

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа №2

Выполнил:

Викторова Анастасия

M3220d

Проверил:

Добряков Д. И.

Санкт-Петербург

2024 г.

Задача

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

Ход работы

Создание проекта

- 1) инициализация проекта
`npm init`
- 2) установка пакетов express, sequelize, sqlite3, sequelize CLI
`npm i express -S`
`npm i sequelize -S`
`npm i sqlite3-S`
`npm install --save-dev sequelize-cli`
- 3) создание структуры
`npx sequelize init`
- 4) замена значения поля "dialect" - "mysql" на "sqlite" в файле config.json

```
config > {} config.json > {} development > dialect
1  {
2    "development": {
3      "username": "root",
4      "password": null,
5      "database": "database_development",
6      "host": "127.0.0.1",
7      "dialect": "sqlite"
```

- 5) создание модели User
`npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email:string,password:string`

В итоге после редактирования модель имеет данный вид

```
id: {
  allowNull: false,
  autoIncrement: true,
  primaryKey: true,
  type: Sequelize.INTEGER
},
firstName: {
  allowNull: false,
```

```

    type: Sequelize.STRING
  },
  lastName: {
    type: Sequelize.STRING
  },
  email: {
    allowNull: false,
    type: Sequelize.STRING
  },
  password: {
    allowNull: false,
    type: Sequelize.STRING
  },
  createdAt: {
    allowNull: false,
    type: Sequelize.DATE
  },
  updatedAt: {
    allowNull: false,
    type: Sequelize.DATE
  }
}

```

6) создание миграции

`npx sequelize db:migrate`

7) реализация CRUD-методов

```

7 //create
8 app.post("/users", async (req, res) => {
9   await db.User.create(req.body)
10  return res.send({msg: "user has been created"})
11 })
12
13
14 //read
15 app.get('/users', async (req, res) => {
16   const users = await db.User.findAll()
17
18   if (users) {
19     return res.send(users)
20   }
21
22   return res.send({msg: "users are not found"})
23 })
24

```

Рисунок 1. Реализация CRUD-методов

```

47 //update
48 app.put('/users/:id', async (req, res) => {
49   const num = await db.User.update(req.body, { where: { id: req.params.id } })
50
51   if (num == 1) {
52     return res.send({msg: 'user has been updated'})
53   }
54
55   return res.send({msg: 'user is not found'})
56 })
57
58 //delete
59 app.delete('/users/:id', async (req, res) => {
60   const user = await db.User.findByIdPk(req.params.id)
61
62   if (user) {
63     await user.destroy()
64   }
65   return res.send({msg: "user is not found"})
66 });

```

Рисунок 2. Реализация CRUD-методов

8) запрос для получения пользователя по id/email

```

25 //read id
26 app.get('/users/:id', async (req, res) => {
27   const user = await db.User.findByIdPk(req.params.id)
28
29   if (user) {
30     return res.send(user.toJSON())
31   }
32
33   return res.send({msg: "user is not found"})
34 })

```

Рисунок 3. По id

```

36 //read email
37 app.get('/users/:email', async (req, res) => {
38   const user = await db.User.findByIdPk(req.params.email)
39
40   if (user) {
41     return res.send(user.toJSON())
42   }
43
44   return res.send({msg: "user is not found"})
45 })

```

Рисунок 4. По email

The screenshot shows a REST client interface. At the top, a GET request is configured for the URL `http://localhost:3000/users/2`. Below the URL bar, tabs for Params, Auth, Headers (8), Body, Pre-req., Tests, and Settings are visible. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response status is 200 OK, with a response time of 60 ms and a body size of 412 B. The JSON data represents a user object with the following fields:

Key	Value	Description
		...

```
{
  "id": 2,
  "firstName": "Test5",
  "lastName": "Test5",
  "email": "test5@example.com",
  "password": "password5",
  "createdAt": "2024-03-13T17:51:32.529Z",
  "updatedAt": "2024-03-13T17:51:32.529Z"
}
```

Рисунок 5. Результат работы поиска по id

Вывод

В ходе работы я получила навыки по работе с express и OPM sequelize.