**CRUD**

**CONFIGURAÇÃO PHP**

Este código em PHP lida com a configuração e a criação de uma conexão com um banco de dados MySQL.

**<?php**

As primeiras linhas declaram as configurações de conexão com o banco de dados, como o nome do host, nome de usuário, senha e nome do banco de dados:

**// Configurações de conexão com o banco de dados**

**$host = 'localhost';**

**$username = 'root';**

**$password = '';**

**$dbname = 'dynatrace';**

**// Conexão com o banco de dados**

Em seguida, o código cria uma nova instância da classe mysql para estabelecer a conexão com o banco de dados. A função new mysql() recebe as configurações de conexão como argumentos: host, nome de usuário, senha e nome do banco de dados. Esses parâmetros são usados para estabelecer uma conexão com o banco de dados especificado.

**$conn = new mysql($host, $username, $password, $dbname);**

**// Verifica se houve algum erro na conexão**

A próxima linha verifica se houve algum erro na conexão. A propriedade connect\_error do objeto de conexão ($conn) é verificada para determinar se ocorreu algum problema durante a conexão.

**if ($conn->connect\_error) {**

Se houver um erro, a função die() é chamada com uma mensagem de erro, que será exibida no navegador, e a execução do script será interrompida.

**die('Erro na conexão com o banco de dados: ' . $conn->connect\_error);}?>**

Portanto, esse código tem a finalidade de estabelecer uma conexão com um banco de dados MySQL usando as configurações fornecidas. Se a conexão for bem-sucedida, você poderá usar o objeto $conn para executar consultas e interagir com o banco de dados. Caso contrário, se ocorrer algum erro na conexão, uma mensagem de erro será exibida e a execução será interrompida.

**CREATE**

Este código PHP lida com a criação de um registro de cliente em um banco de dados MySQL e exibe um formulário HTML para inserir os dados.

A linha <?php include\_once 'menu.php'; ?> inclui o conteúdo do arquivo menu.php no código, o que sugere que há um menu sendo adicionado antes do formulário. O conteúdo do menu.php não está presente no código fornecido, mas presumimos que ele define o menu de navegação do site:

**<?php include\_once 'menu.php'; ?>**

**<?php**

A linha require\_once 'config.php'; inclui o arquivo config.php, que contém as configurações de conexão com o banco de dados. O uso de require\_once garante que o arquivo seja incluído apenas uma vez:

**require\_once 'config.php';**

A classe Cliente é definida para representar um cliente com propriedades como nome, email, senha, CNPJ e data POC. Ela possui um método construtor para inicializar os valores das propriedades e métodos getter e setter para acessar e modificar as propriedades:

**class Cliente**

**{**

**// Propriedades da classe Cliente**

**private $nome;**

**private $email;**

**private $senha;**

**private $cnpj;**

**private $dataPoc;**

**// Método construtor da classe Cliente**

**public function \_\_construct($nome, $email, $senha, $cnpj, $dataPoc)**

**{**

**$this->nome = $nome;**

**$this->email = $email;**

**$this->senha = $senha;**

**$this->cnpj = $cnpj;**

**$this->dataPoc = $dataPoc;**

**}**

**// Métodos getter para obter os valores das propriedades**

**public function getNome()**

**{**

**return $this->nome;**

**}**

**public function getEmail()**

**{**

**return $this->email;**

**}**

**public function getSenha()**

**{**

**return $this->senha;**

**}**

**public function getCnpj() {**

**return $this->cnpj; }**

**public function getDataPoc() {**

**return $this->dataPoc; }**

**// Método setter para definir o valor da propriedade senha**

**public function setSenha($senha) {**

**$this->senha = $senha; }**

**// Método para criar um registro de cliente no banco de dados**

O método criarRegistro da classe Cliente é responsável por inserir um novo registro de cliente no banco de dados. Ele utiliza a conexão global $conn (definida no arquivo config.php) para preparar e executar uma query INSERT para adicionar o registro com os valores das propriedades do objeto Cliente:

**public function criarRegistro() {**

**global $conn;**

**// Prepara e executa a query para inserir o registro**

O resultado da inserção é verificado e uma mensagem apropriada é exibida:

**$stmt = $conn->prepare('INSERT INTO cliente (nome, email, senha, cnpj, dataPoc) VALUES (?, ?, ?, ?, ?)');**

**$stmt->bind\_param('sssss', $this->nome, $this->email, $this->senha, $this->cnpj, $this->dataPoc);**

**$stmt->execute();**

**// Verifica se o registro foi criado com sucesso**

**if ($stmt->affected\_rows > 0) {**

**echo 'Registro criado com sucesso!';**

**} else {**

**echo 'Erro ao criar o registro.'; }**

**$stmt->close(); }**

**}**

**// Verifica se o método da requisição é POST**

O código verifica se a requisição é do tipo POST usando $\_SERVER['REQUEST\_METHOD'] === 'POST'. Isso indica que o formulário foi submetido:

**if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {**

**// Verifica se os campos foram preenchidos**

Dentro da verificação do tipo de requisição POST, o código verifica se todos os campos do formulário (nome, email, senha, cnpj e data\_poc) foram preenchidos. Se todos os campos estiverem preenchidos, os valores são obtidos a partir do $\_POST e um novo objeto Cliente é criado com esses valores. Em seguida, o método criarRegistro é chamado para inserir o registro no banco de dados:

**if (!empty($\_POST['nome']) && !empty($\_POST['email']) && !empty($\_POST['senha']) && !empty($\_POST['cnpj']) && !empty($\_POST['data\_poc'])) {**

**$nome = $\_POST['nome'];**

**$email = $\_POST['email'];**

**$senha = $\_POST['senha'];**

**$cnpj = $\_POST['cnpj'];**

**$dataPoc = $\_POST['data\_poc'];**

**// Cria uma instância da classe Cliente com os dados fornecidos**

**$cliente = new Cliente($nome, $email, $senha, $cnpj, $dataPoc);**

**// Chama o método criarRegistro para inserir o registro no banco de dados**

**$cliente->criarRegistro();**

**} else {**

Se algum campo estiver vazio, uma mensagem de erro é exibida solicitando o preenchimento de todos os campos:

**echo 'Por favor, preencha todos os campos.'; }**

**}?>**

O código HTML exibe um formulário com os campos necessários para criar um novo registro de cliente:

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Criar Registro</title>**

O arquivo img/style.css é incluído para fornecer estilos ao formulário e ao layout da página:

**<link rel="stylesheet" href="img/style.css">**

**</head>**

**<body>**

Os campos estão vinculados ao método POST e a ação do formulário é definida como create.php:

**<h2>Criar Registro</h2>**

**<form method="POST" action="create.php">**

**<!-- Campos do formulário para criação de registro -->**

Os campos são nome, email, senha, cnpj e data\_poc:

**<label for="nome">Nome:</label>**

**<input type="text" name="nome" required><br>**

**<label for="email">Email:</label>**

**<input type="email" name="email" required><br>**

**<label for="senha">Senha:</label>**

**<input type="password" name="senha" required><br>**

**<label for="cnpj">CNPJ:</label>**

**<input type="text" name="cnpj" required><br>**

**<label for="data\_poc">Data POC:</label>**

**<input type="date" name="data\_poc" required><br>**

**<!-- Botões de envio e cancelamento -->**

Além disso, existem botões para enviar o formulário e cancelar a operação. O botão de envio possui o valor "Salvar":

**<input type="submit" value="Salvar">**

O botão de cancelamento redireciona o usuário para index.php:

**<button onclick="location.href='index.php'">Cancelar</button>**

**<!-- Botão para excluir o registro -->**

Também há um botão "Deletar Registro" que redireciona o usuário para delete.php, para excluir o registro:

**<button onclick="location.href='delete.php'">Deletar Registro</button>**

**</form>**

**</body>**

**</html>**

Em resumo, o código PHP é responsável pela criação de um registro de cliente no banco de dados, validando os dados enviados por meio de um formulário HTML. O layout criado consiste em um formulário com campos para inserir os dados do cliente e botões para enviar, cancelar e excluir o registro.

**READ**

Este código PHP é responsável por listar registros de clientes armazenados em um banco de dados e exibi-los em uma tabela HTML.

A linha <?php include\_once 'menu.php'; ?> inclui o conteúdo do arquivo menu.php no código. Isso demonstra que há um menu de navegação sendo adicionado antes da tabela de registros:

**<?php include\_once 'menu.php'; ?>**

**<?php**

A linha require\_once 'config.php'; inclui o arquivo config.php, que contém as configurações de conexão com o banco de dados. O uso de require\_once garante que o arquivo seja incluído apenas uma vez.

**require\_once 'config.php';**

A classe Cliente é definida para representar um cliente com propriedades como id, nome, cnpj, email, senha e dataPoc.

Além disso, há um método getDataPocFormatted que retorna a data de POC formatada no formato "dia/mês/ano".

**class Cliente**

**{**

**private $id;**

**private $nome;**

**private $cnpj;**

**private $email;**

**private $senha;**

**private $dataPoc;**

A classe Cliente possui um método construtor para inicializar os valores das propriedades e métodos getter para acessar as propriedades:

**public function \_\_construct($id, $nome, $cnpj, $email, $senha, $dataPoc)**

**{**

**$this->id = $id;**

**$this->nome = $nome;**

**$this->cnpj = $cnpj;**

**$this->email = $email;**

**$this->senha = $senha;**

**$this->dataPoc = $dataPoc;**

**}**

**public function getId()**

**{**

**return $this->id;**

**}**

**public function getNome()**

**{**

**return $this->nome;**

**}**

**public function getCnpj()**

**{**

**return $this->cnpj;**

**}**

**public function getEmail()**

**{**

**return $this->email;**

**}**

**public function getSenha()**

**{**

**return $this->senha;**

**}**

**public function getDataPocFormatted()**

**{**

**$dataPocTimestamp = strtotime($this->dataPoc);**

**return date('d/m/Y', $dataPocTimestamp);    }**

**}**

**// Executa a query para buscar os registros**

É executada uma query SQL para buscar todos os registros da tabela cliente:

**$query = 'SELECT \* FROM cliente';**

O resultado é armazenado na variável $result:

**$result = $conn->query($query);**

**$clientes = [];**

Um loop while é usado para iterar sobre cada linha do resultado da query:

**while ($row = $result->fetch\_assoc()) {**

Em cada iteração, um novo objeto Cliente é criado com os valores das colunas do resultado e adicionado à array $clientes:

**$cliente = new Cliente($row['id'], $row['nome'], $row['cnpj'], $row['email'], $row['senha'], $row['dataPoc']);**

Uma array vazia $clientes é criada para armazenar os objetos Cliente:

**$clientes[] = $cliente;}**

**?>**

O código HTML exibe uma tabela para listar os registros de clientes:

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Listar Registros</title>**

**</head>**

**<body>**

**<h2>Listar Registros</h2>**

**<table>**

**<tr>**

As colunas são "ID", "Nome", "Email" e "Data POC":

**<th>ID</th>**

**<th>Nome</th>**

**<th>Email</th>**

**<th>Data POC</th>**

**</tr>**

Cada registro é exibido em uma linha da tabela usando um loop foreach que itera sobre a array $clientes:

**<?php foreach ($clientes as $cliente) : ?>**

**<tr>**

Dentro do loop foreach, os métodos getter dos objetos Cliente são chamados para obter os valores das propriedades e exibi-los nas células da tabela:

**<td><?php echo $cliente->getId(); ?></td>**

**<td><?php echo $cliente->getNome(); ?></td>**

**<td><?php echo $cliente->getEmail(); ?></td>**

**<td><?php echo $cliente->getDataPocFormatted(); ?></td>**

**</tr>**

**<?php endforeach; ?>**

**</table>**

**</body>**

**</html>**

Em resumo, o código PHP busca registros de clientes em um banco de dados e os exibe em uma tabela HTML. Cada registro é representado por um objeto Cliente e as propriedades do objeto são exibidas nas células da tabela. O layout criado consiste em uma tabela com colunas para exibir as informações dos clientes.

**UPDATE**

A funcionalidade de atualização permite modificar os dados de um cliente existente no sistema, fornecendo um formulário preenchido com os dados atuais do cliente e permitindo que sejam realizadas alterações.

A linha <?php include\_once 'menu.php'; ?> inclui o conteúdo do arquivo menu.php no código. Isso demonstra que há um menu de navegação sendo adicionado antes do formulário de atualização de registro:

**<?php include\_once 'menu.php'; ?>**

**<?php**

A linha require\_once 'config.php'; inclui o arquivo config.php, que contém as configurações de conexão com o banco de dados. O uso de require\_once garante que o arquivo seja incluído apenas uma vez:

**require\_once 'config.php';**

A partir daqui, o código lida com a atualização de registros com base no método de requisição HTTP utilizado.

Se o método de requisição for POST ($\_SERVER['REQUEST\_METHOD'] === 'POST'), o código verifica se o campo cnpj foi preenchido. Se sim, o valor do CNPJ é obtido de $\_POST['cnpj'].

**if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {**

**// Verifica se o campo CNPJ foi preenchido**

**if (isset($\_POST['cnpj'])) {**

**$cnpj = $\_POST['cnpj'];**

**// Prepara e executa a query para buscar o registro pelo CNPJ**

Em seguida, é preparada e executada uma query SQL para buscar um registro na tabela cliente com base no CNPJ fornecido.

**$stmt = $conn->prepare('SELECT \* FROM cliente WHERE cnpj = ?');**

A query usa uma instrução preparada para evitar possíveis ataques de SQL injection:

**$stmt->bind\_param('s', $cnpj);**

**$stmt->execute();**

O resultado da query é armazenado em $result:

**$result = $stmt->get\_result();**

Se $result tiver pelo menos uma linha ($result->num\_rows > 0), significa que o registro foi encontrado:

**if ($result->num\_rows > 0) {**

Os valores das colunas do registro são atribuídos a variáveis individuais, como $id, $nome, $email, $senha e $dataPoc. Essas variáveis serão usadas para preencher o formulário de atualização:

**$row = $result->fetch\_assoc();**

**$id = $row['id'];**

**$nome = $row['nome'];**

**$email = $row['email'];**

**$senha = $row['senha'];**

**$dataPoc = $row['dataPoc'];**

**} else {**

Caso contrário, se o registro não for encontrado, a mensagem "Registro não encontrado." é exibida:

**echo 'Registro não encontrado.';}**

Após o processamento da busca ou da exibição da mensagem de erro, a instrução $stmt->close(); é usada para fechar o statement preparado:

**$stmt->close();**

**} else {**

**echo 'Por favor, preencha o campo CNPJ.';    }**

**}**

**?>**

**<!DOCTYPE html>**

**<html>**

**<head>**

O código HTML exibe um formulário para atualizar registros:

**<title>Atualizar Registro</title>**

**</head>**

**<body>**

O título "Atualizar Registro" é exibido como um cabeçalho de nível 2 (<h2>):

**<h2>Atualizar Registro</h2>**

O formulário tem o atributo method definido como "POST" e o atributo action definido como "update.php". Isso indica que, ao enviar o formulário, os dados serão enviados para a página "update.php" para processamento adicional:

**<form method="POST" action="update.php">**

Dependendo das variáveis definidas durante a busca de registro, o formulário é exibido de duas maneiras diferentes usando uma estrutura condicional if e else. Se a variável $id estiver definida, o que indica que o registro foi encontrado, o formulário é exibido para atualizar os campos do registro. Os campos do formulário são preenchidos com os valores do registro encontrados no passo anterior. Um campo de entrada oculto é adicionado para armazenar o valor do ID do registro. Caso contrário, se a variável $id não estiver definida, o que significa que o registro não foi encontrado, o formulário é exibido com apenas o campo CNPJ para localizar o registro:

**<?php if (isset($id)) : ?>**

**<input type="hidden" name="id" value="<?php echo $id; ?>">**

Independentemente do modo de exibição do formulário, os elementos <label> são usados para exibir rótulos para os campos do formulário:

**<label for="nome">Nome:</label>**

Os campos de entrada <input> são usados para coletar dados do usuário:

**<input type="text" name="nome" value="<?php echo $nome; ?>" required><br>**

**<label for="cnpj">CNPJ:</label>**

Os campos obrigatórios são definidos usando o atributo required:

**<input type="text" name="cnpj" value="<?php echo $cnpj; ?>" required><br>**

**<label for="email">Email:</label>**

**<input type="email" name="email" value="<?php echo $email; ?>" required><br>**

**<label for="senha">Senha:</label>**

**<input type="password" name="senha" value="<?php echo $senha; ?>" required><br>**

**<label for="dataPoc">Data POC:</label>**

**<input type="date" name="dataPoc" value="<?php echo $dataPoc; ?>" required><br>**

O botão "Atualizar" é exibido quando o registro é encontrado e o usuário pode clicar nele para enviar o formulário:

**<input type="submit" value="Atualizar">**

**<?php else : ?>**

**<label for="cnpj">CNPJ:</label>**

**<input type="text" name="cnpj" required><br>**

O botão "Localizar" é exibido quando o registro não é encontrado e permite ao usuário enviar o formulário para realizar a busca:

**<input type="submit" value="Localizar">**

**<?php endif; ?>**

Um botão "Cancelar" é exibido, e ao clicar nele, o usuário é redirecionado para a página "index.php":

**<button onclick="location.href='index.php'">Cancelar</button>**

**</form>**

**</body>**

**</html>**

Em resumo, o código PHP lida com a atualização de registros com base no método de requisição HTTP. Ele busca um registro na tabela "cliente" com base no CNPJ fornecido pelo usuário. Se o registro for encontrado, exibe um formulário preenchido com os dados do registro para permitir a atualização. Caso contrário, exibe um formulário com apenas o campo CNPJ para localizar o registro. O layout criado consiste em um formulário com campos de entrada para atualização de dados do cliente, incluindo nome, CNPJ, email, senha e data POC.

**DELETE**

Esta linha inclui o conteúdo do arquivo "menu.php" no local onde está posicionada. Isso permite que o código do menu seja reutilizado em várias páginas:

**<?php include\_once 'menu.php'; ?>**

Essa linha requer o arquivo "config.php" uma vez, garantindo que as configurações de conexão com o banco de dados sejam carregadas antes de executar o restante do código:

**<?php require\_once 'config.php';**

Essa seção de código é executada quando o método da requisição é POST.

**if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {**

**// Verifica se os campos 'cnpj' e 'senha' foram preenchidos**

Ela verifica se os campos 'cnpj' e 'senha' foram preenchidos no formulário enviado pelo usuário:

**if (!empty($\_POST['cnpj']) && !empty($\_POST['senha'])) {**

Obtenção dos valores dos campos CNPJ e senha:

**$cnpj = $\_POST['cnpj'];**

**$senha = $\_POST['senha'];**

**// Verifica se o CNPJ e a senha correspondem a um registro na tabela 'cliente'**

Se ambos os campos estiverem preenchidos, o código executa uma consulta SQL para verificar se existe um registro na tabela "cliente" com o CNPJ e a senha fornecidos:

**$stmt = $conn->prepare('SELECT \* FROM cliente WHERE cnpj = ? AND senha = ?');**

**$stmt->bind\_param('ss', $cnpj, $senha);**

**$stmt->execute();**

**$result = $stmt->get\_result();**

**if ($result->num\_rows > 0) {**

**// O CNPJ e a senha estão corretos, prossegue com a exclusão do registro**

Se for encontrado um registro correspondente, o código prepara uma declaração DELETE para excluir o registro com base no CNPJ:

**$stmt = $conn->prepare('DELETE FROM cliente WHERE cnpj = ?');**

**$stmt->bind\_param('s', $cnpj);**

**$stmt->execute();**

Em seguida, executa a declaração DELETE e verifica se pelo menos uma linha foi afetada. Se sim, exibe a mensagem "Registro excluído com sucesso!":

**if ($stmt->affected\_rows > 0) {**

**echo 'Registro excluído com sucesso!';**

**} else {**

Caso contrário, exibe a mensagem "Erro ao excluir o registro.".

**echo 'Erro ao excluir o registro.'; }**

**$stmt->close();**

**} else {**

Se o CNPJ e/ou a senha fornecidos estiverem incorretos ou se os campos CNPJ e senha não forem preenchidos, o código exibe mensagens de erro apropriadas:

**echo 'CNPJ e/ou senha incorretos.'; }**

**} else {**

**echo 'Por favor, preencha os campos CNPJ e senha.'; }**

}

Esta é uma página HTML responsável por exibir um formulário para excluir um registro. Essas linhas definem a estrutura básica de uma página HTML. O <!DOCTYPE html> declara que o documento é um documento HTML5 válido. As tags <html>, <head> e <title> definem o início do elemento HTML, o cabeçalho da página e o título da página, respectivamente.

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Excluir Registro</title>**

**</head>**

A tag <body> indica o início do corpo da página HTML:

**<body>**

A linha <h2>Excluir Registro</h2> exibe o título "Excluir Registro" como um cabeçalho de segundo nível:

**<h2>Excluir Registro</h2>**

A tag <form> define um formulário HTML. O atributo method indica que o formulário será enviado usando o método POST. O atributo action especifica o arquivo PHP para o qual os dados do formulário serão enviados quando o formulário for submetido:

**<form method="POST" action="delete.php">**

Essas linhas exibem os rótulos e campos de entrada para o CNPJ e a senha. A tag <label> é usada para criar um rótulo descritivo para cada campo de entrada. O atributo “for” de cada <label> faz referência ao atributo name correspondente nos campos de entrada. Os campos de entrada são criados usando a tag <input>. O tipo de campo para o CNPJ é "text" e para a senha é "password". O atributo required é adicionado para tornar os campos obrigatórios, o que significa que o usuário deve preenchê-los antes de enviar o formulário:

**<label for="cnpj">CNPJ:</label>**

**<input type="text" name="cnpj" required><br>**

**<label for="senha">Senha:</label>**

**<input type="password" name="senha" required><br>**

Essas linhas definem os botões de envio e cancelamento do formulário. A tag <input> com type="submit" cria um botão de envio que, quando clicado, envia o formulário para o arquivo especificado no atributo action. O texto exibido no botão é definido pelo atributo value:

**<input type="submit" value="Excluir">**

A tag <button> cria um botão de cancelamento que, quando clicado, redireciona o usuário para a página "index.php"

**<button onclick="location.href='index.php'">Cancelar</button>**

**</form>**

**</body>**

**</html>**