**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Лабораторная работа №6**

**Добавление авторизации и пользовательских сессий**

Выполнила студент группы №**М33091**

**Чу Тхи Фыонг Тхао**

Проверил

**Приискалов Роман Андреевич**

САНКТ-ПЕТЕРБУРГ

2022

Install the required Passport packages.

$npm install --save @nestjs/passport passport @nestjs/jwt passport-jwt

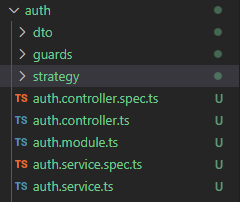
$npm install --save-dev @types/passport-jwt

Создать ресурс “auth”

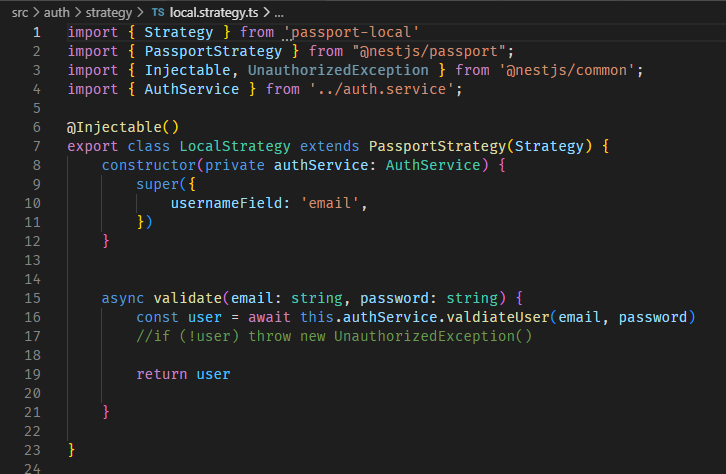
Используйте интерфейс командной строки Nest для создания «resource» для «auth».

$nest g resource auth

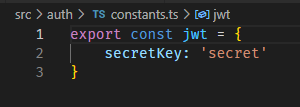
результат как:



Создала файл **auth/strategy/local.strategy.ts**:

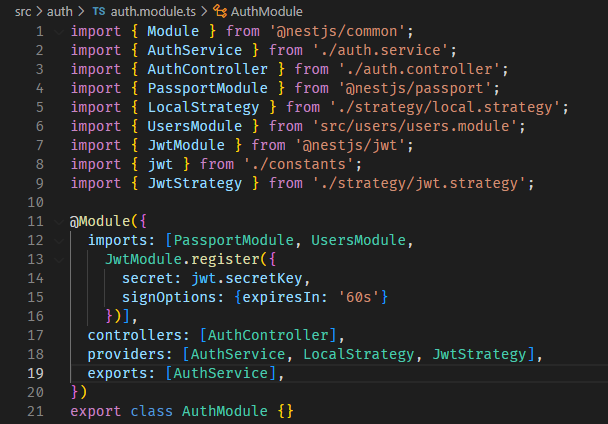


Создал файл **constants.ts** в катогори auth



Чтобы использовать Passport, нужно импортировать PassportModule и JwtModule в AuthModule.

**auth.module.ts**

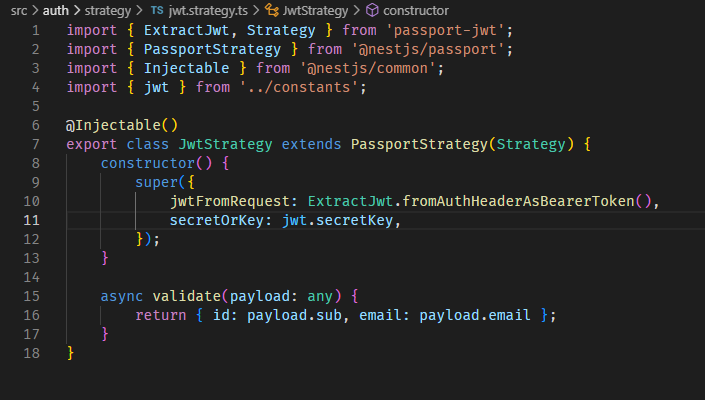


Create **JwtStrategy**

Следующим шагом является создание «JwtStrategy», для которого требуется действительный токен JWT для защищенной конечной точки.

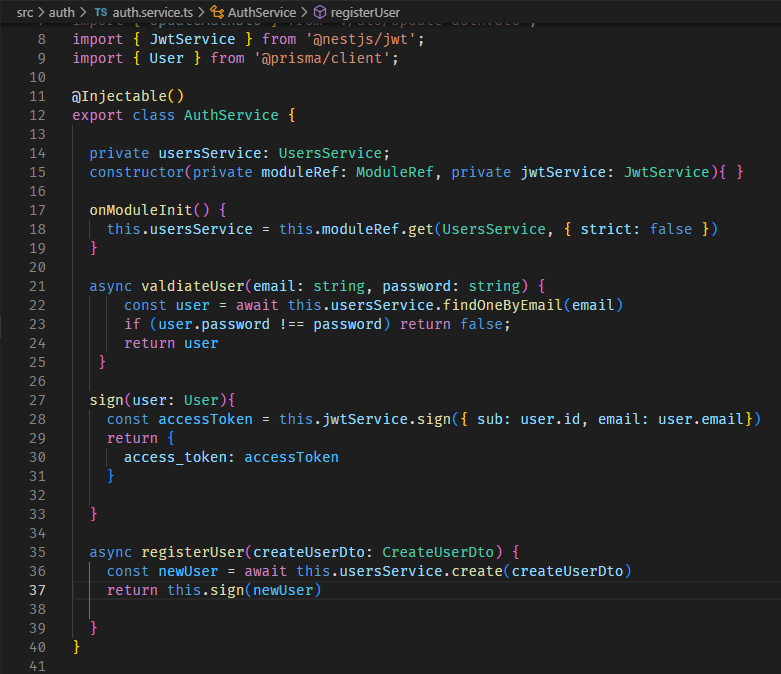
Начните с создания класса JwtStrategy в каталоге auth/strategy и выполните следующие шаги:

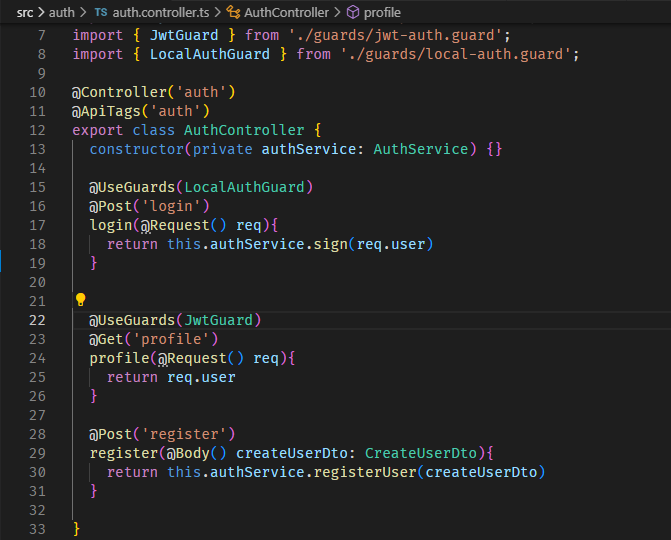
**jwt.strategy.ts**



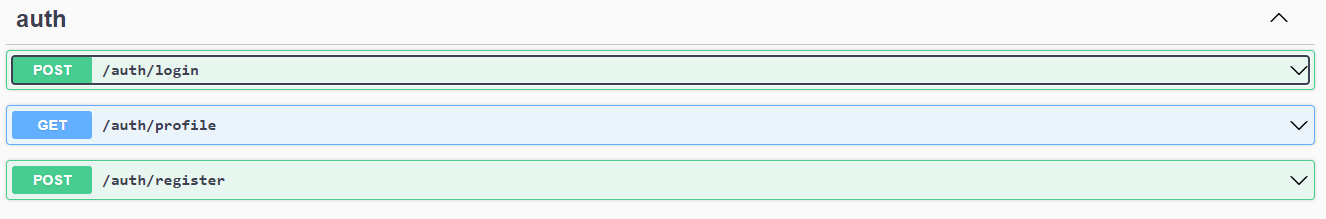
Реализовать логику в AuthService и AuthController

**auth.service.ts**

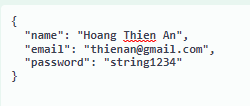


**auth.controller.ts** 

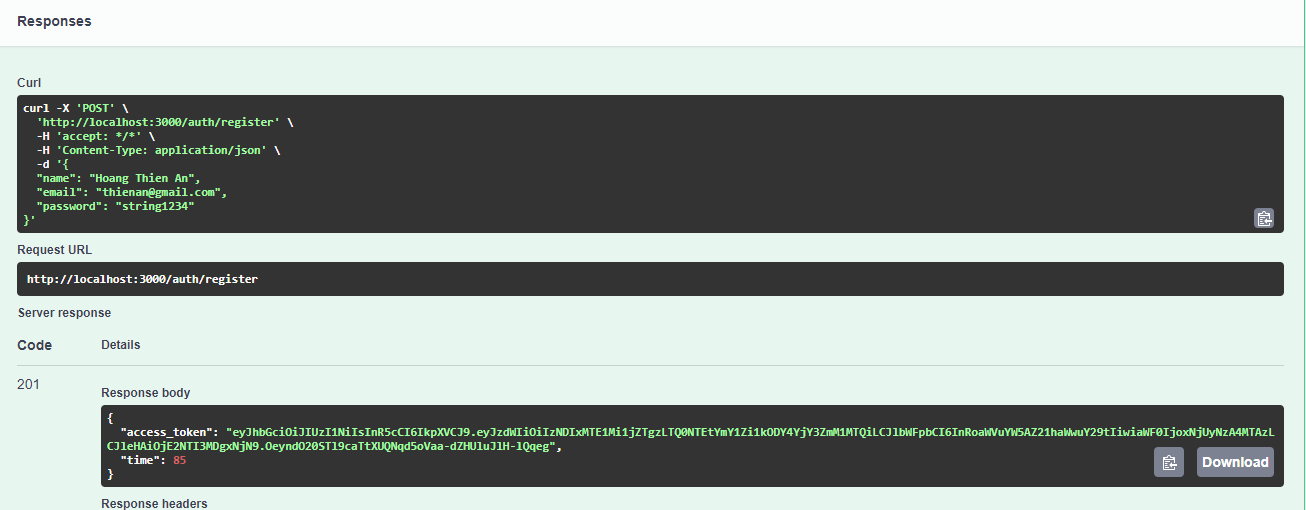
Результат



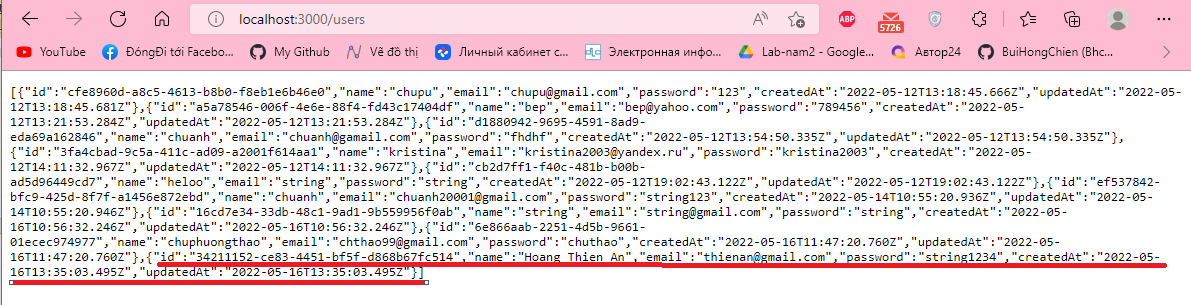
Пример register:



Получила результат:



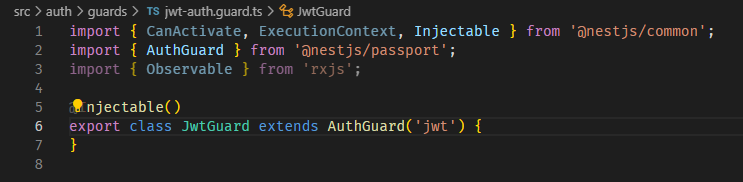
Проверила в Uesrs:



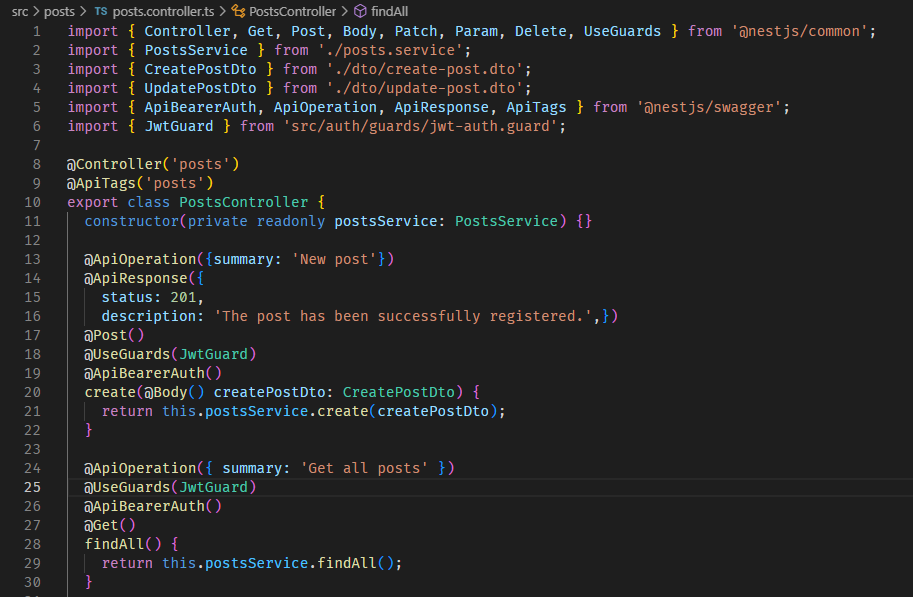
Create **JwtAuthGuard** and protect endpoints

Теперь можете создать защиту для защиты конечных точек и проверки токена JWT. Интерфейс командной строки Nest позволяет вам сгенерировать шаблон для защиты, но здесь вам понадобится только простой класс защиты, расширенный AuthGuard, предоставленный @nestjs/passport. Передайте имя стратегии «jwt» в AuthGuard.

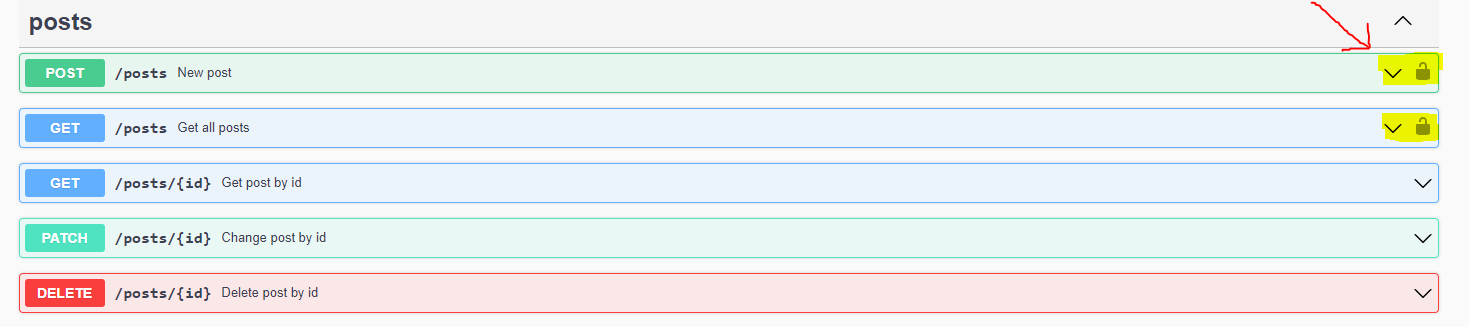
**/src/guards/jwt-auth.guard.ts**



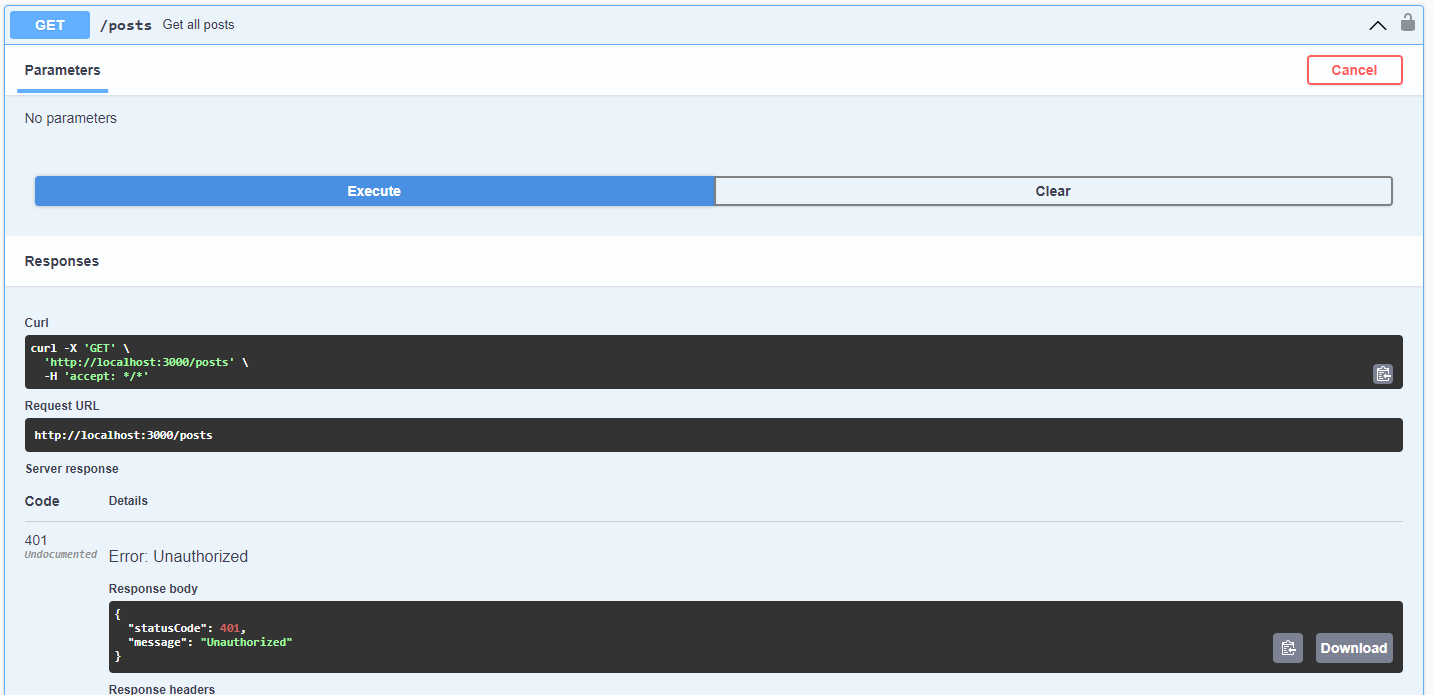
Примените **JwtGuard** с декоратором **@UseGuards()** для каждой конечной точки, например для защиты. Swagger еще не знает о защищенных конечных точках. Отметьте эти маршруты с помощью декоратора **@ApiBearerAuth()** и впоследствии обновите Swagger API. Пример для Post:

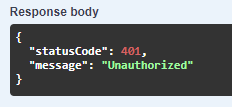


Результат получила:



Test protected endpoints





copy the JWT access token and paste it into the Authorize 🔓 input field

