Multi-Vendor Food Ordering Platform

Order from multiple restaurants in a single cart — Real-time tracking, vendor dashboards & delivery management

---

1. Problem Statement

Traditional food delivery apps usually allow ordering from only **one restaurant per order**, creating inconvenience for users who want variety and limiting revenue for restaurants.

This project solves that problem by building a **full-stack Multi-Vendor Food Ordering System** that allows:

✓ Ordering from **multiple restaurants in one cart**
✓ Real-time order tracking
✓ Vendor dashboards for menu & order management
✓ Delivery personnel module
✓ Admin control panel
✓ Secure payments

The system provides a seamless experience for **customers, restaurants, delivery personnel, and admins**.

---

2. Features

Customer Features

- Browse restaurants, cuisines, dishes
- Add items from **multiple restaurants** into one cart
- Apply filters, search, promotions
- Payment via Razorpay / Stripe / UPI
- Real-time order tracking
- Ratings & reviews

Restaurant/Vendor Features

- Vendor registration & login
- Menu CRUD with images
- Order acceptance/rejection
- Update order status:
    - Preparing → Ready → Out for delivery

- Sales analytics dashboard

Delivery Boy Features

- Accept/reject delivery tasks
- Live location tracking using Google Maps API
- Update delivery status

Admin Features

- Manage users, vendors, delivery partners
- Monitor orders, disputes, activity logs
- Generate reports

---

3. Tech Stack

Frontend

- React + Vite / Next.js
- TailwindCSS
- Axios
- Socket.IO client
- Zustand / Redux Toolkit

Backend

- Node.js + Express
- Prisma ORM
- PostgreSQL
- Socket.IO (real-time updates)
- JWT Authentication
- Multer / Cloudinary for image uploads

Infrastructure & Tools

- Render → Backend hosting
- Vercel → Frontend hosting
- PostgreSQL → Supabase / Railway
- GitHub → CI/CD
- Google Maps API
- Razorpay / Stripe payments

---

4. Folder Structure

```
.
├── apps
│   ├── frontend
│   │   ├── src
│   │   ├── public
│   │   ├── package.json
│   │   └── vite.config.js
│   └── backend
│       ├── src
│       │   ├── controllers
│       │   ├── routes
│       │   ├── middlewares
│       │   ├── utils
│       │   └── server.js
│       ├── prisma
│       │   └── schema.prisma
│       ├── package.json
├── README.md
├── .env.example
```

---

5. Setup Instructions

A. Clone Repository

git clone https://github.com/your-username/multi-vendor-food-app.git

cd multi-vendor-food-app

B. Install Dependencies

Frontend:

cd apps/frontend

npm install

Backend:

cd ../backend

npm install

C. Setup Environment Variables

Create .env inside **backend** folder:

DATABASE_URL=your_postgresql_url

JWT_SECRET=your_secret

CLOUDINARY_KEY=xxx

CLOUDINARY_SECRET=xxx

RAZORPAY_KEY_ID=xxx

RAZORPAY_KEY_SECRET=xxx

Frontend .env

VITE_API_URL=https://your-backend-url.onrender.com

D. Setup Database

cd apps/backend

npx prisma migrate dev --name init

npx prisma generate

E. Run Project Locally

Backend:

npm run dev

Frontend:

npm run dev

Project runs at:

- Frontend → http://localhost:5173
- Backend → http://localhost:5000

---

6. API Documentation (Summary)

Auth APIs

Method	Endpoint	Description
POST	/auth/register	User/Vendor registration
POST	/auth/login	Login & token generation

Restaurant APIs

Method	Endpoint	Description
GET	/restaurants	List all restaurants
GET	/restaurants/:id	Get restaurant details

Menu APIs

Method	Endpoint	Description
POST	/menu/add	Add menu item
GET	/menu/:restaurantId	Get menu items
PUT	/menu/:id	Update menu
DELETE	/menu/:id	Delete menu

Order APIs

Method	Endpoint	Description
POST	/orders	Create multi-vendor order
GET	/orders/user	User orders
PATCH	/orders/status	Update order status
GET	/orders/live/:id	Real-time tracking

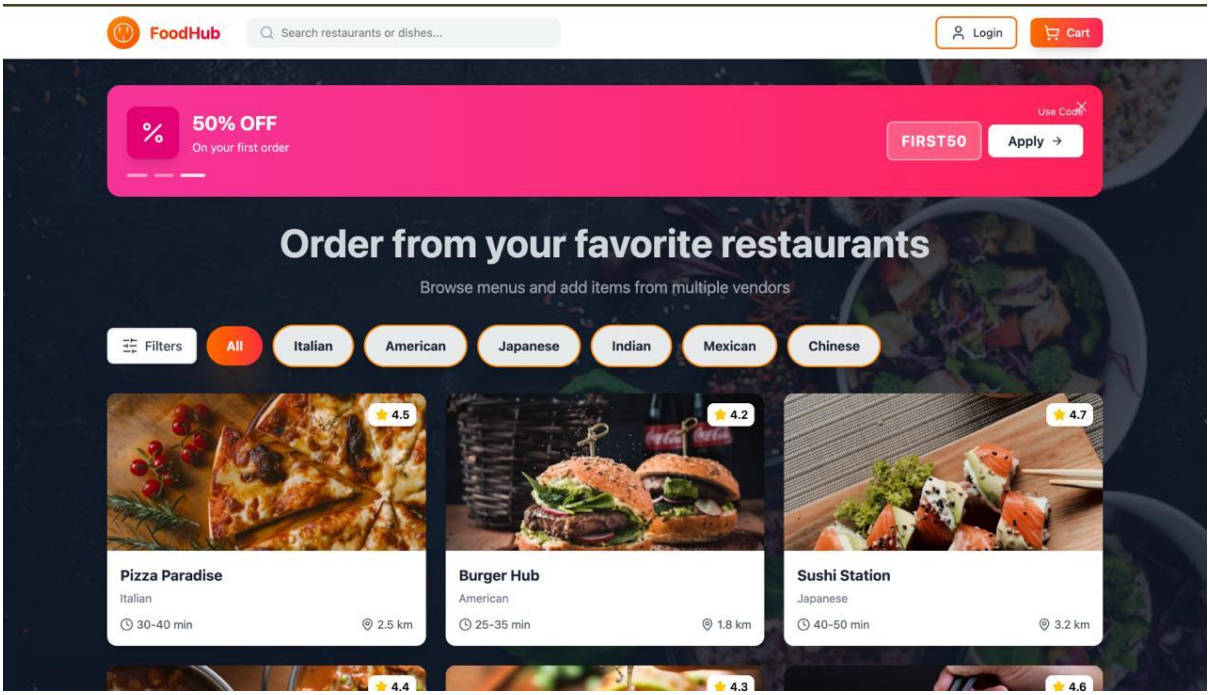
(You can ask me to generate the **full Swagger API docs**.)

---

7. Screenshots (Add in your project)

You should include screenshots like:

- Homepage


FoodHub
Search restaurants or dishes...
Login
Cart
50% OFF
On your first order
Use Code
FIRST50
Apply →
Order from your favorite restaurants
Browse menus and add items from multiple vendors
Filters
All
Italian
American
Japanese
Indian
Mexican
Chinese
4.5
Pizza Paradise
Italian
30-40 min
2.5 km
4.2
Burger Hub
American
25-35 min
1.8 km
4.7
Sushi Station
Japanese
40-50 min
3.2 km
4.4
4.3
4.6

- Cart

Your Cart

Pizza Paradise

Margherita Pizza
₹299

− 1 +

Pepperoni Pizza
₹399

− 1 +

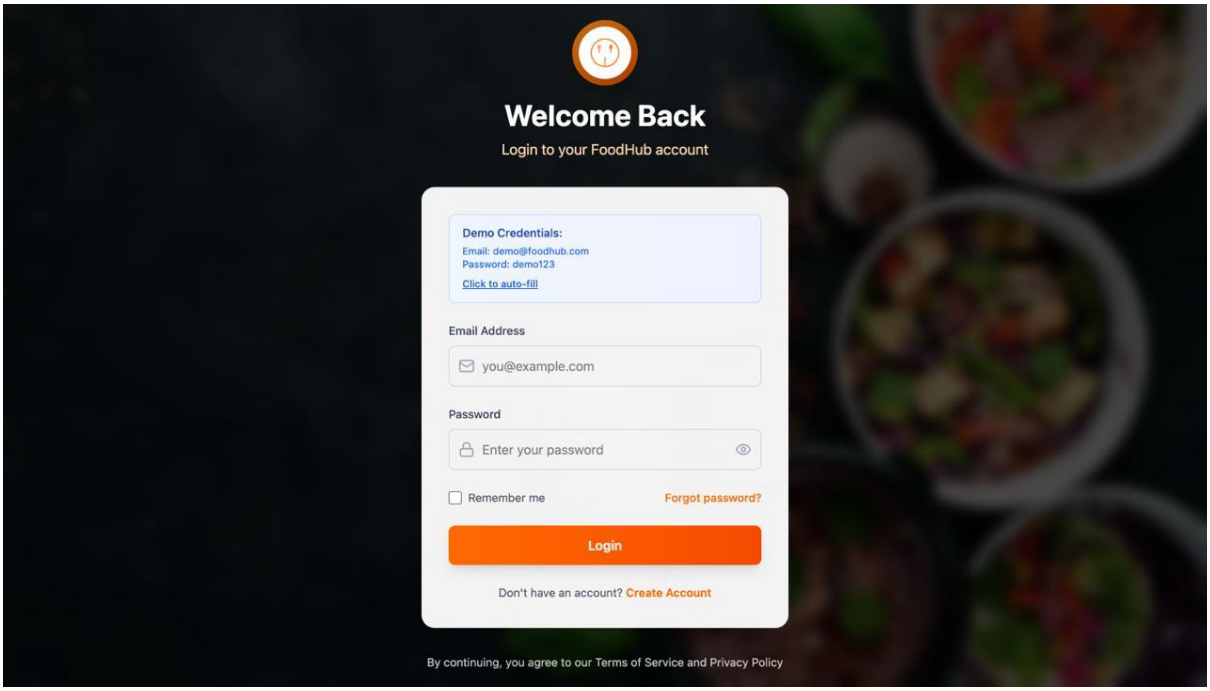
BBQ Chicken Pizza
₹449

− 1 +

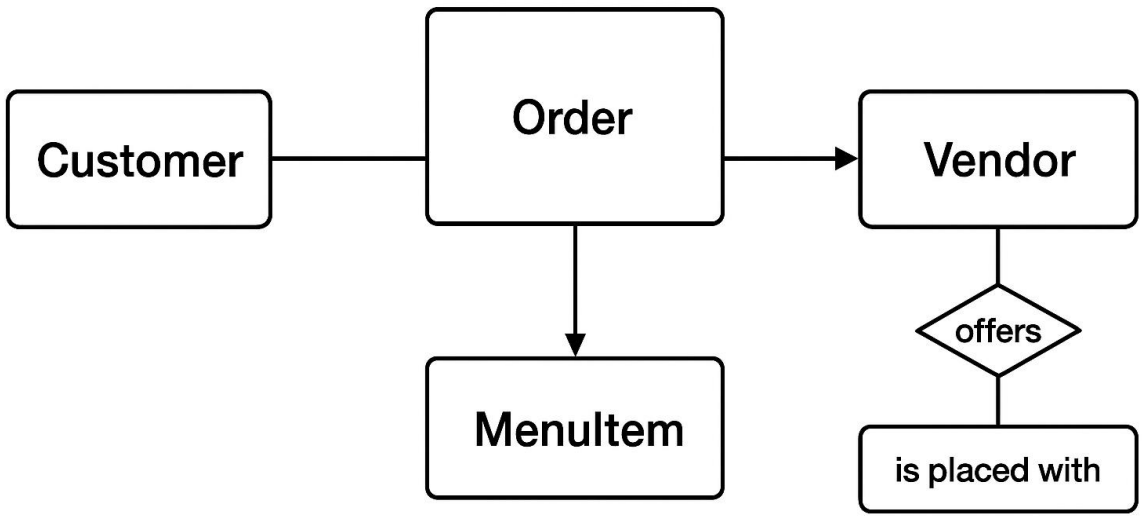
Subtotal ₹1147

Total ₹1147

Proceed to Checkout

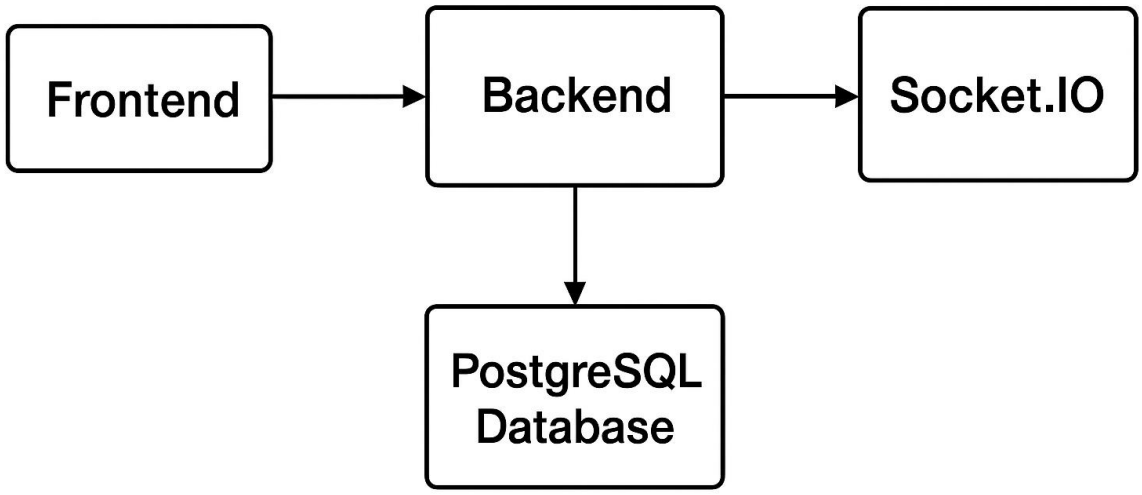


- ER diagram



- Architecture diagram

System Architecture



-

---

8. Database Schema (Prisma Example)

```
model User {
  id       String   @id @default(uuid())
  name     String
  email    String   @unique
  password String
  role     Role
  orders   Order[]
}

model Restaurant {
  id      String   @id @default(uuid())
  name    String
  address String
```

```
  menu    MenuItem[]
}

model MenuItem {
  id           String  @id @default(uuid())
  name         String
  price        Float
  restaurantId  String
  Restaurant    Restaurant @relation(fields: [restaurantId], references: [id])
}

model Order {
  id       String   @id @default(uuid())
  userId    String
  status    String
  items     Json
  total     Float
}
```

I can also generate a **full SQL dump** if needed.

---

9. System Architecture

Frontend (React/Next.js)

↓ API calls

Backend (Node.js + Express)

↓ Prisma ORM

PostgreSQL Database

↓

Socket.IO → Real-time order updates

---

10. Deployments

Frontend → Vercel

- Select apps/frontend as root
- Add VITE_API_URL
- Deploy

Backend → Render

- New Web Service
- Set root directory to apps/backend
- Add .env
- Deploy

---

11. Outcome

A production-ready, scalable, real-time **Multi-Vendor Food Ordering Platform** built for customers, vendors, delivery partners, and admins — with end-to-end functionality and deployment.