**CONNECT-MANAGER**



**Realizado por:** Mario Martín Godoy

**Curso:** 2º Desarrollo Aplicaciones Web

**Módulo:** Desarrollo de entorno servidor

**Tecnologías:** HTML, CSS, PHP

**ÍNDICE DE CONTENIDOS**

1. Explicación del proyecto...................................... página 2
2. Ventajas patrón MVC…....................................... página 3
3. Justificación tecnología utilizada......................... página 5
4. Demostración funcionamiento............................. página 6
5. Tecnologías utilizadas......................................... página 10
6. **EXPLICACIÓN DEL PROYECTO**

**ConnectManager: Aplicación web de consumición de API-Rest.**

Bienvenido a ConnectManager, una plataforma online para visualizar el contenido de tu API REST favorita. Obtén el resultado de tus peticiones con o sin parámetros. Copia la estructura JSON resultante de la petición.

* En ConnectManager, podrás realizar peticiones a tu API con o sin parámetros mediante un buscador con filtro modificable a tu gusto.
* Podrás comprobar los resultados obtenidos en tus peticiones con una interfaz agradable, además de copiar la estructura JSON del contenido devuelto.

**Características Principales:**

* **Peticiones a API:** Modifica la URL a la que solicitas y realiza peticiones.
* **Filtrar por parámetros:** Realiza peticiones parametrizadas usando nuestro filtro.
* **Visualiza el contenido:** Podrás ver toda la información obtenida de cada elemento en una interfaz intuitiva y agradable.
* **Observa la estructura:** Comprueba la estructura de datos obtenida en la petición en un visor JSON. Podrás expandir y reducir el contenido. Además, podrás copiar toda la estructura al portapapeles.
* **Interfaz Intuitiva:** Una interfaz de usuario amigable que facilita la navegación y el uso de las funcionalidades de la aplicación.

Este proyecto busca proporcionar una aplicación sencilla y fácil de usar para que un usuario estándar pueda realizar peticiones a una API y observar su contenido de una manera cómoda e intuitiva.

Siéntete libre de explorar, contribuir y adaptar este proyecto según tus necesidades específicas. ¡Disfruta de tu experiencia en ConnectManager!

**Innovación:**

¿Qué nos hace únicos?: ConnectManager va más allá de ser una simple web que consume datos de una API. Buscamos constantemente innovar con características únicas que mejoren la experiencia de lectura de datos, manteniendo a nuestros clientes contentos y garantizando calidad en su negocio.

Con ConnectManager, estamos construyendo una plataforma de servicio virtual donde los clientes sean capaces de realizar peticiones a la API y obtener el resultado en formato JSON, además de observar los datos obtenidos en un formato visualmente agradable. ¡Sumérgete en la esencia de ConnectManager!

1. **VENTAJAS PATRÓN MVC**
2. **Separación de preocupaciones**: MVC divide la aplicación en tres componentes principales: el Modelo (encargado de la lógica del negocio y el acceso a datos), la Vista (encargada de la presentación de la interfaz de usuario) y el Controlador (encargado de manejar las interacciones del usuario y la lógica de la aplicación). Esta separación facilita el mantenimiento y la escalabilidad del proyecto al aislar diferentes aspectos del desarrollo.
3. **Reutilización de código**: Al separar la lógica de presentación de la lógica de negocio, es más fácil reutilizar el código en diferentes partes de la aplicación. Por ejemplo, el mismo modelo de datos y lógica de negocio pueden ser utilizados por diferentes vistas y controladores, lo que reduce la duplicación de código y aumenta la cohesión.
4. **Facilita el desarrollo en equipo**: MVC proporciona una estructura clara y organizada para el desarrollo de aplicaciones, lo que facilita la colaboración entre diferentes miembros del equipo. Cada componente tiene responsabilidades específicas, lo que permite a los desarrolladores trabajar de manera independiente en diferentes partes de la aplicación.
5. **Mantenibilidad**: La estructura modular de MVC facilita la identificación y corrección de errores, así como la introducción de nuevas funcionalidades. Los cambios en una parte de la aplicación tienen un impacto limitado en otras partes, lo que simplifica el proceso de mantenimiento a largo plazo.
6. **Pruebas unitarias y de integración**: Debido a la separación de preocupaciones, es más fácil escribir pruebas unitarias y de integración para cada componente de la aplicación. Esto ayuda a garantizar la calidad del código y facilita la detección temprana de errores.

**Comparación con otros patrones:**

* **Patrón Monolítico**: A diferencia de un enfoque monolítico donde la lógica de presentación y la lógica de negocio están entremezcladas, MVC proporciona una separación clara de estas preocupaciones, lo que mejora la modularidad y la mantenibilidad del código.
* **Patrón de Capas (Layered Pattern)**: Si bien ambos enfoques buscan separar las diferentes responsabilidades de una aplicación, MVC proporciona una estructura más específica y definida, lo que facilita la comprensión y el desarrollo del proyecto.
* **Patrón Front Controller**: MVC puede considerarse como una extensión del patrón Front Controller, donde se separan aún más las responsabilidades del controlador en diferentes capas (controladores específicos para cada acción, por ejemplo).
* **Patrón MVVM (Modelo-Vista-VistaModelo)**: Si bien MVVM es popular en aplicaciones basadas en frameworks de JavaScript como Angular o Vue.js, MVC sigue siendo relevante especialmente en el desarrollo web tradicional con PHP, donde la separación de responsabilidades entre el Modelo, la Vista y el Controlador es fundamental para la organización y mantenibilidad del código.

En resumen, el patrón MVC ofrece una serie de ventajas clave para el desarrollo de aplicaciones web en PHP y JavaScript, incluyendo una mejor organización del código, reutilización de componentes, facilidad de mantenimiento y facilitación del trabajo en equipo. Comparado con otros patrones, MVC proporciona una estructura clara y definida que simplifica el desarrollo y la evolución del proyecto a lo largo del tiempo.

1. **JUSTIFICACIÓN TECNOLOGÍA UTILIZADA**

En esta ocasión, he decidido utilizar un framework de JavaScript para la realización del proyecto. El uso de un framework facilita la forma de trabajar con componentes, el tratamiento de la información y simplifica el uso de librerías.

El framework utilizado ha sido VUE.

**¿Por qué VUE?**

La elección del framework Vue.js para desarrollar la aplicación web que consume datos de una API REST en PHP y muestra el documento JSON obtenido, así como los datos recuperados en forma de tarjetas, se basa en varias consideraciones clave:

1. **Simplicidad y Facilidad de Aprendizaje**: Vue.js es conocido por su sintaxis intuitiva y fácil de aprender, lo que permite a los desarrolladores crear aplicaciones web de manera eficiente y con menos curva de aprendizaje. Esto es especialmente beneficioso cuando se trabaja en proyectos con plazos ajustados o cuando se requiere una rápida implementación de prototipos.
2. **Reactividad**: Vue.js ofrece un sistema de reactividad que permite actualizar automáticamente la interfaz de usuario en función de los cambios en los datos subyacentes. Esto es fundamental en una aplicación que consume datos de una API REST, ya que los datos pueden cambiar dinámicamente con el tiempo. La capacidad de Vue.js para manejar esta reactividad de manera eficiente simplifica el proceso de desarrollo y mejora la experiencia del usuario final.
3. **Componentización**: Vue.js promueve el concepto de componentización, lo que significa que la interfaz de usuario se divide en componentes reutilizables y autónomos. En el contexto de la aplicación mencionada, esto permite separar la lógica de presentación de la lógica de negocio, lo que facilita el mantenimiento y la escalabilidad del código. Por ejemplo, los datos obtenidos de la API REST pueden ser representados por un componente separado, lo que facilita su reutilización en otras partes de la aplicación.
4. **Gran Comunidad y Ecosistema**: Vue.js cuenta con una comunidad activa y un ecosistema próspero que proporciona una amplia gama de herramientas, bibliotecas y recursos para facilitar el desarrollo de aplicaciones web. Esto incluye complementos como Vue Router para el enrutamiento y Vuex para la gestión del estado de la aplicación, que pueden ser fácilmente integrados en el proyecto según sea necesario.
5. **Adaptabilidad y Flexibilidad**: Vue.js es conocido por su enfoque progresivo, lo que significa que puede ser utilizado de manera incremental en proyectos existentes sin necesidad de reescribir todo el código. Esto proporciona una mayor flexibilidad a la hora de integrar Vue.js en una aplicación preexistente que ya consume una API REST en PHP, permitiendo una transición suave y sin problemas.

En resumen, la elección de Vue.js como framework para desarrollar la aplicación web que consume datos de una API REST en PHP y muestra los resultados en forma de tarjetas se fundamenta en su simplicidad, reactividad, capacidad de componentización, comunidad activa y flexibilidad. Estas características hacen de Vue.js una opción sólida y adecuada para crear aplicaciones web modernas y dinámicas con una experiencia de usuario mejorada.

Principio del formulario

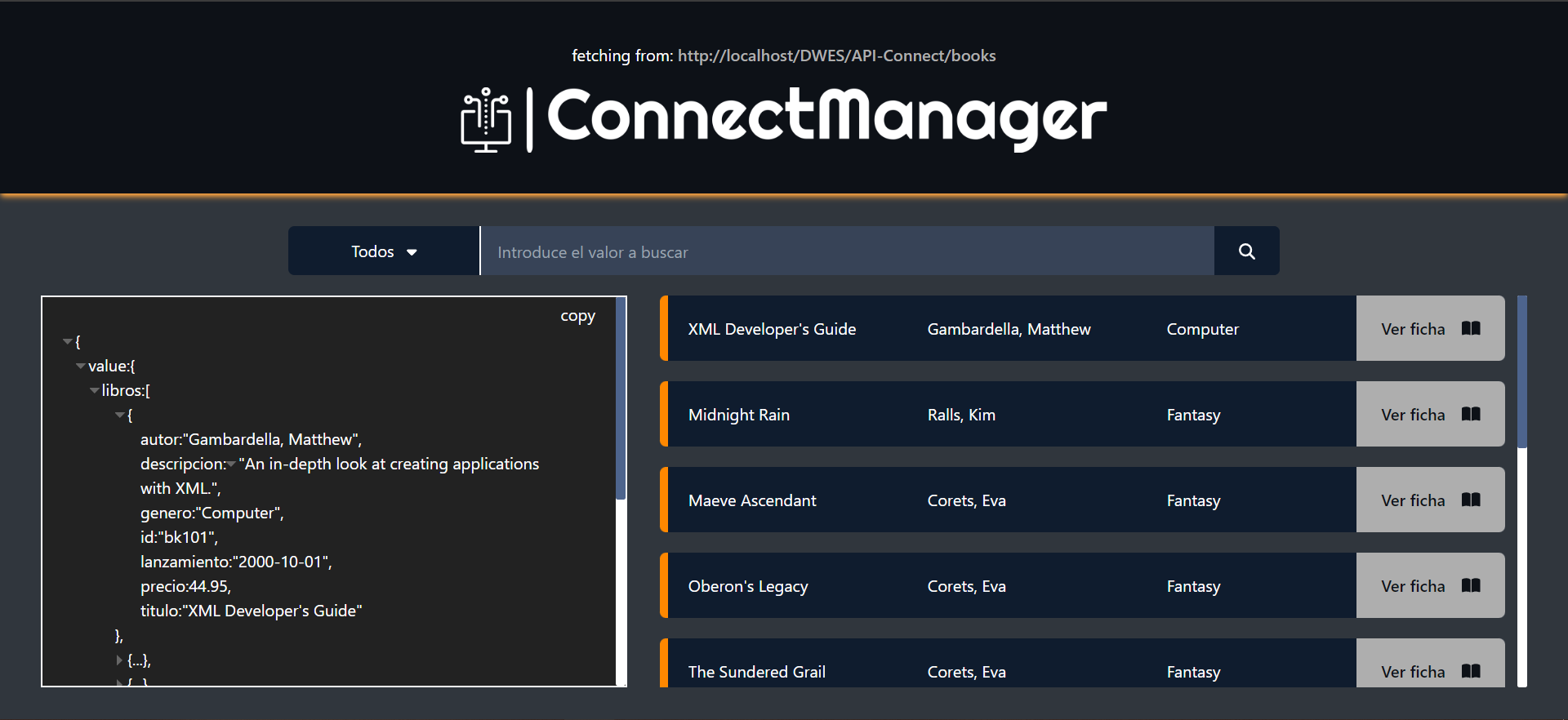
**4. DEMOSTRACIÓN FUNCIONAMIENTO**

ConnectManager necesita la configuración de ciertos archivos para su correcto funcionamiento.

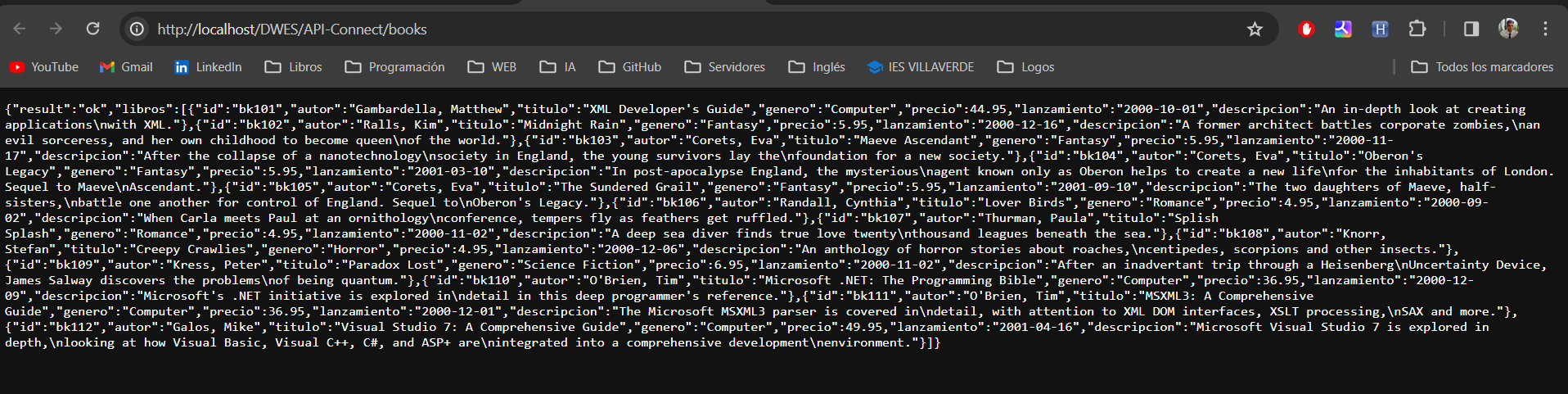
En el archivo src/config/APIConfig.js, configuraremos 3 variables necesarias para nuestra aplicación.

* + En API\_URL, introduciremos la dirección de la API a la que realizaremos las peticiones.
  + En table, introduciremos el nombre del array al que intentaremos acceder.
  + En filters, introduciremos un array de strings con los parámetros permitidos mediante los cuales vamos a realizar las peticiones parametrizadas. Estos strings se verán reflejados en nuestro componente filtro.

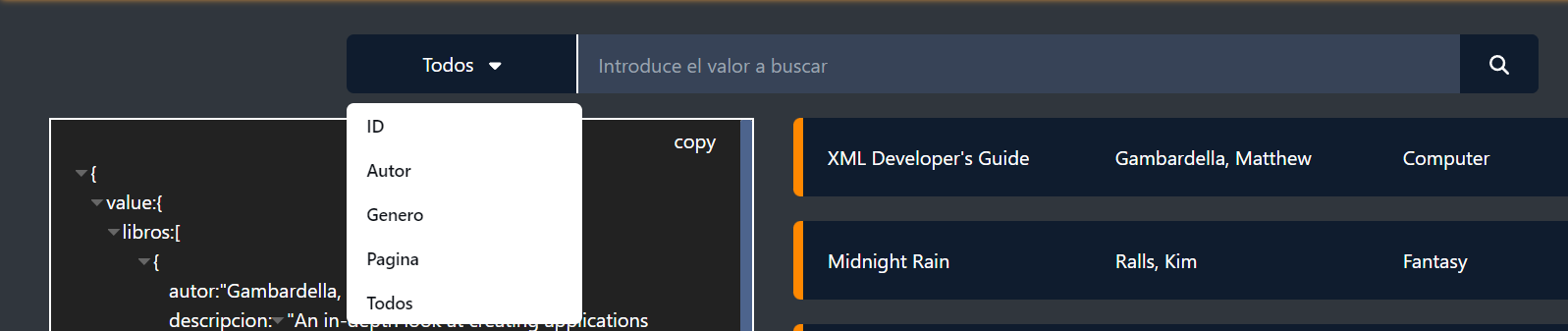
En el primer acceso a la aplicación encontramos la única pantalla que hay en la web. En la cabecera encontramos un enlace a la API-REST que está siendo consumida.



Al hacer click en el enlace, se abre una pestaña nueva mostrando todos los resultados que contiene dicha API, en formato JSON. De la manera en la que lo haría si lo introdujésemos en el navegador.



La siguiente parte importante es el filtro de parametrización de las consultas. Este componente mostrará las opciones permitidas por las peticiones de nuestra API-REST. Las opciones del filtro habrán sido introducidas previamente en el array de APIConfig.js.

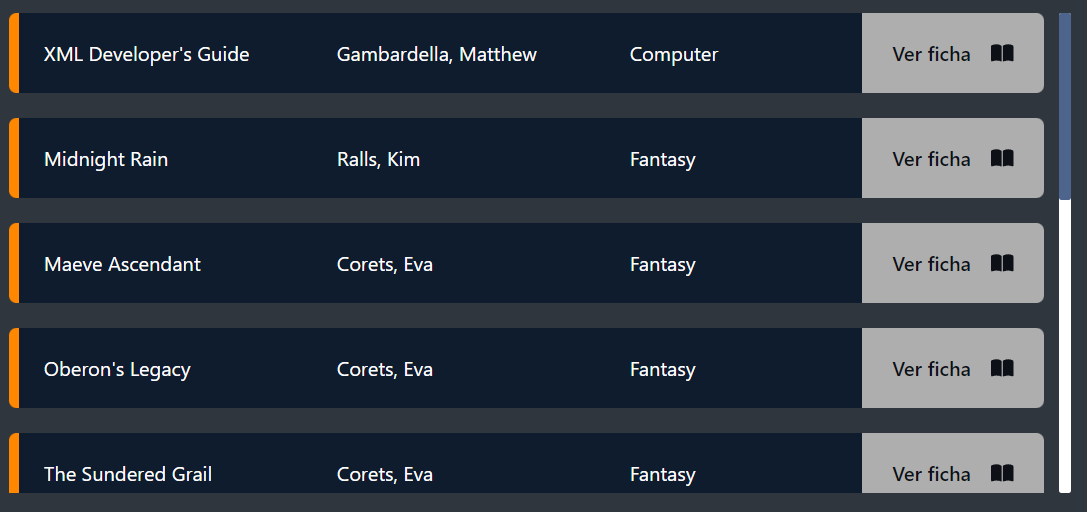


El siguiente componente a mostrar es un cuadro que muestra el resultado obtenido en formato JSON. Aquí se mostrará toda la respuesta de la petición en un formato copiable para el usuario. En la parte superior derecha del recuadro se encuentra el botón de copiar, que nos pasará todo el documento recibido al portapapeles.

Este recuadro hace que cada objeto sea desplegable, por tanto, resulta cómodo y fácil de manejar, de la misma manera que en muchas aplicaciones de consumición y prueba de APIs. Como, por ejemplo, Postman.



Los resultados obtenidos se mostrarán también en un formato de diseño agradable. Por tanto, encontramos un componente scrolleable con todas las tarjetas, cada una contiene los datos de cada uno de los objetos obtenidos en la petición.



Cada tarjeta tiene un botón de ver ficha, que abrirá una ventana modal mostrando todos sus datos en una vista más detallada del objeto.



**5. TECNOLOGÍAS UTILIZADAS**

En el desarrollo de esta aplicación web, he utilizado las siguientes herramientas:

Un entorno de desarrollo de código: Visual Studio Code con diversos plugins para facilitar y agilizar el trabajo en los lenguajes utilizados.

3 lenguajes de programación: HTML para la estructura base de la web, CSS para estilar los diseños. Y, por último, JavaScript, para aplicar la lógica a la aplicación web y la obtención de datos.

Un framework de JavaScript llamado VUE.js, que ha facilitado el desarrollo del proyecto y la componetización y reusabilidad del mismo.

Una aplicación para el control de versiones: GIT y GITHUB.

Git lo he utilizado para subir todo el proyecto en su respectiva versión mediante comandos por consola. Y en GitHub se encuentra el repositorio en la nube al que se suben los archivos.