

**UNISUAM** (Centro Universitário Augusto Motta)

## DOCUMENTAÇÃO DA ARQUITETURA E FUNCIONAMENTO DO SITE

### **PERIGO TECH**

Participantes: Breno Morais, Diogo Bello, Italo Romulo, Luiz Dias, Yan Oliveira

Campo Grande/RJ, 24de Setembro de 2025  
Atualizado em 20 de Novembro de 2025

## **PARTE 1 da DOCUMENTAÇÃO**

### **1. INTRODUÇÃO**

Este documento detalha a arquitetura e o funcionamento de um sistema web composto por um portal de e-commerce e um módulo de gerenciamento de usuários. O objetivo é descrever cada componente, desde a estrutura dos bancos de dados até a lógica de programação dos arquivos PHP, JavaScript e MySQL, oferecendo uma visão completa dos fluxos operacionais, funcionalidades e considerações de segurança da aplicação.

### **2. ARQUITETURA GERAL DO PROJETO**

O sistema foi projetado de forma modular, com uma separação clara entre a gestão de usuários e a vitrine de produtos. Essa abordagem utiliza dois bancos de dados distintos para garantir a organização e a segurança dos dados.

#### **2.1 Sistema de Cadastro e Autenticação de Usuários**

Este módulo é responsável por todas as operações relacionadas aos usuários. Utiliza o banco de dados `cadastro-tech` e seus principais arquivos são `cadastro.php`, `login.php`, `testeLogin.php` e o conjunto de arquivos do painel administrativo. A conexão é gerenciada pelo arquivo `config.php`.

## 2.2 Sistema da LojaVirtual (E-commerce)

Este módulo corresponde à interface pública da loja. Utiliza o banco de dados perigotech para armazenar e exibir informações de produtos. Seus arquivos principais são loja.php, prod.php e carrinho.php. A configuração de conexão está contida diretamente nos próprios arquivos que a utilizam.

## 2.3 Estrutura do Banco de Dados de Usuários (cadastro\_tech.sql)

O arquivo `cadastro_tech.sql` define a estrutura da tabela principal para o sistema de usuários. A tabela, denominada `cadastro_tech`, é projetada para armazenar todas as informações relevantes dos usuários registrados na plataforma. As colunas incluem dados pessoais (nome, e-mail, CPF, data de nascimento), informações de contato (telefone, endereço, CEP), credenciais de acesso (login, senha) e dados para recuperação de conta (nome da mãe). O campo `idusuarios` serve como chave primária auto-incrementável, garantindo um identificador único para cada registro.

## 3. FLUXO DE CADASTRO E AUTENTICAÇÃO DO USUÁRIO

O fluxo de autenticação descreve o percurso completo do usuário, desde o primeiro acesso e registro até a validação de segurança para acesso a áreas restritas.

### 3.1 Cadastro de Novo Usuário (cadastro.php)

A página `cadastro.php` serve como ponto de entrada para novos usuários. O front-end apresenta um formulário para coleta de dados pessoais, enquanto o back-end em PHP processa essas informações. O script PHP utiliza Prepared Statements para inserir os dados no banco de forma segura, prevenindo ataques de SQL Injection. Em caso de sucesso, o usuário é redirecionado para a página de login.

### 3.2 Configuração do Banco de Dados (config.php)

O arquivo `config.php` centraliza as credenciais de conexão com o banco de dados `cadastro-tech`, estabelecendo uma instância de conexão `mysqli` que é reutilizada em todos os scripts que manipulam dados de usuários.

### 3.3 Tela de Login (login.php)

Apresenta um formulário para que o usuário insira suas credenciais (login e senha). Os dados são submetidos via método POST ao arquivo testeLogin.php para validação.

### 3.4 Processamento de Login (testeLogin.php)

Este script valida as credenciais do usuário. Ele recebe os dados, conecta-se ao banco cadastro-tech e executa uma consulta para verificar a existência do usuário. Em caso de sucesso, armazena os dados do usuário na sessão (`$_SESSION`) e o redireciona para a área apropriada (neste caso, sistema.php). Caso contrário, o usuário é redirecionado para a loja.

### 3.5 Autenticação de Dois Fatores – 2FA (2fa.php e back.php)

Após o login, o arquivo 2fa.php implementa uma camada adicional de segurança. Ele exibe uma pergunta de segurança selecionada aleatoriamente (ex.: "Qual o nome da sua mãe?"). A resposta do usuário é enviada de forma assíncrona ao back.php, que valida a informação contra os dados armazenados no banco. Após três tentativas malsucedidas, o usuário é desconectado.

## 4. A LOJAVIRTUAL

Esta seção descreve os componentes que formam a interface de e-commerce da aplicação.

### 4.1 Estrutura do Banco de Dados da Loja (BD\_tela\_inicial.sql)

O arquivo BD\_tela\_inicial.sql contém os comandos para criar o banco de dados perigotech e a tabela produtos. Ele também inclui registros INSERT para popular a loja com um catálogo inicial de produtos, definindo a base de dados sobre a qual a loja opera.

### 4.2 Página da Loja (loja.php)

A página loja.php atua como a vitrine principal da loja virtual. O back-end em PHP

conecta-se ao banco perigotech e busca os produtos, organizando-os por categorias em carrosséis gerados dinamicamente.

Para padronizar a exibição de cada item, foi criada a função renderProduto(), que garante consistência visual e facilita a manutenção do código.

#### 4.3 Página de Detalhes do Produto (prod.php)

Ao clicar em um produto na loja, o usuário é direcionado para a página prod.php. O script captura o ID do produto via URL (`$_GET['id']`), busca suas informações detalhadas no banco de dados e as exibe de forma organizada. Além disso, a lógica em PHP calcula e apresenta diferentes opções de parcelamento do preço, garantindo clareza para o usuário.

#### 4.4 CarrinhodeCompras(carrinho.php)

Atualmente, o carrinho.php apresenta uma interface estática do carrinho de compras. Os produtos exibidos são fixos (hardcoded).

As interações, como adicionar ou remover itens, são simuladas por alertas em JavaScript.

Esta abordagem foi adotada como protótipo, sendo prevista a implementação futura de funcionalidades dinâmicas, mantendo a consistência visual da página.

### 5. PAINEL ADMINISTRATIVO

Esta seção é de acesso restrito e destina-se ao gerenciamento de usuários cadastrados.

#### 5.1 Dashboard do Sistema (sistema.php)

O sistema.php é o painel central da administração. O acesso é protegido por uma verificação que garante que apenas usuários logados e com permissão de administrador (presentes no array `$admin_users`) possam visualizá-lo. A página exibe uma tabela com todos os usuários cadastrados e oferece opções para editar ou excluir cada registro.

#### 5.2 Edição de Cliente (edit.php)

Este arquivo apresenta um formulário pré-preenchido com os dados de um usuário específico, selecionado a partir do dashboard. A função `htmlspecialchars()` é utilizada para exibir os dados de forma segura, prevenindo ataques de Cross-Site Scripting (XSS). As alterações são submetidas ao `atualizar_cliente.php`.

#### 5.3 Atualização do Cliente (atualizar\_cliente.php)

Recebe os dados do formulário de edição e executa uma consulta `UPDATE` no banco de dados. Para prevenir SQL Injection, utiliza a função `mysqli_real_escape_string()` para tratar os dados recebidos antes de construir a consulta. Após a atualização, o administrador é redirecionado de volta ao dashboard.

#### 5.4 Exclusão de Cliente (delete.php)

Este script é responsável por remover um usuário do banco de dados. Ele recebe o ID do usuário, verifica sua existência e, se confirmado, executa um comando `DELETE`. Uma confirmação via JavaScript é solicitada no front-end para prevenir exclusões acidentais.

### 6. CONSIDERAÇÕES DE SEGURANÇA

Durante a análise do projeto, foram identificados pontos críticos de segurança no script `testeLogin.php`. A construção de consultas SQL por concatenação direta de variáveis torna a aplicação vulnerável a SQL Injection. Além disso, a comparação direta de senhas indica que elas estão armazenadas em texto claro no banco de dados. Recomenda-se enfaticamente a adoção de Prepared Statements também para o login e a implementação de hashing de senhas, utilizando funções como `password_hash()` e `password_verify()`.

### 7. CONCLUSÃO

O projeto apresenta uma arquitetura bem definida, com uma clara separação entre a lógica de negócios do e-commerce e o gerenciamento de usuários. As funcionalidades

cobrem o ciclo completo de vida do usuário e a exibição de produtos. Embora funcional, a aplicação requer melhorias de segurança, especialmente no processo de autenticação, para garantir a proteção dos dados e a integridade do sistema.

## **PARTE 2 da DOCUMENTAÇÃO**

### **8. LOG DO SISTEMA**

#### **8. REGISTRO DE LOGS DO SISTEMA (log.php)**

O log.php implementa uma função essencial para monitoramento de ações e diagnóstico de falhas. Esse registro automático permite: Auditoria de eventos do sistema Rastreabilidade de ações executadas por usuários, apoio a manutenção e correções futuras.

### **9. TELAS DE AVISO DE POSSÍVEIS ERROS**

#### **8. TELA DE ERRO (erro.php)**

A tela de erro é exibida quando ocorre alguma falha no processamento de ações do usuário, seja por dados inválidos, acesso não permitido ou erro interno do sistema. O objetivo é garantir uma comunicação clara e orientativa, evitando que o usuário fique sem direção dentro do fluxo do sistema. Entre suas funcionalidades, destacam-se:

Exibição de mensagem amigável sobre o erro ocorrido

Orientação ao usuário para tentar novamente ou retornar à página anterior

Manutenção da identidade visual do sistema e navegação do usuário.

Esta página melhora a usabilidade ao impedir que o usuário permaneça em uma rota inválida no sistema.

## 10. GERAÇÃO DE PDF DO SISTEMA DE LOG

### 9. Geração de PDF no .log (gerar\_pdf.php)

O arquivo gerar\_pdf.php tem a função de gerar automaticamente um documento em PDF com base nos dados registrados no sistema. Principais responsabilidades:

Exibição da tela de usuários registrados no LOG

Conversão estruturada para PDF

Disponibilização do arquivo para download.

Esse recurso amplia a experiência do usuário, agregando valor ao sistema com a possibilidade de emissão de usuários registrados no LOG

## 11. TELA DE VISUALIZAÇÃO DA ESTRUTURA DO BANCO DE DADOS

### 11. Tela do Banco de Dados (telabd.php)

A página telabd.php é utilizada para exibir diretamente a estrutura do banco de dados empregado no sistema. Ela apresenta as tabelas existentes, seus nomes e organização interna, permitindo que desenvolvedores e testadores confirmem a correta modelagem do banco durante o desenvolvimento.

Principais funcionalidades:

Conexão com o banco de dados utilizado pelo sistema

Listagem das tabelas existentes

Exibição da estrutura organizacional do banco (schema).

Esta tela exibe exatamente como está a organização do Banco de Dados, todas as tabelas e todos os seus dados

#### 4.4 CARRINHO DE COMPRAS — VERSÃO ATUALIZADA

A página carrinho.php implementa de forma completa a lógica de gerenciamento de compras do usuário dentro do e-commerce.

Ela permite que o usuário:

Adicione itens diretamente a partir da loja

Remova produtos do carrinho

Atualize quantidades de itens

Visualie o valor total da compra em tempo real

Prossiga para as próximas etapas do processo de finalização.

As ações são controladas via sessão PHP, garantindo que o carrinho seja mantido enquanto o usuário navega pelo sistema. Toda a interação é feita de forma dinâmica e integrada ao banco de dados, assegurando consistência na exibição das informações.

Esta funcionalidade consolida a experiência de compra e permite que o usuário simule diferentes escolhas antes de finalizar o pedido.

## CONCLUSÃO GERAL

A aplicação Perigo Tech apresenta-se como um sistema web funcional que integra um módulo de gerenciamento de usuários com uma loja virtual de produtos. A separação desses dois ambientes garante maior organização dos dados e possibilita a evolução independente de cada componente. O fluxo operacional cobre desde o cadastro e autenticação dos usuários, incluindo mecanismos de segurança como verificação de informações pessoais, até a navegação pela vitrine da loja com exibição dinâmica de produtos. O painel de administração e as páginas de suporte ao desenvolvimento — como a visualização da estrutura do banco de dados e o registro de logs — reforçam o aspecto técnico do sistema e sua capacidade de manutenção. Ainda que o projeto apresente uma arquitetura sólida para um ambiente acadêmico e experimental, oportunidades de aprimoramento foram identificadas, principalmente na proteção de dados sensíveis e no fortalecimento dos mecanismos de segurança da autenticação. Além disso, a evolução do carrinho de compras para um ambiente totalmente dinâmico contribuirá para a experiência do usuário final. Assim, o Perigo Tech demonstra um potencial elevado para expansão futura, podendo incorporar novas funcionalidades e camadas de segurança à medida que o sistema evolui e sua complexidade aumenta. A documentação aqui apresentada oferece suporte para continuidade do desenvolvimento, garantindo compreensão clara sobre a estrutura, o funcionamento e as perspectivas do projeto.