### **Desafio de Programação - Desenvolvedor Java**

**Objetivo**: Avaliar suas habilidades em desenvolvimento Java com foco em design patterns, manipulação de dependências externas e escrita de testes unitários.

#### **Descrição do Desafio**

Implemente uma aplicação Java que simule um sistema de gerenciamento de produtos com funcionalidades básicas de CRUD (Create, Read, Update, Delete). Este sistema deve utilizar o padrão de projeto **Singleton** para a classe de serviço e incluir um repositório para gerenciar os dados dos produtos.

#### **Requisitos Técnicos**

* **Java**: Utilize Java SE 11.
* **Maven**: O projeto deve estar configurado para utilizar o Maven, mas sem o uso de dependências externas no pom.xml (os JARs necessários serão adicionados manualmente).
* **Mockito e JUnit 4**: Utilize Mockito para mockar o repositório nos testes e JUnit 4 para criar e rodar os testes unitários. Certifique-se de que o Mockito está na versão 3.6, e que ambos os JARs estão adicionados ao projeto.

#### **Funcionalidades**

Implemente as seguintes operações no sistema de gerenciamento de produtos:

1. **Adicionar Produto**: Método para adicionar um novo produto com ID, nome e preço.
2. **Buscar Produto por ID**: Método para retornar um produto com base no seu ID.
3. **Atualizar Produto**: Método para atualizar o nome e preço de um produto existente.
4. **Deletar Produto**: Método para remover um produto com base no ID.
5. **Listar Produtos**: Método para listar todos os produtos.

#### **Estrutura da Aplicação**

1. **Produto**: Classe que representa um produto, com id, nome, e preco como atributos.
2. **ProdutoService**: Classe de serviço responsável pelo gerenciamento dos produtos. Esta classe deve ser implementada como um **Singleton** e utilizar um ProdutoRepository para manipular os dados.
3. **ProdutoRepository**: Interface para operações básicas de CRUD.
4. **ProdutoRepositoryImpl**: Implementação da interface ProdutoRepository, que irá armazenar os dados dos produtos em uma estrutura de dados simples, como uma lista.

#### **Testes Unitários**

Implemente testes unitários para cada operação do ProdutoService usando JUnit 4 e Mockito. Os testes devem verificar o seguinte:

* A adição de um produto chama o método salvar no repositório.
* A busca por ID retorna o produto correto.
* A atualização de um produto existente modifica o nome e o preço corretamente.
* A tentativa de atualizar um produto inexistente não causa erros.
* A exclusão de um produto chama o método deletar no repositório.
* A listagem de produtos retorna todos os produtos adicionados.

**Nota**: Nos testes, utilize Mockito para mockar o ProdutoRepository e validar que os métodos corretos estão sendo chamados.

#### **Dicas para Configuração no IntelliJ**

* Para adicionar manualmente os JARs de dependências do Mockito e JUnit, vá para File > Project Structure > Modules > Dependencies, e adicione os arquivos .jar manualmente.

#### **Entrega**

* Envie o código-fonte do projeto em um repositório Git ou arquivo ZIP, com instruções de como rodar a aplicação e os testes.
* Inclua um arquivo README.md explicando as decisões de implementação.