**Introducción**

Para hacer que las imágenes de los personajes de la serie *Rick & Morty* aparecieran en nuestra aplicación web utilizamos Django, entre los tres hicimos una serie de pasos clave. A continuación, se describe el proceso de implementación, destacando las funciones desarrolladas y los ajustes necesarios en los archivos de código relevantes.

**Desarrollo**

**1. Configuración Inicial**

Inicialmente, nuestra aplicación contaba con varias plantillas HTML y una estructura de carpetas bien organizada, pero las funciones clave no estaban completamente implementadas. Identificamos las funciones faltantes y procedimos a completarlas.

**2. Obtención de Datos desde la API**

Para obtener los datos de los personajes, utilizamos la función ***fetch\_characters()*** desde el archivo transport.py. Esta función se encarga de llamar a la API de *Rick & Morty* y obtener la lista de personajes.

**3. Conversión de Datos Crudos a Objetos Card**

Implementamos la función ***getAllImages*** en el archivo services.py. Esta función:

* Llama a ***fetch\_characters*** para obtener los datos crudos.
* Recorre cada dato crudo y lo convierte en un objeto Card utilizando la función ***fromRequestIntoCard***.
* Filtra los resultados basándose en el parámetro de búsqueda input si está presente.
* Devuelve una lista de objetos Card que se utilizarán para mostrar en la web.

**4. Renderización de la Página Principal**

En el archivo views.py, completamos la función home, que:

* Llama a ***getAllImages*** para obtener la lista de imágenes.
* Pasa estas imágenes al template **home.html** para ser renderizadas.

**Modificaciones en el Template home.html**

Primero, se corrigieron las condiciones en el bloque **{% if %}** dentro del bucle **{% for img in images %}** del archivo home.html. Anteriormente, se intentaba comparar una variable true con los posibles estados, lo cual era incorrecto y no producía el resultado deseado. Se ajustaron las condiciones para comparar correctamente el atributo **img.status** con las cadenas 'Alive', 'Dead', y otros valores posibles.

**Modificación para mostrar un “borde” de color según el estado de personaje**

Para mejorar el estilo de las tarjetas de los personajes en la página home.html, hice una serie de modificaciones que permiten cambiar el borde de las tarjetas según el estado del personaje (vivo, muerto o desconocido). A continuación, paso a detallar un poco más lo que hice.

En primer lugar, me aseguré de identificar correctamente la sección del código donde se renderizan las tarjetas de los personajes dentro de un bucle **for**.

El siguiente paso fue añadir una clase condicional al contenedor principal de cada tarjeta. Utilicé los condicionales de Django para verificar el estado del personaje (**img.status**) y así aplicar la clase de borde correspondiente. Añadí una lógica para que, si el personaje está vivo (Alive), la tarjeta tenga un borde verde (**border-success**). Si está muerto (Dead), el borde sea rojo **(border-danger**). Y si el estado es desconocido (unknown), el borde sea naranja (**border-warning**).

***Sitios utilizados:***

1. [***https://getbootstrap.com/docs/4.0/components/card/***](https://getbootstrap.com/docs/4.0/components/card/)*“Bootstrap’s cards”*
2. [***https://medium.com/powered-by-django/if-else-conditions-in-django-templates-5b3658cbf287***](https://medium.com/powered-by-django/if-else-conditions-in-django-templates-5b3658cbf287) *“How To Use If/Else Conditions In Django Templates”*
3. [***https://es.stackoverflow.com/questions/461609/como-hacer-una-condicional-en-plantilla-html-de-django-python***](https://es.stackoverflow.com/questions/461609/como-hacer-una-condicional-en-plantilla-html-de-django-python)*“*[*Como hacer una condicional en plantilla html de django. Python*](https://es.stackoverflow.com/questions/461609/como-hacer-una-condicional-en-plantilla-html-de-django-python)*”*