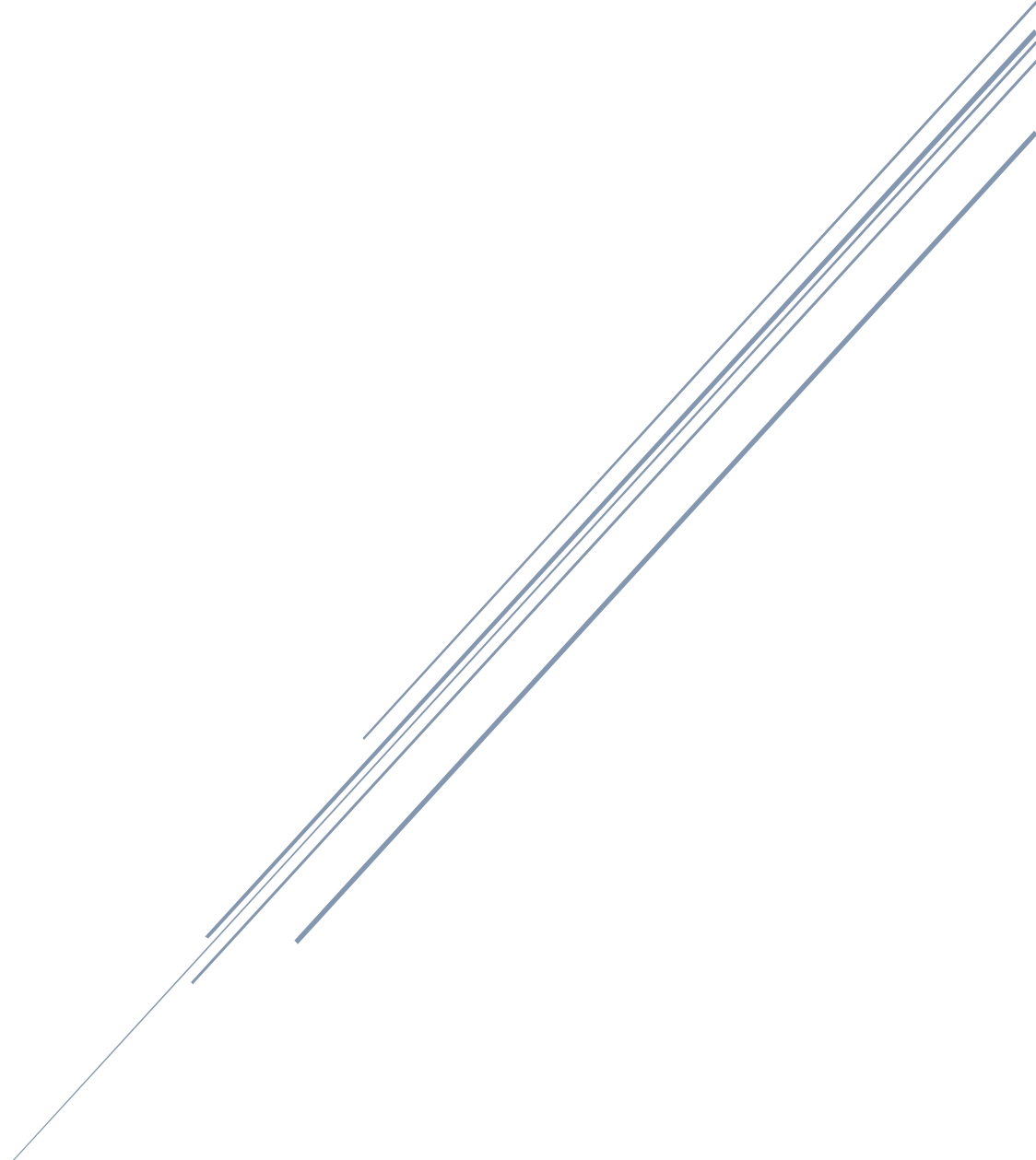


**TRABAJO PATRACTICO FINAL INTEGRADOR:**  
**INTRODUCCION A LA PROGRAMACION**  
**COM 7**



Profesores: Sergio Santa Cruz – Nahuel Sauma – Naza Avalos

Integrantes: Nahuel Hernán Gómez DNI: 40193507 – Diego Iván Gómez DNI: 42459934

## INFORME:

Estimados, este es un informe acerca del Trabajo Practico Final de "Introducción a la Programación" este trabajo tiene el fin de desarrollar una página para poder mostrar información publicada por la N.A.S.A. dicho trabajo es un código incompleto en donde nosotros debemos implementar los conocimientos adquiridos para lograr hacer correr la página.

Este trabajo consta de dos partes en primer lugar poder desarrollar las primeras tres funciones necesarias para poder aprobar el Trabajo Practico Final llamadas como "home", "getAllImagesAndFavouriteList" y "getAllImages"; en segundo lugar, poder desarrollar distintos "adicionales" para generar más interacciones en las paginas tales como, por ejemplo: implementar un buscador, lograr un inicio de sesión exitoso, lograr tener una lista de favoritos por usuario, entre otras.

Es correcto decir que al principio nos costó mucho porque no teníamos conocimientos sobre GitHub, no trabajamos nunca en un proyecto de programación y tratamos de entender todo lo que teníamos enfrente lo cual nos frustró mucho a tal punto que pensamos que no íbamos a poder lograr ni siquiera poder desarrollar las primeras tres funciones que nos ayudarían a aprobar el Trabajo Practico Final. Esto sucedió hasta que decidimos empezar por las funciones necesarias para la aprobación y buscar el orden entre estas, el cual fue el siguiente:

### GET ALL IMAGES (getAllImages):

Logramos entender que la función en "transport.py" bajo el mismo nombre nos daba una lista, lista que se nombraba de la misma manera en "services\_nasa\_image\_gallery.py", ahí tuvimos el primer problema indagando y entendiendo que debíamos de llamar, en donde estaba la función, que queríamos correr, como hicimos "transport.getAllImages(input)", de igual manera, como sabíamos que era una lista el siguiente paso era leerla y convertir esa información que teníamos por objeto en una "NasaCards" para luego incluir las mismas en una lista llamada "images" la cual retorna la función.

```
def getAllImages(input=None):  
    json_collection = transport.getAllImages(input)  
    images = []  
    for json_obj in json_collection:  
        nasacard = mapper.fromRequestIntoNASACard(json_obj)  
        images.append(nasacard)
```

### GET ALL IMAGES AND FAVOURITE LIST (getAllImagesAndFavouriteList):

Entendimos que esta función separaba las “NasaCards” de las imágenes en lista de favoritos y como no buscábamos desarrollar el punto de favoritos tomamos la decisión de implementar toda la lista "images" en una lista nombrada de igual manera para retornar lo mismo.

```
def getAllImagesAndFavouriteList(request, input):  
    images = services_nasa_image_gallery.getAllImages(input)
```

### GALERIA (HOME):

De igual manera que la función anterior hicimos que "getAllImagesAndFavouriteList" devuelva la lista de “NadaCards” de la lista "images" nuevamente a otra lista nombrada de la misma manera para lograr correrla.

```
def home(request, input=None):  
    images = getAllImagesAndFavouriteList(request, input)  
  
    return render(request, 'home.html', {'images': images} )
```

En principio no fue difícil lograr hacer funcionar las primeras tres funciones, luego de esto decidimos buscar implementar las funciones "search", "login\_view", el gif de carga y hacer modificaciones en la página como adicionales las cuales nos llevó mucho tiempo en conjunto con mucha búsqueda de información ya sea en videos explicativos tanto en inglés como en español, sino que además buscas funciones de Python adicionales que utilizaban en dichos videos.

### BUSCADOR (SEARCH)

Buscamos poder hacer correr la función pre armada, nombramos una variable y buscamos la manera de implementarle el “input” que el usuario ponga en el buscador, luego con un “If” que redirecciona a la función home ,que son las listas de imágenes que desarrollamos anteriormente, en caso que el usuario buscara alguna palabra usamos de ejemplo "Sun" o "Moon" y busco dentro de la lista de “Nasa-Cards” información relacionada con dicho input, el único problema que tuvimos fue el entender su lógica.

```
def search(request):  
    search_msg = request.GET.get('query', '')  
  
    if search_msg == "":  
        return home(request)  
    else:  
        return home(request, search_msg)
```

### INICIO DE SECCION (LOGIN VIEW):

Se busco poder desarrollar el inicio de sesión de la siguiente manera, según encontramos en internet, se busca verificar que se quiera iniciar sesión se obtiene el nombre de usuario y su contraseña para luego autenticar que sea un usuario, si la autenticación es correcta directamente redirige al usuario al inicio para que empiece a navegar, si es incorrecto alguno de los dos inputs muestra un mensaje que aparece diciendo "usuario o contraseña incorrectos", el problema que tuvimos fue que cuando se ponían inputs incorrectos mostraba una lista de carteles diciendo "usuario o contraseña incorrectos" el cual solucionamos cuando encontramos que debíamos de poner una variable que marque que el mensaje ya fue usado para que no vuelva a repetirlo.

```
def login_view(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('index')
        else:
            messages.get_messages(request).used = True
            messages.error(request, 'Usuario o contraseña incorrectos')
    return render(request, 'registration/login.html')
```

### GIF DE CARGA

Con este punto podemos encontrar su función (script) en home.html, index.html y login.html si bien no encontramos la forma de implementarlo en Python si es cierto que encontramos tutoriales que lo hacían según entendemos con JavaScript y es por eso que es un código que no conocemos pero a pesar de estas limitantes logramos hacerlo correr la explicación de la función es que en primer lugar el gif se va a ejecutar cada vez que el documento este completamente cargado y listo, luego cada vez que se envié un formulario, se haga cualquier click en algún enlace o botón funcione el "logo de carga", pero no funcionaba en las siguientes instancias: si estábamos en el inicio de sesión y queríamos ir a galería no funcionaba, si estábamos en el inicio y queríamos ir a galería no funcionaba, si estamos iniciados como "Admin" y salíamos no funcionaba si logiábamos no funcionaba, tras buscar y buscar por qué no funcionaba nos topamos con un foro que decía de implementarlo en los HTML que queramos que funcione buscamos la manera y lo hicimos, a sabiendas que no es lo más prolijo, pero ahora funciona con cada comando de la página.

```

<script>
    $(document).ready(function(){
        // Mostrar el spinner al enviar cualquier formulario
        $('form').on('submit', function() {
            $('#spinner').show();
        });

        // Mostrar el spinner al hacer clic en cualquier enlace
        $('a').on('click', function() {
            $('#spinner').show();
        });

        // Mostrar el spinner al hacer clic en cualquier botón
        $('button').on('click', function() {
            $('#spinner').show();
        });
    });
</script>

```

### **RETOQUES EN LA PAGINA:**

Se realizaron algunas modificaciones en el estilo de la página, aunque no son la gran cosa, cambia un poco la visualización de la misma.

Dichas modificaciones fueron:

Nav-bar: cambiar el color del fondo, cambiar el color de la tipografía y ponerla en negrita, arreglar la distancia entre el logo de “Bienvenidos” y el de “Inicio de sesión” con respecto al nav ya que generaba una colisión y al cambiar el color del nav quedaba mal.

Galería: se modificó las “Nasa-cards” ya que no ocupaban todo el espacio disponible y eso ocasionaba que haya algunas cards más chicas que otras.

Generales: se colocaron todos los estilos en una única pagina de estilos (styles.css), aunque algunas que ya estaban implementadas en el código nativo no se movieron.

### **CONCLUCIO FINAL:**

Para finalizar tras haber logrado terminar los puntos claves para la aprobación y los puntos adicionales ya nombramos por los títulos "search", "login", "gif de carga" y “retoques en la página” alguna modificación para que pueda verse más estético concluimos que: logramos aprender que la programación no es algo sencillo para un programador principiante, hay un universo muy amplio el cual uno debe de investigar para poder entender el funcionamiento de cada función y que además de esto debemos desarrollar nuestra lógica comprensiva acerca de las funciones establecidas o a desarrollar que nos toque trabajar. Gracias por su tiempo.