Práctica Integrada de 50 Minutos: Automatización y Manejo de Datos en Pyth

Objetivo

El objetivo de esta práctica es que los estudiantes desarrollen un sistema de monitoreo automatizado utilizando Python, bases de datos, archivos de texto, módulos y manejo de excepciones.

La práctica está diseñada para que puedan utilizar inteligencia artificial (IA) como apoyo, pero deberán estructurar correctamente el proyecto y comprender cómo integrarlo todo.

Requisitos Previos

- Conocimiento básico de Python.
- Familiaridad con bases de datos SQLite.
- Experiencia en manejo de archivos y excepciones.
- Uso de librerías estándar de Python como os, sqlite3, platform, datetime, random.

Parte 1: Diseño del Sistema de Monitoreo

Se debe desarrollar un programa que:

1. Obtenga información del sistema (Nombre del SO, versión, procesador, uso de CPU y memoria).
2. Almacene los datos en una base de datos SQLite.
3. Genere un archivo de registro con los resultados.
4. Implemente manejo de excepciones para evitar errores en la ejecución.
5. Use módulos y funciones propias para mejorar la organización del código.
6. Ejecute tareas automatizadas cada cierto tiempo.

Parte 2: Creación de la Base de Datos

- Crear una base de datos llamada monitoreo.db.
- Crear una tabla llamada registros con los siguientes campos:
  - id (clave primaria, autoincrementable).
  - fecha_hora (almacena la fecha y hora del registro).

  - cpu_uso (porcentaje de uso de CPU).
  - memoria_total (memoria total en MB).
  - memoria_disponible (memoria disponible en MB).
  - procesador (información del procesador).
  - sistema_operativo (nombre y versión del SO).

Parte 3: Captura de Información del Sistema
- Obtener el nombre del sistema operativo, versión y procesador usando platform.
- Obtener el uso de CPU y memoria utilizando comandos del sistema operativo (os.system).
- Registrar estos valores en la base de datos y en un archivo de texto log_monitoreo.txt.

Parte 4: Escritura y Lectura de Archivos
- El sistema debe escribir cada registro en un archivo de texto con el siguiente formato:
  [Fecha y Hora] - CPU: 40% - Memoria Disponible: 2048 MB / 8192 MB - Procesador: Intel i7
- El programa debe permitir leer los registros almacenados en el archivo y mostrarlos en consola.

Parte 5: Modularización del Código
- Separar el código en funciones:
  - obtener_info_sistema(): Devuelve información del sistema.
  - obtener_uso_cpu_memoria(): Obtiene el uso de CPU y memoria.
  - guardar_en_db(): Almacena la información en SQLite.
  - guardar_en_archivo(): Guarda los registros en un archivo de texto.
  - monitoreo_automatico(): Ejecuta el monitoreo en intervalos de tiempo.

Parte 6: Manejo de Excepciones
- Capturar errores cuando:
  - No se puede acceder a la base de datos.
  - No se puede leer o escribir en el archivo.
  - Falla la ejecución de un comando del sistema operativo.
- Implementar try-except en todas las funciones para evitar que el programa se detenga

abruptamente.

Parte 7: Ejecución Periódica

- Utilizar time.sleep(10) para que el monitoreo se ejecute cada 10 segundos por un total de 5 ciclos.

- Cada ejecución debe capturar la información del sistema y almacenarla.

- Permitir que el usuario interrumpa el monitoreo con Ctrl + C, manejando la excepción KeyboardInterrupt.

Entrega Final

Los estudiantes deben entregar un script en Python que cumpla con los requisitos especificados, incluyendo:

1. Base de datos monitoreo.db con registros.

2. Archivo log_monitoreo.txt con los registros.

3. Un código limpio, modularizado y con excepciones manejadas correctamente.

Extra Opcional (Para estudiantes avanzados)

- Implementar una interfaz gráfica en tkinter para visualizar los registros en tiempo real.

- Generar un reporte en formato .csv con los datos almacenados en la base de datos.

- Enviar alertas por correo electrónico si el uso de CPU supera el 80%.

Resumen

Esta práctica integra varios conceptos fundamentales de Python en un proyecto realista y desafiante.

Se permite el uso de IA para ayudar en la resolución de problemas, pero los estudiantes deben comprender y estructurar correctamente el programa.

Se espera que al finalizar la práctica, los participantes tengan una mejor comprensión sobre la automatización de tareas en Python, bases de datos, archivos y manejo de errores, habilidades esenciales para el desarrollo profesional en el área de programación y administración de sistemas.