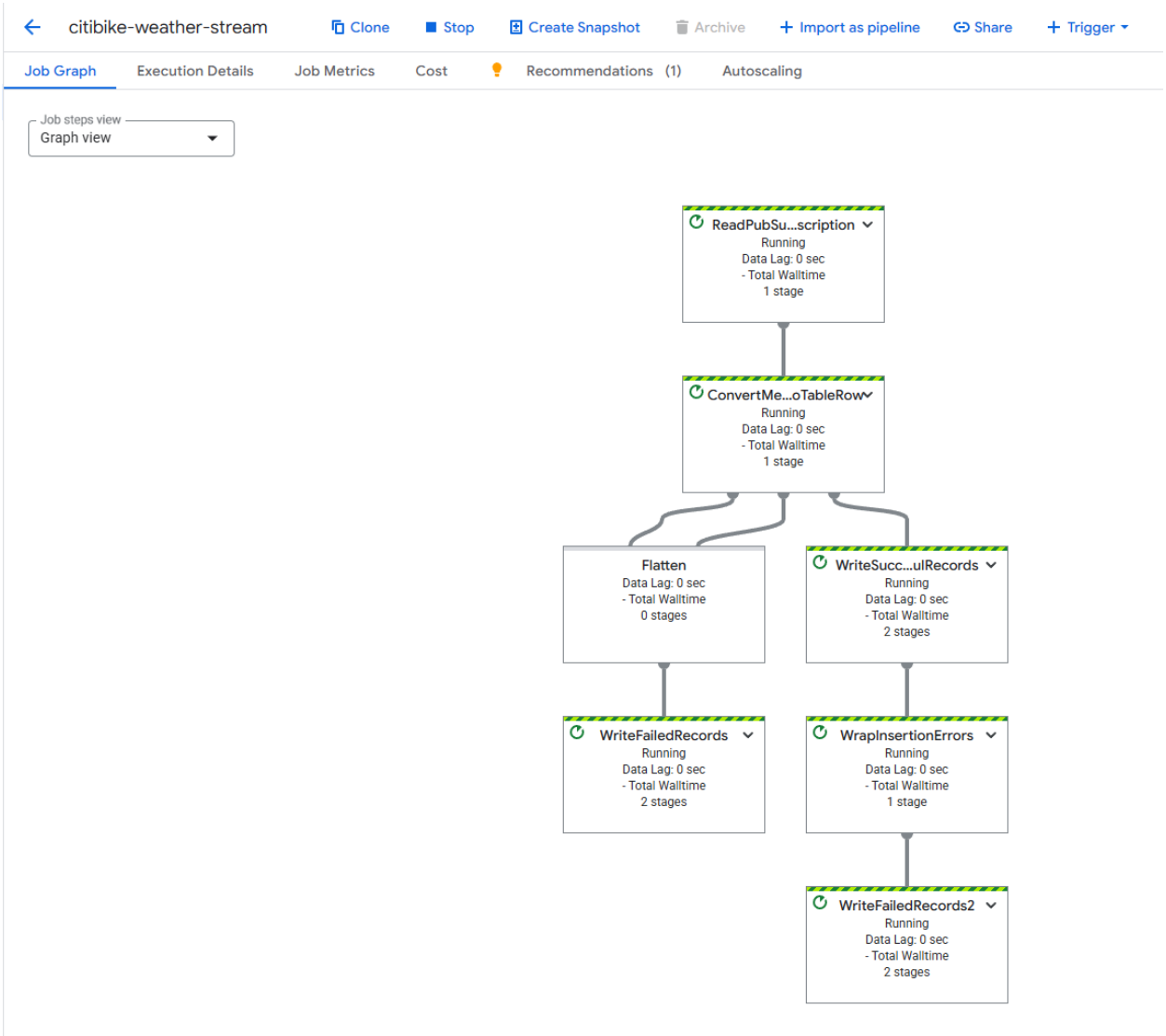
Our goal with our Dataflow design was to allow continuous streaming of 7-day weather forecasts from Open-Meteo into BigQuery so that we can create real-time NYC CitiBike trip predictions and feed them into Looker Studio to visualize.



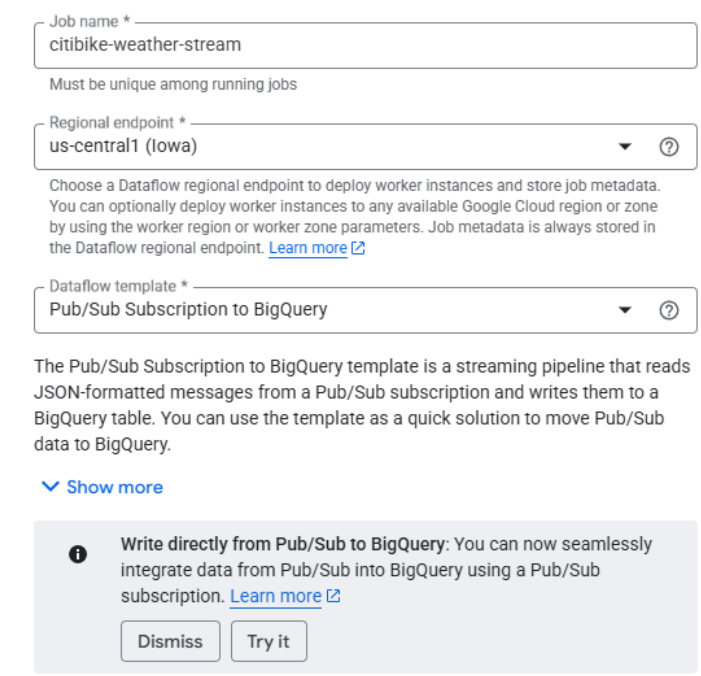
We do this by utilizing four components:

1. **Cloud Run service (open-meteo):** Every minute, Cloud Scheduler calls our Cloud Run service to call the Open-Meteo API which then publishes a JSON message per forecast day to our Pub/Sub topic “citibike-weather-topic.” While calling the API every 60 isn’t absolutely necessary in our application due to weather forecasting not

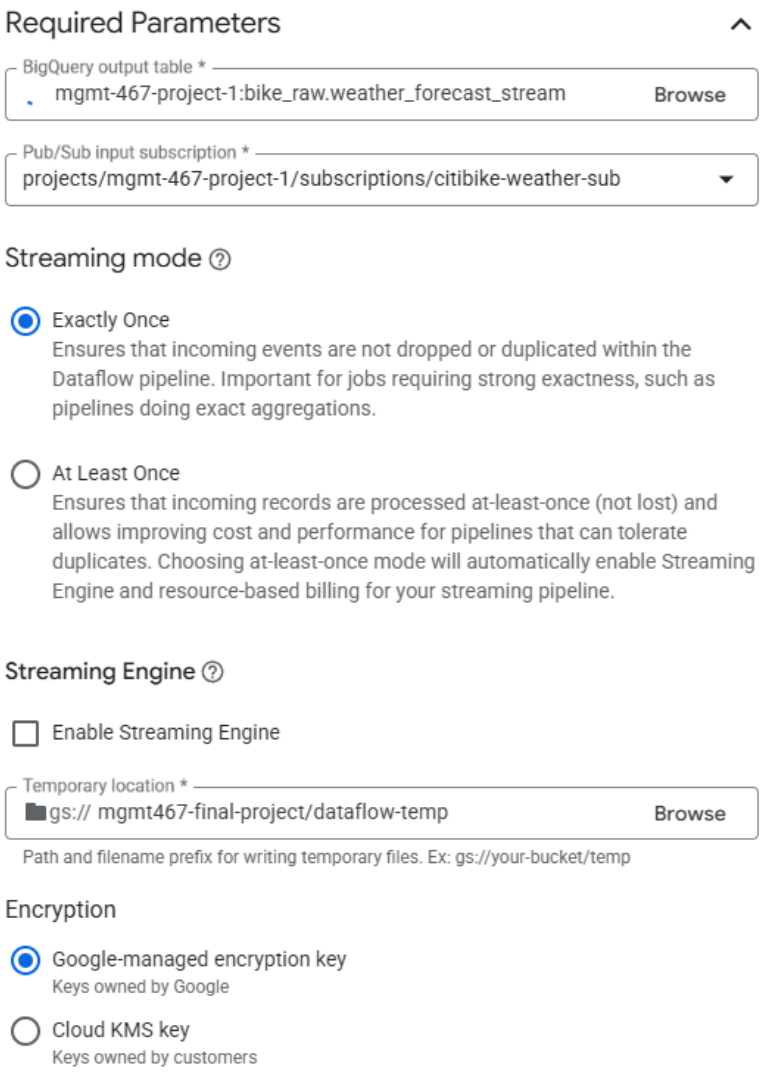
being highly volatile, we still fall well within the allowed requests/day and wanted to try working with a high frequency of calls so we did it anyways.

2. **Pub/Sub subscription (citibike-weather-sub):** This is a pull subscription that acts as the input queue for Dataflow by receiving the messages from the Cloud Run.
3. **Dataflow streaming job:** We use the "Pub/Sub Subscription to BigQuery" template to read the JSON messages from the subscription, then insert them into our BigQuery raw streaming table "weather_forecast_stream."
4. **Downstream BigQuery SQL:** We create a curated features table called "weather_forecast_features" that holds our desired forecast columns and engineered features "dow", "month", and "is_weekend." We then feed it into our prediction models where one predicts the number of member trips while the other predicts the number of casual trips, then sums them for total predicted trips. Those are fed into the prediction table "trips_forecast" which then are fed into Looker Studio where our dashboards display the trends.

Dataflow Template Parameters



Our job is named "citibike-weather-stream" and our chosen template is the "Pub/Sub Subscription to BigQuery" so that we can enable real-time streaming.



Our selected output table is “weather_forecast_stream” which is a table we use to store our raw streaming forecast data from the API. Our downstream curated tables and ML models utilize features and tables derived from this initial table. Our input subscription is our “citibike-weather-sub” which is how Dataflow pulls the data in. The subscription is connected to our topic where our Cloud Run service publishes the Open-Meteo messages to. We selected exactly once for our streaming mode to ensure that there are no dropped or duped records because aggregation is important for tracking our KPI’s and demand forecasting, so having duplicates would cause significantly poor reporting and insights. We use dataflow-temp to store temporary files so that we can keep them separated from our CitiBike and weather data folders to ensure that nothing interferes with the pipeline or causes issues with our dashboards. Finally, we used Google-managed encryption because it’s the default option and satisfactory for academic related work.