

Студент: Маринченко Игорь Игоревич

Группа: БПИ218

Домашнее задание №4

Система обработки заказов ресторана

Цель:

Разработать два отдельных микросервиса на основе RESTful API для системы обработки заказов в ресторане, первый из которых реализует авторизацию пользователей с различными ролями, а второй – управляет заказами и отслеживает запас блюд.

Используемые средства:

Задание реализовано на языке программирования Python 3.8 с использованием фреймворка Flask. В качестве веб-сервера используется Gunicorn – Python WSGI веб-сервер для UNIX систем. Приложение разворачивается в системе на основе механизма контейнеризации с использованием Docker и Docker-compose. Для каждого микросервиса создается свой экземпляр базы данных PostgreSQL в отдельном контейнере.

Механизм авторизации и аутентификации пользователя:

Когда пользователь регистрируется или авторизуется, в качестве ответа сервера вместе с прочей информацией он получает JWT токен (каждый раз новый), в котором содержится уникальный идентификатор пользователя и его роль: клиент или менеджер. Также сервер автоматически встраивает JWT токен в cookie пользователя. В токен встроен срок его истечения, после чего он становится не валидным.

После того, как пользователь получил токен, он получает доступ ко всем ресурсам обоих микросервисов. Всю необходимую информацию о личности пользователя они получают из токена. Токен может располагаться, как внутри cookie, так и в заголовках запроса.

После выхода пользователя из аккаунта, токен автоматически удаляется из его cookie.

Формат ответов API:

Любой ответ API приходит в JSON формате. Любой ответ содержит поле “status”, которое либо равно “OK” в случае успешного запроса, либо равно “failed” в случае некорректного запроса. Во втором случае в ответе также присутствует поле “error”, которое содержит информацию об ошибке.

HTTP статусы ответов:

- 1) Ответы на успешные запросы имеют статус 200.
- 2) Ответы на некорректные запросы имеют статус 400.
- 3) Ответы на неавторизованные запросы (JWT токен отсутствует, либо устарел) имеют статус 401.

Микросервис авторизации пользователей

Конечные точки:

- 1) `/api/register` (POST) – для регистрации нового пользователя

Пример запроса:

```
{
  "first_name": "Name",
  "last_name": "Name",
  "login": "user",
  "email": "email@email.com",
  "password": "Password1",
  "password_repeat": "Password1",
  "role": "Client"
}
```

Пример ответа:

```
{
  "status": "OK",
  "user": {
    "id": 1,
    "login": "user",
    "first_name": "Name",
    "last_name": "Name",
    "email": "email@email.com",
    "password": "$2b$15$7X3sN1ZHub.wp8Ep1em6NOe5SBZkGa41JdlUu.5...",
    "role": "CLIENT",
    "jwt_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmcmVz...",
    "token_expiration_time": "2023-05-29 01:10:22",
    "orders": []
  }
}
```

Пример ответа:

```
{
  "status": "failed",
  "error": "пользователь с таким именем пользователя уже существует"
}
```

2) /api/login (POST) – для авторизации пользователя

Пример запроса:

```
{  
  "login_or_email": "user1",  
  "password": "Pass"  
}
```

Пример ответа:

```
{  
  "status": "failed",  
  "error": "пользователя с таким именем пользователя/email не существует"  
}
```

3) /api/logout (GET) – для выхода пользователя из аккаунта

Пример ответа:

```
{  
  "status": "OK"  
}
```

4) /api/user (GET) – для просмотра сведений о пользователе

Пример ответа:

```
{  
  "status": "OK",  
  "user": {  
    "id": 1,  
    "login": "user",  
    "first_name": "Name",  
    "last_name": "Name",  
    "email": "email@email.com",  
    "password": "$2b$15$7X3sN1ZHub.wp8Ep1em6NOe5SBZkGa41JdlUu.5...",  
    "role": "CLIENT",  
    "jwt_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmcmVz...",  
    "token_expiration_time": "2023-05-29 01:10:22",  
    "orders": []  
  }  
}
```

При любом запросе сведений о пользователе, микросервис авторизации запрашивает у микросервиса заказов все заказы текущего пользователя и включает их в свой ответ.

Микросервис обработки заказов

Конечные точки:

- 1) `/api/new-order` (POST) - для создания нового заказа

Пример запроса:

```
{
  "special_requests": "",
  "dishes": [
    {
      "id": 7,
      "quantity": 2,
      "total_price": "500.00"
    },
    {
      "id": 10,
      "quantity": 1,
      "total_price": "80.35"
    }
  ]
}
```

Пример ответа:

```
{
  "status": "OK",
  "order": {
    "id": 1,
    "status": "created",
    "special_requests": "",
    "order_dishes": [
      {
        "id": 1,
        "dish_id": 7,
        "quantity": 2,
        "total_price": 500,
        "name": "Бокал",
        "description": "Пустой",
        "created_at": "2023-05-29 01:41:55",
        "updated_at": "2023-05-29 01:41:55"
      },
      {
        "id": 2,
        "dish_id": 10,
        "quantity": 1,
        "total_price": 80.35,

```

```

    "name": "Щи",
    "description": "Кислые",
    "created_at": "2023-05-29 01:41:55",
    "updated_at": "2023-05-29 01:41:55"
  }
],
"created_at": "2023-05-29 01:41:55",
"updated_at": "2023-05-29 01:41:55"
}
}

```

2) **/api/order?id=** (GET) – для просмотра сведений о конкретном заказе

3) **/api/dishes** (GET) – для просмотра списка доступных блюд

4) **/api/orders** (GET) – для просмотра всех заказов пользователя

Обработчик заказов:

Извлекает из базы данных заказы в статусе “created”, и готовит их в течении некоторого времени (в этот момент статус заказа равен “cooking”). После чего статус заказа становится равен “completed”.

Настройки приложения:

Настройки приложения реализованы через переменные виртуального окружения, которые Docker-compose импортирует из файла .env в корне проекта.

Список переменных и их значения по умолчанию:

```

# Порт микросервиса авторизации
AUTHORIZATION_MS_PORT=5001
# Порт микросервиса заказов
ORDER_MS_PORT=5002

# Количество экземпляров сервера микросервиса авторизации
AUTHORIZATION_MS_WORKERS_COUNT=3
# Количество экземпляров сервера микросервиса заказов
ORDER_MS_WORKERS_COUNT=3

# Время жизни JWT токена в минутах
JWT_TOKEN_LIFETIME=1
# Минимальное время приготовления заказа в секундах
MIN_ORDER_COOKING_TIME=30
# Максимальное время приготовления заказа в секундах
MAX_ORDER_COOKING_TIME=90
# Заполнять ли таблицу блюд тестовыми данными
FILL_DISHES_TABLE_WITH_EXAMPLE_DATA=false

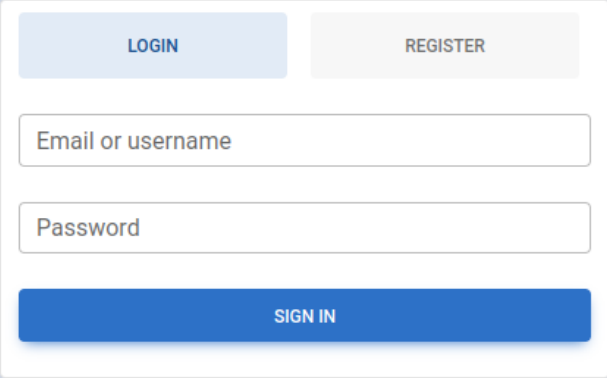
```

FRONTEND

Также на втором микросервисе был реализован frontend для интерактивного взаимодействия с API.

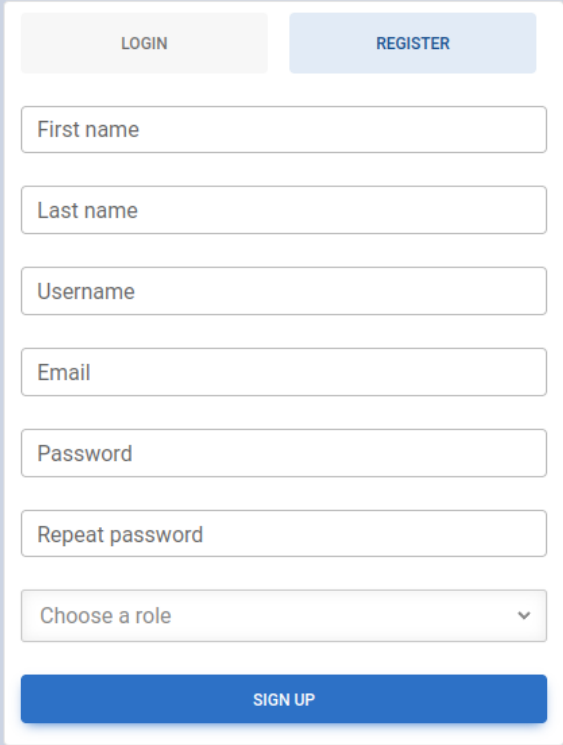
Конечные точки:

1) **/login** – для авторизации пользователя



The login form is displayed on a light blue background. It features a white container with two buttons at the top: 'LOGIN' (blue) and 'REGISTER' (light gray). Below these are two input fields: 'Email or username' and 'Password'. At the bottom is a large blue button labeled 'SIGN IN'.

2) **/register** – для регистрации нового пользователя



The register form is displayed on a light blue background. It features a white container with two buttons at the top: 'LOGIN' (light gray) and 'REGISTER' (blue). Below these are six input fields: 'First name', 'Last name', 'Username', 'Email', 'Password', and 'Repeat password'. At the bottom is a dropdown menu labeled 'Choose a role' with a downward arrow. At the very bottom is a large blue button labeled 'SIGN UP'.

3) /dish – для управления блюдами (доступна только менеджерам)

List

Create

Name *

Description

Price *

Quantity *

Is Available

☐

Created At

Updated At

Save

Save and Add Another

Save and Continue Editing

Cancel

ВЫХОД

4) /create-order – для создания заказов (доступна только клиентам)

Меню

Доступно 10 позиции

Зло-Кола

Напиток (0.01мл)

-

0

+

Р 2300.00

Картофель

В мундире (11 клубней)

-

1

+

Р 111.11

Котлетки

С макаронками

-

2

+

Р 120.00

Котлеты

С пюрешкой

-

1

+

Р 130.00

Чизбургер

Просто бургер, но с

-

0

+

Р 59.90

Итого

Блюд 4

ДОПОЛНИТЕЛЬНЫЕ ПОЖЕЛАНИЯ

СТОИМОСТЬ Р 481.11

ОФОРМИТЬ ЗАКАЗ