

 UNIVERSIDAD DON BOSCO	<p style="text-align: center;"><b>UNIVERSIDAD DON BOSCO</b>  <b>FACULTAD DE ESTUDIOS TECNOLÓGICOS</b>  <b>COMPUTACIÓN</b></p>
<p style="text-align: center;"><b>CICLO 1</b></p>	<p style="text-align: center;"><i>GUIA DE LABORATORIO #10</i></p> <p><b>Nombre de la Practica:</b> Modelo, Vista y Controlador (MVC).  <b>Lugar de Ejecución:</b> Centro de cómputo  <b>Tiempo Estimado:</b> 2 horas con 30 minutos  <b>MATERIA:</b> Desarrollo de Aplicaciones Web con Software Interpretado en el Servidor</p>

## I. OBJETIVOS

Que el estudiante:

- Comprender el uso y división del patrón de desarrollo MVC.
- Aplicar correctamente el uso de la carpeta htaccess para las rutas limpias dentro de sus proyectos.

## II. INTRODUCCION TEORICA

### EL PATRÓN MVC

Cuando hablamos de arquitectura Modelo Vista Controlador nos referimos a un patrón de diseño que especifica cómo debe ser estructurada una aplicación, las capas que van a componer la misma y la funcionalidad de cada una. Según este patrón, la capa intermedia de una aplicación Web puede ser dividida en tres grandes bloques funcionales:

- Modelo
- Vista
- Controlador

**El Modelo:** En la arquitectura MVC la lógica de negocio de la aplicación, incluyendo el acceso a los datos y su manipulación, está encapsulada dentro del modelo. El Modelo lo forman una serie de componentes de negocio independientes del controlador y la vista, permitiendo así su reutilización y el desacoplamiento entre las capas.

- Representa las entidades de negocio o los datos y encapsula en ellos el estado de la aplicación.
- Responde a preguntas sobre su estado.
- Notifica a la vista sus cambios.

**La Vista:** Tal y como se puede deducir de su nombre, la vista es la encargada de generar las respuestas que debe ser enviadas al cliente. Cuando esta respuesta tiene que incluir datos proporcionados por el controlador, el código HTML de la página no será fijo si no que deberá ser generado de forma dinámica,

- Visualiza el modelo. Es su representación gráfica.
- Se encarga de la interacción con el usuario. Envía la información necesaria al controlador.
- Recibe notificación de las modificaciones del modelo, para que la vista pueda ser actualizada.
- Permite al controlador seleccionar la vista, ya que puede haber varias.

**El Controlador:** Se puede decir que el Controlador es el “cerebro” de la aplicación. Todas las peticiones a la capa intermedia que se realicen desde el cliente son dirigidas al Controlador, cuya misión es determinar las acciones a realizar para cada una de estas peticiones e invocar al resto de los componentes de la aplicación

(Modelo y Vista) para que realicen las acciones requeridas en cada caso, encargándose también de la coordinación de todo el proceso.

Por ejemplo, en el caso de que una petición requiera enviar como respuesta al cliente determinada información existente en una base de datos, el Controlador solicitará los datos necesarios al modelo y una vez recibidos, se los proporcionará a la Vista para que ésta les aplique el formato de presentación correspondiente y envíe la respuesta al cliente. La centralización del flujo de peticiones en el controlador proporciona varias ventajas al programador, entre ellas:

- Hace que el desarrollo sea más sencillo y limpio.
- Facilita el posterior mantenimiento de la aplicación haciéndola más escalable.
- Facilita la detección de errores en el código.

Resumiendo, el controlador:

- Encaminan las peticiones del usuario.
- Define el comportamiento de la aplicación.
- Convierte las acciones del usuario en modificaciones del modelo.
- Selecciona la vista a devolver.

### III. MATERIALES Y EQUIPO

Para la realización de la guía de práctica se requerirá lo siguiente:

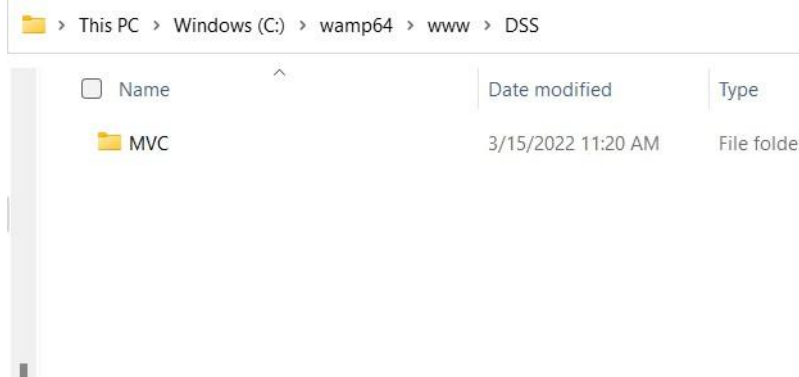
No.	Requerimiento	Cantidad
1	Guía de práctica #10: MVC	1
2	Computadora con Wamp Server y Sublime Text instalados	1
3	Memoria USB o disco flexible	1

### IV. PROCEDIMIENTO

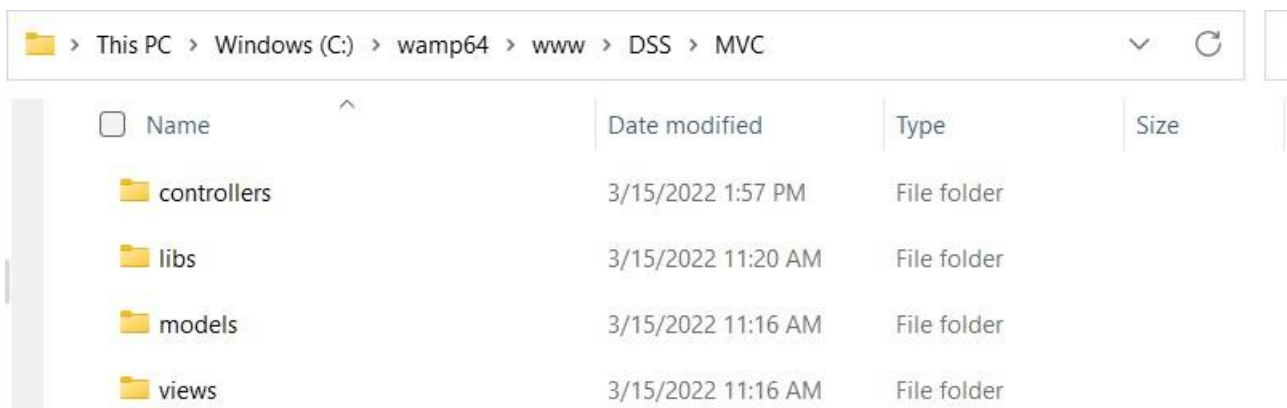
#### Implementación básica del MVC.

##### Paso 1: configuración de rutas limpias, controladore y vistas

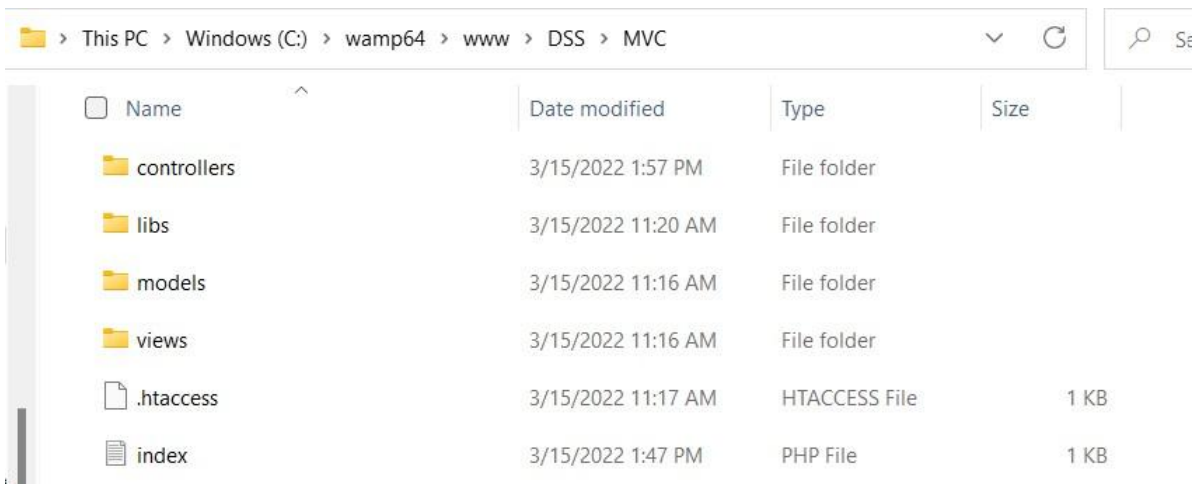
- 1- Crear el directorio MVC, dentro de su carpeta www o httdocs.



- 2- Deberá crear las carpetas **libs**, **controllers**, **models** y **views**, dentro de su proyecto.



- 3- Procederemos a crear el archivo al cual llamaremos **.htaccess** e **index.php**, dentro del directorio principal del proyecto. Finalmente, lo que tendrá serán los siguientes directorios presentados en la figura.



- 4- Dentro del archivo de configuración **.htaccess**, escribirá las siguientes líneas de código.  
**NOTA:** Podrá notar que este archivo manda a llamar el archivo **index.php** con el parámetro **URL**, notará entonces que a pesar de escribir cualquier parámetro después de el nombre principal del proyecto, este siempre invocará a **index.php**. Haremos las pruebas en siguientes pasos. Es importante recalcar que este es un archivo de configuración, por lo cual no lleva extensión.

```
RewriteEngine On
RewriteRule ^(.*)$ index.php?url=$1 [L,QSA]
```

- 5- Crearemos ahora dentro de la carpeta **libs** el archivo **app.php** y digitaremos el siguiente código.

```
<?php
require_once 'controllers/ErrorController.php';
class App{

    function __construct(){

        $url = isset($_GET["url"]) ? $_GET["url"] : null;//obtenemos todo lo que venga despues de
        localhost/dss/mvc y validamos que si el campo viene vacio este sera null
    }
}
```

```
$url = rtrim($url,'/');//limpiamos la url para evitar parametros como
localhost/dss/mvc/////Controller
$url = explode('/', $url);//convertimos a un arreglo todo lo que venga despues de la raiz del
proyecto

if(empty($url[0])){
    $archivoController = 'controllers/MainController.php';//creamos la url para invocar el archivo
del controlador
    require_once($archivoController);
    $controller = new MainController();
    return false;
}

$archivoController = 'controllers/' . $url[0]. '.php';//creamos la url para invocar el archivo del
controlador

if(file_exists($archivoController)){//validamos que el archivo exista
    require_once $archivoController;//invocamos el archivo con el controlador
    $controller = new $url[0];//instanciamos la clase del controlador

    if(isset($url[1])){
        $controller->{$url[1]}();//aqui invocaremos a las acciones o funciones de cada controller
    }

}else{
    $controller = new ErrorController();//de lo contrario invocamos a controller de error
}

}

}

?>
```

6- Ahora, siempre dentro de la carpeta **libs**, crearemos los recursos **controller.php** y **view.php**

- **controller.php**

```
<?php

class Controller{

    function __construct(){
        echo "<p>soy el controller base</p>";
        $this->view = new View();
    }

}
```

```
?>
```

- **view.php**

```
<?php

class View{

    function __construct(){
        echo "<p>soy la vista base</p>";
    }

    function renderView($vista){
        require 'views/' . $vista;
    }

}
```

- 7- Procederemos ahora a crear nuestros primeros controllers, con lo cual deberemos crear dos archivos a los cuales llamaremos **MainController.php** y **ErrorController.php**. Esto lo haremos dentro del directorio **controllers**.

- **MainController.php**

```
<?php

class MainController extends Controller{

    function __construct(){
        parent::__construct();
        $this->view->mensaje1= "Parametro enviado a la vista";
        $this->view->renderView('main/main.php');
    }

    function accion(){
        echo "<p>hola soy la accion</p>";
    }

}

?>
```

- **ErrorController.php**

```
<?php

class ErrorController extends Controller{

    function __construct(){
```

```
        parent::__construct();
        $this->view->mensaje= "Error al cargar la pagina o recurso";
        $this->view->renderView('error/error.php');

    }

}
```

?>

8- Finalmente crearemos el archivo **index.php** en el directorio principal.

```
<?php

require_once "libs/controller.php";
require_once "libs/view.php";
require_once "libs/app.php";

$app = new App();

?>
```

9- Crearemos el directorio **main** y **error** dentro de la carpeta **views**, y finalmente agregara sus primeras vistas, a las cuales llamara **main.php** y **error.php**.

- **Main.php**

```
<!DOCTYPE html>

<html>
  <head>
    <meta charset="utf-8">
    <title>Vista main</title>
  </head>
  <body>
    <h1>Soy la vista main</h1>
    <p><?php echo $this->mensaje1; ?> </p>
  </body>
</html>
```

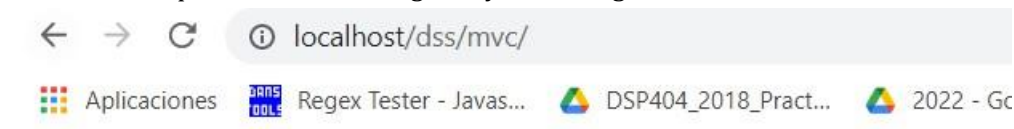
- **Error.php**

```
<!DOCTYPE html>

<html>
  <head>
    <meta charset="utf-8">
    <title>Vista error</title>
  </head>
```

```
<body>
  <h1 style="color: red;"><b>Error.php:</b>  <?php echo $this->mensaje; ?></h1>
</body>
</html>
```

10- Corra su aplicación en el navegador y vera lo siguiente.

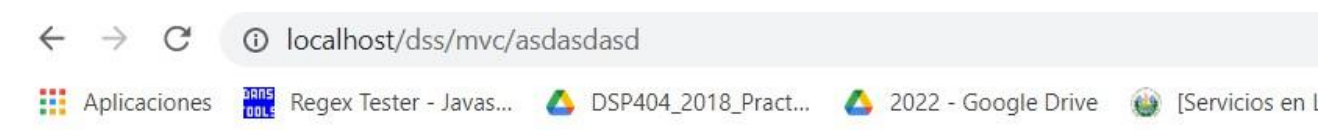


soy el controller base

soy la vista base

## Soy la vista main

Parametro enviado a la vista



soy el controller base

soy la vista base

## Error.php: Error al cargar la pagina o recurso

### Aclaraciones:

- Notara que por el momento no se han creado modelos, pero estamos planteando la estructura base de nuestro proyecto.
- App.php es el archivo quien gestiona todas las peticiones por la url.
- Las redirecciones a las vistas no usan mas que el concepto de herencia y el comando requiere para invocar el código externo.

### Implementación del Modelo y conexiones a base de datos.

**Paso 1:** Creación de la base de datos.

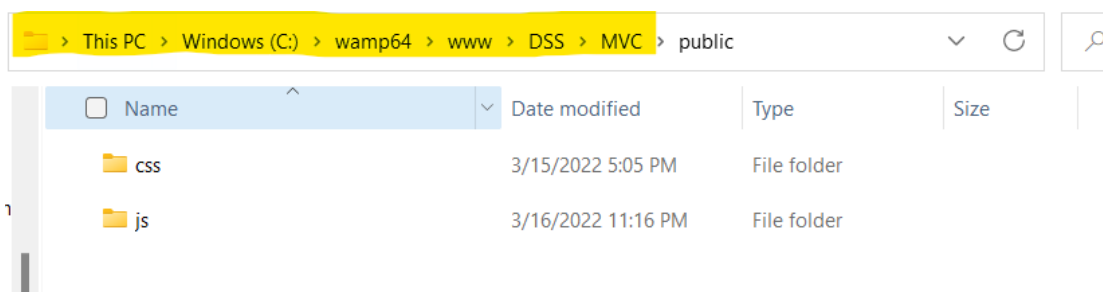
1- Crear la base de datos. Digite el siguiente script.

```
CREATE DATABASE personabdd;
USE personabdd;
```

```
CREATE TABLE ocupaciones(  
id_ocupacion int(11) NOT NULL AUTO_INCREMENT,  
ocupacion varchar(50) NOT NULL,  
PRIMARY KEY (`id_ocupacion`)  
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT  
CHARSET=latin1;  
  
CREATE TABLE persona (  
id_persona int(11) NOT NULL AUTO_INCREMENT,  
nombre_persona varchar(100) NOT NULL,  
edad_persona int(11) NOT NULL,  
telefono_persona varchar(9) NOT NULL,  
sexo_persona varchar(50) NOT NULL,  
id_ocupacion int(11) NOT NULL,  
fecha_nac date NOT NULL,  
PRIMARY KEY (`id_persona`),  
KEY id_ocupacion (id_ocupacion)  
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT  
CHARSET=latin1;  
  
ALTER TABLE persona  
ADD CONSTRAINT persona_ibfk_1  
FOREIGN KEY (id_ocupacion)  
REFERENCES ocupaciones (id_ocupacion);  
  
INSERT INTO `ocupaciones` (`id_ocupacion`, `ocupacion`)  
VALUES(1, 'Doctor'), (2, 'Emprendedor'), (3, 'Profesor');  
  
INSERT INTO persona (id_persona, nombre_persona, edad_persona, telefono_persona,  
sexo_persona, id_ocupacion, fecha_nac) VALUES  
(1, 'Alejandro Pineda', 45, '7722-4455', 'Masculino', 1, '1999-01-05'),  
(2, 'Fernando Calderón', 21, '7667-7890', 'Masculino', 2, '2001-05-07'),  
(3, 'Emerson Torres', 22, '7123-9800', 'Masculino', 3, '1999-08-03');
```

**Paso 2:** crearemos los archivos en los directorios creados al inicio de la guía. También se modificarán algunos de estos para que sean funcionales con el modelo.

- 1- Crearemos el directorio public en la raíz del proyecto, donde guardaremos los recursos proporcionados para el desarrollo de la guía. Se le sombreará la raíz del proyecto.



- 2- Modificar el archivo **.htaccess**, creado al inicio de la guía.



**Nota:** Notara que asignamos 3 reglas; las cuales definen el acceso a directorios, archivos y otros; esto para ser visibles dentro de mi proyecto, acceder a la carpeta public y mis recursos.

```
RewriteEngine On
```

```
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-l
RewriteRule ^(.*)$ index.php?url=$1 [L,QSA]
```

3- Modificaremos el archivo **index.php** creado al inicio de la guía.

**Nota:** Notara que en este lugar se importan los otros archivos de configuración. Recaltar que este **siempre se ejecutara**, por ende la instancia de App será quien decida el rumbo de mi navegación.

```
<?php

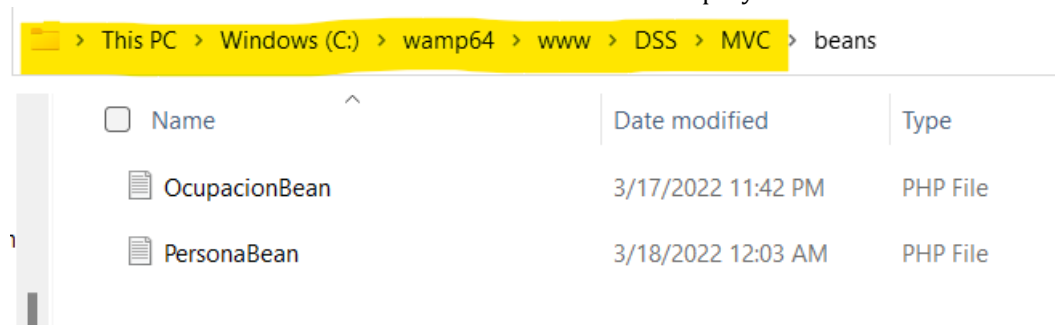
require_once "libs/conexion.php";
require_once "libs/controller.php";
require_once "libs/model.php";
require_once "libs/view.php";
require_once "libs/app.php";

require_once "config/config.php";

$app = new App();

?>
```

4- Ahora crearemos un nuevo directorio en la raíz de nuestro proyecto al cual llamaremos **beans**.



5- Notara que hay dos recursos dentro de este fichero. Crearemos entonces dentro **OcupacionBean.php** y **PersonaBean.php**.

**Nota:** En efecto notara que ellos guardan el mapeo de mis entidades dentro de mi bdd.

- **OcupacionBean.php**

```
<?php

class OcupacionBean
{
```

```
private $idOcupacion;
private $ocupacion;

function setIdOcupacion($idOcupacion){
    $this->idOcupacion=$idOcupacion;
}
function getIdOcupacion(){
    return $this->idOcupacion;
}

function setOcupacion($ocupacion){
    $this->ocupacion=$ocupacion;
}
function getOcupacion(){
    return $this->ocupacion;
}
}

?>
```

- **PersonaBean.php**

```
<?php

class PersonaBean
{

    private $idPersona;
    private $nombre;
    private $edad;
    private $telefono;
    private $sexo;
    private $ocupacion;
    private $fecha;

    function setIdPersona($idPersona)
    {
        $this->idPersona = $idPersona;
    }
    function getIdPersona()
    {
        return $this->idPersona;
    }

    function setNombre($nombre)
    {
```

```
$this->nombre = $nombre;
}
function getNombre()
{
    return $this->nombre;
}

function setEdad($edad)
{
    $this->edad = $edad;
}
function getEdad()
{
    return $this->edad;
}

function setTelefono($telefono)
{
    $this->telefono = $telefono;
}
function getTelefono()
{
    return $this->telefono;
}

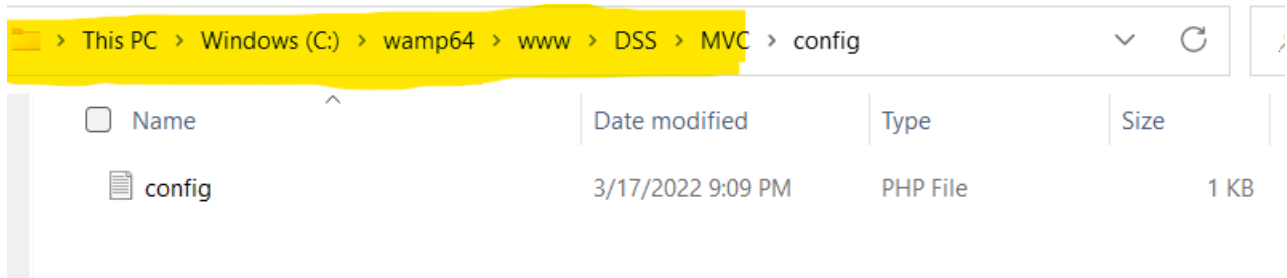
function setSexo($sexo)
{
    $this->sexo = $sexo;
}
function getSexo()
{
    return $this->sexo;
}

function setOcupacion($ocupacion)
{
    $this->ocupacion = $ocupacion;
}
function getOcupacion()
{
    return $this->ocupacion;
}

function setFecha($fecha)
{
    $this->fecha = $fecha;
}
```

```
function getFecha()
{
    return $this->fecha;
}
}
?>
```

6- Ahora crearemos directorio **config**.



Name	Date modified	Type	Size
config	3/17/2022 9:09 PM	PHP File	1 KB

7- Notara que también hay un archivo al cual llamaremos **config.php**.

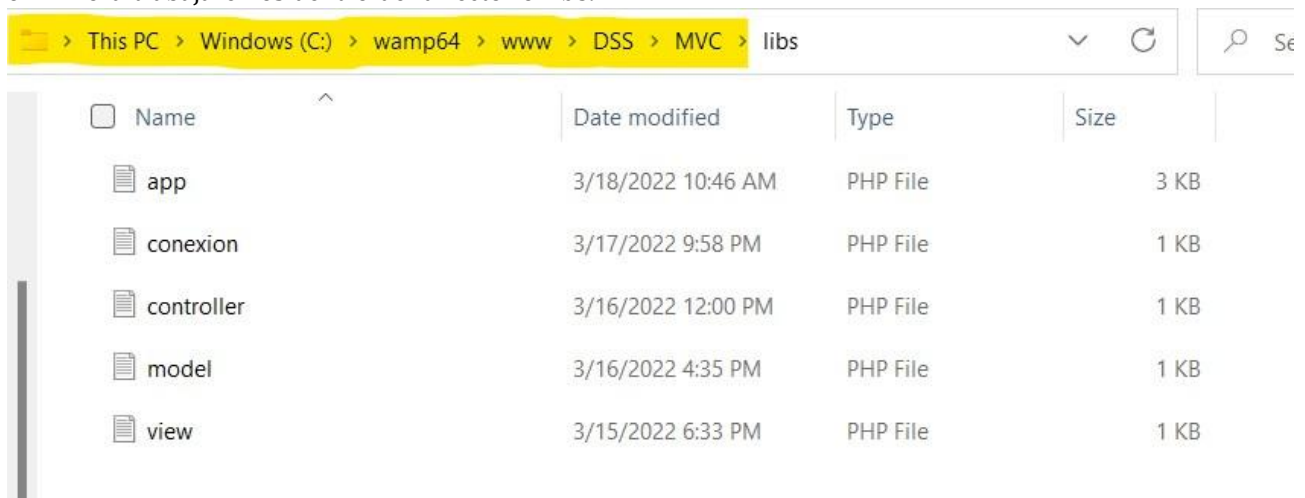
**Nota:** Aquí guardaremos las configuraciones básicas del proyecto. Nótese que si queremos migrar nuestro sistema, bastara con que cambiemos los parámetros en este archivo.

```
<?php

define('URL', 'http://localhost/dss/mvc/');
define('HOST', 'localhost');
define('DB', 'personabdd');
define('USER', 'root');
define('PASSWORD', null);

?>
```

8- Ahora trabajaremos dentro del directorio **libs**.



Name	Date modified	Type	Size
app	3/18/2022 10:46 AM	PHP File	3 KB
conexion	3/17/2022 9:58 PM	PHP File	1 KB
controller	3/16/2022 12:00 PM	PHP File	1 KB
model	3/16/2022 4:35 PM	PHP File	1 KB
view	3/15/2022 6:33 PM	PHP File	1 KB

9- Modificaremos el archivo **app.php**.

```
<?php
require_once 'controllers/ErrorController.php';
class App{

    function __construct(){

        $url = isset($_GET["url"]) ? $_GET["url"] : null;//obtenemos todo lo que venga despues de
        localhost/dss/mvc y validamos que si el campo viene vacio este sera null

        $url = rtrim($url,'');//limpiamos la url para evitar parametros como
        localhost/dss/mvc/////Controller
        $url = explode('/', $url);//convertimos a un arreglo todo lo que venga despues de la raiz del
        proyecto

        if(empty($url[0])){
            $archivoController = 'controllers/MainController.php';//creamos la url para invocar el archivo
            del controlador
            require_once($archivoController);
            $controller = new MainController();
            $controller->loadModel('Main');
            $controller->principal();
            return false;
        }

        $archivoController = 'controllers/' . $url[0]. 'Controller.php';//creamos la url para invocar el
        archivo del controlador

        $clase= $url[0]. 'Controller';//nombre de la clase a instaciarse MainController
        if(file_exists($archivoController)){//validamos que el archivo exista
            require_once $archivoController;//invocamos el archivo con el controlador
            $controller = new $clase;//instanciamos la clase del controlador
            $controller->loadModel($url[0]);
            if(isset($url[1])){

                if(method_exists($controller,$url[1])){//verificamos si el metodo existe en el controller
                    if(isset($url[2])){//verificamos si viene un parametro metodo o funcion del controller
                        $controller->{$url[1]}($url[2]);
                    }
                    $controller->{$url[1]}();//aqui invocaremos a las acciones o funciones de cada controller
                }else{
                    $controller = new ErrorController();//de lo contrario invocamos a controller de error
                }

            }else{
                $controller = new ErrorController();//de lo contrario invocamos a controller de error
            }
        }
    }
}
```

```
    }else{
        $controller = new ErrorController();//de lo contrario invocamos a controller de error
    }

}

}

?>
```

10- Modificaremos el archivo **controller.php**.

```
<?php

class Controller{

    function __construct(){//cada instancia de controller o invocacion a este instanciara una nueva vista

        $this->view = new View();
    }

    function loadModel($model){//se manda a llamar para cargar el modelo, notara que cada controller
    esta ligado a cada model
        $url = 'models/'.$model.'Model.php';// Para este caso el MainController y MainModel estan
    estechamente ligados
        if(file_exists($url)){
            require $url;
            $modelName = $model. 'Model';
            $this->model = new $modelName;
        }
    }

}

?>
```

11- Modificaremos el archivo **view.php**.

```
<?php

class View{

    function __construct(){

    }

    function renderView($vista){//Notara que nunca hacemos un redirect puntual a una vista
        require 'views/' . $vista; // Entonces llamamos ese codigo y añadimos el recurso vista
    }

}

}
```

12- Ahora crearemos el archivo **conexion.php**.

```
<?php

class Conexion{

    private $host;
    private $username;
    private $password;
    private $bd;

    function __construct(){
        $this->host=constant('HOST');//llamamos las constantes del archivo config.php
        $this->username=constant('USER');
        $this->password=constant('PASSWORD');
        $this->bd=constant('DB');
    }

    function conectar(){
        try{
            $con = new PDO("mysql:dbname=$this->bd;host=$this->host", $this->username,$this->password);//usaremos la libreria o clase PDO para las conexiones
            return $con;//retornaremos la bdd, que en este caso sera personabdd
        }catch(Exception $e){
            $error = 'Error encontrado en conexión de Base de datos :(' . $e->getMessage(). "\n";
            return $error;
        }
    }

    function desconectar($conexion){
        try{
            $conexion->query('KILL CONNECTION_ID()');//eliminaremos cualquier conexion
            $conexion = null;//haremos null la bdd, para dar fin a cualquier comunicacion al servicio de mysql
        }catch(Exception $e){
            $error = 'Error encontrado en conexión de Base de datos :(' . $e->getMessage(). "\n";
            return $error;
        }
    }
}

?>
```

13- Finalmente crearemos el archivo **model.php**.

```
<?php

class Model{

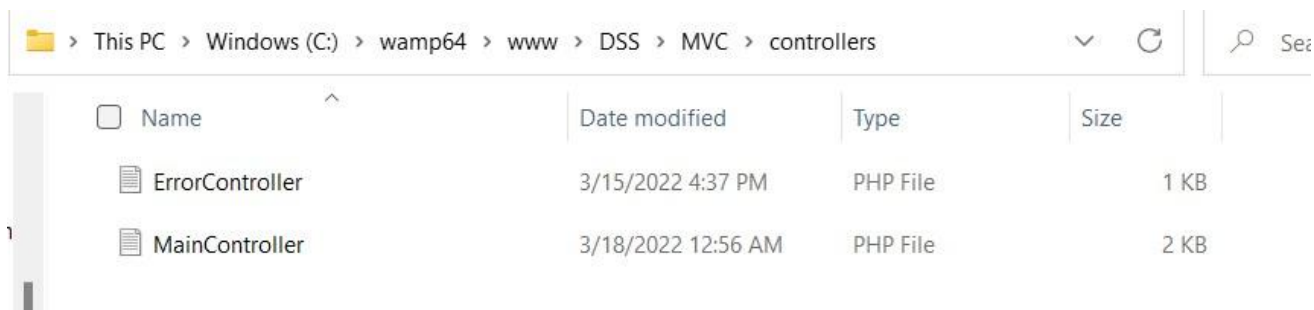
    function __construct(){
```

```
$this->con= new Conexion();//instanciamos la clase conexion, para que cada vez que accedamos a este constructor invoquemos una conexion diferente

}

}
```

14- Ahora trabajaremos con el directorio **controllers**.



15- Modificaremos el archivo **MainController.php**.

```
<?php

class MainController extends Controller{//extenderemos de controller para poder acceder a sus funciones

    function __construct(){
        //parent::__construct(); codigo de la primera parte de la guia
        //$this->view->mensaje1= "Parametro enviado a la vista";
        //$this->view->renderView('main/main.php');
    }

    function principal(){
        parent::__construct();//llamamos el constructor de Controller, para poder acceder a la instancia de view
        $this->view->listaPersonas= $this->model->listaPersonas();//enviamos arreglos de objetos a las vistas
        $this->view->listaOcupaciones= $this->model->listaOcupaciones();
        $this->view->persona= $this->model->obtenerPersona();//enviamos un objeto en particular a la vista
        $this->view->listaOcupaciones2= $this->model->listaOcupaciones();
        $this->view->renderView('main/main.php');//llamando al metodo renderView para pintar la vista
    }

    function agregarPersona(){
        parent::__construct();
        $nombre = $_POST["nombre"];
        $edad = $_POST["edad"];
        $telefono = $_POST["telefono"];
```



```
$sexo = $_POST["sexo"];
$ocupacion = $_POST["ocupacion"];
$fecha = $_POST["fecha"];
$this->model->agregarPersona($nombre,$edad,$telefono,$sexo,$ocupacion,$fecha);//invocamos al
model y a la funcion del modelo
    header('Location:'. constant('URL')."Main/principal");//notese el redirect al metodo principal, esto
para no enviar nuevamente los parametros de ese metodo
}
function eliminarPersona($id){
    header('Location:'. constant('URL')."Main/principal");
}
}
?>
```

16- Modificaremos el archivo **ErrorController.php**.

```
<?php

class ErrorController extends Controller{

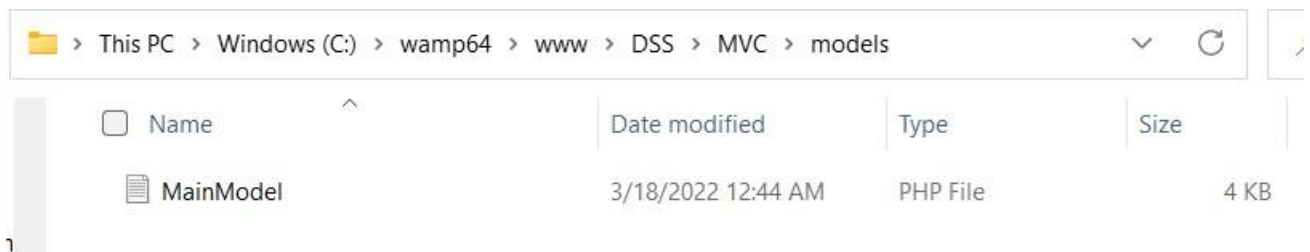
    function __construct(){
        parent::__construct();
        $this->view->mensaje= "Error al cargar la pagina o recurso";
        $this->view->renderView('error/error.php');

    }

}

?>
```

17- ¡Casi terminamos!, ahora trabajaremos con el directorio **models**.



18- Crearemos el archivo **MainModel.php**.

```
<?php
require "beans/PersonaBean.php";//llamamos los recursos beans, para acceder a sus clases.
```

```
require "beans/OcupacionBean.php";//llamamos los recursos beans, para acceder a sus clases.
class MainModel extends Model
{
    function __construct()
    {
        parent::__construct();//accedemos al constructor de Model, para poder usar la bdd
    }

    function listaPersonas()
    {
        $query = "SELECT * FROM persona a INNER JOIN ocupaciones o ON a.id_ocupacion = o.id_ocupacion";
        $this->conexion = $this->con->conectar();//accedemos a la funcion conectar, y por ende su return, el cual recordara es la bdd
        $resultado = $this->conexion->prepare($query); //preparamos la consulta
        $resultado->execute();//ejecutamos la consulta

        $array =array();
        while ($row = $resultado->fetch()) { //obtenemos los resultados de la consulta, aqui se convertiran en arreglos nativos de php que podemos recorrer y usar
            $persona = new PersonaBean();//instanciamos una nueva persona
            $ocupacion = new OcupacionBean();//instanciamos una nueva ocupacion, recordara que en Persona Bean hay un atributo llamado Ocupacion
            $persona->setIdPersona($row['id_persona']);
            $persona->setNombre($row['nombre_persona']);
            $persona->setEdad($row['edad_persona']);
            $persona->setTelefono($row['telefono_persona']);
            $persona->setSexo($row['sexo_persona']);
            $persona->setFecha($row['fecha_nac']); //setiamos los valore de persona
            $ocupacion->setOcupacion($row['ocupacion']);
            $ocupacion->setIdOcupacion($row['id_ocupacion']);
            $persona->setOcupacion($ocupacion);//finalmente cargamos el atributo ocupacion con un objeto ocupacion
            $array[] = $persona;//cargamos el arreglo
        }
        $this->con->desconectar($this->conexion);//cerramos la conexion
        return $array;//retornamos el arreglo
    }

    function listaOcupaciones()
    {
        $query = "SELECT * FROM ocupaciones";
        $this->conexion = $this->con->conectar();
        $resultado = $this->conexion->prepare($query);
        $resultado->execute();

        $array =array();
        while ($row = $resultado->fetch()) { //obtenemos los resultados de la consulta, aqui se convertiran en arreglos nativos de php que podemos recorrer y usar
```

```
$ocupacion = new OcupacionBean();
$ocupacion->setOcupacion($row['ocupacion']);
$ocupacion->setIdOcupacion($row['id_ocupacion']);

$array[] = $ocupacion;//cargamos el arreglo
}

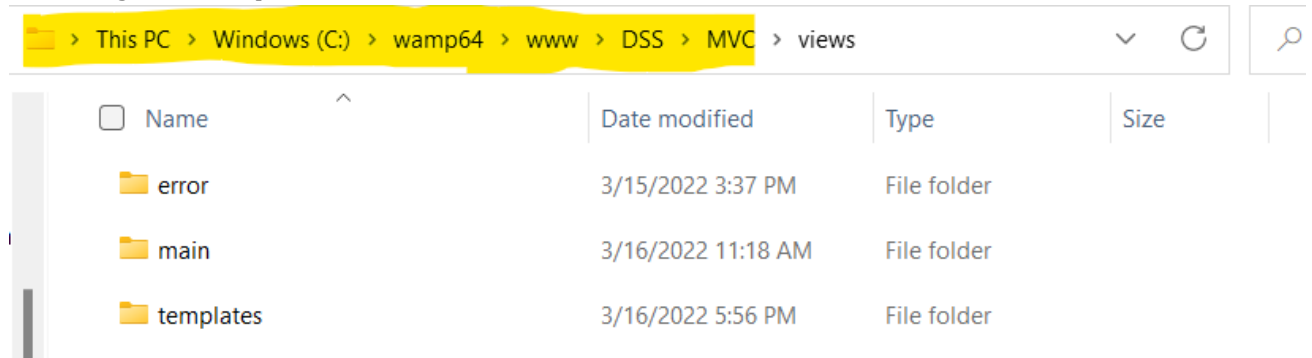
$this->con->desconectar($this->conexion);
return $array;
}

function agregarPersona($nombre, $edad, $telefono, $sexo, $ocupacion, $fecha)
{
    $query = "INSERT INTO persona (nombre_persona, edad_persona, telefono_persona,
    sexo_persona, id_ocupacion, fecha_nac) VALUES(:nombre,:edad,:telefono,:sexo,:ocupacion,:fecha)";
    $this->conexion = $this->con->conectar();
    $row = $this->conexion->prepare($query);
    $row->bindParam(':nombre', $nombre);//enviamos parametros a la consulta, esto evita inyecciones
    SQL
    $row->bindParam(':edad', $edad);
    $row->bindParam(':telefono', $telefono);
    $row->bindParam(':sexo', $sexo);
    $row->bindParam(':ocupacion', $ocupacion);
    $row->bindParam(':fecha', $fecha);
    return $row->execute();//devolvera un booleano dependiendo si la consulta y conexion fue exitosa
}

function obtenerPersona()
{
    $id = 1;
    $query = "SELECT * FROM persona WHERE id_persona = :valor";
    $this->conexion = $this->con->conectar();
    $row = $this->conexion->prepare($query);
    $row->bindParam(':valor', $id);
    $row->execute();
    while ($row = $row->fetch()) {
        $persona = new PersonaBean();
        $ocupacion = new OcupacionBean();
        $persona->setIdPersona($row['id_persona']);
        $persona->setNombre($row['nombre_persona']);
        $persona->setEdad($row['edad_persona']);
        $persona->setTelefono($row['telefono_persona']);
        $persona->setSexo($row['sexo_persona']);
        $persona->setFecha($row['fecha_nac']);
        $ocupacion->setOcupacion(null);
        $ocupacion->setIdOcupacion($row['id_ocupacion']);
        $persona->setOcupacion($ocupacion);
        return $persona;
    }
    return $persona;
}
```

```
}  
}  
?>
```

- 19- Finalmente crearemos todas las vistas de nuestra web, para ello crearemos dentro del directorio **views** las siguientes carpetas.



- **error:** Guardara la vista predefinida para cualquier error en el ruteo de la web.
- **main:** Tendrá la vista de la pagina principal de mi proyecto.
- **templates:** Guardara las porciones de código html estáticas, que solo deberé llamar a mi view para poder usarla.

- 20- Iniciaremos con los recursos dentro de **templates**. Crearemos las paginas **footer.php**, **header.php**, **modal.php** y **nav.php**. Notara que este es el contenido estático de la vista, por ende, cualquier plantilla que use, bastara con que modifique los recursos para cambiar el apartado visual.

- **footer.php**

```
<footer id="sticky-footer" class="flex-shrink-0 py-4 bg-dark text-white-50">  
  <div class="container text-center ">  
    <small style="color: white;">Copyright &copy; MVC</small>  
  </div>  
</footer>  
<script src="<?php echo constant('URL') ?>public/js/funcionesform.js"></script>  
<script>  
  function alerta(id)  
  {  
    var mensaje;  
    var opcion = confirm("Esta seguro de eliminar este registro");  
    if (opcion == true) {  
      location.href = '<?php echo constant('URL') ?>Main/eliminarPersona/'+id;  
    }  
  }  
  document.getElementById('sexo').value = '<?php echo $this->persona->getSexo(); ?>';  
  document.getElementById('ocupacion').value = '<?php echo $this->persona->getOcupacion()-  
>getIdOcupacion(); ?>';  
</script>
```

- **header.php**

```
<html lang="es">
<head>

    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Main
    </title>
    <link rel="stylesheet" href="<?php echo constant('URL') ?>public/css/bootstrap.min.css">
    <link rel="stylesheet" href="<?php echo constant('URL') ?>public/css/estilo.css">
    <link          rel="stylesheet"          href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <script src="<?php echo constant('URL') ?>public/js/jquery-3.2.1.slim.min.js"></script>
    <script src="<?php echo constant('URL') ?>public/js/bootstrap.min.js"></script>

</head>
```

- **modal.php.**

```
<!-- Modal -->
<div      class="modal      fade"      id="exampleModal"      tabindex="-1"      aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Agregar persona</h5>
                <button      type="button"      class="btn-close"      data-bs-dismiss="modal"      aria-
label="Close">X</button>
            </div>
            <div class="modal-body">
                <form      role="form"      id="formactualiza"      action="<?php      echo      constant("URL")
?>Main/agregarPersona" method="POST">
                    <div class="col-md-12" id="conten">
                        <input type="hidden" name="id" id="idpersona">

                        <div class="form-group">
                            <label for="nombre">Ingrese el nombre de la persona:</label>
                            <div class="input-group">
                                <input type="text" class="form-control" name="nombre" placeholder="Ingresa
el nombre" required>

                            </div>
                        </div>
                        <div class="form-group">
                            <label for="edad">Ingrese la edad de la persona:</label>
                            <div class="input-group">
                                <input type="number" class="form-control" name="edad" placeholder="Ingresa
la edad" required>
                            </div>
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>
```

```
</div>
<div class="form-group">
  <label for="edad">Ingrese el telefono de la persona:</label>
  <div class="input-group">
    <input type="tel" class="form-control" name="telefono" placeholder="Ingresa
el telefono" required>
  </div>
</div>
<div class="form-group">
  <label for="sexo">Ingrese el sexo de la persona:</label>
  <div class="input-group">
    <select name="sexo" class="form-control" required>
      <option value="Masculino">Maculino</option>
      <option value="Femenino">Femenino</option>
    </select>
  </div>
</div>
<div class="form-group">
  <label for="ocupacion">Ingrese la ocupacion de la persona:</label>
  <div class="input-group">
    <select name="ocupacion" class="form-control" required>
      <?php
foreach($this->listaOcupaciones2 as $lista ){
  ?>
    <option value="<?php echo $lista->getIdOcupacion(); ?>"><?php echo $lista-
>getOcupacion(); ?></option>
      <?php
      }
      ?>
    </select>
  </div>
</div>
<div class="form-group">
  <label for="fecha">Ingrese la fecha de nacimiento de la persona:</label>
  <div class="input-group">
    <input type="date" class="form-control" name="fecha" placeholder="Ingresa la
fecha" required>
  </div>
</div>
</div>
</form>

</div>
<div class="modal-footer">

  <input type="submit" name="submit" class="btn btn-success btn-md" form="formactualiza"
value="Enviar">
</div>
</div>
</div>
</div>
```

- **nav.php**

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top" role="navigation">
  <div class="container">
    <a class="navbar-brand" href="#">Guia MVC</a>
    <button class="navbar-toggler border-0" type="button" data-toggle="collapse" data-
target="#exCollapsingNavbar">
      &#9776;
    </button>
    <div class="collapse navbar-collapse" id="navbarNavDropdown">
    <ul class="navbar-nav">
      <li class="nav-item active">
        <a style="color: white; cursor:pointer;" type="button" data-bs-toggle="modal" data-bs-
target="#exampleModal">Agregar persona <i class="fa fa-user"></i></a>
      </li>

    </ul>
    </div>
  </div>
</nav>
```

21- Crear dentro de **error** el archivo **error.php**.

```
<!DOCTYPE html>

<?php require "views/templates/header.php"; ?>

<body>

<h1 style="color: red; padding-top:15%;" class="text-center"><b>Error.php:</b> <?php echo $this-
>mensaje; ?></h1>

</body>

<footer id="sticky-footer" style="bottom: 0; position: absolute; width: 100%;" class="flex-shrink-0 py-4
bg-dark text-white-50">
  <div class="container text-center ">
    <small style="color: white;">Copyright &copy; MVC</small>
  </div>
</footer>

</html>
```

22- Crear dentro de **main** la página principal, llamada **main.php**.

```
<!DOCTYPE html>
```

```
<?php require "views/templates/header.php"; ?>
<?php require "views/templates/nav.php"; ?>

<body>
  <br>
  <br>
  <br>
  <div class="container">
    <h1 class="text-center">Gestion personas</h1>

    <div style="padding: 0;" id="conten">
      <form role="form" action="<?php echo constant("URL") ?>Main/modificarPersona"
method="POST">
        <div class="col-md-12" id="conten">
          <input type="hidden" name="id" id="idpersona">

          <div class="form-group">
            <label for="nombre">Ingrese el nombre de la persona:</label>
            <div class="input-group">
              <input type="text" class="form-control" name="nombre" value="<?php echo $this-
>persona->getNombre(); ?>" id="nombre" placeholder="Ingresa el nombre" required>

            </div>
          </div>
          <div class="form-group">
            <label for="edad">Ingrese la edad de la persona:</label>
            <div class="input-group">
              <input type="number" class="form-control" id="edad" name="edad" value="<?php echo
$this->persona->getEdad(); ?>" placeholder="Ingresa la edad" required>
            </div>
          </div>
          <div class="form-group">
            <label for="edad">Ingrese el telefono de la persona:</label>
            <div class="input-group">
              <input type="tel" class="form-control" id="telefono" name="telefono" value="<?php
echo $this->persona->getTelefono(); ?>" placeholder="Ingresa el telefono" required>
            </div>
          </div>
          <div class="form-group">
            <label for="sexo">Ingrese el sexo de la persona:</label>
            <div class="input-group">
              <select name="sexo" id="sexo" class="form-control" required>
                <option value="Masculino">Maculino</option>
                <option value="Femenino">Femenino</option>
              </select>
            </div>
          </div>
          <div class="form-group">
            <label for="ocupacion">Ingrese la ocupacion de la persona:</label>
            <div class="input-group">
```



```
<select name="ocupacion" id="ocupacion" class="form-control" required>
  <?php
  foreach($this->listaOcupaciones as $lista ){
    ?>
    <option value="<?php echo $lista->getIdOcupacion(); ?>"><?php echo $lista-
>getOcupacion(); ?></option>
    <?php
    }
    ?>
  </select>
</div>
</div>
<div class="form-group">
  <label for="fecha">Ingresa la fecha de nacimiento de la persona:</label>
  <div class="input-group">
    <input type="date" class="form-control" id="fecha" value="<?php echo $this->persona-
>getFecha(); ?>" name="fecha" placeholder="Ingresa la fecha" required>
  </div>
</div>

<div style="margin-left: 30%;">
  <input type="submit" class="btn btn-success col-md-6 " >
</div>
</div>
</form>

</div>
<br>
<div>

  <div class="">
    <table class="table table-striped table-hover table-dark">
      <thead class="table-dark table-striped">
        <tr>
          <th>Id</th>
          <th>Nombre</th>
          <th>Edad</th>
          <th>Telefono</th>
          <th>Sexo</th>
          <th>Ocupacion</th>
          <th>Fecha nacimiento</th>
          <th colspan="2">Operaciones</th>
        </tr>
      </thead>
      <tbody>
        <?php

        foreach($this->listaPersonas as $lista ){
          ?>
          <tr>
```

```
<th><?php echo $lista->getIdPersona(); ?></th>
<th><?php echo $lista->getNombre(); ?></th>
<th><?php echo $lista->getEdad(); ?></th>
<th><?php echo $lista->getTelefono(); ?></th>
<th><?php echo $lista->getSexo(); ?></th>
<th><?php echo $lista->getOcupacion()->getOcupacion(); ?></th>
<th><?php echo $lista->getFecha(); ?></th>
<th><button onclick="alerta('<?php echo $lista->getIdPersona() ?>')" class="btn btn-
danger">Eliminar</button></th>
<th><button onclick="modificar('<?php echo $lista->getIdPersona() ?>',
'<?php echo $lista->getNombre(); ?>',
'<?php echo $lista->getEdad(); ?>',
'<?php echo $lista->getTelefono(); ?>',
'<?php echo $lista->getSexo(); ?>',
'<?php echo $lista->getOcupacion()->getIdOcupacion(); ?>',
'<?php echo $lista->getFecha(); ?>')" class="btn btn-info">Modificar</button></th>

</tr>
<?php
}

?>
</tbody>
</table>

</div>

</div>

</div>

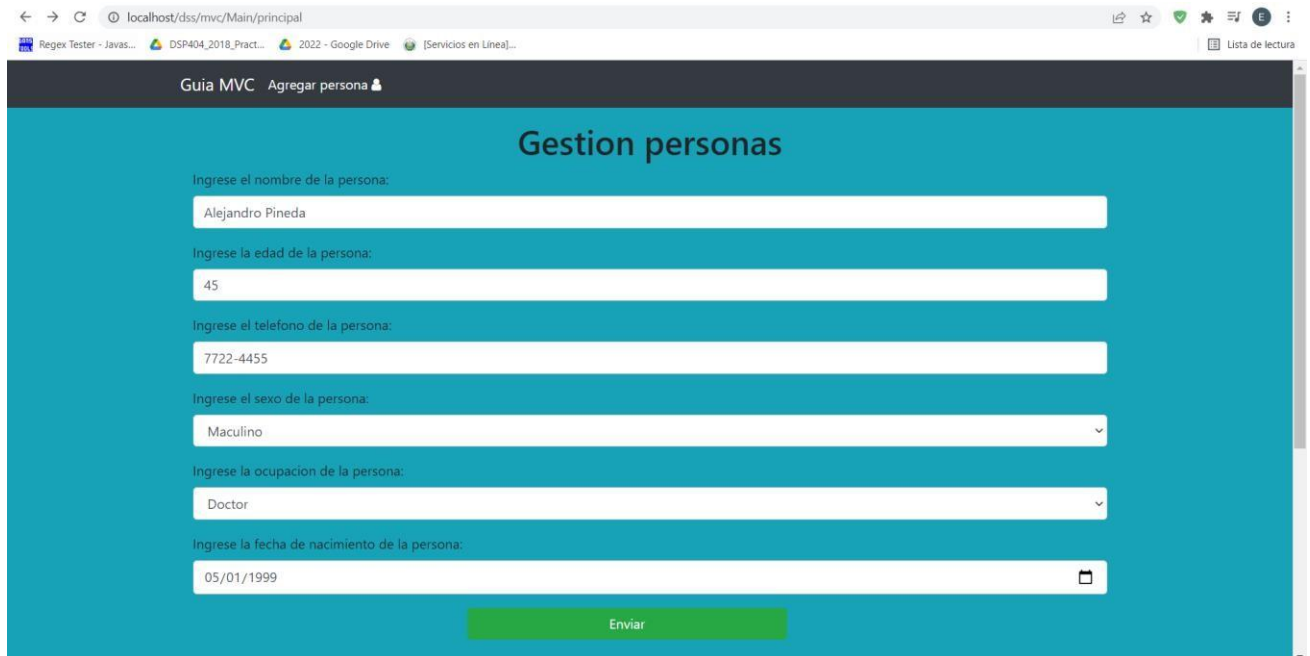
</body>

<?php require "views/templates/modal.php"; ?>
<?php require "views/templates/footer.php"; ?>

</html>
```

- 23- Finalmente correremos nuestra aplicación. Esto lo haremos digitando <http://localhost/dss/mvc/> o también lo podemos hacer invocando a la función principal <http://localhost/dss/mvc/Main/principal>. Podrá ver el siguiente resultado en la web.

## Guía #8: ¡Error! No se encuentra el origen de la referencia.



localhost/dss/mvc/Main/principal

Regex Tester - Javas... DSP404\_2018\_Pract... 2022 - Google Drive [Servicios en Línea]... Lista de lectura

Guía MVC Agregar persona

### Gestion personas

Ingrese el nombre de la persona:

Ingrese la edad de la persona:

Ingrese el telefono de la persona:

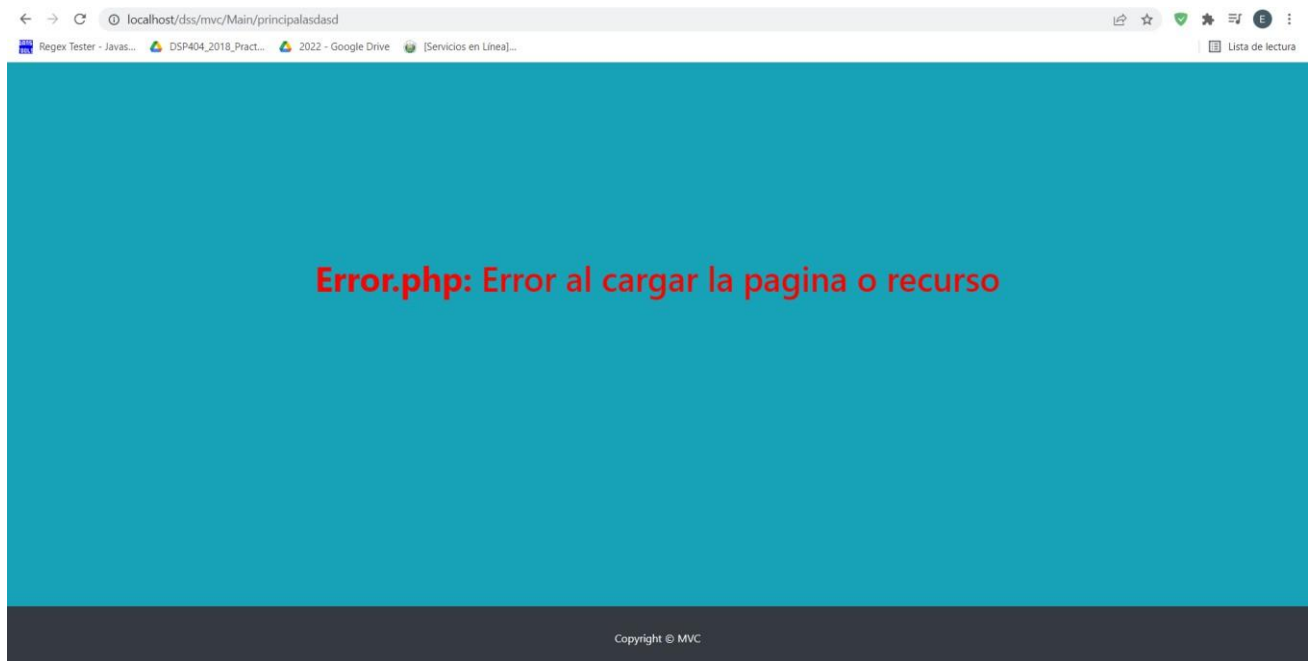
Ingrese el sexo de la persona:

Ingrese la ocupacion de la persona:

Ingrese la fecha de nacimiento de la persona:

Enviar

24- Si busca una función inexistente en el controller, será redirigido a la vista de error.



## V. DISCUSION DE RESULTADOS

1. Notara que faltan realizar las operaciones Eliminar y modificar de la guía, todo ya está predefinido en el cliente. Se le solicita entonces crear la función modificar en el controller y el model. Finalmente concluir la operación eliminar en el model, para lo cual ya está creada la función en el controller, bastara con que envíe los parámetros al model y haga la redirección.
2. Se le solicita enviar a la vista un mensaje de **operación realizada con éxito** con las operaciones insert, update y delete; pintar dicho mensaje en la vista de forma creativa.

**Guía #8:** ¡Error! No se encuentra el origen de la referencia.

3. Deberá hacer el crud de una de las tablas de su proyecto de catedra, implementando la lógica mvc de la presente guía. Para lo cual deberá crear otro proyecto y modificarlo con el fin de adaptarlo a la lógica de su base de datos.

**Resultado personal:**

Guia MVC

Agregar persona

Persona eliminada correctamente.

## Gestion personas

Ingrese el nombre de la persona:

Ingrese la edad de la persona:

Ingrese el telefono de la persona:

Ingrese el sexo de la persona:

Ingrese la ocupacion de la persona:



Ingrese la fecha de nacimiento de la persona:

Enviar

Id	Nombre	Edad	Telefono	Sexo	Ocupacion	Fecha nacimiento	Operaciones
1	Alejandro Pinedaaaae	45	7722-4455	Masculino	Doctor	1999-01-05	<div>Eliminar</div> <div>Modificar</div>
2	Fernando Calderón	21	7667-7890	Masculino	Emprendedor	2001-05-07	<div>Eliminar</div> <div>Modificar</div>
3	Emerson Torres	22	7123-9800	Masculino	Profesor	1999-08-03	<div>Eliminar</div> <div>Modificar</div>

Guía MVC

Agregar persona

 Persona modificada con éxito. 

## Gestion personas

Ingrese el nombre de la persona:

Alejandro Pinedaaaae

Ingrese la edad de la persona:

45

Ingrese el telefono de la persona:

7722-4455

Ingrese el sexo de la persona:

Maculino

Ingrese la ocupacion de la persona:

Doctor

Ingrese la fecha de nacimiento de la persona:

05/01/1999

Enviar

Id	Nombre	Edad	Telefono	Sexo	Ocupacion	Fecha nacimiento	Operaciones	
1	Alejandro Pinedaaaae	45	7722-4455	Masculino	Doctor	1999-01-05	Eliminar	Modificar
2	Fernando Calderóncito	21	7667-7890	Masculino	Emprendedor	2001-05-07	Eliminar	Modificar
3	Emerson Torres	22	7123-9800	Masculino	Profesor	1999-08-03	Eliminar	Modificar

Desarrollo de Aplicaciones Web con Software Interpretado en el Servidor

29

localhost dice  
Esta seguro de eliminar este registro

Aceptar Cancelar

Guía MVC Agregar persona

Ingrese el nombre de la persona:  
Alejandro Pinedaaaae

Ingrese la edad de la persona:  
45

Ingrese el telefono de la persona:  
7722-4455

Ingrese el sexo de la persona:  
Maculino

Ingrese la ocupacion de la persona:  
Doctor

Ingrese la fecha de nacimiento de la persona:  
05/01/1999

Enviar

Id	Nombre	Edad	Telefono	Sexo	Ocupacion	Fecha nacimiento	Operaciones
1	Alejandro Pinedaaaae	45	7722-4455	Masculino	Doctor	1999-01-05	Eliminar Modificar
2	Fernando Calderóncito	21	7667-7890	Masculino	Emprendedor	2001-05-07	Eliminar Modificar
3	Emerson Torres	22	7123-9800	Masculino	Profesor	1999-08-03	Eliminar Modificar

Copyright © MVC - Adrian Lopez - LM242664 - Tecnico en Ingenieria en Computación - 2025

## VI. BIBLIOGRAFIA

- Cabezas Granado, Luis Miguel. PHP 6 Manual Imprescindible. 1ra. Edición. Editorial Anaya Multimedia. Madrid, España. 2010.
- Matt Doyle. Fundamentos PHP Práctico. 1ra. Edición. Editorial Anaya Multimedia. Madrid, España 2010.
- F. Javier Gil Rubio / Santiago Alonso Villaverde. Creación de sitios web con PHP5. 1ra Edición. Editorial McGraw-Hill. España, 2006.
- Welling, Luke / Thomson, Laura. Desarrollo web con PHP y MySQL. Traducción de la 3ra Edición en inglés. Editorial Anaya Multimedia. 2005. Madrid, España.
- Gutierrez, Abraham / Bravo, Ginés. PHP 5 a través de ejemplos. 1ra Edición. Editorial Alfaomega. Junio 2005. México.
- John Coggeshall. LA BIBLIA DE PHP 5. 1ra. Edición. Editorial Anaya Multimedia. Madrid, España 2005.
- Sitio web: <http://es.php.net/manual/es/features.sessions.php>. Sitio web oficial de PHP.