A project that ties together **Data Pipelines, ETL, Power BI, Databricks, and C#**, and host it in **Azure**. Since you'll be building with **Claude's multi-agent coding setup**, the PRD needs to be structured with enough clarity that multiple AI agents (or devs) can implement it piece by piece without ambiguity.

Use Case 1 (Connected Products / Customer 360) — because it naturally integrates **Sales + Customer + IoT telemetry** and is very relatable for Power BI dashboards.

Here's a structured PRD for you:

📄 **Product Requirements Document (PRD)**

Project Name: Azure Data Analytics Platform (ETL + Power BI + Databricks + C# APIs)
Author: Navin
Date: 2025-08-18
Version: v1.0

---

1. Problem Statement

The company collects raw data from multiple operational systems (CRM, ERP, and IoT devices). The data is siloed and unstructured, making it difficult for business users to generate insights. The goal is to build a **cloud-hosted data analytics solution** on Azure that ingests raw data, transforms it via ETL pipelines in Databricks, and serves clean data to Power BI dashboards. A **C# API layer** will provide controlled access to curated datasets.

---

2. Goals & Objectives

1. **Data Ingestion:** Automate ingestion from CSV, JSON (IoT), and SQL databases into Azure Data Lake.
2. **ETL with Databricks:** Cleanse, transform, and standardize the ingested data for analytics.
3. **Data Modeling:** Store curated datasets in Delta Lake / Synapse Analytics for downstream consumption.
4. **Visualization:** Build Power BI dashboards connected to curated datasets.

5. **APIs in C#:** Expose curated datasets via REST APIs for external applications.
6. **Scalability & Cost Optimization:** Ensure pipelines can scale with minimal overhead.

---

3. In-Scope Deliverables

- **Data Pipelines**
    - Batch ingestion from Azure Blob Storage (CSV, JSON).
    - Streaming ingestion from IoT Hub (optional stretch goal).
    - Data transformations in Databricks using PySpark/SQL.
- **ETL Jobs**
    - Data quality checks (nulls, duplicates, invalid values).
    - Data standardization (date formats, currency, IDs).
    - Dimensional modeling (star schema for reporting).
- **Storage**
    - Bronze (raw), Silver (clean), Gold (curated) layers in Delta Lake.
    - Optionally, publish to Azure Synapse Analytics for Power BI.
- **Analytics**
    - Power BI dashboards: Sales, Customer KPIs, IoT Device Health.
- **API Layer (C# / .NET 8 Web API)**
    - Authentication: Azure AD.
    - Endpoints:
        - /api/customers/{id} → Returns curated customer profile.
        - /api/sales/daily → Returns daily sales aggregates.
        - /api/devices/alerts → Returns IoT device alerts.

Each deliverable should be tied to a specific business need. Here's a breakdown with use cases for each component:

Data Pipelines

- **Batch Ingestion: Marketing Campaign Analysis.** A marketing team wants to analyze the success of recent email campaigns. The raw data (CSV files with customer clicks, opens, and unsubscribes) is dropped into Azure Blob Storage daily. The pipeline must ingest these files, enriching the data with a **campaign ID** and **load timestamp** before it's sent to the Bronze layer.
- **Streaming Ingestion: Real-Time Inventory Monitoring.** A retail company uses IoT sensors in their warehouses. Each sensor sends a JSON event to Azure IoT Hub whenever an item is picked or moved. The streaming pipeline must ingest this data to update inventory counts in near real-time, allowing store managers to see accurate stock levels and prevent stockouts.
- **ETL with Databricks: Customer Segmentation.** The sales team wants to identify high-value customers. The ETL job must join data from the **CRM system** (customer demographics) with **ERP data** (purchase history). It will then apply transformations, such as calculating **Total Lifetime Value** and **Average Order Value**, before storing the final, enriched customer data in the Gold layer.

API Layer (C# / .NET 8 Web API)

- **/api/customers/{id}**: **Customer 360 View.** A customer service representative (CSR) using an internal support application needs to quickly access a complete view of a customer's profile, including their recent orders, support tickets, and contact information. This endpoint allows the application to retrieve all relevant data for a given customer ID to display a single, unified view.
- **/api/sales/daily**: **External Partner Reporting.** A third-party logistics (3PL) partner needs to access daily sales aggregates to forecast shipping volumes. This endpoint provides a secure, controlled way for the partner's system to retrieve high-level sales data without needing direct access to the company's internal databases.
- **/api/devices/alerts**: **Maintenance Dashboard.** A maintenance team uses a custom application to monitor IoT devices. This endpoint is called every 5 minutes to fetch a list of any devices that have sent a critical alert (e.g., temperature threshold exceeded, low battery). This allows the team to be proactive in addressing potential issues before they become major problems.

---

4. Out-of-Scope

- Advanced ML models (future phase).
- Real-time stream analytics (beyond IoT ingestion).
- Custom Power BI visuals.

---

5. Functional Requirements

Data Ingestion

- **FR1:** System shall ingest CSV files from Azure Blob on a daily schedule.
- **FR2:** System shall ingest IoT JSON events from Azure IoT Hub in near real-time.
- **FR3:** System shall log all ingestion metadata (file name, load time, record count).

ETL Processing

- **FR4:** System shall deduplicate data based on primary keys.
- **FR5:** System shall handle schema drift (new columns in source).
- **FR6:** System shall persist data in Bronze → Silver → Gold Delta tables.

Data Storage & Modeling

- **FR7:** System shall maintain star schema for Sales & Customer analytics.
- **FR8:** Gold layer tables shall be queryable from Power BI & APIs.

API Layer

- **FR9:** System shall authenticate API users via Azure AD OAuth2.
- **FR10:** System shall provide REST endpoints for Sales, Customers, and IoT Alerts.

Reporting

- **FR11:** System shall expose datasets in Power BI workspace.
- **FR12:** Dashboards shall refresh daily from Gold layer or Synapse.

---

6. Non-Functional Requirements (NFRs)

- **Scalability:** Pipelines must scale to 100GB/day ingestion.
- **Security:** Data must be encrypted at rest and in transit.

- **Performance:** API responses < 1 sec for aggregate queries.
- **Monitoring:** Azure Monitor + Databricks job logs + API telemetry.
- **Cost:** Optimize compute with Databricks job clusters vs. all-purpose clusters.

---

7. Tech Stack

- **Data Lake & Storage:** Azure Data Lake Storage Gen2
- **Compute & ETL:** Azure Databricks (PySpark + SQL)
- **Orchestration:** Azure Data Factory (trigger Databricks jobs)
- **Data Warehouse (optional):** Azure Synapse Analytics
- **Visualization:** Power BI
- **API Layer:** C# (.NET 8) Web API + Azure App Service
- **Authentication:** Azure AD + Managed Identity

---

8. Architecture Diagram (Conceptual)

[CRM/ERP/IoT]

↓

[Azure Blob / IoT Hub]

↓

[Azure Data Factory] → Orchestrates → [Databricks ETL: Bronze → Silver → Gold]

↓

[Delta Lake / Synapse]

↙ ↘

[Power BI Dashboards] [C# .NET API Layer in Azure App Service]

---

9. Milestones & Timeline

- **Week 1:** Setup Azure environment (ADLS, ADF, Databricks workspace, App Service).

- **Week 2:** Build ingestion pipelines (Blob + IoT Hub).
- **Week 3:** Develop ETL transformations (Bronze → Silver → Gold).
- **Week 4:** Deploy C# API layer.
- **Week 5:** Build Power BI dashboards.
- **Week 6:** Testing, Monitoring setup, Documentation.

---

10. Success Metrics

- 95% successful ingestion jobs.
- <5% data quality errors.
- <1s response time for API queries.
- Power BI adoption: 50+ daily active users.
