**Relatório de desenvolvimento e execução do projeto “*Auto\_dashboard*”**

**por Vitor Gustavo Falavinha**

O projeto foi motivado pela curiosidade em aprender o funcionamento de processos automatizados via RPA (Robotic Process Automation) e conhecer as ferramentas utilizadas aplicando os conhecimentos adquiridos a um projeto prático de desenvolvimento de software.

Os dados para o desenvolvimento deste projeto foram coletados através do método *WebScraping*, que varre uma ou mais páginas web para a ingestão de dados. No caso deste projeto os dados coletados são: o número de visualizações e comentários de um determinado vídeo no YouTube. Através da obtenção e organização desses dados é possível, por exemplo, medir o desempenho e as interações dos usuários da plataforma com este conteúdo ao longo do tempo.

**1. MÉTODOS**

**1.1. PLANEJAMENTO E ESTIMATIVA DO PROJETO**

“*Auto\_dashboard*” consiste em uma série de etapas automatizadas, que vão desde a coleta de dados passam pela alimentação do banco de dados (planilha Excel) até a visualização das informações organizadas em um dashboard dinâmico em BI. O desenvolvimento do ambiente foi de aproximadamente uma semana, e a coleta dos dados amostrais se estendeu ao longo de um mês.

**1.2. ANÁLISE DE REQUISITOS DE SOFTWARE E SISTEMA**

O algoritmo do projeto foi desenvolvido no editor de código *Visual Studio Code* e a linguagem principal escolhida para o desenvolvimento da automação foi *Python*. Também foi utilizado VBA para a execução de macros na planilha onde os dados foram salvos.

Para a execução do projeto foi necessária a instalação das seguintes bibliotecas/*frameworks*: *Selenium*, para a automação Web, *Pyautogui* para a automação local e biblioteca *Openpyxl* para realizar a integração com a planilha Excel.

**1.3. PROJETO DE ESTRUTURA DE DADOS**

Os dados escolhidos para o desenvolvimento do projeto foram o número de visualizações e o número de comentários de um determinado vídeo do YouTube.

Os dados foram armazenados em uma planilha do Microsoft Excel.

1.4. ALGORÍTIMO DE PROCESSAMENTO

1.5. CODIFICAÇÃO

1.6. TESTE

1.7. MANUTENÇÃO

**2. FERRAMENTAS**

**2.1. Configuração do Selenium e webdrivers**

**2.2. Extração do número de visualizações e comentários**

Tags HTML para coleta dos dados

**2.3. Conversão, manipulação e tratamento dos dados (regex)**

Os dados chegavam como *string* e precisavam ser convertidos para números inteiros a fim de realizar as manipulações de desempenho.

**2.4. Automação com Pyautogui**

A biblioteca pyautogui permite automatizar o controle de mouse e teclado. isso foi útil em vários momentos...

**2.5. Atualização da planilha Excel (VBA)**

A biblioteca *Openpyxl*  foi a responsável por adicionar os itens coletados pelo webscraping à planilha Excel.

A cada “run” do software “auto\_dashboard” os dados eram inseridos na planilha e atualizados através de macros em VBA.

**2.6. Agendamento da automação (Agendador de tarefas do Windows)**

A pior parte, muitos erros!!!!!

**2.7. Integração com o PowerBI**

Foi criado um dashboard na ferramenta PowerBI que recebeu os dados da planilha Excel e permitiu sua visualização dinâmica.

**3. PROCEDIMENTOS**