УНИВЕРСИТЕТ ИТМО

Санкт-петербургский национальный исследовательский университет информационных технологий, механики и оптики

Факультет инфокоммуникационных технологий

Домашняя работа #2

По дисциплине Бекенд-разработка

Выполнил: Каратецкая М.Ю.
Группа № К33402
Проверил: Добряков Д.И.

Санкт-Петербург
2022

Задачи работы:

- 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. Модель пользователя

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

Создание модели

```
PS C:\Users\Professional\Desktop\ITMO-ICT-Backend-2022\homeworks\K33402\Karatetskaya_Maria\HW2> npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email
:string,town:string,dateBirth:date

Sequelize CLI [Node: 16.13.0, CLI: 6.4.1, ORM: 6.17.0]

New model was created at C:\Users\Professional\Desktop\ITMO-ICT-Backend-2022\homeworks\K33402\Karatetskaya_Maria\HW2\models\user.js .
New migration was created at C:\Users\Professional\Desktop\ITMO-ICT-Backend-2022\homeworks\K33402\Karatetskaya_Maria\HW2\migrations\20220315193305-create-user.js .
PS C:\Users\Professional\Desktop\ITMO-ICT-Backend-2022\homeworks\K33402\Karatetskaya_Maria\HW2>
```

2. Реализация Crud-методов

Создание/удаление/обновление

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

```
app.delete('/users/:id', async (req, res) => {
    const user = await db.User.findByPk(req.params.id)
        if (user) {
            user.destroy();
            res.sendStatus(200)
    }
    res.send({ "msg": "user is not found" })
})

app.put('/users/:id', async (req, res) => {
    const user = await db.User.findByPk(req.params.id)
    if (user) {
        await db.User.update(req.body, {
            where: {
                id: req.params.id
            }
        }
    )} else {
        await db.User.create(req.body)
    }
    res.send(req.body)
})
```

3. Запрос для получения пользователя по email/id

Так как id числовой тип, а email – текстовый(при обязательном наличии @), то они никогда не совпадут. Поэтому можно занести в один метод поиска, используя оператор or.

```
app.get('/users/:text', async (req, res) => {
    const user = await db.User.findAll({
        where: {
            [Op.or]: [
                {
                    id: req.params.text
                },
                {
                    email: req.params.text
                }
            ]
        }
    })
    if (user) {
        res.send(user)
    }

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

Вывод:

В ходе выполнения практической работы я ознакомилась с express и sequelize, а также смогла написать простой код для управлением пользователем и взаимодействие с бд.