# Code Assessment Document (v2.0)

# By

# Chitwan Humad

**Email: chitwanhumad@gmail.com**

## Document Revision History

| version | Created by | Reviewed by | Summary | Published Date |
|---|---|---|---|---|
| 1.0 | Chitwan Humad | **Sahil Satpute** | Initial assessment understanding document | 18-08-2025 |
| 2.0 | Chitwan Humad | **Sahil Satpute** | Assignment completion document | 20-08-2025 |

# Contents


Assessment Details ..... 5

Problem Statement: ..... 5

Tech Stack: ..... 5

Delivery: ..... 5

ETA: ..... 6

Assessment – In scope: ..... 6

Assessment – Out of scope: ..... 6

Assessment – Use Case Success Criteria: ..... 6

Assessment – Tech Stack Selection: Tentative ..... 7

Assessment – Deliverables: ..... 8

Assessment – Completion Date: Tentative ..... 8

Solution High Level ..... 8

Prerequisites ..... 9

Sync Git Repository and Installation Steps ..... 9

Set up local Prefect Environment ..... 9

Start Prefect Server ..... 11

Set up and Run Data pipeline ..... 11

Set up Local Superset Environment ..... 12

Start Superset Server: ..... 14

Create Database Connection ..... 15

Import Dashboards ..... 15

Execute Dashboard ..... 16

Optional Steps ..... 17

Deploy Pipeline ..... 17

Schedule Pipeline ..... 17

Troubleshoot ..... 17

Disclaimer ..... 17

# Assessment Details

## Problem Statement:

- Create an ELT pipeline that ingests a CSV dataset (choose any sufficiently dense source eg. https://www.kaggle.com/datasets/abdullah0a/telecom-customer-churn-insights-for-analysis).
- Load up the dataset into a staging database of your choice.
- Design a transformation layer to process the input dataset for missing values (use defaults) and anonymising PII.
- The destination for the processed data should be a database ideal for generating reports.
- Establish an orchestration workflow for this pipeline to accept a feed every hour (should be configurable).
- Integrate any open-source reporting tool to generate statistics about the flow.
- Ensure the entire setup is available through composable container definition(s).

## Tech Stack:

- Language/frameworks/solutions of your choice. Please just ensure, the solution is easy to run on a laptop.
- Please use open-source solutions wherever possible.

## Delivery:

- Please share the entire source code as a public Github repository.
- Do add relevant instructions to run the code.
- Please also ensure it stays accessible for the duration of the discussions with HGI.

## ETA:

- Please ensure the assignment is completed in about 16-20 hours (can be split over days if practical schedules demand).

## Assessment – In scope:

- Bronze Layer (Data Lake): Read and writing csv data via a pipeline to store in the database table – csv data will be pulled up from Kaggle
- Silver Layer (Transformed Data layer): Data transformation and stored the refined data into database tables – up to 3 use cases
- Gold Layer (Reporting Layer): Pre-aggregated data for reporting purposes – up to 3 use cases

## Assessment – Out of scope:

- Containerization of the solution

## Assessment – Use Case Success Criteria:

The solution should be considered as successful if the following use cases are achieve during the user acceptance testing:

1. Tech stack selection: should be Open source as far as possible
2. Each run should have internal runid to track pipeline runs
3. CSV file(s) should be able read from <>/source/<name>.csv folder and load into the database without any change in the data in the raw_customer table of bronze_db database
4. Solution to enable hourly to ingest a new file hourly
5. The processes file should be moved/archived in the processed file into
   <>/archive/runid_<>_ allfiles_gooddata_datetimeid.csv
   <>/archive/runid_<>_ allfiles_baddata_datetimeid.csv
   Presently bad data is computed based on non-integer customerids only
6. Solution should follow 3 use cases to conduct data transformations:

a. Check for NaN or missing values for a few fields (field names – TBD)
b. Check valid values for Age – should be a positive integer only
c. Check for valid values from the data dictionary for ContractType field as Month-to-Month, One-Year, Two-Year

7. Bad data rows based on the above should be saved into the <>/baddata/<name>_runid _datetimeid _done.csv
8. Read bronze_db.raw_customer table data and perform following transformation to make presentable reports:
    a. Transform InternetService missing values to None
    b. Round off TotalCharges values to 2 decimals
    c. Define new dimension as Tenure_Range for each 10 blocks, e.g. 1-10, 11-20 so on
    d. Define Age_band dimension 20-25, 36-30 so on every 5 years
    e. Drop Age field to preserve PII information
    f. Define new dimension Category High/Medium/Low for MonthlyCharges < 50 Low, 51-100 medium and > 100 high
9. The transformed data should be stored into silver_db.customer table
10. Produce a aggregated data models to generate various reports in the gold_db.<table_names>, like:
    a. Count of customers by Categories (i.e. High/Medium/Low)
    b. Aggregated revenue (TotalCharges) by Contract Types
    c. Aggregated revenue (TotalCharges) by InternetService
    d. Customer demographic Presentation who availed technical support facility by Age_band and gender
11. A run and log table to record runs

## Assessment – Tech Stack Selection: Tentative

- OS – Windows laptop
- Prefect for data pipeline – Open source

- Superset – Open source
- Database – Sqlserver Express using sa credentials

## Assessment – Deliverables:

- Git repo url https://github.com/chitwanhumad/hg_datapipeline

(Kindly confirm you can access the url)

## Assessment – Completion Date: Tentative

22-Aug-2025

---

## Solution High Level

1. SCD 2 Implementation – maintained all history however report shows all latest data for each customer. Example –

| Runid | New CustomerID in the input file | Updated CustomerID in the input file | Report Data |
|---|---|---|---|
| 1 | 1- 100 | NA | All 1 - 100 |
| 2 | 101 - 120 | 5, 50 | All 1 – 120** |
| 3 | NA | 61, 71 | All 1 – 120** |

** updated records data with the latest rows

2. No archival of old data has been provisioned however it has to be there. Suggested solution could be, based on the business requirements last N days data should be kept into silver_db.customers tables as per business policy. The system performance will degrade without data archival policies.
3. Only one condition of Bad data has been assumed for now. It is for non int customerID.

4. It is assumed that there could be more than one customer files may be loaded at a time. All good data and/or bad data will be saved in the /archive/ folder with the runid in the file.
5. Runid is to track every run. The same runid will be used to read logs from the dbo.acr_log table.
6. Runid will also be used for data lineage purpose.
7. Bronze_db will have all data, each row will have a runid and inserttime associated with it.
8. Siliver db will have soft delete of the older rows, reference column is is_current = 'Y'
9. Gold_db will have up-to-date aggregated data only. Users can refer lastrefreshtime field for their reference.

## Prerequisites

1. Make sure you have python 3.10 environment on your windows

## Sync Git Repository and Installation Steps

1. Sync the git repo in your windows laptop
   https://github.com/chitwanhumad/hg_datapipeline
2. Open config.ini file to update your paths and SQL Server database credentials. Update server name, user and password.
3. Configure your root directory where the source file will be placed.
4. Run \ddl\dbsetup.sql using SQL Server management studio. This script will create all required databases, tables and other db objects.

## Set up local Prefect Environment

1. Prefect Install packages from requirements.txt for \prefect\ requirements.txt
   pip install -r requirements.txt
2. Refer \Environment.txt (complete - Solution Installation Steps first and come back here) or follow the below steps:
   - create project folder
     D:\

```
mkdir HGInsights
cd HGInsights
```

- set up virtual env

  ```
  python -m venv hg_venv
  ```

- activate hg_venv

  ```
  hg_venv\Scripts\activate
  ```

- install prefect

  ```
  pip install prefect
  ```

- create root directory

  ```
  mkdir D:\HGInsights\source\
  ```

- create Archive directory

  ```
  mkdir D:\HGInsights\archive\
  ```

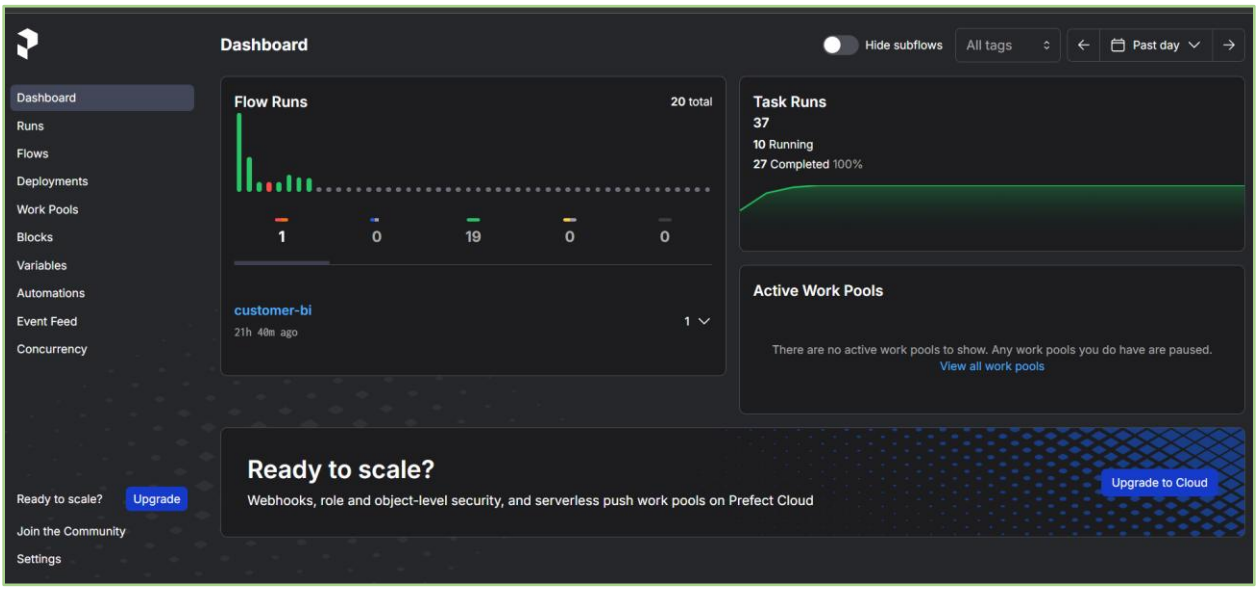
- create Git directory

  ```
  mkdir D:\HGInsights\Git\
  ```

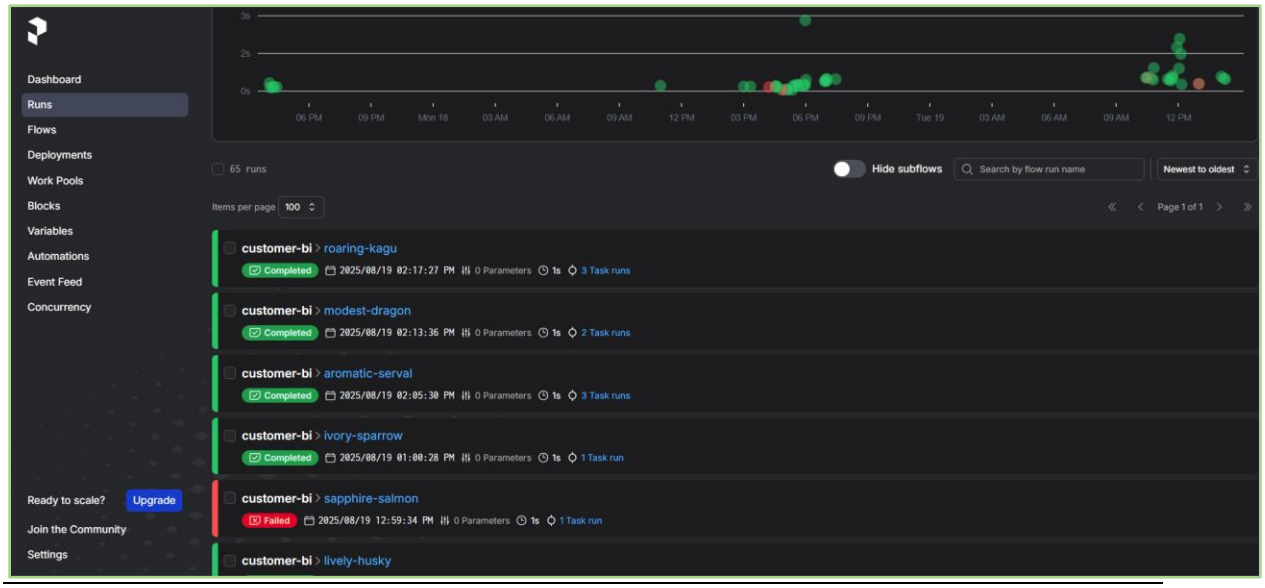
3. Access Prefect dashboard here:

   http://127.0.0.1:4200

   Dashboard looks as follows:



Runs:




## Start Prefect Server

1. Run command to start server
    - Open file \prefect\start_prefect_server.bat
    - Review and update your virtual environment paths
    - Run following to start server

      start_prefect_server.bat

## Set up and Run Data pipeline

1. Copy the source file into the <>/source/*.csv folder, there could be more than 1 incoming files.
2. **Run /workflow/main.py**
3. Check Run status and logs in by using following queries

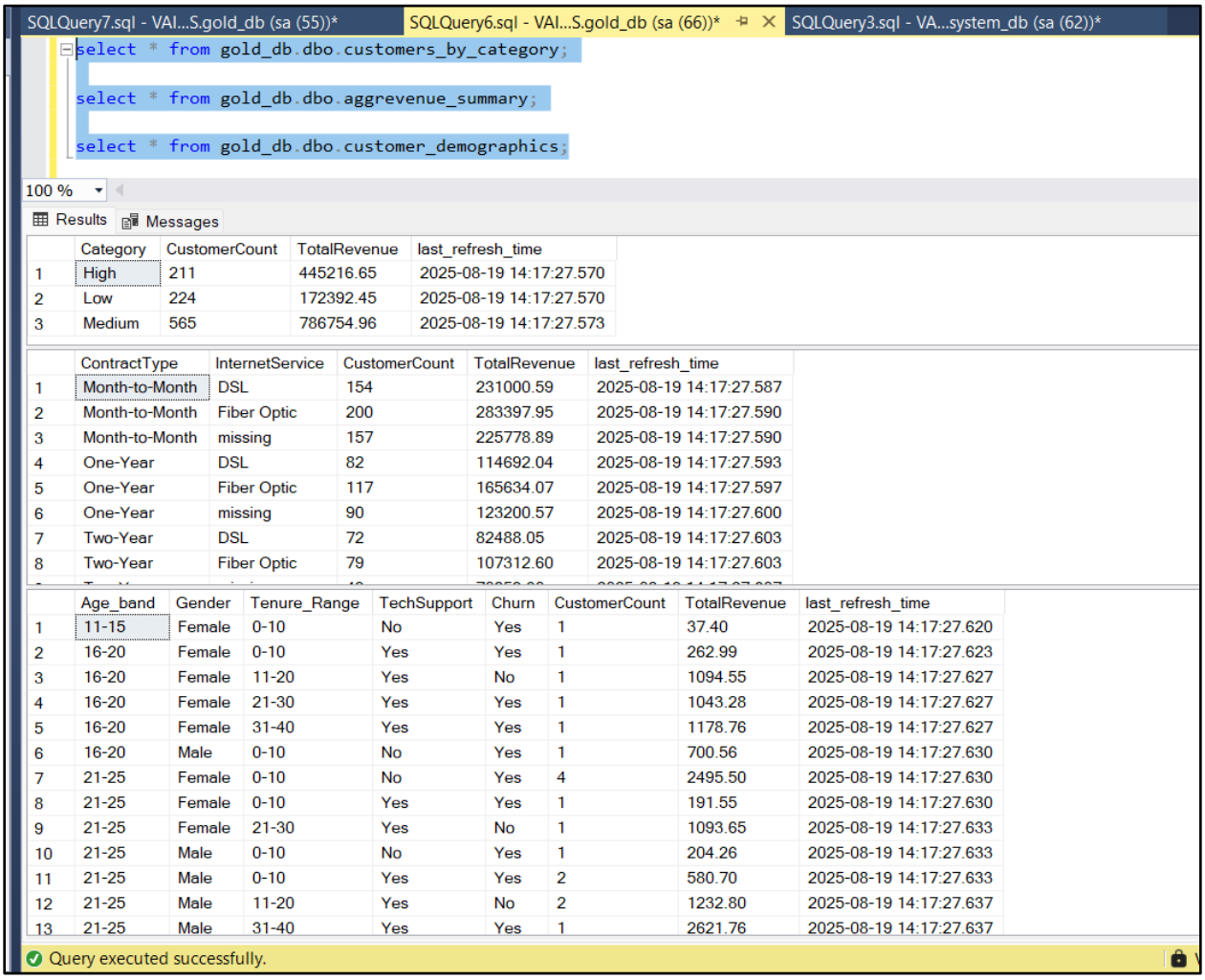
```
select * from system_db.dbo.ach_runs where runid = ?;

select * from system_db.dbo.ach_logs where runid = ?;
```

4. If now ERROR in the above step, verify your data in bronze and silver layers databases by using following queries

```
select * from bronze_db.dbo.raw_customers where runid = 51

select * from silver_db.dbo.customers where is_current = 'Y' order by CustomerID;
```

5. To view the modeled data for reports, run following sqls:

```
select * from gold_db.dbo.customers_by_category;

select * from gold_db.dbo.aggrevenue_summary;

select * from gold_db.dbo.customer_demographics
```




## Set up Local Superset Environment

a. Download install Superset (superset should be installed outside prefect directory. e.g. D:\superset\ directory). Refer git **\superset\requirements.txt** for all packages.

   a. Python virtual environment

python -m pip install --upgrade pip setuptools wheel

python -m venv venv

**venv\Scripts\activate**
(every time when you start server)

b. Install apache-superset

pip install apache-superset

c. create a new secret

python -c "import secrets; print(secrets.token_urlsafe(64))"

d. save this secret inside into **D:\superset\superset_config.py**

import os

SECRET_KEY = "my_random_long_secret_key_123!@#"

Note: example superset_config.py file canbe referred from

\superset\superset_config.py

e. Make sure you have the below package

pip install marshmallow==3.20.1

pip install pymssql # sql server connector

pip install sqlalchemy==2.0.25

pip install pyodbc

**pip install --upgrade apache-superset**

f. Set up variables and flask application

**set SUPERSET_CONFIG_PATH=D:\superset\superset_config.py**

**set FLASK_APP=superset.app:create_app()**

(every time when you start server)

g. run db upgrade command

superset db upgrade
(one time only)

h. To create admin user/password for your set up, complete the prompts:

superset fab create-admin

(one time only)

i. Load examples

superset load_examples
(one time only)

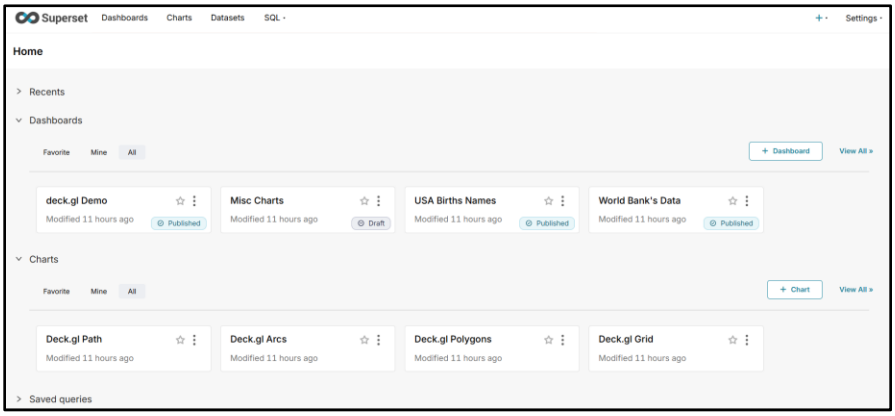
j. Initialize superset

**superset init**
(every time when you start server)

k. Start server
**superset run -h localhost -p 8088**
(every time when you start server)

l. Access server
http://localhost:8088

m. With the login with admin user you should be able to see the home page on Superset




## Start Superset Server:

1. Refer file \superset\start_superset_server.bat
2. Set up path where your python virtual environment for superset is available
3. Run the batch file

**\superset\start_superset_server.bat**

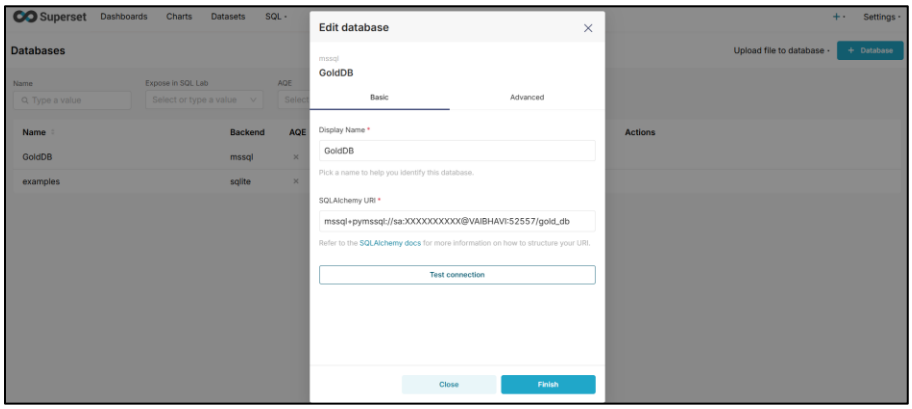
4. Break it to stop server

## Create Database Connection

1. Create datasource for gold_db

   Name: GoldDB

   SQL Alchemy URI: mssql+pymssql://sa:unica*03@VAIBHAVI:52557/gold_db

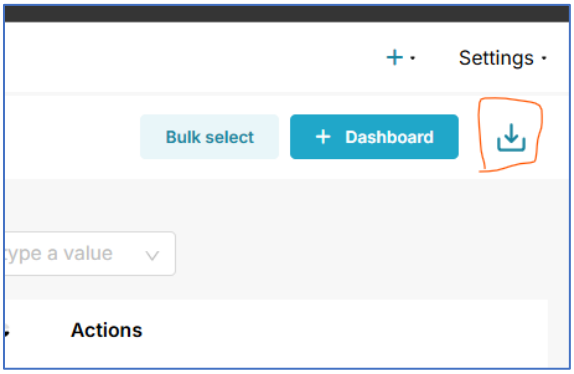



   NOTE: In my case the instance name is SQLEXPRESS, it's dynamic port is 52557 so used dynamic port, check TCP/IP settings to fetch the same.

2. Test connection, save datasource name as **GoldDB**

## Import Dashboards

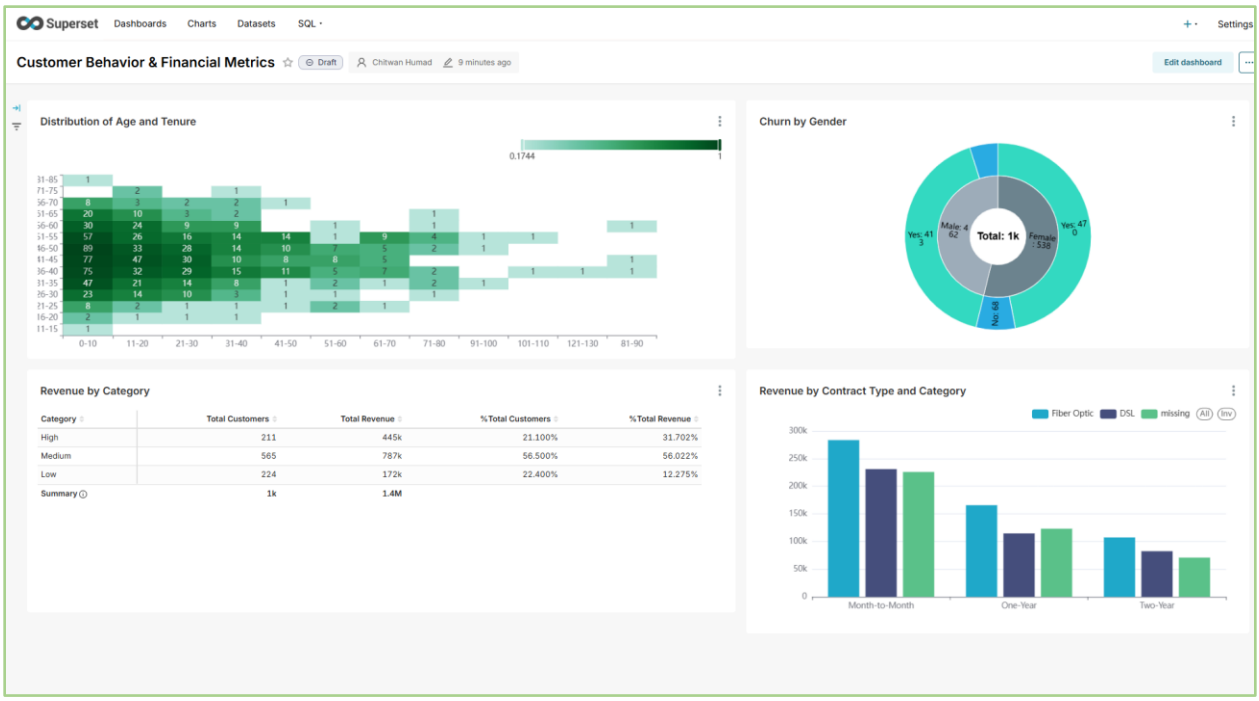
1. Access /dashboards/customer_dashboard.zip from the git repository
2. Click Import



3. Select the zip file and import
4. Verify import
    a. Go to Datasets, you should see
        i. customers_by_category
        ii. customer_demographics
        iii. aggrevenue_summary
    b. Go to Charts, you should see
        i. Distribution of Age and Tenure
        ii. Churn by Gender
        iii. Revenue by Category
        iv. Revenue by Contract Type and Category
    c. Go to Dashboards, you should see
        i. Customer Behavior & Financial Metrics
5. Recheck the Database connection and re-test the GoldDB datasource

## Execute Dashboard

Click **Customer Behavior & Financial Metrics**



## Optional Steps

### Deploy Pipeline

### Schedule Pipeline

## Troubleshoot

- Make sure you work in different python virtual environment for prefect and superset setups
- Run pip upgrade commands to fix issues:
    1. pip install --upgrade prefect
    2. pip install --upgrade apache-superset

## Disclaimer

- This document is for Chitwan's assessment only.
- The code and the content should not be used for any other purposes.
- The code has not been tested hence it may have some functional or non-function data issues.
- In case, the reviewer is unbale to follow any step of the document or caught any error, reach out to author Chitwan Humad for assistance. Email chitwanhumad@gmail.com