Introducción a la Programación

Trabajo Practico:

Galería de imágenes de la Nasa 🚀

Alumnos:

Nicolás Falzetta

Natalia Evangelista

Ezequiel Díaz

Comisión: 10

Docentes:

José Luis Mieres

Sergio Santa Cruz

Miguel Rodríguez

2024

# Introduccion

En este proyecto, tenemos la intención de implementar una aplicación web fullstack utilizando el framework Django. La aplicación permitirá a los usuarios consultar y explorar imágenes obtenidas de la API pública de la NASA.

Funcionalidades claves de la aplicación:

Visualización de Imágenes: La aplicación realizará consultas a la API de la NASA y presentará las imágenes obtenidas en una serie de cards, la cual contendrá la imagen, un título y una breve descripción.

Buscador: Se implementará un buscador que permitirá a los usuarios filtrar las imágenes mostradas según sus intereses.

Autenticación: También incluirá un sistema de autenticación básica que permitirá a los usuarios iniciar sesión.

Favoritos: Los usuarios autenticados podrán marcar imágenes como favoritas. Estas imágenes serán almacenadas en la base de datos y estarán disponibles para consulta posterior.

Paginación: La funcionalidad de paginación se implementará para manejar y presentar un gran volumen de imágenes de manera eficiente. Los usuarios podrán seleccionar el número de página y la cantidad de elementos a mostrar en ella.

# Lo que esta implementado

Para la realización del proyecto se nos proporcionó en términos de estructura 4 HTML:

El header.html incluye la lógica para determinar si un usuario está logueado y muestra el nombre del usuario si está autenticado.

El home.html es la sección principal donde se muestran las imágenes de la API y se incluye un buscador, así como un algoritmo que recorre y dibuja cada objeto de la API en pantalla.

El footer.html contiene el pie de página, aunque no tiene acciones de código relevantes para el desarrollo.

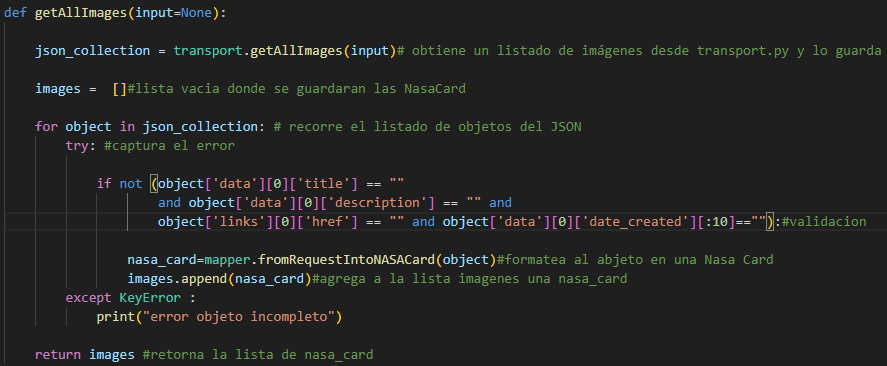
El index.html es el contenedor principal que incluye los 3 templates anteriormente mencionados.

En el archivo views.py, se encuentran funciones clave para el funcionamiento de la aplicación. La función index\_page(request) renderiza el contenido de index.html. La función home(request) obtiene todas las imágenes de la API y los favoritos del usuario mediante la función auxiliar getAllImagesAndFavouriteList(request), que devuelve dos listas: una con las imágenes de la API y otra con las imágenes marcadas como favoritas por el usuario. Esta función luego renderiza home.html y le pasa la información obtenida.

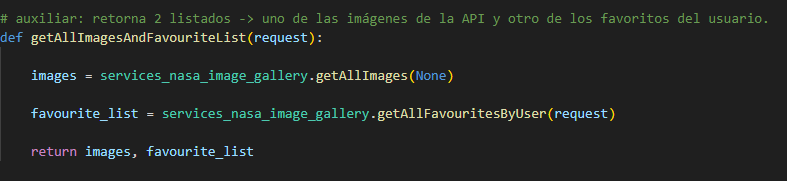
La lógica de servicio está contenida en transport.py, que tiene todo el código necesario para consumir la API de la NASA, gestionando las solicitudes a la API y la recepción de datos. En mapper.py, se incluye la lógica para convertir y mapear los resultados de la API en objetos NASA Card, que son los objetos utilizados en el template para dibujar los resultados en pantalla.

# Funciones básicas requeridas

Éstas son las encargadas de hacer que las imágenes de la galería se muestren/rendericen.

**getAllImages(input=None):**

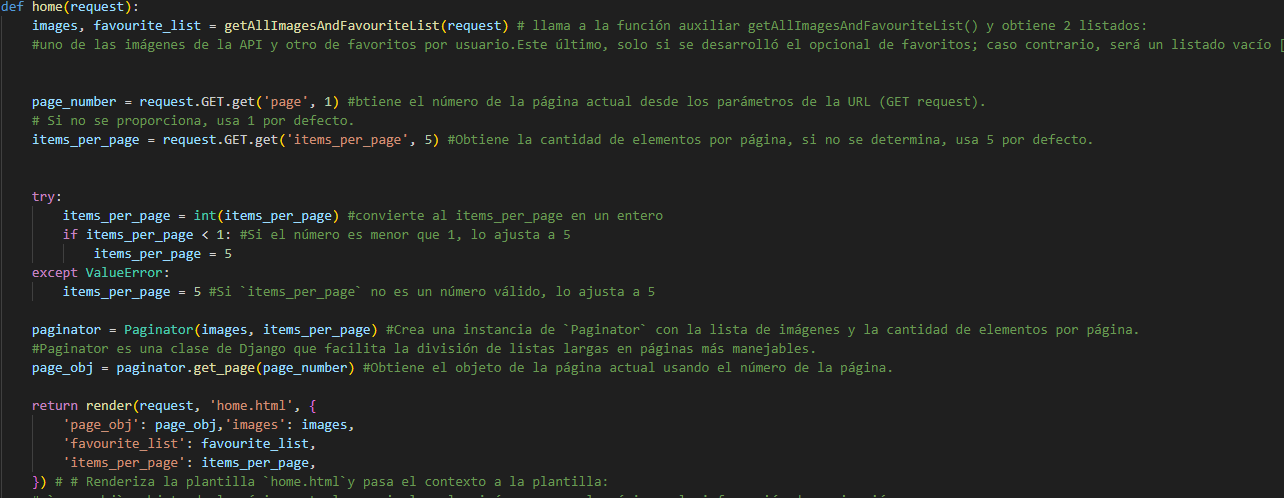
La finalidad de esta función es retornar una lista de imágenes, las cuales tendrán una estructura de Nasa Card. Para ello, en la variable json\_collection se asigna la función getAllImages que se encuentra en el archivo transport.py para obtener la información de la API de la Nasa. Una vez obtenida la misma, recorre cada objeto del json con la intención de convertirlo en Nasa Card a través de la función fromRequestIntoNasaCard, para luego agregarlo en la lista images. Fue necesario validar cada campo requerido, eso quiere decir que el objeto del json debía tener imagen, titulo, descripción y fecha antes de sumarlo a la lista, ya que en algunas de nuestras pruebas (buscando la palabra marte) aparecía un Key Error. El mismo indicaba que el campo de descripción no existía en el objeto.

**getAllImagesAndFavouriteList:**

El objetivo de esta función es retornar dos listas:

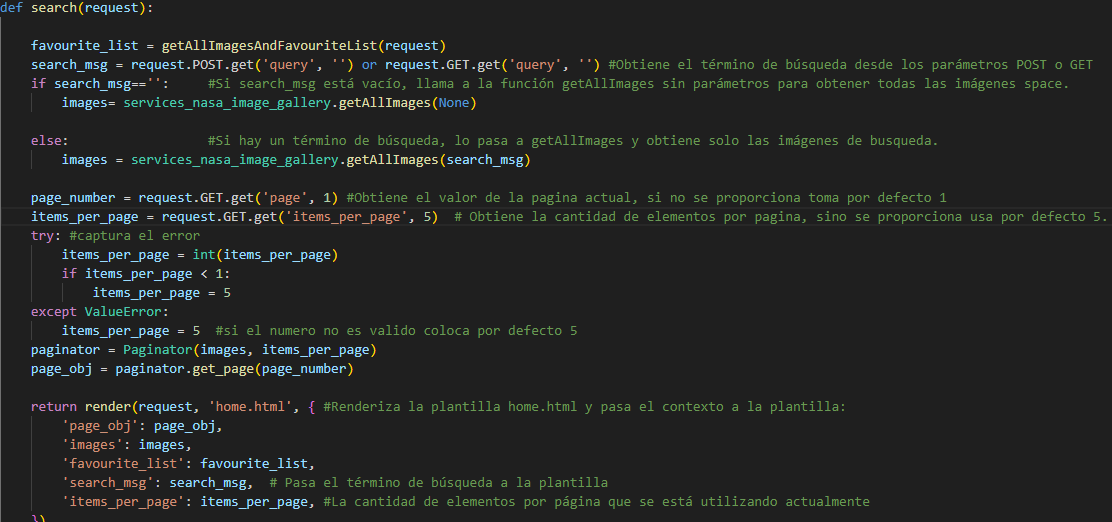
Images: contiene una lista de Nasa Card a través de la función getAllImages donde el parámetro none indica que no se ha realizado una petición de búsqueda y por lo tanto mostrará al valor determinado “space”.

Favourite\_list: contiene una lista de Nasa Card de los favoritos del usuario a través de la función getAllFavouriteByUser siempre y cuando el mismo se encuentre autenticado, de no ser así mostrará una lista vacía.

**def home (request):**

El objetivo de la función home es redirigir las listas de imágenes y favoritos a home.html utilizando la función auxiliar getAllImagesAndFauvoriteList. Además, hemos implementado la paginación de resultados la misma se explicará en “Opcionales”.

# Opcionales

**def search (request)**

La función buscar es la encargada de filtrar las imágenes según el criterio de búsqueda que coloque el usuario, con algunas consideraciones. La variable search\_msg obtiene el termino de búsqueda desde Post o Get, si la misma se encuentra vacía devuelve una lista de Nasa Card de “space” según la configuración del sistema ubicado en config.py. Sin embargo, si search\_msg contiene un término devuelve una lista de Nasa Card según el criterio definido por el usuario. En conclusión, la función renderiza a home.html la lista images según el parámetro de búsqueda y la lista de favourite\_list por defecto.

**Paginación de Resultados**

Con el propósito de que el usuario pueda definir la cantidad de Nasa Card a visualizar por página y además se desplace por las mismas a través de las opciones: anterior, siguiente, ultima y primera, se importó el módulo de Django core.paginator. Fue necesario aplicar la paginación en la función home como search, porque si el usuario no realiza ninguna búsqueda debe permitirle de todos modos recorrer por página. La variable page\_number obtiene el valor de la página actual, sino se proporciona toma por defecto 1. Item\_per\_page obtiene la cantidad de elementos por página, sino se asigna algún valor mostrará por defecto 5 imágenes. Luego se convierte a ítem\_per\_page en un entero, con el propósito de evitar algún tipo de error en el ingreso del numero por parte del usuario, para ello se realizó un condicional donde si el ítem es menor a 1, muestre 5 imágenes o si existe algún número no valido mostrará por defecto 5 imágenes. En la variable paginator se asigna Paginator de Django donde se abre una instancia de paginación con la lista de imágenes y de ítems por página. Por último, page\_obj obtiene el objeto de la página actual usando el número de página.

Ambas funciones tanto home como search rederizan a home html además de las listas de imágenes, page\_obj y ítems\_per\_page.