UNIVERSIDADE FEDERAL DE SÃO PAULO

Lucas Miranda Giannella - 159324
Pedro Henrique Cavalcante - 163968
Gabriela Dobbert Sanches - 163760
Giovanna Szabloczky Venancio - 156535

RELATÓRIO DE IMPLEMENTAÇÃO: SISTEMA DE PONTO DE VENDA (PDV) PARA MERCADINHO

São José dos Campos, 2025

SUMÁRIO

1 INTRODUÇÃO

Este relatório detalha o desenvolvimento de um sistema de Ponto de Venda (PDV) para um mercadinho, utilizando Python e Streamlit para a interface e um banco de dados NoSQL (TinyDB) para o armazenamento de dados. O projeto visa demonstrar a implementação de um sistema transacional simples e eficaz para gerenciar vendas, estoque e produtos, com uma interface amigável para o usuário.

2 OBJETIVOS

O principal objetivo do projeto foi criar uma plataforma funcional de PDV que permitisse:

- **Gestão de Produtos:** Adicionar, visualizar, editar e remover produtos, incluindo nome, preço e quantidade em estoque.
- **Processamento de Vendas:** Simular um processo de venda, adicionando produtos a um carrinho, calculando o subtotal e o total da compra, e registrando a venda.
- **Controle de Estoque:** Atualizar automaticamente o estoque dos produtos após cada venda.
- **Histórico de Vendas:** Manter um registro detalhado de todas as vendas realizadas.
- **Interface Intuitiva:** Desenvolver uma interface de usuário (UI) clara e fácil de usar para as operações diárias do mercadinho.

A escolha das tecnologias foi fundamentada na agilidade de desenvolvimento e na simplicidade para um projeto de demonstração.

3 METODOLOGIA EMPREGADA

A metodologia de desenvolvimento adotada seguiu uma abordagem iterativa, focando na construção modular do sistema e na integração progressiva das funcionalidades.

3.1 Tecnologias Utilizadas

- **Python:** Linguagem de programação principal para a lógica de backend e a interface.
- **Streamlit:** Framework para a criação rápida e interativa da interface web.
- **TinyDB:** Um banco de dados NoSQL leve, baseado em documentos, utilizado para armazenar os dados de produtos e vendas de forma persistente. A escolha do TinyDB se deu pela sua simplicidade e facilidade de integração com Python para um projeto de menor escala, evitando a complexidade de configurar SGBDs mais robustos para este protótipo.
- **Pandas:** Biblioteca para manipulação e análise de dados, utilizada para exibir produtos e vendas em formato tabular na interface.

3.2 Estrutura do Projeto

O projeto é composto por dois arquivos principais:

- `app.py`: Contém a lógica da aplicação Streamlit, definindo as diferentes páginas (PDV, Estoque, Adicionar Produto, DB Control) e a interação com o repositório de dados.
- `database.py`: Contém a classe `MercadoRepositorio`, que encapsula as operações de banco de dados (TinyDB) para produtos e vendas.

3.3 Fases de Desenvolvimento

1. **Modelagem de Dados:**
    - **Produtos:** Cada produto é representado por um dicionário contendo `nome`, `preco`, e `estoque`.
    - **Vendas:** Cada venda é registrada como um dicionário contendo `data_venda`, uma lista de `itens` (cada item com `produto_id`, `nome_produto`, `quantidade`, `preco_unitario`, `subtotal`), e o `total_venda`.
2. **Implementação do Repositório de Dados (`database.py`):**
    - Criada a classe `MercadoRepositorio` para abstrair as operações de leitura, escrita, atualização e exclusão no TinyDB.
    - Funções para listar produtos e vendas, adicionar produtos, atualizar estoque e registrar vendas foram implementadas.
3. **Desenvolvimento da Interface (`app.py`):**
    - **Ponto de Venda (PDV):** Interface para adicionar produtos ao carrinho, ajustar quantidades e finalizar a compra.
    - **Produtos e Estoque:** Página para visualizar todos os produtos cadastrados, com filtros de preço e estoque disponível.
    - **Adicionar Produto:** Formulário para cadastrar novos produtos no sistema.
    - **DB Control:** Seção administrativa para editar produtos diretamente em uma tabela editável e visualizar o histórico de vendas.
    - Utilização do `st.session_state` do Streamlit para gerenciar o estado do carrinho de compras e o total da venda entre as interações do usuário.

4 RESULTADOS E DISCUSSÃO

O sistema de PDV foi implementado com sucesso, oferecendo as funcionalidades essenciais para um mercadinho. A interface Streamlit provou ser eficaz para o rápido desenvolvimento e prototipagem, permitindo uma interação intuitiva com o usuário.

4.1 Funcionalidades Detalhadas

- **Ponto de Venda:**
    - Os produtos disponíveis são listados em um `selectbox`, facilitando a seleção.

    - O estoque disponível de cada produto é exibido, e a quantidade a ser adicionada ao carrinho é limitada pelo estoque.
    - O carrinho de compras é atualizado dinamicamente, mostrando os itens, quantidades e subtotais.
    - É possível remover itens do carrinho individualmente.
    - O total da venda é calculado em tempo real.
    - Ao finalizar a compra, o estoque é automaticamente decrementado e a venda é registrada.
- **Produtos e Estoque:**
    - Todos os produtos são exibidos em uma tabela interativa.
    - Filtros por faixa de preço e opção de mostrar apenas produtos com estoque permitem refinar a visualização.
- **Adicionar Produto:**
    - Um formulário simples permite inserir o nome, preço e estoque inicial de novos produtos.
- **DB Control:**
    - A funcionalidade de edição de produtos diretamente na tabela é poderosa para a gestão.
    - A exclusão de produtos é tratada pela identificação de IDs ausentes na tabela editada.
    - O histórico de vendas é apresentado de forma organizada, com a data formatada e os itens da venda detalhados em um `expander`.

4.2 Exemplos de Operação

A seguir, são demonstrados alguns exemplos de interações com o sistema:

- **Cadastro de Produto:** Na página "Adicionar Novo Produto", um usuário insere "Coca Cola 2L", "9.50" e "50" nos campos e clica em "Cadastrar Produto". Uma mensagem de sucesso "Produto 'Coca Cola 2L' adicionado com sucesso!" é exibida.
- **Adicionar ao Carrinho:** Na página "Ponto de Venda", o usuário seleciona "Coca Cola 2L" e informa a quantidade "2". Ao clicar em "Adicionar ao Carrinho", o item é adicionado e o total da venda é atualizado.
- **Finalizar Compra:** No carrinho, com alguns itens adicionados, o usuário clica em "Finalizar Compra". Uma mensagem de sucesso "Venda registrada com sucesso!" é exibida, o carrinho é esvaziado, e o estoque dos produtos vendidos é atualizado.
- **Visualizar Produtos com Estoque:** Na página "Produtos e Estoque", o checkbox "Mostrar apenas produtos com estoque" está marcado por padrão, exibindo apenas os produtos disponíveis para venda.
- **Visualizar Histórico de Vendas:** Na página "DB Control", a seção "Histórico de Vendas" exibe todas as vendas registradas, com a data e o total. Clicar no `expander` de uma venda revela os produtos comprados, suas quantidades, preços unitários e subtotais.

5 VANTAGENS E DESVANTAGENS DA ABORDAGEM

5.1 Vantagens do Uso do TinyDB e Streamlit

- **Facilidade de Uso e Configuração:** TinyDB é extremamente leve e não requer um servidor separado, facilitando a configuração para projetos pequenos e protótipos. Streamlit permite a criação de interfaces de usuário com poucas linhas de código.
- **Rapidez no Desenvolvimento:** A combinação de Python, Streamlit e TinyDB acelera significativamente o ciclo de desenvolvimento, ideal para demonstrações e MVPs.
- **Modelagem Flexível:** Como um banco de dados de documentos, o TinyDB oferece flexibilidade na modelagem de dados, adaptando-se facilmente a mudanças de esquema.
- **Persistência Simples:** Embora leve, o TinyDB garante a persistência dos dados em um arquivo JSON.

5.2 Desvantagens da Abordagem

- **Escalabilidade Limitada:** TinyDB não é adequado para aplicações com alto volume de dados ou alta concorrência. Para um sistema de PDV de grande escala, um SGBD mais robusto (como PostgreSQL, MongoDB ou até mesmo Redis para caching de alta performance) seria necessário.
- **Recursos de Consulta Limitados:** As capacidades de consulta do TinyDB são mais básicas em comparação com bancos de dados relacionais ou NoSQL com linguagens de consulta mais ricas.
- **Gerenciamento de Transações:** Para transações complexas que exigem alta consistência, o TinyDB pode não ser a melhor escolha.
- **Integração e Ecossistema:** Embora funcional, a integração com outros sistemas empresariais pode ser mais desafiadora em comparação com soluções mais maduras.

6 CONCLUSÕES

A implementação do sistema de Ponto de Venda utilizando Python, Streamlit e TinyDB atingiu seus objetivos de demonstrar um sistema funcional para a gestão de vendas e estoque em um mercadinho. A abordagem escolhida se mostrou eficaz para o desenvolvimento rápido e a criação de uma interface de usuário intuitiva.

Para projetos de maior escala ou com requisitos de alta performance e concorrência, a migração para um SGBD mais robusto e a adoção de frameworks web mais completos seriam consideradas. No entanto, para fins de prototipagem e aprendizado, esta combinação de tecnologias oferece uma excelente base.

7 DIFICULDADES ENCONTRADAS

Tentamos com Cassandra, tentamos com NodeJS, juntamos com Blink - marca de roupa do Pedro Henrique -, adaptamos o banco da tabela estoque para juntar com a tabela item, tentamos integrar pagamento mercado livre mas é muito complexo comparado com o nosso sistema que tem foco na simplicidade. Após isso, tentamos utilizar o Stripe deu problema com o pagamento pois o mesmo token estava em uso pelo Pedro na BLink.

As principais dificuldades foram menores e mais relacionadas a detalhes de implementação no Streamlit, como a necessidade de gerenciar o estado da sessão para o

carrinho de compras e a atualização dinâmica da interface. A persistência de dados com TinyDB foi bastante direta, mas o entendimento de como as edições no `st.data_editor` se traduzem em operações de `update` e `delete` no banco de dados exigiu um cuidado extra.

8 DEZ QUERIES RELEVANTES

1. Esta consulta é útil para saber quais itens precisam ser repostos.
```
def produtos_com_baixo_estoque(self):
        Produto = Query()
        return self.tabela_produtos.search(Produto.estoque < 10)
```

2. Filtro por preço, definindo mínimo e máximo.
(Implementado num slider no app)

3. Buscar produtos por uma palavra-chave no nome (ex: "Coca Cola")
```
def buscar_produto_por_palavra_chave(self, keyword):
        Produto = Query()
        self.tabela_produtos.search(Produto.nome.search(keyword,
flags=re.IGNORECASE))
```

4. Encontrar todas as vendas que ultrapassaram um determinado valor
```
def vendas_acima_de_valor(self, valor):
        Venda = Query()
        return self.tabela_vendas.search(Venda.total_venda > valor)
```

5. Listar todas as vendas que contêm um produto específico
```
def vendas_contendo_produto(self, nome_produto):
        Venda = Query()
        return
self.tabela_vendas.search(Venda.itens.any(Query().nome_produto ==
nome_produto))
```

6. Encontrar produtos que estão completamente sem estoque
```
def produtos_esgotados(self):
        Produto = Query()
        return self.tabela_produtos.search(Produto.estoque == 0)
```

7. Listar todos os produtos que são "Cigarros" ou "Bebidas" (Exemplo, pode ser alterado)

```
def produtos_por_categoria_regex(self, padrao):
        Produto = Query()
        return
self.tabela_produtos.search(Produto.nome.matches(padrao,
flags=re.IGNORECASE))
```

8. Contar quantas vendas foram realizadas

```
def contar_total_de_vendas(self):
        return len(self.tabela_vendas)
```

9. Encontrar vendas realizadas em uma data específica

```
def vendas_por_data(self, data_iso):
        Venda = Query()
        return self.tabela_vendas.search(Venda.data_venda.test(lambda
d: d.startswith(data_iso)))
```

10. Encontrar produtos onde o nome tenha exatamente duas palavras

```
def produtos_com_nome_duplo(self):
        Produto = Query()
        return self.tabela_produtos.search(Produto.nome.test(lambda
nome: len(nome.split()) == 2))
```

REFERÊNCIAS

MUSA, Daniela Leal. **Material da disciplina "Aspectos de Implementação de Banco de Dados"**. São José dos Campos: UNIFESP, 2025.

PANDAS. **Documentação oficial do Pandas**. Disponível em: https://pandas.pydata.org/docs/. Acesso em: 21 jul. 2025.

STREAMLIT. **Documentação oficial do Streamlit**. Disponível em: https://docs.streamlit.io/. Acesso em: 21 jul. 2025.

TINYDB. **Documentação oficial do TinyDB**. Disponível em: https://tinydb.readthedocs.io/. Acesso em: 21 jul. 2025.