Estrutura do Repositório GitHub para Deploy no Streamlit Community Cloud

Para que o Streamlit Community Cloud consiga encontrar e executar sua aplicação corretamente, a organização dos arquivos dentro do seu repositório no GitHub é importante. Para o seu projeto `Dashboard de Leads`, a estrutura recomendada é bastante simples:

Estrutura Recomendada (Arquivos na Raiz do Repositório):

```
SeuRepositorio/
├── .gitattributes       (Opcional, o GitHub pode criar)
├── .gitignore           (Opcional, mas recomendado para ignorar .venv, etc.)
├── app_streamlit_folium.py (Seu script principal do Streamlit)
├── requirements.txt     (Arquivo com as dependências Python)
├── leads_baixada.csv    (Seu arquivo de dados)
└── README.md            (Opcional, mas bom para descrever o projeto)
```

Explicação dos Arquivos Essenciais:

1. **`app_streamlit_folium.py`**: Este é o arquivo Python que contém o código da sua aplicação Streamlit. O Streamlit Cloud procurará por este arquivo (ou o arquivo que você especificar durante o deploy) para executar.
2. **`requirements.txt`**: Este arquivo é **crucial**. Ele lista todas as bibliotecas Python que sua aplicação precisa para funcionar (streamlit, pandas, folium, streamlit-folium). O Streamlit Cloud usará este arquivo para instalar automaticamente essas dependências no ambiente onde sua aplicação rodará.
3. **`leads_baixada.csv`**: Este é o seu arquivo de dados. Como o seu script `app_streamlit_folium.py` está configurado para carregar um arquivo que o usuário faz upload na barra lateral, para o deploy no Streamlit Cloud, você tem duas opções principais:
    - **Opção A (Mais Simples para Começar):** Incluir o `leads_baixada.csv` diretamente no repositório GitHub (como mostrado na estrutura acima). O Streamlit Cloud terá acesso a ele. Você precisaria **modificar levemente** o script `app_streamlit_folium.py` para carregar este arquivo automaticamente em vez de esperar pelo upload. Por exemplo, trocar a seção de upload por algo como: ```python # Em vez de: # uploaded = st.sidebar.file_uploader(...) # if uploaded: # df = load_data(uploaded)

Carregar diretamente:

try: # Assumindo que o CSV está na mesma pasta do script no repositório df = pd.read_csv("leads_baixada.csv") # Aplicar as mesmas validações e limpezas da função load_data original # ... (copiar lógica de validação/limpeza de load_data aqui ou chamar uma função) ... if df is None: # Se a validação falhar st.error("Erro ao carregar ou validar o arquivo leads_baixada.csv do repositório.") st.stop() # Impede a execução do resto do script except FileNotFoundError: st.error("Erro: Arquivo 'leads_baixada.csv' não encontrado no repositório.") st.stop() except Exception as e: st.error(f"Erro inesperado ao carregar 'leads_baixada.csv': {e}") st.stop()

O restante do código continua a partir daqui, usando o 'df' carregado

if df is not None: # ... (filtros, mapa, etc.) ... `` * **Opção B (Mais Flexível/Segura):** Não incluir o .csv no repositório. Modificar o script para carregar os dados de outra fonte online (ex: uma planilha Google Sheets pública, um banco de dados, um arquivo em um serviço de armazenamento como AWS S3 ou Google Cloud Storage). Isso é mais seguro se os dados forem sensíveis e evita que o repositório fique grande. * **Opção C (Manter Upload):** Manter o código como está, com o st.sidebar.file_uploader . A aplicação funcionará no Streamlit Cloud, mas *cada usuário* que acessá-la precisará fazer o upload do arquivo .csv` toda vez que usar o dashboard. Isso pode não ser ideal para compartilhar com um cliente que espera ver os dados diretamente.

Recomendação: Para um deploy inicial e simples, a **Opção A** (incluir o CSV e modificar o script para carregá-lo automaticamente) é geralmente a mais direta.

Arquivos Opcionais:

- `.gitignore` : Útil para dizer ao Git quais arquivos ou pastas ignorar (ex: a pasta `.venv` , arquivos temporários, etc.).
- `README.md` : Arquivo de texto onde você pode descrever seu projeto.

Certifique-se de que seu repositório no GitHub seja **público** para usar o plano gratuito do Streamlit Community Cloud.