САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа 2

Выполнила:

Афанасьева Ирина Максимовна

Группа:

К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2023 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. Создана модель пользователя, включающая в себя следующие поля:
    - Фамилия
    - Имя
    - Отчество
    - Email
    - Дата рождения

    Создание модели:

```
PS C:\Users\Irina\Desktop\Семинары\Бекендеки\HW2> npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,fath
erName:string,email:string,dateOfBirth:date

Sequelize CLI [Node: 18.15.0, CLI: 6.6.0, ORM: 6.31.1]

New model was created at C:\Users\Irina\Desktop\Семинары\Бекендеки\HW2\models\user.js .
New migration was created at C:\Users\Irina\Desktop\Семинары\Бекендеки\HW2\migrations\20230528175557-create-user.js .
```

    Модель пользователя:

```
User.init({
  firstName: DataTypes.STRING,
  lastName: DataTypes.STRING,
  fatherName: DataTypes.STRING,
  email: DataTypes.STRING,
  dateOfBirth: DataTypes.DATE
}, {
  sequelize,
  modelName: 'User',
});
return User;
```

2. Реализованы запросы CRUD методов с помощью фреймворка Express и Sequelize.
    - Get/users – список всех пользователей

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

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "firstName": "Irina",
        "lastName": "Afanasieva",
        "fatherName": "Maksimovna",
        "email": "multifructina@gmail.com",
        "dateOfBirth": "2005-12-15T00:00:00.000Z",
        "createdAt": "2023-05-28T20:37:59.763Z",
        "updatedAt": "2023-05-28T21:28:15.127Z"
    },
    {
        "id": 2,
        "firstName": "Ksenia",
        "lastName": "Afanasieva",
        "fatherName": "Maksimovna",
        "email": "nena@gmail.com",
        "dateOfBirth": "2001-04-10T00:00:00.000Z",
        "createdAt": "2023-05-28T20:53:35.565Z",
        "updatedAt": "2023-05-28T21:28:44.248Z"
    }
]
```


- Post/users – создание пользователя

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

POST http://127.0.0.1:3000/users/

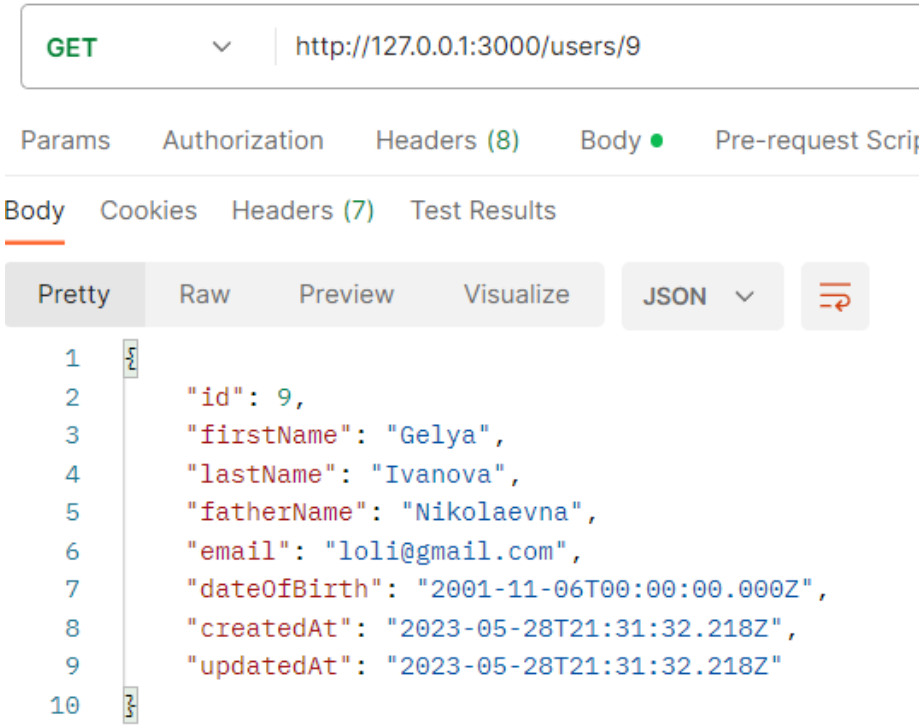
Params Authorization Headers (8) Body Pre-request Script Tests Settings

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


```
{
    "firstName": "Gelya",
    "lastName": "Ivanova",
    "fatherName": "Nikolaevna",
    "email": "loli@gmail.com",
    "dateOfBirth": "2001-11-06"
}
```

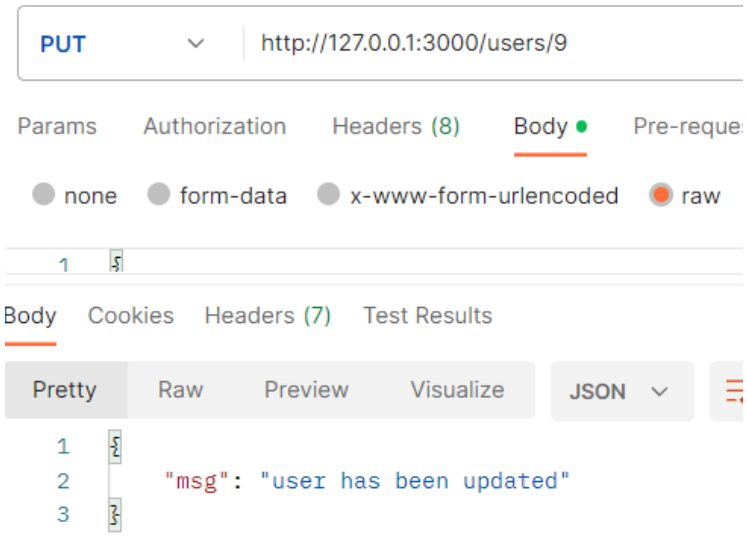
- Get/users/:id – пользователь по id

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



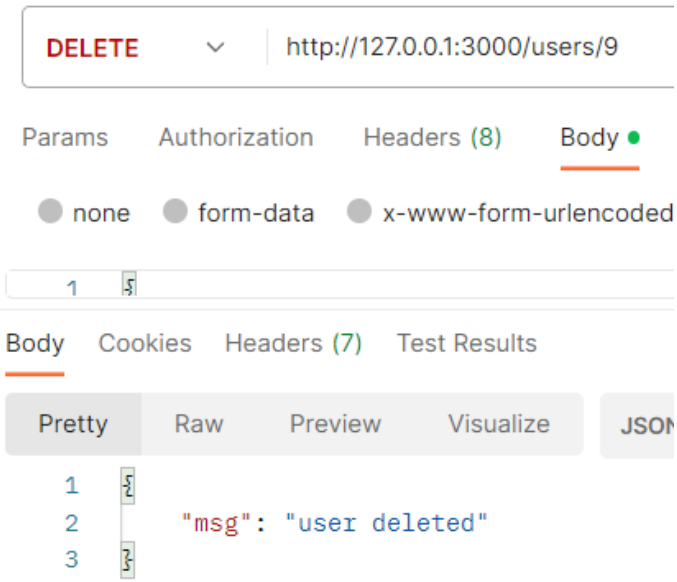
- Put/users/:id – редактирование пользователя по id

```
app.put('/users/:id', async (req, res) => {
  const num = await db.User.update(req.body, { where: { id: req.params.id } })
  if (num == 1) {
    res.send({'msg': 'user has been updated'})
  } else {
    res.status(404).send({'msg': 'user not found'})
  }
})
```



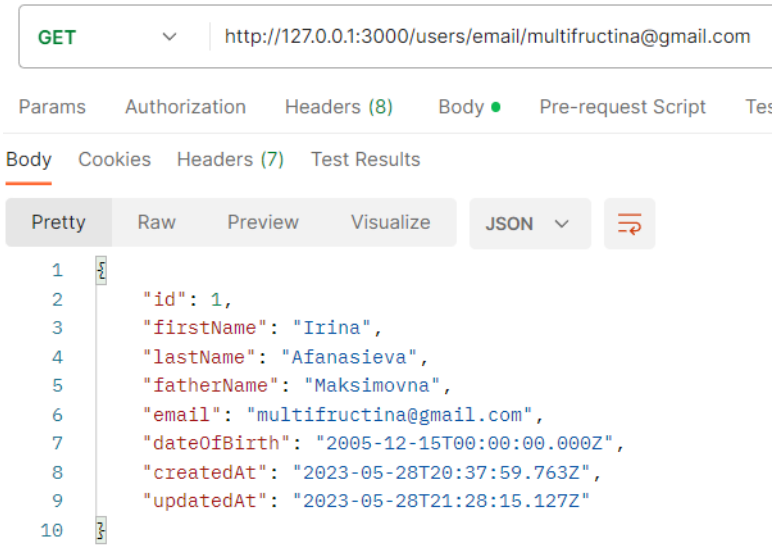
- Delete/users/:id – удаление пользователя по id

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



3. Реализован метод получения пользователя по email.

```
app.get('/users/email/:email', async (req, res) => {
  const user = await db.User.findOne({ where: { email: req.params.email } })
  if (user) {
    res.send(user.toJSON())
  } else {
    res.status(404).send({'msg': 'user not found'})
  }
})
```



Вывод

В ходе работы был освоен фреймворк Express и ORM Sequelize, основы работы с моделями, создание CRUD-методов и фильтров.