

Desarrollo Web en Entorno Servidor

AEV2 -

Modelo Vista Controlador con Doctrine

Ciclo Formativo de Desarrollo de Aplicaciones Web

Realizado por: **Manuel Lillo Boix**

ÍNDICE

[1. Introducción a la actividad. 2](#_Toc116845202)

[1.1. Enunciado. 2](#_Toc116845203)

[1.2. Recursos Entregados. 2](#_Toc116845204)

[1.3. Objetivos. Rúbrica. 2](#_Toc116845205)

[2. Solución de la actividad. 3](#_Toc116845206)

[2.1. Descripción de la actividad y porques. 3](#_Toc116845207)

[2.1.1. Porque has implementado esta solución. 3](#_Toc116845208)

[2.1.2. Y por qué consideras que es la solución adecuada. 3](#_Toc116845209)

[2.1.3. Y por qué otra solución no es más adecuada. 3](#_Toc116845210)

[2.2. Repositorio Git de la actividad. 3](#_Toc116845211)

[2.3. Enlace del Vídeo Explicativo. 3](#_Toc116845212)

[2.4. Árbol y estructura del proyecto. 4](#_Toc116845213)

[2.5. Auto-Rúbrica. 5](#_Toc116845214)

# Introducción a la actividad.

# Enunciado.

Esta actividad evaluable consiste en crear una aplicación que cumpla los criterios vistos en los temas 1, 2, 3 y 4. Debemos crear un proyecto que cumpla con el patrón MVC, que tenga un sistema de enrutamiento mediante el autoload de composer y los namespaces. Usaremos TWIG para implementar las plantillas visuales.

# Recursos Entregados.

Presentaciones y videos de los temas 1, 2, 3 y 4: • Introducción a MVC  
• Introducción a TWIG  
• Introducción a Doctrine ORM

• Doctrine Asociaciones • CRUD con Doctrine

Material de apoyo:

• Todo el entregado en los temas.

# Objetivos. Rúbrica.

Existen cuatro niveles de corrección para esta actividad:

* **Mínimos (hasta 5 puntos)**o La aplicación debe conectar con la BB.DD.  
  o Deberán estar implementados los **puntos del 1 al 8** de la sección de actividades

más los puntos:  
o **Punto 9.1** 🡪 Que la ruta raíz, “/”, se muestre al iniciar el proyecto desde la carpeta

public, ejecutando los requisitos.  
o **Punto 9.12** 🡪 Deberá estar implementado que si se introduce una ruta no definida

en la aplicación debe mostrar un mensaje como indica el punto 6.  
o **Punto 10 y subpuntos** ◊ Usar el formato predefinido para TWIG.  
o **Punto 11** 🡪 Realizar memoria según documentación subida en Florida Oberta.

o **Punto 12** 🡪 Realizar video explicativo de la actividad.  
o **Punto 13** 🡪 Rellenar la Auto rúbrica

* **Suficiencia (hasta 6 puntos)**o Cumplir todos los requisitos del punto mínimos.  
  o Añadir los puntos **9.2** 🡪 **/products, 9.3** 🡪 **/wharehouses y 9.4** 🡪 **/bunisses.**
* **Notoriedad (hasta 8 puntos)**o Cumplir todos los requisitos del punto suficiencia.  
  o Añadir los puntos **9.5** 🡪 **/orders, 9.9** 🡪 **/update y 9.10** 🡪 **/insert.**
* **Excelencia (hasta 10 puntos)**o Cumplir todos los requisitos del punto notoriedad.  
  o Añadir los puntos **9.6** 🡪 **/ordersDetail, 9.7** 🡪 **/invoices, 9.8** 🡪 **/invoice y**

**9.11** 🡪 **/delete.**

# Solución de la actividad.

# Descripción de la actividad y porques.

En esta actividad hemos procedido a la realización de una aplicación web estructurada con el patrón Modelo Vista Controlador. A través del archivo routes.json hemos implementado las diferentes rutas ligadas con sus respectivas acciones. Además, hemos utilizado estas rutas junto con sus controladores para implementar la funcionalidad de la inserción, actualización y borrado de datos (CRUD). También hemos tenido que modelizar todas las entidades a Doctrine e implementar las diferentes asociaciones entre todas las tablas de la base de datos. Todo esto lo hemos renderizado con plantillas Twig heredadas a partir de una estructura html proporcionada.

# Porque has implementado esta solución.

Hemos implementado esta solución para poder afianzar el Modelo Vista Controlador que ya hemos ido trabajando a lo largo del curso. Hemos modelado las entidades para trabajar con orientación a objetos, ya que Doctrine permite manejar los datos de la BBDD como si fueran objetos. Además, hemos asociado estos objetos (tablas) para poder obtener informaciones de una tabla a otra. Después los controladores han utilizado la tecnología de Doctrine para hacer las peticiones y demás acciones a la BBDD y así poder facilitar el manejo de datos. Por último, hemos heredado plantillas de Twig y reutilizado también. Según el tipo de variable que llegara a la plantilla la muestra de datos era de una forma u otra.

# Y por qué consideras que es la solución adecuada.

Aplicar esta solución nos permite que la aplicación tenga un acoplamiento cada vez más bajo y que el modelo vaya dependiendo menos del resto de la aplicación. Esto permite escalar más fácilmente la aplicación y poder adaptarla de forma más rápida y eficiente a otros proyectos.

# Y por qué otra solución no es más adecuada.

Podemos decir que debido a los conocimientos que tenemos no haber utilizado el potencial de Doctrine, a pesar de le tamaño reducido de la aplicación, hubiera sido una solución menos eficiente y con mucho más acoplamiento que si no hubiéramos utilizado Doctrine y el Modelo de Vista Controlador.

# Repositorio Git de la actividad.

<https://github.com/manlibo/AEV2_Manuel_Lillo.git>

# Enlace del Vídeo Explicativo.

[Video-AEV2-Manuel-Lillo-Boix.mov](https://floridauniversitaria-my.sharepoint.com/:v:/g/personal/malibo_floridauniversitaria_es/ERcSEnJEX6xGsITIQzYlPbcBwe4xQw-hmeK96auwz-jGZQ?e=5yfGAi)

# Árbol y estructura del proyecto.

Pantalla de computadora con letras

Descripción generada automáticamente

# Auto-Rúbrica.

|  |  |  |  |
| --- | --- | --- | --- |
| Nivel de Rúbrica | Superado | No superado | Observaciones |
| Mínimos | Sí | -Falta Botón atrás  -€ con php |  |
| Suficiencia | Sí |  |  |
| Notoriedad | Si |  |  |
| Excelencia | /orderDetail  /invoices  /delete | -Variables de sesión  -Relacionar /invoice con pedidos |  |