

## A tener en cuenta

- No se corregirá código javascript en ningún archivo html.
- La única modificación admitida en los archivos html será la declaración del script correspondiente.
- Puesta en marcha de la API: npm install npm start
- Puesta en marcha del proyecto Vite: npm install npm run dev

# HARRY POTTER: PERSONAJES

RA 6	Desarrolla aplicaciones web analizando y aplicando las	Puntuación	% RA
RA 0	características del modelo de objetos del documento.	10	60%
CEv	<ul> <li>a) Se ha reconocido el modelo de objetos del documento de una página web.</li> <li>b) Se han identificado los objetos del modelo, sus propiedades y métodos.</li> <li>c) Se ha creado y verificado un código que acceda a la estructura del documento.</li> <li>d) Se han creado nuevos elementos de la estructura y modificado elementos ya existentes.</li> <li>e) Se han asociado acciones a los eventos del modelo.</li> <li>h) Se han independizado las tres capas de implementación (contenido, aspecto y comportamiento), en aplicaciones web.</li> </ul>		
RA 7	Desarrolla aplicaciones web dinámicas, reconociendo y aplicando mecanismos de comunicación asíncrona entre cliente y servidor.	Puntuación 10	% RA 100%
CEv	b) Se han analizado los mecanismos disponibles para el establecimiento de la comunicación asíncrona. c) Se han utilizado los objetos relacionados. d) Se han identificado sus propiedades y sus métodos. e) Se ha utilizado comunicación asíncrona en la actualización dinámica del documento web. f) Se han utilizado distintos formatos en el envío y recepción de información. h) Se han clasificado, analizado y utilizado librerías que faciliten y frameworks que faciliten la incorporación de las tecnologías de actualización dinámica a la programación de páginas web. i) Se han creado y probado y documentado aplicaciones web que utilicen estas librerías y frameworks.		



La aplicación permite a los usuarios visualizar una lista de personajes de la saga **Harry Potter** en un formato de tarjetas y añadir nuevos personajes a través de un formulario. A continuación, se describen las **funcionalidades** y los **archivos proporcionados**:

#### API JSON-mock Personajes:

Se proporciona una API simulada que devuelve un conjunto de personajes de Harry Potter en formato JSON. Esta API permite realizar las siguientes operaciones:

- lectura de todos los personajes
- lectura de un personaje
- **añadir** un personaje
- eliminar un personaje

Formato Datos: cada personaje tiene los siguientes atributos:

- id: identificador único del personaje.
- nombre: nombre del personaje.
- casa: la casa a la que pertenece.
- patronus: el patronus del personaje.
- **fechaNacimiento**: fecha de nacimiento del personaje.
- sangre: estado de sangre del personaje.
- varita: descripción de su varita.

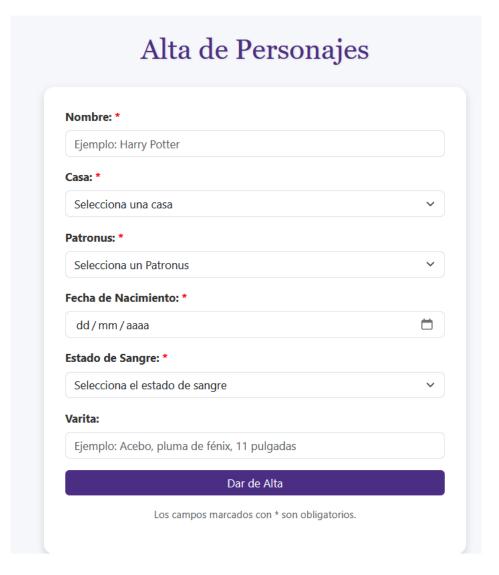
#### Formulario de Añadir Personaje:

Se proporciona el archivo **add-person.html** que contiene un formulario que permite a los usuarios **añadir nuevos personajes**. El formulario contiene los siguientes campos

- nombre: campo de texto para ingresar el nombre del personaje.
- casa: campo desplegable para indicar entre las distintas casas de Hogwarts, la del personaje.
- **patronus**: campo desplegable para indicar el patronus del personaje entre distintos *patronus*.
- **fechaNacimiento**: campo de tipo fecha para ingresar la fecha de nacimiento del personaje.
- sangre: campo desplegable donde el usuario podrá elegir los diferentes tipos de sangre (pura, mestiza, hijo de muggles) del personaje.
- varita: campo de texto para describir la varita del personaje.





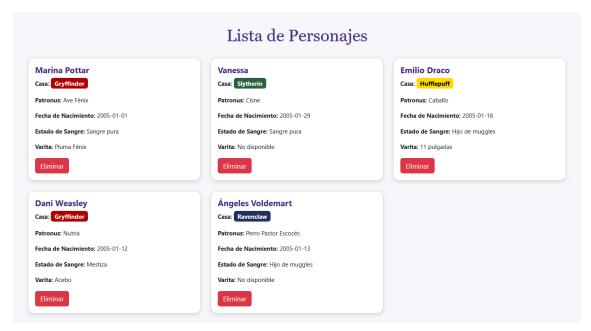


Todos los campos son obligatorios, menos el campo de texto Varita. Sobre el formulario **no se ha de realizar ninguna validación**.



### Listado de Personajes:

Se proporciona el archivo index.html que contiene un contenedor identificado como **personsContainer** donde se mostrará el listado de personajes, como se muestra a continuación:



Cada personaje se presentará en una tarjeta con su nombre, casa, patronus, fecha de nacimiento, estado de sangre y varita; por lo que, el usuario podrá visualizar todos los personajes disponibles. También contará con la opción de eliminar cualquier personaje de la lista.



Implementa las siguientes acciones sobre el proyecto de Vite proporcionado:

- Utilización de alguna clase para la creación de los Personajes.
   RA 7 (1 punto)
- 2. El formulario enviará los datos a la API JSON-mock Personajes, para que el nuevo personaje sea añadido a la lista. Para realizar el envío solo se podrá utilizar la API Fetch.
  RA 7 (2,5 puntos)
- 3. Cada vez que se añada un nuevo personaje, el nombre y la casa del mismo quedarán almacenados localmente en el navegador de manera permanente; siendo la clave el nombre del personaje y el valor la casa del personaje: RA 7 (0,75 puntos)



- 4. Cuando un personaje es añadido o cuando clicas en Inicio, se redirigirá al usuario a la página de inicio (punto de entrada a la app); mostrándose el listado de personajes de la API JSON-mock Personajes. Para obtener los datos de los personajes solo se podrá utilizar la API Fetch. RA 7 (2,5 puntos)
- 5. El archivo **index.html** contiene la *plantilla simplificada* de visualización de cada personaje. Muestra los mismos siguiendo este formato de tarjeta: RA 6 (5 puntos)







- 6. Cuando se pulsa sobre el botón Eliminar, el personaje será borrado de la API JSON-mock Personajes; así como de la visualización. Para eliminar los datos de los personajes solo se podrá utilizar XMLHttpRequest (XHR). RA 7 - (2,5 puntos)
- 7. Cuando se elimina el personaje de la API JSON-mock Personajes, también habrá que eliminar los datos almacenados localmente en el navegador para ese personaje.

RA 7 - (0,75 puntos)

8. Estructura del código. RA 6 - (5 puntos)

