**PROMPT**

Gera uma API em Python que recebe como input um ficheiro de código e a API faz o code review desse código. Para além do code review deve também auditar issues de segurança a vulnerabilidades. A API deve retornar um ficheiro ASCII com a lista de erros, e vulnerabilidades detetadas. O outuput deve ter o seguinte formato, sendo um erro/vulnerabilidade por linha:

<#linha de código>|<Criticidade>|<Descrição do Erro>|<Ação necessária para resolução>

o nível de criticidade deve pertencer à seguinte lista:

CRI: Crítico - Se vulnerabilidade de segurança Crítica ou erro de código que implique que a aplicação termine ou bloqueie

MAI : MAjor - Se provocar redução de consumo de recursos ou alguma vulnerabilidade MAJOR

MEN - Minor - Se vulnerabilidade de Segurança Minor ou códiog que afete ligeiramente a *perfomance* do código

WRN: Warning -Se o for somente um aviso sobre utilização de funções deprecated ou em fase de descontinuação ou propostas de melhoria do código.

**RESPOSTA**

from flask import Flask, request, send\_file

import ast

app = Flask(\_\_name\_\_)

def analisar\_codigo(codigo):

resultado = []

linhas = codigo.split('\n')

for numero\_linha, linha in enumerate(linhas, start=1):

if 'eval(' in linha:

resultado.append(f"{numero\_linha}|CRI|Uso da função eval pode ser inseguro|Substituir eval por alternativas mais seguras")

elif 'exec(' in linha:

resultado.append(f"{numero\_linha}|CRI|Uso da função exec pode ser inseguro|Evitar o uso de exec para prevenir vulnerabilidades")

elif 'import \*' in linha:

resultado.append(f"{numero\_linha}|MAI|Importação global pode levar a conflitos|Importar apenas os módulos necessários")

elif 'print(' in linha:

resultado.append(f"{numero\_linha}|WRN|Uso da função print encontrado|Utilizar logging em vez de print para produção")

# Adicionar mais regras de análise conforme necessário

return resultado

@app.route('/analisar', methods=['POST'])

def analisar():

ficheiro = request.files['ficheiro']

codigo = ficheiro.read().decode('utf-8')

resultado = analisar\_codigo(codigo)

with open('resultado.txt', 'w', encoding='utf-8') as f:

for linha in resultado:

f.write(linha + '\n')

return send\_file('resultado.txt', as\_attachment=True)

if \_\_name\_\_ == '\_\_main\_\_':

app.run()