Sistema MyLogin – Documentação Completa

Sistema de Login Seguro

Versão 1.0

por PapaiSmurf2050 _ Switch

Introdução

Objetivo

Este projeto implementa um sistema de login seguro e moderno, desenvolvido em PHP com banco de dados MySQL. O sistema foi criado para demonstrar boas práticas de desenvolvimento web, com foco em segurança, usabilidade e design responsivo. Oferece funcionalidades de autenticação completas, incluindo registro de usuários, login seguro, gerenciamento de perfil e proteção contra vulnerabilidades comuns.

Tecnologias Utilizadas

- **Frontend**: HTML5, CSS3 (com variáveis CSS e design responsivo), JavaScript (validação client-side)
- **Backend**: PHP 7.4+, MySQL 5.7+, PDO para conexão com banco de dados
- **Segurança**: Password hashing (bcrypt), Proteção contra SQL Injection, Proteção CSRF, Sanitização de dados, Controle de sessão seguro

Estrutura do Projeto

```
├─── assets/           # Recursos estáticos (imagens, scripts)
├─── docs/             # Documentação do projeto
│    ├─── documentacao.md # Documentação técnica detalhada
│    ├─── README.md      # Instruções rápidas
│    └─── PDF/          # Documentos em PDF
├─── screenshots/      # Capturas de tela do sistema
├─── uploads/          # Diretório para fotos de perfil
├─── config.php        # Configurações e funções globais
├─── database.sql      # Estrutura do banco de dados
├─── landing.php       # Página inicial pública
├─── login.php         # Autenticação de usuários
├─── register.php      # Cadastro de usuários
├─── dashboard.php     # Painel do usuário
├─── logout.php        # Script de logout
├─── index.php         # Página principal do sistema
├─── style.css         # Estilos CSS centralizados
└─── add_column.php    # Script para adicionar coluna foto_perfil
```

Banco de Dados

O sistema utiliza um banco de dados MySQL com as seguintes tabelas:

Tabela users

```
CREATE TABLE users (
id INT AUTO_INCREMENT PRIMARY KEY,
username VARCHAR(50) NOT NULL UNIQUE,
password VARCHAR(255) NOT NULL,
email VARCHAR(100) NOT NULL UNIQUE,
nome VARCHAR(100) NOT NULL,
data_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
foto_perfil VARCHAR(255) DEFAULT NULL
);
```

Tabela remember_tokens

```
CREATE TABLE remember_tokens (
id INT AUTO_INCREMENT PRIMARY KEY,
user_id INT NOT NULL,
token VARCHAR(255) NOT NULL,
expires_at DATETIME NOT NULL,
used TINYINT(1) DEFAULT 0,
FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);
```

Funcionalidades

Sistema de Autenticação

- **Login com username/senha**: Validação de credenciais, Tratamento de tentativas inválidas, Feedback claro de erros
- **Registro de novos usuários**: Validação de dados, Verificação de usernames e emails duplicados, Confirmação de cadastro
- **Função "Lembrar-me"**: Cookies seguros, Tokens de autenticação persistente, Rotação de tokens para maior segurança
- **Segurança avançada**: Limite de tentativas de login, Bloqueio temporário após tentativas falhas, Proteção contra ataques de força bruta

Gerenciamento de Perfil

- **Visualização de dados do usuário**: Dashboard personalizado, Exibição de informações cadastrais
- **Atualização de informações**: Edição de dados pessoais, Validação em tempo real, Feedback de sucesso/erro
- **Alteração de senha**: Verificação de senha atual, Requisitos de força de senha, Atualização segura
- **Upload de foto de perfil**: Suporte a formatos comuns de imagem, Armazenamento seguro, Remoção de fotos existentes

Interface

- **Design responsivo**: Adaptação a diferentes dispositivos, Layout fluido, Uso de media queries
- **Temas e estilos**: Variáveis CSS para fácil personalização, Elementos visuais consistentes, Paleta de cores harmônica
- **Feedback visual**: Mensagens de sucesso/erro, Indicadores de carregamento, Validação em tempo real

Implementação Técnica

Diferenciação Client-Side vs Server-Side

Client-Side (Frontend): Código executado no navegador do usuário para validação de formulários, feedback visual e interatividade.

Server-Side (Backend): Código executado no servidor para processamento de dados, autenticação e acesso ao banco de dados.

Estruturas de Controle

Condicionais: Utilizadas para validação de dados e tomada de decisões no fluxo da aplicação.

Loops: Empregados para processar conjuntos de dados e realizar operações repetitivas.

Manipulação de Dados

Arrays: Estruturas para armazenar e organizar dados relacionados.

Banco de Dados: Operações CRUD (Create, Read, Update, Delete) para persistência de dados.

Fluxos Principais

Registro de Usuário: Desde o preenchimento do formulário até a criação da conta.

Autenticação: Processo de validação de credenciais e estabelecimento de sessão.

Gerenciamento de Perfil: Visualização e atualização de informações do usuário.

Segurança

Proteção contra SQL Injection

Todas as consultas ao banco de dados são realizadas usando PDO com prepared statements, evitando a concatenação direta de strings SQL:

```
$stmt = $pdo->prepare("SELECT * FROM
users WHERE username = ?");
$stmt->execute([$username]);
$user = $stmt->fetch();
```

Proteção XSS

Todo o conteúdo gerado pelo usuário é sanitizado antes de ser exibido:

```
echo htmlspecialchars($user['nome'],
ENT_QUOTES, 'UTF-8');
```

E para dados recebidos:

```
function limparDados($dados) {
  if (is_array($dados)) {
    return array_map('limparDados',
$dados);
  }
  return htmlspecialchars(trim($dados),
ENT_QUOTES, 'UTF-8');
}
```

Proteção CSRF

Uso de tokens para proteger contra ataques de falsificação de requisição:

```
function gerarCSRFToken() {
  if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] =
bin2hex(random_bytes(32));
  }
  return $_SESSION['csrf_token'];
}

function validarCSRFToken($token) {
  return isset($_SESSION['csrf_token']) &&
    hash_equals($_SESSION['csrf_token'],
$token);
}
```

Segurança de Senhas

Senhas são armazenadas usando o algoritmo bcrypt através da função password_hash():

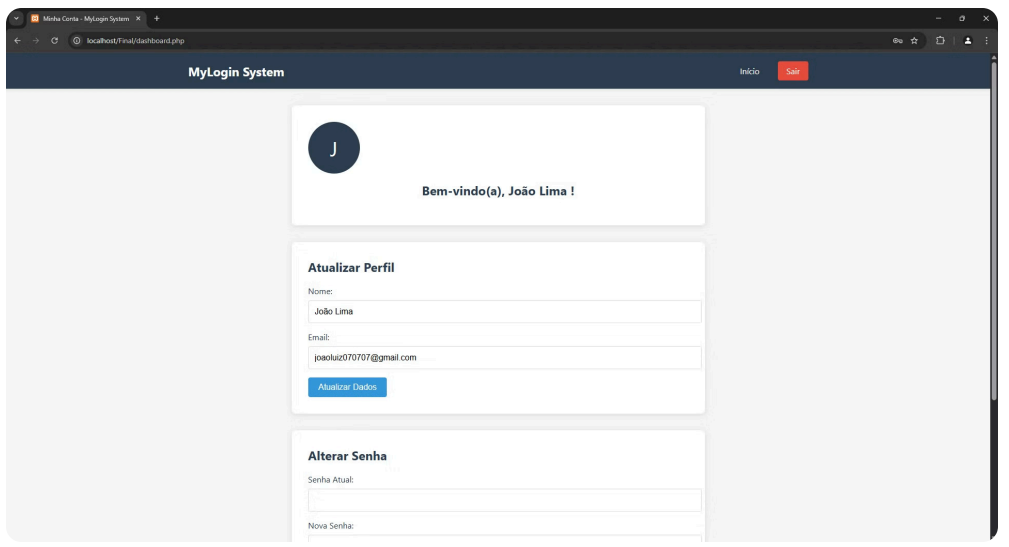
```
// Hash seguro
$hash = password_hash($senha,
PASSWORD_DEFAULT);

// Verificação
if (password_verify($senha_fornecida,
$hash_armazenado)) {
  // Senha correta
}
```

Controle de Sessão e Cookies

Configurações seguras para sessão e cookies "lembrar-me" com parâmetros de segurança adequados.

Interface do Usuário e Implementação da Foto de Perfil



Design Responsivo

Design que se adapta a diferentes tamanhos de tela:

```
/* Design base */
.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 0 20px;
}

/* Responsividade */
@media (max-width: 768px) {
  .nav-container {
    flex-direction: column;
    gap: 1rem;
  }

  .features-grid {
    grid-template-columns: 1fr;
  }
}
```

Feedback Visual

Feedback claro para ações do usuário:

```
// Exemplo de exibição de mensagem
if ($mensagem) {
  echo '
```

```
' . htmlspecialchars($mensagem) . '
';
}
```

Estilização das mensagens:

```
/* Estilo para mensagens de feedback */
.message {
  padding: 1rem;
  margin-bottom: 1rem;
  border-radius: 4px;
  background-color: #d4edda;
  color: #155724;
  border: 1px solid #c3e6cb;
}

.message-error {
  background-color: #f8d7da;
  color: #721c24;
  border: 1px solid #f5c6cb;
}
```

Implementação da Foto de Perfil

O sistema permite que usuários façam upload de fotos de perfil, com processamento seguro e exibição adequada no perfil do usuário.

Instalação, Screenshots e Conclusão

Instalação e Configuração

Requisitos:

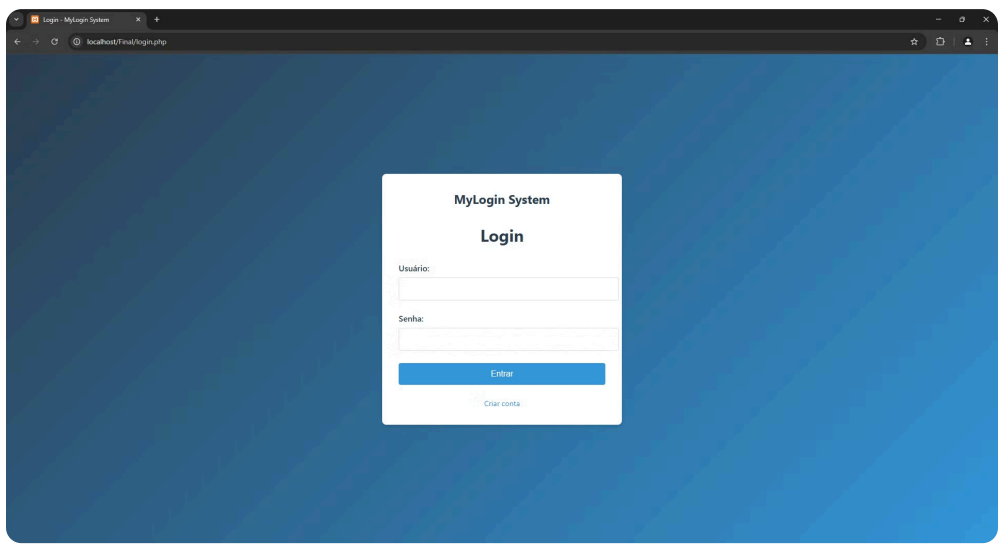
- PHP 7.4 ou superior
- MySQL 5.7 ou superior
- Extensões PHP: PDO, GD (para manipulação de imagens)
- Servidor web (Apache/Nginx)
- Permissões de escrita para o diretório uploads/

Processo de Instalação:

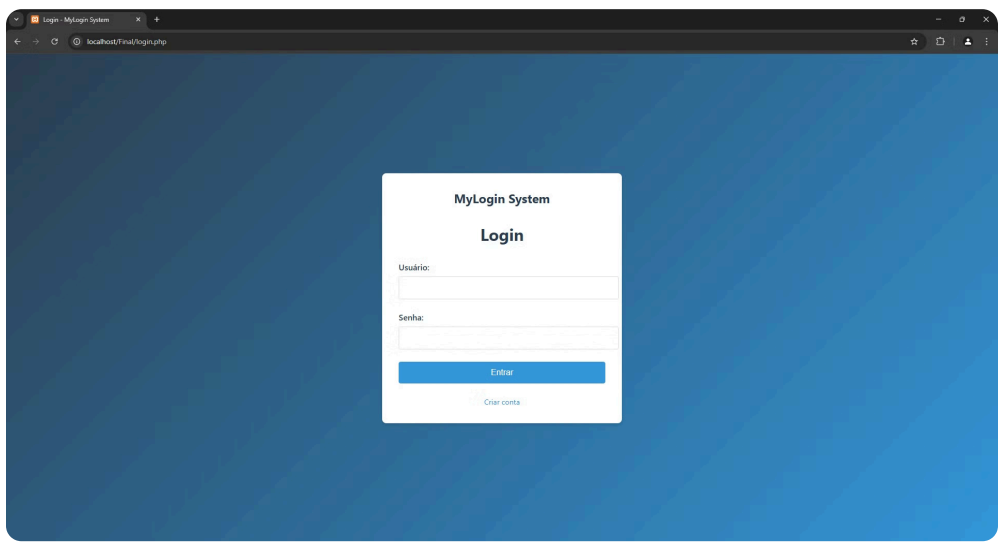
1. Clone ou baixe os arquivos para seu servidor web
2. Importe o banco de dados
3. Configure as permissões do diretório uploads/

Configuração: Edite o arquivo config.php conforme suas configurações.

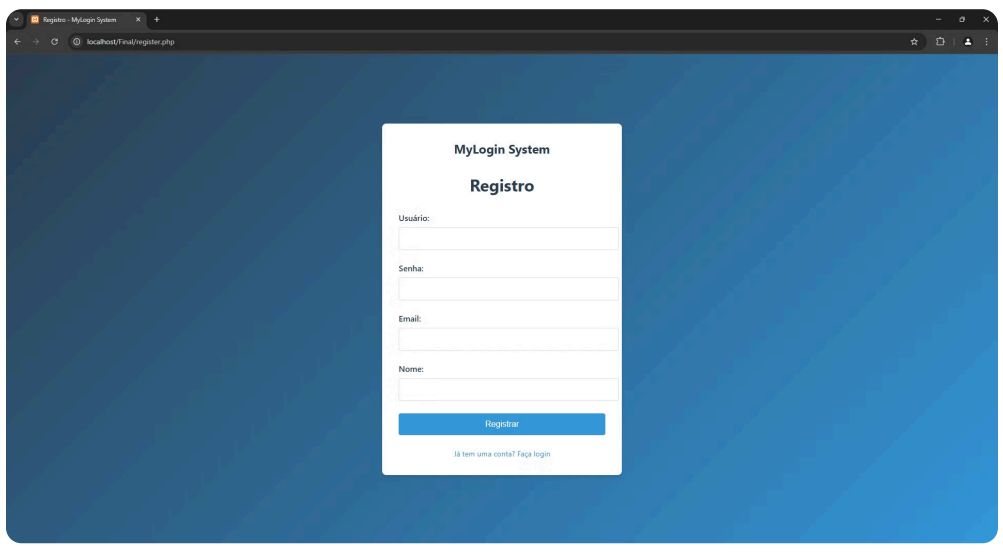
Screenshots



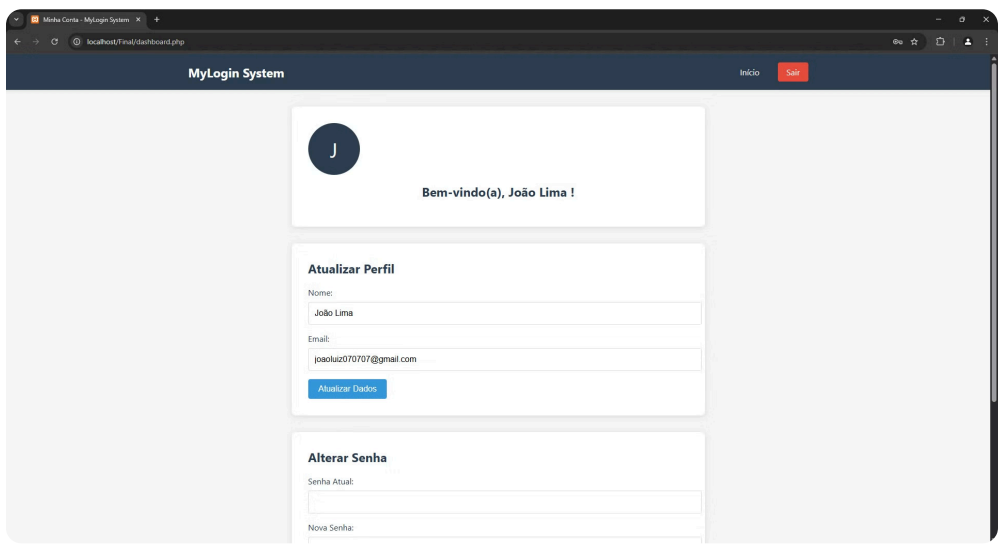
Página de Login



Página de Registro



Dashboard



Conclusão

Objetivos Alcançados:

- Sistema de login seguro implementado
- Interface responsiva e moderna
- Validações client e server-side
- Proteção contra vulnerabilidades comuns
- Upload e gerenciamento de foto de perfil
- Documentação completa

Melhorias Futuras:

- Implementação de recuperação de senha
- Autenticação em duas etapas
- Integração com redes sociais
- Histórico de login
- Tema escuro
- API REST

Referências: PHP Documentation, MySQL Documentation, MDN Web Docs, OWASP Security Guidelines, Material didático de cursos e livros sobre desenvolvimento web.