САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бек-энд разработка

Отчёт

Домашнее задание №2 «Основы работы с ORM Sequelize»

Выполнил:

Шугинин Юрий

К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2022 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) Модель пользователя (файл “./models/user.js”).

```
'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,
    birthday: DataTypes.DATE,
    nationality: DataTypes.STRING,
    passport: DataTypes.STRING
  }, {
    sequelize,
    modelName: 'User',
  });
  return User;
};
```

2) CRUD-методы для работы с пользователями (файл “./index.js”).

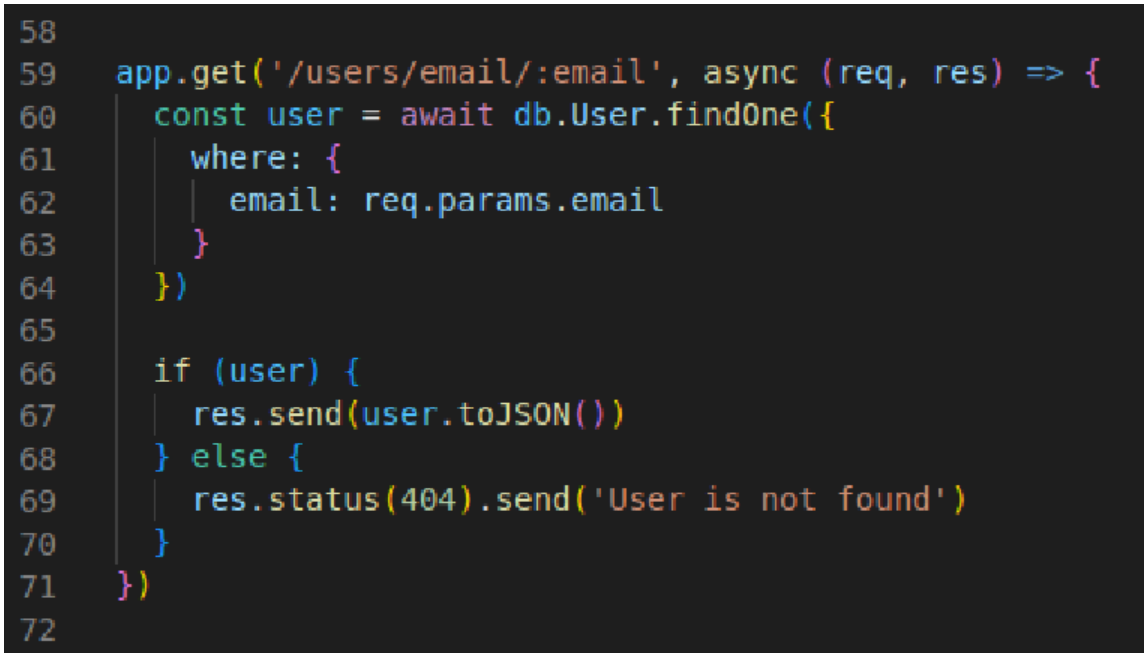
```

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

```

```
18
19    // Create a new user
20    app.post('/users', async (req, res) => {
21      const user = await db.User.create(req.body)
22      res.send(user.toJSON())
23    })
24
25    // Update user by id
26    app.put('/users/:id', async (req, res) => {
27      const user = await db.User.findByPk(req.params.id)
28
29      if (user) {
30        await user.update(req.body, { where: { id: req.params.id } })
31        res.send(user.toJSON())
32      } else {
33        res.status(404).send('User is not found')
34      }
35    })
36
37    // Delete user by id
38    app.delete('/users/:id', async (req, res) => {
39      const user = await db.User.findByPk(req.params.id)
40
41      if (user) {
42        await user.destroy({ where: { id: req.params.id } })
43        res.send('User deleted successfully')
44      } else {
45        res.status(404).send('User is not found')
46      }
47    })
48
```

3) Запрос для получения пользователя по id/email (файл “./index.js”).

```
48
49  app.get('/users/id/:id', async (req, res) => {
50    const user = await db.User.findByPk(req.params.id)
51
52    if (user) {
53      res.send(user.toJSON())
54    } else {
55      res.status(404).send('User is not found')
56    }
57  })
58
```

Рисунок 1 – Эндпоинт для получения пользователя по id



```
58
59  app.get('/users/email/:email', async (req, res) => {
60    const user = await db.User.findOne({
61      where: {
62        email: req.params.email
63      }
64    })
65
66    if (user) {
67      res.send(user.toJSON())
68    } else {
69      res.status(404).send('User is not found')
70    }
71  })
72
```

Рисунок 2 – Эндпоинт для получения пользователя по email

Вывод

В процессе выполнения домашнего задания были получены базовые навыки работы с микро-фреймворком Express и ORM Sequelize: создание моделей и миграций, реализация CRUD-методов, а также реализация эндпоинтов для конкретных задач.