UNIVERSIDAD NACIONAL DE GENERAL SARMIENTO

Introducción a la Programación

Trabajo práctico: galería de imágenes de la NASA

Docentes: Sergio Santa Cruz, Nazareno Avalos , Nahuel Sauma.

Estudiantes: Zarate, Agustin. Benítez, Lorena, A.

Comisión: 07

Fecha de entrega: 28/06/2024

Tabla de contenido:

Galería de la Imágenes NASA

Introducción

El presente proyecto tiene como objetivo la implementación de una aplicación web fullstack utilizando el framework Django, que permita consultar las imágenes provenientes de la API pública de la NASA. El contenido obtenido de esta API será presentado en cards que incluirán, al menos, la imagen correspondiente, un título y una descripción. Nuestra tarea implica el desarrollo de las funcionalidades necesarias para habilitar un buscador central y un sistema básico de autenticación (usuario/contraseña) para almacenar y gestionar favoritos, de forma tal, que los usuarios puedan marcar imágenes como favoritas y acceder a ellas una vez iniciada sesión, requiriendo que la aplicación disponga de la lógica adecuada para verificar esta información.

Para la realización del trabajo, se nos pide trabajar como mínimo con tres funciones básicas del código, las cuales se encontraban distribuidas de la siguiente manera:

- Carpeta ***view.py***: Estaban las funciones ***home*** y ***getAllImagesAndFavouriteList***;
- Carpeta ***service_nasa_image_gallery.py***, dentro de la capa ***services***: Esteban la función ***getAllImages***.

Funciones

def getAllImaches():

```
def getAllImages(input=None):
    json_collection = []
    images = []
    return images
```

Comenzamos usando el *array json_collection* para guardar en forma de objeto los datos de cada una de las imágenes (foto, título y descripción). Luego iniciamos un *array* vacío con el nombre *imágenes* e hicimos un código *for* para iterar cada uno de los objetos dentro del *array json_collection* (anteriormente llenado con los objetos de las imágenes). Posteriormente armamos un card con la función *mapper.py.* Como última acción a cada *nasaCard* generada se la appendea al *array* imágenes(anteriormente iniciado vacío) y se retorna el *array* completo con todas las cards por pintar en las otras dos funciones.

```
def getAllImages(input=None):
    # obtiene un listado de imágenes desde transport.py y lo guarda en un json_collection.
    # ¡OJO! el parámetro 'input' indica si se debe buscar por un valor introducido en el buscador.
    json_collection = transport.getAllImages(input)

    images = []
    for object in json_collection:
        nasaCard = mapper.fromRequestIntoNASACard(object)
        images.append(nasaCard)
    # recorre el listado de objetos JSON, lo transforma en una NASACard y lo agrega en el listado de images. Ayuda: ver mapper.py.

    return images
```

def getAllImachesAndFavouriteList():

```
def getAllImagesAndFavouriteList(request):
    images = []
    favourite_list = []
   return images, favourite_list
```

Importamos la función *getAllImages* desde layers.services.services_nasa_images_gallery y la guardamos en la variable *images* de la función *getAllImagesAndFavouriteList*.

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

    return images, favourite_list
```

def home():

```
def home(request):
    images,favourite_list = []
    return render(request, 'home.html', {'images':
images,'favourite_list':, favourite_list})
```

En *views.py,* dentro de la función *home,* guardamos la lista de *getAllImagesAndFavouriteList* en las variables *images, favourite_list,* que eran listas vacías.

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

```

def search():

```
def search(request):
    search_msg = request.POST.get('query', '')
    images =
    favourite_list = []
    return render(request, 'home.html', {'images': images,
'favourite_list': favourite_list} )
```

Esta función le da marcha al buscador de imágenes, lo que se buscó es tomar el dato que guarda la función y condicionar, si el usuario no ingresa ningún dato, pero oprime el botón de buscar, se hará una búsqueda automática de space. Si el usuario realiza una búsqueda se le devolverá todas aquellas cards que tengan relación con la palabra buscada

```
# función utilizada en el buscador.
def search(request):
    images, favourite_list = getAllImagesAndFavouriteList(request)
    search_msg = request.POST.get('query', '')
    if not search_msg:
        # Si el usuario NO ingresa dato alguno y hace clic sobre el botón 'Buscar', debe filtrar por el valor predeterminado (space).
        images = getAllImages('space')
        return render(request, 'home.html', {'images': images, 'favourites': favourite_list})
    else:
        # Si el usuario ingresa algún dato (ej. sun -sol, en inglés-), al hacer clic se deben desplegar las imágenes filtradas relacionadas a dicho valor.
        images = getAllImages(search_msg)
        return render(request, 'home.html', {'images': images, 'favourites': favourite_list})
```

Además, se agrega la librería googletrans, en la condición falsa, para mejorar la experiencia de usuario y realizar las búsquedas deseadas indiferentemente del idioma en que se lo haga.

```
translator = Translator()
# Si el usuario ingresa algún dato (ej. sun -sol, en inglés-), al hacer clic se deben desplegar las imágenes filtradas relacionadas a dicho valor.
translated_search_msg = translator.translate(search_msg).text
images = getAllImages(translated_search_msg)
return render(request, 'home.html', {'images': images, 'favourites': favourite_list})

```

Dificultades
Desafíos y decisiones (dificultades encontradas y decisiones técnicas):

El primer reto que se nos presentó a la hora de abordar el proyecto fue el de adaptarnos a la nueva estructura de trabajo, que suponía organización grupal e investigación acerca de las nuevas herramientas a aplicar en la estructura de operaciones.
En segundo lugar, surgieron inconvenientes en la coordinación del grupo. El equipo se formaba originalmente por tres miembros, de los cuales, uno de ellos, por razones personales, no pudo continuar en la tarea.
El tercer inconveniente fue la incompatibilidad de horario por razones laborales. Si bien no representaba una gran dificultad, el tiempo con el que contábamos para trabajar de forma sincronizada se vio reducido.
Por último, se presentaron ciertos inconvenientes a niveles técnicos relacionados al equipamiento con el que contábamos ambos miembros del equipo. Si bien, pudimos resolver este inconveniente mediante el reemplazo, nos significó un extra en el consumo del tiempo con el que contábamos para la elaboración del trabajo.

Conclusión

El desarrollo del proyecto nos significó un gran desafío puesto que nos propuso una forma de trabajo diferente a la que veníamos acostumbrados. Esto implicó, mucha investigación y colaboración activa en equipo.

A través de este proceso, hemos adquirido la capacidad para desenvolvernos en un entorno de trabajo más dinámico. Experimentamos el salirnos de una estructura rígidamente delimitada (y cómoda), que resulta de las actividades de las guías de ejercitación de la materia.

Esta experiencia nos retó a extender las posibilidades en la aplicación del conocimiento que adquirimos durante la cursada y al mismo tiempo a tener una colaboración activa de grupo.