Trabajo Práctico – 1er cuatrimestre del 2024

El trabajo realizado en la creación de una página web, la cual permite la navegación y el acceso de imágenes ya proporcionadas por la NASA. Esta trata de una página que consiste en una galería de imágenes acompañada por un título y su descripción. Como extra se le agrego la opción de un buscador, para que los usuarios puedan obtener las imágenes que más les interesen, un registro para que puedan crear sus cuentas y queden registrados en la página, un inicio de sesión para los que ya tengan una cuenta registrada, y por último una sección de favoritos para que guarden las imágenes que mas les gustaron, esta función solo si cuentan con un usuario en la página.

Galería:

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

    #Obtiene un listado de imagenes desde transport.py
    json_collection=transport.getAllImages(input)

    # recorre el listado de objetos JSON, lo transforma en una NASACard y lo agrega en el listado de images.
    images = []

    for objetos in json_collection: #Recorro la lista json_collection.
        nasa_card=mapper.fromRequestIntoNASACard(objetos)
        images.append(nasa_card) #Agrega los objetos (pasados a nasacard) a la lista imagenes.
    return images
```

Para la creación de la galería de imágenes primero implemente una comunicación con la API de la NASA para obtener información sobre las imágenes, como segundo paso fue la recolección de la mayor cantidad de las imágenes de la página de la NASA, para luego, como tercer paso, mostrar en la galería la imagen con su título y la descripción.

```python
def getAllImagesAndFavouriteList(request):

    images=services_nasa_image_gallery.getAllImages() #lista de toda las imagenes de la API

    favourite_list=services_nasa_image_gallery.getAllFavouritesByUser(request) #listado de los fav del usuario

    return images, favourite_list

# función principal de la galería.

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*.

    images,favourite_list=getAllImagesAndFavouriteList(request) #retorno de las dos listas de la funcion anterior
    return render(request, 'home.html', {'images': images, 'favourite_list': favourite_list} )
```

Por último, para que las imágenes se vean en la página web, muestro la información recolectada de los pasos anteriores.

Buscador:

```python
# función utilizada en el buscador.
def search(request):
    images=[]
    favourite_list=[]
    search_msg = request.POST.get('query', '')
    if not search_msg: #Si no pone nada en el buscador
        images=services_nasa_image_gallery.getAllImages("space") #busca "space" por default
    else:
        images=services_nasa_image_gallery.getAllImages(search_msg) #busca lo que introdujo en search_msg


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

Para el buscador, como se puede ver si el usuario no escribe nada en el buscador, la página por defecto buscara la palabra “space” (espacio en inglés), de lo contrario que la persona escriba algo, lo buscara esto debe buscarse si o si con palabras en inglés, debido a que los títulos y descripciones están en ese idioma.

La complicación que tuve y no pude resolver fue que muestre los favoritos cuando busco una palabra, aparece la imagen, pero no con el corazón tildado, pero si voy a mis favoritos guardados si aparece la imagen.

Registro:

```python
from django import forms

from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User

#Modelo de creacion del formulario.
class CustomUserCreationForm(UserCreationForm):
    class Meta(UserCreationForm.Meta):
        model = User
        fields =  ('email', 'first_name', 'last_name','username', 'password1', 'password2')
```

Lo primero que hice fue crear un modelo del formulario, anteriormente lo estaba creando de cero, pero indagando un poco más encontré uno genérico. Esta muestra las casillas a rellenar en el registro.

También se agregó la opción de registro que te mande directo al formulario.

```
{% extends 'header.html' %} {% block content %}
<div>
    <h1>Registro</h1>
    {% if messages %}
        {% for message in messages %}
            <p style="color: red">{{ message }}</p>
        {% endfor %}
    {% endif %}
    <form action="{% url 'registro' %}" method="POST">
        {% csrf_token %}
        {{ form }}
        <input type="submit" value="Submit">
    </form>
</div>
{% endblock %}
```

```python
def registro(request):
    form = CustomUserCreationForm()
    if request.method == 'POST':
        form = CustomUserCreationForm(request.POST)
        if form.is_valid():
            form.save()
            subject = 'Registro completado.'
            message = 'Bienvenido ' + form.cleaned_data.get('first_name') + ',\nEste es tu usuario y contraseña: ' +
            form.cleaned_data.get('username') + ' ' + form.cleaned_data.get('password1')
            recipient = form.cleaned_data.get('email')
            send_mail(subject,message, settings.EMAIL_HOST_USER, [recipient], fail_silently=False)
            messages.success(request, 'Success!')
            return redirect('registro')
    return render(request, 'registro.html', {'form': form})
```

Por último, en el registro se implementó una función, que al finalizar el formulario mande una notificación al Gmail ingresado por el usuario, indicándole un registro exitoso, y una bienvenida con su nombre de pila más su usuario y contraseña por si se le olvida.

En cuanto a complicaciones, Cuando agrego el botón de ‘Registro’, queda por debajo del inicio de sesión, investigando un poco puede ponerlo al costado del inicio de sesión, pero cuando después quería iniciar sesión el registro no desaparecía, seguía estando ahí, aunque tenia una cuenta ya iniciada, entonces decidí dejarlo como estaba.

Inicio de sesión:

```python
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    , (module) accounts
    'accounts',
]
```

Como principal instale el formato de inicio de sesión que Django te brinda.

```python
#Extrae informacion de rutas
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('nasa_image_gallery.urls')),
    path('accounts/', include('django.contrib.auth.urls')),
```

El ultimo código extrae la información descargada del formato de Django.

```python
urlpatterns = [
    path('', views.index_page, name='index-page'),
    path('login/', views.index_page, name='login'),
    path('home/', views.home, name='home'),
    path('buscar/', views.search, name='buscar'),
    path('registro/', views.registro, name='registro'), #url faltante para el registro.
```

```
<a class="nav-link" href="{% url 'logout' %}">Salir</a> {% else %}
<a class="nav-link" href="{% url 'login' %}">Iniciar sesión</a>
```

Por último, inicializando las urls con ‘logout’(salir) y ‘login’ (iniciar sesión), ahora cuando clickeemos en iniciar sesión nos mandara la pagina a rellenar con nuestros datos si es que tenemos una cuenta ya registrada.

Favoritos:

Lógica:

```python
# añadir favoritos (usado desde el template 'home.html')
def saveFavourite(request):
    fav = mapper.fromTemplateIntoNASACard(request) # transformamos un request del template en una NASACard.
    fav.user = get_user(request) # le asignamos el usuario correspondiente.

    return repositories.saveFavourite(fav) # lo guardamos en la base.
```

En un principio cuando el usuario decide guardar una foto en favoritos, el programa transforma la petición, para luego guardarlo en la base de datos de la página. Cabe aclarar que esta función solo se permitirá su uso, si el usuario tiene una cuenta iniciada, por eso con la función de abajo aclaramos que debemos asignarle un usuario.

```python
# usados en el template 'favourites.html'
def getAllFavouritesByUser(request):
    if not request.user.is_authenticated: #si no loguea
        return [] #no devuelve favoritos (debido a que no tiene cuenta)
    else:

        user = get_user(request)

        favourite_list = [] # buscamos desde el repositorio TODOS los favoritos del usuario (variable 'user').
        favourite_list=repositories.getAllFavouritesByUser(user)
        mapped_favourites = []

        for favourite in favourite_list:
            nasa_card = mapper.fromRepositoryIntoNASACard(favourite) # transformamos cada favorito en una NASACard, y
            lo almacenamos en nasa_card.
            mapped_favourites.append(nasa_card)

        return mapped_favourites
```

Si el usuario inicio sesión, busca los favoritos marcados, el programa lo recorre y luego vuelve a transformarlos y le agrega una identificación, que nos servirá luego para poder eliminar la foto de favoritos si el usuario desee.

```python
def deleteFavourite(request):
    favId = request.POST.get('id')
    return repositories.deleteFavourite(favId) # borramos un favorito por su ID.
```

Muestra:

```python
@login_required
def getAllFavouritesByUser(request):
    favourite_list = []
    favourite_list=services_nasa_image_gallery.getAllFavouritesByUser(request) #lista de los favoritos del user
    return render(request, 'favourites.html', {'favourite_list': favourite_list})
```

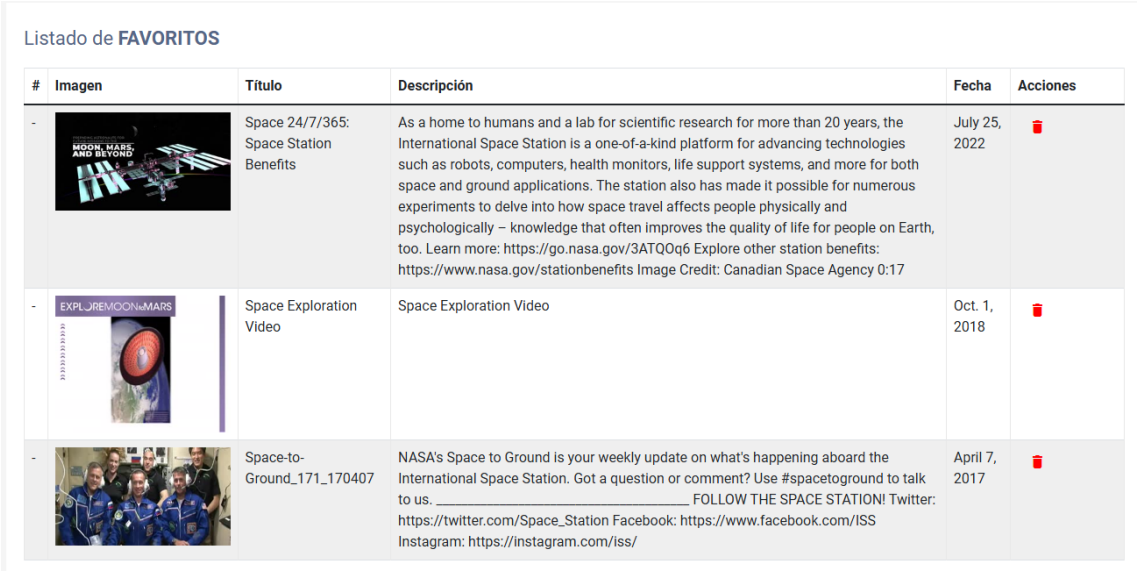
Esta función muestra la lista de favoritos del usuario para que la vea.

```python
@login_required
def saveFavourite(request):
    #Guarda el nuevo favorito en la lista
    services_nasa_image_gallery.saveFavourite(request)
    #Obtener la lista de favoritos actuales del usuario
    images,favourite_list=getAllImagesAndFavouriteList(request)
    #Renderiza la pagina home
    return render(request, 'home.html', {'images': images, 'favourite_list': favourite_list} )
```

Lo que cumple esta funcion es que, cuando la persona cliquea en añadir a favoritos de una imagen, esta procesa y luego vuelve a mostrar la pagina de la galeria.

```python
@login_required
def deleteFavourite(request):
    #Eliminar el favorito de la lista
    services_nasa_image_gallery.deleteFavourite(request)
    #Obtener la lista de favoritos actualizada del ususario
    favourite_list=services_nasa_image_gallery.getAllFavouritesByUser(request)
    return render(request, 'favourites.html', {'favourite_list': favourite_list})
```

Por último el deleteFavourite procesa la eliminacion de una imagen que esta en favoritos. Una vez que la elimina actualiza dicha página.



Al finalizar si guardo imágenes en favoritos se observará de esta manera, la imagen, título, descripción, fecha que se tomo la foto, y la acción de poder eliminar la foto.