# DESAFIO DTI

## Ana Cláudia Gomes de Mendonça

### Tecnologias Utilizadas

- Java 8;
- Angular 8;
- Spring Boot;
- PostgreSQL

### Decisões de Projeto

Considerando que foi dado um prazo para a entrega do projeto, é comum que ocorram imprevistos, e, com eles, os atrasos. Visando manter uma relação próxima e transparente com o cliente – afinal, ele é o maior interessado que o projeto fique dentro do combinado – proponho melhorias ao que foi pedido, de modo a compensar o atraso e adicionar valor à entrega.

Um usuário que deseja manter um bom controle sobre seu estoque precisa saber:

- O que ele tem atualmente em estoque;
- O que ele vendeu, e quando;
- O que entrou no estoque, e quando.

Futuramente, o usuário pode querer funcionalidades para avisá-lo quando o estoque está chegando ao fim, ou pode querer ter um controle sobre a época do ano em que determinado produto é mais vendido. A partir dos dados atuais, ele também pode saber quanto tem, quanto gastou, quanto pode lucrar, etc.

Pensando nisso, foram criadas três tabelas:

- Produtos;
- Histórico de vendas;
- Histórico de cadastros.

Devido aos registros em históricos, cada exclusão de produto apenas o coloca como indisponível e registra o cadastro de “zero” na tabela de histórico.

Por se tratar de uma prova que visa analisar todos os aspectos do código desenvolvido, o deploy não foi priorizado, e a execução foi mantida diretamente nas IDEs utilizadas no desenvolvimento.

### Como Executar

- Clonar o projeto;
- Na IDE de Java (foi utilizado o Eclipse), abrir a classe “DtiApplication.java”. Clicar com o botão direito sobre o código, “Run As”, “Java Application”.

- Com o terminal aberto na pasta “desafio-dti-angular”, node e npm devidamente instalados e atualizados, executar o comando “npm start”.
- Abrir uma aba no navegador “localhost:4200”.

**Possíveis Melhorias e Alterações**

- Usar bibliotecas que facilitem o desenvolvimento, como bibliotecas Java;
- Utilizar apenas uma tabela para histórico caso o usuário deseje manter as mesmas informações sobre cada tipo de dado. Por exemplo, a tabela de histórico de vendas pode ter relação com uma outra tabela de clientes para fazer um controle do cliente que efetuou a compra, e assim seria interessante haver tabelas separadas;
- Cadastro de clientes e/ou de funcionários que têm acesso ao estoque;
- Cadastro de unidades da loja e/ou de depósitos, para o caso de transferência de produtos de uma unidade para outra, e novas colunas nas tabelas existentes para sinalizar se a operação foi de venda ou de transferência para outra unidade da loja.