# Fundamentos de Desenvolvimento de Software

# Trabalho Final – 2021/2

# Sistema de Vendas utilizando Arquitetura CLEAN

## Introdução

A pandemia provocou um salto na participação das vendas online no faturamento das empresas do comércio varejista brasileiro. Antes da crise sanitária que obrigou o fechamento das lojas físicas, o e-commerce representava, em média, 9,2% da receita. Mas, em julho do ano passado, com apenas quatro meses de pandemia, essa marca mais do que dobrou e foi para 19,8%. E, em junho deste ano, já estava em 21,2%. (<https://www.em.com.br/app/noticia/economia/2021/10/16/internas_economia,1314417/com-pandemia-comercio-online-mais-que-dobra-e-ja-chega-a-21-das-vendas.shtml>)

A picture containing text, keyboard, indoor, typewriter

Description automatically generated

Neste contexto, uma rede lojas tomou a decisão de desenvolver um módulo de vendas online. Nele o usuário seleciona produtos a partir de uma lista e indica a quantidade desejada. O Sistema verifica a disponibilidade (o produto só é inserido na venda se houver disponibilidade), calcula o custo básico, impostos e o total a pagar. Se o usuário efetivar a compra é feito registro da venda e a baixa no estoque. O registro da venda inclui o número da venda, a relação dos itens vendidos (código do produto, preço unitário, quantidade vendida e preço cobrado pelo item), subtotal da venda, impostos e custo final ao consumidor.

O sistema é composto por dois módulos:

1. um módulo *frontend* responsável apenas pela interação com o usuário (*frontend thin*). Este será aproveitado de uma outra loja do mesmo grupo (ver figura 1). Este módulo foi implementado usando HTML/CSS/JavaScript usando o padrão MVC em uma arquitetura em camadas. Todo o conteúdo a ser exibido é solicitado para o módulo *backend*.
2. Um módulo *backend* capaz de atender as requisições do módulo *frontend* dentro da lógica de negócios atual da loja. Para este módulo foi construído um protótipo simples que visa apenas mostrar a viabilidade do sistema. Agora é necessário construir o módulo *backend* propriamente dito.

|  |
| --- |
| Figura - Tela do sistema de frontend |

## Requisitos do sistema

A versão atual do sistema não terá autenticação. Quando a página é aberta o sistema já se encontra pronto para a criação de uma venda. O usuário indica os produtos desejados e a quantidade de cada um. Sempre que um produto é inserido na venda o sistema verifica antes a disponibilidade do produto exibindo mensagem caso não exista disponibilidade. À medida que os produtos vão sendo inseridos o subtotal, os impostos e o custo final vão sendo atualizados. Quando o usuário confirma a venda (supõem-se que o pagamento é feito em outro sistema e que os usuários são honestos) o sistema faz a baixa dos produtos vendidos no estoque, registra a venda e já deixa a tela pronta para a próxima venda. Se a venda for cancelada então apenas a tela é limpa.

A rede de lojas opera em diferentes países. Para cada país o sistema é instanciado em um servidor diferente. Em cada país as regras relativas aos impostos variam. Por exemplo, no país “Um” cobram-se 12% de IVA. Já no país “Dois” o IVA é de 15% para compras até $8000,00 e de 20% para o que exceder valores maiores. Em outros países o imposto pode variar de acordo com a categoria do produto (alcoólicos, alimentos, bazar, eletrônicos etc.).

Por enquanto o controle de estoque deve manter a quantidade em estoque e ter operações para a entrada e saída de produtos. Em um futuro próximo sabe-se que novas informações serão agregadas tais como estoque mínimo, custo operacional e outras que ainda estão sendo planejadas.

Outro aspecto importante é que podem ser impostos limites para a quantidade de produtos que podem compor uma venda ou para a quantidade de cada item ou ainda para o valor total da venda. Estas restrições normalmente ocorrem em função do horário (vendas quando as lojas físicas estão abertas tem menos restrições). Quando o limite do horário for excedido uma mensagem é exibida e o produto não é acrescentado à venda.

Finalmente, o sistema deve prever um módulo gerencial que permita à gestão um acompanhamento do sistema – valor do estoque, valor total de vendas, relatório de vendas por categoria.

Como o módulo *frontend* é destinado a clientes, ele não permite consultar o histórico de vendas nem o módulo gerencial, mas este devem ser implementados o sistema deve possuir *endpoints* que permitam esta consulta através de um software tipo *postman* ou *insomnia*.

## Interface de comunicação com o módulo frontend

O módulo *frontend* comunica-se com o módulo *backend* através de requisições HTML. Embora não tenhamos a especificação destas requisições, conhecemos os *endpoints* expostos pelo protótipo. Os parâmetros de entrada e o retorno destes *endpoints* devem ser mantidos, pois é através deles que o *frontend* se comunica. O *frontend* não será alterado neste momento, mas o *backend* deve ser refatorado para atender os requisitos apresentados, de acordo com o modelo de arquitetura limpa estudada.

## Instruções para o desenvolvimento do trabalho

O objetivo do trabalho é evoluir o protótipo do módulo *backend* para uma versão capaz de atender todos os requisitos descritos acima atendendo os princípios SOLID e a arquitetura CLEAN. Padrões de projeto deverão ser explorados sempre que adequado. O módulo deve ser escrito em Java usando a tecnologia Spring. A persistência dos dados deverá ser feita usando JPA. A escolha do SGBD é livre podendo ser usado o H2 em memória.

O trabalho deverá ser desenvolvido em etapas:

1. Instalação e compreensão do protótipo
2. Modelagem da solução segundo a arquitetura CLEAN
3. Implementação do *backend*
4. Implementação de drivers de teste para pelo menos uma das classes do sistema que contenham a lógica de negócio;
5. Elaboração de relatório contendo:
   1. Diagrama de classes da solução
   2. Detalhamento dos padrões de projeto utilizados e o objetivo de cada um
6. Apresentação do relatório e da respectiva solução funcionando de maneira integrada com o módulo *frontend* fornecido.

## Entregas

O trabalho pode ser desenvolvido individualmente ou em grupos de até 4 alunos. O projeto deve ser mantido, até a apresentação, em um repositório privativo e o professor convidado como colaborador. O uso adequado do *git* (*commits* e uso de *branches*) será considerada na avaliação do trabalho, bem como a efetiva participação de todos os integrantes do grupo.

Os entregáveis que não correspondem a código fonte devem ser armazenados em uma pasta chamada “OutrosEntregaveis” a ser criada na raiz do projeto no “GitHub”. Para as entregas/apresentações intermediárias é necessária a presença de apenas um dos integrantes do grupo no momento síncrono. Na data da apresentação final todos os integrantes devem estar *presentes*. Caso algum grupo perca alguma entrega/apresentação intermediária deverá contatar o professor para agendar horário alternativo. Os atrasos terão impacto na avaliação final.

O cronograma das entregas parciais será discutido em aula e o cronograma postado no Moodle. Agende com o professor a data da sua apresentação.