# **Documentação de Teste – Sistema de Login PHP Simples**

## **1. Informações Gerais**

* **Nome do Projeto:** Sistema de Login PHP Simples
* **Autores:** Pedro Augusto, Jonas Maia
* **Data de Registro:** 17/06/2025

## **2. Objetivo dos Testes**

Validar as funcionalidades principais do sistema de login, com foco nos seguintes pontos:

* Autenticação de usuários com credenciais válidas.
* Sanitização e validação de dados de entrada.
* Validação de campos obrigatórios.
* Interação correta com o banco de dados (PDO e password\_verify).
* Gerenciamento de sessões (criação e persistência pós-login).

## **3. Escopo dos Testes**

### **Incluído:**

* Testes de autenticação (credenciais válidas e inválidas).
* Testes de sanitização de entrada (espaços, caracteres especiais, HTML, barras invertidas).
* Testes de validação de campos obrigatórios.
* Testes de interação com o banco de dados.
* Testes de criação e controle de sessão.

### **Não Incluído:**

* Cadastro de novos usuários.
* Recuperação de senha.
* Testes de desempenho ou carga.
* Testes de interface gráfica (UI/UX).
* Testes de segurança avançada (ex.: SQL Injection).
* Configuração de ambiente.

## **4. Ambiente de Testes**

* **Sistema Operacional:** Windows 10
* **Versão do PHP:** 8.1.2
* **Banco de Dados:** MySQL
* **Ferramentas Utilizadas:** XAMPP, Visual Studio Code, Microsoft Edge

## **5. Critérios de Aceitação**

* Login deve ser autorizado apenas com email e senha corretos.
* Campos obrigatórios devem exibir mensagens de erro quando vazios.
* Entradas devem ser sanitizadas antes de qualquer interação com o banco.
* Sessões devem ser criadas apenas em login bem-sucedido.

## **6. Casos de Teste**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Descrição** | **Entrada (Dados de Teste)** | **Resultado Esperado** | **Status** |
| TC01 | Login bem-sucedido | Email: [usuario\_valido@teste.com](mailto:usuario_valido@teste.com) Senha: senha\_correta | Mensagem: "Login bem-sucedido!" Sessão criada ($\_SESSION['logado'] = true) | ✅ Feito |
| TC02 | Email inválido | Email: [nao\_existe@teste.com](mailto:nao_existe@teste.com) Senha: qualquer\_senha | Mensagem: "Email ou senha incorretos." Sem sessão criada | ✅ Feito |
| TC03 | Senha inválida | Email: [usuario\_valido@teste.com](mailto:usuario_valido@teste.com) Senha: senha\_incorreta | Mensagem: "Email ou senha incorretos." Sem sessão criada | ✅ Feito |
| TC04 | Email vazio | Email: "" Senha: qualquer\_senha | Mensagem de erro (campo obrigatório) Sem sessão criada | ✅ Feito |
| TC05 | Senha vazia | Email: [usuario\_valido@teste.com](mailto:usuario_valido@teste.com) Senha: "" | Mensagem de erro (campo obrigatório) Sem sessão criada | ✅ Feito |
| TC06 | Login com espaços extras | Email: " [usuario\_valido@teste.com](mailto:usuario_valido@teste.com) " Senha: senha\_correta | Login bem-sucedido (uso de trim()) | ✅ Feito |
| TC07 | Login com caracteres HTML | Email: <script>alert(1)</script> Senha: senha | Login falha (sanitização com htmlspecialchars) Sem XSS | ✅ Feito |
| TC08 | Login com barras invertidas | Email: [usuario\_valido@teste.com](mailto:usuario_valido@teste.com) Senha: senha\_correta | Login bem-sucedido (uso de stripslashes()) | ✅ Feito |
| TC09 | Verificação de sessão pós-login | Executar TC01 | $\_SESSION['logado'] e $\_SESSION['usuario\_id'] definidos | ✅ Feito |
| TC10 | Verificação de ausência de sessão | Executar TC02, TC03, TC04 ou TC05 | $\_SESSION['logado'] e $\_SESSION['usuario\_id'] não definidos | ✅ Feito |

## **7. Análise de Resultados**

* **Total de Testes Planejados:** 10
* **Testes Executados com Sucesso:** 10
* **Taxa de Sucesso:** 100%

## **8. Observações**

### **Segurança:**

* Uso de **PDO** para evitar SQL Injection.
* Validação de senha com **password\_verify**.
* Sanitização aplicada com **trim()**, **stripslashes()**, e **htmlspecialchars()**.

### **Gerenciamento de Sessões:**

* Sessões básicas implementadas (logado e usuario\_id).

### **Oportunidades de Melhoria:**

* Adicionar validação de formato de email (ex.: filter\_var($email, FILTER\_VALIDATE\_EMAIL)).
* Implementar **testes automatizados** (ex.: com **PHPUnit**).
* Futuramente incluir cenários de ataque (testes de segurança avançados).

## **9. Conclusão Final**

O sistema de login PHP atende aos requisitos funcionais básicos estabelecidos para esta fase do projeto. Os testes confirmaram a correta autenticação de usuários, a segurança mínima contra entradas maliciosas e o correto gerenciamento de sessões.

Para etapas futuras, recomenda-se fortalecer a validação de entrada e implementar uma suíte de testes automatizados.

## **10. Anexos (Trechos de Código Relevantes)**

### **10.1 Sanitizacao.php**

php

CopiarEditar

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

### **10.2 UsuarioDAO.php (Validação)**

php

CopiarEditar

public function validarLogin($email, $senha) {  
 $usuario = $this->buscarPorEmail($email);  
 if ($usuario && password\_verify($senha, $usuario['senha\_hash'])) {  
 return $usuario; // Retorna dados do usuário válido  
 }  
 return null;  
}

### **10.3 process\_login.php (Lógica Principal)**

php

CopiarEditar

$email = Sanitizacao::sanitizar($\_POST['email']);  
$senha = Sanitizacao::sanitizar($\_POST['senha']);  
  
$usuarioDAO = new UsuarioDAO();  
$usuario = $usuarioDAO->validarLogin($email, $senha);  
  
if ($usuario) {  
 $\_SESSION['logado'] = true;  
 $\_SESSION['usuario\_id'] = $usuario['id'];  
 echo "Login bem-sucedido!";  
} else {  
 echo "Email ou senha incorretos.";  
}