# Product Backlog: Acme-Sub Future State System Implementation

## Feature 1: Secure OT Data Ingestion and Lakehouse Foundation

**Description:** Establish a robust, scalable, and secure pipeline for ingesting high-volume, high-velocity IoT sensor data from on-premises plant equipment into Fabric's OneLake, laying the groundwork for advanced analytics. This includes configuring direct data transmission from on-premises HiveMQ brokers to Event Hub and subsequently to a Bronze Lakehouse.

### User Story 1.1: Ingest Raw OT Data into Event Hub

**Description:** As an OT Data Engineer, I want to securely transmit high-volume, high-velocity IoT sensor data directly from on-premises HiveMQ brokers to Event Hub, so that cloud-based ingestion and processing can begin without relying on a cloud-based HiveMQ instance.

**Acceptance Criteria:**

* HiveMQ on-prem successfully sends real-time IoT messages to Event Hub.
* Data transmission is encrypted using industry-standard protocols such as TLS/SSL.
* Robust device authentication mechanisms (e.g., certificate-based authentication) are implemented for secure communication from on-prem HiveMQ to Event Hub.
* Event Hub is configured to receive and store incoming IoT data streams reliably.

**Tasks for US 1.1:**

#### Task 1.1.1: Provision Event Hub Namespace and Event Hubs

*Description:* Create an Event Hubs Namespace and individual Event Hubs within the Acme-Sub subscription, configuring throughput units and partition counts (e.g., 32-64 partitions for high concurrency) suitable for ingesting 2.7 billion rows/month of IoT data per plant line.

#### Task 1.1.2: Configure Network Connectivity to Event Hub

*Description:* Establish secure network connectivity from the on-premises environment (via existing VPN and ACME's peered VNet/firewall) to the Event Hub. Investigate and configure Private Endpoints for Event Hub to ensure private connectivity and reduce public internet exposure, if current limitations on Fabric integration with private endpoints are resolved.

#### Task 1.1.3: Update HiveMQ Configuration for Event Hub Integration

*Description:* Modify the on-premises HiveMQ configuration (Enterprise version) to establish a direct native connection to Event Hub, replacing the current direct Data Lake connection used in POC. Ensure message formats are compatible with Event Hub ingestion.

#### Task 1.1.4: Implement TLS/SSL and Device Authentication for HiveMQ-Event Hub

*Description:* Enable and enforce TLS/SSL (e.g., TLS 1.2 or 1.3) for all communications between on-premises HiveMQ and Event Hub. Implement certificate-based authentication or mutual TLS to verify the identity of HiveMQ clients and prevent unauthorized data injection.

#### Task 1.1.5: Validate IoT Data Flow to Event Hub

*Description:* Conduct comprehensive end-to-end testing, starting from SCADA (Ignition) and HighByte, to confirm that IoT signals are successfully transmitted via HiveMQ and ingested into Event Hub. Monitor Event Hub metrics (e.g., incoming messages, throughput) to validate expected volume and velocity.

### User Story 1.2: Ingest Streaming IoT Data into Bronze Lakehouse

**Description:** As a Data Engineer, I want to use Event Streams within Fabric to process the streaming IoT data from Event Hub into a Bronze Lakehouse in OneLake, so that raw data is readily available for initial processing within Fabric.

**Acceptance Criteria:**

* Event Streams in Fabric are configured to consume data from the designated Event Hub.
* Streaming IoT data is continuously ingested into the Bronze Lakehouse in OneLake.
* The Bronze Lakehouse successfully stores the raw parquet files and maintains the historical data from the plant line for future use cases.

**Tasks for US 1.2:**

#### Task 1.2.1: Provision Bronze Lakehouse in Fabric

*Description:* Create a dedicated Bronze Lakehouse within the Fabric environment, ensuring appropriate storage configuration (e.g., using Delta Lake format for schema evolution and ACID properties) for raw IoT data.

#### Task 1.2.2: Configure Event Streams for Bronze Lakehouse Ingestion

*Description:* Set up Event Streams within Fabric, defining the Event Hub as the source and the Bronze Lakehouse as the destination. Explore basic stream processing capabilities of Event Streams for light data enrichment or parsing (e.g., extracting fields from complex JSON payloads) if beneficial before landing in Bronze.

#### Task 1.2.3: Validate Event Stream to Bronze Lakehouse Flow

*Description:* Monitor the Event Stream jobs and the Bronze Lakehouse for continuous and reliable ingestion of streaming data, verifying data freshness, format (e.g., Parquet or Delta), and completeness.

#### Task 1.2.4: Implement Initial Data Integrity Checks for Bronze Layer

*Description:* Establish basic data integrity checks (e.g., row counts, schema validation) for the ingested raw data in the Bronze Lakehouse to identify any immediate issues like data corruption or missing data during ingestion.

--------------------------------------------------------------------------------

## Feature 2: Advanced Data Transformation and Medallion Architecture

• **Description:** Implement the Medallion Architecture (Bronze, Silver, Gold) within Fabric to refine and transform raw IoT data into standardized, enriched, and modeled datasets, suitable for advanced analytics, machine learning, and reporting.

### User Story 2.1: Cleanse and Standardize Data in Silver Lakehouse

**Description:** As a Data Engineer, I want to use Spark Structured Streaming notebooks and Fabric Pipelines to cleanse, standardize, and enrich raw IoT data from the Bronze Lakehouse into a Silver Lakehouse, so that data quality is improved and it's prepared for initial modeling and analysis.

**Acceptance Criteria:**

* Fabric Pipelines and Spark Structured Streaming notebooks are developed to efficiently process data from Bronze to Silver.
* Raw data (e.g., topic, value, timestamp) is parsed and separated into structured columns (e.g., site, line, sensor, value, timestamp).
* Identified data inconsistencies or errors are cleansed (e.g., handling encrypted values, nulls, outliers).
* Enrichment with relevant metadata (e.g., lookup tables for sensor definitions) and partitioning (e.g., by date or site) for performance are applied.
* The Silver Lakehouse contains standardized and enriched data, forming the initial modeling layer.

**Tasks for US 2.1:**

#### Task 2.1.1: Design Silver Lakehouse Schema

*Description:* Define a robust and flexible schema for the Silver Lakehouse tables, incorporating cleansed, standardized, and enriched columns derived from the raw Bronze data, including appropriate data types and primary/foreign key relationships.

#### Task 2.1.2: Develop Spark Structured Streaming Notebooks for Silver Transformation

*Description:* Write PySpark notebooks within Fabric to perform structured streaming transformations. Implement parsing logic for the topic column to extract plant, line, and sensor/motor details, and add derived columns like date/time components. Optimize Spark configurations (e.g., executor memory, core counts) for efficient processing of high-volume data.

#### Task 2.1.3: Create Fabric Pipelines for Silver Layer Orchestration

*Description:* Build Fabric Pipelines to orchestrate the execution of the Spark notebooks. Configure scheduling for continuous or batch processing (e.g., daily) from the Bronze Lakehouse to the Silver Lakehouse, ensuring robust error handling and retry mechanisms.

#### Task 2.1.4: Implement Data Quality Checks for Silver Layer

*Description:* Introduce automated data quality checks (e.g., using Great Expectations or custom PySpark logic) post-transformation to ensure the accuracy, completeness, and consistency of data in the Silver Lakehouse before it's consumed by downstream layers.

### User Story 2.2: Create Modeled Datasets in Gold Workspace

**Description:** As a Data Engineer, I want to create advanced data models in the Gold Workspace (Data Warehouse) using cleansed and enriched data from the Silver Lakehouse, so that refined business insights and analytics can be derived, and data is ready for ML operations and Power BI reporting.

**Acceptance Criteria:**

* A dedicated Gold Workspace and Data Warehouse are established in Fabric.
* Advanced data modeling (e.g., dimensional modeling, star schemas, aggregations) is applied to data from the Silver Lakehouse to support specific business needs.
* The Gold Data Warehouse is optimized with Delta Lake and VertiParquet for high-performance analytics via Direct Lake.
* Data is accessible from the Gold layer for ML operations and Power BI semantic models.
* The Gold layer can successfully combine various data types (e.g., OT data with ENTERPSYS\_A data) for comprehensive reporting.

**Tasks for US 2.2:**

#### Task 2.2.1: Design Gold Data Warehouse Model

*Description:* Define the logical and physical data model for the Gold Data Warehouse, including fact tables, dimension tables, and necessary aggregations to support various analytical and ML use cases. Prioritize star schema designs for Power BI reporting optimization.

#### Task 2.2.2: Develop Gold Layer Transformation Logic

*Description:* Create Fabric Notebooks or Data Pipelines to transform and load data from the Silver Lakehouse into the Gold Data Warehouse, applying advanced modeling techniques such as slowly changing dimensions, aggregation, and pre-calculated metrics (e.g., OEE calculations for plant operations).

#### Task 2.2.3: Optimize Gold Data Warehouse for Performance

*Description:* Configure and optimize the Gold Data Warehouse tables using Delta Lake format with V-Order (VertiParquet) optimization to ensure highly performant query execution for Direct Lake connections from Power BI. Implement regular compaction and Z-ordering where applicable.

#### Task 2.2.4: Integrate ENTERPSYS\_A Data into Gold Layer

*Description:* Explore and implement Fabric Shortcuts to bring ENTERPSYS\_A business data into the Fabric environment. Develop data pipelines to join or relate this ENTERPSYS\_A data with the OT data in the Gold layer, enabling combined analytics such as assigning dollar amounts to plant operations based on sensory data.

--------------------------------------------------------------------------------

## Feature 3: ML Model Development and Operationalization (MLOps)

• **Description:** Establish a comprehensive MLOps framework within Fabric to manage the end-to-end lifecycle of machine learning models, from development and training in the cloud to secure and resilient deployment and monitoring on-premises at the plant floor.

### User Story 3.1: Train and Manage ML Models in Fabric with MLFlow

**Description:** As a Data Scientist, I want to develop and train ML models in Fabric, leveraging integrated MLFlow for experiment tracking, version management, and model registration, so that model development is organized, reproducible, and ready for deployment.

**Acceptance Criteria:**

* ML models are developed and trained using Fabric notebooks and ML capabilities.
* MLFlow is configured and used to track experiments, parameters, and metrics for each model run.
* Model versions are managed and registered in a central model registry (e.g., MLFlow Model Registry).
* The trained model is containerized (e.g., Docker image) for deployment to the edge.
* Historical OT data from Fabric (e.g., from the Gold Lakehouse) is incorporated into model training once sufficient data is available.

**Tasks for US 3.1:**

#### Task 3.1.1: Set up Fabric ML Workspace and MLFlow

*Description:* Configure the ML development environment within the Fabric workspace, ensuring MLFlow is integrated for tracking experiments, managing model artifacts, and supporting the model lifecycle.

#### Task 3.1.2: Develop ML Model Training Notebooks

*Description:* Create PySpark or Python notebooks in Fabric for training the initial ML model (e.g., for flake moisture prediction), leveraging prepared data from the Silver or Gold Lakehouse. Ensure efficient data loading and processing for training.

#### Task 3.1.3: Implement MLFlow Tracking for Experiments

*Description:* Integrate MLFlow APIs within the training notebooks to automatically log model parameters, evaluation metrics (e.g., RMSE for regression), and artifacts (e.g., trained model files). Enable robust experiment tracking for reproducibility and comparison.

#### Task 3.1.4: Containerize Trained ML Models

*Description:* Develop a standardized process to containerize the trained ML models into Docker images, including all necessary dependencies and inference code, preparing them for deployment to the on-premises edge environment.

### User Story 3.2: Deploy and Monitor ML Models On-Premises at the Edge

**Description:** As an Operations Engineer, I want to securely deploy containerized ML models from the cloud to on-premises Kubernetes clusters via IoT Edge, and monitor their performance continuously, so that plant operations can leverage real-time ML inference even during disconnected periods.

**Acceptance Criteria:**

* Containerized ML models are deployed to on-premises Kubernetes clusters via IoT Edge.
* ML model inference runs locally on the plant floor, receiving inputs from and publishing outputs to the on-premises MQTT broker.
* IoT Edge compute is managed and monitored through Arc, providing cloud-based control over edge deployments.
* Model inputs and outputs are stored in a local historian database for offline capability and digital twin analysis.
* Telemetry and ML model performance (e.g., data drift, prediction accuracy) are monitored via Log Analytics and Power BI dashboards.
* A clear strategy for retraining and re-deployment of models based on performance degradation or process changes is defined and supported by automated pipelines.

**Tasks for US 3.2:**

#### Task 3.2.1: Configure IoT Edge for On-Premises Deployment

*Description:* Install and configure IoT Edge runtime on the on-premises Kubernetes clusters (or target VMs) at the plant, ensuring secure communication with IoT Hub for module deployment and management.

#### Task 3.2.2: Integrate Arc for Edge Management

*Description:* Onboard on-premises Kubernetes clusters to Arc to enable centralized management and monitoring of IoT Edge compute and resources from Azure, leveraging Arc's capabilities for GitOps deployments and policy enforcement.

#### Task 3.2.3: Implement Container Registry for Model Storage

*Description:* Utilize Container Registry or a compatible registry (e.g., GitHub Container Registry if aligning with GitHub Enterprise) to securely store the containerized ML models, making them available for IoT Edge deployment.

#### Task 3.2.4: Develop ML Model Deployment Pipeline to Edge

*Description:* Create an automated deployment pipeline (e.g., using Azure DevOps or GitHub Actions) to push trained ML models from the model registry to the Container Registry and then deploy them to IoT Edge on the on-premises Kubernetes cluster. Implement rolling updates and canary deployments where appropriate.

#### Task 3.2.5: Configure Local Historian for ML Data Storage

*Description:* Ensure the on-premise historian database (e.g., Pi Historian, transitioning to SQL Server) is configured to reliably store ML model inputs, predicted outputs, and relevant operational data. This will serve as a local digital twin and source for future offline analysis or retraining.

#### Task 3.2.6: Establish Model Monitoring and Alerting for Edge Deployments

*Description:* Implement mechanisms to collect telemetry data (e.g., inference requests, model latency, prediction results) from edge-deployed ML models and forward it to Log Analytics. Configure Power BI dashboards for real-time model performance visualization and alerts (potentially via Data Activator for plant operators) to detect data drift or performance degradation.

--------------------------------------------------------------------------------

## Feature 4: Business Data Integration and Advanced Reporting

• **Description:** Enable comprehensive business intelligence and reporting capabilities by integrating operational data with financial and other business data, leveraging Fabric's reporting features and Power BI's visualization tools.

### User Story 4.1: Integrate ENTERPSYS\_A Data for Unified Analytics

**Description:** As a BI Developer, I want to securely integrate ENTERPSYS\_A data from the ACME tenant into Fabric using Dataverse shortcuts, so that it can be married with OT data for new levels of reporting, such as assigning dollar values to plant operations.

**Acceptance Criteria:**

* Dataverse shortcuts are successfully established for ENTERPSYS\_A data access in Fabric.
* ENTERPSYS\_A data is accessible and integrated within the Fabric Gold Workspace, reflecting necessary business information.
* The combined OT and ENTERPSYS\_A data enables new reporting scenarios, including cost accounting, operational cost analysis, and executive-level KPIs with dollar amounts.

**Tasks for US 4.1:**

#### Task 4.1.1: Configure Dataverse Shortcut to ENTERPSYS\_A

*Description:* Set up the Dataverse shortcut from the Acme-Sub Fabric tenant to the ENTERPSYS\_A instance residing in the ACME tenant, ensuring proper authentication and addressing cross-tenancy considerations for secure data access.

#### Task 4.1.2: Define ENTERPSYS\_A Data Integration Strategy

*Description:* Identify key ENTERPSYS\_A entities and attributes (e.g., labor costs, material costs, production volumes) required for marriage with OT data. Design the data integration logic to bring this data into the Fabric Gold layer (Lakehouse or Data Warehouse), considering data granularity and update frequency.

#### Task 4.1.3: Develop Data Engineering for Combined Data

*Description:* Create Fabric notebooks or pipelines to perform necessary joins, aggregations, and transformations to integrate ENTERPSYS\_A data with OT data in the Gold Lakehouse/Warehouse. This includes logic for assigning financial values to operational metrics for comprehensive analytics.

### User Story 4.2: Develop and Deploy Power BI Reports Across Environments

**Description:** As a Power BI Developer, I want to author, test, and deploy Power BI reports and dashboards using a structured deployment pipeline across Dev, Test, and Production workspaces, so that consistent and high-performance analytics are available to various internal stakeholders.

**Acceptance Criteria:**

* Dedicated Dev, Test, and Production workspaces are set up for Power BI asset deployment within Fabric.
* Power BI semantic models are directly connected to Gold Workspace datasets via Direct Lake for optimal performance.
* Deployment pipelines are configured to automate the promotion of semantic models, dashboards, and reports through environments.
* Power BI apps are published from production workspaces to end-users (Acme-Sub and ACME internal users) with appropriate access controls.
* Executive-level KPIs and operational dashboards are functional and provide actionable insights.

**Tasks for US 4.2:**

#### Task 4.2.1: Establish Power BI Workspace Hierarchy

*Description:* Create Dev, Test, and Production Power BI workspaces within Fabric to support a structured deployment pipeline for Power BI assets, ensuring clear separation of environments.

#### Task 4.2.2: Configure Direct Lake Connectivity for Semantic Models

*Description:* Configure Power BI semantic models in each workspace to use Direct Lake mode for optimal performance when querying the Gold Workspace Data Warehouse. This avoids data movement and leverages the VertiParquet engine directly on OneLake.

#### Task 4.2.3: Implement Power BI Deployment Pipelines

*Description:* Set up and configure Power BI deployment pipelines within Fabric (or Azure DevOps pipelines for more complex scenarios) to automate the promotion of semantic models, reports, and dashboards from Dev to Test, and then to Production. Incorporate pull requests and review processes for quality assurance.

#### Task 4.2.4: Design and Develop Core Power BI Reports/Dashboards

*Description:* Create initial Power BI reports and dashboards, focusing on critical operational KPIs, ML model performance monitoring (e.g., model output vs. actual machine settings), and combined OT/ENTERPSYS\_A insights (e.g., cost per unit of production). Design for various users, from plant operators to executives.

#### Task 4.2.5: Publish Power BI Apps and Manage User Access

*Description:* Publish Power BI apps from the Production workspace, bundling reports and dashboards for easy consumption. Define granular role-based access control (RBAC) policies within Power BI apps to ensure only authorized internal users (Acme-Sub and ACME employees) can access relevant reports based on their team and function.

--------------------------------------------------------------------------------

## Feature 5: Enterprise-Grade Security and Governance

• **Description:** Implement comprehensive security measures and governance policies across the entire Fabric and on-premises infrastructure, ensuring data protection, access control, compliance, and unified monitoring to minimize security risks.

### User Story 5.1: Centralize Identity and Access Management

**Description:** As a Security Administrator, I want to centralize identity and access management using Entra ID with conditional access and MFA, and govern guest accounts via cross-tenant synchronization, so that all users accessing Fabric and related resources are authenticated and authorized securely.

**Acceptance Criteria:**

* Entra ID is established as the central identity provider for all Fabric and resources.
* Conditional Access policies are implemented to enforce strong authentication (MFA) and device posture checks for all access to Fabric and sensitive resources.
* Cross-tenant synchronization is effectively used to manage ACME identities as guest accounts in the Acme-Sub tenant for collaborative access.
* Periodic access reviews are configured and conducted for guest accounts and privileged roles to ensure least privilege.
* Privileged Identity Management (PIM) is configured for just-in-time access to critical administrative roles.

**Tasks for US 5.1:**

#### Task 5.1.1: Verify Entra ID Sync with On-Premises Active Directory

*Description:* Confirm that Acme-Sub's on-premises Active Directory is correctly synchronizing identities to the Acme-Sub Entra ID tenant, and that the security posture for both Acme-Sub and ACME ADs adheres to ACME's centralized security policies.

#### Task 5.1.2: Configure Conditional Access Policies for Fabric and Resources

*Description:* Define and implement Conditional Access policies in Entra ID for all access to Fabric workspaces, Power BI apps, and critical resources (e.g., storage accounts, VMs). Enforce MFA for all users and consider device compliance requirements for specific access scenarios.

#### Task 5.1.3: Review and Optimize Cross-Tenant Synchronization for Guest Accounts

*Description:* Evaluate the existing cross-tenant synchronization setup (Acme-Sub to ACME for guest accounts) to ensure efficient provisioning and de-provisioning of external/vendor identities. Confirm that trusted MFA settings from the home tenant are honored.

#### Task 5.1.4: Implement Access Reviews for Guest Accounts and Privileged Roles

*Description:* Set up periodic access reviews in Entra ID for all guest accounts and users assigned to high-privileged roles (e.g., Fabric Administrator, Owner/Contributor on critical resource groups) to ensure access is regularly validated and revoked when no longer needed.

#### Task 5.1.5: Configure Privileged Identity Management (PIM) for Administrative Roles

*Description:* Apply PIM to critical administrative roles within CloudProviderA (e.g., Subscription Owner, Contributor on resource groups containing HiveMQ VM) and Fabric (e.g., Fabric Administrator). Configure just-in-time access, requiring activation and MFA for elevated permissions, with time-bound access (e.g., 8 hours).

### User Story 5.2: Implement Unified Security Monitoring and Logging

**Description:** As a Security Operations Analyst, I want to centralize security monitoring using Sentinel with consolidated instances across tenants and ensure comprehensive logging and auditing of all system activities, so that potential threats and compliance issues can be detected and responded to effectively.

**Acceptance Criteria:**

* A consolidated Sentinel instance (preferably in the ACME tenant) is used for multi-tenant monitoring, optimizing cost and providing a single pane of glass.
* Sign-in logs and activity logs from both Acme-Sub and ACME Entra ID tenants are forwarded to a central Log Analytics workspace ingested by Sentinel.
* Key security alerts and playbooks are configured in Sentinel for critical events (e.g., unauthorized access, VM compromise, data exfiltration attempts).
* Comprehensive logging and auditing are enabled for all Fabric resources, services (ADLS, Event Hub, IoT Edge), and on-premises components with long-term retention policies.
* Defender for Cloud is utilized for continuous resource posture management and threat protection.

**Tasks for US 5.2:**

#### Task 5.2.1: Consolidate Sentinel Instances for Multi-Tenant Monitoring

*Description:* Collaborate with ACME and Acme-Sub security teams to consolidate existing Sentinel instances into a single, centralized Log Analytics workspace, ideally in the ACME tenant, to provide unified security monitoring across both tenants and reduce duplicate ingestion costs.

#### Task 5.2.2: Configure Comprehensive Log Forwarding to Central Log Analytics

*Description:* Ensure all relevant sign-in logs, audit logs, activity logs from Entra ID, resource logs (from ADLS, Event Hub, VMs, Fabric), and security logs from on-premises components are securely forwarded to the centralized Log Analytics workspace. Verify that Policy enforces logging requirements for newly deployed resources.

#### Task 5.2.3: Define and Implement Key Security Alerts in Sentinel

*Description:* Work with the Security Operations Center (SOC) team to define and configure critical security alerts and incident response playbooks within Sentinel, tailored to detect suspicious activities related to Fabric data access, ML model deployments, cross-tenant interactions, and on-premises compromises (e.g., using Defender for Identity signals).

#### Task 5.2.4: Validate Defender for Cloud Implementation

*Description:* Confirm that Defender for Cloud is fully enabled and configured to provide continuous security posture management, vulnerability assessments, and threat protection across all resources within the Acme-Sub subscription, including servers, storage accounts, and Kubernetes clusters.

#### Task 5.2.5: Establish Auditing and Compliance Monitoring Practices

*Description:* Implement a regular schedule for security auditing and compliance monitoring activities. This includes reviewing access logs, security alerts, and system configurations against industry standards (e.g., ISA/IEC 62443 for OT security) and internal policies to identify gaps and ensure continuous compliance.

### User Story 5.3: Implement Secure Credential and Secret Management

**Description:** As a DevOps Engineer, I want to manage all credentials, secrets, and certificates securely using Key Vault with defined rotation policies, so that sensitive information is protected and automated processes can access resources without exposing credentials.

**Acceptance Criteria:**

* Key Vault is the designated secure store for all credentials, secrets, and certificates required for service integrations.
* A clear and automated rotation policy is defined and implemented for all secrets stored in Key Vault.
* Managed Identities are utilized wherever possible to eliminate the need for manual credential management for services.
* Access to Key Vault is strictly controlled using Entra ID and the principle of least privilege.

**Tasks for US 5.3:**

#### Task 5.3.1: Provision and Configure Key Vault

*Description:* Create an Key Vault instance (or confirm the existing one) in the Acme-Sub tenant. Define granular access policies (e.g., using Entra ID groups and Managed Identities) to ensure only authorized applications and users can retrieve or manage secrets, following the principle of least privilege.

#### Task 5.3.2: Migrate Existing Credentials to Key Vault

*Description:* Conduct an audit to identify all existing hardcoded or manually managed credentials (e.g., for SFTP, CloudDB\_A, VendorR APIs) and securely migrate them into Key Vault. Update applications and scripts to retrieve these secrets from Key Vault rather than storing them directly.

#### Task 5.3.3: Define and Implement Automated Secret Rotation Policy

*Description:* Establish a comprehensive policy for automated rotation of secrets within Key Vault (e.g., for database connection strings, API keys, SFTP passwords). Implement automated rotation mechanisms using Functions or Automation where feasible, considering dependencies and minimizing service disruption.

#### Task 5.3.4: Implement Managed Identities for Services

*Description:* Configure services (e.g., Data Factory, Fabric workspaces, Event Streams, VMs hosting MQTT broker) to use Managed Identities for authentication to other resources (e.g., ADLS, Key Vault). This eliminates the need to store and rotate credentials for these services, enhancing security and manageability.

--------------------------------------------------------------------------------

## Feature 6: Platform Management and Optimization

• **Description:** Optimize the Fabric environment for performance, cost efficiency, and ease of management, ensuring the platform is resilient, troubleshoot-able, and aligns with best practices for a lean team.

### User Story 6.1: Optimize Fabric Capacity Utilization

**Description:** As a Fabric Administrator, I want reliable visibility into Fabric capacity metrics and performance, so that I can optimize resource utilization, troubleshoot issues like idle time and random notebook failures, and ensure cost efficiency.

**Acceptance Criteria:**

* The Fabric capacity metrics dashboard is functional and accurately displays usage and trends.
* Root causes for random notebook failures and excessive idle time are identified and addressed, leading to improved reliability.
* Capacity scaling (e.g., from F2 to F64) is planned and executed based on workload demands and cost analysis.
* Capacity is managed efficiently for both development (F2) and production (F64) workloads to balance performance and cost.

**Tasks for US 6.1:**

#### Task 6.1.1: Develop Comprehensive Fabric Capacity Scaling Plan

*Description:* Create a detailed plan for scaling Fabric capacity (e.g., current F2 for dev to F64 for production) based on current and projected data volumes (e.g., 2.7 billion rows/month per line), processing needs, and user concurrency. Include a cost analysis to justify upgrades and potential cost-saving measures during off-peak hours.

#### Task 6.1.2: Familiarize with Fabric PowerShell Modules for Automation

*Description:* Review and experiment with the Fabric PowerShell modules (e.g., PowerBI Mgmt) to extract tenant-level settings, automate workspace management (e.g., creating/deleting workspaces, managing access), and programmatically troubleshoot issues for efficient administration by the small team.

### User Story 6.2: Standardize Development and Deployment Workflows

**Description:** As a Developer, I want to use standardized source control, CI/CD, and MLOps pipeline for all Fabric artifacts (notebooks, Power BI reports, data pipelines), so that development is collaborative, consistent, and deployments are automated and reliable.

**Acceptance Criteria:**

* GitHub Enterprise is the centralized source control repository for all Fabric development artifacts.
* A clear branching strategy (e.g., dev, QA, main) is implemented with mandatory pull request reviews.
* Automated CI/CD pipelines are established for deploying Fabric notebooks, pipelines, and Power BI assets across environments (Dev, Test, Prod).
* MLOps practices are integrated into the pipeline for automated ML model lifecycle management.

**Tasks for US 6.2:**

#### Task 6.2.1: Centralize Fabric Artifacts in GitHub Enterprise

*Description:* Migrate all Fabric development artifacts (e.g., PySpark notebooks, Fabric Data Pipeline definitions, Power BI report .pbix files, semantic model definitions) to GitHub Enterprise for centralized source control. Implement a folder structure that aligns with workspaces and medallion layers. (CD) of Fabric notebooks and data pipelines. This should include automated linting, basic testing, and deployment to respective Dev, Test, and Prod workspaces upon successful build.

#### Task 6.2.4: Automate Power BI Deployment with Fabric Deployment Pipelines

*Description:* Fully integrate Power BI artifacts (semantic models, reports, dashboards) with Fabric's native deployment pipelines, enabling automated promotion across Dev, Test, and Prod Power BI workspaces. Ensure these pipelines are linked to GitHub Enterprise for source control and support rollbacks if needed.

(e.g., On-Premises, Cloud, Fabric Workspaces, External Services) and responsibilities of different teams (e.g., OT, Data Engineering, Data Science, Security, IT) into clear horizontal or vertical lanes. This will help visualize data flow, points of integration, and responsibilities, making it easier to track progress, identify bottlenecks, and ensure everyone understands their role in the overall system. We can then overlay the backlog items onto this diagram to show where each feature, user story, and task contributes to the overall architecture.