INTRODUCCIÓN A LA PROGRAMACÍON
Comisión: 02
Docentes:
Bottino, Flavia
Winograd, Natalia

26-6-2024

Galería de Imágenes de la Nasa

App Web

GRUPO: 04
INTEGRANTES:
Antúnez, Agustina
Casco, Agustín
Núñez, Gonzalo
Ramírez, Facundo

Informe

Introducción:

En el presente informe vamos a explicar y demostrar todo lo aprendido y entendido acerca de cómo desarrollar una galería de imágenes conformada por objetos, a los que llamamos NASACard. La cual, está constituida por una imagen\un título\una descripción; Cuya consigna es presentada de la siguiente manera:

Desarrollar tres funciones: *home*, *getAllImagesAndFavouriteList* y *getAllImages* las cuales, permiten visualizar la galería de imágenes mencionada anteriormente. El objetivo es tratar de lograr que se puedan desplegar las NASACard en el buscador junto con toda su información.

Para ello, debimos realizar una serie de pasos para poder realizar el proyecto. A continuación, pasaremos a enumerar dichos procedimientos:

1- Descargar e instalar Visual Studio Code

2- Instalar Python

3- Crear una cuenta en GitHub (para poder ejecutar y avanzar con el desarrollo del proyecto).

Una vez realizados los pasos, procedimos con su elaboración.

DESARROLLO:

Como grupo resolvimos nuestras diferencias e inquietudes sobre el tema. Debatimos y analizamos la mejor manera de llevar a cabo los procedimientos que utilizamos para el correcto funcionamiento del proyecto.

Seguidamente, pasaremos a explicar con más detalles los métodos empleados:

a- Archivo *views.py* (lugar donde se encuentran dos funciones semi desarrolladas)

- *getAllImagesAndFavouriteList*

```
def getAllImagesAndFavouriteList(request):
    images = []
    favourite_list = []
    images= services_nasa_image_gallery.getAllImages()

    return images, favourite_list
```

En esta función, utilizamos la variable ya creada *images.* En ella, le agregamos *services_nasa_image_gallery,* puesto que es el lugar donde se encuentran todas las imágenes de la api. También, colocamos la función getAllImages de esa manera, el usuario puede obtener las imágenes y alojarlas en su lista de favoritos.

- *home*

```
# función principal de la galería.
def home(request):
    # llama a la función auxiliar getAllImagesAndFavouriteList() y obtiene 2 listados: uno de l
    # (*) este último, solo si se desarrolló el opcional de favoritos; caso contrario, será un
    images = []
    favourite_list = []
    images,favourite_list= getAllImagesAndFavouriteList(request)
    return render(request, 'home.html', {'images': images, 'favourite_list': favourite_list} )
```

En esta función (la principal) para poder traer las imágenes de la api, utilizamos la función ya desarrollada *getAllImagesAndFavouriteList.* Para lo cual, juntamos en una sola variable a *images y favourite_list* .

b- Archivo *services_nasa_image_gallery.py*

- *getAllImages*

```
    def getAllImages(input=None):
        # obtiene un listado de imágenes desde transport.py y lo
        # ¡OJO! el parámetro 'input' indica si se debe buscar po
        json_collection = []
        images = []
        json_collection= transport.getAllImages(input)
        for foto in json_collection:
            images.append(mapper.fromRequestIntoNASACard(foto))

        # recorre el listado de objetos JSON, lo transforma en u

        return images
```

En esta última función, utilizamos la variable *json_collection* para poder llamar a la función *getAllImages* puesto que es la encargada de obtener las

imágenes que se encuentran dentro de la galería. Luego, recorremos la lista *Json_collection,* así pues, obtenemos las NASACard.

De esta forma, pudimos lograr visualizar las imágenes de la galería.

Finalizadas las funciones obligatorias, a modo de complemento realizamos la siguientes funciones:

a- Buscador

```
# función utilizada en el buscador.
def search(request):
    images, favourite_list = getAllImagesAndFavouriteList(request)
    search_msg = request.POST.get('query', '')
    images= services_nasa_image_gallery.getAllImages(search_msg)

    # si el usuario no ingresó texto alguno, debe refrescar la página; caso contrario, debe filt
    return render (request, 'home.html', {'images': images, 'favourite_list': favourite_list})
```

En esta función, agregamos la variable *images*. En ella, colocamos *services_nasa_image_gallery* puesto que, es el lugar donde se encuentran las imágenes de la api. Luego, utilizamos la función *getAllImages pasándole* como parámetro la variable *search_msg* de esa manera, el usuario podrá buscar las imágenes que desee.

b- Inicio de Sesión

Para poder implementar esta función realizamos las siguientes modificaciones:

- Archivo main/ *urls.py*

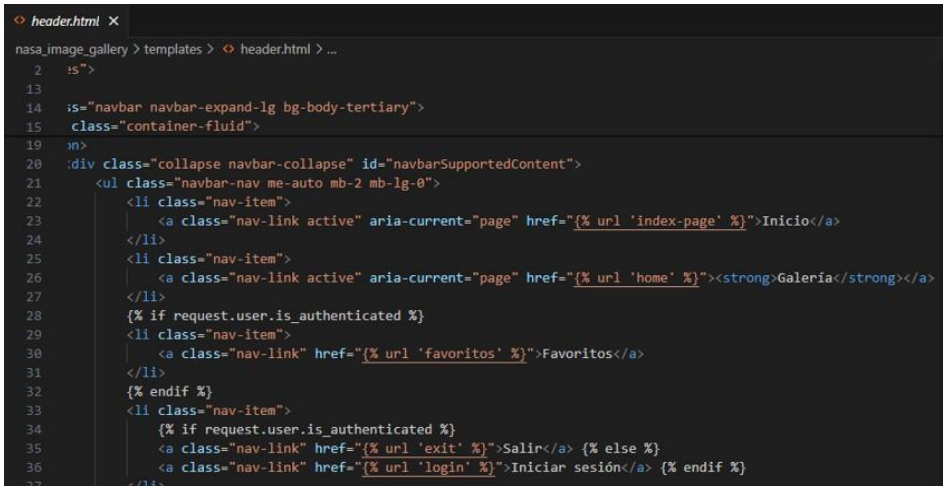
urls.py

main > urls.py > ...

```
1  from django.contrib import admin
2  from django.urls import include, path
3
4  urlpatterns = [
5      path('admin/', admin.site.urls),
6      path('', include('nasa_image_gallery.urls')),
7      path('accounts/',include('django.contrib.auth.urls'))
8  ]
```

Acá, agregamos path (‘accounts/’,include (‘django.contrib.auth.urls’)) para referenciar el sistema de autenticación de Django.

Luego,



En este archivo *HTML* agregamos la *url*, que sirve tanto para salir, como para iniciar sesión. (El usuario y contraseña a utilizar, preliminarmente, es **admin** / **admin** ya se encuentra guardado en la base SQLite, tabla *auth_user*). Biblioteca donde se halla la base de datos del proyecto.

Por último,

```
@login_required
def exit(request):
    logout(request)
    return redirect('/')
```

Cambiamos esta función, para poder indicarle al usuario que debe "desloguearse" es decir, cerrar sesión. De esa manera, podrá redirigirse a la página principal.

A modo de cierre, podemos decir que todo el desarrollo de este proyecto nos resultó bastante interesante, pero a la vez difícil. Al final logramos resolverlo de forma ordenada y prolija.