Backend Implementation Guide

1. Project Initialization

- Created Django project and app:

- Project: backend/

- App: backend/core/

2. Models (Database Structure)

- File: backend/core/models.py

- MenuItem: Stores menu items (name, description, price, category, status, image).

- Order: Stores customer orders (name, contact, table number, status, GST, etc.).

- OrderItem: Stores items within an order (linked to MenuItem and Order).

3. Serializers (API Data Formatting)

- File: backend/core/serializers.py

- Serializers for MenuItem, Order, and OrderItem to convert model data to JSON for APIs.

4. API Views (Endpoints Logic)

- File: backend/core/views.py

- MenuItemListView: Lists all menu items.

- OrderCreateView: Allows order creation.

- OrderDetailView: Retrieves order status/details.

5. URL Routing

- App URLs: backend/core/urls.py

- /api/menu/ — List menu items

- /api/orders/ — Create order

- /api/orders/<order\_id>/ — Get order status

- Project URLs: backend/urls.py

- Includes app URLs under /api/

- Configures media file serving for images

6. Admin Panel

- File: backend/core/admin.py

- Registers MenuItem, Order, and OrderItem for management via /admin

7. Sample Data

- File: backend/core/migrations/0002\_sample\_data.py

- Adds initial menu items for testing

8. Settings

- File: backend/settings.py

- Configures installed apps, media file handling, and database

9. Migrations

- Commands run:

- python manage.py makemigrations

- python manage.py migrate

- Loads sample data

10. Superuser Creation

- Command: python manage.py createsuperuser

- For admin access at /admin

Frontend Configuration & Integration Guide (Step-by-Step)

Step 1: Choose Your Frontend Stack

- Recommended: React (or Vue/Angular)

- Why? Easy to connect with REST APIs, lots of community support

// no need to perform this step

{ Step 2: Set Up Your Frontend Project

- React Example:

- Run: npx create-react-app foodcourt-frontend

- Go to folder: cd foodcourt-frontend }

Step 3: Install HTTP Client

- Install Axios (for API calls):

- Run: npm install axios

Step 4: Configure API URLs

- Create a config file (e.g., src/api.js):

js

export const API\_BASE = "http://localhost:8000/api/";

Step 5: Fetch Menu Items

- In your menu page/component:

- Use Axios to GET /api/menu/

- Display items, images, prices, etc.

Step 6: Order Submission

- In your order form/component:

- Collect user input (name, table number, selected items)

- POST to /api/orders/ with order data

Step 7: Show Order Status

- After order is placed:

- Save the returned order ID

- Use Axios to GET /api/orders/<order\_id>/ to show status

Step 8: Handle Images

- Menu item images: Use the image URL from the API to display images in your frontend.

Step 9: CORS Setup (if needed)

- If frontend and backend are on different ports:

- Install Django CORS headers: pip install django-cors-headers

- Add to INSTALLED\_APPS and middleware in backend/settings.py

- Set CORS\_ALLOW\_ALL\_ORIGINS = True (for development)

Step 10: Run Both Servers

- Backend: python manage.py runserver

- Frontend: npm start (in your React app folder)

Summary Table: Backend File/Topic Reference

Topic File/Path Purpose

Models backend/core/models.py Database structure

Serializers backend/core/serializers.py API data formatting

API Views backend/core/views.py API endpoint logic

App URLs backend/core/urls.py API endpoint routing

Project URLs backend/urls.py Main URL config, media setup

Admin Registration backend/core/admin.py Admin panel setup

Sample Data | backend/core/migrations/0002\_sample\_data.py | Initial menu items

Settings backend/settings.py Project config, media, apps, DB