# Documento de Registro de Teste

## Informações Gerais

* **Nome do Projeto:** Sistema de Cadastro e Login PHP
* **Nome do Aluno:** João Vitor, Vitor
* **Data do Registro:** 17/06/2025

Objetivo do Teste

O objetivo deste teste é validar as funcionalidades de cadastro e autenticação de usuários no sistema PHP fornecido, garantindo a correta implementação da sanitização de dados, validação de entradas, armazenamento seguro de senhas, verificação de credenciais e gerenciamento de sessões.

## Escopo do Teste

**Inclui:** \* Testes de cadastro de usuários com validação de senha e email. \* Testes de autenticação (login) com credenciais válidas e inválidas. \* Testes de sanitização de dados de entrada para prevenção de vulnerabilidades. \* Testes de validação de campos obrigatórios e regras de negócio (tamanho mínimo de senha, confirmação de senha). \* Testes de verificação de unicidade de email no cadastro. \* Testes de gerenciamento de sessão após autenticação bem-sucedida.

**Não Inclui:** \* Testes de recuperação de senha (funcionalidade não implementada). \* Testes de desempenho ou carga do sistema. \* Testes de integração com sistemas externos. \* Testes de interface do usuário além da funcionalidade básica dos formulários. \* Testes de segurança aprofundados além da sanitização básica e hash de senha. \* Testes de configuração de ambiente ou dependências externas.

## Ambiente de Teste

* **Sistema Operacional:** Windows 10
* **Versão do PHP:** 8.1.2
* **Banco de Dados:** MySQL 8.1
* **Ferramentas Utilizadas:** Edge, Xampp 3.3.0

## Casos de Teste

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID do Caso | Descrição | Entrada | Resultado Esperado | Status |
| TC01 | Cadastro com dados válidos | Email: “usuario@teste.com”, Senha: “senha123”, Confirmar senha: “senha123” | Mensagem “Cadastro realizado com sucesso!” e registro inserido no banco de dados | Sucesso |
| TC02 | Cadastro com senhas diferentes | Email: “usuario@teste.com”, Senha: “senha123”, Confirmar senha: “senha456” | Mensagem “As senhas não coincidem.” | Sucesso |
| TC03 | Cadastro com senha curta | Email: “usuario@teste.com”, Senha: “123”, Confirmar senha: “123” | Mensagem “A senha deve ter pelo menos 6 caracteres.” | Sucesso |
| TC04 | Cadastro com email já existente | Email: “existente@teste.com” (já cadastrado), Senha: “senha123”, Confirmar senha: “senha123” | Mensagem “Erro ao cadastrar. O email já pode estar em uso.” | Sucesso |
| TC05 | Login com credenciais válidas | Email: “usuario@teste.com”, Senha: “senha123” | Mensagem “Login bem-sucedido!” e criação de sessão de usuário | Sucesso |
| TC06 | Login com email inválido | Email: “naoexiste@teste.com”, Senha: “qualquer” | Mensagem “Email ou senha incorretos.” | Sucesso |
| TC07 | Login com senha incorreta | Email: “usuario@teste.com”, Senha: “senhaerrada” | Mensagem “Email ou senha incorretos.” | Sucesso |
| TC08 | Cadastro com espaços extras | Email: " usuario2@teste.com “, Senha:” senha123 “, Confirmar senha:” senha123 " | Cadastro bem-sucedido após remoção dos espaços | Sucesso |
| TC09 | Login com espaços extras | Email: " usuario@teste.com “, Senha:” senha123 " | Login bem-sucedido após remoção dos espaços | Sucesso |

## Análise de Resultados

* **Total de testes planejados:** 9
* **Testes passados:**

**Observações:** \* O sistema implementa sanitização de entrada através da classe Sanitizacao, que aplica trim(), stripslashes() e htmlspecialchars() aos dados recebidos. \* A validação de senha no cadastro verifica a correspondência entre os campos e o tamanho mínimo (6 caracteres). \* O sistema utiliza PDO para conexão com o banco de dados, o que ajuda a prevenir injeção de SQL. \* As senhas são armazenadas de forma segura utilizando password\_hash() e verificadas com password\_verify(). \* O sistema verifica a unicidade do email antes de permitir um novo cadastro. \* O gerenciamento de sessão é implementado após login bem-sucedido, definindo variáveis de sessão para controle de acesso.

**Sugestões:** \* Implementar validação mais robusta de formato de email no backend. \* Adicionar validação de complexidade de senha (letras, números, caracteres especiais). \* Implementar mecanismo de recuperação de senha. \* Adicionar proteção contra ataques de força bruta (limite de tentativas de login). \* Implementar expiração de sessão por inatividade. \* Melhorar as mensagens de erro para o usuário, mantendo o equilíbrio entre informação e segurança.

## Conclusão

Os testes planejados para o sistema de cadastro e login abrangem as principais funcionalidades e cenários de uso, com foco na validação de dados, segurança e experiência do usuário. O sistema demonstra implementação de boas práticas de segurança como sanitização de entrada, uso de PDO e hash de senhas, mas poderia ser aprimorado com validações adicionais e recursos de recuperação de senha. A execução dos testes fornecerá uma avaliação completa da robustez e confiabilidade do sistema dentro do escopo definido.

## Anexos

**Trecho de código de sanitização:**

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

**Trecho de código de cadastro:**

// Validações básicas  
if ($senha !== $confirmar\_senha) {  
 die("As senhas não coincidem.");  
}  
  
if (strlen($senha) < 6) {  
 die("A senha deve ter pelo menos 6 caracteres.");  
}  
  
// Processa o cadastro  
$usuarioDAO = new UsuarioDAO();  
$resultado = $usuarioDAO->cadastrarUsuario($email, $senha);

**Trecho de código de validação de login:**

public function validarLogin($email, $senha) {  
 $usuario = $this->buscarPorEmail($email);  
 if ($usuario && password\_verify($senha, $usuario['senha\_hash'])) {  
 return new Usuario($usuario);  
 }  
 return null;  
}