POLITECNICO
MILANO 1863

Travlendar+ project Neroni, Pozzi, Vetere

Design Document

Deliverable:	DD
Title:	Design Document
Authors:	Cristiano Neroni, Davide Pozzi, Maurizio Vetere
Version:	1.0
Date:	10-January-2021
Download page:	https://github.com/pollo-fritto/PozziNeroniVetere
Copyright:	Copyright © 2021, Neroni| Pozzi| Vetere – All rights reserved

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of this document is to provide more technical and detailed information about the software discussed in the RASD document. It will represent a strong guide for the programmers that will develop the application considering its different parts: the basic service and the two advanced functions. In this DD we present hardware and software architecture of the system in terms of components and interactions among those components. Furthermore, this document describes a set of design characteristics required for the implementation by introducing constraints and quality attributes. It also gives a detailed presentation of the implementation plan, integration plan and the testing plan. In general, the main different features listed in this document are:

- The high-level architecture of the system
- Main components of the system
- Interfaces provided by the components
- Design patterns adopted

Stakeholders are invited to read this document in order to understand the characteristics of the project being aware of the choices that have been made to offer all the functionalities also satisfying the quality requirements.

1.2 Scope

Clup is an application that aims to avoid users from crowding outside supermarkets when doing grocery shopping in pandemic times.
The application can be used both by store customers and store managers. On one hand users can virtually queue by Clup to enter the supermarket and they are provided with real time information about the line, in this way they can arrive at the entrance only when they are allowed to enter. On the other hand the application monitors and stores the information about people fluxes; this data is then provided to store managers who can take actions depending on the situation. The few paragraphs just read represent an overview of the main functionalities offered by the system: more detailed information can be found on the RASD document.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

1.3.2 Acronyms

1.3.3 Abbreviations

1.4 Revision history

1.5 Reference Documents

1.6 Document Structure

- Chapter 1 describes the scope and purpose of the DD, including the structure of the document and the set of definitions, acronyms and abbreviations used.
- Chapter 2 contains the architectural design choice, it includes all the components, the interfaces, the technologies (both hardware and software) used for the development of the application. It also

includes the main functions of the interfaces and the processes in which they are utilised (Runtime view and component interfaces). Finally, there is the explanation of the architectural patterns chosen with the other design decisions.

- Chapter 3 shows how the user interface should be on the mobile and web application.
- Chapter 4 describes the connection between the RASD and the DD, showing the matching between the goals and requirements described previously with the elements which compose the architecture of the application.
- Chapter 5 traces a plan for the development of components to maximize the efficiency of the developer team and the quality controls team. It is divided in two sections: implementation and integration. It also includes the testing strategy.
- Chapter 6 shows the effort spent for each member of the group.
- Chapter 7 includes the reference documents.

2 Architectural Design

2.1 Overview

CLup's architecture is layered as follows:

- **Presentation layer** (P) handles the interaction with users. It contains the interfaces able to communicate with them and it is responsible for rendering of the information. Its scope is to make understandable the functions of the application to the customers.
- **Application layer** (A) takes care of the functions to be provided for the users. It also coordinates the work of the application, making logical decisions and moving data between the other two layers.
- **Data access layer** (D) which takes care of the information management, database access control. It also handles data retrieval and passes them to upper level layers.

The architecture style chosen for CLup is the **multi-tier** one. As previously anticipated in the Requirements Analysis and Specifications Document, there will be at least one server for each one of the following interest areas:

- Bookings
- Queues
- Notifications
- Stores
- Staff members
- Customers

This is mainly done to distribute workload as well as making the overall system more robust.
There will be also at least two servers for the two following functionalities:

- Customer related functionalities
- Staff related functionalities

The bookings, queue and notifications databases will be distributed and replicated all over the entire store list. Each store will have its own instance of bookings, queue and notifications database while a central logic server will act as a request redirector towards them whenever needed.

In Figure 1 is represented the high-level architecture of the system: customers clients, through an internet connection, can connect to Clup's App Servers, which represent the A layer of the Clup customer side. Client's operations store o retrieve information from People DB Servers or directly from the Local DB of the store, depending by the kind of information. Staff clients instead are connected through a LAN connection to their Local Web Server, who will redirect their requests to the Staff Operations Manager server. This machine is expected to process the logic of requests and then submit them to the Local Application server who also store or retrieve the information from the Local DB or People DB server depending by the situation. Local Application Server is fundamental also for Totem users, in fact the Totem rely on this server for its requests. Staff clients may need to connect to Clup Centralized Hardware, to do this they need an internet connection to get in contact with the dedicated Central Web server, the P layer who then send the request to the Store Operations Manager, the A layer. This server stores and retrieves information from Stores DB servers

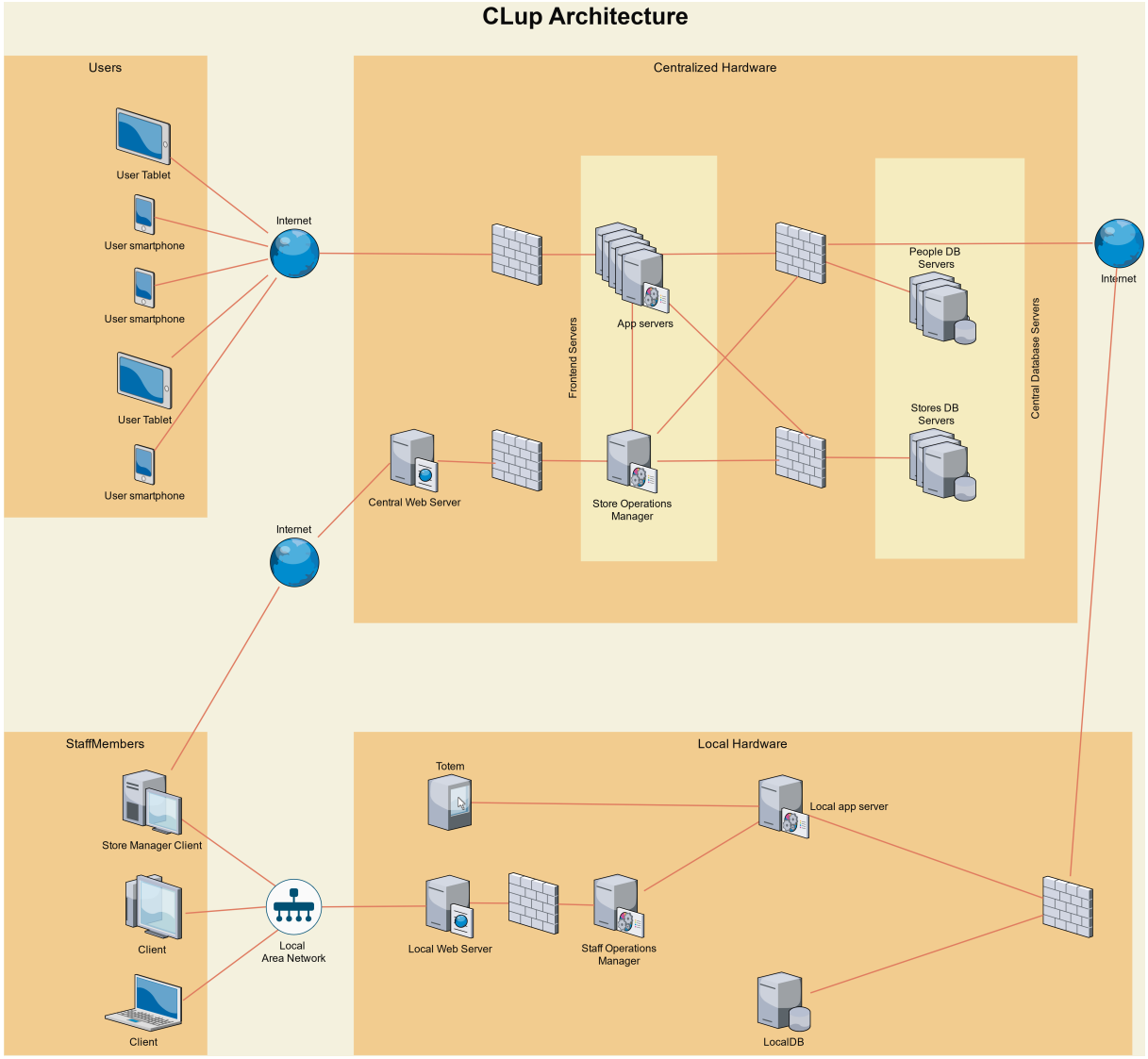


Figure 1: High-level architecture

2.2 Component view

In Figure 2 is represented the component diagram of the internal structure Application layer, showing how its components and subsystems interact. As previously said, the A layer contains Clup's functional business logic which drives the application's core capabilities, it builds a bridge between the presentation layer and the data layer. What follows is a brief description of every component and subsystem:

- **Staff Operation Services**: This subsystem contains 2 components:
  - **Store Availability Management**: This is the component responsible for both interruption and recovery of availability of a store to accept new virtually queued customers
  - **User Reporting**: This component's function is to take note of customers bad behaviours
- **Store Operation Services**: This subsystem contains 2 components:
  - **Staff Members Management**: A subscribed store can grant access and credentials to CLup's store services through this component
  - **Store Management**: A store can enroll to CLup's network through this component

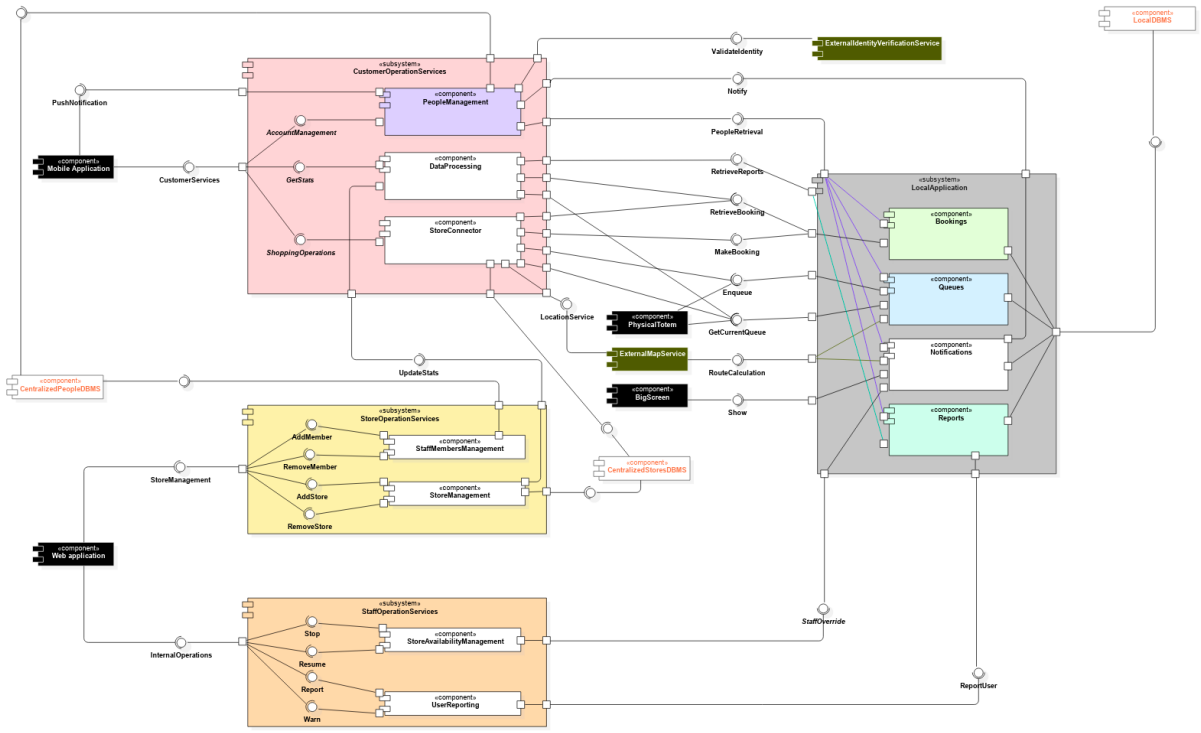


Figure 2: Component Diagram

- **Customer Operation Services**: This subsystem has 2 subcomponents
  - **People Management**: This component is responsible for customers subscription, login t CLup and, thanks to the association device-person, notifications by CLup and Supermarkets
  - **Data Processing**: As says the name, the Data Processing component takes care of data about availability of stores, time slots and queues status. It also evaluate time slots, days and store crowdness level and infer better solutions.
  - **Store Connector**:TODO
- **Local Application** :
  - **Bookings**: This component makes possible booking an entrance, it takes care of the logic of the booking procedure and retrieves information of a previously made reservation
  - **Queues**: This is the component responsible for the basic function of CLup, it has read-write access to stores queues and allows also to block and unblock them
  - **Notifications**: Here is were all kind of users notifications are generated
  - **Reports**: This is the component that instead generate bad behaviours reports
- **External Map Services**: This is the component that provides the map interface
- **Physical Totem**: This component is a simplified version of the mobile app, it has only what is needed to perform the basic function.

2.3 Deployment view

2.4 Runtime view

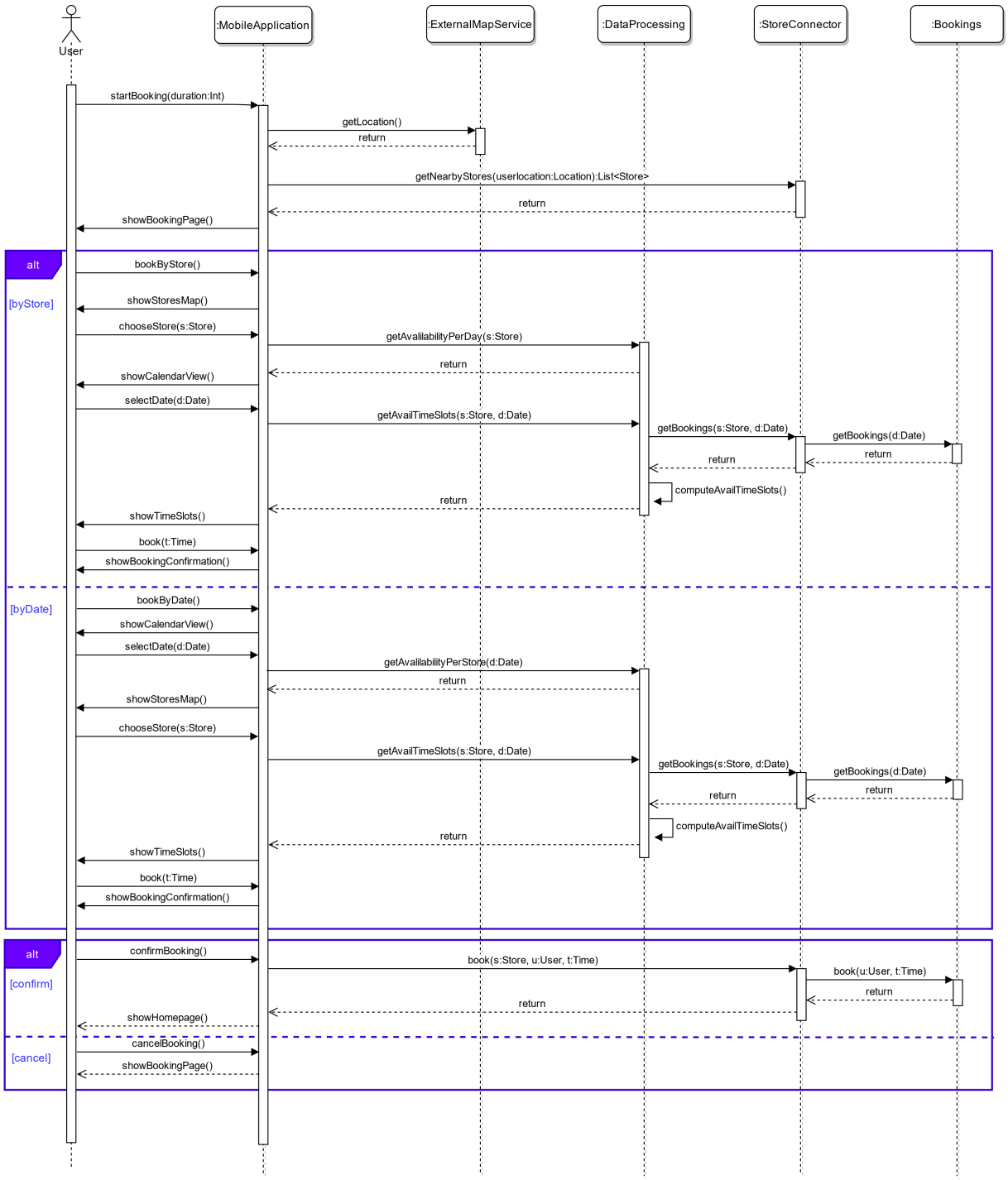


Figure 3: Sequence Diagram: Customer Booking

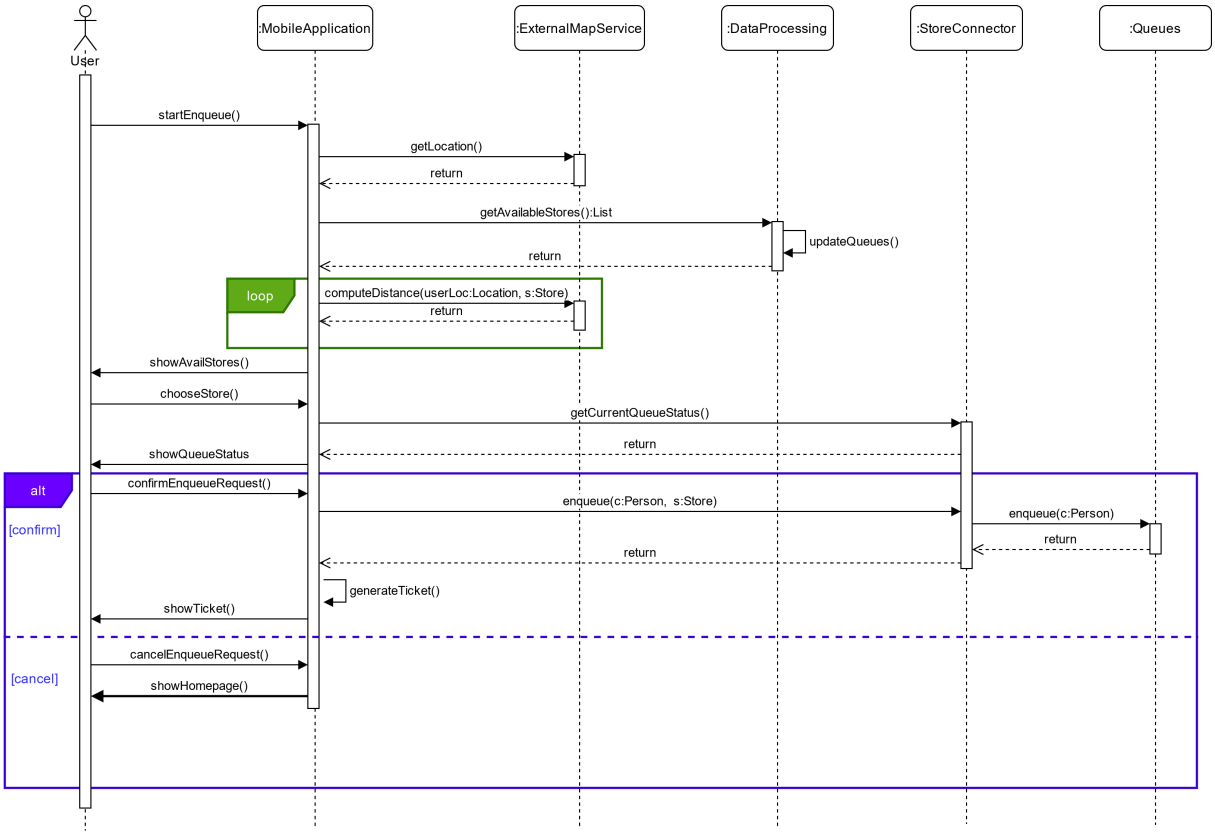


Figure 4: Sequence Diagram: Customer Enqueue

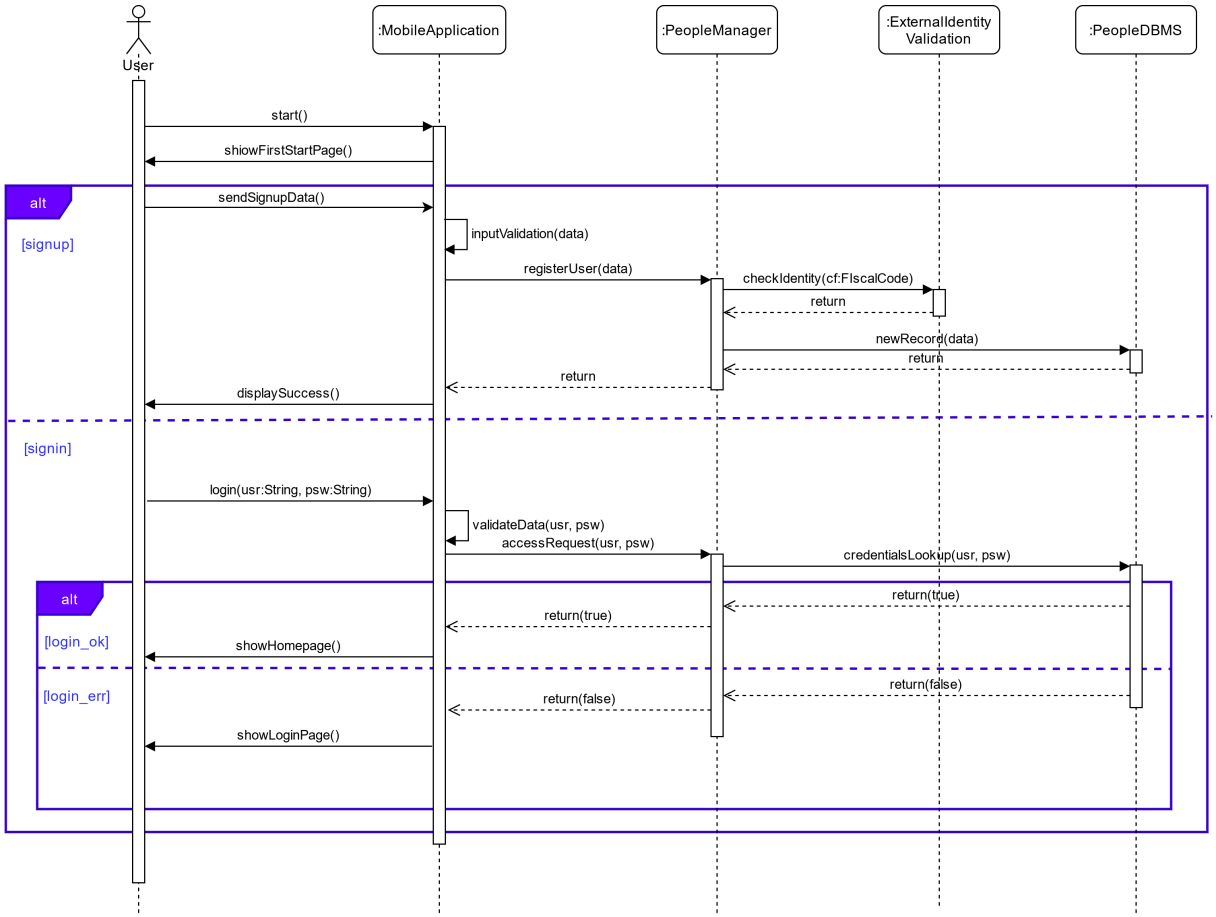


Figure 5: Sequence Diagram: Customer Signup

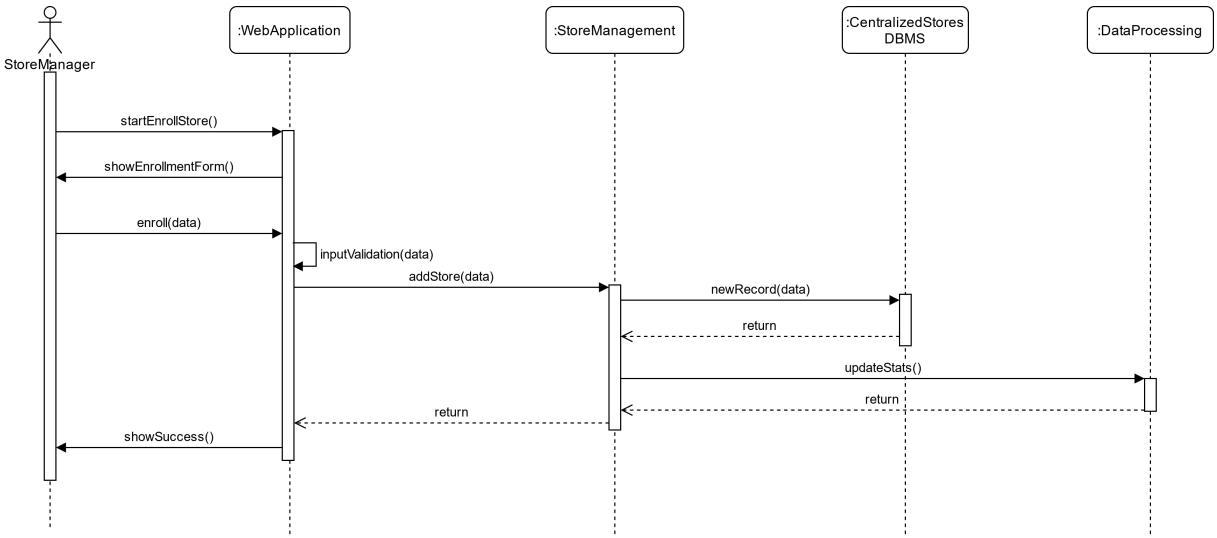


Figure 6: Sequence Diagram: Store Enroll

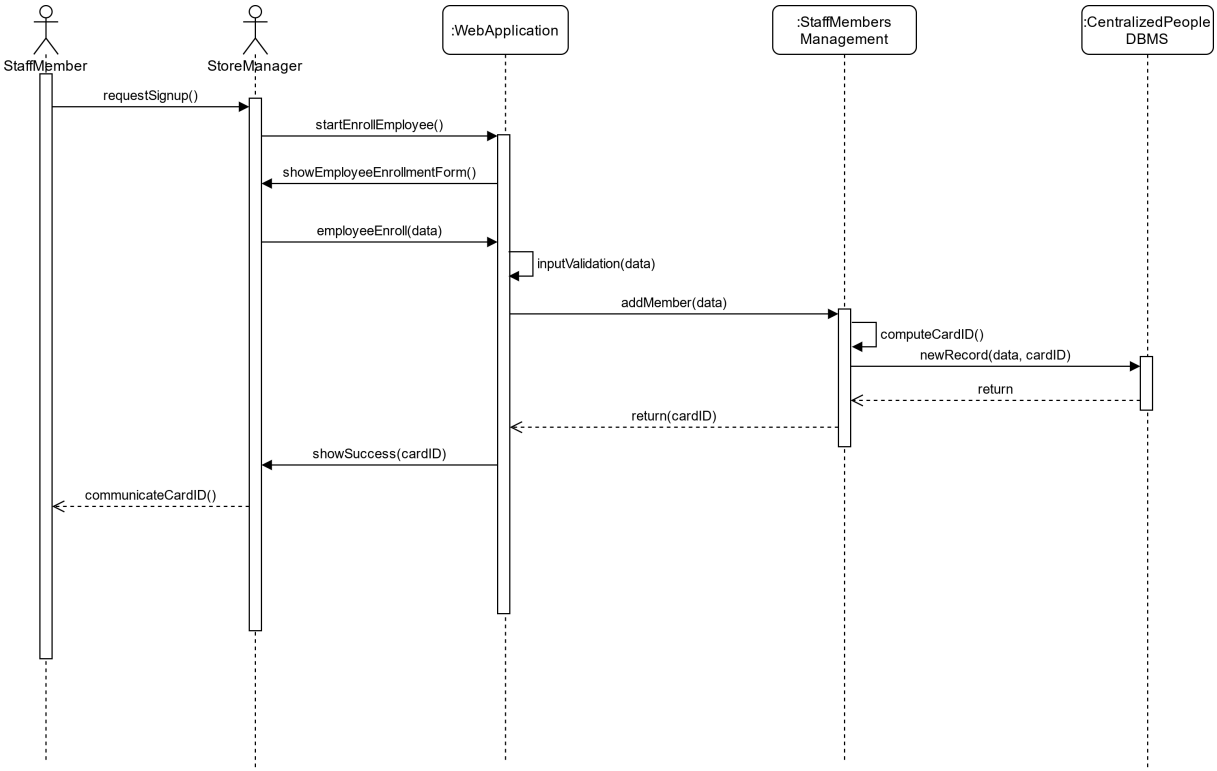


Figure 7: Sequence Diagram: Staff Enroll

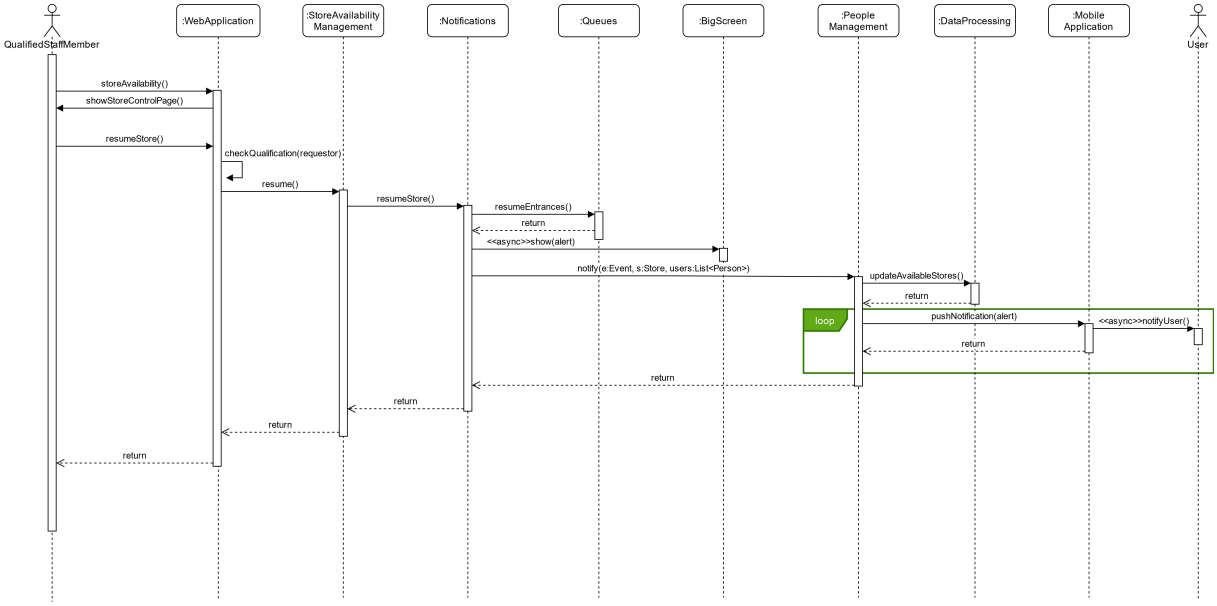


Figure 8: Sequence Diagram: Store Resume

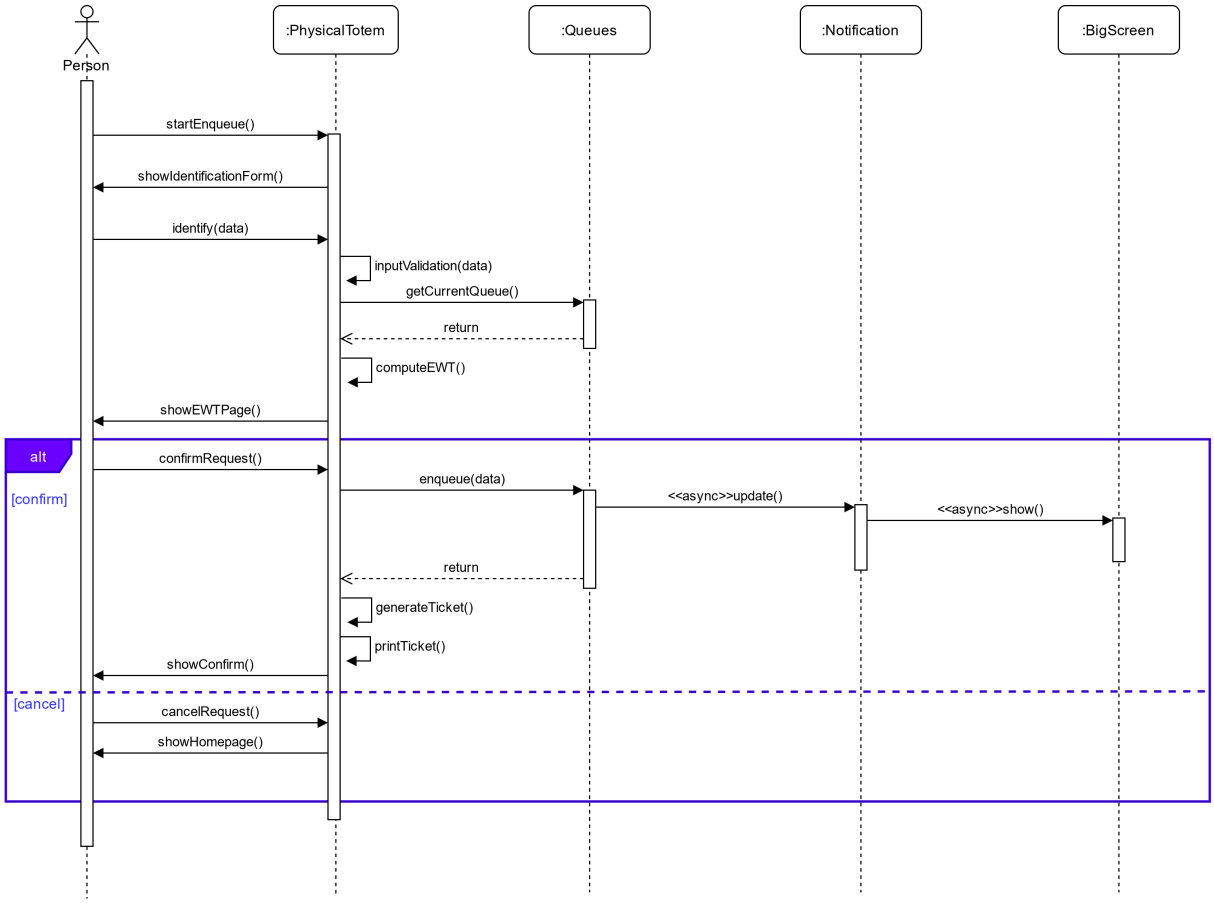


Figure 9: Sequence Diagram: Totem Enqueue

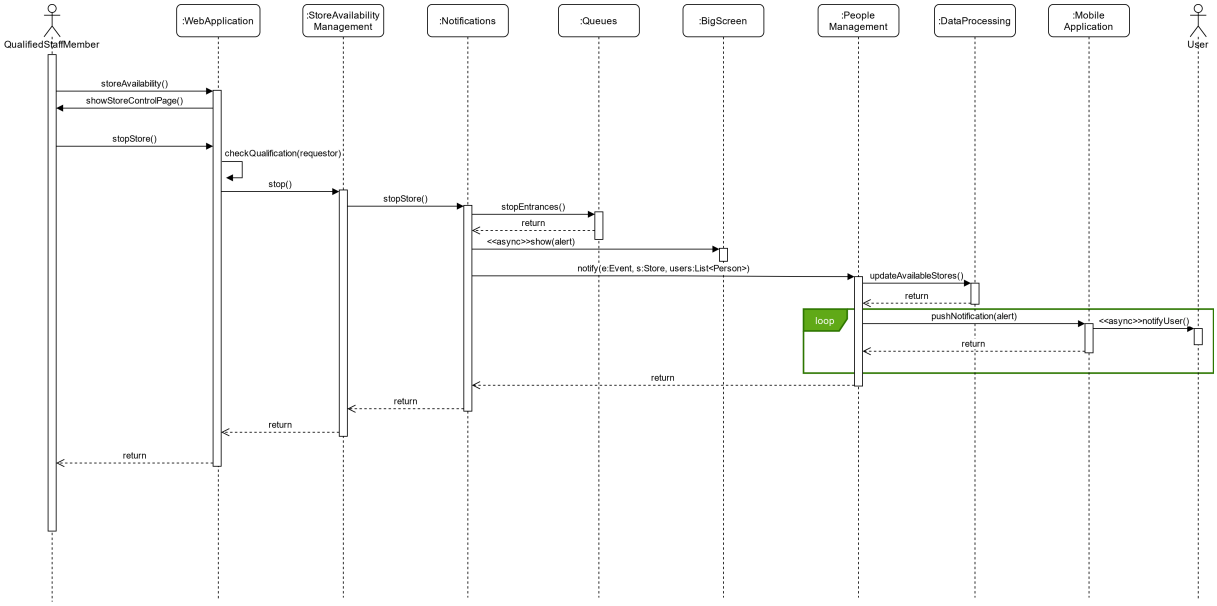


Figure 10: Sequence Diagram: Store Override

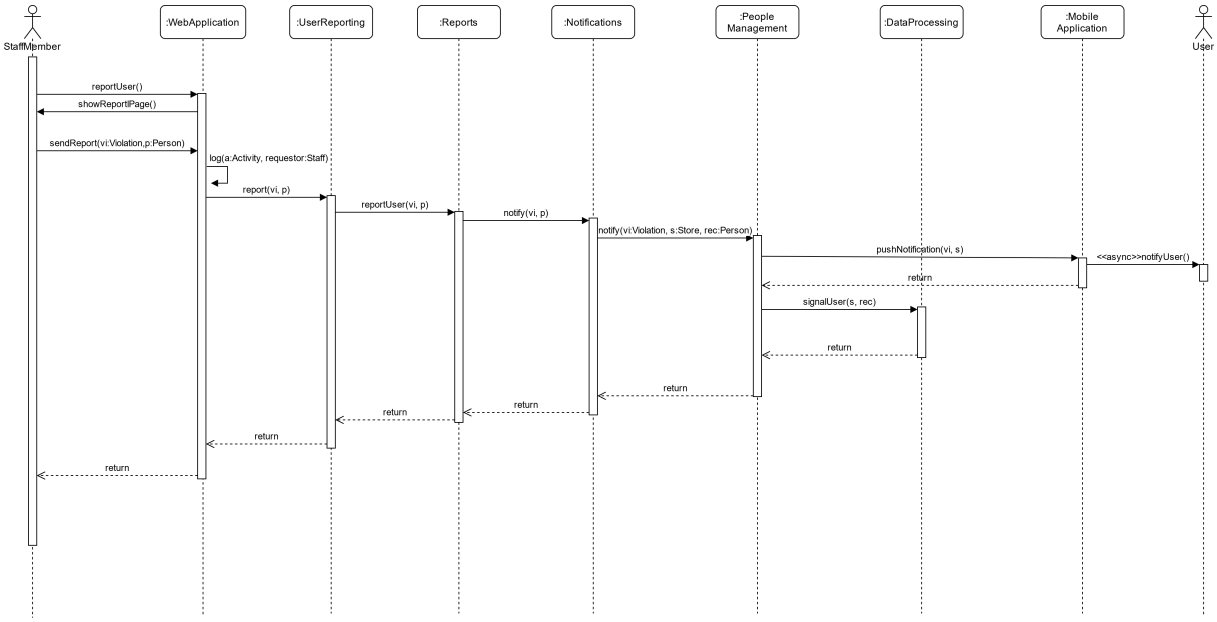


Figure 11: Sequence Diagram: User Report

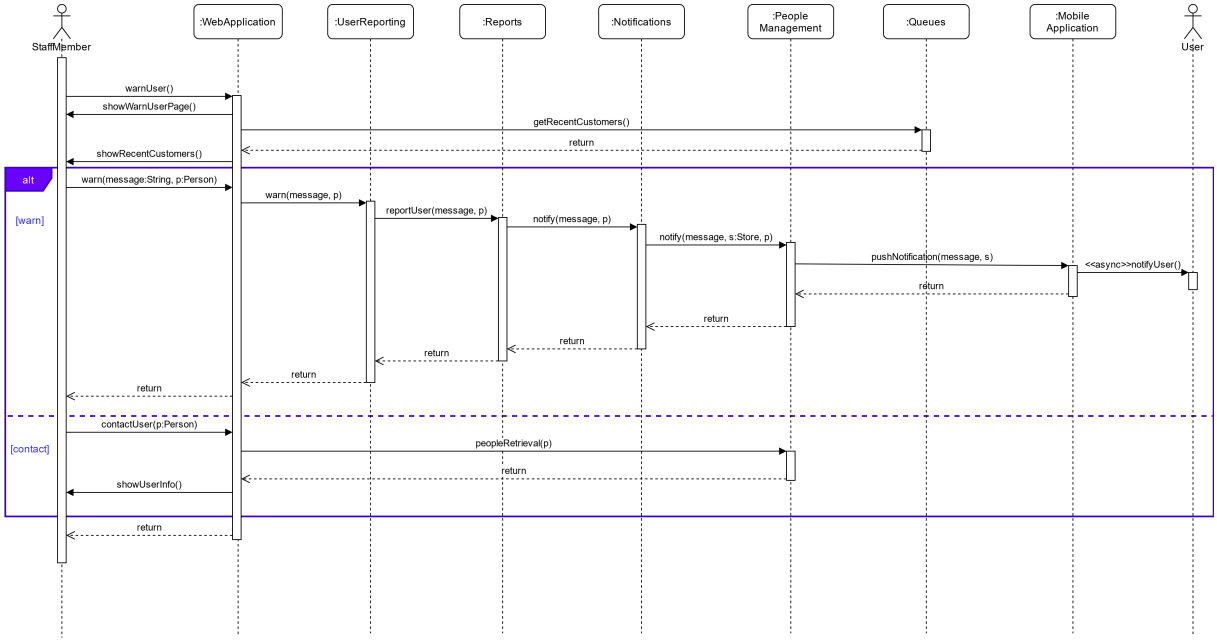


Figure 12: Sequence Diagram: User Warn

2.5 Component interfaces

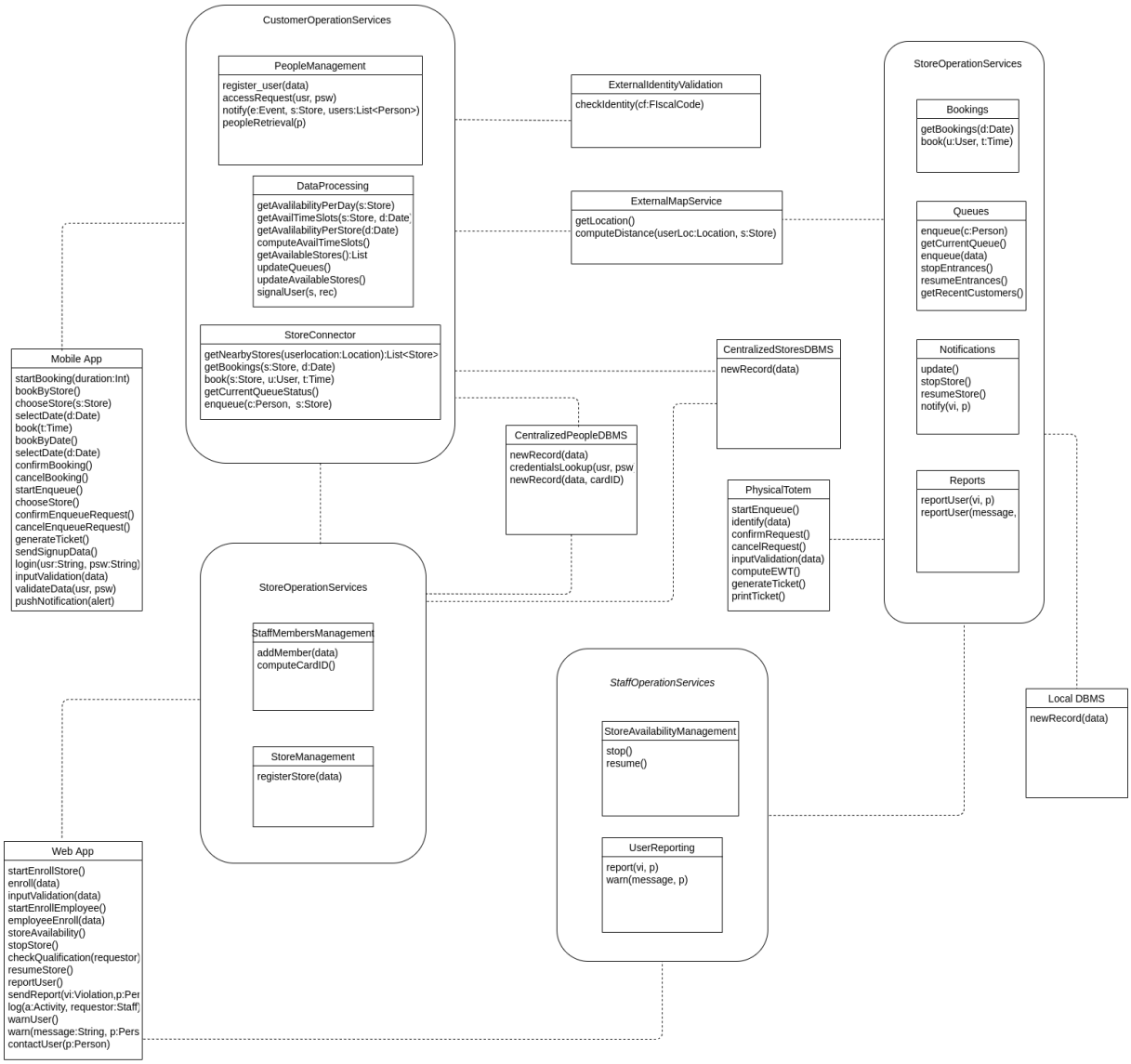


Figure 13: Component Interface Diagram

2.6 Selected architectural styles and patterns

2.7 Other design decisions

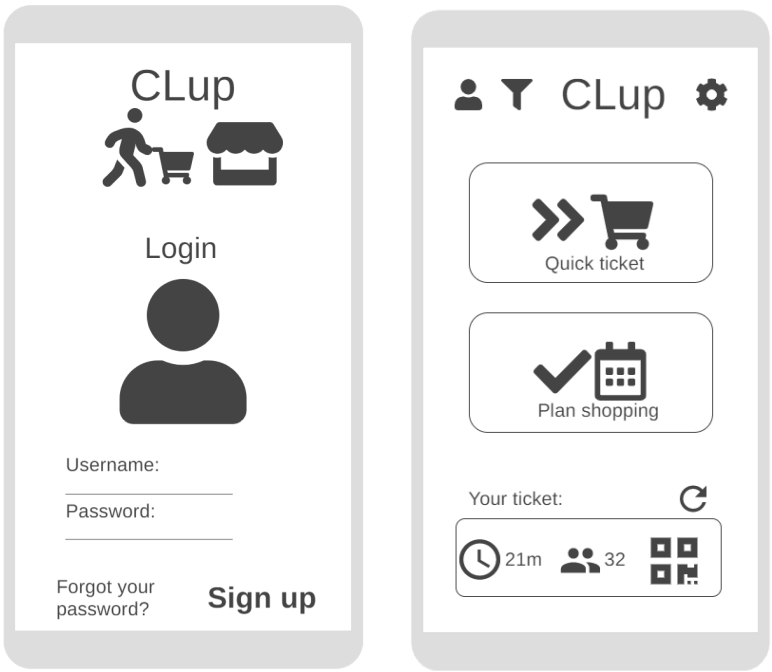
3 User Interface Design

3.1 Overview

CLup is an application aimed at decreasing the probability of contracting COVID-19 (diseases in general) when going shopping to a supermarket. There are two fundamental components: the main one targets customers of supermarkets while the second one is available for store managers.

3.1.1 User Interfaces

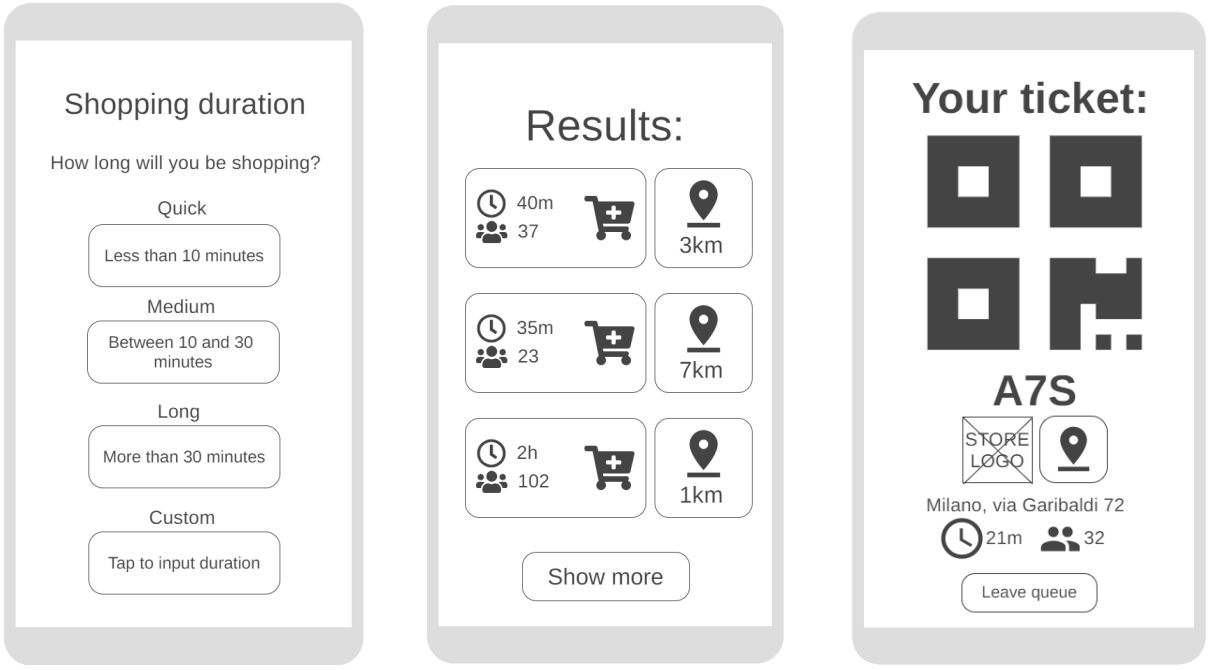
End user functionalities Regarding the first target - customers - they will be required to register to the service the first time they use it by inserting their full name, email address, ID card, phone number and a password. The customer will also be requested to specify his physical address, or to enable the GPS, in order to allow CLup to find stores nearby; this last information can be changed anytime the user needs. If the customer is not willing to register or share his address, the service will not be available.



(a) Login screen (b) Main screen

Figure 14: App startup

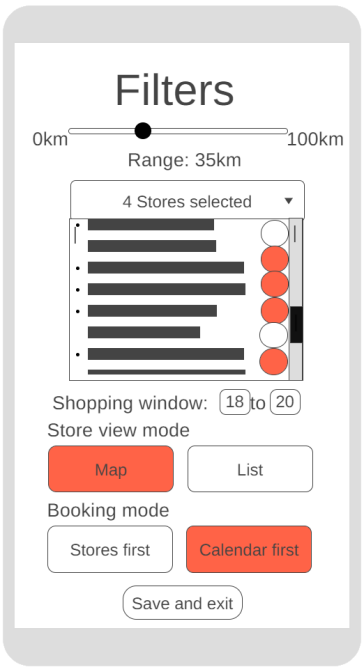
Once the setup is done, customers will be able to access the homepage of the application, where they can tap on the “Quick ticket” button that will allow them to see a list of stores inside a specified range from their current location: for each store a distance in kilometers from the user position will be outlined, as well as the number of people inside the store and its maximum capacity; whenever a store is full, the current number of people in line and an *EWT* are displayed.
It is also possible to visualize stores on a map and, by tapping on one of them, to see the same information displayed in the list. Now, if the user chooses to reserve a spot in the line, the application will open a confirm dialog specifying EWT and the expiration of the ticket. If the user refuses nothing happens, if he accepts instead CLup will process the request, show his ticket and the real time evolution of the line; the ticket is also visible from the home screen.
The tickets consist of a QR code and an easy to remember alphanumeric code alternative to enter the store. At the stores entrances there will also be monitors that show the numbers allowed to enter and,



(a) Shopping time (b) Store list (c) Your ticket

Figure 15: Quick ticket procedure

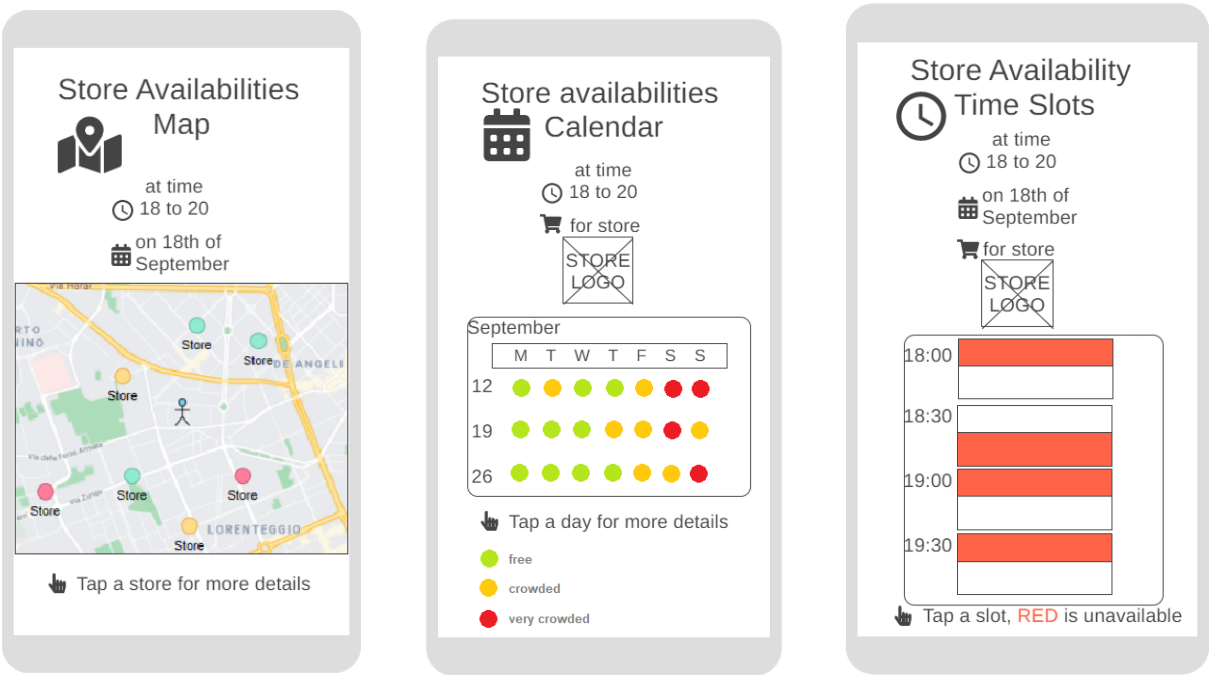
eventually, delays.
The distance range in which CLup will look for supermarkets is specified by the user through the filter button in the homepage, this button will in fact open the filter screen in which, among other parameters, a sliding bar controls the distance and a drop-down list allows the user to filter the chains of supermarkets.
Another important feature is the possibility to book an entrance later in the day or in another day. The



(a) Filters screen

Figure 16: Filters

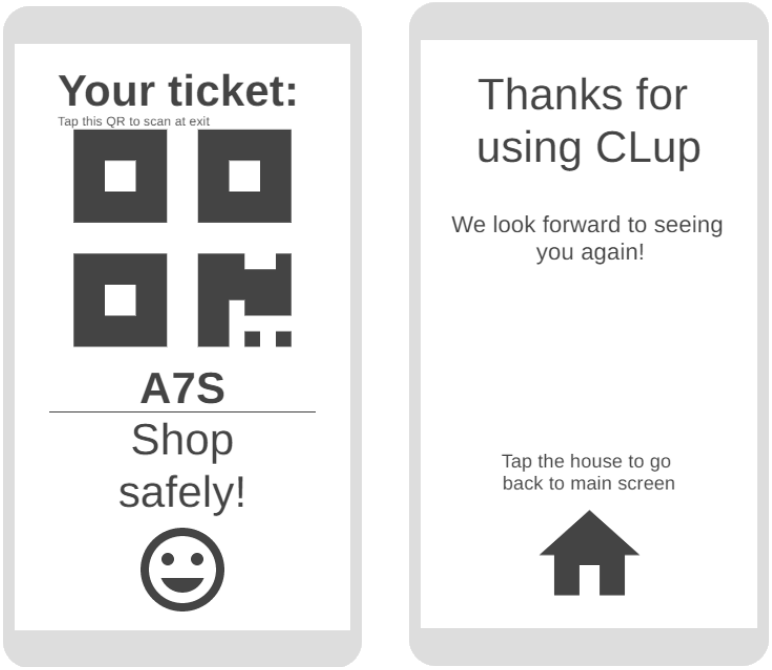
user can specify from the filters whether he prefers to choose the day or the store first and he can set the time range in which he wants to book. There is a dedicated button in the app's main screen that redirects the user to either the list/map of supermarkets or the calendar, and once the user chooses he will be respectively shown the calendar or the list/map, this time with colours to indicate the average crowdedness of stores/days given the set time range. When the user chooses the day and supermarket combination, a timetable spanning the chosen time range is shown, divided in 15 minutes time slots each one marked to show its availability. The user will be able to check his reservation on the home page exactly like previously for the quick ticket, and near the entrance time he will be provided an actual ticket.



(a) Map screen (b) Calendar screen (c) Time slot choice

Figure 17: Booking procedure

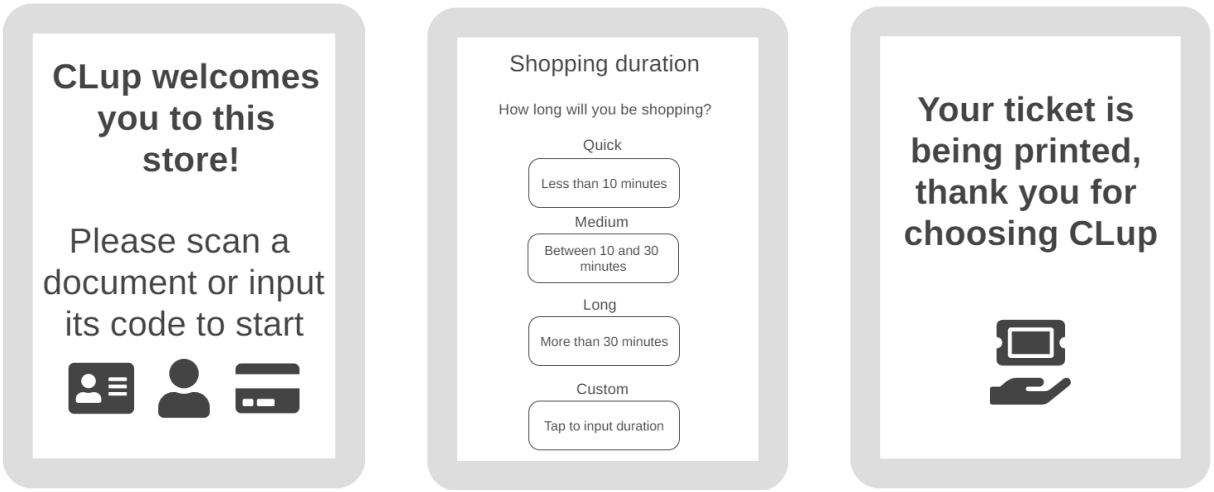
The access at the supermarket is restricted by turnstiles with QR code readers, a staff member is expected to verify that nobody waits his turn in front of the entrance, jumps the turnstile or does anything irresponsible. During the shopping the user will still be able to view his ticket in order to use it to unlock the turnstile upon exiting the store. Customers who, for any reason, do not use the app will still be able to



(a) Shopping screen (b) Exit screen

Figure 18: Shop and exit

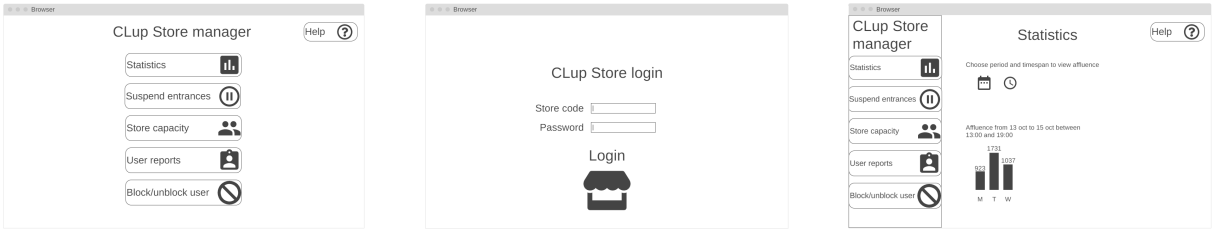
queue in CLup supermarkets by obtaining a printed ticket from a physical totem located near such stores; the functioning of the application will be similar to the “Quick ticket” app function with the difference that the user can only obtain a ticket for the current totem’s store.



(a) Totem welcome screen (b) Totem shopping time input (c) Totem print screen

Figure 19: Totem ticket procedure

Internal use functionalities Another component of CLup targets store managers: when the store decides to join the CLup network, ad-hoc credentials to access the web app will be given. Special staff-only functions will then be achieved by the use of a web application, accessible via internal-use terminals.



(a) Manager login (b) Manager main screen (c) Manager statistics screen

Figure 20: Store manager web app

4 Requirements Traceability

G1	**Anybody is guaranteed possibility to make shopping at any supermarket in reasonable time (def. reasonable)** Requirements: R1, R6, R7, R10, R15 Components: • People Management • Data Processing • Store Connector • Queues • Notifications • Physical Totem

Table 1: G1 Mapping

G2	**Users can get to know the least crowded time slots** Requirements: R2, R11, R12, R18, R21 Components: • Customers • Store Connector • Notification • Data Processing

Table 2: G2 Mapping

G3	**Fair users can make a reservation to enter in a supermarket** Requirements: R9, R13, R15, R21, R23 Components: • Customers • Store Connector • Data Processing • Bookings • User Reporting

Table 3: G3 Mapping

G4	**Stores can easily monitor fluxes** Requirements: R3, R11, R22 Components: • Bookings • Queues • Store Management

Table 4: G4 Mapping

G5	**Only authorized users can access** Requirements: R4, R6, R13, R15, R19 Components: • People Management • Physical Totem • Store Connector • Queues • User Reporting

Table 5: G5 Mapping

G6	**Crowds are dramatically reduced outside supermarket stores** Requirements: R1, R2, R5, R6 Components: • Store Connector • Queues • Bookings • Notifications • Store Management

Table 6: G6 Mapping

G7	**CLup should not decrease customer affluence beyond a reasonable level w.r.t. to normal (define reasonable)** Requirements: R1, R5, R6, R7, R9, R10, R20, R21, R23 Components: • Data Processing • Notifications • Queues • Bookings • Store Management

Table 7: G7 Mapping

G8	**Same shopping capabilities guaranteed to offline users** Requirements: R1, R7, R10 Components: • Physical Totem • Data Processing • Queues

Table 8: G8 Mapping

G9	**Find the best (less crowded, soonest available) alternative among local supermarket stores** Requirements: R3, R11, R12, R20 Components: • Store Connector • Notifications • Data Processing

Table 9: G9 Mapping

G10	**Supermarkets do not overcrowd** Requirements: R2, R3, R4, R11, R12, R20 Components: • People Management • Notifications • Data Processing • Store Availability Management

Table 10: G10 Mapping

R1	Every user can generate a quick ticket for any store
R2	Whenever user initiates a booking procedure, CLup must be able to compute a suggested least crowded time slot based on historical data
R3	CLup must elaborate and upload data about current global customer affluence to the store during use
R4	CLup must admit only valid QR codes for entrance
R5	CLup must allow users to know current queue status
R6	CLup must update user on tickets' validity change
R7	CLup must inform offline users about new tickets (un)availability
R8	CLup must allow users to indicate which product category they are going to purchase while booking
R9	CLup must suggest alternative stores when the combination of selected store|time gives no results
R10	CLup must reserve a non null number of paper tickets at any time for offline customers use
R11	CLup must gather all stores' data about entrance fluxes
R12	CLup is able to cross affluence data of any supermarket
R13	CLup keeps track of people who book an entrance and don't come
R14	CLup allows store managers to stop quick tickets availability
R15	CLup is able to generate QR codes
R16	CLup is able to authenticate users
R17	CLup is able to store users' data
R18	CLup is able to process users' data
R19	CLup makes quick ticket invalid after 15 minutes delay
R20	CLup can use stores' data to sort every store by crowdedness
R21	Users can see available day/time slots of a supermarket through CLup
R22	CLup shows to store managers flux data about their supermarket
R23	CLup must be able to process reservations

Table 11: Requirements list

5 Implementation, Integration and Test Plan

5.1 Overview

Figure 21: Implementation

In this part there will be an explanation on how and in which order the development, integration and testing of this project should be done. The idea is to complete a end user testable version with the bare minimum functionalities as soon as possible so that then it is possible to test this base extensively while adding the rest of the features.

5.2 Implementation Plan

The chosen approach for implementing, testing and integrating the system is a combination of bottom-up and threads given the limited number of complex components that compose it. This way it is in fact possible to firstly develop completely and properly test the core functionality, which is the virtual queuing on the user part, and then add the other functionalities i.e. booking the entrance on a certain time and date (which can partially be seen as an extension of the virtual queuing), ordered stores suggestions given filters and current queues and lastly the store management functions on the store manager's part.
The aim of this approach is to implement, integrate and test an initial version of all the components as soon as possible so that then the missing features can be developed on a functioning base with the minimum need of drivers and stubs and by keeping them simple when they are strictly needed. Thus the development of each component will be incremental once an initial version of the system with basic features from start to end is finished.

The threads approach anyway has to be guided by the bottom-up one because there are some dependencies between the components.

The development has to start with the **Local DBMS** and **Local Application** component (an instance of it will be hosted by every registered store) which is the core of the management of information about queues, bookings, user reports and user notifications, and practically feeds the CLup system and clients with operational information. For now only the queues and notifications subcomponents will be developed according to the threads approach. Also the interface with the physical totems will be developed in order to allow people to queue in the stores even if they don't use the client application, which is part of the basic functionality. At this stage the **external components** will be connected to the system, they will not require any testing on their own since they come from sources that are assumed trustable and reliable.

After this the development of three more components can start: on the store side **Staff Operation Services**, which is needed for the functioning of the web app's control panel available to the store managers, for now only its functionality to allow and stop store entrances will be developed; on CLup's server side the components **Store Operation Services** together with **Centralized Store and People DBMSs** and **Customer Operation Services** can be developed, the former in its entirety while the latter only with the functionalities that can already be integrated and tested according to what was developed in Local Application (e.g. no booking management).

Now it is possible to develop the user interfaces for each module: the **Mobile Application** for the smartphone users, the **Web Application** for the store managers and the **Physical Totem** software for those who don't use the application. For each of these interfaces (for now) it's enough to implement interactions just for those functions of the system that have already been developed.

After these steps there should be a functioning system composed of a mobile application, a program to be installed on store servers, a program to be deployed on physical totems and a system to be used on the main CLup servers to allow interactions between the previous: with these you should be able to accomplish basic tasks objective of the scope of this project such as register users, stores, do logins, get tickets to virtually queue in stores and enter and exit them with such tickets. The testing can then proceed over this basic version of CLup while the development keeps going on by implementing the missing features described in the RASD: the possibility for users to book entrances, algorithms to properly sort and suggest stores and times to go shopping and the remaining functions for the store manager's web app. Component wise the method to use in order to implement these features is the same as the one previously described, and still in a thread manner if needed because of developers' availability. Summing up the implementation order goes in 3 phases with respective components:

1. The stores' local application: Local DBMS and Local Application

2. Central CLup system and local store interaction with this system: Staff Operation Services, Store Operation Services, Customer Operation Services, Centralized People and Centralized Store DBMSs

3. The users and store staff interfaces: Web Application, Mobile Application and Physical Totem

5.3 Integration Strategy

The integration of the components follows their implementation, so that no components will remain isolated at any time during development in order to avoid the risk of inconsistencies between them.

1. In a first moment the Local DBMS and Local Application are integrated, also the two external components are integrated in the system. A stub and a driver are used to simulate the components supposed to interact with the local application, anyway they don't need to be complex at all since all that is being implemented and integrated for now is just the basic functionalities and subcomponents, so Bookings and Reports will be left out for now.

Figure 22: Implementation

2. Now Staff Operation Services is integrated with local application, also Store Operation Services (with its DBMSs) comes into the picture since they both share a Driver for the Web Application. The other part of the central CLup's system can also be integrated since Local Application is ready: Customer Operation Services, there will be a driver for this part representing the mobile app and also a driver for the totem will remain from the previous step.

Figure 23: Implementation

3. Lastly the user interfaces are integrated in order to complete the initial version of the project, now the remaining sub components will integrated following the order in which their respective components were integrated explained in this section.

Figure 24: Implementation

5.4 System Testing

Upon the completion of the basic version of the system, again at the completion of each system feature, and most importantly after the implementation and integration of every feature, the system must be tested on its entirety. The tests are meant to verify the satisfaction of both functional and non functional requirements.

5.5 Additional Specification on Testing

6 Effort Spent

References