PROJECT DOCUMENTATION

Orderonthego : Your on-demand food ordering

Technlogy Stack : React.js | Node.js | Express.js | MongoDB

Modules : User Management | Admin | Food Details | Cart | Order | Authentication

Project Title : Orderonthego: Your on-demand food ordering

Team ID : **LTVIP2026TMIDS24286**

Team Size : 4

Team members and Roles :

Team Leader	**:**	Palli Deepthi Sai Sree (Frontend Developer)
Team member	**:**	Polagani Lakshmi Venkata Sai Ram (Frontend Support)
Team member	**:**	Sanaka Gowthami (UI/UX Designer)
Team member	**:**	Shaik Farooq (Full Stack Developer, Backend Developer)

Table of Contents

1 . Introduction

2 . Project Overview

3 . Architecture

4 . Setup Instructions

5 .Folder Structure

6 . Running the Application

7 . API Documentation

8 . Authentication

9 . User Interface

10 . Testing

11 . Screenshots / Demo

12 . Know Issues

13 . Future Enchacements

14 . Appendix A: UI Screenshots

Orderonthego: your on-demand food ordering

Full Stack MERN Project Documentation

1. Introduction

Project Title

orderonthego: your on-demand food ordering

About the Project

OrderOnTheGo is a full-stack web application developed using the MERN stack (MongoDB, Express.js, React.js, Node.js). It functions as an online food ordering platform where users can browse food items, view detailed descriptions, add items to their cart, and place orders easily.

The system is designed to simulate a real-world on-demand food delivery service. Users can register and login securely, explore various food categories, search for their favorite dishes, manage their cart, and track their orders.

The platform supports role-based functionality for different types of users:

• Customers can browse food items, add items to cart, place orders, and track order history.
• Admin can manage food items, categories, users, and monitor all orders through an admin dashboard.

The application ensures secure authentication, smooth API communication between frontend and backend, and a responsive user interface that works across desktop and mobile devices.

Objective

• To develop a responsive and scalable on-demand food ordering web application using the MERN stack.

• To implement secure authentication and authorization using JWT (JSON Web Token).

• To allow customers to browse food items, filter by category, and manage their cart efficiently.

• To enable users to place orders and maintain order history.

• To provide an admin panel to manage food items, users, and orders effectively.

• To design a user-friendly and interactive interface for a seamless food ordering experience.

2. Project Overview

Purpose

The purpose of OrderOnTheGo is to design and implement a fully functional, user-friendly, and responsive online food ordering platform. In today's fast-paced digital world, customers prefer ordering food online due to convenience, speed, and accessibility. OrderOnTheGo fulfills this need by providing a centralized platform where food items are displayed with details such as name, description, price, category, and images.

The platform supports role-based access such as Customer and Admin. Each role has its own permitted actions and dashboard features. This structure reflects a real-world online food ordering workflow.

Goals

• Build a complete MERN stack application with frontend, backend, and database integration.

• Implement secure authentication and protected routes using JWT.

• Provide core food ordering features such as food browsing, cart management, and order placement.

• Allow customers to track their order history.

• Provide admin features to manage food items, users, and orders.

• Ensure responsive design for both desktop and mobile users.

Key Features / Modules

The OrderOnTheGo platform includes the following key modules:

• Authentication Module (Register / Login with secure JWT authentication)

• Food Browsing Module (View food items with images, categories, and prices)

• Search & Filter Module (Search foods by name or category)

• Food Details Module (View detailed description of selected item)

• Cart Module (Add to cart, update quantity, remove items)

• Order Module (Place order, view order history)

• Admin Dashboard (Manage food items, users, and orders)

• User Profile Module (View user information and order history)

3. Architecture

OrderOnTheGo follows a modern 3-tier MERN architecture:

- MongoDB (Database Layer)
- Express.js + Node.js (Backend/API Layer)
- React.js (Frontend/Presentation Layer)

This architecture ensures scalability, maintainability, and clear separation of concerns.

Front-end Architecture – React.js

The frontend is built using React.js, enabling dynamic and responsive user interfaces through reusable components and page-based routing.

- React Router DOM is used for navigation between pages (Home, Login, Register, Cart, Orders, Admin Dashboard).

- Conditional rendering is used to show features based on user authentication and role.

- Axios is used to communicate with backend REST APIs.

- The UI is styled using CSS and Bootstrap to provide a clean and responsive design.

- Components are modularized (Navbar, FoodCard, CartPage, LoginPage, etc.) for better maintainability.

Back-end Architecture – Node.js & Express.js

The backend is developed using Node.js and Express.js. It exposes REST APIs for authentication, food management, cart management, and order processing.

- Routes are modularized by feature (auth, foods, users, cart, orders).

- Controllers contain business logic for each API endpoint.

- Middleware verifies JWT tokens and protects routes.

- Passwords are securely hashed before storing in the database.

- Environment variables are managed using dotenv for security.

Database Architecture – MongoDB

MongoDB is used as the NoSQL database for storing user data, food items, cart details, and order information. MongoDB provides flexibility and scalability for modern web applications.

- Users collection stores user credentials and role information.

- Foods collection stores food item details such as name, price, category, and image.

- Orders collection stores order details, total amount, and status.

• Cart collection stores user-specific cart items.

The database structure ensures efficient data retrieval and relationship management between users and orders.

4. Setup Instructions

Prerequisites

Tool	Purpose	Recommended Version
Node.js	JavaScript runtime environment	v18+
Npm	Node packadge manager	v9+
MongoDB(Local)	Database for storing data	Any stable version
Git	Version control	Latest
VS code	Code editor	Optional
Postman/Thunder Client	API testing	Optional

Installation Guide

Step 1: Clone the Project Repository

```
git clone https://github.com/your-username/orderonthego.git

cd orderonthego
```

Step 2: Install Dependencies

Install frontend packages:

```
cd frontend

npm install
```

Install backend packages:

```
cd ../backend
npm install
```

Step 3: Set Up Environment Variables

Inside the /backend folder, create a .env file and add:

```
MONGO_URI=mongodb://127.0.0.1:27017/orderonthego

JWT_SECRET=your_secret_key

PORT=5000
```

Step 4: Run the Application

Start backend server:

```
cd backend
```

npm start

Start frontend React app:

cd ../frontend

npm start

Once started:

- Frontend runs on: http://localhost:3000
- Backend runs on: http://localhost:5000

5. Folder Structure

Frontend Folder Structure (React)

Folder/File	**Purpose**
/frontend	Root folder for React app
/public	Static assets
/src/cpmponents	Reusable UI components(Navbar,FoodCard,etc.)
/src/pages	Pages like Home, Login, Register, Cart, Orders
App.js	Central router
index.js	React entry point
Package.json	Frontend dependencies

Backend Folder Structure (Node.js + Express)

Folder/File	**Purpose**
/backend	Root folder for backend
/models	Mongoose schemas (User,Food, Order,Cart)
/routes	Express routes
/controller	Business logic
/middleware	JWT authentication middleware
Server.js	Backend entry point
.env	Environment variables
Package.json	Backend dependencies

6. Running the Application

Start frontend:

cd frontend

npm start

Start backend:

cd backend

npm start

Expected URLs:

• Frontend UI: http://localhost:3000
• Backend API: http://localhost:5000/api

7. API Documentation

The OrderOnTheGo backend exposes REST-style APIs. All routes return JSON format.

Authentication Routes

Method	**Endpoint**	**Description**
POST	/api/users/register	Register new user
POST	/api/users/login	Login user

Food Routes

Method	**Endpoint**	**Description**	**Access**
GET	/api/foods	Fetch all food items	Public
GET	/api/foods/:id	Fetch single food	Public
POST	/api/foods	Add new food	Admin
PUT	/api/foods/:id	Update food	Admin
DELETE	/api/foods/:id	Delete food	Admin

Cart Routes

Method	**Endpoint**	**Description**
POST	/api/cart	Add item to cart
GET	/api/cart	Get cart items
DELETE	/api/cart/:id	Remove cart item

Order Routes

Method	**Endpoint**	**Description**	**Access**
POST	/api/orders	Place an order	User
GET	/api/orders/myorders	Get user order	User
GET	/api/orders	Get all orders	Admin
PUT	/api/orders/:id	Update order status	Admin

8. Authentication

OrderOnTheGo uses secure JWT-based authentication with role-based access (User / Admin).

Authentication Flow

Step	Action	What Happens
1	Register	User details stored in MongoDB after password hashing (bcrypt)
2	Login	Server validates credentials and returns JWT token
3	Protected Requests	Frontend sends token in header: Authorization: Bearer <token>

JWT Token Details

Field	Purpose
userId	MongoDB ID of the logged-in user
email	Email address of the user
role	user / admin
iat	Token issued time
exp	Token expiry time

9. User Interface

The OrderOnTheGo user interface is designed for simplicity, responsiveness, and smooth navigation. The UI supports two main roles: User and Admin.

UI Screens / Pages

• Home Page: Landing page displaying food items and categories.

• Login Page: User login form with email and password.

• Signup Page: User registration form.

• Food Listing Page: Displays all available food items as cards.

• Food Details Page: Shows detailed information about selected food item.

• Cart Page: Displays selected food items with quantity and total price.

• Orders Page: Shows logged-in user's order history.

• Admin Dashboard: Manage food items, users, and orders.

• Admin Food Management Page: Add, update, delete food items.

• Admin Orders Page: View and update order status.

10. Testing

Manual testing was performed to ensure proper functionality, stability, and role-based access control.

Front-end Testing

• Validated Login and Signup forms.

• Verified correct navigation after login.

• Checked food listing and food details rendering.

• Verified cart operations (add, remove, update quantity).

• Tested order placement functionality.

• Verified responsive layout using browser developer tools.

Backend Testing

• Tested Register and Login API endpoints.

• Verified JWT-protected routes with valid and invalid tokens.

• Tested CRUD operations for food items (Admin access).

• Tested cart APIs for adding and removing items.

• Tested order placement and order retrieval APIs.

• Tested admin-only routes for managing foods and orders.

11. Screenshots / Demo

The following screenshots demonstrate the key pages and modules of the OrderOnTheGo platform.

Screenshots include:

• Home Page
• Login Page
• Signup Page
• Food Listing Page
• Cart Page
• Orders Page
• Admin Dashboard
• Admin Food Management
• Admin Orders Page

12. Known Issues

- No online payment gateway integration (orders are placed without payment processing).
- Limited search and filtering options (basic food listing only).
- No pagination (all food items load at once).
- No automated testing framework (manual testing only).
- No email verification during registration.

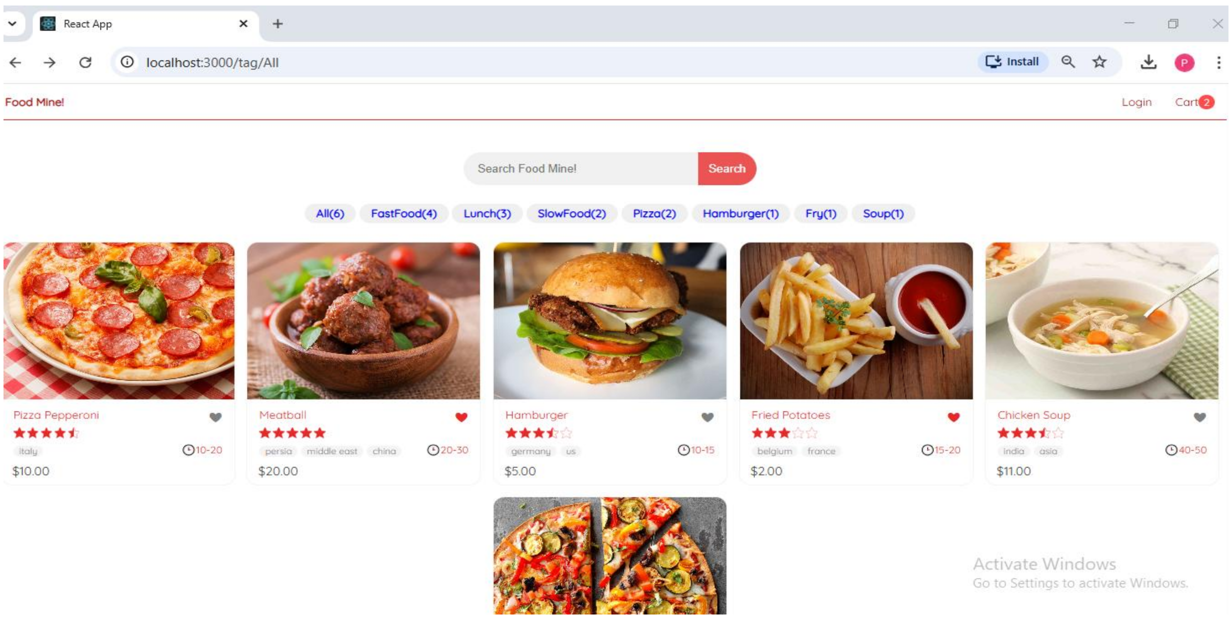
13. Future Enhancements

- Integrate online payment gateway (Razorpay / Stripe).
- Add advanced search and filtering (category, price range).
- Implement pagination or infinite scroll for better performance.
- Add order tracking with real-time status updates.
- Add email verification and forgot password functionality.
- Deploy frontend and backend to cloud platforms (Vercel/Netlify + Render/Railway).
- Improve UI design with animations and better UX.

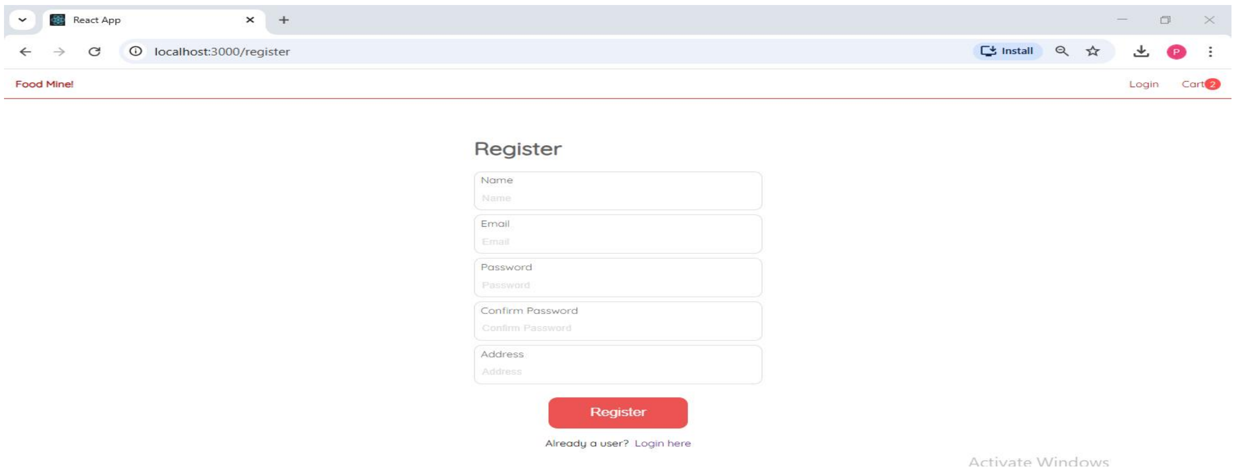
14 . Appendix A: UI Screenshots

The screenshots are captured from the running application on localhost and demonstrate the working system and role-based dashboards.

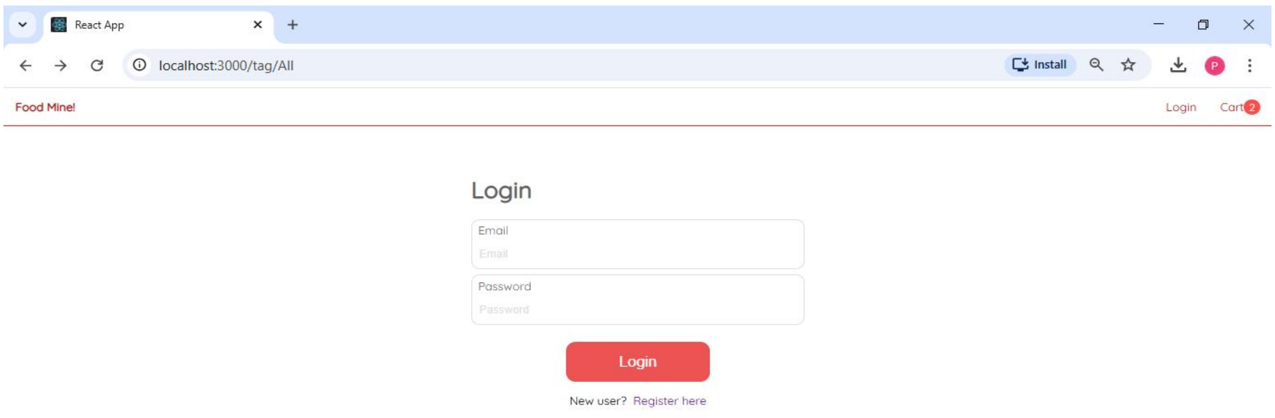
Home Page (Landing Screen):



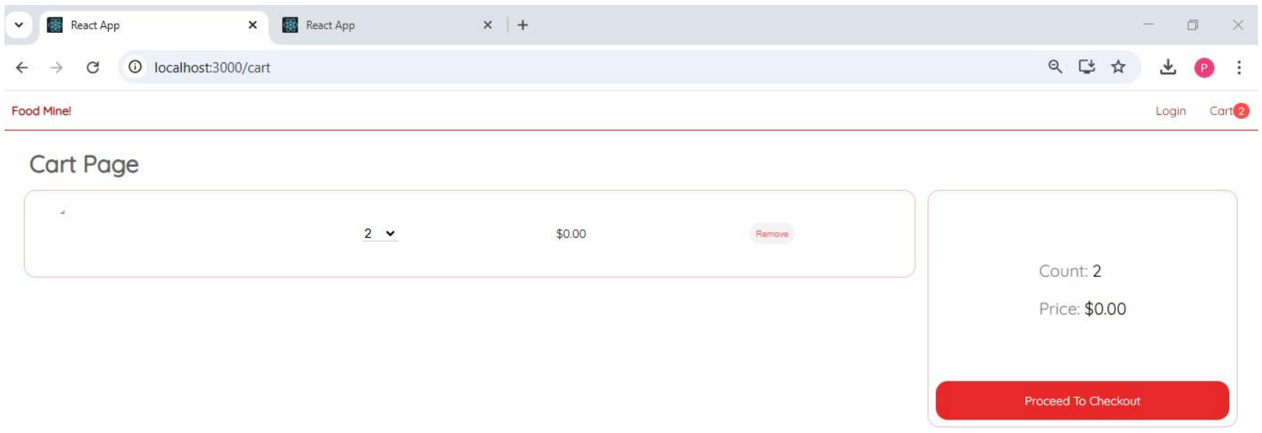
Registration Page:



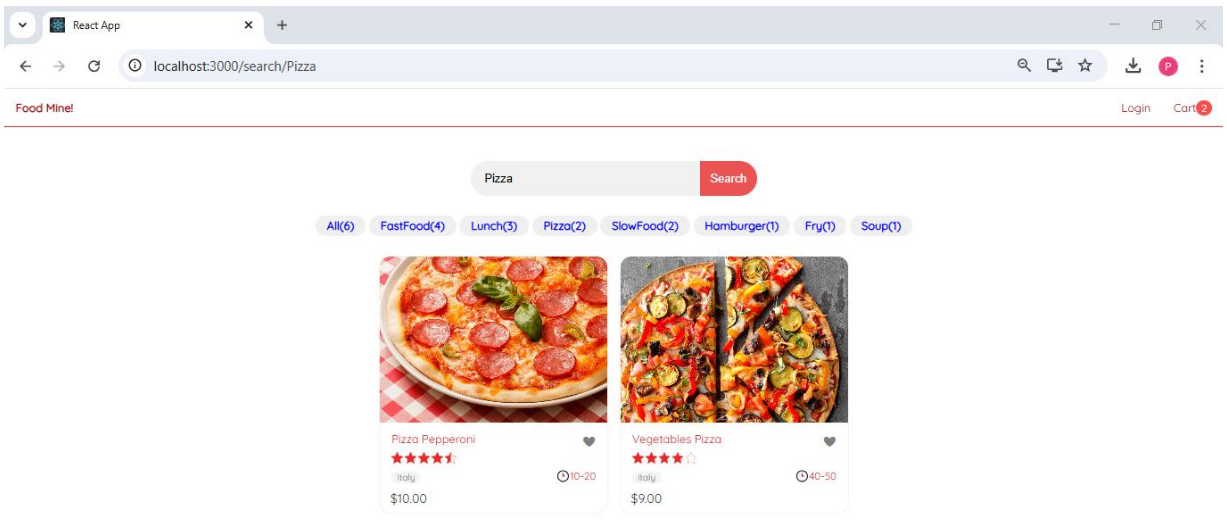
Login Page:

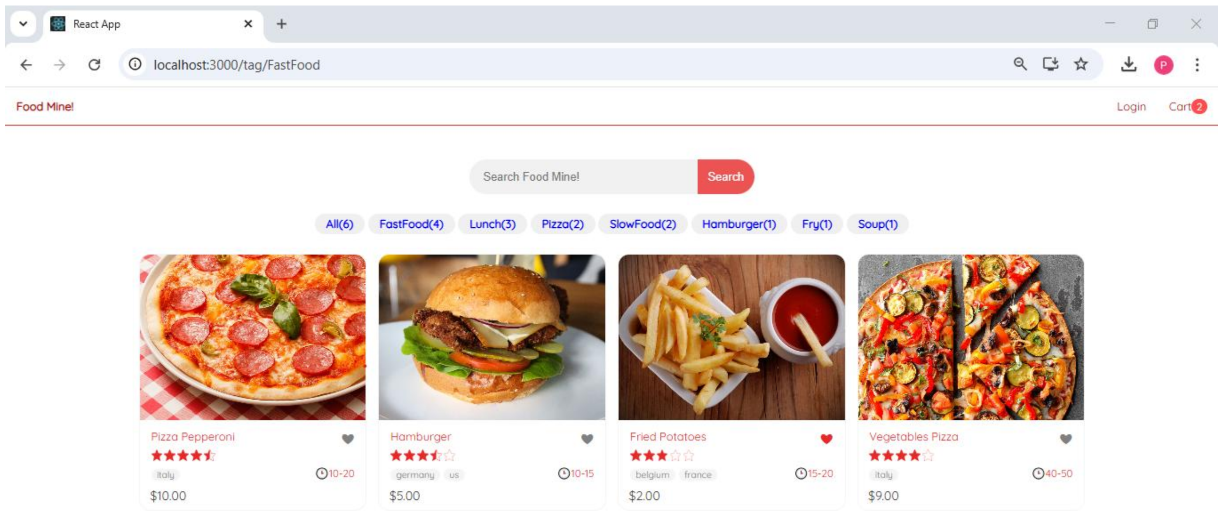


Cart Page :



Search page :


React App
localhost:3000/search/Pizza
Food Mine!
Login
Cart 2
Pizza
Search
All(6)
FastFood(4)
Lunch(3)
Pizza(2)
SlowFood(2)
Hamburger(1)
Fry(1)
Soup(1)
Pizza Pepperoni
italy
10-20
$10.00
Vegetables Pizza
italy
40-50
$9.00


React App
localhost:3000/tag/FastFood
Food Mine!
Login
Cart 2
Search Food Mine!
Search
All(6)
FastFood(4)
Lunch(3)
Pizza(2)
SlowFood(2)
Hamburger(1)
Fry(1)
Soup(1)
Pizza Pepperoni
italy
10-20
$10.00
Hamburger
germany
us
10-15
$5.00
Fried Potatoes
belgium
france
15-20
$2.00
Vegetables Pizza
italy
40-50
$9.00