

### Licenciatura en Sistemas

# Trabajos Práctico

Nombre del TP

### Introducción a la Programación

(1er Semestre - 2024)

Resumen: Este trabajo practico final consiste en desarrollar una aplicación web utilizando Django para mostrar una galería de imágenes utilizando la API de la NASA.

Integrantes: Nicolas Fernando Vargas *nicofvargass@gmail.com*

**1. Introducción** El trabajo se centra en completar y crear funciones faltantes para visualizar una galería de imágenes a través de una API. La aplicación debe permitir al usuario visualizar estas imágenes y realizar busquedas.

**2. Desarrollo** Para resolver este problema analizamos los siguientes archivos: *views.py, mapper.py, transport.py* y *services\_nasa\_image\_gallery.py. El inicio de la aplicación empieza por transport.py haciendo un llamado a la API obteniendo un json que es usado por mapper.py para convertir los datos a un NASACard. El principal problema fue encontrar las variables que se relacionan entre si y la forma de tratar la información.*

**2.1 Descripción general:** La solución fue modificar dos archivos views.py y services\_nasa\_image\_gallery.py el cual son los encargados de hacer uso de los NASACard para visualizar las imágenes y buscar información a través de la API.

**2.2 Funcionalidades principales:**

En views.py utlizamos las siguientes funciones:

**-** getAllImagesAndFavouriteList: Esta función hace llamado de la función getAllImages en el archivo services\_nasa\_image\_gallery.py la cual se encarga de leer el json y convertirlo a un NASACard individual que contiene las imágenes.

def getAllImagesAndFavouriteList(request):

    images = services\_nasa\_image\_gallery.getAllImages()

    favourite\_list = []

    return images, favourite\_list

**-** home: Esta función es la principal de views.py la cual se encarga de renderizar el template de inicio llamado home.html. A su vez, hace llamado de la función establecida anteriormente getAllImagesAndFavouriteList

def home(request):

    images, favourite\_list = getAllImagesAndFavouriteList(request)

    return render(request, 'home.html', {'images': images, 'favourite\_list': favourite\_list} )

**-** search: Esta función es la encargada de utilizar el buscador, la función consiste en analizar el contenido de la variable search\_msg obtenido a traves de un POST desde el template home.html. Si el valor de dicha variable es vacio se establece su valor como predeterminado “space” como indica la consigna, caso contrario, deja el valor obtenido a través del POST.

def search(request):

    search\_msg = request.POST.get('query', '')

    #si search\_msg no contiene un valor se le da el valor predeterminado de busqueda como "space"

    if not search\_msg:

        search\_msg = "space"

    images = services\_nasa\_image\_gallery.getImagesBySearchInputLike(search\_msg)

    favourite\_list = []

    return render(request, 'home.html', {'images': images, 'favourite\_list': favourite\_list})

En services\_nasa\_image\_gallery.py utlizamos la función getAllImages la cual es la principal encargada de que se muestren las imágenes.

**-** getAllImages: Esta función guarda el json obtenido desde transport.py en la variable json\_collection, para luego, iterar sobre ese conjunto de datos y convertirlo en un NASACard individual que obtiene imagen, titulo, descripción y demás.

def getAllImages(input=None):

    json\_collection = transport.getAllImages(input)

    images = []

    for objeto in json\_collection:

        images.append(mapper.fromRequestIntoNASACard(objeto)) #usamos mapper llamando desde la funcion Request que se usa cuando la info viene desde la api

    return images

**3. Conclusiones:** En conclusión para poder lograr este trabajo fue primordial entender cual es el orden de ejecución de las funciones principales para luego poder trabajar sobre ellas. Muy útil para familiarizarse en el uso de apis para proyectos y pruebas.