Final Presentation — Cloud-Native Data Pipeline (GDP per Capita x CO_2 per Capita)

Este documento faz o walkthrough explicativo do projeto, cobrindo: (1) explicação do output analítico, (2) insights identificados, (3) como o output foi produzido a partir do dataset curado, (4) arquitetura em alto nível, (5) decisões técnicas e trade-offs, e (6) desafios enfrentados e como foram resolvidos.

1) Explicação do Output Analítico

O projeto gera dois artefatos principais a partir do dataset curado:

- gdp_vs_co2_scatter.png
  - Dispersão do ano de 2023 com regressão linear e R^2.
  - Eixos: X = `gdp_per_capita_usd`, Y = `co2_tons_per_capita`.
  - Cor: intensidade baseada em `co2_per_1000usd_gdp`.
  - Outliers (por resíduo da regressão) são anotados automaticamente.
  - Local (execução local): `analysis/gdp_vs_co2_scatter.png`.

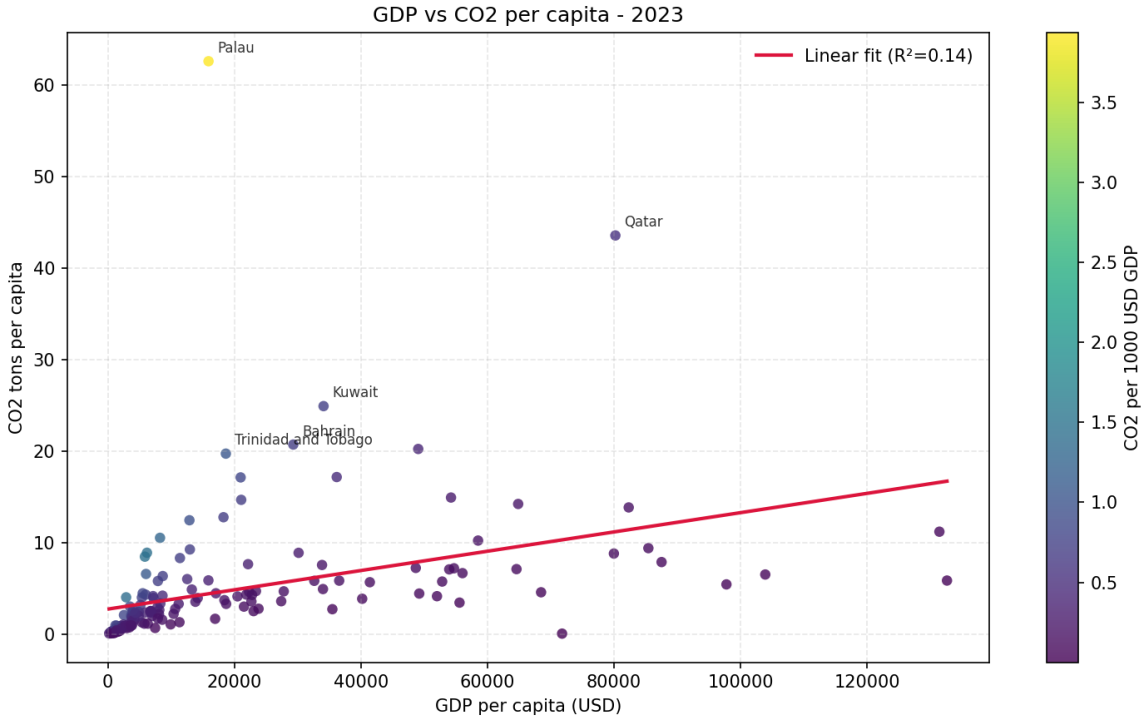

Figura 1 — Dispersão PIB per capita x CO_2 per capita para o ano de 2023

- correlation_summary.csv
  - CSV com uma linha por ano (2000 e 2023), contendo:
    - `year`
    - `pearson_correlation_gdp_co2`
    - `top5_countries_highest_co2_per_1000usd_gdp`
    - `top5_countries_lowest_co2_per_1000usd_gdp`
  - Local (execução local): `analysis/correlation_summary.csv`.

Ano	Correlação (Pearson)	Top 5 — maior CO2/1000 USD	Top 5 — menor CO2/1000 USD
2000	0,3831	Palau 14,192; Turcomenistão 13,742; Ucrânia 11,281; Uzbequistão 9,475; Mongólia 7,920	Groenlândia 0,001; Ilhas Faroé 0,002; Bermudas 0,042; Granada 0,116; Belize 0,118
2023	0,3799	Palau 3,937; Mongólia 1,447; Líbia 1,439; Uzbequistão 1,393; Turcomenistão 1,277	Ilhas Faroé 0,001; Bermudas 0,044; Ilhas Cayman 0,055; Suécia 0,062; Irlanda 0,063

Tabela de correlação e ranking, anos 2000 e 2023

Observação: quando executado em nuvem (AWS Lambda), os artefatos são gravados em S3 sob `analytics/<YYYYMMDD>/` usando o `StorageAdapter` (mesma lógica de negócio).

2) Insights Identificados

A partir do arquivo `analysis/correlation_summary.csv` gerado pelo pipeline (última execução local registrada), destacamos:

- Correlação PIB per capita x CO_2 per capita (Pearson)
  - 2000: ≈ 0,3831 (correlação positiva moderada)
  - 2023: ≈ 0,3799 (correlação positiva moderada)

- Top 5 países com maior `CO2 por 1000 USD de PIB` (eficiência menor)

  - 2000: Palau 14,192; Turcomenistão 13,742; Ucrânia 11,281; Uzbequistão 9,475; Mongólia 7,920
  - 2023: Palau 3,937; Mongólia 1,447; Líbia 1,439; Uzbequistão 1,393; Turcomenistão 1,277

- Top 5 países com menor `CO2 por 1000 USD de PIB` (eficiência maior)
  - 2000: Groenlândia 0,001; Ilhas Faroé 0,002; Bermudas 0,042; Granada 0,116; Belize 0,118
  - 2023: Ilhas Faroé 0,001; Bermudas 0,044; Ilhas Cayman 0,055; Suécia 0,062; Irlanda 0,063

- Observações
  - Focamos especificamente nos anos de 2000 e 2023 porque são os dois únicos anos em que a tabela da Wikipedia possui colunas completas e confiáveis para CO_2 per capita comparável globalmente.
  - A correlação permaneceu moderada entre 2000 e 2023, sugerindo que renda média e emissões per capita guardam relação positiva, mas com forte variação entre países.
  - Nota-se queda acentuada dos maiores valores de `CO2 por 1000 USD de PIB` entre 2000 e 2023 (ex.: Palau de 14,192 para 3,937), possivelmente refletindo ganhos de eficiência e/ou variações setoriais/energéticas. Interpretações causais requerem investigação adicional.
  - A dispersão para o ano de 2023 evidencia outliers relevantes, úteis para aprofundar hipóteses por região, mix energético e estrutura produtiva.

3) Como o Output Foi Produzido a partir do Dataset Curado

- Dataset curado (camada CURATED)
  - Tabela: `curated/env_econ_country_year/year=<ano>/snapshot_date=<YYYYMMDD>/curated_econ_environment_country_year.parquet`.
  - Colunas principais: `country_code`, `country_name`, `year`, `gdp_per_capita_usd`, `co2_tons_per_capita`, `co2_per_1000usd_gdp` (derivada), além de metadados de ingestão.
  - Construção: join por `(country_code, year)` entre PROCESSED do World Bank (PIB per capita) e PROCESSED da Wikipedia (CO_2 per capita), calculando `co2_per_1000usd_gdp` quando `gdp_per_capita_usd > 0`.

- Geração dos artefatos analíticos
  - Scatter (2023): lê CURATED do ano 2023, descarta nulos/inválidos, plota dispersão, ajusta regressão linear e anota outliers por resíduo. Função: `src/analysis/econ_environment_analytics.py:build_gdp_vs_co2_scatter`.

- Correlação (2000 e 2023): lê CURATED para os anos, calcula correlação de Pearson entre `gdp_per_capita_usd` e `co2_tons_per_capita` e computa os top 5 mais/menos eficientes via `co2_per_1000usd_gdp`. Função: `src/analysis/econ_environment_analytics.py:build_correlation_summary`.
- Para evitar duplicidades entre snapshots, a leitura usa apenas o snapshot mais recente por ano (sem sobrescrever históricos), garantindo reprodutibilidade do run.

4) Arquitetura em Alto Nível

- Fontes de dados públicas
  - World Bank API — indicador `NY.GDP.PCAP.CD` (PIB per capita, US$ correntes).
  - Wikipedia — tabela de CO_2 `per capita` via crawler com guarda de revisão.

- Orquestração serverless (nuvem)
  - EventBridge agenda a execução (diária por padrão).
  - AWS Lambda (imagem container) executa o pipeline end-to-end.

- Camadas de dados (S3 / Data Lake)
  - RAW: JSONL com campos de auditoria e `record_hash`.
  - PROCESSED: Parquet tipado e particionado por `year`.
  - CURATED: Parquet com partições `year=<ano>/snapshot_date=<YYYYMMDD>`.
  - ANALYTICS: `analytics/<YYYYMMDD>/` (scatter e CSV de correlação).

- Metadados e incremental
  - DynamoDB guarda histórico de runs e checkpoints.
  - Checkpoints: `last_year_loaded_world_bank` (World Bank) e `last_revid_wikipedia_co2` (Wikipedia).

- Portabilidade
  - Adapters padronizam I/O e metadados: `StorageAdapter` (Local/S3) e `MetadataAdapter` (Local JSON/DynamoDB). O mesmo código roda localmente e em nuvem.

- Infra como código
  - Dockerfile (Lambda container), CloudFormation (Lambda + Role + EventBridge). Script `cloud/lambda/build_and_deploy.sh` empacota e publica.

- Diagrama da arquitetura

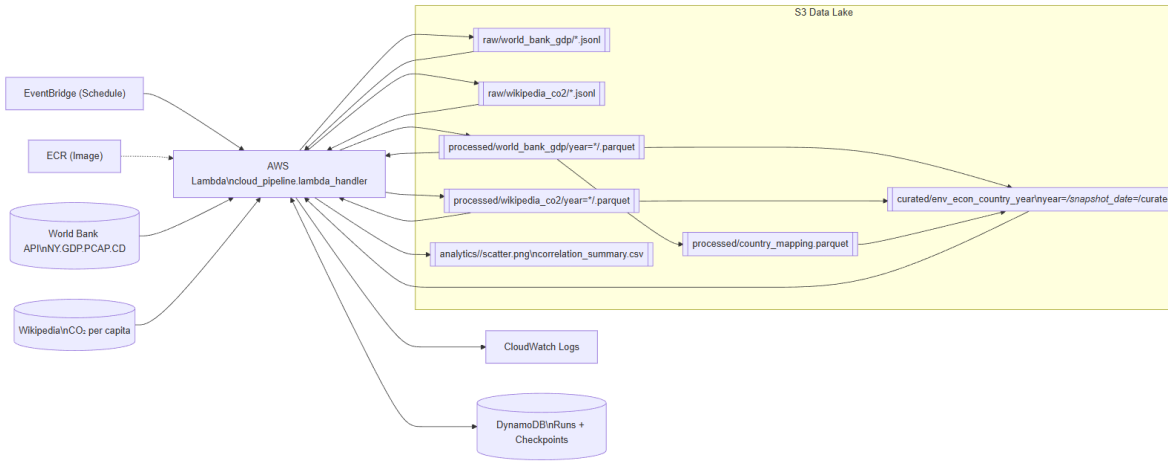

Figura 2 — Arquitetura em alto nível do pipeline (renderizada a partir de Mermaid).
Link para imagem: https://ibb.co/vvcYds05

5) Decisões Técnicas e Trade-offs

- Formatos e layout de dados
  - RAW em JSONL com `raw_payload`/`record_hash` para auditabilidade e idempotência simples.
  - PROCESSED/CURATED em Parquet tipado + particionamento por `year` (leitura eficiente, tipos explícitos, menor custo/armazenamento).
  - CURATED com `snapshot_date` imutável por run: facilita time-travel e auditoria ao custo de múltiplas versões por ano.

- Incrementalidade e idempotência
  - World Bank por ano com checkpoint `last_year_loaded_world_bank` (simple e previsível).
  - Wikipedia por guarda de revisão (`revid`): evita diffs por célula; quando muda, reprocessa a tabela inteira (trade-off: mais simples e robusto, custo de reprocesso completo em alterações raras).

- Adapters (Local ↔ S3/DynamoDB)
  - Abstração `StorageAdapter`/`MetadataAdapter` permite executar o mesmo fluxo localmente e na nuvem com zero mudança de regra de negócio.
  - Trade-off: camada de abstração adiciona código, mas reduz acoplamento e facilita testes/migração.

- Empacotamento e execução em nuvem
  - Lambda em imagem container para suportar deps nativas (numpy/pyarrow/matplotlib) e backend headless (`MPLBACKEND=Agg`).
  - Trade-off: imagem maior e pipeline de build (ECR) vs. zip mais leve sem deps nativas.

- Qualidade de dados e mapeamento de países
  - Normalização de nomes com função determinística e overrides CSV para exceções.
  - Trade-off: requer manutenção eventual do arquivo de overrides quando surgirem divergências.

- Observabilidade e custos
  - CloudWatch Logs e métricas implícitas da Lambda; sem dashboards/alertas dedicados (simplicidade vs. visibilidade avançada).

6) Desafios e Como Foram Resolvidos

- Estrutura volátil da tabela da Wikipedia
  - Desafio: mudanças de HTML e múltiplas "wikitables" possíveis.
  - Solução: heurísticas por legenda/keywords com fallback para a primeira `wikitable`; parse robusto de cabeçalhos/linhas; guarda `table_html` e JSON da tabela para rastreabilidade.

- Incrementalidade do World Bank
  - Desafio: evitar recarregar todo o histórico a cada run.
  - Solução: checkpoint simples por ano e filtro; metadados registram `rows_processed` e último ano carregado.

- Mapeamento de países (chaves instáveis entre fontes)
  - Desafio: divergência de nomes entre Wikipedia e World Bank.
  - Solução: `country_name_normalized` + Parquet de mapping derivado do World Bank, com overrides manuais para exceções.

- Cálculo de métricas derivadas e valores faltantes
  - Desafio: zeros/nulos em PIB e CO_2 contaminam métricas e gráficos.
  - Solução: `co2_per_1000usd_gdp` só quando `gdp > 0`; filtros de nulos; tratamento cuidadoso de parsing numérico.

- Robustez de rede e limites de API
  - Desafio: 429/5xx e variabilidade de latência nas fontes públicas.

  - Solução: retries leves com backoff exponencial + jitter para HTTP externo; SDK já cobre S3/DynamoDB.

- Empacotamento do ambiente científico na Lambda
  - Desafio: bibliotecas com dependências nativas (numpy/pyarrow/matplotlib).
  - Solução: imagem container baseada no runtime da Lambda + wheels binários; backend headless do matplotlib.

Referências úteis (arquivos do repositório)

- Código
  - `src/local_pipeline.py` — orquestração local (7 etapas).
  - `src/cloud_pipeline.py` — orquestração em nuvem (Lambda + S3 + DynamoDB via adapters).
  - `src/ingestion_api/world_bank_ingestion.py` — ingestão RAW do World Bank.
  - `src/crawler/wikipedia_co2_crawler.py` — crawler RAW da Wikipedia com guarda de revisão.
  - `src/transformations/world_bank_gdp_processed.py` — PROCESSED (World Bank).
  - `src/transformations/wikipedia_co2_processed.py` — PROCESSED (Wikipedia).
  - `src/transformations/curated_econ_environment_country_year.py` — join e escrita da camada CURATED.
  - `src/analysis/econ_environment_analytics.py` — geração de `gdp_vs_co2_scatter.png` e `correlation_summary.csv`.
  - `src/adapters/storage.py` e `src/adapters/metadata.py` — abstrações Local/S3 e Local/DynamoDB.

- Infraestrutura (AWS)
  - `cloud/lambda/Dockerfile` — imagem container da Lambda.
  - `cloud/lambda/template.yaml` — CloudFormation (Lambda, IAM Role, EventBridge, recursos opcionais).
  - `cloud/lambda/build_and_deploy.sh` — build/push ECR e deploy do stack.

- Evidências locais
  - `analysis/gdp_vs_co2_scatter.png` — gráfico de dispersão (2023).
  - `analysis/correlation_summary.csv` — correlação e rankings (2000/2023).
  - `curated/env_econ_country_year/year=2000|2023/snapshot_date=*/curated_econ_environment_country_year.parquet` — snapshots curados mais recentes por ano.