Faculty of Science and Engineering

Referred Coursework – 2023/24 Academic Year

Module Code: PUSL3120

Module Title: Full Stack Development

Module Leader: Dr Mark Dixon

School: Engineering, Computing and Mathematics

Name: RMT RASHAN

Index number: 10819545

YouTube link: https://youtu.be/h3Lr7avQHrE

GitHub link: https://github.com/ThemiyaRajapaksha/Full-Stack

Contents

1. Requirements 3
1.1 Who is the application aimed for? 3
1.2 What are the features? 3
2. Design 4
2.1 System architecture 4
2.2 How do the components interact? 6
2.3 How are the data and code structured? 7
2.4 Why are these structures appropriate? 9
3 Testing 10
3.1 What automated testing have you performed? 10
3.2 What usability testing have you performed? 11
4. DevOps pipeline 11
4.1 Development environment 11
4.2 Continuous integration pipeline and how used it 12
5. Personal reflection 13
5.1 What worked/ didn't work well? 13
Appendix 14
Reference 19

1. Requirements

1.1 Who is the application aimed for?

My innovative web is designed to simplify the process of ordering food from the favorite restaurants. With a clean and straightforward interface, it allows users to browse through various menu options and place orders effortlessly. Our goal is to provide a seamless online ordering experience. The app's search feature helps you quickly locate the dishes or cuisines you're craving, making the selection process easy and enjoyable. Once the chosen meal, ordering is a breeze. A few simple steps let you customize the order, proceed with payment, and confirm the purchase—all without needing to leave the web or contact the restaurant directly. Our platform offers a modern, hassle-free way to enjoy restaurant meals delivered straight to the door. In essence, our food ordering system is all about convenience. With a user-friendly design, efficient search tools, and a smooth ordering process, it's the perfect solution for anyone looking to enjoy quality restaurant food from home

1.2 What are the features?

Customer Features

a) Create an account and log in.

b) Browse through the available menu items.

c) View current offers and discounts.

d) Place food orders easily.

e) Add items to the shopping cart.

f) Contact the admin for any inquiries.

Admin Features:

a) Admins can log in securely using their credentials.

b) Manage users by adding, updating, viewing, and removing them.

c) Manage promotional offers by adding, updating, viewing, and removing them.

d) Oversee the menu by adding, updating, viewing, and removing items.

e) Manage admin accounts, including adding, updating, viewing, and removing.

f) Handle customer messages by adding, updating, viewing, and deleting them.

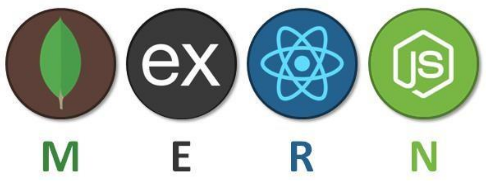
In addition to the core features, our system is designed to provide a comprehensive and seamless experience for both customers and administrators. The website boasts a responsive design, ensuring that users enjoy a consistent and intuitive interface across all devices, whether they are on a smartphone, tablet, or desktop. Real-time updates are another key feature, allowing administrators to make changes to the menu or promotional offers instantly, with these updates immediately visible to users.

Security is also a top priority, with secure payment gateways integrated into the platform, ensuring that customer transactions are protected and confidential.

Moreover, the system includes an order tracking feature, enabling customers to monitor their orders from placement through to delivery, enhancing transparency and convenience. Administrators benefit from data analytics tools that provide insights into user behavior and sales trends, facilitating data-driven decisions to optimize business operations. Additionally, automated email notifications keep users informed about important actions such as account creation, order confirmations, and status updates, improving communication and engagement. Together, these advanced features make the system a robust and efficient solution for managing restaurant takeaway operations, enhancing both user satisfaction and administrative efficiency.

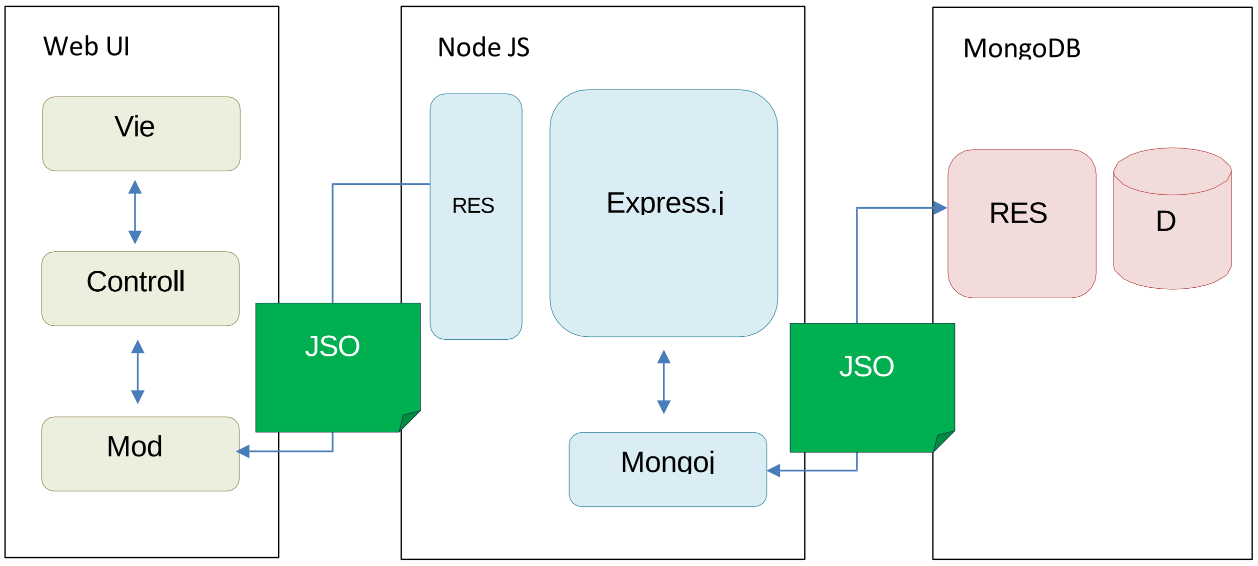
2. Design

2.1 System architecture



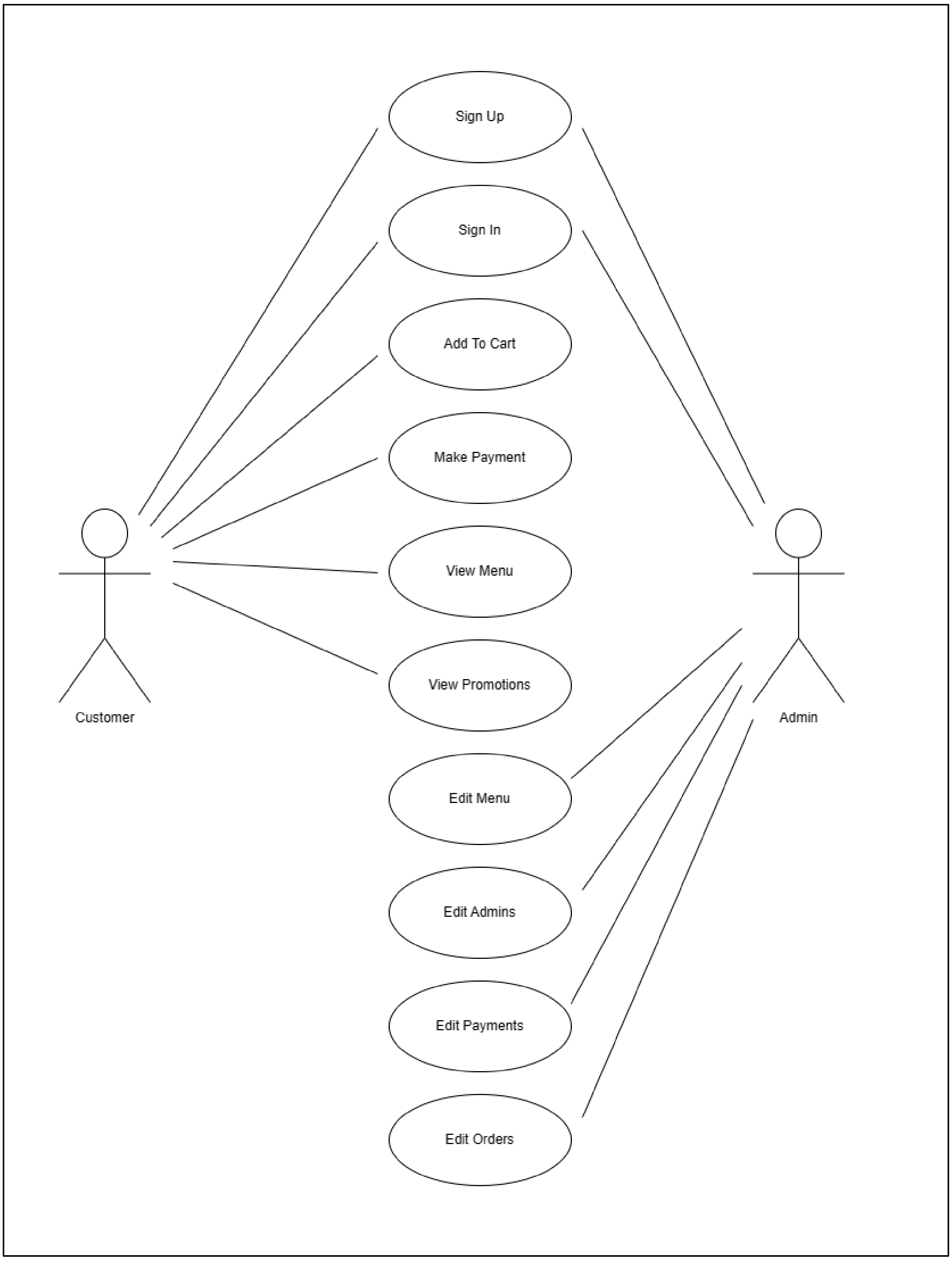
Our restaurant takeaway website was built using the MERN stack, a collection of technologies specifically designed for developing web applications. The acronym 'MERN' stands for MongoDB, Express.js, React.js, and Node.js. MongoDB serves as the database for storing data, Express.js provides a framework for creating web applications, React.js is a JavaScript library for crafting user interfaces, and Node.js acts as the runtime environment for running server-side code. These technologies work together to handle everything from data management and storage to user interface creation and server-side logic. The MERN stack is favored for its versatility, scalability, and the convenience of using JavaScript across the entire development process, enhancing overall efficiency.

This web application allows users to browse the menu, select their preferred items, add them to their cart, complete payment, and arrange for delivery. In this system, MongoDB is utilized to store customer data and other relevant information.


Web UI
Vie
Controll
Mod
JSO
Node JS
RES
Express.j
Mongoj
JSO
MongoDB
RES
D

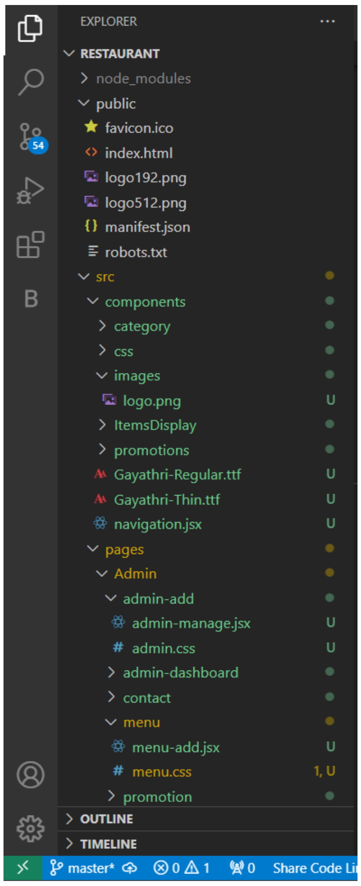
2.2 How do the components interact?

Use case diagram

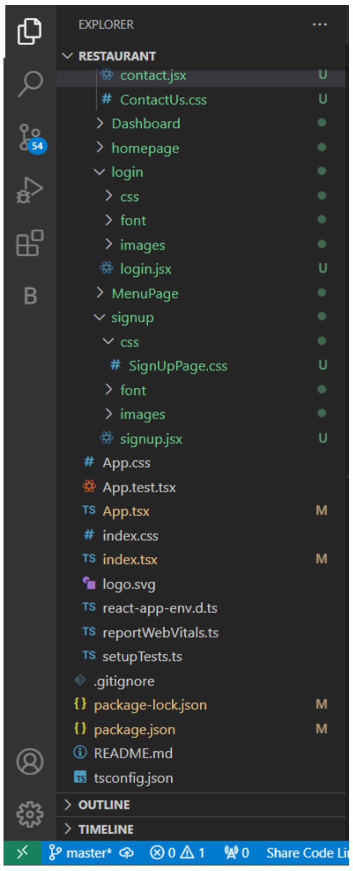


2.3 How are the data and code structured?

Front data and code structure



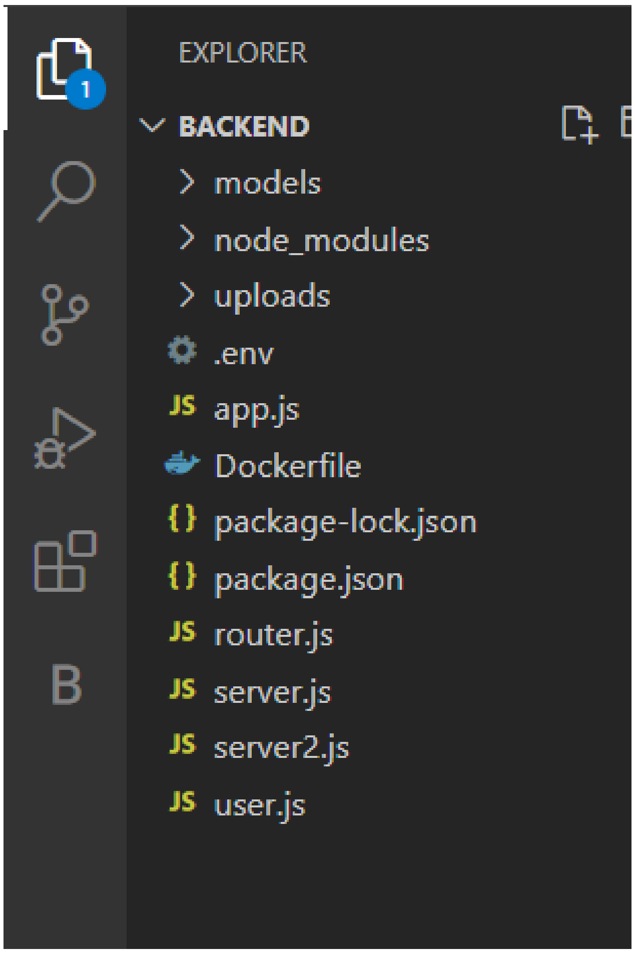
```
EXPLORER                 ...
∨ RESTAURANT
  > node_modules
  ∨ public
    favicon.ico
    index.html
    logo192.png
    logo512.png
    manifest.json
    robots.txt
  ∨ src
    ∨ components
      > category
      > css
      ∨ images
        logo.png                 U
      > ItemsDisplay
      > promotions
      Gayathri-Regular.ttf       U
      Gayathri-Thin.ttf          U
      navigation.jsx             U
    ∨ pages
      ∨ Admin
        ∨ admin-add
          admin-manage.jsx       U
          admin.css              U
        > admin-dashboard
        > contact
        ∨ menu
          menu-add.jsx           U
          menu.css            1, U
        > promotion
> OUTLINE
> TIMELINE
master*   ⊗ 0 ⚠ 1   0   Share Code Li
```



```
EXPLORER                 ...
∨ RESTAURANT
      contact.jsx                U
      ContactUs.css              U
    > Dashboard
    > homepage
    ∨ login
      > css
      > font
      > images
      login.jsx                  U
    > MenuPage
    ∨ signup
      ∨ css
        SignUpPage.css           U
      > font
      > images
      signup.jsx                 U
    App.css
    App.test.tsx
    App.tsx                      M
    index.css
    index.tsx                    M
    logo.svg
    react-app-env.d.ts
    reportWebVitals.ts
    setupTests.ts
  .gitignore
  package-lock.json              M
  package.json                   M
  README.md
  tsconfig.json
> OUTLINE
> TIMELINE
master*   ⊗ 0 ⚠ 1   0   Share Code Li
```

I organized the React code for this website with an emphasis on simplicity and scalability. The project was built using a component-based architecture, where the functionality was broken down into distinct components, each corresponding to specific views within the application. I maintained a separate directory to house all the pages, with each page having its own dedicated folder. The templates defined the HTML structure and bindings, while CSS was used to handle the styling.

Backend data and code structure



EXPLORER

BACKEND
- models
- node_modules
- uploads
- .env
- app.js
- Dockerfile
- package-lock.json
- package.json
- router.js
- server.js
- server2.js
- user.js

In developing a Node.js backend with an Express server, I adopted a structured approach to ensure that our code remains maintainable and scalable. I began by creating a models directory to store the data models utilized throughout the application. This folder includes files like Admin.js, contact.js, MenuItem.js, orderSchema.js, and promotion.js. The application's dependencies are managed through the package.json and package-lock.json files, with the installed dependencies stored in the node_modules directory. This methodical approach ensures that the code is well-organized and easy to understand, promoting ease of maintenance and scalability. Additionally, our code structure is reinforced by the use of a models folder for data models, and the careful management of dependencies through package.json files. The inclusion of a Dockerfile and a .github directory for workflows further simplifies the integration with external tools and automates the CI/CD pipeline, making deployment more efficient.

2.4 Why are these structures appropriate?

Front End

I carefully structured the React-based Restaurant takeaway website to ensure ease of maintenance, feature expansion, and testing. The project was divided into components, serving as modular building blocks, which kept the code well-organized and manageable. This modularity also enabled the reuse of components across various sections of the project, enhancing efficiency. I centralized important data and logic within services, streamlining management and maintenance tasks. For smoother navigation throughout the application, I organized the routing paths into a separate module. These deliberate design choices resulted in a codebase that is both efficient and robust, making it simpler to maintain, extend with new features, and thoroughly test the Restaurant takeaway website.

Backend

I organized the Node.js backend using an Express server to make it easy to manage and grow. I split things up by creating folders for different tasks – one for data models called "models" and another for routing logic named "routes." This separation makes it simpler to understand and change different parts of the code. I set up the server using "index.js" as the starting point and kept track of dependencies with "package- lock.json" and "package.json," saving them in the "node_modules" folder. For containerizing the web and automating the testing and deployment process, I added a "Dockerfile" and a "github workflow" folder.

Following this organized approach, I've built a clear and well-structured codebase. It's easy to understand, maintain, and expand, contributing to the application's long-term success.

3 Testing

3.1 What automated testing have you performed?

I employed Jest and Enzyme to automate testing for our front-end. Enzyme, a tool specifically designed for React, enables developers to effectively test React components by providing methods to interact with and analyze their output. This helps simulate user interactions and ensures that the components function as expected. For our backend testing, I also used Jest along with SuperTest. Jest, a popular testing framework created by Facebook, is ideal for JavaScript applications, particularly those built with React. SuperTest complements this by allowing us to test HTTP requests and responses, making it perfect for validating the behavior of web APIs. Together, these tools ensure that both our front-end and back-end are thoroughly tested and reliable.

Backend Testing

For testing the backend, I utilized Jest and SuperTest. I developed test cases to verify the functionality of our code and ensure it performs as expected. Jest provided us with built-in tools for making assertions and creating mock data, which streamlined the testing process. SuperTest, a library that integrates with Jest, was used specifically for testing our APIs. It allowed us to simulate HTTP requests to our API and check if the server responses were correct. Although I initially had limited knowledge of how to implement Jest and SuperTest, we leveraged YouTube tutorials to successfully write test cases for tasks like fetching the menu and submitting new orders.

Front-End Testing

For testing the front end, I employed Jest and Enzyme. Enzyme made it easier to test React components by enabling us to simulate user interactions, examine the rendered output, and verify that the components functioned as intended.

3.2 What usability testing have you performed?

To ensure our website met the needs of our users and delivered a satisfying experience, we conducted thorough usability testing. We began by selecting a diverse group of 10 customers to participate in this testing process, ensuring we gathered feedback from a range of perspectives. Before the testing commenced, we carefully developed a series of test cases designed to evaluate different aspects of the website's functionality, user interface, and overall user experience. We then created a controlled test environment to simulate real-world usage conditions, allowing us to observe how participants interacted with the website.

Throughout the testing phase, we gathered valuable feedback from the participants, who were asked to perform specific tasks and share their experiences. The results of the testing were promising, as the majority of customers found the web application to be user-friendly and intuitive. They appreciated the ease of navigation and the clarity of the design, which contributed to their overall satisfaction. However, during the testing, we also identified a few minor bugs that required attention. These findings underscored the importance of usability testing in refining the system. In conclusion, the testing process highlighted the website's strengths while also providing actionable insights for further improvement.

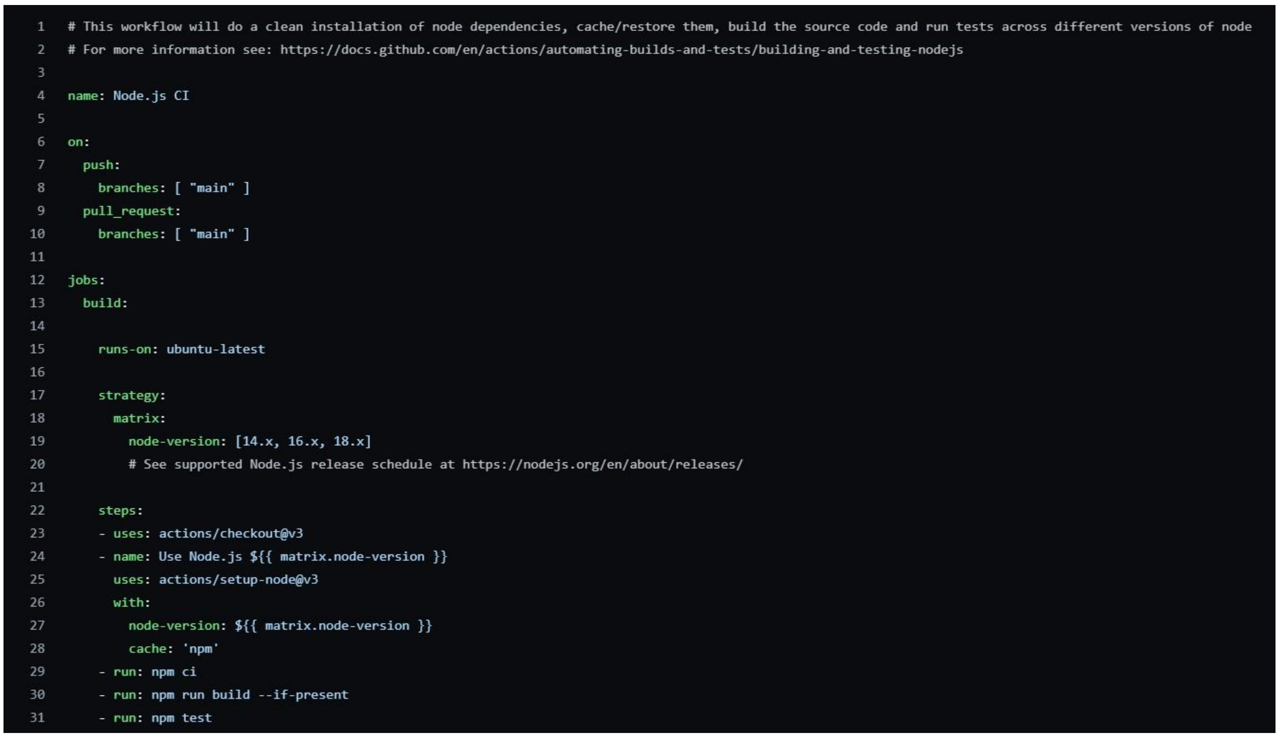
4. DevOps pipeline

4.1 Development environment

For developing this restaurant takeaway website, I utilized the MERN stack. The front end was built using React.js due to its declarative syntax, which simplifies the process of writing and understanding UI components. By defining the desired state, React handles the necessary UI updates automatically. The backend was managed by Node.js, a platform that allows developers to write server-side code in JavaScript, ensuring a consistent language throughout the application. I chose Express.js to manage HTTP requests and responses. This lightweight and flexible Node.js framework provides essential tools for building both web and mobile applications, giving developers the freedom to structure their projects effectively. Our database of choice was MongoDB, a NoSQL database that stores data in a flexible BSON (Binary JSON) format. This format is particularly useful for handling and manipulating complex data structures with ease.

4.2 Continuous integration pipeline and how used it.

CI pipeline for front end



```yaml
# This workflow will do a clean installation of node dependencies, cache/restore them, build the source code and run tests across different versions of node
# For more information see: https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-nodejs

name: Node.js CI

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest

    strategy:
      matrix:
        node-version: [14.x, 16.x, 18.x]
        # See supported Node.js release schedule at https://nodejs.org/en/about/releases/

    steps:
    - uses: actions/checkout@v3
    - name: Use Node.js ${{ matrix.node-version }}
      uses: actions/setup-node@v3
      with:
        node-version: ${{ matrix.node-version }}
        cache: 'npm'
    - run: npm ci
    - run: npm run build --if-present
    - run: npm test
```

I set up a continuous integration pipeline for our React front-end application using Git, GitHub Actions, and Docker Hub. This automated system ensures that the code is built, tested, and deployed automatically whenever updates are made. The process began by creating a Git repository to host the React front-end code, after which the initial codebase was pushed to the repository. This pipeline has greatly improved our development process, ensuring a more efficient and reliable release cycle for the front-end application.

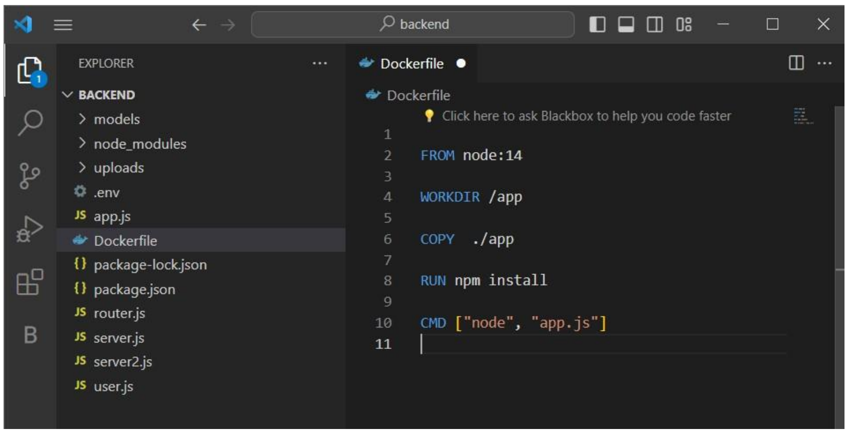
CI pipeline for back end

I

streamlined the process for maintaining and updating our Node.js application by implementing a continuous integration (CI) pipeline. Using Git, GitHub Actions, and Docker Hub, we established an automated system that runs tests and updates the code effortlessly. This approach minimizes errors and simplifies the introduction of new features or bug fixes. The process began with the creation of a Git repository to store our Node.js code, where the initial codebase was committed. Next, I configured GitHub Actions to automatically execute tests and validations whenever the code is modified, utilizing

GitHub Action workflows to manage this process. Additionally, I created a Dockerfile that outlines the steps required to build a Docker image of the Node.js backend for the website. Once the image is built, it is pushed to a Docker Hub repository for easy distribution.

Docker file



```
FROM node:14

WORKDIR /app

COPY  ./app

RUN npm install

CMD ["node", "app.js"]
```

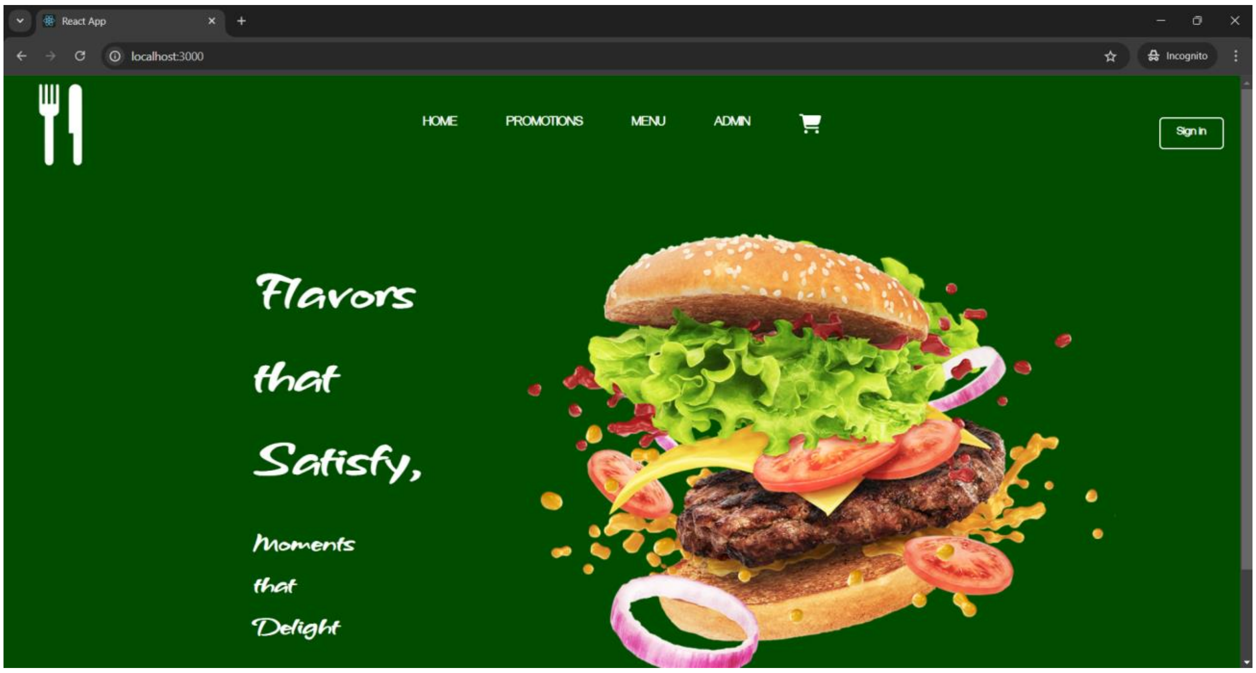
5. Personal reflection

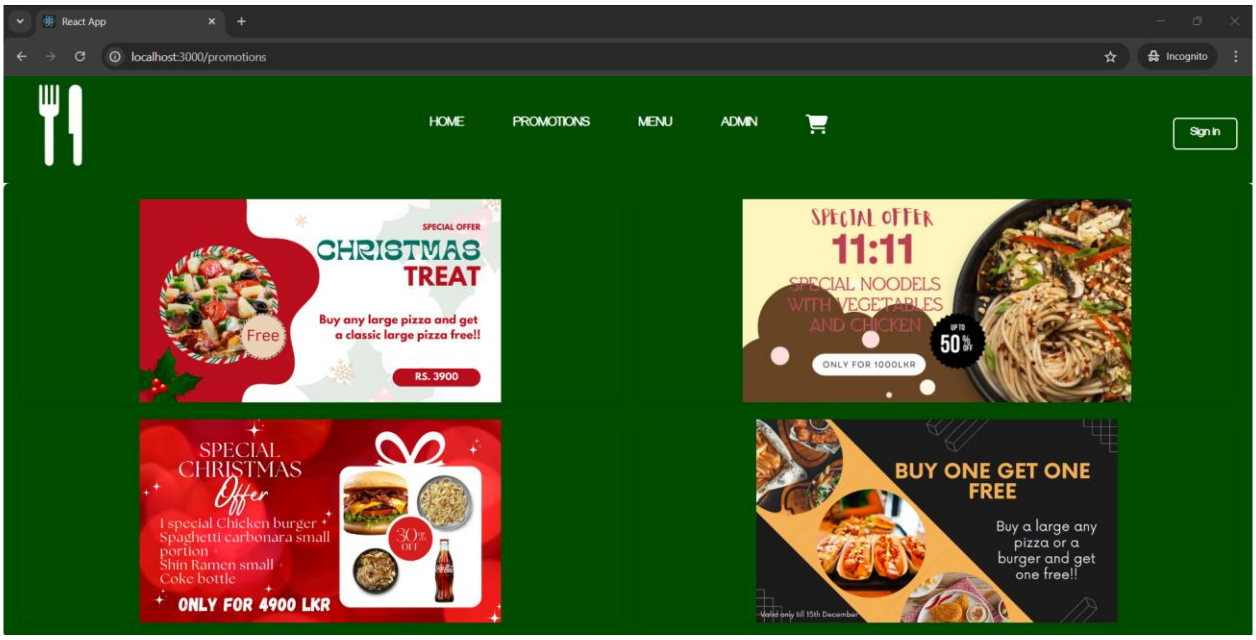
5.1 What worked/ didn’t work well?

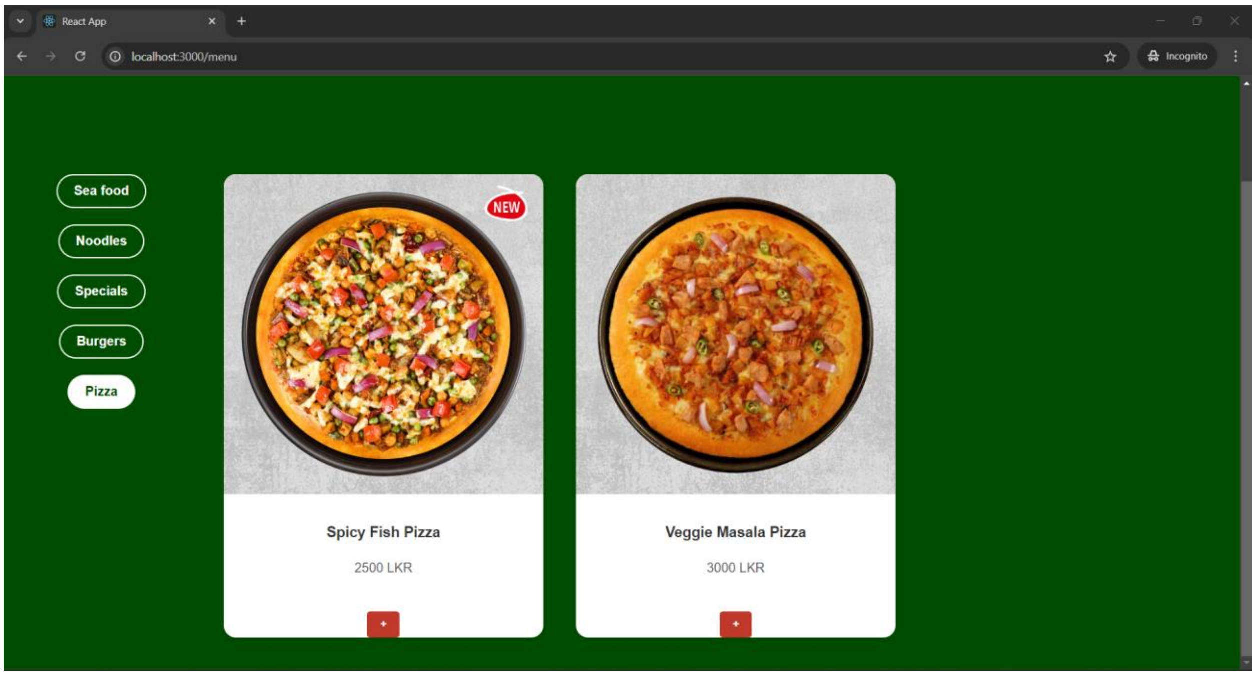
Reflecting on the project’s completion, there were both successes and challenges. One significant achievement was the decision to use the MERN stack—MongoDB, Express.js, React.js, and Node.js. The smooth integration of these technologies provided a strong foundation for the restaurant takeaway website, facilitating efficient development and resulting in a responsive, dynamic user interface.

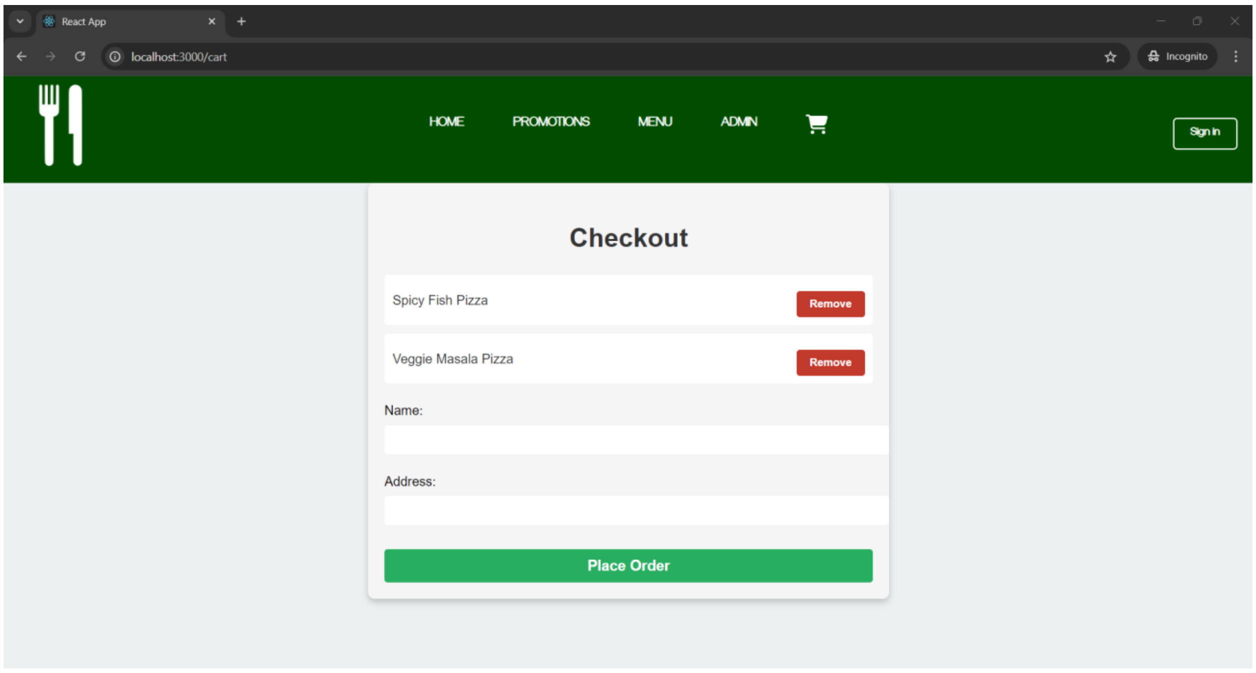
Despite these successes, I faced some challenges along the way. Implementing real-time chat functionality with Socket.io required extra effort and introduced a learning curve. Additionally, debugging and maintaining consistent performance across various environments proved challenging, underscoring the importance of thorough and comprehensive testing.

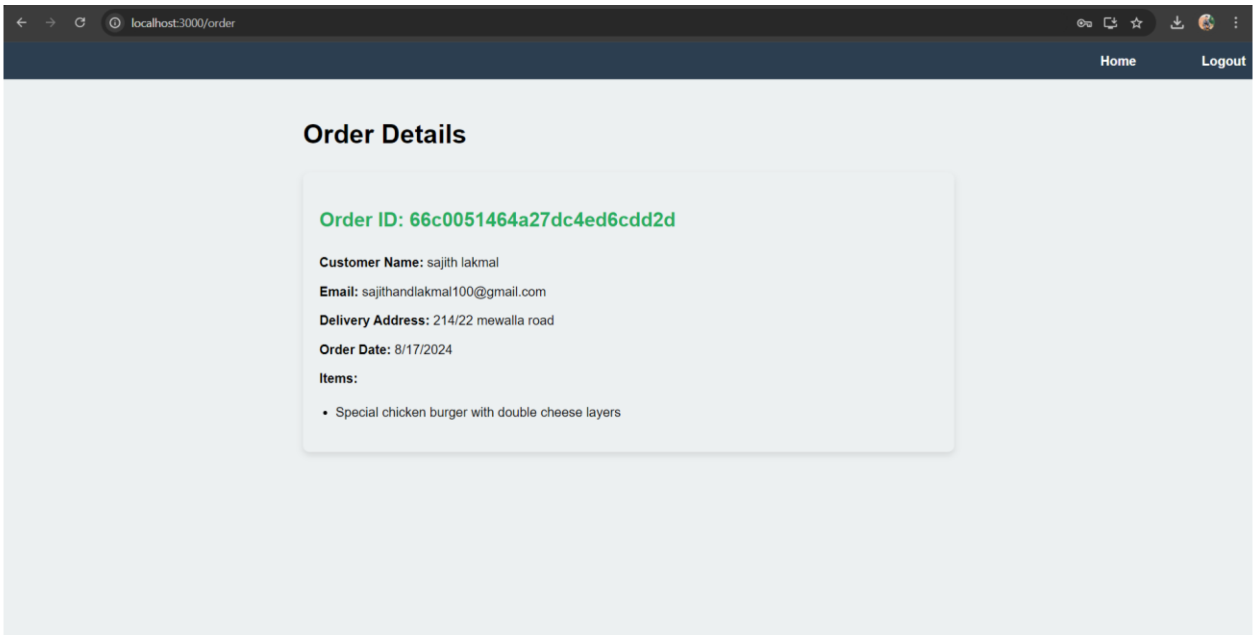
Appendix


React App
localhost:3000
Incognito
HOME
PROMOTIONS
MENU
ADMIN
Sign in
Flavors
that
Satisfy,
Moments
that
Delight


React App
localhost:3000/promotions
Incognito
HOME
PROMOTIONS
MENU
ADMIN
Sign in
SPECIAL OFFER
CHRISTMAS TREAT
Buy any large pizza and get a classic large pizza free!!
Free
RS. 3900
SPECIAL OFFER
11:11
SPECIAL NOODELS WITH VEGETABLES AND CHICKEN
UP TO 50% OFF
ONLY FOR 1000LKR
SPECIAL CHRISTMAS Offer
1 special Chicken burger
Spaghetti carbonara small portion
Shin Ramen small
Coke bottle
30% OFF
ONLY FOR 4900 LKR
BUY ONE GET ONE FREE
Buy a large any pizza or a burger and get one free!!
Valid only till 15th December


React App
localhost:3000/menu
Incognito
Sea food
Noodles
Specials
Burgers
Pizza
NEW
Spicy Fish Pizza
2500 LKR
+
Veggie Masala Pizza
3000 LKR
+


React App
localhost:3000/cart
Incognito
HOME
PROMOTIONS
MENU
ADMIN
Sign in
Checkout
Spicy Fish Pizza
Remove
Veggie Masala Pizza
Remove
Name:
Address:
Place Order



Order Details

Order ID: 66c0051464a27dc4ed6cdd2d

Customer Name: sajith lakmal

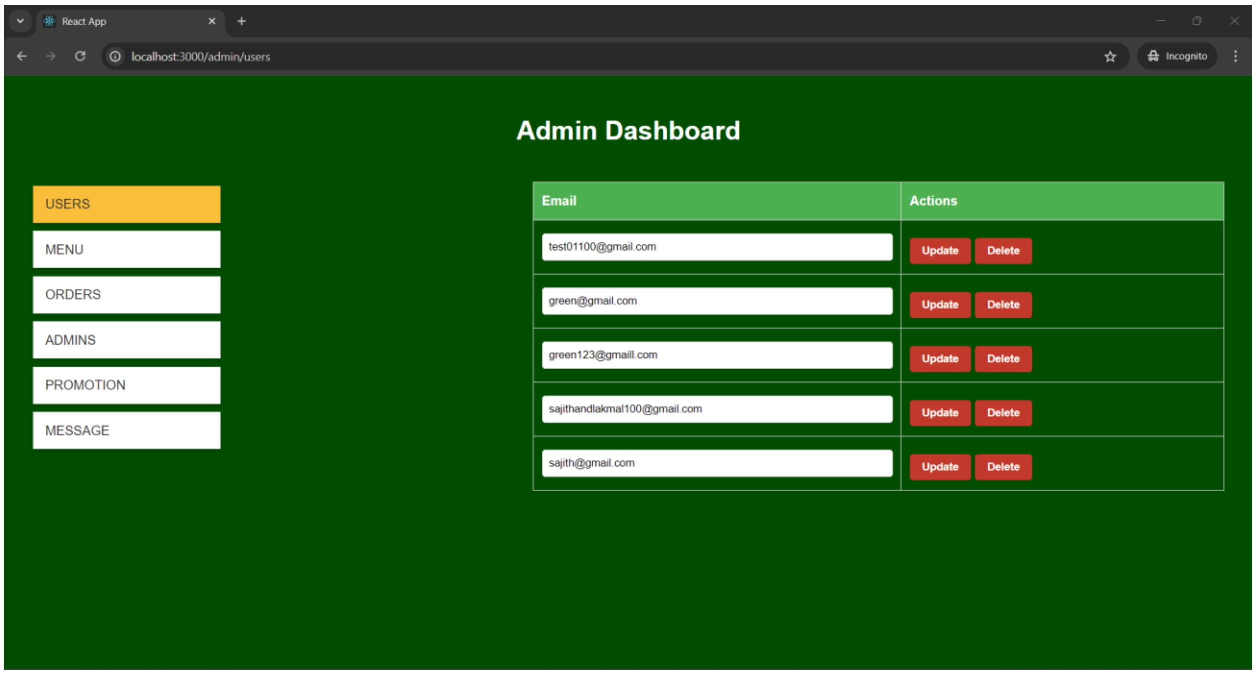
Email: sajithandlakmal100@gmail.com

Delivery Address: 214/22 mewalla road

Order Date: 8/17/2024

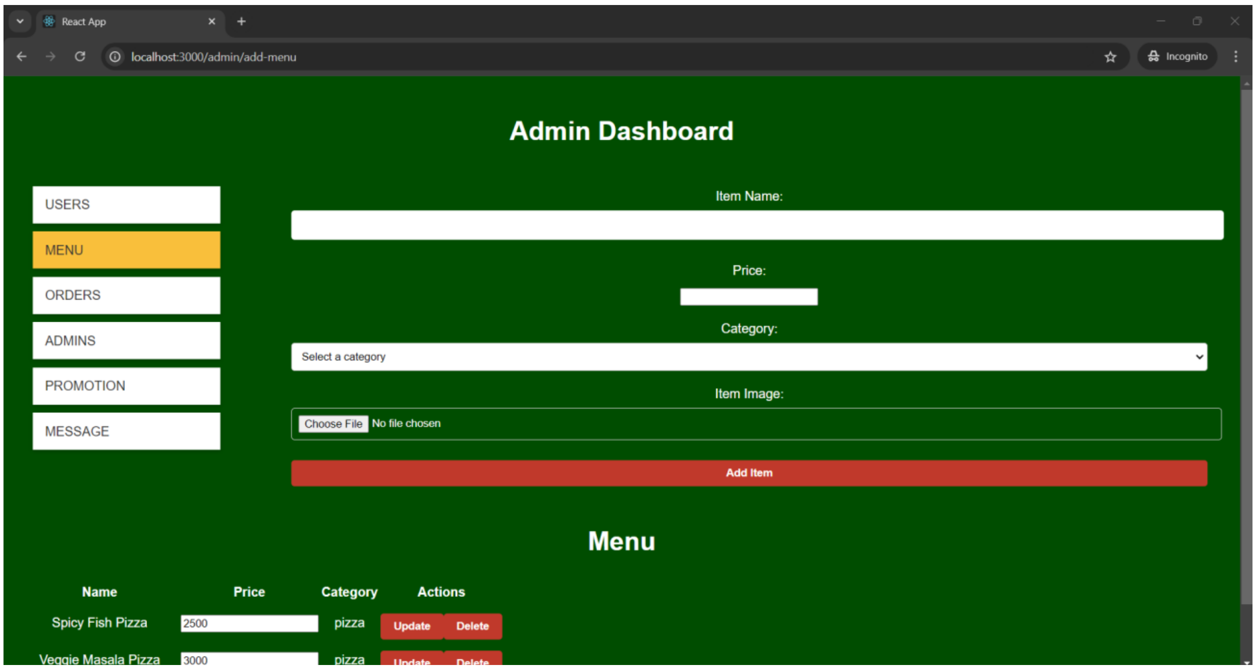
Items:

- Special chicken burger with double cheese layers



Admin Dashboard

Email	Actions
test01100@gmail.com	Update Delete
green@gmail.com	Update Delete
green123@gmaill.com	Update Delete
sajithandlakmal100@gmail.com	Update Delete
sajith@gmail.com	Update Delete



Admin Dashboard

Item Name:

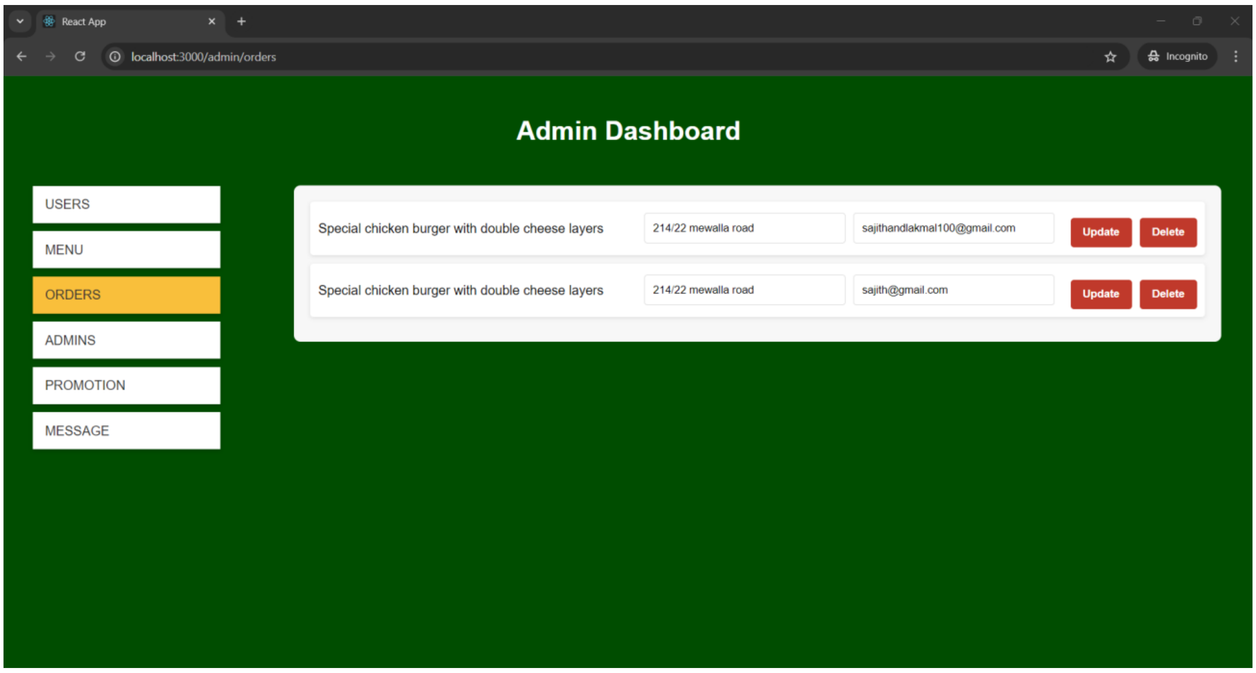
Price:

Category: Select a category

Item Image: No file chosen

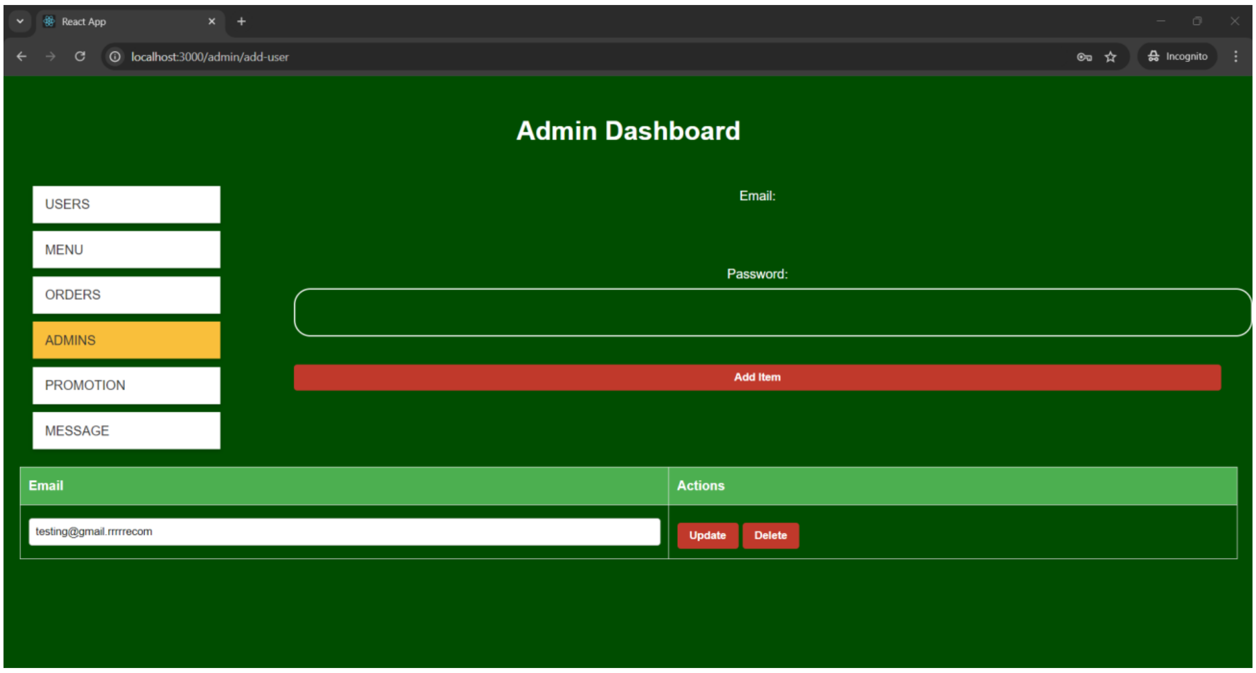
Menu

Name	Price	Category	Actions
Spicy Fish Pizza	2500	pizza	Update Delete
Veggie Masala Pizza	3000	pizza	Update Delete



Admin Dashboard

Special chicken burger with double cheese layers	214/22 mewalla road	sajithandlakmal100@gmail.com	Update Delete
Special chicken burger with double cheese layers	214/22 mewalla road	sajith@gmail.com	Update Delete



Admin Dashboard

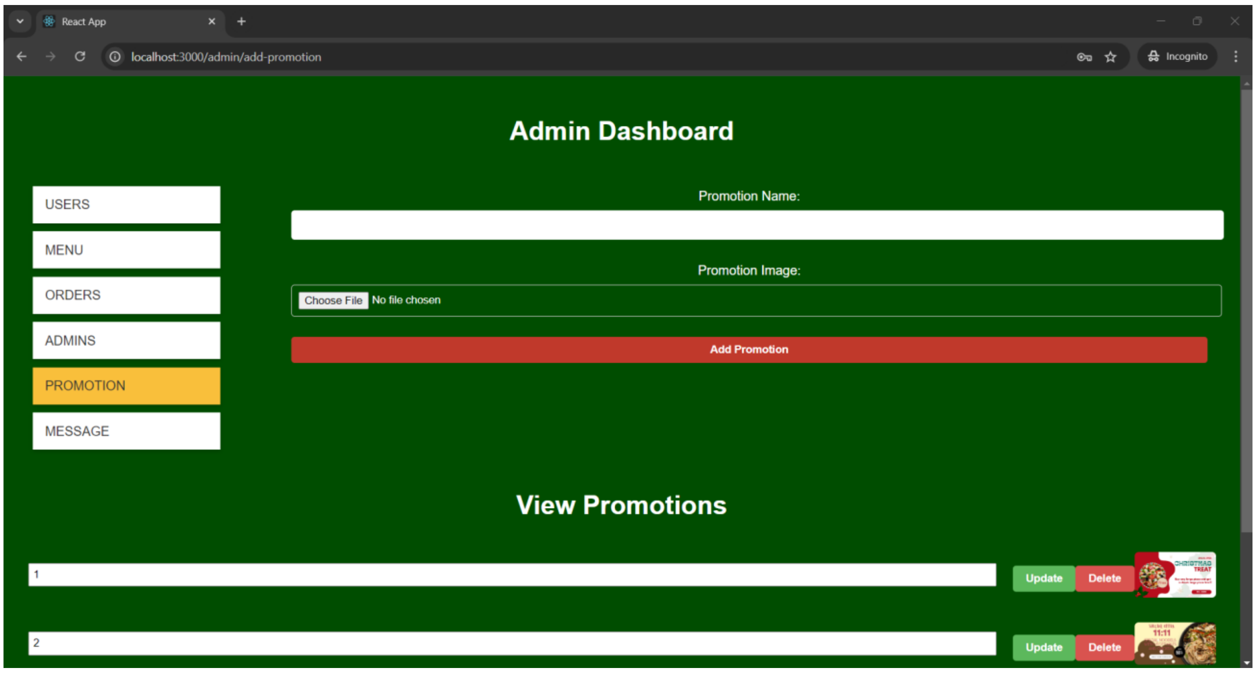
USERS
MENU
ORDERS
ADMINS
PROMOTION
MESSAGE

Email:

Password:

Add Item

Email	Actions
testing@gmail.rrrrecom	Update Delete



Admin Dashboard

USERS
MENU
ORDERS
ADMINS
PROMOTION
MESSAGE

Promotion Name:

Promotion Image:

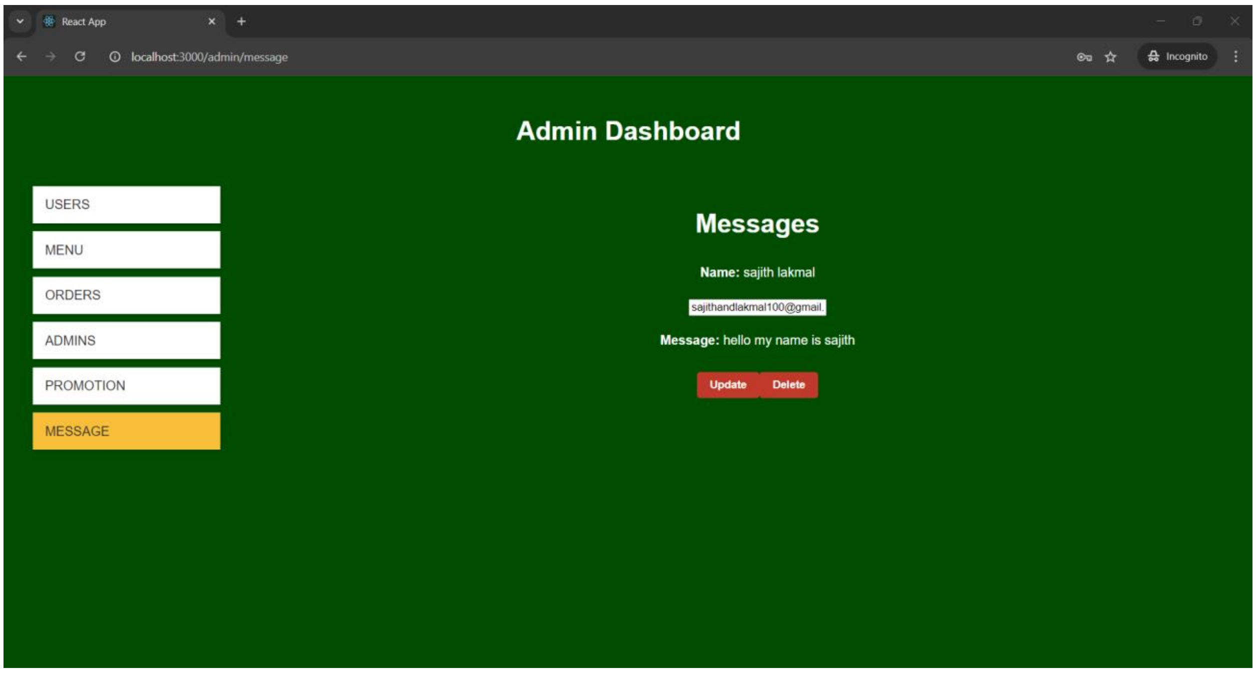
Choose File No file chosen

Add Promotion

View Promotions

1 Update Delete

2 Update Delete



Name: sajith lakmal

sajithandlakmal100@gmail

Message: hello my name is sajith

Reference

GeeksforGeeks. (2024). *Restaurant Reservation System using MERN Stack*. [online] Available at: https://www.geeksforgeeks.org/restaurant-reservation-system-using-mern-stack/.

Sufiyan, T. (2023). *What is Node.js: A Comprehensive Guide*. [online] Simplilearn.com. Available at: https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs.

Hygraph. (2022). *A Complete Guide to Routing in React*. [online] Available at: https://hygraph.com/blog/routing-in-react.

www.w3schools.com. (n.d.). *React Router*. [online] Available at: https://www.w3schools.com/react/react_router.asp.

reactjs.org. (n.d.). *Testing Overview – React*. [online] Available at: https://legacy.reactjs.org/docs/testing.html.