
Plano de Implementação de Automação em Processos

1. Identificação de Processos para Automação

- **Objetivo:** Identificar tarefas repetitivas e demoradas que podem ser automatizadas.
- **Ações:**
 - Realizar um levantamento das atividades realizadas diariamente ou com frequência.
 - Priorizar tarefas que consomem mais tempo ou apresentam maior risco de erros humanos.
 - Exemplos de processos que podem ser automatizados:
 - Organização de arquivos em pastas específicas.
 - Coleta de dados de sites ou sistemas.
 - Envio de notificações automáticas.
 - Conversão de formatos de arquivos.

2. Definição de Ferramentas e Tecnologias

- **Objetivo:** Escolher as ferramentas adequadas para a automação.
- **Ações:**
 - Utilizar Python como linguagem principal, devido à sua versatilidade e ampla biblioteca de suporte.
 - Selecionar bibliotecas específicas para cada tipo de automação:
 - **Organização de arquivos:** `os`, `shutil`.
 - **Web scraping:** `requests`, `BeautifulSoup`.
 - **Bots de comunicação:** `python-telegram-bot`.
 - **Processamento de arquivos:** `pandas`.

3. Desenvolvimento de Scripts

- **Objetivo:** Criar scripts personalizados para cada processo identificado.
- **Ações:**
 - Escrever scripts para cada tarefa, utilizando os exemplos do portfólio de Emanuel como base.
 - **Exemplo prático:**

Organização de arquivos:

```
import os
import shutil
```

```
def organizar_arquivos(caminho):
    tipos = {
        "Imagens": [".jpg", ".png", ".jpeg"],
```

```

"Documentos": [".pdf", ".docx", ".txt"],
"Planilhas": [".xlsx", ".csv"],
"Vídeos": [".mp4", ".avi"]
}
for arquivo in os.listdir(caminho):
    nome, extensao = os.path.splitext(arquivo)
    extensao = extensao.lower()
    for pasta, extensoes in tipos.items():
        if extensao in extensoes:
            destino = os.path.join(caminho, pasta)
            if not os.path.exists(destino):
                os.makedirs(destino)
            shutil.move(os.path.join(caminho, arquivo), os.path.join(destino, arquivo))
            break

organizar_arquivos("C:/Downloads")
print("Arquivos organizados com sucesso!")

```

■

Web scraping:

```

import requests
from bs4 import BeautifulSoup

url = "https://www.exemplo.com/produtos"
resposta = requests.get(url)
sopa = BeautifulSoup(resposta.text, "html.parser")

produtos = sopa.find_all("div", class_="produto")
for produto in produtos:
    nome = produto.find("h2").text
    preco = produto.find("span", class_="preco").text
    print(f"Produto: {nome}, Preço: {preco}")

```

■

Bot para Telegram:

```

from telegram.ext import Updater, CommandHandler

def start(update, context):
    update.message.reply_text("Olá! Como posso ajudar você hoje?")

updater = Updater("SEU_TOKEN_AQUI", use_context=True)
dp = updater.dispatcher
dp.add_handler(CommandHandler("start", start))
updater.start_polling()
updater.idle()

```



4. Teste e Validação

- **Objetivo:** Garantir que os scripts funcionem corretamente e atendam às necessidades.
- **Ações:**
 - Testar os scripts em um ambiente controlado com dados reais ou simulados.
 - Identificar e corrigir possíveis erros ou falhas.
 - Validar os resultados obtidos para garantir precisão e eficiência.

5. Implementação e Monitoramento

- **Objetivo:** Colocar os scripts em produção e monitorar seu desempenho.
- **Ações:**
 - Integrar os scripts aos sistemas existentes, se necessário.
 - Configurar alertas ou relatórios para monitorar o funcionamento dos scripts.
 - Realizar ajustes conforme necessário para melhorar a performance ou corrigir problemas.

6. Treinamento e Documentação

- **Objetivo:** Garantir que a equipe esteja capacitada para utilizar e manter os scripts.
- **Ações:**
 - Criar documentação detalhada para cada script, incluindo instruções de uso e manutenção.
 - Treinar a equipe para operar os scripts e realizar ajustes básicos.

7. Avaliação e Melhoria Contínua

- **Objetivo:** Avaliar os resultados da automação e identificar oportunidades de melhoria.
- **Ações:**
 - Monitorar os indicadores de desempenho dos processos automatizados.
 - Coletar feedback dos usuários para identificar problemas ou oportunidades de melhoria.
 - Atualizar os scripts conforme necessário para atender a novas demandas ou corrigir falhas.

Conclusão

Este plano fornece um guia prático para implementar automação em processos, utilizando as habilidades e exemplos do portfólio de Emanuel. A automação pode trazer benefícios significativos, como maior eficiência, redução de erros e economia de recursos. Com os exemplos de scripts fornecidos, é possível começar a implementar soluções personalizadas para atender às necessidades específicas de qualquer projeto ou empresa.

