Order on the Go: Your On-Demand Food Ordering Solutions

Introduction

Project Title: Order on the Go: Your On-Demand Food Ordering Solutions

Team Members And Their Roles:

Member-1:- K S V Satya Sai - [Backend Development & Project Implementation & Execution]

Member-2:- G S Pradeepthi – [Frontend Development & Database Development]

Member-3:- J Yasasvee – [Project Setup And Configuration & Frontend Development]

Member-4:- K Siddarda – [Backend Development & Project Setup And Configuration]

Project Overview

Purpose

The 'Order on the Go' project (SB Foods) is a MERN-based food ordering application designed to connect users with multiple restaurants. It allows customers to explore menus, place orders, and track them conveniently. For restaurants, it offers a platform to manage listings and process orders, while administrators oversee operations.

Features

- Comprehensive product catalog with multiple restaurant listings.

- Detailed order placement page for entering delivery and payment details.

- Secure and efficient checkout process.

- Post-order confirmation and details page.

- Restaurant dashboard for managing products and orders.

Application Flow

User Flow

- Register for an account and log in.

- Browse products and add items to the cart.

- Enter delivery and payment details to complete the order.

- Track order history through the profile section.

Restaurant Flow

- Authenticate and request admin approval.

- List, update, and manage menu items.

Admin Flow

- Log in to the admin dashboard.

- Manage users, restaurants, products, and orders.

Architecture

Frontend

Built with React.js, the frontend delivers a responsive interface for users and restaurant partners. Pages include product listings, cart, checkout, order history, and dashboards.

Backend

Developed using Node.js and Express.js, the backend handles core logic including user authentication, product handling, order processing, and database communication.

Database

Uses MongoDB for storing all structured data. Below are the key schemas used:

- userSchema: User details (username, email, password).

- productSchema: Product information (name, price, description).

- ordersSchema: Order records (userId, productId, quantity, size, date).

- cartSchema: Temporary cart data linked to a user.

- adminSchema: Admin data (categories, promoted restaurants).

- restaurantSchema: Restaurant data including profile and menu.

Setup Instructions

Prerequisites

- Node.js (v14+)

- MongoDB (Compass or Atlas)

- npm or yarn

Installation
- Create project folder and initialize package.json

- Install Express.js, Mongoose, dotenv, cors, body-parser

- Create index.js and set up Express server

- Configure MongoDB and environment variables

Folder Structure

Client
- src/components: React components for UI

- src/pages: Individual pages (Home, Cart, Checkout)

- src/context: State management using Context API

Server
- server/index: Route handlers for users, products, orders, admin

- server/Schema: Mongoose schemas and models

- server/index: Logic for handling API calls

- server/: Authentication and error handling logic

Running the Application
The project uses concurrently to streamline development by running both frontend and backend servers in parallel from the root directory.

To start the full stack application:

➔ npm start

| This command triggers scripts defined in the root package.json, simultaneously launching:

- **Frontend**: React app located in /client
- **Backend**: Express server located in /server

No need to navigate into individual folders manually—everything is wired up for convenience and efficiency.

API Documentation

Method	Endpoint	Description
POST	/api/register	Register a new user with email and password. performs validation and hashes passwords using byscript.
POST	/api/login	Authenticates a user and returns a JWT token.
GET	/api/products	Retrieves a list of available products. Supports filtering and pagination (optional).
POST	/api/orders	Submits a new order. Requires authentication.
GET	/api/orders/:id	Fetches details for a specific order by ID. Secured accessible by order owner or admin.

Authentication

User authentication is managed via **JSON Web Tokens (JWT)**. The backend issues a token upon successful login, which must be included in the Authorization header of protected requests.

Access Control:

- Regular users: Can view products, create orders, and view their own orders.
- Admins: Have extended privileges like viewing all orders and managing products.

User Interface

The frontend is built with **React**, focusing on simplicity and responsiveness. Key features include:

- **Navigation**: Seamless transitions between views using React Router.
- **Pages**:

▪ Home
▪ Resturant Listing
▪ Food items
▪ Cart
▪ Checkout
▪ Admin/User Dashboards

The layout is responsive and styled with **Bootstrap**, ensuring optimal experience across devices.

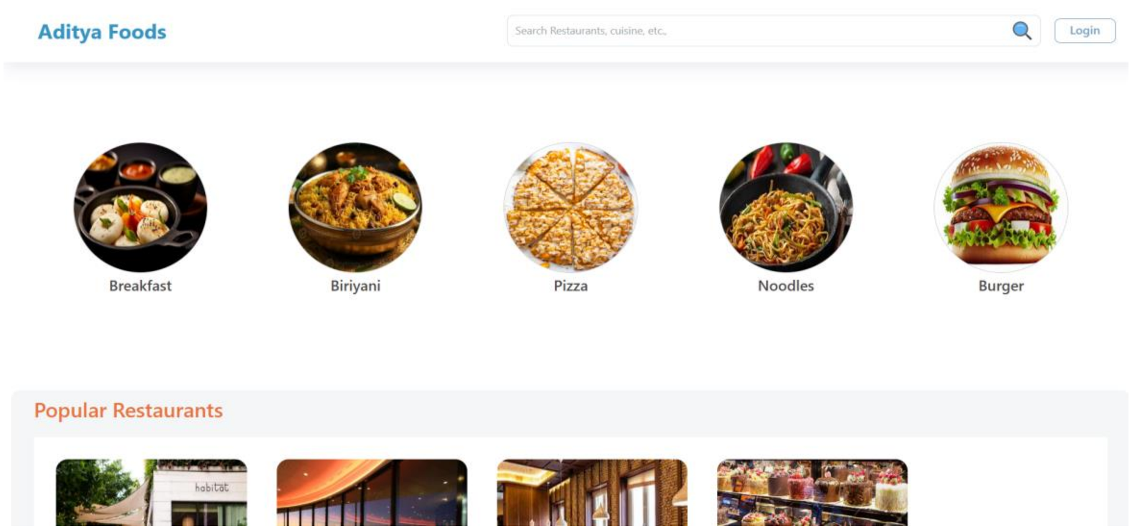
Testing

- **Manual Testing**: Conducted during development by interacting with the frontend and verifying backend responses.
- **Postman**: Used to test and document APIs, verify headers, request bodies, and token validity.

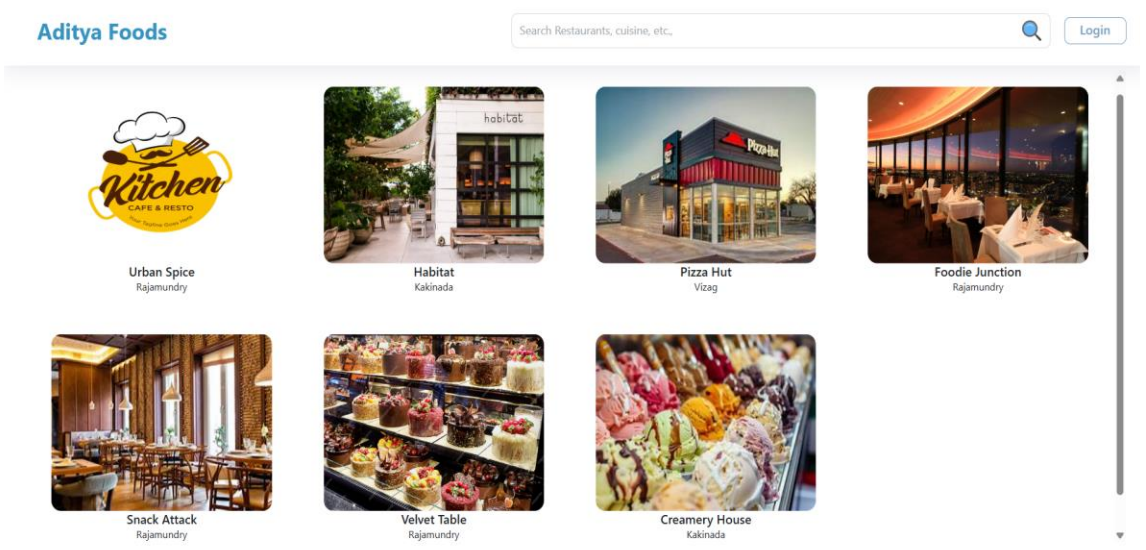
  Future enhancement: Add unit and integration tests using frameworks like Jest or Mocha.

Screenshots or Demo

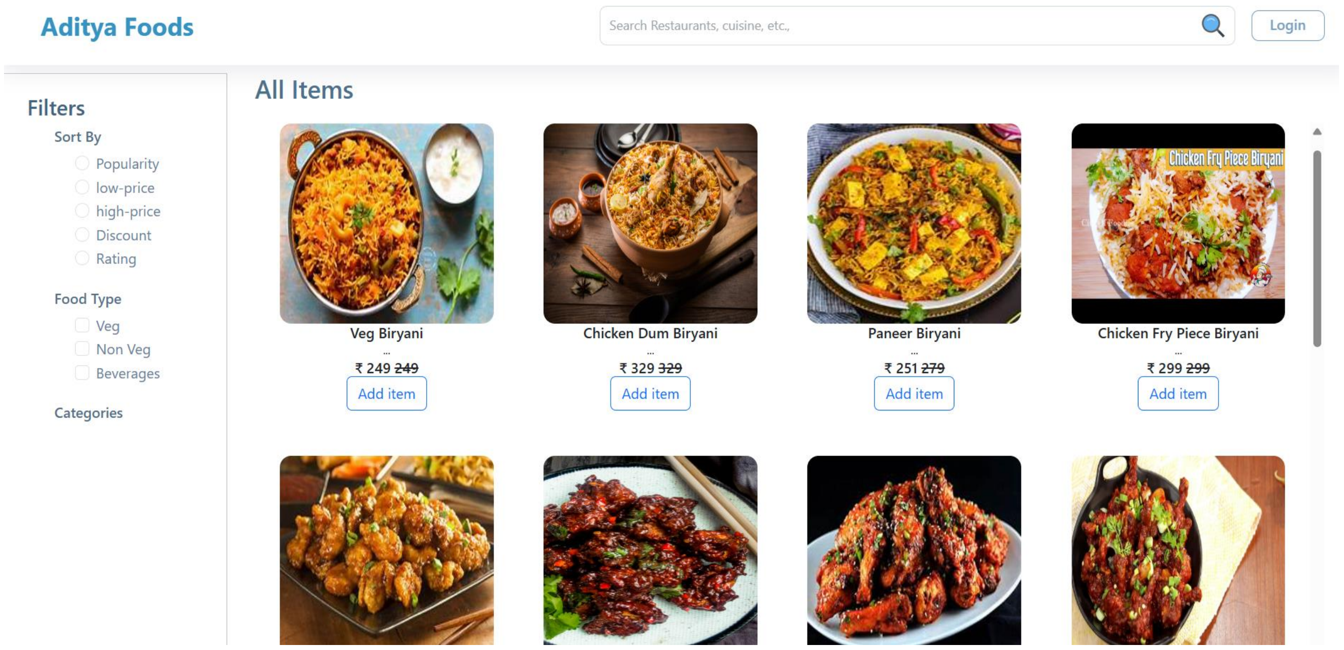
- **Landing page**



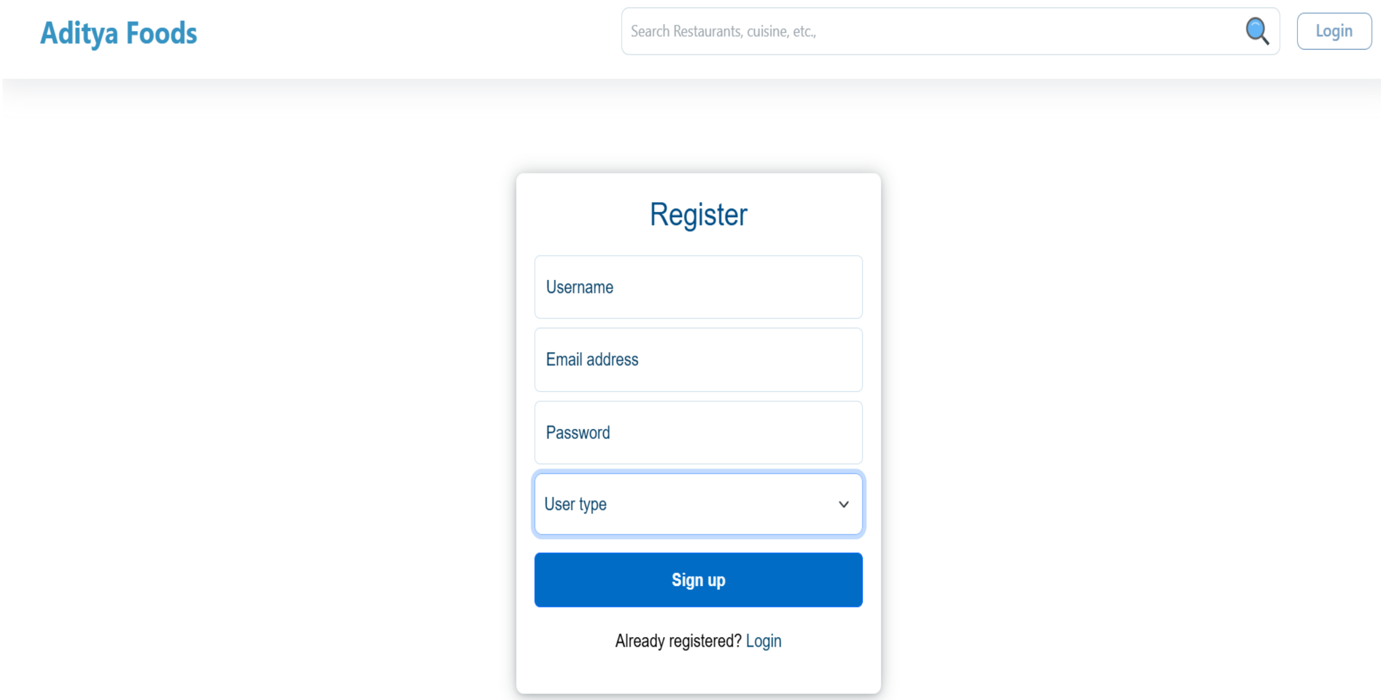
- **Restaurant**



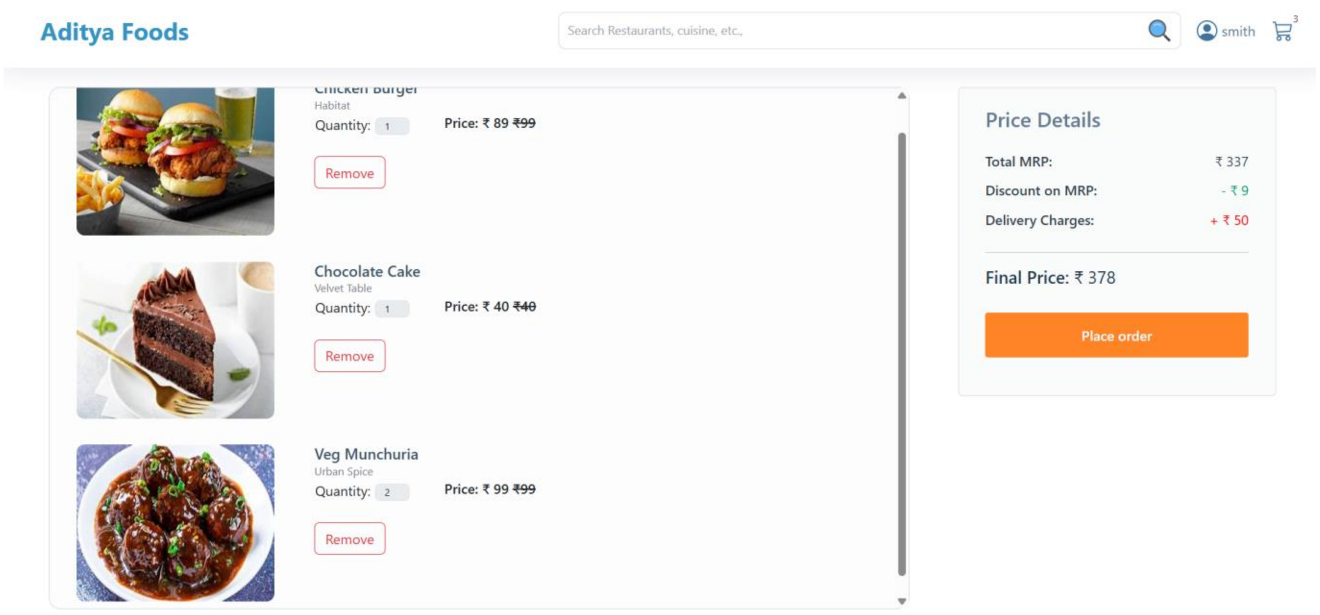
- **Restaurant Menu**



- **Authentication**



- **User Profile & Cart**



Aditya Foods

Search Restaurants, cuisine, etc.,

smith 3

Chicken Burger
Habitat
Quantity: 1
Price: ₹ 89 ~~₹99~~
Remove

Chocolate Cake
Velvet Table
Quantity: 1
Price: ₹ 40 ~~₹40~~
Remove

Veg Munchuria
Urban Spice
Quantity: 2
Price: ₹ 99 ~~₹99~~
Remove

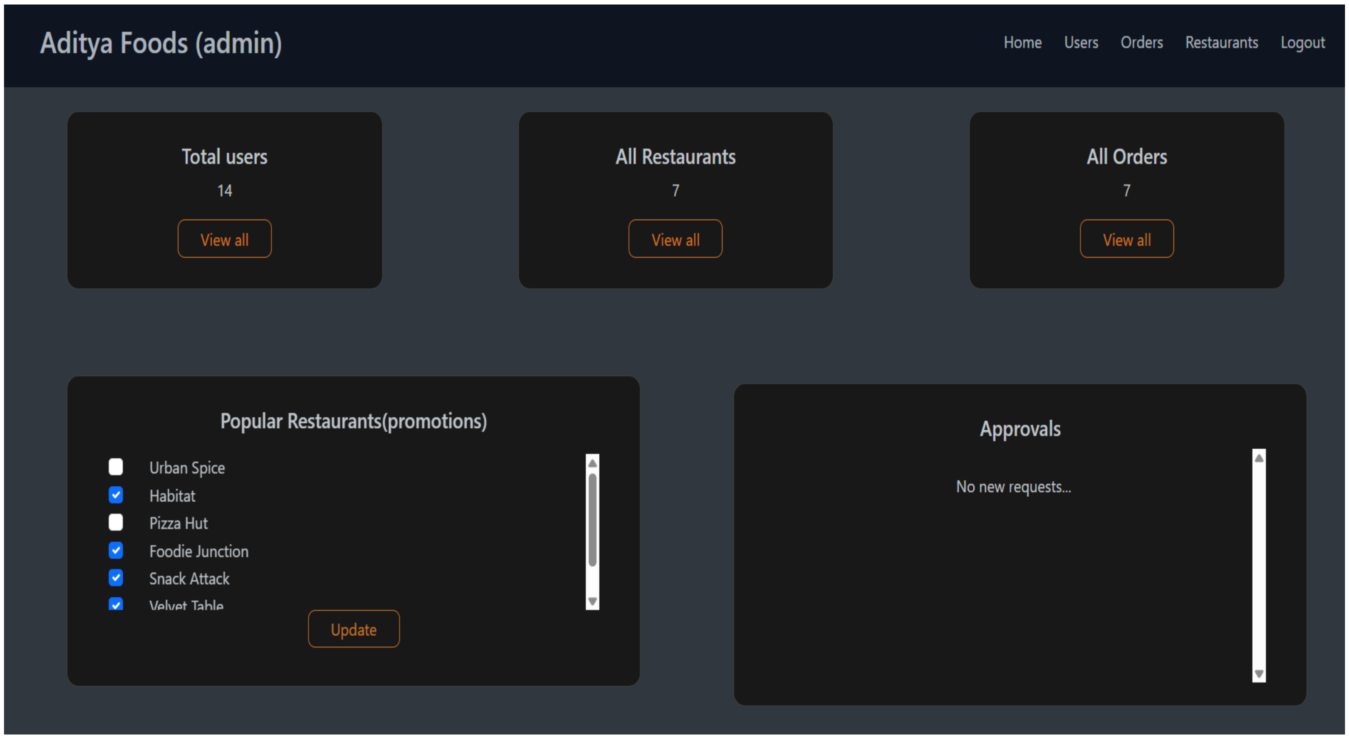
Price Details

Total MRP:	₹ 337
Discount on MRP:	- ₹ 9
Delivery Charges:	+ ₹ 50

Final Price: ₹ 378

Place order

- **Admin dashboard**



Aditya Foods (admin)

Home Users Orders Restaurants Logout

Total users
14
View all

All Restaurants
7
View all

All Orders
7
View all

Popular Restaurants(promotions)

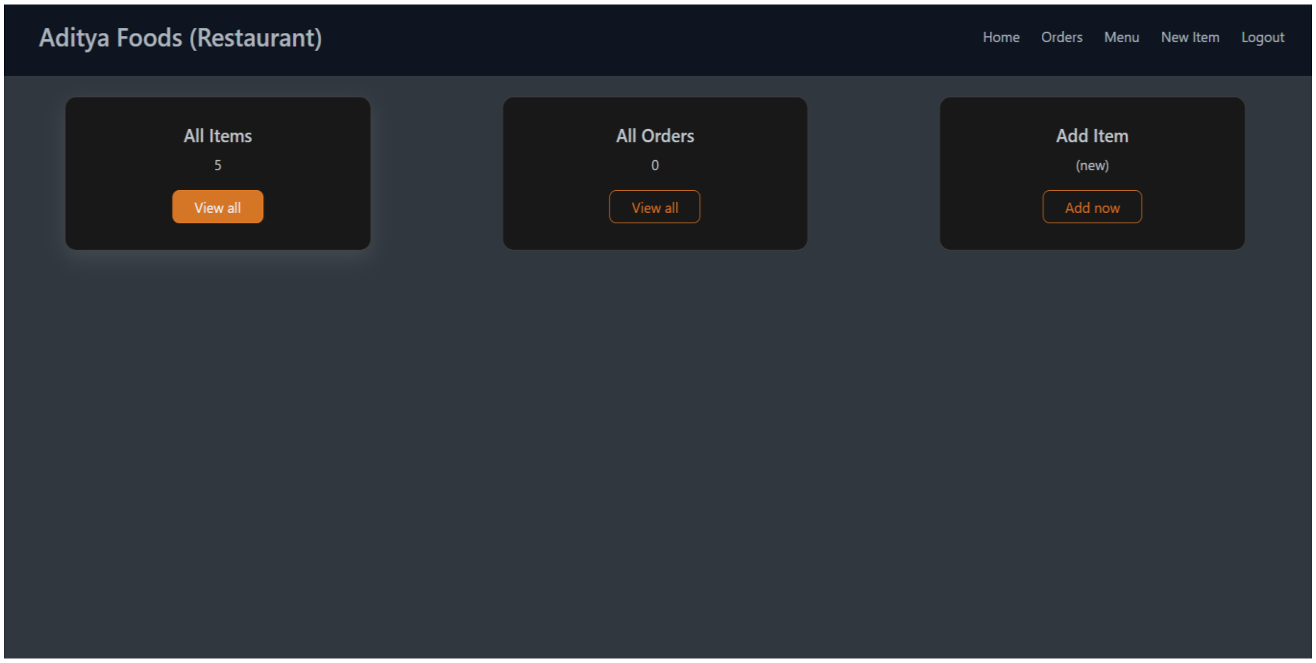
- [ ] Urban Spice
- [x] Habitat
- [ ] Pizza Hut
- [x] Foodie Junction
- [x] Snack Attack
- [x] Velvet Table

Update

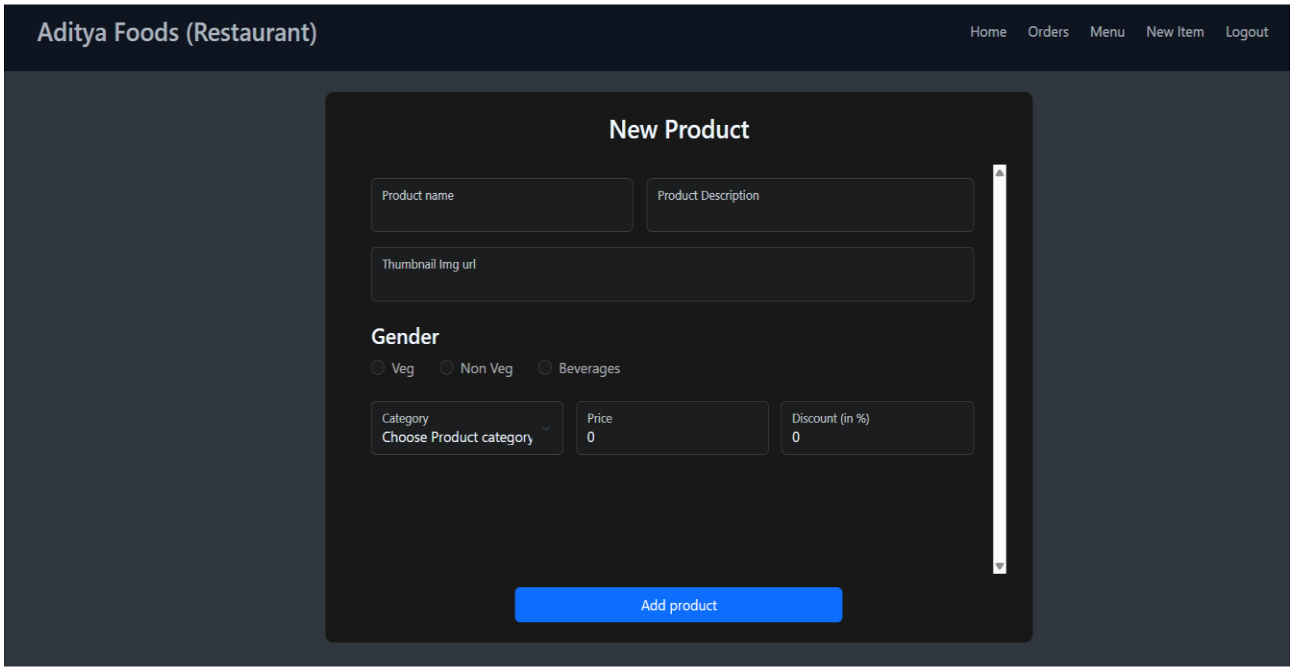
Approvals

No new requests...

- **Restaurant Dashboard**



- **New Item**



Demo Video :

https://github.com/ksatyasai/Food-Ordering-Website-Mern/tree/main/video%20demo

Known Issues

- Lack of automated testing (unit/integration).
- No continuous integration or deployment pipeline in place.
- Error handling and form validation can be improved further.
- Product and order data are not paginated.

Future Enhancements

- Integrate **payment gateway** (e.g., Stripe, Razorpay) for secure transactions.
- Implement **real-time order status updates** using WebSockets (e.g., Socket.IO).
- Add **push notifications** for users on order activity.
- Set up **CI/CD pipeline** for automated deployment and testing.
- Improve **admin panel** with analytics and product insights.
- Add **user profile management** (update info, view past orders).