Cristina Margaret Calvario Serrano

Módulo 8. Optimización de código con herramientas basadas en IA.

Caso práctico 8. Crear los casos de uso de prueba funcional para las REST API del backend.

Este script monitorea el tiempo de respuesta de una URL (página web o API) y alerta si supera un umbral definido. Es útil para verificar el rendimiento de un sitio web o servicio en tiempo real.

|  |
| --- |
| # Importamos las librerías necesarias  import requests  # Para hacer solicitudes HTTP a una URL  import time      # Para medir el tiempo y pausar el monitoreo  *def* get\_response\_time(*url*):      """      Realiza una solicitud HTTP GET a la URL proporcionada y calcula      cuánto tiempo tardó en obtener la respuesta.      """      # requests.get(url) envía una solicitud GET, HTTP GET a la URL      response = requests.get(*url*)      # Obtenemos el tiempo de respuesta en segundos desde que se envió hasta que se recibió      # .elapsed es un objeto timedelta, y .total\_seconds() lo convierte a segundos flotantes      response\_time = response.elapsed.total\_seconds()      # Retornamos el tiempo de respuesta      return response\_time  *def* monitor\_performance(*url*, *threshold*):      """      Verifica el rendimiento de una URL comparando el tiempo de respuesta      con un umbral definido por el usuario.      """      # Llama a la función que obtiene el tiempo de respuesta actual      response\_time = get\_response\_time(*url*)      # Compara el tiempo de respuesta con el umbral definido      if response\_time > *threshold*:          # Si se excede el umbral, se imprime una alerta con la información relevante          print(              "ALERTA: El tiempo de respuesta de la URL '{}' es de {*:.2f*} segundos. "              "Esto supera el umbral de {*:.2f*} segundos.".format(*url*, response\_time, *threshold*)          )      else:          # Si está dentro del rango normal, se muestra un mensaje informativo          print(              "OK: El tiempo de respuesta de '{}' es {*:.2f*} segundos, dentro del umbral.".format(*url*, response\_time)          )  if \_\_name\_\_ == "\_\_main\_\_":      """      Bloque principal del programa: define la URL a monitorear y el umbral      y ejecuta el monitoreo en un bucle continuo.      """      # Definimos la URL de la aplicación web a monitorear      #url = "https://www.google.com"      #url = "https://www.example.com"      url = "http://127.0.0.1:5000/users"      #url = "http://127.0.0.1:5000"      #url = "http://localhost/phpmyadmin"        # Definimos el umbral de tiempo de respuesta permitido (en segundos)      threshold = 2.0      # Inicia un bucle infinito que monitorea la URL cada segundo      while True:          # Llama a la función que evalúa el rendimiento          monitor\_performance(url, threshold)          # Espera 1 segundo antes de volver a comprobar          time.sleep(1.0) |

Funciones principales

1. get\_response\_time(url)

* Realiza una petición HTTP GET a la url y mide cuánto tarda en recibir la respuesta.
* Retorna el tiempo en segundos (ej. 1.25).

1. monitor\_performance(url, threshold)

* Compara el tiempo de respuesta con el umbral (threshold).
* Si el tiempo es mayor al umbral, imprime una alerta.
* Si es menor, imprime un mensaje de éxito.

1. Bucle principal (while True)

* Ejecuta el monitoreo indefinidamente cada 1 segundo.

En mi archivo monitoreoapp.py

* Agregamos la URL a monitorear

Texto

El contenido generado por IA puede ser incorrecto.

* Se ajusta el umbral y el intervalo de monitoreo.

Texto

El contenido generado por IA puede ser incorrecto.

Probaremos con el proyecto de app.py, es una app de usuarios, lo ejecutamos:

Texto

El contenido generado por IA puede ser incorrecto.

Después, ejecutamos el monitoreo, nos da como salida lo siguiente:

Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Nos da el tiempo en tarda em responder la URL

Texto

El contenido generado por IA puede ser incorrecto.