POLITECNICO
MILANO 1863

Software Engineering II

# CLup - Customers Line-up

## Requirements Analysis and Specification Document

*Authors:*
*Cosimo Sguanci*
*Roberto Spatafora*
*Andrea Vergani*

**1 Introduction** **4**
- 1.A Purpose 4
- 1.B Scope 5
- 1.C Definitions, Acronyms, Abbreviations 6
  - 1.C.1 Definitions 6
  - 1.C.2 Acronyms 6
  - 1.C.3 Abbreviations 7
- 1.D Revision history 7
- 1.E Reference Documents 7
- 1.F Document structure 8

**2 Overall description** **9**
- 2.A Product perspective 9
  - 2.A.1 Further details on the Shared Phenomena 9
  - 2.A.2 UML Class Diagram 11
  - 2.A.3 UML Statechart Diagrams 12
  - 2.A.4 Scenarios 13
- 2.B Product functions 16
- 2.C User characteristics 19
- 2.D Assumptions, dependencies and constraints 20
  - 2.D.1 Domain assumptions 20
  - 2.D.2 Dependencies 20

**3 Specific Requirements** **21**
- 3.A External Interface Requirements 21
  - 3.A.1 User Interfaces 21
    - 3.A.1.1 CLup mobile application interface 21
    - 3.A.1.2 CLup ticket machine interface 23
  - 3.A.2 Hardware Interfaces 25
  - 3.A.3 Software Interfaces 26
  - 3.A.4 Communication Interfaces 26
- 3.B Functional Requirements 26
  - 3.B.1 Use case diagram 26
  - 3.B.2 Use cases 28
  - 3.B.3 Sequence diagrams 46
  - 3.B.4 Requirements 46
  - 3.B.5 Use cases mapping on requirements 48
  - 3.B.6 Goals mapping on requirements and domain assumptions 53
- 3.C Performance Requirements 55
- 3.D Design Constraints 55
  - 3.D.1 Standards compliance 55
  - 3.D.2 Hardware limitations 55
- 3.E Software System Attributes 55
  - 3.E.1 Reliability 55

3.E.2 Availability 55
3.E.3 Security 56
3.E.4 Maintainability 56
3.E.5 Portability 56

**4 Formal Analysis using Alloy 57**

4.A Main objectives 57
4.B Signatures 57
4.C Functions 58
4.D Facts 59

# 1 Introduction

## 1.A Purpose

*Customers Line-up* is an application that allows users to make reservations for visiting a grocery store.
The idea arises in a context of sanitary emergency, in which people experience a lockdown situation and should be as safe as possible, in order to prevent the evolution of a pandemic and all its consequences on society. Of course, grocery shopping is an essential need, but all the activities connected to it must be highly regulated, so that crowds are avoided and safety is guaranteed.
A typical rule for supermarkets, in an epidemic situation, is to restrict access, in order for people to keep enough distance while doing the shopping. However, the immediate consequence of this measure is physical lining up, which is in turn a problem: crowds form and social distancing can become far from reality.
*Customers Line-up* is thought for avoiding this kind of situation, enabling a way to queue up virtually and prevent any sanitary risk: the influx of people inside the store is regulated, customers have interests in reserving a ticket (to enter) from their cars or homes, and rules to contrast the pandemic are respected on all sides.
To resume, the goals of the software system are those of granting social distance outside supermarkets, managing entrances and avoiding crows inside them. A list of the application's goals is presented here.

| **GOALS** | |
|---|---|
| **G1** | Grant social distance outside the grocery store<br><br>**G1.1:** Avoid physical lining up outside the grocery store * |
| **G2** | Manage entrances in the grocery store |
| **G3** | Avoid crowds (too many people) inside the grocery store (at the same time) |

** G1's subgoal; from now on, references to this subgoal will be made using **G1.1** abbreviation*

## 1.B Scope

According to Michael Anthony Zackson and Pamela Zave's standard model for requirements engineering, this section proposes an overview of World and Shared phenomena connected to the environment where *Customers Line-up* is thought to work.

| **WORLD PHENOMENA** | |
|---|---|
| **WP1** | A customer wants/needs to go to the grocery shop |
| **WP2** | A customer gets to (or approaches) the grocery shop by car/by bike/on foot/any other means of transportation |
| **WP3** | A customer inside the grocery shop decides to buy an item |
| **WP4** | A customer does the grocery shopping in a particular order |
| **WP5** | A customer pays at the check-out |
| **WP6** | |

| **SHARED PHENOMENA - WORLD CONTROLLED** | |
|---|---|
| **SPW1** | A customer gets a ticket |
| **SPW2** | A customer deletes his booked ticket |
| **SPW3** | A customer books a visit |
| **SPW4** | A customer deletes his booked visit |
| **SWP5** | A customer exits from the grocery shop |
| **SWP6** | A time slot is available for a reservation |
| **SPW7** | A customer gets a ticket/books a visit, but does not go to the grocery shop |
| **SPW8** | A customer gets a ticket/books a visit, but arrives at the grocery shop too late (after his turn and the associated time limit to enter) |

| SHARED PHENOMENA - MACHINE CONTROLLED | |
|---|---|
| **SPM1** | A customer enters in the grocery shop |
| **SPM2** | A customer is notified for entrance |
| **SPM3** | A customer is notified to leave from home |
| **SPM4** | A customer receives a notification suggesting to book a visit in his habitual grocery shop, day of the week and hour |
| **SPM5** | After a failure in the attempt to reserve a visit (due to time slots unavailability), a customer receives a suggestion to book in a different grocery shop |
| **SPM6** | After a failure in the attempt to reserve a visit (due to time slots unavailability), a customer receives a suggestion to book in a different date/time and same grocery shop |

# 1.C Definitions, Acronyms, Abbreviations

## 1.C.1 Definitions

- **Customer:** a person who does/is going to do the grocery shopping
- **Ticket machine:** a machine equipped with a touchscreen display, a printer system, a QR code reader and an *ad-hoc* version of *Customers Line-up* application
- **User:** a person who has downloaded *Customers Line-up* mobile application on his smartphone/tablet and has successfully logged in **OR** a person who uses *Customers Line-up* services through a ticket machine **OR** a person who uses *Customers Line-up* services by calling the call center

## 1.C.2 Acronyms

- **QR:** *Quick Response*
- **GPS:** *Global Positioning System*
- **RASD:** *Requirements Analysis and Specification Document*
- **CLup:** *Customers Line-up*
- **IVR:** *Interactive Voice Response*
- **SMS:** *Short Message Service*
- **DB:** *Database*
- **SIM:** *Subscriber Identity Module*
- **UML:** *Unified Modeling Language*

### 1.C.3 Abbreviations

| | | |
|---|---|---|
| **Gn** | Goal number n | *Defined in section 1.A* |
| **WPn** | World phenomena number n | *Defined in section 1.B* |
| **SPWn** | Shared phenomena (World controlled) number n | *Defined in section 1.B* |
| **SPMn** | Shared phenomena (Machine controlled) number n | *Defined in section 1.B* |
| **DAn** | Domain assumption number n | *Defined in section 2.D.1* |
| **UCn** | Use case number n | *Defined in section 3.B.2* |
| **Rn** | Requirement number n | *Defined in section 3.B.4* |

## 1.D Revision history

| Version | Date | Authors | Summary |
|---|---|---|---|
| 1.0 | | Cosimo Sguanci, Roberto Spatafora, Andrea Mario Vergani | First release |

## 1.E Reference Documents

- Software Engineering 2 slides (available on the Beep page of the course)
- Project assignment document (“R&DD Assignment A.Y. 2020-2021.pdf” available on the Beep page of the course)
- RASDs developed by colleagues of past years (available on the Beep page of the course or on GitHub)

## 1.F Document structure

- **Section 1** provides an overview of *Customers Line-up*'s goals and the context in which it is thought to work. In addition, all released versions of this document are summarized in an appropriate paragraph.
- **Section 2** …
- **Section 3** …
- **Section 4** …
- **Section 5** summarizes the total effort spent for realizing the *Requirements Analysis and Specification Document* by each group member.
- **Section 6** lists all references that helped the team during analysis and document writing.

# 2 Overall description

## 2.A Product perspective

### 2.A.1 Further details on the Shared Phenomena

**Shared phenomena controlled by the World and observed by the Machine**

- **SPW1:** A customer gets a ticket
  A customer can acquire a ticket reservation by using one of *CLup*'s application interfaces; a ticket corresponds to virtual lining up and is associated with a code for entering the selected grocery shop. After getting a ticket, the user waits until his turn comes.
- **SPW2:** A customer deletes his booked ticket
  A customer can delete a ticket reservation by using one of *CLup*'s application interfaces. When a ticket is deleted, the user implicitly leaves the waiting "queue" which regulates entrances in the grocery shop.
- **SPW3:** A customer books a visit
  A customer can book a visit reservation by using one of *CLup*'s application interfaces; a visit is associated with a grocery shop, date, hour and a code for entrance. A visit can be reserved either for the current day or in advance.
- **SPW4:** A customer deletes his booked visit
  A customer can delete a visit reservation by using one of *CLup*'s application interfaces. When a visit is deleted, the user frees the time slot associated with it.
- **SPW5:** A customer exits from the grocery shop
  When a customer exits from a grocery shop, he has to show the code associated with his reservation to an appropriate system: in this way, *Customer Line-up* can know the actual number of people inside the supermarket (and other information) and has the possibility to manage the influx of new people.
- **SPW6:** A time slot is available for a reservation
  When a time slot is available, every customer using *CLup* services can book it for having access to the corresponding grocery shop. Of course, time slots availability is associated with already registered reservations.
- **SPW7:** A customer gets a ticket/books a visit, but does not go to the grocery shop
  A customer might, for many reasons, not respect a reservation: this is the case of a person who completely forgets about his booked visit, or does not delete it (after deciding not to go to do the shopping). *Customers Line-up* in a sense tries to prevent these situations, and on the other hand reacts in an appropriate way when it faces them.

- **SPW8:** A customer gets a ticket/books a visit, but arrives at the grocery shop too late (after his turn and the associated time limit to enter)
  A customer might be late for many reasons: wrong time estimation to get to the grocery shop, other commitments, ...
  *Customers Line-up* in a sense tries to prevent these situations (making it easy to arrive on time), and on the other hand reacts in an appropriate way when it faces them.

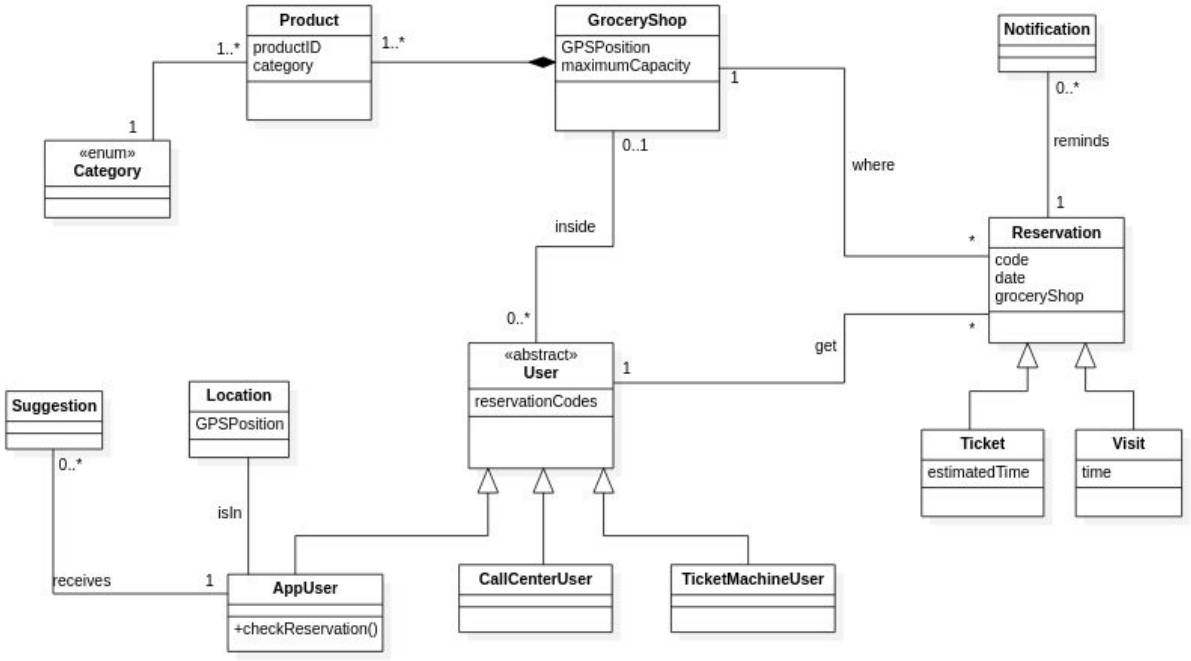
**Shared phenomena controlled by the Machine and observed by the World**

- **SPM1:** A customer enters in the grocery shop
  A customer can access the grocery shop only if he has a reservation. In practice, the user waits in the supermarket's proximity until his turn comes; then, he can enter only by showing (or typing) the code associated with his visit/ticket at the shop's entrance. Once in, the person can do the shopping.
- **SPM2:** A customer is notified for entrance
  *CLup* mobile application users, waiting for their turn outside the supermarket, receive a notification (on the app and via SMS) when they are allowed to enter. This system is very useful in order to guarantee social distance, permitting people to wait where they prefer: in fact, every communication message can arrive through *Customers Line-up* mobile application.
- **SPM3:** A customer is notified to leave from home
  *CLup* mobile application users, who have already booked a reservation for the current day, receive a notification (on the app) when it is time to leave from the place they are in order to reach the supermarket (on time). In this case, information about the customer's position is known thanks to GPS.
- **SPM4:** A customer receives a notification suggesting to book a visit in his habitual grocery shop, day of the week and hour
  *CLup* mobile application users might receive notifications (on the app) when an appealing time slot is available. In practice, the system tries to infer customer's habits and proposes ideal options to do the grocery shopping; most likely days, hours and supermarkets for every user are derived from customized statistics.
- **SPM5:** After a failure in the attempt to reserve a visit (due to time slots unavailability), a customer receives a suggestion to book in a different grocery shop
  Due to general restrictions and imposed maximum number of people inside every grocery shop, it might be the case that many time slots for reservation are unavailable (already booked). Of course, this might be stressing for a customer, because finding an option for grocery shopping can be harder than expected. *Customers Line-up* tries to help mobile app users, by proposing alternatives when their first choice for a visit is not available: a

proposal consists in a different shop (close to the selected one), in the same day and (approximately) hour chosen by the customer.

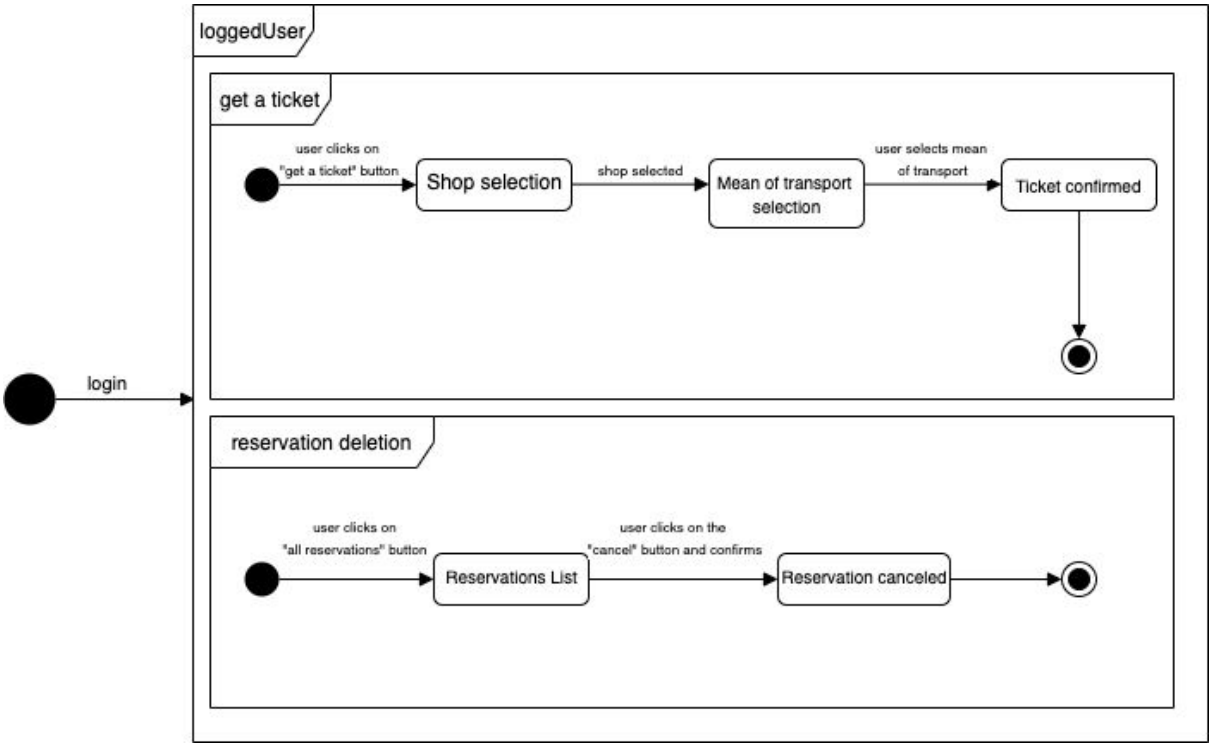
- **SPM6:** After a failure in the attempt to reserve a visit (due to time slots unavailability), a customer receives a suggestion to book in a different date/time and same grocery shop
  As described before, *Customers Line-up* helps mobile app users in finding the perfect visit for them, in case their first choice is not available. A proposal can be a different date/hour (with similarities to the selected ones) and same shop. The customer can accept the suggestion or decide to manually look for other options.

## 2.A.2 UML Class Diagram

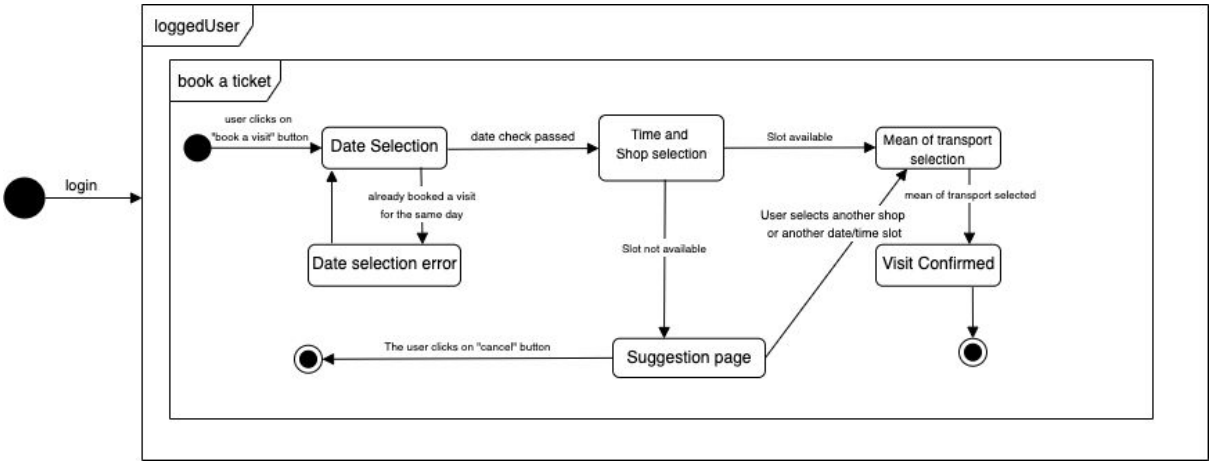


## 2.A.3 UML Statechart Diagrams

1) State diagrams that show the flow followed by a user when it wants to get a ticket for a shop, and when it needs to delete an already booked visit or ticket.




2) State diagram that highlights the flow followed by a user when it wants to book a visit for a specific supermarket, showing possible errors and suggestions given to the customer.



## 2.A.4 Scenarios

- **Get a ticket from CLup app**
  Alice, on Thursday afternoon, on the way back home from her office, decides to go to the supermarket to buy the necessary for dinner. For this reason, she uses the CLup app to check if there is any ticket available with less than 15 minutes of estimated waiting time at her favorite shop (ABC shop). The app shows her that the estimated waiting time for the next ticket in the ABC shop is 10 minutes. So she gets the ticket and continues her way to the shop.

- **Book a visit from CLup app**
  Stefania really likes planning all her activities for the coming week on Monday morning. She works in the city hospital and according to her timetable, she has only time to go to the supermarket on Wednesday. Therefore, in order to have the access to the BCD shop granted, on Monday she decides to use the CLup app to book a visit for Wednesday at 10am. The requested time slot for the visit is available so, after the insertion of the expected duration of the visit and the list of items she plans to buy, the reservation is confirmed and she can go on the day desired.

- **Suggest an alternative, the requested visit is not available**
  Andrea would like to go to his usual grocery, Esselunga (very close to his house), tomorrow at 5pm. It is very important to go tomorrow, because he has almost nothing in the fridge and his wife is complaining a lot about this fact.
  Andrea uses the CLup app on his smartphone to book a visit for tomorrow: he selects his usual supermarket, date and time for the visit. Unfortunately, a message tells him that there is no availability for his request; anyway, the system suggests him another date for his favourite Esselunga (in two days' time, always at 5pm) or another Esselunga shop (a little bit further from his house, but not very distant), available tomorrow at 5pm. Since it is very important for him to do the grocery shopping tomorrow, he accepts this latter advice: he books a visit for tomorrow at 5pm at the suggested Esselunga shop.

- **Available slot suggestion**
  Marco is used to going to the grocery shop close to his house on Saturday afternoon; he always books a visit using *CLup* application on his smartphone, in order to avoid crowds and respect social distance. Due to his high number of reservations in the last months, the system knows Marco's habits about grocery shopping: in fact, the man receives a notification on Monday about an available slot for next Saturday at 4pm. As soon as Marco sees the notification on his smartphone, he fills all requested data for the reservation and books the suggested visit for Saturday: in this way, his habits will continue to be respected also for this week.

- **Notification based on GPS position**
  Antonella has a very busy life, with two little children and a full-time job. Three days ago, she used *CLup* application for booking a visit to her usual supermarket: her turn is today at 4pm. However, her busy life has made her forget about the reservation, and at 3.30pm she is still totally unaware of it. But her smartphone has GPS services running, so at 3.32pm she receives a notification about the approaching turn: so, she immediately gets dressed and goes to the supermarket (approximate time from home to the shop is 10 minutes by car: this information is known by the system thanks to GPS position). She is able to get there at 3.55pm, totally in time for her turn.

- **Delete a ticket reservation**
  Alessandro, because of a sanitary emergency in his country, would like to avoid waiting for his turn in the supermarket's proximity. He has a smartphone and, through CLup app, decides to get a ticket: the expected time before being called is 2 hours. But in the meanwhile, he receives an important job call: he will be busy for the next four hours, so he deletes his ticket reservation through the application; the system notifies him that this operation went fine.

- **Delete a visit reservation**
  Marta works from Monday to Friday as a teacher, so she usually goes to do the grocery shopping on Saturday afternoon. For this reason, she decides, using *CLup* mobile application, to book a visit to the supermarket for next Saturday, at 2pm. But as soon as she remembers that next Saturday she must pick her nephew up from school, she decides to delete her visit reservation (always with the app): the system notifies that this operation went fine, and the 2pm time slot for Saturday is now free for some other customer(s).

- **Entrance in the grocery**
  Matteo is waiting in his car just outside his favourite grocery. Using *CLup* app on his smartphone, he has already got a ticket and his turn is approaching: he knows it thanks to the notification, received a few minutes ago. When it is time to enter, a new notification arrives and Matteo opens the app: he sees the QR code for entrance and a message telling him that it is his turn; QR code will expire in 10 minutes, as shown by another message (in the upper part of the smartphone’s screen).
  Matteo gets out of the car and enters the supermarket by showing his QR code to the scanner at the entrance: doors open and he can get into the grocery.

- **Get a ticket from call center**
  Gabriele, a single forty eight-year old man, wants to go to the grocery shop, on the way home from work by car, to buy the necessary food for the next few days. When he tries to log in the CLup app, he notices that he has no mobile connection available at that moment. So, he decides to call at the call center in order to get a ticket and be able to buy something for the current day. The call center operator tells him that the next ticket available in the requested supermarket has an expected waiting time of 20 minutes. After

Gabriele's confirmation, the operator tells him a numeric code that he will use for the entrance. Moreover the call center operator reminds Gabriele that he will be notified, with a SMS, at the moment he can access the shop. Therefore, Gabriele decides to approach the supermarket and waits for his turn on the car. At the moment he receives the SMS notification he approaches the entrance, shows its numeric code and enters the supermarket.

- **Book a visit from call center**
  Lucia is a seventy-year-old woman without a smartphone. On Wednesday, she decides to go to the grocery shop to buy food for the next few days and, given the current rules about safety distance, she knows that she must get a ticket (for entrance) at one of the ticket machines. She approaches the ticket machine, but she reads on its screen a message saying "Expected waiting time for tickets is: 4 hours". Of course, 4 hours are much more than what Lucia was expecting, so she decides to come back home and call the call center to book a visit for another day. On the phone, she establishes a reservation next Friday at 3p.m., with the call center operator which confirms her reservation telling her a numeric code that she will use to enter the market on Friday. Lucia will come to the supermarket on Friday.

- **Get a ticket from ticket machine**
  Antonio, an eighty-year-old man without a smartphone, approaches a grocery shop on foot, with the intention to buy all necessary for next week. Given the current rules about safety distance, he knows that he must get a ticket (for entrance) at one of the ticket machines outside the supermarket: he selects "Ticket" option on the touchscreen (wearing disposable gloves available near the ticket machine itself) and a ticket with a QR code is printed, together with a 6-digits number identifying his turn; in addition, the screen shows a message telling him that the expected waiting time before his turn is 15 minutes. Antonio decides to walk around and comes back after 10 minutes; two (more) minutes later, his turn comes and the identification number appears on the waiting-screen at the top level of the queue. The man approaches the shop's entrance and shows his QR code (on paper) to the scanner positioned there: doors open and he can get into the supermarket.

- **Ticket machine suggests to reserve the next available slot: no more ticket for the current day are available**
  Giuseppe, an eighty-year old man without a smartphone, approaches a shop on foot, with the intention to buy all necessary for next week. Given the current rules about safety distance, he knows that he must get a ticket (for entrance) at one of the ticket machines outside the supermarket: he selects the "Ticket" option on the touchscreen (wearing disposable gloves available near the ticket machine itself). Unfortunately there are no tickets available for the current day anymore. Thus, the screen shows a message which invites him to reserve the first available slot in the shop in the next few days. The proposal refers to a reservation for the next day at 3pm. Giuseppe accepts the suggestion and reserves a slot for the next day.

## 2.B Product functions

*Customers Line-up* is an application born with the intention of avoiding (physical) crowds outside grocery stores, during a critical epidemic situation. Of course, together with basic functions associated with the need of respecting social distance, the software provides a series of additional features, further detailed in this section.
A list of *CLup* main functionalities follows.

- **Get a ticket**
  Customers who would like to reserve a ticket for accessing the supermarket, without physically lining up, can get one using *Customers Line-up* service. The feature of getting a ticket can be achieved in three ways: the first one is using a smartphone/tablet, the second one calling *CLup*'s call center, the last one with a ticket machine (outside the grocery shop).
  Tickets correspond to "virtual" lining up: when someone gets a ticket, he becomes the last one to wait in the "queue"; if current day's time slots are all reserved (according to the supermarket's opening hours), no more tickets are assigned. A relevant aspect is that visits fill the same "queue" as tickets, but they can be reserved in advance (so, also some days/weeks before the date of the visit itself); in addition, in case of delay or incorrect waiting time estimation because of external factors, visits have priority in being called: visit time is more likely to be respected rather than ticket expected time (a person with the ticket can wait more because entrance hour is not guaranteed, while visit one tends to be, according to real world situations).
  Every customer can download *CLup* application on a mobile device; in order to get a ticket with the app, the guest needs to be registered and specify which grocery shop he wants to visit.
  People not owning a smartphone/tablet can get a ticket reservation by making a phone call to *Customers Line-up* freephone number: in this case, the user can complete the procedure interacting with an IVR system featured with voice recognition, or talking to a human operator; the selected grocery shop and a mobile phone number for SMS notifications (when his turn comes) must be specified.
  In alternative, a person can go directly to the supermarket and get his ticket (if available for that day) using a ticket machine: in this case, there is no need to specify the grocery shop (implicit: it is the closest one), nor to be registered; tickets are printed on paper in the form of QR code, with the addition of a 6-digits number identifying the turn in the waiting "queue".

- **Book a visit**
  People, through *CLup* service, can book a visit for accessing the supermarket, avoiding physical lining up. The functionality of booking a visit, with strong analogy to a ticket reservation, can be achieved in three ways: using a mobile device, calling the call center or with a ticket machine (outside the grocery shop).
  Every customer can download *CLup* mobile application; in order to book a visit with the app, he needs to be registered and specify which grocery shop

he wants to visit, in which day and time. A user might also specify the approximate duration of his visit and the categories of items (if not exact items) he is going to buy, in order to help the system to coordinate other customers' visits/entrances with tickets.
The second option to book a visit is through *Customers Line-up* call center: the procedure is very similar to the one of getting a ticket; the only difference consists in specifying, in addition to the selected grocery shop and a mobile phone number, the day and time for the reservation.
Alternatively, if there aren't available tickets for the rest of the day, a visit for the first available slot can be booked directly from outside the supermarket, using a ticket machine. Visit reservations are printed on paper in the form of QR code (with the addition of a reservation receipt containing all relevant information: supermarket, date, time, 6-digits number for turn identification in the "queue").

- **Delete a reservation**
  Users can delete reservations connected to a ticket or to a booked visit. In order to do so with the mobile application, the user should open the page regarding information about his tickets/visits and select the one he wants to delete.
  The equivalent operation can be performed for tickets or visits connected to a call center reservation: the user should call *Customers Line-up* freephone number and follow the steps until deletion has been confirmed.
  Finally, for deleting reservations booked using a ticket machine, the customer should simply scan the printed QR code with the machine's scanner.

- ***CLup* suggestion mechanism**
  The system, for registered customers using the mobile application, is able to suggest time slots (for visits) based on specific users' habits. In particular, *CLup* stores all data about customers, days and time for visits, as well as most visited supermarkets; analyzing this data, the application can send customized notifications when an attractive visit can be booked, for example when the habitual shop has available visits for the day the customer usually makes use of it.
  Other features for suggestions include giving alternatives after failures in getting a ticket or booking a visit (because no slots are available): for tickets, the system proposes to get one in the closest supermarket (with respect to the selected one) still available for current day; for visits, suggestion coincides with the proposal of a slot for a different hour/a different day/a close supermarket.
  An important remark consists in saying that these features are available only for customers using the mobile application.

- ***CLup* avoids people crowding inside the supermarket**
  The system can manage slots according to known (or inferred) duration of customers' visits and categories of items to buy. The general idea is that, knowing more information, *CLup* can more precisely stagger reservations in order to avoid crowds inside the supermarket. Data is collected by the system; the user can specify it (approximate visit duration and categories/list of items) when booking a visit. In any case, additional relevant information is also retrieved on the spot: time duration spent inside the supermarket and bought items. The information about the average time spent inside the shop by a single user is used to adjust the timetable for other users that may want to book an entrance for the same day: if a user with an average time of visit of one hour and a half booked a place for 6 pm, other users will see his available place starting from 7:30 pm.
  At the same time, data about which items are usually bought by a single customer is used to show different available slots to different users: there is a risk of crowding inside the shop if too many customers that buy similar or near products book a slot for the same date and time.
  The described feature is customized for mobile application users; for customers without the app, estimations are performed (according to average data collected for all the users).

- ***CLup* sends notifications based on GPS position**
  The system, for registered customers using the mobile application and having an active GPS connection on their smartphones, is able to notify them when their reservation (ticket or visit) is approaching. In particular, every time a visit is being booked, the user is asked which kind of means of transport he/she is going to use. This information, combined with the real-time GPS position, allows *CLup* to compute the time needed to get to the supermarket and send the customer a push notification so that he can get on time to the grocery shop, without forming unnecessary crowds outside the shop.
  If GPS connection is not active, the user still receives notifications about an approaching reservation, but only in predefined times. However, both mobile app and call center users receive a notification/SMS when their turn has come.

## 2.C User characteristics

Considering that supermarkets usually sell primary goods, the range of possible users of *CLup* is very wide: basically, every person is a potential user of the system.
If we consider the various types of customers, we can identify the following three categories:

- **Mobile App User**: a person who can take advantage of all the functionalities of *CLup* through the use of the mobile application. Once the user is registered and logged in, he is able to get a ticket or book a visit at any shop, and handle all the reservations he has made. By taking advantage of the real time GPS position and mobile app push notifications, the customer can be notified in order to know when it is time to leave from the current position and when it is his turn to enter the shop.

- **Ticket Machine User**: a person who does not have the mobile app, therefore is not registered and wants to get a ticket as he arrives at the supermarket. This kind of user is only able to get tickets or, if there are no tickets available for the current day, he can book a visit for the first available slot.

- **Call Center User**: a person who does not have the mobile app, therefore is not registered but still wants to get a ticket or book a visit before getting to the supermarket. Giving his mobile number to the call center operator or IVR system, the user can be notified by SMS when his turn comes, avoiding the need to line up physically outside the shop.

# 2.D Assumptions, dependencies and constraints

## 2.D.1 Domain assumptions

| | |
|---|---|
| **DA1** | The grocery shop has a maximum capacity for people inside |
| **DA2** | Only one person enters the grocery shop for every ticket |
| **DA3** | Only one person enters the grocery shop for every visit |
| **DA4** | People inside the grocery shop respect social distance as imposed by safety rules of the country/region |
| **DA5** | People at the grocery shop's entrance respect social distance as imposed by safety rules of the country/region (in case a user's turn has come, but he sees another customer at the code reader at the entrance) |
| **DA6** | Only a very little part of customers uses ticket machines for reservations, because mobile application and call center guarantee a more comfortable service accessible to almost everyone (people with a smartphone/tablet use the application, people without it can call and book) |
| **DA7** | GPS provides the exact location with an error of 10 metres at most |
| **DA8** | |

## 2.D.2 Dependencies

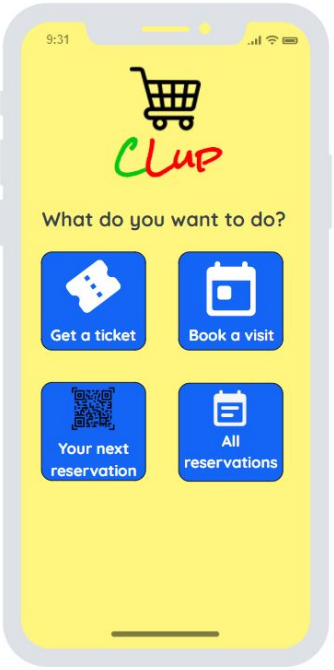
- The *CLup* system will use the internet connectivity of app users' devices.
- The *CLup* system will use the GPS position of mobile app users' devices.
- The *CLup* system will interact with an external push notification service to send notifications to mobile app users.
- The *CLup* system will use a third party API to show maps and compute the time needed to get to the selected shop.
- The *CLup* backend services will use a DBMS service to store and retrieve data about users and their reservations.

# 3 Specific Requirements

## 3.A External Interface Requirements

### 3.A.1 User Interfaces

#### 3.A.1.1 *CLup* mobile application interface




- **Home page**

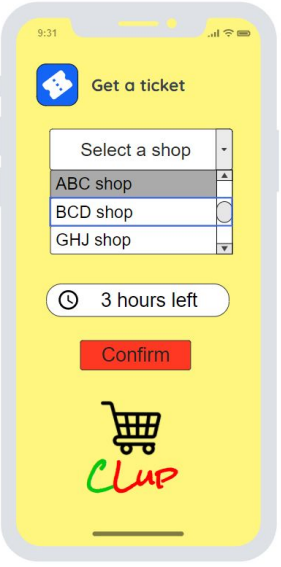
The home page of *CLup* mobile application for logged users shows the following features:

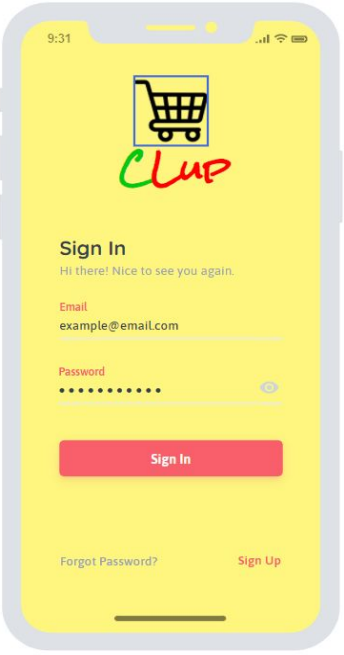
- Get a ticket
- Book a visit
- See the list of reservations
- Manager user (login/logout)

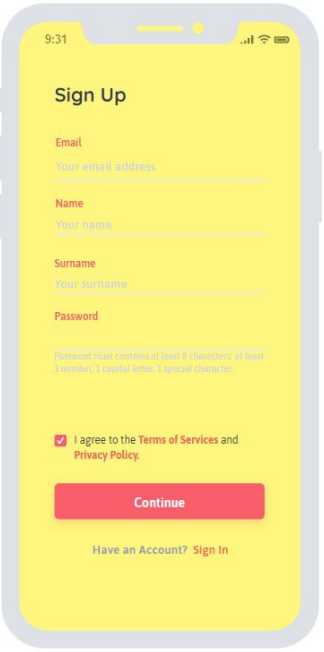
If a user is not logged in, *CLup* only shows a button to sign in/sign up.

- **Get a ticket**

This page allows users to select an available supermarket from a list (shops can also be searched using associated keywords). Once the grocery store has been chosen, users should only confirm to get the ticket reservation.
When users select the shop, a message shows expected time before being called for entrance.





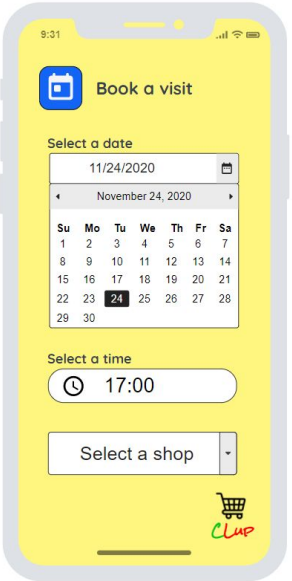



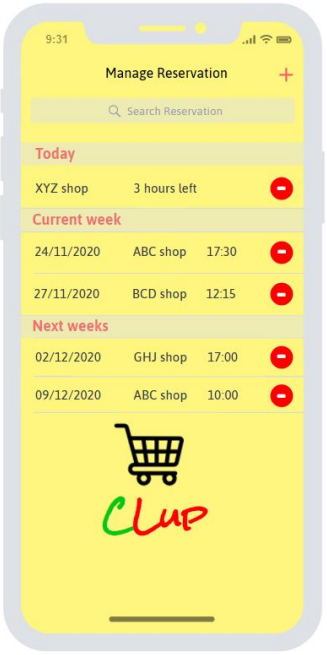

- **Sign up/sign in**

This page allows users to register to the system or log in (if already registered). The user interface allows to insert the email and password; in case of sign up, also name and surname must be added.

- **Book a visit**

  This page allows users to select the supermarket from a list (where customers can also search), in addition to the day and the time for the visit from a calendar. Moreover, there are two optional fields for the duration of the visit and the categories of items to buy (or exact items): both duration and categories should be selected from a predefinite list.






- **List of reservations**

The page allows users to get information about estimated remaining time before the turn (in case of ticket) or day and time of the visit. In addition, there is an option for deleting any reservation. Clicking on a single reservation allows the user to update the estimated duration of his visit and the categories of products he is going to buy.

### *3.A.1.2 CLup* ticket machine interface


- **Home page**

The home page of CLup ticket machine system shows the following options:

  - Get a ticket
  - Delete a reservation

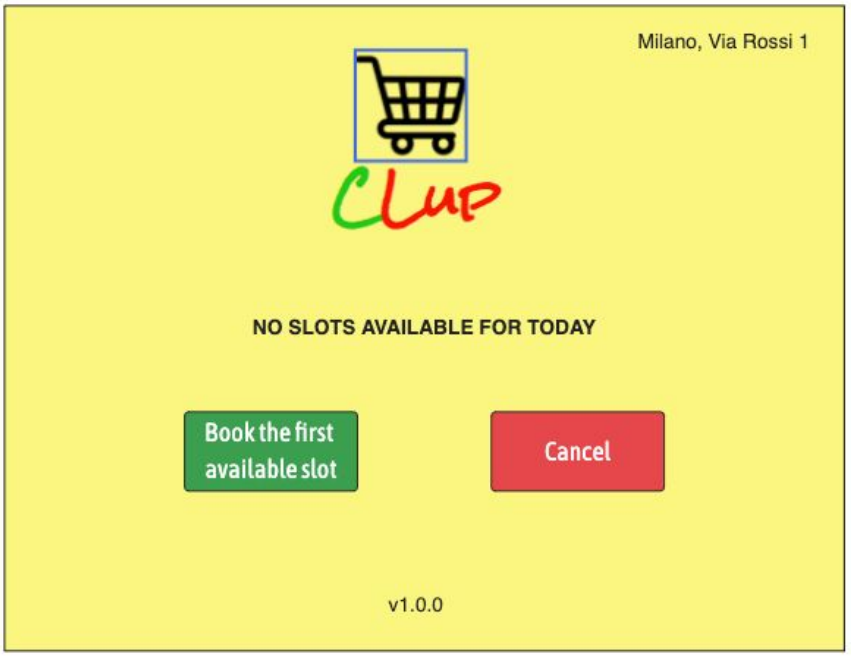
Customers do not need to register or log in (in fact, there is no option for it).

- **Get a ticket**

This page shows the

expected waiting time if any slot is available for the current day.

If no slots are available, an option to book the first available slot is given to the user.





- **Book a visit**

This page shows the expected scheduled date and time for the visit the user is booking.

- **Delete a reservation**

After the reservation's QR Code has been shown to the ticket machine barcode reader, this page displays the details regarding the reservation the user is deleting, and asks for confirmation.


## 3.A.2 Hardware Interfaces

Users must have a mobile device (smartphone/tablet) equipped with a GPS system and mobile data (Internet connection).
Ticket machines are equipped with a touchscreen display, a QR code scanner and Internet access.
A QR code scanner is present at the grocery store's entrance: it is a scanner which can be activated through a movement sensor on the upper part of it. The scanner is connected with the supermarket's doors, which open when an active QR code has been scanned.

There are screens outside the grocery store which are used for displaying turns, and they must be big enough so that reading from 15 metres away is possible.

### 3.A.3 Software Interfaces

Regarding the frontend for the mobile application, the app should be compatible with the following operating system:

- Android (from version 5.0 Lollipop in order to be compatible with the 94.1% of the devices [data regarding December 2020, source: android.com])
- iOS (from version 12.0 in order to be compatible with the 94% of the devices [source: apple.com])

The mobile app must interact with the following external software component:

- Geolocalization service (e.g Google Maps SDK)
- Push Notification Service (e.g Google Firebase Cloud Messaging)

Moreover, the CLup's mobile application interacts with CLup's backend services. In particular:

- User authentication services
- Reservation handling services (also used by the operators in call center and the ticket machine)

The backend components make use of a database (relational database or NoSQL) to store data about Users and Reservations. Using a database implies the integration of a software library to interact with the chosen DBMS.

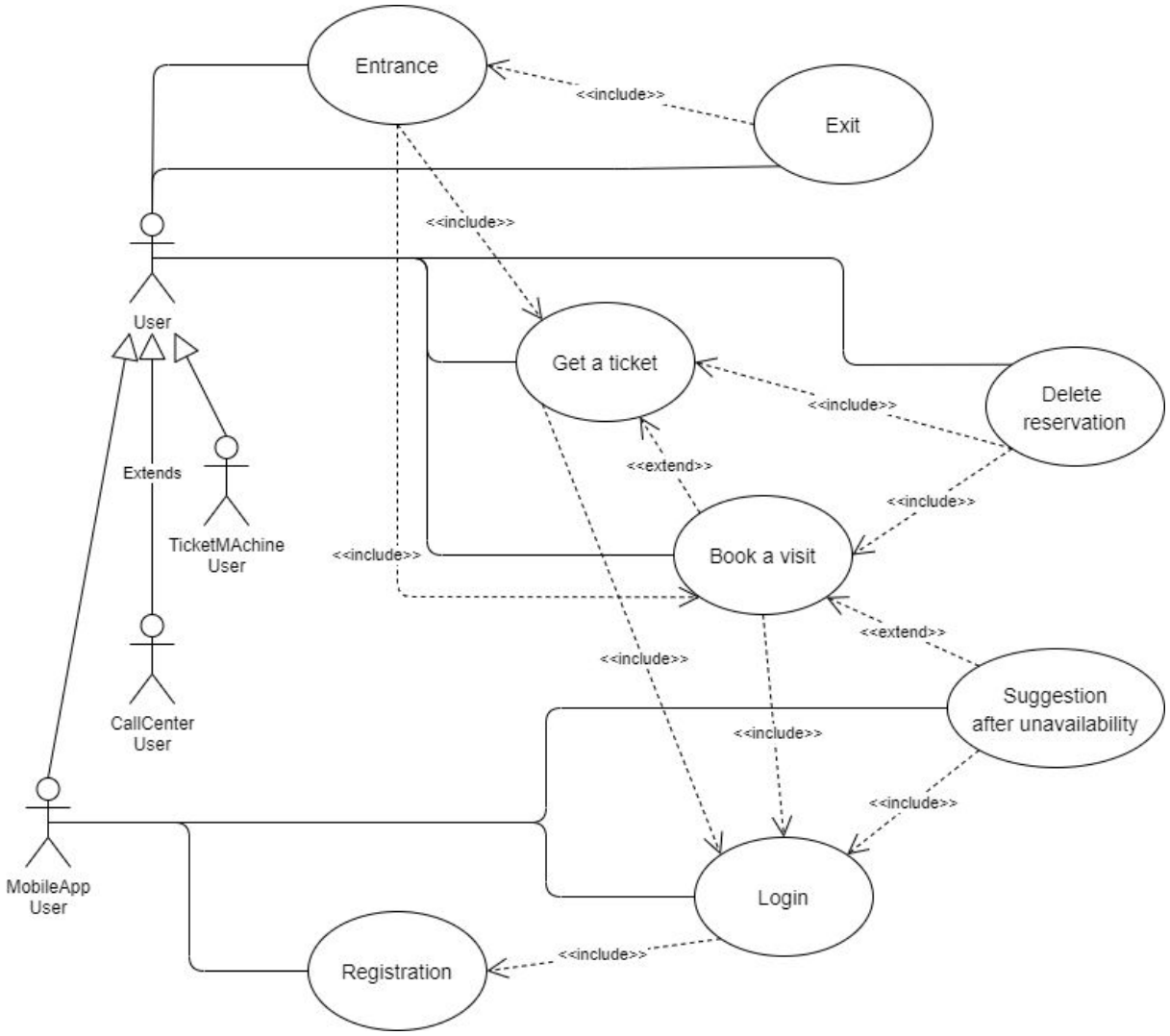
### 3.A.4 Communication Interfaces

HTTPS protocol is used to grant secure data transmission over the Internet.

## 3.B Functional Requirements

### 3.B.1 Use case diagram

a




**NOTES ON THE DIAGRAM**

- *<<include>>* is mainly used for use cases which precede other use cases; for example, the activity of getting a ticket is only possible after logging in (for mobile app users)
- *Entrance* is preceded either by getting a ticket or booking a visit: only one is enough
- *Delete reservation* is preceded either by getting a ticket or booking a visit: only one is enough
- *<<extend>>* between *Book a visit* and *Get a ticket* has the following meaning: if no tickets are available, a user can still book a visit (for another day)
- *<<extend>>* between *Suggestion after unavailability* and *Book a visit* has the following meaning: if a visit slot is not available, a mobile app user receives proper suggestions

## 3.B.2 Use cases

- Registration of mobile app user

| | |
|---|---|
| **ID** | UC1 |
| **Name** | Registration of mobile app user |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks on “Sign up” button<br>5. User inserts email, name and surname in appropriate fields<br>6. User chooses his password for the service, according to security standards<br>7. User accepts *CLup* “Terms and conditions”<br>8. User clicks on “Continue” button<br>9. User sees the page for registering his mobile phone number<br>10. User inserts his mobile phone number and clicks on “Send a SMS” button<br>11. The system sends a SMS to the specified phone number, containing a 8-digits code<br>12. User receives the SMS and fills the appropriate field in *CLup* page with the received code<br>13. User clicks on “Confirm” button<br>14. The system confirms registration with an email; in the meanwhile, the app goes back to the home page<br>15. User opens the email and validates his registration by opening the suggested link |
| **Exit condition** | ● The user is now able to login and use the application services |
| **Exceptions** | 1. User does not fill some of the mandatory fields in the first page (email, name, surname, password, “Terms and conditions” acceptance)<br>2. Password does not match security standards<br>3. Email is already registered in the system’s DB<br>4. User inputs a non-valid email address<br>5. Email does not exist<br>6. User is not the owner (or, in any case, has no visibility) of the specified email address<br>7. User does not fill some of the mandatory fields in |

| | |
|---|---|
| | the second page (mobile phone number, received code)<br>8. Mobile phone number is already registered in the system's DB<br>9. User inputs a code different from the one sent by the system<br>10. Mobile phone number does not exist<br>11. User is not the owner (or, in any case, has no visibility) of the specified mobile phone number<br><br>In cases 1,2,3,4, the system does not let the "Continue" button to be pressed until all fields are correctly filled in.<br>In cases 5, 6, the user never receives an email, so he will never be able to complete registration.<br>In cases 7,8, the system does not let the "Confirm" button to be pressed until all fields are correctly filled in.<br>In case 9, the user can ask for a new code with a click on the appropriate button.<br>In cases 10,11, the user does never receive a SMS, so he will very likely not be able to proceed. |

** until the end of UC1 specification, references to "mobile app user" will be made using the word "user"*

- Login of mobile app user

| | |
|---|---|
| **ID** | UC2 |
| **Name** | Login of mobile app user |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks on "Sign in" button<br>5. User inserts email and password in appropriate fields<br>6. User clicks on "Sign in" button<br>7. The system checks for a matching with all registered users (with their passwords) in the DB<br>8. The system confirms login and shows the home page |

| | |
|---|---|
| **Exit condition** | ● User is logged in<br>● User can use all *CLup* services |
| **Exceptions** | 1. User does not fill some of the mandatory fields (email, password)<br>2. The system does not find a matching with registered users (and corresponding passwords) in the DB<br>3. User account has been blocked by *CLup*<br><br>In case 1, the system does not let the "Confirm" button to be pressed until all fields are correctly filled in.<br>In cases 2,3, the system notifies an appropriate error message and goes back to the login page (where to insert again email and password). |

** until the end of UC2 specification, references to "mobile app user" will be made using the word "user"*

- Mobile app user gets a ticket

| | |
|---|---|
| **ID** | UC3 |
| **Name** | Mobile app user gets a ticket |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service<br>● User is logged in |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks on "Get a ticket" button<br>5. User selects a supermarket from the list, directly or after searching<br>6. User clicks "Confirm" button<br>7. The system notifies "Success" message and displays expected waiting time before being called |
| **Exit condition** | ● User has a ticket reservation<br>● User sees the ticket in "All reservations" page of *CLup* application |
| **Exceptions** | 1. User has already a "pending" (booked but not called yet) ticket reservation<br>2. A "pending" (booked but not called yet) ticket is already associated to the user's phone number, by a |

| | |
|---|---|
| | call center reservation<br><br>In cases 1,2, the system does not let the “Confirm” button to be pressed. |

** until the end of UC3 specification, references to “mobile app user” will be made using the word “user”*

- Mobile app user books a visit

| | |
|---|---|
| **ID** | UC4 |
| **Name** | Mobile app user books a visit |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service<br>● User is logged in |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks on “Book a visit” button<br>5. User selects desired day on the appeared calendar, then the time for the visit<br>6. User selects a supermarket from the list, directly or after searching**<br>7. User clicks “Confirm” button<br>8. The system notifies “Success” message |
| **Exit condition** | ● User has a visit reservation<br>● User sees booked visit in “All reservations” page of *CLup* application |
| **Exceptions** | 1. No availability for selected supermarket, day and time<br>2. User has already a “pending” (booked but not called yet) reservation for the selected day (in any grocery shop)<br>3. A “pending” (booked but not called yet) reservation is already associated to the the selected day and the user’s phone number, by a call center reservation<br><br>In case 1, refer to use case 5) ^<br>In cases 2,3, the system does not let the “Confirm” button to be pressed and shows a proper message to the user. |

** until the end of UC4 specification, references to "mobile app user" will be made using the word "user"*
*** note that, at 6), the customer also has the option to specify the items he/she plans to buy and the expected duration for the visit*
*^ the situation is completely described in another use-case, so read and refer only to it*

- Mobile app user receives suggestions after failure in booking a visit

| **ID** | UC5 |
|---|---|
| **Name** | Mobile app user receives suggestions after failure in booking a visit |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service<br>● User is logged in |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks on "Book a visit" button<br>5. User selects desired day on the appeared calendar, then the time for the visit<br>6. User selects a supermarket from the list, directly or after searching<br>7. User clicks "Confirm" button<br>8. The system shows a message telling the user about no availability for selected day, time and supermarket<br>9. The systems suggests the user available slots for: same supermarket, same day and different hour (±2 hours); same supermarket, same hour and different day (±2 days); same day, same hour, different supermarket (among the five closest to the selected one) |
| **Exit condition** | ● User has a visit reservation<br>● User sees booked visit in "All reservations" page of *CLup* application |
| **Exceptions** | 1. Availability for selected supermarket, day and time<br>2. User has already a reservation for the selected day (in any grocery shop)<br><br>In case 1, refer to use case 4) ^<br>In case 2, the system does not let the "Confirm" button to be pressed and shows a proper message to the user. |

** until the end of UC5 specification, references to "mobile app user" will be made using the word "user"*
*^ the situation is completely described in another use-case, so read and refer only to it*

- Mobile app user deletes a ticket reservation

| | |
|---|---|
| **ID** | UC6 |
| **Name** | Mobile app user deletes a ticket reservation |
| **Actors** | Mobile app user* |
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service<br>● User is logged in<br>● User has got a ticket for current day and his turn has not come yet |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks the button to see all his reservations in the home page of the application<br>5. User clicks the deletion button for the ticket he wants to delete<br>6. The system shows "Are you sure you want to delete your ticket?" message dialog<br>7. User clicks "Yes" button<br>8. The system notifies "Success" message |
| **Exit condition** | ● User has no tickets for the current day<br>● User does not see any ticket in "All reservations" page of *CLup* application |
| **Exceptions** | |

** until the end of UC6 specification, references to "mobile app user" will be made using the word "user"*

- Mobile app user deletes a visit reservation

| | |
|---|---|
| **ID** | UC7 |
| **Name** | Mobile app user deletes a visit reservation |
| **Actors** | Mobile app user* |

| | |
|---|---|
| **Entry condition** | ● User has the internet connection available<br>● User has downloaded *CLup* app on his smartphone/tablet<br>● User is registered to the service<br>● User is logged in<br>● User has booked a visit and his turn has not come yet |
| **Event flow** | 1. User unlocks his smartphone<br>2. User opens *CLup* app<br>3. User sees the home page of the app<br>4. User clicks the button to see all his reservations in the home page of the application<br>5. User clicks the deletion button for the visit he wants to delete<br>6. The system shows "Are you sure you want to delete your visit?" message dialog<br>7. User clicks "Yes" button<br>8. The system notifies "Success" message |
| **Exit condition** | ● User does not have the selected visit reservation anymore<br>● User does not see the selected visit in "All reservations" page of *CLup* application |
| **Exceptions** | |

** until the end of UC7 specification, references to "mobile app user" will be made using the word "user"*

- Call center user gets a ticket

| | |
|---|---|
| **ID** | UC8 |
| **Name** | Call center user gets a ticket |
| **Actors** | Call center user* |
| **Entry condition** | ● User has a mobile device enabled to phone calls, with a SIM card<br>● User's phone number is not blocked neither by *CLup* call center nor by *CLup* services (in general)<br>● User is in a place where there is signal<br>● User knows *CLup* call center telephone number<br>● Date and time correspond to one of *CLup* call center's opening hours |

| | |
|---|---|
| **Event flow** | 1. User unlocks his mobile device<br>2. User opens the program/app to make phone calls<br>3. User dials *CLup* call center telephone number<br>4. The IVR system lists a set of options: “press 1 to get a ticket”; “press 2 to book a visit”, “press 3 to delete a reservation”, “press 0 to talk to an operator”<br>5. User presses 1 on the keypad<br>6. The IVR system proposes the following options: “after the beep sound, pronounce the name, the town and the street of the grocery shop (if there is only one shop in the selected town, street is optional)”; “press 0 to talk to an operator”<br>7. User hears a beep sound<br>8. User pronounces name, town and (optionally) street, as suggested by the IVR system<br>9. The system looks for the grocery shop which better fits the one pronounced by the user, and repeats it together with the expected waiting time (before the turn comes)<br>10. The IVR system proposes the following options: “press 1 to get the ticket”; “press 2 to pronounce another grocery shop”; “press 0 to talk to an operator”<br>11. User presses 1 on the keypad<br>12. The IVR system proposes the following options: “press 1 to confirm you would like to receive text messages for entrance to the phone number you are using for this call”; “press 2 to insert another phone number”<br>13. User presses 1 on the keypad<br>14. The IVR system confirms the ticket, repeats the expected waiting time and tells the code for entrance<br>15. The system ends the phone call<br>16. User receives a SMS confirming the ticket reservation, together with a code for entrance |
| **Exit condition** | ● User has a ticket reservation<br>● User can see all the information about the ticket reservation (grocery shop, expected waiting time) in the received SMS |
| **Exceptions** | 1. No more tickets are available for current day and selected supermarket<br>2. The phone call ends before the user explicits on which phone number he would like to receive text messages<br>3. User presses 0, in order to talk to an operator<br>4. The IVR system does not recognize the grocery shop pronounced by the user |

| | |
|---|---|
| | 5. User inputs a non-valid number, with respect to the options given by the IVR system (at any moment)<br>6. User inputs while the IVR system is talking<br>7. User inputs a non-valid phone number for SMS notifications<br>8. User inputs a phone number for SMS notifications which has been blocked by *CLup*<br>9. A "pending" (booked but not called yet) ticket is already associated to the inserted phone number (either from a call center reservation or by a mobile app user with the same phone number)<br><br>In case 1, the IVR system tells the user about unavailability and proposes only to pronounce another grocery shop or to talk to an operator.<br>In case 2, the procedure to get a ticket is not completed and the user does not have a ticket reservation.<br>In case 3, the user is put on hold until the first free operator is able to talk to him.<br>In case 4, the IVR system still proposes to pronounce another grocery shop or to talk to an operator, so the user can repeat.<br>In cases 5,7, the system waits until the first valid input has been written (for phone number, a specific termination sequence is given, so that the system can automatically know when the input terminates).<br>In case 6, all inputs received before the sentence has finished are ignored.<br>In cases 8,9, the user is invited to input another phone number in order to complete the reservation request; an information message about the cause of the fault follows. |

** until the end of UC8 specification, references to "call center user" will be made using the word "user"*

- Call center user books a visit

| | |
|---|---|
| **ID** | UC9 |
| **Name** | Call center user books a visit |
| **Actors** | Call center user* |
| **Entry condition** | ● User has a mobile device enabled to phone calls, with a SIM card<br>● User's phone number is not blocked neither by *CLup* call center nor by *CLup* services (in general)<br>● User is in a place where there is signal<br>● User knows *CLup* call center telephone number |

| | |
|---|---|
| | • Date and time correspond to one of *CLup* call center's opening hours |
| **Event flow** | 1. User unlocks his mobile device<br>2. User opens the program/app to make phone calls<br>3. User dials *CLup* call center telephone number<br>4. The IVR system lists a set of options: "press 1 to get a ticket"; "press 2 to book a visit", "press 3 to delete a reservation", "press 0 to talk to an operator"<br>5. User presses 2 on the keypad<br>6. The IVR system proposes the following options: "after the beep sound, pronounce the name, the town and the street of the grocery shop (if there is only one shop in the selected town, street is optional)"; "press 0 to talk to an operator"<br>7. User hears a beep sound<br>8. User pronounces name, town and (optionally) street, as suggested by the IVR system<br>9. The system looks for the grocery shop which better fits the one pronounced by the user, and repeats it<br>10. The IVR system proposes the following options: "press 1 to book a visit"; "press 2 to pronounce another grocery shop"; "press 0 to talk to an operator"<br>11. User presses 1 on the keypad<br>12. The IVR system proposes the following options: "if you want to complete the reservation soon, write with your keypad four digits, in order to specify the day and month of your visit (for example, if you would like to visit on the third of July, write zero three zero seven)"; "press # to talk to an operator"<br>13. User dials a four digits number on the keypad<br>14. The IVR system proposes the following options: "if you want to complete the reservation soon, write with your keypad four digits, in order to specify the exact hour of your visit (for example, if you would like to visit at half past nine, write zero nine three zero)"; "press # to talk to an operator"<br>15. User dials a four digits number on the keypad<br>16. The system looks for the closest earlier (in time, since supermarket is fixed) available time slot (<=) and the closest later (>) to the one selected by the user, and repeats them<br>17. The IVR system proposes to press 1 to book the closest earlier (<=) visit, to press 2 to book the closest later (>) visit, to press 3 to input another date, to press 4 to input another time (keeping the date fixed), to press 4 to select another grocery shop, to press 0 to talk to an operator |

| | |
|---|---|
| | 18. User presses 1 on the keypad<br>19. The IVR system proposes the following options: “press 1 to confirm you would like to receive text messages for entrance to the phone number you are using for this call”; “press 2 to insert another phone number”<br>20. User presses 1 on the keypad<br>21. The IVR system confirms the visit reservation, repeats date, time and tells the code for entrance<br>22. The system ends the phone call<br>23. User receives a SMS confirming the visit reservation, together with a code for entrance |
| **Exit condition** | ● User has a visit reservation<br>● User can see all the information about the visit reservation (grocery shop, date, time) in the received SMS |
| **Exceptions** | 1. The phone call ends before the user explicits on which phone number he would like to receive text messages<br>2. User presses the digit for talking to an operator<br>3. The IVR system does not recognize the grocery shop pronounced by the user<br>4. User inputs a non-valid number, with respect to the options given by the IVR system (at any moment)<br>5. User dials a number with less than four digits, when he is asked to specify date/hour for the visit<br>6. User dials a number with more than four digits, when he is asked to specify date/hour for the visit<br>7. User inputs while the IVR system is talking<br>8. User inputs a non-valid phone number for SMS notifications<br>9. User inputs a phone number for SMS notifications which has been blocked by *CLup*<br>10. A “pending” (booked but not called yet) visit is already associated to the inserted date and phone number (either from a call center reservation or by a mobile app user with the same phone number)<br><br>In case 1, the procedure to book a visit is not completed and the user does not have a visit reservation.<br>In case 2, the user is put on hold until the first free operator is able to talk to him.<br>In case 3, the IVR system still proposes to pronounce another grocery shop or to talk to an operator, so the user can repeat.<br>In cases 4,5,8, the system waits until the first valid input has been written (for phone number, a specific termination sequence is given, so that the system can automatically |

| | |
|---|---|
| | know when the input terminates).<br>In case 6, the system ignores all digits dialed after the required four and before the IVR has listed all the options of the following step.<br>In case 7, all inputs received before the sentence has finished are ignored.<br>In cases 9,10, the user is invited to input another phone number in order to complete the reservation request; an information message about the cause of the fault follows. |

** until the end of UC9 specification, references to "call center user" will be made using the word "user"*

- Call center user deletes a reservation

| | |
|---|---|
| **ID** | UC10 |
| **Name** | Call center user deletes a reservation |
| **Actors** | Call center user* |
| **Entry condition** | ● User has a mobile device enabled to phone calls, with a SIM card<br>● User's phone number is not blocked neither by *CLup* call center nor by *CLup* services (in general)<br>● User is in a place where there is signal<br>● User knows *CLup* call center telephone number<br>● Date and time correspond to one of *CLup* call center's opening hours<br>● User has got a ticket for current day / booked a visit and his turn has not come yet<br>● User has at his disposition the code associated to his reservation<br>● User has at his disposition the mobile device associated to his reservation |
| **Event flow** | 1. User unlocks his mobile device<br>2. User opens the program/app to make phone calls<br>3. User dials *CLup* call center telephone number<br>4. The IVR system lists a set of options: "press 1 to get a ticket"; "press 2 to book a visit", "press 3 to delete a reservation", "press 0 to talk to an operator"<br>5. User presses 3 on the keypad<br>6. The IVR system proposes the following options: "if you want to delete your reservation soon, write with your keypad the code associated to it"; "press # to talk to an operator"<br>7. User dials the code on the keypad<br>8. The IVR system repeats the input code and all data |

| | |
|---|---|
| | connected to the reservation; then, it proposes the following options: “press 1 to delete your reservation”; “press 2 if you want to keep (not delete) this reservation, but you want to input another reservation code for deletion”; “press 0 to talk to an operator”<br>9. User presses 1 on the keypad<br>10. The system sends a SMS to the phone number connected to the reservation, then informs the user to input the security code just received; the option to talk to an operator is present, too<br>11. User receives a SMS with a security code for deletion<br>12. User dials the security code<br>13. The system confirms deletion and ends the phone call<br>14. User receives a SMS confirming the reservation deletion |
| **Exit condition** | ● User does not have the selected reservation anymore<br>● The code connected to the selected reservation is no more active |
| **Exceptions** | 1. The phone call ends before the user dials the security code<br>2. User presses the digit for talking to an operator<br>3. User inputs a non-valid reservation code<br>4. User inputs a wrong security code<br>5. User inputs a non-valid number, with respect to the options given by the IVR system (at any moment)<br>6. User dials a number with less digits than expected, when he is asked to specify reservation/security code<br>7. User dials a number with more digits than expected, when he is asked to specify reservation/security code<br>8. User inputs while the IVR system is talking<br><br>In case 1, the procedure to delete a reservation is not complete and ticket/visit is still valid for the user.<br>In case 2, the user is put on hold until the first free operator is able to talk to him.<br>In case 3, if the code is associated with an existing reservation, the IVR system proposes to delete it or to digit another code (so, a user who dialed in a wrong way can repeat his right code); if not, the system only proposes to input another code.<br>In case 4, the system notifies the error and asks for a new security code, after sending another SMS. After 5 errors in the same phone call, the system ends it and the calling |

| | |
|---|---|
| | number is blocked for the following 30 minutes.<br>In cases 5,6, the system waits until the first valid input has been written.<br>In case 7, the system ignores all digits dialed after the required ones and before the IVR has completed the sentence of the following step.<br>In case 8, all inputs received before the sentence has finished are ignored. |

** until the end of UC10 specification, references to "call center user" will be made using the word "user"*

- Ticket machine user gets a ticket

| | |
|---|---|
| **ID** | UC11 |
| **Name** | Ticket machine user gets a ticket |
| **Actors** | Ticket machine user* |
| **Entry condition** | • User is approaching the grocery shop during its opening hours |
| **Event flow** | 1. User waits for his turn in order to use the ticket machine<br>2. User clicks on "Get a Ticket" button<br>3. The ticket machine shows the expected waiting time before being called<br>4. User clicks on "Confirm" button<br>5. A paper with the QR code and a unique queue number gets printed |
| **Exit condition** | • User has a ticket reservation<br>• User can wait for his queue number to show on the display, as his turn to enter begins |
| **Exceptions** | 1. There are no tickets available for the current day<br>2. After clicking on "Get a Ticket" button, the user clicks on "Cancel" button<br>3. After clicking on "Get a Ticket" button, the user remains inactive for more than 3 minutes<br><br>In case 1, refer to use-case 12) ^<br>In cases 2,3, the ticket machine returns to its initial state. |

** until the end of UC11 specification, references to "ticket machine user" will be made using the word "user"*
*^ the situation is completely described in another use-case, so read and refer only to it*

- Ticket machine user books a visit

| | |
|---|---|
| **ID** | UC12 |
| **Name** | Ticket machine user books a visit |
| **Actors** | Ticket machine user* |
| **Entry condition** | • User is approaching the grocery shop during its opening hours |
| **Event flow** | 1. User waits for his turn in order to use the ticket machine<br>2. User clicks on “Get a Ticket” button<br>3. The ticket machine shows an error message telling the user that there are no available tickets for the current day<br>4. User clicks on “Book the first available slot” button<br>5. The ticket machine shows the date and time of the visit<br>6. User clicks on “Confirm” button<br>7. A paper with the QR code, a unique queue number and all information about the visit gets printed |
| **Exit condition** | • User has a visit reservation<br>• User can wait for his visit date and time to come |
| **Exceptions** | 1. After clicking on “Get a Ticket” button, the user remains inactive for more than 3 minutes<br>2. After clicking on “Get a Ticket” button, the user clicks on “Cancel” button<br>3. After clicking on “Book the first available slot” button, the user remains inactive for more than 3 minutes<br>4. After clicking on “Book the first available slot” button, the user clicks on “Cancel” button<br>5. There is at least one available ticket for the current day<br><br>In cases 1,2,3,4, the ticket machine returns to its initial state.<br>In case 5, refer to use-case 11) ^ |

** until the end of UC12 specification, references to “ticket machine user” will be made using the word “user”*
*^ the situation is completely described in another use-case, so read and refer only to it*

- Ticket machine user deletes a reservation

| | |
|---|---|
| **ID** | UC13 |
| **Name** | Ticket machine user deletes a reservation |

| | |
|---|---|
| **Actors** | Ticket machine user* |
| **Entry condition** | ● User is approaching the grocery shop during its opening hours |
| **Event flow** | 1. User waits for his turn in order to use the ticket machine<br>2. User clicks on “Delete a reservation” button<br>3. The ticket machine shows a message telling the user to put the reservation QR code in front of the barcode reader of the machine<br>4. The ticket machine shows a message asking the user if he really wants to delete the reservation, showing the reservation details<br>5. User clicks on “Confirm” button<br>6. The ticket machine shows a confirmation message telling the user that the reservation has been deleted |
| **Exit condition** | ● The QR code of the reservation deleted by the user is no longer valid |
| **Exceptions** | 1. After clicking on “Delete a reservation” button, the user remains inactive for more than 3 minutes<br>2. After clicking on “Delete a reservation” button, the user clicks on “Cancel” button<br>3. When the ticket machine asks the user if he really wants to delete the reservation, the user remains inactive for more than 3 minutes<br>4. When the ticket machine asks the user if he really wants to delete the reservation, the user clicks on “Cancel” button<br>5. The ticket machine can not find a reservation associated to the QR code showed by the user<br><br>In cases 1,2,3,4, the ticket machine returns to its initial state.<br>In case 5, the ticket machine shows an error message. |

** until the end of UC13 specification, references to “ticket machine user” will be made using the word “user”*

- User enters the grocery shop

| | |
|---|---|
| **ID** | UC14 |
| **Name** | User enters the grocery shop |
| **Actors** | User |

| | |
|---|---|
| **Entry condition** | ● User has a ticket/visit reservation<br>● User has received a notification (mobile app and/or SMS) for entering, or has seen the call for the number associated to his reservation on the display (outside the supermarket)<br>● User is outside the grocery shop's entrance, in front of the code reader<br>● User has the code for entrance at his immediate disposition |
| **Event flow** | 1. User activates the code reader with a movement near the sensor, as shown by an information plaque<br>2. Code reader screen shows a message, inviting the user to scan/input the code<br>3. User shows the QR code for entrance to the reader (mobile app user, ticket machine user), or inputs the code with the reader's keypad after wearing disposable gloves (call center user, mobile app user who prefers/has the necessity to use the code received by SMS)<br>4. The system processes the code, checking for the possibility to enter<br>5. Doors open and code reader screen shows a blank page<br>6. User gets into the grocery shop<br>7. Doors close |
| **Exit condition** | ● User is inside the supermarket and can do the grocery shopping |
| **Exceptions** | 1. The movement near the sensor does not turn the code reader on<br>2. The code for entrance is not recognized (impossibility to read QR code and/or wrong code)<br>3. The code for entrance expired<br><br>In case 1, the user can try again with a movement near the sensor.<br>In case 2, the code reader shows an error message, turns off and the user can repeat the whole procedure.<br>In case 3, the code reader screen shows a message to inform about the expiration. |

- User pays and exits the grocery shop

| | |
|---|---|
| **ID** | UC15 |
| **Name** | User pays and exits the grocery shop |

| | |
|---|---|
| **Actors** | User, cashier (in case of non-automatic check-out) |
| **Entry condition** | ● User has a ticket/visit reservation<br>● User has already received a notification (mobile app and/or SMS) for entering, or has already seen the call for the number associated to his reservation on the display (outside the supermarket); he has already entered the supermarket<br>● User is at the check-out inside the grocery shop<br>● User has the code for entrance at his disposition |
| **Event flow** | 1. User (in case of automatic cashier) / cashier (in case of traditional check-out) scans all the products the customer would like to buy<br>2. The cashier asks the user to show/input the code associated to his reservation to the code reader<br>3. The cashier activates the code reader<br>4. User picks the code associated to his reservation<br>5. User shows the QR code to the reader (mobile app user, ticket machine user), or inputs the code with the reader's keypad after wearing disposable gloves (call center user, mobile app user who prefers/has the necessity to use the code received by SMS)<br>6. The system processes and checks the code<br>7. The system recognizes the code and notifies the cashier<br>8. The cashier asks the user to pay<br>9. User pays<br>10. User leaves the check-out |
| **Exit condition** | ● User is going to exit the grocery shop<br>● The system knows that the user is going to exit soon |
| **Exceptions** | 1. The code associated to the reservation is not recognized (impossibility to read QR code and/or wrong code)<br><br>In case 1, the user is asked to show/input the code again, after making sure that the code itself is the right one. |

***Alternative***

In case of a user who is going to buy nothing, the exit way turnstyle opens after the customer shows/inputs the code to the code reader placed there; the way is analogous to UC14 for entrance (but here it is valid for exiting).

### 3.B.3 Sequence diagrams

Da aggiungere

### 3.B.4 Requirements

| | |
|---|---|
| **R1** | The system allows a mobile app user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his profile) |
| **R2** | The system allows a call center user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his phone number) |
| **R3** | The system allows a ticket machine user to get a ticket for the corresponding (implicitly) grocery shop |
| **R4** | The system allows a mobile app user to book one or more visits |
| **R5** | The system allows a call center user to book one or more visits |
| **R6** | The system allows a ticket machine user to book a visit only if no slots for tickets are available in the current day for the selected (implicitly) grocery shop |
| **R7** | The system allows a user to book slots for shopping for the next month from current date and time |
| **R8** | The system allows a user to delete a reservation |
| **R9** | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.1:** The system generates a unique QR code associated to every ticket/visit booked from user application<br>**R.9.2:** The system generates a unique numeric code associated to every ticket/visit booked calling the call center<br>**R.9.3:** The system generates a unique QR code associated to every ticket/visit booked from a ticket machine |
| **R10** | The system generates a unique numeric code for the position in the waiting “queue” associated to every ticket/visit booked from a ticket machine Specificare unique per supermercato e giorno? |
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R12** | The system allows a customer to have the possibility to enter only when it is his turn and for the next ten minutes |

| | |
|---|---|
| **R13** | The system notifies a user when his turn for entrance is approaching<br><br>**R.13.1:** The system sends a notification two minutes before entrance (in case of mobile app user and call center user)<br>**R.13.2:** The system shows on a display the waiting “queue” numeric code associated to a ticket machine reservation when the latter’s turn comes |
| **R14** | The system does not notify a user for entrance if the grocery shop's maximum capacity has been reached |
| **R15** | The system does not allow a person to enter if the grocery shop's maximum capacity has been reached<br><br>**R15.1:** The system does not allow a person to book a reservation if the grocery shop's maximum capacity has been reached for the selected slot(s) |
| **R16** | The system allows a mobile app user to get information about estimated time before being called for entrance (with relation to a booked reservation) |
| **R17** | The system allows a ticket machine user to get information about estimated time before being called for entrance (with relation to a booked reservation), through a display in the grocery shop’s proximity |
| **R18** | The system is able to check availability for all the grocery shops that it manages |
| **R19** | The system, in case of mobile application, sends notifications basing on user's actual position and time remaining before his turn (with relation to a booked reservation) |
| **R20** | The system registers time spent inside the supermarket for every customer |
| **R21** | The system is able to build statistics basing on time spent in the grocery shop and items/categories of items purchased |
| **R22** | The system, in case of mobile application, allows to insert a list/categories of items (that user would like to buy) when booking a visit |
| **R23** | The system manages available slots depending on reservations and grocery shops’ capacities |
| **R24** | The system manages available slots depending on statistics built on specific user’s habits (usual time spent in the grocery shop, items/categories of items normally purchased) |
| **R25** | The system tracks user habits in order to detect his habitual grocery |

| | |
|---|---|
| | shop and date/time of visits, in order to provide booking suggestions in advance if any “appealing” slot is available |
| **R26** | Call centers are open in the grocery shops’ opening hours |
| **R27** | Ticket machines are available for booking in the grocery shop’s opening hours |
| **R28** | A user application is associated to a profile (personal account) |

### 3.B.5 Use cases mapping on requirements

| **UC1** | **Registration of mobile app user** |
|---|---|
| **R1** | The system allows a mobile app user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his profile) |
| **R4** | The system allows a mobile app user to book one or more visits |
| **R28** | A user application is associated to a profile (personal account) |

| **UC2** | **Login of mobile app user** |
|---|---|
| **R1** | The system allows a mobile app user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his profile) |
| **R4** | The system allows a mobile app user to book one or more visits |
| **R28** | A user application is associated to a profile (personal account) |

| **UC3** | **Mobile app user gets a ticket** |
|---|---|
| **R1** | The system allows a mobile app user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his profile) |
| **R9** | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.1:** The system generates a unique QR code associated to every ticket/visit booked from user application |

| | |
|---|---|
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R13** | The system notifies a user when his turn for entrance is approaching<br><br>**R.13.1:** The system sends a notification two minutes before entrance (in case of mobile app user and call center user) |
| **R16** | The system allows a mobile app user to get information about estimated time before being called for entrance (with relation to a booked reservation) |
| **R18** | The system is able to check availability for all the grocery shops that it manages |
| **R19** | The system, in case of mobile application, sends notifications basing on user's actual position and time remaining before his turn (with relation to a booked reservation) |
| **R28** | A user application is associated to a profile (personal account) |

| **UC4** | **Mobile app user books a visit** |
|---|---|
| **R4** | The system allows a mobile app user to book one or more visits |
| **R7** | The system allows a user to book slots for shopping for the next month from current date and time |
| **R9** | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.1:** The system generates a unique QR code associated to every ticket/visit booked from user application |
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R13** | The system notifies a user when his turn for entrance is approaching<br><br>**R.13.1:** The system sends a notification two minutes before entrance (in case of mobile app user and call center user) |
| **R16** | The system allows a mobile app user to get information about estimated time before being called for entrance (with relation to a booked reservation) |
| **R18** | The system is able to check availability for all the grocery shops that it manages |

| | |
|---|---|
| **R19** | The system, in case of mobile application, sends notifications basing on user's actual position and time remaining before his turn (with relation to a booked reservation) |
| **R22** | The system, in case of mobile application, allows to insert a list/categories of items (that user would like to buy) when booking a visit |
| **R24** | The system manages available slots depending on statistics built on specific user's habits (usual time spent in the grocery shop, items/categories of items normally purchased) |
| **R28** | A user application is associated to a profile (personal account) |

| **UC5** | **Mobile app user receives suggestions after failure in booking a visit** |
|---|---|
| **R4** | The system allows a mobile app user to book one or more visits |
| **R7** | The system allows a user to book slots for shopping for the next month from current date and time |
| **R15.1** | The system does not allow a person to book a reservation if the grocery shop's maximum capacity has been reached for the selected slot(s) |
| **R18** | The system is able to check availability for all the grocery shops that it manages |
| **R21** | The system is able to build statistics basing on time spent in the grocery shop and items/categories of items purchased |
| **R23** | The system manages available slots depending on reservations and grocery shops' capacities |
| **R24** | The system manages available slots depending on statistics built on specific user's habits (usual time spent in the grocery shop, items/categories of items normally purchased) |
| **R28** | A user application is associated to a profile (personal account) |

| **UC6** | **Mobile app user deletes a ticket reservation** |
|---|---|
| **R8** | The system allows a user to delete a reservation |
| **R28** | A user application is associated to a profile (personal account) |

| UC7 | Mobile app user deletes a visit reservation |
|---|---|
| R8 | The system allows a user to delete a reservation |
| R28 | A user application is associated to a profile (personal account) |

| UC8 | Call center user gets a ticket |
|---|---|
| R2 | The system allows a call center user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his phone number) |
| R9 | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.2:** The system generates a unique numeric code associated to every ticket/visit booked calling the call center |
| R18 | The system is able to check availability for all the grocery shops that it manages |
| R26 | Call centers are open in the grocery shops’ opening hours |

| UC9 | Call center user books a visit |
|---|---|
| R5 | The system allows a call center user to book one or more visits |
| R7 | The system allows a user to book slots for shopping for the next month from current date and time |
| R9 | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.2:** The system generates a unique numeric code associated to every ticket/visit booked calling the call center |
| R18 | The system is able to check availability for all the grocery shops that it manages |
| R26 | Call centers are open in the grocery shop’s opening hours |

| UC10 | Call center user deletes a reservation |
|---|---|
| R8 | The system allows a user to delete a reservation |
| R26 | Call centers are open in the grocery shop’s opening hours |

| UC11 | Ticket machine user gets a ticket |
|---|---|
| R3 | The system allows a ticket machine user to get a ticket for the corresponding (implicitly) grocery shop |
| R9 | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.3:** The system generates a unique QR code associated to every ticket/visit booked from a ticket machine |
| R10 | The system generates a unique numeric code for the position in the waiting "queue" associated to every ticket/visit booked from a ticket machine |
| R18 | The system is able to check availability for all the grocery shops that it manages |
| R27 | Ticket machines are available for booking in the grocery shop's opening hours |

| UC12 | Ticket machine user books a visit |
|---|---|
| R6 | The system allows a ticket machine user to book a visit only if no slots for tickets are available in the current day for the selected (implicitly) grocery shop |
| REQ | The system allows the user to book slots for shopping for the next month from current date and time |
| R9 | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.3:** The system generates a unique QR code associated to every ticket/visit booked from a ticket machine |
| R10 | The system generates a unique numeric code for the position in the waiting "queue" associated to every ticket/visit booked from a ticket machine |
| R18 | The system is able to check availability for all the grocery shops that it manages |
| R27 | Ticket machines are available for booking in the grocery shop's opening hours |

| UC13 | **Ticket machine user deletes a reservation** |
|---|---|
| **R8** | The system allows a user to delete a reservation |
| **R27** | Ticket machines are available for booking in the grocery shop's opening hours |

| UC14 | **User enters the grocery shop** |
|---|---|
| **R9** | The system generates a unique code for entrance associated to every ticket/visit |
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R12** | The system allows a customer to have the possibility to enter only when it is his turn and for the next ten minutes |
| **R13** | The system notifies a user when his turn for entrance is approaching |
| **R14** | The system does not notify a user for entrance if the grocery shop's maximum capacity has been reached |
| **R15** | The system does not allow a person to enter if the grocery shop's maximum capacity has been reached |

| UC15 | **User pays and exits the grocery shop** |
|---|---|
| **R9** | The system generates a unique code for entrance associated to every ticket/visit |
| **R20** | The system registers time spent inside the supermarket for every customer |

### 3.B.6 Goals mapping on requirements and domain assumptions

| G1 | **Grant social distance outside the grocery store** * |
|---|---|
| **DA5** | People at the grocery shop's entrance respect social distance as imposed by safety rules of the country/region (in case a user's turn has come, but he sees another customer at the code reader at the entrance) |
| **DA6** | Only a very little part of customers uses ticket machines for |

| | |
|---|---|
| | reservations, because mobile application and call center guarantee a more comfortable service accessible to almost everyone (people with a smartphone/tablet use the application, people without it can call and book) |
| **DA7** | GPS provides the exact location with an error of 10 metres at most |
| **R1** | The system allows a mobile app user to get a ticket only when no other ticket is "pending" (booked but not called yet) for his profile |
| **R2** | The system allows a call center user to get a ticket only when no other ticket is "pending" (booked but not called yet) for his phone number |
| **R3** | The system allows a ticket machine user to get a ticket for the corresponding (implicitly) grocery shop |
| **R4** | The system allows a mobile app user to book one or more visits |
| **R5** | The system allows a call center user to book one or more visits |
| **R6** | The system allows a ticket machine user to book a visit only if no slots for tickets are available in the current day for the selected (implicitly) grocery shop |
| **R8** | The system allows a user to delete a reservation |
| **R12** | The system allows a customer to have the possibility to enter only when it is his turn and for the next ten minutes |
| **R13.1** | The system sends a notification two minutes before entrance (in case of mobile app user and call center user) |
| **R16** | The system allows a mobile app user to get information about estimated time before being called for entrance (with relation to a booked reservation) |
| **R19** | The system, in case of mobile application, sends notifications basing on user's actual position and time remaining before his turn (with relation to a booked reservation) |
| **R26** | Call centers are open in the grocery shops' opening hours |
| **R27** | Ticket machines are available for booking in the grocery shop's opening hours |
| **R28** | A user application is associated to a profile (personal account) |

** the same mapping is valid also for G1.1 (G1's subgoal):* ***Avoid physical lining up outside the grocery store***

| | |
|---|---|
| **G2** | Manage entrances in the grocery store |
| **DA2** | Only one person enters the grocery shop for every ticket |
| **DA3** | Only one person enters the grocery shop for every visit |
| **R1** | The system allows a mobile app user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his profile) |
| **R2** | The system allows a call center user to get a ticket (only when no other ticket is “pending” - booked but not called yet - for his phone number) |
| **R3** | The system allows a ticket machine user to get a ticket for the corresponding (implicitly) grocery shop |
| **R4** | The system allows a mobile app user to book one or more visits |
| **R5** | The system allows a call center user to book one or more visits |
| **R6** | The system allows a ticket machine user to book a visit only if no slots for tickets are available in the current day for the selected (implicitly) grocery shop |
| **R7** | The system allows a user to book slots for shopping for the next month from current date and time |
| **R8** | The system allows a user to delete a reservation |
| **R9** | The system generates a unique code for entrance associated to every ticket/visit<br><br>**R.9.1:** The system generates a unique QR code associated to every ticket/visit booked from user application<br>**R.9.2:** The system generates a unique numeric code associated to every ticket/visit booked calling the call center<br>**R.9.3:** The system generates a unique QR code associated to every ticket/visit booked from a ticket machine |
| **R10** | The system generates a unique numeric code for the position in the waiting “queue” associated to every ticket/visit booked from a ticket machine |
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R12** | The system allows a customer to have the possibility to enter only when it is his turn and for the next ten minutes |
| **R13** | The system notifies a user when his turn for entrance is approaching<br><br>**R.13.1:** The system sends a notification two minutes before entrance (in case of mobile app user and call center user) |

| | |
|---|---|
| | **R.13.2:** The system shows on a display the waiting “queue” numeric code associated to a ticket machine reservation when the latter’s turn comes |
| **R14** | The system does not notify a user for entrance if the grocery shop's maximum capacity has been reached |
| **R15** | The system does not allow a person to enter if the grocery shop's maximum capacity has been reached<br><br>**R15.1:** The system does not allow a person to book a reservation if the grocery shop's maximum capacity has been reached for the selected slot(s) |
| **R16** | The system allows a mobile app user to get information about estimated time before being called for entrance (with relation to a booked reservation) |
| **R17** | The system allows a ticket machine user to get information about estimated time before being called for entrance (with relation to a booked reservation), through a display in the grocery shop’s proximity |
| **R19** | The system, in case of mobile application, sends notifications basing on user's actual position and time remaining before his turn (with relation to a booked reservation) |
| **R26** | Call centers are open in the grocery shops’ opening hours |
| **R27** | Ticket machines are available for booking in the grocery shop’s opening hours |
| **R28** | A user application is associated to a profile (personal account) |

| | |
|---|---|
| **G3** | Avoid crowds (too many people) inside the grocery store (at the same time) |
| **DA1** | The grocery shop has a maximum capacity for people inside |
| **DA2** | Only one person enters the grocery shop for every ticket |
| **DA3** | Only one person enters the grocery shop for every visit |
| **DA4** | People inside the grocery shop respect social distance as imposed by safety rules of the country/region |
| **R11** | The system allows for entrance only people with a reservation (ticket/visit) in a proper state |
| **R15** | The system does not allow a person to enter if the grocery shop's maximum capacity has been reached |

| | |
|---|---|
| **R20** | The system registers time spent inside the supermarket for every customer |
| **R21** | The system is able to build statistics basing on time spent in the grocery shop and items/categories of items purchased |
| **R22** | The system, in case of mobile application, allows to insert a list/categories of items (that user would like to buy) when booking a visit |
| **R23** | The system manages available slots depending on reservations and grocery shops' capacities |
| **R24** | The system manages available slots depending on statistics built on specific user's habits (usual time spent in the grocery shop, items/categories of items normally purchased) |
| **R28** | A user application is associated to a profile (personal account) |

## 3.C Performance Requirements

The system should be able to answer a ticket/visit reservation request in less than 10 seconds. In case there is no availability for the requested reservation, it should provide appropriate suggestions in a reasonable time (together with the requested answer or at most 3 seconds later). Moreover, the system should delete a reservation (when requested) in less than 5 seconds.
CLup, after monitoring client's visit time and duration habits, should be able to extract information from data every 3 days, in order to use it to build statistics and provide suggestions.
The system should monitor real time GPS positions of users who have a reservation in the next hours, in order to be able to send notifications according to requirements.

## 3.D Design Constraints

### 3.D.1 Standards compliance

There is no particular necessity to deviate from usual web-standards suggested by the World Wide Web Consortium (W3C); for this reason, the development of the system should follow W3C recommendations.
Any further compliance, if needed, will be specified during the design analysis.

### 3.D.2 Hardware limitations

The mobile application can be installed only on a smartphone/tablet equipped with a GPS system and the possibility to activate mobile data.

## 3.E Software System Attributes

### 3.E.1 Reliability

*Customers Line-up* should be available 24/7 in order to book visits and manage reservations whenever users want. However, the crucial time in which the system should work coincides with supermarkets' opening hours: for this reason, 4-6 hours of maintenance do not represent a problem, if programmed outside opening hours.

### 3.E.2 Availability

*CLup* does not have a very critical nature (it is not connected with emergency situations); however, in case the system is down during opening hours, supermarket's business becomes much more complicated: entrances have to be managed directly from its employees and ticket reservations become impossible; the only thing employees could do is to guarantee the entrance to an appropriate number of people that are waiting outside, without realizing any other feature that the software system guarantees.
For these reasons, 95% availability of the system should be guaranteed.

### 3.E.3 Security

The system uses HTTPS for a secure communication between users and the server. Since a customer can have more than a visit booked at the same time (for different days), there should be a limit to the number of bookings a user can have simultaneously, in order to guarantee the correct functioning of the system. Moreover, all passwords are encrypted and, in case of recovery, they are not sent in clear.
Possible additional choices will be discussed in the *Design Document*.

### 3.E.4 Maintainability

Code should follow good software engineering practices and be properly commented. In addition, the use of proper design patterns, as suggested in the *Design Document*, is mandatory.

### 3.E.5 Portability

*Customers Line-up's* frontend is composed mainly by a mobile application: to ensure an high grade of portability between Android and iOS an hybrid framework should be used to build the frontend, so to be able to share the most of the code between the two platforms.

# 4 Formal Analysis using Alloy

## 4.A Main objectives

The main objective of using the Alloy specification tool in the CLup project is to check the correctness and consistency of all the constraints, requirements and assumptions that have been made during the previous section of this document. Another reason that drives Alloy modeling is to show possible worlds and therefore exemplify some situations that could be instances of the system in a real world scenario.

## 4.B Signatures

```
abstract sig User{
     userStatus: UserStatus
}


// Here we enlist the three kind of user that can be found in
// CLup, see User Characteristics at section 2.C
sig AppUser extends User{}
sig TicketMachineUser extends User{}
```

```
sig CallCenterUser extends User{}

sig Date{}

sig Time{
     hour: Int,
     min: Int
}

// queueNumber is the number present in the display and shown
// only to TicketMachine users
// ticketMachineReservation is the TM used for the reservation
abstract sig Reservation{
     code: Code,
     date: Date,
     time: Time,
     res: User -> GroceryShop,
     queueNumber: lone Code,
     ticketMachineReservation: lone TicketMachine
}

sig Visit extends Reservation {}
sig Ticket extends Reservation {}


sig GroceryShop {
     capacity: Int,
     openingHour: Time,
     closingHour: Time
}{
     // Shops are open for a reasonable time
     openingHour.hour < closingHour.hour
}

sig Code {
     id: Int
}

abstract sig UserStatus{}
one sig BLOCKED extends UserStatus{}
one sig NORMAL extends UserStatus{}

abstract sig Status{}
one sig PLANNED extends Status{}  // reserved, still not used
one sig EXPIRED extends Status{}  // reservation already used
one sig ACTIVE extends Status{}   // user is inside the shop

sig ReservationStatus{
     resStatus: Reservation -> Status
}{
     #resStatus = 1
```

```
}

sig TicketMachine {
     shop: GroceryShop
}
```

## 4.C Functions

```
// returns all the reservations for a given groceryShop
fun getShopReservations [g : GroceryShop]: set Reservation {
     (res.g).User
}

// returns all the ACTIVE reservations given a set of
// Reservations as parameter
fun getActiveReservations [reservations: set Reservation]:
set Reservation {
     ((resStatus.ACTIVE).reservations).(resStatus.ACTIVE)
}



// returns all the PLANNED reservations given a set of
// Reservations as parameter
fun getPlannedReservations [reservations: set Reservation]:
set Reservation {
     ((resStatus.PLANNED).reservations).(resStatus.PLANNED)
}

// returns all the Tickets associated to the User given as
// parameter
fun getUserTickets [u: User]: set Ticket {
     (res.GroceryShop).u
}

// returns all the Reservations associated to the User given
// as parameter
fun getUserReservations [u: User]: set Reservation {
     (res.GroceryShop).u
}

// returns all the PLANNED tickets associated given a set of
// Tickets as parameter
fun getPlannedTickets [tickets: set Ticket]: set Ticket {
     ((resStatus.PLANNED).tickets).(resStatus.PLANNED)
}
```

## 4.D Facts

```
// The number of active reservations is always less than the
// maximum capacity of the shop
fact activeReservationsLowerThanCapacity {
     all g: GroceryShop |
       (let resInG = getShopReservations[g] |
          (let resActiveInG = getActiveReservations[resInG] |
                    #resActiveInG <= g.capacity
               )
          )
}

// The number of PLANNED tickets for every AppUser is 0 or 1
fact justATicketPerAppUser {
     all au: AppUser |
     (let userTicket = getUserTickets[au] |
      (let userPlannedTicket = getPlannedTickets[userTicket] |
                   #userPlannedTicket <= 1
               )
          )
}



// The number of PLANNED tickets for every CallCenterUser is 0
// or 1
fact justATicketPerCallCenterUser {
     all ccu: CallCenterUser |
          (let userTicket = getUserTickets[ccu] |
               (let             userPlannedTicket             =
getPlannedTickets[userTicket] |
                   #userPlannedTicket <= 1
               )
          )
}

// It is not possible that two different tickets have same
// code
fact uniqueCodeForReservation {
     no disj r1, r2: Reservation | r1.code = r2.code
}

// The queueNumber is unique for a shop in a day
fact uniqueQueueNumber {
     no disj r1, r2: Reservation |
          r1.date = r2.date and
          r1.queueNumber = r2.queueNumber and
          User.(r1.res) = User.(r2.res)
}
```

```
// Blocked users cannot have any ACTIVE reservation
fact noMoreTicketForBlockedUser {
     no u: User | u.userStatus = BLOCKED and
          (#getActiveReservations[getUserReservations[u]] > 0)
}

// Blocked users cannot have any PLANNED reservation
fact noMoreTicketForBlockedUser {
     no u: User | u.userStatus = BLOCKED and
     (#getPlannedReservations[getUserReservations[u]] > 0)
}

// TicketMachine users cannot be blocked
fact noBlockedTicketMachineUsers {
     no tmu: TicketMachineUser | tmu.userStatus = BLOCKED
}


// TicketMachine reservations are valid only for the same shop
of the TicketMachine
fact sameShopForTicketMachineReservations {
     all tmu: TicketMachineUser, r : Reservation |
          tmu.(r.res) = r.ticketMachineReservation.shop
}

// Reservations can be only for shops' opening hours
fact reserveForOpeningHours {
     all r: Reservation |
     (let g = User.(r.res) |
     ((r.time.hour > g.openingHour.hour) and (r.time.hour <
g.closingHour.hour))
               or
((r.time.hour = g.openingHour.hour) and (r.time.min >=
g.openingHour.min))
               or
((r.time.hour = g.closingHour.hour) and (r.time.min <
g.closingHour.min)))

}

// Every GroceryShop has at least a TicketMachine
fact atLeastATicketMachineForEachShop {
     no g: GroceryShop |
          no tm: TicketMachine |
               tm.shop = g
}

// A single user cannot have more than one active reservation
fact singleUserCannotHaveMoreThanOneActiveReservation {
     no disj r1, r2: Reservation |
```

```
          r1.(ReservationStatus.resStatus) = ACTIVE and
          r2.(ReservationStatus.resStatus) = ACTIVE and
          (r1.res).GroceryShop = (r2.res).GroceryShop
}

// A single user cannot have more than 1 active or planned
reservation for a single day
fact noMultiplePlannedOrActiveReservationForAUserInSingleDay {
     all disj r1, r2: Reservation |
  ((r1.date = r2.date) and
  ((r1.res).GroceryShop = (r2.res).GroceryShop))

     implies

  ((r1.(ReservationStatus.resStatus) = EXPIRED)
or (r2.(ReservationStatus.resStatus) = EXPIRED))

}

// Every Reservation has only a single Status associated to it
fact everyReservationHasJustOneStatus {
     all r: Reservation |
          #r.(ReservationStatus.resStatus) = 1
}
```

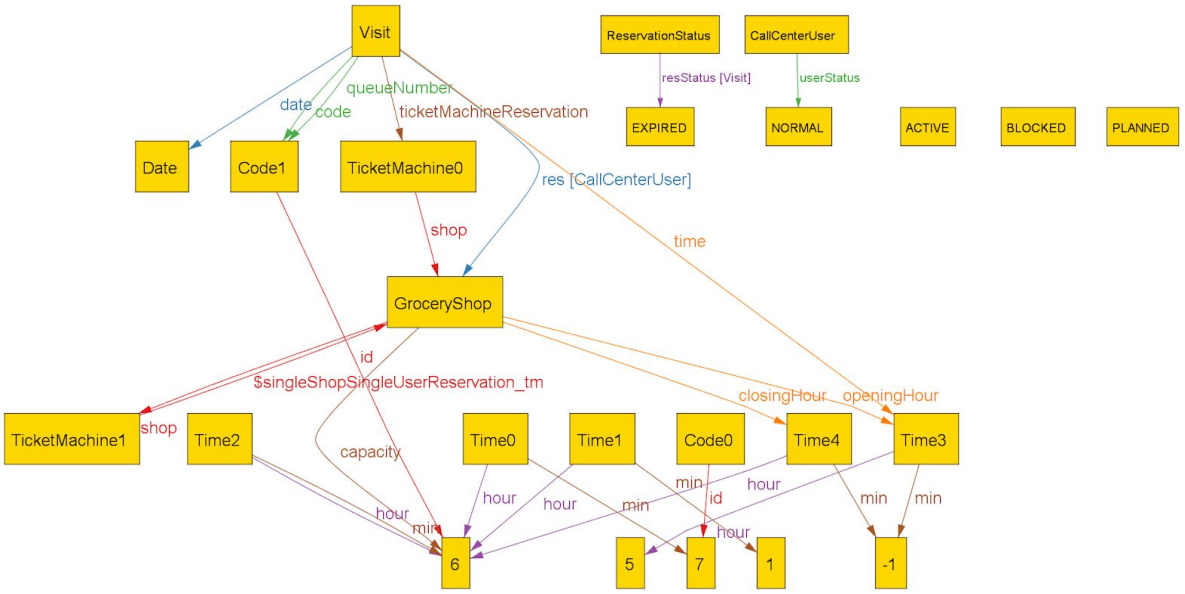
## 4.E Predicates and generated worlds

Alloy predicates have been used to show meaningful instances of the system and to demonstrate the constraints that this model enforces. There are also examples that are useful to have a better understanding of situations which cannot happen, given the requirements, goals, and domain assumptions made on the CLup system.

### First example

```
// This predicate shows a possible model for a single
// GroceryShop, single User and single Reservation scenario
pred singleShopSingleUserReservation {
     #GroceryShop = 1
     #User = 1
     #Reservation = 1
}
```

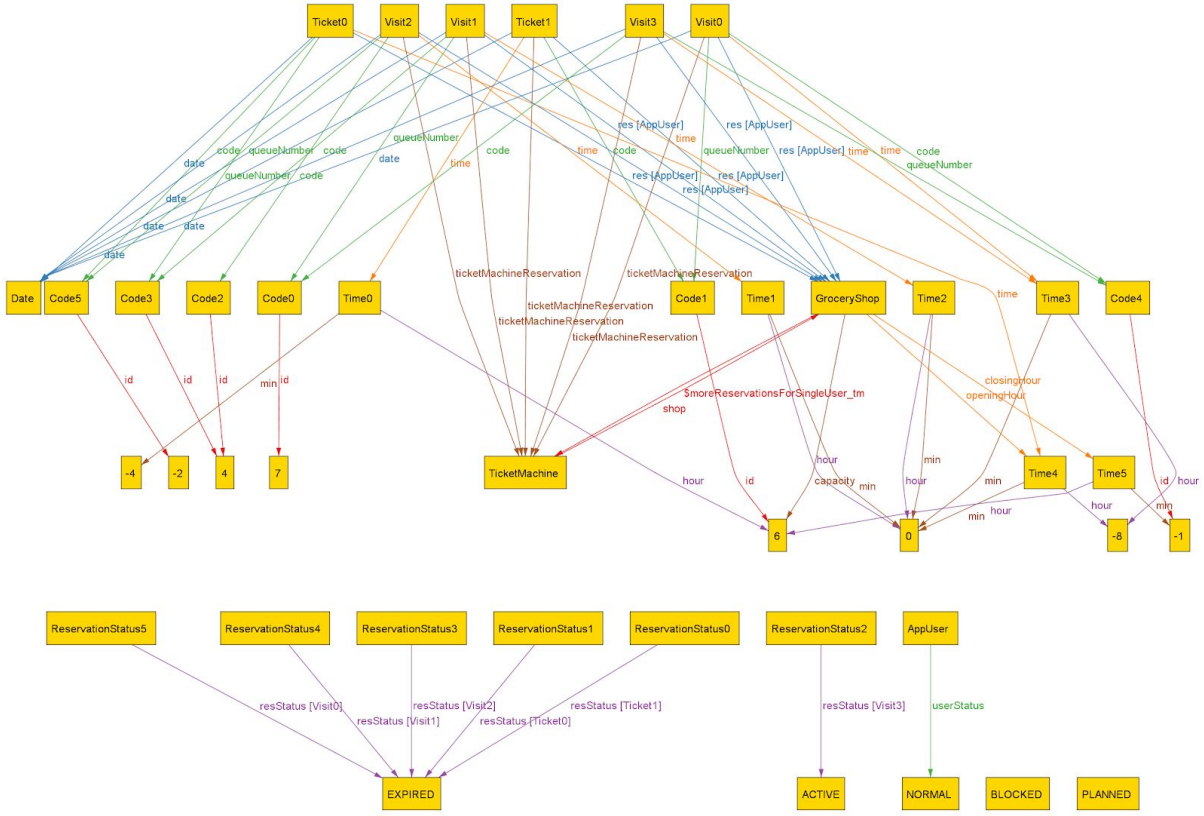
Here the very first and simple example can be visualized, we have one User, one Reservation and one Grocery Shop. In particular, the reservation involved is a Visit, booked by a call center User and already expired (the customer has already been to the shop).




## Multiple Reservations for a single User

```
// More reservations for a single User: just ONE can be
// ACTIVE
pred moreReservationsForSingleUser {
      #Reservation = 6
      #AppUser = 1
      #TicketMachineUser = 0
      #CallCenterUser = 0
      #Date = 1
      #TicketMachine = 1
      #GroceryShop = 1
}
```

This generated world presents the possibilities provided to the user regarding a single day; a customer can have an unlimited number of reservations (without exceeding the threshold set to be blocked), but only one single reservation can be ACTIVE at a certain moment.




## Exceeding Grocery Shop maximum capacity

```
// It is not possible to have more ACTIVE reservations than
// #capacity inside the shop (ACTIVE reservations)
pred moreActiveReservationThanShopCapacity (
u1: User, u2: User, r1: Reservation, r2: Reservation,
rStatus1: ReservationStatus, rStatus2: ReservationStatus,
g: GroceryShop )
{
      g.capacity = 1
      u1.(r1.res) = g
      u2.(r2.res) = g
      r1.(rStatus1.resStatus) = ACTIVE
      r2.(rStatus2.resStatus) = ACTIVE
      r1 != r2
}
```

Given the facts of our model, if this predicate is executed it's possible to observe

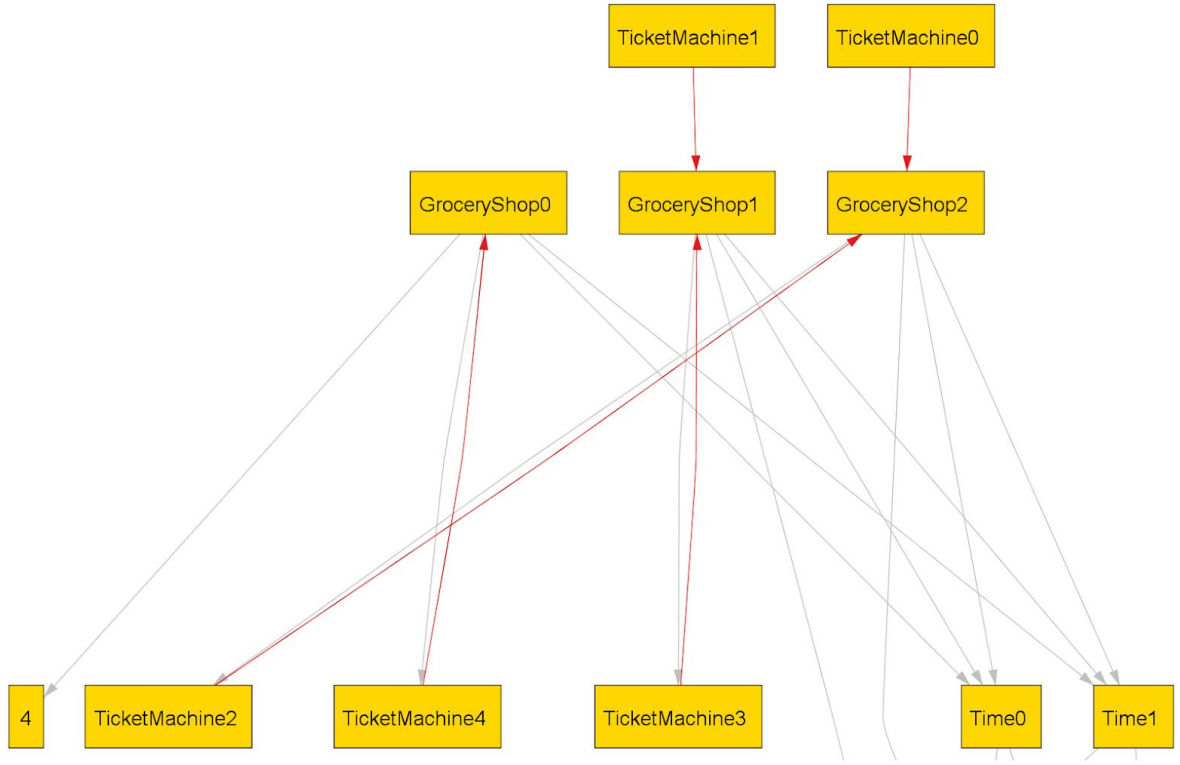
that no instance is found. This proves the fact that in our model it is not possible to have, for every GroceryShop, a number of active reservations which is greater than the shop maximum capacity. In fact, if this predicate is changed by increasing the capacity to a value greater than or equal to 2, instances are indeed found.

### Every Ticket Machine belongs to one Grocery Shop

```
// This predicate shows that a TicketMachine belongs to one
// GroceryShop
pred eachTicketMachineBelongToSingleGroceryShop {
      #GroceryShop = 3
      #TicketMachine = 5
      #Ticket = 0
      #Visit = 0
      #Code = 0
      #User = 0
      #Time = 2
      #Date = 0
      #Reservation = 0
}
```

The world generated by running this predicate highlights the fact that in our model every shop can be associated with many ticket machines (and this is reasonable, given the fact that more than one ticket machine could be necessary in order to

avoid crowds outside shops), but every ticket machine must be linked to one and only one shop.

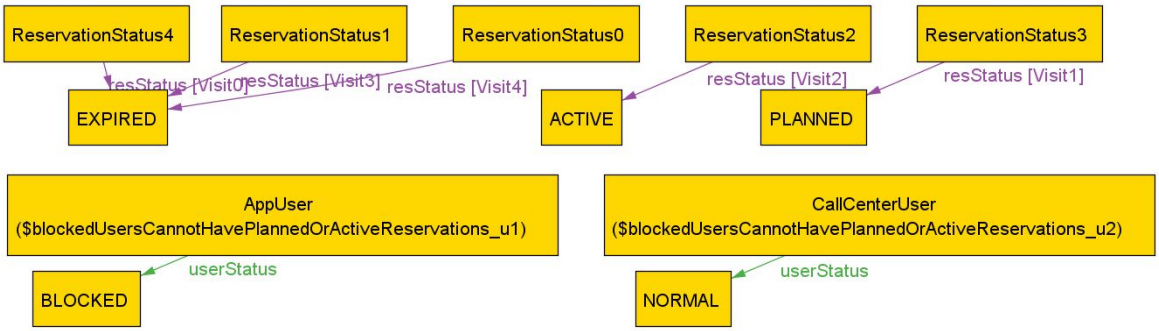



## Blocked users cannot have ACTIVE or PLANNED reservations

```
// Blocked users cannot have Planned or Active Reservations
pred blockedUsersCannotHavePlannedOrActiveReservations (u1,
u2: User, v1, v2, v3, v4, v5: Visit)
{
      u1.userStatus = BLOCKED
      u2.userStatus = BLOCKED
      v1.(ReservationStatus.resStatus) = EXPIRED
      v2.(ReservationStatus.resStatus) = EXPIRED
      v3.(ReservationStatus.resStatus) = ACTIVE
      v4.(ReservationStatus.resStatus) = EXPIRED
      v5.(ReservationStatus.resStatus) = PLANNED
      #Visit = 5
      #Ticket = 0
      #GroceryShop = 1
      #TicketMachine = 1
      #Time = 2
      #Code = 5
```

```
    #User = 2
    #Date = 2
    v1 != v2 and v3 != v4
    u1 != u2
}
```

This predicate has been created in order to show the impossibility for blocked users to have PLANNED or ACTIVE reservations. In fact, no instances are found and the predicate is not consistent given the model's facts. If the status of one of the users becomes NORMAL, then instances are indeed found, and one of the possible generated worlds is shown below.






The only two visits which are not expired are Visit1 and Visit2, and they are both associated with the user that is not blocked: the CallCenterUser. This does not violate any constraint since they are reservations relative to different dates.