

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет по домашней работе
Знакомство с ORM Sequelize

Выполнил:

Малышенко А. Р.

Группа К33392

Проверил:

Добряков Д. И.

Санкт-Петербург

2024 г.

Содержание отчета

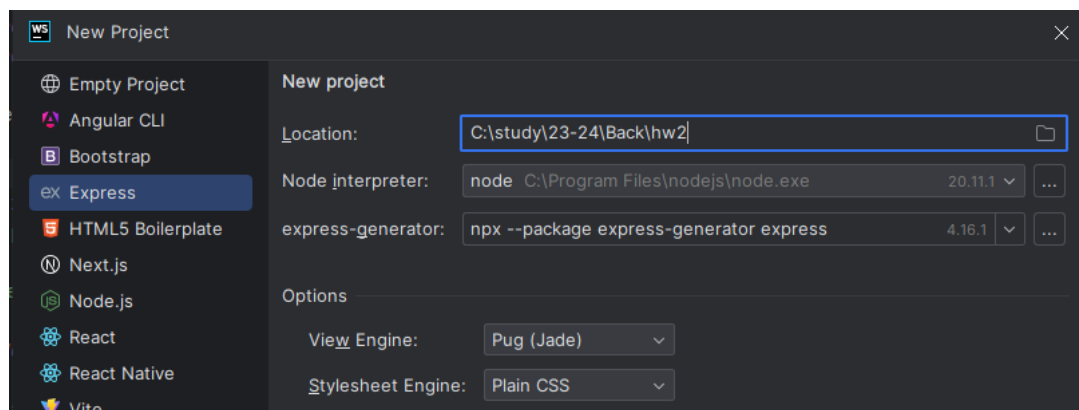
1) Задача	3
2) Создание проекта	3
3) Установка зависимостей	4
4) Инициализация	4
5) Создание модели User	4
6) Создание эндпоинтов	5

1) 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.

2) Создание проекта

Проект создается в ide WebStorm, благодаря чему express устанавливается в проект по умолчанию



```
"dependencies": {  
  "cookie-parser": "~1.4.4",  
  "debug": "~2.6.9",  
  "express": "~4.16.1",  
  "fs": "^0.0.1-security",  
}
```

3) Установка зависимостей

Устанавливаем необходимые зависимости:

> *npm install sqlite3 sequelize sequelize-cli*

```
"dependencies": {
  "cookie-parser": "~1.4.4",
  "debug": "~2.6.9",
  "express": "~4.16.1",
  "fs": "^0.0.1-security",
  "http-errors": "~1.6.3",
  "morgan": "~1.9.1",
  "nodemon": "^3.1.0",
  "path": "^0.12.7",
  "pug": "2.0.0-beta11",
  "sequelize": "^6.37.1",
  "sequelize-cli": "^6.6.2",
  "sqlite3": "^5.1.7"
}
```

4) Инициализация

> *npx sequelize-cli init*

```
PS C:\study\23-24\Back\examples\hw2_ex> npx sequelize-cli init

Sequelize CLI [Node: 20.11.1, CLI: 6.6.2, ORM: 6.37.1]

Created "config\config.json"
Successfully created models folder at "C:\study\23-24\Back\examples\hw2_ex\models".
Successfully created migrations folder at "C:\study\23-24\Back\examples\hw2_ex\migrations".
Successfully created seeders folder at "C:\study\23-24\Back\examples\hw2_ex\seeders".
```

5) Создание модели User

> *npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email:string,age:integer*

```
PS C:\study\23-24\Back\examples\hw2_ex> npx sequelize-cli model:generate --name User --attributes firstName:string,lastName:string,email:string,age:integer

Sequelize CLI [Node: 20.11.1, CLI: 6.6.2, ORM: 6.37.1]

New model was created at C:\study\23-24\Back\examples\hw2_ex\models\user.js .
New migration was created at C:\study\23-24\Back\examples\hw2_ex\migrations\20240313103638-create-user.js .
```

Для миграции используем следующую команду:

> *npx sequelize db:migrate*

```
PS C:\study\23-24\Back\examples\hw2_ex> npx sequelize db:migrate

Sequelize CLI [Node: 20.11.1, CLI: 6.6.2, ORM: 6.37.1]

Loaded configuration file "config\config.json".
Using environment "development".
== 20240313103638-create-user: migrating =====
== 20240313103638-create-user: migrated (0.019s)
```

Так же перед миграцией надо не забыть изменить тип используемой ORM в config.json на sqlite:

```
host: null,
"dialect": "sqlite",
"storage": "database.sqlite",
"logging": false
```

6) Создание эндпоинтов

Создание пользователя

```
// create new user
app.post('/users', async (req, res) => {
  await db.User.create(req.body).then(() => {
    res.send({"msg": 'user created'})
  })
})
```

Вывод пользователей всех пользователей

```
// get users
app.get('/users', async (req, res) => {
  const users = await db.User.findAll()
  if (!users) {
    return res.send({"msg": "users is not found"})
  }

  return res.send(users)
})
```

Вывод пользователя по его id

```
// get user by id
app.get('/users/:id', async (req, res) => {
  const user = await db.User.findById(req.params.id)
  if (user) {
    return res.send(user)
  }

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

Удаление пользователя по его id

```
// delete user by id
app.delete('/users/:id', async (req, res) => {
  const user = await db.User.findByPk(req.params.id)
  if (user) {
    await user.destroy()
    return res.send({"msg": "user deleted"})
  }

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