# Automação de Cadastro de Clientes

Este é um script de automação em Python projetado para ler dados de clientes de um arquivo CSV, validá-los, preencher um formulário web local com os dados válidos, registrar os resultados em um banco de dados SQLite e enviar um e-mail de boas-vindas para cada cliente cadastrado com sucesso.

## Funcionalidades

* **Leitura de Dados**: Lê informações de clientes a partir de um arquivo dados.csv.
* **Validação de Dados**: Valida os formatos de CPF e e-mail para garantir a integridade dos dados.
* **Log de Erros**: Registros com dados inválidos são separados e salvos em logs\_erros.csv.
* **Automação Web**: Utiliza o Selenium para abrir um formulário HTML local (form.html), preencher os campos e submeter o cadastro.
* **Persistência de Dados**: Cria e utiliza um banco de dados SQLite (cadastro.db) para armazenar o status de cada tentativa de cadastro (sucesso ou falha).
* **Notificação por E-mail**: Envia um e-mail de boas-vindas para os clientes cadastrados com sucesso usando o servidor SMTP do Gmail.

## Pré-requisitos

Antes de executar o script, você precisará ter o seguinte instalado:

* [Python 3.x](https://www.python.org/downloads/)
* O navegador [Google Chrome](https://www.google.com/chrome/)
* [ChromeDriver](https://googlechromelabs.github.io/chrome-for-testing/) compatível com a sua versão do Google Chrome. Certifique-se de que o executável do ChromeDriver esteja no PATH do seu sistema ou no mesmo diretório do script.

## Instalação

1. Clone este repositório ou baixe os arquivos para o seu computador.
2. Navegue até o diretório do projeto e instale as dependências Python necessárias:  
   pip install pandas selenium schedule

## Configuração

Antes de executar o script, você **precisa** configurar alguns caminhos e credenciais diretamente no código:

1. **Caminho do CSV de Entrada**: Na linha 16, atualize o caminho para o seu arquivo dados.csv.  
   df = pd.read\_csv(  
    r"C:\caminho\completo\para\seus\dados.csv",  
    # ...  
   )
2. **Caminho do Formulário HTML**: Na linha 101, atualize o caminho para o seu arquivo form.html.  
   driver.get("C:/caminho/completo/para/seu/form.html")
3. **Credenciais de E-mail**: Na função enviar\_email, insira o e-mail do remetente e a senha.  
   def enviar\_email(nome, email):  
    remetente = "seu-email@gmail.com" # SEU E-MAIL AQUI  
    senha = "SUA\_SENHA\_DE\_APP\_AQUI" # SUA SENHA AQUI  
    # ...  
   **⚠️ Aviso de Segurança**: Para usar uma conta do Gmail, é altamente recomendável gerar uma **"Senha de App"** em vez de usar sua senha principal. Para isso, ative a verificação em duas etapas na sua Conta Google e depois gere a senha em [Senhas de app](https://myaccount.google.com/apppasswords).

## Estrutura do dados.csv

O arquivo dados.csv deve ter um separador de ponto e vírgula (;) e conter as seguintes colunas:

* NOME COMPLETO
* CPF
* E-MAIL
* TelefoneContato
* Nascimento

## Como Executar

Após concluir a instalação e a configuração, basta executar o script a partir do seu terminal:

python nome\_do\_script.py

*(Substitua nome\_do\_script.py pelo nome real do seu arquivo Python)*

O script iniciará o processo, exibindo o status de cada cliente no console.

## Arquivos Gerados

* **logs\_erros.csv**: Contém os registros do dados.csv que falharam na validação inicial de CPF ou e-mail. Erros de cadastro durante a automação também são anexados a este arquivo.
* **cadastro.db**: Banco de dados SQLite que armazena todos os clientes processados e seu status final (Deu bom! para sucesso ou a mensagem de erro para falhas).