O Desafio a seguir tem como objetivo avaliar a capacidade do candidato de:

1. Utilizar conceitos fundamentais de **orientação a objetos**;
2. Manipular banco de dados relacional.
3. Desenvolver uma aplicação simples em **Java** que funcione como backend para operações CRUD básicas.
4. Desenvolver uma interface gráfica para interagir com o backend.
5. Demonstrar boas práticas de codificação e organização de código.
6. Utilizar **Git** para versionamento de código.

#### **Descrição do Desafio**

A empresa XWZ precisa de um sistema simples para **gerenciar veículos** de uma frota, classificando-os em duas categorias: **carros** e **motos**. Os veículos possuem informações comuns, mas também características específicas de cada tipo.

O candidato deverá implementar uma aplicação que permita:

1. **Cadastrar veículos**.
2. **Consultar veículos pelo tipo (adicional, filtrar também por modelo, cor e ano de fabricação)**.
3. **Excluir um veículo pelo seu ID**.
4. **Atualizar informações de um veículo existente**.
5. **Listar todos os veículos cadastrados**.

### **Regras de Negócio**

1. Todo veículo deve ter os seguintes atributos:
   1. id (inteiro, auto-gerado)
   2. modelo (obrigatório)
   3. fabricante (obrigatório)
   4. ano (obrigatório)
   5. preco (obrigatório)
2. Os **carros** devem ter, adicionalmente:
   1. QuantidadePortas (obrigatório)
   2. tipoCombustivel (obrigatório – gasolina, etanol, diesel, flex)
3. As **motos** devem ter, adicionalmente:
   1. cilindrada (obrigatório)
4. O sistema deve ser capaz de persistir e recuperar os dados utilizando **JDBC** para manipular diretamente um banco de dados relacional (sugere-se MySQL ou PostgreSQL).

### **Requisitos**

1. **Banco de Dados**
   1. O candidato deve criar as tabelas indicadas no contexto do desafio, garantindo integridade referencial e normalização.
   2. Utilizar *Native query*
   3. O esquema SQL deve ser entregue junto com o projeto (scripts SQL para criação das tabelas e inserção de dados iniciais).
2. **Backend**
   1. Desenvolvido em **Java**.
   2. A aplicação deve ser um projeto **Maven**.
   3. Seguir boas práticas de orientação a objetos.
3. **Interface Gráfica**
4. A interface pode ser desenvolvida em qualquer tecnologia/framework escolhido pelo candidato, porém sugere-se o desenvolvido com HTML, CSS e JavaScript.
5. A interface deve permitir ao usuário realizar todas as operações CRUD;
6. Recomenda-se que a interface do sistema exiba a lista de veículos em uma tabela interativa, permitindo ao usuário selecionar um registro para as seguintes ações:
   1. **Edição**: Modificar/atualizar os dados do veículo selecionado.
   2. **Detalhamento**: Visualizar informações completas do registro.
   3. **Exclusão**: Remover o veículo do sistema.

A partir dessa interface principal, o usuário também pode ter acesso à funcionalidade de inclusão de novos veículos. Além disso, a interface de consulta deve oferecer filtros para que o usuário possa refinar a busca por tipo, cor, modelo e ano de fabricação dos veículos, facilitando a localização de registros específicos.

1. **Documentação**

Fornecer um README com instruções para:

* 1. Configurar o banco de dados.
  2. Executar a aplicação.

1. **Git**
   1. O código deve ser versionado e enviado em um repositório Git.
   2. É interessante que o texto de cada commit efetuado represente exatamente o que foi implementado;
   3. Manter histórico de commits.

### **Entrega**

* O candidato deve enviar o link do repositório Git contendo o código-fonte, o esquema SQL e o README com as instruções.
* Prazo para entrega: até **27/05/2025 (**não serão considerados as entregas fora do prazo);