

## Desarrollo de Software VII

### Laboratorio # 21–Prof. Regis Rivera

Objetivo: Patrón de diseño Modelo-Vista-Controlador (MVC) desde PHP sin frameworks

#### Parte I – Introducción a MVC

##### Definición

Trygve Reenskaug inventó el patrón MVC (Model View Controller) a finales de los 70 para su posterior implementación en Smalltalk-80. La idea de este patrón era ser usado en pequeños componentes. Imaginemos un checkbox, sí, uno como éste:

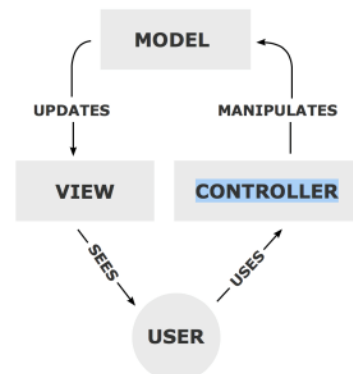
☐ «soy un checkbox»

Este checkbox posee:

- **Un modelo:** que se encarga de almacenar los datos del checkbox, en este caso un solo dato que puede ser verdadero (checked) o falso (not checked).
- **Una vista:** la representación gráfica del checkbox, es decir una cajita, la vista por supuesto toma los datos del modelo para saber cómo debe ser representada (con o sin marca). El usuario por supuesto ve e interactúa con la aplicación a través de la vista.
- **Un controlador:** El controlador se encarga de tomar una acción del usuario (por ejemplo el usuario presionó el checkbox) y actualizar el modelo (en este caso ahora el modelo contendrá «verdadero» porque el checkbox está presionado o volverá a «falso» si se presiona otra vez).

Una vez que se realicen cambios en **el modelo**, éste actualiza **la vista**. Gráficamente se representa:

Ésta es la razón por la cual hace años le costaba a los desarrolladores entender qué era MVC. Porque MVC se creó con la idea de representar pequeños componentes dentro de una aplicación para desktop, no para representar aplicaciones web modernas.



##### MVC en la actualidad

Hoy en día, para desarrollar una aplicación web de forma exitosa, necesitamos mucho más que 3 capas. Por ejemplo en MVC:

- No tenemos una manera de representar las **rutas** que se encargan de analizar las URLs y asignarlas a un método o función,
- No tenemos forma de representar un **middleware** que restringe ciertas áreas de la aplicación basado en el estado del usuario
- Una aplicación moderna **requiere mucho más que la representación de datos** a través de un modelo

Es por ello que muchos frameworks han tomado de referencia el MVC original y le han incorporado estos otros aspectos, entre ellos laravel. Prácticamente, MVC como tal ha evolucionado.

## Parte II – Creación de la base de datos y sus tablas

1. Crear la siguiente base de datos en MySQL: prodsmvc
2. Crear la siguiente tabla (puede copiar/pegar):

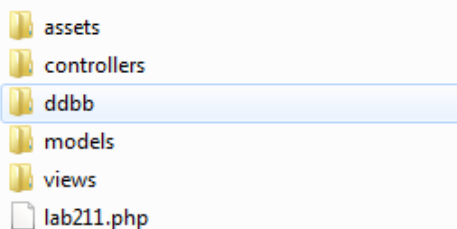
```
CREATE TABLE `products` (  
  `cod` int(11) NOT NULL,  
  `short_name` varchar(20) NOT NULL,  
  `pvp` decimal(5,2) NOT NULL,  
  `nombre` varchar(100) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

3. Agregarle los registros a la tabla (puede copiar/pegar):

```
INSERT INTO `products` (`cod`, `short_name`, `pvp`,  
  `nombre`) VALUES  
(1, 'Monitor', '400.00', 'Dell 21 full HD'),  
(2, 'Teclado', '9.99', 'Teclado inalámbrico Logitech'),  
(3, 'iPad Pro', '900.00', 'Apple iPad Pro 9');
```

## Parte III – Aplicación Web y su estructura

4. Descomprimir el adjunto y colocarlo bajo la carpeta lab21 (o laboratorio21, según su nomenclatura de laboratorios en clases)
5. Observar la estructura:



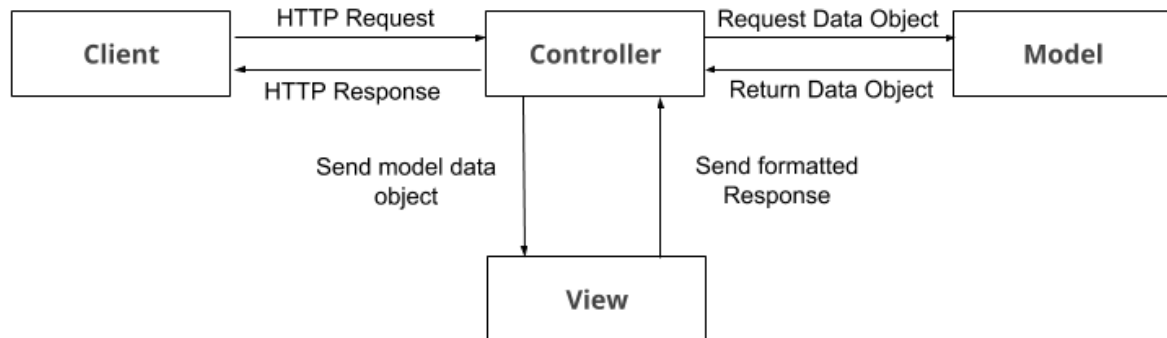
Conformada por un controlador, modelo, vista

Adicional una clase para la conexión a base de datos (ddbb) y css (assets)

6. Observar el código fuente enviado, su estructura, y llamados entre sí
7. Probar la aplicación, invocando a lab211.php, notar aparecen los registros de la tabla:

Lista de Productos	
<b>Codigo:</b> 1	<b>Nombre:</b> Dell 21 full HD <b>Descripcion breve:</b> Monitor <b>Precio:</b> USD\$ 400.00
<b>Codigo:</b> 2	<b>Nombre:</b> Teclado inalámbrico Logitech <b>Descripcion breve:</b> Teclado <b>Precio:</b> USD\$ 9.99
<b>Codigo:</b> 3	<b>Nombre:</b> Apple iPad Pro 9 <b>Descripcion breve:</b> iPad Pro <b>Precio:</b> USD\$ 900.00

## Parte IV – Arquitectura de la aplicación web bajo el patrón MVC



### Funcionamiento del MVC

- El usuario realiza una petición.
- El controlador captura la petición.
- El controlador hace la llamada al modelo correspondiente.
- El modelo interactúa con la base de datos.
- el controlador recibe la información del modelo (base de datos) y la envía a la vista.
- La vista muestra la información.

### Existen 3 niveles de abstracción:

- **Modelo:** Es quien define la lógica de negocio. Son las clases y los métodos que se comunican directamente con la base de datos.
- **Vista:** Muestra la información al usuario de manera lógica y legible.
- **Controlador:** Es el intermediario entre la vista y el modelo. Controla las interacciones del usuario en la vista. Pide los datos al modelo y los devuelve a la vista para que los muestre. Es el encargado de realizar las llamadas a las clases y los métodos.