POLITECNICO
MILANO 1863

# - RASD -

Requirements Analysis and
Specification Document

---

Computer Science and Engineering

Software Engineering II

A.A. 2020/2021

Daniele Mammone - 10625264
Gianmarco Naro - 10610374
Massimo Parisi - 10583470

# Contents

# 1 Introduction

## 1.1 Purpose

The main target of this document is to describe the software through functional and non-functional requirement and is used as contractual basis between the customer and the developer. The structure of the document follows the one studied during lectures and aim to describe faithfully the software behaviour in all of its aspects.

The software in question is *CLup*, a mobile service usable through app, made both for store managers and customers. It facilitates customers to book a visit to a store and, on the other hand, to help store managers to observe the new strict rules due to *Covid-19*.

## 1.2 Scope

The main purpose of *CLup* is to facilitate customers to access at a store in **security**, both allowing them to reserve a spot on the queue for entering the store through the app and to book a visit at the store in a determined time window, selected by the user. Thanks to this, store managers can manage the **affluency** in their store more easilier, and moreover can reduce the crowd in front of the store, that is the main purpose of the application. The main idea is that when a person's number is called, he can enter the supermarket. Moreover, the app should generate a *QR Code* that the customer will scan at the entry and at the exit of the store, so that the system knows in real time how many people there are in the store. *CLup* should also estimates the *ETA* from the turn of a person. Moreover, it is able to suggest people other store options when there is a high waiting time to enter to the store requested by the user. The app also allows people to reserve their spot at the supermarket and, in order to optimize the waiting time, people can select specifics departments where they want to go in the store.

Summing up, *CLup* has this main functionalities:

- **Manage of lining up of the store**: the app will manage the accesses to the store, based on numbered tickets released to people. When it's the turn of a person, it will be authorized to enter the shop. Futhermore, the store manager is able to manage the access and the affluency to the store.
- **Booking visit**: users can book a visit at the store, in a certain time frame decided in the booking process. For them, there is no requirement of ticket, since are able to access the store only scanning the *QR Code* at the store entrance. The system will grant access if the time of entering is correct.
- **Alternatives**: the app is able to suggest other stores options and time if some store is full, or comfortable times aren't available at the moment of the booking.

### 1.2.1 World Phenomena

| | |
|---|---|
| WP1 | A user enters a supermarket |
| WP2 | A user waits in a lineup |
| WP3 | A user exits the supermarket |
| WP4 | A certain number of people is inside the supermarket |
| WP5 | A certain number of people is at a specific department of the supermarket |

### 1.2.2 Shared Phenomena

| | | |
|---|---|---|
| SP1 | The user gets a ticket/QR | M |
| SP2 | The user books a visit to the store | M |
| SP3 | The store generates in presence a ticket | M |
| SP4 | The user comes to knows how much time they have to wait before entering | U |
| SP5 | The users know how many people there are in a store at a certain moment | U |
| SP6 | The user scan the QR code and enters in the supermarket | U |
| SP7 | The user scan the QR code and exits from the supermarket | U |
| SP8 | The user can indicate the categories of items that he intend to buy | U |

### 1.2.3 Goals

| | |
|---|---|
| G1 | Allow customers to select a store and book a spot on the queue from *CLup* app |
| G2 | Allow customers to select a store and take a spot on the queue to enter as soon as possible the store from *CLup* app |
| G3 | Allow customers to book a spot on the queue from a physical ticket dispenser |
| G4 | Allow customers to select a better store option to avoid waisting a lot of time waiting to entry at a specific store |
| G5 | Allow customers to decrease waiting times specifing departments they want to visit |
| G6 | Allow customers users to manage their spots on the queue and their reservation |
| G7 | Allow customers to depart from their location in time to avoid waiting too much, and to avoid losing their turn in lineup |
| G8 | Allow store managers to decide how many people, and how many booked clients to have in the store and in which department |
| G9 | Allow the store manager to know the real situation of people that are inside the building and in which department of his store |
| G10 | Allow to grant a fair managment of users that can access the building |
| G11 | Allow to manage optimally the influx in the building and avoid gathering inside it |
| G12 | Allow the store manager to manage customers' bookings |

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

| | |
|---|---|
| QR Code | Bidimensional bar code that allows the user to check-in/check-out |
| Customer | The clients of the store, that uses the application part reserved to bookings |
| Store manager | The user that access to stores' bookings and occupancy, in order to manage the flow of customers |
| QR Code Reader | Device used to scan customers' QR Code |
| Totem | Electronic device that allows customers to book a store visit, allowing them to specifiy the same parameters that can be inserted through the app |
| QR Code Printer | Device used to print QR Code at the stores |
| Department | Part of the store that contains the same category of products |

### 1.3.2 Acronyms

| | |
|---|---|
| RASD | Requirement Analysis and Specification Document |
| ETA | Estimated Time of Arrival |
| GPS | Global Positioning System |

### 1.3.3 Abbreviations

| | |
|---|---|
| WPn | World phenomena number n |
| SPn | Shared phenomena number n |
| Gn | Goal number n |
| Rn | Requirement number n |

## 1.4 Revision History

| Version | Date | Changelog |
|---|---|---|
| 1.0 | 10/11/2020 | Overview of the specifications and initial draft of the document |

## 1.5 Reference Documents

- Specification Document
- Slides of the lectures

## 1.6 Document Structure

The structure of the document is thought with the intention of allowing simple navigation throght it. Also, various abbreviations, highlighted in Abbreviations section, have been used to make the content smoother. Hence, the structure of the document is the following one:

- **Introduction**: The main purpose of this section is to introduce in a general way the scope of the application throght the analysis of the *World Phenomena*, *Shared Phenomena* and *Goals*. Moreover, the main functions of the software are illustrated and the abbreviations, acronyms and definitions are defined in order to allow an easy reading.
- **Overall Description**: The section starts with a summary description of the UML of the software, so as to have a general presentation of the operation of the application. Then, in order to clarify the behaviour of the application, there are state charts of the most important functions and the detailed description of all software functions. Subsequently, the section ends with the most important phenomena that cannot be managed by the system.
- **Specific Requirements**: The main focus of this section is to describe the essential hardware and software interfaces and requirements, indeed *CLup* uses this interfaces to provide its services to the external world. After this, there is the core of the section represented by use cases that provides detailed information about the relation with requirements.
- **Formal Analysis Using Alloy**: This section describes formally the model using Alloy language, highlighting the main problems of the software, solving them in a formal way.
- **Effort Spent**: The main focus of this section is to track the time spent to complete this project. In particular, is highlighted the subdivision of the working hours of the various sections
- **References**: This section is dedicated to all refercences used in this project.

# 2 Overall Description

## 2.1 Product Perspective

In Figure 1 is reported an **UML Class Diagram** that represents the domain of the application with main concepts and data involved, including their relationships.

The store managers registers to the application providing all necessary information and can decide at a later stage to modify the capability options (regarding each department of the store). The customer simply downloads the application on his device to be able to use it. Here we can identify the main aspects related to CLup:

- The customer can generate a reservation, choosing between the registered chain store (and one of their specific store) or a normal store, a time slot and, optionally, the departments that they want to access; *CLup* will retrieve a ticket containing the number of the reservation and a *QR Code*.
- The customer can entry in the store where he has a reservation (on the right time slot) scanning the *QR Code* with a totem/the help of a store manager.
- The customer exit the store reusing the *QR Code*, notifying the application that a new spot in now free (on certain department).

The UML does not include every class of the actual implementation of the system.

### 2.1.1 UML Description

The **UML Class Diagram** in Figure 1 contains many classes and in this section we are going to explain shortly their functions and their scope in the project.

- **Transportation**: Is an abstract class that defines the generic means of transport that could be chosen by the customer.
  - Public transport
  - On foot
  - Bike
  - Car
- **User**: Is an abstract class that defines the generic user that can use the application. An user could be either a customer or a store manager based on their privileges in the application.
  - **Customer**: A customer can booking a visit in the store and manage his reservation

  - **Store manager**: A store manager can manage the store affluency modifing the store parameters

- **Statistic**: Is an abstrac class that defines the generic statistic that can be used from the software for the customers or the store manager. Indeed there are two types of statistics.

  - **Customer statistics**: The system uses the customer statistic in order to provide the average time spent during a visit in the store. If the customer, during a booking, decides to not specify the time that will be used during his shopping, the system make an estimation based on his previous time.
  - **Department statistics**: The system uses the statistic obtained from customers that visit a certain department's store in order to calculate the average time spent by customers in the whole store and the average time spent, always by the customers, in each department's store. Having done this, if a customer does not have his personal statistic and decide to not specify the time that he will use during his shopping, the system can based its estimation on statistic of other customers.

- **Store**: This is one of the main class and represents the store with his unique *ID*. Each store is releted to its departments that increases the granularity of the store, analyzing it in more detail, to store manager that can control the store parameters and to position that is used in order to provide to the customer both the list of bookable store sorted by distance and the notification that alerts the customers to depart from their position with the aim of arriving in time at the store. Moreover, the store class manages the queue.

- **Department**: This class is related to store an contribute to generate the statistic for each department. The store manager can modify the **maxCapacity** and the **maxAllowedBooking** parameters for each department's store in order to avoid an overcrowding inside the store.

- **Chain store**: Each store could be part of a store chain.

- **Reservation**: Each reservation its managed by the store and it has many attributes that provides informations about both the entry and exit time (expected and effective).

- **Ticket**: Every reservation has a ticket that provides the most important thing: *QR Code*. Indeed, if a customer want enter or exit the shop, must scan his *QR Code*.

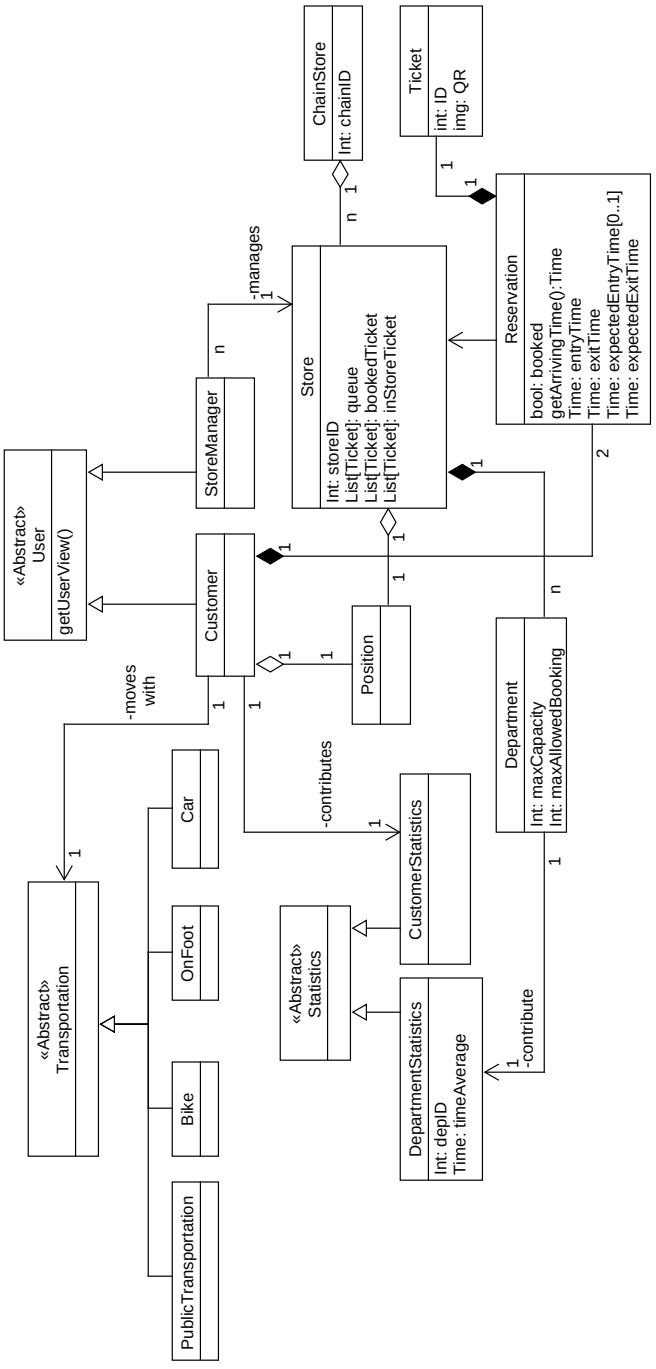


Figure 1: *UML Class Diagram*

### 2.1.2 State Charts

Now we are going to examine some essential aspects of the application, modelling their behaviours and showing the evolutin over time of their states through adequate state diagrams, which are reported below.

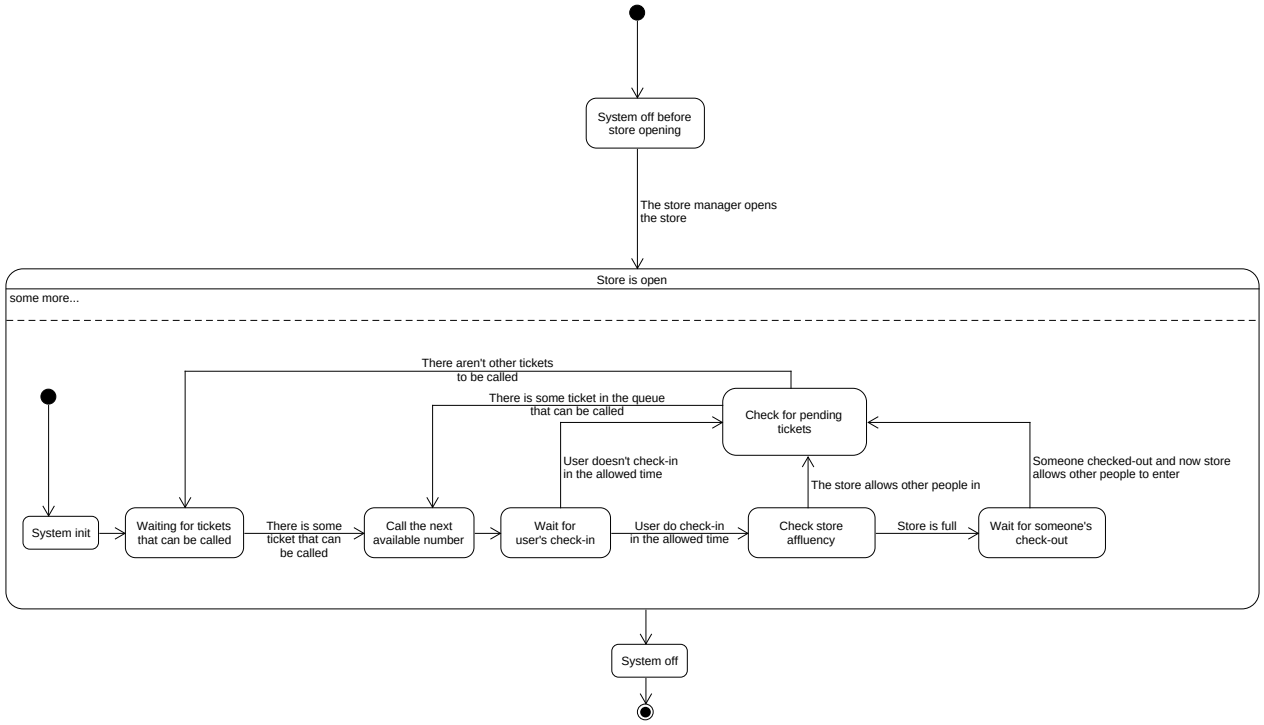


Figure 2: *Number calling system*

When the store manager opens the store, the system initialize itself and waits for some ticket that can be called. For example, tickets related to the *FIFO* queue can be called as soon as the requested zones are available, and tickets related to bookings can be called only after the start time of the booked time slot, and will be called around this time, when all the booked zones become available (to avoid starvation on calling bookings, other tickets requesting at least one booked zone won't be called until the booked tickets enters the supermarket). After a ticket is available to be called, the system notify in some way (e.g. through a store employee) that now certain ticket is allowed to enter the store. At this point, the system waits for the scan of the associated *QR code*. If the customer doesn't check-in in the assigned time, the system discard the ticket and checks if there are other available tickets. If so, it return in the state of calling the number; else, it will wait for an eligible ticket to be called. If the user, otherwise, scan his *QR code* in time, the system checks the affluence of

the store. If it's full, the system will wait for someone's check out, in order to check if some ticket can be called. Else, if the store isn't full, the system doesn't have to wait for a check out to check if there is some ticket eligible to be called. When at some time the store manager will close the store, the system begins its shutdown procedure. It's assumed that at the closure there isn't any other uncalled ticket, since at a certain point the generation of tickets will be blocked, so that that the last ticket will be served around the time of closure.

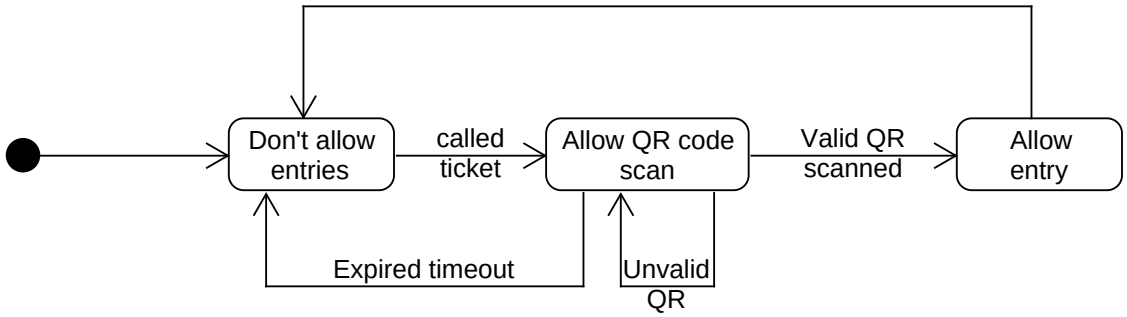


Figure 3: *QR Code scanner state machine*

The system doesn't allow entries since a ticket is called by the system described above. Than, the *QR Code* reader waits for a *QR code* to be scanned. When a *QR Code* is scanned, the system checks if it's a valid one; if so, allows the entry, otherwise notify the wrong *QR Code* and continue waiting for the right one. If the valid *QR Code* isn't scanned in time, the system block the entry and waits for the next called ticket. Otherwise, if the valid *QR Code* is scanned in time, the system allows the entry and, detected the entry, or expired the timeout, block the entries and waits for the next ticket to be called.

To avoid generating tickets that will be called after the closing hour of the store, thanks to the estimations of customers' shopping time, the system is able to decide if it can issue a ticket for entering as soon as possible in the store, or a new ticket's calling time goes over the store opening. After the system detects it can't issue tickets, the ticket generation system blocks ticket issuing until the store is closed, so that the next day the ticket generation can resume.

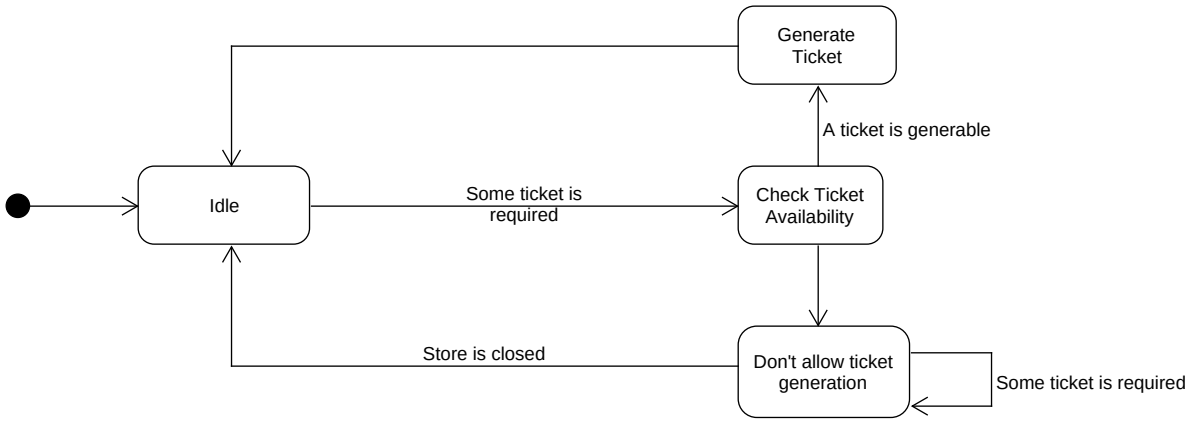


Figure 4: *Ticket generator system state machine*

## 2.2 Product Functions

In the previous chapter there were introduced, sketchily, the main features of the software in order to understand, in general, its functioning. Alternatively, in this section will be illustrated and described accurately all the functions that the software allows to do.

### 2.2.1 Getting a ticket

In order to manage the line of the supermarket, it's used a system based on the "call of numbers". Each user in the queque has a unique ID, and when his number is called (eg. by a store employee), the user is authorized to enter. This mechanism works also for the booking feature, since the system generates a number that won't be called until the beginning of the time slot selected by the user. Users can get a ticket both on the application and at a totem installed at the entry of the store (obviously, on the app users are required to select some specific store of their preference from a given list, sorted by the nearest one from the position obtained from the *GPS*). With the generation of the ticket, the user will get both an ID and a *QR Code* associate to it, to be scanned at the enter of the store. When getting a ticket on the application, users must be logged in with their credentials (if not, they must register at the services following the procedure that will be explained later), while at the Totem they can simply get the ticket without giving any data. On app, customers get a digital ticket with its associated *QR Code*; at totem, the ticket and the qr code will be printed. On both of them, there is displayed the expeted time of call. Getting the ticket, the user can indicate the department in which he is interested, and using the app can decide both to have a ticket to enter asap in the store or to plan his visit. At the store's totem, he can only get a ticket to enter as soon as possible.

Once completed the operation on the app, the user will be able to see at any moment the time needed to reach the store by the preferred mean of transport selected in the settings.

### 2.2.2 Calling process

The process to admit people in the store follows a *FIFO* logic: the first one that got a ticket, is the first to access the store. When called, a person is authorized to access the store. To register his entry, he is requested to check-in at the entry in a certain amount of time.

### 2.2.3 Check-in/Check-out

At the store entries and exities, users have to scan their *QR Codes* to respectively Check-in and Check-Out in the supermarket. If someones doesn't check-in in a prefixed amount of time, definible by the store manager, he'll lost his turn in the queue. *QR Code* scan is required for two important reasons: register someone's entry in the store for contact tracing purposed, and to check that the person entering the store is the one called, to avoid stoles of turns.

### 2.2.4 Plan a visit

The app allows to brand's customers to plan a visit in a store of their preference. At the end of the process of getting a ticket, if the user select to plan his visit to the store, the software shows the user the time table highlighting the days and the time slots available, in such a way as to allow the user to select the best option for him. If the options do not satisfy the user, he can ask the application for help to find other supermarkets with more comfortable hours. If a reservation process is interrupted in the middle, the user will be able to resume it reopening the app.

### 2.2.5 Managing single department

The system can infer the average time spent by a user in the supermarket using the previous visits as informations. If the client is taking a spot on the queue or booking a visit on the app, he will be asked to insert a reasonable duration of his visit, or let the application to fill this parameter for him: if there aren't enough informations on the specific client, the app will use informations from other clients that did a similar shopping, else will infer the information from the specific client's previous visits. The same thing happens at the totem, where or the client inserts the estimation, or the system will use the other datas in its possession.

### 2.2.6 Notify users

If some account have non completed visits, depending on the position of the client (retrieved by *GPS*), and the preferred option of reaching the selected store

(eg. on foot, by car or public transport), the app will notify the client when he can exit home, to arrive in time to enter the supermarket, without losing its turn and avoiding long waiting times. Notice that not all of the moving options may be available in all the places, since it depends on the used map service.

### 2.2.7 Cancel a visit

If the user can not reach the store in time, he can decide to cancel the booked visit or the generated ticket. In this case the software delete the customer from the queue and rearrange the last one.

### 2.2.8 Store capacity

The app, also, allows store managers to see the store's affluency and to modify the store capacity in every single repart. When a repart capacity is changed, the already made bookings will be rescheduled at the first available entry time, notifiying each user of the change. The priority is to allow customers that booked the visit to enter in the same day, if it's possible. If any time on the same day can be allocated, the bookings will be cancelled and the user notified. A such operation may reduce in the already booked days the number of admitted non-booked clients. If this operation is made on working hours, it may be cause an anticipation of the time when the system will not deliver anymore other tickets for the queue.

### 2.2.9 Infer waiting time from users

The system can infer the average time spent by a user in the supermarket using the previous visits as informations. If the client is taking a spot on the queue or booking a visit on the app, he will be asked to insert a reasonable duration of his visit, or let the application to fill this parameter for him: if there aren't enough informations on the specific client, the app will use informations from other clients that did a similar shopping, else will infer the information from the specific client's previous visits. The same thing happens at the totem, where or the client inserts the estimation, or the system will use the other datas in its possession.

## 2.3 Allow a fair management of the accesses

The system is able to manage in a fair way the process of releasing tickets and accepting bookings. In fact, depending on the store's closing time, and others datas in its possession, such as the number of people in queue, the estimated time of permanency and some other statistics, it's able to block the generation of new tickets, to avoid the generation of turns that goes over the closing time. Moreover, the store manager is able to decide the maximum number of contemporary bookings allowed in the whole store and each department, to avoid that it's really difficult to enter the store without a booking.

## 2.4 User Characteristics

*CLup* gives access to two different sets of functionalities based on the two different category of users:

- **Customer**: Allows user to book a visit to the supermarket. The software generates a *QR Code* that the user can use to enter in the supermarket. The user can indicate the exact list of items that he intends to purchase, or, at least, the categories of items that he intends to buy and, also, he can indicates the approximate expected duration of the visit. Moreover, the user can see the *ETA* to enter the store. *CLup* also allows clients to obtain their book at the store entry.

- **Store Manager**: The Store Manager can view information about the store, in particular he can access to the reservations made by users to predict the future affluency and check the level of affluency in the store (and also in specific zones of the store).

## 2.5 Assumptions, Dependencies, Constraints

### 2.5.1 Domain Assumptions

| | |
|---|---|
| DA1 | Date and time on the devices on which *CLup* runs are always correct |
| DA2 | Internet connection works always without errors |
| DA3 | Customer's position retrieved by *GPS* is accurate |
| DA4 | In each store, different objects belonging to the same category are in the same department |
| DA5 | No one maliciously forge a ticket !!!!!!!!!! |
| DA6 | Totems always work properly and are not damaged |
| DA7 | The customer's smartphone screen is not damaged and the *QR Code* is readable |
| DA8 | The *Maps API* always calculate the optimal route |
| DA9 | Every store has a unique name and address combination |
| DA10 | *QR Code* readers are always working |
| DA11 | The store capacity inserted by store manager is always correct |
| DA12 | Customer respects his time slot, without remaining over time |
| DA13 | Each customer scans his QR Code at the enter and enters the supermarket only through the allowed entries. |
| DA14 | Each paper ticket is not ruined and readable. |

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

The user can use the service using a personal mobile device (eg. a smartphone, a smartwatch and a tablet), or through a totem externally installed at each store using CLup to manage the access of clients. Store managers can also use a personal mobile device to manage the store. In this section we'll present the general mockups relative to the smartphone version of both customer and manager management, since it is the landmark for the development of the entire system.

### 3.1.2 Hardware Interfaces

The managers' side of the application doesn't require any special hardware requirement except for a network module installed in their device, since all of the information required is already stored in the system. Clients' side, anyways, requires a working GPS module, a working network module and a not broken display to scan the generated QR Code at the store entry. Instead, at the store we need: a device with a ticket printer and a working network module to generate and print tickets, some (at least one for entry and one for exit) optical scanners to scan QR Codes, and another device with a working network module to manage ticket calling.

### 3.1.3 Software Interfaces

The system can send notification to the customer with regard to notify him to depart in time for reach the store and to avoid losing his turn in lineup. To do this, the system uses a public API to provide customer his exact position and, so, calculate the optimal route from customer's position to the store. The exact position is also used to sort by distance the list of bookable store. Moreover, the system permits to store manager to communicate with customers in case of a reservation must be deleted or postponed for every reason.

### 3.1.4 Communication Interfaces

It is necessary to develop a communication interface in order to allow the different systems to communicate one to each other, achieving the integration of the store's side and of the user's side.

## 3.2 Functional Requirements

### 3.2.1 List of Requirements

| | |
|---|---|
| R1 | The system must allow the customers to register |
| R2 | The system must allow store managers to register their store |
| R3 | The system must allow customers to log in |
| R4 | The system must allow store managers to log in |
| R5 | The system allows the customers to view their visits |
| R6 | The system allows the customers to cancel their visits |
| R7 | The system allows the customers to modify their visits |
| R8 | The system allows the customers to select their favorite means of transportation |
| R9 | The system allows the customers to select some or all the departments in which the customers are interested in doing shopping |
| R10 | The system must let in customers only if is their turn |
| R11 | The system must consider the extimate shopping time insert by customers |
| R12 | The system must show the customers of the time periods in which they can enter the store in order to respect the time selected by the customers |
| R13 | The system have to make a reasonable estimate of when a user with a spot on the queue is able to enter the store |
| R14 | The system can send notification to the clients |
| R15 | The system is able to ask for the position of the customers |
| R16 | The system permits to store manager to modify the maximum capacity of the store and its departments |
| R17 | The system permits to store manager to view the store parameters |

| | |
|---|---|
| R18 | The system allows the manager to establish the maximum simultaneously allowed booked clients in the store (or in a specific department) |
| R19 | The store manager can view the reservation of each client |
| R20 | The store manager can modify the reservation of each client |
| R21 | The store manager can cancel the reservation of each client |
| R22 | The store manager can handle the opening and closing time of the store |
| R23 | The system knows the situation in real time of each store |
| R24 | The system takes trace of each customer entry and exit from the store |
| R25 | The system contains a list of bookable stores |
| R26 | The systemis able to print a paper ticket |
| R27 | The system can reasonably estimate the time needed from a specific user to complete his shopping |
| R28 | The system must save clients' tickets |
| R29 | The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store |
| R30 | The system is able to smartly call clients with a ticket to enter the building depending on reservations and people inside the building |
| R31 | The system is able to scan and analyze a QR Code |
| R32 | The system is able to send emails |

### 3.2.2 Mapping with goals

- **G1: Allow customers to select a store and book a spot on the queue from CLup app**
  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log in
  - **R9**: The system allows the customers to select some or all the departments in
  - **R11**: The system must consider the extimate shopping time insert by customers
  - **R12**: The system must show the customers of the time periods in which they can enter the store in order to respect the time selected by the customers
  - **R15**: The system is able to ask for the position of the customers
  - **R25**: The system contains a list of bookable stores
  - **R28**: The system must save clients' tickets
  - **R23**: The system knows the situation in real time of each store which the customers are interested in doing shopping

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA3**: Customer's position retrieved by GPS is accurate
  - **DA9**: Every store has a unique name and address combination
- **G2: Allow customers to select a store and take a spot on the queue to enter as soon as possible the store from CLup app**
  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log inn
  - **R9**: The system allows the customers to select some or all the departments in
  - **R11**: The system must consider the extimate shopping time insert by customers
  - **R15**: The system is able to ask for the position of the customers
  - **R23**: The system knows the situation in real time of each store which the customers are interested in doing shopping
  - **R25**: The system contains a list of bookable stores
  - **R28**: The system must save clients' tickets

  - **R29**: The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA3**: Customer's position retrieved by GPS is accurate
  - **DA9**: Every store has a unique name and address combination

- **G3: Allow customers to take a spot on the queue from a physical ticket dispenser**

  - **R9**: The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
  - **R11**: The system must consider the extimate shopping time insert by customers
  - **R26**: The system is able to print a paper ticket
  - **R28**: The system must save clients' tickets
  - **R29**: The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA6**: Totems always work properly and are not damaged

- **G4: Allow customers to select a better store option to avoid wasting a lot of time waiting to enter a specific store.**

  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log in
  - **R8**: The system allows the customers to select their favorite means of transportation
  - **R15**: The system is able to ask for the position of the customers
  - **R23**: The system knows the situation in real time of each store
  - **R25**: The system contains a list of bookable stores
  - **R29**: The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store

  - **DA1**: Date and time on the devices on which CLup runs are always correct

  - **DA2**: Internet connection works always without errors
  - **DA3**: Customer's position retrieved by GPS is accurate
  - **DA8**: The Maps API always calculate the optimal route
  - **DA9**: Every store has a unique name and address combination

- **G5: Allow customers to decrease waiting times specifying departments they want to visit**
  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log in
  - **R8**: The system allows the customers to select their favorite means of transportation
  - **R9**: The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
  - **R13**: The system have to make a reasonable estimate of when a user with a spot on the queue is able to enter the store
  - **R24**: The system takes trace of each customer entry and exit from the store
  - **R27**: The system can reasonably estimate the time needed from a specific user to complete his shopping

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors

- **G6: Allow customers users to manage their spots on the queue and their reservation**
  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log in
  - **R5**: The system allows the customers to view their visits
  - **R6**: The system allows the customers to cancel their visits
  - **R7**: The system allows the customers to modify their visits
  - **R28**: The system must save clients' tickets

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors

- **G7: Allow customers to depart from their location in time to avoid waiting too much, and to avoid losing their turn in lineup**
  - **R1**: The system must allow the customers to register
  - **R2**: The system must allow customers to log in
  - **R8**: The system allows the customers to select their favorite means of transportation
  - **R9**: The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
  - **R14**: The system can send notification to the clients
  - **R15**: The system is able to ask for the position of the customers
  - **R24**: The system takes trace of each customer entry and exit from the store
  - **R27**: The system can reasonably estimate the time needed from a specific user to complete his shopping
  - **R28**: The system must save clients' tickets

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA8**: The Maps API always calculate the optimal route
  - **DA9**: Every store has a unique name and address combination
- **G8: Allow store managers to decide how many people, and how many simultaneous booked clients to have in the store and in which department**
  - **R2**: The system must allow store managers to register their store
  - **R4**: The system must allow store managers to log in
  - **R14**: The system can send notification to the clients
  - **R16**: The system permits to store manager to modify the maximum capacity of the store and its departments
  - **R18**: The system allows the manager to establish the maximum simultaneously allowed booked clients in the store (or in a specific department)
  - **R22**: The store manager can handle the opening and closing time of the store
  - **R28**: The system must save clients' tickets

  - **DA1**: Date and time on the devices on which CLup runs are always correct

  - **DA2**: Internet connection works always without errors

- **G9: Allow the store manager to know the real situation of people that are inside the building and in which department of his store**
  - **R2**: The system must allow store managers to register their store
  - **R4**: The system must allow store managers to log in
  - **R17**: The system permits to store manager to view the store parameters
  - **R24**: The system takes trace of each customer entry and exit from the store
  - **R31**: The system is able to scan and analyze a QR Code
  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA10**: QR Code readers are always working
  - **DA13**: Each customer scans his QR Code at the enter and enters the supermarket only through the allowed entries.

- **G10: Allow to grant a fair management of users that can access the building.**
  - **R2**: The system must allow store managers to register their store
  - **R4**: The system must allow store managers to log in
  - **R10**: The system must let in customers only if is their turn
  - **R18**: The system allows the manager to establish the maximum simultaneously allowed booked clients in the store (or in a specific department)
  - **R22**: The store manager can handle the opening and closing time of the store
  - **R29**: The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store
  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors

- **G11: Allow to manage optimally the influx in the building and avoid gathering inside it**
  - **R2**: The system must allow store managers to register their store
  - **R10**: The system must let in customers only if is their turn
  - **R24**: The system takes trace of each customer entry and exit from the store
  - **R27**: The system must save clients' tickets
  - **R30**: The system is able to smartly call clients with a ticket to enter the building depending on reservations and people inside the building
  - **R31**: The system is able to scan and analyze a QR Code

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors
  - **DA10**: QR Code readers are always working
  - **DA7**: The customer's smartphone screen is not damaged and the QR Code is readable
  - **DA13**: Each customer scans his QR Code at the enter and enters the supermarket only through the allowed entries
  - **DA14:** Each paper ticket is not ruined and readable
- **G12: Allow the store manager to manage customers' bookings**
  - **R2**: The system must allow store managers to register their store
  - **R4**: The system must allow store managers to log in
  - **R19**: The store manager can cancel the reservation of each client
  - **R20**: The store manager can modify the reservation of each client
  - **R21**: The store manager can view the reservation of each client
  - **R27**: The system must save clients' tickets

  - **DA1**: Date and time on the devices on which CLup runs are always correct
  - **DA2**: Internet connection works always without errors

### 3.2.3 Use Cases

#### 3.2.3.1 Registration of a customer

| | |
|---|---|
| Name | Registration of a customer |
| Actors | Customer |
| Entry Condition | Customer has the internet connection available and has accessed the application on its device |
| Event Flow | 1. Customer visualizes the initial page of the app<br>2. Customer clicks on "Sign up as customer" button<br>3. Customer inserts a username, a password, an e-mail, his name, his surname and his phone number as mandatory fields<br>4. The system checks the datas of the customer<br>5. The system saves the information of the customer |
| Exit Conditions | Customer is successfully registered to the application |
| Exception | 1. The username is already present in the system<br>2. The customer did not fill up all the mandatory fields with valid data<br>If one or more of the above situations occur, the application will throw an error message and will return to the registration form page |

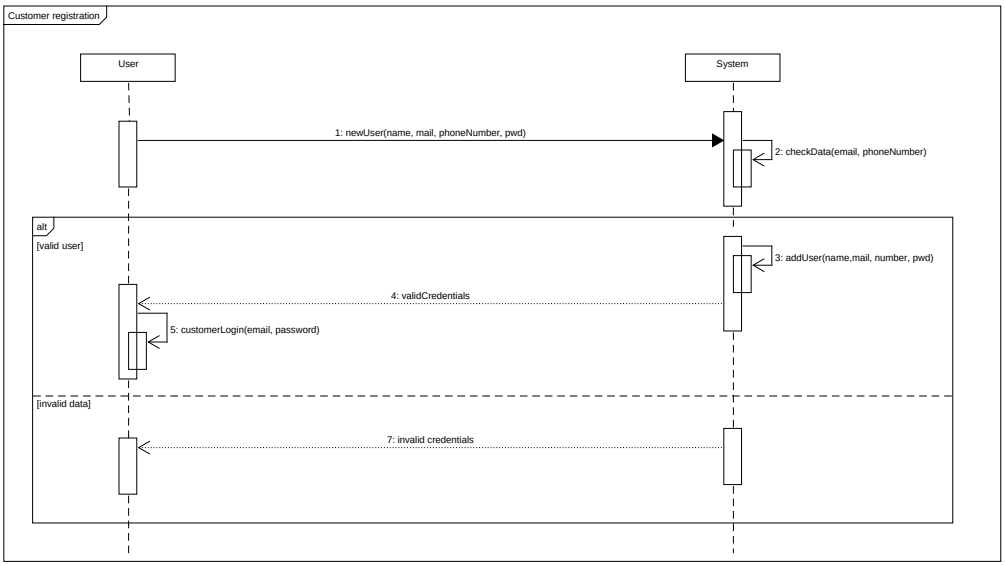


Figure 5: Sequence Diagram of User Registration

**Required functional requirements:**

- **R1:** The system must allow the customers to register
- **R5:** The system must save the customers data registration
- **R8:** The system must guarantee that each customer e-mail is unique

#### 3.2.3.2 Registration of a store

| Name | Registration of a store |
|---|---|
| Actors | Store manager |
| Entry Condition | Store manager has the internet connection available and has accessed the application on its device |
| Event Flow | 1. Store manager visualizes the initial page of the app<br>2. Store manager clicks on "Sign up as store" button<br>3. Store manager compile all the mandatory fields concerning the store<br>4. Store manager loads a certification document which proves that it is a real store<br>5. The system validates the certification<br>6. The system confirms the registration of the store<br>7. The system saves the information of the store |
| Exit Conditions | The store is successfully registered to the application |
| Exception | 1. The store is already present in the system<br>2. The store manager did not fill up all the mandatory fields with valid data<br>3. The certification is invalid<br>If one or more of the above situations occur, the application will throw an error message and will return to the registration form page |

**Required functional requirements:**

- **R2:** The system must allow store managers to register their store
- **R6:** The system must save the store managers data registration and store parameters
- **R7:** The system must guarantee that each store manager ID is unique

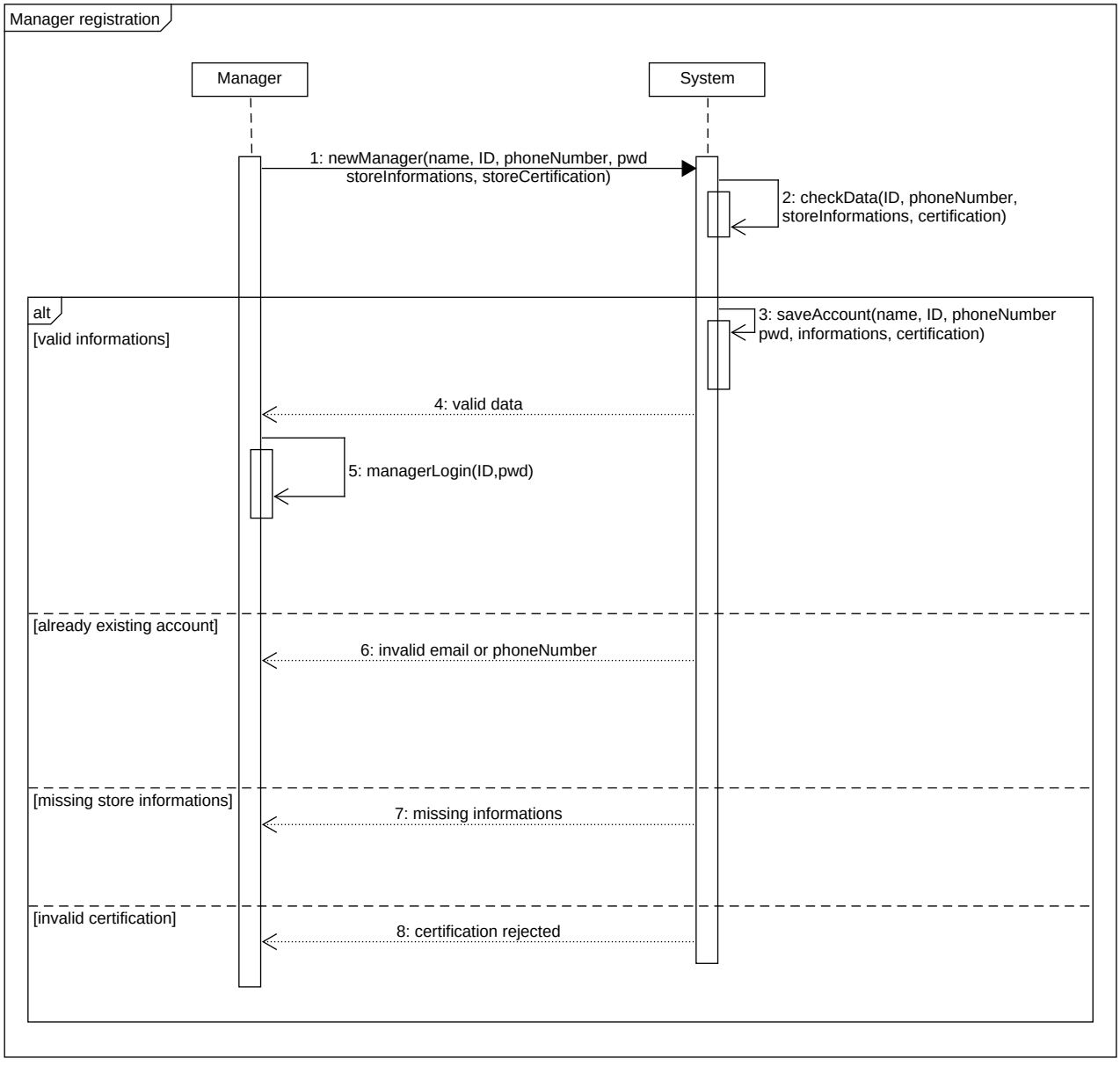


Figure 6: Sequence Diagram of Store Manager Registration

#### 3.2.3.3 Login of a customer

| | |
|---|---|
| Name | Login of a customer |
| Actors | Customer |
| Entry Condition | Customer is already registered to the application service |
| Event Flow | 1. Customer accesses the application through its device<br>2. Customer clicks on “Login as customer” button<br>3. The system opens the “Login as customer” page<br>4. Customer compiles the fields “Username” and “Password”<br>5. Customer clicks on “Login” button<br>6. The system opens the “Customer menu” page |
| Exit Conditions | Customer has successfully logged in |
| Exception | 1. Customer enters invalid email<br>2. Customer enters invalid password<br>If one or more of the above situations occur, the application will throw an error message and will return to the “Login as customer” page |

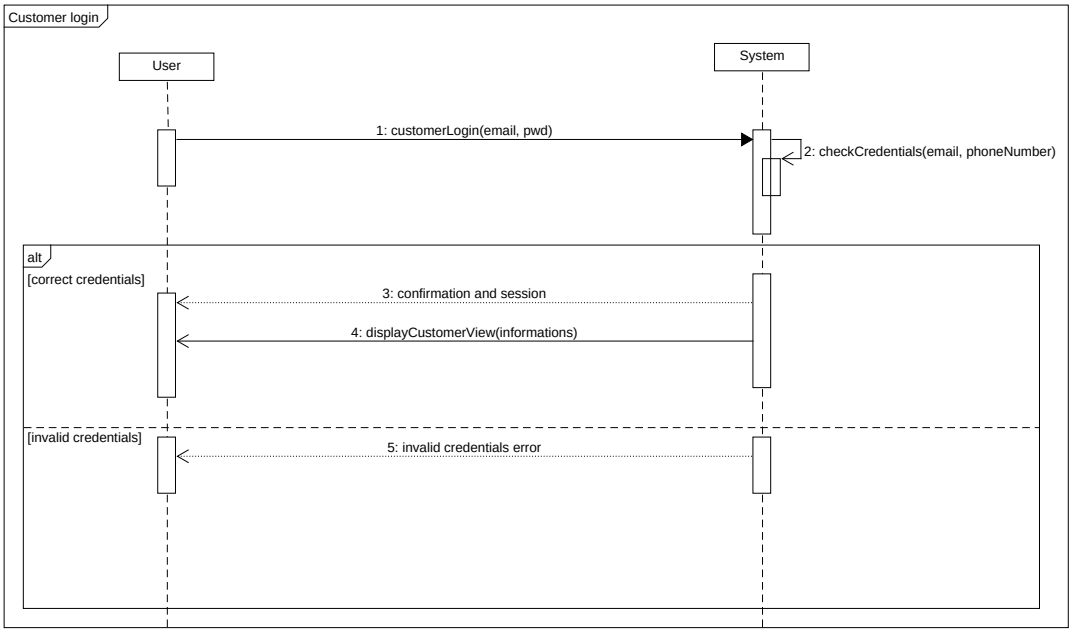


Figure 7: Sequence Diagram of Customer Login

**Required functional requirements:**

- **R3:** The system must allow customers to log in
- **R5:** The system must save the customers data registration
- **R8:** The system must guarantee that each customer e-mail is unique

#### 3.2.3.4 Login of a store manager

| | |
|---|---|
| Name | Login of a store manager |
| Actors | Store manager |
| Entry Condition | Store manager's store is already registered to the application service |
| Event Flow | 1. Store manager accesses the application through its device<br>2. Store manager clicks on "Login as store" button<br>3. The system opens the "Login as store" page<br>4. Store manager compiles the fields "ID" and "Password"<br>5. Store manager clicks on "Login" button<br>6. The system opens the "Store menu" page |
| Exit Conditions | Store manager has successfully logged in |
| Exception | 1. Store manager enters invalid ID<br>2. Store manager enters invalid Password<br>If one or more of the above situations occur, the application will throw an error message and will return to the "Login as store" page |

**Required functional requirements:**

- **R4:** The system must allow store managers to log in
- **R6:** The system must save the store managers data registration and store parameters
- **R7:** The system must guarantee that each store manager ID is unique

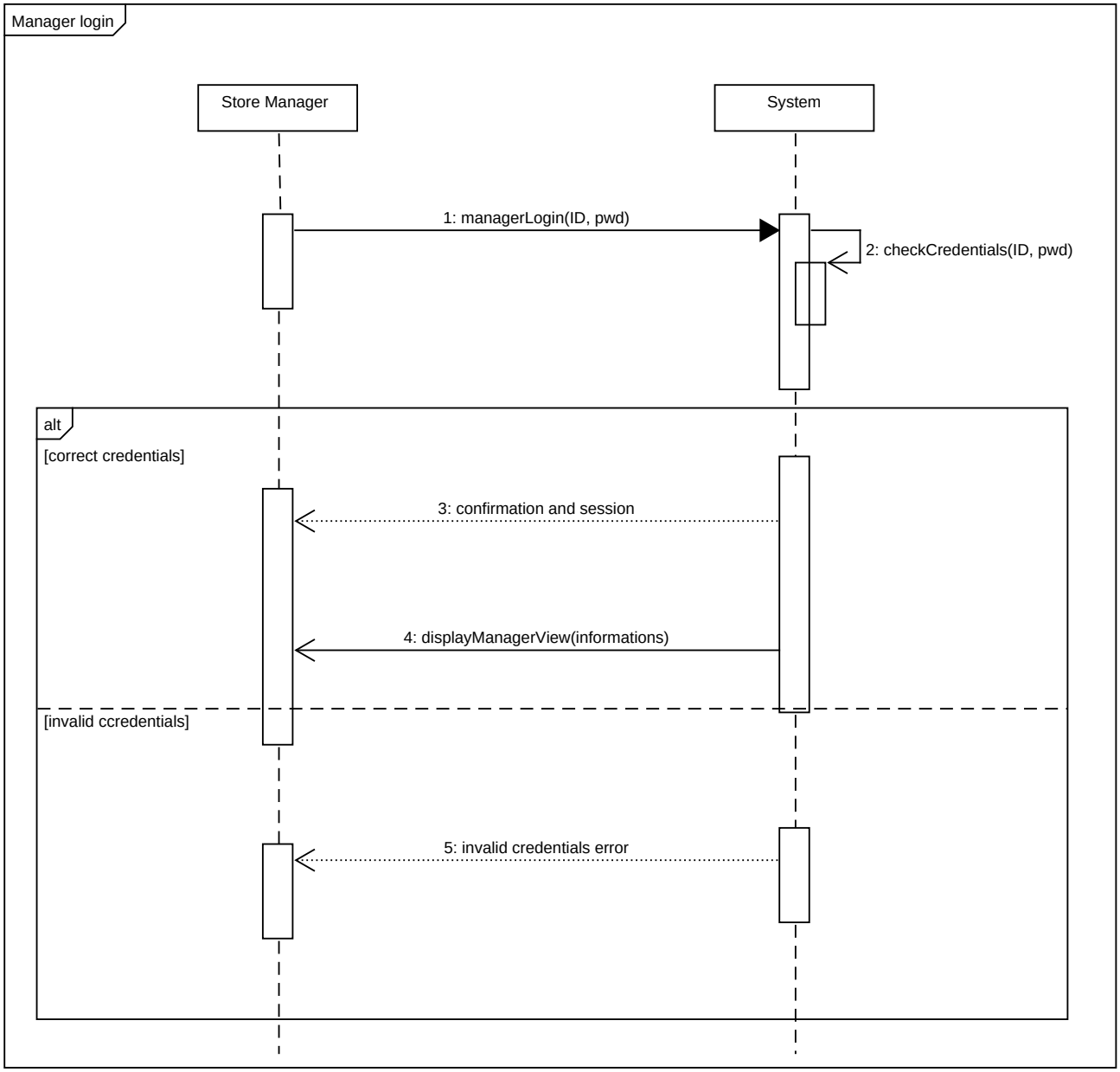


Figure 8: Sequence Diagram of Store Manager Login

#### 3.2.3.5 Customer makes a reservation

| | |
|---|---|
| Name | Customer makes a reservation |
| Actors | Customer |
| Entry Condition | Customer is already logged in the application service |
| Event Flow | 1. Customer clicks on "Make a reservation"<br>2. The customer can see the list of stores in his city and can filter this list by choosing a specific chain from a drop down menu, selects a store and clicks on Next<br>3. Customer can see the list of all possible objects' category and can select some of them (optional), and then clicks on Next<br>4. Customer can estimate a duration for his shopping, or let the system to do it, and then clicks on Next<br>5. Customer can see two button, "As soon as possible" and "Choose a time slot"<br>(a) if the clicks on "As soon as possible" button, the system will check if it's possible to assign a ticket with a reasonable waiting times in the day of the request, and by assuring it won't be over the store's closing time.<br>If a ticket with low waiting time is generable, the system will generate it calculating the time nedded to reach the store, saves it on both itself and user's app, showing it on the latter. otherwise, the system suggests him stores less crowded, if any. If so, user can choose between the options proposed, including staying at the selected store. Else, if the ticket is generable, it will be generated even if with a high waiting time.<br>(b) if the customer clicks on "Choose a time slot", the user can choose a time slot among those availables on the "Time slots" page to generate and save a ticket, calculating the time needed to reach the store. |
| Exit Conditions | Customer has successfully made a reservation |
| Exception | 1. Customer click on a timeslot no longer available or tries to book more than the set limit of reservations per week<br>The application will throw an error message and will reload the "Time slots" page (updating it)<br>2. Customer doesn't select any product category<br>The application will throw an error and ask to select at least some item.<br>3. Customer tries to get a ticket while he already have a unused one<br>The application shows an error message and invite to use the pending ticket<br>4. In the selected store where get a spot on the queue there isn't availability in the day, and there isn't any available near store<br>If the above situation occurs, the application will throw an error and leave the client the possibility to book his visit. |

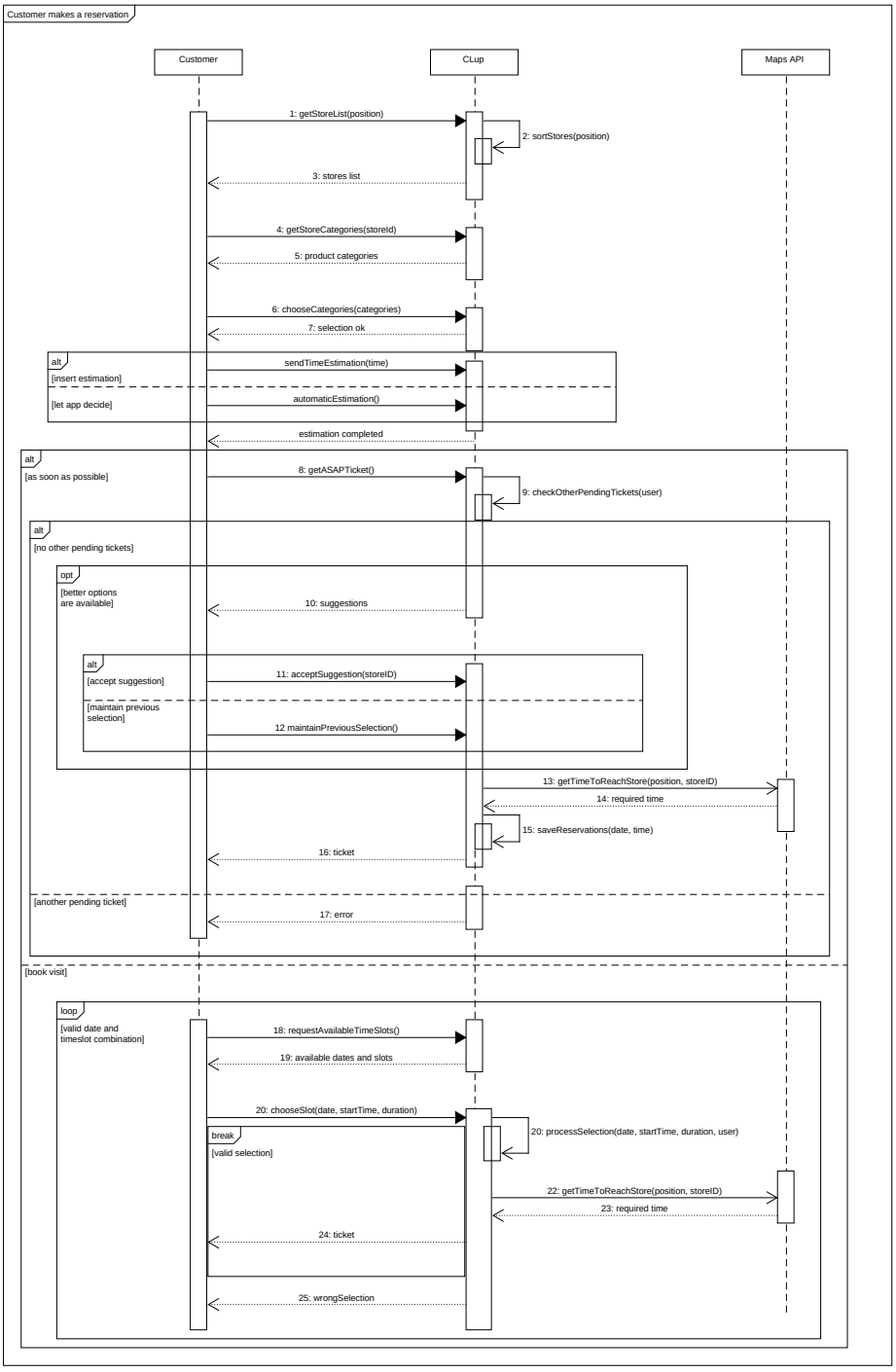


Figure 9: Sequence Diagram of Getting Ticket from App Procedure

**Required functional requirements:**

- **R8:** The system must guarantee that each customer e-mail is unique
- **R9:** The system must guarantee that each reservation ID is unique
- **R14:** The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
- **R15:** The system must show the customers of the time periods in which they can enter the store in order to respect the time selected by the customers
- **R16:** The system have to make a reasonable estimate of when a user with a spot on the queue is able to enter the store
- **R18:** The system is able to ask for the position of the customers
- **R26:** The system knows the situation in real time of each store
- **R27:** The system takes trace of each customer entry and exit from the store on the queue is able to enter the store
- **R28:** The system contains a list of bookable stores
- **R30:** The system can reasonably estimate the time needed from a specific user to complete his shopping
- **R31:** The system must save clients' tickets
- **R32:** The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store

#### 3.2.3.6 Customer visualizes reservations

| | |
|---|---|
| Name | Customer visualizes reservations |
| Actors | Customer |
| Entry Condition | Customer is already logged in the application service |
| Event Flow | 1. Customer clicks on "Show requests" button<br>2. The app show the list of bookings made and tickets requested<br>3. The customer select the desired option<br>4. The system opens the detail page of the selected option, that includes the "QR Code" and the number that should be called, among with the scheduled entry date and time and expected time when depart for the store. |
| Exit Conditions | Customer can visualize the reservation |
| Exception | 1. The client has not pending requests If the above situation happen, the app shows an error message and bring the client to the home page. |

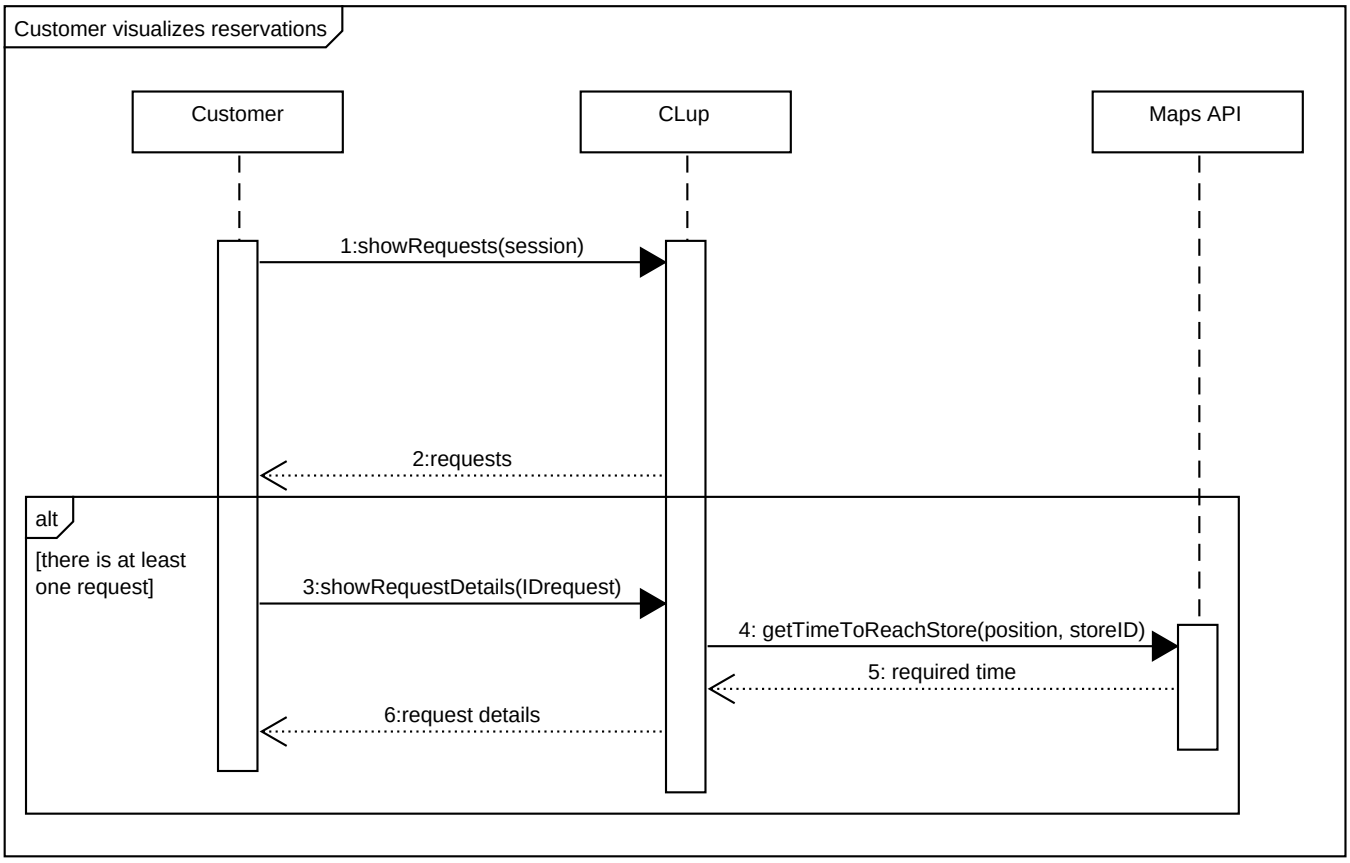


Figure 10: Sequence Diagram of Visualizing Customers' Reservations in app

**Required functional requirements:**

- **R10:** The system allows the customers to view their visits
- **R18:** The system is able to ask for the position of the customers
- **R31:** The system must save clients' tickets

#### 3.2.3.7 Manager modifies store parameters

| | |
|---|---|
| Name | Store manager modifies store parameters |
| Actors | Store manager |
| Entry Condition | Store manager is already logged in the application service |
| Event Flow | 1. Store manager clicks on "Modify parameters" button<br>2. Store manager can see both the parameters relatives to the whole store (eg. ID, Pwd, Store Capacity, Max Simultaneous Reservations, Opening and Closing Time), and each department, with their parameters (max capacity and simultaneous allowed bookings)<br>(a) Store manager modify one, more or none of the parameters of the store or of some departements (clicking on the pencil shaped button of the desidered department)<br>(b) Store manager choose to add/delete some department respectively clicking on "Add a department" and on the bin shaped button of the desidered department<br>(c) Store manager clicks on Save Changes<br>3. The system saves and processe the changes (notifying clients if necessary) and bring him back to "Store menu" page |
| Exit Conditions | Store manager has successfully updated store parameters |
| Exception | 1. Store managers enters an invalid value for one or more parameters<br>2. The store manager tries to add an already existing departement<br>3. The sum of the capacities of the single departments is greater than the store capacity<br>If the above situation occur, the application will throw an error message and will return to "Modify parameters" page |

**Required functional requirements:**

- **R6:** The system must save the store managers data registration and store pa- rameters
- **R9:** The system must guarantee that each reservation ID is unique
- **R19:** The system permits to store manager to modify the maximum capacity of the store and its departments
- **R21:** The system allows the manager to establish the maximum simultaneously allowed booked clients in the store (or in a specific department)
- **R17:** The system can send notification to the clients

- **R25:** The store manager can handle the opening and closing time of the store

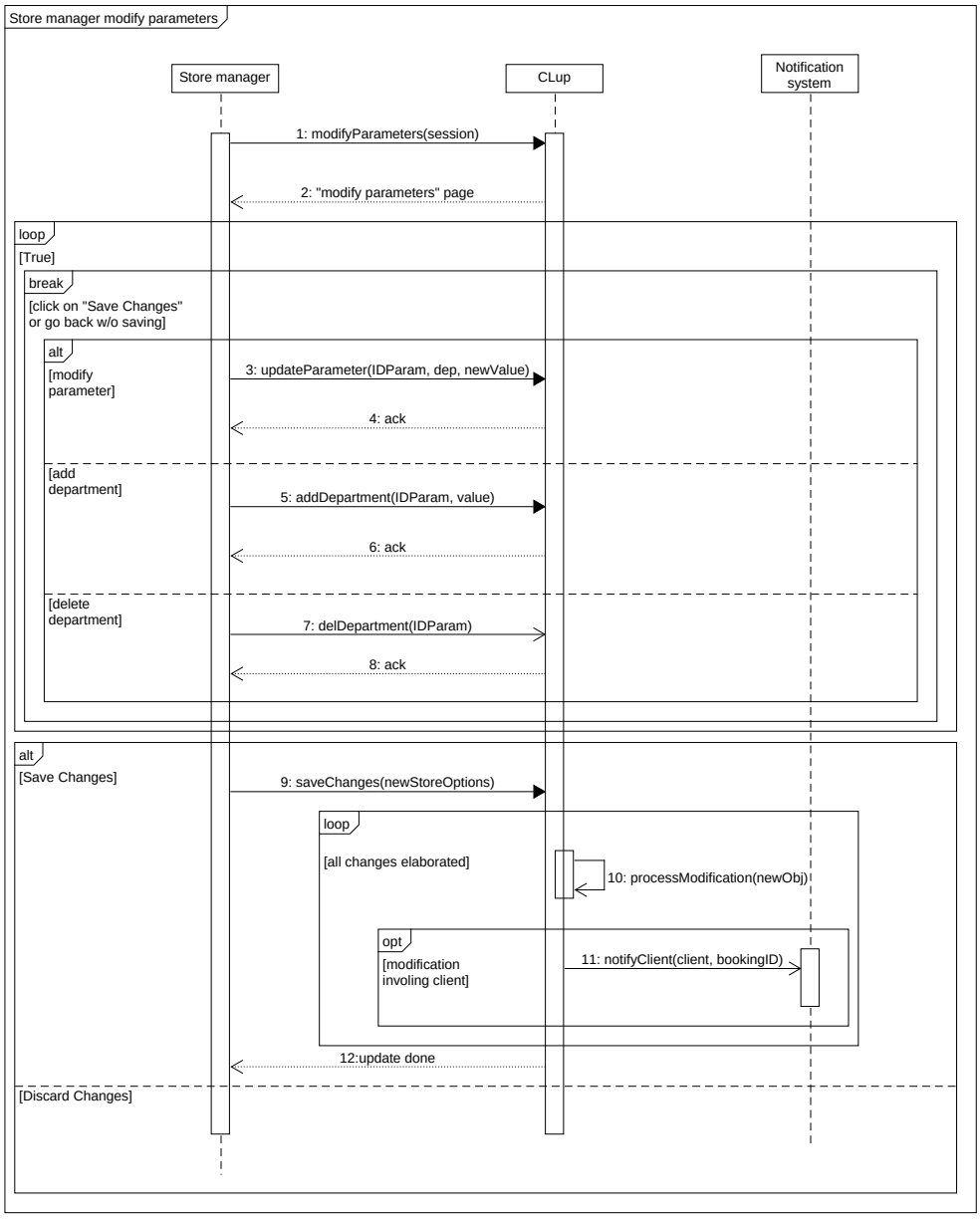


Figure 11: Sequence Diagram of Visualizing Customers' Reservations in app

#### 3.2.3.8 The store manager monitors the store situation

| | |
|---|---|
| Name | Store manager monitors store situation |
| Actors | Store manager |
| Entry Condition | Store manager has opened the app and is already logged in |
| Event Flow | 1. Manager clicks on "Monitor store" button<br>2. The app shows a page with statistics on the store, including number of people inside the store, percentage of store occupation and the number of daily access to the building<br>(a) If the manager wants, he can see the same statistics per store zone by clicking the "monitor zones" button. The system will show all the zones sorted by criticity |
| Exit Conditions | Store manager can see statistics on the store |
| Exception | None |

**Required functional requirements:**

- **R9:** The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
- **R17:** The system permits to store manager to view the store parameters
- **R24:** The system takes trace of each customer entry and exit from the store

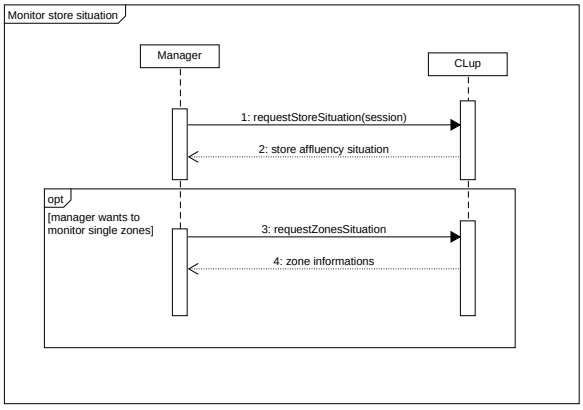


Figure 12: Sequence Diagram of Visualizing Store and Departments Situation

#### 3.2.3.9 The store manager manages customers bookings

| | |
|---|---|
| Name | The store manager manages customers bookings |
| Actors | Store manager |
| Entry Condition | Store manager has opened the app and is already logged in |
| Event Flow | 1. Manager clicks on "Manage bookings" button<br>2. The app shows a list of reservations, with some of their details in preview (such as chosen date and time slot)<br>3. The manager can choose one of them, and the app will show some possibilities to the manager<br>(a) The manager can press on the button "Contact client" to contact the client for some reasons.<br>i. The manager can choose to contact the client via mail clicking on the button "Email option"<br>ii. The app will show an interface where the manager can insert the text<br>iii. The manger clicks on the "Send" button to send the message<br>iv. The app returns to the previous page<br>(b) The manager can click on the button "Cancel booking" to cancel a reservation<br>i. The app will show a dialog box where the manager can put-in an optional message, explaining the reasons of the cancel<br>ii. The manager clicks on the "Delete button"<br>iii. The app will show a confirmation box to ask if the manager is sure to proceed<br>iv. The manager clicks on "Yes" to confirm the deletion<br>v. The app closes the dialog box<br>(c) The manager can choose to reschedule a booking<br>i. The app will show a dialog box where the manager can choose a new time slot and insert a message explaining the reasons<br>ii. The manager clicks on "Modify" to modify the booking<br>iii. The app closes the dialog box |
| Exit Conditions | Store manager can manage the store and is able to edit reservations |
| Exception | None |

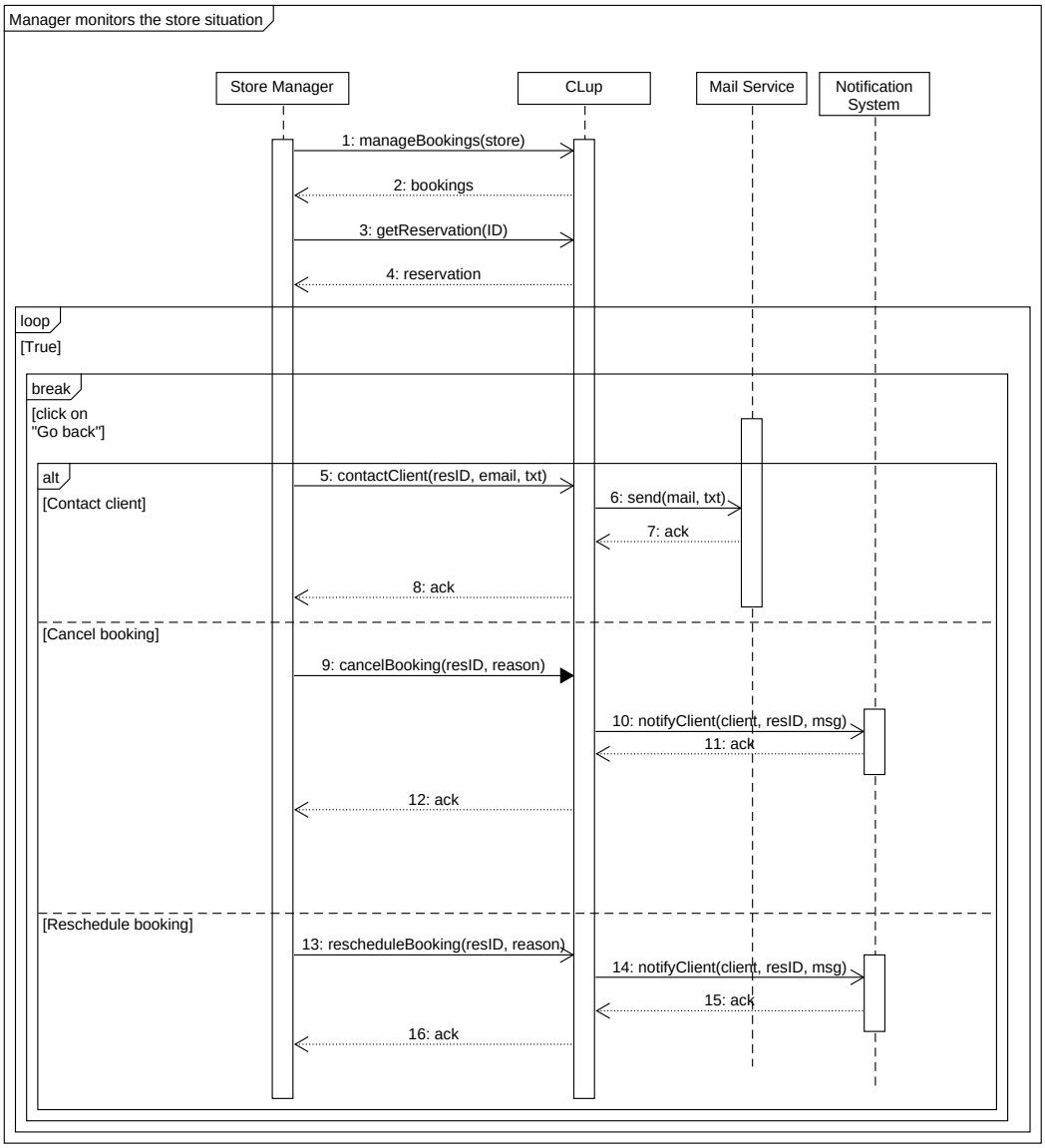


Figure 13: Sequence Diagram of Visualizing Store and Departments Situation

**Required functional requirements:**

- **R19:** The store manager can view the reservation of each client
- **R20:** The store manager can modify the reservation of each client

- **R21:** The store manager can cancel the reservation of each client

#### 3.2.3.10 Customers reservations management

| | |
|---|---|
| Name | Customers reservations management |
| Actors | Customer |
| Entry Condition | The customer has the application opened, is logged in and has at least one pending request |
| Event Flow | 1. The user press on the "Show requests" button<br>2. The app shows a page with the requests made by the client<br>3. The user selects the desidered requests<br>4. The app will show the requests details<br>5. The user press the "edit" button<br>(a) If the selected requests is a ticket, the customer can delete it pressing the "Delete button"<br>i. The system will show a confirmation dialogue<br>ii. The client press the yes button<br>iii. The ticket is deleted and the app will return to the previous screen if there are other requests, or to the main menu otherwise<br>(b) If the selected request is a booking, the client can both click the delete button, and follow the above procedure or can select the "modify button"<br>i. If the modify button is selected, the app will restart the "Make a reservation process" described in Use Case n. 5. The customer can modify all the parameters of his reservation, including transforming it in a "As soon as possible ticket |
| Exit Conditions | The client can modify his reservation |
| Exception | None |

**Required functional requirements:**

- **R5:** The system allows the customers to view their visits
- **R6:** The system allows the customers to cancel their visits
- **R7:** The system allows the customers to modify their visits
- **R8:** The system allows the customers to select their favorite means of trans- portation
- **R9:** The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
- **R11:** The system must consider the extimate shopping time insert by customers
- **R12:** The system must show the customers of the time periods in which they can enter the store in order to respect the time selected by the customers
- **R13:** The system have to make a reasonable estimate of when a user with a spot on the queue is able to enter the store
- **R15:** The system is able to ask for the position of the customers
- **R23:** The system knows the situation in real time of each store
- **R27:** The system can reasonably estimate the time needed from a specific user to complete his shopping
- **R28:** The system must save clients' tickets
- **R29:** The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store to complete his shopping

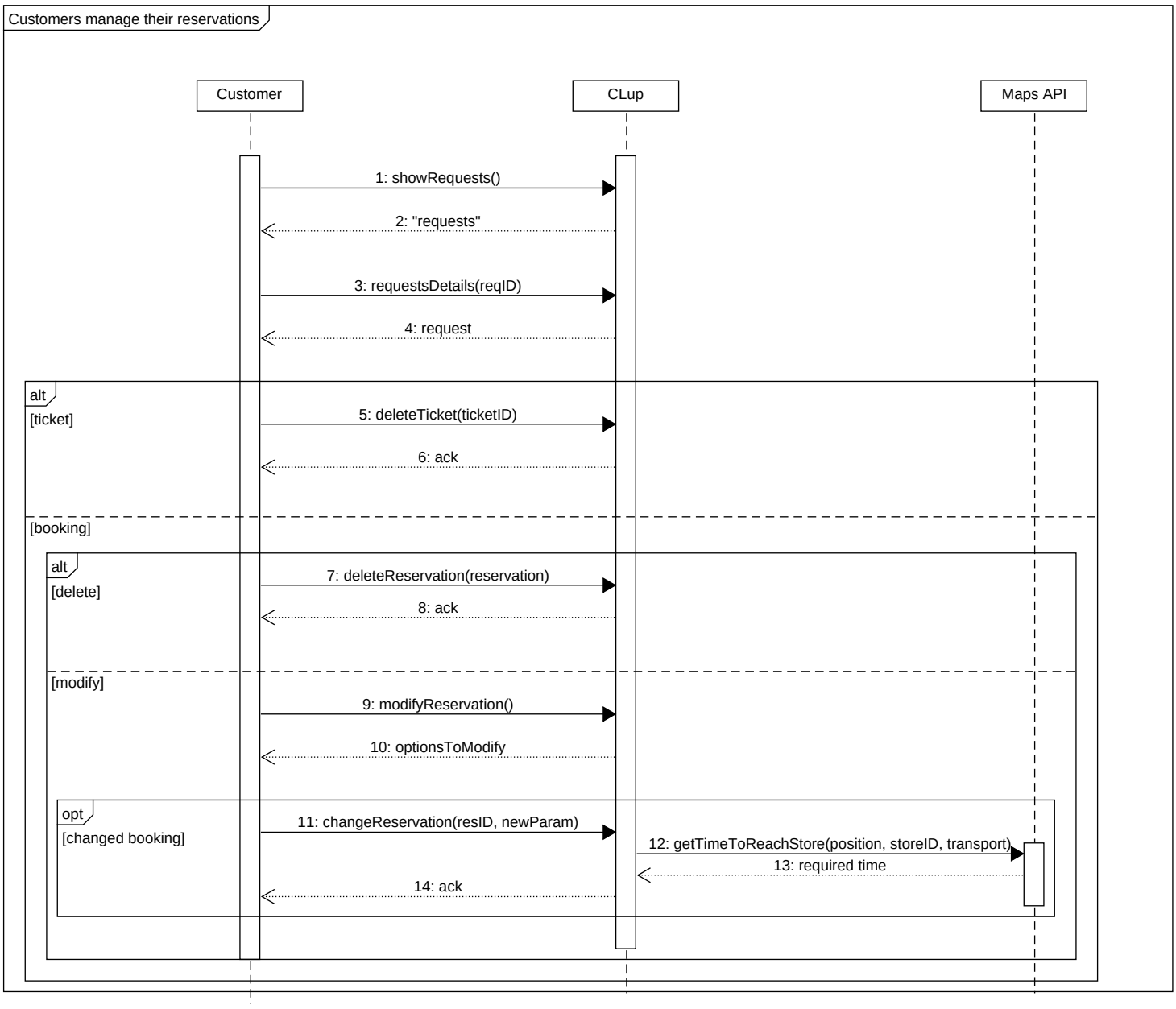


Figure 14: Sequence Diagram of managing reservations customer side

#### 3.2.3.11 Customer get a ticket with the totem

| | |
|---|---|
| Name | Customer get a ticket with the totem |
| Actors | Customer |
| Entry Condition | The customer is at the store and is using the totem |
| Event Flow | 1. Customer clicks the button "Get Ticket"<br>2. Customers can see the list of all possible objects' categories and can select some of them (optional)<br>3. The customer enters the estimation of shopping time, or let the system to infer it<br>4. Client confirms the options selected and the system generates and prints the associated number, the *QR Code* and the *ETA* |
| Exit Conditions | The client gets a ticket |
| Exception | 1. Client has not completed the operation in the prefixed time<br>If the above situation occur, the application will throw an error message and will return to "Home" page |

**Required functional requirements:**

- **R9:** The system allows the customers to select some or all the departments in which the customers are interested in doing shopping
- **R11:** The system must consider the extimate shopping time insert by customers
- **R26:** The system is able to print a paper ticket
- **R28:** The system must save clients' tickets
- **R29:** The system should estimate when it must stop generating other tickets to avoid turns after the closing time of the store

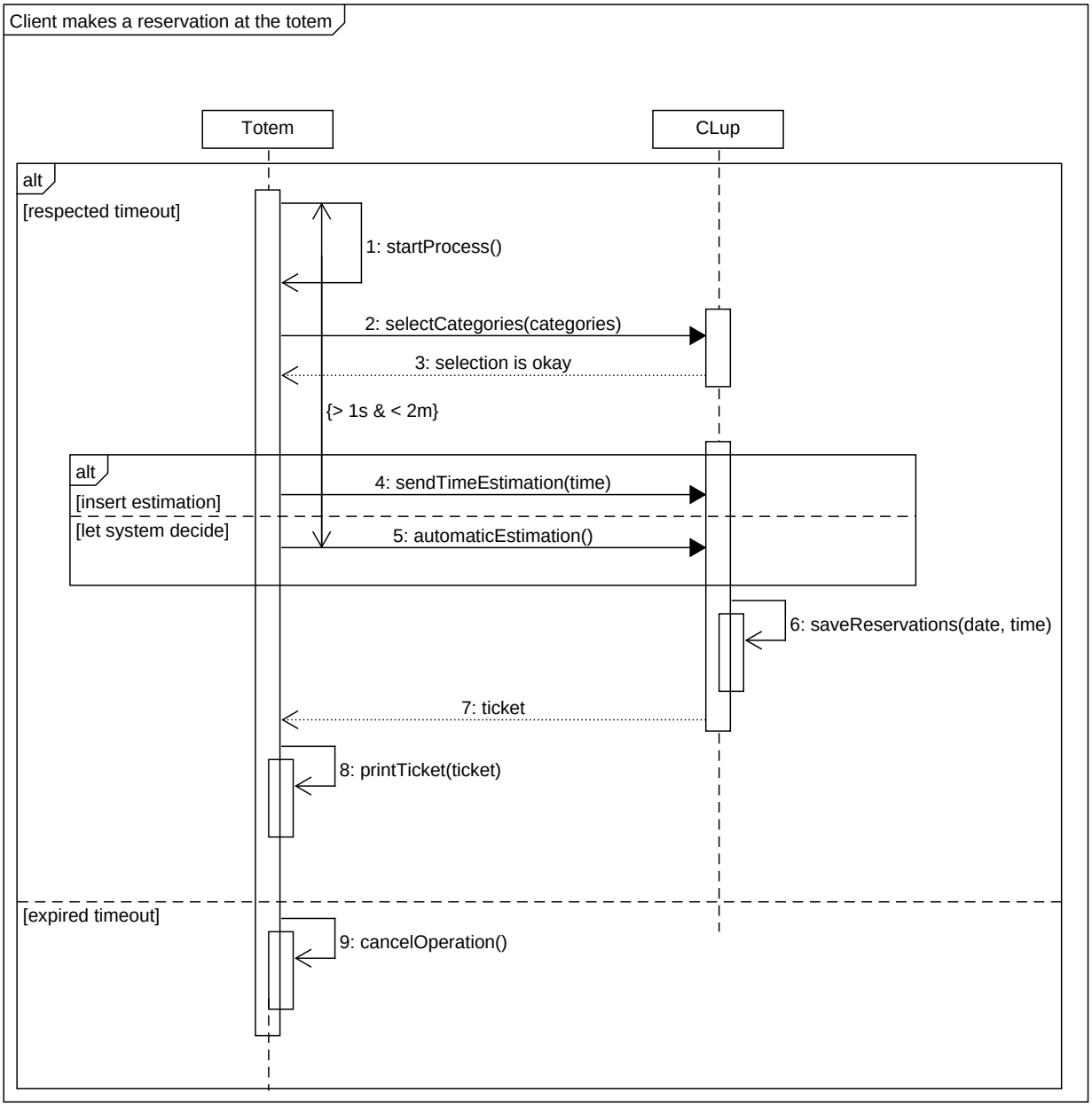


Figure 15: Sequence Diagram of getting a ticket at a physical dispenser

#### 3.2.3.12 Customer selects the prefered means of transport

| | |
|---|---|
| Name | Customer selects the prefered means of transport |
| Actors | Customer |
| Entry Condition | Customer is already logged in the application service |
| Event Flow | 1. Customer clicks the "Means of transport" button<br>2. Customer can see a list of means of transport<br>3. Customer select his preferred means of transport<br>4. The system gets his *GPS* positions and checks if the selected option is available.<br>5. The app saves the option and updates the notifies alerting customers when they should depart for the store. |
| Exit Conditions | The client selects the preferred mean of transport |
| Exception | 1. The selected means of transport is not available<br>If the above situation occur, the application will throw an error message and will ask the client to redo the selection |

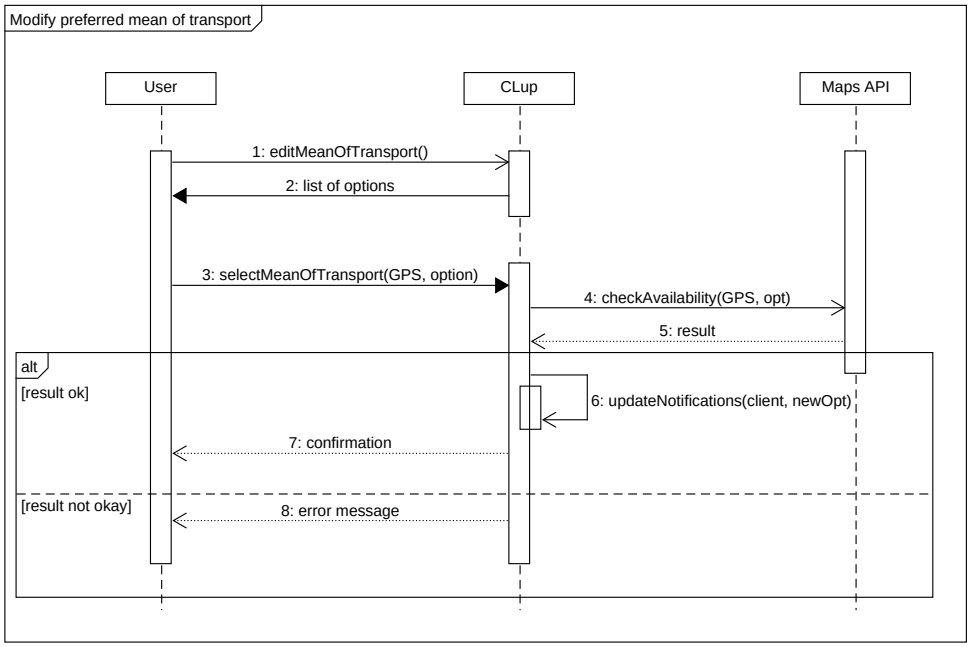


Figure 16: Sequence Diagram of changing preferred mean of transport

**Required functional requirements:**

- **R8:** The system allows the customers to select their favorite means of trans- portation
- **R15:** The system is able to ask for the position of the customers

## 3.3 Use Case Diagram

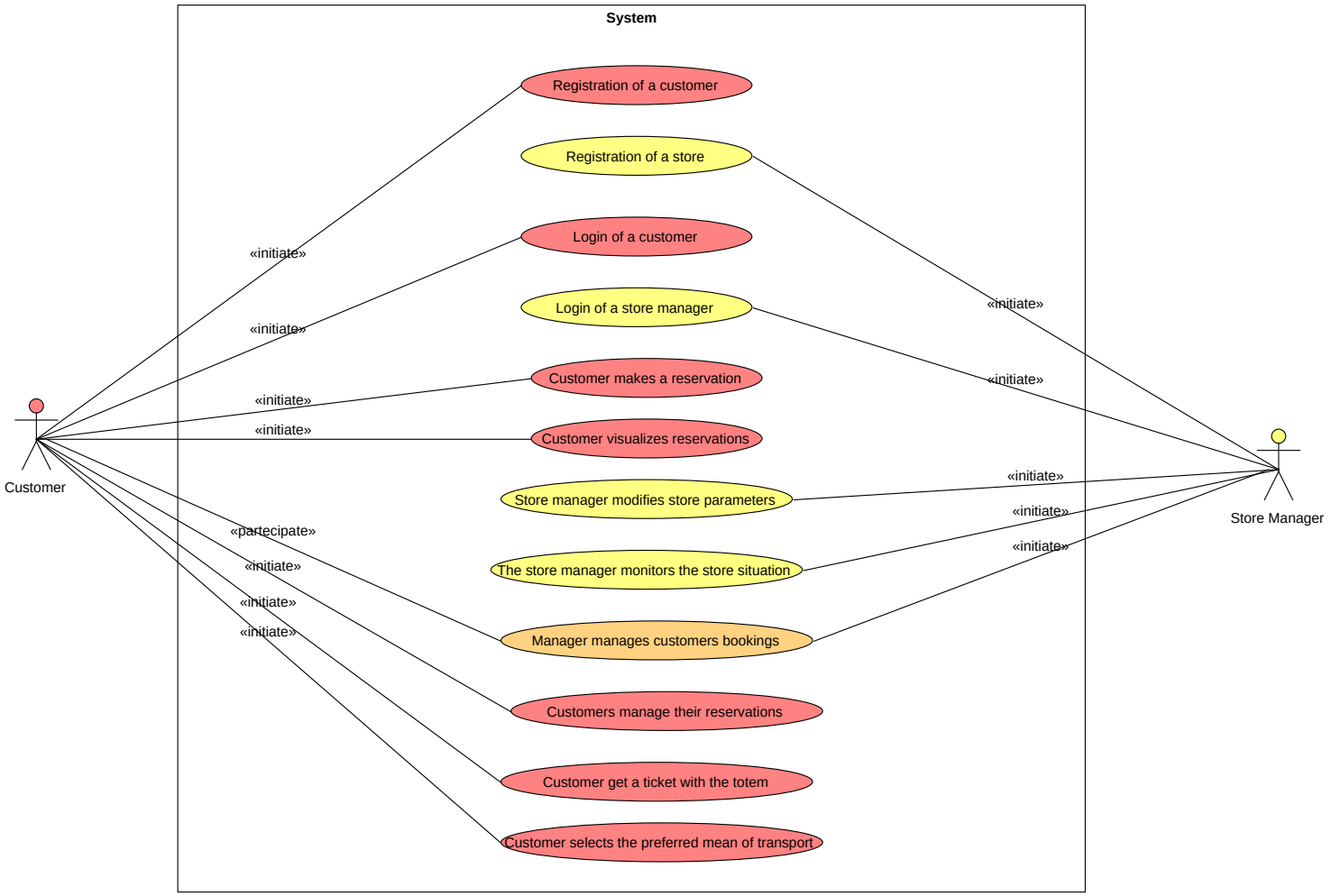


Figure 17: Use Case Diagram

### 3.3.1 Scenarios

**Scenario 1: User without application**
Marta is a university student unfavorable to consumerism and for this reason she does not have a smartphone. She go to her favorite store to do shopping and thanks to our system she can still book a place in the queue to enter in the store. To do this, she just go to the totem placed near the supermarket and fill in the reservation with the requested data, that is the departments she intends to visit and an estimate of the time spent inside the store. In this way, the system will queue her to enter as soon as possible, providing her with an estimated entry time. Knowing this, Marta can still do other activities before being called to enter the store, so as not to create crowds at the store exit. As with the app, Marta has to scan the ticket at the entrance and exit. In this way the system can manage the queue and reservations.

**Scenario 2: Fake QR code**
Jonathan arrives at the store and notices that he has many people in line before him. Jonathan is not a very patient guy, so having kept another *QR Code*, he tries to skip the line trying to scan it. The system recognizes that the *QR Code* is not valid, therefore it does not allow Jonathan to enter the store, forcing him to respect the queue.

**Scenario 3: User books from the application**
Adalgiso must go shopping but does not want to wait a long time outside the store and wants to be sure that he can enter the moment he arrives. So, he opens CLup on his smartphone, sets up a preferred means of transport and begins a reservation. He Selects the store, the departments he wants to visit and the estimated time. Once the booking is complete, the app will send the notification to Adalgiso, inviting him to depart from his position to go to the store in time. When he will arrive, his number will be called shortly.

## 3.4 Performance Requirements

CLup app is aimed to reduce gatherings due to the Covid-19 pandemic. So, some components needs a certain responsiveness. Here there are described some parts of the system that are critical to the scope of entire system

- **Requests saving system:** when a client got a ticket, the system needs to process and save it in a very low time, less than 2 seconds, since the requested tickets in a day are really important to allow or deny people to get another ticket for the same day/book a visit in a certain moment of the day.
- **QR Code processing:** each customer must scan his QR Code when he enters the store. This means that, to avoid delays in the queue, the component processing QR Codes must be really fast, and process them in no more than 5 seconds after the code is retrieved from the optical scanner.
- **Ticket calling system:** as for QR Codes processing, after a customer exits the store, the system must be able to process the next ticket to be called, if any, in a strict time, to avoid that people waits outside the store for long times. This process must not last more than 10 seconds.
- **User notifier:** the system must responsively notify users that they must depart for the store in time.
- **Application data updates:** when the customer need to use the application for any purpose, the app must be responsive, and each object required over the network must be available in no more than 5 seconds from its request; if the retrieve from the net fails, the app will show the local stored data to avoid delays in the whole system (eg. the client must retrieve his QR Code to access the store)
- **Totem ticket printing:** the totem must process each request and print the ticket in no more than 5 seconds.

The above reported performance constraints must be always verified; that implies the system must be highly scalable to respect this timing. There isn't any limit on the number of registered stores and users, so the system must respect these constraints with at least 200 connected users per each store inside the system.

## 3.5 Design Costraints

### 3.5.1 Standards Compliance

System is compliant to some standards, such as:

- Generated QR Codes are compliant to ISO/IEC 18004:2015 standard

- Network messages are exchanged through the internet protocol TCP/IP

The same doesn't applies to requests of travel times, since the uses non-standard API of some map service.

### 3.5.2 Hardware Limitations

To work properly, CLup needs some type of hardware, in base of the considered component.

- **Customer Device:** Customer needs to have a device with a working network module, a GPS module and a high resolution display (minimum 720P).
- **Totem:** the totem must have a working network module, a ticket printer and a module to make possible interfacing between customer and totem
- **QR Code readers:** QR Code readers must have a working network module, and a optical scanner that takes no more than 5 seconds to read and interpret a QR code and its content

## 3.6 Software System Attributes

### 3.6.1 Reliability

The system must have a very low probability of failure (less than 0,0001%) to avoid inconveniences in crowd managing.

### 3.6.2 Availability

Due to the critic aspects the application handles, is required a 99,99% of system reliability. It means that the application will be down only for 52.60 minutes per year, an estimate that is acceptable. Having a greater downtime per year, may lead to inconveniences in handling the queue outside a store, bringing to a possible assembly of people.

### 3.6.3 Security

Each communication between client and server is made over a secure transport protocol (eg. TLS). For each request the system will authenticate the user so that everyone access only to data he is authorized. An encryption and decryption system must be implemented so that QRs' Content is encrypted to avoid someone can forge a malicious ticket. Moreover, each user password is stored using its hash value.

### 3.6.4 Maintainability

Since rules can change in every moment, the software must be developed in an extendible way, so that it'e easy to add new functions required by new restrictions. Moreover, the software can be used even if after the pandemic. So, it might be required to implement new feature to make the system more appealing.

### 3.6.5 Portability

The application must be portable on the majority mobile OSes. For this reasons, the app will use a notification service accessible through the OS APIs. Furthermore, the software is thought so that is possible to implement different map service in different versions.

## 3.7 Additional Specifications

The system guarantees unicity of each customer's email and of each store manager's ID.

# 4 Formal Analysis Using Alloy

The following section considers the fundamental properties and constraints identified for the specification of the problem and formally define them with a formal model in which it is shown how they will be satisfied. The alloy modelling language is used to model the problem, and some possible comments are also provided in order to clarify the most critical aspects.

## 4.1 Alloy Model

```
sig Username{
    usr: one String
    }
    sig Password{
    psw: one String
    }
    sig Email{
    email: one String
    }
    sig Name{
    name: one String
    }
    sig Surname{
    surname: one String
    }
    sig Certification{}
    sig CustomerReg{
    username: one Username,
    password: one Password,
    email: one Email,
    name: one Name,
    surname: one Surname
    }
    sig Customer{
    registration: one CustomerReg
    }
    sig StoreReg{
    name: one Name,
    password: one Password,
    certification: one Certification,
    capacity: one Int
    }
    sig Store{
    registration: one StoreReg
    }
```

# 5 Effort Spent

# 6 References