DOMINE AUTOMAÇÃO COM PYTHON

TRANSFORME TAREFAS REPETITIVAS EM RESULTADOS INCRÍVEIS!

ALLAN CORDEIRO

INTRODUÇÃO DE AUTOMAÇÃO COM PYTHON

Este e-Book é um guia prático para iniciantes que desejam aprender automação com Python de maneira fácil e direta. Utilizando um exemplo real de envio automatizado de e-mails pelo Gmail, o livro cobre desde a configuração do ambiente de desenvolvimento até a manipulação de planilhas e o envio de e-mails personalizados, ideal para quem quer ter uma base para automatizar tarefas repetitivas e aumentar a eficiência no trabalho.

Este e-Book contou com a ajuda de inteligências artificiais para geração de imagens e complemento de textos. Além disso o conteúdo foi gerado com fins didáticos, portanto não foi realizado uma validação minuciosa humana, pode conter erros por IA, ortográficos e de concordância.

01

FERRAMENTAS NECESSÁRIAS

FERRAMENTAS NECESSÁRIAS

Antes de começarmos a programar, precisamos instalar e configurar algumas ferramentas essenciais.

- **Python:** A linguagem de programação que vamos usar. Baixe e instale do site oficial: *python.org*.

- **Visual Studio Code (VS Code)**: Depois de instalar Python, você precisará de um editor de código, esse é gratuito e fácil de usar. Baixe e instale do site oficial: *code.visualstudio.com*

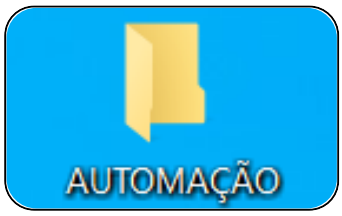
02

PREPARANDO O VS CODE

PREPARANDO O VS CODE

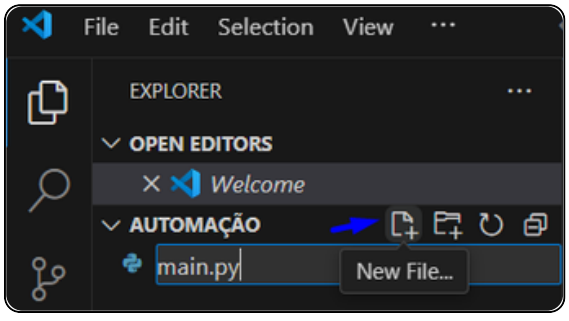
Antes de começarmos a programar, precisamos preparar o ambiente.

- **Pasta principal:** Crie uma pasta em qualquer lugar, nomeie conforme desejar, como neste exemplo : AUTOMAÇÃO.



- **Abrindo VS Code:** Dentro da nova pasta, clice com o lado direito do mouse e abra com code:

- **Criando arquivo .py**: Crie um arquivo chamado main.py para poder programar com python:



03

BIBLIOTECAS NECESSÁRIAS

BIBLIOTECAS NECESSÁRIAS

Após a instalação do python e do ambiente VS Code, precisamos instalar e configurar algumas bibliotecas.

- **Instalando a Pandas**: Biblioteca poderosa para manipulação e análise de dados. Abra o terminal no VS Code e instale usando o seguinte comando:

```
pip install pandas
```

- **Importando a Pandas**: Na parte dos códigos importe pandas como pd(abreviado para facilitar):

```
import pandas as pd
```

- **Bibliotecas MIME**: importe para poder criar e-mails no formato correto:

```
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
```

- **smtplib**: importe para poder enviar e-mails:

```
import smtplib
```

04

BASE DE DADOS: PLANILHA .CSV

BASE DE DADOS: PLANILHA .CSV

Para o nosso exemplo, a base de dados será uma planilha de formato csv.

- **Criando a planilha**: Crie uma planilha dentro da pasta inicial, com as seguintes colunas e preenchendo conforme exemplo abaixo(Utilizado o LibreOffice calc):

	A	B	C	D	E
1	ID	NOME	VENCIMENTO	VALOR	E-MAIL
2	10032	SOFIA MENDES	10.04.2024	1000	Seu_email_para_teste@gmail.com
3	20048	LUCAS OLIVEIRA	09.04.2024	2300	Seu_email_para_teste@gmail.com
4	10458	ISABELA PEREIRA	09.04.2024	45	Seu_email_para_teste@gmail.com
5	10025	MATEUS SANTOS	07.04.2024	52	Seu_email_para_teste@gmail.com
6	20096	LAURA COSTA	06.04.2024	35	Seu_email_para_teste@gmail.com

- **Salvando a planilha**: Salve no formato csv, nomeie conforme desejar, no exemplo ficou *cliente.csv:*

- **Leitura da base de dados**: com a planilha na mesma pasta, basta apenas criar uma variavel e ler com a pandas (pd):

```
clientes = pd.read_csv('./clientes.csv')
```

05

PREPARANDO OS DADOS DOS E-MAILS

PREPARANDO OS DADOS DOS E-MAILS

Vamos estruturar o algoritmo para que a automação selecione os dados e padronize os e-mails.

- **Estrutura dos e-mails**: Agora vamos percorrer cada linha da planilha e enviar um e-mail para cada cliente. Vamos criar um objeto de e-mail para cada cliente e preencher os detalhes necessários:

```
for index, cliente in clientes.iterrows():
    # Dados do e-mail:
    msg = MIMEMultipart()
    msg['From'] = 'Seu_email_para_teste@gmail.com'  # Substitua pelo seu g-mail
    msg['To'] = cliente['E-MAIL']# seleciona os e-mails da a coluna "E-MAIL"
    msg['subject'] = f"COBRANÇA - {cliente['ID']} - {cliente['NOME']}" #seleciona as devidas colunas

    #Corpo do e-mail, cada parte vai selecionar sozinho os dados de cada coluna, o texto é opicional,
    # mas entenda a estrutura para poder fazer alterações
    message = f"Olá {cliente['NOME']},\n\n" \
              f"O boleto de vencimento {cliente['VENCIMENTO']} e valor R$ {cliente['VALOR']} está em aberto.\n\n" \
              f"Poderia verificar?\n\n"\
              f"Atenciosamente"
    msg.attach(MIMEText(message, 'plain'))
```

06

OBTENDO A SENHA DE APLICAÇÃO NO GMAIL

OBTENDO A SENHA DE APLICAÇÃO NO GMAIL

O Gmail não permite fazer aplicações com a senha principal.

- **Ativar a Verificação em Duas Etapas**: Acesse sua conta do Google e vá para "Segurança" nas configurações. Na seção "Como fazer login no Google", clique em "Verificação em duas etapas" e siga as instruções para ativá-la.
- **Gerar uma Senha de Aplicação:** Depois de ativar a verificação em duas etapas, volte para a seção "Segurança". Na seção "Fazer login no Google", clique em "Senhas de app", Se solicitado, faça login novamente para confirmar sua identidade, Selecione "Outro (nome personalizado)" na lista de aplicativos, insira um nome como "Python Email Script" e clique em "Gerar". Uma senha de 16 caracteres será exibida. Copie e cole no nosso código. Dica: E-mails novos que acabaram de ativar a verificação de duas etapas podem não gerar a opção da senha de aplicação, utilize um e-mail mais antigo ou aguarde um tempo, alguns dizem 48 horas.

07

CONECTANDO AO SERVIDOR DO E-MAIL

CONECTANDO AO SERVIDOR E ENVIANDO OS E-MAILS

Como neste exemplo estamos focando no Gmail, então a porta e dados devem ser conforme abaixo

- **Estruturando e conectando ao servidor:** Após obter a senha de aplicações, para enviar os e-mails precisamos nos conectar ao servidor SMTP do Gmail. Vamos usar a biblioteca smtplib para isso.

```
# Conectando ao servidor SMTP
server = smtplib.SMTP('smtp.gmail.com', port=587)
server.starttls()
server.login('Seu_email_para_teste@gmail.com', 'SENHA DE APLICAÇÃO')
server.sendmail(msg['From'], msg['To'], msg.as_string())
server.quit()
```

- **Testando e Depurando:** Depois de escrever o código, é importante testar e garantir que tudo está funcionando como esperado. Verifique se os e-mails estão sendo enviados corretamente e se os detalhes estão corretos.

CONSIDERAÇÕES FINAIS

CONCLUSÃO E AGRECIMENTO

Você aprendeu a criar um script Python que lê uma planilha de clientes e envia e-mails personalizados. Com essas habilidades, você pode automatizar muitas tarefas repetitivas e se tornar mais eficiente no seu trabalho.

Este é apenas o começo da sua jornada com Python. Continue explorando e aprendendo, e logo você se tornará um programador competente e confiante!

Desta forma, deixo aqui meus agradecimentos por ter lido até aqui. Muito obrigado!

AUTOR:

ALLAN CORDEIRO

LINKEDIN

GITHUB