Controle das Etapas do projeto – Back End

Iniciar o projeto

npm i -g @nestjs/cli

nest new nome-do-projeto

Gerar Modulo, controller, service e middleware de auth

nest generate module auth

nest generate controller auth

nest generate service auth

nest generate middleware auth

Gerar o comando para criar a pasta config, migrations e models do banco de dados postgreSql com o ORM Sequelize.

npm install --save sequelize sequelize-typescript pg pg-hstore

npm install --save-dev sequelize-cli

npx sequelize-cli init

Organizar o arquivo config para que se adeque com uma string de conexão.

Organizar o arquivo da pasta Models para que permita criar o banco.

Gerar as colunas da tabela de usuários

npx sequelize-cli migration:generate --name create-users

Criar a tabela no banco de dados.

npx sequelize-cli db:migrate

Criar dentro da pasta config, dentro de src, a pasta models, o qual irá chamar o banco de dados no servidor.

Criar dentro da pasta config, dentro de src, a pasta Database e criar o arquivo database.module.ts, nele é aonde se conectará o banco de dados com o servidor. Depois de tudo deve-se importar o modulo para o app.module.ts, para que o banco se torne global a aplicação.

Criar o authModule, authController, authService e Middleware, importar o authModule em app.module.ts.

Importar o modulo do Sequelize, para que se possa injetar o banco de dados no service.

Criar um centralizador de middlewares chamado middleware.config.ts e colocá-lo na raiz de src.

Chamar o centralizador de middleware no modulo authModule.

Criar o middleware de /auth/register chamado register.middleware.ts.

Criar o controller de /auth/register, sendo a rota /auth global.

Criar o service, afim de se criar as regras de negócios.

Instalar o Bcrypt e criar a classe para ser chamada dentro do service, colocar o Bcrypt em providers para que se possa injetar a dependência no service.

Instalar o Bcrypt e criar a classe para ser chamada dentro do service, colocar o Bcrypt em providers para que se possa injetar a dependência no service.

Criar um método privado para retornar os dados, menos a senha.

Criar o arquivo .http, para se testar a rota. OBS: Para se usar este arquivo, você precisa ter instalado a extensão “httpBook – Rest Client”.

Criar o middleware de login, colocá-lo no middleware.config, criar a rota de login dentro do controller de auth e criar o service.

Criar a classe JWT, chamar nos providers de authModule, para poder injetar sua dependência no service.

Criar o modulo, controller e service de users.

Criar o controller de findAll.

Criar dentro de config que está dentro de src, uma pasta chamada guard, criando o arquivo RoleGuard para configurar autorização entre admin e usuários comuns.

Chamar o RolesGuard abaixo do decorator de controller de user, dentro do decorator de @UseGuards.

Dentro da pasta config, dentro da pasta src, criar uma pasta chamada decorator. Lá, criar o arquivo roles.decorator.ts

Aplicar o decorator de @Roles nos verbos que permitem apenas admin, como no findAll.

Fazer o service de findAll.

Fazer o middleware de findAll e permitir apenas que administradores acessem ao middleware.

Criar controller de findOne.

Criar um middleware de autenticação para que quando o usuário faça login, crie um usuário autenticado.

Criar service de findOne e permitir que o usuário apenas veja os seus próprios dados.

Criar middleware de findOne para que valide se o usuário está recebendo em seu parâmetro um UUID.

Criar controller de update.

Criar service de update e não permitir que usuários comuns atualizem outros usuários e que nem consigam atualizar o campo role.

Criar o middleware de update.

Criar controller de remove e colocar o decorator @Roles para que apenas administradores acessem a rota.

Fazer o service de remove.

Fazer o middleware de remove e permitir apenas que administradores acessem o middleware.

Fazer um endpoint que listasse usuário inativos.

Fazer testes unitários de cada middleware, controller e service.

Fazer testes de API de cada rota.

Integre com um provedor de login de terceiros, como Google ou Microsoft , para permitir autenticação via OAuth 2.0.