# **Project Execution Document**

## **Data Lake Analytics**

## 1. Introduction

The **Data Lake Analytics Project** was designed to demonstrate how cloud-native tools can be used to manage, organize, and analyze large volumes of data. The original scope involved using **Azure Data Factory (ADF)** to orchestrate data movement into **Azure Data Lake Storage (ADLS)**, and then leveraging **Azure Databricks** for advanced data processing.

However, due to environment constraints (Databricks Community Edition not supporting ADLS mounting), the project was executed **entirely using ADF and ADLS**. This showcases the flexibility of Azure services in handling end-to-end data engineering tasks without external compute engines.

## 2. Objectives

* To set up a **Data Lake architecture** using Azure Data Lake Storage.
* To design and implement **ETL pipelines** in Azure Data Factory.
* To organize raw, cleaned, and transformed data into **Bronze, Silver, and Gold layers**.
* To enable structured analytics-ready data for downstream use.
* To document execution challenges, solutions, and outcomes.

## 3. Step-by-Step Execution

### ****Step 1: Requirement Analysis****

* Defined the scope: ingest raw data, clean/transform it, and make it analytics-ready.
* Chose ADLS as the storage backbone.
* Planned data flow architecture using Bronze → Silver → Gold layer design.

### ****Step 2: Environment Setup****

* Created a **Resource Group** in Azure.
* Provisioned **Azure Data Lake Storage Gen2** with hierarchical namespace enabled.
* Created **folders** for Bronze (raw), Silver (cleaned), and Gold (curated).
* Set up **Azure Data Factory** with Managed Identity to access ADLS.

### ****Step 3: Data Ingestion (Bronze Layer)****

* Connected ADF to the data source (CSV/JSON files).
* Built pipelines to copy raw data directly into the **Bronze container**.
* Ensured schema-on-read so that raw data is preserved for future reprocessing.

### ****Step 4: Data Transformation (Silver Layer)****

* Used **ADF Data Flows** to clean and transform data:
  + Removed nulls and duplicates.
  + Standardized column formats (dates, numerical values).
  + Derived new calculated fields.
* Stored the cleaned datasets into the **Silver container**.

### ****Step 5: Data Structuring (Gold Layer)****

* Designed **dimension tables (Customer, Product, Date)** and **Fact table (Sales)**.
* Added surrogate keys (customer\_key, product\_key, date\_key).
* Derived additional attributes (e.g., year, month, day from invoice date).
* Stored the final curated data into the **Gold container**, making it analytics-ready.

### ****Step 6: Validation & Testing****

* Validated record counts between source and target.
* Verified transformations (e.g., duplicates removed, correct keys assigned).
* Ensured pipeline success with monitoring and alerts in ADF.

### ****Step 7: Scheduling & Automation****

* Configured ADF **triggers** for scheduled pipeline runs.
* Enabled monitoring dashboards in ADF to track pipeline execution and failures.

## 4. Challenges Faced & Resolutions

| **Challenge** | **Resolution** |
| --- | --- |
| **Databricks Community Edition did not support ADLS mounting** | Pivoted the project to execute all transformations directly in **ADF Data Flows**, removing dependency on Databricks. |
| **403 Authorization errors while accessing ADLS** | Granted correct **IAM role assignments** and used **Managed Identity** authentication for ADF. |
| **Schema mismatch between raw and transformed data** | Implemented schema drift handling in ADF pipelines. |
| **Performance issues during transformations** | Used partitioning and optimized Data Flows to handle large datasets efficiently. |
| **Testing data integrity** | Created validation checks (record counts, null checks) within pipelines. |

## 5. Results & Deliverables

* Successfully designed and deployed **end-to-end ETL pipelines** in ADF.
* Implemented a **multi-layered data lake structure (Bronze → Silver → Gold)**.
* Created **dimension and fact tables** for structured reporting.
* Automated workflows with scheduling and monitoring in ADF.
* Delivered **analytics-ready data** stored in the Gold layer.

## 6. Summary

The project demonstrates the power of **Azure Data Factory and ADLS** in building a **data lakehouse architecture** without requiring external compute engines like Databricks. While the original plan included Databricks for advanced transformations, all processing was successfully handled within ADF.

This resulted in a scalable, cost-effective, and fully cloud-native data engineering solution. The system ensures clean, structured, and analytics-ready data for business intelligence and reporting.

## 7. Key Learnings & Recommendations

* **Flexibility:** Even without Databricks, ADF and ADLS can handle full ETL processes effectively.
* **Security:** Always configure IAM roles and Managed Identities early to avoid pipeline failures.
* **Optimization:** Data partitioning and schema drift handling are crucial for large datasets.
* **Future Scope:** Integrating Power BI or Synapse Analytics for visualization can enhance reporting.