Projeto prático de Banco de Dados

Grupo: 1

Autores:

- Gustavo Barbosa de Almeida - 202037589
- Ana Beatriz - 180012428
- Lucas da Silva - 180125699
- Hideki Tomiyama - 190014351
- Thiago Silva Ribeiro - 202037702

CRediT (Contributor Roles Taxonomy):

- **Gustavo** configuração do docker, configuração do backend e documentação dos mesmos, configuração e instalação do frontend.
- **Lucas** criação do repositório, instalação do nvm e do nodejs, e documentação do mesmo.
- **Ana Beatriz** instalação do postgresql e configuração do mesmo.
- **Thiago** criação do modelo de banco de dados e documentação do mesmo.

Data da Versão Atual: 24/10/2023

Sistema de Gerenciamento de Materiais para um Laboratório Didático

Descrição

Para auxiliar os estudantes e professores, o seu grupo ficou encarregado de elaborar um sistema de informação para gerenciar livros de ensino e materiais didáticos em um laboratório. O sistema será projetado para organizar e disponibilizar esses recursos para empréstimo através de um sistema computacional.

O foco principal desta especificação é a definição do banco de dados que será utilizado para armazenar informações sobre os livros e materiais. O sistema deve ter diferentes níveis de acesso para os usuários (por exemplo, administradores do sistema computacional, membros do laboratório e estudantes em geral), de maneira que todos os usuários possam pesquisar os livros e materiais, mas apenas membros do laboratório possam pegar os materiais emprestados.

Tecnologias Utilizadas

NestJS

- O NestJS é um framework de desenvolvimento back-end para Node.js que utiliza TypeScript e segue o padrão arquitetural do Angular. Ele oferece uma estrutura robusta e modular para criar aplicativos escaláveis e bem organizados.

Next.js

- O Next.js é um framework de desenvolvimento front-end para React que simplifica a construção de aplicativos React universais. Ele oferece recursos como renderização do lado do servidor, roteamento simples e pré-renderização, tornando-o adequado para aplicativos da web modernos.

Knex.js

- O Knex.js é um construtor de consultas SQL para Node.js. Ele facilita a interação com bancos de dados relacionais, permitindo a criação de consultas de forma programática e intuitiva. É uma escolha popular para lidar com operações de banco de dados em aplicativos Node.js.

Node.js

- O Node.js é um ambiente de tempo de execução JavaScript que permite que você execute código JavaScript do lado do servidor. Ele é amplamente usado para construir aplicativos de servidor escaláveis e em tempo real, graças ao seu modelo de E/S não bloqueante.

PostgreSQL

- O PostgreSQL é um sistema de gerenciamento de banco de dados relacional de código aberto. Ele é conhecido por sua confiabilidade, recursos avançados e extensibilidade. O PostgreSQL é uma escolha popular para aplicativos que requerem um banco de dados robusto e escalável.

Sistema operacional

O sistema operacional utilizado pela maioria da equipe será linux.

1 Documentação de Configuração de Ambiente e Tecnologias

Esta documentação descreve os passos necessários para configurar o ambiente de desenvolvimento e lista as tecnologias utilizadas neste projeto.

Configuração de Ambiente

1. Instalação do NVM (Node Version Manager) e Node.js (Linux)

Antes de começar, é importante garantir que o sistema esteja atualizado.

```
sudo apt update
sudo apt upgrade
```

1.1. Instalação do NVM

Você pode escolher entre dois métodos para instalar o NVM: usando curl ou wget. Escolha um dos seguintes comandos:

```
## Usando curl
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh |
bash
```

```
## Ou usando wget
wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh |
bash
```

Após a instalação, feche e reabra o terminal. Para verificar a instalação do NVM:

```
nvm --version
```

1.2. Instalação do Node.js

Com o NVM instalado, você pode instalar o Node.js. Recomendamos a instalação da versão LTS mais recente:

```
nvm install --lts
```

Para verificar a versão do Node.js:

```
node --version
```

2. Instalação do PostgreSQL (Sistema de Gerenciamento de Banco de Dados)

Para instalar o PostgreSQL no Linux, execute o seguinte comando:

```
sudo apt install postgresql postgresql-contrib libpq-dev
```

Para verificar a instalação do PostgreSQL:

```
pg_config --version
```

2.1. Guia de Uso do Docker com PostgreSQL (Opcional)

Se preferir usar o Docker com o PostgreSQL, siga as instruções em Guia de Uso do Docker com PostgreSQL para configuração e uso do contêiner PostgreSQL.

Pré-requisitos

- Docker instalado em seu sistema.
- Docker Compose (geralmente incluído com a instalação do Docker).

Configuração do Docker Compose

No diretório do projeto, verifique se existe um arquivo docker-compose.yml. Este arquivo contém as configurações necessárias para criar o contêiner PostgreSQL.

Iniciar o Banco de Dados PostgreSQL

Abra um terminal e navegue até o diretório do projeto onde está o arquivo docker-compose.yml.

- *Para iniciar o contêiner PostgreSQL, execute o seguinte comando:*

```
docker-compose up -d
```

Isso criará e iniciará o contêiner PostgreSQL em segundo plano (-d). Aguarde até que o contêiner esteja em execução.

- *Você pode verificar o status do contêiner com o seguinte comando:*

```
docker ps
```

Certifique-se de que o contêiner PostgreSQL esteja listado na saída.

Conectar-se ao Banco de Dados PostgreSQL

Para se conectar ao banco de dados PostgreSQL a partir do terminal, use o seguinte comando:

```

```

```
psql -h localhost -U postgres -d db
```

- `-h` localhost: Especifica o host onde o PostgreSQL está sendo executado (local).
- `-U` postgres: Especifica o nome de usuário (geralmente é "postgres" por padrão).
- `-d` db: Especifica o nome do banco de dados ao qual você deseja se conectar.
- Será solicitada a senha do usuário "postgres". Insira a senha configurada no arquivo docker-compose.yml (por padrão, é "postgres").

Você estará conectado ao banco de dados PostgreSQL e poderá executar comandos SQL.

Encerrar o Contêiner

Quando você terminar de trabalhar com o banco de dados, você pode parar e remover o contêiner PostgreSQL usando o seguinte comando:

```
docker-compose down
```

Isso desligará e removerá o contêiner PostgreSQL. Certifique-se de que nenhum dado importante seja perdido antes de executar este comando.

Tecnologias Utilizadas

1. Frontend com Next.js

Para executar o frontend do projeto com Next.js, siga os passos abaixo:

Instale as dependências:

```
npm install
```

Inicie o servidor de desenvolvimento:

```
npm run dev
```

2. Backend com NestJS

Para executar o backend do projeto com NestJS, siga os passos abaixo:

Instale as dependências:

```
npm install
```

Inicie a aplicação no modo de desenvolvimento:

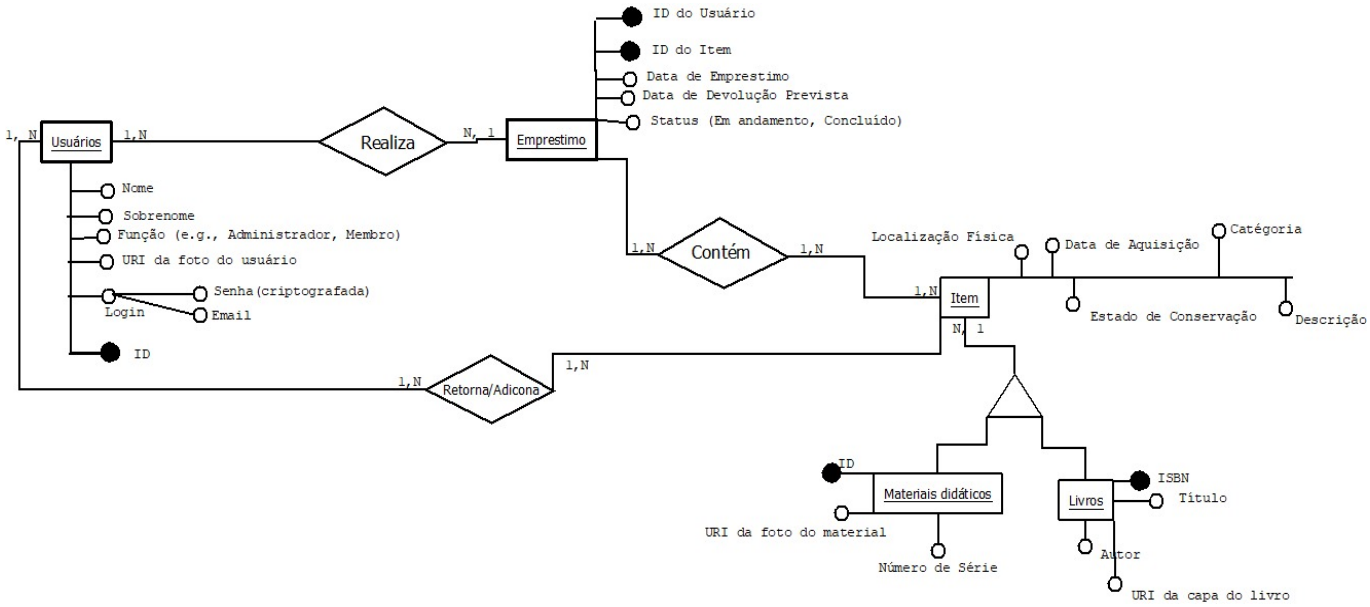
```
npm run start
```

3. Modelo de Banco de Dados

Um Modelo de Banco de Dados é essencial em projetos, definindo a estrutura e organização dos dados, garantindo eficiência, integridade e escalabilidade. É a base para a gestão de informações eficaz.

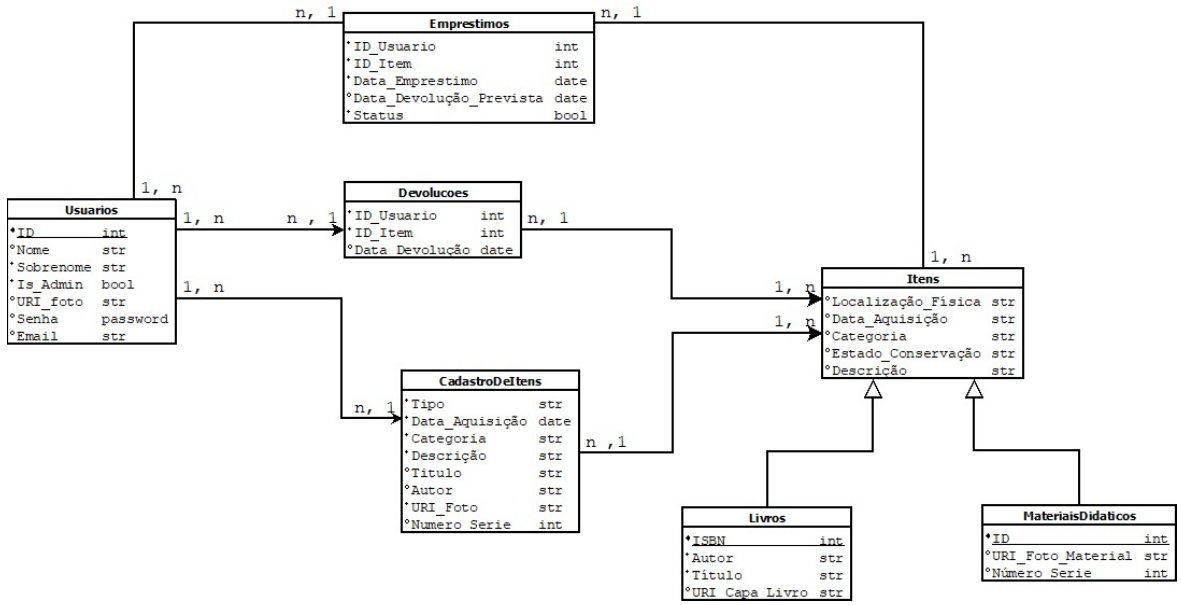
3.1. Diagramas de entidade-relacionamento (DERs)

Diagramas de Entidade-Relacionamento (DERs) são representações visuais que descrevem a estrutura de um banco de dados, mostrando entidades, atributos e relacionamentos entre eles.



3.2 Diagrama do Modelo Lógico (Relacional)

Um Diagrama do Modelo Lógico Relacional é uma representação visual que descreve as tabelas de um banco de dados relacional, seus campos, chaves primárias e chaves estrangeiras. Tabelas representam entidades, campos representam atributos, chaves primárias garantem unicidade e identificação única de registros, e chaves estrangeiras estabelecem relações entre tabelas. Sua importância reside na definição clara da estrutura do banco de dados, permitindo o armazenamento eficiente e a recuperação de informações, garantindo integridade de dados e facilitando o desenvolvimento de consultas e relatórios. Além disso, o modelo lógico serve como guia para a implementação física do banco de dados.


Emprestimos
ID_Usuario int
ID_Item int
Data_Emprestimo date
Data_Devolução_Prevista date
Status bool
Usuarios
ID int
Nome str
Sobrenome str
Is_Admin bool
URI_foto str
Senha password
Email str
Devolucoes
ID_Usuario int
ID_Item int
Data_Devolução date
Itens
Localização_Fisica str
Data_Aquisição str
Categoria str
Estado_Conservação str
Descrição str
CadastroDeItens
Tipo str
Data_Aquisição date
Categoria str
Descrição str
Titulo str
Autor str
URI_Foto str
Numero_Serie int
Livros
ISBN int
Autor str
Título str
URI_Capa_Livro str
MateriaisDidaticos
ID int
URI_Foto_Material str
Número_Serie int
n, 1
1, n
n , 1
n ,1