Politecnico di Milano
Academic Year 2015/2016
Software Engineering 2: “myTaxiService”
Requirement Analysis and Specification
Document

Massimo Schiavo, Marco Edoardo Cittar

November 6, 2015

SUMMARY

SECTION 1

DESCRIPTION OF THE GIVEN PROBLEM

We will design and implement myTaxiService, a new web and mobile application to optimize the taxi service in big cities. It should simplify the access of passengers to the service and guarantee a fair management of taxi queues. The users will have to register and login in order to use the application, then they can request a taxi and be informed about the code of the incoming taxi and the waiting time.
We suppose the city is divided in zones and every one of them has a queue of available taxis present in the zone, whose position is calculated according to GPS. After a request arrives, the system informs the first taxi in the queue of the zone from which the request came. If the taxi accept the request, the system sends the confirmation to the user. If not, the system will put it at the end of the queue and forward the request to the next available taxi. To accept ride requests, taxi drivers will have to login through the mobile app like normal users. Then they can set themselves as available and receive ride requests. A user can also reserve a taxi by inserting origin, destination and time of the ride. The request must be submitted at least two hours before. The system will allocate a taxi and notify the user 10 minutes before the ride.

SECTION 1

ACTORS:

1. **Guest**: the guests are users who are not registered yet. They must sign themselves up into the system in order to use the features available to registered users.

2. **Registered user**: this type of user, after successful login, has access to all the features of the application as a customer. They can request rides, be them simple or detailed, and receive notifications after a ride has been confirmed.

3. **Driver**: this type of user, after a successful login, can set himself as available and so can be notified when a new ride request arrives and, if he sees it, to accepts it.

SECTION 1

OUR GOALS:

We have divided the goals for every categories of actors we have identified.

Guests should be able to:

[G1] Register themselves into the system

[G2] Log themselves into the system

Users should be able to:

[G3] See number of available taxis of the zone they're in

[G4] Make a request for a simple ride

[G5] Make a request for a detailed ride

OUR GOALS:

Drivers should be able to:

[G6] Set themselves as available

[G7] Read and accept ride requests

The system should:

[G8] Notify passengers after the confirmation of a simple request

[G9] Notify passenger ten minutes before the ride reserved through a detailed request

[G10] Forward request to the first taxi in queue

[G11] After 30 seconds, forward the request to the second taxi in queue and put the first at the end

SECTION 2

OVERALL DESCRIPTION:

Product perspettive:

The application we will project is both a web application and a mobile application. It will not interact with any other existing application or system. It will be user based and we will not provide any internal interface for administration.

User characteristics:

People that will use our application are the ones interested in using the taxi service in the city. They will be able to request a taxi for a ride without any voice call but just with few clicks.

SECTION 2

OVERALL DESCRIPTION:

Assumptions:

- When the app or the web page is closed the user automatically logs out.
- If a driver sees a ride request, he always accepts it.
- Users can't cancel ride requests.
- If the system sends a notification the user will always receive it.
- Users can make an unlimited number of daily ride requests.
- In case of empty queue 10 minutes before a detailed ride, the application will always manage to find one within the time of the ride.
- Users insert existing addresses when making requests.
- The system knows if the credentials inserted during login belong to a user or a driver, so it will display the correct personal page afterwards.

SECTION 3

EXTERNAL INTERFACE REQUIREMENTS

User interfaces:

Here are presented some drafts which represent how the application should look like. We decided to represent only the mobile application but the web one will be the same.

1) Login page

This is the page that shows up when the application is started. It allows the guests to log themselves into the system or register themselves if they don't have an account yet.

2) User page

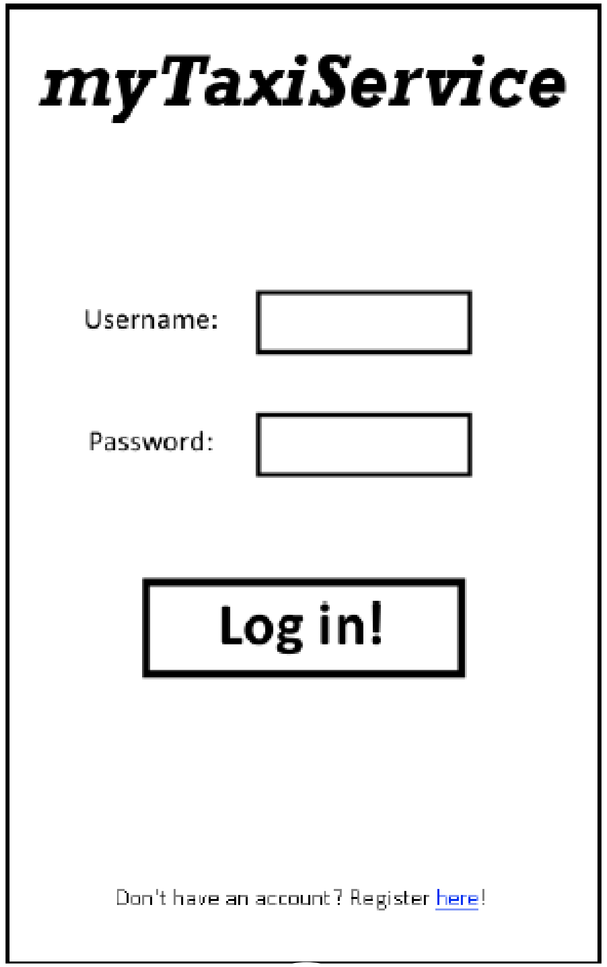
After login, this is what users will see. The graphic is minimal but really user friendly, so that users can easily access to all of their functions.

3) Driver page

After login, this is the page drivers will see. It allows them to set themselves as available and start receiving requests and access to the page where the requests are.

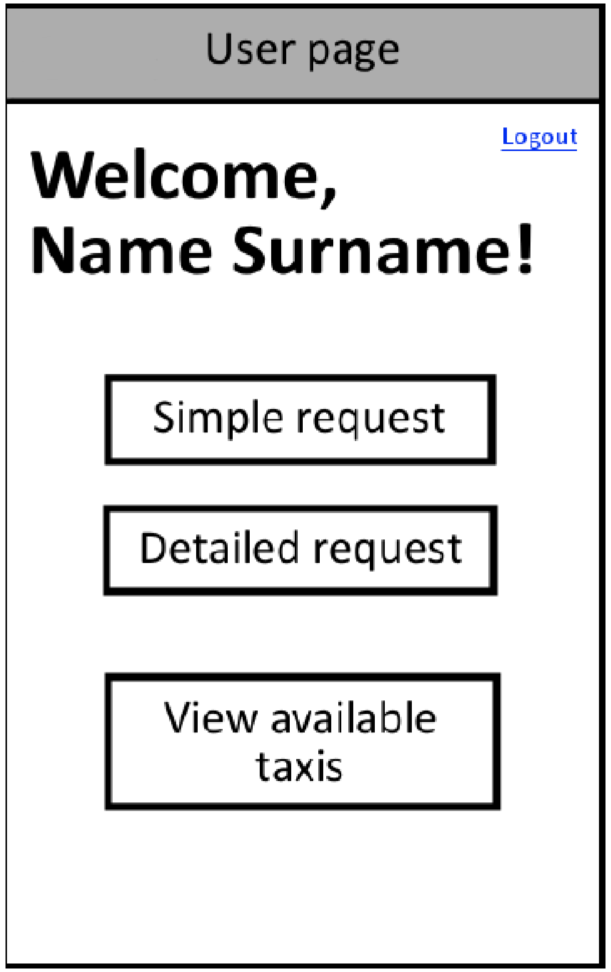
SECTION 3

EXTERNAL INTERFACE REQUIREMENTS

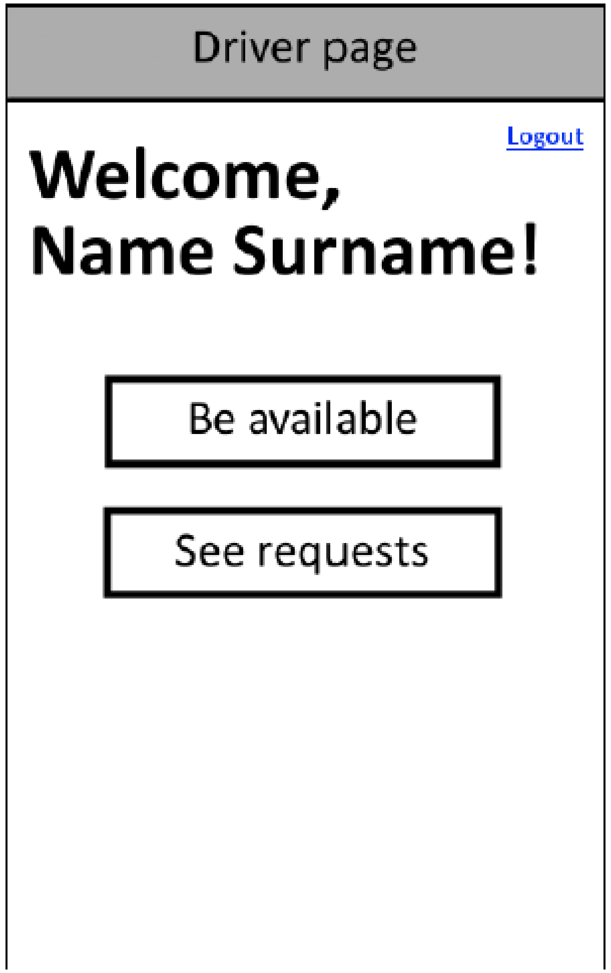


Don't have an account? Register here!

1



2



3

SECTION 3

FUNCTIONAL REQUIREMENTS

- **[G1] Register themselves into the system**
  [R1] Guests can only see the login page.
  [R2] Guests can only access to the registration form.
  [D1] All the fields must be completed in a formal correct way.

- **[G2] Log into the system**
  [R1] Guests have to provide valid username a password in order to log themselves into the system.
  [R2] The wrong insertion of one's own credentials will not allow to log himself into the system.

- **[G3] See number of available taxis of the zone he's in**
  [R1] The system has to visualize on the personal page the number of taxi available in the zone.

SECTION 3

FUNCTIONAL REQUIREMENTS

- **[G4] Make a request for a simple ride**
  [R1] Users have to insert the correct departure's address.
  [D1] The inserted address has to be formed by a street and a civic number.

- **[G5] Make a request for a detailed ride**
  [R1] Users have to insert the correct departure's address.
  [R2] Users have to insert the correct destination's address.
  [R3] Users have to insert feasible date and time for the reservation.
  [D1] The inserted address has to be formed by a street and a civic number.
  [D2] Date must look like this: DD/MM/YY, and time like this: HH:MM.

- **[G6] Set themselves as available**
  [R1] The system has to provide a function through which drivers can inform it of their availability.
  [R2] Once they're available, the system has to insert them into the Taxi Queue of their own zone.

SECTION 3

FUNCTIONAL REQUIREMENTS

- **[G7] Read and accept ride requests**

  [R1] The system has to provide a function to read and accept request

- **[G8] Notify passengers after the confirmation of a simple request**

  [R1] The system has to notify the user who made a simple request as soon as possible, this means as a taxi is available, informing him about the waiting time.

- **[G9] Notify passengers 10 minutes before the ride reserved through a detailed request**

  [R1] The system has to notify the user who reserved a taxi within ten minutes before the time of the ride.

SECTION 3

FUNCTIONAL REQUIREMENTS

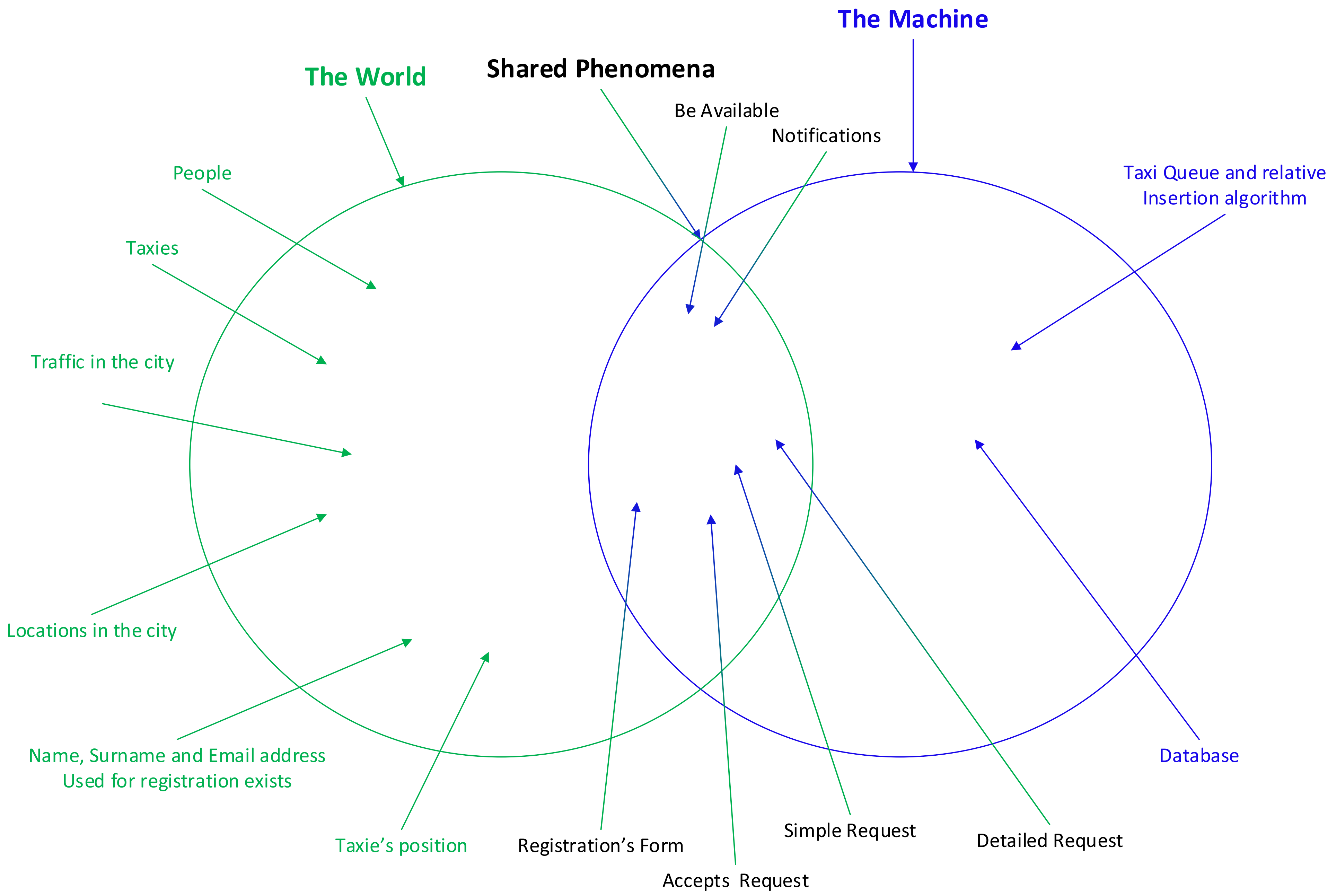
- **[G10] Forward requests to the first taxi in queue**

  [R1] The system has to forward requests to the first taxi in queue allowing him to accept the request.
  [R2] After a confirmation by a driver the system has to move the driver at the end of the queue and set him as not available.

- **[G11] After 30 seconds, forward the request to the second taxi in queue and put the first at the end**

  [R1] After a missed confirmation the system has to move the first driver in queue to the last position and forward the request to the second until one driver confirm the request.

SECTION 3 THE WORLD AND THE MACHINE



SECTION 3

SCENARIOS:

Scenario 1

Bob has planned to go to a workshop that will take place on the other side of Milan. He woke up early because he knows that there's always so much traffic in the city. Bob is ready to go with his car when the unexpected happens: the car seems broken. Without wasting any time he opens myTaxi app on his phone. He logs into the system and request a Taxi for a simple ride. John, already logged into the system as driver, sees the simple request on his phone and accepts it. The system immediately notifies Bob telling him John's taxi code and the relative waiting time. Bob has to attend just a few minutes because John's Taxi is only one kilometre away from him. Bob is happy because he will arrive on time despite the unexpected discovery of having his car broken.

SECTION 3

SCENARIOS:

Scenario 2

Bob, during the lunch break at the workshop, realizes that at the end of the event he doesn't have his car because he arrived in taxi. He also realizes that the end of the workshop is scheduled at 6p.m, critical time for traffic in the city with several waiting time for taxis. In order to this consideration, Bob opens myTaxi app and reserves a taxi for 6.15 p.m. to be sure that a taxi will be there at that moment. At 6.05 p.m. Bob's phone rings. A notification with the code of taxi that will arrive to him has just been received from his phone.

SECTION 3

SCENARIOS:

Scenario 3

It's 12a.m. and John, a taxi driver, decides to have a coffee break but he also decides to not log out from the app because he is the third in the Taxi Queue. He thinks he can drink a coffee and smoke a cigarette before he becomes the first of the Taxi Queue. . . but he is wrong. The other two driver that were in list before him log out from the system and John becomes the first in the Taxi Queue. A request arrives but Bob doesn't see it because he is still drinking his coffee. After 30 seconds John hasn't accepted yet and the system puts him into the last position forwarding the request to the second in list.

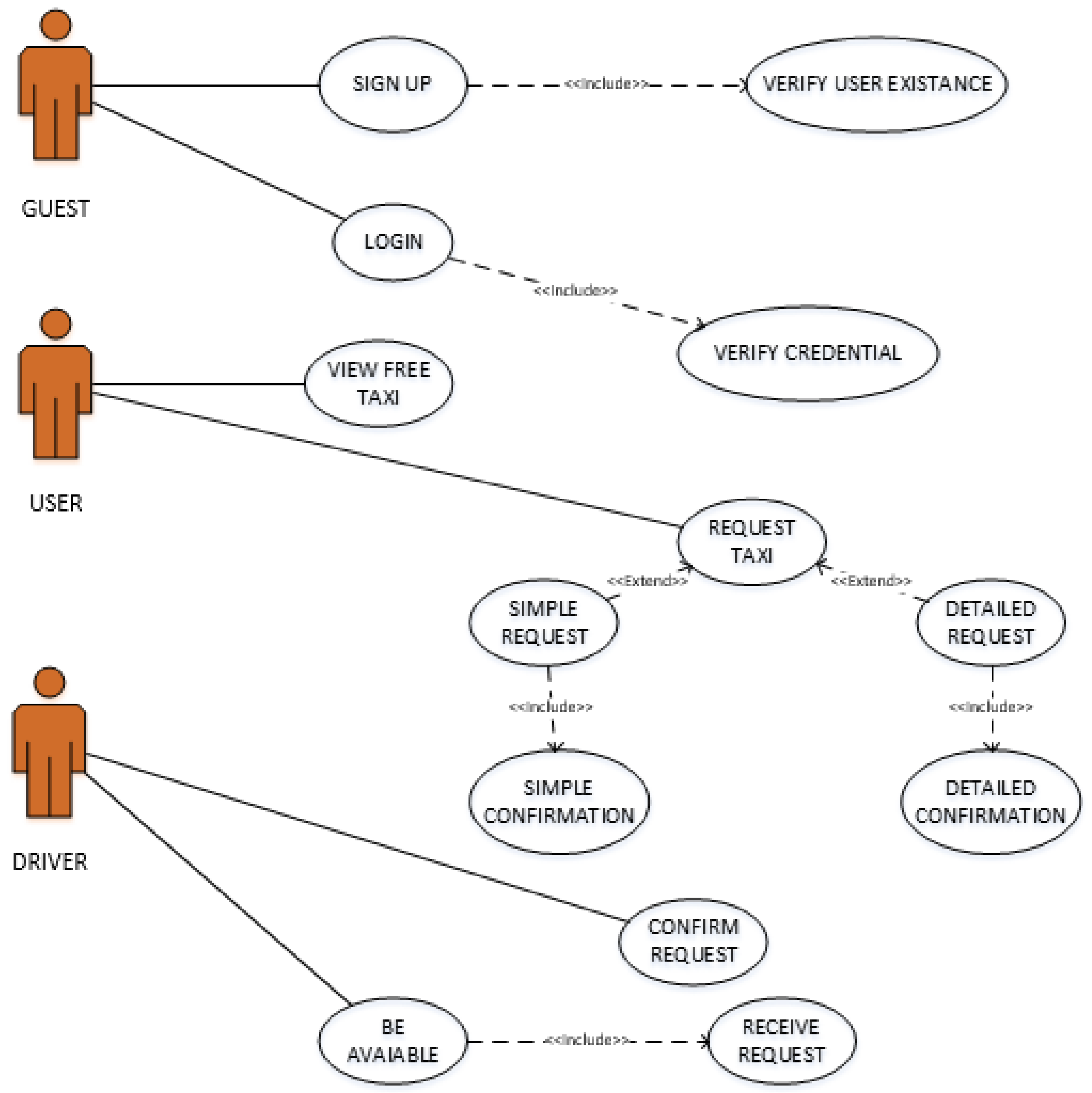
SECTION 3

SCENARIOS:

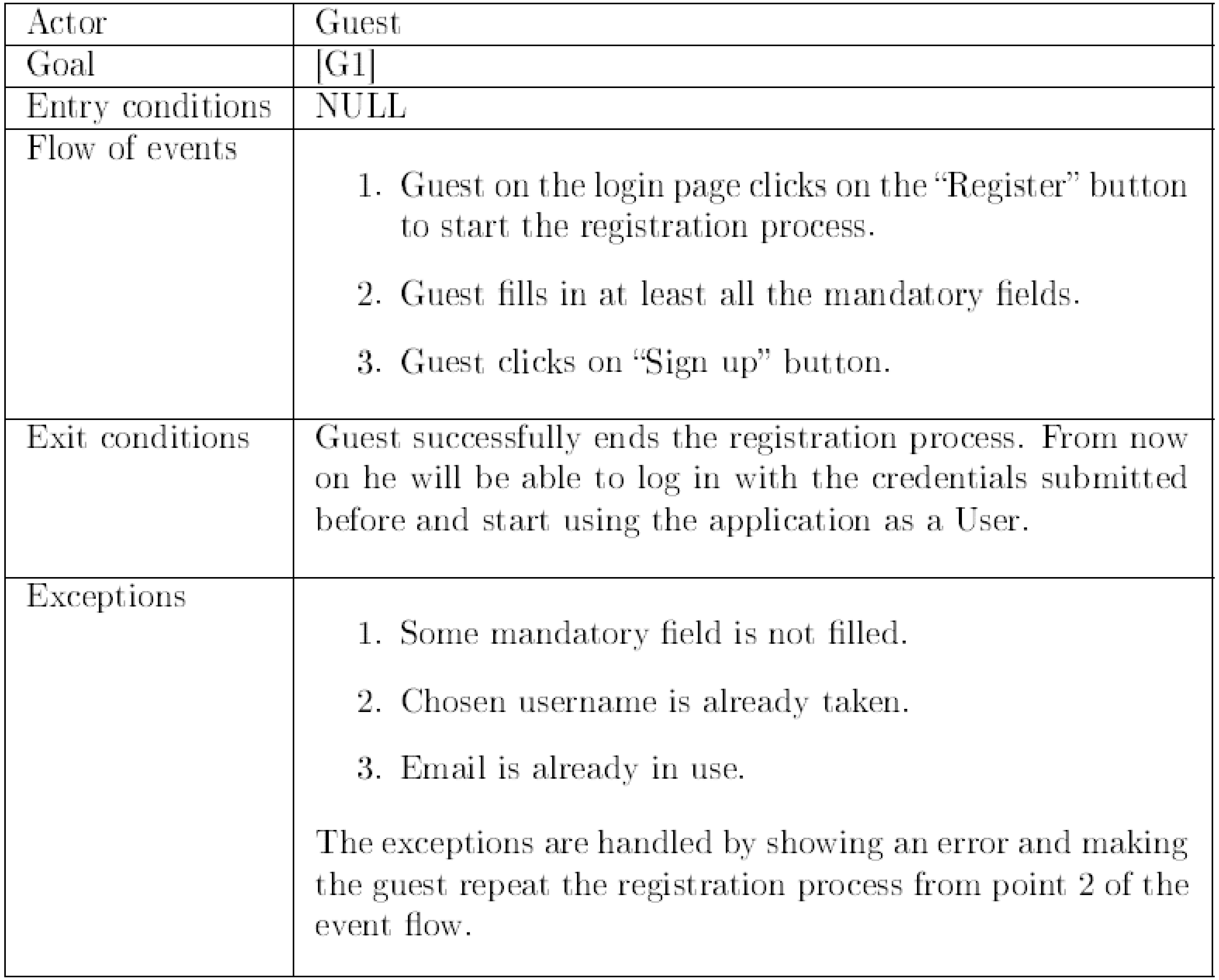
Scenario 4

Giuly decides to go to EXPO in the evening. She knows that the queue to enter is very long so she decides to go there at 5 pm in order to be in a good position when the gate will open. Giuly is scared to take the train to go there because she would be going back home alone. She decides to go there with a taxi and simply she opens myTaxi app and requests a taxi for a simple ride. Unfortunately, the system has detected that there is no one taxi available in her zone at the moment. Giuly immediately receives a notification that informs it of the problem inviting her to attend for the first available taxi.

SECTION 3 – UML MODELS USE CASE:

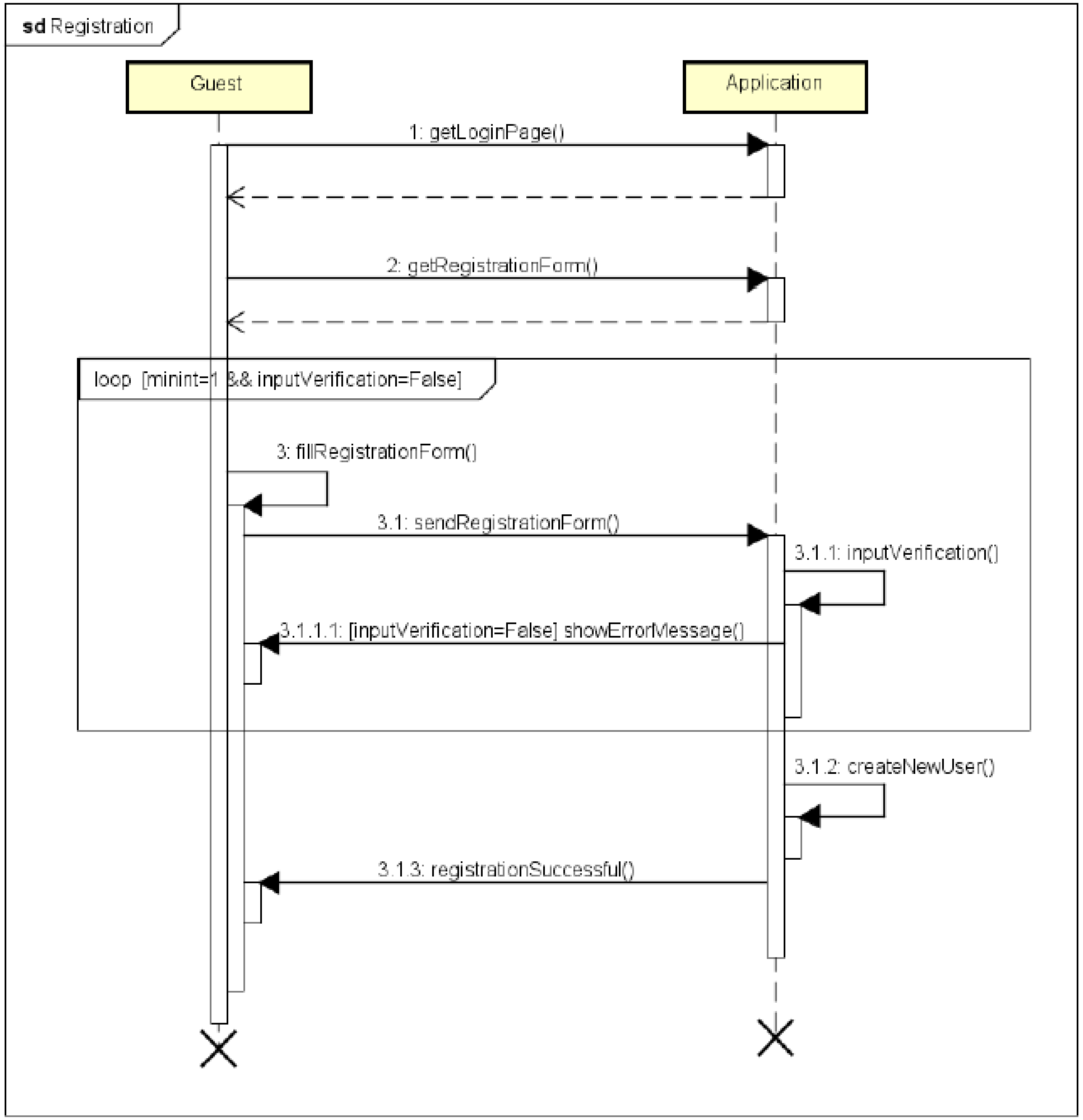


SECTION 3 – UML MODELS SIGN UP:



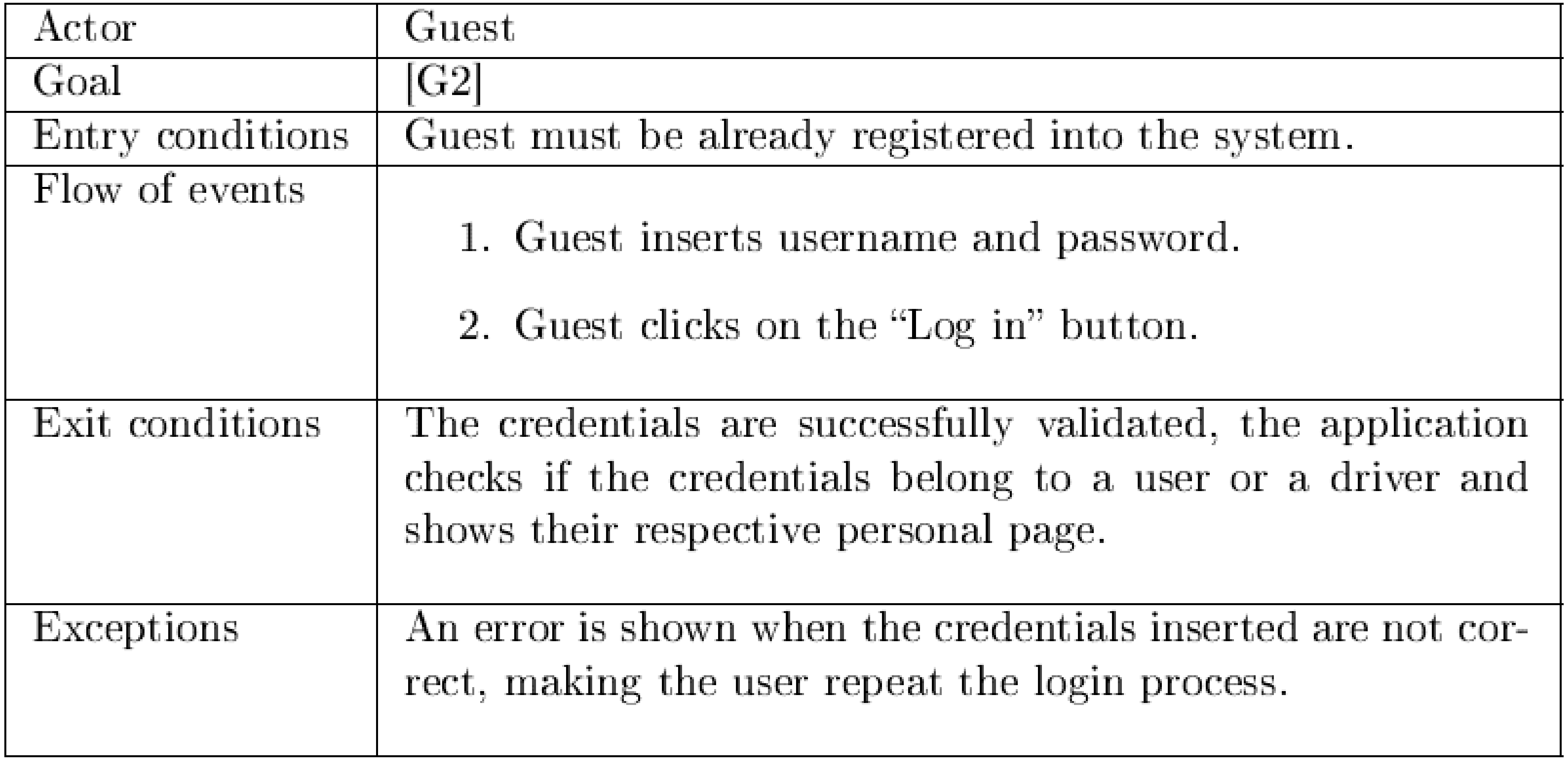
Actor	Guest
Goal	[G1]
Entry conditions	NULL
Flow of events	1. Guest on the login page clicks on the "Register" button to start the registration process. 2. Guest fills in at least all the mandatory fields. 3. Guest clicks on "Sign up" button.
Exit conditions	Guest successfully ends the registration process. From now on he will be able to log in with the credentials submitted before and start using the application as a User.
Exceptions	1. Some mandatory field is not filled. 2. Chosen username is already taken. 3. Email is already in use. The exceptions are handled by showing an error and making the guest repeat the registration process from point 2 of the event flow.

SECTION 3 – UML MODELS SIGN UP:


sd Registration
Guest
Application
1: getLoginPage()
2: getRegistrationForm()
loop [minint=1 && inputVerification=False]
3: fillRegistrationForm()
3.1: sendRegistrationForm()
3.1.1: inputVerification()
3.1.1.1: [inputVerification=False] showErrorMessage()
3.1.2: createNewUser()
3.1.3: registrationSuccessful()

SECTION 3 – UML MODELS

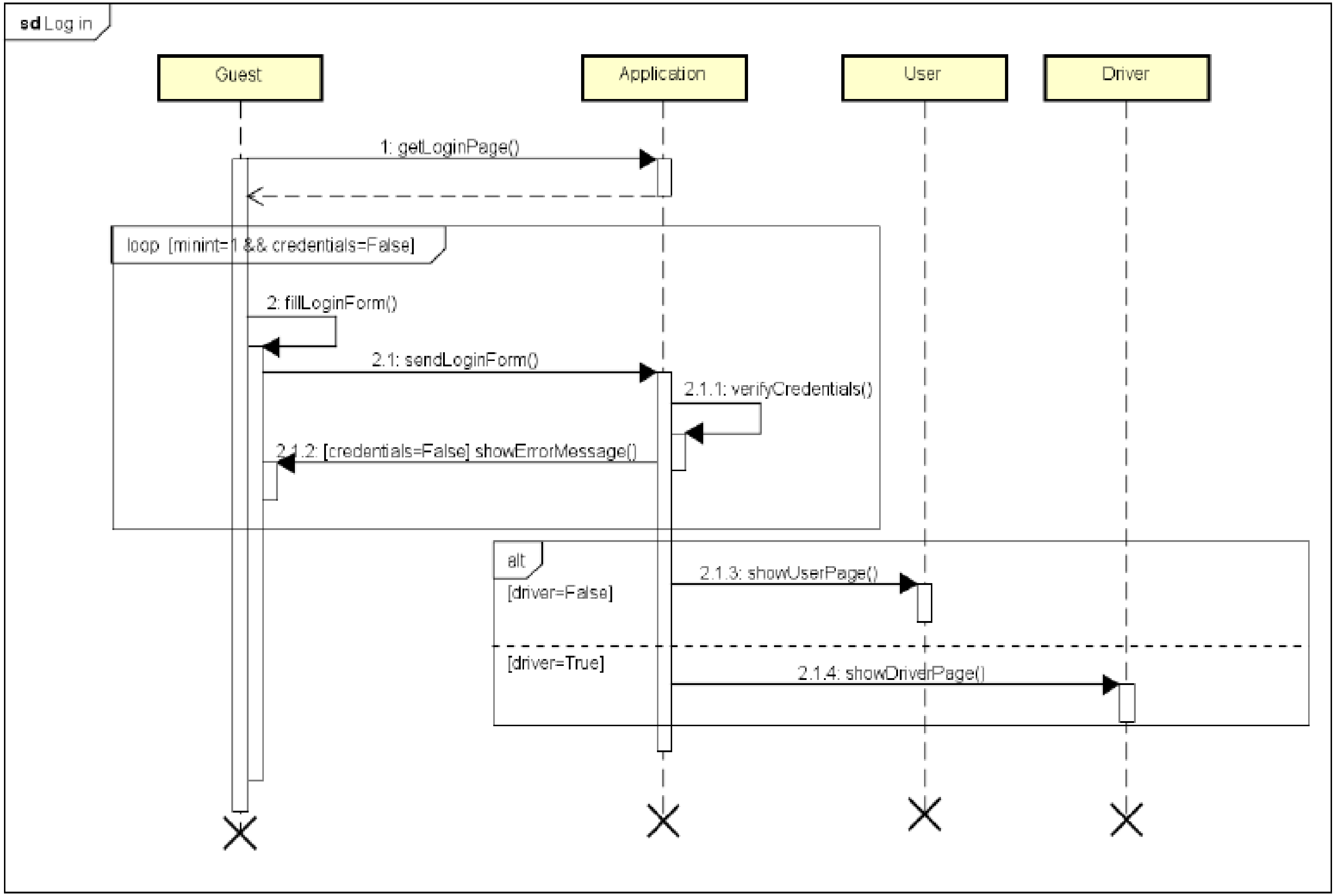
LOG IN:



Actor	Guest
Goal	[G2]
Entry conditions	Guest must be already registered into the system.
Flow of events	1. Guest inserts username and password. 2. Guest clicks on the “Log in” button.
Exit conditions	The credentials are successfully validated, the application checks if the credentials belong to a user or a driver and shows their respective personal page.
Exceptions	An error is shown when the credentials inserted are not correct, making the user repeat the login process.

SECTION 3 – UML MODELS

LOG IN:



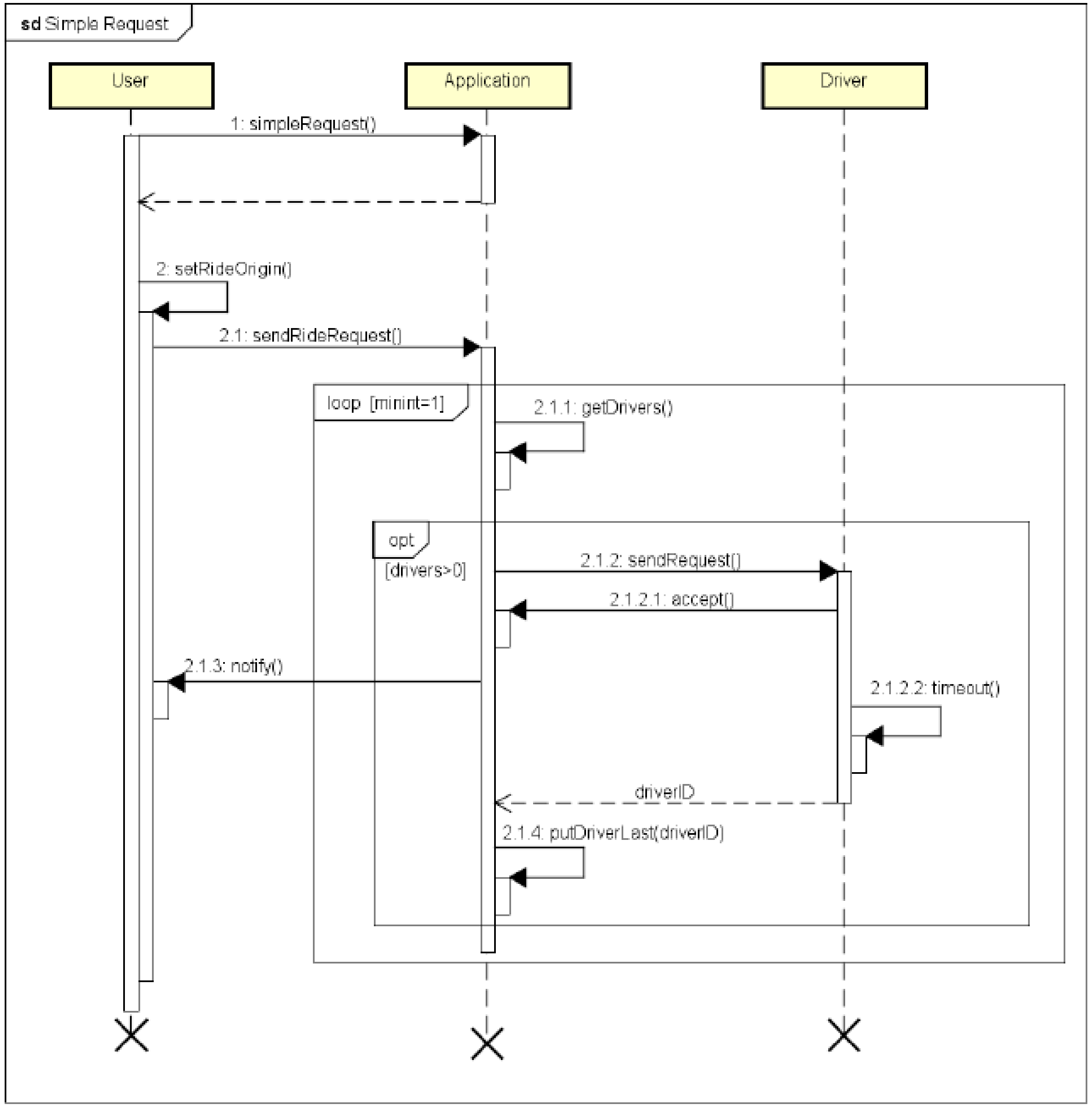
SECTION 3 – UML MODELS

MAKE SIMPLE REQUEST:

Actor	User
Goal	[G4]
Entry conditions	User must be logged in.
Flow of events	1. User clicks on the “Simple request” button.
Exit conditions	User successfully makes a request which is still pending until a driver accepts it.
Exceptions	None. If there’s no taxi, the system waits until one becomes available.

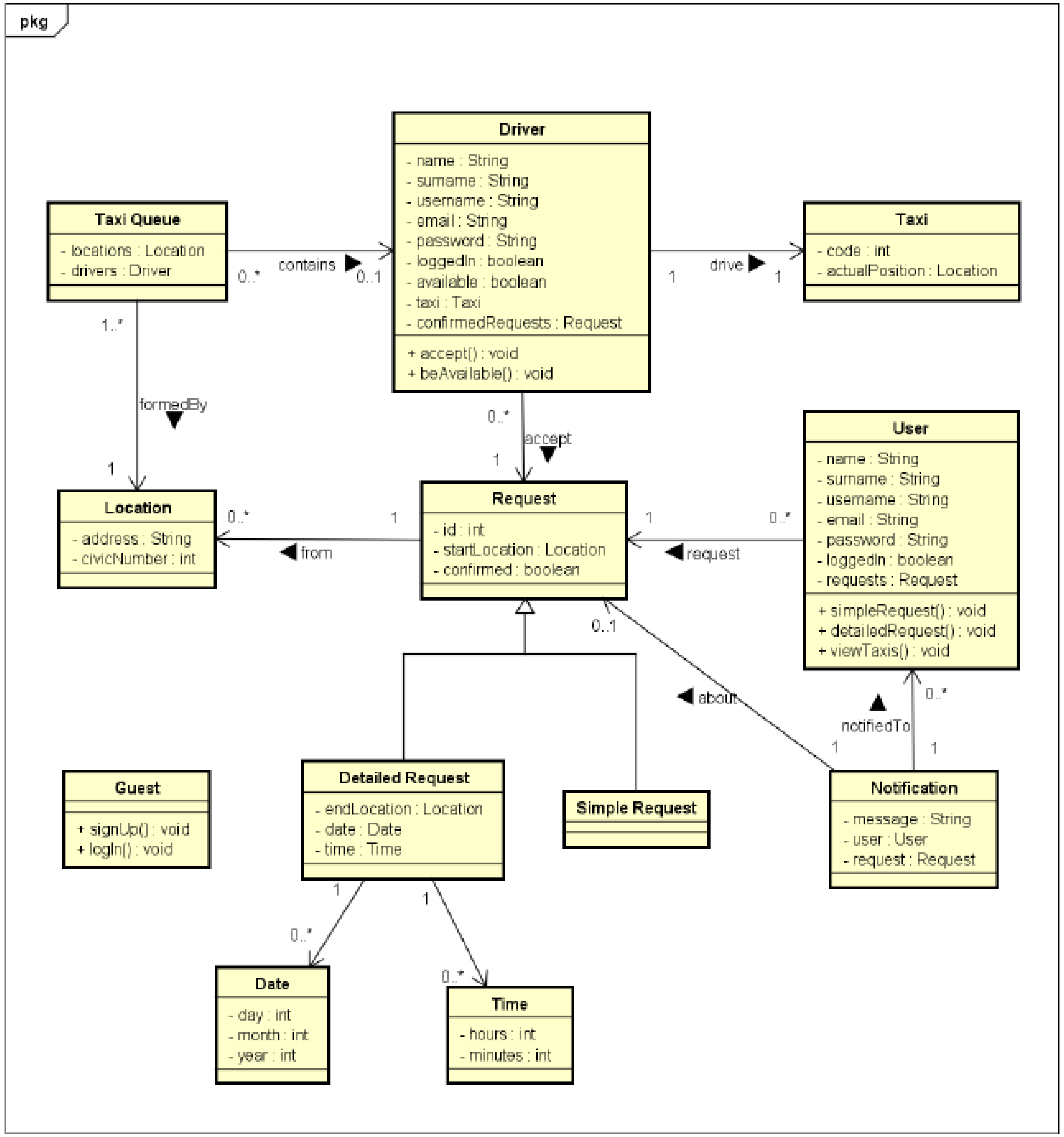
SECTION 3 – UML MODELS

MAKE SIMPLE REQUEST:

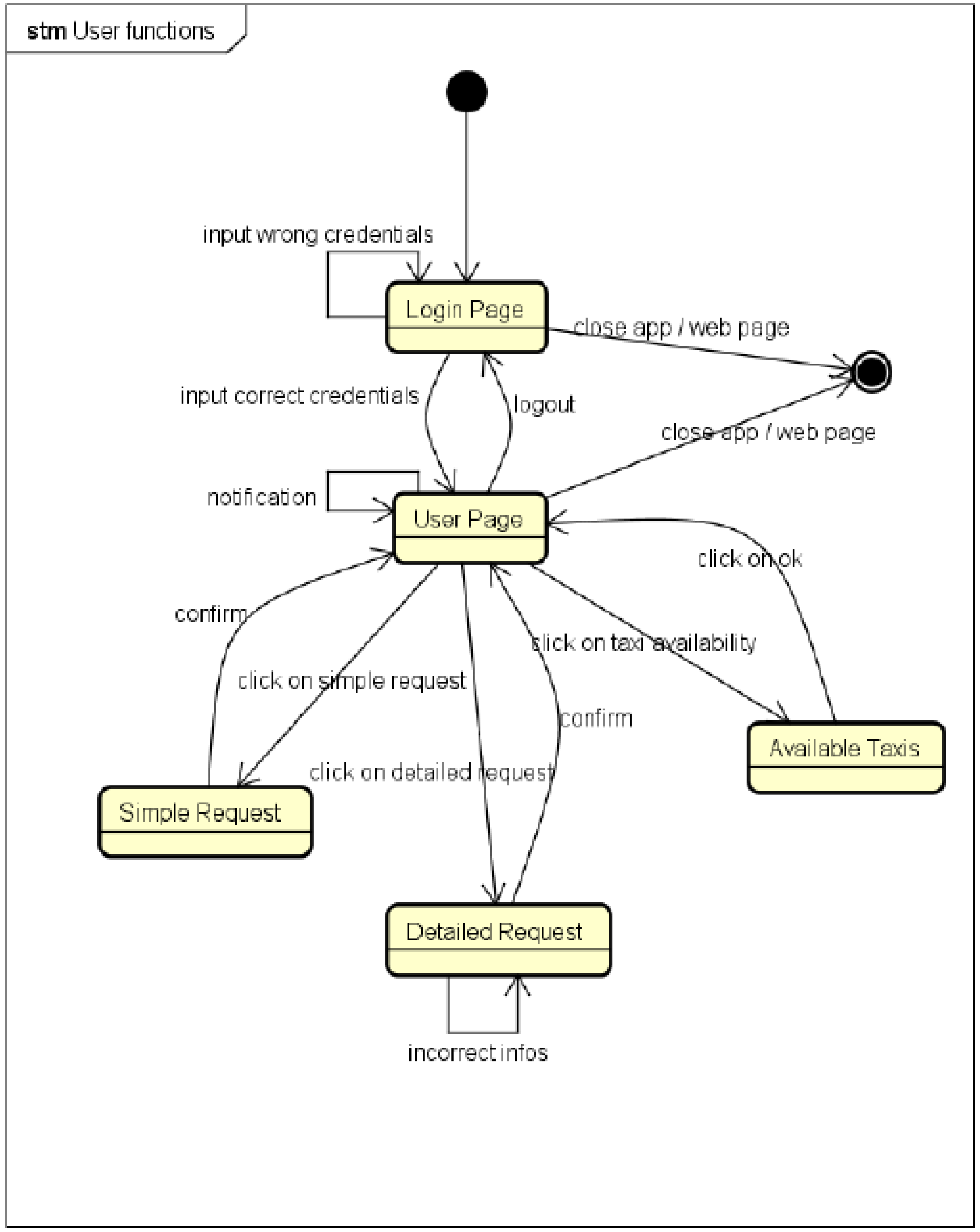


SECTION 3 UML MODELS

CLASS DIAGRAM:



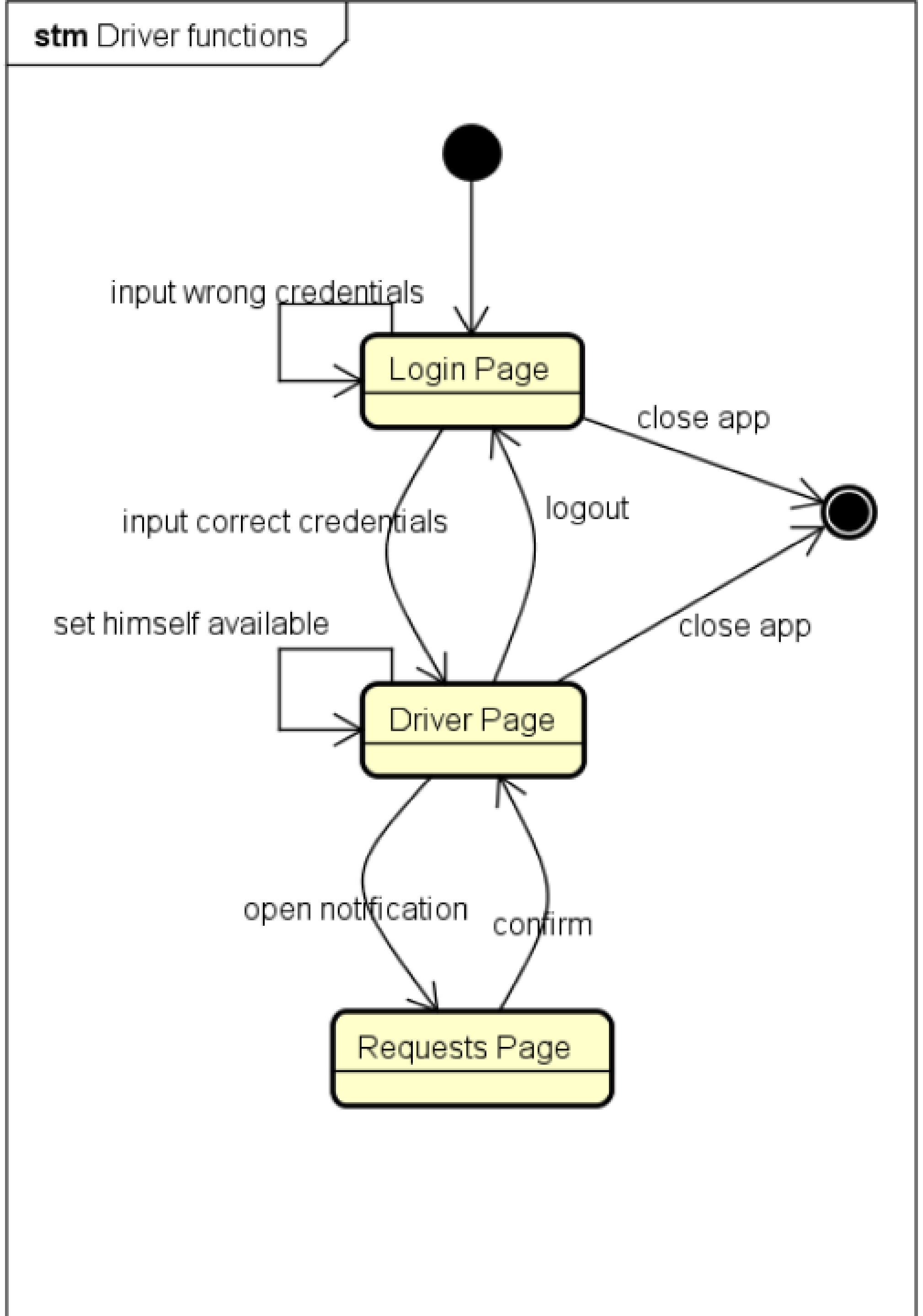
SECTION 3 UML MODELS



STATE MACHINE DIAGRAM

USER FUNCTIONS:

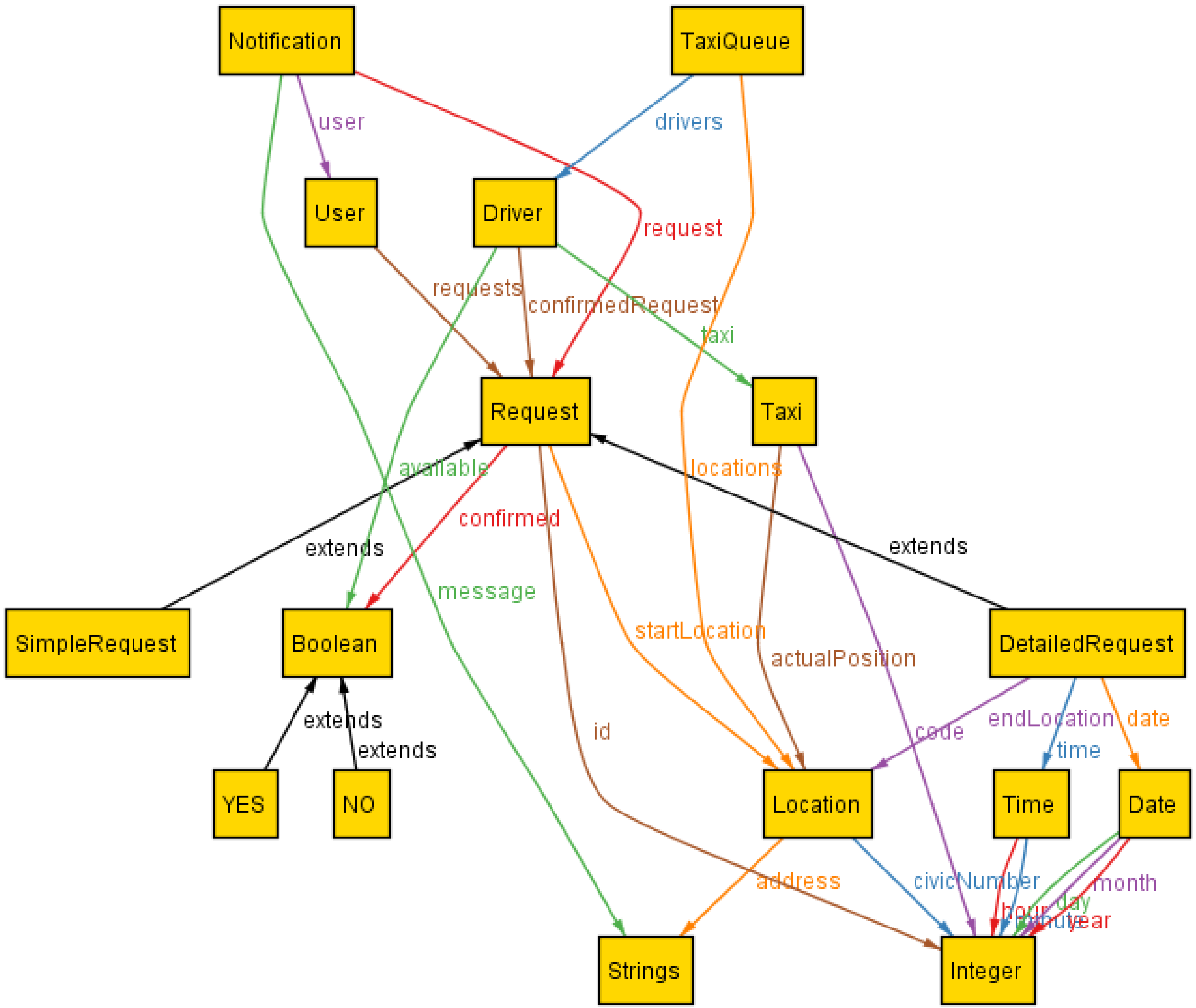
SECTION 3 UML MODELS



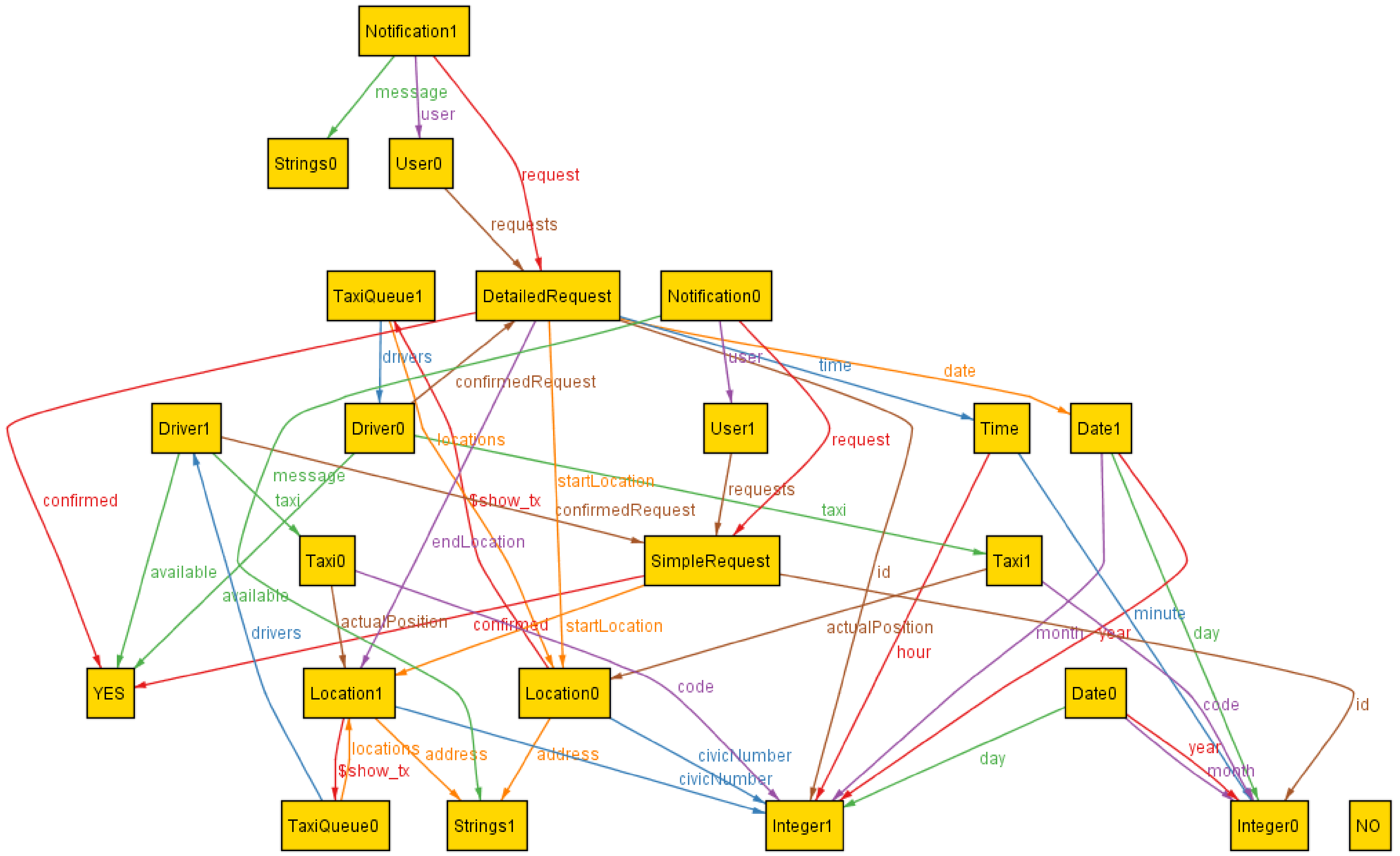
STATE MACHINE DIAGRAM

DRIVER FUNCTIONS:

ALLOY METAMODEL:



SECTION 4 EXAMPLE OF GENERATED WORLD:


Notification1
message
user
request
Strings0
User0
requests
TaxiQueue1
DetailedRequest
Notification0
drivers
confirmedRequest
user
time
date
Driver1
Driver0
User1
Time
Date1
locations
request
confirmed
message
startLocation
requests
taxi
$show_tx
confirmedRequest
taxi
Taxi0
SimpleRequest
Taxi1
endLocation
id
available
available
actualPosition
confirmed
startLocation
actualPosition
minute
month
year
day
drivers
hour
YES
Location1
Location0
code
Date0
code
id
locations
address
address
civicNumber
day
year
$show_tx
civicNumber
month
TaxiQueue0
Strings1
Integer1
Integer0
NO