
Cadastro de Cliente com Upload

Explicação do tutorial

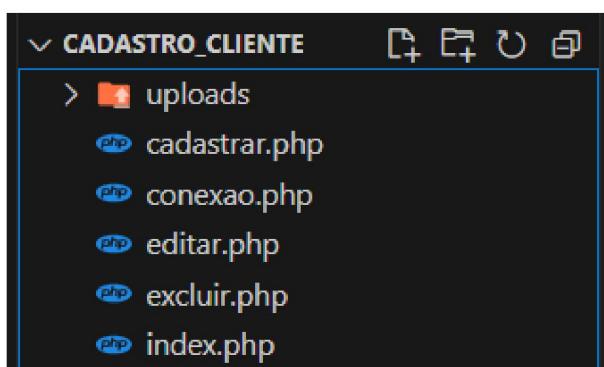
Neste método de CRUD (Create, Read, Update, Delete), vamos utilizar uma abordagem clássica para a conexão com o banco de dados e a estruturação dos arquivos. Não vamos nos preocupar, neste momento, com a separação entre front-end e back-end, nem com a utilização de orientação a objetos. Vamos focar em uma implementação simples e direta.

1. Conexão com o Banco de Dados:

- Utilizaremos a extensão PDO (PHP Data Objects) para conectar ao banco de dados MySQL.
- A conexão será feita de forma direta, sem a criação de classes ou métodos específicos para isso.

2. Estrutura dos Arquivos:

- **conexao.php**: Contém o código para estabelecer a conexão com o banco de dados.
- **index.php**: Exibe a lista de clientes e permite ações como cadastrar, editar e excluir.
- **cadastrar.php**: Formulário para adicionar novos clientes ao banco de dados.
- **editar.php**: Formulário para editar os dados de um cliente existente.
- **excluir.php**: Script para excluir um cliente do banco de dados.



3. CRUD Simples:

- **Create (Criar)**: Adicionar novos registros ao banco de dados.

- **Read (Ler):** Exibir os registros existentes.
- **Update (Atualizar):** Modificar os registros existentes.
- **Delete (Excluir):** Remover registros do banco de dados.

Objetivo

O objetivo é entender como funciona a interação básica entre PHP e MySQL, utilizando uma estrutura de arquivos simples e direta. Esta abordagem é ideal para quem está começando e quer aprender os fundamentos antes de avançar para conceitos mais complexos como MVC (Model-View-Controller) ou orientação a objetos.

Criação do banco

```
CREATE DATABASE sistema_clientes;

USE sistema_clientes;

CREATE TABLE clientes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    arquivo_pdf VARCHAR(255) NOT NULL
);
```

conexao.php

```
<?php
    $host = "localhost";
    $dbname = "sistema_clientes";
    $username = "root";
    $password = "";

    try {
        $conn = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
        $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    } catch (PDOException $e) {
        die("Erro na conexão: " . $e->getMessage());
    }
}
```

Explicação do Código - conexao.php

1. Declaração das Variáveis de Conexão:

- **\$host**: Define o endereço do servidor de banco de dados. Aqui, localhost indica que o servidor está rodando na mesma máquina que o script PHP.
- **\$dbname**: Nome do banco de dados que será utilizado, neste caso, sistema_clientes.
- **\$username**: Nome de usuário para acessar o banco de dados, aqui é root.
- **\$password**: Senha correspondente ao usuário. Está vazia, o que é comum em ambientes de desenvolvimento, mas não recomendado em produção.

2. Bloco `try-catch` para Conexão com o Banco de Dados:

- **`try`**: Tenta executar o código dentro do bloco. Se ocorrer um erro, ele será capturado pelo bloco catch.
- **`new PDO(...)`**: Cria uma nova instância da classe PDO para a conexão com o banco de dados MySQL. Os parâmetros são a string de conexão (mysql:host=\$host;dbname=\$dbname), o nome de usuário e a senha.
- **`\$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION)`**: Configura o modo de erro do PDO para lançar exceções. Isso facilita a depuração, pois os erros serão lançados como exceções.
- **`catch (PDOException \$e)`**: Captura exceções do tipo PDOException que podem ocorrer durante a tentativa de conexão.
- **`die("Erro na conexão: " . \$e->getMessage())`**: Exibe uma mensagem de erro e termina a execução do script se ocorrer uma exceção.

O arquivo conexão.php é essencial para estabelecer a conexão com o banco de dados, que será utilizada em outros scripts PHP para realizar operações como inserção, atualização, exclusão e consulta de dados.

index.php

```
<?php
include 'conexao.php';

$clientes = $conn->query("SELECT * FROM clientes")->fetchAll(PDO::FETCH_ASSOC);
?>

<!DOCTYPE html>
<html lang="pt-br">
```

```
<head>
    <meta charset="UTF-8">
    <title>Lista de Clientes</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>

<body>
    <div class="container">
        <h1 class="mt-4">Lista de Clientes</h1>
        <a href="cadastrar.php" class="btn btn-success mb-3">Cadastrar Cliente</a>
        <table class="table table-bordered">
            <thead>
                <tr>
                    <th>Nome</th>
                    <th>Email</th>
                    <th>Arquivo PDF</th>
                    <th>Ações</th>
                </tr>
            </thead>
            <tbody>
                <?php foreach ($clientes as $cliente): ?>
                <tr>
                    <td><?= htmlspecialchars($cliente['nome']) ?></td>
                    <td><?= htmlspecialchars($cliente['email']) ?></td>
                    <td><a href="uploads/<?= htmlspecialchars($cliente['arquivo_pdf']) ?>" target="_blank">Ver PDF</a></td>
                    <td>
                        <a href="editar.php?id=<?= $cliente['id'] ?>" class="btn btn-warning">Editar</a>
                        <a href="excluir.php?id=<?= $cliente['id'] ?>" class="btn btn-danger" onclick="return confirm('A exclusão será permanente! Tem certeza disso?')">Excluir</a>
                    </td>
                </tr>
            <?php endforeach; ?>
        </tbody>
    </div>
</body>

</html>
```

Nome	Email	Arquivo PDF	Ações
Reginaldo Cândido	reginaldocandido@gmail.com	Ver PDF	Editar Excluir
Maria Genoveva	maria@gmail.com	Ver PDF	Editar Excluir

Explicação do Código

1. Inclusão do Arquivo de Conexão:

- Este comando inclui o arquivo conexao.php, que contém o código para estabelecer a conexão com o banco de dados.

2. Consulta ao Banco de Dados:

- `\$conn->query("SELECT * FROM clientes")`: Executa uma consulta SQL para selecionar todos os registros da tabela clientes.
- `fetchAll(PDO::FETCH_ASSOC)`: Recupera todos os registros retornados pela consulta e os armazena em um array associativo.

3. Estrutura HTML:

- Cabeçalho HTML (`<head>`):**
 - Define o charset como UTF-8 e inclui o CSS do Bootstrap para estilização.
- Corpo HTML (`<body>`):**
 - Container Bootstrap:** Cria um container centralizado.
 - Título e Botão de Cadastro:** Exibe o título "Lista de Clientes" e um botão para cadastrar novos clientes.
 - Tabela de Clientes:**
 - Cabeçalho da Tabela (`<thead>`):** Define os títulos das colunas: Nome, Email, Arquivo PDF e Ações.
 - Corpo da Tabela (`<tbody>`):** Itera sobre o array \$clientes e cria uma linha (`<tr>`) para cada cliente.
 - `htmlspecialchars(\$cliente['campo'])` : Escapa caracteres especiais para evitar ataques XSS.
 - Link para Visualizar PDF:** Cria um link para visualizar o

arquivo PDF do cliente.

- **Botões de Ação:** Inclui botões para editar e excluir o cliente, com confirmação para a exclusão.

Estamos usando o index.php para exibir uma lista de clientes recuperados do banco de dados e permite ações como visualizar, editar e excluir registros.

cadastrar.php

```
<?php
include 'conexao.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $nome = $_POST['nome'];
    $email = $_POST['email'];
    $arquivo = $_FILES['arquivo_pdf'];

    if ($arquivo['error'] === UPLOAD_ERR_OK) {
        $nomeArquivo = uniqid() . "-" . $arquivo['name'];
        move_uploaded_file($arquivo['tmp_name'], "uploads/$nomeArquivo");

        $stmt = $conn->prepare("INSERT INTO clientes (nome, email, arquivo_pdf) VA-
LUES (?, ?, ?)");
        $stmt->execute([$nome, $email, $nomeArquivo]);

        header("Location: index.php");
        exit;
    }
}

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <title>Cadastrar Cliente</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>

<body>
    <div class="container">
        <h1 class="mt-4">Cadastrar Cliente</h1>
        <form method="POST" enctype="multipart/form-data">
            <div class="form-group">
                <label for="nome">Nome</label>
                <input type="text" class="form-control" name="nome" required>
            </div>
        </form>
    </div>
</body>
```

```
<div class="form-group">
    <label for="email">Email</label>
    <input type="email" class="form-control" name="email" required>
</div>
<div class="form-group">
    <label for="arquivo_pdf">Arquivo PDF Assinado</label>
    <input type="file" class="form-control" name="arquivo_pdf" accept=".pdf" required>
</div>
<button type="submit" class="btn btn-primary">Cadastrar</button>
</form>
</div>
</body>
</html>
```

Explicação do Código

1. Inclusão do Arquivo de Conexão:

- Este comando inclui o arquivo conexao.php, que contém o código para estabelecer a conexão com o banco de dados.

2. Verificação do Método de Requisição:

- Verifica se o método de requisição é POST, o que indica que o formulário foi enviado.

3. Captura dos Dados do Formulário:

- Captura os dados enviados pelo formulário: nome, email e o arquivo PDF.

4. Verificação e Upload do Arquivo:

- Verifica se o upload do arquivo foi bem-sucedido (UPLOAD_ERR_OK).
- Gera um nome único para o arquivo usando uniqid() para evitar conflitos de nomes.
- Move o arquivo para a pasta uploads/.

5. Inserção dos Dados no Banco de Dados:

- Prepara uma instrução SQL para inserir os dados do cliente na tabela clientes.
- Executa a instrução com os valores capturados do formulário.

6. Redirecionamento Após Cadastro:

- Redireciona o usuário para a página index.php após o cadastro bem-sucedido.
- exit garante que o script pare de executar após o redirecionamento.

Estrutura HTML

7. Cabeçalho HTML (`<head>`):

- Define o charset como UTF-8 e inclui o CSS do Bootstrap para estilização.

8. Corpo HTML (`<body>`):

- **Container Bootstrap:** Cria um container centralizado.
- **Formulário de Cadastro:** Contém campos para nome, email e upload de arquivo PDF.
 - `enctype="multipart/form-data"`: Necessário para permitir o upload de arquivos.
 - **Campos do Formulário:** Cada campo é um grupo de formulário (form-group) com um rótulo (label) e um campo de entrada (input).
 - **Botão de Envio:** Um botão para enviar o formulário.

O arquivo cadastrar.php permite que os usuários cadastrem novos clientes, incluindo o upload de um arquivo PDF, e insere esses dados no banco de dados.

The screenshot shows a web page titled "Cadastrar Cliente". It contains three input fields: "Nome" with the value "Walter de Castro", "Email" with the value "waltercastro@gmail.com", and a file input field labeled "Arquivo PDF Assinado" which displays the path "Escolher arquivo - Al day - Do zero ao primeiro bot em um dia.pdf". Below these fields is a blue "Cadastrar" button.

editar.php

```
<?php
include 'conexao.php';

$id = $_GET['id'];
$cliente = $conn->prepare("SELECT * FROM clientes WHERE id = ?");
$cliente->execute([$id]);
$cliente = $cliente->fetch(PDO::FETCH_ASSOC);
```

```

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $nome = $_POST['nome'];
    $email = $_POST['email'];
    $arquivoNovo = $_FILES['arquivo_pdf'];

    if ($arquivoNovo['error'] === UPLOAD_ERR_OK) {
        $nomeArquivoNovo = uniqid() . "-" . $arquivoNovo['name'];
        move_uploaded_file($arquivoNovo['tmp_name'], "uploads/$nomeArquivoNovo");

        // Remove o arquivo antigo
        if (file_exists("uploads/" . $cliente['arquivo_pdf'])) {
            unlink("uploads/" . $cliente['arquivo_pdf']);
        }

        $stmt = $conn->prepare("UPDATE clientes SET nome = ?, email = ?, arquivo_pdf = ? WHERE id = ?");
        $stmt->execute([$nome, $email, $nomeArquivoNovo, $id]);
    } else {
        $stmt = $conn->prepare("UPDATE clientes SET nome = ?, email = ? WHERE id = ?");
        $stmt->execute([$nome, $email, $id]);
    }

    header("Location: index.php");
    exit;
}

?>

<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <title>Editar Cliente</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>

<body>
    <div class="container">
        <h1 class="mt-4">Editar Cliente</h1>
        <form method="POST" enctype="multipart/form-data">
            <div class="form-group">
                <label for="nome">Nome</label>
                <input type="text" class="form-control" name="nome" value="= htmlspecialchars($cliente['nome']) ?" required>
            </div>
            <div class="form-group">
                <label for="email">Email</label>

```

```
        <input type="email" class="form-control" name="email" value="<?=
htmlspecialchars($cliente['email']) ?>" required>
    </div>
    <div class="form-group">
        <label for="arquivo_pdf">Arquivo PDF Assinado (deixe em branco se
não quiser alterar)</label>
        <input type="file" class="form-control" name="arquivo_pdf" ac-
cept=".pdf">
    </div>
    <button type="submit" class="btn btn-primary">Salvar Alterações</but-
ton>
</form>
</div>
</body>
</html>
```

Explicação do Código

1. Inclusão do Arquivo de Conexão:

- Este comando inclui o arquivo conexao.php, que contém o código para estabelecer a conexão com o banco de dados.

2. Captura do ID do Cliente:

- Captura o ID do cliente a ser editado a partir da URL.

3. Consulta ao Banco de Dados para Obter os Dados do Cliente:

- Prepara uma consulta SQL para selecionar o cliente com o ID especificado.
- Executa a consulta e armazena os dados do cliente em um array associativo.

4. Verificação do Método de Requisição:

- Verifica se o método de requisição é POST, o que indica que o formulário foi enviado.

5. Captura dos Dados do Formulário:

- Captura os dados enviados pelo formulário: nome, email e o novo arquivo PDF (se houver).

6. Verificação e Upload do Novo Arquivo:

- Verifica se um novo arquivo foi enviado (UPLOAD_ERR_OK).

- Gera um nome único para o novo arquivo e move-o para a pasta uploads/.
- Remove o arquivo antigo se ele existir.
- Atualiza os dados do cliente no banco de dados, incluindo o novo arquivo PDF.
- Se nenhum novo arquivo foi enviado, atualiza apenas o nome e o email do cliente.

7. Redirecionamento Após Edição:

- Redireciona o usuário para a página index.php após a edição bem-sucedida.
- exit garante que o script pare de executar após o redirecionamento.

Estrutura HTML

8. Cabeçalho HTML (`<head>`):

- Define o charset como UTF-8 e inclui o CSS do Bootstrap para estilização.

9. Corpo HTML (`<body>`):

- **Container Bootstrap:** Cria um container centralizado.
- **Formulário de Edição:** Contém campos para nome, email e upload de novo arquivo PDF (opcional).
 - `enctype="multipart/form-data"`: Necessário para permitir o upload de arquivos.
 - **Campos do Formulário:** Cada campo é um grupo de formulário (form-group) com um rótulo (label) e um campo de entrada (input).
 - **Botão de Envio:** Um botão para enviar o formulário.

O arquivo editar.php permite que os usuários editem os dados de um cliente existente, incluindo a opção de atualizar o arquivo PDF associado.

excluir.php

```
<?php
include 'conexao.php';

$id = $_GET['id'];
$cliente = $conn->prepare("SELECT arquivo_pdf FROM clientes WHERE id = ?");
$cliente->execute([$id]);
$cliente = $cliente->fetch(PDO::FETCH_ASSOC);
```

```
// Exclui o arquivo PDF do servidor
if (file_exists("uploads/" . $cliente['arquivo_pdf'])) {
    unlink("uploads/" . $cliente['arquivo_pdf']);
}

$stmt = $conn->prepare("DELETE FROM clientes WHERE id = ?");
$stmt->execute([$id]);

header("Location: index.php");
exit;
```

Explicação do Código

1. Inclusão do Arquivo de Conexão:

- Este comando inclui o arquivo conexao.php, que contém o código para estabelecer a conexão com o banco de dados.

2. Captura do ID do Cliente:

- Captura o ID do cliente a ser excluído a partir da URL.

3. Consulta ao Banco de Dados para Obter o Nome do Arquivo PDF:

- Prepara uma consulta SQL para selecionar o campo arquivo_pdf do cliente com o ID especificado.
- Executa a consulta e armazena o resultado em um array associativo.

4. Exclusão do Arquivo PDF do Servidor:

- Verifica se o arquivo PDF existe no diretório uploads/.
- Se o arquivo existir, ele é excluído usando a função unlink().

5. Exclusão do Registro do Cliente no Banco de Dados:

- Prepara uma instrução SQL para excluir o cliente com o ID especificado.
- Executa a instrução SQL.

6. Redirecionamento Após Exclusão:

- Redireciona o usuário para a página index.php após a exclusão bem-sucedida.
- exit garante que o script pare de executar após o redirecionamento.

O arquivo excluir.php realiza a exclusão de um cliente específico. Ele primeiro remove o arquivo PDF associado ao cliente do servidor e, em seguida, exclui o registro do cliente do banco de dados. Após a exclusão, o usuário é redirecionado de volta para a página principal.