Análise Técnica do Código

Universidade do Algarve

Tecnologias web - Sistemas e tecnologia de informação

Laura Kayamori

Maria Eduarda Pereira

O relatório descreve o desenvolvimento do projeto "Arca Digital", uma aplicação web que simula um servidor FTP.

Ele inclui funcionalidades como carregamento, visualização e transferência de arquivos, além de focar na segurança com autenticação de usuários.

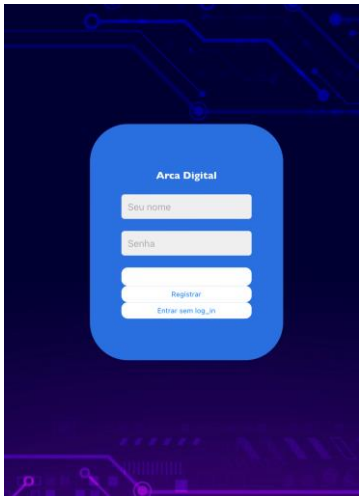
O conteúdo será estruturado em seções que examinam brevemente cada código desenvolvido, iniciando pelo desenvolvimento do usuário dentro da plataforma, para a prática de upload e publicação dos ficheiros.

1.1 Log_in.HTML

Este código corresponde à implementação da página de login da aplicação dentro de uma estrutura HTML. A página possibilita que os usuários insiram suas credenciais para realizar login, registrar-se ou acessar sem senha, com a lógica incorporada para permitir o acesso público a documentos, caso o usuário opte por isso, dispensando a necessidade de senha. Para facilitar a manipulação do DOM e as operações AJAX, é utilizado jQuery, além da biblioteca jQuery Cookie para manipulação de cookies.

Manipulação de Document Ready: Utiliza `$(document).ready` para garantir que o DOM esteja totalmente carregado antes de executar scripts.

Redirecionamento de Botões: Configura ações para os botões de registrar e entrar sem login para redirecionar para as páginas apropriadas (registrar.php)



11.a Registrar.html

Este documento é a página de registro, apresentando um formulário para os usuários inserirem seu nome e senha. Após submeterem os dados, uma requisição é enviada para o servidor, processada pelo arquivo registrar.php. Se o registro for bem-sucedido, o usuário é redirecionado para a página de login.

1.2 Registrar.php

Este trecho de código é responsável por registrar um novo usuário no banco de dados com base nas informações fornecidas no formulário de registro. Ele verifica se o método de requisição é POST e se os campos de usuário e senha não estão vazios. Em seguida, prepara uma instrução SQL para inserir os dados na tabela de usuários. Se a inserção for bem-sucedida, o usuário é redirecionado para a página de login (log_in.html); caso contrário, uma mensagem de erro é retornada. Ao final, a conexão com o banco de dados é fechada.

1.3 ConectDB.php

O terceiro código realiza a conexão com o banco de dados MySQL utilizando as credenciais fornecidas (host, nome de usuário, senha e nome do banco de dados). Em seguida, verifica se a conexão foi bem-sucedida.

Além disso, define uma função chamada formatar_data que recebe uma data no formato “YYYY-MM-DD” e a formata para o formato “DD/MM/YYYY”.

1.4 Verifica_senha.php

Em seguida, temos o código que inicia uma sessão e requer um arquivo de conexão com o banco de dados.

Ele define uma senha correta e verifica se a requisição é do tipo POST. Se for, compara a senha fornecida pelo usuário com a senha correta. Se as senhas coincidirem, retorna um status de sucesso; caso contrário, retorna um status de erro indicando que a senha está incorreta.

Se a requisição não for do tipo POST, retorna um status de erro indicando que o método de requisição é inválido.

1.5 Senha.php

Mantendo continuidade no processo, esse arquivo verifica as credenciais de login dos usuários em um banco de dados MySQL. Primeiro, ele recebe os dados do formulário de login (nome de usuário e senha) por meio de uma requisição POST.

Em seguida, ele executa uma consulta SQL para verificar se existe um usuário com as credenciais fornecidas. Se encontrar um usuário correspondente, ele redireciona para a página index.php; caso contrário, redireciona de volta para a página de login em html

2.1 Upload_privado.php

Neste código PHP, estamos lidando com o processo de envio de arquivos para um servidor e registrando esses arquivos em um banco de dados. Quando um arquivo é enviado através de um formulário HTML, primeiro verificamos se ele é do tipo permitido, como imagens (JPG, JPEG, PNG, GIF), documentos (PDF, DOCX) e outros.

Depois de confirmar o tipo do arquivo, ele é movido para uma pasta específica no servidor, com um nome único gerado aleatoriamente para evitar problemas de nomenclatura. Isso é importante para manter os arquivos organizados e seguros.

Em seguida, os detalhes do arquivo, como nome, caminho e data de envio, são salvos no banco de dados. Isso nos ajuda a manter um registro de todos os arquivos enviados e fornece acesso a esses arquivos quando necessário.

O código também inclui verificações para lidar com possíveis problemas durante o processo de envio, como falhas no envio do arquivo ou erros ao salvar os detalhes do arquivo no banco de dados. Essas verificações garantem que o sistema funcione de maneira confiável. Além disso, há uma medida de segurança implementada para evitar que o formulário seja enviado novamente acidentalmente, redirecionando o usuário após um envio bem-sucedido.

2.3upload_publico.php

No código PHP fornecido, o foco principal é o processo de envio de arquivos para um servidor e o subsequente registro desses arquivos em um banco de dados. Em comparação com o código anterior, upload_privado.php, este script lida com o envio de arquivos para uma pasta de uploads padrão, sem a distinção de arquivos privados.

Quando um arquivo é enviado por meio de um formulário HTML, o script verifica se ocorreram falhas durante o processo de envio, como erros no arquivo ou tipos de arquivos não permitidos. Isso é realizado através de uma série de verificações, incluindo a verificação do tipo de arquivo com base em uma lista de extensões permitidas.

Após uma verificação bem-sucedida, o arquivo é movido para uma pasta específica no servidor, utilizando um nome único gerado aleatoriamente para evitar conflitos de nomenclatura. Em seguida, os detalhes do arquivo, como nome, caminho e data de envio, são registrados no banco de dados.

2.3 retrive_files.php

Essencialmente, esta parte em PHP consulta o banco de dados, utilizando o arquivo conectDB.php para obter informações sobre os arquivos armazenado, executa uma consulta SQL para selecionar todos os registros da tabela arquivos, armazenando o resultado na variável $sql_arquivos, e formata essas informações em uma tabela HTML, pronta para ser exibida na página da web.

A função listarArquivos é responsável por gerar o HTML da tabela.

```
5
6   function listarArquivos($sql_arquivos) {
7       $lista = '';
8       while($arquivo = $sql_arquivos->fetch_assoc()) {
9           $lista .= '<tr>';
10          $lista .= '<td><a target="_blank" href="' .
    $arquivo['path'] . '">' . $arquivo['nome'] . '</a></
    td>';
11          $lista .= '<td>' . date("d/m/Y H:i", strtotime
    ($arquivo['data_upload'])) . '</td>';
12          $lista .= '<td><a href="download.php?file=' .
    urlencode($arquivo['path']) . '">Baixar</a></td>';
13          $lista .= '</tr>';
14      }
15      return $lista;
16  }
```

Ela recebe o resultado da consulta SQL como parâmetro e inicia uma string vazia chamada $lista para armazenar as linhas da tabela.

A função percorre cada registro retornado pela consulta usando um loop `while`. Para cada registro, ela cria uma linha da tabela com três células. A primeira célula contém um link para visualizar o arquivo, a segunda exibe a data de upload formatada, e a terceira oferece um link para baixar o arquivo. Após processar todos os registros, a função retorna a string `$lista` com o HTML gerado.

Se houver um erro na execução da consulta, o script exibe uma mensagem de erro e interrompe a execução.

2.4 retrive_files_priv.php

No código PHP apresentado, há um formulário inserido em cada linha da tabela HTML, o que permite aos usuários inserir uma senha antes de baixar um arquivo.

Esse formulário é gerado através de um loop enquanto percorre os resultados da consulta ao banco de dados, usando o método `fetch_assoc`. Esse método é responsável por recuperar cada linha da tabela do banco de dados como uma matriz associativa, o que facilita o acesso aos valores usando os nomes das colunas como chaves de índice.

```
function listarArquivosPriv($sql_arquivos) {
    $lista = '';
    while($arquivo = $sql_arquivos->fetch_assoc()) {
        $lista .= '<tr>';
        $lista .= '<td><a target="_blank" href="' .
$arquivo['path'] . '">' . $arquivo['nome'] . '</a></
td>';
        $lista .= '<td>' . date("d/m/Y H:i", strtotime
($arquivo['data_upload'])) . '</td>';
```

Assim, o formulário proporciona uma experiência interativa e segura, garantindo que os usuários só possam baixar arquivos privados após inserir a senha correta.

3. Arcadigital.sql

Esse dump SQL foi gerado pelo phpMyAdmin e serve para definir e preencher um banco de dados chamado `arcadigital`. Sendo suas configurações iniciais a definição do conjunto de caracteres para `utf8mb4` para garantir que caracteres Unicode sejam suportados; criação de tabelas `arquivos`, `arquivospriv`, e `usuarios`.

Além disso define as chaves primarias para as tabelos pelo AUTO_INCREMENT garantido uma configuração automática das colunas

Finaliza-se com “COMMIT” para restaurar as configurações anteriores de conjunto de caracteres.

Este script é super importante para fazer backup, restaurar e migrar dados, garantindo que o banco de dados possa ser recriado de forma consistente e confiável.