Group No: 73

Hardik Sharma(G23AI1013)

MD Zaigam Ali Shah(G23AI1068)

Readme File

This document covers the following points:

1. **Introduction**: A summary of the Sales Data Pipeline project.
2. **Setup Instructions**: Steps to clone the repository, install dependencies, set up GCP credentials, and run the Flask app and Cloud Function.
3. **How to Upload Data**: Instructions on how to use the Flask portal to upload sales data, and how it's processed.

Sales Data Pipeline using GCP

Introduction:

The Sales Data Pipeline project is designed to ingest and process sales data from multiple sources. The data is uploaded through a Flask web portal, stored in Google Cloud Storage (GCS), and automatically pushed to BigQuery. After loading the data, users can perform analysis through BigQuery and connect the data with Power BI for visualization and automatic report scheduling.

Setup Instructions:

Prerequisites:

- Google Cloud account with BigQuery and GCS services enabled.
- Python 3.x installed.
- Flask installed (pip install flask).
- Google Cloud SDK installed and authenticated (gcloud auth login).

Steps:

1.Clone the Repository:

```
bash
Copy code
git clone < https://github.com/hardik6212/sales-data-pipeline-using-
gcp >
cd Sales-Data-Pipeline
```

2.Install Dependencies:

```
pip install -r requirements.txt
Pip install flask
Pip install google cloud storage
```

3.Set Up Google Cloud SDK:

Authenticate using Google Cloud SDK:

gcloud auth login
gcloud config set project < salesdatapipeline-437107 >

4.Set application credentials:

```
gcloud auth application-default login
```

5.Run the Flask Application:

```
python main.py
```

Access the portal at http://127.0.0.1:5000/ to upload data files.

Deploy Google Cloud Functions: Deploy the Cloud Function to automate the loading of files from GCS to BigQuery.

How to Upload Data:

1. Access the Flask web portal to upload sales data files (CSV or other formats) to your GCS bucket.
2. Once uploaded, a Cloud Function is triggered that processes the file and loads it into the BigQuery dataset and table.
3. You can monitor the data in BigQuery by running queries or creating views for analysis.