**Monitoramento do Sistema de Transporte Coletivo para o Município de São Paulo usando Apache Airflow**

**Visão Geral do Projeto**

O objetivo desse projeto, é propor uma estrutura de pipeline, para a coleta de dados, visando acompanhar o funcionamento do transporte público, manipular as ferramentas de extração e aplicar os conceitos de KPI. Os dados serão obtidos da API da sptrans(<https://www.sptrans.com.br/desenvolvedores/api-do-olho-vivo-guia-de-referencia/documentacao-api/#docApi-posicao>), usando o endoint Posição, e será carregado em um datalake, logo depois serão construídos visualizações sobre os dados obtidos.

Será feito um webscraping no site (<https://sistemas.sptrans.com.br/PlanOperWeb/detalheLinha.asp?TpDiaID=0&project=OV&lincod=669A-10>), com o objetivo de pegar a empresa e a região no qual a linha opera.

**Tecnologias Utilizadas**

* Apache Airflow,
* Python,
* data lake,
* apache spark para a manipulação de dados
* dash e plotly como ferramentas de visualização:

**Arquitetura do Fluxo de Trabalho**

Coleta de Dados:

O pipeline deverá rodar em um intervalo de 5 minutos, podendo ser modificado posteriormente, obedecendo as seguintes instruções:

* Verificação da conexão da API;
* Mandar email em caso de falha de conexão da api e encerrar o pipeline;
* Obtenção dos dados em caso de sucesso;
* Criação do datalake com base na data e hora de extração

Criação do data Lake:

O data lake será criado com a arquitetura medalhão (bronze, prata e ouro), sendo criado localmente, e seguirá o seguinte padrão:

Camada Bronze: Dentro da camada bronze, irá ser gravada um pasta marcando o ano no formato extração\_ano\_2023, logo depois, irá ser criada a pasta mês no formato extração\_mes\_08, e posteriormente, será criado a pasta do dia da extração no formato “extração\_dia\_26” e dentro da pasta, deverá ser gravado a requisição da api mostrando a hora que foi obtida, exemplo: req\_21\_44.

Hierarquia de exemplo:

extração\_ano\_2023 -> extração\_mes\_08 -> extração\_dia\_26 -> req\_21:46

Figura 1: Prosposta da estrutura do datalake

Camada silver: Irá seguir a mesma hierarquia da pasta da camada Bronze, com o adicional de inclusão da hierarquia Região – Empresa – linha – (registro de ônibus)

Hierarquia de exemplo:

extração\_ano\_2023 -> extração\_mes\_08 -> extração\_dia\_26 -> req\_21:46 -> Região –> Empresa –> linha –> (registro de ônibus)

Figura 2: Estrutura da camada Silver

**Estrutura do Projeto Apache Airflow**

O projeto irá seguir a seguinte estrutura

DAGs (Directed Acyclic Graphs):

Serão criados DAGs para representar os fluxos de trabalho. Cada DAG corresponderá a uma etapa do processo, como coleta, transformação, carregamento e análise.

Cada DAG conterá um conjunto de tarefas que serão executadas em uma sequência lógica.

Tarefas:

Cada DAG será composto por várias tarefas, que representarão as diferentes etapas do fluxo de trabalho.

Exemplos de tarefas: "CHECAR\_CONEXAO\_API", "Mandar Email”.

Após a etapa de transformação, será obtido as seguintes operações:

1 – Quantidade de ônibus em operação;

2 - Quantidade de ônibus em operação por empresa;

3 - Quantidade de ônibus em operação por região;

4 – Permitir a visualização de total de quilômetros por ônibus (ex: Um ônibus fez 600 km);

5 – Permitir a visualização de total de quilômetros por ônibus selecionado a Empresa;

6 – Gerar um mapa que marque a posição de um ônibus selecionado o dia e uma faixa de horário (Animação);

7 – Verificar ônibus que estão na garagem selecionando por empresa;

8 – O dashboard deverá funcionalidade de atualização em tempo real;

9 – O dashboard deverá disponibilizar um botão que permita atualizar manualmente.

10 – Criar relatório de armazenamento do data lake.

Requisito gráficos