

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа

Выполнил:

Шайтор Илья

Группа К3339

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- Реализовать все модели данных, спроектированные в рамках ДЗ1
- Реализовать набор из CRUD-методов для работы с моделями данных средствами nest.js + PrismaORM + TypeScript
- Реализовать API-эндпоинт для получения пользователя по id/email

Ход работы

Нужно написать свой boilerplate на nest.js + PrismaORM

Структура:

```
> dist
> node_modules library root
└─ prisma
    ▲ schema.prisma
└─ src
    └─ application
        ├── application.controller.ts
        ├── application.dto.ts
        ├── application.module.ts
        └── application.service.ts
    └─ company
        ├── company.controller.ts
        ├── company.dto.ts
        ├── company.module.ts
        └── company.service.ts
    └─ conception
        ├── guard.ts
        ├── middleware.ts
        └── pipe.ts
    └─ education
        ├── education.controller.ts
        ├── education.dto.ts
        ├── education.module.ts
        └── education.service.ts
    └─ Industry
        ├── industry.controller.ts
        ├── industry.dto.ts
        ├── industry.module.ts
        └── industry.service.ts
```

- ▼ resume
 - TS resume.controller.ts
 - TS resume.dto.ts
 - TS resume.module.ts
 - TS resume.service.ts

- ▼ users
 - TS users.controller.ts
 - TS users.dto.ts
 - TS users.module.ts
 - TS users.service.ts

- ▼ vacancy
 - TS vacancy.controller.ts
 - TS vacancy.dto.ts
 - TS vacancy.module.ts
 - TS vacancy.service.ts

- > workExperiences
 - TS app.module.ts
 - TS main.ts
 - TS prisma.service.ts
 - TS types.ts

- > test

- ≡ .env

- 🔗 .gitignore

- 📄 .prettierrc

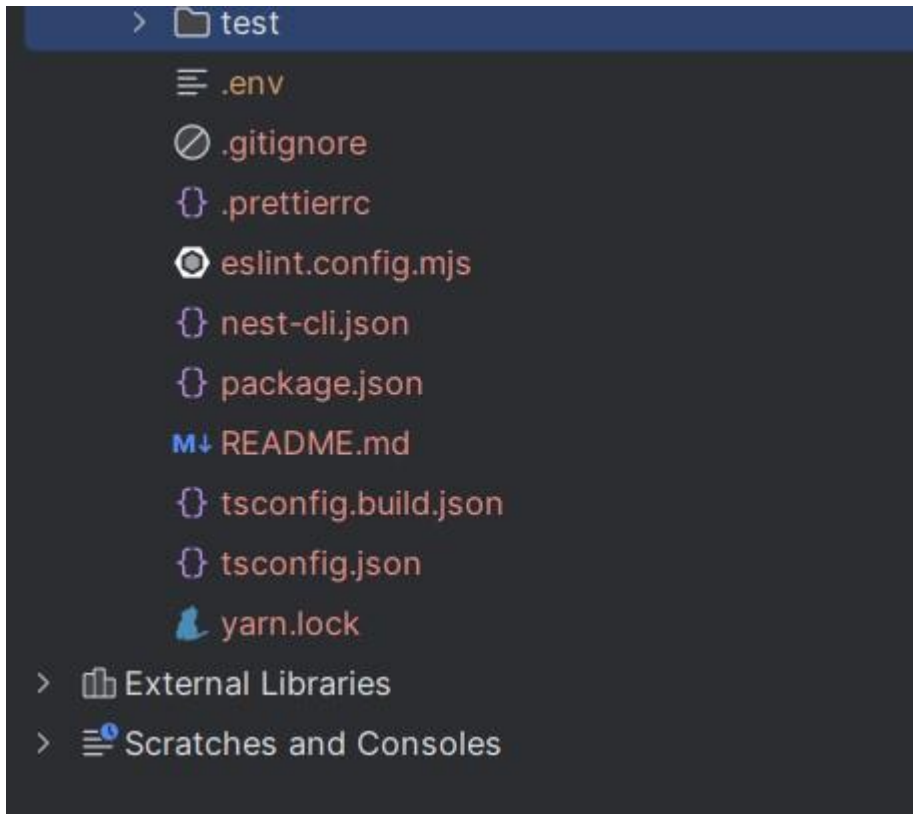
- ⚙️ eslint.config.mjs

- 📄 nest-cli.json

- 📄 package.json

- M↓ README.md

- 📄 tsconfig.build.json



Крудсы в сваггере:

Users			
GET	/api/users	Получить всех пользователей	🔒
POST	/api/users	Создать нового пользователя	
GET	/api/users/{id}	Получить пользователя по ID	
PUT	/api/users/{id}	Обновить пользователя	🔒
DELETE	/api/users/{id}	Удалить пользователя	🔒
POST	/api/users/login	Вход пользователя и получение JWT-токена	

Vacancies			
GET	/api/vacancys	Получить все вакансии	🔒
POST	/api/vacancys	Создать новую вакансию	🔒
GET	/api/vacancys/{id}	Получить вакансию по ID	
PUT	/api/vacancys/{id}	Обновить вакансию	🔒
DELETE	/api/vacancys/{id}	Удалить вакансию	🔒

Work Experiences			^
GET	/api/workExperiences	Получить весь опыт работы	🔒 ▼
POST	/api/workExperiences	Создать новый опыт работы	🔒 ▼
GET	/api/workExperiences/{id}	Получить опыт работы по ID	▼
PUT	/api/workExperiences/{id}	Обновить опыт работы	🔒 ▼
DELETE	/api/workExperiences/{id}	Удалить опыт работы	🔒 ▼

Resumes			^
GET	/api/resumes	Получить все резюме	🔒 ▼
POST	/api/resumes	Создать новое резюме	🔒 ▼
GET	/api/resumes/{id}	Получить резюме по ID	🔒 ▼
PUT	/api/resumes/{id}	Обновить резюме	🔒 ▼
DELETE	/api/resumes/{id}	Удалить резюме	🔒 ▼

Industry			^
GET	/api/industry	Получить все отрасли	🔒 ▼
POST	/api/industry	Создать новую отрасль	🔒 ▼
GET	/api/industry/{id}	Получить отрасль по ID	▼
PUT	/api/industry/{id}	Обновить отрасль	🔒 ▼
DELETE	/api/industry/{id}	Удалить отрасль	🔒 ▼

Вывод: реализовали крудсы