САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа: Знакомство с ORM Sequelize

Выполнила:

Лорс Хава

K33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2023 г.

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

Ход работы

1. Модель пользователя будет состоять из полей: name, username, email, password. Для того чтобы ее создать модель применим sequelize-cli команду и создадим миграции:
    - npx sequelize-cli model:generate --name User --attributes name:string,username:string,email:string,password:string
    - npx sequelize-cli db:migrate

2. Реализуем набор из CRUD-методов для работы с пользователями:

- Метод get, реализующий вывод всех пользователь

```
app.get('/users', async (req, res) => {
    const users = await db.User.findAll()
    return res.send(users)
})
```

- Метод get, реализующий вывод пользователя по id

```
app.get('/users/:id', async (req, res) => {
  const user = await db.User.findByPk(req.params.id)
  if (user) {
    return res.send(user.toJSON())
  }
  return res.send({"msg": "user is not found"})
})
```

- Метод post, реализующий создание пользователя

```
app.post('/users/create', async (req, res) => {
    try {
        const user = await db.User.create(req.body);
        return res.send(user.toJSON());
    } catch (e) {
        return res.send({"msg": "failed to create user"});
    }
})
```

- Метод put, реализующий обновление данных по id пользователя

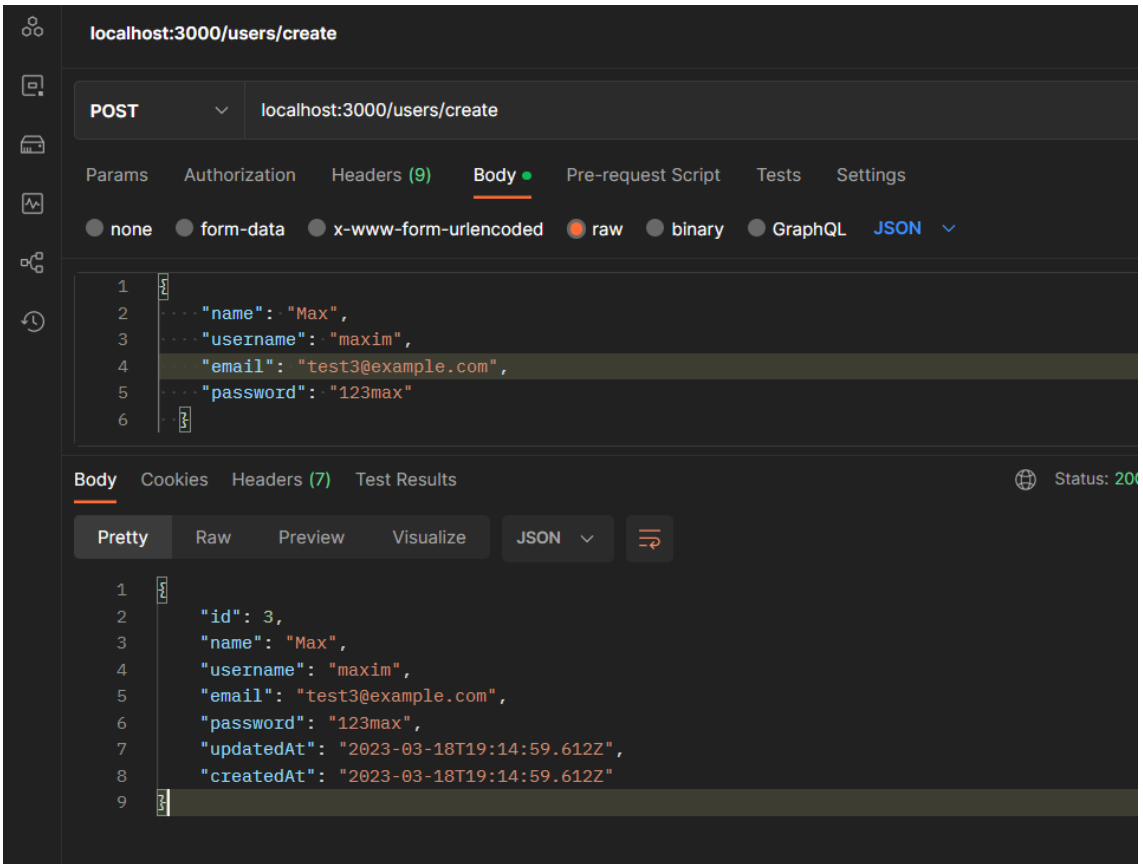
```
app.put('/users/:id', async (req, res) => {
    const user = await db.User.findByPk(req.params.id);
    if (user) {
        try {
            user.update(req.body, {where: {id: req.params.id}});
            return res.send({"msg": "successfully updated!"})
        } catch (e) {
            return res.send({"msg": e})
        }
    }
    return res.send({"msg": "User is not found"})
})
```

- Метод delete, реализующий удаление данных по id пользователя

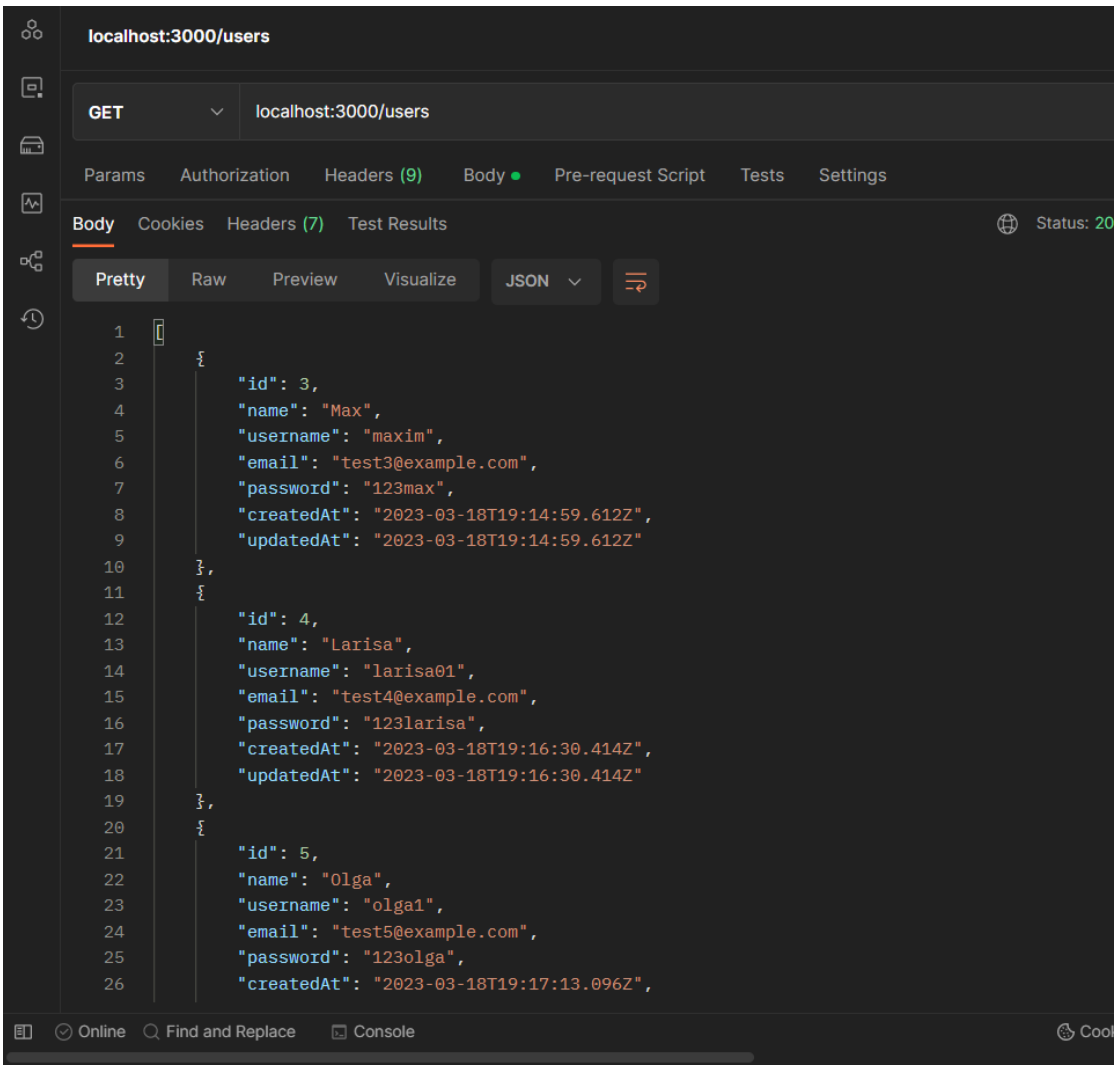
```
app.delete('/users/:id', async (req, res) => {
    const user = await db.User.destroy({where: {id: req.params.id}})
    if (user) {
        return res.send({"msg": "user deleted"})
    }
    return res.status(404).send({"msg": "user not found"})

})
```

3. Напишем запросы для получения, обновления и удаления пользователя по id.

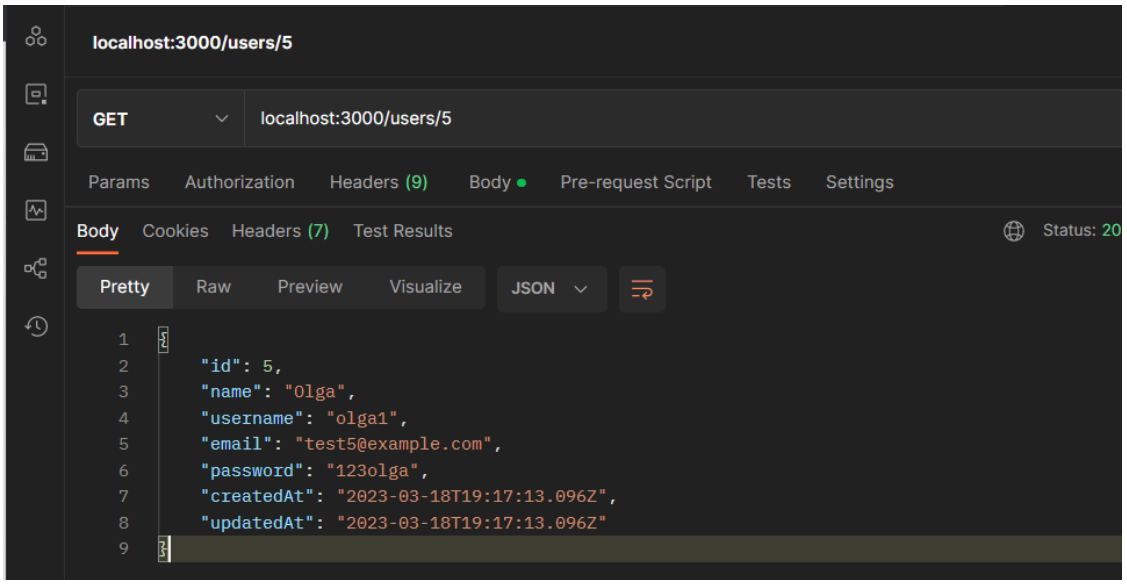
Запрос на создание пользователя:



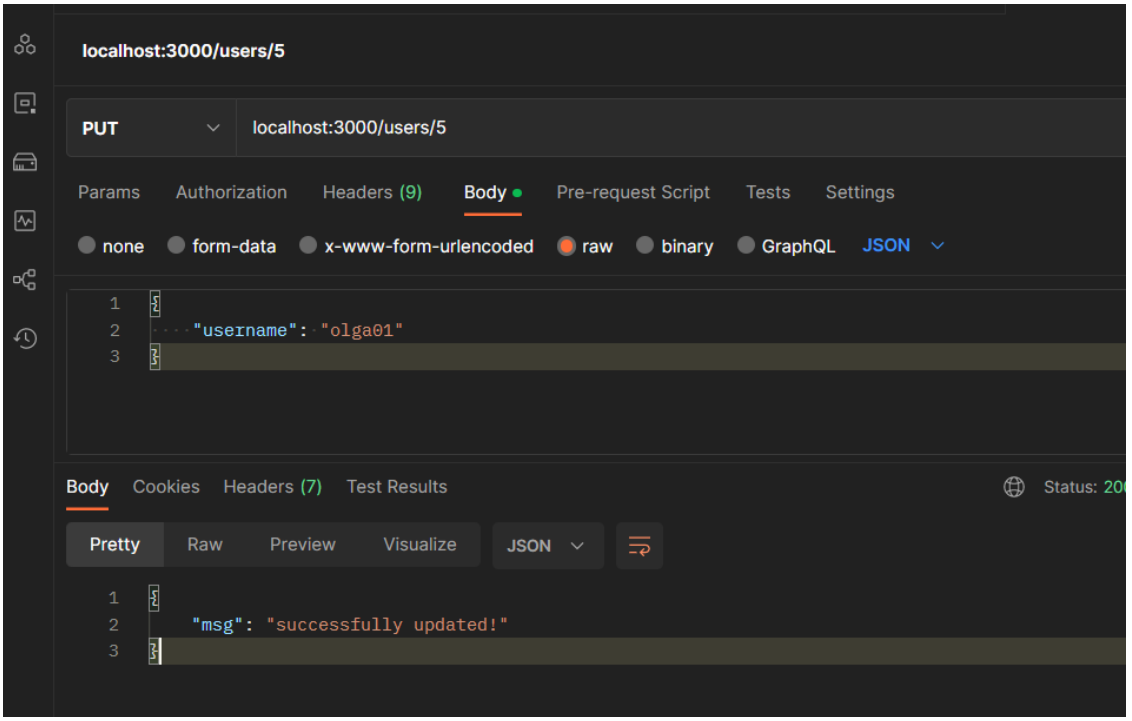
Запрос на получение всех пользователей:



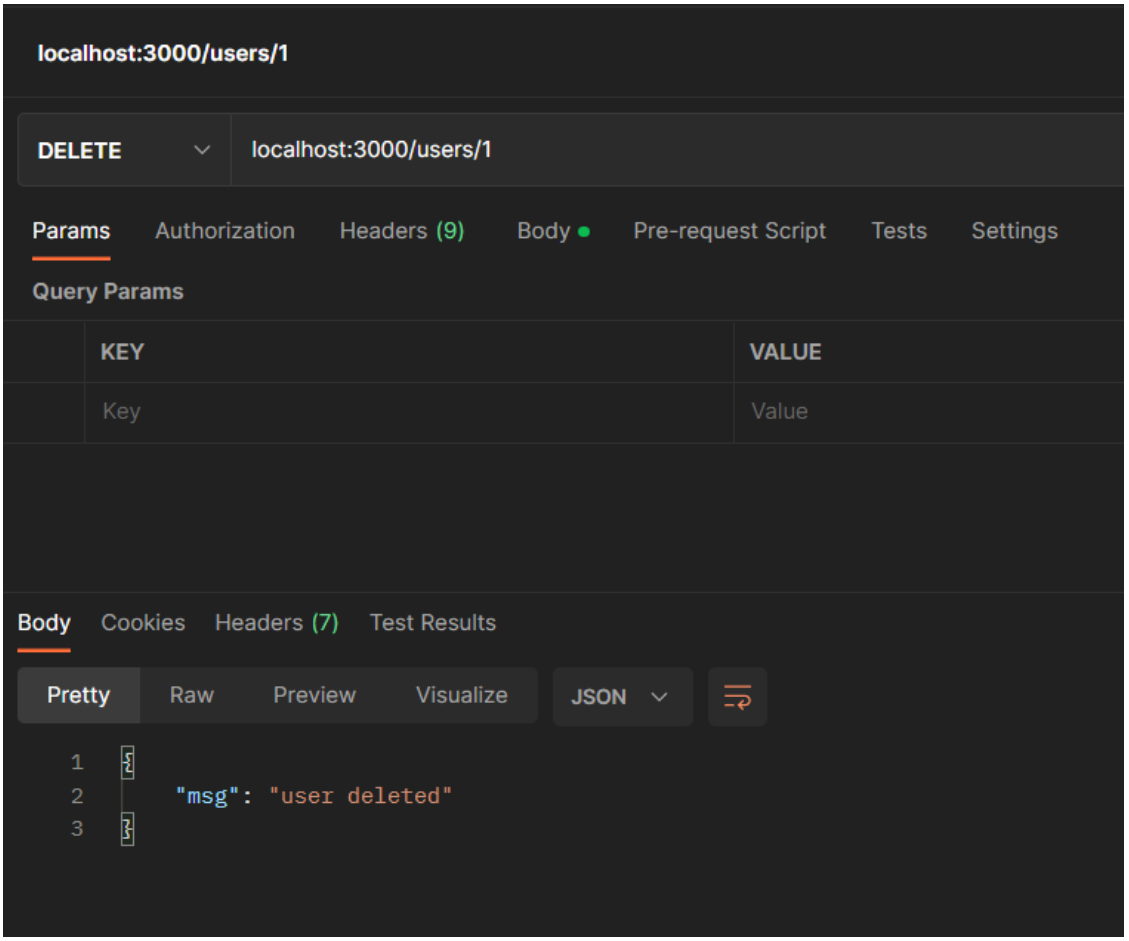
Запрос на получение пользователя по id:



Запрос на обновление данных пользователя по id:



Запрос на удаление пользователя по id:



Вывод

В ходе выполнения домашней работы были получены практические навыки реализации CRUD-методов с микрофреймворком Express и ORM Sequelize.