PyPAH: pipeline incremental e dashboard analítico para a Produção Ambulatorial do SUS no Ceará

Guilherme M. Bessa[1], **Abel Brasil R. Silva**[2]

[1] Departamento de Computação – Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil

[2]Complexo Hospitalar UFC/HU Brasil
Fortaleza – CE – Brasil

Resumo. *A disponibilidade de dados públicos em saúde abre oportunidades relevantes para análise, monitoramento e apoio à tomada de decisão. Entretanto, a exploração dessas bases ainda é frequentemente limitada por desafios como grande volume, heterogeneidade, dificuldade de tratamento e baixa acessibilidade para usuários não especializados. Neste contexto, surge o PyPAH, uma solução desenvolvida em Python para automatização da extração, transformação, armazenamento e visualização de dados da Produção Ambulatorial do Sistema de Informações Ambulatoriais do SUS (SIA/DATASUS), com recorte para o estado do Ceará.*

A proposta adota uma arquitetura em camadas inspirada no padrão medallion, organizada em Bronze, Silver e Gold, combinando processamento incremental, armazenamento em formato Parquet, consolidação dos dados com DuckDB, persistência em nuvem por meio do Cloudflare R2, disponibilização via API REST com FastAPI e exploração interativa por meio de um dashboard em Streamlit. O sistema foi projetado para processar mensalmente novos arquivos publicados no FTP do DATASUS, reduzindo reprocessamentos desnecessários e mantendo uma base consolidada atualizada para consulta analítica. Além dos filtros e gráficos pré-definidos, o dashboard incorpora um agente conversacional baseado em LLM, construído com LangChain, que permite consultas em linguagem natural sobre os dados, ampliando o acesso analítico para além das visualizações fixas.

Como contribuição, é demonstrado uma arquitetura organizada e pronta para que outros desenvolvedores possam replicar a transformação de dados públicos de saúde em produto analítico acessível, com baixo custo operacional e boa separação entre ingestão, tratamento, agregação, visualização e consulta em linguagem natural.

Material Complementar: Link para vídeo e repositório

1. Introdução

A crescente disponibilidade de dados públicos tem ampliado o uso de soluções analíticas para apoiar pesquisa, auditoria e gestão. Na saúde, esses dados permitem acompanhar a distribuição da produção assistencial, identificar desigualdades regionais e subsidiar decisões de planejamento.

No Brasil, o Departamento de Informática do Sistema Único de Saúde (DATASUS) disponibiliza bases públicas amplamente utilizadas para monitoramento do

Sistema Único de Saúde (SUS). Entre elas, o Sistema de Informações Ambulatoriais (SIA)[DATASUS 2026a] reúne registros mensais da produção ambulatorial. Entretanto, o uso direto desses dados ainda é dificultado pelo formato dos arquivos, pela necessidade de etapas de tratamento e pela ausência de uma camada analítica voltada à exploração interativa[DATASUS 2026b].

Neste contexto, este trabalho apresenta o PyPAH, uma solução desenvolvida em Python para automatizar a ingestão, transformação, consolidação e visualização dos dados da Produção Ambulatorial do SIA/DATASUS, com foco no estado do Ceará. A arquitetura emprega processamento incremental, armazenamento em Parquet, consolidação por DuckDB, persistência em Cloudflare R2, disponibilização por API REST em FastAPI e exploração por dashboard em Streamlit, complementado por um agente conversacional baseado em LLM.

Como principal contribuição, o PyPAH demonstra uma arquitetura modular capaz de transformar dados públicos de saúde em um produto analítico de baixo custo operacional, reduzindo reprocessamentos e facilitando sua utilização por usuários técnicos e não técnicos.

2. Base de dados

Os dados utilizados são provenientes do Sistema de Informações Ambulatoriais (SIA/DATASUS), considerando a Produção Ambulatorial do estado do Ceará. Os arquivos são disponibilizados mensalmente pelo DATASUS em formato `.dbc`, por meio de repositório FTP público.

A escolha dessa base deve-se ao seu caráter oficial, à atualização periódica e à adequação para análises temporais da produção ambulatorial. Durante o processamento, os arquivos são convertidos para Parquet[Apache Software Foundation 2026], permitindo maior eficiência de leitura e armazenamento. Também são incorporadas tabelas auxiliares para traduzir códigos em descrições legíveis, facilitando a exploração dos dados na aplicação final.

3. Arquitetura da solução

O PyPAH adota uma arquitetura em camadas que separa as etapas de extração, transformação, agregação, armazenamento e consumo dos dados, conforme ilustrado na Figura 3.

3.1. Pipeline em camadas

A solução segue uma arquitetura inspirada no modelo medallion:

Bronze: realiza o download dos arquivos `.dbc` do DATASUS e sua conversão para Parquet, preservando os dados originais.

Silver: aplica limpeza, seleção de atributos, padronização dos campos e criação de informações derivadas, produzindo uma base tratada.

Gold: agrega os registros por período e disponibiliza uma estrutura otimizada para consultas analíticas pela API e pelo dashboard.

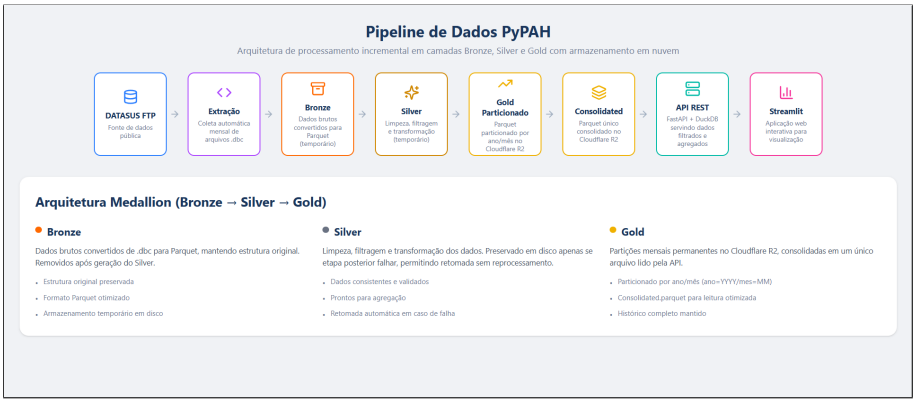

Figura 1. A Figura 1 ilustra a visão geral do fluxo.

3.2. Processamento incremental

O pipeline processa apenas os meses ainda não presentes na base analítica. Essa estratégia reduz tempo de execução, custo computacional e evita reprocessamentos desnecessários, tornando a atualização mensal mais eficiente.

3.3. Armazenamento e disponibilização

As camadas Bronze e Silver são processadas localmente durante a execução do pipeline e descartadas após sua conclusão, permanecendo apenas em caso de falha para permitir retomada. A camada Gold é persistida em Cloudflare R2[Cloudflare 2026], tanto em partições mensais quanto em um arquivo consolidado único utilizado pela API. O consolidado concentra todo o histórico disponível, ele é regenerado a partir das partições apenas quando novos meses são processados. O dashboard Streamlit consome esses dados por meio da API REST.

4. Implementação do sistema

O PyPAH foi desenvolvido em módulos independentes, separando pipeline de processamento, API e interface gráfica. Essa organização facilita manutenção, evolução e reutilização dos componentes.

4.1. Extração e transformação

O processamento incremental compara os meses já armazenados no Cloudflare R2 com os disponíveis no FTP do DATASUS, considerando a defasagem natural de publicação da base. Apenas arquivos ainda não processados são baixados, convertidos de `.dbc` para Parquet e encaminhados para a etapa de transformação.

Na camada Silver são selecionadas as colunas relevantes da Produção Ambulatorial: `PA_MUNPCN` (município do paciente), `PA_CODUNI` (estabelecimento de saúde), `PA_PROC_ID` (procedimento realizado) e as métricas de produção e aprovação `PA_VALPRO`, `PA_VALAPR`, `PA_QTDPRO` e `PA_QTDAPR`. São normalizados os atributos e criados campos auxiliares para análise. Também são preservados artefatos intermediários em caso de interrupção do processamento, evitando a repetição das etapas já concluídas.

4.2. Camada analítica e agregações

A camada Gold armazena os dados já agregados, reduzindo o custo das consultas realizadas pela API. Os registros são organizados por estabelecimento, procedimento e período, contendo as principais métricas de produção e aprovação. Tabelas de dimensão armazenadas separadamente traduzem códigos de estabelecimentos e procedimentos em descrições legíveis, permitindo que tanto o dashboard quanto o agente conversacional utilizem uma representação mais compreensível dos dados.

4.3. Persistência em Cloudflare R2

A camada Gold é armazenada no Cloudflare R2 devido à compatibilidade com a API S3 e ao baixo custo operacional. Os arquivos temporários permanecem apenas durante o processamento, enquanto as partições mensais e o arquivo consolidado são mantidos no armazenamento remoto para consumo pela API.

4.4. Organização modular

O sistema foi dividido em quatro módulos principais:

- Pipeline: ingestão, transformação e agregação dos dados;
- API: disponibilização dos dados por serviços REST;
- Streamlit: interface de visualização;
- Docker: padronização do ambiente de execução.

Essa separação reduz o acoplamento entre os componentes e facilita futuras extensões.

5. API e dashboard

A API foi implementada com FastAPI e utiliza DuckDB[DuckDB 2026, FastAPI 2026] para consultar diretamente os arquivos Parquet consolidados. Essa abordagem elimina a necessidade de um banco de dados tradicional e fornece respostas adequadas ao consumo pelo dashboard desenvolvido em Streamlit.

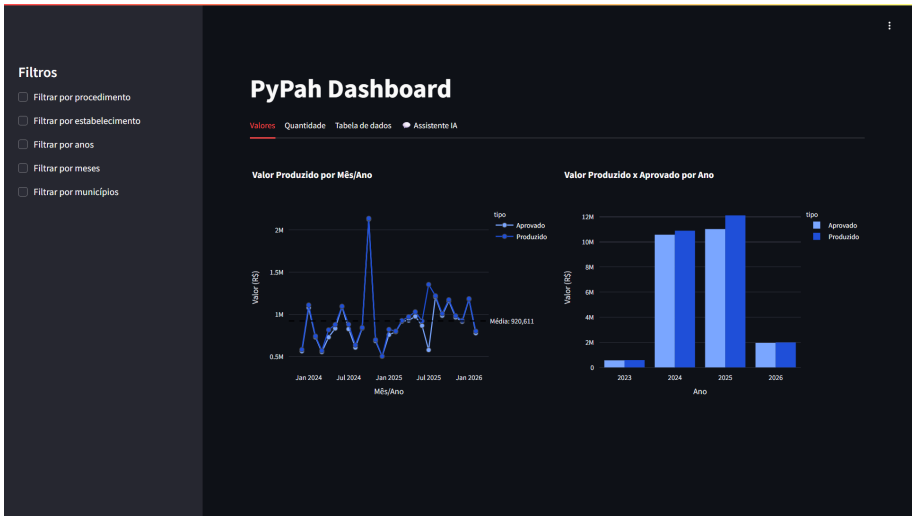

Figura 2. Figura 2: Interface do dashboard sem a aplicação de filtros.

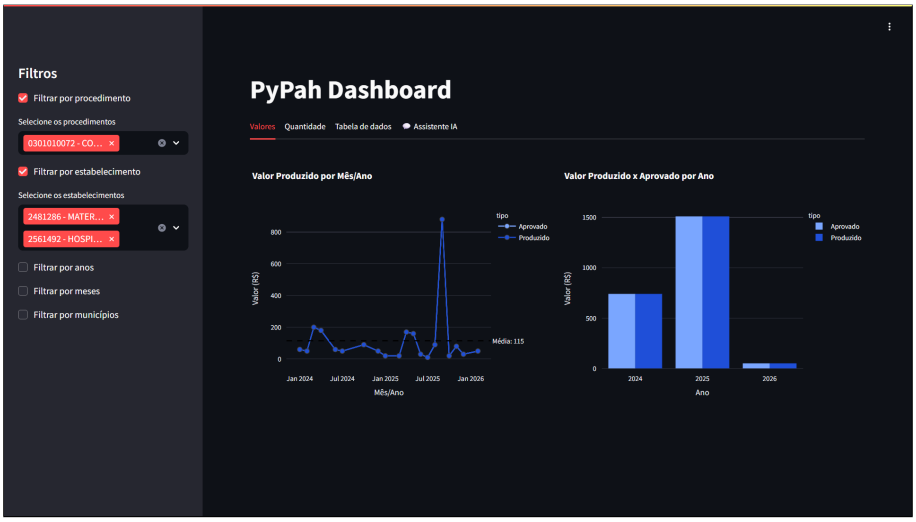

Figura 3. Figura 3: Interface do dashboard com filtros aplicados.

5.1. Endpoints disponibilizados

A API disponibiliza endpoints para consulta das dimensões (anos, meses, municípios, estabelecimentos e procedimentos) e um endpoint principal responsável pela recuperação dos dados agregados com filtros opcionais. As respostas são retornadas em formato adequado para consumo direto pelo dashboard.

5.2. Interface em Streamlit

O dashboard oferece filtros por período, município, estabelecimento e procedimento, além de gráficos e tabelas construídos a partir dos dados retornados pela API. A interface busca simplificar a exploração da base sem exigir conhecimento em SQL ou manipulação direta dos arquivos do DATASUS.

5.3. Agente conversacional com LangChain

Como complemento às visualizações do dashboard, foi desenvolvido um agente conversacional utilizando LangChain e o modelo Llama 3.3 70B, disponibilizado pela Groq. O agente permite consultas em linguagem natural sobre os dados filtrados, mantendo o contexto da conversa e executando análises sobre o conjunto de dados carregado. Além das consultas dinâmicas, o agente fornece estatísticas descritivas, resumos da base e análises agregadas, apresentando os resultados com os nomes legíveis de procedimentos e estabelecimentos obtidos das tabelas de dimensão. Dessa forma, amplia-se a flexibilidade de exploração dos dados sem exigir conhecimento da estrutura interna da base.

6. Resultados e discussão

Os resultados obtidos demonstram que a combinação entre processamento incremental, armazenamento em Parquet e consultas com DuckDB fornece uma solução adequada para exploração analítica de dados públicos em saúde. A arquitetura reduz reprocessamentos, organiza o fluxo de dados em camadas e simplifica a atualização mensal da base.

Sob o ponto de vista funcional, o PyPAH disponibiliza consultas interativas sobre a produção ambulatorial do Ceará por meio de filtros, gráficos e tabelas. A utilização de

tabelas de dimensão para traduzir códigos em descrições legíveis melhora a experiência do usuário, enquanto o agente conversacional amplia as possibilidades de exploração ao permitir consultas em linguagem natural sem exigir conhecimento da estrutura dos dados.

Como o sistema foi desenvolvido para atender demandas semelhantes às já existentes no Complexo Hospitalar da UFC, o trabalho teve como foco a implementação da arquitetura e da solução computacional, dispensando uma avaliação formal de usabilidade nesta etapa.

6.1. Métricas do produto final

- do FTP a disponibilização dos dados em 10-15min por mês;
- 27 meses atualmente por escolha, possível escalonar;
- tamanho dos arquivos por mês. Bronze(50Mb); Silver(30Mb); Gold(1.5Mb);
- tempo médio de resposta da API(instantâneo se em cache, senão até 15s);
- arquivo bruto: 2Mx65 por mês, consolidado: 3Mx26 em 27 meses.
- atualmente é usado um banco truncado de 30k linhas para permitir o deploy gratuito no Render.

7. Conclusão

Este trabalho apresentou o PyPAH, uma solução em Python para automatizar a ingestão, transformação, consolidação e disponibilização dos dados da Produção Ambulatorial do SIA/DATASUS. A arquitetura modular combina processamento incremental, armazenamento em Parquet, consultas com DuckDB e disponibilização por API e dashboard interativo, reduzindo o esforço necessário para atualização e exploração dos dados.

Como principal contribuição, o trabalho demonstra uma implementação prática de engenharia de dados aplicada ao contexto da saúde pública, incluindo suporte a consultas em linguagem natural por meio de um agente conversacional integrado ao dashboard. A organização em módulos independentes favorece manutenção, reutilização e expansão da solução para outras bases do DATASUS e diferentes recortes geográficos. Como trabalhos futuros, destacam-se a ampliação das visualizações analíticas, a incorporação de novos conjuntos de dados do DATASUS e a expansão da cobertura para outros estados brasileiros.

Uso de IA

Os autores fizeram uso de LLM's para assistência na revisão e estruturação do texto.

Referências

[Apache Software Foundation 2026] Apache Software Foundation (2026). Parquet documentation. Acesso em: 06 mai. 2026.

[Cloudflare 2026] Cloudflare (2026). R2 documentation. Acesso em: 06 mai. 2026.

[DATASUS 2026a] DATASUS (2026a). Siasus. Acesso em: 06 mai. 2026.

[DATASUS 2026b] DATASUS (2026b). Transferência de arquivos. Acesso em: 06 mai. 2026.

[DuckDB 2026] DuckDB (2026). Duckdb documentation. Acesso em: 06 mai. 2026.

[FastAPI 2026] FastAPI (2026). Fastapi documentation. Acesso em: 06 mai. 2026.