САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №2: Знакомство с ORM Sequelize

Выполнил:

Кондратьев Алексей

Группа К33412

Проверил:
Добряков Д. И.

Санкт-Петербург

2023 г.

Задача

- 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. **Создать модель пользователя в которых будут следующие поля:**
   - Имя
   - Фамилия
   - Почта
   - Пароль
   - Никнейм

```
'use strict';
const {
  Model
} = require('sequelize');
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,
    password: DataTypes.STRING,
    nickname: DataTypes.STRING
  }, {
    sequelize,
    modelName: 'User',
  });
  return User;
};
```

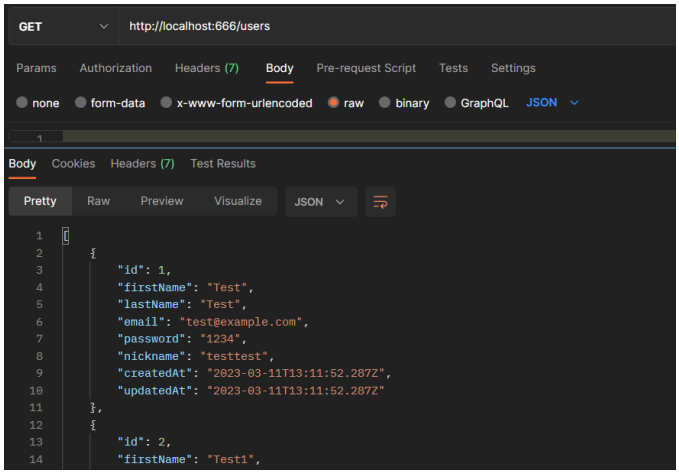
Представления:

```
app.get('/', async (req, res) => {
  res.send('Hello world by HW2')
})

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

app.get('/users/:id', async (req, res) => {
  const user = await db.User.findByPk(req.params.id)
  if (user) {
    return res.send(user.toJSON())
  } else {
    return res.status(404).send({'msg': 'user does not exist'})
  }
})
```

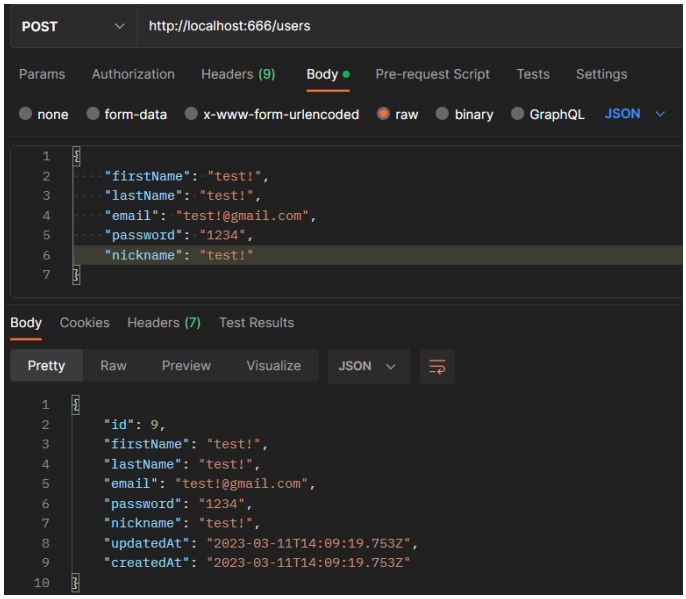
```
app.post('/users', async (req, res) => {
    const user = await db.User.create(req.body)
    return res.send(user.toJSON())
  })

app.put('/users/:id', async (req, res) => {
  console.log(req.body, req.params)
  const num = await db.User.update(req.body, { where: { id: req.params.id } })
  if (num) {
    return res.send({'msg': 'user has been updated'})
  } else {
    return res.status(404).send({'msg': 'user not found'})
  }
})

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'})
  } else {
    return res.status(404).send({'msg': 'user not found'})
  }
})
```

2. Реализовать набор из CRUD-методов для работы с пользователями средствами Express + Sequelize

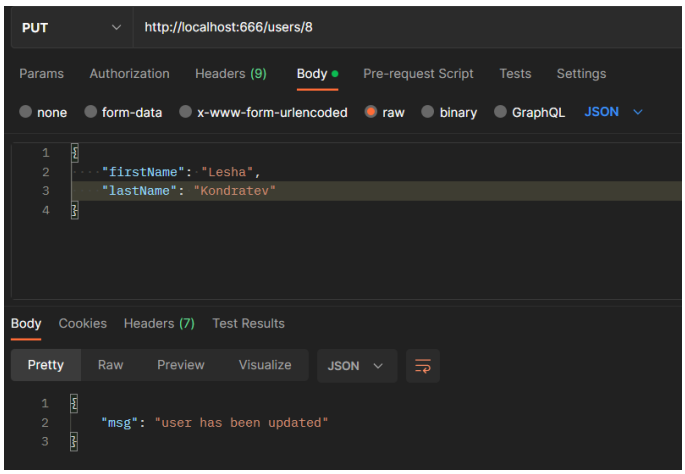
GET users:



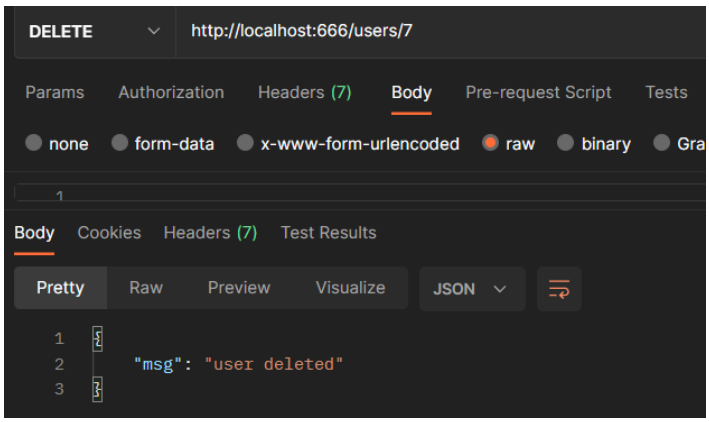
POST users:



Put users:

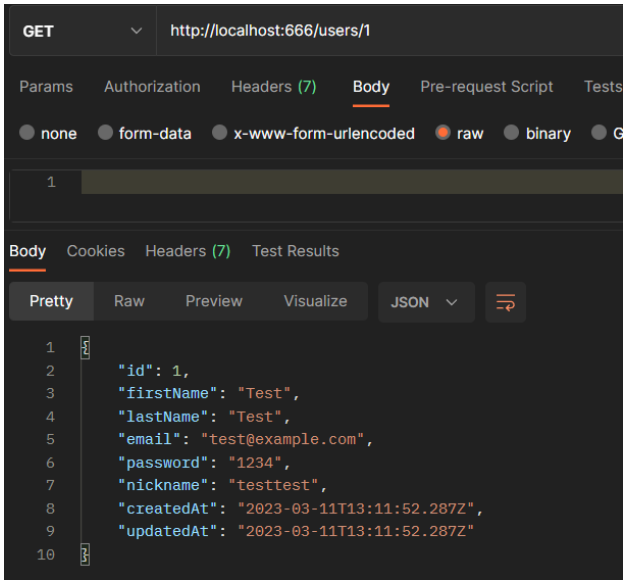


Delete users:



3. Написать запрос для получения пользователя по id/email

Get users/:id:



Вывод

Мы продумали свою собственную модель пользователя и реализовали CRUD методы для работы с моделью