**1.Introduction**

**1.1 Purpose**

This document is written following the standard IEEE 830-1998 (RASD) and helps all those who are in charge to develop the software and eventually those programmers who will have to make updates.

This document contains the requirements for the development of the software and some invented scenario as example of the functionalities included in the system.

This document can also be used during the discussion between customer and developer before or during the actual implementation to avoid misunderstadings.

**1.2 Scope**

The main software that will be developed is “myTaxiService”; the aim of this software is to allow people to use better the taxi service already existing in the city both via web application or via smartphone.

The application grants the client to request a taxi or make a reservation 2 hours before the meeting time and the system must ensure a fair management of the taxis in order to avoid some of them wasting time by do nothing. The client doesn’t need to register to request a taxi.

The taxi driver, to be inserted in the queue of the available taxis, must register by inserting name, surname, e-mail address, the number of the license, username and a password; after this, he can login with a username and a password and start working. Eventually, in case of need, the taxi driver can set his state to “Busy” so that the system doesn’t forward requests to him.

In order to improve service and help for future changes, the software allow technicians to develop additional services via programmatic interfaces.

**1.3 Definitions,acronyms and abbrevations**

* Client: whoever needs the taxi service
* Taxi driver: a person whose job is to take people in a car to the place they want to go to in return for money
* Technicians: those people who add additional service

**1.4 References**

* Specification Documents: MyTaxiServices
* IEEE std 830-1998

**1.5 Overview**

This document is composed by 4 main part:

* Part 1 : Intoduction - In this part you can find a initial presentation of this document and all introductory information about the software.
* Part 2 : Overall Description - the second part contains more accurate information about the software.All about constraints,assumptions,interfaces and also information about the users
* Part 3 : Specific requirements - here there are all the information concerning the requirement of the project
* Part 4 : Appendixes - all the supporting information are included in this part

**2.Overall Description**

**2.1 Product perspective**

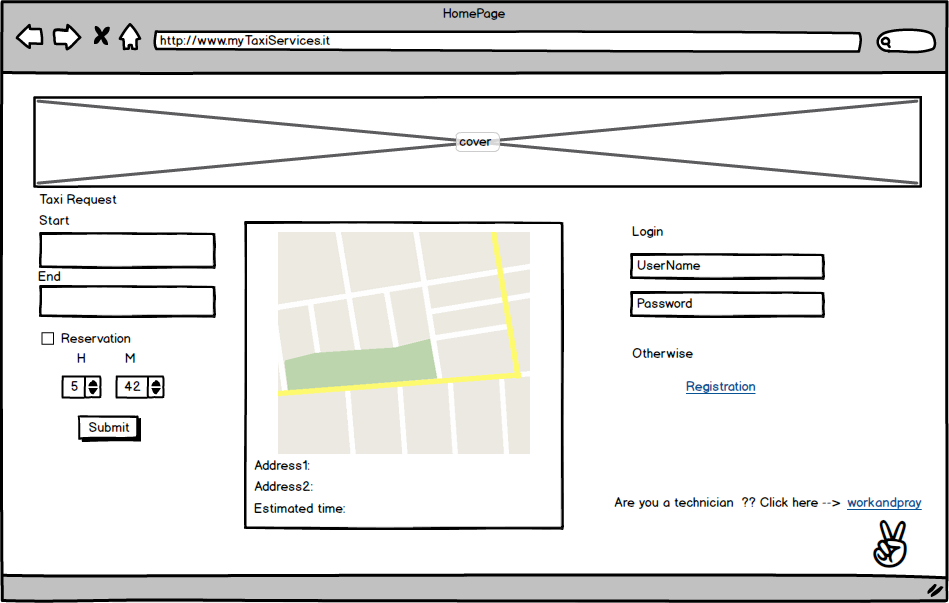
The system consist in a web page and a smartphone app, so it must be available for any browser and an application that works on Android, iOS and Windows phone operative system; this also implies that every taxi driver must have an appropriate smartphone.

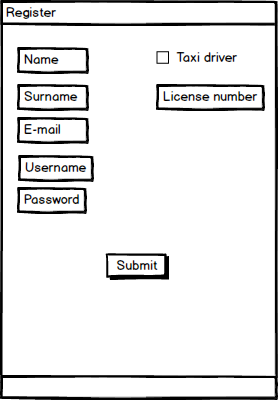
The system is developed without having any kind of support/interaction with other systems, so is totally independentg. Due the fact that the application can be easily updated with the programmatic interface, we don’t exclude an interaction in the future with other systems.

**2.1.1 User interfaces**

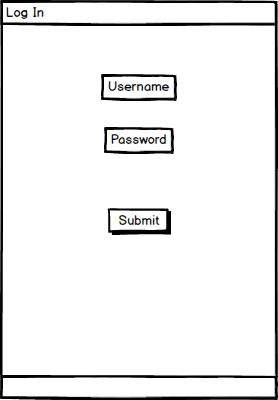
**Homepage**

This is the Homepage of the web application.Here a user can request a ride (normal or reservation)There is the input form for the login and,if you want you there is a link for registration and another one for future implementation of the programmatic interfaces(now you can see only few news about coming idea)



**Registration Page**

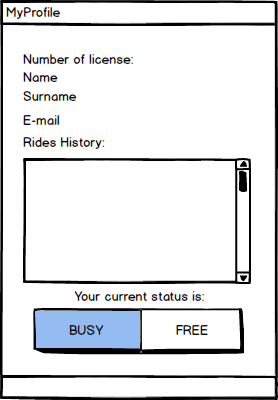
Here there is a example of the registration page (only for the mobile app) This page is equal both for customer and taxi driver.Only taxi driver can fill the field about the license(there will be a control about the validity of the field).The registration is not mandatory to request a ride ,is only additional feature for further funcionalities.



**2.1.1.1 Taxi driver interface**

**Log In:**

This is a standard page for the log in that request username and password to let a registered user to view it’s own profile



**My Profile**

this page contains all the informations about the driver, plus a history of the rides he took.

The buttons below set the state of the taxi and are mutually exclusive (of course can’t be busy and free at the same time).



**Incoming Request**

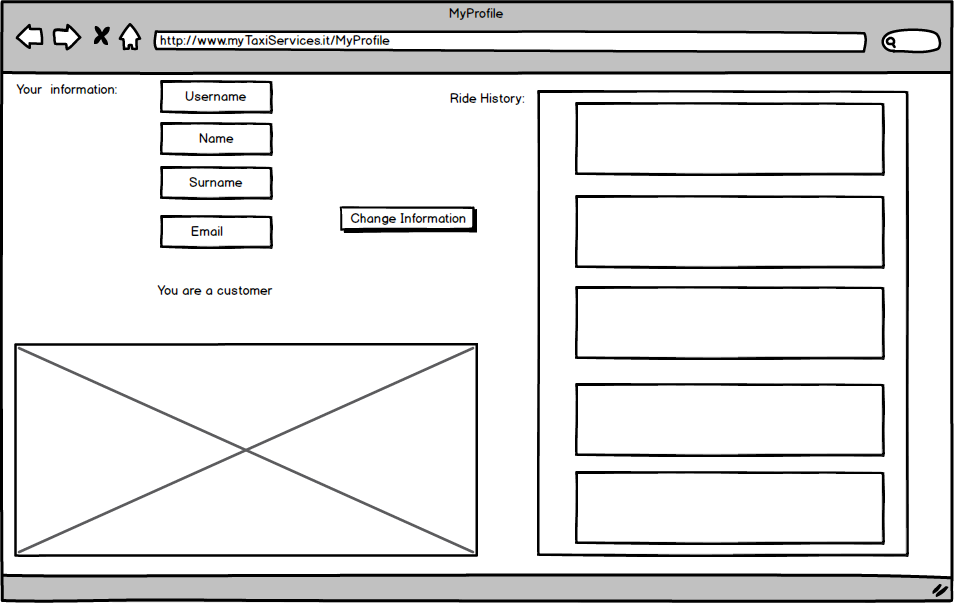
Whenever the system assign a ride to the taxi, both for a simple ride or a reservation, the app shows automatically this window showing the informations about the trip: the taxi driver must send a response within 2 minutes otherwise the window disappear, like he pressed “no”.

If the taxi driver accept the request, the window disappear and goes automatically to “MyProfile” with the status set on “busy”

**2.1.1.2 User Interface**

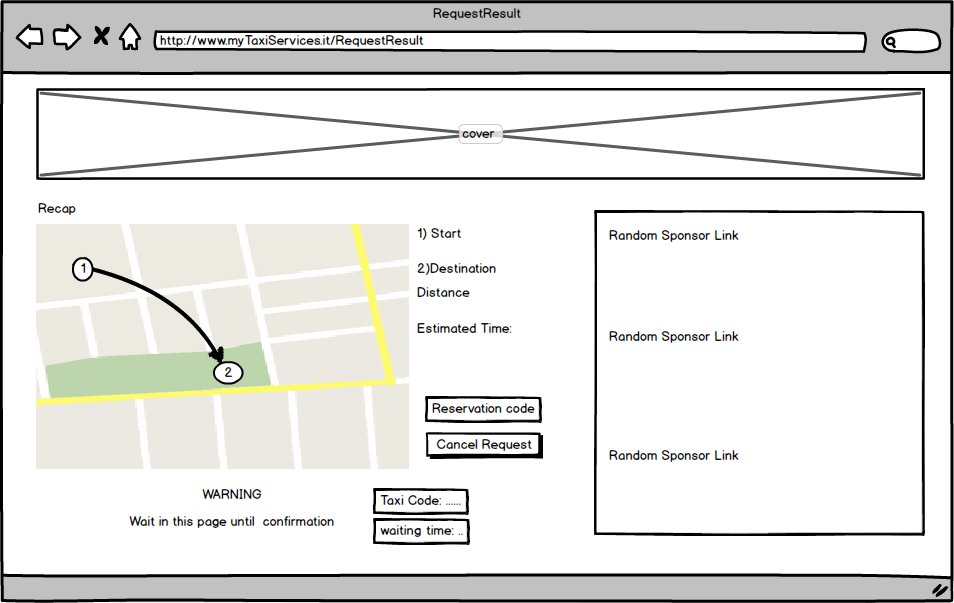
**MyProfile**

Below is represented the profile page of a customer profile (only web app ) which contains all the information.There is also a ride history.In this page a user can also change his information.



**RequestResult**

This page show all information concerning the just made request. After few minutes a customer can see the code of the taxi and the waiting time. If the request is a reservation ,the system only shows a confirmation message and a taxi will be allocated 10 minutes before the meeting time. If a user wants to cancel the reservation can press the cancel request button.



**2.1.2 Hardware interfaces**

there aren’t any kind of support for hardware interfaces

**2.1.3 API interfaces**

We use a map indication API to show the route. In addition there are information about the start and destination addresses and concerning the estimated time of the trip.

**2.1.3 Software interfaces**

* Database Management System(DBMS)

-Name: MySQL

-Version: 5.7

-Source: <http://www.mysql.it/>

* Application server:

– Name: DigitalOcean

– Version: 6.1.

– Source:<https://cloud.digitalocean.com>

**2.1.4 Communication interfaces**

We use theTCP protocol for security reasons.

**2.1.5 Memory**

No memory constraints are required.

**2.2 Goals**

Below are listed the main goal of myTaxiService

1. Allow guest to register the application in order to receive upcoming features
2. Allow user/guest to make a reservation
3. allow user/guest to delete a reservation
4. Allow user/guest to take a ride
5. Allow a taxi driver to accept or decline a request
6. Allow a taxi driver to set its own state
7. Allow the system to allocate correctly a taxi based on the currently location
8. Allow user to change his information

**2.3 User Characteristics**

The audience of this application is anyone who needs a taxi service so the user part doesn’t give any constraint; taxi drivers, the only thing they need to do is to have the license (otherwise they also cannot register to the website).

The technician instead must be a person with a high knowledge about computer science and programming languages used to develop apps for smartphone in general to avoid failure in the system and the service that doesn’t works

**2.4 Constraints**

**2.4.1 Regulatory Policies**

myTaxiService doesn’t meet any regulatory policies

**2.4.2 Hardware Limitation**

myTaxiService has only one kind of hardware limitations : a taxi driver must have a smatphone.

The system must use a database to store all informations about taxi drivers and registered users

**2.4.2 Interfaces to other applications**

myTaxiService doesn’t meet any **I**nterfaces to other applications

**2.4.3 Parallel operation**

myTaxiService must handle all the parallel requests from multiple users and doesn’t lost any kind of data

**2.5 Assumptions**

1. A ride can end in any zone
2. each taxi can stay in one and only one zone
3. A client always receive a response
4. A taxi cannot have any kind of low functions
5. A taxi is never late for the meeting with the customer
6. A taxi can take care about only one request at a time
7. GPS signal is always on work, so we know each instant where each taxi is
8. At least one taxi in the all City is free
9. We already have a database with all taxis and the corresponding license

**3. Requirements**

**3.1 Functional Requirements**

1. Allow guest to register the application in order to receive upcoming features

* The system must have a database to store all registred users and provide a sign up functionality
* A person who already registered, cannot register again unless previous unregistration

1. Allow user/guest to make a reservation

* The system provide a page to make a reservation to any user
* The guest have to insert the information about the trip: starting point, destination, meeting time
* All these informations must be inserted 2 hours before the meeting time; the system send a message either the reservation is done or not.

1. Allow user/guest to delete a reservation

* The system also have to provide the possibility to cancel an existing reservation from web or app; in this last case, no taxi will be allocated

1. Allow user/guest to take a ride

* The system allow any user (from web site or smartphone app) to send a request for a taxi
* After a request is make, the system have to send a message to the taxi at the top of the queue

1. Allow a taxi driver to accept or decline a request

* The app provide the choice to the taxi to accept or refuse a request for allocation
* If the request is refused, a message must be sent to the next taxi driver
* If the request is accepted, the taxi driver’s status is automatically set on “Busy”

1. Allow a taxi driver to set its own state

* The taxi driver must have the possibility to set his own state in case of need from his profile
* The state of the taxi must be mutually exclusive

1. Allow the system to allocate correctly a taxi based on the currently location

* The system automatically gets GPS informations of all taxis and add them to the queue of the correct zone

1. Allow user to change his information

* In the personal page there will be a link to change informations
* Updated informations are stored in the database

**3.2 The World and the Machine**

We use “The World and The Machine” model by M. Jackson & P. Zave to understand

the different kind of environments.In the World section we have all the “real” events and entities that interact with the application.The Machine domain rappresents the entities to be developed .In the intersection are rappresented the shared phenomenal,all world informations that are managed directly by the application.











**3.3 Scenario**

**SCENARIO 1**

Sean is a business man that needs to go to an important meeting but

he is on late so need to call a taxi

The system answer the client that a taxi will come but Sean is nervous and is continuously sending requests that are ignored by the system; after 5 minutes a taxi arrives and brings Sean to the meeting.

**SCENARIO 2**

it’s 17:00 Andrew have to be to the cinema in 1 hour and half and decide to get a reservation for a taxi; when he try to get the reservation for the 18:00, the system sends an error saying that a reservation must be done 2 hours before the meeting time.

**SCENARIO 3**

Jennifer request a taxi in the morning to join a party of her best friend that takes in the evening: the reservation goes through the system that add this reservation in the database. As requirement, 10 minutes before the meeting time, the system search for a free taxi according to the policy specified in the requirements: the taxi driver Joshua confirms the availability and its taxi is allocated by the system for the reservation.

**SCENARIO 4**

Sharon usually uses a taxi for moving around the city. After she discovers this new service, she decide to register in order to see the news of the upcoming features (taxi sharing,etc).

She fills the fields with the requested information (email,payment methods,etc) and at the end she sees the confirmation message.

**SCENARIO 5**

Paul needs to go to a barbecue in the evening so decide to take a reservation in the morning but in the afternoon, before the taxi is allocated, it starts to rain so the barbecue is canceled and Paul doesn’t need anymore a taxi and cancel its reservation; the system receive the request to cancel the reservation and the procedure commit.

In all of this time, no taxi is bothered.

**SCENARIO 6**

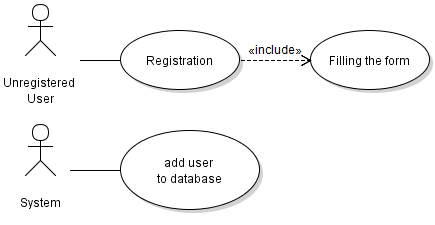
Goffredo is a taxi driver and today worked a lot, so to take a break and get gas ,sets the status on “busy” and go to a petrol pump to relax a bit.

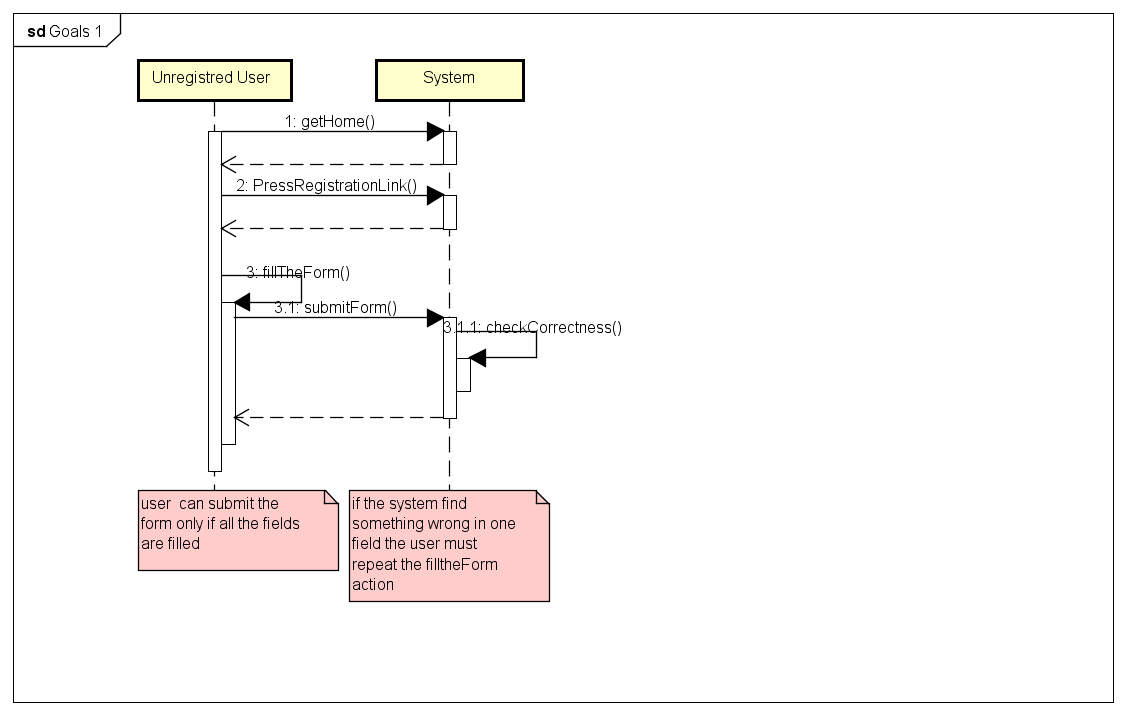
After 1 hour of rest he reset the status on “free” and after a while recieves the request for a ride.

**3.4 UML models**

**Goal 1**

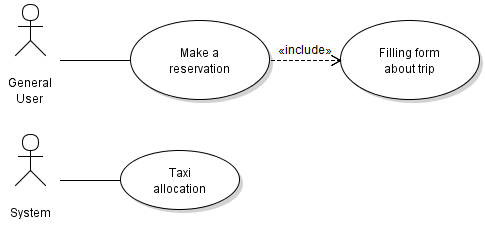
|  |  |
| --- | --- |
| Actor | Unregistered user |
| Goal | Allow guest to register the application in order to receive upcoming features |
| Input condition | NULL |
| event flow | 1. The user enter in the page for registration; 2. The user fills the form and press “submit” |
| output condition | All information are correctly inserted |
| Exceptions | If the form is not filled correctly, send an error message and start from point 2 |

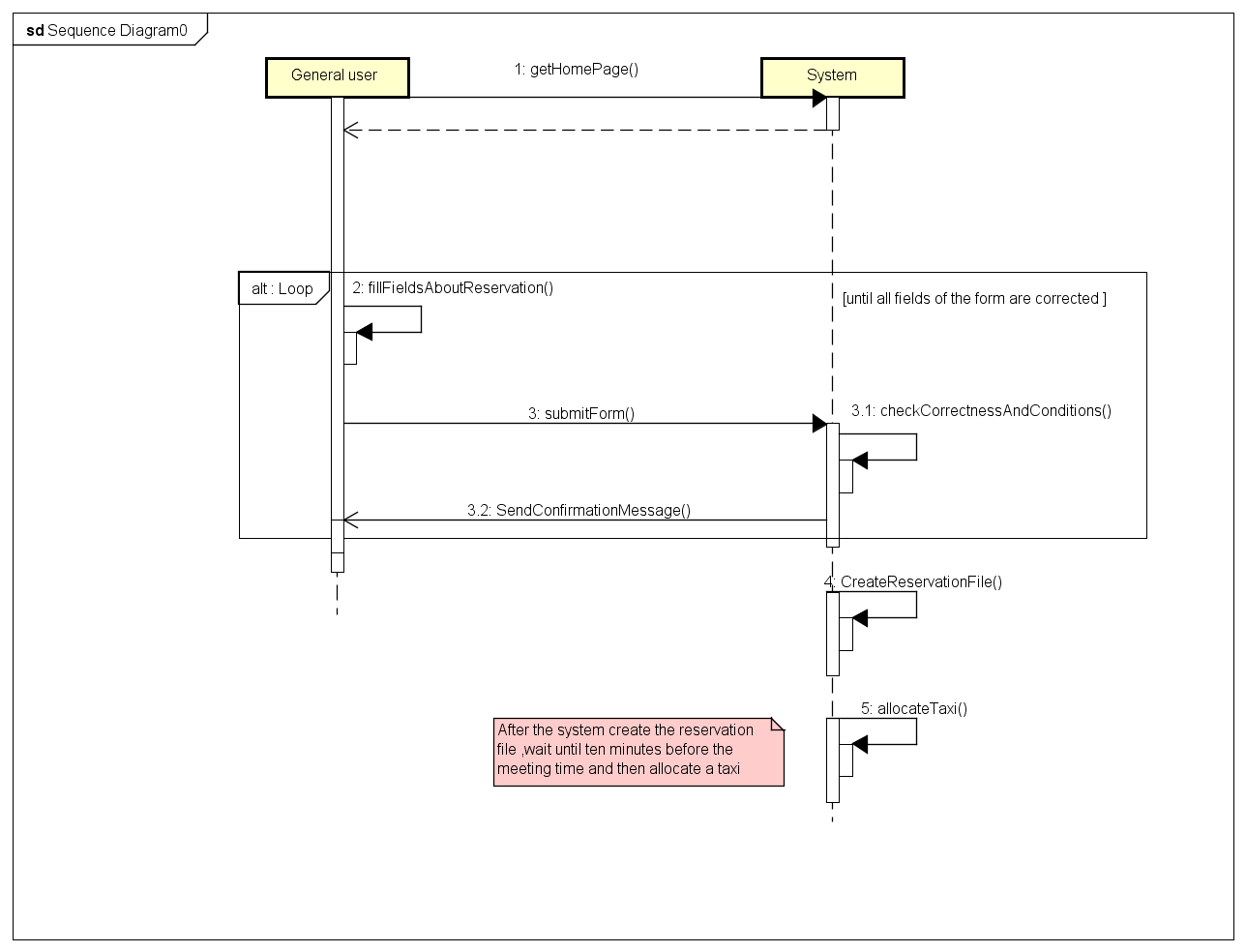




**Goal 2**

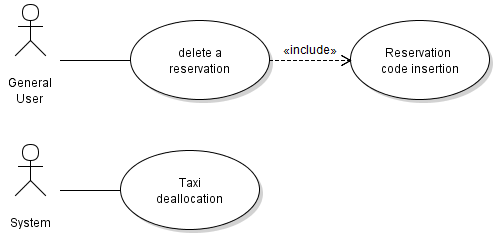
|  |  |
| --- | --- |
| Actor | General user |
| Goal | Allow user/guest to make a reservation |
| Input condition | NULL |
| event flow | 1. The user enters in the web site or application 2. The user insert departure point, departure schedule and destination 3. The user send request |
| output condition | All information are correctly inserted and the user receive the number of the reservation. |
| Exceptions | 1. If the form is not filled correctly, send an error message and start from point 2. 2. If the form is not filled 2 hours before the departure schedule, send an error message. |





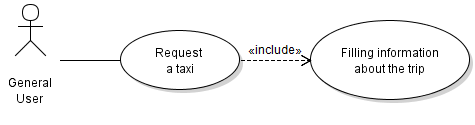
**Goal3**

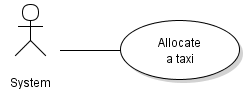
|  |  |
| --- | --- |
| Actor | General user |
| Goal | Allow user/guest to delete a reservation |
| Input condition | The user has made a reservation |
| event flow | 1. The user access the page to delete reservation 2. The user insert the code of reservation 3. The user press the “Confirm” button |
| output condition | The system remove the reservation and doesn’t allocate/deallocate a taxi |
| Exceptions | code reservation is wrong, restart from point 2 |



**Goal 4**

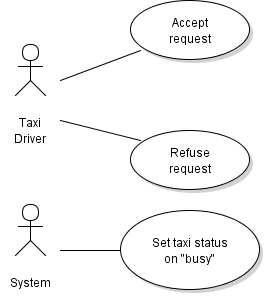
|  |  |
| --- | --- |
| Actor | General user |
| Goal | Allow user/guest to take a ride |
| Input condition | NULL |
| event flow | 1. The user access the application 2. The user insert departure and destination points 3. The user press “Confirm” |
| output condition | The system allocate a taxi and send a confirmation message to the user |
| Exceptions | If the user insert a wrong address both for departure or destination, the system send an error message. |





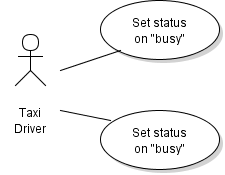
**Goal 5**

|  |  |
| --- | --- |
| Actor | Taxi Driver |
| Goal | Allow a taxi driver to answer a request |
| Input condition | A request arrives to the taxi driver |
| event flow | 1. The taxi driver receive the request 2. The taxi driver choose if accept or refuse the request |
| output condition | The taxi driver make his choice |
| Exceptions | If taxi driver doesn’t respond within 2 minutes, the request is forwarded |



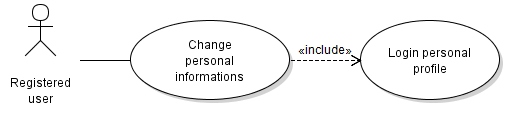
**Goal 6**

|  |  |
| --- | --- |
| Actor | General user |
| Goal | Allow taxi driver to set his own state |
| Input condition | NULL |
| event flow | 1. Taxi driver access his personal page 2. Taxi driver set his state |
| output condition | The status is changed |
| Exceptions | NULL |



**Goal 7**

|  |  |
| --- | --- |
| Actor | Registered User |
| Goal | Allow user to change his information |
| Input condition | NULL |
| event flow | 1. The user log in the application 2. The user press the “change information” button 3. user fills the fields that want to change 4. user confirm the changes |
| output condition | user press the save changes button |
| Exceptions | 1. if user isn’t registred must make a registration first 2. if user don’t fill correctly the fields must repeat the third point until all is corrected |



**3.6 Non Functional Requirements**

**3.6.1 Performance Requirements**

There aren’t any kind of constraints concerning the performance.We assume the system is powerful enough to handle all the requests.

**3.6.2 Design Constraints**

No design constraint are required.

**3.6.3 Software System Attributes**

**3.6.3.1 Availability**

User can access to the application any time they need. The system use a dedicated server with high level of performances to guarantee this attribute.

**3.6.3.2 Maintainability**

For the future implementation a full documentation will be avaible.The documentation will contain a full history of all version of the sofware.At this moment there are a plans for future implementations like taxi sharing and we let tachnicians to be updated about any kind of news.

**3.6.3.3 Portability**

The application can be used on any OS concerning the web application. For the mobile app a non-obsolete smartphone is needed both for taxi driver and customer.

**3.6.4 Security**

**3.6.4.1 External Interface Side**

To ensure an availability of the service 24/7 er decide to introduce the basic forms of security: Username and password cannot contain some special characters (like parenthesis or <>); we will use a captcha code before sending a request to avoid DDoS attacks

**3.6.4.2 Application Side**

The taxi driver interface, to avoid MITM attacks, will be implemented with the SSL protocol

**3.6.4.3 Server Side**

A simple structure with firewalls with basic access rules between Web server, Application server, Database and the internet can be easily implemented; the Web Server will be placed in the Demilitarized Zone while the Database will be in the zone with the most strict access rules

