Requirement Analysis and Specification Document RASD

December 2020

POLITECNICO
MILANO 1863

KONG XIANGYI
&
ZHANG YUEDONG

Contents

Chapter 1

Introduction

1.1 Purpose

This document is Requirement Analysis and Specification Document(RASD). The main purpose of this document is the following points

- Communicates an understanding of the requirements to the audience and explains both the application domain and the system to be developed.
- Contractual: Make this project formal and written so that it has legal effect.
- As the baseline for project planning and estimation. i.e. size, cost, schedule.
- As the baseline for software evaluation

  It can support system testing, verification and validation activities

  It should contain enough information to verify whether the delivered system meets requirements
- As the baseline for change control, such as requirements change, software evolves.

And this RASD has the following intended audiences

- Costumers & Users : Some user may interest in validating system goals and high-level description of functionalities.
- Systems and Requirements Analysts: The RASD may help them to write various specifications of other systems that inter-relate.

- Developers, Programmers: The RASD may help the to implement the requirements
- Testers: The RASD may help the to determine that the requirements have been met
- Project Managers: The RASD may help them to measure and control the analysis and development processes

1.2 Scope

1.2.1 Description of the given problem

At the end of 2019, a global epidemic broke out and swept almost all countries in the world in just a few months. Starting in 2020, people's life rhythm has been completely disrupted by this epidemic, a lot of cities are blocked, people are allowed to exit their homes only for essential needs, everyone had to wear masks and respect the social-distancing at least 1.5 m. In the public area, the human community has to take measures to avoid the crazy spread of the virus. Restaurants began to use dividers to separate the table, supermarkets and museums began to restrict flow of people, the school also adopted into two classes mode: online and onsite.

In this situation, a new problem arises, how to decrease the spread of the virus through technical means?

Since grocery shopping is the most needed activity under the lock-down, so let's narrow the problem to grocery shopping.

In the supermarket, In order to meet these strict rules, many challenges have arisen, so, we can turn to technology, in particular to software applications, to help navigate the challenges created by the imposed restrictions.

So, this project appeared - Customers Line-up(CLup).

1.2.2 World Phenomena

WP_1	Due to the COVID-19, all people in risk.
WP_2	Customer have to visit the Store
WP_3	Some Customers can use the smartphone, and others not.
WP_4	Different Customers need to buy different items.

1.2.3 Shared Phenomena

Controlled by the world and observed by the machine

SP_1	Collect Customers' data through sensors(GPS)
SP_2	Customer Login/ Register process
SP_3	Customer book a visit on CLup
SP_4	Customer retrieve the Ticket on the store
SP_5	Customer Scan-in and Scan-out on the store
SP_6	Manager can set some parameters in the Back-End System

Controlled by the machine and observed by the world

SP_7	Available booking date and time shown to Customers
SP_8	Generate the Ticket to Customers
SP_9	Customers receive the notification

1.3 Definitions, acronyms, abbreviations

1.3.1 Definitions

- Click Customer : The customer has the required technology to access the store. I.e a smartphone. They can use the customer terminal software.
- Brick Customer : The customer doesn't have the required technology to access the store, they have to hand out "tickets" on the spot.
- Store Manager : They have to manage the Store System, include the software and hardware.
- Ticket: The ticket is a document which contains three key information: QR Code, the estimated departure time, the queue number, and the Store Planned Roadmap. To the click customer, it's **E-ticket** but to the brick customer,it's **Paper Ticket**,and doesn't contain the estimated departure time, and just a General Store Map without the Planned Road.
- QR Code : When customer booked a visit, they will received a QR Code.
- QR Code Scanned Machine : A hardware, the Click Customer can use this machine scan their QR code.
- Tickets Hand-Out Machine : A hardware, the Brick Customer can use it retrieve their Ticket.

- Store Planned Roadmap: A store map that includes a finer way which is recommended form Store System.
- Digital Counterpart : A hardware, it with show the queue number.
- Store Back-End System : A software, as the back-end manages all stuffs.
- On-Time Store Data : A dataset that includes the store's on-time date.
  - The current queue
  - The customers in the store
  - The maximum number of people in the store.
- Long-Term Customers: The Click Customer who visited the store more than one time by the CLup mobile application.

1.3.2 Acronyms

- RASD – Requirement Analysis and Specification Document
- CLup - Customers Line-up
- UI - User Interface
- IOS - iPhone OS
- PC - Personal Computer
- IaaS - Infrastructure as a Service
- CRM - Customer Relationship Management
- KPI - Key Performance Indicator
- DAC - Discretionary Access Control

1.3.3 Abbreviations

- WP_n : n-th world phenomena
- SP_n : n-th shared phenomena
- R_n : n-th functional requirement

- NR_n: n-th non functional requirement
- G_n : n-th goal
- D_n : n-th domain assumption

1.4 Reference documents

- Specification Document: "R&DD Assignment A.Y. 2020-2021"
- Slides of the "Software Engineering 2" course A.Y. 2020-2021
- IEEE Recommended Practice for Software Requirements Specifications - IEEE Std 830-1998
- Fondamenti di Sistemi informativi per il Settore dell'Informazione - 7 settembre 2018
- Poste Italiane - www.poste.it

1.5 Overview

The RASD document consists of different chapters in the structure.

Introduction In the first chapter, it gives the overview of CLup. It provided the definitions, acronyms and abbreviations throughout the document. The description of the product and main goal of this project are provided in the test. Describe the world phenomena and shared phenomena.

Overall Description In this chapter, We describe how the product works. In addition, we provide more detailed information about the future details by using well-known diagrams (class diagrams and state diagrams). And product functions gives the more details of requirements. The list of domain assumptions and goals gives the support of the product. The chapter ends with constrains to customer, manager and store back-end system, which are the basic for proceeding the product.

Specific Requirements This chapter gives a more in-depth and detailed introduction to the system from the aspects of user/hardware/software interfaces, use cases, scenarios, sequence diagrams, and complete mapping of requirements. In use cases, it show the view from the click customer, brick

customer and store manager. By using sequence diagram, illustrate the operations between each other. We also provide three small scenarios to explain more details and significant meaningful of CLup, corresponding to the view of click customer, brick customer and store manager. Some specifications regarding performance requirements and design constraints can also be found in the following sections.

Formal Analysis Using Alloy In chapter 4, we use Alloy code as the fomal language to model some key concepts and introduce the purpose of it.

Chapter 2

Overall Description

2.1 Product perspective

The CLup – Customers Line-up system can divide into three parts. Three client ends and a server end, The client ends are divided into Customer end and Store Manager end according to the object-oriented. Moreover, we have divided Customers into two groups according to their characteristics. We referred to the Business direction's "click and brick" concept in the "Fondamenti di Sistemi informativi per il Settore dell'Informazione" teach book, so we decided to name it the "Click Customer" and the "Brick Customer", the Click Customer will use the mobile application, and the Brick Customer will use the Tickets Hand-Out Machine in the store.

The mobile application for the Click Customer can install in the Android and IOS operation system. To make this app simple enough for everyone, we only design two main functions: Sign-up/in and Booking Function. When they book a visit, they have to input four information, the visit date/time, the approximate expected duration of the visit, the categories of items that the Customer intends to buy, and the place they depart. The departing place can input manually by the Customer or recognize by the GPS module. If customers take privacy very seriously, the address can even be fuzzy, as long as it does not affect the calculation of the time to arrive at the store. After they have booked a visit and the Back-End system confirmed it, they will receive the E-Ticket as soon as possible with four data, QR Code, Queue Number, the estimated departure time and the Store Planned Roadmap.

The Tickets Hand-Out Machine is usually placed at the entrance of the store. We design only one button that retrieves a Ticket, no book a visit function. There are two reasons for this design. First of all, this Machina has to easy enough to use. On the other hand, we consider that the Customer

going out to pick up the number and wait until the book time to re-come to the store will significantly increase the number of outings, so we did not set up a booking process on this Machine. Otherwise, we mix the queue of two kind of customer through the Store Back-End System's queue schedule function to try to best to reduce the wait time.

The Manager end is a Management System that can install on a simple PC. The manager can use it to monitor the Store's On-Time Data. Furthermore, if there are too many people in some areas or the whole store, the store manager can solve it in two ways. First, he can lower the maximum people value in the store so that the queue will enter slower. On the other hand, he can adjust the queue and let the Customer that will visit the crowded area to enter the store later. After all, if the queue is too long due to these problems, he also can reschedule customers' book, but he can only reschedule the Customers' booking that before the departure time.

The most important part of this system is the server end. The server end deployed the Store Back-End System. This system has to communicate with all other ends and control the Hardware like the Digital Counterpart. It contains three functions, Booking Schedule Function, Queue Schedule Function, Customer Relationship Management System - CRM System. The Booking Schedule Function has to communicate with CRM, take the booking data, schedule the booking, and so on... For the Queue Schedule, We refer to the queuing model of the Poste Italiane. Its model has already been practiced very maturely. To visit the Poste Italiane, you can book your visit on the Ufficio Postale App or retrieve the ticket on the Machine, and the back-end system will mix two queues reasonably. For our Queue Schedule Function, to achieve goal G_2, we have higher requirements. We have to design a better mechanism so that both types of customers do not need to queue for too long, thereby reducing risk. Finally, the CRM System, this sub-system's primary work is communication with the Click Customer's end, like received the booking, integrate all information and generate/send the E-Ticket, by the way, when the Store Manager reschedules the booking, send notification and update the E-Ticket for the Customer. Otherwise, it also does other works like the CRM system usually do. For example, store Customer's data, analyze the Customer's history average duration for the Long-Term Customers.

2.1.1 Class Diagram and State Diagram

Class Diagram

The Class Diagram model as shown in Fig.2.1, These classes implement the three functions of the Store Back-End System,the Customer Relationship

Management sub-System composed by the Customer class, Booking, Ticket, and CRM class, Among them, the CRM class is responsible for communicating with other ends,and it can control customer class's status,the Customer data will store in Database. The BookSchedule class implement the Booking Schedule Function, all booking data will store in database, and this class can control them. The last class is the QueueSchedule class, it will implement the Queue Schedule Function, it will take booking information and get Brick Customer's ticket information to schedule the queue, the store manager can check the length of the queue and the maximum number of customer in-store, when he found that somewhere in the store was too crowded, he can lower the maximum value. The CallNextCustomer method will control the Digital Counterpart to make it display the next customer's queue number according to the maximum number allowed in the store. In summary, these functions form a set of effective mechanisms to achieve the goals.

State Diagram

The State diagram Fig.2.2 illustrated the processing of a Click Customer to require the E-ticket. Customers first register and log-in on the CLup, after they will book an appointment and specify their information(time, date, desired goods, etc.), then the can receive an E-Ticket. They can go to the Store according to the expected departure time given by the application, and scan-in the E-Ticket to enter the store, scan-out the E-Ticket when they leave the store.

The State diagram Fig.2.3 showed us how the manager monitors Store's On-Time data. The manager can view it on web in computer, and he can know the number of customer inside store, and the customer queuing outside the store. He has the right to set the max people number inside the store to balance the capacity of store. In advance, if too many people waiting outside the store, he can also send the notification to the customer who hasn't departed yet.

The State diagram Fig.2.4 represented the serve end state diagram for Click Customer. It receive the registration data and booking information from Click Customer. After it schedule the booking, then the system generate and send the E-Ticket in PDF. After, it will schedule the queue. When the customer enter the store, it will receive the scan-in, after the customer leave the store, it will receive the scan-out.

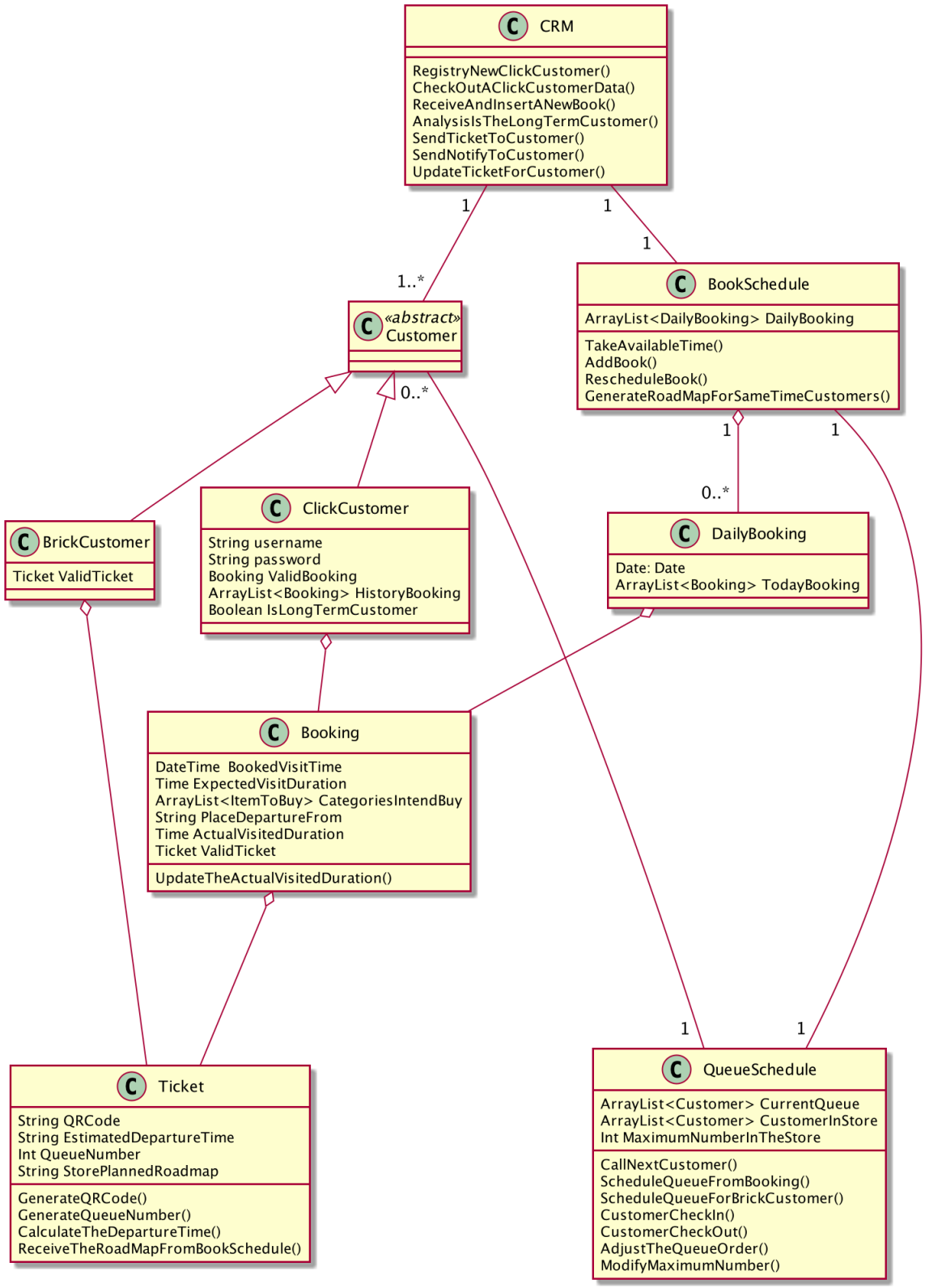

Figure 2.1: CLup Class Diagram

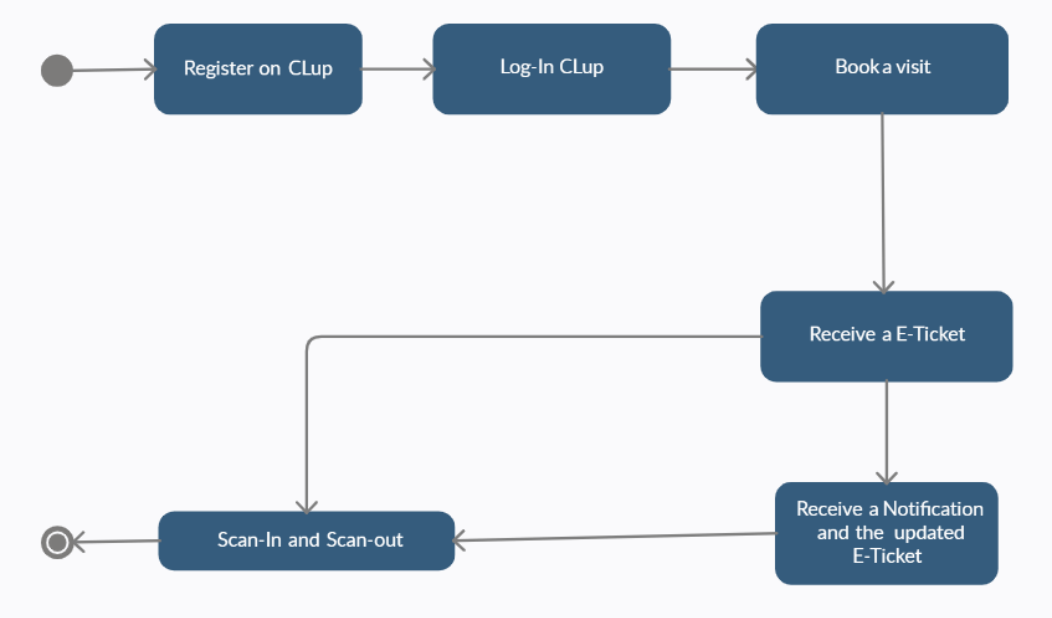

Figure 2.2: Customer State Diagram

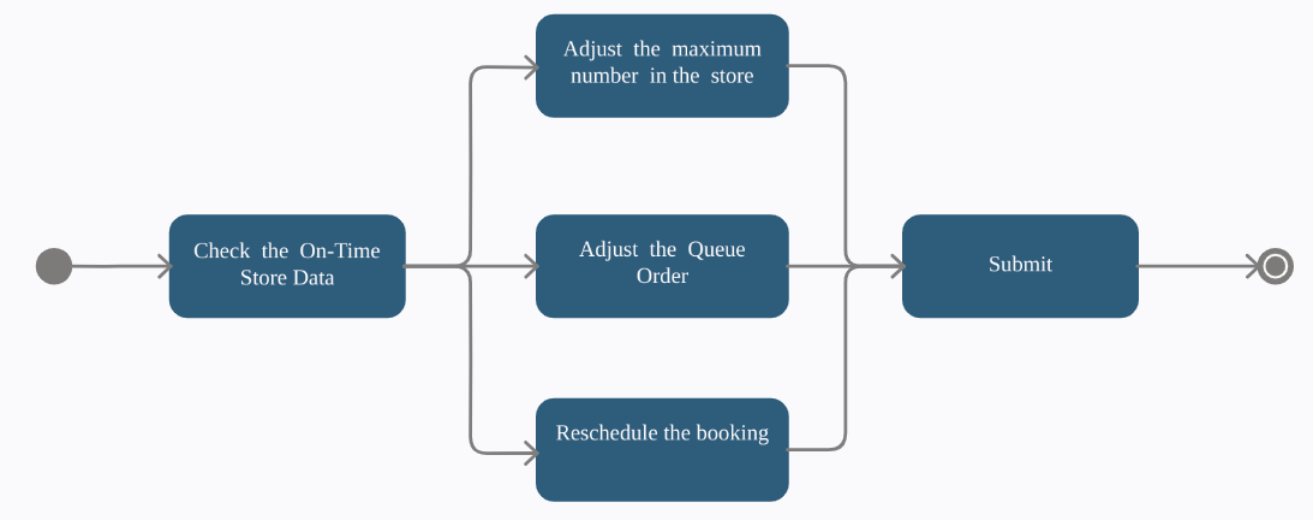

Figure 2.3: Store Manager State Diagram

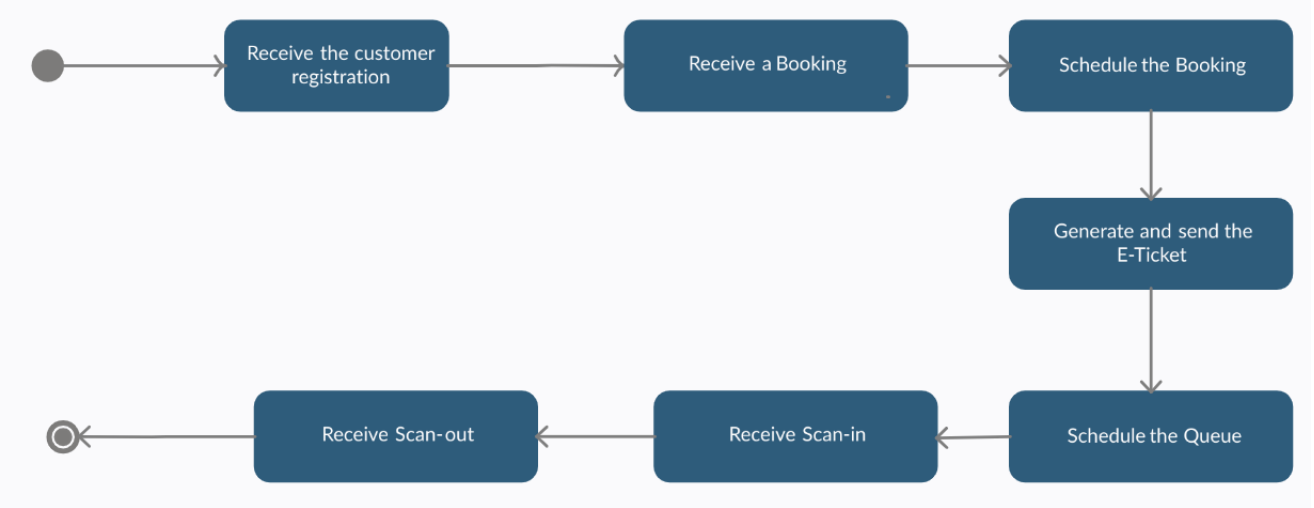

Figure 2.4: Server End to Click-Customer State Diagram

2.2 Product functions

2.2.1 Functional Requirements

- Each Click Customer shall be able to:
  - Sign-up
  - Log-in
  - Book a visit,to complete it, they have to indicate the following data
    - Indicate the desired date and time
    - Indicate the approximate expected duration of the visit
    - Indicate the categories of items that they intend to buy
    - Indicate or given by GPS the current place they want to depart to the shop
  - Check on the E-Ticket
  - Check on the notification from store manager when their book is rescheduled.
  - The customer can scan the QR Code at QR Code scanned machine when they enter **and** leave the store.
- Each Brick Customer shall be able to
  - Retrieve the Ticket from the Tickets Hand-Out Machine and wait for the Digital Counterpart to call them.
  - Scan the QR Code at QR Code Scanned Machine when they enter **and** leave the store.
- Store Manager shall be able to:
  - Check out the On-Time Store Data.
  - Adjust the maximum number of people in the store.
  - Adjust the order of the queue.
  - Check and reschedule the booking.
- The Store Back-End System shall be able to:
  - Send the available time/date to the the click customers.

  - Received and schedule the click customers' book. The scheduling must refer to the duration time of each customer and the categories of items that the customer intends to buy
  - Calculate the time from the click customer's departure place to the store and put the estimated departure time on the E-Ticket.
  - Plan and put the Store Planned Roadmap on the E-Ticket
  - Send the E-Ticket to the click customers.
  - Send a notification and update the E-Ticket to the customer when their book is rescheduled.
  - Store the customer's data,include:
    * Username
    * Password
    * Valid Booking data
    * History visit data.
    * Is long-term customers
  - Analysis of the historical visit data and calculate the average history duration.
  - Calculate and store the On-Time Store Data.
  - Schedule or reschedule the queue order from the click customer's book and the brick customer's retrieved Ticket.
  - Control the Digital Counterpart and display the queue number.
  - Receive the information from the QR Code Scanned Machine.
  - Receive the information from the Tickets Hand-Out Machine.

2.2.2 Non-Functional Requirements

- The time from the click customer's departure place to the store that calculates from the Store Back-End System be precise enough to avoid the Customer arriving at the store too early/late.
- The Store Back-End System must schedule the queue reasonably to minimize the wait time.
- The Store Back-End System must mix the book and brick customer's retrieved ticket reasonably to allow the click customers to enter the store near the book time by avoiding making the brick customers wait too long.

- Cause everyone needs to do grocery shopping, the software for the click customer should be simple enough to use.

2.3 User characteristics

The system will include three categories of user, each of them with different needs:

- The Click Customer: They can access the required technology, like access our mobile application by a smartphone, and they also have a strong desire to save queuing time through our application.
- The Brick Customer: They can not or do not want to use the required technology, and they want to access the Store in the "most convenient" way, just like visiting the Poste Italiane office, retrieve the queuing number from the Ticket Machine, wait for the call, and then enter the Store, without other operations.
- The Store Manager: They hope to effectively manage the Store through our system in this particular period. Monitor entrance and exit, and reasonably use the well-designed mechanism of our system so that the outside queue will not be too long, inside will not be too crowded, and reduce the customer's risk.

2.4 Assumptions, Dependencies, and Constraints

2.4.1 Domain Assumptions

- D_1 : Everyone will leave the departure place at the departure time indicated by the system.
- D_2 : Everyone who leaves on time can arrive at the store on time.
- D_3 : After shopping, everyone can leave the store in time according to their estimated time.
- D_4 : If something unexpected happens, the store manager can adjust the maximum number of people in the store or reschedule the queue & customer's book reasonably.
- D_5 : If someone's book is rescheduled, he can find the notification in time and set off according to the updated E-Ticket.

- D_6 : Everyone can consciously scan the QR code at the entrance and exit.
- D_7 : Everyone can follow the Planned Roadmap in the store.
- D_8 : If is possible, everyone tries to best book the visit by the software(be a Click Customer) instead of picking up tickets on the spot(not be a Brick Customer).

2.4.2 Goals

There are only three main goals of this system.

- G_1 : Allows store managers to regulate the influx of people in the building.
- G_2 : Saves people from having to line up and stand outside of stores for hours on end.
- G_3 : The application plan visits in a finer way to allow more people in the store, at the same time, let the customer occupy different spaces in the store to keep enough distance between them.

2.4.3 Constraints

There are not many constraints on the Customer's device. They only need a smartphone with the Android or IOS operation system. When they book a visit, the smartphone has to connect with the Internet. At other times, no need for a stable internet connection. They only need to be able to connect to the Internet discontinuously to receive notifications that may appear. The manager needs a PC with a stable network connection. For the Store Back-End System, we buy an Amazon EC2 IaaS to implement this system.

Chapter 3

Specific Requirements

3.1 External Interface Requirements

CLup can be used by the customer who has ability to use smart cellphone. The system will collect data, analyze the data and create the 'Ticket'. Common users can interface themselves with CLup through a mobile application, the manager has ability to access the system's functionalities in mobile application or web-based dashboard. Here is shown every kind of interface that the front-end offers or needs to interact with the users and the back-end.

3.1.1 User Interfaces

In this section there are mock-ups of the user interfaces for the mobile application used by customer.

Login In the Login page Fig.3.1 there are text fields for entering username and password in order to access the personal profile. In case that the customer don't have an account, there is also the option to register.

Sign-up In login page Fig.3.2, the application requests the user to insert the username, email address and a password to register to the system.

Homepage Once registered, the user will be welcomed by the homepage Fig.3.3, that shows the optional of booking, ticket and notification. From booking, users can go to book an appointment for visiting the Store. In tickets button, customers can view all their tickets(ticket history, current ticket and up-coming ticket). As for notification part, if users has an unread notification, there will be a red point shown on the right top, after the user read it, the red point will disappear. With this, if users' appointment has been

rescheduled by the manger, they can easily receive the notification from the manager.

Booking After the user enter the booking button, users can see the page Fig.3.4, customers need to choose time and date from the given selection. Meanwhile, they need to approximate their duration time, for long-term customers, the system will also shown the estimate duration time rely on his shopping duration time history. After, the customer need to select what categories to buy in the store. On the end, there are two ways to insert the depart address: one is manually insertion of the address by customer, another one is automatic positioning by GPS.

Category In this page Fig.3.5, there are lists of product categories in the Store, the product categories corresponding to the different zone in Store. Users can choose one or more categories which they want to buy during the shopping.

Manager Dashboard In the manager dashboard Fig.3.6, the Store Manager can see the number of Customers both inside and outside of the Store. He has the right to control the **Max** number of people inside the Store. If too many or too few people in the same zone of the Store, he can reset the Max number of customer inside the store on web or adjust the Queue. And also if too many people waiting outside the Store, he can click the 'Reschedule booking' button, to reschedule the Click Customer's Booking who haven't departed yet.

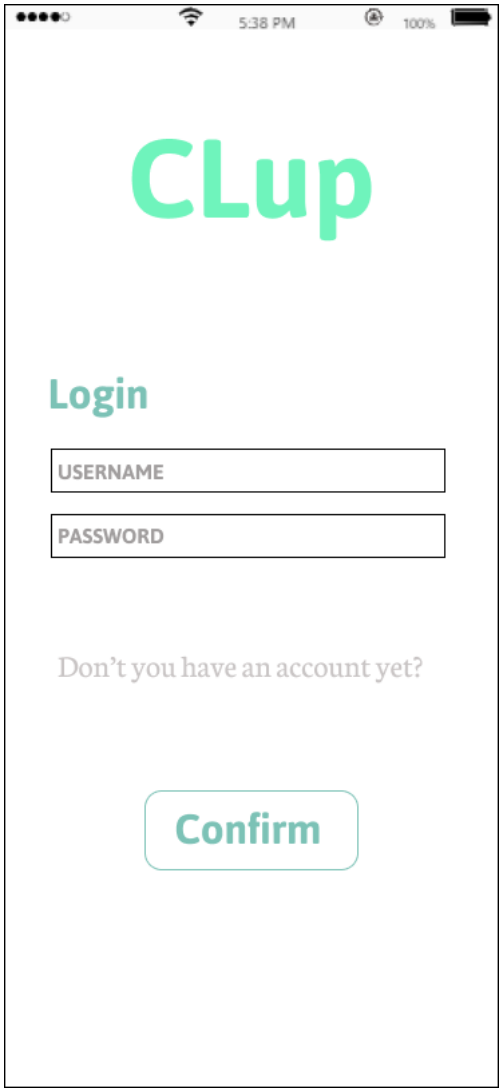

Figure 3.1: UI-Login

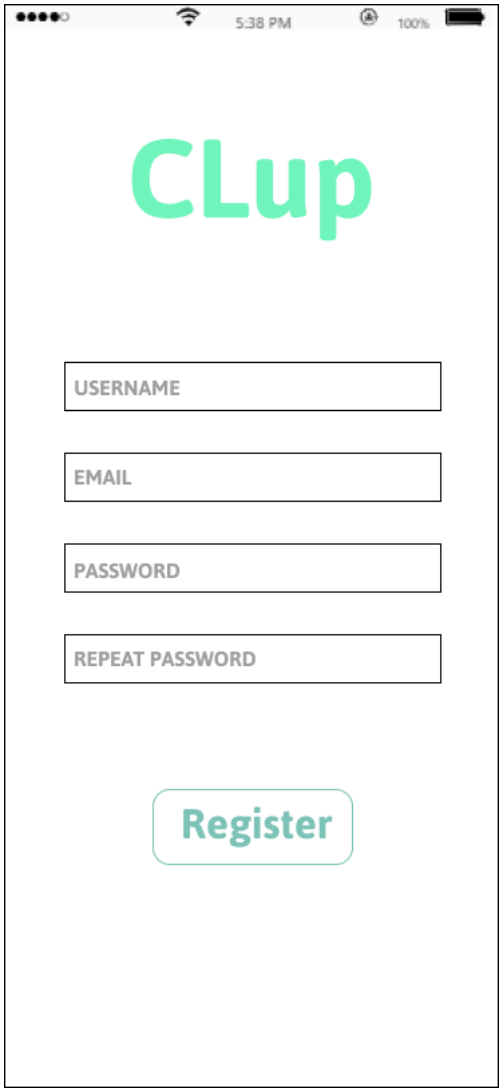

Figure 3.2: UI-Register

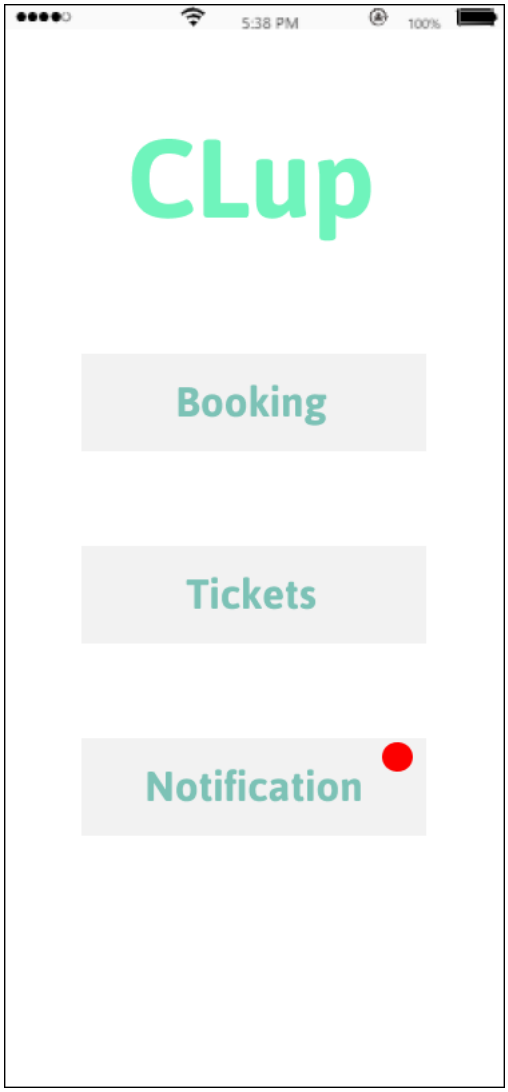

Figure 3.3: UI-Home

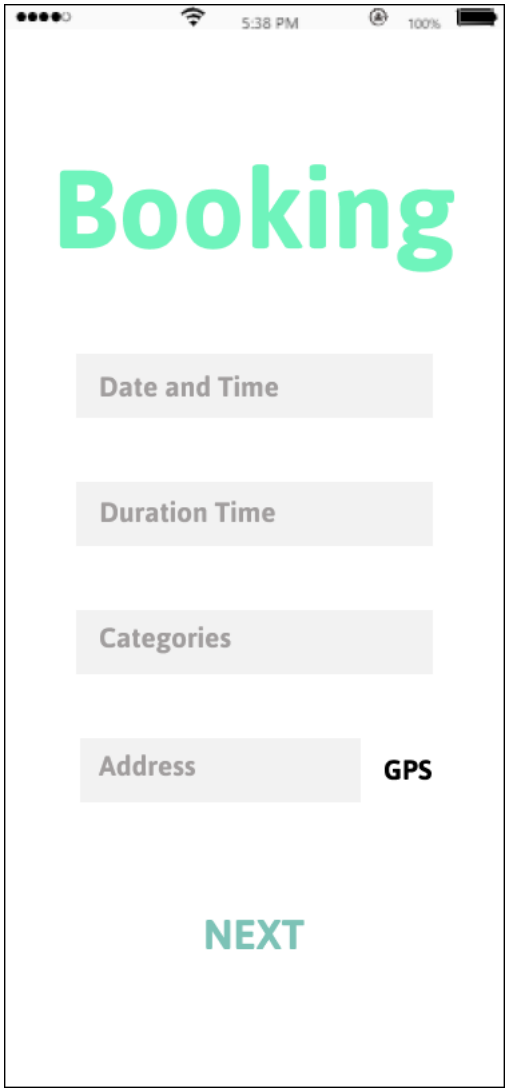

Figure 3.4: UI-Booking

Figure 3.5: UI-Category

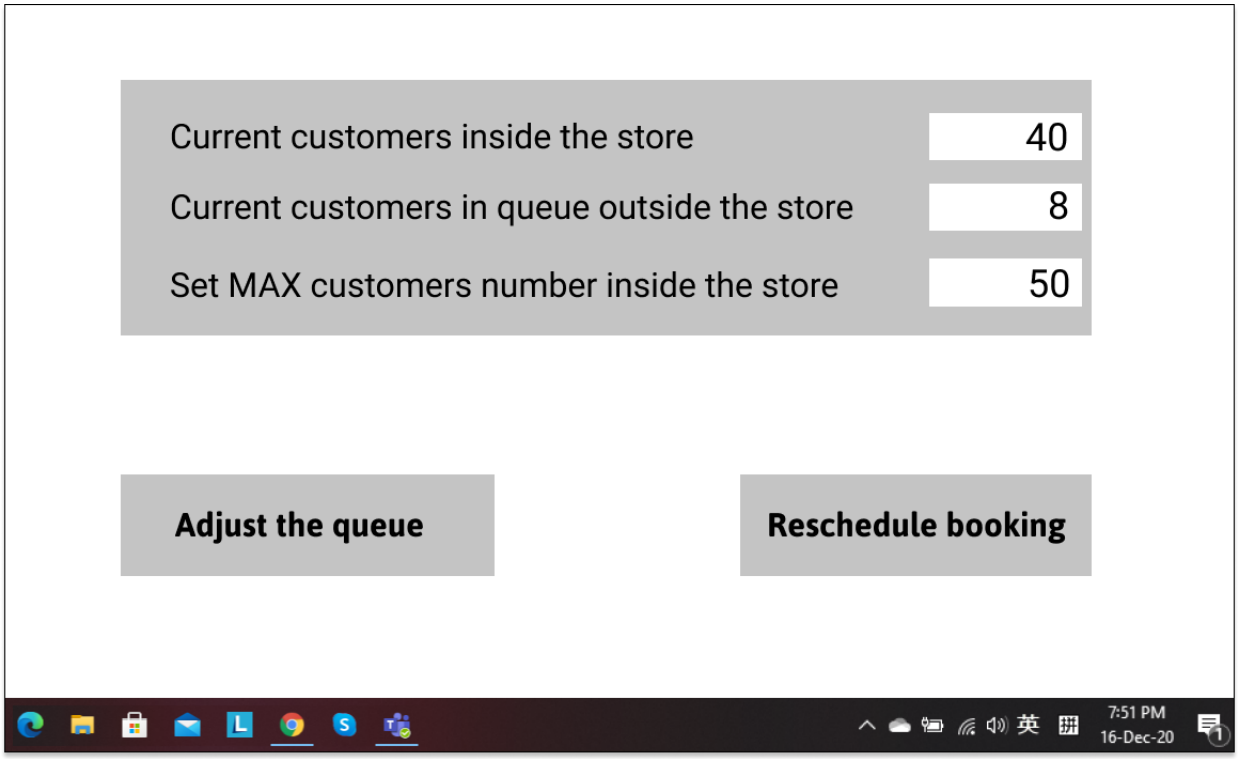

Figure 3.6: UI-Dashboard

3.1.2 Hardware Interfaces

Here is the hardware requirement for users.

- For **Click Customer**: the basic requirement is that a smartphone can access the Internet, and the GPS module is optional. If their smartphone, they can automatically input the address information by the GPS module. Otherwise, they have to input it manually.
- For **Store Manager**: It needs a simple pc can access our Back-End System through the Web Browser. Because we adopt the Thin Client strategy, the hardware requirements for PC configuration are very low. If there is a need to control costs, only a Raspberry Pi can connect to the network cable, and a monitor, mouse, keyboard are enough.
- For the **Back-End System**: We adopted the Buy strategy. We will buy an Amazon EC2 IaaS to implement this system. So the server supplier Amazon will provide the hardware.

3.1.3 Software Interfaces

Mobile applications require the Android operating system or IOS to run on mobile devices. Manager dashboard requirements:

- Microsoft Windows runs downloadable software
- Web browser to access online dashboard

3.1.4 Communication Interfaces

We have some kinds of devices that have to communicate with each other, Click Customer's mobile application, Store Manage's PC, the back-end system, the Digital Counterpart, and the Tickets Hand-Out Machine. The Mobile Phone communicates with the Back-End system via the internet with TCP protocol in the transport layer. For the Store Manager's PC, we plan to use the Web Technology, and the Store Manager's dashboard will show on the HTTP page so that the communication from the PC to Back-End System will hold with HTTP Protocol.

Finally, two hardware devices: Digital Counterpart and Tickets Hand-Out Machine. They will communicate with the Back-End System via a physical network cable work in Ethernet. A design like this is because they are fixed facilities, they do not need "Mobility", but need high availability and stability.

3.2 Functional Requirements

3.2.1 Use Cases

Click Customer's perspective

The following tables describe the use cases from the perspective of the Click Customer.

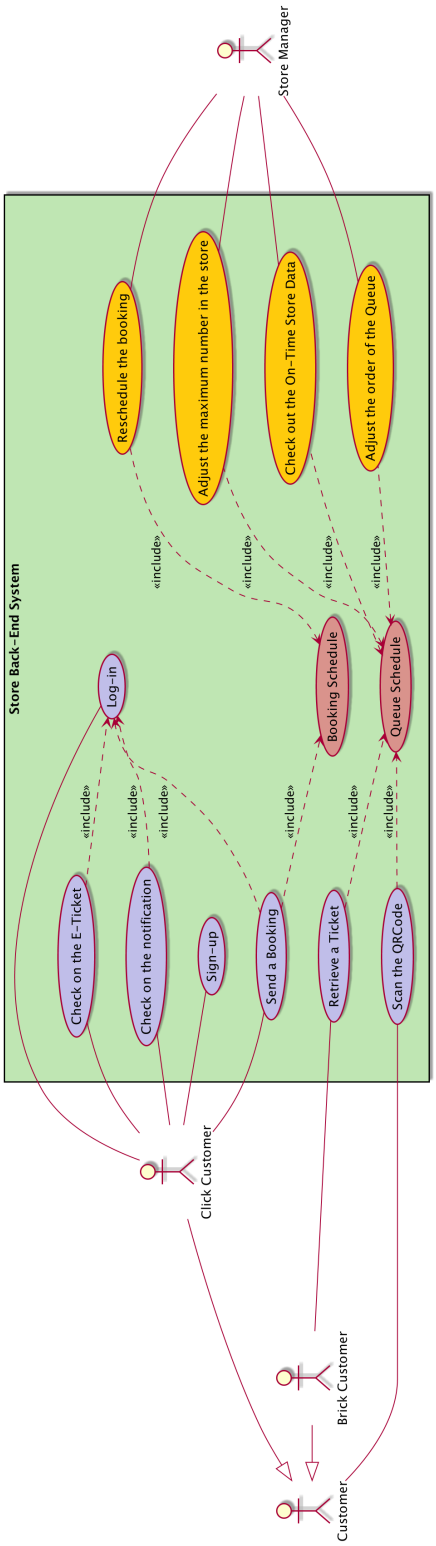

Figure 3.7: CLup Use Case Diagram

Sign-up

Actors	Unregistered Click Customer
Description	A new customer who wants to visit the store by booking, can register himself using the application
Pre-Condition	The customer has a smart phone with Android or IOS system
Flow of events	1. The customer download and install the CLup application 2. The customer open the Clup application and click the "Register" button 3. The CLup application show a form with username, password, confirm password fields and a submit button 4. The customer fill in the form and click the submit button 5. The back-end system validates data and restore this data in the CRM database. 6. The CLup application turns back automatically to the log-in page.
Post-Conditions	The customer now is registered in system, can do the following operations.
Exceptions	If the data inserted are not correct, the flow of events will restart from point 3.

Log-in

Actors	Click Customer
Description	If the Click Customer want to do following operation, they have to log-in first
Pre-Condition	The Click Customer have already registered, and input their username and password correctly
Flow of events	1. The Click Customer open the CLup application. 2. The system will show a form with username and password fields. 3. The customer completes this form with the correct information and clicks the Log-in button. 4. After the back-end system validated the account, the system will enter the main page automatically.
Post-Conditions	The Click Customer is logged-in the app, and he can do the following operations.
Exceptions	If the account info is not correct, the flow of events will restart from point 2.

Send a booking

Actors	Click Customer
Description	After the Log-in, the Click Customer can use this function to book their visit.
Pre-Condition	Already Logged-in.
Flow of events	1. The customer clicks on the book button on the main page. 2. The system will show a form with the available visit date/time, the approximate expected duration of the visit, the categories of items they intend to buy, and the Current Place fields. 3. The customer completes this form with the correct information. For the Current Place field, if they want, they can click on the button near the fields so that the GPS model will provide the Current Place information. 4. The customer clicks on the Submit button. 5. After the back-end system validated the information, the system will store this booking on the booking schedule database. This page will show a "book successful" String and automatically go back to the main page.
Post-Conditions	After the booking schedule system completes the schedule, the Customer will receive the E-Ticket and the notifications. They can check on this via the following operations.
Exceptions	If this booking is not successful, this page does not go back to the main page, the flow of events will restart from point 2.

Check on the E-Tickets

Actors	The Click Customer
Description	The customer can click on the "Check on the E-Ticket" button at all times to check on their bookings.
Pre-Condition	They have to log-in first.
Flow of events	1. The customer clicks on the "Check on the E-Ticket" button on the main page. 2. This page shows all valid E-Ticket and the historical no valid E-ticket. 3. The customer clicks on the E-Ticket they are interested in. 4. The E-Ticket is in PDF format, the application will jump to the PDF reader to open it.
Post-Conditions	After reading it, the system will still stop on the E-Ticket page.
Exceptions	If they did not book any visit, this page would show the "No booking yet" field.

Check on the Notifications

Actors	The Click Customer
Description	The customer can click on the "Check on the Notification" button at all times.
Pre-Condition	They have to log-in first.
Flow of events	1. The customer clicks on the "Check on the Notification" button on the main page. 2. This page shows all notifications, and the unread notifications will mark with a little red icon. 3. The customer clicks on the notification they are interested in. If they read an unread notification, the little red icon will cancel.
Post-Conditions	After reading it, the system will still stop on this page, and if they clicked on all unread notifications so that there is no unread notification, the "Check on the Notification" button on the main page will become the standard color.
Exceptions	If they did have any notifications, this page would show the "No notification yet" field.

Brick Customer's perspective

The following table describe the use cases from the perspective of the Brick Customer.

Retrieve a Ticket

Actors	The Brick Customerw
Description	The Brick Customer has to go to the store and retrieve the paper Ticket on the Tickets Hand-Out Machine.
Pre-Condition	The Tickets Hand-Out Machine is available.
Flow of events	1. The Brick Customer goes to the store and finds the Tickets Hand-Out Machine. 2. Click on the "Retrieve the Ticket" button on the machine. 3. After the Back-End system schedule it in the Queue Schedule system, the machine prints the Ticket. 4. The Brick Customer retrieves the Ticket.
Post-Conditions	
Exceptions	

Customer's perspective

All kinds of customers have to do this operation.

Scan the QRCode

Actors	The Click & Brick Customer
Description	All customers have to scan the QRCode when they enter and exit the store.
Pre-Condition	They hold the Ticket.
Flow of events	1. The customer shows the QRCode in the Ticket. 2. Scan it on the QR Code Scanned Machine when they enter the store. 3. Scan it on the QR Code Scanned Machine when they leave the store.
Post-Conditions	After they scan the QR Code when they leave, the QRCode is invalid.
Exceptions	

Store Manager's perspective

The following tables describe the use cases from the perspective of the Store Manager.

Check out the On-Time Store Data

Actors	The Store Manager
Description	The Store Manager can view the On-Time Store Data at any time.
Pre-Condition	The Back-End system is working well.
Flow of events	1. The manager enters the manage page. 2. The page shows all On-Time Store Data right on the main page.
Post-Conditions	
Exceptions	

Reschedule the booking

Actors	The Store Manager
Description	When the manager considers some area is too crowded, the queue is too long, or other necessary cases make them have to reschedule some customers' booking, they can do this operation.
Pre-Condition	The Back-End system is working well, there is at least one modifiable booking.
Flow of events	1. The manager enters the manage page. 2. Click on the "Reschedule Booking" button. 3. The system jumps to the booking page that shows all the bookings. 4. The system marks the modifiable booking as red color (the bookings with departure time after the current time are modifiable) 5. The manager clicks or searches for a booking or selects some bookings. 6. The manager modifies the modifiable booking and clicks the "Confirm" button. 7. Wait for the Back-End system to deal with all operations and update the booking page automatically.
Post-Conditions	The Back-End system will send the notification and update the E-Ticket for the Click Customer.
Exceptions	If the Store Manager submits the wrong information, or this process is not successful, the flow of events will restart from point 3.

Adjust the order of the Queue

Actors	The Store Manager
Description	When the manager views some area is too crowded, they want customers who will visit these areas to enter the store later, or other necessary cases, they can do this operation to adjust the queue order.
Pre-Condition	The Back-End system is working well, and the queue is more than two customers.
Flow of events	1. The manager enters the manage page. 2. Click on the "Adjust the queue order" button. 3. The system jumps to the queue schedule page that shows the current queue in visualization mode. 4. The manager clicks or searches for a kind of customer or selects some customers. 5. The manager modifies the queue order and clicks the "Confirm" button. 6. Wait for the Back-End system to deal with all operations and update the queue schedule page automatically.
Post-Conditions	After this operation, the Digital Counterpart will call the customer's queue number according to the new queue order.
Exceptions	If the Store Manager submits the wrong information, or this process is not successful, the flow of events will restart from point 3.

Adjust the maximum number in the store

Actors	The Store Manager
Description	When the manager considers there are too many/few people in the store, they can do this operation to regulate the influx of people in the building.
Pre-Condition	The Back-End system is working well.
Flow of events	1. The manager enters the manage page. 2. Just modify the maximum number in the store value on the main page.
Post-Conditions	If the current number is more than the maximum, the Digital Counterpart will stop to call the next customer until the current number few than the maximum. Instead, when the current number is fewer than the maximum, the Digital Counterpart will call fastly, until they are equal.
Exceptions	If the Store Manager submits the wrong information, or this process is not successful, the maximum value will just come back to the previous value.

Sequence Diagram

We have integrated the sequence diagrams from the perspective of the customer and store manager, respectively.

From the customer's perspective Fig.3.8, two kinds of customers are independent and do not interfere. The queue schedule function calls the queue number via the Digital Counterpart, and the customer has to view or listen well their queue number. When their number is called, they scan their QRCode.

From the manager's perspective Fig.3.9, their four operations are independent and do not interfere with each other.

Scenarios

In order to describe these processes more vividly, we give three scenarios from three perspectives.

Scenario 1 - Leonardo lives in a small town with a severe epi-

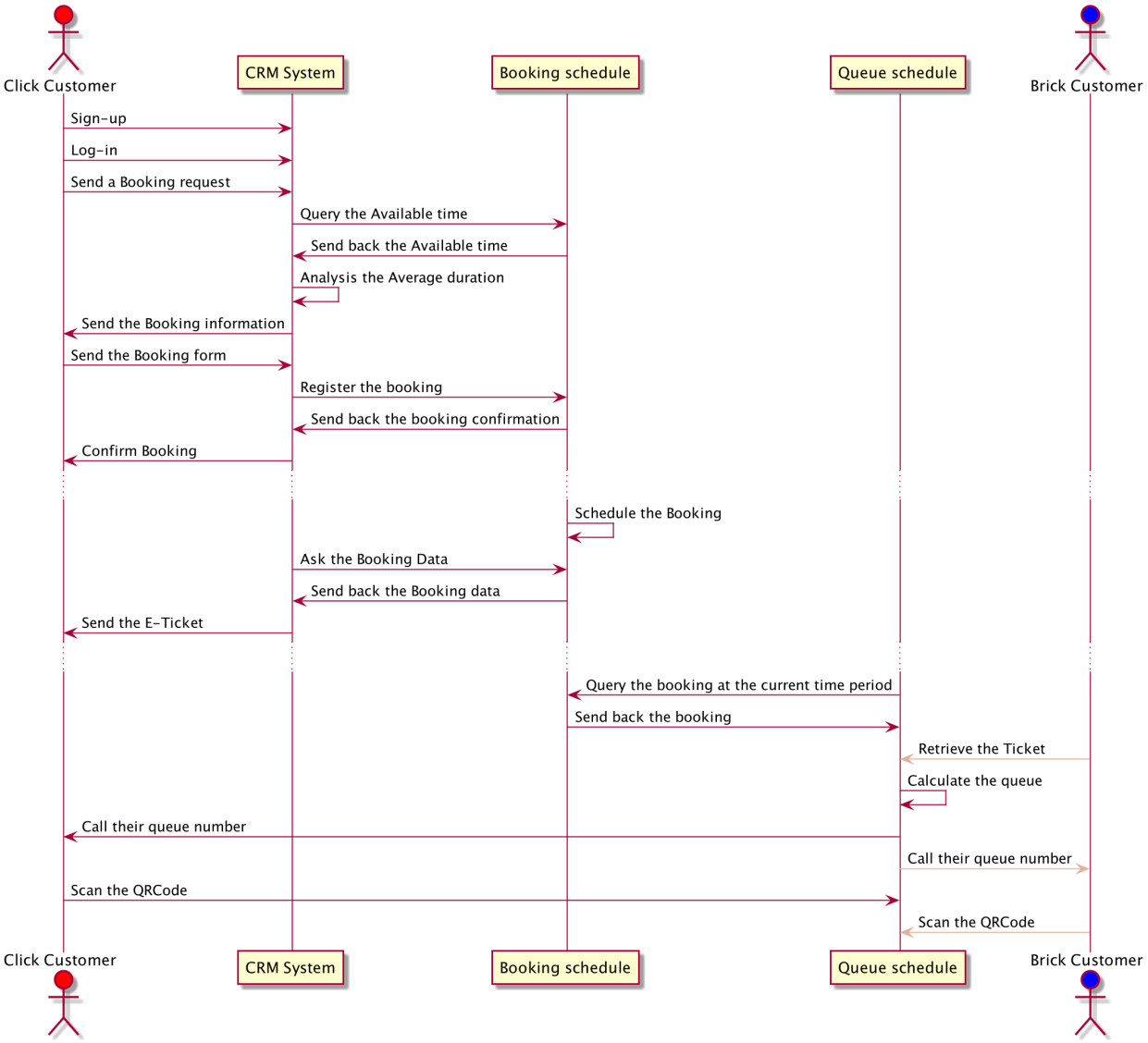

Figure 3.8: Customers' Sequence Diagram

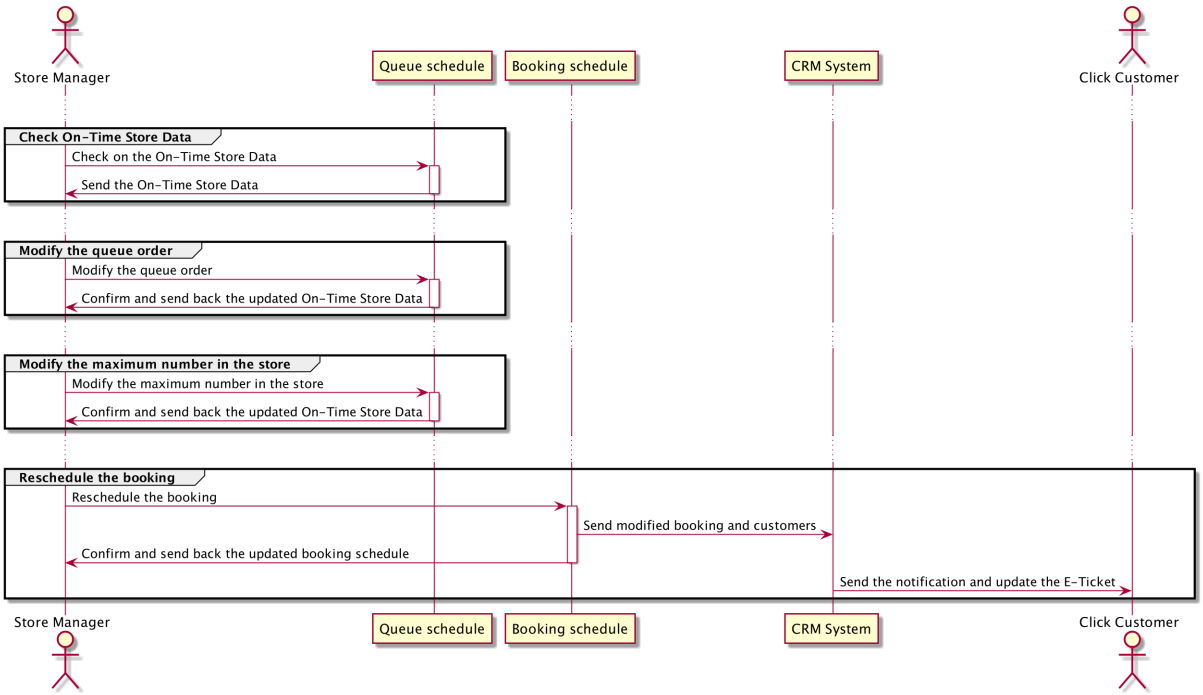

Figure 3.9: Manager's Sequence Diagram

demic: Leonardo, an excellent researcher, loves science, and he studies hard every day. His life is peaceful and happy in that small town. On a cold rainy day, a horrifying thing happened, his town found the COVID-19 cases! All people are terrified, and the mayor urges everyone not to go out. At this time, Leonardo cannot go to his university. Even he hardly goes out except going to the supermarket to buy food. In this challenging time, he discovered that the local supermarket "EsseCorta" was as crowded as usual, and there were also many people gathering. Even because other stores closed, the supermarket was even more crowded. To resolve this problem, Leonardo called on Essecorta to let everyone use CLup to reasonably organize shopping orders. The Essecorta owner was pleased, so within a few days, the system was deployed well. Now Leonardo does not have to take the risk any more in Essecorta. Every weekend, he books a visit on Monday morning at 9:30. Cause lives near it, so from the E-Ticket, Leonardo has to depart from 9:10. When he arrives, he found he even does not have to wait. The Digital Counterpart calls his queue number immediately. He takes out the E-Ticket, scans his QRCode, enters the supermarket, and following the Planned Roadmap on the E-Ticket. How to say? Such convenience! When he bought all items he wants, he rescans the QRCode at the exit and goes back home happy.

Scenario 2 - A lovely Nonna in the same town: Nonna Angola live in this town much year. Facing this epidemic, she was very calm. She believes that this country has experienced many incidents, and people can handle the epidemic. Her life is simple, a coffee in the Morning, a Tea in the evening, that is all. She loves her simple life, and she does not want to change anything. A day, she goes to the supermarket like usual day, the store manager tells her to take a Ticket from a machine, this machine has only a simple button like this lovely Nonna's life, this button is huge, almost occupied all space of the screen, to Nonna, it is not very difficult to use. After getting her Ticket, Nonna waits for the Digital Counterpart to call her number. A few seconds later, Nonna enters the market. The store manager helps her scan the QRCode when she enters and leaves the market.

Scenario 3 - The manager who want everyone safe: Luca is a store manager. He was a bus driver before, but the cause of this unfortunate time, his bus was stopped, he had to find another job. The Essecorto owner found him because the driver has to keep every passager safe, so this job too! He has to look at the market at all times. One day, he finds that there are many people in the Gelato area. He realizes that it is not a good thing, so he reduces the store's maximum number. After a few minutes, he finds that is not working, other areas have a few people, but the Gelato area still crowed. Why? Maybe this day has many children like Gelato, and these children did not book this visit. Instead, they all piked the Tiket on the Machine,

so the system did not monitor all this. Then Luca decided to adopt plan B. He checks the queue schedule and puts the customers who are also visiting the Gelato area later. Five minutes later, other managers tell him the adjustment is not enough, so Luca decided to take the last resort - reschedule the Gelato's customers' booking! He opens the booking schedule page and searches the keyword "Gelato" and modifies all valid results' customers' booking and postpones this for about 1 hour. Finally, ten minutes later, this area was clean. He did an excellent job!

3.2.2 Mapping on requirements

G_1 : Allows store managers to regulate the influx of people in the building

- Requirements:

- R_7: The Store Manager shall be able to Check out the On-Time Store Data.
- R_8: The Store Manager shall be able to Adjust the maximum number of people in the store.
- R_9: The Store Manager shall be able to Adjust the order of the queue.
- R_{10}: The Store Manager shall be able to Check and reschedule the booking.
- R_{12}: The Store Back-End System shall be able to received and schedule the click customers' book. The scheduling must refer to the duration time of each Customer and the categories of items that the Customer intends to buy.
- R_{15}: The Store Back-End System shall send a notification and update the E-Ticket to the Customer when their book is rescheduled.
- R_{17}: The Store Back-End System shall analyze the historical visit data and calculate the average history duration for the Long-Term Customers.
- R_{18}: The Store Back-End System shall be able to Calculate and store the On-Time Store Data.
- R_{19}: The Store Back-End System shall schedule or reschedule the queue order from click customer's book and the brick customer's retrieved ticket.

- R_{20}: The Store Back-End System shall be able to Control the Digital Counterpart and display the queue number.
- R_{21}: The Store Back-End System shall be able to receive the information from the QR Code Scanned Machine. Receive the information from the Tickets Hand-Out Machine.

- Domain Assumptions:

- D_4 : If something unexpected happens, the store manager can reasonably adjust the maximum number of people in the store or reschedule the queue & Customer's book.
- D_5 : If someone's booking is rescheduled, he can find the notification in time and set off according to the updated E-Ticket.

G_2 : Saves people from having to line up and stand outside of stores for hours on end

- Requirements:

- R_1: Each Click Customer shall be able to sign-up and Log-in.
- R_2: Each Click Customer shall be able to book a visit.
- R_3: Each Click Customer shall be able to check on the E-Ticket.
- R_4: Each Click Customer shall check on the store manager's notification when their book is rescheduled.
- R_5: Each Brick Customer shall be able to retrieve the Ticket from the Tickets Hand-Out Machine and wait for the Digital Counterpart to call them.
- R_6: Each Customer shall scan the QR Code at QR Code scanned machine when they enter and leave the store.
- R_{11}: The Store Back-End System shall send the available time/date to the click customers.
- R_{12}: The Store Back-End System shall be able to received and schedule the click customers' book. The scheduling must refer to the duration time of each Customer and the categories of items that the Customer intends to buy.

- R_{13}: The Store Back-End System shall calculate the time from the click customer's departure place to the store and put the estimated departure time on the E-Ticket.
- R_{15}: The Store Back-End System shall send a notification and update the E-Ticket to the Customer when their book is rescheduled.
- R_{16}: The Store Back-End System shall be able to store the Customer's data.
- R_{17}: The Store Back-End System shall analyze the history visit data and mark the Long-Term Customers.
- NR_1:The time from the click customer's departure place to the store that calculates from the Store Back-End System must be precise enough to avoid the Customer arriving at the store too early/late.
- NR_2:The Store Back-End System must schedule the queue reasonably to minimize the wait time.
- NR_3:The Store Back-End System must mix the book and brick customer's retrieved ticket reasonably to allow the click customers to enter the store near the book time by avoiding making the brick customers wait too long.
- NR_4:Cause everyone needs to do grocery shopping, the software for the click customer should be simple enough to use.

- Domain Assumptions:

- D_1 : Everyone will leave the departure place at the departure time indicated by the system.
- D_2 : Everyone who leaves on time can arrive at the store on time.
- D_3 : After shopping, everyone can leave the store in time according to their estimated time.
- D_5 : If someone's booking is rescheduled, he can find the notification in time and set off according to the updated E-Ticket.
- D_6 : Everyone can consciously scan the QR code at the entrance and exit.
- D_8 : If possible, everyone tries to book the visit via the application(be a Click Customer) instead of picking up tickets on the spot(not be a Brick Customer).

G_3 **: The application plan visits in a finer way to allow more people in the store, at the same time, let the customer occupy different spaces in the store to keep enough distance between them**

- Requirements:

- R_{12}: The Store Back-End System shall be able to received and schedule the click customers' book. The scheduling must refer to the duration time of each Customer and the categories of items that the Customer intends to buy.
- R_{14}: The Store Back-End System shall Plan and put the Store Planned Roadmap on the E-Ticket Send the E-Ticket to the click customers.
- R_{17}: The Store Back-End System shall analyze the history visit data and mark the Long-Term Customers.

- Domain Assumptions:

- D_3 : After shopping, everyone can leave the store in time according to their estimated time.
- D_7 : Everyone can follow the Planned Roadmap in the store.

Traceability Matrix

Goals	Requirements	Domain Assumptions
G_1	R_7, R_8, R_9, R_{10}, R_{12}, R_{15}, R_{17}, R_{18}, R_{19}, R_{20}, R_{21}	D_4, D_5
G_2	R_1, R_2, R_3, R_4, R_5, R_6, R_{11}, R_{12}, R_{13}, R_{15}, R_{16}, R_{17}, NR_1, NR_2, NR_3, NR_4	D_1, D_2, D_3, D_5, D_6, D_8
G_3	R_{12}, R_{14}, R_{17}	D_3, D_7

In summary, we proved the **Requirements Completeness** of our system, so that **R and D** $\models$ **G**.

3.3 Performance Requirements

The Click Customer must have a smartphone with the Android or IOS operation system. When they open the book a visit page, the Back-End system has to send them the available data/time within 1 second. After they send

the form, the Back-End system has to confirm it within 5 seconds. And then, the Back-End system has to schedule this booking, generate QRCode, Queue Number, Estimated departure time, and Planned Roadmap, integrate them to the E-Ticket, and send back the E-Ticket so soon as possible. Cause calculate the Planned Roadmap Operation has to wait for the other Customer's data, so this operation can not complete quickly. However, this operation must be finished more than 24 hours before departure. If they booked the visit on the current day, the Back-End has to send the E-Ticket before the departure time.

For the Brick Customer, no need for them to hold any personal drives. They have to learn how to use the Ticket Hand-Out Machine. As the range of users includes all demographics, this machine has to be easy enough to use. When the Brick Customer retrieves the Ticket, the paper Ticket has to print in 1 second.

For the Store Manager, their management system will implement in a PC with a stable network connection. When they do any operation, this system has to react in 5 seconds.

For the Back-End system, we will buy an Amazon EC2 IaaS to implement it. This system will communicate with all other ends, so we have to keep this system always working well, cause it is Iaas, we still have to keep our software function, the KPI is every month the unavailable service time does not exceed 1 hour.

3.4 Design Constraints

3.4.1 Standards compliance

Our system complies with the following standard for the GPS module:

- GPS data will give latitude and longitude degrees, but our software will use the Google API and convert it to a readable address.

3.4.2 Hardware limitations

For our Click Customer application to run well on smartphones, we must follow the following hardware limitations:

- Minimum 256 MB disk space
- Minimum 64 MB RAM
- Optional GPS module

- Network Module

For our Management System run on the PC, the hardware limitations are:

- Minimum 256 MB disk space
- Minimum 64 MB RAM
- Ethernet network cable

Furthermore, the hardware limitation of the Back-End System are:

- Minimum 1TB disk space or Elastic space.
- Minimum 1 GB RAM.

3.5 Software System Attributes

The CLup System includes different services, Mobile Application, Management System For Manager, the Back-End System, and some other hardware devices. We will separately talk about those specific system attributes.

3.5.1 Reliability

For the Back-End System, we adopted the Buy strategy so that the Iaas provider will ensure the system's reliability below the infrastructure level. We only have to care about our software. In the development phase, we will strictly observe the Verification and Validation process. In daily operation, we will regularly check the database data to find out if any errors occur.

For the Mobile Application, when some errors occur, we will ask the Customer's permission to send back the error log, and we will fix it in the next version of our software.

3.5.2 Availability

For the Back-End System's queuing function, we must ensure that 99% of the time is available during Store business hours. But for CRM and Booking function, we will provide 95% of the time is available on 24/24 up to 7.

And for the hardware in the Store, the Digital Counterpart, and the Ticket Hand-Out Machine, we have to ensure that 99.99% of the time is available during Store business hours.

For the Management System of the Store Manager, available at least 99% of the time.

3.5.3 Security

Our system is not an army or government system, and there is no very high level of data security requirements. In our Management System, based on the HTTP protocol, we will use the SSL protocol to ensure data security.

To protect our back-end database, we decided to adopt DAC (Discretionary Access Control) data access policy, the Store Manager will manage all data.

3.5.4 Maintainability

We Back-End System's code will write in Java. To ensure the readable, all our developers have to refer to the "Clean Code: A Handbook of Agile Software Craftsmanship" handbook. And after all, we will write and general the JavaDoc to improve maintainability.

3.5.5 Portability

For the Click Customer's Application, it has to can run well in Android and IOS systems. And for our Management System, it must support some popular web browsers, for example, Microsoft Edge, Google Chrome, and Safari.

Chapter 4

Formal Analysis Using Alloy

Here are the Sig and Fact part of our Alloy Code. They define the types, relationships, and properties of our models. In our model, the most important part is the relationship of Booking, two kinds of Customers, and two Schedules. We showed them perfectly through this Alloy model.

```
1  sig TimePeriod{ }
2
3  sig UserName{ }
4
5  sig Ticket{ }
6
7  sig Booking{
8    BookedVisitTime: one TimePeriod,
9    ValidTicket: lone Ticket
10 }
11
12
13 one sig QueueSchedule{
14
15   MaximumCustomerInStor: Int,
16   CustomerInStore: set Customer,
17   CurrentQueue: set Customer
18
19 }{
20
21   MaximumCustomerInStor >= 0
22
23   // CustomerInStore must be less than or equal to
       MaximumCustomerInStor
24   #CustomerInStore <= MaximumCustomerInStor
25
26   //One customer can only appear one time in QueueSchedule
```

```
27   CustomerInStore & CurrentQueue = none
28
29 }
30
31
32 abstract sig Customer{
33
34 }
35
36
37 sig ClickCustomer extends Customer{
38
39   Username: one UserName,
40   ValidBooking: lone Booking,
41   HistoryBooking: set Booking
42
43 }{
44
45   //Hostory Booking must have Valid Ticket
46   all hb: HistoryBooking | hb.ValidTicket != none
47
48 }
49
50 sig BrickCustomer extends Customer{
51
52   ValidTicket: one Ticket
53
54 }
55
56 one sig BookingSchedule{
57
58   AllBooking: TimePeriod -> set Booking,
59   MaximumBookingInEachTimePeriod: Int
60
61 }{
62
63   //Each TimePeriod's booking less than Maximum value
64   all tp: TimePeriod | #tp.AllBooking <=
      MaximumBookingInEachTimePeriod
65
66 }
67
68
69
70 //The intersection of ValidBooking and HistoryBooking must be
       empty
71 fact VaildHistoryBookingMustBeDifferent{
72
```

```
73    all c: ClickCustomer | c.ValidBooking != none and c.
      HistoryBooking != none =>
74      c.ValidBooking not in c.HistoryBooking
75
76    all disj c1, c2 : ClickCustomer | c1.ValidBooking != none
      and c2.ValidBooking != none =>
77      c1.ValidBooking != c2.ValidBooking
78
79    all disj c1, c2 : ClickCustomer | c1.HistoryBooking != none
        and c2.HistoryBooking  != none =>
80      c1.HistoryBooking & c2.HistoryBooking = none
81
82    all disj c1, c2 : ClickCustomer | c1.ValidBooking != none
      and c2.HistoryBooking  != none =>
83      c1.ValidBooking not in c2.HistoryBooking
84
85  }
86
87
88  //Each Click Customer's username must be different
89  fact UniqueUsername{
90
91    all disj c1, c2 : ClickCustomer | c1.Username != c2.
      Username
92
93  }
94
95
96  fact noUselessUsername{
97
98     all u : UserName | one cc : ClickCustomer | u = cc.
      Username
99  }
100
101
102 fact noUselessBooking{
103
104   all b : Booking | some  bsa: BookingSchedule.AllBooking | b
        in TimePeriod.bsa
105
106 }
107
108 fact noUselessTicket{
109
110   all t : Ticket | some b: Booking | some bc: BrickCustomer
      |
111     t = b.ValidTicket or t = bc.ValidTicket
112
113 }
```

```
114
115
116 //All ClickCustomer's Booking must in BookingSchedule
117 fact AllBookingStoreInBookingSchedule{
118
119   all c: ClickCustomer | some b: BookingSchedule.AllBooking |
        c.ValidBooking in TimePeriod.b
120   all chb: ClickCustomer.HistoryBooking | some b:
      BookingSchedule.AllBooking | chb in TimePeriod.b
121
122 }
123
124 //All BookingSchedule's Booking must in  ClickCustomer's
      Booking
125 fact AllBookingScheduleBookingMustInClickCustomerBooking{
126
127   all b: TimePeriod.(BookingSchedule.AllBooking) |
128     some c: ClickCustomer | b = c.ValidBooking or b in c.
      HistoryBooking
129
130 }
131
132
133
134 //All Ticket must be Different
135 fact AllTicketMustBeDifferent{
136
137   all disj bc1, bc2: BrickCustomer | bc1.ValidTicket != bc2.
      ValidTicket
138   all disj b1,b2: Booking | b1.ValidTicket != b2.ValidTicket
139   all b: Booking | no bc: BrickCustomer | b.ValidTicket !=
      none => b.ValidTicket = bc.ValidTicket
140
141 }
142
143 //All Click Customer in QueueSchedule must have a valid
      booking with a valid ticket
144 fact
      ClickCustomerInQueueScheduleMustHaveValidBookingWithValidTicket
      {
145
146   all cicq : QueueSchedule.CurrentQueue |
147     cicq in ClickCustomer => cicq.ValidBooking != none and
148     (cicq.ValidBooking).ValidTicket != none
149
150   all cis : QueueSchedule.CustomerInStore |
151     cis in ClickCustomer => cis.ValidBooking != none and
152     (cis.ValidBooking).ValidTicket != none
153
```

```
154 }
155
156
157 //Cause the BrickCustomer's property, so they must always be
       in the QueueSchedule.
158 fact BrickCustomerMustBeInQueueSchedule{
159
160   all bc: BrickCustomer |
161   some is: QueueSchedule.CustomerInStore |
162   some cq: QueueSchedule.CurrentQueue |
163     (bc in is or bc in cq)
164
165 }
```

In the **pred** part, we want to show this model separately from two customer perspectives, and after all, we will show a general world.

From the BrickCutomer's perspective Fig.4.1, they can retrieve a ticket from Machina and don't have to Booking. Everyone has to hold a valid Ticket at the hold time, and every Brick Customer will appear in the set of Queue Schedule, either CustomerInStore or CurrentQueue.

```
1 pred BrickCustomer{
2
3   #BrickCustomer = 5
4   #ClickCustomer = 0
5   #Booking = 0
6
7 }
8
9 run BrickCustomer for 10
```

```
Executing "Run BrickCustomer for 10"
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
  27558 vars. 4012 primary vars. 47525 clauses. 86ms.
  Instance found. Predicate is consistent. 59ms.
```

From the Click Customer's perspective Fig.4.2, they have a unique Username, and some of them may do not hold a Booking, for example, a new User. To the Click Customer with a valid Booking, his Valid Booking's Valid Ticket may still not out. However, when they are in the QueueSchedule, they

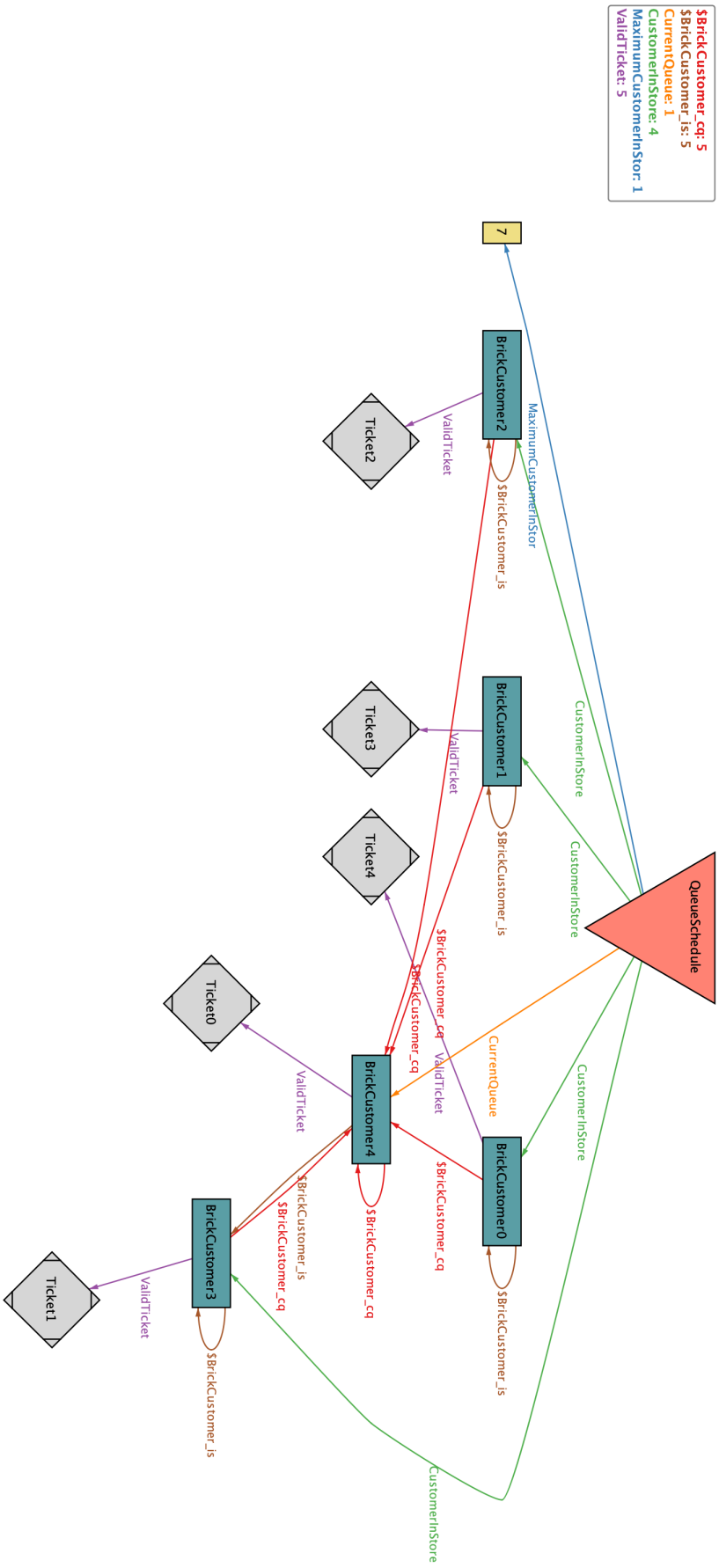

Figure 4.1: The Alloy world of the Brick Customer's perspective

must hold a Valid Booking with a Valid Ticket. This constraint has already been declared in fact part. Now we ran a case with some different situations.

```
pred ClickCustomer {

  #ClickCustomer  > 1
  one c: ClickCustomer | c.ValidBooking = none and #c.
   HistoryBooking = 0
  one c: ClickCustomer | c.ValidBooking != none and #c.
   HistoryBooking = 0
  some c: ClickCustomer | c.ValidBooking != none and #c.
   HistoryBooking > 1
  #TimePeriod.(BookingSchedule.AllBooking) > 1
  #BrickCustomer = 1
  #QueueSchedule.CustomerInStore > 1
  #QueueSchedule.CurrentQueue > 1

}

run ClickCustomer for 10
```

```
Executing "Run ClickCustomer for 10"
  Sig this/QueueSchedule scope <= 1
  Sig this/BookingSchedule scope <= 1
  Sig this/TimePeriod scope <= 10
  Sig this/UserName scope <= 10
  Sig this/Ticket scope <= 10
  Sig this/Booking scope <= 10
  Sig this/Customer scope <= 10
  Sig this/ClickCustomer scope <= 10
  Sig this/BrickCustomer scope <= 10
  Sig this/TimePeriod in [[TimePeriod$0], [TimePeriod$1], [TimePeriod$2], [TimePeriod$3], [TimePeriod$4], [TimePeriod$5], [TimePeriod$6], [TimePeriod$7], [TimePeriod$8], [TimePeriod$9]]
  Sig this/UserName in [[UserName$0], [UserName$1], [UserName$2], [UserName$3], [UserName$4], [UserName$5], [UserName$6], [UserName$7], [UserName$8], [UserName$9]]
  Sig this/Ticket in [[Ticket$0], [Ticket$1], [Ticket$2], [Ticket$3], [Ticket$4], [Ticket$5], [Ticket$6], [Ticket$7], [Ticket$8], [Ticket$9]]
  Sig this/Booking in [[Booking$0], [Booking$1], [Booking$2], [Booking$3], [Booking$4], [Booking$5], [Booking$6], [Booking$7], [Booking$8], [Booking$9]]
  Sig this/QueueSchedule == [[QueueSchedule$0]]
  Sig this/Customer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/ClickCustomer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/BrickCustomer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/BookingSchedule == [[BookingSchedule$0]]
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
  36517 vars. 4322 primary vars. 64208 clauses. 66ms.
  Instance found. Predicate is consistent. 465ms.
```

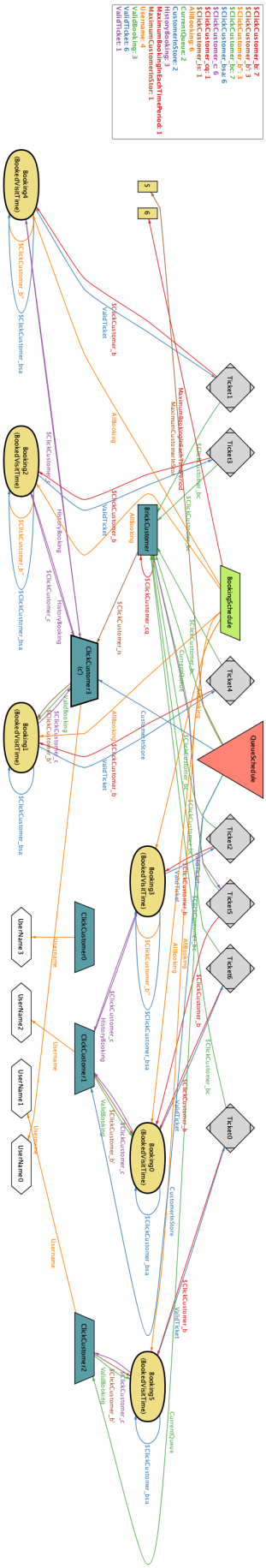

Figure 4.2: The Alloy world of the Click Customer's perspective

Finally, we are trying to give a "General World" of our Alloy model, shown in Fig.4.3.

```
pred show {

  #ClickCustomer > 3
  #ClickCustomer.ValidBooking > 1
  #ClickCustomer.HistoryBooking > 3
  #TimePeriod.(BookingSchedule.AllBooking) > 1
  #BrickCustomer > 3
  #QueueSchedule.CustomerInStore > 2

}

run show for 10
```

```
Executing "Run show for 10"
  Sig this/QueueSchedule scope <= 1
  Sig this/BookingSchedule scope <= 1
  Sig this/TimePeriod scope <= 10
  Sig this/UserName scope <= 10
  Sig this/Ticket scope <= 10
  Sig this/Booking scope <= 10
  Sig this/Customer scope <= 10
  Sig this/ClickCustomer scope <= 10
  Sig this/BrickCustomer scope <= 10
  Sig this/TimePeriod in [[TimePeriod$0], [TimePeriod$1], [TimePeriod$2], [TimePeriod$3], [TimePeriod$4], [TimePeriod$5], [TimePeriod$6], [TimePeriod$7], [TimePeriod$8], [TimePeriod$9]]
  Sig this/UserName in [[UserName$0], [UserName$1], [UserName$2], [UserName$3], [UserName$4], [UserName$5], [UserName$6], [UserName$7], [UserName$8], [UserName$9]]
  Sig this/Ticket in [[Ticket$0], [Ticket$1], [Ticket$2], [Ticket$3], [Ticket$4], [Ticket$5], [Ticket$6], [Ticket$7], [Ticket$8], [Ticket$9]]
  Sig this/Booking in [[Booking$0], [Booking$1], [Booking$2], [Booking$3], [Booking$4], [Booking$5], [Booking$6], [Booking$7], [Booking$8], [Booking$9]]
  Sig this/QueueSchedule == [[QueueSchedule$0]]
  Sig this/Customer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/ClickCustomer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/BrickCustomer in [[Customer$0], [Customer$1], [Customer$2], [Customer$3], [Customer$4], [Customer$5], [Customer$6], [Customer$7], [Customer$8], [Customer$9]]
  Sig this/BookingSchedule == [[BookingSchedule$0]]
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
  35697 vars. 4312 primary vars. 61256 clauses. 61ms.
  Instance found. Predicate is consistent. 3689ms.
```

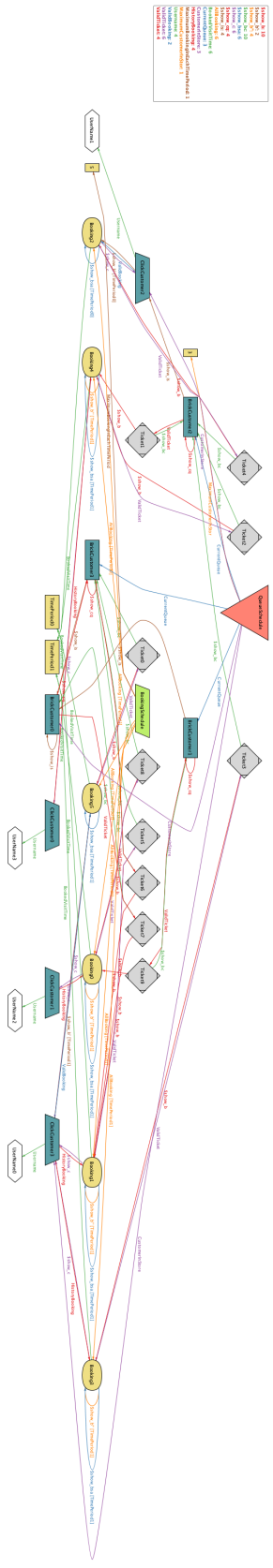

Figure 4.3: The Alloy General World

Chapter 5

Effort Spent

- **Kong Xiangyi**

Date	Task	Hours
2020/10/10	Group discussion project plan	4h
2020/10/31	Modified the purpose and scope of the RASD	2h
2020/11/14	Drawn the state diagram in the Section 2.1.1	2h
2020/11/26	Design and drawn the UI in section 3.1.1	2h
2020/12/5	Completed the User Interface requirement	3h
2020/12/6	Inspected the section 2	1h
2020/12/10	Wrote the overview	1h
2020/12/18	Completed the section 3.4 and fixed the problem in chapter 3	2h
2020/12/21	Completed phenomena and modify the state diagram	1h
2020/12/21	Review and last discussion	3h

- **Zhang Yuedong**

Date	Task	Hours
2020/10/10	Group discussion project plan	4h
2020/10/19	Added the project's architecture	1h
2020/10/30	Added the purpose and scope of the RASD	2h
2020/11/16	Wrote the product functions part	4h
2020/11/17	Drawn the class diagram in the Section 2.1.1	2h
2020/11/27	Fixed the problem of the second chapter	1h
2020/12/04	Completed the use case, sequence, and the scenario in the Section 3.2	4h
2020/12/05	Completed the Mapping on requirements in the section 3.2	2h
2020/12/06	Completed the Performance Requirements	1h
2020/12/20	Coding for Alloy	4h
2020/12/22	Did the communication interface subsection and some TODO mark	2h
2020/12/21	Review and last discussion	3h