Desafio técnico:

Desenvolvedor (a) Backend - Produtos de dados

Contextualização do problema:

O time de desenvolvedores recebeu como trabalho, desenvolver um algoritmo para otimização da agenda de anúncios da TV Globo. O objetivo do algoritmo é atingir um volume de audiência específico, dentro de um período determinado da campanha na TV. Esse algoritmo de otimização necessita receber como entradas: informações de audiência e da grade de horários disponíveis para anúncios.

Sua contribuição nesse desafio será o fornecimento de dados para o algoritmo através de uma REST API em Python. Essa API será responsável por calcular a audiência e fornecer a quantidade de segundos disponíveis para anúncios em um determinado dia da semana, programa e localização.

Como resultado, espera-se a realização de duas tarefas nesse desafio: uma de desenvolvimento de código e outra conceitual.

- **tarefa 1:** desenvolvimento de uma REST API para fazer o cálculo das informações que serão fornecidas para o algoritmo.

- **tarefa 2:** descrição de uma arquitetura para implantação da REST API na nuvem.

Os detalhes de cada tarefa são descritos a seguir.

Tarefa 1: REST API

Para o desenvolvimento dessa tarefa são disponibilizados dois arquivos CSVs.

O arquivo em anexo **tvaberta_program_audience.csv** contém informações históricas de audiência de programas que já foram exibidos. O conteúdo desse arquivo é composto pelos seguintes atributos:

Atributo	Descrição do atributo
signal	localização (sinal) onde a audiência foi mensurada
program_code	código do programa
exhibition_date	data de exibição do programa
program_start_time	horário de início do programa
average_audience	audiência média durante a exibição do programa

O arquivo em anexo **tvaberta_inventory_availability.csv** contém informações do tempo disponível para anúncios em cada programa. O conteúdo desse arquivo é composto pelos seguintes atributos:

Atributo	Descrição do atributo
signal	localização (sinal) de exibição do programa
program_code	código do programa
date	data de exibição do programa
available_time	tempo disponível em segundos para anúncios

Os dados de **audiência histórica** serão utilizados para previsão de audiências em futuras exibições dos programas. Especificamente, espera-se que exibições futuras de um mesmo programa, em uma mesma localização, e em um mesmo dia da semana, possuam uma **audiência** similar à **mediana das últimas 4 exibições** desse mesmo programa, nessa mesma localização e no mesmo dia da semana.

Sendo assim, para que o algoritmo de otimização consiga operar com sucesso, é necessário calcular um novo atributo de dados contendo a **mediana da audiência** para servir como previsão da audiência para futuras exibições.

Para isso, crie um script em Python de pré-processamento de dados que faça a agregação dos dados em um único dataframe utilizando Pandas ou Spark e calcule esse novo atributo. O dataframe resultante deverá conter as seguintes informações:

Atributo	Descrição do atributo
signal	localização (sinal) de exibição do programa
program_code	código do programa
weekday	dia da semana de exibição do programa
available_time	tempo disponível em segundos para anúncios
predicted_audience	mediana das audiências nas 4 últimas exibições do programa por sinal e dia da semana

Para consultar os dados desse dataframe, desenvolva uma REST API com um endpoint para:

- **retornar dados por programa:** recebe como parâmetros um código de programa e uma data de exibição, retornando a quantidade de segundos disponíveis para anúncios na data especificada e a mediana calculada do programa.

- **retornar dados por período:** recebe como parâmetros uma data de início e uma data de fim de um período, retornando as quantidades de segundos disponíveis para os programas exibidos durante esse período e as medianas da audiência calculadas.

Por fim, crie um script com um cliente REST para testes do endpoint e inclua testes unitários para o código desenvolvido.

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

Para facilitar o entendimento da solução, desenhe um diagrama da arquitetura proposta.

Dicas e entrega do desafio

Disponibilizar o código fonte desenvolvido no exercício e apresentação. Sua solução deve ser organizada, documentada, legível e limpa. Tente ser sucinto, mas preze pela qualidade do código.

Importante:

Qualquer algoritmo/código gerado tem por finalidade atender ao exercício destinado ao processo de qualificação desta vaga. Portanto, é vetado o uso do material em publicações em sites (blogs, redes sociais, Github, etc), assim como em qualquer situação que não seja exclusiva a este processo de seleção. Os valores disponibilizados são fictícios.