Sri Lanka Institute of Information Technology

Online cake ordering system

Project Report

Information Systems Project 2024

Project ID: 2024_S2_05

Submitted by: Pooliyadda T.M.V.M.B

IT22134844	Pooliyadda T.M.V.M.B
IT22214652	Wathsani S.A.D.M.D.
IT22098696	Dissanayake D.M.L.A
IT22242372	Siriwardana L.G.A.P

Submitted to: Mrs. Narmada Gamage

……..

Mrs. Narmada Gamage

Date of submission: 28/11/2025

Abstract

This project involved developing a website for Cake Town Bakers, a bakery that previously operated only through a Facebook account. The new website provides customers with an easy way to browse and order a variety of cakes online. Customers can securely pay using their cards, ensuring a smooth and safe transaction process. Additionally, the website includes a review and rating system that allows customers to share feedback and rate the cakes they purchase. This project aims to enhance Cake Town Bakers' online presence, offering a more convenient and accessible way for customers to place orders and improving overall customer satisfaction.

Acknowledgement

We would like to express our heartfelt gratitude to all those who have supported and guided us throughout the development of this project, the Online Cake Ordering System.

We extend our sincere thanks to Ms.Narmada Gamage and Ms.Buddima Attanayake for their invaluable guidance, constructive feedback, and encouragement during this project. Their expertise and support were instrumental in shaping the project's success.

Declaration

We declare that the this project report or part of it was not a copy of a document done by any organization, university any other institute or a previous student project group at SLIIT and was not copied from the Internet or other sources.

Project Details

Project Title	Online cake ordering system
Project ID	2024_S2_05

Group Members

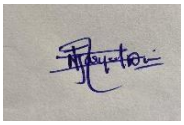
Reg. No	Name	Signature
IT22134844	Pooliyadda T.M.V.M.B	
IT22214652	Wathsani S.A.D.M.D.	
IT22098696	Dissanayake D.M.L.A	
IT22242372	Siriwardana L.G.A.P	

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

1. UML Diagrams for the Project

The Unified Modeling Language (UML) provides a set of diagrams that help visualize and document various components and interactions in a software system. For our online cake ordering system:

- **Use Case Diagram:** Illustrates the interactions between the actors (customers, admin) and the system (e.g., placing orders, adding reviews, managing inventory).
- **Class Diagram:** Represents the structure of the system with entities like User, Cake, Order, Review, and their relationships.
- **Sequence Diagram:** Describes interactions in processes like placing an order, managing reviews, or payment.
- **Activity Diagram:** Outlines workflows such as user registration, order placement, and payment.

2. SDLC for the Online Cake Ordering System

The System Development Life Cycle (SDLC) ensures structured development:

1. **Communication:** Gather requirements from stakeholders like customers and admins.
2. **Requirement Collection:** Define features such as ordering cakes, leaving reviews, and inventory management.
3. **Feasibility Study:** Analyze technical, economic, and operational feasibility to implement the system.
4. **Requirement Analysis:** Document functional and non-functional requirements for the system.
5. **Software Designing:** Create UML diagrams and define database schema and architecture.
6. **Coding:** Develop the front-end (HTML/CSS/JavaScript) and back-end (PHP/SQL) functionalities.
7. **Testing:** Conduct unit tests, integration tests, and user acceptance tests for all features.
8. **Implementation:** Deploy the system for real-time use.
9. **Maintenance:** Regularly update the system and fix bugs.
10. **Disposition:** In future, retire the system when a better alternative is available.

3. SQL for the Online Cake Ordering System

SQL is integral for managing the database. Examples include:

- Retrieving Data
- Inserting Data
- Updating Data
- Deleting Data

SQL is a database programming language for retrieving and managing data in relational databases. Structured Query Language (SQL) is an acronym for Structured Query Language.

1. Introduction

1.1 Problem Statement

In today's world, where most businesses are moving online, cake shops often face challenges in reaching a wider audience and managing orders efficiently. Many customers prefer the convenience of online shopping but are unable to order cakes easily due to the lack of proper online platforms.

The absence of an online ordering system results in missed opportunities for businesses and inconvenience for customers. Issues like long wait times, lack of real-time updates on orders, and difficulty in managing inventory and payments also arise.

This project aims to create an **Online Cake Ordering System** that allows customers to browse cakes, add items to their cart, adjust quantities, and pay using online payment methods. The system will also calculate the total cost, including shipping fees, and allow customers to input shipping and payment details. On the admin side, the system will simplify order management, ensuring a smooth and efficient process.

1.2 Product Scope

The **Online Cake Ordering System** is designed to digitize the cake ordering process for both customers and business owners. The system's main purpose is to make ordering cakes simple, fast, and accessible from anywhere.

Objectives and Goals:

- Provide customers with an easy-to-use platform to select cakes, customize their orders, and make payments online.
- Enable the business owner (admin) to manage orders without requiring advanced technical skills.
- Ensure secure transactions and smooth communication between customers and the admin.
- Reduce manual workload and errors in the ordering process.

Benefits:

- Enhanced customer convenience with online ordering and payment options.
- Improved business efficiency through automated order and inventory management.
- Increased customer reach, contributing to higher sales and brand visibility.

Key Findings:

- Customers highly value the ability to view products, customize orders, and pay online.
- Admins need a simple interface to manage orders, as they may not have technical expertise.
- Security and accurate calculations, including shipping fees, are essential for trust and satisfaction.

1.3 Project Report Structure

1.4 1. Introduction

- Project Title: Online Cake Ordering System
- Objective:
The main objective of the online cake ordering system is to streamline the process of ordering cakes, improve customer experience, and enhance operational efficiency. This system allows customers to browse, select, and order cakes conveniently online, while also enabling the admin to manage orders, inventory, and customer interactions effectively.

2. Scope Statement

Inclusions:

- A user-friendly web interface for customers to browse cakes by categories, flavors, or ratings.
- Features for registered users to create accounts, manage profiles.
- A robust rating and reviews system to collect and display customer feedback.
- Inventory management for updating cake availability and stock levels.
- Payment gateway integration to support multiple payment methods securely.
- Admin dashboard to manage user accounts, reviews, and orders efficiently.
- Report generation for analyzing product performance and customer feedback.

Exclusions:
- The system will not handle third-party delivery logistics.
- The system will not support international currency transactions at this stage.

3. Deliverables

Functional Deliverables:
- Customer portal for placing orders.
- Admin portal for managing cakes, orders, and customer feedback.
- Database for storing user, order, and inventory details.

Non-Functional Deliverables:
- A responsive and mobile-friendly design.
- Secure login and payment systems.
- Scalable infrastructure for handling increased user traffic.

4. Timeline

- Requirement Analysis: 2 weeks
- System Design: 3 weeks
- Implementation and Integration: 6 weeks
- Testing and Debugging: 3 weeks
- Deployment: 1 week

5. Methodology

- Requirement Analysis:
  Conducted to gather and document functional and non-functional requirements from stakeholders.
- System Design:
  Includes creating wireframes, class diagrams, and state diagrams to represent system workflows and data interactions.
- Implementation:
  Divided into four key modules:
  *User Management System
  *Rating and Reviews System
  *Inventory Management System
  *Payment and Order Tracking System
- Testing:
  Comprehensive testing using unit tests, integration tests, and performance tests to ensure system reliability.

6. Evaluation

- Outcomes:
  - Efficient online cake ordering process with a seamless user experience.
- Lessons Learned:
  - Importance of user feedback during the design phase.
  - Challenges in integrating payment systems securely.

7. Conclusion

- Summary:

This system achieved its primary objectives of enabling customers to order cakes online conveniently while providing the admin with tools to manage inventory, orders, and feedback efficiently.

Advantages:

- Enhanced customer satisfaction. - Increased operational efficiency.

Disadvantages:

- Limited to local currency transactions. - Dependency on a stable internet connection.

Limitations:

- The system does not integrate third-party delivery services.
- Scalability for international orders is not implemented.

- Client Recommendations:

The client is encouraged to leverage the advantages of this system, such as streamlined order processing and detailed customer feedback, to improve business operations further.

This document was created by IEEE standards, and as a result, it can be recommended as meeting high standards.

- This project report is edited and written according to the provided project report template by using Microsoft Office Word.
- Table of content is included at the beginning of the document
- Line spacing for the text is set as 1.5.
- Font of the whole document is, Times New Roman.
- Font size of the text body is 12.
- Font size of the headings is 14.
- Headings are bolded.
- Headings are numbered.
- All the diagrammatic illustrations are included.
- All figures and tables are numbered.
- All the pages are numbered in the bottom, right corner.
- In the second chapter of our final report, we described the process we used to complete the project. The key components of methodology are requirement analysis, design, implementation, and testing.
- Our project's third chapter is evaluation. This section focuses on the project's outcomes, as well as the lessons learned throughout the process.
- We summarized the entire project in the final chapter. This section discusses our project's goals and objectives. We also evaluated the project's advantages and disadvantages, as well as its limitations. The client company was informed about the advantages of developing this project.

2. Methodology

2.1 Requirements and Analysis

To gather comprehensive requirements for this project, various techniques, including interviewing, use case diagrams, and activity diagrams, were employed. Each approach provided valuable insights into the system's functionality and user needs.

Interviewing

A structured approach to interviewing was adopted to uncover the client's requirements. Initial sessions included open-ended questions to explore the client's general expectations and vision for the system. Follow-up sessions utilized more specific and probing questions to clarify and refine the identified needs. These interviews were conducted both in person and over the phone. Although this method required a significant time investment, it proved highly effective in uncovering critical system requirements and addressing potential concerns.

Use Case Diagram

A use case diagram was developed to visually represent user interactions with the system. It illustrates the relationships between users and the various use cases they engage with, highlighting how the system supports their activities. This diagram serves as a high-level overview of user-system interactions, making it a crucial tool for understanding user roles and system functionality.

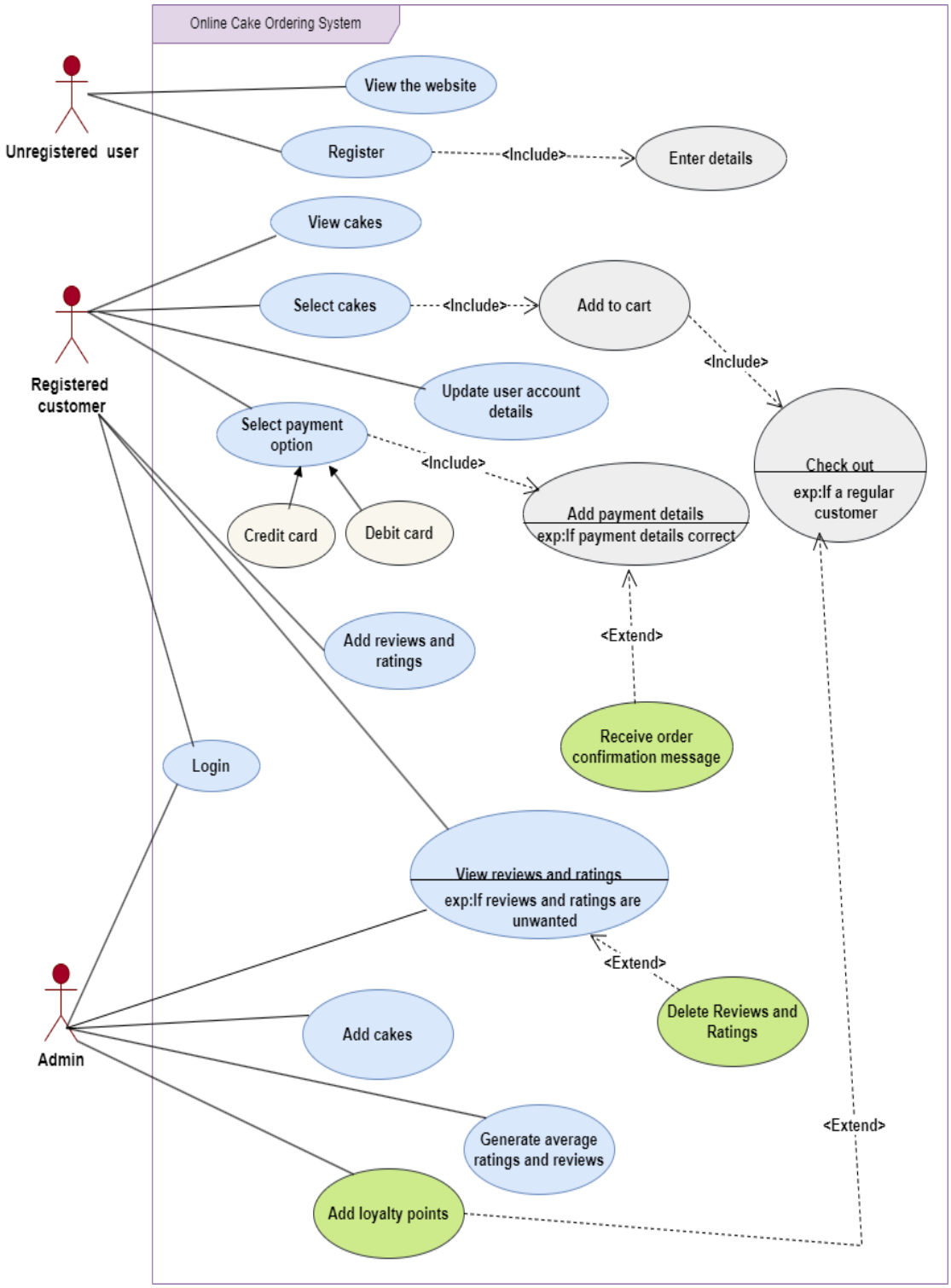


Figure 2.1.1: Use Case Diagram

Use Case Scenarios

The use case scenario describes the interactions between users and the system, capturing the system's behavior in various situations. It outlines how users achieve their goals while interacting with specific system functionalities. For this project, use case scenarios were developed based on information gathered during the requirements analysis phase. These scenarios focus on user goals, system responses, and the flow of actions, ensuring a clear understanding of user needs and system capabilities.

Use Case Scenario 01

Table 2.1.1: Use Case Scenario1

Use case Name	Checkout	
Pre- conditions	The customer should login to the system	
	The customer has already selected items and added them to the cart	
Post- condition	The order is successfully placed	
Primary actor	Registered customer	
Main success scenario	Step	Action
	1	The user clicks “proceed to checkout” after reviewing the cart
	2	The system directs to the payment page and displays the total cost and available payment methods
	3	The user selects a payment method and input the required payment details and click on “pay now”
	4	The system validates the payment information
	5	The system processes the payment through a payment gateway
	6	Upon successful payment, system confirms the transaction and order confirmation message

Extensions	Step	Branching Actions
	4a	If the payment information is invalid, system displays an error message asking the users to re-enter correct information
	4b	If the payment processing fails, system rejects the payment and informs to retry payment or suggests alternatives

Use Case Scenario 02

Table 2.1.2.1: Use Case Scenario user registration

Use case Name	User Registration	
Pre-conditions	Users should visit the web page.	
Post-conditions	Direct to Login page.	
Primary actors	System user (Customer/Admin)	
Main scenario	Step	Action
	1	Click on the signup icon.
	2	Display the login page.
	3	Click on the Register at the bottom of the login page.
	4	Fill in the details.
	5	Click Register.
Extensions	Step	Branching Actions

	4a	Display error message when user enter enter password without a special character.

Table 2.1.2.2: Login Use Case Scenario

Use case Name	Login	
Pre-conditions	User visit the website.	
Primary actors	Users (Admin/Customers)	
Main Scenario	**Step**	**Action**
	1	Click on the sign-up icon on the top of the page.
	2	System displays the login page.
	3	Users enter a valid username and password into the system.
	4	System validates the username and password.
Extensions	**Step**	**Branching Actions**
	3a. If the username or password is incorrect, the system shows an error message.	
	4a. If the validation fails, the user must log in again.	

Use Case Scenario 03

Table 2.1.3.1: Use Case Scenario 3

Use case Name	Inventory Management
Pre-conditions	The customer must be registered in the system with valid login credentials. They should have the necessary permissions to browse and request inventory items.
Primary actors	Customer, Administrator, System
Main scenario	Step Action
	1. Browse Inventory □ The customer starts by browsing the inventory through the system interface.
	If they need to filter by category, they can use the search functionality. The System processes the search request, applies filters, and displays the relevant inventory items
	2 Select Items The customer selects the required items from the inventory list.
	3. Fill a Request Form After selecting the items, the customer fills out a request form to formally request the inventory.
	4 Notify administrator Once the form is submitted, the System automatically notifies the Administrator about the new request.
	5 Review the requests The Administrator reviews the request to determine if it is suitable. - If the request is not suitable, the Administrator rejects the request, and the process ends. - If the request is suitable, the Administrator accepts it.
	6 Update the Inventory After acceptance, the System updates the inventory database to reflect the changes. The System then notifies the customer about the status of their request.
	7 Add items to the cart
	8 Issue Reporting If the Customer encounters any issues with the request or inventory, they have the option to submit feedback. The System then notifies the Administrator and stores the feedback as a report

Use Case Scenario 04

Table 2.1.4.1: Use Case Scenario 4

Use Case Name	Rating and reviews management for customers	
Use Case Id	PC004	
Summary	Customers can rate cakes, add reviews, update their feedback.	
Priority	5	
Pre-conditions	User must be logged in to the system..	
Post-conditions	- Rating and review are successfully added, updated. - Average rating for the cake is recalculated.	
Primary Actor	Registered customer	
Trigger	The customer wants to provide feedback on a cake.	
Main Success Scenario	**Step**	**Action**
	1	The user logs into the system.
	2	The user navigates to the cake details page
	3	The user selects the "Rate and Review" option.
	4	The user enters a star rating (1-5) and types their review.
	5	The system validates the input and saves the review.
	6	The system calculates and updates the average rating for the cake in real-time.
	7	The system displays a success message confirming the review submission.
Extension	**Step**	**Branching Action**
	1a	If the user login credentials are invalid, the system displays a message to re-enter login details.

	5a	If the user inputs invalid data (e.g., no rating), the system prompts them to correct the input.
	5b	If the user wants to update or delete their review, the system displays a popup to confirm the action.

Table 2.1.4.2: Use Case Scenario 4

Use Case Name	Rating and Review Management for Admin	
Use Case Id	PC005	
Summary	Admins can view, delete reviews, and reply to customer feedback.	
Priority	5	
Pre-conditions	Admin must be logged in to the system.	
Post-conditions	Ratings or reviews are successfully managed (deleted or replied to).	
Primary Actor	Admin	
Trigger	The admin wants to manage customer feedback.	
Main Success Scenario	**Step**	**Action**
	1	The admin logs into the admin dashboard.
	2	The admin navigates to the "Manage Reviews" section.
	3	The admin views a list of customer reviews and ratings.
	4	The admin selects a specific review to delete or reply to.
	5	The system validates the action and executes it.
	6	For deletions, the system removes the review and recalculates the average rating.

	7	For replies, the system links the admin's reply to the respective review and displays it under the review.
	8	The system displays a confirmation message for the action.
Extension	**Step**	**Branching Action**
	1a	If the admin login credentials are invalid, the system displays a message to re-enter login details.
	5a	If the admin deletes a review, the system recalculates the average rating for the respective cake.
	5b	If the admin's reply fails to save, the system displays an error message and logs the failure.

Activity Diagrams

Activity diagrams were utilized to model the dynamic aspects of the system, representing workflows and processes associated with user interactions. These diagrams provide a detailed visualization of the sequence of activities, decisions, and outcomes within the system. By combining use case scenarios and activity diagrams, the project team ensured a thorough analysis of user requirements, enabling the development of a system tailored to meet client expectations effectively.

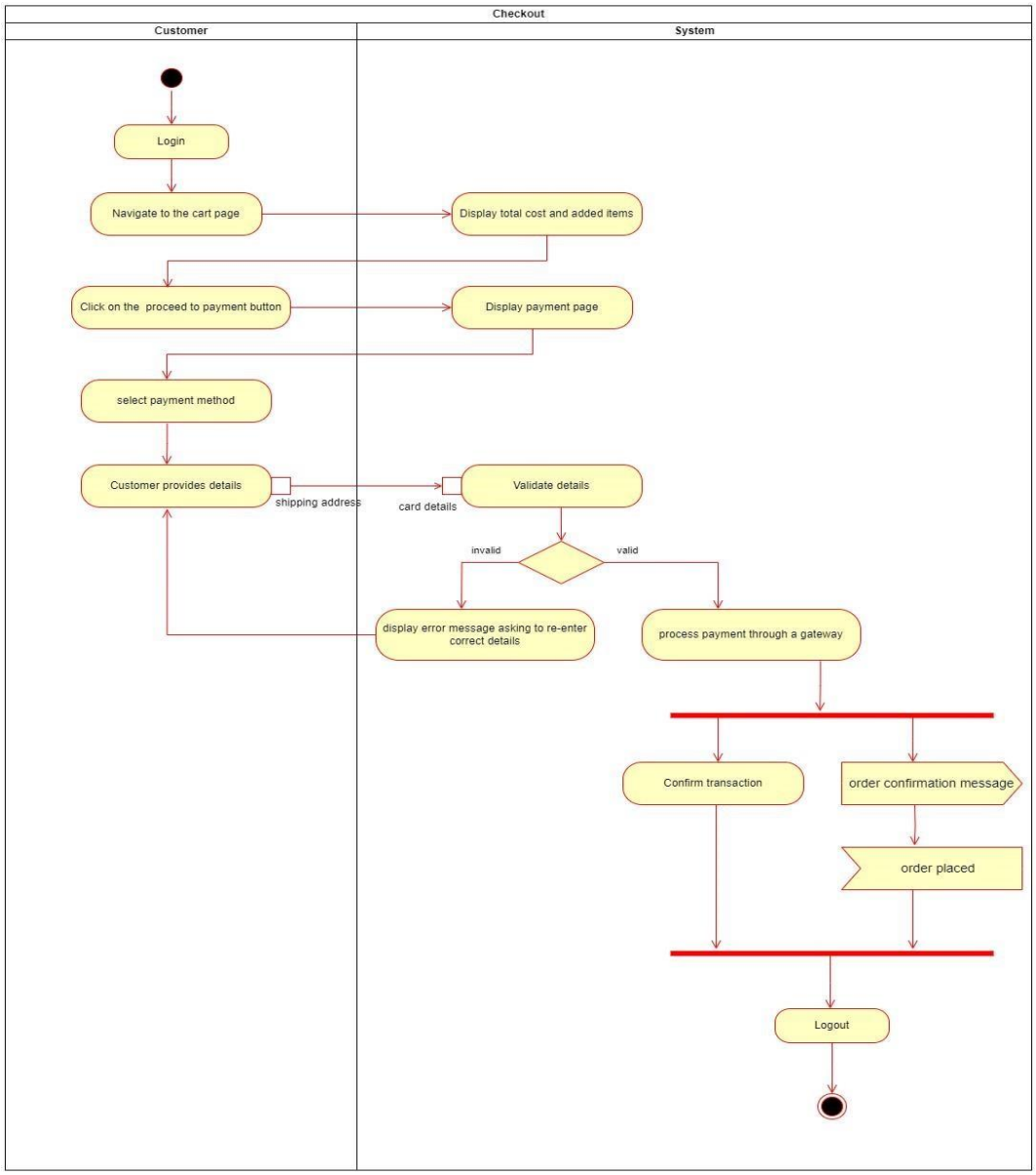


Figure 2.1.6: Activity Diagram 1

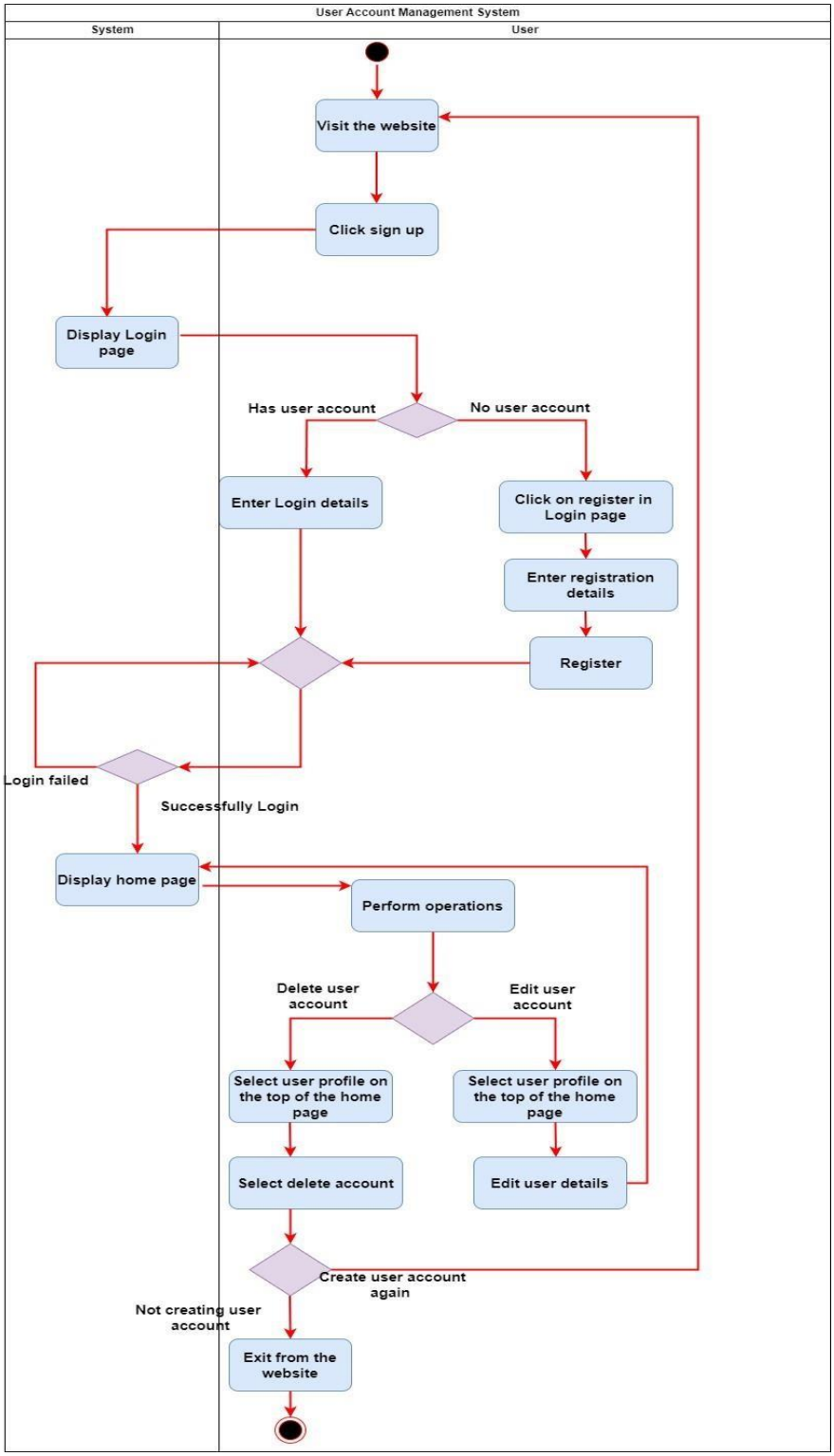


Figure 2.1.7: Activity Diagram 2

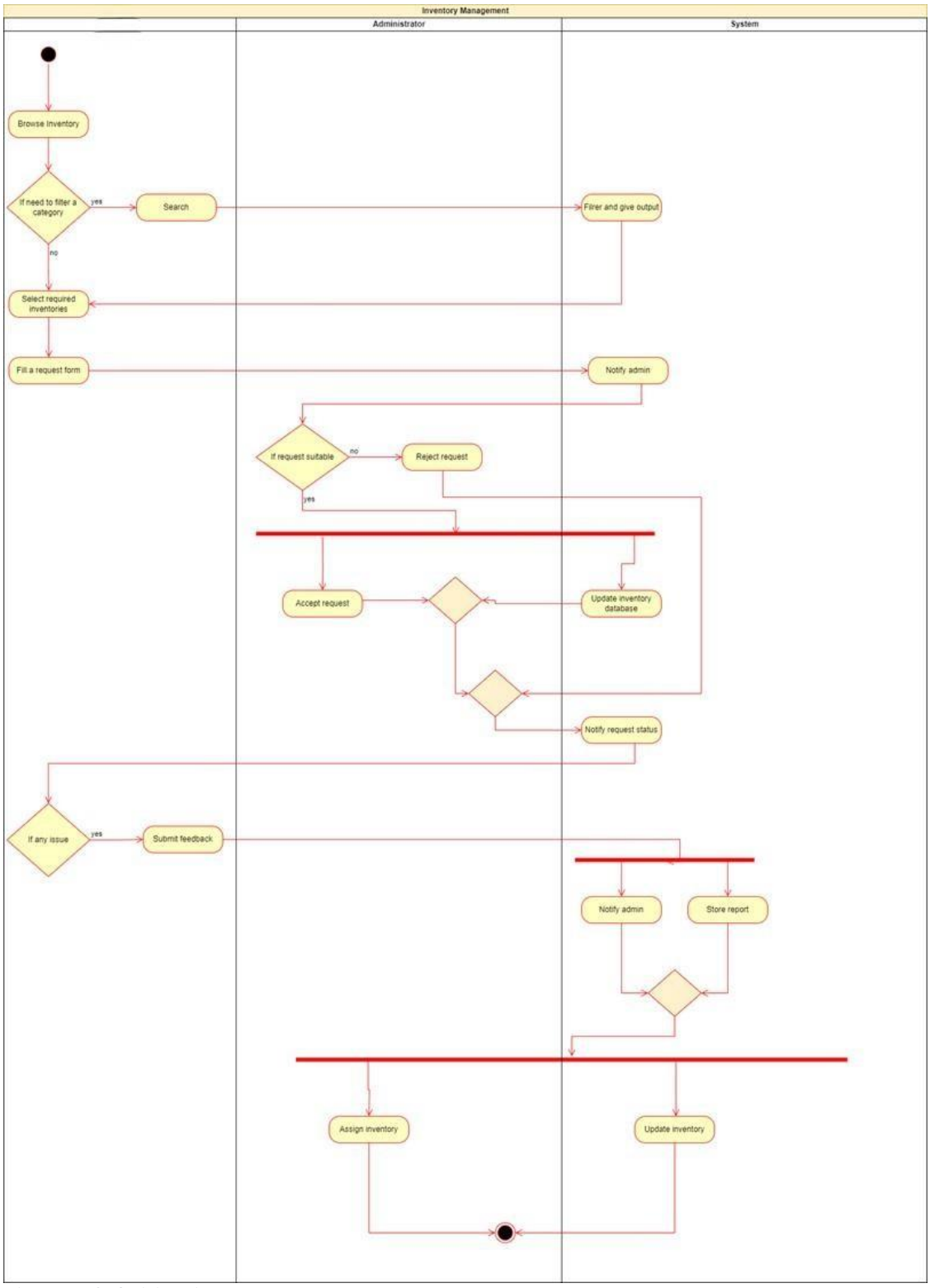


Figure 2.1.8: Activity Diagram 3

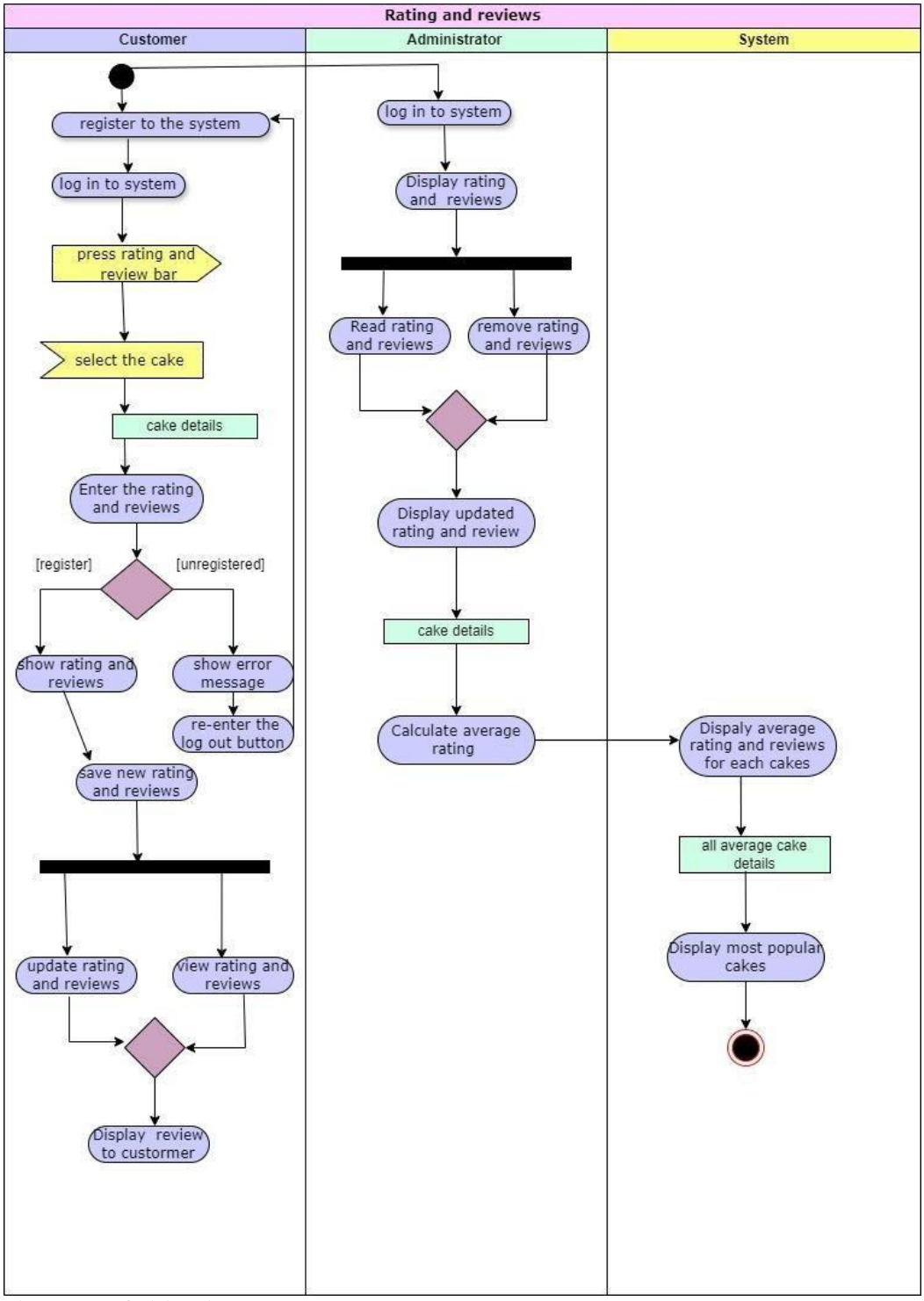


Figure 2.1.9: Activity Diagram 4

2.2 Design

- **Logical Design**

Class Diagram

Class diagrams were employed to model the static structure of the system by defining its classes, attributes, methods, and the relationships between them. This diagram provides a blueprint for system implementation, highlighting key entities and their interactions. It captures essential details such as inheritance, composition, and associations, ensuring a well-structured and maintainable design.

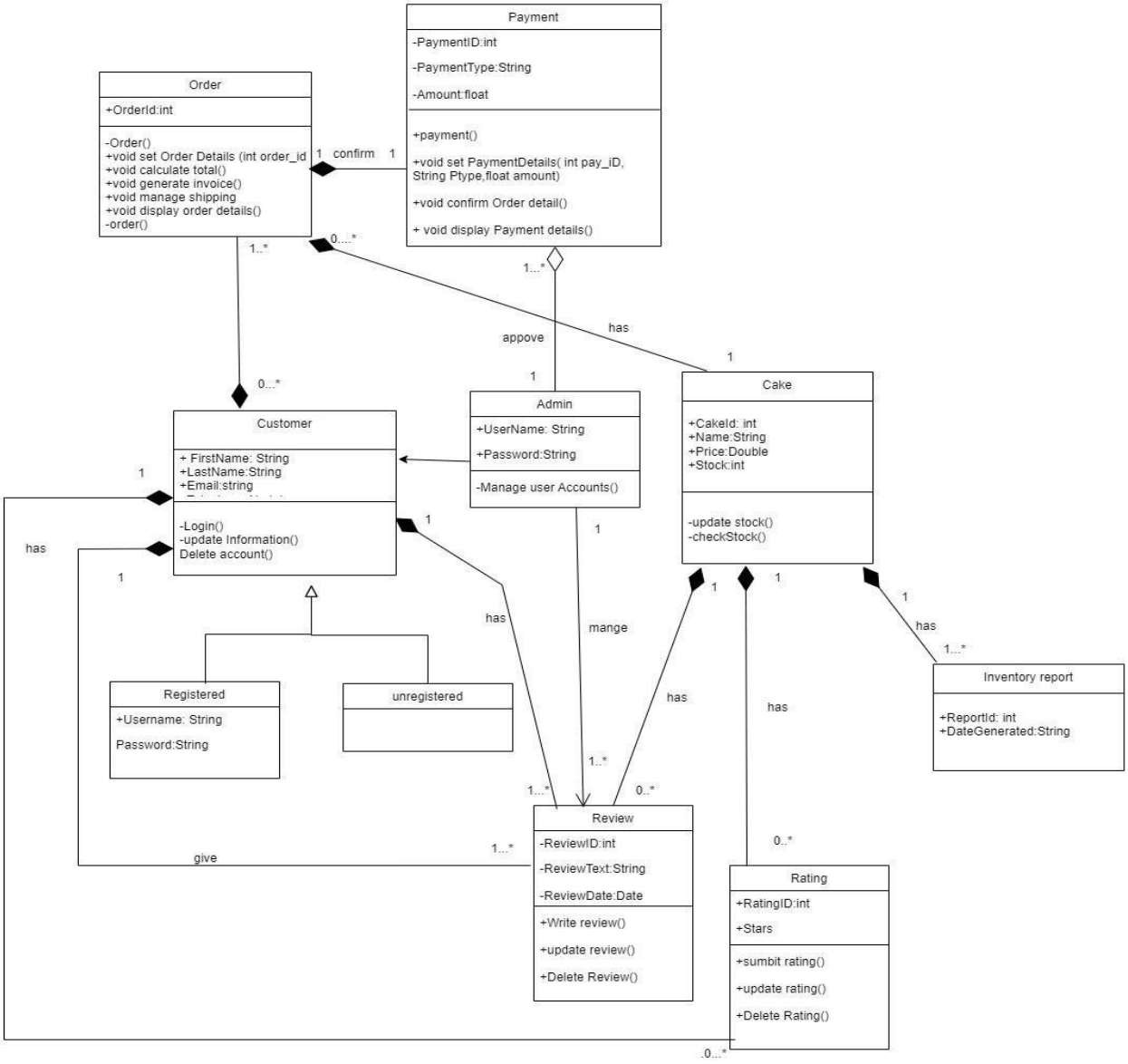


Figure 2.2.1: Class Diagram

Database Design

ER diagrams

Data flow, data stores, processors, and entities are the system's main components. ER diagrams were utilized to model the system's data structure by identifying entities, their attributes, and the relationships between them. This diagram provides a detailed representation of the database schema, ensuring that all necessary data relationships are captured and organized efficiently. By creating ER diagrams, the team gained a clear understanding of the data requirements and constraints, enabling them to design a database that supports the system's functionalities while maintaining data integrity and consistency.

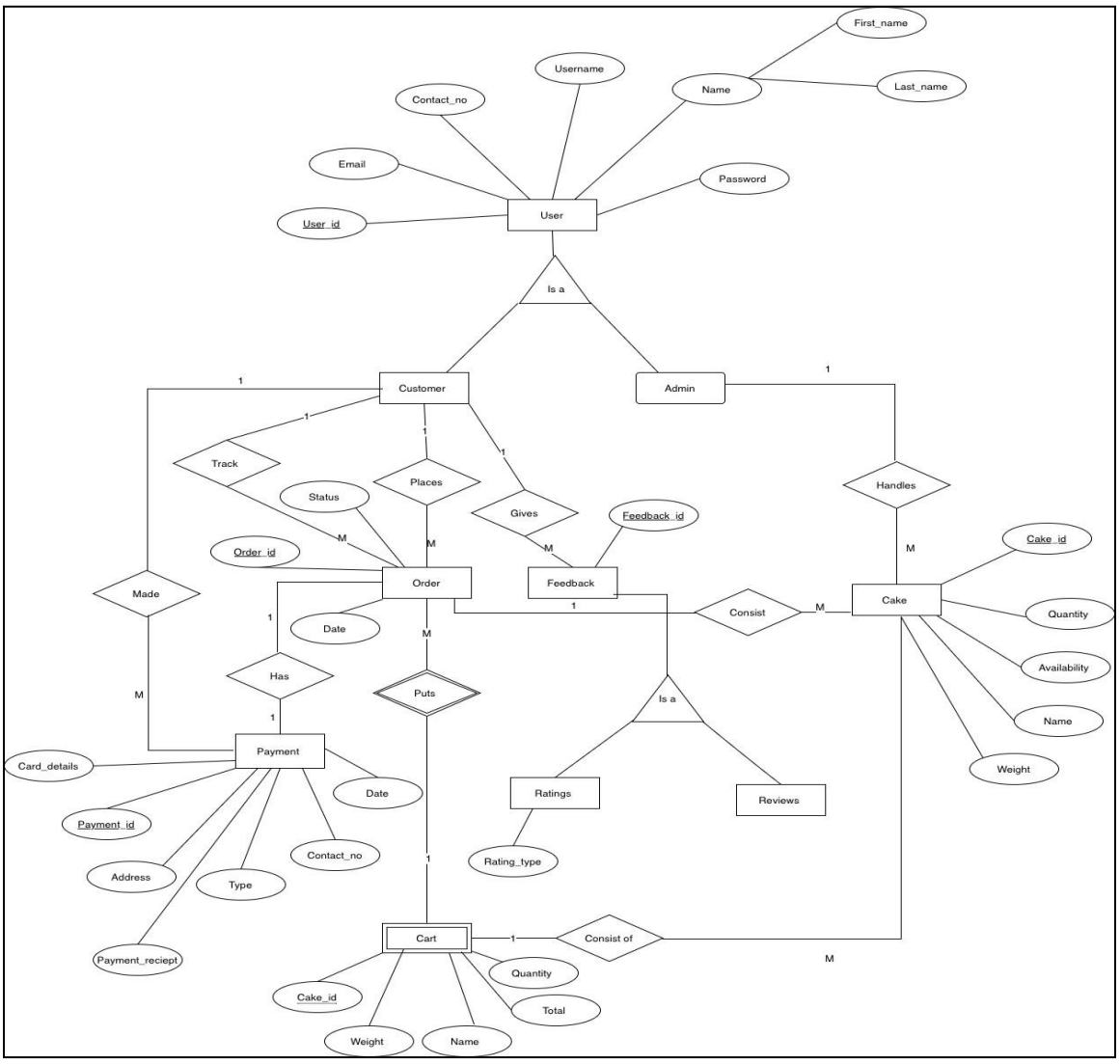


Figure 2.2.2: ERdiagram

Object diagram

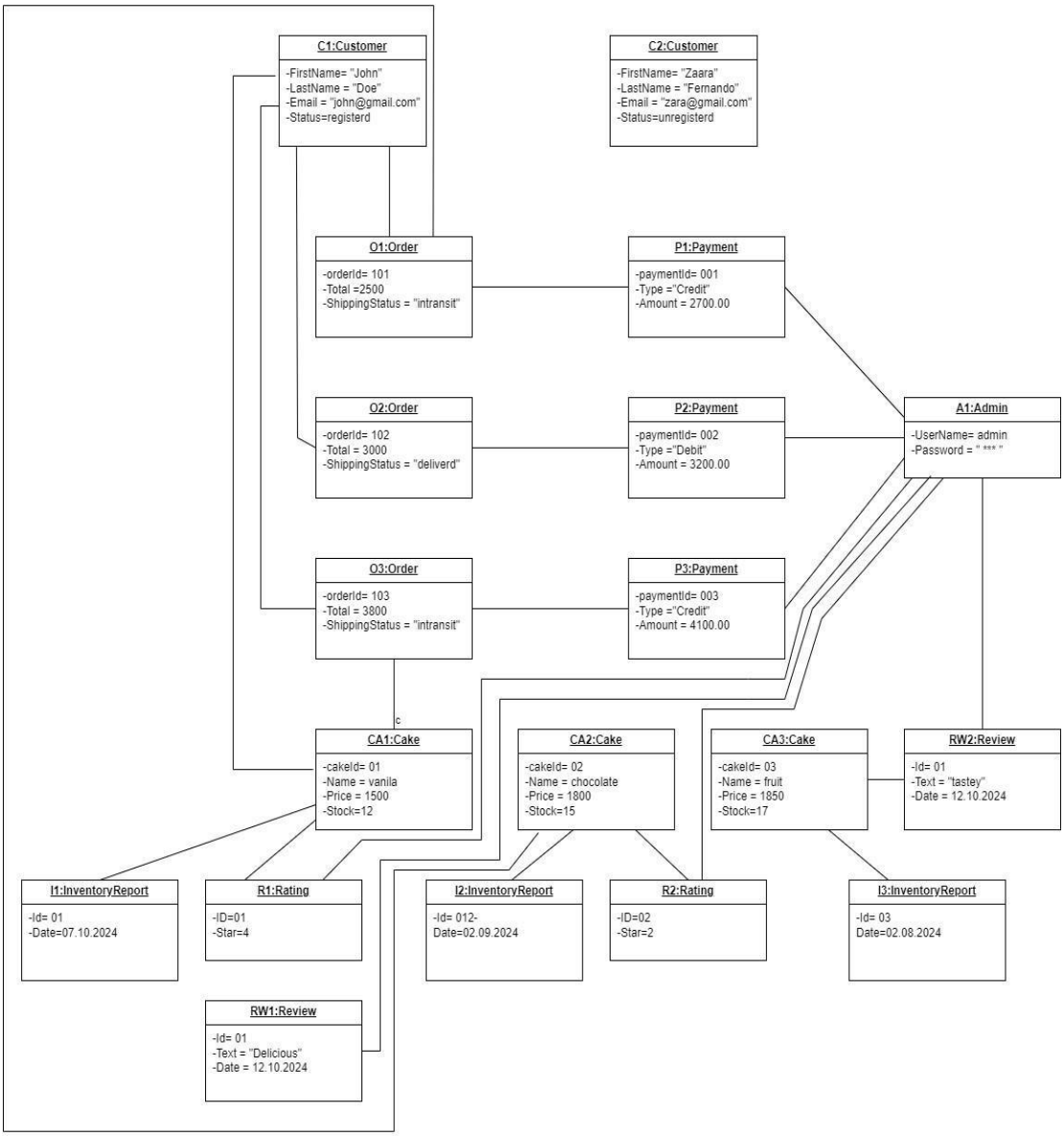


Figure 2.2.3: Object Diagram

State machine diagram

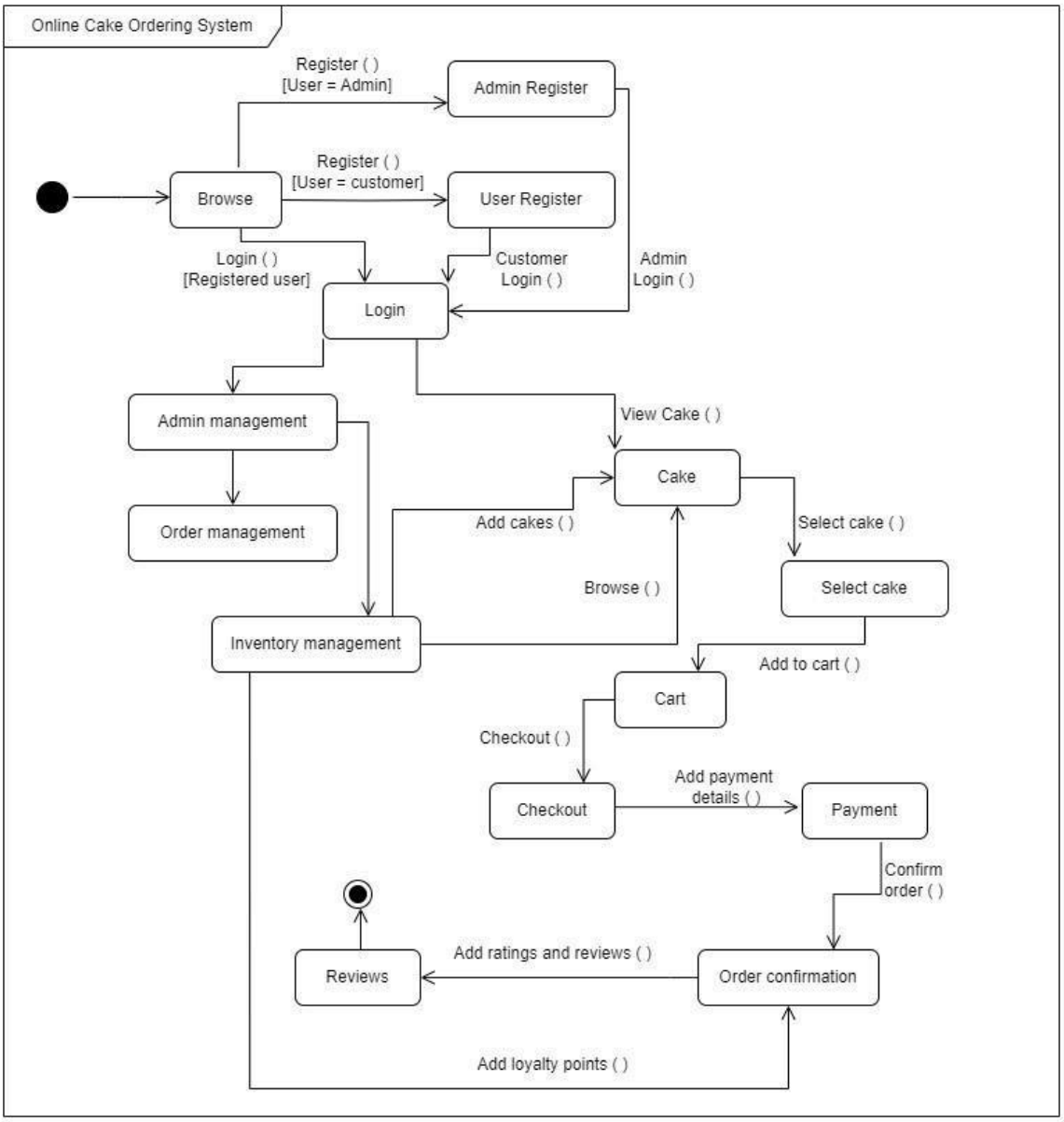


Figure 2.2.4: State machine Diagram

Sequence Diagram

Sequence Diagram 1- *Payment and cart management*

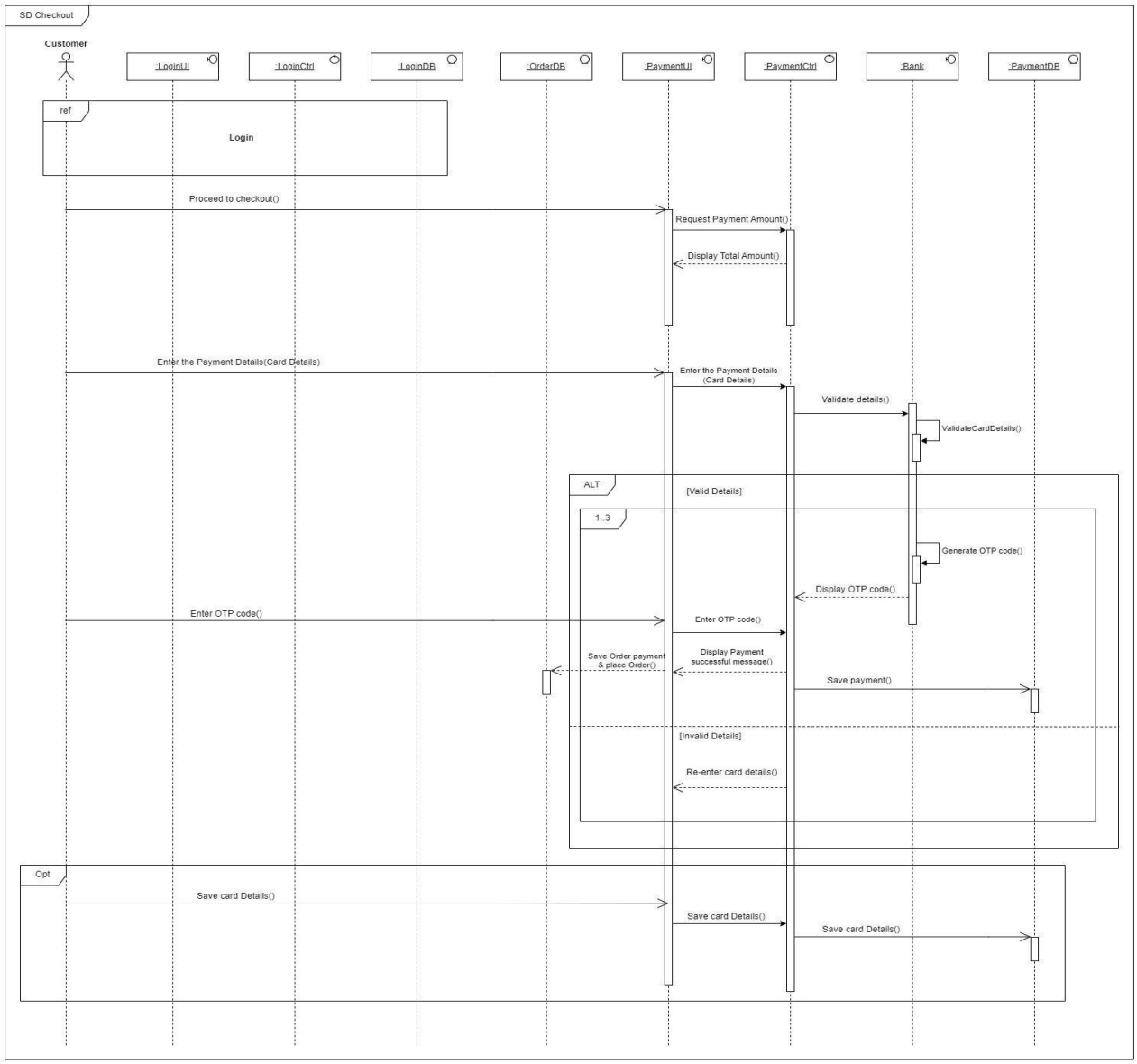


Figure 2.2.5: Sequence Diagram 1

Sequence Diagram 2- User Management System

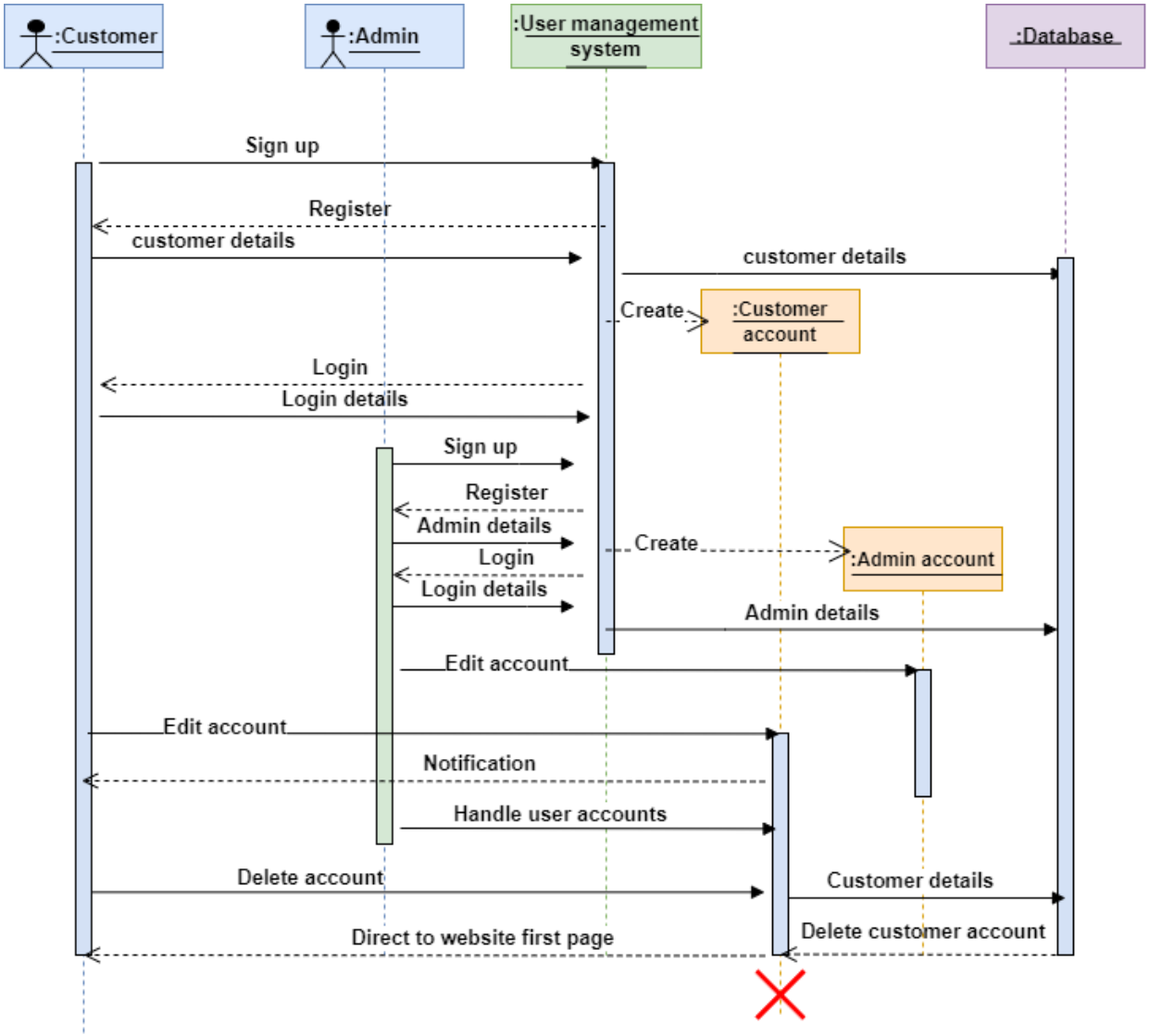


Figure 2.2.6: Sequence Diagram 2

Sequence Diagram 3- Inventory Management System

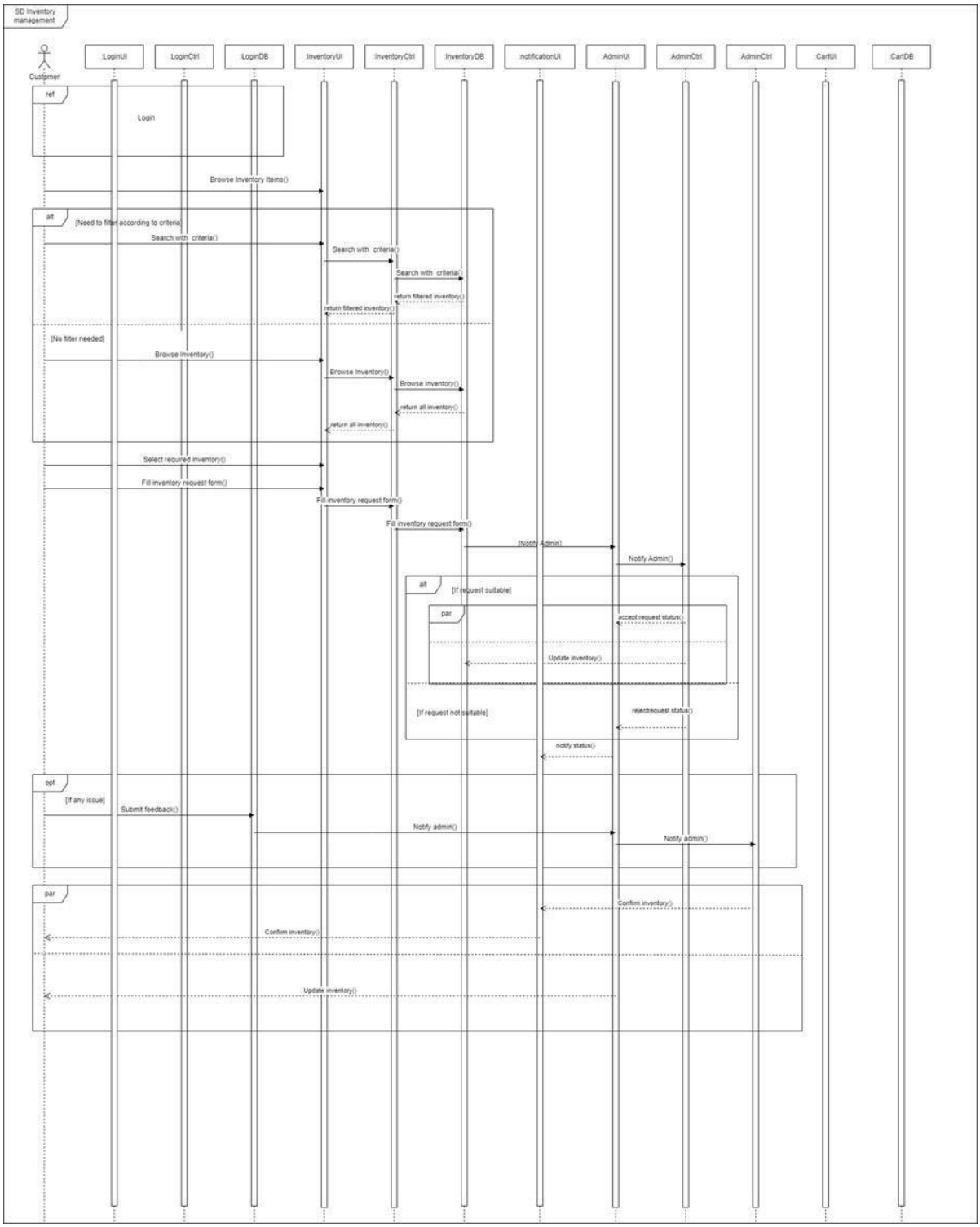


Figure 2.2.7: Sequence Diagram 3

Sequence Diagram 4- Reviews and Ratings management

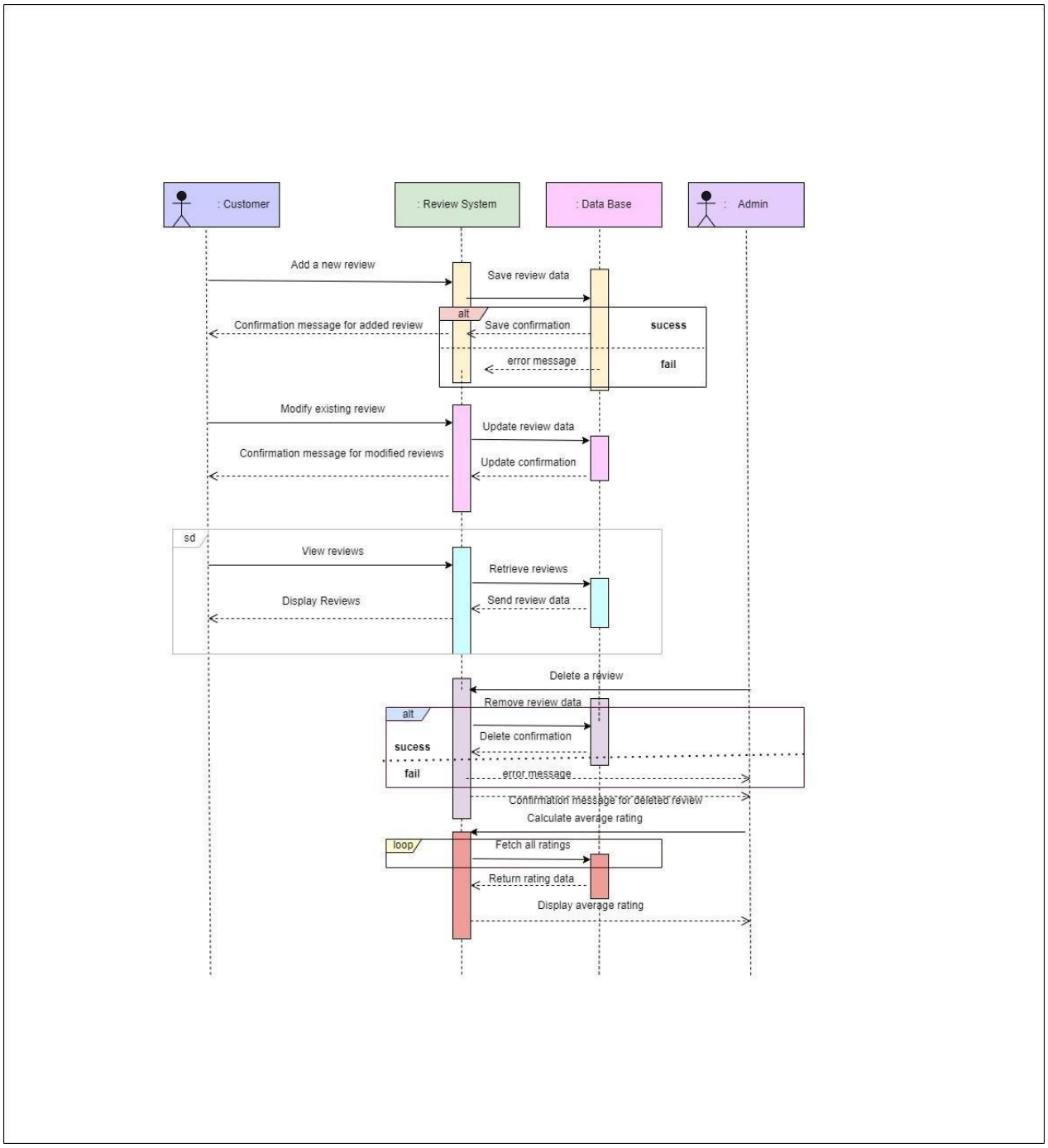


Figure 2.2.8: Sequence Diagram 4

2.3 Implementation

For the implementation of the online cake ordering system, developers must possess a thorough understanding of the chosen programming language and technologies. The technologies used in the system are designed to be fully compatible with the client's existing hardware infrastructure. To ensure optimal performance, the organization must have sufficient RAM available. Furthermore, the selected technologies are tailored to align with the company's operational environment, ensuring seamless integration and functionality.

Programming Language: C

The system's core functionalities are developed using the C programming language, known for its speed, reliability, and ability to work closely with system-level components. C provides robust support for modular programming, enabling developers to create efficient and scalable code for the cake ordering system.

Programming Tool: Visual Studio

Visual Studio is the Integrated Development Environment (IDE) used for coding and debugging the system. It offers a user-friendly interface, powerful debugging tools, and features that enhance productivity. With its support for version control and extensive extensions, Visual Studio simplifies the development process and improves collaboration among team members.

Database Design Tool: phpMyAdmin

phpMyAdmin is utilized to design and manage the system's database. It provides an intuitive web-based interface for creating, modifying, and querying databases. phpMyAdmin simplifies database management tasks such as defining relationships, indexing, and optimizing queries, ensuring a well-structured and efficient database for the system.

Database Server: XAMPP

XAMPP serves as the database server, providing a complete package with Apache, MySQL, PHP, and Perl. It is lightweight and easy to set up, making it ideal for testing and deploying the online cake ordering system. With XAMPP, the system benefits from reliable database hosting, seamless integration with phpMyAdmin, and local development capabilities.

This combination of tools and technologies ensures the online cake ordering system is robust, efficient, and well-suited to meet the client's requirements.

2.4 Testing

1. Inventory Management

Test Case ID	Feature Description	Prerequisite	Test Description (Steps)	Input Data	Expected Result	Pass/ Fail
001	Fulfillment of Fruit Cake order	Sufficient inventory available for ingredients	1. Check inventory levels for Fruit Cake ingredients . 2. Place an order for Fruit Cake. 3. Verify that the order is fulfilled, and inventory is updated.	Fruit Cake: Flour (300g), Sugar (200g), Fruits (400g).	The order is processed successfully, and inventory decreases for Flour (-300g), Sugar (-200g), Fruits (-400g).	pass
002	Prevent order for out-of-stock cake	Insufficient inventory for a required ingredient	1. Set stock for Chocolate to 0. 2. Attempt to place an order for Chocolate Cake. 3. Verify that the order is blocked and an error message is displayed.	Chocolate Cake: Chocolate (300g), Flour (400g), Eggs (3). Chocolate = 0g.	System blocks the order and displays: "Chocolate Cake cannot be fulfilled due to insufficient stock of Chocolate."	pass
003	Fulfillment of multiple cake orders	Inventory contains sufficient ingredients for multiple cakes	1. Place simultaneous orders for multiple cakes. 2. Verify that the system correctly adjusts inventory and fulfills the orders.	Black Forest: Flour (500g), Sugar (300g), Cream (200g). Strawberry Cake: Strawberries (400g),	Both orders are processed successfully, and inventory is updated correctly for all ingredients.	pass

				Sugar (250g).		
004	Cake fulfillment with customizations	Customization feature enabled in the system	1. Place an order for Vanilla Cake with added frosting. 2. Verify the system deducts additional frosting from inventory and processes the order.	Vanilla Cake: Flour (300g), Sugar (200g), Eggs (2). Customization: Frosting (200g).	Order is fulfilled with updated inventory: Flour (-300g), Sugar (-200g), Eggs (-2), Frosting (-200g).	pass
005	Bulk fulfillment for the same cake	Bulk ordering feature enabled	1. Place a bulk order for 10 Fruit Cakes. 2. Verify the system deducts ingredients for all 10 cakes and fulfills the order.	Fruit Cake (10 orders): Flour (3kg), Sugar (2kg), Fruits (4kg).	All 10 Fruit Cakes are processed, and inventory decreases by the required amounts.	pass
006	Generate fulfillment report	Reporting feature enabled	1. Place orders for multiple cakes. 2. Generate a fulfillment report. 3. Verify the report lists all fulfilled orders,	Orders: Vanilla Cake (3), Chocolate Cake (2), Black Forest Cake (1).	The fulfillment report accurately lists all completed orders with details.	pass

Table 2.4.1: Test Case

2. Checkout

Test Case ID	Feature Description	Prerequisite	Test Description (Steps)	Input Data	Expected Result	Pass/Fail
1	Add cakes to the cart	Valid URL to access the system	1. Access the system. 2. Browse cakes. 3. Click on the "Add to Cart" button for selected cakes. 4. View the cart page to confirm added items.	Cake Quantity	Cakes are successfully added to the cart and displayed on the cart page.	pass
2	Proceed to checkout	Items are present in the cart	1. Add items to the cart. 2. Click on the "Checkout" button. 3. Redirect to the payment page.	Cart items Quantities	The system redirects to the payment page successfully.	pass
3	Validate payment details	User is on the payment page	1. Enter valid payment details (card number, expiry date, CVV). 2. Click the "Pay Now" button.	Card Number: 4111111111111111 Expiry Date: 12/25 CVV: 123	Payment is processed successfully, and a confirmation message is displayed.	pass
4	Display order	Payment is successfully processed	1. Complete the	Order details Delivery details	The system displays an order	pass

	confirmation message		payment process. 2. Verify if the system shows an order confirmation message with the order details.		confirmation message	

Table 2.4.2: Test Case

3 User account management

Test case ID	**Feature description**	**Prerequisite**	**test Description (test steps)**	**Input data**	**Expected result**	**(pass/ fail)**
1	Check whether the login can be done without using the correct username and password.	Valid URL to access the system.	1. Access to the system 2. View home page 3. Click on the signup icon. 4. Register page display. 4. Click the "Admin/user " login button at the bottom.	User login Username: Wathsani Password:1234	Login cannot be done without the correct username and password.	pass
2	Check whether the update and delete buttons are working properly or not.	Valid URL to access the system.	1. Access to the system. 2. Login as a user/admin. 3. Click on the signup icon in the top right corner. 4. Update user /admin details.	Email: abcw@gmail.com First name: Mariyan Last name: Diluni Telephone number:0114257123	Users can update user details and delete user accounts.	pass

			5. Delete user/admin details OR user/admin account.			
3	Check whether the buttons of the navigation bar are navigated or not.	Valid URL to access the system.	1. Access to the system. 2. Register. 3. Login to the system. 4. View the relevant home page (user/admin home page). 5. Click on the buttons in the navigation bar one by one.	Option: Product	A new page should display the cake items according to the selected option.	pass
4	When registering, check whether the user and admin are registered according to their user type.	Valid URL to access the system.	1. Access to the system. 2. click on the signup button on the top right corner of the home page. 3. Register the user/admin with the relevant details by selecting the user type.	User type: admin First name: Sandun Last name: Wathsana Email: sw@gmail.com Contact number:0702347893	The user will be directed to the admin/user(customer) home page.	pass

Table 2.4.3: Test Case

4. Rating and Review Management

Test case ID	Feature description	prerequisites	Test Description (test steps)	Input data	Expected result	Actual result
1	Check whether rating and reviews are add,update,delete successfully.	-Valid URL to access the system. -User is logged in and on a product page.	1. Navigate to a cake product page. 2. Check for the presence of "Add," "Delete," and "Update" buttons. 3. Click each button to ensure it functions properly.	Rating: click 4 stars Reviews: Good product.	Buttons should appear correctly and execute their respective functionalities when clicked.	pass
2	Check whether searching bar is working properly or not.	-Valid URL to access the system. -User is - Admin logged in and reviews are present.	1. Log in as an admin. 2. Enter a keyword into the search bar to searcj reviews. 3. Submit the search query. 4. Verify that only relevant reviews are displayed.	Keywords (e.g., "chocolate," "vanilla").	The system should return all reviews matching the input keyword.	pass
3	Verify can not submit rating without reviews(validation)	-Valid URL to access the system. -User is logged in and on a product page.	1. Enter a star rating without adding text in the review input field. 2. Click the "Submit" button. 3. Observe the system response.	Star rating only (e.g., 4 stars) without a review comment.	The system should display a validation error: "Review cannot be empty."	pass
4	Check whether average rating sort	-Reviews and ratings are	1. Navigate to the	Sorting options	Reviews should appear	pass

	by correctly low to high and high to low.	available for a product.	product reviews section. 2. Select "Sort by Low to High" and observe the order. 3. Select "Sort by High to Low" and observe the order.	("Low to High" and "High to Low")	in the correct order based on the selected sort option.	
5	Check whether add reply correctly and delete reply correctly.	Admin logged in and reviews are present.	1. Log in as an admin. 2. Navigate to a customer review. 3. Add a reply and ensure it appears below the review. 4. Delete the reply and ensure it is removed.	Reply text (e.g., "Thank you for your feedback!").	The reply should be successfully added and deleted, with appropriate confirmation messages.	pass
6	Check whether average rating and reviews calculate correctly when add rating and update rating.	Reviews with varying ratings are available.	1. Add ratings for a product (e.g., 3, 4, 5 stars). 2. Update an existing rating. 3. Verify the recalculated average rating displayed on the product page.	Ratings data (e.g., 3, 4, 5 stars).	The displayed average rating should match the calculated value based on the ratings provided or updated.	pass

Table 2.4.4: Test Case

Evidence for Test Cases

1. User Management

1. User Registration as an admin/user(customer)

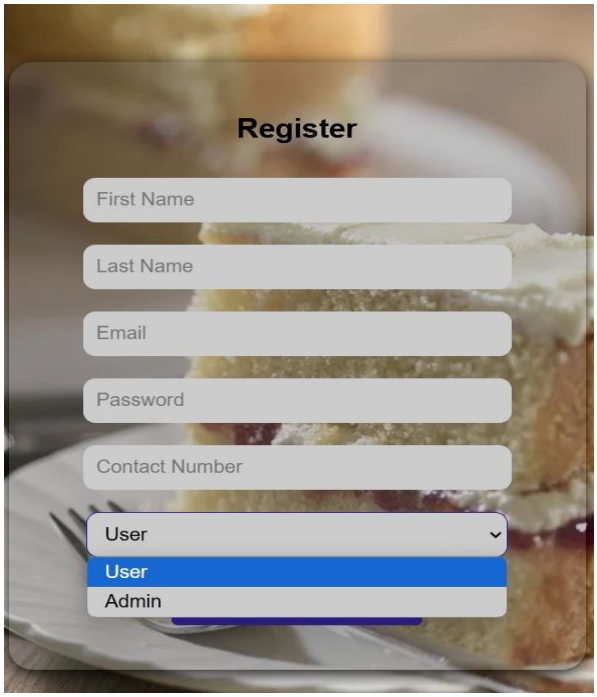


Figure 2.4.1:user registration

2. Notification when successfully registered to the system.

Figure 2.4.2:Notification for successfully registered

3. Login to the system.

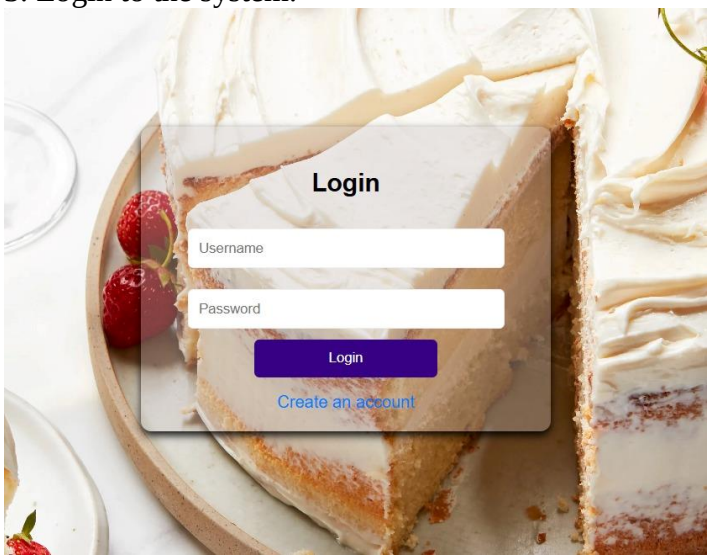


Figure 2.4.3:Login

4.Login failed

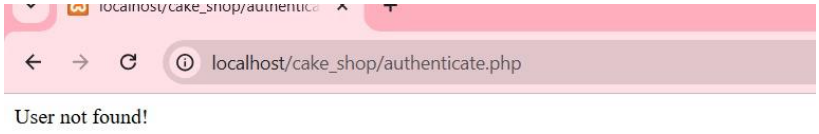


Figure 2.4.4:Login failed

5. Update and Delete user(customer) account

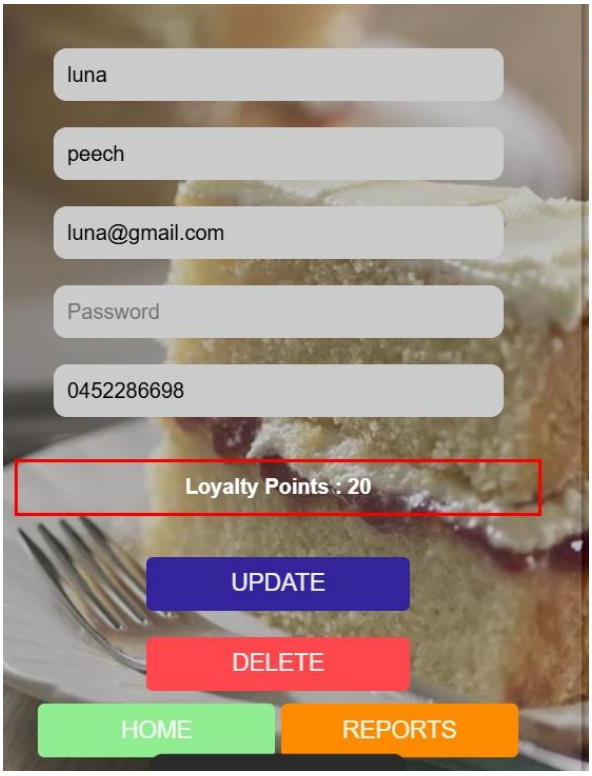


Figure 2.4.5: update and delete user(customer) account

6. Successfully updated notification

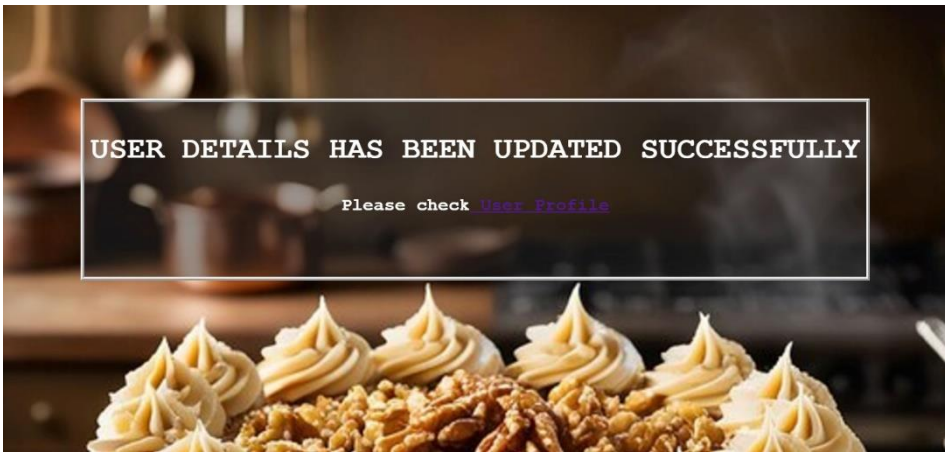


Figure 2.4.6:Updated notification

2. Checkout

01.Add cakes to the cart

Figure 1: TC1.1-. Add cakes to the cart

Figure 2: TC1.2-. Add cakes to the cart

02.Proceed to checkout

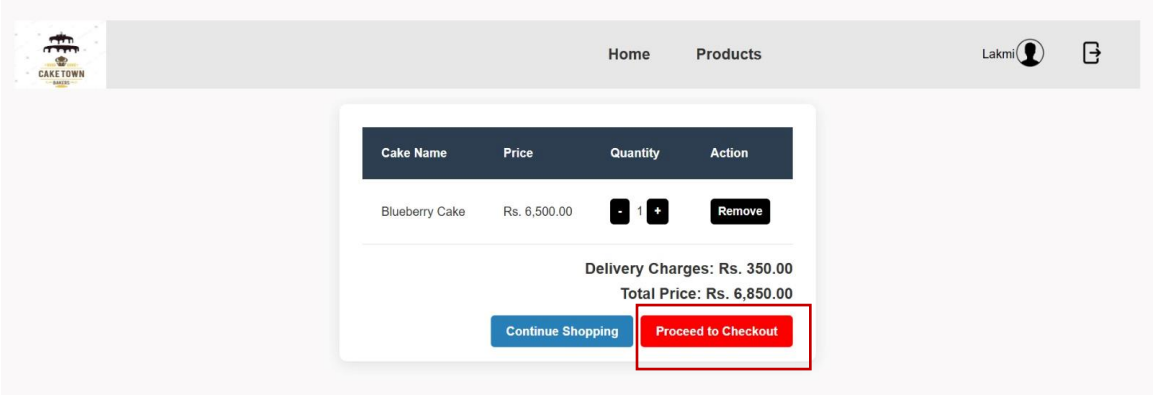


Figure 3: TC2.1-. Proceed to checkout

03.Validate payment details

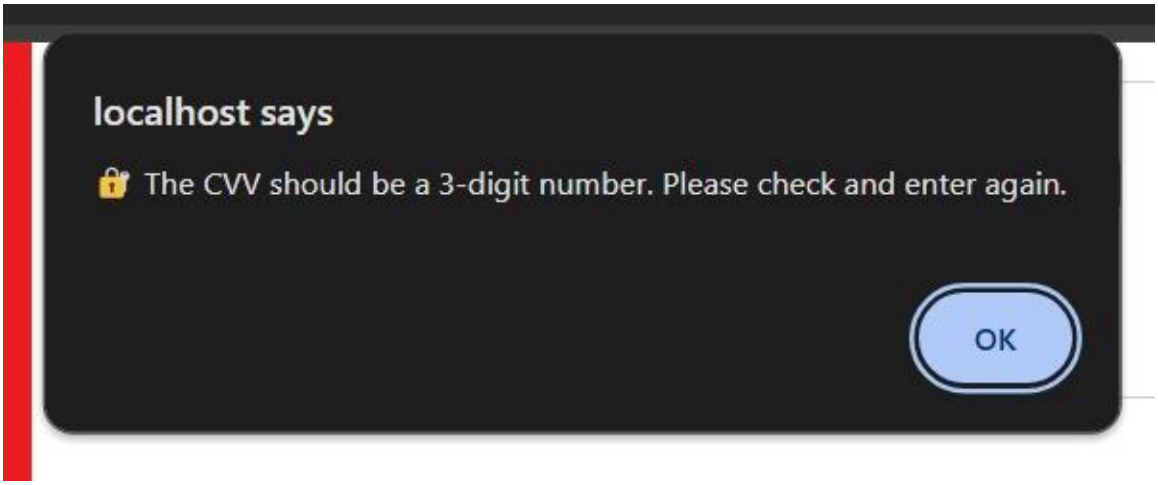


Figure 4: TC3.1-. Validate cvv

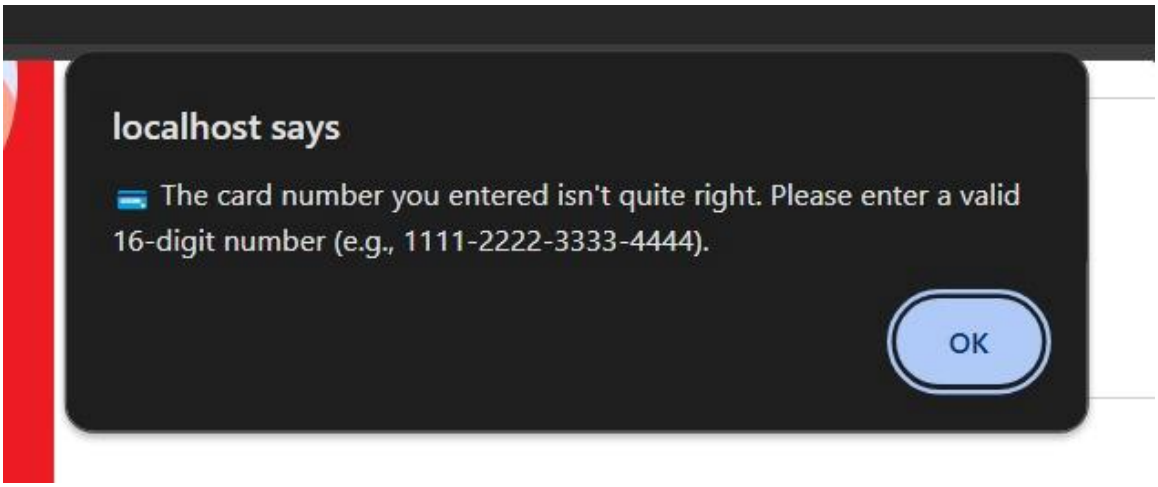


Figure 5: TC3.2-. Validate card number

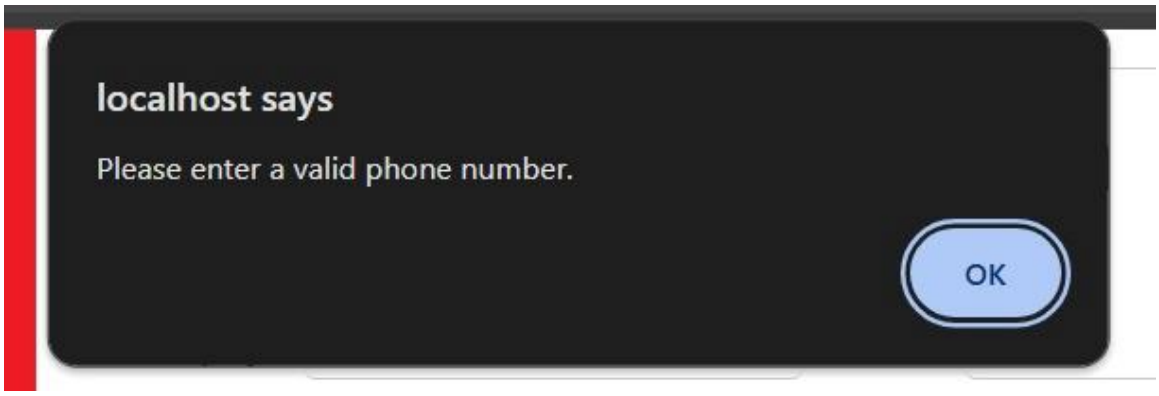


Figure 6: TC3.3-. Validate card number

04.Display order confirmation message

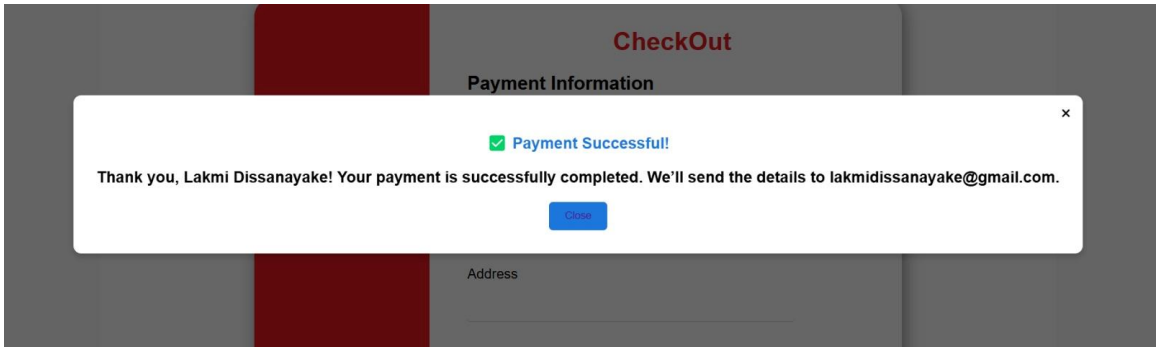
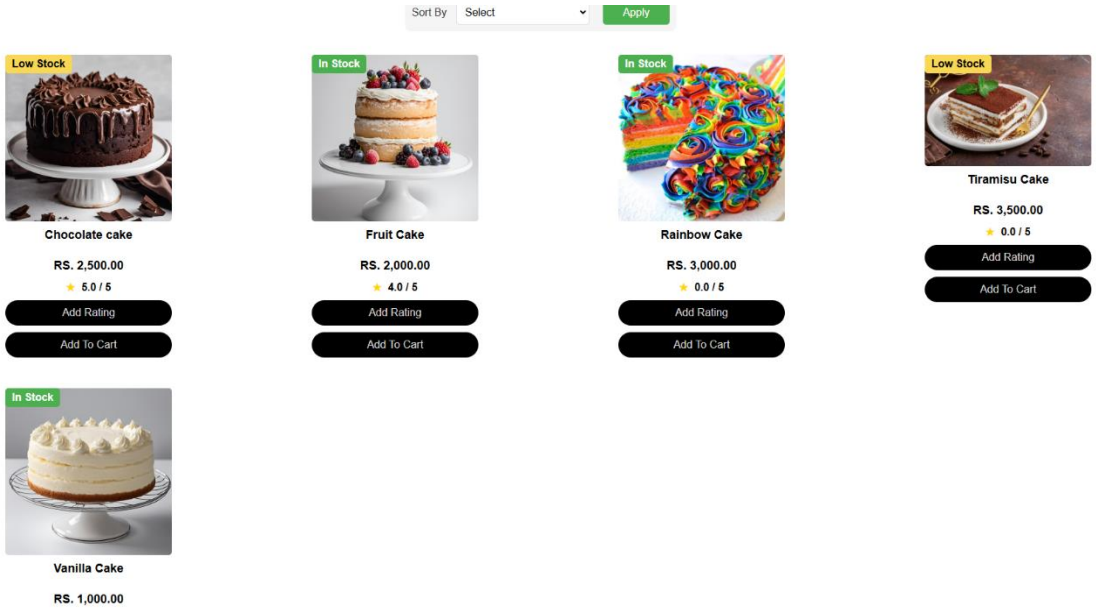


Figure 4: TC1.4-.Display order confirmation message

3. Inventory Management

1. User will be shown with availability status



2. Reorder status report is generated

CakeTown Bakers

Contact Number: 0112789654

Address: 2nd Lane, Malabe

Cake Inventory Report

Cake Name	Price (Rs)	Stock	Stock Status
Fruit Cake	2,000.00	10	In Stock
Chocolate cake	2,500.00	1	Low Stock
Vanilla Cake	1,000.00	15	In Stock
Tiramisu Cake	3,500.00	4	Low Stock
Rainbow Cake	3,000.00	10	In Stock

4. Rating and review management

1. . Check whether rating and reviews are add,update,delete successfully.

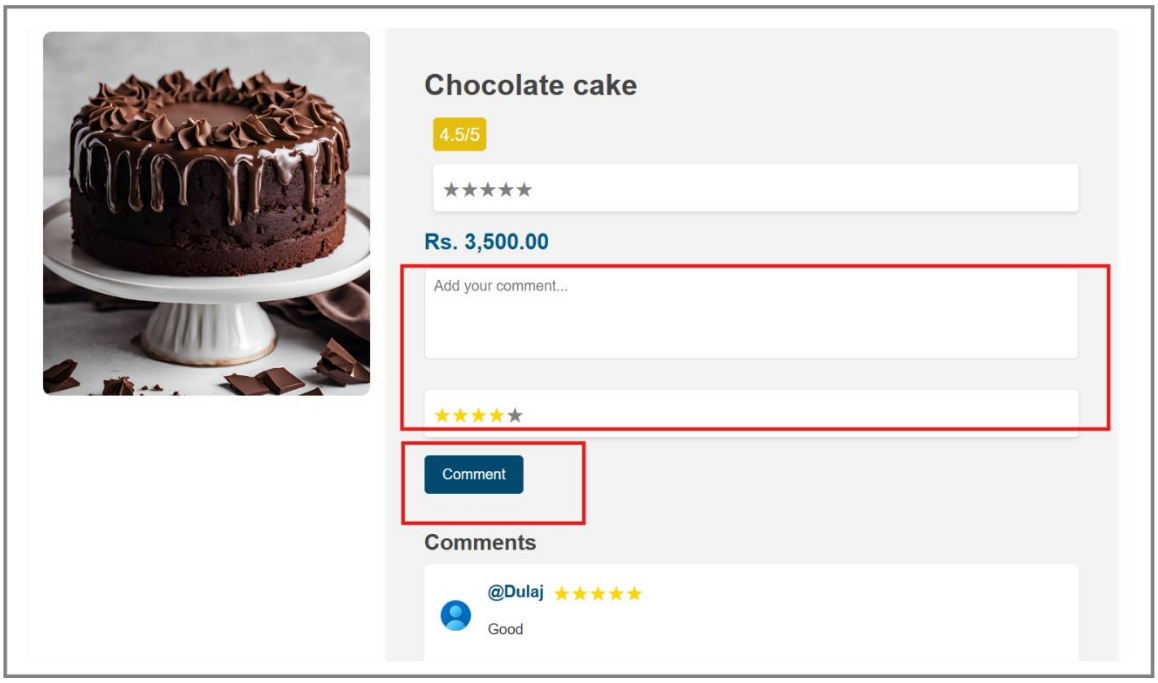


Figure 1: TC1.1-. Check whether rating and reviews are add correctly

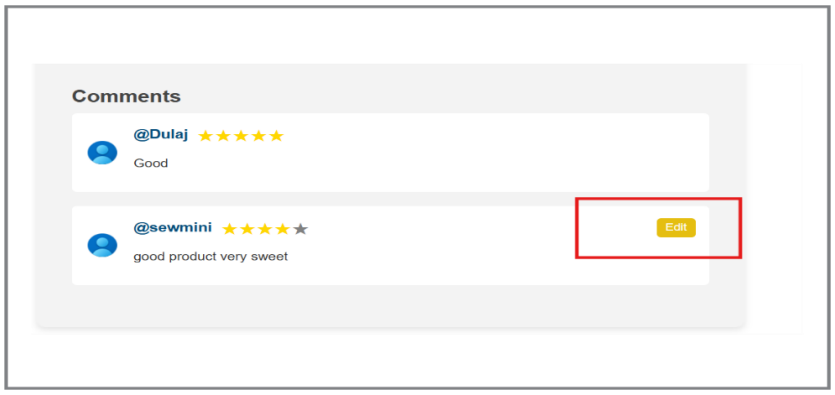


Figure 2: TC1.2-.. Check whether rating and reviews are update correctly Evaluation

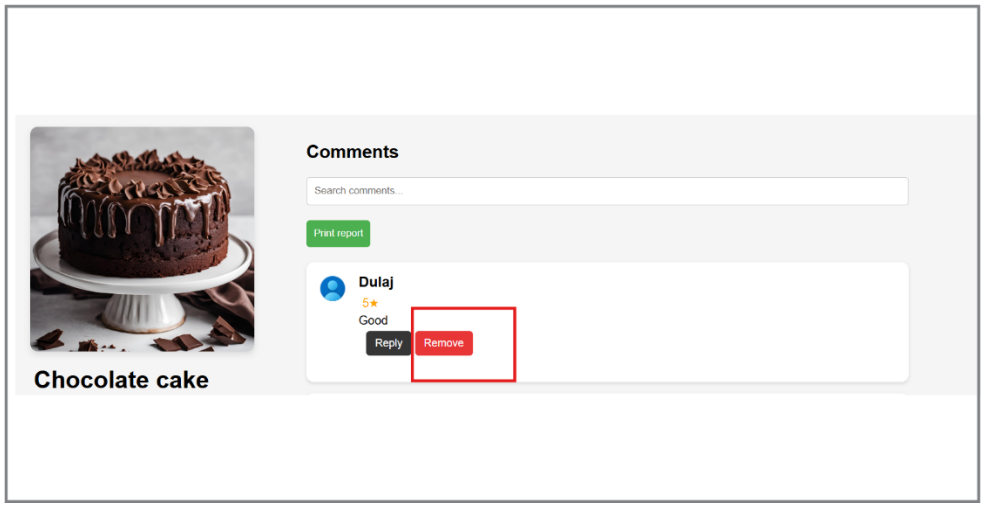


Figure 3. TC1.3- Check whether rating and reviews are delete correctly

2. Check whether searching bar is working properly or not

Figure 4.TC2-. Check whether searching bar is working properly or not

3. Verify can not submit rating without reviews (validation)

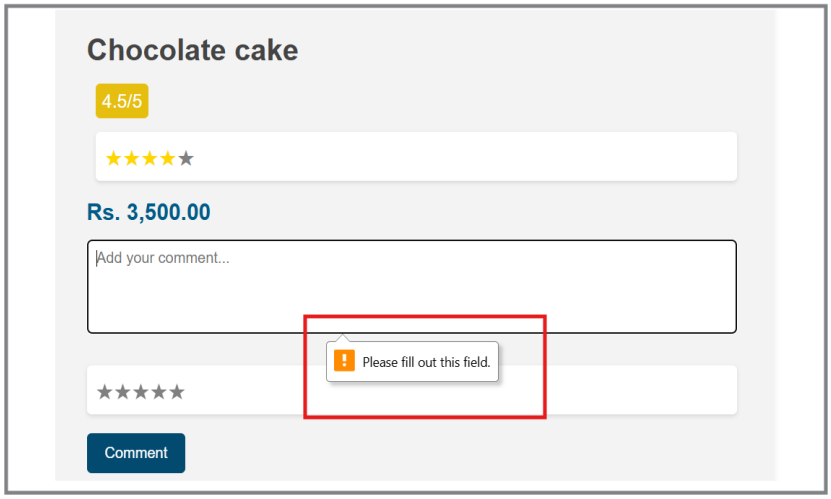


Figure 5.TC3.Verify can not submit rating without reviews (validation)

3. Check whether average rating sort by correctly low to high and high to low.

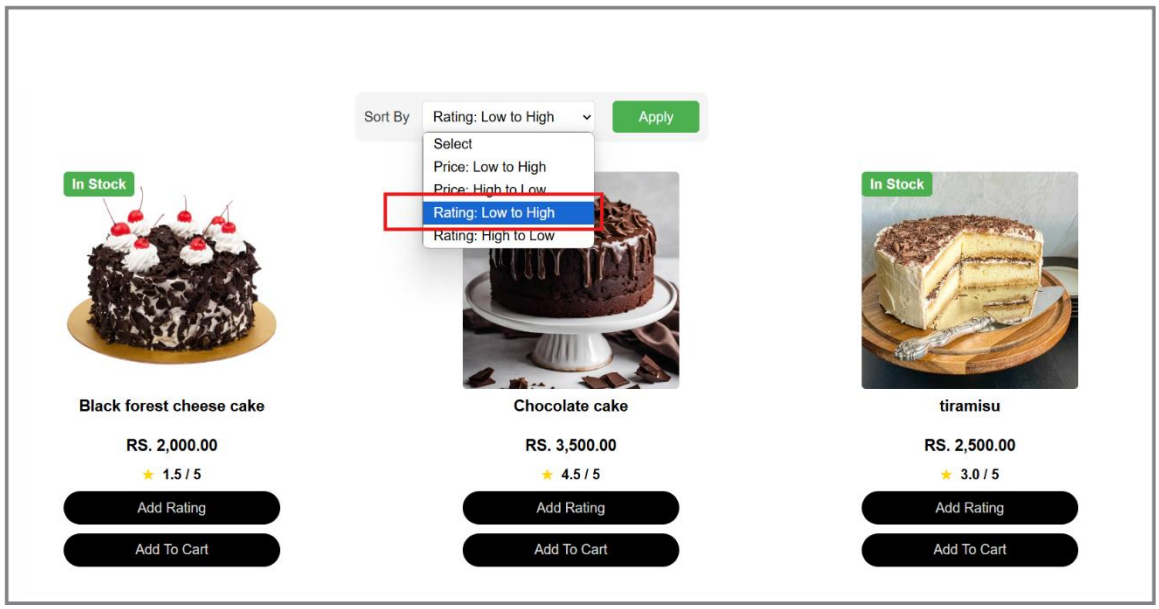
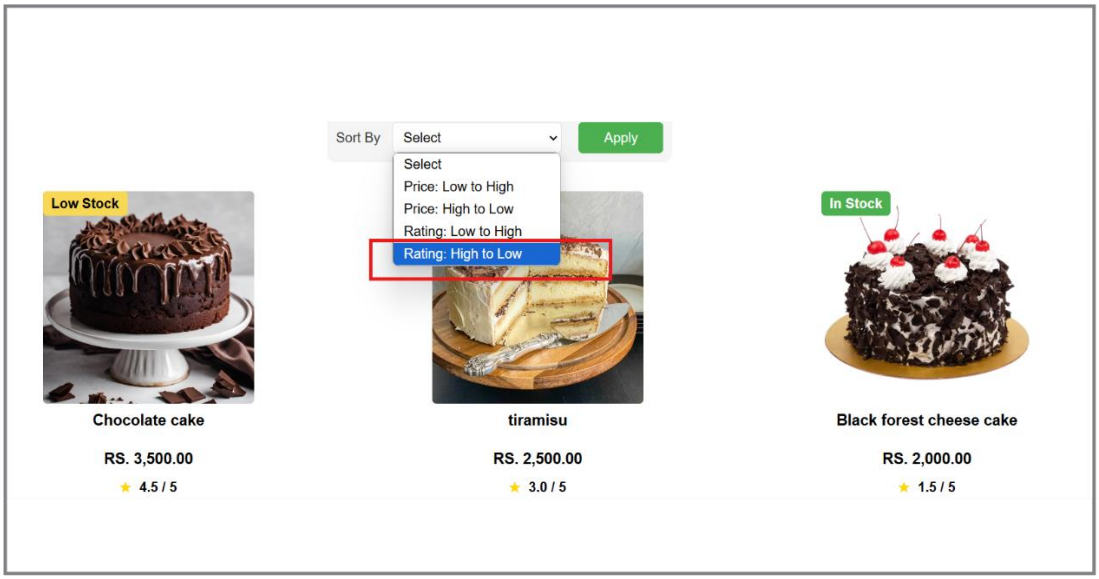


Figure6.TC4.Check whether average rating sort by correctly low to high and high to low.



4. Check whether add reply correctly and delete reply correctly.

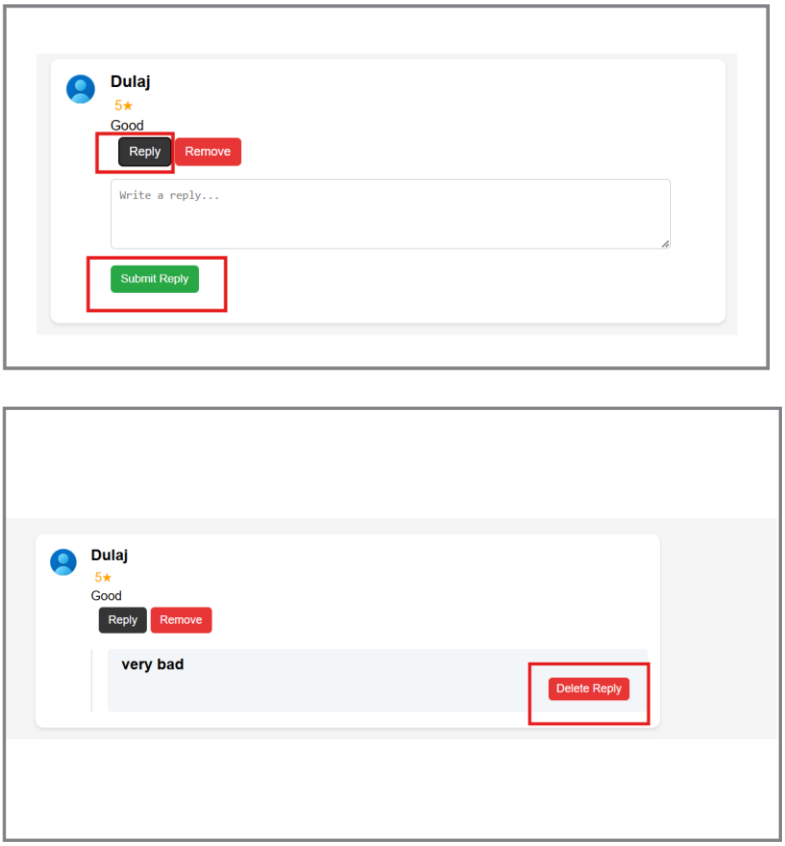


Figure7-TC5-Check whether add reply correctly and delete reply correctly.

5. Check whether average rating and reviews calculate correctly when add rating and update rating.

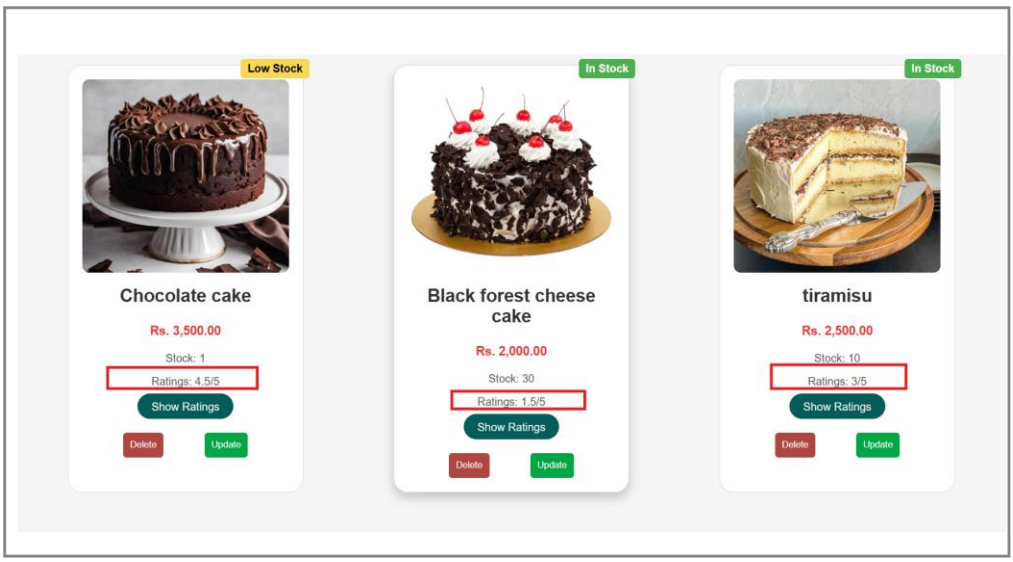


Figure 8-TC6-Check whether average rating and reviews calculate correctly when add rating and update rating.

3. Evaluation

3.1 Assessment of the Project results

The data and information were analyzed using various approaches. Interviews, discussions, and analyses of corporate documents that may be used in the development of the system were among them. The system will display error messages if the user does not input proper values. we can add, edit, update, and delete the details in the system. JavaScript was planned to be used for this website. But it was not used due to the time constraint. And instead of that HTML, CSS, and SQL were used.
We have conducted several tests to analyze the data. we conducted user testing using a small group of people to identify the issues arising when using the system and feedback was collected to determine the overall satisfaction through surveys and interviews. Functional testing was done by ensuring all the predefined requirements were achieved through the system and focused on features like cake browsing, placing orders, etc. Performance testing was done by evaluating the load time, and responsiveness of the website query testing was done by verifying the accuracy and efficiency of the SQL queries used. We did not find any failures in this online cake-ordering system. Some of the improvements we have to make are advanced user interfaces, an order tracking option, and cash on delivery option.

3.2 Lessons Learned

We learned the importance of planning and Time management during this project and were able to effectively finish the job in less than three months and recognized that we needed to balance time between designing, backend and front-end development, and testing. During this period, we gained experience and understanding of the SDLC model and understood how to do a feasibility study and gather requirements, particularly by meeting with a real-world customer and addressing difficulties that arose in the firm. In addition, while working on this project, we all gained a better grasp of UML diagrams and created several UML diagrams for the project documentation. We realized that taking user feedback throughout the development process will help to identify the issues in the design and ensure that user expectations are met. Throughout this project, we paid more attention to the security of the user details. Attractive user interfaces will help to retain the customers and communication with the client is critical to make a project successful.

3.3 Future Work

As previously said, this system was created and built to meet the needs of the users.
We hope to find a new payment calculation and Billing system and a quantity survey system and make it more friendly to users and managers. Adding an online order tracking system and cash on delivery options are some future works we need to do.

4. Conclusion

Cake Town Bakers is a small-scale business where the owner wants to expand the business by making an online cake ordering system for her business. This has four main components such as They are user management system, payment management, Feedback, and ratings and inventory management system.
After the case study, the system will save more money for this Cake Town Bakers cake shop which uses a manual way of handling and recording their orders, and customer management. This system will be helpful for the customers to easily order the cakes they want and the owner can expand her business more through this.

5. References

1. "C Programming Tutorial," 2021. [Online]. Available: https://www.learn-c.org/. [Accessed: 27 Nov 2024].

2. "Visual Studio Code Documentation," 2024. [Online]. Available: https://code.visualstudio.com/docs. [Accessed: 27 Nov 2024].

3. "PHPMyAdmin Documentation," 2024. [Online]. Available: https://www.phpmyadmin.net/docs/. [Accessed: 27 Nov 2024].

4. "XAMPP Official Documentation," 2024. [Online]. Available: https://www.apachefriends.org/index.html. [Accessed: 27 Nov 2024].

5. "Cake Ordering System with PHP and MySQL," 2021. [Online]. Available: https://www.tutorialspoint.com/php/php_mysql_example.htm. [Accessed: 27 Nov 2024].

6. "W3Schools C Programming Tutorial," 2024. [Online]. Available: https://www.w3schools.com/c/. [Accessed: 27 Nov 2024].

7. "Getting Started with XAMPP," 2024. [Online]. Available: https://www.geeksforgeeks.org/how-to-install-and-setup-xampp-on-windows/. [Accessed: 27 Nov 2024].

8. "PHPMyAdmin User Guide," 2024. [Online]. Available: https://docs.phpmyadmin.net/en/latest/. [Accessed: 27 Nov 2024].

9. "Debugging C Code in Visual Studio Code," 2024. [Online]. Available: https://code.visualstudio.com/docs/cpp/cpp-debug. [Accessed: 27 Nov 2024].

10. "C Language File Handling for Beginners," 2024. [Online]. Available: https://www.programiz.com/c-programming/c-file-input-output. [Accessed: 27 Nov 2024].

Appendix A: Design Diagrams

1. Context diagram

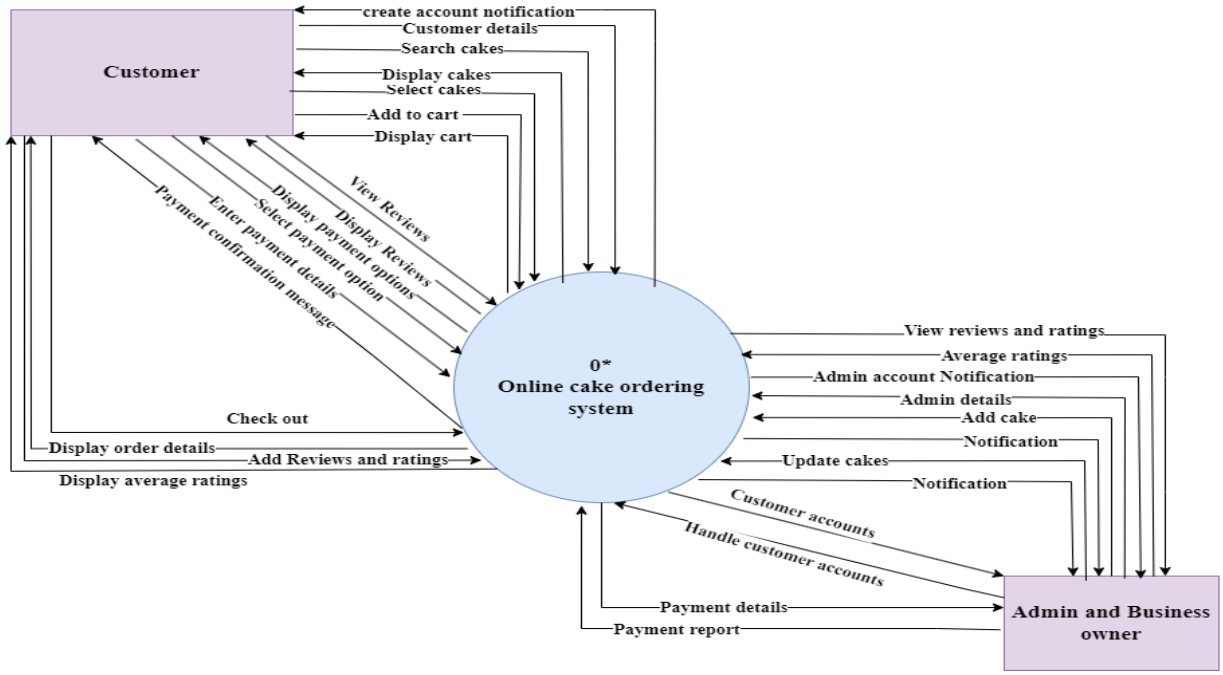


Figure 1.Appendix A. Context diagram

Level 1 DFD:

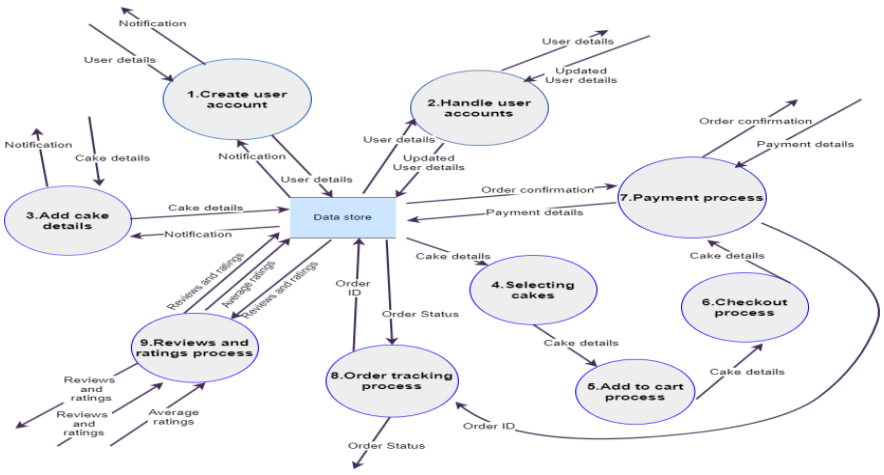


Figure 2.LEVEL1 DFD diagram

Appendix B: Test Results

Rating and review management

1.Check whether average rating update when add new rating

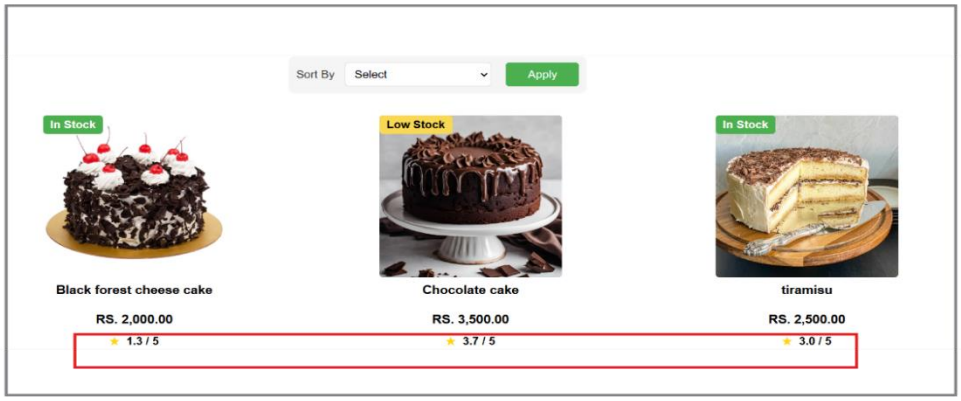


Figure 1 check correctly sort or not

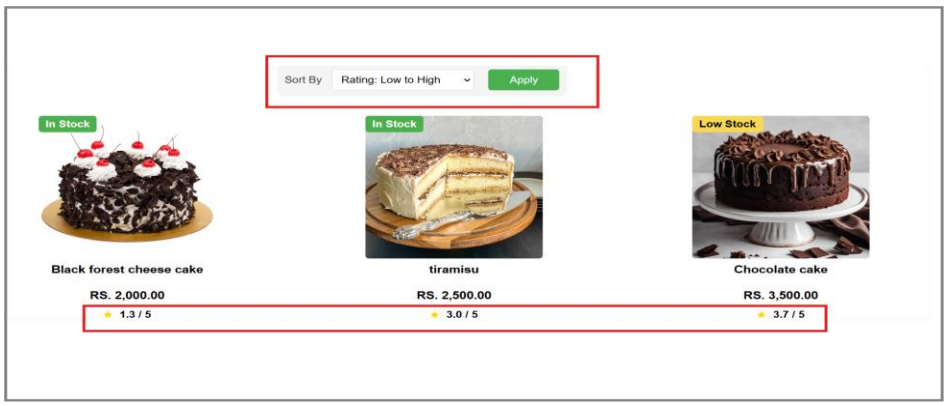
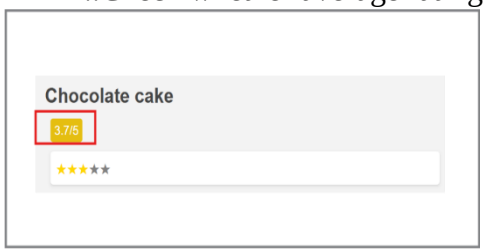


Figure 2 low to high

Figure 3 high to low

2..Check whether average rating update when add new rating



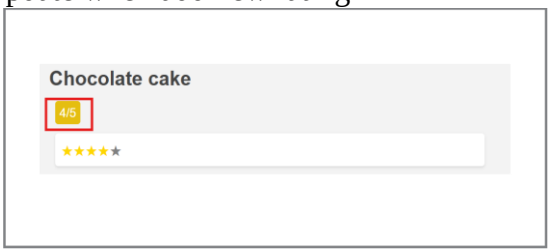


Figure 4 before update average rating

Figure 5 after update average rating

Checkout

01.Add cakes to the cart

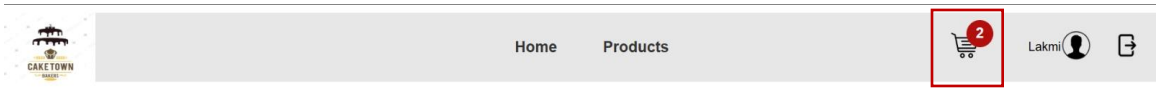


Figure 1add cakes to the cart

02.Proceed to checkout

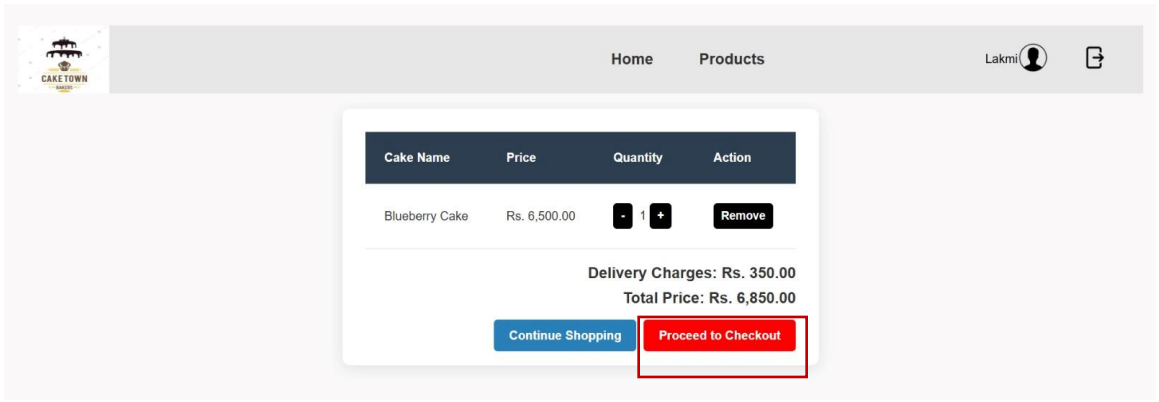


Figure 2 proceed to checkout

03.Validate payment details

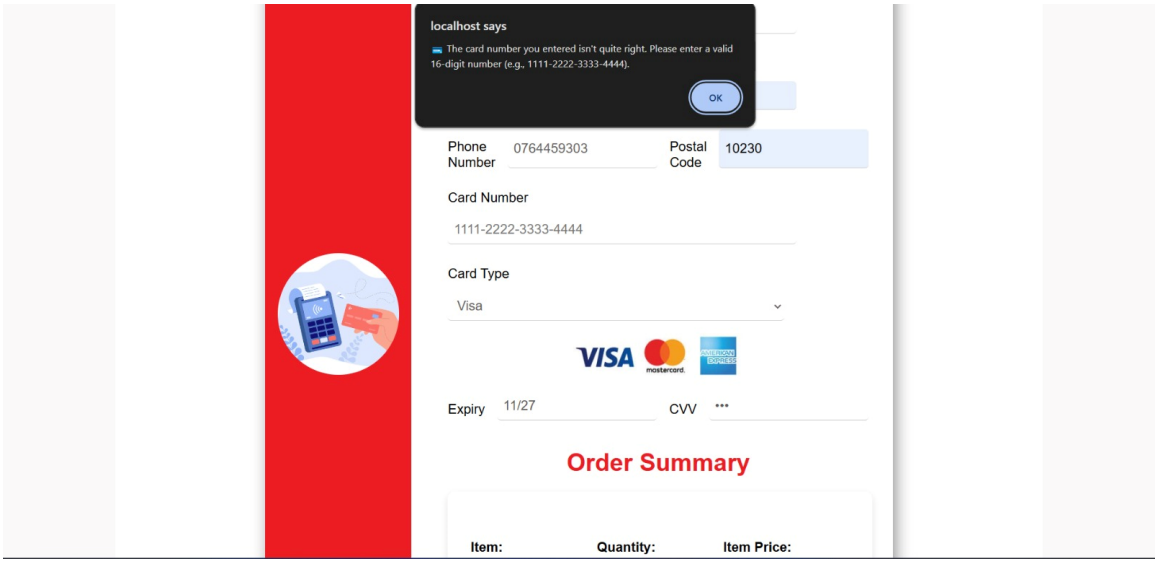


Figure 3 validate payment details

04.Display order confirmation message

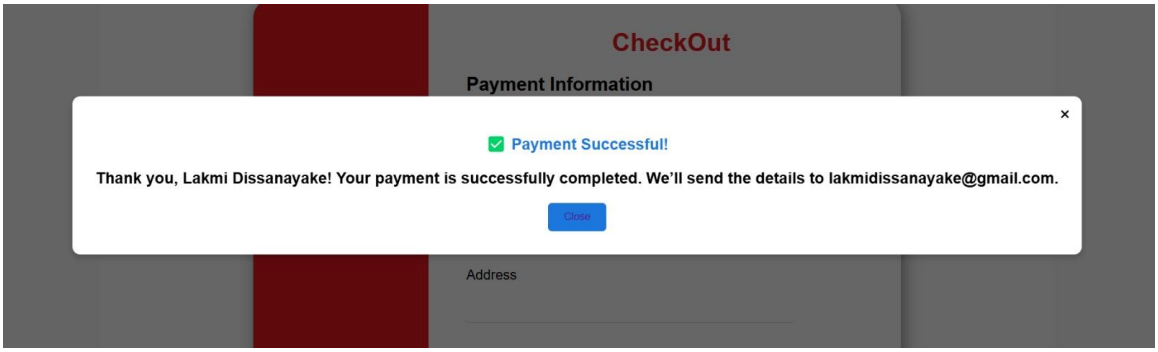


Figure 4 display order confirmation message

3. User account management

1. Display loyalty points

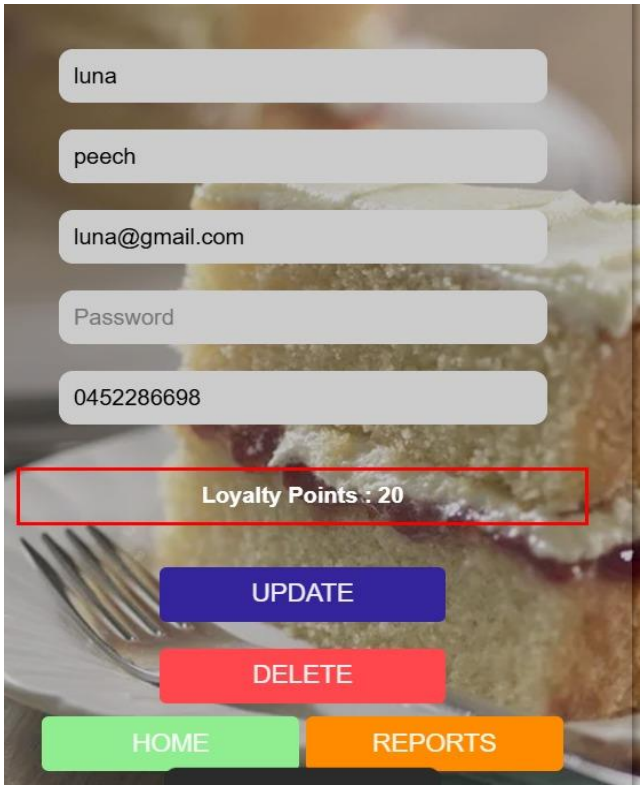


Figure 3.1:Display loyalty points

Appendix C: Selected Code Listings

Rating and review management

A. The code section is for Check whether average rating sort by correctly low to high and high to low correctly.

```
<!-- Filter section -->
<div class="filter-section">
    <form method="POST" action="" style="display: flex;">
        <div class="filter-group" >
            <label for="sort_by" style="margin-right: 10px;">Sort By</label>
            <select name="sort_by" id="sort_by">
                <option value="">Select</option>
                <option value="price_low_high">Price: Low to High</option>
                <option value="price_high_low">Price: High to Low</option>
                <option value="rating_low_high">Rating: Low to High</option>
                <option value="rating_high_low">Rating: High to Low</option>
            </select>
        </div>
        <button type="submit" class="apply-btn">Apply</button>
    </form>
</div>
```

Figure 1.A.The code section is for Check whether average rating sort by correctly low to high and high to low correctly.

```
// Construct the SQL query based on the selected filter
$orderBy = "name"; // Default ordering
if ($sort_by == 'price_low_high') {
    $orderBy = "price ASC";
} elseif ($sort_by == 'price_high_low') {
    $orderBy = "price DESC";
} elseif ($sort_by == 'rating_low_high') {
    $orderBy = "average_rating ASC";
} elseif ($sort_by == 'rating_high_low') {
    $orderBy = "average_rating DESC";
}
```

Bode section for check whether average rating calculate correctly when add and update rating.

```
$averageRatingResult = $conn->query("SELECT AVG(rating) AS average_rating FROM comments WHERE cake_id = $cake_id");
$averageRating = $averageRatingResult->fetch_assoc()['average_rating'];
$averageRating = round($averageRating ?? 0, 1);
```

Figure 2B.Code section for check whether average rating calculate correctly when add and update rating.

C..Code section for check whether search bar is working properly.

```php
if (isset($_POST['search'])) {
    $search_keyword = $_POST['search'];
    $search_param = "%" . $search_keyword . "%";
    $comments_query = $conn->prepare("SELECT * FROM comments WHERE cake_id = ? AND comment LIKE ?");
    $comments_query->bind_param("is", $cake_id, $search_param);
    $comments_query->execute();
```

Figure 3C..Code section for check whether search bar is working properly.

D. Code section for verify can not submit rating with out reviews (validation).

```javascript
    function validateForm() {
        const commentText = document.getElementById('comment-text').value.trim();
        const errorMessage = document.getElementById('error-message');

        if (commentText === '') {
            errorMessage.textContent = 'Comment cannot be empty.';
            errorMessage.style.display = 'block';
            return false; // Prevent form submission
        }

        errorMessage.style.display = 'none'; // Hide error message if validation passes
        return true; // Allow form submission
}
```

Figure 4D Code section for verify can not submit rating with out reviews (validation).

Checkout

A. Code for verifying the successful payment

```
559
560   <div id="popupMessage" class="popup hidden">
561       <div class="popup-content">
562           <span class="close-btn" onclick="closePopup()">&times;</span>
563           <h2>✅ Payment Successful!</h2>
564           <p id="popupText"></p>
565           <button class="popup-btn" ><a href="./Inventory/products.php">Close</a></button>
566           </div>
567   </div>
568
```

User account management

```php
$type = $_POST['user_type'];

// Validate user type
if ($type !== "user" && $type !== "admin") {
    die("Invalid user type provided.");
}

// Proceed with user registration
$sql = "INSERT INTO user (user_id, fname, lname, email, pass, contact, loyalty, type) VALUES(?,?,?,?,?,?,?,?)";

$stmt = $conn->stmt_init();

if (!$stmt->prepare($sql)) {
    die("SQL Error: " . $conn->error);
}

$stmt->bind_param("ssssssis", $new_user_id, $fname, $lname, $email, $password_hash, $contact, $loyalty, $type);

if ($stmt->execute()) {
    header("Location: register-success.php");
    exit;
} elseif ($conn->errno === 1062) {
    die("Email already exists");
} else {
    die($conn->error . " " . $conn->errno);
```

A. User registration

B. check whether Login credentials true

```php
  // Check if the user exists
  if ($stmt->num_rows > 0) {
if (isset($_POST['delete'])) {                                    e, $type);
    include 'config.php';
    $user_id = $_SESSION['loggedin_id'];
```

Figure 1.A: User registration

```php
    $sql1 = "DELETE from user WHERE user_id='$user_id'";

    if ($conn->query($sql1) === TRUE) {

      session_start();
      session_unset();
      session_destroy();
      header("Location: login.php");

    } else {
      echo "Error updating record: " . $conn->error;
    }

    $conn->close();
```

Figure 1.B: Login credentials checking

```php
?>
```

Figure 1.C :Delete user account

C. Delete user account

D. Update user details

```php
if(empty($password)){

  $sql1 = "UPDATE user SET fname='$fname', lname='$lname', email='$email', contact='$contact' WHERE user_id='$user_id'";

  if ($conn->query($sql1) === TRUE) {

    echo "<div class='signup-sucess'>";
    echo "<center>";
    echo "<h1>" . "USER DETAILS HAS BEEN UPDATED SUCCESSFULLY" . "</h1>";
    echo "<h3>" . "Please check" . "<a href='profile.php'>" . " User Profile " . "</a>";
    echo "</h3>";
    echo "</center>";
    echo "</div>";

  } else {
    echo "Error updating record: " . $conn->error;
  }

  $conn->close();
} else {
  $password_hash = password_hash($password, PASSWORD_DEFAULT);
  $sql1 = "UPDATE user SET fname='$fname', lname='$lname', email='$email' ,pass='$password_hash' ,contact='$contact' WHERE user_id='$user_id

  if ($conn->query($sql1) === TRUE) {

    echo "<div class='signup-sucess'>";
    echo "<center>";
    echo "<h1>" . "USER DETAILS HAS BEEN UPDATED SUCCESSFULLY" . "</h1>";
    echo "<h3>" . "Please check" . "<a href='profile.php'>" . " User Profile " . "</a>";
    echo "</h3>";
    echo "</center>";
    echo "</div>";
```

Figure 1.D: Update user details

```php
$totalprice = $_POST['totalprice'];
$points = round($total_price/100);
$loyalty = $loyalty + $points;
```

Figure 1.E: calculate loyalty points

E. Calculate loyalty points