POLITECNICO
MILANO 1863

Software Engineering II project

CLup

Requirements Analysis and Specifications Document

Authors:

Alberto MAGGIOLI
Lorenzo PORETTI

Professor:

Matteo Giovanni Rossi

Version 3.0

Contents

1.Introduction

Because of the pandemic situation due to Covid-19 there is a need to make sure that people can be spaced out and ensure their safety. For this reason, supermarkets also need to ensure social distancing.

1.A Purpose

The aim of this document is to point out all the requirements and specifications of the software descripted below. The document contains the analysis through different models and diagrams that highlight functionalities of the final software. The principal aim of this project is to point out all the requirements and goals of the shareholders interested on this project; it can be used also by developers in order to have a high level view on the entire project. The structure of this document accords to the IEEE 830 standard.

1.A.1 Goals

G1	Avoid high number of customers in the supermarket
G2	Avoid the creation of aggregation of customers in front of the supermarket
G3	Let people saving time
G4	Infer the duration time of customers visit
G5	Monitor the entrances of customers
G6	Make sure the order of the waiting queue is preserved

1.B Scope

The aim of this project is to design an application which help store managers to regulate the influx of people in the building monitoring entrances to comply with the coronavirus safety regulations.
Moreover the application has the goal to prevent people from waiting outside the store safeguarding clients health and not wasting time.

The application has several features:

- The user can take a ticket with which he is placed in a virtual queue. Right before his turn he is notified by the application so that he can go to the store on time.
- The user can book a visit to the supermarket at the time and date he prefers, according to the remaining availabilities.
- The application suggests to the user in the booking process some different times with a smaller number of people inside the store, other supermarkets of the same chain or other chains.
- The user in the booking process can also insert in detail the products he/she intends to buy so that the application can control and distribute in the best way the users inside the store.
- The application is able to predict the duration of the visit by inferring on the duration of a particular customer's past visits. Alternatively, the duration of the visit can be specified by the user during booking.

The application is not implied in handling supermarket tasks, we assume that another application manages the database updating all the information about the supermarket.

The application should be very simple to use, as the range of users include all demographics and it should be real time application to let user to allow clients to be informed.

1.B.1 World Phenomena

WP1	Customers go to the supermarket
WP2	Customers wait for their turn
WP3	Supermarket workers enter the store
WP4	Customers pay before going out
WP5	The user arrives too early to the supermarket

1.B.2 Shared Phenomena

SP1	Customers take their tickets on the spot
SP2	Customers reserve tickets from the application
SP3	Customers book a visit
SP4	Approximate time of the visit of a customer
SP5	Customers point out the categories of items needed
SP6	Customers exit the supermarket
SP7	Customers scan the QR code before entering the supermarket
SP8	The user is notified in advance when it is his turn
SP9	The user arrives too late to the supermarket
SP10	The application sends a notification of available slots

1.C Definitions, Acronyms, Abbreviations

1.C.1 Definitions

Visit	A customer goes to the supermarket
Reservation	an arrangement that you make to have a visit in a supermarket.
Slot	Period of time in which the calendar is divided
QR code	It is a type of matrix barcode. A barcode is a machine-readable optical label that contains information about the item to which it is attached.
QR scanner	An object in charge to read QR codes

1.C.2 Acronyms

RASD	Requirement Analysis and Specification Document
QR	Quick response
DBMS	Database Management System
API	Application Programming Interface
GPS	Global Positioning System
OS	Operating System
GDPR	General Data Protection Regulation

1.C.3 Abbreviations

WPn	World Phenomenon number n
SPn	Shared Phenomenon number n
Gn	Goal number n
Dn	Domain assumption number n
Rn	Requirement number n

1.D Revision history

Version	Date	Modifications
Version 1.0	16/12/2020	First Version
Version 2.0	03/01/2021	We added two requirements, in particular login and registration requirements. We have introduced also an external API used for credentials recovery
Version 3.0	15/01/2021	Added an introduction to the document. Rewrote Goal six in a more comprehensible way. Removed Word phenomena 2,7,8 of the previous version because they are considered static. Modified scenario 7 in a clearer way. Used aggregation between ticket and user in the UML diagram. Used composition between reservation and user in the UML diagram. Added Personas section in which we defined Personas and we represent their mapping with Goals and Scenarios. Removed three domain assumptions, in particular D1 D11 and D12 of the previous document Removed the condition that mobiles need GPS in Hardware Interfaces. Modified some parameters in performance requirements. Added GPDR reference to Security section. Added version of Android in Portability.

1.E Reference Documents

- Specification Document: “R&DD Assignment A.Y. 2020-2021”
- Slides of the lectures
- Book: “Software engineering principles and practice”, Hans Van Vliet

1.F Document Structure

The document is divided in six main sections according to IEEE 830 standard:

1. **Introduction**: This section aims to describe with an high level approach the goals of the CLup system pointing out world and shared phenomena which could play an important role in the development of the application.
   It's also described some of the definitions and acronyms used in this document.

2. **Overall description**: In the very first part there are some scenarios which describe some possible situations which could arise. This part is followed by class diagrams which represent, in a graphical way, the main classes of the system. Each class is then described in a detailed way. State diagrams are then used to described how some critical components of the system evolve. Production functions are then described to highlight the most important functionalities that the system must guarantee.
   In the last part of the section domain assumptions are presented. Domain assumptions have an important role as they are facts or statements taken for granted.

3. **Specific requirements**: This section presents the interface requirement as user, hardware, software and communication interfaces. The main aim of this section is to presents functional requirements which allows to achieve the goals presented in the first section, for this reason a mapping between goals, requirements and domain assumption is presented. Functional requirements are followed by use cases and corresponding sequence diagrams used to point out specific cases in which the application can be used.
   In the last part of this section are listed some of the non-functional requirements as performance requirements, design constraints and software system attributes.

4. **Formal Analysis using Alloy**: In this section is presented a model of the system and it is described using Alloy in a declarative way. This section points out the most critical aspects of the system.
5. **Effort spent**: This section is used to keep track of the hours spent by each team member to complete each section.

6. **References**: This section Includes all the references used to define the document

2. Overall Description

2.A Product Perspective

2.A.1 Scenarios

The following scenarios clarify through practical examples the impact on the user of the main functionalities of CLup:

1. Someone is late

Marco uses the application to get in line and he is given a ticket: currently he has 34 tickets in front of him. Thinking that the wait could be long, Marco decides to go and do physical activity outside.
Unfortunately 10 of 34 tickets do not show up while the remaining ones complete the shopping very quickly. Once back at home Marco realizes that he has three notifications from the application: the first one warns him that it is about to be his turn, the second one that it is his turn and the third one warns him that, not having scanned the entry QR within 10 minutes from taking his turn his ticket, it has expired. Marco decides to take another ticket and pay more attention to the phone.

2. Multiple reservations

Roberto, consulting his agenda, realizes that in the following week he has Wednesday afternoon free, so he decides to book a slot to go shopping in his favourite supermarket.
The following Wednesday, not remembering to have booked a slot for 15:30 and not having seen the notification that reminds him, Roberto at 14:40 decides to take a ticket to go shopping. The application prevents him from generating a ticket by reminding him of the reservation already present for the same day, at a very similar time and in the same supermarket.
Roberto, happy not to have to pay attention to the notifications, goes to the supermarket when it is his turn.

3. Ticket on the spot

Bruno is 91 years old and does not own a smartphone, so he cannot take online tickets or make reservations. Unconscious of the new system to regulate the income at the supermarket near home, he goes as usual to do his weekly shopping. Once he arrives at the supermarket he is informed about the new system; he then takes a ticket from the totem at the entrance and waits until his number is called. Once arrived at the entrance he scans the first QR on the ticket and completes his shopping, once the shopping is finished he is reminded to scan the second QR on the ticket at the scanner located after the checkouts so that the line can flow.

4. Users do a reservation

Giacomo lives in Cremona but works in a consulting firm in Milan, between office hours and travel to work he is very busy and has to plan the best use of the day. Since Giacomo got to know CLup, he immediately started to use it; in fact it is very important for him not to have to queue outside the supermarket which would make him waste a lot of time. Giacomo, according to his weekly schedule, identifies the right moment. He books, using the application, a slot and finally he goes to the supermarket on the selected day and time, avoiding to waste time in the queue.

5. CLup does suggestions to balance the number of people on the slots

Alessia is very organized and regularly books her shopping in advance with the CLup app. Unfortunately, due to family commitments in the last week she remembered late to book her slot at the supermarket next door. Alessia tries to book through the app; although there is still a place for the selected slot, she is suggested to choose a slot the next morning or a slot at the same time at the supermarket of the same chain 5 kilometres away. Alessia decides to book at the supermarket a little further away so as not to postpone the shopping.

6. CLup suggests another slot to a user based on his usual visits

Maria is usually very meticulous about shopping and she likes to spend some time inside the supermarket to get inspired, based on the products she sees, on the weekly recipes. Usually Maria goes to the supermarket once a week or even once every two weeks in order to reduce the number of visits to the supermarket as much as possible. Unfortunately, on her usual Thursday shopping trip, Maria forgot to pick up apples and butter to make her special apple pie, so she decides to book a visit to the supermarket through CLup. The application, knowing the average duration of her long visits, sets a visit duration of 1 hour and a half; obviously, this time such a long visit is not necessary, so Maria indicates in the reservation the minimum possible duration (half an hour).

7. CLup modifies slot availabilities based on the indicated shopping list

Tommaso is used to book his weekly shopping through CLup choosing his trusted supermarket. During the Christmas vacations Tommaso realizes that the shopping he did the day before is not complete and some essential ingredients for the Christmas Eve dinner are missing. Opening the application Tommaso realizes that until December 27th there is not even a place in any nearby supermarket, disconsolate he is going to resign and give up the traditional Christmas dinner tortelli when he remembers that when booking you can indicate specific items that you intend to buy. Since it is only missing flour and an egg and those products are on the same shelf, he is notified that a slot is available for the 23rd afternoon, Tommaso is very happy to book.

8. User lost his ticket during shopping

Christian has recently moved from Denmark and he is very distracted and disorderly and he often misses something. Like every Friday morning, Christian goes to the big shopping mall next to the station to do his weekend shopping. When he tries to get into the supermarket, an employee explains to him that since Tuesday there is a new system to regulate the influx into the store and advises him to take a ticket at the totem at the entrance. Christian waits until his turn is called, once he has scanned the QR of his ticket valid for the entrance, he starts to do his shopping. When Christian has finished paying, he realizes that he has lost the ticket that also contained the QR to be scanned at the exit; submitted the problem to an employee is provided with an ad hoc QR for the exit so that the queue can flow once it exits the supermarket.

9. User take a ticket to save time

Ian is very busy and does not want to waste time queuing in front of the supermarket, so when he learns about CLup, he is enthusiastic about the innovative system with which he can save time. When he has a free morning, he generates a ticket to go shopping in the morning. Once he knows how many people are in front of him and the approximate time his number will be called, Ian can spend the time he would have lost queuing at the supermarket keeping fit with exercise. As soon as he hears a notification coming, Ian checks his phone so he can get to the supermarket in time.

2.A.2 Class diagrams

In the following image it is represented the class diagram, which gives an high level view of the main components of the project. The main objects in the UML class diagram are:

- **User**
  This class manages the user's profile. User are the clients of the application who can take a ticket or do a reservation. This class also has information about all the reservations that the user booked.

- **Ticket**
  This class contain all the information about the ticket taken by the user. Each class has a QR attribute which let user go in the supermarket. The ticket has a direct link with the queue to which it belongs. There are two type of tickets:
  - **Real ticket**
    Represent the ticket taken on the spot. The exitQr attribute is unique for all the tickets and let the user go out from the supermarket.
  - **Virtual ticket**
    Represent the ticket taken from the application.

- **Reservation**
  This class contains all the information about the reservation booked by the user. One of the main functionalities is to create a ticket associated to the reservation. Information specified are: duration of the visit, the date of the reservation and the sectors in which the customer will shop.

- **Slot**
  This class represents a slot on supermarket calendar, for each one is possible to reservemore visits. Once a reservation is booked, slots availability is updated. If the slot is free, the user can book a reservation.

- **Supermarket**
  This class contains information about the physical store like its position and opening hours. It contains the list of slots and its queue reference.

- **SupermarketChain**
  Here we can access to all the supermarket which belonging to a specific chain.

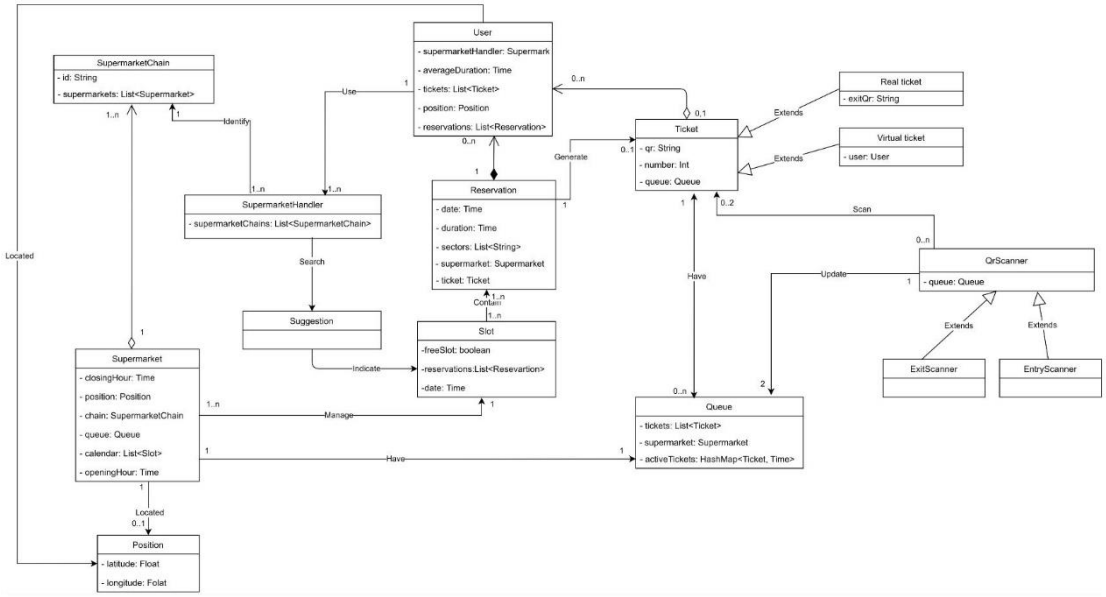
- **SupermarketHandler**

  This class is very important for system functionalities. Here we have a reference to all possible supermarket chains. This class is useful to make reservations, take tickets and visualize supermarket information. It also helps users by doing some suggestion of alternative slots.

- **Suggestion**
  The aim of this class is to identify some alternative slots to make suggestion regarding a specific request.

- **QrScanner**
  This class represent QR scanner into the supermarket, when a QR is scanned it checks its validity and it updates the queue.
    - **EntryScanner**
      This class represents the entry QR scanner, when a valid QR is scanned, the system updates the queue
    - **ExitScanner**
      This class represents the exit QR scanner, if the QR scanned correspond to a VirtualTicket it checks if there is a valid scanned entry for this QR code, then updates the queue.

- **Queue**
  This class represents supermarket virtual queue, it has information about all the tickets in the queue and all active tickets. When a client scans exit QR queue class manages the flow of the queue and notifies user's turn.



2.A.3 State Charts

This section points out the main functionalities which let components change their state and the most complex aspects described in the UML diagram.

We presents the following aspects as a state diagrams:

- Request for available slots
- Generate a ticket from the application
- QR entry scanner functionality

Request of available slots

The process starts when it receives a request from the user to ask for available slots in a specific supermarket and in a given hour.

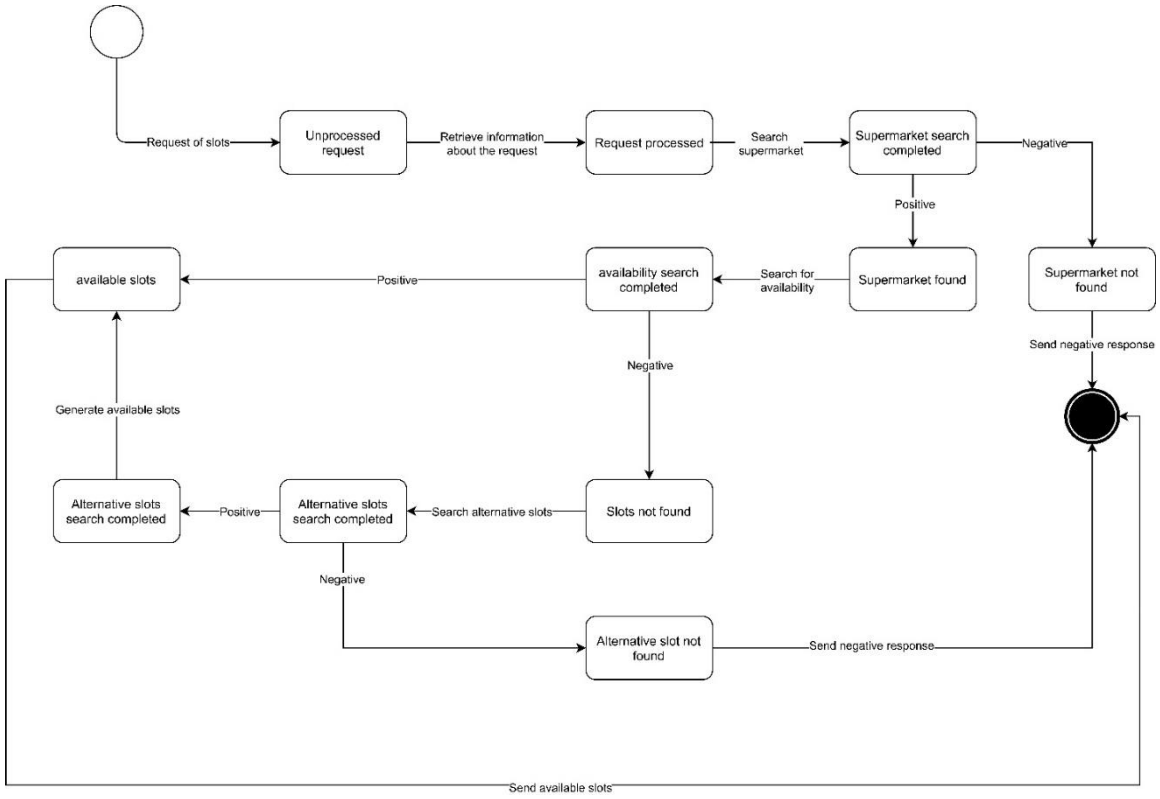
Once the request is analyzed by the system, it begins to look for the supermarket.

If the system doesn't find the supermarket , it sends a negative response to the user and the process ends.

When the system finds the supermarket it search for available slots. if the query is successful, slots are send to the user and the process terminates successfully.

If the supermarket doesn't find free slots in the supermarket it suggests other alternative slots in other supermarket or in another hours.

If the supermarket doesn't find neither alternative slots, a negative response is sent to the user and the process ends, otherwise alternative slots are sent to the user.

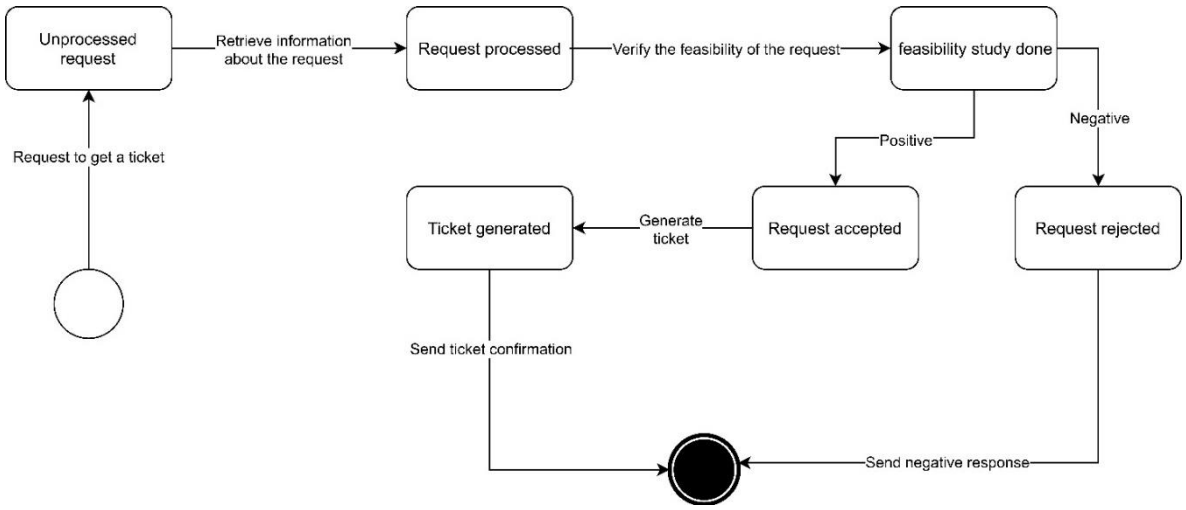


Generate a ticket from the application

The process starts when a user does a request to get a ticket. Once the request is processed a feasibility study is done to prove certain things necessary to generate a new ticket, for example the supermarket must exists in the system and it has to be currently opened.

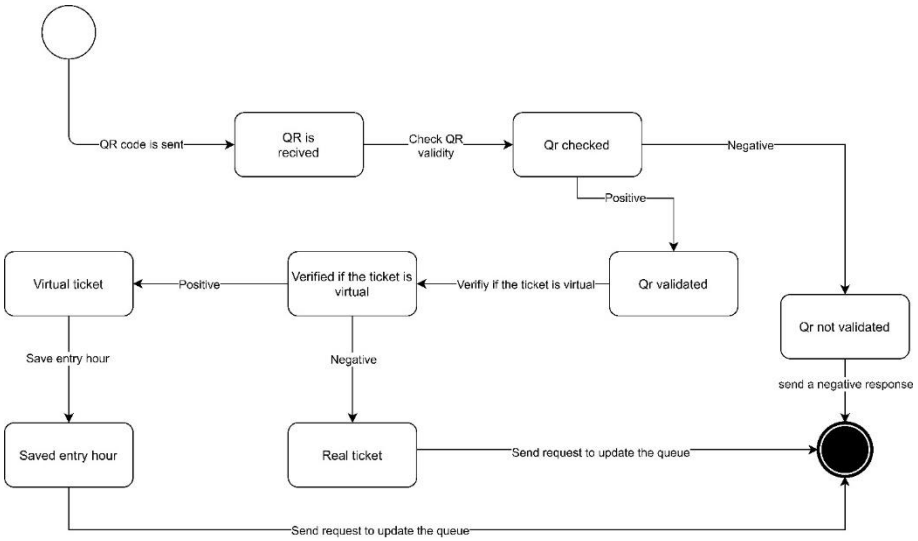
When the request is accepted, the system generates a ticket with a QR code, a number and a link to the supermarket. The System sends a confirmation to the user and the process ends.

In case the request is not accepted, user is notified with a negative response.



QR entry scanner functionality scan

The process starts when a QR scanned is sent to the system. First of all the system checked if the QR is the one expected. If the validation gives a negative result, the process terminates and a negative response is sent to the supermarket. If the QR code is validated, the system checks if it is a real ticket or a virtual one. In the first case a request is sent to update the queue, in the second case an intermediate step is taken and the entry hour is saved (entry hour is useful to the application to estimate duration of future visits); both cases terminates successfully with the request to update the queue.



2.B Product functions

CLup is an application that allows people to eliminate queues in front of stores to improve their safety during the COVD-19 outbreak. On the other hand, it also allows customers to save time by avoiding long queues. We will describe in detail below all the features of the application, of course it is assumed that the user has already logged into the system.

Take a ticket function

This function is the main one of the application. In order to take the ticket the user activates the GPS so that the duration of the journey to the supermarket can be calculated, if the user denies GPS permissions, the system supposes the duration of the journey as a constant . A user decides to go to the supermarket, first he selects the supermarket chain he prefers and then the exact supermarket where he wants to shop. Once the supermarket is selected, the application shows how many people are queuing at the moment and the estimated time of entry. When the user decides to take the ticket, it is generated and inserted into the queue. The user then waits to be notified of the proximity of his turn, the system calculates an estimate of the time taken by the user to reach the supermarket and sends the notification on time. Once at the store, the customer waits until his turn is called, then, before entering, he scans the QR code that was generated for his ticket. Once the purchases are completed, the user scans again the QR code of his ticket at the QR scanner located at the exit before leaving the supermarket. Thanks to the scanning at the exit, the queue can advance ensuring that the maximum number of customers inside the store is never exceeded.

A user without a smartphone can still shop at the supermarket, in fact at the store there is a totem that dispenses physical tickets for users without a smartphone. These tickets are placed in the queue on an equal footing with the tickets taken through the app, but unlike these, they have two different QR codes: one for entry and one for exit. While the one for the entrance has the same function as the digital one, the one for the exit is the same for all physical tickets: this trick avoids problems in case of loss of the ticket, in fact it allows the advancement of the queue by scanning a model of the code present at the exit.

Book a visit function

The user can also decide to book a visit to the supermarket. In this case he has to choose the supermarket where he wants to do the shopping, once selected them, he indicates the chosen date. If no bookable slots are available on the indicated date, the system automatically suggests another supermarket of the same chain or of another chain with some availability on the indicated day, if none is found, the result is notified. Once the date has been chosen, the user can specify which items he wants to buy and how long he thinks his visit will last; according to the preferences indicated, the system automatically updates the availability of the slots. When the user selects the slot he prefers, the system checks if the slot is particularly full compared to others on the same day, notifying the user of the result with the advice to choose another one. Once the time is chosen, the system generates a reservation for the user; on the day and time corresponding to the reservation, the user is notified of the generation of a ticket that will be used as in the "Take a ticket" feature.

Every time a registered user makes purchases using CLup, their average visit duration statistics are updated, which can be calculated by scanning the QR code on entry and exit. When a user with a valid visit statistic makes a reservation, the system automatically applies a filter to the availability corresponding to the average duration of his purchases. The user can still specify the duration of the visit he wants to book, in this case only the latter will be considered valid.

In addition to the functions "Take a ticket" and "Book a visit" the user can at any time view his valid tickets that have expired less than a day ago and his future bookings or those made up to 24 hours before.

2.C User characteristics

User characteristics are very different in terms of professional level, cultural level and age, although it is expected that there are few users under 20 and few over 70 who prefer take their ticket on the spot instead of using the app.

We have pointed out some different personas who have different roles in our application:

- **Supermarket manager**: He is interested in ensuring that all regulations are respected in in order to guarantee the safety of his customers, but at the same time to maximize sales and optimize the number of people in his supermarket.
- **Old style customer**: We identify with this persona all the people who don't want to use an application to take a ticket and prefer to take a ticket on the spot. We expect these people to have an average age over 60, but also these who do not often go to the supermarket and therefore do not feel the need to register for the CLup app.
- **Tech-friendly customer**: We identify with this persona all the people who will be interested in the application.

We decide to point out some Personas characteristics used in Scenario section.

Name	**Age**	**Studies**	**Job**	**Status**
Marco	35	Master	Doctor	Married
Roberto	22	Graduated	Student	Single
Bruno	91	High school	Retired	Married
Giacomo	40	Graduated	Consultant	Married with two children
Alessia	52	PhD	Professor	Married with one child
Maria	65	Eighth grade	Retired	Single
Tommaso	32	Graduated	Designer	Engaged
Christian	28	High school	Footballer	Married with one child
Ian	35	Graduated	CEO	Married

This table represents the mapping between Scenarios and Personas

Name	**Persona**
Marco	Tech-friendly customer
Roberto	Tech-friendly customer
Bruno	Old style customer
Giacomo	Tech-friendly customer
Alessia	Tech-friendly customer
Maria	Tech-friendly customer
Tommaso	Tech-friendly customer
Christian	Old style customer
Ian	Tech-friendly customer

This table represents the mapping between Goals and Personas

Goals	**Persona**
Avoid high number of customers in the supermarket	Supermarket manager
Avoid the creation of aggregation of customers in front of the supermarket	Supermarket manager
Let people saving time	Tech-friendly customer
Infer the duration time of customers visit	Supermarket manager
Monitor the entrances of customers	Supermarket manager
Make sure the order of the waiting queue is preserved	Tech-friendly customer Old style customer

2.D Assumptions, Dependencies and Constraints

2.D.1 Domain Assumptions

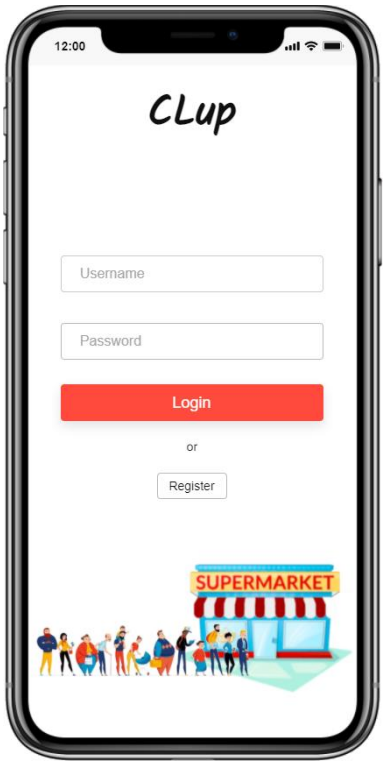
D2	Clients scan the QR code before entering the supermarket
D3	The supermarket must have internet connection
D4	Clients respect the tickets order
D5	Clients respect the zone that they select in the “book a visit” process
D6	Clients scan the QR code before exiting the supermarket
D7	Clients respect the duration of a visit as they indicate in the “book a visit” process
D8	The supermarket provides a QR scanner
D9	The tickets have a QR code
D10	The supermarket workers enter and exit without scanning the QR code
D13	All the supermarkets provide the opening hours
D14	All the supermarkets provide their location
D15	Customers respect the distance from each other’s
D16	The supermarket respects the opening hours
D17	The user provides his current position
D18	Each ticket corresponds to a single customer

3. Specific Requirements

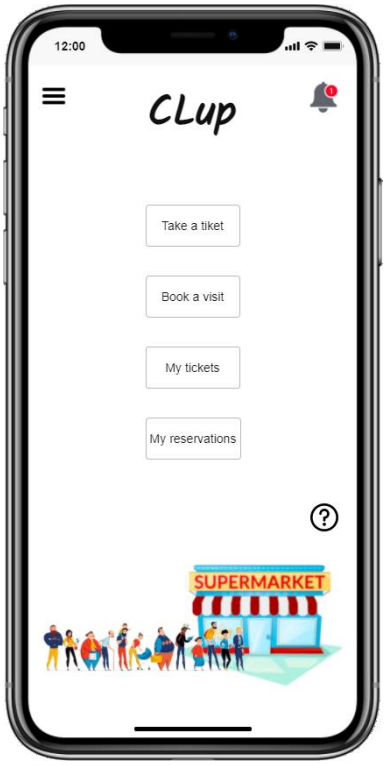
3.A External Interface Requirements

3.A.1 User Interfaces

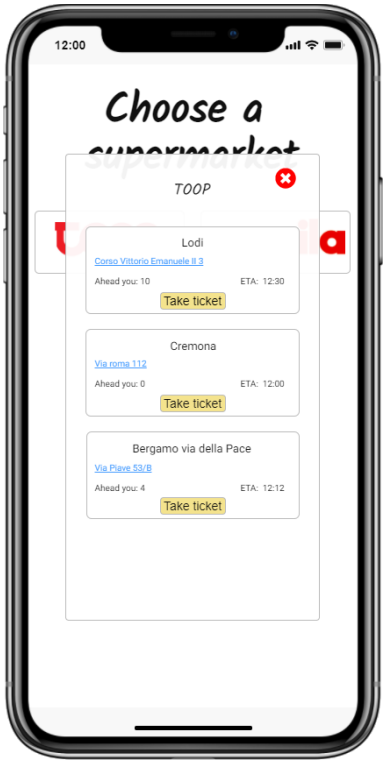
The following images are mockups to show how the application will look like.



1. Login Page



2. Home Page



3. Take a ticket

4. Do a reservation

3.A.2 Hardware Interfaces

The supermarket will need to have a totem that acts as a proxy to request a ticket and two QR scanners to be able to track entrances and exits.

3.A.3 Software Interfaces

External software interfaces concern the functionalities for which communication between CLup and the user is necessary. In particular are necessary:

- for the creation of tickets
- for booking a visit
- to check the status of a reservation or a ticket
- to check the availability of slots in a particular supermarket.

In addition, an interface to the external API will be needed to estimate the time needed for the user to reach the supermarket. An external API is also needed to handle recovery credentials. As for the internal software interfaces, it is necessary for the interaction with the DBMS in order to manage the data reading/writing.

3.A.4 Communication interfaces

The device connects to the CLup server via an internet connection

3.B Functional Requirements

This section aims to give a complete description of the functional requirements of our system by defining them together with the domain assumptions to achieve the identified goals.

3.B.1 List of requirements

R1	The application must consider the tickets taken on the spot
R2	The application creates a virtual queue for customers
R3	The application tracks the entrance of customers
R4	The application tracks the exit of customers
R5	User indicates the items that will be purchased
R6	User indicates the approximate duration of the visit
R7	The application must assign a ticket to each user asking for it
R8	The application let the user to select a date in a calendar to allow him to book a visit
R9	The application estimates the time needed to reach the supermarket
R10	The application tracks the number of customers who booked a slot
R11	The application knows the number of customers inside all the supermarkets of the chain
R12	The application notifies the customers that their turn is coming
R13	The application notifies the customer that his ticket expires
R14	The application advices the customer to choose a different slot if the selected one is occupied
R15	The application advices the customer to choose a different slot if the selected one is more crowded
R16	The application estimates the duration of a visit of customers and store the information in a database
R17	The user can log in
R18	The user can register himself in the application

3.B.2 Mapping

G1	D3 D4 D6 D15	R1 R2 R5 R6 R10 R14 R15 R16
G2	D3 D10 D11 D12 D13	R2 R3 R4 R9 R11 R12 R13
G3	D3 D6 D13 D14	R2 R5 R6 R8 R9 R12 R13 R14 R15
G4	D1 D2 D5 D7	R3 R4 R16 R17 R18
G5	D1 D7 D8 D9 D15	R1 R3 R7
G6	D1 D2 D3 D5 D8	R1 R2 R3 R4 R7 R8 R12

G1 Avoid high number of customers in the supermarket

- D3 Clients respect the tickets order
- D4 Clients respect the zone that they select in the "book a visit" process
- D6 Clients respect the duration of a visit as they indicate in the "book a visit" process
- D15 Each ticket corresponds to a single customer
- R1 The application must consider the tickets taken on the spot
- R2 The application creates a virtual queue for customers
- R5 User indicates the items that will be purchased
- R6 User indicates the approximate duration of the visit
- R10 The application tracks the number of customers who booked a slot
- R14 The application advices the customer to choose a different slot if the selected one is occupied
- R15 The application advices the customer to choose a different slot if the selected one is more crowded
- R16 The application estimates the duration of a visit of customers and store the information in a database

G2 Avoid the creation of aggregation of customers in front of the supermarket

- D3 Clients respect the tickets order
- D10 All the supermarkets provide the opening hours
- D11 All the supermarkets provide their location
- D12 Customers respect the distance from each other's
- D13 The supermarket respects the opening hours
- R2 The application creates a virtual queue for customers
- R3 The application tracks the entrance of customers
- R4 The application tracks the exit of customers
- R9 The application estimates the time needed to reach the supermarket
- R11 The application knows the number of customers inside all the supermarkets of the chain
- R12 The application notifies the customers that their turn is coming
- R13 The application notifies the customer that his ticket expires

G3 Let people saving time

- D3 Clients respect the tickets order
- D6 Clients respect the duration of a visit as they indicate in the "book a visit" process
- D13 The supermarket respects the opening hours
- D14 The user provides his current position
- R2 The application creates a virtual queue for customers
- R5 User indicates the items that will be purchased
- R6 User indicates the approximate duration of the visit
- R8 The application let the user to select a date in a calendar to allow him to book a visit
- R9 The application estimates the time needed to reach the supermarket
- R12 The application notifies the customers that their turn is coming
- R13 The application notifies the customer that his ticket expires
- R14 The application advices the customer to choose a different slot if the selected one is occupied
- R15 The application advices the customer to choose a different slot if the selected one is more crowded

G4 Infer the duration time of customers visit

- D1 Clients scan the QR code before entering the supermarket
- D2 The supermarket must have internet connection
- D5 Clients scan the QR code before exiting the supermarket
- D7 The supermarket provides a QR scanner
- R3 The application tracks the entrance of customers
- R4 The application tracks the exit of customers
- R16 The application estimates the duration of a visit of customers and store the information in a database
- R17 The user can log in
- R18 The user can register himself in the application

G5 Monitor the entrances of customers

- D1 Clients scan the QR code before entering the supermarket
- D7 The supermarket provides a QR scanner
- D8 The tickets have a QR code
- D9 The supermarket workers enter and exit without scanning the QR code
- D15 Each ticket corresponds to a single customer
- R1 The application must consider the tickets taken on the spot
- R3 The application tracks the entrance of customers
- R7 The application must assign a ticket to each user asking for it

G6 Make sure the order of the waiting queue is preserved

- D1 Clients scan the QR code before entering the supermarket
- D2 The supermarket must have internet connection
- D3 Clients respect the tickets order

- D5 Clients scan the QR code before exiting the supermarket
- D8 The tickets have a QR code
- R1 The application must consider the tickets taken on the spot
- R2 The application creates a virtual queue for customers
- R3 The application tracks the entrance of customers
- R4 The application tracks the exit of customers
- R7 The application must assign a ticket to each user asking for it
- R8 The application let the user to select a date in a calendar to allow him to book a visit
- R12 The application notifies the customers that their turn is coming

3.B.3 Use cases

1. Login

Name	User's login
Actors	User
Entry Condition	User has the application running on his device, he is already registered and he has to login.
Event flow	1. User compiles Username and Password fields. 2. User clicks on the login button. 3. CLup sends the application login result. 4. The application opens the home page. 5. If there is a notification pending, CLup sends it to the application.
Exit condition	The user can use CLup functionalities.
Exception	When User indicates a wrong username and/or password CLup sends an error message

2. Registration

Name	User's registration
Actors	User
Entry Condition	User has the application running on his device and he isn't registered on CLup
Event flow	1) User clicks on "Register" button. 2) User compiles the registration form. 3) User clicks on Confirm registration. 4) CLup notifies the user the registration result. 5) The application returns on login page.
Exit condition	User is registered on CLup platform and he can login on the application.
Exception	1) The indicated email is already registered on CLup. 2) Some mandatory fields are incorrect. The application throws an error message and returns on the registration form

3. Take a ticket

Name	Take a ticket
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. User is on the home page of the application.
Event flow	1) User clicks on “Take a ticket button”. 2) The user chooses the supermarket chain for which he wants to take the ticket. 3) CLup sends information about all the stores of the selected chain. 4) User selects the store for which he wants to take a ticket. 5) CLup creates a ticket and sends it to the user. 6) The application notifies the user operation result
Exit condition	The user receives a ticket and is in a virtual queue to enter in the desired supermarket.
Exception	When the device is not connected to internet an error is displayed on the user device.

4. Delete a ticket

Name	Delete a ticket
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. User has a ticket. User is on the home page.
Event flow	1) User clicks on “My tickets”. 2) The user selects the ticket that he wants to delete. 3) User clicks on the delete button. 4) CLup sends operation result to the user. 5) The application returns to My tickets page.
Exit condition	The selected ticket is deleted from “My tickets”.
Exception	When the device is not connected to internet an error is displayed on the user device.

5. Visualize ticket status

Name	Visualize ticket status
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. User is on the home page of the application and he wants to visualize the status of the ticket
Event flow	1) User clicks on “Visualize tickets” button in the home page 2) CLup sends list of user tickets 3) User clicks on a ticket widget, which he wants to visualize, among the all tickets shown 4) CLup sends updated information of the ticket status
Exit condition	Updated information of the ticket status are displayed on the user device
Exception	When the device is not connected to internet an error is displayed on the user device

6. Visualize reservation status

Name	Visualize reservation status
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. User is on the home page of the application and he wants to visualize the status of the reservation
Event flow	1) User clicks on “Visualize reservations” button in the home page 2) CLup sends list of user reservations 3) User clicks on a reservation widget, which he wants to visualize, among the all reservation shown 4) CLup sends updated information of the reservation status
Exit condition	Updated information of the ticket status are displayed on the user device
Exception	When the device is not connected to internet an error is displayed on the user device

7. Entrance to the supermarket

Name	Entrance to the supermarket
Actors	Supermarket
Entry Condition	A ticket is scanned at the entrance of the supermarket
Event flow	1) The supermarket notifies CLup system that a person is entered 2) The CLup system updates the queue
Exit condition	The system receives the notification that a person is entered in the supermarket
Exception	Information is collected when the supermarket has connection problems and once the connection is re-established, it will be sent

8. Exit from the supermarket

Name	Exit from the supermarket
Actors	Supermarket
Entry Condition	A ticket is scanned at the exit of the supermarket
Event flow	1) The supermarket sends to CLup system Information about the visit (ticket, visit duration) 2) The CLup system updates the queue
Exit condition	The CLup system updates the queue
Exception	Information is collected when the supermarket has connection problems and once the connection is re-established, it will be sent

9. Build statistics

Name	Build statistics
Actors	Supermarket
Entry Condition	A ticket is scanned at the exit of the supermarket
Event flow	1) The supermarket sends to CLup system Information about the visit (ticket, visit duration) 2) The system elaborates the data received from the supermarket based on past data 3) The system builds statistics (average duration of the visit of a specific user)
Exit condition	The system builds statistics
Exception	Information is collected when the supermarket has connection problems and once the connection is re-established, it will be sent

10. Book a visit

Name	Book a visit
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. User is on the home page of the application and he wants to book a visit
Event flow	1) User clicks on “Book a visit” button in the home page 2) User selects a supermarket 3) User indicates the date when he wants to book the visit 4) User clicks on “Search” button in the reservation page 5) The CLup system looks for free slots that meet user’s demands, otherwise the system returns alternative slots. 6) The user select a slot 7) The CLup system looks for alternative slots. 8) The CLup system returns a list of alternative slots as a suggestion 9) The user clicks on “reserve” button 10) The system books the visit in the selected slot 11) The system returns an ack to the user to confirm the reservation
Exit condition	confirmation of the reservation is shown on the screen
Exception	When the device is not connected to internet an error is displayed on the user device. When the selected slot is no longer available the user is notified

11. Add information about the visit during the reservation

Name	Add information about the visit during the reservation
Actors	User
Entry Condition	The system has already returned a list of slots. User is on the reservation page of the application and he wants to add some optional information about the visit.
Event flow	1) The user indicates the duration of the visit 2) The user indicates the categories of items that he intends to buy 3) The user clicks on “update” button 4) The CLup system returns a new list of slots basing on user indications
Exit condition	The CLup system returns a new list of slots basing on user indications
Exception	When the device is not connected to internet an error is displayed on the user device.

12. Selection of an alternative slot during reservation

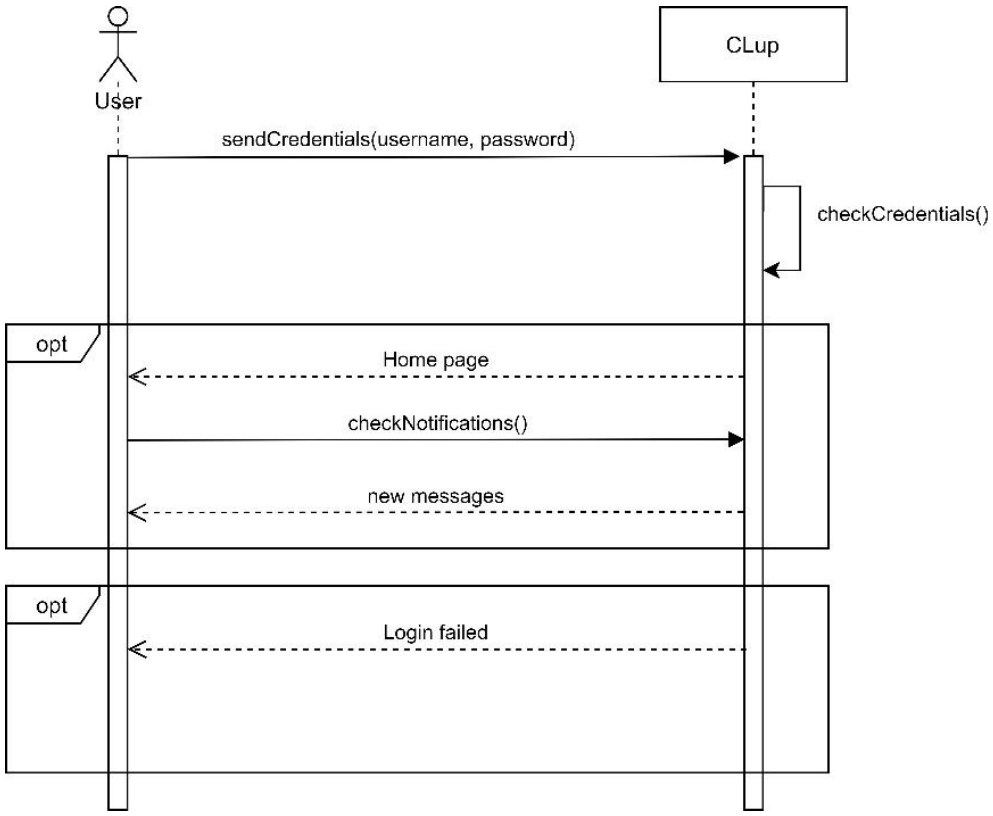
Name	Selection of an alternative slot during reservation
Actors	User
Entry Condition	User has the application running on his device and he has already logged in. The CLup system has already returned a list of possible slots
Event flow	1) The user selects a slot 2) The CLup system looks for other alternative slots. 3) The CLup system returns a list of alternative slots as a suggestion 4) The user select an alternative slot
Exit condition	The user select an alternative slot
Exception	When the device is not connected to internet an error is displayed on the user device.

13. Turn notifications

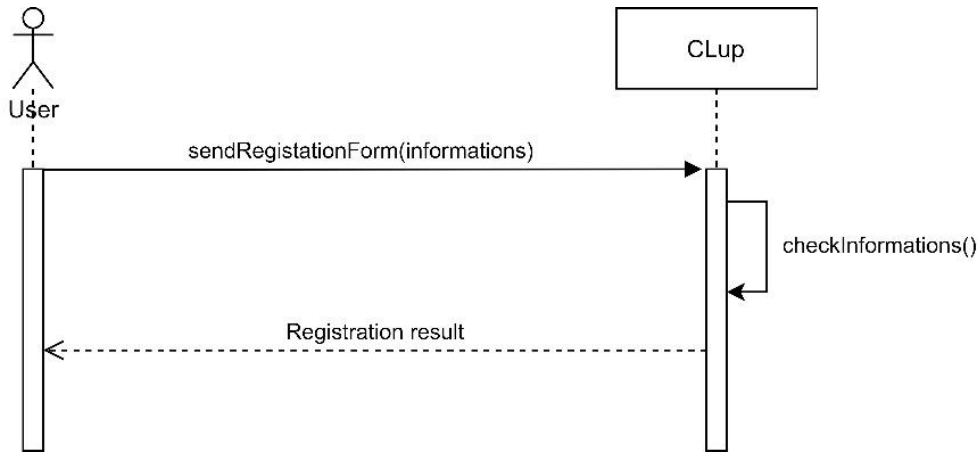
Name	Turn notifications
Actors	User
Entry Condition	The time the user takes to reach the supermarket is less than waiting time estimation in the queue
Event flow	1) CLup system send a notification to the user that it's his turn
Exit condition	User receives the notification
Exception	

3.B.4 Sequence diagrams

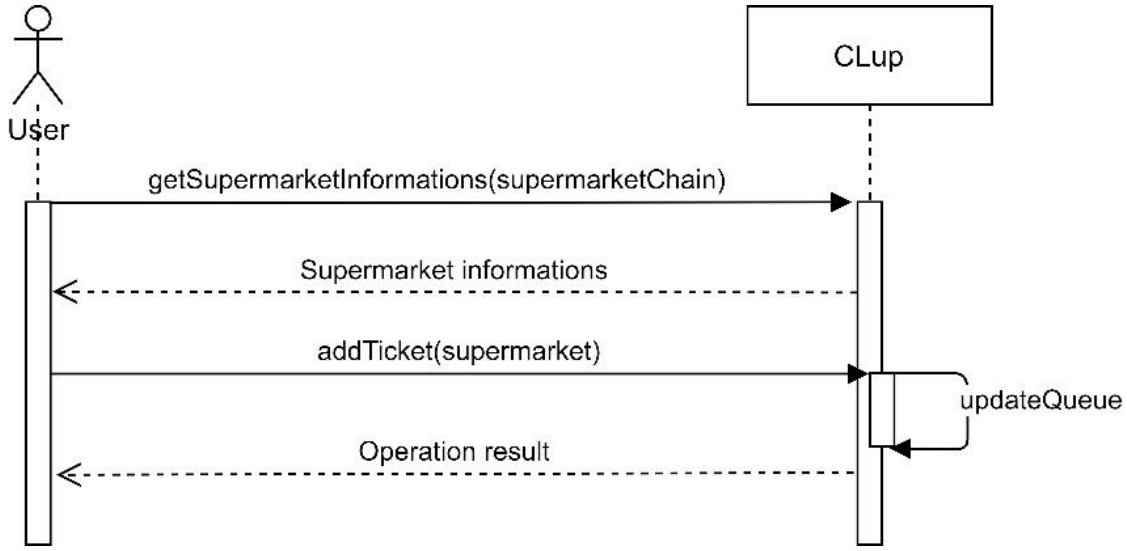
1. User's login



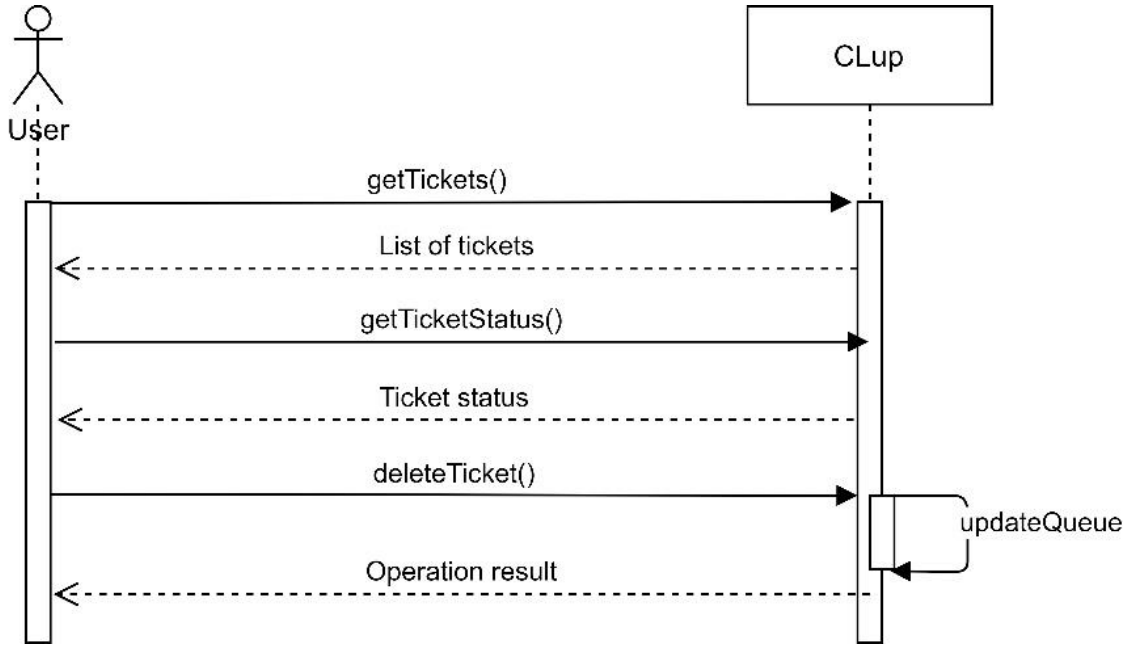
2. User's registration



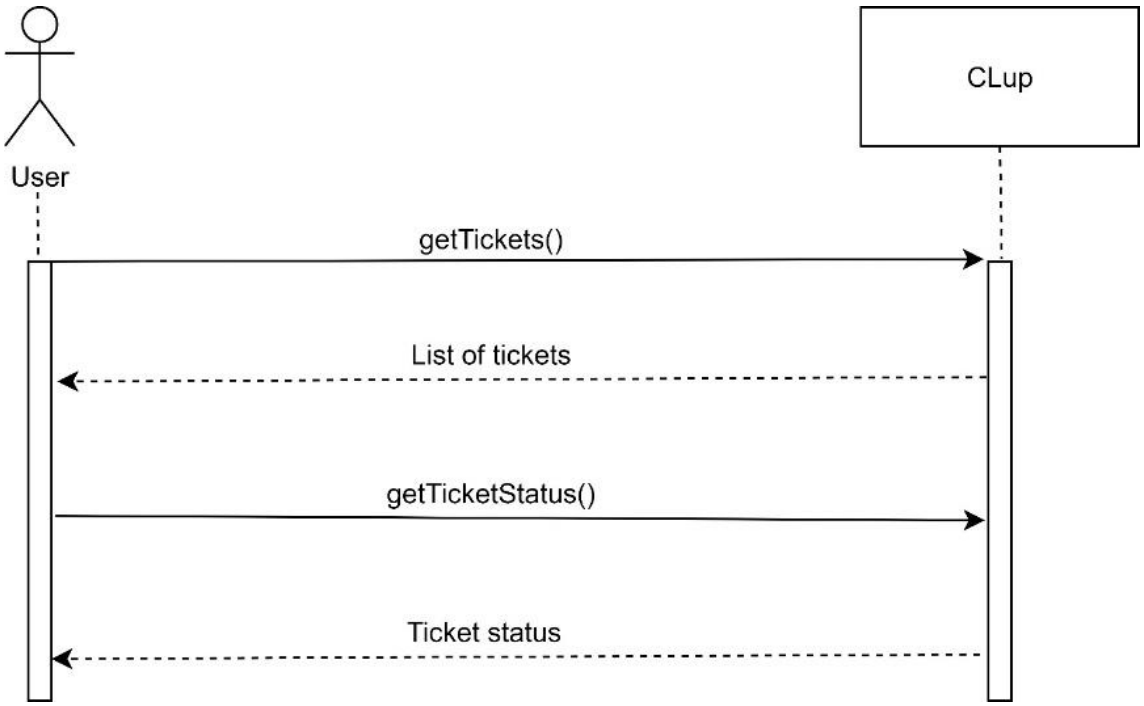
3. Take a ticket



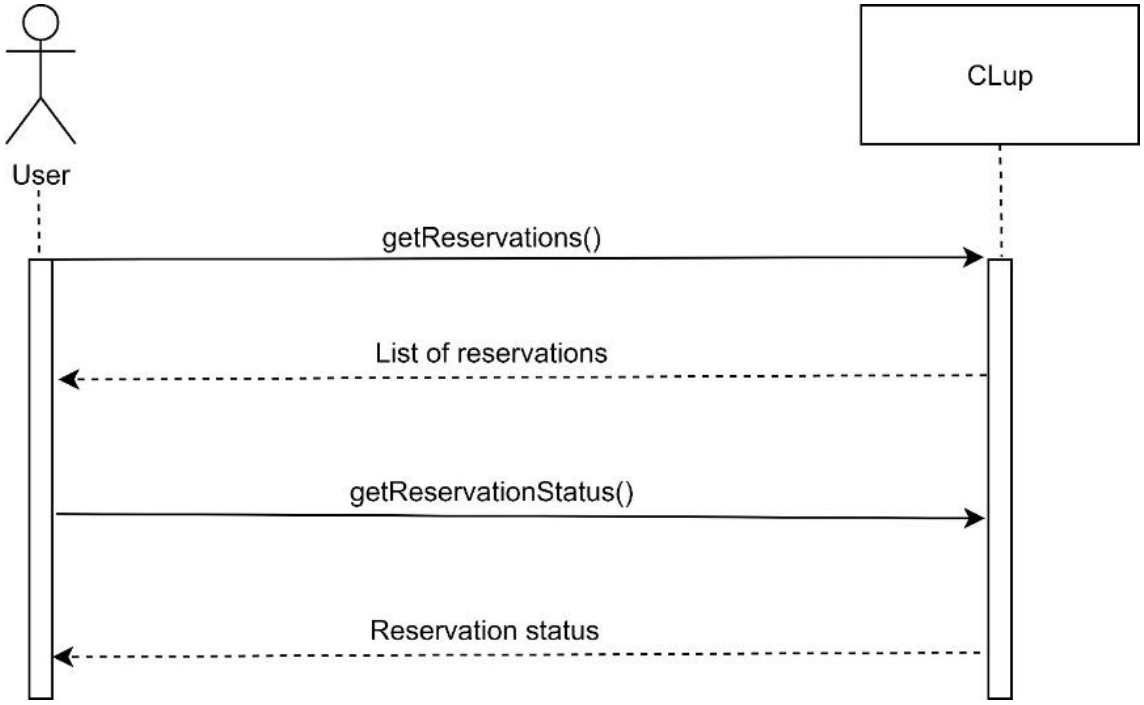
4. Delete a ticket



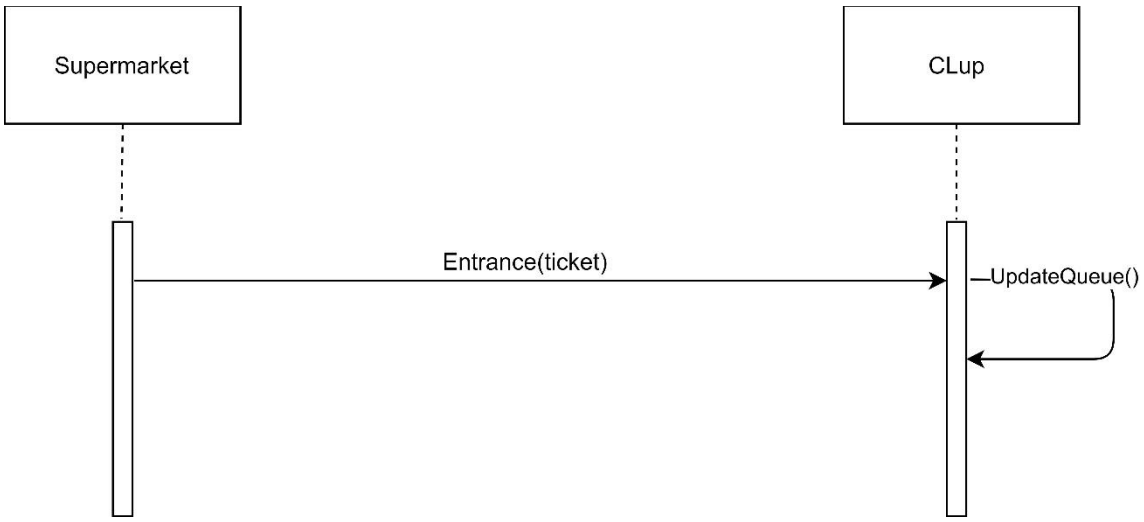
5. Visualize ticket status



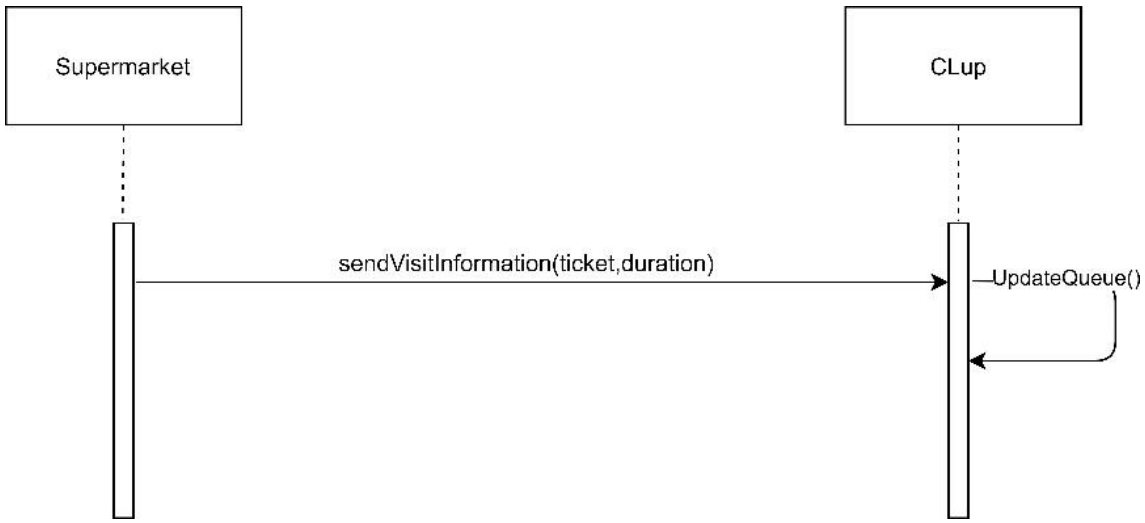
6. Visualize reservation status



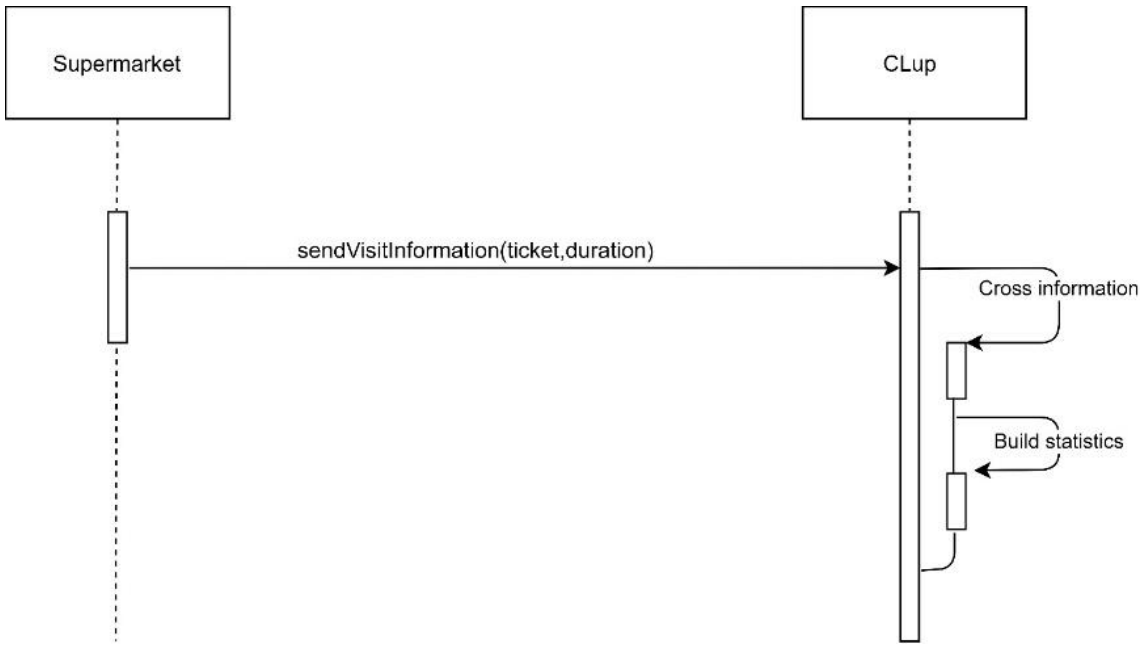
7. Entrance to the supermarket



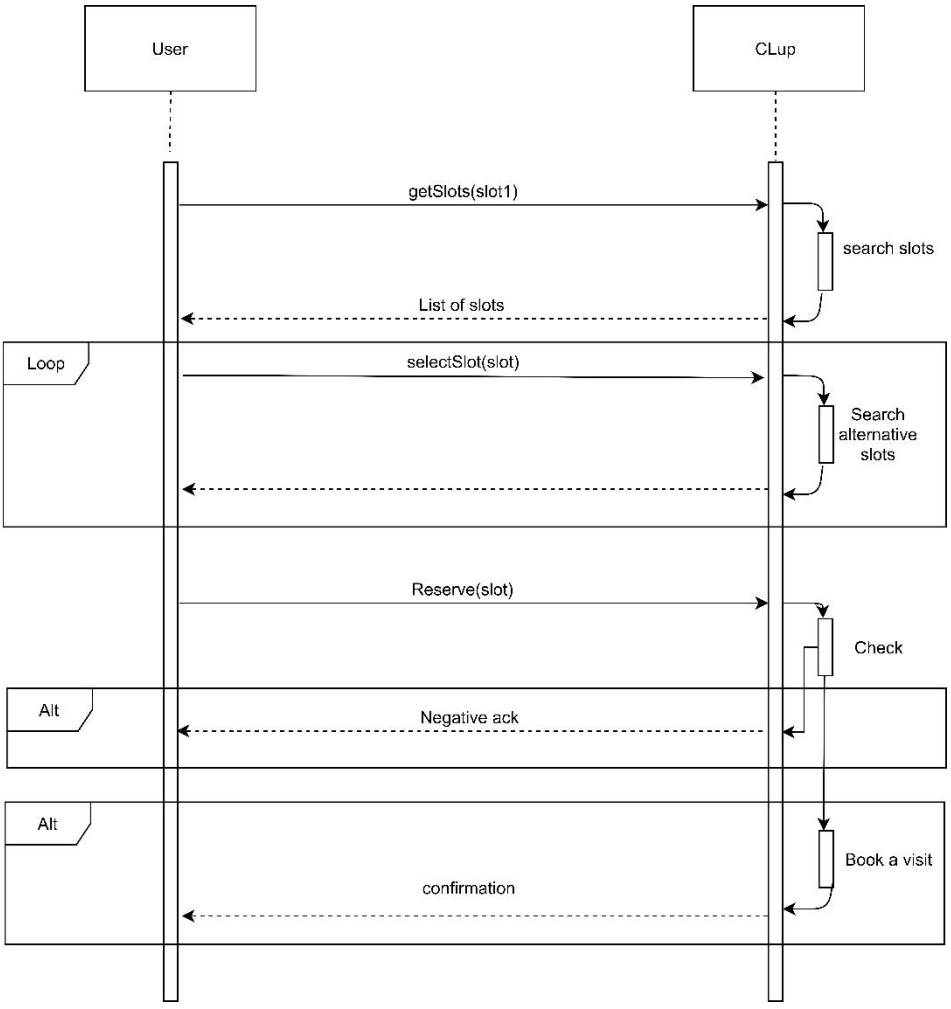
8. Exit from the supermarket



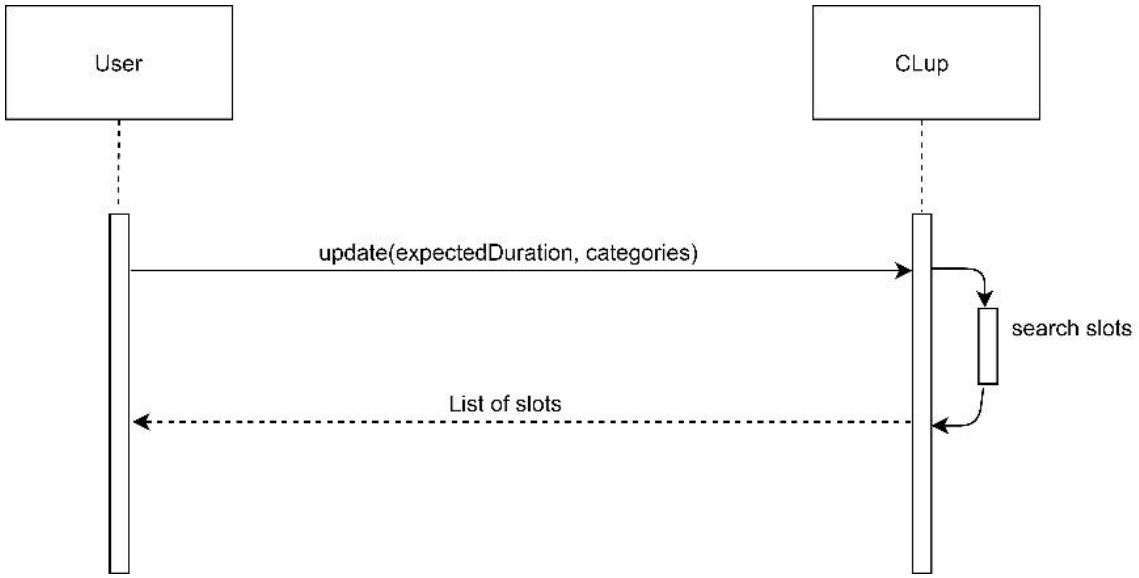
9. Build statistics



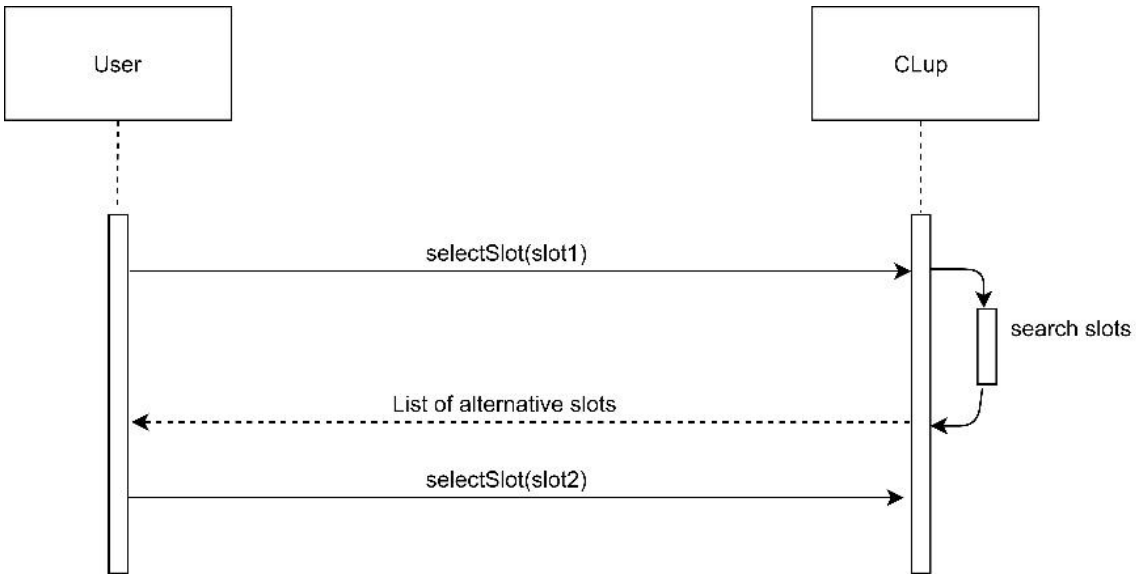
10. Book a visit



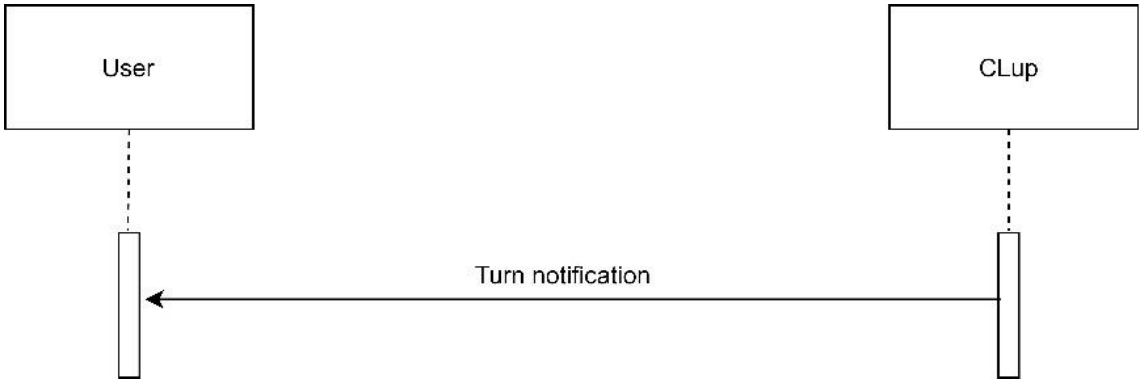
11. Add information about the visit during the reservation



12. Selection of an alternative slot during reservation



13. Turn notifications



3.C Performance Requirements

The system must be able to respond reactively to the consumer's requests.
For these reasons application services must guarantee:

- To provide a ticket or to create a reservation, to those who ask for them, in an average time which must be less than 2 seconds and above 95% less than 3 seconds.
- The waiting time to validate the login must not exceed 5 seconds, as well as regarding to the registration of a new user.
- The waiting time to handle a consumer request for available slot suggestions must not exceed 4 seconds.
- The system must guarantee to manage up to 15 000 requests por second.
- The system must be capable to handle up to 5000 supermarket queues simultaneously.
- The system must be able to hold up to 500 tickets for each queue.

The server must be online at least 99% of the time to ensure good service for the customer.
The system, to deal with user-side internet connection problems, must ensure that requests are sent as soon as the connection is re-established.

3.D Design Constrains

3.D.1 Standards compliance

The iOS and Android application respects respectively the “App Store Review guidelines” and the “Android Compatibility Definition Document”.

3.D.2 Hardware limitations

The application requires that user uses an Android or iOS platform mobile device with an internet connection to send requests to the service and a GPS sensor to localize the user.

3.D.3 Any other constrains

The system must ask permission to user to collect some information (email address, supermarket articles the user wants to buy, duration of the visit in the supermarket) at the moment of registering to the services.
If the user doesn’t provide permissions he can’t use the main features of the application. The system must also ask user’s position permission in order to estimate the time needed to reach the supermarket. If the user does not provide permissions the system considers the time to reach the supermarket as a constant.
The system must guarantee to protect user information.

3.E Software System Attributes

3.E.1 Reliability

The system maintenance is done regularly in order to prevent future problems and reduce the MTTR (Mean Time To Repair).
The reliability, which is the probability that a component will work properly during a period of time without interruptions is:

- for a whole day approximately 99,7%
- for a whole week approximately 98%
- for a whole month (30 days) approximately 92%
- for a whole year approximately 36,4%

3.E.2 Availability

The uptime (Mean Time To Failures) of the services is at least 361 days, 8 hours and 24 minutes a year. The downtime (Mean Time to Repair) is at most 87 hours and 36 minutes a year. To respect these values, the services must be available 99% of the time. This choice is a good compromise between the complexity of the IT infrastructure architecture and the availability of the services.

3.E.3 Security

All client requests are encrypted using a public-private key protocol.
Personal data are stored safely in the system and they are not visible to those who are not authorized. User credentials are encrypted and stored in the system as well. Password must not be sent in clear in case of password recovery. Personal data are protected according to General Data Protection Regulation (GDPR).

3.E.4 Maintainability

The components of the system are modular to provide greater flexibility and easier maintenance.
This modular structure could help to identify the problem and fix it, in this way the application could operate normally except for the functionalities of the component to fix. Testing code must covers at least 70% of the entire code. User may update software through the app store.

3.E.5 Portability

The application must be available from Android 4.4 KitKat which a coverage of 98.1 % of android mobiles. The application must be available also from iOS platforms.
The functionalities must be independent of the operating system.

4. Formal Analysis using Alloy

In this part we used alloy to model in a declarative way the structure and the behaviour of CLup. With this model we want to check some functionalities of the system, we focused on these 4 predicates:

- With the “World1” predicate we create the structure of CLup system. There are several supermarkets which are associated with its chain, each supermarket has two QR scanner and a queue. There are also instances of users with their registrations. This model represents the system without any interaction between entities.

- With the “World2” predicate we point out the interactions between ticket and the queue when users take tickets. There are two types of tickets: real tickets taken on the spot and virtual tickets taken on the application. A ticket is contained in in a list of active tickets if the user is in the supermarket or in queue tickets if the user is in the queue

- With the “World3” predicate we highlight the interactions between the system and the reservation. Each reservation can be linked to a virtual ticket and each reservation is contained in a slot. You can notice how a single slot can contain multiple reservations.

- With the “World4” predicate we present a complete view over all the possible combination between signatures. This predicate reflects what can be a normal situation in the system.

```
sig Username {}

sig Password {}

sig Registration {
      username : one Username ,
      password : one Password
}

sig User{
      userRegistration: one Registration,
      averageDurationSeconds: lone Int,
      tickets: set VirtualTicket,
      userReservations: set Reservation
}{
averageDurationSeconds > 0
}

abstract sig Ticket{
      ticketNumber: one Int,
      queue: one Queue,
      qr: one Qr,
}{
ticketNumber >0
}

sig RealTicket extends Ticket{
      exitQr: one Qr
}

sig Qr{}

sig VirtualTicket extends Ticket{
      user: one User,
}{
}

sig Queue{
      queueTickets: set Ticket,
      queueSupermarket: one Supermarket,
      activeTickets: set ActiveTicket,
}

sig Reservation{
      date: one Time,
      durationSeconds: one Int,
      sectors: set Sector,
      ticket: lone VirtualTicket
}{
durationSeconds > 0
}

sig Sector{
}

sig Time{
}
```

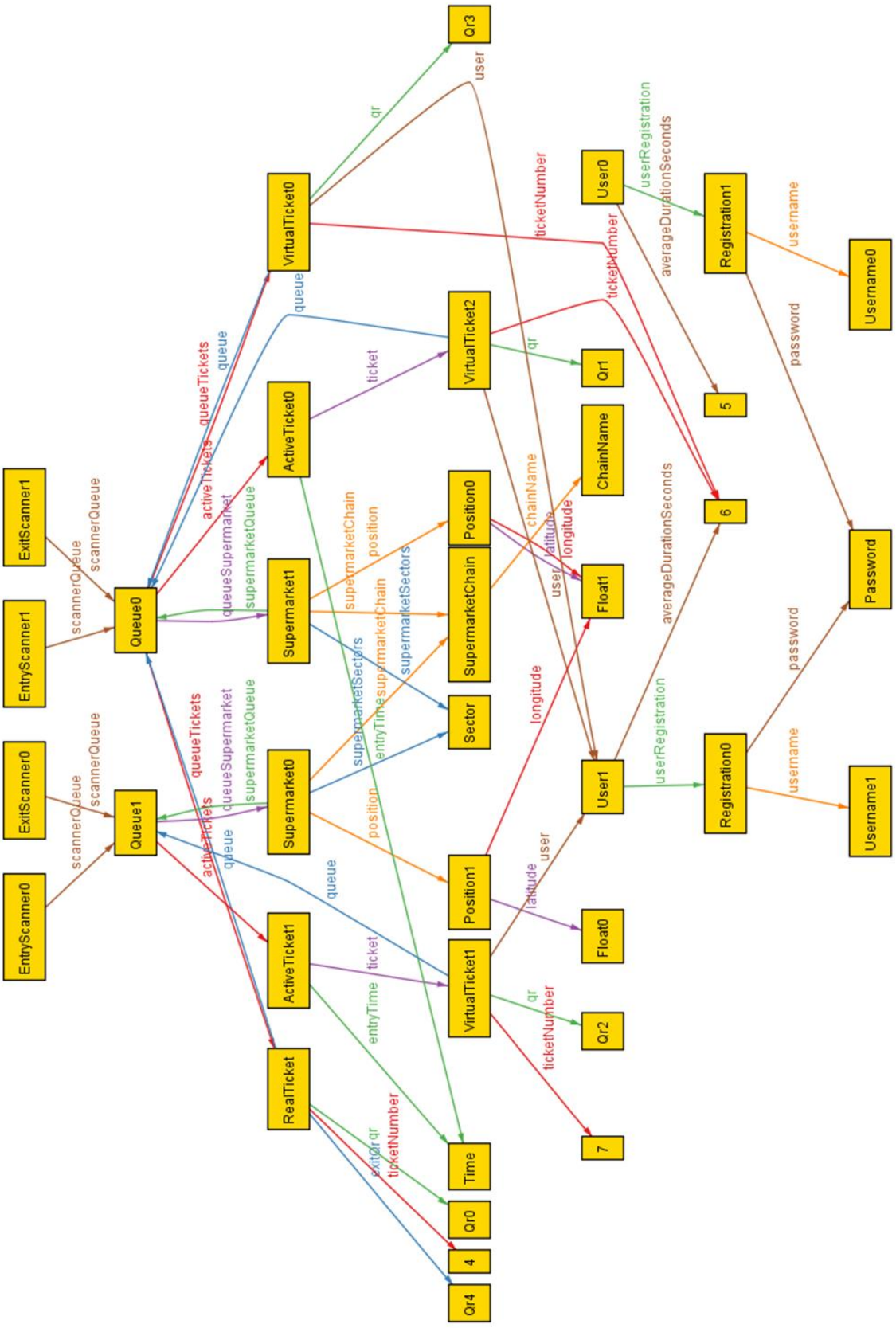
```
sig Supermarket{
      position: one Position,
      supermarketChain: one SupermarketChain,
      supermarketQueue: one Queue,
      slots: set Slot,
      supermarketSectors: set Sector
}

sig Float{
}{
}

sig Position{
      latitude: one Float,
      longitude:one Float
}

sig ChainName{}

sig SupermarketChain{
      chainName: one ChainName,
      supermarkets: set Supermarket
}

sig Slot{
      freeSlot: one Bool,
      slotReservations: set Reservation,
      slotDate: one Time,
}

abstract sig Bool{}

one sig True extends Bool{}

one sig False extends Bool{}

abstract sig QrScanner{
      scannerQueue:one Queue,
}{}

sig EntryScanner extends QrScanner{}

sig ExitScanner extends QrScanner{}

sig ActiveTicket{
      entryTime: one Time,
      ticket: one Ticket,
}
```

```
//FACTS

fact TicketQrIsUnique{
all disj T, T':Ticket| T.qr != T'.qr
}

fact ExitQrIsDifferentFromAnyQr{
all T:Ticket| all R:RealTicket|T.qr != R.exitQr
}

fact ExitQrIsUnique{
all disj T,T': RealTicket| T.exitQr = T'.exitQr
}

fact NoSupermarketInSamePosition{
all disj S,S':Supermarket| S.position.latitude != S'.position.latitude
or S.position.longitude != S'.position.longitude
}

fact UsernameIsUnique{
all disj U,U':Registration|U.username != U'.username
}

fact Registration{
all disj U,U':User|U.userRegistration != U'.userRegistration
}

fact NoSameNumbersInQueue{
all disj T,T':Queue.queueTickets| T.ticketNumber != T'.ticketNumber
}

fact NoPositionWithoutSupermarket{
all P:Position| some S:Supermarket| P in S.position
all F:Float| some P:Position|F in P.latitude or F in P.longitude
}

fact NoSlotAtSameTime{
all S:Supermarket|all disj SL,SL':S.slots| SL.slotDate != SL'.slotDate
}

fact NoSlotWithoutSupermarket{
all S:Slot| one SU:Supermarket| S in SU.slots
}

fact NoUsernameAndPasswordWithoutRegistration{
all U:Username| some R:Registration| U in R.username
all P:Password| some R:Registration| P in R.password
}

fact UniqueActiveTicket{
all disj A,A':ActiveTicket| A.ticket != A'.ticket
}

fact NoRegistrationWithoutUser{
all R:Registration| some U:User| R in U.userRegistration
}

fact NoQueueWithoutSupermarket{
all Q:Queue| some S:Supermarket| Q in S.supermarketQueue
}
```

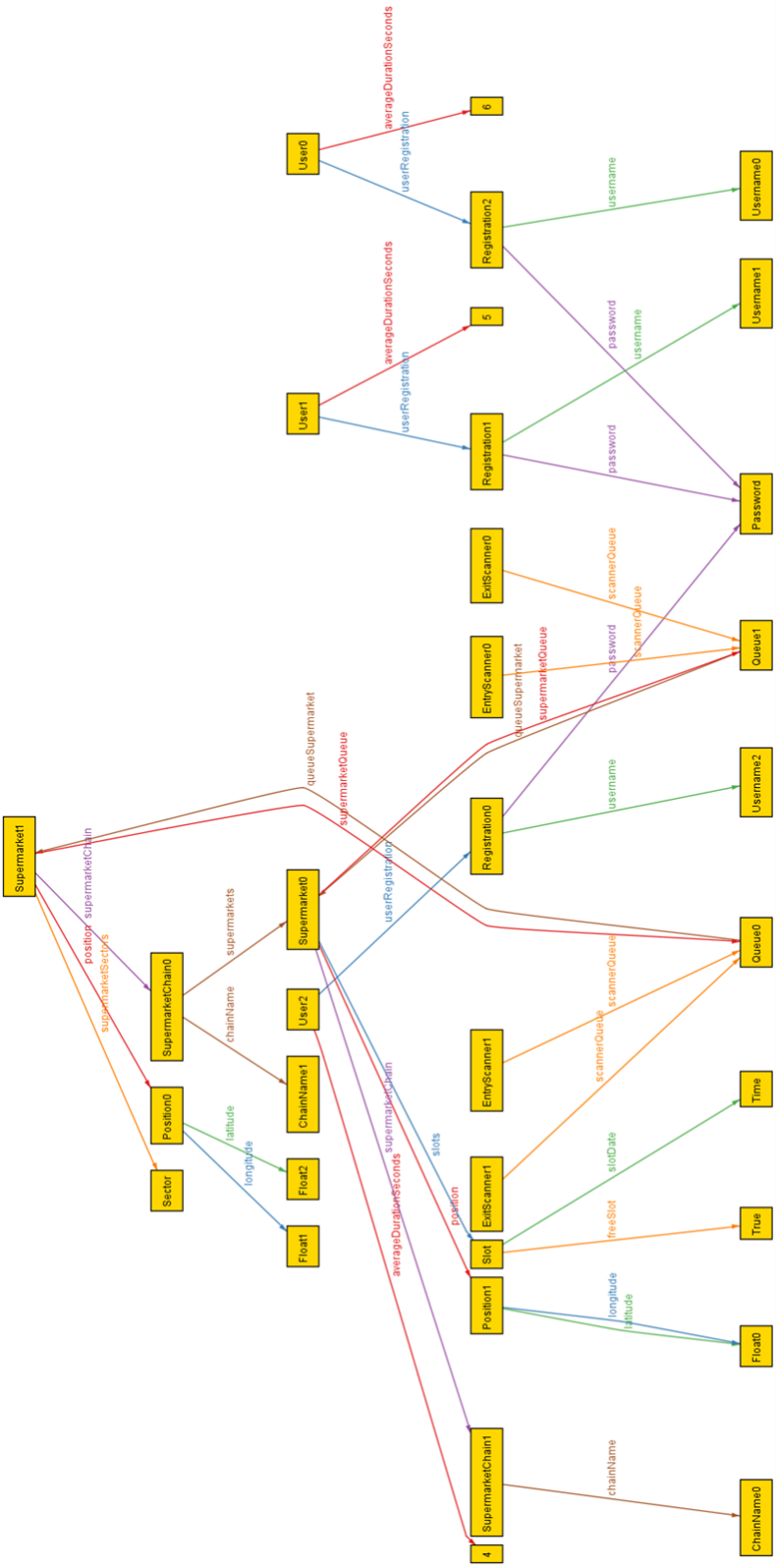
```
fact NoSupermarketChainWithoutSupermarket{
all C:SupermarketChain| some S:Supermarket| C in S.supermarketChain
}

fact SupermarketChainIsUnique{
all N:ChainName| one C:SupermarketChain| C.chainName =N
}

fact QrScanner{
all disj QU:Queue| one Q:EntryScanner| one Q':ExitScanner|
Q.scannerQueue = QU and Q'.scannerQueue = QU
}

fact NoReservationWithoutSlot{
all R:Reservation|one S:Slot| R in S.slotReservations
}

fact SupermarketQueueIsUnique{
all disj Q,Q':Queue| Q.queueSupermarket != Q'.queueSupermarket
all S: Supermarket| S.supermarketQueue.queueSupermarket = S
}

fact ReservationTicketSameSupermarket{
all S:Supermarket| all R:S.slots.slotReservations| all T:R.ticket|
T.queue.queueSupermarket = S
}

fact NoMultipleActiveTicket{
all U:User| lone A:ActiveTicket.ticket| A in U.tickets
}

fact ReservationSameSlotDate{
all S:Slot| all R:S.slotReservations| S.slotDate =R.date
}

fact NoReservationImpliesFreeSlot{
all S:Slot| #S.slotReservations = 0 implies S.freeSlot = True
}

fact ReservationSectorExistsInSupermarket{
all S:Supermarket|all SL:S.slots|all R:SL.slotReservations|all
SE:R.sectors| SE in S.supermarketSectors
}

fact NoSectorWithoutSupermarket{
all S:Sector|some S':Supermarket|S in S'.supermarketSectors
}

fact AllActiveAreInActiveQueue{
all A:ActiveTicket| one Q:Queue| A in Q.activeTickets
}

fact SameQueueForActiveTickets{
all A:ActiveTicket| A in A.ticket.queue.activeTickets
}

fact TicketQueue{
all T:Ticket|(T in T.queue.queueTickets and T not in
T.queue.activeTickets.ticket )or
(T not in T.queue.queueTickets and T in T.queue.activeTickets.ticket )
}
```

```
fact NoreservationWithoutUser{
all R:Reservation|one U:User| R in U.userReservations
}

//PREDICATES

pred world4{
#Reservation = 2
#User = 2
#VirtualTicket = 2
#Slot=2
#RealTicket =1
#Supermarket =2
#SupermarketChain =2
}

run world4 for 4

pred world3{
#Reservation = 4
#User = 1
#VirtualTicket = 2
#Slot=2
#RealTicket =0
all V:VirtualTicket| one R:Reservation| V in R.ticket
#Supermarket =1
#SupermarketChain = 1
}

run world3 for 7

pred world2{
#Reservation = 0
#Ticket = 4
#ActiveTicket = 2
#User = 2
#Supermarket =2
#SupermarketChain = 1
}
run world2 for 7

pred world1{
#Reservation =0
#Ticket =2
#User = 3
#Supermarket =2
#SupermarketChain = 2
}
run world1 for 5
```

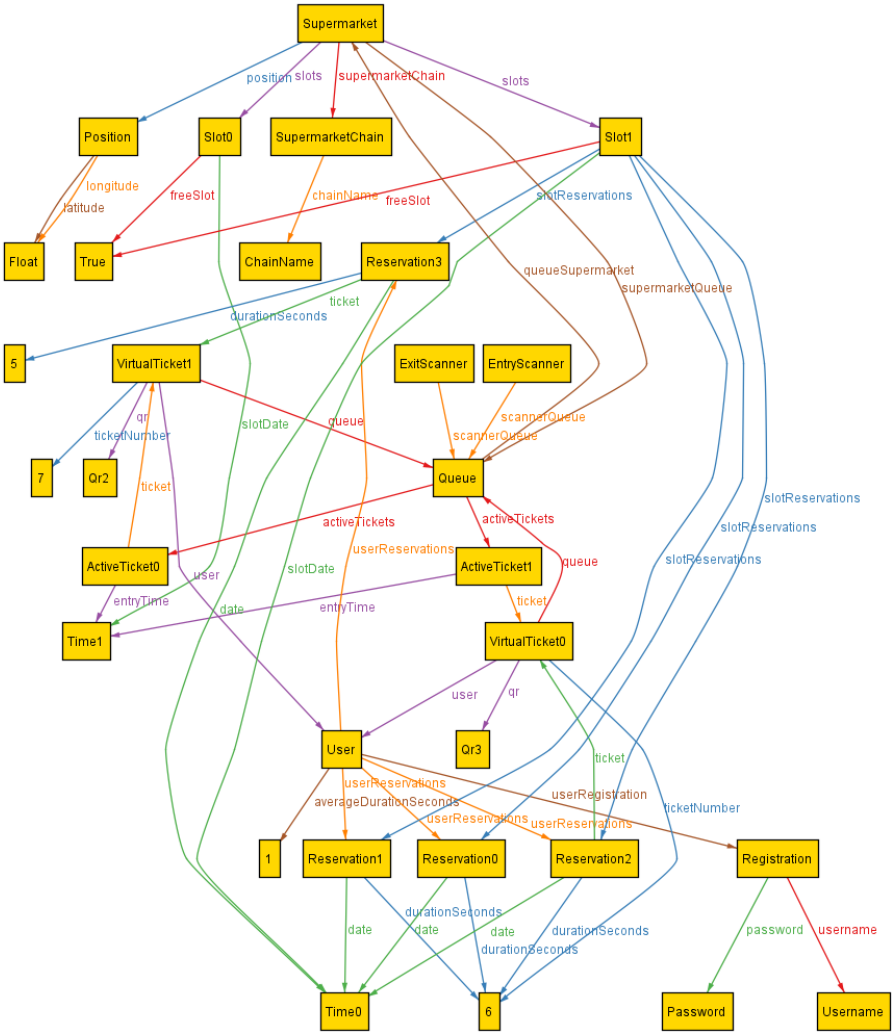
World 1

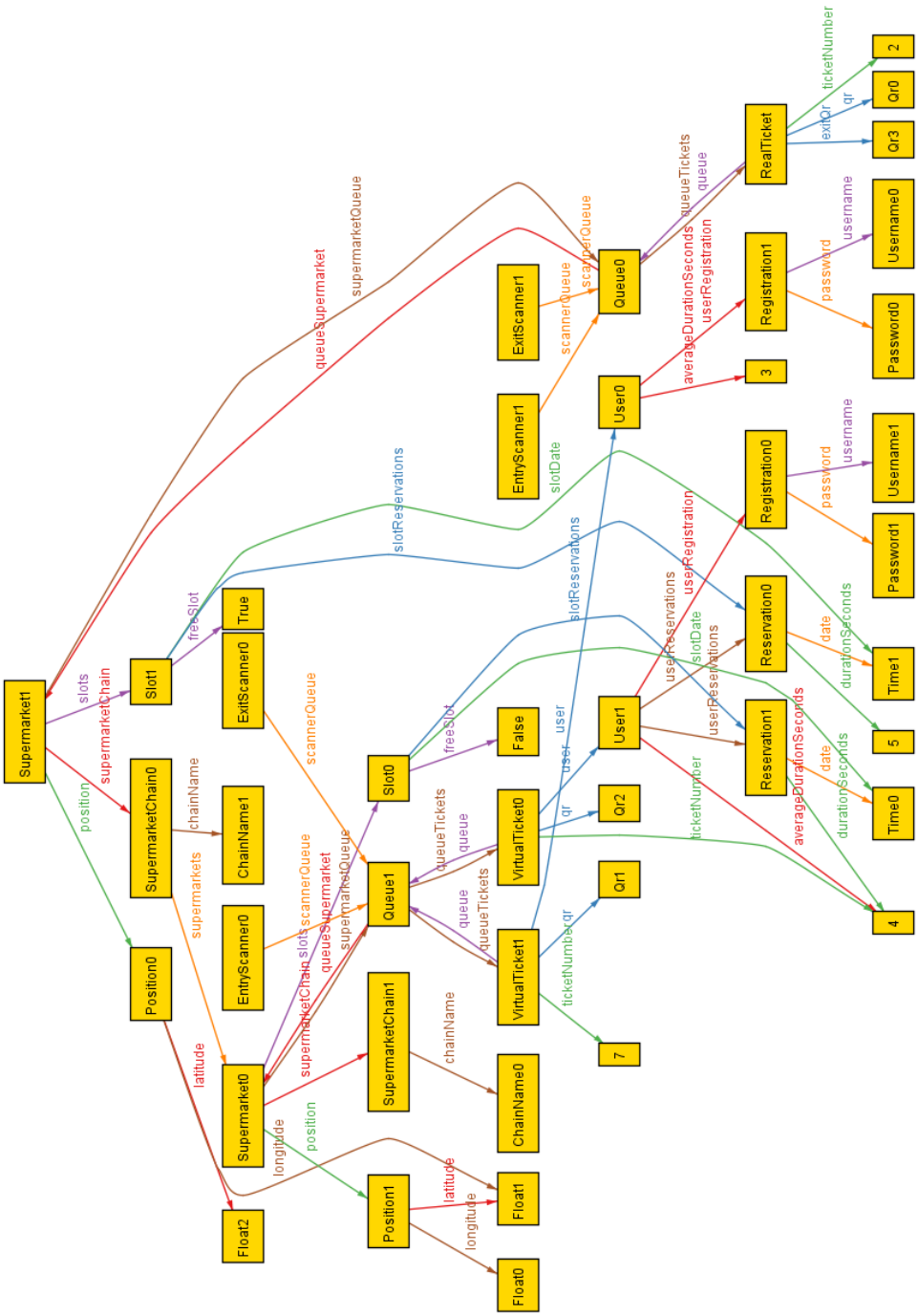

Qr3
VirtualTicket0
User0
Registration1
Username0
VirtualTicket2
Qr1
5
ActiveTicket0
ChainName
Position0
6
ExitScanner1
EntryScanner1
Queue0
Supermarket1
SupermarketChain
Float1
Password
ExitScanner0
Queue1
Supermarket0
Sector
User1
Registration0
Username1
Position1
Float0
ActiveTicket1
VirtualTicket1
Qr2
EntryScanner0
RealTicket
7
Time
Qr0
4
Qr4
qr
user
ticketNumber
userRegistration
averageDurationSeconds
username
password
queue
queueTickets
activeTickets
queueSupermarket
supermarketQueue
ticket
scannerQueue
supermarketChain
position
supermarketSectors
chainName
longitude
latitude
entryTime
exitQr

World 2


Supermarket1
SupermarketChain0
Position0
Sector
Supermarket0
User2
ChainName1
Float2
Float1
Registration0
EntryScanner1
ExitScanner1
Slot
Position1
SupermarketChain1
4
EntryScanner0
ExitScanner0
Registration1
Registration2
User0
User1
6
5
Queue0
Queue1
Password
Username2
Time
True
Float0
ChainName0
Username0
Username1
supermarketChain
positions
supermarketSectors
supermarkets
chainName
latitude
longitude
queueSupermarket
supermarketQueue
userRegistration
averageDurationSeconds
slots
position
scannerQueue
slotDate
freeSlot
password
username

World 3


Supermarket
position
slots
supermarketChain
slots
Position
Slot0
SupermarketChain
Slot1
longitude
latitude
freeSlot
chainName
freeSlot
slotReservations
Float
True
ChainName
Reservation3
queueSupermarket
supermarketQueue
ticket
durationSeconds
5
VirtualTicket1
ExitScanner
EntryScanner
qr
ticketNumber
slotDate
queue
scannerQueue
scannerQueue
7
Qr2
ticket
Queue
slotReservations
activeTickets
activeTickets
slotReservations
slotReservations
userReservations
ActiveTicket0
user
slotDate
ActiveTicket1
queue
entryTime
date
entryTime
ticket
Time1
VirtualTicket0
user
qr
User
Qr3
ticket
userReservations
averageDurationSeconds
userReservations
userReservations
userRegistration
ticketNumber
1
Reservation1
Reservation0
Reservation2
Registration
durationSeconds
date
date
date
durationSeconds
durationSeconds
password
username
Time0
6
Password
Username


Supermarket1
slots
supermarketChain
position
Slot1
freeSlot
True
ExitScanner0
scannerQueue
SupermarketChain0
chainName
ChainName1
supermarkets
EntryScanner0
Position0
latitude
Supermarket0
longitude
position
Float2
queueSupermarket
supermarketQueue
supermarketChain
slot
queueSupermarket
scannerQueue
Slot0
slotReservations
False
freeSlot
Queue1
queueTickets
queue
SupermarketChain1
chainName
ChainName0
Position1
latitude
longitude
Float1
Float0
VirtualTicket0
VirtualTicket1
queue
queueTickets
user
qr
ticketNumber
qr
Qr2
Qr1
7
User1
slotReservations
userRegistration
userReservations
slotDate
userReservations
ticketNumber
ExitScanner1
EntryScanner1
scannerQueue
slotDate
Queue0
User0
queueTickets
queue
averageDurationSeconds
userRegistration
3
RealTicket
exitQr
ticketNumber
qr
Qr3
Qr0
2
Registration1
password
username
Password0
Username0
Registration0
password
username
Password1
Username1
Reservation1
date
durationSeconds
averageDurationSeconds
Reservation0
date
durationSeconds
Time0
Time1
5
4

5.A Team effort

5.A.1 Alberto Maggioli effort

Task	Hours
Scenarios	3 h
Class diagram	1 h
Product functions	1 h
User interfaces	5 h
Hardware interfaces	0.25 h
Software interfaces	0.5 h
Communication interfaces	0.25 h
Use cases	1.50 h
Sequence diagrams	1 h
Total	13.5 h

5.A.2 Lorenzo Poretti effort

Task	Hours
Document Structure	0.5 h
Class diagram	1 h
State Charts	3 h
Use cases	3 h
Sequence diagrams	2 h
Performance requirements	1.5 h
Design constrains	1 h
Software system attributes	2 h
Total	14 h

5.A.3 Team effort

Task	Hours
Purpose	3h
Scope	4h
Definitions, Acronyms, Abbreviation	0.5 h
Class diagram	6 h
User characteristics	2 h
Domain assumptions	3 h
Software interfaces	0.5 h
List of requirement	5 h
Mapping	2 h
Use cases	1 h
Formal Analysis using Alloy	10 h
Total	37 h

6. References

[1] Alloy Documentation: http://alloytools.org/documentation.html

[2] Slides of the lectures of “Software Engineering II” course

[3] Book: “Software engineering principles and practice”, Hans Van Vliet

[4] Diagrams: made with Draw.io: https://app.diagrams.net

[5] Screens: https://mockittapp.wondershare.com