### **(C) Contexto (Context)**

Você é um cientista de dados sênior e desenvolvedor Python, especialista na criação de ferramentas de análise interativas para desafios socioambientais. Uma liderança comunitária da Amazônia, preocupada com o impacto das variações climáticas na saúde e na segurança alimentar, solicitou sua ajuda para transformar dados brutos em insights acionáveis.

Você recebeu duas bases de dados "sujas" em formato CSV (base\_climatica.csv e base\_socioeconomica.csv), que contêm problemas conhecidos como valores nulos, registros duplicados, erros de formatação (ex: "nao", "Sim"), outliers extremos (ex: chuva > 700mm) e datas com erros sistemáticos (ex: ano incorreto).

O objetivo final não é um relatório estático, mas sim uma **aplicação web multi-páginas robusta e interativa**, construída com Streamlit, que sirva como uma ferramenta de análise reutilizável.

### **(O) Objetivo (Objective)**

Desenvolver uma aplicação Streamlit completa, organizada em 5 páginas, que guie um usuário (analista ou líder comunitário) através de todo o ciclo de vida de um projeto de análise de dados. A aplicação deve ser totalmente interativa, dando ao usuário controle sobre as decisões críticas de limpeza e análise.

**Funcionalidades por Página:**

1. **Página 1 - Introdução:** Apresentar o contexto do projeto, seus objetivos e o dicionário de dados das bases de entrada.
2. **Página 2 - Carga e Exploração Inicial:** Permitir que o usuário faça o upload dos dois arquivos CSV. Após o upload, deve exibir um diagnóstico inicial para cada base, incluindo as primeiras linhas (.head()), informações gerais (.info()), contagem de valores nulos e duplicados.
3. **Página 3 - Limpeza e Preparação (Totalmente Interativa):** Esta é a página mais crítica.
   * **Passo 1: Painel de Validação Configurável:** Exibir as estatísticas descritivas (.describe()) dos dados brutos e fornecer widgets interativos (st.slider) para que o usuário defina as regras de validação (ex: faixas aceitáveis para temperatura e umidade).
   * **Passo 2: Painel de Decisão de Tratamento:** Após uma validação bem-sucedida, apresentar um novo formulário onde o usuário escolhe:
     + Como tratar outliers (st.radio com opções para "Não tratar" ou "Usar método estatístico IQR").
     + Como tratar cada coluna com valores ausentes (st.selectbox com opções como "Mediana", "Média", "Remover Linhas").
   * **Passo 3: Finalização:** Um botão final aplica todas as decisões do usuário, executa a limpeza, trata os outliers, imputa os nulos, une as duas bases de dados, **corrige o ano da data para 2024**, e **ordena o DataFrame final pela data**.
   * **Funcionalidade Extra:** Permitir que o usuário faça o download do CSV limpo e também que carregue um arquivo já limpo, pulando o processo interativo.
4. **Página 4 - Análise Exploratória (EDA):** Usando os dados limpos, gerar visualizações interativas com Plotly.
   * **Análise Univariada:** Histograma de uma variável numérica selecionável.
   * **Séries Temporais:** Gráfico de linhas com seleção múltipla de variáveis.
   * **Análise Sazonal:** Incluir uma decomposição de série temporal (statsmodels.tsa.seasonal\_decompose) para uma variável selecionável, mostrando os gráficos de Tendência, Sazonalidade e Resíduos.
   * **Análise Bivariada:**
     + Um **Box Plot** para incidencia\_doencas vs. acesso\_agua\_potavel, utilizando uma **escala logarítmica no eixo Y (log\_y=True)** para melhor visualização.
     + Um **Scatter Plot** para chuvas\_reais\_mm vs. volume\_producao\_tons.
   * **Análise Multivariada:** Um mapa de calor da matriz de correlação.
5. **Página 5 - Conclusões:**
   * Apresentar um resumo dos principais achados em linguagem clara.
   * Re-exibir os gráficos mais importantes (Box Plot e Scatter Plot).
   * **Crucial:** Realizar e apresentar o resultado de um **teste de significância estatística (Teste t de Welch)** para comparar a incidência de doenças entre os dois grupos, incluindo a interpretação do p-valor.

### **(S) Estilo (Style)**

Adote a persona de um **"Engenheiro de Dados Analítico"**. O código deve ser limpo, eficiente, robusto e bem comentado, explicando as decisões técnicas. As explicações na interface da aplicação devem ser didáticas, claras e objetivas, guiando o usuário sem sobrecarregá-lo com jargões.

### **(T) Tom (Tone)**

O tom deve ser profissional, confiável e colaborativo. A aplicação deve passar a sensação de ser uma ferramenta poderosa e precisa, mas ao mesmo tempo acessível e fácil de usar.

### **(A) Audiência (Audience)**

A audiência final da aplicação é um analista de dados, um estudante ou um líder comunitário com boa capacidade de interpretação, mas que não é necessariamente um programador. A ferramenta deve empoderá-lo a realizar a análise por conta própria.

### **(R) Resposta (Format)**

A resposta deve ser o **código Python completo e final para cada um dos 5 arquivos .py** (app.py e os 4 arquivos na pasta pages), além do conteúdo para um arquivo requirements.txt. Cada arquivo deve ser apresentado em um bloco de código separado e claramente identificado.