Full Stack Development with MERN
Project Documentation format

1. Introduction:

Project Title: SB Food Delivery app

Team Members:

1. Janani. R
2. B. Lokesh Kumar
3. C. Santhosh
4. M. Dhivakar

2. Project Overview:

purpose:

The purpose of a food delivery app is to provide a convenient and efficient platform for users to order food from local restaurants and have it delivered directly to their location.

Features:

1. Enhance Convenience: Make it easy for users to browse menus, place orders, and receive food without leaving their homes or offices.
2. Improve Accessibility: Offer a wide selection of restaurants and cuisines, allowing users to explore different food options based on their preferences.
3. Streamline Delivery: Ensure quick and reliable food delivery through a user-friendly interface that tracks orders in real-time.
4. Support Local Businesses: Connect restaurants with a broader customer base, helping them increase sales and visibility.
5. Provide Personalization: Offer recommendations, save preferences, and offer promotions or loyalty rewards based on user behaviour and ordering habits.

3. Architecture:

Frontend: Building the frontend architecture for a food delivery app using React requires a solid and scalable design to ensure a seamless user experience, and Below is a breakdown of how you could structure the frontend architecture.

Components:

1. Footer: It is an essential element that shares the details of the food in the app.
2. Login: This feature is a critical part of any food delivery app, allowing users to access personalized services like order history, preferences, and saved payment methods.
3. Navbar: A navigation bar is a key element in any web app, including a food delivery app, as it allows users to navigate between different sections (home, menu, cart, profile, etc.). It should be easily accessible, intuitive, and mobile-friendly.
4. Popular restaurants: It feature or showcase a list of popular restaurants within an app
5. Register: To implement a(sign-up) feature in a food delivery app using the tool React, you would typically need a form where users can enter their details.
6. Restaurants: feature in a food delivery app, you're likely talking about a listing of available restaurants that users can browse, filter, and interact with. This could include displaying details like restaurant name, ratings, cuisine type, delivery time, price range, and more. Additionally, users should be able to view menus, place orders, and maybe even read reviews.

Backend: Below is an outline of a backend architecture for a food ordering app using **Node.js** and **Express.js**. This outline covers the essential components, including API routes, database structure, authentication, and scaling considerations.

Components:

1. node modules: building an app using Node.js and Express.js, you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.
2. env: building a app using Node.js and Express.js you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.
3. Index.js: building a app using Node.js and Express.js , you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.
4. Package-lock.json: This file is automatically generated when you first install dependencies using npm install. It is crucial for ensuring consistent and deterministic dependency installations across different environments or among different developers on a project.
5. package.json: This file is a crucial part of a Node.js project. It serves as the manifest for your project, defining metadata, dependencies, scripts, and other configuration details for your application.
6. Schema.js:This file typically defines the structure of your data model, especially if you're using an ORM (Object-Relational Mapping) library like (for SQL databases) or (for MongoDB). The schema defines how data is structured in your database and often includes the attributes, data types, relationships, validation rules, and any other constraints on that data.

Database Schema:

In a food ordering application, the following entities are important,

1. User: Represents a customer or admin of the system.
2. MenuItem: Represents a food item on the restaurant's menu.
3. Order: Represents a customer's order, which includes multiple menu items.
4. Payment: Represents a payment for an order.
5. Cart (Optional): Represents items that a user has added to their cart before placing an order.

4.Setup Instruction:

Prerequisites:

- Node.js (v14 or higher)
- MongoDB

Installation:

1.install directory:

```
cd server
npm install
```

2.Run client and server

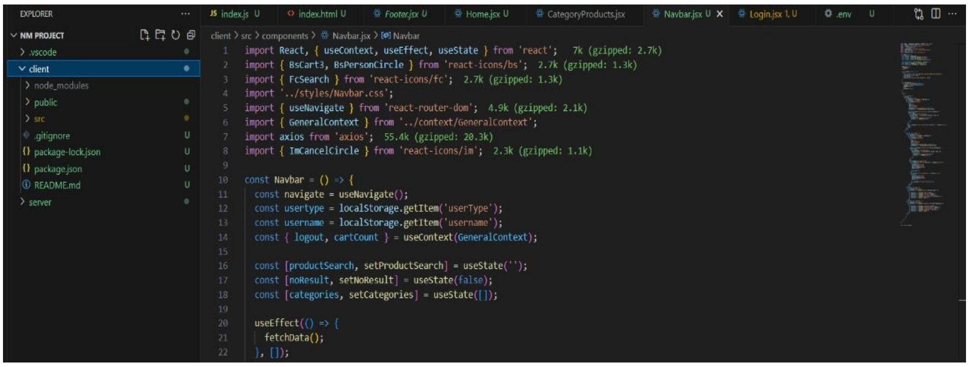
```
cd client
npm start
cd server
node index.js
```

5.Folder Structure:

Client: Building the frontend architecture for a food delivery app using React requires a solid and scalable design to ensure a seamless user experience, and Below is a breakdown of how you could structure the frontend architecture.

Components:

1. Footer: It is an essential element that shares the details of the food in the app.
2. Login: A login feature is a critical part of any food delivery app, allowing users to access personalized services like order history, preferences, and saved payment methods.
3. Navbar: A navbar (navigation bar) is a key element in any web app, including a food delivery app, as it allows users to navigate between different sections (home, menu, cart, profile, etc.). It should be easily accessible, intuitive, and mobile-friendly.
4. Popular restaurants: It feature or showcase a list of popular restaurants within a food delivery app
5. Register: To implement a register (sign-up) feature in a food delivery app using React, you would typically need a form where users can enter their details, such as name, e-mail ,password and perhaps other details like phone number or address.
6. Restaurants: The feature in a food delivery app, you're likely talking about a listing of available restaurants that users can browse, filter, and interact with. This could include displaying details like restaurant name, ratings, cuisine type, delivery time, price range, and more. Additionally, users should be able to view menus, place orders, and maybe even read reviews.



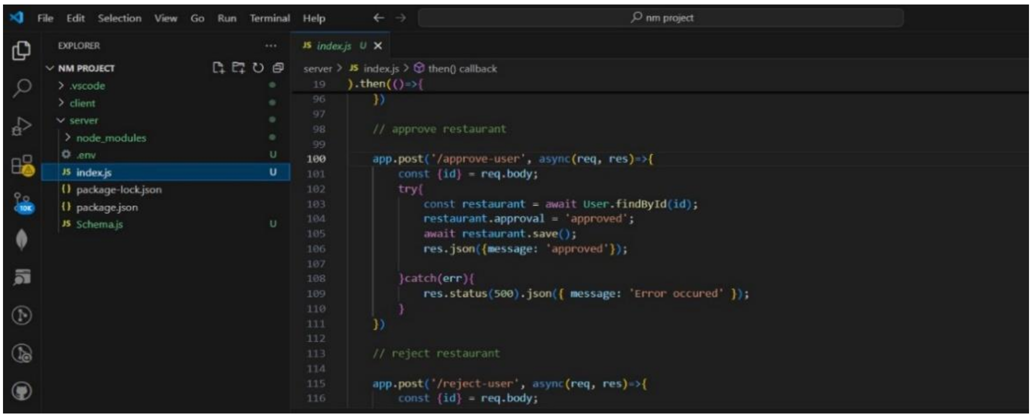
```
import React, { useContext, useEffect, useState } from 'react';   7k (gzipped: 2.7k)
import { BsCart3, BsPersonCircle } from 'react-icons/bs';  2.7k (gzipped: 1.3k)
import { FcSearch } from 'react-icons/fc';  2.7k (gzipped: 1.3k)
import '../styles/Navbar.css';
import { useNavigate } from 'react-router-dom';  4.9k (gzipped: 2.1k)
import { GeneralContext } from '../context/GeneralContext';
import axios from 'axios';  55.4k (gzipped: 20.3k)
import { ImCancelCircle } from 'react-icons/im';  2.3k (gzipped: 1.1k)

const Navbar = () => {
  const navigate = useNavigate();
  const usertype = localStorage.getItem('userType');
  const username = localStorage.getItem('username');
  const { logout, cartCount } = useContext(GeneralContext);

  const [productSearch, setProductSearch] = useState('');
  const [noResult, setNoResult] = useState(false);
  const [categories, setCategories] = useState([]);

  useEffect(() => {
    fetchData();
  }, []);
```

server: Below is an outline of a backend architecture for a food ordering app using Node.js and Express.js. This outline covers the essential components, including API routes, database structure, authentication, and scaling considerations.

Components:

1. node modules: building a food delivery app using Node.js and Express.js, you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.
2. env: building a food delivery app using Node.js and Express.js, you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.
3. Index.js: building a food delivery app using Node.js and Express.js, you'll need several Node.js modules (or packages) to handle various tasks like routing, authentication, database management, real-time communication, and payment integration.

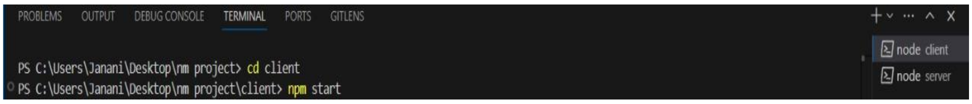
4. Package-lock.json: This file is automatically generated when you first install dependencies using npm install. It is crucial for ensuring consistent and deterministic dependency installations across different environments or among different developers on a project.
5. package.json: This file is a crucial part of a Node.js project. It serves as the manifest for your project, defining metadata, dependencies, scripts, and other configuration details for your application.
6. Schema.js:This file typically defines the structure of your data model, especially if you're using an ORM (Object-Relational Mapping) library like (for SQL databases) or (for MongoDB). The schema defines how data is structured in your database and often includes the attributes, data types, relationships, validation rules, and any other constraints on that data.
7.



```
    ).then(()=>{
        })

        // approve restaurant

        app.post('/approve-user', async(req, res)=>{
            const {id} = req.body;
            try{
                const restaurant = await User.findById(id);
                restaurant.approval = 'approved';
                await restaurant.save();
                res.json({message: 'approved'});

            }catch(err){
                res.status(500).json({ message: 'Error occured' });
            }
        })

        // reject restaurant

        app.post('/reject-user', async(req, res)=>{
            const {id} = req.body;
```

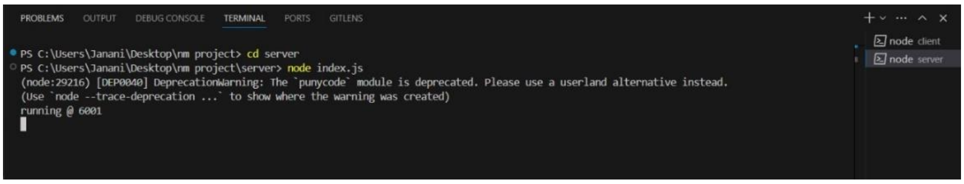
8. Running the Application:

Client:
Command:



```
PS C:\Users\Janani\Desktop\nm project> cd client
PS C:\Users\Janani\Desktop\nm project\client> npm start
```

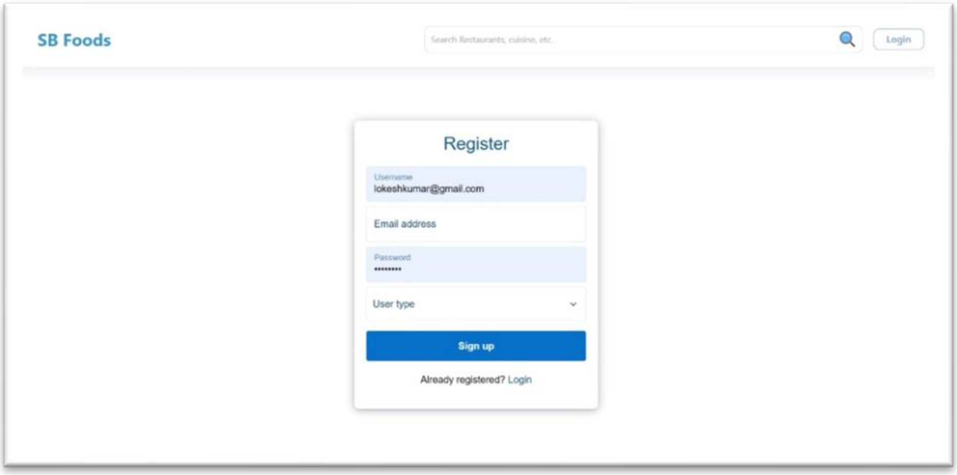
Server:
Command:



```
PS C:\Users\Janani\Desktop\nm project> cd server
PS C:\Users\Janani\Desktop\nm project\server> node index.js
(node:29216) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
running @ 6001
```

7.API DOCUMENTATION:

Register user:



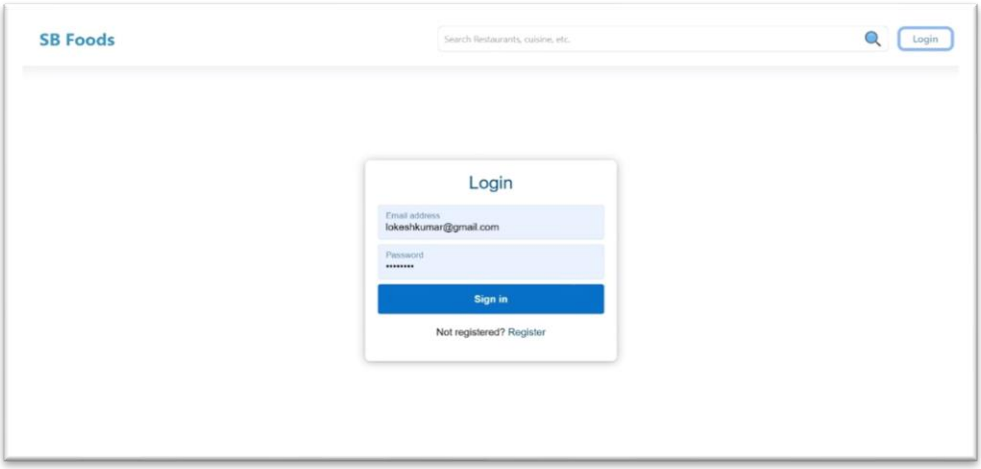
Username
lokeshkumar@gmail.com

Email address

Password
••••••••

User type

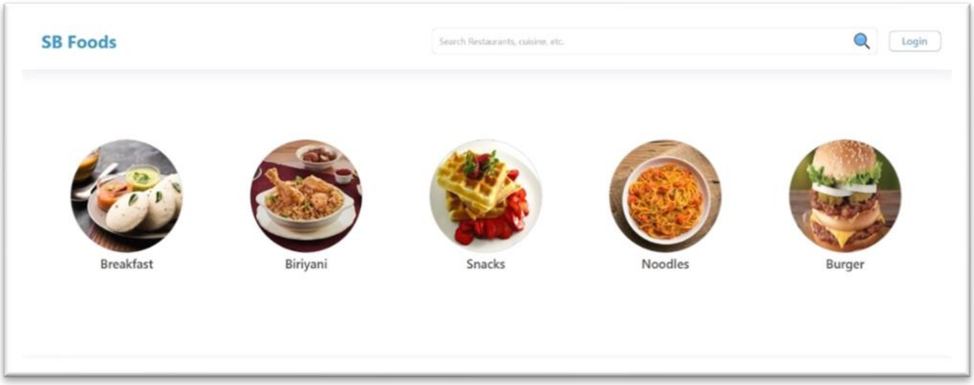
Login:



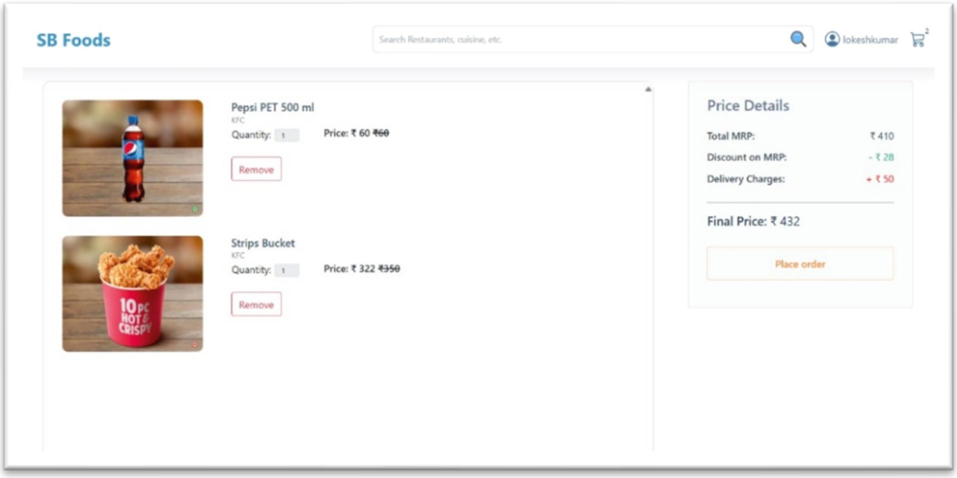
Email address
lokeshkumar@gmail.com

Password
••••••••

Food Menu:



Order:



SB Foods

Search Restaurants, cuisine, etc.

lokeshkumar

Pepsi PET 500 ml
KFC
Quantity: 1 Price: ₹ 60 ~~₹60~~

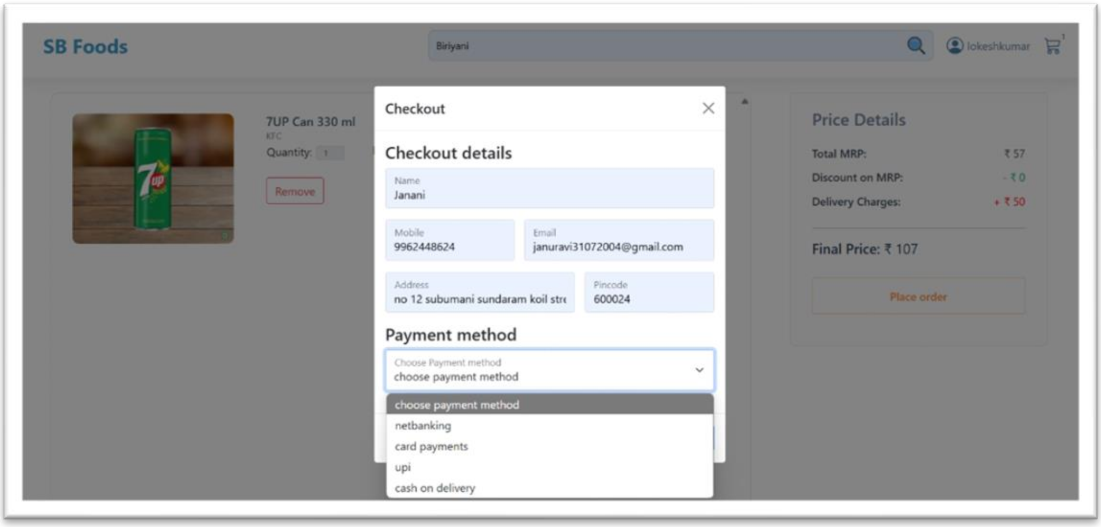
Strips Bucket
KFC
Quantity: 1 Price: ₹ 322 ~~₹350~~

Price Details

Total MRP:	₹ 410
Discount on MRP:	- ₹ 28
Delivery Charges:	+ ₹ 50

Final Price: ₹ 432

.payment:



SB Foods

Biriyani

lokeshkumar

7UP Can 330 ml
KFC
Quantity: 1

Checkout

Checkout details

Name: Janani

Mobile: 9962448624

Email: januravi31072004@gmail.com

Address: no 12 subumani sundaram koil stre

Pincode: 600024

Payment method

Choose Payment method: choose payment method

Price Details

Total MRP:	₹ 57
Discount on MRP:	- ₹ 0
Delivery Charges:	+ ₹ 50

Final Price: ₹ 107

8.Authentication:

Authentication and authorization are crucial for a food delivery app to ensure that users can securely log in, place orders, and manage their accounts, while restricting access to specific features and actions based on user roles (e.g., customer, restaurant owner, delivery personnel, admin). Here's an overview of how these can be implemented:

1. User Authentication

Purpose: To verify that a user is who they claim to be, allowing access to their account in the food delivery app.

- **Registration**:
  - New users (customers, restaurant owners, or delivery personnel) sign up by providing necessary details (e.g., email, password, name).
  - Passwords are hashed using a secure hashing algorithm like bcrypt before being stored in the database.
- **Login**:
  - Users enter their email and password to log in.
  - The app verifies the password by comparing the hashed password stored in the database with the hashed version of the provided password.

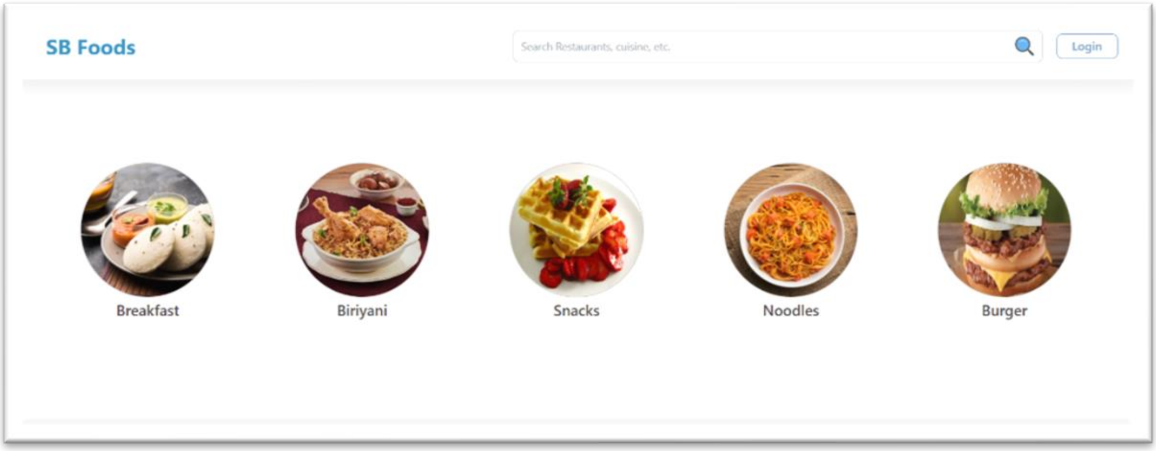
User Authorization

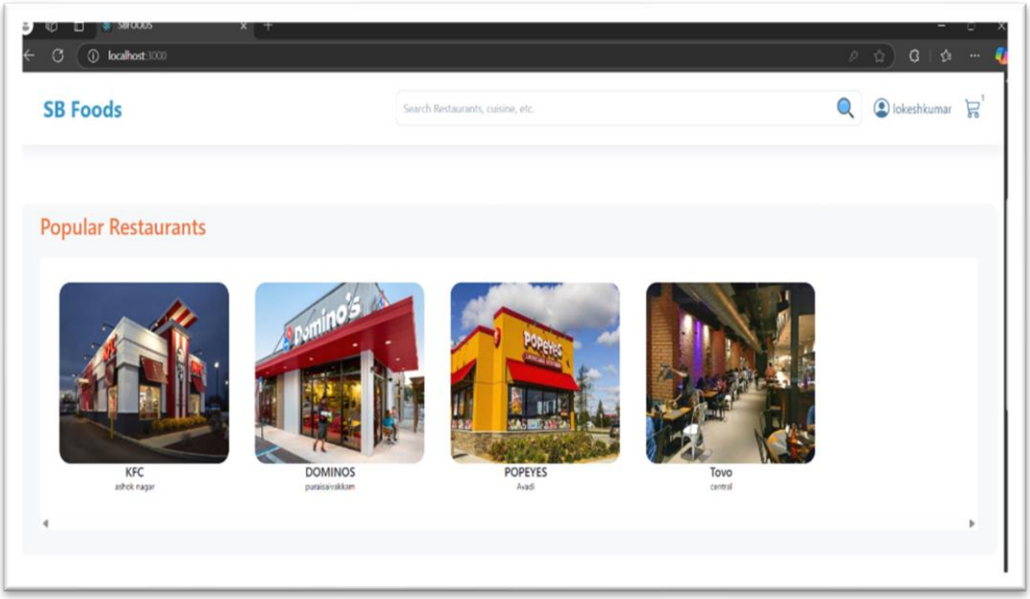
Purpose: To control what actions a user can perform within the app based on their role.

- **Customer**: Can browse food items, place orders, view order history.
- **Restaurant Owner**: Can manage menu items, view and accept orders, and track delivery.
- **Delivery Personnel**: Can view assigned orders, update delivery status.
- **Admin**: Has higher-level control, such as viewing all data and managing users.

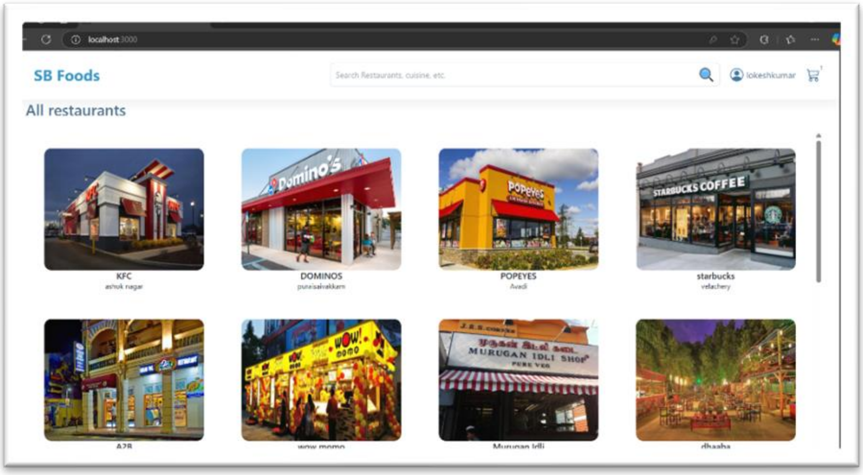
9.User Interface:

Home screen: User interface of the application for the app was displayed here.

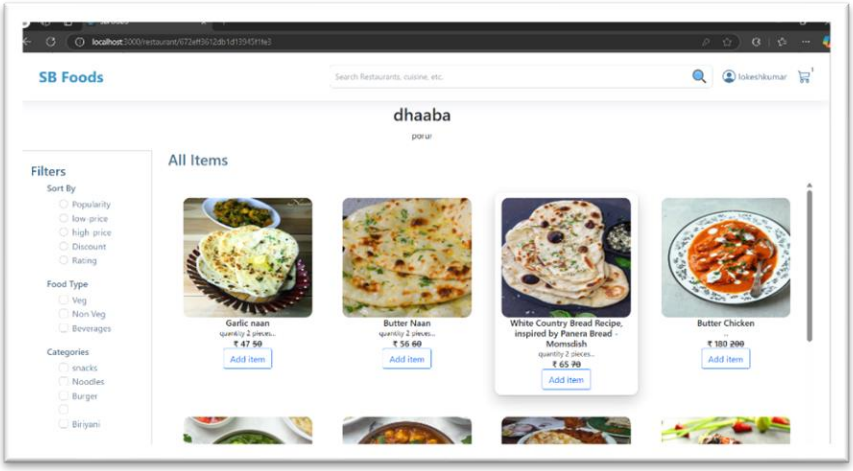




Restaurant Listing Screen:



Restaurant Menu Screen:

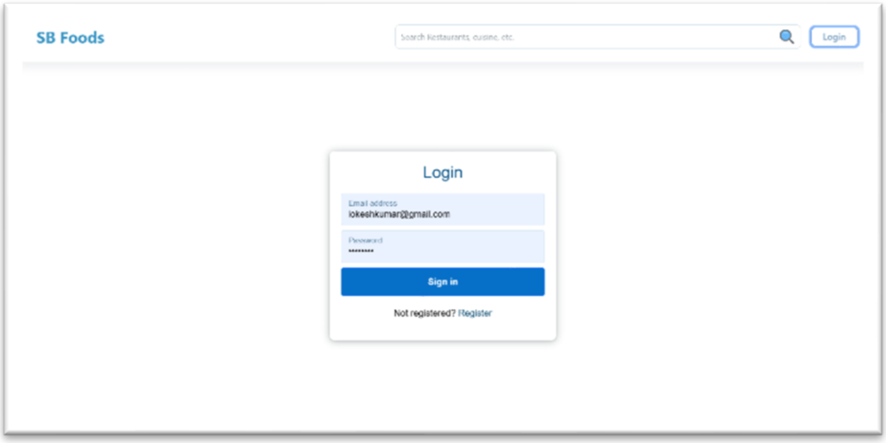


10.Testing:

Integration Testing: Ensures different modules (e.g., checkout and payment) work together smoothly.

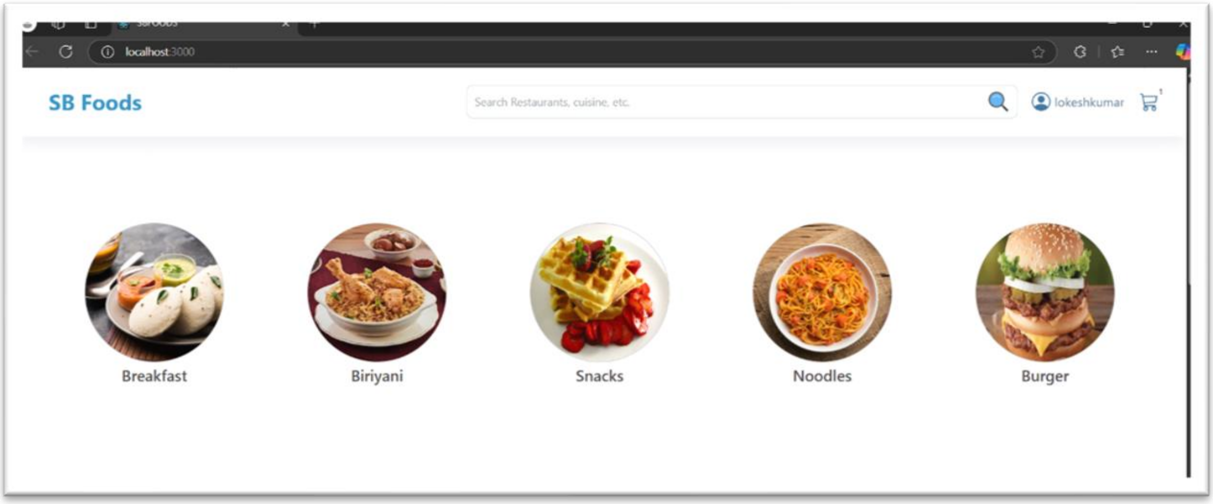
End-to-End (E2E) Testing: Simulates full user journeys, from login to placing an order.

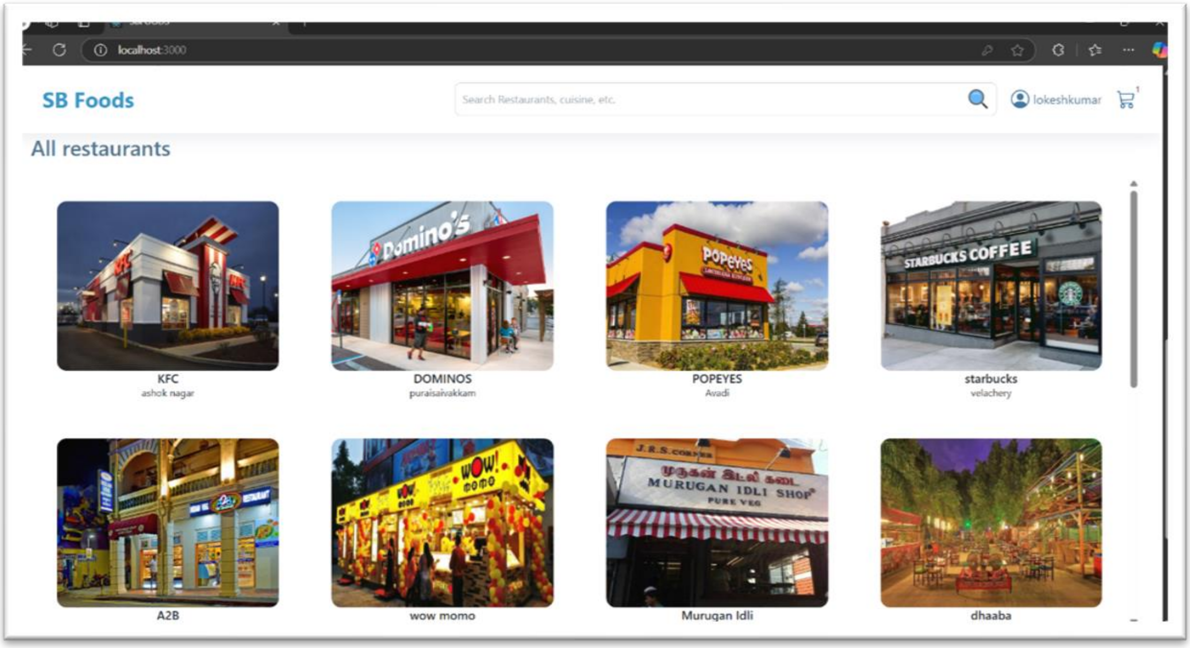
11. Screenshots or Demo:



Email address
lokeshkumar@gmail.com

Password
****





All restaurants

KFC
ashok nagar

DOMINOS
puraisaivakkam

POPEYES
Avadi

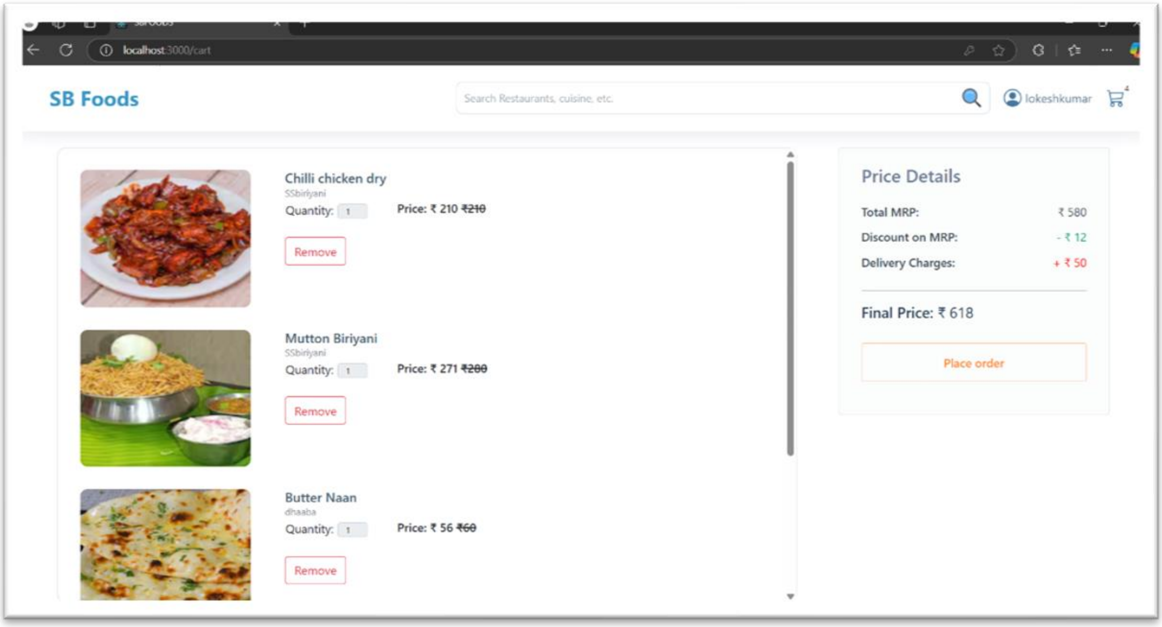
starbucks
velachery

A2B

wow momo

Murugan Idli

dhaaba



Chilli chicken dry
SSbiriyani
Quantity: 1 Price: ₹ 210 ~~₹210~~
Remove

Mutton Biriyani
SSbiriyani
Quantity: 1 Price: ₹ 271 ~~₹280~~
Remove

Butter Naan
dhaaba
Quantity: 1 Price: ₹ 56 ~~₹60~~
Remove

Price Details

Total MRP:	₹ 580
Discount on MRP:	- ₹ 12
Delivery Charges:	+ ₹ 50

Final Price: ₹ 618

Place order

12. Known Issues :

Issues:

- Irrelevant pictures for the updated app.
- Multiple error in the client and server database code.

13.Future Enhancement:

1. Personalized Recommendations
Use machine learning to suggest dishes and restaurants based on users' past orders, preferences, and current trends.
2. Enhanced Order Tracking
Integrate real-time GPS tracking with ETA updates, and show order preparation stages (e.g., "Preparing," "Out for Delivery") with visual cues.
3. Subscription or Loyalty Programs
Offer membership options for benefits like free delivery, exclusive discounts, or reward points on each purchase.
4. Voice-Activated Ordering
Integrate voice assistants like Alexa, Google Assistant, or Siri to allow hands-free ordering.
5. Multi-Restaurant Ordering
Allow users to order from multiple restaurants in a single order, with separate tracking for each restaurant's delivery.