POLITECNICO
MILANO 1863

Software Engineering 2 Project
a.y. 2020-21

Customers Line-up

Requirements Analysis and Specifications Document

Version 0.0

Banfi Stefano Alessandro
Bresciani Matteo

Referent professor: Di Nitto Elisabetta

December 19, 2020

Contents

Chapter 1

Introduction

1.1 Purpose

1.1.1 General Purpose

The main scope of this document is to define requirements for the application development, in order to make a correct project planification.

To do this, we will analyze:

- system;
- functional and unfunctional requirements;
- constraints;
- relationships between stakeholders;
- possible scenarios and tests;

These will be shown using different languages such as Alloy and UML. Then, we will define the context in which our application will be developed.
During Sars-Cov-2 emergency, several countries imposed the lockdown in order to hinter the virus diffusion. People had to change their habits, in fact they could go out only for necessary needs, such as going to the market or pharmacy. For instance people must pay attention while they're entering in the market due to the long queue, which could increase the possibility of virus diffusion. This fact obliged people to stay for a long time standing up and waiting their turn losing a lots of time.

1.1.2 Goals

This are the goals:

G1 User enters once arrived at the market.

G2 Put a limit to the number of Users in the market.

G3 Smart User can make a Reservation of a seat in the market's queue.

G4 Smart User can book in advance a Visit in the market.

G5 Mobile User can make a Reservation of a seat in the market's queue.

G6 Mobile User can book in advance a Visit in the market.

G7 Smart User can cancel a booking which can be either a Visit or a Reservation.

G8 Mobile User can cancel a booking which can be either a Visit or a Reservation.

1.2 Scope

The aim of the project is to develop an application which, thanks to an intuitive interface, will avoid customers waiting outside the market. In order to do that we offer customers three grocery shopping option:

- **Visit**: it's a planned appointment with given date and range time;
- **Reservation**: it consits in reserving a virtual seat in market's queue;
- **Direct Entrance**: it allows aged customers to enter without any bookings;

The main options shown in this documents are the first two in order to avoid customers to line up outside the market. Those are avaiable only for **Smart** and **Mobile Users** (their definitions are in 1.3.1 subsection).

In particular, in order to avoid to wait in line, Users will be allerted by **notifications** (Smart Users) or a **SMS** (Mobile Users). These inform them about his time schedule required to reach the market in time. Only for Smart User the application will provide the position in queue and the time to reach the market.

An alphanueric string will be provided to Mobile Users for going in and out from the market. This string for Smart User is converted in two-dimensional bar code using the standard QRCode. This must be submitted at the entry of the market.

Instead, the third option is only avaiable if an user is older than 65 years old, but with limitations. For instance they can go grocery shopping only in certain days and time slots.
In particular the Direct Entrance is avaiable only from Monday to Friday in the

range time between 9.00 A.M and 13.00 P.M. The reason for this selection is due the fact that in this ranges there are fewer customers than any other periods because are working hours.

This third option is made especially for aged people who don't have even a mobile phone.

Anyway the RASD document is not focused on this last option.

1.2.1 World

It represents the environment in which the system is placed. In particular it's composed by events which are affected by the system, but not directly connected with it. The main *World Phenomena* are:

- an User can access to the market;
- limit number of Users in the market;
- User can book his appointment for grocery shopping;
- User use his mobilephone / smartphone;

1.2.2 Machine

It represents the portion of system to be developed. The main *Machine Phenomena* are:

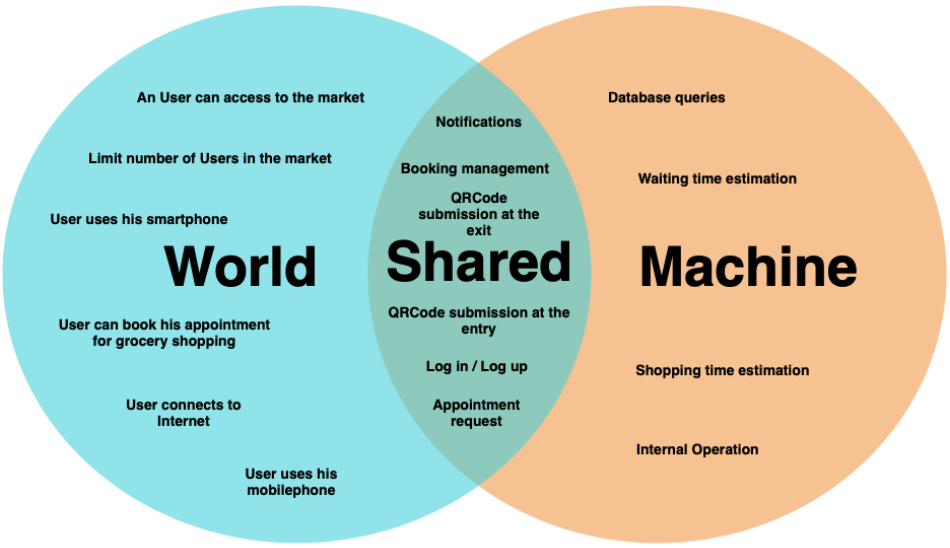
- internal operations;
- managing any bookings;
- waiting time estimation;
- data storage operations;

1.2.3 Shared Phenomena

In this model it needs a common interface to link World and Machine which is composed by the Shared phenomena.
Graphically is represent by an intersection between the World and the Machine. In this way World and Machine phenomena are observed from each other. The main application Shared Phenomena are the following:

- Notifications;
- sign in / sign up;
- Booking management;
- QRCode submission;
- appointment request;

Figure 1.1: World and Machine rapresentation



1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- **User**: Generic customer who plan to shop in the market. He/she could be a Smart or Mobile User;
- **Smart User**: User who has got a smartphone with CLup app. She/He's able to manage bookings by herself/himself;
- **Mobile User**: User who hasn't got a smartphone (only mobilephone) and so he's not able to manage Booking by himself. A Mobile User allows a Receptionist to manage his booking by calling a telephone number by interacting with him;
- **CLup Operator**: desktop application used by Receptionists to manage bookings of Mobile Users;
- **Shopping Size**: it's the size of the grocery shopping. It could be *small*, *medium*, *large* depending on the number of items that users will buy. The duration of the permanence inside the market is based on it;
- **Booking**: it indicates the generic appointment of a User in the market. It could be a Reservation or a Visit;
- **Reservation**: it's a type of booking. Users simply book a seat at the market's queue. In addition User have to indicate the Shopping Size;
- **Visit**: it's a Booking planned in advance by Users. It is planned by putting the date and the range time in which the User is going grocery shopping. In addition User have to indicate the Shopping Size;
- **Reader**: it reads QRCode at the market's entrance. It allows User to go in / out;
- **Booking submitted**: it means that the QRCode related to it is already submitted in the Reader;
- **Visit activated**: it's a Visit which already booked but not yet submitted by the User;
- **Reservation activated**: it's a Reservation which already booked but not yet submitted by the User;
- **Timestamp**: a digital record of the time of occurrence of a particular event;
- **Mirroring**: Mirroring is a technique to allow server to automatically maintain a dual backup of the system;

1.3.2 Acronyms

- **RASD**: Requirement Analysis and Specification Document;
- **HW**: Hardware;
- **SW**: Hardware;
- **API**: Application Programming Interface;
- **HTTPS**: Hypertext Transfer Protocol Secure;
- **DBMS**: Database Management System;
- **FOL**: First Order Logic;
- **UML**: Unified Modeling Language;

1.3.3 Abbreviations

- **App**: Application;
- **Gn**: n-th goal;
- **Rn**: n-th requirement;

1.4 Revision history

1.5 Reference Documents

This document is strictly based on:

- The specification of the **RASD and DD assignment** of the Software Engineering II corse, held by professor Matteo Rossi and Elisabetta Di Nitto at the Politecnico di Milano, A.Y 2020/2021;
- **Slides** of Software Engineering 2 course on BEEP;

1.6 Document Structure

Mainly the current document is divided in 4 chapters, which are:

1 **Introduction**: it aims to describe the environment and the demands taken into account for this project. In particular it's focused on the reasons and the goals that are going to be achieved with its development;

2 **Overall Description**: it's a high-level description of the system by focusing on the shared phenomena and the domain model (with its assumption);

3 **Specific Requirements**: it describes in very detail the requirements needed to reach the goals. In addition it contains more details useful for developers (i.e information about HW and SW interfaces);

4 **Formal Analysis**: this section contains a formal description of the main aspect of the World phenomena by using Alloy;

5 **Effort Spent**: it shows the time spent to realize this document, divided for each section;

6 **References**: it contains the references to any documents and to the Softwares used in this document.

Chapter 2

Overall Description

2.1 Product perspective

The project aims to build a system that manage the users' booking without wasting time while they're waiting their own turn outside the market.

An other important aspect is that the application should be for multiple market. Each user at the beginning will choose the market in which are going to go grocery shopping.
Indeed, we think that it had better that more markets adopt this system, due to increase safety in social distance.

For a **Reservation** the system provides User infomation about queue's dynamic in real time (i.e the number of people ahead). To do this, the application should try, in the best way, to estimate the time that the customers will spend in the market, in order to notify in advance users who are waiting their own turn outside.

In addition Users have to indicate how long the shopping time will be, putting potentially the **size** of the expenditure. This could be *small*, *medium* or *large*. The application analyses the customers' statistics and computes the average shopping time. The calculation considers the time range between the moments in which the User goes in and out. In this way, the system will schedule for each User *2 timestamp* by 2 notifications:

- the moment in which the User, from his position, have to leave to reach the market;
- the moment in which the User is estimated to enter in the market;

In this way, the other customers in queue will be notified as soon as it's the right time to leave and reach the market in time due to the queue's congestion and his position.

Moreover, it's possible to book a visit choosing the date and the schedule in advance in the **Visit** option.
The timetable will be splitted into 30 minutes slots, and customers will choose which they want among the free ones.

For each Visit an User can occupy at most 3 slots consecutively. In addition each market will provide customer, who don't have the application or even a smartphone, a toll-free number in order to make an appointment. This would be easy-manageble because for each market a **Receptionist** handles the booking and advises the customers with the best shopping option.
At the beginning, the user will be sign up for booking: so the receptionist will ask the required data from him and will create a customer profile in the system. The user can also manage (either *delete* or *postpone*), in a later moment, his reservation by calling the same number of the registration.
An **SMS** will provide:

- the time in which he have to access to the market;
- the identification string which will be submitted at the entrance;

Once the user finished his shopping, he will scan the code at the exit to open the doors. In the UML diagram below will be list the main classes in order to

understand how the whole system works.

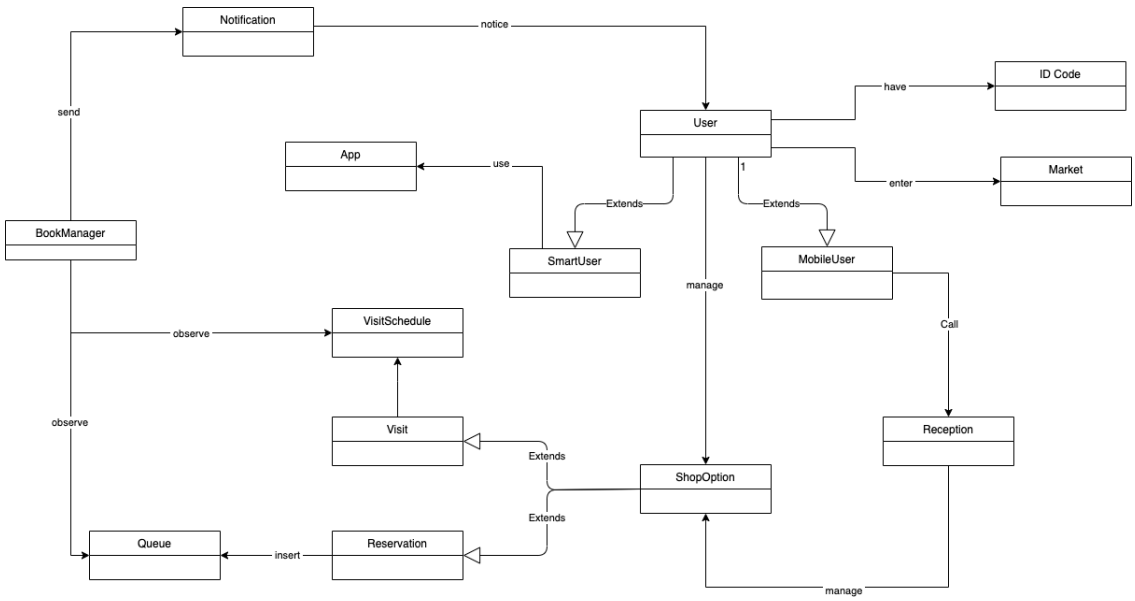


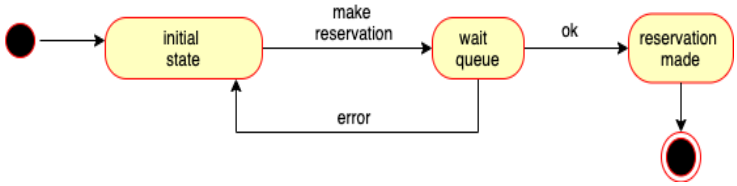
Figure 2.1: Class diagram with UML

As we can see from the class diagram (figure 2.1), the User can book once or a Visit or a Reservation. In both cases will be provide him a QRcode, which will be submitted to enter in the market. If the User decided to undo his Reservation in the queue could do it by making a cancellation from the application.

In addition, the system will notice to Smart User through a notification when it's almost his turn.

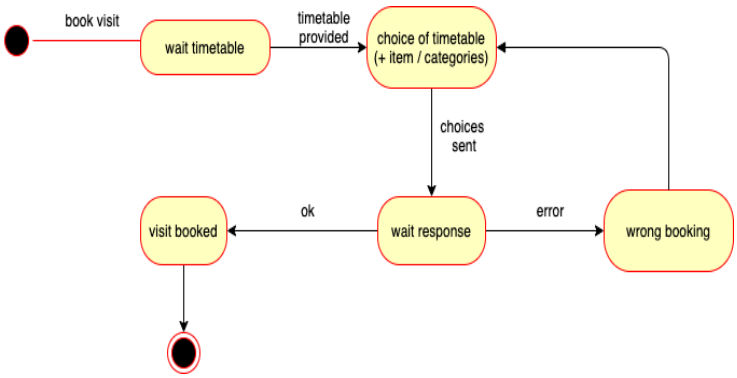
Now we will analyze the interaction between the User and the system, in order to understand possible criticities.

Figure 2.2: State diagram of the Reservation in the queue



In the first state diagram (figure 2.2) it can be osserved how a generic User can make a reservation through the system. It's sufficient making this action to be added in queue to enter in the market.
If something goes wrong the User will be redirect to the initial state.
Usually the system reject the request if the market is next to closure (or for logistic problem).

Figure 2.3: State diagram of booking a Visit



Instead, the figure 2.3 explains how to book a visit in the market. Once the timetable is provided, the generic user have to select the range time of the visit and the size bag.

If the choice made by the user is wrong (i.e timetable's slot full or market closed), the system notify him; the User will try again until his choice is correct.

2.2 Product functions

2.2.1 Queue manager

This is one of the most important function that our system must provide because it must avoid the users waiting too much their turn in the case of a wrong use of notifications. In fact, it have to foresee, through statistics from users' information, the correct time in which the users will enter in the market. This can be done thanks to the notifications sent to the user. It will also have to decide whether to accept or refuse an appointment, depending on the shop closing time and the number of people in queue.

To anticipated the unexpected, the system will use a tolerance. Let us assume that the maximum number of Users into the market is 100. Well, the system will provide only 90 users.
The remaining 10 will be used only for emergency or in the time period in which the customer will enter or exit from the market. This tolerance will be a useful resource for the system because it can manage non-deterministic event, like user's shopping time.
The system will determine the average shopping time according to the information who the user enter during his booking.

In particular accuracy of the computation depends on the quality of User's data. In this way, when the shopping time will be near to the average shopping time, the system will notice the first user in queue to reach the supermarket. Our goal is to monitor the User's influx and to keep it under critical boundary (more or less 95%). Therefore this could be possibile by slowing down the virtual queue.

2.2.2 Data Collection

The *Data Collection* is essential for the correct regulation of users' flow, because it will have to provide precise dates according to client's information. Therefore, the system will have to ask clients precise information according to keep useful knowledge for estimating the shopping time into the supermarket, without violating users' privacy.
In order to achieve this goal, it needs to oblige the user to register himself in the system and to check the item to allow the processing of his personal data, necessary to reserve virtually the seat in the queue.
One of the possible information asked could be the shopping duration. This will be used, with the entry time, to track the number of user inside the market who is finishing. Then, will be possibile to notify in advance users in queue about the closeness of their turn.

2.3 User characteristics

We distinguish the actors into our application based on actions and interactions with the external world:

- **Mobile User**: She/He's a User that hasn't a phone able to running CLup. The user will interface with receptionist to registers and manages the own bookings;
- **Smart User**: She/He's a User that has downloaded the CLup and uses it to manages the own bookings;
- **Receptionist** : A Market employee that receive the Mobile Users calls and help them to manages their bookings and account settings;
- **Security**: A Market employee, located at the entrance that thanks to business phone will manage the access to supermarket checking the clients QR Code.

2.4 Assumptions, dependencies and constraints

D1 The system can delete the Reservation when the Smart User accumulates a delay to reach the market greater than 10 minutes;

D2 The system can delete the Reservation the Mobile User accumulates a delay to reach the market greater than 15 minutes;

D3 The system handles the threshold number of Users allowed in the market;

D4 The Smart User have to be connected to Internet through Wi-Fi/Cellular network;

D5 The Mobile User have to be connected to his own mobile operator;

D6 A Visit is associated to a Date and a period of time (start/end time);

D7 A Booking is associated to one and only one QRCode;

D8 User must have one and only one Visit activated;

D9 User must have one and only one Reservation activated;

D10 A Booking belongs to one and only one User;

D11 The time of entrance plus the duration of the grocery shopping mustn't exceed the time in which the market will be closed;

D12 Each Date and Slot Time must contain at most N threshold value depended on the market's area;

D13 An User mustn't have a Reservation and a Visit active and submitted at the same time;

Chapter 3

Specific Requirements

3.1 External Interface Requirements

The following mockups provide a model of the graphic interface provided by the system.
We will create a user interface that will be user friendly in order to be easy to use also from non-technical users.
The mockups has the aim to illustrate the main aspects of the CLup and CLup operator.

3.1.1 User Interfaces (CLup)

The development of CLup app aims to have a easy-to-use interface for User who has a Smartphone. In particular the application have to be intuitive and reliable in order to give Users the possibility to book an appointment without any obstacles.
The following mockups show the main operation allowed by Clup to the Smart User:

Figure 3.1: Login: Users already registered signes in with their credential.

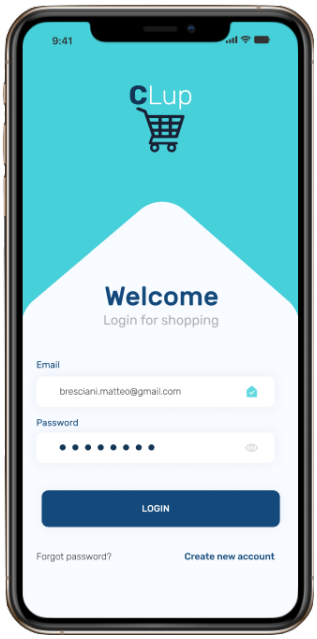
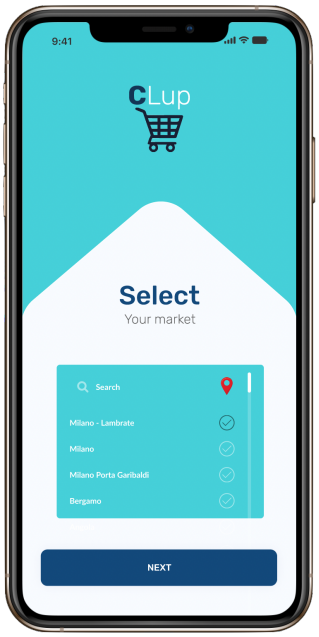
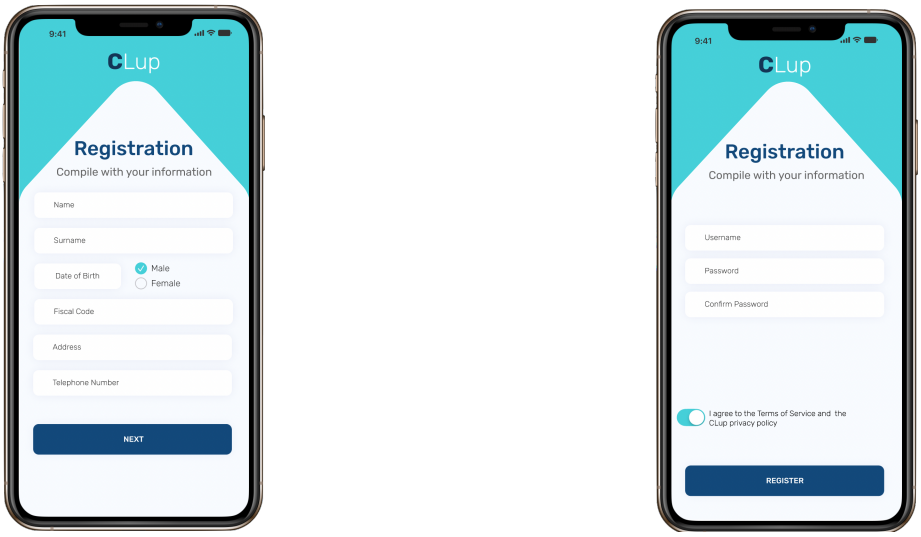


Figure 3.2: Market selection: After the login, the user have to select in which market wants to go shopping, due to his position.





(a) Page 1.

(b) Page 2.

Figure 3.3: Registration: Users not registered can sign up himself putting his own data, e-mail and password. In addition Users have to accepts the Term of Service and the CLup privacy policy to proceed.

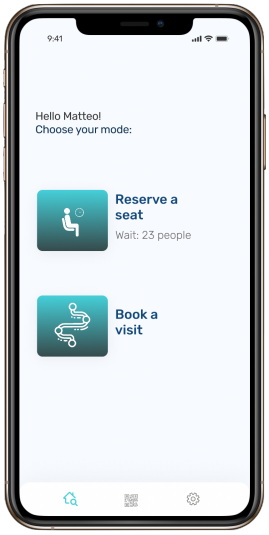
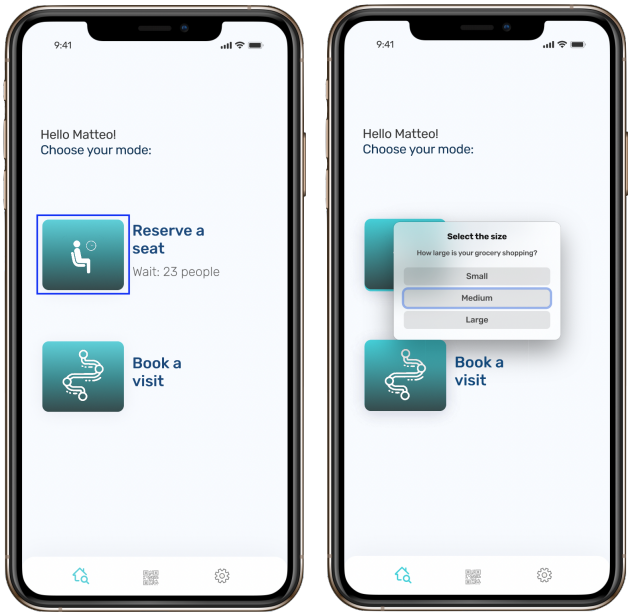
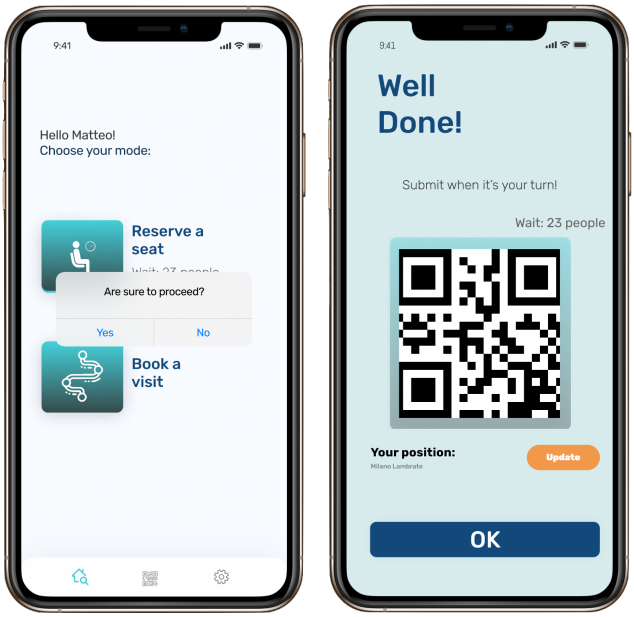


Figure 3.4: Home: Homepage of CLup from which the User can select to book a Visit or a Reservation.

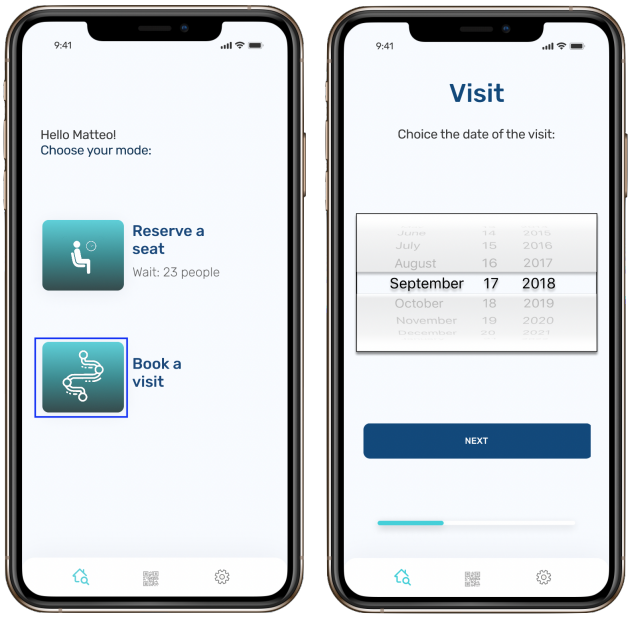


(a) User chooses Reservation. (b) Shopping size selection.



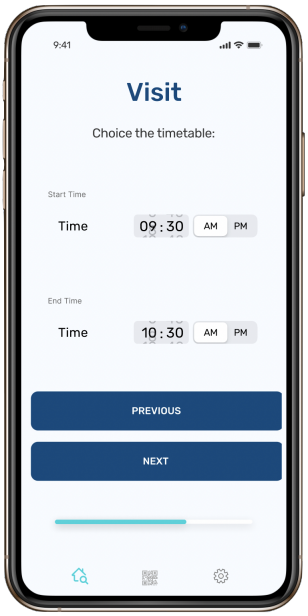
(c) Booking confirm. (d) Reservation's QRCode.

Figure 3.5: Reservation: The Smart User can book a Reservation with the following steps.

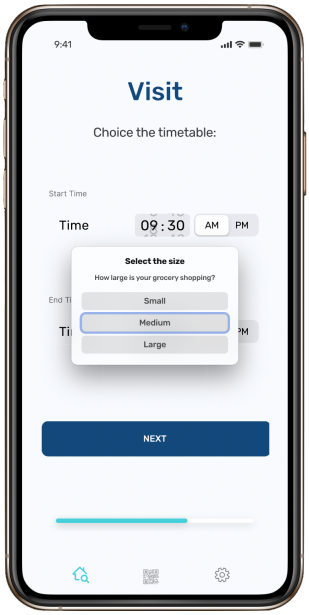


(a) Smart User chooses Visit

(b) Date selection.



(c) Time period selection.

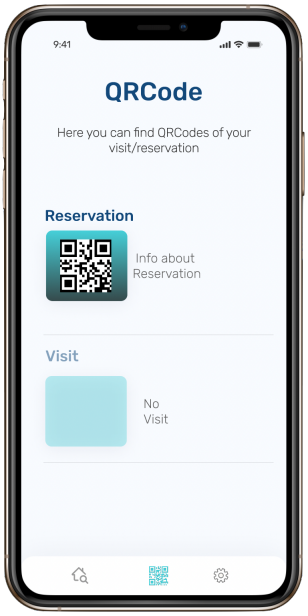


(d) Shopping size selection.

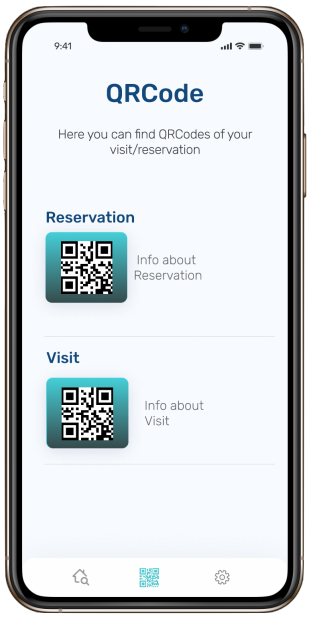
Figure 3.6: Visit: The Smart User can book a Visit with the following steps. At the end Visit's QRCode is provided.

Figure 3.7: QRCode section: the User can manage his booking. A Smart User can cancel, delay (only for Reservation), or visualize his QRCode in order to access to the market.

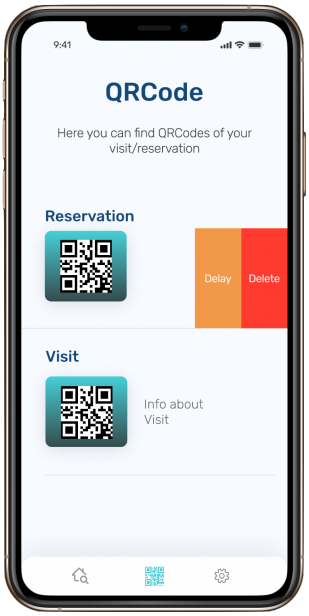
(a) Only Reservation booked.



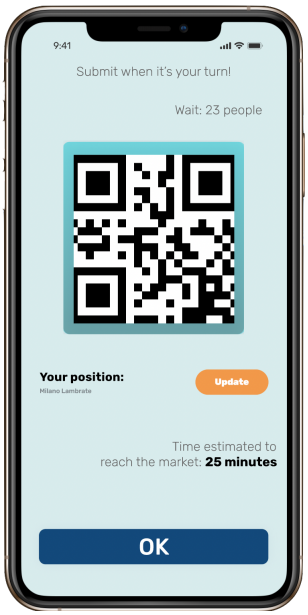
(b) Visit and Reservation booked.

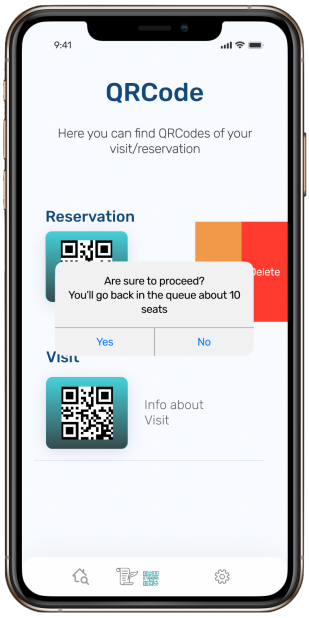


(c) Smart User swipes left.

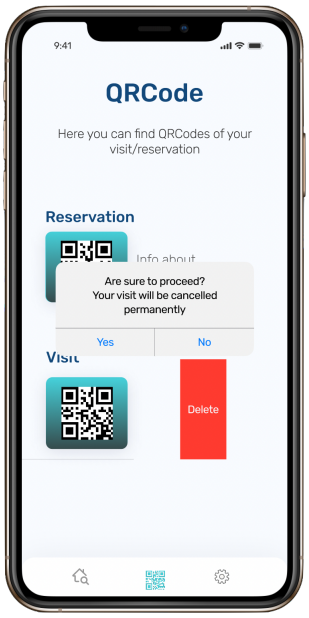


(d) Booking's QRCode.



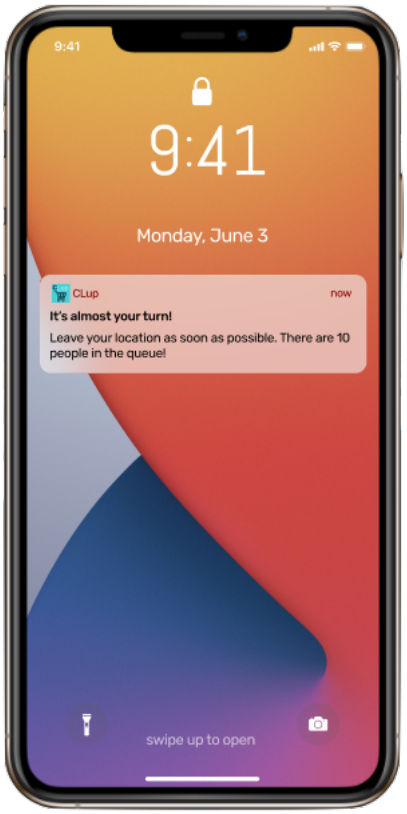


(e) Delay confirmation of a Reservation

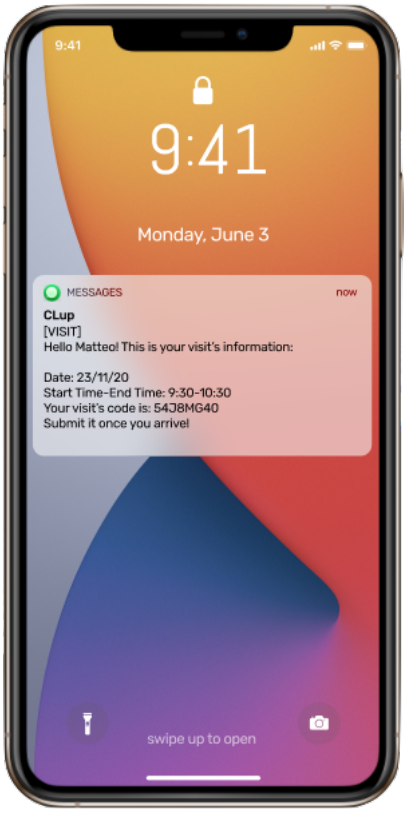


(f) Cancellation confirmation of a Visit

Figure 3.8: Cancel and Delay action: A alert will be shown to Smart User to confirm his action.



(a) A notification is sent to a Smart User to inform him to leave in order to reach the market in time.



(b) An SMS is sent to a Mobile User with booking information of his Visit. In particular contains the Booking's schedule and the Identification Code to submit.

Figure 3.9: Notification and SMS: User receive his information about his Booking depending on whether is a Smart or Mobile User.

3.1.2 Receptionist Interfaces (CLup Operator)

CLup Operator, the desktop app, is introduced to give Users who has no Smartphone the possibility to book an appointment at the market. Mobile Users have to call a Receptionist through a customer service number.
The Receptionist so aims to interact between the User and the system in order to manage his appointment, acting like an intermediary. Even this application must be simple, in order to allow Receptionist to interact with Mobile User in a proper and effective way.
The following mockups show the main operation allowed by Receptionist through his application to manage Booking of the Mobile Users.

Figure 3.10: Receptionist signes in with their credential.

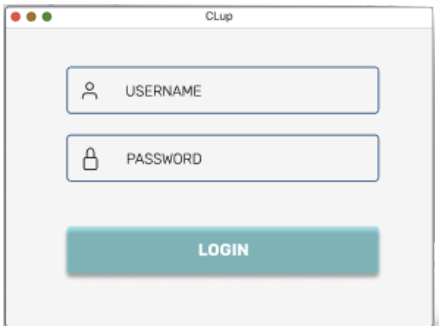


Figure 3.11: Receptionist can register a new Mobile User or select a new one if it's already registered.

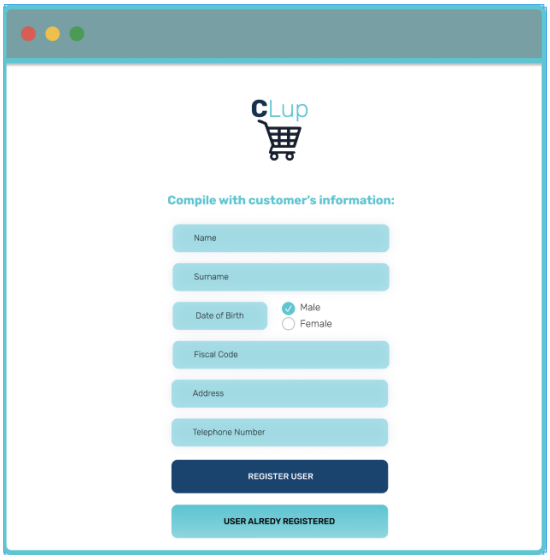


Figure 3.12: Receptionist selects the Mobile User already registered.

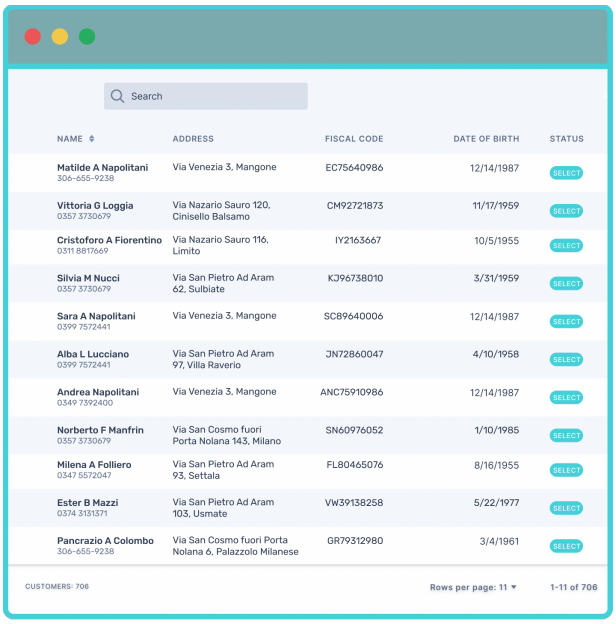


Figure 3.13: Receptionist can manage Mobile User's Booking.

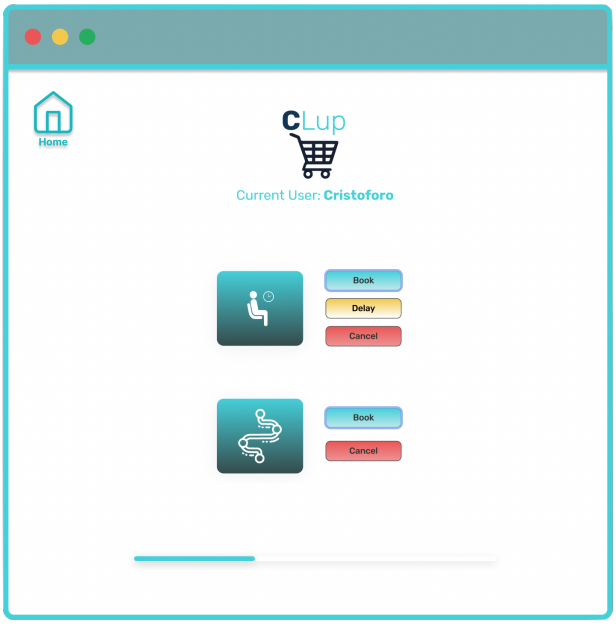


Figure 3.14: Receptionist selects the grocery shopping size told by Mobile User.

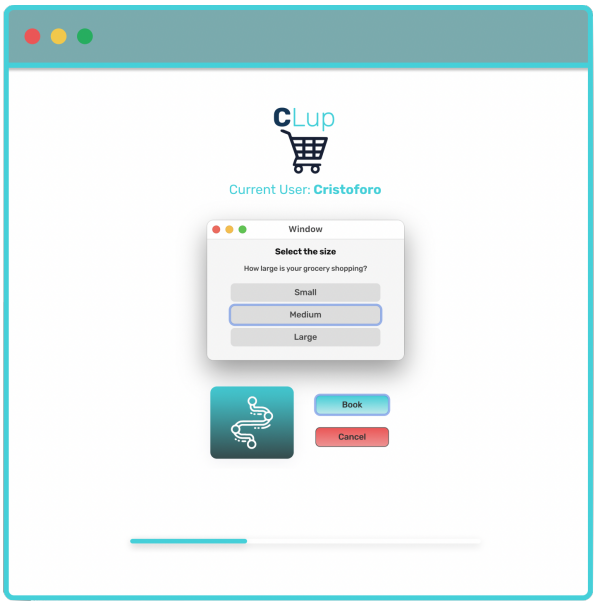


Figure 3.15: Receptionist inserts visit information told by Mobile User.

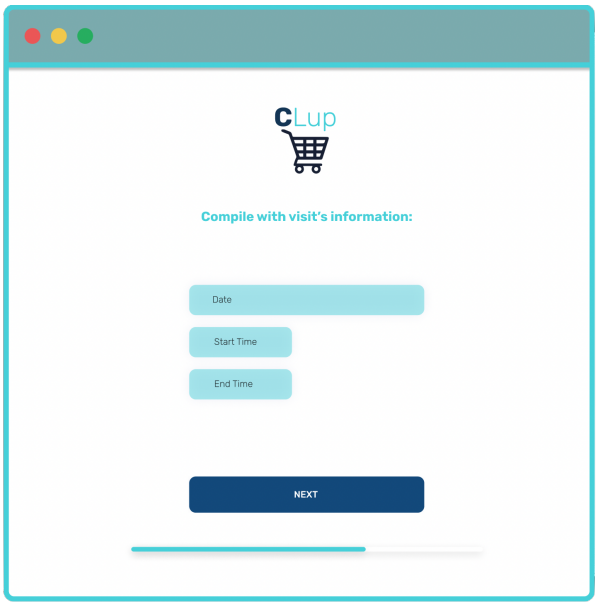
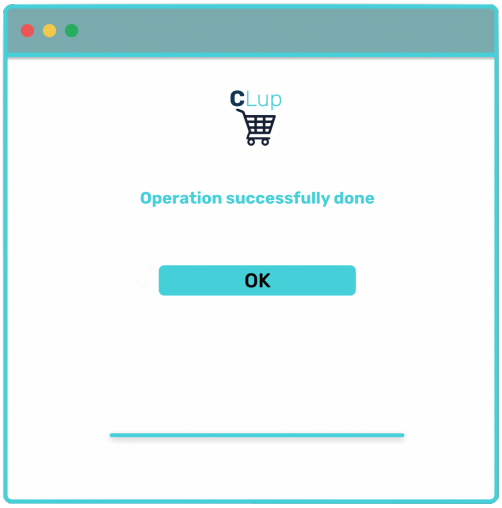


Figure 3.16: Booking made by Receptionist is finished.



3.1.3 Hardware Interfaces

The supermarket will have two scanners associated to an employee's device in order to validate the booking:

- One at the entry, that will read QR code and will confirm client reservations in case they will be valid.
  The scanner will reject reservation if QR code will result invalid or will have passed to much time from its call.
- One at the exit, that when clients will finish shopping allow them to exit to supermarket opening the doors.
  Another usage of exit QR code is to monitor the numbers of client into the shop.

The scanners will be used to obtain information about client shopping time.

3.1.4 Software Interfaces

CLup notices User when leave in order to reach the market in time also thanks to Google Maps APIs. Infact CLup use them in order to determine how much time a User have to put due to his position. The choice of using Google Maps APIs derives from the easy usability, frequency update and very large documentation. The system, in addition to CLup and CLup Operator application doesn't use any external interfaces.

3.1.5 Communication Interfaces

Every User uses a HTTPS protocol to make request to CLup's servers. In this way informations sent are safe due to encryption provided by the TLS protocol. The same is used by Receptionist to communicate with the server.

3.2 Functional Requirements

G1 User enters once arrived at the market.

- **R1** Smart User can receive the first notification to reach the market;
- **R2** Mobile User can receive an SMS needed to enter / exit from the market;
- **R3** The system provide the time estimation to reach the market;
- **R4** The User chooses the startTime / endTime of the Visit;
- **R5** The User can postpone his turn by 10 turns;
- **R8** The system provides the number Users in queue;
- **D1** The system can delete the Reservation when the Smart User accumulates a delay to reach the market greater than 10 minutes;
- **D2** The system can delete the Reservation the Mobile User accumulates a delay to reach the market greater than 15 minutes;
- **D3** The system handles the threshold number of Users allowed in the market
- **D4** The Smart User have to be connected to Internet through Wi-Fi/Cellular network
- **D5** The Mobile User have to be connected to his own mobile operator
- **D6** A Visit is associated to a Date and a period of time (start/end time);
- **D7** A booking is associated to one and only one QRCode;

G2 Put a limit to the number of Users in the market.

- **R4** The User chooses the startTime / endTime of the Visit;
- **R6** The system allows the User to enter/exit from the market with the QRCode submission;
- **R7** The system store the enter/exit time from the market after the QRCode submission;
- **D3** The system handles the threshold number of Users allowed in the market;

G3 Smart User can make a Reservation of a seat in the market's queue.

- **R8** The system provides the number Users in queue;
- **R10** The Smart User must be registered;
- **R11** The Smart User must be already logged in;
- **R15** The User must select a date in which the market is opened;

R18 The system provides Smart User the QRCode when the booking is done;

D3 The system handles the threshold number of Users allowed in the market;

D4 The Smart User have to be connected to Internet through Wi-Fi/Cellular network

D7 A Booking is associated to one and only one QRCode

D9 User must have one and only one Reservation activated

D10 A Booking belongs to one and only one User;

D11 The time of entrance plus the duration of the grocery shopping mustn't exceed the time in which the market will be closed;

D13 An User mustn't have a Reservation and a Visit active and submitted at the same time;

G4 Smart User can book in advance a Visit in the market.

R10 The Smart User must be registered;

R11 The Smart User must be already logged in;

R14 User must choose the size of his grocery shopping between small, medium and large;

R15 The User must select a date in which the market is opened;

R16 The User must select a start and end time available;

R18 The system provides Smart User the QRCode when the booking is done;

D3 The system handles the threshold number of Users allowed in the market;

D4 The Smart User have to be connected to Internet through Wi-Fi/Cellular network

D6 A Visit is associated to a Date and a period of time (start/end time);

D7 A Booking is associated to one and only one QRCode

D8 User must have one and only one Visit activated;

D10 A booking belongs to one and only one User;

D11 The time of entrance plus the duration of the grocery shopping mustn't exceed the time in which the market will be closed;

D12 Each Date and Slot Time must contain at most N threshold value depended on the market's area;

D13 An User mustn't have a Reservation and a Visit active and submitted at the same time;

G5 Mobile User can make a Reservation of a seat in the market's queue.

R2 Mobile User can receive an SMS needed to enter / exit from the market;

R8 The system provides the number Users in queue;

R12 The Mobile User must be registered;

R13 The Mobile User provide personal data to the Receptionist;

R14 User must choose the size of his grocery shopping between small, medium and large;

R17 The Mobile User can call the Receptionist;

R21 Receptionist must be logged in;

R22 Mobile User must call the Receptionist with his personal telephone number;

D3 The system handles the threshold number of Users allowed in the market;

D5 The Mobile User have to be connected to Internet through his own mobile operator;

D10 A booking belongs to one and only one User;

D11 The time of entrance plus the duration of the grocery shopping mustn't exceed the time in which the market will be closed;

D13 An User mustn't have a Reservation and a Visit active and submitted at the same time;

G6 Mobile User can book in advance a Visit in the market.

R2 Mobile User can receive an SMS needed to enter / exit from the market;

R12 The Mobile User must be registered;

R13 The Mobile User provide personal data to the Receptionist;

R14 User must choose the size of his grocery shopping between small, medium and large;

R15 The User must select a date in which the market is opened;

R16 The User must select a start and end time available;

R17 The Mobile User can call the Receptionist;

R21 Receptionist must be logged in;

R22 Mobile User must call the Receptionist with his personal telephone number;

D3 The system handles the threshold number of Users allowed in the market;

D5 The Mobile User have to be connected to Internet through his own mobile operator;

D10 A booking belongs to one and only one User;

D11 The time of entrance plus the duration of the grocery shopping mustn't exceed the time in which the market will be closed;

D12 Each Date and Slot Time must contain at most N threshold value depended on the market's area;

D13 An User mustn't have a Reservation and a Visit active and submitted at the same time;

G7 Smart User can cancel a booking which can be either a Visit or a Reservation.

R10 The Smart User must be registered;

R11 The Smart User must be already logged in;

R19 or R20 User must have an activated Visit/Reservation not yet submitted;

D3 The system handles the threshold number of Users allowed in the market;

D4 The Smart User have to be connected to Internet through Wi-Fi/Cellular network;

D10 A booking belongs to one and only one User;

G8 Mobile User can cancel a booking which can be either a Visit or a Reservation.

R12 The Mobile User must be registered;

R13 The Mobile User provide personal data to the Receptionist;

R17 The Mobile User can call the Receptionist;

R19 or R20 User must have an activated Visit/Reservation not yet submitted;

R21 Receptionist must be logged in;

R22 Mobile User must call the Receptionist with his personal telephone number;

D3 The system handles the threshold number of Users allowed in the market;

D5 The Mobile User have to be connected to Internet through his own mobile operator;

D10 A booking belongs to one and only one User;

Requirements

R1 Smart User can receive the first notification to reach the market;

R2 Mobile User can receive an SMS needed to enter / exit from the market;

R3 Smart User can receive the second notification because is his turn;

R4 The User chooses the startTime / endTime of the Visit;

R5 The User can postpone his turn by 10 turns;

R6 The system allows the User to enter/exit from the market with the QRCode submission;

R7 The system store the enter/exit time from the market after the QRCode submission;

R8 The system provides the number Users in queue;

R9 Google Maps APIs provides the estimation time to reach the market;

R10 The Smart User must be registered;

R11 The Smart User must be already logged in;

R12 The Mobile User must be registered;

R13 The Mobile User provide personal data to the Receptionist;

R14 User must choose the size of his grocery shopping between small, medium and large;

R15 The User must select a date in which the market is opened;

R16 The User must select a start and end time available;

R17 The Mobile User can call the Receptionist;

R18 The system provides Smart User the QRCode when the booking is done;

R19 User must have an activated Visit not yet submitted;

R20 User must have an activated Reservation not yet submitted;

R21 Receptionist must be logged in;

R22 Mobile User must call the Receptionist with his personal telephone number;

3.3 Use Case

3.3.1 User

Scenario 1 Giovanna, a career woman who is always in trouble to find free time, needs to go grocery shopping for her family.
Indeed, once she finished working, she goes to the market and, due to the lockdown, have to wait in line for hours to have access to it. The result is that, coming back home later, she can't put some time in her children. However, in the past days, she discovered CLup App which allows her to book a visit in the market in advance, by only putting the range time avaiable and the size of the expenditure.
In this way, Giovanna will save a lot of time and will stay longer with her children, instead of waiting in line outside the market.

Nevertheless, due to Covid-19 emergency, the market will be always filled.

Scenario 2 Jonhathan, on the advice of his grandchild, bought a new smartphone. In addition started using it and installing usefull applications like CLup.
In particular, with it, he'll be able to make a reservation in market's queue. In this way, CLup will notify him when, accordingly to his position, leave to get the market in time for his turn.
Finally, once he arrived, can go in there by simply scanning the QRCode sent before. At the end Jonhathan will go shopping without waiting on feet his turn during a cold winter's day.

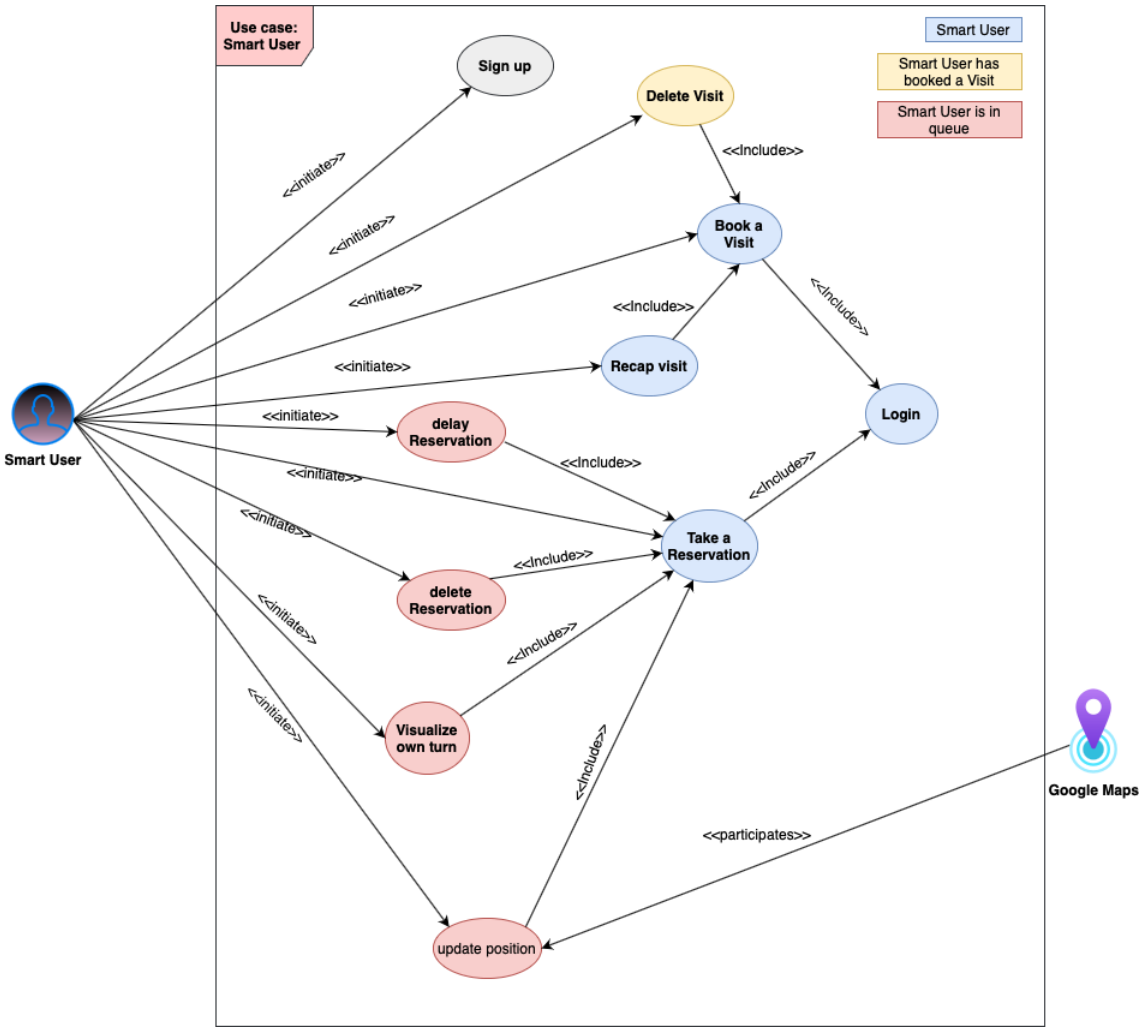


Figure 3.17: Use case of the Smart User.

Name	**Login**
Actor	**Smart User**
Entry condition	• User is already registered with his credential. • User open the application.
Events flow	• User open the application. • The fields of username and password are filled by the User. • User send the request to the system by clicking ”Login”.
Exit condition	User is logged in and the homepage of CLup is provided.
Exceptions	• Username inserted by the User is not registered in the system yet. • Password inserted by the User is wrong. • Communication with the Server is unavailable.

Table 3.1: Login use case (smart User).

Name	**Sign up**
Actor	**Smart User**
Entry condition	• User open the application. • User is not registered with his credential yet.
Events flow	• User selects “Create new account”. • The fields of sensitive data, username and password are filled by the User. • User accepts CLup privacy policy by checking its box. • User clicks on ”Register” button to send to the Server his request to be registered.
Exit condition	• User is registered and already logged in. • CLup provides a page where User will select his market.
Exceptions	• Username inserted by the User is already registered in the system. • Userser did not accept CLup privacy policy. • User has inserted forbidden characters in one or more fields. • Communication with the Server is unavailable.

Table 3.2: Use case of sign up (Smart User).

Name	**Book a Visit**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User hasn't a Visit not yet submitted.
Events flow	• User clicks on "Homepage" option. • User clicks "Book a visit" button. • User selects the date and the time in which want to schedule the Visit. • User chooses the shopping size of his grocery shopping. • User confirms and sends the request to the Server by clicking "Next".
Exit condition	• The user has booked his Visit. • QRCode associated to the Visit is provided to the User.
Exceptions	• The time table choosen by the User is unavailable or already full. • Communication with the Server is unavailable.

Table 3.3: Use case of Visit booking (Smart User).

Name	**Take Reservation**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User hasn't a Reservation not yet submitted.
Events flow	• User clicks on "Homepage" option. • User clicks "Reserve a seat" button. • User confirms and sends the request to the Server by clicking "Yes".
Exit condition	• User is inserted in the queue. • QRCode associated to the Reservation is provided to the User.
Exceptions	• The market choosen is closed in the current moment of the request. • Communication with the Server is unavailable.

Table 3.4: Use case of Reservation booking (Smart User).

Name	**Delete Visit**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Visit not yet submitted.
Events flow	• User clicks on “QRCode” menù. • Visit and Reservation booked are provided. • User clicks on “Delete” button of the Visit. • User send the cancellation request to the Server by confirming it with ”Yes”.
Exit condition	• The system deletes the User’s Visit from his application. • The system release the seat occupied in the slot/slots selected by the User’s Visit.
Exceptions	Communication with the Server is unavailable.

Table 3.5: Use case of Visit delete (Smart User).

Name	**Delete Reservation**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Reservation not yet submitted.
Events flow	• User clicks on “QRCode” menù. • Visit and Reservation booked are provided. • User clicks on “Delete” button of the Reservation. • User send the cancellation request to the Server by confirming it with ”Yes”.
Exit condition	• The system deletes the User’s Reservation from his application. • The system release the seat occupied in the queue by the User.
Exceptions	Communication with the Server is unavailable.

Table 3.6: Delete Reservation use case (Smart User).

Name	**Delay Reservation**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Reservation not yet submitted.
Events flow	• User clicks on “QRCode” menù. • Visit and Reservation booked are provided. • User clicks on “Delay” button of the Reservation. • User send the delay request to the Server by confirming it with ”Yes”, in order to go back in the queue by 10 seats.
Exit condition	• The user turn will be shift ten places after. • The delay button will be disabled.
Exceptions	• Queue is too small for a delay. • Delay option was already used by the User for this Reservation. • Communication with the Server is unavailable.

Table 3.7: Delay Reservation use case (Smart User).

Name	**Recap Visit**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Visit not yet submitted.
Events flow	• User clicks on “QRCode” menù. • User clicks on the QRCode of the Visit.
Exit condition	CLup provided: • QRCode; • startTime of the Visit; • endTime of the Visit;
Exceptions	

Table 3.8: Recap Visit use case (Smart User).

Name	**Visualize turn in queue**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Reservation not yet submitted.
Events flow	• User clicks on “QRCode” menù. • User clicks on the QRCode of the Reservation.
Exit condition	CLup provided: • QRCode; • Number of people ahead the User in the queue; • Time estimated to reach the market;
Exceptions	

Table 3.9: Use case of turn in queue visualization.

Name	**Update position**
Actor	**Smart User**
Entry condition	• User has already logged in CLup with his credential. • User has already booked a Reservation not yet submitted.
Events flow	• User clicks on “QRCode” menù. • User clicks on the QRCode of the Reservation. • User clicks on ”Update” botton.
Exit condition	The system, through Google Maps, provided the time estimated to reach the market depending on his position.
Exceptions	• The GPS position too far from market. • GPS is unavailable.

Table 3.10: Update position use case.

Name	**Market selection**
Actor	**Smart User**
Entry condition	User has already logged in CLup with his credential.
Events flow	• User selects the market in which wants to go grocery shopping. • User confirms and sends his choice clicking on ”Next”.
Exit condition	The homepage of CLup is provided.
Exceptions	Communication with the Server is unavailable.

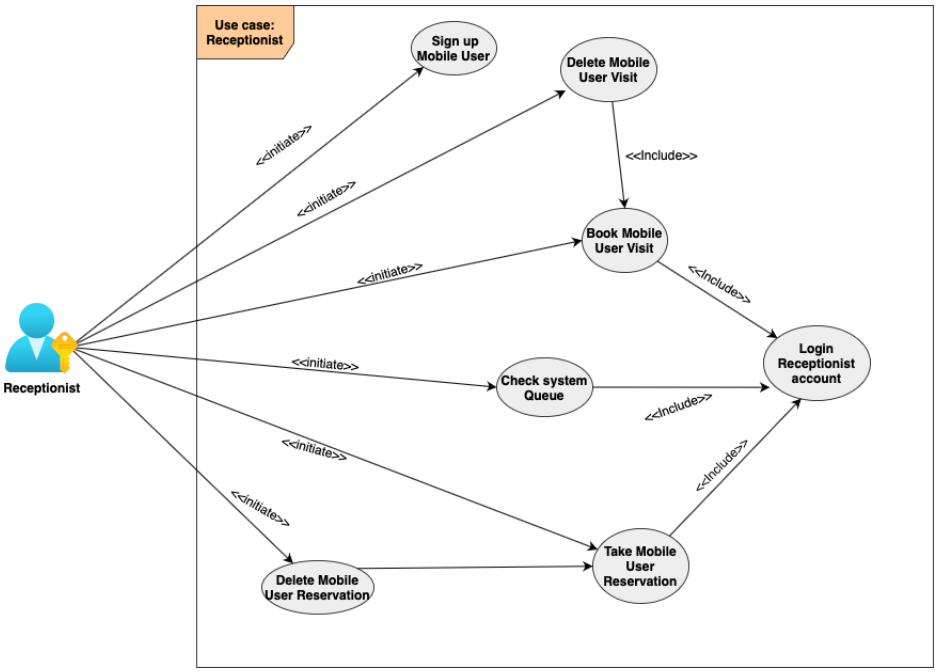
Table 3.11: Use case of market selection.

3.3.2 Reception

Scenario 3

Gustavo, an elderly person, he discovered recently a new time saver and usefull service at the market.
It consists to book a visit at the market by simply calling the number found in an advertisment.
Due to the fact that Gustavo is sick of waiting too much in the queue decide to call this market number to book the visit.
On the other side Marta, a gentle receptionist who works for the market, answers to Gustavo's call; she takes care of the registration of his own data, the credentials and the all visit information (i.e data and range time).
Gustavo will be notified about the appointment with an SMS on his mobilephone in time.
In addition the SMS will provide the schedule for the visit and the code which will be submitted at the entrance.

Figure 3.18: Use case of the Receptionist.



Name	**Sign up Mobile User**
Actor	**Receptionist**
Entry condition	• Receptionist has already logged in CLup Operator. • The Mobile User is not registered yet by a Receptionist. • Mobile User calls the Receptionist to register himself.
Events flow	• Receptionist asks Mobile user the required sensitive data. • Receptionist fills the fields of the registration request. • Receptionist confirm and send the registration to the Server.
Exit condition	The Mobile User is registered in the CLup system.
Exceptions	• The phone call was aborted.

Table 3.12: Use case of sign up (Mobile User).

Name	**Delete Mobile User Visit**
Actor	**Receptionist**
Entry condition	• Receptionist has already logged in CLup Operator. • Mobile User is already registered by a Receptionist. • User has already booked a Visit not yet submitted. • Mobile User calls the Receptionist to delete his Visit.
Events flow	• User provides his sensitive data to the Receptionist in order to be recognized by him. • Receptionist selects the Mobile User once he find him in th system. • Receptionist has accessed to the booking's information of the Mobile User. • Receptionist sends the cancellation request to the Server by clicking the "delete" button near the Visit.
Exit condition	• The system deletes the User's Visit from his application. • The system release the seat occupied in the slot/slots selected by the User's Visit.
Exceptions	Communication with the Server is unavailable.

Table 3.13: Use case of Visit delete (Mobile User).

Name	**Book Mobile User Visit**
Actor	**Receptionist**
Entry condition	• Receptionist has already logged in CLup Operator. • Mobile User is already registered by a Receptionist. • User hasn't a Visit not yet submitted. • Mobile User calls the Receptionist to book his Visit.
Events flow	• User provides his sensitive data to the Receptionist in order to be recognized by him. • Receptionist selects the Mobile User once he find him in th system. • Receptionist has accessed to the booking's information of the Mobile User. • Recepetionist clicks on "Book" button near Visit. • Receptionist selects the date and the time in which Mobile User wants to schedule the Visit. • Receptionist chooses the shopping size of his grocery shopping. • Receptionist confirms and sends the request to the Server by clicking "Next".
Exit condition	• The user has booked his Visit • SMS is sent to the Mobile User with the string code and Visit schedule informations.
Exceptions	• The time table choosen by the Mobile User is unavailable or already full. • Communication with the Server is unavailable.

Table 3.14: Book Visit use case (Mobile User).

Name	**Take Mobile User Reservation**
Actor	**Receptionist**
Entry condition	• Receptionist has already logged in CLup Operator. • Mobile User is already registered by a Receptionist. • User hasn't a Reservation not yet submitted. • Mobile User calls the Receptionist to delete his Reservation.
Events flow	• User provides his sensitive data to the Receptionist in order to be recognized by him. • Receptionist selects the Mobile User once he find him in th system. • Receptionist has accessed to the booking's information of the Mobile User. • Receptionist sends the request to the Server by clicking on "book" button near Reservation.
Exit condition	The user has been queued
Exceptions	The queue is full

Table 3.15: Use case of Reservation booking (Mobile User).

Name	**Delete Mobile User Reservation**
Actor	**Receptionist**
Entry condition	• Receptionist has already logged in CLup Operator. • Mobile User is already registered by a Receptionist. • User has a Reservation not yet submitted. • Mobile User calls the Receptionist to delete his Reservation.
Events flow	• User provides his sensitive data to the Receptionist in order to be recognized by him. • Receptionist selects the Mobile User once he find him in th system. • Receptionist has accessed to the booking's information of the Mobile User. • Receptionist sends the request to the Server by clicking on "delete" button near Reservation.
Exit condition	• The system deletes the User's Visit from his application. • The system release the seat occupied in the slot/slots selected by the Mobile User's Visit.
Exceptions	The user has not booked any reservation

Table 3.16: Use case of Reservation delete (Mobile User).

Name	**Login Receptionist account**
Actor	**Reception**
Entry condition	The receptionist opens CLup Operator on desktop device.
Events flow	• Receptionist fills his credentials. • Receptionist send the request to the Server to be authenticated and handle Mobile Users' booking by clicking "Login" button.
Exit condition	• The receptionist has logged in CLup Operator. • CLup Operator provides the homepage where can handle bookings.
Exceptions	• Username inserted by the Receptionist is not registered in the system yet. • Password inserted by the Receptionist is wrong. • Communication with the Server is unavailable.

Table 3.17: Use case of Receptionist's login.

3.3.3 Sequence Diagrams

In this section are reported the main operation (Reservation and Visit) made by Users to make an appointment in the market.
Note that are attached request of User with a Smartphone. Requests for Mobile Users are the same, with a difference: the presence of the actor Receptionist, who acts as a proxy in order to manage the requests.

Figure 3.19: Sequnce diagram of a Visit request from Smart User.

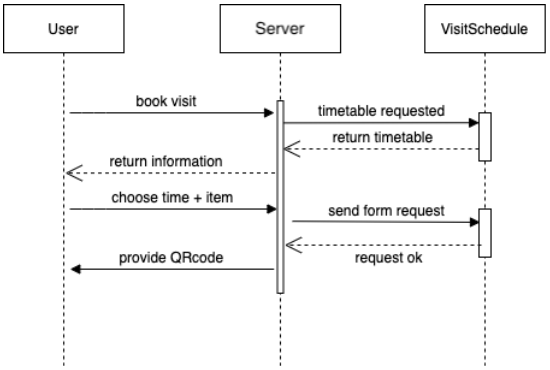
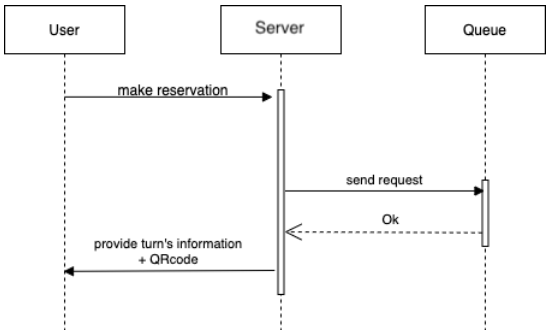


Figure 3.20: Sequnce diagram of a Reservation request from Smart User.



3.4 Performance Requirements

The system has been designed to give possibility to Users to make and to manage their booking every time they need. So it's essential that the CLup's server have to handle a big number of connections.
The most reliable requests must be about the booking. User could book his appointment when he needs without any error or denial. Anyway there's one exception: the booking of a reservation is not avaialable when a market is closed.

It's important that the server has enough resources to accept a large number of request. The number will be estimated depending on the users demand.

3.5 Design Constraints

3.5.1 Standards compliance

Especially, the system will be released in the main digital distribution platform (such as App Store or Google Play). So it must follow their guidelines in order to have a proper and a lawful distribution.
In addition, due to the fact that it retrives and analyses many sensitive data, application must respect tha main privacy guidance. In particular in Europe must follow the General Data Privacy Regulation (GDPR), due to a safe and aware processing of data.

3.5.2 Users in the market

An important constraint that must applied is about the number of presences in the market of the Users. In fact, due to the Covid-19 pandemic, it's important that Users respect the social distance rule that is at least 1 meter.
So it's necessary that the market's provide the maximum number of Users allowed in the market that is, in a general way, it will be indicated with N.

3.5.3 Hardware limitations

- 3G/4G/5G connection: they're essential due to server connection to compile booking request;
- GPS: it's used to allow user to estimate the time spending to reach the market in time. But it's not mandatory for booking;

3.5.4 Any other constraint

Mobile User telephone number. Every Mobile User in the system provides at the registration many data but only one will be subject to a constraint: the telephone number. Indeed, the system memorize it, being for him the unique allowed to call the Receptionist for managing his booking.

The respective recognition made for a Smart User are their credential (e-mail and password).

3.6 Software System Attributes

3.6.1 Reliability

The application will be tested for each update for verifying the stability and correctness. Furthermore, will be scheduled a periodic maintenance to ensure the system operation. To improve fault tolerance will be used a technique like **mirroring**.

3.6.2 Availability

Due to the fact that nowadays grocery shopping is avaiable almost all day, the availability of the system is very high. So the required availability is close to 99%.
However, the reservation function has a lower avaibility because of the inability to book a seat in queue if the market is closed in some hours of the day.

3.6.3 Security

Security is one of the most critical aspect of CLup. So User's sensitive data are stored safely in DBMS accessibile only by strict level of privilege.
In addition, communication towards the application server (like login or booking requests) are implemented using HTTPS protocol, which, with TLS protocol, ensure the encryption of every packets. Another important risk that must by avoid is the Denial of Service, which occurs when the server is unavailable due to incoming traffic flooding from malevoulous Users.
So it's important that the Server mitigates this, by using for instance SYN cookies or any possible arrangements in order to allocate less resources if it's not necessary.

3.6.4 Maintainability

The application implementation must be oriented towards an high scalability in order to gurantee an efficient and cheaper maintenance. This could be done by using design patterns.

3.6.5 Portability

The system must be smoothly portable almost for the main smartphone on the market (i.e Android or iOS). So CLup must be distributed for the main mobile store (i.e App Store and Google Play), coding it with the main program languages such as Android or Swing. Instead, CLup Operator must be compatible with the main Operating system such as Windows and Mac OS X.

Chapter 4

Formal Analysis Using Alloy

.

4.1 Alloy Model

In this section we show the model of the system, by formalizing it with Alloy. This step is strictly important, due to its consistency. In particular it describes the basic structure and the behavior of our system based on FOL. In our alloy model we will show a typical day in the market.
We will describe the main objects involved as signature, with relative constraints, trying to reach a compromise between readability and complexity.

We will exclude from the model some classes, and some tests, because they are useless to describe the model. For example, we will exclude:

- the *Elderly people* that are not involved with Reservation/Visit and therefore they will enter directly in the market in certain time slots;
- the *Receptionist* that acts as intermediary between Mobile User and the system.
- *Market maximum capacity test*: even if this is one of most important goal to reach, we could not describe it in a sufficient way.

In the following model, we discuss the possible choices that the users can do, such as booking a Reservation/Visit, and the constraints imposed by their choices. In particular we describe:

- *Reservation and its insertion* on the Queue with the number assigned to it;
- *Visit booking* between the time slots entered by the users and the shopping time based on its size choosen.

An other important aspect it's the formalization of the QRCode. Users enters in the market with a QRCode if and only if it's valid but not submitted yet. Otherwise, with different boolean values the QRCode signature could have different meaning. All possibile type of QRCode could be:

- *Valid and not submitted*: QRCode owner has booked an appointment (either a Reservation or a Visit) and he hasn't entered yet;
- *Submitted and valid*: the QRCode owner is in the market ;
- *Submitted and not valid*: the QRCode is already used by the owner;

The model is organized temporaly in time slots of 30 minutes each. This way make easier the Visit schedule. The time estimation of each appointment is determined by the bag size. In particular it could be:

- *Small*: it occupies 1 slot in the schedule, that is 30 minutes;
- *Medium*: it occupies 2 slots in the schedule, that is 60 minutes;
- *Large*: it occupies 3 slots in the schedule, that is 90 minutes;

In the following model, we describe the situation in which one reservation is booked and consequently inserted in the queue. (figure 4.5)

In particular we will list 2 queue:

- Q is the queue before the insertion.
- Q2 is the queue after the insertion.

Instead, in the figure 4.6 describes the situation in which one reservation is deleted from the queue. This means that the reservation will be removed by its user. We prefer not to update the reservation number, because it implies to duplicate all reservation in the queue behind the one that was cancelled. This could involve less readability.

```
open util / boolean

------SIGNATURE------
//abstract signature for a generic user. It will be extended
abstract sig User{
   age: Int,
   hasVisit: set Visit,
   hasReservation: set Reservation
}

sig SmartUser extends User
{
hasPosition: lone Position,
hasApp: Bool
}{
hasApp = True
}

sig MobileUser extends User
{
number:  phoneNum,
hasApp: Bool
}{
hasApp = False
}

one sig Date
{
day : String,
time : Time
}
{
day in ("Monday"+"Tuesday"+"Wednesday"+"Thursday"+"Friday"+"Saturday"+"Sunday
    ↪ ")
}

sig phoneNum{}
sig Position{}


sig Visit{
     hasSize: Size,
     startTime:  Time,
     endTime: Time,
        ID_code:  QRCode
}{
startTime.minutes in (0+30)
endTime.minutes in (0+30)
startTime.seconds = 0
endTime.seconds = 0
}

sig Reservation{
        hasSize: Size,
        ID_code:  QRCode,
        reservationTime : Time,
        numberInQueue: lone Int
}
//Size bag for a grocery shopping
sig Size{
        dimension : String
}{dimension in ("Small"+"Medium"+"Large")}


sig QRCode{
        isSubmitted: Bool,
        isValid: Bool
}
```

```
//List of the Users who are wiaitng to enter in the market
sig Queue{
listOfReser : set Reservation
}

//List of Users that has scheduled a Visit
one sig VisitSchedule{
listOfVisits : set Visit
}

one sig Market
{
userInShop : set User,
state: String
}
{
state in ("Opened"+"Closed")
}

sig Time
{
hours: Int ,
minutes : Int,
seconds: Int
}
{
hours > 0 and hours ≤24
minutes ≥0 and minutes < 60
seconds ≥0 and seconds < 60
}



----PREDICATE----
pred reservePrec [ t1 : Time , t2 : Time]
{
t1.hours < t2.hours or ( t1.hours = t2.hours and t1.minutes < t2.minutes) or
    ↪  ( t1.hours = t2.hours and t1.minutes = t2.minutes and t1.seconds ≤
    ↪ t2.seconds)
}

pred showReservation{
Market.state = "Opened"
#SmartUser = 2
#MobileUser = 1
#Visit =0
#Reservation = 3
#Queue = 1
#Queue.listOfReser = 2
}

pred showVisit{
#Market.userInShop > 0
Market.state = "Opened"
#SmartUser = 1
#MobileUser = 1
#Visit = 2
#VisitSchedule.listOfVisits = 1
#Reservation=0
#Queue = 0
}

pred showMarketOpened{
Market.state = "Opened"
#SmartUser = 2
#MobileUser = 2
#Visit >0
```

```
#Reservation> 2
#Queue = 1
#Queue.listOfReser >1
}



pred showMarketClosed{
Market.state = "Closed"

}


pred addInQueue[q, q2 :Queue, r: Reservation]
{
q2.listOfReser = q.listOfReser + r
#q2.listOfReser = add[#q.listOfReser,1]
}

pred showAdd[q, q2:Queue, r: Reservation]
{
#Queue = 2
 addInQueue[q,q2, r]
#Reservation > 1
#q.listOfReser > 1
}

pred delInQueue[q, q2 :Queue, r: Reservation, u: SmartUser, u2 : SmartUser]
{
r in u.hasReservation and r in q.listOfReser
u2.hasReservation = u.hasReservation - r
u2.hasVisit = u.hasVisit
u2.age = u.age
u2.hasPosition = u.hasPosition
q2.listOfReser = q.listOfReser - r
#q2.listOfReser = sub[#q.listOfReser,1]
}

pred showDel[q, q2:Queue, r: Reservation, u: SmartUser, u2 : SmartUser]
{
#Queue = 2
 delInQueue[q,q2, r,u,u2]
#Reservation > 1
#q.listOfReser > 1
}


----FACT----

// The Reservation can be booked only when the shop is opened
fact onlyInOpenTime
{
all r: Reservation | r.reservationTime.hours > 8 and r.reservationTime.hours
    ↪ < 20
all v: Visit | v.startTime.hours > 8 and v.endTime.hours < 20 and reservePrec
    ↪ [v.startTime,v.endTime]
}


// A Visit is accepted by the reader if and only if
fact visitAccepted
{
all v: Visit |   (v.ID_code.isSubmitted = True and  v.ID_code.isValid = True)
    ↪   <=> (reservePrec[v.startTime,Date.time] and  reservePrec[Date.time,v
    ↪ .endTime])
}
```

```
// Condition for a Visit finished
fact visitFinish
{
all v: Visit |   (v.ID_code.isSubmitted = True and  v.ID_code.isValid = False
    ↪ )  <=> reservePrec[v.endTime,Date.time]
}

// Condition for a Visit not started yet
fact visitEarly
{
all v: Visit |   (v.ID_code.isSubmitted = False and  v.ID_code.isValid = True
    ↪ )  <=> reservePrec[Date.time,v.startTime]
}


// The reservation time cannot exceed data time
fact reservationNotInFuture
{
all r : Reservation | reservePrec[r.reservationTime,Date.time]
}

//  Every User must have at most 1 Reservation active
fact sameMomentReserv
{
no r1 : Reservation, r2 : Reservation | r1.reservationTime = r2.
    ↪ reservationTime and r1 not = r2
}

//Function that returns the number of user in queue before r1 (r1 included)
fun numberQueue [r1: Reservation] : set Reservation {
{  r: Reservation | r.ID_code.isSubmitted = False and r.ID_code.isValid =
    ↪ True and  reservePrec[r.reservationTime, r1.reservationTime] }

}

//Assign the queue's cardinality to itself
fact cardinalityQueue {
all r: Queue.listOfReser |  r.numberInQueue = #numberQueue[r]
}


//Assign none if a Reservation is not in queue
fact reservationQueue
{
all r: ( Reservation - Queue.listOfReser ) | r.numberInQueue = none
}


//For each Visit assignes time slot based on the bag size
fact visitSlot
{
all v: Visit | v.hasSize.dimension = "Small" <=>  ((  minus[v.endTime.hours,v.
    ↪ startTime.hours]= 1 and  (v.startTime.minutes - v.endTime.minutes) =
    ↪ 30 ) or (( v.endTime.hours = v.startTime.hours) and minus[v.endTime.
    ↪ minutes, v.startTime.minutes] = 30 ))
all v1: Visit | v1.hasSize.dimension = "Medium" <=>  (( minus[v1.endTime.hours
    ↪ ,v1.startTime.hours] = 1 ) and  v1.startTime.minutes = v1.endTime.
    ↪ minutes )
all v2: Visit | v2.hasSize.dimension = "Large" implies  (( minus[v2.endTime.
    ↪ hours, v2.startTime.hours] = 2 and  minus[v2.startTime.minutes, v2.
    ↪ endTime.minutes] = 30 ) or (minus[ v2.endTime.hours ,v2.startTime.
    ↪ hours] = 1 and minus[v2.endTime.minutes ,v2.startTime.minutes] = 30 )
    ↪ )
}
```

```
// When the market is closed the number of people in the market is equal to 0
fact marketClose
{
Market.state = "Closed" implies ( #User = 0 )
}

// For each User only one QRCode is valid and submitted at each time
fact sserOneActive{
all u : User | lone h : (u.hasVisit.ID_code + u.hasReservation.ID_code ) | h.
    ↪ isSubmitted = True and h.isValid = True
}

// Condition for a User to be in the Market
fact userInShop{
all u :  Market.userInShop  | one q : QRCode | q in ( u.hasVisit.ID_code + u.
    ↪ hasReservation.ID_code) and q.isSubmitted = True and q.isValid = True
}

// Condition for a User to be in the Market
fact userInShop2
{
no u: (User - Market.userInShop) | one q : QRCode |   q in ( u.hasVisit.
    ↪ ID_code + u.hasReservation.ID_code) and q.isSubmitted = True and q.
    ↪ isValid = True

}

// It doesn't exist a QrCode not valid and not submitted
fact notExistQR_codeMalevolus{
no q : QRCode | q.isSubmitted = False and q.isValid = False
}

//Every position is associated to a SmartUser
fact smartUserCanOwnPosition
{
all p : Position | one su : SmartUser |  p in su.hasPosition
}

//The market is open from Monday to Frida and from 8 A.M to 8 P.M
fact openMarket
{
Market.state = "Opened" <=> (Date.day in ("Monday"+"Tuesday"+"Wednesday"+"
    ↪ Thursday"+"Friday") and Date.time.hours > 8 and Date.time.hours < 20
    ↪ )
}

//Otherwise che Market is close
fact closeMarket
{
not Market.state ="Opened" implies Market.state ="Closed"
}

//Telephone number unique for every user
fact uniqNumber
{
all pn : phoneNum | one mb : MobileUser |  pn in mb.number
}

//Age constraint
fact noUnderAge
{
all u : User | u.age > 18 and u.age < 100
}

// Each Visit belongs to one User
fact visitBelongUser
{
all v : Visit | one u1 : User | v in u1.hasVisit
```

```
}

// Each User must have at most one Visit active
fact only1VisitActive
{
all u : User |  lone v1 : Visit |  v1 in u.hasVisit and v1.ID_code.isValid =
    ↪ True
}

// Condition for a Visit to be scheduled
fact visitInSchedule
{
all v : Visit | (v.ID_code.isValid = True and v.ID_code.isSubmitted= False) ≤
    ↪ > ( v in VisitSchedule.listOfVisits)
}

// Each Reservation belongs to one User
fact reservBelongUser
{
all r : Reservation | one u : User | r in u.hasReservation
}

// Each User must have at most one Reservation active
fact only1ReservationActive
{
all u : User | lone r : Reservation | r in u.hasReservation and r.ID_code.
    ↪ isValid = True
}

// Condition for a Rservation to be insterted in the Queue
fact reservationQueue
{
all r : Reservation |  ( r.ID_code.isSubmitted = False and r.ID_code.isValid
    ↪ = True ) <=> ( r in Queue.listOfReser )
}

// For each booking the QRCode is unique
fact oneQRforVisit {
all q : QRCode | (q in Visit.ID_code or q in Reservation.ID_code)
no v1 : Visit, v2 : Visit | v1.ID_code = v2.ID_code and v1 not = v2
no r1 : Reservation, r2 : Reservation | r1.ID_code = r2.ID_code and r1 not =
    ↪ r2
no r : Reservation, v: Visit | v.ID_code = r.ID_code
}
-------------------------------------------------------------

--<LET>

//Cardinality of QRCode submitted and valid
let many [QRCode] = {
 q: QRCode | q.isValid = True and q.isSubmitted=True
}

--</LET>
-------------------------------------------------------------


--<ASSERT>
//Check that user in shop are equal to person that have a QRCode valid and
    ↪ submitted
assert totPerson{
#Market.userInShop = #many[QRCode]
}

//Check that if the market is closed implies that there are no users with
    ↪ either Reservation or Visit submitted and valid
assert noBooking{
Market.state = "Closed" implies #{Visit + Reservation} = 0
```

```
}



--</ASSERT>--
/*
        check totPerson
        check noBooking
*/

run showMarketOpened  for 8 but  8 Int
/*
        run showMarketClosed for  8 Int
        run showReservation for 8 but  8 Int
        run showVisit for 6 Int
        run showAdd for 6 Int
        run showDel for 6 Int
*/
```

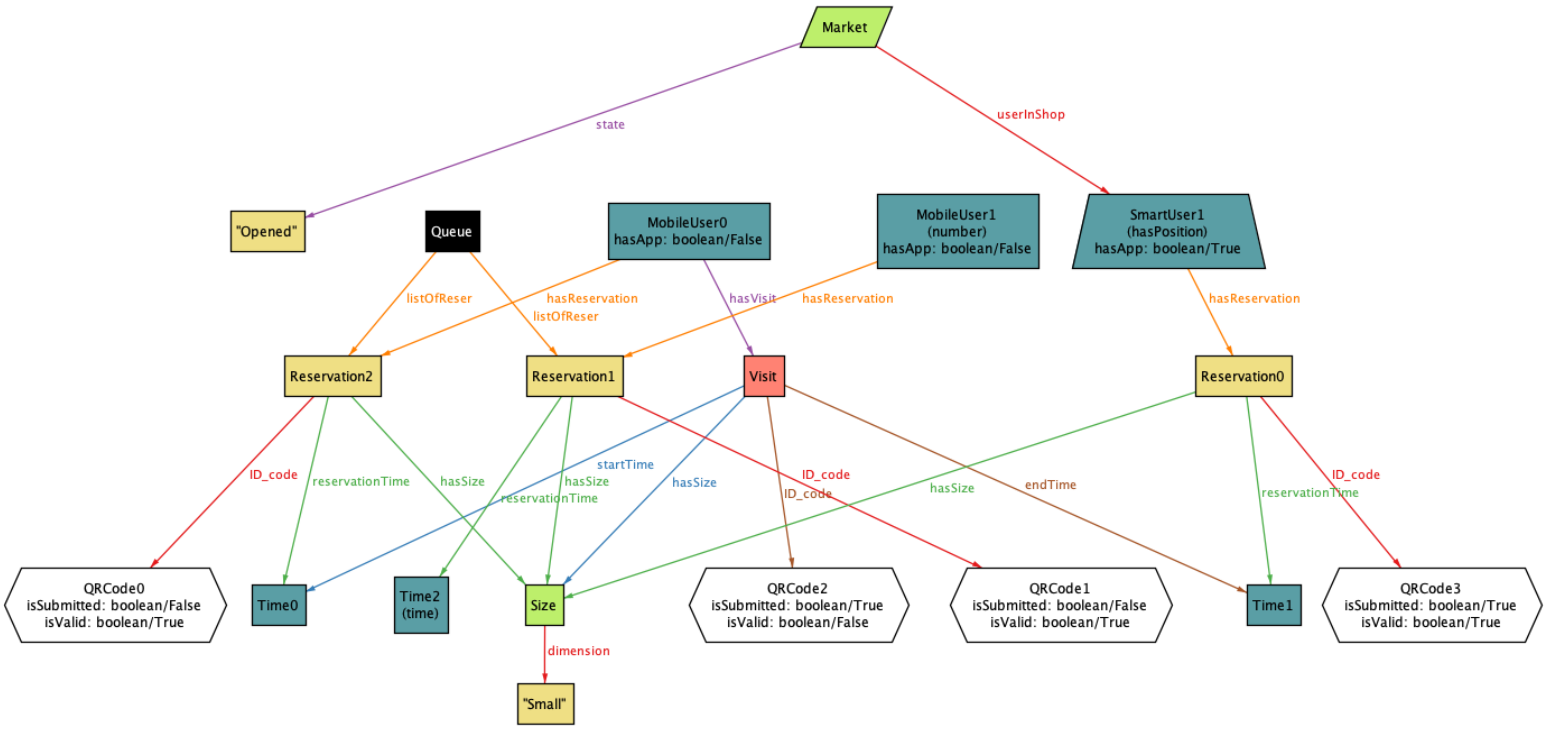


Figure 4.1: Rapresentation of the market opened in a generic day.

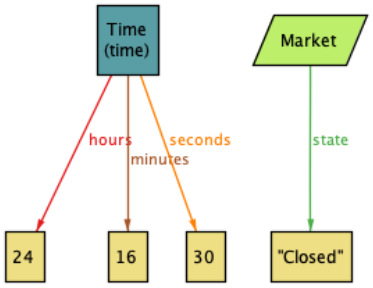


Figure 4.2: The market is closed. In this case, there aren't any users in the market.

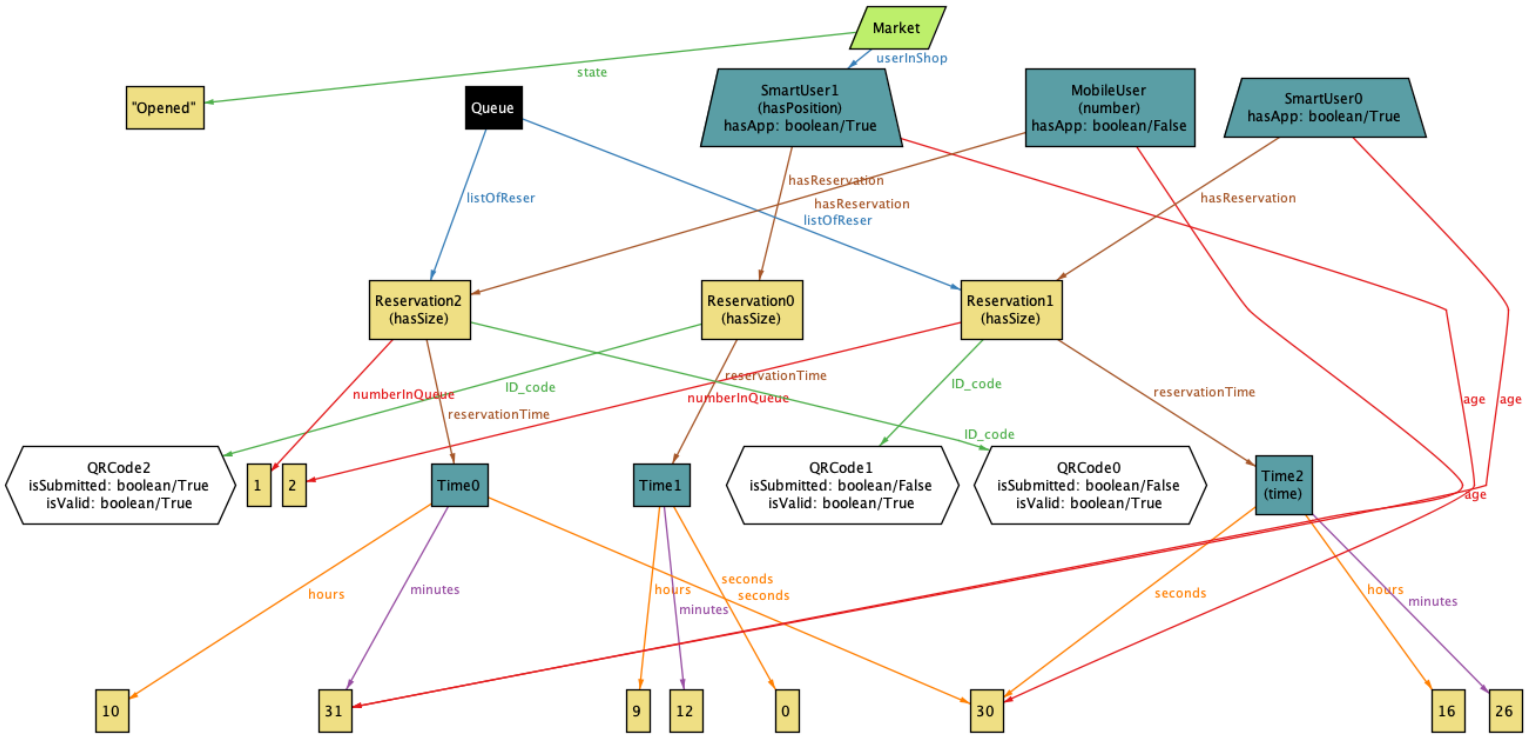


Figure 4.3: Rapresentation of the market opened in a generic day. This is the result of the showReservation predicate that focuses on Reservations.

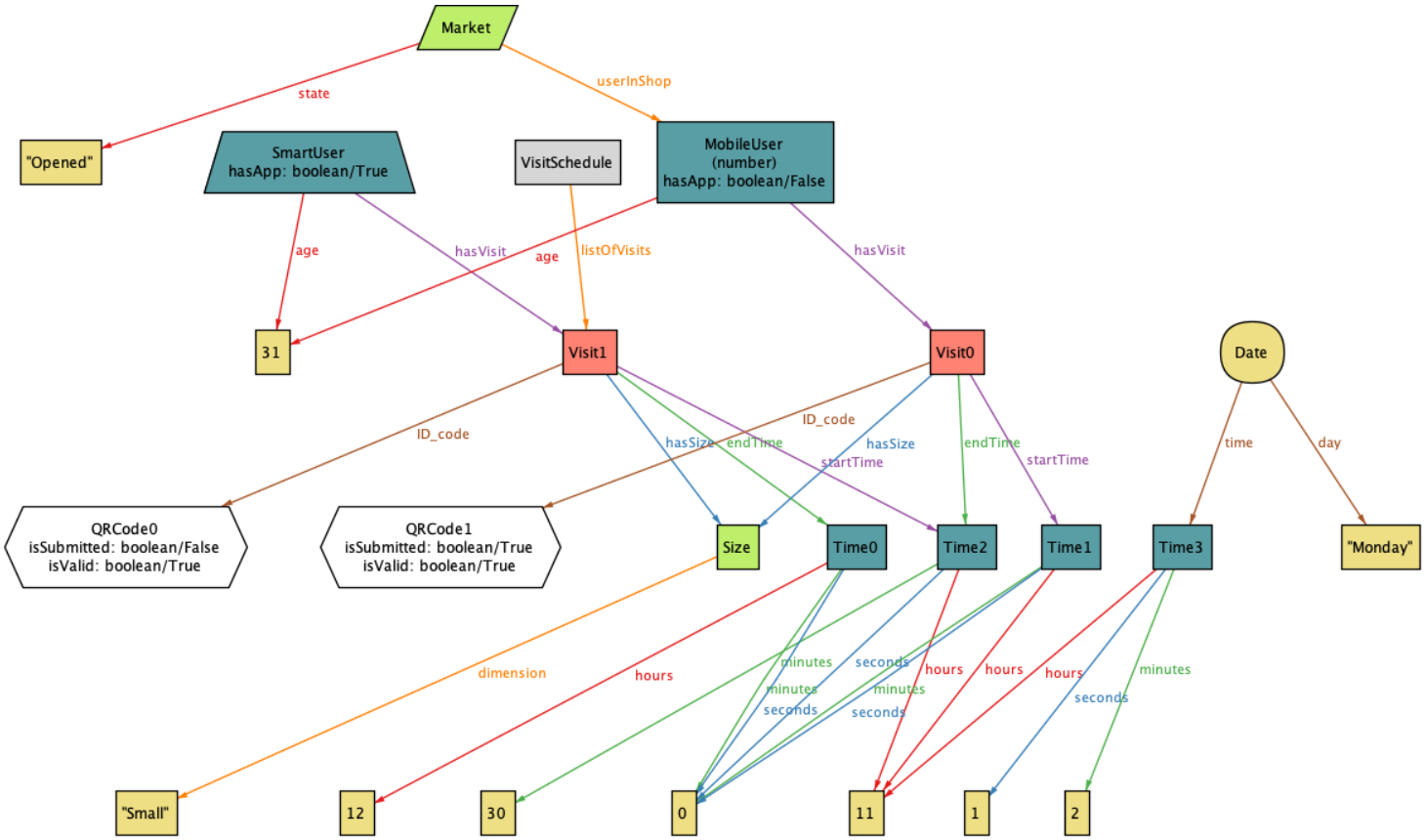


Figure 4.4: Rapresentation of the market opened in a generic day. This is the result of the showVisit predicate that focuses on Visits.

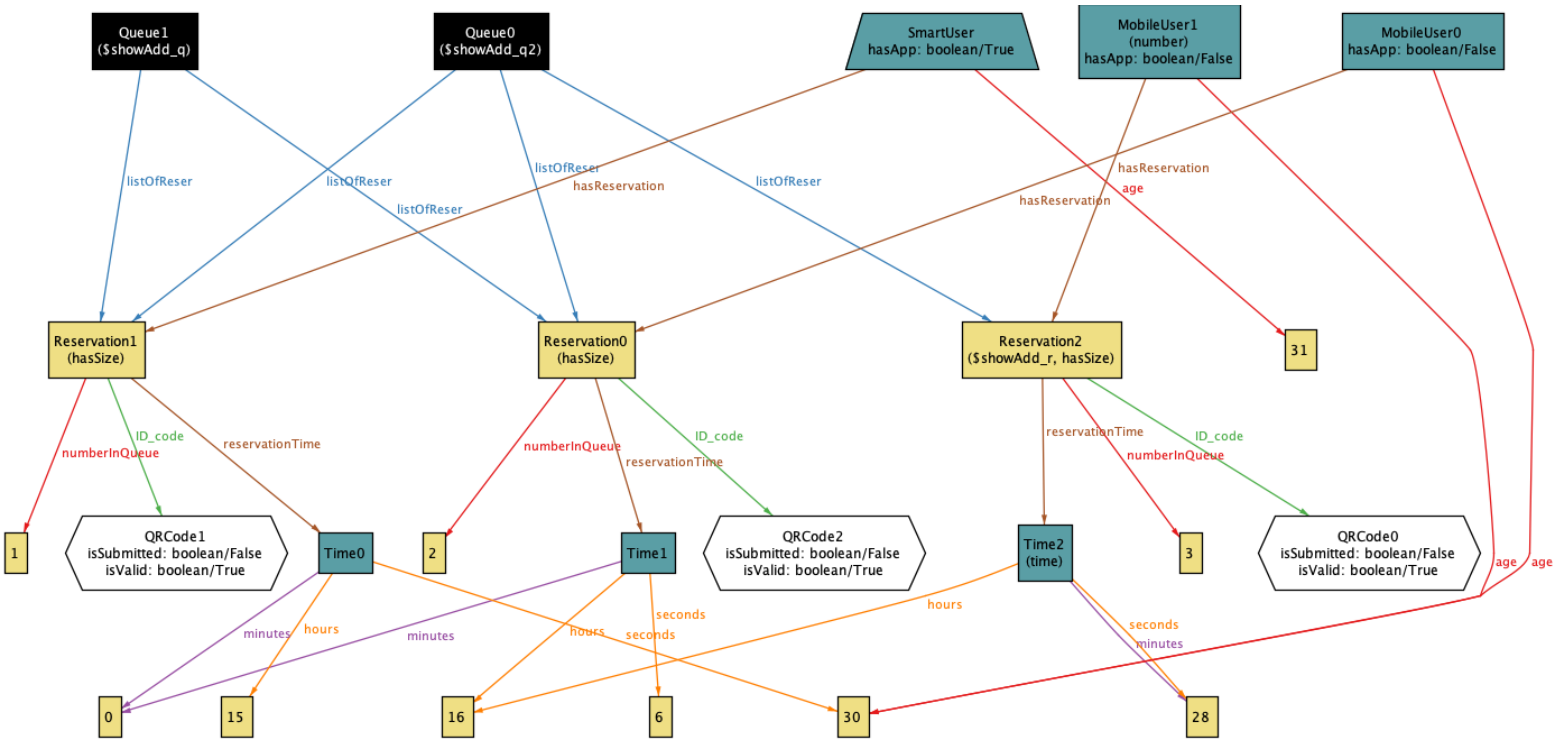


Figure 4.5: Queue insert reservation

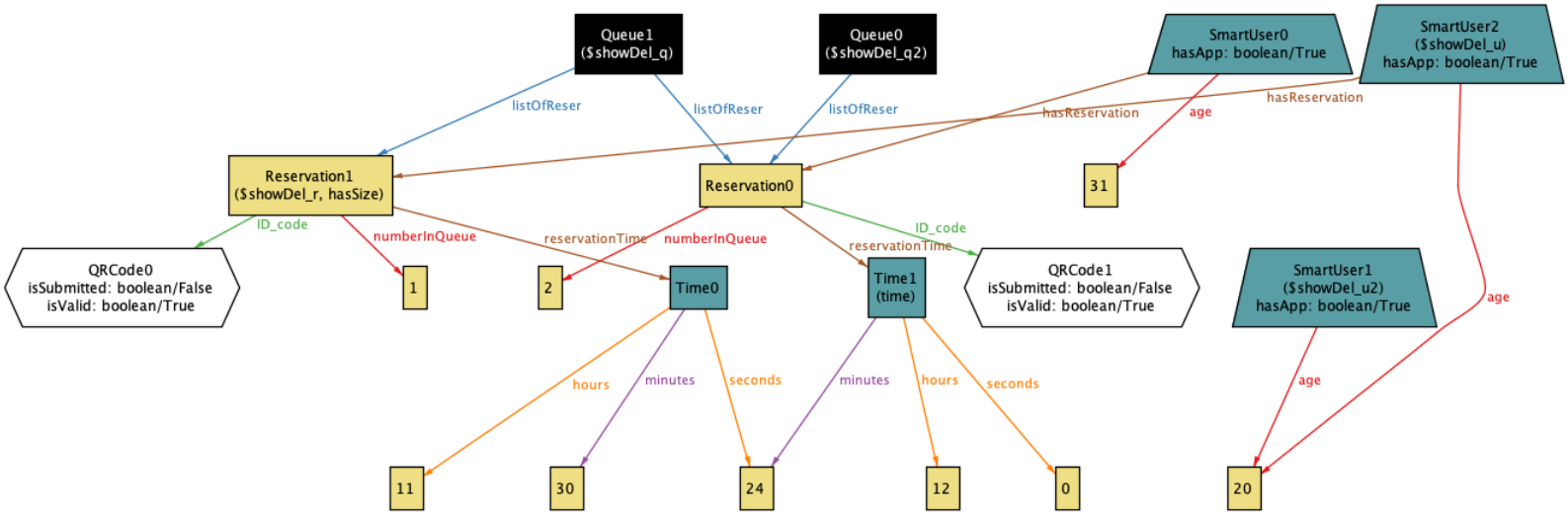


Figure 4.6: Queue remove reservation

4.2 Alloy Results

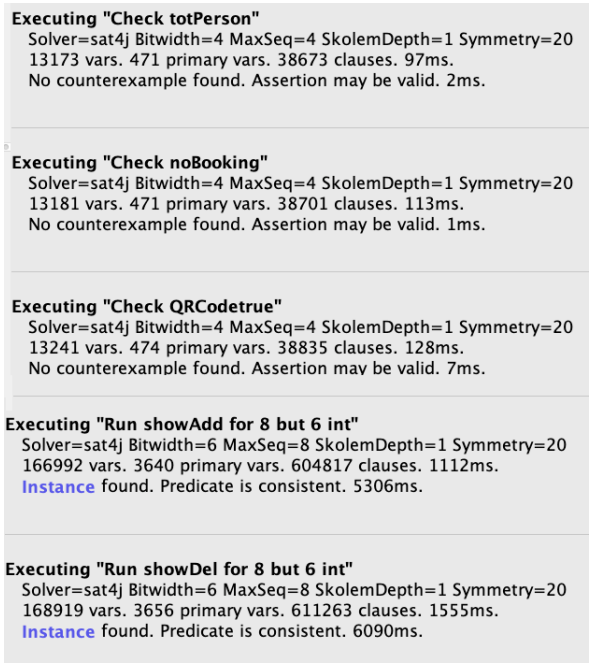


Figure 4.7: Results of the predicates

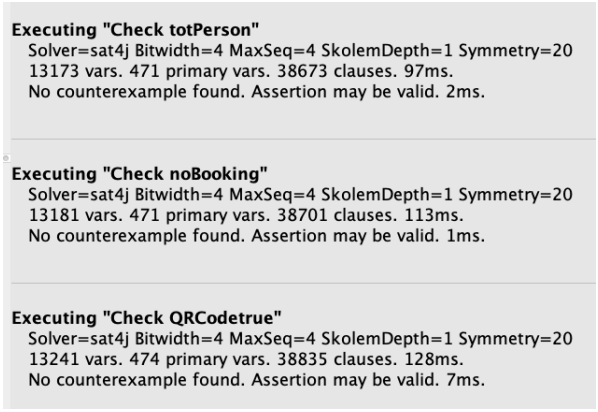


Figure 4.8: Results of the assers

Chapter 5

Effort Spent

In the following tables it's summerized the effort time spent from us.

Chapter/Task	**Hours**
Git	4
Introduction	8
Overall Description	11
Specific Requirements	12
Formal Analysis with Alloy	4
Review	5
	Total
	42

Table 5.1: Matteo Bresciani's effort

Chapter/Task	**Hours**
Introduction	6
Overall Description	8
Specific Requirements	14
Formal Analysis with Alloy	12
Review	3
	Total
	43

Table 5.2: Stafano Banfi's effort

Chapter 6

References

6.1 Software used

- **LaTeX**: used to write and to build the document [https://www.draw.io/];
- **Draw**: used to create class, sequence and case diagram [https://www.draw.io/];
- **GitHub**: used to store and manage project repository [https://github.com/];
- **GitHub Desktop**: is the official GitHub application which allows us to contribute to the project repository in an easy way [https://desktop.github.com];
- **Figma**: used to design mockups provided in this document [https://www.draw.io/];
- **Alloy Tool**: used to describe the system model in Alloy language [https://www.draw.io/];

6.2 Bibliography

- Slides of *Software Engineering 2* course [https://beep.metid.polimi.it/];
- *R&DD Assignment A.Y. 2020-2021* [https://beep.metid.polimi.it/];
- *GDPR* [https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679];