

# Programación Web Avanzada

## PEC 2: Componentización

Esta PEC abordará la creación de los componentes fundamentales del caso de estudio, y la comprensión del funcionamiento de propiedades, parámetros, y división de elementos en componentes.

## Conocimientos, Habilidades y Competencias

En esta PEC se desarrollan los siguientes conocimientos (K), habilidades (S) y Competencias (C) del Grado en Multimedia 2023:

- **K1.** Identificar y diferenciar los elementos de la teoría de la multimedia para analizar, conceptualizar y diseñar productos interactivos multimedia.
- **S3.** Utilizar de manera adecuada los lenguajes de programación, las herramientas de desarrollo y las tecnologías disponibles para el análisis, el diseño y la implementación de aplicaciones multimedia.
- **S7.** Analizar y sintetizar información para evaluar soluciones tecnológicas y elaborar propuestas de proyectos multimedia teniendo en cuenta los recursos, las alternativas disponibles y las condiciones de mercado.


## Objetivos

Los objetivos específicos de esta PEC son:

- Familiarizarse con el funcionamiento de Vue.js.
- Conocer conceptos básicos de componentes en Vue.

## Recursos de lectura

Tal y como hicimos en la PEC anterior, a continuación os indicamos los recursos de lectura que consideramos importantes que reviséis y estudiéis para realizar esta PEC. También os indicamos las secciones (o capítulos concretos) que deberíais leer con atención para la correcta ejecución de esta PEC. Para cada una de las partes recomendadas incluimos un comentario para motivar nuestra recomendación. Por supuesto, si queréis profundizar, podéis revisar y leer las secciones no recomendadas en este listado de cada recurso.

 *El concepto de componente no se crea en el ámbito Web, sino que ha existido en el área de ingeniería del software desde hace tiempo. En uno de los libros de referencia sobre desarrollo de software orientado a objetos, titulado “Object-oriented Software Construction”, escrito por Bertrand Meyer y publicado en 1988; ya se introducía este concepto. Si quieres revisar el concepto de componente desde el punto de vista de ingeniería del software (y no de ingeniería web), te recomendamos leer el capítulo 8 del libro [Fundamentals of Software Architecture](#), escrito por Mark Richards y Neal Ford*


 **Libro: Azaustre Rodríguez, C. Conceptos y técnicas avanzadas de programación web. FUOC, 2024**

### 2. El concepto de componente

En este primer capítulo, exploraremos el proceso de registro de componentes y su ciclo de vida. Es importante comprender estos conceptos clave antes de pasar al capítulo 2.5, donde aprenderemos a crear componentes en los frameworks de JavaScript más populares: Vue, React, Angular o Svelte.

#### 8.1.1. Single-file components


En este breve subcapítulo, veremos un ejemplo de cómo crear un componente de archivo único en Vue.

 *Como habrás podido leer en los recursos, los componentes en Vue están relacionados con Componentes Web, un concepto más abstracto que pretende ofrecer un conjunto de funcionalidades para crear elementos reutilizables en la Web. Si quieres profundizar más en el concepto de Componentes Web, te recomendamos leer [el artículo](#) de Mozilla Developer Network.*

### Vue docs

**Essentials.** <https://vuejs.org/guide/essentials/application.html>

En esta serie de capítulos se hace un pequeño repaso a las características básicas de Vue, su estructura y los principios básicos para usar este framework. Presta atención especial a los capítulos Template Syntax, Reactivity Fundamentals y Component Basics, ya que es el lugar donde se desarrolla la mayor parte de los conceptos necesarios para realizar esta PEC.

 *Vue 3 cuenta con dos tipos de API: Options API y Composition API. El uso de cada API implica un estilo de programación diferente y el uso de métodos diferentes. En [este recurso](#) puedes estudiar sus diferencias. **En esta PEC haremos uso de la Options API y de su documentación.***

## Formato de envío

A través del aula, debes entregar un **archivo zip con el contenido de la carpeta src del proyecto y un archivo PDF con las respuestas a las preguntas formuladas**. El documento PDF debe tener al menos el nombre del estudiante en el encabezado.

## Puntuación

- La puntuación de cada pregunta se detalla en su enunciado, **con un total de 10 puntos**.
- Las preguntas otorgan un **máximo de 2 puntos**. destacando los puntos claves de cada respuesta es suficiente.
- El desarrollo de los componentes otorga una **puntuación máxima de 8 puntos**. La corrección tendrá en cuenta:
  - Creación correcta de componentes y estructura de archivos correcta (20%)
  - Creación correcta de parámetros/datos (20%)
  - Creación correcta de plantillas en base a la información proporcionada y los estilos CSS adjuntos (20%)
  - Creación de color personalizado para cada tarjeta vinculado a los datos (10%)
  - Creación de casos de ejemplo con datos de prueba. (20%)
  - Calidad general del código (10%)

## Contacto

En caso de que tengáis que consultar algo mediante el foro o por correo electrónico, copiad vuestro código a una plataforma online y enviad el link para evitar problemas con el correo de la UOC (ya que elimina los ficheros `.js` y `.vue` para evitar inyectar código malicioso así como la indentación del código). Os recomendamos utilizar cualquiera de estas dos opciones:

- [Codepen](#) (para sencillos snippets de código)
- [CodeSandbox](#) (para ejercicios más complejos)

**Nota:** En caso de publicar algún código en el foro, deberán ser consultas genéricas y no directamente soluciones a los ejercicios. Hacer accesible soluciones de ejercicios a otros compañeros, aunque pueda no tener una intención directa, se considerará copia y se penalizará académicamente a nivel de asignatura.

## Propiedad intelectual y plagio

La Normativa académica de la UOC dispone que el proceso de evaluación se cimenta en el trabajo personal del estudiante y presupone la autenticidad de la autoría y la originalidad de los ejercicios realizados.

En esta PEC no está permitido el uso de herramientas de Inteligencia Artificial. En el plan de estudios y en [la web sobre integridad académica y plagio de la UOC](#) encontrarás más información sobre las faltas y las consecuencias del plagio. Recuerda que si utilizas material externo siempre tienes que hacer referencia a las fuentes, también puedes leer <https://biblioteca.uoc.edu/es/pagina/Como-citar/>

El estudiante será calificado con un suspenso (D/0) si se detecta falta de originalidad en la autoría de alguna prueba de evaluación continua (PEC) o final (PEF), sea porque haya utilizado material o dispositivos no autorizados, sea porque ha copiado textualmente de internet, o ha copiado apuntes, de PEC, de materiales, manuales o artículos (sin la cita correspondiente) o de otro estudiante, o por cualquier otra conducta irregular.

## Preguntas

Responde las siguientes preguntas. No es necesario desarrollarlos en un proyecto real:

### Ejercicio 1. (0,5 puntos)

Crea un nuevo componente llamado **ProductCard** con las siguientes propiedades y datos:

Props requeridos:

- title: String
- price: Number
- inStock: Boolean

Data:

- quantity: number. Valor inicial: 0
- shipping: array. Valores:
  - {id: 1, cost: 3.95, days: 3, country: Germany, freeShippingItems: 10}
  - {id: 2, cost: 0.50, days: 15, country: China, freeShippingItems: 5}
- selectedShipping: Object. Valor inicial: objeto vacío.

Template:

```
<h2>{{ title }}</h2>
<p>Price: ${{ price }}</p>
```

### Ejercicio 2. (0,25 puntos)

Añade un método llamado `increaseQuantity` en el componente creado anteriormente, que **incremente en 1** el valor de “quantity” siempre que el array de shipping tenga algún valor. Crea un botón con el texto **“Add to cart”** que llame a esta función.

### Ejercicio 3. (0,25 puntos)

Modifica la plantilla del componente para que muestre un nuevo texto en caso de que el producto no esté en stock:

```
<small>Out of stock</small>
```

### Ejercicio 4. (0,5 puntos)

Modifica el botón de “Add to cart” para que incluya la propiedad “disabled” tanto si no hay valores en el array Shipping como si no hay stock del producto.

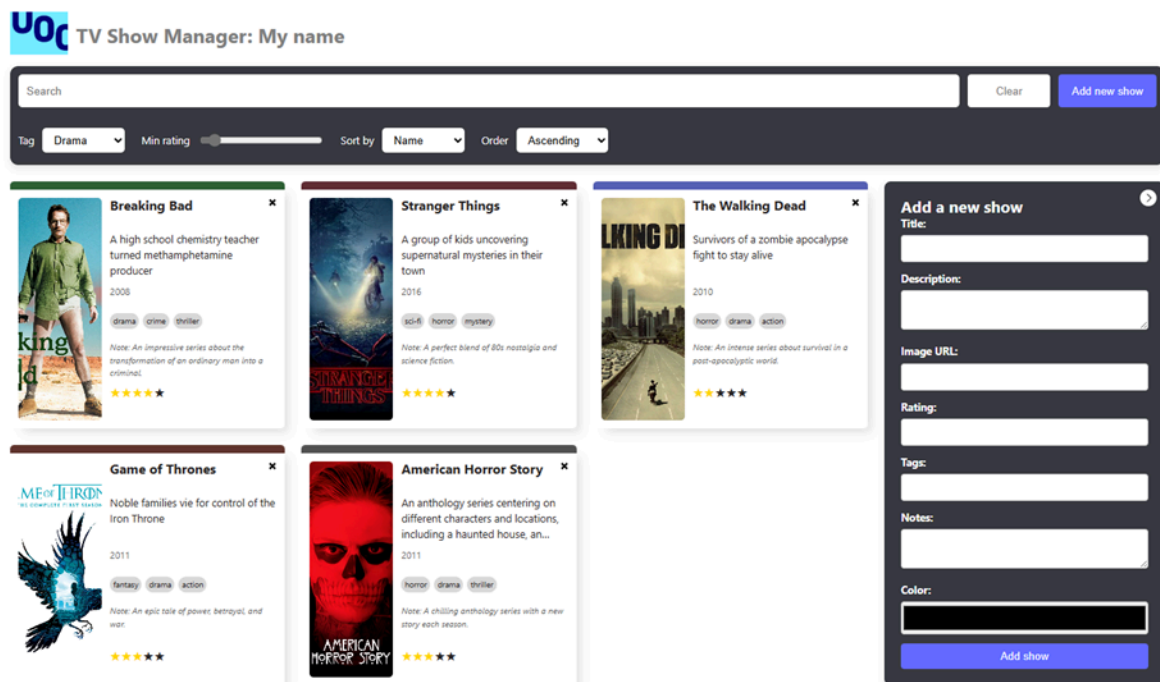
### Ejercicio 5. (0,5 puntos)

Muestra un mensaje bajo el botón de “Add to cart” tras hacerle click si faltan items para que el envío sea gratuito. Por ejemplo, si se ha seleccionado la opción del país “Germany”, su “freeShippingItems” es 10 y tenemos 3 productos al carrito, debe salir el siguiente mensaje:

```
<p>Buy 7 more items to have free shipping!</p>
```

## Presentación del caso de estudio

Durante esta PEC y las siguientes, vamos a crear una aplicación web en Vue.js para gestionar nuestro visionado de series de televisión, donde podremos añadir nuevas series, borrarlas, poner nuestra valoración, comentarios, etc. Para facilitar el desarrollo, esta PEC y las siguientes abordarán cada uno de los diferentes aspectos de Vue.js que trabajaremos durante el curso.



## Estilos

Antes de empezar, aunque podéis utilizar estilos propios, en los materiales encontraréis los estilos globales y para cada componente para que sea más fácil montar la aplicación. El resultado final debe quedar como en la captura de pantalla en formato escritorio (no hace falta que sea responsivo).

También disponéis, en la carpeta “assets”, los iconos e imágenes de la aplicación.

## Componentes

En esta PEC, abordaremos la estructura y desarrollo básico de los componentes que componen la aplicación, sin tener en cuenta el origen de los datos, ni la comunicación entre ellos.

**Nota: Para esta entrega, sólo será necesario probar el comportamiento del tablero con datos de prueba. Los métodos, *watchers* o cualquier tipo de lógica de negocio no serán evaluados en esta PEC y se realizarán en futuras PEC.**

### ShowCard.vue

En primer lugar, tenemos nuestro componente ShowCard: Este componente es la unidad mínima de información que contendrá los datos de una serie, en forma de tarjeta.

Este componente recibirá un objeto con los siguientes parámetros:

- id: ID (en formato int)
- title: Nombre
- description: Descripción
- image: URL de la imagen
- rating: Número float entre 0 y 5
- tags: Array de strings
- notes: Un string
- color: Un string en formato de color hexadecimal. Por ejemplo, "#2e6136"
- releaseDate: Un string con la fecha de estreno

**Nota: Estos parámetros deberán pasarse al componente como parte del objeto "Show", no como parámetros por separado.**

Con esta información, deberíais poder montar un componente con su template, script y style utilizando un código HTML semántico.



A tener en cuenta:

- El color del borde de la tarjeta es dinámico está relacionado con el parámetro “color” del objeto.
- Las estrellas se pintan utilizando el emoji ★. Siempre aparecerán 5 estrellas y se pintarán de color amarillo aquellas que, redondeando al número “int” más bajo, correspondan a la valoración actual. Por ejemplo, un 4.5 de rating pintará 4 estrellas de color amarillo y una estrella de color gris.

## CardBoard.vue

Este componente se encargará de cargar la lista de tarjetas de series, por lo que se deberá iterar a través de una serie de componentes de tipo ShowCard.vue. Este componente recibirá los siguientes parámetros:

- showList: Array de items.

## FilterBar.vue

Este componente se encargará en el futuro de realizar búsquedas y filtrar los resultados. También contiene un botón que permitirá mostrar el formulario para crear series. Cuenta con los siguientes inputs:

- Barra de búsqueda.
- Selector de etiquetas. Podéis incluir de manera aleatoria 4 o 5 etiquetas para identificar una serie (Comedia, drama, etc.).
- Rating: Un rango que mostrará los ítems que tengan, como mínimo, lo que se ha seleccionado. Va de 0 a 5.
- Sort by: Se encargará de ordenar los elementos dependiendo de tres características: Nombre, fecha de estreno y valoración.
- Order: Vinculado a lo anterior, ordenará los elementos de manera ascendente o descendente.

El botón “Add new show” se podrá ocultar en el futuro.

## CardForm.vue

Este componente, situado a la derecha de la aplicación, será el encargado de gestionar la creación de nuevos elementos en una futura PEC. Básicamente es un formulario que, en el futuro, emitirá nuevos elementos para agregar a la lista de items.

Contiene un botón arriba a la derecha que en el futuro permitirá ocultarse a sí mismo, al igual que el botón de “Add new show”. De momento, podéis incluir un condicional que permita ocultarse o mostrarse con una variable.

## App.vue

Este componente se encargará de la gestión global de la aplicación y a su vez compondrá la estructura general.

Para esta PEC, **se deben generar 5 series pasando valores de prueba como parámetros**, para simular la apariencia de la imagen de presentación del caso de estudio.

Este componente debe contener las siguientes características:

- Cuenta con una cabecera que contendrá el logo y el título “TV Show Manager” seguido de tu nombre y apellido.
- Importa los componentes FilterBar, CardBoard y CardForm, colocados como en la captura de pantalla.
- Pasa el listado de datos de prueba a CardBoard.
- Los datos de prueba deben ser coherentes para darle sentido a nuestra aplicación.