CICLO: D.A.W.
MÓDULO DE DESARROLLO WEB
ENTORNO CLIENTE

Tarea Nº 05

Alumno:
Armando Herrero Silva
76442135W

Contenido

1. RA05_b.

Se han analizado tecnologías y mecanismos que permiten realizar esta separación y sus características principales.

- ***Enumerar tres frameworks que utilicen el patrón MVC.***

Hay muchos frameworks que usan el patrón MVC. Tres de ellos son:

- Symfony
- Angular
- Django

2. RA05_f.

Se han escrito aplicaciones Web con mantenimiento de estado y separación de la lógica de negocio.

- ***Implementar un MVC, correspondiente a un catálogo de artículos de una tienda online, con los siguientes requerimientos:***
  - ***El controlador frontal se llamará index.php***
  - ***Añadir un controlador llamado "controladores.php".***
  - ***modelo.php contiene el catálogo de artículos, almacenado en un array llamado $artículos. Los datos de los artículos se dejan a criterio del alumno. Se valorará la personalización de la aplicación.***
  - ***Por defecto, se muestra el listado de artículos.***
  - ***Para la url index.php/articulo?id=n. Se muestra el detalle del artículo n.***
- ***Nota: Es posible desarrollar el MVC en Symfony, en este caso las rutas serán las siguientes (para un proyecto tienda semejante al anterior):***
  - ***Listado de artículos: /tienda/public***
  - ***Detalle de un articulo n: /tienda/public/articulo/n***
- ***Añadir vistas para todos los controladores, se recomienda utilizar plantillas con Twig o con PHP Template Inheritance. Se valorará la personalización de la aplicación.***

Primero creamos la estructura MVC con sus correspondientes archivos en cada una de las carpetas:

Ahora creamos el modelo con sus correspondientes Contructor, Getter, Setter(add) y sus funciones:

```php
class Articulos_Modelo{

private $articulos = [];

    //Constructor

    public function __construct() {
        $this->articulos = [
            ['id' => 1, 'nombre' => 'Detector de humo', 'precio' => 10.99, 'stock' => 10, 'categoria' => 'Electronica'],
            ['id' => 2, 'nombre' => 'Lampara de escritorio', 'precio' => 5.99, 'stock' => 0, 'categoria' => 'Electronica'],
            ['id' => 3, 'nombre' => 'Caja de herramientas', 'precio' => 29.99, 'stock' => 15, 'categoria' => 'Herramientas'],
        ];
    }

    //Getter

    public function getArticulos() {
        return $this->articulos;
    }

    //Setter(Add)

    public function addArticulo($articulo, $precio, $stock, $categoria) {
        $id = count($this->articulos) + 1;
        $this->articulos[] = ['id' => $id, 'nombre' => $articulo, 'precio' => $precio, 'stock' => $stock, 'categoria' => $categoria];
    }

    //Funciones

    public function ArticuloPorId($id){
        foreach ($this->articulos as $articulo) {
            if ($articulo['id'] == $id) {
                return $articulo;
            }
        }
        return null;
```

Creamos también el controlador y la vista:

```php
<?php

    require_once 'models/modelo.php';

    class articuloController{

        private $modelo;

        //Contructor

        public function __construct(){
            $this->modelo = new Articulos_Modelo();
        }

        //Getter

        public function listArticulos() {
            return $this->modelo->getArticulos();
            require 'view/articulos.php';
        }

        //Setter

        public function addArticulo($articulo, $precio, $stock, $categoria) {
            $this->modelo->addArticulo($articulo, $precio, $stock, $categoria);
            $this->listArticulos();
        }

        //Funciones

        public function ArticuloPorId($id){
            return $this->modelo->ArticuloPorId($id);
            $this->listArticulos();
        }
    }
?>
```

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Artículos</title>
    <style>
        p{
            font-size: 20px;
        }
        #no_disponible{
            color: red;
            font-size: 20px;
        }
    </style>
</head>
<body>
    <h1>Artículos</h1>
    <ul>
        <?php foreach ($articulos as $articulo): ?>
            <li>
                <?php echo("<p>Articulo: ". $articulo['nombre'] . " | Precio: " . $articulo['precio'] . "€ | Categoría: "
                . $articulo['categoria'] . "</p>"); ?>
                <?php if($articulo['stock'] > 0){
                    echo("<p>Disponible</p>");
                }else{
                    echo("<p id='no_disponible'>No disponible</p>");
                } ?>
            </li>
            <?php endforeach; ?>
    </ul>
</body>
</html>
```

Por último, creamos el index, el cual actualmente es muy simple:

```php
<?php

    require_once 'controllers/controladores.php';

    $controller = new articuloController();

    $articulos = $controller->listArticulos();

    require 'view/articulos.php';
?>
```

Resultado de la web:

Artículos

- Articulo: Detector de humo | Precio: 10.99€ | Categoría: Electronica

  Disponible
- Articulo: Lampara de escritorio | Precio: 5.99€ | Categoría: Electronica

  No disponible
- Articulo: Caja de herramientas | Precio: 29.99€ | Categoría: Herramientas

  Disponible

3. RA05_d.

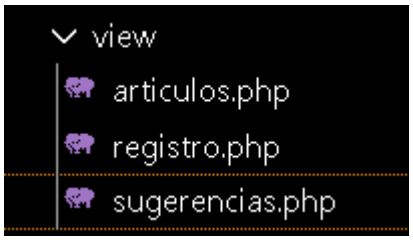
Se han utilizado formularios generados de forma dinámica para responder a los eventos de la aplicación Web.

- ***Añadir dos nuevos controladores, que crean formularios de forma automática, con sus vistas correspondientes:***
    - ***index.php/sugerencias: Carga una lista de las sugerencias dados por los usuarios. Permite crear nuevas sugerencias en memoria. No se guardan en ningún sitio.***
    - ***index.php/registro: Gestiona el registro de un usuario. Solamente se hace en memoria. No se guardan en ningún sitio.***

Controlador:

```
    // Métodos de Sugerencias
    public function listSugerencias() {
        $sugerencias = $this->sugerencias;
        require 'view/sugerencias.php';
    }

    public function addSugerencia($texto) {
        $this->sugerencias[] = $texto;
        $this->listSugerencias();
    }

    // Métodos de Registro
    public function showFormRegistro() {
        require 'view/registro.php';
    }

    public function registrarUsuario($nombre, $email) {
        $this->usuarios[] = ['nombre' => $nombre, 'email' => $email];
        echo "<p>Usuario registrado: $nombre ($email)</p>";
        echo '<a href="index.php?action=registro">Volver al registro</a>';
    }
}
```

Carpeta vista:



Tarea_5 > view > registro.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registro</title>
</head>
<body>
    <h1>Registro de Usuario</h1>
    <form method="POST" action="index.php?action=registrar_usuario">
        <label>Nombre:</label>
        <input type="text" name="nombre" required><br>
        <label>Email:</label>
        <input type="email" name="email" required><br>
        <button type="submit">Registrar</button>
    </form>
</body>
</html>
```

Tarea_5 > view > sugerencias.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sugerencias</title>
</head>
<body>
    <h1>Lista de Sugerencias</h1>
    <ul>
        <?php foreach ($sugerencias as $sugerencia): ?>
            <li><?php echo htmlspecialchars($sugerencia); ?></li>
        <?php endforeach; ?>
    </ul>
    <a href="index.php?action=nueva_sugerencia">Añadir una nueva sugerencia</a>
    <h1>Añadir Nueva Sugerencia</h1>
    <form method="POST" action="index.php?action=guardar_sugerencia">
        <textarea name="sugerencia" rows="4" cols="50" required></textarea>
        <br>
        <button type="submit">Guardar</button>
    /form>
</body>
</html>
```

Nuevo index:

```php
<?php
    $action = $_GET['action'] ?? 'view';

    switch ($action) {
        case 'sugerencias':
            $controller->listSugerencias();
            break;
        case 'nueva_sugerencia':
            require 'view/form_sugerencias.php';
            break;
        case 'guardar_sugerencia':
            if ($_SERVER['REQUEST_METHOD'] === 'POST') {
                $controller->addSugerencia($_POST['sugerencia']);
            }
            break;
        case 'registro':
            $controller->showFormRegistro();
            break;
        case 'registrar_usuario':
            if ($_SERVER['REQUEST_METHOD'] === 'POST') {
                $controller->registrarUsuario($_POST['nombre'], $_POST['email']);
            }
            break;
        case 'view':
        default:
            $controller->listArticulos();
            break;
    }
?>
```

4. RA05_e.

Se han identificado y aplicado los parámetros relativos a la configuración de la aplicación Web.

Crear una regla en el fichero modelo.php y en las vistas para evitar que puedan ser llamados directamente

- ***Comprobar, en las primeras líneas de los ficheros, que está definida la constante CON_CONTROLADOR. Si no está definida significará que se ha llamado al fichero sin pasar por el controlador. En ese caso, mostrar un mensaje de error indicando que no se puede llamar a ese fichero directamente y terminar la ejecución.***
- ***En el caso de necesitar terminar la ejecución, se puede ejecutar un comando que "mate" el proceso php (función die)***
- ***En el caso de utilizar Symfony, indicar si utilizando este framework es posible llamar directamente a una vista, y si es necesario utilizar una constante como en el caso anterior.***

models > modelo.php > Articulos_Modelo

```php

<?php

    if (!defined('CON_CONTROLADOR')) {
        die('Error: No se puede acceder directamente a este archivo.');
    }

    class Articulos_Modelo{

```

view > articulos.php > ...

```php
<?php
    if (!defined('CON_CONTROLADOR')) {
        die('Error: No se puede acceder directamente a este archivo.');
    }
?>

```

view > sugerencias.php > ...

```php
<?php
    if (!defined('CON_CONTROLADOR')) {
        die('Error: No se puede acceder directamente a este archivo.');
    }
?>

```

index.php > ...

```php

<?php

    define('CON_CONTROLADOR', true);

    require_once 'controllers/controladores.php';

    $action = $_GET['action'] ?? 'view';

```

5. RA_h.

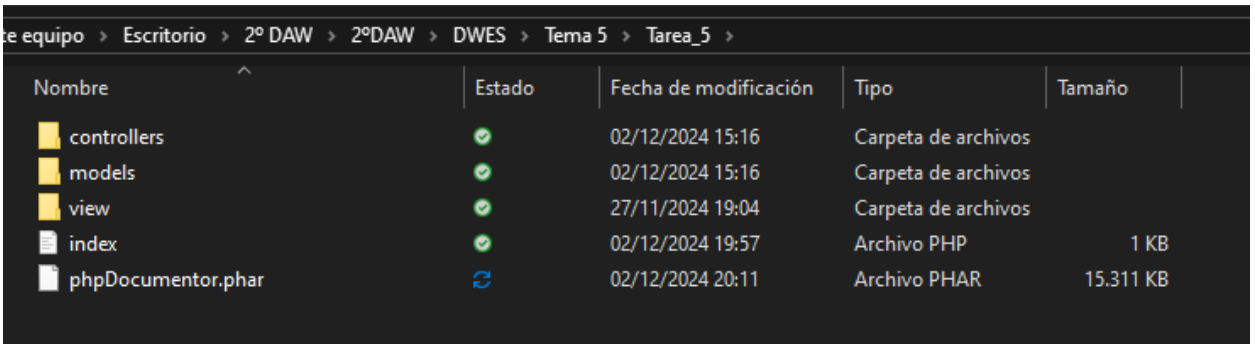
- ***Utilizando la sintaxis básica de PHPDoc, comentar las funciones creadas. Crear un comentario /** */ en cada función que incluya un comentario sobre la función***
- ***Utilizar @param para describir los parámetros que recibe la función y @return para comentar el valor resuelto***
- ***Se puede verificar el correcto funcionamiento de los comentarios creados usando PHPDocumentor.***
    - ***Descargar el fichero phpdocumentor.phar de la página de PHPDocumentor***
    - ***Crear un directorio doc fuera del directorio donde se esté creando la página web***
    - ***Ejecutar c:\xampp\php\bin\php.exe phpDocumentor.phar run -d c:\directorio de trabajo -t c:\directorio de documentacion creado. Estos directorios son de ejemplo***
    - ***Tras esto se debería haber creado una página web con el contenido de la documentación.***

```php
    /**
    * Añade un nuevo artículo a la lista.
    *
    * @param string $articulo Nombre del artículo.
    * @param float $precio Precio del artículo.
    * @param int $stock Cantidad disponible en stock.
    * @param string $categoria Categoría del artículo.
    * @return void
    */

    public function addArticulo($articulo, $precio, $stock, $categoria) {
```

```php
/**
* Busca un artículo por su ID.
*
* @param int $id ID del artículo a buscar.
* @return array|null Devuelve el artículo si se encuentra, o null si no existe.
*/

public function ArticuloPorId($id){
    foreach ($this->articulos as $articulo) {
```

```php
public function __construct(){
    $this->modelo = new Articulos_Modelo();
}

//Getter

/**
* Obtiene la lista de todos los artículos.
*
* @return array Lista de artículos.
*/

public function listArticulos() {
    return $this->modelo->getArticulos();
    require 'view/articulos.php';
}

//Setter

/**
* Añade un nuevo artículo al modelo.
* @param string $articulo Nombre del artículo.
* @param float $precio Precio del artículo.
* @param int $stock Cantidad disponible en stock.
* @param string $categoria Categoría del artículo.
* @return void
*/
```

```php
//Funciones

/**
* Busca un artículo por su ID.
* @param int $id ID del artículo a buscar.
* @return array|null Devuelve el artículo si se encuentra, o null si no existe.
*/

public function ArticuloPorId($id){
    return $this->modelo->ArticuloPorId($id);
    $this->listArticulos();
}
```



Vamos a cambiar el proyecto de ubicación porque nos da el siguiente error:


```
Usuario@ALUM-0016 c:\xampp
# c:\xampp\php\php.exe C:\xampp\htdocs\Tarea_5\phpDocumentor.phar run -d "C:\xampp\htdocs\Tarea_5" -t "C:\xampp\htdocs\Tarea_5\doc"
phpDocumentor v3.5.3

Parsing files
 6/6 [============================] 100%
Applying transformations (can take a while)

All done in 0 seconds!

Usuario@ALUM-0016 c:\xampp
#
```

Todo correcto.