САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая/Лабораторная работа

Выполнил:

Поляков Андрей

Группа
К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2022 г.

Задача

1. Продумать свою собственную модель пользователя
2. Реализовать набор из CRUD-методов для работы с пользователями средствами Express + Sequelize
3. Написать запрос для получения пользователя по id/email

Ход работы

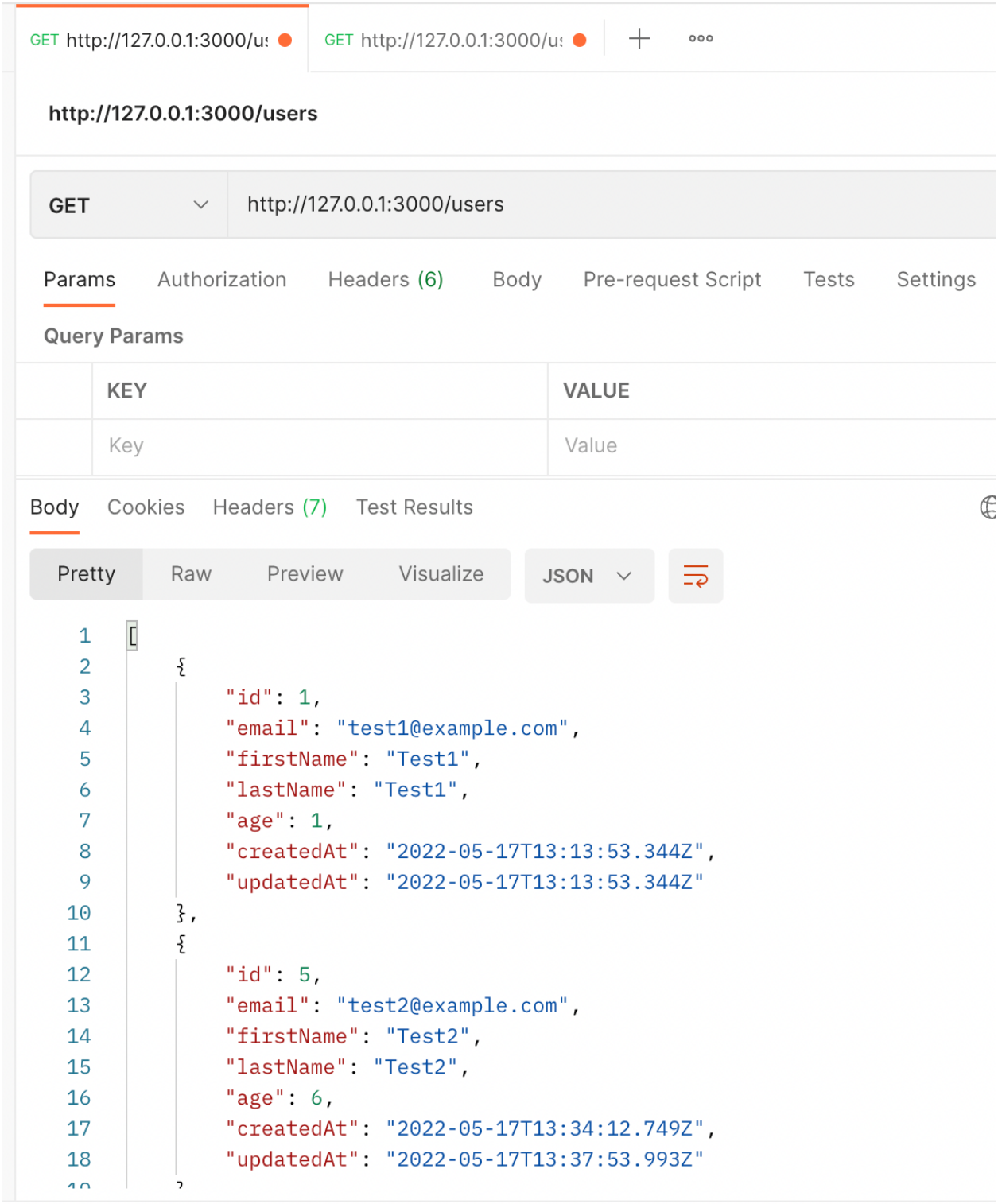
1) Создана модель пользователя, имеющая следующие поля:

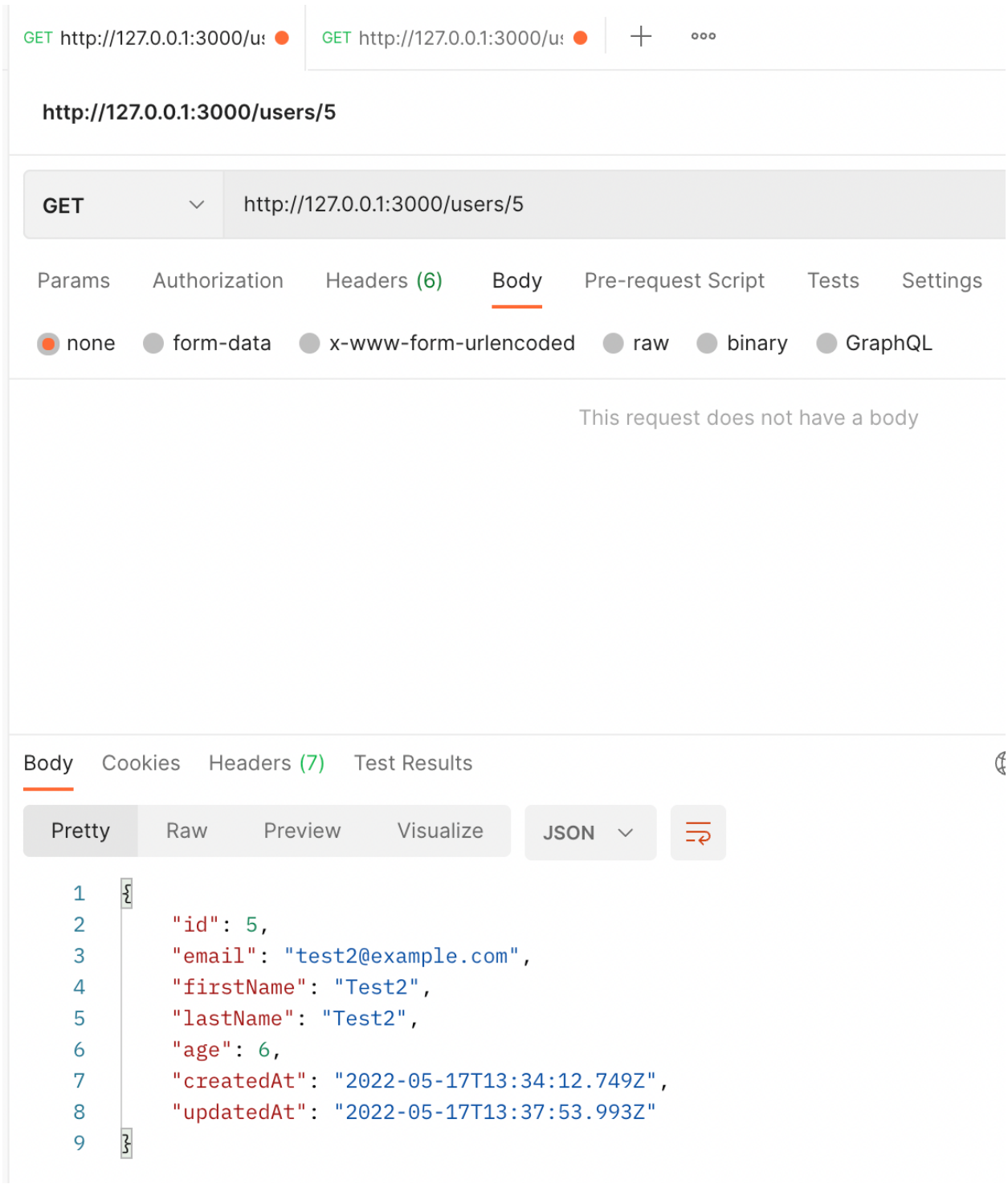
- Email
- Имя
- Фамилия
- Возраст

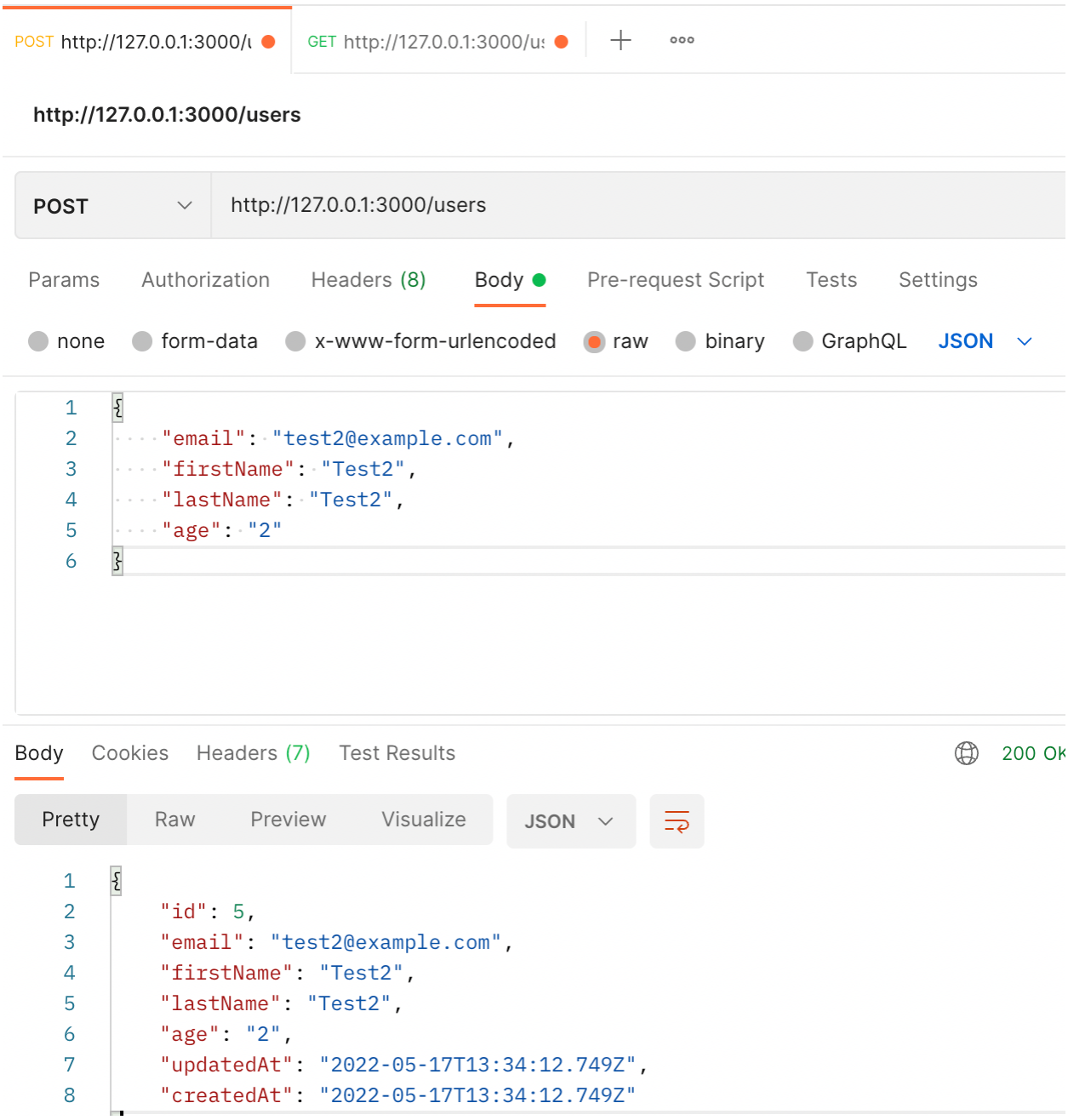
```
'use strict';
const {
  Model
} = require('sequelize');
module.exports = (sequelize, DataTypes) => {
  class User extends Model {
    /**
     * Helper method for defining associations.
     * This method is not a part of Sequelize lifecycle.
     * The `models/index` file will call this method automatically.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

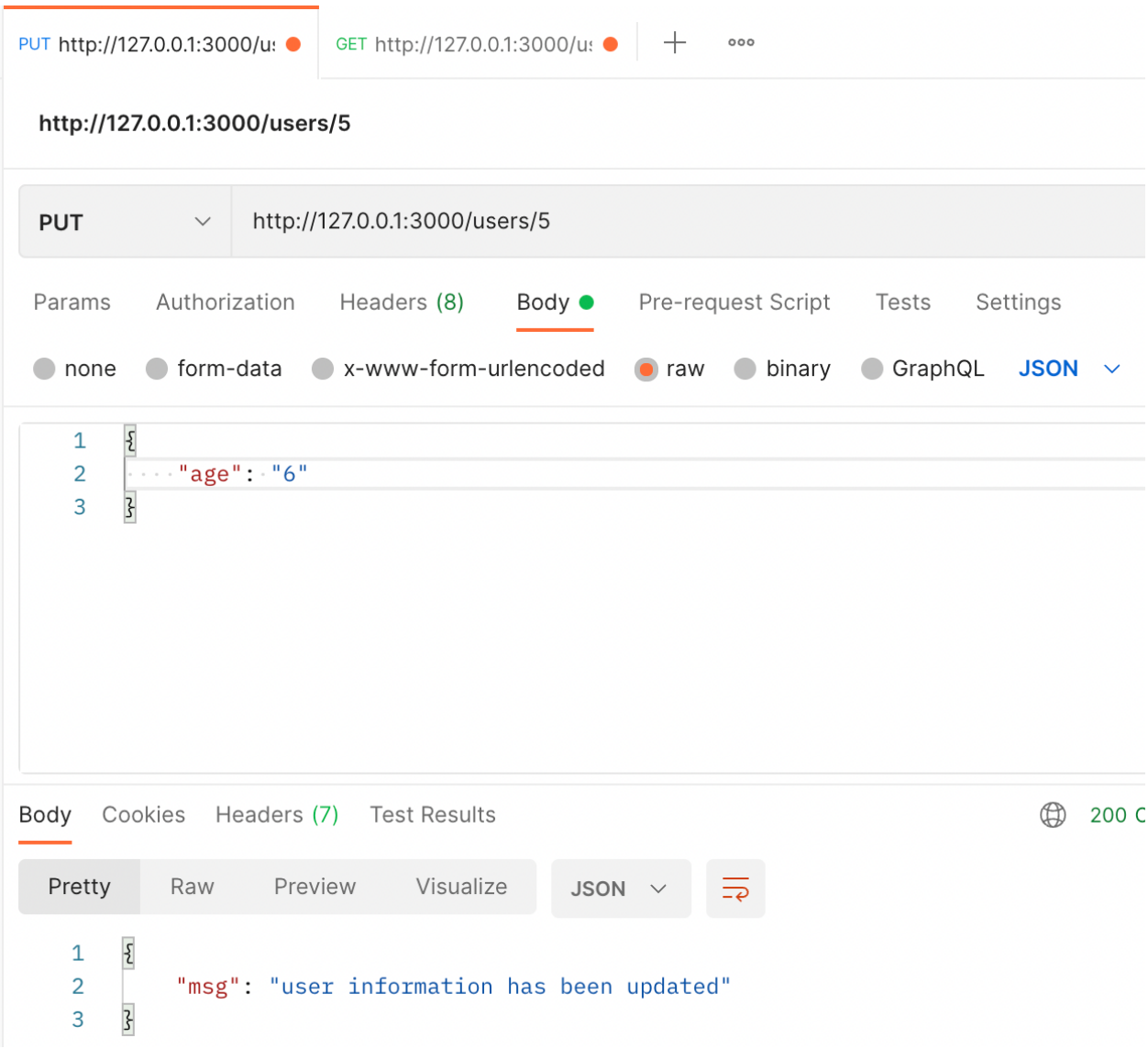
2) Реализованы запросы CRUD методов с помощью Express и Sequelize:

- Get /users – список всех пользователей
- Get /users/:id – информация о пользователе
- Put /users/:id – изменить информацию о пользователе
- Delete /users/:id – удалить пользователя

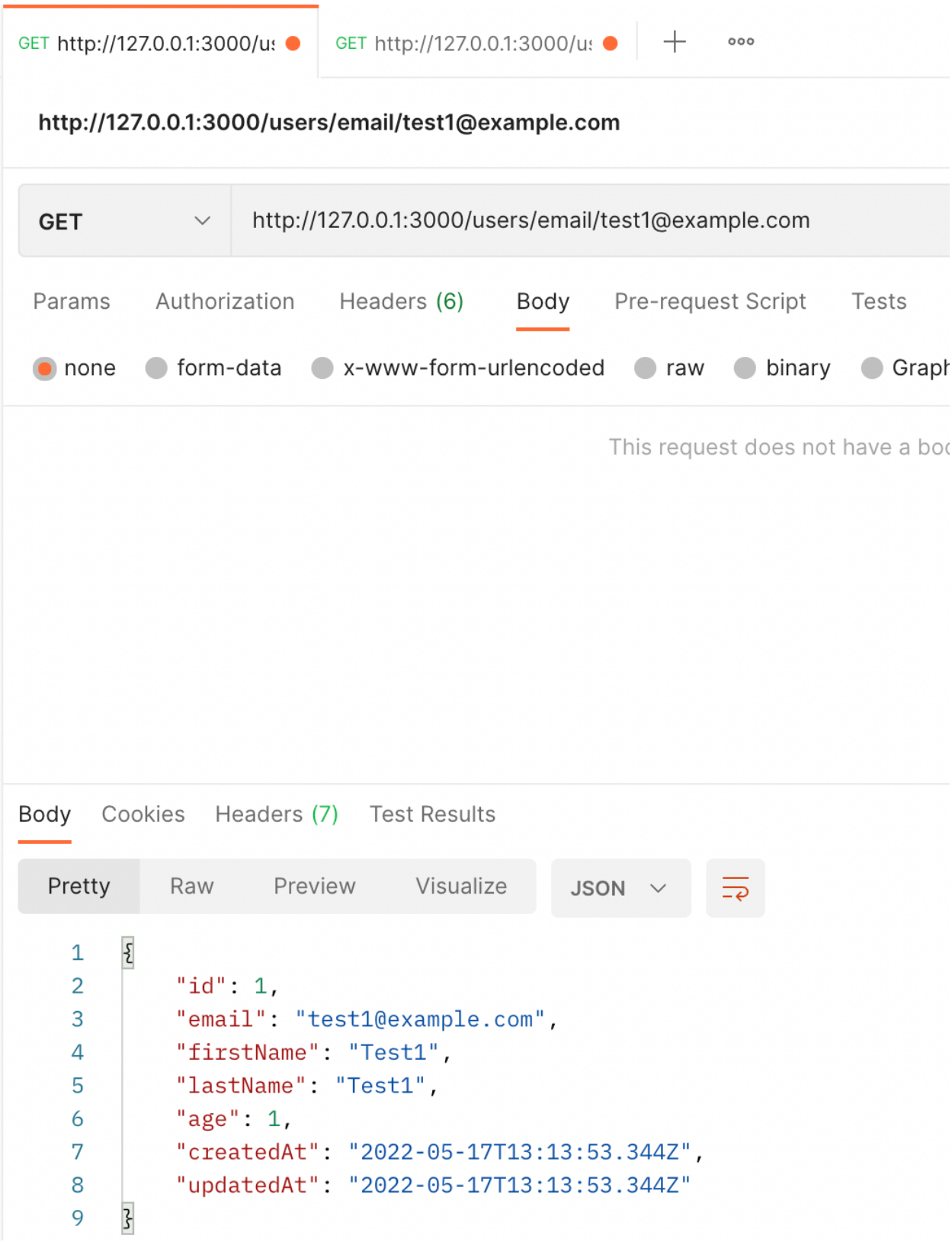

GET http://127.0.0.1:3000/us
GET http://127.0.0.1:3000/us
http://127.0.0.1:3000/users
GET
http://127.0.0.1:3000/users
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Query Params
KEY
VALUE
Key
Value
Body
Cookies
Headers (7)
Test Results
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 1,
"email": "test1@example.com",
"firstName": "Test1",
"lastName": "Test1",
"age": 1,
"createdAt": "2022-05-17T13:13:53.344Z",
"updatedAt": "2022-05-17T13:13:53.344Z"
},
{
"id": 5,
"email": "test2@example.com",
"firstName": "Test2",
"lastName": "Test2",
"age": 6,
"createdAt": "2022-05-17T13:34:12.749Z",
"updatedAt": "2022-05-17T13:37:53.993Z"


GET http://127.0.0.1:3000/us
GET http://127.0.0.1:3000/us
http://127.0.0.1:3000/users/5
GET
http://127.0.0.1:3000/users/5
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
This request does not have a body
Body
Cookies
Headers (7)
Test Results
Pretty
Raw
Preview
Visualize
JSON
{
"id": 5,
"email": "test2@example.com",
"firstName": "Test2",
"lastName": "Test2",
"age": 6,
"createdAt": "2022-05-17T13:34:12.749Z",
"updatedAt": "2022-05-17T13:37:53.993Z"
}


POST http://127.0.0.1:3000/
GET http://127.0.0.1:3000/u
http://127.0.0.1:3000/users
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
{
"email": "test2@example.com",
"firstName": "Test2",
"lastName": "Test2",
"age": "2"
}
Body
Cookies
Headers (7)
Test Results
200 OK
Pretty
Raw
Preview
Visualize
JSON
{
"id": 5,
"email": "test2@example.com",
"firstName": "Test2",
"lastName": "Test2",
"age": "2",
"updatedAt": "2022-05-17T13:34:12.749Z",
"createdAt": "2022-05-17T13:34:12.749Z"



3) Реализован метод получения пользователя по email.



Вывод

В ходе работы были освоены фреймворк Express и ORM Sequelize, основы работы с моделями, создание CRUD методов и фильтров.