Orbidi – Data Engineer Challenge

Context

First of all, thank you for taking the time to apply to **Orbidi** and for being willing to complete this technical take-home challenge. We are well aware that your time is precious and are grateful that you are willing to take the time to demonstrate your skills via this challenge.

The challenge is made up of **two parts**:

1. A **design challenge**
2. A **coding challenge**

Once you submit your solution, we will review it and then schedule a meeting for you to walk us through your work and explain the process that led you to your design and code solution.

Generic Guidelines

There is no single *perfect* solution for these challenges – just as with the real projects you would work on at Orbidi. Problems can be tackled in many different ways, and we are just as interested in your **thought process** as in the **final results**:

> "The journey matters more than the destination."

Some guidelines we highly recommend you follow:

- **Documentation is as important as the code**

  We should be able to understand everything you did by reading:

  - The `README`
  - Docstrings in your code
  - Any short tutorials or walkthroughs (e.g. Loom videos)

- **Do not reinvent the wheel**

  Savvy developers excel at using existing tools and patterns. Often:

  - A library or example similar to what you are trying to do already exists.

    - Your job is to assemble the right “Lego blocks” and make them work together seamlessly. We therefore encourage you to search the web and take inspiration from similar problems.

- **Automation is key**

  We are all about automation. While the challenges do not enforce specific DevOps/DataOps constraints, we highly encourage you:

    - To **automate as much as you can** through GitOps practices and CI/CD pipelines.
    - To **deploy cloud resources via IaC** (Infrastructure as Code), e.g. Terraform.

Submission

Once you are done with the challenge, please follow these steps:

1. For the **design challenge**, save your deliverables as a **PDF**.
2. For the **coding challenge**:
    - Put your code in a **public GitHub repository**.
    - Include the **link to your Looker Studio (Data Studio) dashboard** in the `README.md`.
    - Share the dashboard with
      - `alejandro@astrafy.io`
      - felipe.bereilh@orbidi.com
    -
3. Reply to the email you received with this “Take-home Challenge”:
    - Attach the **PDF** from step 1.
    - Include the **link to the GitHub repository** (step 2).

Part 1: Design Challenge

a. Task

There is a conversation in the office between Orbidi colleagues about a new customer project.

The customer has many different sources of data, ranging from:

- Transactional databases: **PostgreSQL**, **MySQL**, **MongoDB**

- Well-known applications: **SAP**, **Salesforce**, **SurveyMonkey**

The customer wants to **fully revamp** its current analytical approach and has asked the data department at Orbidi to design a potential solution. The end goal is:

- To serve **BI dashboards** for various departments.
- To develop **ML models** for recommendations and predictions.

The customer has the following requirements:

- **Google Cloud** as the hosting platform
- **Open-source technologies** as much as possible
- High focus on **GitOps** and **DataOps**
- **Data mesh** paradigm:
  - Current data domains: `customers`, `products`, and `maisons`
  - The design should be **extensible** to add additional data products in the future.
- **Federated data governance** for:
  - Accessing data
  - Data observability
  - Data cataloguing

b. Deliverables

- An **architecture diagram** for the solution in **PDF** format. You can use Lucidchart's free tier or any other diagramming tool of your choice.

  No code is expected for this task.

Part 2: Coding Challenge

a. Introduction

Orbidi is helping the **city of Chicago** with some of its analytics.

One of the many datasets of interest is the **"Chicago Taxi Trips"** dataset, which is **publicly available on BigQuery**.

https://console.cloud.google.com/marketplace/details/city-of-chicago-public-data/chicago-taxi-trips

The mayor of Chicago suspects that **weather conditions affect the duration of trips**. Because of this, he wants us to build a small **Looker Studio dashboard** to explore and visualize whether weather conditions affect trip duration.

b. Deliverables

- A **GitHub repository** with all relevant code (or any other publicly accessible repository registry).
- A **Looker Studio dashboard** showing the analysis and insights.

c. Instructions

- The taxi trip data of interest must be **filtered on** `trip_start_timestamp`:
  - From **01/06/2023** (June 1, 2023)
  - Until **31/12/2023** (December 31, 2023)
- **Weather data ingestion**:
  - New weather data for the **day before** must be ingested **every day at a specific time**.
  - The pipeline must:
    - Ingest **all historical data** on the first run.
    - Then run **daily** to ingest the **previous day's data**, even if this new data will not be used in the dashboard (since the trip data ends in 2023).
- **Infrastructure requirements**:
  - All infrastructure must be hosted in **Google Cloud** (except anything that can be stored in the GitHub repository, such as CI/CD pipelines).
  - **Do not create any resources manually**:
    - All resources must be created via **Terraform**.
  - Use **dbt** to perform the data transformations.
- **Optional requirement**:
  - Only **your email** should have access to the column `payment_type`. (You may handle this via column-level security, views, or similar approaches.)

Important:

By selecting only 6 months of data, your queries will remain within the **free tier** offered by Google Cloud for any new projects, so you should not incur query costs.