancelled
<b>Goal</b>	[G4] Manage a system of taxis’ reservations
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. The customer is registered to the system</li> <li>2. Internet connection available</li> <li>3. The customer must have a taxi request.</li> <li>4. The taxi driver is registered to the system</li> </ol>

	5. It must be elapsed less time than half the time of taxi arrival
<b>Scenario</b>	
John, while is waiting for a taxi, receives an important to call and he has to cancel the taxi request. He opens the myTaxiService application and clicks on "Request Chronology" and "Cancel Request".	

Scenario 5: Cancel a pending request

Modify personal information	
<b>Code</b>	SC-PASSENGER- 006
<b>Description</b>	Describe how a user modifies his personal information.
<b>Goal</b>	[G1] Put in relationship a customer and a taxi driver.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available</li> <li>2. The customer is registered to the system.</li> </ol>
<b>Scenario</b>	
<p>John, a registered user, realizes he has inserted a wrong information about himself. In fact, he has written "Jonh" instead of "John" during the registration and he wants to correct it.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he goes to the profile section and modifies his name in the correct form. Therefore, he clicks the Save button and sends the correct name to the system.</p>	

Scenario 6: Modify personal information

Deleting from the system	
<b>Code</b>	SC-PASSENGER- 007
<b>Description</b>	Describe how a user can delete his account from the system.
<b>Goal</b>	[G1] Put in relationship a customer and a taxi driver.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available</li> <li>2. The customer is registered to the system.</li> </ol>
<b>Scenario</b>	
<p>John, a registered user, is unsatisfied with the myTaxiService application and he would like to delete his account from the system.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he goes to the profile section and pushes the Delete Account button. Therefore, a successful operation message is shown.</p>	

Scenario 7: Deleting from the system

Taxi driver registration	
<b>Code</b>	SC-TAXI-001
<b>Description</b>	Describe how a taxi driver is registered in the service.
<b>Goal</b>	[G1] Put in relationship a customer and a taxi driver.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection available</li> <li>2. Taxi driver is not registered to the system</li> </ol>
<b>Scenario</b>	
The taxi driver Mario would like to improve his job and he's looking for a solution.	

One day he is surfing the Net and he discovers myTaxiService. He is really interested and he goes to the section for taxi driver. He has to fill a form, putting his personal details, his driving license and the relative documentation about his job. At the end he sends the module. He will be contact in the next few days by the service administrator, who will confirm or decline his proposal.

Scenario 1: taxi driver registration

Accept/Decline taxi request	
<b>Code</b>	SC-TAXI-002
<b>Description</b>	Describe how a taxi driver can accept/decline a taxi request
<b>Goal</b>	[G4] Manage a system of taxis' reservations and requests
<b>Assumptions</b>	3. The customer is registered to the system 4. Internet connection available 5. Taxi driver is registered to the system
<b>Scenario</b>	
<p>The taxi driver Luca is waiting in his car. His mobile phone receives a notification from myTaxiService application. He has a new pending request from the customer John. The application gives him the position of the passenger.</p> <p>Luca has a problem with his car and he cannot take care of John's request. He clicks on "decline request".</p> <p>Another taxi driver Paolo is waiting for a taxi request. His mobile phone receives a notification from myTaxiService application. He has a new pending request from the customer John. The application gives him the position of the passenger.</p> <p>Paolo is free and he can take care of John's request. He clicks on "Accept Request" and he sends the confirmation with the expected time arrival. The system takes in charge everything and informs immediately John.</p>	

Scenario 2: Updating a pending request (delay)

Update a pending request (delay)	
<b>Code</b>	SC-TAXI-003
<b>Description</b>	Describe how a pending request is updated
<b>Goal</b>	[G4] Manage a system of taxis' reservations and requests
<b>Assumptions</b>	5. The customer is registered to the system 6. Internet connection available 7. The customer must have a taxi request. 8. Taxi driver is registered to the system
<b>Scenario</b>	
<p>The taxi driver Paolo, while is driving to John to pick him up, encounters a traffic jam due to an accident. He wants to tell John he arrives with a delay of few minutes. Therefore, he open myTaxiService and change the expected arrival time.</p>	

Scenario 3: Update a pending request (delay)

Update a pending request (car problem)	
<b>Code</b>	SC-TAXI-004
<b>Description</b>	Describe how a pending request is updated
<b>Goal</b>	[G4] Manage a system of taxis' reservations and requests
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>5. The customer is registered to the system</li> <li>6. Internet connection available</li> <li>7. The customer must have a taxi request.</li> <li>8. The taxi driver is registered to the system</li> </ol>
<b>Scenario</b>	
The taxi driver Paolo, while is driving to John to pick him up, has a problem with his car. He wants to tell John that he cannot take care of his request and that another taxi is going to arrive. Therefore, he open myTaxiService and cancel the pending request.	

Scenario 4: Updating a pending request (car problem)

Cancel a pending request	
<b>Code</b>	SC-TAXI- 005
<b>Description</b>	Describe how a pending request is cancelled
<b>Goal</b>	[G4] Manage a system of taxis' reservations and request
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>6. The customer is registered to the system</li> <li>7. Internet connection available</li> <li>8. The customer must have a taxi request.</li> <li>9. The taxi driver is registered to the system</li> </ol>
<b>Scenario</b>	
The taxi driver, who has to pick John up, receives the notification on his mobile phone that he has cancelled the request. Now he has to follow the instructions, which the system gives to him (change area or stay in that).	

Scenario 5: Cancel a pending request

Deleting from the system	
<b>Code</b>	SC-TAXI- 006
<b>Description</b>	Describe how a taxi driver can remove from the system.
<b>Goal</b>	[BG1] Simplify the access of passengers to the service.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available.</li> <li>2. The taxi driver is already registered in the system.</li> </ol>
<b>Scenario</b>	
<p>Matt, a taxi driver, is unsatisfied with the myTaxiService applications and he would like to delete his account from the system.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he goes to the profile section and clicks the Delete Account button. Therefore, a successful operation message is shown.</p>	

Scenario 6: Deleting from the system

Add a new taxi zone	
<b>Code</b>	SC-ADMIN-001
<b>Description</b>	Describe how a new taxi zone is added
<b>Goal</b>	[BG1] Simplify the access of passengers to the service.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available</li> <li>2. Admin is already registered in the system</li> </ol>
<b>Scenario</b>	
<p>The admin would like to add a new taxi zone in order to improve the management of the taxis in the city.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he fills out the form with all details of the new taxi zone, such as the minimum number of taxis and the position on the map and clicks the “add” button.</p>	

Scenario 1: Add a new taxi zone

Modify the minimum number of taxis in a zone	
<b>Code</b>	SC-ADMIN-002
<b>Description</b>	Describe how a taxi zone is modified
<b>Goal</b>	[BG1] Simplify the access of passengers to the service.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available</li> <li>2. Admin is already registered in the system</li> <li>3. Taxi zone is already present in the system</li> </ol>
<b>Scenario</b>	
<p>Due to a new opening of a fair in a certain zone of the city and the consequent growth of requests for taxis, the admin would like to increase the minimum number of taxis in that zone.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he selects the zone interested, and modify the minimum number of taxis. After he presses the appropriate button to send the modified zone to the system.</p>	

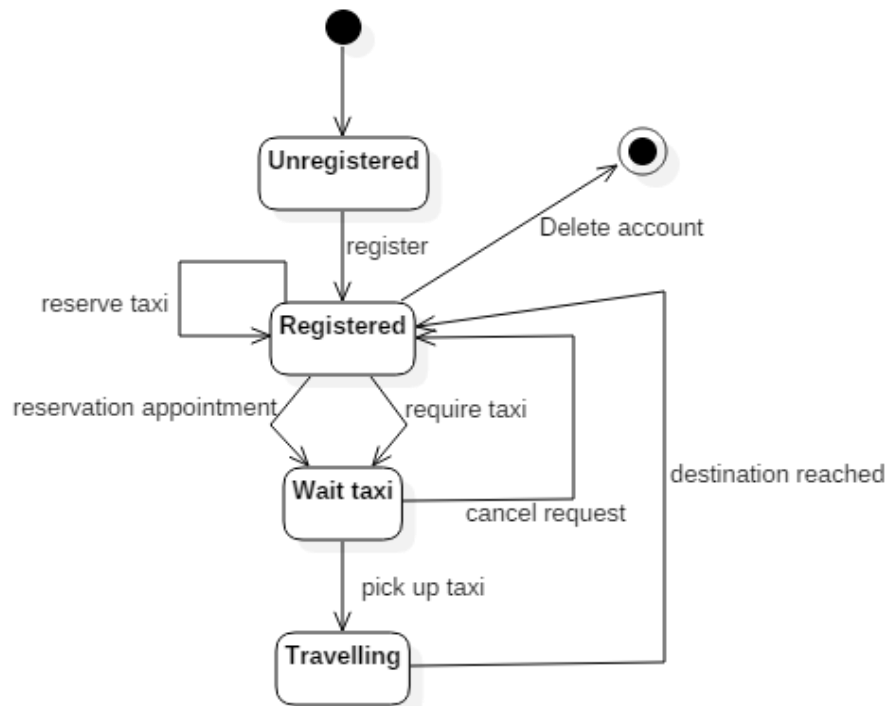
Scenario 2: Modify the minimum number of taxi in a zone

Delete a taxi zone	
<b>Code</b>	SC-ADMIN-003
<b>Description</b>	Describe how a taxi zone is deleted
<b>Goal</b>	[BG1] Simplify the access of passengers to the service.
<b>Assumptions</b>	<ol style="list-style-type: none"> <li>1. Internet connection is available</li> <li>2. Admin is already registered in the system</li> <li>3. Taxi zone is already present in the system</li> </ol>
<b>Scenario</b>	
<p>The admin realizes that he have made a mistake entering a wrong zone and he would like to delete it.</p> <p>He connects to the internet and goes to the myTaxiService homepage. After logged in, he selects the zone interested and clicks the “Delete” button. Therefore, a successful operation message is shown.</p>	

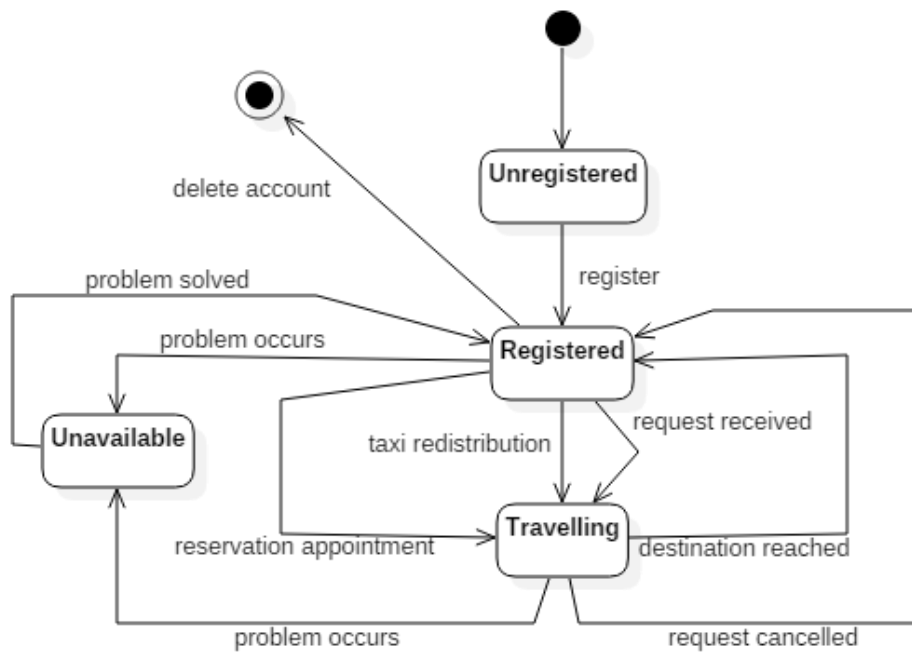
Scenario 3: Delete a taxi zone



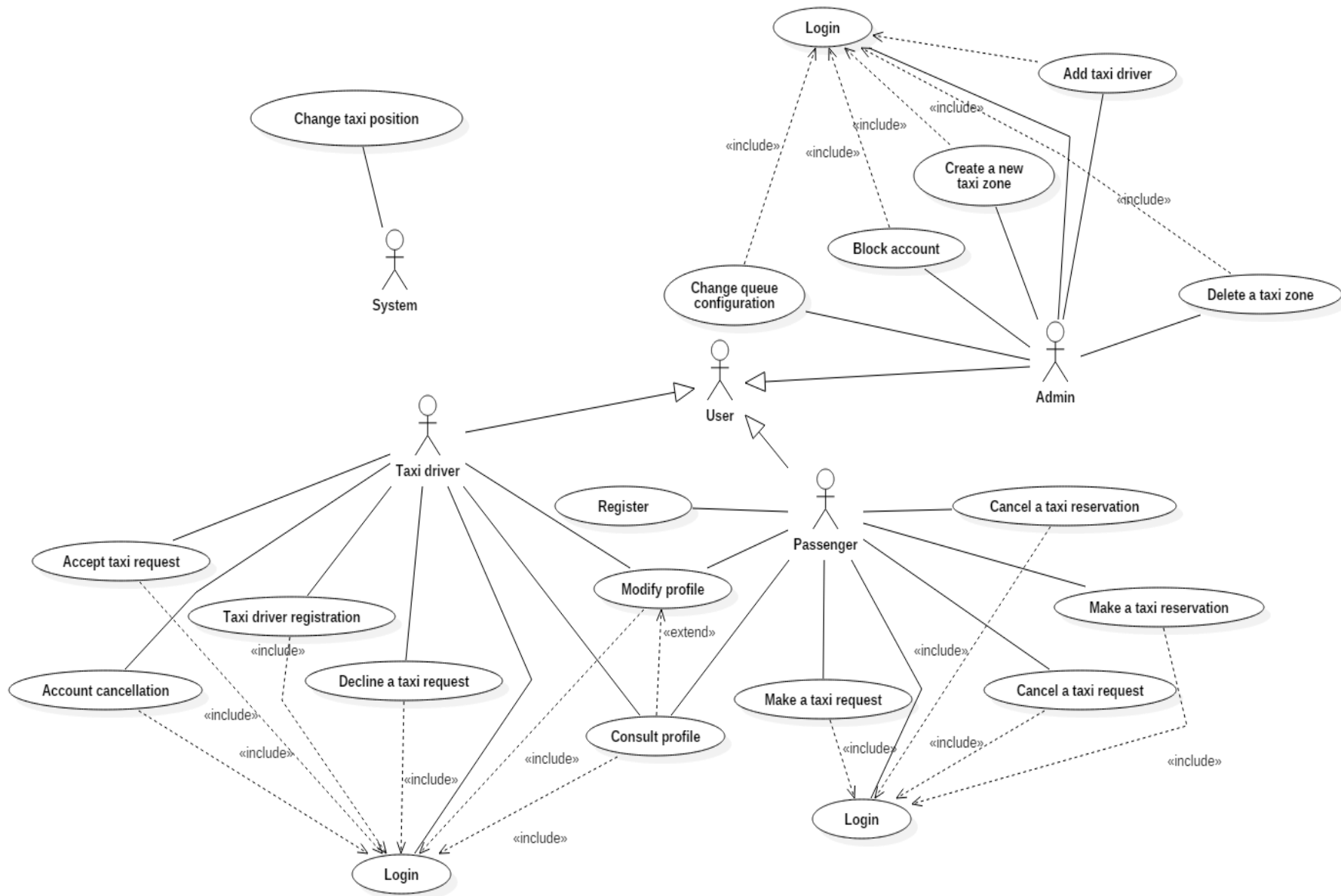




State chart 1: Passenger user state chart



State chart 2: Taxi driver state chart



General Use Cases Diagram

### 4.2.3 Use Cases

During this phase, we identified the following actors:

- System administrator
- System service
- Taxi driver
- Passenger

The following diagram shows the general use cases for each of these actors, in the next subsections we explained them in detail:

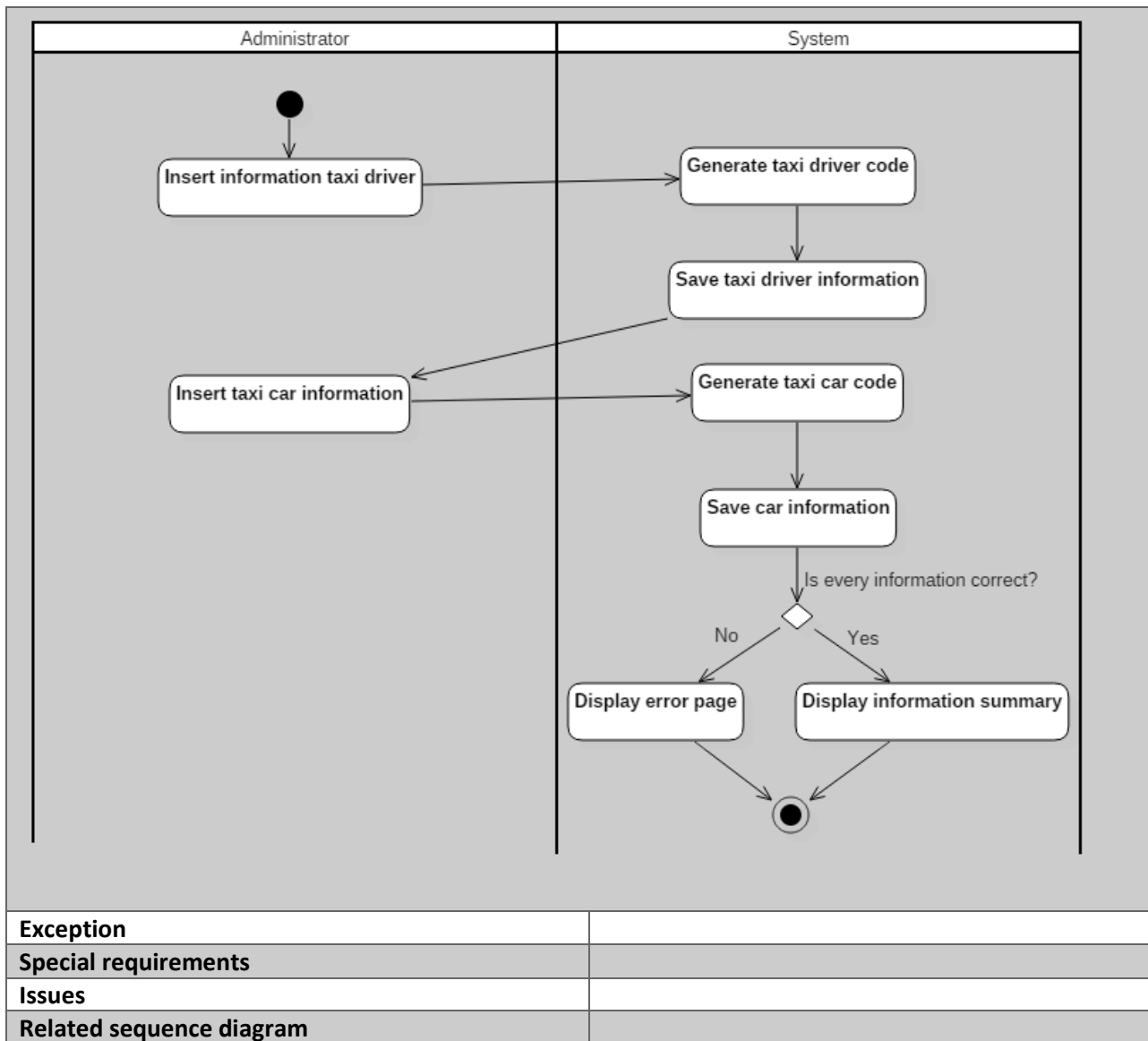
#### 4.2.3.1 System administrator

We have identified the following use cases, which belongs to system administrator:

1. New taxi registration.
2. Change queue configuration.
3. Add a taxi zone.
4. Remove a taxi zone
5. Block an account.

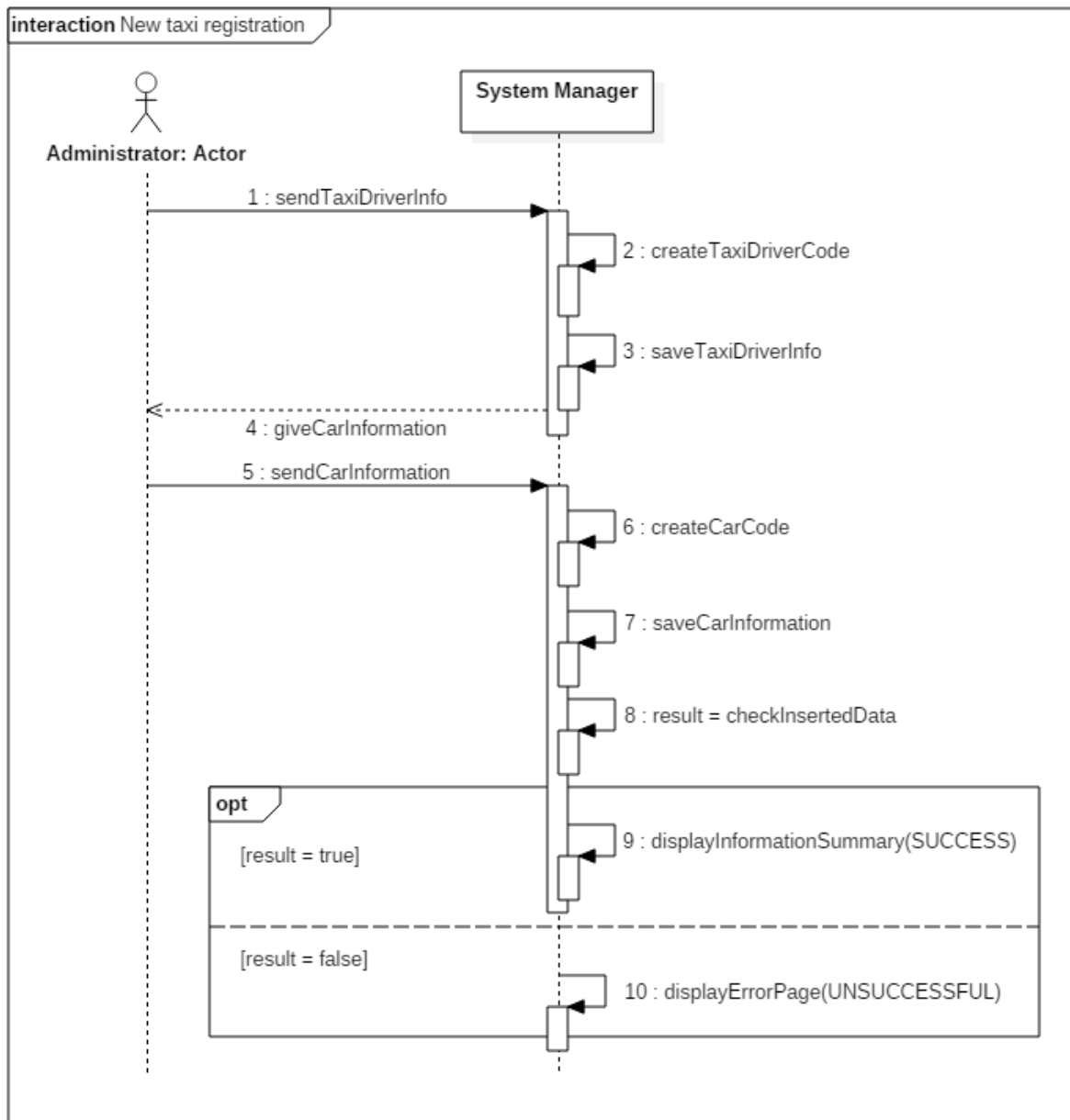
##### 4.2.3.1.1 New taxi registration

New taxi registration	
<b>Code</b>	UC-ADMIN-001
<b>Description</b>	The admin can register a new taxi in the system
<b>Goal</b>	[G1] [G2]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Administrator
<b>Entry condition</b>	Administrator is logged in
<b>Exit condition</b>	A new taxi is registered in the system
<b>Flow of event</b> (see next page)	

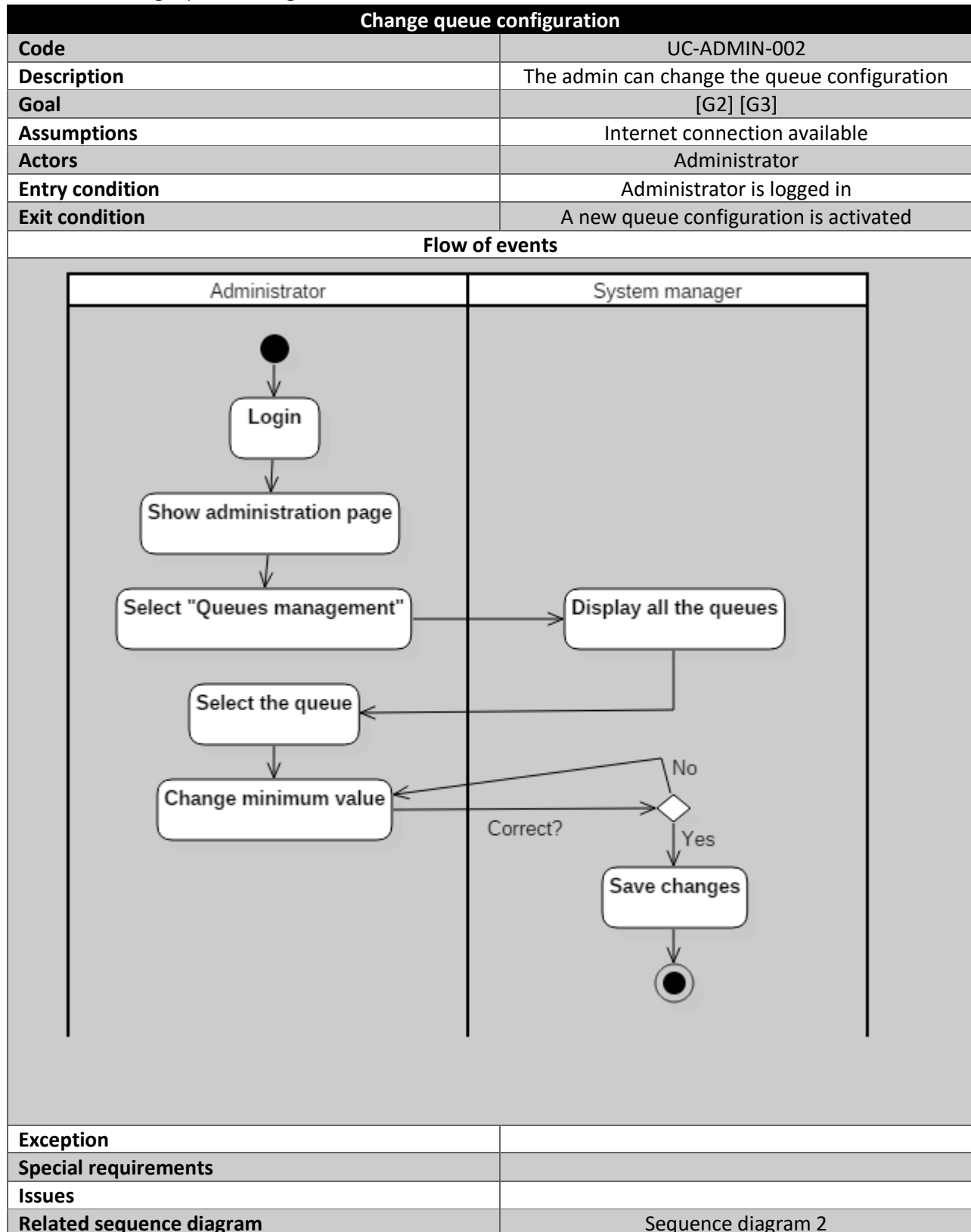


Use Cases 1: New taxi registration

Sequence diagram 1

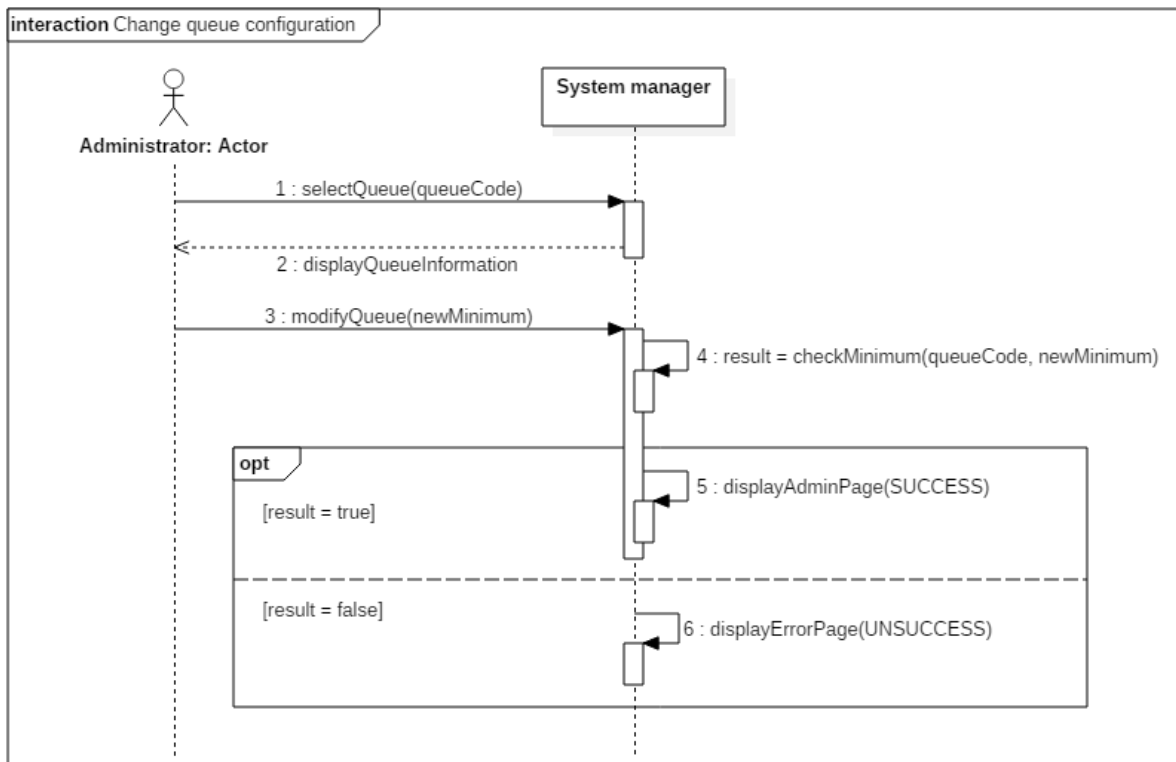


#### 4.2.3.1.2 Change queue configuration



Use cases 2: change queue configuration

## Sequence diagram 2

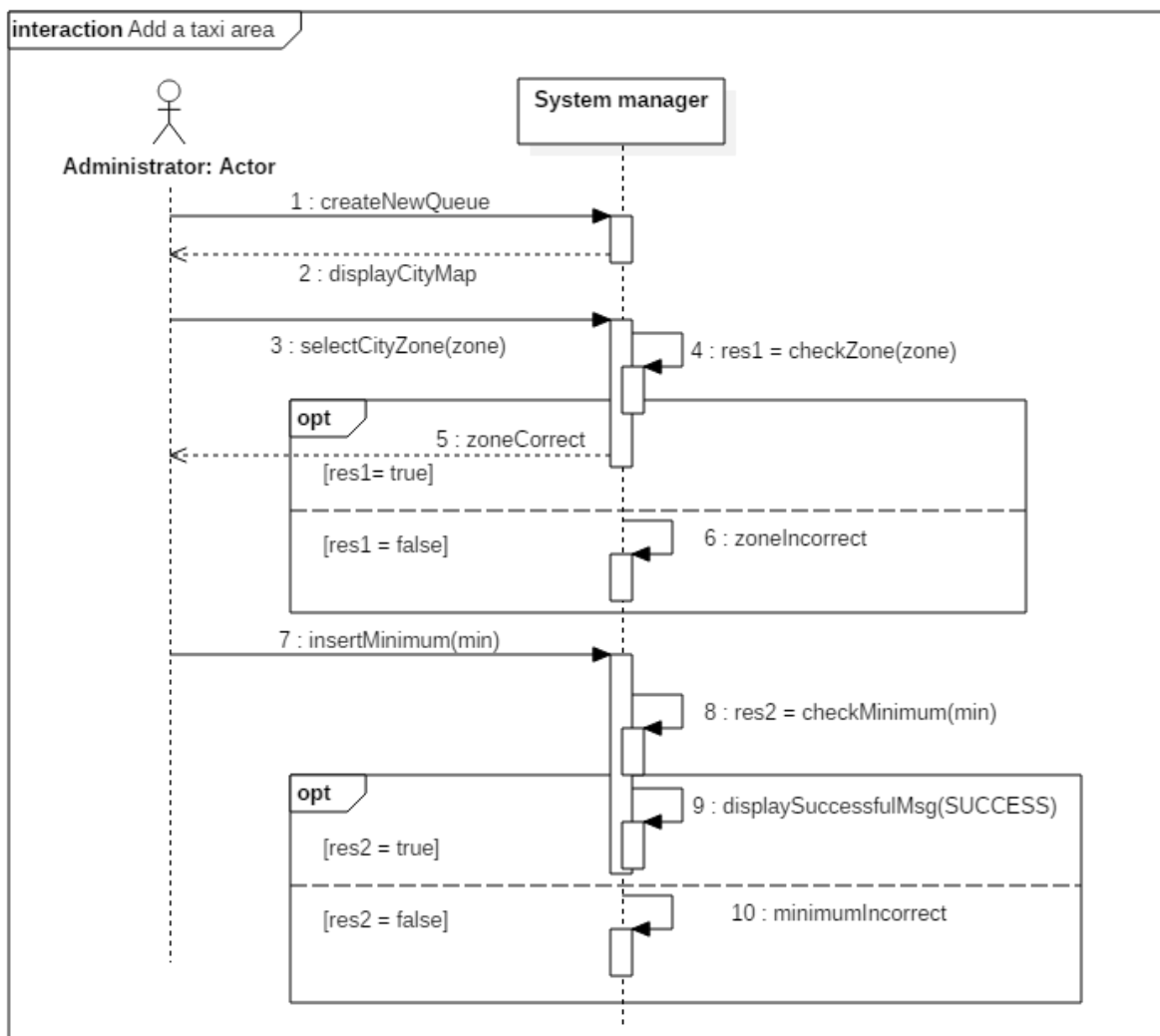


### 4.2.3.1.3 Add a taxi area

Add a taxi area	
Code	UC-ADMIN-003
Description	The admin can add a new taxi area
Goal	[G2] [G3]
Assumptions	Internet connection available
Actors	Administrator
Entry condition	Administrator is logged in
Exit condition	A new taxi area is added
Flow of events	
Exception	
Special requirements	Every new area must be approximately two km <sup>2</sup> , with a minimum of one km <sup>2</sup> and a maximum of three
Issues	
Related sequence diagram	Sequence diagram 3

Use Case 3: Add a taxi area

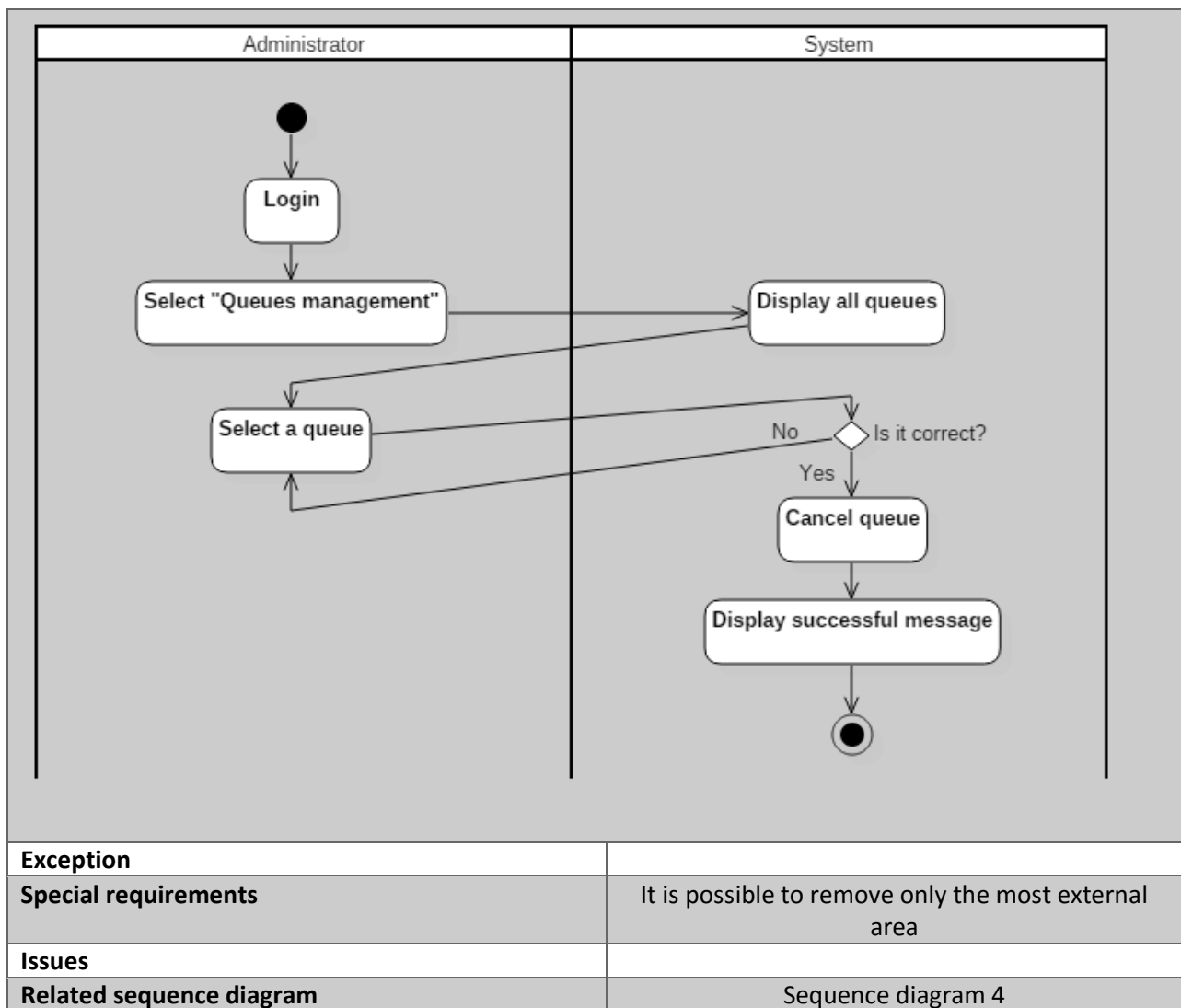
Sequence diagram 3



#### 4.2.3.1.4 Remove a taxi area

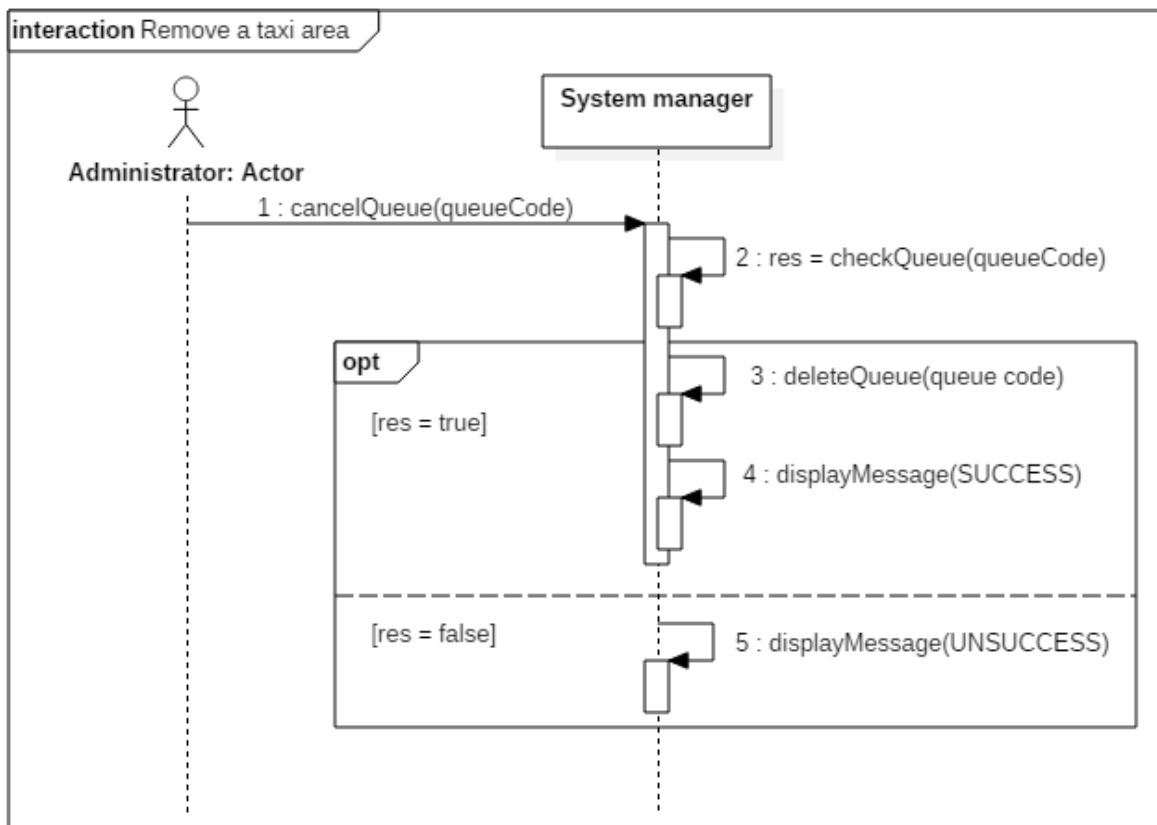
Remove a taxi area	
Code	UC-ADMIN-004
Description	The admin can remove a taxi area
Goal	[G2] [G3]
Assumptions	Internet connection available
Actors	Administrator
Entry condition	The administrator must be logged in
Exit condition	A taxi area is removed
Flow of events	





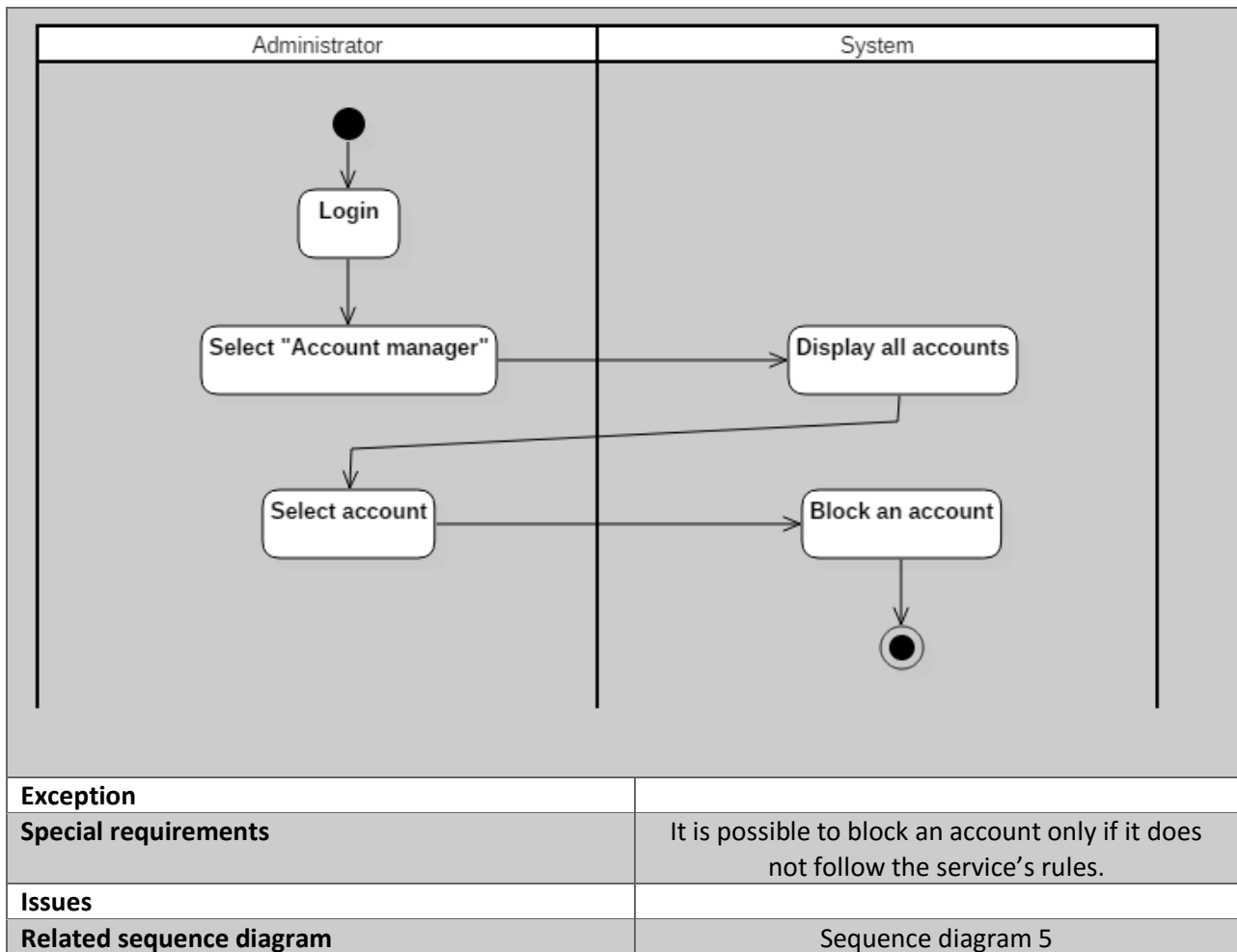
Use Cases 4: remove a taxi area

Sequence diagram 4



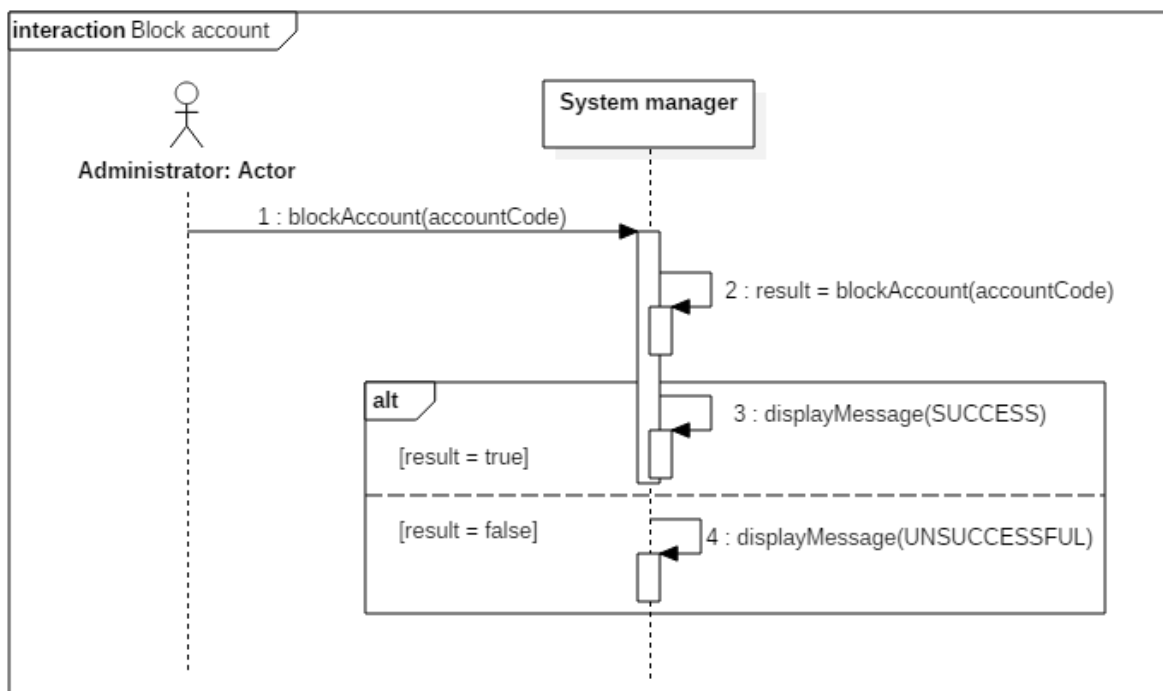
#### 4.2.3.1.5 Block an account

Block an account	
Code	UC-ADMIN-005
Description	The admin can block an account
Goal	[G2] [G3]
Assumptions	Internet connection available
Actors	Administrator
Entry condition	The administrator must be logged in
Exit condition	An account is blocked
Flow of events	



Use cases 5: Block an account

Sequence diagram 5



#### 4.2.3.2 System service

We have identified the following use cases, which belongs to system service:

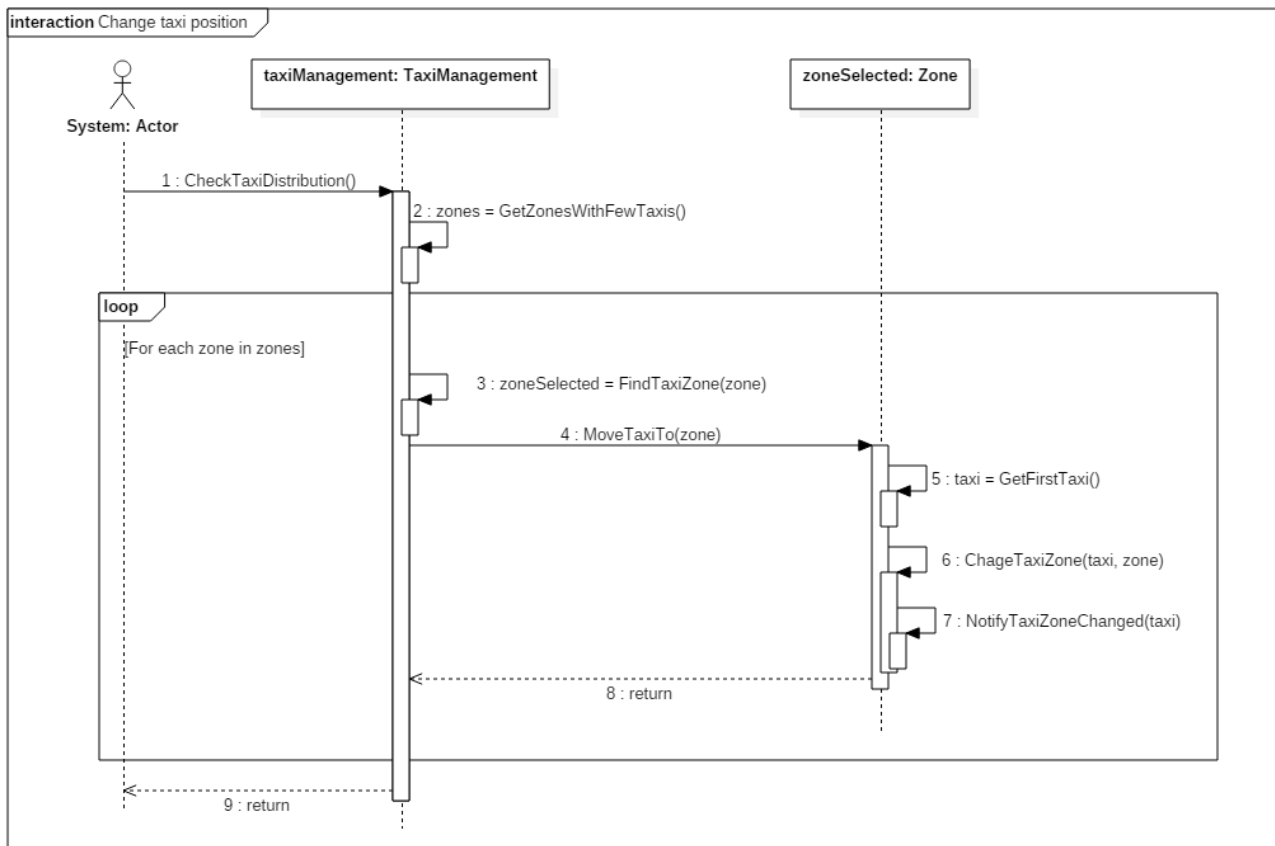
1. Change taxi position.

##### 4.2.3.2.1 Change taxi position

Change taxi position	
Code	UC-SYS-001
Description	How the system can change the area assigned to a taxi.
Goal	[G2]
Assumptions	Internet connection available
Actors	System
Entry condition	The number of the taxis of an area is under the minimum.
Exit condition	The positions of the taxis have changed.
Flow of events	
<pre>graph TD     Start(( )) --&gt; Select[Select the adjacent zones]     Select --&gt; Find[Find the adjacent zone with the largest number of taxis available]     Find --&gt; Found{Found?}     Found -- yes --&gt; Move[Move one taxi to the zone]     Found -- no --&gt; End((( )))     Move --&gt; End</pre> <p>The diagram is a UML Use Case Diagram for the 'Change taxi position' use case. It is contained within a rectangular frame labeled 'System'. The process begins with a solid black circle (start node) at the top. An arrow points down to a rounded rectangle labeled 'Select the adjacent zones'. Another arrow points down to a second rounded rectangle labeled 'Find the adjacent zone with the largest number of taxis available'. From there, an arrow points down to a diamond-shaped decision node labeled 'Found?'. A 'yes' branch leads left and then down to a rounded rectangle labeled 'Move one taxi to the zone'. A 'no' branch leads right and then down to a bullseye symbol (end node). Both the 'Move one taxi to the zone' box and the 'no' branch lead to the same bullseye end node.</p>	
Exception	
Special requirements	
Issues	

## Use case 1: Change taxi position

Sequence Diagram 1



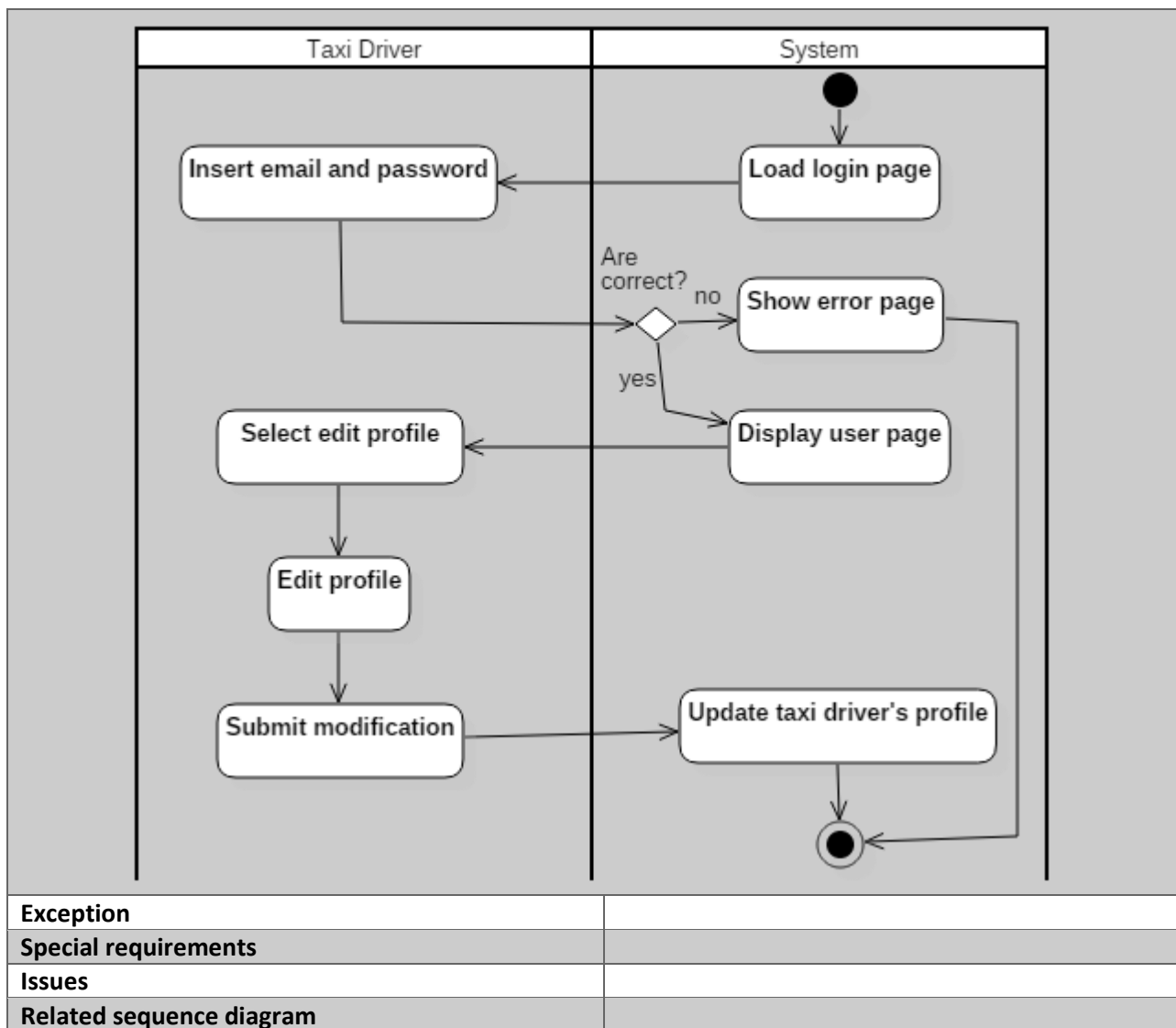
## 4.2.3.3 Taxi driver

We have identified the following use cases, which belongs to taxi driver:

1. Update profile.
2. Consult profile.
3. Accept taxi request.
4. Decline a taxi request.
5. Taxi driver registration.
6. Account cancellation.

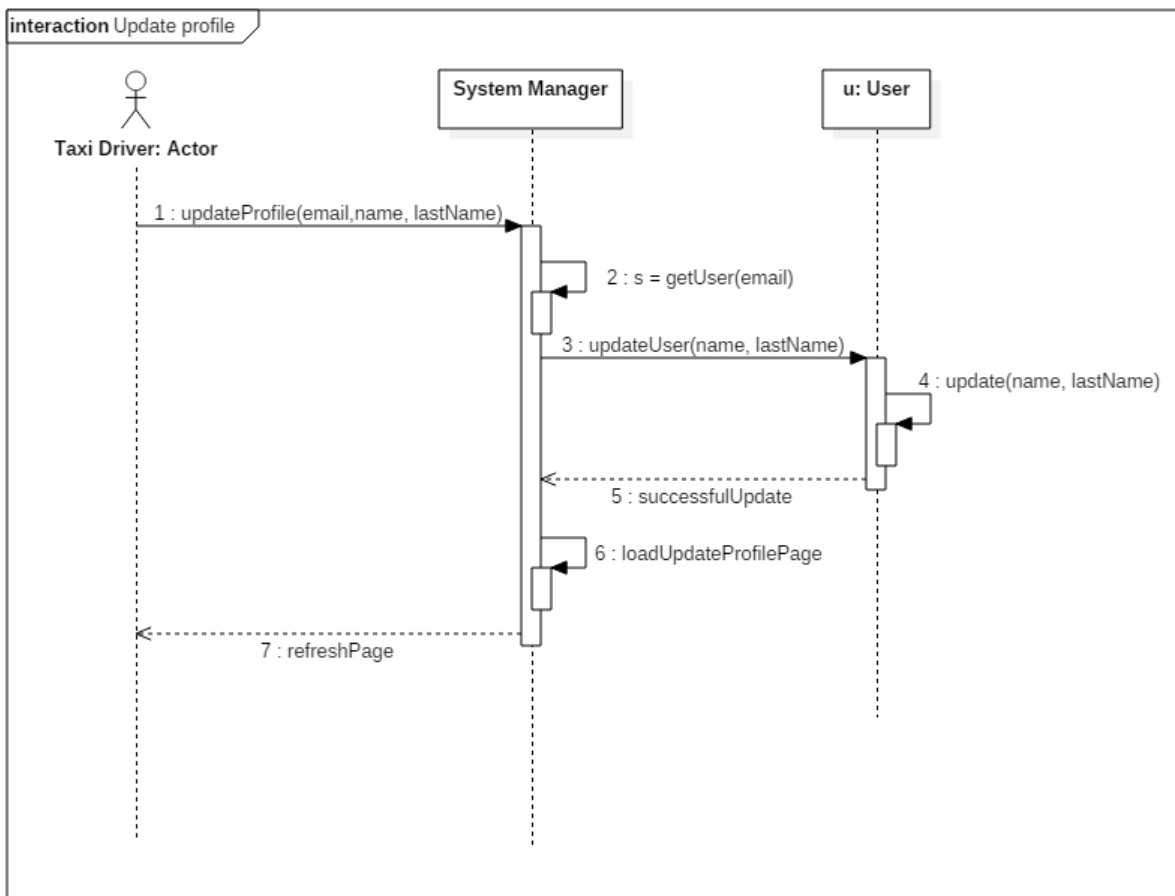
## 4.2.3.3.1 Update profile

Update profile	
Code	UC-TAXI-001
Description	Every taxi driver can update its profile
Goal	[G1]
Assumptions	Internet connection available
Actors	Taxi driver
Entry condition	The taxi driver logs in his homepage.
Exit condition	The taxi driver's profile is updated.
Flow of events	



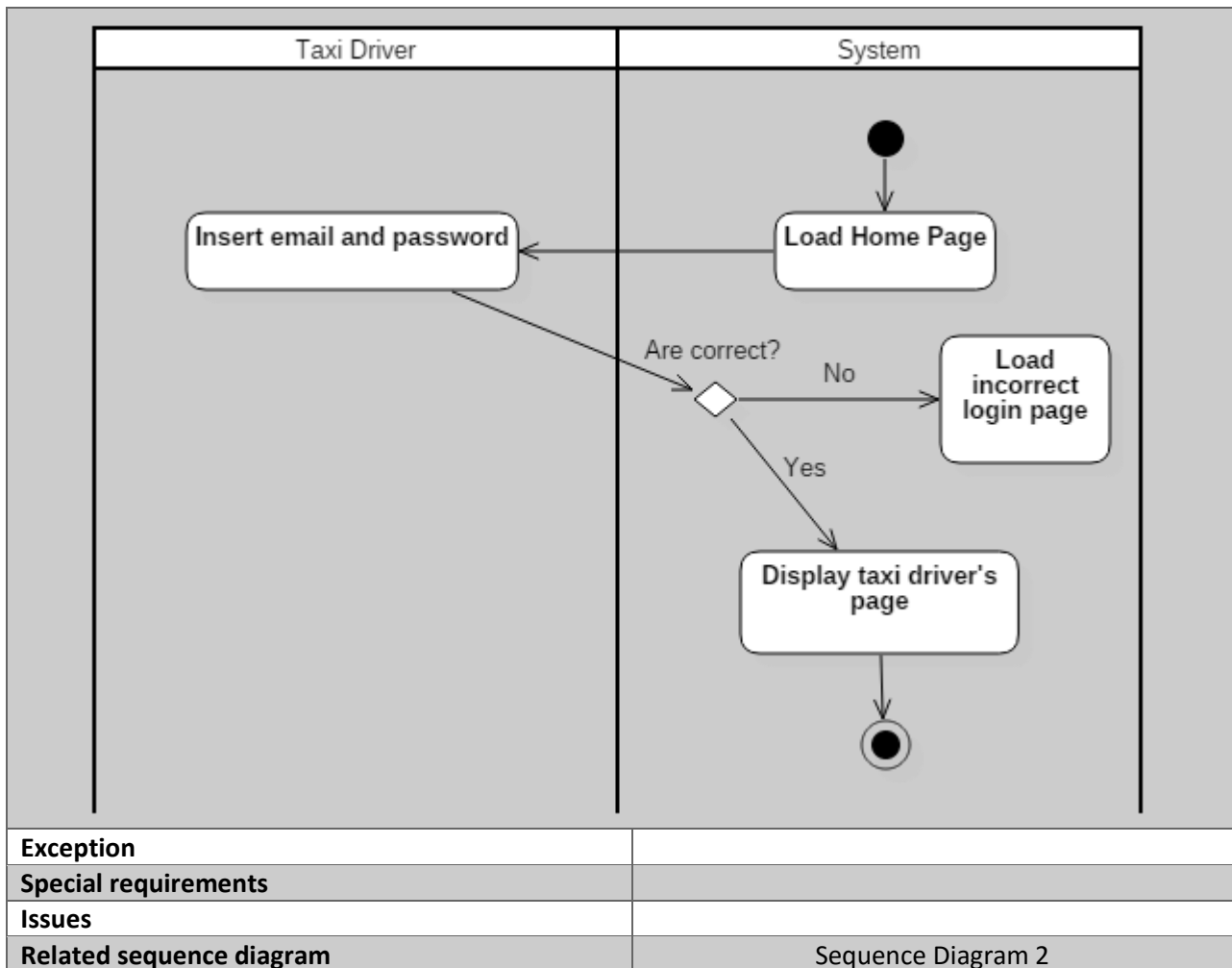
Use case 1: Update profile

## Sequence Diagram 1



### 4.2.3.3.2 Consult profile

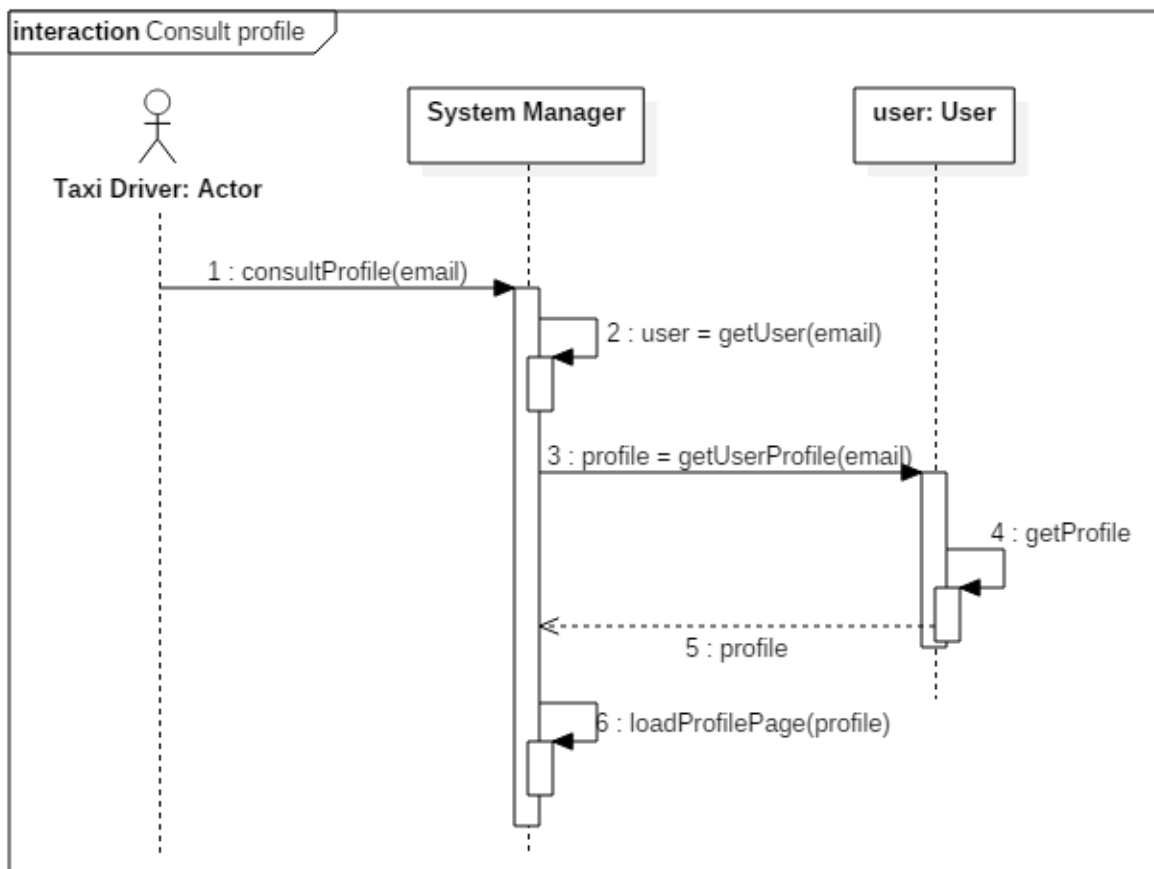
Consult Profile	
<b>Code</b>	UC-TAXI-002
<b>Description</b>	A taxi driver can consult its profile.
<b>Goal</b>	[G1]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Taxi Driver
<b>Entry condition</b>	The taxi driver logs in his homepage.
<b>Exit condition</b>	Taxi driver's profile is shown.
Flow of events	



Use case 2: Consult profile

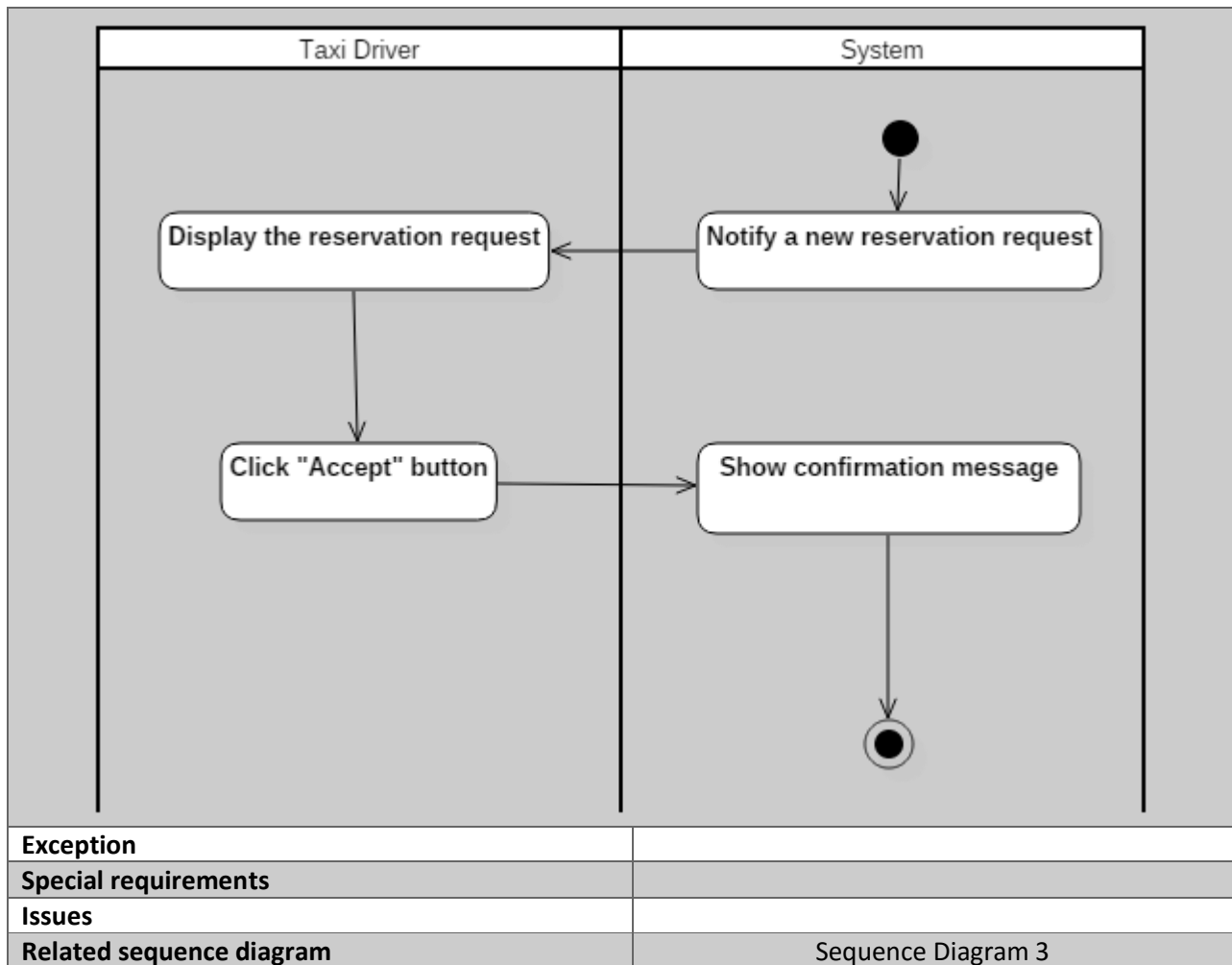


## Sequence Diagram 2



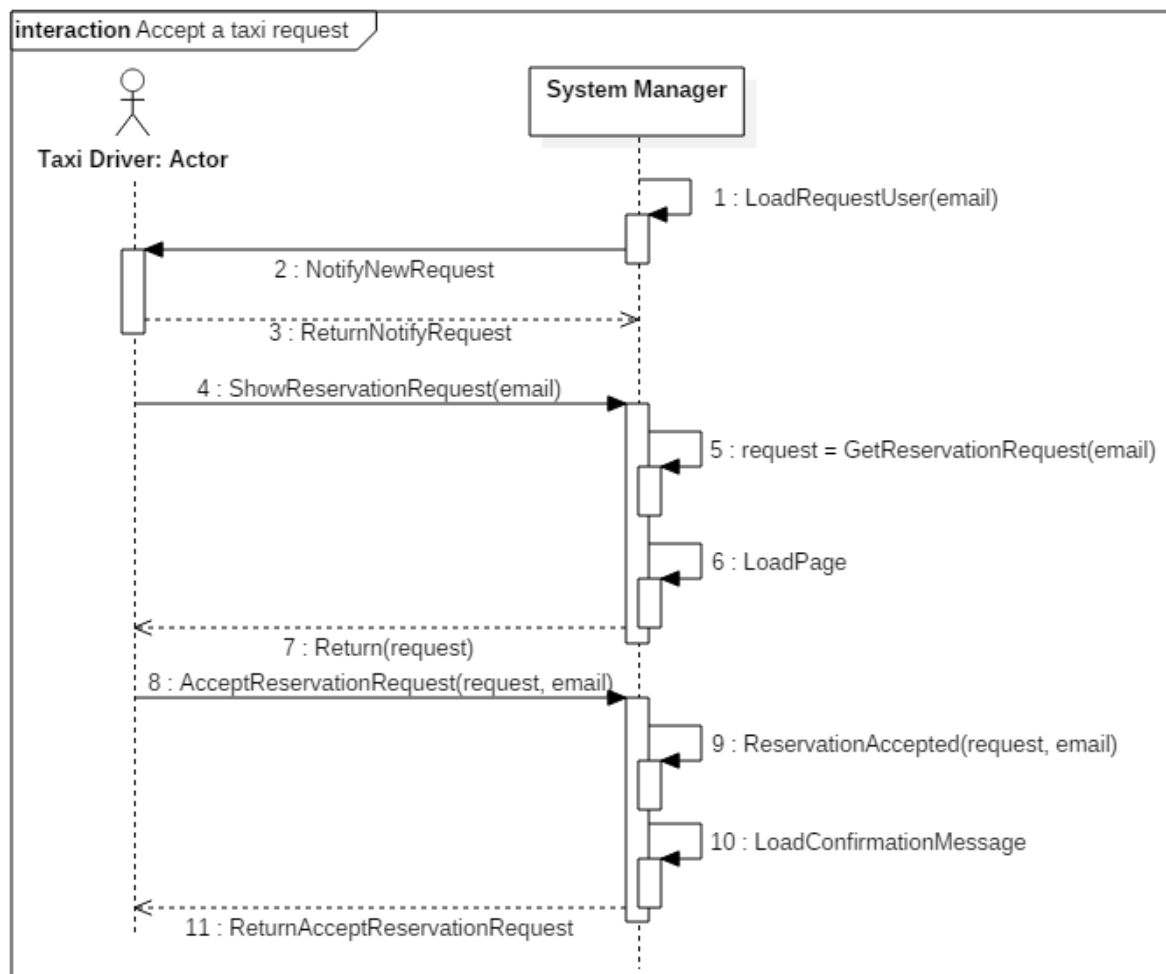
### 4.2.3.3.3 Accept taxi request

Accept taxi request	
Code	UC-TAXI-003
Description	How a taxi driver can accept a request.
Goal	[G1] [G4]
Assumptions	Internet connection available. Taxi driver must be logged in.
Actors	Taxi Driver
Entry condition	The system sends to the taxi driver a new reservation request.
Exit condition	The taxi driver accepts the request.
Flow of events	



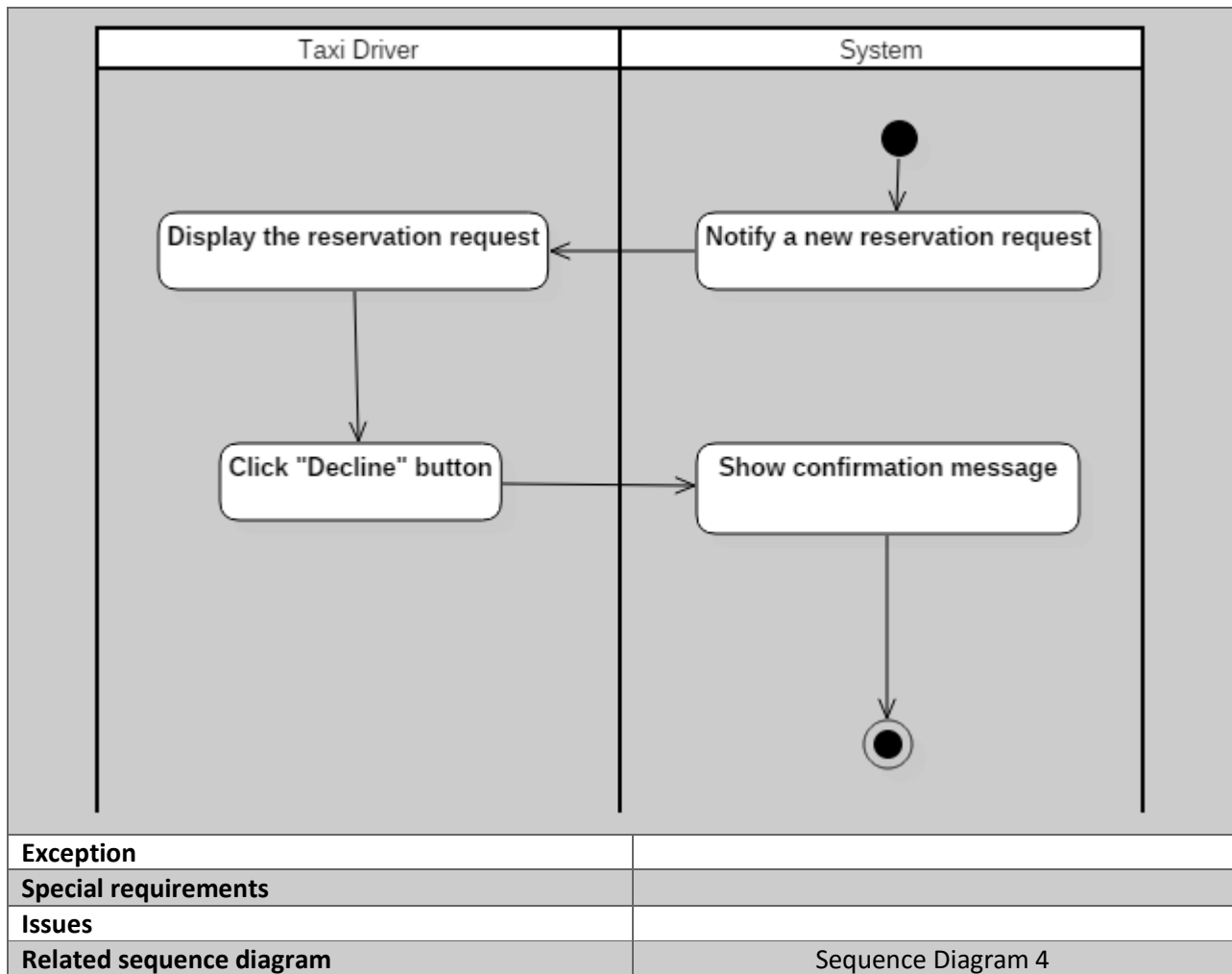
Use case 3: Accept taxi request

Sequence Diagram 3



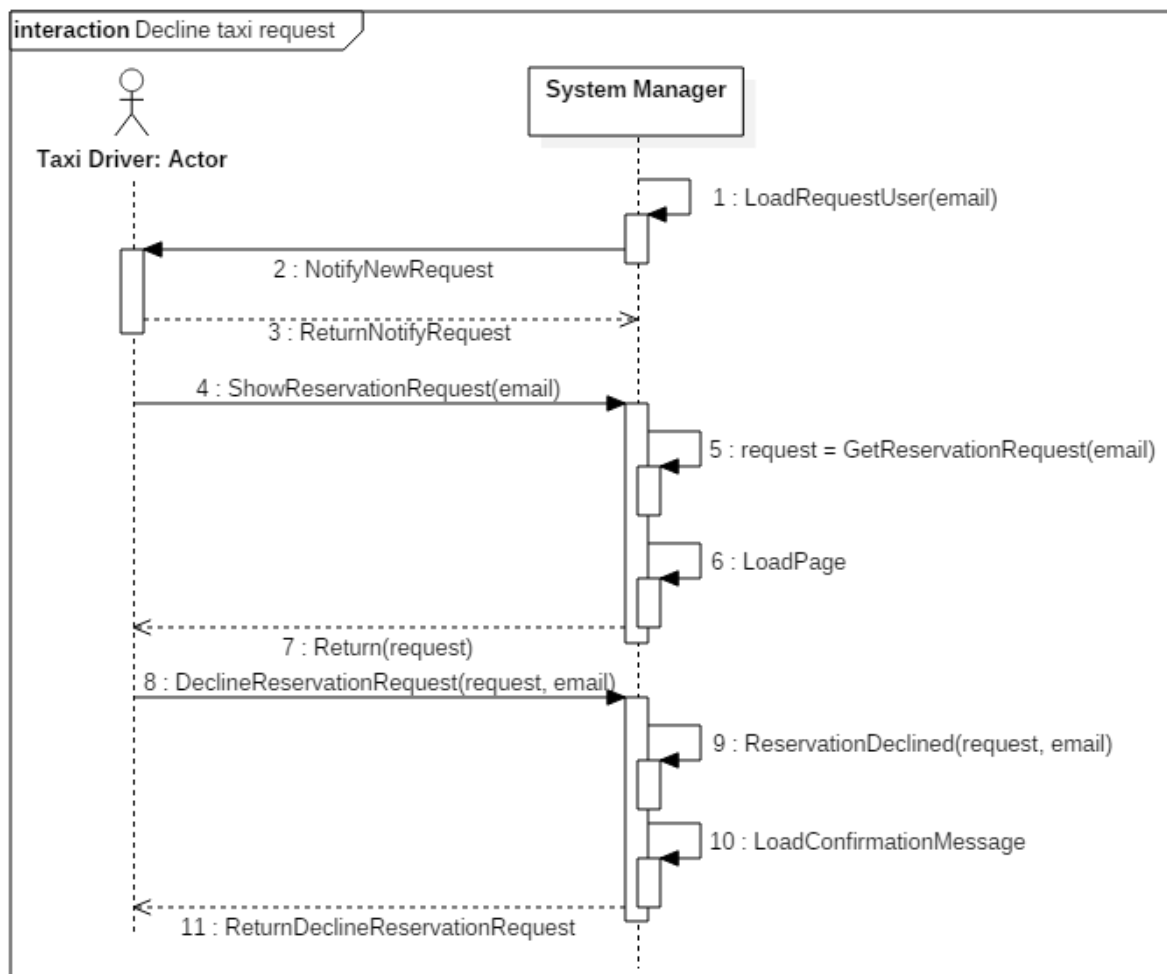
#### 4.2.3.3.4 Decline taxi request

Decline taxi request	
<b>Code</b>	UC-TAXI-004
<b>Description</b>	How a taxi driver can decline a taxi request.
<b>Goal</b>	[G1] [G4]
<b>Assumptions</b>	Internet connection available. Taxi driver must be logged in.
<b>Actors</b>	Taxi driver
<b>Entry condition</b>	The system sends to the taxi driver a new reservation request.
<b>Exit condition</b>	The taxi driver declines the request.
Flow of events	



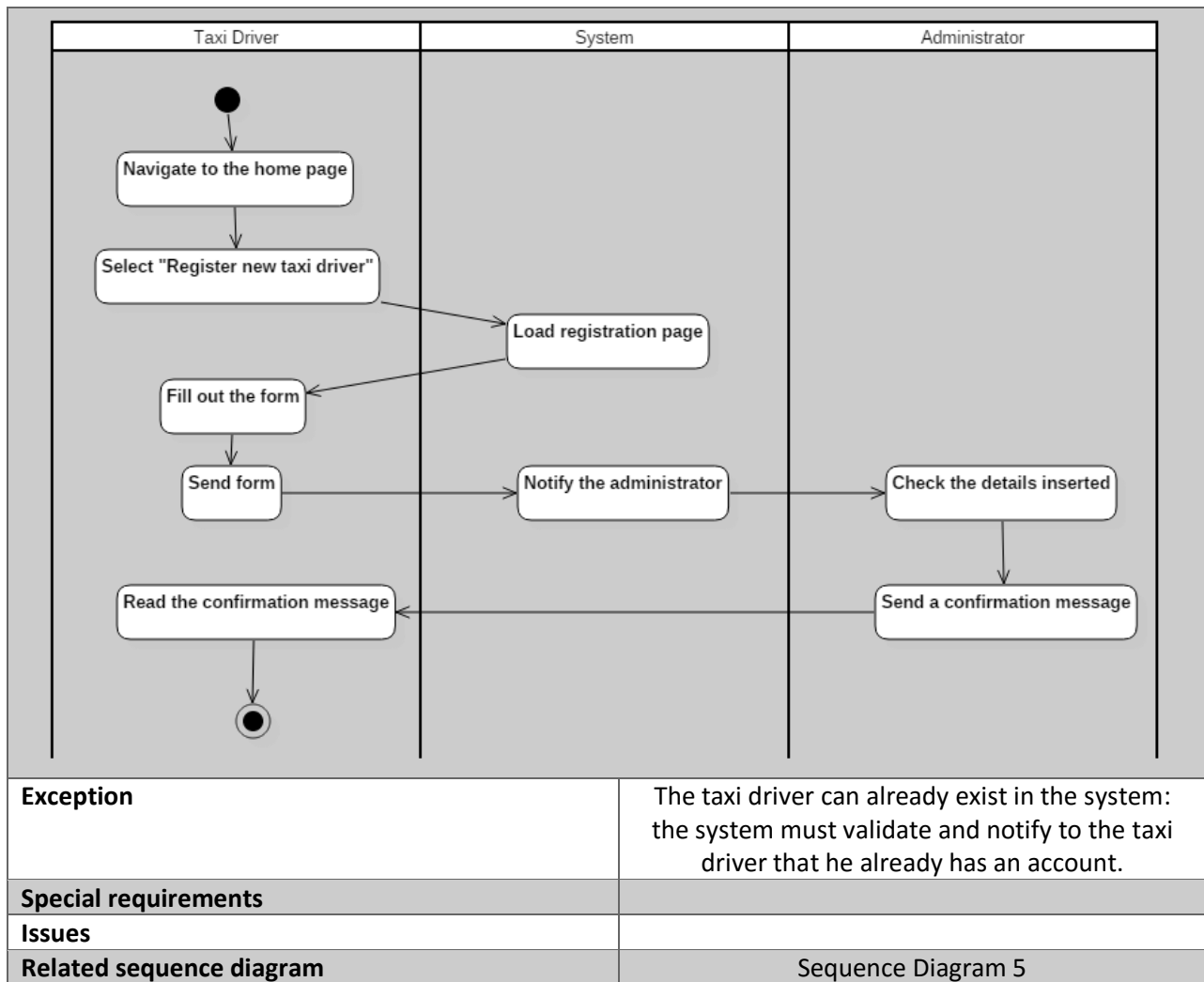
Use case 4:Decline taxi request

Sequence Diagram 4



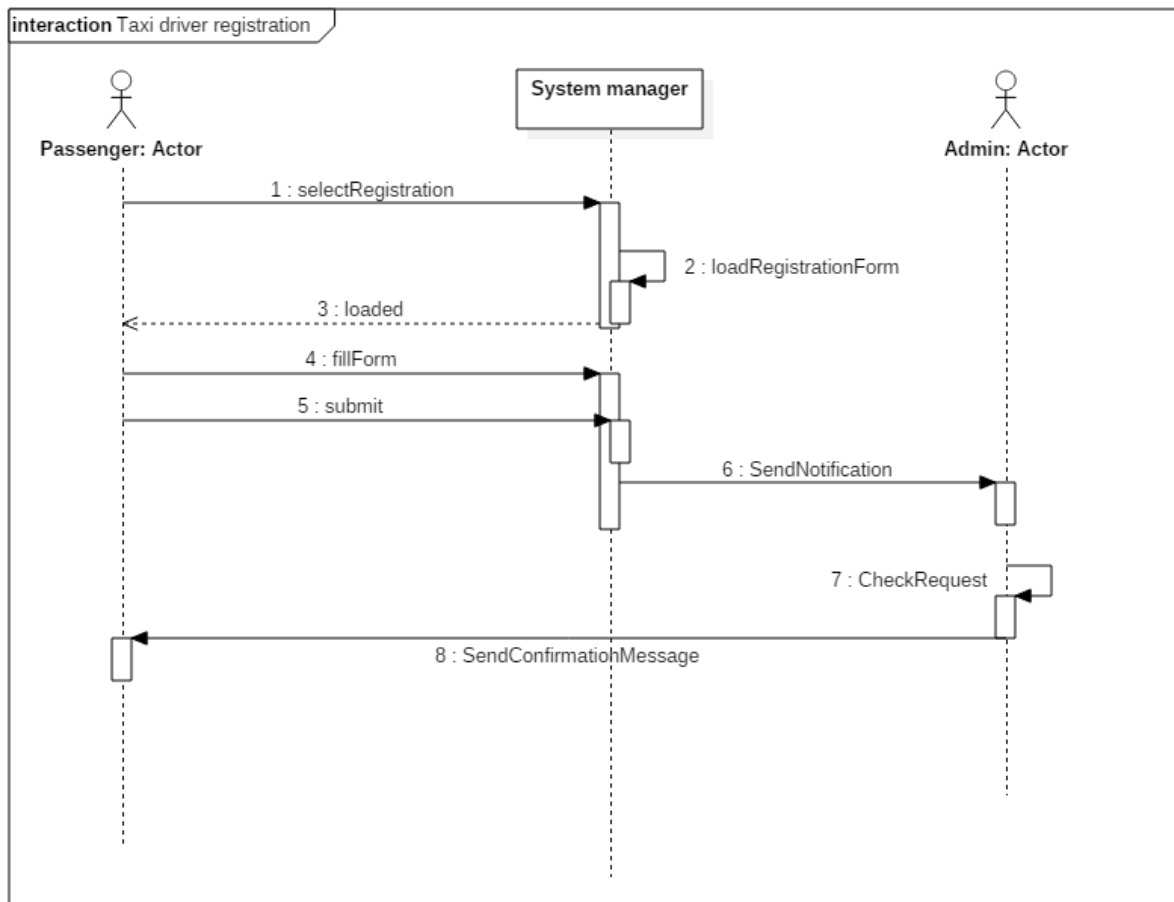
#### 4.2.3.3.5 Taxi driver registration

Taxi driver registration	
<b>Code</b>	UC-TAXI-005
<b>Description</b>	How a taxi driver can register in the system.
<b>Goal</b>	[G1]
<b>Assumptions</b>	Internet connection available. The taxi driver is not registered in the system.
<b>Actors</b>	Taxi Driver, Administrator
<b>Entry condition</b>	The taxi driver is not registered.
<b>Exit condition</b>	The registration of the taxi driver
Flow of events	



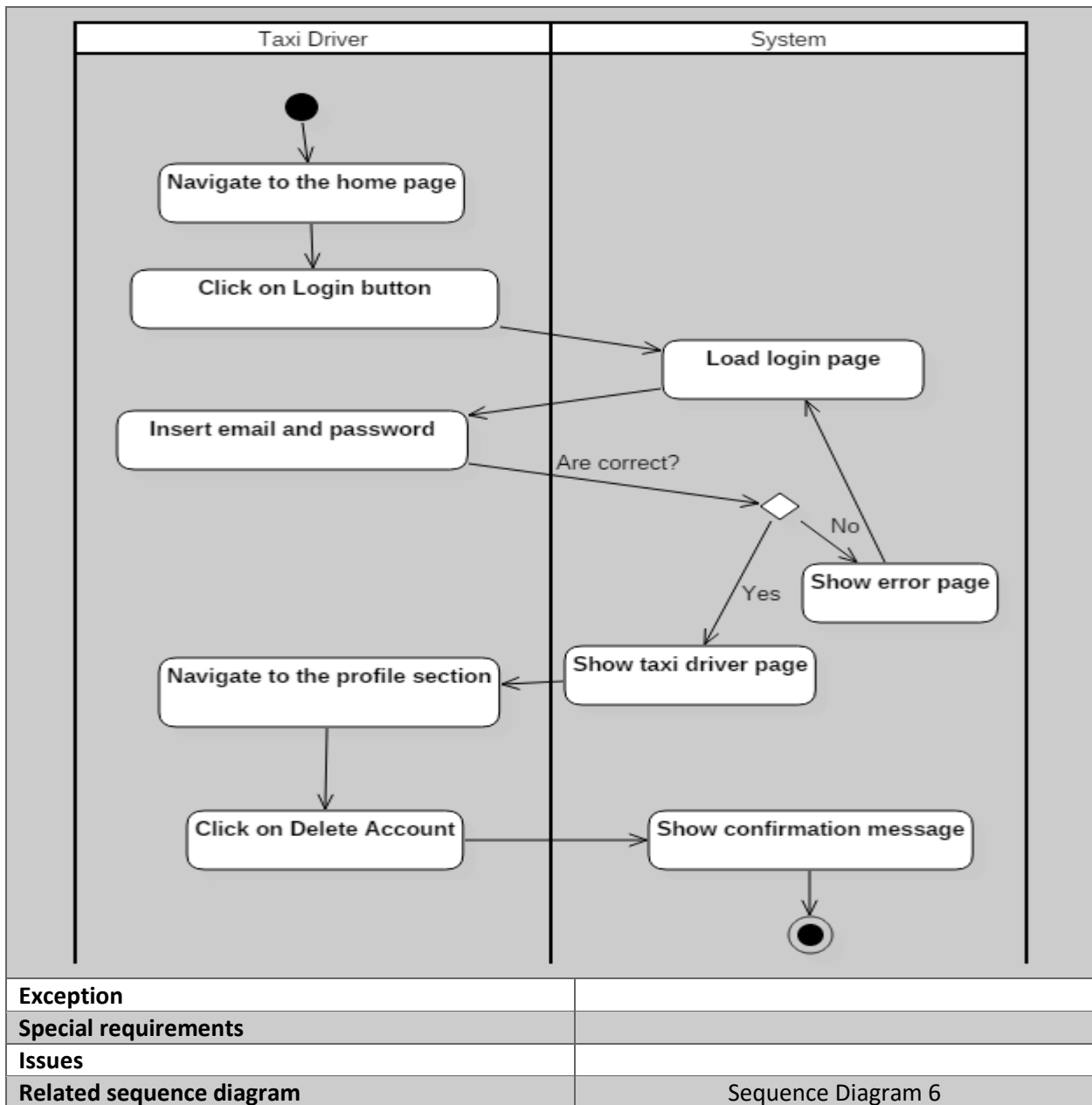
Use case 5: Taxi driver registration

Sequence Diagram 5



#### 4.2.3.3.6 Account cancellation

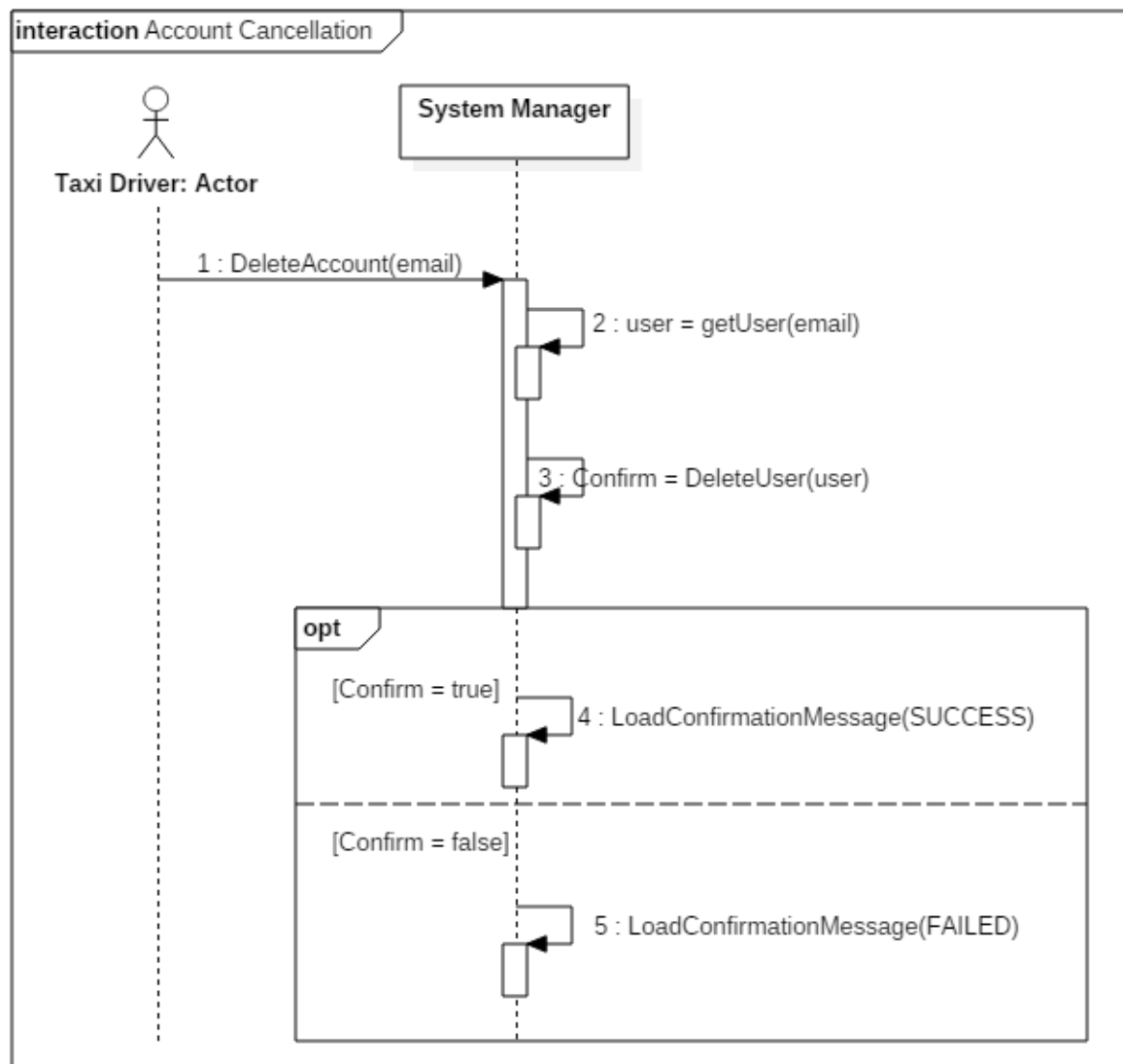
Account cancellation	
Code	UC-TAXI-006
Description	How a taxi driver can delete his account.
Goal	[G1]
Assumptions	Internet connection available. The taxi driver is registered in the system.
Actors	Taxi Driver
Entry condition	The taxi driver is registered.
Exit condition	The taxi driver is no longer registered.
Flow of events	



Use case 6: account cancellation



Sequence Diagram 6



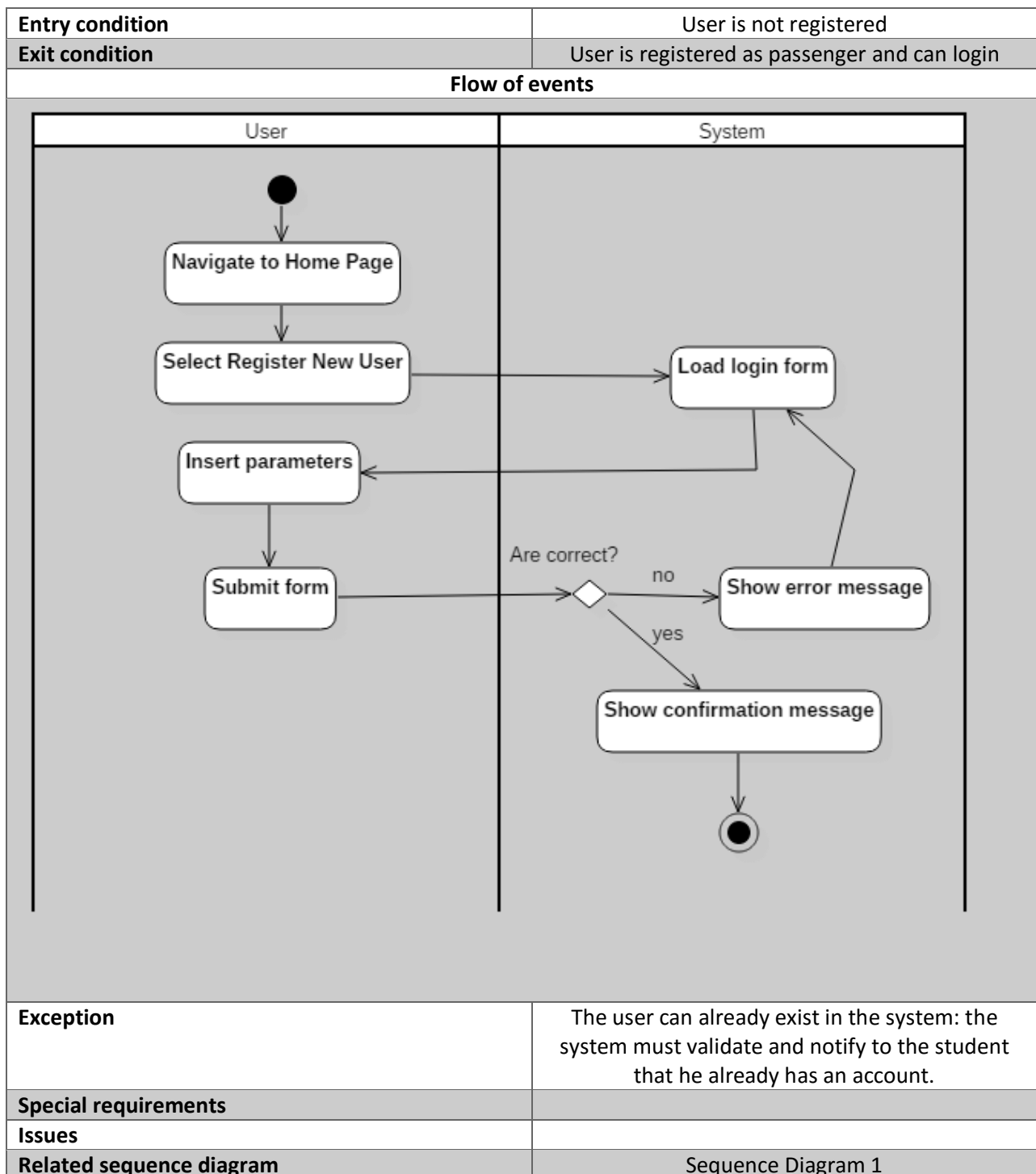
#### 4.2.3.4 Passenger

We have identified the following use cases, which belongs to taxi passenger:

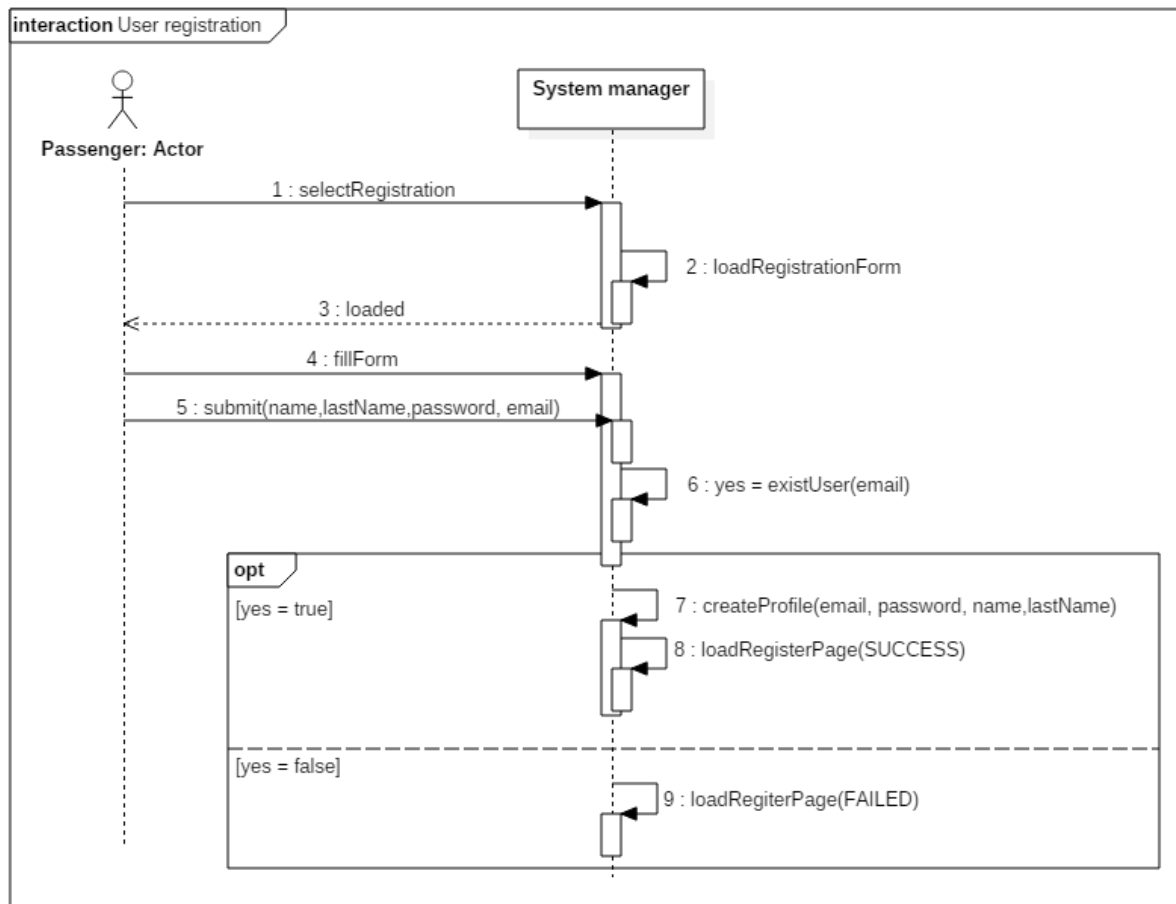
1. User registration.
2. Consult profile.
3. Update profile.
4. Make a taxi request.
5. Cancel a taxi request.
6. Make a taxi reservation.
7. Cancel a taxi reservation.

##### 4.2.3.4.1 User registration

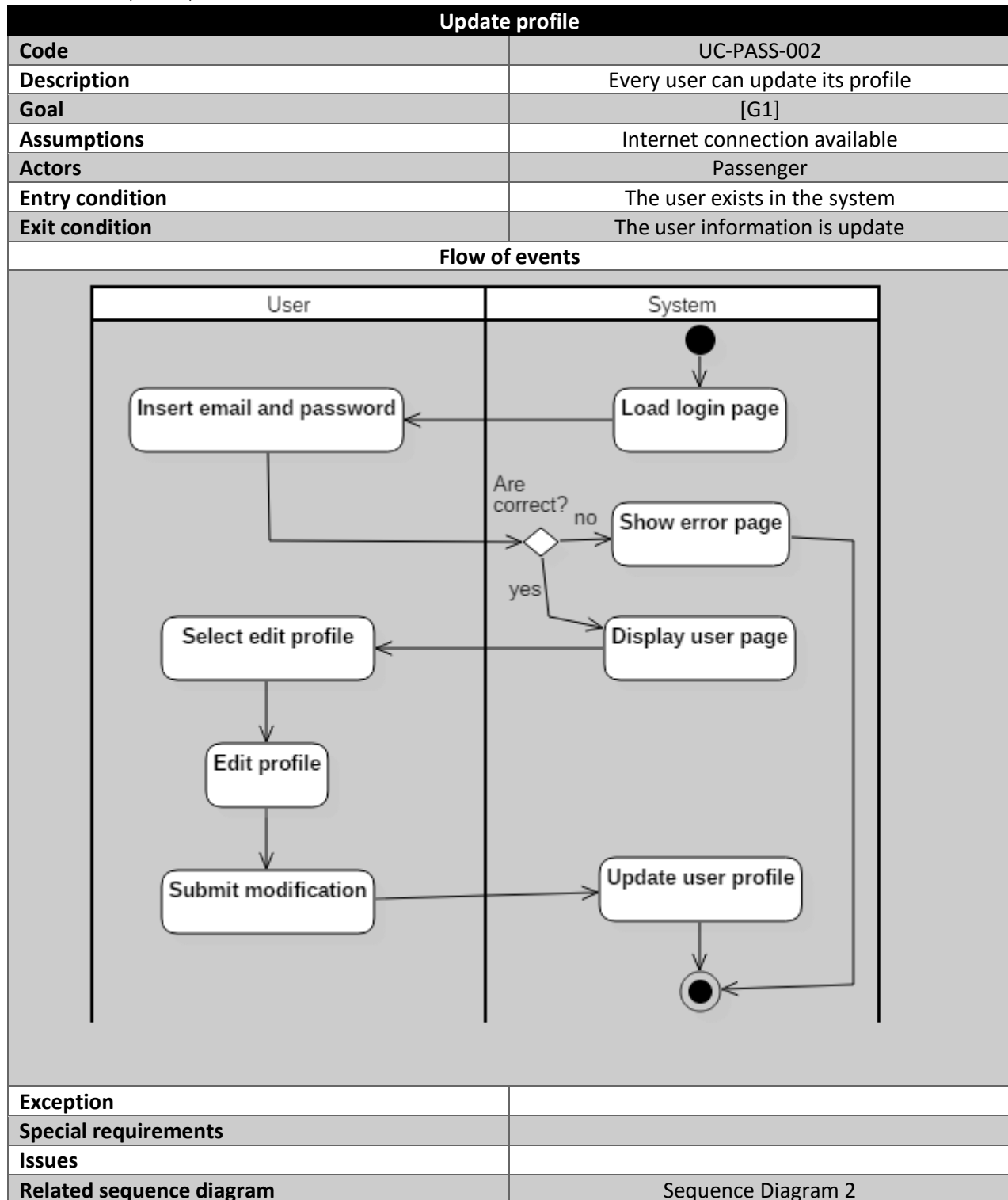
User registration	
<b>Code</b>	UC-PASS-001
<b>Description</b>	Describe how a user can register
<b>Goal</b>	[G1]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Passenger



## Sequence Diagram 1

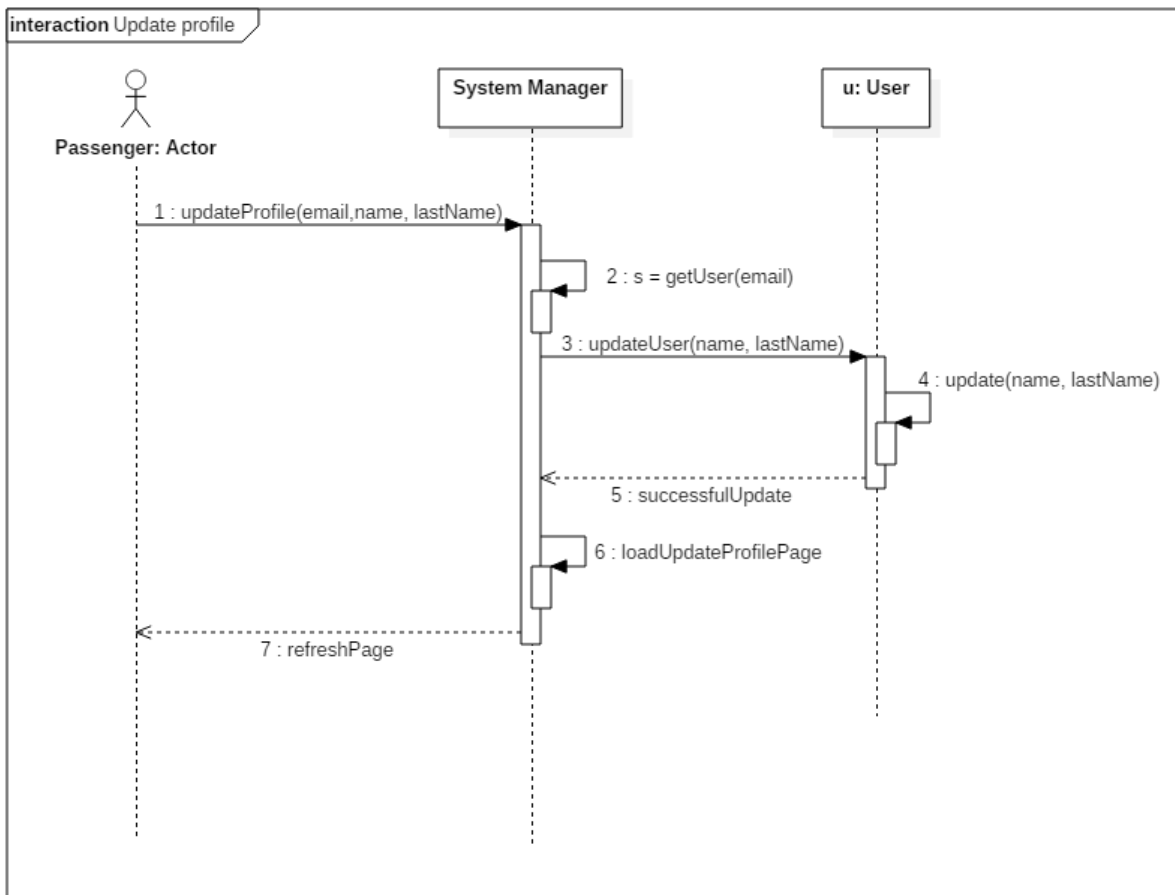


#### 4.2.3.4.2 Update profile

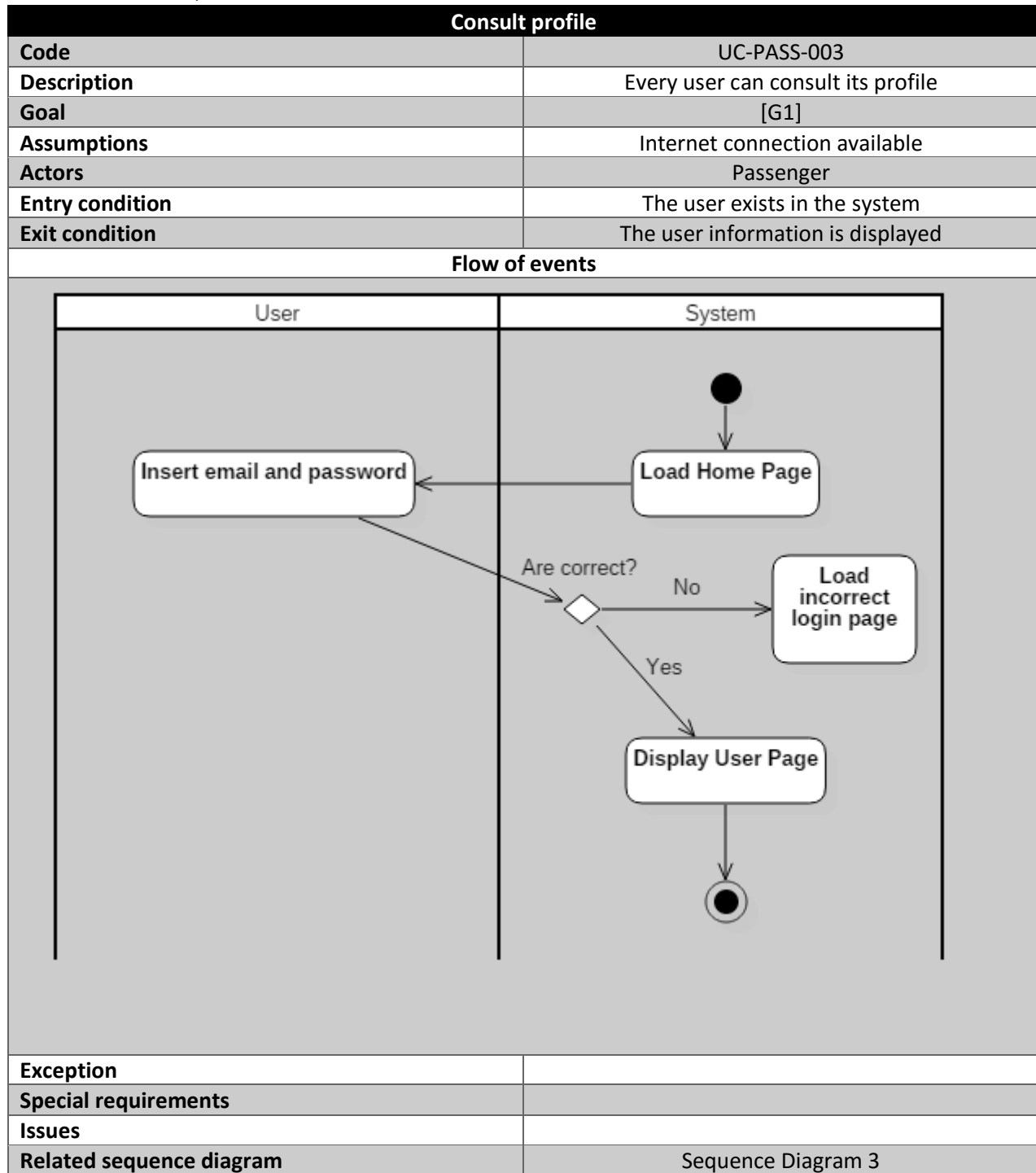


Use Case 2: Update profile

## Sequence Diagram 2

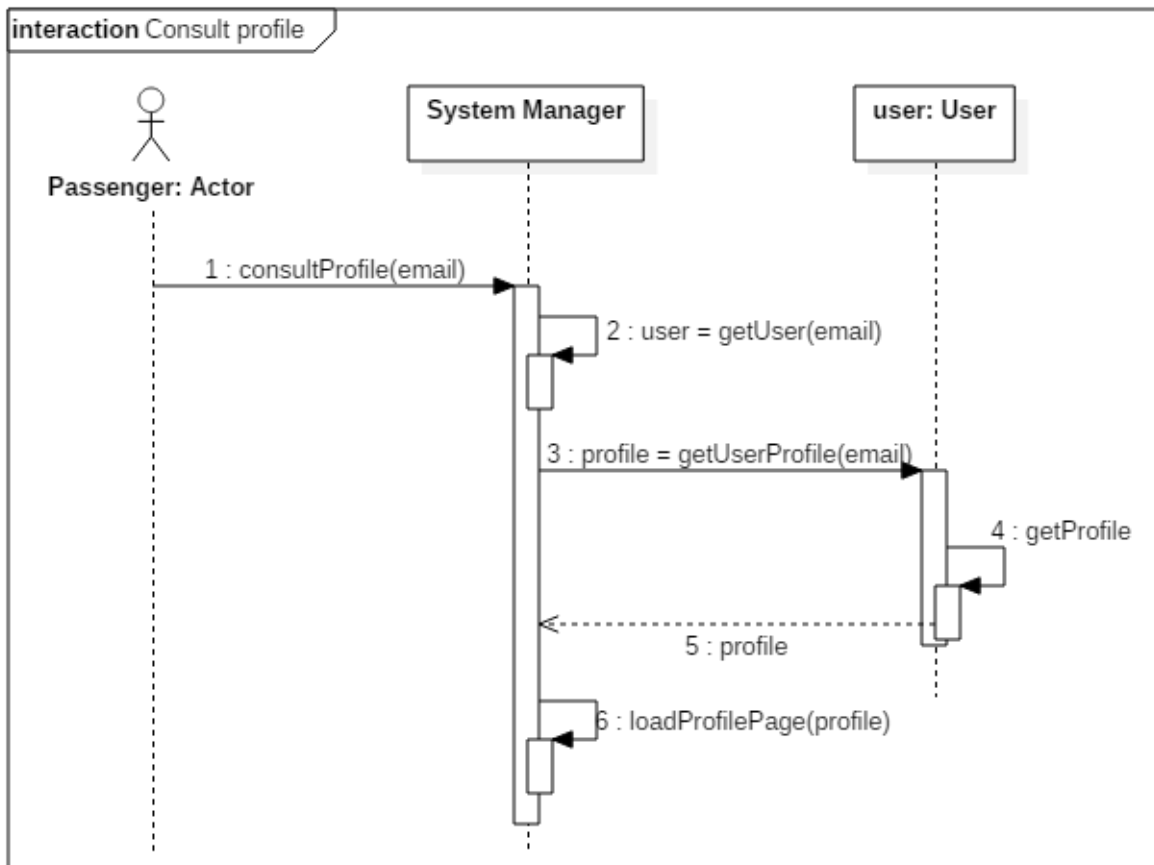


#### 4.2.3.4.3 Consult profile



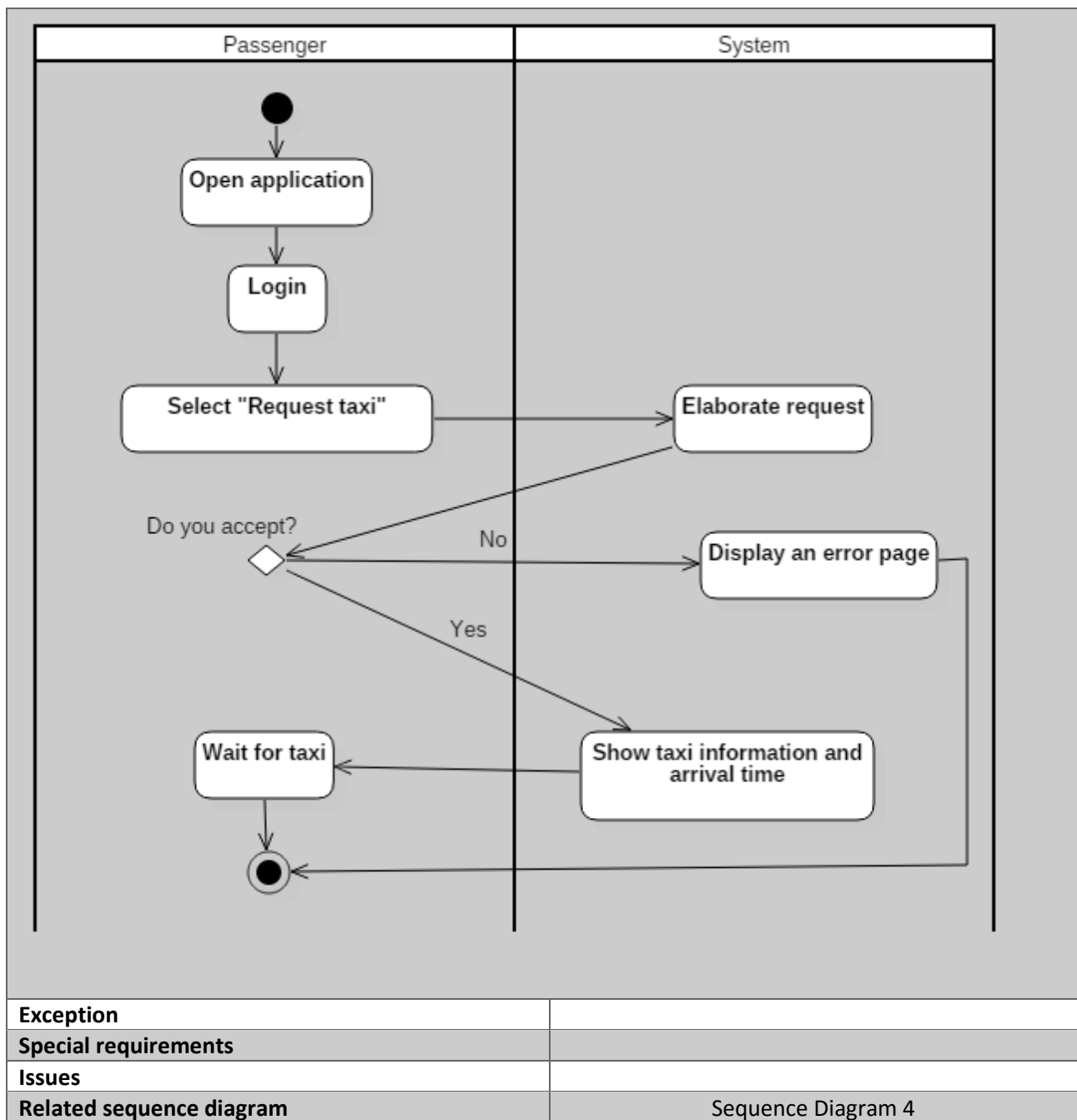
Use cases 3: Consult profile

Sequence Diagram 3



#### 4.2.3.4.4 Make a taxi request

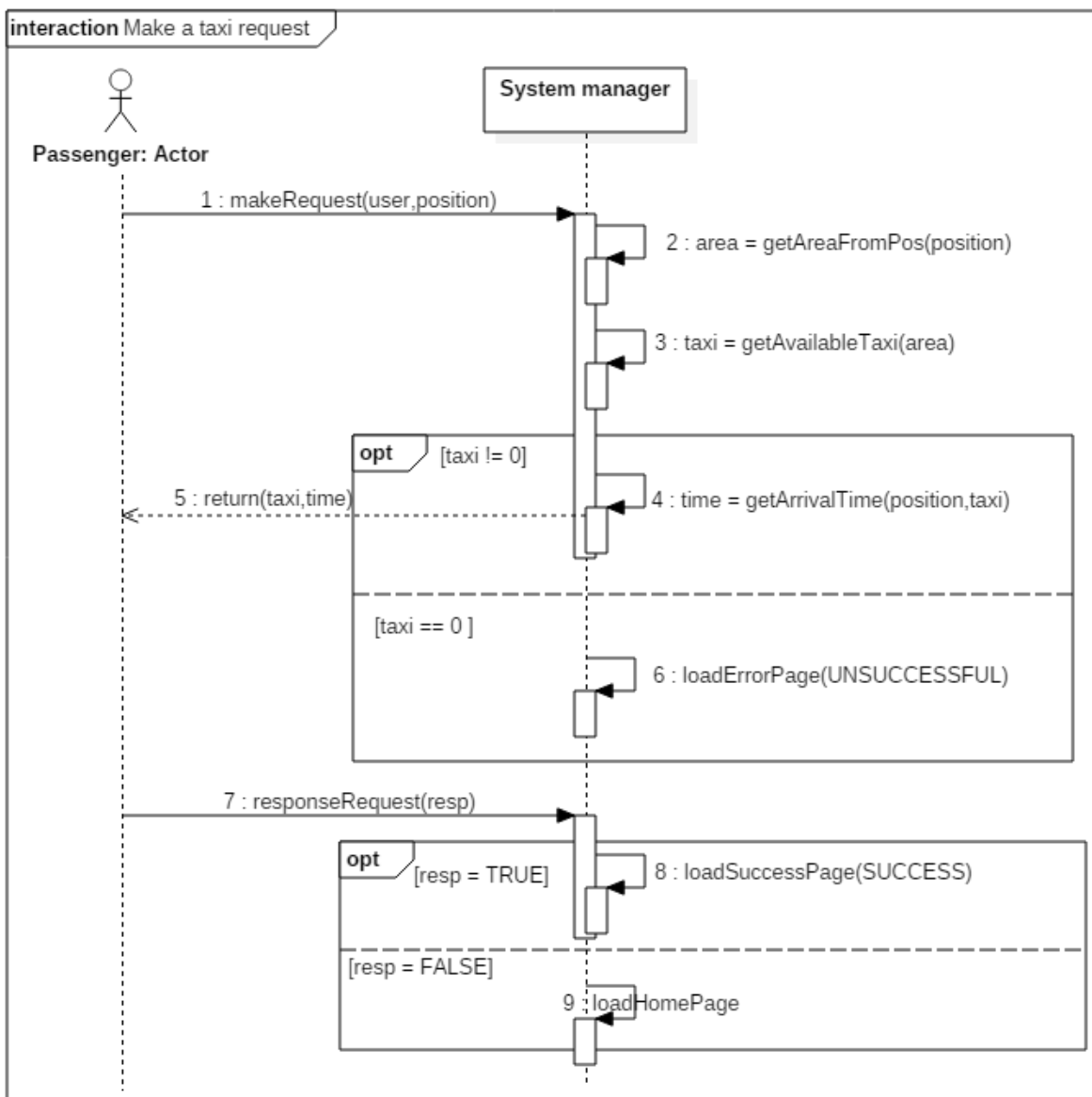
Make a taxi request	
<b>Code</b>	UC-PASS-004
<b>Description</b>	Every user can make a taxi request
<b>Goal</b>	[G4]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Passenger
<b>Entry condition</b>	The user exists in the system
<b>Exit condition</b>	The user has a taxi request
Flow of events	



Use cases 4: Make a taxi request

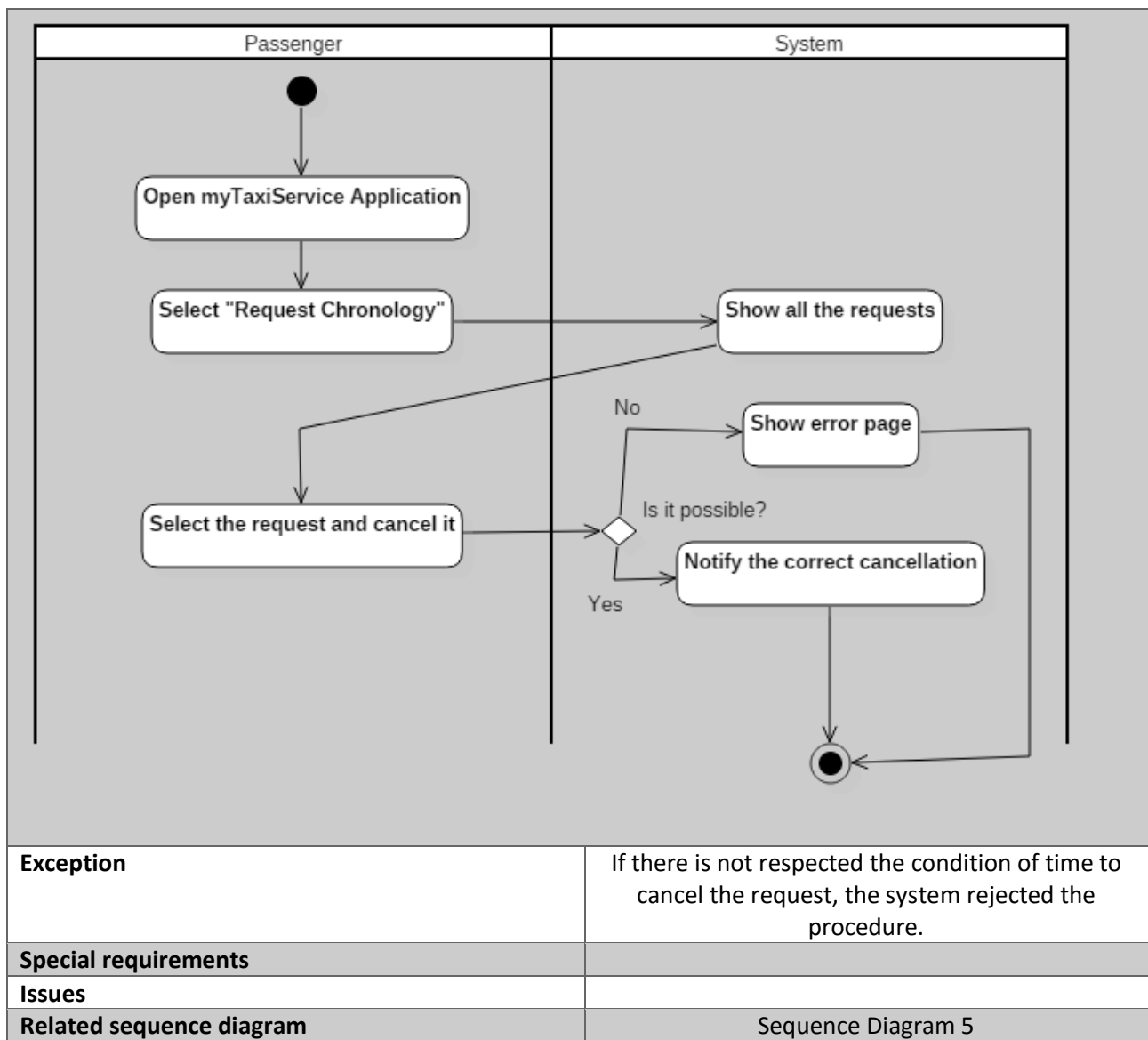


Sequence Diagram 4



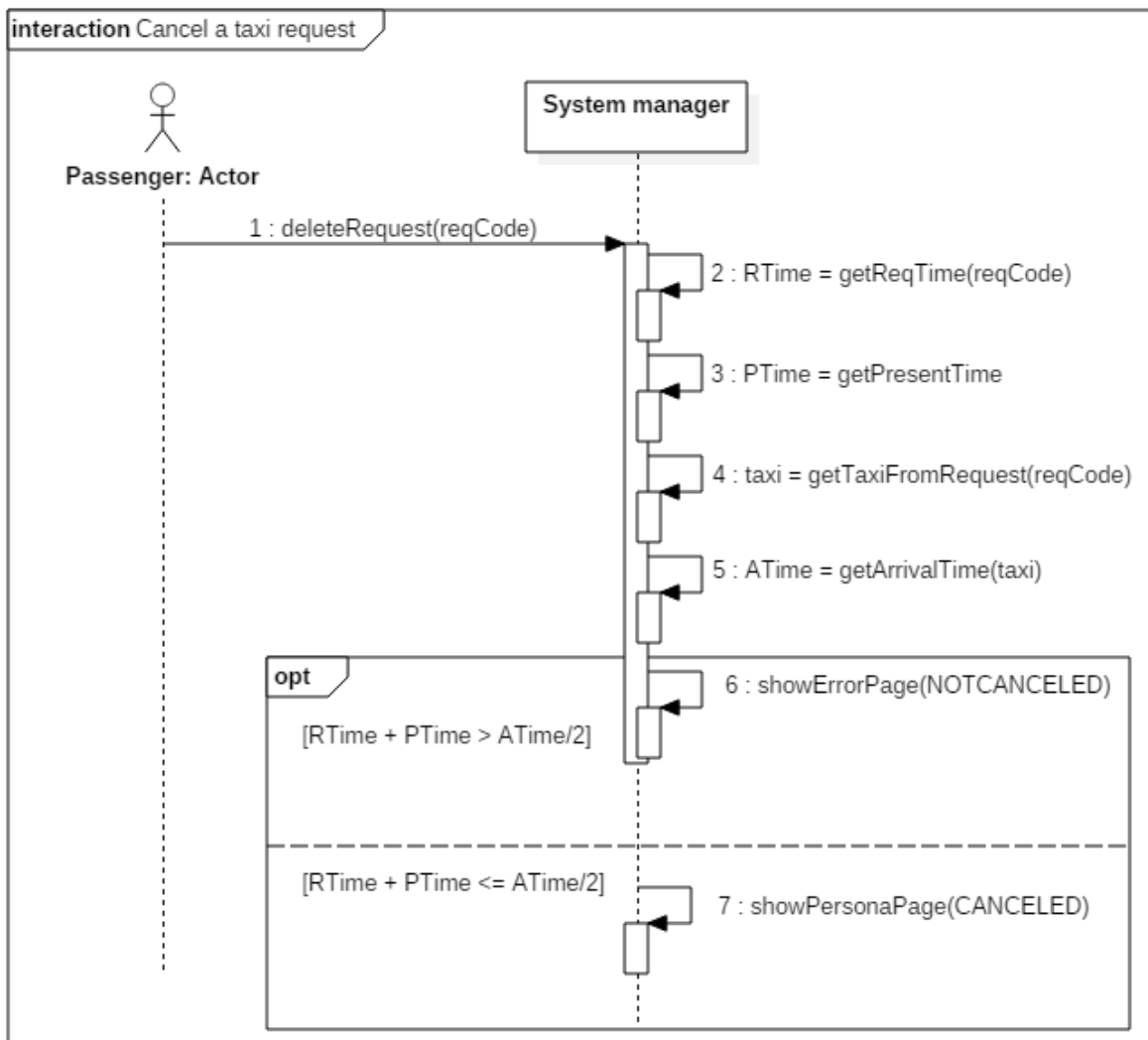
#### 4.2.3.4.5 Cancel a taxi request

Cancel a taxi request	
<b>Code</b>	UC-PASS-005
<b>Description</b>	Every user can cancel a taxi request
<b>Goal</b>	[G4]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Passenger
<b>Entry condition</b>	The user exists in the system
<b>Exit condition</b>	The user has canceled a taxi reservation
<b>Flow of events</b>	



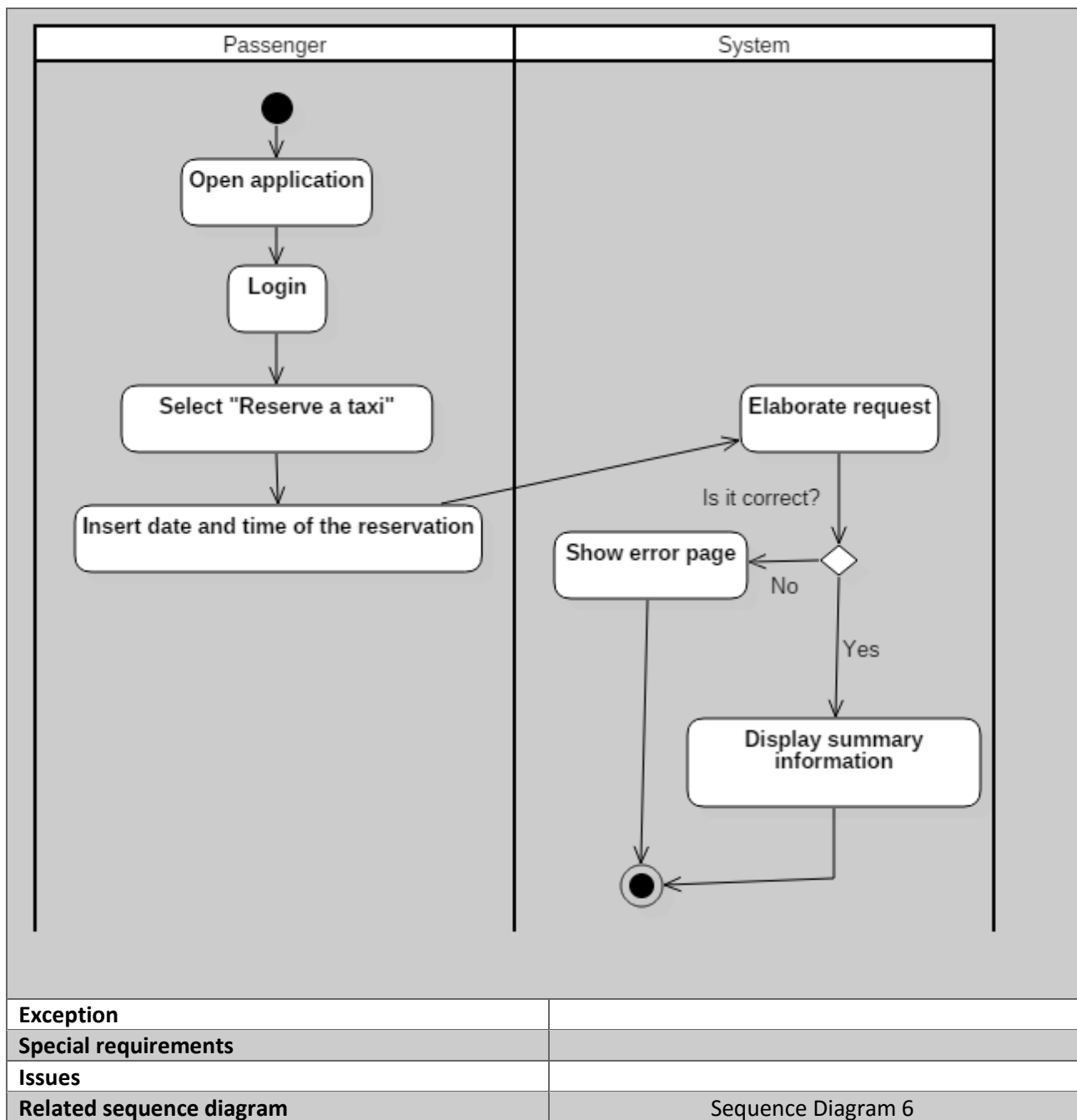
Use cases 5: Cancel a taxi request

Sequence diagram 5



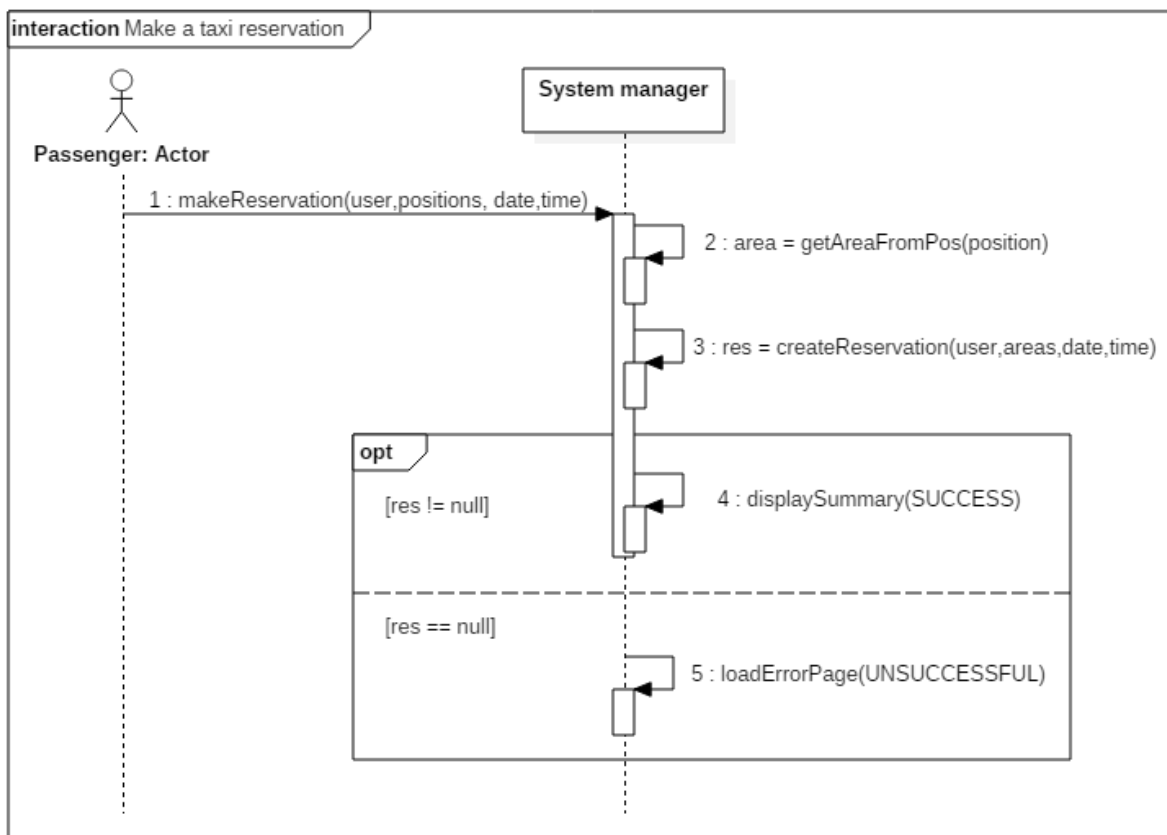
#### 4.2.3.4.6 Make a taxi reservation

Make a taxi reservation	
<b>Code</b>	UC-PASS-006
<b>Description</b>	Every user can make a taxi reservation
<b>Goal</b>	[G4]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Passenger
<b>Entry condition</b>	The user exists in the system
<b>Exit condition</b>	The user has a taxi reservation
<b>Flow of events</b>	



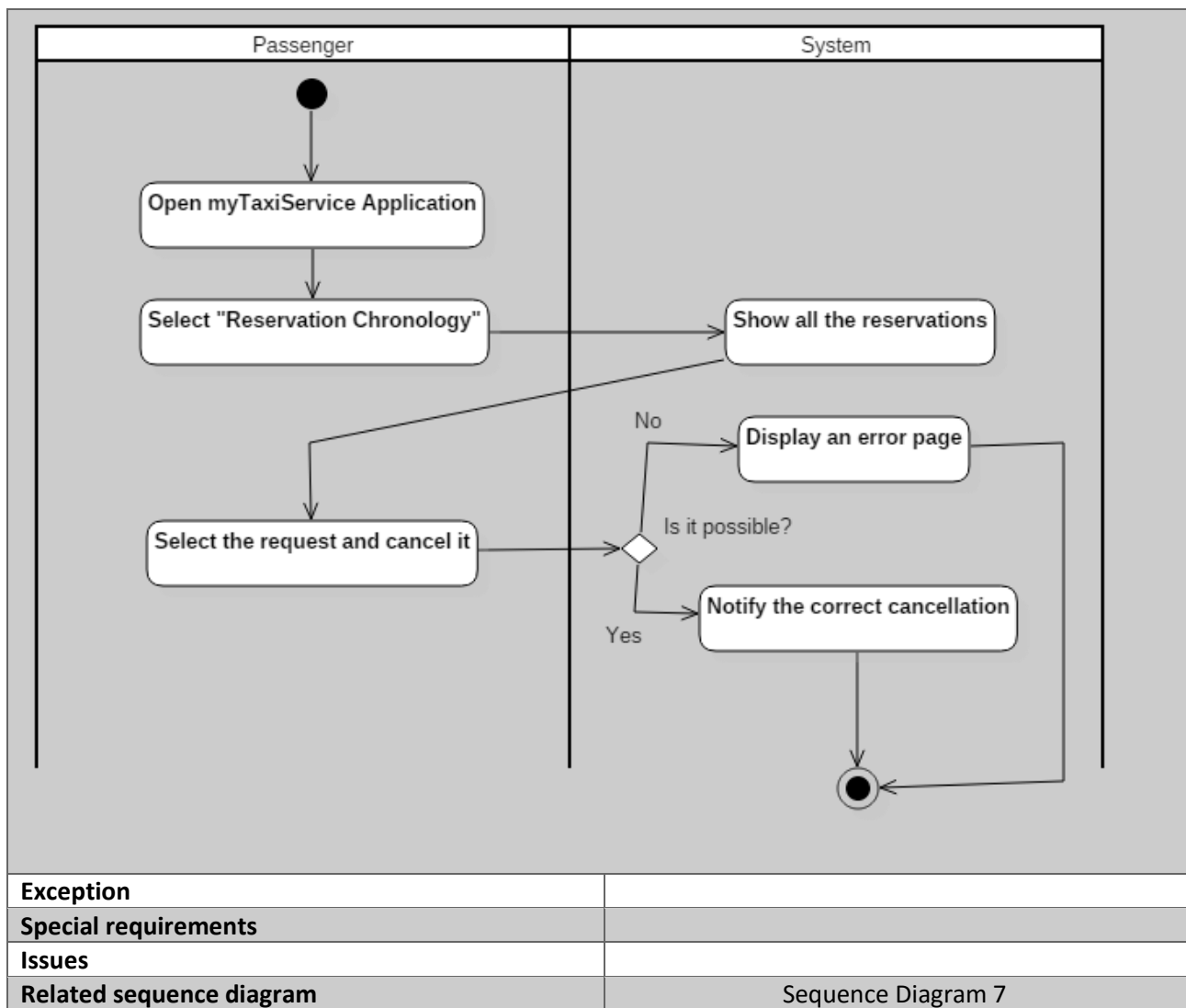
Use cases 6: Make a taxi reservation

## Sequence Diagram 6



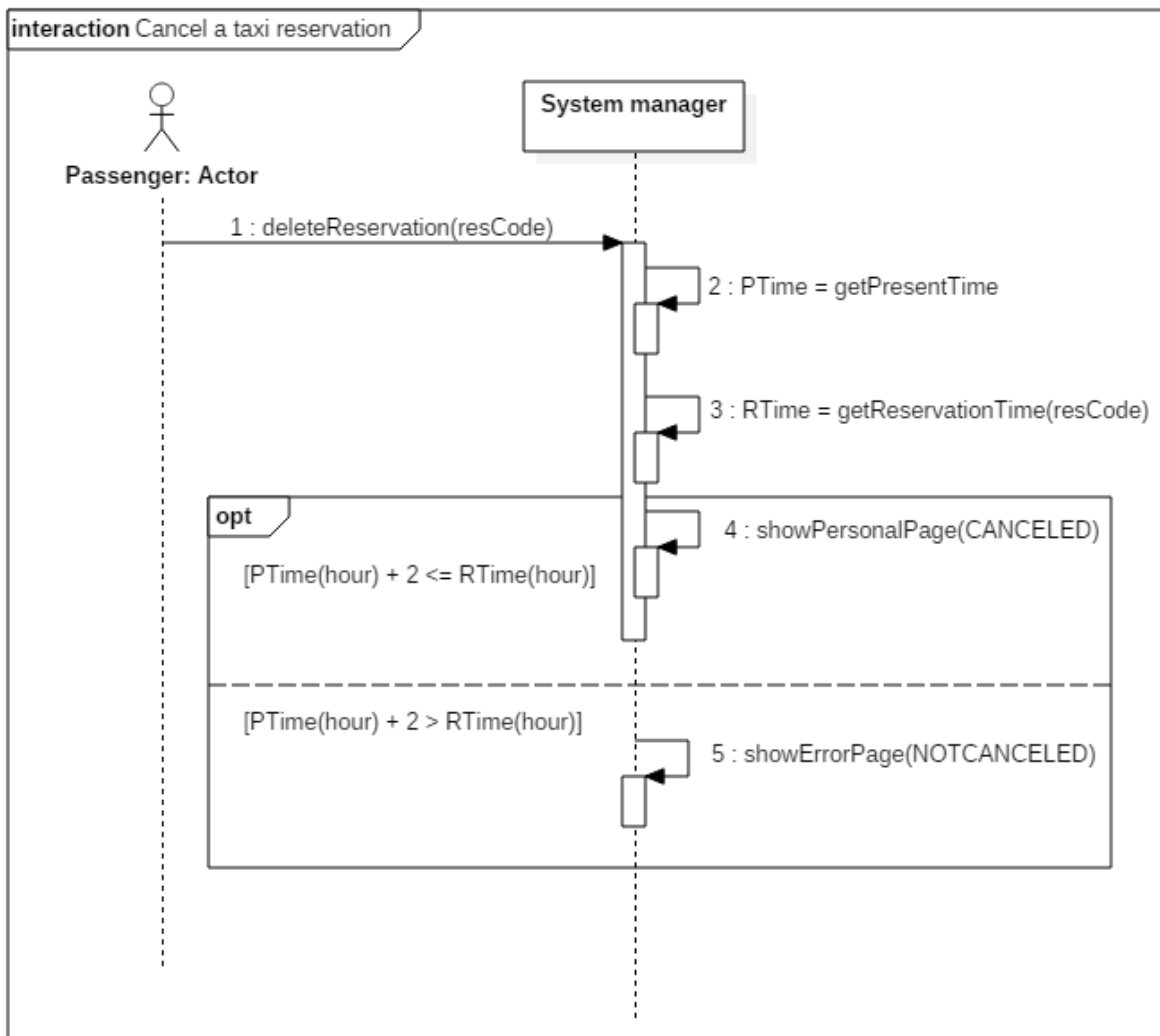
### 4.2.3.4.7 Cancel a taxi reservation

Cancel a taxi reservation	
<b>Code</b>	UC-PASS-007
<b>Description</b>	Every user can cancel a taxi reservation
<b>Goal</b>	[G4]
<b>Assumptions</b>	Internet connection available
<b>Actors</b>	Passenger
<b>Entry condition</b>	The user exists in the system
<b>Exit condition</b>	The user cancel a taxi reservation
<b>Flow of events</b>	



Use Cases 7: Cancel a taxi reservation

## Sequence Diagram 7



### 4.3 PERFORMANCE REQUIREMENTS

The system must meet high standards of performance with regard to the computation of the algorithms of taxi redistribution and to the bandwidth management, because of the many requests, which the system has to manage in real time.

As we mention before it is necessary to support a high level of concurrent users and concurrent taxi requests.

These following are example of important requirements, which the system should satisfy:

- a. The system shall be able to process 100 request/reservation transactions per minute in peak load.
- b. In standard workload, the CPU usage shall be less than 50%, leaving 50% for background jobs.
- c. Production of a request/ reservation shall take less than 10 seconds for 95% of the cases.
- d. The general navigation of the mobile application shall take at most 1 second.

### 4.4 LOGICAL DATABASE REQUIREMENTS

A relational database is a perfect solution for this project. It must save the entities identified in the analysis diagram (par. 4.2.2). It will manage users (taxi driver and customer), requests, reservations and city areas. This server will be subjected to high stress, as numerous request and reservations can be sent at the same time. In order to satisfy the performance requirements, it will be necessary to manage buffer memories to improve the read/write operations.

### 4.5 DESIGN CONSTRAINTS

The system must be designed and implements in JEE technology. The service is a work-in-progress application so that it needs to be implemented in the best way to make new changes in the future.

### 4.6 STANDARD COMPLIANCE

The system should be compliant with national standards of privacy and security, especially linked with traffic laws and the use of mobile devices. The service must satisfy all the UNI-ISO and European regulations.

### 4.7 SOFTWARE SYSTEM ATTRIBUTES

#### 4.7.1 Reliability

The system must assure the integrity and the durability of the taxi request and reservation, nevertheless it must assure the integrity of personal information.

Here some reliability measure the system should satisfy:

- a. The precision of car localization shall be at least 95%.
- b. The system defect rate shall be less than one failure per 1000 hours of operation.
- c. No more than one per 1000000 database transactions shall result in a failure requiring a system restart.
- d. The estimated loss of data in case of a disk crash shall be less than 0.01%.
- e. The system shall be able to handle up to 10000 concurrent users when satisfying all their requirements and up to 25000 concurrent users with browsing capabilities.

#### 4.7.2 Availability

- a. The system must be online 24/7 during all the year.



- b. The system myTaxiService shall have an availability of 999/1000 or 99%.
- c. The system shall not be unavailable more than 1 hour per 1000 hours of operation.
- d. Less than 20 seconds shall be needed to restart the system after a failure 95% of the time. (This is a MTTR (Mean-Time to Repair) requirement)

#### 4.7.3 Security

The software must protect both users' email and password as they travel in the city. These requirements can be achieved using high quality secure protocols and effective cryptography algorithm, such as AES or RSA.

Other security requirements are:

- a. Unauthorized access to the system and its data is not allowed.
- b. Ensure the integrity of the system from accidental or malicious damage.
- c. The system's data administrator may only change the access permissions for system data.
- d. All system data must be backed up every 24 hours and the backup copies stored in a secure location, which is not in the same building as the system.
- e. All external communications between the system's data server and clients must be encrypted.
- f. When an account is deleted, all its information, requests and reservations are removed.
- g. At least 99% of intrusions shall be detected within 10 seconds.
- h. The system must stop providing service to legitimate users while under denial of service attack (resistance to DoS attacks)

#### 4.7.4 Maintainability

The system needs maintainability in:

- a. Active/inactive users
- b. Servers files system backup and restoring
- c. Every program module must be assessed for maintainability. 70% must obtain "highly maintainable" and none "poor".
- d. The cyclomatic complexity of code must not exceed 7. No method in any object may exceed 200 lines of code.
- e. Installation of a new version shall leave all database contents and all personal settings unchanged.
- f. The product shall provide facilities for tracing any database field to places where it is used.
- g. The delivered system shall include unit tests that ensure 100% branch coverage.
- h. Development must use regression tests allowing for full retesting in 12 hours.

#### 4.7.5 Portability

The system can be installed in every mobile phones, which can be localized and can connect to Internet, but also in every computer, which supports a Java Virtual Machine and has an internet browser. The portability is the key element of the entire project. People can make a request from everywhere through the device they prefer.

### 4.8 OTHER REQUIREMENTS

The user interface should be usable for all the type of users, such that minimize the number of clicks and maximize the system functionality and interaction with the system users. Taxi driver have to use application.

Example of usability requirements, which the system should satisfy:

- a. Four out of five users shall be able to require or reserve a taxi within 5 minutes after a 2-hour introduction to the system.
- b. Novice users shall perform basic application functions in at most 3 minutes.
- c. Experienced users shall perform basic application functions in 1 minute.
- d. At least 80% of customers polled after a 3 months usage period shall rate their satisfaction (in their respective application store) with the system at 7 and more on a scale of 1 to 10.

## 5 APPENDIXES

---

### 5.1 ALLOY

Please refer to the file called 2015\_project\_Alloy\_myTaxiService.als

In according to the class diagram of the application, we have designed the first part of the Alloy file that includes only the signatures as shown below.

```
sig wString {}
sig posInt {value: Int} {value > 0}
sig boolean {value: Int} {value >= 0 && value <= 1}
```

```
abstract sig RegisteredUser
{
    username: one wString,
    email: one wString,
    password: one wString,
    firstname: one wString,
    lastname: one wString,
    phone: lone wString
}
```

```
sig TaxiDriver extends RegisteredUser
{
    driverCode: one wString,
    PersonalTaxi: one Taxi,
    drivingLicCode: one wString,
    drivingLicExp: one wString
}
```

```
sig Administrator extends RegisteredUser
{
}
```

```
sig Passenger extends RegisteredUser
{
    PersonalRequest: set Request,
    PersonalReservation: set Reservation
}
```

```

sig Taxi
{
    taxiCode: one wString,
    licensePlate: one wString,
    numPass: one posInt,
    taxiArrivalTime: lone posInt,
    available: one boolean,
    presentPosition: lone CityArea
}

sig Request
{
    requestCode: one wString,
    numPass: one posInt,
    requestTime: one wString,
    origin: one Position,
    destination: one Position,
    taxi: one Taxi,
    passUsername: one wString,
    active: one boolean
} {
    origin != destination
    taxi.numPass.value >= numPass.value
}

sig Reservation
{
    reservationCode: one wString,
    meetingTime: one wString,
    reservationDate: one wString,
    pickupDate: one wString,
    numPass: one posInt,
    origin: one Position,
    destination: one Position,
    taxi: lone Taxi,
    passUsername: one wString,
    active: one boolean
} {
    origin != destination
    taxi.numPass.value >= numPass.value
}

sig CityArea
{
    areaCode: one wString,
    fieldArea: one posInt,
    numTaxi: one Int,
    minNumTaxi: one Int,
    taxiArea: set Taxi,
    positionInArea: set Position
}
{
    numTaxi = #taxiArea
    numTaxi >= 0
}

```

```

    minNumTaxi >= 0
    fieldArea.value <= 3 && fieldArea.value >= 1
}

sig Position
{
    address: one wString,
    latitude: one posInt,
    longitude: one posInt
}

```

#### 5.1.1 Constraints

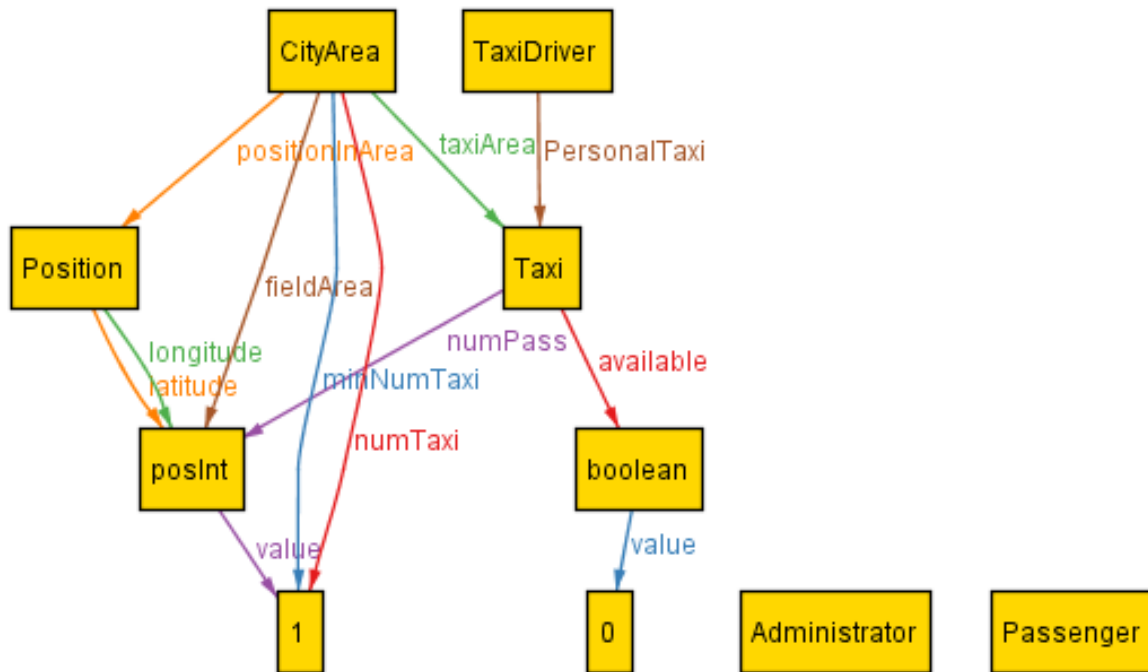
In order to obtain a right model of Alloy, we have inserted the following constraints:

- I. Identification code of each signature must be unique.
- II. The system has one and only one administrator.
- III. Origin and destination of a request/reservation must be different.
- IV. If a taxi exists then it must be owned only by a taxi driver. Therefore, there are not two taxi drivers with the same taxi.
- V. All positions in an area must be different.
- VI. All taxis of an area must be different.
- VII. All reservations of a passenger must be different.
- VIII. All request of a passenger must be different.
- IX. Two positions with the same address are not allowed.
- X. Two users with the same e-mail or username are not allowed.
- XI. A city area must have at least one position.
- XII. There must not be two city areas with positions in common.
- XIII. If a taxi is available then it must be in a position.
- XIV. All passengers must have at most one request activated.
- XV. Each request must have a user who have generated it.
- XVI. Each reservation must have a user who have generated it.
- XVII. Each active request must have an available taxi.
- XVIII. Each active reservation must have an available taxi.
- XIX. It must not exists two active request with the same taxi.
- XX. It must not exists two active reservation with the same taxi.

### 5.1.2 Worlds generated

We have inserted the constraints and the signatures shown above and generated many worlds, for which we are showing only few of them.

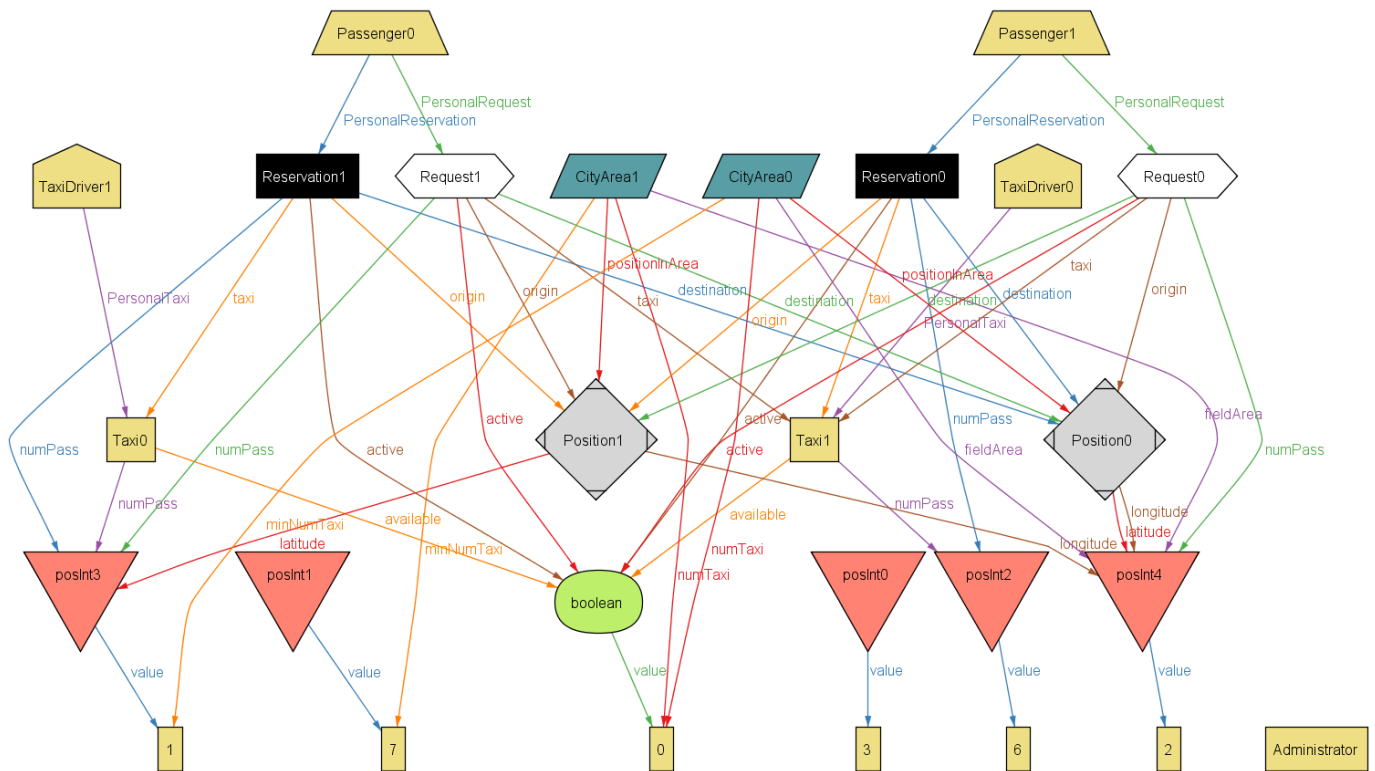
#### 5.1.2.1 World 1



The first world we have generated is composed by:

- One taxi driver with his personal taxi that is not available and has a number of passengers equal to one.
- One administrator.
- One passenger who does not have any request or reservation.
- One city area with only one position.

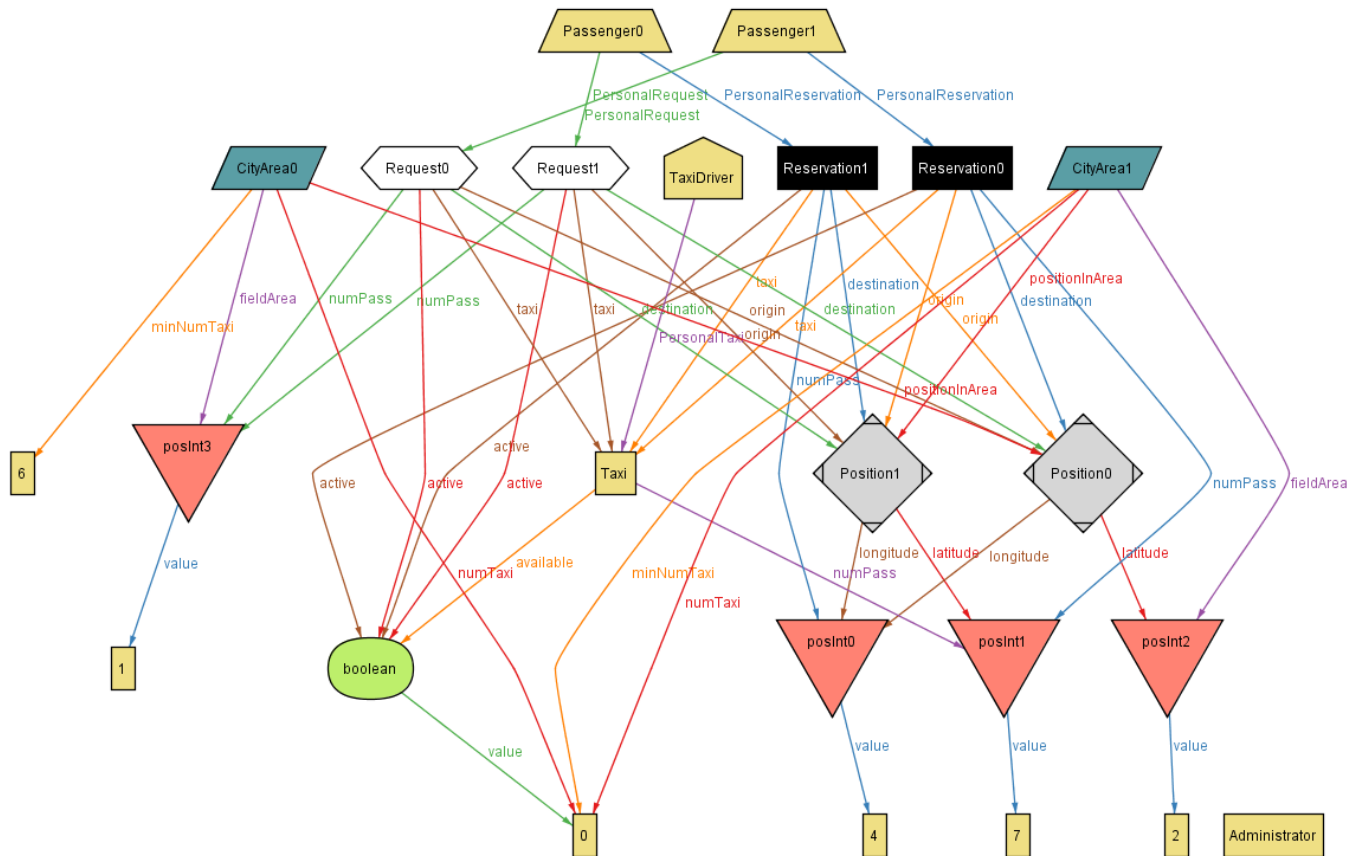
### 5.1.2.2 World 2



The second world is composed by:

- One and only one administrator.
- Two passengers: the first passenger has one reservation and one request served by TaxiDriver1 and TaxiDriver2; the second passenger has one reservation and one request served by the same taxi driver (TaxiDriver0).
- Two taxi drivers: TaxiDriver1 has as personal taxi Taxi0 and TaxiDriver0 has as personal taxi Taxi1.
- Two city areas: CityArea0 and CityArea1 that have only one position respectively Position1 and Position0.

### 5.1.2.3 World 3



The third world is composed by:

- Two passengers: the first passenger, labeled as Passenger0, who has one request and one reservation served by the same taxi driver; the second passenger, labeled as Passenger1, who has one request and one reservation served by the same taxi driver.
- Only one taxi driver with his personal taxi.
- One and only one administrator.
- Two city areas: CityArea0 and CityArea1 that have only one position respectively Position and Position1.