JOÃO VITOR NOGUEIRA SIMOE

DASHBOARD INTERATIVO COM PYTHON E DASH PARA ANÁLISE DE DADOS MULTIDISCIPLINARES:

PROVA DE CONCEITO PARA CORRELAÇÕES

Sorocaba

2025

JOÃO VITOR NOGUEIRA SIMOE

DASHBOARD INTERATIVO COM PYTHON E DASH PARA ANÁLISE DE DADOS MULTIDISCIPLINARES:

PROVA DE CONCEITO PARA CORRELAÇÕES

Trabalho de Conclusão de Curso apresentado à Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba, como parte dos requisitos para obtenção do grau de Bacharel(a) em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Leopoldo André Dutra Lusquino Filho

Coorientador: Prof. Dr. Galdenoro Botura Junior

Sorocaba

2025

S593d

Simoe, João Vitor Nogueira

Dashboard interativo com Python e Dash para análise de dados multidisciplinares : prova de conceito para correlações / João Vitor Nogueira Simoe. -- Sorocaba, 2025

70 p. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Controle e Automação) - Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba

Orientador: Leopoldo André Dutra Lusquino Filho

Coorientador: Galdenoro Botura Junior

1. Análise. 2. Correlação (Estatística). 3. Administração pública. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

JOÃO VITOR NOGUEIRA SIMOE

DASHBOARD INTERATIVO COM PYTHON E DASH PARA ANÁLISE DE DADOS MULTIDISCIPLINARES:

PROVA DE CONCEITO PARA CORRELAÇÕES

Trabalho de Conclusão de Curso apresentado ao Instituto de Ciência e Tecnologia de Sorocaba, Universidade Estadual Paulista (UNESP), como parte dos requisitos para obtenção do grau de Bacharel(a) em Engenharia de Controle e Automação.

Data da defesa: 29/08/2025

BANCA EXAMINADORA:

Prof. Dr. Leopoldo André Dutra Lusquino Filho
UNESP – Instituto de Ciência e Tecnologia – Campus de Sorocaba

Prof. Dr. Eduardo Verri Liberado
UNESP – Instituto de Ciência e Tecnologia – Campus de Sorocaba

Dra. Liliane Moreira Nery
UNESP – Instituto de Ciência e Tecnologia – Campus de Sorocaba

A Deus e à minha família que me dão forças para seguir em frente.

AGRADECIMENTOS

Não sou uma pessoa de muitas palavras, porém, gostaria de aproveitar este espaço e oportunidade para realizar o meu agradecimento a determinados indivíduos que foram de extrema importância não somente para o desenvolvimento deste projeto, mas também para minha trajetória acadêmica na UNESP Sorocaba

Desejo agradecer primeiramente a todos os professores da UNESP Sorocaba que apesar de rigorosos transferiram tudo o que sabiam para todos nós de forma proveitosa e interessante. Dentre estes docentes, agradecer imensamente ao meu orientador, Prof. Dr. Leopoldo André Dutra Lusquino Filho, ao co-orientador, Prof. Dr. Galdenoro Botura Junior, a um dos idealizadores desta jornada, Prof. Dr. Darllan Collins da Cunha e Silva e aos meus avaliadores, Prof. Dr. Eduardo Verri Liberado e Profa. Dra. Liliane Moreira Nery.

Além dos meus professores, gostaria de realizar um agradecimento a todos meus colegas de turma e meus amigos que levarei para a vida, que conviveram comigo no dia a dia ou por alguns momentos, em especial, Rafael Yamamoto, Leonardo Vargas, Felipe Bueno e Seiryo Kashiwaba, que tiveram muita paciência e parceria ao me aguentarem por tantos dias sob o mesmo teto e, ao Vinicius Rodella, Felipe Henrique Teixeira, Gustavo Brocanelli e todos dos Servos da 51, os quais tive o prazer de compartilhar de momentos memoráveis, com muitos aprendizados e ideias mirabolantes.

Por fim, gostaria de realizar um agradecimento mais do que especial às pessoas mais importantes da minha vida. Gostaria de começar pela minha família, Mirian Nogueira, Alvino Simoe, Wanessa Nogueira, Edson Maruyama, Zaha e Lucia Shimoe que nunca deixaram meu lado, sempre acreditaram em mim e em meus sonhos e são a fonte de todo meu conhecimento, alegria e do ser humano que sou hoje. Agradecer também aos meus irmãos de outras mães Vinicius Scarpelli e Daniel Binenbojm que estiveram, pelos últimos 17 anos, me apoiando e incentivando em todas minhas empreitadas, bem como minha namorada, Sthefani Ribeiro, que demonstrou ser uma verdadeira guerreira e parceira, nunca me permitindo esmorecer e sendo sempre minha fonte de luz.

Obrigado a todos que participaram direta ou indiretamente deste projeto e que tornaram possível meu sonho de me tornar um Engenheiro pela UNESP Sorocaba.

RESUMO

Este trabalho apresenta o desenvolvimento e a implantação de um dashboard interativo para a análise integrada de dados multidisciplinares (emissões atmosféricas, epidemiologia e socioeconomia), visando solucionar a problemática da fragmentação de dados públicos no Brasil e o alto custo de ferramentas comerciais de *Business Intelligence* (BI). Utilizando tecnologias *open-source*, como Python e o framework Dash, foi construída uma plataforma que permite a exploração de dados de forma autônoma e acessível. A metodologia empregou uma abordagem ágil por meio de iterações, com versionamento via GitHub, e uma arquitetura de software inspirada no padrão Model-View-Controller (MVC) de Reenskaug para garantir a manutenibilidade do código. A plataforma resultante, cuja interface foi guiada por princípios de design e visualização de dados, oferece filtros em cascata, visualizações dinâmicas e uma seção dedicada à análise estatística, incluindo a detecção de *outliers* pelo método do Intervalo Interquartil (IQR) de Tukey e análise de correlação com múltiplos coeficientes (Pearson; Spearman; Kendall). O ciclo de desenvolvimento foi concluído com a implantação em nuvem na plataforma Heroku, onde a otimização do desempenho, por meio de cache de dados em memória, foi um passo crucial para assegurar a responsividade e a estabilidade da aplicação em um ambiente de produção. O resultado é uma prova de conceito funcional que valida a viabilidade de se construir ferramentas de BI de baixo custo e alto impacto, democratizando o acesso à análise de dados complexos e fornecendo uma base sólida para futuras expansões, como a integração de modelos preditivos e análises geoespaciais.

Palavras-chave: Dashboard Interativo, Python, Business Intelligence, Visualização de Dados, Análise de Correlação.

ABSTRACT

This work presents the development and deployment of an interactive dashboard for the integrated analysis of multidisciplinary data (atmospheric emissions, epidemiology, and social economy), aiming to solve the problem of fragmented public data in Brazil and the high cost of commercial *Business Intelligence* (BI) tools. Using open-source technologies, such as Python and the Dash framework, a platform was built to allow for autonomous and accessible data exploration. The methodology employed an agile approach through the usage of iterations, with version control via GitHub, and software architecture inspired by Reenskaug's Model-View-Controller (MVC) pattern to ensure code maintainability and scalability. The resulting platform, whose interface was guided by universal design and data visualization principles, offers cascading filters, dynamic visualizations, and a dedicated section for statistical analysis, including outlier detection using Tukey's Interquartile Range (IQR) method and correlation analysis with multiple coefficients (Pearson; Spearman; Kendall). The development cycle was completed with the cloud deployment on the Heroku platform, where performance optimization through in-memory data caching was a crucial step to ensure the application's responsiveness and stability in a production environment. The result is a functional proof-of-concept that validates the feasibility of building low-cost, high-impact BI tools, democratizing access to complex data analysis and providing a solid foundation for future enhancements, such as the integration of predictive models and geospatial analyses.

Keywords: Interactive Dashboard, Python, Business Intelligence, Data Visualization, Correlation Analysis.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BI	Inteligência de Negócio *(Business Inteligence)*
CO2	Dióxido de Carbono
CPCB	Conselho de Controle de Poluição
DATASUS	Departamento de Informática do Sistema Único de Saúde
DCU	Design Centrado no Usuário
DU	Design Universal
DWH	Armazém de Dados *(Datawarehousing)*
EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
ETL	Extração, Transformação e Carregamento (*Extract, Transform, Load*)
HCI	Interface Humano-Computador (*Human-Computer Interface*)
IBGE	Instituto Brasileiro de Geografia e Estatística
IQR	Intervalo Interquartil
I/O	Entrada/Saída (*Input/Output*)
KPI	Indicadores de Performance (*Key Performance Indicators*)
MPN	Número Mais Provável (*Most Probable Number*)
MVC	Modelo-Visualização-Controlador (*Model-Viewer-Controller*)
MVVM	Modelo-Visualização-VisualizaModelo (*Model-View-ViewModel*)
MVP	Modelo-Visualização-Apresentador (*Model-Viewer-Presenter*)
PIB	Produto Interno Bruto
POC	Prova de Conceito (*Proof of Concept*)
TDS	Transformação de Dados e Scripts *(Data Transformation and Scripting)*
URL	Localizador Uniforme de Recursos *(Uniform Resource Locator)*
UI/UX	Interface de Usuário/ Experiência de Usuário *(UserInterface / User Experience)*

SUMÁRIO

1. INTRODUÇÃO

1.1 Contexto e justificativa

1.1.1 O desafio da análise de dados multidisciplinares

Na era da informação, organizações governamentais e instituições de pesquisa geram um volume massivo de dados públicos. No contexto brasileiro, informações cruciais sobre epidemiologias, emissões atmosféricas e desenvolvimento socioeconômico são frequentemente disponibilizadas por diferentes órgãos, como o IBGE (Instituto Brasileiro de Geografia e Estatística), o DataSUS (Departamento de Informática do Sistema Único de Saúde) e a EMBRAPA (Empresa Brasileira de Pesquisa Agropecuária).

Embora valiosos, esses dados se encontram dispersos em repositórios distintos, com formatos heterogêneos e sem uma plataforma unificada para análise integrada. Essa fragmentação impõe barreiras significativas para pesquisadores, gestores públicos e a sociedade civil, que necessitam cruzar informações de diferentes domínios para obter uma compreensão holística de problemas complexos, como o impacto da urbanização na saúde pública ou a relação entre desenvolvimento econômico e emissões de poluentes (Silva et al., 2019).

A problemática central reside na dificuldade de acesso e na complexidade de consolidar e analisar esses dados de forma simultânea. Ferramentas tradicionais, como planilhas eletrônicas, são insuficientes para lidar com a escala e a dimensionalidade desses conjuntos de dados (Alpar; Schulz, 2021), enquanto soluções de *Business Intelligence* (BI) comerciais, como Tableau ou Power BI, embora poderosas, representam um alto custo de licenciamento, tornando-se inviáveis para muitos contextos acadêmicos e para organizações de pequeno e médio porte, como dito na introdução deste projeto.

Surge, portanto, a necessidade de uma ferramenta que atenda a um duplo requisito: ser acessível financeiramente e, ao mesmo tempo, robusta analiticamente. Uma solução ideal deve ser capaz de centralizar dados de múltiplas fontes, oferecer diversas formas de análise interativa e ser construída sobre uma base tecnológica que permita controle, personalização e escalabilidade. Este cenário justifica a exploração de tecnologias *open-source*, que emergem como alternativas viáveis para a democratização da análise de dados, permitindo a criação de soluções com um nível de profissionalismo e funcionalidade próximo ao de ferramentas consolidadas no mercado, mas sem as barreiras de custo (Perkel, 2022).

1.1.2 Justificativa do Projeto

Este trabalho se justifica pela carência de plataformas analíticas no Brasil que integrem dados socioeconômicos, de epidemiologias e emissões atmosféricas de maneira interativa e acessível. A proposta é desenvolver uma prova de conceito, ou *Proof of Concept* (POC), de um dashboard que não apenas visualize dados, mas que também capacite o usuário a explorar correlações e extrair *insights* de forma autônoma. A plataforma projetada visa democratizar o acesso a dados complexos, tornando-se uma ferramenta valiosa para a análise de políticas públicas, pesquisa acadêmica e tomada de decisões embasadas em evidências.

1.2 Considerações Iniciais

O *Business Intelligence* (BI), ou Inteligência de Negócios, refere-se ao conjunto de estratégias, processos e tecnologias voltados à coleta, organização, análise e disponibilização de informações, com o objetivo de apoiar a tomada de decisões informadas nas organizações. Segundo definição da consultoria Gartner ([s.d.]), o conceito de BI engloba um conjunto de práticas e tecnologias, como aplicações e infraestrutura, que viabilizam o acesso e a análise de informações para otimizar o desempenho e a tomada de decisão nas organizações.

Historicamente, o BI se concentrou na análise descritiva e diagnóstica, utilizando dados estruturados para gerar relatórios e dashboards que descrevem o que aconteceu e por quê. Porém, com o aumento do volume, variedade e velocidade dos dados, surgiu a necessidade de abordagens mais avançadas. O *Data Analytics* expandiu esse escopo, incorporando análise preditiva e prescritiva, utilizando técnicas de Inteligência Artificial (IA) e aprendizado de máquina para prever tendências futuras e recomendar ações.

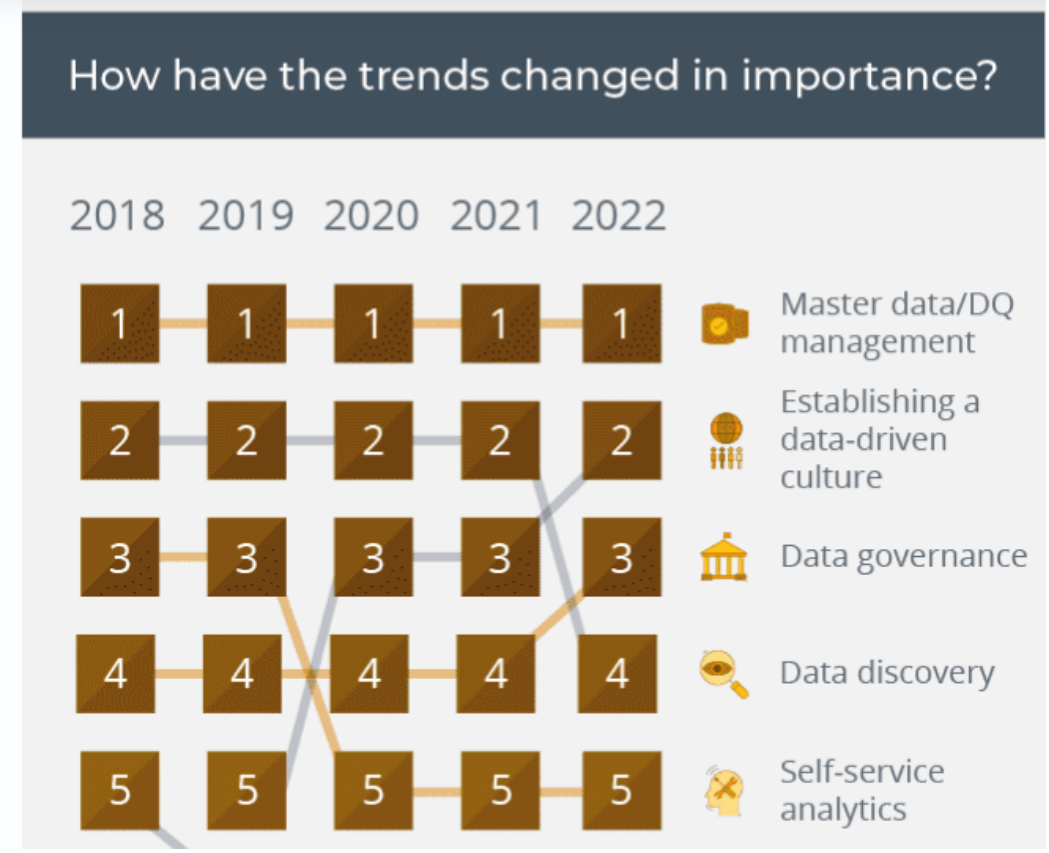
Essa transição ampliou o escopo do BI para além do papel informativo, integrando-o com o campo de *Business Analytics* (BA) e ciência de dados, em que algoritmos de mineração de dados, aprendizado de máquina e inteligência artificial passaram a complementar as plataformas de inteligência de negócios na geração de insights aprofundados.

Outra mudança importante no panorama é a disseminação do chamado BI de autosserviço (*self-service BI*). Diferentemente do modelo tradicional – no qual as análises e relatórios eram desenvolvidos predominantemente por equipes de Tecnologia da Informação (TI) ou analistas especializados – o *self-service BI* capacita os usuários de negócio a eles próprios explorarem os dados e criarem análises, com mínima dependência de técnicos. O objetivo do *self-service BI* é: *"empoderar usuários casuais a realizar análises personalizadas*

e derivar informações acionáveis a partir de grandes volumes de dados multifacetados, sem precisar envolver especialistas de BI" (Alpar; Schulz, 2016, p. 151).

Esse empoderamento dos usuários finais tem sido viabilizado por ferramentas cada vez mais interativas e amigáveis, e reflete uma tendência destacada na última década. Pesquisas contemporâneas confirmam a relevância do *self-service*: por exemplo, o relatório *"Data, BI & Analytics Trend Monitor 2022"* do *Business Application Research Centre* (BARC), resumido na Figura 1, indicou que a implementação de BI de autosserviço manteve-se entre as cinco principais tendências em BI nos últimos cinco anos, evidenciando a busca das organizações por maior flexibilidade e rapidez na obtenção de insights.

Figura 1 - "Como as tendências mudaram em importância?", da BARC que indica o BI de autosserviço como a quinta maior tendência no mundo de dados e a sua importância.

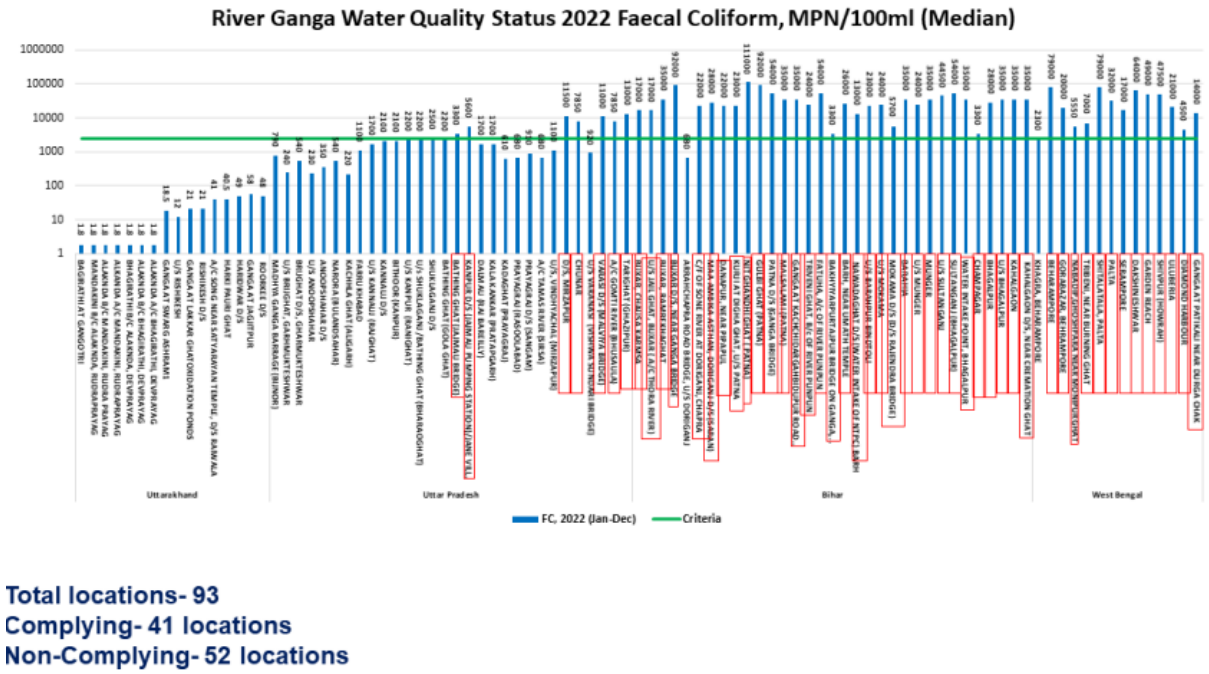


Fonte: BARC Data, BI & Analytics Trend Monitor 2022.

Um estudo de caso que ilustra a importância dos dashboards na análise de dados é o projeto da plataforma do Ministério do Meio Ambiente, Florestas e Mudança Climática da Índia, para monitoramento da poluição de ar, água e sonora em cidades indianas, desenvolvido pelo Conselho Central de Controle de Poluição da Índia (CPCB). Utilizando uma série de dashboards interativos, o projeto consegue capturar dados em tempo real sobre a qualidade do

ar, água e som em diversas cidades, oferecendo uma visão abrangente para governos e cidadãos. A interface amigável e intuitiva permite que usuários explorem os dados por diferentes parâmetros, como localização, tempo e tipo de poluente, o que tem sido fundamental para a implementação de políticas públicas de controle de poluição e aumento da conscientização pública (Central Pollution Control Board, 2021).

Figura 2- Qualidade da água do Rio Ganga durante o período de Jan-Dez de 2022. Coliformes Fecais, MPN/100ml.



Fonte: CPCB Annual Report 2022-23, p. 24, Fig. 5.4.

Na figura 2, é possível verificar um gráfico gerado pelo CPCB que auxilia no entendimento da qualidade da água do Rio Ganges em diversas localidades banhadas por este. No gráfico, é possível observar a linha que delimita o critério em verde, as localidades que não cumprem com o critério de MPN/100 ml (Número Mais Provável por 100 ml), destacadas em vermelho e corroborando com suas barras azuis acima da linha delimitadora e, por fim, 3 Key Performance Indicators (KPIs) indicando o total de localidades e quantas cumprem e quantas não cumprem com o critério de MPN/100 ml.

Em suma, o Business Intelligence contemporâneo se configura como um domínio dinâmico e em constante evolução. Seu propósito central permanece o apoio à decisão baseada em fatos, mas os meios para atingi-lo tornaram-se mais sofisticados e abrangentes. Da Extração a Transformação e Carregamento (ETL) rigoroso que alimenta os dados confiáveis, passando

pela integração de análises preditivas/prescritivas e pela adoção de ferramentas flexíveis (inclusive de código aberto), o BI evoluiu de um enfoque puramente descritivo para um sistema inteligente de análise capaz de explicar o passado, prever o futuro e orientar ações, tudo isso com maior participação dos usuários finais no processo analítico.

1.3 Objetivos

1.3.1 Objetivos Gerais

Desenvolver um dashboard interativo utilizando Python e o framework Dash, com arquitetura escalável, para a análise integrada de dados multidisciplinares, focando em usabilidade, performance e acessibilidade, disponibilizando-o em plataforma *cloud.*

1.3.2 Objetivos Específicos

1. Implementar filtros em cascata para seleção dinâmica de variáveis, anos e municípios.
2. Criar visualizações interativas (gráficos de barras, linhas e *boxplots*) para análise de tendências e anomalias.
3. Incorporar cálculos estatísticos, como detecção de outliers e análise de correlação, para suportar análises robustas.
4. Desenvolver uma interface responsiva, adaptável a diferentes dispositivos.
5. Gerenciar o desenvolvimento com versionamento no GitHub.
6. Implantar o dashboard em ambiente produtivo em nuvem.

1.4 Estrutura do Trabalho

O trabalho está organizado em cinco capítulos:

- **Capítulo 2**: Fundamentação teórica sobre Engenharia de Dados, Estatística Aplicada, Detecção de *Outliers*, Correlação e Usabilidade.
- **Capítulo 3**: Etapas de Desenvolvimento, metodologia, materiais, tecnologia utilizada, framework, defendendo escolhas técnicas, estatísticas, visuais e funcionais.
- **Capítulo 4**: Resultados, limitações e dificuldades.
- **Capítulo 5**: Conclusão, reflexões sobre objetivos e recomendações futuras.

2. FUNDAMENTAÇÃO TEÓRICA

Para o desenvolvimento de uma aplicação analítica robusta, é imperativo fundamentar o projeto em conceitos consolidados da ciência de dados e da engenharia de software. Este capítulo aborda os pilares teóricos que sustentam o dashboard: o processo de tratamento de dados (ETL), as métricas estatísticas essenciais para a geração de *insights*, e o padrão de arquitetura de software Model-Viewer-Controller (MVC) que garante a manutenibilidade do sistema.

2.1 O Processo ETL: Da Extração à Transformação de Dados

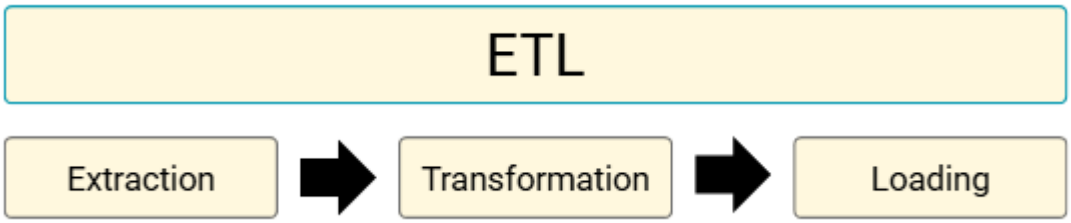
2.1.1 Engenharia de Dados e Processo ETL

A Engenharia de Dados é uma disciplina da Ciência de Dados voltada para a coleta, tratamento e armazenamento de grandes volumes de dados, de forma a transformá-los em informações valiosas para as organizações. Em termos práticos, isso envolve o desenvolvimento e manutenção de *pipelines* de dados robustos, capazes de extrair dados brutos de múltiplas fontes, aplicar procedimentos de limpeza e transformação, e então armazenar os dados processados em repositórios adequados (como *data warehouses*) para futura análise. Esse trabalho é fundamental para viabilizar análises de qualidade, pois garante que os dados cheguem aos analistas em um formato consistente e confiável.

Dentro desse contexto, destaca-se o processo ETL como etapa crucial no preparo de dados para análise. De acordo com Kimball e Ross (2013), ele consiste em um paradigma padrão para aquisição e preparação dos dados, tornando-os prontos para exposição às ferramentas de Business Intelligence. Esse processo preenche a lacuna entre os sistemas transacionais de origem (onde os dados são gerados) e a camada de análise, consolidando e refinando as informações para uso gerencial.

Tipicamente, o ETL envolve três fases sequenciais que podem ser observadas na Figura 3: **(i)** Extração dos dados de diversas fontes (muitas vezes heterogêneas, incluindo bancos de dados operacionais legados, planilhas, logs, etc.); **(ii)** Transformação, na qual são aplicadas regras de negócio, limpeza, padronização e integração dos dados para melhorar sua qualidade e consistência; e **(iii)** Carga dos dados transformados em um sistema de destino, normalmente um *data warehouse* ou outro repositório analítico (Khan; Jan; Khan; Chughtai ,2024).

Figura 3 – Etapas do processo de ETL do inglês (Extração, Transformação e Carregamento).



Fonte: Autoria Própria.

2.1.2 Estatística Aplicada

As medidas descritivas são ferramentas fundamentais para resumir e caracterizar conjuntos de dados. Entre as medidas de tendência central, destacam-se a média aritmética (soma dos valores dividida pelo número de observações) e a mediana (valor central quando os dados são ordenados). A média é útil por utilizar todos os valores, mas é sensível a valores extremos (outliers); de fato, o que significa que dados muito altos ou muito baixos podem distorcer significativamente a média, enquanto a mediana permanece mais robusta nesses casos. Como complemento, a moda representa o valor mais frequente, mas pode ser pouco informativa em distribuições contínuas ou multimodais (Medri, 2011).

Para quantificar a dispersão dos dados em torno de uma medida de centralidade, utilizam-se medidas como a variância e o desvio padrão. O desvio padrão (σ) é definido como a raiz quadrada da variância e indica o espalhamento médio dos dados em relação à média. Assim, valores de desvio padrão maiores indicam dados mais dispersos. No entanto, o desvio padrão também é influenciado por valores atípicos; em comparação com medidas mais robustas como o intervalo interquartil. Isso limita sua adequação em distribuições muito assimétricas ou com outliers, casos em que é recomendável utilizar estatísticas robustas ou transformar os dados (Medri, 2011).

Outra medida descritiva importante é a curtose, que relaciona a forma da distribuição com a concentração de dados nas caudas e pico. A curtose tradicional (excesso de curtose) é 0 para a distribuição normal, sendo classificadas distribuições como leptocúrticas quando possuem caudas pesadas (excesso de curtose positivo) e platocúrticas quando têm caudas leves (excesso negativo). Em termos práticos, uma distribuição com alta curtose apresenta maior probabilidade de valores extremos (*outliers*) do que a normal, enquanto uma de baixa curtose tem caudas mais curtas do que a normal. (Menon, 2025).

2.1.3 Detecção de Outliers e Intervalo Interquartil (IQR)

Outliers são observações que se diferenciam marcadamente do restante dos dados e podem sinalizar variabilidade natural, erros de medição ou fenômenos não previstos pelo modelo. Identificar *outliers* é crucial porque eles podem distorcer estimativas estatísticas (como média e variância) e violar pressupostos de métodos analíticos. Entretanto, a decisão de eliminar ou manter outliers deve ser tomada com cautela; a remoção indiscriminada é controversa, especialmente em amostras pequenas ou sem justificativa clara de erro experimental (Tukey, 1977).

Um método clássico e não-paramétrico para detectar outliers é o Intervalo Interquartil (IQR), popularizado por John Tukey (1977) em seu livro *Exploratory Data Analysis*. Nesse método, calcula-se o primeiro e terceiro quartis (Q1 e Q3) da distribuição de dados – valores que delimitam 25% e 75% dos dados ordenados, respectivamente – e então define-se o intervalo interquartil como $IQR = Q_3 - Q_1$. A partir dele, estabelecem-se limites, frequentemente chamados de *fences*: qualquer observação abaixo de $Q_1 - k\,.IQR$ ou acima de $Q_3 + k\,.IQR$ é considerada um outlier (enquanto valores além de $3\,.IQR$ dos quartis são às vezes etiquetados como *outliers extremos*).

Essa regra de Tukey, com o fator k = 1.5, foi adotada empiricamente, sendo que ele próprio justificava de forma pragmática a escolha dizendo que *1 era muito pouco e 2 muito* (Stack Exchange, 2021, online). Por não fazer suposições distributivas e basear-se em quartis, o método IQR é robusto e pouco afetado por outliers nos cálculos iniciais. Em outras palavras, o IQR captura a dispersão central ignorando os 25% menores e 25% maiores valores, oferecendo um critério resistente a extremos para delimitar o que é “muito distante” do grosso dos dados.

Esse procedimento de detecção via IQR é visualizado no diagrama de caixa (*boxplot*), no qual os outliers aparecem como pontos fora das *fences*. As vantagens do método residem na sua simplicidade e independência de pressupostos de normalidade. Conforme discutido na comunidade Stack Exchange (2021), o método do Intervalo Interquartil (IQR) para detecção de *outliers* não pressupõe a normalidade dos dados; pelo contrário, sua principal vantagem é a robustez, pois não é influenciado por valores extremos. Contudo, existem limitações: em distribuições muito assimétricas, a aplicação direta de $1{,}5\,.IQR$ pode marcar uma quantidade excessiva de pontos em uma cauda e não detectar na outra.

Em resumo, a Estatística Aplicada fornece um conjunto de medidas descritivas para condensar informações e métodos para detecção de outliers com embasamento matemático. Cada medida possui um fundamento analítico por exemplo, a média minimiza o erro quadrático total e o desvio padrão é a raiz da média dos quadrados dos desvios, e pressupostos implícitos.

2.1.4 Correlação

Correlação é a técnica estatística que quantifica o grau e a direção da associação entre duas variáveis. Em termos simples, um coeficiente de correlação procura expressar se, e em que medida, duas variáveis tendem a variar conjuntamente – seja aumentando ou diminuindo em conjunto (correlação positiva), seja uma aumentando enquanto a outra diminui (correlação negativa), ou não exibindo qualquer padrão consistente (correlação próxima de zero). É crucial notar que correlação não implica causalidade; duas variáveis podem apresentar correlação alta devido a influências de uma terceira variável ou pura coincidência (correlações espúrias), como no clássico exemplo do número de sorvetes vendidos e o número de afogamentos, ambos aumentam no verão, mas um não causa o outro (Schober; Schwarte, 2018).

Existem diferentes tipos de coeficiente de correlação, adequados a diferentes naturezas de dados e relações: a Correlação de Pearson mede associações lineares em dados intervalares/razão; a Correlação de Spearman mede associações monótonas (não necessariamente lineares) com base em postos; e a Correlação de Kendall foca em concordâncias em ordem entre pares de observações, sendo também uma medida não paramétrica ordinal. A seguir, discutiu-se cada uma em detalhe, com seus pressupostos, fórmulas e limitações, e foram apresentados estudos relevantes que comparam esses métodos.

2.1.5 Correlação de Pearson (Produto-Momento)

O coeficiente de correlação de Pearson, também conhecido como ***r* de Pearson**, foi proposto por Karl Pearson no fim do século XIX e é definido pela razão entre a *covariância* de duas variáveis (X e Y) e o produto de seus desvios padrão. Sua fórmula populacional é dada pela equação (1):

$$r = \frac{\sum(X-\bar{X})(Y-\bar{Y})}{\sqrt{\sum(X-\bar{X})^2.\sum(Y-\bar{Y})^2}} \qquad (1)$$

onde $\bar{X}$ *e* $\bar{Y}$ são as médias de X e Y. Esse coeficiente varia de -1 a +1, indicando correlação perfeitamente negativa (-1), nula (0) ou perfeitamente positiva (+1). Pearson *r* mede especificamente o grau de relação linear entre as variáveis, ou seja, quão próximos os pontos $(x_i\ , y_i)$ estão de uma linha reta no plano cartesiano.

Para que a inferência paramétrica com r de Pearson seja válida (teste de significância, intervalos de confiança), requer-se que: (1) as variáveis X e Y sejam aproximadamente continuamente distribuídas (intervalares ou racionais); (2) a relação entre elas seja linear (isto é, a nuvem de pontos segue aproximadamente uma reta); (3) idealmente, que X e Y sigam conjuntamente uma distribuição normal bivariada (pressuposto mais forte, necessário para testes exatos) (Schober; Schwarte, 2018); e (4) ausência de outliers relevantes. Na prática, discute-se que a normalidade perfeita não é estritamente necessária para usar *r* de Pearson, pois ele é relativamente robusto a violações de normalidade.

Por outro lado, a linearidade é fundamental: se a relação verdadeira for não linear (por exemplo, curvilínea), o coeficiente de Pearson pode ser próximo de zero enganadoramente, mesmo havendo forte dependência (nesse caso, métodos não lineares seriam necessários). Além disso, *outliers* podem influenciar drasticamente o valor de *r*, já que a covariância e o desvio padrão são ambos não robustos – um único ponto aberrante pode inflar ou reduzir *r* indevidamente (Schober; Schwarte, 2018). Por isso, recomenda-se inspecionar visualmente os dados (gráficos de dispersão) antes de interpretar a correlação de Pearson, excluindo ou tratando outliers extremos e verificando se o padrão aparenta ser linear.

2.1.6 Correlação de Spearman (por Postos)

Quando os dados violam os pressupostos de Pearson – por exemplo, contendo outliers, distribuições altamente assimétricas, ou relações que embora monotônicas não sejam lineares –, é comum recorrer ao coeficiente de **Spearman (ρ).** Proposto por Charles Spearman em 1904, esse coeficiente é essencialmente uma aplicação do coeficiente de Pearson aos ranks (posições ordenadas) dos dados (Hauke; Kossowski, 2011). O processo consiste em, para cada variável (X e Y), substituir os valores originais por suas respectivas posições ordinais, exemplo, o menor valor recebe o *rank* 1, o segundo menor o *rank* 2, e assim por diante. O coeficiente ρ é então o coeficiente de Pearson calculado sobre estas novas variáveis ranqueadas, R(X) e R(Y).

Sua forma mais conhecida, aplicável quando não há empates nos dados, é a fórmula de cálculo baseada na diferença dos postos, como pode ser visto nas equações (2) e (3):

$$\rho_s = \frac{\sum_{i=1}^{n}(R(x_i)-\overline{R(x)})(R(y_i)-\overline{R(y)})}{\sqrt{\sum_{i=1}^{n} R(x_i)-\overline{R(x)})^2.(R(y_i)-\overline{R(y)})^2.\sum_{i=1}^{n}(R(y_i)-\overline{R(y)})^2}} = 1 - \frac{6\sum_{i=1}^{n}(R(x_i)-R(y_i))^2}{n(n^2-1)} \quad (2)$$

Logo,

$$\rho_s = \frac{\sum d_i^2}{n(n^2-1)} \qquad (3)$$

onde d_i é a diferença entre os postos de X e Y para o mesmo i-ésimo indivíduo, e *n* é o tamanho da amostra (Stack Exchange, 2020, online). Em caso de empates, utiliza-se a definição via Pearson nos ranks com correção de postos médios.

Em suma, a correlação de Spearman oferece uma alternativa confiável ao coeficiente de Pearson diante de violações de pressupostos, expandindo a análise de correlação para um espectro mais amplo de situações.

2.1.7 Correlação de Kendall (Tau de Kendall)

O coeficiente **Tau (τ) de Kendall** é outra medida não-paramétrica de associação ordinal, proposta por Maurice Kendall em 1938. Conceitualmente, 'τ' avalia a concordância de ordenação entre duas variáveis: para todos os possíveis pares de observações (i, j), verifica-se se as diferenças em X e Y têm o mesmo sinal (par *concordante*) ou sinais opostos (par *discordante*). O tau é definido pela equação (4):

$$\boldsymbol{\tau_a = \frac{C-D}{C+D}} \qquad (4)$$

onde C é o número de pares concordantes e D o número de pares discordantes. Intuitivamente, 'τ' aproxima a probabilidade de concordância menos a probabilidade de discordância entre duas observações tomadas ao acaso. Assim, τ = 1 indica que todas as comparações de pares são concordantes (ordens totalmente alinhadas), τ = -1 indica ordens exatamente opostas, e τ = 0 corresponde à independência em termos de ordem (metade das vezes concorda, metade discorda, como esperado ao acaso) (Brideau, 2025).

Tanto τ de Kendall quanto ρ de Spearman são medidas de correlação ordinal e tendem a concordar qualitativamente (mesmo sinal, relação monotônica similar). Entretanto, τ costuma ter magnitude menor que ρ_s para um mesmo grau de associação, porque é uma medida mais direta de probabilidade de concordância (Stack Exchange, 2020).

Uma regra empírica é que $\tau \cong \frac{2}{3}\rho_s$ em muitas situações de dados contínuos, embora não haja equivalência exata. Em termos de eficiência estatística sob normalidade bivariada, a τ de Kendall tem cerca de 0,91 da eficiência de ρ_s (e ~0,65 da eficiência de *r* de Pearson) – em outras palavras, para pequenas amostras, Spearman pode detectar correlação monotônica com um pouco mais de poder que Kendall (Stack Exchange, 2020). Por outro lado, τ tem uma

interpretação mais direta: τ corresponde a uma diferença de probabilidades $n_c - n_d$, onde $n_c = Pares\ concordantes\ e\ n_d = Pares\ discordantes$. Isso significa, por exemplo, que τ = 0,5 indica que concordâncias superam discordâncias em 50% dos pares, o que é um entendimento tangível do grau de associação – essa clareza interpretativa é frequentemente citada como vantagem conceitual de Kendall sobre Spearman.

No geral, a inclusão de τ de Kendall nesta discussão reforça a ideia de que correlação não é um conceito único: há diferentes maneiras de quantificar associação, cada qual com hipóteses matemáticas e interpretações distintas. Assim, um pesquisador deve selecionar o coeficiente apropriado considerando a distribuição dos dados e a forma esperada da relação, pois é fundamental que compreendam não apenas como aplicar os coeficientes de correlação, mas também, como alertam Schober, Boer e Schwarte (2018), como evitar seu uso e interpretação incorretos.

2.1.8 Arquitetura de Software

No contexto de desenvolvimento de aplicações interativas – como dashboards desenvolvidos em Python com Dash – a arquitetura de software define a forma como o sistema é modularizado e como seus componentes interagem. Boas arquiteturas promovem separação clara de responsabilidades, facilitando a manutenção, a escalabilidade e o trabalho colaborativo de equipes de desenvolvimento. Neste contexto que surge o Model-View-Controller (MVC) e suas variantes, como Model-View-ViewModel (MVVM), Model-View-Presenter (MVP.

O padrão MVC é possivelmente o mais conhecido paradigma de arquitetura para interfaces de usuário e foi pioneiro na separação entre dados, interface e lógica de controle. Seu surgimento remonta ao final da década de 1970 por Trygve Reenskaug, durante o desenvolvimento do ambiente de programação Smalltalk-80 (Trygve, 2003). A motivação original do MVC era tornar as interfaces gráficas mais fáceis de desenvolver e manter, criando componentes modulares que refletissem a maneira como usuários pensam sobre a aplicação (modelos mentais do domínio) e permitindo múltiplas visualizações e formas de interação sobre os mesmos dados.

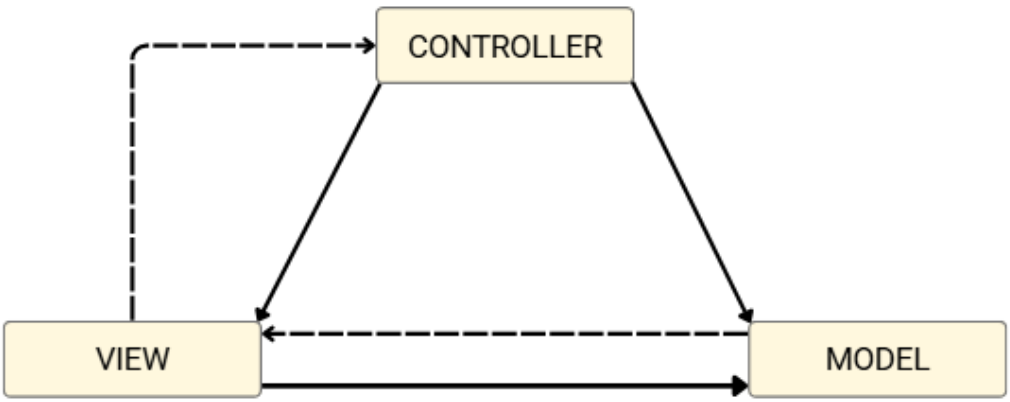
Em essência, o MVC propõe dividir a aplicação em três componentes principais, cada um com responsabilidades bem definidas resumidas na figura 4:

- ***Model*** **(Modelo):** representa os “dados” da aplicação e as regras de negócio ou lógica de domínio. O modelo encapsula o estado do aplicativo, por exemplo em um dashboard, o modelo poderia ser um conjunto de objetos ou estruturas contendo os dados

carregados e talvez a lógica de agregação ou cálculo. Ele é independente de como esses dados são apresentados ao usuário. Idealmente, o modelo pode ser testado isoladamente e reutilizado em diferentes interfaces.

- ***View* (Visão ou Visor):** corresponde à "interface de usuário" propriamente dita, as telas, elementos gráficos, ou componentes de apresentação que o usuário vê e com os quais interage. A *View* tem a responsabilidade de obter do *Model* os dados (através do *Controller* ou via mecanismos de *binding*, a depender da implementação) e exibi-los de forma adequada. Uma *View* deve refletir as mudanças nos dados do Model imediatamente, seguindo o princípio de sincronização entre modelo e visão.
- ***Controller* (Controle):** atua como o "intermediário" que coordena a interação entre *View* e *Model*. Ele recebe as entradas do usuário na *View* (eventos como cliques, teclas pressionadas, etc.) e as traduz em operações sobre o *Model* ou na própria *View*. Em suma, o *Controller* interpreta as intenções do usuário e aciona as mudanças necessárias – por exemplo, ao clicar um botão de filtro no dashboard, o *Controller* recebe esse evento e pode solicitar ao *Model* que atualize os dados filtrados, e então informa a *View* para se atualizar. No MVC original do *Smalltalk*, o *Controller* também era responsável por algumas funcionalidades de edição e interação direta, mas em implementações modernas web, ele costuma ser a camada que lida com requisições e decide quais dados prover à *View*.

Figura 4 - Um diagrama simples exemplificando a relação entre *Model*, *View* e *Controller*. As linhas sólidas indicam associação direta e as tracejadas indicam associação indireta.



Fonte: Autoria Própria.

2.1.9 Usabilidade e UI/UX

A qualidade de uma aplicação interativa não depende apenas de sólidos fundamentos estatísticos ou de uma boa arquitetura de software: a Usabilidade e a Experiência do usuário (UI/UX) desempenham um papel crucial na eficácia do sistema. *UI* (*User Interface*) refere-se à facilidade com que um usuário consegue empregar a ferramenta para atingir seus objetivos, englobando critérios como eficiência, eficácia e satisfação. Já *UX* (*User Experience*) amplia o escopo para as percepções e respostas do usuário decorrentes do uso – incluindo emoções, percepção de valor, estética, etc.

Em um dashboard de dados, por exemplo, uma boa usabilidade assegura que o usuário consiga descobrir informações nos gráficos e tabelas de forma intuitiva e rápida, enquanto uma boa UX garante que essa interação seja prazerosa, confiável e agregue valor às decisões do usuário. Nesta seção, exploramos princípios clássicos de usabilidade (como as Heurísticas de Nielsen), conceitos de Design Universal (pautados na inclusão de todos os perfis de usuários) e abordagens contemporâneas de design de interfaces e visualização de dados.

2.1.10 Heurísticas de Nielsen e Design Universal de Mace

Do ponto de vista histórico, pode-se dizer que a engenharia de usabilidade consolidou-se na década de 1990 com pesquisadores como Jakob Nielsen e Rolf Molich. Nielsen, em especial, formulou um conjunto de 10 heurísticas de usabilidade que até hoje servem como diretrizes de avaliação de interfaces (publicadas inicialmente em 1994, baseadas em trabalho anterior com Molich em 1990). Essas heurísticas são princípios gerais (*"regras de bolso"*) que designers devem observar para tornar a interface mais amigável. Em termos práticos, isso significa que um dashboard deve exibir informações de forma clara, sem poluição visual ou termos técnicos confusos, utilizando rótulos que façam sentido no domínio do usuário.

Apesar de formuladas há quase 30 anos, continuam relevantes inclusive para interfaces modernas (web, mobile). Por exemplo, a heurística de visibilidade do status implica que, em um dashboard, qualquer processo de carregamento de dados ou filtro em aplicação deve fornecer feedback visual imediato (um ícone de progresso, mensagem "Carregando...") para que o usuário saiba que o sistema está ativo – caso contrário, ele pode ficar incerto ou repetir ações desnecessariamente. Da mesma forma, consistência é vital: uso consistente de cores, ícones e interações significa que o usuário não precisa reaprender operações em diferentes partes do sistema. Essas diretrizes refletem conhecimentos de psicologia cognitiva, como as limitações da memória de trabalho humana e a propensão a erros, daí a ênfase em *reconhecimento em vez de lembrança* e *prevenção de erros*.

Paralelamente à preocupação com usabilidade geral, surgiu a noção de Design Universal, cunhada pelo arquiteto Ronald Mace na metade dos anos 1980. O Design Universal (DU) preconiza que produtos e ambientes devem ser concebidos, desde o início, para serem usáveis pelo maior espectro de pessoas possível, sem necessidade de adaptações posteriores. Mace definiu o objetivo do DU como *"desenvolver produtos e ambientes para serem utilizáveis por todas as pessoas, na maior extensão possível, sem necessidade de adaptação ou design especializado"* (Bandeira; Rocha; Santana, 2018). Em 1997, um grupo de pesquisadores da Universidade Estadual da Carolina do Norte (que inclui Mace) formalizou os "Sete Princípios do Design Universal":

1. **Uso Equitativo:** o design deve ser útil e comercializável para pessoas com diversas habilidades. Isso significa que a interface não deve excluir nenhum grupo – por exemplo, informações não podem depender apenas de cor (pois usuários daltônicos teriam dificuldade), e funcionalidades principais não devem exigir, digamos, precisão de clique que exclua quem tem limitações motoras. Equitativo também implica que, quando possível, todos os usuários usam a mesma interface básica em igualdade (evitando segregar usuários com deficiência em interfaces "especiais").

2. **Flexibilidade no Uso:** o design acomoda uma ampla gama de preferências e habilidades individuais. Ou seja, oferecer opções de interação, por exemplo: permitir tanto uso do mouse quanto do teclado, suportar tanto usuários destros quanto canhotos, personalização de tamanho de fonte, etc. Isso torna a interface adaptável a diferentes necessidades. Em um dashboard, flexibilidade pode significar permitir que o usuário reordene componentes conforme sua preferência ou escolha entre modos de visualização claros e escuros.

3. **Uso Simples e Intuitivo:** a interface deve ser fácil de entender, independentemente da experiência do usuário, conhecimento, habilidades de linguagem ou nível de concentração no momento. Os processos devem seguir lógicas claras e familiares. Por exemplo, controles importantes devem ser destacados e nomeados de forma intuitiva (um botão de exportar dados usando o ícone de uma seta saindo de uma caixa, acompanhado do texto "Exportar", em vez de algo ambíguo).

4. **Informação Perceptível:** o design comunica efetivamente as informações ao usuário, independentemente de condições ambientais ou capacidades sensoriais do usuário. Isso requer redundância de modos (visual, auditivo, tátil se aplicável) e bom contraste visual.

Por exemplo, textos legíveis (contraste adequado com o fundo, tamanho de fonte suficiente), indicadores sonoros complementares a alertas visuais para usuários com baixa visão, legendas em vídeos para usuários com deficiência auditiva, entre outros. Em um dashboard, um princípio aplicado seria garantir que cores em gráficos tenham contrastes suficientes e talvez padrões ou rótulos diretos em pontos de dados para não depender só da cor.

5. **Tolerância a Erros:** o design minimiza riscos e as consequências adversas de ações acidentais. Isso se alinha à heurística de Nielsen de prevenção de erros: incluir confirmações em ações destrutivas (como "Tem certeza de que deseja deletar este relatório?"), desfazer/recuperar fácil (função *undo*), e evitar colocação de elementos perigosos próximos de elementos comuns (para não clicar errado). Em um contexto de dashboard interativo, por exemplo, se o usuário pode redefinir todo o dashboard, talvez pedir uma confirmação ou ter como recuperar o estado anterior evita frustração por cliques acidentais.

6. **Baixo Esforço Físico (Conforto):** o design pode ser usado eficientemente e confortavelmente, com um mínimo de fadiga. Isso aplica-se mais a dispositivos físicos, mas em interfaces considera-se reduzir interações desnecessárias. Por exemplo, navegação enxuta – não obrigar o usuário a clicar ou arrastar excessivamente. Interfaces que exigem muitos cliques para tarefas rotineiras acabam prejudicando a experiência. Em dashboards, permitir filtrar múltiplos itens com poucos cliques (caixas de seleção múltipla, seleção de intervalo de datas facilitada, etc.) aumenta o conforto.

7. **Tamanho e Espaço para Aproximação e Uso:** disposição e tamanho apropriados para acesso, manipulação e uso, independentemente do tamanho do usuário, postura ou mobilidade. Em UI, isso se traduz em elementos clicáveis suficientemente grandes (botões que não sejam minúsculos a ponto de serem difíceis de selecionar, especialmente em touch screens), espaçamento adequado para evitar cliques em elementos errados, e layout adaptável para diferentes dispositivos. De fato, o design *responsivo* que se adapta a telas de tamanhos variados (desktop, tablet, celular) é uma aplicação desse princípio – garantindo que pessoas usando um celular (muitas vezes com uma mão só) consigam interagir tão bem quanto quem usa no computador.

Esses sete princípios formam um arcabouço para avaliar e guiar o design centrado na acessibilidade e inclusão. Para orientar a prática do Design Universal, cada um de seus

princípios foi desdobrado em diretrizes específicas que servem para guiar tanto o processo de avaliação quanto o de desenvolvimento de projetos (Bandeira; Rocha; Santana, 2018).

Cabe ressaltar que seguir DU não beneficia apenas pessoas com deficiência permanente, mas melhora a experiência para todos – por exemplo, legendas em vídeos ajudam quem está em ambiente barulhento; botões maiores facilitam cliques para usuários idosos e para quem tem mãos grandes ou está usando um dispositivo móvel em movimento. Essa é a beleza do design universal: ao projetar *para todos*, obtém-se um design melhor no geral.

2.1.11 Design Centrado no Usuário e Visualização de Dados

Enquanto heurísticas e princípios universais fornecem diretrizes gerais, o Design Centrado no Usuário (DCU) ou *Human-Centered Design* propõe um processo de desenvolvimento iterativo pautado pelas necessidades reais dos usuários finais. No DCU, desde o planejamento, envolvem-se usuários ou representantes para entender seus objetivos, contextos e *pain points*; então protótipos são criados e avaliados com usuários, refinando-se o design em ciclos sucessivos.

Abras, Maloney-Krichmar e Preece (2004, p.763-767) definem DCU como um *"termo abrangente que descreve processos de design nos quais os end-users influenciam ativamente como um sistema toma forma"*, enfatizando a incorporação do conhecimento do mundo real dos usuários no design. Em outras palavras, o produto final deve espelhar as expectativas, linguagem e fluxos de trabalho dos usuários, e não forçar o usuário a se adaptar à lógica interna dos desenvolvedores.

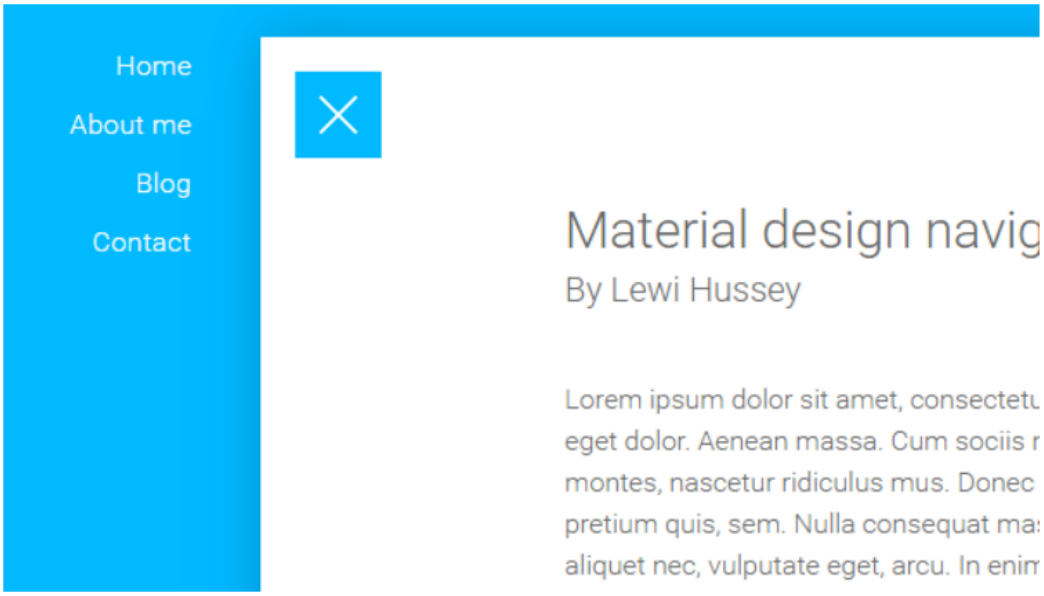
No contexto de dashboards interativos, aplicar DCU significa, por exemplo: realizar levantamentos com futuros usuários (analistas, gestores etc.) para descobrir quais métricas eles realmente precisam, quais tarefas executam com os dados, quais terminologias usam. Em seguida, construir protótipos (mesmo que em papel ou *wireframes*) e submeter a usuários para feedback – observar se conseguem encontrar as informações, se algo os confunde. Iterar conforme esse feedback.

Muitas vezes, designers internos têm suposições diferentes dos usuários; só envolvendo-os é possível alinhar. Além disso, DCU possui princípios éticos de respeito e inclusão – garantir que as vozes de minorias de usuários sejam ouvidas, evitar vieses. Um dos preceitos do Design Centrado no Usuário é a criação de modelos conceituais que sejam consistentes e alinhados com o conhecimento de mundo que o próprio usuário possui (Pagan et al., 2019), ou seja, em vez de estruturar o dashboard espelhando o banco de dados ou a API (que reflete a visão do

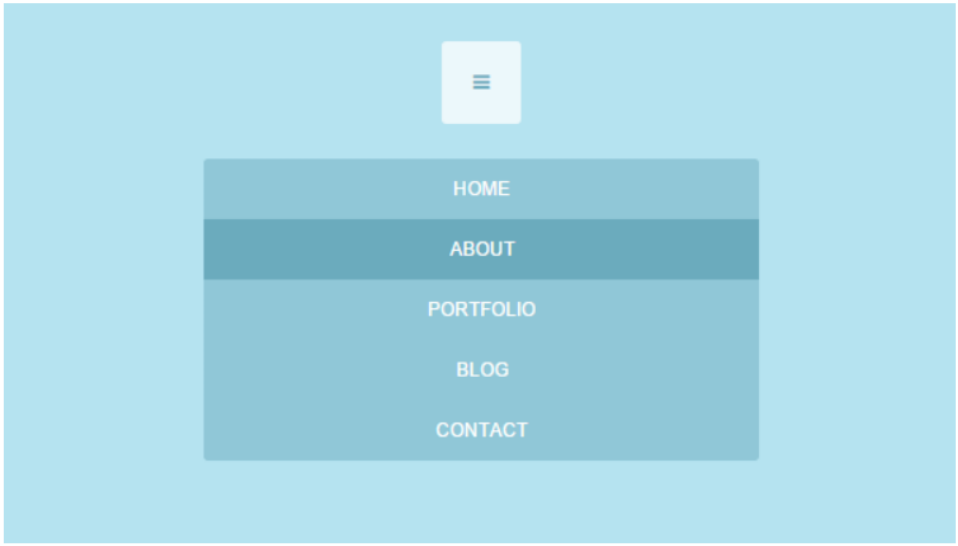
desenvolvedor), estruturá-lo conforme a lógica do negócio que o usuário entende (por exemplo, se o usuário pensa em “vendas por região” e “vendas por produto” como coisas distintas, a interface deveria refletir essas categorias claramente, ao invés de exigir que ele componha manualmente filtros por códigos de região ou produto).

Outro conceito relevante é o Design Responsivo e multiplataforma: atualmente espera-se que dashboards funcionem em diferentes dispositivos. Interfaces responsivas, conceito popularizado por Ethan Marcotte (2010), adaptam o layout ao tamanho de tela. Esse cuidado faz parte da UX – se um gestor abre o dashboard no celular, ele deve conseguir ver o essencial facilmente (talvez através de um layout simplificado móvel), mantendo funcionalidade. Um exemplo claro disso está na representação de um menu em formato longo quando aberto em *desktop* e em formato compacto quando em *mobile*, Figuras 5 (a) e (b), respectivamente.

Figura 5 – Ilustração de um Menu de dashboard na versão para *desktop*, com menu de navegação lateral ao conteúdo apresentado (a) e para *mobile*, com o menu centralizado em forma de dropdown (b).



(a)



(b)

Fonte: BRASILCODE [S.d.].

Por fim, “*feedback* visual” e “*feedback* de sistema” merecem destaque: heurísticas de Nielsen enfatizam "visibilidade do estado do sistema" e "*feedback* imediato". Em dashboards interativos, isso significa: após o usuário aplicar um filtro ou clicar num ponto do gráfico para ver detalhes, o sistema deve reagir rapidamente ou pelo menos indicar que está processando. De acordo com a literatura de Norman (2013) sobre psicologia, as pessoas esperam respostas em frações de segundo para sentir fluidez; podemos inferir, pelas situações descritas na literatura, que atrasos de mais de 1 segundo começam a quebrar a sensação de controle, e acima de 10 segundos a atenção se perde. Portanto, otimizações de desempenho não são apenas questões técnicas, mas de UX.

Concluindo, boa usabilidade e design centrado no usuário não são elementos acessórios, mas parte integral do sucesso de sistemas de análise de dados interativos. Uma UI bem projetada amplifica o valor da análise, ou seja, usuários conseguem explorar dados mais profundamente, cometem menos erros de interpretação e adotam a ferramenta com entusiasmo ao invés de resistência. Como sintetiza a perspectiva moderna em HCI (Interação Humano-Computador), discutida por Andreas Holzinger e Martina Ziefle em 2010, pesquisadores atuantes em HCI e usabilidade em saúde, a interface é a ponte entre a mente humana e os dados e uma ponte sólida, estável e bem sinalizada leva a travessias melhores.

3. DESENVOLVIMENTO

3.1 Etapas de Desenvolvimento

O desenvolvimento foi dividido em cinco iterações principais, cada uma correspondendo a um conjunto de funcionalidades após o tratamento dos dados, gerenciadas em um repositório GitHub para versionamento e controle de versões (Simoe, 2025). O GitHub foi essencial para organizar o código, utilizando *branches* para desenvolvimento paralelo (ex.: feature/filtros-cascata, feature/correlação) e *tags* para marcar entregáveis estáveis (ex.: v1.0-layout, v2.0-graficos). *Commits* descritivos (ex.: "Implementa *dropdowns* dinâmicos para filtros") garantiram rastreabilidade, permitindo experimentações sem comprometer a versão principal (Dingsøyr; Moe, 2010).

1. **Iteração 1: Estrutura Básica e Navegação**:
    - Implementação do layout com Dash, incluindo uma navbar para dispositivos móveis e uma sidebar para desktops (branch: feature/layout).
    - Criação das seções (Emissão de Gases, Epidemiológicos, Socioeconômicos, Predição, Correlação) com navegação via dcc.Location (tag: v1.0-layout).
    - Integração de *dash-bootstrap-components* para responsividade.
2. **Iteração 2: Filtros em Cascata**:
    - Desenvolvimento de *dropdowns* dinâmicos para variáveis, anos e municípios, com lógica hierárquica (*branch*: feature/filtros-cascata).
    - Uso de *callbacks* para atualizar opções com base nos dados carregados via Pandas (*tag*: v1.1-filtros).
    - Implementação inicial de cache com *dcc.Store* para otimizar o carregamento de filtros.
3. **Iteração 3: Visualizações e Estatísticas**:
    - Integração de gráficos Plotly (barras, linhas, *boxplots*) para exibir dados filtrados por ano e município (*branch*: feature/graficos).
    - Cálculo de estatísticas descritivas (média, mediana, desvio padrão, curtose) e cercas de outliers com base no método IQR (*tag*: v2.0-graficos).

- Adição de tabelas responsivas para exibir outliers, utilizando *dash-bootstrap-components*.

4. **Iteração 4: Análise de Correlação**:
   - Criação de uma seção dedicada à correlação, com *dropdowns* para seleção de seção, variáveis, anos, municípios e método (Pearson, Spearman, Kendall) (*branch*: feature/correlação).
   - Implementação de *heatmaps* e *scatter plots* interativos com Plotly, incluindo exportação de dados em formato Excel (*tag*: v2.1-correlação).
5. **Iteração 5: Melhorias de UI/UX**:
   - Ordenação alfabética de filtros (variáveis, municípios, seções) para maior previsibilidade (*branch*: feature/ui-ux).
   - Implementação de busca sem acentos nos *dropdowns* de municípios, utilizando unicodedata para normalização de texto (*tag*: v3.0-ui-ux).
   - Ajustes nos gráficos de barras para exibição lado a lado (barmode='group') e marcação de outliers com pontos vermelhos.
6. **Iteração 6: Transformação para MVC:**
   - Modularização do código a partir da metodologia MVC mais compatível com o Dash em arquivos diferentes de Modelo, Controlador, Visualização, Configuração e Disponibilização da aplicação.
   - Criação do Modelo que detém o código de carregamento dos arquivos de uma determinada seção a partir de um caminho absoluto. das estatísticas e da normalização de texto.
   - Criação do Controle, composto por todos os callbacks reunidos em uma única função denominada *"register_callbacks"*.
   - Separação do Layout, denominado também de *Template*. Ele detém toda a lógica de construção do layout do dashboard, menos a parte de tema.
   - Alteração do arquivo app.py que agora somente faz a importação dos *callbacks*, da configuração de temas e layout, determinando qual porta deve

ser usada para rodar e se o debug deve funcionar ou não, para hospedagem local.

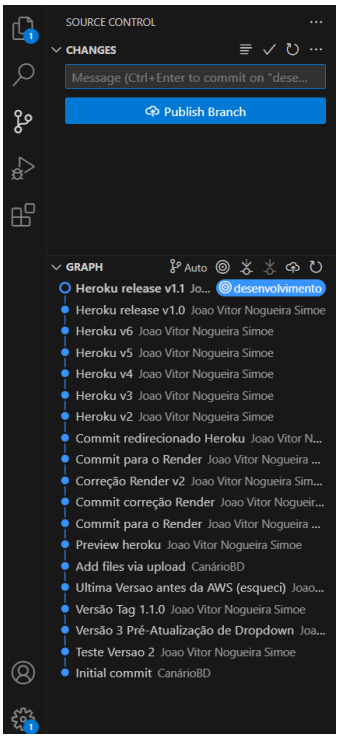
- Realização das configurações para a próxima etapa (hospedagem em nuvem).

7. **Iteração 7: Implantação em nuvem**

- Criação de uma conta no Heroku e configuração de um novo projeto para hospedagem em nuvem.
- Atualização do GitHub e conexão com Heroku.
- Realização do *deploy* e monitoramento falhas e uso de recursos.

Cada iteração foi documentada no GitHub (Simoe, 2025), como pode ser observada na Figura 6, com requisitos priorizados com base nos objetivos do projeto (ex.: usabilidade, interatividade). Essa abordagem iterativa, combinada com o controle de versão, permitiu a entrega de um sistema funcional, com refinamentos contínuos, conforme recomendado por Dingsøyr e Moe (2010).

Figura 6 – Interface de controle de fontes do Visual Studio com a origem remota do GitHub e a Branch "desenvolvimento".



Fonte: Autoria Própria.

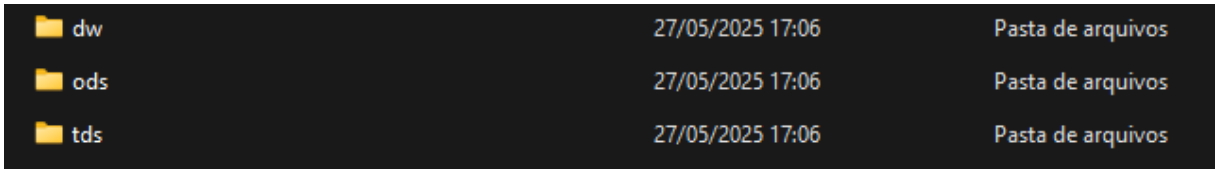
3.2 Abordagem Metodológica

O desenvolvimento do dashboard seguiu uma abordagem iterativa, inspirada em metodologias ágeis, especificamente o Scrum adaptado (Schwaber; Sutherland, 2020). Segundo Dingsøyr e Moe (2010), metodologias ágeis são ideais para projetos com requisitos dinâmicos, pois permitem ciclos curtos de desenvolvimento, teste e refinamento, incorporando feedback contínuo. No contexto deste projeto, a abordagem iterativa foi escolhida devido à complexidade de integrar múltiplas funcionalidades, como filtros dinâmicos, visualizações interativas e análises estatísticas, em uma interface coesa, portanto, não está em pauta as metodologias ágeis.

O processo foi organizado em iterações de curta duração, cada uma focada em um conjunto específico de funcionalidades (ex.: filtros, gráficos, correlação). Cada iteração incluiu planejamento, implementação, testes funcionais e refinamento, com ajustes baseados em avaliações de usabilidade e desempenho. Essa abordagem garantiu a entrega incremental do sistema, permitindo a identificação precoce de problemas e a incorporação de melhorias, como a ordenação alfabética dos filtros e a busca sem acentos para municípios.

No caso da estrutura de dados, a fim de se obter os dados de forma consistente para o uso, foi necessária a criação de duas aplicações em Python para realizar um pequeno tratamento, tornando possível a estruturação de diversas formas de modelagem dos dados a partir da construção de uma estrutura baseada no DWH, como pode ser observado na figura 7.

Figura 7 – Divisão recomendada por Kimball e Ross (2013) com o uso de pastas ODS, TDS e DWH, garantindo a separação dos dados para eventuais análises e manutenções.



Fonte: Autoria Própria.

Este tratamento visa tornar fácil a leitura das tabelas que possuíam dados de variáveis diferentes para diversos valores no campo de Município e colunas por Ano, tornando incompreensível a leitura dos arquivos. Optou-se então por uma estrutura que trazia os dados repetidos por Município das variáveis escolhidas e por Ano.

Outro ponto é que, similarmente do que seria obtido se implementado o framework do Django, no projeto foi utilizado um padrão que une as características modulares do MVC com

outra metodologia citada no capítulo 2, MVVM, trazendo uma modularização ainda maior e mais independência para os componentes. Este padrão adaptado, conta ainda com uma mudança de nomenclatura de *View* para *Template,* devido ao uso extenso do "*dashboard-components*" que traz consigo *templates* e temas pré-fabricado. Essa escolha foi motivada pela separação clara de responsabilidades, facilidade de manutenção e expansão, e código mais legível e organizado.

A grande sacada deste tipo de metodologia é a **separação de preocupações**: cada componente foca em um aspecto – dados, apresentação ou controle – minimizando sobreposições. Para alimentar os dados, trago um arquivo load_data.py, utilizado para fazer o carregamento dos dados do Excel, sua otimização de cache e todo o tratamento dos campos para filtragem e seleção das variáveis, e um arquivo model.py, que serve para trazer meus modelos de análise estatística, normalizar todo o texto carregado pelo load_data.py e fazer a ponte de busca das seções escolhidas pelo usuário na aplicação. Assim, esta estrutura corresponde ao *Model* tradicional.

Já o template.py e app.py seriam o equivalente ao *View*, sendo o primeiro responsável por armazenar toda a minha estrutura de *front-end* e suas interações com o usuário, enquanto o segundo faz a inicialização da aplicação em Dash e registra as chamadas realizadas pelo usuário, interagindo com o controlador. Por fim, meu controllers.py seria o *Controller* ou controlador, que registra todas as interações realizadas pelo usuário e utiliza de sua conexão com o model.py e template.py para enviar e receber as solicitações, ou seja, ele é o orquestrador da aplicação.

Isso traz diversas vantagens clássicas apontadas na literatura de engenharia de software. (Majeed; Rauf, 2018):

- **Manutenibilidade e Extensibilidade:** Alterações na lógica de negócio (*Model*) podem ser feitas sem afetar diretamente a interface, e vice-versa. Por exemplo, se a fonte de dados do dashboard mudar (digamos, de um banco de dados local para uma API web), apenas o *Model* precisaria ser adaptado, enquanto as *Views* e *Controllers* poderiam permanecer iguais desde que a interface do *Model* (métodos de acesso aos dados) se mantenha.
- **Reutilização de código:** Como o *Model* é independente de *View*, os mesmos dados podem ser apresentados em múltiplas interfaces (por exemplo, tabelas, gráficos,

relatórios) reutilizando a lógica. Também é possível ter diferentes *Views* para diferentes plataformas (desktop, web, mobile) operando sobre o mesmo *Model*.

- **Testabilidade:** Componentes isolados são mais fáceis de testar. Pode-se fazer testes unitários do Model (já que ele independe de UI) e testes das Views via simulação de Controllers. Controladores também podem ser testados alimentando-os com eventos simulados. Um dos testes realizados foi feito dentro do Model para validação das bibliotecas e do join no load_data.py. O intuito era verificar se os números não estavam sendo afetados pelo join, acarretando resultados equivocados na análise estatística.

Por fim, para a implementação do projeto em etapa produtiva, foram utilizados recursos do Heroku, plataforma de hospedagem de sites e aplicações, e o GitHub, plataforma de versionamento de códigos online baseada em Git. Para este trabalho, utilizou-se o Git para salvar as versões do código-fonte no servidor do GitHub a partir de diversos comandos, conectando o diretório local com o remoto nesta plataforma, podendo manter um histórico de todas as versões e alterações realizados neste projeto. Quando uma versão limpa, estivesse apta a publicação, bastava realizar o "*deploy*" do último *"commit"* criado no GitHub na plataforma do Heroku.

3.3 Materiais

3.3.2 Python e Dash

A escolha de Python como linguagem principal foi motivada por sua posição de liderança em ciência de dados, oferecendo um ecossistema robusto que integra manipulação, visualização e desenvolvimento web (Perkel, 2022). Diferentemente de linguagens como R, que são mais restritas a análises estatísticas, ou JS, que exige maior complexidade para interfaces web, Python proporciona flexibilidade e acessibilidade, sendo amplamente adotado em projetos acadêmicos e profissionais (Larson; Chang, 2020). Em suma sua escolha para o projeto se deu devido a integração de manipulação de dados (Pandas), visualização (Plotly) e desenvolvimento web (Dash) em um único ambiente, facilitando a criação de um pipeline eficiente.

O *framework* Dash, construído sobre Python, foi selecionado por sua capacidade de criar aplicativos web interativos sem a necessidade de conhecimento avançado em desenvolvimento *front-end* (Plotly, 2023). Comparado a alternativas como Streamlit, que é mais simples, mas menos personalizável, ou Flask com JS, que exige maior esforço de integração, Dash oferece um equilíbrio ideal entre funcionalidade e facilidade de uso. Segundo Zhang e Wang (2023),

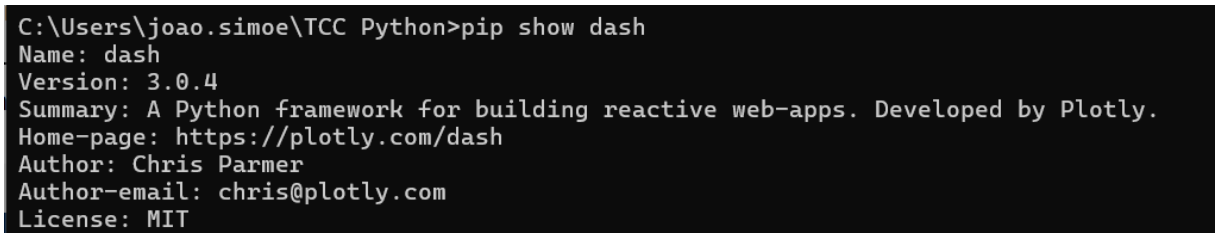
Dash é particularmente adequado para dashboards de BI, pois suporta visualizações dinâmicas e callbacks reativos, características essenciais para o projeto.

3.3.3 Plotly e Pandas

Plotly foi escolhido para as visualizações devido à sua capacidade de gerar gráficos interativos com recursos como *zoom, hover* e exportação, que enriquecem a experiência do usuário (Plotly, 2023). Comparado a bibliotecas como Matplotlib, que produz gráficos estáticos, ou Seaborn, que tem interatividade limitada, Plotly oferece maior dinamismo e compatibilidade nativa com Dash.

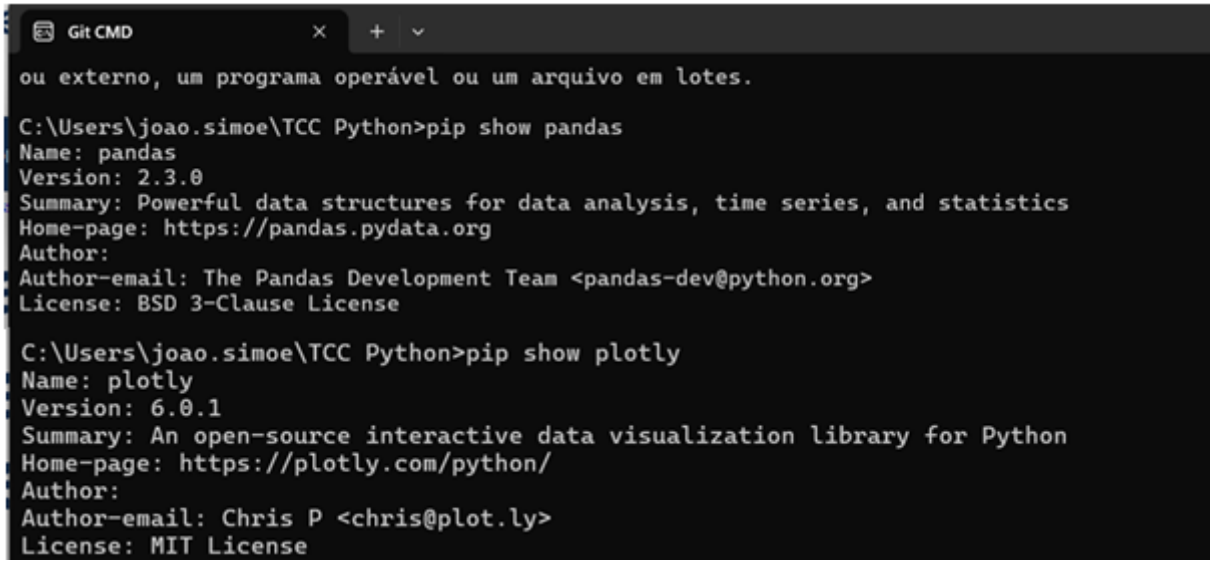
Pandas foi adotado para manipulação de dados por sua eficiência no processamento de *datasets* estruturados, como os arquivos Excel utilizados. Sua integração com Dash e Plotly cria um pipeline fluido, desde a leitura de dados até a visualização. Alternativas como NumPy ou SQL foram descartadas, pois Pandas oferece uma sintaxe mais intuitiva para filtragem e cálculos estatísticos, essenciais para as funcionalidades do dashboard (Perkel, 2022). É válido destacar que, assim como para dash, para o uso destas ferramentas, utilizou-se do comando, em CMD (pip install pandas plotly). Neste sentido, as figuras 8 e 9 ilustram a versão instalada dos programas no ambiente.

Figura 8 – Uso do comando pip show dash para ilustrar a versão 3.0.4 instalada no ambiente de desenvolvimento.



Fonte: Autoria Própria.

Figura 9 – Uso do comando pip show pandas (acima) e pip show plotly (abaixo) para demonstrar as versões 2.3.0 e 6.0.1, respectivamente, das bibliotecas instaladas no ambiente de desenvolvimento.



Fonte: Autoria Própria.

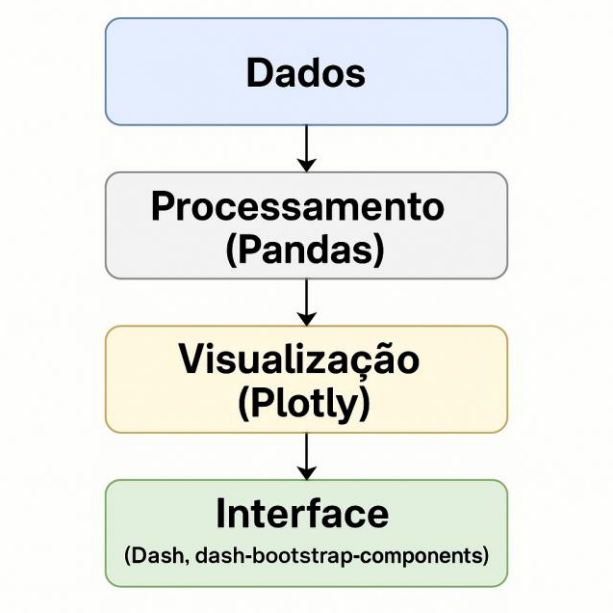
3.3.4 Plataformas de ajuda

O desenvolvimento do Dashboard Brasil contou com a ajuda de diversas plataformas e documentações como a do Dash (Plotly, 2025) e do dash-bootstrap-components (Faculty AI, 2025), que guiaram a criação de layouts interativos e responsivos. Exemplos de notebooks no Kaggle (Kaggle, 2025) inspiraram a implementação de gráficos dinâmicos e a manipulação de dados. Além disso, conceitos estatísticos, como o uso do IQR para detecção de outliers, foram reforçados por insights da comunidade Data Science Stack Exchange (Stack Exchange, 2025). Esses recursos foram adaptados e expandidos com base nas necessidades específicas do projeto, incorporando feedback de usuários para melhorar a usabilidade.

3.4 Framework

A arquitetura do dashboard é baseada no framework **Dash**, que integra componentes web interativos com bibliotecas de ciência de dados em Python. A figura 10 abaixo ilustra a arquitetura do sistema:

Figura 10 – Arquitetura do processo em camadas.



Fonte: Autoria Própria.

- **Camada de Dados**: Arquivos Excel são carregados via Pandas, que realiza filtragem e pré-processamento. Um componente dcc.Store armazena dados de filtros em cache, otimizando o desempenho.
- **Camada de Processamento**: Pandas executa operações como junção de *datasets*, cálculo de estatísticas (média, mediana, IQR) e preparação de dados para visualização.
- **Camada de Visualização**: Plotly gera gráficos interativos (barras, linhas, *boxplots, heatmaps*) com recursos como zoom e *hover*, integrados diretamente ao Dash.
- **Camada de Interface**: Dash gerencia o layout, utilizando *dash-bootstrap-components* para responsividade e estilização. *Callbacks* reativos conectam inputs (ex.: *dropdowns*) a outputs (ex.: gráficos), garantindo interatividade.

Essa arquitetura segue as recomendações de Larson e Chang (2020) para sistemas de BI, que enfatizam a separação de camadas para modularidade e escalabilidade, enquanto a escolha de Dash foi motivada por sua capacidade de criar interfaces web sem exigir conhecimento avançado em JavaScript, conforme destacado por Perkel (2022).

3.5 Cálculos Estatísticos

Os cálculos estatísticos implementados no dashboard foram selecionados para fornecer uma análise robusta e acessível dos dados, atendendo às necessidades de usuários acadêmicos e profissionais. A “detecção de outliers” utiliza o método do Intervalo Interquartil (IQR), que identifica valores fora das cercas (limites) definidas por (Q1 - k . IQR) e (Q3 + k . IQR), onde

(k) é um multiplicador ajustável (padrão: 1.5). Segundo Montgomery (2013), o IQR é uma abordagem não paramétrica robusta, adequada para *datasets* com distribuições assimétricas, como os utilizados no projeto (ex.: emissões de CO2, PIB). Assim, a possibilidade de ajustar (k) via interface aumenta a flexibilidade, permitindo ao usuário personalizar a sensibilidade da análise.

As “estatísticas descritivas” (média, mediana, desvio padrão, curtose) foram incluídas para resumir as características dos dados filtrados, fornecendo uma visão geral antes da exploração visual. Knaflic (2015) enfatiza que estatísticas descritivas são fundamentais em dashboards, pois contextualizam as visualizações, ajudando o usuário a interpretar tendências e anomalias, sendo que, a curtose, embora menos comum, foi adicionada para avaliar a forma da distribuição, sendo relevante em análises atmosféricas e epidemiológicas, onde distribuições leptocúrticas ou platicúrticas podem indicar fenômenos específicos.

Para realizar estas análises descritivas, foram utilizadas as bibliotecas pandas e scipy que permite o uso de variáveis que buscam os cálculos em biblioteca para a média, mediana, desvio padrão e curtose, como pode ser visto nos códigos da Figura 11 e que foram instalados segundo a Figura 9.

Figura 11 – Importação das bibliotecas e definição das variáveis e dos cálculos para as estatísticas descritivas.

models.py

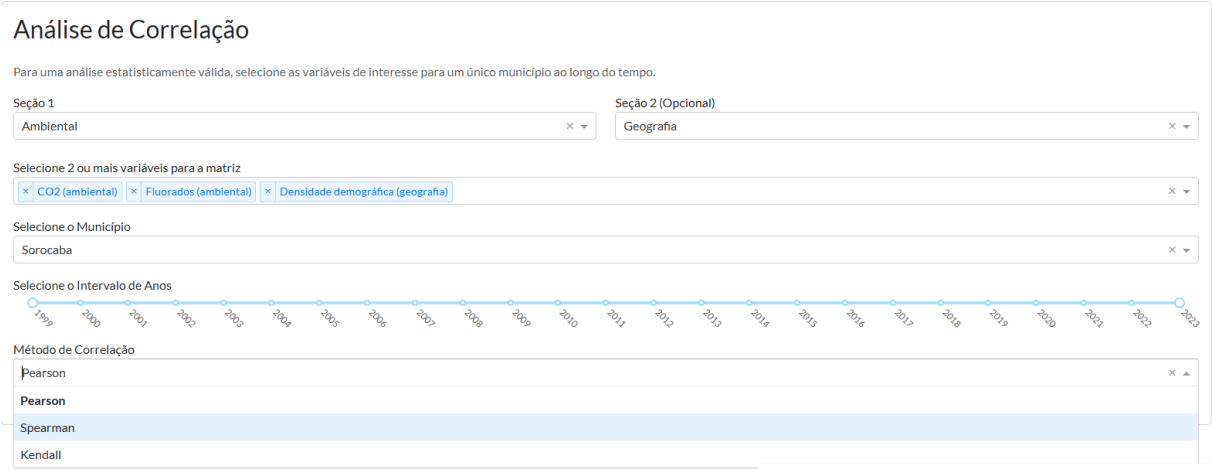
```python
import pandas as pd
from scipy import stats
import os
import unicodedata
from load_data import load_filters, load_variable_data, combine_data_with_filters, load_multiple_variables, DATA_PATH

## Definição para as variáveis para calcular estatisticas
stats_result = {}
if variavel in df.columns:
    df[variavel] = pd.to_numeric(df[variavel], errors='coerce')
    df_clean = df[variavel].dropna()
    if not df_clean.empty:
        stats_result['mean'] = df_clean.mean()
        stats_result['median'] = df_clean.median()
        stats_result['std_dev'] = df_clean.std()
        stats_result['kurtosis'] = stats.kurtosis(df_clean, nan_policy='omit', fisher=False)
        # Calcular cercas
        q1 = df_clean.quantile(0.25)
        q3 = df_clean.quantile(0.75)
        iqr = q3 - q1
        stats_result['lower_fence'] = q1 - iqr_multiplier * iqr
        stats_result['upper_fence'] = q3 + iqr_multiplier * iqr
    else:
        stats_result = {key: None for key in ['mean', 'median', 'std_dev', 'kurtosis', 'lower_fence', 'upper_fence']}
else:
    stats_result = {key: None for key in ['mean', 'median', 'std_dev', 'kurtosis', 'lower_fence', 'upper_fence']}
return stats_result
```

Fonte: Autoria Própria.

A análise de correlação suporta três métodos (Pearson, Spearman, Kendall), permitindo ao usuário escolher a abordagem mais adequada ao tipo de dado, como pode ser visto na figura 12. Pearson é ideal para relações lineares, enquanto Spearman e Kendall são robustos para dados não paramétricos ou ordinais (Cohen et al., 2003). Essa diversidade atende à natureza multidisciplinar do projeto, que inclui variáveis como índices de epidemiologias (ordinais) e emissões (contínuas). Daí, a inclusão de métodos múltiplos, refletindo as práticas de BI modernas, que priorizam a personalização da análise (Alpar; Schulz, 2021), ou o *Self-BI*.

Figura 12 - Filtros para a análise de correlação que segue a orientação em cascata e possui a seleção dos métodos de correlação à critério do usuário.



Fonte: Autoria Própria.

3.6 Visualizações Gráficas

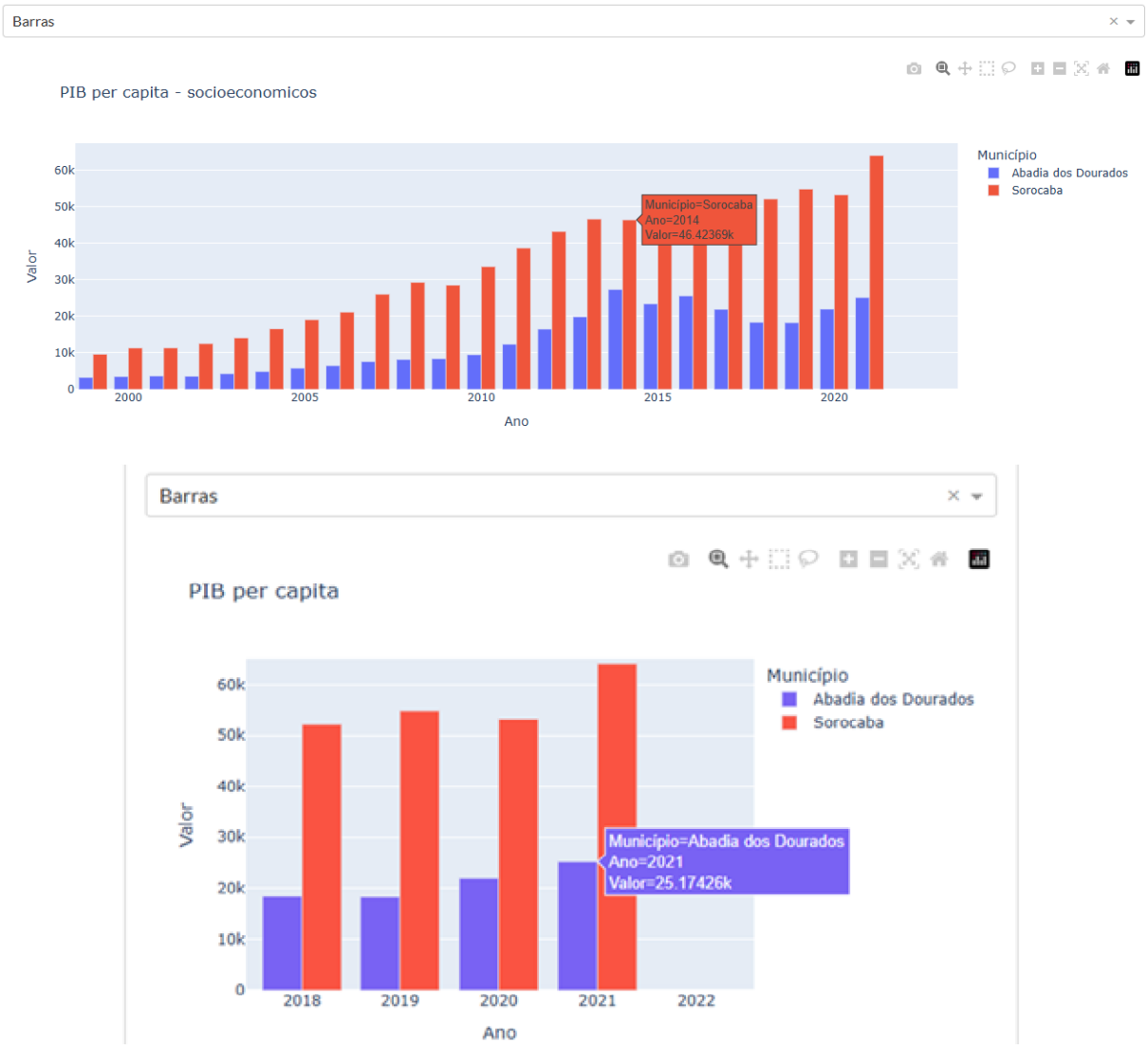
Os gráficos selecionados — barras, linhas e *boxplots* — foram escolhidos por sua adequação aos objetivos analíticos do dashboard, sendo que, gráficos de barras são ideais para comparações categóricas entre anos e municípios, com barras agrupadas (*barmode*='group') para evitar empilhamento e facilitar a interpretação (Cairo, 2019); os gráficos de linhas destacam tendências temporais, como a evolução de emissões de CO2 ao longo dos anos, alinhando-se às recomendações de Knaflic (2015) para narrativas visuais claras; e, os boxplots revelam distribuições e outliers, sendo particularmente úteis para análises estatísticas, como a variabilidade de indicadores epidemiológicos por ano. Isso fica claro nas figuras 13 (a), (b) e (c).

Figura 13 – Gráfico de PIB per capita para as cidades de Abadia de Dourados (Azul) e Sorocaba (Vermelho) em barras (a), em linhas (b) e em boxplot (c).

Fonte: Autoria Própria.

A interatividade dos gráficos, proporcionada por Plotly, inclui recursos como *hover* (exibição de valores ao passar o mouse) e *zoom*, que enriquecem a exploração de dados, seguindo a recomendação de Zhang e Wang(2023) quando afirmam que a interatividade é essencial em dashboards modernos, pois empodera o usuário a investigar detalhes sem sobrecarga cognitiva. Além disso, a responsividade, garantida por *dash-bootstrap-components*, assegura que os gráficos se adaptem a diferentes dispositivos, seguindo os princípios de design universal (Mace, 1997), como pode ser visto nas figuras 14.

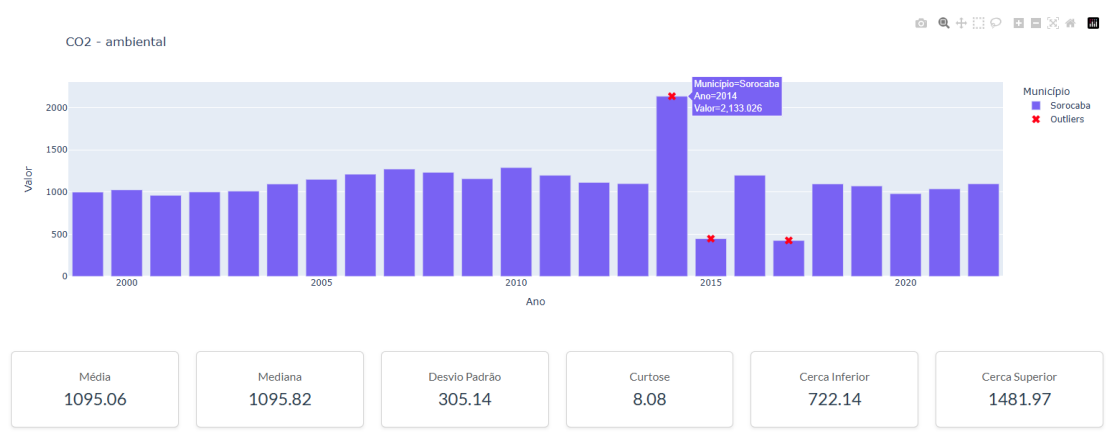
Figura 14 – Funcionalidades de Zoom e de *Hover* no gráfico de barras para *desktop* (acima) e para *mobile* (abaixo).



Fonte: Autoria Própria.

Comparados a alternativas, como gráficos estáticos ou mapas geográficos (não implementados devido à complexidade), os gráficos escolhidos equilibram simplicidade e funcionalidade, atendendo às necessidades de usuários não especialistas. Na figura 15, é possível observar a marcação de outliers com pontos vermelhos em barras e linhas reforça a clareza visual, um princípio defendido por Tufte (2001), mas adaptado a um contexto interativo.

Figura 15 – Gráfico de barras demonstrando o CO2 por ano para a cidade de Sorocaba, revelando 3 anos que são outliers demarcados por um ‘x’ vermelho.



Fonte: Autoria Própria.

3.7 Filtros

Os "filtros em cascata" (variável → ano → município → tipo de gráfico → multiplicador IQR) foram implementados para proporcionar uma exploração hierárquica e intuitiva dos dados. Segundo Holzinger e Ziefle (2010), filtros dinâmicos são fundamentais em ferramentas de visualização, pois reduzem a complexidade e guiam o usuário na análise.

Outra funcionalidade que viu-se ser essencial foi a ordenação alfabética de variáveis e municípios que, combinada com a possibilidade de buscar as cidades escrevendo tanto em *Mobile* como *Desktop*, agilizam a análise, evitando que o usuário permaneça preso na filtragem e não na análise em si.

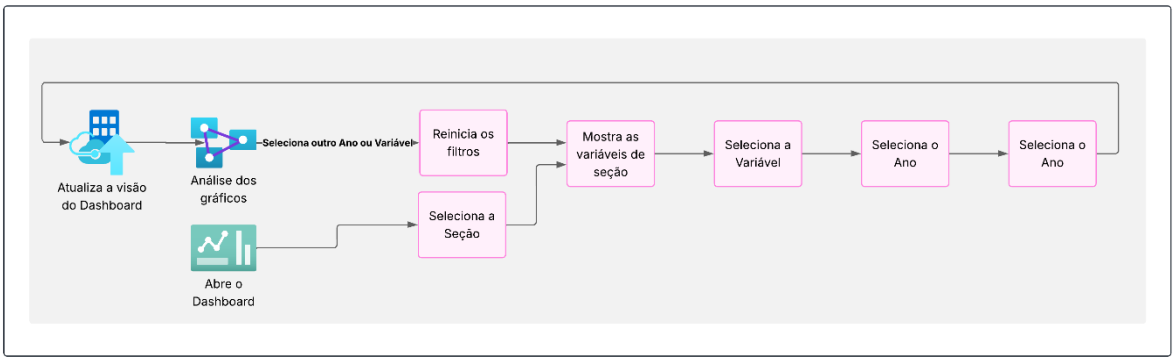
Além disso, optou-se pelo uso dos filtros em forma de *slider* facilitando a seleção de períodos de análise que, por regra, devem ser selecionados antes da escolha do município, assim, sempre que o ano for selecionado, ele reinicia a seleção da cidade. Por fim, comparados a filtros estáticos ou botões clicáveis (estilo Tableau), os *dropdowns* são mais escaláveis para *datasets* com muitas opções, justificando sua adoção. A partir das figuras 16 (a) e (b) é possível observar algumas destas funcionalidades

Figura 16 – Interface baseada em filtros por cascata, com *placeholder* e ordenação alfabética (a), e filtros de anos em forma de slider (b).

Fonte: Autoria Própria.

É válido observar que existe somente os filtros de variável, ano, município e tipo de gráfico disponíveis para filtragem, tendo em vista que a seleção das próprias seções já é uma variável. Então para cada seção, aparecerá somente as variáveis respectivas a esta. Assim, a interface baseada em filtros por cascata, funciona de forma a guiar o usuário pelos filtros, obtendo o resultado mais objetivo possível. Este fluxo de análise pode ser observado na figura 17.

Figura 17 – Diagrama de uso do dashboard, focando no processo de seleção dos filtros**.**



Fonte: Autoria Própria.

3.8 Análise de Correlação

A seção de análise de correlação, com *heatmaps*, *scatter plots* e exportação de dados, foi incluída para suportar análises multidisciplinares, como a relação entre PIB e indicadores epidemiológicos. *Heatmaps* oferecem uma visão geral das correlações, enquanto *scatter plots* permitem exploração detalhada, com dados de *hover* (município, ano). Alpar e Schulz (2021) destacam que ferramentas de *BI self-service* devem oferecer análises avançadas acessíveis a não especialistas, um objetivo alcançado por essa funcionalidade, isso pode ser observado nas visualizações apresentadas pela figura 18.

Figura 18 – Matriz de correlação de Pearson entre Fluorados, CO2 e Densidade Demográfica para o município de Sorocaba (a). Gráfico de dispersão Fluorados vs. CO2 para o município de Sorocaba, com linha de tendência em vermelho (b).

(a)

(b)

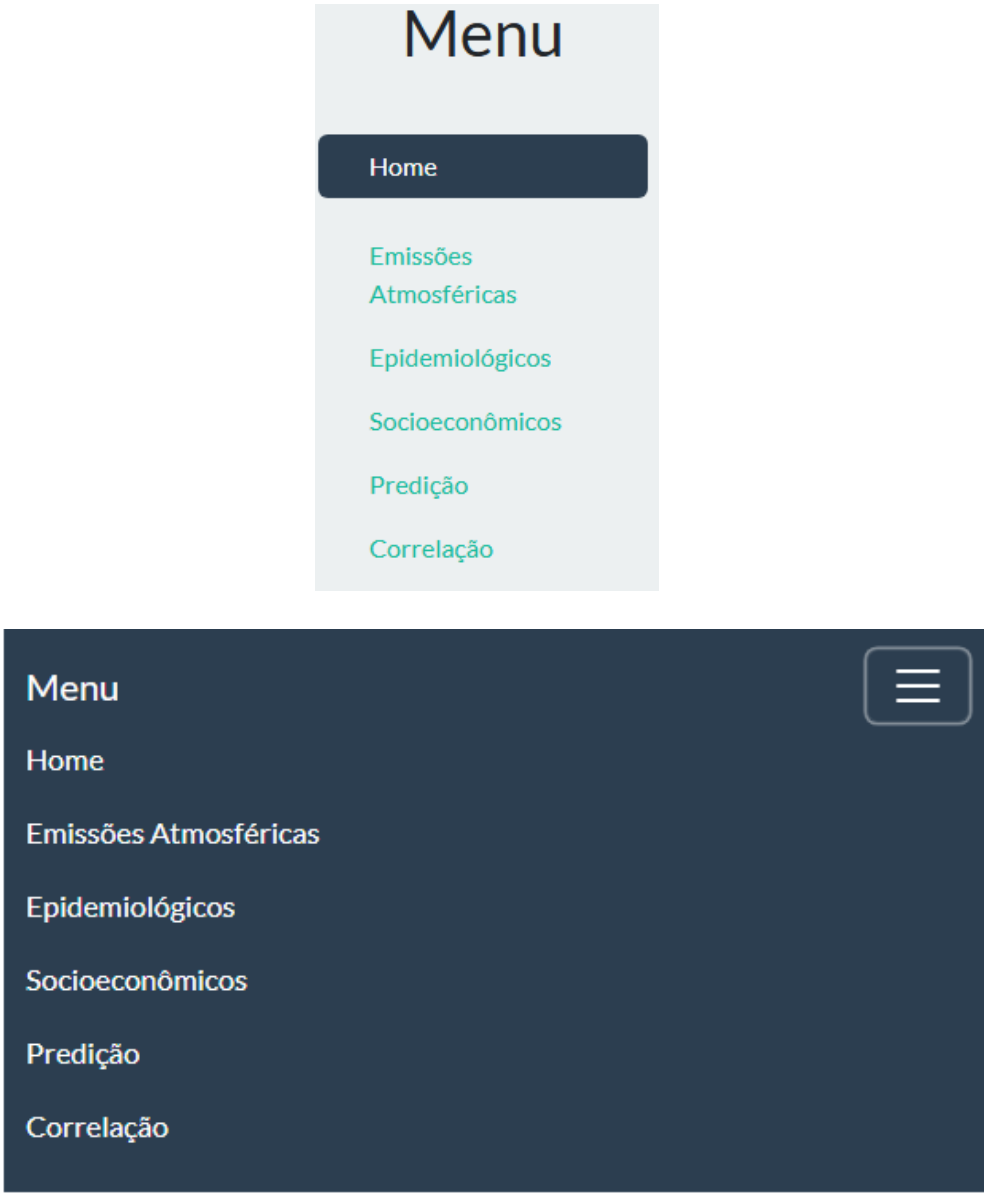
Fonte: Autoria Própria

3.9 Melhorias de UI/UX e Layout Escolhido

As melhorias de UI/UX foram projetadas para maximizar a usabilidade e a acessibilidade, com um layout cuidadosamente escolhido para atender às necessidades de usuários acadêmicos e profissionais. O dashboard adota uma *navbar* para dispositivos móveis e uma *sidebar* para desktops, inspiradas em ferramentas de BI como Tableau e Power BI, que priorizam navegação intuitiva (Holzinger; Ziefle, 2010). A *navbar*, implementada com *dash-bootstrap-components*, colapsa em um menu hamburguer em telas menores, garantindo acesso rápido às seções (Emissões Atmosféricas, Epidemiológicos, Socioeconômicos, Predição, Correlação). A *sidebar*, visível em desktops, organiza as seções verticalmente, reduzindo a sobrecarga visual e facilitando a transição entre análises, conforme discutido por Norman (2013) sobre design centrado no usuário, conforme figura 19.

A organização do layout por seções temáticas reflete a modularidade dos dados, permitindo ao usuário focar em domínios específicos (ex.: emissões de CO2 na seção “Emissões Atmosféricas”) antes de explorar correlações interdisciplinares. Cada seção apresenta um painel com filtros em cascata no topo, gráficos e tabelas no centro, e estatísticas descritivas na parte inferior, seguindo as práticas de narrativa visual de Knaflic (2015). Essa disposição hierárquica guia o usuário da seleção de dados à interpretação de resultados, minimizando a curva de aprendizado.

Figura 19 – Imagem ilustrativa do Menu no *desktop* (acima) e na versão *mobile* (abaixo).



Fonte: Autoria Própria.

A estilização visual utiliza cores neutras (ex.: tons de cinza e azul) para evitar distrações, fontes legíveis e espaçamento consistente para clareza. Cairo (2019) destaca que a estética deve reforçar a funcionalidade, e o uso de *dash-bootstrap-components* assegura responsividade, adaptando o layout a diferentes tamanhos de tela sem comprometer a experiência. A busca sem acentos nos *dropdowns* de municípios, implementada com a biblioteca unicodedata, resolve problemas de usabilidade em listas extensas, enquanto a ordenação alfabética de filtros proporciona previsibilidade.

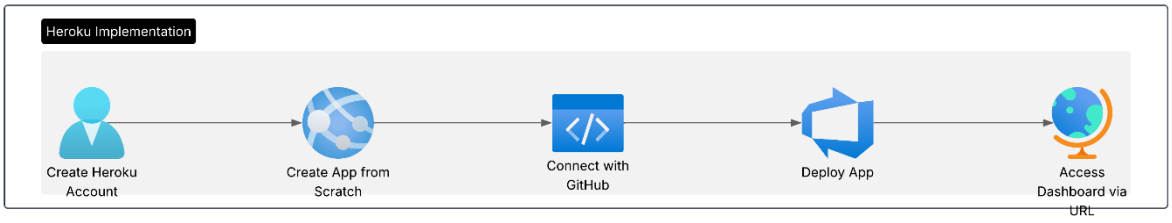
Comparado a layouts alternativos, como abas (*tabs*) ou uma única página com *scroll*, o design *navbar/sidebar* foi escolhido por sua escalabilidade e clareza em projetos com múltiplas seções e, a forte inspiração em ferramentas comerciais garante familiaridade, enquanto a

personalização via Dash permite ajustes específicos ao contexto acadêmico, como a ênfase em análises estatísticas detalhadas.

3.10 Implementação em nuvem

Finalizado o desenvolvimento do projeto e publicada a versão de lançamento "Heroku v2", foi possível realizar a conexão do Heroku, já configurado com uma conta pessoal, com uma conta pessoal do GitHub, permitindo que o primeiro realizasse a leitura de todas as informações disponíveis no *commit* mais atual do GitHub. Foi um processo bem direto de implementação na nuvem, sendo necessário somente a realização de configurações e conexões, conforme visto na figura 20.

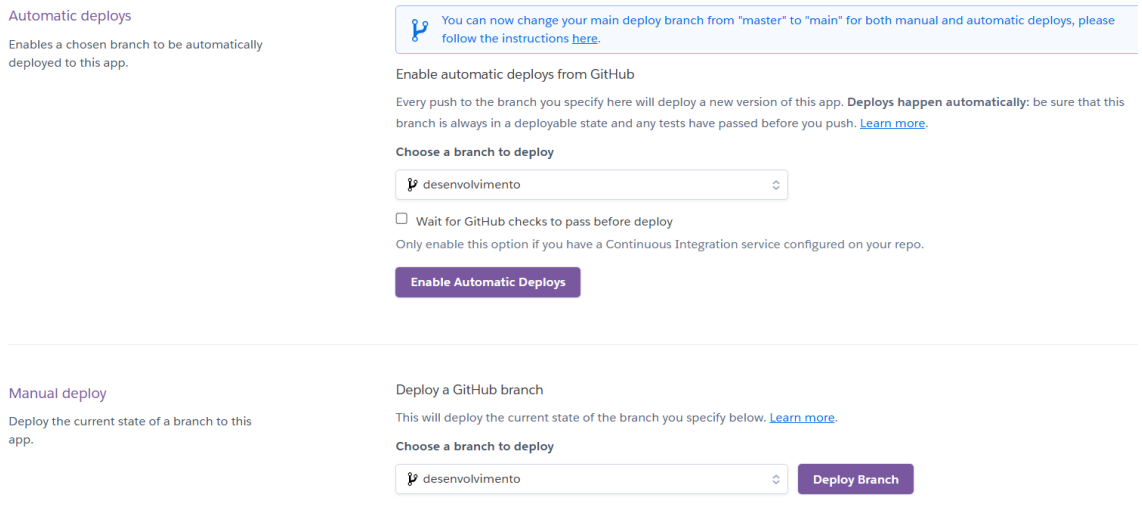
Figura 20 – Diagrama do processo de implementação em Heroku.



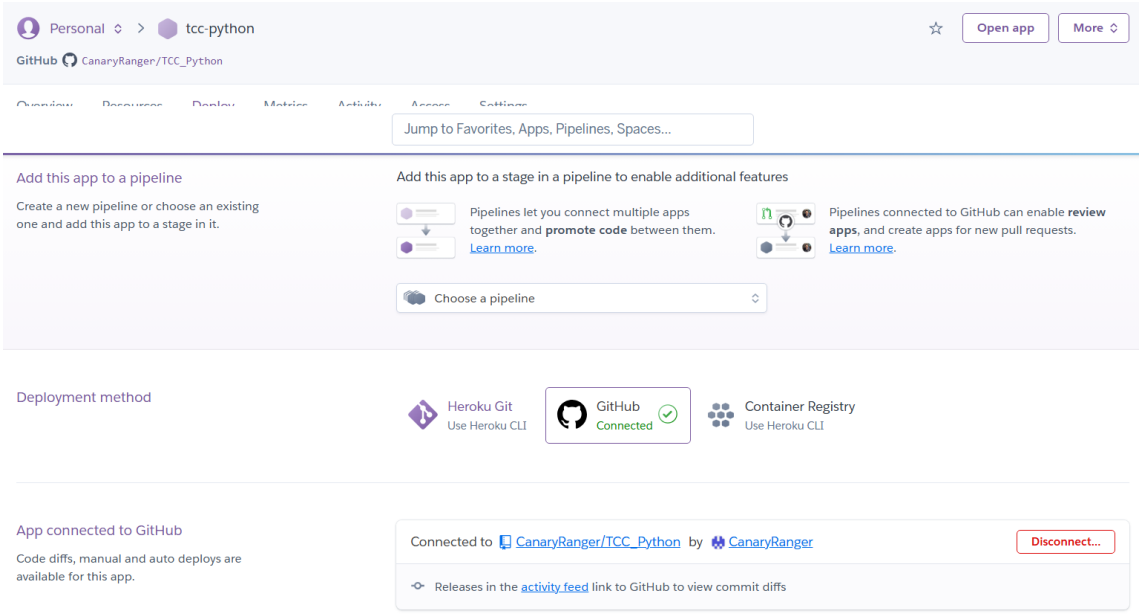
Fonte: Autoria Própria.

Desta forma, bastou configurar os recursos da aplicação e realizar o *deploy*, a partir da *branch* "desenvolvimento", tendo em vista que não fora criada uma nova para o lançamento, tornando assim, esta aplicação disponível em nuvem pela primeira vez de forma produtiva. Sua versão mais atual é a versão Heroku Release v1.2, conforme figuras 21 (a) e (b).

Figura 21 - Interface do Heroku para a realização da conexão com o GitHub (a) e as configurações da aplicação para o deploy da aplicação em produção (b).



(a)



(b)

Fonte: Autoria Própria.

4. RESULTADOS ALCANÇADOS

O dashboard desenvolvido atende de forma integral aos objetivos propostos no Capítulo 1 deste trabalho, ao fornecer uma ferramenta robusta, interativa e acessível para análise de dados relacionados às áreas de emissões atmosféricas, epidemiologia, socioeconomia e preditiva. Os principais resultados obtidos durante a implementação e os testes são apresentados a seguir, com base em critérios de funcionalidade, usabilidade e aplicabilidade prática.

Em primeiro lugar, destaca-se a exploração intuitiva proporcionada pela interface. Os filtros em cascata implementados permitem que o usuário selecione variáveis específicas (como emissões de dióxido de carbono), anos e municípios de forma hierarquizada, com suporte a buscas insensíveis a acentuação gráfica. Testes funcionais realizados durante a fase de validação indicaram tempos de resposta inferiores a um segundo nas atualizações visuais, evidenciando boa performance mesmo em cenários com alta densidade de dados.

No que se refere às visualizações gráficas, os resultados foram igualmente satisfatórios. Foram utilizados gráficos de barras, linhas e *boxplots* para representar os dados com precisão e clareza. A funcionalidade interativa (como exibição de valores por meio de *hover*, e possibilidade de zoom) favoreceu a análise exploratória. Em um dos casos práticos testados, a análise do PIB por município revelou tendências de crescimento acentuado nas capitais, enquanto municípios de menor porte apresentaram valores discrepantes, identificados como outliers por meio da técnica de IQR. Essa interatividade, aliada a possibilidade de exportação dos dados brutos para análises mais profundas fora da aplicação, a tornam robusta e versátil, além de ser uma funcionalidade que auxilia na validação de dados.

Com relação à análise estatística, a seção dedicada à correlação entre variáveis permitiu identificar relações significativas entre os dados. Como exemplo, pode-se citar a correlação positiva entre o PIB e indicadores de epidemiológicos, observada em determinados anos, com coeficiente de Pearson em torno de 0,75. Essas relações foram visualizadas por meio de *heatmaps* e gráficos de dispersão, facilitando a interpretação dos padrões existentes.

Além disso, observou-se forte aplicabilidade prática do sistema, por meio da funcionalidade de exportação de dados. Os dados podem ser exportados tanto de forma bruta em formato de Tabela no Excel, como visto nas Tabelas do Anexo A e B, quanto em forma de matrize de correlação (também em excel, mas formatada para matriz) observada na Tabela 1, viabilizando sua integração com outras ferramentas analíticas e promovendo maior utilidade para pesquisadores e gestores públicos.

Tabela 1 - Matriz de correlação entre CO2, Fluorados e Densidade demográfica extraída em excel a partir do botão de extração da matriz.

	CO2	**Fluorados**	**Densidade demográfica**
CO2	1	0,171571	-0,03141
Fluorados	0,171571	1	0,180035
Densidade demográfica	-0,03141	0,180035	1

Fonte: Autoria Própria.

Em termos de acessibilidade e responsividade, o layout foi estruturado com uso de componentes de navegação lateral e superior (*sidebar e navbar*), adaptando-se automaticamente a diferentes resoluções de tela, incluindo dispositivos móveis. A biblioteca *dash-bootstrap-components* foi utilizada para garantir essa flexibilidade visual. Testes qualitativos apontaram que a navegação entre as seções é intuitiva, com uma organização clara dos blocos informativos e visuais.

Entre os exemplos de uso destacam-se análises de emissões de dióxido de carbono por município, que evidenciaram picos em anos específicos (como nos anos de 2014, 2015 e 2017) para a cidade de Sorocaba, como pode ser visto na Figura 15, bem como investigações sobre correlações entre variáveis ambientais e indicadores geográficos como CO2 e Fluoratos versus Densidade Demográfica, respectivamente, os quais possuem potencial para subsidiar decisões políticas e estratégicas, como visto na Figura 17. Durante os testes preliminares com usuários atuantes na área de dados, como Analistas de Dados e Negócios, o feedback qualitativo foi majoritariamente positivo, ressaltando a facilidade de uso e a disposição funcional dos elementos. No entanto, algumas sugestões foram registradas, como a inclusão de *sliders* para a seleção de anos e botões de exportação que, em seguida, foram implementadas. Outras sugestões foram registradas, porém, não foram implementadas como a troca do tema para modo escuro e o gráfico comparativo entre dados de diferentes municípios.

4.1 Limitações

Apesar dos resultados positivos, o dashboard apresenta limitações que abrem oportunidades para melhorias futuras:

- **Mapas Geográficos**: A ausência de visualizações espaciais limita a análise geográfica que poderia ser uma visualização poderosa e bastante criativa, trazendo em tempo real, o resultado das filtragens pelo Brasil. A integração de bibliotecas como “Folium” ou “Plotly Geo” poderiam realizar uma mudança interessante na forma como o dashboard se comporta.
- **Exportação de Gráficos**: Atualmente, os gráficos não podem ser exportados como imagens (PNG/SVG), uma funcionalidade comum em ferramentas de BI (Zhang; Wang, 2023), somente se atendo a exportação em formato CSV.
- **Interatividade Avançada**: A falta de *drill-down* (ex.: clicar em um gráfico para detalhar dados) e *tooltips* personalizados, reduz a profundidade da exploração.
- **Estilização Visual**: O design, por opção, está bastante amarrado aos *templates* oferecidos pela biblioteca do dbc e é pobre em funcionalidades voltadas para a acessibilidade como troca de tema, aumento de letra e leitura por voz.

O Dash utiliza componentes básicos derivados do HTML e CSS por meio das bibliotecas *dash_html_components e dash_core_components* e, apesar de permitir a aplicação de estilos *inline* ou via arquivos externos .css - por meio da pasta “assets” -, a personalização visual dos componentes é limitada. Por exemplo, alguns elementos como *sliders, dropdowns e tooltips* possuem estrutura interna rígida, dificultando a aplicação de estilos mais refinados ou responsivos, como transições, animações, efeitos de *hover*, temas dinâmicos (*dark/light mode*) e ajustes de layout por resolução de tela. Sendo assim, pode-se dizer que a interação entre arquivos externos CSS e os componentes base do dbc como as classes, é bastante limitada e traz poucos dispositivos para alternar estilos, temas e afins.

Essas limitações não comprometem a funcionalidade atual, mas indicam direções para evolução. Futuras iterações poderiam incorporar inteligência artificial para predições automáticas ou visualizações 3D para análises complexas, ampliando o escopo do projeto.

Como dito na seção 3.3, a escolha de Dash foi motivada por sua capacidade de criar interfaces web sem exigir conhecimento avançado em JavaScript, porém, como todo *framework*, a estilização em Dash é bastante limitada devido a ele não possuir um sistema de estilização interno tão robusto quanto *frameworks front-end* modernos como React, Vue ou bibliotecas especializadas como Tailwind CSS, então existe ainda uma curva de aprendizado e fez-se necessária a leitura da documentação de Dash, principalmente para escolha de temas.

Para contornar essa limitação, focou-se na utilização da biblioteca *dash-bootstrap-components,* que permite a estruturação da interface com *Container, Row, Col*, entre outros recursos do "Bootstrap" que foram bastante importantes para a criação de elementos responsivos. Além disso, foi possível, com o uso dos componentes de *template* do "bootswatch", site e aplicação com alguns temas para uso gratuito, trazer um frescor à interface. Contudo, mesmo com o apoio dessa biblioteca e a inclusão de um arquivo CSS personalizado no diretório "assets", ainda assim foi complexo alcançar um nível de refinamento visual satisfatório, especialmente no que tange à responsividade e à estética avançada já comentados.

4.2 Dificuldades

No âmbito das dificuldades, foram várias, principalmente com a adaptação aos componentes do framework Dash como o *dash-bootstrap-components.* Apesar de muito mais intuitivo que o JavaScript e com uma curva de aprendizado menor que o React, por exemplo, o *framework* mostrou-se desafiador quanto ao uso de suas classes para estilização, já que é feita de forma bastante diferente ao CSS e o discernimento para o encapsulamento do código em forma de *callbacks*, ou seja, a dificuldade girou em torno, principalmente, da implementação de layouts responsivos, gráficos dinâmicos e interatividade.

O desenvolvimento de uma aplicação de dados, desde a concepção até a implantação em um ambiente de produção, invariavelmente apresenta desafios que transcendem a escrita inicial do código. A trajetória deste projeto revelou três categorias principais de dificuldades: (4.2.1) desafios de arquitetura de software e manutenibilidade; (4.2.2) desafios de integração e consistência de dados; (4.2.3) desafios da arquitetura *serverless*; e (4.2.4) desafios de desempenho e implantação (*deployment*).

4.2.1 Desafios de Arquitetura de Software e Manutenibilidade

Inicialmente, o projeto foi desenvolvido em um único arquivo app.py, uma abordagem comum para protótipos em Dash. Contudo, à medida que novas funcionalidades foram adicionadas, esta estrutura monolítica revelou suas limitações, resultando em um código extenso, de difícil navegação e com alto acoplamento entre a lógica de interface (layout), a lógica de interatividade (*callbacks*) e a lógica de dados. A principal dificuldade técnica encontrada foi o risco iminente de importações circulares, um problema notório em aplicações Dash complexas, onde o arquivo de *callbacks* precisa importar componentes do layout, enquanto o arquivo de layout precisa da instância do app para funcionar.

A solução foi uma refatoração completa do projeto para um padrão de arquitetura inspirado no MV, separando as responsabilidades em arquivos distintos. Essa modularização, embora tenha exigido um esforço inicial de reestruturação, provou-se essencial para a escalabilidade e a manutenção do sistema a longo prazo.

4.2.2 Desafios de Integração e Consistência de Dados

Esta foi a área que apresentou os obstáculos técnicos mais persistentes e sutis, todos originados da necessidade de integrar múltiplos arquivos Excel de fontes heterogêneas.

A dificuldade mais impactante foi a inconsistência implícita nos tipos de dados das colunas-chave (CD_MUN, ANO), pois a biblioteca Pandas, ao ler diferentes arquivos Excel, denotava tipos distintos para a mesma coluna (ex: int64 em um arquivo, float64 ou object em outro que continha uma célula vazia). Isso ocasiona falhas silenciosas durante as operações de junção (*merge*), resultando em comportamentos inconsistentes na visualização, como a exibição de eixos de anos com valores incorretos, exemplo valor de 0 a 2023. A solução definitiva foi implementar uma função de sanitização (set_key_types) que, logo após a leitura de qualquer arquivo, força a conversão das colunas-chave para um tipo de dado numérico consistente (Int64), garantindo a integridade referencial em todas as junções.

Ao combinar múltiplas variáveis para a análise de correlação, o sistema enfrentou erros do tipo "*MergeError*: *... duplicate columns"*. Isso ocorria porque a função de carregamento tentava juntar *DataFrames* que possuíam colunas com nomes idênticos (notavelmente NM_MUN), fazendo com que o Pandas não conseguisse resolver a ambiguidade após a primeira iteração. Para solucionar o problema, foi necessário refatorar a lógica de junção para que, de cada arquivo de variável, fossem selecionadas apenas as colunas essenciais (CD_MUN, ANO e VALOR), descartando colunas redundantes antes que pudessem causar conflito no merge final.

4.2.3 Desafios da arquitetura *serverless*

A arquitetura *serverless* foi escolhida por sua escalabilidade, baixo custo e simplicidade, alinhada às necessidades de um projeto inicialmente pequeno. O S3 ofereceria o armazenamento durável e acessível, enquanto o Lambda elimina a necessidade de servidores, reduzindo a complexidade operacional. O uso do *Free Tier* garantiria que o projeto fosse implantado sem custos, sendo sustentável para demonstrações. Porém, nada disso foi possível nesta etapa do projeto, pois a arquitetura mostrou-se mais desafiadora do que o imaginado. Incontáveis horas foram gastas com vídeos e pesquisas, para a implementação da aplicação em

AWS, sem sucesso. Em todas as ocasiões, uma única mensagem se apresentou, "*BAD GATEWAY*", demonstrando que, seja as chaves API, seja os *EndPoints*, nenhum estava configurado corretamente.

A solução, portanto, foi abandonar o uso da AWS como forma de hospedagem e de armazenamento dos dados, substituindo-a por uma metodologia mais prática e direta com o uso do GitHub para o armazenamento das versões e a hospedagem e publicação da aplicação no Heroku, como apresentado na seção 3.

4.2.4 Desafios de Desempenho e Implantação (*Deployment*)

A transição do ambiente de desenvolvimento local para um servidor de produção gratuito (Heroku, Render) expôs gargalos críticos de desempenho. O erro mais recorrente foi o *[CRITICAL] WORKER TIMEOUT*, indicando que uma requisição estava demorando mais de 30 segundos para ser processada, o que levava o servidor a encerrar o processo, muitas vezes com um erro subsequente de falta de memória ("*Perhaps out of memory?*").

A investigação revelou que a causa raiz era a operação de leitura de arquivos Excel (pd.read_excel), que é intensiva em I/O e computacionalmente "cara". Em um ambiente com recursos de CPU e RAM limitados, como os dos planos gratuitos, a leitura repetitiva desses arquivos a cada interação do usuário era insustentável. A solução adotada foi a implementação de uma estratégia de cache em memória utilizando o decorador "@functools.lru_cache" nas funções de carregamento de dados. Essa otimização garante que cada arquivo seja lido do disco apenas uma única vez durante o ciclo de vida da aplicação, assim nas chamadas subsequentes, o *DataFrame* já processado é servido diretamente da memória, reduzindo o tempo de resposta de segundos para milissegundos e eliminando completamente os erros de timeout.

As dificuldades apresentadas evidenciam uma característica estrutural do Dash: o framework foi concebido prioritariamente para atender às demandas de cientistas de dados e engenheiros, com foco na funcionalidade e na interatividade dos dados, e não como uma ferramenta completa e rica em desenvolvimento visual.

Em suma, os obstáculos enfrentados no projeto não foram meramente erros de sintaxe, mas sim desafios inerentes à engenharia de aplicações de dados: a necessidade de uma arquitetura escalável, a garantia de robustez na manipulação de dados heterogêneos e a otimização de desempenho para ambientes de produção com recursos limitados.

5. CONSIDERAÇÕES FINAIS E CONCLUSÃO

5.1 Resumo das Contribuições

O desenvolvimento do dashboard interativo proposto neste trabalho resultou em uma ferramenta robusta, acessível e eficaz, que tem como principal mérito a democratização da análise de dados complexos por meio de uma interface intuitiva e de funcionalidades interativas e analíticas avançadas. A seguir, são sintetizadas as principais contribuições do projeto, considerando tanto os aspectos técnicos quanto os acadêmicos e práticos.

Em primeiro lugar, destaca-se a acessibilidade e democratização do acesso à análise de dados. A adoção de tecnologias de código aberto, como Python, Dash, Plotly e Pandas, elimina barreiras financeiras comuns em ferramentas comerciais de *BI*, como Tableau e Power BI, permitindo que estudantes, pesquisadores e gestores públicos utilizem a solução sem custos adicionais (Alpar; Schulz, 2021). Tal escolha reforça o compromisso com a inclusão tecnológica e a difusão do conhecimento.

A segunda contribuição refere-se à interatividade e usabilidade da interface desenvolvida. O uso de filtros em cascata, gráficos interativos (como gráficos de barras, linhas e *boxplots*) e visualizações dinâmicas de correlação (por meio de mapas de calor e diagramas de dispersão) oferece uma experiência fluida e orientada à descoberta. Esses elementos foram guiados por princípios de usabilidade em ambientes computacionais e de boas práticas em narrativa visual aplicada à análise de dados e no conceito de design universal.

Observa-se, principalmente, que o dashboard seguiu de forma criteriosa os 7 fundamentos de Mace vistos no capítulo 2.1.11, pois o design possui:

(1) Uso equitativo, de forma que todos os usuários usam e são capazes de usar a mesma interface, tendo em vista as diferentes visões e a filtragem guiada;

(2) Flexibilidade, já que o usuário pode usar mouse e teclado, as mãos ou personalizar a visão da forma que achar melhor dentro dos limites de desenvolvimento, tendo em vista o uso de Dash que já traz esta flexibilidade;

(3) Uso Simples, visto que os processos seguem uma lógica clara, familiar e possui interações nomeadas de forma intuitiva quando possui nomes em gráficos, páginas, legendas, cores e textos acompanhando botões;

(4) Informação Perceptível, pois todos os textos são legíveis, indicadores possuem diferentes cores, as legendas são claras e os gráficos possuem contraste;

(5) Tolerância a erros, de forma que segue uma filosofia de narrativa guiada, dificultando erro do usuário e, permitindo que este tenha confiança nas análises vez que elas não se redefinem sozinhas;

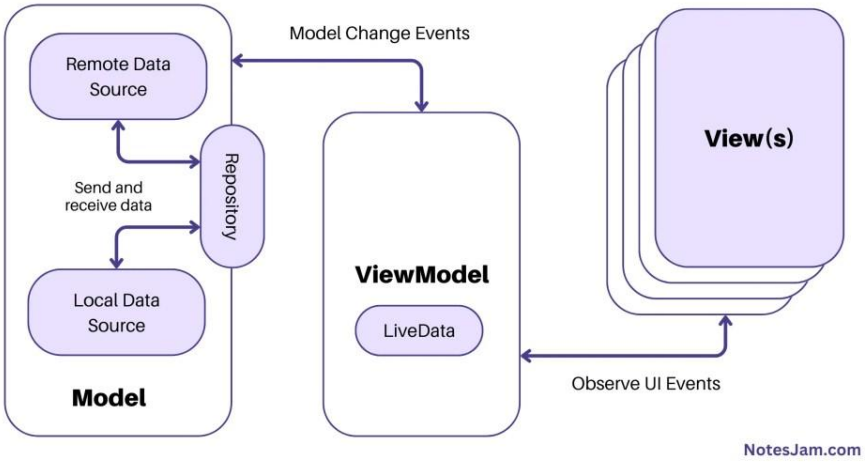
(6) Baixo esforço físico, tendo em vista que o usuário não é obrigado a navegar de forma intensa, visto que o dashboard requer somente uma filtragem inicial que pode ser feita com mouse e teclado;

(7) Tamanho e Espaço, já que o design é responsivo e nenhum elemento é pequeno nem se encontram "apertados" dentro da interface.

Adicionalmente, o projeto proporcionou a incorporação de técnicas de análise estatística descritiva e inferencial, como a detecção de outliers com o uso do método do intervalo interquartil (IQR) e a implementação de diferentes métodos de correlação (Pearson, Spearman e Kendall). Essas abordagens ampliam as possibilidades analíticas do sistema, tornando-o aplicável a múltiplas áreas do conhecimento, como saúde pública, sustentabilidade e planejamento urbano (Cohen et al., 2003).

A organização estrutural do projeto também merece destaque. A divisão modular por seções temáticas (Emissão de Poluentes, Epidemiológicos e Socioeconômicos), aliada ao layout baseado em barra de navegação superior e menu lateral, facilita a escalabilidade futura da aplicação, permitindo a integração de novos conjuntos de dados e funcionalidades adicionais. Além disso, tem-se o uso de um modelo semelhante ao MVVM, que pode ser analisado pelo diagrama da figura 22, de forma que o código se torna mais adaptável e compreensível, ou seja, de fácil manutenção. Tal abordagem segue os princípios de desenvolvimento pensado por Tukey e do *Business Intelligence* modular e escalável (Larson; Chang, 2020).

Figura 22 - Diagrama da arquitetura MVVM.



Fonte: Model View ViewModel (MVVM) Architecture in Android. (NOTEJAM.COM)

Outro aspecto relevante está na implantação moderna do sistema. A utilização de uma arquitetura do tipo *cloud*, implementada no ambiente Heroku, podendo ser escalado com o uso do banco de dados S3 oferecido pela AWS gratuitamente durante 1 ano, viabiliza a escalabilidade da solução e o acesso remoto por múltiplos usuários. Essa escolha técnica mostrou-se especialmente adequada ao contexto acadêmico e demonstrativo.

5.2 Recomendações para Aprimoramentos Futuros

Embora o dashboard cumpra integralmente os objetivos propostos, seu caráter de prova de conceito abre um vasto leque de oportunidades para aprimoramentos e extensões. A arquitetura modular e a base tecnológica *open-source* facilitam a incorporação de novas funcionalidades que podem elevar a ferramenta de uma plataforma de análise exploratória para um sistema de suporte à decisão estratégica. As recomendações a seguir detalham potenciais evoluções, agrupadas por natureza analítica.

5.2.1 Análise Comparativa e de *Benchmarking*

Uma limitação inerente à análise isolada de municípios é a ausência de um referencial comparativo. Para sanar essa lacuna, sugere-se a implementação de funcionalidades de *benchmarking*, que permitiriam ao usuário comparar os indicadores de um município selecionado com a média de um grupo (ex: média estadual, nacional ou de municípios com perfis socioeconômicos semelhantes). Adicionalmente, a criação de rankings dinâmicos possibilitaria a classificação de todos os municípios com base em uma variável de interesse, oferecendo um panorama competitivo e contextualizado que é fundamental em ferramentas de BI modernas (Wexler et al., 2017).

5.2.2 Análise de Séries Temporais e Projeções (*Forecasting*)

Atualmente, o dashboard permite a visualização de tendências históricas por meio de gráficos de linha. Uma evolução natural dessa funcionalidade seria a incorporação de modelos de projeção de séries temporais. Utilizando métodos estatísticos consagrados, como ARIMA (*AutoRegressive Integrated Moving Average*) ou mesmo modelos de regressão mais simples, o sistema poderia extrapolar tendências e oferecer projeções de curto e médio prazo para variáveis selecionadas. Essa capacidade transformaria o dashboard de uma ferramenta reativa, que descreve o passado, em uma ferramenta proativa, que auxilia no planejamento futuro, alinhando-se à crescente demanda por análise preditiva em plataformas de dados.

5.2.3 Análise Espacial e Visualização Geográfica

A natureza intrinsecamente geográfica dos dados municipais representa a oportunidade de aprimoramento mais impactante. Sugere-se a integração de mapas interativos, utilizando bibliotecas como Plotly Geo ou Folium. Tais visualizações permitiriam plotar os indicadores diretamente em um mapa do Brasil, onde a intensidade da cor de cada município representaria o valor de uma variável. Segundo Zhang e Wang (2023), a análise espacial é capaz de revelar padrões, *clusters* regionais e desigualdades territoriais que são virtualmente invisíveis em gráficos e tabelas tradicionais, agregando uma dimensão de análise fundamental para áreas como saúde pública e planejamento urbano.

5.2.4 Análise Preditiva com Aprendizado de Máquina

A análise de correlação existente estabelece a existência de uma relação entre variáveis, mas não quantifica seu poder preditivo. Um avanço significativo seria a implementação de uma seção de análise preditiva baseada em modelos de aprendizado de máquina (*machine learning*). Nesta seção, o usuário poderia selecionar uma variável-alvo (variável dependente, ex: "Índice de Saúde") e um conjunto de variáveis preditoras (variáveis independentes, ex: PIB, saneamento, emissões). Um modelo de regressão (ex: Regressão Linear Múltipla, *Random Forest*), treinado nos bastidores, poderia então não apenas identificar quais variáveis possuem maior impacto preditivo, mas também funcionar como um "simulador", permitindo que gestores explorem o impacto potencial de alterações em variáveis de entrada (Larson; Chang, 2020).

5.2.5 Melhorias em UI/UX

A interface do dashboard, embora intuitiva, pode ser aprimorada para oferecer uma experiência mais moderna e personalizável, alinhada às expectativas de usuários contemporâneos. As seguintes funcionalidades são recomendadas:

- **Drill-Down Interativo**: Permitir que o usuário clique em elementos gráficos (ex.: uma barra representando um município) para acessar dados detalhados (ex.: variáveis específicas por ano) adicionaria profundidade à análise. Zhang e Wang (2023) consideram o *drill-down* essencial em dashboards de BI modernos.
- **Temas Visuais Personalizáveis**: Implementar um seletor de temas (ex.: modo claro/escuro, esquemas de cores dinâmicos) aumentaria a acessibilidade e a estética. Holzinger e Ziefle (2010) recomendam personalização visual para atender a preferências de usuários diversos.
- **Tooltips Avançados**: Enriquecer *tooltips* com informações contextuais (ex.: descrição da variável, fonte do dado) e estilização (ex.: bordas arredondadas, cores contrastantes) melhoraria o feedback visual. Knaflic (2015) enfatiza que *tooltips* bem projetados reforçam a narrativa visual.
- **Animações e Transições**: Adicionar animações suaves (ex.: *fade-in* em gráficos, transições em filtros) tornaria a interação mais fluida, alinhando-se às tendências de UI/UX modernas.
- **Suporte a Múltiplos Idiomas**: Incorporar tradução para inglês e outros idiomas, usando bibliotecas como *dash-i18n*, facilitaria o uso por audiências internacionais, especialmente em contextos acadêmicos globais.
- **Feedback Visual Dinâmico**: Implementar indicadores de carregamento (ex.: *spinners*) e mensagens de erro personalizadas (ex.: “Nenhum dado disponível para esta seleção”) melhoraria a experiência, garantindo visibilidade do estado do sistema.

5.2.6 Outras Funcionalidades

Além de UI/UX e segurança, várias funcionalidades podem expandir o escopo e a aplicabilidade do dashboard, alinhando-se às tendências de BI e ciência de dados:

- **Filtros Avançados**: Implementar busca por expressões regulares (“regex”) nos *dropdowns* e filtros condicionais (ex.: “valores acima de X”) aumentaria a flexibilidade da exploração de dados (Alpar; Schulz, 2021).
- **Relatórios Automatizados em PDF**: Gerar relatórios PDF com gráficos, estatísticas e análises selecionadas, usando bibliotecas como *reportlab*, facilitaria a disseminação de resultados.

- **Integração com APIs Externas**: Conectar o dashboard a APIs públicas (ex.: IBGE, DataSUS) para dados em tempo real, como indicadores de saúde atualizados, ampliaria sua relevância (Larson; Chang, 2020).
- **Suporte a Grandes Datasets**: Otimizar o desempenho para *datasets* maiores, usando técnicas como *lazy loading* ou integração com bancos de dados (ex.: SQLite no Lambda), garantiria escalabilidade.

Assim, pode-se finalizar dizendo que do ponto de vista acadêmico e, em vista um cenário onde a análise de dados é cada vez mais crítica, a proposta contribui para o avanço do uso de ferramentas *open-source* no campo, atualmente dominado por soluções proprietárias. Em termos práticos, oferece uma ferramenta versátil, para democratizar o acesso ao conhecimento. Trabalhos futuros podem explorar parcerias com instituições (ex.: IBGE, DataSUS) para expandir os *datasets* e aplicações, consolidando o dashboard como uma solução de impacto global.

REFERÊNCIAS

ABRAS, C.; MALONEY-KRICHMAR, D.; PREECE, J. **User-centered design**. *In*: BAINBRIDGE, W. (ed.). Berkshire encyclopedia of human-computer interaction. [Massachusetts: Berkshire Publishing Group, 2004]. Disponível em: https://archive.org/details/berkshireencyclo0002unse/page/n9/mode/2up. Acesso em: 30 jun. 2025

ALPAR, P.; SCHULZ, M. **Self-service business intelligence.** Business & Information Systems Engineering, v.58, p. 151-155, 2016. DOI: 10.1007/s12599-016-0424-6.

ALPAR, P.; SCHULZ, M. **Self-service business intelligence:** empowering users with data analytics**.** Business & Information Systems Engineering, v. 63, p. 529-541, 2021.

AMAZON WEB SERVICES. **Armazenamento de objetos na nuvem Amazon S3**. **Cátalogo**: Amazon Web Services, [s.d.]. Disponivel em: https://aws.amazon.com/pt/pm/serv-s3/. Acesso em: 27 jun. 2025.

AMAZON WEB SERVICES. **AWS Lambda:** definição de preço. **Catálogo**: Amazon Web Services, [s.d.]. Disponivel em: https://aws.amazon.com/pt/lambda/. Acesso em: 27 jun. 2025.

BANDEIRA, A.; ROCHA, C.; SANTANA, V. (org.). **Acessibilidade**: práticas culturais e tecnologia assistiva para a cidadania. Goiânia: Gráfica da UFG, 2018. (Coleção Invenções). *E-book.*

BRIDEAU, L. **Correlation:** Pearson, Spearman, and Kendall's tau. [Charlottesville]: University of Virginia Library, [s.d.]. Disponível em: https://library.virginia.edu/data/articles/correlation-pearson-spearman-and-kendalls-tau. Acesso em: 18 jun. 2025

CAIRO, A. **How charts lie:** getting smarter about visual information. New York: W.W. Norton & Company, 2019.

CENTRAL POLLUTION CONTROL BOARD. [Parivesh Bhawan]: Ministry Of Environment, Forest And Cimate Change, 2024. 137 pages. Relatório. **Annual Report 2022-2023**. Disponível em: https://cpcb.nic.in/openpdffile.php?id=UmVwb3J0RmlsZXMvMTY2OV8xNzI3NDE0NTc1X21lZGlhcGhvdG8yOTAyNy5wZGY=. Acesso em: 27 jun. 2025.

COHEN, J. *et al.* **Applied multiple regression/correlation analysis for the behavioral sciences**. 2. ed. Mahwah: Lawrence Erlbaum Associates, 2003.

DINGSØYR, T.; MOE, N. B. **Agile software development:** current research and future directions. Berlin: Springer, v. 178, 2010. *E-book.*

DOS SANTOS, Robson. **30 Menus responsivos**. [2023]. Disponível em: https://www.brasilcode.com.br/30-menus-responsivos-em-html-e-css-gratuitos/. Acesso em: 25 jun. 2025.

FOWLER, 2020. **Refactoring**. Novatec Editorra, 2. ed. São Paulo: Novatec Editora Ltda, 2020.

GARTNER. **Glossary**. Stamford: Gartner, [2025]. Disponível em: https://www.gartner.com/en/information-technology/glossary/business-intelligence-bi. Acesso em 18 jun. 2025.

HOLZINGER, A., ZIEFLE, M., RÖCKER, C. Human-Computer Interaction and Usability Engineering for Elderly (HCI4AGING): Introduction to the Special Thematic Session. *In:* Computers Helping People with Special Needs. ICCHP 2010, vol 6180, p. 556-559. Berlin: Springer: 2010. DOI: https://doi.org/10.1007/978-3-642-14100-3_83. Disponível em: https://link.springer.com/chapter/10.1007/978-3-642-14100-3_83. Acesso em: 14 mai. 2025.

KHAN, B.; JAN, S.; KHAN, W.; CHUGHTAI, M. I. (2024). An Overview of ETL Techniques, Tools, Processes and Evaluations in Data Warehousing. **Journal on Big Data**. v. 6, p. 1–20. DOI: https://doi.org/10.32604/jbd.2023.046223. Disponível em: https://www.techscience.com/jbd/v6n1/55252. Acessado em: 18 jun. 2025.

KIMBALL, R.; ROSS, M. **The data warehouse toolkit: the definitive guide to dimensional modeling.** 3. ed. Indianapolis: Wiley, 2013.

KNAFLIC, C. N. **Storytelling with data: a data visualization guide for business professionals.** Hoboken: Wiley, 2015.

LARSON, D.; CHANG, V. **A review of business intelligence and analytics in small and medium enterprises**. Decision Support Systems, v. 135, 2020.

MACE, R. **The principles of universal design.** Raleigh: North Carolina State University, 1997.

MEDRI, W. **Análise Exploratória de Dados**. Londrina, 2011. Apostila. Disponível em: http://docs.ufpr.br/~benitoag/apostilamedri.pdf. Acesso em: 15 out. 2024.

MONTGOMERY, D. C. **Introduction to statistical quality control.** 7. ed. Hoboken: John Wiley & Sons, 2013.

NORMAN, D. A. **The design of everyday things.** Ed. rev. e aum. New York: Basic Books, 2013.

NOTEJAM. **Model View ViewModel (MVVM)**: Architecture in Android. 2024. 1 figura, color. Disponível em: https://notesjam.com/model-view-viewmodel-mvvm-architecture/. Acesso em: 19 jun. 2025.

OPPMANN, A-K. **BARC Data, BI & Analytics Trend Monitor 2022.** Würzburg: BARC, 2022. Infográfico. Disponível em: https://barc.com/data-bi-analytics-trend-monitor-2022-infographic/. Acesso em: 18 jun. 2025.

PAGAN, A. S.; SIMPLICIO, G. C.; REZENDE, E. J. C. User-centered design and its ethical principles as guidelines in teaching design. **Estudos em design**, Rio de Janeiro, RJ, v. 27, n. 1, p. 131 – 170, 2019. Disponível em: https://estudosemdesign.emnuvens.com.br/design/article/viewFile/680/368. Acesso em 12 Abr. 2025.

PERKEL, J. M. Python for data science: tools and trends. **Nature**, v. 605, n. 7908, p. 190-192, May 2022. DOI: 10.1038/d41586-022-01162-6.

PLOTLY. **Dash Documentation & User Guide.** Plotly: [2024]. Disponível em: https://dash.plotly.com/. Acesso em: 12 Abr. 2025.

SCHOBER, P.; BOER, C.; SCHWARTE, L. A. Correlation coefficients: appropriate use and interpretation. **Anesthesia & Analgesia**, Amsterdam, v. 126, n. 5, p. 1763-1768, May 2018. DOI: 10.1213/ANE.0000000000002864. Disponível em: https://journals.lww.com/anesthesia-analgesia/Fulltext/2018/05000/Correlation_Coefficients__Appropriate_Use_and.50.aspx. Acesso em: 20 abr. 2025.

SCHWABER, K.; SUTHERLAND, J. **The Scrum guide.** 2020. Disponível em: https://www.scrumguides.org/. Acesso em: 20 abr. 2025.

SIMOE, J. V. **TCC_Python.** Versão 1.2. [s.l.]: GitHub, 2025. Disponível em: https://github.com/CanaryRanger/TCC_Python.git. Acesso em: 12 set. 2025.

STACK EXCHANGE. **Does classic outlier detection assume normality?**. Disponível em: https://stats.stackexchange.com/questions/554631/does-classic-outlier-detection-assume-normality. Acesso em: 18 mai. 2025.

THE Complete Guide to Skewness and Kurtosis [por] Kartik Menon. Aug, 2024. 1 vídeo-aula (6:01 min), 15 de 24. Simplilearn. Disponível em: https://www.simplilearn.com/tutorials/statistics-tutorial/skewness-and-kurtosis. Acessado em: 18 jun. 2025.

TRYGVE, R. **The Model-View-Controller (MVC) Its Past and Present.** 2003 Disponível em: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=4ef90a7b9c1b1cd02acf273694e4059a70c7d198. Acesso em: 18 jun. 2025.

TUFTE, E. R. **The visual display of quantitative information.** 2. ed. Cheshire: Graphics Press, 2001.

TUKEY, J.W. **Exploratory Data Analysis.** Reading: Addison-Wesley Publishing Company, Apr. 1977.

WEXLER, S.; SHAFFER, J.; COTGREAVE, A. **The big book of dashboards:** visualizing your data using real-world business scenarios. Hoboken: John Wiley & Sons, 2017.

ZHANG, Y.; WANG, Z. Interactive data visualization for decision-making in business intelligence. **IEEE Transactions on Visualization and Computer Graphics**, v. 29, n. 4, p. 1234-1245, Apr. 2023. DOI: 10.1109/TVCG.2021.3114810.

BIBLIOGRAFIA CONSULTADA

DAVIES, L., & GATHER, U. (1993). The Identification of Multiple Outliers. **Journal of the American Statistical Association**, v. 88, Sep. 1993. DOI: https://doi.org/10.2307/2290763. Disponível em: https://www.jstor.org/stable/2290763. Acesso em: 25 Jun. 2025.

DBC docs. In: FACULTY AI. Disponível em: https://www.dash-bootstrap-components.com/docs/components. Acesso em: 25 Jun. 2025.

HAVLICEK, L. L.; PETERSON, N. L. **Robustness of the Pearson Correlation against Violations of Assumptions.** 1 ed. Perceptual and Motor Skills, v.43, 1976.

MODEL-view-presenter. In WIKIPÉDIA: a enciclopédia livre. [São Francisco, CA: Wikimedia Foundation, 2024]. Disponível em: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter. Acesso em: 27 jun. 2025.

PLOTLY. **Dash enterprise documentation.** Plotly: [2023]. Disponível em: https://dash.plotly.com/. Acesso em: 12 abr. 2025.

APÊNDICE A – Tabela de dados brutos da seção Emissões Atmosféricas.

Tabela 2 - Dados brutos extraídos com o botão de Extração da Seção “Emissões Atmosféricas”.

CD_MUN	NM_MUN	ANO	NM_MUN_variavel	VALOR	VARIAVEL
3552205	Sorocaba	1999	SOROCABA	998,1652	CO2
3552205	Sorocaba	2000	SOROCABA	1024,452	CO2
3552205	Sorocaba	2001	SOROCABA	959,8427	CO2
3552205	Sorocaba	2002	SOROCABA	999,7969	CO2
3552205	Sorocaba	2003	SOROCABA	1009,504	CO2
3552205	Sorocaba	2004	SOROCABA	1093,914	CO2
3552205	Sorocaba	2005	SOROCABA	1149,127	CO2
3552205	Sorocaba	2006	SOROCABA	1209,98	CO2
3552205	Sorocaba	2007	SOROCABA	1270,745	CO2
3552205	Sorocaba	2008	SOROCABA	1231,848	CO2
3552205	Sorocaba	2009	SOROCABA	1157,027	CO2
3552205	Sorocaba	2010	SOROCABA	1288,374	CO2
3552205	Sorocaba	2011	SOROCABA	1196,95	CO2
3552205	Sorocaba	2012	SOROCABA	1112,428	CO2
3552205	Sorocaba	2013	SOROCABA	1098,638	CO2
3552205	Sorocaba	2014	SOROCABA	2133,026	CO2
3552205	Sorocaba	2015	SOROCABA	446,2241	CO2
3552205	Sorocaba	2016	SOROCABA	1197,287	CO2
3552205	Sorocaba	2017	SOROCABA	425,1449	CO2
3552205	Sorocaba	2018	SOROCABA	1094,895	CO2
3552205	Sorocaba	2019	SOROCABA	1070,964	CO2
3552205	Sorocaba	2020	SOROCABA	979,5095	CO2
3552205	Sorocaba	2021	SOROCABA	1036,832	CO2
3552205	Sorocaba	2022	SOROCABA	1096,74	CO2

Legenda: CD_MUN: Código de Município; NM_MUN: Nome do Município; ANO: Ano selecionado; NM_MUN_VARIAVEL: Nome do Município tratado para leitura; VALOR: Quantidade de CO2 em Kg/Km²; VARIAVEL: Métrica, no caso CO2.

Fonte: Autoria Própria.

APÊNDICE B – Tabela de dados brutos da seção Correlação.

Tabela 3 - Dados brutos extraídos com o botão de extração na seção de Correlação.

CD_MUN	NM_MUN	ANO	CO2	Fluorados	Densidade demográfica
3552205	Sorocaba	1999	998,1652	2233,189	1037,679
3552205	Sorocaba	2000	1024,452	2291,518	1096,907
3552205	Sorocaba	2001	959,8427	2219,765	1131,094
3552205	Sorocaba	2002	999,7969	2254,503	1150,44
3552205	Sorocaba	2003	1009,504	2284,829	1175,287
3552205	Sorocaba	2004	1093,914	2499,98	1227,446
3552205	Sorocaba	2005	1149,127	2598,862	1256,312
3552205	Sorocaba	2006	1209,98	2757,887	1284,96
3552205	Sorocaba	2007	1270,745	2913,187	1242,924
3552205	Sorocaba	2008	1231,848	2888,189	1281,057
3552205	Sorocaba	2009	1157,027	2738,446	1298,842
3552205	Sorocaba	2010	1288,374	2979,38	1304,406
3552205	Sorocaba	2011	1196,95	2917,173	1319,875
3552205	Sorocaba	2012	1112,428	2715,234	1335,25
3552205	Sorocaba	2013	1098,638	2698,823	1398,688
3552205	Sorocaba	2014	2133,026	2062,241	1416,373
3552205	Sorocaba	2015	446,2241	2121,153	1433,56
3552205	Sorocaba	2016	1197,287	2976,731	1450,37
3552205	Sorocaba	2017	425,1449	2432,95	1466,796
3552205	Sorocaba	2018	1094,895	2619,915	1491,948
3552205	Sorocaba	2019	1070,964	2564,967	1510,158
3552205	Sorocaba	2020	979,5095	2348,11	1527,894
3552205	Sorocaba	2021	1036,832	2485,53	1545,612
3552205	Sorocaba	2022	1096,74	2629,142	1545,612

Legenda: CD_MUN: Código de Municipio; NM_MUN: Nome do Municipio; ANO: Ano selecionado; CO2: Quantidade de CO2 por Kg/Km²; Fluorados: Quantidade de Fluor por Kg/Km²; Densidade Demográfica: Quantidade de Hab/Km².

Fonte: Autoria Própria.