dti

Case Técnica

Processo Seletivo Estágio em
Desenvolvimento - Rubix

Objetivos

O objetivo deste teste é avaliar suas habilidades práticas em desenvolvimento de software, especificamente na criação de uma aplicação de console utilizando SQLite como banco de dados.

Você deverá demonstrar sua capacidade de implementar funcionalidades básicas de CRUD (Create, Read, Update, Delete) para um recurso à sua escolha, além de documentar adequadamente seu trabalho.

Orientações

Fique à vontade para escolher a tecnologia que considerar melhor para o projeto como um todo. Só para constar, todas as tecnologias que usamos na DTI hoje são orientadas a objetos.

dti

Desenvolvimento da aplicação

Parte 01

dti

Escolha do Recurso:

Escolha um recurso simples para gerenciar na sua aplicação. O recurso deve ser suficientemente simples para ser representado por apenas uma tabela no banco de dados. Aqui estão alguns exemplos de recursos que você pode considerar:

• Livro • Filme • Música • Petshop (Animal) • Produto • Evento

Definição das Propriedades do Recurso:

Você tem autonomia para definir as propriedades específicas do recurso escolhido. No entanto, certifique-se de incluir:

Campos Obrigatórios: Que devem ser sempre preenchidos.

Campos Opcionais: Que podem ser deixados em branco se desejado.

Tipos de Dados Variados: Inclua pelo menos um campo de texto (string), um campo numérico e um campo de data.

Banco de Dados:

Utilize o SQLite, um banco de dados leve e embutido, para armazenar os dados da aplicação.

Crie um script SQL que defina a estrutura das tabelas necessárias com base no recurso e nas propriedades que você escolheu.

Aplicação Console:

Desenvolva uma aplicação de linha de comando que permita ao usuário interagir com o recurso escolhido.

A aplicação deve implementar as seguintes funcionalidades:

Listar Recursos: Apresentar todos os registros do recurso armazenados no banco de dados, exibindo informações relevantes de forma clara.

Buscar por ID: Permitir que o usuário insira um ID específico para buscar e exibir detalhes de um recurso correspondente.

Cadastrar Recurso: Fornecer uma interface para que o usuário adicione novos recursos ao banco de dados. É essencial que você implemente validações para garantir que os dados inseridos sejam válidos e completos, respeitando a distinção entre campos obrigatórios e opcionais.

Atualizar Recurso: Permitir que o usuário atualize um recurso existente. O usuário deve informar o ID do recurso a ser atualizado e fornecer novamente todas as propriedades do recurso, permitindo alterações conforme necessário.

Deletar Recurso: Oferecer a opção de remover um recurso existente do banco de dados com base no seu ID.

Validação de Entrada:

Durante o processo de cadastro, implemente verificações para assegurar que os dados fornecidos pelo usuário atendam aos critérios esperados. Por exemplo, campos obrigatórios não devem ser deixados em branco, valores numéricos devem estar dentro de intervalos aceitáveis, e campos opcionais podem ser deixados vazios sem causar erros.

Documentação

Parte 02

Documentação da Aplicação:

Elabore um documento abrangente que explique como configurar e executar a aplicação. Este documento deve incluir:

- Explique qual recurso foi escolhido e descreva suas propriedades, especificando quais são obrigatórias, quais são opcionais e os tipos de dados utilizados.
- Linguagem escolhida
- Passos para instalar qualquer dependência necessária.
- Instruções detalhadas sobre como iniciar a aplicação.
- Descrição de como utilizar cada funcionalidade oferecida pela aplicação, incluindo exemplos de uso quando apropriado.

Diferenciais (Opcional)

Testes Unitários:

Escreva testes unitários para validar as principais funcionalidades da aplicação. Isso demonstra sua habilidade em garantir a qualidade e a robustez do código.

Conteinerização da Aplicação:

Utilize Docker para conteinerizar a aplicação, permitindo que ela seja executada em um ambiente isolado e padronizado. Isso facilita a distribuição e execução da aplicação em diferentes ambientes.

Logs:

Implemente um sistema de logs para registrar operações importantes e possíveis erros que ocorram durante a execução da Entrega dos Artefatos

Entregável:

Ao final do teste, o candidato deve enviar os seguintes artefatos:

Código Fonte: Todo o código fonte da aplicação, organizado em pastas e arquivos conforme necessário.

Script SQL: O script utilizado para criar as tabelas no banco de dados SQLite.

Documentação: Um documento detalhado contendo instruções de configuração, execução e uso da aplicação, além da descrição do recurso e suas propriedades.

Testes Unitários (se aplicável): Arquivos de teste, caso tenha optado por implementar testes unitários.

Dockerfile e Configurações (se aplicável): Arquivos relacionados à conteinerização da aplicação, caso tenha optado por este diferencial.

Arquivos de Log (se aplicável):

Exemplos de logs gerados pela aplicação, caso tenha implementado logging. Isso é útil para depuração e monitoramento.

Estes artefatos devem ser enviados em um formato compactado (por exemplo, .zip ou .tar.gz) ou através de um repositório de controle de versão, como GitHub, garantindo que todas as partes do projeto estejam acessíveis para avaliação da aplicação

Observação Importante:

Caso o seu código não esteja funcionando perfeitamente ou encontre algum erro durante o desenvolvimento, não desanime. Você ainda pode enviar seu projeto. É importante incluir uma explicação clara sobre o erro que está ocorrendo. Saber identificar e explicar problemas é uma habilidade valiosa para um desenvolvedor.

Prazo

Você vai ter um prazo de 4 dias corridos para completar o teste e enviar suas soluções. Certifique-se de gerenciar bem o tempo para cobrir todos os requisitos obrigatórios e, se possível, explorar os diferenciais opcionais

Obrigado.

Equipe de Pessoas Rubix

dti · enterprise
rubix