Пояснение к коду:

Архитектура системы:

Контроллеры:

- UserController: Контроллер, отвечающий за регистрацию/авторизацию/получение информации по пользователе.
- OrderController: Контроллер, обрабатывающий запросы, связанные с заказами и меню.

DTO объекты

- DishCreationDto: Объект передачи данных (DTO) для блюда, используемый для передачи информации о блюде между клиентом и сервером при запросе на создание блюда.
- DishDelitionDto: Объект передачи данных (DTO) для блюда, используемый для передачи информации о блюде между клиентом и сервером при запросе на удаление блюда.
- OrderCreationDto: Объект передачи данных (DTO) для заказа, используемый для передачи информации о заказе между клиентом и сервером при запросе на создание заказа.
- UserAuthorizationDto: Объект передачи данных (DTO) для пользователя, используемый для передачи информации о пользователе между клиентом и сервером при запросе на авторизацию.
- UserRegistrationDto: Объект передачи данных (DTO) для пользователя, используемый для передачи информации о пользователе между клиентом и сервером при запросе на регистрацию.

Модели для БД таблиц:

- Dish: Модель блюда, представляющая информацию о блюде в меню ресторана.
- Order: Модель заказа, представляющая информацию о заказе в системе.
- OrderDish: Модель заказа блюда, представляющая информацию о конкретном заказанном блюде.
- Session: Модель сеанса, представляющая информацию о сеансе аутентификации пользователя.
- User: Модель пользователя, представляющая информацию о пользователе в системе.

Выбрасываемые исключения

(Их довольно много, но они с говорящими названиями, так что не буду перечислять их).

Модели

- ERole - отдельный Enum, в котором хранятся все роли.
- EStatus - отдельный Enum, в котором хранятся все статусы заказов.

Репозитории

- DishRepo - репозиторий для работы с таблицей БД dish.
- OrderRepo - репозиторий для работы с таблицей БД orders.
- SessionRepo - репозиторий для работы с таблицей БД session.
- UserRepo - репозиторий для работы с таблицей БД user_info.

Сервисы(часть нужна для того, чтобы не писать бизнесс-логику в контроллерах)

- EmailValidator - проверка почты на корректность
- JwtService - класс по работе с JWT токенами

- OrderService - класс по обработке входящих запросов 2-ого микросервиса
- UserService - класс по обработке взодящих запросов 1-ого микросервиса

Вцелом система разбита на 2 микросервиса, которые можно спокойно вынести в 2 отдельных файла, но поскольку я не до конца конял, почему в условии 2 микросервис напрямую использует БД 1-ого микросервиса, то я решил объединить их в один проект, при этом полностью отделив функционал друг от друга.

Также есть папка с POSTMAN тестами на основные команды.