POLITECNICO MILANO 1863

# Requirement Analysis and Specification Document 2020-2021

Alessandro Polidori (Codice persona 10573078)
Olimpia Rivera (Codice persona 10617517)

## Table of Contents

# 1 Introduction

## 1.1 Purpose

This document provides an analysis of the CLup system in terms of assumptions, functional and non functional requirements needed to fulfill its main goals. It describes the domain in which the system will be deployed by presenting relevant scenarios and use cases and it highlights the software's limits and constraints.

The document is addressed to all the stakeholders affected by the software and is meant to be used by developers in order to realize a system that meets the purpose for which it was intended.

The CLup system is designed to be an easy-to-use software application, meant to help both common users and store managers to face some significant difficulties due to the coronavirus emergency. In particular, the system wants to prevent people from standing in long lines outside stores and to offer them the possibility to organize in advance visits to stores. Indeed, thanks to the CLup system, people would have the chance to virtually join stores' queues and approach the stores only when their turn number is close to being called. In addiction, the system allows users to make reservations to visit stores during their preferred dates and time slots. Given the emergency situation we are living, many stores were forced to restrict accesses in order to avoid crowds inside. For this reason, store managers as well would benefit from the CLup system, because it would give them the opportunity to keep track of the the number of people entering their stores. All these services offered by the CLup system have the purpose of minimizing risks and helping people respecting the restrictions imposed by the virus emergency.

## 1.2 Scope

### 1.2.1 Description of the given problem

As stated before, the CLup system is designed to regulate accesses to stores and manage lines in real time in order to respect restrictions imposed by the virus emergency and avoid crowds and long lines. In particular, the software is offered to both stores managers to monitor the influx of people in their buildings and to common users, allowing them to virtually "line-up" from home and book visits to the stores.

The main functionalities offered by CLup are the following:

- Basic service: allows stores' customers to line up from home and to approach the store only when their turn is about to arrive. In order for this lining up mechanism to work effectively, the software generates an

estimation of the waiting time and alerts users when is the moment to reach the store, taking into account the time they need to get to the shop from the place they are located. The estimated waiting time is calculated considering the number of people in the virtual line and the time spent inside the store by the costumers that are already shopping. The system is also able to optimize or increase the waiting times of users in the queue when needed. In addiction, when customers enter and exit the stores, a QR code generated by the application is scanned, allowing store managers to monitor the influx of people.

- Advanced service: allows users to book visits to stores in their preferred date and time slot. When booking, users provide the categories of the items that they intend to purchase, so that the system will be able to manage visits in a finer way taking into account the store's departments that customers are going to occupy.

When customers line up or make a reservation they are asked to provide the expected duration of the visit. Alternatively, they can let the software itself infer it (this works only for long-term customers by analyzing the customer's previous visits).
Be aware that the functions described above will be further detailed in the next sections of the document.

### 1.2.2 World Phenomena

| | |
|---|---|
| **WP1** | Stores' customers either owning or not a smartphone |
| **WP2** | Customers approaching the store |
| **WP3** | Customers arriving too early at the shop for unexpected reasons |
| **WP4** | Customers doing the shopping |
| **WP5** | Customers walking away from store |

### 1.2.3 Shared Phenomena

| | |
|---|---|
| **SP1** | User enters the store (QR code scanned) - World controlled |
| **SP2** | User gets in line - World controlled |
| **SP3** | System shows number and estimated waiting time - Machine controlled |
| **SP4** | System generated QR code and line number - Machine controlled |
| **SP5** | System notifies the user to reach the store - Machine controlled |
| **SP6** | System updates waiting time - Machine controlled |
| **SP7** | User doesn't arrive in time - World controlled |
| **SP8** | User cancels reservation (either slot or line) - World controlled |
| **SP9** | User books a slot - World controlled |
| **SP10** | User indicates expected duration of visit or requests system to infer it - World controlled |
| **SP11** | User makes list of items - World controlled |
| **SP12** | User exits the store (QR code scanned) - World controlled |

### 1.2.4 Goals

| | |
|---|---|
| **G1** | Provide customers a safe environment in the stores |
| **G2** | Allow users to line up at the stores from home |
| **G3** | Avoid formation of long lines outside stores |
| **G4** | Allow store managers to monitor the influx of people in their buildings |
| **G5** | Allow users to book visits to stores |
| **G6** | Allow users to arrive on time for their turns |

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

- Product type: a set of similar products. Usually, items with the same product type are located in the same store's department.
- Smart Turnstile: a turnstile equipped with a QR codes scanner.
- Store's department: a specific section of the store, containing only a subset of all the product types

- Store manager: the person responsible for the store and for the interactions between the store and the system. He can be helped by his employees.

- Waiting time: an estimation of the time needed to the queue to get to a specific turn number.

- Reservation: booked visit to a store.

### 1.3.2 Acronyms

- GPS: Global Positioning System

- API: Application Programming Interface

### 1.3.3 Abbreviations

- WPn: n-th World Phenomena

- SPn: n-th Shared Phenomena

- Gn: n-th Goal

- Dn: n-th Domain Assumption

- Rn: n-th Functional Requirement

## 1.4 Overview

The RASD document is structured in the following 5 chapters:

- **Chapter 1** describes the document's purpose and brief identification of the context in which the application is going to work given its main functionalities. It also provides lists of world phenomena, shared phenomena and goals that the system is supposed to achieve. Finally, useful specifications (definitions, acronyms, abbreviations) are included for a better understanding of the next sections of the document.

- **Chapter 2** gives an overall description of the project. The class diagram in the Product Perspective provides a conceptual overview of the main elements of the system and the state charts describe the evolution of relevant objects. In the Product Function section, the system's high level functionalities are further detailed and clarified. The expected type of actors that will interact with the system are listed in User Characteristics. Ultimately, chapter 2 describes the system's constraints and the domain properties assumed to hold in the world.

- **Chapter 3** describes the external interface requirements (user, hardware and software interfaces). Some scenarios are listed to clarify how the system works in real world situations. Both functional and non-functional requirements are described. The former are defined by several use cases and

their relatives sequence diagrams, while the latter are identified though performance requirements, design constraints and software system attributes.

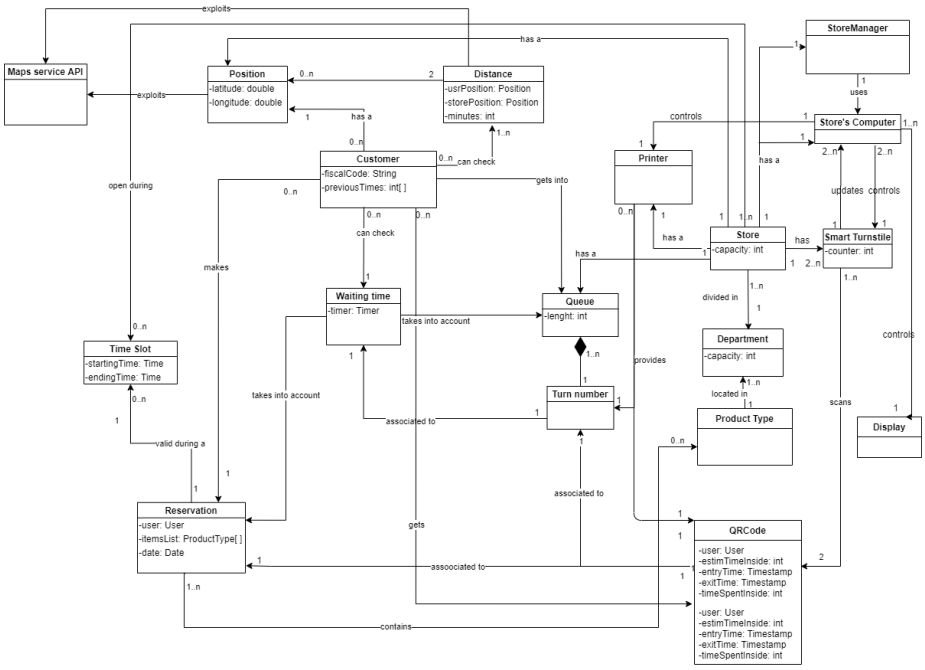
- **Chapter 4** includes the Alloy model of some relevant aspects and a brief discussion of its purpose.
- **Chapter 5** presents the effort spent by the group members while working on this project.
- **Chapter 6** includes the reference documents.

# 2 Overall Description

## 2.1 Product Perspective

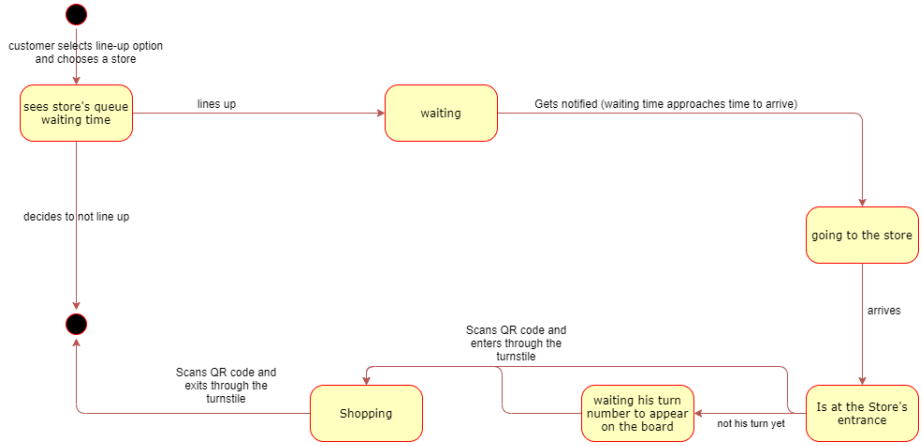
### 2.1.1 Class diagram

The class diagram below gives a high level representation of the system's domain. Classes contains only the necessary attributes to highlight the most important dynamics. **QR Code** is associated either to a **Turn Number** or to a **Reservation** and is provided with attributes required to associate the measured duration of the stay to the user. Reservation contains a list of **Product Types** and every product type is located in a specific store **Department**. A **Queue** is composed by a certain amount of **Turn numbers**. The **Distance** needs two **Positions** and can access to the time estimated by the **Maps Service API**. The **Store-Manager** is able to use the webApp on the **Store's Computer**,needed also to control the **Printer** and the **Display**. Moreover, **Customer** has a direct access to a list of the previous shopping times (calculated by the system using **QR Code** attributes).

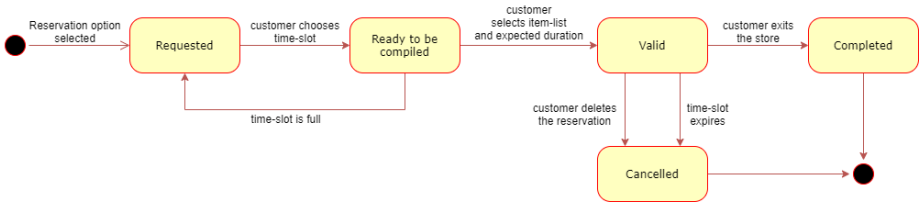

exploits
has a
StoreManager
Maps service API
Position
-latitude: double
-longitude: double
Distance
-usrPosition: Position
-storePosition: Position
-minutes: int
exploits
has a
uses
controls
Store's Computer
Customer
-fiscalCode: String
-previousTimes: int[ ]
can check
Printer
has a
open during
gets into
updates
controls
Store
-capacity: int
has
Smart Turnstile
-counter: int
can check
makes
has a
Waiting time
-timer: Timer
takes into account
Queue
-lenght: int
divided in
Department
-capacity: int
provides
Time Slot
-startingTime: Time
-endingTime: Time
takes into account
Turn number
located in
Product Type
scans
Display
controls
valid during a
associated to
associated to
Reservation
-user: User
-itemsList: ProductType[ ]
-date: Date
gets
QRCode
-user: User
-estimTimeInside: int
-entryTime: Timestamp
-exitTime: Timestamp
-timeSpentInside: int
-user: User
-estimTimeInside: int
-entryTime: Timestamp
-exitTime: Timestamp
-timeSpentInside: int
assocociated to
contains

### 2.1.2 Statechart diagrams

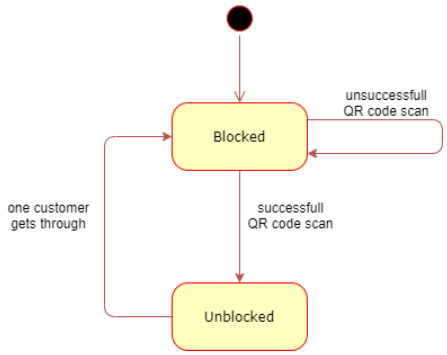
The diagrams below represent the states relative to some classes we wanted to analyse.




The first diagram shows the states relative to the **User** in the context of the line-up function.




The second diagram models the states in which a **Reservation** goes through in the context of the "booking a visit" function.




The last one is a simple diagram, showing the functioning of the **Smart Turnstiles** positioned at the stores' entrances.

## 2.2 Product Functions

The main functions that the system will provide are listed in the following section. The identified high level requirements will be further broken down and detailed in section 3, with respect to the previously recognized goals of the system.

### 2.2.1 Line up

As previously stated, the basic functionality allows users to virtually stand in line to enter the stores. Firstly, the user selects the store and chooses among the two available options: line up or book a visit. Whenever the choice is to line up, the system will show to the user the estimated waiting time to enter. At that point, if the user decides to actually get in the line, he/she will be asked to provide the expected duration of the visit and finally, he/she will receive the turn number and the QR code that has to be scanned when entering to the store. The user will be asked wether he/she will reach the store on foot or by car, so that the system will be able to calculate the user's expected time needed to reach the store (exploiting a Maps Service API). This time will be shown to the user while he/she is waiting, together with the waiting time (constantly updating). The user will be notified by the system when the time he/she needs to get to the store equals the time left for the user to enter. Once arrived to the store, the user might need to wait few minutes until his/her number appears on the queue board. Finally, the smart turnstile will scan the QR code and unlock, letting the user enter the store and start shopping.

### 2.2.2 Book a visit

As mentioned before, after choosing the store, the user also has the option to book a visit. He/she will select the preferred time slot and make a reservation whenever the slot is not already full. A QR code is generated and the user can

access to the store at any moment of the time slot, as long as he/she exits before the end of it. When making the reservation, the user is asked to provide the expected duration of the visit and to select the categories of items that he/she intends to purchase. This last information is used by the system to understand which store's department the customers are going to occupy in order to manage entrances in a finer way. In particular the system can either let more people in if the distribution is homogeneous inside the store or slow down the influx of people if customers are gathered in certain departments.

#### 2.2.3 Manage lines

The CLup system has to manage the lining up mechanism effectively. First of all, the waiting time assigned to customers that get in line, can change depending on several events. In particular, whenever a customer either gets out of the line or cancels a slot reservation, the system is able to decrease the waiting times of the users currently in line and optimize the influx of people. The waiting time can also decrease in the event that someone does not show up when his/her turn is arrived (there will be a fixed maximum delay tolerated by the system). Each user (for both the basic and advanced functionality) is asked to provide the expected amount of time they intend to spend inside the store, alternatively, users can choose to let the system infer that time (only if the system has enough data). Indeed, for each user, the system will collect and store informations about the duration of his/her visits and estimate an average. As a consequence, whenever a user's visit lasts longer or less than the estimated time, the waiting time of the users in line can be optimized (increased or decreased). Ultimately, the system is also able to handle people who do not have access to the required technology. The store manager (or one of his employees) will print a number and a QR code for those who have to physically get in line and the system will take them into account when scheduling the next entrances (they will be assigned a fixed time of stay in the shop).

#### 2.2.4 Keep track of accesses

The CLup system is also meant to be used by store managers to keep track of entrances in their buildings. This is possible thanks to the smart turnstiles that scan the QR codes. The system check the validity of the QR codes (not expired or false) and allow the turnstiles to unlock to let people in or out. Consequentially, a customer counter is updated so that stores managers can check whether their store's capacity is respected.

### 2.3 User Characteristics

The application is supposed to be used by the following actors:

1. **Registered user**: someone who downloaded the application on his/her device, registers to CLup and uses its services.

   Depending on the service used, the following distinction can be done:

- Line up: the user is someone that wants to enter the store as soon as possible and might not live too far from it. Indeed, he/she prefers to line up and is ready to move even if the waiting time is shortened.
- Book a visit: the user is someone who plans on visiting the store in advance and at a fixed time.

2. **Unregistered user**: someone who do not have access to the required technology and therefore is not registered to CLup services. The only way in which he/she interacts with the system is by physically retrieving a turn number and a QR code from the ticket dispenser located at the store. The system will take into account this actor when scheduling the next entrances

3. **Store manager**: someone who is responsible for managing a store and registers to CLup to monitor the influx of people in his/her building.

## 2.4 Assumptions and Dependancies

### 2.4.1 Domain assumptions

| | |
|---|---|
| **D1** | GPS provides the exact location with an error of 5 meters at most |
| **D2** | The time calculated from the Maps Service to reach the store is correct |
| **D3** | The user does not to arrive earlier than the indicated time |
| **D4** | The user follows the shortest path to the store, without making deviations and stops |
| **D5** | Only one person shows up at the store for each turn number |
| **D6** | Only one person shows up at the store for each booked visit |
| **D7** | The turnstiles unlock only if the system allows them |
| **D8** | One person at a time passes through the turnstiles every time a QR code is scanned |
| **D9** | The user does not change location while is waiting for his/her turn |
| **D10** | The user does not change the indicated means of transport |
| **D11** | The user respects slots' time boundaries |
| **D12** | The user respects social distancing inside the store |
| **D13** | The user visits only the departments related to the product types he/her specified |
| **D14** | The user only gets in the line of stores that are not too far from him/her (the time to arrive should never exceed the turn validity interval) |

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

The following mockups give an idea of how the application looks like on the user's smartphone. They show some screenshots of the most relevant interactions between the system and the registered users.

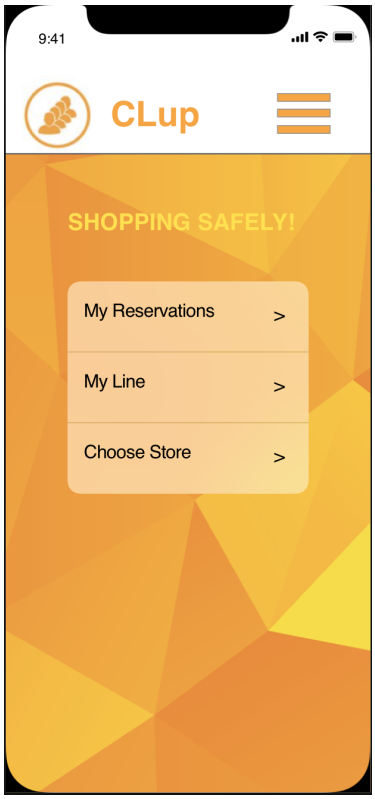



Figure 1: Home

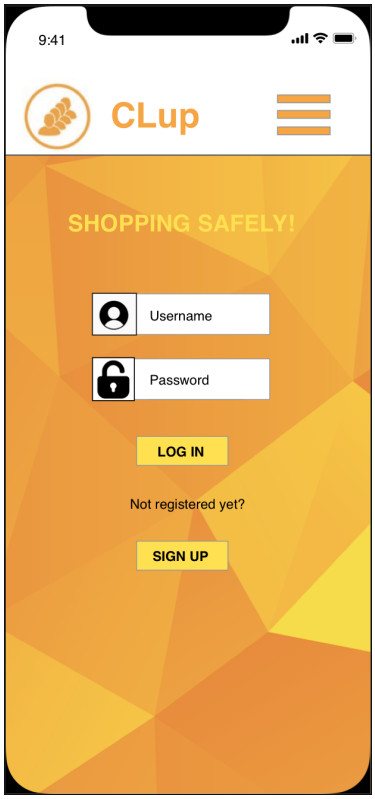



Figure 2: Login

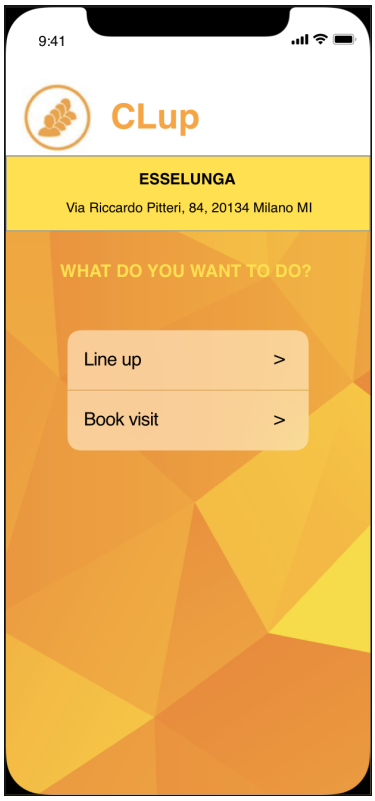



Figure 3: Functionality selection

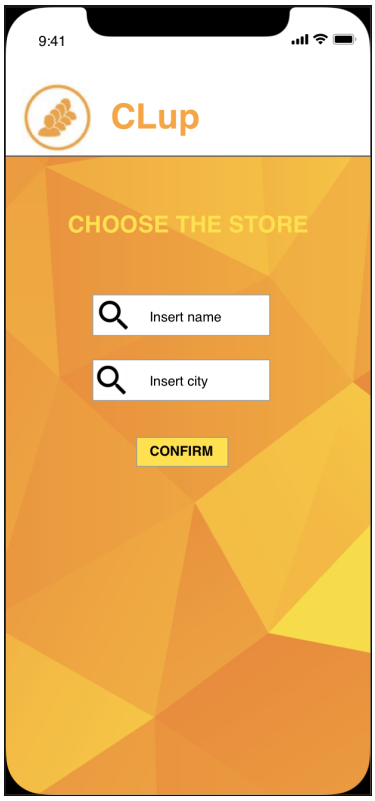



Figure 4: Store's name and city selection

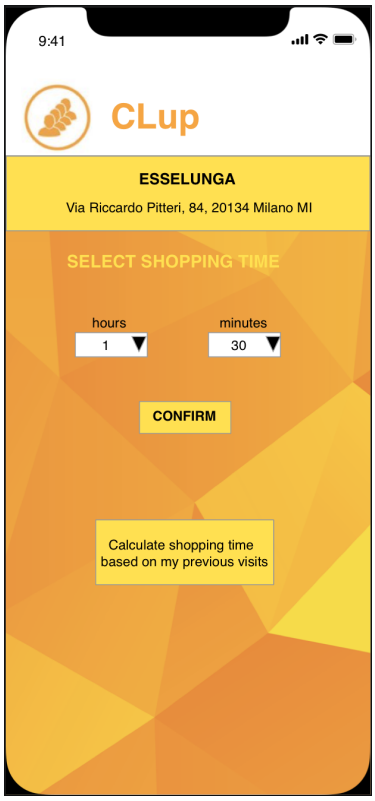



Figure 5: Shopping time selection

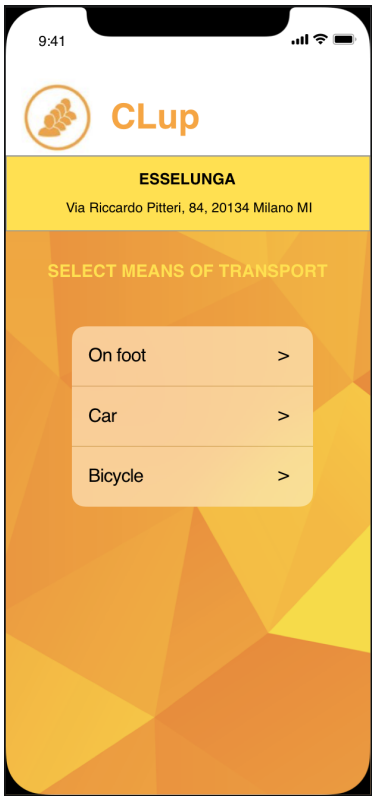



Figure 6: Means of transport selection


Figure 7: QR code

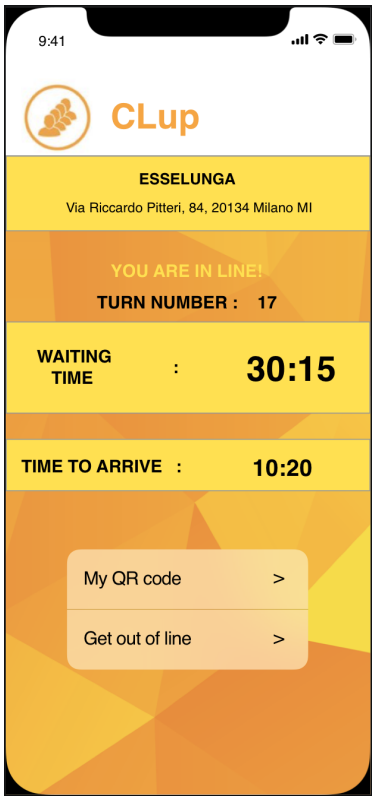



Figure 8: Line informations

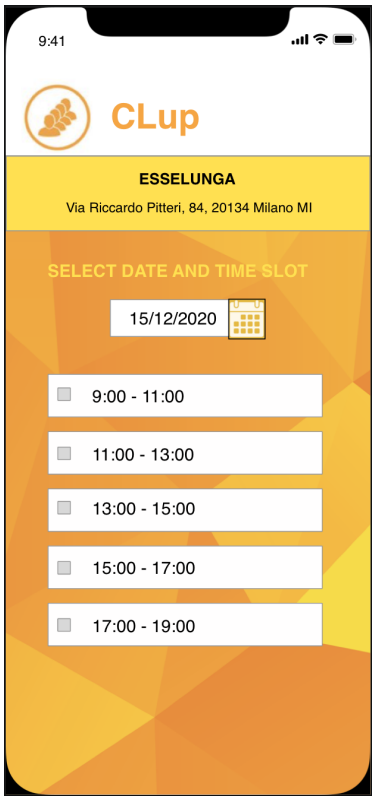



Figure 9: Timeslot and date selection

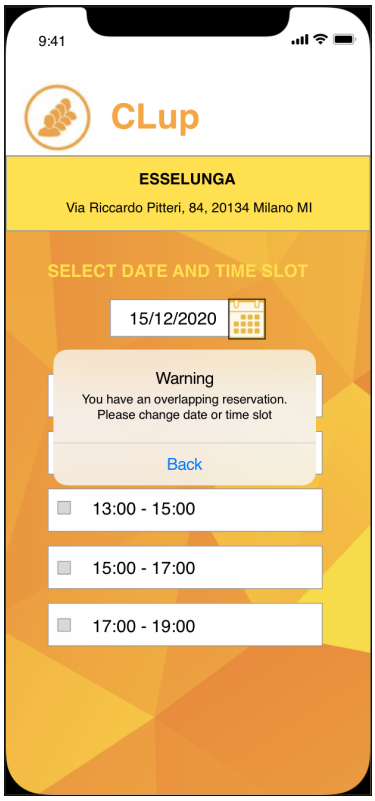



Figure 10: Overlapping reservations warning

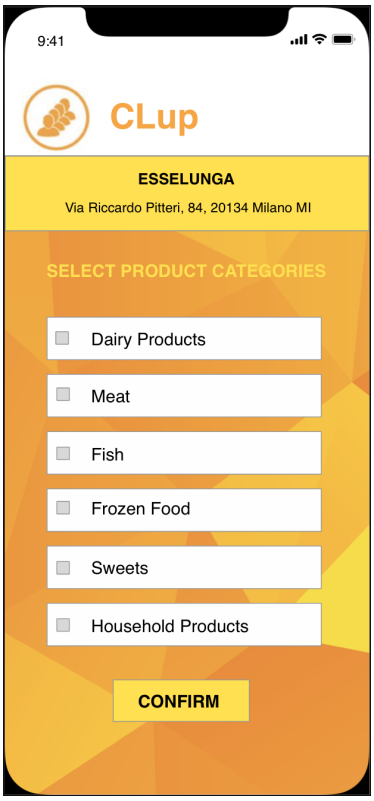



Figure 11: Product categories selection

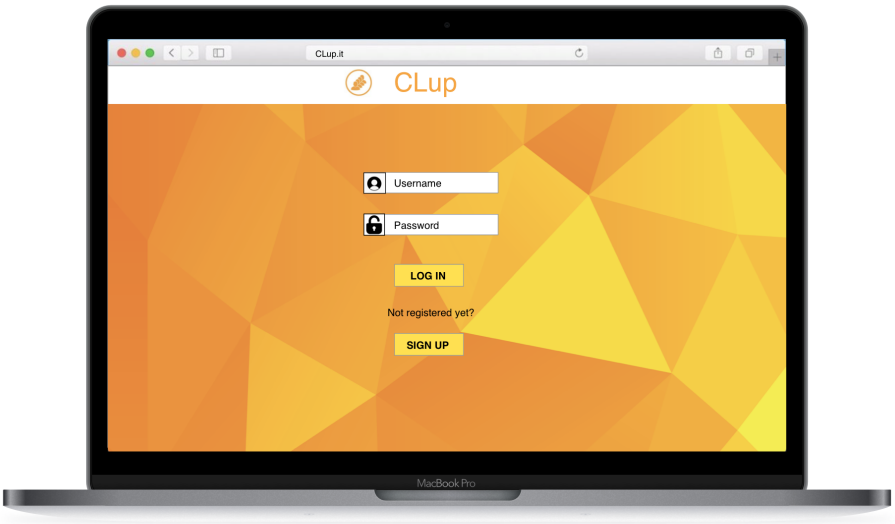



Figure 12: WebApp Login

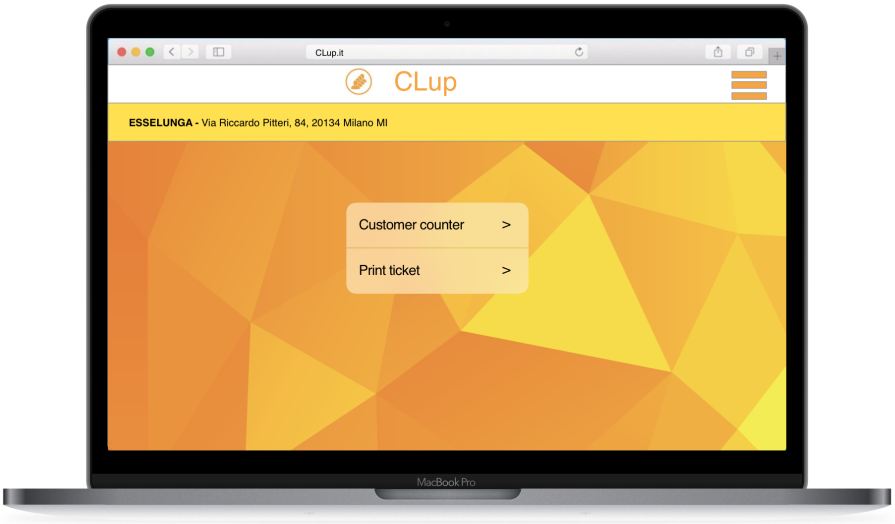



Figure 13: WebApp Home

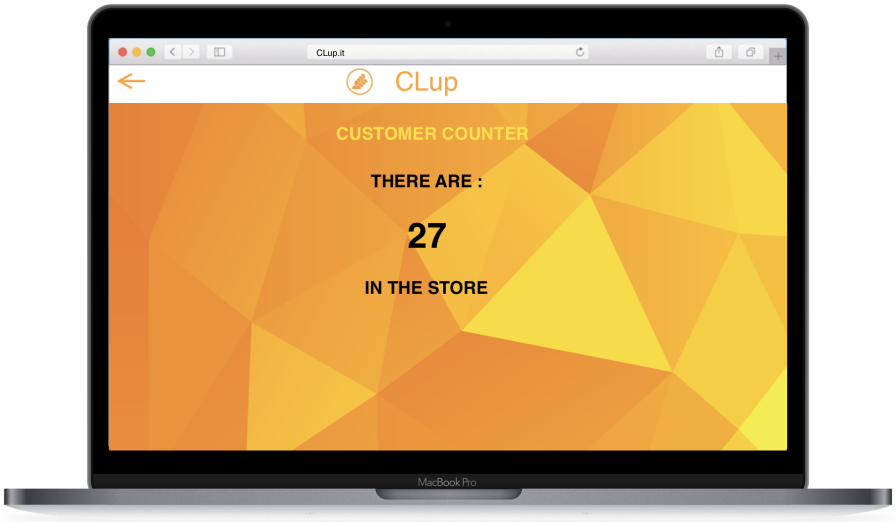



Figure 14: Customer counter

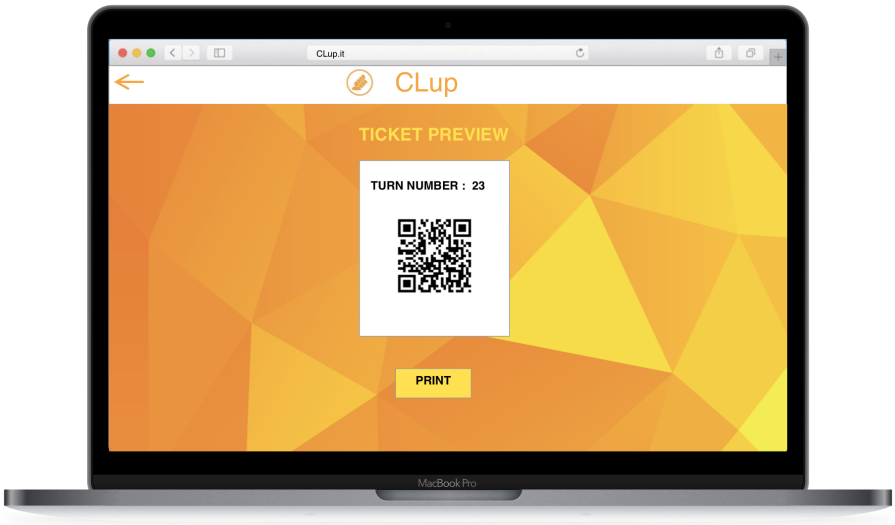



Figure 15: Ticket preview

### 3.1.2 Hardware Interfaces

The required hardware interfaces are listed below:

- Users must own a smartphone in order to download CLup's application and use its services.
- At least one computer and a printer are required in the stores, so that the store manager (or one of his employees) is able to access CLup's WebApp, check the customer counter and print physical tickets.
- Turnstiles able to scan QR codes must be placed at the entrances of stores in order to control the influx of people.
- Stores must be equipped with a display that shows and communicates (by generating sound) the current turn number.

### 3.1.3 Software Interfaces

The external interfaces required by the system are listed below:

- Position: the system uses a Maps Service API to retrieve the position of the user when he/she gets into a line in order to calculate his/her distance from the store and to show to the user the estimated time needed to reach the store.
- Turnstile: the system needs to communicate with the turnstiles' software in order to receive the informations related to the scanned QR codes and to the entrances/exits timestamps. This communication occurs between the store's computer and the turnstiles' integrated controller over LAN.

## 3.2 Functional Requirements

### 3.2.1 List of Requirements

| | |
|---|---|
| **R1** | The system allows users to create an account on the CLup's application by providing username and password |
| **R2** | The system allows users/store managers to log into the application by providing the username and password correctly associated with their accounts |
| **R3** | The system allows users to select a store among the registered ones |
| **R4** | The system is able to generate turn numbers uniquely associated with users |
| **R5** | The system is able to generate QR codes uniquely associated with the relative turn number or reservation |
| **R6** | The system allows users to always check their turn number/QR code |
| **R7** | The system manages lines and reservations by taking into account the maximum stores' capacities |
| **R8** | The system updates in real time the customer counter by keeping track of the scanned QR codes |
| **R9** | The system is able to manage accesses by taking into account the departments' crowdedness |
| **R10** | The system is able to check whether the QR codes displayed by users are valid and let the turnstiles unlock |
| **R11** | The system is able to display turn numbers on a board at the entrance and communicate them by emitting sounds |
| **R12** | The system calculates the shortest path from the location of users to the store they want to line up to |
| **R13** | The system doesn't let users get in more than one line at a time |
| **R14** | The system is able to calculate the durations of users' visits by using time stamps associated to the scanned QR codes |
| **R15** | The system is able to delete users from the queue whenever they do not arrive within a fixed maximum tolerated delay |
| **R16** | The system is able to recalculate waiting times of people in line whenever users don't show up or cancel their reservations |
| **R17** | The system is able to recalculate waiting times of people in line whenever users stay inside the stores for more/less than expected |
| **R18** | The system doesn't let users get in line in case they would not have enough time to do the shopping |
| **R19** | The system is able to store informations about users and their relative visits' durations |
| **R20** | The system allows users to provide the expected amount of time they intend to spend inside the store |
| **R21** | If the system has enough data on user's previous visits, it can allow users not to provide the expected amount of time they intend to spend inside the store by letting the system infer it |
| **R22** | The system allows users to get out of the line they are in |

| | |
|---|---|
| **R23** | The system allows store managers to register their stores by providing all the necessary informations |
| **R24** | The system allows store managers to create an account on CLup's WebApp by providing username, password and a certification document |
| **R25** | The system allows store managers to authenticate |
| **R26** | The system allows store managers to visualize the customer counter on the Web page |
| **R27** | The system allows users to create a list of product types they intend to purchase during their reserved visits |
| **R28** | The system allows users to select the preferred time slot for their reserved visits |
| **R29** | The system allows users to select a new time slot if the one they selected is full |
| **R30** | The system allows users to always check the reservations they made |
| **R31** | The system doesn't let users book visits that overlap each other |
| **R32** | The system allows users to cancel reservations |
| **R33** | The system allows users to check the current state of the line they are in |
| **R34** | The system is able to retrieve the GPS position of the user when he/she gets into a line |
| **R35** | The system uses the Maps Service API to calculate the distance between the user and the store |
| **R36** | The system shows to the user the estimated time needed to reach the store |
| **R37** | The system is able to send a notification to the user when the waiting time is almost equal to the time needed to reach the store |

### 3.2.2 Mapping

In this section, each Goal is associated with the Requirements that ensure the satisfaction of it, in the context of the Domain Assumptions.

| | |
|---|---|
| **G1** | Provide customers a safe environment in the stores |
| **R4** | The system is able to generate turn numbers uniquely associated with users |
| **R5** | The system is able to generate QR codes uniquely associated with the relative turn number or reservation |
| **R7** | The system manages lines and reservations by taking into account the maximum stores' capacities |
| **R8** | The system updates in real time the customer counter by keeping track of the scanned QR codes |
| **R9** | The system is able to manage accesses by taking into account the departments' crowdedness |
| **R10** | The system is able to check whether the QR codes displayed by users are valid and let the turnstiles unlock |
| **R29** | The system allows users to select a new time slot if the one they selected is full |
| **D5** | Each reservation is meant to be for one and only one person |
| **D6** | The turnstiles unlock only if the system allows them |
| **D7** | One person at a time passes through the turnstiles every time a QR code is scanned |
| **D10** | The user respects slots' time boundaries |
| **D11** | The user respects social distancing inside the store |
| **D12** | The user visits only the departments related to the product types he/her specified |

| | |
|---|---|
| **G2** | Allow users to line up at the stores from home |
| **R1** | The system allows users to create an account on the CLup's application by providing username and password |
| **R2** | The system allows users to log into the application by providing the username and password correctly associated with their accounts |
| **R3** | The system allows users to select a store among the registered ones |
| **R4** | The system is able to generate turn numbers uniquely associated with users |
| **R5** | The system is able to generate QR codes uniquely associated with the relative turn number or reservation |
| **R6** | The system allow users to always check their turn number/QR code |
| **R13** | The system doesn't let users get in more than one line at a time |
| **R18** | The system doesn't let users get in line in case they would not have enough time to do the shopping |
| **R19** | The system is able to store informations about users and their relative visits' durations |
| **R20** | The system allows users to provide the expected amount of time they intend to spend inside the store |
| **R21** | If the system has enough data on user's previous visits, it can allow users not to provide the expected amount of time they intend to spend inside the store by letting the system infer it |
| **R22** | The system allows users to get out of the line they are in |
| **R33** | The system allows users to check the current state of the line they are in |
| **D5** | Only one person shows up at the store for each turn number |
| **D14** | The user only gets in the line of stores that are not too far from him/her (the time to arrive should never exceed the turn validity interval) |

| | |
|---|---|
| **G3** | Avoid formation of long lines outside stores |
| **R7** | The system manages lines and reservations by taking into account the maximum stores' capacities |
| **R9** | The system is able to manage accesses by taking into account the departments' crowdedness |
| **R14** | The system is able to calculate the durations of users' visits by using time stamps associated to the scanned QR codes |
| **R15** | The system is able to delete users from the queue whenever they do not arrive within a fixed maximum tolerated delay |
| **R16** | The system is able to recalculate waiting times of people in line whenever users don't show up or cancel their reservations |
| **R17** | The system is able to recalculate waiting times of people in line whenever users stay inside the stores for more/less than expected |
| **R18** | The system doesn't let users get in line in case they would not have enough time to do the shopping |
| **R19** | The system is able to store informations about users and their relative visits' durations |
| **R20** | The system allows users to provide the expected amount of time they intend to spend inside the store |
| **R21** | If the system has enough data on user's previous visits, it can allow users not to provide the expected amount of time they intend to spend inside the store by letting the system infer it |
| **R27** | The system allows users to create a list of product types they intend to purchase during their reserved visits |
| **R28** | The system allows users to select the preferred time slot for their reserved visits |
| **D3** | The user does not to arrive earlier than the indicated time |
| **D5** | Only one person shows up at the store for each turn number |

| | |
|---|---|
| **G4** | Allow store managers to monitor the influx of people in their buildings |
| **R2** | The system allows store managers to log into the application by providing the username and password correctly associated with their accounts |
| **R5** | The system is able to generate QR codes uniquely associated with the relative turn number or reservation |
| **R8** | The system updates in real time the customer counter by keeping track of the scanned QR codes |
| **R10** | The system is able to check whether the QR codes displayed by users are valid and let the turnstiles unlock |
| **R23** | The system allows store managers to register their stores by providing all the necessary informations |
| **R24** | The system allows store managers to create an account on CLup's WebApp by providing username, password and a certification document |
| **R25** | The system allows store managers to authenticate |
| **R26** | The system allows store managers to visualize the customer counter on the Web page |
| **D7** | The turnstiles unlock only if the system allows them |
| **D8** | One person at a time passes through the turnstiles every time a QR code is scanned |

| | |
|---|---|
| **G5** | Allow users to book visits to stores |
| **R1** | The system allows users to create an account on the CLup's application by providing username and password |
| **R2** | The system allows users to log into the application by providing the username and password correctly associated with their accounts |
| **R3** | The system allows users to select a store among the registered ones |
| **R5** | The system is able to generate QR codes uniquely associated with the relative turn number or reservation |
| **R6** | The system allows users to always check their turn number/QR code |
| **R20** | The system allows users to provide the expected amount of time they intend to spend inside the store |
| **R21** | If the system has enough data on user's previous visits, it can allow users not to provide the expected amount of time they intend to spend inside the store by letting the system infer it |
| **R27** | The system allows users to create a list of product types they intend to purchase during their reserved visits |
| **R28** | The system allows users to select the preferred time slot for their reserved visits |
| **R29** | The system allows users to select a new time slot if the one they selected is full |
| **R30** | The system allows users to always check the reservations they made |
| **R31** | The system doesn't let users book visits that overlap each other |
| **R32** | The system allows users to cancel reservations |
| **D6** | Only one person shows up at the store for each booked visit |
| **D11** | The user respects slots' time boundaries |
| **D13** | The user visits only the departments related to the product types he/her specified |

| | |
|---|---|
| **G6** | Allow users to arrive on time for their turns |
| **R12** | The system calculates the shortest path from the location of users to the store they want to line up to |
| **R16** | The system is able to recalculate waiting times of people in line whenever users don't show up or cancel their reservations |
| **R17** | The system is able to recalculate waiting times of people in line whenever users stay inside the stores for more/less than expected |
| **R33** | The system allows users to check the current state of the line they are in |
| **R34** | The system is able to retrieve the GPS position of the user when he/she gets into a line |
| **R35** | The system uses the Maps Service API to calculate the distance between the user and the store |
| **R36** | The system shows to the user the estimated time needed to reach the store |
| **R37** | The system is able to send a notification to the user when the waiting time is almost equal to the time needed to reach the store |
| **D1** | GPS provides the exact location with an error of 5 meters at most |
| **D2** | The time calculated from Google Maps to reach the store is correct |
| **D4** | The user follows the shortest path to the store, without making deviations and stops |
| **D9** | The user does not change location while is waiting for his/her turn |
| **D10** | The user does not change the indicated means of transport |
| **D14** | The user only gets in the line of stores that are not too far from him/her (the time to arrive should never exceed the turn validity interval) |

### 3.2.3 Scenarios

**Scenario 1: Line up**

Mario is a 40 years old teacher living in Pisa. During the weekend, he wants to take advantage of some free time to go grocery shopping for the coming week. Since the covid-19 pandemic is getting worse, he does not want to risk ending up in crowded lines or stores. Therefore, he decides to exploit the virtual lining service, offered by CLup application on his smartphone. He chooses the nearest Eurospin on the app and finds out that the waiting time is 40m 32s. He decides to get in line and selects "on foot" as means of transport, to get some fresh air. He wants to do a lot of shopping, so he indicates an hour as estimated time to spend inside the store. After approximately 15 min, Mario receives a notification from CLup saying that it's time to leave and therefore, he prepares for his 20 min walk to reach the store. Once arrived, he sees his number on the board, he selects the QR code on his smartphone to have it scanned by the turnstile at the entrance and, finally, he starts shopping.

**Scenario 2: Get out of line**

Elisa is a young mum of two kids who decides to go shopping for Christmas' presents on a Wednesday morning, while her children are at school. Considering that Christmas is coming, she expects to find many people shopping and so she decides to use the CLup application to get in the line of her favorite gift shop. The current waiting time is such that she will be perfectly in time to pick her kids up from school. Unfortunately, it seems like some store's departments are particularly crowded, or maybe some customers are staying inside more than they should. As a result, Elisa's waiting time has increased a little and she realizes she would be late for the end of school. In the end, she gets out of the line and decides to try again the following day.

**Scenario 3: Book a visit**

Anna is a single 36 years old business woman currently living in Milan. She has a very busy schedule and therefore she cannot risk wasting too much time buying groceries. For this reason, she often uses CLup application on her smartphone to book visits to her usual supermarket. On Tuesday, she will be getting off work at 4 pm and she has exactly two hours, before a late afternoon video call. Among the available time slots, she chooses the one from 5:30 pm till 6:30 pm, as she is sure she will not need to stay inside for more than half an hour. Last time she went to the supermarket, it took her 20 min to do the shopping, so she selects that same duration, shown by the system, for her Tuesday's visit as well. Lastly, she select the categories of products that she intends to purchase and confirms the whole reservation.

**Scenario 4: Cancel a reservation**

Giorgio is a young man working as a consultant in Turin. While he's in the office, he decides to make a reservation to go to the supermarket later in the afternoon, because he wants to surprise his wife with the grocery. He uses CLup application to book the visit for the preferred time slot. At lunch time, Giorgio's wife, who does not trust her husband's sense of initiative, calls Giorgio to inform him that she is going grocery shopping. At this point, the poor Giorgio is forced to cancel his booked visit. He opens CLup app, goes in the "My Reservations" section and cancels his last reservation.

**Scenario 5: Customer without a smartphone gets a ticket**

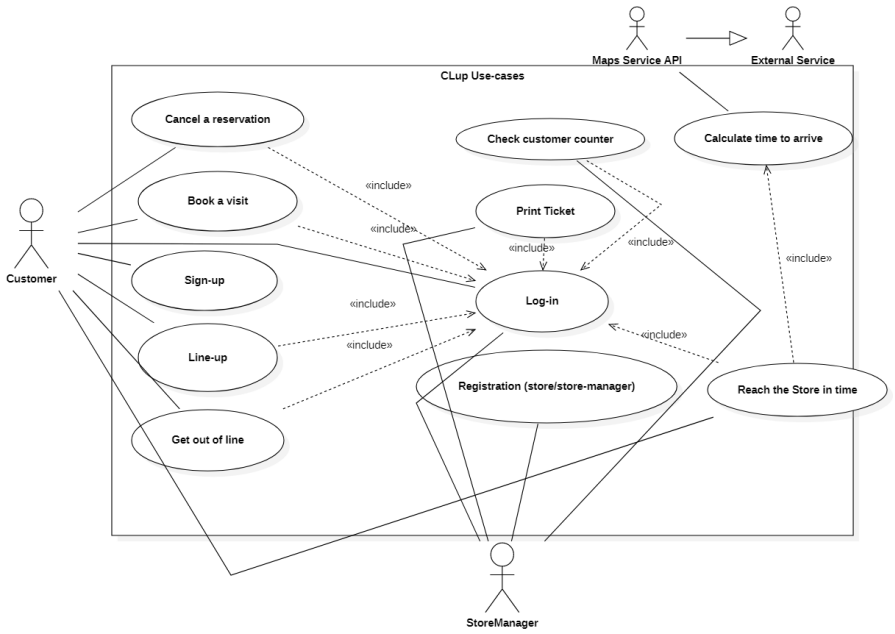
Antonio is a 71 years old retired man who lives in Monza during the period of lockdown, due to the COVID-19 pandemic. Antonio is reluctant to technology and in fact, the cellphone he owns is suitable only for basic functionalities. For this reason, he is not able to download CLup's application and use its services. He needs to go grocery shopping, so he walks up to the nearest supermarket. He reaches the info desk and finds out that there is a virtual lining mechanism going on and that he will have to wait for his turn at the entrance. Angela, a store's employee who is responsible for taking care of customers who do not have access to the required technology, accesses to CLup's WebApp from her computer and prints for Antonio a ticket, with a turn number and a QR code.

She explains to him how and where to scan the QR code as soon as his turn number appears on the led board at the entrance.

**Scenario 6: Check customer counter**

Paola is the manager of a big Esselunga store in Milan. During the day, she periodically checks that everything is proceeding as planned. In particular, she takes a look at each store department to make sure that there are no crowds and she also checks the number of people currently in the store. To do so, she accesses CLup's WebApp from her office's computer and clicks on "Check customer counter" button. In this way, she is always able to verify whether the CLup system is efficiently working, without violating the maximum capability of her store.

### 3.2.4 Use Cases Diagram



### 3.2.5 Use Cases

| | |
|---|---|
| **Name** | Login |
| **Actors** | User/Store manager |
| **Entry condition** | User/Store manager has successfully signed up in/registered to the application |
| **Event flow** | 1) User/Store manager opens the application/accesses the WebApp<br>2) User/Store manager enters his/her credentials in the “Username” and “Password” fields<br>3) User/Store manager clicks on the “Log in” button |
| **Exit conditions** | User/Store manager is successfully redirected to the home page and he/she can start using CLup’s services |
| **Exceptions** | 1) The User/Store manager enters invalid Username<br>2) The User/Store enters invalid Password<br>If one of the above exceptions is raised, an error message will be displayed and the system will take the user back to the login page |

| | |
|---|---|
| **Name** | Sign up |
| **Actors** | User |
| **Entry condition** | The user has installed the application on his/her device and has the internet connection available |
| **Event flow** | 1) User opens the application<br>2) User clicks on “Sign up” button<br>3) User fills all the mandatory fields and provides the necessary information<br>4) User accepts the privacy terms and conditions<br>5) User clicks on “Allow CLup to access GPS position while using the application”<br>6) User clicks on “Confirm” button<br>7) The system saves and stores the data |
| **Exit conditions** | The user has successfully registered and now he’s able to use the application’s services. |
| **Exceptions** | 1) The user is already signed up<br>2) The user didn’t fill all of the mandatory fields with valid data<br>3) The username is already taken<br>If one of the above exceptions is raised, an error message will be displayed and the system will take the user back to the registration form page |

| | |
|---|---|
| **Name** | Registration of store and store manager |
| **Actors** | Store manager |
| **Entry condition** | Store manager has the internet connection available and has accessed the Web Browser on his/her device |
| **Event flow** | 1) Store manager accesses CLup's WebApp<br>2) Store manager clicks on "Register store" button<br>3) Store manager fills all the mandatory fields<br>4) Store manager uploads a certification document that proves that he/her is the manager of the specified store<br>5) Store manager accepts the privacy terms and conditions<br>6) Store manager clicks on "Confirm" button<br>7) The system validates the document<br>8) The system confirms the registration of the store manager<br>9) The system saves and store the data |
| **Exit conditions** | The registered store appears on CLup's stores list, the store manager has successfully registered and now he's able to use CLup's services |
| **Exceptions** | 1) The store manager is already registered<br>2) The user didn't fill all of the mandatory fields with valid data<br>3) The certification document is not valid<br>If one of the above exceptions is raised, an error message will be displayed and the system will take the user back to the registration form page |

| | |
|---|---|
| **Name** | Line up |
| **Actors** | User (provided with the app) |
| **Entry condition** | User has already logged in the application |
| **Event flow** | 1) User clicks on “Choose store” button in the home page<br><br>2) User inserts city and name of store and selects the preferred store<br><br>3) User clicks on “Line up” button<br><br>4) The system shows expected waiting time and only if the user decides not to line up:<br>3.1) The user clicks on “Cancel” button<br>3.2) The system takes the user back to the home screen<br><br>5) Customer chooses means of transport (car/bicycle/on foot)<br><br>6) User indicates expected duration of stay inside the store, selecting the one of his/her previous visit (if system has enough data to display it) or explicitly specifying it<br><br>7) The system stores the data and calculates user’s distance from store and time required to reach it<br><br>8) The system shows the estimated waiting time and the calculated time needed to reach the store<br><br>9) The system generates QR code and turn number |
| **Exit conditions** | The user can see: waiting time, the time needed to reach the store, turn number and QR code |
| **Exceptions** | 1) User is already lined up for another store<br><br>2) User selects a store that is closed or is about to close<br><br>If one of the above exceptions is raised, an error message will be displayed and the system will take the user back to the home page |

| | |
|---|---|
| **Name** | Book a visit |
| **Actors** | User (provided with the app) |
| **Entry condition** | User has successfully logged in the application |
| **Event flow** | 1) User clicks on “Choose store” button in the home page<br>2) User inserts city and name of store and selects the preferred store<br>3) User clicks on “Book visit” button<br>4) User selects the preferred date and time slot<br>5) User indicates expected duration of stay inside the store, selecting the one of his/her previous visit (if system has enough data to display it) or explicitly specifying it<br>6) User selects the item categories he/she intends to purchase form a list<br>7) User checks and confirm the reservation<br>8) The system stores the reservation into the database<br>9) The system generates QR code and turn number |
| **Exit conditions** | The reservation is valid and the user is able to display the QR code associated to it |
| **Exceptions** | 1) The chosen time slot is full<br>2) The chosen time slot overlaps with other user’s reservations’ time slots<br>If the above exception is detected, an error message will be displayed and the system will take the user back to the time slot choice |

| | |
|---|---|
| **Name** | Cancel a reservation |
| **Actors** | User (provided with the app) |
| **Entry condition** | User has already logged in and made a reservation |
| **Event flow** | 1) User clicks on “My reservations” button<br>2) User selects the reservation he/she wants to delete<br>3) User click on “Cancel” button<br>4) The system deletes the reservation from the database |
| **Exit conditions** | The reservation is canceled and the QR code is not valid anymore |
| **Exceptions** | None |

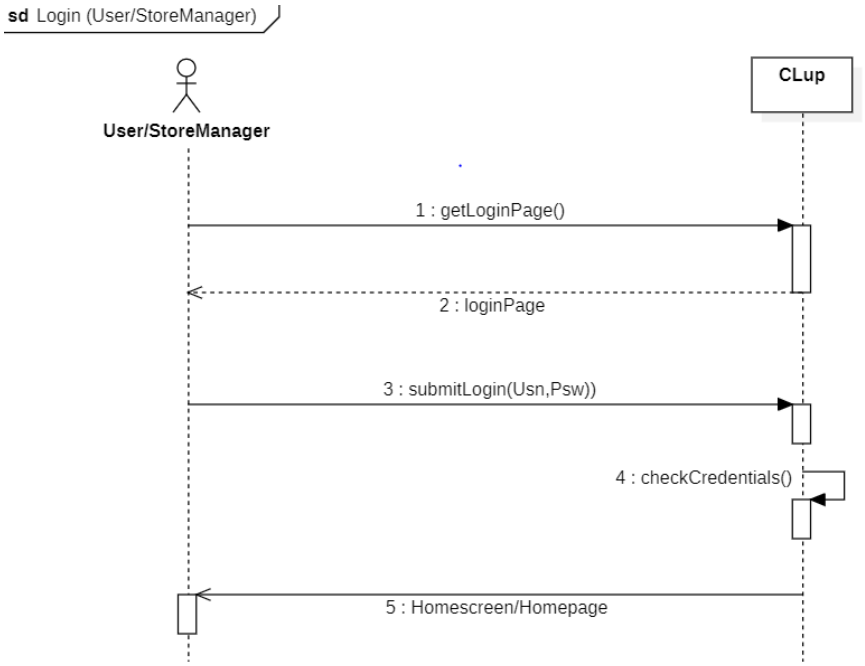
| | |
|---|---|
| **Name** | Get out of line |
| **Actors** | User (provided with the app) |
| **Entry condition** | User has successfully logged in and got in a line |
| **Event flow** | 1) User clicks on “My Line” button<br>2) User click on “Get out of line button” |
| **Exit conditions** | The system removes user from the line and takes him back to the home screen |
| **Exceptions** | None |

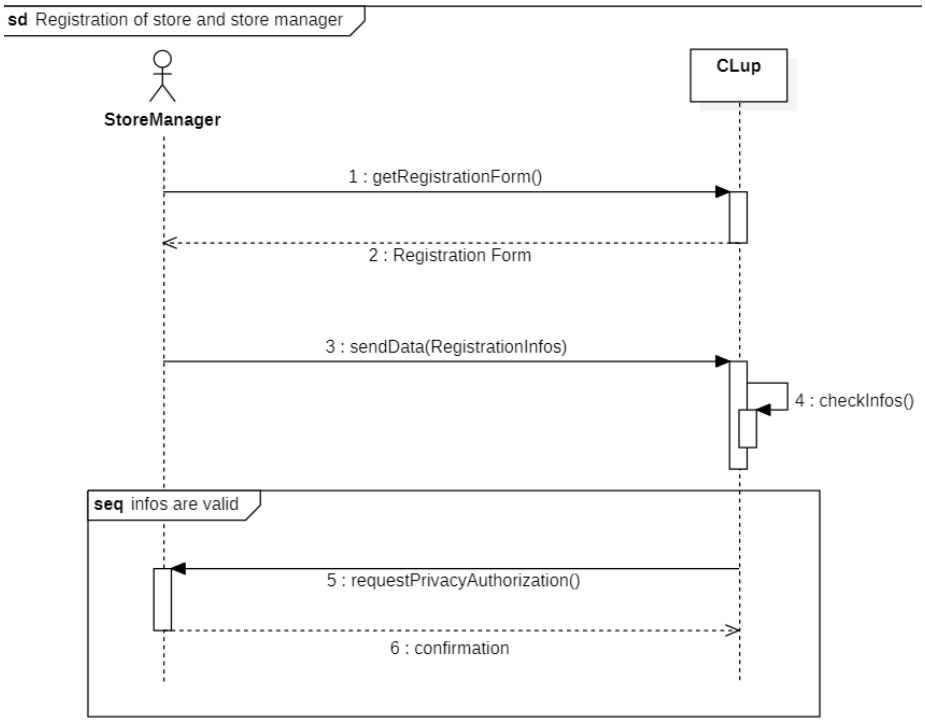
| | |
|---|---|
| **Name** | Print ticket |
| **Actors** | Store manager |
| **Entry condition** | Store manager has already logged in the application |
| **Event flow** | 1) Store manager clicks on “Print ticket” button in the web app home page<br>2) The system generates a QR code<br>3) Store manager checks the preview of the ticket, containing turn number and QR code<br>4) Store manager clicks on “Print” button<br>5) The ticket is printed |
| **Exit conditions** | The store manager has a physical ticket |
| **Exceptions** | 1) The store is closed<br>If the above exception is detected, an error message will be displayed and the system will take the store manager back to the home page<br>2) The printer is not available<br>If the above exception is detected, an error message will be displayed and the system will take the store manager back to the previous page |

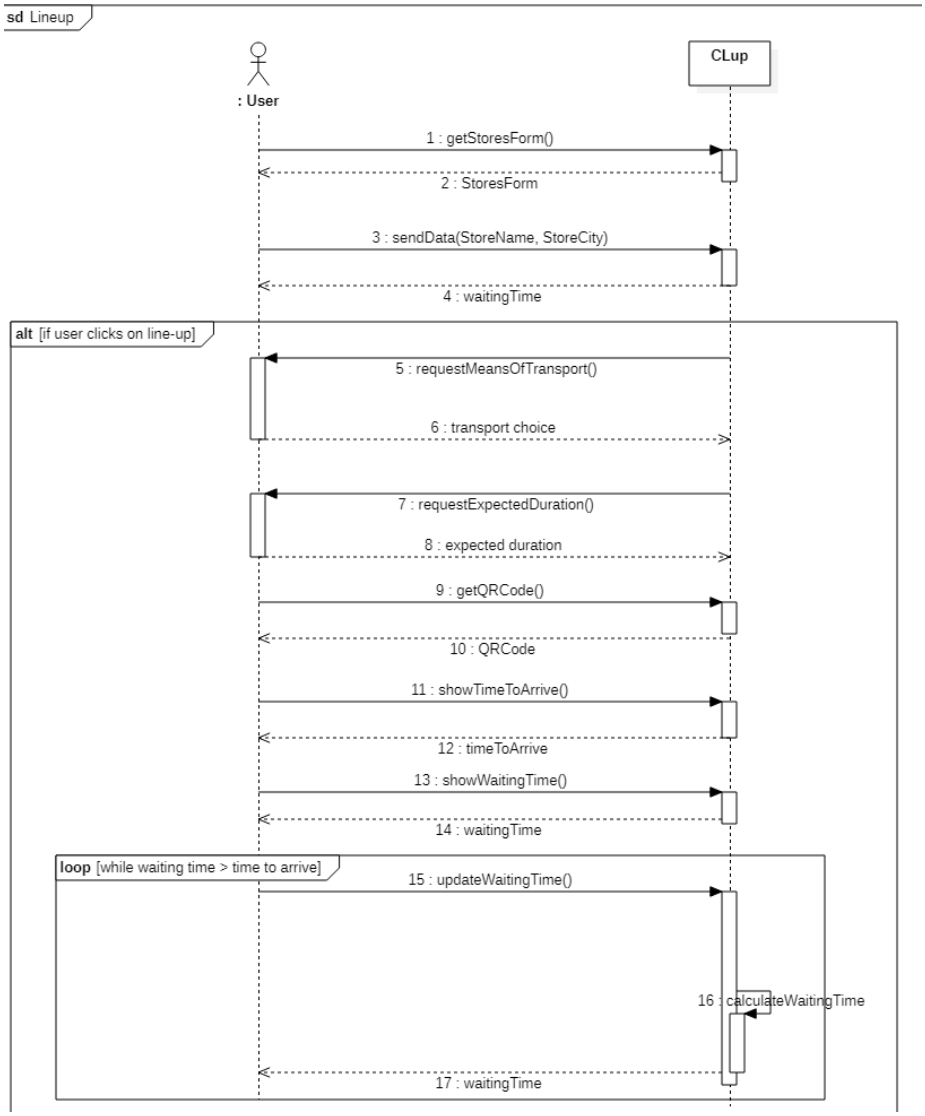
| | |
|---|---|
| **Name** | Reach the store on time |
| **Actors** | User (provided with the app) |
| **Entry condition** | User has successfully logged in and got in a line |
| **Event flow** | 1) The system notifies the user when waiting time is almost equal to time the user needs to reach the store<br>2) User reaches the store and sees his turn number on the queue board<br>3) User clicks on “My QR code” button<br>4) User gets the QR code scanned by the turnstile<br>5) The system checks the validity of the QR code |
| **Exit conditions** | The QR code is successfully scanned and checked and the system allows the turnstile to unlock. The system updates the customer counter. |
| **Exceptions** | If the QR code is not valid (not expired or false), the smart turnstile will not let him/her in |

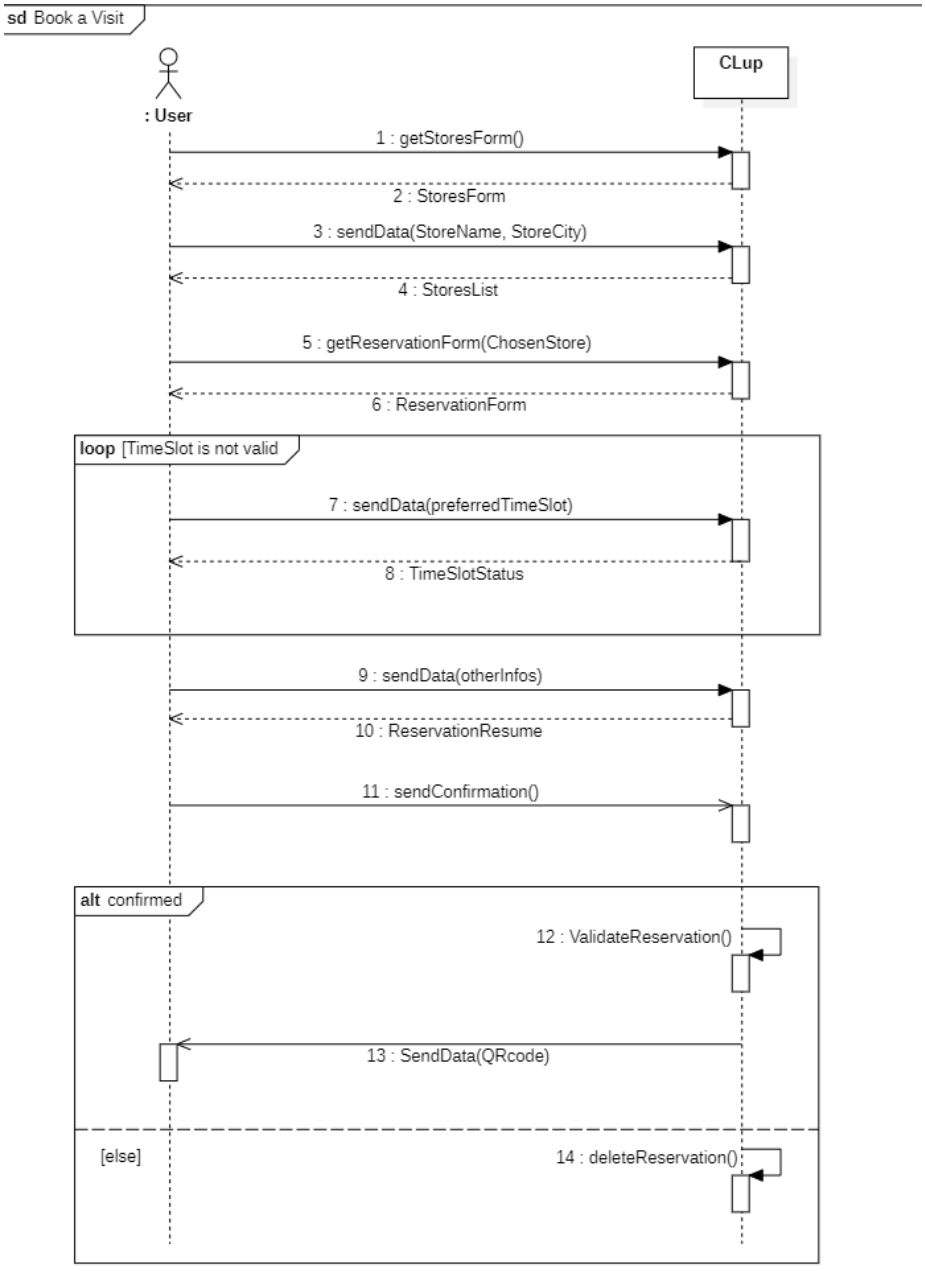
| | |
|---|---|
| **Name** | Check customer counter |
| **Actors** | Store manager |
| **Entry condition** | Store manager has successfully logged in the application |
| **Event flow** | 1) Store manager clicks on “Check customer counter” button in the web app home page |
| **Exit conditions** | The counter is displayed |
| **Exceptions** | 1) The store is closed<br>If the above exception is detected, an error message will be displayed and the system will take the store manager back to the home page |

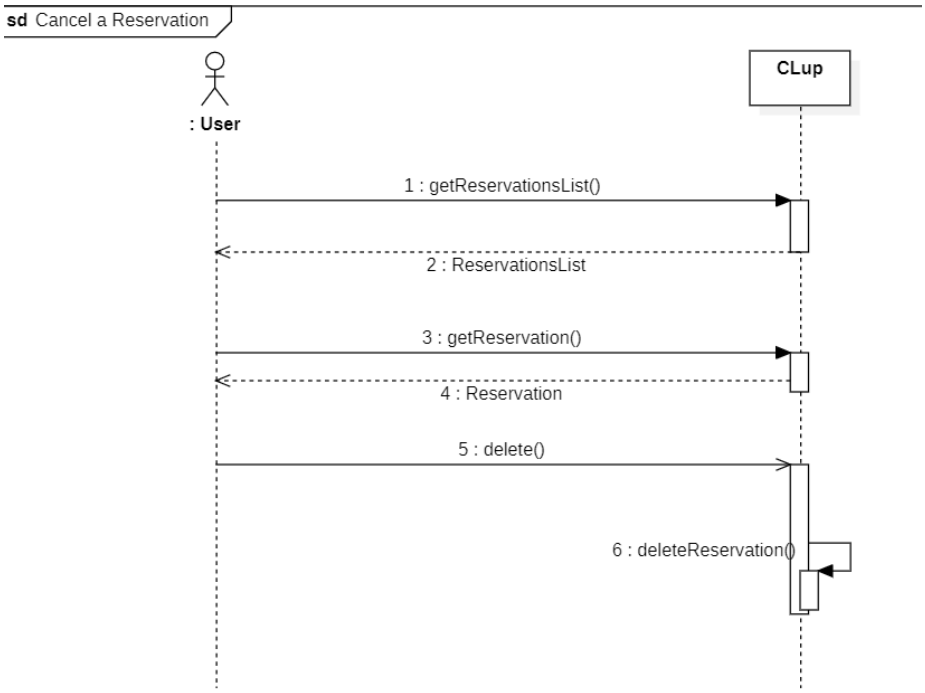
### 3.2.6 Sequence Diagrams

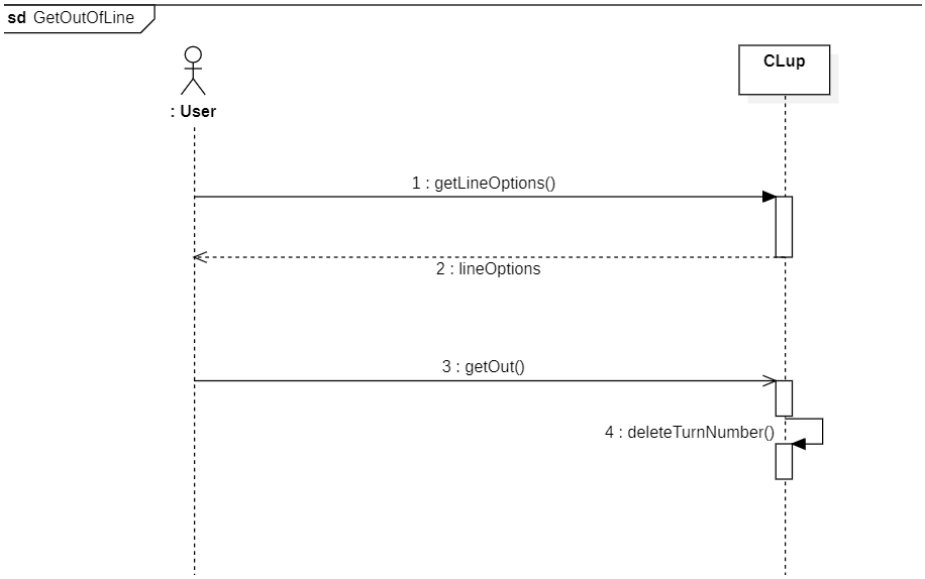


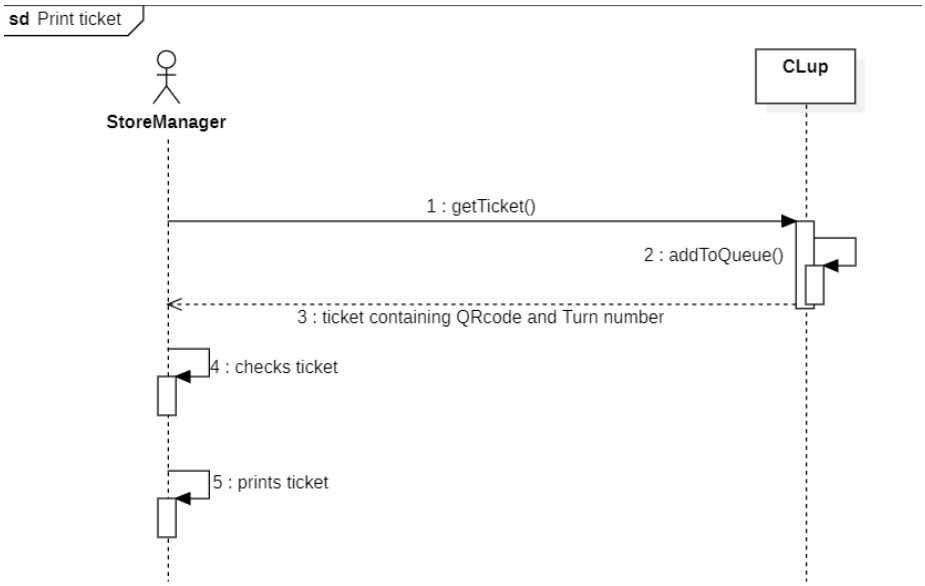


sd Registration of store and store manager
StoreManager
CLup
1 : getRegistrationForm()
2 : Registration Form
3 : sendData(RegistrationInfos)
4 : checkInfos()
seq infos are valid
5 : requestPrivacyAuthorization()
6 : confirmation

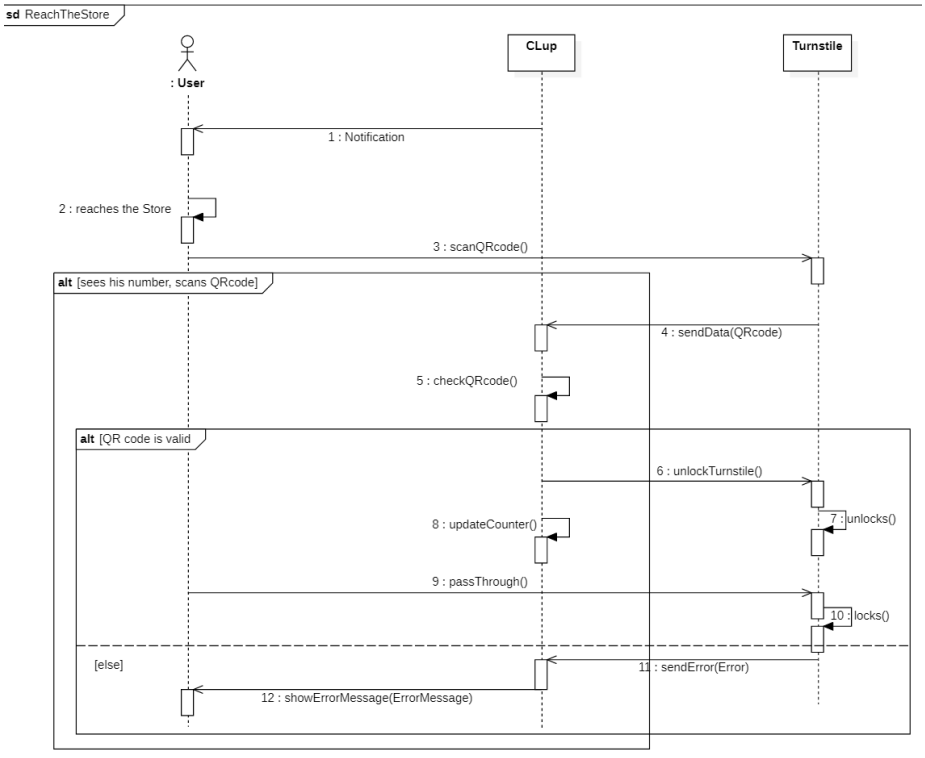

sd Lineup
: User
CLup
1 : getStoresForm()
2 : StoresForm
3 : sendData(StoreName, StoreCity)
4 : waitingTime
alt [if user clicks on line-up]
5 : requestMeansOfTransport()
6 : transport choice
7 : requestExpectedDuration()
8 : expected duration
9 : getQRCode()
10 : QRCode
11 : showTimeToArrive()
12 : timeToArrive
13 : showWaitingTime()
14 : waitingTime
loop [while waiting time > time to arrive]
15 : updateWaitingTime()
16 : calculateWaitingTime
17 : waitingTime


sd Book a Visit
: User
CLup
1 : getStoresForm()
2 : StoresForm
3 : sendData(StoreName, StoreCity)
4 : StoresList
5 : getReservationForm(ChosenStore)
6 : ReservationForm
loop [TimeSlot is not valid
7 : sendData(preferredTimeSlot)
8 : TimeSlotStatus
9 : sendData(otherInfos)
10 : ReservationResume
11 : sendConfirmation()
alt confirmed
12 : ValidateReservation()
13 : SendData(QRcode)
[else]
14 : deleteReservation()


sd Cancel a Reservation
: User
CLup
1 : getReservationsList()
2 : ReservationsList
3 : getReservation()
4 : Reservation
5 : delete()
6 : deleteReservation()


sd GetOutOfLine
: User
CLup
1 : getLineOptions()
2 : lineOptions
3 : getOut()
4 : deleteTurnNumber()


sd Print ticket
StoreManager
CLup
1 : getTicket()
2 : addToQueue()
3 : ticket containing QRcode and Turn number
4 : checks ticket
5 : prints ticket


sd ReachTheStore
: User
CLup
Turnstile
1 : Notification
2 : reaches the Store
3 : scanQRcode()
alt [sees his number, scans QRcode]
4 : sendData(QRcode)
5 : checkQRcode()
alt [QR code is valid
6 : unlockTurnstile()
7 : unlocks()
8 : updateCounter()
9 : passThrough()
10 : locks()
[else]
11 : sendError(Error)
12 : showErrorMessage(ErrorMessage)

## 3.3 Performance Requirements - Non-functional Requirements

It is important to remark that the CLup system provides services that help people to avoid risks and to respect the limitations imposed by the coronavirus emergency. For this reason, the system has to be able to serve a great number of users and store managers simultaneously. The system also has to be able to update the waiting times seen by the users in a quick, reactive and correct way, as soon as they change in the system's database.

## 3.4 Design Constraints - Non-functional Requirements

### 3.4.1 Hardware limitations

The hardware requirements needed in order to use the system's functionalities on the mobile app are the following:

- iOS or Android smartphone
- GPS
- Connection to internet (Wi-Fi/4G/3G/2G)

### 3.4.2 Any other constraints

The system will have to ask users' permission to retrieve and use their positions without storing them. Moreover, the collection and treatment of personal data must comply with the principles and rules of the General Data Protection Regulation (GDPR).

## 3.5 Software System Attributes - Non-functional Requirements

### 3.5.1 Reliability and Availability

The system must guarantee a 24/7 service. It is expected to be available 99.99% of the time, considering the critical situation due to the coronavirus pandemic.

### 3.5.2 Security

Passwords and sensitive informations like the recordings of users' durations of visits must be protected using encryption.

### 3.5.3 Maintainability

The development of the application must guarantee a high level of maintainability and flexibility, in order to make the addiction of new features and options easy and cheap. For this reason, appropriate design patterns will be used, which will be detailed in an adequate document.

### 3.5.4 Scalability

The system is expected to be built considering a high level of scalability on the number of stores (and therefore the number of users).

### 3.5.5 Compatibility

The WebApp must be compatible with the most popular browsers, while the smartphone application must be compatible with Android Oreo or later versions and on iOs 11 or later versions in order to be available on a wide range of devices.

# 4 Formal Analysis Using Alloy

In this section we present our alloy code and results. We decided to focus on some relevant system's aspects with the aim to model these constraints:

- A notification is shown to a lined-up user if waiting time approaches time to arrive
- No overlaps between same user's reservations
- Every QRcode is associated to only one reservation or turn number
- In a queue, waiting time related to a turn number is <= waiting time related to the next turn numbers
- A store's counter can't exceed store's capacity

To simplify alloy modelling we decided to show the notification if "waiting time" is equal to the "time to arrive", even if in the real application the notification is shown when one is approaching the other (with a delta of few minutes).

```
open util/integer


sig Time {
    minutes: one Int
}

{ minutes ≥ 0 }

sig Date { }

sig Location {
        timeToArrive: one Time
}
{
        timeToArrive.minutes>0
}

sig QRcode { }

sig User {
    reservations: set Reservation,
    turnNumber: lone TurnNumber,
    notification: lone Notification,
```

```
    queue: lone Queue
}

sig Notification {
        user: one User
}

sig TurnNumber {

    user: one User,
    qrCode: one QRcode,
    number: one Int,
    location: one Location,
    queue: one Queue,
    waitingTime: one Time,


 }{

     number>0 and
     (waitingTime.minutes ≥ 0) and
     (location.timeToArrive.minutes ≥ 0) and
     (waitingTime.minutes ≥ location.timeToArrive.minutes)

 }

sig Reservation {

    startTime: one Time,
    endTime: one Time,
    date: one Date,
    user: one User,
        duration: one Int,
    qrCode: one QRcode
}{

(sub[endTime.minutes, startTime.minutes] ≥ duration) and
(startTime.minutes < endTime.minutes) and
(duration > 0)

}

sig Queue {
    turnNumbers: set TurnNumber,
    store: one Store
}{

    all disj t,t': TurnNumber | t.number ≠ t'.number
    all disj t,t',t'': TurnNumber | all x: Int | all y: Int | x>0 and y < x
        ↪ and (t''.number = t.number+x implies t'.number = t.number+y )


}


sig Store {
    queue: one Queue,
    capacity: one Int,
    counter: lone Counter
}{
        capacity>0
}

sig Counter{
        count: one Int,
    store: one Store
}{
        count ≥ 0
```

```
}

--All Dates must be associated to a Reservation
fact DatesToReservation{

        all d: Date | some r: Reservation | d = r.date
}

--All locations must be associated to a TurnNumber
fact LocationsToTurnNumber{

        all l: Location | some t: TurnNumber | l = t.location
}

--Various relations between signatures


fact reservationsUsersRelation{

    all u: User | all r: Reservation | (r in u.reservations implies r.user =
        ↪ u) and (r.user = u implies r in u.reservations)
}

fact queuesStoresRelation{

    all q: Queue | all s: Store | (q = s.queue implies q.store = s) and (s =
        ↪ q.store implies s.queue = q)
}

fact counterStoreRelation{
        all c: Counter | all s: Store | (c = s.counter implies c.store = s)
            ↪ and (c.store = s implies c = s.counter )
}

fact uniqueTimes{
        no disj t1,t2: Time | t1.minutes = t2.minutes
}

--A turnNumber can't be associated to more than one QRCode  or to more than
    ↪ one user

fact turnNumbersUsersUniqueness{

    all u: User | all t: TurnNumber | (t = u.turnNumber implies t.user = u)
        ↪ and (u = t.user implies u.turnNumber = t)
}

fact turnNumbersQueuesUniqueness{

    all q: Queue | all t: TurnNumber | (t in q.turnNumbers implies t.queue =
        ↪ q) and (q = t.queue implies t in q.turnNumbers)
}

--Users can't be in more than one queue

fact usersQueuesUniqueness{
        all u: User | all disj q1,q2: Queue | (u in q1.turnNumbers.user)
            ↪ implies (u not in q2.turnNumbers.user)
}

--notifications can't exist on their own
fact notificationsUsersRelation{

        all n: Notification | all u: User | (n = u.notification implies n.
            ↪ user = u) and (u = n.user implies u.notification = n)

}
```

```
--QRcodes are associated to only one turnNumber/Reservation
fact uniqueQRcodes{

    all disj r,r': Reservation | r.qrCode ≠ r'.qrCode
    all disj t,t': TurnNumber | t.qrCode ≠ t'.qrCode
    all r: Reservation | all t: TurnNumber | r.qrCode ≠ t.qrCode
}

--There are no overlaps between same user's reservations on the same date
fact noReservationsOverlaps{

        no disj r',r'': Reservation | r'.date = r''.date and r'.user = r''.
            ↪ user and r'.startTime.minutes ≤ r''.startTime.minutes and
    r'.endTime.minutes ≥ r''.startTime.minutes
}

--Waiting time associated to a TurnNumber is <= waiting time associated to
    ↪ the next turnNumbers

fact turnNumbersOrder{

        all disj t1,t2: TurnNumber | (t1.number > t2.number and t1.queue = t2
            ↪ .queue) implies (t1.waitingTime.minutes > t2.waitingTime.
            ↪ minutes)

}

--Different locations have different timeToArrive

fact locationsTimes{
        no disj l1,l2: Location | l1.timeToArrive = l2.timeToArrive
}

--if waitingTime = timeToArrive, user receives the notification

fact notificationArrived{

   all u: User | (u.turnNumber.waitingTime.minutes ≠ u.turnNumber.location.
        ↪ timeToArrive.minutes ) implies (no u.notification) and (no u.
        ↪ notification) implies (u.turnNumber.waitingTime.minutes ≠ u.
        ↪ turnNumber.location.timeToArrive.minutes )
   all u: User | (u.turnNumber.waitingTime.minutes = u.turnNumber.location.
        ↪ timeToArrive.minutes ) implies (u.notification ≠ none) and   (u.
        ↪ notification ≠ none) implies (u.turnNumber.waitingTime.minutes = u
        ↪ .turnNumber.location.timeToArrive.minutes )

}

--A store's counter can't be higher than his capacity
fact CounterNotHigherThanCapacity{
        all s: Store | s.counter.count ≤ s.capacity

}

pred show{}
pred noOverlapReservations{}
pred yesNotification{

        some u:User | #u.turnNumber.waitingTime.minutes = #u.turnNumber.
            ↪ location.timeToArrive.minutes

}
pred noNotification{}

--QrCode is associated to only one thing (Reservations or TurnNumbers)
assert QrCodeDifferentForDifferentThings{
        all t: TurnNumber | all r: Reservation | t.qrCode ≠ r.qrCode
    all disj t1,t2: TurnNumber | t1.qrCode ≠ t2.qrCode
```

```
    all disj r1,r2: Reservation | r1.qrCode ≠ r2.qrCode
}


run storeCapacity for 4 but exactly 3 Store, 0 User, 0 QRcode, 0 Time
run noOverlapReservations for 6 but 1 User, exactly 3 Reservation, 0
    ↪ Notification
run yesNotification for 3 but 1 User, 1 Store
run noNotification for 3 but exactly 0 Notification, 1 User, 1 Store, 0
    ↪ Reservation

check QrCodeDifferentForDifferentThings
```

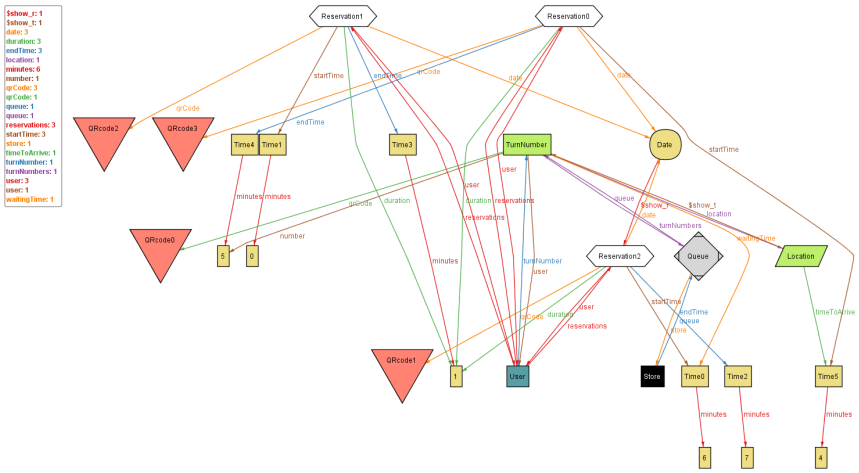



Figure 16: No overlaps in reservations' timeslots

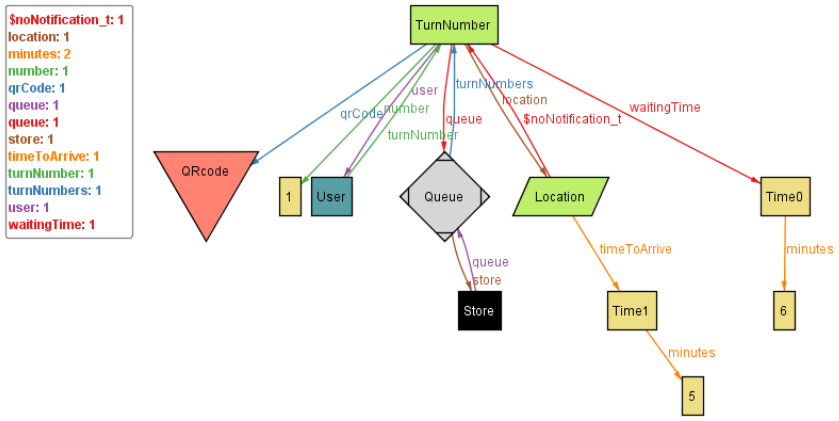



Figure 17: No notifications when waiting time is not equal to time to arrive

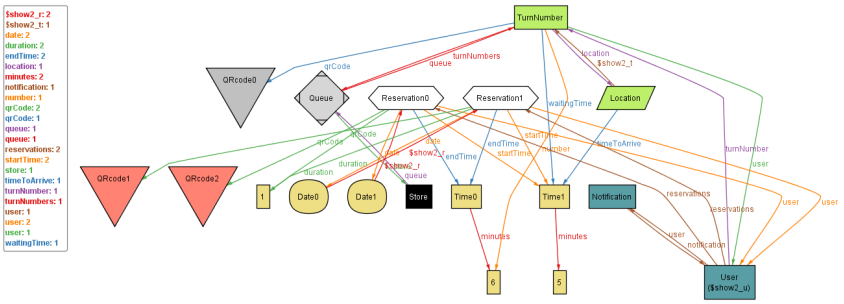



Figure 18: User gets a notification when waiting time is equal to time to arrive

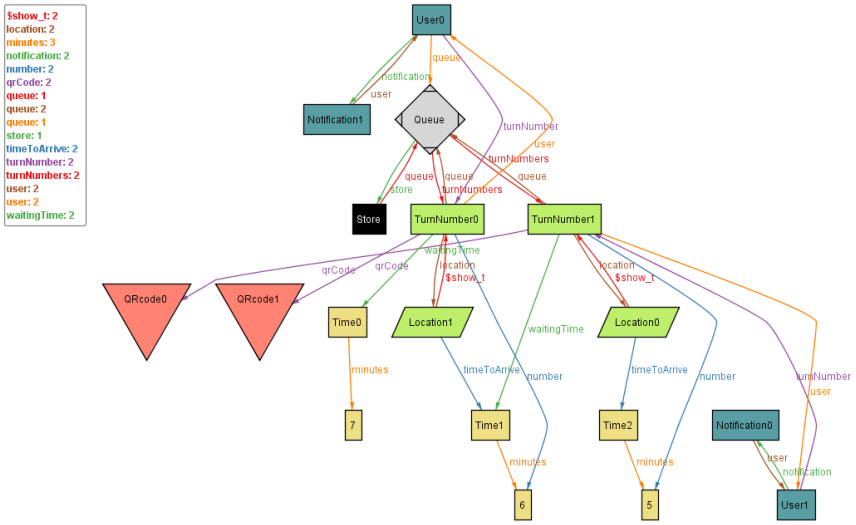



Figure 19: Greater TurnNumbers have greater waiting times

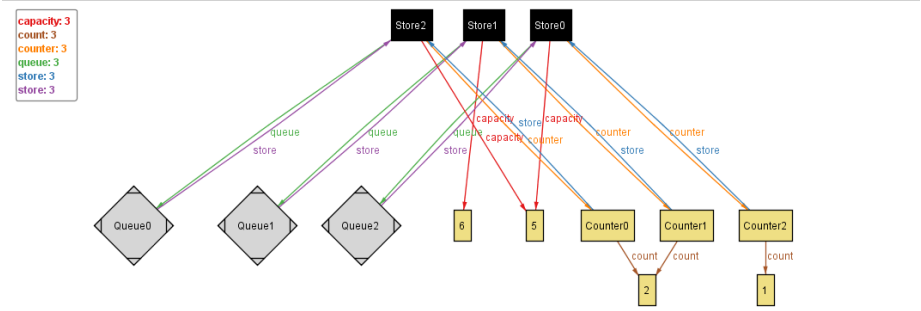



Figure 20: Store's counter is less or equal store's capacity

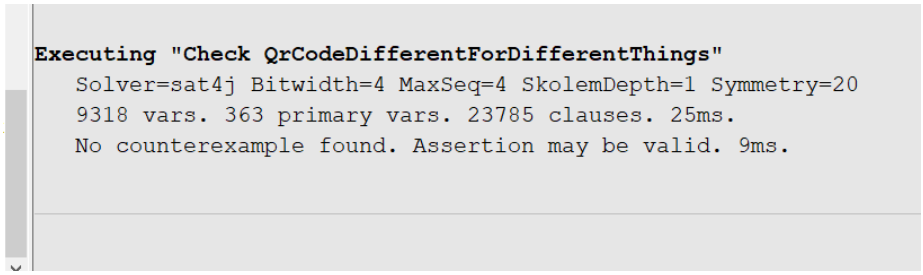



Figure 21: QrCode uniqueness assertion's result

# 5 Effort Spent

### Student 1

| Task | Hours |
|---|---|
| Purpose, Scope, Definitions | 2 |
| World & Shared Phenomena, Goals | 3 |
| Product perspective | 6 |
| Product functions | 1 |
| Domain assumptions | 3 |
| External Interface Requirements | 2 |
| List of Functional Requirements | 5 |
| Mapping | 5 |
| Scenarios and Use Cases | 3 |
| Sequence Diagrams | 5 |
| Non-functional Requirements | 2 |
| Formal analysis using Alloy | 8 |

### Student 2

| Task | Hours |
|---|---|
| Purpose, Scope, Definitions | 4 |
| World & Shared Phenomena, Goals | 3 |
| Product perspective | 2 |
| Product functions | 3 |
| Domain assumptions | 3 |
| External Interface Requirements | 7 |
| List of Functional Requirements | 5 |
| Mapping | 5 |
| Scenarios and Use Cases | 5 |
| Sequence Diagrams | 2 |
| Non-functional Requirements | 4 |
| Formal analysis using Alloy | 3 |

# 6 Reference Documents

- The diagrams in the document have been made with `https://staruml.io` and `https://www.draw.io`
- Alloy code documentation supported the code deployment: `http://alloy.mit.edu/`