POLITECNICO DI MILANO
Computer Science and Engineering
Project of Software Engineering 2

myTaxiService

Requirements Analysis and Specification Document

Ver. 1.1

Authors: Simone Rosmini (853949)
Vincenzo Viscusi (858689)
Matteo Zambelli (776162)

Reference Professor: Mirandola Raffaela

INDEX

CHANGE HISTORY:

v. 1.1: small changes to UML, the world and machine diagram and other grammar corrections.

1. INTRODUCTION

This document is a Requirement Analysis and Specification Document (RASD) and his main goal is to provide a specification of the requirements that must be satisfied by the system.
This document presents a description of the functional and non-functional requirements, using various UML diagrams and a high level specification. There is a part where it is instead presented a formal model of specifications and the results of its analysis, made by Alloy.
This document is addressed to myTaxiService project developers and all other stakeholders of the project. Regarding the firsts, the document becomes important when you need to establish a coherence between the development and the requests processed. For stakeholders, this document is a trace of the analytical work done by developers, useful for interpreting their development work.

1.1 DESCRIPTION OF THE GIVEN PROBLEM

The problem is the one to try to develop a taxi service which is more efficient than the actual system by using a mobile application or a web application.
Using the new taxi services the user should be able to call a taxi to start a ride from his actual position or make a reservation of a ride.
In order to save money, users can choose to make a reservation for a share ride without extend the duration of the ride more than they really want by inserting the maximum delay time they can accept. In this way the system joins only the compatible rides allowing the users to arrive where they want saving money and never being late.

1.2 APPLICATION SCOPE

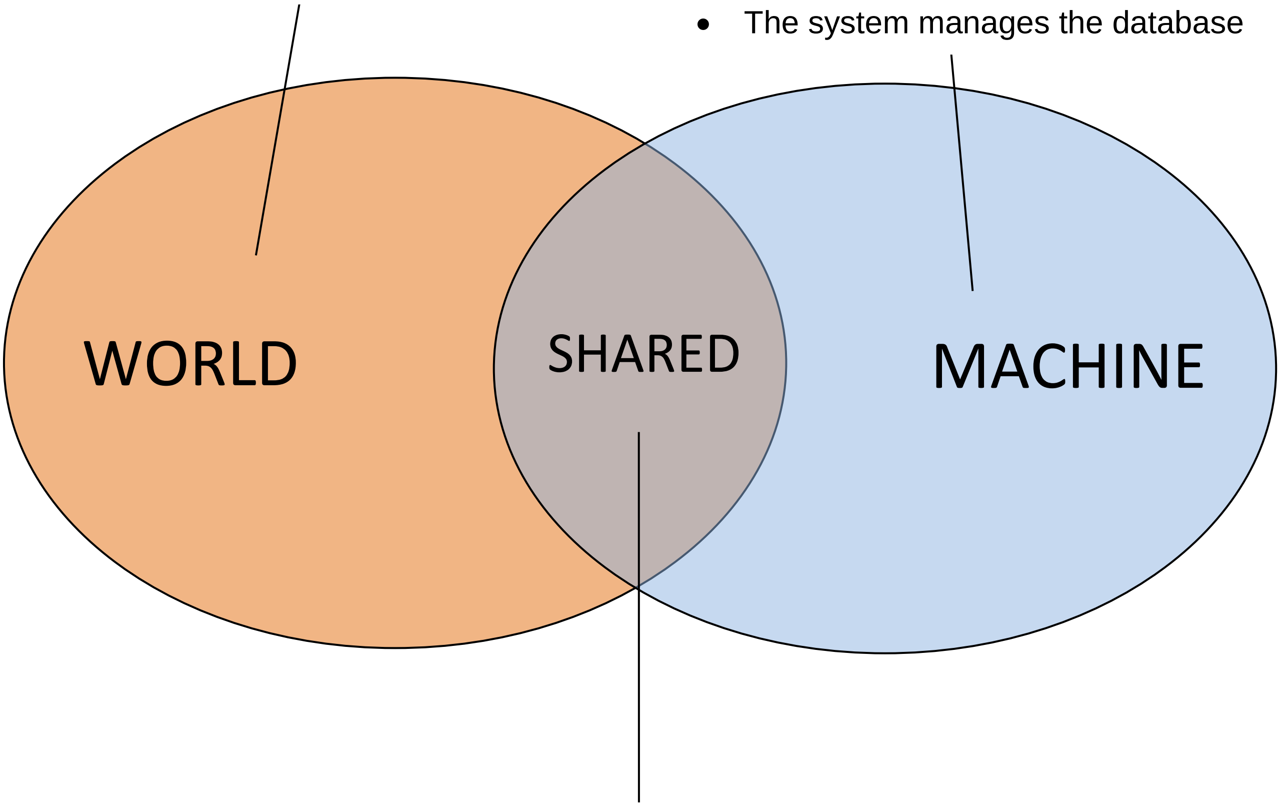
The main scope of the product is allow people to call taxies more easily and guarantee a fair management of taxi queues. User, once registered, should be able to request a taxi make a reservation, in a single mode or in a shared mode, or modify an already done ones.
On the taxi driver's view he can communicate at the system when is ready for a taxi ride, when is busy or when he is not ready for other problems, if he is ready and receive a taxi call from a user he can decide at the moment if accept or not the call with the application.

1.3 WORLD AND MACHINE

The following diagram represents the Jackson & Zave contest for the application that has to been developed, it underlines the most relevant events that can occur in the application domain.

- The user gets on the taxi
- The user gets off the taxi
- The user arrives at the appointme address

- The system computes the routes and the taxis travel times
- The system manages the taxi queues
- The system allocates a taxi for a call
- The system manages the database



- A user subscribes or access the application
- A taxi driver logs into the application
- A user calls a taxi
- A taxi takes on a user
- A taxi leaves a user at destination
- A taxi driver accepts or refuses a request
- A taxi performs a ride
- A taxi driver sets the own availability
- The system sends the requests to the taxis

1.4 GOALS

-G1: Allow the users to request a real time ride.
-G2: Allow the user to reserve a ride.
-G3: Reserve a ride shared among more users.
-G4: Allow the user to sign up for the taxi service.
-G5: Allow the login of a subscribed user.
-G6: Keep constant, as much as possible, the number of taxi for each zone.
-G7: Allocate a taxi for a ride.
-G8: Provide to the taxi administrator some statistics to choose the number of taxi for each zone.
-G9: Allow the user to create a calendar with the scheduled rides.

1.5 DOMAIN PROPERTIES

- A Shared call can be joined in the same ride only with other shared call.
- If a ride is serving a single call then it isn't serving other calls.
- If a ride is serving a real time call then it isn't serving other calls.
- Each call is associated to a single ride.
- A user can be a passenger of a taxi only if the taxi is serving a call made by the user.
- All the shared calls associated to the same ride have the same starting zone.
- A shared ride must stop to each start address and end address specified by the call that it is serving.
- Each taxi is associated to a single queue according with its position.
- Each ride is associated to a single taxi.
- Each taxi can serve only one ride at the same time.
- Two calls are shared in the same ride only if the new estimated arrival time for each call doesn't exceed the maximum delay accepted from the corresponding user.
- Two calls are shared in the same ride only if they start from the same zone.
- A taxi receives a request for a ride only if its state is free
- A reservation is accepted only if there is a time interval of at least 15 minutes among the new reservation and other scheduled reservations of the user (both before and after the reservation)
- A taxi driver can serve a ride only if he accepts the request within 30 seconds from its arrival.

1.6 ASSUMPTIONS

- We assume that the taxies don't bump into any kind of problem during the ride.

- We assume that the taxi driver manages the fuel by himself and that he sets his status to unavailable when he has to fill up the tank.

- We assume that the users cannot modify the destination address during a ride

- We assume that when a taxi accepts a ride than he starts immediately to serve the ride following, without unnecessary deviations or stops, the route received by the system.

- We assume that the gps positions of the taxis are known in an accurate way.

- We assume that the users pay with a credit card or paypal when a taxi ride is confirmed and so we don't need to introduce any type of forfeit.

1.7 GLOSSARY

Here there are explained the meaning of the terms used in this RASD :

CALL: Each request done by a user for a ride (real time, shared or single)

ROUTE: is the path between the source and the destination

TAXI RIDE: represents the service done by the taxis to accommodate one or more user calls, to each taxi ride is associated a route that will be followed by the taxi.

TAXI ALLOCATION: It's a phase that starts when the system sends a request to the taxis in a queue and ends when the taxi that has accepted the ride receives the route to follow to serve the ride. To do that the system chooses the right taxi queue and forward the ride request from the first to the last taxi in the queue, until someone accepts the request.

TAXI DRIVER STATE: a taxi driver with the application can set its status on three modes:

1. Available : when he is waiting to accept a call
2. Busy : when he is driving for a call and can't serve another one
3. Not available : when for personal reasons he can't serve a call

1.8 PROPOSED SYSTEM

We suppose that the actual system that provides the service is the traditional call to the

Taxi centerline. Now we are developing a new service that will substitute the actual system and it will be based on web app service.
The system that we will implement is composed by a central server that works 24hperday and communicates with clients from a web application (on the website or on a smartphone), the information about the traffic are taken from Google Maps that also provide the GPS information about the taxi and the virtual space where we store and compute the information about the taxi's ride.

1.9 IDENTIFY STAKEHOLDERS

Our main stakeholder is the professor who gave us the delivery of the project, the person who will use the system when it will be delivered on the market (user, guest and taxi driver) and Google because give us the main information about the street and provide a system for analyze and compute the GPS information about the taxi.

2. ACTORS IDENTYFING

The Actors of our system are three:

GUEST: a guest is a person who hasn't signed up yet. Guests have less power in the system than users, they don't have the users' skills, and the only function they can use is the sign up.

USER : is someone who has already signed up for the taxi service, he can see and request the main features of the system like CALL a taxi ride for the same moment or RESERVE a taxi ride for another day (shared or not). A user has a profile that includes the following information: Name, Surname, Email, Username, Password, Payment Card and optionally a Picture.

TAXI DRIVER: a taxi driver is a person that works for the taxi company and drives the taxi during his worktime. A taxi driver has the capability of accept or decline the calls.

ADMIN: the admin is the person who controls the correct execution of the service, he can add new taxi or new taxi drivers and he can see all the actual information about the zones. In particular he can set up the preferred number of taxis for each zone and see the statistics reported by the system about the request rate of each zone.

3. REQUIREMENTS

3.1 FUNCTIONAL REQUIREMENTS

-G1: Allow the users to request a real time ride

- R1: The system should require the user to fill in the following fields:
  - Meeting address
  - Destination address
- R2: The system should allocate a taxi for the request
- R3: The system should create a path for the ride and send it to the taxi
- R4: The system should send to the user the code of the taxi that has taken on the ride and the waiting time.

-G2: Allow the user to reserve a ride

- R1: The system, to create a new reservation, should require the user to fill in the following fields:
  - Meeting Time
  - Meeting address
  - Destination Address
  - Travel mode
  - Repetition (No/Daily/Weekly/Monthly)
- R2: The system should compute the duration of a ride and the expected arrival time using an external map service.
- R3: The system should accept the reservation verifying that it's not in conflict with other scheduled reservations of the user.
- R4: The details of the new reservation are stored only if the reservation is accepted.
- R5: The system 10 minutes before de meeting time mast allocate a taxi for the reservation.

-G3: Reserve a ride shared among more users:

- R1: The system should ask to each user that wants to share a ride, the maximum delay on the estimate arrival time that he wants to accept for sharing the ride.
- R2: The system mast be able to join different paths to create a single route that passes through all the start and end destination addresses of each call request.

- G4: Allow the user to sign up for the taxi service

- R1: The system should require the user to fill in the following fields:
  - E-mail with confirmation
  - Password with confirmation
  - Surname and Name
  - Preferred payment way(credit card or paypal)
- R2: The system should require the user to insert a list of the zone from he often takes a taxi in order to support the statistic elaborations.
- R3: The system uses the email as the user identifier

- R4: The subscription is accepted only if all the fields have been fulfilled and the email doesn’t already exists in the database of the registered users
- R5: The system stores all the user information only if the subscription is accepted
- R6: The system should send a welcome email to the user with all the necessary information to use the service.

- G5: Allow the login of a subscribed user

- R1: The system must require the user to fill in the following fields:
    - E-mail used in the subscription phase
    - Password used in the subscription phase
- R2: The login is accepted only if the password inserted corresponds with the email in the database of the registered users.
- R3: The user can access the service only if the login is accepted.

- G9: Keep constant, as much as possible, the number of taxi for each zone (world):

- R1: Every time a taxi passes from a zone A to a zone B then a taxi in the zone B moves towards the zone A.

-G3: Allocate a taxi for a ride

- R1: the system should send the details of a ride request to the first taxi in the queue of the zone that contains the start address of the ride:
    - Tracking number of the ride.
    - Path of the ride.
    - Path to reach the start address
    - Call type (Real Time or Reservation).
    - Travel Mode (Single or Shared).
- R2: The system should pass the ride request to the next taxi in the queue only if the taxi driver has refused the request or it passes more than 30 seconds from the arrival of the request.
- R3: If the taxi driver refuses the request or he takes more than 30 seconds to accept it, the system should remove his taxi from the head and place it in the tail of the queue.
- R4: If no taxis are available in the zone the system should pass the request to the first taxi in the queue of a near zone. From the various near zones the system should choose the one in which the position of the first taxi in the corresponding queue is as near as possible to the call start address.

-G6: Provide to the taxi administrator some statistics to choose the number of taxi for each zone:

- R1: The system should calculate the average number of ride request per hour for each zone (request rate).
- R2: The system should be able to show to the taxi administrator a table with the following information:
    - Actual number of taxi for each zone
    - Priority of the zones
    - Request rate for each zone.

-G8: Allow the user to create a calendar with the scheduled rides

- R1: The system should display to the user a calendar with the scheduled reservations.
- R2: The system should ask the user to select the day in which he wants to schedule a new reservation or see the details of once already scheduled
- R3: The system, to create a new reservation, should require the user to fill in the following fields:
    - Meeting Time
    - Meeting address
    - Destination Address
    - Travel mode
    - Repetition (No/Daily/Weekly/Monthly)
- R4: The system should compute the duration of a ride and the expected arrival time
- R5: The details of the new reservation are stored only if the reservation is accepted.

3.2 NON-FUNCTIONAL REQUIREMENTS

- The service mast be available 24 hours per day

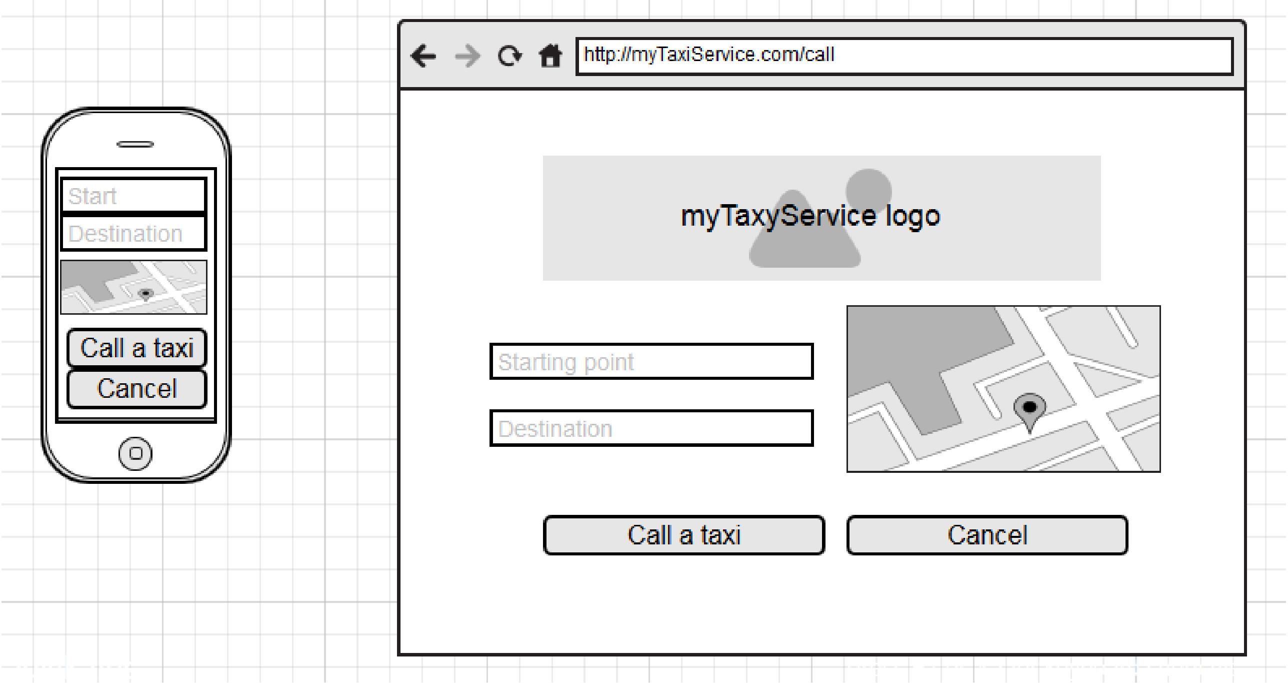
- The system mast chose always the fastest path for the moving of the taxies

- The system mast be available through web application and mobile application

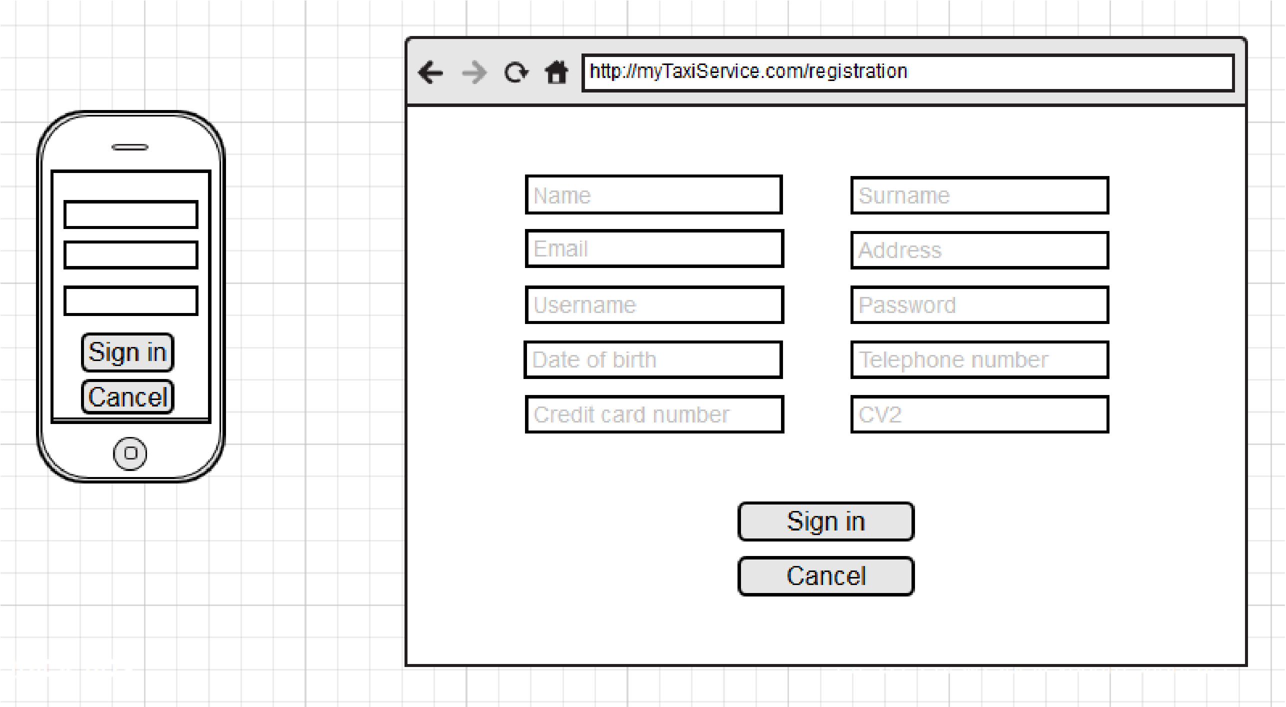
3.2.1 User Interface

Here are presented some mockup that represent an idea of the structure of the application pages:

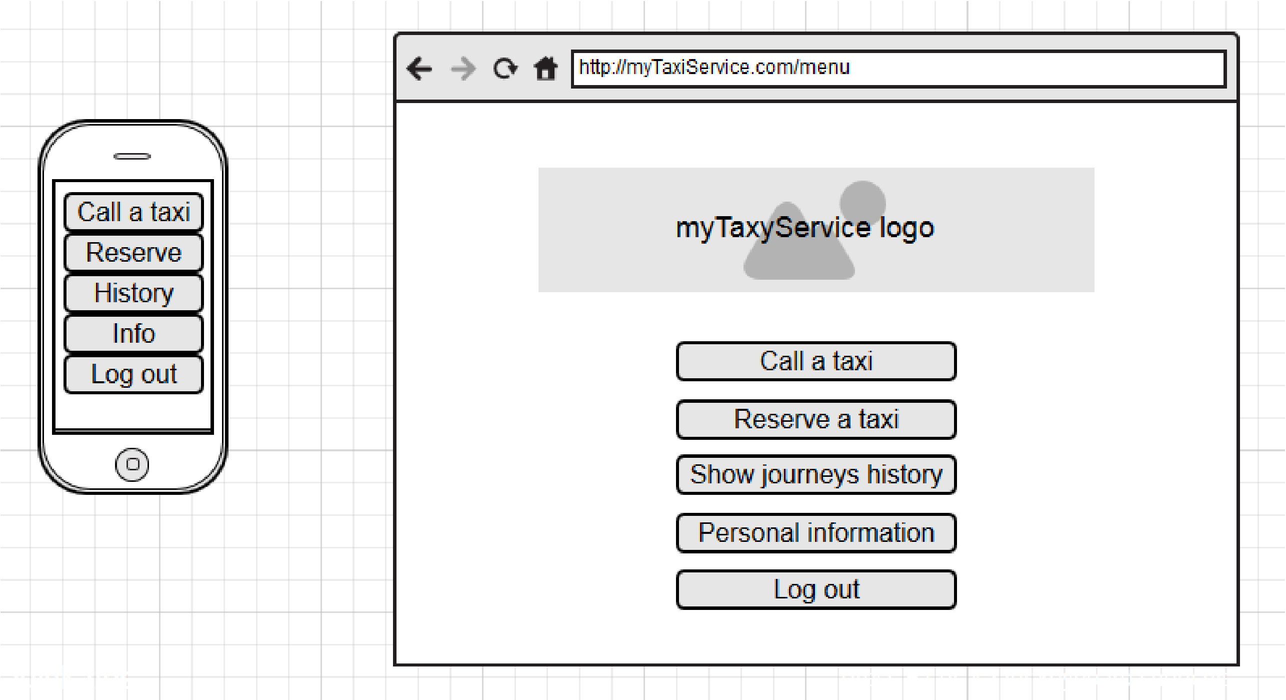
-Login



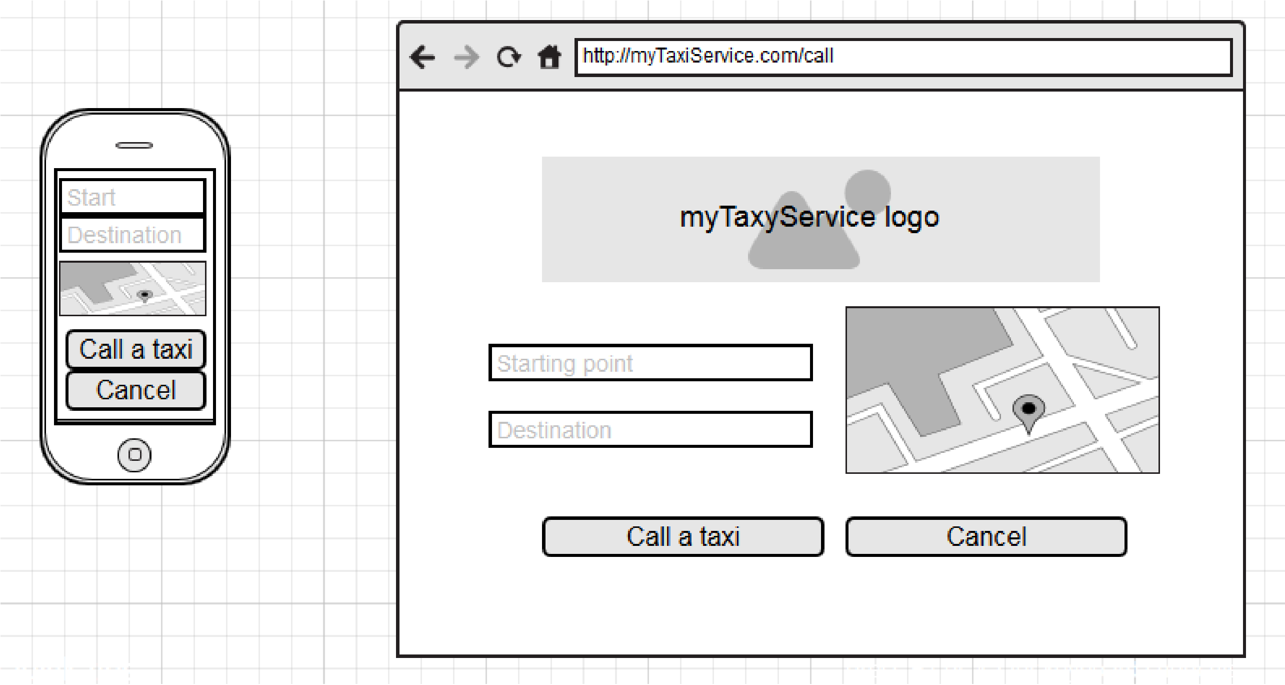
-Registration



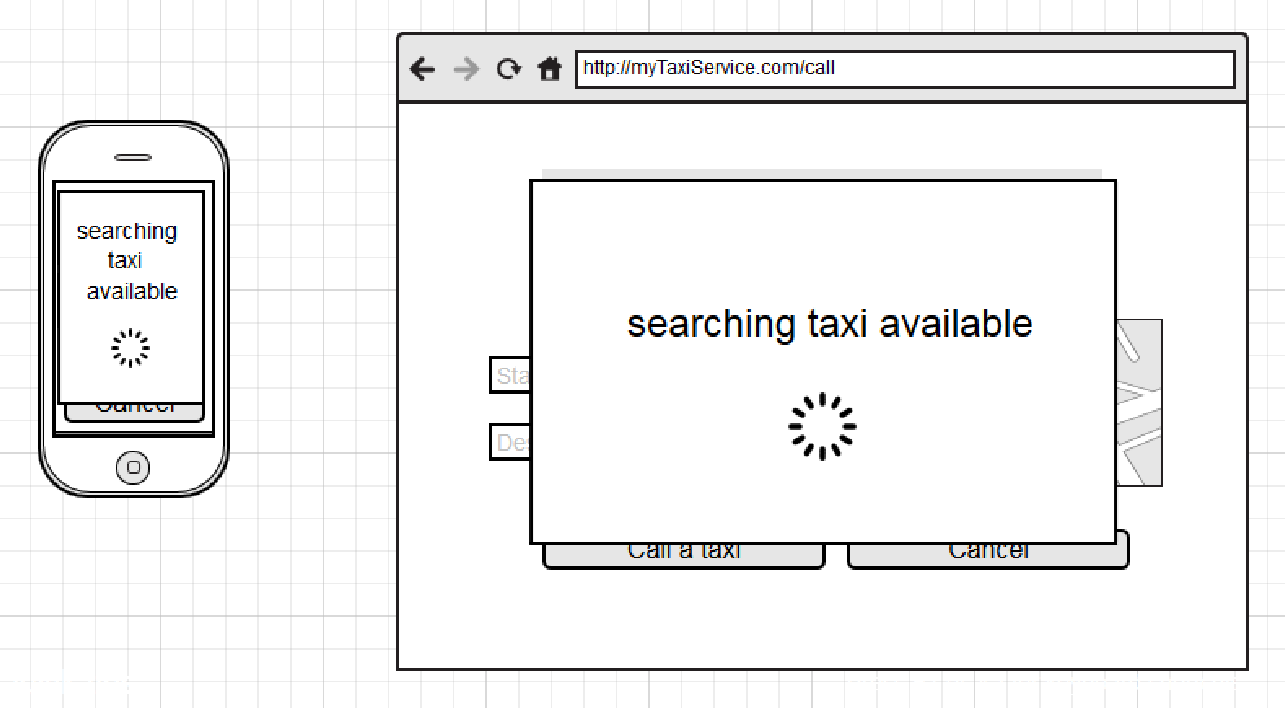
-Menu



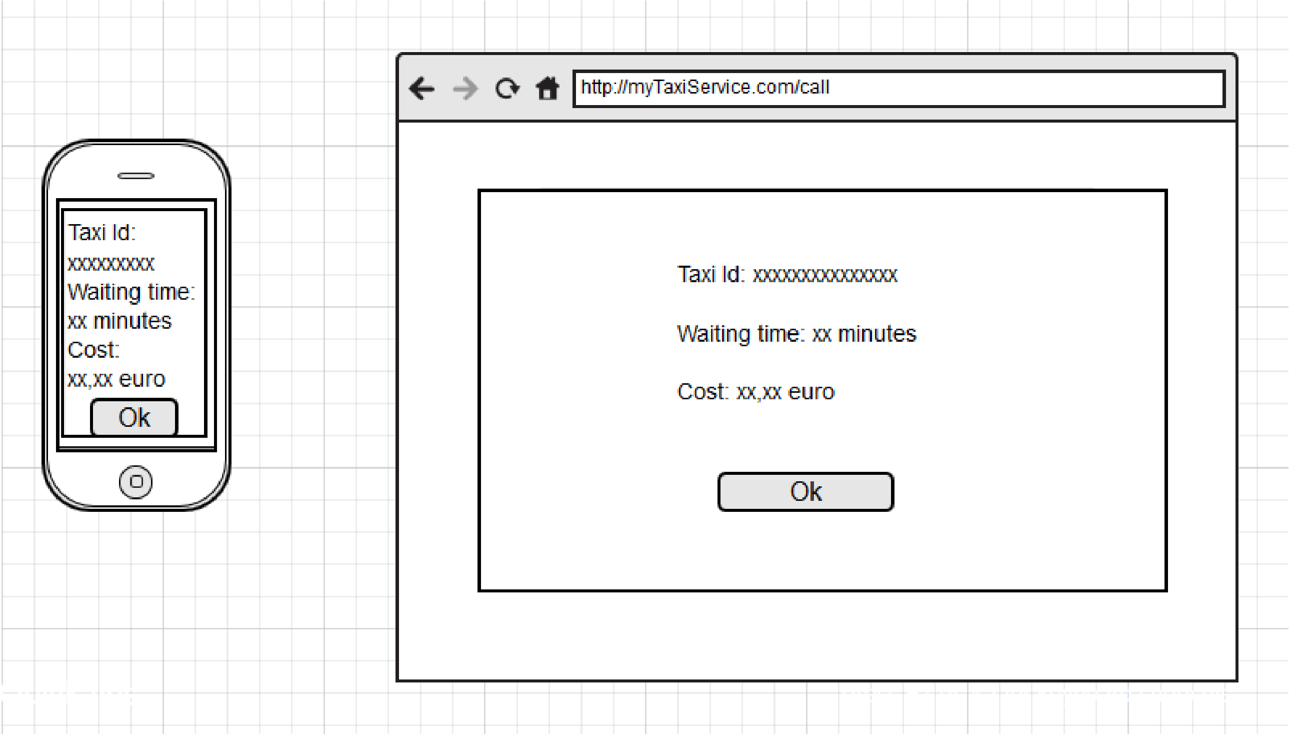
-Calling of a taxi



-Waiting



-Confirm call



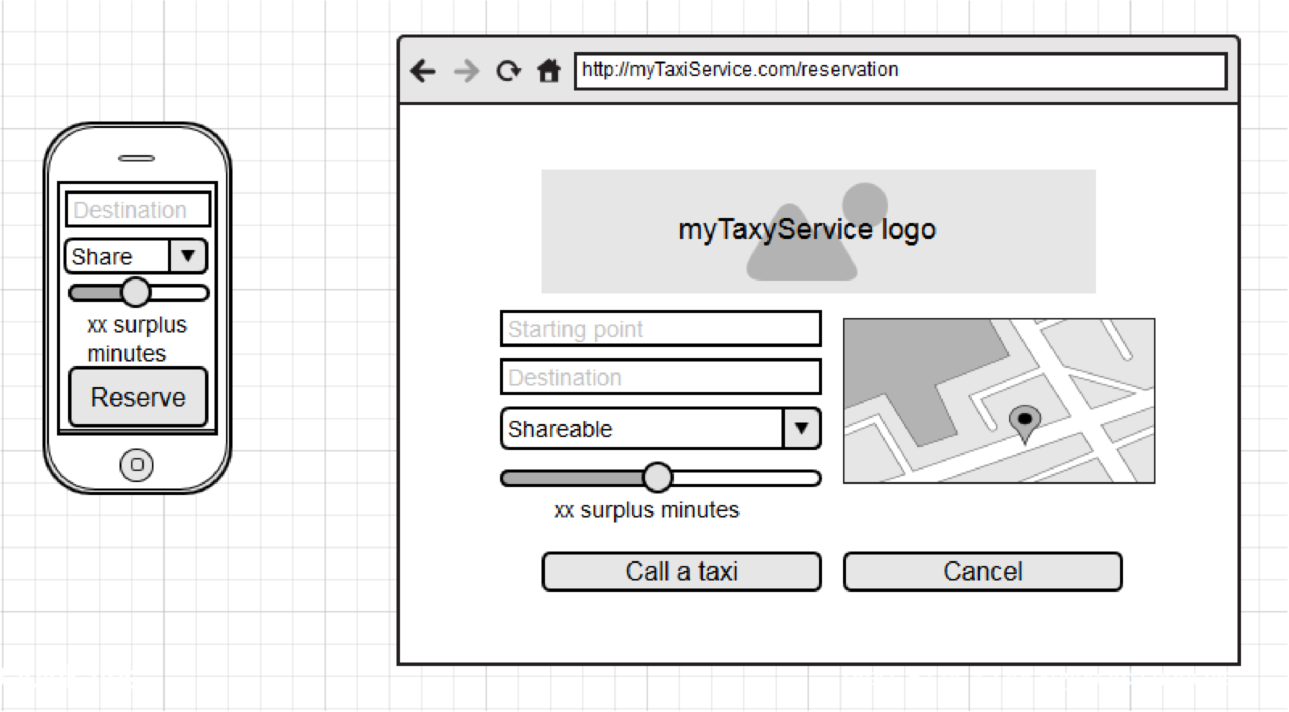
http://myTaxiService.com/call

Taxi Id: xxxxxxxxxxxxxxx

Waiting time: xx minutes

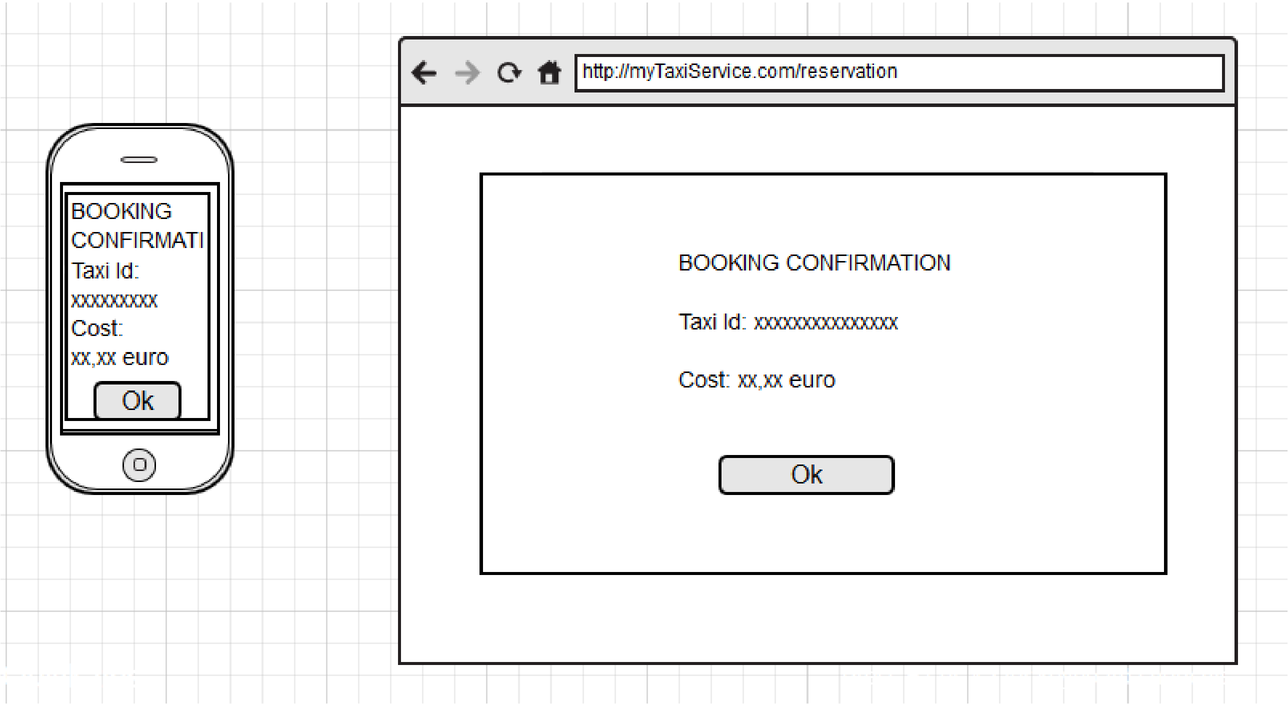
Cost: xx,xx euro

-Reservation of a taxi



http://myTaxiService.com/reservation

-reservation confirm



BOOKING CONFIRMATI

Taxi Id:

xxxxxxxxx

Cost:

xx,xx euro

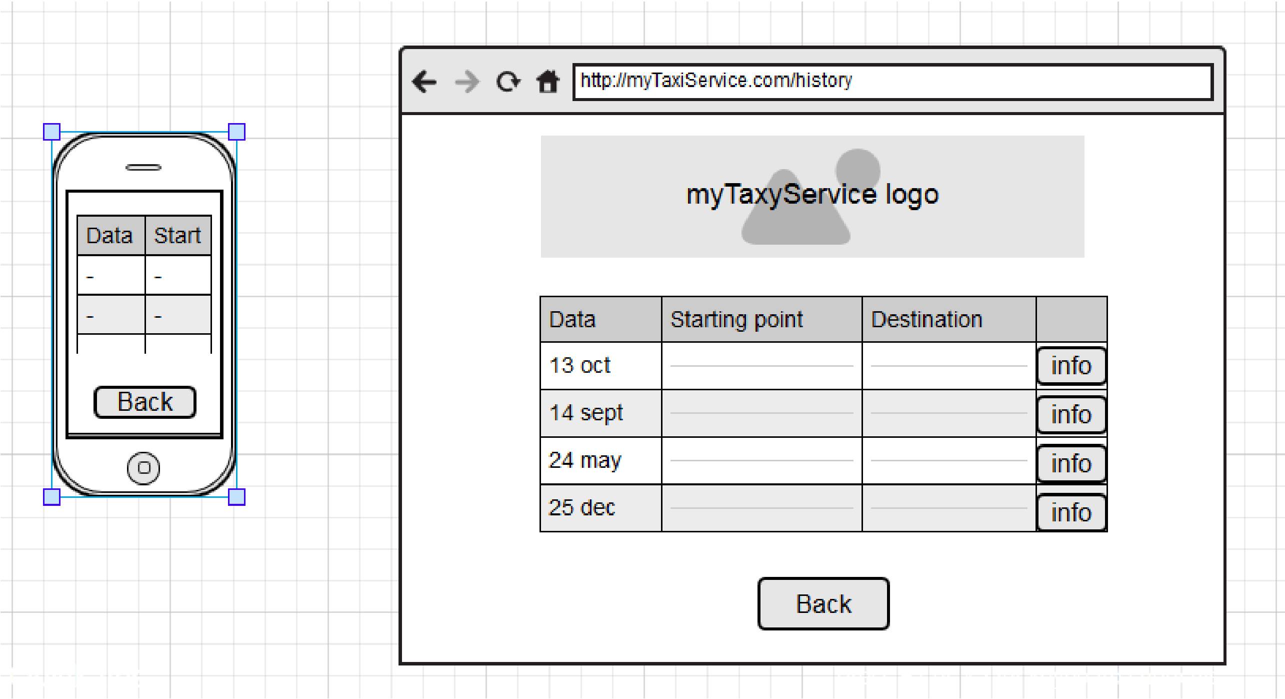
http://myTaxiService.com/reservation

BOOKING CONFIRMATION

Taxi Id: xxxxxxxxxxxxxxx

Cost: xx,xx euro

-Journeys history

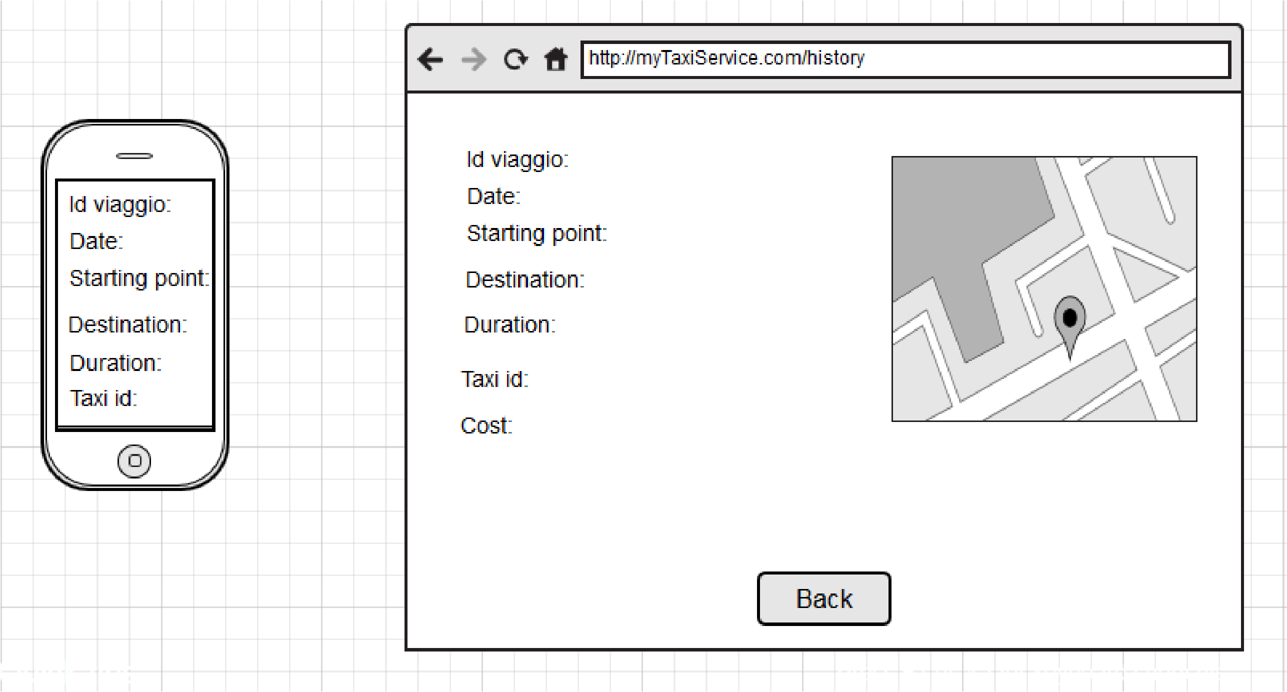


Data	Start
-	-
-	-

http://myTaxiService.com/history

Data	Starting point	Destination	
13 oct			info
14 sept			info
24 may			info
25 dec			info

-Information of a journey



4. SCENARIO IDENTIFYING

Scenario 1

Simone as every day is doing his job as a taxi driver for myTaxiService. He is on duty for two hours but he has not received calls yet.

Around noon the integrated system in the car displays a request of a customer. He accept and heads towards the position of the passenger.

As soon as Vincenzo has settled in the car, Simone starts moving toward the destination.

When Vincenzo leaves the car, Simone informs the system and becomes available.

Scenario 2

Matteo as every year decides to visit his grandparents in Milan during the Christmas holidays.

Knowing that their house is not accessible by public transport, this year Matteo decided to sign up on myTaxiService, a service that allows to call and book a taxi with a simple mobile app.

He, arrived to the central station, logs in with the app and immediately calls a taxi to take him to the grandparents' address.

He waits for a few seconds until the system tells him the taxi MIT278 will come to pick him up after 5 minutes.

A few minutes later therefore Matteo takes the taxi of Pasquale that will arrive directly in front of his grandparents' house.

After thanking Pasquale, Matteo descends from the taxi and heads for home.

Scenario 3

Luca must participate in a working meeting to be held at a hotel across town in the afternoon. He decides to book a taxi with the app myTaxiService. He completes the reservation form with the time and place of the meeting and to spend less decides to sharing the journey with others by specifying with the appropriate bar an extension of time up to 15 minutes.

10 minutes before the meeting, Luca receives the identifier of the taxi that is coming. After a few minutes after departure, the taxi of Massimo stops to pick up Elisa. During the journey the two passengers talking to each other and Elisa tells Luca that is heading to a clinic for a medical examination.

Reached her destination, Elisa greets Luca, thanks Massimo and gets out of the taxi, Luca instead continues until his destination.

Once arrived, he thanks Massimo and heads for the door of the hotel.

Scenario 4

Gianluigi is a taxi driver of myTaxiService and he is in the stadium area. He just brought a passenger to the destination and now is in a parking to rest since it has a big headache.

Not even 10 minutes from when he turned off the car, he gets a call request from the system. The passenger should be brought into the area Lambrate which is across town.

He thinks for a few seconds before putting safety before the money, because he does not feel in a position to do that distance at the moment.

Gianluigi then refuses the request, knowing that he will be sent to the back of the queue.

Gianluigi puts himself as " not available " to try to recover.

Scenario 5

Adalberto wants to go see an exhibition in the Navigli area in the afternoon. Not being a lover of public transport such as metro and tram, he decides to use myTaxiService that he has already used another time remaining satisfied.

He logs in with the app with the smartphone and inserts the travel information.

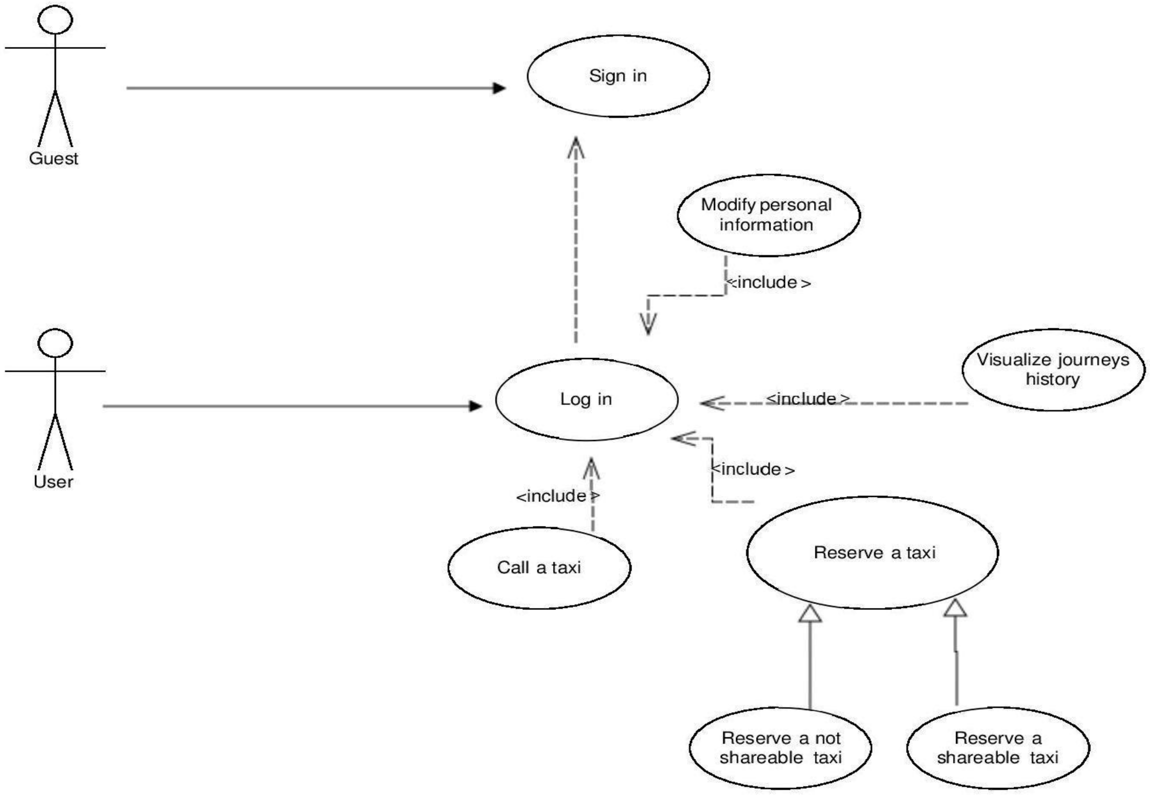
After lunch he decides to take one hour of rest before going to the meeting place with the taxi. At one point he is awakened by the sound of a notification about the confirmation of booking of the taxi that will come to the meeting point after 10 minutes. He prepares in a hurry but still reaches the starting point with 20 minutes late and not even a ride of a taxi. Cursing himself for being lying on the couch after dinner, he decides to call another taxi, aware that he will have to repay the journey.

5. UML MODELS

5.1 USE CASE LIST

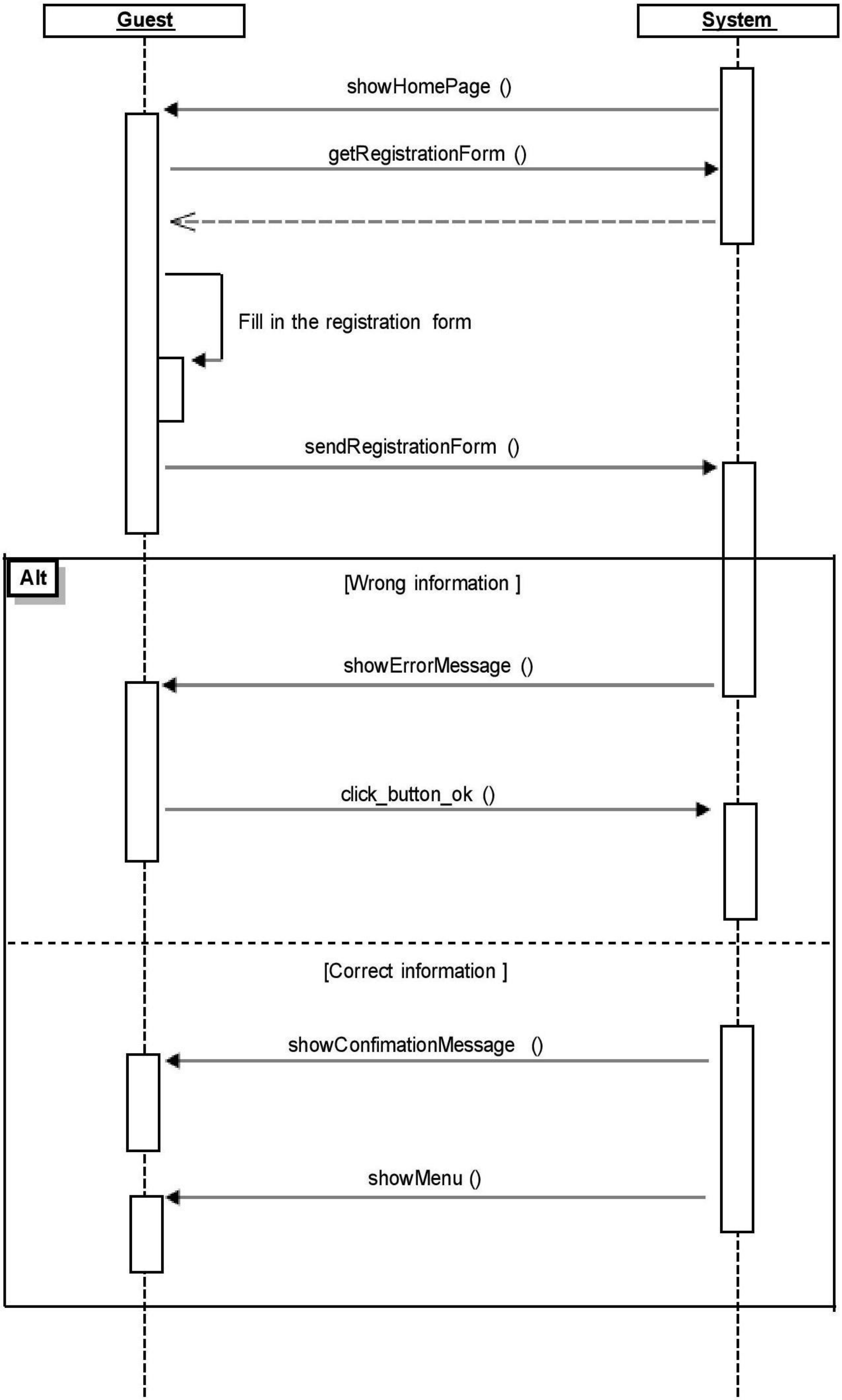
- Sign up
- Log in
- Calling of a taxi with acceptance of the taxi driver
- Calling of a taxi with the driver's refusal
- Reservation of a not shareable taxi
- Reservation of a shareable taxi
- Display information of a journey

5.2 USE CASE DIAGRAM

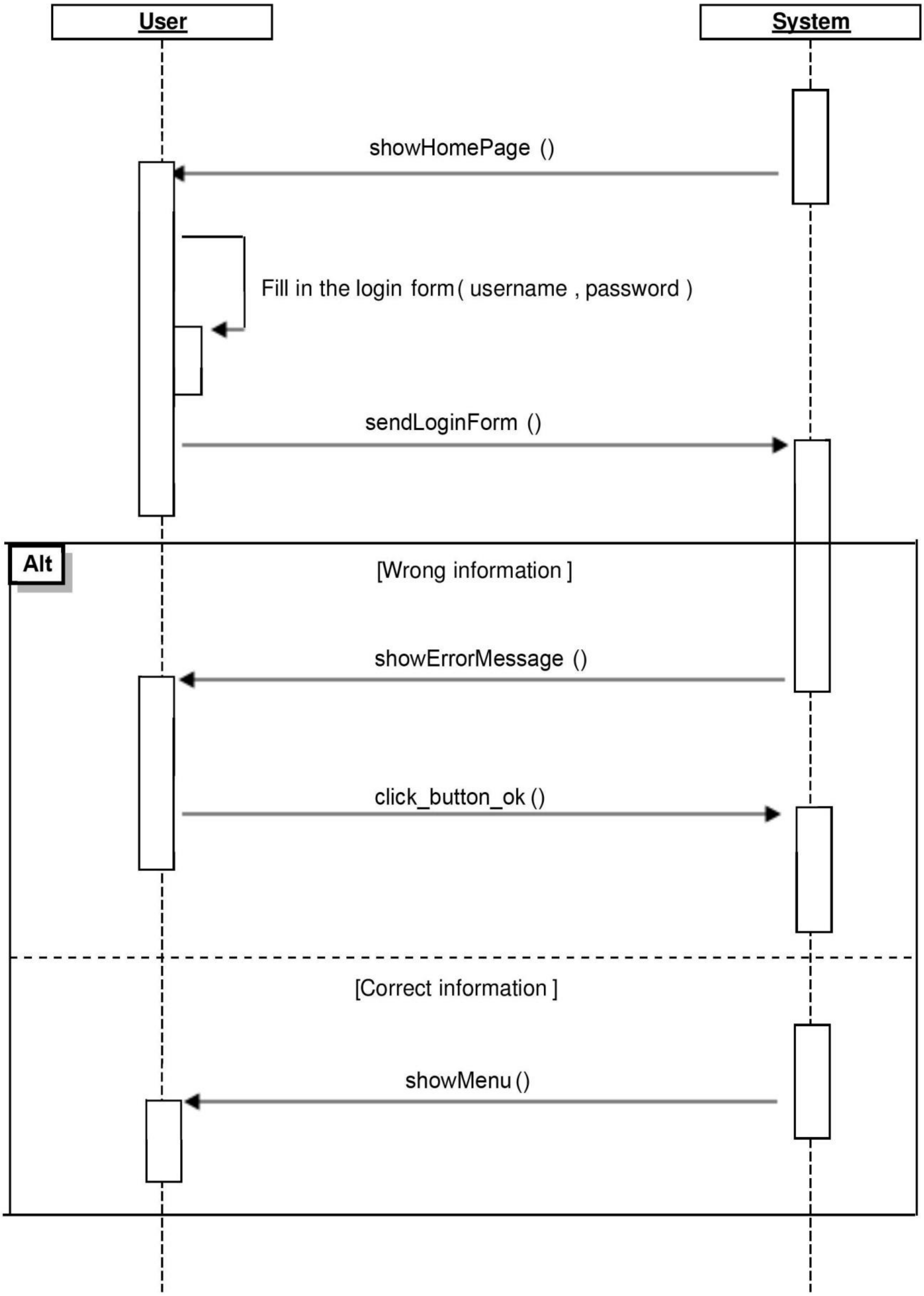


5.3 USE CASE DESCRIPTION AND RELATIVE SEQUENCE DIAGRAM

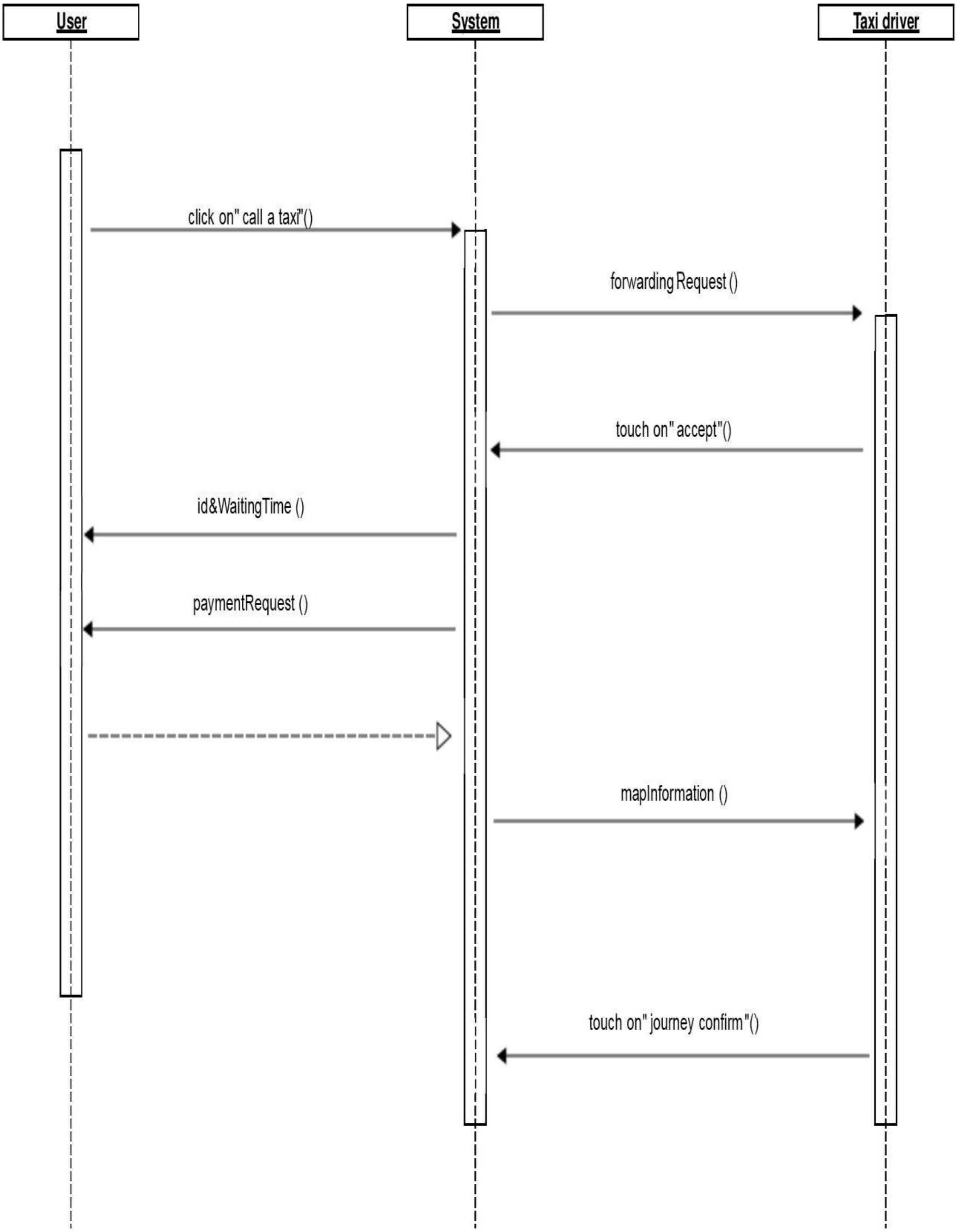
Name	Sign up
Actors	Guest
Entry Conditions	The guest isn’t registered.
Flow of events	• The guest enters the website or the app; • The guest clicks on the “SIGN UP” button; • The guest fills in the form where he has to write: -Name -Surname -Email -Address -Username -Password -Date of birth -Telephone number •The guest inserts the number of credit card and the CV2 security code. • The guest clicks the “DONE” button; • The system shows him the menu.
Exit conditions	Registration successfully done.
Exceptions	An exception can be caused if the username the guest inserts already exists or if some field aren’t filled.


Guest
System
showHomePage ()
getRegistrationForm ()
Fill in the registration form
sendRegistrationForm ()
Alt
[Wrong information]
showErrorMessage ()
click_button_ok ()
[Correct information]
showConfimationMessage ()
showMenu ()

Name	Log in
Actors	User
Entry Conditions	User has successfully signed up to the system.
Flow of events	• The user enters the web site or the app. • The user fills in the text fields username and password. • The user clicks on the “LOG IN” button.
Exit conditions	The system shows the user the menu.
Exceptions	The password and/or username inserted by the user are wrong. The System shows an error message to the user.

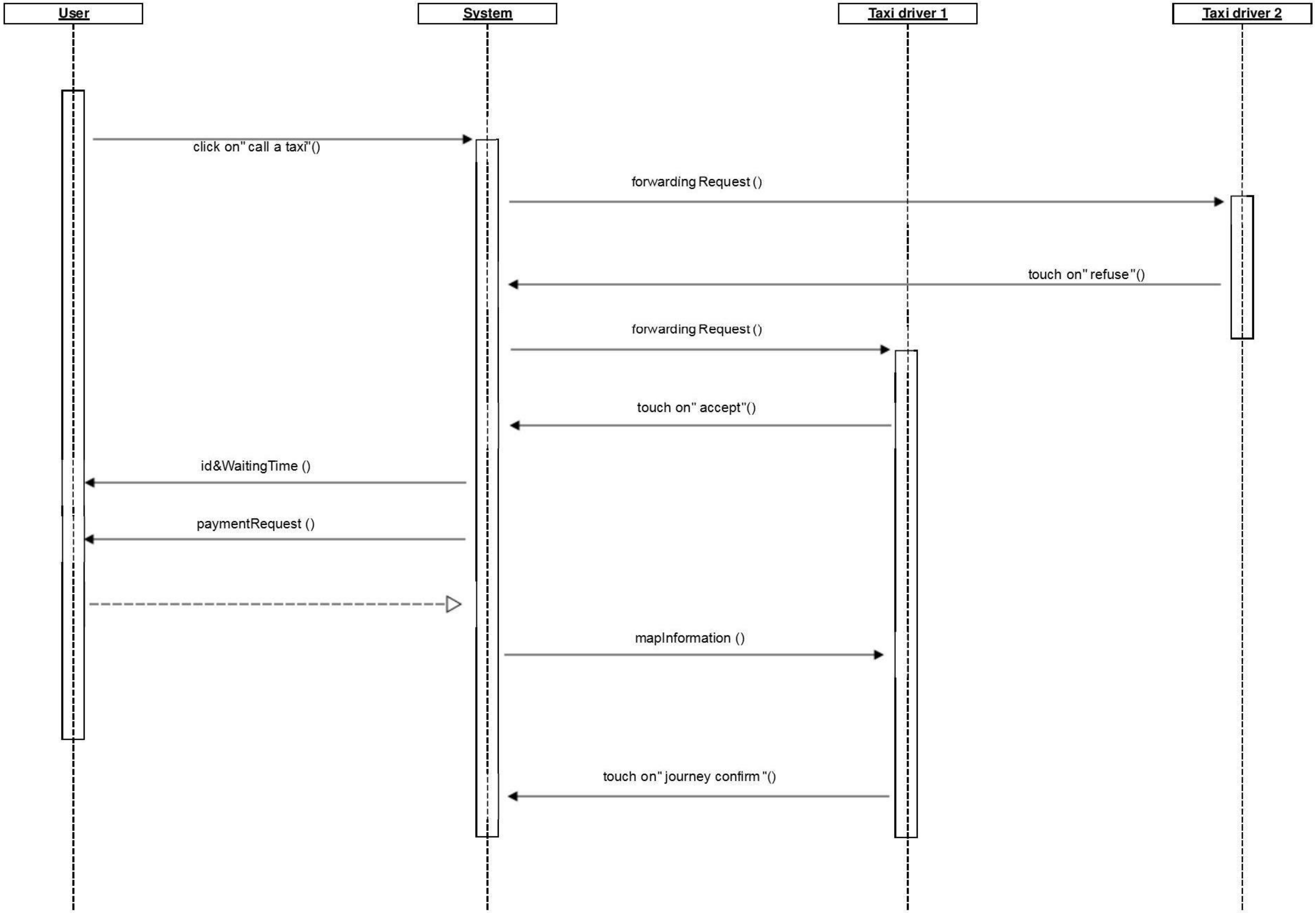

User
System
showHomePage ()
Fill in the login form(username , password)
sendLoginForm ()
Alt
[Wrong information]
showErrorMessage ()
click_button_ok ()
[Correct information]
showMenu ()

Name	Calling of a taxi with acceptance of the taxi driver
Actors	User, taxi driver
Entry Conditions	The user must be logged in The taxi driver must be in service
Flow of events	• The user clicks on the button “Call a taxi” in the menu. • The user inserts the origin and destination places. • The user waits while the system looks for an available taxi. • The taxi driver receives the call request with the position of the passenger and the destination and has 20 seconds to choose to accept or not. • The taxi driver touches "accept" on the screen in the car and heads for the passenger. • The user receives the identifier of the taxi and the waiting time. • The system takes from the user's credit card the amount due for the journey. • When the passenger leaves the taxi at his destination, the driver touches "journey confirm" on the screen and becomes available again.
Exit conditions	The user is where he asked to go.
Exceptions	The user could not be present, in this case the taxi driver notifies, by touching "user absent", the system that has already taken upfront the payment for the journey.


User
System
Taxi driver
click on" call a taxi"()
forwarding Request ()
touch on" accept"()
id&WaitingTime ()
paymentRequest ()
mapInformation ()
touch on" journey confirm"()

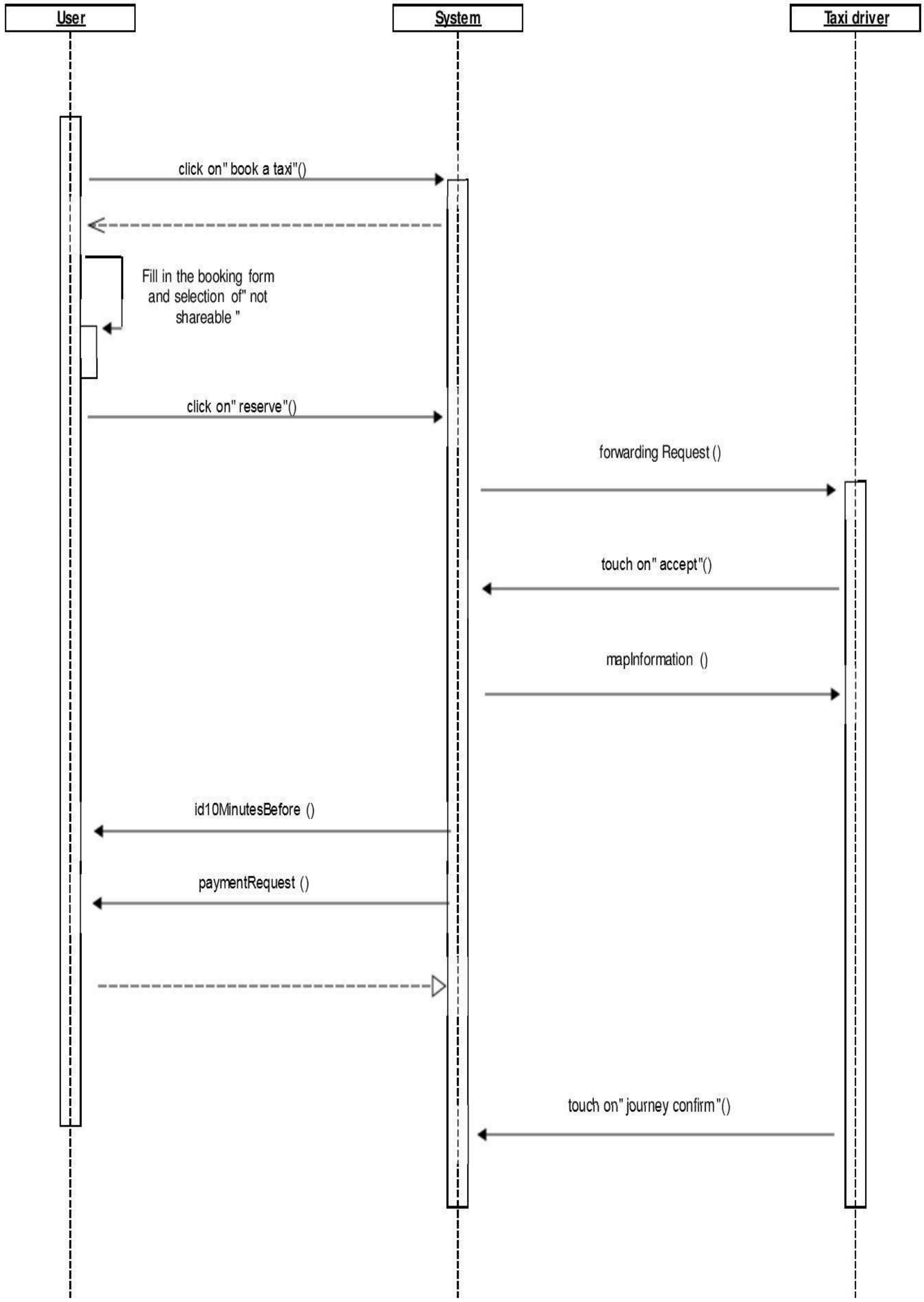
Name	Calling of a taxi with the driver's refusal
Actors	User, taxi drivers
Entry Conditions	The user must be logged in The taxi driver must be in service
Flow of events	• The user clicks on the button “Call a taxi” in the menu. • The user inserts the origin and destination places. • The user waits while the system looks for an available taxi. • The first taxi driver in the queue receives the call request with the position of the passenger and the destination, and has 20 seconds to choose to accept or not. • The driver is not available to go and take the user, so he touches "refuse" on the screen. • The system moves the taxi driver in the back of the queue and contact the next. • If also the next taxi driver isn't available, the situation is repeated until someone is available to accept the request. • The available taxi driver touches "accept" on the screen in the car and heads for the passenger. • The user receives the identifier of the taxi and the waiting time. • The system takes from the user's credit card the amount due for the journey. • When the passenger leaves the taxi at his destination, the driver touches "journey confirm" on the screen and becomes available again.

Exit conditions	The user is where he asked to go.
Exceptions	The user could not be present, in this case the taxi driver notifies the system, by touching "user absent", that has already taken upfront the payment for the journey.


User
System
Taxi driver 1
Taxi driver 2
click on" call a taxi"()
forwarding Request ()
touch on" refuse"()
forwarding Request ()
touch on" accept"()
id&WaitingTime ()
paymentRequest ()
mapInformation ()
touch on" journey confirm "()

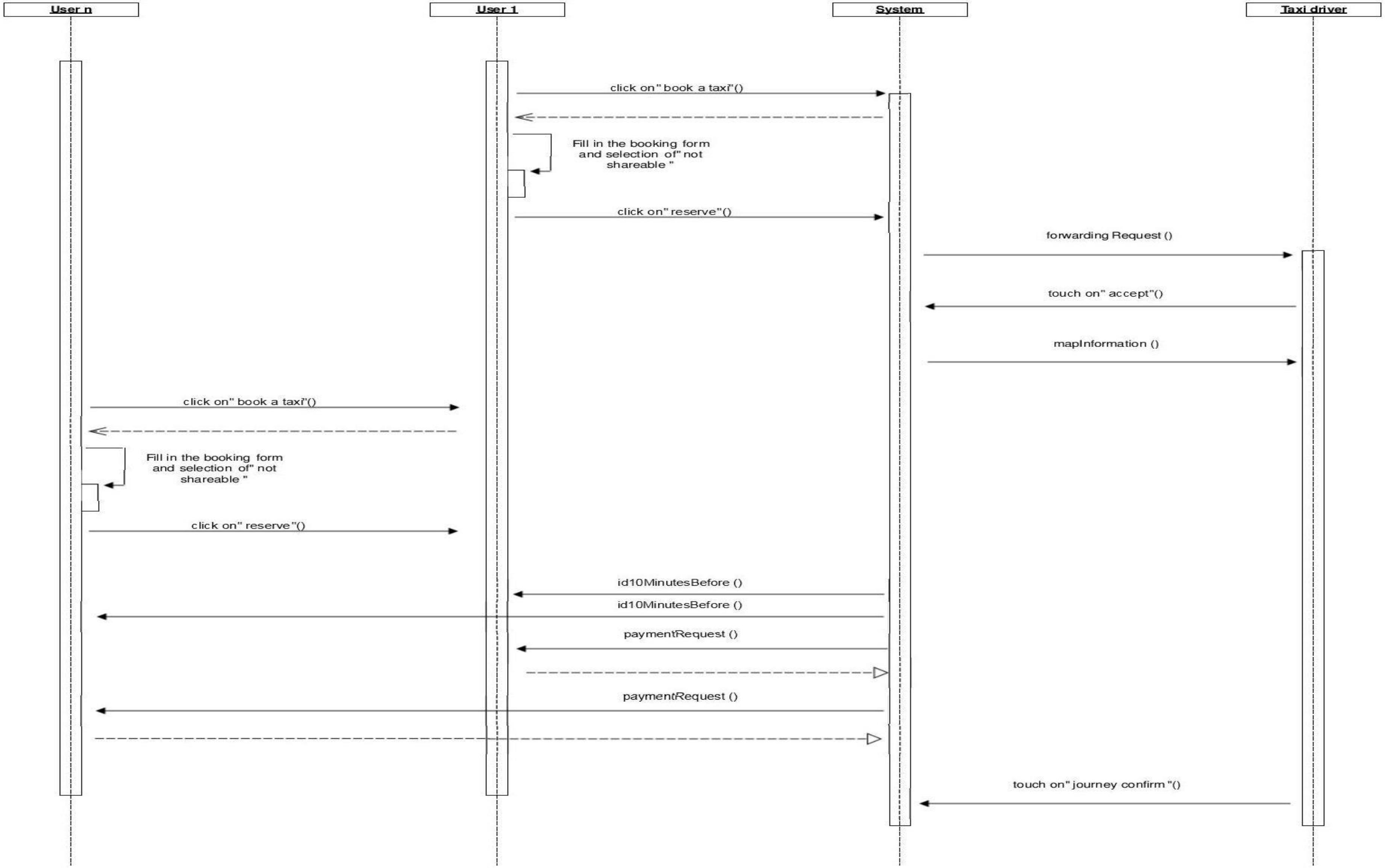
Name	Reservation of a not shareable taxi
Actors	User, taxi driver
Entry Conditions	The user must be logged in The taxi driver must be in service. There are at least two hours before the user wants to take the taxi.
Flow of events	• The user clicks "book a taxi". • The user completes the booking form with departure, destination and time of meeting. • The user selects "not shareable" in the appropriate field and click "reserve". • The first taxi driver in the queue receives the call request with the position of the passenger and the destination, and has 20 seconds to choose to accept or not. • The driver is not available to go and take the user, so he touches "refuse" on the screen. • The system moves the taxi driver in the back of the queue and contact the next. • If also the next taxi driver isn't available, the situation is repeated until someone is available to accept the request. • The available taxi driver touches "accept" on the screen in the car and he arranges to be present at the time received by the user. • 10 minutes before the meeting, the user receives the booking confirmation and the identification of the taxi. • The system takes from the user's credit card the amount due for the

	journey. • When the passenger leaves the taxi at his destination, the driver touches "journey confirm" on the screen and becomes available again.
Exit conditions	The user is where he asked to go.
Exceptions	The user could not be present, in this case the taxi driver notifies, by touching "user absent", the system that has already taken upfront the payment for the journey.

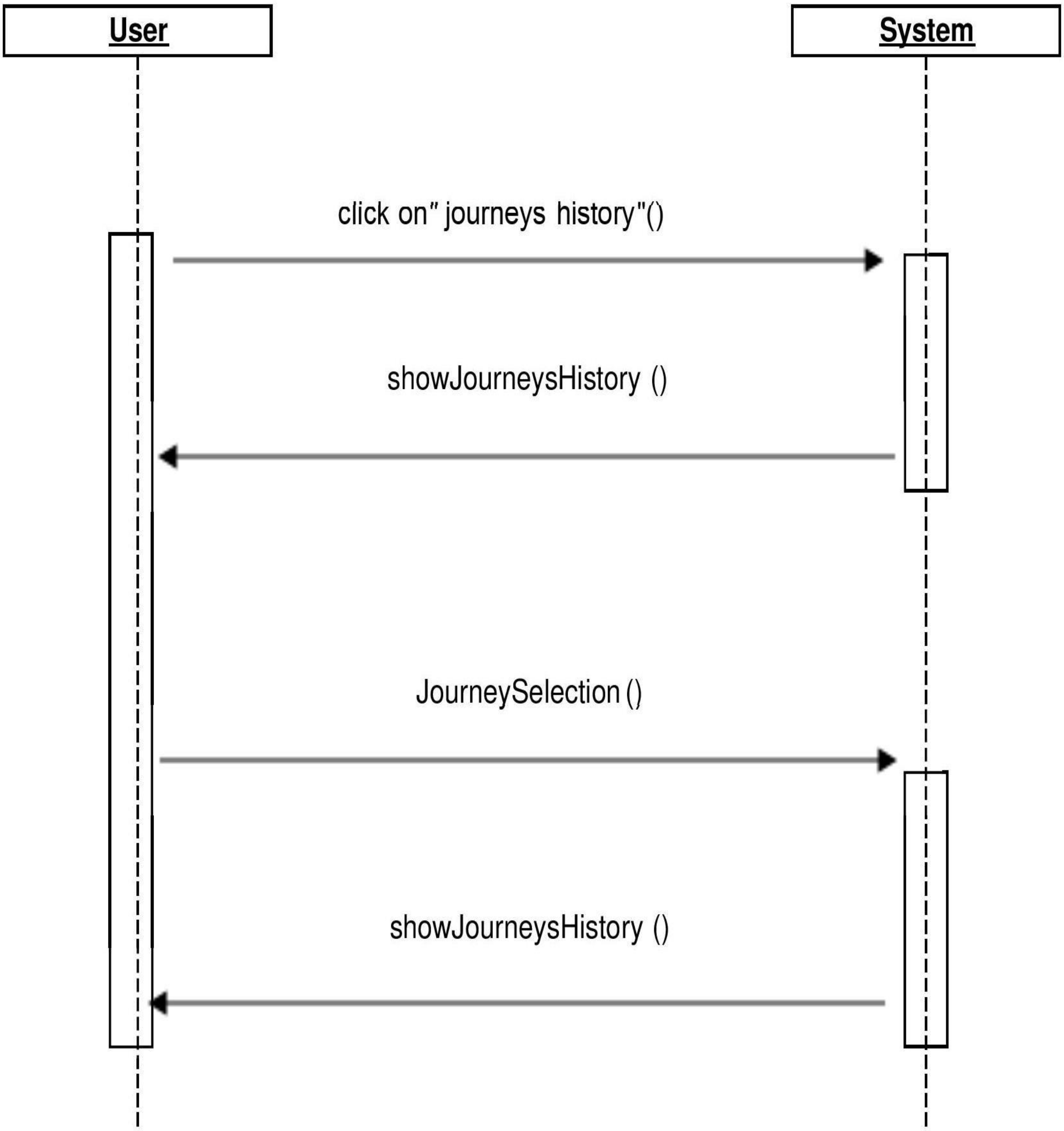

User
System
Taxi driver
click on" book a taxi"()
Fill in the booking form
and selection of" not
shareable "
click on" reserve"()
forwarding Request ()
touch on" accept"()
mapInformation ()
id10MinutesBefore ()
paymentRequest ()
touch on" journey confirm"()

Name	Reservation of a shareable taxi
Actors	Users, taxi driver
Entry Conditions	The user must be logged in The taxi driver must be in service There are at least two hours before the user wants to take the taxi.
Flow of events	• The user clicks "book a taxi". • The user completes the booking form with departure, destination and time of meeting. • The user selects "shareable" in the appropriate field, indicates the acceptable range of extension time of the journey, the number of people and then click "reserve". • The first taxi driver in the queue receives the call request with the position of the passenger and the destination, and has 20 seconds to choose to accept or not. • The driver is not available to go and take the user, so he touches "refuse" on the screen. • The system moves the taxi driver in the back of the queue and contact the next. • If also the next taxi driver isn't available, the situation is repeated until someone is available to accept the request. • The available taxi driver touches "accept" on the screen in the car and he arranges to be present at the time received by the user. • The situation is repeated if other users want to make a journey

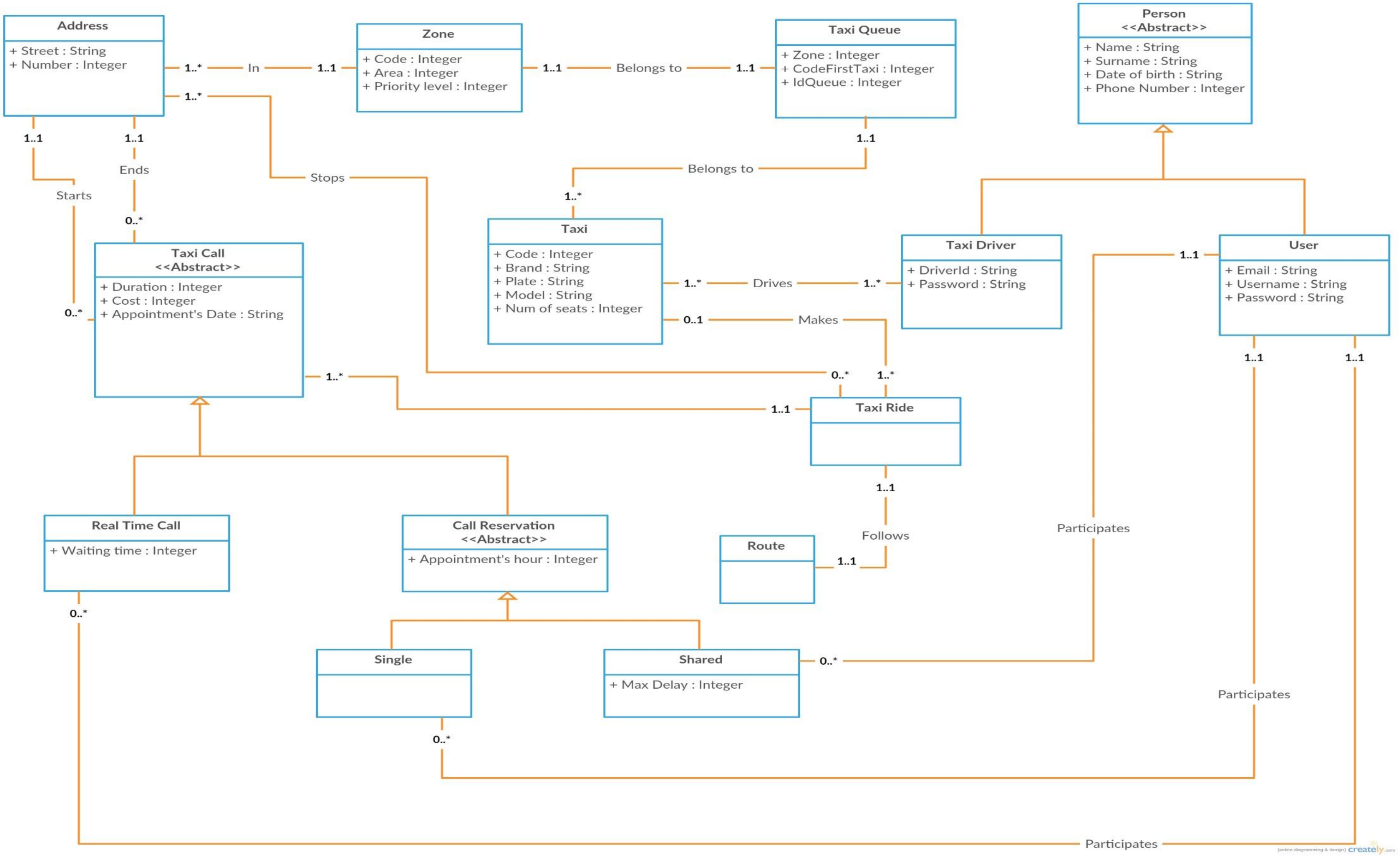
	shareable that the system detects compatible with the range selected by all users. In this case the taxi driver is obliged to accept and therefore receives meeting time and place of new users. • 10 minutes before the meeting, users receive the booking confirmation and the identification of the taxi. • The system takes from users' credit card the amount due for the journey. • When a passenger leaves the taxi at his destination, the driver touches "journey confirm" on the screen and continues the journey. • When the last passenger has arrived, the taxi driver becomes available again.
Exit conditions	Users are where they asked to go.
Exceptions	A user could not be present, in this case the taxi driver notifies, by touching "user absent", the system that has already taken upfront the payment for the journey.


User n
User 1
System
Taxi driver
click on" book a taxi"()
Fill in the booking form and selection of" not shareable "
click on" reserve"()
forwarding Request ()
touch on" accept"()
mapInformation ()
click on" book a taxi"()
Fill in the booking form and selection of" not shareable "
click on" reserve"()
id10MinutesBefore ()
id10MinutesBefore ()
paymentRequest ()
paymentRequest ()
touch on" journey confirm "()

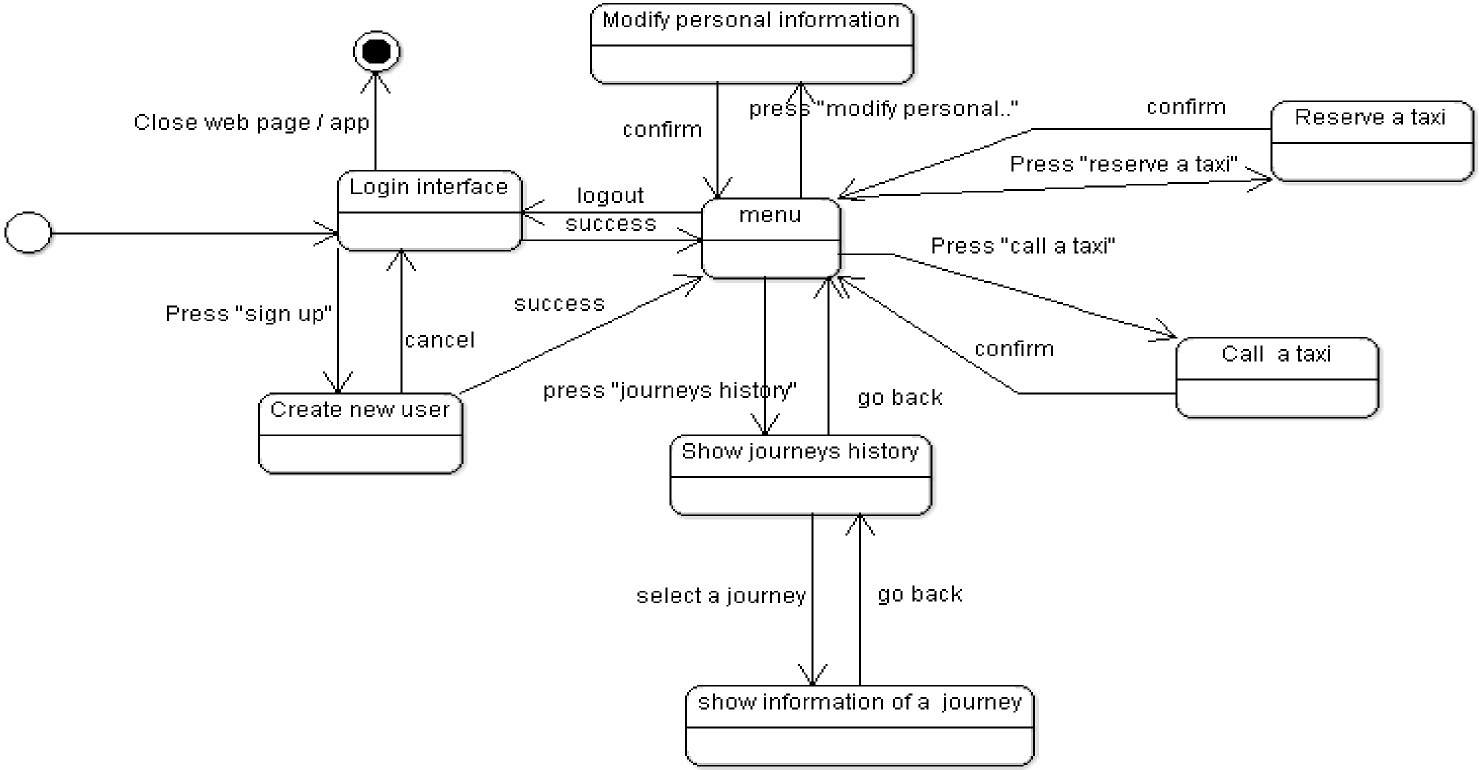
Name	Display information of a journey
Actors	User
Entry Conditions	The user must be logged in
Flow of events	• The user visualizes a list of journeys done by clicking on "journeys history" • The user selects a journey and a window appears with the following information: - identification of the taxi; - places of origin and destination; - arrival and departure times; - way; - duration of the journey; - fare paid;
Exit conditions	The system shows the user the list of the selected journey' information.
Exceptions	No exceptions.


User
System
click on" journeys history"()
showJourneysHistory ()
JourneySelection ()
showJourneysHistory ()

5.4 CLASS DIAGRAM



5.5 STATE CHART DIAGRAMS

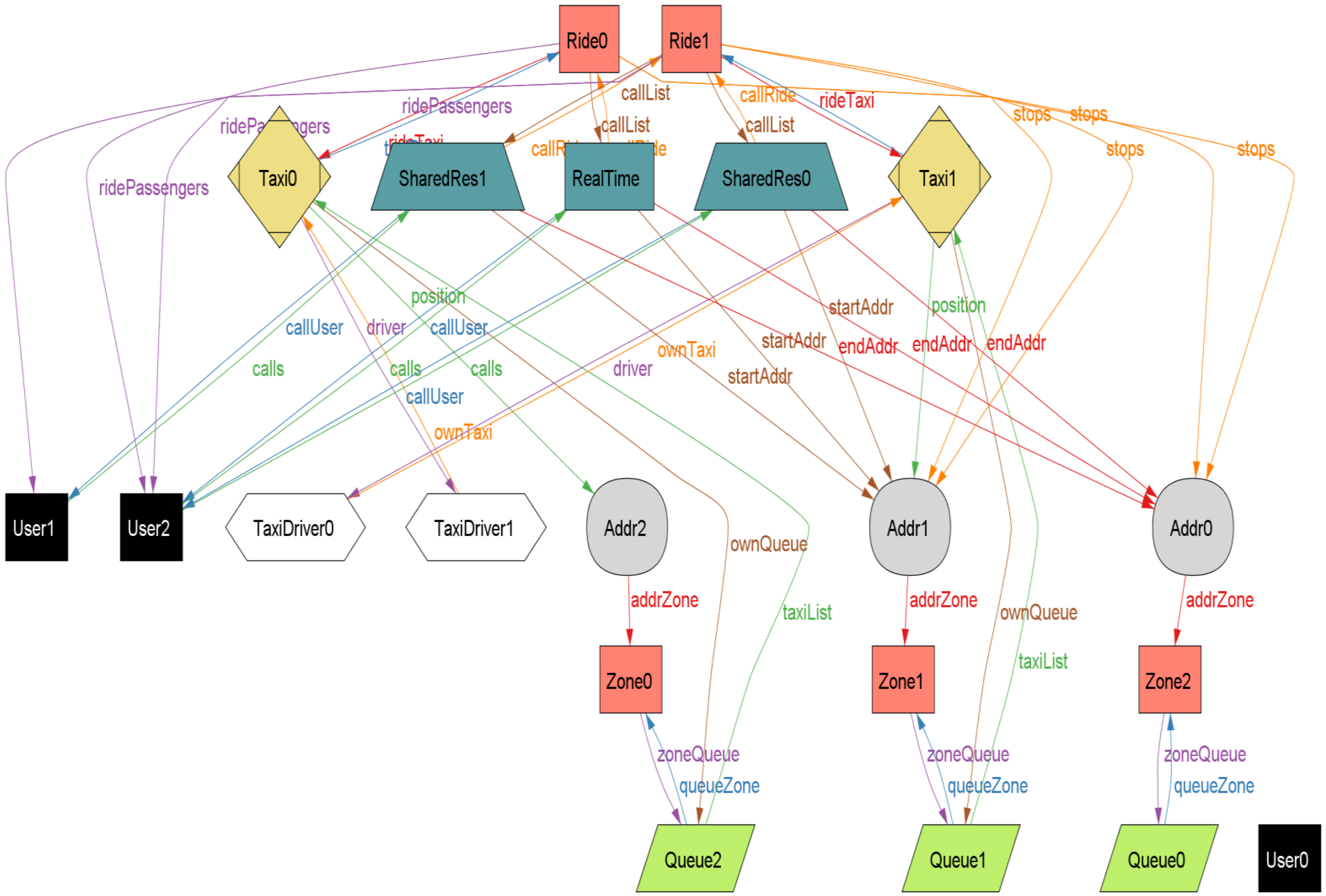


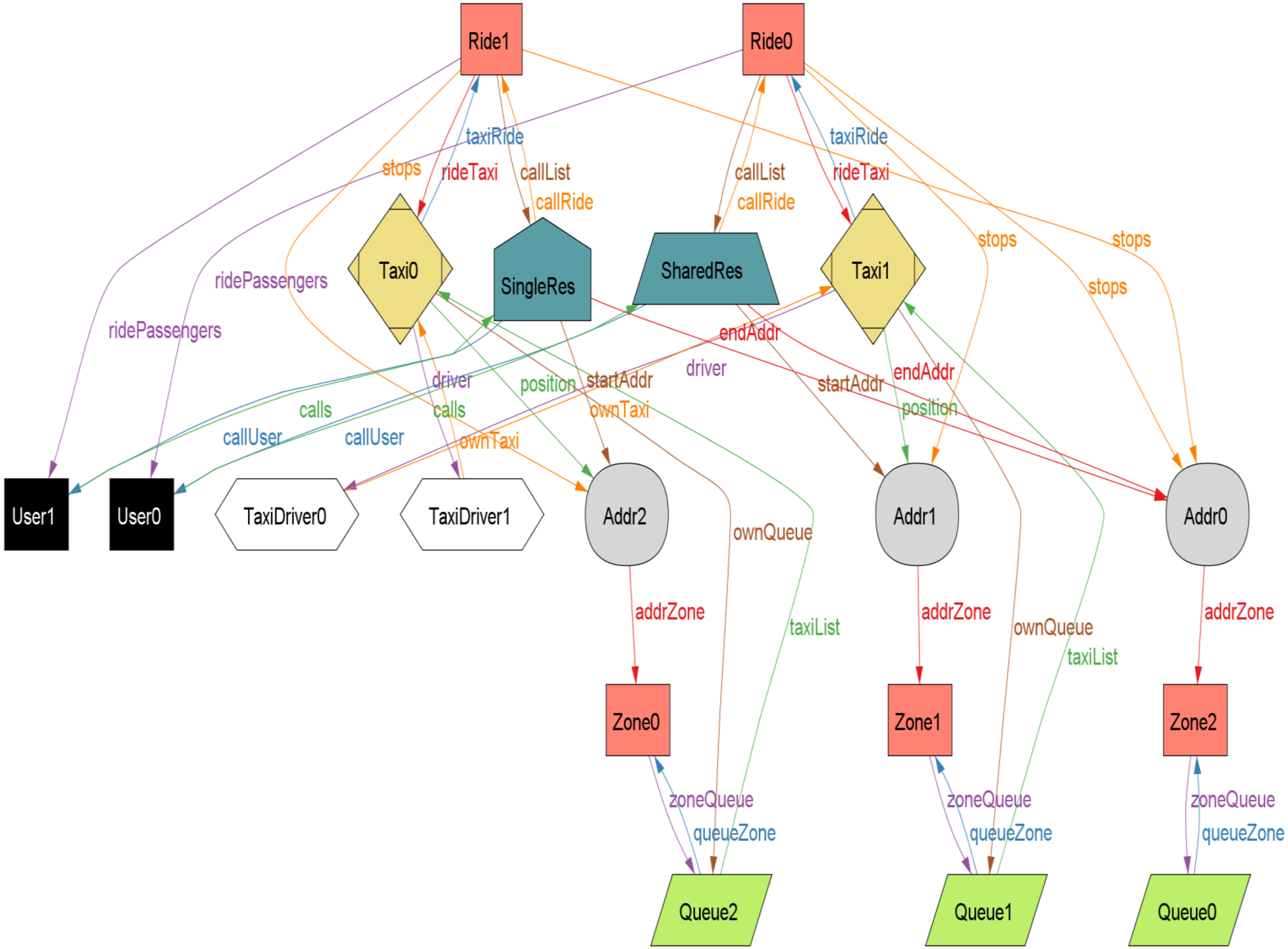
6. ALLOY MODELING

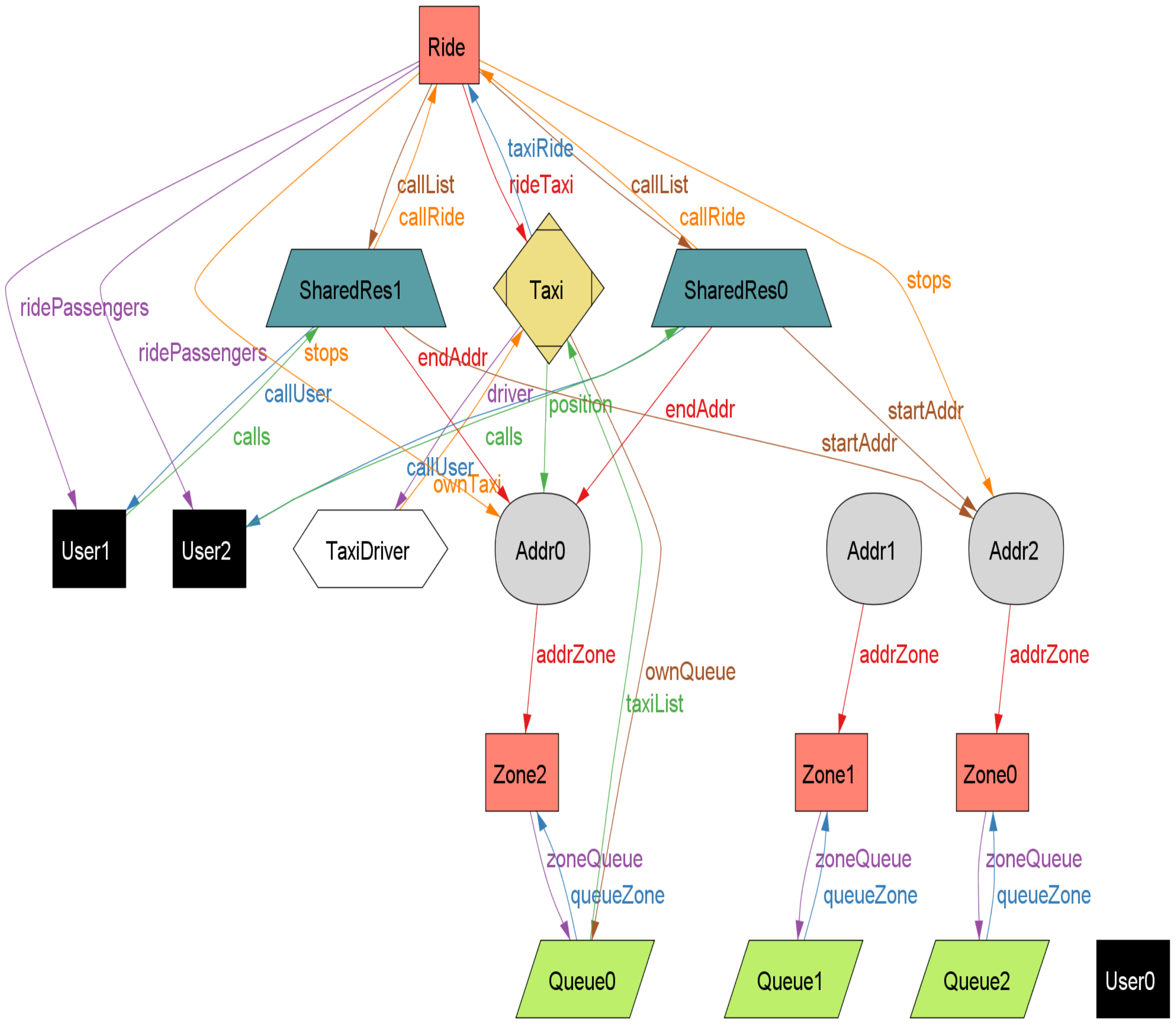
6.1 ALLOY CODE

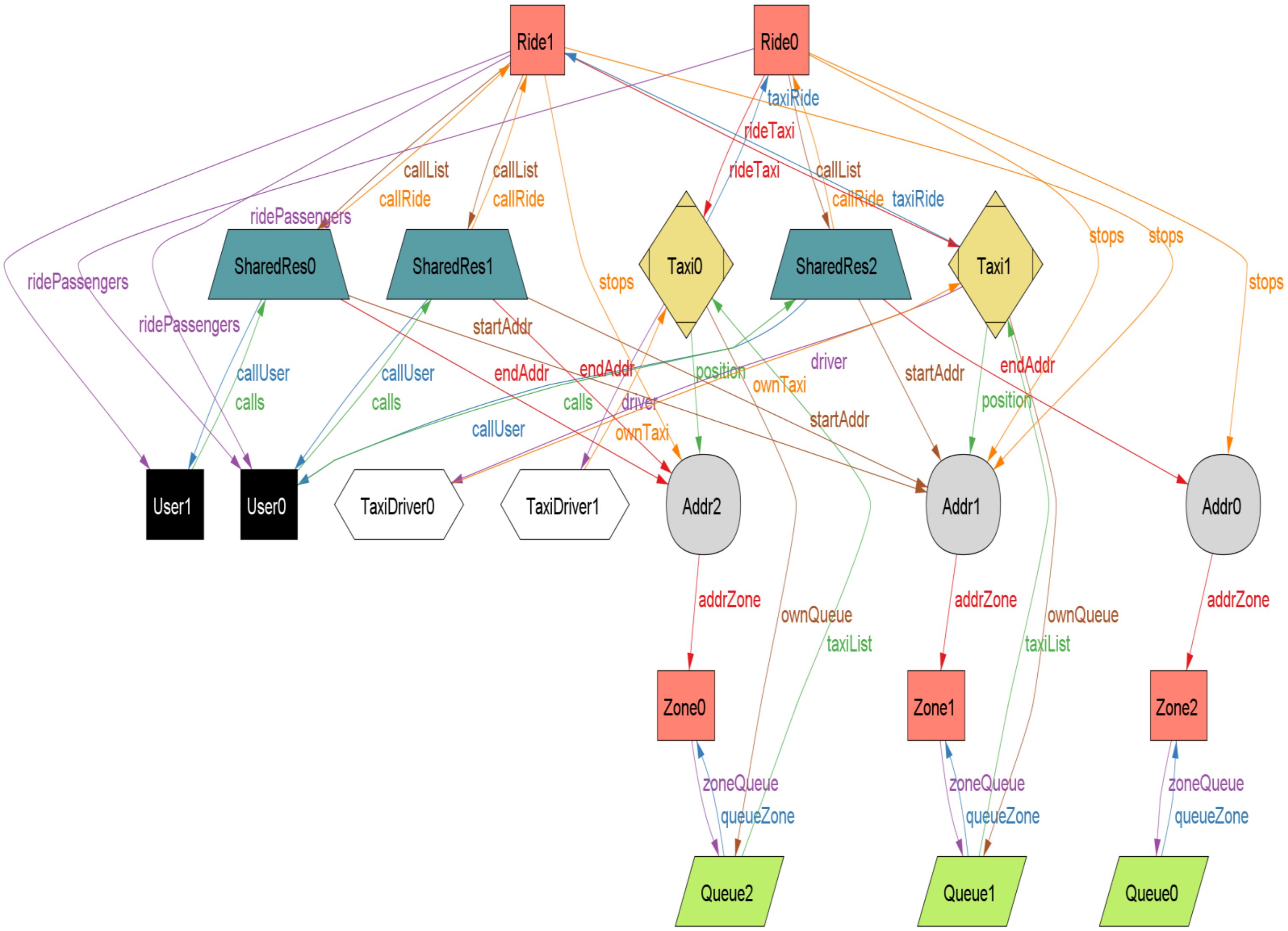
```
sig User{
        calls: set Call,
}

sig TaxiDriver{
        ownTaxi: Taxi,
}

sig Taxi{
        driver: TaxiDriver,
        taxiRide: Ride,
        ownQueue: Queue,
        position: Addr,
}

sig Queue{
        queueZone: Zone,
        taxiList: set Taxi,
}

sig Zone{
        zoneQueue: Queue,
}

sig Addr{
        addrZone: Zone,
}

abstract sig Call{
        callUser: User,
        callRide: Ride,
        startAddr: Addr,
        endAddr: Addr,
}{
        startAddr != endAddr
}

sig RealTime extends Call{

}

abstract sig Reservation extends Call{}

sig SingleRes extends Reservation{}

sig SharedRes extends Reservation{}
sig Ride{
        rideTaxi: Taxi,
        ridePassengers: set User,
        callList: set Call,
        stops: set Addr,
}{
        #ridePassengers > 0
        #ridePassengers <= 5
        #callList >0
        #callList <= 5
}

fact{

// non esistono due taxi che servono una stessa corsa
        no disj t1,t2 : Taxi | t1.taxiRide = t2.taxiRide

//c'è una corrispondenza biunivoca tra taxi driver e Taxi
        all d : TaxiDriver, t: Taxi | d.ownTaxi = t iff t.driver = d

//corrispondenza biunivoca tra zone e code
        all q : Queue , z : Zone | q.queueZone = z iff z.zoneQueue = q

//se un taxi è associato ad una coda allora quel taxi deve essere nella lista dei taxi in
quella coda
        all t : Taxi, q : Queue | t.ownQueue = q iff t in q.taxiList

// non esistono due corse servite dallo stesso taxi contemporaneamente
        no disj r1,r2 : Ride | r1.rideTaxi = r2.rideTaxi

//corrispondenza biunivoca tra corsa e taxi
        all r : Ride, t : Taxi | t.taxiRide = r iff r.rideTaxi = t

//corrispondenza biunivoca tra chiamate e utente
        all u: User, c: Call | c.callUser = u iff c in u.calls

//se una corsa serve due sharedRes allora queste sono di due utenti diversi
        no c1,c2: SharedRes| c1 !=c2 and c1.callUser = c2.callUser and c1.callRide =
c2.callRide

//nessuna corsa puo essere associata ad una singleRes e ad un altra chiamata di diverso
tipo
         no c1: SingleRes, c2: Call | c1 != c2 and (c1.callRide = c2.callRide or one r : Ride |
c1 in r.callList and c2 in r.callList)

//nessuna corsa puo essere associata ad una RealTime e ad un altra chiamata di diverso
tipo
         no c1: RealTime, c2: Call | c1 != c2 and (c1.callRide = c2.callRide or one r : Ride |
c1 in r.callList and c2 in r.callList)
//non esistono due corse associate ad una stessa chiamata
        no disj r1,r2:Ride, c: Call | c in r1.callList and c in r2.callList

//corrispondenza tra corse e chiamate
        all c: Call, r:Ride | c in r.callList iff c.callRide = r

//un utente puo essere un passeggero di una corsa se e solo se quella corsa sta servendo
una chiamata fatta da quell utente
        all u: User, r: Ride | u in r.ridePassengers iff one c : Call | c in u.calls and c in
r.callList

//tutte le chiamate associate ad una corsa condivisa hanno la stessa zona di partenza
        all c1,c2: SharedRes | c1.callRide = c2.callRide implies c1.startAddr.addrZone =
c2.startAddr.addrZone

//i taxi vengono assegnati alla coda corrispondente alla zona in cui si trovano
        all t: Taxi | t.ownQueue = t.position.addrZone.zoneQueue

//una corsa condivisa si ferma a tutti gli indirizzi di partenza specificati nelle chiamate
corrispondenti a quella richiesta
        all r: Ride, c: Call | c in r.callList implies (c.startAddr in r.stops and c.endAddr in
r.stops)


}
```

6.2 ALLOY WORLDS




Ride1
Ride0
taxiRide
rideTaxi
callList
callRide
stops
taxiRide
rideTaxi
callList
callRide
stops
stops
stops
Taxi0
SingleRes
SharedRes
Taxi1
ridePassengers
ridePassengers
endAddr
endAddr
driver
driver
position
startAddr
startAddr
ownTaxi
ownTaxi
calls
calls
position
callUser
callUser
User1
User0
TaxiDriver0
TaxiDriver1
Addr2
Addr1
Addr0
ownQueue
ownQueue
addrZone
addrZone
addrZone
taxiList
taxiList
Zone0
Zone1
Zone2
zoneQueue
queueZone
zoneQueue
queueZone
zoneQueue
queueZone
Queue2
Queue1
Queue0


Ride
callList
callRide
taxiRide
rideTaxi
callList
callRide
stops
ridePassengers
ridePassengers
SharedRes1
Taxi
SharedRes0
stops
endAddr
callUser
driver
position
endAddr
startAddr
calls
calls
startAddr
callUser
ownTaxi
User1
User2
TaxiDriver
Addr0
Addr1
Addr2
addrZone
ownQueue
addrZone
addrZone
taxiList
Zone2
Zone1
Zone0
zoneQueue
queueZone
zoneQueue
queueZone
zoneQueue
queueZone
Queue0
Queue1
Queue2
User0


Ride1
Ride0
taxiRide
rideTaxi
rideTaxi
callList
callRide
callList
callRide
callList
callRide
taxiRide
ridePassengers
ridePassengers
ridePassengers
stops
stops
stops
stops
SharedRes0
SharedRes1
Taxi0
SharedRes2
Taxi1
startAddr
endAddr
endAddr
callUser
calls
callUser
calls
calls
driver
position
ownTaxi
driver
ownTaxi
startAddr
startAddr
endAddr
position
callUser
User1
User0
TaxiDriver0
TaxiDriver1
Addr2
Addr1
Addr0
addrZone
addrZone
addrZone
ownQueue
taxiList
ownQueue
taxiList
Zone0
Zone1
Zone2
zoneQueue
queueZone
zoneQueue
queueZone
zoneQueue
queueZone
Queue2
Queue1
Queue0

7. DOCUMENT PERSPECTIVE

7.1 REFERENCE DOCUMENTS

- Specification Document: Assignments 1 and 2.pdf
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications

7.2 FUTURE IMPLEMENTATION

- Improve the security
- New functions like the possibility of made a feedback on the taxi drivers
- The application now doesn’t consider all the particular concerning the payments, in future implementations will be added the connection with an external payment service like for example paypal.

7.3 HARDWARE CONSTRAINTS

For the server there are no limits because is all maintained from Google, for the smartphone application its size is not over 100MB and is portable over the three main S.O. android, Microsoft and IOS with no constraints about the hardware (RAM,CPU).

8. USED TOOLS

Microsoft Office: to write this document.
Creately tool: to draw the Class Diagram.
Gliffy.com and ArgoUML: to create the other UML diagrams.
Alloy Analizer 4.2: to prove the consistency of our model.

Hours of work: Vincenzo about 25 hours

Simone about 25 hours

Matteo about 25 hours