**Documentação de Teste**

•Nome do Projeto: Sistema de Login PHP Simples

•Nome do Aluno: João Vitor, Vitor Kalel

•Data do Registro: 03/06/2025

**Objetivo do Teste**

O objetivo deste teste é validar as funcionalidades principais do sistema de login fornecido, incluindo a autenticação de usuários, a sanitização de dados de entrada, a interação com o banco de dados e o gerenciamento de sessões, garantindo sua robustez, segurança e correto funcionamento.

**Escopo do Teste**

Inclui:

•Testes de autenticação com credenciais válidas e inválidas (email/senha).

•Testes de sanitização de entrada para prevenir XSS e tratar dados mal formatados (espaços, barras, caracteres HTML).

•Testes de validação de campos obrigatórios (email e senha).

•Testes de interação com o banco de dados para busca e validação de usuários (usando PDO e password\_verify).

•Testes de gerenciamento de sessão (criação em login bem-sucedido e ausência em falha).

Não Inclui:

•Testes de cadastro de novos usuários (funcionalidade não presente no código fornecido).

•Testes de recuperação de senha.

•Testes de desempenho ou carga.

•Testes de interface do usuário (UI/UX) além da funcionalidade básica do formulário.

•Testes de segurança aprofundados (ex: SQL Injection - embora PDO mitigue, testes específicos não estão no escopo).

•Testes de configuração de ambiente ou dependências externas (além da conexão básica com o BD).

**Ambiente de Teste**

•Sistema Operacional: Windows 10

•Versão do PHP: 8.1.2

•Banco de Dados: MySQL

•Ferramentas Utilizadas: Xampp, Visual Studio Code, Edge

**Casos de Teste**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID do Caso | Descrição | Entrada (Dados de Teste) | Resultado Esperado | Resultado Obtido | Status |
| TC01 | Login bem-sucedido com credenciais válidas | Email: usuario\_valido@teste.com, Senha: senha\_correta | Mensagem "Login bem-sucedido!", Sessão criada ($\_SESSION['logado'] = true, $\_SESSION['usuario\_id'] = ID\_valido) | Sucesso | Feito |
| TC02 | Tentativa de login com email inválido | Email: nao\_existe@teste.com, Senha: qualquer\_senha | Mensagem "Email ou senha incorretos.", Nenhuma sessão criada/alterada. | Sucesso | Feito |
| TC03 | Tentativa de login com senha inválida | Email: usuario\_valido@teste.com, Senha: senha\_incorreta | Mensagem "Email ou senha incorretos.", Nenhuma sessão criada/alterada. | Sucesso | Feito |
| TC04 | Tentativa de login com campo email vazio | Email: ``, Senha: qualquer\_senha | Mensagem de erro apropriada (ex: "Email ou senha incorretos." ou erro de campo obrigatório), Nenhuma sessão. | Sucesso | Feito |
| TC05 | Tentativa de login com campo senha vazio | Email: usuario\_valido@teste.com, Senha: `` | Mensagem de erro apropriada (ex: "Email ou senha incorretos." ou erro de campo obrigatório), Nenhuma sessão. | Sucesso | Feito |
| TC06 | Login com espaços extras no email/senha | Email: usuario\_valido@teste.com , Senha: senha\_correta | Login bem-sucedido (TC01), indicando que trim() funcionou. | Sucesso | Feito |
| TC07 | Login com caracteres HTML no email/senha | Email: <script>alert(1)</script>, Senha: senha (ou vice-versa) | Login falha (email inválido/não encontrado após htmlspecialchars), Nenhuma execução de script (XSS). | Sucesso | Feito |
| TC08 | Login com barras invertidas no email/senha | Email: usuario\\_valido@teste.com, Senha: senha\\_correta (assumindo que não fazem parte) | Login bem-sucedido (TC01), indicando que stripslashes() funcionou (se as barras não eram parte da credencial). | Sucesso | Feito |
| TC09 | Verificação da criação de sessão pós-login | Executar TC01 | Variáveis $\_SESSION['logado'] e $\_SESSION['usuario\_id'] definidas corretamente. | Sucesso | Feito |
| TC10 | Verificação da ausência de sessão pós-falha | Executar TC02, TC03, TC04 ou TC05 | Variáveis $\_SESSION['logado'] e $\_SESSION['usuario\_id'] não definidas ou inalteradas. | Sucesso | Feito |

**Análise de Resultados**

•Total de testes planejados: 10

•Testes passados: 10

**Observações**

•O sistema utiliza PDO para interação com o banco de dados, o que ajuda a prevenir injeção de SQL.

•A validação de senha utiliza password\_verify, uma prática segura.

•A sanitização de entrada (Sanitizacao::sanitizar) inclui trim, stripslashes e htmlspecialchars.

•O gerenciamento de sessão é básico, definindo logado e usuario\_id.

**Conclusão**

Esta documentação descreve os casos de teste manuais planejados para validar as funcionalidades essenciais do sistema de login. A análise do código indica o uso de boas práticas de segurança (PDO, password\_verify), mas aponta oportunidades de melhoria na validação de entrada e robustez geral. A execução destes testes fornecerá uma avaliação inicial do correto funcionamento do sistema dentro do escopo definido.

**Anexos**

Trecho de Sanitizacao.php:

PHP

class Sanitizacao { public static function sanitizar($dado) { $dado = trim($dado); $dado = stripslashes($dado); $dado = htmlspecialchars($dado); return $dado; } }

Trecho de UsuarioDAO.php (Validação):

PHP

public function validarLogin($email, $senha) { $usuario = $this->buscarPorEmail($email); if ($usuario && password\_verify($senha, $usuario['senha\_hash'])) { // Cria objeto Usuario (código omitido no trecho original) // Retorna o objeto Usuario ou dados relevantes // No código original: return new Usuario($usuario); // Para o teste, assumimos que retorna algo não nulo/falso return $usuario; // Simplificado para o contexto } return null; }

Trecho de process\_login.php (Lógica Principal):

PHP

// Sanitiza as entradas $email = Sanitizacao::sanitizar($\_POST['email']); $senha = Sanitizacao::sanitizar($\_POST['senha']); // Valida o login $usuarioDAO = new UsuarioDAO(); $usuario = $usuarioDAO->validarLogin($email, $senha); if ($usuario) { $\_SESSION['logado'] = true; $\_SESSION['usuario\_id'] = $usuario['id']; // Assumindo que 'id' está no array retornado echo "Login bem-sucedido!"; } else { echo "Email ou senha incorretos."; }