



Fundamentos de Programação Web

UNIDADE 07

PHP + BD relacional

Olá!

Uma vez que a **estrutura de dados** que apoiará a persistência da nossa aplicação web esteja definida e implementada em um banco de dados, o próximo passo é habilitar o acesso da aplicação desenvolvida em **PHP** ao **banco de dados**.

Observe que este momento de integração *front-end* e *back-end* apenas ocorre após definirmos o **projeto** da nossa aplicação, que envolve:

1. **Especificar a aplicação.** O que é, quem são os interessados na aplicação e o que cada um faz na aplicação (requisitos funcionais).
2. **Definir a estrutura de dados (*back-end*).** Quais dados são utilizados e precisam ser persistidos, quais seus tipos e como esses dados se relacionam (cardinalidade entre PK e FK), para apoiar os requisitos funcionais identificados.
3. **Definir a interface de usuário (*front-end*).** Preparar as telas para que o usuário realize suas tarefas na aplicação (interface para os requisitos funcionais).
4. **Integrar *back-end* e *front-end*.** Definir como enviar e recuperar os dados do BD.

Nos exemplos apresentados na VIDEOAULA, já começamos a demonstrar como o acesso aplicação PHP e BD é realizado, usando uma rede TCP/IP e inserindo comandos SQL no código PHP.

Em resumo, nesta unidade, você trabalhará para:

1. Definir e criar uma estrutura para manter dado de uma aplicação *web*.
2. Integrar *front-end* e *back-end* de uma aplicação *web*.

Ao final da implementação da **1ª *sprint***, nossa aplicação estará configurada como na Figura a seguir.

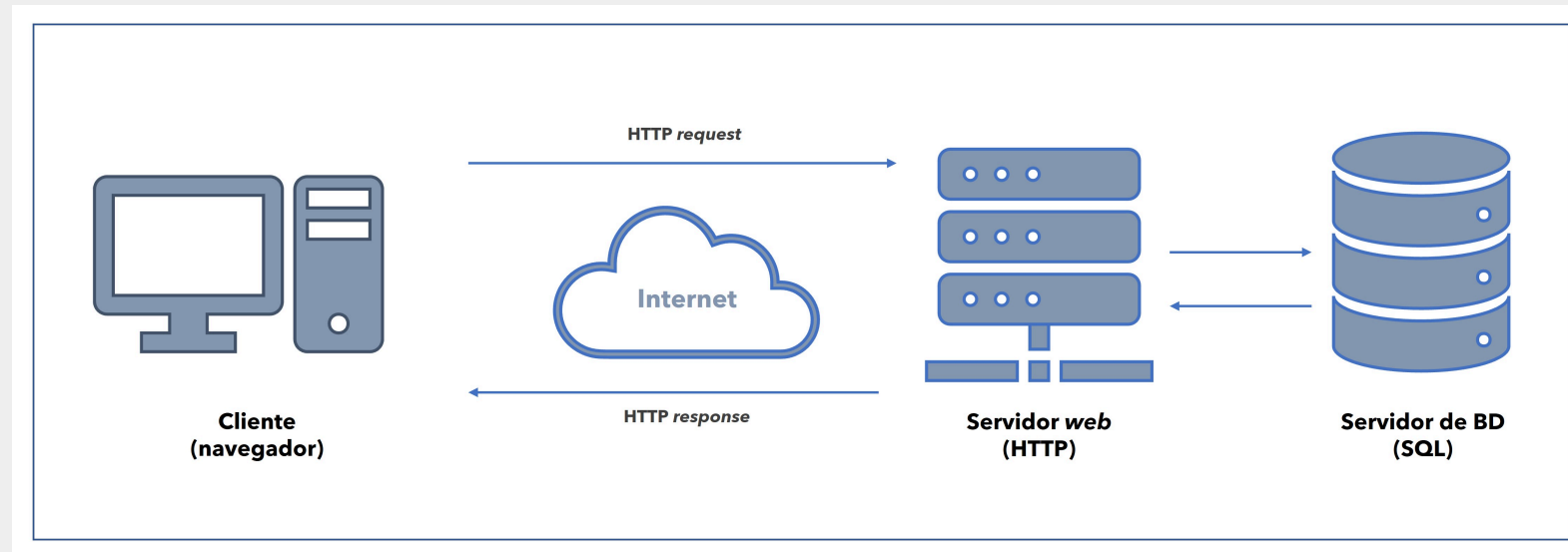


Figura 1: Aplicação web com banco de dados. Fonte: A autora (2023).



SAIBA MAIS

PHP + MySQL

- **Tutorial PHP MySQL DataBase:** https://www.w3schools.com/php/php_mysql_intro.asp.
- MILETTO, Evandro Manara; BERTAGNOLLI, Silvia de Castro. **Desenvolvimento de Software II: introdução ao desenvolvimento web com html, css, javascript e php**. Porto Alegre: Bookman, 2014. 276 p. (Capítulo 8 – Integração de PHP e MySQL, página 205).

Antes de explorarmos os temas de estudo e os exemplos desta unidade, vamos verificar os passos básicos da comunicação entre o **PHP** e o **MySQL**, utilizando o nosso ambiente de trabalho preparado com o servidor de banco de dados **MySQL** e o servidor HTTP **Apache**, ambos do **XAMPP**. Após, estaremos prontos para ver os detalhes da recuperação e envio de dados entre o **PHP** e o **MySQL**.

Trabalhando com o PHP e banco de dados relacional



| Integração PHP e MySQL


Como vimos na VIDEOAULA desta unidade, primeiramente, precisamos que o nosso ambiente de trabalho esteja com um servidor Apache e um servidor MySQL operacionais.


Também, precisamos saber qual a **porta TCP**¹ em que o **MySQL** receberá as requisições. Por padrão, essa porta é a **3306**, como podemos identificar no painel de controle do XAMPP, conforme indicado na Figura a seguir.

¹ A comunicação entre **HTTP server** e **DB server** é feita pelo conjunto de **protocolos de comunicação TCP/IP**, o mesmo utilizado na internet. A **porta de comunicação**, ou **socket**, identifica qual a **entrada** que uma aplicação irá processar as requisições de

comunicação. Já o **endereço IP** identifica um equipamento em rede. Juntos, **socket** e **IP** permitem que duas aplicações distintas, que podem estar em diferentes equipamentos, se comuniquem e gerem resultados.



 XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]



XAMPP Control Panel v3.3.0

Modules

Service	Module	PID(s)	Port(s)	Actions
<input type="checkbox"/>	Apache	32376 15784	80, 443	Stop Admin Config Logs
<input type="checkbox"/>	MySQL	16712	3306	Stop Admin Config Logs
<input type="checkbox"/>	FileZilla			Start Admin Config Logs
<input type="checkbox"/>	Mercury			Start Admin Config Logs
<input type="checkbox"/>	Tomcat			Start Admin Config Logs

Config

Netstat

Shell

Explorer

Services

Help

Quit

16:06:42 [Apache] Attempting to stop Apache (PID: 29384)

16:06:44 [mysql] Attempting to stop MySQL app...

16:06:44 [mysql] Status change detected: stopped

16:06:46 [Apache] Status change detected: stopped

21:34:43 [Apache] Attempting to start Apache app...

21:34:43 [Apache] Status change detected: running

21:34:43 [mysql] Attempting to start MySQL app...

21:34:45 [mysql] Status change detected: running

Apache recebe requisições nas portas padrão 80 para HTTP e 443 para HTTPS, enquanto o MySQL recebe requisições na porta-padrão 3360.

Figura 2: Painel de Controle do XAMPP, com Apache e MySQL em execução. Fonte: A autora (2023).

Considere que nos exemplos a seguir, temos uma tabela chamada “**minhaTabela**” na base de dados “**minhaBase**”, preenchida com os seguintes registros.

Tabela minhaTabela			
id (PK)	Nome	Sobrenome	Email
1	Maria	Alves	maria.a@email.com
2	José	Soares	jose.s@email.com
3	Carla	Ribeiro	carla.r@email.com

1. Conexão com o BD

Existem diferentes formas de realizar uma conexão com o BD via **PHP**. Nos nossos exemplos, utilizaremos uma **extensão do PHP** muito comum, que é a extensão de conexão **mysqli**. Toda a comunicação entre o PHP e o BD MySQL acontecerá via essa conexão. Na Figura a seguir, está a sintaxe básica para criar e utilizar a conexão.

Figura 3: Conexão com o MySQL, via PHP

</>

```
1
2 $servername = "localhost"; // define o endereço de acesso do MySQL: equipamento local
3 $username = "root";        // define o usuário de BD para acesso: root é o usuário padrão
4 $password = "";            // define a senha de acesso do usuário: padrão é senha vazia
5 $dbname = "minhaBase";     // define a base de dados a ser acessada
6
7 // Cria a conexão
8 $conn = new mysqli($servername, $username, $password, $dbname);
9
10 // Verifica se a criação da conexão foi bem sucedida
11 if (!$conn) { // Se a conexão não foi criada, encerra a execução do PHP com a mensagem de erro.
12     die("Falha de conexão: " . mysqli_connect_error());
13 }
14 echo "Conexão bem sucedida!";
15 ?>
```

Texto alternativo de acessibilidade: A Figura 3 apresenta um trecho de código em PHP para realizar uma conexão com um servidor MySQL. O código destaca o comando que cria a conexão, linha iniciada com “\$conn = new ...”, e também o comando que testa se a conexão foi criada com sucesso, linha iniciada por “if (!\$conn) { ...”.

Fonte: A autora (2023).



SAIBA MAIS

PHP + MySQL: conexão

- Tutorial PHP MySQL *DataBase*: https://www.w3schools.com/php/php_mysql_intro.asp.

- **Conexão:** https://www.w3schools.com/php/php_mysql_connect.asp

2. Inserção de dados no BD (INSERT)

Iremos direto para a inserção de dados no BD (*INSERT INTO ...*), pois a criação da base de dados (*CREATE DATABASE ...*) e a criação de tabelas (*CREATE TABLE ...*) não devem, geralmente, ser realizadas no código **PHP** (aplicação). Ao invés, toda a estrutura da base de dados já deve estar pronta para acesso via aplicação, independente da linguagem utilizada.

Assim, para inserir dados na base a partir do PHP, devemos ter uma sequência de comandos básicos, como demonstrado na Figura a seguir.

Figura 4: Inserção de dados no MySQL, via PHP




```
1
2 $servername = "localhost"; // define o endereço de acesso do MySQL: equipamento local
3 $username = "root";        // define o usuário de BD para acesso: root é o usuário padrão
4 $password = "";            // define a senha de acesso do usuário: padrão é senha vazia
5 $dbname = "minhaBase";     // define a base de dados a ser acessada
6
7 // Cria a conexão
8 $conn = new mysqli($servername, $username, $password, $dbname);
9 // Verifica se a criação da conexão foi bem sucedida
10 if ($conn->connect_error) {
11     die("Falha de conexão: " . $conn->connect_error);
12 }
13 // Cria o comando SQL adequado para o INSERT
14 $sql = "INSERT INTO minhaTabela (nome, sobrenome, email)
15 VALUES ('Maria', 'Alves', 'm.alves@email.com')";
16
17 // Executa e testa se o comando foi bem sucedido
18 if ($conn->query($sql) === TRUE) {
19     echo "Registro gravado com sucesso";
20 } else {
21     echo "Erro: " . $sql . "
22 " . $conn->error;
23 }
24
25 $conn->close(); // Encerra a conexão com o BD
26 ?>
```

Texto alternativo de acessibilidade: A Figura 4 apresenta um trecho de código em PHP para realizar a inserção de dados, via comandos SQL, em tabela de uma base MySQL. O código destaca a criação do comando SQL dentro do PHP, linha iniciado por “\$sql =

"INSERT INTO ... ". Também está destacado comando que executa e testa, a partir do PHP, a requisição de inserção no MySQL, linha iniciada por "if (\$conn->query(\$sql) === TRUE) ...".

Fonte: A autora (2023).



SAIBA MAIS

PHP + MySQL: inserção de dados

- Tutorial PHP MySQL DataBase: https://www.w3schools.com/php/php_mysql_intro.asp.
 - INSERT: https://www.w3schools.com/php/php_mysql_insert.asp
 - INSERT múltiplo: https://www.w3schools.com/php/php_mysql_insert_multiple.asp

3. Leitura de dados do BD (SELECT)

A leitura, ou recuperação, de dados de tabelas (*SELECT*) é uma das operações mais utilizadas quando trabalhamos com banco de dados. Talvez o maior cuidado deva ser no momento de tratar o retorno do comando. Como visto na VIDEOAULA, basicamente, devemos observar bem o que solicitamos para referenciar corretamente cada um dos campos recebidos: lembrar que o **PHP** é “*case sensitive*”, logo a escrita do nome de cada campo no comando SQL deve ter total correspondência no código PHP.

Assim, para recuperar dados na base a partir do PHP, devemos ter uma sequência de comandos básicos, como demonstrado na Figura a seguir.

Figura 5: Leitura (recuperação) de dados no MySQL, via PHP

</>

```
1
2 $servername = "localhost"; // define o endereço de acesso do MySQL: equipamento local
3 $username = "root";        // define o usuário de BD para acesso: root é o usuário padrão
4 $password = "";            // define a senha de acesso do usuário: padrão é senha vazia
5 $dbname = "minhaBase";     // define a base de dados a ser acessada
6
7 // Cria a conexão
8 $conn = new mysqli($servername, $username, $password, $dbname);
9 // Verifica se a criação da conexão foi bem-sucedida
10 if ($conn->connect_error) {
11     die("Falha de conexão: " . $conn->connect_error);
12 }
13 // Cria o comando SQL adequado para o SELECT
14 $sql = "SELECT id, nome, sobrenome FROM minhaTabela";
15 $result = $conn->query($sql); // Executa o comando SQL
16
17 if ($result->num_rows > 0) { // Verifica se são retornadas linhas
18     // Exibe os dados de cada linha retornada
19     while($row = $result->fetch_assoc()) {
20         echo "id: " . $row["id"]. " - Nome: " . $row["nome"]. " " . $row["sobrenome"]. "
21 ";
22     }
23 } else {
24     echo "Não foram retornados registros."; // Não há linhas (registros) retornados
25 }
26
27 $conn->close(); // Encerra a conexão com o BD
28 ?>
```

Texto alternativo de acessibilidade: A Figura 5 apresenta um trecho de código em PHP para realizar a recuperação de dados, via comandos SQL, de tabela de uma base MySQL. O código destaca a criação do comando SQL dentro do PHP, linha iniciado por “`$$sql = "SELECT id, nome, ... "`”. Também está destacado comando que executa, a partir do PHP, a requisição de leitura de dados no MySQL, linha iniciada por “`$result = $conn->query($sql) ...`”. Se o PHP recebeu esses dados ele os exibe, linha iniciada por “`while($row = $result->fetch_assoc()) ...`”, seguida pela linha “`echo "id: ..."`”.

Fonte: A autora (2023).



SAIBA MAIS

PHP + MySQL: leitura de dados

- Tutorial PHP MySQL DataBase: https://www.w3schools.com/php/php_mysql_intro.asp.
 - SELECT: https://www.w3schools.com/php/php_mysql_select.asp
 - SELECT + WHERE: https://www.w3schools.com/php/php_mysql_select_where.asp
 - SELECT + ORDER: https://www.w3schools.com/php/php_mysql_select_orderby.asp
 - SELECT + limitação do total de registros retornados: https://www.w3schools.com/php/php_mysql_select_limit.asp

4. Exclusão de dados do BD (*DELETE*)

A exclusão, ou remoção, de dados de tabelas (*DELETE*) também segue o mesmo tratamento no PHP que os comandos SQL já mencionados. Basicamente, precisamos identificar no SQL o que queremos excluir da base.

Assim, para excluir dados na base a partir do PHP, devemos ter uma sequência de comandos básicos, como demonstrado na Figura a seguir.

Figura 6: Exclusão (remoção) de dados no MySQL, via PHP

```
</>

1
2 $servername = "localhost"; // define o endereço de acesso do MySQL: equipamento local
3 $username = "root";        // define o usuário de BD para acesso: root é o usuário padrão
4 $password = "";            // define a senha de acesso do usuário: padrão é senha vazia
5 $dbname = "minhaBase";     // define a base de dados a ser acessada
6
7 // Cria a conexão
8 $conn = new mysqli($servername, $username, $password, $dbname);
9 // Verifica se a criação da conexão foi bem-sucedida
10 if ($conn->connect_error) {
11     die("Falha de conexão: " . $conn->connect_error);
12 }
13 // Cria o comando SQL adequado para o DELETE
14 $sql = "DELETE FROM minhaTabela WHERE id=3";
15
16 // Executa e testa se o comando foi bem-sucedido
17 if ($conn->query($sql) === TRUE) {
18     echo "Registro excluído com sucesso.";
19 } else {
20     echo "Erro ao excluir registro: " . $conn->error;
21 }
22 $conn->close(); // Encerra a conexão com o BD
23 ?>
```

Texto alternativo de acessibilidade: A Figura 6 apresenta um trecho de código em PHP para realizar a exclusão de dados, via comandos SQL, de tabela de uma base MySQL. O código destaca a criação do comando SQL dentro do PHP, linha iniciado por “\$sql = ”DELETE FROM ... “. Também está destacado comando que executa e testa, a partir do PHP, a requisição de inserção no MySQL, linha iniciada por “if (\$conn->query(\$sql) === TRUE) ...”.

Fonte: A autora (2023).



SAIBA MAIS

PHP + MySQL: exclusão de dados

- Tutorial PHP MySQL DataBase: https://www.w3schools.com/php/php_mysql_intro.asp.
- DELETE: https://www.w3schools.com/php/php_mysql_delete.asp

5. Atualização de dados do BD (UPDATE)

A atualização de dados de tabelas (*UPDATE*) também segue o mesmo tratamento no PHP que os comandos SQL já mencionados. Basicamente, precisamos identificar no SQL o que queremos atualizar na base.

Assim, para excluir dados na base a partir do PHP devemos ter uma sequência de comandos básicos, como demonstrado na Figura a seguir.

Figura 7: Atualização de dados no MySQL, via PHP

</>

```
1
2 $servername = "localhost"; // define o endereço de acesso do MySQL: equipamento local
3 $username = "root";        // define o usuário de BD para acesso: root é o usuário padrão
4 $password = "";            // define a senha de acesso do usuário: padrão é senha vazia
5 $dbname = "minhaBase";     // define a base de dados a ser acessada
6
7 // Cria a conexão
8 $conn = new mysqli($servername, $username, $password, $dbname);
9 // Verifica se a criação da conexão foi bem-sucedida
10 if ($conn->connect_error) {
11     die("Falha de conexão: " . $conn->connect_error);
12 }
13 // Cria o comando SQL adequado para o UPDATE
14 $sql = "UPDATE minhaTabela SET sobrenome ='Alencar' WHERE id=1";
15
16 // Executa e testa se o comando foi bem-sucedido
17 if ($conn->query($sql) === TRUE) {
18     echo "Registro atualizado com sucesso.";
19 } else {
20     echo " Erro ao atualizar registro: " . $conn->error;
21 }
22 $conn->close(); // Encerra a conexão com o BD
23 ?>
```

Texto alternativo de acessibilidade: A Figura 7 apresenta um trecho de código em PHP para realizar a atualização de dados, via comandos SQL, de tabela de uma base MySQL. O código destaca a criação do comando SQL dentro do PHP, linha iniciado por “\$sql = “UPDATE... “. Também está destacado comando que executa e testa, a partir do PHP, a requisição de inserção no MySQL, linha iniciada por “if (\$conn->query(\$sql) === TRUE) ...”.



SAIBA MAIS

PHP + MySQL: atualização de dados

- Tutorial PHP MySQL DataBase: https://www.w3schools.com/php/php_mysql_intro.asp.
- UPDATE: https://www.w3schools.com/php/php_mysql_update.asp

| Exibição de dados para preenchimento de FK

Quando existe um **relacionamento** entre tabelas no banco de dados, é preciso igualar seus campos de **chave primária (PK)** e **chave estrangeira (FK)** para que seja possível recuperar a combinação correta dos dados das tabelas envolvidas no relacionamento.

Lembre-se:

Chave primária – *Primary Key* ou PK

- É a **campo** (coluna) escolhida para a **identificação única** de **registro** (linha).
- O valor mantido na chave primária não se repete na tabela.

Chave estrangeira – *Foreign Key* ou FK

Apenas mantém valores que são chave primária em outra tabela.

Indicar o relacionamento **PK x FK** também é necessário no **PHP**, sempre que precisamos apresentar dados mais completos na interface de usuário.

A seguir, execute a prática para criar uma **aplicação web PHP** exemplo “**consultorio**”, que acessa o **MySQL**, já vista na VIDEOAULA desta **unidade**. Esta aplicação está disponibilizada no material de apoio, arquivo compactado **ConsultorioSemana7.zip**.



DOCUMENTOS

Material de Apoio

[Consultorio_Semana7.zip](#)



EXPERIMENTE

PRÁTICA – Aplicação web exemplo “Consultório”

Para acompanhar esta aplicação exemplo, siga os passos:

1. Descompacte o arquivo **ConsultorioSemana7.zip** na sua pasta de trabalho: **C:/xampp/htdocs**.
2. Inicialize os servidores **Apache** e **MySQL** do **XAMPP**.
3. No **phpMyAdmin**, crie uma base de dados chamada **ClinicaA**.
4. No **phpMyAdmin**, com a base de dados **ClinicaA** selecionada, execute o script SQL: **C:/xampp/htdocs /consultorio/bd/bd_clinica.sql**. Este *script* **cria** e **preenche** as tabelas.

5. Após a criação da base, tabelas e preenchimento das tabelas, aplicação já deve estar funcional. Então, tente acessar a aplicação, digitando na barra de endereços do navegador a URL: <http://localhost/consultorio>.
6. A tela do navegador deve apresentar a relação de médicos, como na Figura a seguir.

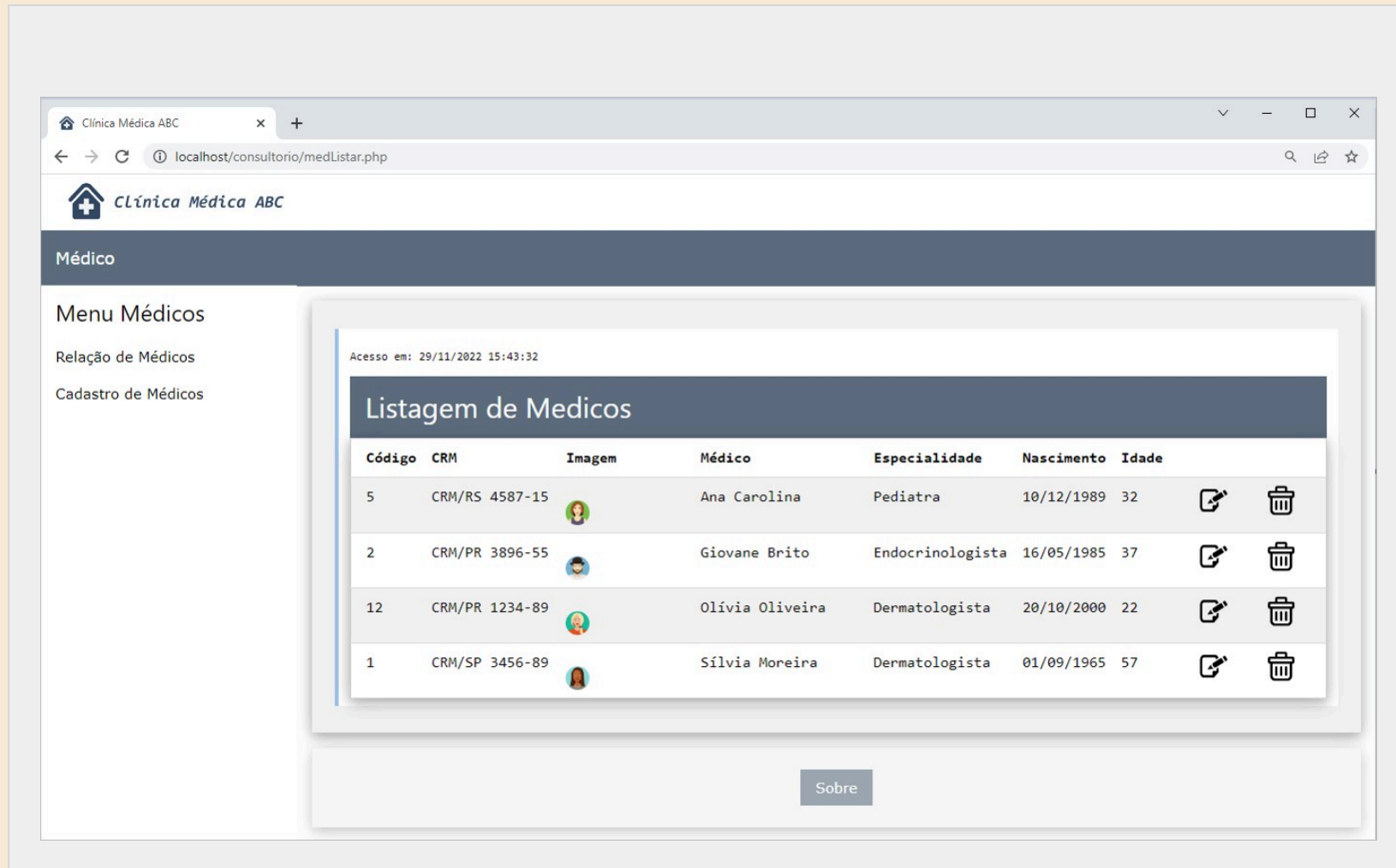


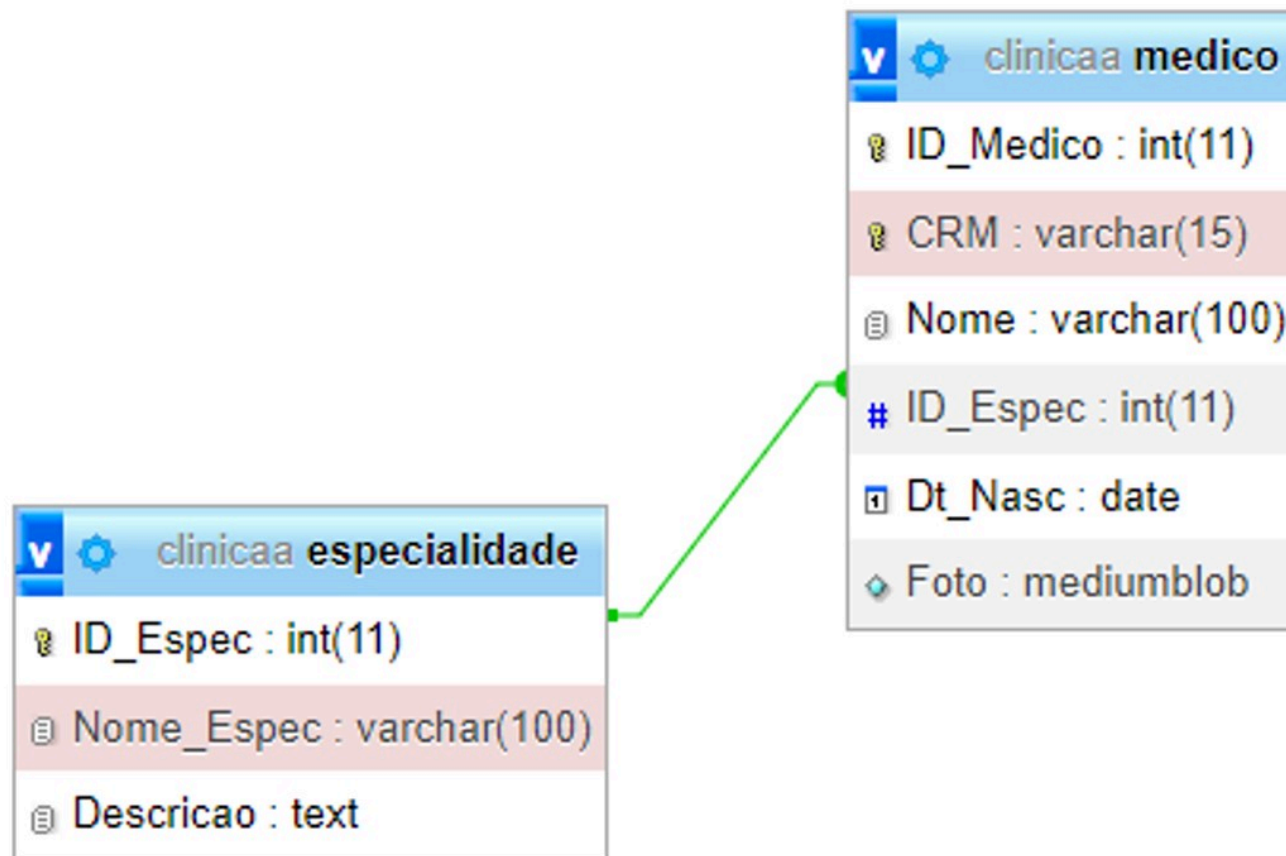
Figura 8: Interface da aplicação web exemplo: Consultório. Fonte: A autora (2023).

A imagem apresenta a aplicação, que tem um *menu* lateral, em que é possível escolher entre as opções “**relação de médicos**”, que permite visualizar o resultado de um *SELECT*, ou “**cadastro de médicos**”, que permite a entrada de dados para um *INSERT*.

A tela inicial apresenta uma listagem, exibida em uma tabela HTML, do resultado de um *SELECT* na tabela **Medico**.

Base de dados

Vamos continuar o exemplo apresentado na VIDEOAULA. Para isso, precisamos entender quais são os dados mantidos na base utilizada pela aplicação, que tem as tabelas **médico** e **especialidade**, que se relacionam como exibido na Figura a seguir.



Relacionamento 1 x N: significa que **1** Especialidade pode ser de **N** Médicos:

- **Chave primária (PK):** Especialidade.ID_Espec
- **Chave estrangeira (FK):** Medico.ID_Espec

Após executar o script SQL **bd_clinica.sql** para criar e preencher as tabelas, incluindo o código binário das fotos, podemos verificar as tabelas preenchidas com os comandos SQL:

```
1 | SELECT * FROM `medico`
2 | SELECT * FROM `especialidade`
```

O conteúdo das tabelas deve ser:

Especialidade		
ID_Espec	Nome_Espec	Descricao
1	Dermatologista	Médico especialista no diagnóstico, tratamento e p...
2	Endocrinologista	Médico especialista no diagnóstico, tratamento e p...
3	Pediatra	Médico especializado na assistência a crianças e a...

Médico					
ID_Medico	CRM	Nome	ID_Espec	DT_Nasc	Foto
1	CRM/SP 3456-89	Sílvia Moreira	1	1965-09-01	[BLOB - 9.9 KB]
2	CRM/PR 3896-55	Giovane Brito	2	1985-05-16	[BLOB - 10.4 KB]
5	CRM/RS 4587-15	Ana Carolina	3	1989-12-10	NULL
12	CRM/PR 1234-89	Oli Oli Oli	1	2000-10-20	[BLOB - 10.2 KB]
13	CRM/SP 0456-89	Jose da Silva	3	1950-03-05	[BLOB - 9.6 KB]

Seleção e apresentação de dados

É possível verificar que a tabela **médico** não possui o **nome** da especialidade do médico, apenas a **chave estrangeira (FK)** que faz referência a uma **chave primária (PK)** na tabela **especialidade**.

Contudo, a relação de médicos no navegador apresenta o nome da especialidade. Para isso, precisamos combinar os dados de **médico** e **especialidade**, em uma associação (comando SQL *JOIN*) que permite trazer o nome da especialidade, mantido na tabela **especialidade**.

Assim, para visualizar os dados combinados das duas tabelas, podemos verificar o resultado do comando SQL, diretamente no **phpMyAdmin**:

```
1 | SELECT ID_Medico, CRM, Nome, Nome_Espec AS Especialidade, Foto, Dt_Nasc
2 | FROM Medico AS M INNER JOIN Especialidade AS E ON (M.ID_Espec = E.ID_Espec)
3 | ORDER BY M.Nome
4 | /* Combina os dados que têm correspondência entre PK e FK (INNER JOIN) e apresenta 6 colunas com 4 linhas, com c
```

O resultado da consulta apenas traz os dados em que há **correspondência** entre **PK** e respectiva **FK**:

Resultado da Consulta					
ID_Medico	CRM	Nome	Especialidade	Foto	Dt_Nasc
5	CRM/RS 4587-15	Ana Carolina	Pediatra	NULL	1989-12-10
2	CRM/PR 3896-55	Giovane Brito	Endocrinologista	[BLOB - 10.4 KB]	1985-05-16
13	CRM/SP 0456-89	Jose da Silva	Pediatra	[BLOB - 9.6 KB]	1950-03-05
12	CRM/PR 1234-89	Oli Oli Oli	Dermatologista	[BLOB - 10.2 KB]	2000-10-20
1	CRM/SP 3456-89	Sílvia Moreira	Dermatologista	[BLOB - 9.9 KB]	1965-09-01

Observamos que o nome das colunas (campos) do resultado da consulta traz o “apelido” dos campos, como o caso do campo especialidade, que na tabela **especialidade** é Nome_Espec.



IMPORTANTE

Verifique, sempre que possível, o comando **SQL** no **SGBD (phpMyAdmin)** para ter certeza de que ele está funcionando corretamente para só então ser usado no **PHP**.

Framework W3.CSS:

Na aplicação *web* exemplo, utilizamos um *framework CSS* da **W3Schools** o **W3.CSS**. Ele é um arquivo de estilo que pode ser baixado e incluído nos nossos arquivos, ou é possível fazer a referência à sua URL:



```
1  ...
2  <head>
3      <title>Clínica Médica ABC</title>
4      <link rel="icon" type="image/png" href="imagens/favicon.png" />
5      <meta name="viewport" content="width=device-width, initial-scale=1">
6      <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
7      <link rel="stylesheet" href="css/customize.css">
8  </head>
9  ...
```

W3.CSS:

- É um *framework* CSS moderno, responsivo e preparado para dispositivos móveis.
- Adequado para vários navegadores: Chrome, Firefox, Edge, Safári e Ópera.
- Adequado para vários os dispositivos: *desktop*, *notebook*, *tablets* e dispositivos móveis.
- Usa apenas CSS padrão (sem jQuery ou biblioteca JavaScript).



SAIBA MAIS

- **SQL INNER JOIN:** https://www.w3schools.com/sql/sql_join_inner.asp
- **W3.CSS:** <https://www.w3schools.com/w3css/default.asp>
- **W3.CSS Templates:** https://www.w3schools.com/w3css/w3css_templates.asp

Com um conjunto de dados já selecionados, o próximo passo é exibi-los no **PHP**. Vamos verificar como habilitar essa exibição, acompanhando no código da aplicação os itens a seguir.

1. Criar uma conexão com o MySQL. Como essa conexão é comum para todas as páginas PHP, criamos o arquivo `c:\xampp\htdocs\consultorio\bd\conectaBD.php`, que deve ser inserido nos códigos PHP que precisam obter ou enviar dados para o banco.

Trecho do arquivo `c:\xampp\htdocs\consultorio\bd\conectaBD.php`

```
1  
2  global $servername ;  
3  global $username;  
4  global $password;  
5  global $database;  
6  
7  $servername = "localhost:3306"; // URL do MySQL  
8  $username = "root";           // Usuário do MySQL que acessará a base  
9  $password = "";              // Senha do Usuário do MySQL  
10 $database = "ClinicaA";       // Base de dados do MySQL que será acessada  
11 ?>
```

Trecho do arquivo `c:\xampp\htdocs\consultorio\medlistar.php`


```
1  ...
2  require 'bd/conectaBD.php'; ?>
3  ...
4
5      // Cria conexão
6      $conn = new mysqli($servername, $username, $password, $database);
7
8      // Verifica conexão
9      if ($conn->connect_error) {
10         die(" Falha de conexão: " . $conn->connect_error);
11     }
12  ...
13  ?>
```

2. Executar o comando no MySQL. Uma vez que a conexão for estabelecida, é possível interagir com o MySQL para obter ou enviar dados.

Trecho do arquivo c:\xampp\htdocs\consultorio\medlistar.php

```

1  ...
2  // Faz Select na Base de Dados
3  $sql = "SELECT ID_Medico, CRM, Nome, Nome_Espec AS Especialidade, Foto, Dt_Nasc FROM Medico AS M INNER JOIN Es
4  $result = $conn->query($sql); // Executa o comando no MySQL e recebe o resultado
5  echo "<div class='w3-responsive w3-card-4'>";
6  if ($result->num_rows > 0) {           // Se existem registros/linhas retornadas,
7      echo "<table class='w3-table-all'>"; // prepara cabeçalho para exibição
8      echo "  <tr>";
9      echo "    <th width='7%'>Código</th>";
10     echo "    <th width='14%'>CRM</th>";
11     echo "    <th width='14%'>Imagem</th>";
12     echo "    <th width='18%'>Médico</th>";
13     echo "    <th width='15%'>Especialidade</th>";
14     echo "    <th width='10%'>Nascimento</th>";
15     echo "    <th width='8%'>Idade</th>";
16     echo "    <th width='7%'> </th>";
17     echo "    <th width='7%'> </th>";
18     echo "  </tr>";
19  ...

```

3. Inserir os dados retornados do MySQL na página PHP para criar o HTML dinamicamente. Os dados retornados do SELECT precisam ser apresentados de acordo com o padrão visual da interface.

Trecho do arquivo c:\xampp\htdocs\consultorio\medlistar.php

```

1  ...
2  <?php
3  // Apresenta cada linha da tabela
4      while ($row = $result->fetch_assoc()) {
5          $data = $row['Dt_Nasc'];
6          list($ano, $mes, $dia) = explode('-', $data);
7          $nova_data = $dia . '/' . $mes . '/' . $ano;
8          // data atual
9          $hoje = mktime(0, 0, 0, date('m'), date('d'), date('Y'));
10         // Descubra a unix timestamp da data de nascimento do médico
11         $nascimento = mktime(0, 0, 0, $mes, $dia, $ano);
12         // cálculo da idade (arredondamento para baixo)
13         $idade = floor((((($hoje - $nascimento) / 60) / 60) / 24) / 365.25);
14         $cod = $row["ID_Medico"];
15         echo "<tr>";
16         echo "<td>";
17         echo $cod; // variável $cod = PK do Médico da linha corrente
18         echo "</td><td>";
19         echo $row["CRM"];

```

4. Finalizar a conexão com o MySQL. É uma boa prática finalizar a conexão com o BD, quando não foi mais necessário, para evitar erros aleatórios.

Trecho do arquivo c:\xampp\htdocs\consultorio\medlistar.php

```

1  ...
2  $conn->close();
3  ?>
4  ...

```

Neste exemplo, também apresentamos uma sugestão para inclusão de dados na tabela médico. Iniciamos montando um formulário que receberá os dados a serem enviados ao MySQL com o objetivo de criar mais um registro de médico.



IMPORTANTE

Precisamos orientar o usuário da aplicação a inserir as especialidades previstas no consultório, que já estão registradas na tabela **especialidade**: apenas valores previamente cadastrados podem ser usados para registrar qual a especialização do médico – é a implementação de uma **regra de negócio** do consultório!

Vamos verificar como criar um formulário com a relação correta de especializações que podem ser atribuídas a um médico, acompanhando no código da aplicação os itens a seguir.

1. Criar uma conexão com o MySQL e recupera os valores de especialização cadastrados. Para acessar ou enviar qualquer dado no MySQL, precisamos estabelecer a conexão. Em seguida, recuperamos os dados das especializações cadastradas:

- a. **Chave primária (PK)**, que será usada na inserção do registro do médico, nesse caso como FK, e
- b. **Nome da especialização**, dado a ser exibido ao usuário, para selecionar o que for adequado ao médico.

Trecho do arquivo `c:\xampp\htdocs\consultorio\medincluir.php`

```

1  ...
2  <?php
3      // Cria conexão
4      $conn = new mysqli($servername, $username, $password, $database);
5      // Verifica conexão
6      if ($conn->connect_error) {
7          die("<strong> Falha de conexão: </strong>" . $conn->connect_error);
8      }
9      // Obtém as Especialidades Médicas na Base de Dados para um combo box
10     $sqlG = "SELECT ID_Espec, Nome_Espec FROM Especialidade";
11     $result = $conn->query($sqlG);
12     ?>

```

2. Criar um *menu* com as opções para o usuário escolher uma especialidade. Com os dados das especializações cadastradas, criamos um menu de opções, lista ou *drop-down list* com o HTML para entrada de dados <select>, que tem uma sintaxe como a seguir:

```

<label>Qual o tipo de acomodação? </a>
<select name="tipo">
    <option value="1">Single</option>
    <option value="2">Double</option>
</select>

```

Criaremos então, para os dados na tabela especialidade, o trecho a seguir em HTML, gerado dinamicamente no PHP, em que <option value="1">Dermatologista</option>:

- <option value="1"> é a **PK ID_Espec**, de **especialidade**, que será gravada como **FK** em **Medico**.
- <option value="1">Dermatologista</option> é o **Nome_Espec** da **especialidade**.

```
<select name="Especialidade" id="Especialidade" required>
    <option value=""></option>
    <option value="1">Dermatologista</option>
    <option value="2">Endocrinologista</option>
    <option value="3">Pediatria</option>
</select>
```

Código PHP que gera o trecho HTML anterior:

Trecho do arquivo c:\xampp\htdocs\consultorio\medincluir.php

```
1  ...
2  <?php
3  ...
4  $sqlG = "SELECT ID_Espec, Nome_Espec FROM Especialidade";
5  $result = $conn->query($sqlG);
6  $optionsEspec = array(); // Cria um array para manter os dados
7
8  if ($result->num_rows > 0) {
9      while ($row = $result->fetch_assoc()) { // Preenche o array
10         array_push($optionsEspec, "\t\t\t<option value='" . $row["ID_Espec"] .
11             "'>" . $row["Nome_Espec"] . "</option>\n");
12     }
13 } else {
14     echo "Erro executando SELECT: " . $conn->connect_error;
15 }
16 $conn->close();
17 ?>
18 ...
```

3. Cria o formulário para entrada de dados. Nesta interface com o usuário, precisamos garantir que os dados recebidos estão de acordo com o tipo de dados esperado na tabela Medico. Por isso, utilizamos os recursos do HTML (*pattern* , *required*, *placeholder*, *input type* etc.) para garantir dados corretos.

Trecho do arquivo c:\xampp\htdocs\consultorio\medincluir.php (sem as classes CSS para facilitar a leitura)

```
1  ...
2  <div>
3    <div>
4      <h2>Informe os dados do novo do Médico</h2>
5    </div>
6    // enctype = form preparado para enviar dados binários da imagem
7    <form action="medIncluir_exe.php" method="post" enctype="multipart/form-data">
8    <table>
9    <tr>
10   <td style="width:50%;">
11     <p><label><b>Nome</b>*</label>
12     <input name="Nome" type="text" pattern="[a-zA-Z\u00C0-\u00FF ]{10,100}$"
13       title="Nome entre 10 e 100 letras." required>
14   </p>
15   <p><label><b>CRM</b>*</label>
16   <input name="CRM" id="CRM" type="text" maxlength="15"
17     placeholder="CRM/UF XXXX-XX" title="CRM/UF XXXX-XX"
18     pattern="CRM\/([A-Z]{2}) [0-9]{4}-[0-9]{2}$" required>
19   </p>
```

Trecho do arquivo c:\xampp\htdocs\consultorio\js\MyScriptClinic.js

```
1 function validaImagem(input) {
2     var caminho = input.value;
3
4     if (caminho) { // Verifica se o caminho para o arquivo está correto
5         var comecoCaminho = (caminho.indexOf('\\') >= 0 ? caminho.lastIndexOf('\\') : caminho.lastIndexOf('/'));
6         var nomeArquivo = caminho.substring(comecoCaminho);
7
8         if (nomeArquivo.indexOf('\\') === 0 || nomeArquivo.indexOf('/') === 0) {
9             nomeArquivo = nomeArquivo.substring(1);
10        }
11
12        var extensaoArquivo = nomeArquivo.indexOf('.') < 1 ? '' :
13            nomeArquivo.split('.').pop();
14        // Verifica se extensão do arquivo corresponde a de imagem
15        if (extensaoArquivo != 'gif' &&
16            extensaoArquivo != 'png' &&
17            extensaoArquivo != 'jpg' &&
18            extensaoArquivo != 'jpeg') {
19            input.value = ''; // Imagem inválida
20        }
21    }
22}
```

4. Aciona o **INCLUDE** dos dados do *form*. A ação do formulário indica a página `medincluir_exe.php`, que tenta executar o SQL *INSERT* no MySQL.

Trecho do arquivo `c:\xampp\htdocs\consultorio\medincluir_exe.php`


```
1 ...
2 <?php
3 // Recupera os dados do form, enviados por POST
4 $nome      = $_POST['Nome'];
5 $CRM       = $_POST['CRM'];
6 $dtNasc    = $_POST['DataNasc'];
7 $espec     = $_POST['Especialidade'];
8
9 // Cria conexão
10 $conn = new mysqli($servername, $username, $password, $database);
11 // Verifica conexão
12 if ($conn->connect_error) {
13     die("<strong> Falha de conexão: </strong>" . $conn->connect_error);
14 }
15 // Cria comando SQL INSERT adequado para imagem NÃO ENVIADA
16 if ($_FILES['Imagem']['size'] == 0) { // Não recebeu uma imagem binária
17     $sql = "INSERT INTO Medico (Nome, CRM, Dt_Nasc, ID_Espec, Foto)
18           VALUES ('$nome', '$CRM', '$dtNasc', '$espec', NULL)";
19 // Cria comando SQL INSERT adequado para imagem ENVIADA
```

Considerações finais

Experimente executar todo o exemplo no seu equipamento individual:

1. Acrescente mais especialidade na tabela, diretamente no **phpMyAdmin**. O que acontece na interface de cadastro de médico?
2. Experimente acrescentar mais uma **coluna** da tabela **médico**, por exemplo, para receber o **celular**. O que será necessário alterar no PHP para que esta informação também apareça na relação de médicos?

Projeto – 1ª *sprint*: desenvolver aplicação *web* para acesso a BD relacional.

| Atividade Formativa

A **atividade formativa** desta **unidade** é justamente passar pela 1ª *sprint* do projeto de uma aplicação *web*. Você pode utilizar o exemplo apresentado para alterar o tipo de dado mantido no MySQL e a forma de apresentar esses novos dados (interface), referentes a uma nova área de negócio, ou domínio de aplicação. Siga os passos apresentados a seguir.

Crie a sua própria aplicação web (SELECT + INSERT)

1. **Definição da uma área de negócio, ou domínio de aplicação.** Com o domínio de aplicação, é possível entender quais dados precisam ser mantidos no BD. Exemplo: comércio eletrônico, academia esportiva, rede social etc.
2. **Criação de uma base no MySQL para manter os dados da área de negócio da sua aplicação web.** Desenvolva uma nova base de dados com pelo menos uma tabela preenchida (ou duas tabelas com relacionamento 1xN) para manter dados adequados a esse negócio. Por exemplo: cadastro de produtos de uma loja, ou cadastro de professores da academia, ou cadastro de participantes de uma rede social, ou cadastro de carros de uma revendedora etc.
3. **Desenvolvimento do *front-end* da aplicação.** Crie uma interface padronizada (HTML + CSS + JavaScript), em que será possível visualizar o resultado de *SELECT* (tabela com o resultado) e de *INSERT* (formulário para obtenção de dados). O *framework* W3.CSS da W3Schools é usado para criar os estilos da aplicação exemplo.
4. **Desenvolvimento do *back-end* da aplicação.** Habilite o acesso ao BD no **PHP** para recuperar os dados do MySQL (*SELECT*) e exibi-los no navegador. Crie uma página **PHP** para receber os dados enviados de um formulário e persisti-los no MySQL (*INSERT*).

Esta atividade pode ser utilizada para iniciar a **avaliação somativa 2** a ser desenvolvida na próxima **unidade**. Lá, completaremos o CRUD (acrescentaremos o *DELETE* e o *UPDATE*) e veremos como realizar um *login* para acessar as páginas criadas.

Portanto, aproveite esta atividade formativa, a fim de estar pronto para as nossas próximas práticas!

| Conclusão

Olá!

Nesta unidade, aprendemos como unir o **PHP** com o **SQL**, especificamente do servidor de **BD MySQL**. Vimos a sequência de comandos básicos para gerenciar o tipo de tarefa realizada no banco. Também, analisamos um exemplo mais completo para *SELECT* e *INSERT*, pois apresentava uma interface finalizada (HTML + CSS + JavaScript) para exibir os resultados desejados.

Por fim, com a **atividade formativa**, iniciamos nosso projeto de **aplicação web**, com o desenvolvimento da sua **1ª sprint**. Essa primeira entrega será evoluída para a **2ª sprint**, que será a **avaliação somativa 2**, a ser realizada na próxima **unidade**.

Até lá!

| Referências Bibliográficas

ALVES, W. P. **Desenvolvimento e design de sites**. São Paulo: Erica, 2014.

MILETTO, E. M.; BERTAGNOLLI, S. C. **Desenvolvimento de software II: Introdução ao desenvolvimento web com HTML, CSS, Java Script e PHP**. Porto Alegre: Bookman, 2014.

TERUEL, E. C. **HTML 5: Guia prático**. 2. ed. Porto Alegre: Bookman, 2014.



© PUCPR - Todos os direitos reservados.