



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

# **Tarea N° 05**

**Alumno:**  
**Armando Herrero Silva**  
**76442135W**

*Los documentos, elementos gráficos, vídeos, transparencias y otros recursos didácticos incluidos en este contenido pueden contener imprecisiones técnicas o errores tipográficos. Periódicamente se realizan cambios en el contenido. Fomento Ocupacional FOC SL puede realizar en cualquier momento, sin previo aviso, mejoras y/o cambios en el contenido.*

*Es responsabilidad del usuario el cumplimiento de todas las leyes de derechos de autor aplicables. Ningún elemento de este contenido (documentos, elementos gráficos, vídeos, transparencias y otros recursos didácticos asociados), ni parte de este contenido puede ser reproducida, almacenada o introducida en un sistema de recuperación, ni transmitida de ninguna forma ni por ningún medio (ya sea electrónico, mecánico, por fotocopia, grabación o de otra manera), ni con ningún propósito, sin la previa autorización por escrito de Fomento Ocupacional FOC SL.*

*Este contenido está protegido por la ley de propiedad intelectual e industrial. Pertenecen a Fomento Ocupacional FOC SL los derechos de autor y los demás derechos de propiedad intelectual e industrial sobre este contenido.*

*Sin perjuicio de los casos en que la ley aplicable prohíbe la exclusión de la responsabilidad por daños, Fomento Ocupacional FOC SL no se responsabiliza en ningún caso de daños indirectos, sean cuales fueren su naturaleza u origen, que se deriven o de otro modo estén relacionados con el uso de este contenido.*

*© 2022 Fomento Ocupacional FOC SL todos los derechos reservados.*

## Contenido

1.	RA05_b.	2
2.	RA05_f.	2
3.	RA05_d.	5
4.	RA05_e.	9
5.	RA_h.	10

## 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 artículo 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 Constructor, Getter, Setter(add) y sus funciones:

```
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
require_once 'models/modelo.php';

class articuloController{

    private $modelo;

    //Constructor

    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();
    }
}
```

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Articulos</title>
  <style>
    p{
      font-size: 20px;
    }
    #no_disponible{
      color: red;
      font-size: 20px;
    }
  </style>
</head>
<body>
  <h1>Articulos</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:

```

1  <?php
2
3      require_once 'controllers/controladores.php';
4
5      $controller = new articuloController();
6
7      $articulos = $controller->listArticulos();
8
9      require 'view/articulos.php';
10 ?>
11

```

Resultado de la web:

## Artículos

- Artículo: Detector de humo | Precio: 10.99€ | Categoría: Electronica  
Disponible
- Artículo: Lampara de escritorio | Precio: 5.99€ | Categoría: Electronica  
**No disponible**
- Artículo: 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
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Registro</title>
7  </head>
8  <body>
9      <h1>Registro de Usuario</h1>
10     <form method="POST" action="index.php?action=registrar_usuario">
11         <label>Nombre:</label>
12         <input type="text" name="nombre" required><br>
13         <label>Email:</label>
14         <input type="email" name="email" required><br>
15         <button type="submit">Registrar</button>
16     </form>
17 </body>
18 </html>
```



Tarea\_5 &gt; view &gt; sugerencias.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Sugerencias</title>
7 </head>
8 <body>
9   <h1>Lista de Sugerencias</h1>
10  <ul>
11    <?php foreach ($sugerencias as $sugerencia): ?>
12      <li><?php echo htmlspecialchars($sugerencia); ?></li>
13    <?php endforeach; ?>
14  </ul>
15  <a href="index.php?action=nueva_sugerencia">Añadir una nueva sugerencia</a>
16  <h1>Añadir Nueva Sugerencia</h1>
17  <form method="POST" action="index.php?action=guardar_sugerencia">
18    <textarea name="sugerencia" rows="4" cols="50" required></textarea>
19    <br>
20    <button type="submit">Guardar</button>
21  </form>
22 </body>
23 </html>
```

Nuevo index:

```
<?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
1
2 <?php
3
4     if (!defined('CON_CONTROLADOR')) {
5         |     die('Error: No se puede acceder directamente a este archivo.');

```

```
view > articulos.php > ...
1 <?php
2     if (!defined('CON_CONTROLADOR')) {
3         |     die('Error: No se puede acceder directamente a este archivo.');

```

```
view > sugerencias.php > ...
1 <?php
2     if (!defined('CON_CONTROLADOR')) {
3         |     die('Error: No se puede acceder directamente a este archivo.');

```

```

index.php > ...
1
2 <?php
3
4     define('CON_CONTROLADOR', true);
5
6     require_once 'controllers/controladores.php';
7
8     $action = $_GET['action'] ?? 'view';
9

```

## 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.*

```

/**
 * 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) {

```

```

/**
 * 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 ArtículoPorId($id){
    foreach ($this->articulos as $articulo) {

```

```

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
 */

```

```
//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 ArtículoPorId($id){
    return $this->modelo->ArtículoPorId($id);
    $this->listArticulos();
}

```

te equipo > Escritorio > 2º DAW > 2ºDAW > DWES > Tema 5 > Tarea_5 >					
Nombre	Estado	Fecha de modificación	Tipo	Tamaño	
controllers	✓	02/12/2024 15:16	Carpeta de archivos		
models	✓	02/12/2024 15:16	Carpeta de archivos		
view	✓	27/11/2024 19:04	Carpeta de archivos		
index	✓	02/12/2024 19:57	Archivo PHP	1 KB	
phpDocumentor.phar	🔄	02/12/2024 20:11	Archivo PHAR	15.311 KB	

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

```
[warning] Your documentationset seems to be empty!
Parsing files
In Local.php line 112:

Impossible to create the root directory "phar:///C:/Users/Usuario/OneDrive%20-%20Instituto%20FOC/Desktop/2%20BA%20
AW/2%20BADAW/DWES/Tema%205/Tarea_5/phpDocumentor.phar/src/phpDocumentor/../../data/templates", mkdir(): phar error
: cannot create directory "phar:///C:/Users/Usuario/OneDrive%20-%20Instituto%20FOC/Desktop/2%20BA%20DAW/2%20BADAW/
DWES/Tema%205/Tarea_5/phpDocumentor.phar/src/phpDocumentor/../../data/templates", write operations disabled

project:run [-t|--target [TARGET]] [--cache-folder [CACHE-FOLDER]] [-f|--filename [FILENAME]] [-d|--directory [DIRECTORY]] [--encoding [ENCODING]] [--extensions [EXTENSIONS]] [--ignore [IGNORE]]
[--ignore-tags [IGNORE-TAGS]] [--hidden] [--ignore-symlinks] [--no-ignore-symlinks] [-m|--markers [MARKERS]] [--title [TITLE]] [--force] [--validate] [--visibility [VISIBILITY]] [--defaultpackagename
[DEFAULTPACKAGENAME]] [--sourcecode|--no-sourcecode] [--template [TEMPLATE]] [--examples-dir [EXAMPLES-DIR]] [--s|--setting [SETTING]] [--list-settings] [--parseprivate]

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

```
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.