Desafio técnico: Desenvolvedor (a) Backend - Produtos de dados

Tarefa 2: Arquitetura em nuvem

Nessa tarefa, você irá descrever uma arquitetura de solução em nuvem (Google Cloud, AWS ou Azure) para implantação da API desenvolvida. Não há necessidade de desenvolvimento de código ou criação de um projeto real em uma nuvem. O objetivo é criar uma apresentação que especifique e justifique a escolha das ferramentas e tecnologias em cada componente de sua arquitetura.

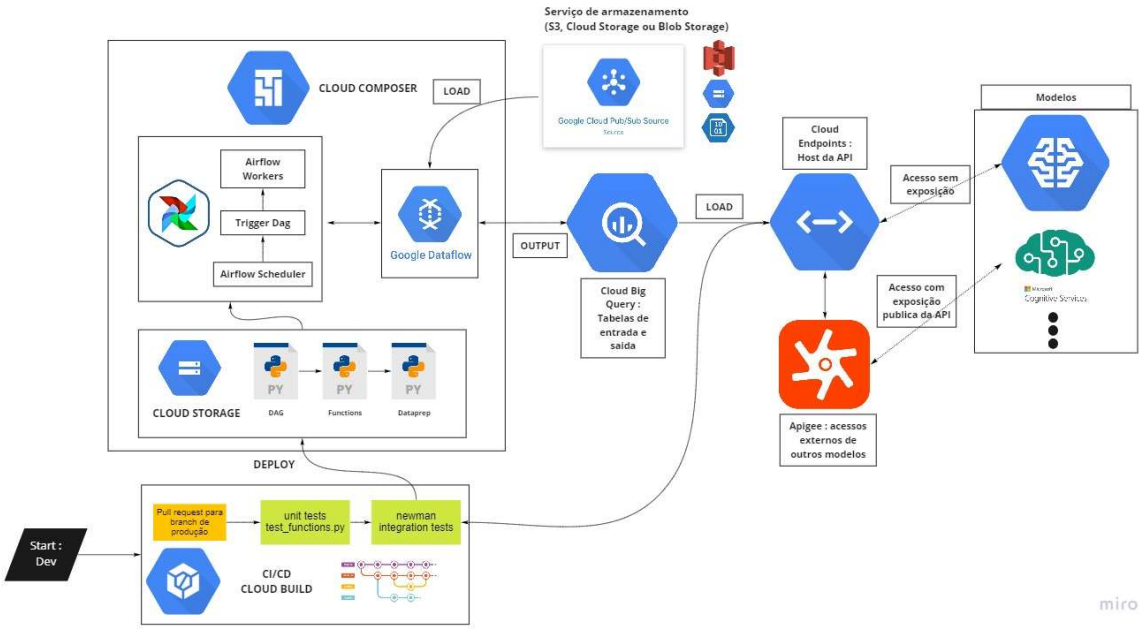
A arquitetura deve contemplar os seguintes requisitos e funcionalidades:

• O sistema será responsável por rodar um job diário para processamento de dados dos arquivos de audiência e disponibilidade consumidos de um serviço de armazenamento (S3, Cloud Storage ou Blob Storage). O conteúdo dos dois arquivos e resultado do script de pré-processamento deverá ser salvo em uma base de dados que servirá para consultas pela API desenvolvida;

• O job roda uma vez ao dia. O sistema não precisa ficar ligado o restante do tempo. Com o sistema desligado é necessário guardar seu estado de alguma forma;

• O sistema pode falhar no meio do processo e precisa voltar de onde parou; • O sistema precisa saber se a gravação no banco de dados ocorreu com sucesso;

• O ideal é ser serverless, se não for possível no momento, a solução deve evoluir para esse objetivo



CI/CD e GOVERNAÇA DE DESENVOLVIMENTO:

O início do desenvolvimento do projeto se dá por meio de um controle de CI/CD com regras de gitflow integradas ao Cloud Build, no bloco start.

Os desenvolvedores trabalham em branches de desenvolvimento e após homologado o código com os devidos testes é realizado um pull request do código para a branch de produção.

A ferramenta cloud build, é acionada por meio do pull-request de produção e realiza testes unitários de preprocessamento com stubs dos dataframes, e testes de integração com a API de ingestão e fornecimento dos dados.

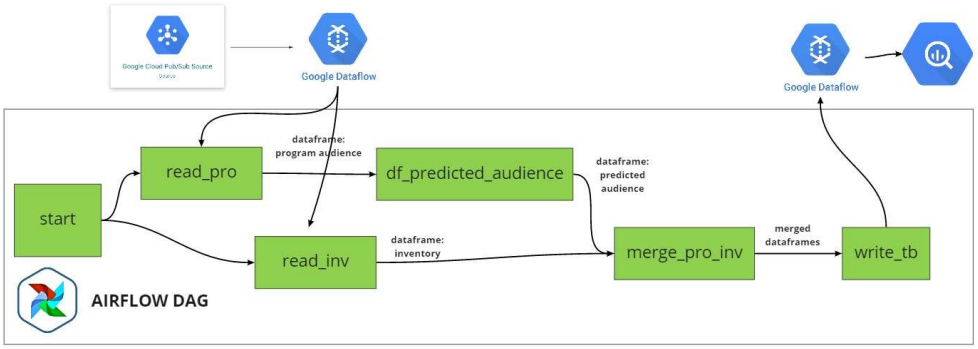
Após concluído os testes, os códigos python são salvos no storage de produção para o trabalho do cloud-composer

Se não concluído os testes o Cloud Build realiza um roll-back do pull request de código e dispara os alertas à equipe de desenvolvimento.

AUTOMAÇÃO E AGENDAMENTO COM O CLOUD-COMPOSER:

Na ferramenta cloud composer é desenvolvido uma DAG do Airflow que orquestrará as tasks, com agendamento diário.

O diagrama da dag se encontra abaixo:



A dag inicializa 2 trabalhos de leitura dos dados de audiência e inventário de tempo disponível por anúncio em paralelo nas tasks read_pro e read_inv, que faz a carga dos dados pelo pipeline Google Dataflow e Pub/Sub para dados em armazenamentos externos.

A task deixa o dataframe inventory que contêm os dados de tempo disponível por anuncio preparados, enquanto paralelamente a task df_predicted_audience calcula a mediana das estimativas de audiência.

A task merge_pro_inv é executada após a conclusão das 2 tasks anteriores, realizando um merge de tempo disponível por programa no dataframe inventory com a audiência estimada por programa no dataframe das estimativas.

Os registros que de qualquer um dos dataframes que não coincidirem no mesmo índice de programa, localidade e data serão preenchidos com um valor default de -1 ou NaN nas features que faltarem.

A última task write_tb através do Google Dataflow grava o dataframe final como uma tabela no Google Cloud Bigquery.

As tasks podem realizar os trabalhos com os dataframes utilizando o pacote pandas-gbq para melhor adaptação dos códigos desenvolvidos localmente com pandas.

HOSTING DA API DE CONSULTA DOS DADOS PROCESSADOS:

A api de consulta dos dados processados, similar ao código api.py na raiz deste projeto, pode ser desenvolvida e hospedada na ferramenta Cloud Endpoints, onde as devidas rotas de consulta por período ou por programa, acionam uma consulta integrada com o big-query.

Um proxy do apigee também pode ser desenvolvido e em suas rotas é possível implementar diretamente as consultadas ao big-query, sobrepondo a tarefa do cloud endpoint, porém a atuação ideal do apigee neste cenário seria para expor as consultas à modelos de otimização externos.

Os modelos de otimização se hospedados dentro do GCP do projeto como por exemplo na ferramenta Cloud ML, podem realizar as requisições dos dados, via API, diretamente na ferramenta Cloud Endpoints. Se as requisições das APIS forem de acessos externos e o endpoint tiver exposição pública, é ideal adicionar algumas camadas de segurança com o APIGEE para controle, segurança, e tratamento dos dados.

O apigee também pode atuar como uma camada interna para também um controle de tratamento dos dados nas requisições e algum processo de analytics das requisições no endpoint.