Scope of Work (SoW) for Transportation & Logistics Data Processing & Analysis

1. Project Overview

This project focuses on processing and analyzing transportation and logistics data using PySpark, MySQL, and Power BI, following the Medallion Architecture. The goal is to ingest, clean, transform, and aggregate data related to transportation routes, vehicle performance, delivery times, and shipment statuses to generate meaningful business insights. The project will be completed within 30 hours using a small dataset stored in CSV files. The final output will be a Power BI dashboard displaying key metrics like route optimization, delivery efficiency, and fleet performance.

2. Software & Tools Required

The intern will set up the following tools on their local machine:

PySpark: Data processing & transformation

Jupyter Notebook: Running PySpark scripts interactively

MySQL: Storing transformed data & final aggregated results

Power BI: Data visualization & reporting

3. Medallion Architecture Overview

The project follows the Medallion Architecture, which consists of three structured layers:

Bronze Layer: Stores raw data with minimal processing (Parquet files)

Silver Layer: Cleansed, transformed, and enriched data (Parquet & MySQL tables)

Gold Layer: Aggregated data for reporting & dashboards (MySQL tables)

4. Data Sources & Schema

Raw Data (CSV Files) → Bronze Layer

The following CSV files will serve as the raw input data, which will be ingested into the Bronze Layer (Parquet format):

1. delivery_data.csv

Column Name	Type	Description

delivery_id	int	Unique Delivery ID (Primary Key)
vehicle_id	int	Foreign Key to Vehicle
route_id	int	Foreign Key to Route
driver_id	int	Foreign Key to Driver
delivery_date	date	Date of the delivery
delivery_time	float	Time taken for delivery (in hours)
distance_covered	float	Distance covered in kilometers
delivery_status	varchar	Status of delivery (Completed/Failed)
source_file	varchar	Name of the source file

2. vehicle_data.csv

Column Name	Type	Description
vehicle_id	int	Unique Vehicle ID (Primary Key)
vehicle_type	varchar	Type of vehicle (e.g., Truck, Van)
fuel_efficiency	float	Fuel efficiency (km per liter)
capacity	int	Maximum capacity (in tons)

3. route_data.csv

Column Name	Type	Description
route_id	int	Unique Route ID (Primary Key)
start_location	varchar	Starting location of the route
end_location	varchar	End location of the route
distance	float	Distance of the route in kilometers

4. driver_data.csv

Column Name	Type	Description
driver_id	int	Unique Driver ID (Primary Key)
driver_name	varchar	Name of the driver
experience_years	int	Years of experience
rating	float	Driver rating (1-5)

5. Bronze Layer (Raw Data Storage in Parquet Format)

The Bronze Layer stores the raw CSV data as-is but in Parquet format for efficiency. Each bronze table contains the raw data plus audit columns (ingestion_date, source_file) to track data lineage.

delivery_data_bronze.parquet → delivery_data.csv

vehicle_data_bronze.parquet → vehicle_data.csv

route_data_bronze.parquet → route_data.csv

driver_data_bronze.parquet → driver_data.csv

6. Silver Layer: Data Cleaning, Enrichment & Transformation

Objective

The Silver Layer prepares cleaned, structured, and enriched data for analysis by performing:

Data Cleaning: Removing records with null values in critical columns.

Joining Related Tables: Adding vehicle details, driver details, and route descriptions.

Computing Additional Fields: E.g., fuel_consumed = distance_covered / fuel_efficiency.

Storing Data in Both Parquet and MySQL for further aggregation.

delivery_data_silver.parquet → MySQL Schema: silver_db.delivery_data_silver

Column Name	Type	Description
delivery_id	int	Unique Delivery ID
vehicle_type	varchar	Type of vehicle (e.g., Truck, Van)
driver_name	varchar	Name of the driver
route_name	varchar	Name of the route (Start to End Location)
delivery_time	float	Time taken for delivery (in hours)
distance_covered	float	Distance covered (in km)
delivery_status	varchar	Status of delivery (Completed/Failed)
fuel_consumed	float	Fuel consumed during the delivery (in liters)
processed_date	datetime	Date when the data was transformed

7. Gold Layer: Data Aggregation for Reporting

Objective

The Gold Layer will create a final aggregated dataset in MySQL for Power BI reports, computing key business metrics.

Operations Performed in MySQL (Using Silver Layer as Source):

Route Optimization Analysis

Total deliveries per route.

Average delivery time per route.

Average fuel consumption per route.

Fleet Performance

Total deliveries per vehicle.

Average distance covered per vehicle.

Average fuel efficiency per vehicle.

Driver Performance

Total deliveries per driver.

Average delivery time per driver.

Driver rating analysis (average rating).

Gold Table Schema in MySQL (gold_db.transportation_gold)

Column Name	Type	Description
route_name	varchar	Name of the route
total_deliveries	int	Total number of deliveries per route
avg_delivery_time	float	Average delivery time per route (in hours)
avg_fuel_consumed	float	Average fuel consumed per delivery (in liters)
vehicle_id	int	Vehicle ID
total_distance	float	Total distance covered by the vehicle (in km)
fuel_efficiency	float	Average fuel efficiency of the vehicle (km/l)
driver_name	varchar	Name of the driver
total_deliveries_by_driver	int	Total deliveries completed by the driver
driver_rating	float	Average driver rating
report_date	date	Date for time-based analysis

8. Power BI Reports & Dashboard

After processing data through the Bronze, Silver, and Gold layers, the Gold tables in MySQL will serve as the data source for Power BI. Interns are expected to create a dashboard with the following:

KPIs:

1. Total Deliveries (Sum of total deliveries per route/vehicle/driver)
2. Average Delivery Time (Average delivery time per route/vehicle/driver)
3. Fuel Efficiency (Average fuel efficiency per vehicle)
4. Driver Performance (Average driver rating)
5. Delivery Status (Completed vs Failed)

Visualizations:

1. **Route Optimization Chart** – Line chart showing average delivery times and fuel consumption per route.
2. **Fleet Performance Analysis** – Bar chart visualizing total deliveries per vehicle and fuel efficiency.
3. **Driver Performance Analysis** – Scatter plot to show driver performance based on deliveries completed and average ratings.
4. **Delivery Status Distribution** – Pie chart showing the distribution of completed and failed deliveries across routes.
5. **Time-based Delivery Trends** – Line chart to show monthly/weekly delivery performance trends.

Additional Instructions:

- Connect MySQL Gold Layer to Power BI.
- Ensure data refresh is properly configured to update automatically.
- Apply basic formatting, labels, and filters for usability and clarity.

9. Final Deliverables

The following deliverables will be provided upon the completion of the project:

1. **Silver & Gold Layers in MySQL:** Containing cleaned, enriched, and aggregated data ready for reporting.
2. **Power BI Dashboard:** Visualizations based on the Gold Layer data, displaying KPIs and supporting the key business insights.
3. **Project Documentation:** Detailed explanations of the data transformations, aggregation steps, and business logic applied. Documentation will also include:
   - Overview of each layer (Bronze, Silver, Gold).
   - Explanation of data cleaning, enrichment, and transformation steps.
   - Instructions for using the Power BI dashboard and accessing the data.
4. **Git Repository:** A Git repository containing all code, SQL scripts, and relevant files, along with a readme to guide through the setup and execution process.

Project Instructions:

1. Follow the Medallion Architecture for data processing.
2. Use proper naming conventions for all target files and MySQL database tables.
3. Maintain clean and consistent coding standards in Python scripts and SQL queries for the Gold table.
4. Automate the entire data pipeline from raw data ingestion to the final Gold layer.
5. Implement proper auditing at each stage of the pipeline to track data processing.
6. Ensure clear documentation of the entire project, including transformation steps and usage guidelines.
7. Use GIT to store all code, SQL scripts, and relevant files.
8. Include the GIT repository URL in the final documentation for submission.