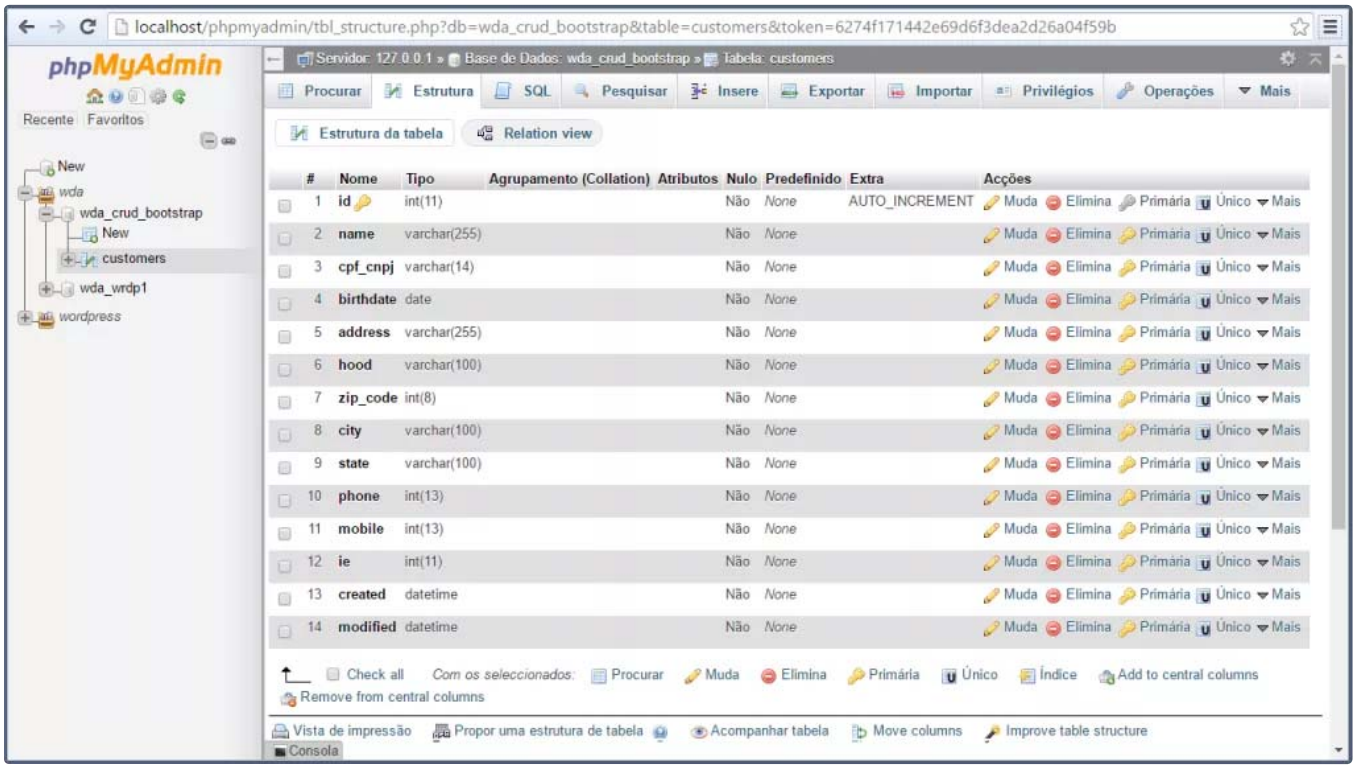
CRUD com Bootstrap, PHP & MySQL – Parte I

A maior parte dos sistemas em PHP é feito com um banco de dados no *back end*, e normalmente, esse banco de dados é o MySQL. Por isso, é importante criar essa camada de acesso a dados de uma forma bem genérica e independente, para que depois ela possa ser reaproveitada em todo o sistema.

Neste tutorial, você vai ver como deve ser esse banco de dados para o nosso sistema de CRUD e como criá-lo, executando o SQL pelo phpMyAdmin. Depois disso, vamos criar um arquivo de configurações para usar em todo o sistema. E ,por fim, vamos criar o script de conexão com esse banco de dados, usando PHP.



Ao final, você vai ter uma estrutura inicial do banco de dados para o nosso CRUD, e a tabela de clientes vazia, além dos arquivos iniciais do projeto.

Aqui, nós vamos utilizar o XAMPP, que já traz o PHP, o MySQL (ou MariaDB), e o phpMyAdmin para manipular o banco de dados.

Se você ainda não estiver com o ambiente pronto, é só ver a introdução desta série e instalar os pré-requisitos.

Passo 1: Crie o Banco de Dados e a Tabela de Clientes

Para este tutorial, vamos criar um banco de dados bem simples, com uma tabela de clientes apenas.

Eu vou usar aqueles campos de clientes que são usados em notas fiscais para criar essa tabela, e assim ela vai servir de exemplo para um cadastro de clientes bem genérico.

O mapeamento das colunas da tabela de clientes ficou assim:

```
Clientes (codigo, nome, cpf/cnpj, data de nascimento,
          endereço, bairro, cep, cidade, estado,
          telefone, celular, inscrição estadual,
          data de cadastro, data de atualização)
```

Uma boa prática, que eu sempre recomendo, é você traduzir os nomes de tabelas e os campos para o inglês, e a partir daí criar seu banco de dados. Isso facilita muito na hora de escrever as consultas, e na hora de escrever o código do sistema. Você ainda acaba aprendendo um pouco mais do idioma.

Então, nós vamos ter os seguintes campos:

customers

PK	***id***	int
	name	varchar
	cpf_cnpj	varchar

hood	varchar
zip_code	int
city	varchar
state	varchar
phone	int
mobile	int
ie	int
created	datetime
modified	datetime

E, convertendo tudo isso em SQL, fica assim:

```
CREATE TABLE customers (
  id int(11) NOT NULL,
  name varchar(255) NOT NULL,
  cpf_cnpj varchar(14) NOT NULL,
  birthdate date NOT NULL,
  address varchar(255) NOT NULL,
  hood varchar(100) NOT NULL,
  zip_code int(8) NOT NULL,
  city varchar(100) NOT NULL,
  state varchar(100) NOT NULL,
  phone int(13) NOT NULL,
  mobile int(13) NOT NULL,
  ie int(11) NOT NULL,
  created datetime NOT NULL,
  modified datetime NOT NULL
);

ALTER TABLE customers
  ADD PRIMARY KEY (id);
```



Agora, abra o phpMyAdmin do XAMPP e crie um banco de dados. Eu coloquei o nome como *wda_crud*.

Se preferir, pode fazer via SQL direto (na aba SQL, do phpMyAdmin):

```
CREATE DATABASE wda_crud;
```

Depois que criar, use o SQL acima para criar a tabela '*customers*' nesse banco de dados.

Passo 2: Crie o Arquivo de Configurações do Sistema

Agora, vamos criar um arquivo de configurações, que vai guardar todos os dados que vão ser usados em todos os outros *scritps* PHP do sistema.

Crie um arquivo chamado *config.php*, **na pasta principal**, e coloque o código a seguir:

```
<?php

/** O nome do banco de dados*/
define('DB_NAME', 'wda_crud');

/** Usuário do banco de dados MySQL */
define('DB_USER', 'root');

/** Senha do banco de dados MySQL */
define('DB_PASSWORD', '');

/** nome do host do MySQL */
define('DB_HOST', 'localhost');

/** caminho absoluto para a pasta do sistema **/
if ( !defined('ABSPATH') )
        define('ABSPATH', dirname(__FILE__) . '/');

/** caminho no server para o sistema **/
if ( !defined('BASEURL') )
        define('BASEURL', '/crud-bootstrap-php/');

/** caminho do arquivo de banco de dados **/
```

Este arquivo de configurações é baseado no *wp-config* do WordPress.

Os quatro primeiros itens são as constantes que vão guardar as credenciais para acessar o banco de dados:

- o **DB_NAME** define o nome do seu banco de dados;
- o **DB_USER** é o usuário para acessar o banco de dados;
- o **DB_PASSWORD** é a senha deste usuário (no XAMPP, este usuário *root* não tem senha);
- e o **DB_HOST** é endereço do servidor do banco de dados;

Você deve modificar esses valores de acordo com o seu ambiente de desenvolvimento, ou de produção.

Além dessas constantes, temos mais duas outras que são muito importantes:

O **ABSPATH**, na linha 17, define o caminho absoluto da pasta deste *webapp* no sistema de arquivos.

Ela vai ser usada para chamar os outros arquivos e templates via PHP (usando o *include_once*), já que ela guarda o caminho físico da pasta.

E o **BASEURL**, na linha 21, define o caminho deste *webapp* no servidor web.

Esse valor deve ser igual ao nome da pasta que você criou no começo do projeto. Ela será usada para montar os links da aplicação, já que ela guarda a URL inicial.

Depois, nós vamos adicionar mais itens neste arquivo de configurações. Por enquanto, esses aí são suficientes para conectar com o banco.

Você pode criar outros itens próprios, também, se precisar.

Passo 3: Implemente o script de Conexão com o Banco de Dados

Vamos criar um arquivo que vai ter todas as funções de acesso ao banco de dados. Assim, podemos reaproveitar código nas outras partes do CRUD.

Crie um arquivo chamado *database.php*, na **pasta inc** do seu projeto, e coloque o código a seguir:

```
1  <?php
2
3  mysqli_report(MYSQLI_REPORT_STRICT);
4
```

```
        } catch (Exception $e) {
                echo $e->getMessage();
                return null;
        }
}

function close_database($conn) {
        try {
                mysqli_close($conn);
        } catch (Exception $e) {
                echo $e->getMessage();
        }
}
```

database.php hosted with ♥ by GitHub view raw

Este será um arquivo de funções do banco de dados. Tudo que for relativo ao BD estará nele.

Vamos entender esses códigos…

Primeiramente, na linha 3, configuramos o MySQL para avisar sobre erros graves, usando a função *mysqli_report()*. Depois, temos duas funções.

A primeira função – ***open_database()*** – abre a conexão com a base de dados, e retorna a conexão realizada, se der tudo certo. Se houver algum erro, a função dispara uma exceção, que pode ser exibida ao usuário.

Já a segunda função – ***close_database($conn)*** – fecha a conexão que for passada. Se houver qualquer erro, a função dispara uma exceção, também.

Observe as constantes sendo usadas (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME). Dessa forma, caso você mude de servidor ou de BD, basta alterar o arquivo de configurações; sem precisar mexer no código principal.

Passo 4: Teste a Conexão

Agora sim, vamos ver se o banco de dados está conectado ao CRUD.

Crie um arquivo chamado *index.php*, **na pasta principal**, e coloque o código a seguir:

```
1  <?php require_once 'config.php'; ?>
2  <?php require_once DBAPI; ?>
3
4  <?php
5          $db = open_database();
```



```
10                  echo '<h1>ERRO: Não foi possível Conectar!</h1>';
11          }
12  ?>
```

index.php hosted with ❤ by GitHub view raw

Para entender, rapidamente:

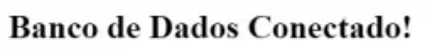
Primeiro, adicionamos o arquivo de configurações e o arquivo de funções do banco de dados (ou API de Banco de Dados), usando o require_once.

Depois, usamos a função para abrir a conexão. E, então, é feito um teste para saber se a conexão existe: *if ($db) …*

Vamos ver se funciona… Acesse o CRUD pelo navegador:

http://localhost/crud-bootstrap-php

Se aparecer a mensagem 'Banco de Dados Conectado!', sua conexão está OK.



Senão, verifique se o MySQL está iniciado e verifique se as credenciais estão corretas (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME). Qualquer item errado causa falha na conexão.

Até aqui, seu projeto deve estar assim:

- crud-bootstrap-php
  - css
  - fonts