

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО"**

**ОТЧЕТ**

**ДОМАШНЯЯ РАБОТА №2**

**ЗНАКОМСТВО С ORM SEQUALIZE**

Работу выполнил:  
Костылев Иван,  
студент группы К33401

Работу проверил:  
Добряков Давид Ильич

Санкт-Петербург  
2022 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

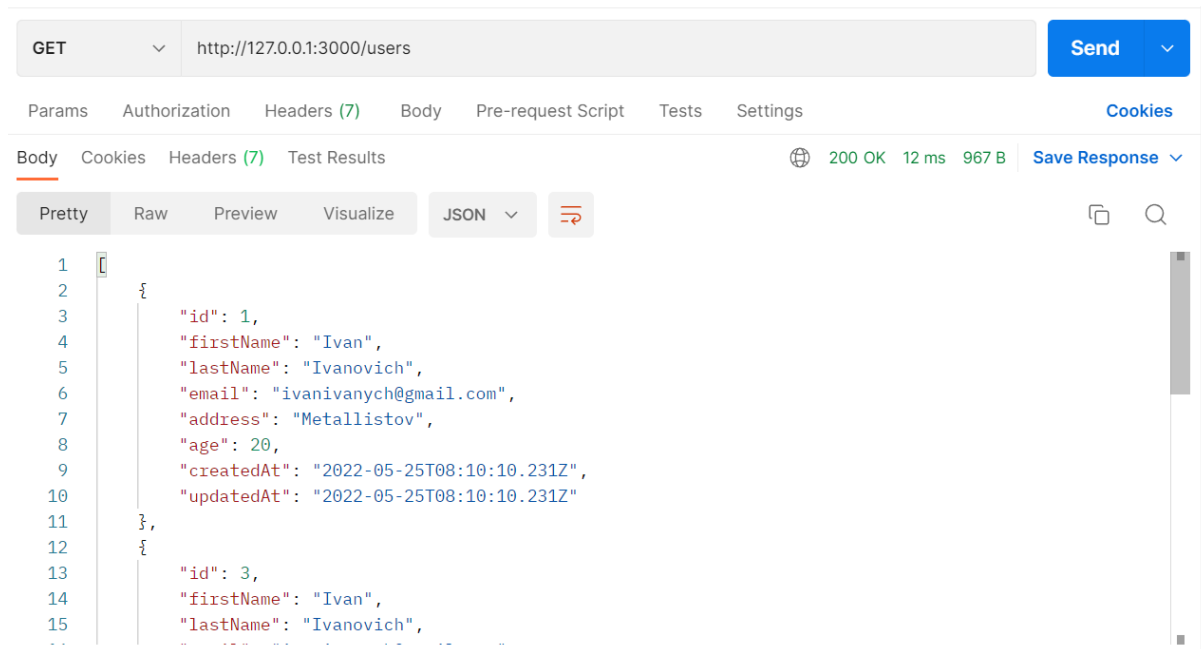
1. Продумать свою собственную модель пользователя. Данная модель будет содержать 4 поля:
  - a. Имя
  - b. Фамилия
  - c. Электронная почта
  - d. Домашний адрес
  - e. Возраст

```
module.exports = (sequelize, DataTypes) => {  
  class User extends Model {  
    /**  
     * Helper method for defining associations.  
     * This method is not a part of Sequelize lifecycle.  
     * The `models/index` file will call this method automatically.  
     */  
    static associate(models) {  
      // define association here  
    }  
  }  
  User.init({  
    firstName: DataTypes.STRING,  
    lastName: DataTypes.STRING,  
    email: DataTypes.STRING,  
    address: DataTypes.STRING,  
    age: DataTypes.INTEGER  
  }, {  
    sequelize,  
    modelName: 'User',  
  });  
  return User;  
};
```

2. Реализовать набор из CRUD-методов для работы с пользователем

Для работы с базой данных нами были реализованы следующие методы:

a. GET /users – список всех пользователей



GET `http://127.0.0.1:3000/users` Send

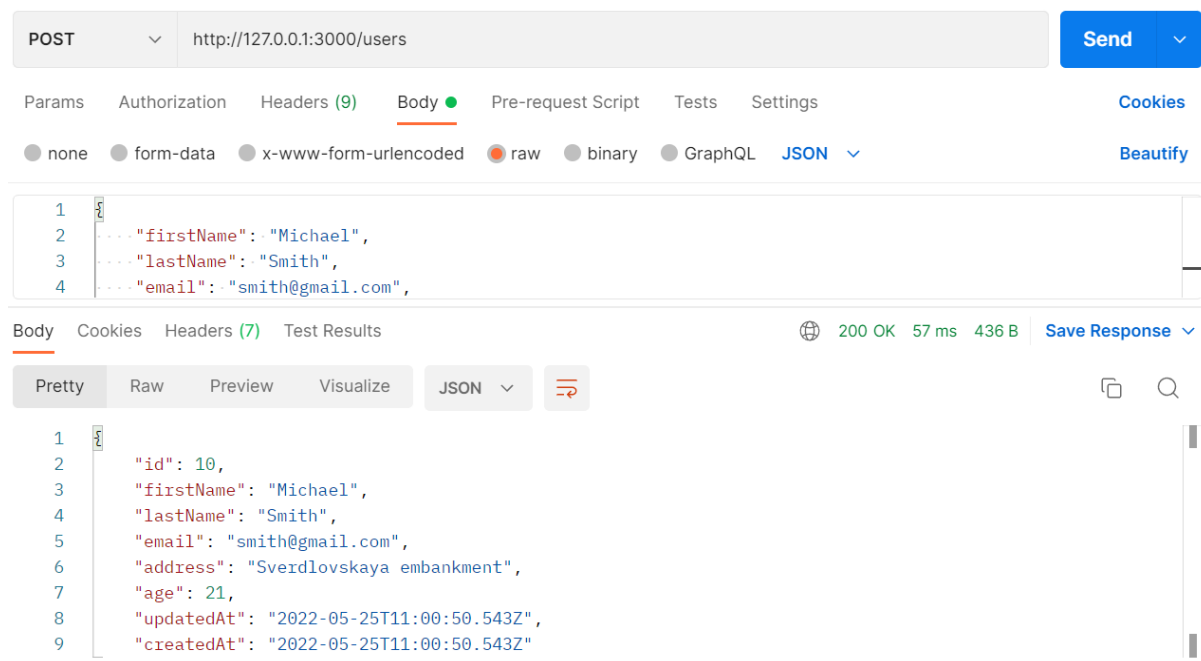
Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (7) Test Results 200 OK 12 ms 967 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "id": 1,
4     "firstName": "Ivan",
5     "lastName": "Ivanovich",
6     "email": "ivanivanych@gmail.com",
7     "address": "Metallistov",
8     "age": 20,
9     "createdAt": "2022-05-25T08:10:10.231Z",
10    "updatedAt": "2022-05-25T08:10:10.231Z"
11  },
12  {
13    "id": 3,
14    "firstName": "Ivan",
15    "lastName": "Ivanovich",
```

b. POST /users – создание пользователя



POST `http://127.0.0.1:3000/users` Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

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

Body Cookies Headers (7) Test Results 200 OK 57 ms 436 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "firstName": "Michael",
3   "lastName": "Smith",
4   "email": "smith@gmail.com",
5   "id": 10,
6   "firstName": "Michael",
7   "lastName": "Smith",
8   "email": "smith@gmail.com",
9   "address": "Sverdlovskaya embankment",
10  "age": 21,
11  "updatedAt": "2022-05-25T11:00:50.543Z",
12  "createdAt": "2022-05-25T11:00:50.543Z"
```

c. GET /users/:id – пользователь по id

GET http://127.0.0.1:3000/users/1 Send

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
--	-----	-------	-------------	-----	-----------

Body Cookies Headers (7) Test Results 200 OK 10 ms 429 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "firstName": "Ivan",
4   "lastName": "Ivanovich",
5   "email": "ivanivanych@gmail.com",
6   "address": "Metallistov",
7   "age": 20,
8   "createdAt": "2022-05-25T08:10:10.231Z",
9   "updatedAt": "2022-05-25T08:10:10.231Z"
10 }
```

d. PUT /users/:id – редактирование пользователя по id

PUT http://127.0.0.1:3000/users/1 Send

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies

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

```
1 {
2   ... "firstName": "Denis",
3   ... "lastName": "Ivanovich",
4   ... "email": "denisivanych@gmail.com",
5   ... "address": "Metallistov",
6   ... "age": 20
7 }
```

Body Cookies Headers (7) Test Results 200 OK 32 ms 265 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "status_code": "user_updated"
3 }
```

e. GET /users/:age - поиск всех пользователей по возрасту

GET http://127.0.0.1:3000/users/age/20 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (7) Test Results 200 OK 12 ms 785 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "id": 1,
4     "firstName": "Denis",
5     "lastName": "Ivanovich",
6     "email": "denisivanych@gmail.com",
7     "address": "Metallistov",
8     "age": 20,
9     "createdAt": "2022-05-25T08:10:10.231Z",
10    "updatedAt": "2022-05-25T11:03:20.684Z"
11  },
12  {
13    "id": 3,
14    "firstName": "Ivan",
15    "lastName": "Ivanovich",
16    "email": "ivanivanych@gmail.com",
```

f. DELETE /users/:id - удаление пользователя по id

DELETE http://127.0.0.1:3000/users/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (7) Test Results 200 OK 19 ms 265 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "status_code": "user_deleted"
3 }
```

## **Вывод**

В результате выполнения данной домашней работы нами были освоены и использованы инструмент для работы с базой данных sequelize и микрофреймворк express. Основная сложность заключалась в том, чтобы правильно настроить конфиг, т.к. без этого node все время требовал установить mysql2. Построение моделей и описание запросов по роутам сложностей не вызвало.