Declaration

This is to declare that no part of this report or the project has been previously submitted elsewhere for the fulfillment of any other degree or program. Proper acknowledgment has been provided for any material that has been taken from previously published sources in the bibliography section of this report.

.....
Mohammad Adib Khan
ECE Department
North South University, Bangladesh

.....
Rifat Arefin Badhon
ECE Department
North South University, Bangladesh

.....
Sarwat Islam Dipanzan
ECE Department
North South University, Bangladesh

Approval

The Senior Design Project entitled "NSU Canteen Automation" by Mohammad Adib Khan (ID#1430420042), Rifat Arefin Badhon (ID#1511738042), and Sarwat Islam Dipanzan (ID#1510117042) has been accepted as satisfactory and approved for partial fulfillment of the requirement of BS. in CSE degree program on May 2019.

Supervisor's Signature

Mirza Mohammad Lutfe Elahi
Senior Lecturer

Department of Electrical and Computer Engineering

North South University

Dhaka, Bangladesh.

Department Chair's Signature

Dr. K. M. A. Salam

Professor

Department of Electrical and Computer Engineering

North South University

Dhaka, Bangladesh.

Acknowledgment

We would like to extend our heartfelt gratitude and appreciation to our honorable advisor and course instructor, **Mirza Mohammad Lutfe Elahi**, a senior faculty member at North South University.

We are very lucky to have the opportunity to work under his guidance for our Capstone Design Project. We have received the utmost level of support when it came to guidance along with valuable advice and thoughtful criticism which nurtured the timely completion of our project while maintaining strict standards and regulations.

We would also like to thank North South University Electrical and Computer Science (ECE) Department for providing us the resources and knowledge to let us pursue an undergraduate program in Computer Science and Engineering. We would also like to express our appreciation for giving us the opportunity to showcase our project during the Capstone event which would not have been possible without the support from North South University.

Lastly, we would like to thank all the wonderful faculty members for their immense support and dedication, helping us get a better education and making the journey a memorable and wonderful experience.

Abstract

North South University's (NSU) canteen has seen a major revamp in both appearance and in the way the cafeteria operates. Over the years it has been the go-to place for students and teachers alike for a quick meal or refreshments in between hectic schedules. The recent changes, however, have affected the average waiting times substantially. The new canteen has adopted many new technologies such as electronic displays and Point of Sale (POS) systems for completing payments which is more advanced and offers credibility. But this creates an additional bottleneck in the total time spent on the actual service of "getting the food".

The current mode of operation relies on having to stand in long queues for payments and then taking the payment slip (receipt) to another terminal to receive the food. Since the first queue builds up during peak hours, it creates a backlog of customers waiting in frustration.

This is not only inefficient but time-consuming as well. This has affected a large number of daily customers for the cafeteria as time is scarce in the daily lives of academics, students and teachers alike. The solution we have devised is to offload the manual payment process to an online payment system using a payment processor hosted on a mobile device using a web application and Android.

This will enable users to order and pay inside the canteen without having to resort to long queues. Users can use their bank accounts/cards for transferring funds to the application (canteen - account) which can be used for purchasing food inside the cafeteria. This will avoid the need for cash exchange at the counters, saving time and resources, reducing the need for additional queues.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

OVERVIEW

1.1 Introduction

The recent advent of technology has made it possible for the general masses to own a smart mobile device with relative ease and low cost. The Android platform has made significant advancements in both ease of use and usability and acts as an excellent pocket companion. The services provided through these mobile platforms are very rich and offer a diverse amount of options to the user base.

This serves as an excellent gateway for proposing the automation of payments inside the cafeteria using a mobile application preferably under the Android platform. This is achieved by the means of converting user credits into an electronic balance system which is maintained inside the application. The concept can be further applied to other intra-university services such as the photocopy center and the bookshops in order to reduce cash transactions.

The Android platform is relatively cheaper and is generally the more chosen platform, hence the application built for this project is using Android and Web Technologies at its core. The underlying principle of this project is to utilize a smartphone by using an online system which will replace the current model of payment interaction. The incentive is to provide the user with an intuitive user interface to order food from their smartphones and handle payments directly through the device. This will help eliminate payment queues and offload service onto food serving terminals. The user application will provide a menu for the cafeteria which is updated daily and track order status until the food is received which is confirmed after verification through QR code service using the device.

The intended effect is to reduce the load on the employees working inside the cafeteria while improving service and catering more efficiently during peak hours. The advantage of automating such a system also has an opportunity to track customer behaviors and food choices making way for personalized recommendations and promotions.

1.2 Problem Statement

The current mode of operation inside the canteen has inconvenienced a lot of students due to insufferably long queues. The number of customers during peak hours gets unmanageable with the existing number of cash terminals. This has been rather an unpleasant situation for the students as they have to stand in queues for a long period of time just to get the cash across and then wait for a second time to receive the food.

As such, students are facing the dilemma of ordering food inside the cafeteria between tight class schedules and mostly go without food in between class breaks. Even though the new canteen is supposed to be more efficient and better in terms of operations, it is still inferior when it comes to saving time.

The main problem is primarily caused because of the limited number of terminals dedicated to payments whilst the remaining cater to serving food. Since there are no electronic menus, the customer has to inquire about the food items and their respective prices and availability face to face with the employee. This kills valuable time and wastes effort in getting to the end goal of ordering food efficiently.

1.3 Alternate Solutions

- **Bulk Cash Deposit**
- **RFID Coupled Electronic Accounts**
- **Scratch Prepaid Cards**

Bulk Cash Deposit - The provision for re-charging or topping up a student account happens at the start of the semester when the student is paying their tuition fees. This enables the said account to be used inside the cafeteria to purchase food without having to carry cash. The account is valid until the credit balance is zero or has reached the expiration date, After which the student/account holder must recharge in bulk again. The recharging is done through affiliated banks which are endorsed by the canteen authority or NSU administration. This solves one of the problems of cash which is susceptible to getting lost or inconvenience of carrying.

RFID Coupled Electronic Accounts - Upgrading the student/faculty ID cards inside the university to hold more information such as points or credits which can be used to avail goods and services inside the campus premise. The RFID card is then able to act both as an identification card and also as a payment gateway inside the university freeing the need to carry cash. The card inherently becomes more useful as it has multipurpose use. The problem, however, is that there is a tedious process to writing additional information to the card. The current infrastructure of holding just an identification has to be changed which can be costly and time-consuming.

Scratch Prepaid Card - The shops which sell goods and services such as the canteen, bookshops and the photocopy center allows the use of payments through credits topped up by scratch cards. This procedure works the same way how mobile credits are redeemed through recharge vouchers. This, in theory, works very well when used for certain services which are not that active such as buying books which happens seldom. The services which require very fast transactions and where credibility is very important is not feasible to such a solution.

1.4 Brief Overview

Online Payment Processor using a Mobile Device:

- **Graphical User Interface**
- **Online Push Notifications**
- **Electronic Balance**
- **Payment Processor**
- **Food Expenditure Tracking**

The use of an online payment processor using a mobile device through a dedicated application centered to the canteen having the major components listed above. The application lets the customer choose food through the online menu and the respective prices are shown without having to resort to traditional menus or face to face conversation.

The payment process is taken care of by an external payment processor, this is deliberately done so that payments are guaranteed through a secured channel using sophisticated technologies offered by the payment processor. This is also done so that there is no reinventing the wheel when it comes to security as this is a very delicate subject when it comes to money transactions.

The application serves mainly as a front-end which has the job of displaying food items in a friendly graphical user interface. This makes it easier for the customer to choose their desired items without much hassle. Another benefit of having such a system over classical methods is that the canteen does not have to waste resources in updating prices since there is no involvement of hard-coded menus.

1.5 Summary

The chapter describes the implication of implementing such a service and a brief overview of the features that will be in the end product. The end goal is free the load on employees at the payment terminals to have an easier time processing payments. The net effect is such that more employees will be able to serve food through the terminals instead of having to process payments manually.

CHAPTER 2

Literature Review

2.1 Introduction

This chapter discusses solutions that can be applied to restaurants and cafeteria services from various papers. The proposed solutions by the respective authors are discussed in detail and how they might be applicable to the project. Each solution is documented in a series of summaries and a discussion about their findings.

2.2 Related Paper Summaries

In paper [1] the idea was to provide a more efficient solution to the canteen system. Using standard simulation steps several services were modeled and evaluated based on customer system time. After a few simulations, these students came to an understanding that the restaurant can become more efficient with five servers. A five-person setup with three cashiers, a soup server and a sandwich server could reduce customer system time by over two minutes per customer. Conducting other experiments they came to the conclusion that adding a third cash register was the optimal solution and it reduces the customer time considerably.

Paper [2] talks about the concerns regarding the speed of transactions and the security involved without complicating the process of reading which is undesirable for the users. This project uses the information of the bank and implements the public and private key generation method for their security measures. Use of NFC communication lets the customers get item price and merchant details. Once the merchant and the customers get each other's details through the NFC a transaction is made. To verify the transactions a QR scanner is used. This process made the transactions faster and more secure at the same time.

In paper [3], a fully automated ordering system was proposed where paper-based ordering would be replaced by a fully centralized touch screen ordering system. The system consisted of microcontrollers which interfaced with input and outputs. At each table, an LCD panel was integrated which displayed the food items and their respective prices. The process of ordering was that the customers would select the items from the touch screen and confirm the food

they want to order. At the receiver's end, the employee would be able to see which table has ordered which food items and can easily serve the food without having to worry about taking wrong orders.

In paper [4] a simulation program called WITNESS 2008 was used to figure out the best possible way to reduce waiting time to serve food. The whole objective was to reduce the waiting time which was the net time between client arrival and client leave. There are four types of queue: SQSS, SQMQ, MQSS, MQMS. The best results were obtained using MQMS because it reduces the queue length significantly which plays a major role (influence factor: 20%) in the overall system.

In paper [5] their system is designed with a more convenient and viable payment solutions-QR code-based payment scheme of 5 sale terminals, its payment supports mobile users and mobile terminals to trade at any time and anywhere. Mobile payment is a very important and critical solution to the mobile business industry. There exist many mobile based solutions such as NFC and RFID which offer the same solution as the QR code scanning but what makes the QR more efficient is that it requires less update, unlike the other solutions. The main objective and organization of their paper are as follows- the system frame, security solution, system implementation, and the conclusion.

In paper [6], an experiment was done in an Iranian restaurant in order to find how to affect consumers' food ordering behavior. According to the study findings, customers can change their behavior under the influence of menu design. In other words, when a customer sees the calorie content of the food on the menu, they may experience obsession and order fewer high-calorie foods. This also opens the path for higher authorities to control obesity among the population by making the consumer buy healthy food by themselves without setting up direct restrictions on dietary on the whole population by using authoritarian powers.

The proposed solution in the review paper [7] makes use of modern technologies like Wireless Local Area Network (WLAN) and Radio-Frequency Identification (RFID) as a viable approach to replace the traditional way of managing a restaurant. The rationale behind using RFID and WLAN is to enable the restaurant workforce to reduce the time needed to

process an order thus automating in the sense and having a more customer appeal. The combination of these two technologies facilitates the following activities: tracking expenditure habits, wireless, RFID payment, promotions. The authors imply that installing such a system could reduce costs and human errors when operating a restaurant to bring about profits and a smoother dining experience.

Another different approach in paper [8] was to implement an Android solution as a platform to replace waiters. Each table inside the restaurant is fitted with an Android-based device which acts both as an electronic menu in place of the menu cards. This has the added advantage that the manager of the restaurant can oversee all the orders placed from each individual table real time without having to wait for waiters during rush hours. A bonus point is that each table can host updated menus which would have been very costly to do for minute changes with hardcover menus.

The main idea of the paper [9] revolves around maintaining security when it comes to mobile transactions in the real world. The authors discuss the implications of using mobile technology to facilitate payments and stresses the fact that they are unreliable. Traditional systems have the risk of data being stolen by perpetrators. The authors choose PKI using QR Code, which enables: identification, non-repudiation, integrity, confidentiality. To enable this experiment, the algorithm used was the famous RSA.

2.3 Problems/Similarities of Approaches

The summaries discussed from these papers gave us important distinctions to approaches taken by the respected authors. It is important to explore every possible angle when attacking a problem and devising its solution. We took ideas from these papers and implemented functionality to which we found best for our needs.

A manual approach of changing the number of employees (cashiers and servers) to see whether a certain combination would favor the other. [1] This could only be tested in theory since it's not possible to reproduce such a setting in actual testing.

The use of a QR scanner for completing transactions is one of the core components in our project. [2] The idea stems from the fact that a certain cryptographic setup is needed to ensure security when completing payments. The idea of using a QR scanner using the smartphone's camera and an application which can detect QR is a cost-effective approach used in our solution.

The use of a touch screen based ordering system inside a restaurant is a probable solution. [3] We took that idea further and used it to develop web-centric applications that could be run on the user's own smartphone without having to install additional hardware inside the canteen.

Using simulation software tests were conducted to see which setting would net the fastest service inside a cafeteria. [4] This too could only be tested in theory as it's not possible to conduct such a test in an actual setting.

The use of NFC and RFID for payments versus QR is the seamless use of the latter when considering efficiency and cost. [5] Our solution to these problems heavily relies on open source QR technologies so that it is robust and economically viable.

An electronic menu of foods along with their calorie content based on the amount. This has the net effect of users getting to know the caloric quantity and how they might affect food

choices. [6] The application we have developed uses this concept and has a calorie tracker in order to track user's food habits and alarm them if exceeding a certain amount.

Using RFID based technology to hold user credentials and communication through WLAN networks. [7] Our solution relies on WiFi networks or cellular internet applications to hold data, similar concepts but different design.

Using GSM network technologies and Bluetooth for communication inside a restaurant or cafeteria premise for ordering. [8] The solutions that are devised in our application uses basic Internet connectivity for all communications which are more robust.

The use of Public Key Cryptography in securing payments through QR. [9] This concept has been heavily utilized in our application for completing transactions and generating unique QR strings.

2.4 Summary

Taking into consideration all the alternate approaches and being aware of their shortcomings. Our solution makes use of the available technologies that are optimal and cost-effective. The proposed solution that we have devised uses many of the concepts described in the papers discussed. The use of an external payment processor makes our solution unique to the approaches discussed and also solves the shortcomings associated with them. The literature review has been very helpful in identifying the inherent problems in designing such a system and gave an idea of how such systems work in an actual real-world setting.

CHAPTER 3

DESIGN APPROACH

3.1 Introduction

With all the ideas in place, this chapter will outline the blueprint of the project followed throughout its life cycle. It will introduce the notion that we followed in identifying all the required components was needed to make a complete solution. Our main line of thought was to make sure the application is useful to the end user and secure in operation. The series of steps taken to ensure that notion is detailed below using figures and diagrams where necessary along with proper explanations.

3.2 Identifying User Requirements

The main criteria for user requirements for such a system boils down to ease of use, intuitiveness and ultimately security regarding transactions.

The main requirements as seen from a user perspective are as follows:

- Intuitive Graphical Interface
- Ease of Use
- Secure Transactions

The requirements from an employee/admin side perspective are as follows:

- Easy and Efficient Operation
- Transaction Logs
- Electronic Menus
- Real-time Push Notifications
- Complaints System

3.3 Design Methodology

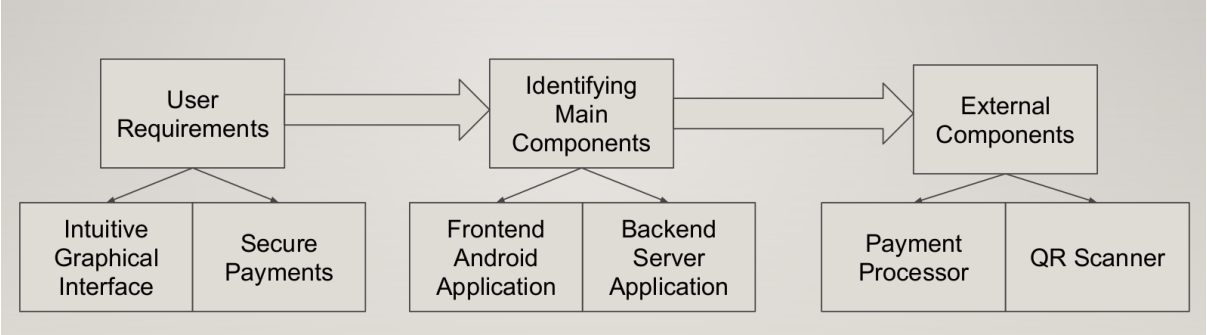


Figure 1.1: Design Methodology

A series of well thought out plans led us into viable approaches and solutions to problems that are currently affecting the canteen system. After going through the feasible solutions to the problem stated, we chose an optimal solution that would best tackle the problem. Analyzing the system design we came up with a design methodology that would alleviate the intense hard work and at the same time would provide an optimal solution. We divided and constructed a strategy to be followed throughout the project session so that the development is modularised. We plan to spend a different amount of time at each separate segment as required and at the end integrate them together to make a final working system.

The end goal was to devise a combination of the requirements listed below and tackle each problem one by one:

1. Identify User Requirements:
    - Intuitive Graphical Interface.
    - Ease of use, friendly.
    - Secure Payments.
    - Accountability/credibility.

2. Identify Main Components:
    - Hardware - Frontend devices, servers, internet connectivity.
    - Software - Web applications, native Android application.

3. Identify External Components:
    - Payment processor - Subscribing to an external service.
    - QR - Displays/monitors.

4. Integration:
    - Developing identified components.
    - Integrate components for a working prototype.
    - Testing - individual testing of components.
    - Adding real-world constraints.
    - A test system as a cohesive whole against constraints.

5. Deployment:
    - Monitor critical features for performance.
    - Debug if systems show unintended results.

3.4 Estimated Cost

Parts/Software	Cost (BDT)
PC	~10000
Monitor/Display Unit	~5000
Peripherals	~1000
Hosting/Domain	~1000
SSL Certificate	~4000
Total Cost	~21000

Figure: 1.2 Estimated Cost of Installation

The estimated cost of system installation is depicted in the table above. The amount calculated might vary depending on system configuration and units purchased.

3.5 Summary

This chapter outlines the design strategy that we followed throughout the course of the project. We allowed for certain changes along the path to account for actual setbacks during the development phase but the entire project has closely followed the design guidelines that we came up with during the initial planning stage. A brief idea about all the requirements needed helped to identify major components to facilitate the project. Careful planning led to the timely premise of a working prototype which will be discussed in the coming chapters.

CHAPTER 4

SYSTEM DESIGN

4.1 Introduction

In this chapter, we will outline a detailed overview of all the components talked in the previous chapters. The figures will be guided with explanations to help conceptualize the idea to the reader so that they get an understanding to form a mental picture of the project's framework and how individual components are tied together. This will help to realize the certain choices and decisions taken during the course of development and how to project that idea into a real-world application.

4.2 System Architecture

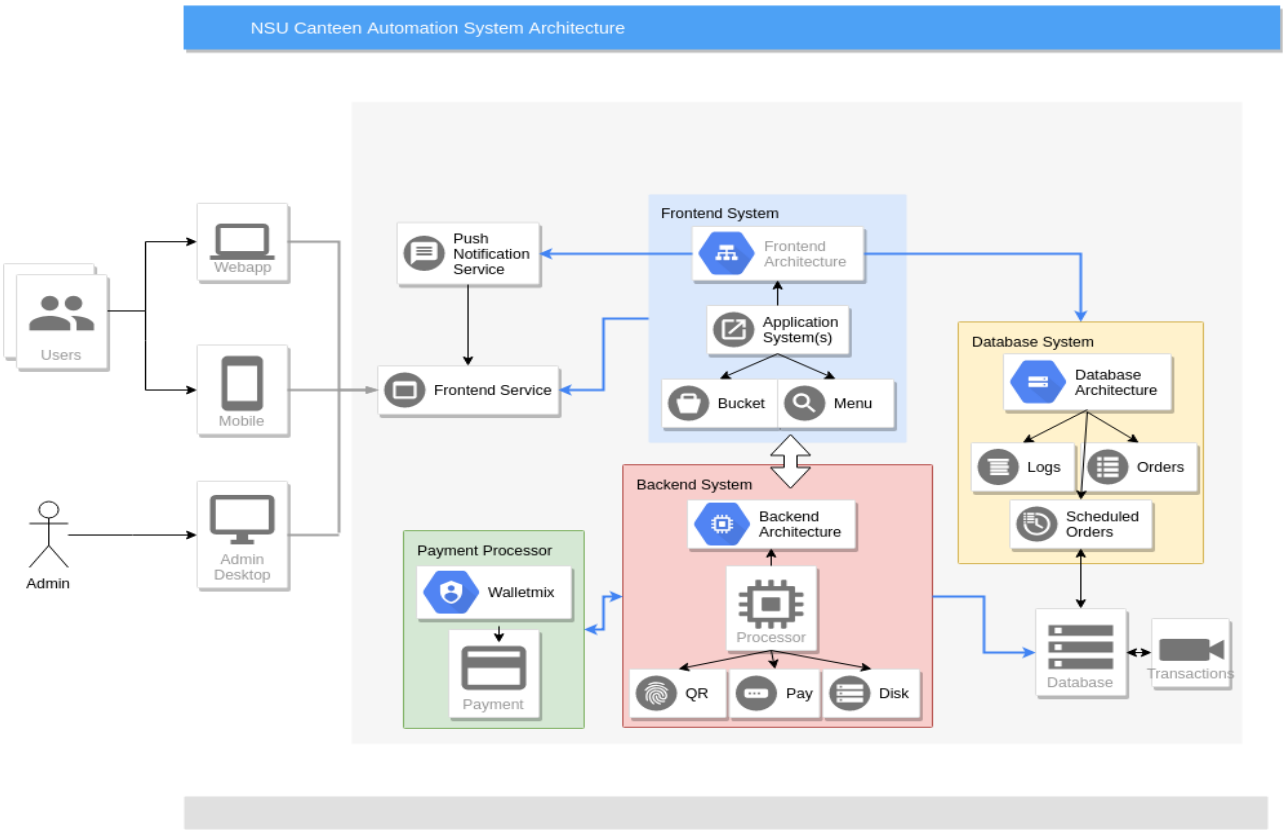


Figure 1.3: System Architecture

The above figure shows the main system architecture which depicts all the components that are connected to one another to make a cohesive whole for the application being designed. The architecture diagram lists all internal and external components that were not developed by ourselves but is otherwise used by the application. There are also instances of users albeit using icons to demonstrate the interaction between the systems. A more detailed diagram concerning use cases will be given the in following chapters.

The system architecture employs a basic integration of a frontend system along with a backend system and central storage. The payment mechanism is handled by an external processor from a company named Walletmix. A detailed chapter will explain the inner workings of the external payment processor along with documentation.

The system architecture involves components that are modularized so that they can be tested individually. This is done in purpose so that development is easier and said components are fully ensured to work before integration. The loose coupling relation between the components will ensure interchangeability and substitutions if necessary. Certain external Application Programming Interface (APIs) were also used for services and are treated as an external entity by the application.

This is done so that the critical components can be separately designed which will favor easy integration with external services and pave way for future upgrades without affecting other parts of the application once deployed. Notable mentions, in this case, are the push notification services module and the QR camera module installed through external APIs.

A basic rundown of the schematics used in the figure: 1.2 is given, along with their detailed explanations in the upcoming parts of this chapter:

- Users - Students/faculties/university staff (employees).
- Admin - Administrator enabled accounts for managing backend servers.
- Web app - Web application running inside a browser (mobile/PC).
- Mobile - Android.
- Admin Desktop - Interface for handling the backend server (Web application).

4.3 Frontend System

The frontend design of the application for the web app is done using standard Hypertext Markup Language (HTML), Cascading Style Sheets (CSS3), and Javascript (JS). Since the client side usage will be limited to handheld mobile displays, a framework called Bootstrap is used to make sure the interface is responsive and mobile-ready. Bootstrap utilizes its own combination of the languages mentioned above along with precompiled style settings for various mobile screens of different shapes and sizes.

This has the added benefit of not having to write everything from scratch whilst ensuring compatibility with different manufacturer devices and configurations. It also serves the purpose of having a good looking frontend design which is ready and working and gives a better-looking end product compared to writing the code ourselves completely. Although certain elements were tweaked to the needs of the application where necessary. Even though it is Bootstrap underneath, the framework was heavily modified to suit the design criteria in the context of the application, i.e being a food serving application.

For the Android application, in order to minimize having different looking interfaces and compatibility, the application was ported to the native platform using Android's WebView interface. This allows users who want to install the application without having to resort to browsers experiencing the same user interface (UI) as the non-native web application. This was done to incorporate the need to maintain a similar but already working UI provided through Bootstrap's framework along with the heavy customizations built for our needs.

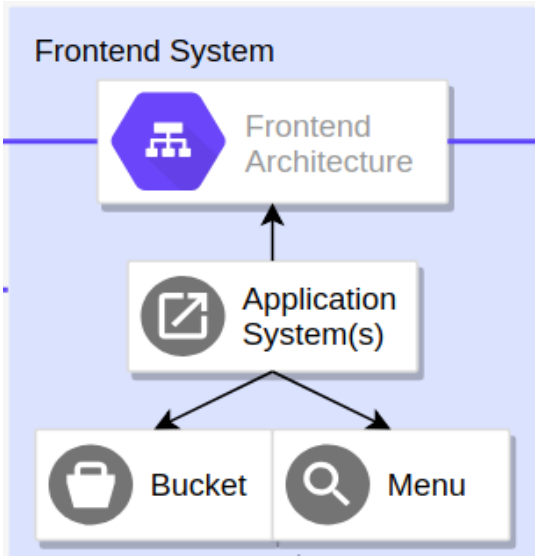


Figure 1.4: Frontend System

The frontend system depicted in Blue houses the following components:

- Application System(s):
  - ➢ Bucket - Storing/saving temporary sessions/carts.
  - ➢ Menu - Daily menus (items/prices) from the server.
  - ➢ Displaying Electronic Balance.
- Push Notification Service:
  - ➢ For sending instant notifications through Android.
  - ➢ Web application notifications through JavaScript.

The frontend system is heavily favored towards mobile users as the main use case of this application will be to order inside the cafeteria using a mobile device. The graphical user interface (GUI) will, therefore, be designed primarily for a companion web application for users who want to use the system from their mobile/PC browsers along with an Android application. Since both of these systems are powered through the web application and thus use the same technologies, the components mentioned above are synergistic.

The “Bucket” component is using built-in Javascript instances to store carts and receipt sessions while a customer is using the application through their mobile phone. The menu system is designed to be easy using buttons and on the fly, options to add/remove items to the

cart. The “Menu” component is being generated from the backend using an API to fetch data from the central database that the application is using underneath for storage purposes.

The Push Notification Service will be discussed in a later chapter for external components. For a brief introduction, the Push Notifications is using an external service named “One Signal” which can be implemented inside a web application that is enforced through Hypertext Transfer Protocol of Secure Socket Layer (HTTPs).

The notifications system for both the web application and Android is used for instant notifying the users of any updates regarding menu changes or prices. It is also used heavily for relaying information about current orders that are active or in progress. This has the added benefit of acting as a subtle reminder to the user without having the hindrance of running the application in the foreground all the time.

4.4 Backend System

The backend system of the application is done using a backend server-side scripting language called PHP (Hypertext Preprocessor). During the initial design phase, the choice of PHP seemed the most suitable for developing such a system with all the components mentioned. It also aided the fact that most external services that we sought for such the notification system and QR Camera service were supported by PHP. This compatibility ensured that integration of external services was easy and seamless and having familiarity with the language itself it was a no brainer choice.

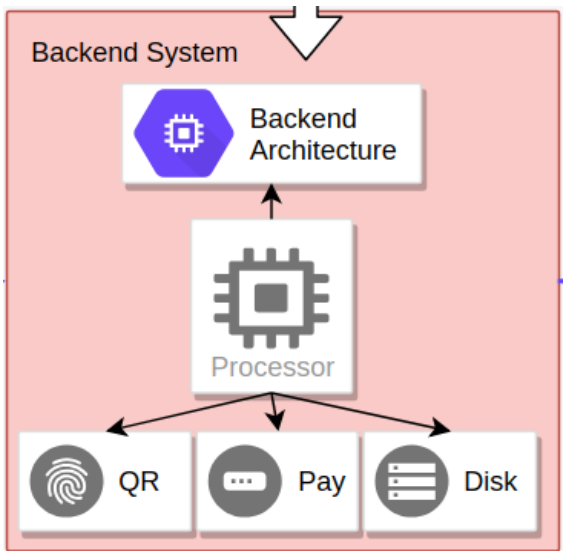


Figure 1.5: Backend System

The backend system houses the crucial components that are needed by the application to deliver a working solution to the user. This includes all the components that we have talked about in the previous parts. The system will be hosted on a regular desktop machine using the standard x86 architecture. The system developed will be a web application and also cater to the employees and servers. Being a non-native application means that there is no hassle of installation or updating the system and future maintenance. The system is automatically updated if there is any change in the hosting machine where the live codebase lives. This proves to be a better solution as more and more commercial applications such as supermarket stores and shops are relying on such software that is accessible through a browser for ease of use and familiarity when using the browser interface.

The mode of access for these servers will be focused on a web-based interface. The administrator accounts will be accessible from these servers and monitor the overall operation of the system. The administrator accounts are responsible for updating menu selection, prices and inventory. These accounts can also monitor user accounts and handle disputes if there are failed transactions.

The administrator accounts are enabled to track users' IDs and their order status (delayed/pending/ready) and transaction confirmations from the external processor. Interaction with the payment processor is discussed in further chapters using interaction diagrams and flowcharts to understand system behavior in a step by step process.

There are different routers for handling requests made by the users and administrator accounts. The separation of routers means that functionality is not affected in the event of changes or upgrades to a different router. This philosophy is entailed so that a modularised approach is possible when thinking about future upgrades or adding features on the fly without halting the entire system.

The backend system is mainly responsible for the following events:

- Prepare Transaction Details.
- Generate QR using Unique Strings.
- Log Completed Payments.
- Real-time Menus.
- User Tracking.

The “QR” module is facilitated by an external service called “Instascan.js”. The use of this service is free and is handled by a browser using JavaScript which allows the user to take a client-side camera interface to recognize QR patterns from an external display/device. Using said service, recognizing QR images can be done on the client side with no additional hardware needed such as a barcode scanner. This substantially reduces the cost of having to install additional hardware, as every smartphone nowadays has camera functionality. The module is also tweaked so that if it fails to initialize camera access (primarily the back camera), it will automatically switch to the front camera as a secondary.

The transaction details are prepared from the data received by the external payment processor. Each deposit for a user is recorded both in the database system and in the external payment processor’s log details which will be further discussed in the external component’s chapter.

Completed payments made through the application are logged by the internal disk (database system). This is done for credibility and to tackle future disputes if it arises in any such occasion. The accountability has to be taken into consideration when using such an application as such this is a very important part to keep logs of each transaction.

The real-time menus are used to feature items on the fly and instantaneously on the client side applications through the means of an API. The generated data of items can be fetched easily by clients, and the administrator accounts can add, remove or update items easily through the dedicated router responsible for handling item inventory.

The backend system also has an option to generate sales information dynamically from all the receipts that are stored inside the database. The external service used is called “CanvasJS” which enables the system to draw pie-charts and bar graphs of sales stats seamlessly. This is useful for tracking the month-month or day-to-day output of sales and finding which items are best sold through the application.

One of the main tasks of the backend system is to generate unique QR strings based on the user details to keep each log identifiable from other logs. This is done to ensure privacy and security of transactions made so that users can easily confirm their transactions through scanning the QR code generated on the server side monitors/displays. Having each QR string unique means that no user will be able to confirm a separate order made through a different smartphone/device that is not tied to their account. This makes it harder for fraudulent activities or erroneous attempts at the server terminal when collecting food.

The backend system is also able to order food on the fly for customers that prefer to pay cash, and as such a dedicated option is enabled by default for these types of manual transactions. The interface is similar to the frontend design and administrator enabled accounts will be able to order food for customers that insist on paying up front instead of using the in-app credits.

4.5 Database System

The database system is designed using a web-based interface called PHPMyAdmin which uses MySQL as its database query language for talking to the system. The rudimentary design of the database is done using an approach that is to have each entity a separate table for storing information. Since MySQL is a relational database system, we can chain together multiple tables and have the concept of foreign keys couple these entities.

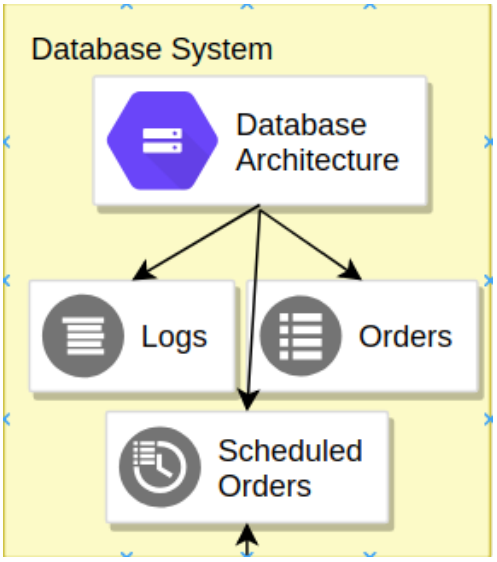


Figure 1.6: Database System

The database system is responsible for acting as a permanent disk of all transactions, records, logs generated when using the system. This gives an added advantage of not having to waste additional space when managing storage. The database can query information that is needed on the fly using PHP's "mysqli" extensions. This makes it easy to access the database where the user/administrator can access information tied to their respected accounts in a fast and efficient manner.

The database system keeps track of sessions generated when a user(client/admin) logs into the system, the session generated is responsible for showing appropriate options depending on the user level and the appropriate router from the backend is called whenever a user is operating within the application. The privilege level determines whether a user can access

certain elements inside the application, only privileged users(administrators) can access other user accounts and modify if deemed necessary. This type of session tracking is facilitated by PHP's "session" variable and can keep track of users logged inside the application.

A high-level overview of all the tables that are being used by the database system is given below. The connection between the tables is shown using green and red arrows and helps to conceptualize the data flow inside the application through various backend routers.

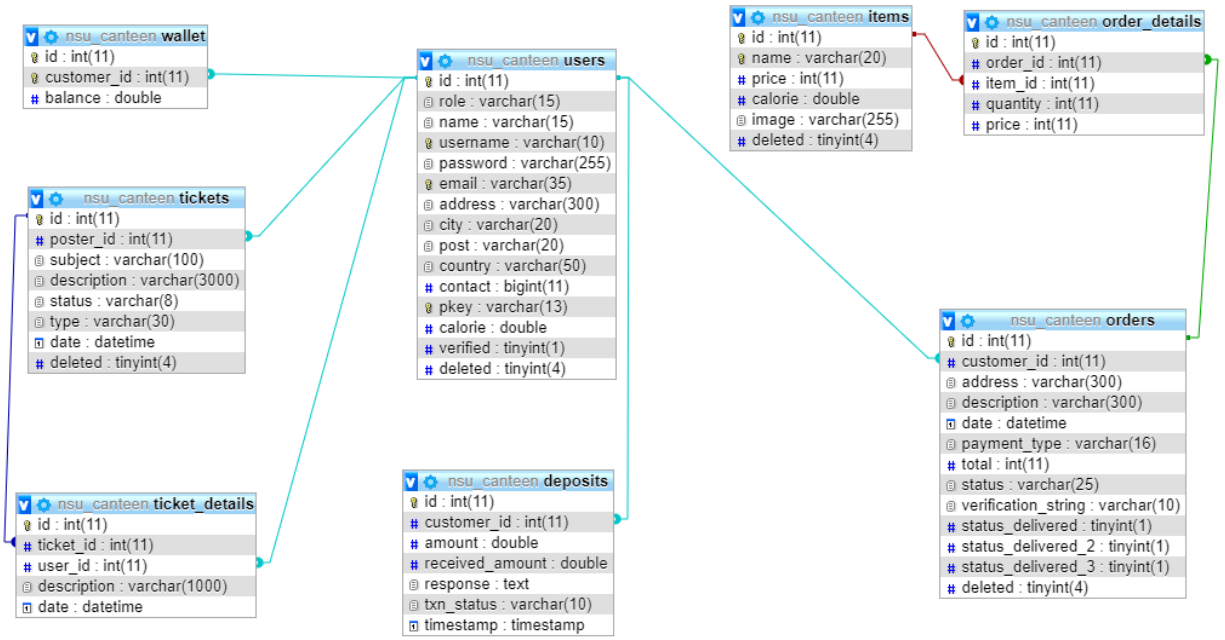


Figure 1.7: Database Tables

The list of individual tables and their entities with types are listed below:

Entity	Type
id	int(11)
role	varchar(15)
name	varchar(10)
username	varchar(255)
password	varchar(255)
email	varchar(35)
address	varchar(300)
city	varchar(20)
post	varchar(20)
country	varchar(50)
contact	bigint(11)
pkey	varchar(13)
calorie	double
verified	tinyint(1)
deleted	tinyint(4)

Figure 1.8: nsu_canteen - users

Entity	Type
id	int(11)
customer_id	int(11)
balance	double

Figure 1.9: nsu_canteen - wallet

Entity	Type
id	int(11)
customer_id	int(11)
amount	double
received_amount	double
response	text
txn_status	varchar(10)
timestamp	timestamp

Figure 2.0: nsu_canteen - deposits

Entity	Type
id	int(11)
name	varchar(20)
price	int(11)
calorie	double
image	varchar(255)
deleted	tinyint(4)

Figure 2.1: nsu_canteen - items

Entity	Type
id	int(11)
customer_id	int(11)
address	varchar(300)
description	varchar(300)
date	datetime
payment_type	varchar(16)
total	int(11)
status	varchar(25)
verification_string	varchar(10)
status_delivered	tinyint(1)
status_delivered _2	tinyint(1)
status_delivered _3	tinyint(1)
deleted	tinyint(4)

Figure 2.2: nsu_canteen - orders

Entity	Type
id	int(11)
order_id	int(11)
item_id	int(11)
quantity	int(11)
price	int(11)

Figure 2.3: nsu_canteen - order_details

Entity	Type
id	int(11)
poster_id	int(11)
subject	varchar(100)
description	varchar(3000)
status	varchar(8)
type	varchar(30)
date	datetime
deleted	tinyint(4)

Figure 2.4: nsu_canteen - tickets

Entity	Type
id	tinyint(11)
ticket_id	tinyint(11)
user_id	tinyint(11)
description	varchar(1000)
date	datetime

Figure 2.5: nsu_canteen - ticket_details

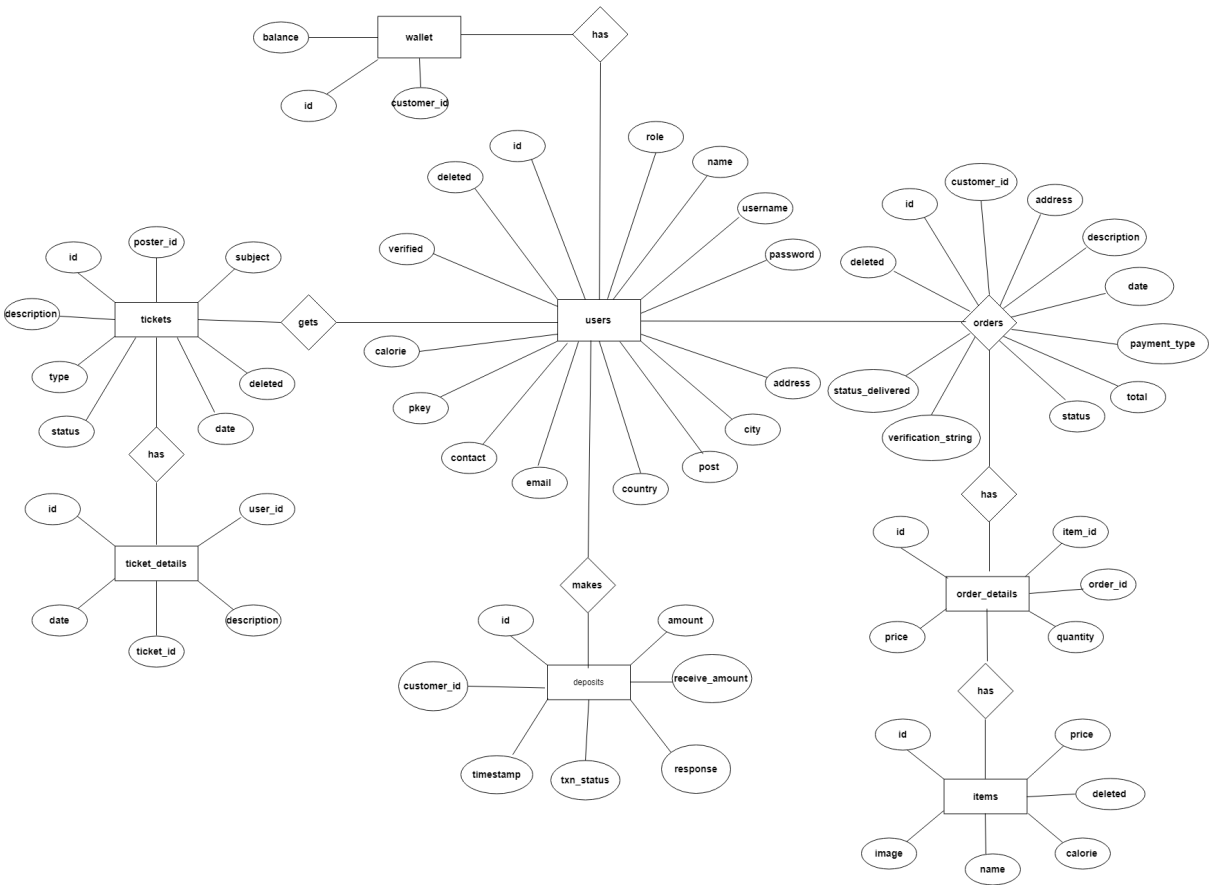


Figure 2.6: ER Diagram

The Entity Relation (ER) diagram is an excellent way to help understand database systems, and the one being used by our application is given above in figure-2.6. With the help of the ER diagram, we can see the data flow inside the application. The ER diagram in a way acts like a flowchart and illustrates how different entities like objects or concepts relate to each other within a system. The diagram shows how the “user” accounts are related to entities such as “balance” and “orders” and how might they be interlinked using the concept of “foreign keys” using “ids”. This not only solves the problem of duplicate entries which are redundant but also gives constraints which apply to a real-world scenario. For example, a user account can only see order summaries and details that are tied to that particular account whereas an administrator account can view all details inside the database system.

4.6 Summary

This chapter, therefore, concludes the working schematic of the underlying application. All the related components that were conceptualized in the design phase are given a formal introduction using diagrams and explanations were necessary to help understand the inner works of the system. The different subsystems of the architecture defined in this chapter form a cohesive whole and form the moving parts of the application. A thorough emphasis on modularity was given so that future upgrades or changes are quick and easy without affecting the main structures of the program. The external components are discussed in the next chapter.

CHAPTER 5

EXTERNAL COMPONENTS

5.1 Introduction

In this chapter, we will formally introduce all the external components that we visited in the schematics of the previous chapter. The external systems and their working components will be explained in detail along with their documentation and implementation details for the project. The components used inside the project are all open source and can be obtained free of cost as long proper citation is done and credit is given to the original developer.

The core workings of the application would not have been possible without the inclusion of these third party solutions.

5.2 External Payment Processor

The external payment processor is responsible for all and any transactions that require the user to provide credentials for their cards or bank account numbers. The use of an external payment processor is primarily because of security concerns since it involves actual user bank accounts and card numbers as credentials. If any of those details are compromised, both the user and the use of this application is affected in terms of security. It also alleviates potential unseen loopholes in the case of transferring money, and as such an external solution is a much superior choice compared to an in-house solution.

The external payment processor is facilitated by **Walletmix** (https://www.walletmix.com/).

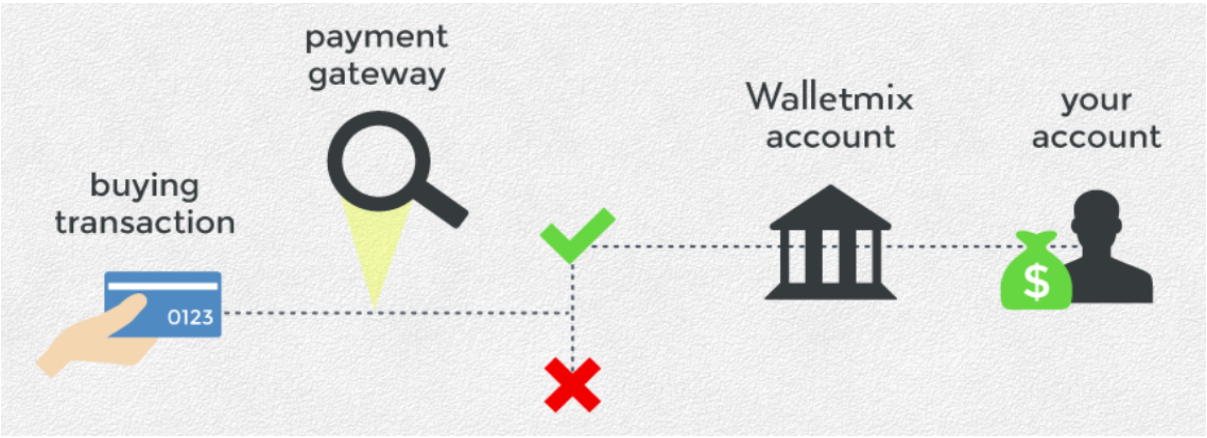


Figure 2.7: Payment Processor - Walletmix

The reasons why we are using a payment processor instead of coming up with a payment solution of our own is because it would take time to develop and there are security risks. So, we chose a well-established payment processor with state-of-the-art infrastructures to take care of security risk and also help us reduce the workload by a lot. The payment processor we selected after reviewing other suitable options is Walletmix.

Walletmix is a local brand specialized in handling a massive amount of transactions and offers almost all sort of Bangladeshi payment methods available (except physical cash) till date (e.g. credit cards, online banking, debit cards, mobile payments, bKash, Rocket, Ucash, etc.). They already have a good reputation in the industry sector of Bangladesh for online payments and their variety of payment methods would certainly help attain a diversified customer base which would not have been possible otherwise.

The use of Walletmix as the payment carrier is a welcome addition to the application where security concerns are handled by a company that specializes in this field and provides easy interfaces to manage fund transfer and logs. The accountability when using Walletmix is further by the strict rules enforced when using their APIs, and without adhering to them, any balance transfer will fail if it detects fraudulent activity.

The series of figures below will describe the interface when operating with Walletmix publicly available APIs.

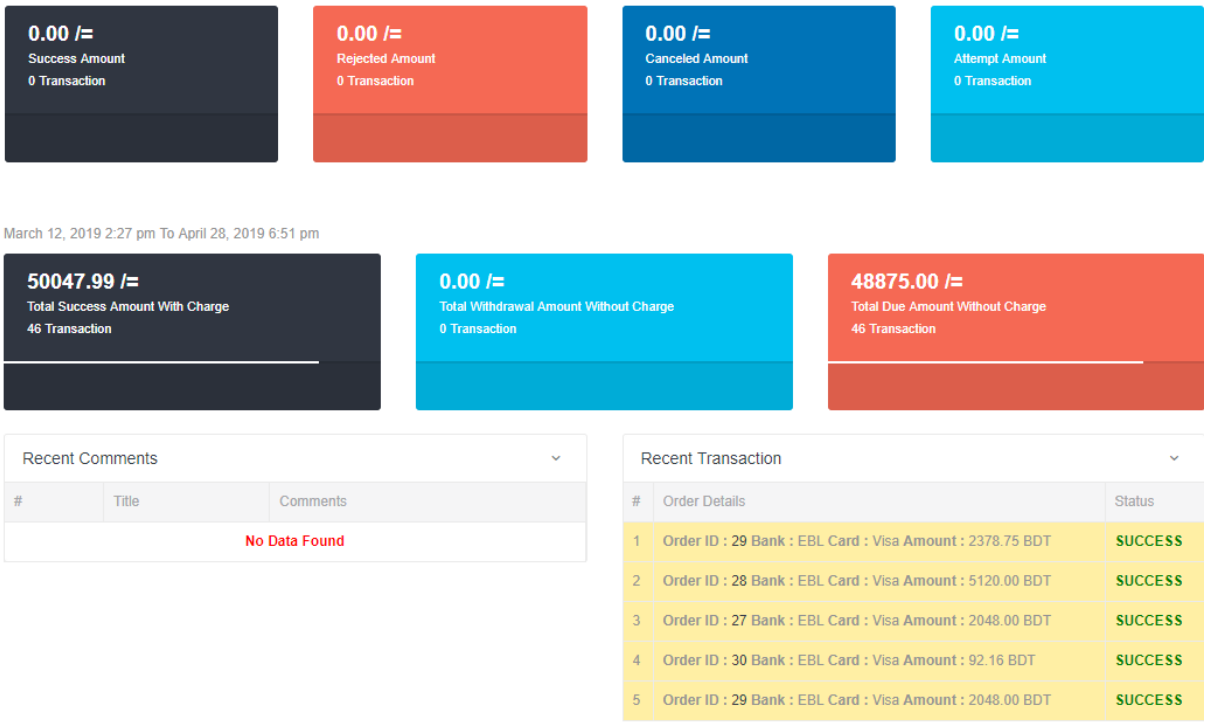


Figure 2.8: Walletmix Logs

The most recent transactions are shown in “cardboard” boxes in bright colors and a list is also given for past transactions. There is also a comment section available to mark certain transactions in the event of a failure or suspicion.

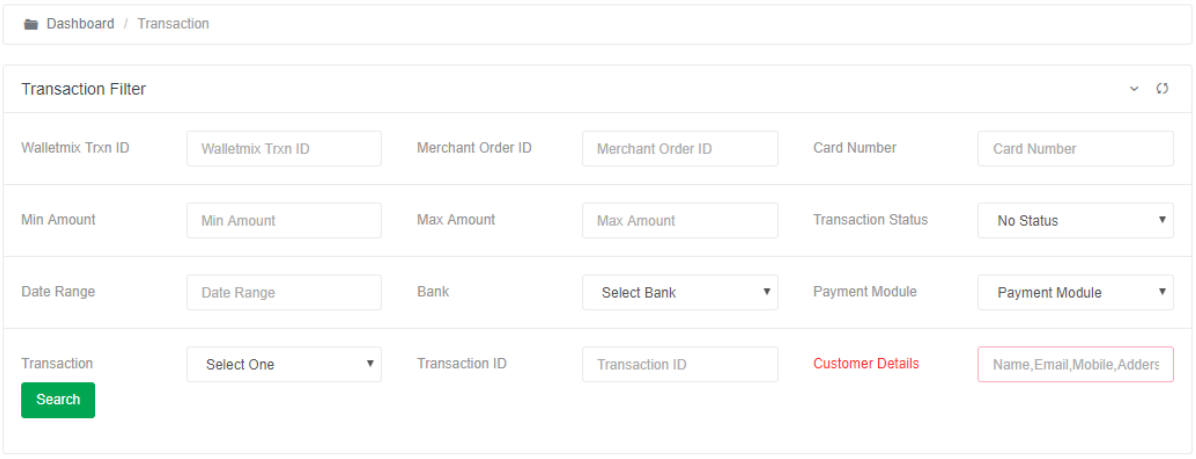


Figure 2.9 Walletmix Dashboard

The dashboard features handling of multiple accounts registered through their API and provides a clean interface.

The most important features accessible through the dashboard are as follows:

- Merchant Accounts
- Bank Accounts
- Payment Module
- Transactions Status
- Customer Details

Figure 3.0: Walletmix Detailed Logs Panel

The detailed logs panel provided through Walletmix web application references all data that is responsible for completing a transaction successfully. This is where all receipts and logs are stored permanently so that any concerns regarding disputes are handled easily ensuring accountability. The list of all credentials and details regarding a transaction is noted below.

Merchant Order Details	Merchant Trxn Details	Card & Customer Details
Walletmix Trxn ID	Merchant Amount	Card Number
Merchant Order ID	Card Charge	Bank
Bank	Extra Charge	Country
Card	Paid Amount	Card Type
Reference ID	WMX Payable Amount	E-mail
Merchant ID	Request IP	Attempt Time
Transaction Type	Bank Status	Trxn Time
Website	Release Status	
Process Completion		

Figure 3.1: Walletmix Detailed Log Entries

Figure 3.2: Walletmix Supported Banks/Cards

The above banks and cards are currently supported by Walletmix in figure-3.2. For demo purposes, only EBL Mastercard transactions are allowed, but all other banks/cards can be supported through the API once deployed.

Figure 3.3: Walletmix Security Features

The key security features of Walletmix is outlined below:

- **Virtual Terminal** - Walletmix payment gateway has virtual terminal processing credit cards over the Internet as compared to a physical POS card swiper.
- **SSL** - Walletmix payment gateway uses SSL data transmission for establishing an encrypted link between a client and a server.
- **CCV** - Walletmix payment gateway uses CCV (Card Code Verification) as an extra measure of security against fraudulent credit card transactions.
- **AVS** - Walletmix payment gateway uses AVS (Address Verification Service) for checking and verifying the address of a person claiming to own a credit card.
- **PCI DSS Compliant** - Walletmix payment gateway follows the Payment Card Industry Data Security Standard.

INSTRUMENTS	BASIC	BUSINESS
CARD	3.5%	2.8%
MFS	2.5%	2%
INSTALLATION CHARGES	8000	12000
MONTHLY	0	3000
	SIGNUP	SIGNUP

Figure 3.4: Walletmix Pricing

The pricing structure of deploying a commercial application built on top of Walletmix services is listed above in figure-3.4. For demonstration purposes, a demo emulator has been used which can be obtained free of cost from Walletmix.

5.3 QR Scanner

The QR Scanner that is implemented inside the application is responsible for identifying all QR codes generated from the backend during food delivery confirmation. The choice of a QR scanner that is both fast and compatible with a wide range of devices is very important since the demographic that is being targeted is inside a university canteen. The QR scanner, therefore, has to work under different platforms most notably on Android and Apple's iOS.

The QR scanner that we opted for is from an open source library named "**instascan.js**" (https://github.com/schmich/instascan). The API provided by Instascan fulfills most of the required functions needed by the application while providing very fast performance.

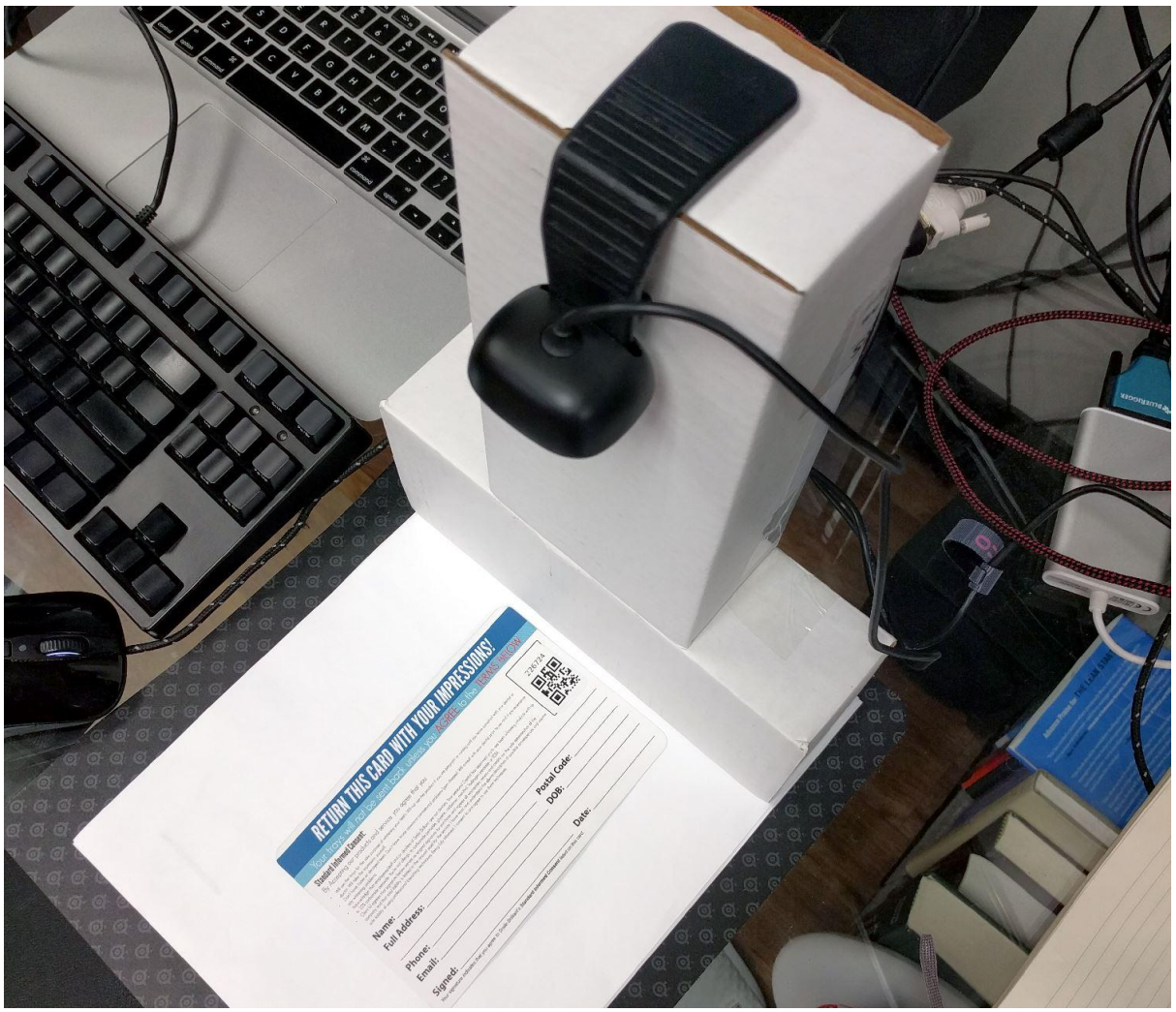


Figure 3.5: Hardware Setup using Instascan

The above picture is using a Microsoft LifeCam HD-3000 along with the QR Scanner library provided by Instascan. We can implement the same setup using the smartphone's front or back cameras instead of having a dedicated webcam or QR scanner. This provides the much-needed benefit of not having to install additional hardware driving down costs of an actual setup of the project.

Instascan works on non-iOS platforms in any browser that supports the WebRTC/getUserMedia API which currently includes Chrome, Firefox, Opera, and Edge. The browsers that are not supported are IE and Safari.

Some example functions of using Instascan API are given below:

- let scanner = new Instascan.Scanner(opts)
- scanner.start(camera)
- scanner.stop()
- let result = scanner.scan()
- scanner .addListener('scan', callback)
- Instascan.Camera.getCameras()

Guidelines when scanning for a QR image on a display are given below:

- QR code orientation doesn't matter.
- Higher resolution video is better but is more CPU intensive.
- Direct, orthogonal scanning is better than scanning at an angle.
- Blurry video greatly reduces scanner performance.
- Auto-focus can cause lags in detection as the camera adjusts focus. Consider disabling it or using a fixed-focus camera with the subject positioned at the focal point.
- Exposure adjustment on cameras can cause lags in detection. Consider disabling it or having a fixed white backdrop.

The service that is being used to generate the actual QR image on the display/monitors is using the service(http://goqr.me/api/doc/create-qr-code/). From the backend server, using goqr.me's API, the function create-qr-code() can be used to create an embedded QR image on a webpage inside a browser. This is ultimately displayed to the end user when they are at the food serving terminal for confirming their order before pickup.

5.4 Push Notification Service

The push notification service is one of the most important aspects of the application. This module is responsible for alerting the user the instant they place their orders and receive status updates. Instead of developing this push notification service, we have opted for an industry level solution from OneSignal (https://onesignal.com/). The main component from OneSignal that is being used inside the application is the web push notifications. The web push notifications is setup inside the application using their API which allows for quick and easy integration.

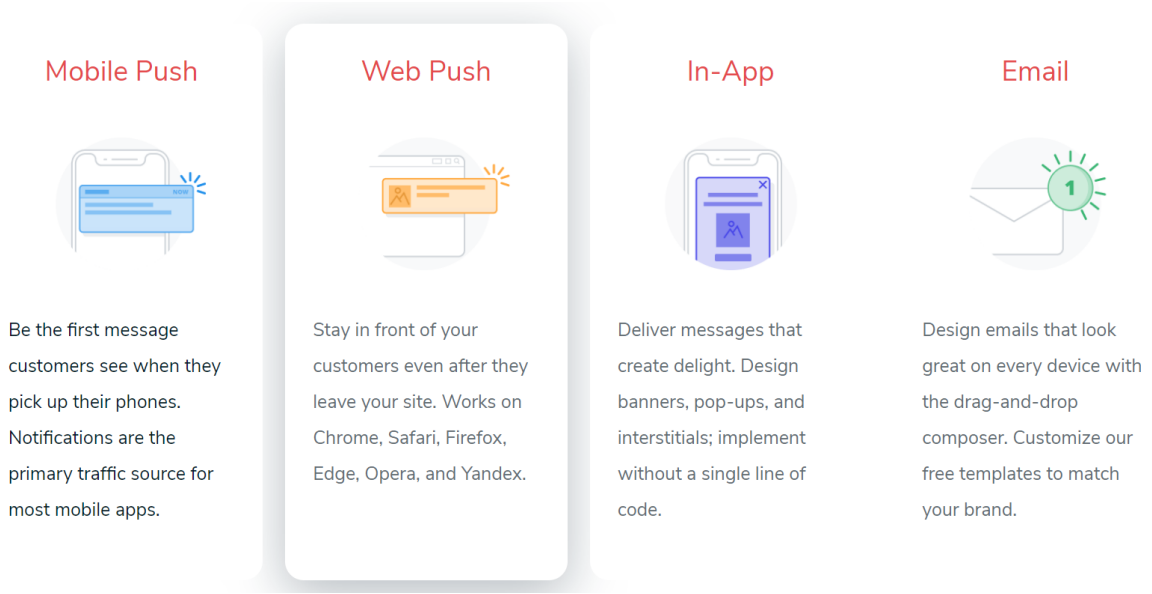


Figure 3.6: OneSignal Notification Types

The API allows for different types of notifications based on the needs of the application being designed. The primary web push notifications allow the application to send the following important messages throughout the service lifetime:

- Order Status Updates.
- Deposit Status Updates.
- Menu Changes.
- QR Confirmation Updates.
- Custom Messages.

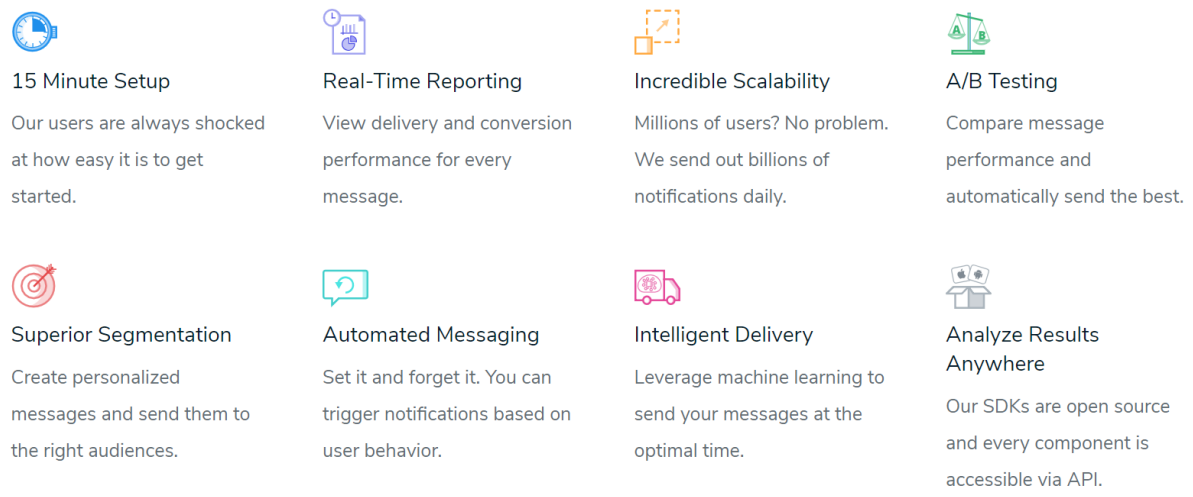


Figure 3.7: OneSignal Features

We have heavily used the "Automated Messaging" and "Superior Segmentation" in figure-3.7 for completing the instant notification service built inside the application. This allows users to have real-time updates based on the feedback from the backend server automatically to the client devices without having to manually notify. OneSignal API makes this very easy to implement without backend routers as it is PHP compatible and also allows for custom tuning of messages and message banner types such as images and logos.

5.5 Summary

In this chapter, we have defined all the inner workings of the external components mentioned in the schematic architecture diagram from chapter 4. This includes any and all services that were not developed by us but nonetheless included in the project to avail certain features that are needed by the application. This chapter also has the links mentioned about the names of the servers along with their APIs should the need arise to check them in more detail.

CHAPTER 6

SYSTEM COMMUNICATION

6.1 Introduction

In this chapter, we will introduce the inner workings of the whole system when it is live guided with flow charts and diagrams. This will help to visualize and understand the dataflow from one component to another without the need to understand any underlying code which will be present in the appendix section. The whole system, even though it is a monolith, can be broken down into simple working solutions because a modular approach was taken early during the designing phase. The inner interactions between the different kinds of routers are presented in the upcoming parts along with explanations.

6.2 System Flowcharts

The following flowcharts will help illustrate how the core components are related and provide working solutions in a step by step approach.

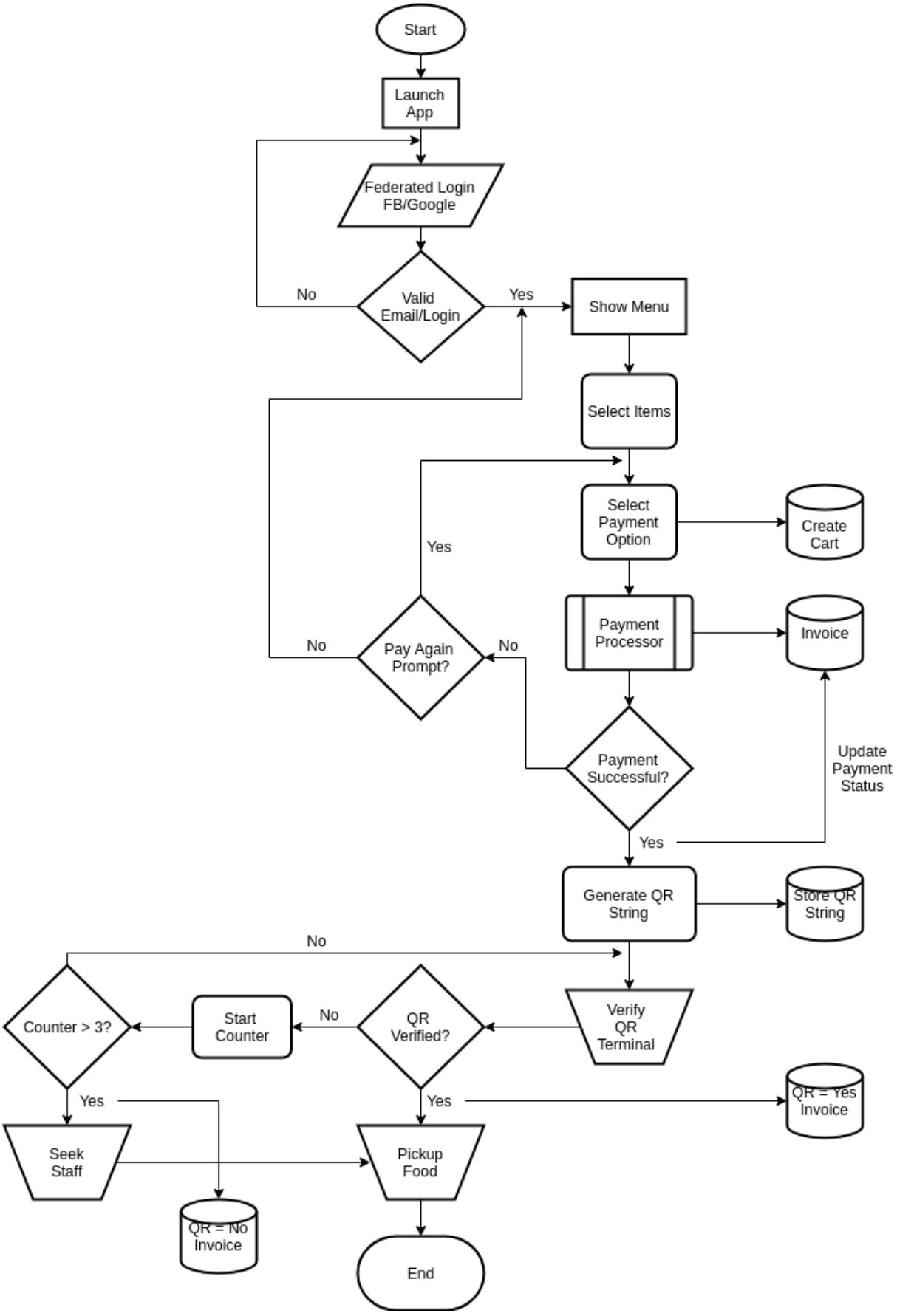


Figure 3.8: System Flowchart

In figure-3.8(System Flowchart), the user first needs to sign-up to the application by registering for an account. Once the backend server sends the authentication successful response, the user would be logged in to the food ordering system and the menu should be displayed for all available food to order. If the server receives a failed authentication response then it shall relay it back to the user and he/she can retry to log-in.

Next, the user shall select the foods from the displayed menu which he/she would like to order and then click on view cart. There the user should be displayed with the payment options available by the payment processor. Once the user clicks checkout an invoice should be generated in the system and stored in the database along with a request to process the payment for that specific invoice would be sent to the payment processor where the user should be redirected to for completing the payment.

If the payment is unsuccessful then the user should be asked to retry with a different payment method via the payment processors portal until the user cancels out, in which case the user should be taken back to the menu again. If the payment is successful, the payment processor should inform this status back to the server and this will generate a QR string in the database.

Once the order is processed by the servers and the user (customer) is ready to pick up the food, he/she should come in front of the counter and scan the generated QR code through the application. If he/she is the actual customer then the QR string should match and notification shall be sent to the server's terminal indicating the verification is complete and the server can handover the food.

In case it does not match, the system should allow another two times of retry and then ping a server to help out the user. Afterward, the system should update the invoice as complete and mark the order as delivered.

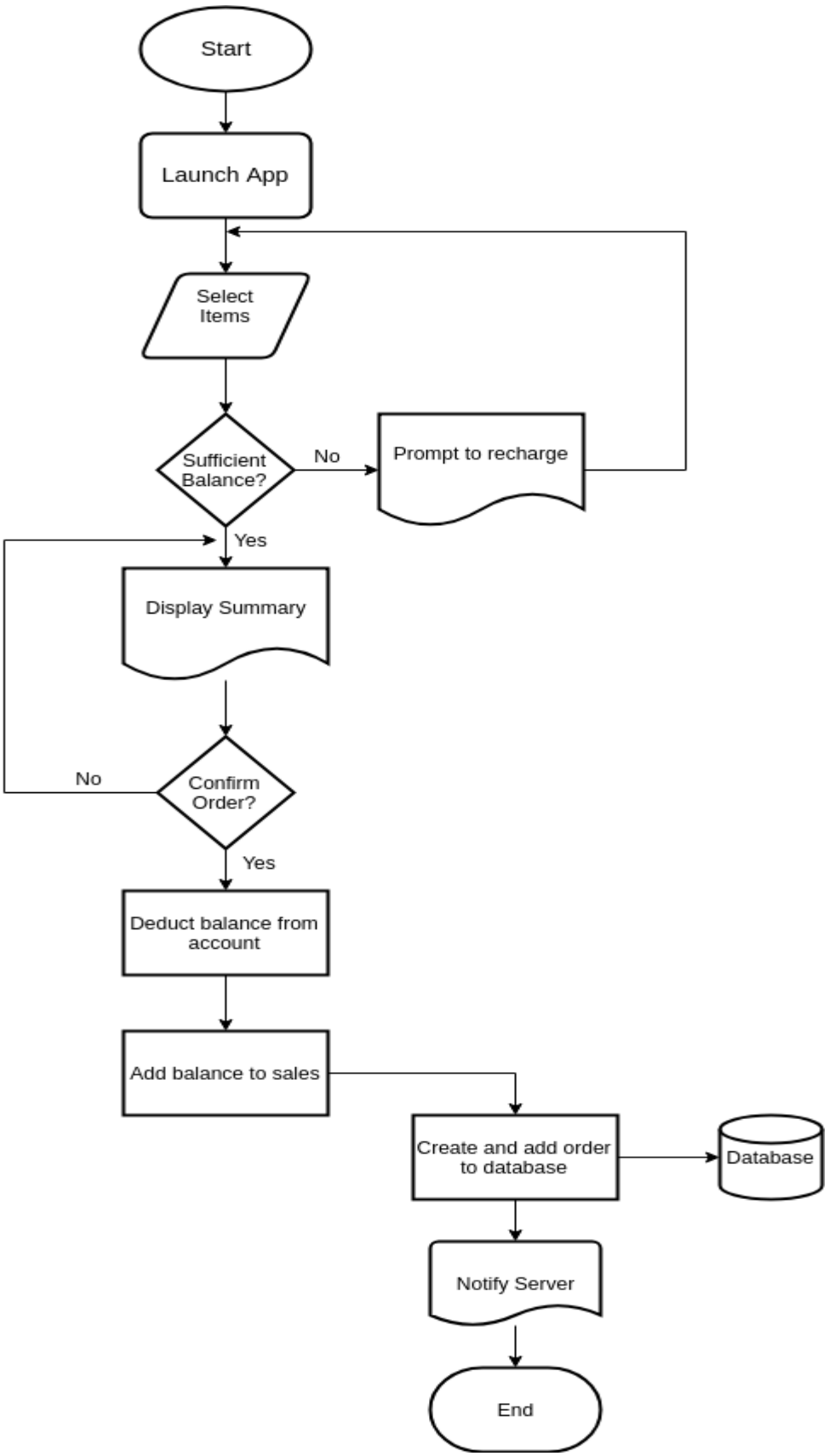


Figure 3.9: User Place Order

In Figure-3.9(User Place Order), the user should first launch the application from their device. Upon launch (assuming the sign-up process is complete) the user is displayed with all the available food items.

Next, the user should select items to the cart which he/she wants to order and follow the payout procedure. If the user does not have sufficient balance to pay for the order then the application should prompt the user to recharge their account. Once the recharge is complete it should take the user back to the item selection from where he/she can retry to pay for the items in the cart.

If the balance is sufficient then an order summary should be generated to the user for confirmation. Upon their confirmation, the cost of the order should be deducted from users' balance and added to the sales of the food catering company and then the order would be added to the database and notify the food servers.

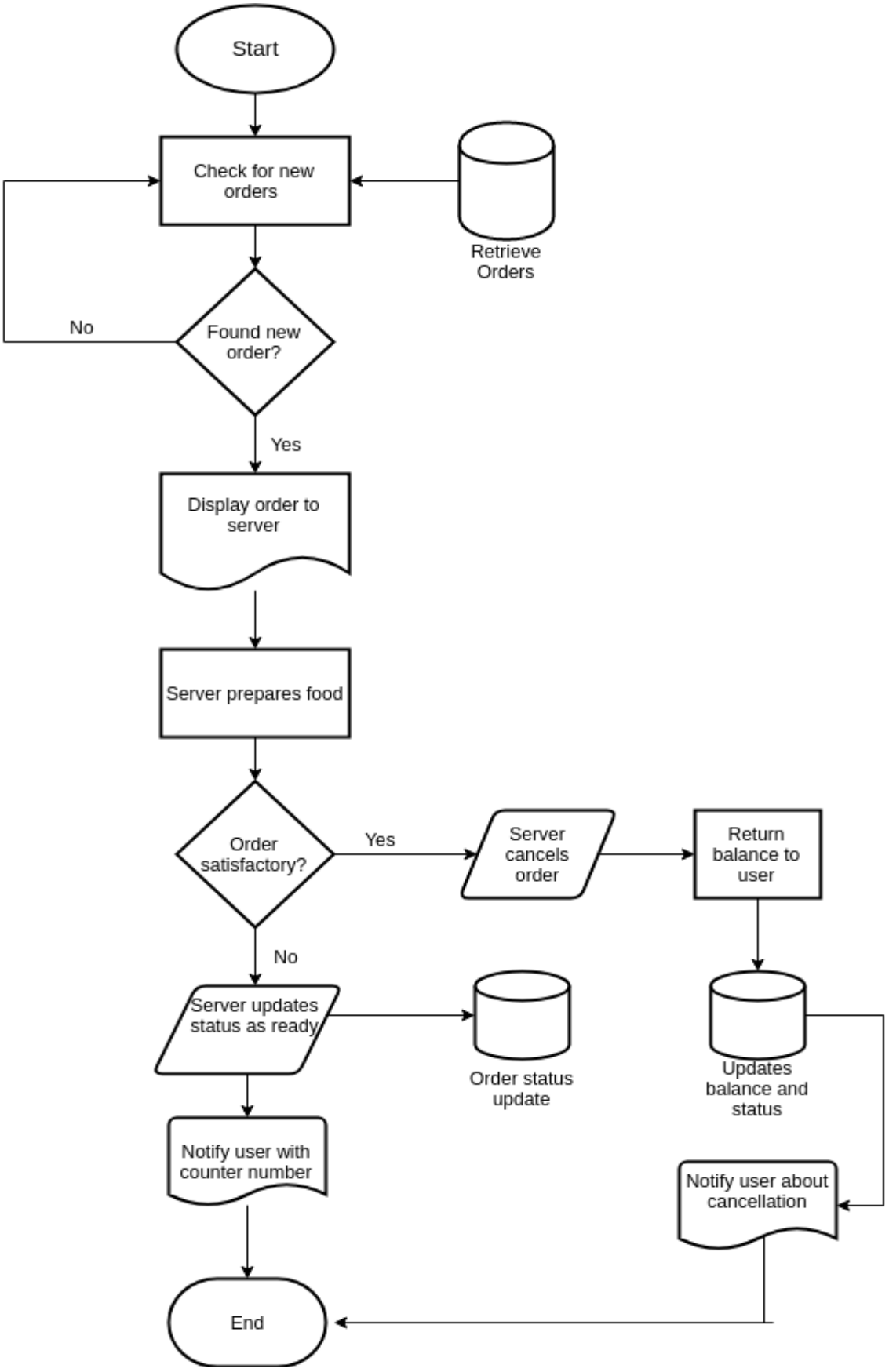


Figure 4.0: Server Processing Order

In figure-4.0(Server Processing Order), the server application should continuously keep pooling/listening for the arrival of new orders in the database. Once a new order arrives it should display the order details to the food servers. The servers should immediately start preparing the foods listed in the order. While preparing the order if the server notices that they ran out of any of the listed food or cannot provide it for any reason, they should be able to cancel the order and in that case, the balance should get refunded to the users account automatically.

The server would then notify the user about the cancellation so that he/she can decide to order something else and not waste time waiting on a canceled order. If the order is satisfactory for the server to prepare then once it's prepared, the server simply updates the status of the order to be ready for pickup from the server's application. This will immediately send a notification to the user's device which shall include from which counter to pick up the ordered food.

The interaction diagrams are also given below to help understand the underlying assumptions of the system. These diagrams allow a mental picture of what is happening underneath the hood when the subsystems of the architecture are talking to one another and exchanging data as the application is being used. It also shows which database entries are evoked and created when orders are being created and updated as the user is interacting.

6.3 System Interaction

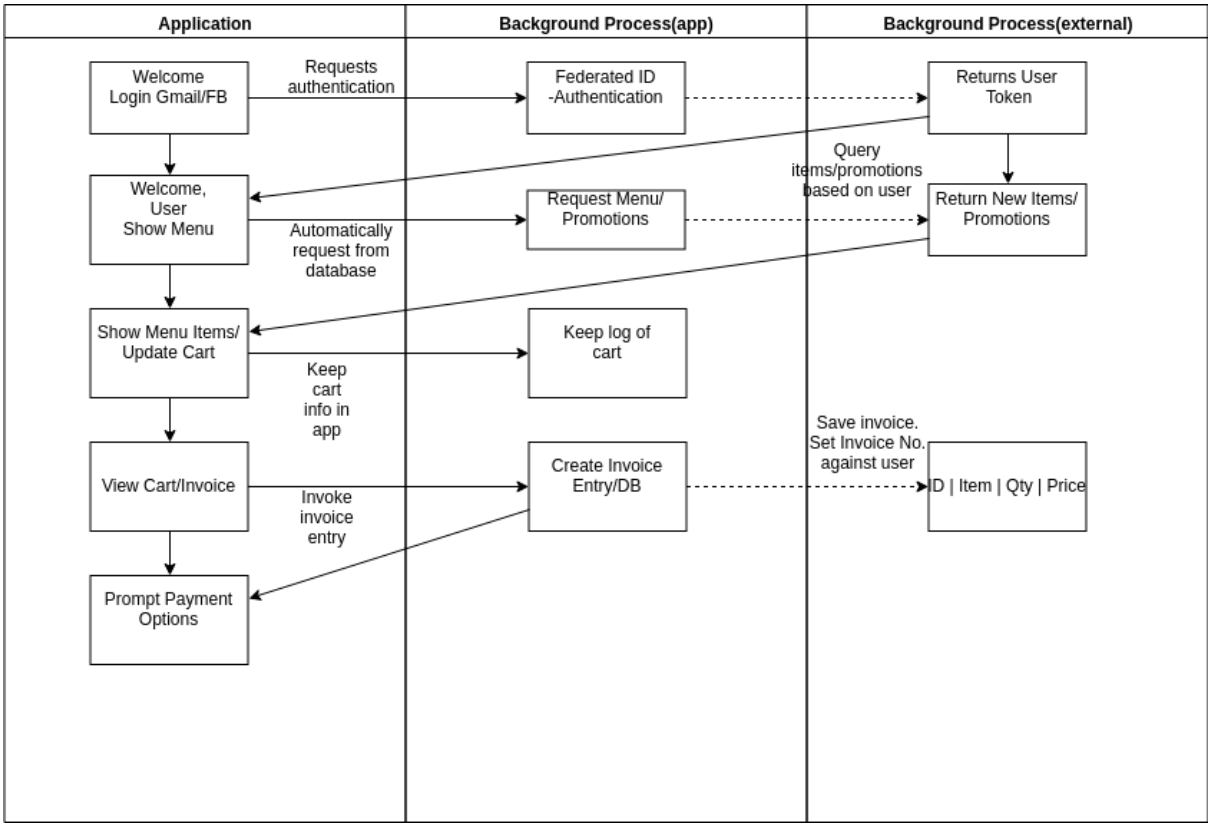


Figure 4.1: System Interaction Diagram Part 1

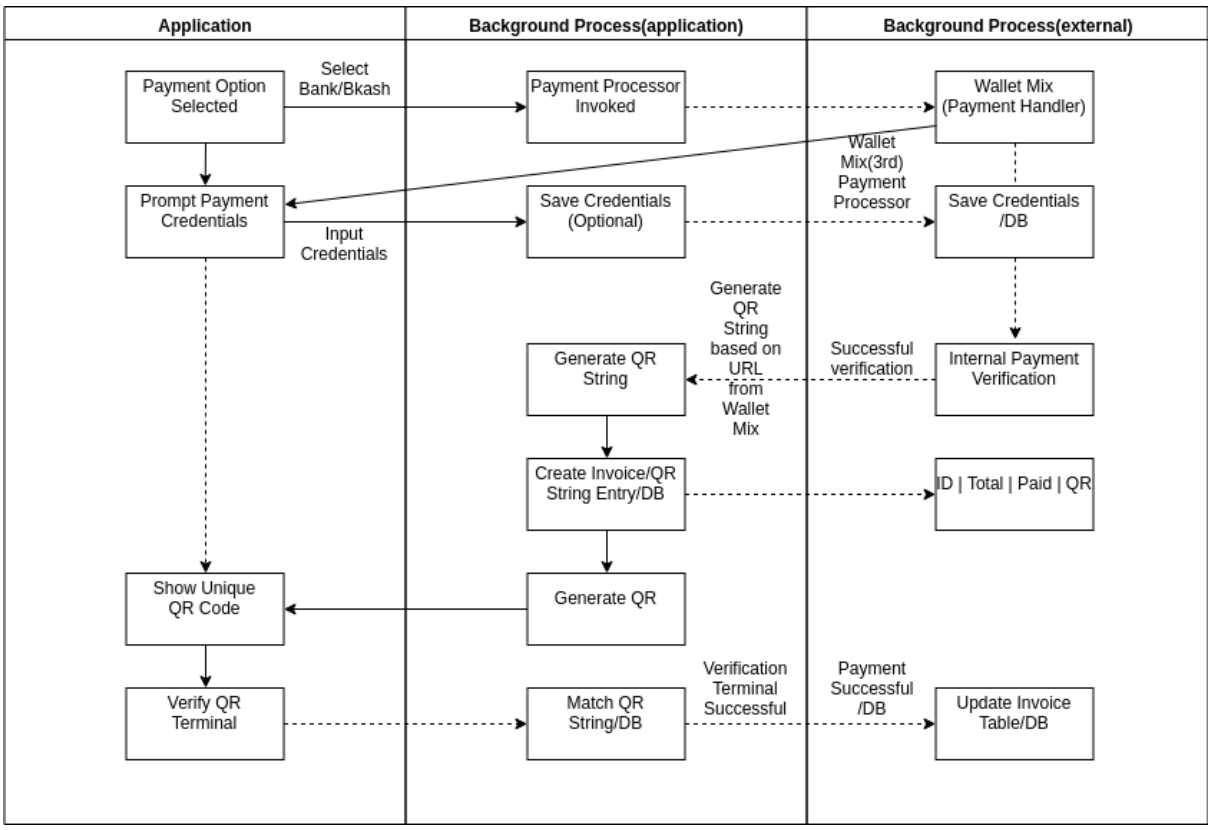


Figure 4.2: System Interaction Diagram Part 2

1. The frontend application grants the user a log-in to the system which is authenticated using the backend server.
2. A user token is turned, typically a JSON object containing names/uID or a PHP session variable.
3. Upon successful login, the frontend application automatically requests for the latest menu and returns a list of prices and promotions if available from a database query.
4. A user cart is generated and the session is stored on the device/browser.
5. When the view cart/invoice is selected, the entry is pushed to the database against the user ID. The cart generated holds user-ID, Item, Quantity and Price.
6. The payment selection options are automatically prompted.
7. Selecting a payment option (bank/bKash) invokes the external payment processor.
8. Depending upon the payment type, the appropriate credentials are prompted on the frontend.

9. Once the payment is processed, Walletmix should trigger a callback URL to the server where they would send the status of the payment and invoice number through HTTP GET method.
10. Once the payment is verified, the server generates a unique QR string. The QR string is stored on the database against the receipt generated.
11. The QR pattern is then sent to the display/monitor at the food serving terminal based on the unique string.
12. Once the user matches the QR pattern against the terminal, the receipt is updated to ”paid” status and the transaction is complete.

6.4 System Use Cases

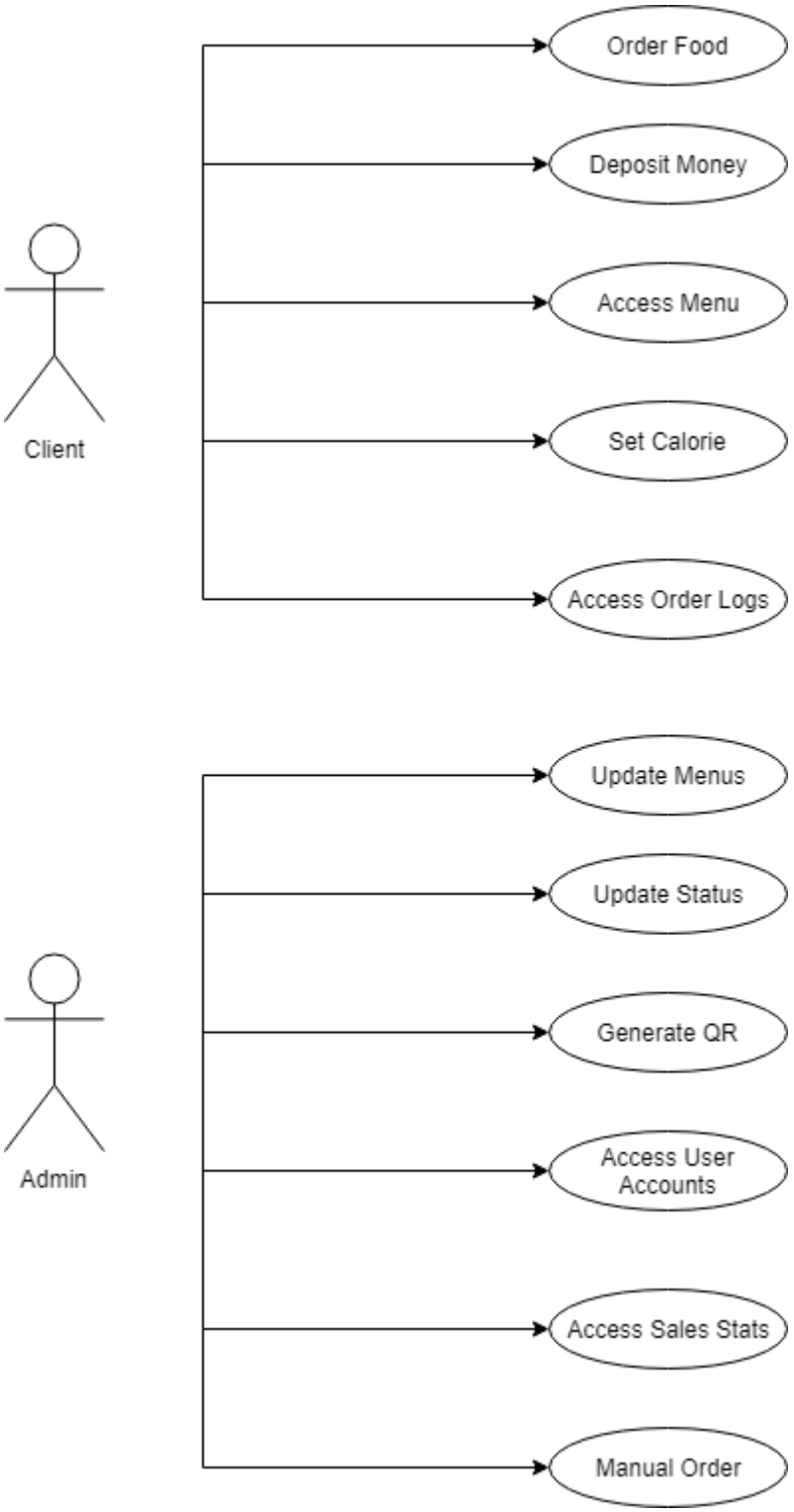


Figure 4.3: System Use Cases

The use cases of the system from the user and administrator perspective are given in figure-4.3. This will help to identify and distinguish between user accounts and privileged staff/employee accounts. The use cases of the system enlist all the basic functionality that each kind of account is expected to have in a working application unless otherwise revoked.

6.5 Summary

In this chapter, we have outlined all the inner workings of the system using detailed explanations and step by step points where necessary. Any underlying assumptions regarding the system are laid out in diagrams with the components formally introduced in the previous chapter. It is paramount to understand that not all of the nitty gritty details can be put on paper about the working code used to build the system and therefore some parts have been simplified for ease of understanding.

CHAPTER 7

SYSTEM INTERFACE

7.1 Introduction

In this chapter, we will introduce you to the actual application software that is deployed on a live hosting site (https://nsucanteen.ddns.net). The domain was already available and hence the working application is uploaded to that site. The application is already running and can be tested anytime with real-world users before actual deployment. We will show the interfaces that the users of this application will see when they interact with it and guide you through the process using figures and explanations.

7.2 Client User Interface

In this section, we will guide you through the user (client) interface as well as a typical order through the application in a step by step manner. Certain assumptions will be made that the user knows his/her credentials and can operate a smartphone keyboard/similar technologies on a typical Android/iOS device.

Step - 1: Provide user credentials (name/password) into the designated fields.

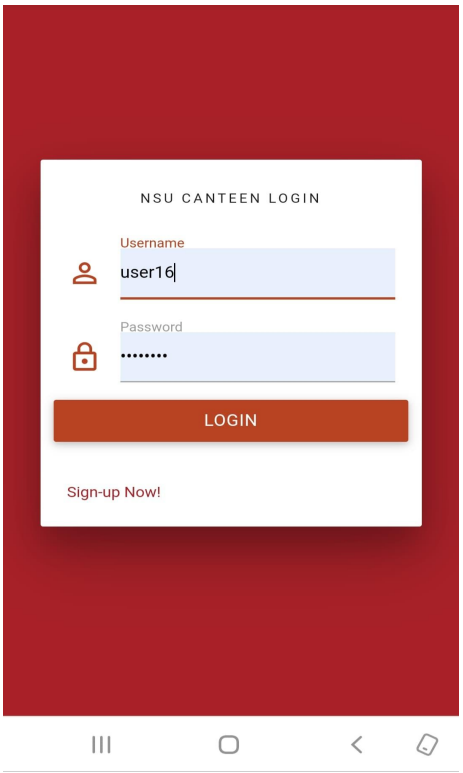


Figure 4.4: Client Login

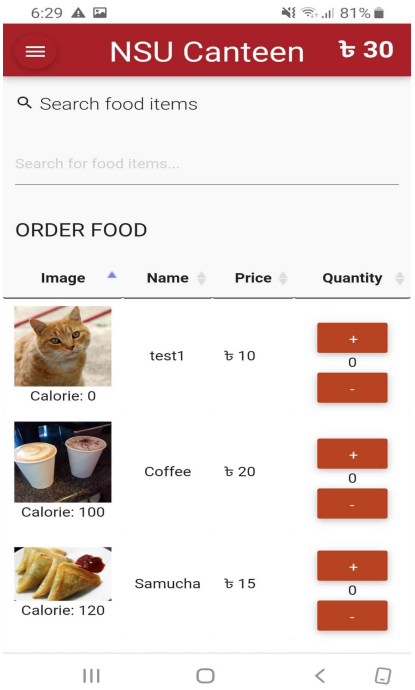


Figure 4.5: Successful Login

Step - 2: Choose items by clicking the “+” buttons for adding food to the cart.

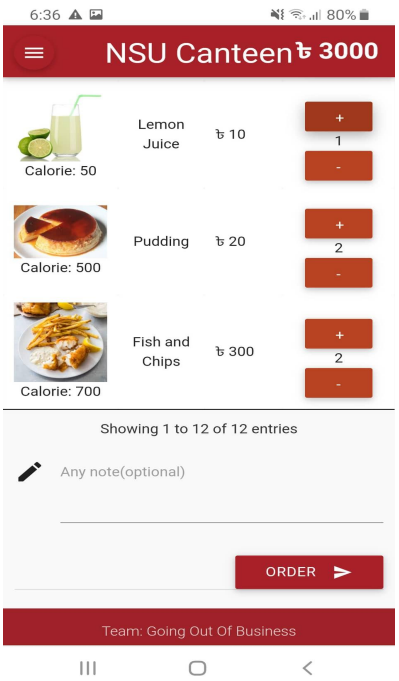


Figure 4.6: Items Added to Cart

Step - 3: Check Estimated Receipt

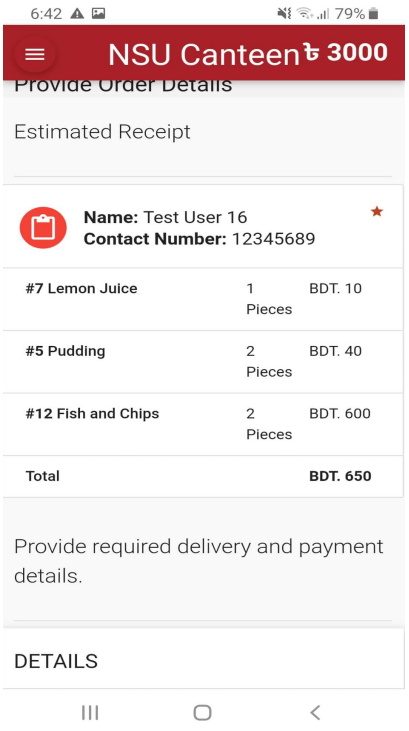


Figure 4.7: Estimated Receipt

Step - 4: Choose option “Wallet” and hit “Pay”

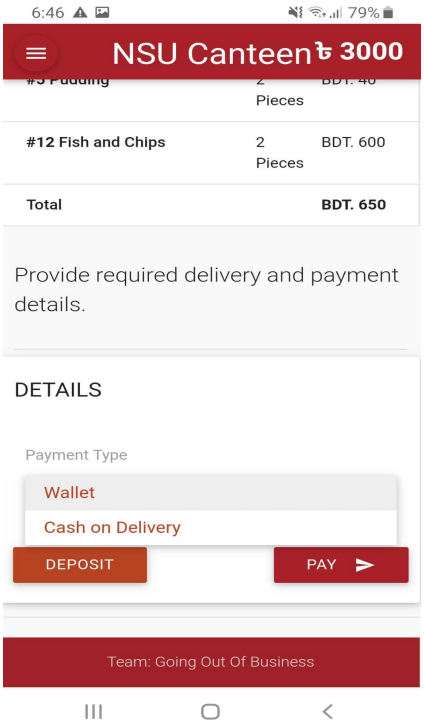


Figure 4.8: Choose Wallet and Pay

Step-5: Check balance deduction and confirm the order.

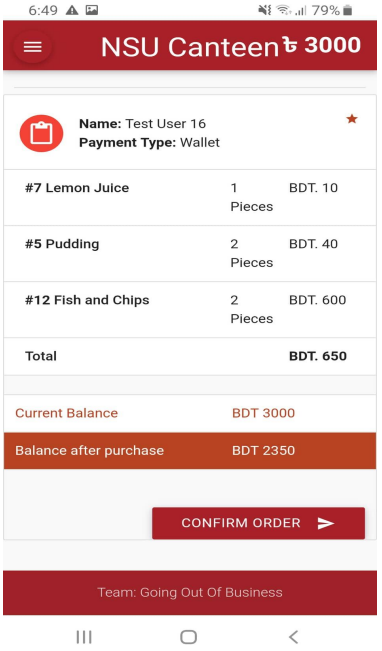


Figure 4.9: Confirm Order

Step-6: Wait for Admin side confirmation, and auto refresh will change the “Cancel Order” button to “Verify QR”.

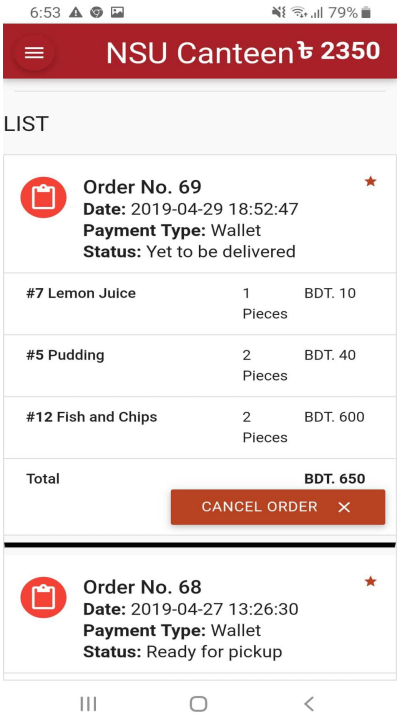


Figure 5.0: Pending Order

Step-7: Admin side updated automatically, push notification received.

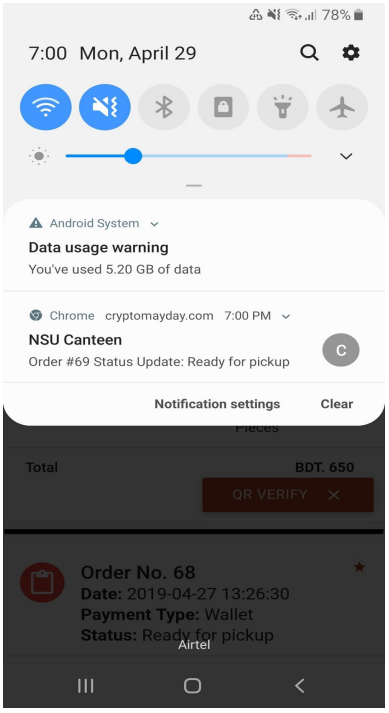


Figure 5.1: Notification Received

Step-8: Hit “QR Verify” to open Camera.

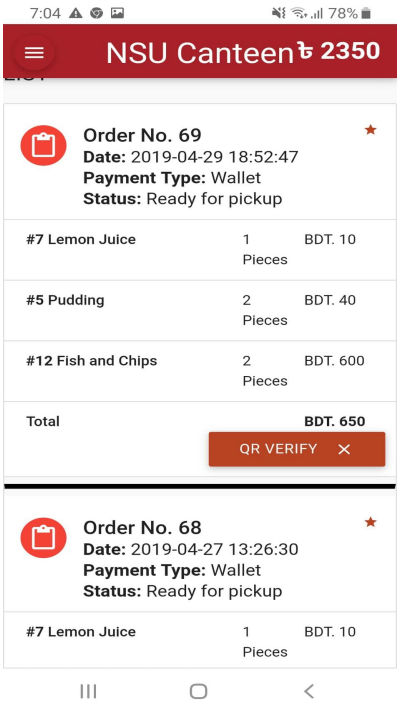


Figure 5.2: QR Verify

Step-9: Go to the food terminal to scan the QR image generated from the admin side.

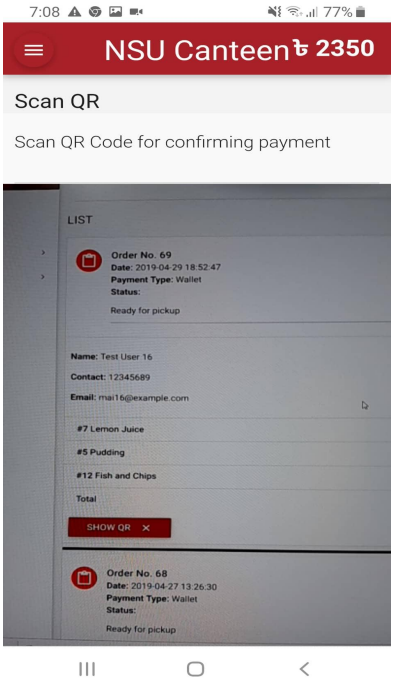


Figure 5.3: QR Camera

Step-10: After successful verification, pickup and receive food from the terminal.

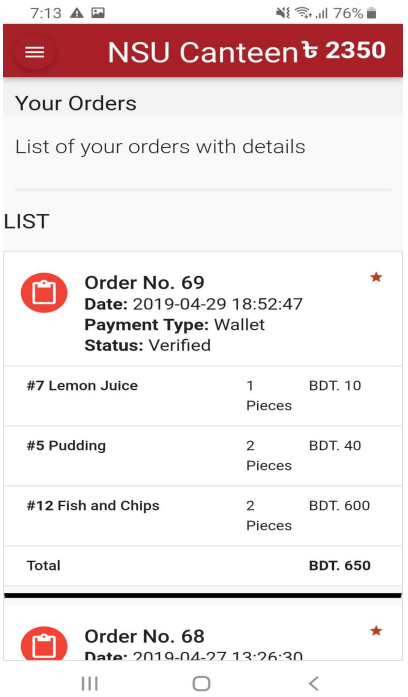


Figure 5.4: Order Verified

7.3 Server Administrator Interface

In this section, we will outline what goes on during an administrator session when an order is placed. This panel will be typically handled by food servers and employees catering to canteen service. Again certain assumptions are made that the user is comfortable using web-based technologies and is familiar with modern browser interfaces.

Step - 1: Provide administrator credentials (name/password) into the designated fields.

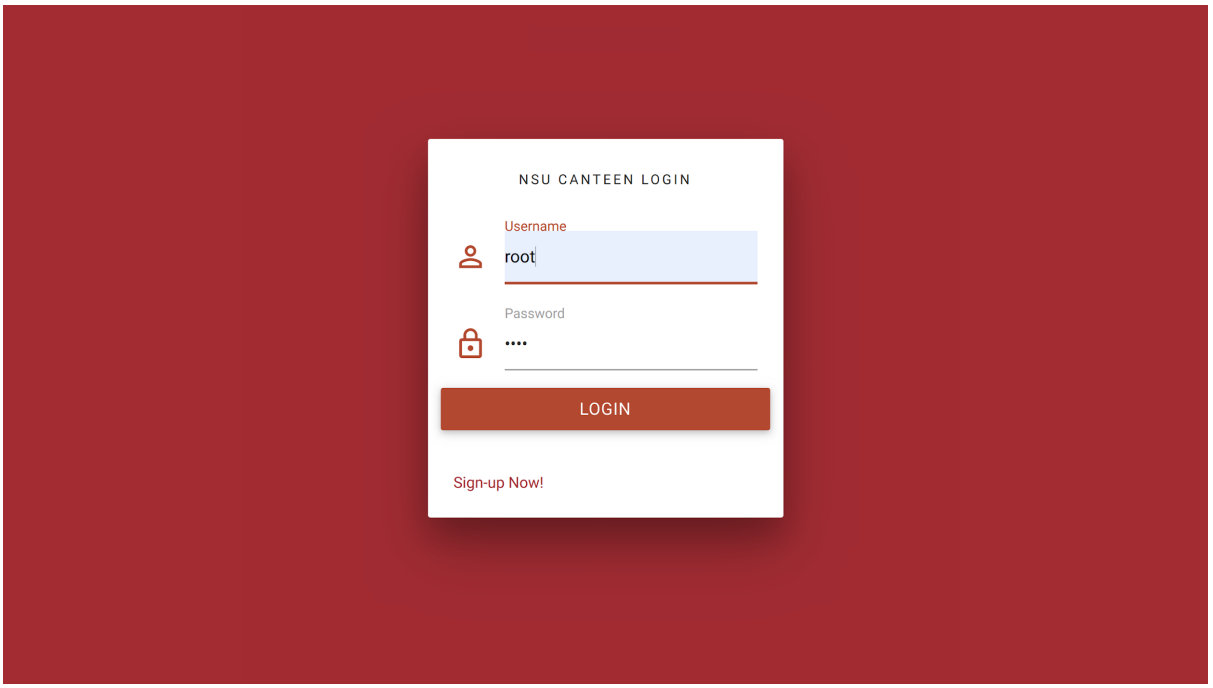


Figure 5.5: Admin Login

Step - 2: Choose "Yet to be delivered" from the left-hand side menu

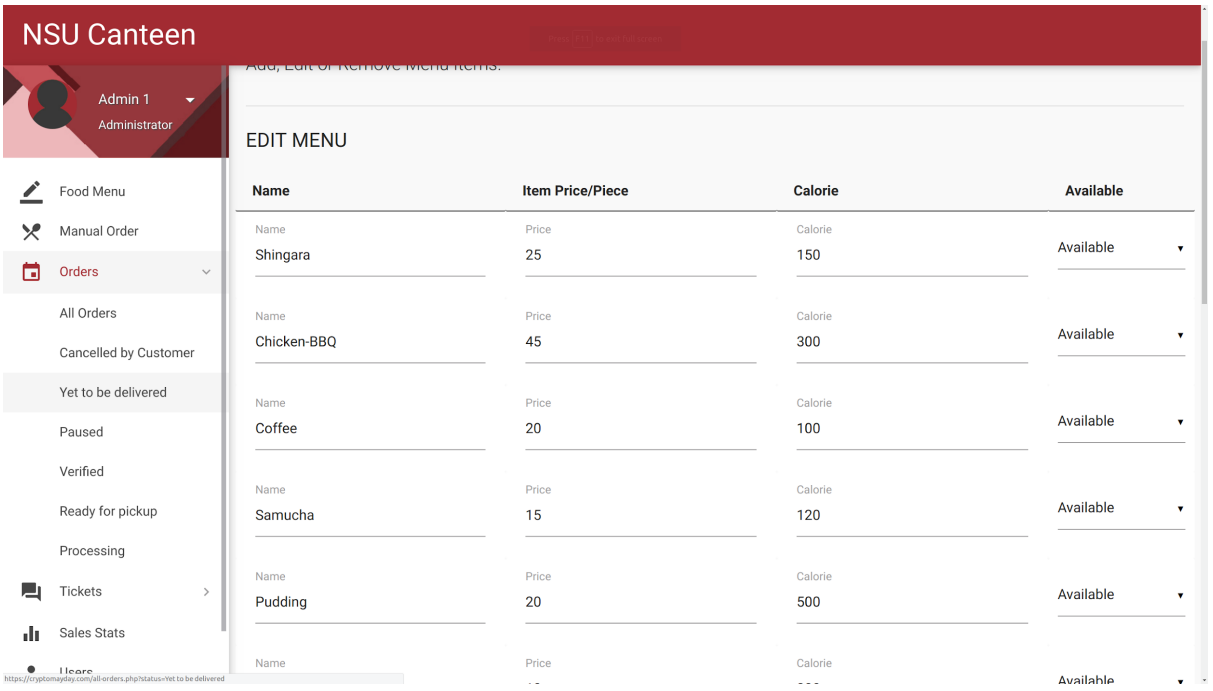


Figure 5.6: Choose Yet to be delivered

Step - 3: Select “Ready for pickup” option and hit “Change Status”

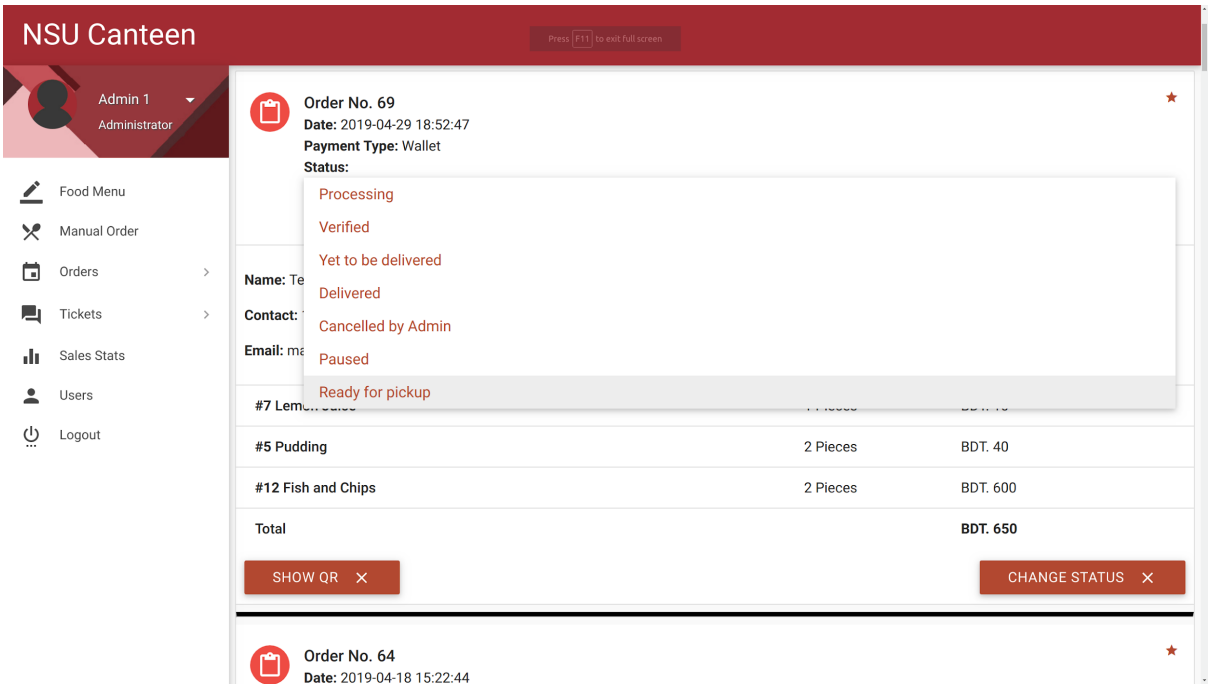


Figure 5.7: Select Ready for Pickup Status

Step - 4: Instant notification has gone to the client, hit “Show QR” when the client comes to the terminal.

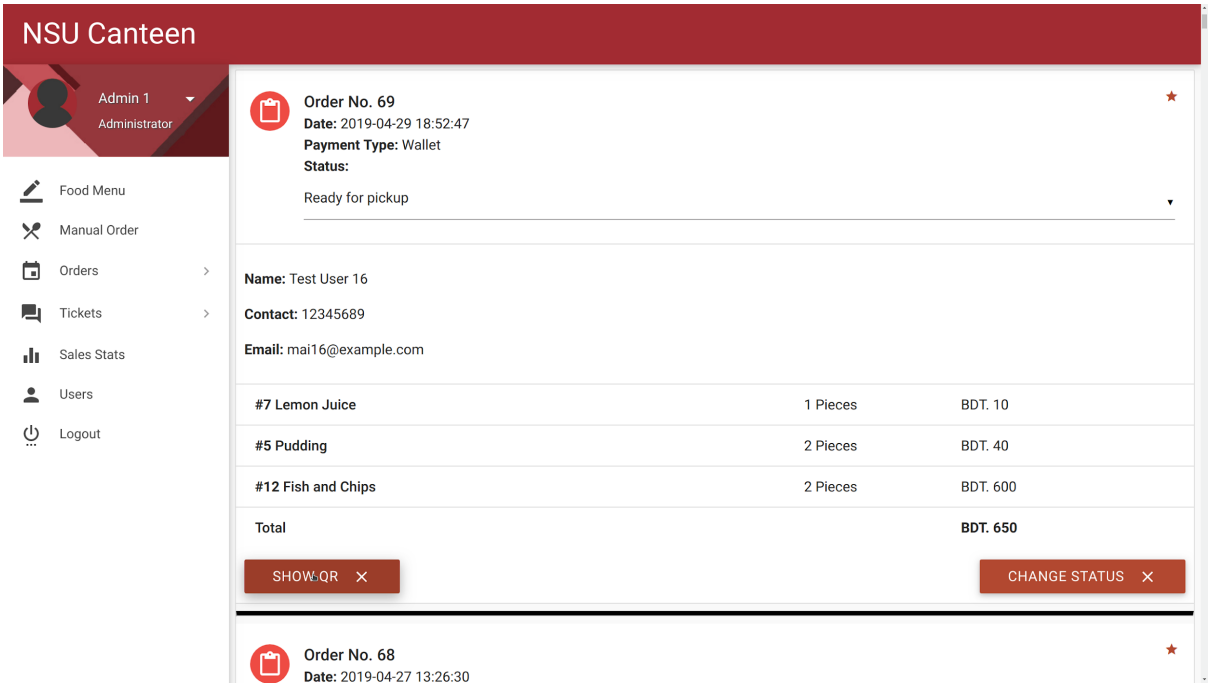


Figure 5.8: Status changed to Ready for Pickup

Step - 5: QR is automatically generated. Wait for client verification through their device.

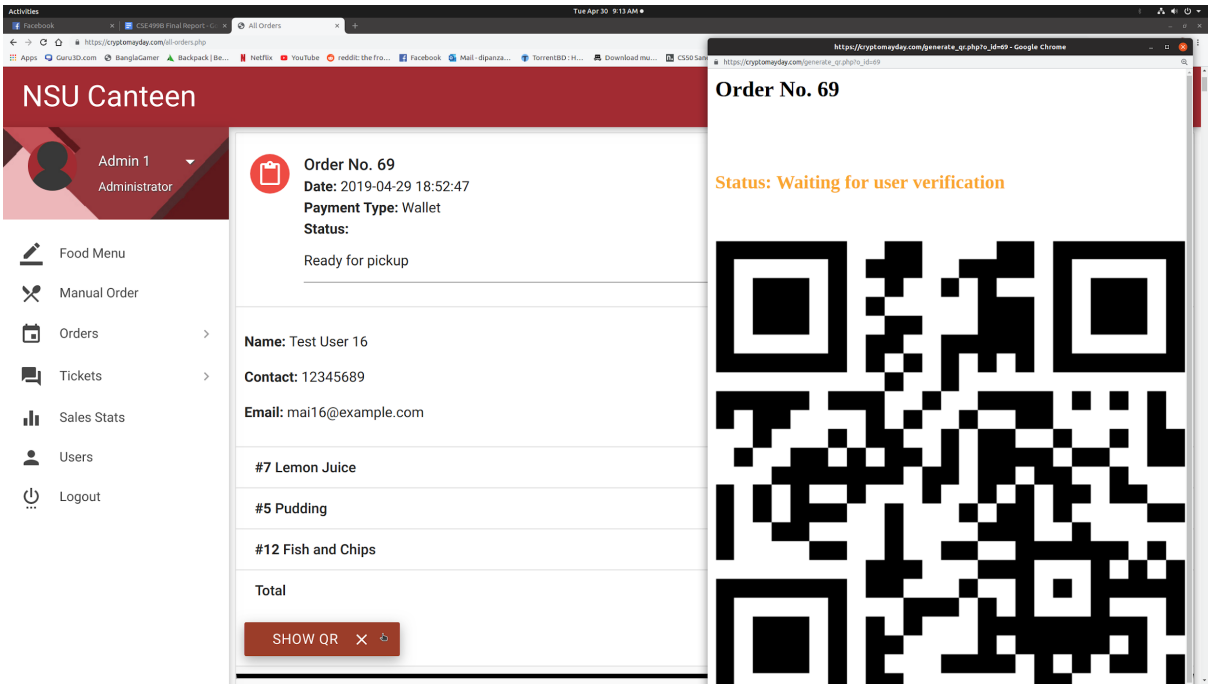


Figure 5.9: QR Generated to Display

Step-6: Order is verified, the employee can hand over food to the client.

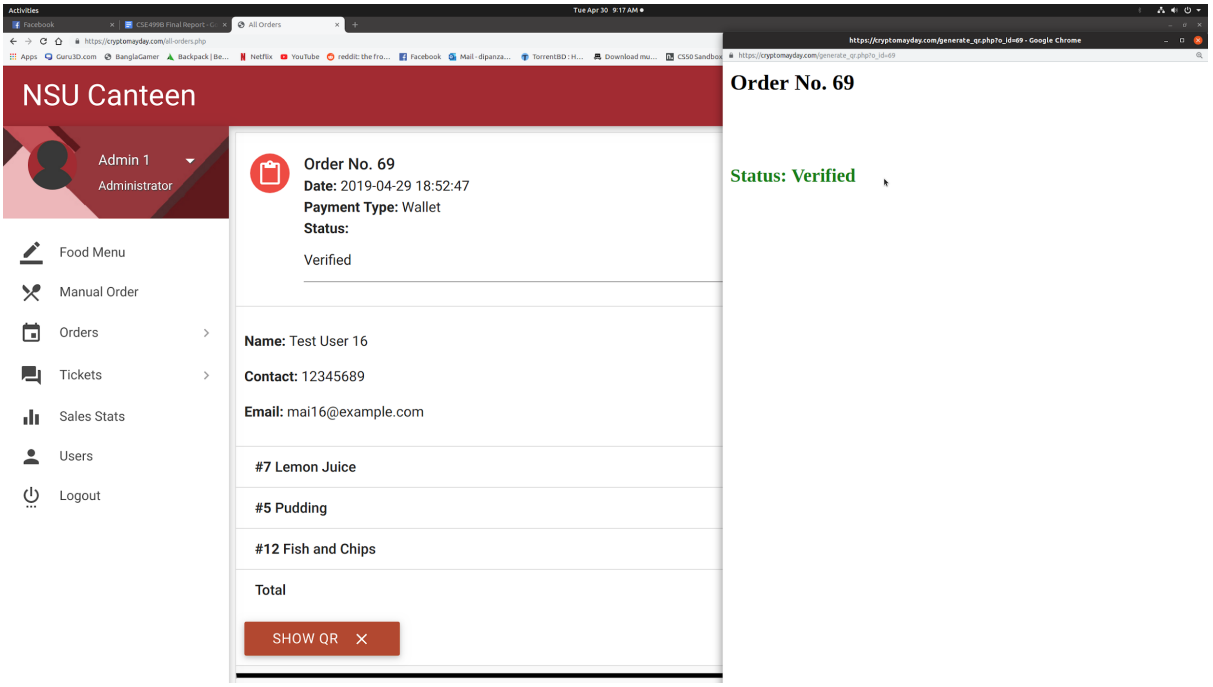


Figure 6.0: QR Verified on Display

7.4 Summary

In this chapter, we have explained in a step by step manner on how to get through the actual interface when using the application for a live demonstration. The most important part of catering to the canteen service has been shown from both the client and administrator perspectives. This will give a finer understanding of what to expect when a typical usage scenario is initiated.

CHAPTER 8

APPLICATION BUILDING BLOCKS

8.1 Introduction

In this chapter, we will introduce all the programming languages needed and the expertise involved in using the software to build such a system. We have used all open source software as part of the design scheme in order to avoid any costly penalties that come with commercial software. The technical skills involved in developing this application required the use of many different kinds of technologies for which an outline will be given in the next few sections.

8.2 Programming Languages

- **HTML5 (HyperText Markup Language)**

  HTML5 is the latest evolution of the standard that defines HTML. The term represents two different concepts. It is a new version of the language HTML, with new elements, attributes, and behaviors, and a larger set of technologies that allows the building of more diverse and powerful Web sites and applications. This set is sometimes called HTML5 & friends and often shortened to just HTML5.
  https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

- **CSS3 (Cascading Style Sheet)**

  CSS3 is the latest evolution of the *Cascading Style Sheets* language and aims at extending CSS2.1. It brings a lot of long-awaited novelties, like rounded corners, shadows, gradients, transitions or animations, as well as new layouts like multi-columns, flexible box or grid layouts. Experimental parts are vendor-prefixed and should either be avoided in production environments or used with extreme caution as both their syntax and semantics can change in the future.
  https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3

- **JS (Javascript)**

  JavaScript (**JS**) is a lightweight interpreted or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based,

multi-paradigm, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. Read more about JavaScript.
https://developer.mozilla.org/en-US/docs/Web/JavaScript

- **PHP 7.2 (Hypertext Preprocessor)**

  PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.
  https://www.php.net/manual/en/intro-whatis.php

- **MySQL (Structured Query Language)**

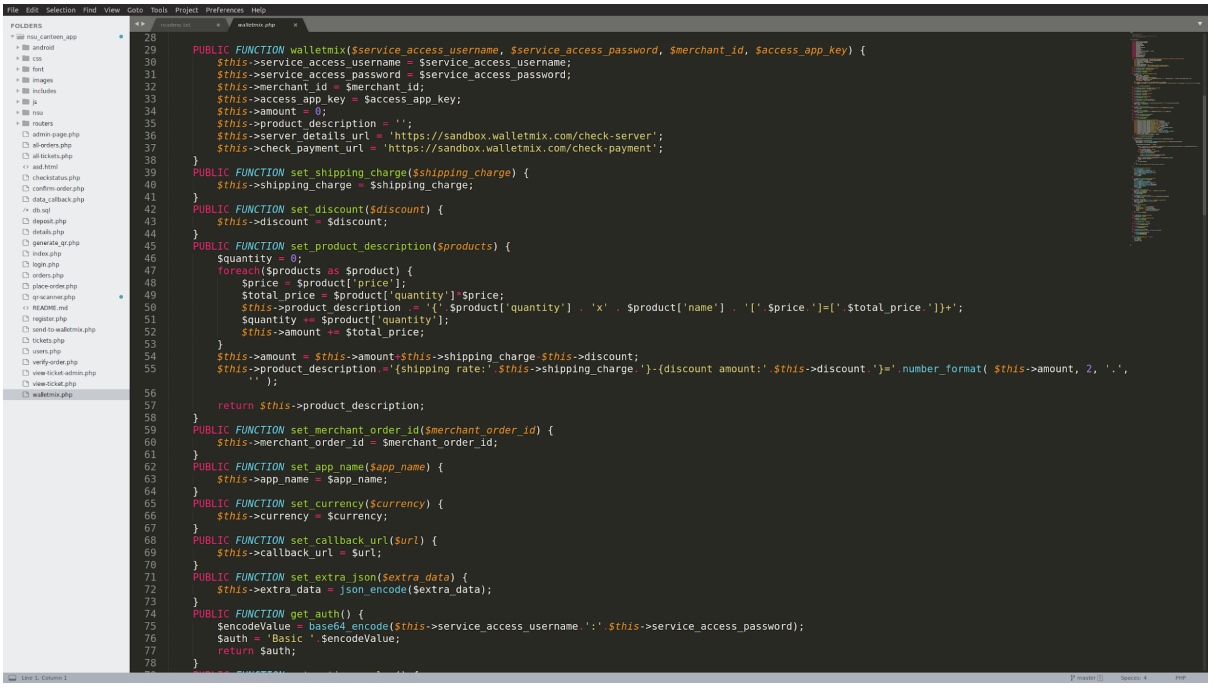
  MySQL is a database management system. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.
  https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

- **Java**

  Java is a programming language that produces software for multiple platforms. When a programmer writes a Java application, the compiled code (known as bytecode) runs on most operating systems (OS), including Windows, Linux, and Mac OS. Java derives much of its syntax from the C and C++ programming languages.
  https://www.techopedia.com/definition/3927/java

8.3 Technologies Used

- **Sublime Text 3 (Code Editor)**



```
    PUBLIC FUNCTION walletmix($service_access_username, $service_access_password, $merchant_id, $access_app_key) {
        $this->service_access_username = $service_access_username;
        $this->service_access_password = $service_access_password;
        $this->merchant_id = $merchant_id;
        $this->access_app_key = $access_app_key;
        $this->amount = 0;
        $this->product_description = '';
        $this->server_details_url = 'https://sandbox.walletmix.com/check-server';
        $this->check_payment_url = 'https://sandbox.walletmix.com/check-payment';
    }
    PUBLIC FUNCTION set_shipping_charge($shipping_charge) {
        $this->shipping_charge = $shipping_charge;
    }
    PUBLIC FUNCTION set_discount($discount) {
        $this->discount = $discount;
    }
    PUBLIC FUNCTION set_product_description($products) {
        $quantity = 0;
        foreach($products as $product) {
            $price = $product['price'];
            $total_price = $product['quantity']*$price;
            $this->product_description .= '{'.$product['quantity'] . 'x' . $product['name'] . '['.$price.']=['.$total_price.']}+';
            $quantity += $product['quantity'];
            $this->amount += $total_price;
        }
        $this->amount = $this->amount+$this->shipping_charge-$this->discount;
        $this->product_description.='{shipping rate:'.$this->shipping_charge.'}-{discount amount:'.$this->discount.'}='.number_format( $this->amount, 2, '.',
            '' );

        return $this->product_description;
    }
    PUBLIC FUNCTION set_merchant_order_id($merchant_order_id) {
        $this->merchant_order_id = $merchant_order_id;
    }
    PUBLIC FUNCTION set_app_name($app_name) {
        $this->app_name = $app_name;
    }
    PUBLIC FUNCTION set_currency($currency) {
        $this->currency = $currency;
    }
    PUBLIC FUNCTION set_callback_url($url) {
        $this->callback_url = $url;
    }
    PUBLIC FUNCTION set_extra_json($extra_data) {
        $this->extra_data = json_encode($extra_data);
    }
    PUBLIC FUNCTION get_auth() {
        $encodeValue = base64_encode($this->service_access_username.':'.$this->service_access_password);
        $auth = 'Basic '.$encodeValue;
        return $auth;
    }
```

Figure 6.1: Sublime Text 3

Sublime Text is a proprietary cross-platform source code editor with a Python application programming interface. It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically community-built and maintained under free-software licenses.

https://www.sublimetext.com/

- **XAMPP (Database Interface)**

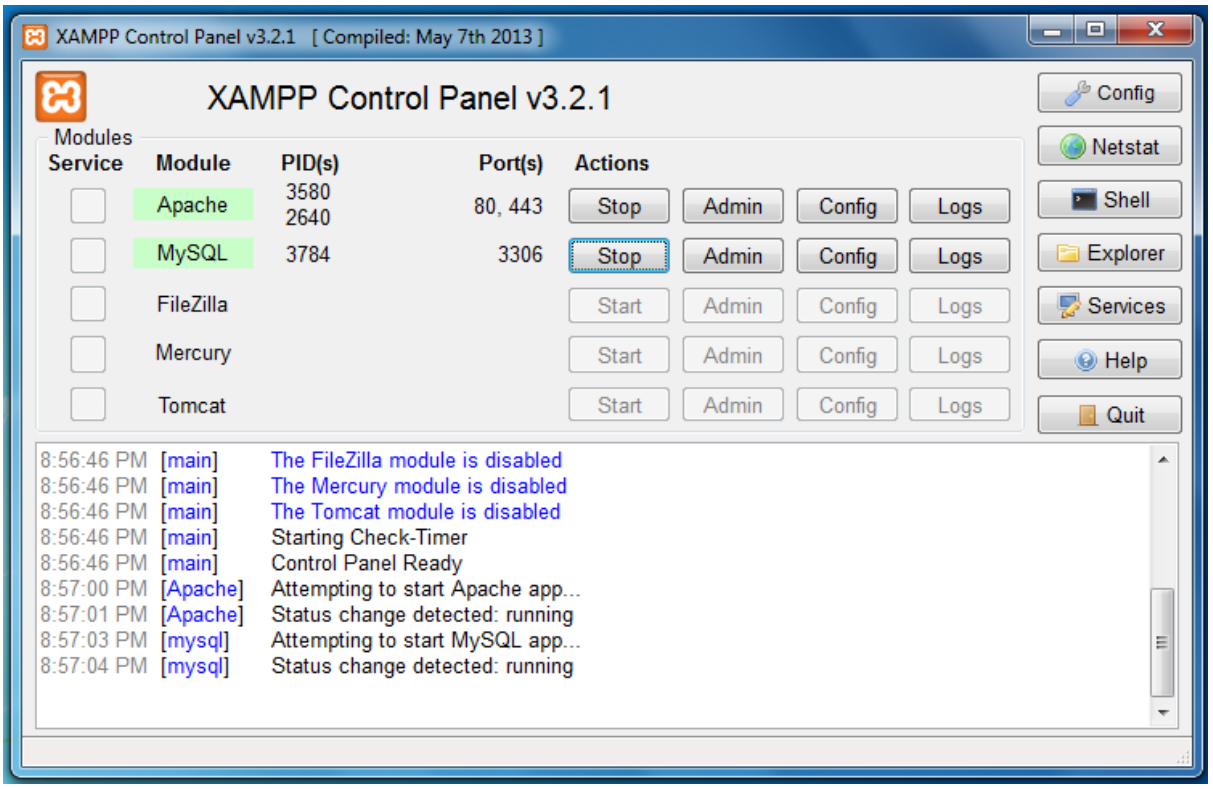


Figure 6.2: XAMPP Control Panel

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.

https://www.apachefriends.org/index.html

- **Android Studio**

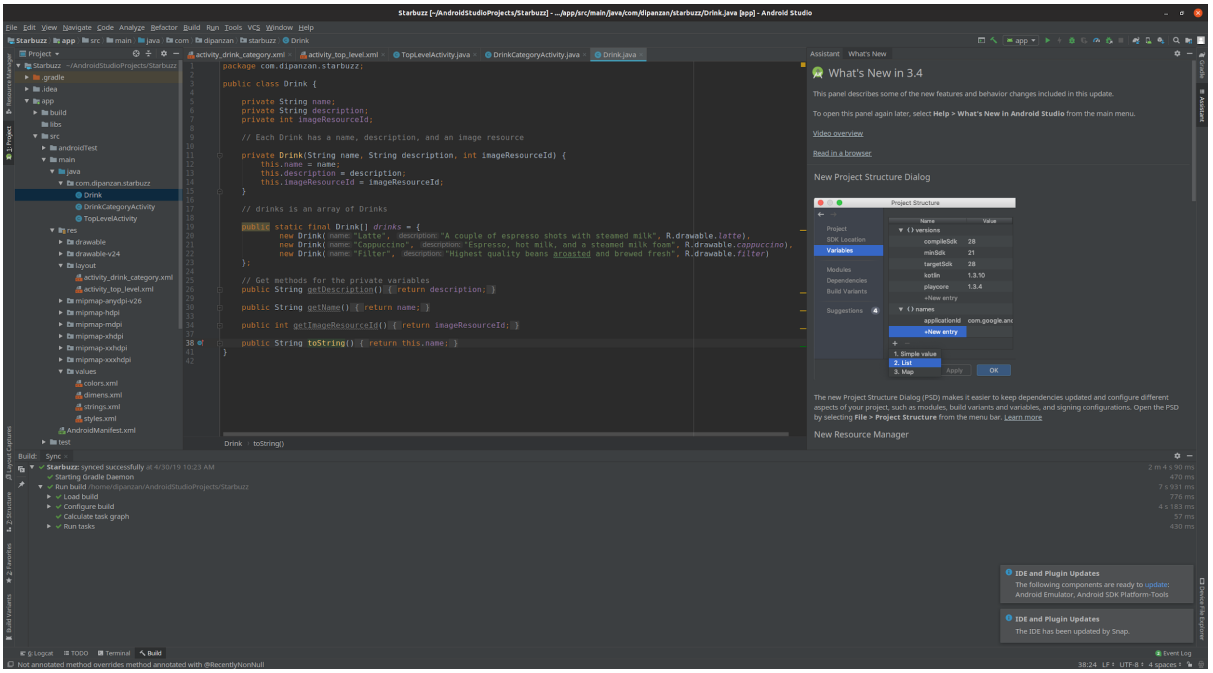


Figure 6.3: Android Studio

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems.

https://developer.android.com/studio

8.4 Requirements

In order to use this software and technologies and the live application that was developed, certain requirements must be met which are as follows:

1. Internet Connectivity
2. Web Hosting (Servers)
3. Domain
4. Computer Hardware (for Servers)
5. Operating System (Windows/Linux)
6. Smartphones (Android / iOS)

8.5 Summary

In this chapter, we have introduced the technical aspects and software that were needed to complete the working application. All these technologies that were used in development are available free of cost and have been of monumental help to complete the project to fruition. The requirements sections put emphasis on the actual hardware and combination software needed for a live setup.

CHAPTER 9

PROJECT TIMELINE

9.1 Introduction

In this chapter, we will talk about the project's timeline and how everything was planned from the beginning to the end. This includes all the work and effort from members of the project and their contribution. The developmental phase started from the course's perspective CSE499A, where we have proposed an initial design schematic for the project. The second part of the project CSE499B was the actual development phase during which the application was created and tested. The features were added on a part by part basis as it passed functionality requirements.

9.2 Work Breakdown

The work breakdown is listed below from the project's birth till the end using Gantt charts. All the scheduling information is listed and the structure of the development phase can be visualized with the aid of these charts.

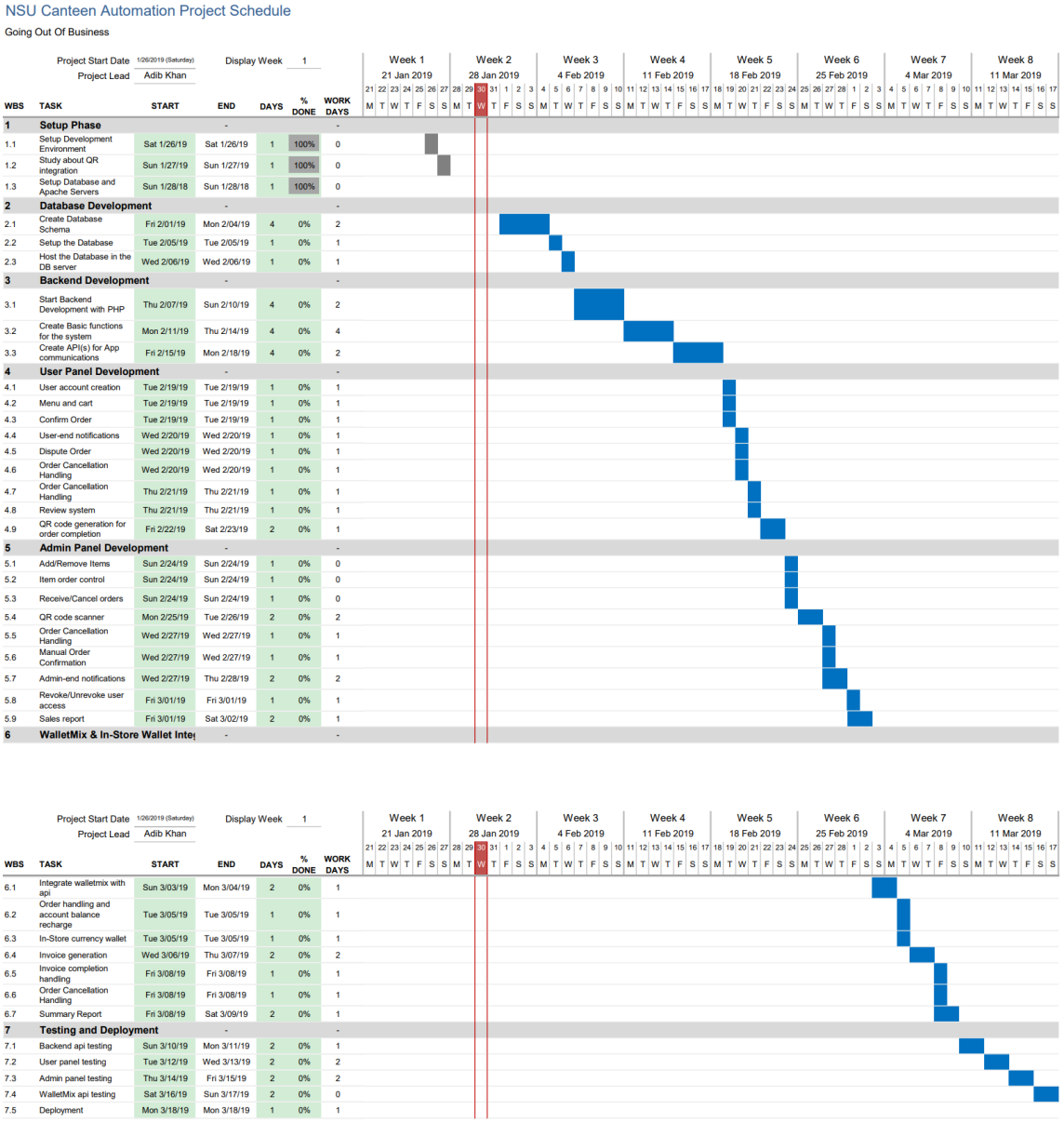
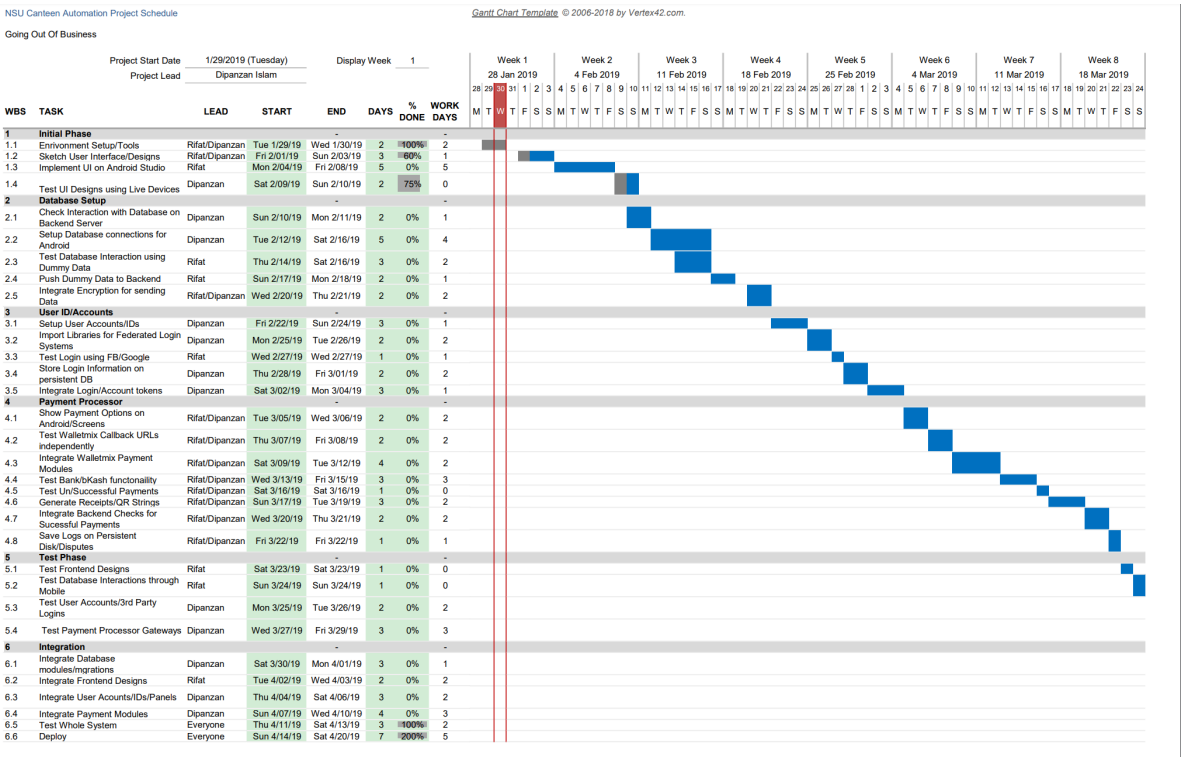


Figure 6.4: Project Timeline Part 1



WBS	TASK	LEAD	START	END	DAYS	% DONE	WORK DAYS
1	**Initial Phase**			-			-
1.1	Enrivonment Setup/Tools	Rifat/Dipanzan	Tue 1/29/19	Wed 1/30/19	2	100%	2
1.2	Sketch User Interface/Designs	Rifat/Dipanzan	Fri 2/01/19	Sun 2/03/19	3	60%	1
1.3	Implement UI on Android Studio	Rifat	Mon 2/04/19	Fri 2/08/19	5	0%	5
1.4	Test UI Designs using Live Devices	Dipanzan	Sat 2/09/19	Sun 2/10/19	2	75%	0
2	**Database Setup**			-			-
2.1	Check Interaction with Database on Backend Server	Dipanzan	Sun 2/10/19	Mon 2/11/19	2	0%	1
2.2	Setup Database connections for Android	Dipanzan	Tue 2/12/19	Sat 2/16/19	5	0%	4
2.3	Test Database Interaction using Dummy Data	Rifat	Thu 2/14/19	Sat 2/16/19	3	0%	2
2.4	Push Dummy Data to Backend	Rifat	Sun 2/17/19	Mon 2/18/19	2	0%	1
2.5	Integrate Encryption for sending Data	Rifat/Dipanzan	Wed 2/20/19	Thu 2/21/19	2	0%	2
3	**User ID/Accounts**			-			-
3.1	Setup User Accounts/IDs	Dipanzan	Fri 2/22/19	Sun 2/24/19	3	0%	1
3.2	Import Libraries for Federated Login Systems	Dipanzan	Mon 2/25/19	Tue 2/26/19	2	0%	2
3.3	Test Login using FB/Google	Rifat	Wed 2/27/19	Wed 2/27/19	1	0%	1
3.4	Store Login Information on persistent DB	Dipanzan	Thu 2/28/19	Fri 3/01/19	2	0%	2
3.5	Integrate Login/Account tokens	Dipanzan	Sat 3/02/19	Mon 3/04/19	3	0%	1
4	**Payment Processor**			-			-
4.1	Show Payment Options on Android/Screens	Rifat/Dipanzan	Tue 3/05/19	Wed 3/06/19	2	0%	2
4.2	Test Walletmix Callback URLs independently	Rifat/Dipanzan	Thu 3/07/19	Fri 3/08/19	2	0%	2
4.3	Integrate Walletmix Payment Modules	Rifat/Dipanzan	Sat 3/09/19	Tue 3/12/19	4	0%	2
4.4	Test Bank/bKash functonality	Rifat/Dipanzan	Wed 3/13/19	Fri 3/15/19	3	0%	3
4.5	Test Un/Successful Payments	Rifat/Dipanzan	Sat 3/16/19	Sat 3/16/19	1	0%	0
4.6	Generate Receipts/QR Strings	Rifat/Dipanzan	Sun 3/17/19	Tue 3/19/19	3	0%	2
4.7	Integrate Backend Checks for Sucessful Payments	Rifat/Dipanzan	Wed 3/20/19	Thu 3/21/19	2	0%	2
4.8	Save Logs on Persistent Disk/Disputes	Rifat/Dipanzan	Fri 3/22/19	Fri 3/22/19	1	0%	1
5	**Test Phase**			-			-
5.1	Test Frontend Designs	Rifat	Sat 3/23/19	Sat 3/23/19	1	0%	0
5.2	Test Database Interactions through Mobile	Rifat	Sun 3/24/19	Sun 3/24/19	1	0%	0
5.3	Test User Accounts/3rd Party Logins	Dipanzan	Mon 3/25/19	Tue 3/26/19	2	0%	2
5.4	Test Payment Processor Gateways	Dipanzan	Wed 3/27/19	Fri 3/29/19	3	0%	3
6	**Integration**			-			-
6.1	Integrate Database modules/mgrations	Dipanzan	Sat 3/30/19	Mon 4/01/19	3	0%	1
6.2	Integrate Frontend Designs	Rifat	Tue 4/02/19	Wed 4/03/19	2	0%	2
6.3	Integrate User Acounts/IDs/Panels	Dipanzan	Thu 4/04/19	Sat 4/06/19	3	0%	2
6.4	Integrate Payment Modules	Dipanzan	Sun 4/07/19	Wed 4/10/19	4	0%	3
6.5	Test Whole System	Everyone	Thu 4/11/19	Sat 4/13/19	3	100%	2
6.6	Deploy	Everyone	Sun 4/14/19	Sat 4/20/19	7	200%	5

Figure 6.5: Project Timeline Part 2

9.3 Financial Plan

We managed to get information from the internet and by asking local sources here in Dhaka about the cost of installing restaurant management systems. Most of these systems use POS for their operation with a cashier. These systems in retail use go for around 50, 000 to 70, 000 BDT and an extra charge for software.

We used our own systems (Laptops/Desktop) for a testbed and for the actual application development. The extra cost that was incurred was due to buying a Domain Name and Secure Socket Layer (SSL) certificate which cost around ~4000 BDT in total.

9.4 Feasibility Study

A feasibility study is an analysis that takes all of a project's relevant factors into account including economic, technical, legal, and scheduling considerations to ascertain the likelihood of completing the project successfully. A feasibility study includes an estimate of the level of expertise required for a project and who can provide it, quantitative and qualitative assessments of other essential resources, identification of critical points, a general timetable, and a general cost estimate.

With that in mind, we emphasize on the major decisions using the points below:

- Ease of use
    - ➢ How users will perceive the use of the application in their day to day activities within the canteen.
    - ➢ ease of use.
- Security implications
    - ➢ How will users perceive the use of a device for payments as opposed to cash?
    - ➢ Trusting the system to store virtual currency.
- Cost
    - ➢ How will it reduce costs in terms of maintenance?
    - ➢ Will it introduce a burden on the current system?
- Time
    - ➢ Will this reduce the need for a queue regarding payments.
    - ➢ Will this save overall time from food ordering and receiving of said orders.
    - ➢ Less busy terminals.

Feasibility Analysis:

1. The need to appoint a project/team leader.
2. Evaluate proposed systems against one another.
3. Carefully distinguish the unique properties of the proposed system.
4. Evaluate the performance stats of each system.
5. Take into consideration the costs of each system.
6. Compare the performance versus cost ratio.
7. Choose the best system from the comparison.
8. Prepare a detailed outline/schematic of the system for submission to authority.

Technical Feasibility

The technical feasibility study is the complete study of the project in terms of input, processes, output, fields, programs and procedures. It is a very effective tool for long term planning and troubleshooting. The technical feasibility study should most essentially support the financial information of an organization.

Major questions that need to ask when concerning technical feasibility are:

1. Is it possible to carry out the project/development using currently available resources(hardware/software)?
2. Is the system open to upgrades after development?
3. Will the system accept new features in the future.

The feasibility choices ultimately boil down to the software that is being developed and how it will be perceived by its end users. The application must be robust, and not prone to failures during operation and must present a working interface that is intuitive and gets the job done.

For the user interface, the following points must be considered:

1. Having a graphical user interface (GUI) which enables the use of this system without much technical knowledge.
2. Is open to upgrades regarding extensions (change to UI).
3. Must represent the organization for which the application is being developed.
4. Platform agnostic.

5. Have good credibility when using the software. Provide past logs and receipts of transactions.
6. Easy maintenance.

For the backend interface, the following points must be considered:

1. Robust in nature.
2. Handle multiple points of requests from clients.
3. Security regarding transactions for accountability.
4. Keeping past logs in secured disks for future inspection.
5. The operating system is independent.
6. Easy and efficient means of data extraction through database dumps.
7. Compatibility with current and future frontend interfaces (loosely coupled).
8. Ease of installation and upgrades.

For the points explained above for user and backend interfaces, the choice of programming languages and technologies has been discussed in the previous chapter with technical definitions and explanations.

Economic Feasibility

Economic feasibility, therefore, is the aggregation of all the cost justifications made when designing the project and one of the major concerns is a cost-benefit analysis. This section further discusses if the justifications made were appropriate concerning the organization for which the application was developed.

The questions that were ultimate drivers for economic feasibility are the financial standpoints that were asked in the design phase and the following points best describe them in detail:

1. The cost that is incurred to run a full system diagnostic.
2. The total cost of the application both hardware and software combined.
3. The benefits of our system over the traditional system in place in terms of cost.
4. The number of resources saved using the developed system (i.e receipt papers saved due to electronic receipts and virtual payments).
5. The funding needed to incorporate such a development scheme.

Operational Feasibility

The operational feasibility describes what changes it will bring to the current system if deployed and is closely related to the company's administration department and their organizational aspects.

The following points are noteworthy regarding operational feasibility:

1. What will the application bring to the table?
2. Which current systems inside the organization are affected.
3. How much new training will be required in order to use the new system.
4. How will the employees perceive in using the new application if deployed
5. Will it still maintain the organization's ability to perform its core functions.

With these points in mind, we believe the newer system is an upgrade from the old style. Plus anyone familiar with using a standard desktop/PC environment can operate the new system efficiently.

Schedule Feasibility

During the development phase, it is critical to value the extent of what is available in terms of development time. It should include all the components that were developed within an acceptable amount of time without disrupting sales or hurting company policies. There should be as less delay as possible when migrating to the new system and efficient setup up to speed.

9.5 Summary

This chapter concludes with the project’s timelines that were initiated during the initial design phase and ended with the actual live prototype. The feasibility study talks about the importance of deployment into the actual organization replacing the old system and how that might affect the surrounding structures involving humans, machines and their interactions.

CHAPTER 10

FUTURE WORK

10.1 Introduction

In this chapter we will discuss the future scopes of the system that we have in mind, things we like to add and modify in our existing system for better function and to sustain more users.

10.2 Future Scope of Work

Our main objective is to develop the system into a more improved one which will be easier to use and also recommend healthy food items. We plan to do this by implementing a better user interface and implement machine learning which will suggest a user food item based on their previous choices. The application will also show the exact calorie a certain food item contains, which will help users not to exceed a certain amount of calorie they must not consume in order to stay healthy and fit. For now, our application runs on android but in the future, it will be cross-compatible running both on ios/android platforms.

We can also update the signup section by including the popular social media site credentials for login purposes, such as Facebook or Twitter's profile. We would like to implement a chat option where a customer can chat directly to canteen employees regarding their order or if they want to complain about a certain order. We will also leave a section where users can ask for features they want to see in the future.

10.3 Summary

In this chapter, we talked about the possible features and modifications we would like to make in our existing system. We believe by doing so more users will be using our applications. We want to make the environment inside the canteen stress-free and more organized and attain a better performance by saving time per serving customer.

CHAPTER 11

DESIGN IMPACT

11.1 Introduction

In this chapter, we will be focusing on the various impacts that our application is able to generate in the real world setting. This is important to understand when the application is deployed for which the necessary effects must be discussed in depth so that potential users of this software are aware.

11.2 Environmental Impact

By introducing such an application the whole scenario inside the canteen can change as there will be fewer people standing in queues and roaming around clueless. The environment will be a lot suitable to all as the canteen will be less noisy and at the same time, it will be less congested.

11.3 Economic Impact

Previously there would have been more employees working because of the greater number of people standing, but now after the launch of the new system the number of queues will be reduced and the number of employees required will be relatively less. The employees who were there to collect cash will not be needed anymore, either they will lose their job or they will be relocated to another part of the canteen to work in a different sector. This will save the company a lot of money and hassle at the same time.

11.4 Social Impact

Talking to a few people who are regulars at the canteen greatly like the idea of the new system as they have faced the problems mentioned above and now see an effective solution to the daily problems in the canteen. Some were even greatly excited when they heard they can order from anywhere inside the campus, intriguing as it sounds, an order can be placed even from outside the campus. The other bright side is that occasionally some ends up in arguments while standing in lines, this would be eliminated while the new system is in effect. Food lovers would not have to feel uncomfortable standing in many lines.

11.5 Sustainability

Our system is not just a working system but a system which is fast and effective at the same time, reducing most of the stressful act while buying in the canteen. Ordering is really easy from the application and even depositing money to the account is easier as well. Anyone who heard or used our system left us with compliments saying our system is stable and effective. Based upon the tests we ran and the impressions we get from people its right to say our system is sustainable.

11.6 Summary

This chapter briefly covers the different aspects of our system and why it is so effective compared to the current system. Taking the aspects into consideration and the benefits our system provides we can conclude that our system is a good choice to be implemented in such scenarios.

CHAPTER 12

COMPLIANCE WITH IEEE STANDARDS

12.1 Introduction

In this chapter, we look into the aspects and discuss the consistency of our system with the different standards out there. Among the standards, we will look into mainly the IEEE standards, US standards, and European standards.

12.2 Compliance with IEEE standard

The *IEEE Transactions on Software Engineering* is interested in well-defined theoretical results and empirical studies that have a potential impact on the construction, analysis, or management of software. The scope of these transactions ranges from the mechanisms through the development of principles to the application of those principles to specific environments. We thoroughly analyzed our system to see if it meets some of the requirements mentioned in the standards, it does meet the specified environments. Specific topic areas include: a) development and maintenance methods and models, e.g., techniques and principles for the specification, design, and implementation of software systems, including notations and process models; b) assessment methods, e.g., software tests and validation, reliability models, test and diagnosis procedures, software redundancy and design for error control, and the measurements and evaluation of various aspects of the process and product; c) software project management, e.g., productivity factors, cost models, schedule and organizational issues, standards; d) tools and environments, e.g., specific tools, integrated tool environments including the associated architectures, databases, and parallel and distributed processing issues; e) system issues, e.g., hardware-software trade-off; and f) state-of-the-art surveys that provide a synthesis and comprehensive review of the historical development of one particular area of interest. Most of the factors mentioned above are implemented in our system and it's okay to say that our system meets the IEEE standards.

12.3 Compliance with US standards

According to ANSI copyright software should only be used for information only, or in forms which do not mandate particular implementations of the standard. Implementation of object code should never be included in a standard as a normative requirement. While ANSI opposes the use of software standards to mandate particular implementations and believes that the use of software in standards should be avoided to the extent possible. ANSI thinks the inclusion of some software may benefit the competitive market. Examples of such software could include:

1. Data structure definitions
2. Pseudo codes
3. ASN.1 structure definitions
4. ABNF
5. Sample programming instructions provided solely for conformance testing purposes.

Most of the principles mentioned in the ANSI standard are implemented in our system.

12.4 Summary

This chapter concludes by going over the definitions of the strict standards imposed by the resolutions and how the project adheres to these notions to achieve better quality and hold up to industry standards.

CHAPTER 13

RESULTS

13.1 Introduction

This chapter will talk about the application's usefulness in a real-world setting. In chapter 7 we have already seen a detailed guide on how to operate the software, so the results will be outlined in the next section.

13.2 Results Achieved

During the demonstration of the project inside our classroom, we have found that it takes on average 1-2 minutes to order through the application and confirm the payment in about 1 minute depending on the food serving queue.

The following points further illustrate the capabilities that were evident:

- Ease of use, the application can be operated easily without much technical knowledge.
- The application has the capability for secure transactions.
- Connectivity, it can be accessed from anywhere if the Internet is available.
- The ability to store all transaction and receipt data for future viewing.
- Secured accounts.
- Faster operation compared to traditional methods.

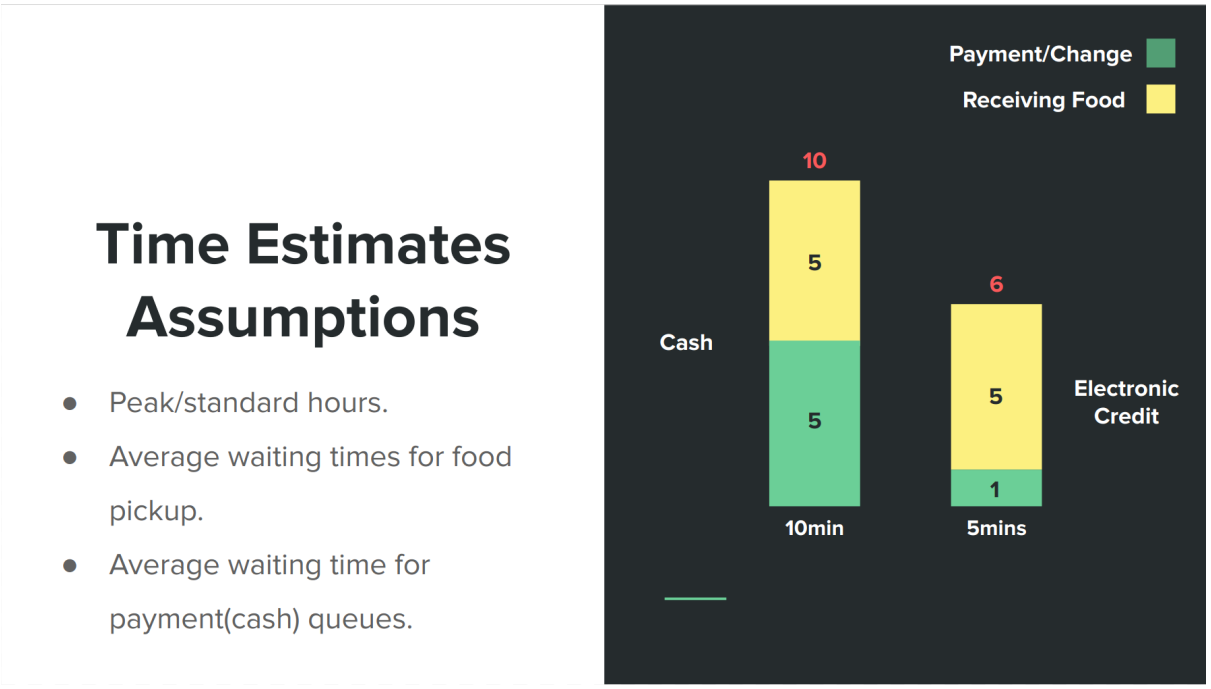


Figure 6.6: Time Estimates

13.3 Summary

This chapter concludes with the most important aspect of the application, that is how much time is saved over the current mode of operation which is manual.

CHAPTER 14

CONCLUSION

Though there has been a recent renovation at the NSU cafeteria, it still does not serve to be the expected hassle-free place everyone desires. There are still those whines coming from the regular students and staff members regarding the long standing queues. This has been a major complaint against the new overhaul of the beloved cafeteria and it is evident that the older way of getting food which was ”first come first serve” basis was more efficient at the cost of a disruptive atmosphere. The rationale behind this project is to design a system that can tackle the problems of a busy cafeteria using available hardware and software to provide a viable solution. The solution even though very simple requires a multitude of thought since security is a big concern when it comes to public expenditure through a computerized or electronic-based system. The main focus is to reduce the waiting time in an organized manner using a systemic approach. Since there is no way to test queue/waiting times using alternate approaches on a live setting, it will only be apparent after a working prototype is ready using a bare-bones design and crucial functions to test the system and validate its usefulness. The scope of this project aims to deliver a working product (mobile application) which will enable end users and canteen employees for smoother communication when making transactions. The application will hopefully provide a rich experience while ordering food ensuring reliable security. A big concern is to use as little resource as possible whilst trying to address the main problem and come out with a productive solution.

Thus we hope that our project “NSU Canteen Automation”, the application we have developed, makes the lives of students and faculties a bit easier and saves valuable time.

BIBLIOGRAPHY

[1] S. A. Curin, J. S. Vosko, E. W. Chan, O. Tsimhoni, “Reducing service time at a busy fast food restaurant on campus,” Proceedings of the Winter Simulation Conference, Orlando, FL, 2005, pp. 8.

[2] S. Nseir, N. Hirzallah and M. Aqel, “A secure mobile payment system using QR code,” 5th International Conference on Computer Science and Information Technology, Amman, 2013, pp. 111-114.

[3] R. Shriwas, N. Patel, A. Bherani, A. Khajone and M. Raut, “Touchscreen based ordering system for restaurants,” International Conference on Communication and Signal Processing, Melmaruvathur, 2014, pp. 1021-1024.

[4] X. Ju and Y. Wang, “Simulation and Improvement of Multiple Queue Multiple Serve System Based on Witness,” International Conference on Multimedia Technology, Ningbo, 2010, pp. 1-4.

[5] T. Ma, H. Zhang, J. Qian, X. Hu and Y. Tian, “The Design and Implementation of an Innovative Mobile Payment System Based on QR Bar Code,” International Conference on Network and Information Systems for Computers, Wuhan, 2015, pp. 435-440.

[6] R. Shafei, S. A. Rastad and A. Kamangar, “Effecting of electronic-tablet-based menu and its impact on consumer choice behavior (An empirical study in Iranian restaurant),” 10th International Conference on e-Commerce in Developing Countries: with focus on e-Tourism (ECDC), Isfahan, 2016, pp. 1-8.

[7] Ching-Su Chang, Che-Chen Kung and Tan-Hsu Tan, “Development and implementation of an e-restaurant for customer-centric service using WLAN and RFID technologies,” International Conference on Machine Learning and Cybernetics, Kunming, 2008, pp. 3230-3235.

[8] B. K. Mishra, B. S. Choudhary and T. Bakshi, “Touch based digital ordering system on Android using GSM and Bluetooth for restaurants,” Annual IEEE India Conference (INDICON), New Delhi, 2015, pp. 1-5.

[9] A. T. Purnomo, Y. S. Gondokaryono and C. Kim, “Mutual authentication in securing mobile payment system using encrypted QR code based on Public Key Infrastructure,” 6th International Conference on System Engineering and Technology (ICSET), Bandung, 2016, pp. 194-198.