ORDER ON THE GO

**Project Tittle: ORDER ON THE GO**

**INTRODUCTION**

"Order on the Go" refers to the ability to place and manage orders conveniently through mobile apps, websites, or other digital platforms, typically while on the move. This modern approach allows users to bypass long lines, save time, and have products or services ready for pickup or delivery at their convenience.

This concept is increasingly popular in various sectors like:

1. **Food and Beverage**: People can order meals or drinks from restaurants, cafes, or fast-food outlets without having to wait in person.
2. **Retail**: Shoppers can buy products from stores and have them delivered or ready for pickup at their nearest location.
3. **Services**: Users can book appointments or schedule services such as salon visits, car repairs, or fitness classes through apps while on the go.

**Team members:**

1. P. Mounika – Team leader
2. G. Ashok Kumar – Coding
3. K. Sai sagar - Documentation
4. P. Bhanu Prakash - Documentation

**PROJECT OVERVIEW:**

**Purpose:**

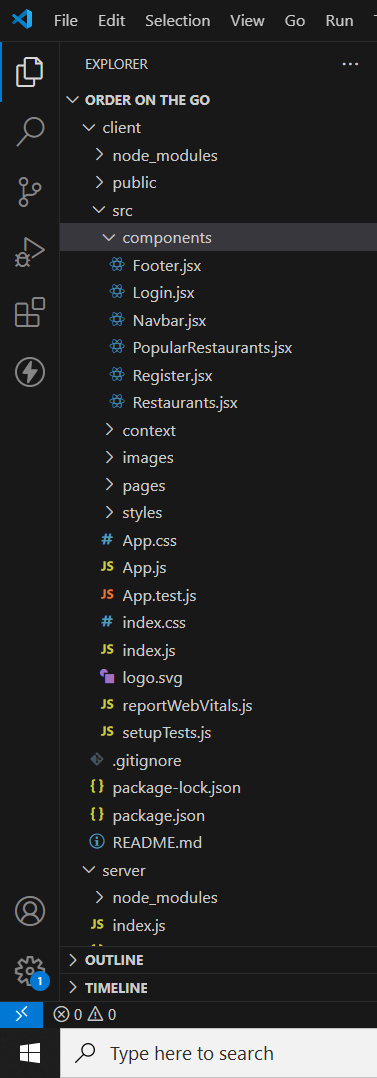
The purpose of an "Order on the Go" project is to streamline and improve the customer experience by enabling users to place and manage orders quickly and conveniently, anytime and anywhere.

**Features:**

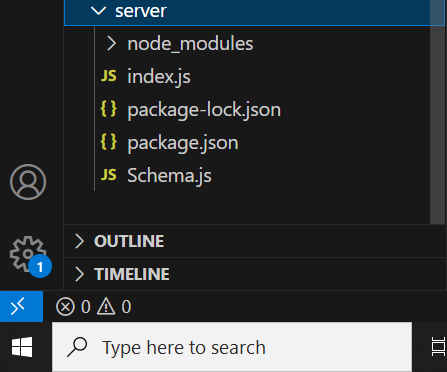
* Real-Time Analytics for Businesses.
* Customizable Delivery Options.
* User-Friendly Mobile App/Website Interface.
* Customization Options.

**ARCHITECTURE**

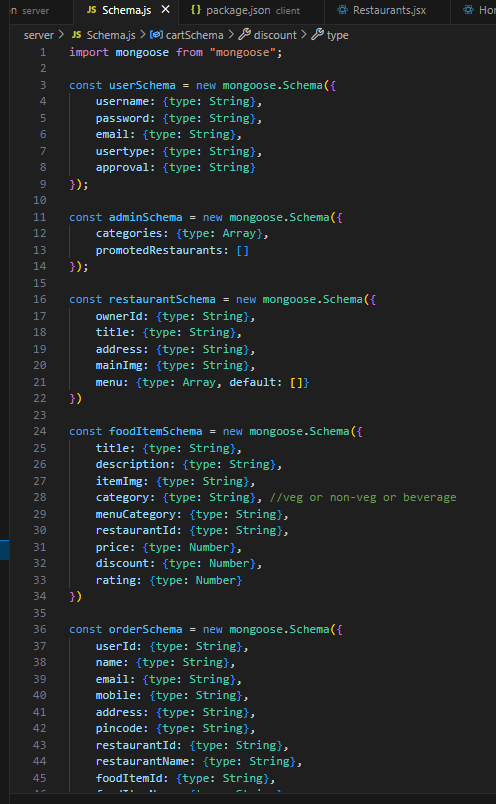
Frontend :



Backend :



Database:



**SETUP INSTRUCTIONS**

**Prerequisites:**

For a project like ORDER ON THE GO, which is typically a real estate listing platform, here a list of possible software dependencies based on a common tech stack. If your project is built with specific technologies, you can adjust the list accordingly. Below is a general overview:

Node.js and npm:  <https://nodejs.org/en/download/>

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

**Mongo DB:** Set up a Mongo DB database to store hotel and booking information. Install Mongo DB locally or use a cloud-based Mongo DB 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 Mongo DB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the Mongo DB database and perform CRUD (Create, Read, Update, and 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 Git Hub or Bit bucket 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 Web Storm.

 • Visual Studio Code: Download from https://code.visualstudio.com/download

• Sublime Text: Download from https://www.sublimetext.com/download

• Web Storm: Download from <https://www.jetbrains.com/webstorm/download>

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

Link: https://www.section.io/engineering-education/nodejs- mongoose.js-mongo dB

**To run the existing SB Foods App project downloaded from github:**

Follow below steps:

**Clone the repository:**

• Open your terminal or command prompt.

• Navigate to the directory where you want to store the e-commerce app.

• Execute the following command to clone the repository:

**Git clone: https://github.com/harsha-vardhan-reddy-07/Food-Ordering-App-MERN Install Dependencies:**

• Navigate into the cloned repository directory:

**Cd Food-Ordering-App-MERN**

• Install the required dependencies by running the following command:

**Npm install**

**Start the Development Server:**

• To start the development server, execute the following command:

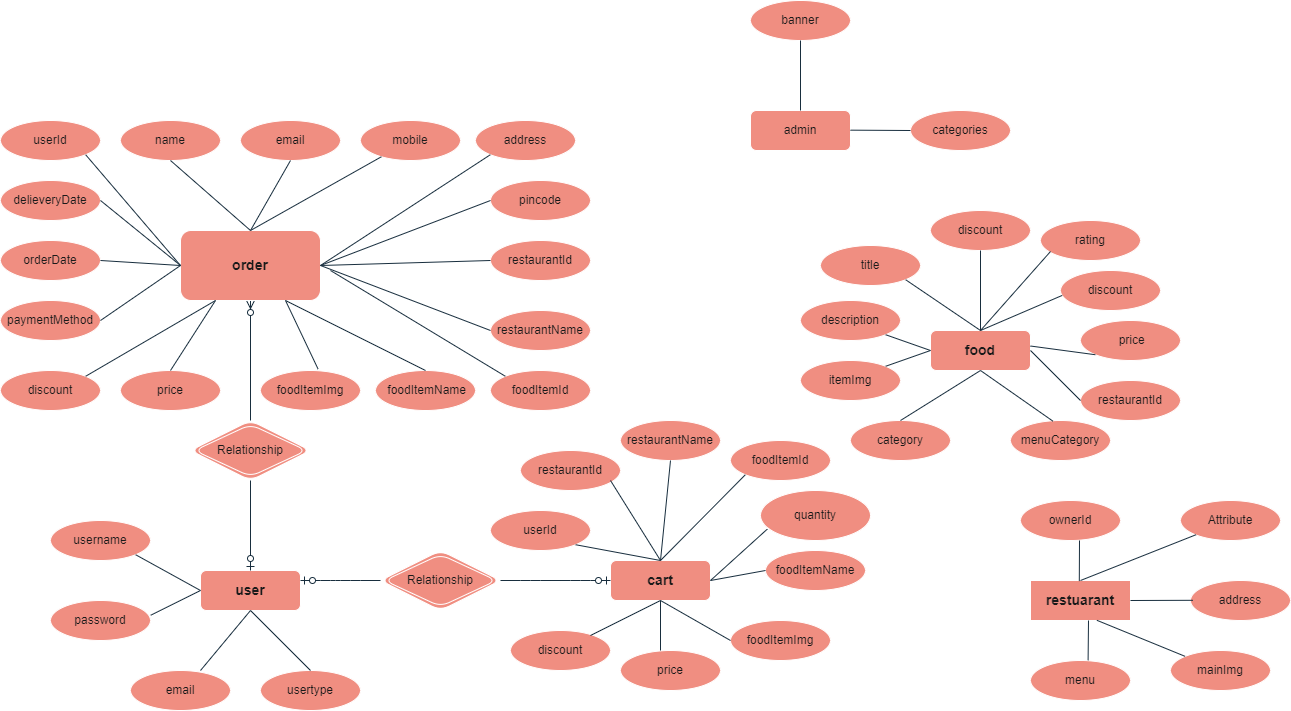
**Npm run dev. or npm run start**

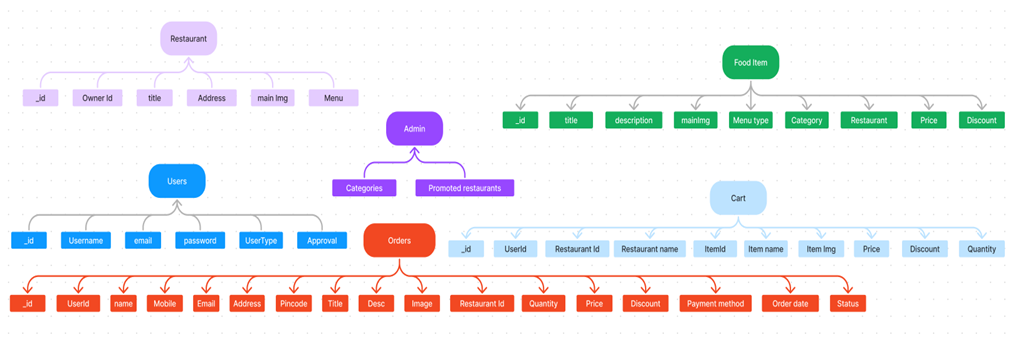
• The e-commerce app will be accessible at http://localhost:3000 by default. You can change the port configuration in the file if needed.

**Access the App:**

* Open your web browser and navigate to http://localhost:3000.
* You should see the flight booking app's homepage, indicating that the installation and setup were successful.
* You have successfully installed and set up the MS Foods app on your local machine. You can now proceed with further customization, development, and testing as needed.

## ER DIAGRAM:

****

****

 The MS Foods 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.

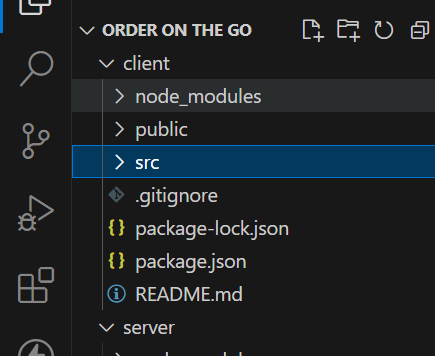
**Cart:** This collection stores all the products that are added to the cart by users. Here, the elements in the cart are  differentiated by the user Id.

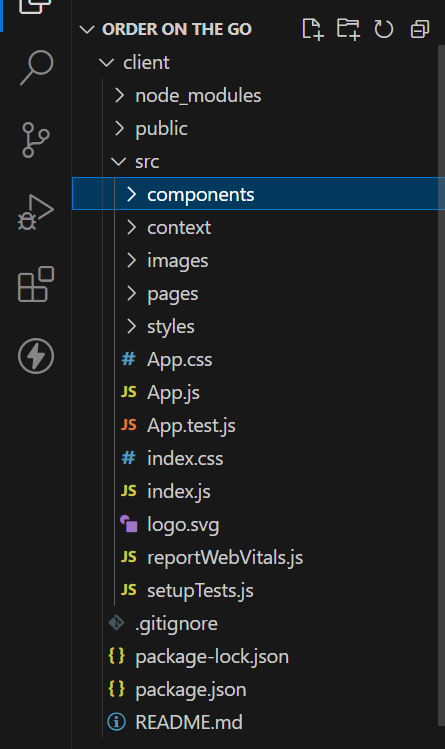
**Orders:** This collection stores all the orders that are made by the users in the platform.

**FOLDER STRUCTURE**

**Client:**

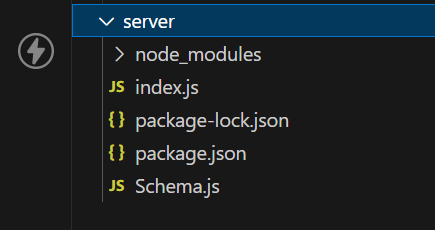
The below directory structure represents the directions and files in the client folder(front end) where, react js is used along with Api’s .





**Server:**

The below directory structure represents the directions and files in the client folder(front end) where, node js ,express js, and mongoDB are used along with Api’s .



Running the application :

Provide commands to start the frontend and backend servers locally.

FRONTEND : npm start in the client directory.

BACKEND : npm start in the server directory.a

us

**USER & ADMIN FLOW:**

**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 order.

• They can then proceed by entering address and payment details. • After ordering, they can check them in the profile section.

**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.

**3. Admin Flow:**

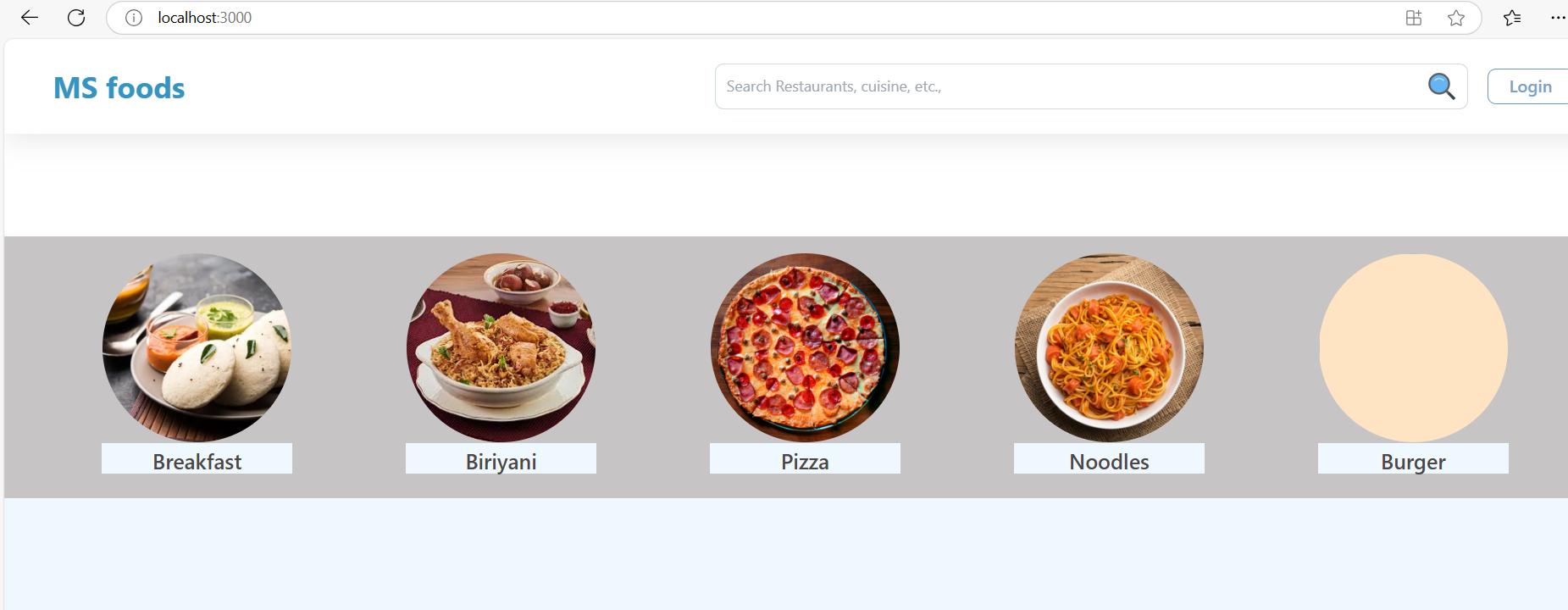
• Admins start by logging in with their credentials.

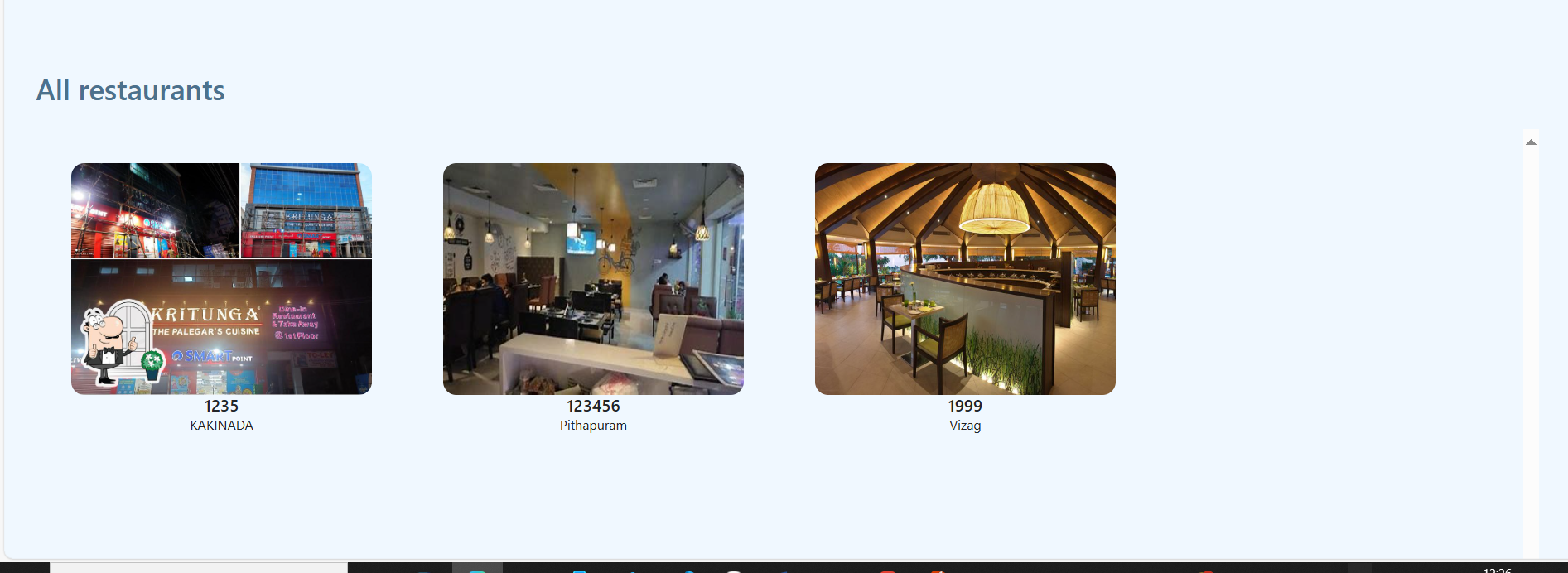
• Once logged in, they are directed to the Admin Dashboard.

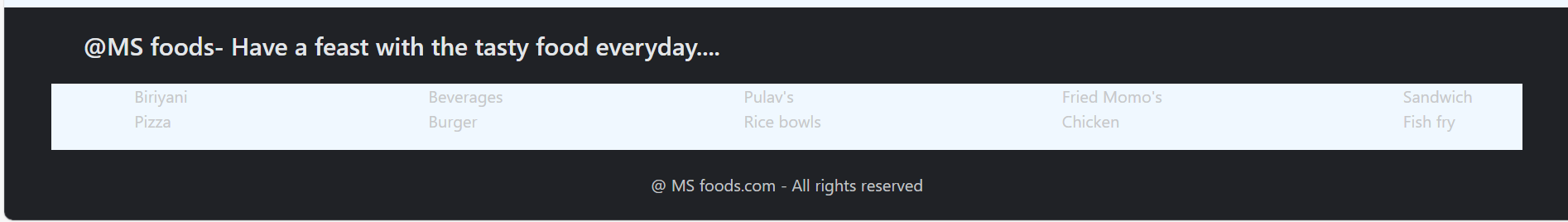
• Admins can access the users list, products, orders, etc.

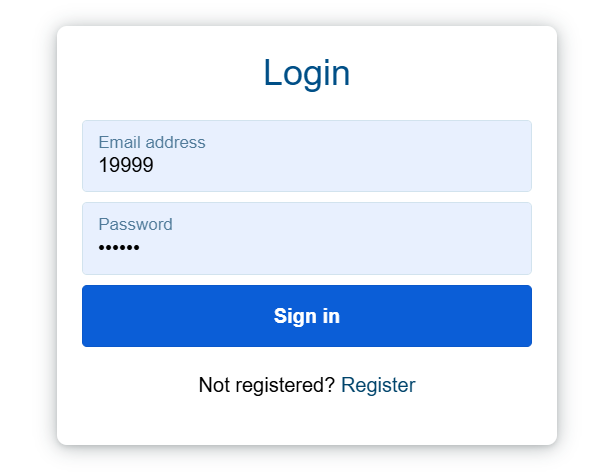
er and receive an access token.

**SCREENSHOTS:**

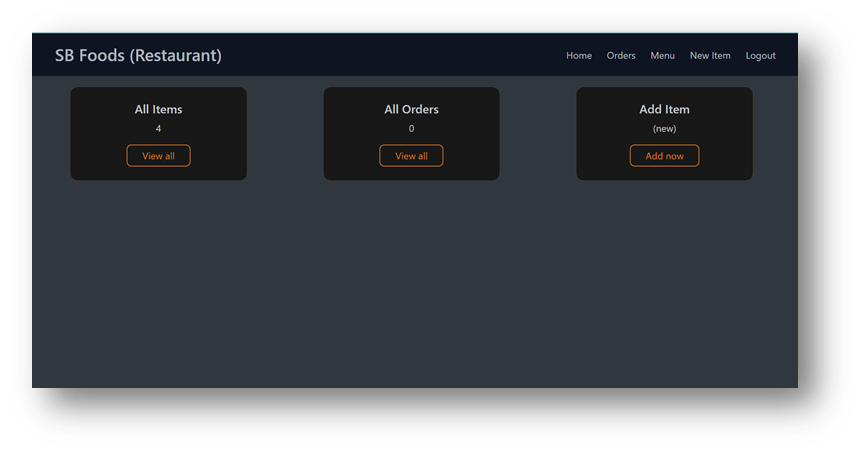
****

****

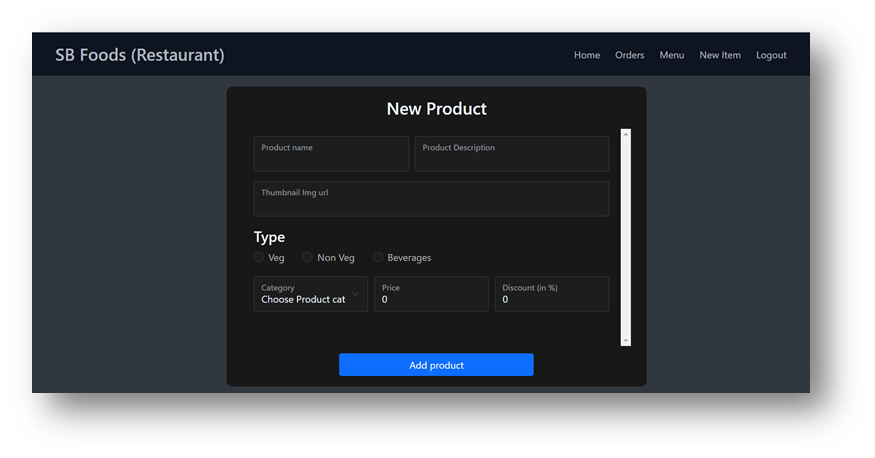
****



**·       Restaurant Dashboard**

****

**·       New Item**

****

\*\* Happy Coding \*\*