

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа №2

Выполнил:

Егоров Мичил

Группа

К33401

Проверил:

Добряков Д. И.

Санкт-Петербург

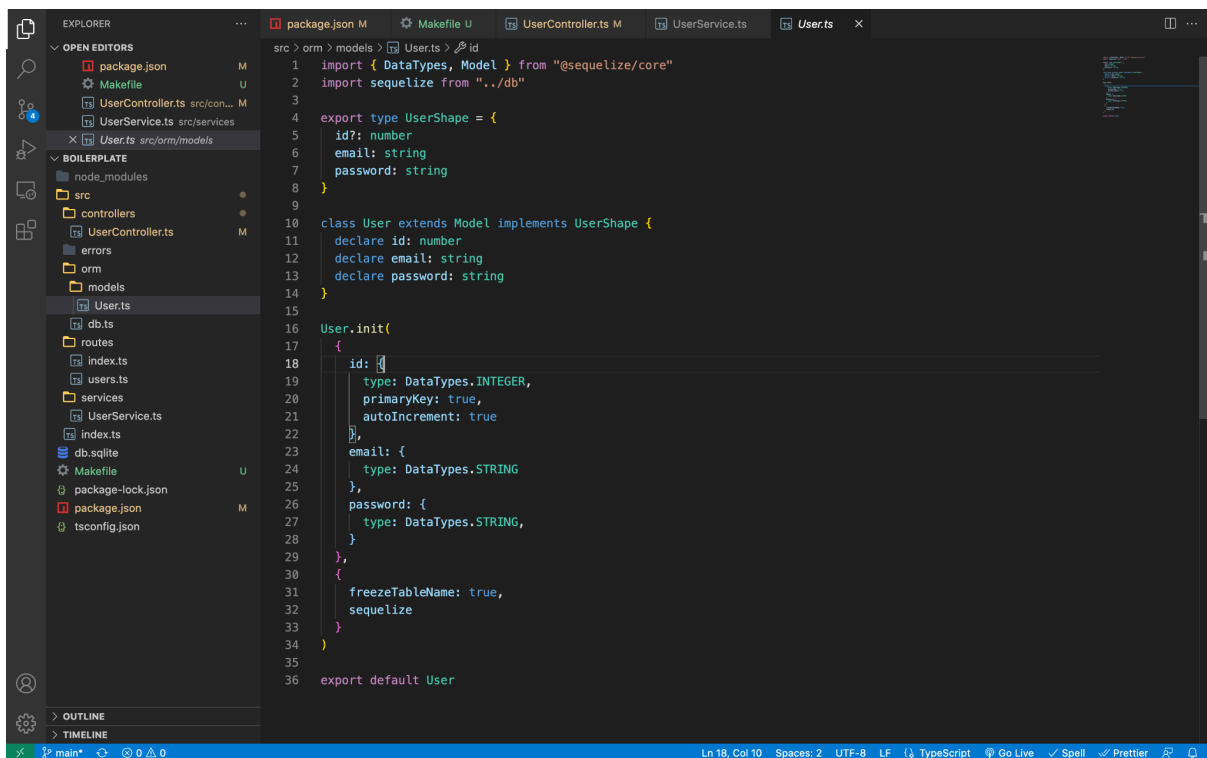
2022 г.

Задача

- Продумать свою собственную модель пользователя
- Реализовать набор из CRUD-методов для работы с пользователями средствами Express + Sequelize
- Написать запрос для получения пользователя по id

Ход работы

1. Модель пользователя



The screenshot shows a VS Code editor with the following components:

- EXPLORER (Left Panel):** Displays the project structure. The 'models' directory is expanded, showing 'User.ts' selected.
- EDITOR (Main Panel):** Shows the code for 'User.ts'. The code defines a 'User' model that extends 'Sequelize.Model' and implements the 'UserShape' interface. It includes fields for 'id', 'email', and 'password'.
- Code Content:**

```
1 import { DataTypes, Model } from "@sequelize/core"
2 import sequelize from "../db"
3
4 export type UserShape = {
5   id?: number
6   email: string
7   password: string
8 }
9
10 class User extends Model implements UserShape {
11   declare id: number
12   declare email: string
13   declare password: string
14 }
15
16 User.init(
17   {
18     id: {
19       type: DataTypes.INTEGER,
20       primaryKey: true,
21       autoIncrement: true
22     },
23     email: {
24       type: DataTypes.STRING
25     },
26     password: {
27       type: DataTypes.STRING,
28     },
29   },
30   {
31     freezeTableName: true,
32     sequelize
33   }
34 )
35
36 export default User
```
- STATUS BAR (Bottom):** Shows the current file is 'main.ts' at line 18, column 10, using UTF-8 encoding and LF line endings. It also indicates the file is a TypeScript file.

2. CRUD-методы

The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project structure with folders like 'src', 'controllers', 'orm', 'models', 'db.ts', 'routes', 'index.ts', 'users.ts', 'services', 'User.ts', 'db.sqlite', 'Makefile', 'package-lock.json', 'package.json', and 'tsconfig.json'. The file 'UserController.ts' is selected in the file explorer. The editor shows the following code:

```
src > controllers > UserController.ts > UserController > get > user
1 import UserService from '../services/UserService'
2
3
4 class UserController {
5   private userService: UserService
6
7   constructor() {
8     this.userService = new UserService()
9   }
10
11   get = async (request: any, response: any) => {
12     try {
13       const user = await this.userService.getById(
14         Number(request.params.id)
15       )
16       response.send(user)
17     } catch (error: any) {
18       response.status(404).send({
19         "error": error.message
20       })
21     }
22   }
23 }
24
25 export default UserController
26
```

3. Получение пользователя по id

The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project structure with folders like 'src', 'controllers', 'orm', 'models', 'db.ts', 'routes', 'index.ts', 'users.ts', 'services', 'User.ts', 'db.sqlite', 'Makefile', 'package-lock.json', 'package.json', and 'tsconfig.json'. The file 'UserService.ts' is selected in the file explorer. The editor shows the following code:

```
src > services > UserService.ts > UserService > create
1 import User from "../orm/models/User"
2
3
4 class UserService {
5   public async getById(id: number): Promise<User> {
6     const user = await User.findPk(id);
7
8     if(!user) {
9       throw new Error(`User (id=${id}) does not exist`);
10    }
11
12    return user;
13  }
14
15  public async create(userData: {email: string, password: string}): Promise<User> {
16    const user = await User.create(userData)
17    return user
18  }
19 }
20
21 export default UserService
22
```

Вывод

Во время работы научились создавать модели с помощью Sequelize, написали CRUD-методы с помощью обращения в базу данных.