Restaurant Management System

🧠 Project Overview1

This is a Java-based Restaurant Management System that operates via the terminal, incorporating role-specific functionalities for customers, staff, and administrators. It replicates real-world processes such as table reservations, order handling, kitchen operations, billing, and administrative oversight, utilizing JDBC to manage interactions with the database.

🚀 Project Goals

- Simulate a real-world restaurant workflow in code.
- Handle **multiple user roles** (Customer, Staff, Admin).
- Perform **CRUD operations** via a CLI menu-driven interface.
- Store and retrieve data using **PostgreSQL**.
- Follow **modular architecture** and **OOP principles**.

🧩 Logical Flow Breakdown

1. Customer Flow: Table Booking

- A customer can **book a table** by providing their name/contact/number of guests.
- Booking details are stored in the `table_booking` table with foreign keys to customer and branch.

2. Order Placement

- Customers can view a **menu** and place one or more items in an order.
- Items are added to the `order` table with a status of `PENDING`.
- Orders are linked to a table and booking ID.

3. Kitchen Workflow

- Kitchen staff can view all **pending order**
- Once prepared, they mark orders as `COMPLETED`.

- This simulates food preparation and readiness.

4. Billing System

- The billing module pulls all completed orders for a customer/table.
- Total bill is calculated based on the price of items and displayed.
- A new record is inserted into the `billing` table for audit/tracking.

5. Admin Controls

- Admin can:
    - Manage menu items (add/update/delete dishes).
    - View sales for custom day

GitHub repo : https://github.com/sharathtn_Zeta/restaurant-management-service

---

Instructions to run the entire project.

✅ 1. Prerequisites

Ensure the following are installed:

- **Java 11+**
- **PostgreSQL** (or your preferred RDBMS)
- **Maven** *(optional, if you want to manage dependencies)*
- Any Java IDE or terminal with `javac` and `java` CLI

📦 2. Database Setup

Step 1: Create a PostgreSQL Database

```
CREATE DATABASE restaurant_db;
```

Step 2: Create Tables and Triggers

Use the provided schema.sql file or run these manually:

```
-- Create customer table
CREATE TABLE public.customer (
        id serial PRIMARY KEY,
        name text NOT NULL,
        phone text NOT NULL
);
-- Create menu table
CREATE TABLE public.menu_item (
        id serial PRIMARY KEY,
        name text NOT NULL,
        price numeric(10, 2) NOT NULL
);
-- Create table_booking table
CREATE TABLE public.table_booking (
        id serial PRIMARY KEY,
        customer_id int REFERENCES public.customer(id) ON DELETE CASCADE,
        booking_time timestamp NOT NULL,
        num_people int NOT NULL,
```

```
    is_active boolean DEFAULT true
);

-- Create orders table
CREATE TABLE public.orders (
    id serial PRIMARY KEY,
    table_booking_id int REFERENCES public.table_booking(id) ON DELETE CASCADE,
    waiter_name text NOT NULL,
    status text DEFAULT 'Pending',
    created_at timestamp DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT orders_status_check CHECK (status IN ('Pending', 'Prepared', 'Paid'))
);

-- Create order_items table
CREATE TABLE public.order_items (
    id serial PRIMARY KEY,
    order_id int REFERENCES public.orders(id) ON DELETE CASCADE,
    menu_id int REFERENCES public.menu(id) ON DELETE CASCADE,
    quantity int NOT NULL CHECK (quantity > 0)
);

-- Create payments table
CREATE TABLE public.payments (
    id serial PRIMARY KEY,
    booking_id int REFERENCES public.table_booking(id) ON DELETE CASCADE,
    amount numeric,
```

```
    payment_method varchar(20),
    payment_time timestamp DEFAULT now()
);
```

🧠 3. Update Database Credentials

Open the `DatabaseConnection.java` file and update the following:

```
String url = "jdbc:postgresql://localhost:5432/restaurant_db";
String user = "your_db_username";
String password = "your_db_password";
```