

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

по лабораторной работе №1

Выполнил:

Кузгиев Адам

**Группа
К33402**

Проверил:

Добряков Д. И.

Санкт-Петербург

2022 г.

Задача

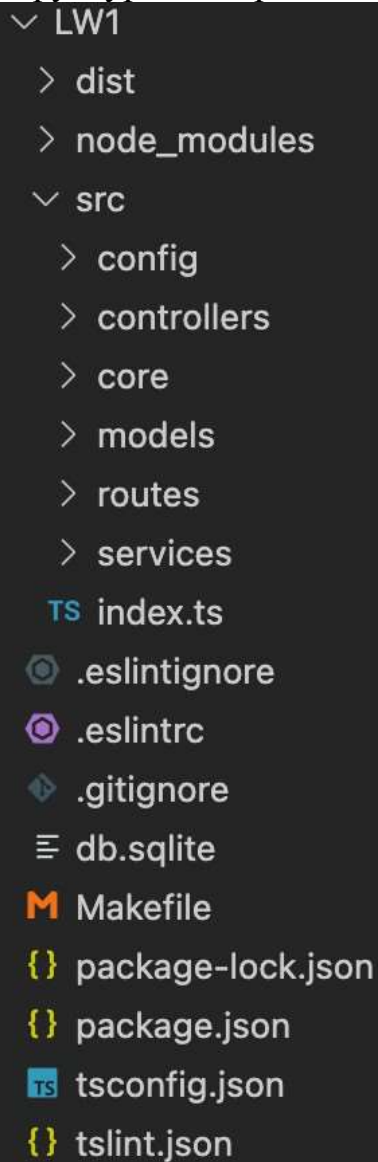
Написать свой boilerplate на express + sequelize / TypeORM + typescript.

Должно быть явное разделение на:

- модели
- контроллеры
- роуты
- сервисы для работы с моделями (реализуем паттерн “репозиторий”)

Ход работы

Структура boilerplate



```

✓ LW1
  > dist
  > node_modules
  ✓ src
    > config
    > controllers
    > core
    > models
    > routes
    > services
    TS index.ts
  .eslintignore
  .eslintrc
  .gitignore
  ≡ db.sqlite
  M Makefile
  {} package-lock.json
  {} package.json
  TS tsconfig.json
  {} tslint.json
```

Модель

```

1  import { Table, Column, Model } from 'sequelize-typescript'
2
3  @Table
4  export default class User extends Model {
5      @Column
6      nickname: string
7
8      @Column
9      password: string
10
11     @Column
12     email: string
13 }

```

Контроллеры

```

1  import DefaultService from '../services/userService'
2
3  export default class ExampleController {
4
5      private service = new DefaultService()
6
7      post = async (request: any, response: any) => {
8          try {
9              const user = request.body
10             await this.service.add(user.nickname, user.password, user.email)
11             response.send('Added' + " " + user.nickname + " " + "user")
12         } catch (error: any) {
13             response.status(400).send(error.message)
14         }
15     }
16
17     get = async (request: any, response: any) => {
18         try {
19             const data = await this.service.get()
20             response.send(data)
21         } catch (error: any) {
22             response.status(400).send(error.message)
23         }
24     }
25 }

```

Сервисы

```

1  import User from '../models/User'
2  import { sequelize } from '../config/config'
3
4  export default class DefaultService {
5
6      private repo = sequelize.getRepository(User)
7
8      add(nickname: string, password: string, email: string) {
9          this.repo.create({ nickname: nickname, password: password, email: email })
10     }
11
12     get() {
13         return this.repo.findAll()
14     }
15 }

```

Роуты

```

1  import express from "express"
2  import ExampleController from '../controllers/userController'
3
4  const router: express.Router = express.Router()
5
6  const exampleController = new ExampleController()
7
8  router
9      .route('/')
10     .get(exampleController.get)
11     .post(exampleController.post)
12
13 export default router

```

Примеры работы

http://127.0.0.1:8008/v1

POST	▼	http://127.0.0.1:8008/v1
------	---	--------------------------

Params Authorization Headers (8) Body ● Pre-request Script

Query Params

	KEY	VALUE
	Key	Value

Body Cookies Headers (7) Test Results

Pretty	Raw	Preview	Visualize	HTML ▼	≡
--------	-----	---------	-----------	--------	---

```
1 Added 'test01' user
```

http://127.0.0.1:8008/v1

GET	▼	http://127.0.0.1:8008/v1
-----	---	--------------------------

Params Authorization Headers (6) Body Pre-request Script

Query Params

	KEY	VALUE
	Key	Value

Body Cookies Headers (7) Test Results

Pretty	Raw	Preview	Visualize	JSON ▼	≡
--------	-----	---------	-----------	--------	---

```
1  {
2    {
3      "id": 1,
4      "nickname": "test01",
5      "password": "test010101",
6      "email": "test01@gmail.com",
7      "createdAt": "2022-05-25T10:01:00.364Z",
8      "updatedAt": "2022-05-25T10:01:00.364Z"
9    }
10 }
```

Вывод

В ходе работы был создан boilerplate на express, Sequelize, typescript. Созданный boilerplate можно использовать для создания полноценного backend приложения.