POLITECNICO
MILANO 1863

# CLup

# Requirement Analysis and Specification Document

Simone Abelli
Stefano Azzone

| | |
|---|---|
| **Deliverable:** | RASD |
| **Title:** | Requirement Analysis and Verification Document |
| **Authors:** | Simone Abelli, Stefano Azzone |
| **Version:** | 1.0 |
| **Date:** | 16 December 2020 |
| **Download page:** | https://github.com/StefanoAzzone/AbelliAzzone |
| **Copyright:** | Copyright © 2020, Simone Abelli, Stefano Azzone – All rights reserved |

# Contents



# 1 Introduction

## 1.1 Purpose

The coronavirus emergency has put a strain on society on many levels, in particular, grocery shopping can become a challenge in the presence of such strict rules: supermarkets need to restrict access to their stores to avoid having crowds inside and long lines outside. The goal of this project is to develop an easy-to-use application that, on the one side, allows store managers to regulate the influx of people in the building and, on the other side, saves people from having to line up and stand outside of stores for hours on end.

The application will allow customers to "line up" (i.e., retrieve a number) from their home, and then wait until their number is called (or is close to being called) to approach the store. In addition, the application could be used to generate QR codes that would be scanned upon entering the store, thus allowing store managers to monitor entrances.

The system must attain the following goals:

G1 CLup should allow customers to queue up remotely

G2 CLup should allow customers to queue up on premise

G3 CLup should allow customers to book future visits to stores

G4 CLup should allow store owners to regulate the maximum number of customers in their stores

G5 CLup should provide the customer with an estimate of waiting time

G6 CLup should alert the customers when it is time to get to the shop taking into account travel time

## 1.2 Scope

The system to be allows to avoid creating queues in front of stores. This is accomplished by enabling the customers to queue up remotely. Moreover, the shop owners can oversee customers entering and exiting stores.
The system offers the following functionalities:

F1 it allows customers to line up remotely or on premise

F2 it allows customers that requested to queue up, to cancel their request

F3 it allows customers whose position in queue allows it to enter and exit the store

F4 it schedules customers in order to minimize overcrowding inside and outside of the store

F5 it alerts customers when they should head to the store

F6 it allows customers to book a visit and optionally specify duration and desired categories of products

F7 it allows store owners to monitor flux of people inside the store and set how many customers can be simultaneously inside

F8 it allows store owners to register stores

F9 it builds statistics on entrance and exit data, and preferred categories

User can access the system using a phone application provided that they own a smartphone with an internet connection. This application allows registered users to easily access the functionality of the system.
Moreover, for those who do not have access to the required technology or don't want to use the phone application, there is also a web app. Therefore, users will be able to log in and perform the same operations of the phone application using an internet browser.
Finally the customers of the store will also have the possibility to create reservations on premise: every shop will be equipped with a device to print tickets in order to give every client a very easy way to line up. Of course customers that wants to queue up "on the spot" must have a means of identification (e.g. document), in order to prevent fake reservations that would slow down the queue. For the same reason users that use the phone application or the web app must log in to identify themselves.
Only customer with a valid reservation (i.e it is their turn) will be allowed to enter the store; to achieve this, each store must have a device (e.g. a QR reader) to every entrance, and clients must validate their reservation before being admitted inside. They should also do the same to exit, in order to monitor the store's occupation.

### 1.2.1 World Phenomena

WP1 Customer reaches a store by foot

WP2 Customer reaches the store by car

WP3 Customer reaches the store by bike

WP4 Customer enters or exits a store

WP5 Store owner controls current number of customers in one of his stores

WP6 Customer buys products

### 1.2.2 Shared Phenomena

SP1 Customer queues up

SP2 Customer is identified

SP3 Turnstiles lock and unlock

SP4 Customer is alerted

SP5 Customer books a visit to a store

SP6 Printer prints a ticket

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

| | |
|---|---|
| Reservation | Virtual or physical artifact used to identify the position of a customer in a queue |
| Queue up | Customers are lined up in a FIFO queue |
| Enqueued | A customer is enqueued when he has provided the system with a means of identification and requested a reservation |
| Authorized | A customer is authorized when he has been enqueued and is allowed temporary access to the store. |
| Occupation | Number of customers currently present in the store |
| Printer | Device that can read a social security card and print tickets that contains a progressive number and an estimate of the waiting time. |

### 1.3.2 Acronyms

| | |
|---|---|
| RASD | Requirement Analysis and Specification Document |
| GPS | Global Positioning System |
| S2B | Software to be |
| UI | User Interface |
| FIFO | First in first out |

### 1.3.3 Abbreviations

| | |
|---|---|
| Gn | Goal number n |
| Rn | Requirement number n |
| Dn | Domain Assumption number n |
| WPn | World Phenomenon number n |
| SPn | Shared Phenomenon number n |

## 1.4 Revision history

Version 1.0: 16/12/2020
Version 1.1: 7/1/2021 fixed a requirement and minor changes

## 1.5 Reference Documents

1. IEEE Std 830-1998 Recommended Practice for Software Requirements Specifications
2. Specification Document: R&DD Assignment A.Y. 2020/2021
3. uml-diagrams.org

## 1.6 Document Structure

- Chapter 1: gives an introduction about the project, describing the purpose of the system informally and defining its scope, its main goals, world and shared phenomena. Moreover this section contains specifications such as the definitions, acronyms, abbreviation, revision history of the document and the references.

- Chapter 2: contains the overall description of the project, with a more in-depth look at its functionalities. Here are identified the main actors involved in the application's usage lifecycle, some scenarios useful to identify specific cases in which the application can be utilized, and all the necessary domain assumptions, dependencies and constraints. This section also provides a class diagram, which aid to better understand the general structure of the project, and some state diagrams, to make the evolution of the crucial objects clear.
- Chapter 3: This section contains the core of the document: first it presents the interface requirement including user, hardware, software and communication interfaces. Then it offers the specification and the description of all the functional requirements necessary in order to reach the goals; is also provided a list of use cases, with their corresponding sequence diagrams and their mapping on the requirements. Finally non-functional requirements are defined, including performance, design and the software systems attributes.
- Chapter 4: includes the alloy code and the corresponding metamodels generated from it, in order to show how the project has been modeled and represented through the language.
- Chapter 5: shows the effort which each member of the group spent working on the project.
- Chapter 6: includes the reference documents.

# 2 Overall Description

## 2.1 Product perspective

The S2B will be used by two kinds of users: customers and store owners. The system will provide customers with a simple UI, in order to be accessible for all demographics. To use the software the customers need either a smartphone or a computer (a printer may be required in this case to print a QR code on paper) with a working internet connection. The software will use a location API (e.g. Google Maps) to pinpoint the location of the customers. To allow on premise reservations the store must install a monitor and a ticket printer outside of the store, and connect them to the system.
The system offers to each of the users different functionalities:

### 2.1.1 Customer

The customer can create reservations for a desired store: such reservation is then inserted into a queue. This queue is managed in order to avoid overcrowding, and to serve customers with a first come first served policy (FIFO queue). To achieve this, statistics about visits' durations are exploited, taking also into account information that customers may provide (e.g. which department of the store they want to visit).
There are multiple types of reservations that can be requested:

1. Immediate reservation: the costumer wants to queue up immediately to a store. He/she is also provided with an estimation of the time he has to wait before his/her turn.

2. Future reservation: the costumer wants to book a future visit at a desired time and store.
   When creating the reservation the customer can specify how long he/she intends to stay and which departments of the market he/she plans to go to, in order to provide a better plan. If he/she does not specify the duration the system can infer it using some statistic built on his/her previous visits. Then the customer will be provided whit the actual time he will be able to access the store (considering the bookings from other customers).

3. On premise reservation: the customer is allowed to enqueue directly at the store. Each of them will provide a system to print tickets so that those who do not have access to the required technology for the previous options can still line up. These tickets contain a reservation number and an estimation of the waiting time before being able to enter.

Each customer will be assigned a number that represents his/her position in queue. A monitor outside the store will display the number of the last customer allowed to enter. Customers who requested an immediate or a future reservation can receive an alert when they need to depart to reach the store through their smartphone app. This alert is based on the location of the customer, so that the time needed to reach the store is taken into account. If, for any reason, a customer cannot to go to a store he/she has lined up to it is possible for him/her to delete the immediate or future reservation. The on premise reservation cannot be deleted.

Reservations can be in one of the following states:

- Pending: customer cannot yet access the store
- Authorized: customer can access the store
- Current: customer is in the store
- Expired: customer has exited the store or the reservation has been canceled

### 2.1.2 Store owner

The store owner can

1. Register a store to the system so that it can be accessed by customers
2. Set the maximum occupation threshold for each of his stores.
3. Monitor the number of customers inside at any time
4. Visualize statistics about the flow of clients and department occupation

When setting the maximum occupation of a store, the owner can specify a threshold for each department or for the whole store. In the latter case the occupation of each department is set to the same value. In case customers select which sections of the store they intend to visit their presence will be considered only in those specific departments, in order to allow more customers in, provided that they will not come in contact with each other. For safety, if customers don't specify anything they are supposed to visit each department.

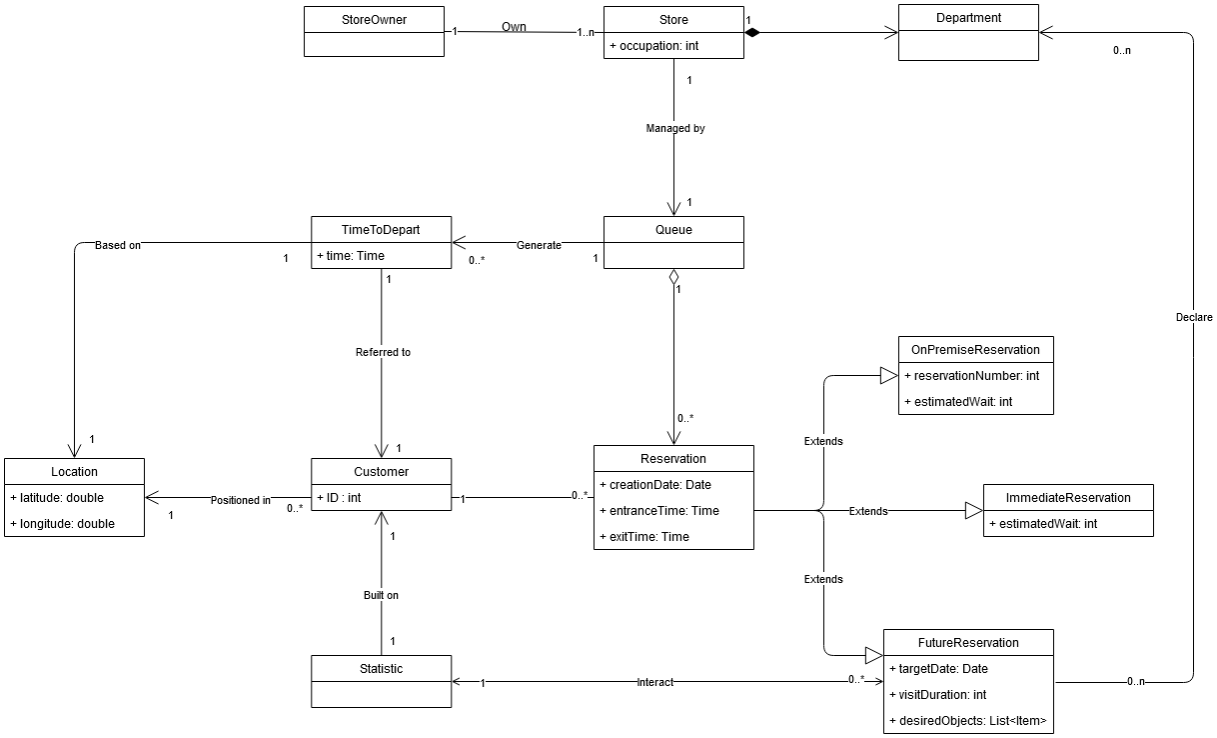



Figure 1: Class Diagram.

### 2.1.3 State charts

Here we show the main processes that the system will manage, and the states in which the system will find itself.

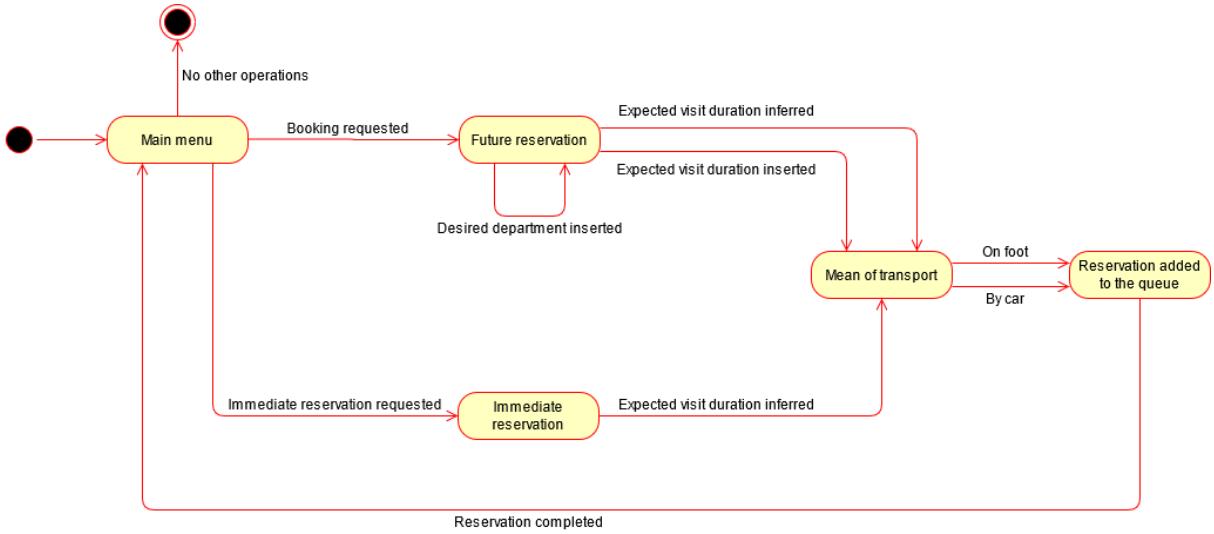



Figure 2: State Diagram 1: Creation of a reservation.

The figure above illustrates how the customer creates a reservation (virtual customer). The customer accesses the main menu. From there he/she can request an immediate reservation or a future reservation. If he/she requests an immediate reservation, it is created and the process terminates. If he/she requires a future reservation he can add the departments he/she intends to visit, and how much time he/she is going to spend at the store. Then the reservation is created and the process terminates.

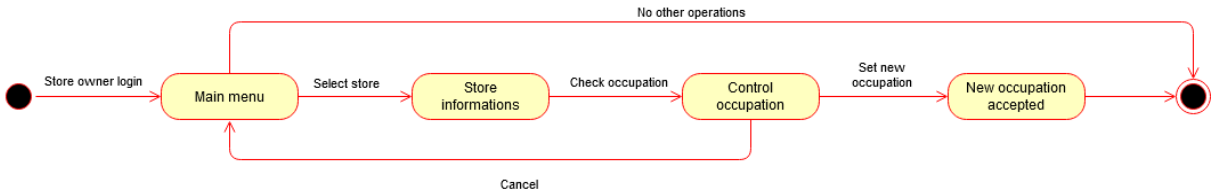



Figure 3: State Diagram 2: Store owner sets occupation.

The figure above illustrates how the store owner sets the maximum occupation of one of his stores. The store owner logs in and is provided a main menu. He/she selects one of his stores, and is presented with the current maximum occupation of his selected store. He can change it to a value to his liking, or go back to the main menu.

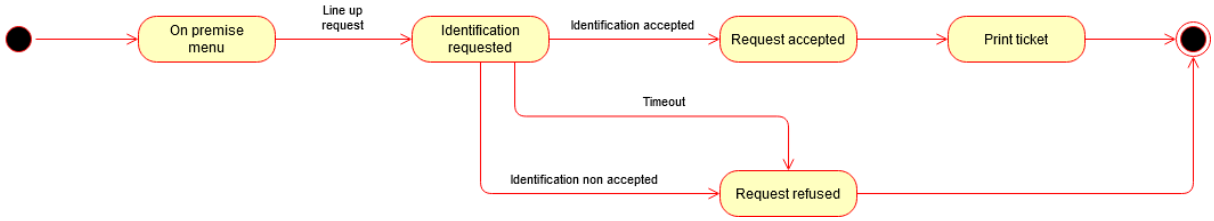



Figure 4: State Diagram 3: On premise creation of a reservation.

The figure above illustrates how a customer can queue up on premise (physical customer). The customer is presented with an on premise main menu. From there he/she can request to queue up (create a reservation) immediately. He will be prompted to identify himself (e.g. social security card). Once identified the customer will be provided a ticket with a number that represents his position in queue and an estimate of the waiting time.

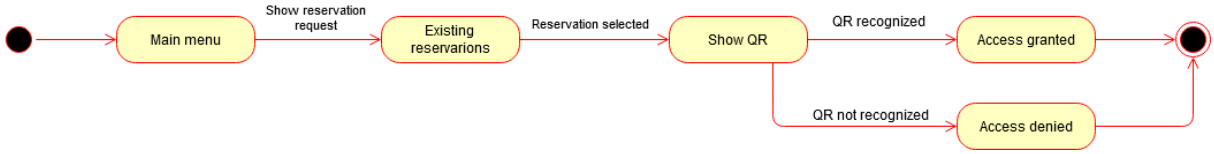



Figure 5: State Diagram 3: Customer enters a store.

The figure above illustrates how a customer who is authorized can access the store. At the entrance of his/her store of choice he is enabled to demonstrate his authorization. Once he/she does, he is allowed to enter the store.

### 2.1.4 Scenarios

**2.1.4.1 Customer queues up remotely** Gordon comes home from work at 5:30 PM. He would like to go to the store. He is a registered user of CLup. His smartphone has internet connection and GPS on. He opens the CLup mobile application and requests an immediate reservation to a store of his choice, selecting the car as means of transport. After some time he is alerted that he needs to depart to avoid being late at the store and losing the reservation. Gordon departs and arrives to the store. He opens the application and selects the reservation he created which now contains the number that represents his position in the queue. After some time the monitor outside the store shows his number, which means it is his turn to enter. He activates the turnstiles with the QR now shown in the reservation page of the mobile application and enters the store. He buys all the products he needs. He opens the application and selects the reservation he created which now contains a QR code. He activates the turnstiles and exits the store.

**2.1.4.2 Customer queues up on premise** Marco is an old man. He does not have a smartphone. He would like to go to the store next to his house on foot. He reaches the store, and retrieves a queue reservation ticket from the printer outside the store, by inserting his social security card. The ticket displays a QR code, a number that represents his position in queue and an estimated waiting time. He can now come back home and wait the specified amount of time, then come back to the store. At that point the monitor outside the store shows his number, which means it is his turn to enter. He activates the turnstiles with the QR printed on his ticket and enters the store. He buys all the products he needs. He activates the turnstiles with his ticket and exits the store.

**2.1.4.3 Store owner registering a store and setting occupation** Paolo is the proud owner of the famous supermarket chain "Junes" and today there is the first opening of a new store of the chain. Due to Covid restrictions, he already use CLup to avoid overcrowding inside his stores. He decide to use the service also for the new building, so he access the system through his PC. Once logged in he register his

new supermarket by inserting all necessary data. Then he select the newly registered store and set an appropriate occupation for each of its department, based on how many people it can contain, ensuring social distances.

**2.1.4.4 Customer books a visit** Lara is very organized woman; she likes to manage situations in advance, and for this reason, in the recent period, she uses CLup to book her visit to the store. Every Sunday she open the CLup web page from her phone, logs in and create a new reservation for the next Saturday. Because she request this reservations in advance af almost a week, she usually has no problems to book exactly for the time she prefer, so she is able to get an appointment for the 9.30 and print the ticket. In this way when the time arrives she can enter the store with no delays and rows.

## 2.2 Product functions

We describe here functions that the system will support.

1. Customers can queue up virtually or physically, immediately or in the future at one of the stores registered to the service.
2. Customers can delete a virtual reservation.
3. Customers can access the desired store when its occupation reaches acceptable levels.
4. Customers that queued up virtually can be notified when it is time for them to depart to reach the store.
5. Store owners can monitor the occupation in each of their stores.
6. Store owners can define the maximum occupation of each department of their stores.
7. Store owners can register stores to the system.

## 2.3 User characteristics

1. **Customers**: a physical person that needs to access any of the stores registered to the system. The customers belong to all demographics, thus the need for a user-friendly interface for both virtual and physical customers. The customer needs a reasonably precise estimate of waiting time and time to get to the store.
2. **Store owners**: a physical or legal person that owns any number of stores and needs to enable customer access through a queue system.

## 2.4 Assumptions, dependencies and constraints

### 2.4.1 Domain Assumptions

D1 The stores have QR activated turnstiles.

D2 Turnstiles let one and only one person in each time they unlock.

D3 Outside stores is a social security card activated ticket printer.

D4 Outside stores there is a monitor.

D5 There is no way for a customer to enter a store except from entrance and exit.

D6 Each customer has either a telephone number or an identification document.

D7 When provided, user location has maximum error of 5 meters.

D8 To register to the S2B users must have either a smartphone or a computer.

D9 To register and use the S2B users must have an internet connection.

# 3 Specific Requirements

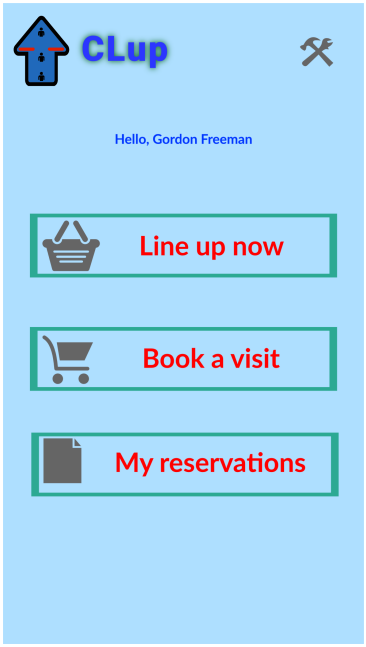
## 3.1 External interface requirements

### 3.1.1 User interfaces

There are two categories of users that have different interface requirements:

- **Customers**
  Customers belong to all demographics so a user friendly interface is needed. The customer is presented with a main menu which allows him/her to:
  - line up immediately (immediate reservation) at a specific store
  - book a visit (future reservation) at a specific store
  - view and delete existing reservations

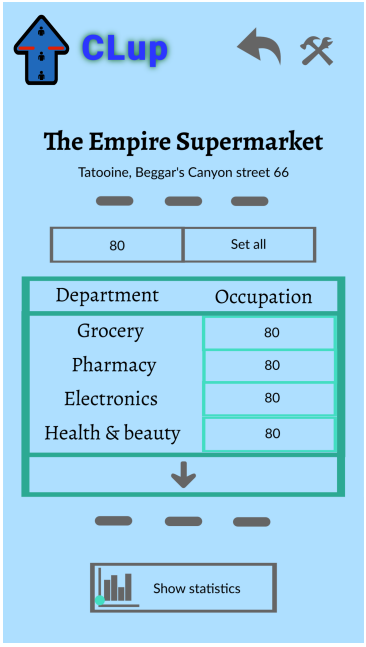
  The customer will receive a notification when it is time for him/her to depart to reach the shop.






- **Store owners**
  The store owner is presented with a main menu from which he/she can:
  - register a store to the system
  - delete a store from the system
  - view and edit occupation for currently registered stores


### 3.1.2 Hardware interfaces

The S2B requires the following hardware interfaces:

1. **Computer or smartphone**
   Users will need a computer or a smartphone to access the system's services which are provided via an application (only for smartphone owners) and a web application. Only users with an application will be able to receive notifications that alert them when it is time for them to depart and reach the store.

2. **Turnstiles**
   Turnstiles allow authorized customers to enter or exit the store by providing a means of identification. (i.e. QR, NFC)

3. **Ticket printer**
   The ticket printers located outside stores allows potential customers to queue up on premise provided that they identify themselves (e.g. social security card).

4. **Monitor**
   A monitor located outside stores allows customers that queue up on premise to know when it is time for them to access the store.

### 3.1.3 Software interfaces

The system uses a public API to locate the customer and provide him/her with notifications about the time he will need to depart to reach the store.

### 3.1.4 Communication interfaces

Customers can access the system through a working internet connection.

## 3.2 Functional requirements

### 3.2.1 List of requirements

R1 Turnstiles unlock if and only if activated by authorized customers.

R2 The number of customers in each department of the store never exceeds the occupation set by the owner.

R3 The monitor outside the store displays the number of the last authorized customer.

R4 The system allows customers and store owners to register and log in.

R5 The system validates the authenticity of the identifying information provided.

R6 The system allows customers to search for a store among those registered by their owners.

R7 Registered customers can send a reservation request to the system.

R8 Non registered customers with an identifying document can request reservations through the printer.

R9 Registered customers that book a visit can specify estimated visit duration.

R10 Registered customers that book a visit can specify desired product categories.

R11 The system provides customers with a QR code to enter the store once authorized.

R12 The system uses gathered data to build statistics.

R13 Registered customers with a smartphone are alerted when their turn is near.

R14 Registered customers can delete a pending or authorized reservation.

R15 Authorized reservations expire if they do not become current in a certain time window (specified by store owner)

R16 Registered customers must specify desired means of transport while requesting a reservation in order to receive notifications.

R17 Reservations are authorized according to a FIFO policy

### 3.2.2 Mapping on requirements

| | | |
|---|---|---|
| G1 | D1, D2, D5, D6, D8, D9 | R1, R2, R4, R5, R6, R7, R11, R16, R17 |
| G2 | D1, D2, D3, D4, D5, D6 | R1, R2, R3, R5, R8, R11, R17 |
| G3 | D1, D2, D3, D4, D5, D6, D8, D9 | R1, R2, R3, R4, R5, R6, R7, R9, R10, R11, R16, R17 |
| G4 | D1, D2, D5, D8, D9 | R1, R2 |
| G5 | D8, D9 | R9, R12, R14, R15 |
| G6 | D4, D7, D8, D9 | R3, R13 |

Here we remind the goals for ease of use:

G1 CLup should allow customers to queue up remotely

G2 CLup should allow customers to queue up on premise

G3 CLup should allow customers to book future visits to stores

G4 CLup should allow store owners to regulate the maximum number of customers in their stores

G5 CLup should provide the customer with a reasonably precise estimate of waiting time

G6 CLup should alert the customers when it is time to get to the shop taking into account travel time

Here we remind the domain assumptions for ease of use:

D1 The stores have QR activated turnstiles.

D2 Turnstiles let one and only one person in each time they unlock.

D3 Outside stores is a social security card activated ticket printer.

D4 Outside stores there is a monitor.

D5 There is no way for a customer to enter a store except from entrance and exit.

D6 Each customer has either a telephone number or an identification document.

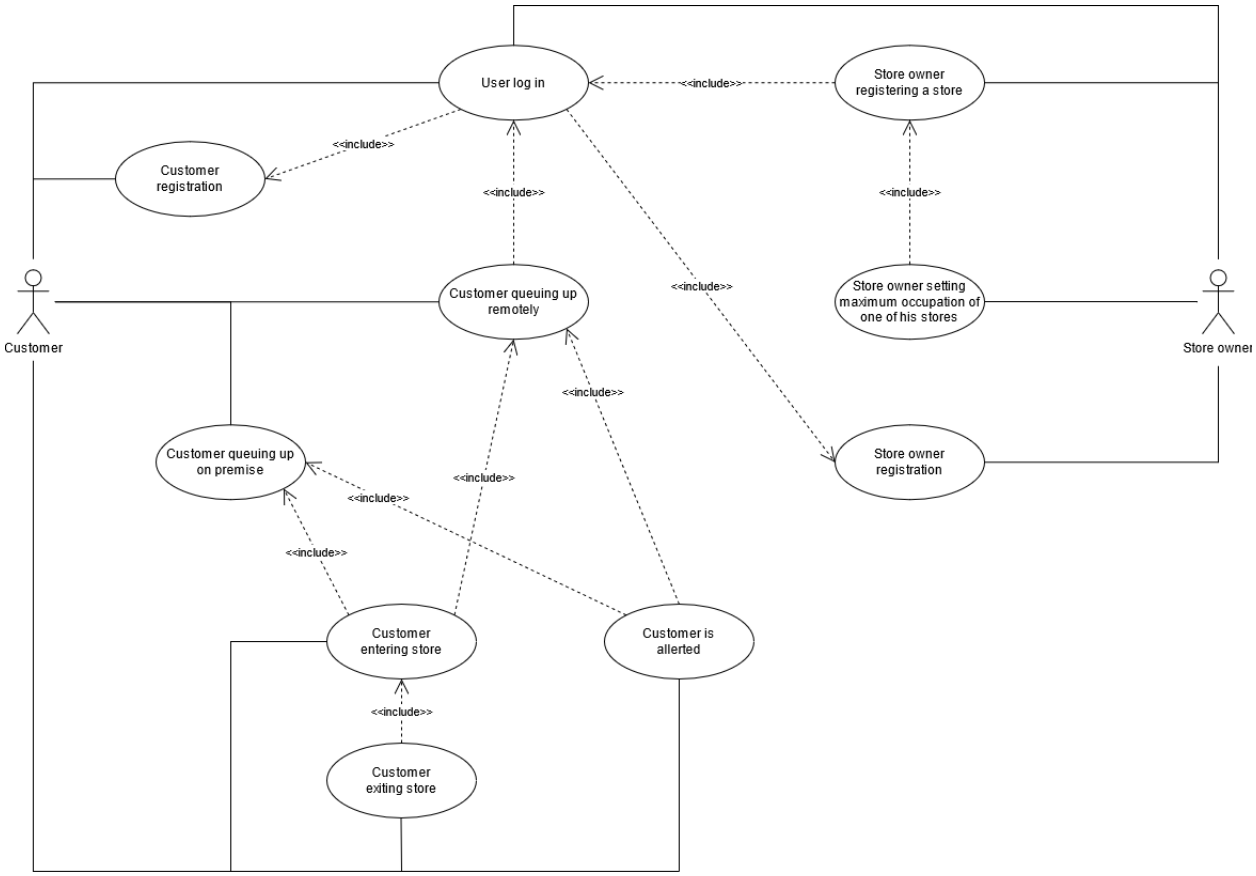
D7 When provided, user location has maximum error of 5 meters.

D8 To register to the S2B users must have either a smartphone or a computer.

D9 To register and use the S2B users must have an internet connection.

### 3.2.3 Use case diagram



### 3.2.4 Use cases

#### 3.2.4.1 Customer registration

| Name | Customer registration |
|---|---|
| Actors | Customer |
| Entry condition | Customer has opened the smartphone application or the web app on his computer but has not logged in |
| Event flow | 1. a registration menu is provided to the customer<br>2. from such menu the customer selects the option to sign up as customer<br>3. the customer is then prompted to insert identifying information<br>4. the customer inserts requested information<br>5. the system validates provided information<br>6. the system confirms the registration of the customer and saves information provided |
| Exit conditions | Customer has registered to the system |
| Exceptions | 1. a customer with same identifying information already exists<br>2. validation of identifying information is not successful<br>3. the customer decides to cancel the registration<br><br>If one of the first two events described above occur, the application will alert the customer and provide him with the possibility to retry or go back to the initial menu.<br>If event 3 occurs the customer is redirected to the main page. |

#### 3.2.4.2 Store owner registration

| Name | Store owner registration |
|---|---|
| Actors | Store owner |
| Entry condition | Store owner has opened the smartphone application or the web app on his computer but has not logged in |
| Event flow | 1. a registration menu is provided to the store owner<br>2. from such menu the store owner selects the option to sign up as store owner<br>3. the store owner is then prompted to insert identifying information<br>4. the store owner inserts requested information<br>5. the system validates provided information<br>6. the system confirms the registration of the store owner and saves information provided |
| Exit conditions | Store owner has registered to the system |
| Exceptions | 1. a store owner with same identifying information already exists<br>2. validation of identifying information is not successful<br>3. the store owner decides to cancel the registration<br><br>If one of the first two events described above occur, the application will alert the store owner and provide him with the possibility to retry or go back to the initial menu.<br>If event 3 occurs the store owner is redirected to the main page. |

#### 3.2.4.3 User logs in

| Name | User logs in |
|---|---|
| Actors | Customer or store owner |
| Entry condition | The user has opened the application on his device, and has already registered to the system |
| Event flow | 1. User chooses to log in from the welcome menu<br>2. User identifies him/herself with created credentials |
| Exit conditions | User is logged in |
| Exceptions | 1. User credentials are invalid<br><br>If the event described above occurs, the application will alert the user and allows him to retry |

#### 3.2.4.4 Customer queuing up remotely

| Name | Customer queuing up remotely |
|---|---|
| Actors | Customer |
| Entry condition | Customer has logged in on the smartphone application or the web app on his computer |
| Event flow | 1. customer selects to queue up from main menu<br>• with an immediate reservation<br>• with a future reservation (and optionally inserts desired categories of item he/she intends to buy and how much time he/she intends to spend at the store)<br>2. customer selects a store from the list of stores registered to the system<br>3. customer specify whether he is going to reach the store by car or on foot<br>4. customer submits reservation request |
| Exit conditions | Customer reservation is confirmed |
| Exceptions | 1. customer decides to cancel the reservation<br>2. customer requests immediate reservaiton but store is closed at that time<br>If one of the events described above occur, the application will alert the customer and go back to the initial menu |

#### 3.2.4.5 Customer queuing up on premise

| Name | Customer queuing on premise |
|---|---|
| Actors | Customer |
| Entry condition | Customer has reached the store ticket printer |
| Event flow | 1. customer selects the option to queue up from main menu of the ticket printer<br>2. customer provides the ticket printer with a means of identification (e.g. social security card) |
| Exit conditions | Customer reservation is confirmed and a ticket is printed containing the following information:<br>• how much time he needs to wait before being able to enter the store<br>• a progressive number that will allow him to know when his/her turn is |
| Exceptions | 1. customer decides to cancel the reservation<br>If one of the events described above occur, the application will alert the customer and go back to the initial menu |

#### 3.2.4.6 Customer entering store

| Name | Customer entering store |
|---|---|
| Actors | Customer |
| Entry condition | Customer has logged in on the smartphone application and is at the entrance of a store or has printed authorization from the web app |
| Event flow | If the customer has not printed the reservation ticket he must use his smartphone:<br>1. customer selects the option to show existing reservations from main menu of the phone app<br>2. customer selects an existing reservation and if authorized (i.e. it is his turn to enter the store) he/she is given the means to identify him/herself (e.g. display QR or activate NFC)<br>Then the customer identifies him/herself at the turnstiles |
| Exit conditions | Customer enters the store |
| Exceptions | 1. customer is not authorized to enter the store<br>If the event described above occurs, the turnstiles will not let the customer in |

#### 3.2.4.7 Customer exiting store

| | |
|---|---|
| Name | Customer exiting store |
| Actors | Customer |
| Entry condition | Customer has logged in on the smartphone application and is in a store |
| Event flow | If the customer has not printed the reservation ticket he must use his smartphone:<br>1. customer selects the option to show existing reservations from main menu<br>2. customer selects an existing reservation and if authorized (i.e. it is his turn to enter the store) he/she is given the means to identify him/herself (e.g. display QR or activate NFC)<br>Customer identifies him/herself at the turnstiles |
| Exit conditions | Customer exits the store |
| Exceptions | |

#### 3.2.4.8 Store owner registering a store

| | |
|---|---|
| Name | Store owner registering a store |
| Actors | Store owner |
| Entry condition | Store owner has logged in on the smartphone application or the web app on his computer |
| Event flow | 1. store owner selects the option register a store from main menu<br>2. store owner inserts necessary information and sets up equipment (i.e. connect printer, monitor and turnstiles to the system)<br>3. store owner submits store registration request |
| Exit conditions | Store registration is confirmed |
| Exceptions | 1. information is missing or incorrect<br>2. equipment is not working properly<br>3. store owner decides to cancel the store registration<br>If one of the events described above occur, the application will alert the store owner and provide him with the possibility to retry or go back to the initial menu |

#### 3.2.4.9 Store owner setting maximum occupation of a store or a department

| | |
|---|---|
| Name | Store owner setting maximum occupation of a store or a department |
| Actors | Store owner |
| Entry condition | Store owner has logged in on the smartphone application or the web app on his computer |
| Event flow | 1. store owner selects one of his stores from the list reachable form the main menu<br>2. store owner views current occupation of each department of his store, and current occupation threshold<br>3. sets the new desired occupation threshold for a department or for the whole store |
| Exit conditions | The new occupation threshold is set |
| Exceptions | 1. the threshold value is inadequate<br>If the event described above occurs, the application will alert the store owner and provide him with the possibility to retry or go back to the initial menu |

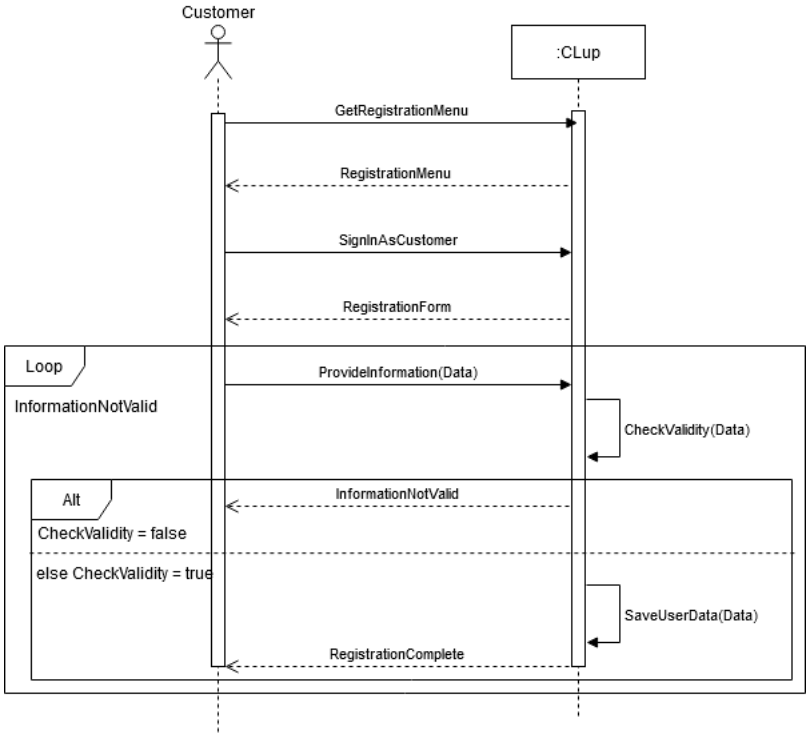
#### 3.2.4.10 Customer is alerted

| | |
|---|---|
| Name | Customer is alerted |
| Actors | Customer |
| Entry condition | Customer is logged into his smartphone application and the time he/she needs to wait becomes less or equal to the time he/she needs to reach the store |
| Event flow | 1. The system sends a notification to the smartphone of the customer |
| Exit conditions | Customer is notified |
| Exceptions | 1. the threshold value is inadequate<br>If the event described above occurs, the application will alert the store owner and provide him with the possibility to retry or go back to the initial menu |

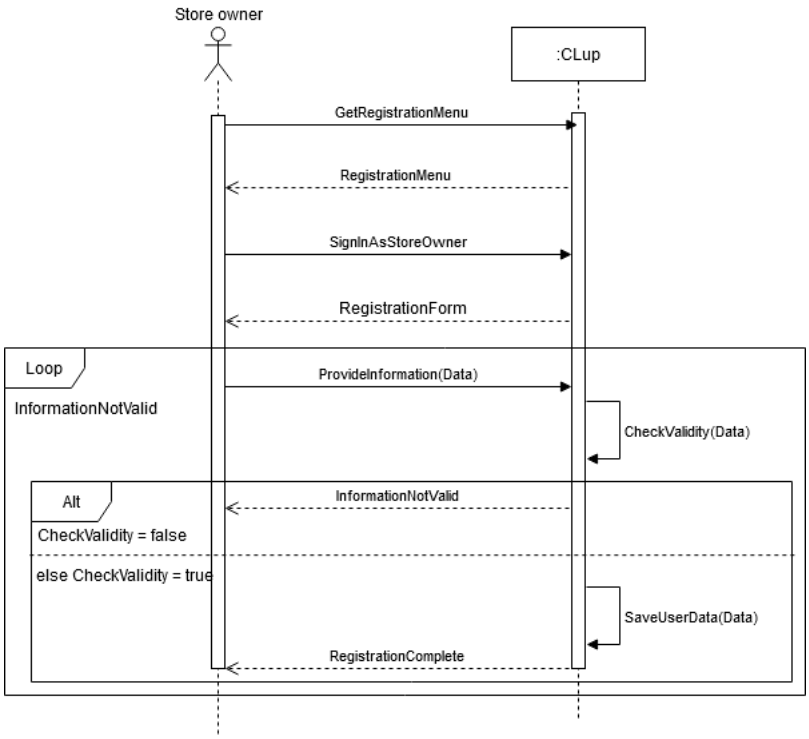
 

### 3.2.5 Sequence diagrams

#### 3.2.5.1 Customer registration

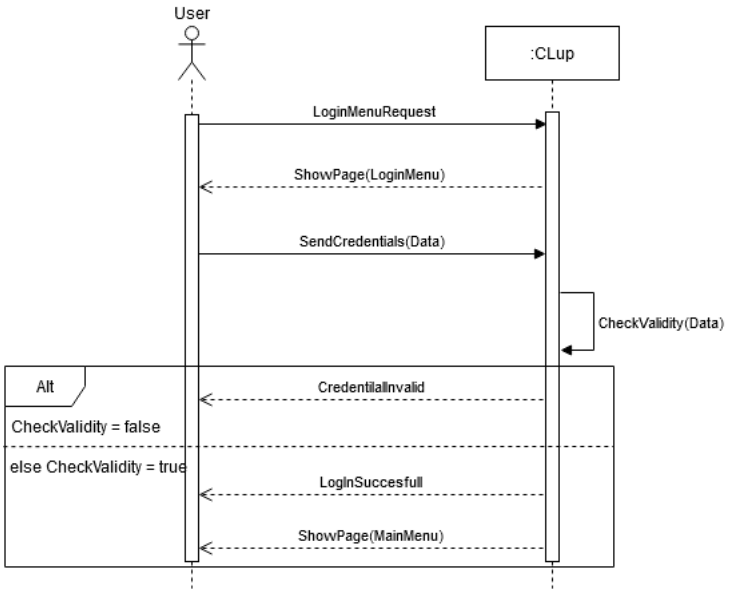



#### 3.2.5.2 Store owner registration

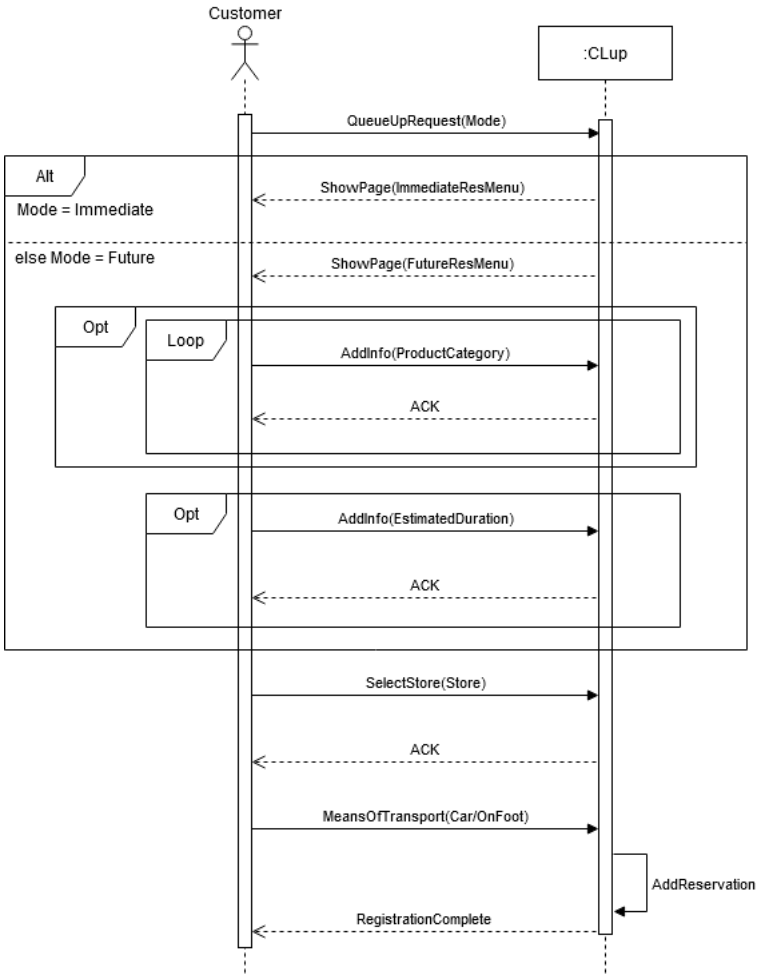


#### 3.2.5.3 User logs in

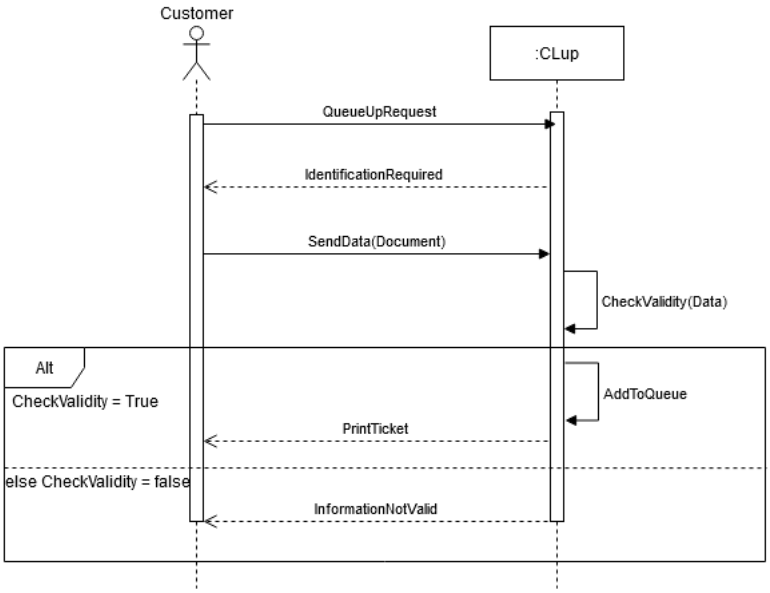



#### 3.2.5.4 Customer queuing up remotely

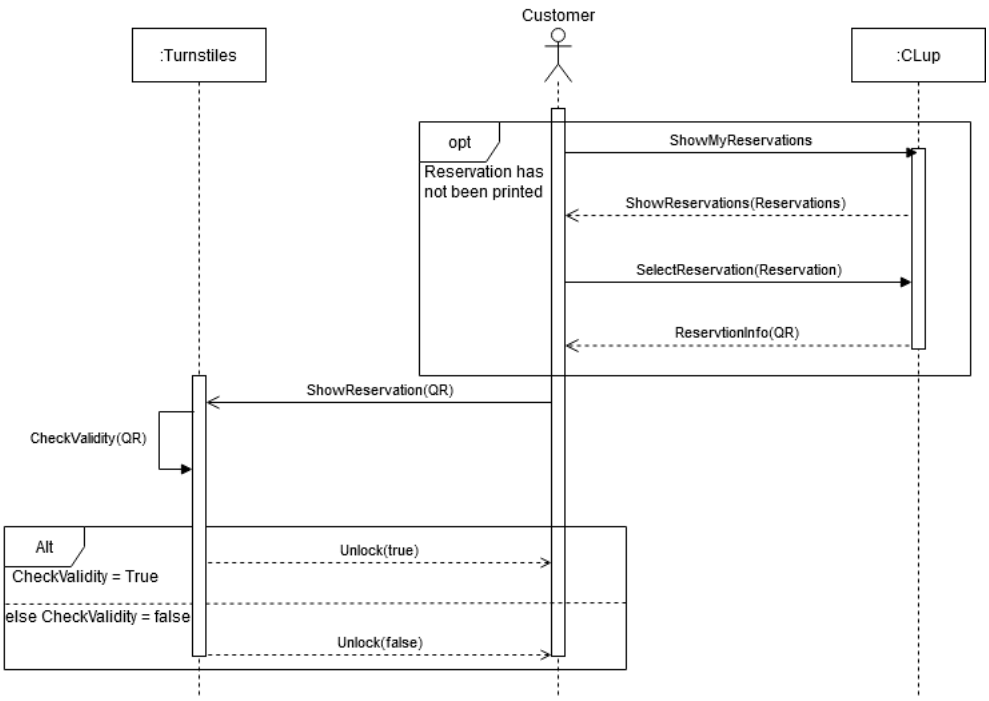


#### 3.2.5.5 Customer queuing up on premise

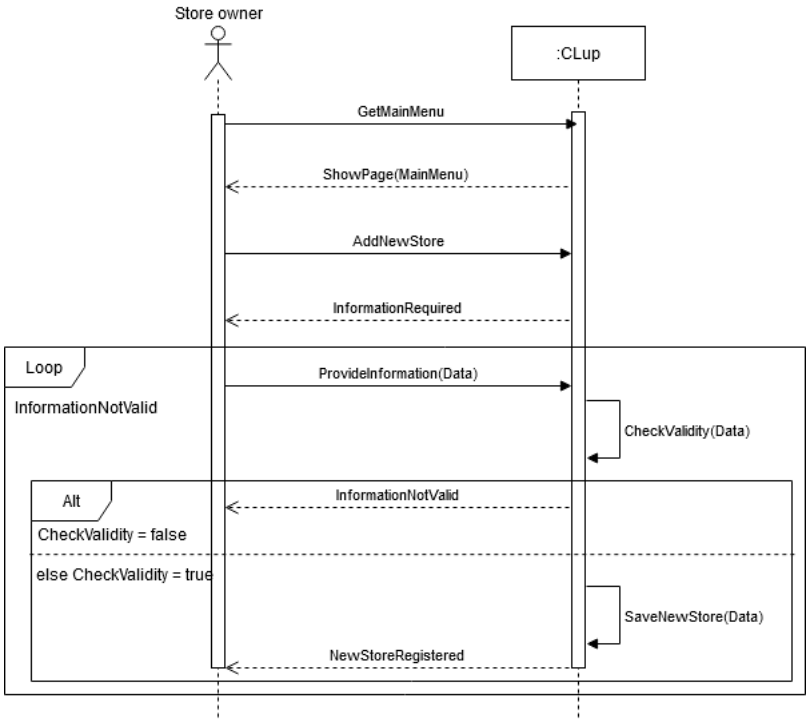



#### 3.2.5.6 Customer entering and exiting store

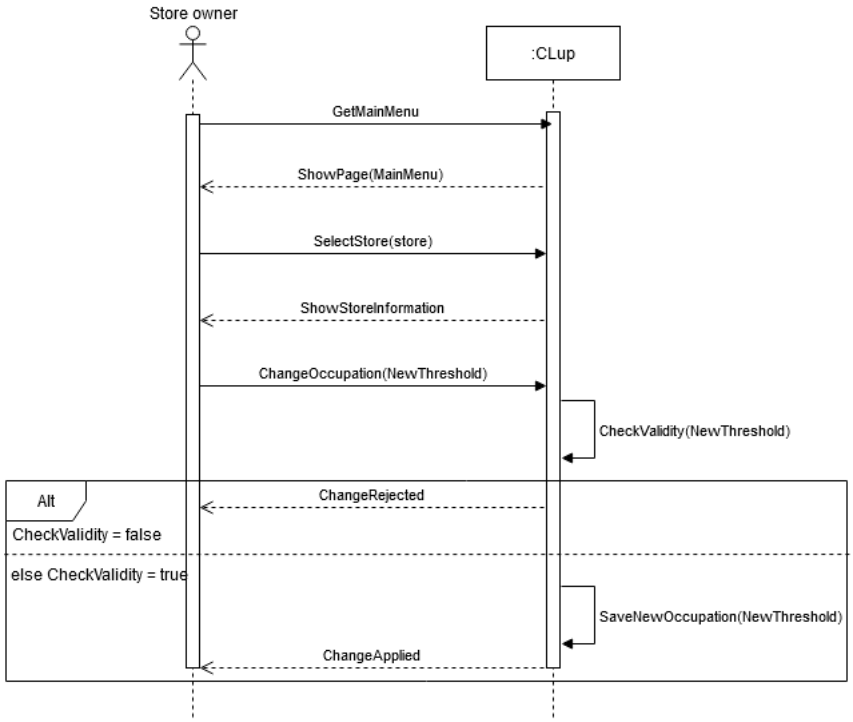


#### 3.2.5.7 Store owner registering a store

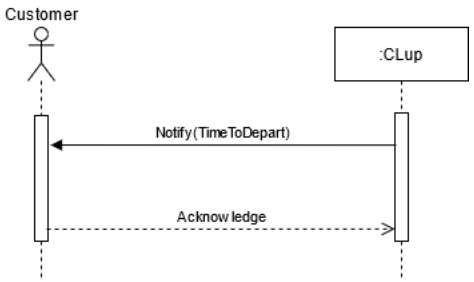



#### 3.2.5.8 Store owner setting maximum occupation of a store or a department



#### 3.2.5.9 Customer is alerted




## 3.3 Performance Requirements

The system cannot guarantee that a customer can enter a store at a precise time unless it requires users to exit after a maximum time (which it doesn't). The system guarantees that if

- the customer uses a smartphone
- the customer never disconnects from the internet and GPS
- the customer specified the correct means of transport when creating the reservation

he/she will be alerted when he needs to depart to reach a store (in order not to be late) with a delay of at most 10 seconds.

## 3.4 Design Constraints

### 3.4.1 Standards compliance

The code should follow the requirements contained in this document, and be thoroughly commented.

### 3.4.2 Hardware limitations

The software the customer uses requires either:

- a smartphone to use the smart application
- a computer to use the web application and a home printer to print reservation

The software the store owner uses requires:

- turnstiles activated by QR
- reservation printer activated by social security card
- monitor to alert on premise customers when it is their turn

### 3.4.3 Any other constraint

Customers cannot have more than five active reservation requests for different stores and more than two for the same store on the same day to avoid fake reservations.

## 3.5 Software System Attributes

### 3.5.1 Reliability

The system must have an appropriate infrastructure with a full backup system located in an separate office distant at least 100km (nuclear fallout radius). Adequate personnel will guarantee recovery time to substitute faulty hardware.

### 3.5.2 Availability

The system should be up for 99.9% of the time (0.365 MTTR). Its temporary downtime does not cause emergency situations, but the system is an essential service: it should be possible to buy food every day. The system is fully automated. The users are alerted about system downtime with a delay of at most 10 minutes. The users are alerted that the system is up again with a delay of at most 10 minutes.

### 3.5.3 Security

The location of customers is sensitive information and therefore is never stored. Customers and store owners provide identifying information during registration: the databases containing such information must be protected against internal and external attacks. Communication between central system and users is encrypted.

### 3.5.4 Maintainability

The system is easy to maintain: its code is thoroughly commented and modular. Appropriate design patterns are exploited.

### 3.5.5 Portability

The smartphone application runs under Android and iOS. The web application runs under Android, iOS, Windows, MacOS.

# 4 Formal Analysis Using Alloy

## 4.1 Alloy Code

This section is dedicated to the Alloy model of the CLup software, included all the functions of the application and their most important constraints.
In the model are represented the two categories of users: the store owners and the customers. Each store owner owns a set of stores, and each customer opens a set of reservations. Those reservations can be of three types: immediate, on premise and future. Every reservation refers to a store, has an estimated time for the customer who opened it to reach the store, a time for that customer to wait before he/she is able to enter, a value representing in which moment the reservation has been created, and a boolean value, representing whether the customer has already been alerted (he/she must depart to reach the store). Moreover each reservation can be in one of 4 states:

- PENDING: reservation has been requested but customer cannot yet access the store
- AUTHORIZED: customer can access the store for a time window
- CURRENT: customer has accessed the store and has not exited yet
- EXPIRED: customer has exited the store or the time window to enter the store ended or the reservation was deleted

Finally a future reservation keeps track of entry time. We apply some constraints to the model so that it can represent the application domain of our S2B. First, customers are queued up following a First In First Out policy. We ensure that the parameters in the reservations are meaningful with respect to the state of the reservation itself, e.g. a customer is alerted when the time remaining before it is his/her turn is less or equal than the time for him/her to reach the store. For every store we check that the current occupation is equal to the number of customer inside the store and that this value never exceeds the maximum occupation of that store.

```
                --------------------------
                --      SIGNATURES      --
                --------------------------

sig Customer{
        -- list of the reservations he/she requested
        opens: set Reservation
}

sig StoreOwner{
        -- list of the stores he/she/it owns
        owns: set Store
}

sig Store{
        -- current occupation is the number of people currently in the store
        currentOccupation: one Int,
        -- maximum occupation is the maximum number of people that can fit in a store at
            ↪ any given time
        maximumOccupation: one Int
}
{
        -- store has at most one owner
        no s: Store, disj o1, o2: StoreOwner | s in o1.owns and s in o2.owns
        --store has at least one owner
        all s: Store | some o: StoreOwner | s in o.owns

        -- non negativity constraints
        currentOccupation ≥ 0
        maximumOccupation ≥ 0
```

```
}

abstract sig Reservation{
        -- store to which the reservation refers
        refersTo: one Store,
        -- state can be: pending, authorized, current, expired
        state: one ReservationState,
        -- estimate of time needed to reach the store from the customer location using
            ↪ selected means of transport
        timeToReachStore: one Int,
        -- estimate of time before the customer will be authorized to enter
        timeToWait: one Int,
        -- True iff the customer has been alerted
        alertedCustomer: one Bool,
        -- time of creation of the reservation
        creationTime: one Int
}
{
        -- reservation is opened by at most one customer
        no r: Reservation, disj c1, c2: Customer | r in c1.opens and r in c2.opens
        -- reservation is opened by at least one customer
        all r: Reservation | some c: Customer | r in c.opens

        creationTime ≥ 0
        timeToReachStore ≥ 0
        timeToWait ≥ 0
}

-- customer wants to queue up immediately
sig ImmediateReservation extends Reservation {}
-- customer wants to queue up immediately on premise
sig OnPremiseReservation extends Reservation {}
-- customer wants to book a visit to the store for a future date
sig FutureReservation extends Reservation {
        -- when booking a future reservation a customer specifies a target time.
        -- In most cases target time and entry time coincide; the latter may be
        -- greater than the former in case of delays.
        entryTime: one Int
}

abstract sig ReservationState{}
{
        -- a reservation state has meaning if a reservation is in that state
        all rs: ReservationState | some r: Reservation | r.state = rs
}

-- reservation has been requested but customer cannot yet access the store
sig PENDING extends ReservationState{}
-- customer can access the store for a time window
sig AUTHORIZED extends ReservationState{}
-- customer has accessed the store and has not exited yet
sig CURRENT extends ReservationState{}
-- customer has exited the store or the time window to enter the store ended or the
    ↪ reservation was deleted
sig EXPIRED extends ReservationState{}

-- returns reservations referring to store s
fun storeReservations [s: Store] : set Reservation {
        refersTo.s
}

-- returns states of reservations referring to store s
fun storeReservationStates [s: Store] : set ReservationState {
        storeReservations[s].state
}

abstract sig Bool {}
one sig True, False extends Bool{}

pred isTrue[b: Bool] { b in True }
pred isFalse[b: Bool] { b in False }
```

```
//--------------------------
//        FACTS            --
//--------------------------

// non ubiquity of customer, customer cannot be inside two different stores at the same time
fact nonUbiquityCurrentCustomer{
        all c: Customer | no disj r1, r2: Reservation | r1 in c.opens and r2 in c.opens and r1.state = CURRENT and r2.state = CURRENT
}

// the number of current reservation referring to a store s equals to s.currentOccupation
fact currentStoreOccupationEqualsCurrentReservations {
        all s: Store | let rs = storeReservationStates[s] | s.currentOccupation = #(rs :> CURRENT)
}

// each reservation has its own state
fact oneStateForEachReservation {
        no disj r1, r2: Reservation | r1.state = r2.state
}

// for every store s, s.currentOccupation never exceeds s.maximumOccupation
fact noExceedMaximumOccupation {
        all s: Store | s.maximumOccupation ≥ s.currentOccupation
}

//  customer that created reservation r is alerted when r.timeToWait <= r.timeToReachStore
fact customerAlertedWhenNecessary {
        all r: Reservation | (r in FutureReservation or r in ImmediateReservation)
        and r.timeToWait ≤ r.timeToReachStore implies isTrue[r.alertedCustomer]
}

// on premise customers cannot be alerted (since they have no smartphone)
fact onPremiseCustomersCantBeAlerted {
        all r: OnPremiseReservation | isFalse[r.alertedCustomer]
}

// for every Reservation r that is not in PENDING state, r.timeToWait is zero
fact nonPendingReservationHaveNoWaitTime {
        all r: Reservation | (r.state not in PENDING) implies r.timeToWait = 0
}

// for every Reservation r that is in PENDING state, r.timeToWait is greater than zero
fact pendingReservationHasPositiveTimeToWait {
        all r: Reservation | r.state in PENDING implies r.timeToWait > 0
}

// for every OnPremiseReservation r, r.timeToReachStore is assumed equal to zero
fact onPremiseReservationsHaveNoTimeToReachStore {
        all r: OnPremiseReservation | r.timeToReachStore = 0
}

// for all virtual reservations r (ImmediateReservation or FutureReservation),
// r.timeToReachStore is greater than zero, which is to say they form no physical queue
// outside the store
fact  noPhysicalQueueOutsideStore {
        all r: Reservation | ((r in ImmediateReservation or r in FutureReservation)
        and r.state not in CURRENT) implies r.timeToReachStore > 0
}

// after entering a store time to reach the store is zero
fact  afterEnteringTimeToReachStoreIsZero {
        all r: Reservation | r.state in CURRENT implies r.timeToReachStore = 0
        all r: Reservation | r.state in EXPIRED implies r.timeToReachStore = 0
}

// no customer can make more than one ImmediateReservation
fact noDoubleImmediateReservationForSameCustomer {
```

```
        no disj r1, r2: ImmediateReservation | some c: Customer | r1.state ≠ EXPIRED and
            ↪ r2.state ≠ EXPIRED and r1 in c.opens and r2 in c.opens
}

-- no customer can make more than one OnPremiseReservation
fact noDoubleOnPremiseReservationForSameCustomer {
        no disj r1, r2: OnPremiseReservation | some c: Customer | r1.state ≠ EXPIRED and
            ↪ r2.state ≠ EXPIRED and r1 in c.opens and r2 in c.opens
}

-- Physical reservations (on premise and immediate) follow a FIFO policy:
-- The customer that creates the reservation first, enters first.
fact FIFOQueue {
        all r1, r2: Reservation | ((r1 in OnPremiseReservation or r1 in
            ↪ ImmediateReservation)
        and (r2 in OnPremiseReservation or r2 in ImmediateReservation) and
        (r1.creationTime < r2.creationTime)) implies (r1.timeToWait ≤ r2.timeToWait)
}

-- Future reservations cannot be created after the entry time
fact noTimeTravel {
        no r: FutureReservation | r.creationTime > r.entryTime
}

-- When creating a future reservation the target time must be greater that the current
    ↪ time for an amount
-- represented in this case by the constant 1
fact DelayBetweenCreationAndTargetTime {
        all r: FutureReservation | r.entryTime - r.creationTime > 1
}

-- for every FutureReservation r, r.timeToWait is dominated by the difference between r.
    ↪ entryTime and r.creationTime
fact timeToWaitLessThanDifference {
        all r: FutureReservation | r.timeToWait ≤ r.entryTime - r.creationTime
}

-- two future reservation for the same store opened by the same customer cannot have the
    ↪ same target time;
-- between the two there must be a delay represented in this case by the constant 1
fact DelayBetweenTwoFutureReservationForSameStore {
        all r1, r2: FutureReservation | some c : Customer | (r1.refersTo = r2.refersTo
            ↪ and r1 in c.opens and r2 in c.opens) implies (r1.entryTime ≥ 1 + r2.
            ↪ entryTime
        or r2.entryTime ≥ 1 + r1.entryTime)
}



            --------------------------
            --      PREDICATES      --
            --------------------------

pred World1(s: Store) {
        s.currentOccupation = 5

        #Customer ≥ 3
        #EXPIRED > 0
        #AUTHORIZED > 0
        #PENDING > 0
        #Store = 1
        #ImmediateReservation ≥ 0
        #OnPremiseReservation = 1
        #FutureReservation = 1
}

run World1 for 20

pred World2(r1, r2 : ImmediateReservation) {
        r1.alertedCustomer = False
        r2.alertedCustomer = True
```

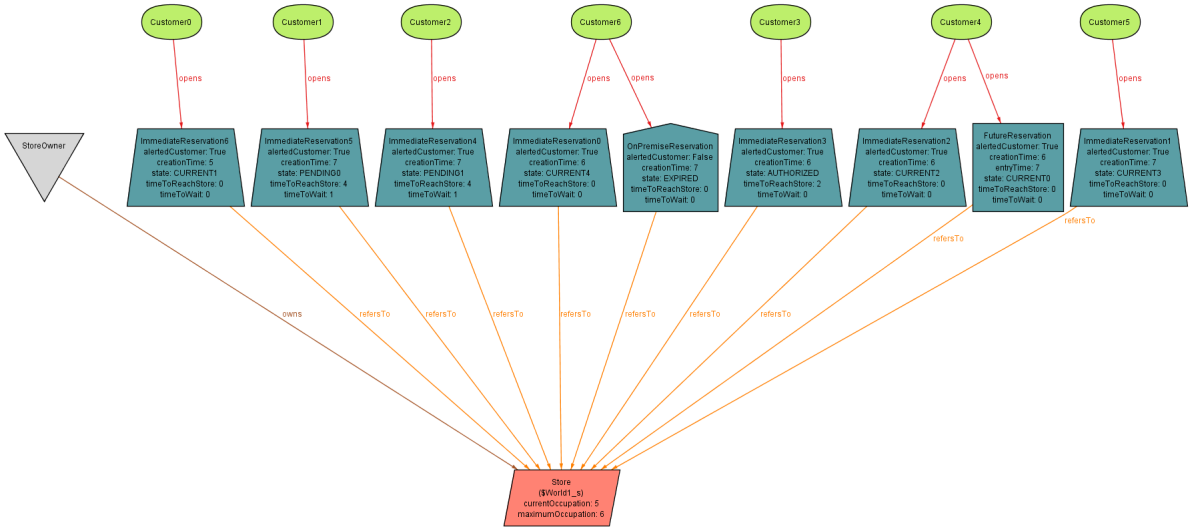
 

```
        r1.creationTime > r2.creationTime
        r1.timeToWait > r2.timeToWait

        r1.refersTo = r2.refersTo

        #Reservation > 3
        #Customer ≥3
        #Store ≥ 3
        #StoreOwner ≥ 2
}

run World2 for 20

pred World3() {

        #Customer ≥ 3
        #CURRENT > 0
        #EXPIRED > 0
        #AUTHORIZED > 0
        #PENDING > 0
        #ImmediateReservation ≥ 0
        #OnPremiseReservation = 1
        #FutureReservation > 2
}

run World3 for 20
```

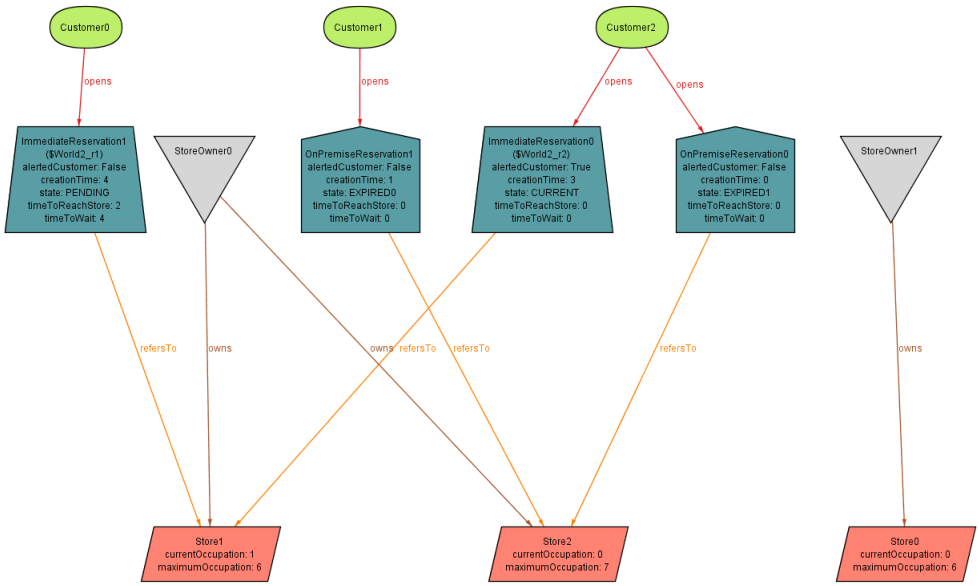
 

## 4.2 Scenarios

Those are some instances of the Alloy model based on the predicates above.

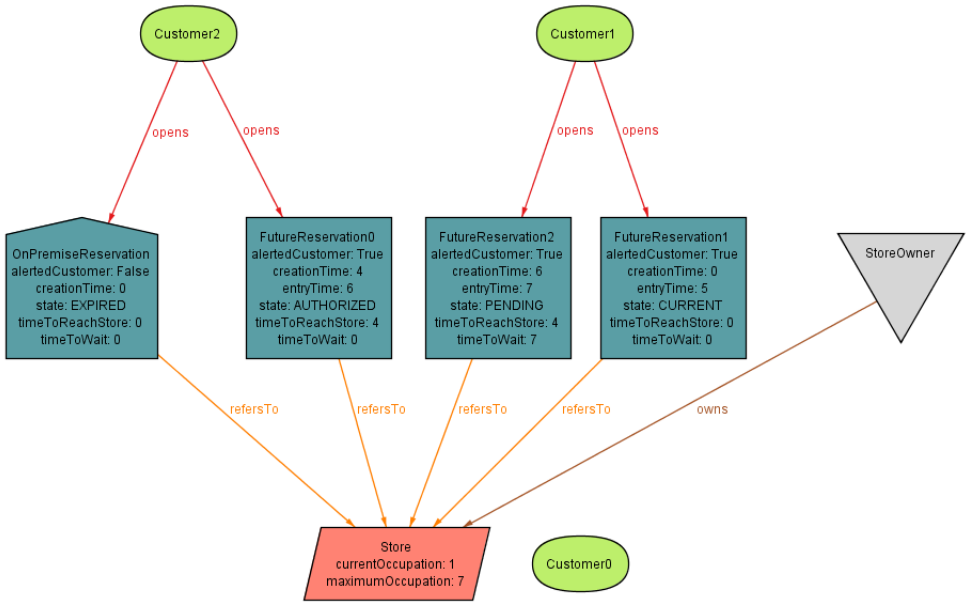



The first scenario shows a store with a current occupation of 5, which represents the actual number of current reservations, and we observe that it is still less than the maximum occupation.

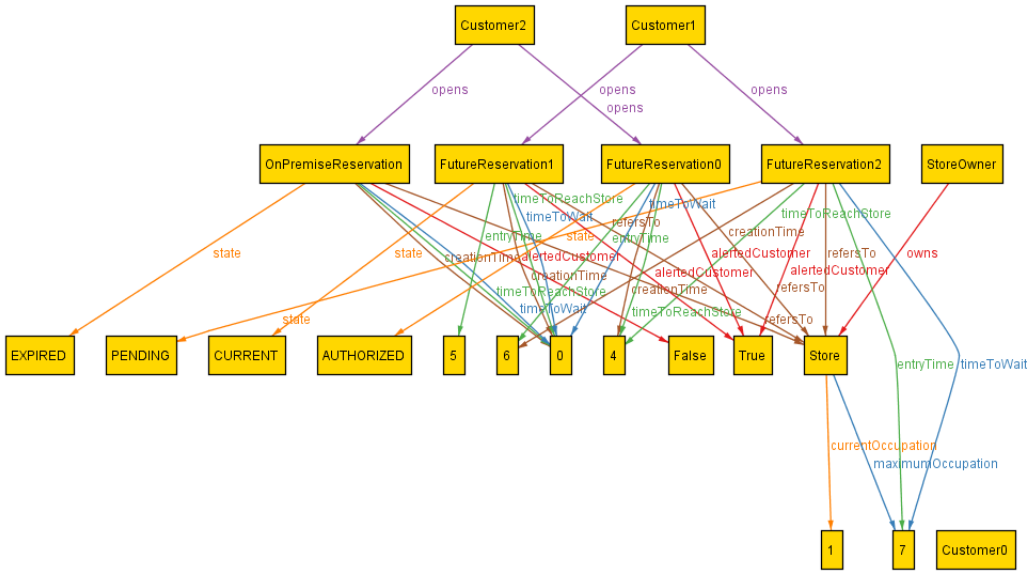



In the second scenario there are two immediate reservations r1 and r2 for the same store. r1 has been created after r2, so r2 has less time to wait with respect to r1.
Moreover the customer that opened r2 has been alerted because his time to wait is less or equal than the time for him to reach the store, while for r1 this property doesn't hold, so the customer has not been alerted.




In the third scenario there is a customer which has not opened any reservation yet.
Customer 1 has booked two reservations and they have different entry time.




We used a theme in the diagrams above to make them more readable; without the theme the third diagram would look like this.

# 5 Effort Spent

## 5.1 Simone Abelli

| | |
|---|---|
| Introduction | 3.5 |
| UML class diagram | 4 |
| Product perspective | 3 |
| State charts, External interface requirements | 4.5 |
| User interfaces | 4.5 |
| Requirements | 6.5 |
| Sequence diagrams | 3.5 |
| Scenarios | 0.5 |
| Alloy | 7 |

## 5.2 Stefano Azzone

| | |
|---|---|
| Introduction | 3 |
| UML class diagram | 3 |
| Product perspective | 3 |
| Product functions, user characteristics, domain assumptions | 2.5 |
| State charts, External interface requirements | 3 |
| Use cases | 3.5 |
| Alloy | 11.5 |
| Requirements | 6.5 |
| Performance Requirements, Design constraints, Software system attributes, first two scenarios | 1 |

## References

- **drawio.org** was used to draw diagrams
- **alloy.mit.edu** was the reference for alloy model
- **uml-diagrams.org** was the reference for uml diagrams