**1 – Introduction**

*1.1 – Purpose*

This document represents the Requirements Analysis and Specification Document (RASD).

The document goals are to analyse the customer real needs and to show the solution the software offers, its goals and the relationship between these and the functional and non-functional requirements, attributes and constraints. In addition to that, it will provide possible use cases and system behaviour after the release.

This document is addressed to: procurement that can use it as a contractual base, project manager to control the developing and to have a base for change (or risk) management, to management that can estimate a first baseline (of time and cost) and to developer

*1.2 – Scope*

1.2.1 – Brief problem description

The coronavirus emergency has put a strain on society on many levels. Many countries imposed the lockdowns, which allow people to go out from their home only for essential needs as for working or for grocery shopping. They usually enforced strict rules to the stores and to the people that can go out in specific situation, such as limiting the number of accesses to a building or keeping the distance of at least one meter between people. Often these rules guarantee a store with few people inside, but a long-crowded queue outside.

Grocery store is the main challenge. Everyone need to do grocery shopping, food is an important part of everyone life, so it’s important to minimize the possible risks that a queue outside could take and, also, it’s needed an easy way to manage it, avoiding useless long waiting time, and a simple way to manage the influx of people. In addition to that, because of situation uncertainty, stores don’t want to hire more people to manage this problem.

1.2.2 – Brief system description

“CLup – Customer Line Up” is a system that manages the queue automatically without the people in a physical line. It should, obviously, cover all the functionalities that a good physical queue has, such as not losing the position and knowing when it’s customer’s turn to enter the store. In addition to that, the system will provide the estimated waiting time, a software part will be dedicated to the notifies, so when the customer must go out to reach the store in time, depending on his/her position. Moreover, the customer could decide to book for a specific time or putting in a normal queue.

By a manager view, this app could permit an automated management of the queue, regulating the flux in the building by setting the maximum number of people that could be inside the store at the same time. It will also manage the delays without any human intervention. It could also be integrated with proxies made by the store.

Moreover, in case of booking or line-up impossibility, the system will suggest other stores of the same chain/congregate near the user (if they exist) or other booking time (if they exist and only in case of booking attempt)

1.2.3 – Current systems

The main app in the market is UFirst that guarantees only the possibility to retrieve a number and to know how many people precede the customer. The time in which the customer can “line-up” is statically imposed, so, usually, it’s an hour before the closure. In addition to that there’s not any type of suggestion in case of impossibility to be inserted in the queue. Fortunately, the CLup has more features that can satisfy most of customer and store manager need, as described before.

*1.3 – Goal*

The goals written below are the mandatory parts that this application must have to achieve the purpose it has.

* [G1]: The system must allow customers to line up (on and offline)
* [G2]: The system must allow store managers to regulate the influx of people in the building
* [G3]: The system must provide customers with an estimate of the waiting time
* [G4]: The system must advice the customers that’s theirs turn to enter
* [G5]: The system must allow customer to book a visit in the store
* [G6]: The system must communicate the store manager the possibility to let in the store more people than the limit
* [G7]: The system must alert the customers when they have to leave to arrive in time at the store

[G2] and [G6] are about the management of the flux by the store manager: so how many people could be inside at the same time, or even what is the maximum “size” of every department. In particular [G6] allows the store manager to temporarily increase the maximum number, due to special condition, such as people specifies that they will buy items from different departments, so the maximum number in every department will never be reached and, even with more people, the health measures will be respected. Obviously, the system will communicate and not decide by itself because it isn’t aware about DPCM rules or whatever, so the final decision will be owned by store manager.

[G3] and [G7] are useful to let the customer organize and allow him/her to not lose the turn.  
[G4] could substitute a person outside the building that coordinate who must enter, so every management part of the queue is provided by the system,

*1.4 – Definition, acronyms and abbreviations*

1.4.1 – Definition

* Active customers: customers inside the building at the same time

1.4.2 – Acronyms

* RASD: Requirements Analysis and Specification Document

1.4.3 – Abbreviations

* CLup; Clup – Customer Line up
* [Gn]: n-goal.
* [Dn]: n-domain assumption.
* [Rn]: n-functional requirement.

*1.5 – References*

* IEEE Recommended Practice For Software Requirements Specifications - IEEE Std 830-1998

*1.6 – Overview*

The RASD is composed by five parts:

1. The first part focus on a briefly introduction to the current situation, to the solutions offered by the product, emphasizing its goals. This section analyses the current system available in the market, highlighting the main differences between them. Moreover, it specifies which conventions the document will adopt, to understand better the following part.
2. The second part offers a perspective of the product, underlining the main functions, attributes, constraints and user characteristic. It will also provide an initial explanation of requirements.
3. The third part focuses on all requirements and on use cases. It will also provide scenarios to justify all the functions.
4. The fourth part provides an explanation with Alloy, reporting also the results.
5. The fifth part is about the effort spent by the team to build this document.

**2 – Overall description**

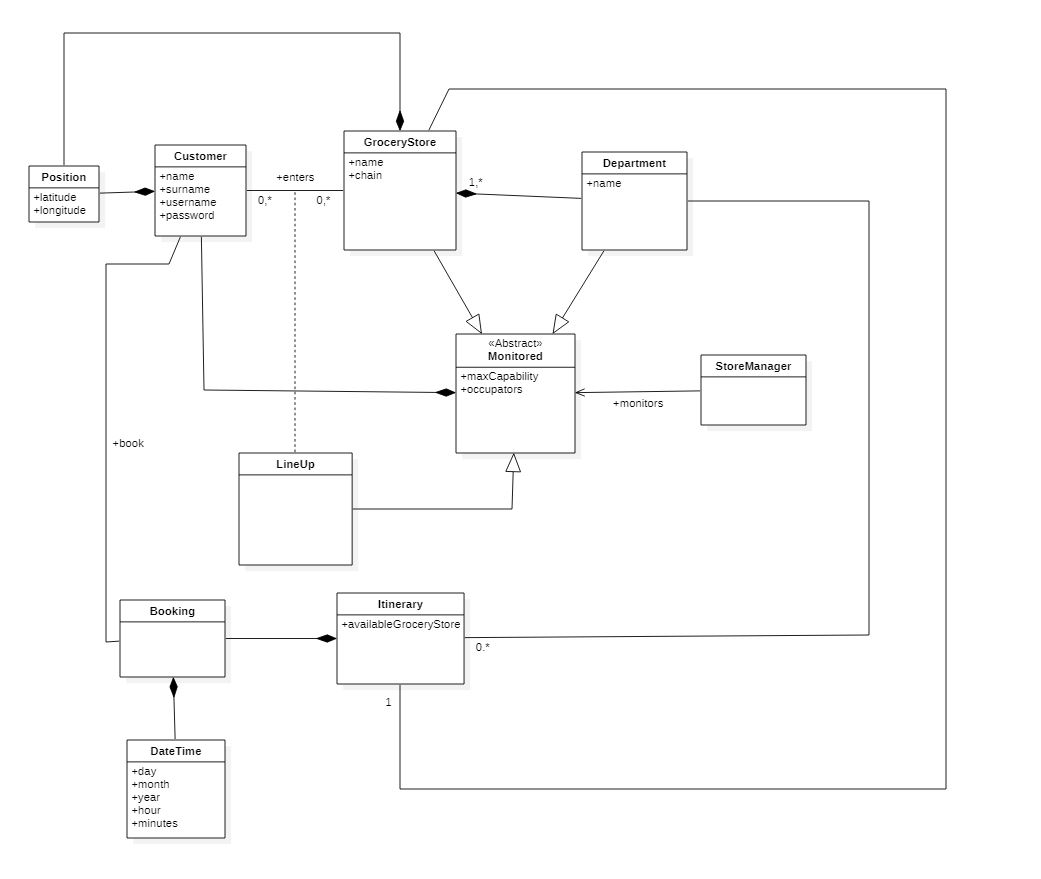
This section of the SRS should describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3 of the SRS, and makes them easier to understand.

*2.1 – Product perspective*

This product will be developed from scratch and, as said in the next section, it will use an external tool (Google Maps) to collect user position on which the alert and calculation system is based. It will ensure precision to our calculation and a better user experience because, usually, the user is accustomed to this tool. \*Moreover, to guarantee data security, an external third part will develop the authentication system. \*

The product, except for those, is almost totally self-contained.

To analyse better the structure, a UML class diagram is presented below, which is a high level description of the component of the system.

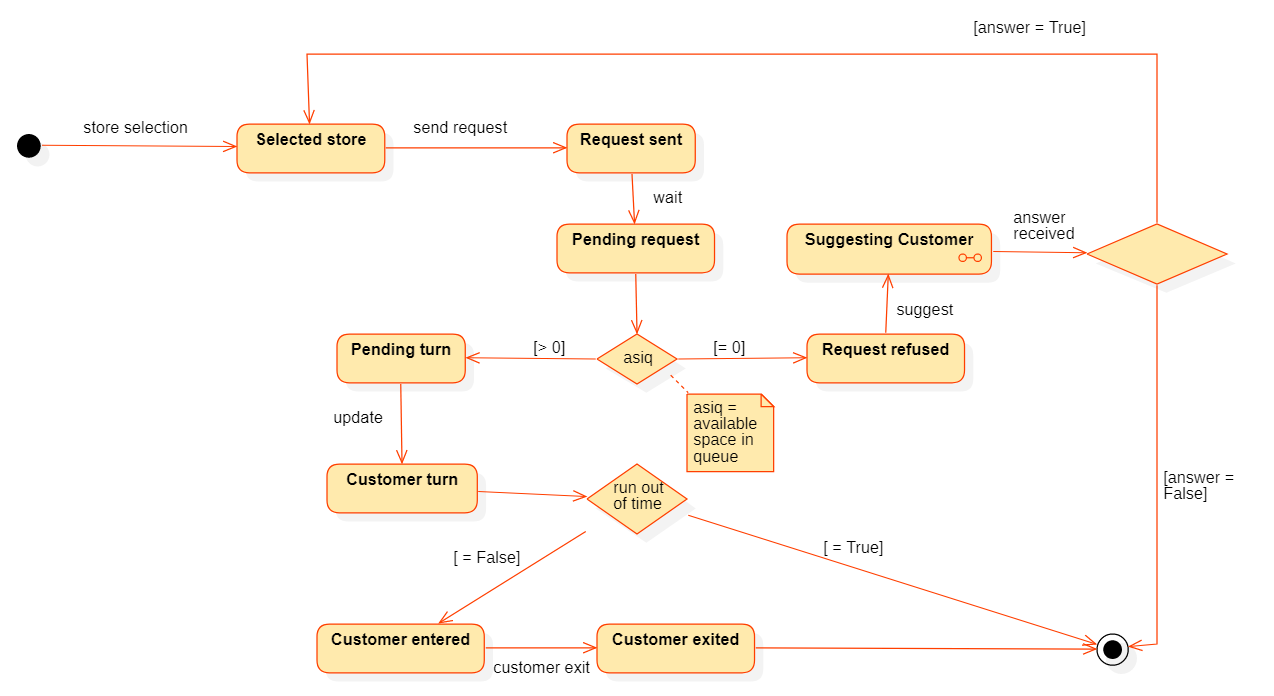


The classes *Grocery Store*, *Department* and *LineUp* are a generalization of the abstract class *Monitored* because they all have the attributes: “maxCapability” and “occupators” (this last one is actually a list of objects of class *User*) and because they logically are all monitored by a *StoreManager* which can change and monitor them, so the *StoreManager* can be seen as an observer for the *Monitored* which is the observed.

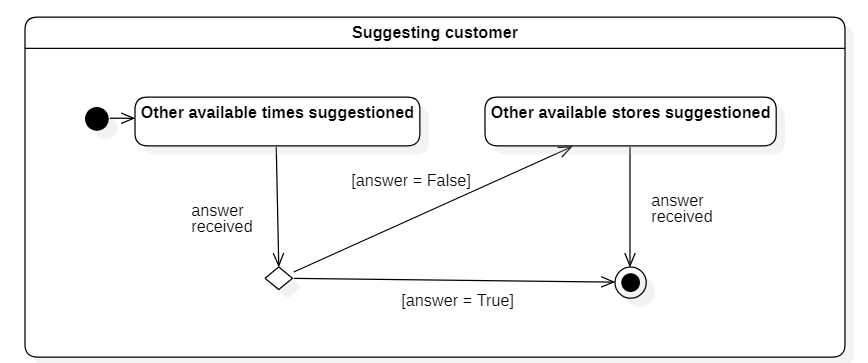
There is a composition between *Department* and *GroceryStore* because there is no sense for a *Department* to exist if it is not referenced by a *GroceryStore* object.

The *LineUp* class is the one that logically represents the queue of users waiting to enter the store.

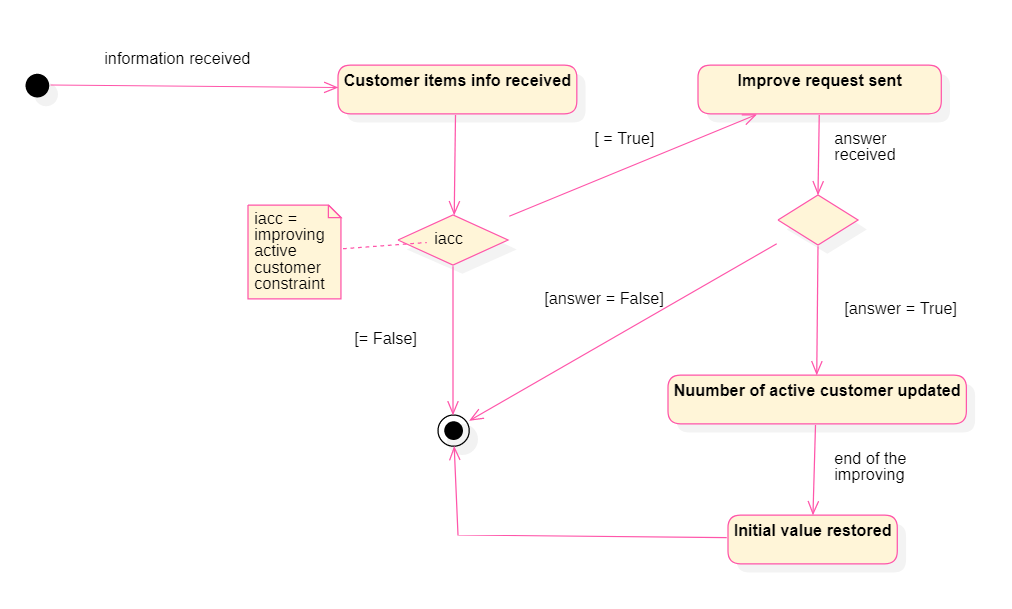
According to the possible situations and to the different point of view (e.g. for the store manager or the customer), the system will change its state many times and to explain the development of the changing the state-charts will be introduced below.



In this first diagram a customer view is offered, it covers the both cases (“retrieve number online” and “book a visit). Initially a store is selected, a request of line-up formulated and sent. If it is confirmed, the customer is officially in waiting for his/her turn until it’s time to enter. If he/she entered before the running out of time, the application will remain in this state until he/she will have finished the visit, otherwise it will directly reach the final state.   
If the request is rejected, the system will suggest the customer with other possibilities that are analysed below the next diagram. If the customer will accept the suggestion, the process will restart, otherwise it will finish.



The “suggesting customer” state varies depending what case is. In the “retrieve a number online”, this part contains only the “other available stores suggested”. In the “book a visit”, this state contains what the diagram above shows, so, in the beginning, it recommends other available times and, if the user rejects it, then it behaves like the first case.



The last critical aspect is the case of improving the customer inside the building due to the specified items they will buy. The system will send a request to the store manager, who can accept it or reject it. In case of positive answer, the number is improved, then restored after the specific situation.

In addition, it will be presented a brief description of the main characteristics of the interfaces hereunder.

*2.1.1 – User interfaces*

The user interfaces must contain short message with a neutral font. It must be intuitive due to the vast audience nature. The notifications must be simple and, also, with short message, such as “It’s time to go” for the incoming time of departure, or “It’s your turn”. In addition to that, the used colours can’t be green and red to explain different concepts, such as “red” for “not your turn” and “green” for “it’s your turn” because of the possible presence of daltonism. Hereunder, some examples are showed.

Mockup 1 – Icon

Mockup 2 – Login / Sign Up

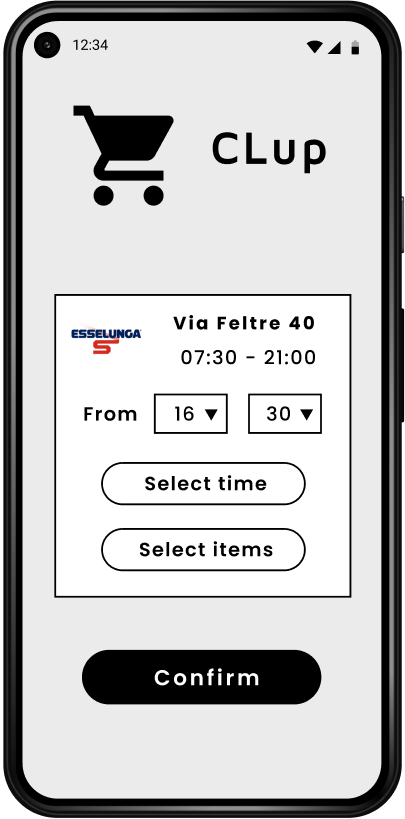
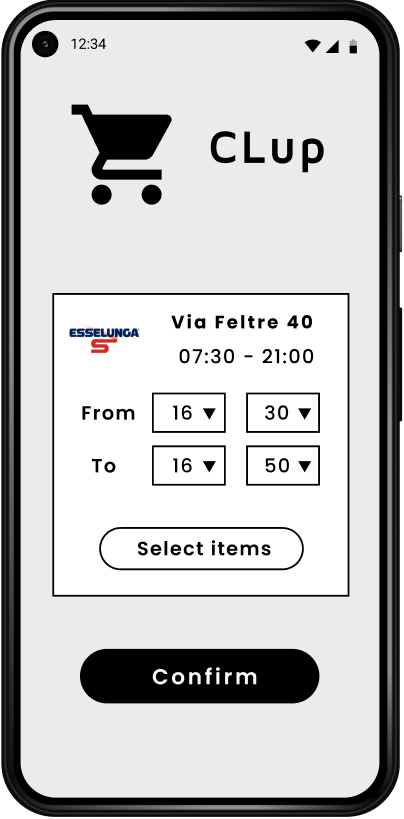
Mockup 3 – Sign Up Form

Mockup 4 – Login Form

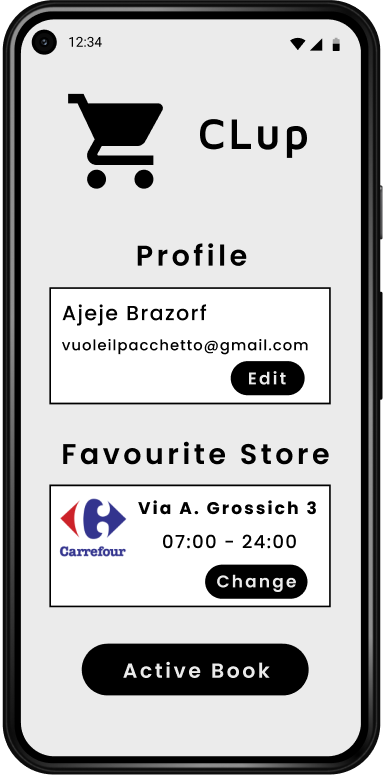
Mockup 6 – List of Store

Mockup 5 – Customer Home

Mockup 7 – Reservation after selecting store

Mockup 8 – Reservation after selecting time of visit

Mockup 10 – Customer Profile

Mockup 9 – List of Departments

Immagine che contiene testo

Descrizione generata automaticamente Immagine che contiene testo

Descrizione generata automaticamente

Mockup 11 – Active Booking Info

Mockup 12 – Store Manager Home ile



Mockup 13 – Departments Live Info

*2.1.2 – Hardware interfaces*

The system has no hardware interfaces.

*2.1.3 – Software interfaces*

The system doesn’t provide any API to external applications.

*2.2 – Product functions*

In the direction of defining unambiguous boundaries for the system in development, a specific list of the functions that the product will/won’t offer is provided and precisely explained. The first four is related to the user need, the second part regards the store manager uses.

*2.2.1 – User Registration/Authentication*

It guarantees that a user can register himself/herself to the application to gain all the service the system provides.

*2.2.2 – Retrieve a number / Book a visit*

The two functions are quite similar: “retrieve a number online” could be seen as a “book a visit” with the first time available. The system permits the possibility to put the customer in the online queue. A unique number will be assigned to every customer and their waiting time will be continuously updated. A notification will be also sent in two main cases:

1. The waiting time is updated to 0 and it is customer turn
2. The waiting time is equal (or greater with a range of tolerance) to the estimating spent time in reaching the store.

*2.2.3 – Cancel the reservation*

In case of possible mistakes, the users are able to cancel the reservations done and they can make another one. The possibility to delete that is forbidden in case the waiting time is lower than a certain tolerance value.

*2.2.4 – Queue management*

The system manages all aspects around the queue: it maintains the customers position in the “line”, it supports an integration “by hand” from the manager, it keeps tracking of the user inside the building to avoid the overgoing of the constraint. In addition to that, it controls and manages the user delays (with a tolerance parameter) or reservations cancels and organizes the queue in a way that can support them, avoiding, for instance, the possibility to give 5 minutes to a user that needs 10 minutes to reach the store.

The end of the queue is managed dynamically, indeed from the closure store time the system calculates when to stop accepting reservations.

*2.2.5 – Registration of the store*

It’s possible to register every store by own manager through authentication, also in case of store chain. The app supports the insertion of the timetable, the type of items, the mean average of the customers visit time, the maximum number of people inside the store, the maximum number of people per department.

*2.2.6 – Values management*

The application dynamically supports the changing of the main value previously described by the store manager

*2.2.7 – Collecting user data*

The system collects for every user the time of his/her visit to calculate a better approximation of the same, in a way that the queue management is more effective.

*2.2.8 – Store Manager Authentication*

Every store manager is able to login in the application and accesses to the other management functions.

*2.2.9 – Creation of special accounts*

Not all people could access the required technology, therefore the system permits every store to create a special account that can make how many reservations they want. These are seen as the same made by online users, so they don’t have privilege to skip the line or to access immediately, except for an absent customer and the next one could reach the store in the necessary time.

*2.2.10 – Display store’s info*

The store manager is able to see dynamically the collected data about the store, so what is the average of the visit, how many customers are inside the building, how long the queue is.

**3 – Specific Requirements**

*3.1 – Functional Requirements*

*3.1.1 – User*

*Scenario 1*

Lilli is a desperate housewife that wants to cook a lasagna for her husband’s birthday. Opening the fridge, she realizes there is no bechamel. Outside it’s very cold and Lilli don’t want to wait in the queue and to risk a flu. She has also to clean the kitchen before cooking and she can’t be that if she must wait outside the grocery store.  
Fortunately, she downloaded CLup: so she opens the app and lines up for the nearest grocery store. The app estimates the arrival time from her position to the store is about 7 minutes and it shows her the total waiting time (that’s about 20 minutes), so she can start cleaning the kitchen. She hasn’t always to control the time because the app notifies that her turn is coming in 10 minutes, so Lilli successes in cleaning the kitchen and goes to the grocery to buy bechamel.

*Scenario 2*

Sandrino is a retired veteran, he doesn’t like technology, so he doesn’t have a smartphone. Therefore, he must go to the store and retrieve hand out ticket at the totem on the outside that shows his position in the queue. He’s still worried about his waiting time because a lot of people with the app could line up faster.

Fortunately to him, the guy before Sandrino is Giuseppino, a moody guy, that suddenly decides to not go to the store. Thanks to the queue management system of CLup, after 5 minutes it recognizes that Giuseppino won’t arrive, it controls the hand-out ticket queue (noticing that’s not empty) and, as a result, it allows Sandrino to enter the store and to buy the Breasola he loves

*Scenario 3*

Franco is an off-campus student who is really busy due to lessons, laboratories and sports. His timetable is a very strict schedule, he has a determined free interval time for every day, in which he must doing shopping. He can use CLup to solve this problem: instead of wasting time in a queue and risking to not doing the shopping because of that, he can book a visit to his favourite store. Moreover, he can specify the expected time of a visit and the items he’s going to buy. By booking Franco is ensured he will go grocery shopping at the time he wants and be safe at home as soon as possible where he can eat his favourite dish: “pasta con tonno”.

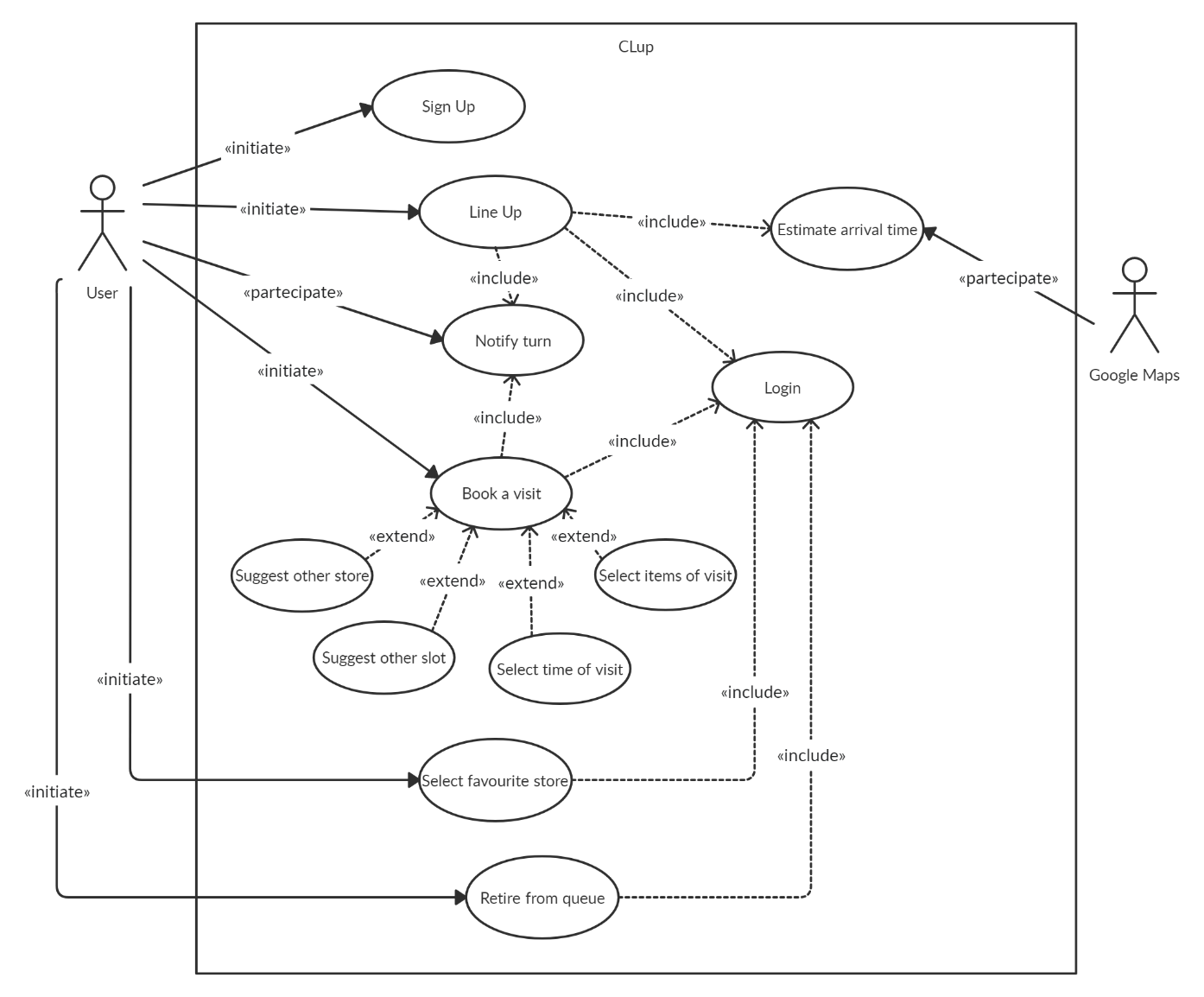
*Scenario 4*

Alberto is a delivery man who works all day, he prefers to do grocery shopping very late, but not always his favourite grocery store is open. When he books his visit and his favourite grocery shop is closed or full, the system suggests him other stores open at that time so he can keep doing grocery shopping whenever he wants.

*Scenario 5*

Michele is a scatterbrain guy. He forgot that this evening he will have to prepare a dinner for his childhood friends. Using CLup, he’s trying to book a visit in the only grocery store that has sushi packs. Unfortunately, the interval time he planned to go is already full because he chose it late. In this case, CLup suggests to Michele alternative slots in the same store, so he can fix to his forgetfulness and pick the sushi packs for the dinner.

*Use Case Diagram*



*Use Cases*

|  |  |
| --- | --- |
| Name | Sign Up |
| Actor | User |
| Entry Conditions | The user has opened the app on his/her device |
| Events Flow | 1. The user chooses the “Sign up” option. 2. The user fills the mandatory fields. 3. The user fills the optional fields with not mandatory data. 4. The user chooses the confirmation option. 5. The system saves the data. |
| Exit Conditions | The user is registered and the system has his data stored. |
| Exceptions | 1. The user is already signed up. In this case the system warns the user and suggests him/her to login. 2. The user didn’t fill all of the mandatory fields with valid data. 3. The username is already taken. 4. The e-mail is already registered.   All the exceptions are handled by notifying the user and taking him/her back to the sign up activity. |

|  |  |
| --- | --- |
| Name | Login |
| Actor | User |
| Entry Conditions | The user is previously successfully signed up and has the application installed on his/her device. |
| Events Flow | 1. The user clicks on the “Login” button. 2. The user enters username and password in the respective fields. 3. The user clicks on the “Confirm” button. |
| Exit Conditions | The user is logged in and the system take him to the home page |
| Exceptions | 1. The user enters invalid Username. 2. The user enters invalid Password.   All the exceptions are handled by notifying the user and taking him/her back to the login activity. |

|  |  |
| --- | --- |
| Name | Select favourite store |
| Actor | User |
| Entry Conditions | The user has already logged in. |
| Events Flow | 1. The user clicks on the “Profile” button. 2. The user clicks on the “Change favourite store” button. 3. The user chooses between the possible stores. |
| Exit Conditions | The user chooses his/her favourite store and the preference is saved to his/her profile |
| Exceptions | - |

|  |  |
| --- | --- |
| Name | Line Up |
| Actor | User |
| Entry Conditions | The user has already logged in. |
| Events Flow | 1. The user clicks on the “Line Up” button. 2. The user select the store he/she wants to line up. 3. The user clicks on “Confirm” button. |
| Exit Conditions | The user is regularly lined up to the selected store |
| Exceptions | 1. The store he/she has selected is closed. 2. The queue ends 15 minutes before the store closes. |

|  |  |
| --- | --- |
| Name | Book a visit |
| Actor | User |
| Entry Conditions | The user has already logged in. |
| Events Flow | 1. The user clicks on the “Book Visit” button. 2. The user select the store he/she wants to book for. 3. The user select the time of his/her visit 4. The user clicks on “Confirm” button. |
| Exit Conditions | The user has booked a visit to the selected store starting from the selected time |
| Exceptions | - |

|  |  |
| --- | --- |
| Name | Suggest other store |
| Actor | User |
| Entry Conditions | 1. The user has booked a visit to a store. 2. At the time of the visit the store is either closed or full. |
| Events Flow | 1. *The user select “Change store”* 2. The user select the store he/she wants to book for, different from the previous. 3. The user clicks on “Confirm” button. |
| Exit Conditions | The user has booked a visit to the selected store starting from the selected time |
| Exceptions | - |

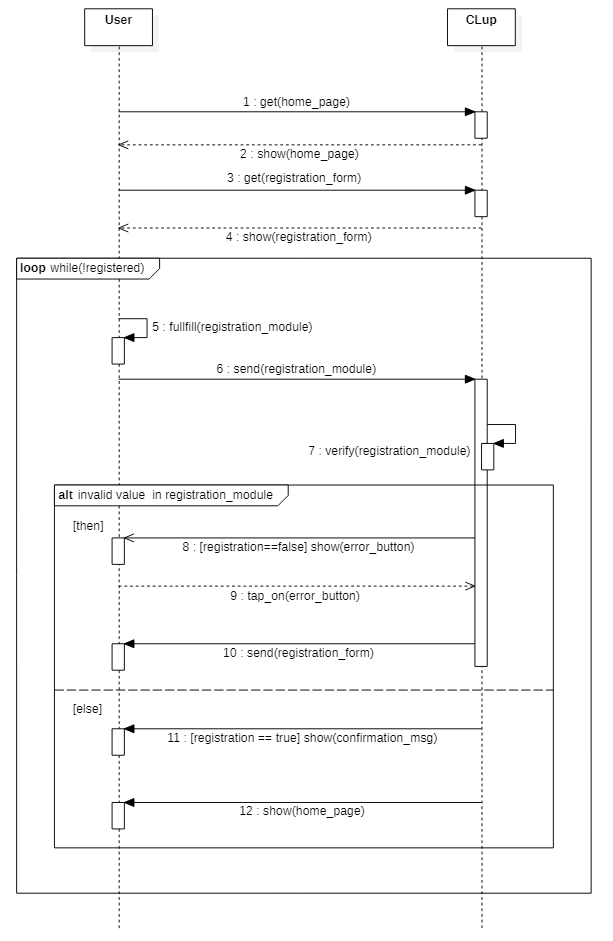
|  |  |
| --- | --- |
| Name | Suggest other slot |
| Actor | User |
| Entry Conditions | 1. The user has booked a visit to a store. 2. At the time of the visit the store is either closed or full. |
| Events Flow | 1. *The user select “Change slot”* 2. The user select the time he/she wants to book for, different from the previous. 3. The user clicks on “Confirm” button. |
| Exit Conditions | The user has booked a visit to the selected store starting from the selected time |
| Exceptions | - |

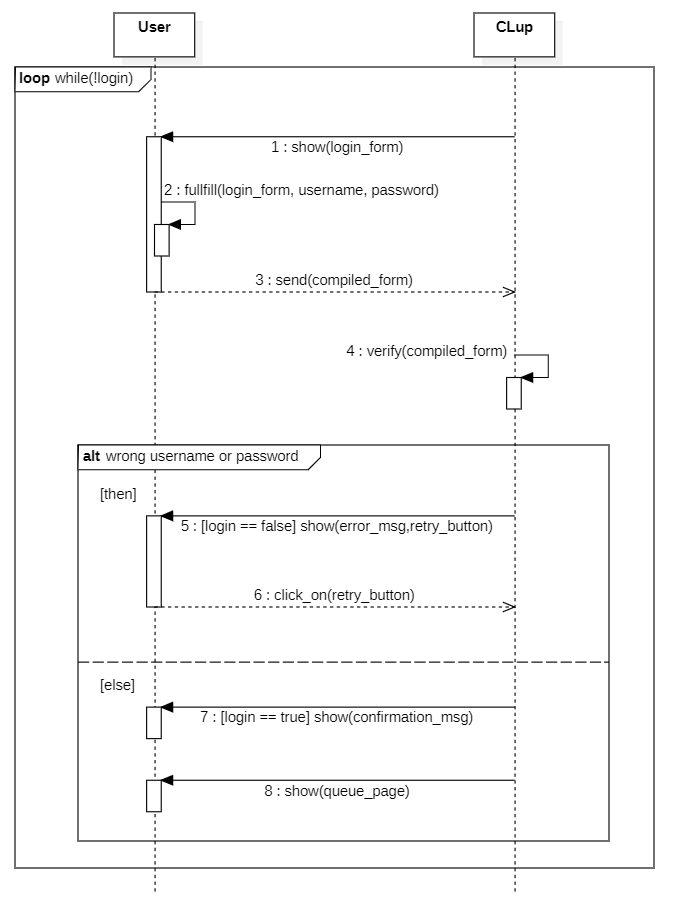
|  |  |
| --- | --- |
| Name | Select time of visit |
| Actor | User |
| Entry Conditions | The user has booked a visit to a store. |
| Events Flow | 1. The user clicks on the “Select time” button. 2. The user select the time of his/her visit. 3. The user clicks on the “Confirm” button |
| Exit Conditions | The user has selected the time of his/her visit |
| Exceptions | 1. The time of the visit exceeds the store timetable   All the exceptions are handled by notifying the user and taking him/her back to the select time of visit activity. |

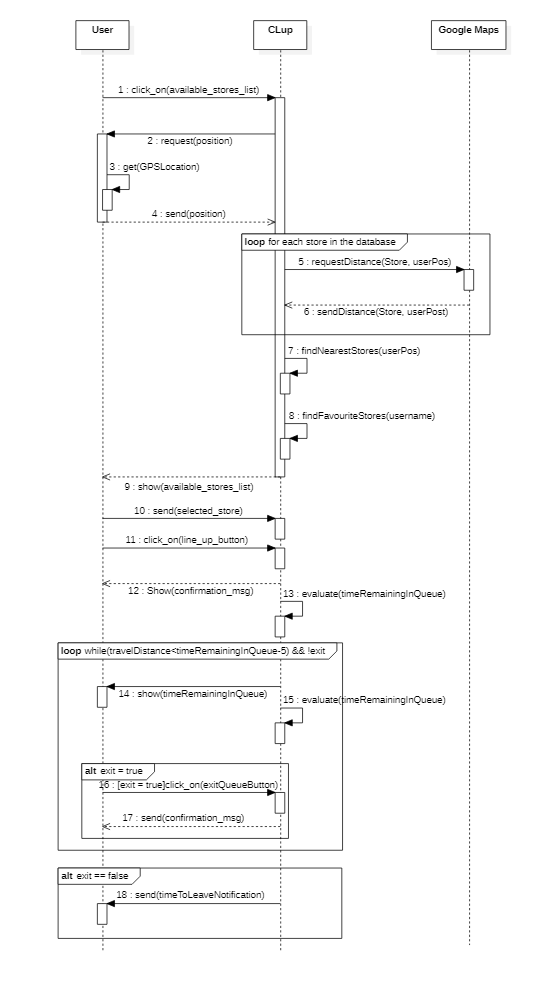
|  |  |
| --- | --- |
| Name | Select items of visit |
| Actor | User |
| Entry Conditions | The user has booked a visit to a store. |
| Events Flow | 1. The user clicks on the “Select items” button. 2. The user select the departments of his/her visit. 3. The user clicks on the “Confirm” button |
| Exit Conditions | The user has selected the items of his/her visit |
| Exceptions | 1. The store doesn’t sell at least one selected item   All the exceptions are handled by notifying the user and taking him/her back to the select items of visit activity. |

|  |  |
| --- | --- |
| Name | Retire from a queue |
| Actor | User |
| Entry Conditions | The user has lined up or booked a visit to a store |
| Events Flow | 1. The user clicks on the “Profile” button. 2. The user clicks on the “Active Book” button. 3. The user clicks on the “Retire” button |
| Exit Conditions | The user has exited the queue of the store he has previous selected |
| Exceptions | 1. The user has already entered the store   All the exceptions are handled by notifying the user and taking him/her back to the active book screen. |

*Sequence Diagram*







*3.1.2 – Store Manager*

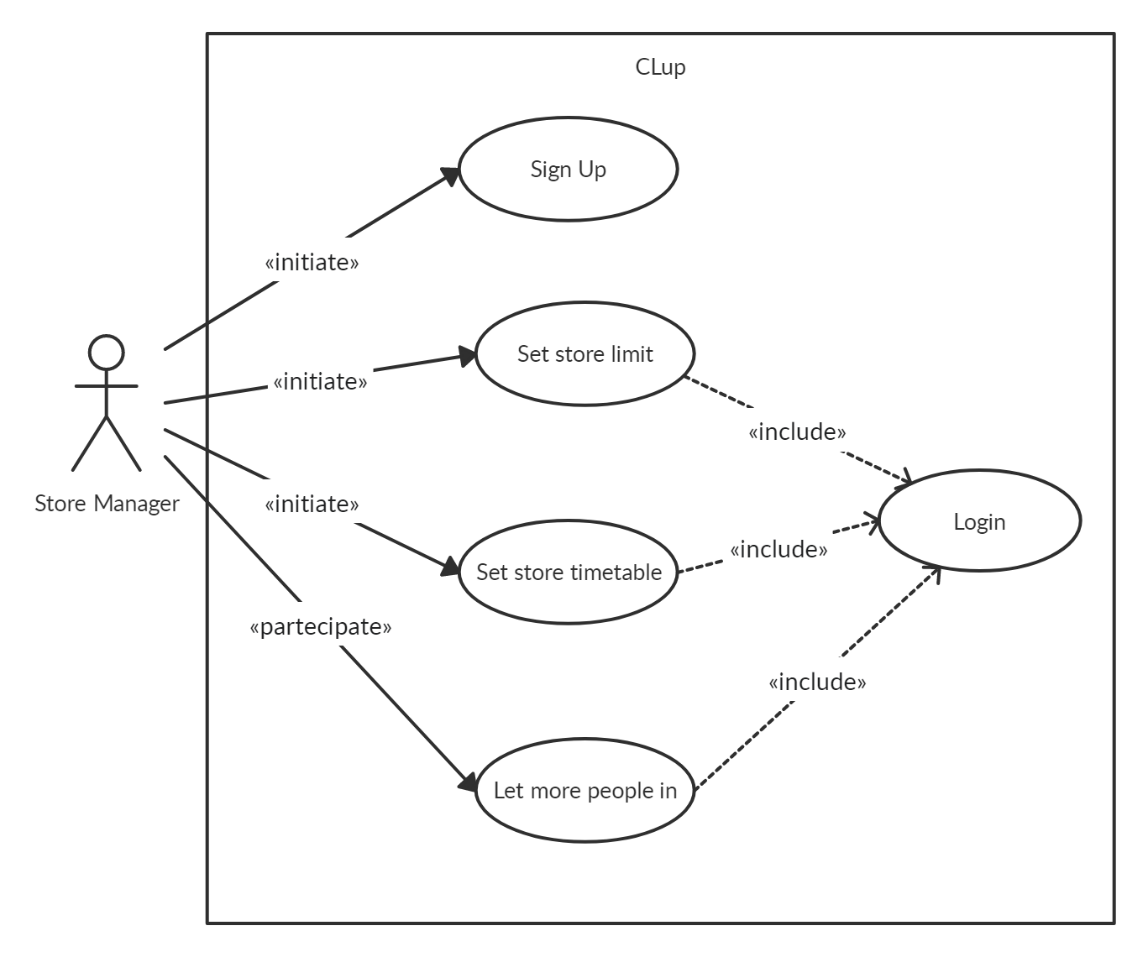
*Scenario 1*

While the emergency is going on, due to this, Italian Prime Minister issues DPCM that handles lockdowns and red zones. Fabio, a Carrefour store manager, reading the last issued DPCM that puts more strict rules (in particular, for the crowds in grocery stores). He must find a solution to guarantee the sanitary measures, the correctness of the queue and that in the store can access a determined number of people.  
Therefore, he decides to use CLup because he can set the maximum number and changing it every time he wants and the system will register this value and it will manage queue according to that.

*Scenario 2*

Riccardo is a Coop store manager. He decides to set as maximum number of admissible people in store a value that’s 5 less than what is written in the last DPCM because he’s aware that more people can go in the same grocery section. Thanks to the features of CLup, the people who choose to line up or to book a visit, select different type of items. In this case, the system calculates that the number of customers inside the store could be increased by 2 (due to their item choice and also respecting all the safety measures) and it communicates this to the store manager. Now Riccardo can choose to increment the number of available spaces for this case, in total safety. After that, the value will automatically return to the initial one.

*Use Case Diagrams*



*Use Cases*

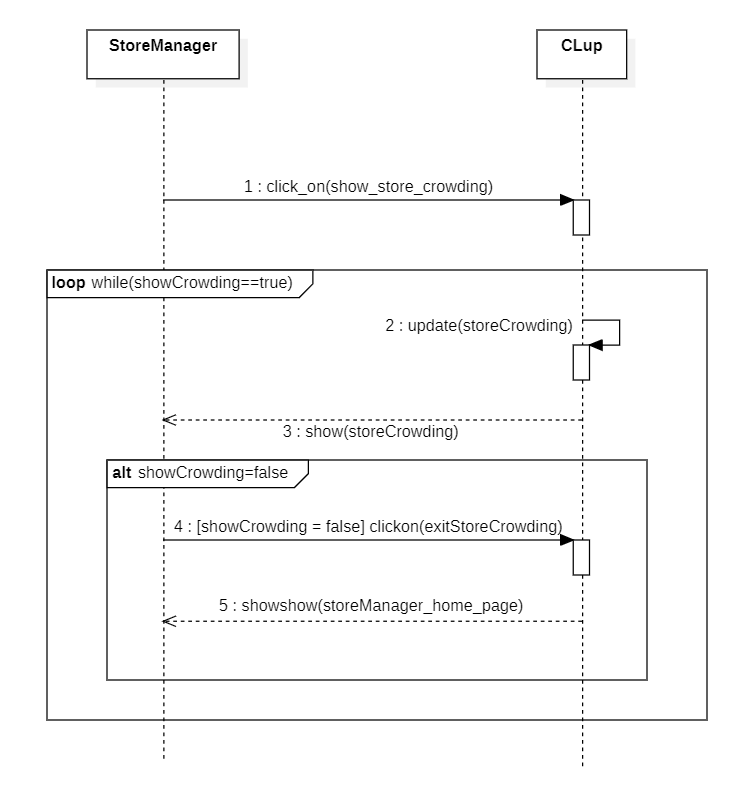
|  |  |
| --- | --- |
| Name | Sign Up |
| Actor | Store Manager |
| Entry Conditions | The store manager has opened the app on his/her device |
| Events Flow | 1. The store manager chooses the “Sign up” option. 2. The store manager fills the mandatory fields. 3. The store manager fills the optional fields with not mandatory data. 4. The store manager chooses the confirmation option. 5. The system saves the data. |
| Exit Conditions | The store manager is registered and the system has his data stored. |
| Exceptions | 1. The store manager is already signed up. In this case the system warns the user and suggests him/her to login. 2. The store manager didn’t fill all of the mandatory fields with valid data. 3. The username is already taken. 4. The e-mail is already registered.   All the exceptions are handled by notifying the store manager and taking him/her back to the sign up activity. |

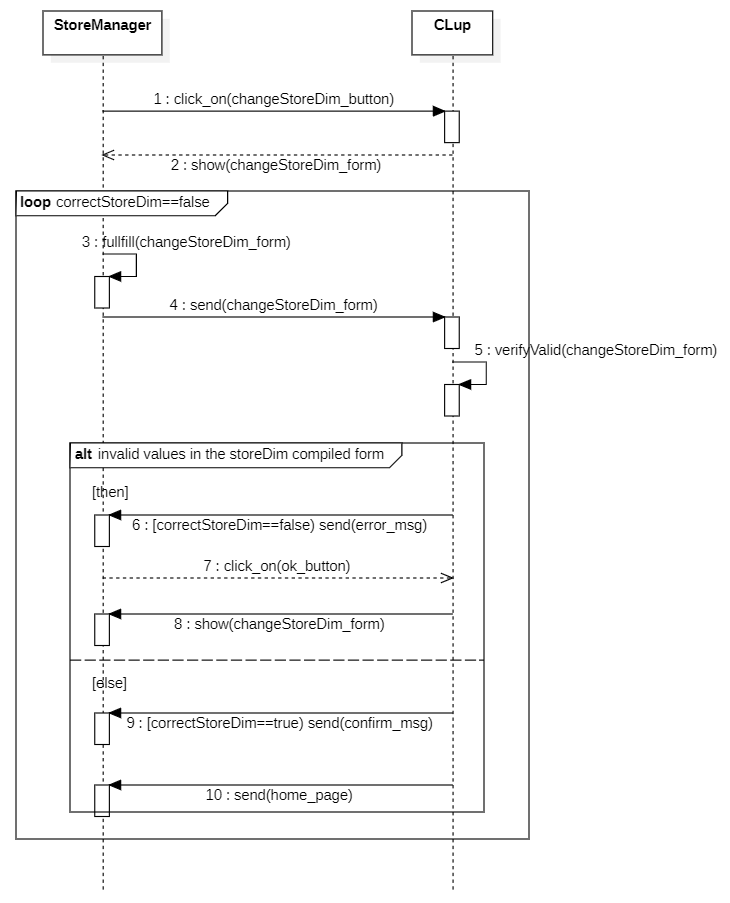
|  |  |
| --- | --- |
| Name | Login |
| Actor | Store Manager |
| Entry Conditions | The store manager is previously successfully signed up and has the application installed on his/her device. |
| Events Flow | 1. The store manager clicks on the “Login” button. 2. The store manager enters username and password in the respective fields. 3. The store manager clicks on the “Confirm” button. |
| Exit Conditions | The store manager is logged in and the system take him to the home page |
| Exceptions | 1. The store manager enters invalid Username. 2. The store manager enters invalid Password.   All the exceptions are handled by notifying the store manager and taking him/her back to the login activity. |

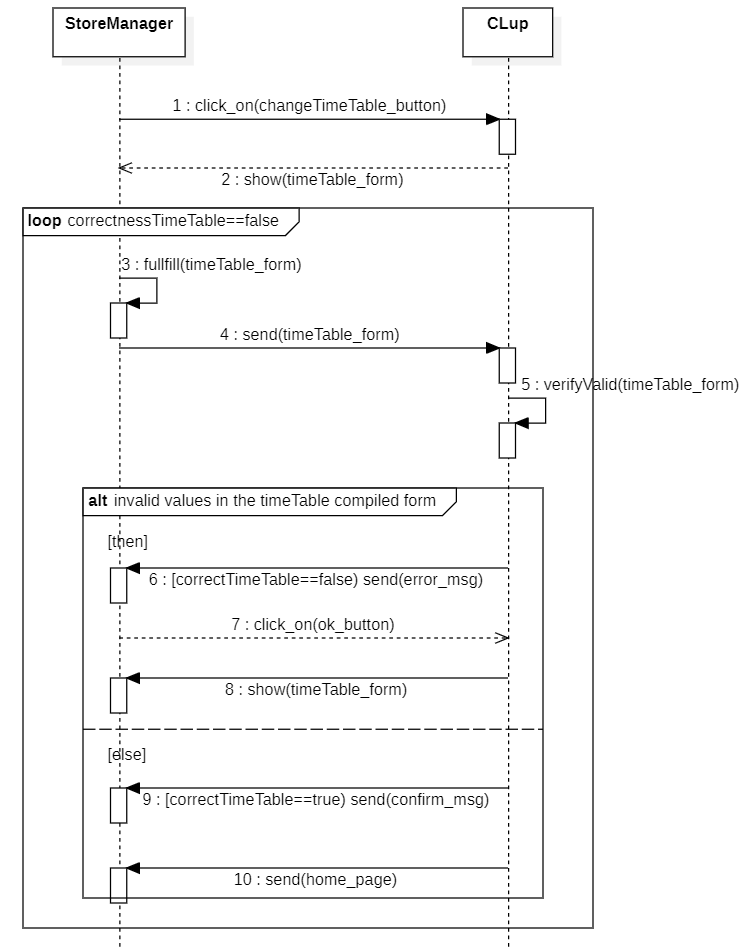
|  |  |
| --- | --- |
| Name | Set store limit |
| Actor | Store Manager |
| Entry Conditions | The store manager has already logged in. |
| Events Flow | 1. The store manager clicks on the “People Limit” button. 2. The store manager enters the limit. 3. The store manager clicks on the “Confirm” button. |
| Exit Conditions | The new store limit is set and saved in store manager profile |
| Exceptions | 1. The new store limit is either zero or negative   All the exceptions are handled by notifying the store manager and taking him/her back to the set store limit activity. |

|  |  |
| --- | --- |
| Name | Set store timetable |
| Actor | Store Manager |
| Entry Conditions | The store manager has already logged in. |
| Events Flow | 1. The store manager clicks on the “Timetable” button. 2. The store manager enters the timetable. 3. The store manager clicks on the “Confirm” button. |
| Exit Conditions | The new store timetable is set and saved in store manager profile |
| Exceptions | 1. Opening time is after closing time |

*Sequence Diagram*







*3.2 Requirements*

**G1: The system must allow customers to line up (on and offline):**

* **R1**: The application allows customers to retrieve a number only if it is possible to enter the store
* **R2**: The number is unique in the queue
* **R3**: The last person that retrieves a number is the last person who will enter the store
* **R4**: The system frees customers number from the queue if they’re late more than 5 minutes
* **R5**: When a customer enters the store he/she is removed from the queue
* **R6**: The system must allow store managers to change the size of the queue
* **R7**: The system must calculate the average flux of people in the store in a certain time
* **R8**: The system must show the number of the position in the queue of the customer
* **R9**: The system must update the queue when an active customer exits (e.g. the first customer in the queue may enter)
* **R10**: The customer who retrieves a number is inserted in the queue
* **R11**: If there is a free space available in the store and no online customer can reach the store in time, the first hand out customer in the queue can enter the store
* **D1**: The customers will retrieve 1 number at a time
* **D13**: The internet connection works properly without failure

**G2: The system must allow store managers to regulate the influx of people in the building**

* **R12**: The application limit access to the store if the number of people is equal to the maximum possible
* **R13**: The system must allow store managers to monitor the entrances
* **R14**: The system must allow store managers to monitor the leavings
* **R15**: The system must allow store managers to change the limit of people in the store
* **D2**: The store managers set the maximum number of people the store can let in simultaneously
* **D3**: The technology to monitor entrances and leavings never fail
* **D4**: The number of people in the store is a positive integer
* **D5**: The customers leave the store using the same ticket they used to enter
* **D13**: The internet connection works properly without failure

**G3: The system must provide customers with an estimate of the waiting time**

* **R16**: The system must keep track of the average time of permanence of customers in the store
* **R17**: The system sends a notification if the new waiting time is less than 20 minutes in case of missed customers
* **D13**: The internet connection works properly without failure

**G4: The system must advice the customers that’s theirs turn to enter**

* **R18**: The system must send a notification to the first customer in the queue when there is at least a free space available in the store
* **R19**: When a customer enter the store his/her number is removed from the queue
* **R20**: When a customer enter the store his/her number is inserted into the store active customers
* **D13**: The internet connection works properly without failure

**G5: The system must allow customer to book a visit in the store**

* **R21**: The system provides the user a list of the stores open at that time if the preferred one is unavailable
* **R22**: The system permits the user to select which store he wants to book a visit
* **R23**: The system permits the user to select when he/she will go to the store
* **R24**: The system confirms the user the acceptance of the asked book
* **R25**: In case of confirmed book the system must insert the user in the queue of the selected store
* **R26**: The system must allow customers to specify the time of their visit
* **R27**: The system must allow customers to specify the categories of the items they want to buy
* **R28**: The system include suggestions of alternative slots to customers if the selected time isn’t available
* **R29**: The system must suggest different stores of the same or even different chain if the preferred one isn’t available
* **R30**: The system must allow customers to select a preferred store
* **D6**: The customers don’t book multiple visits in the same time slots of the same day
* **D7**: The customers buy items of the categories they have selected
* **D13**: The internet connection works properly without failure

**G6: The system must communicate the store manager the possibility to let in the store more people than the limit**

* **R31**: The system allow store manager to define the zones of the store
* **R32**: The system allow store manager to define the maximum number of people for every zone in the store
* **R33**: The system must provide the store manager the number of customers for every zone at the moment
* **R34**: The system must provide the store manager how much he can increase the number of people in the store based on the customers route
* **R35**: If a customer doesn’t specify any route it’s like he/she will visit every zone of the store
* **D8**: The store manager set the zones of the store
* **D9**: The store manager set the maximum number of people for every zone in the store
* **D10**: The customers buy items of the categories they have selected
* **D13**: The internet connection works properly without failure

**G7: The system must alert the customers when they have to leave to arrive in time at the store**

* **R36**: The system must calculate the distance between the customer and the store
* **R37**: The system must acquire the customer position
* **R38**: The system must send a notification if the waiting time of the queue is less than 5 minutes plus the arrival time to the store from their position
* **D11**: The devices that acquire users’ position provide location with an error of 20 meters at most
* **D12**: The system knows the position of every registered store
* **D13**: The internet connection works properly without failure

*Traceability Matrix*

|  |  |
| --- | --- |
| **Requirement** | **Use Case** |
| R1 | Book a Visit  Line Up |
| R2 | Book a Visit  Line Up |
| R3 | Line Up |
| R4 | Book a Visit  Line Up |
| R5 | Book a Visit  Line Up |
| R6 | Set Store Limit |
| R7 | Line Up |
| R8 | Line Up |
| R9 | Book a Visit  Line Up |
| R10 | Book a Visit  Line Up |
| R11 | Line Up |
| R12 | Book a Visit  Line Up |
| R13 | Set Store Limit |
| R14 | Set Store Limit |
| R15 | Set Store Limit |
| R16 | Notify Turn |
| R17 | Notify Turn |
| R18 | Notify Turn |
| R19 |  |
| R20 |  |
| R21 | Book a Visit |
| R22 | Book a Visit |
| R23 | Book a Visit |
| R24 | Book a Visit |
| R25 | Book a Visit |
| R26 | Select Time of Visit |
| R27 | Select Items of Visit |
| R28 | Suggest other Slot |
| R29 | Suggest other Store |
| R30 | Select Favourite Store |
| R31 | Set Store Limit |
| R32 | Set Store Limit |
| R33 | Set Store Limit |
| R34 | Let more People In |
| R35 |  |
| R36 | Estimate Arrival Time |
| R37 | Estimate Arrival Time |
| R38 | Estimate Arrival Time |

*3.3 – Performance Requirements*

The system has to be able to serve a great number of users simultaneously. It has to guarantee quick, reactive and correct responses.

*3.4 – Design Constraints*

*3.5 – Software System Attributes*

*3.5.1 – Reliability*

The system must run continuously without interruptions, so it must have a frequency of failures of 0.01%. To make the system more fault tolerant redundancy might be applied so the central database and the running processes might be duplicated; there will also be done reliability testing for this purpose.

*3.5.2 – Availability*

The System must be available 99.8% of the time, this can be obtained by also implementing the redundancy and fault tolerance mentioned before in the Reliability section.

*3.5.3 – Security*

Users password and GPS location, which are sensitive informations, must be protected by encryption when they are sent through the network; also the central database containing the data reside must be protect (through techniques of Database Security) to avoid external and internal attack and also to handle malfunctions of the hardware.

*3.5.4 – Maintainability*

The development of the application must be done in a modular way to make easier to fix and modify it in the future and also to make the system capable to facilitate addition of new features and options to meet new possible requirements in the future; for that there might be used Design patterns with standard terminology. To improve the maintainability the code will be provided with Documentation.

*3.5.5 – Compatibility*

This is a mobile application, so it needs to be compatible with Android and IOS systems to work in the 99% of the smart phones.

**4 – Formal Analysis Using Alloy**

**5 – Effort Spent**

Student 1

|  |  |
| --- | --- |
| **Chapter** | **Hours** |
| 1 – Introduction |  |
| 2 – Overall Description |  |
| 3 – Specific Requirements |  |
| 4 – Formal Analysis Using Alloy |  |

Student 2

|  |  |
| --- | --- |
| **Chapter** | **Hours** |
| 1 – Introduction |  |
| 2 – Overall Description |  |
| 3 – Specific Requirements |  |
| 4 – Formal Analysis Using Alloy |  |

Student 3

|  |  |
| --- | --- |
| **Chapter** | **Hours** |
| 1 – Introduction |  |
| 2 – Overall Description |  |
| 3 – Specific Requirements |  |
| 4 – Formal Analysis Using Alloy |  |

CHOICES AND REASONS:

1. WHY DID YOU PUT INTERFACES IN SECTION 2 AND NOT IN THE 3rd?   
   🡪 We choose it because the customer and the user are a part of the RASD audience: they’re interested in high-level functions and not in specific requirements. Surely, interfaces are a key point for customer (because they will use them to manage the system) and for customer (bad interfaces implied a bad app/system according to them [and to us]). So why we have to put in the third part that are more technical and less interesting for them. It’s a customer/user-oriented choice.
2. ASSUMPTION: booking 🡪 select store 🡪 select time slot 🡪 if not available: suggest other slots in same store 🡪 if rejected: suggest other stores available in previous time selected.