covenant
TECHNOLOGY
Crowe Macro Group

Desafio Técnico – Automação RPA (Google Maps)

Descrição do Desafio

Desenvolver uma automação simples em **Python (RPA)** que acesse o **Google Maps** e colete informações sobre estabelecimentos de diferentes tipos:

- **Academias**
- **Restaurantes**
- **Sorveterias**

A automação deverá realizar as buscas no Google Maps e gerar um **arquivo JSON** contendo os seguintes dados:

- **Nome do estabelecimento**
- **Tipo** (restaurante, academia ou sorveteria)
- **Nota do estabelecimento**
- **Quantidade de avaliações**
- **Endereço completo**

Requisitos Técnicos

1. **Estrutura do Projeto**
    - O projeto deverá conter:
        - Um **arquivo de configuração** separado (por exemplo, config.py) com **constantes e caminhos** utilizados na automação.
        - Um ou mais **arquivos de biblioteca/utilitários** (por exemplo, utils.py) contendo **funções auxiliares** para o desenvolvimento.
        - Um **arquivo principal** (main.py) responsável por executar a automação completa.
        - Um **arquivo de log** (automation.log) registrando toda a execução.
        - Um **arquivo JSON** (resultados.json) com os dados coletados.

- Uma **planilha Excel (.xlsx)** gerada a partir do JSON, com:
  - Colunas nomeadas.
  - Espaçamento adequado entre as colunas.
  - Dados organizados de forma legível.

2. **Logs**
   - O script deverá registrar **eventos com diferentes níveis de log**
     - DEBUG – para informações detalhadas de depuração.
     - INFO – para eventos normais da execução.
     - WARNING – para avisos de possíveis problemas.
     - ERROR – para erros durante a execução.
     - CRITICAL – para falhas graves que interrompam a automação.
   - Cada entrada do log deve conter **data e hora da ocorrência**.
3. **Tratamento de Erros**
   - O código deve possuir **tratativas de exceções** para lidar com erros como:
     - Falha de conexão.
     - Erro ao carregar a página.
     - Falha na extração de dados.
     - Problemas ao gerar arquivos (JSON/planilha).
4. **Execução e Portabilidade**
   - O **README.md** deverá conter:
     - Instruções de instalação e execução.
     - Dependências necessárias.
     - Dados que precisam ser alterados caso o projeto seja executado em outro computador (por exemplo, caminhos locais, variáveis de ambiente, etc).
     - Sugestão de estrutura de pastas.

5. **Entrega**
   O candidato deverá disponibilizar o projeto:
   - Em um **repositório público no GitHub**.

Critérios de Avaliação

Os seguintes pontos serão avaliados durante a análise do desafio:

- **Estrutura e organização do código:** deve ser limpo, modularizado e com boa separação de responsabilidades entre os arquivos.
- **Comentários e documentação:** o código deve conter comentários claros explicando a lógica e o raciocínio por trás de cada parte da automação.
- **Tratamento de erros:** o uso de exceções (try/except) deve ser adequado e eficiente.
- **Uso de logs:** deve haver registro de eventos relevantes com níveis (DEBUG, INFO, WARNING, ERROR, CRITICAL) e timestamps.
- **Velocidade e eficiência:** o tempo de execução da automação deve ser razoável, evitando esperas desnecessárias.
- **Completude dos entregáveis:** todos os arquivos solicitados devem ser entregues e em pleno funcionamento.
- **Lógica e clareza:** a solução deve ser concisa, de fácil leitura e demonstrar boa compreensão do problema.
