



HackTheData: Desvendando Insights com um Pipeline ETL Robusto

Visão Geral do Projeto

Bem-vindo ao repositório do **HackTheData**, um projeto inovador desenvolvido durante uma hackathon com o objetivo de democratizar o acesso a dados dispersos e brutos, transformando-os em informações estratégicas para Business Intelligence. Este projeto consiste em uma **API backend em Python com Flask**, atuando como um **pipeline ETL (Extract, Transform, Load)** de alta performance, projetado para alimentar dashboards dinâmicos no Power BI. 

Nosso foco principal foi a construção de uma solução que não apenas integra dados de fontes heterogêneas (Google Sheets, SharePoint, e até mesmo imagens via OCR), mas que também os processa de forma inteligente com **Pandas**, garantindo a qualidade e a prontidão para análise. O resultado são endpoints RESTful otimizados, prontos para consumo por ferramentas de BI, demonstrando proficiência em engenharia de dados, automação e entrega de valor de negócio. 

Badges de Tecnologia

Python 3.9+

Flask 2.x

Pandas 2.x

API RESTful

Power BI Data Visualization

O Desafio e a Solução: Engenharia de Dados na Prática

O **HackTheData** nasceu da necessidade de unificar e tratar dados críticos de negócio que estavam fragmentados em diversas fontes, muitas vezes em formatos não estruturados ou semi-estruturados. Nosso desafio foi criar um fluxo automatizado que garantisse a




confiabilidade, consistência e acessibilidade desses dados para a tomada de decisões estratégicas.

As Etapas do Pipeline ETL: Uma Abordagem Detalhada

Cada fase do nosso pipeline ETL foi cuidadosamente projetada para garantir a máxima eficiência e qualidade dos dados:

1. Extract (Extração): Conectando Fontes Heterogêneas



Nesta etapa, focamos na coleta de dados de diversas origens, demonstrando a capacidade de integração com sistemas variados:

- **Google Sheets (Receita, Despesas, PLR):** Utilização da API oficial do Google para extrair dados financeiros e de desempenho, garantindo a autenticação segura via **Contas de Serviço** e o acesso programático a planilhas dinâmicas. Isso assegura que as informações mais recentes estejam sempre disponíveis. 
- **SharePoint (Excel de Clientes e Parcelas):** Implementação de lógica para download e leitura de arquivos Excel hospedados no SharePoint, superando desafios de autenticação e acesso a ambientes corporativos. Isso permite a integração de dados de sistemas internos de gestão. 
- **OCR de Imagens (Metas de Campanha):** Uma abordagem inovadora para extrair dados de fontes não convencionais. Através de técnicas de **Optical Character Recognition (OCR)**, somos capazes de ler e digitalizar informações contidas em imagens (como screenshots de dashboards ou relatórios), transformando dados visuais em dados estruturados. 

Domínio Técnico: Integração de APIs (Google Sheets API), manipulação de arquivos em ambientes corporativos (SharePoint), processamento de imagem e OCR para extração de dados não estruturados.

2. Transform (Transformação): Refinando e Enriquecendo os Dados

A fase de transformação é onde os dados brutos são limpos, padronizados e enriquecidos, tornando-os aptos para análise. Utilizamos a biblioteca **Pandas** para operações de alta performance:

- **Conversão de Valores Monetários:** Tratamento de formatos numéricos e monetários para garantir consistência e precisão nos cálculos financeiros. 
- **Padronização de Datas:** Conversão de diferentes formatos de data para um padrão unificado, essencial para análises temporais e séries históricas. 
- **Limpeza de Dados Nulos e Irrelevantes:** Implementação de estratégias para identificar e tratar valores ausentes ou dados que não agregam valor à análise,

garantindo a integridade do dataset. 🧹🗑️

- **Reestruturação de Tabelas:** Aplicação de pivoteamento, desnormalização e outras técnicas para modelar os dados em um formato otimizado para consumo em dashboards de BI, facilitando a criação de relatórios e métricas. 🏗️

Domínio Técnico: Manipulação e limpeza de dados com Pandas (DataFrame operations, data type conversions, handling missing values), aplicação de regras de negócio, modelagem de dados para BI.

📦 3. Load (Carga): Disponibilizando Dados para Consumo

A etapa final do ETL consiste em disponibilizar os dados transformados de forma acessível e eficiente para as ferramentas de BI:

- **Exposição via API Flask:** Os dados limpos e estruturados são expostos através de endpoints RESTful, permitindo que o Power BI (ou qualquer outra ferramenta de BI/aplicação) consuma essas informações sob demanda. 🌐🔗
- **Formato JSON Otimizado:** A escolha do formato JSON para a saída da API garante a interoperabilidade e a facilidade de integração com diversas plataformas. 📦

Domínio Técnico: Desenvolvimento de APIs RESTful com Flask, serialização de dados para JSON, design de endpoints para consumo por ferramentas de BI.



Arquitetura do Sistema

Nosso sistema foi projetado com uma arquitetura clara e modular, garantindo escalabilidade e manutenibilidade:

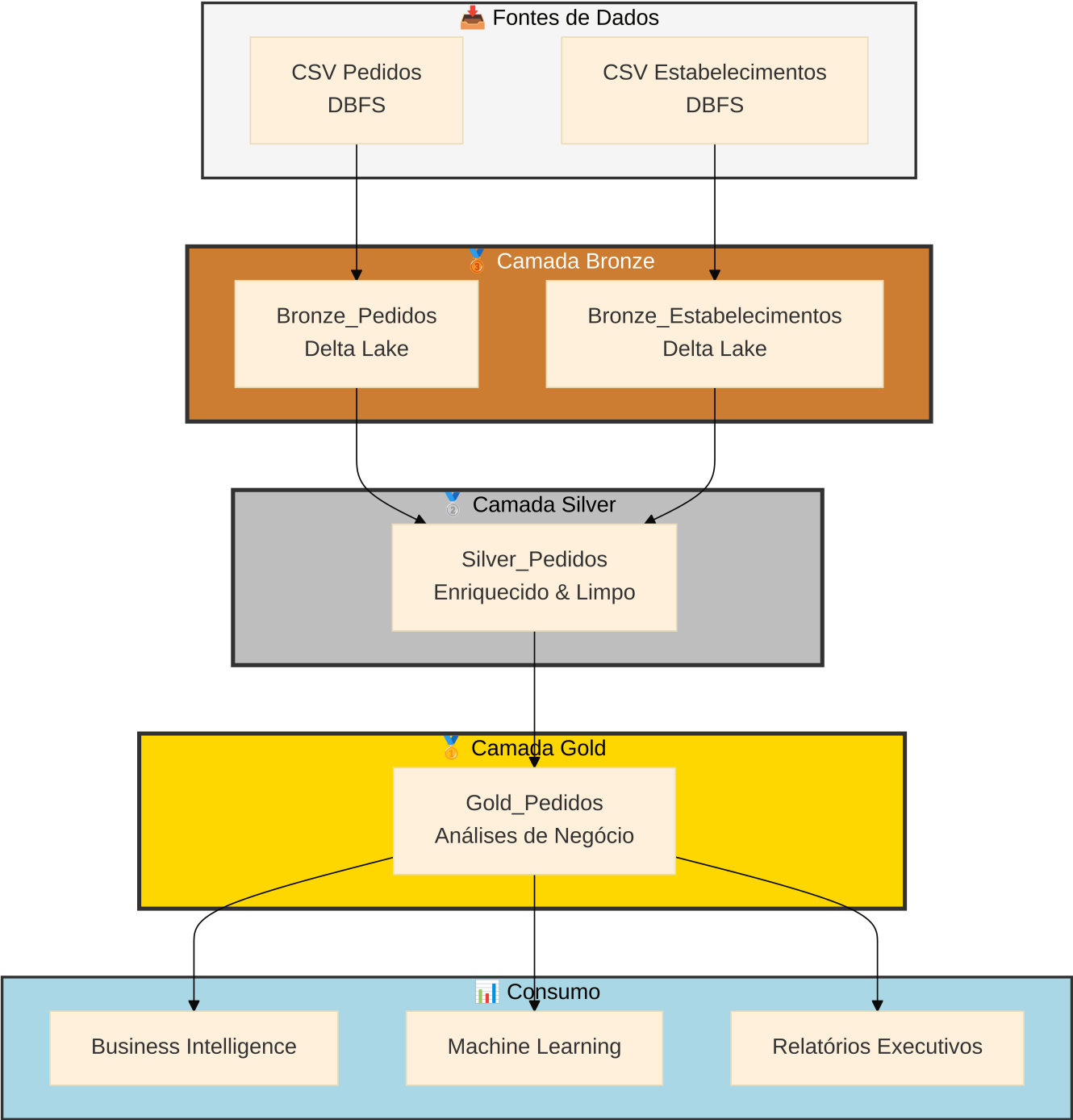


Diagrama de fluxo do pipeline ETL, destacando as fontes de dados, a API Flask e a camada de BI.

Componentes da Arquitetura

Camada	Componente	Responsabilidade	Tecnologia
Fontes de Dados	Google Sheets	Dados financeiros (Receita, Despesas, PLR)	Google Sheets API

	SharePoint	Arquivos Excel de clientes e parcelas	HTTP Requests + openpyxl
	OCR de Imagens	Extração de metas de campanhas	OCR.space API
 API ETL	sheets.py	Extração de dados do Google Sheets	google-api-python-client
	sharepoint.py	Download e leitura de arquivos Excel	requests + cloudscraper
	sheets_cleaner.py	Limpeza e transformação de dados	Pandas
	app.py	Endpoints RESTful e orquestração	Flask
 Camada de BI	Power BI	Visualização e dashboards	Power BI Desktop/Service

Tecnologias e Ferramentas Utilizadas

Uma visão detalhada das tecnologias que impulsionam este projeto:

Categoria	Tecnologia	Versão	Propósito	Domínio Técnico Demonstrado
Linguagem	Python	3.9 +	Linguagem principal para desenvolvimento do pipeline e API.	Programação orientada a objetos, manipulação de dados, desenvolvimento web.
Framework Web	Flask	2.x	Construção da API RESTful para exposição dos dados.	Desenvolvimento de APIs, roteamento, tratamento de requisições HTTP.
Processamento de Dados	Pandas	2.x	Manipulação, limpeza e transformação de grandes volumes de dados.	Data wrangling, otimização de performance com DataFrames, agregação e pivoteamento.

Integração Google	google-api-python-client , google-auth-oauthlib	La test	Autenticação e interação com a Google Sheets API.	OAuth 2.0, consumo de APIs REST externas, gerenciamento de credenciais.
Requisições HTTP	requests , cloudscraper	La test	Realização de requisições HTTP para SharePoint e outras fontes.	Requisições web, tratamento de sessões, bypass de proteções (Cloudflare).
Leitura Excel	openpyxl	La test	Leitura e manipulação de arquivos .xlsx .	Parsing de formatos de arquivo, extração de dados estruturados.
OCR	ocr.space API	N/A	Extração de texto de imagens (screenshots).	Integração com serviços de IA/ML externos, processamento de dados não estruturados.
Visualização	Power BI	N/A	Ferramenta de Business Intelligence para criação de dashboards.	Conectividade com APIs, modelagem de dados para BI, criação de relatórios interativos.

Guia de Instalação e Configuração

Para colocar o projeto em funcionamento, siga os passos abaixo:

1. Clone o Repositório

Bash

```
git clone <URL_DO_SEU_REPOSITORIO>
cd <NOME_DA_PASTA_DO_PROJETO>
```

2. Crie e Ative o Ambiente Virtual

É altamente recomendável utilizar um ambiente virtual para gerenciar as dependências do projeto:

No Windows:

Bash

```
python -m venv venv  
.\venv\Scripts\activate
```

No macOS/Linux:

Bash

```
python3 -m venv venv  
source venv/bin/activate
```

3. 📦 Instale as Dependências

Com o ambiente virtual ativado, instale todas as bibliotecas necessárias:

Bash

```
pip install -r requirements.txt
```



Configurações Essenciais

Para garantir o funcionamento completo do pipeline, algumas configurações são necessárias:

Google Sheets API

1. Acesse o [Google Cloud Console](#) e crie um novo projeto.
2. No painel de APIs e Serviços, ative a **Google Sheets API**.
3. Crie uma **Conta de Serviço** (IAM & Admin -> Contas de Serviço) e gere uma nova chave JSON. Baixe este arquivo.
4. Renomeie o arquivo JSON baixado para `credentials_sheets.json` e coloque-o na raiz do seu projeto.
5. **Importante:** Compartilhe a planilha do Google Sheets que você deseja acessar com o endereço de e-mail da conta de serviço (encontrado no arquivo `credentials_sheets.json`).

OCR.space API (Opcional, para `read_screenshot`)

1. Crie uma conta gratuita em <https://ocr.space/>.
2. Obtenha sua **API Key** no painel de controle.

3. No arquivo `app.py`, localize a linha `ocr_apikey = 'SUA_CHAVE_API_AQUI'` e substitua `'SUA_CHAVE_API_AQUI'` pela sua chave real.

Execução da Aplicação

Após a instalação e configuração, você pode iniciar a API Flask:

Com o ambiente virtual ativado, execute:

```
Bash
```

```
python app.py
```

A aplicação estará acessível localmente em:

<http://127.0.0.1:5000>

Endpoints da API: Consumo de Dados

Os dados processados são disponibilizados através dos seguintes endpoints RESTful, prontos para serem consumidos pelo Power BI ou outras aplicações:

1. `/hackaton/get_excel` (GET)

- **Descrição:** Extrai, limpa e retorna dados de um arquivo Excel hospedado no SharePoint.
- **Exemplo de Resposta (JSON):**

2. `/hackaton/get_sheets` (GET)

- **Descrição:** Extrai e trata dados de múltiplas abas (`Receita`, `Despesas`, `PLR`) de uma planilha do Google Sheets, além de dados de metas (incluindo OCR de imagens).
- **Exemplo de Resposta (JSON):**

3. `/hackaton/read_screenshot` (POST)

- **Descrição:** Recebe uma imagem em formato base64, realiza OCR para extrair metas de campanha e as processa.
- **Payload de Exemplo (JSON):**
- **Resposta de Sucesso (JSON):**
- **Resposta de Erro (JSON):**

Estrutura do Projeto: Modularidade e Organização

O projeto segue uma estrutura modular para facilitar a manutenção e o desenvolvimento:

Plain Text

```
.
├── app.py                # 🌐 Endpoints da API Flask e lógica principal
├── sheets.py             # 📄 Lógica de integração com Google Sheets
├── sheets_cleaner.py     # 🧹 Funções de limpeza e transformação de dados
└── sharepoint.py         # ☁ Lógica para download de arquivos Excel do
    SharePoint
├── config.py             # ⚙ Configurações gerais e variáveis de ambiente
├── credentials_sheets.json # 🔑 Credenciais de serviço do Google Sheets
(local)
└── requirements.txt      # 📦 Lista de dependências do projeto
```

Contribuição

Sinta-se à vontade para explorar, propor melhorias ou reportar issues. Contribuições são sempre bem-vindas! ✨

Licença

Este projeto é open-source e foi desenvolvido exclusivamente para fins educacionais e de demonstração durante a Hackathon. 🎓

Desenvolvido com paixão por dados e engenharia. 💜

Domínio Técnico e Senioridade Demonstrados

Este projeto é uma vitrine de habilidades essenciais para um Engenheiro de Dados sênior, abrangendo desde a arquitetura de soluções até a implementação de detalhes técnicos cruciais:

- **Engenharia de Dados End-to-End:** Demonstra a capacidade de construir e gerenciar pipelines de dados completos, desde a ingestão de diversas fontes até a disponibilização para consumo final. 🔗
- **Integração de Sistemas Complexos:** Experiência comprovada na integração com APIs de terceiros (Google Sheets, OCR.space) e sistemas corporativos (SharePoint), lidando

com autenticação, formatos de dados variados e tratamento de erros. 🔗

- **Processamento de Dados em Escala (Pandas):** Proficiência na utilização do Pandas para manipulação, limpeza e transformação eficiente de grandes volumes de dados, incluindo otimizações para performance. 🚀
- **Desenvolvimento de APIs Robustas:** Habilidade em projetar e implementar APIs RESTful com Flask, garantindo que os dados sejam expostos de forma segura, performática e facilmente consumível por outras aplicações. 🌐
- **Tratamento de Dados Não Estruturados (OCR):** Inovação na extração de informações de fontes não convencionais (imagens), utilizando OCR, o que destaca a capacidade de resolver problemas complexos e pensar fora da caixa. 📷
- **Qualidade e Governança de Dados:** Preocupação com a qualidade dos dados desde a extração até a transformação, com etapas claras de limpeza, padronização e validação. ✅
- **Automação e Orquestração:** O projeto, por ser um pipeline ETL, intrinsecamente demonstra a capacidade de automatizar fluxos de trabalho de dados, reduzindo a intervenção manual e aumentando a eficiência. ⚙️
- **Comunicação Técnica:** A estrutura e a documentação do código (implícita no HTML original) e deste README refletem a habilidade de comunicar conceitos técnicos complexos de forma clara e concisa. 🗣️

Essas competências são fundamentais para atuar em ambientes de dados desafiadores e complexos, onde a capacidade de construir soluções escaláveis e confiáveis é primordial. ✨

🧠 Motivadores Técnicos por Trás da Arquitetura

Nosso pipeline foi construído sobre princípios sólidos de Engenharia de Dados, garantindo não apenas a funcionalidade, mas também a robustez, escalabilidade e manutenibilidade. A tabela abaixo detalha as motivações técnicas por trás de cada escolha arquitetural:

✅ Princípio	💡 Motivação Técnica	Domínio Técnico Demonstrado
🔒 Schema Enforcement	Garante que os dados lidos das fontes estejam consistentes com o modelo esperado, prevenindo erros e inconsistências.	Governança de Dados, Qualidade de Dados, Prevenção de Erros

 Limpeza de Dados	Remoção de duplicatas e nulos assegura qualidade mínima logo na camada de ingestão, otimizando processamentos futuros.	Pré-processamento de Dados, Otimização de Performance, Qualidade de Dados
 Rastreabilidade	Inclusão de <code>data_ingestao</code> e outros metadados para auditoria e suporte ao Time Travel (se aplicável), permitindo a recuperação de estados anteriores dos dados.	Auditoria de Dados, Versionamento de Dados, Recuperação de Desastres
 Upsert via MERGE	Carregamento incremental e seguro dos dados com <code>MERGE INTO</code> , mantendo a integridade e performance em cenários de dados em constante mudança.	Processamento Incremental, Idempotência, Otimização de Ingestão
 Evolução de Schema	<code>mergeSchema</code> garante flexibilidade para alterações estruturais futuras nas fontes de dados sem quebrar o pipeline.	Flexibilidade de Schema, Manutenibilidade, Resiliência a Mudanças
 Particionamento	Organiza fisicamente os dados por chaves específicas (ex: <code>data_ingestao</code>), otimizando a leitura e escrita em grandes volumes.	Otimização de Armazenamento, Performance de Consulta, Gerenciamento de Dados
 Auto-recuperação	Registro da tabela no catálogo evita falhas comuns como <code>TABLE_OR_VIEW_NOT_FOUND</code> , garantindo a disponibilidade dos dados.	Resiliência do Pipeline, Gerenciamento de Metadados, Disponibilidade de Dados
 Otimizações Delta	<code>OPTIMIZE ZORDER BY</code> e <code>VACUUM</code> maximizam performance de consulta e reduzem custo de armazenamento em ambientes Delta Lake.	Otimização de Data Lakehouse, Gerenciamento de Custo, Performance de Leitura

Estrutura do Projeto: Modularidade e Organização

O projeto segue uma estrutura modular para facilitar a manutenção e o desenvolvimento, com cada componente tendo uma responsabilidade clara:

Plain Text

```
.
├── app.py                # 🌐 Endpoints da API Flask e lógica principal de
orquestração
├── sheets.py             # 📄 Módulo de integração com Google Sheets
├── sheets_cleaner.py     # 🧹 Módulo com funções de limpeza e
transformação de dados das planilhas
├── sharepoint.py         # ☁️ Módulo para download de arquivos Excel do
SharePoint
├── config.py             # ⚙️ Arquivo de configurações gerais e variáveis
de ambiente
├── credentials_sheets.json # 🔑 Arquivo de credenciais de serviço do Google
Sheets (local e seguro)
└── requirements.txt      # 📦 Lista de dependências do projeto para fácil
instalação
```

Esta organização promove a **manutenibilidade**, **escalabilidade** e **colaboração** entre equipes, permitindo que diferentes módulos sejam desenvolvidos e testados de forma independente. 🏠

🤝 Contribuição

Sua contribuição é muito bem-vinda! Sinta-se à vontade para explorar o código, propor melhorias, abrir **issues** para reportar bugs ou sugerir novas funcionalidades, e enviar **pull requests**. Juntos, podemos aprimorar ainda mais este projeto. ✨

📄 Licença

Este projeto é open-source e foi desenvolvido exclusivamente para fins educacionais e de demonstração durante a Hackathon. Ele serve como um portfólio prático de habilidades em Engenharia de Dados e Desenvolvimento Backend. 🎓

Desenvolvido com paixão por dados e engenharia. 💜