Software Challenge

In this challenge you can prove your expertise as a software engineer and convince us that you are ready to join our Team at PlanerAI!

Description

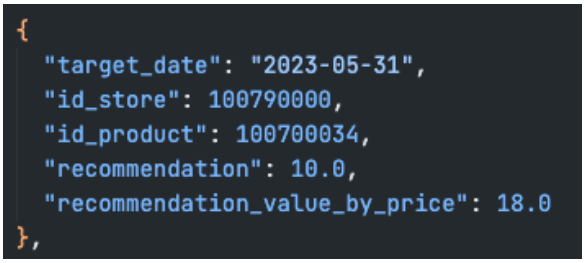
Anna, one of our colleagues at the customer solutions department has noted that her newest bakery customer still lacks trust into our AI generated order recommendations and therefore regularly adjusts and manually overwrites these orders. Anna is convinced that these manual adjustments often result in even higher leftovers at the end of the day and asks you to help her visualize the impact of these overwrites.

Her idea is to create a **reporting view** with which she can

1. analyze how often past recommendation and delivery quantities **differ** (which means the customer has adjusted the order in the end) and
2. see whether these **adjustments improved** (the actual delivery quantity is closer to the demand than our order recommendation) **or deteriorated the final orders** (our order recommendation would have been closer to actual demand).

Since the BäckerAI solution is used for planning orders in multiple stores and for multiple products, she asks you to build this reporting view that **can filter for specific store and product combinations** and **visualizes the information in at least one graph**.

To develop a first prototype, one of our colleagues from the backend team has provided you with a static data set that comprises the required customer information over the past two weeks[1].



Recommendations

Deliveries

The data set consists of 5 files that you can find in JSON format in the appendix. The base data is in the following files:

- Products.json
- Stores.json

The required transaction data for the analysis are split into three files and can be joined via the keys id_product, id_store and date:

- Sales.json
- Recommendations.json
- Deliveries.json

[1] The entire dataset is in the appendix as JSON files

Requirements

Technical:

- Programming language: JavaScript or TypeScript
- Framework: VueJS (or any other comparable framework that you have experience with)

Functional:

- Reporting view:
  - Visualize the differences between AI-based ordering recommendation and actual delivery quantity.
  - Mark where original order recommendations were improved or deteriorated.
- Filter visualized products and stores.
- Include at least one graph.

Non-functional:

- Create an appealing interface. Use the tools of your choice.
- Use tools that are common for software development, especially for larger projects.
- Think of how you can guarantee the correctness of the displayed information.

Additional information

As a software engineer at PlanerAI, you will be involved in the whole development process from prototyping over testing to deployment. For this reason we have formulated this challenge to cover a wide range of these facets. However, it should be completed in about 3 hours. It would be great if you could let us know how much time you have spent.

The layout of the application should work on the desktop, optimizing for other resolutions is not necessary.

The challenge is intentionally formulated openly to give you as much freedom as possible for solving the problem. If you have something in mind that you think is important but too extensive for this challenge, feel free to write it down for us in comments, a readme or similar.

Bonus

If you want to invest more time to solve our challenge, you are more than welcome to implement further functionality for the reporting view. Next, we have some further ideas:

- Anna would potentially review the report once a week and therefore would need to filter the dataset by calendar week.
- To further improve user experience in the reporting view you could implement additional click or hover actions in the graph.