Rapport Projet Python

Outil de Surveillance et d'Analyse des Données IoT

Anne universitaire 2024/2025

Table des matières

I. Introduction

Ce document présente une vue d'ensemble de l'application simulée de surveillance IoT, y compris l'utilisation de threads pour simuler le fonctionnement des objets connectés, et l'intégration potentielle avec des outils comme Zabbix. L'application permet de générer des données, de les consommer en temps réel, et de visualiser les tendances via une interface utilisateur construite avec Streamlit.

II. Fonctionnalités Principales

- **Simulation IoT :** Utilisation de threads Python pour simuler plusieurs dispositifs IoT générant des données en temps réel.
- **Consommation des Données :** Un consommateur traite les données à partir d'une file d'attente.
- **Export des Données :** Les données sont sauvegardées dans des fichiers CSV, puis affichées dans une interface graphique.
- **Visualisation :** Utilisation de graphiques pour afficher les tendances des mesures collectées.
- **Rapports PDF :** Génération de rapports PDF contenant les statistiques et les graphiques des données collectées.

III. Simulations et Outils de Collecte

1. Threads pour Objets IoT :
   - Chaque objet IoT est modélisé par une classe IoTDevice qui hérite de threading.Thread.
   - Les dispositifs génèrent des données aléatoires toutes les 10 secondes (température, humidité, niveau de batterie, etc.).
2. Zabbix comme Exemple d'Intégration Future :
   - Si des objets réels étaient disponibles, Zabbix pourrait être utilisé pour surveiller les dispositifs.
   - Les données collectées par Zabbix seraient intégrées dans l'application via des appels API REST.

IV. Architecture de l'Application

- **Backend :** Python pour la simulation et la gestion des données.
- **Frontend :** Streamlit pour l'interface utilisateur.

- **Stockage des Données :** Fichiers CSV pour l'archivage local.
- **Visualisation :** Graphiques interactifs avec Plotly.

V. Guide d'Exécution

1. Prérequis :

- Python 3.8 ou supérieur.
- Modules requis : threading, pandas, plotly, streamlit, csv, queue, datetime, random, os, fpdf.

2. Installation des Dépendances :

```
pip install pandas plotly streamlit fpdf
```

3. Exécution du Code :

- Simulation **:** `python iot_listenner.py`
- Interface Utilisateur : `streamlit run web_app.py`

VI. Conclusion

Cette application démontre l'utilisation des threads pour simuler un système IoT. Elle est extensible pour intégrer des dispositifs réels via des outils comme Zabbix et des API REST. Ce rapport inclut un guide d'exécution pour faciliter la reproduction de l'environnement. Pour plus de détails sur l'architecture et les composants, consultez les fichiers sources inclus.