WEBSITE FOR ROOSTER RESTAURANT

A PROJECT REPORT

Submitted by

DHANUSH S

(Reg.No:22BSR006)

KRITHIK M S

(Reg.No:22BSR028)

MOULIDHARAN P

(Reg.No:22BSR032)

In partial fulfilment of the requirement

for the award of the degree of

BACHELOR OF SCIENCE

IN

SOFTWARE SYSTEMS

DEPARTMENT OF COMPUTER TECHNOLOGY – UG

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638 060

OCTOBER 2024

DEPARTMENT OF COMPUTER TECHNOLOGY – UG

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638 060

OCTOBER 2024

BONAFIDE CERTIFICATE

This is to certify that the project entitled "**WEBSITE FOR ROOSTER RESTAURANT**" is the bonafide record of project work done by **DHANUSH S (REG. NO: 22BSR006), KRITHIK MS (REG. NO: 22BSR028) and MOULIDHARAN P (REG. NO: 22BSR032)** in partial fulfilment for the award of Degree of Bachelor of science in SOFTWARE SYSTEMS of Anna University, Chennai during the academic year 2024 -2025.

SUPERVISOR **HEAD OF THE DEPARTMENT**

(Signature with Seal)

Date:

Submitted for the end semester viva-voce examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DEPARTMENT OF COMPUTER TECHNOLOGY – UG

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638 060

DECLARATION

We affirm that the project titled "**WEBSITE FOR ROOSTER RESTAURANT**" being submitted in partial fulfilment of the requirements for the award of Bachelor of Science in **SOFTWARE SYSTEMS** is the original work carried out by us. It has not formed part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this occasion on any other candidate.

DHANUSH S(Reg.No:22BSR006)

KRITHIK M S(Reg.No:22BSR028)

MOULIDHARAN P(Reg.No:22BSR032)

Date :

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date: **Name and Signature of the Supervisor**

ACKNOWLEDGEMENT

I respect and thank our Correspondent **Thiru A.K.ILANGO B.Com., MBA., LLB.,** and our Principal **Dr.V.BALUSAMY B.E (Hons)., M.Tech., PhD.,** and all others trust members of Kongu Vellalar Institute of Technology Trust who have always encouraged us in the academic and co-curricular activities.

I convey my gratitude and heartfelt thanks to our Head of the Department **Dr.S.KALAISELVI MCA, M.E, PhD.,** Department of Computer Applications, Kongu Engineering College, for his perfect guidance and support that made this work to be completed successfully.

We are in immense pleasure to express our hearty thanks to our beloved Project coordinator **Dr.K. SARASWATHI Msc., MPhil., PhD.,** and our guide **Dr.S.POORANI MPhil.,PhD.,** for providing valuable guidance and constant support throughout the course of our project. We also thank the teaching, non-teaching staff members, fellow students and our parents who stood with us to complete our project successfully.

TABLE OF CONTENTS

ABSTRACT

In today's corporate world, it is crucial to respond to customer needs as efficiently and in a timely manner. The project's primary objective is to create a method for clients to make reservations for restaurants online. This makes life easier for busy customers in their daily life. Now people are looking for rapidly developing technology. People invent and implement new technologies in all fields according to customer needs.

The project aims to implement a restaurant idea that helps people to save time. People prefer quick methods to get things done. Through this website the customer can pre-order products online before going to the restaurant and, Book a table for upcoming events, it also shows offers in the restaurant dynamically.

HTML, CSS and React.js are used for Front-end design, Node.js and MongoDB are used for Back-end development.

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The Pre-Order Food and Table Booking System for rooster restaurant is designed to streamline the dining experience by allowing guests to reserve tables and pre-order meals in advance. This system leverages modern technology to provide a seamless, efficient, and user-friendly interface for both hotel staff and guests.

Enhanced Customer Experience: Allow guests to pre-book tables and order meals before arrival, reducing wait times and improving overall satisfaction. It optimizes the table management and kitchen operations by providing advance notice of guest preferences and orders should increase efficiency. Data-Driven Insights: Collect and analyze data on dining preferences and peak times to improve service quality and menu planning.

Our site offers a table reservation system, enabling guests to select and book tables at their preferred times effortlessly. It features a comprehensive menu for pre-ordering meals, allowing guests to choose and customize their orders in advance. With real-time availability updates, users can stay informed about table availability and order status. The site supports secure user profiles, saving preferences, past orders, and special dietary requirements. Additionally, an automated notification system sends reminders and confirmations via email or SMS. Seamless integration with existing restaurant reservation and billing systems ensures a smooth and efficient experience for both guests and restaurant staff.

CHAPTER 2

SYSTEM SPECIFICATION

2.1 EXISTING SYSTEMS

In the existing system, there is no online platform for Rooster Restaurant. Customers have to visit the restaurant physically or call to make reservations or inquire about menu items. This process is time-consuming and inefficient. The restaurant currently uses a manual system for managing reservations and billing.

2.1.1 Drawbacks of the Existing System

The existing system has the following disadvantages:

1. Available offers and discounts may not reach customers effectively.
2. The admin cannot efficiently track table reservations and their statuses.
3. The restaurant has limited customer engagement due to the absence of an online presence.
4. The billing process is time-consuming and can lead to errors.

2.2 PROPOSED SYSTEM

The proposed system is a web application for Rooster Restaurant. It provides an interactive menu where customers can browse dishes and make table reservations online. The system allows users to filter menu items based on preferences (e.g., vegetarian, gluten-free), check table availability, and reserve tables. The admin can manage reservations, track customer

details, and generate reports. The admin also has full control over the menu details, such as updating item availability and pricing**.**

2.2.1 Advantages of the Proposed System

1. Data is stored in a database, ensuring efficient management and access.
2. The working time for both customers and staff is significantly reduced.
3. The admin can easily update menu availability and manage table reservations through the backend system.

2.3 FEASIBILITY STUDY

Three key aspects of feasibility analysis are:

- Economic Feasibility
- Operational Feasibility
- Technical Feasibility

2.3.1 Economic Feasibility

By using this website, customers can save time and make reservations online without needing to call or visit the restaurant. The restaurant benefits from improved efficiency in managing table bookings, leading to potential revenue increases by optimizing seating arrangements and reducing wait times.

2.3.2 Operational Feasibility

The admin will maintain the menu and reservation details, including dish images, descriptions, and prices. The admin will also be responsible for tracking table bookings. This

system is operationally feasible as it simplifies many current manual processes, making it easier to manage.

2.3.3 Technical Feasibility

The software technologies used are **ReactJS**, **CSS**, **Node.js**, and **MongoDB**. The system is scalable and can be updated easily in the future. No special hardware is required for using the system. Therefore, the project is technically feasible.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SOFTWARE REQUIREMENTS

Operating System	: Windows 11
Working Environment	: Visual Studio, Chrome
Frontend	: React JS, CSS
Backend	: MongoDB, Node JS

3.2 HARDWARE REQUIREMENTS

Processor	: Intel(R) Core (TM) i5-7020U CPU
RAM	: 4.00GB
Hard Disk	: 1TB
System Type	: 64-bit operating System, x64-based processor

3.3 SOFTWARE DESCRIPTION

3.3.1 FRONT END

React.js

Often referred to simply as React, is an open-source JavaScript library for building user interfaces. It was developed by Facebook and is now maintained by Facebook and a community of individual developers and companies. React is widely used for creating dynamic and

interactive web applications. Component-based React is centered around the concept of components. Developers create self-contained, reusable UI components that can be composed to build complex user interfaces. This promotes modularity and code reusability.

Enabling developers to create modular and composable UI components that can be reused across different parts of an application or even in multiple applications. Server-Side Rendering (SSR): React can be used for server-side rendering, which improves performance, search engine optimization (SEO), and initial page load times. Facebook and Instagram: React was initially developed by Facebook for use in its web applications, and it's also used by Instagram. This real-world usage demonstrates its robustness and scalability.

Cascading Style Sheets

"Cascading Style Sheet" is what CSS stands for. The layout of Web pages is formatted using Cascading Style Sheets. They can be used to specify font styles, table sizes, and other characteristics of web pages that were previously solely defined in the HTML of the page. CSS allows Web designers to establish a consistent look across several pages of a website. Rather than setting the style of each table and text block inside the HTML of a website, widely used styles can be defined once in a CSS sheet. Any page that references the CSS file can utilize the style provided in the cascading style sheet. CSS also makes it simple to update the style of several pages at once.

3.3.2 BACK END

MongoDB

MongoDB is a popular and versatile NoSQL database system that has gained widespread adoption in recent years. It stands out for its flexible, document-oriented data model, making it an excellent choice for applications that require dynamic, rapidly changing, or unstructured data. MongoDB uses a JSON-like format called BSON to store data, allowing

for the easy storage and retrieval of complex data structures. Its ability to horizontally scale across multiple servers makes it a scalable and high-performance solution for handling large volumes of data and high-traffic applications. One of MongoDB's notable features is its support for automatic sharding, which enables data distribution across multiple servers, improving both data distribution and fault tolerance. Additionally, MongoDB is known for its strong query capabilities, including support for geospatial queries, text search, and aggregation. Developers often find MongoDB appealing due to its use of familiar programming languages like JavaScript and its integration with a variety of programming frameworks and platforms. However, MongoDB's schema-less design can sometimes lead to challenges in maintaining data consistency and integrity, which requires careful consideration during application development. Overall, MongoDB has become a favored choice for modern, data-intensive applications, where its scalability, flexibility, and support for various data types make it a valuable addition to the database landscape.

Node JS

Node.js, a server-side JavaScript runtime, revolutionizes web development by enabling the execution of JavaScript code outside the browser. Built on Chrome's V8 JavaScript engine, Node.js offers high performance and scalability, making it ideal for building fast and efficient network applications. Its event-driven, non-blocking I/O model allows for handling concurrent connections with ease, enhancing application responsiveness. Node.js boasts a rich ecosystem of libraries and frameworks, such as Express.js, facilitating rapid development and deployment of web applications. With its lightweight and efficient design, Node.js empowers developers to build real-time, data-intensive applications that meet the demands of modern web development.

CHAPTER 4

SYSTEM DESIGN

4.1 MODULE DESCRIPTION

The project “WEBSITE FOR ROOSTER RESTAURANT” contains the following modules.

4.1.1 Admin Module

4.1.2. User Module

4.1.3. Cart Module

4.1.4. Payment Module

4.1.1. Admin module

The admin module contains the following functions, Admins have access to the dashboard. In the dashboard, admins can see all features in brief, like no of registered user, no of products and update the process of product ordered by the customer

Admin manage product categories (add, update and delete) in the category module as well as in the product page, He can order the products and cancel or return products.

4.1.2 User module

A new user will have to register in the system by providing details in order to view the products in the system. Already existing user can login using email and password. The user can view the product and order the product for them.

This module is divided into different sub-modules. Such as

- Register
- product
- Profile
- Order

4.1.3. Cart Module

This module is designed to help users easily manage their shopping experience while browsing and purchasing the products available in our website.

The features in this module are,

- Add product to cart with a single click.
- View and manage the products in the cart easily.
- Adjust quantities, remove products and update cart contents as needed.
- Seamlessly proceed to check out when the user ready to make a purchase.

4.1.4 Payment Module

This is module is an integral part of website, allowing user to securely and conveniently complete their purchase for product. Once at the checkout, the user can select payment methods and complete the transaction securely. The user can fill the details like address, mobile number and card details to make a payment.

Upon successful payment processing, user will receive a confirmation mail for the respective mail id.

4.2 USE CASE DIAGRAM

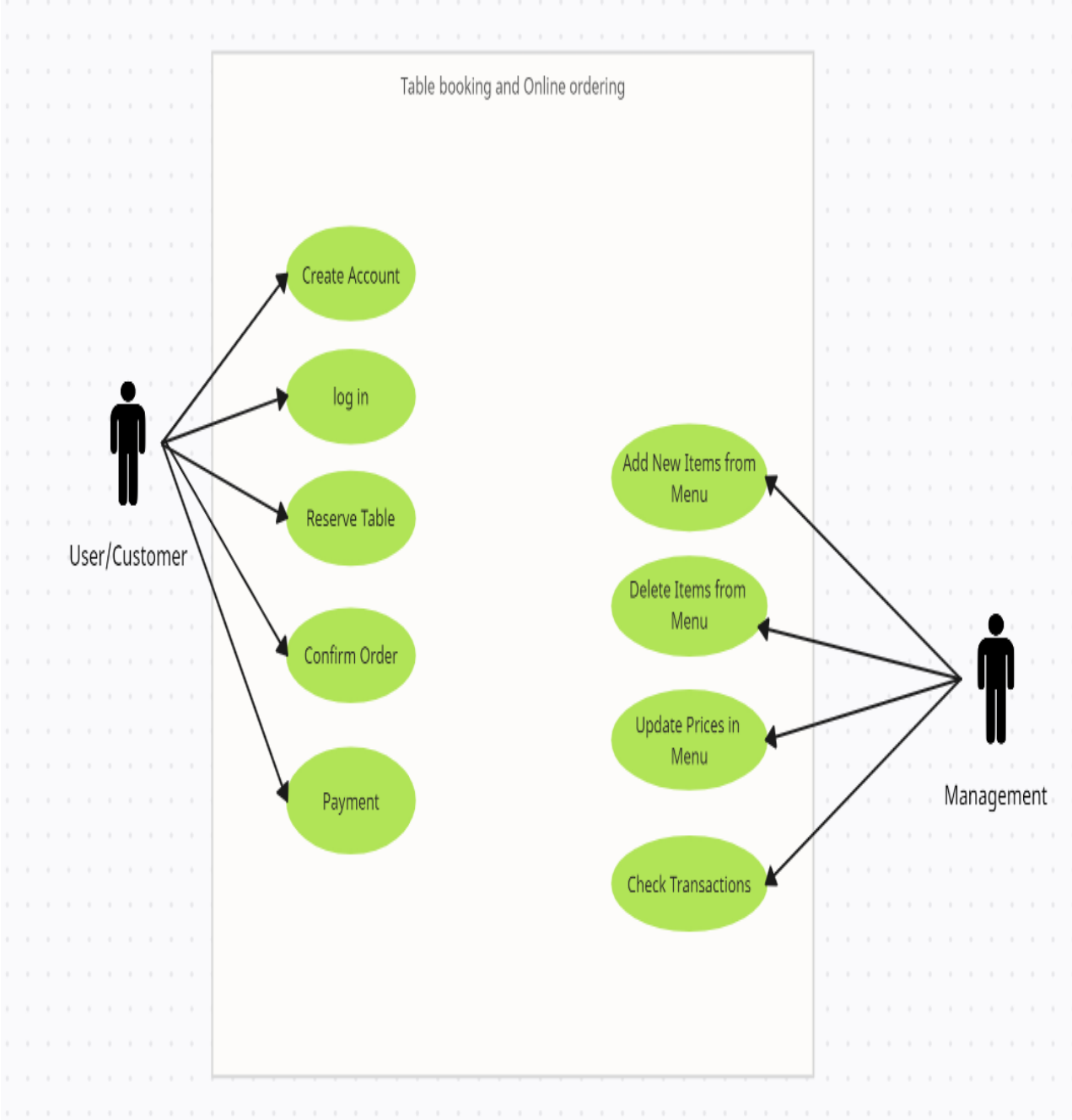

Figure 4.1 Use Case Diagram

4.3 DATA FLOW DIAGRAM

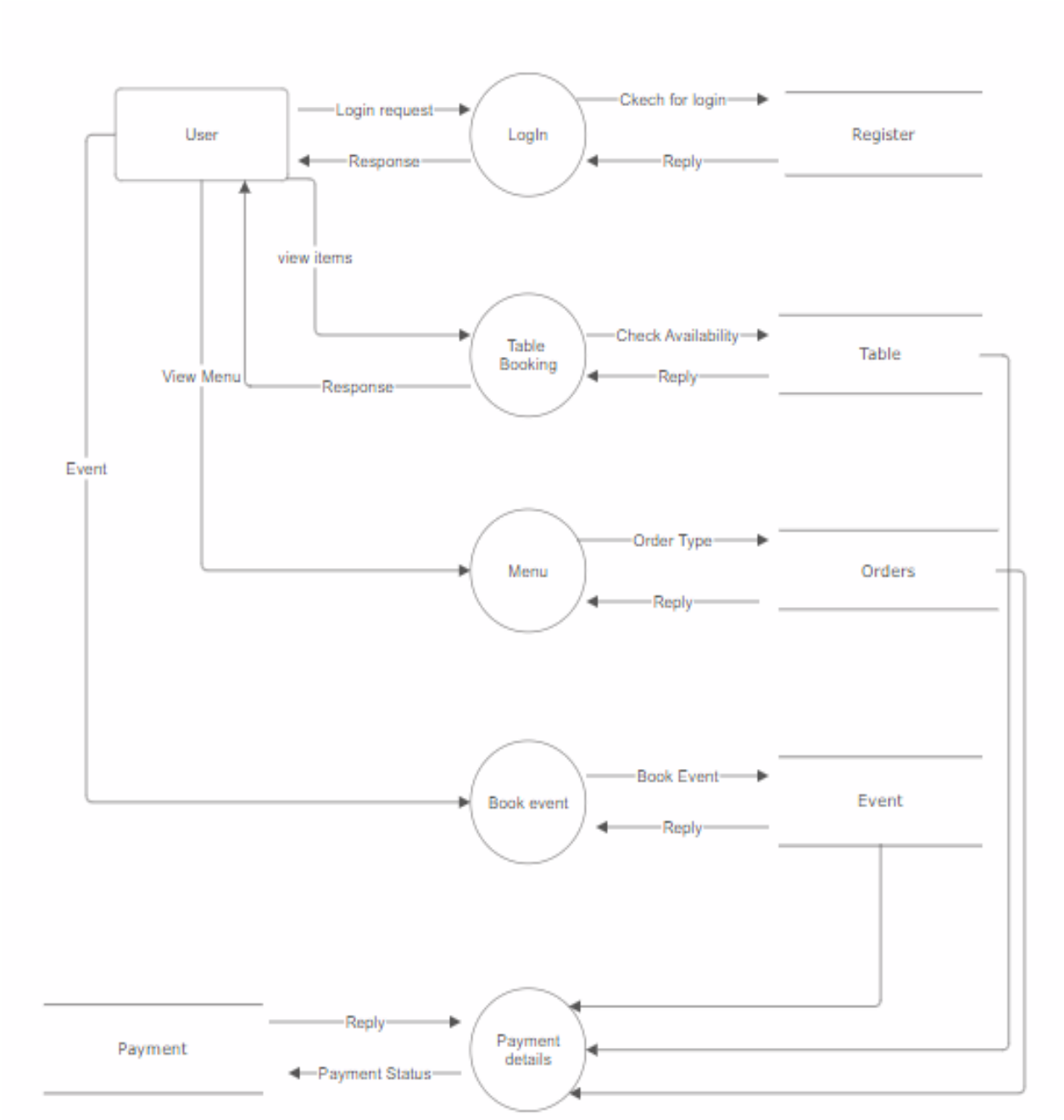

Figure 4.2 Data Flow Diagram

4.4 SYSTEM FLOW DIAGRAM

Figure 4.3 System Flow Diagram

4.5 TABLE DESIGN

Table 4.1 User Table

Field Name	Data Type	Length
User_Name	string	20
Email	symbol	25
Password	symbol	20

Table 4.2 Product Table

Field Name	Data Type	Length
Item_Name	symbol	50
Price	integer	3
Availability	string	15
Category	string	25
Image	symbol	300

Table 4.3 Order Table

Field Name	Data Type	Length
Product_Name	symbol	50
Quantity	integer	25
Product_Price	integer	3
Order_Date	date	-
Price	integer	4

4.6 INPUT DESIGN

Input design plays a crucial role in enhancing the user experience, ensuring that customers can seamlessly interact with the restaurant's website to browse available tables, make reservations, and complete payments.

It simplifies the process of inputting information such as name, email, reservation details, and payment options.

4.6.1 Search Input

The search input field is prominently placed at the top of the menu page, enabling users to easily search for available food. As users type in their preferences, the search function will display available food items.

4.6.2 Account Creation and Login

For new users, there is an option to create an account. The account creation form includes fields for entering basic information such as name, email address, and password. After creating an account, users can log in with their credentials to manage reservations or update their personal information easily.

4.6.3 Reservation and Payment Form

During the reservation process, users will need to provide their booking details, such as the date, time, number of guests, and any special requests. The reservation form collects essential information like the name, phone number, and email address to confirm the booking.

Users can then proceed to the payment section, where they can choose from various payment methods like credit or debit cards, UPI payments, or opt for "Pay at Restaurant." This process ensures a smooth and convenient booking and payment experience.

The input design ensures that users can efficiently reserve tables, track bookings, and complete payments, enhancing the overall user experience for the restaurant’s online platform.

4.7 OUTPUT DESIGN

The output design outlines how information is displayed and structured within the restaurant's online system. This includes menu listings, search results, dish details, the order process, reservation management, and administrative reports.

4.7.1 Menu Listings

A visually organized layout that presents various categories of dishes (e.g., appetizers, main courses, desserts, beverages). This page allows users to browse the restaurant’s offerings.

4.7.2 Food Cart

A summary page showing all the dishes that have been added to the user's order. It allows users to review, modify, and finalize their selection before placing an order.

4.7.3 Checkout Process

The final step before placing the order, where users review their order details and input necessary information for payment and delivery.

4.7.4 Order Confirmation

A page or email confirmation sent to the user after a successful order placement.

4.7.5 Order Report

A report generated by the admin that displays the total number of dishes sold, order statuses, and customer feedback. The report can be filtered by date or specific dish/category.

CHAPTER 5

SYSTEM TESTING

5.1 VALIDATION TESTING

Validation testing for a mobile shop typically involves ensuring that the website or application functions correctly across various devices and browsers that the purchasing process works smoothly and that users can easily find and select products. This includes Email validation, Password validation, Admin validation, and User validation. When these fields are filled correctly with the given constraints then only it navigates to other pages or modules.

Positive Test Case

Module	: Register
Test Type	: Register new user
Input	: Username with alphabets
Expected Output	: Registered Successfully

Sample Test

Input	: Krithik
Output	: Registered Successfully
Analysis	: If the output is as excepted, it navigates to the Next page(Home page).

Negative Test Case

Module	: Register
Test Type	: Register new user
Input	: Username with number (e.g., "Mouli11")
Expected Output	: Username only contains letters and spaces.

Sample Test

Input	: mouli01
Output	: Username only contains letters and spaces.
Analysis	: The system should reject the input and prompt the user to correct Analysis it.

5.2 UNIT TESTING

This involves verifying individual components or units of the software to ensure they function correctly. Each unit, such as functions or modules, is tested in isolation to validate its behaviour. By conducting through unit testing, we can identify and fix any issues early in the development process, ensuring the overall reliability and stability of our mobile shop management website.

Positive Test Case

Module	: Admin Login
Test Type	: Loading of the appropriate form for the admin dashboard.
Input	: Email and Password
Expected Output	: Display admin dashboard

Sample Test

Input	: krithik@gmail.com and krithik07.
Output	: Redirect to main page and displays the admin dashboard while

Analysis : In this form, email and password of the admin are Tested.

Negative Test Case

Module : Admin Login

Test Type : Loading of the appropriate form for the admin dashboard.

Input : Email and Wrong Password

Expected Output : Display error message of password is incorrect.

Sample Test

Input : krithik@gmail.com and Krithik.

Output : Display error message of password is incorrect.

Analysis : In this form, email and password of the admin are Tested.

5.3 INTEGRATION TESTING

Integration testing is a sort of testing that focuses on identifying problems caused by a number of different components. Integration testing ensures that key design aspects fulfil functional, performance, and reliability requirements.

Positive Test Case

Module : User

Test Type : Cart Management

Input : Add a product to the cart

Excepted Output : Cart item added.

Sample Test

Input : When selecting a product (e.g., "Product A") and clicking the "Add to Cart" button, the system should confirm that the product has been added.

Output : Cart item added

Analysis	: A success message is displayed, and the cart icon updates to reflect the new item count, confirming that "Product A" is now in the cart

Negative Test Case

Module	: User
Test Type	: Cart Management
Input	: Attempt to add a product to the cart when the product is out of stock.
Excepted Output	: The add to cart button want to get disable

Sample Test

Input	: When attempting to add an out-of-stock product (e.g., "Product B") to the cart Output
Output	: The add to cart button want to get disable.
Analysis	: The system prevents the addition of "Product B" to the cart and the add to cart button want to get disable, ensuring users are informed about stock limitations

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 IMPLEMENTATION SUPPORT

This section describes the support software, materials, equipment, and facilities required for the implementation, as well as the personnel requirements and training necessary for the implementation. The information provided in this section is not software-specific.

If there are additional support requirements not covered by the subsequent sections, others may be added as needed.

6.1.1 HARDWARE, SOFTWARE AND FACILITIES

This section lists the support software, equipment, and facilities required for the implementation.

HARDWARE

This section provides a list of support equipment and includes all hardware used for testing time implementation. The hardware such as processor Intel core 5, 16-GB RAM,15HD display, 4.7 GB hard disk, standard 103 keys keyboard, Standard Scroll Mouse is used.

SOFTWARE

This section provides a list of software and databases require supporting the implementation. This project can be implemented by software such as Visual Sudio. the Visual Studio is used to develop the project by using REACT.JS.

FACILITIES

To implement the project in the company, they must equip a computer system with essential components, including an up-to-date operating system, a keyboard, and a mouse. Additionally, a printer should be provided for report generation, along with a Wi-Fi connection to ensure seamless modern network access.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

Finally, for the pre-ordering system, we created a secure, user friendly food ordering administration system. Whether they are Admin or Customers, this system can look after them all. This system will let them manage client meals, delivery data, and expand without creating any disruption. This system is completely secure because each user is assigned a unique user ID and password, preventing unauthorised access. With online payment, registration, and cancellation, it's a lot easier to use. As a result, using this strategy will help to minimise labour expenses while also giving clients more opportunity to enjoy the services.

7.2 FUTURE ENHANCEMENT

i. The work that will be implemented with future editions of the software is described in the following section.

ii. Allow customers to modify orders: Allow clients to customise their food orders.

iii. Improve the user interface by include more interactive features for the user. Add information about deals and promotional offers on the home page. Add a week/ worth days of recipes to the home page.

iv. Payment Options: PayPal, cash, and gift cards are just a few of the options available. Allows you to save payment details for future use.

APPENDIX 1

SAMPLE CODING

Login.jsx

```
import React, { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import axios from 'axios';
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import { FaEye, FaEyeSlash } from 'react-icons/fa'; // Import eye icons
import './App.css';
function Login() {
const [email, setEmail] = useState('');
const [password, setPassword] = useState('');
const [showPassword, setShowPassword] = useState(false); // State for password visibility
const navigate = useNavigate();
const handleSubmit = async (e) => {
e.preventDefault();
try {
const response = await axios.post('http://localhost:3001/api/login', { email, password });
const { message, user } = response.data;
if (message === 'success') {
localStorage.setItem('user', JSON.stringify(user));
localStorage.setItem('isLoggedIn', 'true');
navigate(user.role === 'admin' ? '/app' : '/app');
} else {
toast.error(message || 'An unexpected error occurred.');
}
} catch (error) {
console.error('Error during login:', error);
```

```
toast.error('An error occurred. Please try again.');
}
};
return (
<div className='log-bg'>
<div className='login-container'>
<div className='login-form'>
<h2>Login</h2>
<form onSubmit={handleSubmit}>
<div className='mb-3'>
<label htmlFor='email'>
<strong>Email</strong>
</label>
<input
type='email'
placeholder='Enter Email'
autoComplete='on'
name='email'
className='inputbox'
onChange={(e) => setEmail(e.target.value)}
value={email}
/>
</div>
<div className='mb-3 password-container'>
<label htmlFor='password'>
<strong>Password</strong>
</label>
<input
type={showPassword ? 'text' : 'password'} // Toggle input type based on showPassword state
placeholder='Enter Password'
```

```
autoComplete='on'
name='password'
className='inputbox'
onChange={(e) => setPassword(e.target.value)}
value={password}
/>
<span
className='eye-icon'
onClick={() => setShowPassword(!showPassword)} // Toggle password visibility
style={{ cursor: 'pointer' }}
>
{showPassword ? <FaEyeSlash /> : <FaEye />}
</span>
</div>
<button type='submit' className='btn btn-success m'>
Login
</button>
</form>
<p className='links'>
Don't Have an account?{' '}
<Link to='/register'>
Register
</Link>
</p>
</div>
<ToastContainer />
</div>
</div>
);
}
```

```
export default Login;
```

Home.jsx

```
import React, { useState, useEffect } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import rooster from './assets/rooster.png';
import halal from './assets/halal.png';
import './App.css';

function Header() {
const navigate = useNavigate();
const [user, setUser] = useState(null);

useEffect(() => {
// Retrieve user data (including role) from localStorage
const loggedInUser = JSON.parse(localStorage.getItem('user'));
setUser(loggedInUser);
}, []);

const handleLogout = () => {
localStorage.removeItem('user');
setUser(null);
navigate('/login'); // Redirect to login page
};

const handleUsernameClick = () => {
if (user && user.role === 'admin') {
```

```
navigate('/admin'); // Navigate to admin page if user is admin
} else {
navigate('/allorders'); // Navigate to order page for other users
}
};

const handleEventClick = (e) => {
if (!user) {
e.preventDefault(); // Prevent the navigation
alert('Please log in to view the event.');
}
};

const handleMenuClick = (e) => {
if (!user) {
e.preventDefault(); // Prevent the navigation
alert('Please log in to view the menu.');
}
};

return (
<div className='header'>
<img src={rooster} width={100} height={100} alt='Rooster Logo' />
<div className='hc'>
<h1><b>Rooster Restaurant</b></h1>
</div>
<nav>
<ul className='headerheading'>
<li>
<Link to='/event' className='link' onClick={handleEventClick}>Event</Link>
```

```
</li>
<li>
<Link to='/menu' className='link' onClick={handleMenuClick}>Menu</Link>
</li>
<li>
<Link to='/about' className='link'>About</Link>
</li>
<li>
<Link to='/contact' className='link'>Contact</Link>
</li>
{!user ? (
<>
<button className='login-btn'><Link to='/login'>LogIn</Link></button>
<button className='register-btn'><Link to='/register'>Register</Link></button>
</>
) : (
<>
{/* Display username as a button */}
<button className='username-btn' onClick={handleUsernameClick}>
{user.username}
</button>
<button className='logout-btn' onClick={handleLogout}>Logout</button>
</>
)}
</ul>
</nav>
<div className='logo img'>
<img src={halal} width={80} height={80} alt='Halal Logo' />
</div>
</div>
```

```
);
}

export default Header;
```

APPENDIX 2

SCREEN SHOTS

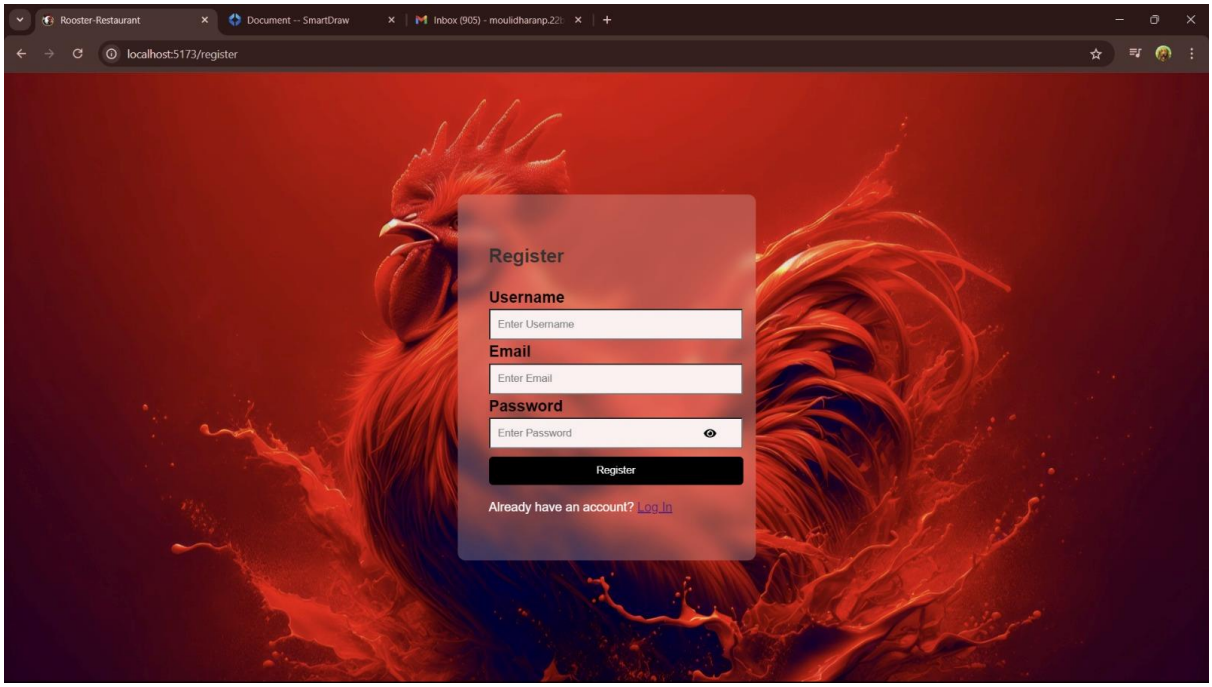

Figure A.2.1 Register Page

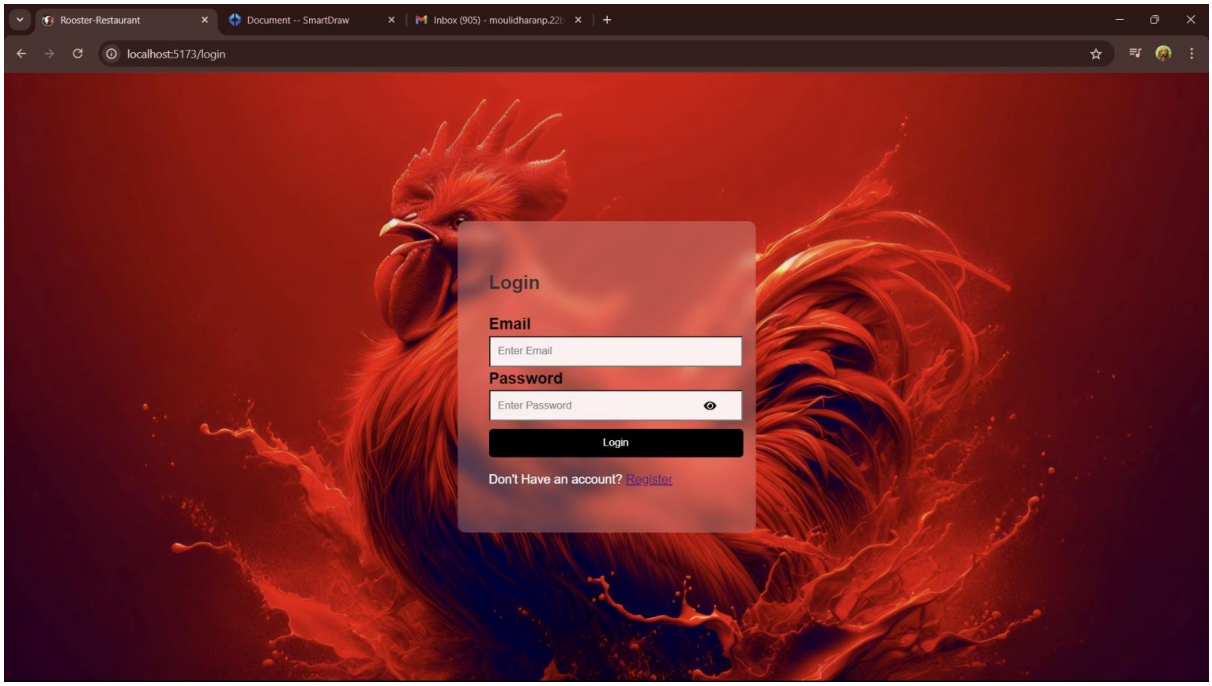

Figure A.2.2 Login Page

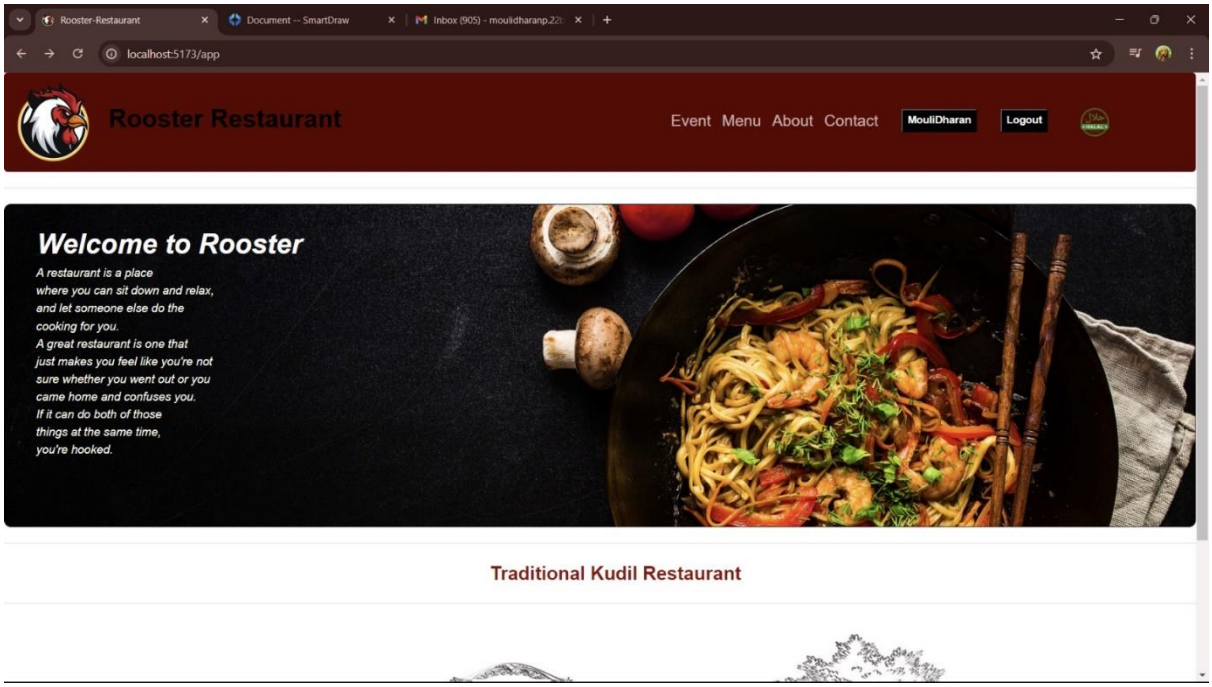

Figure A.2.3 Home Page

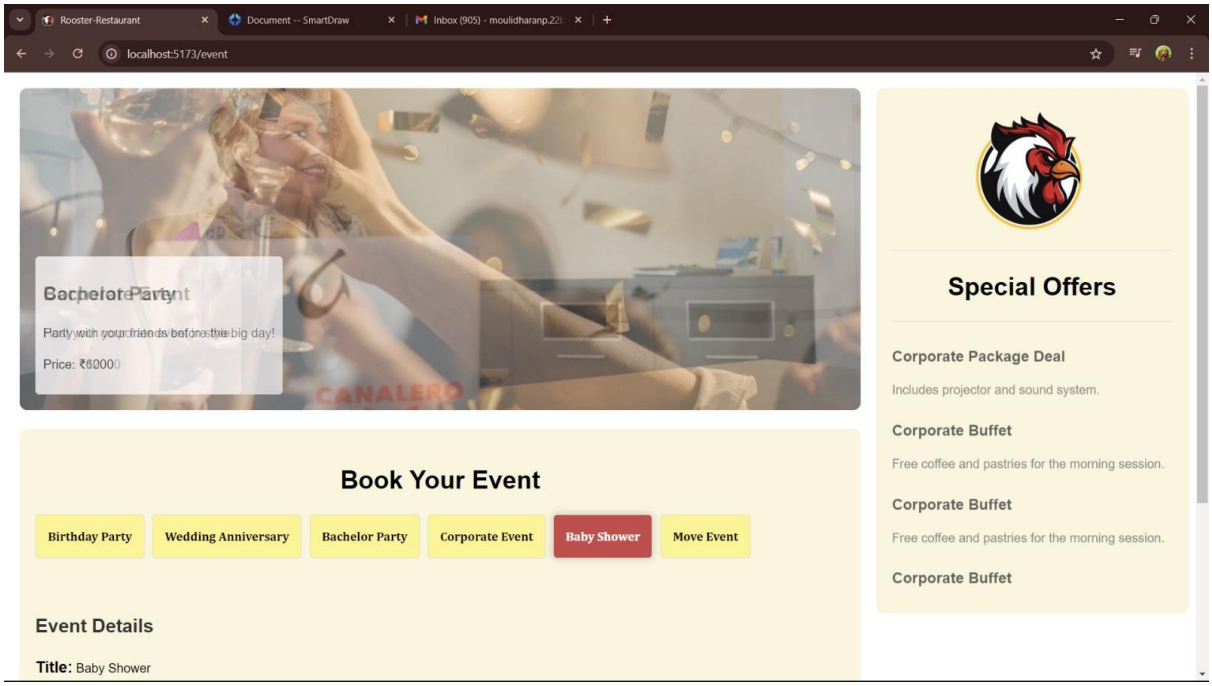

Figure A.2.4 Event Page

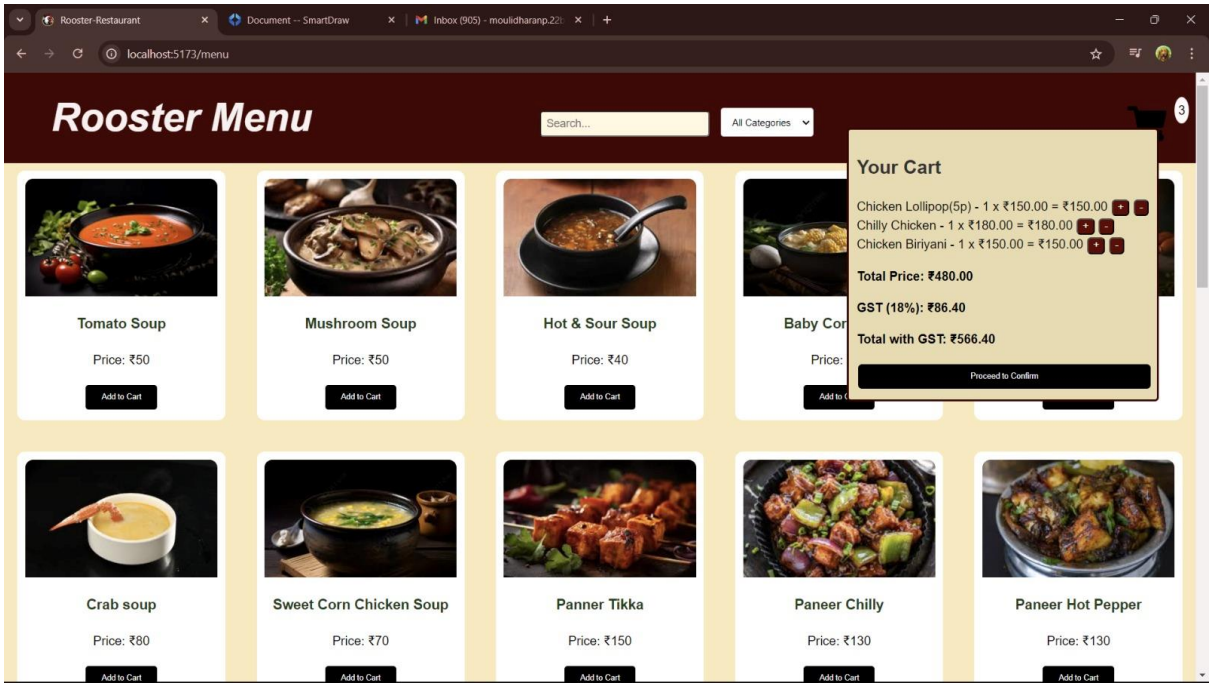

Figure A.2.5 Menu Page

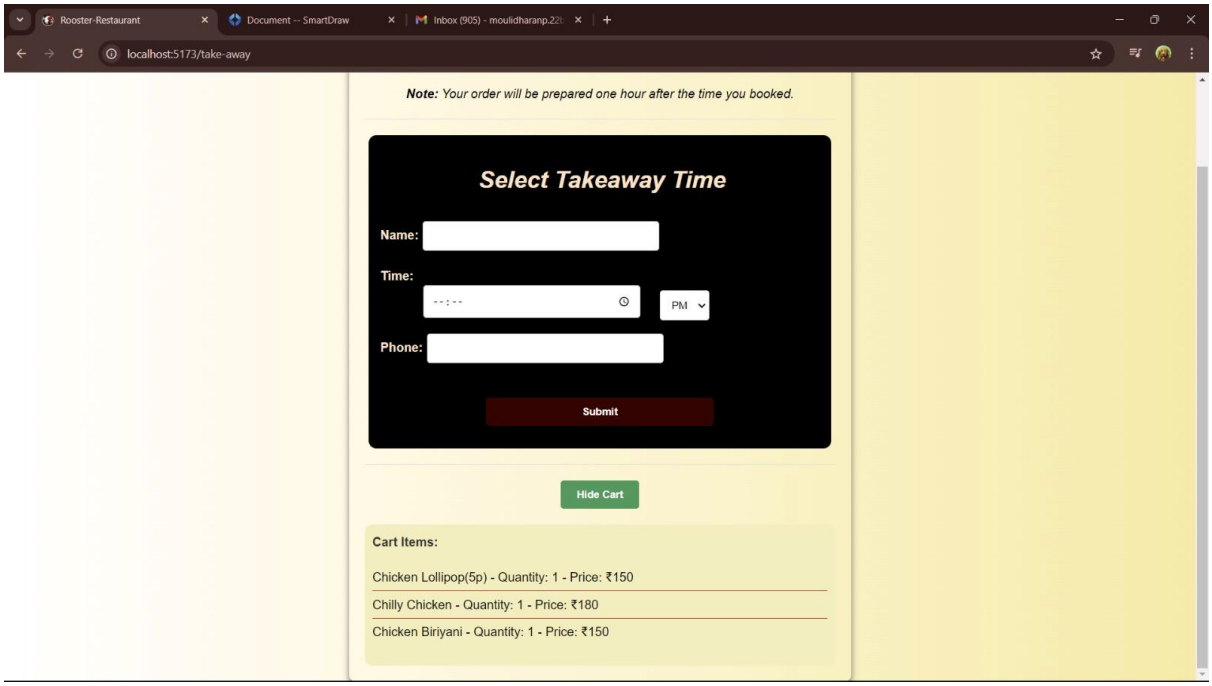

Figure A.2.6 Takeaway Order Page

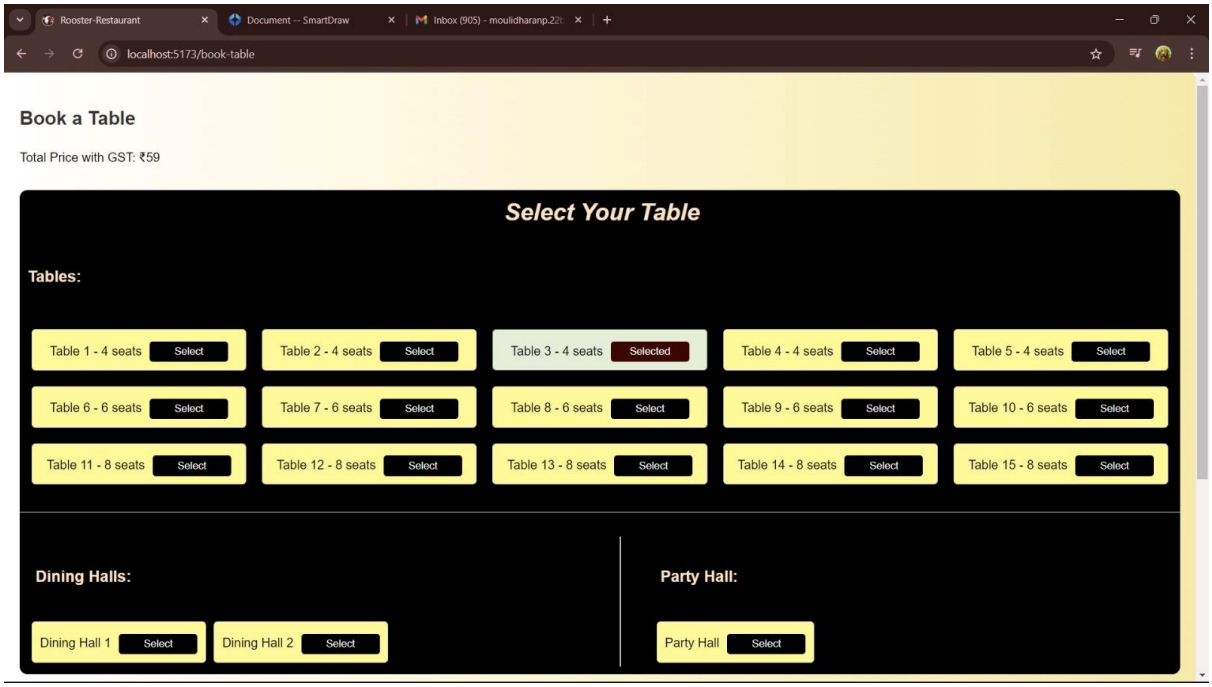

Figure A.2.7 Table Reservation

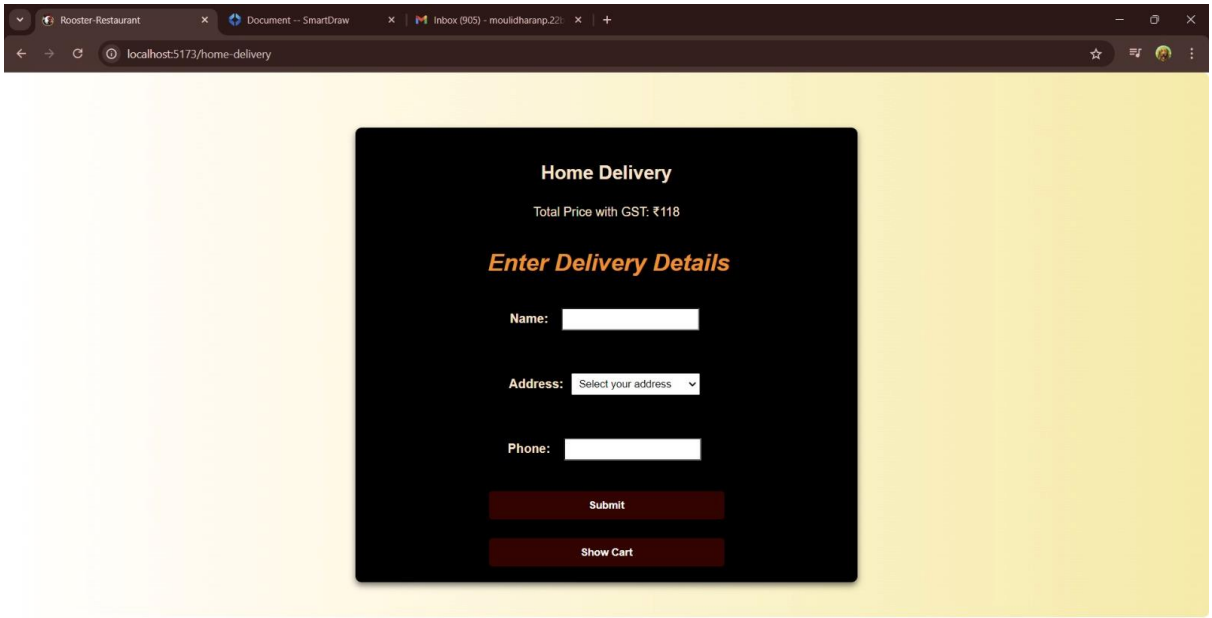

Figure A.2.8 Home Delivery

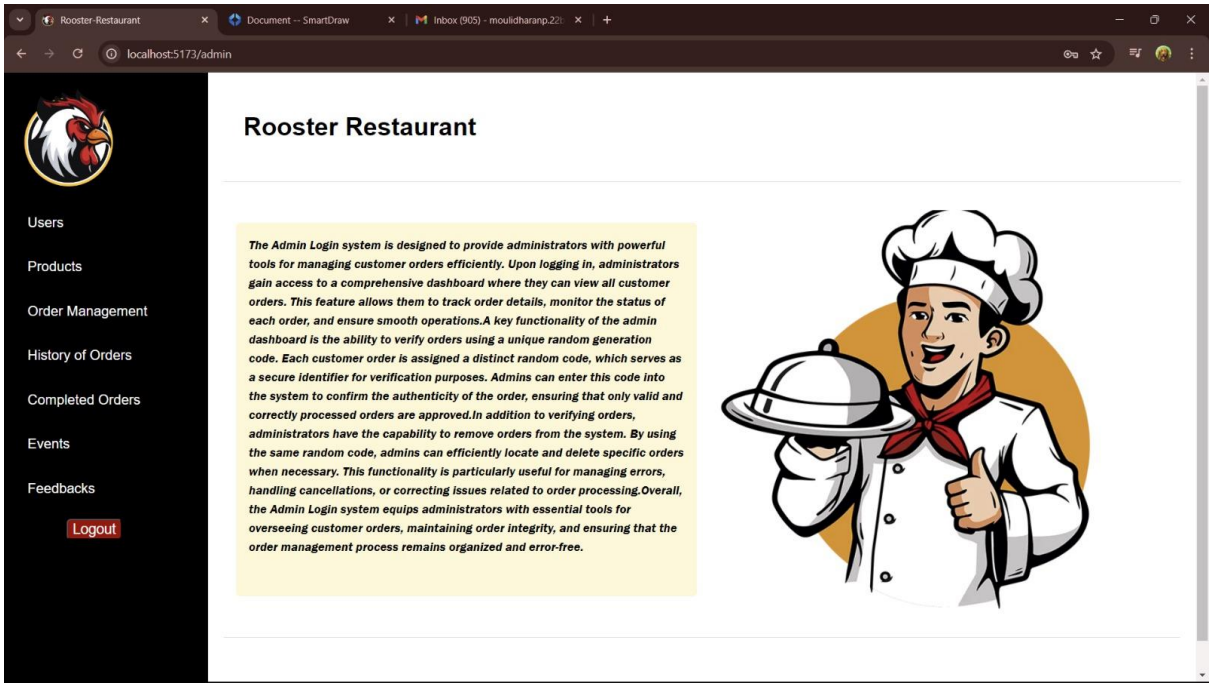

Figure A.2.9 Admin Dashboard

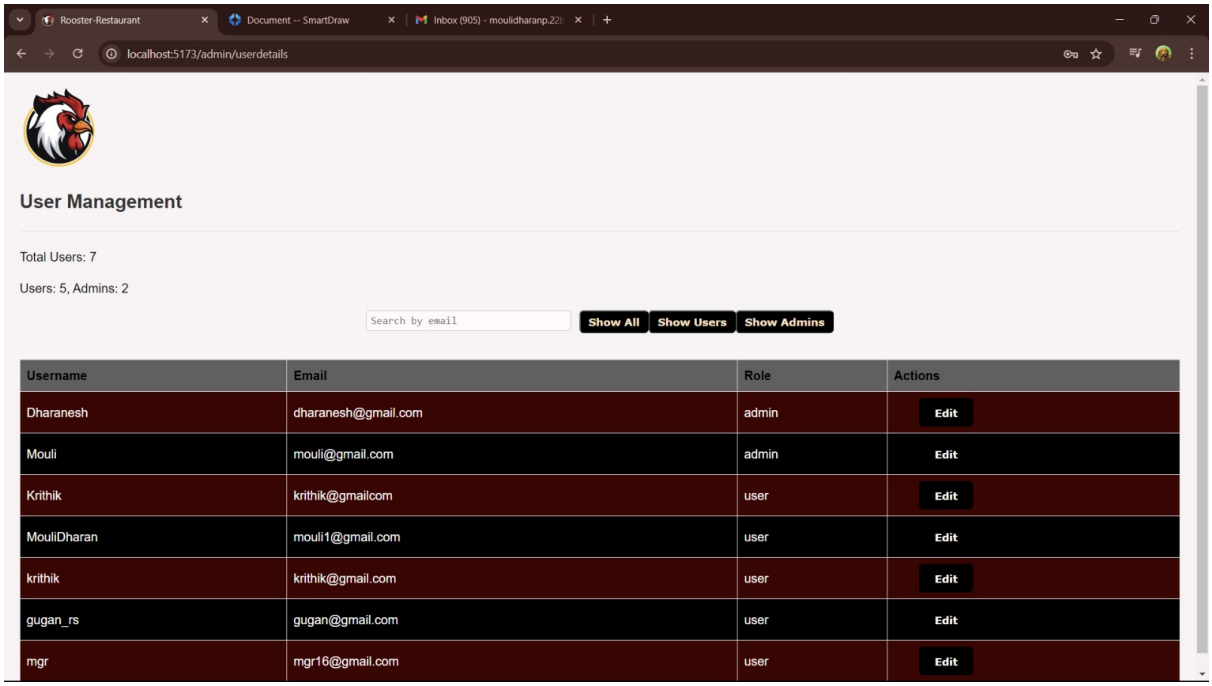

Figure A.2.10 User Dashboard

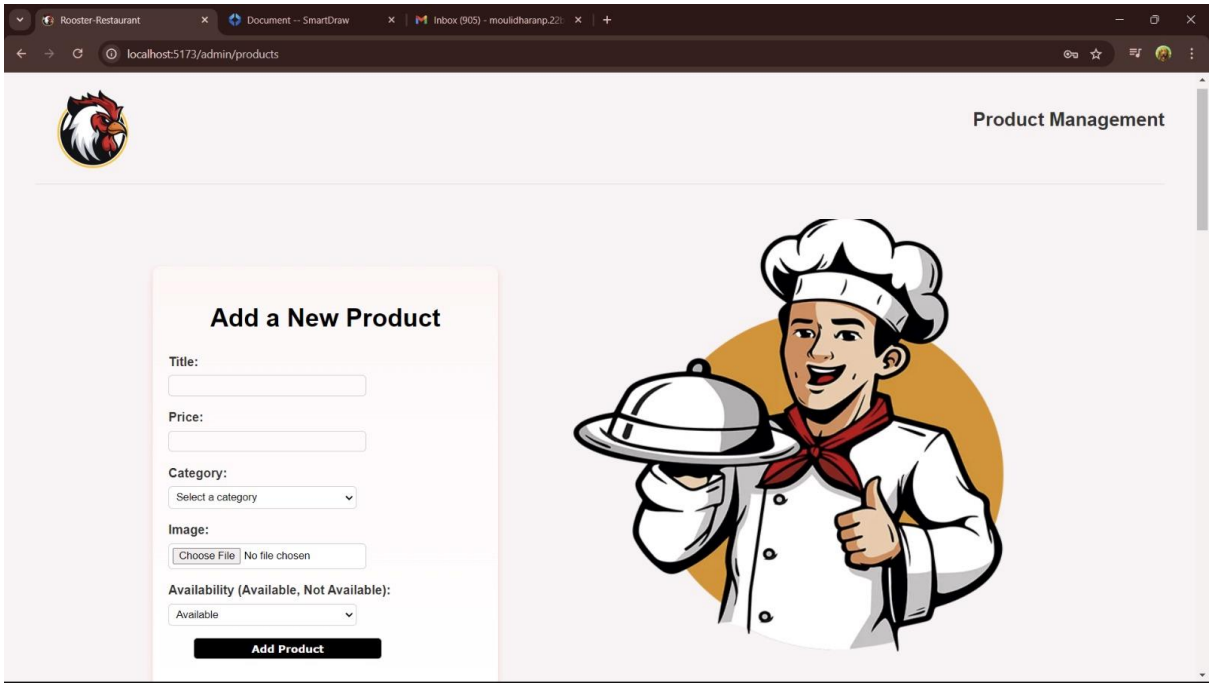

Figure A.2.11 Product Management

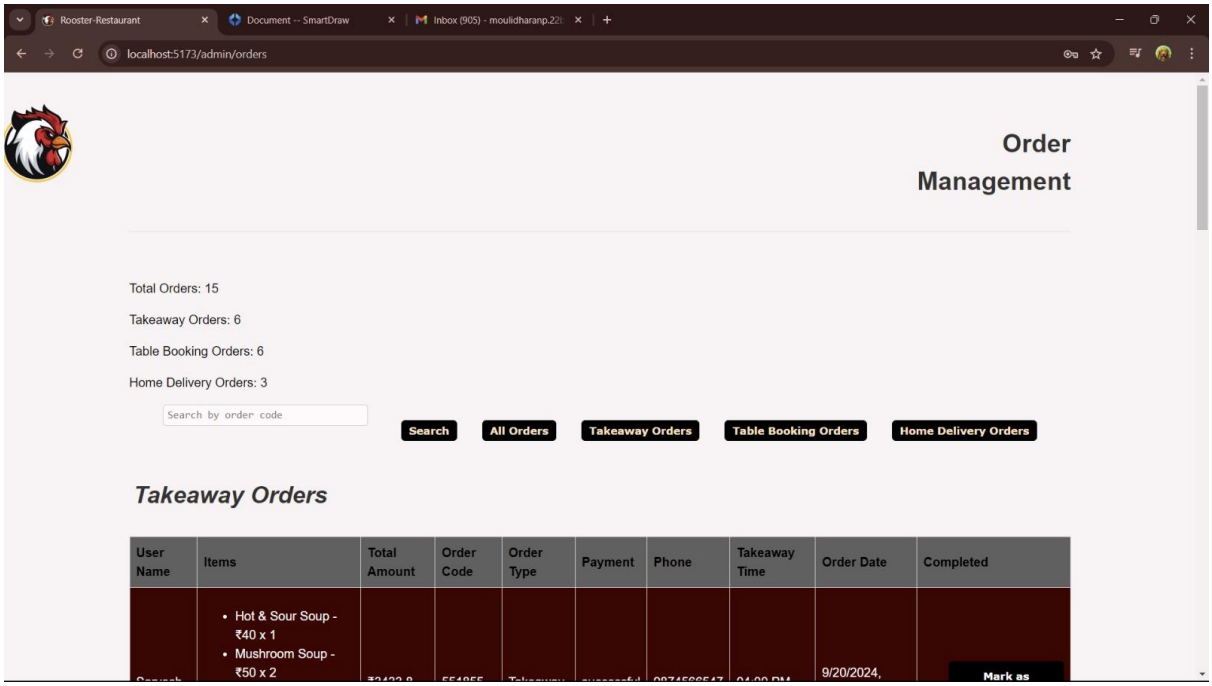

Figure A.2.12 Order Management

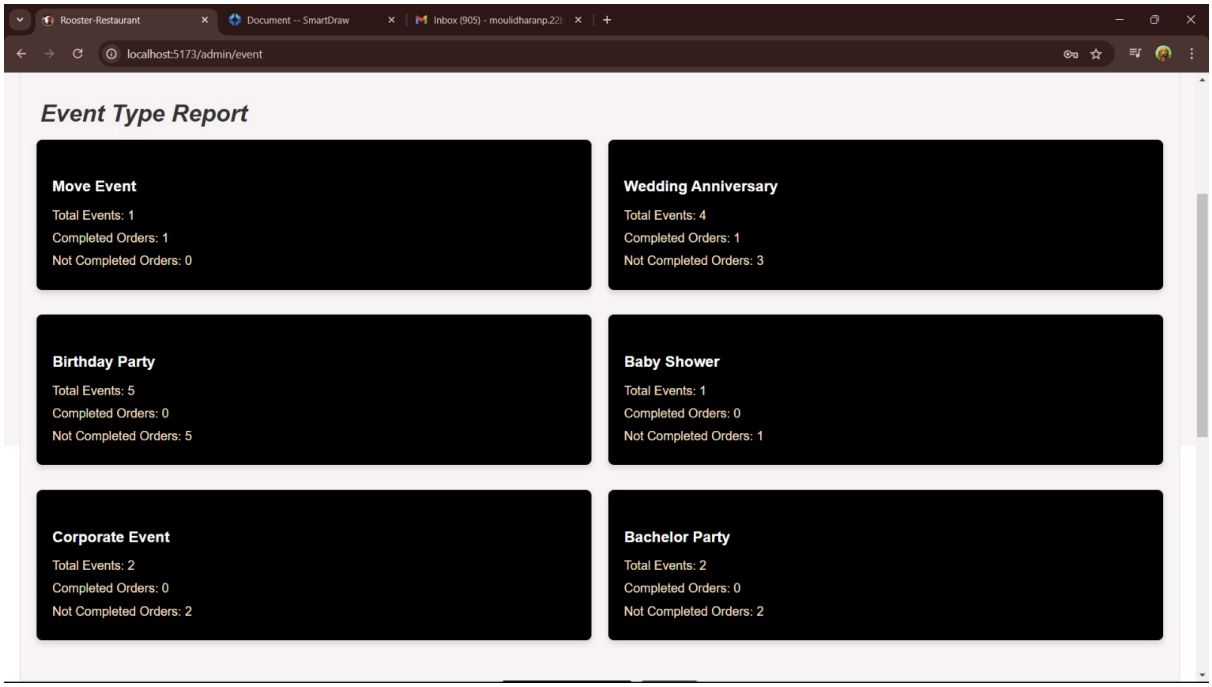

Figure A.2.13 Event Management

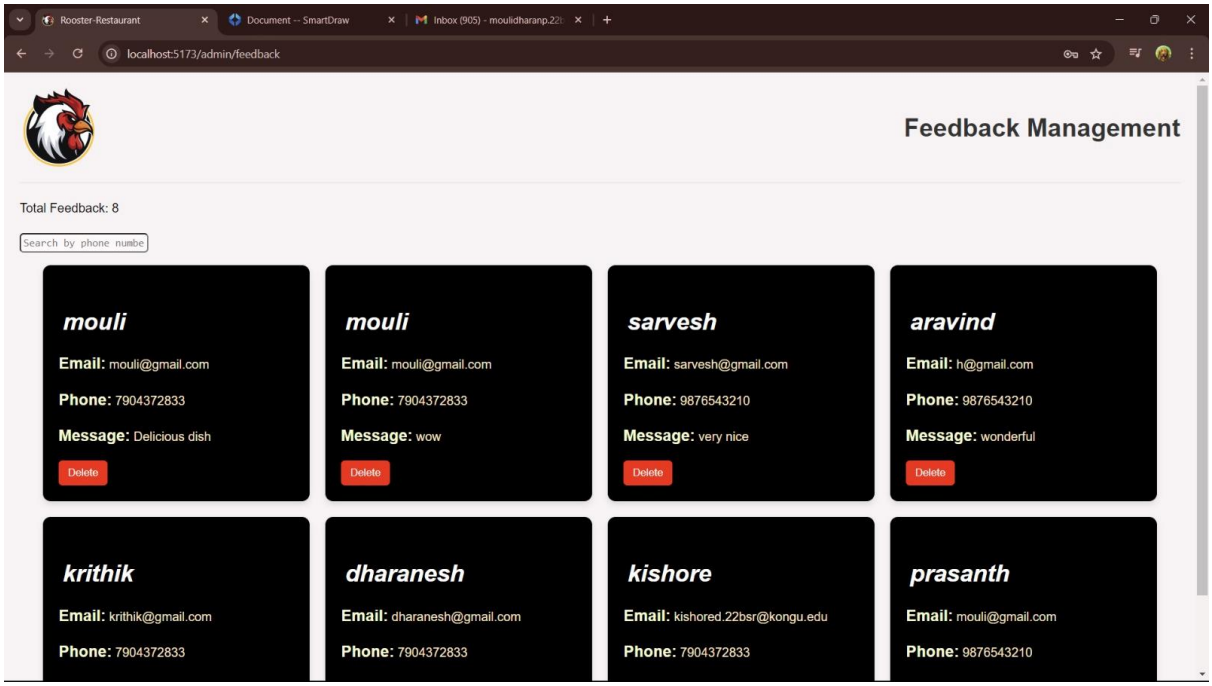

Figure A.2.14 Feedback

REFERENCES

[1] MongoDB The Definitive Guide by Kristina Chodorow and Shannon Bradshaw Published in 2010

[2] Learning MongoDB by https://www.mongodb.com/developer/books/

[3] React Up and Running Building Web Applications by Stoyan Stefanov

[4] Learning React Modern Patterns for Developing React Apps by Alex

[5] React.js Essentials by Artemij Fedosejev Published in 2015

[6] MongoDB in Action by Kyle Banker published on December 2011

[7] www.stackoverflow.com

[8] https://www.geeksforgeeks.org/

ROOSTER RESTAURANT

Dated: 01.10.2024

To

The Head of the Department

Computer Technology Department – UG

Kongu Engineering college - Autonomous

Perundurai, Erode – 638060

Dear Sir/Madam,

Subject: Project completion -Reg

We thank the following students of Kongu Engineering College in Department of Computer Technology-UG for making a website to manage reservations, sales, and inventory of our restaurant. The project is successfully completed and implemented in our working environment. **S Dhanush** (22BSR006), **M S Krithik** (22BSR028), **P Moulidharan** (22BSR032), and Project Guide **: Dr S Poorani.**

Thanking You

Yours faithfully

Rooster Restaurant

Office : +91 6381813434

Mobile : +91 9952365722

E-Mail : roosterrestaurant2023@gmail.com

GSTIN : 33ABRPE4105A1ZD

M Sivaganthi

Proprietor

ROOSTER RESTAURANT

Dated: 05.07.2024

To

The Head of the Department

Computer Technology Department – UG

Kongu Engineering college - Autonomous

Perundurai, Erode – 638060

Dear Sir/Madam,

Subject: Website Design – regarding

We request the students of **Kongu Engineering College** namely **S Dhanush** (Roll Number: 22BSR006) , **M S Krithik** (Roll Number: 22BSR028) and **P Moulidharan** (Roll Number: 22BSR032) to provide the website under the guidance of **Dr.S.Poorani** (Project Guide - CT-UG Department - Kongu Engineering College) for promote and market our Restaurant Food. We agree to pay Rs.7080.00 + GST Extra (**Seven Thousand Eighty Rupees Only**) for this website design and development.

Thanking You

Yours faithfully

Rooster Restaurant

Office : +91 6381813434

Mobile : +91 9952365722

E-Mail : roosterrestaurant2023@gmail.com

GSTIN : 33ABRPE4105A1ZD

M Sivagandhi

Proprietor