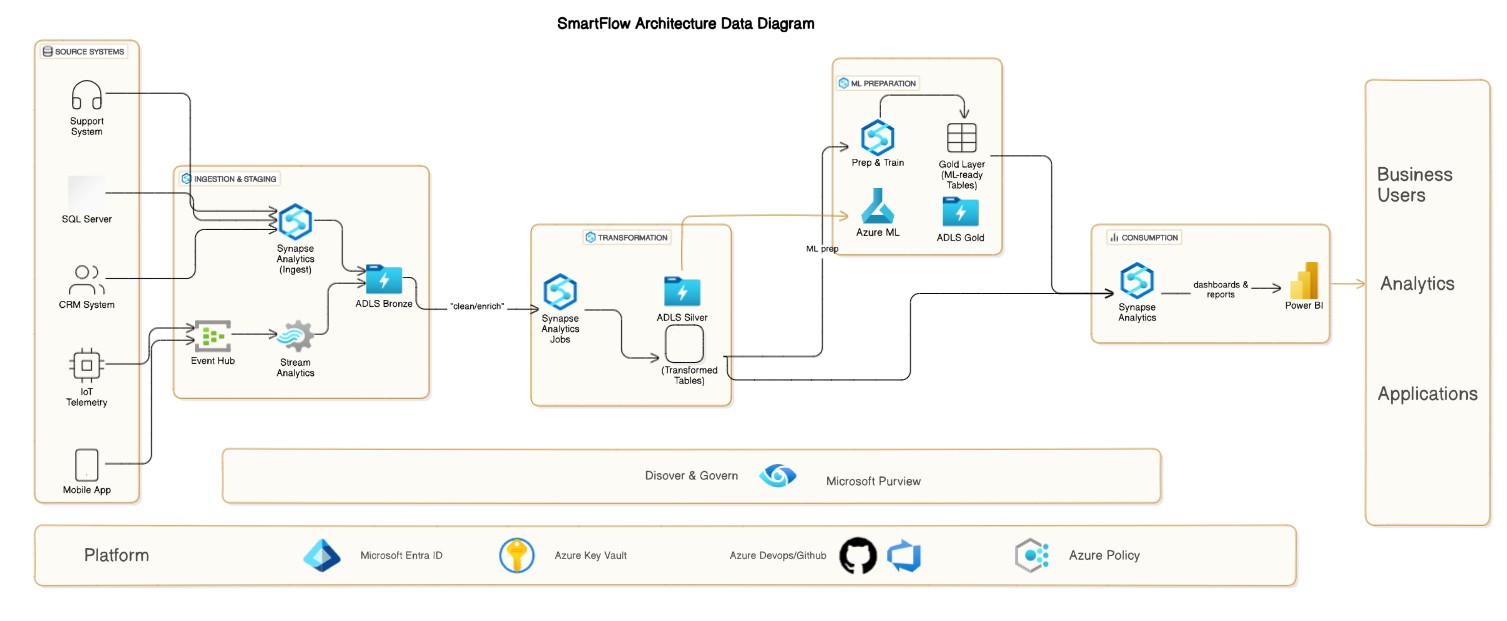
**SmartFlow Data Architecture,**

**Modeling, Governance, and ML/AI Serving Documentation**

1. Architecture Diagram:



**1. Data Flow & Medallion Architecture**

**1.1 Overview**

SmartFlow ingests data from heterogeneous source systems—including CRM, SQL Server OLTP, IoT telemetry devices, mobile applications, and support systems—via both batch and streaming pipelines. All raw data lands in a **Bronze** zone for durability and replay, progresses to **Silver** where it is cleansed and conformed, and culminates in **Gold** tables optimized for analytics, reporting, and ML workloads. Microsoft Purview provides end‑to‑end discovery and lineage, while platform services such as Azure Key Vault and Azure Policy enforce security and guardrails.

**1.2 Batch Ingestion Path**

1. **Source Extraction**: Nightly or hourly extracts from SQL Server and CRM via Synapse Analytics pipelines ( i.e Copy Data).
2. **Landing (Bronze)**: Data is written raw into an Azure Data Lake Storage (ADLS) container in a bronze/ folder, partitioned by ingestion date (YYYY/MM/DD).
3. **Transformation to Silver**: Azure Synapse Spark jobs run ELT notebooks to:
   * Deduplicate and normalize business keys
   * Parse, type‑cast, and enrich with reference lookups
   * Write cleaned tables to silver/ folder, partitioned by business date and region
4. **Promotion to Gold**: Scheduled Synapse SQL pipelines aggregate and join multiple silver tables into curated gold models.

**1.3 Streaming Ingestion Path**

1. **Event Hub → Stream Analytics**: IoT and mobile events stream into Azure Event Hubs. Azure Stream Analytics jobs perform lightweight filtering and windowed aggregations, writing output to Bronze.
2. **Bronze → Silver**: Synapse Streaming Spark jobs continuously read bronze folders (or via Azure Synapse pipelines’ Delta Autoloader) to:
   * Enrich with master data
   * Apply business rules (e.g., anomaly detection)
   * Write to silver/telemetry/ as micro‑batch Delta tables
3. **Real‑Time Gold**: Low‑latency Synapse SQL on-demand queries produce real‑time views in Gold, backed by materialized views for dashboards.

**1.4 Medallion Layers Summary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Layer** | **Purpose** | **Storage Location** | **Partitioning** | **Consumers** |
| Bronze | Raw, immutable snapshots | adls://.../bronze/ | ingest\_date=YYYY/MM/DD | Data engineers |
| Silver | Cleaned, conformed, enriched datasets | adls://.../silver/ | business\_date=YYYY-MM-DD/region=XX | BI developers, data scientists |
| Gold | Aggregated, highly curated analytics tables | adls://.../gold/ | Varies by subject area (date, custID) | Power BI, ML models, APIs |

**2. Data Modeling Document**

**2.1 Key Models for Silver & Gold**

* **Silver**: 1) **Customer\_Events\_Silver**; 2) **Transaction\_Facts\_Silver**; 3) **Device\_Telemetry\_Silver**
* **Gold**: 1) **Customer360\_View**; 2) **Daily\_Revenue**; 3) **AnomalyAlerts\_View**

**2.2 Entity Definitions**

**2.2.1 Customer (Silver)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Type | Description | Partition Key | Sort/Cluster |
| customer\_id | STRING | Surrogate key |  |  |
| source\_customer\_id | STRING | Original ID from CRM |  |  |
| name | STRING | Full name |  |  |
| email | STRING | Email address |  |  |
| signup\_date | DATE | Date of first registration | business\_date |  |
| region | STRING | Geographic region |  |  |

**Partitioning**: by signup\_date (YYYY-MM) to support time‑based lookups.  
**Optimization**: Use Z‑ordering on region for skewed regional queries.

**2.2.2 Transaction (Silver)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Type | Description | Partition Key | Sort/Cluster |
| transaction\_id | STRING | GUID |  |  |
| customer\_id | STRING | FK to Customer |  |  |
| product\_id | STRING | FK to product catalog |  |  |
| amount | DECIMAL(10,2) | Transaction value |  |  |
| transaction\_date | TIMESTAMP | Event timestamp | business\_date |  |
| payment\_method | STRING | E.g., CreditCard, PayPal |  |  |

**Partitioning**: by business\_date (daily) for efficient retention.  
**Optimization**: Cluster by customer\_id and Z‑order on transaction\_date.

**2.2.3 Device Telemetry (Silver)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Type | Description | Partition Key | Sort/Cluster |
| device\_id | STRING | Unique device identifier |  |  |
| event\_time | TIMESTAMP | Telemetry timestamp | event\_date |  |
| metric\_type | STRING | E.g., temperature, vibration |  |  |
| metric\_value | DOUBLE | Measured value |  |  |
| region | STRING | Physical deployment region |  |  |

**Partitioning**: by event\_date (daily) to support time series retention.  
**Optimization**: Use data compaction and Delta auto‑optimize enabled.

**2.3 Gold Layer Models & Optimization**

**2.3.1 Customer360\_View (Gold)**

* Joins Customer (silver) + aggregate metrics from Transaction and Telemetry
* Partition on region, year\_month (YYYY-MM)
* Indexing: create materialized view in Synapse providing precomputed KPIs per customer-month.

**2.3.2 Daily\_Revenue (Gold)**

* Pre‑aggregated daily revenue by product and region
* Partition by report\_date
* Use Synapse dedicated SQL pool clustered columnstore tables for large fact scans.

**2.3.3 AnomalyAlerts\_View (Gold)**

* Real‑time view combining latest telemetry anomalies with business rules
* Maintain as an external table over Delta streaming sink with granularity of 5‑minute windows

**3. Data Governance Strategy**

**3.1 Data Quality & Lineage**

* **Quality Rules**: Implement Delta Expectations in Silver tables—uniqueness, not-null, range checks.
* **Automated Validation**: Azure Data Factory Data Flow Assertions, Pyspark Great Expectations integration.
* **Lineage Tracking**: Microsoft Purview scans ADLS and Synapse metadata to generate a graph of lineage from raw ingestion to Gold datasets.

**3.2 Metadata Management**

* **Glossary & Catalog**: Define business terms (e.g., "Customer Lifetime Value") in Purview.
* **Tagging**: Tag all tables/columns with classification labels (PII, Confidential, Public).

**3.3 Sensitive Data Controls**

* **Discovery & Classification**: Purview automated scanners to detect PII in Bronze and Silver.
* **Access Policies**: Azure RBAC combined with Synapse workspace roles to restrict access to PII classified tables.
* **Encryption**: All data encrypted at rest (ADLS SSE with Microsoft-managed keys or BYOK in Key Vault) and in transit (TLS).

**3.4 Monitoring & Alerting**

* **Pipeline Health**: Azure Monitor alerts on ADF/Synapse pipeline failures, latency SLAs breaches.
* **Data Freshness**: Custom Log Analytics queries to detect missing partitions or late arrivals, with email or Teams notifications.
* **Quality Metrics**: Push Delta Expectation results to Azure Metrics and trigger alerts when thresholds exceeded.

**4. ML/AI Data Serving Strategy**

**4.1 Data Preparation for Modeling**

* **Feature Generation**: Use Spark on Silver to compute rolling metrics, aggregations, and one‑hot encodings.
* **Gold Layer for ML**: Create a dedicated set of tables in Gold (e.g., ml\_features.gold\_customer\_features) stored in Delta, partitioned by snapshot\_date.
* **Unstructured Data Handling**: Store raw documents or logs in bronze/unstructured/, process via synapse notebooks into embeddings stored in Delta or Cosmos DB.

**4.2 Feature Store Requirements**

* **Centralized Feature Store**: Deploy Azure ML Feature Store preview to register, share, and serve features.
* **Online & Offline Stores**:
  + **Offline**: Delta tables in ADLS for batch training.
  + **Online**: Azure Cosmos DB or Redis for low-latency feature lookups in production inference.
* **Versioning**: Feature version metadata maintained in the store to ensure reproducibility.

**4.3 Serving to Models**

* **Batch Training**: Orchestrate Azure ML pipelines that read offline features and unstructured embeddings, train models, and log metrics back to MLflow.
* **Real‑Time Inference**:
  + Functions in Azure Functions or FastAPI deployed to AKS call online feature store and serve predictions.
  + Data freshness managed via incremental streaming Delta writes and Event Grid notifications.

**4.4 Structured & Unstructured Integration**

* **Structured**: Ingest via Silver → transform to Gold features → register in Feature Store.
* **Unstructured**: Use Databricks MLFlow pipelines to extract text embeddings or image features, store embedding vectors in a Delta table.
* **Model Training**: Combine structured features join with embeddings on customer\_id or device\_id for end‑to‑end training.