САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая/Лабораторная работа

Выполнил:

Нгуен Хоанг Туан
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. index.js

```
const express = require('express')
const db = require('./models')

const bp = require('body-parser')
const app = express()
const port = 3000

app.use(bp.json())
app.use(bp.urlencoded({ extended: true }))


app.get('/', (req, res) => {
    res.send('Hello wo')
})

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


app.get('/users/email/:email', async (req, res) => {
    const user = await db.User.findOne({ where: { email: req.params.email }} )
    console.log('user is', user)
    res.send(user.toJSON())
})

app.get('/users/:id', async (req, res) => {
    const user = await db.User.findByPk(req.params.id)
    console.log('user is', user)
    res.send(user.toJSON())
})

app.delete('/users/delete/:id', async (req, res) => {
    const user = await db.User.destroy({
            where: {
```

```
                id: req.params.id
            }
        })
    console.log('user was deleted!', user)
    res.send('user was deleted!')
})
app.put('/users/update/:id', async (req, res) => {
    db.User.findOne({ where: {id: req.params.id}})
    .then(record => {
        if (!record) {
            throw new Error('No record found')
        }
        let values = {
            firstNAme: req.body.firstNAme,
            lastNAme: req.body.lastNAme,
            email: req.body.email,
        }
        record.update(values).then( updatedRecord => {
        })
        })
        .catch((error) => {
        throw new Error(error)
        })
        console.log('user was update')
        res.send('user was update')

})
app.post ('/create/', async (req,res) => {
    await db.User.create(req.body)
    console.log('user was create')
    res.send('user was create')
})
app.listen(port, () => {
    console.log(`Example app listening on port ${port}`)
})
```

2. 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
  }, {
    sequelize,
    modelName: 'User',
  });
  return User;
};
```

Вывод

Научился создавать простых пользователей с помощью Nodejs, использовал инструменты Express и Sequelize для создания CRUD-методов для работы с пользователями.