САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 3

Выполнил:
Байков Иван
К33392

Проверил:
Добряков Д. И.

Санкт-Петербург

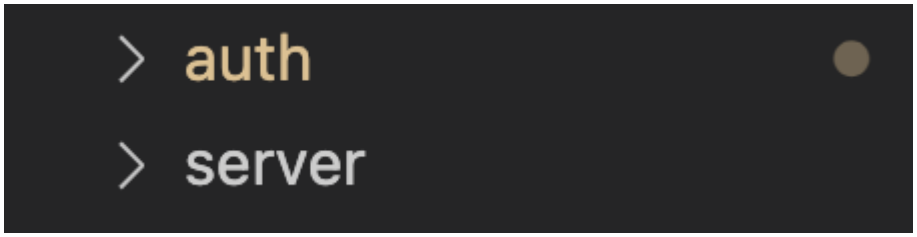
2024 г.

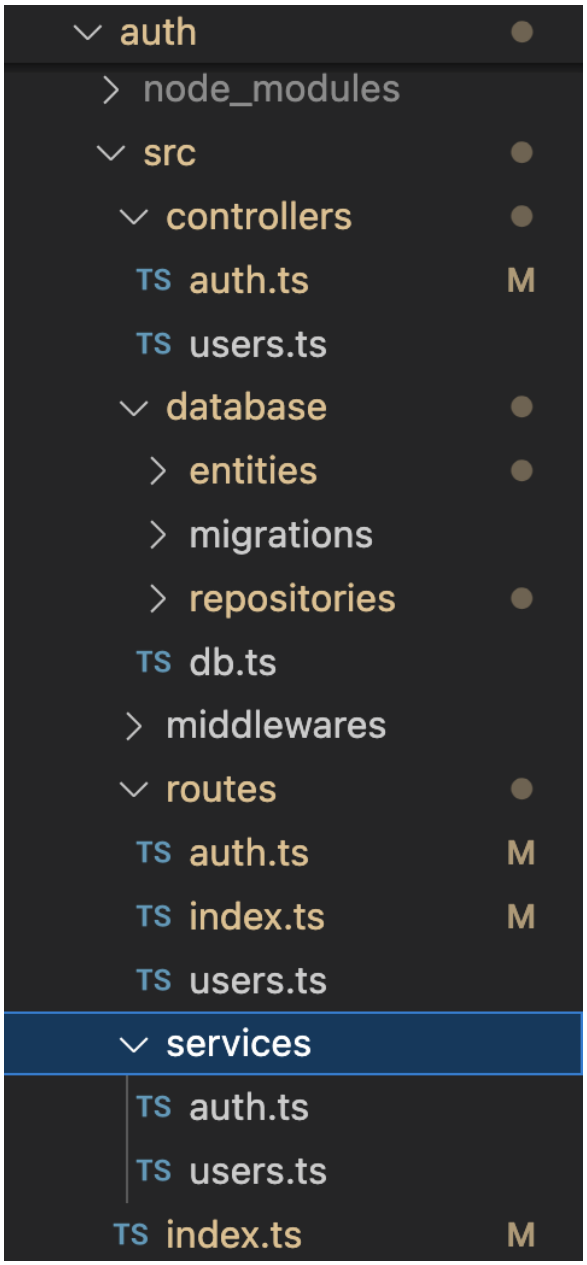
Задача

Необходимо реализовать отдельный микросервис, выполняющий какую-либо содержательную функцию из всего арсенала функций вашего приложения.

Ход работы

1) Вынесем функционал авторизации в отдельный микросервис





2) Обосомим модели в бд от других моделей

auth service/

```ts
import { Entity, PrimaryGeneratedColumn, Column, OneToMany } from "typeorm";

@Entity()
export class Users {
  @PrimaryGeneratedColumn()
  id: number;

  @Column({ type: "varchar", unique: true, nullable: false })
  email: string;

  @Column({ type: "varchar", select: false })
  password: string;
}
```

server/

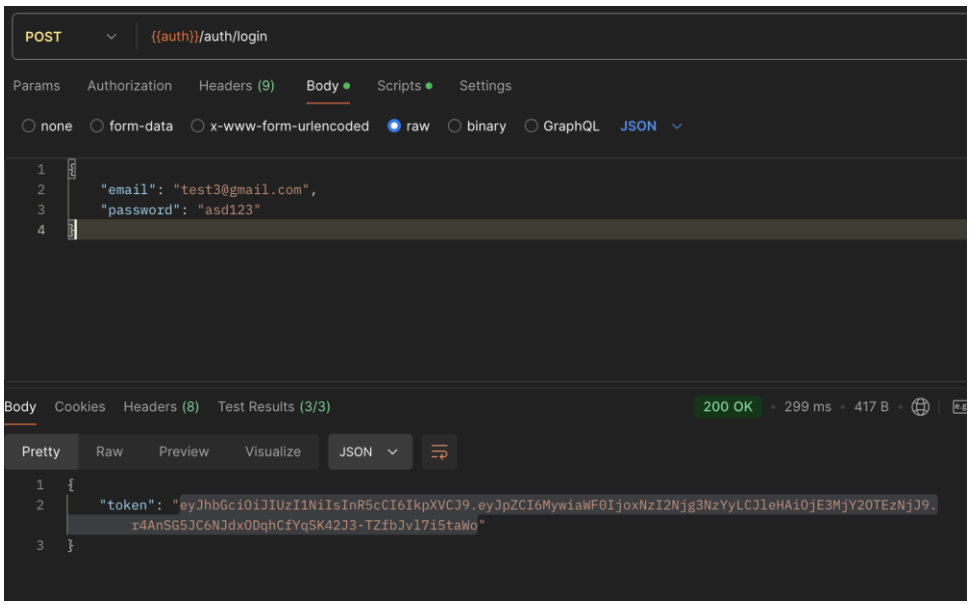
```ts
import { Entity, PrimaryGeneratedColumn, Column, ManyToOne } from "typeorm";

@Entity()
export class Books {
  @PrimaryGeneratedColumn()
  id: number;

  @Column({ type: "varchar" })
  title: string;

  @Column({ type: "varchar" })
  author: string;

  @Column({ type: "int" })
  owner: number;
}
```

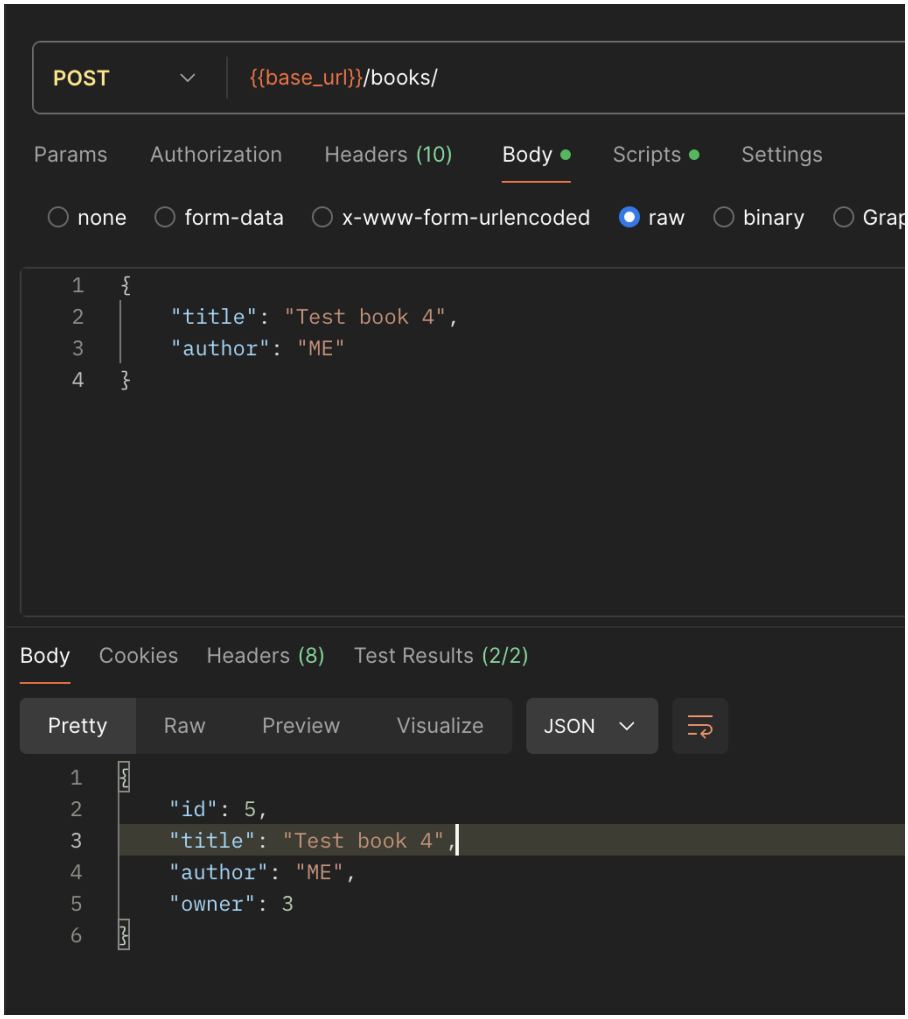
```ts
import { Entity, PrimaryGeneratedColumn, Column, ManyToOne } from "typeorm";
import { Books } from "./Books.js";

@Entity()
export class ExchangeRequests {
  @PrimaryGeneratedColumn()
  id: number;

  @Column({ type: "int" })
  creator: number;

  @ManyToOne(() => Books)
  requestedBook: Books;

  @ManyToOne(() => Books)
  offeredBook: Books;

  @Column({ type: "varchar", default: "pending" })
  status: string;
}
```

3) В сервере перепишем middleware
Теперь он будет ходить в auth service за валидацией токена

```ts
import { Request, Response, NextFunction } from "express";
import axios from "axios";

const authMiddleware = async (req: Request, res: Response, next: NextFunction) => {
  const authHeader = req.headers.authorization;

  if (!authHeader || !authHeader.startsWith("Bearer")) {
    return res.status(401).json({ message: "No token provided" });
  }

  const token = authHeader.split(" ")[1];
  console.log(token);
  try {
    const response = await axios.post(`http://localhost:3000/api/v1/auth/validate`, { token });
    console.log(response.data);
    const user = response.data.user;

    // @ts-ignore
    req.user = user;

    next();
  } catch (error) {
    console.log(error);
    return res.status(401).json({ message: "Invalid token" });
  }
};

export default authMiddleware;
```

4) Тестируем что все работает





Вывод

В ходе работы был реализован отдельный микросервис, занимающийся авторизацией и управлением пользователями