Protect Title: Smart-Restaurant -A Web-Based Application for Restaurant Manager, Chef and Customer to Increase Efficiency and Get the Customer Satisfiction

Submitted by

Abdulla Al Saimun

171-35-1921

Department of Software Engineering

Daffodil International University

Supervised by

Tapushe Rabaya Toma

Lecturer (Senior Scale)

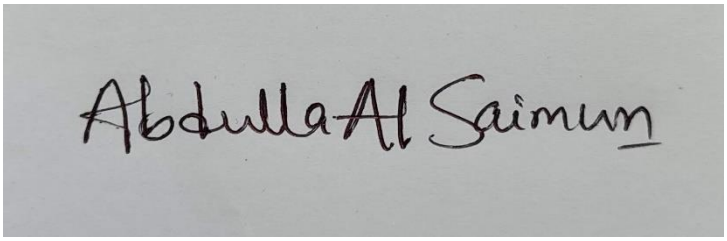
Department of Software Engineering

Daffodil International University

This Project report has been submitted in fulfillment of the requirements for the Degree Bachelor of Science in Software Engineering.

DECLARATION

I hereby declare that, this project has been done by me under the supervisor of Mrs. Tapushe Rabaya Toma, Lecturer (Senior Scale), Department of Software Engineering, Daffodil International University. I also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree of diploma.



……..
Abdulla Al Saimun
ID: 171-35-1921
22nd Batch
Department of Software Engineering
Faculty of Science and Information Technology Daffodil International University

……...
Tapushe Rabaya Toma
Lecturer (Senior Scale)
Department of Software Engineering
Faculty of Science and Information Technology
Faculty of Science and Information Technology Daffodil International University

Acknowledgement

First of all, I am grateful to Almighty Allah who give me opportunity to walk through final year. I have learned many things in my university life. I saw that our university always encourage us to follow ethics, morality, civility. In there, I also learned how to work on stress point. I would not able to complete my degree without helps of my teachers. I am grateful and thankful to my teachers for this.

Then I would like to express my special thanks to gratitude to my supervisor “Mrs. Tapushe Rabaya Toma” Faculty of software Engineering, Daffodil International University, Dhaka for excellent guidance and for having trust in my capability to complete this project.

I want to express my heartiest gratitude to Head, Department of SWE, for encouraging me and providing me with such an opportunity.

I am grateful to my parents as well as family members to give me support and opportunity to stand here. Without them it is impossible for me. I also thanked them for encouraging me when I needed and keep believe on me. I would like to extend my best wishes to all the teachers, friends and stuff member of our department.

Abstract

‘**Smart-Restaurant’** is a system that helps a restaurant to handle customers, managers and chefs efficiently. We know that the restaurant business is one of the toughest businesses. So, restaurant owners always try to increase efficiency and customer satisfaction.

There are many restaurants that can’t handle their clients properly, which reduces customer satisfaction. Nowadays there are shortages of restaurant waiters which increases the difficulty of running a restaurant. Order delays always hamper and make a business unprofitable. ‘**Smart-Restaurant**’ system can overcome this problem. We are now living in a digital era. Almost every person has their smart phone or internet-enabled devices. **‘Smart-Restaurant’** work on any internet enabled devices.

Using this system, customers can view all items and order items in the system. Customer can provide their feedback and view previous feedback. They can view the order status and history of their own.

Manager overviews the all data in the system. He can add new products. If he wants, he can update, delete food items in the system. Manager confirm orders from customers and send those orders to the chefs. He can view all feedback in the system. Manager can also generate PDF reports.

Chefs receive order from manager and deliver them when ready.

This system reduces the order processing time which helps to increase profit. Automate the redundant work and increase efficiency. It is delivering a great customer experience and determining profits and costs. This system can reduce the communication gap between customer, manager, and chef.

Table of Content

DECLARATION I

ACKNOWLEDGEMENT II

ABSTRACT III

CHAPTER 1 1-4

1. INTRODUCTION 1

1.1 Project Overview 1

1.2 Project purpose 1

1.2.1 Background 1

1.2.2 Benefit 1

1.2.3 Goal 2

1.3 Stakeholder 2

1.4 Proposed System Model 2

1.4.1 Agile-Model 3

1.5 Project Schedule 3

1.5.1 Gantt Chart 4

1.6 Release Plan 4

1.7 Related Work 4

CHAPTER 2 5-8

2. SOFTWARE REQUIREMENT SPECIFICATION 5

2.1 Functional Requirements 5

2.2 Non-Functional Requirements 7

2.3 Software Requirements 7

2.4 Hardware Requirements 8

CHAPTER 3 9-41

3. SYSTEM ANALYSIS 9

3.1 Use Case Diagram 9
3.2 Use Case Description 11
3.3 Activity Diagram 32

CHAPTER 4 42-50
4. SYSTEM DESIGN SPECIFICATION 42
4.1 Sequence Diagram 42
4.2 Class Diagram 49
4.3 Entity Relationship Diagram 50

CHAPTER 5 51-59
5 USER INTERFACE 51

CHAPTER 6 60-72
6. SYSTEM TESTING 60
6.1 Introduction to System Testing 60
6.2 Test Case tables 60

CHAPTER 7 73-73
7. Conclusion 73
7.1 GitHub link 73
7.2 Limitations 73
7.3 Obstacles and Achievement 73
7.4 Future work 73
References 73

CHAPTER 1: INTRODUCTION

1.1 Project Overview

‘Smart-Restaurant’ is web-based application for a restaurant that helps many types of people. In a restaurant, there are three stockholders like customer, manager and chef. Customer can easily order their items using the system and can see the state of the order. Manager get the order and approve them to send the order to the chef. When order is completed by the chef then chef will update state of order as delivered

1.2 Project Purpose

‘Smart-Restaurant’ main purpose ensure the efficiency, reliability and get the good customer experience. This is reducing the operation time in the restaurant by automation which will help to generate the more profit. Normally in a restaurant customer can’t complain but, in this system, they can complain or send the feedback. Manager can view all types order and have access to approve or cancel them. If customer approve the order, then order directly go to the chefs. When chef complete the order then order status will change to delivered.

1.2.1 Background

Using traditional ordering system like customer order items in the counter or waiters take orders directly from customer has many drawbacks. Traditional system consumes much time which is not efficient and need much manpower. Now a days people carry internet enabled phones. Customer can scan or browse the web address to go ‘Smart-Restaurant’ system. We can take the advantages of internet using ‘smart-restaurant’ system. There is much communication gap between customer, manager and chef. Customer can express their thoughts or feedback in system. Chef will get order directly using the system when manager approve the order.

1.2.2 Benefits & Beneficiaries

- By this application, Customer, Manager and Chef
- Customer can make order from internet enabled phone or devices.
- Can do operation efficiently and generate profit using less human power.
- Customer can view all existing items in the system and view his order status.
- Customer can send feedback or review.
- Manager can view all orders and filtering them and add, update, delete the items.
- Chef get the order via manager and make order status change when products is delivered.

1.2.3 Goals

With our effort we are trying to

- Ensure restaurant business efficient and productive.
- Reduce the order processing times.
- Customer can directly order using internet enabled device without help of waiters.
- Manager can approve or decline the orders.
- Chef can view the approved orders and change order status to delivered.
- Increase profit and reduce the cost.

1.3 Stakeholders

Stakeholder for 'Smart-Restaurant' are

- Restaurant Owner
- Customer
- Manager
- Chef

1.4 Proposed System Model

System Models are practiced for better development of the project and maintain its development track. Models like Agile, Spiral, Waterfall are commonly practiced among developers. As our requirements are clear and specific but in future, we can include many necessary features in the system. So, I use Agile Methodology for my project.

1.4.1 Agile Model

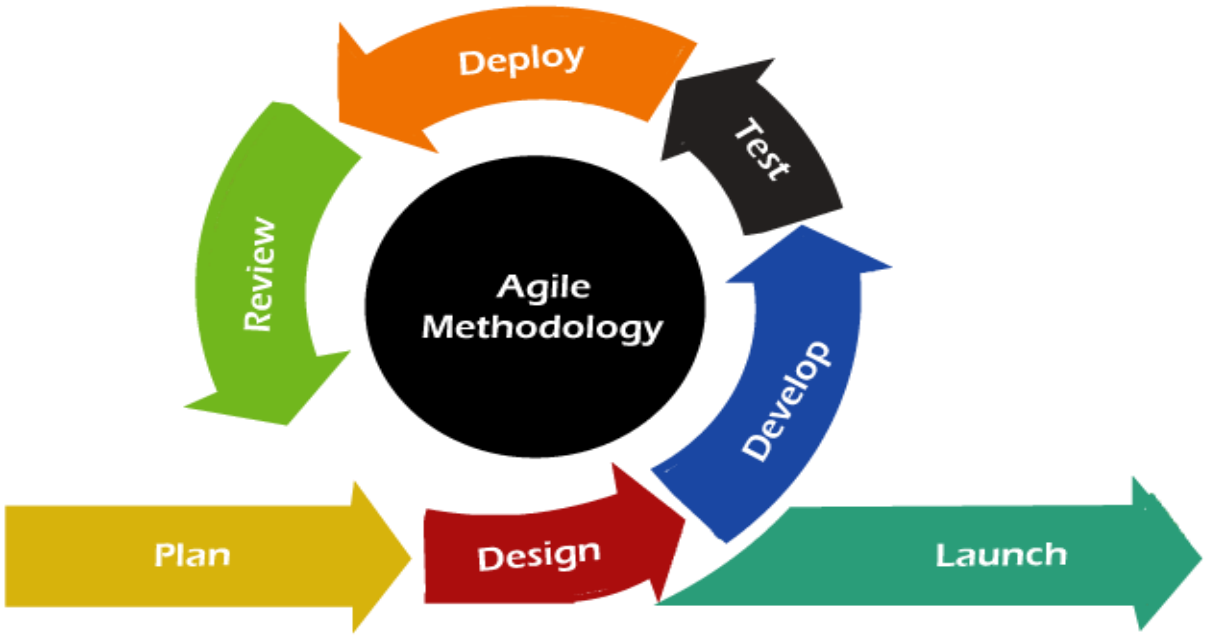


Figure 1: Agile Methodology

1.5 Project Schedule

Activities	Duration (in week)	Total week
Brainstorming	Week-1, Week-2	2
Problem identification	Week-2, Week-3	2
Requirement analysis	Week-4	1
Sketching	Week-5	1
Design specification	Week-6	1
Database design	Week-7	1
Implementation	Week-8, Week-9, Week-10, Week- 11	4
Testing	Week-12, Week-13, Week-14	3
Delivery	Week-15	1

1.5.1 Gantt Chart

Project Tasks	Start	End	Week1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15
Project Proposal & Presentation	Aug 1	Aug 22															
SRS	Sep 13	Oct 8															
Designing	Oct 9	Oct 29															
Coding	Oct 30	Nov 19															
Testing	Nov 20	Dec 3															
Finalization	Dec 4	Dec 17															

Figure 2: Gantt Chart

1.6 Release Plan

The first version of 'Smart-Restaurant' will be launched on 25th December, 2021.

1.7 Related Work

Restaurant Management system is a vast and complex system. There are many projects about restaurant management system with different types of requirements. There are some restaurant project and requirements link here.

- https://www.capterra.com/p/17313/OpenTable-for-Restaurants/
- https://limetray.com/blog/restaurant-management-system/
- https://www.softwareadvice.com/retail/yelp-reservations-profile/

Chapter 2: Software Requirement Specification

2.1 Functional Requirements

The functional requirements referred to a mandatory function which mandatory to the system. It must be able to perform for the web and also all kinds of software systems. All kind of application system has some functional requirements. Now, we are showing to mention functional requirements associating with this project.

FR 01	Log in
Description	Customer, Admin, Chef and Manager can in the system
Stakeholder	Manager, Customer, Chef, Admin

FR 02	Add new admin
Description	Admin can new admin the system
Stakeholder	Admin

FR 03	Add new Manager
Description	Admin can add new manager in the system
Stakeholder	Admin, Manager

FR 04	Add new Chef
Description	Admin can add new chef in the system
Stakeholder	Admin, Chef

FR 05	Create new Menu
Description	Admin and Manager can create new menu in the system
Stakeholder	Admin, Manager

FR 06	Retrieve, Update, Delete the Menu
Description	Admin and Manager can execute the CRUD operation on food menu
Stakeholder	Admin, Manager

FR 07	View all orders
Description	Admin, Manager and Chef can view the all orders
Stakeholder	Admin, Manager, Chef

FR 08	Change the order status
Description	Admin can change the order status to all types.
Stakeholder	Admin

FR 09	Log out
Description	Customer, Admin, Chef and Manager can log out in the system
Stakeholder	Manager, Customer, Chef, Admin

FR 10	Overview
Description	Can overview the total sell, items, total active orders and in process order
Stakeholder	Manager

FR 11	View and Filter order
Description	View all types of orders and filter them according to status.
Stakeholder	Manager

FR 12	Change the order status
Description	Update active order to processing order
Stakeholder	Manager, Chef, Admin

FR 13	Create new Menu
Description	Admin and Manager can create new menu in the system
Stakeholder	Admin, Manager

FR 14	Reject Order
Description	Manager Can reject order made by customer
Stakeholder	Manager, Customer

FR 15	View the feedback
Description	Manager can view the feedback from customer
Stakeholder	Manager, Customer

FR 16	Overview
Description	Can overview the total active orders and in process order
Stakeholder	Chef

FR 17	View and Filter order
Description	View all types of orders and filter them according to status.
Stakeholder	Chef, Manager

FR 18	Change the order status
Description	Update active order to delivered order
Stakeholder	Chef, Manager, Customer

FR 19	View all food items
Description	Chef can view the all-food items in the system.
Stakeholder	chef

FR 20	Log in
Description	Customer can login the system using credentials
Stakeholder	Customer

FR 21	Registration
Description	Customer need to complete registration with necessary data
Stakeholder	Customer

FR 22	View all Food Menu
Description	Customer can view the all-food menus.
Stakeholder	Customer

FR 23	View menu detail
Description	View the specific detail of menu
Stakeholder	Customer

FR 24	Add menu in the Cart
Description	Customer can add single or multiple in the Cart
Stakeholder	Customer

FR 25	Order
Description	Customer make order from cart
Stakeholder	Customer, Manager

FR 26	Feedback
Description	Customer can provide feedback in the system.
Stakeholder	Customer

FR 27	View order status and history
Description	Customer can view current order status and previous history
Stakeholder	Customer

2.2 Non-Functional Requirement

Table 2.2: Non-Functional Requirements

NFR ID	Name	Description	Priority
NFR-01	Availability	The system should work 24/7 as user can get access and service	High
NFR-02	Security	Using token- based authentication, session, validation 2FA it will be secure from unauthorized access.	High
NFR-03	Forget password	If user forgets their password, they can reset their password by email. And set one special character, number adds to reset them password.	High
NFR-04	Accuracy	Data or process requirement concerned with defining the precision which the solution will record or produce data	High
NFR-05	Maintenance	Its way how easy to support, change and enhance the system.	Medium

2.3 Software Requirements

- Operating system: Windows 10 or Linux
- Frontend: HTML, CSS, Bootstrap, JavaScript.
- Backend: Python
- Framework: Django
- Database: PostgreSQL
- Code Editor: VS Code or PyCharm

2.4 Minimum Hardware Requirements

- Processor: Intel core i3
- RAM: 4GB
- Hard Disk: 240GB SSD or 256GB HDD

CHAPTER 3: SYSTEM ANALYSIS

3.1 Use case Diagram

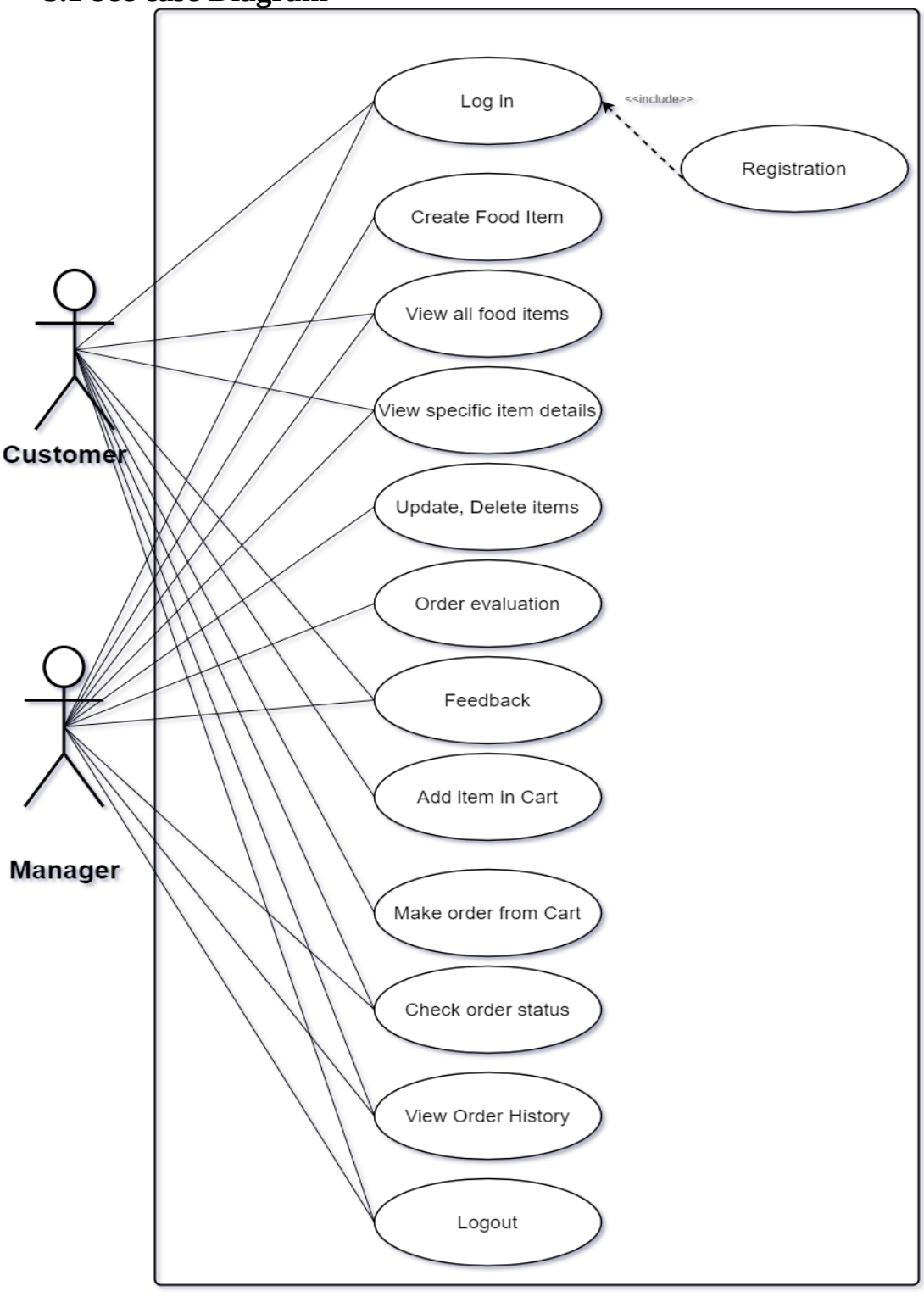


Figure 3: Use case diagram for Customer and Manager

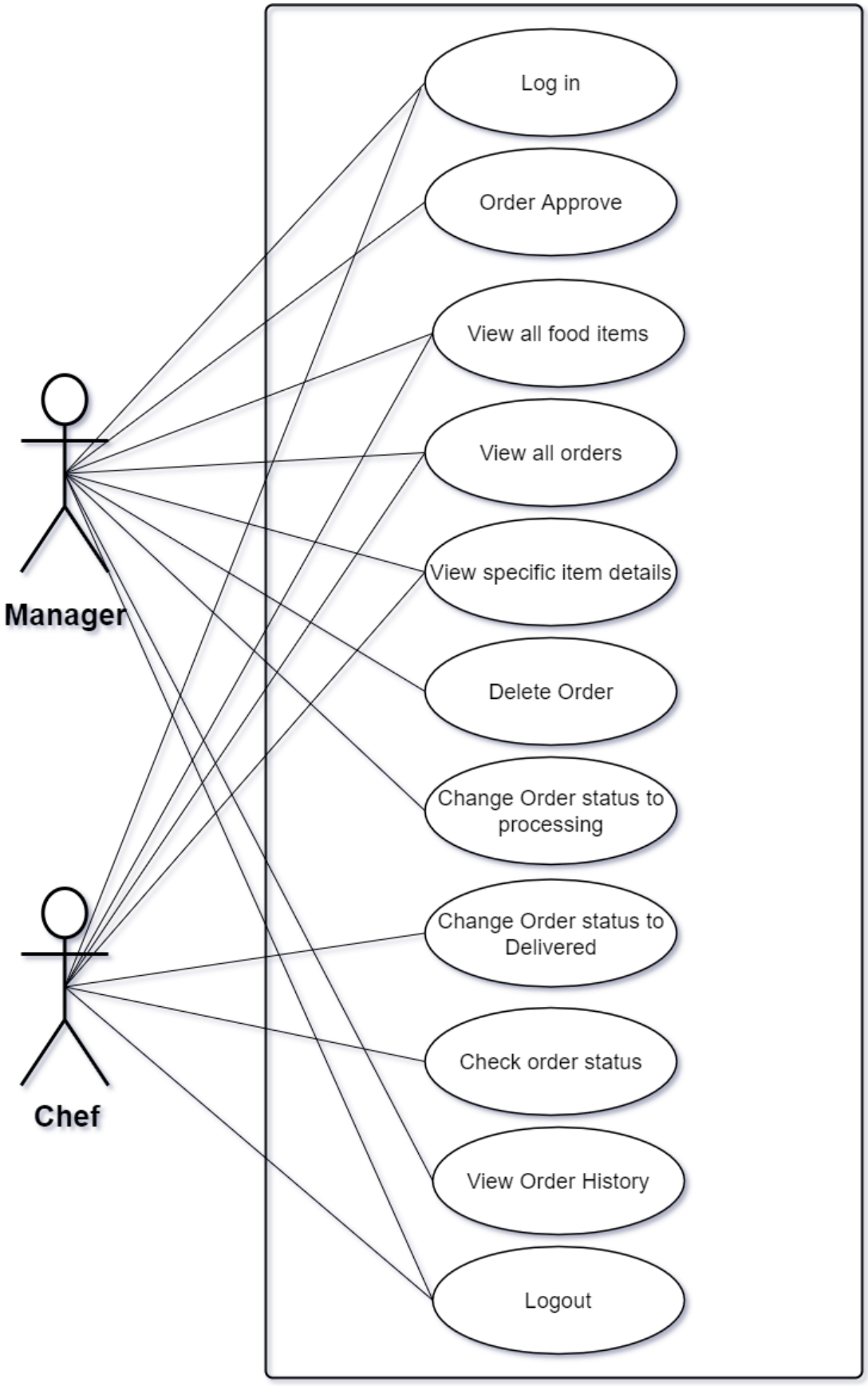


Figure 4: Use case diagram for Manager and Chef

3.2 Use Case Description

3.2.1 Login

Use Case	Login and log out system	
Goal	Users (Customer, Manager and Chef) can login and access the system.	
Preconditions	Must be registered in the system.	
Success End Condition	Users (Customer, Manager and Chef) can login and access the system.	
Failed End Condition	Display login error message	
Primary Actors:	Customer, Manager and Chef	
Secondary Actors	N/A	
Trigger	Login.	
Description/main Success Scenario	**Step**	**Action**
	1.	User activate the application
	2.	The user enters his or her username and password in the returning user section of the sign in screen.
Alternative Flows	**Step**	**Branching Action**
	1.	The user enters his or her username and password
Quality Requirements	**Step**	**Requirement**
	1.	When user login then needs to correct username for login and password.

3.2.2 Registration

Use Case	Registration	
Goal	Users (Customer, Manager and Chef) can register and access the system.	
Preconditions	N/A	
Success End Condition	Users (Customer, Manager and Chef) can registration and access the system.	
Failed End Condition	Display registration error message	
Primary Actors:	Customer, Manager and Chef	
Secondary Actors	Admin can register Manager and Chef.	
Trigger	Registration.	
Description/main Success Scenario	**Step**	**Action**
	1.	User activate the application
	2.	User can go to the Login page to enter the system.
Alternative Flows	**Step**	**Branching Action**
	1.	Go to the registration page again.
Quality Requirements	**Step**	**Requirement**
	1.	Manager and must put the authentic data to get access in the system.

3.2.3 Customer: View all Food Items

Use Case	View all Food Items	
Goal	Customer can view all existing items to purchase.	
Preconditions	N/A	
Success End Condition	View all items with their name, price, image and description	
Failed End Condition	Return in home page	
Primary Actors:	Customer	
Secondary Actors	N/A	
Trigger	Explore	
Description/main Success Scenario	**Step**	**Action**
	1.	Check items in system.
	2.	Evaluate items to purchase.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Items should be up to date.

3.2.4 Customer: View Specific Item

Use Case	View Specific Item	
Goal	Customer view details of specific item	
Preconditions	Must be logged in the system	
Success End Condition	Can view the details of item like title, description, price, quantity and image of the item.	
Failed End Condition	Go to page.	
Primary Actors:	Customer	
Secondary Actors	N/A	
Trigger	View item according to selected id	
Description/main Success Scenario	**Step**	**Action**
	1.	View item in the system.
	2.	Can purchase the item.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Keep items up to date

3.2.5 Customer: Add Item in Cart

Use Case	Add Item in Cart	
Goal	Customer can keep items in the cart which they want to purchase.	
Preconditions	Must be logged in the system	
Success End Condition	Can add items successfully.	
Failed End Condition	Go to home page.	
Primary Actors:	Customer	
Secondary Actors	N/A	
Trigger	Add item in the cart according to id,	
Description/main Success Scenario	**Step**	**Action**
	1.	View cart item
	2.	Check cart items and delete
Alternative Flows	**Step**	**Branching Action**
	1.	Add more items
Quality Requirements	**Step**	**Requirement**
	1.	Keeps items up to date

3.2.6 Customer: Make order from Cart

Use Case	Make Order from the Cart	
Goal	Customer can make final order from the cart.	
Preconditions	Must be logged in the system and must have item in the cart.	
Success End Condition	Order created successfully	
Failed End Condition	Go to home page.	
Primary Actors:	Customer	
Secondary Actors	Manager	
Trigger	Order from cart	
Description/main Success Scenario	**Step**	**Action**
	1.	View order items
	2.	Make order
Alternative Flows	**Step**	**Branching Action**
	1.	Go to home page
Quality Requirements	**Step**	**Requirement**
	1.	Authenticated as customer

3.2.7 Customer: Check Order Status

Use Case	Check Order Status and activity	
Goal	Customer view his/her order status and activity.	
Preconditions	Must be logged in the system	
Success End Condition	View Orders.	
Failed End Condition	Go to home page.	
Primary Actors:	Customer	
Secondary Actors	Manager	
Trigger	View Orders	
Description/main Success Scenario	**Step**	**Action**
	1.	View orders.
Alternative Flows	**Step**	**Branching Action**
	1.	Make another order
Quality Requirements	**Step**	**Requirement**
	1.	Must have completed order.

3.2.8 Customer: View Previous history

Use Case	view customer previous history	
Goal	Customer can view their previous order history which they made.	
Preconditions	Must be logged in the system	
Success End Condition	View Order history	
Failed End Condition	Go to home page.	
Primary Actors:	Customer	
Secondary Actors	N/A	
Trigger	View order	
Description/main Success Scenario	**Step**	**Action**
	1.	View all previous order
	2.	Check details of all previous order.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must have previous order to check history

3.2.9 Customer: Send Feedback

Use Case	Send Feedback	
Goal	Customer can send feedback or review to the system that higher authority can view and make steps.	
Preconditions	Must be logged in the system	
Success End Condition	Show customer feedback.	
Failed End Condition	Go to home page.	
Primary Actors:	Customer	
Secondary Actors	N/A	
Trigger	Feedback	
Description/main Success Scenario	**Step**	**Action**
	1.	Added feedback in system
	2.	Everyone can view the feedback
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Customer must be authenticated.

3.2.10 Manager: Overall Overview

Use Case	Overview of everything	
Goal	Manager can overview everything like total existing products, total sell, total orders.	
Preconditions	Must be logged in the system	
Success End Condition	View Item successfully.	
Failed End Condition	Go to home page.	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	overview	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the overview of all things
	2.	Make decision.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Manager must be authenticated.

3.2.11 Manager: View Orders

Use Case	View Orders	
Goal	Manager can view all types of order like active, processing and delivered order.	
Preconditions	Must be logged in the system	
Success End Condition	View all orders	
Failed End Condition	Go to home page	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	order	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the all types of orders.
	2.	Filter orders according to their status.
Alternative Flows	**Step**	**Branching Action**
	1.	Check active order, check processing order, check delivered order.
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Manager.

3.2.12 Manager: Confirm Order

Use Case	Confirm Orders	
Goal	Manager can confirm orders from customer and send them to chef.	
Preconditions	Must be logged in the system	
Success End Condition	Change the order status	
Failed End Condition <the state of the world if goal abandoned>	Show the error messages.	
Primary Actors:	Manager	
Secondary Actors	Customer, Chef	
Trigger	Confirm Order	
Description/main Success Scenario	**Step**	**Action**
	1.	Change the ordered status in Processing.
	2.	View these orders from processing order.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Manager.

3.2.13 Manager: Cancel Order

Use Case	Cancel Order	
Goal	Manager can cancel orders from customer	
Preconditions	Must be logged in the system	
Success End Condition	Canceled order successfully	
Failed End Condition	Go to order page.	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	Cancel-Order	
Description/main Success Scenario	**Step**	**Action**
	1.	Cancel the specific item and
	2.	Show the success message.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Manager.

3.2.14 Manager: View All Food Items

Use Case	View all food items	
Goal	Manager can view all existing items.	
Preconditions	Must be logged in the system	
Success End Condition	View Item successfully.	
Failed End Condition	Go to home page.	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	View food items	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the all-food items table with necessary information.
	2.	Get the details, update and delete button.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Manager must be authenticated.

3.2.15 Manager: View details, update and delete food item

Use Case	View details, update and delete food items.	
Goal	Manager can do CRUD operation in existing food items.	
Preconditions	Must be logged in the system	
Success End Condition	Action do successfully	
Failed End Condition	All items view	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	Details, Update, Delete	
Description/main Success Scenario	**Step**	**Action**
	1.	Show details
	2.	Update data successfully
	3	Delete item successfully
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Manager must be authenticated.

3.2.16 Manager: Add New Item

Use Case	Add new Item	
Goal	Manager can add new items in system with necessary information.	
Preconditions	Must be logged in the system	
Success End Condition	Item added Successfully	
Failed End Condition	Failed to add.	
Primary Actors:	Manager	
Secondary Actors	N/A	
Trigger	Add-Item	
Description/main Success Scenario	**Step**	**Action**
	1.	Add items with necessary info with category.
	2.	Show details of added items
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Manager.

3.2.17 Manager: View the Customer Feedback

Use Case	View the Feedback	
Goal	Manager can view all feedback from customer.	
Preconditions	Must be logged in the system	
Success End Condition	View all feedback	
Failed End Condition	Go to home page	
Primary Actors:	Manager	
Secondary Actors	Customer	
Trigger	feedback	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the all feedback associated with customer.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Manager.

3.2.18 Chef: Overall Overview

Use Case	Overview of everything	
Goal	Chef Can view overall like total existing products total orders.	
Preconditions	Must be logged in the system	
Success End Condition	View Item successfully.	
Failed End Condition	Go to home page.	
Primary Actors:	Chef	
Secondary Actors	N/A	
Trigger	overview	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the overview of all things
	2.	Make decision.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Chef must be authenticated.

3.2.19 Chef: View Orders

Use Case	View Orders	
Goal	Chef can view all types of order like active, processing and delivered order.	
Preconditions	Must be logged in the system	
Success End Condition	View all orders	
Failed End Condition	Go to home page	
Primary Actors:	Chef	
Secondary Actors	Manager	
Trigger	order	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the all types of orders.
	2.	Filter orders according to their status.
Alternative Flows	**Step**	**Branching Action**
	1.	Check active order, check processing order, check delivered order.
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Chef.

3.2.20 Chef: Complete Order

Use Case	Confirm Orders	
Goal	chef can confirm orders when he completed the order	
Preconditions	Must be logged in the system	
Success End Condition	Change the order status to delivered	
Failed End Condition	Show the error messages.	
Primary Actors:	Chef	
Secondary Actors	Customer, Manager	
Trigger	Confirm Order	
Description/main Success Scenario	**Step**	**Action**
	1.	Change the ordered status to delivered.
	2.	View these orders from delivered order.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	Must authenticated as Chef.

3.2.21 Chef: View All Food Items

Use Case	View all food items	
Goal	Chef can view all existing items.	
Preconditions	Must be logged in the system	
Success End Condition	View Item successfully.	
Failed End Condition	Go to home page.	
Primary Actors:	Chef	
Secondary Actors	N/A	
Trigger	View food items	
Description/main Success Scenario	**Step**	**Action**
	1.	Get the all-food items table with necessary information.
Alternative Flows	**Step**	**Branching Action**
	1.	N/A
Quality Requirements	**Step**	**Requirement**
	1.	chef must be authenticated.

3.3 Activity Diagram for Customer

3.3.1 Customer Login

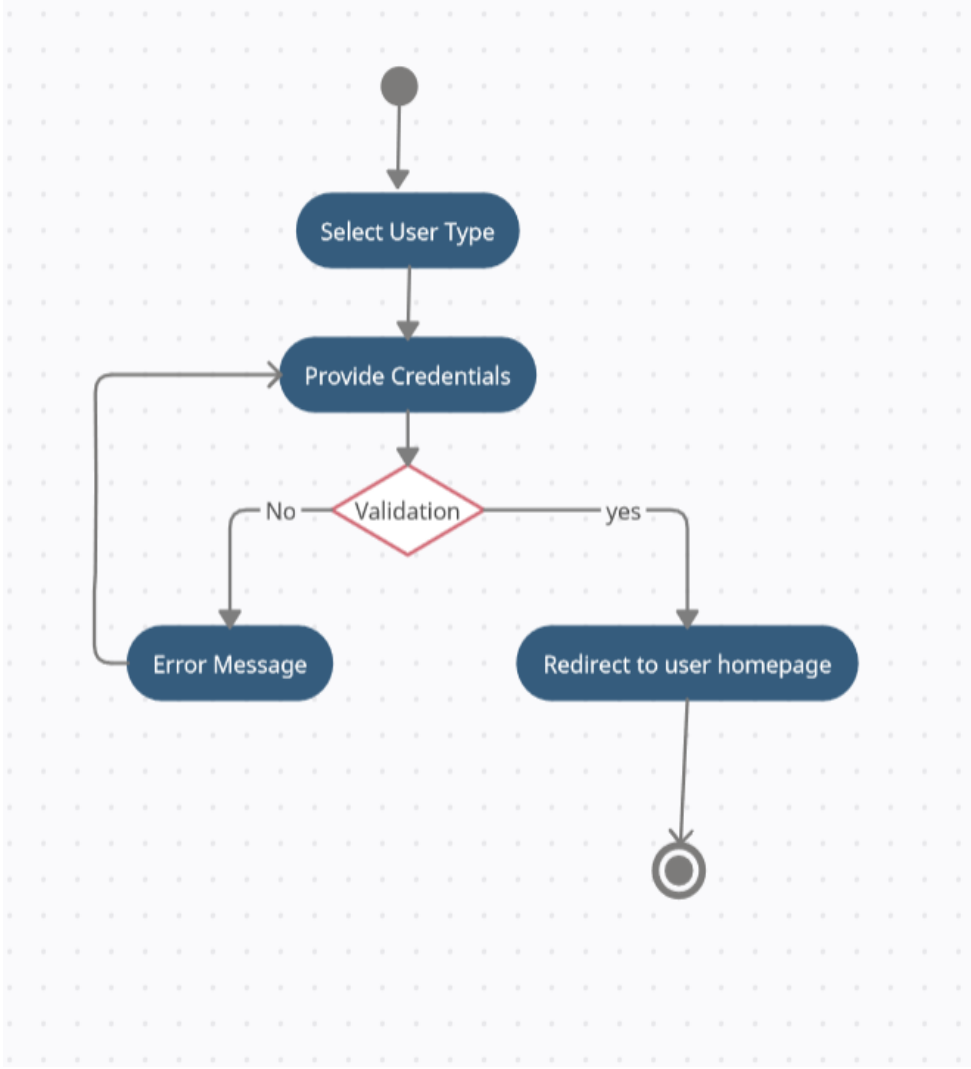


Figure 5: Login

3.3.2 Customer Registrations

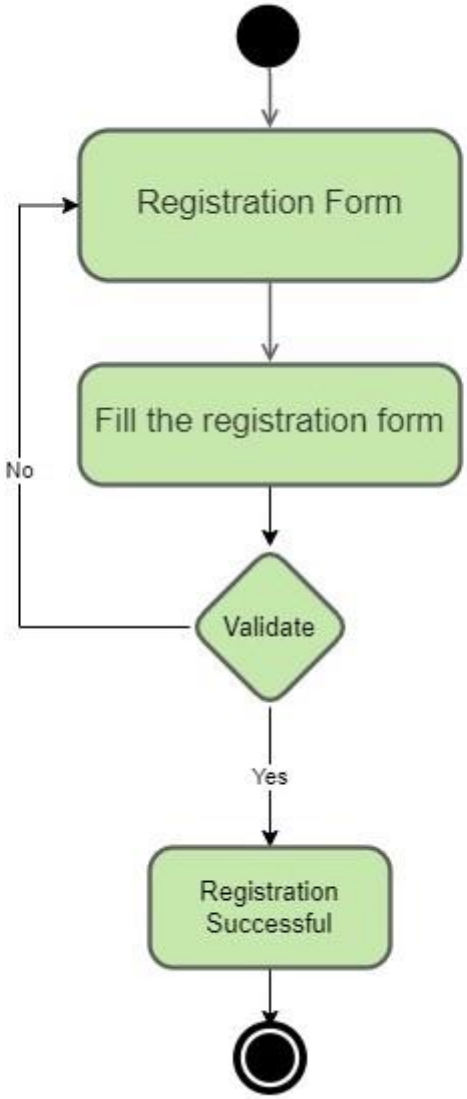


Figure 6: Registration

3.3.3 Customer: View All Food Items

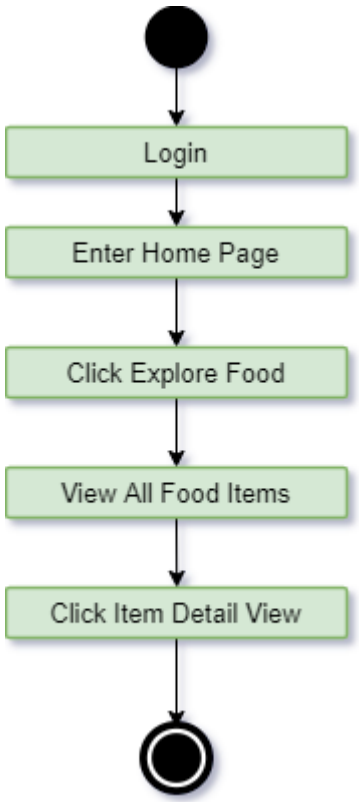


Figure 7: View All Food Items

3.3.4 Customer: Add Item in the Cart and Order from Cart

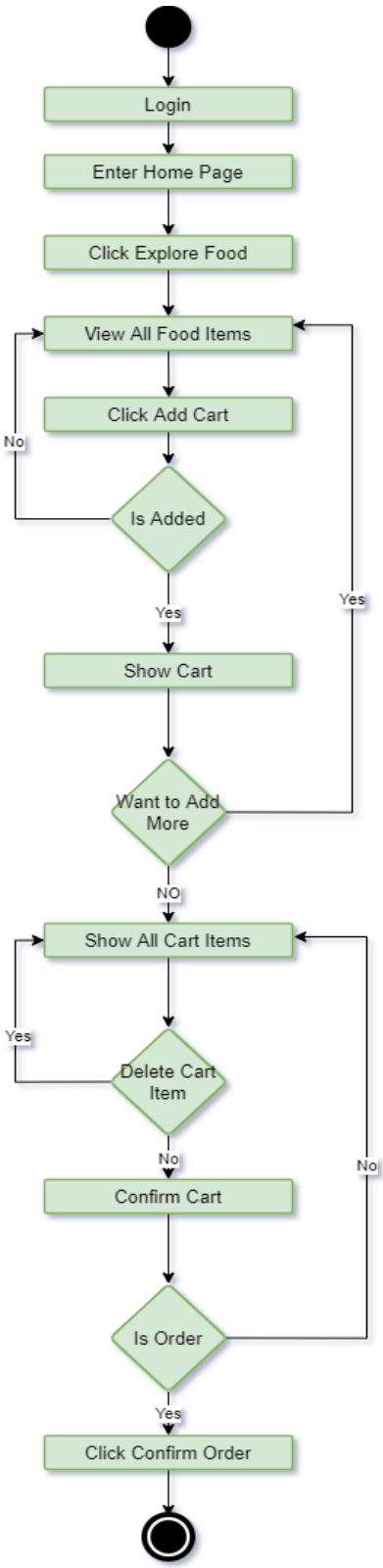


Figure 8: Add Item in the cart and Order from Cart

3.3.5 Customer: Order Items View

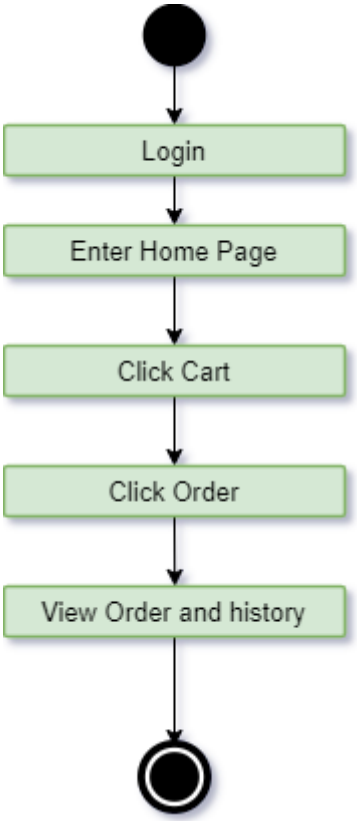


Figure 9: Order Items View

3.3.6 Customer: Feedback

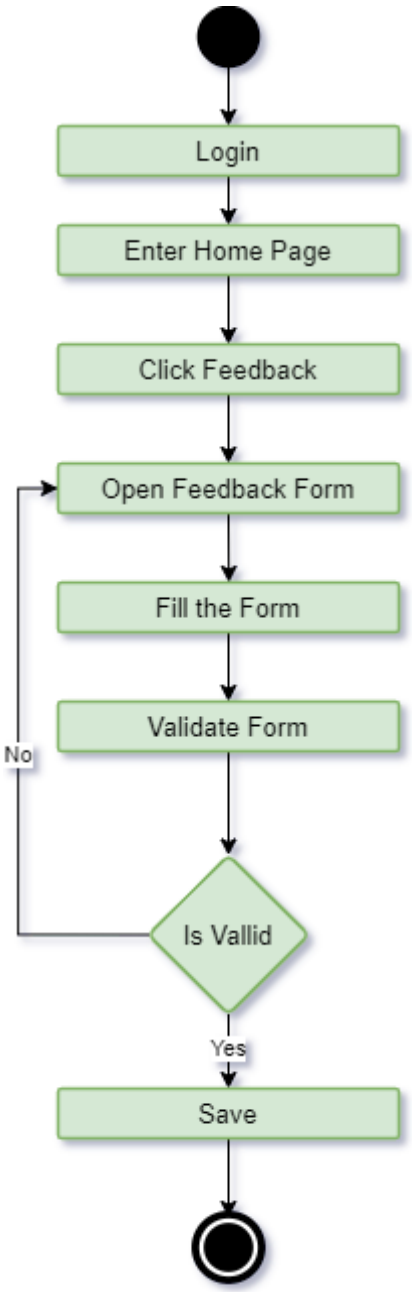


Figure 10: Customer Feedback

3.3.7 Manager: Item Create and Details View

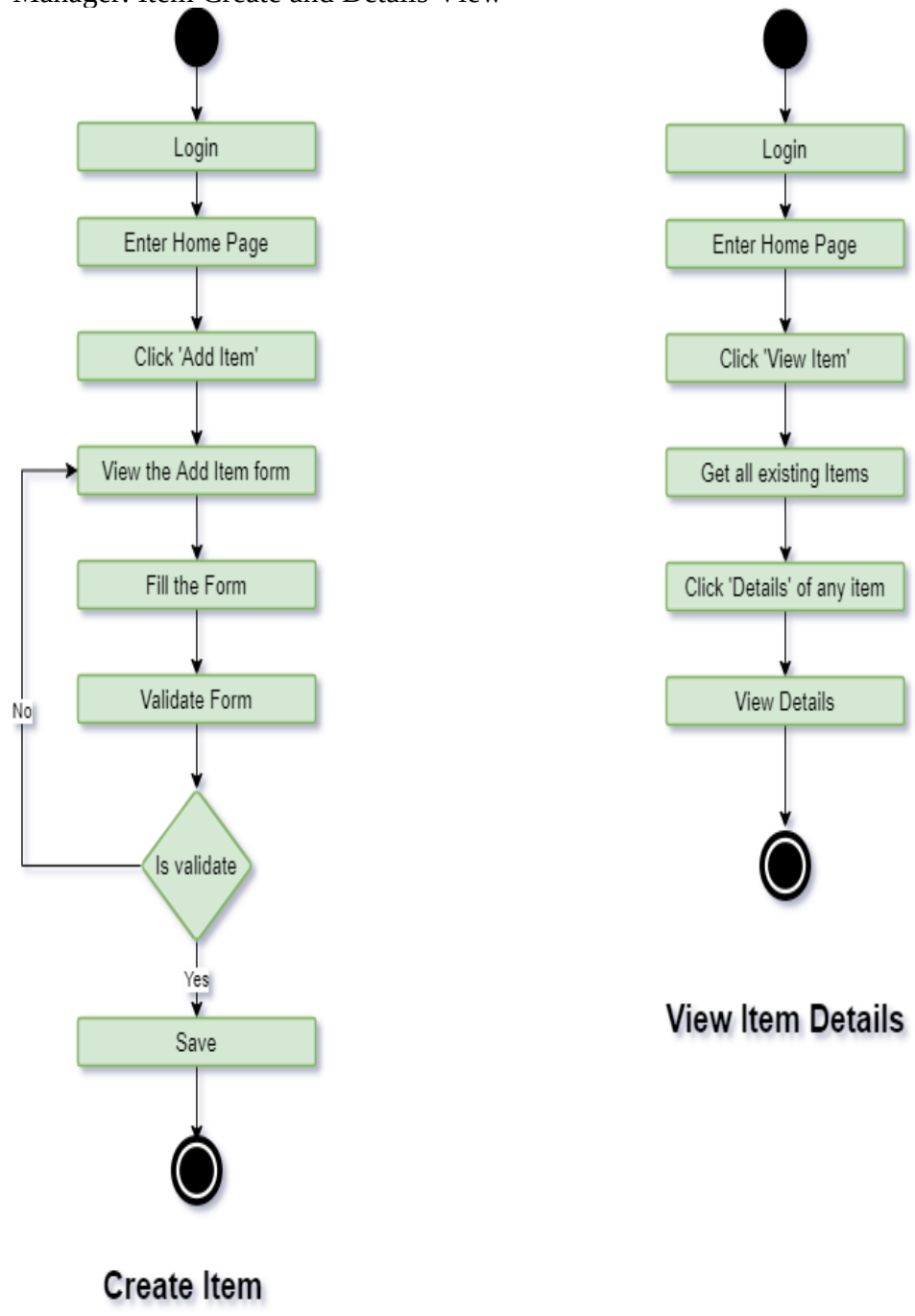


Figure 11: Create Item and View Details

3.3.8 Manager: Item Update, Item Delete

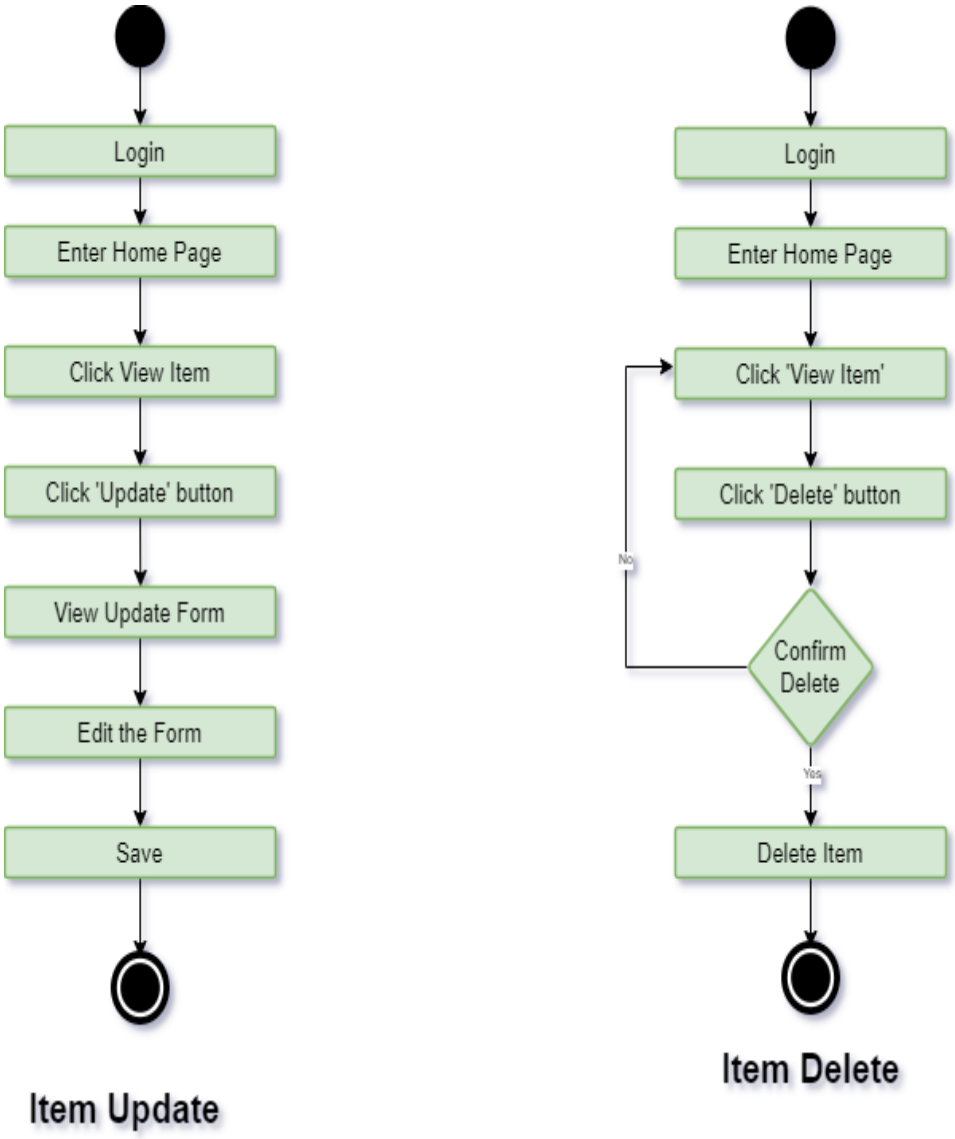


Figure 12: Item Update and delete by Manager

3.3.9 Manager View Order, Confirm Order, Delete Order

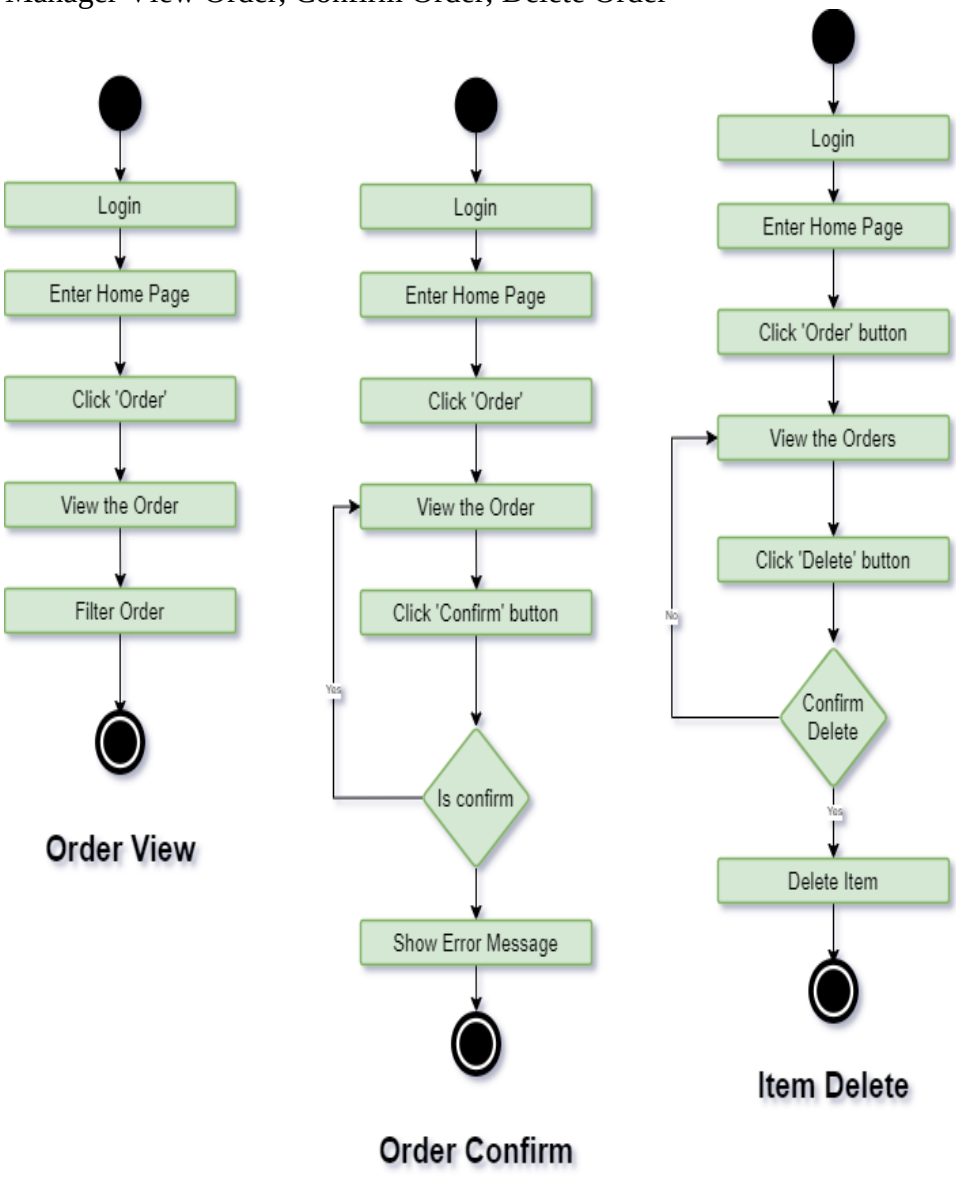


Figure 13: Manager View Order, Confirm Order, Delete Order

3.3.10 Manager Feedback View

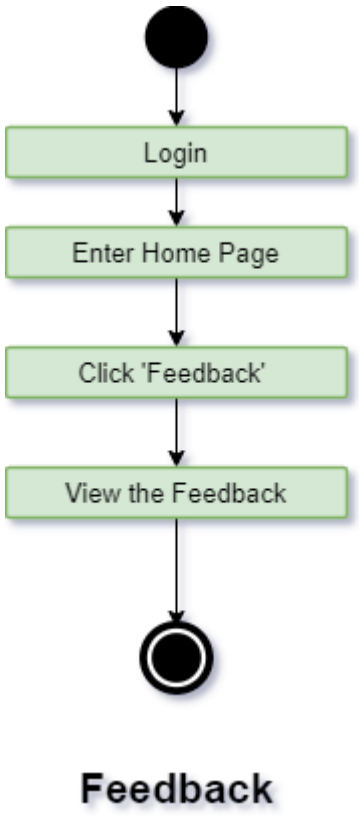
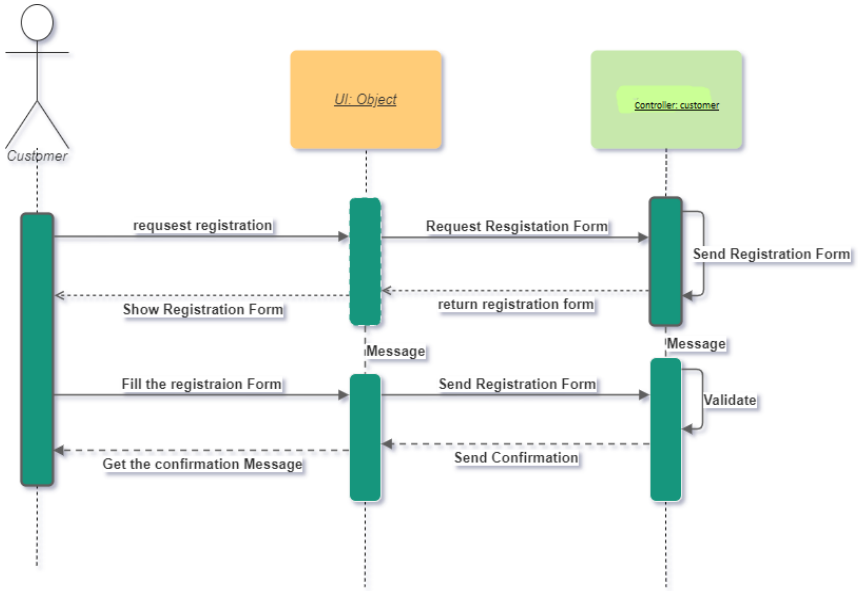


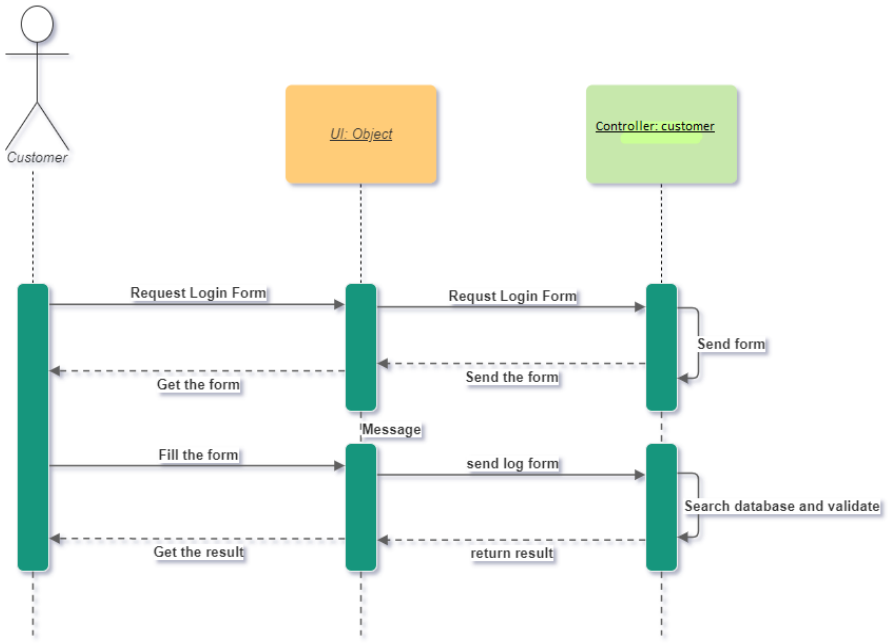
Figure 14: Manager Feedback View

Chapter 4: System Design Specification

4.1 Sequence Diagram



Customer Registration



Customer Login

Figure 15.1: Sequence diagram for Customer

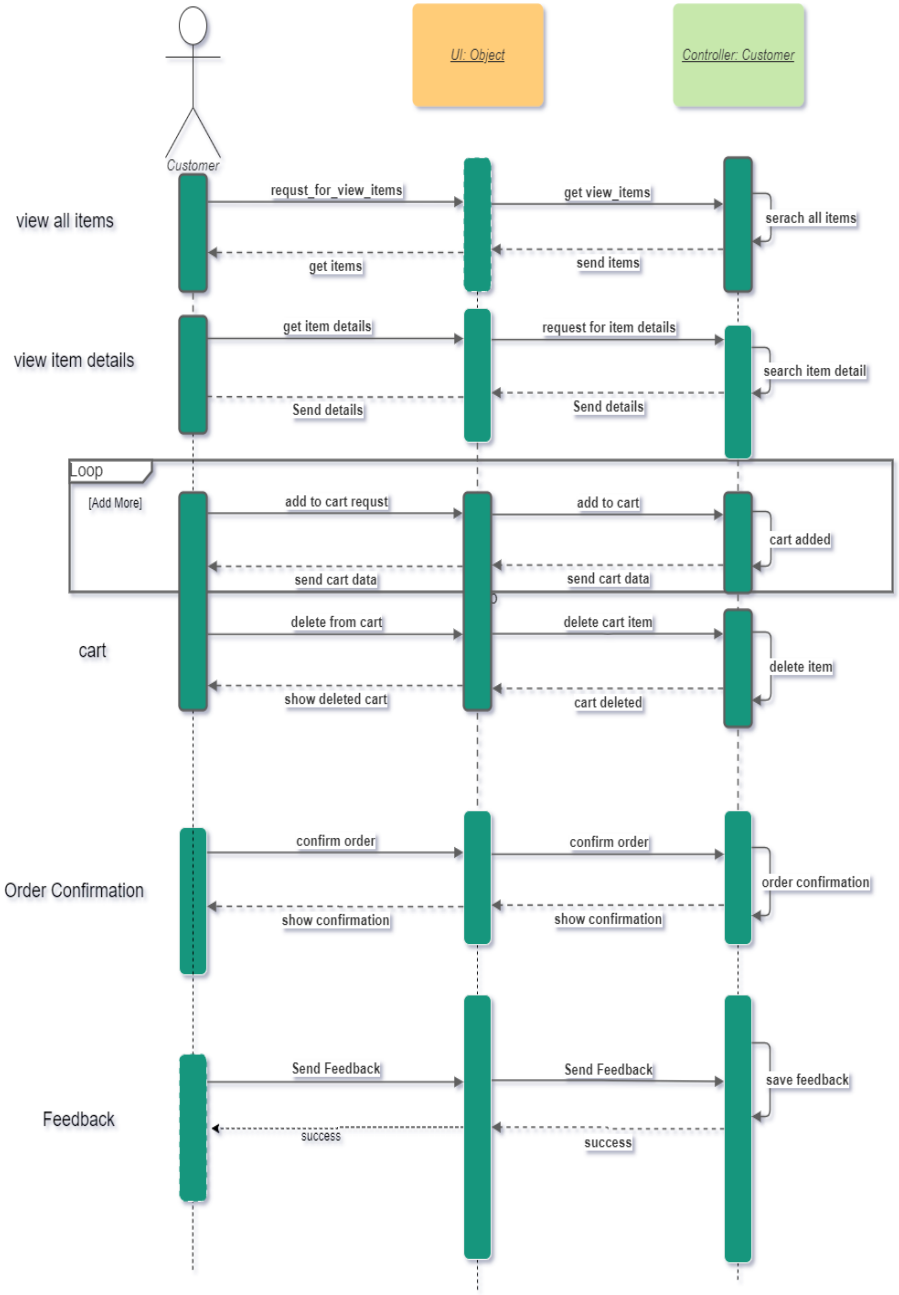


Figure 15.2: Sequence Diagram for Customer

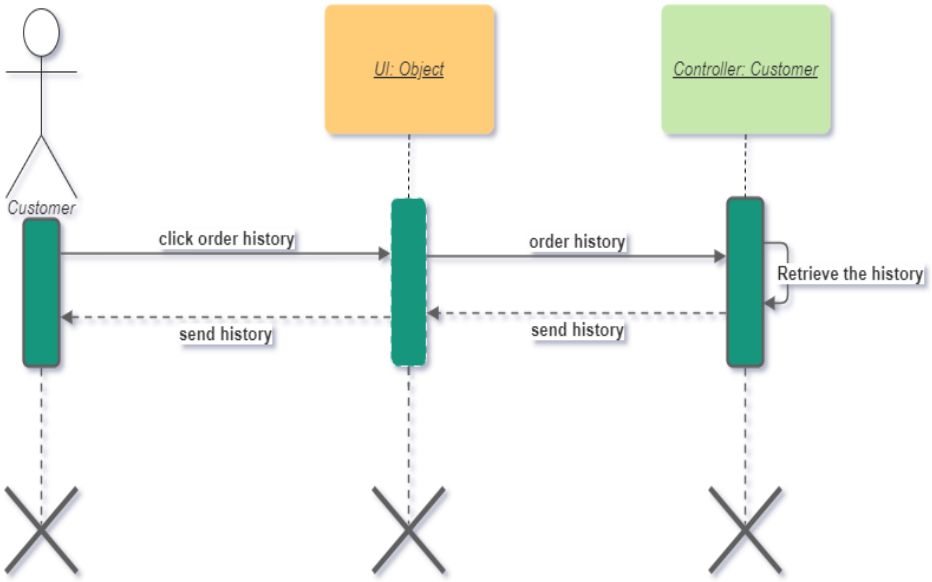


Figure 15.3: Sequence Diagram for Customer

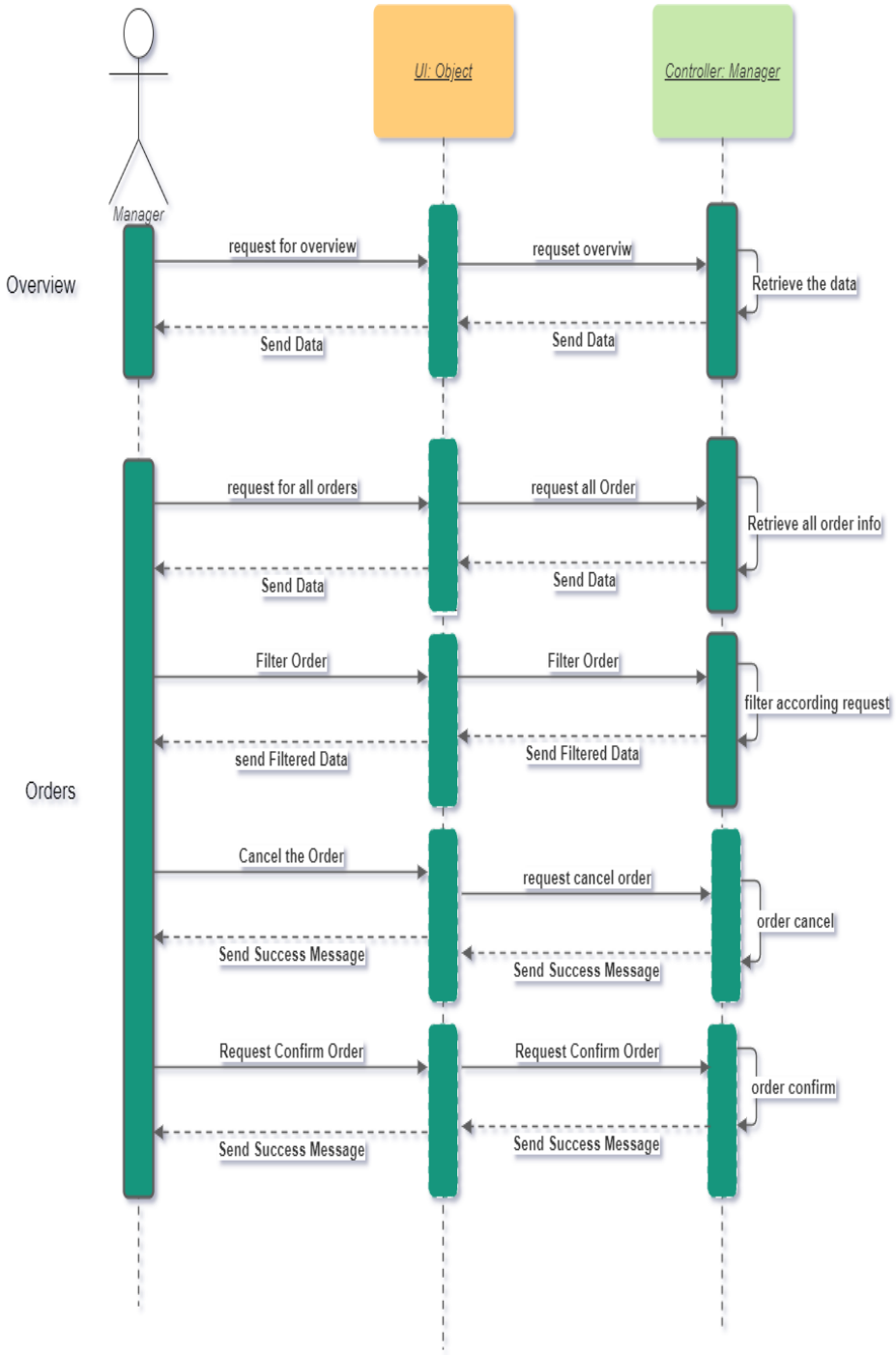


Figure 15.4: Sequence Diagram for Manager

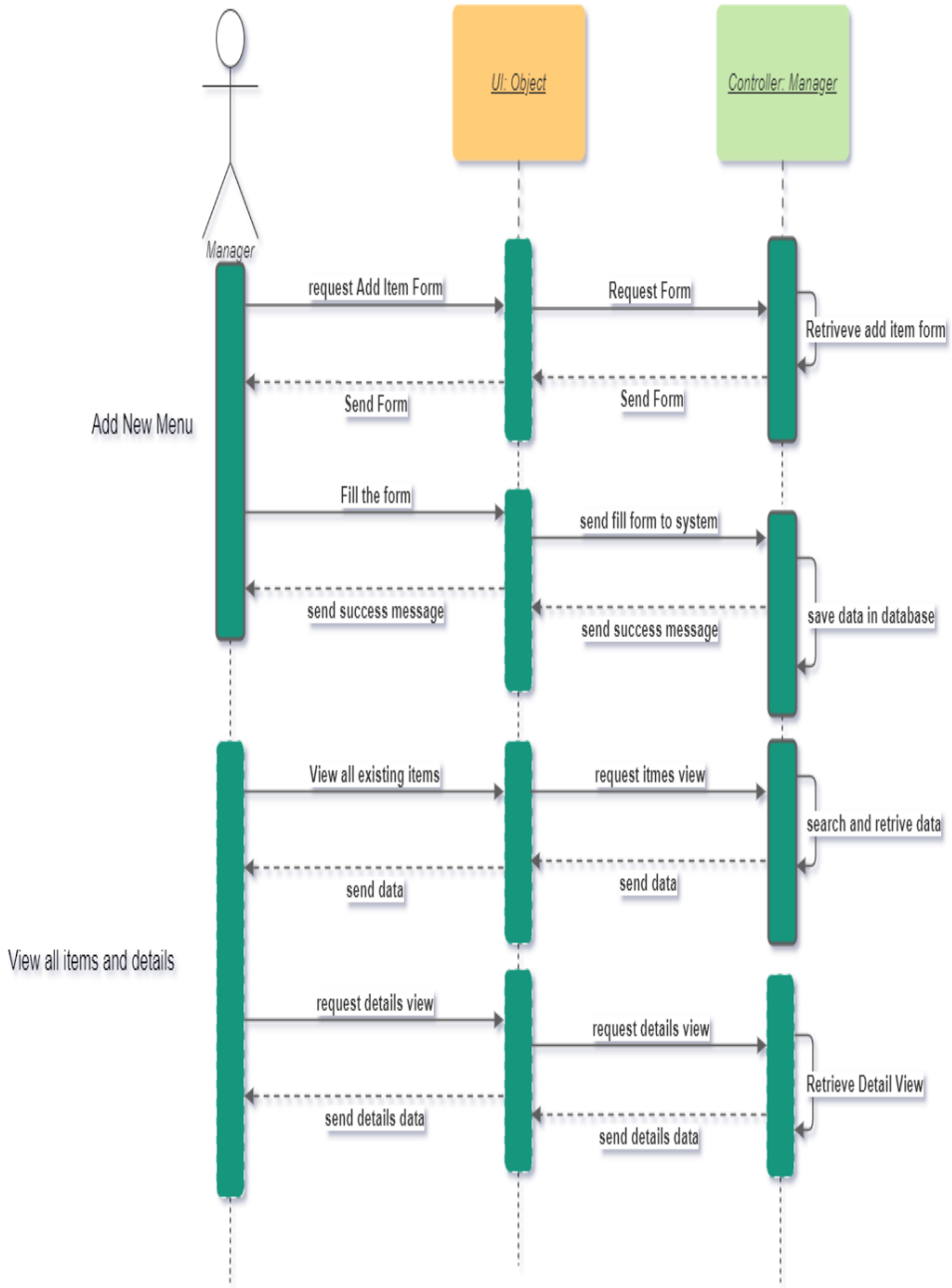


Figure 15.5: Sequence diagram for Manager

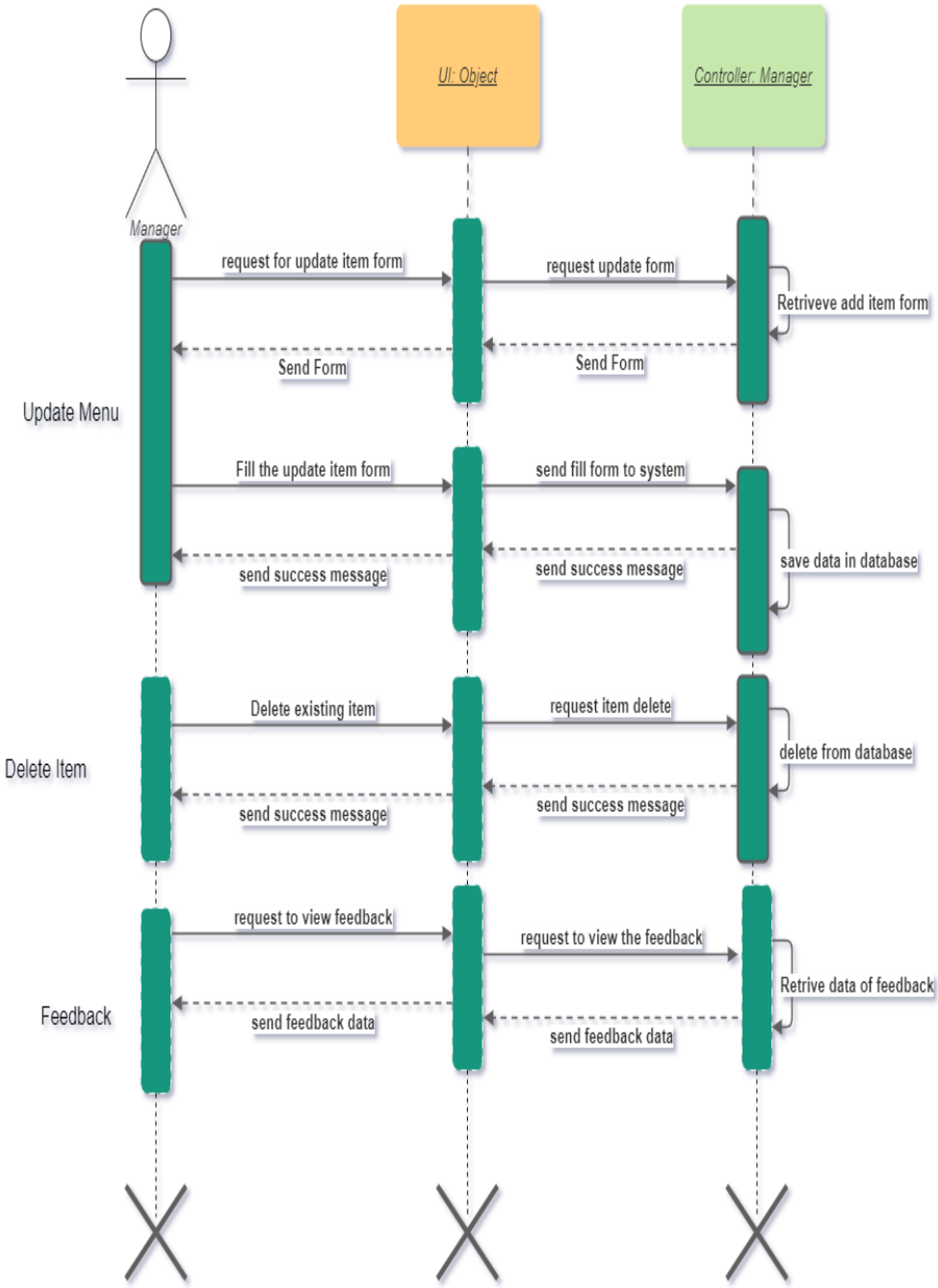


Figure 15.6: Sequence diagram for Manager

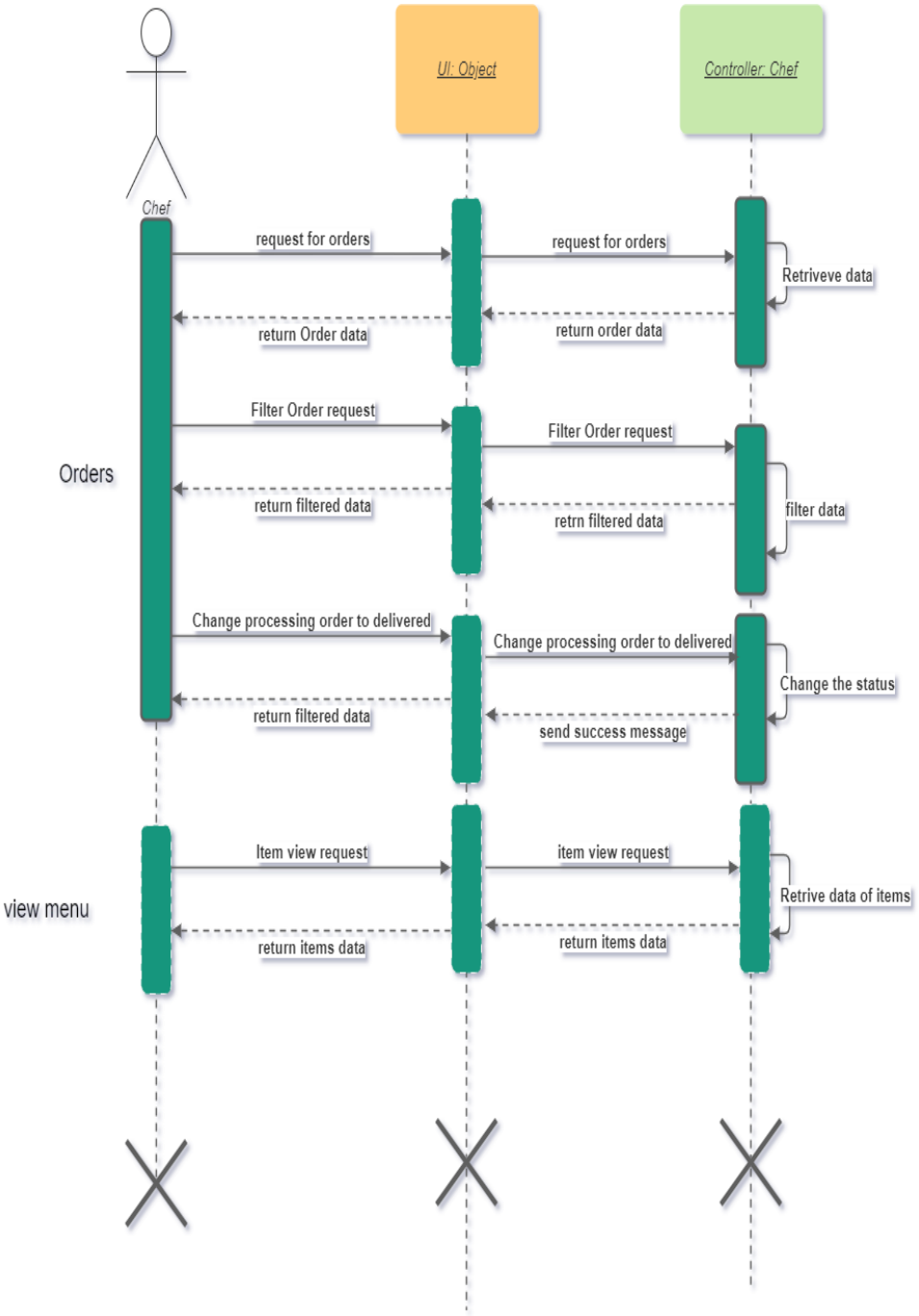


Figure 15.7: Sequence Diagram for Chef

4.2 CLASS DIAGRAM

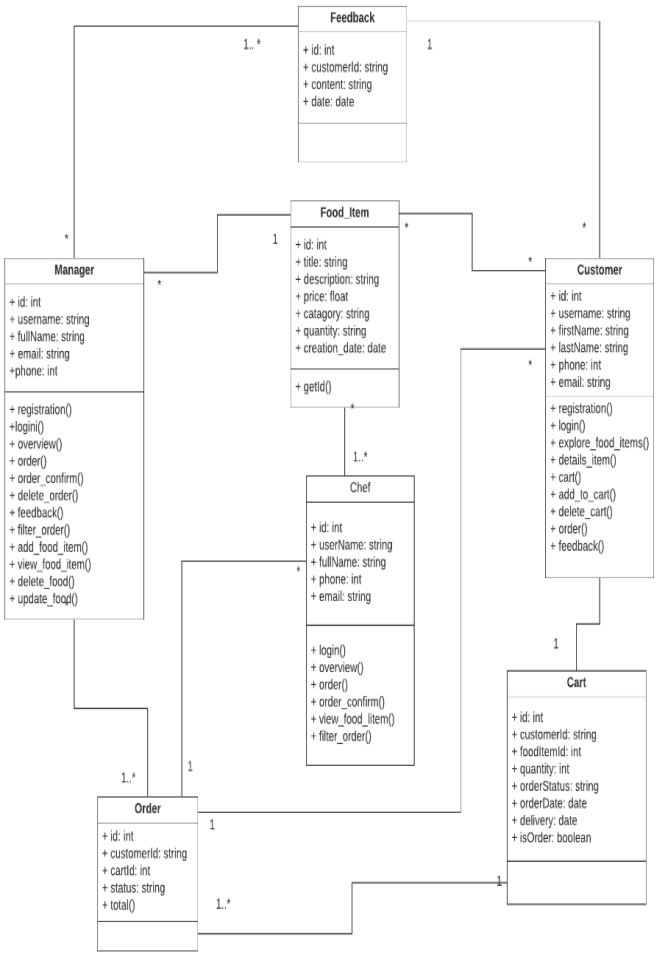


Figure 16: Class Diagram

4.3Entity Relationship Diagram

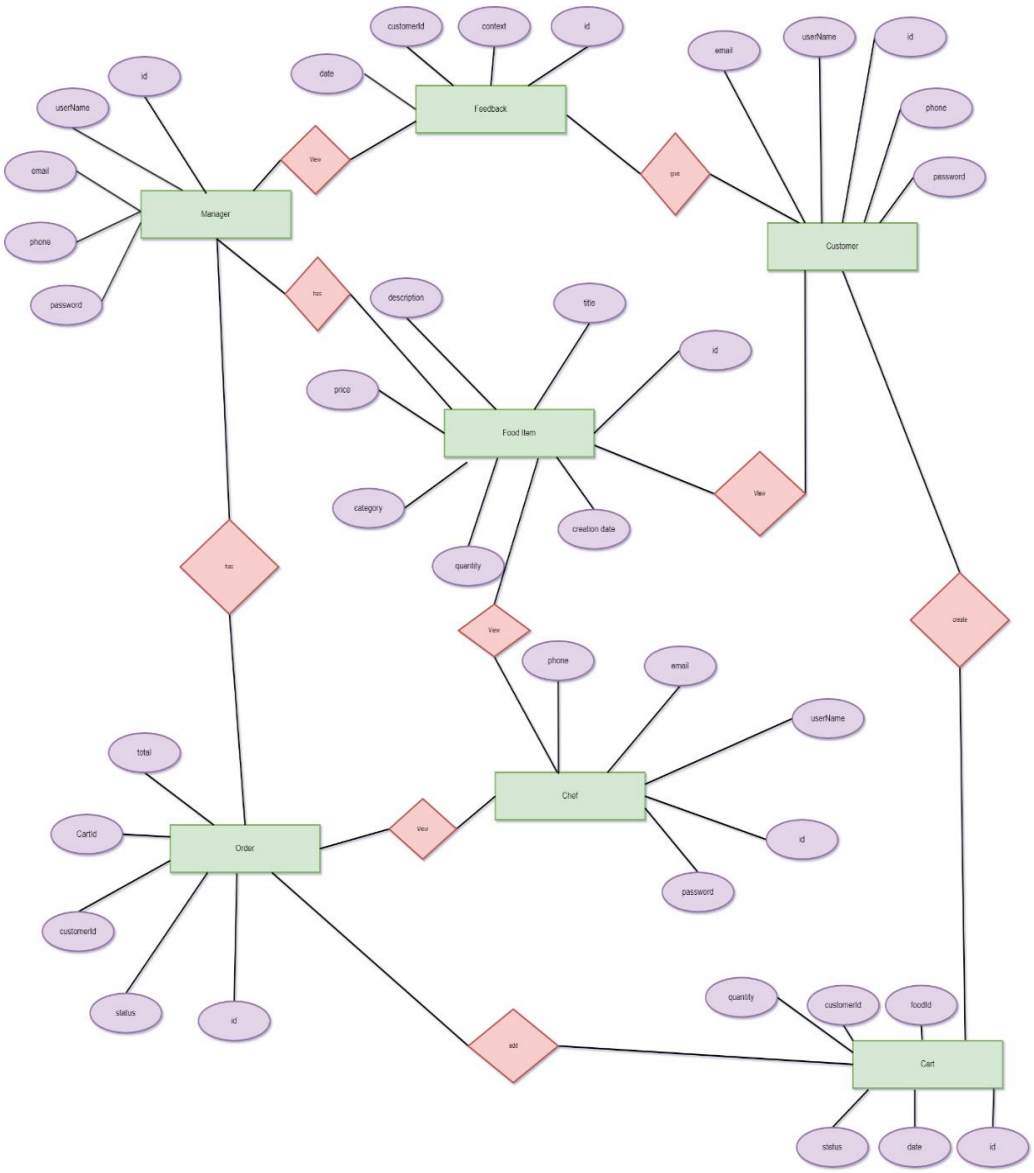


Figure 17: Entity Relationship Diagram

CHAPTER 5: USER INTERFACE

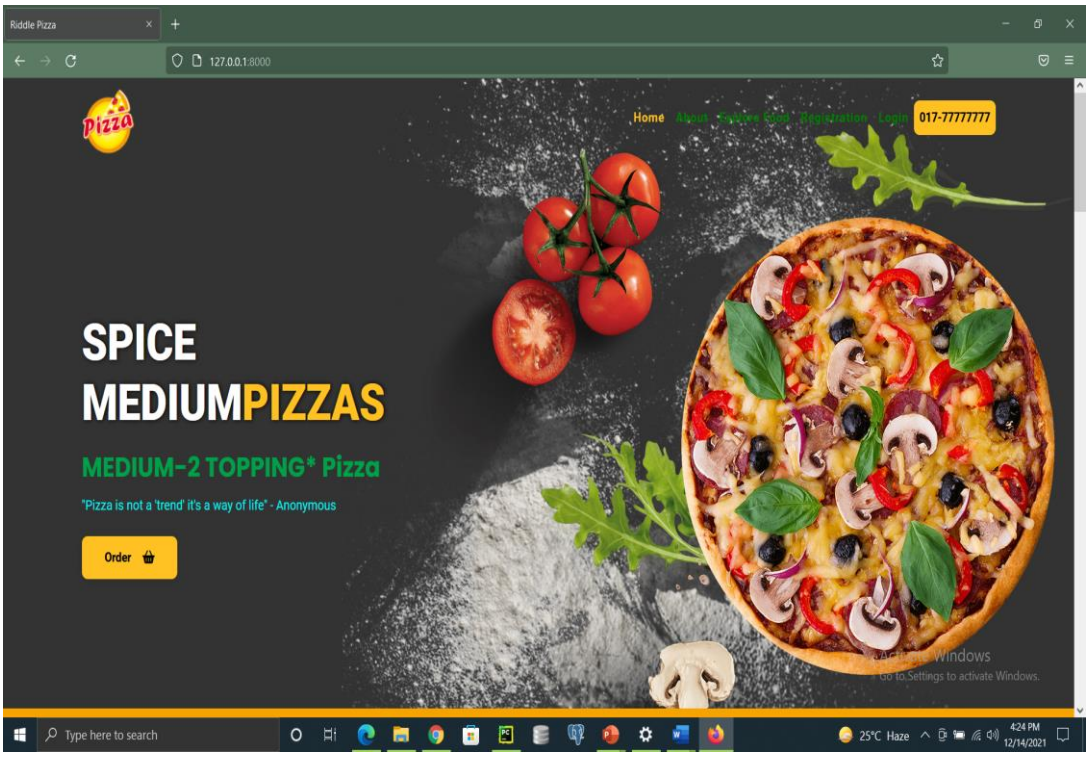


Figure 18 Home Page

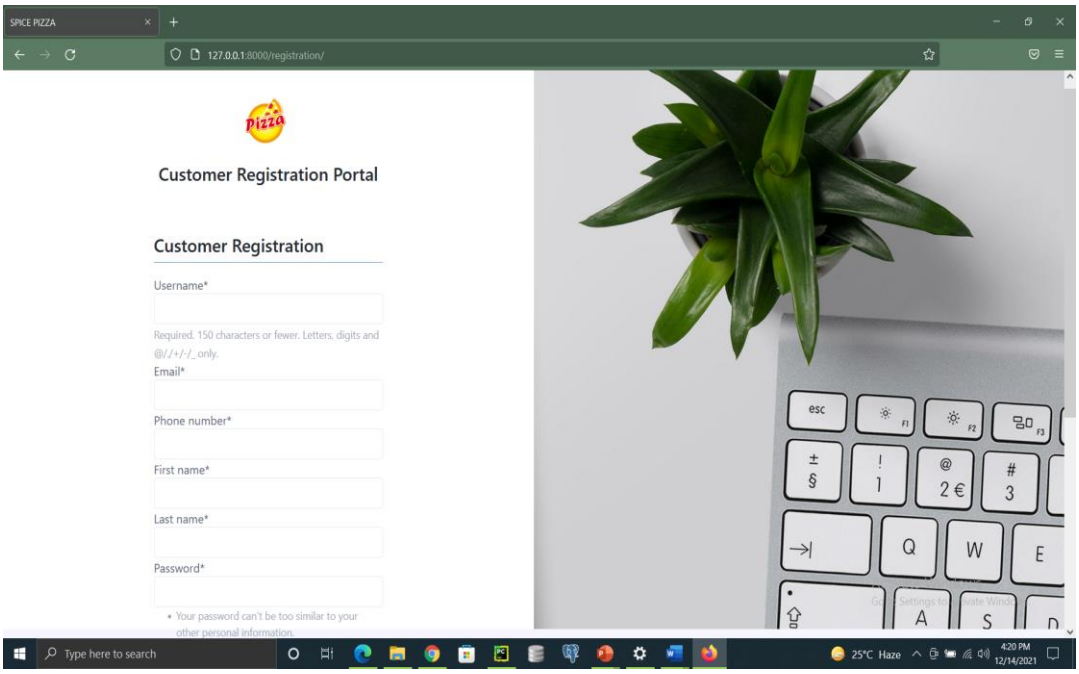


Figure 19: Customer Registration Page

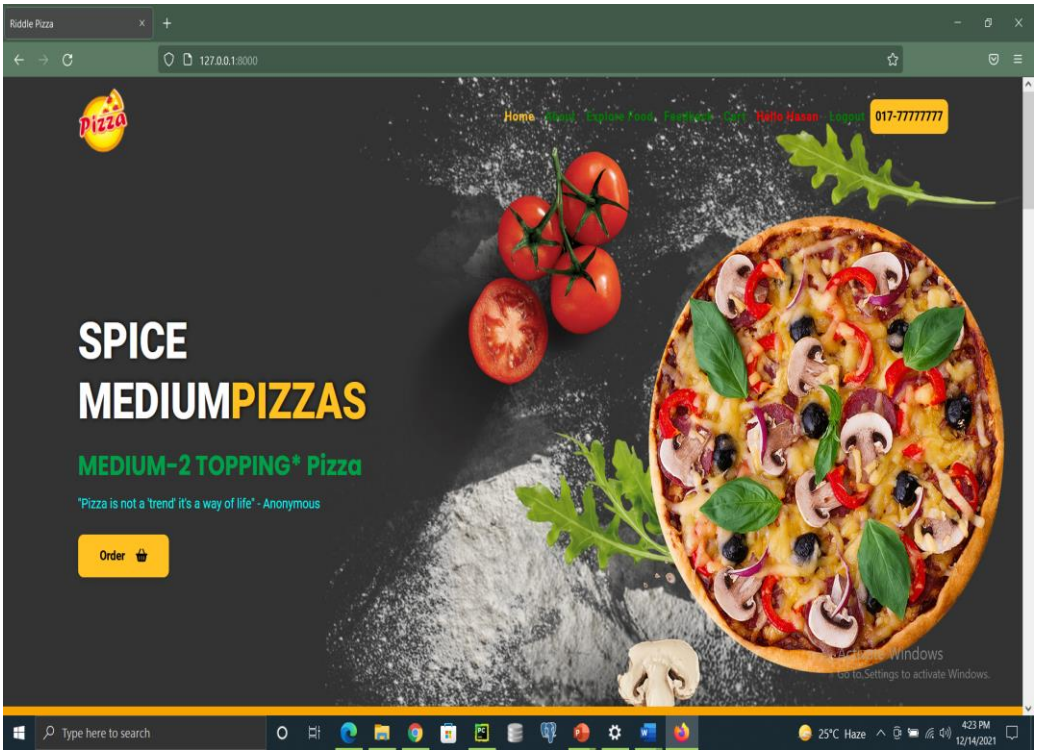


Figure 20: Customer Home page

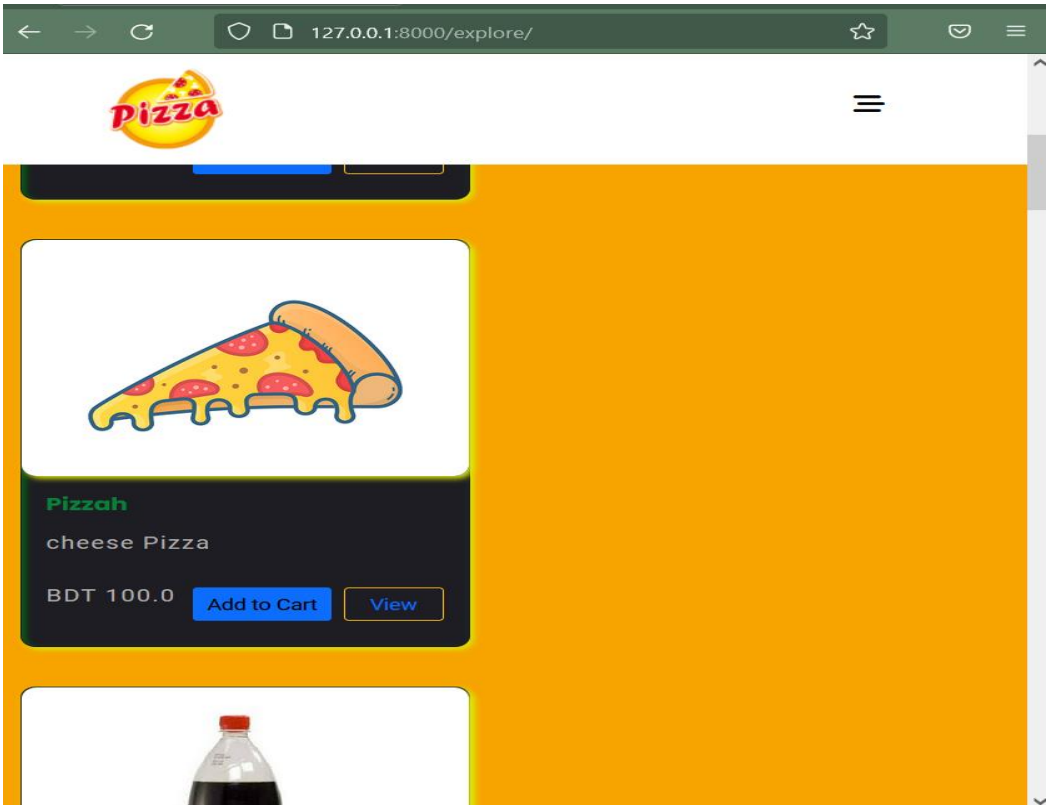


Figure 21: Customer Food View

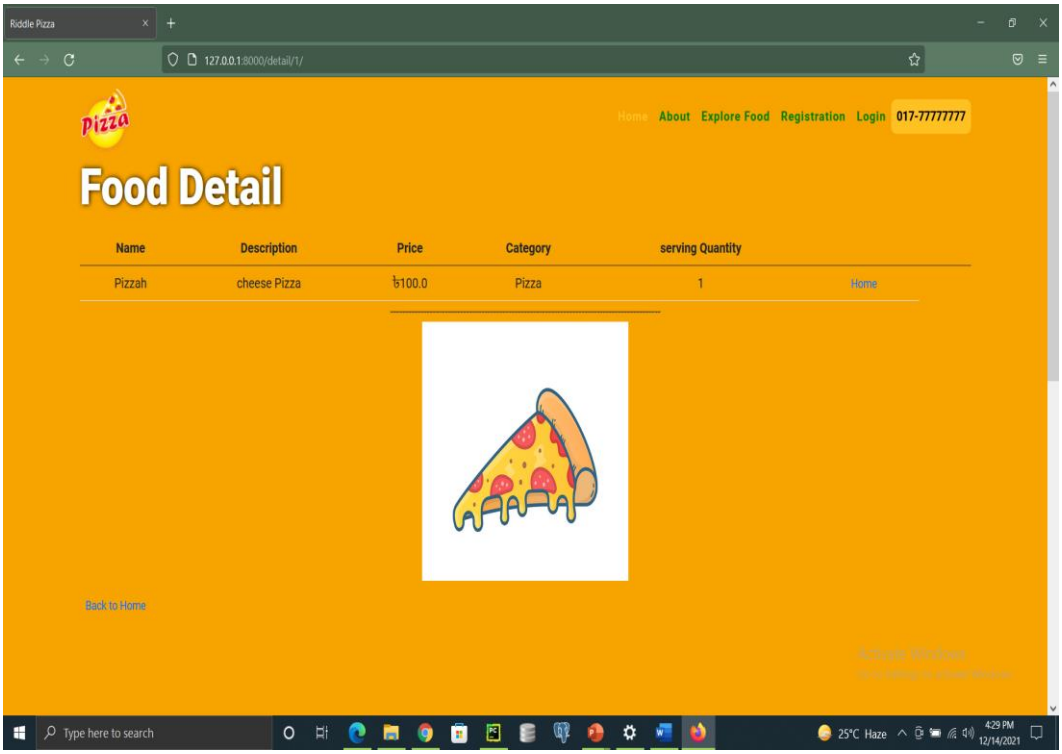


Figure22: View Food Details

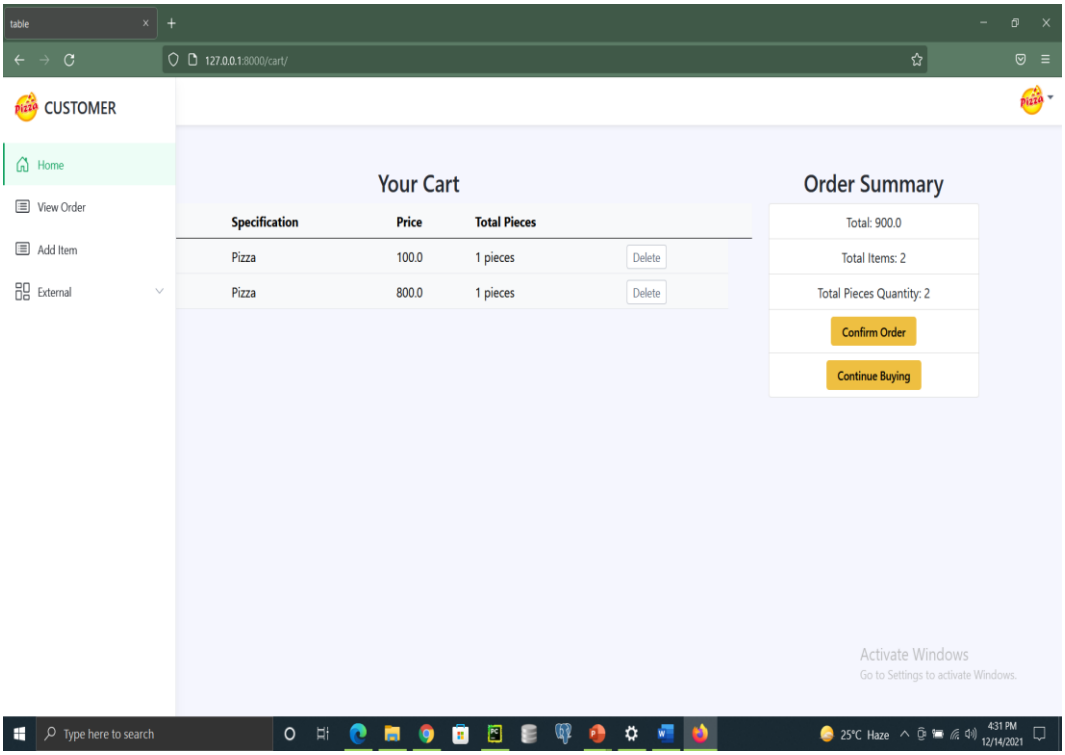


Figure 23: Customer Cart

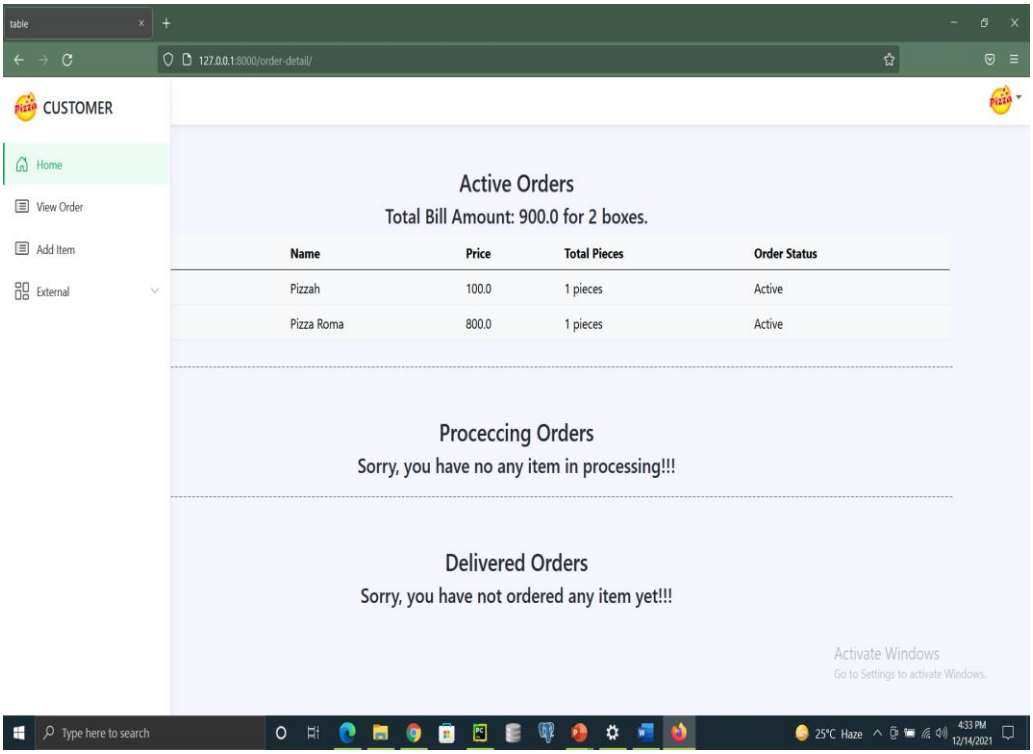


Figure 24: Customer Order Status

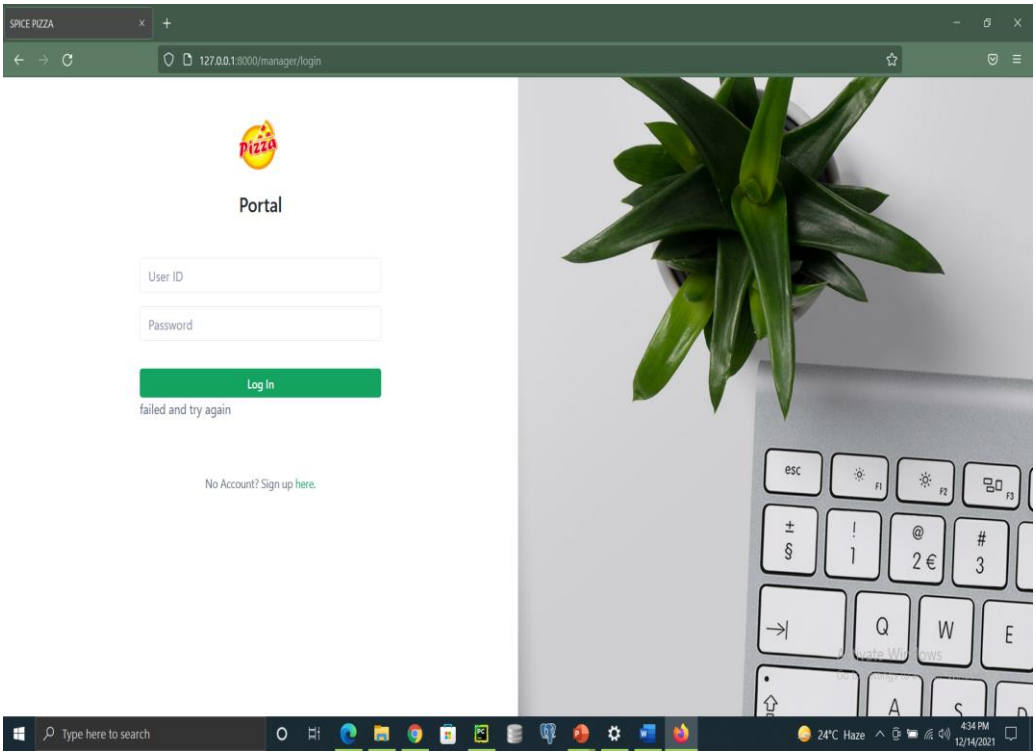


Figure 25: Manager Login Page

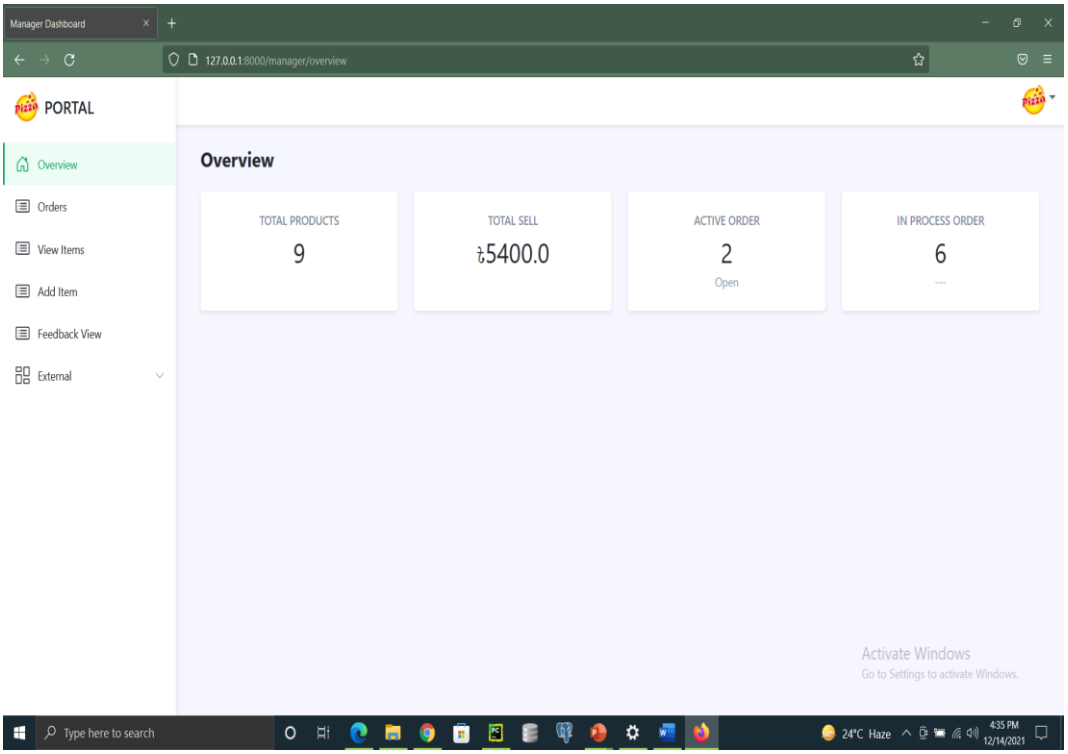


Figure 26: Manager Overview

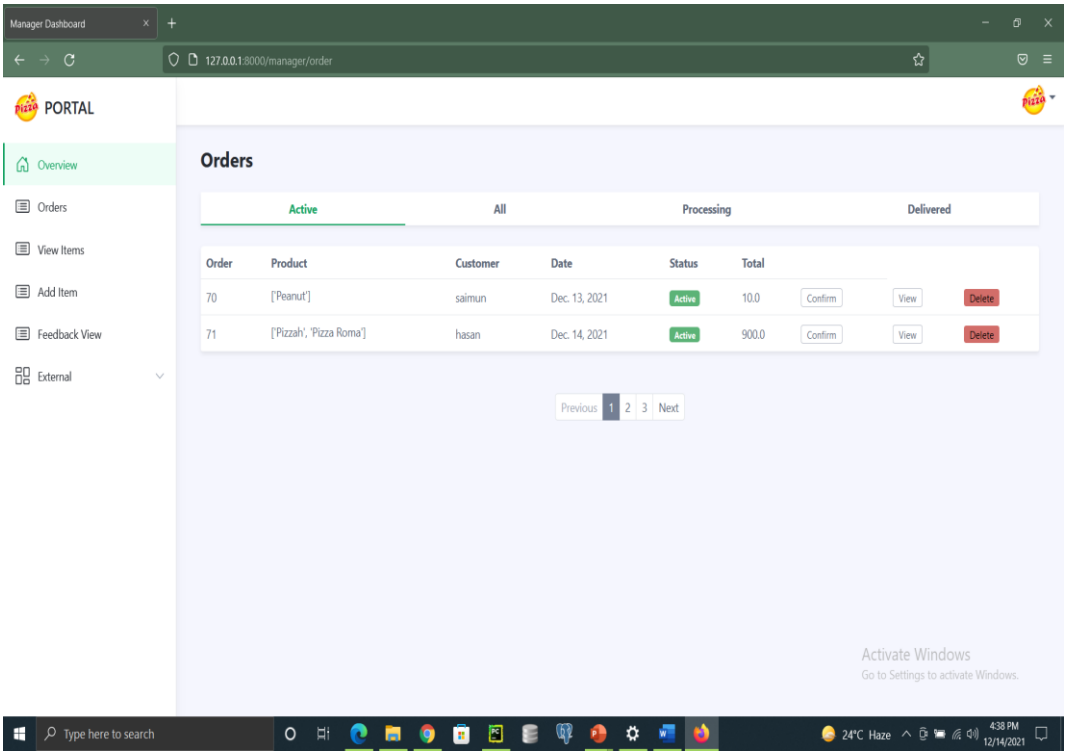


Figure 27: Manager active order view to confirm or delete

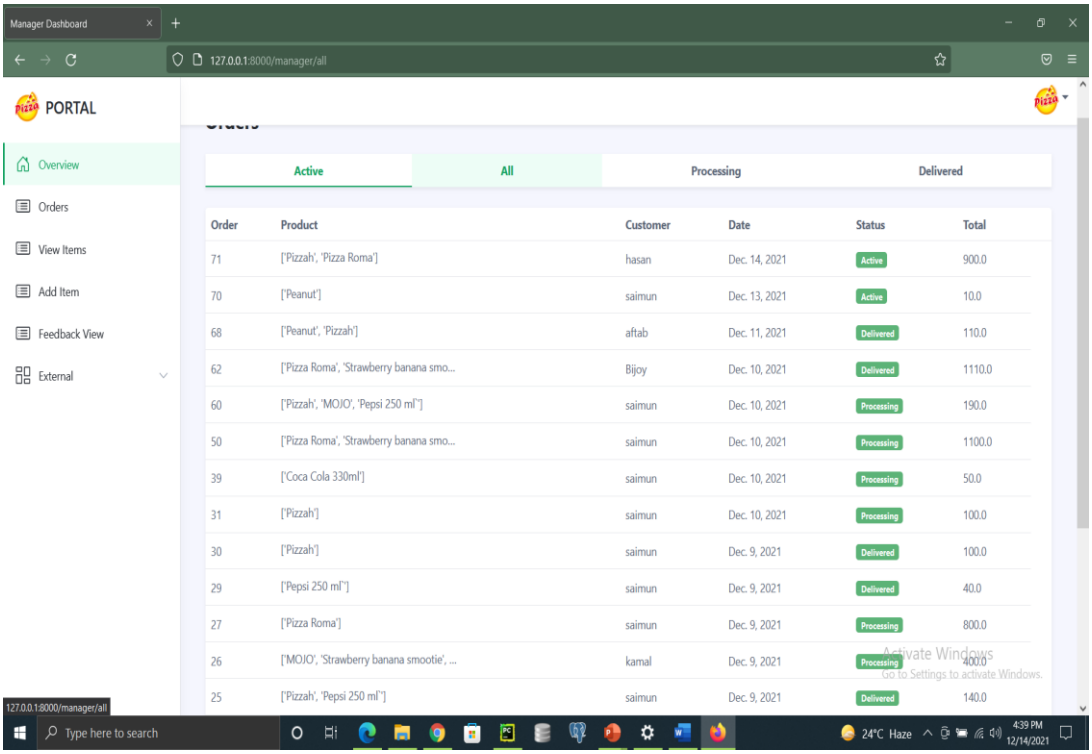


Figure 28: Manager view all order page

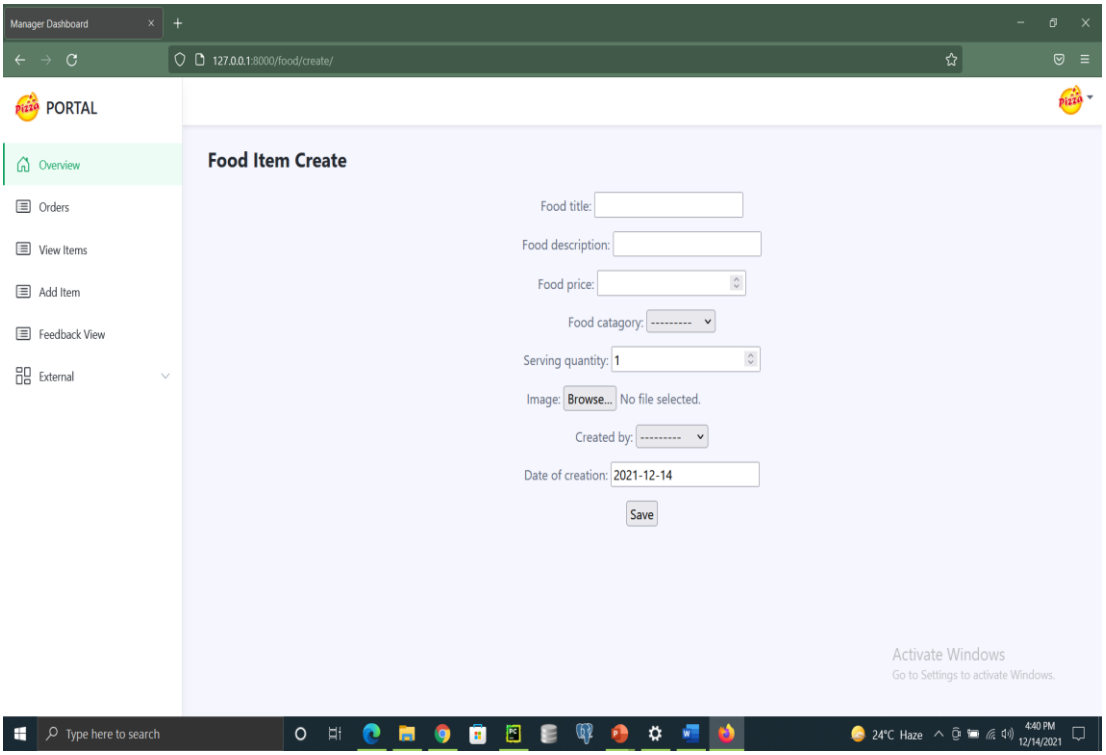


Figure 29: Manager add food form

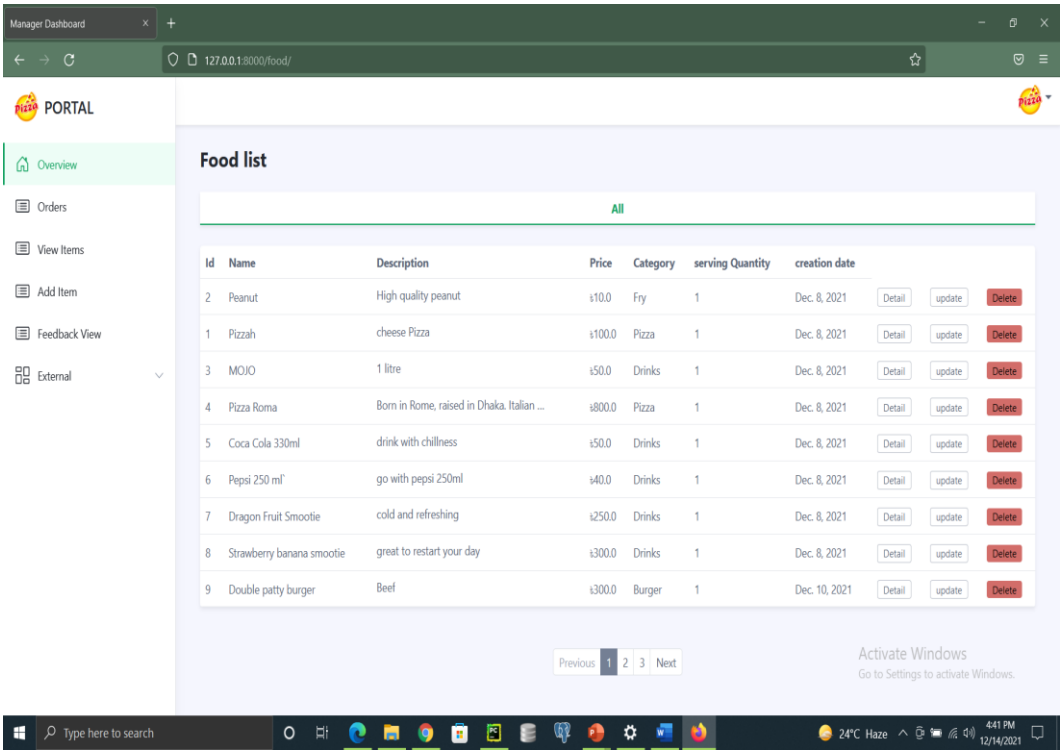


Figure 30: Manager view all saved item

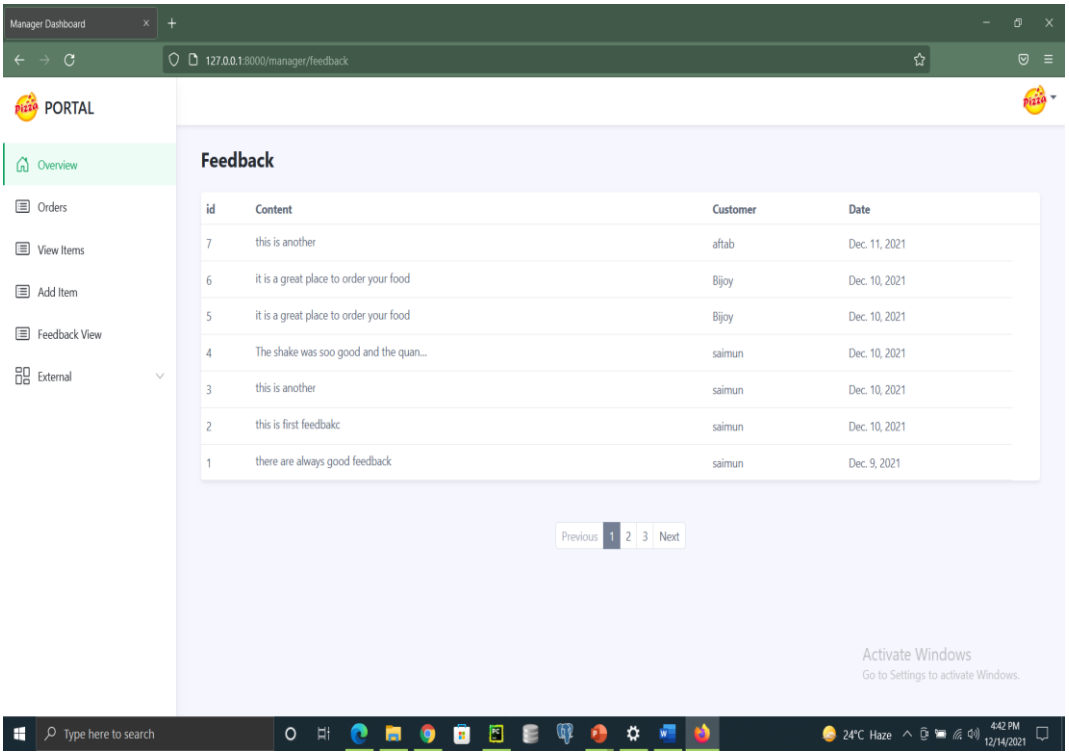


Figure 31: Manager Feedback View

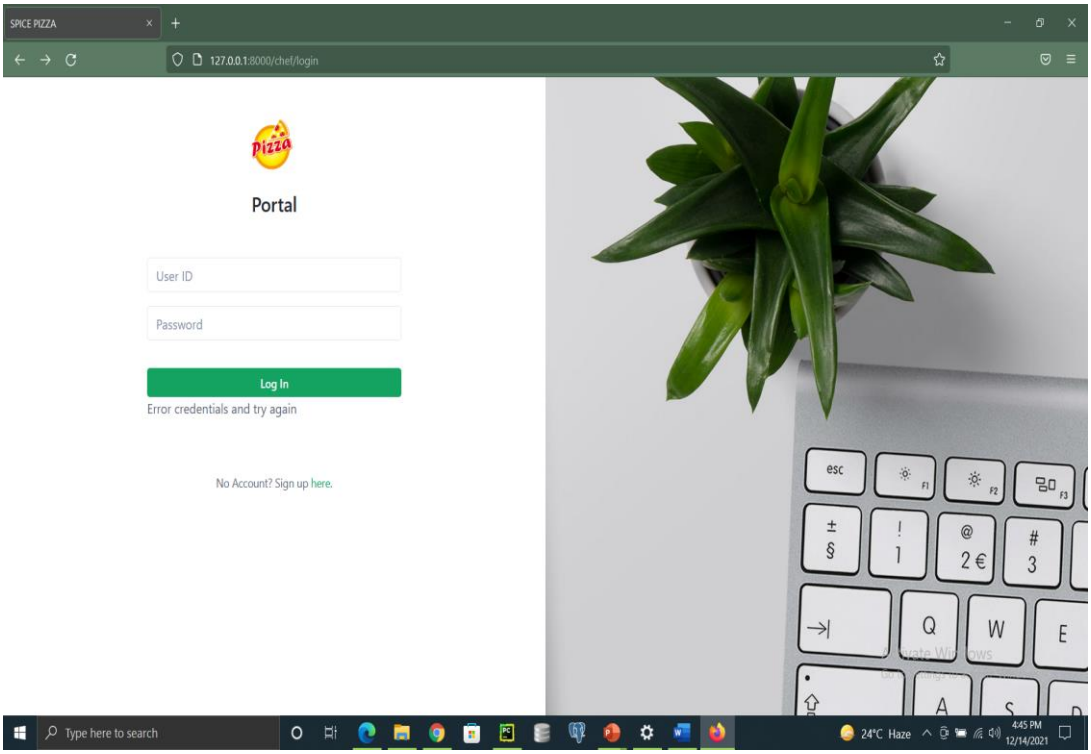


Figure 32: Chef Login Page

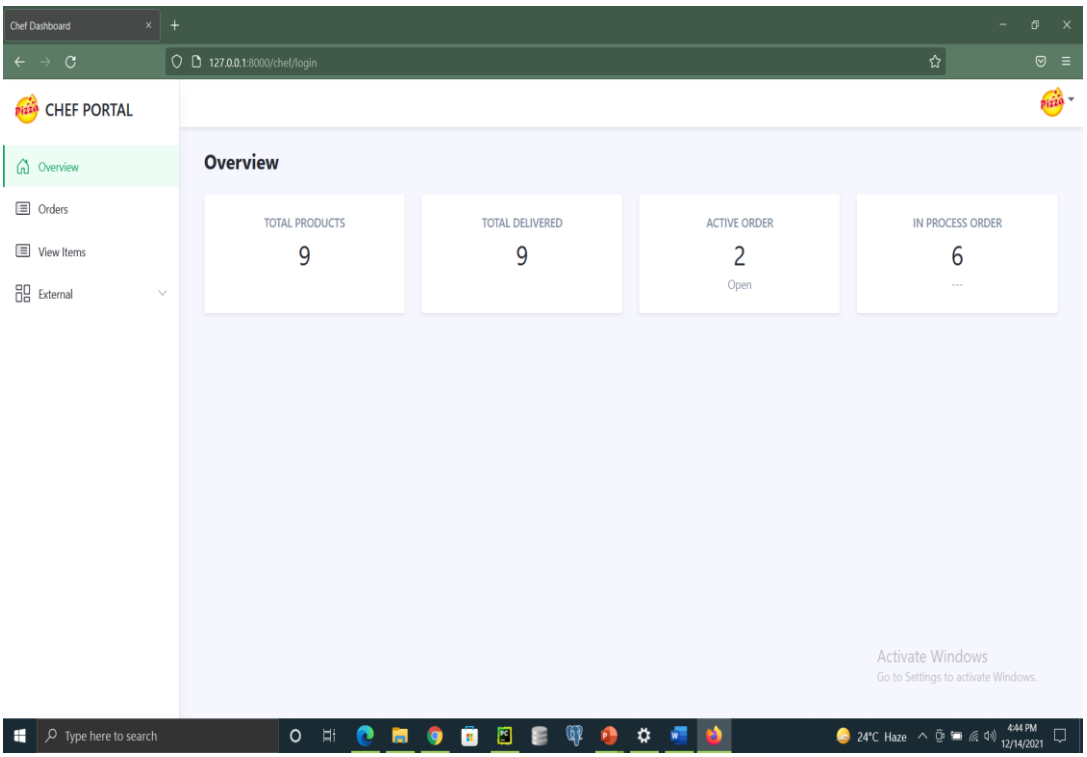


Figure 33: Chef Overview Page

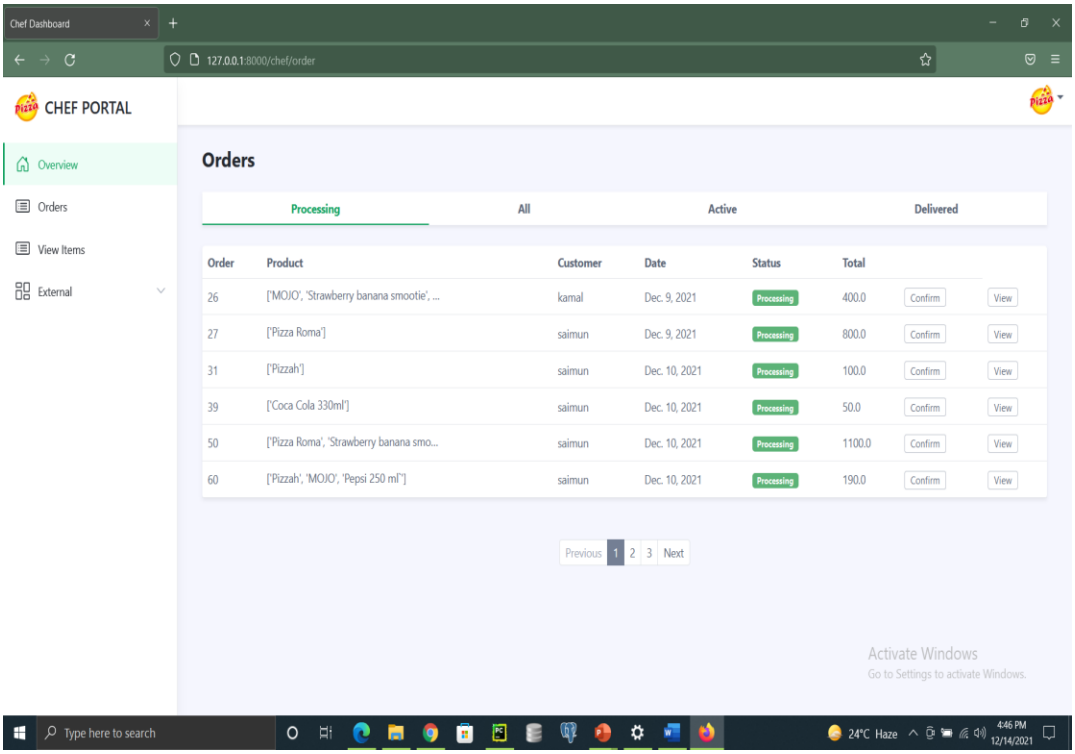


Figure 34: View Chefs Order

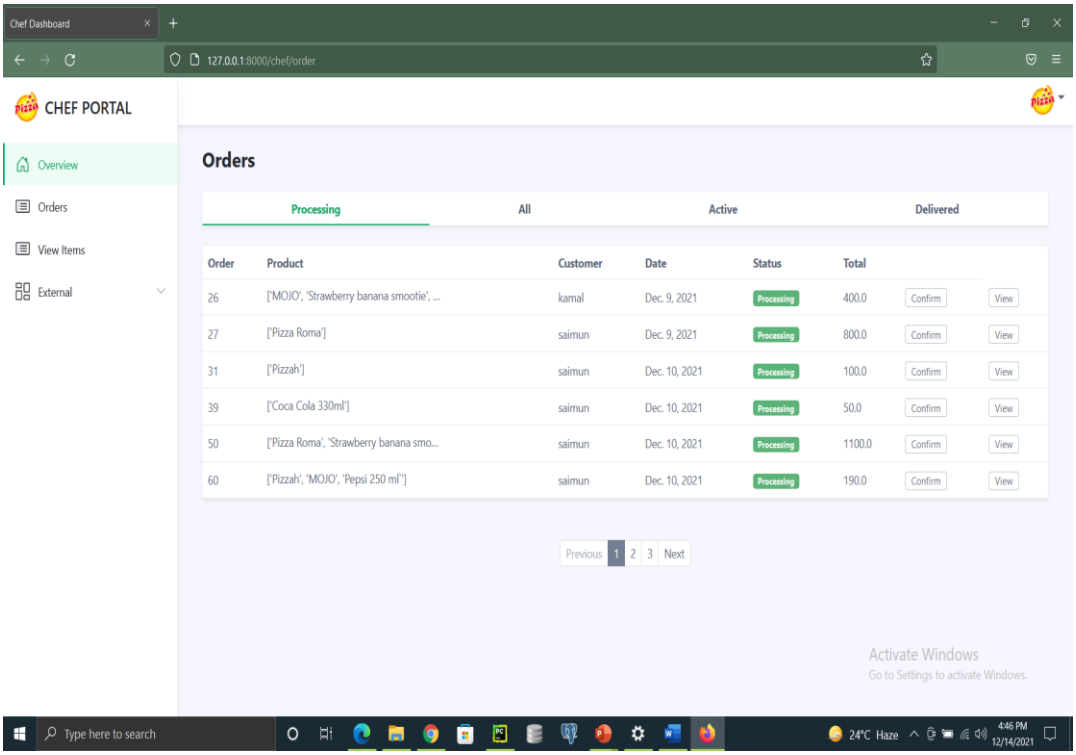


Figure 35: View All Items for Chef

CHAPTER 6: SYSTEM TESTING

6.1 Introduction

'Smart-Restaurant' system is developed for versatile types of users to work with efficiency and smartly. There are different types of users with different technology knowledge. Using the system Customer, manager and chef can work with synchronization. User can report any arising problem to admin to solve. For maintaining standard, we need to test the system to find out any bugs, errors. With testing we can also find out requirement missing component. For clear result a test plan is made with clear documentation. For better product test plan should be written as soon as requirements have been identified. We will also test our system with various sample data to see how it would handle input and output functions. We also test with extreme data and overload, which will directly slow the system that behaves in failure or extreme situations.

6.2Test Case Table

6.2.1 Test case 1 (Login)

<table>
<tr><td colspan="3">Test case #1</td><td colspan="2">Test case name: Login</td></tr>
<tr><td colspan="3">System: 'Smart-Restaurant' System</td><td colspan="2">Subsystem: N/A</td></tr>
<tr><td colspan="3">Design by: Saimun</td><td colspan="2">Design Date:</td></tr>
<tr><td colspan="3">Executed by: Saimun</td><td colspan="2">Executed Date:</td></tr>
<tr><td colspan="5">Short Description: Target of this case is that the user can login to the system.</td></tr>
<tr><td colspan="5">Precondition: User should be registered first</td></tr>
<tr><td>Step</td><td>Action</td><td>Response</td><td>Pass/Fail</td><td>Comment</td></tr>
<tr><td>1</td><td>Select user type Customer
Enter ID and Password</td><td>System Redirect to Customer Home Page</td><td>Pass</td><td></td></tr>
<tr><td>2</td><td>Select user type Manager
Enter ID and Password</td><td>System Redirect to Manager Home Page</td><td>Pass</td><td></td></tr>
<tr><td>3</td><td>Select user type Chef
Enter ID and Password</td><td>System Redirect to chef Home Page</td><td>Pass</td><td></td></tr>
<tr><td colspan="5">Post Condition: User can access to the system.</td></tr>
<tr><td colspan="5">Fail Case: If email and password do not match, users can not access the system.</td></tr>
</table>

6.2.2 Test case 2 (Registration)

Test case #2	**Test case name: Registration**
System: 'Smart-Restaurant'	**Subsystem: N/A**
Design by: Saimun	**Design Date:**
Executed by: Saimun	**Executed Date:**
Short Description: Target of this case is that the user can registration in the system.	
Precondition: User should be registered first	

Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Select user type Customer Fill the registration form	New Customer added	Pass	
2	Select user type Manager Fill the registration form	New Manager added	Pass	
3	Select user type Chef Fill the registration form	New Chef added	Pass	

Post Condition: User can login the system.
Fail Case: Cannot enter the system.

6.2.3 Manager Can Overview of the system

Test case #3		**Test case name: overview**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the overview of the system.				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the overview	View the overview	Pass	
Post Condition: show the dashboard				
Fail Case: show the error message.				

6.2.4 Manager Can view the orders

Test case #4		**Test case name: view orders**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the all types of orders in the system				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the order	Show order tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.5 Manager Filter the Order

Test case #5		**Test case name: Filter Orders**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the all types of orders in the system				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click button 'All', 'Active', 'Processing' and 'Delivered' to filter orders	Show order tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.6 Manager Confirm the Order

Test case #6		**Test case name: Confirm the Order**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can confirm the order made by customer				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'confirm' button	Confirmation completed	Pass	
Post Condition: show success messages				
Fail Case: show the error message.				

6.2.7 Manager Cancel the Order

Test case #7		**Test case name: Cancel the Order**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can cancel the orders made by the customer				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'cancel' button	Deleted Orders	Pass	
Post Condition: show the success message				
Fail Case: show the error message.				

6.2.8 Manager Add Food Items

Test case #8		**Test case name: Add Food Item**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can add new items in the system.				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'Add Item' button	Get the Add Form	Pass	
2	Manager Fill the form and press save	Save item in the system	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.9 Manager Can view Items

Test case #9		**Test case name: View the Food Items**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the all-Food items stored in the system.				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'view items'	Show the food item tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.10 Manager Can view the item details

Test case #10		**Test case name: view the item details**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the details of saved item				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'details' button	Show the details	Pass	
Post Condition: show the details				
Fail Case: show the error message.				

6.2.11 Manager Can update the item

Test case #11		**Test case name: update the item**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can update the item which exist in the system				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click 'update' button	Send the update form	Pass	
2	Update the fill form	Send fill form	Pass	
Post Condition: show the updated item				
Fail Case: show the error message.				

6.2.12 Manager delete the item

Test case #12		**Test case name: delete item**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can delete the item				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'delete' button	Show success message	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.13 Manager Can view the Feedback

Test case #13		**Test case name: Feedback**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Manager can view the all feedback made by customers				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Manager Click the 'feedback' button	Show feedback table	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.14 Chef Can view the overview

Test case #14		**Test case name: overview**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can overview in the system				
Precondition: Chef Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click 'overview' button	View the dashboard	Pass	
Post Condition: show the all data				
Fail Case: show the error message.				

6.2.15 Chef Can view the overview

Test case #15		**Test case name: overview**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can overview in the system				
Precondition: Chef Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click 'overview' button	View the dashboard	Pass	
Post Condition: show the all data				
Fail Case: show the error message.				

6.2.16 Chef Can view the orders

Test case #16		**Test case name: view orders**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can view the all types of orders in the system				
Precondition: Chef Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click the order	Show order tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.17 Chef Filter the Order

Test case #17		**Test case name: Filter Orders**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can view the all types of orders in the system				
Precondition: Chef Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click button 'All', 'Active', 'Processing' and 'Delivered' to filter orders	Show order tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.18 Chef Confirm the Order

Test case #18		**Test case name: Confirm the Order**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can confirm the order processing to delivered				
Precondition: Manager Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click the 'confirm' button	Confirmation completed	Pass	
Post Condition: show success messages				
Fail Case: show the error message.				

6.2.19 Chef Can view Items

Test case #19		**Test case name: View the Food Items**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Chef can view the all-Food items stored in the system.				
Precondition: Chef Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Chef Click the 'view items'	Show the food item tables	Pass	
Post Condition: show the table				
Fail Case: show the error message.				

6.2.20 Customer view Items and details

Test case #20		**Test case name: View the Food Items**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Customer can view the all-Food items stored in the system.				
Precondition: Manager Must be Logged in or without log in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Customer can view all existing items	View all items	Pass	
2	View details	View details of item	Pass	
Post Condition: show the product card				
Fail Case: show the error message.				

6.2.21 Customer: cart

Test case #21		**Test case name: cart**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Customer can add items in the cart and delete				
Precondition: Customer Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Customer click the 'add cart'	Show the cart items	Pass	
2	Delete from cart	Show success message	Pass	
Post Condition: show the cart				
Fail Case: show the error message.				

6.2.22 Customer: Order

Test case #22		**Test case name: order**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Customer can order from cart				
Precondition: Customer Must be Logged in and cart must have value				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Customer click the 'confirm-order' form cart	Order completed	Pass	
2	Click order view	View the order message	Pass	
Post Condition: show the orders				
Fail Case: show the error message.				

6.2.23 Customer: Feedback

Test case #23		**Test case name: Feedback**		
System: 'Smart-Restaurant' System		**Subsystem: N/A**		
Design by: Saimun		**Design Date:**		
Executed by: Saimun		**Executed Date:**		
Short Description: Customer can write feedback in the system				
Precondition: Customer Must be Logged in				
Step	**Action**	**Response**	**Pass/Fail**	**Comment**
1	Customer click 'Feedback' button	Show the feedback form	Pass	
2	Fill the form	Show success message	Pass	
Post Condition: Feedback added in the database.				
Fail Case: show the error message.				

CHAPTER 7: CONCLUSION

7.1 GitHub Link

https://github.com/Abdulla-Saimun/smart-restaurant-2.git

7.2 Project Summary

I started working on this project since July 2021. My goal was to ensure make a smart restaurant system for customer, manager and chef.

7.3 Limitation

I have tried my best to make the application better and fulfill requirements but some of them were not possibly make for time shortage.

- Add a payment method
- Add inventory

7.4 Future Scope

In future we can emphasize the system. All restaurant can use the system to expand their business and get the great customer experience.

REFERENCES

1. How To: Write a Project proposal (Online) URL: https://www.mavenlink.com/resources/project-proposal https://docs.djangoproject.com/en/3.1/.
2. https://www.tutorialspoint.com/index.htm
3. https://www.w3schools.com/
4. https://www.sitepoint.com/writing-software-documentation/
5. https://guides.lib.berkeley.edu/how-to-write-good-documentation
6. https://stackoverflow.com/
7. https://www.youtube.com/watch?v=F5mRW0jo-U4&t=151s
8. https://www.youtube.com/watch?v=OTmQOjsl0eg&t=7649s