Relatório Acadêmico: Desenvolvimento de uma Plataforma de E-commerce

Alunos: Paulo Willian Costa Rodrigues, Rafael Pereira Botelho

Professor: Jeffson Celeiro Souza

Data: 24/11/2025

Trabalho: M.W Calçados (e-comerce de calçados)

Disciplina: Desenvolvimento Web em Html5, Css, JavaScript e Php

Resumo

- Este relatório apresenta o desenvolvimento de uma plataforma de e-commerce completa, focada na implementação de um sistema de carrinho de compras persistente no banco de dados. O projeto integra tecnologias de backend em PHP com MySQL e frontend em JavaScript, utilizando uma arquitetura RESTful para comunicação. As funcionalidades incluem autenticação de usuários, navegação de produtos, gerenciamento de carrinho e validações de segurança. O sistema foi testado e validado, demonstrando funcionalidade em cenários reais de uso. O relatório detalha a metodologia, implementação, resultados e limitações, oferecendo insights para futuras expansões.
- **Palavras-chave:** E-commerce, Carrinho Persistente, API RESTful, PHP, JavaScript, MySQL.

1. Introdução

1.1 Contexto do Projeto

- O e-commerce representa um segmento crescente da economia digital, com plataformas como Amazon e Mercado Livre exemplificando a importância de sistemas robustos para transações online (Laudon & Traver, 2018). Este projeto visa desenvolver uma plataforma básica de e-commerce, enfatizando a persistência de dados do carrinho de compras, um desafio comum em aplicações web onde a sessão do usuário deve ser mantida mesmo após fechamento do navegador.

1.2 Objetivos

- Implementar um sistema de carrinho de compras que armazene itens no banco de

dados, associado ao usuário logado.

- Desenvolver autenticação segura via tokens JWT para proteger operações sensíveis.
- Criar uma interface frontend responsiva para navegação de produtos e gerenciamento de carrinho.
- Garantir validações de segurança, como verificação de estoque e prevenção de acessos não autorizados.

1.3 Escopo

- O projeto abrange o desenvolvimento completo, desde o banco de dados até a interface do usuário, excluindo funcionalidades avançadas como pagamento online ou integração com gateways externos. O foco é na arquitetura modular e na integração frontend-backend.

2. Revisão de Literatura

2.1 Conceitos de E-commerce

- O e-commerce envolve a compra e venda de produtos via internet, com componentes como catálogo de produtos, carrinho de compras e checkout (Chaffey, 2015). A persistência de carrinho é crucial para melhorar a experiência do usuário, reduzindo abandono de compras (Baymard Institute, 2020).

2.2 Tecnologias Utilizadas

- **Backend:** PHP com PDO para acesso a banco de dados MySQL, seguindo padrões RESTful (Richardson & Ruby, 2007).
- **Frontend:** JavaScript ES6 com módulos, HTML5 e CSS3 para interfaces dinâmicas.
- **Banco de Dados:** MySQL para armazenamento relacional, com normalização para evitar redundâncias.
- **Segurança:** Tokens JWT para autenticação *stateless*, prevenindo sessões vulneráveis (Jones et al., 2015).

3. Metodologia

3.1 Abordagem de Desenvolvimento

- O projeto seguiu uma metodologia ágil, com iterações incrementais:
- **Planejamento:** Definição de requisitos e arquitetura.

- **Implementação:** Desenvolvimento modular (backend primeiro, seguido de frontend).
- **Testes:** Validação unitária via Postman e testes manuais no navegador.
- **Refinamento:** Correção de bugs e otimização baseada em feedback.

3.2 Ferramentas e Ambiente

- **Servidor Local:** XAMPP para execução de PHP e MySQL.
- **Editor:** Visual Studio Code para desenvolvimento.
- **Testes:** Postman para API; console do navegador para JS.
- **Versionamento:** Git para controle de versões.

3.3 Critérios de Avaliação

- **Funcionalidade:** Todas as operações (cadastro, login, carrinho) devem funcionar sem erros.
- **Segurança:** Validações de token e estoque implementadas.
- **Usabilidade:** Interface intuitiva com feedback visual.

4. Desenvolvimento e Implementação

4.1 Arquitetura do Sistema

- O sistema adota uma arquitetura cliente-servidor:
- **Backend:** API RESTful centralizada em index.php, roteando para módulos específicos (auth.php, cart.php).
- **Frontend:** Páginas HTML com scripts JS modulares, comunicando via *fetch API*.

4.2 Banco de Dados

- Foram criadas seis tabelas principais:
- categorias: ID e nome de categorias.
- produtos: Detalhes de produtos (nome, preço, imagem, estoque, categoria).
- usuarios: Dados de usuários (nome, email, senha hash, token).
- carrinho: Itens temporários (*usuario_id*, *id_produto*, *quantidade*).
- vendas: Pedidos finalizados.
- tens_pedido: Detalhes de vendas.
- **Exemplo de SQL para criação:**

CREATE TABLE produtos (

id_produto INT AUTO_INCREMENT PRIMARY KEY,
nome_produto VARCHAR(255) NOT NULL,
preco DECIMAL(10,2) NOT NULL,
imagem VARCHAR(255),
categoria_id INT,
estoque INT DEFAULT 0,
FOREIGN KEY (categoria_id) REFERENCES categorias(id_categoria)
);

4.3 API Backend

- **Endpoints:**
- POST /auth/register: Cadastra usuário, valida campos e gera token.
- POST /auth/login: Autentica e retorna token.
- GET /products: Retorna produtos com filtro opcional por categoria.
- POST /cart: Adiciona item ao carrinho (valida estoque e token).
- GET /cart: Busca itens do carrinho com detalhes (nome, preço, imagem).
- PUT /cart: Atualiza quantidade.
- DELETE /cart: Remove item.
- **Validações:** Autenticação obrigatória via *header* Authorization; verificação de estoque para evitar vendas excessivas.

4.4 Frontend

- **Páginas:**
- index.html: Catálogo inicial.
- produtos.html: Produtos filtrados.
- carrinho.html: Gerenciamento de carrinho.
- login.html e cadastro.html: Autenticação.
- **Scripts JS:**
- produtoCard.js: Renderiza produtos e adiciona ao carrinho (verifica token).
- cartRender.js: Exibe carrinho e eventos de +/-/remover.
- register.js: Lida com cadastro.

- main.js: Funções globais (ex.: salvar token).
- **Exemplo de código JS para adicionar ao carrinho:**

```
const token = localStorage.getItem('token');
if (!token) {
    // Redireciona ou exibe mensagem de erro
}
//... lógica de fetch para POST /cart
```

5. Resultados

5.1 Funcionalidades Implementadas

- **Autenticação:** Cadastro e login funcionais, com redirecionamento automático.
- **Navegação de Produtos:** Exibição dinâmica com filtros por categoria.
- **Carrinho Persistente:** Itens salvos no banco, recuperados em qualquer sessão.
- **Gerenciamento de Carrinho:** Botões para ajustar quantidade e remover itens, com atualização em tempo real.
- **Validações:** Estoque verificado; usuários não logados redirecionados para login.

5.2 Testes Realizados

- **Testes de API:** Via Postman, confirmando respostas corretas (ex.: 200 para sucesso, 401 para token inválido).
- **Testes de Frontend:** Navegação completa, incluindo cenários de erro (ex.: adicionar sem login).
- **Cenários de Uso:** Cadastro → Login → Adição de produtos → Gerenciamento de carrinho.
- **Resultado:** Sistema funcional sem *crashes*, com dados persistentes.

5.3 Métricas de Desempenho

- **Tempo de Resposta:** API responde em <1s localmente.
- **Cobertura de Funcionalidades:** 100% dos requisitos básicos atendidos.
- **Erros Identificados:** Inicialmente, inconsistências em nomes de campos (corrigidas via *debug*).

6. Discussão

6.1 Análise dos Resultados

- O projeto demonstrou sucesso na implementação de um e-commerce básico, com

ênfase na persistência de dados. A arquitetura modular facilitou a manutenção, e as validações de segurança preveniram vulnerabilidades comuns. Comparado a plataformas comerciais, este sistema é simplificado, mas escalável.

6.2 Limitações

- **Escopo Limitado:** Sem *checkout* ou pagamento, limitando a usabilidade real.
- **Segurança:** Tokens não expiram; falta HTTPS em produção.
- **Performance:** Não otimizado para alto tráfego (ex.: sem cache).
- **Usabilidade:** Interface básica; poderia incluir *feedback* visual aprimorado.

6.3 Contribuições

- Este trabalho contribui para o entendimento de desenvolvimento web *full-stack*, demonstrando integração de tecnologias modernas. Serve como base para projetos maiores, como *marketplaces*.

7. Conclusão

- O desenvolvimento da plataforma de e-commerce foi concluído com sucesso, atendendo aos objetivos de persistência de carrinho e autenticação segura. A implementação demonstrou competência em PHP, JavaScript e MySQL, resultando em um sistema funcional e modular. Recomenda-se expansões futuras, como integração de pagamentos e otimização de UI, para aplicações comerciais.