

OLA ETL DATA PIPELINE PROJECT DOCUMENTATION

Project Overview-

Welcome to my project.

As we all know that Ola is an important application that takes care of us in every day-to-day activity.

Whether to go to work or roaming around the city with our family and friends, Ola is a must have thing inside our phone.

Today we will go through each process of the Ola rides based on a dataset to build a pipeline to find out all those meaningful information like how much travel people does and how much cost is there to find out what is there for benefit of the company in terms of revenue and cost management.

We can also be able to see the travel destination across the maps in a global based view as well.

Sounds intriguing right? Let's jump right into our project.

I've tried to explain each and every single parts of developing the project from scratch with a clear view by which a person with a very little or no knowledge of the whole data engineering field can also be able to understand what's going on throughout this project.

Technologies used-

1. MAGE AI (For building the pipelines)
2. Google Cloud(Compute VM engine, Cloud Storage, BigQuery)(For Computing, storing and Querying our datas)
3. Google Looker Studio(For making an impactful Dashboard)

If you don't know any of these technologies, no worries. I'll mention each and every technologies in details in the further project development stages for you to understand.

There is only one prerequisite with this project and that is you have to make a Google Cloud account. It has a trial period of 3 months and it will give 300\$ worth of free credits after creating

the account which is more than sufficient to fully utilize all the cloud resources within this trial period. Do register with a credit or debit card(It will not cost any money.It will take 2rs from your account to verify and will return the money to your account immediately after verification).

Let's jump into the project and see the architecture.

Project Architecture-



Fig1- Project Architecture

Detailed Stages for the project-

The Above figure is the project Architecture. Let me explain how the whole project works.

Firstly, for the dataset, I've downloaded the dataset from **Kaggle**, A free opensource website known for providing a wide range of datasets on every single possible fields in the technical field.

The dataset contains coulumns like "fare_amount","date_time", "longitude-Latitude" etc and it is in a csv format.

The Idea behind the whole project is to store this raw data into the cloud storage, then as a computing system, used the VM engine of google cloud and to run the MAGE AI into it to Load the data, Transform the data and feedback the data into the table of BigQuery for Querying.

Let's breakdown the stages-

Step 1-Storing Raw Data-

- i. After creating the google cloud, go to the upper side of Console and you will find a place to save your project name. Give a name to your project.
- ii. Search for “Cloud Storage” and open it.
- iii. Click **+ Create**.
- iv. On the **Create a bucket** page, enter your bucket information. To go to the next step, click **Continue**.
 - a. For **Name your bucket**, enter a name that meets the [bucket name requirements](#).
 - b. For **Choose where to store your data**, select a [Location type](#) and [Location](#) where the bucket data will be permanently stored(Use Multi-region mode for high availability of your data) .
 - c. For **Choose a storage class for your data**, either select a [default storage class](#) for the bucket, or select [Autoclass](#) for automatic storage class management of your bucket's data.

Note: The **Monthly cost estimate** panel in the right pane estimates the bucket's monthly costs based on your selected storage class and location, as well as your expected data size and operations.
 - d. For **Choose how to control access to objects**, select whether or not your bucket enforces [public access prevention](#), and select an [Access control model](#) for your bucket's objects(Use Fine-grained for this project purpose only).
 - e. For **Choose how to protect object data**, configure **Protection tools** if desired, and select a [Data encryption method](#).
- v. Click **Create** and your bucket will be ready to store the dataset.
- vi. After Few seconds, your bucket will appear in that window.
- vii. Click On the bucket, then go to “Upload Files” and upload your dataset.
- viii. For this project only, we will keep this dataset as public to be able to access the data freely.
- ix. Go the three-dot button at the right side of your data, then click on “Edit Access” and then from the “Entity 1” field, choose “Public” as a mode then Click onto “Save” to store the changes.
- x. After that you will be able to see the “Public to Internet” URL. This will be used to access your data from all over the Internet.

Step 2- Creating And installing Libraries into a Compute Engine-

We can do our work on one local machine also for executing the project but for learnings and future scopes, let's work together onto a Compute Engine.

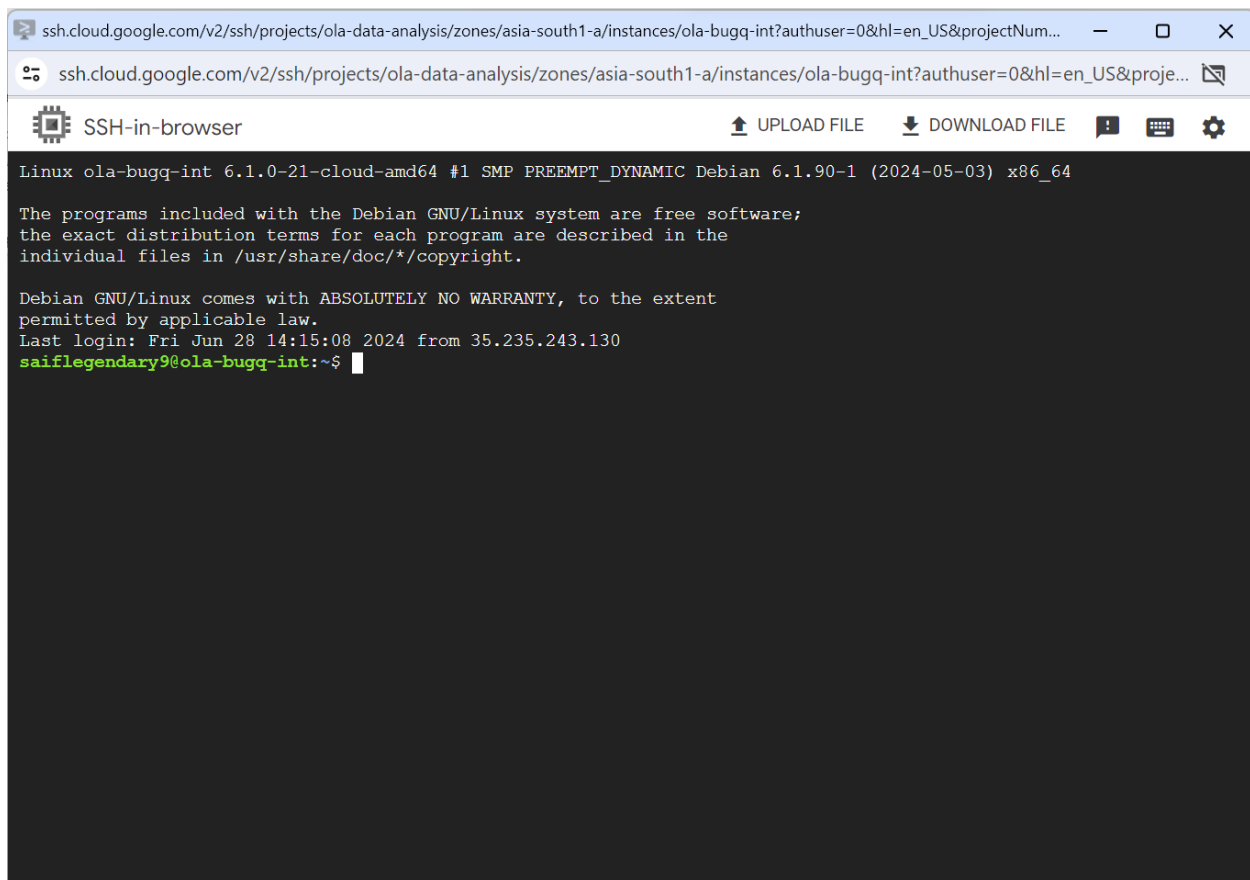
Google Compute Engine (GCE) is **an infrastructure as a service (IaaS) offering that allows clients to run workloads on Google's physical hardware.**

That means we can use it as a computer for all of our project workloads.

Let's walk through each step.

- I. In the Google Cloud console, go to the **VM instances** page.
- II. Select your project and click **Continue**.
 1. Click **Create instance**.
 2. Specify a **Name** for your VM. For more information, see [Resource naming convention](#).
 3. Optional: Change the **Zone** for this VM. Compute Engine randomizes the list of zones within each region to encourage use across multiple zones (Use asia-south1 (Mumbai) as it is our nearest region)
 4. Select a **Machine configuration** for your VM (I've used **E2** as it is **sufficient for this project and for "Machine Type" I've used Standard "e2-standard-4" for a quick response time**).
 5. In the **Boot disk** section, click **Change**, and then do the following:
 - a. On the **Public images** tab, choose the following:
 - Operating system (I've used **Debian GNU/Linux 12 (bookworm)**)
 - OS version
 - Boot disk type
 - Boot disk size
 - b. Optional: For advanced configuration options, click **Show advanced configuration**.
 - c. To confirm your boot disk options, click **Select** (I've used the **default selection**)
 6. In the "Access Scopes" option, click "Allow full access to all Cloud APIs" as we need BigQuery and Cloud access in the compute engine.

7. In the **Firewall** section, to permit HTTP or HTTPS traffic to the VM, select **Allow HTTP traffic** or **Allow HTTPS traffic**. When you select one of these, Compute Engine adds a network tag to your VM, which associates the firewall rule with the VM. Then, Compute Engine creates the corresponding ingress [firewall rule](#) that allows all incoming traffic on tcp:80 (HTTP) or tcp:443 (HTTPS).
 - To turn on [Secure Boot](#), select **Turn on Secure Boot**. Secure Boot is [disabled by default](#).
- III. Keep the rest selections as it is as they are optional changes and then click **Create** to create and start the VM.
- IV. After creating, click on the “SSH” button to run the Linux Machine (Click on Authorize to give access to the machine to run within your cloud environment).



The screenshot shows a web browser window with the address bar displaying the URL: `ssh.cloud.google.com/v2/ssh/projects/ola-data-analysis/zones/asia-south1-a/instances/ola-bugq-int?authuser=0&hl=en_US&projectNum...`. The browser tab is titled "SSH-in-browser". Below the browser window, there is a terminal window with a dark background. The terminal output shows the following text:

```
Linux ola-bugq-int 6.1.0-21-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.90-1 (2024-05-03) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jun 28 14:15:08 2024 from 35.235.243.130
saiflegendary9@ola-bugq-int:~$
```

Fig2- SSH screen

- V. Now we need to run all the commands to install our desired packages into the Linux machine. I have provided a "Commands.txt" file that you need to execute step by step and it is necessary to use them according to the step numbers as every step is crucial, otherwise the environment will not work properly, and you may get errors.

Vi. Let's explain the Commands-

1. `sudo apt-get update`- It downloads the package lists from the repositories and "updates" them to get information on the newest versions of packages and their dependencies.
2. `sudo apt-get install python3-distutils`- "distutils" is the primary way of building and distributing Python packages.
3. `sudo apt-get install python3-apt`- A way to install python(Version 3) into the system.
4. Wget <https://bootstrap.pypa.io/get-pip.py>- A resource to download the "pip" which is a package manager to install all python packages.
5. `python3 -m venv env`- After the above-mentioned steps, the best practice is to use this command to create a separate environment. Venv is a tool that creates isolated Python environments. These isolated environments can have separate versions of Python packages, which allows you to isolate one project's dependencies from the dependencies of other projects.
6. `source env/bin/activate`- To activate The Environment.
7. `python3 get-pip.py`- Now we install the package manager.
8. `pip3 install mage-ai`- Mage Ai is an open-source pipeline tool that basically helps us to build sustainable pipelines for our work. It is an alternative to "Apache -Airflow" and have a very user-friendly environment and is also a powerful AI managed source.
9. `pip3 install pandas`- Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.
10. `pip3 install google-cloud`- To install "Google cloud" environment into the system.
11. `pip3 install google-cloud-bigquery`- To install "Google Bigquery" environment into the system.

12. pip3 install pandas-gbq- "Pandas-gbq" is library that is used to send pandas dataframe datas directly to Google BigQuery dataset table.

Step 3- The ETL process

1. The whole installing process is done and we can now connect to the MAGE-AI tool using SSH.
2. To run a project (i.e. project name-"ola-pipeline") run the command-" mage start "project name" and hit enter.
3. It will then load the mage ai and you will see the port number (by default the port no is 6789).
4. Simply go to your compute engine, copy the external ip and after that put ":" and the put the port number where mage ai is running (i.e. let's say your external ip is 34.47.131.129 and your port number is 6789 then go to the search bar and type 34.47.131.129:6789 and hit enter.
5. If you done your firewall connections (set up the firewall by going into the network interface button and create a separate firewall rule for your mage ai operations. Put ip ranges as 0.0.0.0/0 as it can be accessed from everywhere and put the port number as 6789 and save the rule, restart your compute engine and you are good to go) you will see the MAGE-AI user interface.
6. Click on to create a pipeline and create three pipelines modules to extract the data from cloud storage (I used an API function), transform the data to some particular extent and load back the data into BigQuery.
7. I've given the codes of those three pipelines with this project that I developed into mage-ai. You can go through them to understand each level quite clearly.

Step 4- Loading the data into BigQuery.

1. The transformation part is done, and we are now going to load the values from mage-ai to BigQuery.
2. But before that, the GCP will need to give the mage-ai some access permissions by which the external tool can access the BigQuery dataset.
3. Go to “Services” and create a service Account for the mage-ai connection.
4. Give a name and select “create and continue.”
5. Now you need to give the service account the specific permission it needs. For our project, we need “Big Query admin” access to access the BigQuery and “Big Query Data editor” to be able to access the datas and modify it within Big Query.
6. After that select done and your service account should be ready.
7. Go to the service account dashboard and select the account you just created.
8. Go to keys and create one and save it as .JSON file. It will be downloaded into your system.
9. Go to BigQuery and create a Dataset now and after creating it, copy the dataset id which will be needed into the data exporter function.
10. Now create the Exporter function and you can see the “io_config.yaml” file in the catalog menu on your left side.
11. Put the necessary key under the GOOGLE_SERVICE_ACC_KEY given like the below figure.


```

DRUID_PASSWORD: password
DRUID_PATH: /druid/v2/sql/
DRUID_PORT: 8082
DRUID_SCHEME: http
DRUID_USER: user
# Google
GOOGLE_SERVICE_ACC_KEY:
  type: service_account
  project_id: project-id
  private_key_id: key-id
  private_key: "-----BEGIN PRIVATE KEY-----\nyour_private_key
  client_email: your_service_account_email
  auth_uri: "https://accounts.google.com/o/oauth2/auth"
  token_uri: "https://accounts.google.com/o/oauth2/token"
  auth_provider_x509_cert_url: "https://www.googleapis.com/oauth2/v1/certs"
  client_x509_cert_url: "https://www.googleapis.com/robot/v1/metadata/x509/your_service_account_email@project-id.iam.gserviceaccount.com"
GOOGLE_SERVICE_ACC_KEY_FILEPATH: "/path/to/your/service_account_key.json"
GOOGLE_LOCATION: US # Optional

```

Fig3- io_config.yaml file

12. After putting the datas into the .yaml file. Hit run and if you have done the previous processes like I mentioned, you can see the datas successfully reflected into the BigQuery table.

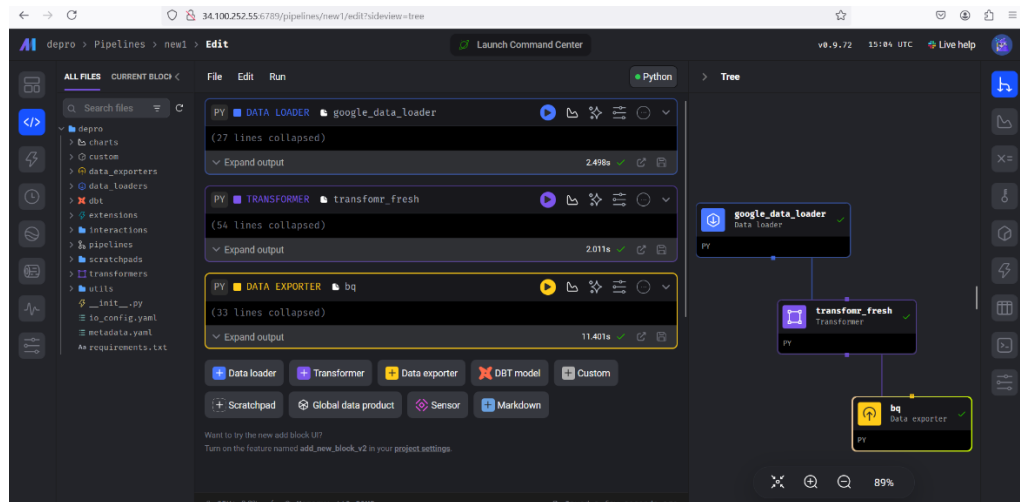


Fig4-The Pipelines

Step5- Querying and presenting the data into looker studio

1. After getting the data into BigQuery, you can query the table according to your need.

2. After that go to looker studio and authenticate Google BigQuery and it will automatically load all the datas that you have from the BigQuery.

3. Design according to what you want based on your datas.

I'm giving out my Looker Studio link of this project-

"<https://lookerstudio.google.com/reporting/b49ebd3a-0950-4355-b10e-715051eabebe>"

PROBLEMS I FACED WHILE BUILDING THE PROJECT-

1. **The first problem that occurred to me while installing the python libraries. I was trying to install the libraries without making a virtual environment, but the engine didn't allow me to do so as it is accepting the engine as an external source.**
2. **Second thing that I faced which was the major issue was the connection between MAGE_AI and GCP. We need to give the all api access to the VM engine that we are using also to be able to access the gcp from mage-ai.**

These are the two problem that I faced mainly while building the project.

Hope you like the project. Share with me if any updates can be needed right here-

<https://www.linkedin.com/in/mohammedsaif09/>

Thank you for giving your time. Have a wonderful day.

