# Aula 7º

# *MVC*

Olá,

Sejam bem-vindos à penúltima aula da disciplina Desenvolvimento Voltada à Web II. Na aula anterior, foi demonstrada a conexão com MySQL. Com ela, é possível realizar operações sobre o banco de dados, às quais chamamos de CRUD (create, read, update e delete). Por fim, foi exemplificada a criação de uma API, que retorna os registros de uma tabela, com ou sem comparação de dados.

Nesta aula, construiremos um projeto com MVC (Model, View, Controller), que divide as responsabilidades entre essas camadas. Cada qual realiza parte do trabalho que ficou responsável.

Bons estudos!

## **Objetivos de** aprendizagem

Ao término desta aula, vocês serão capazes de:

- construir um projeto dividido com MVC.

## Seções de estudo

**1 –** O que é?
**2 –** Como implementar?

## 1- O que é?

O MVC é um padrão de arquitetura de software (architectural pattern) que separa a aplicação em três camadas de abstração: Model é a camada de manipulação dos dados; View é responsável pela interação do usuário; e Controller é a camada de controle (RAMOS, 2015).

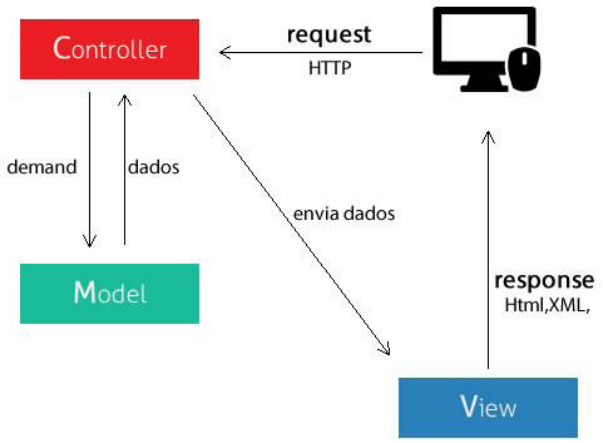


*Figura 1 – Arquitetura MVC.Fonte: RAMOS (2015).*

A Figura 1 mostra a representação gráfica dessa arquitetura.

Além de dividir responsabilidade entre as camadas, outra vantagem de se utilizar essa arquitetura está na manutenção do código, já que no padrão normal de desenvolvimento, com apenas um arquivo, a manutenção pode se tornar realmente cara e complicada (GONÇALVES, 2019).

**Model**

A camada de modelo corresponde às entidades e suas propriedades (no PHP representa as classes, atributos e métodos). Responsável pela manipulação (leitura e escrita) e validação dos dados (SILVA, 2014).

**View**

A camada de visão é responsável por qualquer tipo de retorno de dados, seja HTML, PDF, Json, XML, entre outros. Ela deve interagir com o usuário. Sua função é renderizar corretamente os dados, mesmo sem saber como obter nem quando renderizar (FIGUEIREDO, 2015).

**Controller**

A camada de controle é quem dita quando as coisas devem acontecer. Ela recebe todas as requisições do usuário e através dos seus métodos controla que modelo e que visão serão mostrados para o usuário (RAMOS, 2015).

## 2- Como implementar?

Para exemplificar a implementação de um projeto com a arquitetura MVC, utilizaremos o banco de dados de exemplo que está disponível na plataforma, representado graficamente na Figura 2.

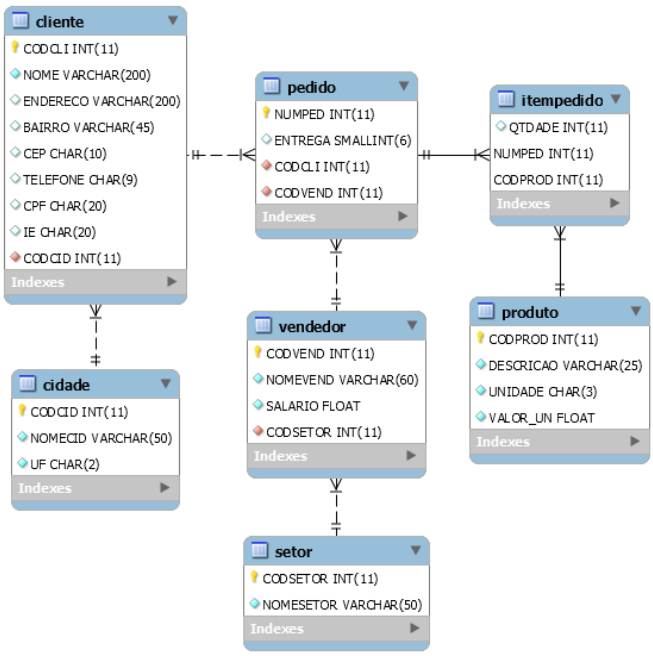


*Figura 2 – Estrutura BD. Fonte: Acervo pessoal.*

Criaremos os arquivos responsáveis pela obtenção e exibição das cidades do banco de dados. A estrutura dos arquivos que serão exemplificados abaixo segue a da Figura 3.

```
mvc
├── model/
│   └── Cidade.php
├── view/
│   └── cidade_view.php
├── controller/
│   └── CidadeController.php
└── index.php
```

*Figura 3 – Estrutura de arquivos MCV. Fonte: Acervo pessoal.*

O modelo é responsável pela abstração do banco de dados, primeiramente implementamos ele, com todos os métodos que irão acessar o banco de dados para inserir, ler, remover ou alterar os dados (WELENIR, 2011).

Dividiremos o modelo em dois trechos para melhor compreensão.

No primeiro a classe Cidade é definida, seus atributos $codcid, $nomecid, $uf e $conn, os três primeiros são as colunas do banco de dados da tabela cidade e o último é a variável para armazenar a conexão com o banco de dados. Depois de definidos os atributos, os métodos de gets e sets deles são codificados.

```
<?php
  /*
  Programa:  model/Cidade.php
   Autor:          Felipe Perez
   Data:           30/10/2019
  */

class Cidade{
  private $codcid;
  private $nomecid;
  private $uf;
   private $conn;

   function __construct() {
       $this->connectaBD();
   }

   public function getCodCid(){
       return $this->codcid;
   }
  public function setCodCid($codcid)
{
       $this->codcid=$codcid;
   }

   public function getNomeCid(){
             return $this->nomecid;
   }
 public function setNomeCid($nomecid)
{
       $this->nomecid=$nomecid;
   }

   public function getUF(){
              return $this->uf;
   }
   public function setUF($uf){
       $this->uf=$uf;
   }
```

Na sequência, são implementados os métodos de abstração do banco de dados como o connectaBD() e os para manipular como o create(), read(), update() e delete().

```
private function connectaBD(){
        $server = "localhost";
        $user = "root";
       $pass = ""; $mydb = "vendas";
    $this->conn = new mysqli($server,
$user, $pass, $mydb);
            if($this->conn->connect_
error){
              die("Conexão Falhou:
".$conn->connect_error);
        }
   }

  public function create(){
   $sql = "INSERT INTO cidade (NOMECID,
UF)  VALUES  ('".$this->getNomeCid()."',
'".$this->getUF()."')";
        $this->conn->query($sql);
   }

   public function update(){
        $sql = "UPDATE cidade SET
NOMECID='".$this->getNomeCid()."',
UF='".$this->getUF()."'          WHERE
CODCID=".$this->getCodCid();
        $this->conn->query($sql);
   }

   public function delete(){
        $sql = "DELETE FROM cidade
WHERE CODCID=".$this->getCodCid();
        $this->conn->query($sql);
   }

   public function read(){
      $sql = "SELECT * FROM cidade";
        $returnValue = array();

            $result  =  $this->conn-
>query($sql);

        if ($result != null) {
            while($row = $result-
>fetch_array(MYSQLI_ASSOC)){
              if (!empty($row)) {
                                array_
push($returnValue,$row);
              }
            }
        }
        return $returnValue;
   }

}
```

O modelo foi desenvolvido, então, com os atributos e métodos da classe Cidade, para poder ler, escrever, alterar e remover dados na tabela Cidade, do banco de dados de exemplo.

A camada visão recebe, então, os dados do controle para exibir para o usuário. No código abaixo, os códigos são recebidos pela variável global $_REQUEST, contendo um array dos objetos, e serão exibidos como uma tabela para o usuário, construída por um foreach.

```
<?php
  /*
  Programa:  view/cidade_view.php
   Autor:          Felipe Perez
   Data:           30/10/2019
  */

  $cidades = $_REQUEST['cidades'];
?>
<!DOCTYPE html>
<html lang="pt-br">
```

```
      <head>
            <title>Cidade MVC</title>
      </head>
      <body>
            <table>
                  <tr>
                        <th>CodCid</th>
                     <th>NomeCid</th>
                     <th>UF</th>
                  </tr>
            <?php foreach ($cidades as
$cidade): ?>
                  <tr>
                              <td><?php echo
$cidade["CODCID"]; ?></td>
                              <td><?php echo
$cidade["NOMECID"]; ?></td>
                              <td><?php echo
$cidade["UF"]; ?></td>
                  </tr>
       <?php endforeach; ?>
            </table>
       </body>
   </html>
```

A camada controle tem o papel de chamar o modelo que é responsável pela abstração do banco de dados e enviar os dados, quando recuperados, para a camada visão.

```
   <?php
      /*
      Programa:  c o n t r o l l e r /
CidadeController.php
       Autor:          Felipe Perez
       Data:           30/10/2019
      */
      require_once "model/Cidade.php";

      class CidadeController{
            public function listar(){
                  $cidade = new Cidade();
                  $cidades   =   $cidade-
>read();

                  $_REQUEST["cidades"] =
$cidades;

                  require_once     "view/
cidade_view.php";
            }
       }
   ?>
```

Para mostrar, então, todas as cidades em uma tabela, devemos criar um arquivo 'index.php' que faça a chamada à camada de controle, que por sua vez fará a ligação entre as camadas de modelo e visão.

```
   <?php
      /*
      Programa:  index.php
       Autor:          Felipe Perez
       Data:           30/10/2019
      */
      require_once          'controller/
CidadeController.php';

      $obj = new CidadeController();
      $obj->listar();

   ?>
```

O resultado dessa implementação pode ser observado na Figura 4, em que uma tabela é preenchida com todas as cidades encontradas no banco de dados.

| CodCid | NomeCid | UF |
|---|---|---|
| 1 | DOURADOS | MS |
| 2 | CAMPO GRANDE | MS |
| 3 | SÃO PAULO | SP |
| 4 | CUIABA | MT |
| 5 | FLORIANÓPOLIS | SC |
| 6 | SÃO SEBASTIÃO | SC |
| 7 | CAARAPÓ | MS |
| 8 | BRASÍLIA | DF |
| 10 | TUPÃ | SP |
| 11 | SÃO JOSÉ DO RIO PRETO | SP |
| 12 | APUCARANA | PR |
| 13 | JARDIM | MS |
| 14 | JATEI | MS |
| 15 | AMAMBAI | MS |

*Figura 4 – Resultado. Fonte: Acervo pessoal.*

## Retomando a aula

Chegamos ao final da sétima aula. Vamos recordar?

**1 – O que é?**

Na primeira seção, o MVC foi detalhado desde a divisão em três camadas até como efetuar a integração delas. Enquanto a camada de controle recebe as requisições do usuário, a camada de modelo cria uma abstração do banco de dados, para facilitar a recuperação e manipulação desses dados, e a camada de visão utiliza os dados recuperados para renderizar os elementos em HTML, ou outro formato.

**2 – Como implementar?**

Em seguida, uma implementação simples do MVC foi exemplificada. As três camadas da arquitetura MVC foram criadas para recuperar os dados da tabela Cidade no banco de dados. Para ampliar os estudos é interessante implementar as camadas para as outras tabelas do banco de dados.

## Vale a pena

### Vale a pena **ler**

SOARES, W. *Programação Web com PHP 5,* Érica, 2014.

BEIGHLEY, L; MORRISON, M. *Use a Cabeça!* PHP & MySQL. Alta Books, 2010.

SILVA, J. M. C. *PHP na prática 200 Exercícios resolvidos.* Elsevier, 2014.

BENTO, E. J. *Desenvolvimento web com PHP e MySQL.* Editora Casa do Código, 2014.

### Vale a pena **acessar**

WELENIR, W. In: Conceito de MVC e sua funcionalidade usando o PHP, 2011. Disponível em: https://www.devmedia.com.br/conceito-de-mvc-e-sua-funcionalidade-usando-o-php/22324. Acesso em: 30 out. 2019.

FIGUEIREDO, E. In: Entendendo o padrão MVC na prática, 2015. Disponível em: https://tableless.com.br/entendendo-o-padrao-mvc-na-pratica/. Acesso em: 30 out. 2019.

RAMOS, A. In: O que é MVC?, 2015. Disponível em: https://tableless.com.br/mvc-afinal-e-o-que/. Acesso em: 30 out. 2019.

GONÇALVES, J. In: Construindo um simples framework MVC com PHP, 2019. Disponível em: https://medium.com/@jardelgoncalves1996/construindo-um-simples-framework-mvc-com-php-349e9cacbeb1. Acesso em: 30 out. 2019.

## Minhas anotações