**A SMART APPLICATION FOR CAFETERIA AUTOMATION**



**PROJECT ID-**

**PROJECT ADVISOR: Usman Rafi**

**Submitted By**

Waqas Hussain Bsf1601492

Bareera Anam Bsf1601497

Muhammad Abu bakkar Bsf1601499

Tamoor Akhtar Bsf1601507

**University of Education**

**A SMART APPLICATION FOR CAFETERIA AUTOMATION**

**BS INFORMATION TECHNOLOGY 2016-2020**

A project submitted in partial fulfillment of the requirements for the award of the degree of

BS in information Technology.

**UNIVERSITY OF EDUCATION LAHORE**

**CAMPUS FAISALABAD**

February 2020

“I hereby declare that I have read this project documentation and in my opinion this project is sufficient in terms of scope and quality for the award of the degree of BS in Information Technology.”

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
Project Primary Supervisor Project: Examiner  
Name: Usman Rafi Name:

Designation: Lecturer Designation:  
University of Education. University of Education.

**DECLARATION**

We declare that this project title entitled “**A smart Application for Cafeteria Automation**” is the result of our own research and development except as cited in the references. This project has not been accepted for any degree and is not concurrently submitted in candidate for any other degree. At any time if our statement is found to be incorrect even afterwards of BS in Information Technology the university has the right to withdraw my BS in Information Technology degree.

Signature: Signature:

Name: WAQAS Name: M.TAMOOR

Date: Date:

Signature: Signature:

Name: BAREERA ANAM Name: M.ABUBAKKAR

Date: Date:

### PLAGIARISM UNDERTAKEN

We solemnly declare that project work presented in this documentation entitles **“A Smart Application for Cafeteria Automation**” is solely our work with no significant contribution from any other person. Small contribution/help wherever taken has been acknowledged and that complete project has been written by us.

We understand that zero tolerance policy of the HEC and University of Education, Lahore towards plagiarism. Therefore, we as an author of the above titled project declare that no portion of our project documentation and any material used as reference is properly referred/ cited.

We undertake that of we found guilty of any formal plagiarism in the above titled project even after award of BS degree, the University reserve the rights to withdraw our BS degree and that HEC and the University has the right to publish our names on the HEC/University Website on which names of students are place who submitted plagiarized projects.

Signature: Signature:

Name: WAQAS Name: M.TAMOOR

Date: Date:

Signature: Signature:

Name: BAREERA ANAM Name: M.ABU-BAKKAR

Date: Date:

|  |
| --- |
| **CERTIFICATE OF APPROVAL**  This is to certify that the project work presented in this documentation entitled, “**A Smart Application for Cafeteria automation”**, was conducted by **“Waqas Hussain”, “Muhammad Abubakkar”, “Bareera Anam”, “Tmoor Akhtar”**, under the supervision of **“ Usman Rafi”**. No part of this project has been submitted anywhere else for any degree. This project is submitted to the **“University of Education/Faisalabad campus ”** is partial fulfillment of the requirements of the degree of BS in Information Technology. |

Signature: Signature:

Name: WAQAS Name: M.TAMOOR

Date: Date:

Signature: Signature:

Name: BAREERA ANAM Name:M.ABUBAKKAR

Date: Date:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
Project Primary Supervisor Project Examiner  
Name**:** Usman Rafi Name:

Designation: Lecturer Designation:  
University of Education. University of Education.

### OFFICE OF CONTROLLER OF EXAMINATION NOTIFICATION

No: \_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_

It is notified for the nomination of all the concerned that Mr. Waqas, Mr. Muhammad Abubakkar, Ms. Bareera Anam, Mr. Tamoor Akhtar are students of BS Information Technology of University of Education has completed all the requirements for the award of BS Degree in the discipline of Information Technology as per detail given hereunder:

#### **BS in Information Technology Cumulative Result**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **Credit Hours:\_\_\_\_\_\_\_\_\_\_\_** | | | **Cumulative**  **Grade Point**  **Average**  **(CGPA)** |
| **Registration No** | **Complete Name** | **Course work** | **Project** | **Total** |
|  | WAQAS |  |  |  |  |
|  | M TAMOOR |  |  |  |  |
|  | M ABUBAKKAR |  |  |  |  |
|  | BAREERA ANAM |  |  |  |  |

Project Title: A smart Application for Cafeteria Automation

Name of Supervisor: Usman Rafi

**Signed by**

**Controller of Examination**

**ACKNOWLEDGEMENT**

In the Name of Allah, the Most Gracious and the Most Merciful.

First of all, we would like to express our deepest gratitude to our supervisor, Usman Rafi, who has always given valuable advice and encouragement at each stage throughout in developing this project successfully.

We would like to thank him for giving the opportunity to learn and work under his guidance, which has been the most memorable experience.

We want to take this opportunity to thanks our parents, teachers, and also special thanks to all lectures of Faculty of information and for their attentions, guidance and advices to help in the development of this project.

Our sincere thanks also go to our fellow friends for their help to finish up this project.

May Allah S.W.T bless all the effort that has been taken to finish this project.

Thank you.

**ABSTRACT**

Traditional system involves paper work in the form of maintain various files and manuals. Maintaining critical information in the files and the manuals is full of risk. Nowadays people don’t have much time to spend in cafeteria by just there and waiting for waiter to take order. Many students and teachers visits cafeteria in lunch break and recess so they have limited time to eat and return to their respective offices and classes. So the purpose of this project is to develop a computerized App for online food ordering system that can be used to revolutionize the traditional system that currently implemented in campus cafeteria. After installing App and creating account customer login to system and navigate the cafeteria menu presented in the campus cafeteria and order food using their smartphones. Students do not have to worry about finding or taking a long time to choose what food they want to eat. This mobile App can save time and effort at the same time. Not only that, this App also offers convenience of checking the food items served, this function will benefit the students especially those with allergies to some food product, and students who want to keep their diet. In addition, it can also provide efficiency for the cafeteria by reducing time consuming, human errors and providing good quality customer service.

**[Chapter 1]**

**[GATHERING AND ANALYZING INFORMATION]**

* 1. **Introduction & Background**

Smartphones have revolutionized the human lifestyle of today. This is because  
"Mobile Applications" (Mobile Apps) an internet application designed specifically to  
help smartphone users by connecting to the Internet. Most of us always want updates  
fast, so Mobile Apps can help users to provide information quickly. The technological  
advancement of this era has made it easier to work and save time and effort. However, student often having a difficulty to find them prefer food in the campus. Meanwhile, there also a case for student who were allergic to certain food product to eat and this problem need to be dealt with. While in the cafeteria itself, there is a weakness in manual food ordering that were commonly used by cafeteria when ordering food. These problems have led to the idea of developing a system that would help to check the food that available in the cafeteria, check food ingredient and online food ordering.

The customers of today are not only attracted because placing an order online is very convenient but also because they have visibility into the items offered, price and extremely simplified navigation for the order. Cafeteria Automation system that we proposed here, greatly simplifies the ordering process for both the students and the cafeteria. System presents an interactive and up-to-date menu with all available options in an easy to use manner. Students can choose one or more items to place an order which will land in the Cart. Students can view all the order details in the cart before checking out. At the end, students get order confirmation details. Once the order is placed it is entered in the database and retrieved in pretty much real time. This allows Cafeteria Employees to quickly go through the orders as they are received and process all orders efficiently and effectively with minimal delays and confusion.

This project aimed to develop system that help customer to get their preferable food and also help to improve food ordering in campus cafeteria all together.

* 1. **Problem Statement**

There are few problems that lead to this project.

1. Firstly, student does not know when or where their favorite food will serve in the cafeteria. Student often struggle to look for their preferable food.
2. The challenges encountered by existing system serves as major drawbacks to the realization of efficiency and customer satisfaction.
3. Student who have allergic to certain food product manual method in food ordering process is slow to serve.
4. Manual order involving the traditional writing on a piece of paper as their food order processing. This method however is not efficient as it bound to encounter with errors  
   when dealing with multiple orders such as food serve not in sequence, missing of food  
   order paper and so on having difficulty to find which food they can eat. This may be seen as small case that most of people to overlook this matter.
5. Students will have to make long queues before placing their orders especially during peak hours. Having placed their order, the customer must then wait near the counter until their order is ready for collection.
   1. **Goals and Objectives**

The goal of this project is to create a system that will display food menu that available in the campus cafeteria. Food deliver service will be provided to help student to shorten their time to get their meal.

1. To build Mobile Application that allow campus student and staff to check food menu that available in campus cafeteria.
2. This application requires very fewer time factors as compared to manual system.
3. This Application provides fast and efficient automated environment instead of slow and error prone manual systems.
4. To improve food ordering system by ordering the food and then the food will be delivered to the customer.
5. To increase efficiency and improve services provided to the customers through better application of technology in daily operations.
6. To help cafeteria owner to improve their sales.
7. To enable customers to have a visual confirmation that the order was placed correctly.
8. Social Distancing
   1. **Work Limitations**

After researching this project system, online payment using apps not possible  
due to the complex of adding bank API into the system. The payment will be going to  
replace with cash on delivery service.

* 1. **Methodology**

A project management methodology is essentially a set of guiding principles and processes for managing a project.

* + 1. **Available Methodologies**

Below, we will take a look at 2 of the most popular available project management methodologies.

* + - 1. **Waterfall**

The Waterfall methodology is the traditional methodology on this list. It was first outlined by [Dr. Winston Royce in 1970](http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf) as a response to managing the increasingly complex nature of software development. Since then, it has become widely adopted, most prominently in the software industry. The Waterfall method is divided into discrete stages. You start by collecting and analyzing requirements, designing the solution (and your approach), implementing the solution and fixing issues, if any.

Graphically, you can represent it as follows:

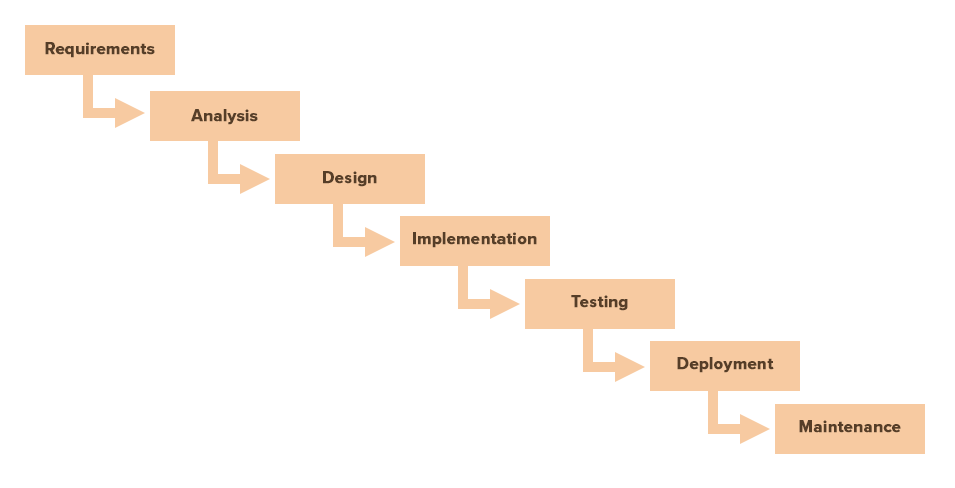


Fig 1.1

The Waterfall methodology is most commonly used in software development. It works best for the following project types:

* Short, simple projects
* Projects with clear and fixed requirements
* Projects with changing resources that depend on in-depth documentation

**Drawbacks of Waterfall Method**

1. You cannot go back a step, if a design phase has gone wrong. Things can get very complicated in the implementation phase.
2. High amounts of risk and uncertainty.
3. Not a good model for complex and component based projects.
   * + 1. **Agile**

Agile, another software development-focused PM methodology, emerged as a response to the failure of Waterfall method for managing complex projects. In approach and ideology, Agile is the opposite of the Waterfall method. As the name implies, this method favors a fast and flexible approach. There is no top-heavy requirements-gathering. Rather, it is iterative with small incremental changes that respond to changing requirements.

Agile methodology described as an “iterative” and “incremental” approach. Actually in waterfall method, development team will get only one chance to get each phase (like design, development, testing etc.) of a project. Whereas in Agile methodology, these phases are continuously revisited periodically to identify/ understand the project’s progress and direction.

* + - * 1. **Different Agile Methodologies**

Different agile methodologies are available but here we discuss only three major methodologies

1. **Scrum**

Scrum is one of the main Agile methodology and is mostly used by project teams. It involves several roles including a Scrum Master (the team leader), a Development Team, and an Initiative Owner. The Scrum process or Scrum Cycle occurs over a set time period, which includes steps like Sprint Planning, The Sprint, Daily Standup Meeting, and Sprint Review.

Scrum can be considered as a framework for project management which can be applied to build a product quickly and with minimum complexities. A scrum team can use methods and practices from other agile flavors as per their needs. Scrum is not something curated specifically for software world problems and can also be used by other industry projects.

1. **Kanban**

Kanban is known within Agile as a product-focused methodology. The Kanban Board is ideal for teams who need visualization in their work process — to physically see what to do and what will come next. Kanban is an ongoing process that adapt as the work goes on, whereas Scrum has a set time process that has a beginning and end date (until a new process starts).

1. **Extreme Programming (XP)**

XP is popular among smaller sized software development teams as it gives a lot of attention to the customer and their relationship with the development team. XP is a set of software practices aimed to deliver a quality product responsive to the changing client requirements. XP teams typically work in iterations that are one or two weeks long, the reason being XP believes in faster releases and quicker feedback.

* + 1. **Chosen Methodology**

Methodology that we choose to complete this project is Scrum Agile Software Development Methodology.

* + 1. **Reason to Choose Scrum Agile Methodology**

Scrum is recommended for smaller teams that mainly focus on project work. Those who use Scrum will benefit from its focus on an iteration format (Sprints) that sets out to execute the entire project process by using repetition. Scrum approach divides the working process into equal sprints - their duration may vary, everything depends on the specific sprint. Before we start a sprint, it is necessary to draw up tasks for this sprint. When it is completed, all results are discussed. This method makes it possible to lower development costs and make management process more efficient.

**1.5.4. Agile vs Traditional(Waterfall) Software Development**

Generally traditional software development includes sequential series of steps like requirement understanding, design, development, testing the product and product release. First, the requirements are clearly discussed with the client and documented. Then the second step will be the design phase which includes the design and visualization of software architecture. Later, the coding phase will start and followed by testing. At the end of the project, the product will be deployed.

In Agile Methodology, project visualization will be built at the beginning of the project itself and then the entire team follows this visualized structure (as shown in Fig 1.2).

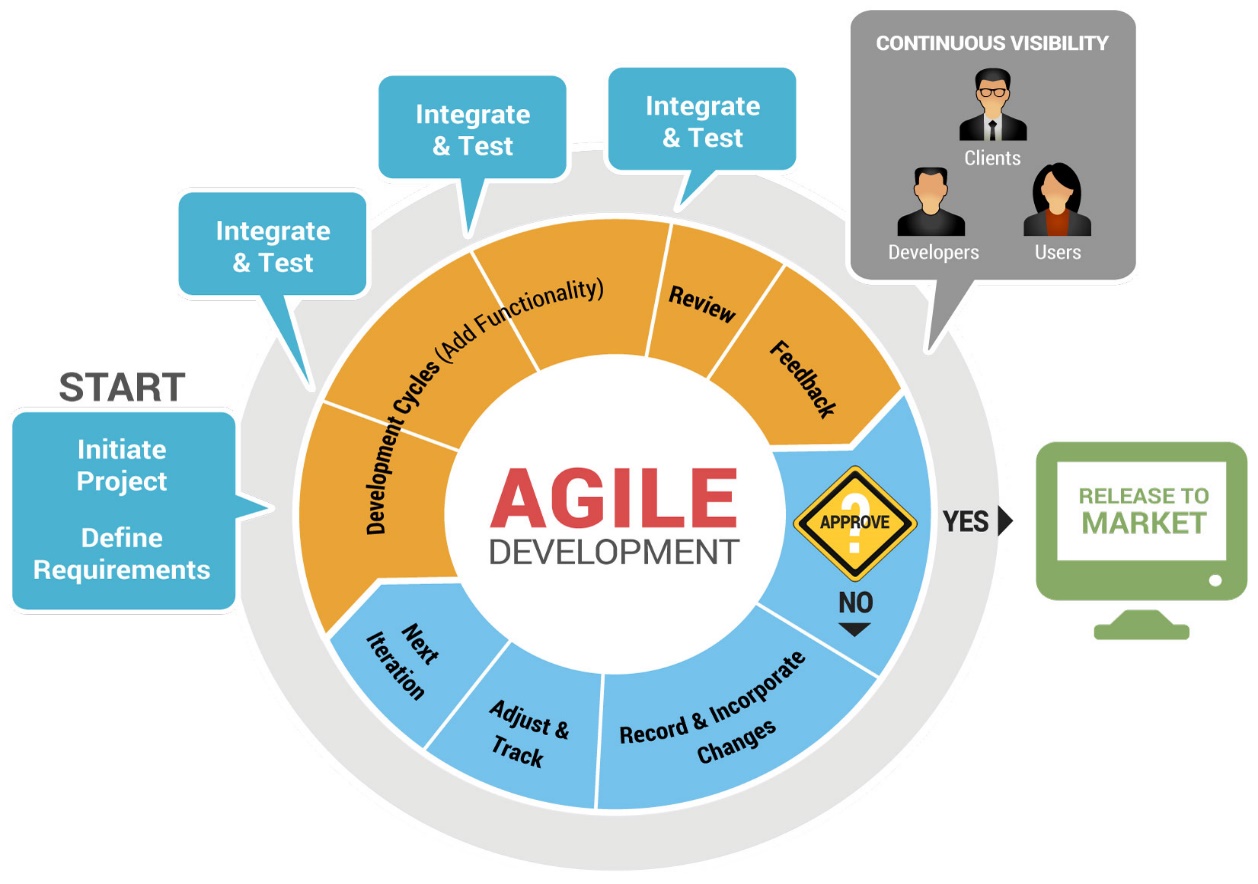


Fig 1.2

The “inspect-and-adapt” approach from Agile methodology, greatly reduces development costs and time to market the product because here teams can develop the software while gathering changes in requirements. The stakeholders can provide feedback to the development team to improve the quality of the product.

Thus Agile methodology help us to build the right product. There are some benefits of using Agile Methodology:

1. Real-Time-Planning
2. Immediate Changes
3. Sprint by Sprint (Iterations).
4. Instant Feedback from Customer
5. Efficient Risk Management

**[CHAPTER 2]**

**[Software Requirement Specification]**

**2.1. Domain Requirements**

When we talk about domain requirements of an Application we mean environmental requirements in which application run. Domain requirements are those requirements in which system operates and satisfied all the requirements. Domain requirements are very important because they often reflect fundamentals of the application domain. If domain requirements are not satisfied it might quite impossible to run application satisfactorily. Domain requirements contains both the design of the device and the development process. Important domain requirements of our application are given below.

**2.1.1. Recommended Operating Systems for Development**

1. Window 10 (that we used)
2. MAC: OS X v10.7 or higher.
3. Linux: Ubuntu (18.04) or higher

**2.1.2. Hardware Requirements for Developments**

Hardware requirements that are necessary to develop this application.

1. Processor minimum 1GHz; Recommended 2GHz or higher.
2. Hard Drive minimum 50 GB; Recommended 128GB or higher.
3. RAM minimum 2GB; Recommended 4GB or higher.

**2.1.3. Supported Browsers for Developments**

1. Chrome (Recommended)
2. Firefox
3. Safari

**2.1.4. Development Tools**

1. Visual Studio Code (code editor)
2. React
3. React Native (used to develop android and iOS application with same code)
4. Node.js (v12.17.0.)
5. Git CLI (for version controlling System of application)
6. GitHub Desktop (used to push and pull code at GitHub cloud)
7. Expo Cli (used to develop multi-platform applications)
8. Expo client android & iOS app download from iOS appstore and google play store used for real time application testing and visualizing same as to Android simulator

**2.1.5. Operating Environments**

Operating environment is an environment in which users run application. Our application will support both android and iOS environments but up till now this application will run only on android devices but for iOS user’s application will become available soon on Apple appstore. This Application support Android 5.0 or higher android versions. Active internet connection is required to run this application.

**2.1.6. Database Tools**

Tool that we used for Database is google firestore. Google firestore is an cloud service provided by google for developer to run and test applications. Services of google firebase used for this application are followings:

1. Firebase Auth for user Authentication
2. Firestore for Database
3. Firebase for server
4. Firebase Storage to store data

**2.1.7. Documentation Tools**

1. **Draw.io**

Draw.io is an online tool also available in desktop version which we used to draw all the diagrams of our documentation.

1. **MS Word**

MS Word is a tool that is used to create document files. MS Word is part of MS Office that we used to complete our documentation process.

**2.2. Stakeholder Requirements**

Stakeholder is a person or organization who is directly or indirectly participate in project and may affect or got affected by the outcome of the system. Stakeholders of our project are given below with their requirements.

**2.2.1. Admin Requirements**

1. Admin user-name
2. Admin password

**2.2.1.1. Admin Functionalities**

1. Create user
2. Edit user
3. Delete user
4. Access database

**2.2.2. Manager Requirements**

1. Manager username
2. Manager password

**2.2.2.1. Manager Functionalities**

1. View products
2. Edit existing products
3. Create new products
4. Delete products
5. View orders
6. Manage orders
7. Manage deliveries
8. Manage payments

**2.2.3. Customer Requirements**

1. Customer username
2. Customer password

**2.2.3.1. Customer Functionalities**

1. View food items
2. Select food items
3. Display menu
4. Place Order
5. Manage his/her orders
6. logout

**2.3. Functional Requirements**

|  |  |  |
| --- | --- | --- |
| **Require-mint No** | **Requirement**  **Name** | **Requirement**  **Description** |
| FR1 | Register –User | Before ordering any food using this application user must have to register first.  Following information will be needed for the registration process:   * User Name * Email-id * Password * Register with Facebook |
| FR2 | Login-User | After completing registration successfully user can login to the application.  Following information will be needed for Login:   * User Name/E-mail id * User Password |
| FR 3 | Forgot Password | If user forgot his password, then he/she can recover the password using Forgot Password option at the Login screen.  Following information must be needed for the recovery of password:   * Registered E-mail id |
| FR4 | Display the Main/Home Screen | After Login successfully user can navigate to the main screen of application. Following information’s will be displayed at the main screen of the app.   * Food Categories * Best Deals * Popular Foods * Menu icon at the left top corner of the app * Cart icon will be displayed at the top right corner of the app |
| FR5 | Drawer Button | User can navigate to the Menu screen after clicking on Drawer icon at the left top corner. User can move to the following features that are shown under the Menu icon.   * Food Menu * Profile * Cart * Order’s * Logout |
| FR6 | Place An Order | Lots of foods categories are available in the cafeteria that are also available in this app with their prices. So user can select one or more than one food items at a time from the list of foods by clicking the food image. |
| FR7 | Add to Cart | After selecting food items user can feel free to place an order by clicking “Add to Cart” button at the bottom of the screen. |
| FR8 | Review the Order | Changes to Customer order means user can make changes in the selected order. Following changes can be done to the order:   * Add more food items * Remove food items * Delete the order   After changing the order customer can review the complete order before placing/confirming the order. |
| FR9 | Checkout | After selecting the food and clicking on “Add to Cart button” at the bottom of the screen user will navigate to the “Checkout” screen where following information will be displayed.   * Selected food items with their prices * Total bill of selected food items * Quantity of each food item   After seeing his/her order user can place the order by Clicking on Checkout button at the Bottom of screen. |
| FR10 | Payment and Address Details. | After Confirming Order user navigate to the Payment and Address details where user will add his/her address details where to order will be delivered only inside the university.  Payment only will be made on time delivery. |
| FR11 | Profile Settings | User can do following operations with his/her profile:   * Update   If the user wants to update his profile he/she can update his/her profile by clicking on profile button under drawer button at the left top corner.  User can update his username, password and profile picture.   * Insert   User can also insert some additional information which are not required to use this application but for the better look profile like that.   * Profile picture * Date of Birth * Course of study * Current semester * Session |
| FR12 | Logout | User can also Logout by clicking on logout button which are present under the drawer button at the left top corner |

**2.4. Non-Functional Requirements**

Some important non-functional requirements of this app are followings:

|  |  |  |
| --- | --- | --- |
| Requirement  No | Requirement  Name | Requirement  Description |
| NFR1 | Security and Privacy | Security and privacy are the most important non-functional requirements of any system not only ours.  Security is about protecting of data whereas privacy is about safeguarding user identity.  Some important requirements regarding to security and privacy are followings:   * Preventing from unauthorized access * E-mail confirmation by sending E-mail to the E-mail provided by user during registration * E-mail strategy for Forgotten Password * Any user will not be allowed to delete or add food items into existing ones of the application. * Users shall receive notification of profile changes via preferred communication method of record when profile information is modified. * Personal data of users will not be shared anyone. |
| NFR2 | Accessibility | The extent to which the application can be used by people with the range of capabilities to achieve a specified goal in a specified context of use.  Such as:   * The system will be accessible to people with disabilities. |
| NFR3 | Availability | App will be available during the university timing and we would try to deliver the food to the boys and girls hostel after university timing for the better user experience. |
| NFR4 | User-Friendly | App is very user-friendly and can be easily used by any person who have little more knowledge about smart phones. |
| NFR5 | Integrity | Application maintained the consistency, accuracy, and trustworthiness of data over its entire life cycle.   * For example, if a user orders a burger using this App then burger will be delivered to the customer not a sandwich or any other product. |
| NFR6 | Usability | User will be able to learn, operate, prepare inputs, and interpret outputs through interaction with this application.   * The Application will be easy to use by users who may only have one hand free. * People with no training will be able to use the Application. * A trained user will have the ability to submit a complete order for a food product chosen from menu list within maximum 5 minutes |

**[CHAPTER 3]**

**Use Case Description and Use Case Model**

* 1. **Use Case Diagram:**

Use case diagram is a behavioral UML Diagram and frequently used to analyze to identify, clarify, and organize system requirements. They enable us to visualize the different types of roles in a system and how those roles interact with the system. Use case diagrams are used to gather usage requirements of a system depending on our requirements that we can use in different ways. A use case diagram can identify the different types of users of a system and the different use cases. In this context, the term “system” refers to something being developed or operated. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. Hence to model the entire system, a number of use case diagrams are used.

* + 1. **Use Case Diagram objects**

Use case diagrams consist of 3 objects.

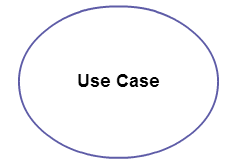
1. Actor
2. Use Case
3. System
   * 1. **Actor**

Actor in ause case diagram is **any entity that performs a role** in one given system. This could be a person or an external system and usually drawn like skeleton shown below.



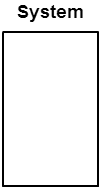
##### **Use Case**

A use case **represents a function or an action within the system**. It’s drawn as an oval and named with the function.



##### **System**

The system is used to **define the scope of the use case** and drawn as a rectangle. This an optional element but useful when we are visualizing large systems. For example, we can create all the use cases and then use the system object to define the scope covered by our project. Or we can even use it to show the different areas covered in different releases.



* 1. **Overall System Use Case Diagram**

Figure 3.1 describe the system use case diagram of our Application which shows relationships of actors of system with their functionalities. A **system use case diagram** displays the relationships between **consumers and providers of application services**. **Application services** are consumed by actors or other application services and the application use case diagram provides added richness in describing application functionality by illustrating how and when that functionality is used. The purpose of the system use case diagram is to help to describe and validate the interaction between **actors**and their **roles**with **applications**. As the architecture progresses, the **use case** can evolve from functional information to include technical realization details.

%3CmxGraphModel%3E%3Croot%3E%3CmxCell%20id%3D%220%22%2F%3E%3CmxCell%20id%3D%221%22%20parent%3D%220%22%2F%3E%3CmxCell%20id%3D%222%22%20value%3D%22Session%20Expired%26amp%3Bnbsp%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22490%22%20y%3D%22650%22%20width%3D%22170%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3C%2Froot%3E%3C%2FmxGraphModel%3E

Customer

Authentication Service

(

Firebase

)

Registration

include

>>

<<

Authentication

Validate

>>

include

<<

Place Order

Add Items to Cart

Preview Order

View Menu

Login

Confirm Order

Logout

Update Food Price

Add New Item

Update Inventory

Service Provider

Receive

Payment

Add/Delete Food Items

**UE Cafe**

**Figure 3.1**

* 1. **FR Based Use Case Diagrams:**

This section briefly explains the FR wise USE case diagrams that are related to our application.

* + 1. **User Registration Use Case Diagram**

**UC Number:** 1

**UC Name:** User Registration

**Functional Requirement No**: FR1

**Primary Actors/Stakeholders:** Customer

## **Secondary Actors/Stakeholders:** Google Firebase

**Description:** Figure 3.2 describes the registration module. This requirement describes that how user get registered with application. In registration procedure, record of user is validated by using user interface constraints and authenticate by matching it with database record that it exists previously are not. If no existence occurs, then record is saved in database.

**Diagram:**

User Information

E-mail & Password

Register

Customer

User Registration

Use Case

>>

<<

extends

<<

t

ends

>>

ex

Figure 3.2

* + 1. **User Login Use Case Diagram:**

**UC Number:** 2

**UC Name:** Login

**Functional Requirement No**: FR2

**Primary Actors/Stakeholders:** User

## **Secondary Actors/Stakeholders:** Google Firebase

**Description:**

Figure 3.3 describes the Login module. This use case describes that how user get login with application. This use case begins when user types his/her username/email and password on login form. System validates username/e-mail and password and logs him/her in to the system and display the Main/Home screen. User can also login with g-mail and Facebook.

**Diagram:**

Registered e-mail address

Authentic Password

Login

Customer

Login

Use Case

<<include>>>

<<Include>>

Login with Facebook

Login with G-Mail

>>

<<

in

c

lude

<<

include

>>

Figure 3.3

* + 1. **Home/Main Screen Use Case Diagram:**

**UC Number:** 3

**UC Name:** Home Screen

**Functional Requirement No**: FR3, FR4

**Primary Actors/Stakeholders:** User

## **Secondary Actors/Stakeholders:** Application and Firebase Fire store

**Description:** Figure 3.4 explains the use case diagram of home screen. This use case begins after user login successfully. After login user navigate to the home screen where he/she can see the images of food items, menu screen, cart icon and drawer icon.

Drawer

Display Cart

Home Screen

Food Items

Customer

Home Screen

Use Case

<<

>>

include

>>

include

<<

<<

include

>>

Figure 3.4

* + 1. **Place Order Use Case Diagram:**

**UC Number:** 4

**UC Name:** Place Order

**Functional Requirement No**: FR5, FR6

**Primary Actors/Stakeholders:** User

## **Secondary Actors/Stakeholders:** Service Provider

**Description:** Figure 3.5 shows the use case diagram of Place Order UC. This UC begins when user select a food item/product from foods list from home screen and then click on Add to Cart button and this use ends by selecting quantity of food.

Add to Cart

Select Quantity

Place an Order

Select Food Item

Customer

Place Order

Use Case

>>

<<

include

>>

include

<<

<<

include

>>

Figure 3.5

* + 1. **Review Order Use Case Diagram:**

**UC Number:** 5

**UC Name:** Review Order

**Functional Requirement No**: FR7, FR8

**Primary Actors/Stakeholders:** User

## **Secondary Actors/Stakeholders:** Service Provider

**Description:** Figure 3.6 shows use case diagram of “Review Order” UC. This UC start after adding food product into cart where user can Add or Remove food items from cart.

Add or Remove food items

Delete Order

Review Order

Customer

Review Order

Use Case

>>

<<

in

c

lude

include

>>

<<

Figure 3.6

* + 1. **Checkout Use Case Diagram:**

**UC Number:** 6

**UC Name:** Checkout Use Case

**Functional Requirement No**: FR9, FR10

**Primary Actors/Stakeholders:** Customer

## **Secondary Actors/Stakeholders:** Service Provider

**Description:** Figure 3.8 shows use case diagram of “Checkout” UC. This UC begins after clicking on checkout button and user add payment and address details where to deliver order.

Payment

Address Details

Checkout

Customer

Checkout

Use Case

extends

>>

<<

<<include>>

Figure 3.7

* + 1. **Profile Use Case Diagram:**

**UC Number:** 7

**UC Name:** Profile Use Case

**Functional Requirement No**: FR11

**Primary Actors/Stakeholders:** Customer

## **Secondary Actors/Stakeholders:** Service Provider

**Description:** Figure 3.8 show use case diagram of “Profile” UC. This UC begins by clicking on profile button inside the drawer where user can update and his/her profile for example if user wants to change his/her name and want to add some extra information such as current semester, Degree, Session, and DOB.

Update

Profile

Insert

<<extends>>

<<

extends

>>

Customer

Profile Use

Case

Figure 3.8

* + 1. **Logout Use Case Diagram:**

**UC Number:** 8

**UC Name:** Logout Use Case

**Functional Requirement No**: FR12

**Primary Actors/Stakeholders:** Customer

## **Secondary Actors/Stakeholders:** Service Provider

**Description:** Figure 3.9 shows the use case diagram of “Logout” use case. This UC begins when user click on logout button inside the drawer. Session will expire and user must have to login again to use this application.

Logout

Session Expired

<<

extends

>>

Customer

Logout

Use Case

Figure 3.9

* 1. **System Model**

Basically a use-case model describes how different types of users interact with the system.  As such, it describes the goals of the users, the interactions between the users and the system. Three important system models of our Application are following:

1. Admin
2. Customer
3. Manager

Now we will see these models in details with context diagram of each.

* + 1. **Admin Model**

Admin is the root user of our system who can perform administrative tasks, configuration and operations of system. Admin of our system can perform following tasks:

1. View
2. Edit system configuration
3. Add users
4. Create users accounts
5. Delete users accounts

For better understanding see figure 3.10 below.

Users

Accept/Deny

Request

Validate

Retrieve

ADMIN

Firebase Authentication

Access/Delete/Edit/Add

View

Retrieve

Save

Firebase Login DB

Figure 3.10

* + 1. **Manager Model**

Manager in our system is the person who can perform the following tasks:

1. View product details
2. Edit product details
3. Add products
4. Delete products
5. View and manage orders
6. Manage sales

For better understanding see figure 3.11 below.

‑‑‑‑

MANAGER

Login

Create product categories &

description

Manage order (view sales

report change product status)

Product DB

Login DB

Order DB

View

Request/Edit/delete

Accept/Deny

Request

Retrieve

Validate

Save

Retrieve

Access/Update

View

Store

Retrieve

* + 1. **Customer Model**

Customer is the user who is going to use our application. Customer can interact with application and place an order. Customer can perform the following functionalities:

1. Customer Registration
2. Login
3. View Products
4. Place an Order

For better understanding see figure 3.12

Product list

Product

DB

DB

CUSTOMER

Order placement

Login

Login DB

Registration

Order DB

Request

View

Retrieve

Request

Request

Place order

Receipt

Accept/Deny

Retrieve

Validate

Save

Confirmation

Response

Request

Submit

Confirmation

Figure 3.12

**[CHAPTER 4]**

**DESIGN**

* 1. **Architecture Diagram**

Architecture diagram help system designer and developers to visualize the high-level, overall structure of system or application for the purpose of ensuring that their system meet’s user needs and requirements. We can also use architecture diagram to describe design pattern that are used throughout the design. Architecture diagram somewhat is like a blue print that can be used as a guide for convenience of discussing, improving and following among their team. For application developers they need application architecture diagram to understand, clarify, and communicate ideas about the application architecture and user requirements that the application must support.

Architecture Diagram is basic framework that can be used at planning phase of application, helping partners understand the architecture, discuss changes, and communicate intentions. A blue print of our application is given below (figure 4.1).

**User**

**Registration**

**User**

**Login**

**Customer**

**Database**

**Products**

**Database**

**Application**

**Interface**

**Display**

**Foods**

**Manage**

**Orders**

**Address**

**Add/Delete**

**Products**

**Manage**

**Sales**

**Payment**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mobile Devices** | **Firebase Auth** | **Application Interface** | **Firebase Firestore** |

Figure 4.1

Figure 4.1 shows architecture diagram of our application. It is clear from diagram that our application use Firebase firestore to save and retrieve data. All data of application store in firebase database. Firebase provide real time tracking of data and upgrading hence it is also used for communication purpose.

* 1. **E-R Diagram**

An entity relationship model describes the overall structure of database of a system or application with the help of diagram which is known as “Entity Relationship Diagram”. An ER Diagram is a design or a blueprint of a database that later can be implemented as database. The main components of E-R diagram are entities and set of relationships. An E-R diagram shows relationship between set of entities of a system/application. An entity set is a group of similar entities and these entities can have attributes. E-R diagram shows complete logical structure of a system/application database.

The main entities of our application are followings:

1. Admin
2. Customer
3. Manager

Let’s have a look at our E-R Diagram (figure 4.2) to understand the concept of application database, set of entities, relationships between these entities, and attributes of each entity.

Users

username

email\_id

user password

M

#User\_Id

Manage

1

Foods

Orders

Payments

M

M

M

Order\_Id

order no

order time

images

Has

food name

image\_id

payment amount

payment\_id

payment type

#user\_id

M

Sales

Has

Has

delivery time

order\_id

food\_id

sales\_id

payment

Application

Has

1

Admin

1

Manager

1

admin\_name

admin\_password

manager password

manager\_name

Place

Figure 4.2

* 1. **Data Flow Diagram**

A data flow diagram is a graphical interface of how data is processed in terms of request and response. Data flow diagram show flow of information through a process or system. Data flow diagrams visually represent the systems and processes that would be hard to describe in the form of text. Visualizing each process makes it easy to identify inefficiencies of a system/application. DFDs are built using standardized notations and symbols to describe the entities and their relationships.

* + 1. **DFD Level (0)**

Context level DF diagram that shows only the top level also known as level0. At this level, there is only one visible process that represent the complete functions of a system/application in regards to how it interacts with external entities. Context DFD is the entrance of a dataflow

Admin

Cafeteria Kitchen

Manager

Customer

System

0

Request

Response

Data Flow (Bidirectional)

Food

Preparation

Request

Response

Data Flow (Bidirectional)

Request

Response

Data Flow (Bidirectional)

Deliverer

External

Entity

Figure 4.3

* + 1. **DFD for User (Level 1)**

Figure 4.4 shows the level 1 DFD of our application. Level1 and higher levels of DFD highlights main function carried out by the application. We must break a single process into its sub processes. Figure 4.4 shows DFD diagram of user registration and Login functions of our application.

Registration

USER

1.2

login

1.1

Response

Check for Login

Request accept/reject

Request for Login

Request for Registration

Check for Registration

Response

Request accept/reject

Figure 4.4

* + 1. **DFD for User (Level 2)**

Figure 4.5 explains the DFD of user at level 2. Level 2 describes the complete functionalities of our application.

USER

Registration

Products

Login

2.2

View Foods

2.1

Place Order

2.4

Add to Cart

2.5

Payment

2.6

Auth

2.3

Show item

Check for show item

Response

Response accept/reject

Check for Registration

Request for Registration

Response

Response accept/reject

Check for login

Response

Check for order detail

Response

Check for cart

Response

Check for payment

Request for Login

Request for food order

Response accept/reject

Response accept/reject

Request for add to cart

Response accept/reject

Request for payment

Figure 4.5

* 1. **Class Diagram**

Class diagram comes under structural diagrams. Structural diagram represents static view or structure of application. Class diagram is the most widely used in documentation of application architectures. Class diagram is the backbone of all object oriented systems. It represents the static structure of system/application. Class diagram displays systems classes, their attributes, access specifier and methods. It is helpful in recognizing the relationships between objects as well as classes. Class diagram describes major responsibilities of a system/application and it incorporate forward and reverse engineering. A collection of class diagrams as a whole represents complete system.

Class diagram made up of three subsections:

1. Upper section
2. Middle section
3. Lower section

|  |
| --- |
| Upper section |
| Middle Section |
| Lower section |

Figure 4.6 shows class diagram of our application.

**Customer**

username: String

-

#Password: Encrypted:

+

Registration ();

+

Login ();

select Food ();

+

send Order ();

+

+

recieveOrder ();

**Payment**

+

total amount: float

payment option

+

**Place Order**

Menu: String

-

+

:

customer ID & Password

Encrypted

order id: int

+

+

payment Option: string

1

n

**Detail Order**

+

quality: String

+

quantity: int

+

tax Status: string

calcTotal();

+

Cafeteria

+

Check the Order

:

+

Deliver the Order

Delivery

of order

Cash on Delivery

+

Amount: float

+

Address: string

1

1.. \*

Figure 4.6

* 1. **Sequence Diagram**

This section covers sequence diagram of our application. Sequence diagram explain the objects interaction with one another. Sequence diagram represents the flow of messages in the system and is also termed as an event diagram and it helps in visualizing several dynamic scenarios. The following figures shows the major objectives of our project and shows objects interaction with one another.

**4.5.1 Signup Sequence Diagram:**

Figure 4.7 describes that how user register itself. First customer interacts with signup interface. Here customer entered required detail. Then system validate entered data with firebase and returned a validated response to customer if entered details are correct otherwise an alert message to enter correct credentials.

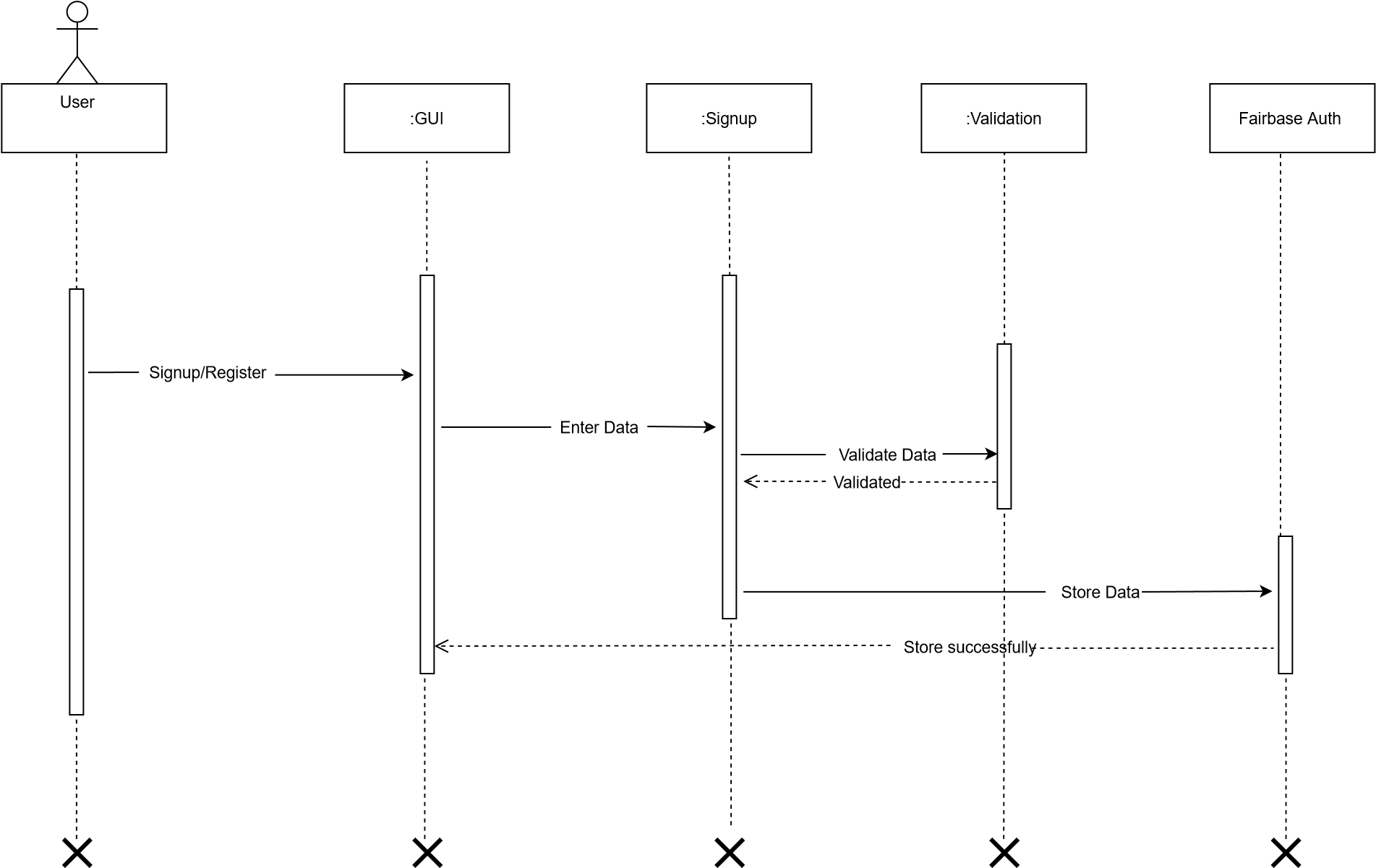


Figure 4.7

* + 1. **Order Management Sequence Diagram**

Figure 4.8 describe order management system of our application. It is clear from diagram that how different objects interact with each other in a flow. When user interact with application first time its session is checked if already logged then not need to login again. After login successfully user select a food item, add to cart, check bill, and the checkout to get services.

Customer

Cafeteria

Order

Menu Card

Bill

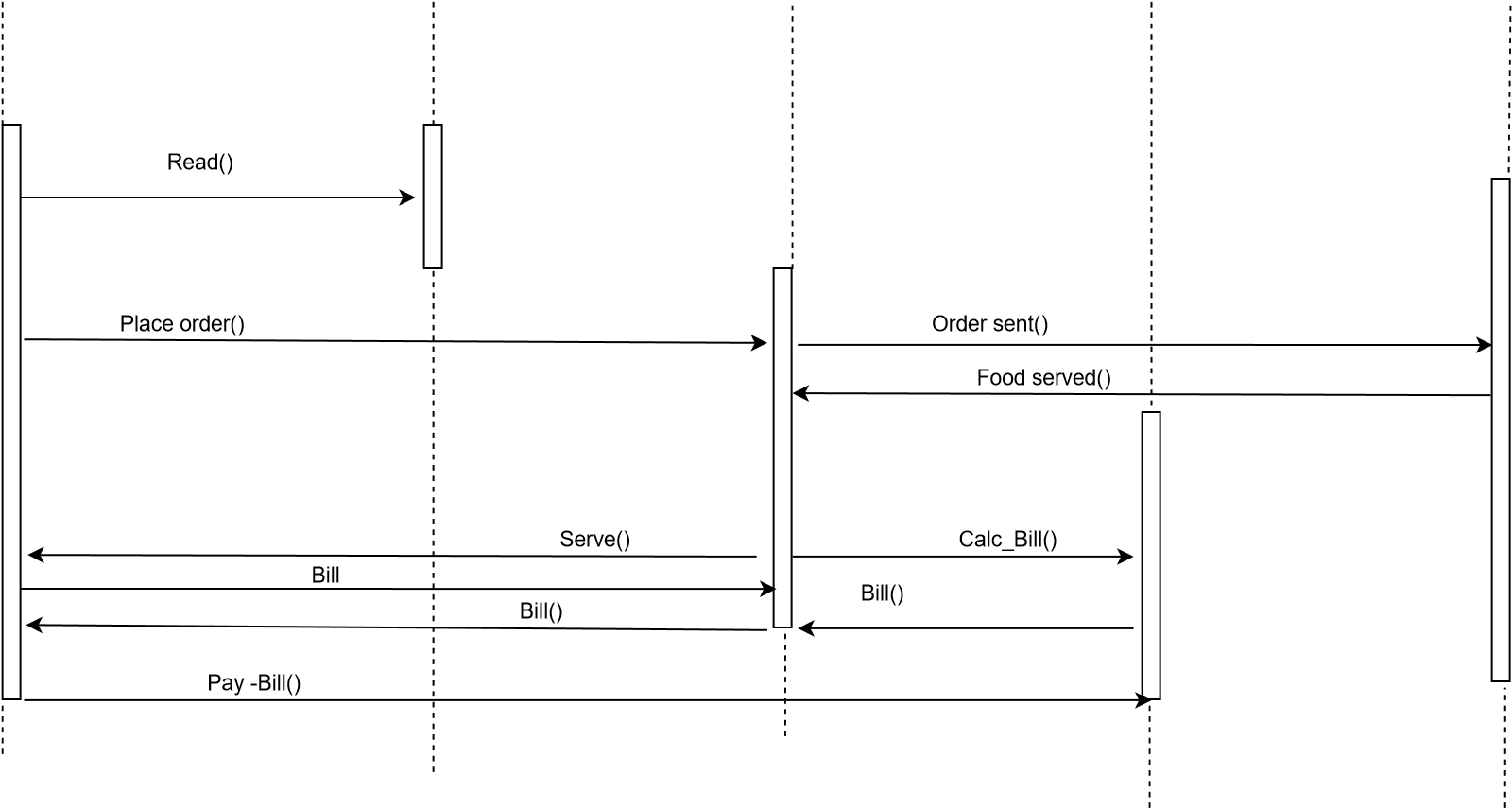


Figure 4.8

**[Chapter 7]**

**Conclusion and Future Work**

**References**

1. Leong Wai Hong (2016), Food Ordering System using Mobile Phone From  
    <http://eprints.utar.edu.my/1943/1/IA-2016-1203135-1.pdf>.
2. Mayurkumar Patel (2015), Online Food Order System for Restaurants, <https://scholarworks.gvsu.edu/cgi/viewcontent.cgi?article=1222&context=cistechlib>.
3. Agile Methodology. (n.d.). Retrieved September 10, 2014, from [*http://agilemethodology.org/*](http://agilemethodology.org/)