Trabajo practico de introduccion a la programación, primer semestre del 2024

Resumen:
Este trabajo práctico tiene como objetivo desarrollar una aplicación web fullstack utilizando el framework Django. La aplicación permite consultar y visualizar imágenes de la API pública de la NASA.

Integrantes:
- Franco Muñoz (franco4409@gmail.com)
- Gaston saldaño (gastonsaldao1234@gmail.com)

1. Introducción

Este trabajo práctico de Introducción a la Programación del primer semestre del 2024 consiste en desarrollar una aplicación web fullstack que permite a los usuarios consultar y visualizar imágenes obtenidas de la API pública de la NASA. La aplicación debe mostrar las imágenes en tarjetas (cards) que incluyan una imagen, un título y una descripción. Además, adicionalmente se puede implementar un buscador para filtrar las imágenes y un sistema de autenticación para que los usuarios puedan guardar imágenes como favoritas y consultarlas posteriormente.

2. Desarrollo

2.1 Descripción general

El proyecto se organiza en varias capas siguiendo una arquitectura multicapa para mantener la separación de responsabilidades. Las capas incluyen DAO, Services y Transport. La capa de transporte se encarga de consumir la API de la NASA, mientras que la capa de servicios contiene la lógica de negocio y la capa DAO maneja las interacciones con la base de datos.

2.2 Funcionalidades principales

Función**:** `getAllImagesAndFavouriteList(request)`

Esta función desempeña un papel fundamental en la obtención de datos para nuestra página de galería de imágenes de la NASA. A través de esta función, se realizan los siguientes pasos:

Inicialización de variables: La función comienza inicializando dos listas vacías, `images` y `favourite_list`, que se utilizarán para almacenar todas las imágenes disponibles y la lista de imágenes favoritas respectivamente.

Obtención de imágenes: Utilizando el módulo `services_nasa_image_gallery`, se invoca la función `getAllImages()`. Esta función definida en otro archivo de servicios, se encarga de comunicarse con la API de la NASA para recuperar todas las imágenes disponibles en la galería.

Retorno de resultados: Finalmente, la función retorna dos valores: `images`, que contiene todas las imágenes recuperadas de la galería, y `favourite_list`, una lista inicialmente vacía que permitirá al usuario agregar imágenes a su lista de favoritos durante la interacción con la página.

Función **`home(request)`**

cumple la función de renderizar la página principal de nuestra aplicación.

Obtención de datos: comienza llamando a la función `getAllImagesAndFavouriteList(request)`. Esta función devuelve dos valores, `images` y `favourite_list`, que representan todas las imágenes disponibles y la lista de imágenes favoritas, respectivamente.

Renderizado de la plantilla: Utilizando la función `render` de Django, se carga la plantilla `home.html`. Esta plantilla es la página principal de nuestra aplicación web y se utiliza para mostrar las imágenes de la galería y las imágenes marcadas como favoritas.

Pasaje de datos a la plantilla: Se pasan dos conjuntos de datos a la plantilla:

`images`: Contiene todas las imágenes recuperadas de la galería mediante la función `getAllImagesAndFavouriteList(request)`.

`favourite_list`: Una lista que inicialmente está vacía, pero que puede ser utilizada para almacenar las imágenes que el usuario marca como favoritas durante su interacción con la página.

Función `getAllImages(input=None)`

Esta función desempeña un papel central en la obtención y procesamiento de imágenes para nuestra aplicación de galería de la NASA:

Inicialización de variables: La función inicializa una lista vacía `json_collection` para almacenar los datos en formato JSON.

Obtención de datos: Utilizando una función `transport.getAllImages(input)`, se recupera una colección de datos en formato JSON. Esta función `getAllImages` maneja la comunicación con la API de la NASA para obtener todas las imágenes disponibles.

Procesamiento de datos: Se itera sobre cada objeto JSON en `json_collection`. Para cada objeto, se utiliza una función `mapper.fromRequestIntoNASACard(json)` para mapear y convertir los datos del JSON en un objeto `NASACard`, que contenga información específica de cada imagen, como título, descripción, fecha, y enlaces relacionados.

Construcción de lista de imágenes: Los objetos `NASACard` resultantes se agregan a la lista `images`, que luego se devuelve como resultado de la función.

3. Conclusiones

El trabajo permitió aplicar conocimientos de desarrollo web fullstack utilizando Django, consumiendo APIs públicas y gestionando autenticación de usuarios. Además, se destacó la importancia de la organización del código en una arquitectura multicapa para mantener una clara separación de responsabilidades.