Project specification

Introduction

What we would like you to do is implement a system that takes a group of campaigns, associated within a campaign group, and performs an optimisation on them based on some criteria - for example:

A group of 5 campaigns each have a budget of €20 per week.

- Campaign 1 budget: 20
- Campaign 2 budget: 20
- Campaign 3 budget: 20
- Campaign 4 budget: 20
- Campaign 5 budget: 20

The impressions per campaign are as follows:

- Campaign 1 impressions: 100
- Campaign 2 impressions: 200
- Campaign 3 impressions: 400
- Campaign 4 impressions: 200
- Campaign 5 impressions: 100

One approach to optimising the performance of campaigns would be to allocate budgets according to impressions, where impressions are seen as an indication of a campaign's success.

So then with impressions as the criteria of success, then budgets are distributed according to which campaigns perform better.

So, for example, the recommended budget allocation based on the impressions would be as follows:

Budgets[x] = (Impressions[x] / sum(impressions)) * sum(budgets)

So the new budget allocations based on the recommendations made by the optimisation system would be as follows:

- Campaign 1 budget: 10
- Campaign 2 budget: 20
- Campaign 3 budget: 40
- Campaign 4 budget: 20
- Campaign 5 budget: 10

Entities:

- CampaignGroup
    - Campaign
    - Optimisation
        - Recommendation

Domain:

CampaignGroup:
- Id
- Name

Campaign (associated with CampaignGroup)
- Id
- CampaignGroup Id
- Name
- Budget
- Impressions
- Revenue

Optimisation (associated with CampaignGroup)
- Id
- CampaignGroup Id
- Optimisation Status

Recommendation (associated with Optimisation and Campaign)
- Id
- Optimisation Id
- Campaign Id
- Recommended Budget

API:

- CampaignGroup
    - Retrieve all campaign groups

- OptimisationGroup
    - Retrieve latest Optimisation
    - Accept latest Recommendations

We will provide a group of campaigns in csv format (campaigns.csv) - you should load them into your system from the csv file.

We should be able to interrogate the API provided with curl or postman and

1. View all campaign groups (through a call to campaign group)
2. View all campaigns for a campaign group (through a call to campaign)
3. View latest optimisations for a campaign group (through a call to optimisation)
4. View latest recommendations for an optimisation (through a call to recommendation)
5. Apply / accept latest optimisation / recommendations to a campaign group / campaigns

You should only be able to apply recommendations once - after an optimisation has been applied, it should not be visible in the call to view the latest recommendations

Please provide the ability and documentation to allow us to run it and interrogate it with postman or curl.

Emphasis should be on a simple solution - nothing fancy required - once it does what is expected - well structured modular code with suitable tests is the most important thing.

Notes:

This project should be delivered in approximately a week's time, Spending anything from a few hours to a day. In follow-up conversation you might reflect on the unfinished parts.

Please use Java with Spring.
Please create a temporary (GitHub) repository for us to review the project.
Please include a README to describe how to build and run the application.
Please get in touch if you need additional information (kevin.osullivan@optily.com)

name	budget	impressions
2021-July-BOF-Books	2108	36358
3_299_BBQ_G-A_CV_SHP	674	29980
3_299_Bulbs_G-A_CV_SHP	2000	57561
3_299_Containers_G-A_OT_SHP	500	25864
3_299_Furniture_G-A_CV_SHP	1023	68640
3_299_Gifts_AOC_G-A_OT_SHP	500	32743
3_299_Lawn_Care_G-A_CV_SHP	4600	31023
3_299_Vegepod_G-A_CV_SHP	1325	15209
3_299_Wild_Bird_G-A_AOC_SHP	500	4931