Projeto Nest.JS e Vue.JS

Alunos: Augusto Castejon, Álvaro Dias, Luíz Felippe e Vinicius Dogonski

Passo 1: Preparação inicial

Abra o terminal. Em seguida:

1.1: criar uma pasta para o projeto e navegue até ele pelo terminal:

→ mkdir Projeto-Nest

→ cd Projeto-Nest

1.2: Atualize os Pacotes (Opcional, mas recomendado) Antes de instalar o Node.js, é uma boa prática atualizar os pacotes do sistema:

→ sudo apt update
→ sudo apt upgrade

1.3: instalar o Volta.

→ curl https://get.volta.sh | bash

1.4 configurar as variáveis de ambiente:

→ export VOLTA_HOME=”$HOME/ .volta”

→ export PATH=”$VOLTA_HOME/bin:$PATH”

1.5 verifique a intalação

volta –version

1.6. Instale a versão LTS mais recente do Node.js:

→ volta install node

1.7. Verifique se o Node.js foi instalado corretamente:

→ node -v

1.8: Instale o npm (Node Package Manager) O npm é instalado automaticamente com o Node.js. No entanto, você pode verificar se ele está funcionando corretamente:

→ npm -v

Agora você deve ter o Node.js e o npm instalados e prontos para uso em seu sistema Linux. Lembre-se de que a versão dos comandos pode variar, então sempre verifique os sites oficiais do Node.js e do Volta para obter as informações mais atualizadas.

1.9: setar o ambiente virtual do projeto

1.9.1 Sentando versão do Node:

→ volta pin node@18.17.1

1.9.2 Sentando versão do npm:

→ volta pin npm@9.6.7

Passo 2: Configuração do Back-End com NestJS

2.1. Instale o Nest CLI globalmente:

→ npm install -g @nestjs/cli

2.2. Crie um novo projeto NestJS:

→ nest new backend

2.3. Navegue para a pasta do projeto:

→ cd backend

Passo 3: Configuração do Banco de Dados MongoDB

Para baixar o mongoDB em sua máquina basta acessar Download MongoDB Community Server | MongoDB e baixar a versão gratuita para a sua máquina e sistema operacional

Para saber a sua versão do linux use o comando:

→ lsb-release -a

Após instalar use o comando:

→ sudo systemctl status mongod

Se tudo ocorreu perfeitamente, irá aparecer o serviço do mongod, contudo ele estará inativo. Então temos que ativa-lo, para isso use o comando:

→ sudo systemctl start mongod

Agora seu banco está funcionando, porém temos que instalar a parte gráfica para ajudar na usabilidade, acesse Download MongoDB Community Server | MongoDB novamente e baixe a interface gráfica MongoDB Compass (GUI)

Após isso abra o MongoDB Compass defina sua URI de acesso ao banco

No NestJS, você pode usar o Mongoose, uma biblioteca ODM (Object-Document Mapping), para interagir com o MongoDB de maneira mais conveniente. Vamos configurar o Mongoose para se conectar ao MongoDB no arquivo app.module.ts.

3.1. Instale o pacote mongoose usando o seguinte comando:

→ npm install mongoose

Instale os pacotes do Mongoose para fazer o acesso ao MongoDB:

→ npm i --save mongoose @nestjs/mongoose

3.2. Abra o arquivo app.module.ts em seu projeto NestJS e adicione as importações necessárias:

```
import { Module } from '@nestjs/common';
import { MongooseModule } from '@nestjs/mongoose';
```

3.3. No decorador @Module, adicione a configuração do Mongoose usando MongooseModule.forRoot():

```
@Module({
  imports: [
    MongooseModule.forRoot('mongodb://localhost:port/nome-do-banco-de-dados', {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    }),
    // Outros módulos
  ],
  ontrollers: [],
  providers: [],
})
export class AppModule {}
```

Substitua 'mongodb://localhost:port/nome-do-banco-de-dados' pela URL de conexão do seu banco de dados MongoDB. Por exemplo, 'mongodb://localhost:port/mydatabase'.

3.4. Para usar o Mongoose em seus módulos, você pode criar módulos separados para cada entidade e importar MongooseModule.forFeature() para definir os esquemas e modelos de dados. Por exemplo:

```
import { Module } from '@nestjs/common';
import { MongooseModule } from '@nestjs/mongoose';
import { CatsController } from './cats.controller';
import { CatsService } from './cats.service';
import { CatSchema } from './schemas/cat.schema';

@Module({
  imports: [
    MongooseModule.forFeature([{ name: 'Cat', schema: CatSchema }]),
  ],
  controllers: [CatsController],
  providers: [CatsService],
})
export class CatsModule {}
```

Lembre-se de substituir CatSchema pelo seu próprio esquema MongoDB.

Passo 4: Configuração do Front-End com Vue.js

4.1 volte para o diretorio geral do projeto:

→ cd ..

4.2. Instale o Vue CLI globalmente:

→ npm install -g @vue/cli

4.3. Crie um novo projeto Vue.js:

→ vue create frontend

Siga as instruções para configurar o projeto Vue.js de acordo com suas preferências.

Passo 5: Integração entre Back-End e Front-End

5.1. Navegue para a pasta do projeto Vue.js:

→ cd frontend

5.2. Instale o pacote axios para fazer solicitações HTTP ao back-end:

→ npm install axios

5.3. Crie as chamadas de API no Vue.js para se comunicar com o servidor NestJS. Por exemplo, você pode usar o Axios para fazer solicitações HTTP para as rotas do seu servidor.

Passo 6: Execução dos Projetos

6.1. Inicie o servidor NestJS:

→ npm run start:dev

6.2. Inicie o servidor de desenvolvimento Vue.js:

→ npm run serve

Agora, você deve ter o back-end do NestJS rodando em um servidor local e o front-end Vue.js sendo servido em outro. Lembre-se de que esta é uma configuração básica. À medida que você avança no desenvolvimento, pode precisar ajustar configurações e instalar mais dependências de acordo com os requisitos do seu projeto.