Technical Test - Data Engineer

If you have any questions or need clarification regarding this case study, do not hesitate to contact us by email.
alexis.simon@meero.com

Context:

You work for a platform that offers concierge services to property owners on Airbnb. The goal is to create a data pipeline to retrieve information from Airbnb listings and cross-reference it with data stored on MinIO, which contains information about our clients: the concierge companies. Your objective is to build an automated pipeline allowing us to retrieve the data from the different sources, consolidate it into a single database and join this information to build a first data mart allowing us to monitor the number of customers relative to the number of listings by city in France.

1) Extraction of Airbnb Data

Retrieve data from Airbnb listings using the Opendatasoft API.
(https://public.opendatasoft.com/explore/dataset/air-bnb-listings/api)

2) Extraction of Customers Data from MinIO

Retrieve customer data by accessing MinIO, which is a storage solution compatible with the various AWS S3 SDKs. The idea is to consider that these files would be updated daily by our back office.

To connect to MinIO, it is necessary to reuse the environment variables defined in the `docker-compose.yaml`, namely:

MINIO_ROOT_USER: minio
MINIO_ROOT_PASSWORD:password

By default (see docker-compose.yaml), the MinIO UI is accessible on port 9001 while port 9000 must be used to carry out operations.

3) Transformation and Crossing of Data

Identify the join key between Airbnb listings data and customer data.

Merge the two datasets using this key.

Apply transformations that you find relevant to the use case.

4) Storage of Results

Store the resulting enriched data in a specified location, ready for use by other applications or services.
Note:

The test aims to assess your skills in designing and implementing data pipelines, as well as your ability to solve real-world problems encountered in the field of data engineering.

Ideally, use Python/SQL to perform this test.

The code should be readable, ready for production, and functional regardless of the date or the number of times it is run.

The code should be documented/organized with installation instructions.

A GitHub repo would be appreciated.