**Enterprise Data Warehouse Implementation Using Medallion Architecture in Microsoft Fabric**

**Project Overview**

Traditional data warehouses lack scalability, while data lakes struggle with governance and performance. This project aims to implement a **Lakehouse Architecture** using **Microsoft Fabric** and the **Medallion Architecture (Bronze, Silver, Gold)** to optimize data ingestion, transformation, and analytics. By leveraging **OneLake, Delta Tables, and Direct Query,** the goal is to enhance **data accessibility, reliability, and performance** for real-time decision-making.

**Business Problem**

Businesses face challenges with scalable data management and real-time analytics. This project leverages Microsoft Fabric’s Medallion Architecture to ensure efficient data processing, quality, and governance for better decision-making.

**Objective**

Implement a Lakehouse-based Medallion Architecture (Bronze, Silver, Gold) in Microsoft Fabric for optimized data storage, transformation, and analytics. Automate data ingestion, cleansing, and aggregation to ensure high-quality, analytics-ready data. Leverage Fabric Pipelines, Delta Tables, and DirectQuery for real-time BI and AI workloads while ensuring scalability and governance**.**

**Scope of Work**

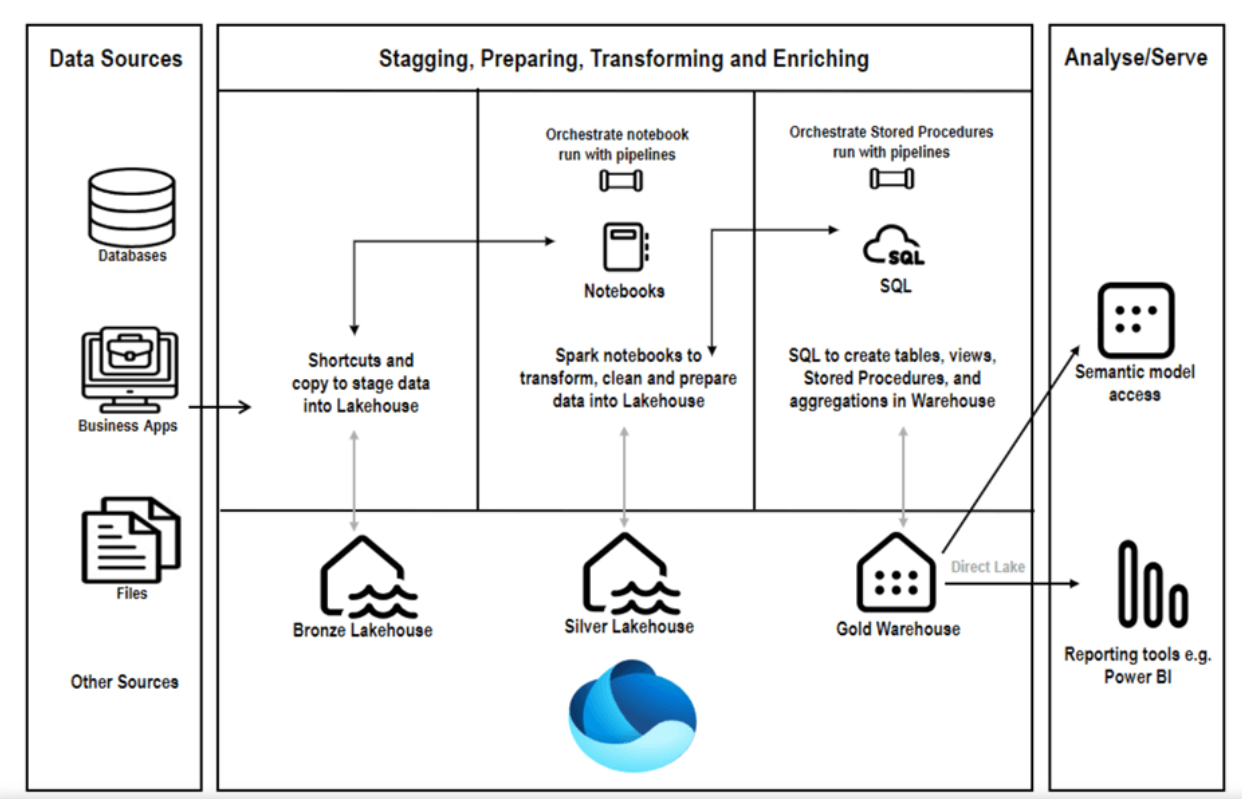
Implementing the Medallion Architecture within Microsoft Fabric involves a structured approach to data management by organizing data into three distinct layers:

1. **Bronze Layer (Raw Data)**
2. **Silver Layer (Cleansed and Processed Data)**
3. **Gold Layer (Curated and Aggregated Data)**

To set up the Medallion Architecture in Microsoft Fabric, the following steps are typically undertaken:

1. **Create Two Fabric Lakehouses (Bronze and Silver Layers)** – Establish separate Lakehouses and log metadata, including **creation timestamps and ownership details**.
2. **Ingest Raw Data into the Bronze Layer** – Record **source details, ingestion time, row count, and any errors** for monitoring failures.
3. **Transform Data in the Silver Layer** – Log **data quality checks, transformation rules applied, and rejected records** for debugging.
4. **Curate Data in the Gold (Data Warehouse) Layer** – Maintain an audit trail of **aggregations, business logic applied, and load success/failure metrics**.
5. **Set Up and Schedule Pipelines for Data Orchestration** – Track **execution time, job status, and anomalies** to ensure pipeline efficiency.
6. **Enable Data Access for Reporting** – Log **user access, query execution details, and modifications** to support compliance and security.

# Architecture



**Technologies & Tools:**

* **PySpark** for data manipulation and analysis.
* **Fabric Notebook** or Python IDE for writing and running the code.
* **Pandas** (optional) for additional data handling or export.

**Timeframe:**

* **Total time estimate**: 20-30 hours.
* Data loading and preprocessing: 5-7 hours.
* Transformation and aggregation: 8-10 hours.
* Power BI Dash Board: 5-7 hours.
* Documentation and final report: 3-5 hours.

**Learning Outcomes:**

* Gain experience in working with PySpark for large-scale data processing.
* Learn how to load, clean, and preprocess data using PySpark.
* Understand how to perform aggregation, grouping, and sorting operations.
* Explore how to identify trends and patterns in large datasets.

**Important Instructions for Project(Validation Parameter Also).**

1.You have to create a Table in such a way that you can use it on Power BI or Fabric .

2.You have to update the code Snippet in Proper format to Git Hub.

3.You have to Add Comment to each code Snippet .

4.You have to use standard naming Convention while writing code.

**Documentation & Reporting**:

1. Document the process in a Fibric Notebook or a Python script, explaining each step.
2. Prepare a brief report summarizing key findings, insights, and any challenges faced during the project.