

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«Национальный исследовательский университет ИТМО»  
(Университет ИТМО)**

**Факультет Инфокоммуникационных технологий (ИКТ)**

**Образовательная программа Мобильные и сетевые технологии**

**О Т Ч Е Т**

по дисциплине: Бек-энд разработка  
по Домашней работе 2

Тема задания: «Знакомство с ORM Sequelize»

Обучающийся: Кириллова В.Е., К33402

Преподаватель: Добряков Д.И.

Оценка: \_\_\_\_

Дата: «\_\_» \_\_\_\_\_ 2023 г.

Санкт-Петербург  
2023

## ЗАДАЧИ

- 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

Модель пользователя в файле user.js

```
1 //lets says if we have user table in our db, so we are creating user mo
2
3
4 const {Model, DataTypes} = require('sequelize');
5 const sequelize = require('./database');
6
7 class User extends Model{}
8 User.init({
9   username: {
10     type:DataTypes.STRING
11   },
12   password:{
13     type:DataTypes.STRING
14   },
15   email:{
16     type:DataTypes.STRING
17   }
18 },{
19   sequelize,
20   modelName: 'user', //table name
21   timestamps: false // for not showing timing in our table
22 })
23
24 module.exports = User;
25
```

CRUD – методы для получения, изменения информации о пользователе и удаления пользователей

```

JS index.js > ...
1  const express = require("express");
2  const port = 4000;
3
4  const sequelize = require('./database');
5  const User = require('./User');
6  sequelize.sync({force: true}).then(()=> console.log("db is ready"))
7  //force: true. is used here to delete old data from table and create table again e
8  const app = express()
9  app.use(express.json()); // this to make express aware what data is coming. unless
10  app.post('/users', async (req,res)=>{
11      await User.create(req.body)
12      res.send("User is inserted")
13  });
14  //Show All Users
15  app.get('/users', async (req, res)=>{
16      const myuser = await User.findAll()
17      res.send(myuser);
18  })
19  //Find By Id
20  app.get('/users/:id', async (req,res)=>{
21      const requestedId = req.params.id;
22      const requestedUser = await User.findOne({where: {id: requestedId} })
23      res.send(requestedUser)
24  })
25
26  //Update By Id
27  app.put('/users/:id', async (req,res)=>{
28      const requestedId = req.params.id;
29      const requestedUser = await User.findOne({where: {id: requestedId} })
30      requestedUser.username = req.body.username;
31      await requestedUser.save()
32      res.send("Updated")
33  })
34
35  //Delete By Id
36  app.delete('/users/:id', async (req,res)=>{
37      const requestedId = req.params.id;
38      const requestedUser = await User.destroy({where: {id: requestedId} })
39      res.send("deleted")
40  })
41
42  app.listen(port, ()=>{
43      console.log(`Server is running on Port ${port}`)
44  })

```

Тестирование CRUD -методов с помощью POSTMAN

**POST** запрос на создание пользователя

HTTP Hw2 / Register

POST http://localhost:4000/users

Send

Params Auth Headers (8) **Body** Pre-req. Tests Settings

raw JSON Beautify

```
1 [
2   {
3     "id": 3,
4     "username": "Test3",
5     "password": "123",
6     "email": "Test3gmail.com"
7   }
8 ]
```

Body Cookies Headers (7) Test Results 200 OK 157 ms 244 B Save as Example

Pretty Raw Preview Visualize HTML

1 User is inserted

## GET запрос на получение всех пользователей

HTTP Hw2 / getAllUsers

GET http://localhost:4000/users

Send

Params Auth Headers (6) Body Pre-req. Tests Settings

Body Cookies Headers (7) Test Results 200 OK 6 ms 377 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "username": "Test1",
5     "password": "123",
6     "email": "Test1gmail.com"
7   },
8   {
9     "id": 3,
10    "username": "Test3",
11    "password": "123",
12    "email": "Test3gmail.com"
13  }
14 ]
```

GET запрос на получение информации о конкретном пользователе по id

HTTP Hw2 / getUserById

GET http://localhost:4000/users/1

Send

Params Auth Headers (6) Body Pre-req. Tests Settings Cookies

Body Cookies Headers (7) Test Results 200 OK 17 ms 304 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "username": "Test1",
4   "password": "123",
5   "email": "Test1gmail.com"
6 }
```

**PUT** запрос на изменение информации о конкретном пользователе по id

HTTP Hw2 / Update

PUT http://localhost:4000/users/3

Send

Params Auth Headers (8) Body Pre-req. Tests Settings Cookies Beautify

raw JSON

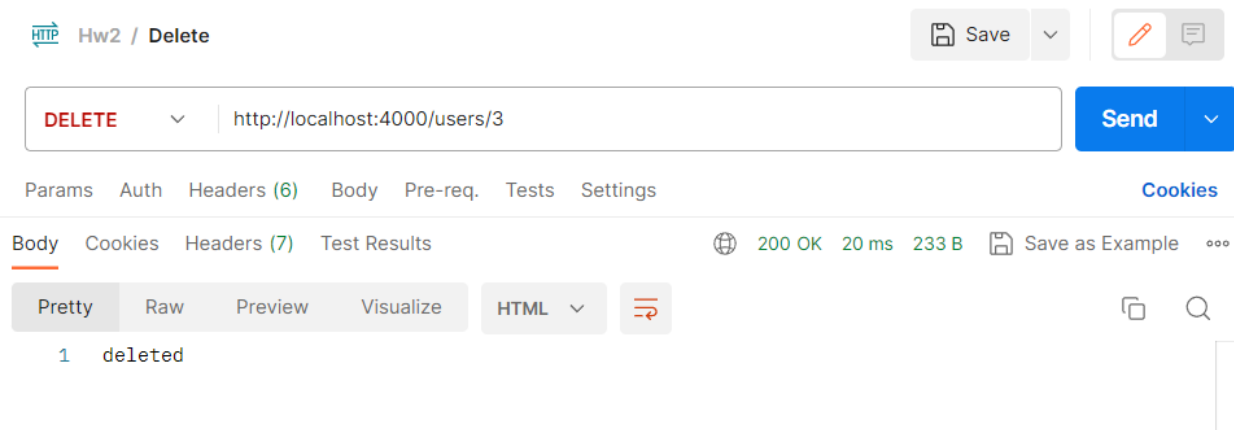
```
1 {
2   "id": 3,
3   "username": "Ilyxxa",
4   "password": "123",
5   "email": "Test3gmail.com"
6 }
```

Body Cookies Headers (7) Test Results 200 OK 73 ms 233 B Save as Example

Pretty Raw Preview Visualize HTML

```
1 Updated
```

**DELETE** запрос на удаление конкретного пользователя по id



## ВЫВОД

По результатам данной домашней работы я научилась создавать модели с помощью sequelize, а также научилась создавать и тестировать CRUD – методы с помощью express. Sequelize и POSTMAN