ISM TECHNICAL SPECIFICATION DOCUMENT

Fabric Setup

Torex Gold

FABRIC IMPLIMENTATION

v.1.0

March 12, 2025

**Document Control**

Owner: Colin Jessop <Colin.Jessop@kyndryl.com>

Requested By: Rohit Tellis <Rohit.Tellis@torexgold.com>

Authors: Krishna kishore dabula ([krishna.kishore.dabula@kyndryl.com](mailto:krishna.kishore.dabula@kyndryl.com), [krishna.kishore.dabula@torexgold.com](mailto:krishna.kishore.dabula@torexgold.com)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| Mar 11 2025 | 1.0 | Initial Draft | KK |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1 Introduction 4](#_Toc192667690)

[1.1 Purpose of this document 4](#_Toc192667691)

[2 Azure and Fabric 5](#_Toc192667692)

[2.1 Fabric configuration Overview 5](#_Toc192667693)

[2.2 Dev ops and Deployment Setup 8](#_Toc192667694)

[3 Approvals 18](#_Toc192667695)

[4 Appendices 18](#_Toc192667696)

[4.1 Annex 1: Data Architecture 18](#_Toc192667697)

[4.2 Annex 2: DWH-End to End Implementation: 18](#_Toc192667698)

# Introduction

This document outlines the requirements for the creating a Microsoft Fabric environment and setting up a structured deployment pipeline in Torex. Microsoft Fabric is an all-in-one data platform that integrates data engineering, data science, real-time analytics, and business intelligence in a single unified platform.

## Purpose of this document

Microsoft Fabric is a unified platform that brings together various components under a single ecosystem, including:

* **Data Engineering** – Manage large-scale data processing.
* **Data Factory** – Build data pipelines for ETL processes.
* **Data Warehouse** – Store structured data for analysis.
* **Lakehouse** – Handle both structured and unstructured data.
* **Power BI** – Create interactive reports and dashboards.
* **Real-Time Analytics** – Process streaming data in real time.
* **Data Science** – Train and deploy machine learning models.

.

# Azure and Fabric

## Fabric configuration Overview

**Create a Workspace:** Begin by creating a workspace in Microsoft Fabric. This workspace will serve as the foundation for your development and deployment activities.

1. **➡️ Create a Workspace**

* Go to the **Microsoft Fabric Portal** → **Workspaces** → **Create New Workspace**
* Provide details like:
  + **Workspace Name**
  + **Capacity Assignment** (if using Premium)
  + **User Access Control**

1. **➡️ Enable Fabric Components**

* Enable specific workloads such as:
  + Data Engineering
  + Data Factory
  + Real-Time Analytics
  + Power BI
  + Data Warehouse
* **🔹 Step 2: Build and Configure Your Solution**

1. **➡️ Data Ingestion (ETL) – Using Data Factory**

* Create a new **Data Pipeline**
* Define data sources and destinations
* Configure data transformations
* Test the pipeline execution

1. **➡️ Create a Data Warehouse**

* Create a new **Fabric Warehouse**
* Define table structure and relationships
* Load sample data

**Workspace Names:**

**Click on the workspace and create workspace as below.**

**A screenshot of a computer

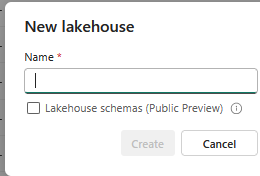
AI-generated content may be incorrect.**

**A screenshot of a phone

AI-generated content may be incorrect.**

**Lake house Names:**

**Click on lakehouse and create new lakehouse**

****

A screenshot of a computer

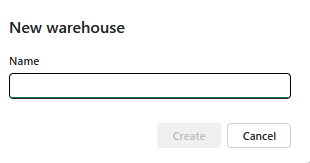
AI-generated content may be incorrect.

**A screenshot of a phone

AI-generated content may be incorrect.**

**Warehouse Name:**

**Click on warehouse and create new warehouse**

****

[**A screenshot of a phone

AI-generated content may be incorrect.**](https://app.fabric.microsoft.com/groups/c6b23ee3-fa7f-4528-a989-143b70cfec15/datawarehouses/f9090848-c31b-4852-92e8-09c4432421d0?experience=data-engineering)

## Dev ops and Deployment Setup

Below are the steps involved for dev ops and deployment setup.

**Integrate with Source Control**

* **Connect to Git Repository:** Navigate to your workspace settings and connect the workspace to an Azure DevOps repository. This integration enables version control and collaborative development.

**Establish Deployment Pipelines**

* **Create Deployment Pipelines:** Utilize Fabric's deployment pipelines to manage content promotion across different environments (e.g., Development, Test, Production). This ensures a structured and controlled release process.

**Automate Deployment Processes**

* **Configure CI/CD Pipelines:** Set up Continuous Integration and Continuous Deployment (CI/CD) pipelines using Azure DevOps. This automation facilitates consistent and efficient deployments

A screenshot of a computer

AI-generated content may be incorrect.

Below are the step by step guide for dev ops creation :

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

Below are the steps for Fabric deployment :

* **Create a Workspace**

Go to **Microsoft Fabric** portal.

Navigate to **Workspaces** → **New Workspace**.

Configure the following:

* + **Name** – Provide a meaningful name.
  + **Premium Capacity** – Assign a Fabric capacity if required.
  + **Access Control** – Set user permissions.
* **Develop Your Solution**

You can create different types of Fabric items based on your requirements:  
✅ **Data Pipeline** – For ETL (Extract, Transform, Load) processes.  
✅ **Lakehouse** – For managing large-scale data lakes.  
✅ **Warehouse** – For structured data storage.  
✅ **Reports/Dashboards** – For visualization using Power BI.  
✅ **Notebooks** – For data science and machine learning models.

* **Set Up Source Control**

Link your workspace to a **Git Repository** in Azure DevOps:

* + Go to **Workspace Settings** → **Git Integration**.
  + Connect to Azure DevOps repository → Select branch.

Commit and track changes directly from the Fabric workspace.

* **Creating a Deployment Pipeline**

**Create a Pipeline**:

* + Go to **Deployment Pipelines** → **Create Pipeline**.
  + Name it based on the environment (e.g., Dev → Test → Prod).
* **Add Artifacts:**
  + Select the Fabric workspace.
  + Include **data pipelines, warehouse, reports**, etc.

**Configure Pipeline Stages**:

* + Add stages for **Development, Testing, and Production**.
  + Set up data source rules and environment-specific settings.

**5. Deploy to Test Environment**

Click **Deploy** to move from **Development** to **Test**.

Validate data integrity, report functionality, and pipeline execution.

**6. Approve and Promote Production**

* After successful testing, promote to the **Production** environment.
* Monitor deployment logs and resolve any issues.
* **7. Monitor and Maintain**
* Use **Fabric Monitoring Hub** to track pipeline performance.
* Configure alerts and logging.
* Continuously update and manage the solution using Git versioning.

A screenshot of a computer

AI-generated content may be incorrect.

* Select stage item from development(Pipleine, lakehouse and warehouse etc)
* Click on deployment to move change from Dev to Prod

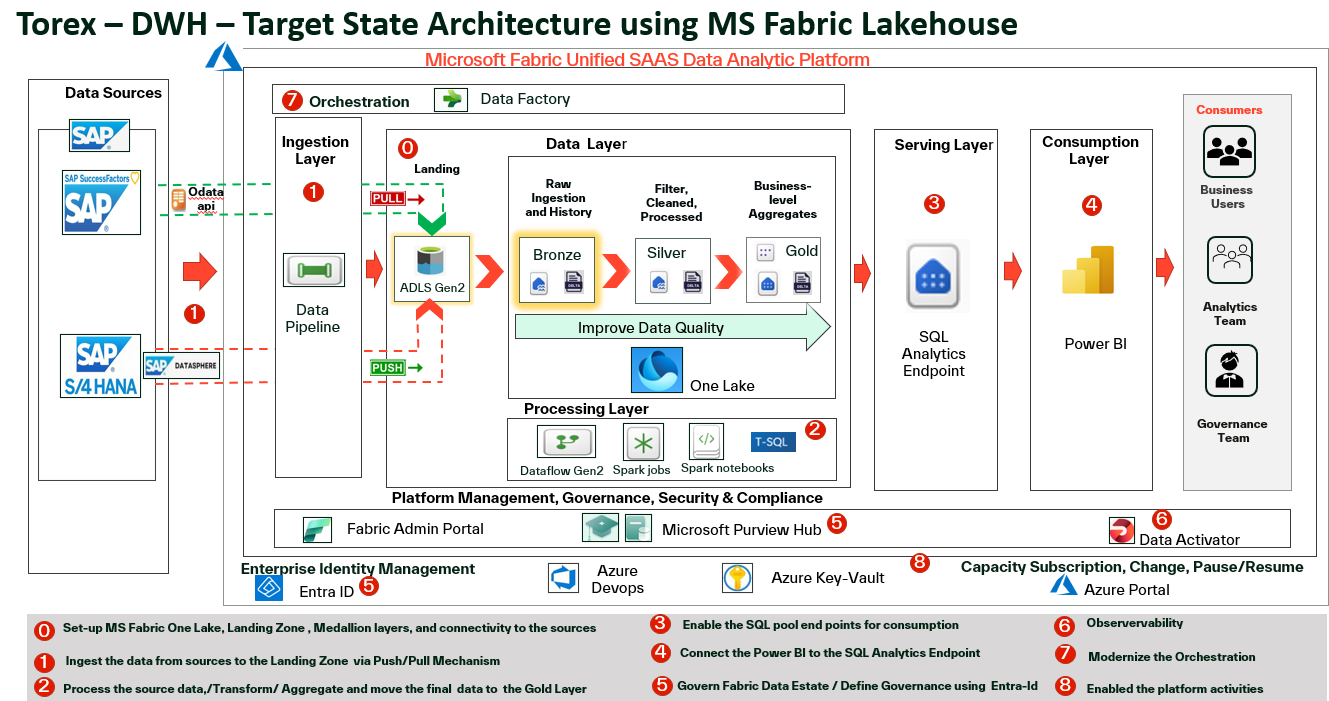
# Approvals

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

# Appendices

## Annex 1: Data Architecture

This diagram presents the data architecture that supports the MML solution:



## Annex 2: DWH-End to End Implementation:

