САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №2

Выполнил:

Шляхов Денис Олегович

Группа К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2024 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. npm init
2. npm i express sequelize sqlite3 sequelize-cli body-parser
3. npx sequelize init
4. Инициализирую модель user через cli

```
PS C:\Users\Денис\PycharmProjects\ITMO-ICT-Backend-2024\homeworks\K33402\shliakhov_denis\hw_2> npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email:string,password:string

Sequelize CLI [Node: 20.11.1, CLI: 6.6.2, ORM: 6.37.1]

New model was created at C:\Users\Денис\PycharmProjects\ITMO-ICT-Backend-2024\homeworks\K33402\shliakhov_denis\hw_2\models\user.js .
New migration was created at C:\Users\Денис\PycharmProjects\ITMO-ICT-Backend-2024\homeworks\K33402\shliakhov_denis\hw_2\migrations\20240314194213-create-user.js .
```

5. npx sequelize db:migrate

Сгенерилась такая вот модель

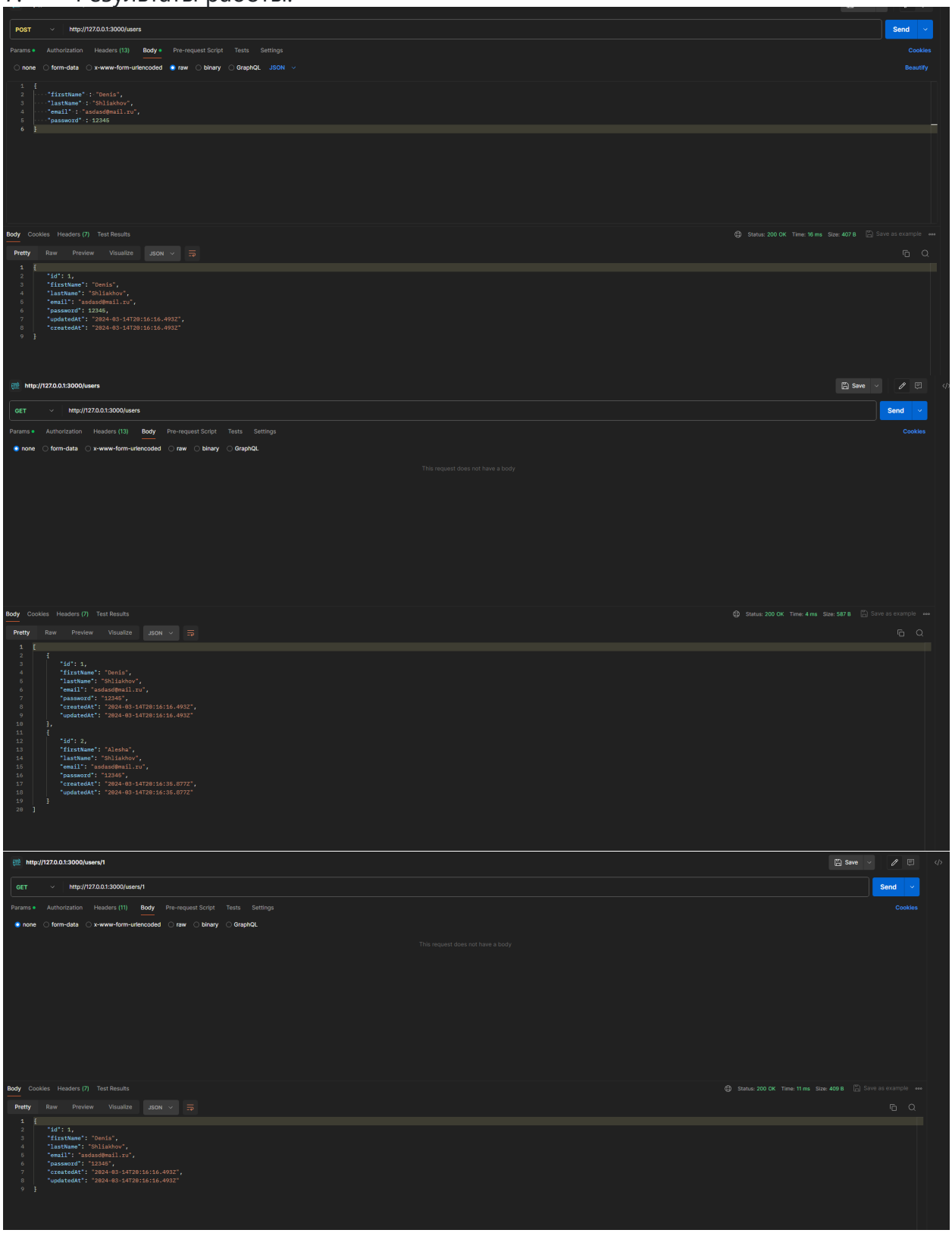
```
'use strict';
const {
  Model
} = require('sequelize');
module.exports = (sequelize, DataTypes) => {
  1+ usages  Puroblast
  class User extends Model {
    /**
     * Helper method for defining associations.
     * This method is not a part of Sequelize lifecycle.
     * The `models/index` file will call this method automatically.
     */
    1+ usages  Puroblast
    static associate(models) {
      // define association here
    }
  }
  User.init( attributes: {
    firstName: DataTypes.STRING,
    lastName: DataTypes.STRING,
    email: DataTypes.STRING,
    password: DataTypes.STRING
  }, options: {
    sequelize,
    modelName: 'User',
  });
  return User;
};
```

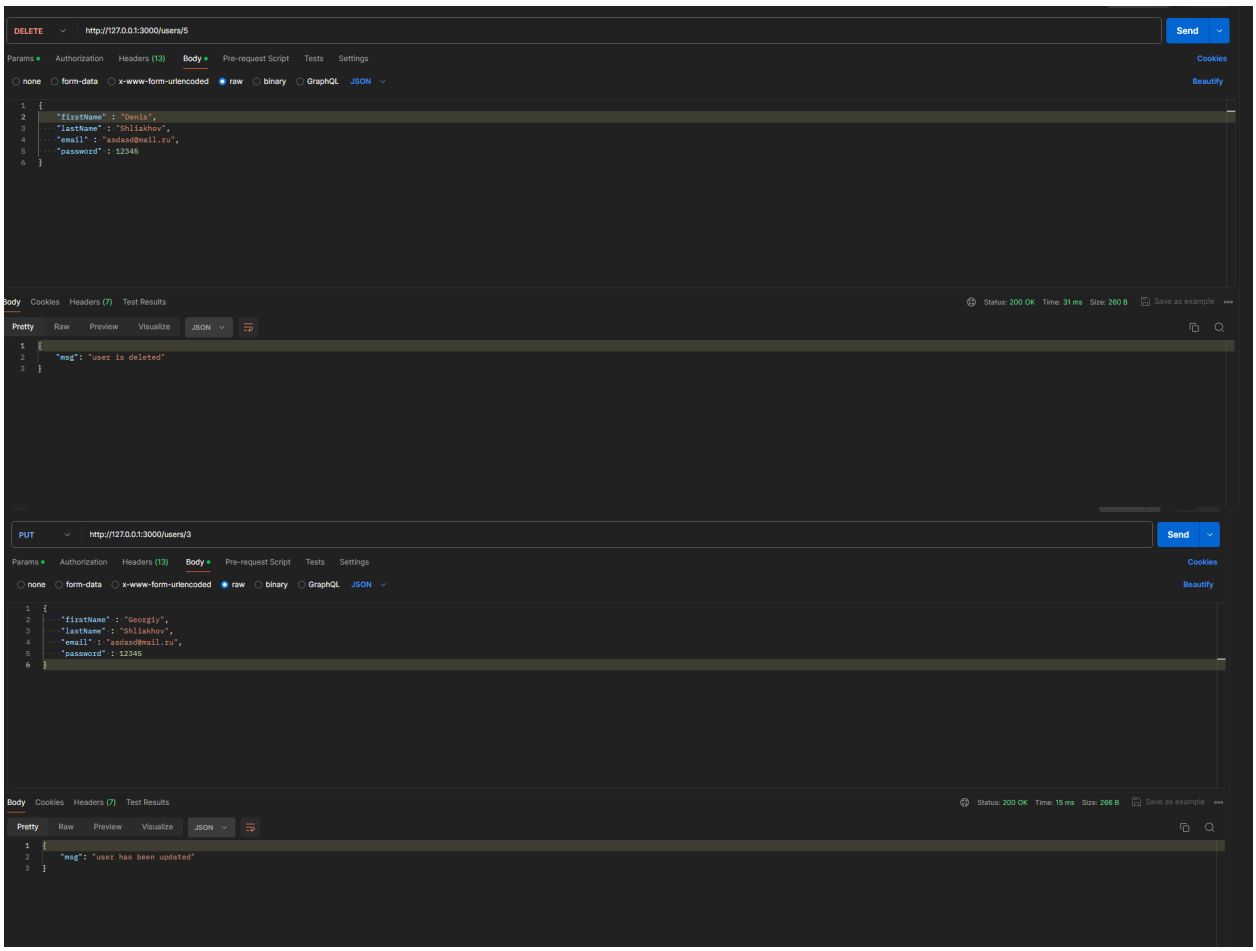
6. Реализация CRUD и GET запросов

```
Puroblast
app.post( path: "/users", handlers: async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    user = await db.User.create(req.body)
    return res.json(user)
})


//read
Puroblast
app.get('/users', async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    const users : Model[]  = await db.User.findAll()

    if (users) {
        return res.send(users)
    }

    return res.send( body: {"msg": "users are not found"})
})

//read id
Puroblast
app.get('/users/:id', async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    const user : Promise<...> | Promise<...>  = await db.User.findByPk(req.params.id)

    if (user) {
      return res.send(user.toJSON())
    }

    return res.send( body: {"msg": "user is not found"})
})
```

```
Puroblast
app.get('/users/:email', async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    const user : Promise<...> | Promise<...>  = await db.User.findByPk(req.params.email)

    if (user) {
      return res.send(user.toJSON())
    }

    return res.send( body: {"msg": "user is not found"})
})

//update
Puroblast
app.put( path: '/users/:id', handlers: async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    const num = await db.User.update(req.body, { where: { id: req.params.id } })

    if (num == 1) {
      return res.send( body: {'msg': 'user has been updated'})
    }

    return res.send( body: {'msg': 'user is not found'})
})

//delete
Puroblast
app.delete( path: '/users/:id', handlers: async (req : Request<P, ResBody, ReqBody, ReqQuery, LocalsObj> , res : Response<ResBody, LocalsObj> ) : Promise<any>  => {
    const user : Promise<...> | Promise<...>  = await db.User.findByPk(req.params.id)

    if (user) {
        await user.destroy()
    }
    return res.send( body: {"msg": "user is deleted"})
});
```

7. Результаты работы:





Вывод

В ходе данной работы были изучены основы работы с express и sequelize