Projet (EF) du Cours de Programmation Avancée en Python

Niveau : Master 1 (IT) - IABD
Crédits : 3
Semestre : 2

26 juin 2025

Projet Individuel d'Examen
Programmation Avancée en Python

Master 1 IABD
Collège de Paris Supérieur – Campus de Lomé

Projet Individuel d'Examen
Programmation Avancée en Python

Master 1 IABD
Collège de Paris Supérieur – Campus de Lomé

Titre du projet

Conception d'un système de surveillance intelligente basé sur des flux de données simulés

Objectif pédagogique

Ce projet vise à évaluer la capacité de l'étudiant à mobiliser les compétences avancées en Python dans le cadre du développement d'une application complète, modulaire, et orientée vers l'analyse de données en temps quasi-réel. Il met en œuvre :

- La programmation orientée objet (POO)
- Le traitement de fichiers et flux de données
- La programmation événementielle ou asynchrone
- L'utilisation de bibliothèques Python avancées (`asyncio`, `pandas`, `logging`, etc.)
- La génération automatique de rapports

Contexte

Une entreprise de cybersécurité souhaite un prototype capable d'analyser des événements issus de fichiers logs simulés pour détecter des comportements anormaux et générer des rapports d'alerte. Vous êtes missionné pour concevoir ce système intelligent.

Travail demandé

1. **Lecture asynchrone de logs**
   Lire le fichier `events.log` ligne par ligne avec un délai simulé (2 secondes entre chaque ligne). Utiliser les modules `asyncio` ou `threading`. Chaque ligne représente un événement au format JSON.

2. **Modélisation orientée objet**
   Implémenter des classes `Event`, `EventAnalyzer`, `EventLogger` :

- Encapsulation du traitement des événements
- Calcul de statistiques sur les types et priorités
- Journalisation des traitements

3. **Détection d'anomalies**
   Définir une alerte comme toute suite de trois événements *critiques* dans un intervalle de 30 secondes. Enregistrer les alertes détectées dans le fichier `alerts.json`.

4. **Génération automatique de rapport**
   Générer un rapport de fin de traitement (format PDF ou texte) contenant :
   - Nombre total d'événements
   - Nombre d'événements critiques
   - Nombre d'alertes
   - Horodatage des alertes

   Bibliothèques suggérées : `reportlab`, `fpdf`, ou simple `write()`.

5. **Interface CLI ou menu interactif (1 point)**
   Proposer un petit menu textuel permettant :
   - de lancer le traitement
   - d'afficher les alertes
   - de générer le rapport
   - de quitter le programme

6. **Visualisation statistique**
   Générer une figure (ex : histogramme) illustrant les fréquences d'événements selon leur type ou niveau de priorité. Bibliothèques suggérées : `matplotlib`, `seaborn`.

Livrables attendus

Date limite : 06/07/2025 à 23h59. Chaque étudiant soumettra au délégué de classe une archive nommée :
`Nom-Prenoms_ProjetFinalPythonAvance.zip`, contenant :

- Le code source organisé et documenté
- Un fichier `README.md` explicatif
- Un fichier `requirements.txt` listant les dépendances
- Le rapport généré automatiquement
- **Une courte vidéo** ($\leq$ 3 minutes) pour illustrer les résultats-clés

Barème de notation (/10)

Critère	Points
Lecture asynchrone et gestion des flux	2
Structure objet et modularité du code	2
Détection correcte des anomalies	2
Rapport clair et complet	2
Interface CLI fonctionnelle	2
Bonus (graphiques, Git, tests unitaires, etc.)	+2

Recommandations

— Évitez le code redondant, préférez des fonctions bien nommées.
— Commentez votre code et documentez les classes et méthodes.
— N'oubliez pas d'utiliser un fichier `main.py` comme point d'entrée.
— L'usage de GitHub pour versionner le projet est recommandé.