GRUPO: GUSTAVO HORNING, RICARDO HORNING HAMMERSCHMIDT, NATAN BATALHA DE ARAÚJO.

Relatório de Configuração e Execução

1. Resumo do Projeto

Este projeto é uma aplicação cliente em C# que implementa um CRUD básico para gerenciar informações de clientes em um banco de dados SQL Server. O banco de dados está sendo executado em um contêiner Docker, permitindo uma fácil configuração e isolamento do ambiente. A aplicação cliente se conecta ao banco de dados para realizar operações como listar, adicionar, atualizar e remover clientes.

2. Tecnologias Utilizadas

- **C# (CSharp)**: Linguagem de programação usada para criar a aplicação cliente.
- **SQL Server**: Banco de dados utilizado para armazenar as informações dos clientes.
- **Docker**: Utilizado para executar o SQL Server em um contêiner.
- **Biblioteca Microsoft.Data.SqlClient**: Biblioteca do .NET para se conectar ao SQL Server.

3. Configuração do Ambiente

3.1. Instalação do Docker

- Certifique-se de ter o Docker instalado no seu sistema. Você pode obter a versão mais recente do Docker no site oficial (https://www.docker.com/).

3.2. Configuração do Contêiner Docker com SQL Server

1. **Baixar e Rodar o Contêiner do SQL Server**:
    - Execute o seguinte comando para criar e iniciar o contêiner do SQL Server:

docker run -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Passw@rd' -p 1433:1433 --name sqlserver -d mcr.microsoft.com/mssql/server:2022-latest

    - Parâmetros utilizados:
        - ACCEPT_EULA=Y: Aceita os termos de uso do SQL Server.

- SA_PASSWORD=Passw@rd: Define a senha do usuário administrador (sa).
- -p 1433:1433: Mapeia a porta 1433 do contêiner para a máquina host, permitindo que a aplicação cliente se conecte ao banco de dados.
- --name sqlserver: Define um nome para o contêiner.
- -d mcr.microsoft.com/mssql/server:2022-latest: Define a imagem do SQL Server para ser executada.

2. **Verificar o Status do Contêiner**:
    - Para verificar se o contêiner está rodando corretamente, use:

```
docker ps
```

3.3. Configuração do Banco de Dados e Tabela

1. **Conectar-se ao SQL Server**:
    - Use uma ferramenta de gerenciamento de banco de dados (como Azure Data Studio, SQL Server Management Studio, ou uma linha de comando SQL) para se conectar ao contêiner.
    - Conectar-se ao servidor usando:
        - **Servidor**: localhost,1433
        - **Usuário**: sa
        - **Senha**: Passw@rd
2. **Criar o Banco de Dados e a Tabela**:
    - Execute os seguintes comandos SQL para criar o banco de dados TDE_Performance_DB e a tabela Clientes:

```
CREATE DATABASE TDE_Performance_DB;
GO

USE TDE_Performance_DB;
GO

CREATE TABLE Clientes (
```

```
    Id INT PRIMARY KEY IDENTITY(1,1),
    Nome NVARCHAR(100) NOT NULL,
    Email NVARCHAR(100) NOT NULL
);
GO
```

4. Configuração da Aplicação Cliente

4.1. Código-Fonte do CRUD em C#

- A aplicação foi implementada em C#, usando a estrutura de classes Program, DatabaseConnection, e ClienteCrud.
- A classe Program contém a lógica principal para executar as operações CRUD, enquanto DatabaseConnection gerencia a conexão com o banco de dados, e ClienteCrud implementa as operações específicas (listar, adicionar, atualizar, remover).

4.2. Configuração do Projeto (.csproj)

- O projeto está configurado para ser compilado como um aplicativo .NET executável (OutputType=Exe).
- A versão-alvo do .NET é .NET 8.0.
- O arquivo de configuração inclui a dependência para o pacote Microsoft.Data.SqlClient versão 5.0.1:

```
<ItemGroup>
  <PackageReference Include="Microsoft.Data.SqlClient" Version="5.0.1" />
</ItemGroup>
```

4.3. Conexão ao Banco de Dados

- A conexão ao banco de dados é estabelecida utilizando a classe DatabaseConnection, que recebe uma string de conexão contendo os detalhes para se conectar ao SQL Server rodando no Docker.
- **String de Conexão**:

csharp

Copiar código

String connectionString
="Server=localhost,1433;Database=TDE_Performance_DB;User
Id=sa;Password=Passw@rd;TrustServerCertificate=True;";

- **Server**: localhost,1433 (conecta ao contêiner Docker através da porta exposta).
- **Database**: TDE_Performance_DB.
- **User Id**: sa.
- **Password**: Passw@rd.
- **TrustServerCertificate=True**: Confia no certificado do servidor.

5. Execução da Aplicação

- Para executar a aplicação, navegue até o diretório do projeto e execute o comando:

dotnet run

- **Funcionalidades Demonstradas**:
  - **Listar Clientes**: Exibe todos os clientes cadastrados no banco de dados.
  - **Adicionar Clientes**: Adiciona novos clientes ao banco, passando Nome e Email.
  - **Atualizar Cliente**: Atualiza um cliente específico através de seu Id.
  - **Remover Cliente**: Remove um cliente específico do banco de dados.

6. Mensagens de Log

- **Conexão**: Mensagens como "Conexão bem-sucedida" e "Erro ao conectar ao banco de dados" são exibidas ao tentar abrir a conexão.
- **Operações CRUD**: Cada operação exibe uma mensagem correspondente ao sucesso ou falha da ação, como "Cliente adicionado com sucesso" ou "Cliente removido com sucesso".

7. Considerações Finais

- A aplicação fornece uma maneira prática de gerenciar dados de clientes com um banco de dados SQL Server rodando em um ambiente de contêiner Docker. Isso facilita o isolamento do banco de dados e simplifica o processo de configuração.
- **Boas Práticas**:

- O uso da classe DatabaseConnection permite um gerenciamento claro da conexão ao banco de dados, evitando problemas como conexões não fechadas.
- Recomenda-se melhorar o tratamento de exceções para capturar erros específicos, como falhas ao executar comandos SQL.