Food ordering Web Application using Mern Stack

Smart Internz(Naan Mudhalvan)

PROJECT REPORT

Submitted by

1.BALAJI R	211121205006
2.BHARANIDHARAN E	211121205007
3.ERAIYANBU J	211121205009
4.RAGHUL S	211121205023

Inpartial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

MADHA ENGINEERING COLLEGE

KUNDRATHUR , CHENNAI-600069

NOV 2024

ABSTRACT

The food ordering website is a comprehensive platform developed using the MERN stack (MongoDB, Express.js, React.js, Node.js) to modernize theprocess of ordering food. This project aims to provide users with a seamless and interactive experience for browsing menus, placing orders, and managing their interactions with restaurants. The use of React.js for the front end ensures a dynamic and responsive interface, while Node.js and Express.js power the back end, enabling efficient data handling and API integration. MongoDB serves as the database, offering scalability and flexibility in managingdata related to users, orders, and menu items.

The platform features user authentication with secure login and registration processes, menu browsing with categorized food items, an integrated shopping cart for managing orders, and real-time order placement. The payment gateway integration provides secure transactions, while an order-tracking module enhances user satisfaction by keeping customers informed about their order status.

This system is designed to optimize both user experience and restaurant operations by digitizing and automating the food ordering process. The MERN stack ensures high performance, scalability, and the potential for integrating future features such as AI-based recommendations, customer reviews, and loyalty programs. The project aligns with the growing demand for online food services, offering a reliable and efficient solution for both customers and food service providers.

TABLE OF CONTENTS

LIST OF FIGURES

1.INTRODUCTION

In today's fast-paced world, online food ordering has revolutionized the way people enjoy their meals. A food ordering website serves as a convenient platform that connects customers with their favorite restaurants, offering a seamless experience to browse menus, place orders, and have food delivered to their doorstep. This innovative solution eliminates the need for traditional dine-in or phone-based orders, saving time and effort for users.

With an extensive range of cuisines, easy payment options, and real-time tracking, food ordering websites cater to diverse customer preferences and enhance the dining experience. They not only benefit customers by providing convenience and variety but also empower restaurants to expand their reach and streamline their operations. In essence, food ordering websites bridge the gap between culinary delights and modern technology, reshaping the way we enjoy food.

1.1 What is a Food Ordering Website?

A food ordering website is an online platform that enables customers to browse restaurant menus, select dishes, place orders, and receive food at their preferred location. It provides a user-friendly interface to simplify the entire process of dining from the comfort of one's home or workplace.

1.2 Importance of Food Ordering Websites

Food ordering websites are crucial in the modern era as they cater to the increasing demand for convenience. They eliminate the need for phone-based or in-person orders and offer real-time access to a variety of dining options.

1.3 How Food Ordering Websites Work

The process involves customers registering or logging in, exploring restaurants or cuisines, placing their order, choosing a payment method, and tracking the delivery in real-time. It integrates multiple services like menu browsing, secure payments, and delivery tracking into a single platform.

2.PURPOSE AND FEATURES

The primary purpose of my food ordering website is to simplify and enhance the process of ordering food online, catering to the needs of both users and restaurant partners. It aims to provide a platform where users can explore diverse cuisines, place orders effortlessly, and enjoy their favorite meals from the comfort of their homes. By connecting customers with a wide range of restaurants, the website ensures convenience, accessibility, and an improved dining experience. Additionally, the platform empowers restaurant owners to reach a larger audience, expand their customer base, and manage orders efficiently through a digital interface.

2.1 Features :

1.User-Friendly Interface :

- Easy-to-navigate design for browsing menus and placing orders.
- Responsive layout for seamless access on desktops, tablets, and smartphones.

2. Restaurant Listings and Menu Display :

- Comprehensive listings of restaurants with detailed menus and pricing.
- Cuisines categorized for quick and easy access.

3. Secure Login and Payment :

- User authentication for secure access to accounts and order history.
- Multiple payment options, including credit/debit cards, digital wallets, and UPI, ensuring safe and convenient transactions.

4. Order Tracking and Updates :

- Real-time tracking of orders from preparation to delivery.
- Notifications to keep users informed about order status.

5. Search and Filter Options :

- Advanced search features to find dishes, cuisines, or restaurants.
- Filters for sorting by ratings, price, distance, or special offers.

3.TECHNICAL ARCHITECTURE

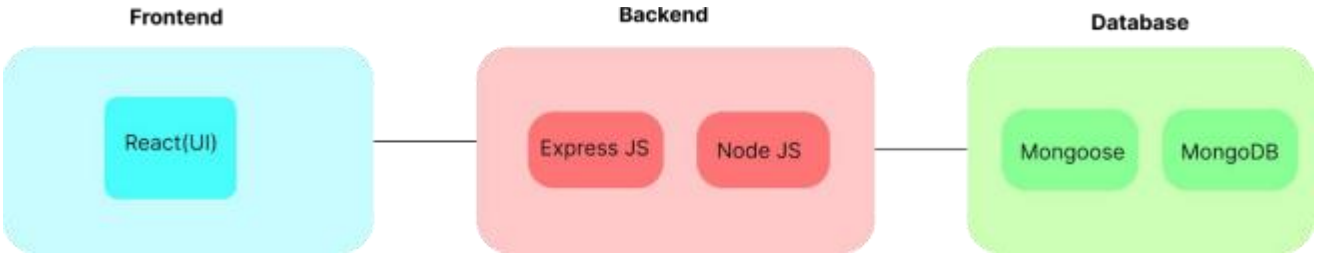


Figure 1.1

In This Architecture Diagram:

- The Frontend is represented by the "Frontend" section, including user interface components such as User Authentication, Cart, Products, Profile, Admin dashboard, etc.,The Frontend is the user's gateway to the website, providing a seamless and visually engaging interface for interaction. Built using React.js, it ensures a responsive, dynamic, and intuitive user experience.
- The Backend is represented by the "Backend" section, consisting of API endpoints for Users, Orders, Products, etc., It also includes Admin Authentication and an Admin Dashboard. The Backend is the engine that powers the website's functionality, handling data storage, processing, and communication between the **Frontend** and the **Database**. Built using **Node.js** and **Express.js**, it establishes a robust, scalable, and secure server environment.
- The Database section represents the database that stores collections for Users, Admin, Cart, Orders, and products.

4.PREREQUISITES

To develop a full-stack food ordering app using ReactJS, Node.js, and MongoDB, there are several prerequisites you should consider. Here are the key prerequisites for developing such an application:

4.1 Tools and Technologies :

Node.js and npm: Install Node.js, which includes npm (Node Package Manager), on your development machine. Node.js is required to run JavaScript on the server side.

- **Download:** https://nodejs.org/en/download/

Installation Instructions :https://nodejs.org/en/download/package-manager/

MongoDB: Set up a MongoDB database to store hotel and booking information. Install MongoDB locally or use a cloud-based MongoDB service.

- **Download:** https://www.mongodb.com/try/download/community
- **Installation instructions:** https://docs.mongodb.com/manual/installation/

Express.js: Express.js is a web application framework for Node.js. Install Express.js to handle server-side routing,middleware, and API development.

- **Installation:** Open your command prompt or terminal and run the following command: **npm install express**

React.js: React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications. To install React.js, a JavaScript library for building user interfaces, follow the installation guide: https://reactjs.org/docs/create-a-new-react-app.html

HTML, CSS, and JavaScript: Basic knowledge of HTML for creating the structure of your app, CSS for styling,and JavaScript for client-side interactivity is essential.

Database Connectivity: Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

Front-end Framework: Utilize Angular to build the user-facing part of the application, including product listings, booking forms, and user interfaces for the admin dashboard.

Version Control: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

Git: Download and installation instructions can be found at: https://git scm.com/downloads

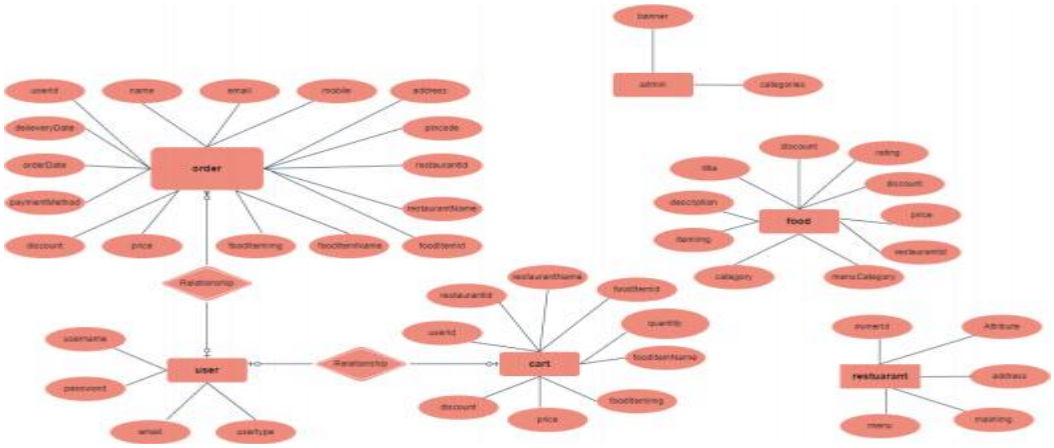
Development Environment : Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

To Connect the Database with Node JS go through the below provided link:

Link: https://www.section.io/engineering-education/nodejs- mongoosejs- mongodb/

- Visual Studio Code: Download from https://code.visualstudio.com/download
- Sublime Text: Download from https://www.sublimetext.com/download
- WebStorm: Download from https://www.jetbrains.com/webstorm/download

5. ER DIAGRAM



Fligure 1.2

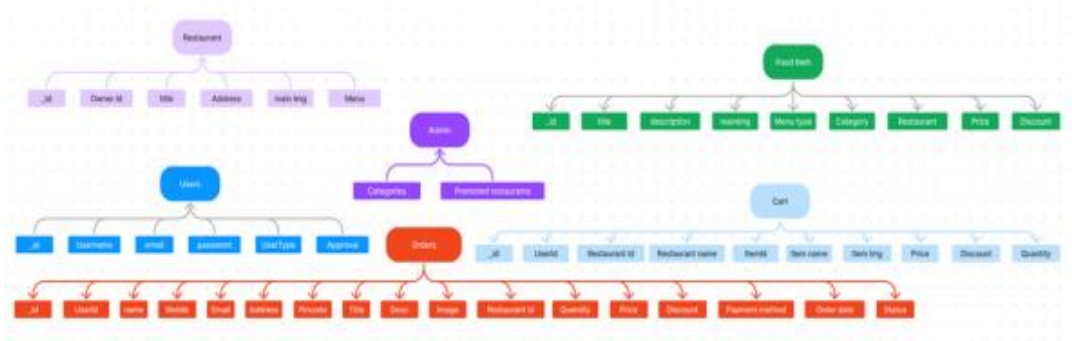


Figure 1.3

The ER-diagram represents the entities and relationships involved in an food

ordering e-commerce system. It illustrates how users, restaurants, products, carts, and orders are interconnected. Here is a breakdown of the entities and their relationships:

User: Represents the individuals or entities who are registered in the platform.

Restaurant: This represents the collection of details of each restaurant in the platform.

Admin: Represents a collection with important details such as promoted restaurants and Categories.

Products: Represents a collection of all the food items available in the platform.

Cart: This collection stores all the products that are added to the cart by users. Here, the elements in the cart are differentiated by the user Id.

Orders: This collection stores all the orders that are made by the users in the platform.

6. USER & ADMIN FLOW

6.1. User Flow :

- Users start by registering for an account.
- After registration, they can log in with their credentials.
- Once logged in, they can check for the available products in the platform.
- Users can add the products they wish to their carts and orders.
- They can then proceed by entering address and payment details.
- After ordering, they can check them in the profile section.

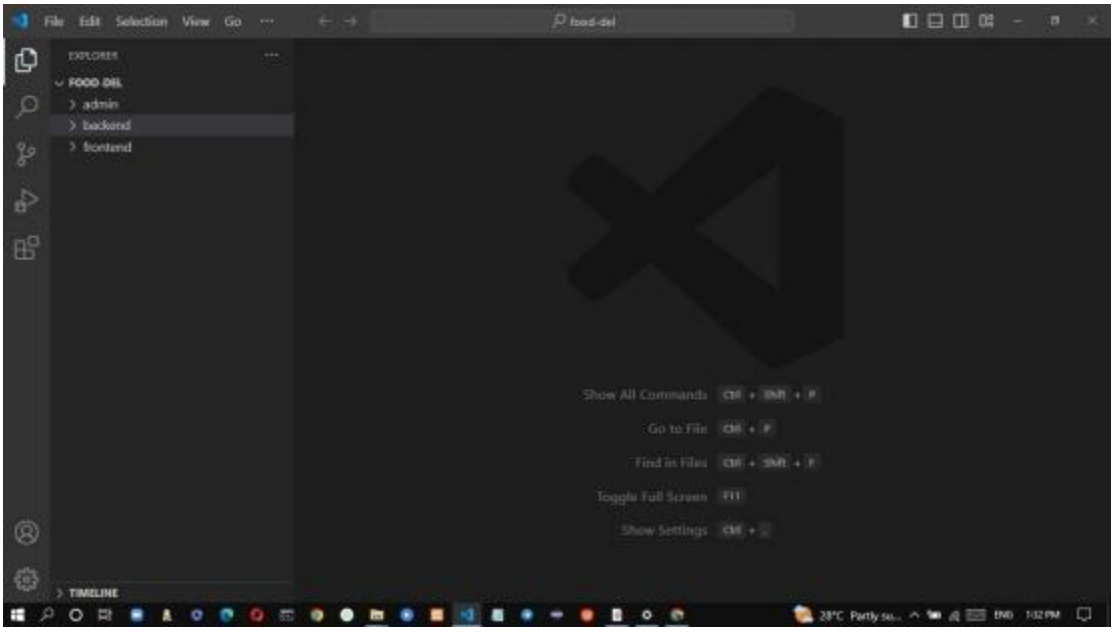
6.2 Restaurant Flow :

- Restaurants start by authenticating with their credentials.
- They need to get approval from the admin to start listing the products.
- They can add/edit the food items.

6.3 Admin Flow :

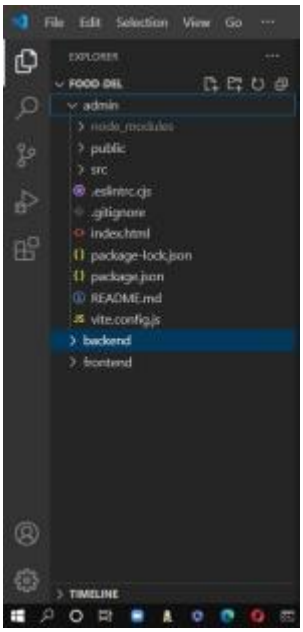
- Admins start by logging in with their credentials.
- Once logged in, they are directed to the Admin Dashboard.
- Admins can access the users list, products, orders, etc.

7 . FOLDER STRUCTURE



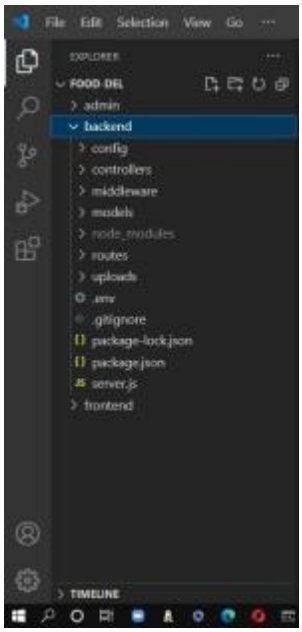
Fligure 1. 4

7.1Admin :



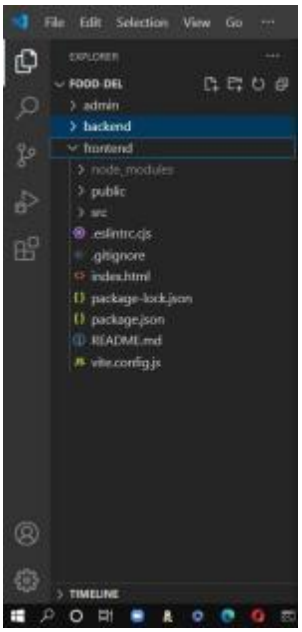
Fligure 1. 5

7.2Backend :



Fligure 1. 6

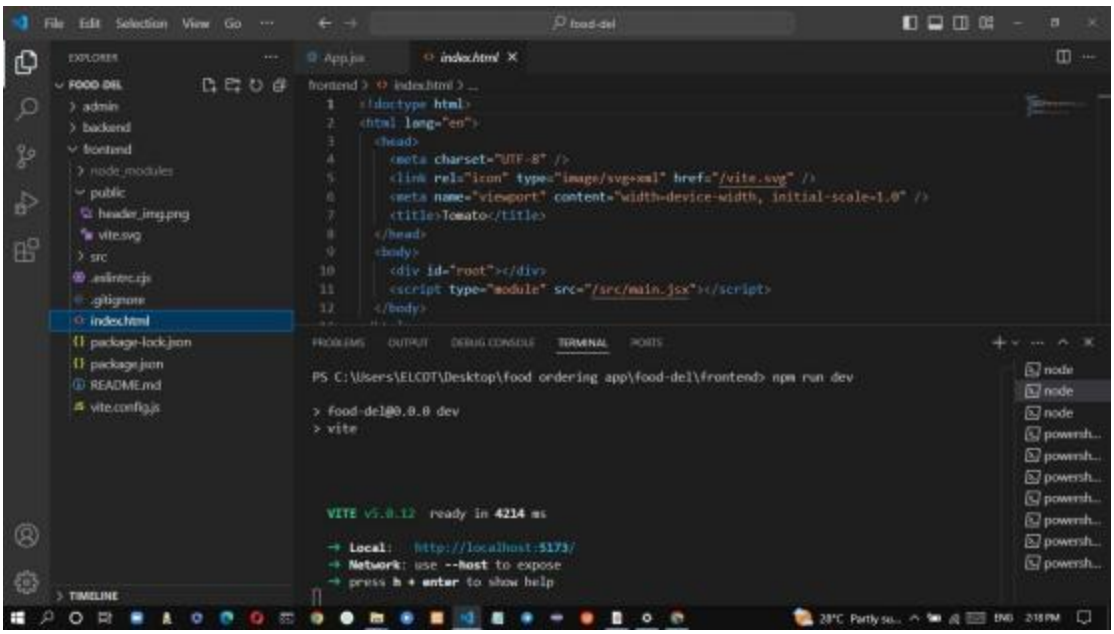
7.3Frontend :



Fligure 1. 7

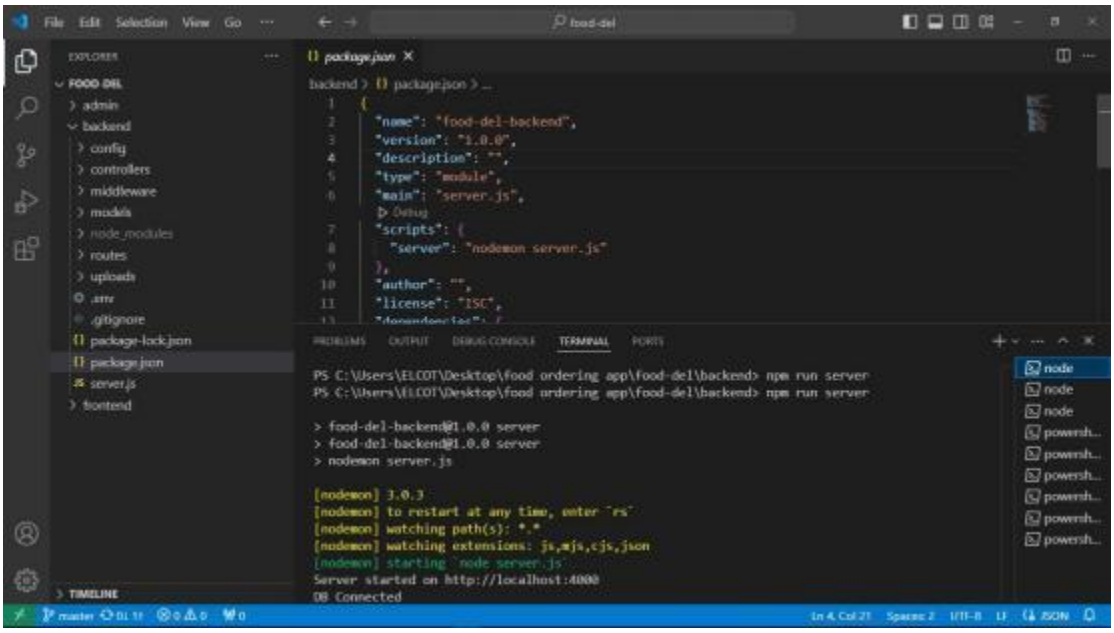
8 . RUNNING THE APPLICATION

8.1 FRONTEND :



Fligure 1. 8

8.2 BACKEND :



Fligure 1. 9

9 . PROGRAM SCREENSHOTS

■ 9.1 Admin pages :

➢ index.html :

```
< !doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>Food Del Admin Panel</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

➢ package.json :

```
{
  "name" : "food-del-admin",
  "private" : true,
  "version" : "0.0.0",
  "type" : "module",
  "scripts" : {
    "dev" : "vite",
    "build" : "vite build",
    "lint" : "eslint . --ext js,jsx --report-unused-disable-directives -
-max-warnings 0",
    "preview" : "vite preview"
  },
  "dependencies" : {
    "axios" : "^1.6.7",
    "react" : "^18.2.0",
    "react-dom": "^18.2.0",
    "react-router-dom" : "^6.22.0",
    "react-toastify" : "^10.0.4"
  },
  "devDependencies" : {
    "@types/react" : "^18.2.55",
    "@types/react-dom" : "^18.2.19",
    "@vitejs/plugin-react": "^4.2.1",
```

```
    "eslint": "^8.56.0",
    "eslint-plugin-react": "^7.33.2",
    "eslint-plugin-react-hooks": "^4.6.0",
    "eslint-plugin-react-refresh": "^0.4.5",
    "vite": "^5.1.0"
  }
}
```

- **asset.js :**

```
import logo from './logo.png'
import add_icon from './add_icon.png'
import order_icon from './order_icon.png'
import profile_image from './profile_image.png'
import upload_area from './upload_area.png'
import parcel_icon from './parcel_icon.png'

export const url = 'http://localhost:4000
' export const currency = '₹'
```

```
export const assets ={
    logo,
    add_icon,
    order_icon,
    profile_image,
    upload_area,
    parcel_icon
}
```

- **Navbar.css :**

```
.navbar{
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 8px 4%;
}
.navbar .logo{
    width: max(10%,80px);
}
.navbar .profile{
    width: 40px;
}
```

- **Navbar.jsx :**

```
import React from 'react'
import './Navbar.css'
import { assets } from '../../assets/assets'

const Navbar = () => {
  return (
    <div className='navbar'>
      <img className='logo' src={assets.logo} alt="" />
      <img className='profile' src={assets.profile_image} alt="" />
    </div>
  )
}
```

```
export default Navbar
```

- **Sidebar.css :**

```
.sidebar{
    width: 18%;
    min-height: 100vh;
    border: 1.5px solid #A9A9A9;
    border-top: 0;
}
.sidebar-options{
    padding-top: 50px;
    padding-left: 20%;
    display: flex;
    flex-direction: column;
    gap: 20px;
}
.sidebar-option{
    display: flex;
    align-items: center;
    gap: 12px;
    border: 1px solid #A9A9A9;
    border-right: 0;
    padding: 8px 10px;
    border-radius: 3px 0px 0px 3px;
    cursor: pointer;
}
.sidebar{
    font-size: max(1vw,10px );
}
```

```
.sidebar-option img{
    width : 20px;
}
.sidebar-option.active {
 background-color: #fff0ed;
 border-color: tomato;
}
@media (max-width:900px) {
    .sidebar-option p{
       display: none;
    }
}
```

- **Sidebar.jsx :**

```
import React from 'react'
import  './Sidebar.css'
import { assets } from '../../assets/assets'
import { NavLink } from 'react-router-dom'

const Sidebar = () => {
  return (
    <div className='sidebar'>
      <div className="sidebar-options">
        <NavLink to='/add' className="sidebar-option">
            <img src={assets.add_icon} alt="" />
            <p>Add Items</p>
        </NavLink>
        <NavLink to='/list' className="sidebar-option">
            <img src={assets.order_icon} alt="" />
            <p>List Items</p>
        </NavLink>
        <NavLink to='/orders' className="sidebar-option">
            <img src={assets.order_icon} alt="" />
            <p>Orders</p>
        </NavLink>
      </div>
    </div>
  )
}
```

```
export default Sidebar
```

- **Add.css:**

```
.add{
    width : 70%;
    margin-left: max(5vw,25px);
    margin-top : 50px;
    color : #6D6D6D;
    font-size : 16px;
}
.add form{
   gap: 20px;
}
.add-img-upload img{
    width : 120px;
}
.add-product-name, .add-product-description{
    width : max(40%,280px);
}
.add-product-name input ,.add-product-description textarea{
    padding : 10px 10px;
}
.add-category-price{
    display : flex;
    gap : 30px;
}
.add-category-price select,.add-category-price input{
    max-width : 120px;
    padding : 10px;
}
.add-btn{
    max-width : 120px;
    border : none;
    padding : 10px;
    background-color : black;
    color : white;
    cursor : pointer;
}
```

➢ **Add.jsx :**

```
.list-table-format{
    display : grid;
    grid-template-columns: 0.5fr 2fr 1fr 1fr 0.5fr;
    align-items: center;
    gap : 10px;
    padding : 12px 15px;
    border : 1px solid #cacaca;
    font-size : 13px;
}
.list-table-format.title{
    background-color : #f9f9f9;
}
.list-table-format img{
    width : 50px;
}
@media(max-width :600px){

    .list-table-format{
        grid-template-columns : 1fr 3fr 1fr;
        gap: 15px;
    }
    .list-table-format.title{
        display: none;
    }
}
```

➢ **List.jsx :**

```
import React, { useEffect, useState } from 'react'
import './List.css'
import { url, currency } from '../../assets/assets'
import axios from 'axios';
import { toast } from 'react-toastify';

const List = () => {
```

```
  const [list, setList] = useState([]);
```

- **Orders.css :**

```
.order-item{
    display : grid;
    grid-template-columns: 0.5fr 2fr 1fr 1fr 1fr;
    align-items: start;
    gap : 30px;
    border : 1px solid tomato;
    padding : 20px;
    margin : 30px 0px;
    font-size : 14px;
    color : #505050;
}
.order-item-food,.order-item-name{
    font-weight: 600;
}
.order-item-name{
    margin-top : 30px;
    margin-bottom: 5px;
}
.order-item-address{
    margin-bottom: 10px;
}
.order-item select{
    background-color : #ffe8e4;
    border : 1px solid tomato;
    width : max(10vw,120px);
    padding : 10px;
    outline :none;
}
@media(max-width :1000px){
    .order-item{
        font-size: 12px;
        grid-template-columns : 0.5fr 2fr 1fr;
        padding: 15px 8px;
    }
    .order-item select{
        padding: 5px;
        font-size: 12px;
    }
    .order-item img{
        width: 40px;
    }
```

```
              <p className='order-item-food'>
                {order.items.map((item, index) => {
                  if (index === order.items.length - 1) {
                    return item.name + " x " + item.quantity
                  }
                  else {
                    return item.name + " x " + item.quantity + ", "
                  }
                })}
              </p>
              <p className='order-item-name'>{order.address.firstName +
" " + order.address.lastName}</p>
              <div className='order-item-address'>
                <p>{order.address.street + ","}</p>
                <p>{order.address.city + ", " + order.address.state + ",
" + order.address.country + ", " + order.address.zipcode}</p>
              </div>
              <p className='order-item-phone'>{order.address.phone}</p>
            </div>
            <p>Items : {order.items.length}</p>
            <p>{currency}{order.amount}</p>
            <select onChange={(e) => statusHandler(e, order._id)}
value={order.status} name="" id="">
              <option value="Food Processing">Food Processing</option>
              <option value="Out for delivery">Out for
delivery</option>
              <option value="Delivered">Delivered</option>
            </select>
          </div>
        ))}
      </div>
    </div>
  )
}
```

```
export default Order
```

- **App.jsx :**

```
import React from 'react'
import Navbar from './components/Navbar/Navbar'
import Sidebar from './components/Sidebar/Sidebar'
import { Route, Routes } from 'react-router-dom'
import Add from './pages/Add/Add'
import List from './pages/List/List'
import Orders from './pages/Orders/Orders'
import { ToastContainer } from 'react-toastify';
```

```
import 'react-toastify/dist/ReactToastify.css';

const App = () => {
  return (
    <div className='app'>
      <ToastContainer />
      <Navbar />
      <hr />
      <div className="app-content">
        <Sidebar />
        <Routes>
          <Route path="/add" element={<Add />} />
          <Route path="/list" element={<List />} />
          <Route path="/orders" element={<Orders />} />
        </Routes>
      </div>
    </div>
  )
}
```

- **index.css :**

```
@import
url('https://fonts.googleapis.com/css2?family=Outfit:wght@100..900&disp
lay=swap');
*{
  padding: 0;
  margin: 0;
  box-sizing: border-box;
  font-family: Outfit;
}
body{
  min-height: 100vh;
  background-color: #FCFCFC;
}
a{
  text-decoration: none;
  color: inherit;
}
hr{
  border: none;
  height: 1px;
  background-color: #A9A9A9;
}
.app-content{
```

```
  display: flex;
}
.flex-col{
  display: flex;
  flex-direction: column;
  gap: 10px;
}
.cursor{
  cursor: pointer;
}
```

- **main.css :**

```
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'
import { BrowserRouter } from 'react-router-dom'

ReactDOM.createRoot(document.getElementById('root
  ')).render( <BrowserRouter>
    <App />
  </BrowserRouter>
```

```
)
```

- **Vite.config.js :**

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'

// https://vitejs.dev/config/
export default defineConfig({
  plugins: [react()],
  server: { port: 5174 },
})
```

■ 9.2 Frontend pages :

➢ index.html :

```
< !doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>Tomato</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

➢ assets.js :

```
import basket_icon from './basket_icon.png'
import logo from './logo.png'
import header_img from './header_img.png'
import search_icon from './search_icon.png'
import menu_1 from './menu_1.png'
import menu_2 from './menu_2.png'
import menu_3 from './menu_3.png'
import menu_4 from './menu_4.png'
import menu_5 from './menu_5.png'
import menu_6 from './menu_6.png'
import menu_7 from './menu_7.png'
import menu_8 from './menu_8.png'

import food_1 from './food_1.png'
import food_2 from './food_2.png'
import food_3 from './food_3.png'
import food_4 from './food_4.png'
import food_5 from './food_5.png'
import food_6 from './food_6.png'
import food_7 from './food_7.png'
import food_8 from './food_8.png'
import food_9 from './food_9.png'
import food_10 from './food_10.png'
```

```
    remove_icon_red
    , app_store,
    play_store,
    linkedin_icon
    ,
    facebook_icon
    ,
    twitter_icon,
    cross_icon,
    selector_icon
    ,
    profile_icon,
    logout_icon,
    bag_icon,
    parcel_icon,
```

```
        _id: "30",
        name: "Veg Noodles",
        image: food_30,
        price: 12,
        description: "Food provides essential nutrients for overall
health and well-being",
        category: "Noodles"
    }, {
        _id: "31",
        name: "Somen Noodles",
        image: food_31,
        price: 20,
        description: "Food provides essential nutrients for overall
health and well-being",
        category: "Noodles"
    }, {
        _id: "32",
        name: "Cooked Noodles",
        image: food_32,
        price: 15,
```

AppDownload.css :

```
.app-download {
    margin: 100px auto;
    font-size: max(3vw, 20px);
    text-align: center;
    font-weight: 500;
}
.app-download-platforms{
    display: flex;
    justify-content: center;
    gap: max(2vw,10px);
    margin-top: 40px;
}
.app-download-platforms img{
   width :max(30vw,120px);
   max-width: 180px;
   transition: 0.5s;
   cursor: pointer;
}
.app-download-platforms img:hover{
```

➢ AppDownload.jsx :

```
import React from 'react'
import './AppDownload.css'
import { assets } from '../../assets/assets'

const AppDownload = () => {
    return (
        <div className='app-download' id='app-download'>
            <p>For Better Experience Download <br />Tomato App</p>
            <div className="app-download-platforms">
                <img src={assets.play_store} alt="" />
                <img src={assets.app_store} alt="" />
            </div>
        </div>
    )
}
```

```
export default AppDownload
```

➢ ExploreMenu.css :

```
.explore-menu{
    display: flex;
    flex-direction: column;
    gap: 20px;
}
.explore-menu h1{
    color: #262626;
    font-weight: 500;
}
.explore-menu-text{
    max-width: 60%;
    color: #808080;
}
.explore-menu-list{
    display: flex;
    justify-content: space-between;
    align-items: center;
    gap: 30px;
    text-align: center;
    margin: 20px 0px;
```

```
    overflow-x : scroll;
}
.explore-menu-list: :-webkit-scrollbar{
    display : none;
}
.explore-menu-list-item img{
    width : 7.5vw;
    min-width : 80px;
    cursor : pointer;
    border-radius: 50%;
    transition : 0.2s;
}
.explore-menu-list-item .active{
    border : 4px solid tomato;
    padding : 2px;
}
.explore-menu-list-item p{
    margin-top : 10px;
    color : #747474;
    font-size : max(1.4vw,16px);
    cursor : pointer;
}
.explore-menu hr{
    margin : 10px 0px;
    height : 2px;
    background-color : #E2E2E2;
    border : none;
}
@media (max-width:1050px) {
    .explore-menu-text{
        max-width: 100%;
        font-size: 14px;
    }
}
```

- **ExploreMenu.jsx :**

```
import React, { useContext } from 'react'
import './ExploreMenu.css'
import { StoreContext } from '../../Context/StoreContext'

const ExploreMenu = ({category,setCategory}) => {
```

```
      <h1>Explore our menu</h1>
      <p className='explore-menu-text'>Choose from a diverse menu
featuring a delectable array of dishes. Our mission is to satisfy your
cravings and elevate your dining experience, one delicious meal at a
time.</p>
      <div className="explore-menu-list">
        {menu_list.map((item,index)=>{
            return (
                <div
onClick={()=>setCategory(prev=>prev===item.menu_name?"All":item.menu_na
me)} key={index} className='explore-menu-list-item'>
                    <img src={item.menu_image}
className={category===item.menu_name?"active":""} alt="" />
                    <p>{item.menu_name}</p>
                </div>
            )
        })}
      </div>
      <hr />
    </div>
  )
}
```

```
export default ExploreMenu
```

➢ FoodDisplay.css :

```
.food-display{
    margin-top: 30px;
}
.food-display h2{
    font-size: max(2vw,24px);
    font-weight: 600;
}
.food-display-list{
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(240px, 1fr));
    margin-top: 30px;
    gap: 30px;
    row-gap:
50px; }
```

➢ FoodDisplay.jsx :

```
import './FoodDisplay.css'
import FoodItem from '../FoodItem/FoodItem'
import { StoreContext } from '../../Context/StoreContext'

const FoodDisplay = ({category}) => {
```

```
  const {food_list} = useContext(StoreContext);
```

```
  return (
    <div className='food-display' id='food-display'>
      <h2>Top dishes near you</h2>
      <div className='food-display-list'>
        {food_list.map((item)=>{
          if (category==="All" || category===item.category) {
            return <FoodItem key={item._id} image={item.image}
name={item.name} desc={item.description} price={item.price}
id={item._id}/>
          }
        })}
      </div>
    </div>
  )
}
```

```
export default FoodDisplay
```

➢ FoodItem.css :

```
.food-item{
    width: 100%;
    margin: auto;
    border-radius: 15px;
    box-shadow: 0px 0px 10px #00000015;
    transition: 0.3s;
    animation: fadeIn 1s;
}
.food-item:hover{
    transform: scale(1.02);
}
.food-item-img-container{
    position: relative;
}
.food-item-img-container .add{
```

```
    cursor : pointer;
    border-radius: 50%;
}
.food-item-img-container .add:hover{
    border : 2px solid #FF4C24;
}
.food-item-image{
   width : 100%;
   border-radius : 15px 15px 0px 0px;
}
.food-item-counter{
    position : absolute;
    bottom : 15px;
    right : 15px;
    display : flex;
    align-items: center;
    gap : 10px;
    padding : 6px;
    border-radius: 50px;
    background-color : white;
}
.food-item-counter img{
    width : 30px;
}
.food-item-info{
    padding : 20px;
}
.food-item-name-rating{
    display : flex;
    justify-content : space-between;
    align-items: center;
    margin-bottom: 10px;
}
.food-item-name-rating p{
    font-size : 20px;
    font-weight: 500;
}
.food-item-name-rating img{
    width : 70px;
}
.food-item-desc{
    color : #676767;
    font-size : 12px;
}
.food-item-price{
    color : #FF4C24;
    font-size : 22px;
    font-weight: 500;
```

```
    margin : 10px 0px;
}
```

➢ FoodItem.jsx :

```
import React, { useContext, useState } from 'react'
import './FoodItem.css'
import { assets } from '../../assets/assets'
import { StoreContext } from '../../Context/StoreContext';

const FoodItem = ({ image, name, price, desc , id }) => {
```

```
    const [itemCount, setItemCount] = useState(0);
    const {cartItems,addToCart,removeFromCart,url,currency} =
useContext(StoreContext);
```

```
    return (
        <div className='food-item'>
            <div className='food-item-img-container'>
                <img className='food-item-image'
src={url+"/images/"+image} alt="" />
                {!cartItems[id]
                ?<img className='add' onClick={() => addToCart(id)}
src={assets.add_icon_white} alt="" />
                :<div className="food-item-counter">
                        <img src={assets.remove_icon_red}
onClick={()=>removeFromCart(id)} alt="" />
                        <p>{cartItems[id]}</p>
                        <img src={assets.add_icon_green}
onClick={()=>addToCart(id)} alt="" />
                    </div>
                }
            </div>
            <div className="food-item-info">
                <div className="food-item-name-rating">
                    <p>{name}</p> <img src={assets.rating_starts}
alt="" />
                </div>
                <p className="food-item-desc">{desc}</p>
                <p className="food-item-price">{currency}{price}</p>
            </div>
        </div>
    )
}
```

```
export default FoodItem
```

➢ Footer.css :

```
.footer{
    color : #D9D9D9;
    background-color : #323232;
    display : flex;
    flex-direction : column;
    align-items: center;
    gap : 20px;
    padding : 20px 8vw;
    padding-top: 80px;
}
.footer-content{
    width : 100%;
    display : grid;
    grid-template-columns: 2fr 1fr 1fr;
    gap : 80px;
}
.footer-content-left , .footer-content-center , .footer-content-right {
    display : flex;
    flex-direction : column;
    align-items: start;
    gap : 20px;
}
.footer-content-left ul li, .footer-content-center ul li, .footer-
content-right ul li{
    margin-bottom: 10px;
    list-style : none;
}
.footer-content-left h2, .footer-content-center h2, .footer-content-
right h2{
    color : white;
}
.footer-social-icons img{
    width : 40px;
}
.footer-social-icons{
    display : flex;
    gap : 15px;
}
.footer li{
    cursor : pointer;
}
.footer hr{
    width : 100%;
    height : 2px;
    margin : 20px 0px;
```

```
}
@media (max-width:750px) {
    .footer-content{
        display: flex;
        flex-direction : column;
        gap: 35px;
    }
    .footer-copyright{
        text-align : center;
    }
}
```

```
import React from 'react'
import './Footer.css'
import { assets } from '../../assets/assets'

const Footer = () => {
  return (
    <div className='footer' id='footer'>
      <div className="footer-content">
        <div className="footer-content-left">
            <img src={assets.logo} alt="" />
            <p>Lorem Ipsum is simply dummy text of the printing and
typesetting industry. Lorem Ipsum has been the industry's standard
dummy text ever since the 1500s, when an unknown printer took a galley
of type and scrambled it to make a type specimen book.</p>
            <div className="footer-social-icons">
                <img src={assets.facebook_icon} alt="" />
                <img src={assets.twitter_icon} alt="" />
                <img src={assets.linkedin_icon} alt="" />
            </div>
        </div>
        <div className="footer-content-center">
            <h2>COMPANY</h2>
            <ul>
                <li>Home</li>
                <li>About us</li>
                <li>Delivery</li>
                <li>Privacy policy</li>
            </ul>
        </div>
        <div className="footer-content-right">
            <h2>GET IN TOUCH</h2>
            <ul>
                <li>+1-212-456-7890</li>
```

```
                <li>contact@tomato.com</li>
            </ul>
          </div>
      </div>
      <hr />
      <p className="footer-copyright">Copyright 2024 © Tomato.com - All
Right Reserved.</p>
    </div>
  )
}
```

```
export default Footer
```

```
.header{
    height : 34vw;
    margin : 30px auto;
    background : url('/header_img.png ') no-repeat;
    background-size:contain;
    position : relative;
}
.header-contents{
    position : absolute;
    display : flex;
    flex-direction : column;
    align-items: start;
    gap : 1.5vw;
    max-width : 50%;
    bottom : 10%;
    left : 6vw;
    animation : fadeIn 3s;
}
.header-contents h2{
    font-weight:500;
    color : white;
    font-size : max(4.5vw,22px);
}
.header-contents p{
    color : white;
    font-size : 1vw;
}
.header-contents button{
    border : none;
    color : #747474;
    font-weight: 500;
    padding : 1vw 2.3vw;
```

```
    background : white;
    font-size : max(1vw,13px);
    border-radius: 50px;
}
@media (max-width:1050px) {
    .header{
        height: 38vw;
    }
    .header-contents{
        max-width: 45%;
    }
}
@media(max-width :750px){
    .header-contents{
        max-width: 55%;
    }
    .header-contents p{
       display: none;
    }
    .header-contents button{
        padding: 2vw 4vw;
    }
}
```

- **Header.jsx :**

```
import React from 'react'
import './Header.css'

const Header = () => {
    return (
        <div className='header'>
            <div className='header-contents'>
                <h2>Order your favourite food here</h2>
                <p>Choose from a diverse menu featuring a delectable
array of dishes crafted with the finest ingredients and culinary
expertise. Our mission is to satisfy your cravings and elevate your
dining experience, one delicious meal at a time.</p>
                <button>View Menu</button>
            </div>
        </div>
    )
}
```

```
export default Header
```

```
.login-popup{
    position : absolute;
    z-index : 1;
    width : 100%;
    height : 100%;
    background-color : #00000090;
    display : grid;
}
.login-popup-container{
    place-self : center;
    width : max(23vw,330px);
    color : #808080;
    background-color : white;
    display : flex;
    flex-direction : column;
    gap : 25px;
    padding : 25px 30px;
    border-radius: 8px;
    font-size : 14px;
    animation : fadeIn 0.5s;
}
.login-popup-title{
    display : flex;
    justify-content : space-between;
    align-items: center;
    color : black;
}
.login-popup-title img{
    width : 16px;
    cursor : pointer;
}
.login-popup-inputs{
    display : flex;
    flex-direction : column;
    gap : 20px;
}
.login-popup-inputs input{
    outline : none;
    border : 1px solid #C9C9C9;
    padding : 10px;
    border-radius: 4px;
}
.login-popup-container button{
    border : none;
    padding : 10px;
```

```
    border-radius: 4px;
    color : white;
    background-color : #FF4C24;
    font-size : 15px;
    cursor : pointer;
}
.login-popup-condition{
    display : flex;
    align-items: start;
    gap : 8px;
    margin-top : -15px;
}

.login-popup-condition input{
    margin-top : 5px;
}
```

```
.login-popup p span{
    color : #FF4C24;
    font-weight: 500;
    cursor : pointer;
}
```

➢ Loginpopup.jsx :

```
import React, { useContext, useState } from 'react'
import './LoginPopup.css'
import { assets } from '../../assets/assets'
import { StoreContext } from '../../Context/StoreContext'
import axios from 'axios'
import { toast } from 'react-toastify'

const LoginPopup = ({ setShowLogin }) => {
```

```
    const { setToken, url,loadCartData } = useContext(StoreContext)
    const [currState, setCurrState] = useState("Sign Up");
```

```
    return (
        <div className='login-popup '>
            <form onSubmit={onLogin} className="login-popup-container">
                <div className="login-popup-title">
                    <h2>{currState}</h2> <img onClick={() =>
setShowLogin(false)} src={assets.cross_icon} alt="" />
                </div>
                <div className="login-popup-inputs">
                    {currState === "Sign Up" ? <input name='name'
onChange={onChangeHandler} value={data.name} type="text"
placeholder='Your name' required /> : <></>}
                    <input name='email' onChange={onChangeHandler}
value={data.email} type="email" placeholder='Your email' />
                    <input name='password' onChange={onChangeHandler}
value={data.password} type="password" placeholder='Password' required
/>
                </div>
                <button>{currState === "Login" ? "Login" : "Create
account"}</button>
                <div className="login-popup-condition">
                    <input type="checkbox" name="" id="" required/>
                    <p>By continuing, i agree to the terms of use &
privacy policy.</p>
                </div>
                {currState === "Login"
```

```
        let new_url = url;
        if (currState === "Login") {
            new_url += "/api/user/login";
        }
        else {
            new_url += "/api/user/register"
        }
        const response = await axios.post(new_url, data);
        if (response.data.success) {
            setToken(response.data.token)
            localStorage.setItem("token", response.data.token)
            loadCartData({token:response.data.token})
            setShowLogin(false)
        }
        else {
            toast.error(response.data.message)
```

```
                    ? <p>Create a new account? <span onClick={() =>
setCurrState('Sign Up')}>Click here</span></p>
                    : <p>Already have an account? <span onClick={() =>
setCurrState('Login')}>Login here</span></p>
                }
            </form>
        </div>
    )
}
```

- **NavBar.css :**

```
import React, { useContext, useState } from 'react'
import './LoginPopup.css'
import { assets } from '../../assets/assets'
import { StoreContext } from '../../Context/StoreContext'
import axios from 'axios'
import { toast } from 'react-toastify'

const LoginPopup = ({ setShowLogin }) => {

    const { setToken, url,loadCartData } = useContext(StoreContext)
    const [currState, setCurrState] = useState("Sign Up");

    const [data, setData] = useState({
        name: "",
        email: "",
        password:
        ""
    })

    const onChangeHandler = (event) =>
        { const name =
        event.target.name
        const value = event.target.value
        setData(data => ({ ...data, [name]: value }))
    }

    const onLogin = async (e) => {
        e.preventDefault()

        let new_url = url;
        if (currState === "Login") {
            new_url += "/api/user/login";
        }
        else {
```

```
        const response = await axios.post(new_url, data);
        if (response.data.success) {
            setToken(response.data.token)
            localStorage.setItem("token", response.data.token)
            loadCartData({token:response.data.token})
            setShowLogin(false)
        }
        else {
            toast.error(response.data.message)
        }
    }
```

```
    return (
        <div className='login-popup'>
            <form onSubmit={onLogin} className="login-popup-container">
                <div className="login-popup-title">
                    <h2>{currState}</h2> <img onClick={() =>
setShowLogin(false)} src={assets.cross_icon} alt="" />
                </div>
                <div className="login-popup-inputs">
                    {currState === "Sign Up" ? <input name='name'
onChange={onChangeHandler} value={data.name} type="text"
placeholder='Your name' required /> : <></>}
                    <input name='email' onChange={onChangeHandler}
value={data.email} type="email" placeholder='Your email' />
                    <input name='password' onChange={onChangeHandler}
value={data.password} type="password" placeholder='Password' required
/>
                </div>
                <button>{currState === "Login" ? "Login" : "Create
account"}</button>
                <div className="login-popup-condition">
                    <input type="checkbox" name="" id="" required/>
                    <p>By continuing, i agree to the terms of use &
privacy policy.</p>
                </div>
                {currState === "Login"
                    ? <p>Create a new account? <span onClick={() =>
setCurrState('Sign Up')}>Click here</span></p>
                    : <p>Already have an account? <span onClick={() =>
setCurrState('Login')}>Login here</span></p>
                }
            </form>
        </div>
    )
}
```

```
              <hr />
              <li onClick={logout}> <img src={assets.logout_icon}
alt="" /> <p>Logout</p></li>
            </ul>
          </div>
        }
```

```
      </div>
    </div>
  )
}
```

```
export default Navbar
```

➢ StoreContext.jsx :

```
import React, { useContext, useEffect, useState } from 'react'
import './Navbar.css'
import { assets } from '../../assets/assets'
import { Link, useNavigate } from 'react-router-dom'
import { StoreContext } from '../../Context/StoreContext'

const Navbar = ({ setShowLogin }) => {
```

```
  const [menu, setMenu] = useState("home");
  const { getTotalCartAmount, token ,setToken } =
useContext(StoreContext);
  const navigate = useNavigate();
```

```
  const logout = () => {
    localStorage.removeItem("token");
    setToken("");
    navigate('/')
  }
```

```
  return (
    <div className='navbar'>
      <Link to='/'><img className='logo' src={assets.logo} alt=""
/></Link>
      <ul className="navbar-menu">
        <Link to="/" onClick={() => setMenu("home")} className={`${menu
=== "home" ? "active" : ""}`}>home</Link>
        <a href='#explore-menu' onClick={() => setMenu("menu")}
className={`${menu === "menu" ? "active" : ""}`}>menu</a>
        <a href='#app-download' onClick={() => setMenu("mob-app")}
className={`${menu === "mob-app" ? "active" : ""}`}>mobile app</a>
```

```
        <a href='#footer' onClick={() => setMenu("contact")}
className={`${menu === "contact" ? "active" : ""}`}>contact us</a>
      </ul>
      <div className="navbar-right">
        <img src={assets.search_icon} alt="" />
        <Link to='/cart' className='navbar-search-icon'>
          <img src={assets.basket_icon} alt="" />
          <div className={getTotalCartAmount() > 0 ? "dot" : ""}></div>
        </Link>
        { !token ? <button onClick={() => setShowLogin(true)}>sign
in</button>
          : <div className='navbar-profile'>
            <img src={assets.profile_icon} alt="" />
            <ul className='navbar-profile-dropdown'>
              <li onClick={()=>navigate('/myorders')}> <img
src={assets.bag_icon} alt="" /> <p>Orders</p></li>
              <hr />
              <li onClick={logout}> <img src={assets.logout_icon}
alt="" /> <p>Logout</p></li>
            </ul>
          </div>
        }
```

```
      </div>
    </div>
  )
}
```

```
export default Navbar
```

➢ App.jsx :

```
import React, { useState } from 'react'
import Home from './pages/Home/Home'
import Footer from './components/Footer/Footer'
import Navbar from './components/Navbar/Navbar'
import { Route, Routes } from 'react-router-dom'
import Cart from './pages/Cart/Cart'
import LoginPopup from './components/LoginPopup/LoginPopup'
import PlaceOrder from './pages/PlaceOrder/PlaceOrder'
import MyOrders from './pages/MyOrders/MyOrders'
import { ToastContainer } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import Verify from './pages/Verify/Verify'

const App = () => {
```

```
  const [showLogin,setShowLogin] = useState(false);
```

```
  return
    ( <>
    <ToastContainer/>
    {showLogin?<LoginPopup setShowLogin={setShowLogin}/>:<></>}
      <div className='app'>
        <Navbar setShowLogin={setShowLogin}/>
        <Routes>
          <Route path='/' element={<Home />}/>
          <Route path='/cart' element={<Cart />}/>
          <Route path='/order' element={<PlaceOrder />}/>
          <Route path='/myorders' element={<MyOrders />}/>
          <Route path='/verify' element={<Verify />}/>
        </Routes>
      </div>
      <Footer />
    </>
  )
}
```

```
export default App
```

➢ index.css :

```
@import
url('https://fonts.googleapis.com/css2?family=Outfit:wght@300;400;500;6
00;700&display=swap');

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Outfit', sans-serif;
  scroll-behavior: smooth;
}
```

■ 9.3 BACKEND :

➢ cartController.js :

```
import userModel from "../models/userModel.js"

// add to user cart
const addToCart = async (req, res) => {
   try {
      let userData = await userModel.findOne({_id:req.body.userId});
      let cartData = await userData.cartData;
      if (!cartData[req.body.itemId]) {
         cartData[req.body.itemId] = 1;
      }
      else {
         cartData[req.body.itemId] += 1;
      }
      await userModel.findByIdAndUpdate(req.body.userId, {cartData});
      res.json({ success: true, message: "Added To Cart" });
   } catch (error) {
      console.log(error);
      res.json({ success: false, message: "Error" })
   }
}
```

```
// remove food from user cart
const removeFromCart = async (req, res) => {
   try {
      let userData = await userModel.findById(req.body.userId);
      let cartData = await userData.cartData;
      if (cartData[req.body.itemId] > 0) {
         cartData[req.body.itemId] -= 1;
      }
      await userModel.findByIdAndUpdate(req.body.userId, {cartData});
      res.json({ success: true, message: "Removed From Cart" });
   } catch (error) {
      console.log(error);
      res.json({ success: false, message: "Error" })
   }
```

```
}
```

```
// get user cart
const getCart = async (req, res) => {
   try {
      let userData = await userModel.findById(req.body.userId);
      let cartData = await userData.cartData;
      res.json({ success: true, cartData:cartData });
   } catch (error) {
      console.log(error);
      res.json({ success: false, message: "Error" })
   }
}
```

```
import foodModel from "../models/foodModel.js";
import fs from 'fs'

// all food list
const listFood = async (req, res) => {
    try {
        const foods = await foodModel.find({})
        res.json({ success: true, data: foods })
    } catch (error) {
        console.log(error);
        res.json({ success: false, message: "Error" })
    }

}

// add food
const addFood = async (req, res) => {

    try {
        let image_filename = `${req.file.filename}`

        const food = new foodModel({
            name: req.body.name,
            description: req.body.description,
            price: req.body.price,
            category:req.body.category,
            image: image_filename,
        })

        await food.save();
        res.json({ success: true, message: "Food Added" })
    } catch (error) {
```

```
        console.log(error);
        res.json({ success: false, message: "Error" })
    }
}
```

```
// delete food
const removeFood = async (req, res) => {
    try {
```

```
        const food = await foodModel.findById(req.body.id);
        fs.unlink(`uploads/${food.image}`, () => { })
```

```
        await foodModel.findByIdAndDelete(req.body.id)
        res.json({ success: true, message: "Food Removed" })
```

```
    } catch (error) {
        console.log(error);
        res.json({ success: false, message: "Error" })
    }
```

```
}
```

```
export { listFood, addFood, removeFood }
```

➢ OrderController.js :

```
import orderModel from "../models/orderModel.js";
import userModel from "../models/userModel.js"
import Stripe from "stripe";
const stripe = new Stripe(process.env.STRIPE_SECRET_KEY);

//config variables
const currency = "inr";
const deliveryCharge = 50;
const frontend_URL = 'http://localhost:5173';
```

```
// Placing User Order for Frontend using stripe
const placeOrder = async (req, res) => {
```

➢ Authentication.js :

```
import jwt from 'jsonwebtoken';

const authMiddleware = async (req, res, next) => {
    const { token } = req.headers;
    if (!token) {
        return res.json({success:false,message: 'Not Authorized Login
Again'});
    }
    try {
        const token_decode =  jwt.verify(token, process.env.JWT_SECRET);
        req.body.userId = token_decode.id;
        next();
    } catch (error) {
        return res.json({success:false,message:error.message});
    }
}
```

```
export default authMiddleware;
```

➢ Package.json :

```
{
  "name": "food-del-backend",
  "version": "1.0.0",
  "description": "",
  "type": "module",
  "main": "server.js",
  "scripts": {
    "server": "nodemon server.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.1.1",
    "body-parser":
    "^1.20.2", "cors":
    "^2.8.5",
    "dotenv": "^16.4.1",
    "express": "^4.18.2",
    "jsonwebtoken":
    "^9.0.2", "mongoose":
    "^8.1.1",
    "multer": "^1.4.5-
    lts.1", "nodemon":
```

```
  }
}
```

➢ Server.js :

```
import express  from "express"
import cors from  'cors'
import { connectDB } from "./config/db.js"
import userRouter from "./routes/userRoute.js"
import foodRouter from "./routes/foodRoute.js"
import 'dotenv/config'
import cartRouter from "./routes/cartRoute.js"
import orderRouter from "./routes/orderRoute.js"

// app config
const app = express()
const port = process.env.PORT || 4000;
```

```
// middlewares
app.use(express.json())
app.use(cors())
```

```
// db connection
connectDB()
```

```
// api endpoints
app.use("/api/user", userRouter)
app.use("/api/food", foodRouter)
app.use("/images",express.static('uploads'))
app.use("/api/cart", cartRouter)
app.use("/api/order",orderRouter)
```

```
app.get("/", (req, res) => {
    res.send("API Working")
  });
```

```
app.listen(port, () => console.log(`Server started on
http://localhost:${port}`))
```

10. API DOCUMENTATION

1. User Registration:

- User sends a request to register with their details (name, email, password, etc.).
- The system validates the input data.
- If validation passes, a new user account is created.
- The system responds with a success message and user ID.

---

2. User Login:

- User sends login credentials (email and password).
- The system verifies the credentials.
- If valid, a token (JWT or similar) is generated and returned.
- User uses the token for authenticated requests.

---

3. Get User Profile:

- User sends a request with the authentication token.
- The system verifies the token and retrieves user details.
- The system returns the user's profile information.

Admin API Steps :

1. Admin Login:

- Admin provides credentials to log in.
- The system verifies the credentials.
- If valid, a token is issued for admin-level access.

---

2. View All Users:

- Admin sends a request with their authentication token.
- The system verifies the token and fetches all user records.
- The system returns the list of users.

---

3. Delete a User:

- Admin sends a request to delete a specific user, providing the user ID.
- The system verifies the admin token and checks if the user exists.
- If valid, the user is deleted, and the system responds with a confirmation.

4. Update User Role:

- Admin sends a request to update a user's role, specifying the user ID and new role.
- The system verifies the admin token and checks the validity of the new role.
- If valid, the user's role is updated, and the system responds with a success message.

11. AUTHENTICATION MECHANISM

1.User/Admin Registration:

- Collect user/admin details (e.g., email, password) and store the password securely after hashing.

2.Login:

- User/admin provides email and password for authentication.

3.Credential Verification:

- Validate the email and password by comparing the provided password (hashed) with the stored hash.

4.Token Generation:

- On successful login, generate a secure token (e.g., JWT) containing user/admin details and expiry.

5.Token Usage:

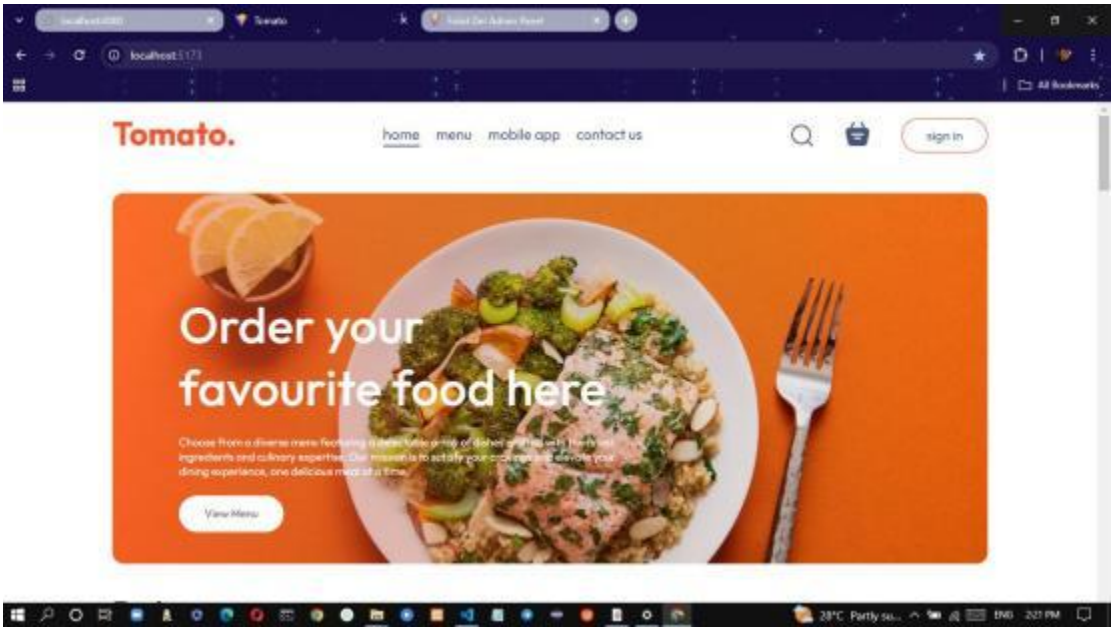
- Client includes the token in the **Authorization header** for subsequent requests.

6.Token Verification:

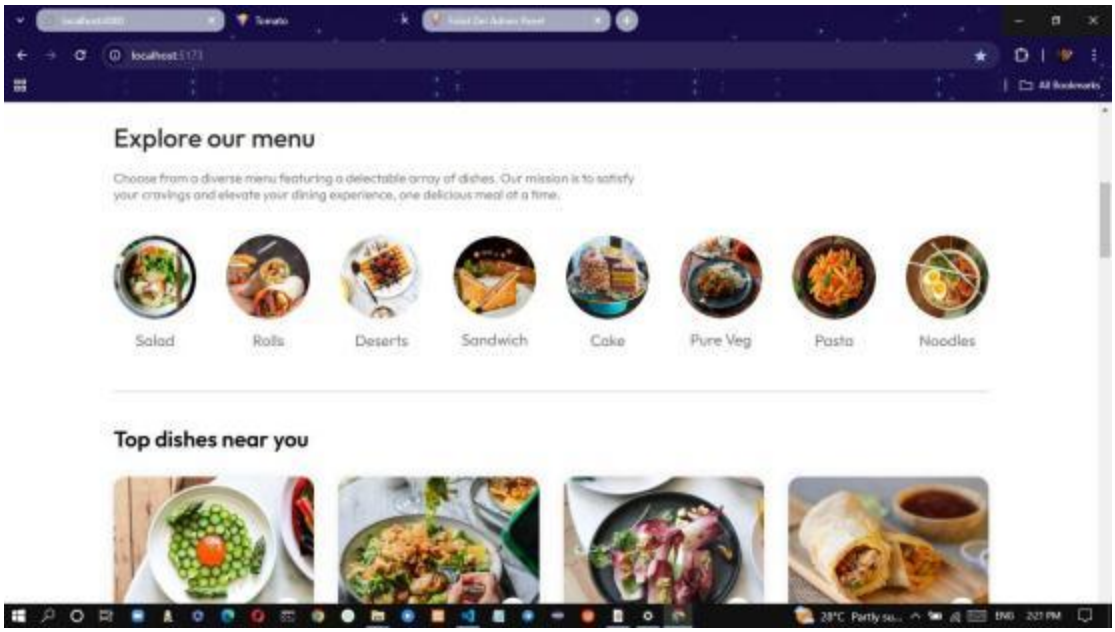
- Server verifies the token, checks user/admin permissions, and processes the request if authorized.

12.USER INTERFACE

LANDING PAGE :



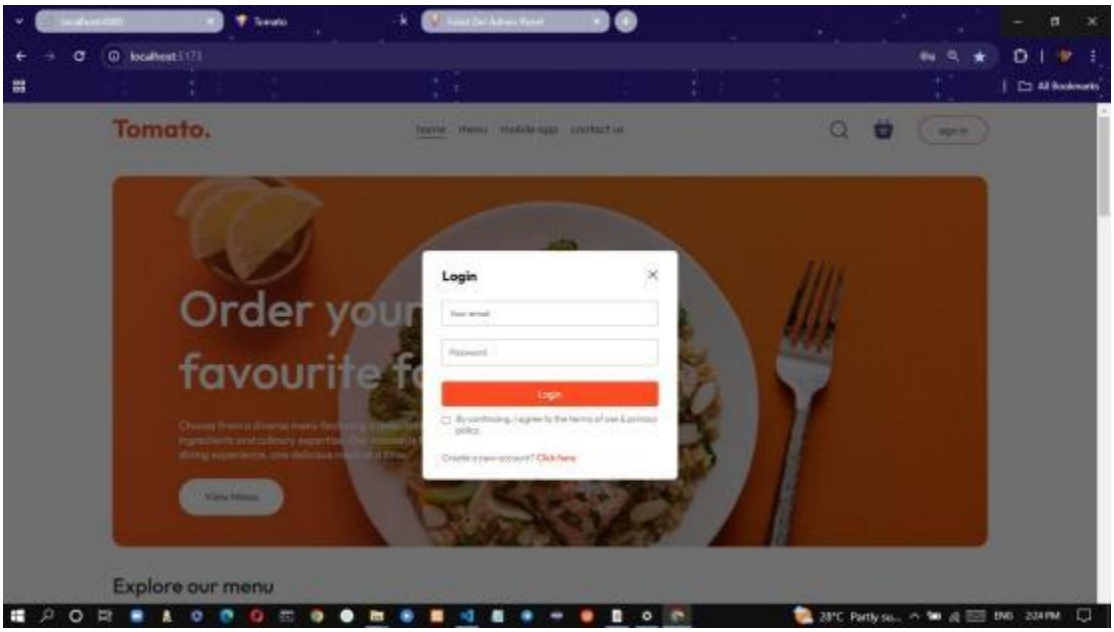
RESTAURANT MENU PAGE:



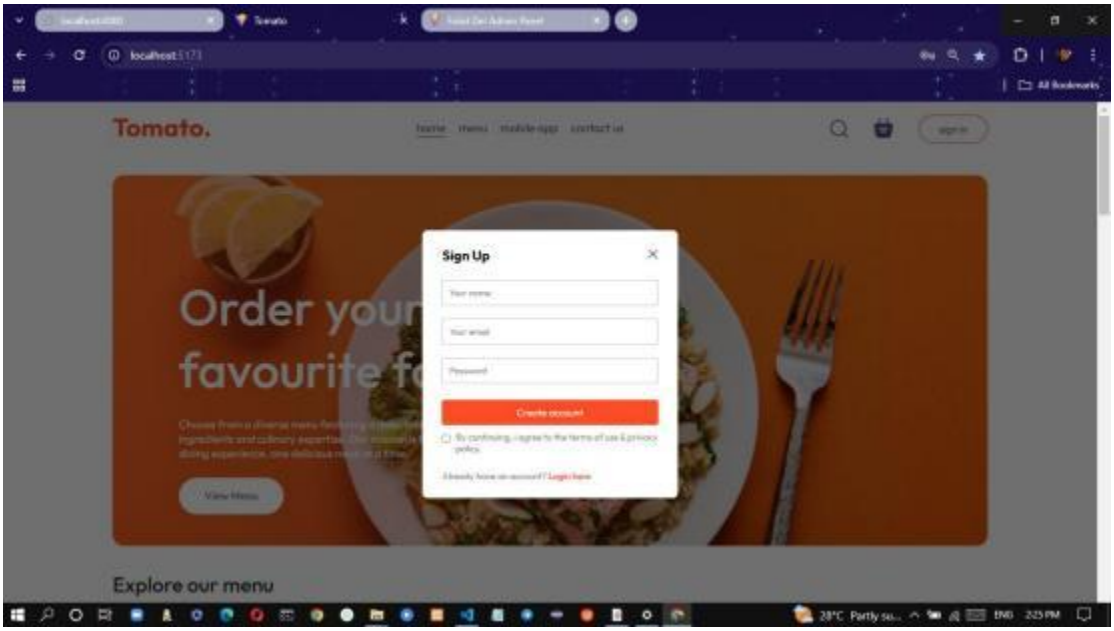
CONTACT DETAILS PAGE:

LOGIN AND SIGNUP PAGES :

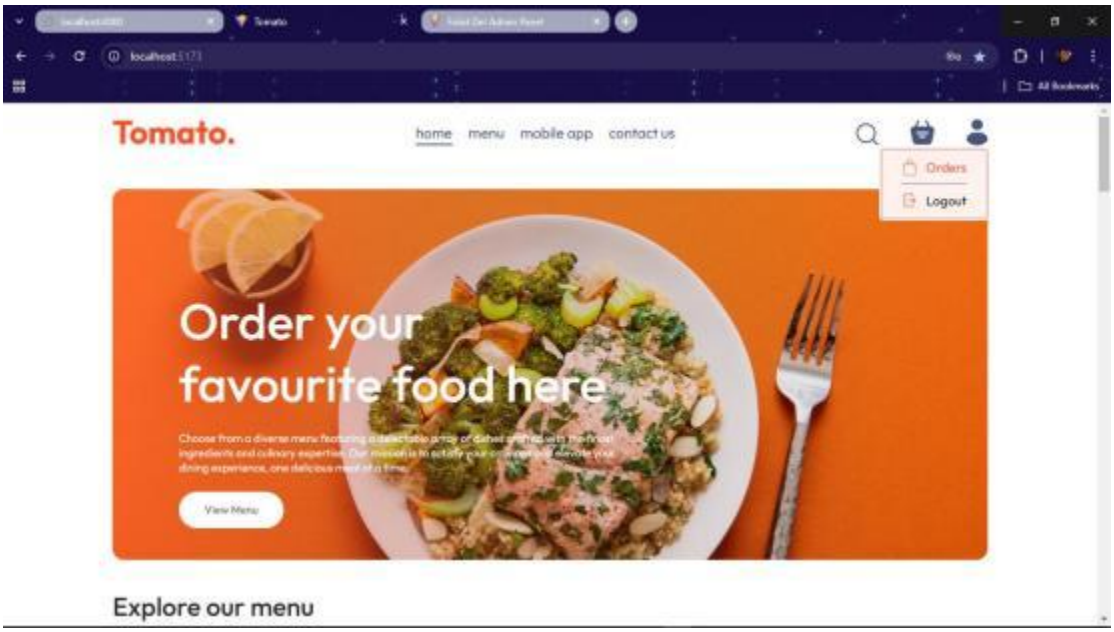
LOGIN PAGE :



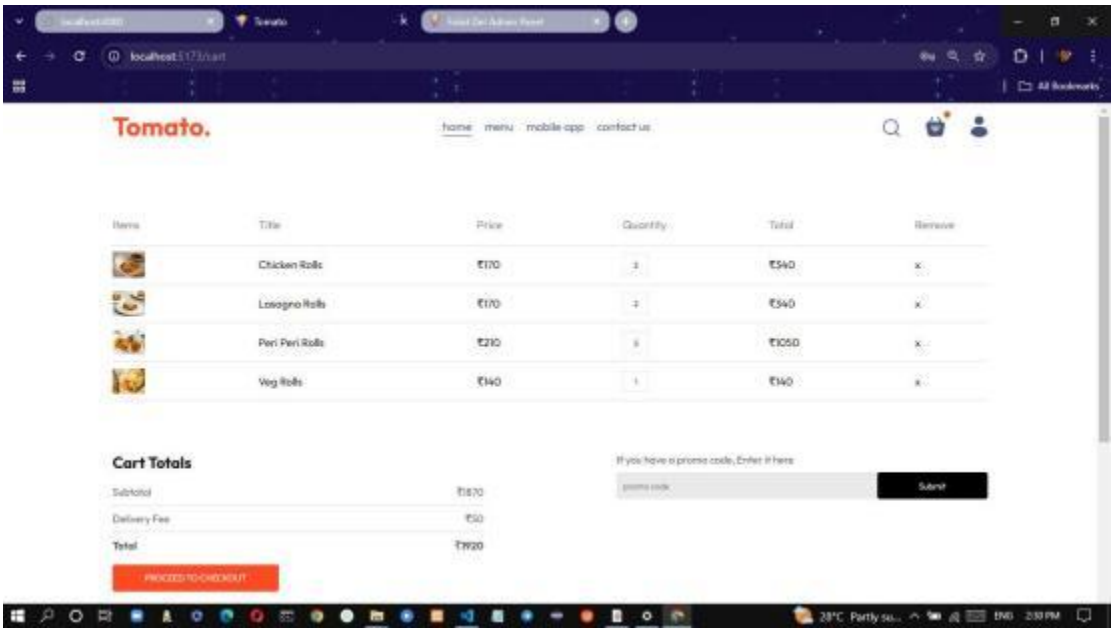
SIGNUP PAGE :



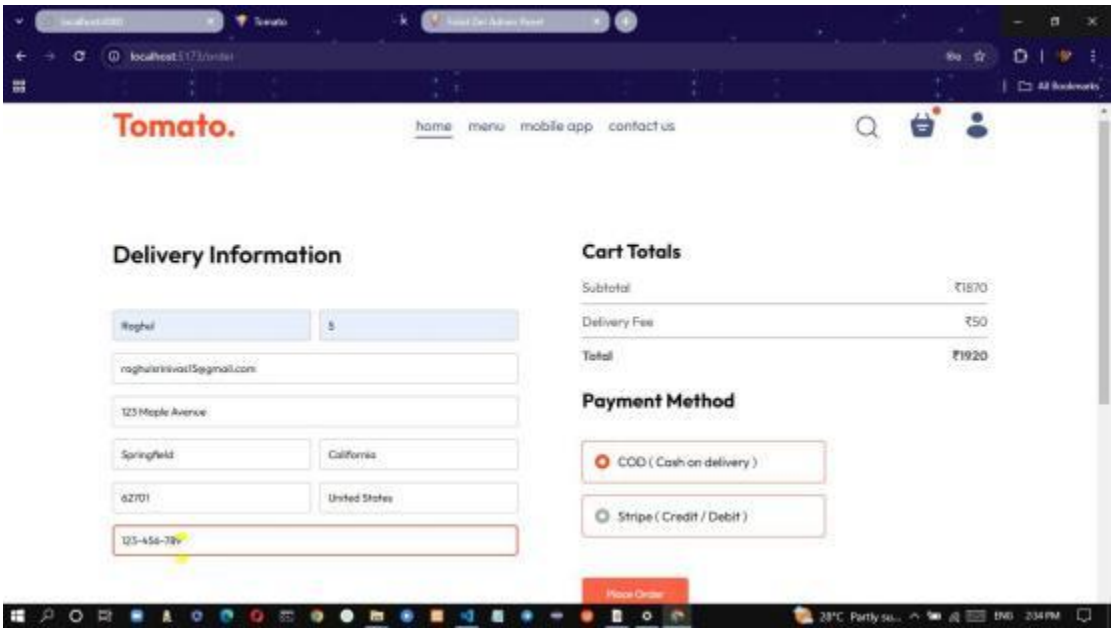
USER PROFILE PAGE:



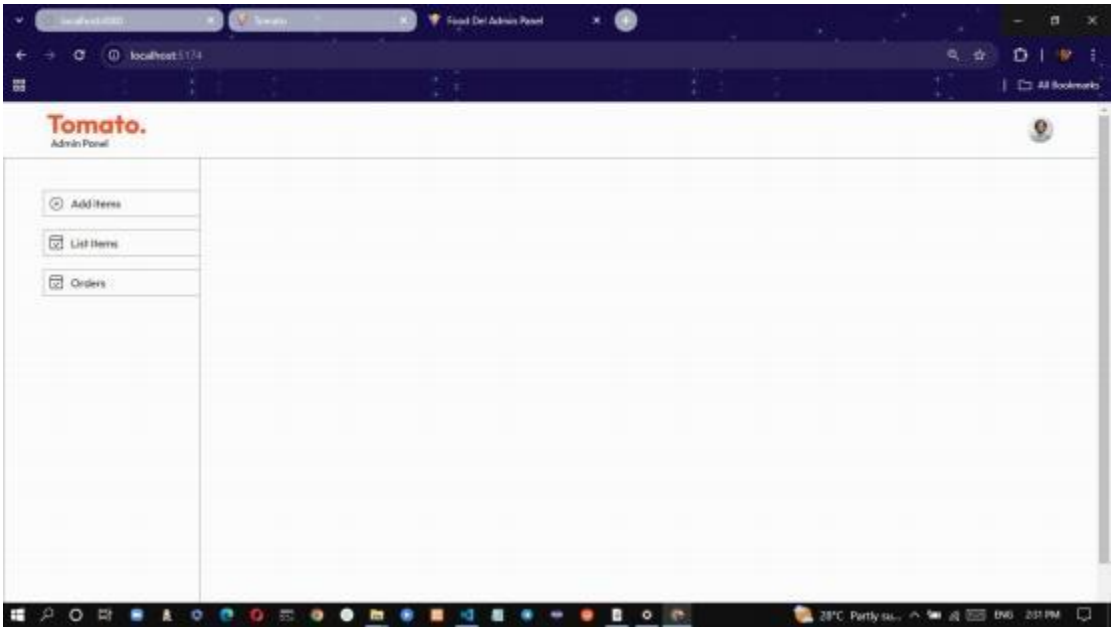
CART ITEMS PAGE :



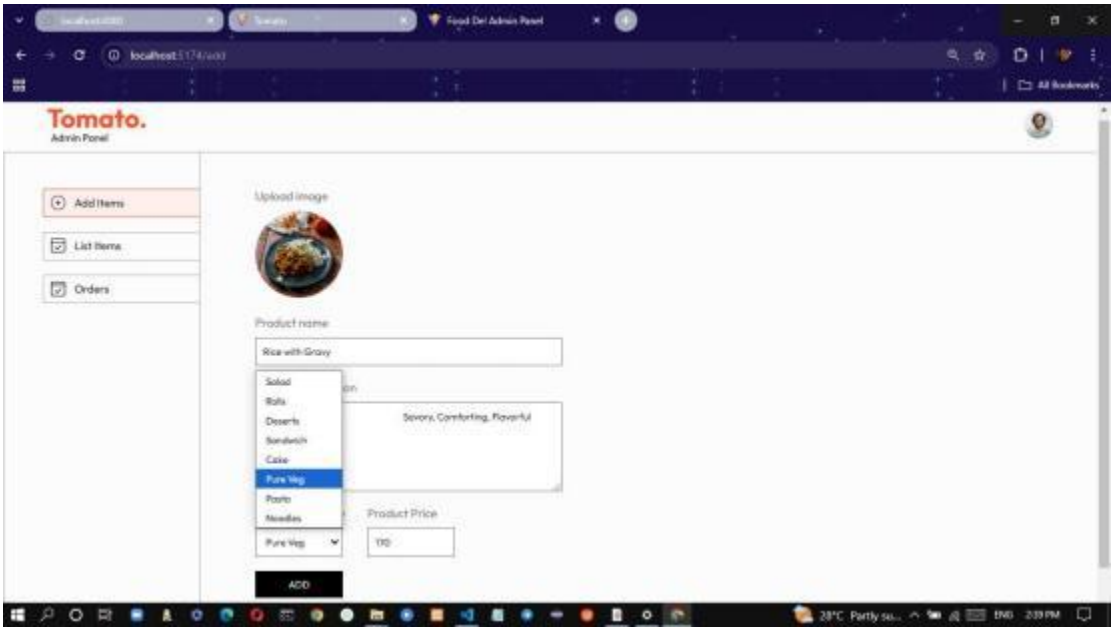
ADDRESS AND PAYMENT METHOD PAGE:



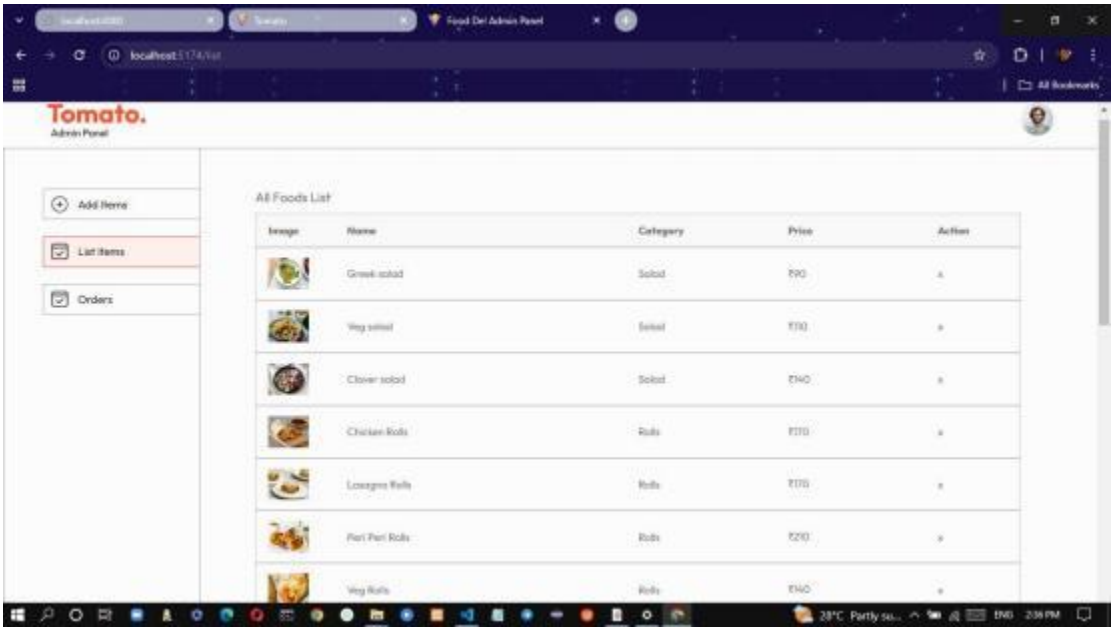
ADMIN DASHBOARD PAGE :



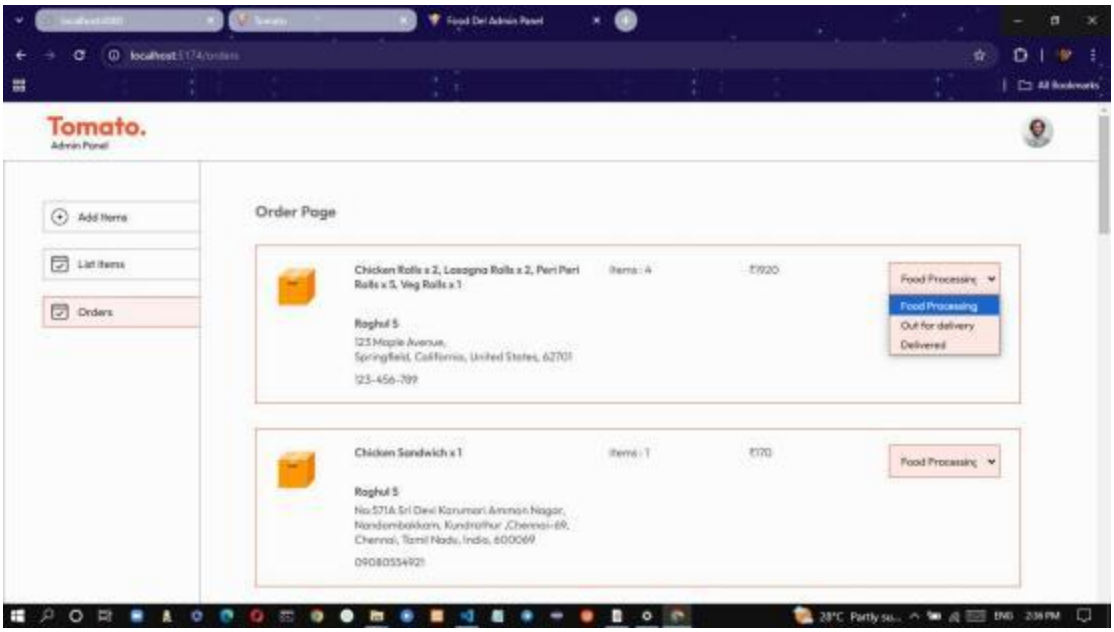
ADDING NEW ITEMS PAGE :



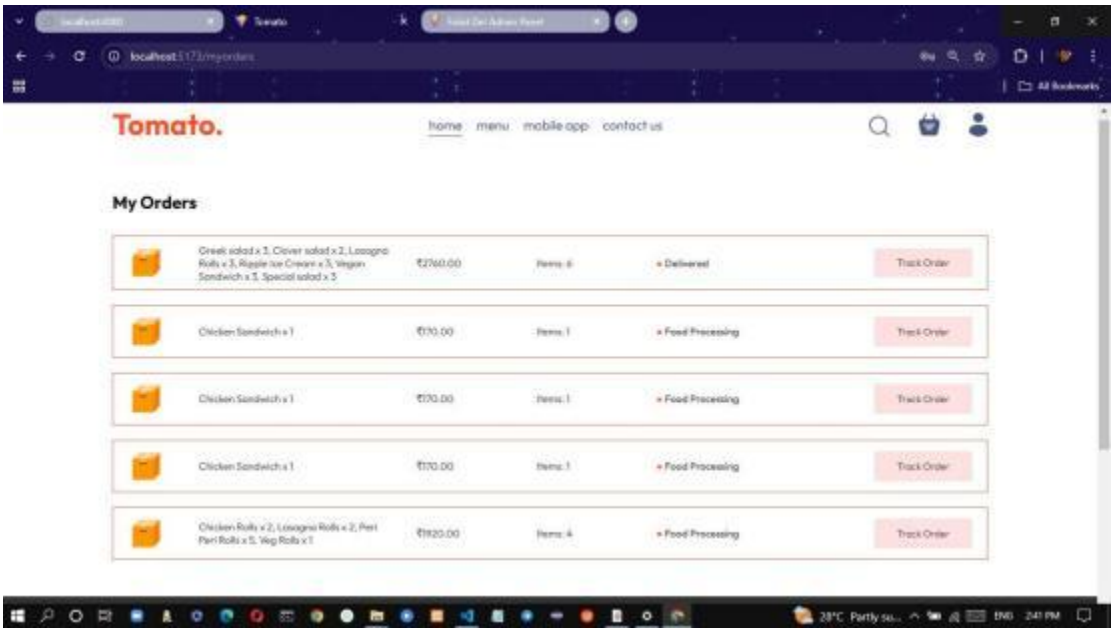
LIST ITEMS PAGE:



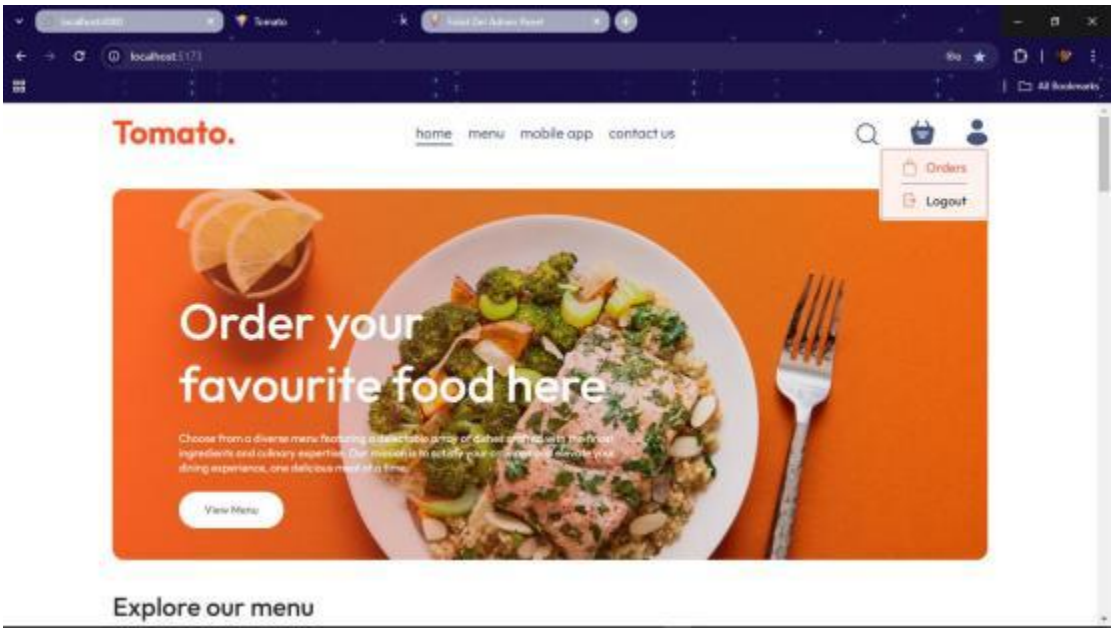
ORDERS PAGE :



USERS CAN ALSO TRACK THE ORDERS :



LOGOUT PAGE :



13.TESTINGSTRATEGY

1.FunctionalTesting:

- **Purpose:** Verify that all core features, like order placement, payment, and item selection, work as expected.
- **Tools:** Selenium, Postman (for API testing), Cypress.

2.SecurityTesting:

- **Purpose:** Check for vulnerabilities like SQL injection, XSS, and ensure secure handling of sensitive data (e.g., passwords and payment details).
- **Tools:** OWASP ZAP, Burp Suite, Acunetix.

3.PerformanceTesting:

- **Purpose:** Test how the system performs under heavy traffic, checking for slowdowns or crashes.
- **Tools:** Apache JMeter, LoadRunner, Gatling.

14. DEMO VEDIO LINK

https://drive.google.com/file/d/1yeZ85Jq7kGxlGPz9tFNGbHd8K_P9G_fC/view?_usp_=_drivesdk

GTHUB REP0:

https://github.com/RaghulExtremze/Food_ordering_web.git

15. KNOWN ISSUES

1. Order processing Errors:

- Incorrect status updates or failed order processing due to transaction issues.

2. Payment Gateway Integration:

- Payment failures or lack of transaction confirmation after successful payment.

3. Database & Inventory Management:

- Incorrect stock levels or delayed queries affecting item availability.

16. FUTURE ENHANCEMENTS

1. Mobile APP Integration:

- **Enhancement :** Develop a mobile application for iOS and Android to reach a broader audience and improve user experience
- **Tools :** React Native, Flutter, Xamarin

2. AI and personalization:

- **Enhancement:** Implement AI-driven features like personalized recommendations, predictive text for faster ordering, and user behavior analysis for targeted promotions.
- **Tools :** TensorFlow, Python, Google Cloud AI, Firebase ML

3. Real-time order Tracking:

- **Enhancement:** Add real-time order tracking so customers can monitor their order status and delivery progress.
- **Tools :** Firebase Realtime Database, WebSockets, Google Maps AP

17.CONCLUSION

The development of the Food Ordering Website using the MERN stack (MongoDB, Express.js, React.js , Node.js) provided an excellent opportunity to explore full-stack web development and integrate cutting-edge technologies. This project demonstrates how efficient and user-friendly web solutions can streamline the food ordering process, catering to the dynamic needs of modern consumers.

The use of MongoDB ensured robust data management, while Express.js and Node.js facilitated seamless backend operations. React.js enhanced the user interface, providing an interactive and engaging experience. Overall, the project successfully achieved its goals of creating a reliable, scalable, and responsive platform for online food ordering.

Furthermore, the project highlighted the importance of implementing secure payment gateways and efficient database management to ensure a seamless experience for both users and administrators. By adhering to best practices in coding and project development, the website maintains a high standard of performance and security, addressing the critical aspects of user data protection and transaction safety.

Looking ahead, this project opens up opportunities for further enhancements, such as incorporating machine learning for personalized food recommendations, advanced analytics for business insights, and expanding the platform's scalability to accommodate a larger user base.

18.REFERENCES

1.Smith, J. (2020). Web Development with React and Node.js. Packt Publishing.A comprehensive guide to building full-stack web applications using the MERN stack.

2.MongoDB Documentation. (n.d.). MongoDB: NoSQL Database. Retrieved from https://www.mongodb.com/docs Official documentation for MongoDB, explaining its setup, queries, and optimization techniques.

3.Express.js Guide. (n.d.). Express.js Documentation. Retrieved from https://expressjs.com Provides insights into building server -side logic for web applications.

4.React Official Documentation. (n.d.). React: A JavaScript Library for Building User Interfaces. Retrieved from https://reactjs.org Covers React concepts, best practices, and component-based architecture.

5.Node.js Foundation. (n.d.). Node.js Documentation. Retrieved from https://nodejs.org/en/docs Official Node.js documentation for developing scalable and high-performance backend services.

6.Stripe API Documentation. (n.d.). Online Payment Processing. Retrieved from https://stripe.com/docs Details on integrating secure payment systems into web applications.

7.W3Schools. (n.d.). HTML, CSS, and JavaScript Tutorials. Retrieved from https://www.w3schools.com A beginner-friendly resource for learning web development technologies.