

Politecnico di Milano

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL’INFORMAZIONE

Computer Science and Engineering

Software Engineering 2 project

**myTaxiService**

**Requirements Analysis and Specification Document**

|  |  |  |
| --- | --- | --- |
| Prof. Elisabetta Di Nitto | Andrea Autelitano | Matr. 849869 |
| Marco De Cobelli | Matr. 858360 |
| Matthew Rossi | Matr. 858880 |

Academic year 2015-2016

Table of Contents

[1. Introduction 4](#_Toc434057310)

[1.1 Purpose 4](#_Toc434057311)

[1.2 Scope 4](#_Toc434057312)

[1.3 Definitions, acronyms, and abbreviations 4](#_Toc434057313)

[1.4 References 4](#_Toc434057314)

[1.5 Overview 4](#_Toc434057315)

[2. Overall description 6](#_Toc434057316)

[2.1 Product perspective 6](#_Toc434057317)

[2.1.1 System interfaces 6](#_Toc434057318)

[2.1.2 User interfaces 6](#_Toc434057319)

[2.1.3 Hardware interfaces 6](#_Toc434057320)

[2.1.4 Software interfaces 6](#_Toc434057321)

[2.1.5 Communications interfaces 6](#_Toc434057322)

[2.1.6 Memory constraints 6](#_Toc434057323)

[2.1.7 Operations 6](#_Toc434057324)

[2.2 Product functions 7](#_Toc434057325)

[2.3 User characteristics 9](#_Toc434057326)

[2.4 Constraints 9](#_Toc434057327)

[2.5 Assumptions and dependencies 10](#_Toc434057328)

[3. Specific requirements 11](#_Toc434057329)

[3.1 External interfaces 11](#_Toc434057330)

[3.2 Functions 15](#_Toc434057331)

[3.3 Scenarios 18](#_Toc434057332)

[Ride in process 18](#_Toc434057333)

[Ride in process 18](#_Toc434057334)

[Ride cancelation 18](#_Toc434057335)

[Ride modification 19](#_Toc434057336)

[Concluded/Pending ride details 19](#_Toc434057337)

[3.4 Use cases 20](#_Toc434057338)

[3.4.1 Sign up 20](#_Toc434057339)

[3.4.2 Login 21](#_Toc434057340)

[3.4.3 Password recovery 22](#_Toc434057341)

[3.4.4 Requesting a ride 23](#_Toc434057342)

[3.4.5 Reserving a ride 24](#_Toc434057343)

[3.4.6 Shared ride 25](#_Toc434057344)

[3.4.7 Payment via Credit Card 26](#_Toc434057345)

[3.4.8 Ride in process 27](#_Toc434057346)

[3.4.9 Ride cancelation 28](#_Toc434057347)

[3.4.10 Ride modification 29](#_Toc434057348)

[3.4.11 Concluded/Pending ride details 30](#_Toc434057349)

[3.5 Non Functional Requirements 31](#_Toc434057350)

[3.5.1 Functional stability 31](#_Toc434057351)

[3.5.2 Performance efficiency 31](#_Toc434057352)

[3.5.3 Compatibility 31](#_Toc434057353)

[3.5.4 Usability 32](#_Toc434057354)

[3.5.5 Reliability 32](#_Toc434057355)

[3.5.6 Security 32](#_Toc434057356)

[3.5.7 Maintainability 33](#_Toc434057357)

[3.5.8 Portability 33](#_Toc434057358)

[3.6 Alloy models 33](#_Toc434057359)

[4. Appendixes 34](#_Toc434057360)

[4.1 Used tools 34](#_Toc434057361)

[4.2 Attached documents 34](#_Toc434057362)

[4.2.1 Client’s document 34](#_Toc434057363)

[4.3 Hours of work 34](#_Toc434057364)

# 1. Introduction

## 1.1 Purpose

The main goal of this document is to completely describe the software “myTaxiService” we are going to develop, laying out its functional and non-functional requirements aspects. It’s included the description of the scenarios and use cases and the interaction between the users and the software. Moreover, it establishes the basis for the contract between customers and suppliers.

The audience of this software are taxi drivers and all the common people that will use the service as passengers.

## 1.2 Scope

The aim of this project is to optimize the taxi service of a large city by means of the simplification of the access to the service for the passengers and the fair management of the taxi queues.

The registered users will be able to book a taxi from a mobile application or a web site and taxi drivers will use the mobile application to accept the reservation of users.

The city is divided into zones of approximately 2km2, each of these with a different queue of taxi.

The taxi chosen from the system will be the nearest to the user that has made the reservation and it’s always chosen starting from the user zone.

Furthermore, when a taxi becomes available, either because its taxi driver logs into the system or the related taxi driver reports the finish of a ride with a passenger, its identifier will be added at the end of the queue of the current zone in which he and his taxi are located.

When a request is receipt by the first taxi driver in a queue, he/she can freely choose to accept it or to refuse it. In the first case, it’s notified to the client that one taxi driver has accepted his/her request, in the second case the request is sent to the second taxi in the queue and so on.

Moreover, besides the traditional interfaces, it’s provided programmatic interfaces to enable the development of additional services like taxi sharing.

Other functionalities are the possibility to reserve a taxi from a particular origin to a specific destination within two hours before the ride and, if the request is accepted, the system will allocate a taxi 10 minutes before the ride to the user.

Another one is the taxi sharing option that allows users to share the cost of a ride with other people after specifying besides the starting point, the destination of his/her ride, the number of people to carry on and the availability to share his/her ride.

In any case, the system automatically calculates the route of the taxi toward the destinations and it provides the fee for each person and inform him/her and the taxi driver before the departure.

## 1.3 Definitions, acronyms, and abbreviations

* CLIENT: a person that uses the app to request rides
* TAXI DRIVER: a person that uses their taxi to provide the ride.
* USER: either a client or a taxi driver

## 1.4 References

* Description of the problem: Assignments 1 and 2 (RASD and DD).pdf.
* IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications
* ISO/IEC 2501:2001 Product Quality Model

## 1.5 Overview

This contents and organization of this document is:

* Section 1: Introduction.

It gives a description of the document and some basic information about the software.

* Section 2: Overall description of the document.

It gives general information about the software with more focus about constraints and assumptions.

* Section 3: Specific requirements.

It lists functional and non-functional requirements, scenarios and use cases with UML diagrams.

* Section 4: Appendix.

It contains some information about the attached .als file and some described screenshots of software used to generate it.

# 2. Overall description

## 2.1 Product perspective

The software we will release is composed by a web site and a mobile application that are used from the users that desire to reserve a taxi and from the taxi drivers to accept or refuse these requests according to their availabilities.

The mobile application and the web site aren’t independent and are used exchanging data with another system that contains all the data related to taxi, taxi drivers, passengers and rides concluded and pending.

### 2.1.1 System interfaces

There are interfaces between web sites, mobile applications and the system in which are stored all the data related to the registered taxi drivers, the registered users and the rides.

All these data are queried during the normal execution of the pieces of software we develop.

### 2.1.2 User interfaces

There are mainly two kinds of user interfaces.

1. **Taxi driver.** There are 4 main interfaces for the taxi driver that allow him/her to:

* Login into the application/web site
* See requests coming from the users and accept/refuse them
* See the history of his/her concluded rides
* See the state of the pending ride with the possibility to handle it

1. **User.** There are 4 main interfaces for the user that allow him/her to:

* Login into the application/web site
* Request/Book a taxi by choosing the number of people to carry, where to start and conclude the ride and the departure time. Moreover, the user can choose to give the possibility to share a ride with other users if possible according to common destinations among them.
* See the history of his/her concluded rides
* See and handle the pending rides

### 2.1.3 Hardware interfaces

There are no hardware interfaces.

### 2.1.4 Software interfaces

There are APIs to allow the access to the GPS and to the mobile application.

### 2.1.5 Communications interfaces

There are interfaces that allow the communication of the data between the database of the system and the mobile applications/web site.

Furthermore, there are programmatic interfaces to allow the development of additional services.

### 2.1.6 Memory constraints

There is constraint due to the availability of memory on the mobile phone of the user and of the taxi driver for install the app, for users only if used (not mandatory).

### 2.1.7 Operations

It’s implemented a function of backup of the data in the central database in order to guarantee the persistency of the data.

## 2.2 Product functions

|  |  |
| --- | --- |
| Name | **[PF00] Passenger registration** |
| Description | The system allows the passenger to sign-up inserting name, surname, gender, date of birth e-mail and password |

|  |  |
| --- | --- |
| Name | **[PF01] Taxi driver registration** |
| Description | The system allows the passenger to sign-up inserting name, surname, gender, date of birth e-mail, password, driving license, taxi license and IBAN |

|  |  |
| --- | --- |
| Name | **[PF02] Login** |
| Description | The system allows the user to login inserting e-mail and password |

|  |  |
| --- | --- |
| Name | **[PF03] Password recovery** |
| Description | The system allows the user to recover his/her password account, by first identifying the account through the e-mail address and then sending a new password randomly generated that will allow the user to access his account to change the password. |

|  |  |
| --- | --- |
| Name | **[PF04] Personal information management** |
| Description | The system allows the users to modify the data previously inserted at the registration phase. The passenger is able to modify only his password, the taxi driver can also modify driving licence, taxi licence and IBAN. |

|  |  |
| --- | --- |
| Name | **[PF05] Add payment option** |
| Description | The system allows the passenger to memorize into it one or more credit cards so that once he/she wants to pay a race with a credit card instead of inserting it each time he/she has just to select it from an existing list. |

|  |  |
| --- | --- |
| Name | **[PF06] Delete payment option** |
| Description | The system allows the passenger to delete a selected credit card. |

|  |  |
| --- | --- |
| Name | **[PF07] Add new taxi** |
| Description | The Taxi driver is able to insert information about the taxis he can use. |

|  |  |
| --- | --- |
| Name | **[PF08] Taxi driver working state and taxi choosing** |
| Description | The Taxi driver is able to specify once he has logged in if he wants to start working using the system support selecting the taxi he is going to use from an existing list and go back to not working state once he wants to stop working. |

|  |  |
| --- | --- |
| Name | **[PF09] Request taxi** |
| Description | The passenger can request a taxi by specifying the place where he wants to get picked up through the address or his/her GPS coordinates, the number of people who will take the ride, the destination and if he wants to share or not the run. |

|  |  |
| --- | --- |
| Name | **[PF10] Reservation of a taxi** |
| Description | The passenger can reserve a taxi, to do that he specifies, when and where he wants to have the ride specifying date, time and address, the number of people that will take the ride, the destination and if he wants to share or not the run |

|  |  |
| --- | --- |
| Name | **[PF11] Zone identification** |
| Description | The system once receives a user position has to identify the zone in which he is located |

|  |  |
| --- | --- |
| Name | **[PF12] Queue handling** |
| Description | The system has to provide a fair queue management assigning at each zone a FIFO (First In First Out) queue of the available taxis |

|  |  |
| --- | --- |
| Name | **[PF13] Taxi driver notification** |
| Description | The system notifies the taxi driver selected in the queue and sends to him the position they are, the destination they want to get to and the number of people that want the ride. If the run is shared with other passengers the system will send to the taxi driver information about each stop like: the position, number of people that will get on the taxi and their destination. The taxi driver can accept or refuse the ride. |

|  |  |
| --- | --- |
| Name | **[PF14] Taxi driver answer notification** |
| Description | The taxi driver can accept or refuse the ride the system choose for him. |

|  |  |
| --- | --- |
| Name | **[PF15] Compute waiting time** |
| Description | The system sends a request to Google Maps servers specifying taxi driver position (starting point) and passenger position (destination point) and analyse the response to extract the waiting time of the fastest path. |

|  |  |
| --- | --- |
| Name | **[PF16] Compute race cost** |
| Description | The system sends a request to Google Maps servers specifying the ride origin (starting point) and the ride destination (destination point) and analyse the response to extract the length of the fastest path. Based on that the system computes preventively the ride cost. |

|  |  |
| --- | --- |
| Name | **[PF17] Passenger notification** |
| Description | The system confirm to the passenger that a taxi has been allocated for his call and send him its identification code, the waiting time and the ride cost. |

|  |  |
| --- | --- |
| Name | **[PF18] Pay ride** |
| Description | The passenger selects a payment option and pays the taxi driver the amount owed. |

|  |  |
| --- | --- |
| Name | **[PF19] Conclude ride** |
| Description | The taxi driver communicate the system the successful conclusion of the ride and the system appends the taxi driver to the queue zone in which he/she is. |

|  |  |
| --- | --- |
| Name | **[PF20] List of rides** |
| Description | The system sends the user the list of rides:   * the list of pending rides with details (taxi code, time of reservation, ETA, date, origin, destination, price, n° people and acceptance state) * the list of rides already done with details (taxi code, time of reservation, Time of arrival, date, origin, destination, price, n° people) and overall statistics |

|  |  |
| --- | --- |
| Name | **[PF21] Modify reservation** |
| Description | The passenger can change the reservation information (date, time, destination and shared). |

|  |  |
| --- | --- |
| Name | **[PF22] Cancel reservation** |
| Description | The passenger can erase the reservation. |

|  |  |
| --- | --- |
| Name | **[PF23] Race give up** |
| Description | The taxi driver has accepted a ride, but he is unable to satisfy the call of the passenger and notifies to the system. |

|  |  |
| --- | --- |
| Name | **[PF24] Race interruption** |
| Description | The taxi driver can interrupt the race once he has already taken the passenger because:   * The taxi or the taxi driver has problems in carrying out the race * The passenger wants to stop the race before reaching the destination (e.g. he is near the destination, there’s huge traffic jam and he can get at the destination faster on foot) |

|  |  |
| --- | --- |
| Name | **[PF25] Administration panel** |
| Description | The administrator can access a dedicated interface whereby he can see overall statistics about the system, the available taxi distribution, all the race that has been done and a list of errors that occurred in the system usage. |

|  |  |
| --- | --- |
| Name | **[PF26] Tutorial** |
| Description | To help the user understand how the web and mobile app work, at the first login a tutorial is shown that summarize system functionalities and how to access them. |

## 2.3 User characteristics

We can identify two main categories of users: taxi drivers and clients.

However, both categories are characterized by heterogeneous levels of education, experience and technical expertise, thus both the web application and the mobile app should satisfy the “Usability” requirement, as specified in section 3.6.4.

## 2.4 Constraints

The service is subjected to the following constraints:

1. **Regulatory policies**

HTTP cookies are used both on the web and the mobile application: therefore, the user must be informed (e.g. using a popup message), in compliance with EU Privacy directives.

Taxi fares calculated by the system must not exceed the limit determined by the National Fares Association.

1. **Interfaces to other applications**

The system must be able to interact with the pre-existing central database containing informations about taxi drivers and IDs.

The system must use the Google Maps APIs to calculate the taxis distribution in each area.

1. **Parallel operations**

The system must support concurrent access by different users.

1. **Safety and security considerations**

Sensitive data from both users and taxi drivers are stored in the central database. Therefore, these data must not be accessible from external subjects (e.g. advertising agencies).

## 2.5 Assumptions and dependencies

* When there’s a request or a notification of a reservation of a taxi and in the zone of the user no taxi are available, it’s chosen a taxi from the nearest zone in which one taxi is available, and so on.
* If a taxi has some problems (like problem to the engine or something similar) during the the way to a passenger or during a ride there are two different cases:
  + *The passenger is currently transported.* The taxi driver has to report the problem into his/her app and only if the passenger agrees, he has to indicate that the passenger wants another taxi right there. Then the system will send immediately the request to another taxi in the queue of that zone, by using the GPS of the taxi driver.
  + *The taxi driver is driving towards a passenger.* The taxi driver has to report the problem into his//her app. Then the system will forward the request to another taxi driver and a notification to the user of the problem and of the probable delay but the user hasn’t to do anything.
* Only in the case of a reservation of a taxi, the client can delete/modify it only before 10 minutes before the departure time. In the case of a simple request, he can’t delete/modify anything.
* There’s a trigger on the mobile app of taxi drivers that send to the system a notification in the case he changes his current zone so that the system can insert him/her at the end of the queue of the new zone.
* The fee is calculated by the system by means of the kilometers of the ride when the taxi driver accepts the request of transportation so that it can be sent to him and also to the passenger. The taxi drivers take the money in cash or can accept the payment with a credit card, whose data must be added into the passenger account for this option.
* When a taxi driver ends a ride, he has to report it into the app. In fact, only after that the system can insert him/her at the end of the queue of his/her current zone. Until he/she doesn’t confirm the end of the ride, he/she stays out from any queue.
* If a taxi driver doesn’t answer to a notification within 30 seconds, he/she is shifted at the end of the queue. This is done to guarantee that a taxi driver is working well and doesn’t weigh on the whole system increasing delays and hence the satisfaction of clients.
* There is a pre-existent database that contains information about taxi drivers’ personal details and licenses. To be able to register to myTaxiService as a taxi driver, they must be already present in the database.
* If a taxi driver wants to use the application as a client, they must create another account.
* The client cannot change the destination during the ride. However, it is possible to suspend the ride if an extraordinary event occurs (e.g. a taxi technical problem, traffic jam…): in this case, the taxi driver must send their position to the system so that it can recalculate the fare.

# 3. Specific requirements

## 3.1 External interfaces

|  |  |  |
| --- | --- | --- |
| Registration page The following images show the registration pages available for a visitor from the web site or from the mobile application. | | |
|  | |  |
|  | | |
| Login page The following images show the login pages available for a visitor from the web site or from the mobile application. | | |
|  | |  |
|  | | |
| Ride request The following images show the pages that allow a client to request a ride right now or to book a new one. | | |
|  | |  |
|  | | |
| User view – Cocluded rides The following images show the pages that allow a user to see the list of his/her concluded rides with all the related information and the possibility to see a graph with the number of his/her concluded rides grouped by a certain period of time. | | |
|  | |  |
|  | | |
| User view – Pending rides The following images show the pages that allow a client to see the list of his/her pending rides with all the related information and the possibility to delete, modify and pay with credit card.  In the image of the web site below you can see the buttons Delete and Modify only related to the ride booked because the others can’t be modified/canceled. Moreover, the button Pay is disable for the ride booked because is not already accepted from a taxi driver. | | |
|  | |  |
|  | | |
| Taxi request | Accepted request | |
|  |  | |
|  |  | |
| Administration panel The following image shows the page available only for the adminitrators of the system with data grouped for kind of information. | | |
|  | | |
|  | | |

## 3.2 Functions

|  |  |
| --- | --- |
| Name | **[PF00] Passenger registration** |
| Requirements | [R0] The visitor must not be already registered  [R1] The visitor must insert valid data according to the format of each field  [R2] The visitor must choose an email that exists and is not already used by another user  [R3] The user must be able to access his email |

|  |  |
| --- | --- |
| Name | **[PF01] Taxi driver registration** |
| Requirements | [R0] The visitor must not be already registered  [R1] The visitor must insert valid data according to the format of each field  [R2] The visitor must choose an email, a driving license, a taxi license and an IBAN that exists and is not already used by another user  [R3] The user must be able to access his email |

|  |  |
| --- | --- |
| Name | **[PF02] Login** |
| Requirements | [R0] The user must be already signed up  [R1] The user must insert valid data according to the format of each field  [R2] The user must know his/her email and password to successfully accomplish the login |

|  |  |
| --- | --- |
| Name | **[PF03] Password recovery** |
| Requirements | [R0] The user must be already signed up  [R1] The user must insert valid email according to its format  [R2] The user must know his/her email to successfully retrieve the password  [R3] The user must be able to access his email |

|  |  |
| --- | --- |
| Name | **[PF04] Personal information management** |
| Requirements | [R0] The user must be already logged in  [R1] The user must insert valid data according to the format of each field  [R2] The user must know his/her password to successfully change the password |

|  |  |
| --- | --- |
| Name | **[PF05] Add payment option** |
| Requirements | [R0] The passenger must be already logged in  [R1] The passenger must insert valid data according to the format of each field  [R2] The passenger must remember his/her credit card information |

|  |  |
| --- | --- |
| Name | **[PF06] Delete payment option** |
| Requirements | [R0] The passenger must be already logged in |

|  |  |
| --- | --- |
| Name | **[PF07] Add new taxi** |
| Requirements | [R0] The taxi driver must be already logged in  [R1] The taxi driver must insert valid data according to the format of each field  [R2] The taxi driver must remember his/her taxi information. |

|  |  |
| --- | --- |
| Name | **[PF08] Taxi driver working state and taxi choosing** |
| Requirements | [R0] The taxi driver must be already logged in  [R1] The taxi driver must know if he/she wants to work  [R2] The taxi driver choose the taxi he/she is going to use  [R3] The taxi driver GPS has to work |

|  |  |
| --- | --- |
| Name | **[PF09] Request taxi** |
| Requirements | [R0] The passenger must be already logged in  [R1] The passenger must insert valid data according to the format of each field  [R2] The number of people inserted by the passenger has to be smaller or equal than 4 |

|  |  |
| --- | --- |
| Name | **[PF10] Reservation of a taxi** |
| Requirements | [R0] The passenger must be already logged in  [R1] The passenger must insert valid data according to the format of each field  [R2] The number of people inserted by the passenger has to be smaller or equal than 4 |

|  |  |
| --- | --- |
| Name | **[PF11] Zone identification** |
| Requirements | [R0] The position to analyze must be inside a defined zone |

|  |  |
| --- | --- |
| Name | **[PF12] Queue handling** |
| Requirements | [R0] At least a taxi driver has to be available in the whole system if the function is used to assign the taxi to a ride |

|  |  |
| --- | --- |
| Name | **[PF13] Compute race cost** |
| Requirements | [R0] The taxi driver has accepted a ride  [R1] Google Maps servers are reachable |
| Name | **[PF14] Taxi driver notification** |
| Requirements | [R0] The taxi driver must be already logged in and working  [R1] A ride has been requested (or reserved and it has to be allocated to a taxi) by a passenger  [R2] At least a taxi driver has to be available in the whole system  [R3] The race cost has been calculated  [R4] The taxi driver has access at a data connection |

|  |  |
| --- | --- |
| Name | **[PF15] Taxi driver answer notification** |
| Requirements | [R0] The taxi driver must be already logged in and working  [R1] The taxi driver has been notified  [R2] The taxi driver is connected at a data network |

|  |  |
| --- | --- |
| Name | **[PF16] Compute waiting time** |
| Requirements | [R0] The taxi driver has accepted a ride  [R1] The taxi driver position is known  [R2] Google Maps servers are reachable |

|  |  |
| --- | --- |
| Name | **[PF17] Passenger notification** |
| Requirements | [R0] The passenger must be already logged in  [R1] The taxi driver has accepted the passenger’s ride  [R2] The passenger has access at a data network |

|  |  |
| --- | --- |
| Name | **[PF18] Pay ride** |
| Requirements | [R0] The passenger must be already logged in  [R1] The taxi driver has accepted the passenger’s ride  [R2] The passenger is connected at a data network |

|  |  |
| --- | --- |
| Name | **[PF19] Conclude ride** |
| Requirements | [R0] The taxi driver must be already logged in and working  [R1] The taxi driver has accepted the passenger’s ride  [R2] The taxi driver has access at a data network  [R3] The passenger payed with cash/credit card |

|  |  |
| --- | --- |
| Name | **[PF20] List of rides** |
| Requirements | [R0] The user must be already logged in  [R1] The user has done at least one ride |

|  |  |
| --- | --- |
| Name | **[PF21] Modify reservation** |
| Requirements | [R0] The passenger must be already logged in  [R1] The passenger has at least one pending reservation  [R2] The passenger must insert valid data according to the format of each field  [R3] At least 10 minuts left from the starting time of the selected ride |

|  |  |
| --- | --- |
| Name | **[PF22] Cancel reservation** |
| Requirements | [R0] The passenger must be already logged in  [R1] The passenger has at least one pending reservation  [R2] At least 10 minuts left from the ride starting time selected by the passenger |

|  |  |
| --- | --- |
| Name | **[PF23] Race give up** |
| Requirements | [R0] The taxi driver has accepted a passenger’s ride  [R1] The taxi driver has not already picked up the passenger  [R2] The taxi driver is connected at a data network |

## 3.3 Scenarios

|  |  |
| --- | --- |
| Title |  |
| Related use case |  |
| Description |  |

|  |  |
| --- | --- |
| Title | Ride in process |
| Related use case |  |
| Description | Marco is headed to his destination into a taxi. When he arrives to the destination he has to pay and chooses to pay with credit card. Thus, he opens his app and he goes into “myTaxiRides” and then clicks “Pending rides”.  At the line of the current ride he finds a button called “Pay” and after clicking it he’s redirected to a page where he chooses from one of his credit cards inserted before or he can insert a new credit card. He decides to choose a credit card previously inserted and then he inserts the CVV of the card.  The transaction is confirmed from his credit card’s company.  Then the taxi driver confirms into his app that the ride is to consider concluded. |

|  |  |
| --- | --- |
| Title | Ride in process |
| Related use case |  |
| Description | Roberto is going to work with a taxi. Before arriving to his destination he sees that there’s a lot of traffic on his roads and decides to go on foot from that position. In that case the taxi driver inserts the current destination into the section of the current rides, after clicking “myTaxiRides” and then “Pending rides”. Consequently, the price is re-calculated and the client can see it into his app clicking on “myTaxiRides” and then “Pending rides”. He is in a hurry and pays the taxi driver with cash.  The taxi driver concludes the ride into his app clicking on “Conclude ride” related to that ride. |

|  |  |
| --- | --- |
| Title | Ride cancelation |
| Related use case |  |
| Description | Martina made a reservation of a taxi for tonight but this morning she remembered that a friend of her could bring her to the party she wants to go. Then, she decides to call her friend and asking to her if she really could bring her to her destination and her friend confirms so.  Thus, Martina decides to open the web site [www.myTaxiService.com](http://www.myTaxiService.com) and she logs into her account with her credentials. She clicks on “myTaxiRides” and then on “Pending rides”. Now she can see the list of all her pending rides and she clicks on the “Delete reservation” button related to the reservation of tonight.  The system checks if the ride can be canceled and notifies her of the correct cancelation because at that moment it’s missing 6 hours and 32 minutes to the ride. |

|  |  |
| --- | --- |
| Title | Ride modification |
| Related use case |  |
| Description | This morning Paolo made a reservation for a taxi for 8.30pm of today because he has to go to the airport to leave for his holidays.  At 7.30pm he receives a notification from the airline company that reports him that his flight is been deleted and postponed to tomorrow morning at 10am.  Hence, Paolo decides to modify his reservation and he opens his myTaxiService app, he logs with his account and he clicks on “myTaxiRides” and then on “Pending rides”. Now he can see the list of his pending rides and he chooses the modify option related to the ride reserved for 8.30pm of today by clicking on the button “Modify reservation”. Finally, he can choose the date and time of the new reservation, then he clicks OK.  Because there is more than 10 minutes left before the hour of the reservation, the system notifies Paolo of the correct modification of the reservation. |

|  |  |
| --- | --- |
| Title | Concluded/Pending ride details |
| Related use case |  |
| Description | Luca is doing a review of his personal calendar of the previous months and he sees that now he’s taking the taxi less and less comparing to the last months, so he has an idea: go into his app myTaxiRides app and have a confirmation of his thoughts.  He logs with his credentials into his app and goes to the section “myTaxiRides” and then into the subsection “Concluded rides”.  The system provides him a complete list of all his concluded rides and he can see the information related to them grouped by month in a graph and also listed with information like the number of people in the rides, date and time. |

## 3.4 Use cases

### 3.4.1 Sign up

|  |  |
| --- | --- |
| Name of the use case | [UC01] Sign-up |
| Actors | Visitor |
| Entry condition | NULL |
| Flow of events | 1. The Visitor clicks on “Sign-up” 2. The Visitor decide which kind of user he wants to be selecting “Taxi driver” or “Passenger” (based on this choice different form are shown 3. The Visitor fills at least all mandatory fields 4. The Visitor clicks on “Confirm” button 5. The system save the data and sends a confirmation email 6. The Visitor clicks on the confirmation link 7. The system activate his account |
| Exit condition | Visitor successfully ends registration process and become a Taxi driver or a Client |
| Exceptions | * 1. The Visitor is already logged-in   1.a. The Visitor is redirected to his personal page   * 1. One or more fields are not valid   4.b The Visitor is alerted and application goes back at point 3   1. The chosen email or other unique information are already associated to another user   4.c The Visitor is alerted and application goes back at point 3   1. The email doesn’t exist   5.d The Visitor is alerted and application goes back at point 3 |
| Special requirements | The email once is sent, it is assumed to be received by the mail account |

### 3.4.2 Login

|  |  |
| --- | --- |
| Name of the use case | [UC02] Login |
| Actors | Visitor, Client, Taxi driver |
| Entry condition | The visitor is registered into the system using either the mobile application or the web application |
| Flow of events | 1. The Visitor clicks on “Login” 2. The Visitor complete the form inserting correct email and password 3. The Visitor clicks on “Confirm” button |
| Exit condition | The Visitor is promoted to Taxi driver or Client |
| Exceptions | * 1. The Visitor is already logged-in   1.a. The Visitor is redirected to his personal page   * 1. The email and password are not valid   3.b The Visitor is alerted and application goes back at point 2   * 1. The email and password combination is wrong   3.c The Visitor is alerted and application goes back at point 2 |
| Special requirements | - |

### 3.4.3 Password recovery

|  |  |
| --- | --- |
| Name of the use case | [UC03] Password recovery |
| Actors | Client, Taxi driver |
| Entry condition | The visitor is registered into the system using either the mobile application or the web application |
| Flow of events | 1. The Client /Taxi driver clicks on “Forgotten password?” 2. The Client/Taxi driver complete the form inserting his registration email 3. The Client /Taxi driver clicks on “Confirm” button 4. The system sent a randomly generated password at the user email |
| Exit condition | The Taxi driver or Client get a password to access his myTaxiService account |
| Exceptions | 1. The Client /Taxi driver is already logged-in   1.a. The Client /Taxi driver is redirected to his personal page   1. The email is not valid   3.b The Visitor is alerted and application goes back at point 2   1. The email doesn’t belong to any user   3.c The Visitor is alerted and application goes back at point 2 |
| Special requirements | Once the email is sent, it is assumed to be received by the mail account |

### 3.4.4 Requesting a ride

|  |  |
| --- | --- |
| Name of the use case | [UC04] Requesting a ride |
| Actors | Client, Taxi Driver |
| Entry condition | The Client is logged in using either the mobile application or the web application |
| Flow of events | 1. The Client clicks on the “Find me a ride” button 2. The Client selects “Actual position” as the origin, the destination and “Right now” as the time of the ride, then clicks on the “Send request” button. 3. The request is sent to the system. 4. The system queries the database containing taxi queues to find the nearest Taxi Driver to the passenger. 5. The system sends a notification to the selected Taxi Driver to ask them whether they accept the request or not   6.i. The taxi driver choses to accepts the request  6.ii. The Taxi Driver doesn’t accept the request: the system moves the Taxi Driver to the last position of the queue and forwards the request to the second taxi in the queue, going back to event 5.   1. The system notifies the Client about the incoming Taxi Driver’s ID and the waiting time. |
| Exit condition | The Client has successfully booked a taxi |
| Exceptions | -- |
| Special requirements |  |

### 3.4.5 Reserving a ride

|  |  |
| --- | --- |
| Name of the use case | [UC05] Reserving a ride |
| Actors | Client, Taxi driver |
| Entry condition | The Client is logged in using either the mobile application or the web application |
| Flow of events | 1. The Client clicks on the “Find me a ride” button 2. The Client selects the origin, the destination and the time of the ride, then clicks on the “Send request” button.The request is sent to the system. 3. The system checks if the specified time of the ride is at least two hours after the current time: if not, the system notifies the Client, who has to change the time. 4. The system queries the database containing taxi queues to find the nearest Taxi Driver to the passenger. 5. The system sends a notification to the selected Taxi Driver to ask them whether they accept the request or not 6. i. The Taxi Driver choses to accepts the request   ii. The Taxi Driver doesn’t accept the request: the system moves the Taxi Driver to the last position of the queue and forwards the request to the second Taxi Driver in the queue, going back to event 5.   1. The system notifies the Client about the incoming Taxi Driver’s ID and the waiting time. |
| Exit condition | The Client has successfully reserved a taxi |
| Exceptions |  |
| Special requirements |  |

### 3.4.6 Shared ride

|  |  |
| --- | --- |
| Name of the use case | [UC06] Shared ride |
| Actors | Client, Taxi Driver |
| Entry condition | The Client is logged in using either the mobile application or the web application |
| Flow of events | 1. The Client clicks on the “Find me a ride” button 2. The Client selects the origin, the destination and the time of the ride and enables the “Shared ride” option, then clicks on the “Send request” button. The request is sent to the system. 3. The system identifies the zone selected as the origin of the ride and checks if there are other requests coming from that zone. 4. The system queries the database containing taxi queues to find the nearest Taxi Driver to the passenger. 5. The system sends a notification to the selected Taxi Driver to ask them whether they accept the request or not 6. i. The Taxi Driver choses to accepts the request: the system arranges the route for the Taxi Driver and computes the fee for all Clients sharing the taxi   ii. The Taxi Driver doesn’t accept the request: the system moves the Taxi Driver to the last position of the queue and forwards the request to the second Taxi Driver in the queue, going back to event 5.   1. The system notifies all passengers about the incoming Taxi Driver’s ID, the waiting time and the estimated fee.. |
| Exit condition | The Client has successfully booked a taxi and will share the ride with one or more other Clients. |
| Exceptions |  |
| Special requirements |  |

### 3.4.7 Payment via Credit Card

|  |  |
| --- | --- |
| Name of the use case | [UC07] Payment via Credit Card |
| Actors | Client |
| Entry condition | The client has successfully booked a ride |
| Flow of events | * + 1. The client opens the “myTaxiRides” tab from either the website or the mobile app and choses the “Pending Rides” tab     2. The client selects the ride that they want to pay for via Credit Card     3. The client choses whether they want to use a Credit Card that is already in their account or with a new one     4. i. The client choses the Credit Card they want to pay with   ii.1 The client selects their Credit Card’s company  ii.2 The system redirects the client to their Credit Card company’s website  ii.3 The client inserts the required information (number, expiration date and CVV) about their Credit Card and confirms the payment   * + 1. The system notifies both the taxi driver and the client about the successful transaction |
| Exit condition | The client has successfully paid for the ride |
| Exceptions | a. The transaction fails  5.a. The system notifies the client about the failure and asks them if they want to try again. If yes, the client repeats from event 3. If no, the client is redirected to the home page |
| Special requirements | The Credit Card needs to be valid |

### 3.4.8 Ride in process

|  |  |
| --- | --- |
| Name of the use case | [UC08] Ride in process |
| Actors | Client, Taxi driver |
| Entry condition | The client starts a ride with a taxi driver |
| Flow of events | 1. The ride proceeds well until its end 2. The client pays the taxi driver with cash/credit card 3. The taxi driver confirms the end of the ride |
| Exit condition | The ride is shifted into the concluded rides and the taxi driver into the queue of his/her current zones |
| Exceptions | 1. The taxi driver and the client agree with an interruption of the ride before its end 2. The taxi driver inserts into the app the current destination of the ride and is recalculates the price to pay 3. The client payment with credit card (SEE USE CASE OF PAYMENT) fails 4. The client needs to pay with cash 5. The data connection of the taxi driver doesn’t work   3. The system can’t put into a queue the taxi driver until he/she confirms the finish of the ride. He/she remains temporally out. |
| Special requirements | - |

### 3.4.9 Ride cancelation

|  |  |
| --- | --- |
| Name of the use case | [UC09] Ride cancelation |
| Actors | Client |
| Entry condition | The passenger is logged in using either the mobile application or the web application |
| Flow of events | 1. The user clicks on the button “myTaxiRides” in the app or in the web site to see the list of his/her concluded/pending rides 2. The user clicks on “Pending rides” 3. The client clicks on the button “Delete” related to a pending ride that is been reserved 4. The system checks if that ride can be canceled and allow the passenger to cancel it 5. The system notifies the client of the correct cancelation of the reservation |
| Exit condition | The client cancels the ride or doesn’t cancel it because there is less than 10 minutes left before the reservation |
| Exceptions | * + - 1. There’s less than 10 minutes left before the reservation   5. The system doesn’t allow the client to cancel the reservation because there’s less than 10 minutes left before the reservation   * + - 1. An error occurs during the cancelation process   6a. The system notifies the client of the non-cancelation of the ride because of an internal error  6b. The system notifies the client of the non-cancelation of the ride because the reservation is in less than 10 minutes |
| Special requirements | - |

### 3.4.10 Ride modification

|  |  |
| --- | --- |
| Name of the use case | [UC10] Ride modification |
| Actors | Client |
| Entry condition | The client is logged in using either the mobile application or the web application |
| Flow of events | 1. The user clicks on the button “myTaxiRides” in the app or in the web site to see the list of his/her concluded/pending rides 2. The user clicks on “Pending rides” 3. The client clicks on the button “Modify” related to a pending ride that is been reserved 4. The system checks if that ride can be modified and allow the passenger to modify it 5. The client inserts into a new page the new date and time desired and clicks on “OK” button 6. The system notifies the client of the correct modification of the reservation |
| Exit condition | The client modifies the ride or doesn’t modify it because there is less than 10 minutes left before the reservation |
| Exceptions | 1. There’s less than 10 minutes left before the ride   5. The system doesn’t allow the client to modify the reservation because there’s less than 10 minutes left before the reservation   1. An error occurs during the modification process   6a. The system notifies the client of the non-modification of the ride because of an internal error  6b. The system notifies the client of the non-modification of the ride because the reservation is in less than 10 minutes |
| Special requirements | - |

### 3.4.11 Concluded/Pending ride details

|  |  |
| --- | --- |
| Name of the use case | [UC11] Concluded/Pending ride details |
| Actors | Client, taxi driver |
| Entry condition | The client is logged in using either the mobile application or the web application |
| Flow of events | 1. The user clicks on the button “myTaxiRides” in the app or in the web site to see the list of his/her concluded/current rides 2. The user clicks on “Concluded rides” 3. The system provides the list of his/her concluded rides with different option 4. The user can see the list of his/her concluded rides and a summary in a graph of the number of rides grouped by a certain period of time |
| Exit condition | The system provides the passenger or the taxi driver the list of his/her concluded/pending rides |
| Exceptions | * + - 1. The user wants to see the “Pending rides”  1. The user clicks on “Pending rides” 2. The system provides the list of his/her pending rides with different option 3. The user can see his/her pending ride with the information about them like estimated arrival time of the taxi, information on the ride and possibility to delete/modify the ride if that is a reservation not yet allocated to a taxi (there’s more than 10 minutes left) |
| Special requirements | - |

## 3.5 Non Functional Requirements

The following non-functional requirements have been defined with compliance to the ISO/IEC 25010:2011 Product Quality Model.

### 3.5.1 Functional stability

|  |  |
| --- | --- |
| Name | [NFR-FS01] Functional completeness |
| Description | The set of function provided by the system and described in Chapter 3, Paragraph 3.2 “Functions” covers all the goals specified in Chapter 1, Paragraph 1.2 “Scope”. |

|  |  |
| --- | --- |
| Name | [NFR-FS02] Functional correctness |
| Description | The system shall provide the correct result 99,95% of the times. An incorrect execution (e.g. the farthest taxi is assigned to a client even though a closer one is available) would nullify the advantages that the system provides. |

|  |  |
| --- | --- |
| Name | [NFR-FS03] Functional appropriateness |
| Description | The set of function provided by the system and described in Chapter 3, Paragraph 3.2 “Functions”, facilitates the tasks specified in Chapter 1, Paragraph 1.2 “Scope”. |

### 3.5.2 Performance efficiency

|  |  |
| --- | --- |
| Name | [NFR-PE01] Time behavior |
| Description | Function ####taxi distribution##### shall be completed in less than 1 second in order to guarantee the ability to answer all requests with the correctness specified in [NFR-FS02] Functional correctness. |

|  |  |
| --- | --- |
| Name | [NFR-PE02] Resource utilization |
| Description | The system shall be stress-tested to verify whether it can maintain the level of performance tested in normal conditions. |

|  |  |
| --- | --- |
| Name | [NFR-PE03] Capacity |
| Description | The system shall be scaled to support concurrent access from at least 500,000 users |

### 3.5.3 Compatibility

|  |  |
| --- | --- |
| Name | [NFR-C01] Co-existence |
| Description | The system shall not have a detrimental impact on the products it shares the environment with. |

|  |  |
| --- | --- |
| Name | [NFR-C02] Interoperability |
| Description | The central database already exchanges information with other software using JSON and no major problems have been encountered. Therefore, the system shall use the same standard to interact with the pre-existent database.  Information about routes and traffic will be obtained using the Google Maps API. |

### 3.5.4 Usability

|  |  |
| --- | --- |
| Name | [NFR-U01] Appropriateness recognizability |
| Description | A beta version of the system shall be tested before its release by a random group of users to verify whether it is appropriate for their needs. |

|  |  |
| --- | --- |
| Name | [NFR-U02] Learnability |
| Description | The system shall provide a tutorial that covers all the main functions as stated in function ###tutorial### |

|  |  |
| --- | --- |
| Name | [NFR-U03] Operability |
| Description | The system shall provide an administration interface (function ###admin###). |

|  |  |
| --- | --- |
| Name | [NFR-U04] User error protection |
| Description | Only the main functions shall be shown in the user interface. |

|  |  |
| --- | --- |
| Name | [NFR-U05] User interface aesthetics |
| Description | The system shall provide clear user interfaces. |

|  |  |
| --- | --- |
| Name | [NFR-U06] Accessibility |
| Description | The system shall provide a tutorial that covers all the main functions as stated in function ###tutorial### and that can be checked at any moment by the user. |

### 3.5.5 Reliability

|  |  |
| --- | --- |
| Name | [NFR-R01] Maturity |
| Description | The system shall be reliable 99.99% of times in normal conditions. |

|  |  |
| --- | --- |
| Name | [NFR-R02] Availability |
| Description | The system shall be operational for at least 364 days per year and can be down at most for 6 consecutive hours. |

|  |  |
| --- | --- |
| Name | [NFR-R03] Fault tolerance |
| Description | The system shall be installed on replicated servers so that the failure of one can be compensated by the other. |

|  |  |
| --- | --- |
| Name | [NFR-R04] Recoverability |
| Description | The system shall be installed on replicated servers so that, in case one encounters a problem, data can be recovered from the other. |

### 3.5.6 Security

|  |  |
| --- | --- |
| Name | [NFR-S01] Confidentiality |
| Description | The system shall use a secure authentication protocol. |

|  |  |
| --- | --- |
| Name | [NFR-S02] Integrity |
| Description | The system shall be monitored by a dual firewall which will create a DMZ to prevent unauthorized access to data. |

|  |  |
| --- | --- |
| Name | [NFR-S03] Non-repudiation |
| Description | The system shall record all accesses and operations in a log file. |

|  |  |
| --- | --- |
| Name | [NFR-S04] Accountability |
| Description | The system shall keep track of users’ actions using a unique identifier. |

### 3.5.7 Maintainability

|  |  |
| --- | --- |
| Name | [NFR-M01] Modularity |
| Description | Each software module shall interact with other software modules using lists of parameters so that the coupling level is minimal and a change to one component has minimal impact on other components. |

|  |  |
| --- | --- |
| Name | [NFR-M02] Reusability |
| Description | Function ###computedistribution### can be used in other systems since it only requires information about taxis’ positions (obtained using GPSs) and geographical data (obtained using Google Maps API). |

|  |  |
| --- | --- |
| Name | [NFR-M03] Analysability |
| Description | The system will receive log files about failures of the application containing information necessary that will easily help identify which component caused the failure (e.g. OS and operation that caused the mobile app to crash) |

|  |  |
| --- | --- |
| Name | [NFR-M04] Modifiability |
| Description | The system shall be modifiable without introducing defects or degrading its quality before the change, as granted by [NFR-M01]. |

|  |  |
| --- | --- |
| Name | [NFR-M05] Testability |
| Description | Test case shall be specified (as comments) in each component of the source code. |

### 3.5.8 Portability

|  |  |
| --- | --- |
| Name | [NFR-M01] Adaptability |
| Description | The mobile application shall be developed for the three major mobile OS (Android, iOS and Windows Phone).  The website shall be easily accessed from the main web browsers (Google Chrome, Internet Explorer, Microsoft Edge, Mozilla Firefox, Safari). |

|  |  |
| --- | --- |
| Name | [NFR-M02] Installability |
| Description | The mobile application shall be easily installed and uninstalled regardless of the mobile operating system (Android, iOS and Windows Phone) using the proprietary application store. |

|  |  |
| --- | --- |
| Name | [NFR-M03] Replaceability |
| Description | The software can easily replace the already existent application “mytaxi”. |

## 3.6 Alloy models

# 4. Appendixes

## 4.1 Used tools

* Microsoft Word 2013 to write this document
* Microsoft Visual Studio 2013 to create UML diagrams
* Balsamiq Mockup 3 to create mockups of the mobile application and of the web site

## 4.2 Attached documents

### 4.2.1 Client’s document

#### The problem: myTaxiService

#### Part I

The government of a large city aims at optimizing its taxi service. In particular, it wants to: i) simplify the access of passengers to the service, and ii) guarantee a fair management of taxi queues.

Passengers can request a taxi either through a web application or 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.

The system guarantees a fair management of taxi queues. In particular, the city is divided in taxi zones (approximately 2 km2 each). Each zone is associated to a queue of taxis. The system automatically computes the distribution of taxis in the various zones based on the GPS information it receives from each taxi. When a taxi is available, its identifier is stored in the queue of taxis in the corresponding zone.

When a request arrives from a certain zone, the system forwards it to the first taxi queuing in that zone. If the taxi confirms, then the system will send a confirmation to the passenger. If not, then the system will forward the request to the second in the queue and will, at the same time, move the first taxi in the last position in the queue.

Besides the specific user interfaces for passengers and taxi drivers, the system offers also programmatic interfaces to enable the development of additional services (e.g., taxi sharing) on top of the basic one.

#### Part II

A user can reserve a taxi by specifying the origin and the destination of the ride. The reservation has to occur at least two hours before the ride. In this case, the system confirms the reservation to the user and allocates a taxi to the request 10 minutes before the meeting time with the user.

#### Part III

A user can enable the taxi sharing option. This means that he/she is ready to share a taxi with others if possible, thus sharing the cost of the ride. In this case the user is required to specify the destination of all rides which he/she wants to share with others. If others are willing to start a shared ride from the same zone going in the same direction, then the system arranges the route for the taxi driver, defines the fee for all persons sharing the taxi and informs the passengers and the taxi driver.

## 4.3 Hours of work

Each member of the group has worked for 28 hours (31/10/15 h 12.00)