INTRODUCCIÓN

A LA

PROGRAMACIÓN

Trabajo práctico: Rick y Morty

Segundo semestre de 2024

Integrantes: Monzón Yésica
Gonzales Ignacio

Este trabajo práctico consistió en adaptar un repositorio de GitHub para que interactúe con la API de Rick y Morti y presente los resultados de búsqueda de los usuarios en un navegador. Para lograrlo, utilizamos un entorno de desarrollo que combinaba Python, Git, Visual Studio Code y Django.

Inicialmente, nos encontramos con algunos inconvenientes a la hora de descargar los programas por errores de los comandos ingresado en la terminal. El siguiente inconveniente fue la conexión al servidor, hasta que comprendimos que cada vez que iniciábamos la computadora había que conectar el proyecto al servidor.
Luego nos enfrentamos al desafío de comprender la estructura del código, dividida en múltiples módulos. Para superar esta dificultad, creamos diagramas que visualizaban el flujo de la aplicación y los relacionamos con las funciones a implementar. Esta colaboración entre el equipo fue fundamental para avanzar en el proyecto.

En la etapa de desarrollo, nos centramos en completar las funciones faltantes de ciertos archivos. Estas funciones eran responsables de la lógica de la aplicación y se encargaban de interactuar con la API de Rick y Morti y presentar los resultados al usuario.

Primero analizamos el archivo views.py, donde nos encontramos con la función home incompleta. Por lo que procedimos a completarla según la lógica que utiliza Python en Django para poder visualizar las imágenes de las cards. Tuvimos que utilizar la función services.getAllImages(). Donde services.py centraliza la lógica de negocio y las interacciones con otras capas del sistema, haciendo que el código sea más modular y fácil de mantener.

```
def home(request):
    images = services.getAllImages()
    favourite_list = []

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

Luego nos encontramos con la función search, donde la completamos para que el buscador funcione como filtro. Tuvimos que agregar images=services.getAllImages(search_msg) para que con cada búsqueda solo retorne las cards con las cadenas pedidas.

```
     if (search_msg != ''):
        print(search_msg)
        images = services.getAllImages(search_msg)
        favourite_list = []
```

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

En el archivo Services.py tuvimos que importar a transport.py, que se utiliza como medio de transporte para que se comuniquen con las otras interfaces. Luego de esto, pudimos completar la función getAllImages. Le añadimos un for para que recorra cada item y lo vaya añadiendo en las images.

```
def getAllImages(input=None):
    # obtiene un listado de datos "crudos" desde la API, usando a
transport.py.
    json_collection = transport.getAllImages(input)

    # recorre cada dato crudo de la colección anterior, lo convierte en una
Card y lo agrega a images.
    images = []
    for item in json_collection:
        images.append(translator.fromRequestIntoCard(item))

    return images
```

En una tercera instancia analizamos el archivo home.html para poder brindarle el color deseado a las cards. Primero tuvimos que colocar correctamente los condicionales para que verifiquen el img.status. Luego tuvimos que inspeccionar la página y localizar cual era la class corrrespondiente al contorno de las cards y así poder modificar los colores con respecto a su status. En style.css, llamamos a la clase .card .Alive, .card.Dead y .card.unknown para colocarle los colores solicitados en el trabajo.

```
48  @login_required
49  def exit(request):
50      # Cierra la sesión del usuario
51      logout(request)
52
53      return redirect('login')  #cuando se cierra secion, se dirige hacia el login.
```

Por otro lado, se completo la funcion exite. La función logout (request) elimina todos los datos de la sesión actual, desconectando al usuario.

De esta manera, cada ves que el usuario abra sesion, con usuario: admin y contraseña: admin, puede cerrar sesion y dirigirlo a la pagina de login nuevamente.