ATIVIDADE DE PHP

PHP

O PHP é uma linguagem de programação para criar aplicações web.

Um site é um conjunto de páginas web com informações diversas que tem o intuito praticamente de informar ou expor alguma informação.

A principal diferença entre um site comum e uma aplicação web é a interatividade e a funcionalidade. Enquanto um site comum serve principalmente para fornecer informações estáticas, uma aplicação web permite uma maior interação do usuário e pode realizar tarefas mais complexas.

Uma aplicação web é um programa que é acessado através de um navegador e que utiliza a web para realizar suas funções. Ela pode oferecer uma variedade de funcionalidades, como comércio eletrônico, colaboração em tempo real, interação com bancos de dados e muito mais. Exemplos de aplicações web incluem sistemas de gerenciamento de conteúdo, webmails, lojas online e redes sociais.

BANCO DE DADOS

Um banco de dados é um local onde são armazenados sistemas e dados relacionados a processos internos e comerciais, redes sociais, controle de fornecedores, atendimento e outras fontes pertinentes às operações de uma empresa.

TABELA

Em um banco de dados, uma tabela é um conjunto de dados organizados em um formato de linha e coluna, semelhante a uma planilha.

SQLite

O SQLite é um Sistema de Gerenciamento de Banco de Dados (SGBD).

SGBD

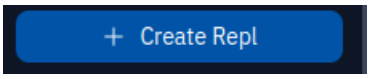
Um SGBD, ou Sistema de Gerenciamento de Banco de Dados, é um software utilizado para gerenciar um banco de dados.

Em termos simples, você pode pensar em um SGBD como um intermediário entre os usuários (ou aplicações) e os dados. Ele permite que os usuários criem, leiam, atualizem e excluam dados em um banco de dados de maneira organizada e segura.

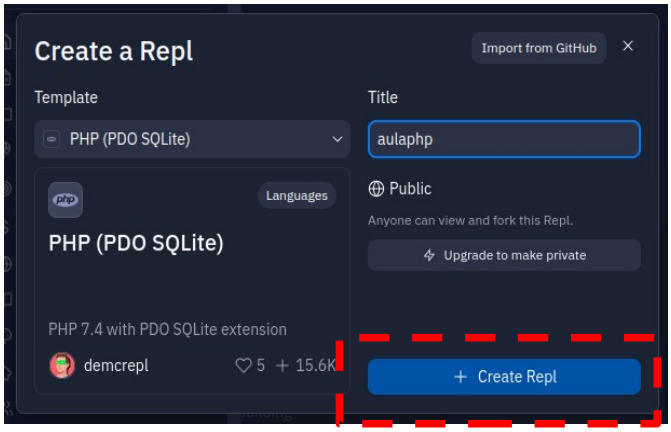
ATIVIDADE PRÁTICA

Vamos acessar o replit.com.

Logue com uma conta.

Clique em: 

Crie um ambiente para programar em PHP e SQLite.



Apague o arquivo database.sqlite.

Altere o arquivo index.php e escreva o seguinte script:

```
<?php
   $db = new SQLite3('agenda.db');

   if($db){
      echo "Conexão com o banco de dados SQLite 'agenda' criado com sucesso!";
   } else {
      echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
   }
?>
```

O código acima cria o arquivo do banco de dados (cria o banco de dados) e verifica se há conexão com o mesmo. Dependendo da verificação ele exibirá na página a mensagem de sucesso da conexão ou insucesso. Clique no botão RUN.

Vamos alterar agora um pouco o código para criação de uma tabela no database.

Clique no botão STOP.

Vamos agora usar as seguintes alterações no index.php.

```
<?php
   $db = new SQLite3('agenda.db');

   if($db){

     $query = "CREATE TABLE IF NOT EXISTS agendatelefonica (id INTEGER PRIMARY
KEY, nome TEXT, telefone TEXT)";

     $db->exec($query);

   } else {
      echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
   }
?>
```

Utilizamos uma instrução SQL para criação da tabela e a executamos.

Renomeie o arquivo index.php para conexao.php.

INSERT

Iremos criar um script em nosso projeto PHP para inserção de dados na tabela.

Crie um script chamado insere.php.

```
<?php
   // Inclua o arquivo de conexão com o banco de dados
   include 'conexao.php';

   // Verifique se a conexão foi bem-sucedida
   if($db){
      // Prepare a consulta SQL para inserir os dados
      $query = "INSERT INTO agendatelefonica (id, nome, telefone) VALUES (1, 'João',
'123456789')";

      // Execute a consulta SQL
      $db->exec($query);

   } else {
      echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
   }
?>
```

No arquivo acima incluímos o arquivo de conexão com o banco de dados.

Casjo haja conexão com o banco de dados haverá a criação do SQL para inserção de dados (INSERT INTO) na tabela criada. O comando exec executa o SQL criado.

Caso haja erro de conexão com o database a mensagem do ELSE é executada.

Vamos executar o primeiro e segundo código usando no Shell o comando: **php arquivo.php**. O painel Shell fica na lateral direita da janela.

No INSERT acima inserimos dados nos campos nome e telefone.

Como visualizar os dados inseridos na página?

Vamos agora criar um arquivo php para visualizar os dados inseridos.

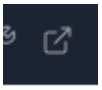
Crie um arquivo chamado consulta.php para visualizar os dados cadastrados em uma tabela.

Digite o código abaixo:

```
<!DOCTYPE html>
<html>
<head>
  <title>Agenda Telefônica</title>
</head>
<body>
  <h1>Agenda Telefônica</h1>
  <table>
    <tr>
      <th>ID</th>
      <th>Nome</th>
      <th>Telefone</th>
    </tr>

    <?php
      // Inclua o arquivo de conexão com o banco de dados
      include 'conexao.php';

      // Verifique se a conexão foi bem-sucedida
      if($db){
        // Prepare a consulta SQL para obter os dados
        $query = "SELECT id, nome, telefone FROM agendatelefonica";

        // Execute a consulta SQL
        $result = $db->query($query);

        // Percorra cada linha do resultado
        while ($row = $result->fetchArray()) {
          echo "<tr>";
          echo "<td>" . $row['id'] . "</td>";
          echo "<td>" . $row['nome'] . "</td>";
          echo "<td>" . $row['telefone'] . "</td>";
          echo "</tr>";
        }
      } else {
        echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
      }
    ?>
  </table>
```

```
</body>
</html>
```

Salve o arquivo.

Clique no botão 

Após a URL digite https://aulaphp.davide106.repl.co/**consulta.php**

Visualize os dados que inserimos antes.

Entendendo o código.

```
<?php
   // Inclua o arquivo de conexão com o banco de dados
   include 'conexao.php';
```

Este trecho inclui o arquivo 'conexao.php', que é onde a conexão com o banco de dados é estabelecida. O comando `include` em PHP pega todo o texto/código/markup que existe no arquivo especificado e copia no arquivo que usa o comando include.

```
   // Verifique se a conexão foi bem-sucedida
   if($db){
```

Aqui, estamos verificando se a conexão com o banco de dados foi bem-sucedida. A variável `$db` deve conter a referência para a conexão do banco de dados.

```
      // Prepare a consulta SQL para obter os dados
      $query = "SELECT id, nome, telefone FROM agendatelefonica";
```

Este trecho está preparando uma consulta SQL para selecionar as colunas 'id', 'nome' e 'telefone' da tabela 'agendatelefonica'.

```
      // Execute a consulta SQL
      $result = $db->query($query);
```

Aqui, a consulta SQL é executada usando o método `query()` do objeto de conexão do banco de dados, e o resultado é armazenado na variável `$result`.

```
      // Percorra cada linha do resultado
      while ($row = $result->fetchArray()) {
         echo "<tr>";
         echo "<td>" . $row['id'] . "</td>";
         echo "<td>" . $row['nome'] . "</td>";
         echo "<td>" . $row['telefone'] . "</td>";
```

```
        echo "</tr>";
    }
```

Este trecho está percorrendo cada linha do resultado da consulta. Para cada linha, ele imprime uma linha HTML (`<tr>`) com três células (`<td>`), cada uma contendo um dos valores das colunas 'id', 'nome' e 'telefone'.

```
} else {
    echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
}
?>
```

Se a conexão com o banco de dados não for bem-sucedida, este trecho imprimirá uma mensagem de erro.

Vamos criar um arquivo html que utilizaremos para preencher os dados em nosso database.

Crie o arquivo index.html e insira o seguinte código:

```
<!DOCTYPE html>
<html>
<head>
  <title>Inserir na Agenda Telefônica</title>
</head>
<body>
  <h1>Inserir na Agenda Telefônica</h1>

  <form action="insere.php" method="post">
    <label for="nome">Nome:</label><br>
    <input type="text" id="nome" name="nome"><br>
    <label for="telefone">Telefone:</label><br>
    <input type="text" id="telefone" name="telefone"><br>
    <input type="submit" value="Inserir">
  </form>
</body>
</html>
```

Observações:

No HTML, os atributos `method` e `action` são usados em conjunto com a tag `<form>` para controlar o que acontece quando um formulário é submetido[32].

- method: Este atributo especifica como os dados do formulário devem ser enviados. Os dois métodos mais comuns são `GET` e `POST`:

    - `GET`: Com este método, os dados do formulário são anexados à URL como parâmetros[3]. Isso é útil para coisas como consultas de pesquisa, onde os resultados podem ser marcados nos favoritos e compartilhados. No entanto, há um limite para o comprimento da URL, então este método não é adequado para a transmissão de grandes quantidades de dados.

- `POST`: Com este método, os dados do formulário são enviados no corpo da solicitação HTTP[3]. Isso permite o envio de grandes quantidades de dados e garante que os dados do formulário não sejam visíveis na URL[3].

- action: Este atributo especifica para onde os dados do formulário devem ser enviados quando o formulário é submetido[2]. O valor deste atributo é normalmente uma URL para o servidor que irá processar os dados do formulário[2]. Se o atributo `action` for deixado em branco, os dados do formulário serão enviados para a URL atual.

Vamos alterar o arquivo insere.php para ficar assim:

```
<?php
   // Inclua o arquivo de conexão com o banco de dados
   include 'conexao.php';

   // Verifique se a conexão foi bem-sucedida
   if($db){
      // Pegue os dados do formulário
      $nome = $_POST['nome'];
      $telefone = $_POST['telefone'];

      // Prepare a consulta SQL para inserir os dados
      $query = "INSERT INTO agendatelefonica (nome, telefone) VALUES ('$nome',
'$telefone')";

      // Execute a consulta SQL
      $db->exec($query);

      echo "Dados inseridos com sucesso na tabela 'agendatelefonica'!";
   } else {
      echo "Desculpe, não foi possível conectar ao banco de dados SQLite 'agenda'.";
   }
?>
```

Salve o arquivo html clique no botão RUN, preencha os dados e aguarde a mensagem de sucesso ao cadastrar.

No arquivo HTML, criamos um formulário com campos para o nome e o telefone. Quando o usuário clica no botão “Inserir”, os dados do formulário são enviados para o arquivo ‘inserir.php’ usando o método POST.

No arquivo PHP, pegamos os dados do formulário usando $_POST['nome'] e $_POST['telefone']. Em seguida, preparamos uma consulta SQL para inserir esses dados na tabela “agendatelefonica”. Finalmente, executamos a consulta SQL para inserir os dados.

Verifique na página de consulta se os dados foram cadastrados.

ASSUNTOS EXTRAS PARA OUTRAS ATIVIDADES

1. UPDATE: A instrução `UPDATE` é usada para modificar os registros existentes em uma tabela[12]. A sintaxe básica do `UPDATE` é a seguinte:

UPDATE nome_da_tabela
SET coluna_1 = novo_valor_1, coluna_2 = novo_valor_2,...
WHERE condição;

Nesta sintaxe:
- `nome_da_tabela` é o nome da tabela onde você deseja atualizar os registros.
- `SET coluna_1 = novo_valor_1, coluna_2 = novo_valor_2,...` é usado para especificar as colunas que você deseja atualizar e os novos valores que você deseja definir.
- `WHERE condição` é usado para especificar quais registros você deseja atualizar. Se você omitir a cláusula `WHERE`, todos os registros na tabela serão atualizados.

2. DELETE: A instrução `DELETE` é usada para excluir uma ou mais linhas de uma tabela. A sintaxe básica do `DELETE` é a seguinte:

DELETE FROM nome_da_tabela
WHERE condição;

Nesta sintaxe:
- `nome_da_tabela` é o nome da tabela onde você deseja excluir os registros.
- `WHERE condição` é usado para especificar quais registros você deseja excluir. Se você omitir a cláusula `WHERE`, todos os registros na tabela serão excluídos.