САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 1

Выполнила:
Штрейх Анна

Группа
К33401

Проверил:
Добряков Д. И.

Санкт-Петербург

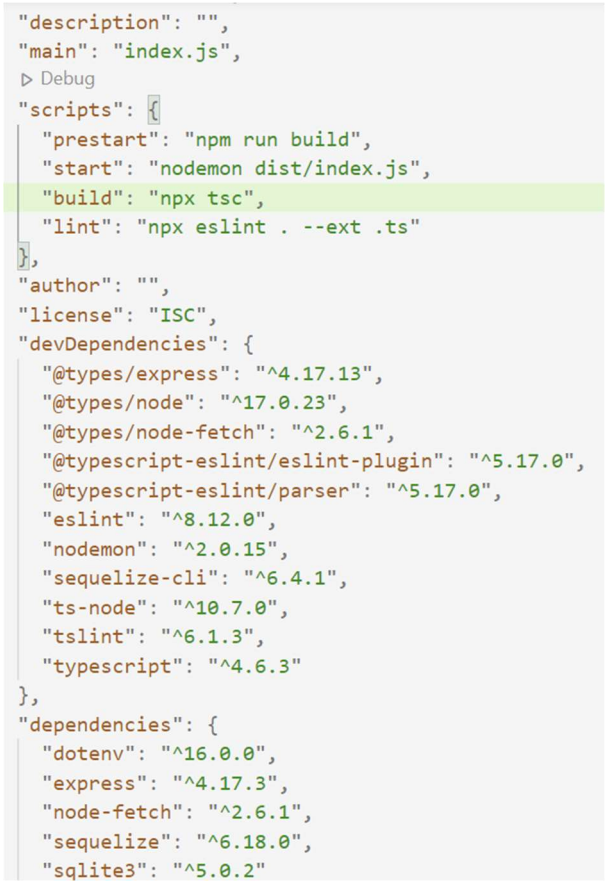
2022 г.

Задача

Нужно написать свой boilerplate на express + sequelize / TypeORM + typescript.

Ход работы

Командой npm init создается основа проекта на express – файлы конфигурации, в том числе package.json. Устанавливаются необходимые зависимости.



```
"description": "",
"main": "index.js",
▷ Debug
"scripts": {
  "prestart": "npm run build",
  "start": "nodemon dist/index.js",
  "build": "npx tsc",
  "lint": "npx eslint . --ext .ts"
},
"author": "",
"license": "ISC",
"devDependencies": {
  "@types/express": "^4.17.13",
  "@types/node": "^17.0.23",
  "@types/node-fetch": "^2.6.1",
  "@typescript-eslint/eslint-plugin": "^5.17.0",
  "@typescript-eslint/parser": "^5.17.0",
  "eslint": "^8.12.0",
  "nodemon": "^2.0.15",
  "sequelize-cli": "^6.4.1",
  "ts-node": "^10.7.0",
  "tslint": "^6.1.3",
  "typescript": "^4.6.3"
},
"dependencies": {
  "dotenv": "^16.0.0",
  "express": "^4.17.3",
  "node-fetch": "^2.6.1",
  "sequelize": "^6.18.0",
  "sqlite3": "^5.0.2"
```

Рисунок 1 – Файл package.json

В директории controllers созданы контроллеры, использующие сервисы UserServices для работы с пользователями (CRUD Методы).

Рисунок 2 – Контроллеры

Рисунок 3 – Сервисы

Все пути(эндпоинты) прописываются в директории Routes.

routes > TS userRoutes.ts > ...

```ts
import express from "express"
import UserController from "../controllers/users/userController"

const router = express.Router()

const userController = new UserController()

router
  .route('/users')
  .get(userController.get)

router
  .route('/users/id/:id')
  .get(userController.getbyID)

router
  .route('/users/email/:email')
  .get(userController.getbyEmail)
```

Рисунок 4 – файл UserRoutes

Для работы с sqlite использовался sequelize.

db > models > TS index.ts > ...

```ts
import { Sequelize } from 'sequelize';

const env = process.env.NODE_ENV || 'development';
const config = require(__dirname + '/../../config.json')[env];

const  sequelize = config.url
  ? new Sequelize(config.url, config)
  : new Sequelize(config.database, config.username, config.password, config);

export { Sequelize, sequelize };
```

Рисунок 5 – Файл index.ts в models

{} config.json > ...

```json
{
  "development": {
    "username": null,
    "password": null,
    "database": "database_development",
    "host": null,
    "dialect": "sqlite",
    "storage": "db.sqlite"
  },
  "test": {
    "username": "root",
    "password": null,
    "database": "database_test",
    "host": "127.0.0.1",
    "dialect": "sqlite"
  },
  "production": {
    "username": "root",
    "password": null,
    "database": "database_production",
```

Рисунок 6 – Файл config.json

```
import { DataTypes, Model, Optional } from 'sequelize';
import { sequelize } from '.';

interface UserAttributes {
    id: number;
    firstName: string;
    lastName: string;
    email: string;
    password: string;
    age: number;
    country: string;
};

interface UserCreationAttributes
    extends Optional<UserAttributes, 'id'> { }

interface UserInstance
    extends Model<UserAttributes, UserCreationAttributes>,
    UserAttributes {
    createdAt?: Date;
    updatedAt?: Date;
}

const User = sequelize.define<UserInstance>(
    'User',
    {
        id: {
            allowNull: false,
            autoIncrement: false,
            primaryKey: true,
```

Рисунок 7 – Модель пользователя, файл User.ts

После создания модели с помощью sequelize-cli проведены миграции. Созданы первые пользователи с помощью сидеров.

```
'use strict';

module.exports = {
  up: (queryInterface, Sequelize) => {
    return queryInterface.bulkInsert('Users', [{
      firstName: 'John',
      lastName: 'Doe',
      email: 'example@example.com',
      password: '1234GGGg',
      age: 21,
      country: 'USA',
      createdAt: new Date(),
      updatedAt: new Date()
    },
    {
      firstName: 'Anna',
      lastName: 'Doe',
```

Рисунок 8 – Файл в seeders

Выводы:

Успешно написан первый шаблон проекта на express, все эндпоинты работоспособны и возвращают информацию о добавленных в базу данных пользователях.