Requirements Analysis and Specifications Document

Giorgio Pea(Mat. 000000), Andrea Sessa(Mat. 850082)

Contents

1 Introduction

1.1 Purpose

This document represent the Requirement Analysis and Specification Document (RASD). The main goal of this document is to completely describe the system in terms of functional and non-functional requirements, to show the constraints and the limit of the software and simulate the typical use cases that will occur after the development. This document is intended to all developers and programmers who have to implement the requirements, to system analysts who want to integrate other systems with this one, and could be used as a contractual basis between the customer and the developer.

1.2 Actual System

The whole project is built on the assumption that no preexistent system exists. Every feature and component of the new system has to be built without modifying or reusing parts of a previous system.

1.3 Scope of the project

The aim of this project is to develop MyTaxiService, a web/mobile application that makes easier and quicker taking taxies within the city's borders. Thanks to MyTaxiService, anyone can request or book a taxi and get realtime information about how long it will take to be picked up or about the taxi's current position and identification code. In addition to that, MyTaxiService provides an efficient way to allocate taxies by dividing the city in zones and using a queue based allocation system, in order to reduce the average waiting time and city's traffic.

1.4 Future Implementations

Further possible modifications/extension are represented by adding to the described system the possibility of managing "Taxi Sharing". In this case users are allowed to "share" a taxi ride with others users going in the same directions.

1.5 Goals

1. Simplify and speed up the process of taking a taxi within the city's borders

   (a) When a registered user has entered his taxi ride details and clicks or taps the "REQUEST TAXI" button, MyTaxiService will find the first available mtaxi that fits for the inserted ride details, and this mtaxi will pick up the registered user as soon as possible

   (b) When a registered user has entered his booking details and clicks or taps the "BOOK TAXI" button, MyTaxiService will book a mtaxi that fits for the inserted booking details and for the indicated pick up time

2. Guarantee an efficient and fair management of taxi queues

    (a) Guarantee a balanced distribution of mtaxies in the city, in order to have them always available for different city zones

    (b) Guarantee short mtaxi availability times and short waiting times

1.6 Assumptions

1. MyTaxiService has been commissioned by the city's local government.

2. The local government provides each mtaxi with a device similar to a small navigator(MYT). This device is used to: see and accept ride/booking requests, see zone change orders, report accidents; it has a built in gps module through which MyTaxiService can track the position of the mtaxi on which is mounted

3. Mtaxies can only serve ride/booking requests forwarded by MyTaxiService(B)

4. Mtaxies operate only whithin the city's borders

1.7 Terms and definitions

User
    A generic person not registered to MyTaxiService

Passenger
    a registered user that has requested a mtaxi ride/booking

Registered User
    A generic person that is registered to MyTaxiService

Mtaxi
    a taxi that joined MyTaxiService

MyTaxiService(B)
    The back end of MyTaxiService, that is to say, those software components that manage the forwarding of ride/booking requests to mtaxi drivers, the search of available mtaxies compatible with the received ride/booking requests, the distribution of mtaxies in the different city's zones and other internal tasks not exposed to users or mtaxi drivers

Ride
    A movement of people, through a taxi cab, from one location to another

Ride request
    An electronic message sent by a registered user to MyTaxiService(B). This electronic message refers to the case in which the registered user wants to be picked up as soon as possible by a mtaxi

Booking request

An electronic message sent by a registered user to MyTaxiService(B). This electronic message refers to the case in which the registered user wants to be picked up by a mtaxi at a specific date and time

Mtaxi booking, booking a ride

If a registered user wants to be picked up by a mtaxi at specific date and time, he books a ride or a mtaxi

MYT

The device the local government provides to taxi drivers that have registered to MyTaxiService. This device integrates a client for MyTaxiService that is used by the mtaxi driver to see ride/booking requests and zone change orders, accept these requests and report accidents

A mtaxi is available

A mtaxi has finished serving a passenger and is ready for a new ride

A mtaxi is unavailable

A mtaxi is considered unavailable if and only if or is serving a passenger or has reported an accident or is stuck in the traffic(and this has been detected by MyTaxiService(B))

Zone

An area of the city about $2km^2$ wide

Credentials

A combination of username and password, used by a registered user to access the MyTaxiService application

Queue

A data structure managed with a FIFO policy.

Ride/booking request acceptance

A mtaxi driver has accepted a ride/booking request if he has left in order to pick up the passenger associated with the request

The city

The city that is served by MyTaxyService and its taxies

1.8 Acronyms

- **RASD:** Requirements Analysis and Specification Document

- **FIFO:** First In First Out

1.9 Reference Documents

- Specification Document: MyTaxiService-AA2015-2016.pdf

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

- IEEE Std 1016 tm -2009 Standard for Information Tecnology-System Design-Software Design Descriptions.

1.10 Main Actors

See Terms and Definitions section

- User

- Registered User

- Mtaxi driver

- Taxi driver

- MyTaxiService(B)

2 Requirements Specification

2.1 Functional requirements

The following table shows the only functionalities of MyTaxiService accessible by the different actors:

1. USER

   (a) User registration

2. REGISTERED USER

   (a) Login
   (b) Remember me (see below for details)

3. LOGGED REGISTERED USER

   (a) Logout
   (b) Request a ride
   (c) Book a ride
   (d) View the status of a ride (requested or booked)
   (e) Undo a ride booking (If possible)

4. TAXI DRIVER

    (a) Taxi driver registration

5. MTAXI DRIVER

    (a) Accept a ride request/booking

    (b) Complete a ride

    (c) View the details of ride request/booking

    (d) Report an accident

6. A user can register to MyTaxiService only and only once

7. MyTaxiService has to be able to remember logged sessions so that when a registered user opens next time the MyTaxiService app or website, he has not to log in again (Remember me functionality)

8. MyTaxiService has to implement a user registration process that requires the user to insert: name, lastname, email, password

9. MyTaxiService has to implement a taxi driver registration process that requires the user to insert: name, lastname, email, password

10. During the user registration process, MyTaxiService has to be able to check the formal validity of the data inserted by the user

11. MyTaxiService has to only accept user registrations with mail addresses not already used by other registered users

12. MyTaxiService has to only accept registrations whose data formal validity has been check successfully

13. Logged registered users can undo a reservation only 10 minutes before the inserted picking up time

14. MyTaxiService has to allow multiple mtaxi bookings by the same logged registered user

15. MyTaxiService has not to allow multiple ride requests by the same logged registered user (a registered user can't make a new ride request until his last requested ride has been completed)

16. MyTaxiService can accept only ride requests that specify a starting location, an ending location and a number of passengers

17. MyTaxiService has to be able to access a list of every valid city's location

18. MyTaxiService has to be able to access realtime data about the city's traffic

19. MyTaxiService has to force a logged registered user to choose as starting/ending locations of a ride only valid city's locations

20. MyTaxiService has to force a logged registered user to chose among a list of only valid city's locations

21. MyTaxiService has to force a logged registered user not choose the same location as starting and ending location of a ride

22. MyTaxiService(B) has to be able to retrieve the position of each mtaxies using the gps system of their MYT devices

23. MyTaxiService(B) has to be able to logically divide the city in zones and associate sets of gps coordinates to these zones

24. MyTaxiService(B) has to be able to map a gps coordinate with a valid city's location

25. MyTaxiService(B) has to be able, using the mtaxies' gps data and the city's public data, to understand the city's traffic situation zone by zone (High traffic, normal traffic, low traffic)

26. MyTaxiService(B) has to be able, using the mtaxies' gps coordinates, to analyze their distribution among the different city's zones and understand if there's the need to change this distribution in relation to the average number of ride/booking requests per city's zone.

27. MyTaxiService(B) has to be able to send change zone order to mtaxies

28. MyTaxiService(B) has to be able, using mtaxies' gps coordinates, to analyze the mtaxies' distribution among the different city's zone and build queues of available mtaxies for each zone

29. MyTaxiDriver(B) has to manage the forwarding of ride/booking requests using the queue based system discussed above: a ride/booking request is forwarded to the first mtaxi in the queue that corresponds to the city's zone of the requested ride's starting location

30. MyTaxiService(B) has to be able to manage a situation in which a mtaxi does not accept a ride/booking request within a fixed time. In this situation MyTaxiService has to find a new mtaxi to forward the request

31. MyTaxiService app and website must have the same functionalities

32. MyTaxiService(B) has to have access to the characteristics of each mtaxies (number of passengers, unique code)

33. MyTaxiService(B) has to have access to the mtaxi drivers' work timetable, in order to know when a mtaxi should start serving passengers, should take a lunch break or should finish all activities

34. MyTaxiService has to consider unavailable only mtaxies that or are serving a ride/booking request or have reported an accident or are taking a break according to the work timetable, no other exceptions are allowed

35. MyTaxiService has to offer to mtaxi drivers the possibility of reporting an accident and therefore be considered unavailable

2.2 Non functional requirements

1. MyTaxiService has to guarantee that a mtaxi arrives at the start location of a ride within 10 minutes

2. A mtaxi driver has to accept a ride/booking request within 2 minutes

3. After a failure MyTaxiService has to be restarted within 1 hour

4. MyTaxiService must be available 24h/7d

5. Registered users have to be able to use properly all MyTaxiService's functionalities after a training time of at most 15 mins

6. All MyTaxiService's functionalities have to be reachable by a registered user with at most 3 clicks/taps from the after login screen of the web/mobile app
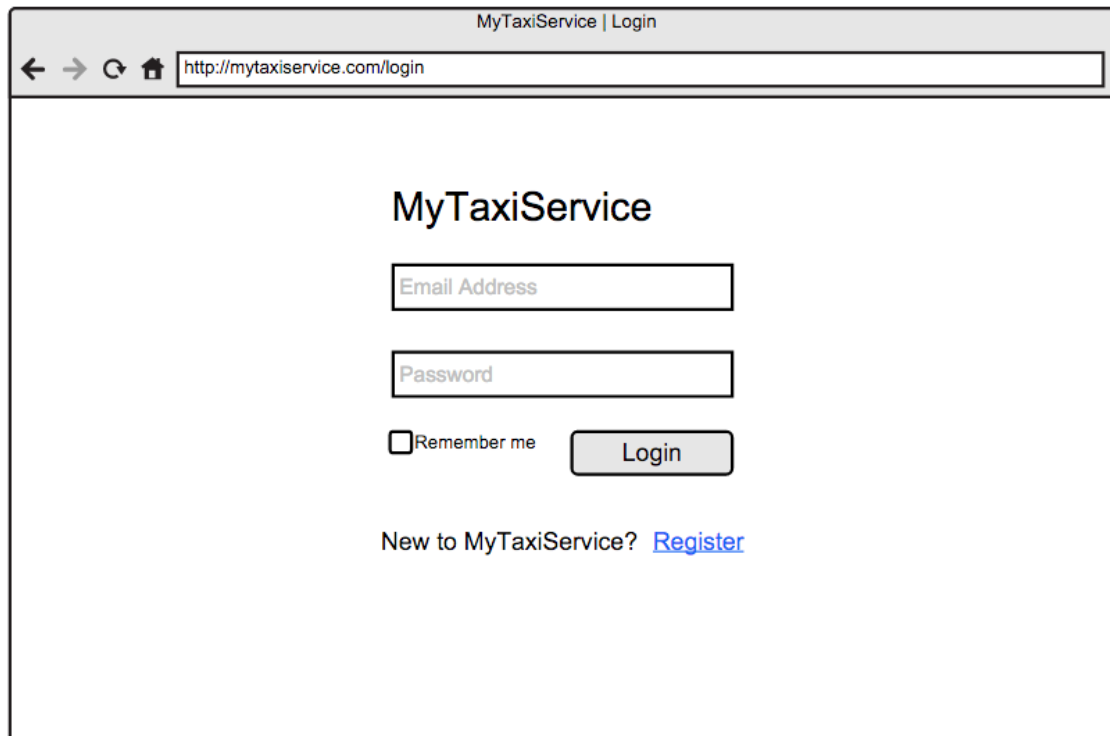
2.2.1 Security

Among non functional requirements particular attention must be devoted to security aspects. During the login MyTaxiService has to prevent malicious users from phishing sensible data(password, username etc) by adopting suitable protection mechanism(password encryption, encrypted communication protocols etc). From the server side MyTaxiService must guarantee that data access from remotes users is controlled and filtered such that no unauthorized access(SQL injection, MITM attack, etc) to sensible data is permitted.

2.2.2 Interfaces Requirements

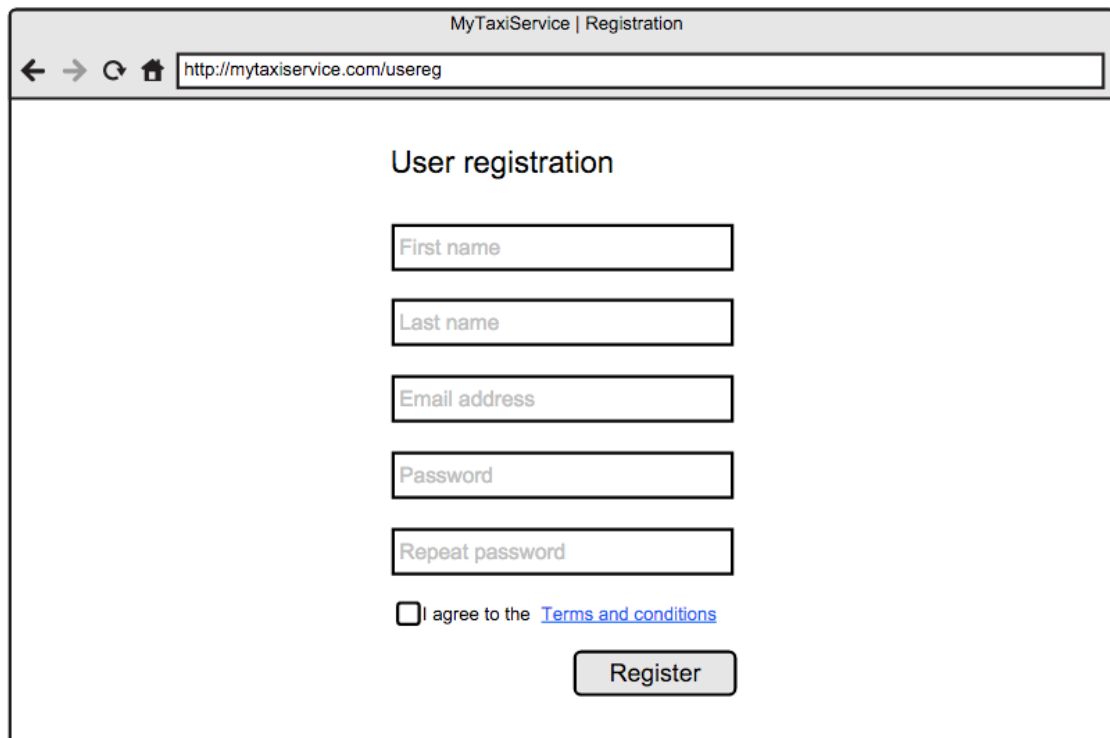In this section is shown a series of mock up that specify a set of requirements over the user interface.
Each mock up is valid both for the web site and mobile application version of MyTaxiService

This mock up shows the user login page

This mock up shows the user registration page

This mock up shows the user ride request page

This mock up shows the user ride booking page

This mock up shows the empty booking/ride request view page
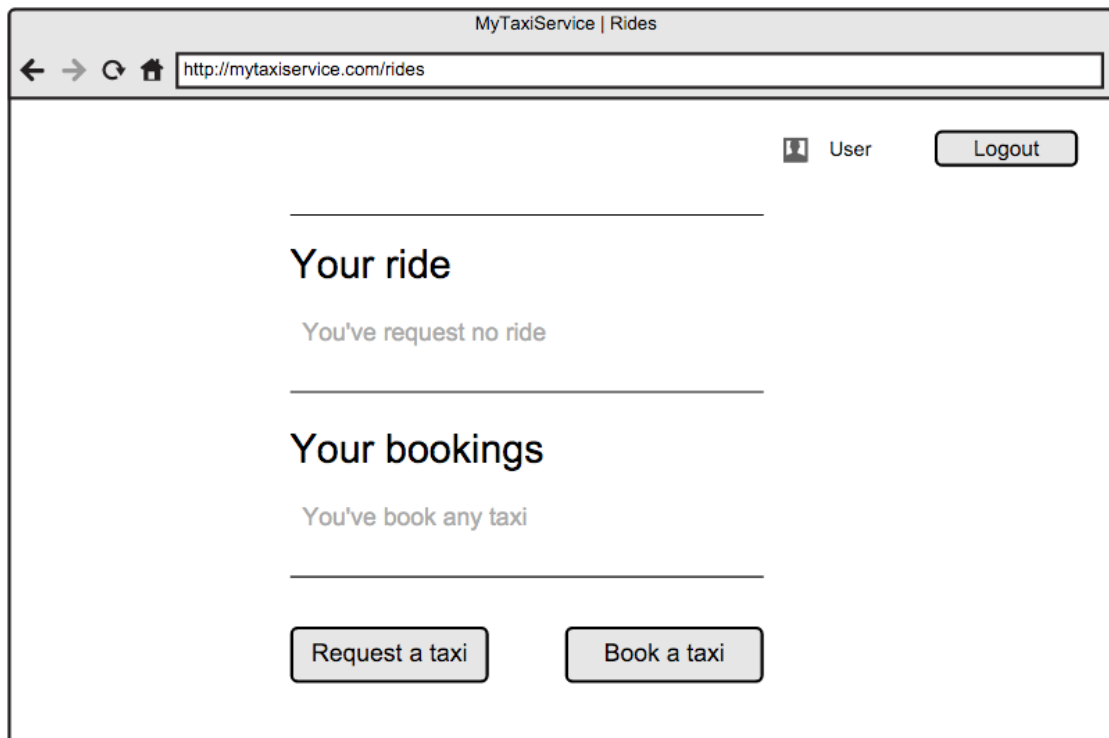
This mock up shows the booking/ride request view page when the user has requested a ride
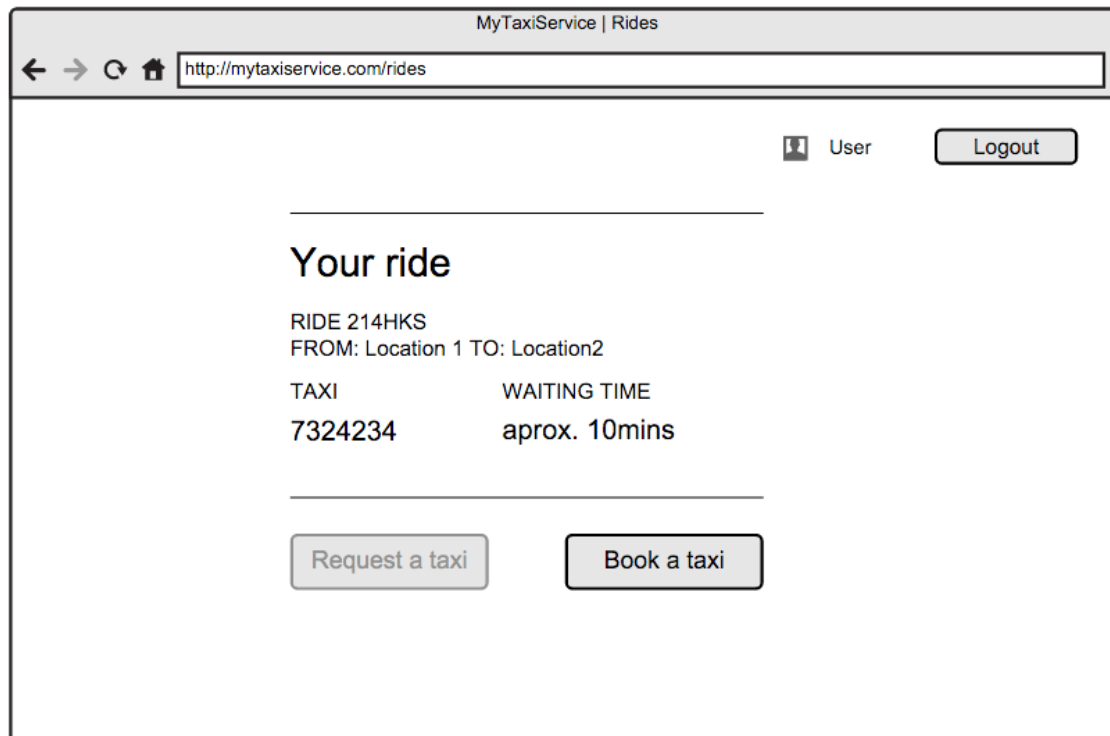
This mock up shows the booking/ride request view page when the user has booked and requested a ride

This mock up shows the deletion of a booking request by a registered user
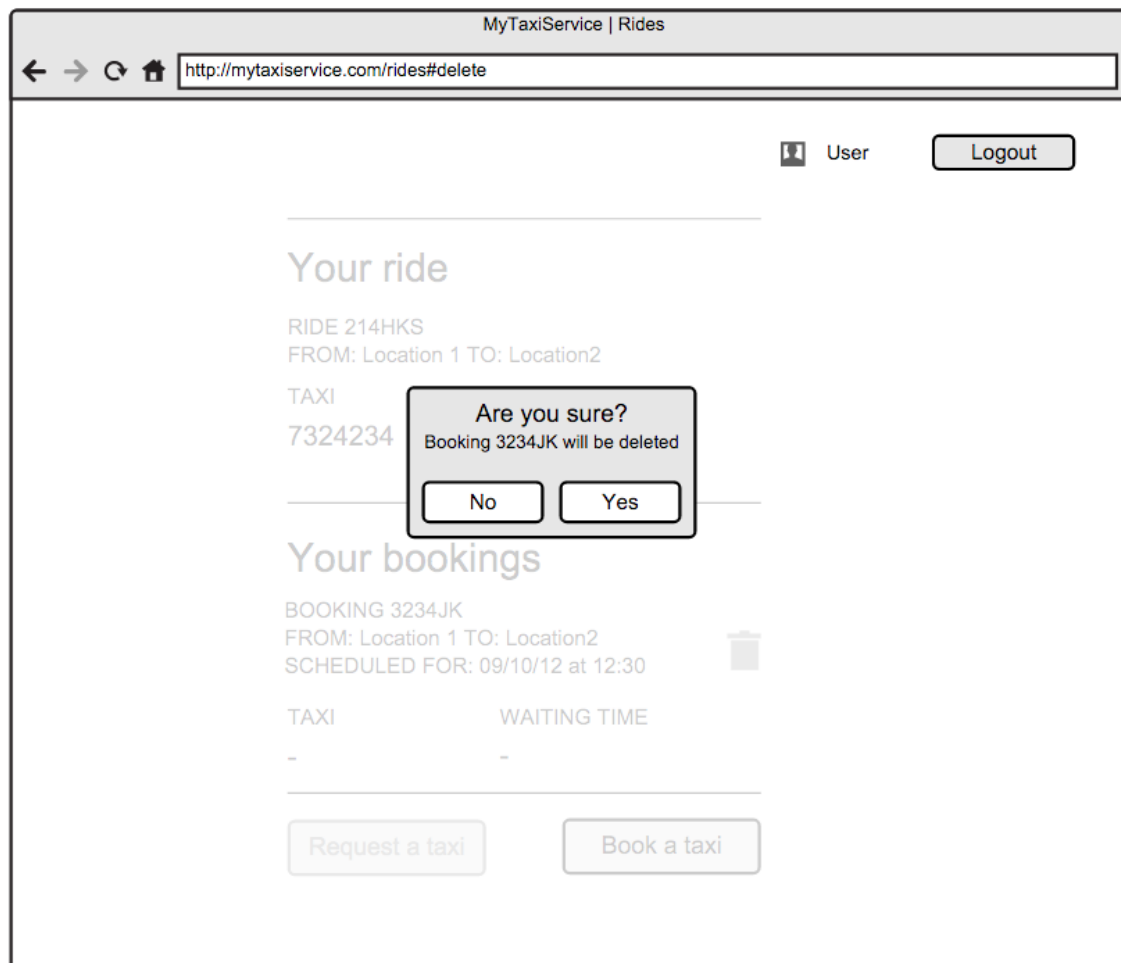
2.3 Constraints

1. MyTaxiService rquires Android 2.0 (or later) or IOS 6.0 (or later)

2.4 Domain Properties

1. Traffic information are reliable

2. GPS data are reliable

3. If a registered user request a RideRequest then then he will be present at the starting point when the taxi arrives

3 Scenarios, Use Case and UML diagrams

3.1 Scenarios

1. Bob is a product manager with a very busy schedule. At 11.00 he has a meeting with a group of clients to discuss about the features of a new product. This meeting is located in a another part of the city. For his shifts Bob usually takes taxies, and, in order to speed up this process, he started using MyTaxiService. It's 10:30 and Bob decides to take a taxi via MyTaxiService. He opens the app, the app remembers his credentials and logs him in. Bob inserts the address of the meeting and of his current location, choses one passenger and taps the "REQUEST TAXI" button. MyTaxiService then alerts him that is finding a mtaxi. Near Bob's location, a mtaxi has finished serving a passenger, so MyTaxiService(B) signals to it Bob's request. Charlene, the driver of the mtaxi, despite the fact that it's still work time, she wants to take a coffee and so ignores the request. After 5 mins MyTaxiService(B) tries to find a new mtaxi for Bob: Albert's mtaxi is available and MyTaxiService(B) forwards Bob's request to it. Albert receives the request and alerts MyTaxiService(B) that he's going to take care of the ride. MyTaxiService alerts Bob that his taxi will arrive in 3 mins. The Taxi arrives, and in 10 mins Bob reaches the meeting place. MyTaxiService(B) signals the behavior of Charlene to her supervisors, and, after a few weeks, Charlene receives a warning and a fine of 200 euro

2. It's Friday, 10 a.m, Paris, Ann, a young american tourist, has to take a flight at 12 a.m to return to New York City. Ann has carefully planned the return trip, infact yesterday she made a taxi booking via MyTaxiService web site for 9:45 a.m , but the mtaxi has not arrived yet. So she take her smartphone opens the MyTaxiService application, enters her credentials and after a successful login she discovers that her mtaxi is stuck in a terrible traffic jam and that MyTaxiService is trying to find a new mtaxi. "I'll miss my flight!" she thinks, and starts to wait with the heart full of hope

3. Dave, an experienced taxi driver of the city, has received, through MyTaxiService, a request for a ride from a passenger. "It's not too much far from where i am now", he thinks, "It'll be easy", and notifies MyTaxiService that he'll take care of the request. He starts driving toward the passenger's position, but suddenly , while he's crossing a very busy crossroad, a reckless driver coming from the left crosses at high speed the crossroad and hits with violence Dave's taxi cab. Luckily nobody gets hurt, but Dave's taxi cabe has been seriously damaged. Dave cannot fulfill the passenger's request so he takes his smartphone, opens the MyTaxiService app, and taps the "REPORT ACCIDENT" button, he accesses a screen where he inserts the accident details and taps the "SUBMIT" button. MyTaxiService(B) receives the Dave's report and after some computations selects and notifies another driver to fulfill the passenger request.

4. Ann, a young female student from Texas, is visiting Italy with his boyfriend Bob. It's Friday morning, they are walking on the streets of Milan and they're thinking about visiting the Expo on Saturday. Ann proposes to use a taxi to reach Expo's location and Bob agrees. Ann is a very cautious person so she thinks it's better to book a taxi, for this reason she searches on Google "taxi milano book" and discovers MyTaxiService. She downloads the app on her Iphone; she opens the app, which displays a login screen with a registration button, and taps that button being so redirected to a registration screen; she inserts her mail address, choses and confirm a password, accepts the legal terms of the app and taps the confirm button; after that Ann is ready to use the app. In order to book a taxi for Saturday for the Expo, Ann accesses the booking section of the app, taps the "ADD BOOKING" button and in the add booking screen inserts the Expo's location by selecting the right item in a dropdown menu, inserts the time and location of the picking , choses two passengers and taps the big "BOOK TAXI" button, MyTaxiService responds by alerting Ann that her booking has been processed successfully. Ann is happy about the fact that everything has proceeded without problems. Bob instead is not so happy because he thinks that the Milan's transportation system is perfect to reach Expo and its far way cheaper than taking a taxi. Bob and Ann have a brief discussion about this and, at the end, Ann agrees with Bob. Given that, Ann tries to undo the booking: she opens the app, she logs in, she accesses the booking section and there she sees her booking for Saturday, she taps the bin icon to delete the booking, she confirms her intention to do that and everything is done.

5. It's Thursday, 14:00 and George, a young mtaxi driver, is driving in the city's center. He noticed many of his colleagues waiting for requests in their cabs. Meanwhile MyTaxiService(B), by making some analysis on the mtaxies' GPS data, realizes that there are too many mtaxies in the city's center in relation with the number of ride requests for that zone. The system also indentifies some zones of the city where the concentration of taxies is too low. According to a policy based on some statistical data MyTaxiService(B) decides to move 10 mtaxies, including the George's one, from the city's center to the city's zones mentioned above. As a consequence of that, George receives by MyTaxiService a notification that alerts him to move to zone C (one of the city's zones with low concentration of mtaxies). George start driving towards zone C and continues his work.

3.2 Use cases of the application

3.2.1 General use case diagram
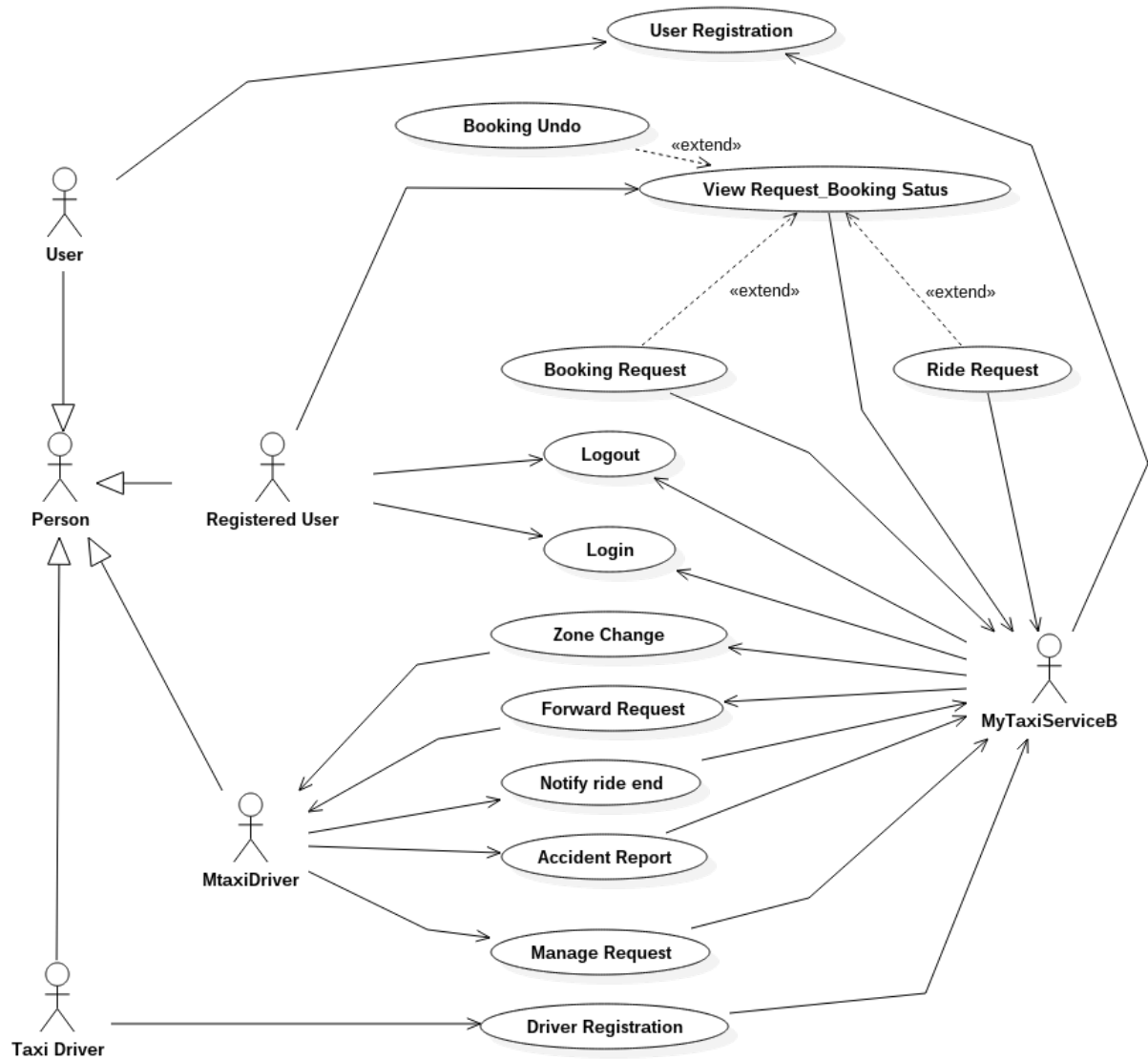
1. A registered user makes a ride request

   Use case name: Ride request

   Actors: A registered user, MyTaxiService(B)

   Pre-cons: The registered user has logged into MyTaxiService.

Post-cons: MyTaxiService(B) receives the ride request and tries to find an available taxi to fulfill it.

Entry Cnd: A registered user wants to be picked up by a mtaxi as soon as possible in a given place.

Exit Cnd: The registered user is acknowledged that MyTaxiService(B) has received successfully the ride request

Exceptions

   (a) Connection lost(for the mobile app): the registered user is immediately notified of the event and asked to try again later
   (b) Invalid input: the registered user is asked to reenter the information

Flow of events

   (a) The registered user taps/clicks the "REQUEST TAXI" button
   (b) The registered user is redirected to a screen dedicated to the insertion of the requested ride's details
   (c) The registered user fills out a form for the ride request inserting the ride's starting and ending location (chosen from a list of every valid city's location) and the number of passengers
   (d) The registered user submits to MyTaxiService(B) the form by clicking/tapping the "CONFIRM REQUEST" button

Sequence diagram
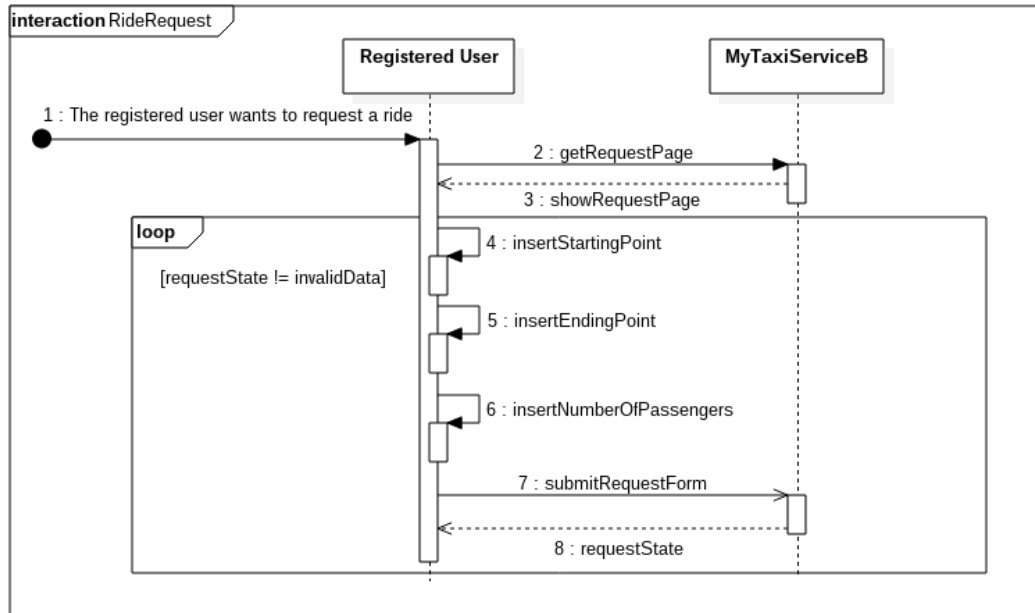
2. A registered user makes a booking request

Use case name: Booking request

Actors: A registered user, MyTaxiService(B)

Pre-cons: The registered user has logged into MyTaxiService.

Post-cons: MyTaxiService(B) receives the booking request and registers it; 10 mins before the chosen picking up time, MyTaxiService(B) will try to find an available taxi to fullfill the request

Entry Cnd: A registered user wants to book a mtaxi in order to be picked up at a specific date and time

Exit Cnd: The registered user is acknowledged that MyTaxiService(B) has received successfully the ride request

Exceptions

    (a) Connection lost(for the mobile app): the registered user is immediately notified of the event and asked to try again later

    (b) Invalid input: the registered user is asked to reenter the information

Flow of events

    (a) The registered user taps/clicks the "BOOK TAXI" button on the booking/request ride page of the app/website

(b) The registered user is redirected to a screen dedicated to the insertion of the requested booking details

(c) The registered user fills out a form for the booking request, inserting the ride's starting and ending location, the number of passengers, the date and the time for which be picked up.

(d) The registered user submits to MyTaxiService(B) the form by clicking/tapping the "CONFIRM BOOKING" button

Sequence diagram
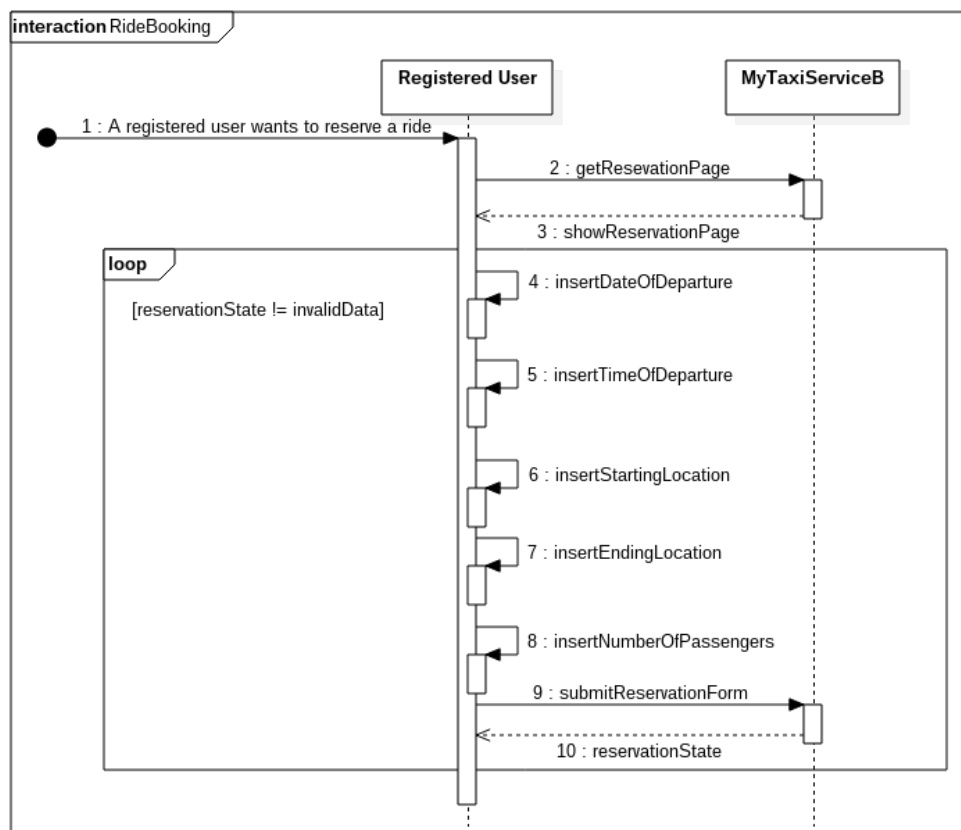
3. MyTaxiService(B) forwards a received ride request to a mtaxi driver

Use case name: Forward request

Actors: A mtaxi driver, MyTaxiService(B)

Pre-cons: The mtaxi driver has informed MyTaxiService(B) that he has finished serving a passenger and so is ready for a new ride

Post-cons: The mtaxi driver knows that a registered user has requested a ride with the specified details and is ready to fulfill it

Entry Cnd: MyTaxService(B) has received a ride request from a logged registered user

Exit Cnd: MyTaxiService(B) is acknowledged that the mtaxi driver has received successfully the ride request

Exceptions

    (a) No mtaxi is available for the city's zone relative to the ride request: MyTaxiService(B) tries to find a mtaxi that fits the ride request details(in terms of number of passengers) and that is located in the city's zones adjacent to the one above mentioned

    (b) No mtaxi available: MyTaxiService(B) notifies the registered user who made the request, that no mtaxi is available and that he should try again later

Flow of Events

    (a) MyTaxiService(B) extracts the city zone relative to the starting location of the ride request

    (b) MyTaxiService(B) selects a mtaxi that suits the ride requests details (in terms of number of passengers) from the queue of available mtaxies for the city zone above mentioned

    (c) MyTaxiService(B) checks if the traffic conditions in the zone where the selected mtaxi is located are acceptable (they allow the mtaxi to reach the registered user to pick up within 10 minutes)

    (d) MyTaxiService(B) forwards the ride request to the driver of the selected mtaxi

Sequence diagram
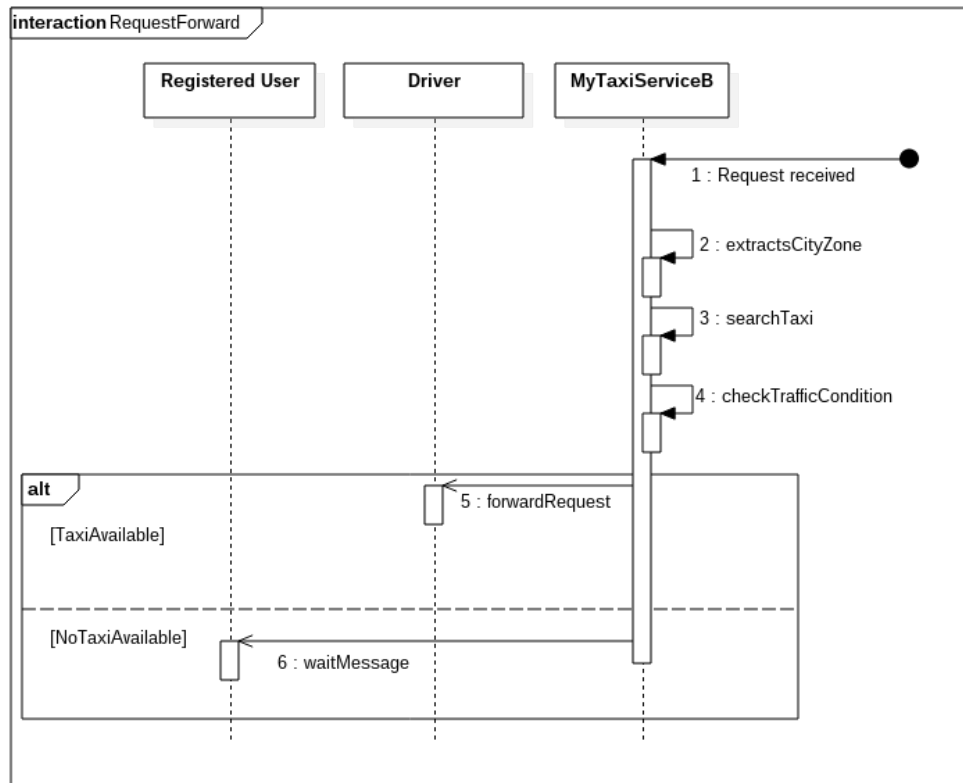
4. A mtaxi driver informs MyTaxiService(B) that he has finished serving a passenger

   Use case name: Notify ride end

   Actors: A mtaxi driver, MyTaxiService(B)

   Pre-cons: The mtaxi current location is the ending location of his last ride

   Post-cons: MyTaxiService(B) considers the mtaxi available

   Entry Cnd: The mtaxi driver has finished serving a passenger

   Exit Cnd: The mtaxi driver is acknowledged that MyTaxiService(B) has received successfully his notification
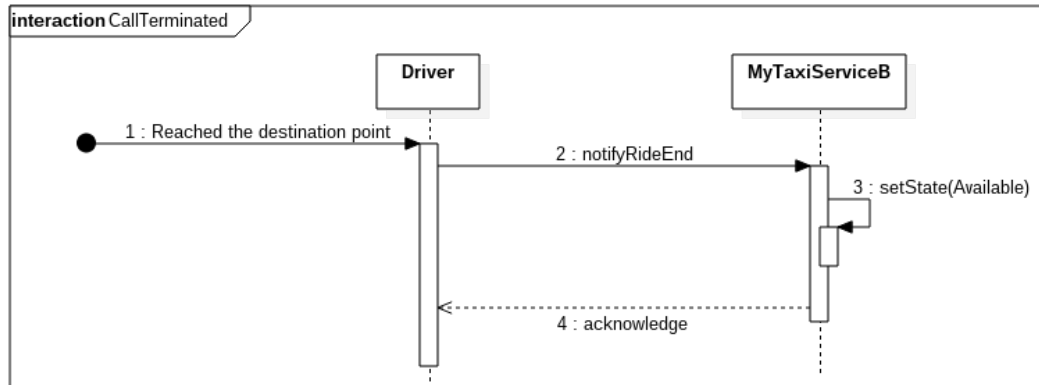
   Exceptions

   (a) Connection Lost: the mtaxi driver is immediately notified of the event and is asked to try again later

   Flow of Events

   (a) The mtaxi driver taps the "END RIDE" button on the screen of his MYT and so notifies MyTaxiService(B) that he has ended serving a passenger and is now available for other ride/booking requests

24

Sequence diagram

5. A mtaxi driver manages a ride request from MyTaxiService(B)

Use case name : Manage request

Actors : A mtaxi driver, MyTaxiService(B)

Pre cons : The mtaxi has to be available

Post cons : The mtaxi driver has started driving to the starting location of the requested ride

Entry Cnd : MyTaxiService(B) has forwarded successfully a ride request to the mtaxi driver

Exit Cnd : The mtaxi driver is acknowledged that MyTaxiService(B) has received successfully his notification of ride acceptance.

Exceptions

(a) If MyTaxiService(B) does not receive the acceptance of the ride request by the mtaxi driver within 5 mins, it selects another mtaxi to fullfil the request, and signals to the supervisors of the mtaxi driver in discussion his behavior

Flow of Events

(a) The mtaxi driver taps the "ACCEPT REQUEST" button on the screen of his MYT, and so notifies MyTaxiService(B) that he's going to take immediately care of the request

Sequence diagram
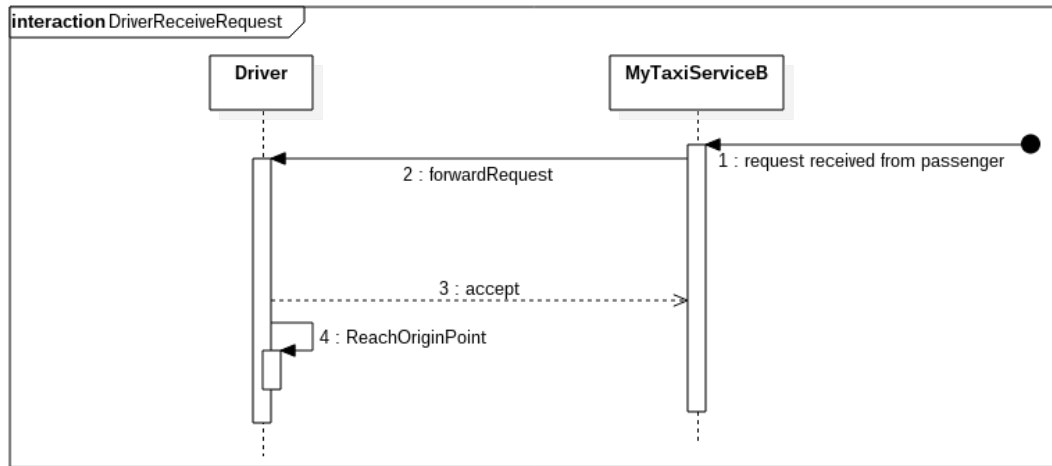
6. A registered user logs into MyTaxiService

Use case name: Log in

Actors: A registered user, MyTaxiService(B)

Pre Cnd: The registered user is not logged into MyTaxiService. The registered user has access to MyTaxiService app or website

Post Cnd: The registered user has access to all the functionalities of MyTaxiService reserved to registered users(see functional reqs)

Entry Cnd: The registered user wants to access the functionalities offered by MyTaxiService

Exit Cnd: The registered user is acknowledged that MyTaxiService(B) has recognized as valid the inserted credentials. MyTaxiService(B) will grant the registered user the access to all the functionalities reserved to registered users(see functional reqs)

Exceptions

    (a) Connection Lost: the registered user is immediately notified of the event and the entire process has to be redone

    (b) Wrong Credentials(credentials that do no match any registered used): the registered user is asked to insert his credentials again

Flow of Events

    (a) The registered user opens the MyTaxiService app or access the MyTaxiService website, and a login screen is shown, along with the option to register to the service and to rembember the login

(b) The registered user inserts his credentials in the relative fields

(c) The registered user clicks/taps the "LOGIN" button

Sequence diagram

7. A registered user logs out from MyTaxiService

Use case name: Log out

Actors: A registered user, MyTaxiService(B)

Pre cons: The registered user is logged into MyTaxiService

Post cons: The registered user has no access to MyTaxiService functionalities until a new login

Entry Cnd: The registered user wants to logout from MyTaxiService

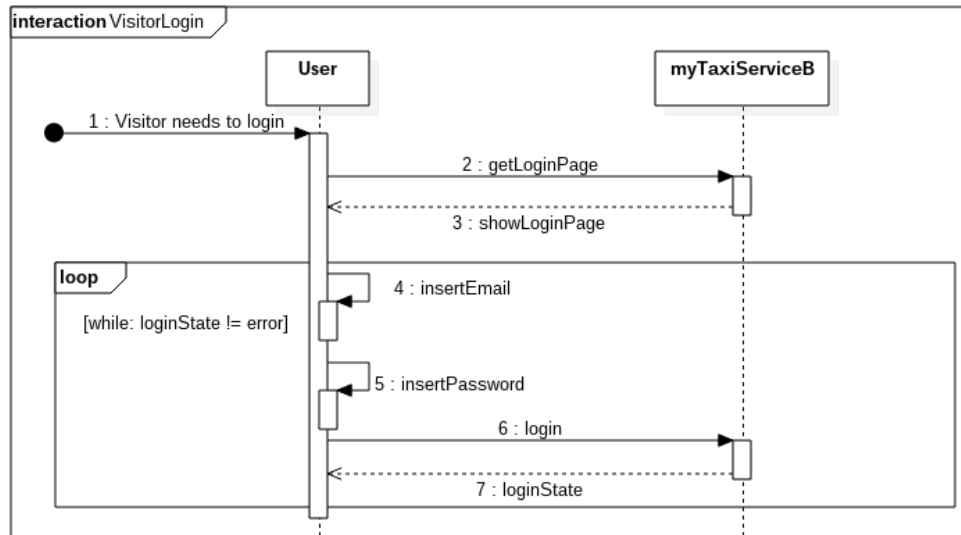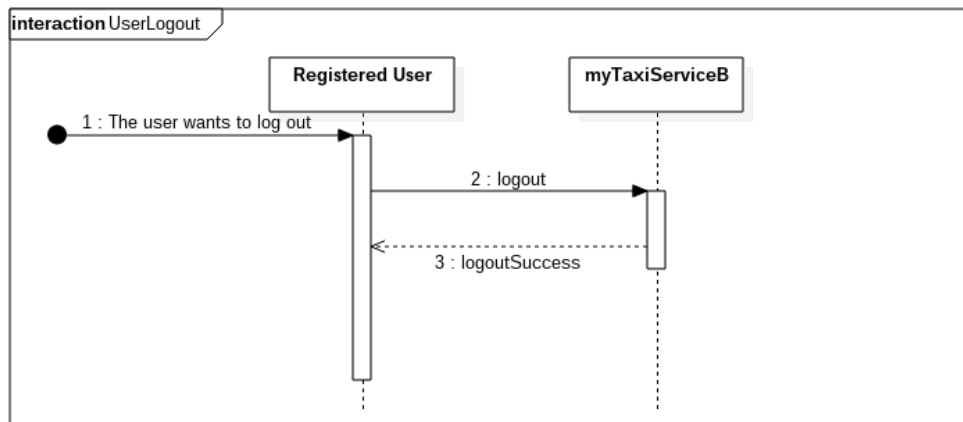**Exit Cnd:** The registered user is acknowledged that MyTaxiService(B) has logged him out from the service

Exceptions

(a) Connection Lost: the registered user is immediately notified of the event and the entire process has to be redone

Flow of Events

(a) The registered user taps/clicks the "LOGOUT" button on a screen of the app/website (every screen of the app/website has the "LOGOUT" button)

Sequence diagram

8. A User wants to register to MyTaxiService

Use case name: User Registration

Actors: A user, myTaxiService(B)

Pre cons: The user has access to the MyTaxiService app or website

Post cons: The user becomes a registered user

Entry Cnd: The user wants to register to MyTaxiService in order to access its functionalities

**Exit Cnd:** The user is acknowledged that MyTaxiService(B) has registered him

Exceptions

    (a) Connection Lost: the user is immediately notified of the event and the entire process has to be redone

    (b) Invalid data(non correctly formatted data or a email already user by another registered user): the user is immediately notified and forced to insert valid data

Flow of Events

    (a) The user opens the MyTaxiService app or access the MyTaxiService website, and a login screen is shown, along with the option to register to the service

    (b) The user taps/click the "REGISTER TO THE SERVICE" button

    (c) The user is redirected to a screen with a form to be filled out with Name, Surname, email a password.

    (d) The user fills the form

    (e) The user press the "CONFIRM BUTTON" and so submits the form to MyTaxiService(B)

Sequence diagram
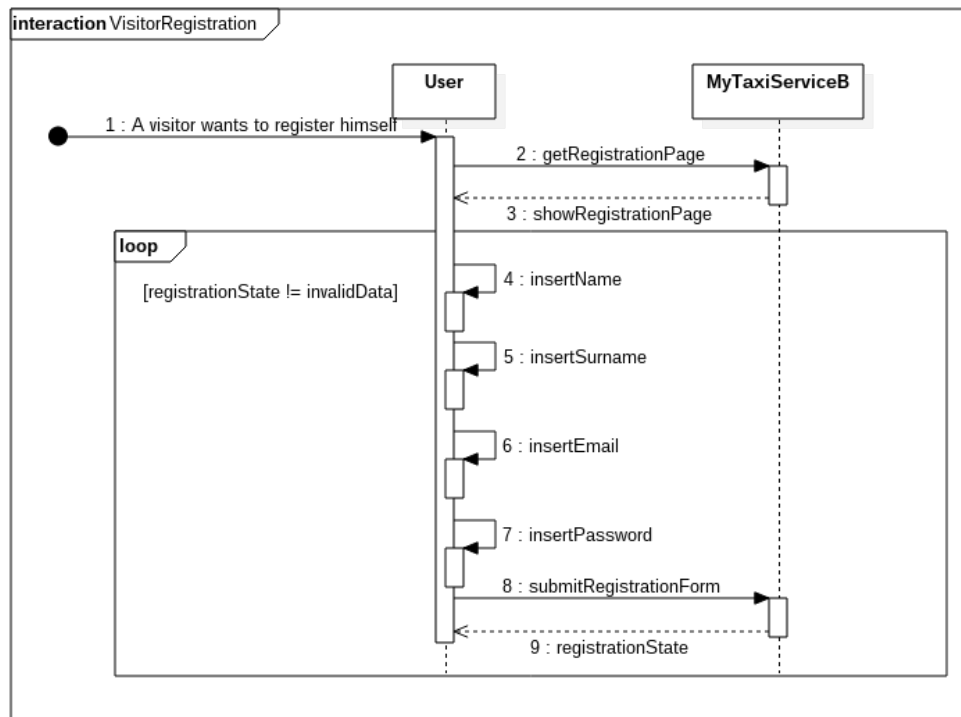
9. A registered user undoes a mtaxi booking

Use case name: Booking undo

Actors: A registered user, MyTaxiService(B)

Pre cons: The registered user is logged into MyTaxiService. The registered user has made a successful booking request. The mtaxi booking the registered user wants to undo specifies a pickup time so that (pickuptime - currentime) ¿ 10 mins

Post cons: The scheduled booking is no more

Entry Cnd: The registered user wants to undo a mtaxi booking

**Exit Cnd:** MyTaxiService(B) acknowledges the registered user that the scheduled booking has been deleted

Exceptions

    (a) Connection Lost: the registered user is immediatly notified of the event and the entire process has to be redone

Flow of Events

(a) The registered user accesses the MyTaxiService app/website area where he can see the list of his bookings

(b) The registered user taps/click the recycle bin icon located in the row that corresponds to the mtaxi booking he wants to undo

(c) The registered user confirm his choice by clicking/tapping the "CONFIRM" button on the popup that has just apparead

Sequence diagram
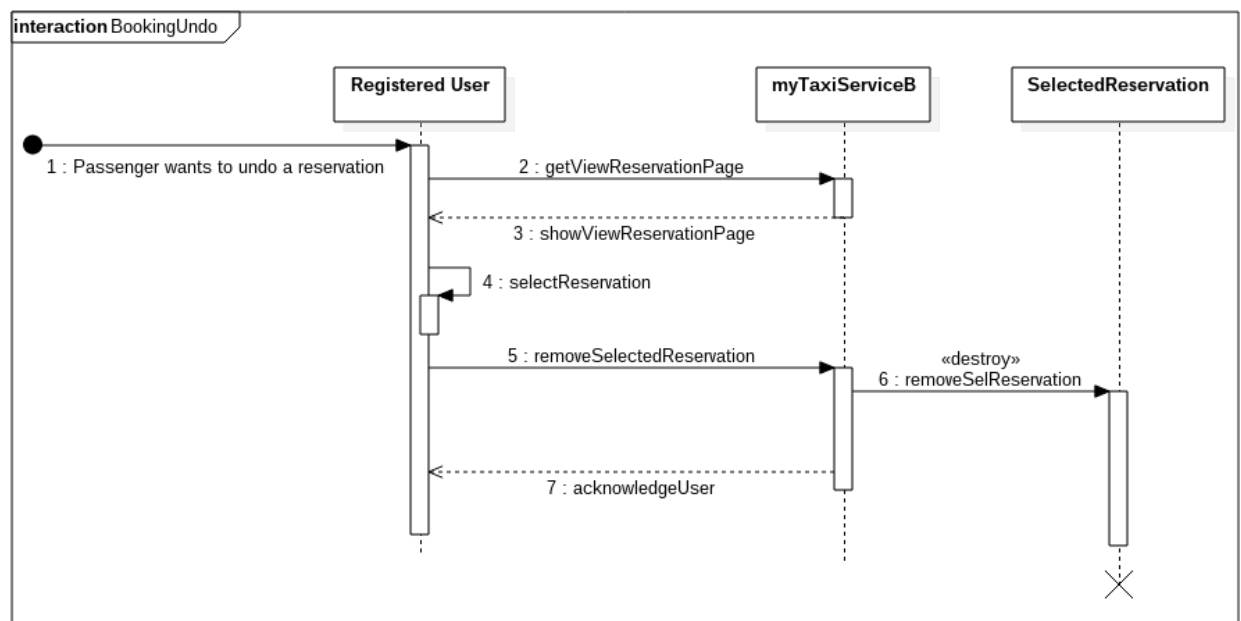
10. A mtaxi driver receives a zone change order

Use case name: Zone change

Actors: A mtaxi driver, MyTaxiService(B)

Pre cons: The mtaxi is available

Post cons: The distribution of mtaxies in the city's zones is more balanced with respect to the average number of ride requests per zone

Entry Cnd: MyTaxiService(B) has detected that some city's zones have too much mtaxies assigned and some others not, all with respect to average number of ride requests per zone. MyTaxiService has assigned the mtaxi in discussion to another city's zone

Exit Cnd: MyTaxiService(B) is acknowledged by the mtaxi driver that he has received successfully the zone change order
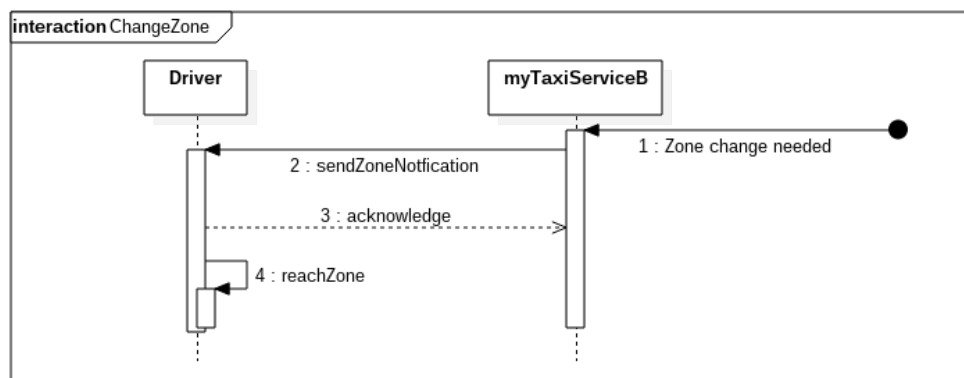
Exceptions

    (a) Connection Lost: the mtaxi driver is immediately notified of the event and the entire process has to be redone

Flow of Events

    (a) MyTaxiService(B) forwards to the mtaxi a change zone order, since the mtaxi has been assigned to another city's zone in order to guarantee a more balanced distribution of mtaxies among all the city's zones

    (b) The mtaxi driver receives the order

Sequence diagram

11. A mtaxi driver reports an accident

Use case name: Accident report

Actors: A mtaxi driver, MyTaxiService(B)

Pre cons: The mtaxi driver has access to and can use MYT. MYT has not been damaged and so is fully functional

Post cons: MyTaxiService(B) considers the mtaxi unavailable and stores the data relative to the accident for statical uses

Entry Cnd: A mtaxi driver had an accident and wants to report it(has to report it)

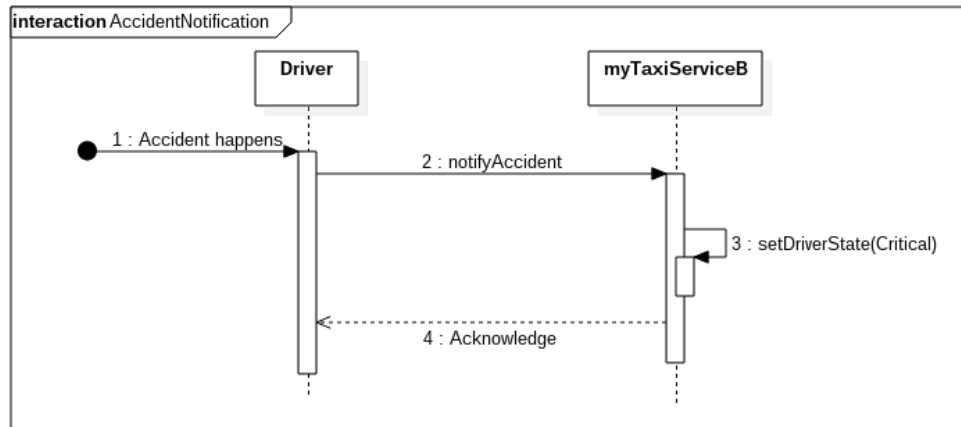**Exit Cnd** MyTaxiService(B) acknowledges the driver about the fact that it has received successfully the accident report

Exceptions

    (a) Connection Lost: the mtaxi driver is immediately notified of the event and the entire process has to be redone

Flow of Events

(a) The mtaxi driver taps the "REPORT ACCIDENT" button on the screen of his MYT

Sequence diagram

12. A taxi driver wants to register to MyTaxiService

Use case name: Driver Registration

Actors : A taxi driver, MyTaxiService(B)

Pre cons : The taxi driver is not yet registered to MyTaxiService

Post cons : The taxi driver becomes a mtaxi driver.

Entry Cnd : A taxi driver wants to register to MyTaxiService

Exit Cnd : The taxi driver is acknowledged that MyTaxiService(B) has registered him. MyTaxiService(B) assigns to the taxi a unique identification number

Exceptions

(a) Connection Lost: the taxi driver is immediately notified of the event

(b) The taxi is not registered as a city's taxi: the taxi driver is immediately notified and forced to change personal and taxi cab information inserted

Flow of Events

(a) The taxi driver access the MyTaxiService website, and a login screen is shown, along with the option to register to the service

(b) The taxi driver click the "REGISTER TO THE SERVICE AS NEW DRIVER" button

(c) The taxi driver is redirected to a screen with a form to be filled out with personal information (name, surname, address, SSN), taxi cab information(brand and model, license plate number) and working timetable information

(d) The taxi driver fills the form

(e) The taxi driver press the "CONFIRM BUTTON" and so submits the form to MyTaxiService(B)

Sequence diagram
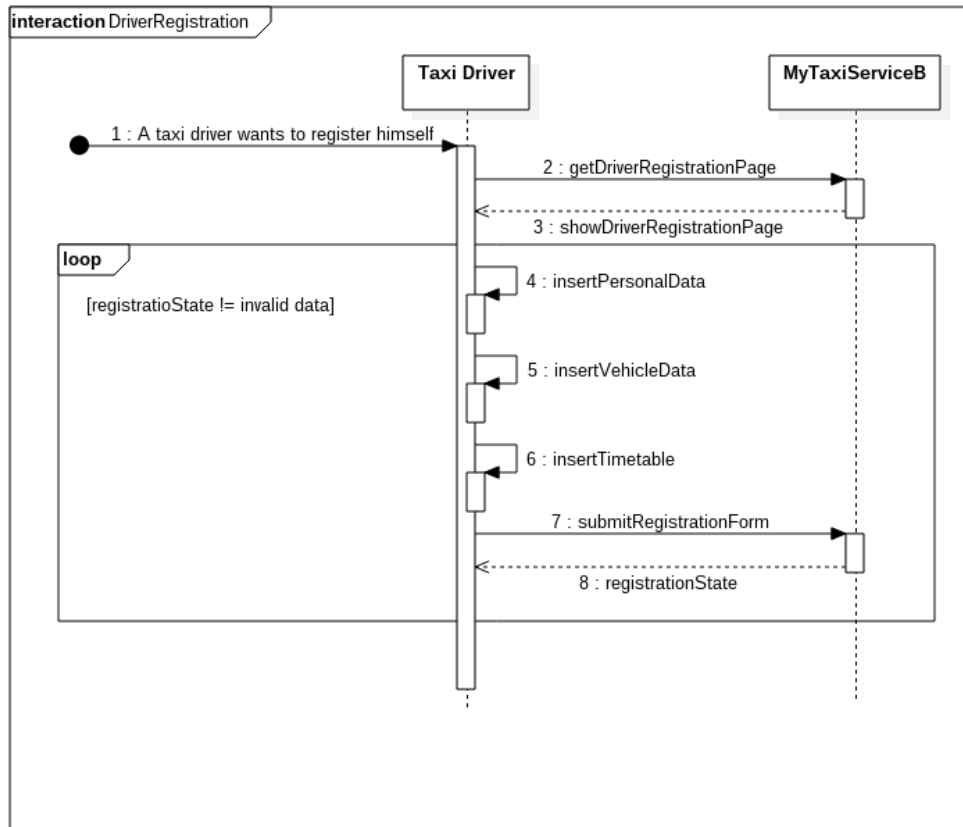
3.3   State chart diagrams
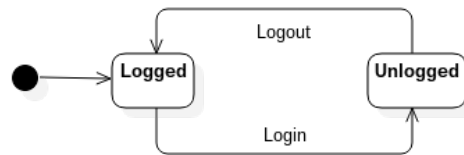
In this section are present some state chart diagrams to illustrate the dynamic behaviour of MyTaxiService.
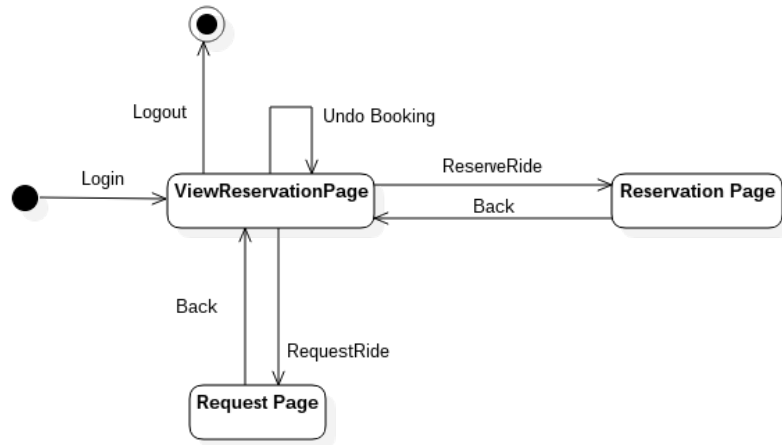
This state chart diagram illustrates the state of a generic Mytaxi driver
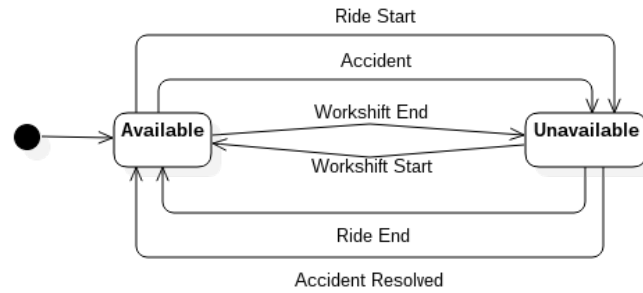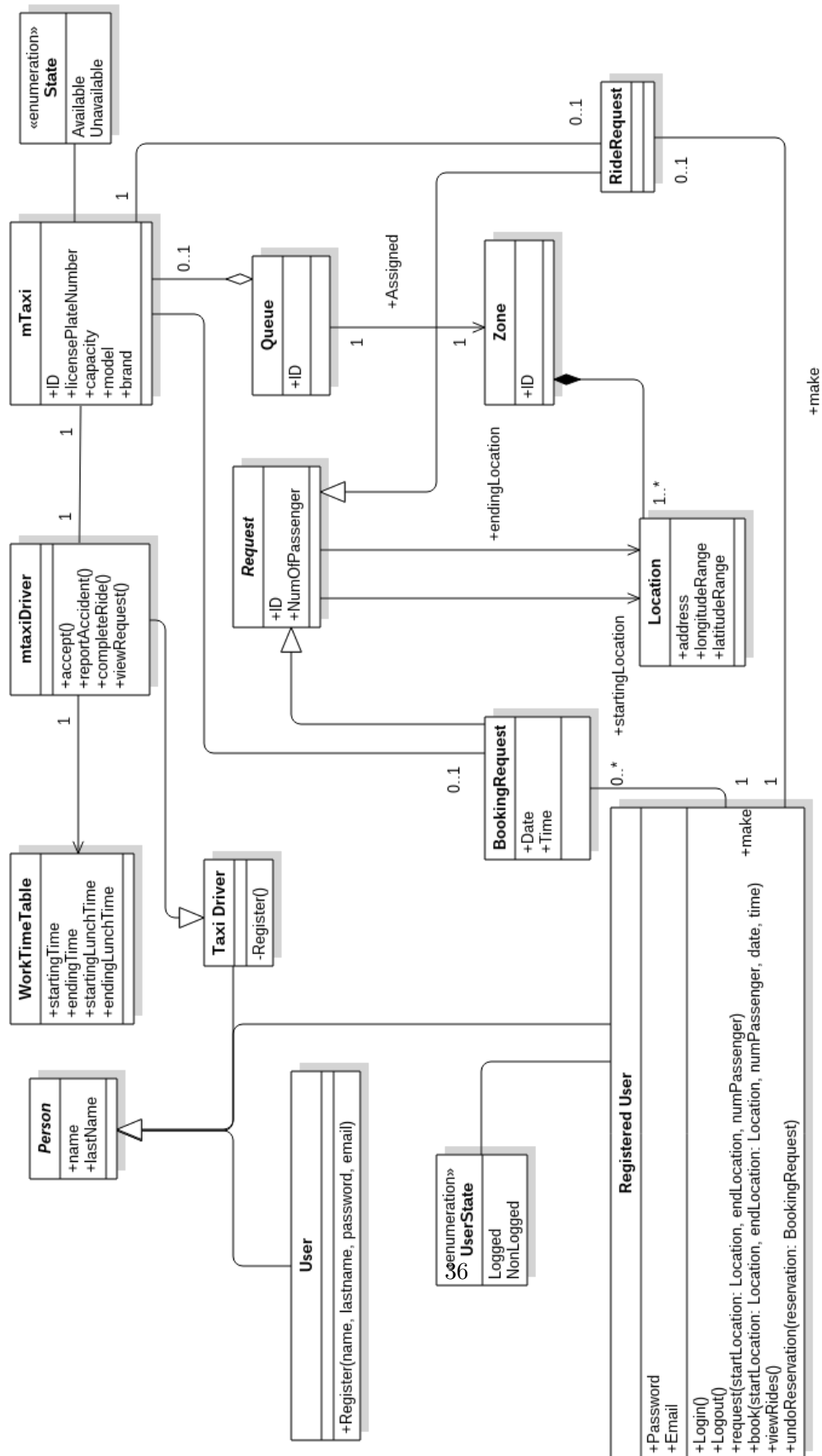

This state chart driver illusrates the state of a generic registered user

This state chart diagram illustrates the behaviour of a generic registered user during the use of MyTaxiService

3.4 Class diagram

4 Alloy model

4.1 Sigs

```
//START ENTITIES

//Models integer values
sig Integer{}
//Models string values
sig Strings{}


abstract sig Person {
    name : one Strings,
    lastName: one Strings
}

sig User extends Person {}

sig RegisteredUser extends Person {
    email : one Strings,
    password: one Strings,
    rideRequest: lone RideRequest,
    bookingRequest: set BookingRequest,
    userState: one UserState
}

abstract sig UserState{}

one sig Logged, NonLogged extends UserState {}

sig Mtaxi {
    id : one Integer,
    licensePlateNumber: one Strings,
    capacity: one Integer,
    model: one Strings,
    brand: one Strings,
    driver: one MtaxiDriver,
    state : one TaxiState
}

abstract sig TaxiState{}

one sig Available, Unavailable extends TaxiState {}

sig TaxiDriver extends Person {}
```

```
sig MtaxiDriver extends TaxiDriver{
    workTimeTable : one WorkTimeTable,
    taxi : one Mtaxi,
}

abstract sig Request{
    id : one Integer,
    numPassengers: one Integer,
    startingLocation : one Location,
    endingLocation: one Location,
    user : one RegisteredUser
}

sig RideRequest extends Request {
    taxi : one Mtaxi,
}

sig BookingRequest extends Request{
    taxi: lone Mtaxi,
    date : one Strings,
    time: one Strings
}

sig Location {
    address : one Strings,
    longitudeRange: one Strings,
    latitudeRange: one Strings
}

sig Zone {
    id: one Strings,
    locations: some Location
}

sig Queue {
    id : one Strings,
    taxies: set Mtaxi,
    zone : one Zone
}

sig WorkTimeTable {
    startingTime: one Strings,
    endingTime: one Strings,
    startingLunchTime: one Strings,
    endingLunchTime: one Strings
}
```

4.2 Facts

```
// D -> Different
//Different requests correspond to different mtaxies
fact DRequestsDMtaxies {
    //Different requests correspond to different mtaxies
    all r1, r2 : RideRequest | r1!=r2  implies  r1.taxi != r2.taxi
    //Different booking requests correspond to different mtaxies if these two different requests are associated to two taxies
    all r1, r2 : BookingRequest | r1!=r2  implies  r1.taxi != r2.taxi
    //Mixed case
    all r1 : RideRequest, r2: BookingRequest | r1.taxi != r2.taxi
}
//A zone is associated to a specif queue and a queue is associated to a specific zone
fact zonesQueues {
    //Every zone is associated with just one queue
    all z : Zone | one q: Queue | q.zone = z
    //Different queues correspond to different zones
    all q1, q2 : Queue |     q1!=q2  implies  q1.zone != q2.zone
}


fact IdsConsistency {
    //Different requests have different request ids
    all r1, r2 : Request |   r1!=r2  implies  r1.id != r2.id
    //Different mtaxies have different mtaxi ids
    all t1, t2 : Mtaxi | t1!=t2  implies  t1.id != t2.id
    //Different queues have different ids
    all q1, q2: Queue | q1 != q2 implies q1.id != q2.id
}
//Different registered users have different emails
fact DUsersDMails {
    all u1, u2 : RegisteredUser |     u1!=u2  implies  u1.email != u2.email
}
//Different requests are associated to different users
fact DRideRequestsDUsers {
    //Ride requests case
    all r1, r2 : RideRequest | r1!=r2  implies  r1.user != r2.user
    //Booking request case
    all r1, r2 : BookingRequest | r1!=r2  implies  r1.user != r2.user
}
//A ride request has to be performed by a logged user
fact RequestLoggedUser {
    all r : RideRequest | (r.user).userState = Logged
}
```

```
//If a registered user is associated with a request than that request is associated with that user
fact TwoWayBindingUserRequests{
    //Ride requests case
    all u : RegisteredUser, r: RideRequest | u.rideRequest=r iff r.user = u
    //Booking requests case
    all u : RegisteredUser, r: BookingRequest | u.bookingRequest = r iff r.user = u
}
//Two different zones aggregate different locations
fact DZonesDLocations {
    all z1, z2: Zone | z1 != z2 implies no l: Location | l in z1.locations and l in z2.locations
}
fact LocationConstency {
    //Two different location can't have the same address
    all l1, l2: Location | l1 != l2 implies l1.address != l2.address
    //Like above but for longitude range
    all l1, l2: Location | l1 != l2 implies l1.longitudeRange != l2.longitudeRange
    //Like above but for latitude range
    all l1, l2: Location | l1 != l2 implies l1.latitudeRange != l2.latitudeRange
}
//Only available mtaxies can fullfil a requests
fact RideRequestAvailableMtaxi {
    //Ride requests case
    all r: RideRequest | (r.taxi).state = Available
    //Booking requests case
    all r: BookingRequest | (r.taxi).state = Available
}
//If a mtaxi is associated with a mtaxi driver than that driver is associated with that mtaxi
fact OneDriverOneTaxi {
    all t: Mtaxi, d: MtaxiDriver | t.driver = d iff d.taxi = t
}
//A request has a diffrent start and ending location
fact DStartingEndingLocations {
    all r: Request | r.startingLocation != r.endingLocation
}
fact WorktimeTableConsistency {
    //All mtaxies are actually supposed to work
    all w: WorkTimeTable | w.startingTime !=  w.endingTime
    //All mtaxies are actually supposed to have time to have lunch
    all w: WorkTimeTable | w.startingLunchTime !=  w.endingLunchTime
}
//A registered user can't book two or more taxies for the same date and time
fact NonUbiquosUsers {
    all u: RegisteredUser | all b1,b2:BookingRequest | b1 != b2 and b2 in u.bookingRequest
        and b1 in u.bookingRequest implies b2.date != b1.date and b2.time != b1.time
}
//A queue aggregates only available mtaxies
fact queuesOfAvailableTaxies {
    all q: Queue | all t: Mtaxi | t in q.taxies and t.state = Available
}
```

4.3 Assertions

```
assert  NoRequestsWithoutUserOrMtaxies {
    //RideRequest case
    all r: RideRequest | one u: RegisteredUser, t: Mtaxi | r.taxi = t and r.user = u
    //BookingRequest case
    all r: BookingRequest | one u: RegisteredUser, t: Mtaxi | r.taxi = t and r.user = u
}
check  NoRequestsWithoutUserOrMtaxies for 5

assert NoUserUbiquos {
    all u: RegisteredUser | no b1,b2 : BookingRequest | b1 != b2 and b1 in u.bookingRequest and b2 in u.bookingRequest
        and b1.date = b2.date and b1.time  = b2.time
}
check NoUserUbiquos for 5

assert NoMtaxiWithoutDriver {
    all t: Mtaxi | one d: MtaxiDriver | t.driver = d
}
check NoMtaxiWithoutDriver for 5

assert RequestsMapOnlyAvailableMtaxies {
    //Ride request case
    all r: RideRequest | (r.taxi).state = Available
    //Booking request case
    all r: BookingRequest | (r.taxi).state = Available
}
check RequestsMapOnlyAvailableMtaxies for 5

//Verify that remove and operation are consistent
assert DelUndoAdd {
    all q,q',q": Queue, t: Mtaxi | addTaxiToQueue[q,q',t] and delTaxiFromQueue[q',q",t] implies q.taxies = q".taxies
}
check DelUndoAdd for 5
```

4.4   Predicates

```
//Simulates the add of a taxi to a queue
pred addTaxiToQueue(q,q': Queue, t: Mtaxi) {
    q'.taxies = q.taxies + t
}
run  addTaxiToQueue for 5

//Simulates the deletion of a taxi from a queue
pred delTaxiFromQueue(q,q': Queue, t: Mtaxi) {
    q'.taxies = q.taxies - t
}
run  addTaxiToQueue for 5

//General world generation
pred show {
    #RegisteredUser = 2
    #RideRequest = 1
    #BookingRequest = 1
    #Mtaxi > 1
}
run show for 5

//Show a world which enlights ride/booking request properties
pred RideBookingRequestProperties {
    #Mtaxi = 2
    #MtaxiDriver = 2
    all d: Mtaxi | d.state = Available
    #RideRequest = 1
    #BookingRequest = 1
}
run RideBookingRequestProperties for 5
```

4.5 Results

In this section is shown the results produced by Alloy Analazyer

9 commands were executed. The results are:
- **#1:** No counterexample found. NoRequestsWithoutUserOrMtaxies may be valid.
- **#2:** No counterexample found. NoUserUbiquos may be valid.
- **#3:** No counterexample found. NoMtaxiWithoutDriver may be valid.
- **#4:** No counterexample found. RequestsMapOnlyAvailableMtaxies may be valid.
- **#5:** No counterexample found. DelUndoAdd may be valid.
- **#6:** Instance found. addTaxiToQueue is consistent.
- **#7:** Instance found. addTaxiToQueue is consistent.
- **#8:** Instance found. show is consistent.
- **#9:** Instance found. RideBookingRequestProperties is consistent.

This graph shows the model produced by the execution of the predicate show

44
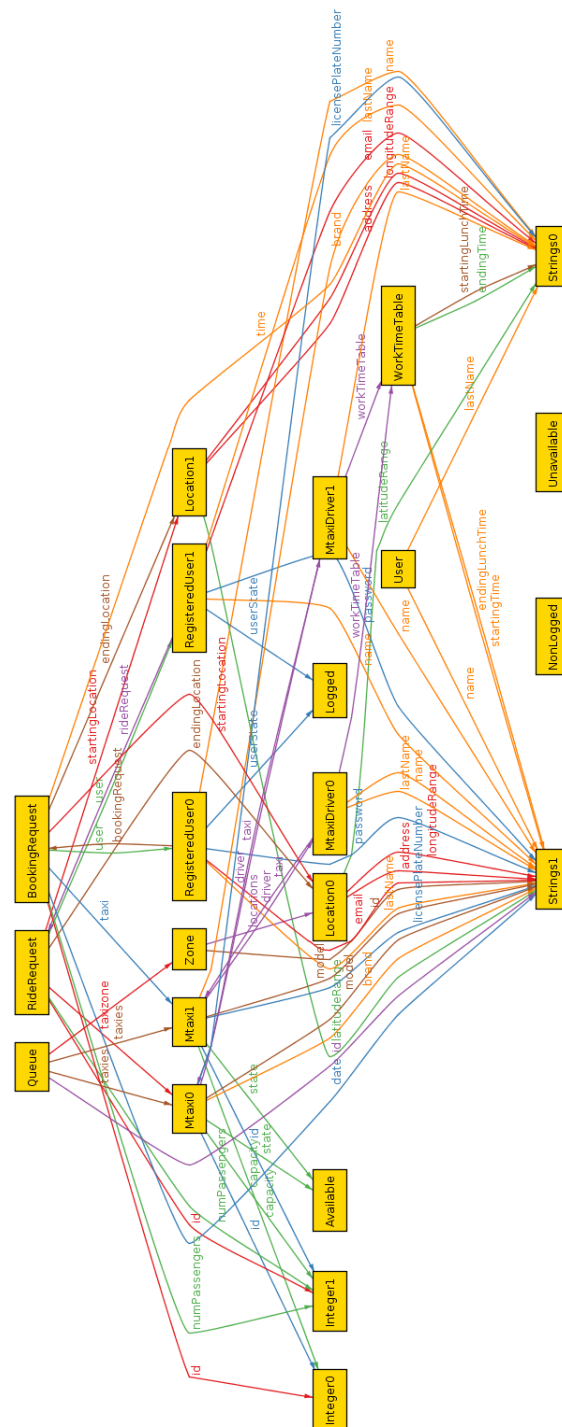
This graph shows the model produced by the execution of the predicate
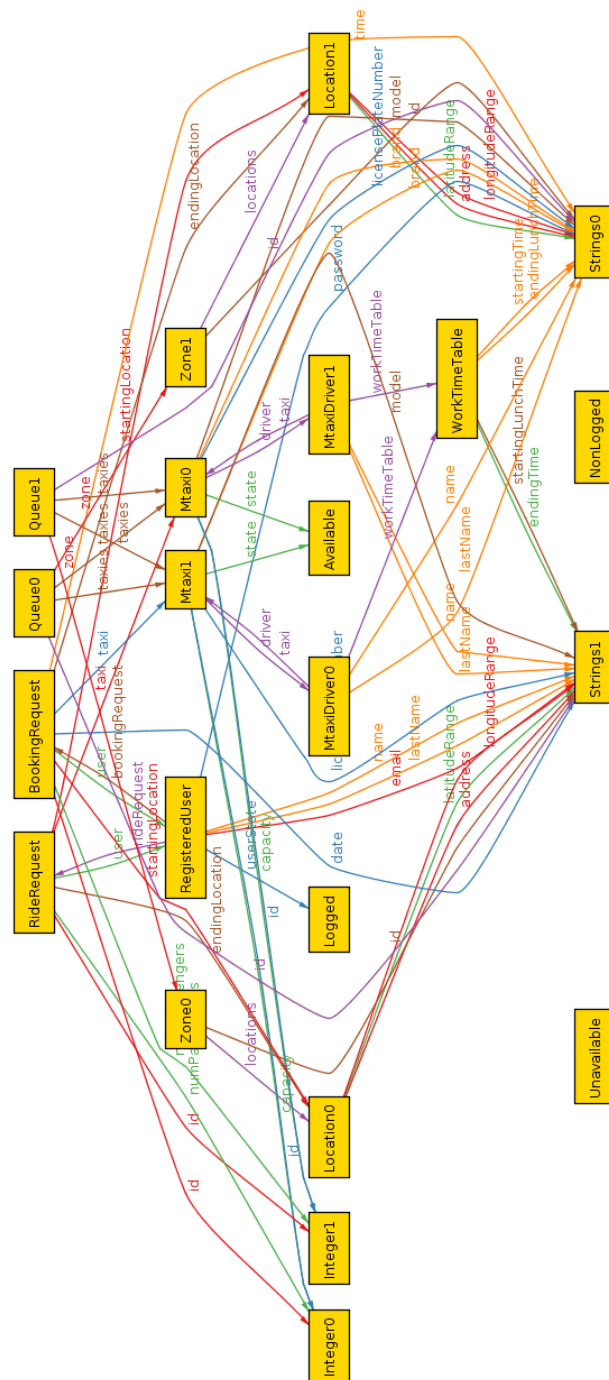
5   Appendix

5.1   Tools

- Latex/Atom: to redact and produce this document

- Git: to manage the collaboration of the team members

- Start UML: to produce the UML diagrams attached to this document

- Alloy Analyzer 4.2: to prove the consistency of the model defined in this document

5.2   Hours of work

- Giorgio Pea: 34 hours

- Andrea Sessa: 36 hours