

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа 4: Документирование API из ЛР2
средствами Swagger, Postman

Выполнил:
Ле Хоанг Чыонг

Группа:
К33392

Проверил:
Добряков Д. И.

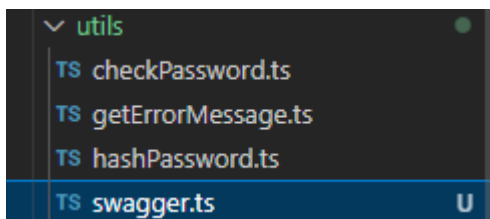
Санкт-Петербург
2024 г.

Задача

Документирование API из ЛР2 средствами Swagger, Postman

Ход работы

1. В папке `utils` добавьте файл `swagger.ts`, он содержит конфигурацию Swagger и настройку документации для приложения Express.



```
import express from 'express';
import swaggerJsdoc from 'swagger-jsdoc';
import swaggerUi from 'swagger-ui-express';

const router = express.Router();

const options = {
  swaggerDefinition: {
    openapi: '3.0.0',
    info: {
      title: 'API Documentation',
      version: '1.0.0',
      description: 'Documentation for API endpoints',
    },
    components: {
      securitySchemes: {
        bearerAuth: {
          type: 'http',
          scheme: 'bearer',
          bearerFormat: 'JWT'
        }
      }
    },
    security: [{
      bearerAuth: []
    }]
  },
  apis: ['./src/routes/v1/**/*.ts'],
};

const specs = swaggerJsdoc(options);
router.use('/api-docs', swaggerUi.serve);
router.get('/api-docs', swaggerUi.setup(specs));

export default router;
```

2. Затем смонтируйте маршрутизатор документации Swagger по определенному пути в приложении Express.

```
app.use('/', swaggerRouter);
```

3. В файлах route (например, Activity.ts) вам необходимо добавить комментарии в формате Swagger для создания документации.

Например:

```
import express from "express";
import ActivityController from "../../controllers/activities/activity";

const router = express.Router();
const controller = new ActivityController();

/**
 * @swagger
 * tags:
 *   name: Activity
 *   description: Operations related to activities
 */

/**
 * @swagger
 * /v1/activities:
 *   post:
 *     summary: Create a new activity
 *     tags: [Activity]
 *     requestBody:
 *       required: true
 *       content:
 *         application/json:
 *           schema:
 *             type: object
 *             properties:
 *               name:
 *                 type: string
 *     responses:
 *       201:
 *         description: Activity created successfully
 *       400:
 *         description: Bad request, missing or invalid parameters
 *       500:
 *         description: Internal server error
 */
router.post('/', controller.create);
```

4. Результаты







Документация по API по ссылке: <http://localhost:8000/api-docs/>

API Documentation 1.0.0 OAS 3.0







Documentation for API endpoints

Authorize 









Activity Operations related to activities

POST	/v1/activities	Create a new activity		
GET	/v1/activities	Get all activities		
GET	/v1/activities/{id}	Get an activity by ID		
DELETE	/v1/activities/{id}	Delete an activity by ID		









Authentication Operations related to user authentication

POST	/v1/login	Authenticate user		
POST	/v1/register	Register a new user		
POST	/v1/refresh-token	Refresh user token		











LocationActivity Operations related to location activities

POST	/v1/locations-activities	Create a new location activity		
GET	/v1/locations-activities/location/{locationId}	Get all location activities by location ID		
GET	/v1/locations-activities/activity/{activityId}	Get all location activities by activity ID		
DELETE	/v1/location-activities/{id}	Delete a location activity by ID		











Location Operations related to locations

POST	/v1/locations	Create a new location		
GET	/v1/locations	Get all locations		
GET	/v1/locations/{id}	Get a location by ID		
DELETE	/v1/locations/{id}	Delete a location by ID		

Offer Operations related to offers

POST	/v1/offers	Create a new offer		
GET	/v1/offers	Get all offers		
GET	/v1/offers/me	Get offer for current user		
GET	/v1/offers/{id}	Get an offer by ID		
DELETE	/v1/offers/{id}	Delete an offer by ID		

Review Operations related to reviews

POST	/v1/reviews	Create a new review		
GET	/v1/reviews	Get all reviews		
GET	/v1/reviews/{id}	Get a review by ID		
DELETE	/v1/reviews/{id}	Delete a review by ID		
PUT	/v1/reviews/{review_id}	Update a review by ID		

Trip Location <small>Operations related to trip locations</small>			^
POST	/v1/trip-locations	Create a new trip location	🔒 ▼
GET	/v1/trip-locations	Get all trip locations	🔒 ▼
GET	/v1/trip-locations/location	Get all trip locations by location ID	🔒 ▼
GET	/v1/trip-locations/trip	Get all trip locations by trip ID	🔒 ▼
DELETE	/v1/trip-locations/{id}	Delete a trip location by ID	🔒 ▼
Trip <small>Operations related to trips</small>			^
POST	/v1/trips	Create a new trip	🔒 ▼
GET	/v1/trips	Get all trips	🔒 ▼
GET	/v1/trips/trip	Get a trip by ID	🔒 ▼
GET	/v1/trips/trip-query	Get trips by filtering	🔒 ▼
DELETE	/v1/trips/{id}	Delete a trip by ID	🔒 ▼
PUT	/v1/trips/{trip_id}	Update a trip by ID	🔒 ▼
User Activity <small>Operations related to user activities</small>			^
POST	/v1/user-activities	Create a new user activity	🔒 ▼
GET	/v1/user-activities	Get all user activities	🔒 ▼
DELETE	/v1/user-activities	Delete a user activity	🔒 ▼
GET	/v1/user-activities/interests	Get user interests	🔒 ▼
User <small>Operations related to users</small>			^
GET	/v1/users/me	Get current user details	🔒 ▼
GET	/v1/users/recommend-location	Get recommended locations for the current user	🔒 ▼
GET	/v1/users	Get all users	🔒 ▼
PATCH	/v1/users	Update current user details	🔒 ▼
DELETE	/v1/users	Delete current user	🔒 ▼

Вывод

Во время домашней работы я научился автоматически генерировать документацию по API с помощью Swagger.