RASD

Requirements Analysis and Specification

POLITECNICO
MILANO 1863

Paolini Giuseppe
Marticola:968534

Pellini Riccardo
Matricola:968866

Panzeri Matteo
Matricola:965568

A.Y. 2020-2021

Professore:
Rossi Matteo

Index

1 Introduction

1.1 Purpose

The requirement analysis and specification document contains the description of the CLup application.

The document contains the analysis of the functionalities of the application and the world of application which is described via models, use cases and scenarios.

The problem analysis extends also on functional and nonfunctional requirements of the application.

This document will track the application development.

1.2 Scope

The application allows, given its user friendly interface, a potential customer of the supermarket to reserve a “temporal slot”.

The user can virtually line up at the store, the application will retrieve the first available temporal slots which adapts to the inserted additional information, in fact the user can insert the time he or she thinks will spend in the supermarket and the groceries that he or she intends to purchase. Than the application will retrieve the best slot which satisfies the user’s requests .

Another option is to receive suggestions from the application, after having requested to reserve a slot the user will receive a list of possible temporal slots (a day and an hour) to choose from.

In any case the application, given the user location, will notify him about the correct time to leave his position and go to the supermarket.

In case that a person is not able to use the application, she can go to the supermarket and ask there for a reservation.

The additional information that a user can insert during a request helps the application to grant a better user experience and to allow more people into the supermarket granting at the same time the safety of the customers.

Long term users benefit from another possibility which is the suggestion of reservation, the system can detect patterns in the users’ shopping behaviors and ask them if they want to book a place for that particular spot.(for example the user goes to the supermarket at the same time and at the same day every week).

In the unfortunate event that no temporal spots are available for the selected supermarket the system can suggests slots of other stores in the same supermarket chain.

1.2.1 World phenomena

WP1	User goes to the supermarket
WP2	User gets a ticket at the store
WP3	User occupies a place in the store
WP4	User goes to a unit which he told he wouldn't go to
WP5	Stores have an opening time
WP6	Stores have a closure time
WP7	A store employee works at the utility desk of the store

1.2.2 Shared phenomena

SP1	User sends a request to go to the supermarket
SP2	Store employee books a visit for the customers
SP3	System sends a QR code to the user
SP4	User shows a QR code in order to enter the supermarket
SP5	User books a visit
SP6	User specifies the approximate duration of the visit
SP7	User chooses among product categories
SP8	Stores specifies to the system the maximum amount of people allowed

1.2.3 Goals

G1	Lining up mechanism is effective
G2	Customer can make requests by the utility desk of the store
G3	Maximize the number of people in the store
G4	Balance out the number of people in the stores

1.3 Definitions, acronyms and abbreviations

1.3.1 Definitions

Temporal slot	Amount of time in which a customer can access the supermarket. Defined by a day and a starting hour.
Ticket	Receipt sent to the user after a line up or a reservation it contains the QR code.
Unit	Defined section of the supermarket which contains the same categories of goods as pasta, vegetables and fruit, meet etc.
Available spots	Given an unit the Available spots are the number of people who can stay in that unit at the same time given a day and an hour.
Store employee	Employee of a supermarket who will act as a proxy between who wants to reserve a slot without using the application.
Line up	Process via an user reserve the first available slot.
Reservation	Process via an user selects the temporal slots given a list of them
Utility desk	Physical place outside the supermarket where people can go in order to reserve a slot if they cannot use the application.
QR code	Code in the ticket which is used to access and go out a supermarket it is unique and related to a single reservation.
QR reader	Device placed at every entrance and exit of a supermarket it is used to detect and read the QR codes of the users.
Time of visit	Time that the user plans to spend in the supermarket if it is not specified it will be assumed as an hour.
Notification	Message sent by the system to the user when the system thinks the user should leave his address in order to reach the supermarket on time.

1.3.2 Acronyms

GPS	Global Positioning System

1.3.3 Abbreviations

Slot	Temporal Slot
Reservation	Sometimes it is also used for the line up
WPn	World phenomena number n
Gn	Goal number n
SPn	Shared phenomena number n
Dn	Domani assumption number n

1.4 Revision history

1.5 Reference Document

This RASD document is based on the
Requirement Engineering and Design Project: goal, schedule, and rules document gived by the professor.

1.6 Document Structure

- Chapter 1 gives a description of the document his objective and how is structured, it also specifies which are the project objectives and describes the world where the project will work.
- Chapter 2 gives an overall description of the project which goes from the UML which describe the classes (it is a very high level description) to the description of the user and the project functionalities.
- Chapter 3 provides a description of the various interfaces used in the project, hardware, software etc.
  Here we can also find use cases, sequence diagram and scenarios.
- Chapter 4 contains the formal definition of the model it is done via Alloy definition language.
- Chapter 5 contains the effort spent by each member of the group.
- Chapter 6 contains references to external documents used in this project.

2 Overall Description

2.1 Product perspective

2.1.1 UML Description

The following UML class diagram describe an high level version of the principal classes that we need to implement in order to create the project. Some classes may be missing but the one which are present grant the fulfillment of the requirements.

Among all the possible comments that can be done about the UML the following are the most relevant.

A user can create a reservation only if she or he is registered and logged. When the user asks to reserve a slot the system provides her/him a list of time slots to choose from. When a time slot is selected the reservation is complete and the QRCode is generated.

The user will use the QRCode to access and go out the supermarket, the system ,thanks to the QR readers placed outside each entrance and exit, can track the costumer's accesses and compute the time that they spend in the supermarket.

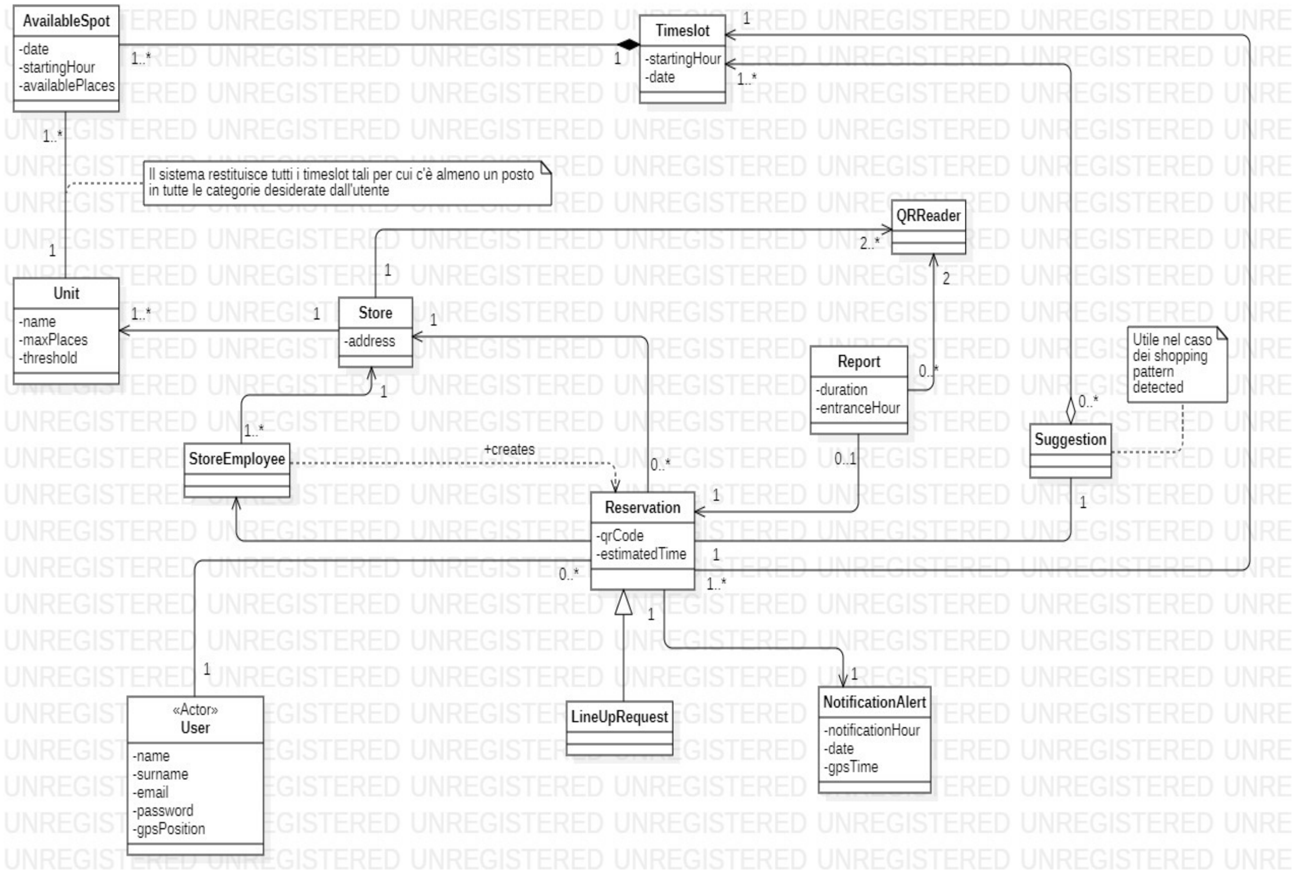
When an user completes a visit a report is created. Thanks to the reports, the system can detect, shopping patterns or compute the average time that the user spends in the supermarket.

A timeslot is an aggregation of available spots which have the same starting hour and the same date and they correspond to units placed in the same supermarket. The division of the supermarket in units aims to maximize the number of people in the supermarket but at the same time observing the social distance measures.

The employee is associated with the supermarket where he works and he can do multiple reservations in the same time slot while the user cannot do that.

The reservations made by the employee don't produce a report: we do not keep track of reservations made by the store.

The notification is associated with a reservation and it is sent to the user by the system when it thinks that the user should leave his address in order to reach the supermarket.



2.2 Product functions

Line up

One of the main features of the application is to regulate the flow of people inside supermarkets by helping to respect the rules imposed for the covid19 emergency situation.
Among the most important ones there are: limited number of people inside a building, distance of at least one meter between one person and another, avoiding the presence of people in line outside the building.

To achieve this goal, users who want to go shopping, instead of physically going to the shop and queue up, they can queue up via apps. It is essential that each user arrives at the supermarket in time in order to avoid queues outside, it is for this reason that the system indicates to each user the waiting time before his turn and the time he takes to get to the supermarket from his home.

Once the user has queued via app, he receives a QR code generated by the system which must be shown at the entrance. For users who do not have the appropriate technological devices to use the app there is the possibility to queue and collect their ticket directly from the supermarket desk.

Book a visit

Users also have the possibility to book a visit to the supermarket on the date and time they prefer directly from the phone, entering the approximate time of permanence in the building and the exact list of products they intend to buy.
Thanks to this data, the system is able to maximize the capacity of people inside a shop, planning visits efficiently.

Slots suggestion

In order to balance the number of people in the various shops, the system is able to suggest free or more appropriate time slots to each specific user and to periodically notify them of the presence of free slots of various durations in the following days. The application also recommends different shops in the same chain if the selected shop is not available.

2.3 User characteristics

The actors of the application are the following:

Users

People who use the app on their phone to queue or to book a visit to the supermarket.

Supermarket employees

People who act as proxies between the system and the user.
They release tickets to users who ask to queue directly in the supermarket and not through the application.

2.4 Assumptions, dependencies and constraints

2.4.1 Domain assumptions

D1	People respect social distance
D2	Only one person per QR code is allowed to enter the supermarket
D3	GPS provides the exact location with an error of 10 meters at most
D4	The supermarket is divided in units.
D5	In each unit of the supermarket there are the same categories of goods.
D6	Customers are always on time.
D7	A limited number of people are permitted in closed space
D8	Waiting time must be greater than or equal to travel time to the store
D9	Stores have employees at the utility desk who work as proxies between users and the system
D10	Stores have QR code readers at the entrance and at the exit
D11	Customers without smartphones don’t make multiple requests in the same slot
D12	If a physical unit is placed in a store it is placed in no other stores

3 Specific Requirements

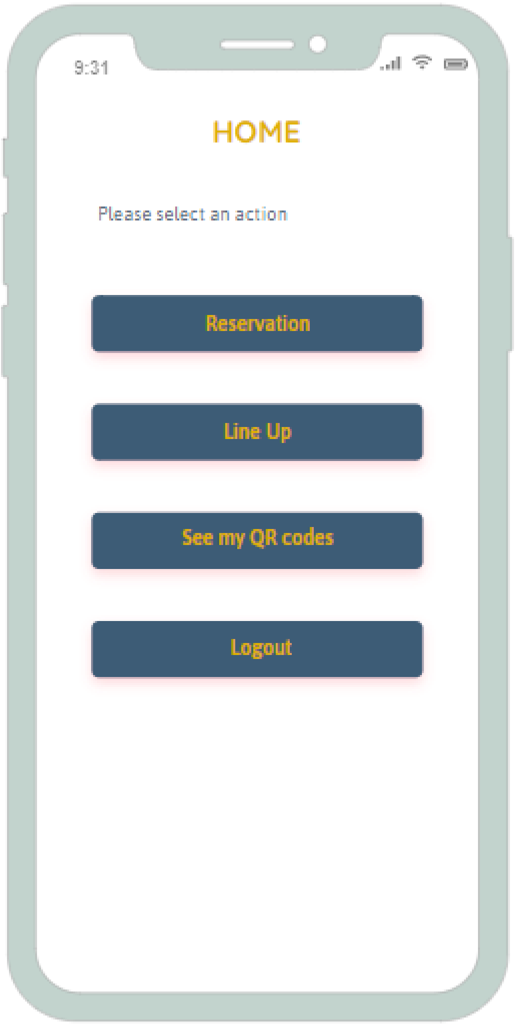
3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface of CLup must be simple and minimal to grant usability.

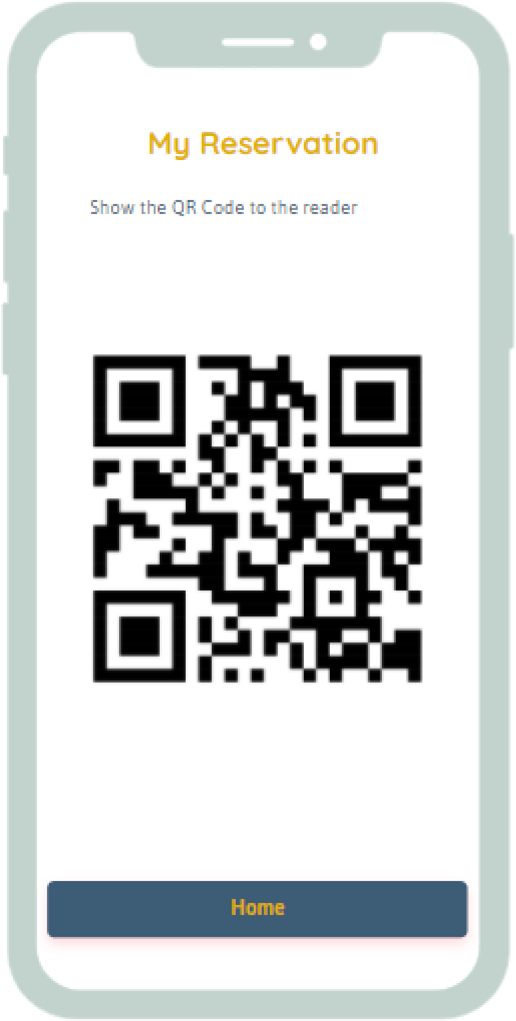
The application will work using different pages which will grant the accomplishment of all the functions of the application.

The following mockups represent some of the most important parts of the application. (Other mockups can be found in the Design Document)



HOME

Please select an action


My Reservation

Show the QR Code to the reader

3.1.2 Hardware Interfaces

The application is installed on the user’s smartphone which is required to have an available internet connection and a working GPS.

At every entrance and exit of the supermarket there must be QR readers that can scan a reservation’s QR code and allows a user to enter the building, when the user wants to go out she will show the code again in order to allow the system to collect the time she has spent in the supermarket.

3.1.3 Software Interfaces

The application will use an external service to detect the GPS position of the user.

The application must be able to communicate with the central system in order to send the users’ requests, it has to be able to receive the system messages and store the received QR Codes.

The System in the supermarket must be able to detect a QR Code read from a code reader and compare it with the ones generated by the system itself.

The communication to the system is made via the internet connection.

3.2 Functional Requirements

3.2.1 List of Requirements

R1	The system allows the user to request to line up, in order to go to the store.
R2	The system allows the user to book a visit at the store.
R3	The system doesn't allow the user to line up or reserve a visit twice in a particular slot.
R4	The system allows the store employee to make multiple requests in particular slot.
R5	The system provides customers with a reasonably precise estimation of the waiting time.
R6	The system alerts customers taking into account the time they need to get to the shop from the place they currently are.
R7	The system can estimate the time of visit of a client.
R8	The system generates a QR code to allow people to enter and exit the store.
R9	The system fixes a standard time of visit for customers who approach for the first time.
R10	The system allows the user to insert the estimated time of visit.
R11	The system allows the user to insert the categories of goods he or she is going to purchase.
R12	The system fixes a standard list of categories as default.
R13	The system records the time that customers spend in the supermarket.
R14	The system saves the categories bought by the user.
R15	The system suggests alternative temporal slots.
R16	The system suggests alternative stores.
R17	The system notifies users of available temporal slots.
R18	The system allows the user to sign up to the application.
R19	The system allows the user to log in.
R20	The system doesn't save any data of the requests made by the store employee.

3.2.2 Mapping

Goal	Functional Requirement	Domain assumption
G1	R1, R2, R3, R4, R5, R6, R7, R9, R15, R16, R17, R18, R19	D1, D3, D6, D8, D9, D11
G2	R1, R2, R4, R5	D9, D11
G3	R1, R2, R7, R8, R9, R10, R11, R12, R13, R14, R18, R19, R20	D1, D2, D4, D5, D6, D7, D10, D11
G4	R1, R2, R5, R6, R7, R10, R11, R12, R13, R14, R15, R16, R17, R18, R19, R20	D1, D2, D4, D5, D6, D7, D10

G1	Lining up mechanism is effective
R1	The system allows the user to request to line up, in order to go to the store.
R2	The system allows the user to book a visit at the store.
R3	The system doesn't allow the user to line up or reserve a visit twice in a particular slot.
R4	The system allows the store employee to make multiple requests in particular slot.
R5	The system allows the user to request to line up, in order to go to the store.
R6	The system alerts customers taking into account the time they need to get to the shop from the place they currently are.
R7	The system can estimate the time of visit of a client.
R9	The system fixes a standard time of visit for customers who approach for the first time.
R15	The system suggests alternative temporal slots.
R16	The system suggests alternative stores.
R17	The system notifies users of available temporal slots.
R18	The system allows the user to sign up to the application.
R19	The system allows the user to log in.
D1	People respect social distance
D3	GPS provides the exact location with an error of 10 meters at most
D6	Customers are always on time.
D8	Waiting time must be greater than or equal to travel time to the store
D9	Stores have employees at the utility desk who work as proxies between users and the system
D11	Customers without smartphones don't make multiple requests in the same slot

G2	Customer can make requests by the utility desk of the store
R1	Stores have employees at the utility desk who work as proxies between users and the system
R2	Customers without smartphones don’t make multiple requests in the same slot
R4	The system allows the user to request to line up, in order to go to the store.
R5	The system allows the user to book a visit at the store.
D9	Stores have employees at the utility desk who work as proxies between users and the system
D11	Customers without smartphones don’t make multiple requests in the same slot

G3	Maximize the number of people in the store
R1	The system allows the user to request to line up, in order to go to the store.
R2	The system allows the user to book a visit at the store.
R7	The system can estimate the time of visit of a client.
R8	The system generates a QR code to allow people to enter and exit the store.
R9	The system fixes a standard time of visit for customers who approach for the first time.
R10	The system allows the user to insert the estimated time of visit.
R11	The system allows the user to insert the categories of goods he or she is going to purchase.
R12	The system fixes a standard list of categories as default.
R13	The system records the time that customers spend in the supermarket.
R14	The system saves the categories bought by the user.
R18	The system allows the user to sign up to the application.
R19	The system allows the user to log in.
R20	The system doesn’t save any data of the requests made by the store employee.
D1	People respect social distance
D2	Only one person per QR code is allowed to enter the supermarket
D4	The supermarket is divided in units.
D5	In each unit of the supermarket there are the same categories of goods.
D6	Customers are always on time.
D7	A limited number of people are permitted in closed space
D10	Stores have QR code readers at the entrance and at the exit
D11	Customers without smartphones don’t make multiple requests in the same slot

G4	Balance out the number of people in the stores
R1	The system allows the user to request to line up, in order to go to the store.
R2	The system allows the user to book a visit at the store.
R5	The system provides customers with a reasonably precise estimation of the waiting time.
R6	The system alerts customers taking into account the time they need to get to the shop from the place they currently are.
R7	The system can estimate the time of visit of a client.
R10	The system allows the user to insert the estimated time of visit.
R11	The system allows the user to insert the categories of goods he or she is going to purchase.
R12	The system fixes a standard list of categories as default.
R13	The system records the time that customers spend in the supermarket.
R14	The system saves the categories bought by the user.
R15	The system suggests alternative temporal slots.
R16	The system suggests alternative stores.
R17	The system notifies users of available temporal slots.
R18	The system allows the user to sign up to the application.
R19	The system allows the user to log in.
R20	The system doesn’t save any data of the requests made by the store employee.
D1	People respect social distance
D2	Only one person per QR code is allowed to enter the supermarket
D4	The supermarket is divided in units.
D5	In each unit of the supermarket there are the same categories of goods.
D6	Customers are always on time.
D7	A limited number of people are permitted in closed space
D10	Stores have QR code readers at the entrance and at the exit

3.2.3 Use Cases

3.2.3.1 Use Cases Description

REGISTRATION

Actors

User.

Entry conditions

The user has downloaded the application on his device and he/she has an internet connection.

Event flow

1. The user opens the application.
2. The user goes on the registration page.
3. The user compiles all the mandatory fields. (da specificare in Additional specifications)
4. The system saves his data.
5. The system confirms the registration.

Exit condition

The system has successfully saved the user’s data and the user is successfully registered to the application .

Exceptions

1. The user is already present in the system (the same data are saved in the system DB).
2. Not all the required data are inserted by the use.
3. The required data inserted by the user were wrong.

If one of these situations occurs then the process aborts and the application returns to an empty registration page, after having notified the user with an error message.

LOGIN

Actor

User

Entry Conditions

The user is already registered and he/she has an active internet connection.

Event Flow

1. The user accesses the app.
2. The user inserts email and password.
3. The system checks if the user is registered.
4. The system grants access to the user.

Exit conditions

The system moves the user on the shopping reservation page.

Exceptions

The user is not registered. The system moves the user on the registration page. The user inserts the wrong password or email. If one of these situations occurs then an empty login page is opened with the following messages: invalid username or password, please try to login again.

RESERVATION

Actor

User

Entry Condition

User is logged and he/she has a working internet connection.

Event Flow

1. The user requests to reserve a slot.
2. The user puts the time that he/she thinks will spend in the supermarket.
3. The user puts the categories of goods he/she's going to to purchase.
4. The system elaborates the data.
5. The system retrieves all the possible slots in two weeks.
6. The user decides what slot reserve.
7. The system confirms the reservation.

Exit condition

The system gives to the user the QR code.

Exceptions:

The user doesn't put the foreseen time in the supermarket:

1. if it is the first time that the user uses the application, the system uses a default forecast of an hour.
2. If the user is an habitual customer (he has already used the App) then the system computes the forecast making the average of the time spent by the user in the supermarket the previous times.

The user doesn't put the categories of goods he/she wants to buy:

1. The system considers the person wants to buy a good for each category.

The system doesn't find any available slots so it suggests the user another supermarket.

FREE SLOT SUGGESTION

Actor

User, system.

Entry condition

The user is reserving a visit but the system detects that for that given day it is receiving more requests than a given threshold or slots aren't available.

Event Flow

1. The system looks for less crowded slots the preceding and following days .
2. The system suggests those slots to the user.
3. The user accepts one of them.
4. The system generates a unique QR code.
5. The system sends the QR code back to the user.

Exit condition

The user has reserved a visit at the supermarket on another time slot.

Exceptions

The user refuses the suggestion

1. The system confirms the reservation for the previous chosen slot.

The system doesn't find a suitable slot

1. The system doesn't notify the user.

SHOPPING PATTERN DETECTED

Actor

User

Entry Condition

The system can build and forecast the shopping day of the user.

Event Flow:

1. The system detects that a given day is particularly crowded.
2. The system looks for all the users that usually go to the supermarket that day of the week.
3. For the found people in the list that haven't already made a reservation, the system looks for a slot another day.
4. The system sends the found slots to the users selected in the precedent search.
5. The user who receives the suggestion decides to accept and reserve the slot.
6. The user puts the category of goods he/she wants to buy.
7. the system generates the reservation and the QR code.

Exit condition

The system gives to the user the QR code.

Exceptions

No users are found

The system does nothing.

A user rejects the suggestion

The system does nothing.

The system doesn’t find acceptable suggestions

The system does nothing.

REQUEST TO LINE UP - APP

Actors

User.

Entry condition

The user is logged and its smartphone has a working internet connection and active GPS

Event Flow

1. The user clicks on the store he/she wants to go from the list of registered supermarkets.
2. The user decides which means of transport he/she’s going to use
3. The system retrieves the position of the user.
4. The system computes the needed travel time for the user to reach the store.
5. The system finds the first free slot in the waiting queue.
6. The system computes the overall waiting time.
7. The system sends the waiting time to the user.
8. The user confirms the request.

Exit condition

The system generates a unique QR code and sends it back to the user.

Exceptions

The user has not allowed the use of GPS.

The system asks the permits to the user. If the user accepts, the procedure continues; otherwise, the system aborts the transaction and sends the user to the homepage.

The user doesn’t confirm the request.

The system aborts and sends the user back to the homepage.

The system detects that all time slots of the current day are not available

The system invites the user to reserve a visit on another day.

REQUEST TO LINE UP – STORE

Actors

Store employee.

Entry condition

The store has a utility desk with a working internet connection and a customer approaches the utility desk in order to request a ticket.

Event flow

1. The employee sends a request of lining up to the system.
2. The system finds the first free slot in the waiting queue.
3. The system computes the overall waiting time.

4. The system sends the waiting time to the store employee.
5. Store employee tells the customer the waiting time.
6. Store employee confirms the request.

Exit condition

The system generates a unique QR code and sends it back to the store employee, who prints a ticket with the QR code and the expected hour of visit, giving it to the customer.

Exceptions

The store employee is asked to deny the request by the customer

The system aborts the transaction and sends the store employee to the homepage.

The system detects that all time slots of the current day are not available

It signals the problem to the store employee and sends it back to the homepage.

RESERVATION BY THE STORE

Actors

Store employee.

Entry condition

The store has a utility desk with a working internet connection and a customer approaches the utility desk in order to reserve a visit at the store.

Event Flow

1. The store employee asks to reserve a slot.
2. The store employee puts the time that the user thinks will spend in the supermarket.
3. The store employee puts the categories of goods the user's going to purchase.
4. The system elaborates the data.
5. The system retrieves all the possible slots in two weeks.
6. The store employee reserves the slot the customer wants.
7. The system confirms the reservation.
8. The store employee prints a ticket with the unique QR code and the hour of visit.

Exit conditions

The system generates a unique QR code and sends it back to the store employee, who prints a ticket with the QR code and the expected date of visit, giving it to the customer.

Exceptions

The customer doesn't tell the store employee how much time he/she will spend at the supermarket.

A default time of an hour is put in the system.

The customer doesn't tell the store employee which category of goods he/she will buy.

Store employee signals the customer will buy at least one product per category.

There isn't any free slot.

The store employee suggests the user to go to another store.

The store employee is asked to cancel the request of reservation

The system aborts and sends the store employee back to the homepage.

REMINDER NOTIFICATION

Actors

User, system.

Entry condition

The user is in line to go to a supermarket or has reserved a visit and 15 minutes are left until his departure time.

Event flow

1. The system sends a notification to the user.

Exit condition

None.

Exceptions

None.

USER AT THE SUPERMARKET

Actors

User, System.

Entry Condition

The user how has the QR code approaches the supermarket.

Event flow

1. The User shows to the QRCode reader his QRCode
2. The System checks if the User can enter the supermarket
3. The System lets the user enter and saves the entrance time.
4. The User proceeds with his purchases.

Exit Conditions

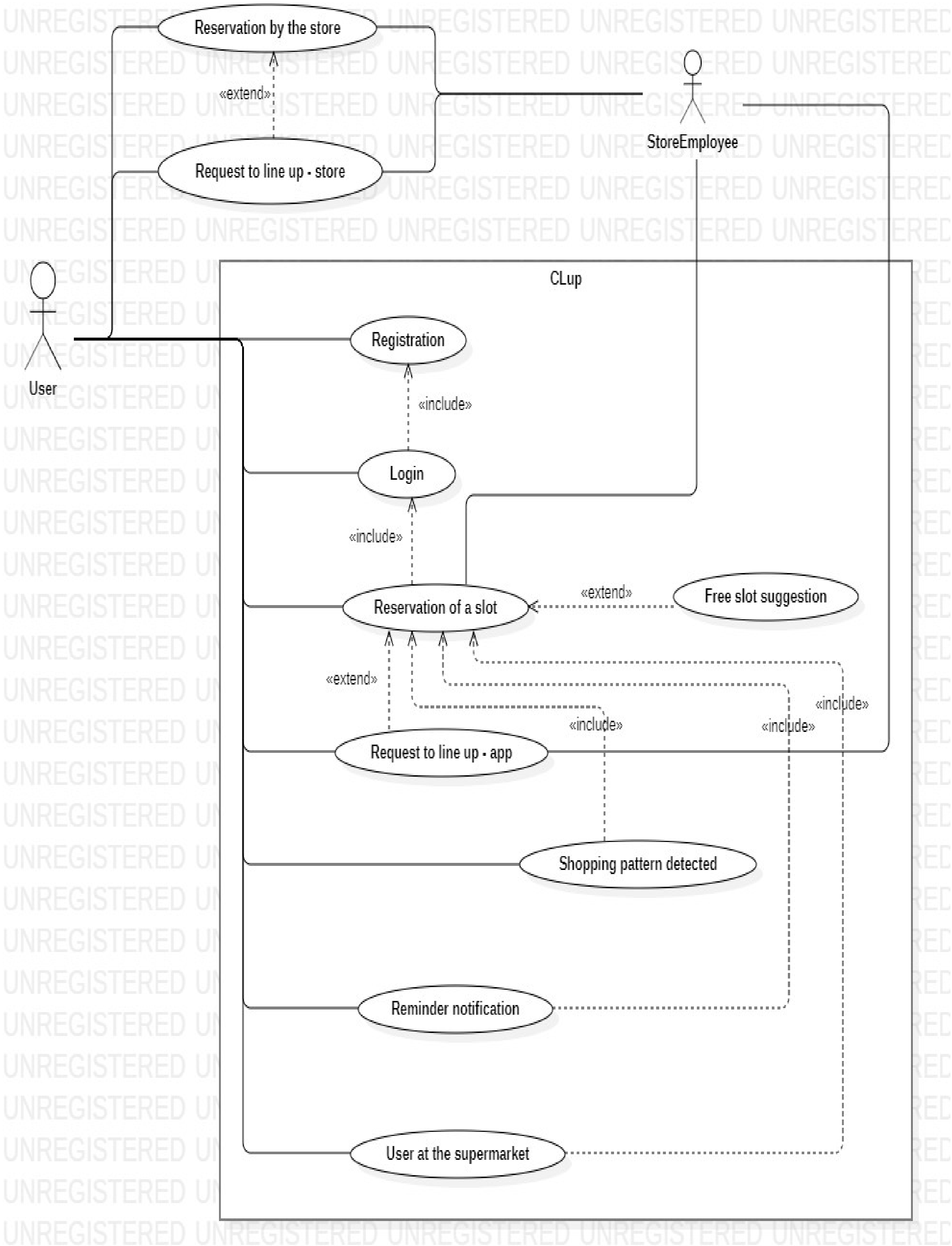
The User approaches the exit and shows the reader the QR Code. The System let the user go out and saves the time spent in the supermarket by the user.

Exceptions

The user has no QR code or his code is wrong.

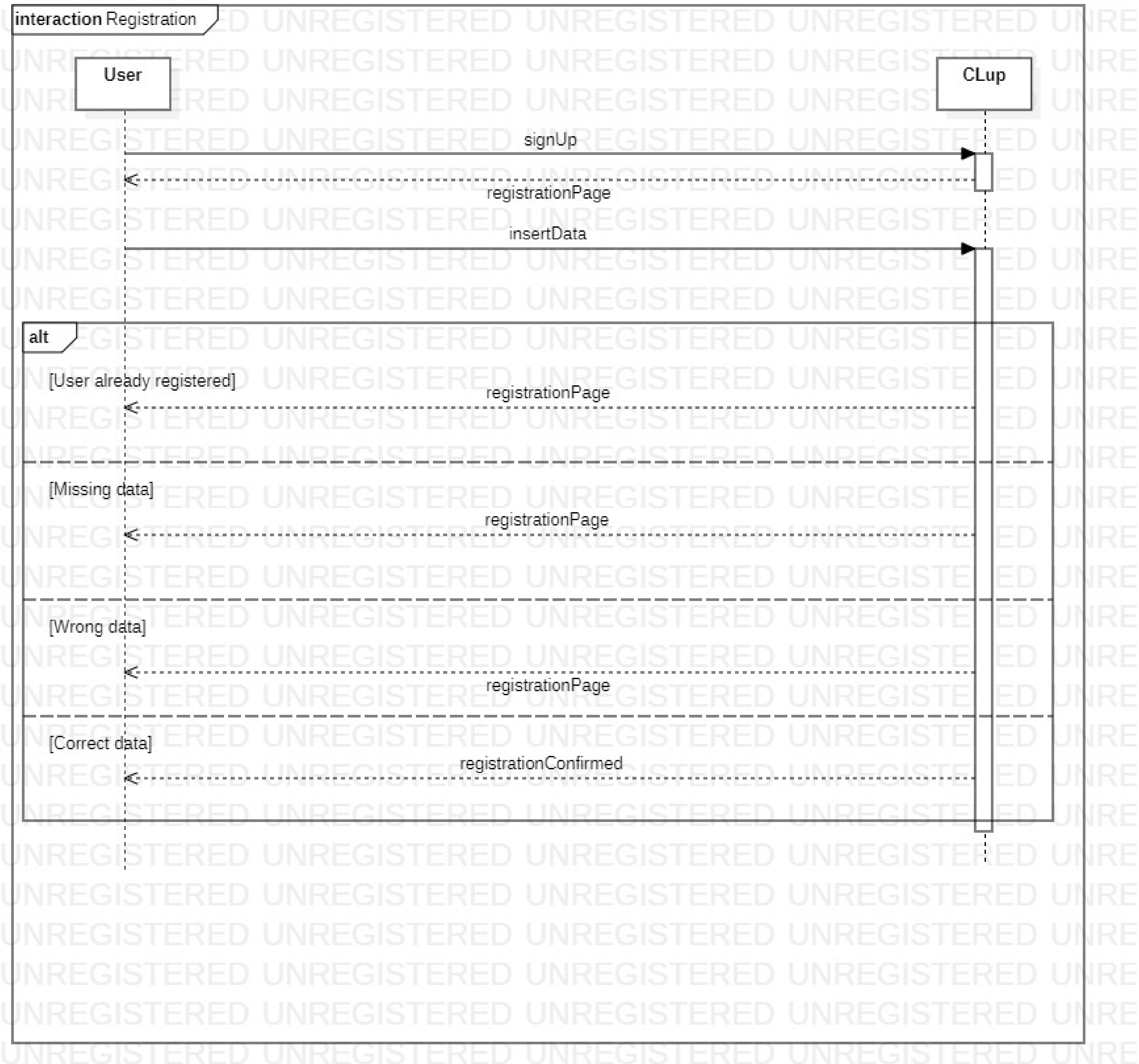
The system doesn’t let him/her go to the supermarket.

3.2.3.2 Use Cases Diagram

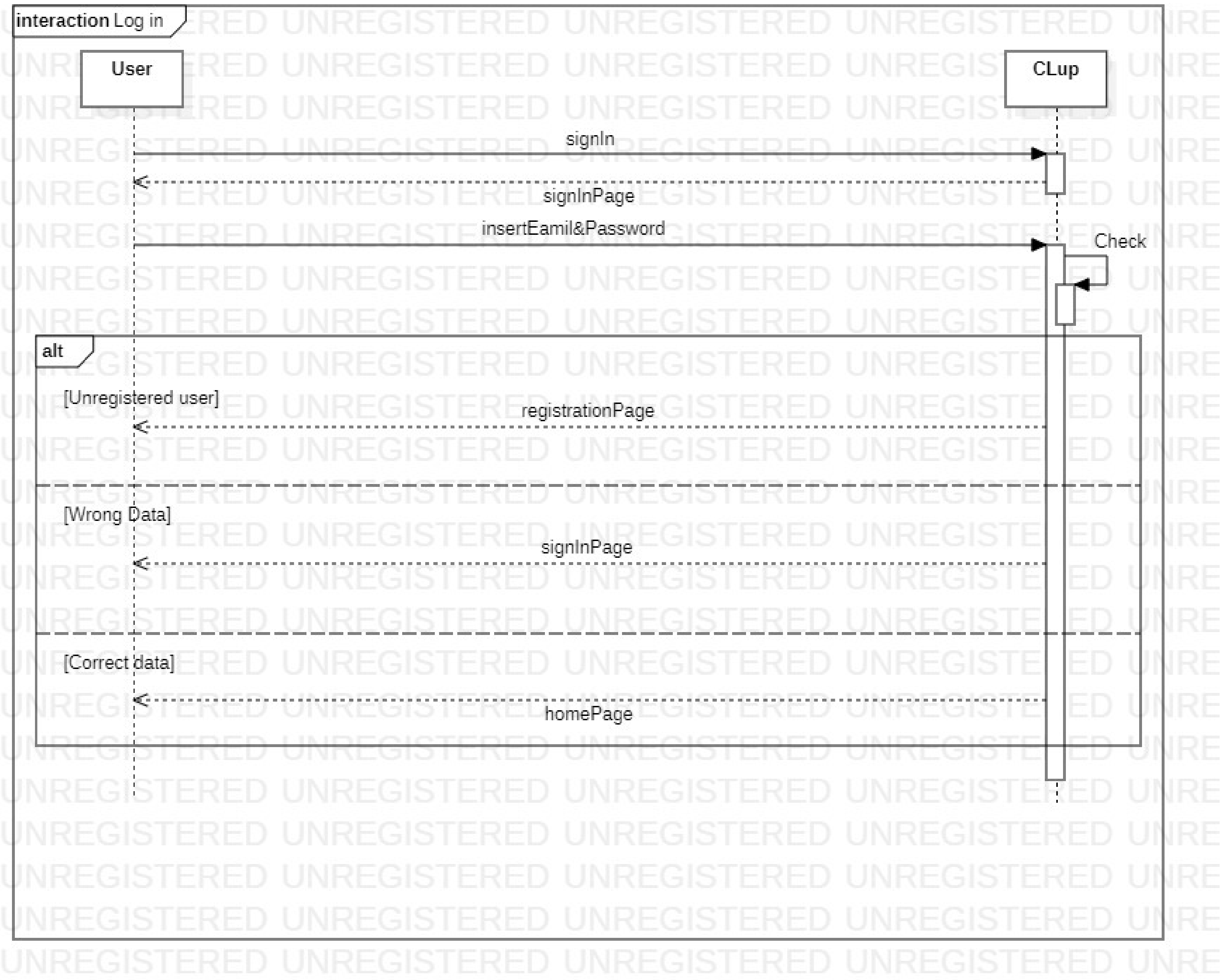


3.2.3.3 Sequence Diagram

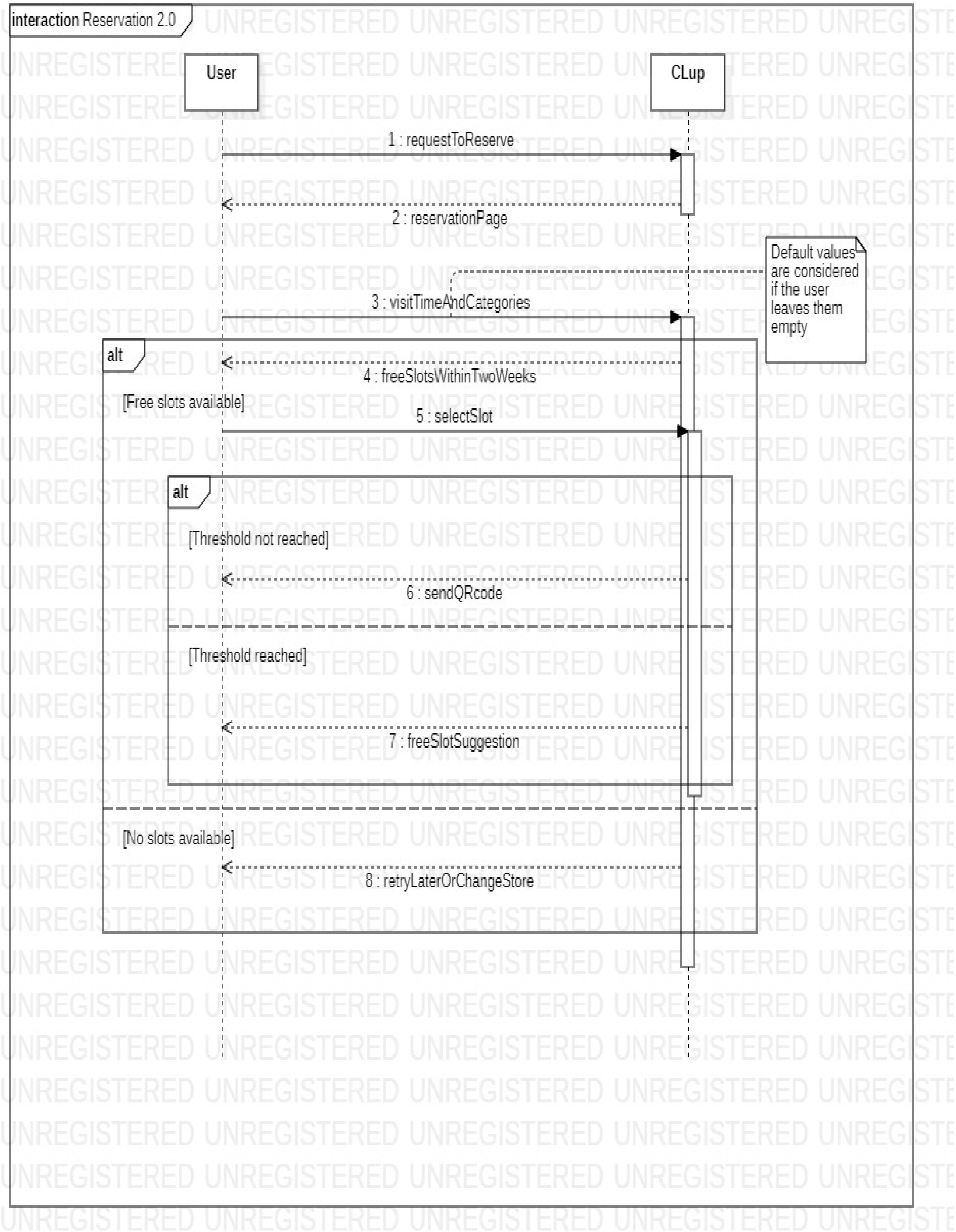
Registration



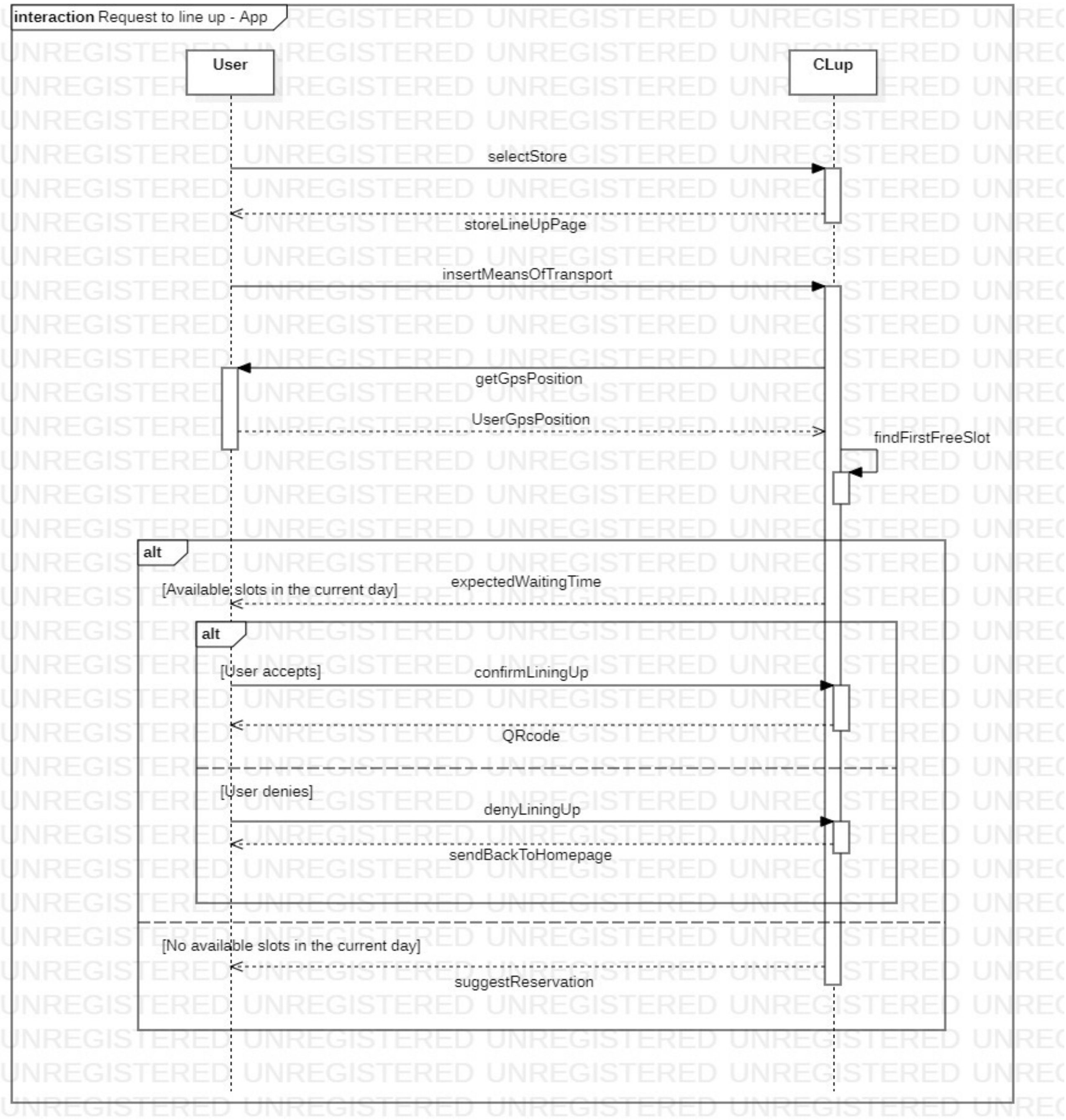
Login


interaction Log in
User
CLup
signIn
signInPage
insertEamil&Password
Check
alt
[Unregistered user]
registrationPage
[Wrong Data]
signInPage
[Correct data]
homePage

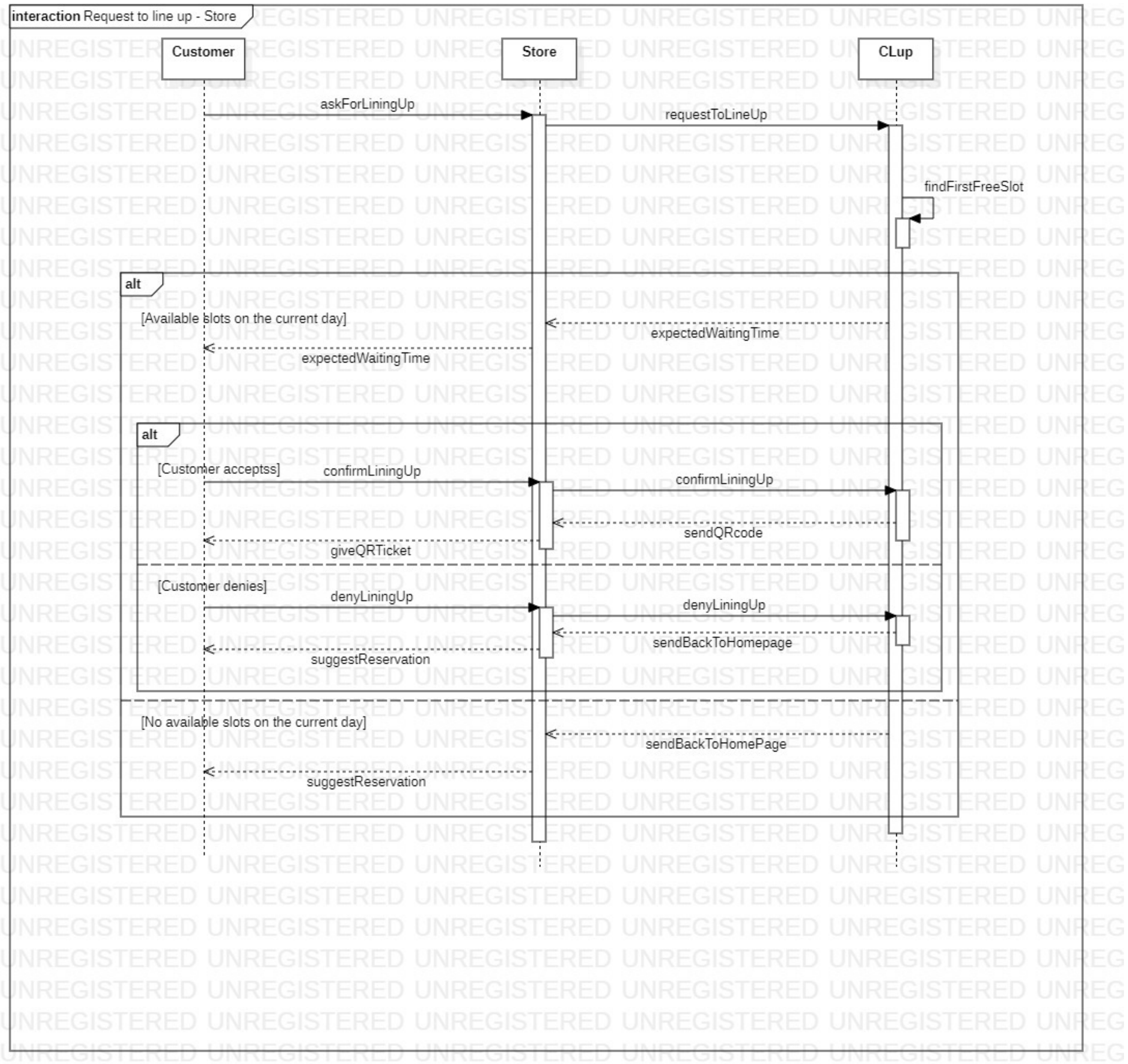
Reservation



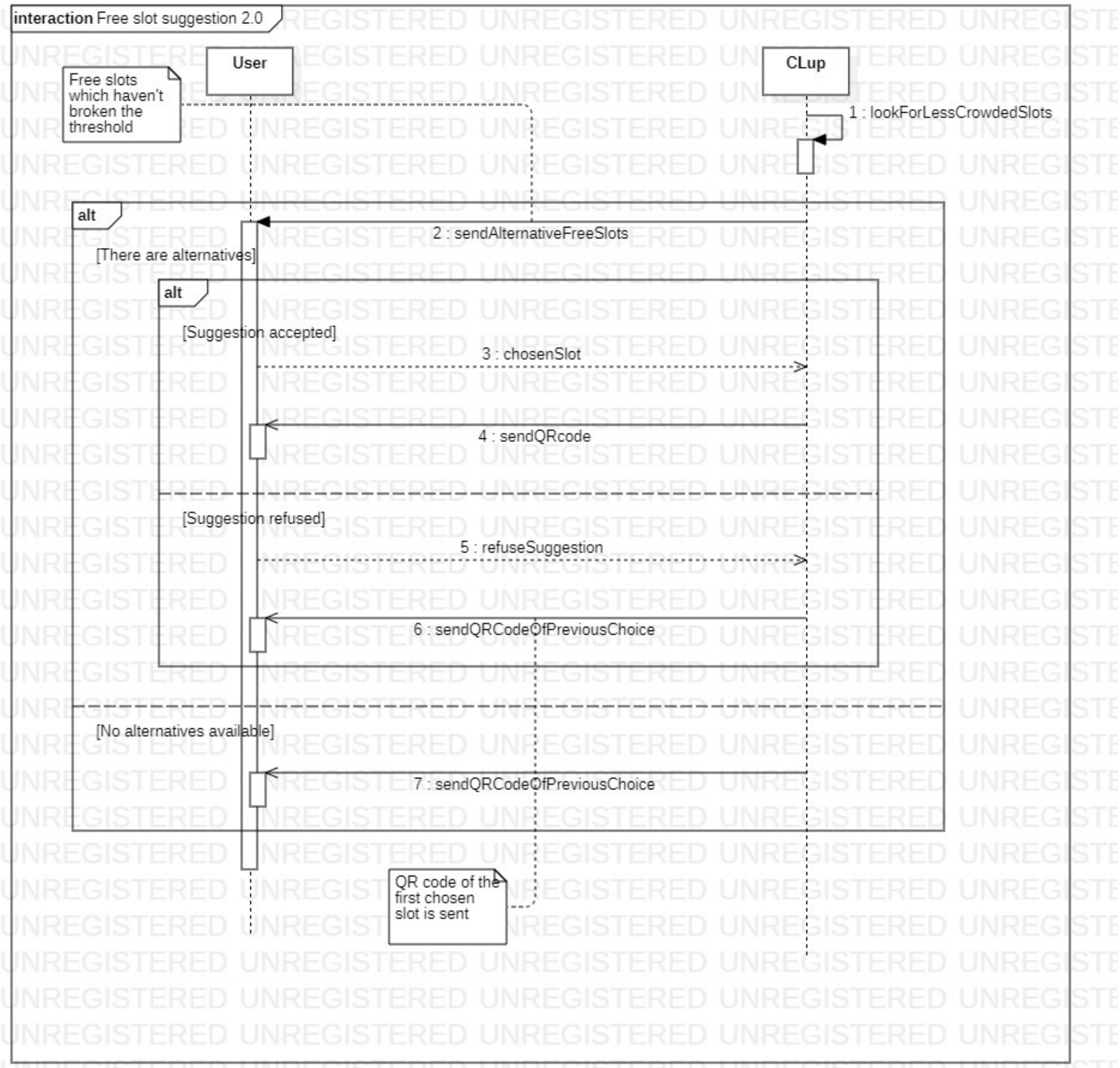
Request to line up using the App



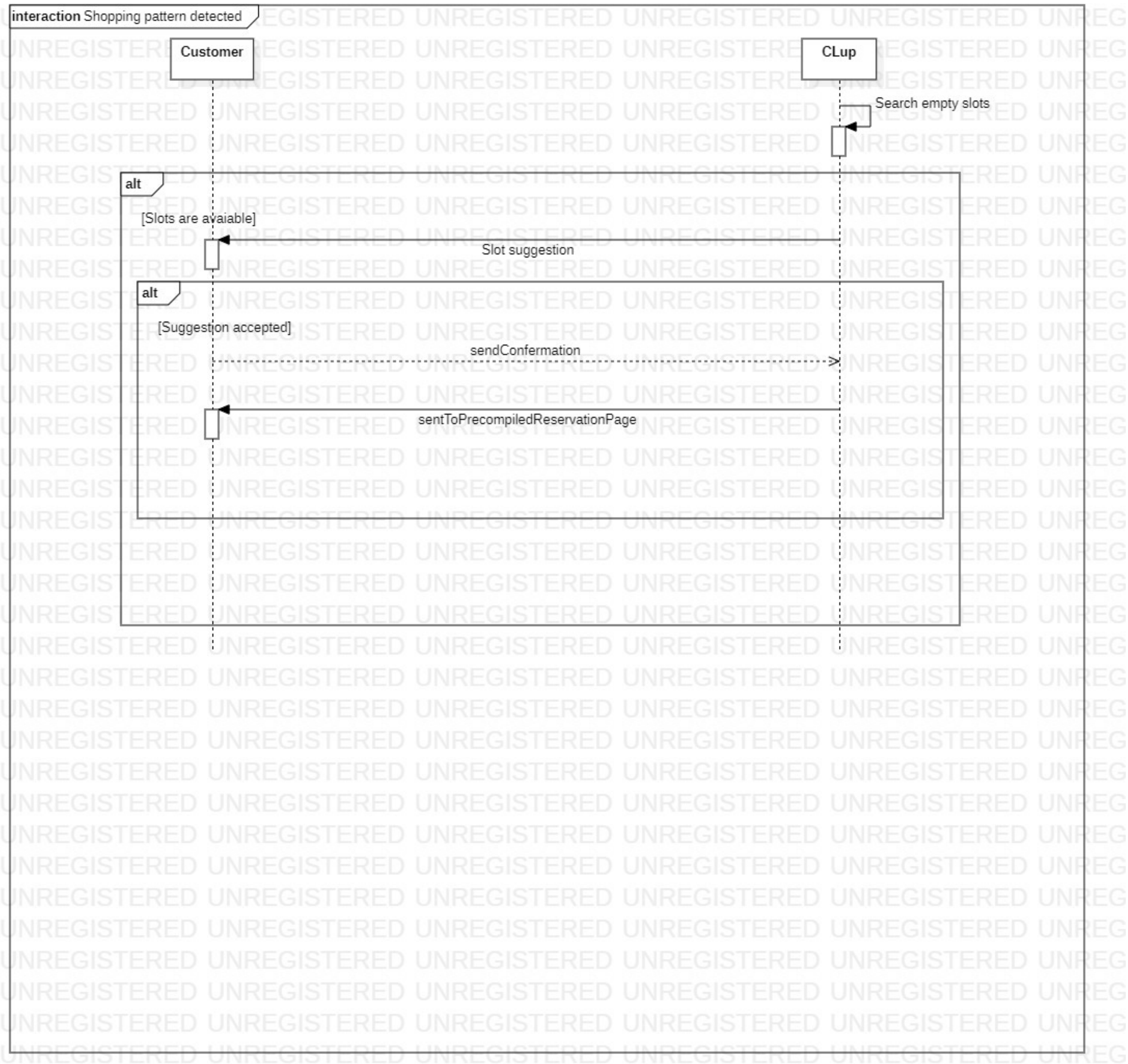
Request to line up at the store



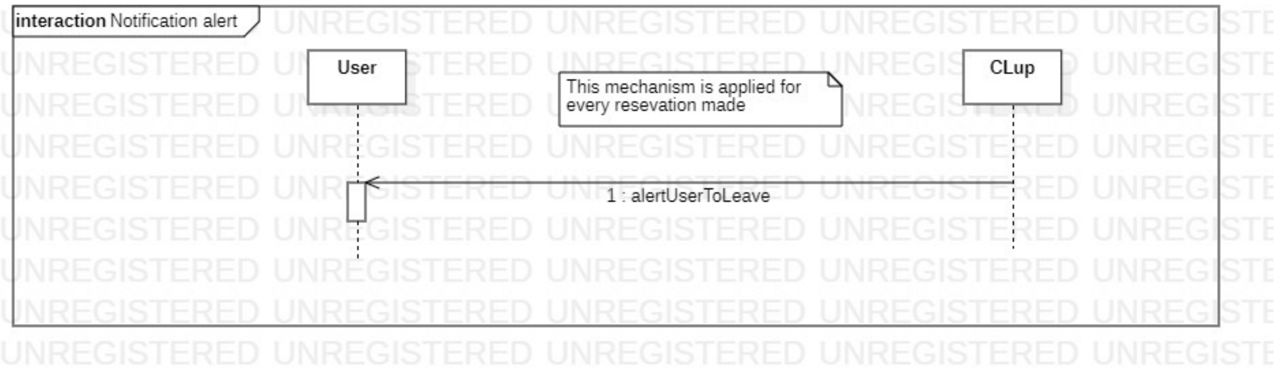
Free slot suggestion


interaction Free slot suggestion 2.0
User
CLup
Free slots which haven't broken the threshold
1 : lookForLessCrowdedSlots
alt
2 : sendAlternativeFreeSlots
[There are alternatives]
alt
[Suggestion accepted]
3 : chosenSlot
4 : sendQRcode
[Suggestion refused]
5 : refuseSuggestion
6 : sendQRCodeOfPreviousChoice
[No alternatives available]
7 : sendQRCodeOfPreviousChoice
QR code of the first chosen slot is sent

Shopping pattern detected



Notification Alert


interaction Notification alert
User
CLup
This mechanism is applied for every resevation made
1 : alertUserToLeave

3.2.4 Scenarios

Scenario 1: Reservation

Mario is a member of a four people family, he usually goes to the supermarket every Wednesday evening at 5 p.m. and even if there is a pandemic he has decided to not change his habit, in order to do so he decides to take a ticket early.

Sunday morning he takes his phone, goes on the supermarket's new app and he logs in. He perfectly knows what he's going to buy, in fact he regularly does a shopping list, so he selects the categories of goods he wants to buy like past, drinks, cleaning products, etc...
Mario usually takes an hour to do his purchases so, when he is asked, he inserts one hour as foreseen time. The system retrieves all the free slots within 2 weeks and luckily, due to the great advance of its reservation, there’s a free slot on Wednesday at 5 p.m. Mario then decides to go as usual on Wednesday, so the system generates the QR and Mario saves it on his phone.

Scenario 2: Line up

Maria is preparing a cake for his daughter’s birthday. She realizes that in the house there is no butter and she has to go to buy it.
She enters the applications and tries to reserve a place in the next hour, specifying that she needs only to buy a good in the “fresh food” category and she will only need five minutes.
The system detects a spot in the next half hour and asks Maria if she wants to reserve the place.
Maria accepts and saves the QR code.

Scenario 3: Denied reservation and suggestion.

Luigi due the pandemic works from home and today has decided to clean his home office in order to work more comfortably but he realized he lacks some cleaning products so he wants to reserve a place in the supermarket and go to buy them.
He doesn’t know how much time he will spend in the shop in order to find what he needs but he specifies that he will buy only “cleaning and house products”.
He tries to reserve a place in the next hour but the system suggests Luigi an hour long slot the next morning because in the afternoon the supermarket will be crowded.
Luigi accepts the proposal and saves the generated QR code.

Scenario 4: Reservation Suggestion

Angela since the pandemic has decided that every Monday evening she goes to do the weekly purchase.
After months of pandemic the system detects the shopping pattern of Angela.
Several days before, the system asks Angela if she wants to reserve a slot the next Monday evening. She accepts and saves her QR code.

Scenario 5: Request to line up by the store

Dino is an old man who needs to go to the supermarket in order to buy primary goods for him and his wife, but he’s not so practiced with his smartphone.
Dino approaches the store and goes to the utility desk, asking for lining up at the store.
The shop assistant, Sonia, helps him, requesting the system to find the first free slot in the waiting queue.
The system finds a free slot 30 minutes later, so Sonia asks Dino if the waiting time is acceptable. Dino accepts and Sonia sends the confirmation to the system.
She prints a ticket with the generated QR code and gives it to Dino.

Scenario 6: Reservation by the store

Clara is an old woman who doesn’t have a smartphone. Before the pandemic she used to go every Monday to the store to buy the needs for the week. Also during the pandemic she wants to preserve her habits.
Clare approaches the utility desk where the shop assistant, Paolo, works, in order to reserve a visit at the store. Paolo retrieves all the free slots within two weeks and shows them to Clara, who decides for next Monday at 9.20. Clara is not sure of what she will buy and even how much she will stay in the store: Pietro leaves blank spaces in the form.
Pietro then sends a confirmation and prints a ticket with a unique QR code generated by the system and the date of visit, giving it to Clara.

Scenario 7: Notification alert

Giovanni is a young student who lives alone. During the pandemic, he has signed up for the new app to avoid gatherings at the supermarket. To try it up, he reserves a visit at the store on Saturday.
On Saturday he starts playing some video games to kill the day. Becoming addicted minute by minute, he’s forgotten about the reservation. Luckily, 15 minutes before the expected departure time, a notification alerts Giovanni. He then stops the game and hurries up, preparing himself to go out.

Scenario 8: Reservation with exception

Carla, a 23-year-old girl, has just moved to another city and decides to book a reservation at the supermarket closest to her new home. She takes her phone, registers on the app, willing to go the next day. Carla puts the items she wants to buy but not the time that she will spend at the supermarket, because she doesn't know the location of the products and doesn't know how much time she needs. The system therefore sets her staying time with a default value. The system retrieves all the free slots within two weeks and luckily the next day is in them. Carla then accepts a slot at 2 p.m., so the system generates and sends her a unique QR code.

Scenario 9: Line up with exception

Luca and other friends organize a dinner to watch the champion league final together. Luca has the responsibility to buy the drinks so he requests an appointment as soon as possible in his trusted supermarket via app. The system finds a free slot and communicates it to Luca waiting for confirmation. In the meantime Luca receives a phone call from his friends, who tell him that they have already bought the drinks and he forgets his request and does not send the confirmation by phone. The system does not receive confirmation and cancels the appointment at the supermarket.

Scenario 10: Request to line up by the store with exception

Grandma Peppina, an old lady who usually goes shopping in the morning, usually goes to her trusted supermarket to reserve a shopping slot as soon as possible. One day something unexpected happens to her and Grandma Peppina arrives at the supermarket around lunchtime. She asks the supermarket employee who works at the desk to get in line to do the shopping. The employee sends a request to view the free slots for the current day to the system. The system shows that there are no free slots on the current day, the employee advises Grandma Peppina to go to the nearest supermarket or make an appointment for the next few days.

3.3 Design Constraints - Non-Functional Requirements

3.3.1 Standards compliance

The project in all his parts either the code or the physical disposition of the QR readers or the presence of the employee outside the supermarket should follow the dispositions given by this document.
The code should follow all the rules of clearness and efficiency given by the common sense.

3.3.2 Hardware limitations

The obvious limitation of our application is that some users may don’t have a smartphone or have the possibility to use the application.
This limitation is overcome by the employee who acts as a proxy between the user and the system but this is only possible when the supermarket is open (for example 8:00-20:00) but all the other time while the application is accessible the “fallback” option is not.+
Another limitation is that the system can’t detect a double reservation made by a user at the front desk, in this situation we’ll follow the assumption made in the domain assumptions.

3.4 Software System Attributes- Non-Functional Requirements

3.4.1 Reliability and availability

The system ensures a safe visit to a store, guaranteeing social distance.
It is needed therefore for the system to be extremely reliable and available.

Due to the danger of crowded places, shutdowns and faults of the system should happen very seldom: an overall reliability of 99,9% is recommended, with a downtime (and a MTTR) of 8.45 hours in a year.

A key role is played by the shop assistants at the store, acting as proxies for the clients: the system should be so extremely reliable in order to guarantee an effective social distancing, preventing lots of people from going to the store attempting uselessly to book a visit or line up.

To prevent the system from losing data about the waiting queue and the reserved slots, an infrastructure hosting a backup of this key information is needed, to guarantee the users to access the store without any problem. In any case, the system should alert the users of the actual faults in 10 minutes at most, to warn them of the impossibility of doing requests. The system should do the same thing when it returns available.

3.4.2 Security

CLup is thought to be a specific app which supports many stores of the same chain: stores can request to be shown on the app, opening an account as the users do. Data of both are kept secure and encrypted.

Whenever a user books a visit or request to line up, the system sends to the store only the needed information (categories of goods, estimated visit duration and the QR code associated with the visit), without any specifications about the user's generalities: privacy of the user is therefore guaranteed.

Customers who don’t have the app installed can use the fallback options by the store without giving particular information about themselves.

In case of forgotten password, an email is sent to the user, enabling him or her to modify it in a new one.

3.4.3 Maintainability

The system must have an high degree of maintainability to grant that we need to develop the application following the standard and the common practice, every relevant part of the code will be commented and explained and any future change in model or in the application's functions will be documented.

The code must be tested in every relevant part.

3.4.4 Portability

The system must be supported by the main OS working on smartphones, as Android, iOS. In order to reduce the costs of buying new devices for the store, the app should run also on computers: it must support Linux, MacOS and Windows OS.

3.4.5 Scalability

Due the obvious need of the people to go to the supermarket this application will be widely used among the population, this means that the application has to be scalable. In particular we will not have to change parts of the code due the number of users.

3.5 Additional Specifications: Types of supermarket units

The various types of units that the supermarket may have are:

- Fruit.
- Vegetables.
- Meat.
- Fish.
- Frozen food.
- Cold cuts.
- Dairy goods.
- Canned goods.
- Drinks.
- House cleaning .
- Personal hygiene.

4 Formal Analysis Using Alloy

4.1 Alloy Code

In order to create a model we have made some hypothesis:

- For each supermarket we have at least one unit.
- At most 3 people can stay at the same time in each unit.
- The threshold is set to 3 people in each unit.
- A person stays in the supermarket one hour

These hypothesis are true only in this model, in the reality they have different values or they are not true at all.

```
abstract sig Date {

        prec: lone Date

}

one sig Monday extends Date {}{ prec = none}

one sig Tuesday extends Date {}{ prec = Monday }

one sig Wednesday extends Date {}{ prec = Tuesday }

one sig Thursday extends Date {}{ prec = Wednesday }

one sig Friday extends Date {}{ prec = Thursday }

abstract sig Hour {

        precHour: lone Hour

}

one sig Hour_1 extends Hour {}{precHour = none}

one sig Hour_2 extends Hour {}{precHour = Hour_1 }

one sig Hour_3 extends Hour {}{precHour = Hour_2 }

one sig Hour_4 extends Hour {}{precHour = Hour_3 }

one sig Hour_5 extends Hour {}{precHour = Hour_4 }

one sig Hour_6 extends Hour {}{precHour = Hour_5 }

one sig Hour_7 extends Hour {}{precHour = Hour_6 }
```

```
sig Store {
        qrReader: some QRReader,
        units: some Unit
}{
        #qrReader > 1
}
sig User {
        reservationSet: set Reservation
}
sig Employee {
        store: one Store
}
sig Reservation {
        acceptedSuggestion: one Suggestion,
        notification: one NotificationAlert,
        chosenStore: one Store,
        user: lone User,
        employee: lone Employee,
        chosenTimeSlot: one TimeSlot,
}{
        notification.notificationHour = chosenTimeSlot.startingHour
        notification.notificationDate = chosenTimeSlot.date
        #user = 1 <=> #employee = 0
}
sig LineUpRequest extends Reservation {}
{let x = acceptedSuggestion.timeSlot | #x = 1}
```

```
sig Report {
        reservation: one Reservation,
        qrReader1: one QRReader,
        qrReader2: one QRReader,
        entranceHour: one Hour
}{
        qrReader1 != qrReader2
        entranceHour = reservation.chosenTimeSlot.startingHour      //User are expected to be on time
}
sig Suggestion {
        timeSlot: some TimeSlot
}
sig QRReader {
}
sig Unit {
        maxPlaces: Int,
        availableSpots: some AvailableSpot
}{
        maxPlaces  = 3
}
sig AvailableSpot {
        date: Date,
        startingHour: Hour,
        relatedUnit: one Unit
}
```

```
sig TimeSlot {
        relatedAvailableSpots: some AvailableSpot,
        startingHour: Hour,
        date: Date
}{
        all avS: AvailableSpot | (avS in relatedAvailableSpots => (avS.date = date) && (avS.startingHour =
startingHour))


}
sig NotificationAlert {
        notificationDate: Date,
        notificationHour: Hour
}
fact noReportWithReservationMadeByEmployee {
        all r: Reservation | no rep: Report | #r.employee = 1 and ( r in rep.reservation)
}
fact employeeDoesntMakeRequestsToOtherStore {
        all r: Reservation | #r.employee = 1 implies r.chosenStore = r.employee.store
}
fact noMultipleReservationAtTheSameTimeForSingleUser {
        all u: User | no disj r1, r2: Reservation |  (r1 in u.reservationSet and r2 in u.reservationSet
                                                                        and r1.chosenTimeSlot.date =
r2.chosenTimeSlot.date
                                                                        and
r1.chosenTimeSlot.startingHour = r2.chosenTimeSlot.startingHour)
}
fact noSharedQrReader {
        all disj s1,s2: Store | no q: QRReader  | (q in s1.qrReader && q in  s2.qrReader)
 }
fact noSharedUnit {
        all u: Unit | no disj s1,s2: Store  | (u in s1.units && u in s2.units)
 }
```

```
fact  noSharedAvailableSpotsAmongUnits {
        all avS: AvailableSpot | no disj u1, u2: Unit | avS in u1.availableSpots and avS in u2.availableSpots
}


//We can't have more than one spot in an hour for a specific unit
fact oneAvailableSpotPerHour {
        all u: Unit | no disj avs1, avs2: AvailableSpot | (avs1 in u.availableSpots) and (avs2 in u.availableSpots) and (avs1.date = avs2.date) and (avs1.startingHour = avs1.startingHour)
}
//notification is injective
fact oneReservationPerNotification {
        no disj r1, r2: Reservation | r1.notification = r2.notification
}
//reservation is injective
fact oneReportPerReservation {
        no disj rep1, rep2: Report | rep1.reservation = rep2.reservation
}
fact oneReservationPerSuggestion {
        no disj r1, r2: Reservation | r1.acceptedSuggestion = r2.acceptedSuggestion
}
fact reflexivityOfAvailableSpot {
        availableSpots = ~relatedUnit
}
fact reflexivityOfReservationAndUser {
        user = ~reservationSet
}
//se due timeslot hanno la stessa ora e la stessa data allora appartengono a due stores diversi
```

```
fact noMultipleTimeSlotsForSameStore{

        all r: Reservation | no disj ts1, ts2: TimeSlot |     ts1 in r.acceptedSuggestion.timeSlot and         ts2
in r.acceptedSuggestion.timeSlot and   ts1.relatedAvailableSpots.relatedUnit in r.chosenStore.units and
ts2.relatedAvailableSpots.relatedUnit in r.chosenStore.units and ts1.date = ts2.date and ts1.startingHour
= ts2.startingHour

}

fact noMultipleEmployeeForSameStore {

        no disj e1, e2: Employee | e1.store = e2.store

}

fact noStoreWithoutReservation {

        no s: Store | all r: Reservation | s not in r.chosenStore

}

fact noNotificationWithoutReservation {

        no n: NotificationAlert | all r: Reservation | n not in r.notification

}

fact noUnitWithoutRelatedStore {

        all u: Unit | some s: Store | u  in s.units

}

fact noTimeSlotWithoutSuggestion {

        all ts: TimeSlot | some s: Suggestion | ts in s.timeSlot

}

fact noSuggestionWithoutReservation {

        all s: Suggestion | some r: Reservation | s in r.acceptedSuggestion

}

fact noAvailableSpotWithoutTimeSlot {

        all avs: AvailableSpot | some ts: TimeSlot | avs in ts.relatedAvailableSpots

}

fact ReservationInSingleStore {

        all r: Reservation| r.acceptedSuggestion.timeSlot.relatedAvailableSpots.relatedUnit in
r.chosenStore.units

}
```

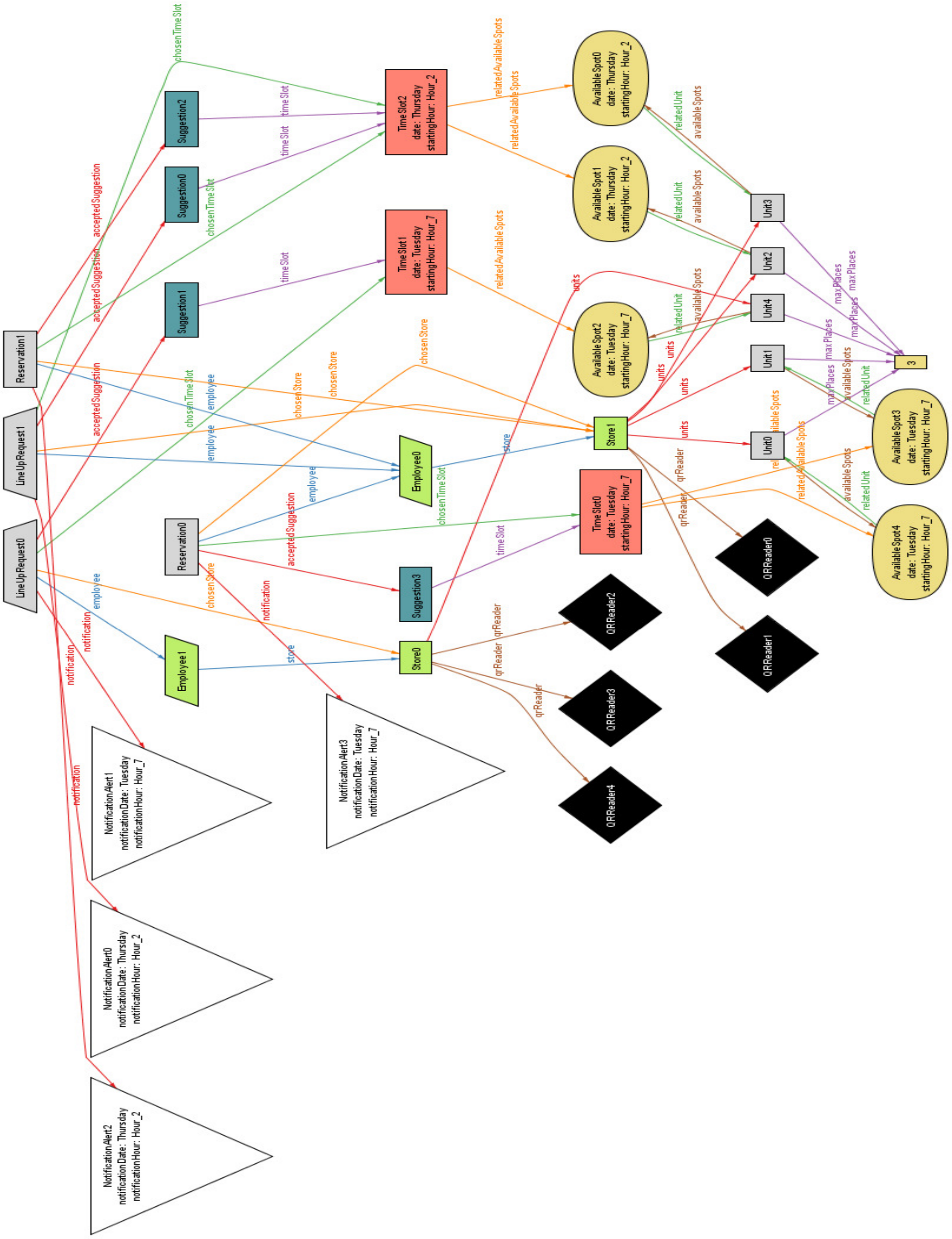
```
fact timeSlotsRefersToSameUnitsForEachReservation {

        all ts1, ts2: TimeSlot | all r: Reservation |
(ts1 in r.acceptedSuggestion.timeSlot) && (ts2 in r.acceptedSuggestion.timeSlot) implies
ts1.relatedAvailableSpots.relatedUnit =ts2.relatedAvailableSpots.relatedUnit

}

fact coherenceBetweenAcceptedSuggestionAndChosenTimeSlot {

        all r: Reservation | r.chosenTimeSlot in r.acceptedSuggestion.timeSlot

}

fact noRedundantTimeSlots {

        no disj ts1, ts2: TimeSlot | ts1.startingHour = ts2.startingHour and ts1.date = ts2.date and
ts1.relatedAvailableSpots = ts2.relatedAvailableSpots

}

fact noReportOnSubsequentDays {

        let prec_tran = ^prec | all r1: Reservation | (one rep: Report, d: Date | rep.reservation = r1 and d =
r1.chosenTimeSlot.date) implies (all r2: Reservation | (r2.chosenTimeSlot.date in
r1.chosenTimeSlot.date.prec_tran) implies (one rep2: Report | rep2.reservation = r2))}

fact noReportOnSubsequentHours {

        let precHour_tranRef = *precHour | all r1: Reservation | (one rep: Report | rep.reservation = r1)
implies (all r2: Reservation | (r2.chosenTimeSlot.date = r1.chosenTimeSlot.date and
        (r2.chosenTimeSlot.startingHour in r1.chosenTimeSlot.startingHour.precHour_tranRef)
        implies (one rep1: Report | rep1.reservation = r2)))}

fact maxAmountOfPeopleInTheStore {

        //no avs: AvailableSpot |  #{r: Reservation | avs in r.chosenTimeSlot.relatedAvailableSpots} > 3

        no avs: AvailableSpot |  #{r: Reservation | avs in
r.acceptedSuggestion.timeSlot.relatedAvailableSpots} > 3

}

//A single user cannot reserve at the same time more than once

assert multipleReservationNotPossible {

        no u: User |some disj r1, r2: Reservation | r1 in u.reservationSet and r2 in u.reservationSet
        and r1.chosenTimeSlot.startingHour = r2.chosenTimeSlot.startingHour and
r1.chosenTimeSlot.date = r2.chosenTimeSlot.date

}

//check multipleReservationNotPossible for 4
```

```
assert maxPlacesCannotBeExceeded{

	no disj r1, r2, r3, r4: Reservation | some avs: AvailableSpot |	avs in
r1.chosenTimeSlot.relatedAvailableSpots and	avs in r2.chosenTimeSlot.relatedAvailableSpots and	avs
in r3.chosenTimeSlot.relatedAvailableSpots and avs in r4.chosenTimeSlot.relatedAvailableSpots }

//check maxPlacesCannotBeExceeded for 5

/* PREDICATES */

/**

	A user reserves or makes multiple requests to line up.

	It is possible to make requests in more than one store, in this case 2 store.

	A user cannot make multiple requests for the same hour and date, even if they are in different
stores

	On the other hand, multiple users can make requests in the same timeslot.

	In this case, it is shown the latter situation

*/

pred reservationOrLiningUpByTheApp {

	#Store = 2

	#User = 2

	#Employee = 0

	some disj r1, r2: Reservation | r1.user != r2.user and r1.chosenTimeSlot = r2.chosenTimeSlot

}

//run reservationOrLiningUpByTheApp for 5

/**

	An employee in the store act as a proxy and reserves or makes requests in place of clients.

	It is possible for him/her to reserve in the same timeslot for multiple requests.

	In this example, an employee reserves for three times in the same timeslot.

	The multiplicity of Suggestions related to the same TimeSlot is to be noted too.

	There can be multiple employee related to a single store

*/
```

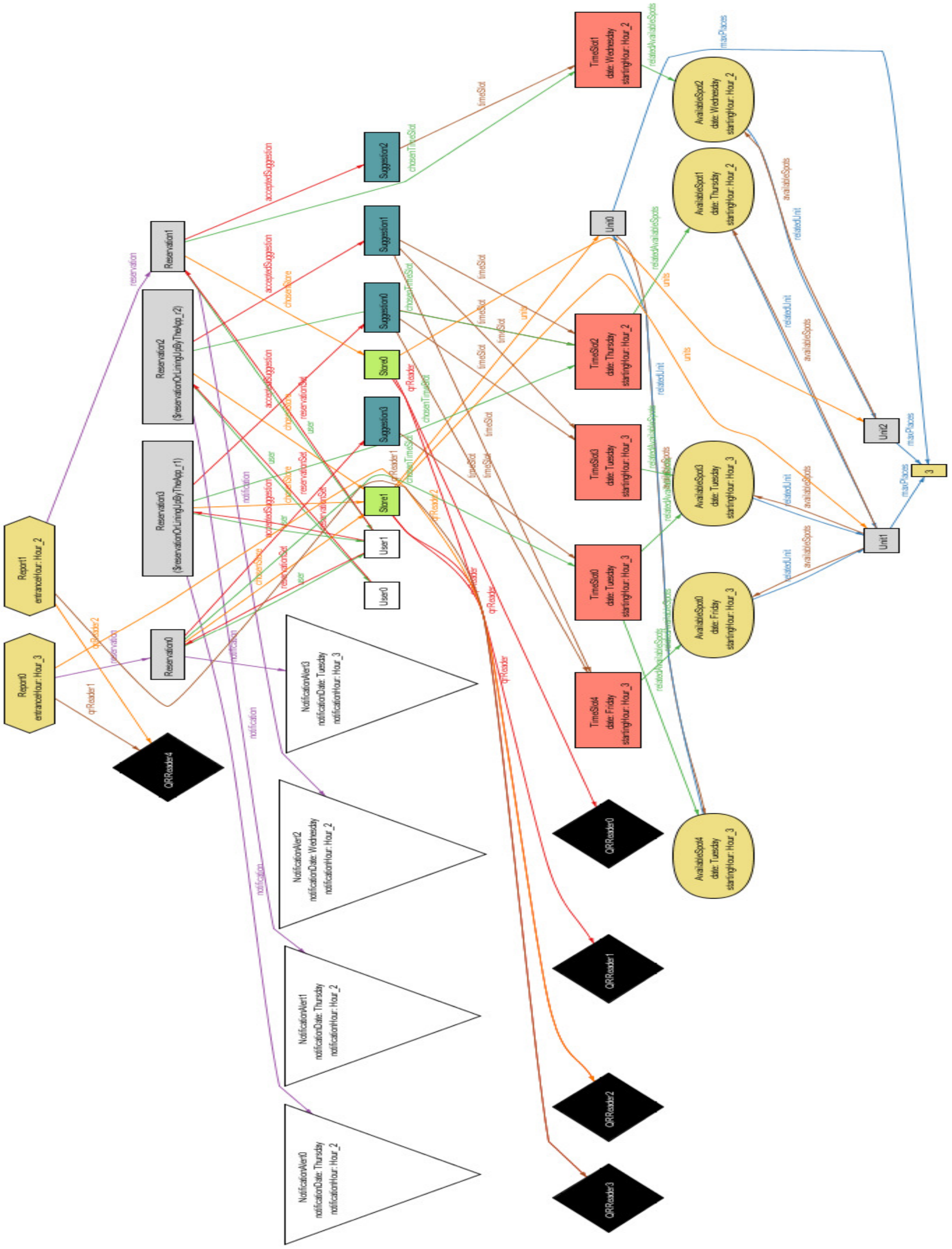
```
pred reservationOrLiningUpByTheStore {
        #Employee = 2
        #User = 0
        #Store = 1
        #TimeSlot = 1
        #Reservation = 3
}
//run reservationOrLiningUpByTheStore for 5
/**A more general situation.*/
pred mixedRequests {
        #Store = 2
        #User = 2
        #Employee = 2
        #Reservation = 5
        some disj r1, r2: Report | r1.reservation.chosenTimeSlot.date !=
r2.reservation.chosenTimeSlot.date
}
run mixedRequests for 5
pred showMaxPlaces {
        #Store = 1
        #TimeSlot = 2
        #AvailableSpot = 3
        all disj a1, a2: AvailableSpot | a1.date = a2.date and a1.startingHour = a2.startingHour
}
```

4.2 Metamodel

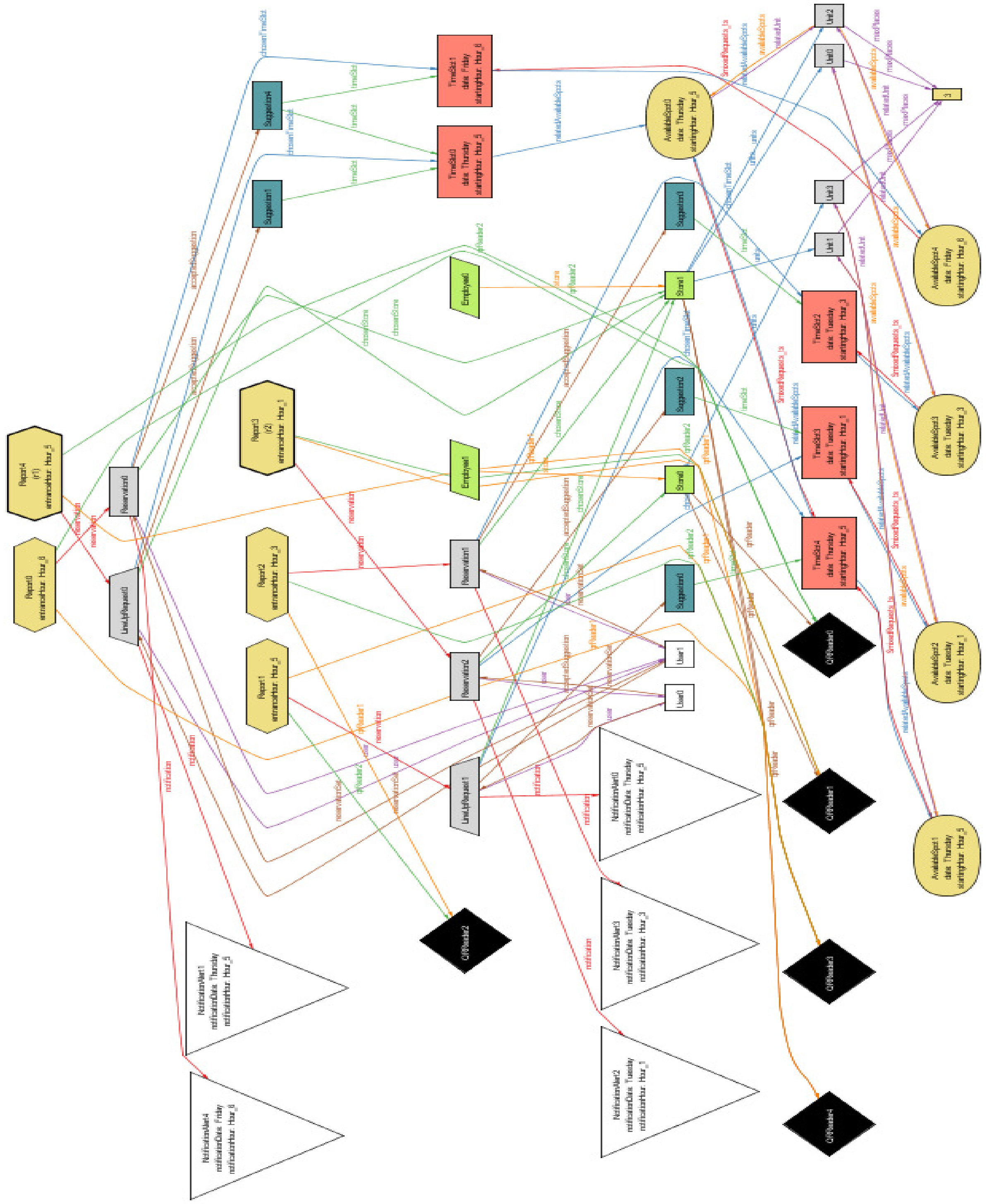
reservationOrLiningUpByTheStore



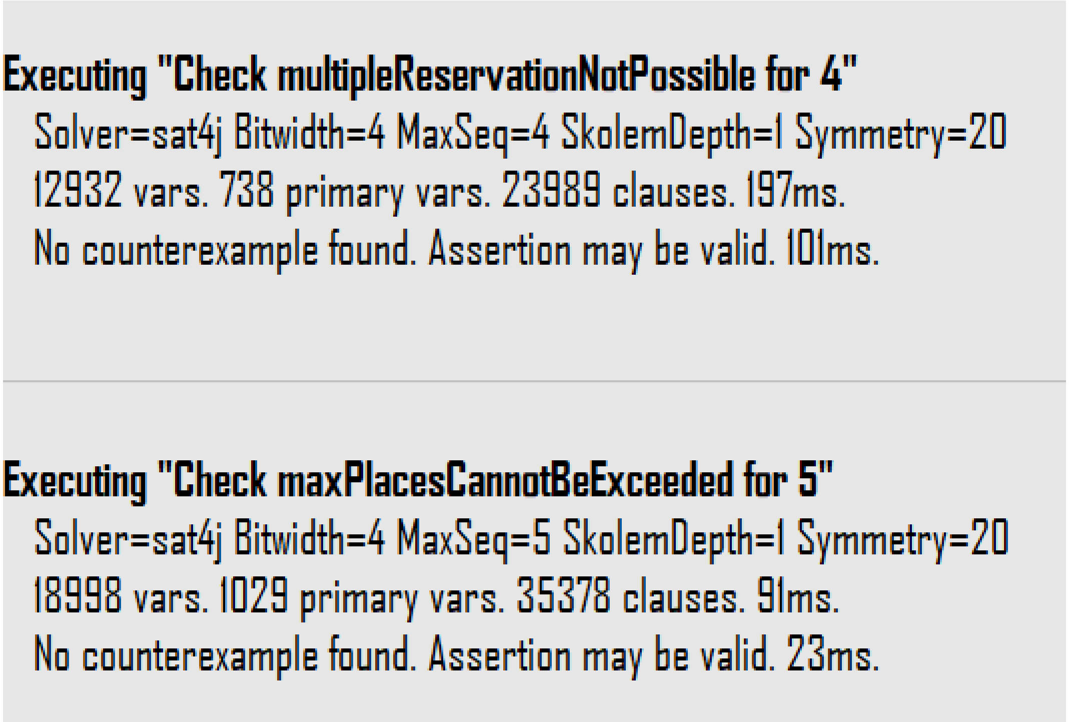
ReservationOrLiningUpByTheApp


TimeSlot1 date: Wednesday startingHour: Hour_2
AvailableSpot2 date: Wednesday startingHour: Hour_2
AvailableSpot1 date: Thursday startingHour: Hour_2
Suggestion2
Suggestion1
Suggestion0
Suggestion3
Store0
Store1
User1
User0
Reservation1
Reservation2 ($reservationOrLiningUpByTheApp_r2)
Reservation3 ($reservationOrLiningUpByTheApp_r1)
Reservation0
Report1 entranceHour: Hour_2
Report0 entranceHour: Hour_3
QRReader4
QRReader0
QRReader1
QRReader2
QRReader3
NotificationAlert3 notificationDate: Tuesday notificationHour: Hour_3
NotificationAlert2 notificationDate: Wednesday notificationHour: Hour_2
NotificationAlert1 notificationDate: Thursday notificationHour: Hour_2
NotificationAlert0 notificationDate: Thursday notificationHour: Hour_2
TimeSlot2 date: Thursday startingHour: Hour_2
TimeSlot3 date: Tuesday startingHour: Hour_3
TimeSlot0 date: Tuesday startingHour: Hour_3
TimeSlot4 date: Friday startingHour: Hour_3
AvailableSpot3 date: Tuesday startingHour: Hour_3
AvailableSpot0 date: Friday startingHour: Hour_3
AvailableSpot4 date: Tuesday startingHour: Hour_3
Unit0
Unit2
Unit1
3
timeSlot
chosenTimeSlot
acceptedSuggestion
chosenStore
reservationSet
user
reservation
notification
qrReader
qrReader1
qrReader2
relatedAvailableSpots
availableSpots
relatedUnit
units
maxPlaces

mixedRequest

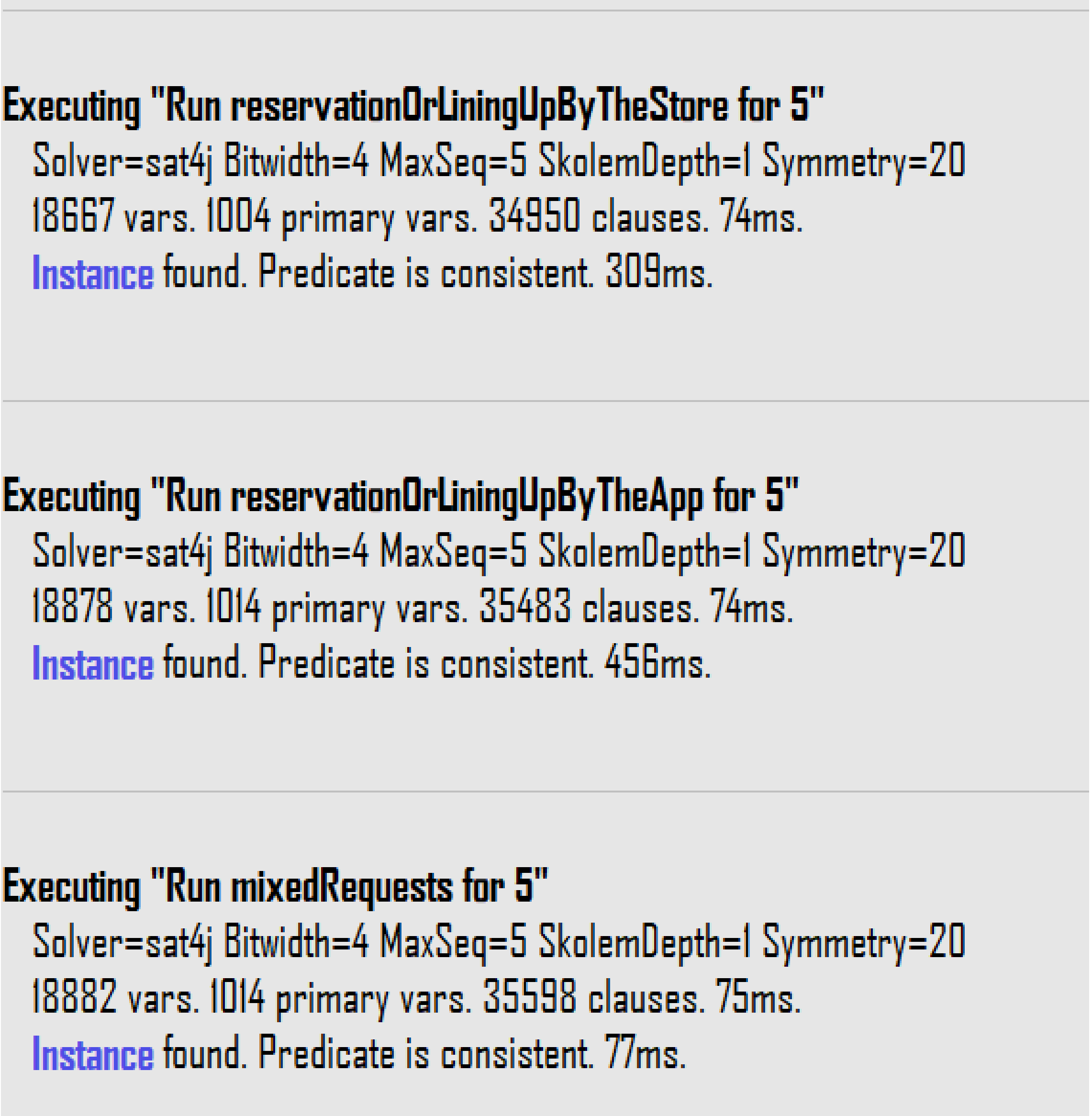


4.3 Results of Assertions



```
Executing "Check multipleReservationNotPossible for 4"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   12932 vars. 738 primary vars. 23989 clauses. 197ms.
   No counterexample found. Assertion may be valid. 101ms.

Executing "Check maxPlacesCannotBeExceeded for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
   18998 vars. 1029 primary vars. 35378 clauses. 91ms.
   No counterexample found. Assertion may be valid. 23ms.
```

4.4 Results of Predicates



```
Executing "Run reservationOrLiningUpByTheStore for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
   18667 vars. 1004 primary vars. 34950 clauses. 74ms.
   Instance found. Predicate is consistent. 309ms.

Executing "Run reservationOrLiningUpByTheApp for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
   18878 vars. 1014 primary vars. 35483 clauses. 74ms.
   Instance found. Predicate is consistent. 456ms.

Executing "Run mixedRequests for 5"
   Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
   18882 vars. 1014 primary vars. 35598 clauses. 75ms.
   Instance found. Predicate is consistent. 77ms.
```

5 Effort Spent

Work in group

22-11-2020	3 h 15 min
25-11-2020	1 h 33 min
27-11-2020	1 h 28 min
29-11-2020	2 h 20 min
2-12-2020	1 h 19 min
5-12-2020	3 h
13-12-2020	2h 6 min
14-12-2020	2 h 55 min
16-12-2020	1 h 59 min
19-12-2020	1h 42 min
23-12-2020	1h 30 min

Matteo Panzeri

Use Cases	1h 30 min
Scenarios	40 min
Purpose and scope	45 min
Mockups and Paragraphs 3.1.2/3.1.3/3.4.1/3.4.2	1h 30 min
Rewriting and editing	6 h

Riccardo Pellini

Use Cases	1h 30 min
Scenarios	35 min
Paragraphs 3.5	1h 5 min
Alloy	7h 30 min
Sequence diagram	1h

Paolini Giuseppe

Scenarios	1h
Paragraphs 2.2/2.3/3.5	1h
Sequence Diagrams	3h 30min

6 References

The diagrams have been made using StarUML.

The Mockups have been made using https://moqups.com/