FOOD ORDERING SYSTEM

A MINI-PROJECT REPORT

Submitted by

LOGESHWARAN T - 2116220701145

in partial fulfilment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

HARDWORK DISCIPLINE REC

RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

RAJALAKSHMI ENGINEERING COLLEGE

AUTONOMOUS, CHENNAI

NOV-DEC 2024

BONAFIDE CERTIFICATE

Certified that this mini project "**Food Ordering System**" is the Bonafide work of "**LOGESHWARAN T (2116220701145)"** who carried out the project work under my supervision.

SIGNATURE
Mrs. JANANEE V,

Assistant Professor,
Computer Science Engineering

Rajalakshmi EngineeringCollege
Thandalam, Chennai -602105.

Submitted for the End semester practical examination to be held on ____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

I express my sincere thanks to my beloved and honourable chairman MR.S.MEGANATHAN and the chairperson DR.M.THANGAM MEGANATHAN for their timely support and encouragement.

I am greatly indebted to my respected and honourable principal Dr. S.N.MURUGESAN for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by my head of the department Dr. P. KUMAR, and my Academic Head Dr. R. SABITHA, for being ever supporting force during my project work.

I also extend my sincere and hearty thanks to my internal guide

Mrs. JANANEE V for her valuable guidance and motivation during the completion of this project.

My sincere thanks to my family members, friends and other staff members of Computer Science and Engineering.

ABSTRACT

Food ordering has evolved significantly with the advent of digital technologies, transforming how people access and consume meals. Traditionally, food ordering involved direct communication between customers and restaurants, primarily through telephone calls or in-person visits. However, the rise of online platforms and mobile applications has revolutionized the process, making it faster, more efficient, and accessible from virtually any location. This shift has been driven by changing consumer preferences, technological advancements, and the growing demand for convenience in everyday life.

Online food ordering systems allow customers to browse a wide variety of restaurant menus, customize their orders, and pay through digital platforms. These systems have integrated features like real-time tracking, loyalty programs, and personalized recommendations based on user preferences. The convenience of accessing multiple cuisines from a single platform has contributed to the widespread adoption of these services. Furthermore, digital payment options have made transactions smoother, with cashless payments gaining popularity due to their ease and security.

The food ordering ecosystem typically involves three primary stakeholders: customers, restaurants, and delivery personnel. Customers benefit from a simplified and convenient ordering process, while restaurants can expand their reach beyond physical foot traffic, increasing their customer base. Delivery personnel, often associated with third-party logistics services, facilitate the rapid and reliable delivery of food.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

The rapid growth of the food delivery industry has also presented several challenges. Environmental concerns have been raised about the increased use of single-use packaging and the carbon footprint associated with food deliveries. The gig economy model used by many delivery platforms has faced scrutiny over issues related to worker rights, including job security, fair wages, and working conditions. In response, some companies have begun exploring more sustainable practices, such as using eco-friendly packaging and electric vehicles for deliveries.

1.2 SCOPE OF THE WORK

The scope of food ordering has expanded significantly in recent years, driven by technological advancements, changing consumer preferences, and the increasing importance of convenience in daily life. Food ordering systems encompass a wide range of services, from traditional in-person restaurant visits and phone orders to modern online platforms and mobile applications that offer delivery and takeaway options.

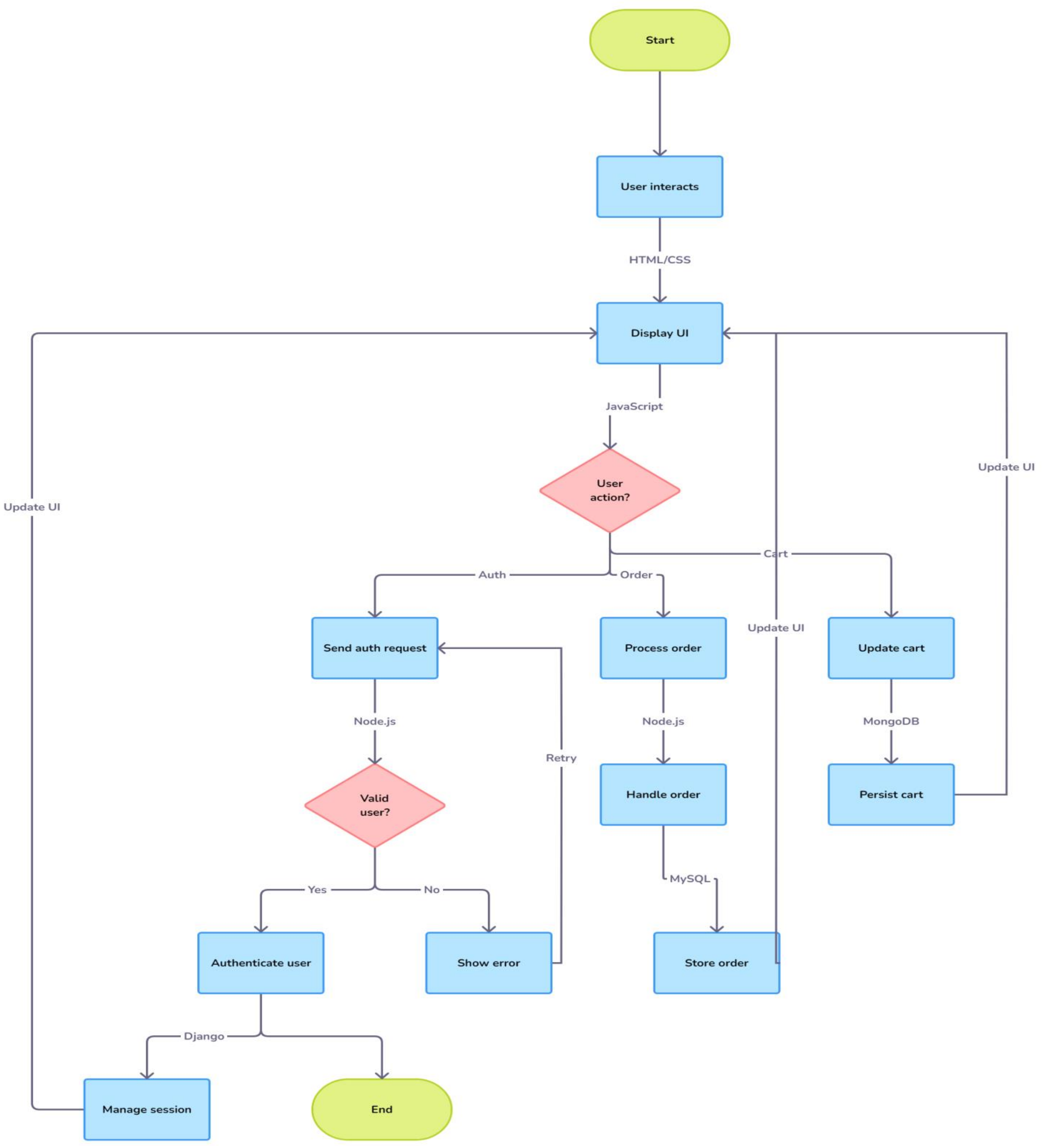
CHAPTER 2

2. SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 11
Front – End	:	HTML,CSS,JAVASCRIPT,SCSS,PHP
Back – End	:	PHP,MYSQL

CHAPTER 3

ARCHITECTURE DIAGRAM



CHAPTER 4

MODULE DESCRIPTION

4.1. User Registration and Login Module:

The User Registration and Login Module for the Food Ordering System is designed to provide secure and efficient access to authorized users, including food lovers and administrators. Users can create profiles with essential information, undergo verification processes, and receive role-based access, ensuring accountability and data integrity.

4.2. Menu Module:

The menu page of the project serves as the heart of the online food ordering system, showcasing a variety of Indian dishes in an engaging and interactive carousel format. Each dish is presented with a vibrant image, name, and price, allowing users to visually browse the menu items. The page features a smooth, user-friendly navigation system with left and right arrows to scroll through the dishes.

4.3. Cart Module:

The cart page in this project is designed to display items that the user has added to their shopping cart. It is styled with a clean and simple interface, using HTML and CSS for layout and presentation. The page features a table where each row represents an item in the cart, displaying the item name, price, and quantity. Users can modify the quantity of items directly in the cart and remove items if needed. Each item is retrieved from local Storage, ensuring the cart persists even when the user navigates away from the page.

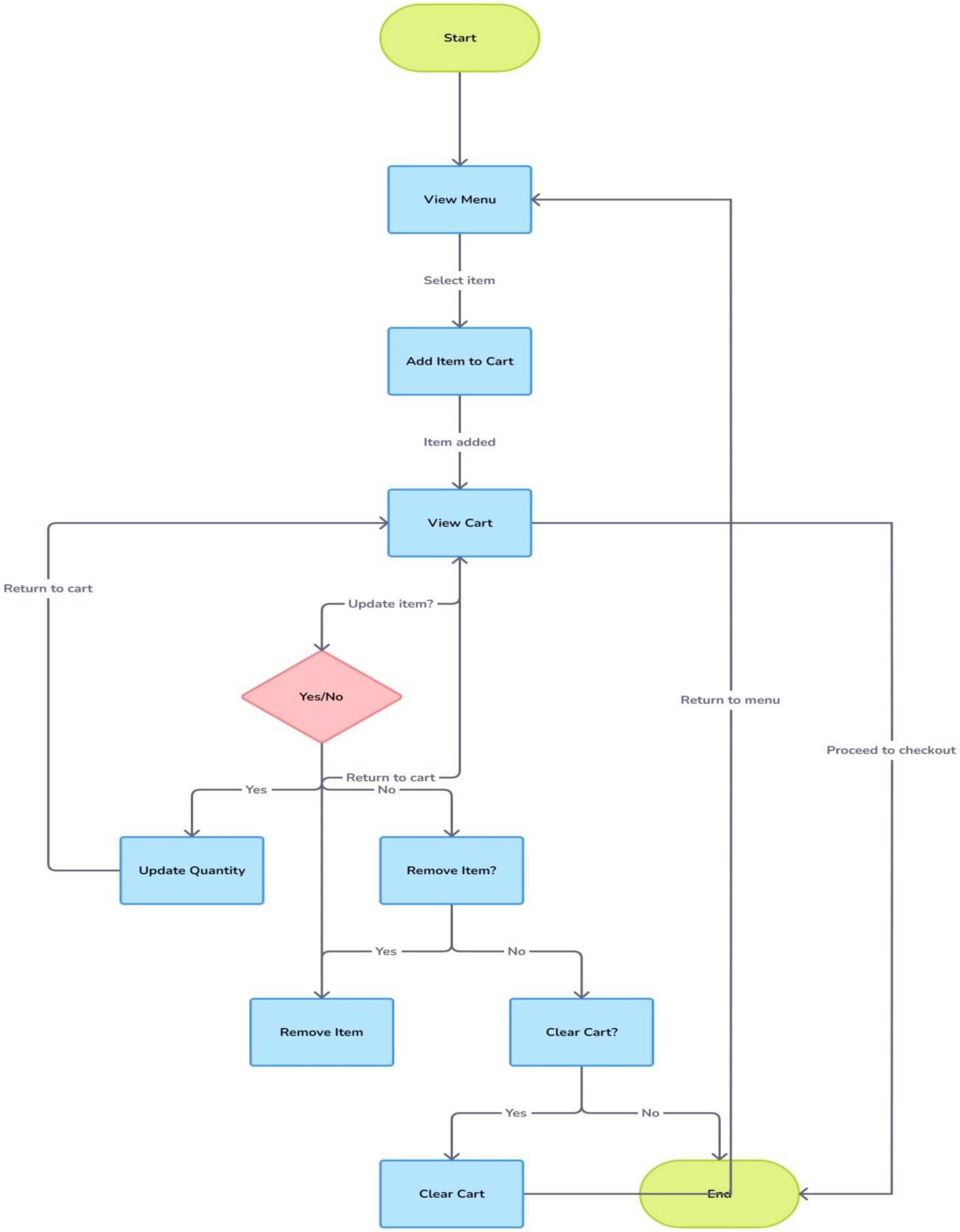
4.4. Admin Module:

The Admin Module within the Food Ordering Management System serves as the command center, equipping administrators with essential tools for overseeing and managing the entire spectrum of activities. It enables efficient user management, allowing the addition, modification, and deactivation of accounts with diverse roles and permissions. The module provides a comprehensive dashboard with analytics and reporting capabilities, facilitating data-driven decision-making and community engagement. Administrators can configure system settings, manage the plant database, and communicate critical updates to users. With robust security features, incident response tools, and collaboration functionalities, the Admin Module ensures a coordinated, secure, and adaptive response to user needs, maintaining the integrity and efficiency of the food ordering and management system.

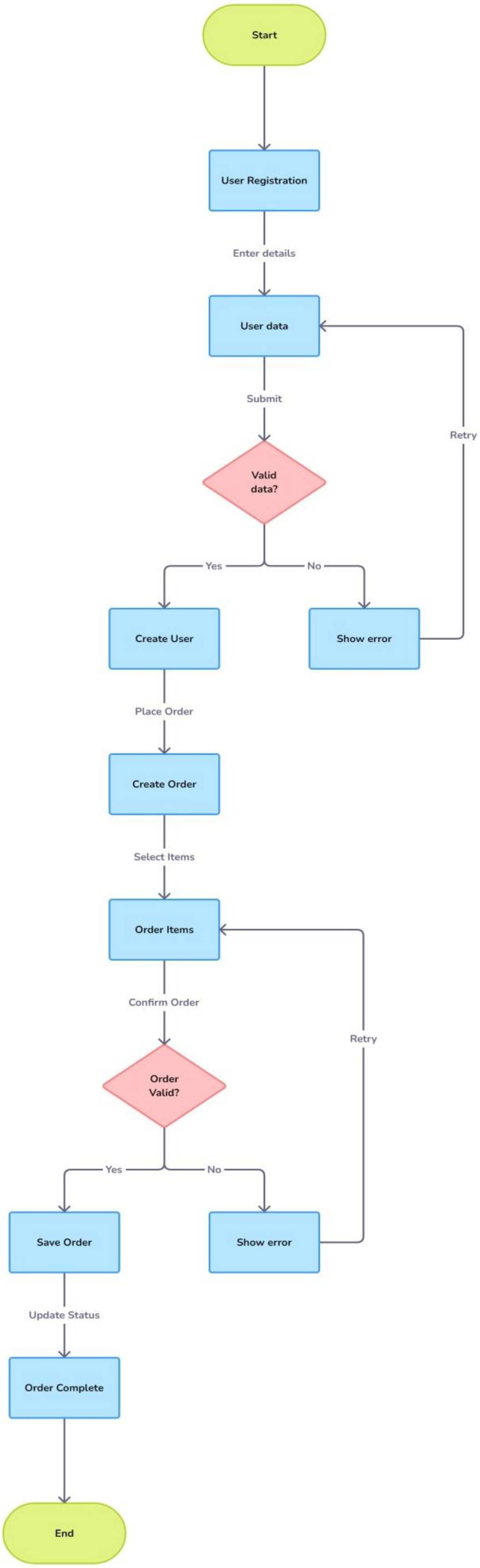
CHAPTER 5

SYSTEM DESIGN

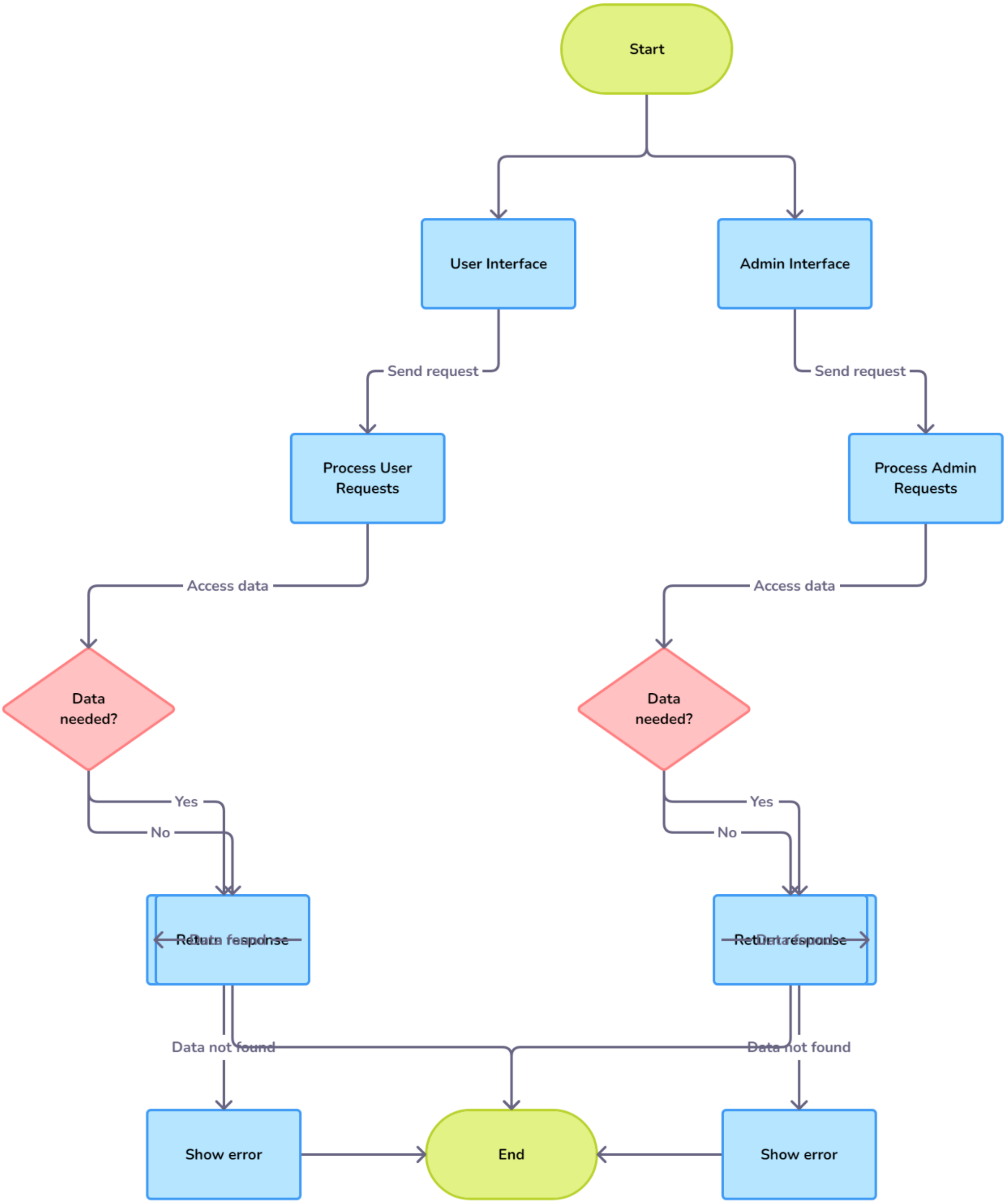
5.1 USE CASE DIAGRAM



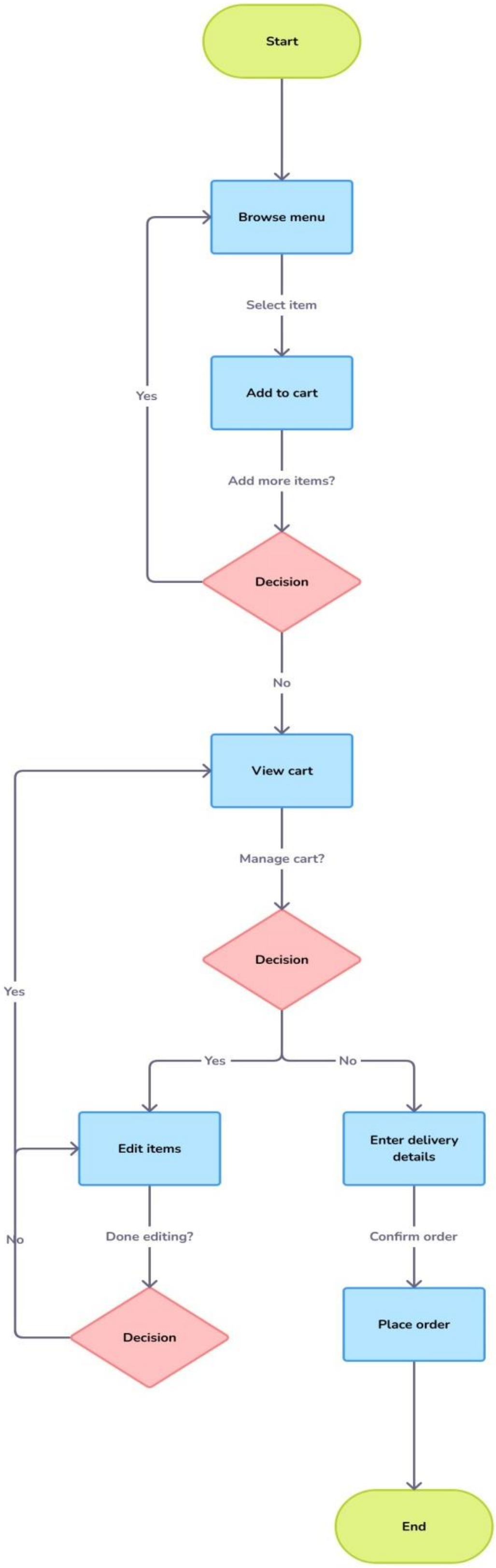
5.2ER DIAGRAM



5.3 DFD DIAGRAM



5.4 ACTIVITY DIAGRAM



CHAPTER 6

SAMPLE CODING

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Shopping Cart</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f0f0f0;
padding: 20px;
}

h1 {
text-align: center;
color: #ff5733;
}

.cart-container {
max-width: 600px;
margin: 0 auto;
background-color: white;
padding: 20px;
```

```
border-radius: 8px;
box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
}

table {
width: 100%;
border-collapse: collapse;
margin-bottom: 20px;
}

table, th, td {
border: 1px solid #ddd;
}

th, td {
padding: 12px;
text-align: left;
}

th {
background-color: #ff5733;
color: white;
}

.total {
font-size: 18px;
font-weight: bold;
```

```
text-align: right;
}

.remove-item {
background-color: #e74c3c;
color: white;
border: none;
padding: 6px 12px;
cursor: pointer;
border-radius: 5px;
}

.remove-item:hover {
background-color: #c0392b;
}

.clear-cart {
display: block;
width: 100%;
background-color: #ff5733;
color: white;
text-align: center;
padding: 10px;
border: none;
border-radius: 5px;
cursor: pointer;
}
```

```
}

.clear-cart:hover {
background-color: #e74c3c;
}

.back-link {
display: block;
text-align: center;
margin-top: 20px;
color: #ff5733;
text-decoration: underline;
}

.back-link:hover {
color: #e74c3c;
}

input[type="number"], input[type="text"] {
width: 100%;
padding: 8px;
margin: 10px 0;
border: 1px solid #ddd;
border-radius: 5px;
}

/* Reduce the size of the quantity input field */
input[type="number"] {
```

```
width: 60px; /* Adjust width to your preference */
}

.order-btn {
width: 100%;
background-color: #28a745;
color: white;
padding: 10px;
border: none;
border-radius: 5px;
cursor: pointer;
}

.order-btn:disabled {
background-color: #ccc;
cursor: not-allowed;
}
</style>
</head>
<body>

<h1>Your Cart</h1>

<div class="cart-container">
<table id="cart-table">
<thead>
<tr>
```

```
<th>Item</th>
<th>Price (₹)</th>
<th>Quantity</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<!-- Cart items will be dynamically inserted here -->
</tbody>
</table>

<p class="total">Total: ₹<span id="total-price">0</span></p>
<button class="clear-cart">Clear Cart</button>

<!-- Add delivery address and phone number fields -->
<div class="delivery-form">
<form id="delivery-form">
<label for="address">Delivery Address:</label>
<input type="text" id="address" name="address" placeholder="Enter your delivery
address" required>

<label for="phone">Phone Number:</label>
<input type="text" id="phone" name="phone" placeholder="Enter your phone
number" required pattern="\d{10}">

<!-- Order button, initially disabled -->
<button type="submit" class="order-btn" id="order-btn" disabled>Place
Order</button>
```

```
</form>
</div>

<a class="back-link" href="menu_page.html">Back to Menu</a>

<script>
// Retrieve cart items from localStorage
let cart = JSON.parse(localStorage.getItem('cart')) || [];

// Display the cart items
function displayCart() {
const cartTable = document.getElementById('cart-table').querySelector('tbody');
const totalPriceElement = document.getElementById('total-price');
cartTable.innerHTML = "; // Clear previous content
let totalPrice = 0;

cart.forEach((item, index) => {
const row = document.createElement('tr');

// Create item name cell
const nameCell = document.createElement('td');
nameCell.textContent = item.name;
row.appendChild(nameCell);

// Create price cell
const priceCell = document.createElement('td');
priceCell.textContent = item.price;
```

```
row.appendChild(priceCell);

// Create quantity cell with editable input
const quantityCell = document.createElement('td');
const quantityInput = document.createElement('input');
quantityInput.type = 'number';
quantityInput.value = item.quantity || 1;  // Default to 1 if no quantity
quantityInput.min = 1;
quantityInput.addEventListener('change', function () {
updateQuantity(index, parseInt(quantityInput.value));
});
quantityCell.appendChild(quantityInput);
row.appendChild(quantityCell);

// Create remove button cell
const removeCell = document.createElement('td');
const removeButton = document.createElement('button');
removeButton.textContent = 'Remove';
removeButton.classList.add('remove-item');
removeButton.addEventListener('click', () => {
removeItem(index);
});
removeCell.appendChild(removeButton);
row.appendChild(removeCell);

cartTable.appendChild(row);
```

```
// Add item price * quantity to total
totalPrice += parseInt(item.price) * (item.quantity || 1);
});

// Update total price
totalPriceElement.textContent = totalPrice;
}

// Update the quantity of an item
function updateQuantity(index, newQuantity) {
cart[index].quantity = newQuantity; // Update quantity in cart
localStorage.setItem('cart', JSON.stringify(cart)); // Save updated cart
displayCart(); // Refresh cart display
}

// Remove an item from the cart
function removeItem(index) {
cart.splice(index, 1); // Remove item at the given index
localStorage.setItem('cart', JSON.stringify(cart)); // Update localStorage
displayCart(); // Refresh cart display
}

// Clear the entire cart
document.querySelector('.clear-cart').addEventListener('click', function () {
cart = []; // Empty the cart
localStorage.setItem('cart', JSON.stringify(cart)); // Update localStorage
displayCart(); // Refresh cart display
```

```
});


// Enable or disable the order button based on form inputs
const addressInput = document.getElementById('address');
const phoneInput = document.getElementById('phone');
const orderButton = document.getElementById('order-btn');


function validateForm() {
const isAddressValid = addressInput.value.trim() !== '';
const isPhoneValid = phoneInput.value.trim().match(/^\d{10}$/); // Valid if exactly 10 digits


if (isAddressValid && isPhoneValid) {
orderButton.disabled = false;
} else {
orderButton.disabled = true;
}
}


// Listen for input changes to validate the form
addressInput.addEventListener('input', validateForm);
phoneInput.addEventListener('input', validateForm);


// Place order and display success message
orderButton.addEventListener('click', function() {
alert("Order successfully placed!");
cart = []; // Clear the cart after the order
```

```
localStorage.setItem('cart', JSON.stringify(cart)); // Update localStorage
displayCart(); // Refresh the cart display
addressInput.value = ''; // Clear the address input
phoneInput.value = ''; // Clear the phone input
orderButton.disabled = true; // Disable the order button
});

// Display the cart on page load
displayCart();
</script>

</body>
</html>
```

CHAPTER 7

SCREEN SHOTS

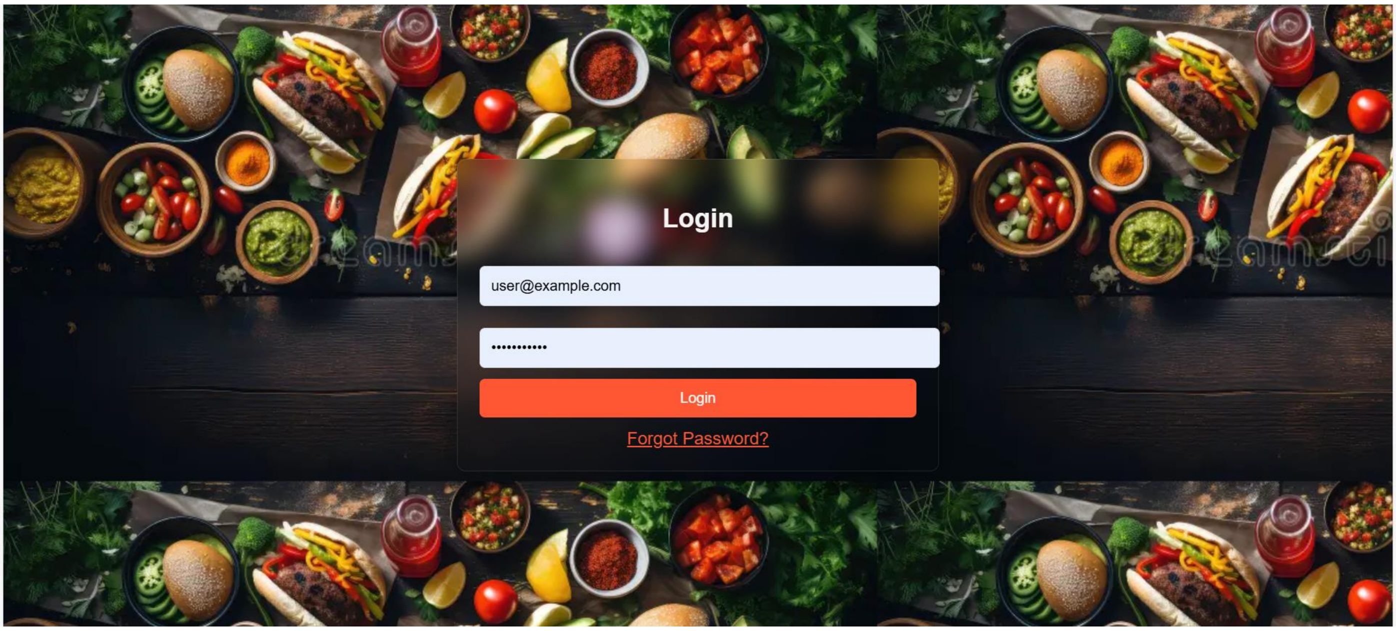


Fig. 7.1 User Login

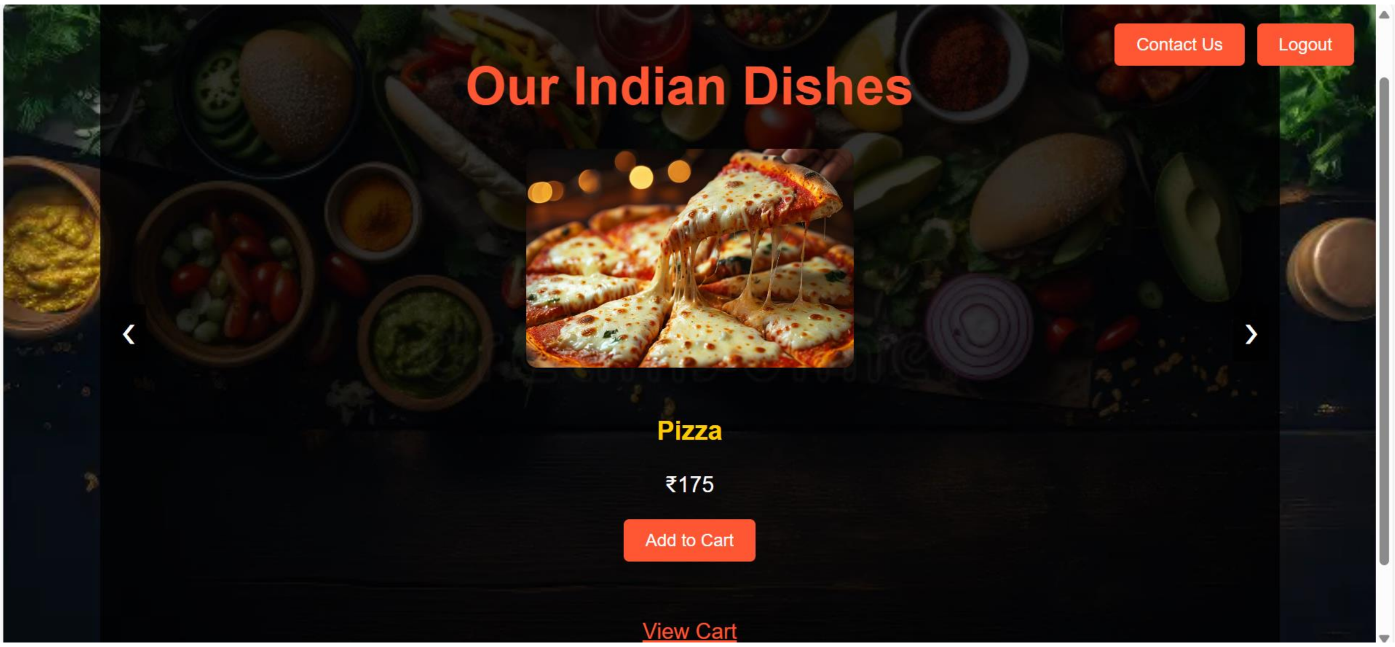
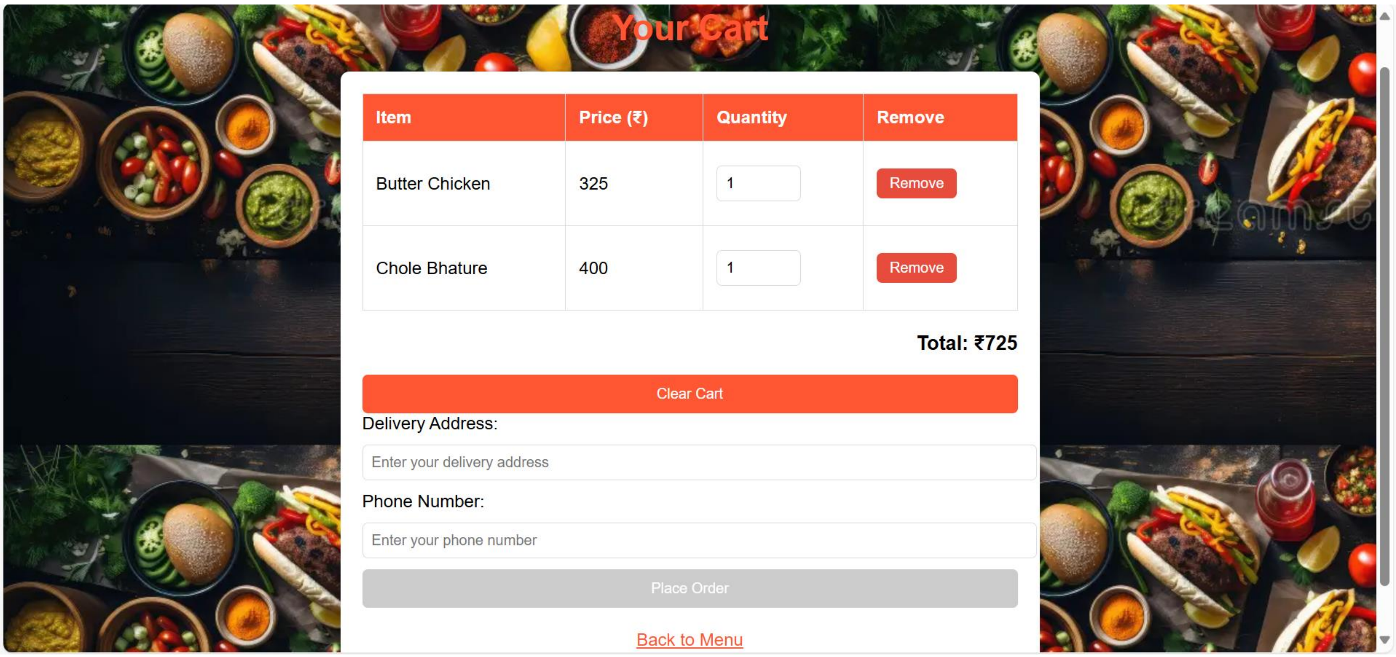


Fig. 7.2 Menu Module



Your Cart

Item	Price (₹)	Quantity	Remove
Butter Chicken	325	1	Remove
Chole Bhature	400	1	Remove

Total: ₹725

Delivery Address: Enter your delivery address

Phone Number: Enter your phone number

Fig. 7.3 Cart Module

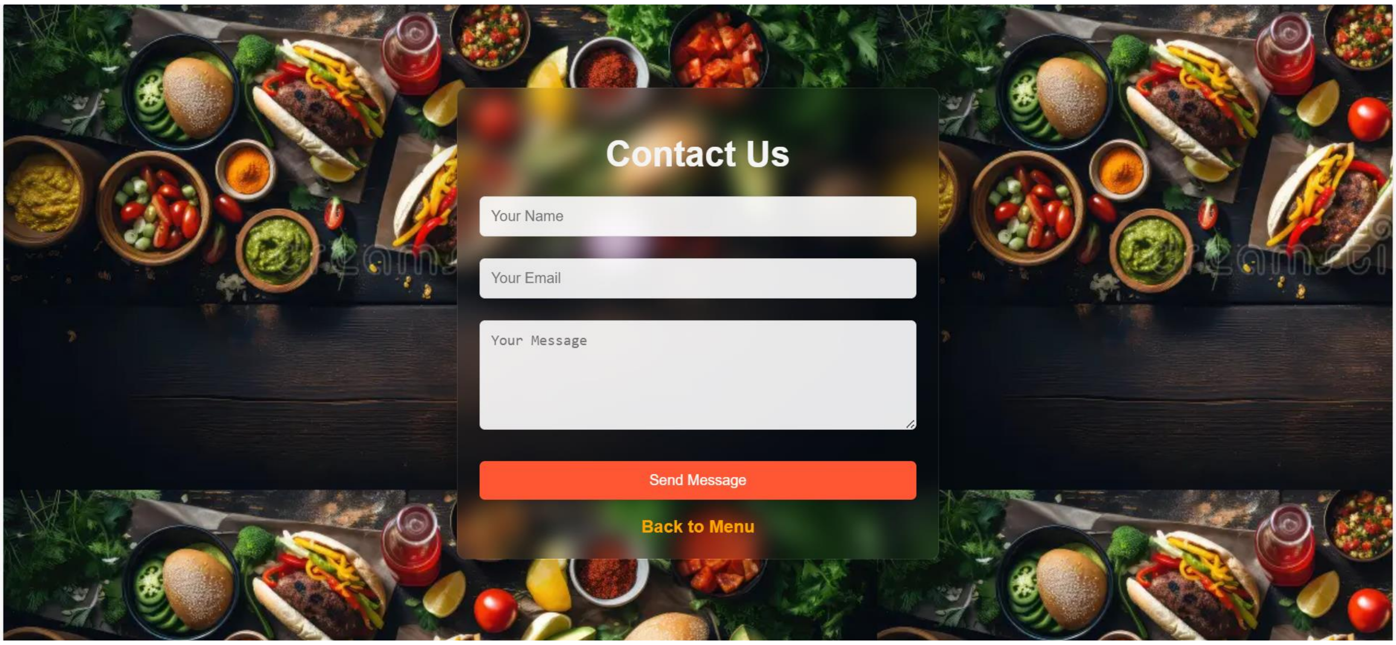


Fig. 7.4 Contact Us

CHAPTER 8

CONCLUSION

In conclusion, the food ordering system provides an efficient, user-friendly platform for customers to browse, select, and purchase their desired dishes online. Key features of the system include a dynamic menu display using a carousel interface, an integrated shopping cart that allows users to modify their orders, and an easy-to-use checkout process with real-time cart updates. The system enhances customer experience by supporting real-time notifications for actions such as adding items to the cart, and it ensures convenience with local storage management, maintaining the order even when the page is refreshed.

For the business, this food ordering system streamlines operations by allowing online orders to be processed efficiently, reducing the burden on staff, minimizing human error, and allowing a smooth transition from order placement to fulfillment. Additionally, features like address and phone verification ensure that orders are processed accurately, helping maintain customer satisfaction.

REFERENCES

1. HTML,CSS,JS–www.w3schools.com
2. PHP, MYSQL–www.youtube.com
3. Font Awesome Icons– www.fontawesome.com
4. PHP MY-ADMIN - https://www.phpmyadminonline.com/
5. Matt Lambert, Learning Bootstrap 4, Second Edition, Packt Publishing, 2016
6. Nate Murray, Felipe Coury, Ari Lerner, and Carlos, ng-book The Complete Guide to Angular, Fullstack.io, 2020