

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа № 2: Знакомство с ORM Sequelize

Выполнил:

Вали Насибулла

Группа К33402

Проверил:

Добряков Д. И.

Санкт-Петербург

2023 г.

Задание:

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

Ход работы

<https://github.com/kantegory/ITMO-ICT-Backend-2023/pull/18>

POST

POST http://localhost:4000/users

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "Checking",
3   ... "password": "12345",
4   ... "email": "check@gmail.com"
5 }
```

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 65 ms Size: 244 B Save Response

Pretty Raw Preview Visualize HTML

```
1 User is inserted
```

GET

GET http://localhost:4000/users Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1

Body Cookies Headers (7) Test Results Status: 200 OK Time: 23 ms Size: 542 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "username": "Checking",
4   "password": "12345",
5   "email": "check@gmail.com"
6 },
7 {
8   "id": 2,
9   "username": "Checking",
10  "password": "12345",
11  "email": "check@gmail.com"
12 },
13 {
14   "id": 3,
15   "username": "Checking",
16   "password": "12345",
17   "email": "check@gmail.com"
18 },
19 }
```

GET by ID

http://localhost:4000/users/3 Save

GET http://localhost:4000/users/3 Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1

Body Cookies Headers (7) Test Results Status: 200 OK Time: 27 ms Size: 310 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 3,
3   "username": "Checking",
4   "password": "12345",
5   "email": "check@gmail.com"
6 }
```

UPDATE

http://localhost:4000/users/2 Save

PUT http://localhost:4000/users/2 Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1

2 {

3 "username": "updateded user"

4 }

Body Cookies Headers (7) Test Results Status: 200 OK Time: 30 ms Size: 233 B Save Response

Pretty Raw Preview Visualize HTML

1 Updated

http://localhost:4000/users/2

GET http://localhost:4000/users/2

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

1

Body Cookies Headers (7) Test Results Status: 200 OK Time: 16 ms Size: 316 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "username": "updated user",
4   "password": "12345",
5   "email": "check@gmail.com"
6 }
```

DELETE

http://localhost:4000/users/2

DELETE http://localhost:4000/users/2

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 25 ms Size: 233 B Save Response

Pretty Raw Preview Visualize **HTML**

```
1 deleted
```

http://localhost:4000/users

GET http://localhost:4000/users

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

1

Body Cookies Headers (7) Test Results Status: 200 OK Time: 14 ms Size: 465 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "username": "Checking",
5     "password": "12345",
6     "email": "check@gmail.com"
7   },
8   {
9     "id": 3,
10    "username": "Checking",
11    "password": "12345",
12    "email": "check@gmail.com"
13  },
14  {
15    "id": 4,
16    "username": "Checking",
17    "password": "12345",
18    "email": "check@gmail.com"
19  }
20 ]
```

The screenshot shows a web application interface for a SQLite database. The top navigation bar includes tabs for 'index.js', 'dev.sqlite', 'User.js', and 'database.js'. The main content area displays the 'dev.sqlite' database with a sidebar on the left showing 'Tables (2)' including 'sqlite_sequence' and 'users'. The 'users' table is selected, showing 3 records. The table has columns: id, username, password, and email. The records are as follows:

id	username	password	email
1	Checking	12345	check@gmail.com
2	Checking	12345	check@gmail.com
3	Checking	12345	check@gmail.com

Вывод:

с помощью этого домашнего задания мы научились создавать и выполнять операции CRUD с базой данных sqlite3.