TP Thème 4 - Métriques système

Contexte pédagogique

Dans le cadre de ce module, vous découvrirez comment utiliser des bibliothèques Python spécialisées pour la surveillance système et réseau. L'objectif est de les initier à l'utilisation de la bibliothèque **psutil**, qui permet d'accéder à de nombreuses métriques système : CPU, mémoire, disque, réseau, etc.

Les systèmes modernes, qu'ils soient serveurs ou postes de travail, nécessitent une **surveillance régulière de leurs ressources** pour garantir leurs performances. Grâce à **psutil**, Python offre un outil puissant, multiplateforme et relativement simple à utiliser pour cette tâche.

Objectifs pédagogiques

À l'issue de ce TP, vous saurez :

- Utiliser **psutil** pour extraire des informations système en temps réel.
- Afficher dynamiquement ces informations dans un tableau de bord textuel.
- Organiser et formater les données de manière lisible pour les utilisateurs.
- Structurer un script Python en suivant de bonnes pratiques.

Consignes

Partie 1 : Installation et découverte

1. Créez un environnement virtuel Python (optionnel mais recommandé).
2. Installez la bibliothèque **psutil** :
3. Indiquez quelle est la fonctionnalité des chacune des fonctions listées ci-dessous et testez-les :
    - psutil.cpu_percent()
    - psutil.virtual_memory()
    - psutil.disk_usage('/')
    - psutil.net_io_counters()

Partie 2 : Création du tableau de bord

Implémentez un script Python qui :

1. **Affiche dynamiquement les informations système** toutes les 5 secondes.
2. **Efface l'écran à chaque cycle** pour un affichage propre.

3. Affiche les sections suivantes :
    - Utilisation CPU (par cœur et totale)
    - Mémoire RAM (totale, utilisée, libre)
    - Utilisation disque par partition
    - Activité réseau (octets et paquets envoyés/reçus)
    - Statistiques réseau **par interface**
4. Ajoutez une option pour que l'utilisateur puisse **quitter le programme proprement** (ex: tapez "quit").

Partie 3 : Organisation du code

Structurez le code selon les recommandations suivantes :

- Encapsulation du tableau de bord dans une fonction **display_dashboard()**.
- Utilisation de boucles et de dictionnaires pour traiter plusieurs partitions et interfaces réseau.
- Utilisation de **os.system('clear')** ou **cls** selon l'OS pour effacer l'écran.

Bonus (Optionnels)

Bonus 1 – Ajout de la température CPU

Utilisez la fonction **psutil.sensors_temperatures()** si elle est disponible sur le système :

python

Bonus 2 – Barre graphique ASCII pour le CPU

Affichez des barres de progression textuelles pour l'utilisation CPU :

Bonus 3 – Export des données dans un fichier log

Écrivez les données dans un fichier .**csv** ou **.txt** à chaque itération.