Restaurant Reservation System

Overall Purpose: The system stores and manages customer reservations for restaurants. It tracks users, restaurant details, operating hours, and reservations through a clean relational structure.

AppUser Table
• Stores information about users who make reservations.
• Each user is uniquely identified by id (Primary Key).
• Email must be unique to prevent duplicate accounts.
• Contains name, phone number, and password hash.
• One user can have many reservations (1 → ∞).

Restaurant Table
• Stores details for each restaurant available in the system.
• Each restaurant has a unique id (Primary Key).
• Includes name, address, city, state, and timestamps.
• One restaurant can have multiple reservations (1 → ∞).
• One restaurant can have multiple schedule entries (1 → ∞).

RestaurantSchedule Table
• Stores each restaurant's opening and closing times for each day of the week.
• Fields: weekday (0 = Monday → 6 = Sunday), open_time, close_time.
• restaurant_id links to Restaurant (Foreign Key).
• One restaurant → many schedule entries (1 → ∞).
• Unique (restaurant_id, weekday) ensures one record per day.

Reservation Table
• Stores each booking made by a user for a specific restaurant.
• Each reservation is uniquely identified by id (Primary Key).
• Foreign keys: user_id (AppUser), restaurant_id (Restaurant).
• Tracks: reservation_at (date/time), party_size, status, confirmation_code, special_requests.•
Rules: confirmation_code unique, party_size > 0, valid user & restaurant references.

App Start & Stop

Action	Command / Steps
Run app	./mvnw spring-boot:run
Stop app	Ctrl + C in terminal

Rebuild after changes	./mvnw clean package then ./mvnw spring-boot:run
Change port	In application-dev.yml → server.port: 8081 (or any free port)

Localhost URLs

Resource	URL
Main app (default)	http://localhost:8080
H2 console	http://localhost:8080/h2-console
Health check	http://localhost:8080/actuator/health
API examples	http://localhost:8080/api/restaurants, /api/reservations

Database Console (H2)

Setting	Value
JDBC URL	jdbc:h2:mem:resdb
User Name	temp
Password	strongpassword
Console Path	/h2-console

Profile Management

Environment Dev (H2) MySQL Postgres

Profile	How to activate
dev	Default in application.yml
mysql	SPRING_PROFILES_ACTIVE=mysql ./mvnw spring-boot:run

postgres	SPRING_PROFILES_ACTIVE=postgres ./mvnw spring-boot:run

Useful Files

File application.yml Purpose
Sets the active profile

application-dev.yml	H2 development configuration
schema.sql / data.sql	Define and seed the database schema
pom.xml	Maven dependencies and build settings
ReservationApplication.java	Main Spring Boot entry point

When Something Breaks

Symptom	Quick Fix
Site can’t be reached	App not running → rerun ./mvnw spring-boot:run
Port already in use	lsof -i :8080 → kill -9 <PID>
Old username showing	Clear browser autofill on /h2-console
Config not applying	Check active profile in application.yml, then restart