TastyTable Backend Project - Code Documentation

# 1. Project Overview

TastyTable is a monolithic Web API built with ASP.NET Core 8, Entity Framework Core (MySQL), and JWT authentication. It follows a clean architecture with separation of concerns across layers: Api, Core, Services, Data, and Tests. The project provides a backend system for managing menu items, user authentication, and customer orders, with support for role-based access control (Admin vs User).

# 2. Project Structure

The solution is organized into the following projects:  
  
• TastyTable.Api → Controllers (API endpoints, Swagger config, Program.cs)  
• TastyTable.Core → Entities (models), DTOs, Interfaces  
• TastyTable.Services → Business logic (services: User, Menu, Orders)  
• TastyTable.Data → EF Core DbContext, Repositories, Database migrations  
• TastyTable.Tests → Unit tests with xUnit + Moq

# 3. Layer Responsibilities

• API Layer:  
 - Hosts controllers (AuthController, MenuController, OrdersController)  
 - Maps HTTP routes to service calls  
 - Configures Swagger, Authentication, and Middleware  
  
• Core Layer:  
 - Defines Entities: User, MenuItem, Order, OrderItem  
 - Contains DTOs for requests (LoginRequest, RegisterRequest, CreateOrderRequest)  
 - Defines Interfaces for services (IUserService, IMenuService, IOrderService)  
  
• Services Layer:  
 - Implements business logic  
 - UserService: Handles registration, login, JWT token generation  
 - MenuService: Manages menu CRUD and availability  
 - OrderService: Creates and fetches orders for users  
  
• Data Layer:  
 - AppDbContext: EF Core DbContext with DbSets for Users, MenuItems, Orders  
 - Configured with MySQL (Pomelo.EntityFrameworkCore.MySql)  
 - Handles migrations and seeding demo data  
  
• Tests Layer:  
 - Includes xUnit tests  
 - Moq used to mock JwtTokenService and isolate services

# 4. Key Files and Their Importance

## Program.cs

Entry point for the API. Configures services, DbContext, Authentication, Swagger, and database seeding.

## AuthController.cs

Manages authentication endpoints: register and login. Generates JWT tokens.

## MenuController.cs

CRUD for menu items. Admin-only for create/update, public for list.

## OrdersController.cs

Handles order creation and retrieval. Requires authenticated user.

## JwtTokenService.cs

Generates JWT tokens with user claims (ID, role, username).

## AppDbContext.cs

DbContext for EF Core with DbSets for Users, MenuItems, Orders, OrderItems.

## DemoDataSeeder.cs

Seeds default Admin user and sample menu items.

## UserService.cs

Business logic for user registration, login, and password hashing (BCrypt).

## MenuService.cs

Business logic for menu CRUD and availability toggling.

## OrderService.cs

Business logic for order creation and retrieval, including totals.

## Tests (xUnit + Moq)

Unit tests validating login, menu, and orders with mocking of JwtTokenService.

# 5. Integration Flow

1. User registers → stored in database  
2. User logs in → receives JWT token  
3. Token is attached in Authorization header for protected requests  
4. Admin can create menu items and toggle availability  
5. Users can browse menu and place orders  
6. Orders are linked to user and persisted in database

# 6. Importance of Each Component

• API Layer → Exposes endpoints, integrates Swagger, auth middleware  
• Core Layer → Defines business entities and interfaces  
• Services Layer → Business logic (users, menu, orders)  
• Data Layer → Database integration with EF Core + MySQL  
• JwtTokenService → Handles authentication and security  
• Tests → Ensure correctness and quality  
• Docker + AWS → Deployment ready

# 7. Conclusion

The TastyTable project is a complete backend solution demonstrating skills in ASP.NET Core, Entity Framework Core, JWT authentication, testing, and deployment. It fulfills client requirements and is suitable for extension with frontends or mobile apps.