Take Home Assignment

Introduction

First of all, we would like to thank you again for taking the time to tackle this take home assignment. It is very much appreciated. We have thought a lot about how to make this an open ended exercise, giving you the opportunity to decide how to shape the solution and scope. If you feel this hasn't worked, we would like to encourage you to give honest feedback so that we can improve

The Opportunity

Our mission at ENLYZE is to democratize manufacturing data. In our vision of the world, all departments and teams of manufacturing companies have easy access to high quality, automatically collected data to make better decision for their companies.

For most decision making processes, raw machine data does not help very much. White collar departments such as controlling or compliance work with KPIs to fuel their decision making processes. For many companies out there, these KPIs are produced by various excel spreadsheets that are based on guesstimates and outdated numbers and nobody really trusts said data. Things get worse when two or more departments have to make decisions and use different versions of spreadsheets to derive their actions.

Thus, in order to enable better decision making processes at our customers, we believe that centralizing the definition, computation and storage of said KPIs is a high value opportunity.

The Assignment

As part of the ENLYZE product team, you have identified this opportunity. Through many user interviews, you and your team have learned that for every customer, the number, shape and form of relevant KPIs looks different.

After consolidating all your findings, you decide it's time to build a proof of concept. You have nailed down the following scope:

You want to build a prototype of the KPI builder that allows you to list, create and edit KPIs

You jot down some details:

- A KPI has a name and is defined by two distinct steps of computation:
  1. **Conditioning:** Computation of a result time series composing one or more time series variables and constants together by means of arithmetic operations. Example: `((v1 + v2) - (v4 - v3)/3`
  2. **Aggregation:** Aggregating the result of the conditioning step. For our PoC, we have decided to calculate KPIs for every production run (think time frame a certain product has been produced). Because this is determined by a different system, we do not have to worry about time frame input and validation.
     Our use research has shown that we need the following aggregations:
     a. median
     b. average
     c. integration.
     d. sum
- You conclude that you will use a fixed set of variables that are defined by a UUID and a human friendly display name:
  `const variables = [{uuid: "my-uuid", displayName: "My Display Name"}, ..]`

Some guidelines

The following points are supposed to act as guard rails so that you don't loose yourself. In case anything is unclear, feel free to reach out whenever you want.

- The user interface conception is completely up to you. Focus on the Create / Update flow, the list view can be completely scrappy
- You can use any library you want
- We don't expect you to build the execution engine, just enough backend to handle list, create and edit
- We don't expect you to use a database, storing the KPI definitnions in memory or a file is completely fine
- In case you're feeling it, you add your own ideas to the mix