Self-Service Reporting: Empowering Product Owners with IoT Data Control

Anurag Sharma 5 min read · Aug 11, 2025

At our research company, we manage IoT ecosystems that power businesses worldwide, connecting thousands of devices to deliver real-time insights. However, with scale comes complexity. Product owners managing large fleets often faced challenges due to incomplete or delayed data in our reporting layer. To address this, we developed a **self-service reporting platform** that puts raw IoT data directly in their hands, enabling them to troubleshoot issues, analyze performance, and make data-driven decisions without IT bottlenecks. This blog explores the problem, our solution, its technical architecture, and its impact.

The Challenge: Gaps in IoT Data Reporting

Deploying large fleets of IoT devices — sometimes thousands across global regions — presents unique challenges. A recurring issue was that up to **50% of devices** in some fleets failed to publish data to our AWS S3-based cloud storage. This led to significant gaps in our reporting layer, which aggregates and summarizes data for dashboards. Product owners, who are responsible for ensuring device performance and customer satisfaction, were frustrated by:

- **Incomplete Data**: Pre-built reports often lacked granular details, such as raw metrics (e.g., memory usage, CPU load, BIOS versions, or connected peripherals).
- **Delayed Insights**: Requesting custom reports from IT teams could take days, hindering timely decision-making.
- **Troubleshooting Barriers**: Without direct access to raw data, product owners couldn't diagnose issues like devices failing to connect or transmitting incomplete data.

The need was clear: a scalable, secure, and user-friendly solution to give product owners direct access to raw IoT data for custom analysis and troubleshooting.

Our Solution: A Self-Service Reporting Platform

To address these challenges, we built a **self-service reporting platform** that empowers product owners to query raw IoT data on demand. The platform combines a simple user interface, dynamic query generation, robust data processing, and interactive visualizations, all while maintaining strict security protocols.

Here’s how it works:

1. User-Friendly Query Interface

Product owners access a web-based UI to create custom reports by specifying:

- **Table Name**: The dataset to analyze (e.g., device_metrics, system_logs).
- **Column Names**: Specific fields, such as cpu_usage, memory_usage, firmware_version, or connection_status.
- **Date Range**: A time window for the data (e.g., last 24 hours, past week).
- **Distinct Device IDs**: Unique identifiers for their fleet, ensuring they only access data for authorized devices.

The UI is designed for non-technical users, with dropdown menus and auto-suggestions for table and column names, reducing the learning curve.

2. Dynamic Query Generation

Behind the scenes, an **API** retrieves metadata about the requested dataset, including:

- **S3 Path:** The location of the raw data in AWS S3 (e.g., s3://iot-data-bucket/device_metrics/).
- **Schema Details:** All column names and their data types (e.g., cpu_usage: float, timestamp: datetime).
- **Partitioning Information:** How data is partitioned (e.g., by date or device ID) for efficient querying.

Using this metadata, the system dynamically generates a **Spark SQL query** tailored to the user's request. For example, a query to fetch CPU usage for specific devices might look like:

```
SELECT device_id, timestamp, cpu_usage
FROM device_metrics
WHERE device_id IN ('device_001', 'device_002')
AND timestamp BETWEEN '2025-07-01' AND '2025-07-15'
```

This automation eliminates the need for manual query writing, making the system adaptable to new datasets and scalable across large fleets.

3. Data Processing and Storage

The generated Spark SQL query runs as a **Spark job** on an AWS EMR cluster, processing massive datasets stored in S3. Spark's distributed computing capabilities ensure efficient handling of millions of data points. The query results are then loaded into **Amazon Redshift**, a fast, scalable data warehouse, where they are stored in a **temporary table** unique to the user's session. This table is accessible for further analysis or visualization.

Where hate has no platform.
Medium members help preserve a kinder, more thoughtful internet.
Learn more

4. Visualization with Power BI

The temporary table in Redshift is exposed to **Power BI**, where product owners can create custom dashboards and visualizations. For example, they can:

- Plot time-series graphs of CPU or memory usage.
- Identify patterns in device connectivity issues.
- Compare performance across different firmware versions.

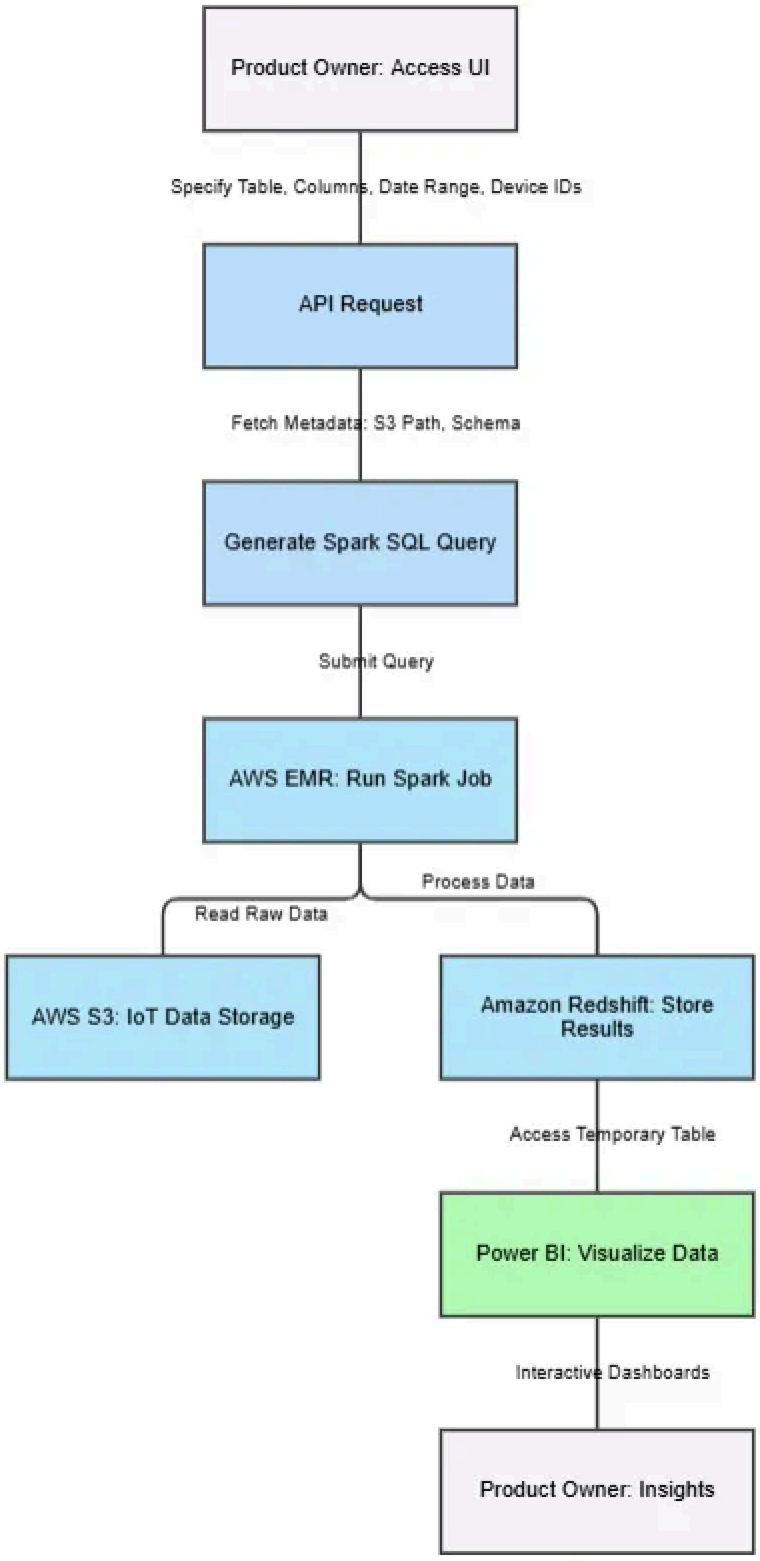
Power BI's drag-and-drop interface allows non-technical users to explore data interactively, turning raw numbers into actionable insights.

5. Security and Access Control

Security is paramount. The platform enforces strict access controls:

- **Device ID Filtering:** Product owners can only query data for devices they are authorized to manage, enforced via AWS IAM roles and Redshift user permissions.

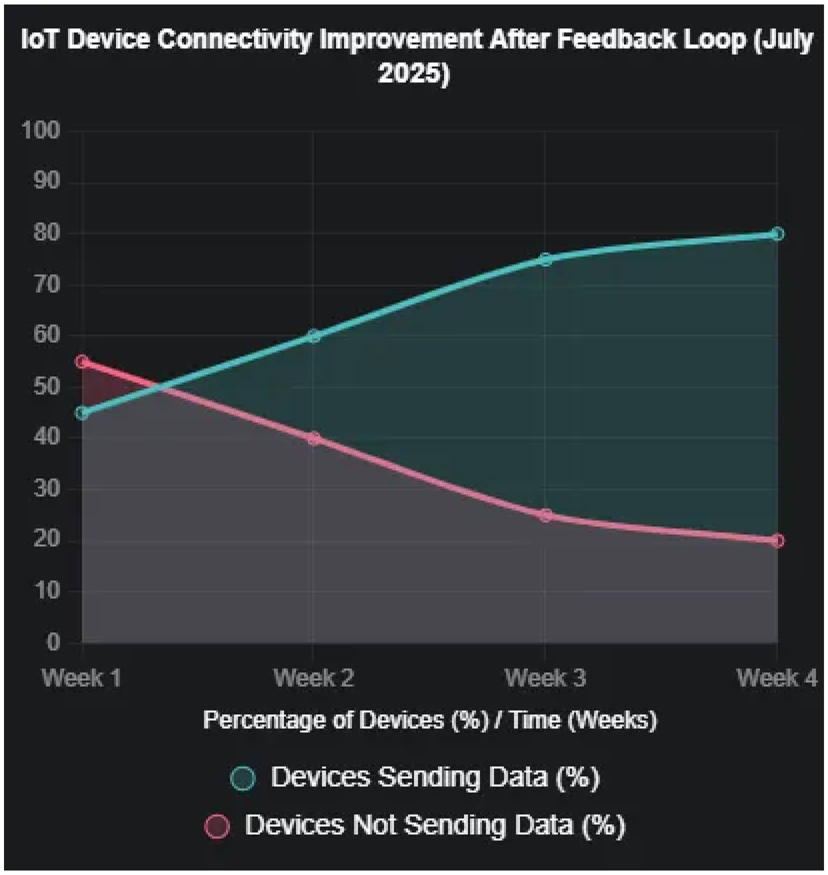
- **Data Encryption:** Data is encrypted in transit (TLS) and at rest (AES-256) in S3 and Redshift.
- **Temporary Tables:** Results are stored in temporary tables that are automatically deleted after the session, minimizing data exposure.
- **Audit Logging:** All queries are logged for compliance and monitoring.



Technical Highlights

The platform's architecture leverages modern cloud and big data technologies to ensure scalability, flexibility, and ease of use:

- **AWS S3:** Stores raw IoT data in a cost-effective, scalable manner, partitioned by date and device ID for efficient querying.
- **AWS EMR with Spark:** Processes large-scale data with Spark SQL, handling complex queries across terabytes of data.
- **Amazon Redshift**: Provides a robust data warehouse for storing query results and enabling fast, SQL-based analysis.
- **Power BI Integration**: Enables intuitive, interactive visualizations without requiring coding skills.
- **API-Driven Metadata:** Dynamically fetches dataset schemas, making the system adaptable to new data types without code changes.
- **Scalability:** The platform handles fleets of thousands of devices, processing millions of data points in minutes.
- **Extensibility:** New datasets can be added by updating metadata, with no changes to the core system.



Impact: Empowering Product Owners

The self-service reporting platform has revolutionized how product owners interact with IoT data.

Key benefits include:

- **Faster Troubleshooting:** Product owners can quickly verify which devices are sending data, addressing the "missing 50%" issue. For instance, one team identified that a batch of devices had outdated firmware, causing connectivity failures, and deployed a targeted update within hours.
- **Granular Insights:** Access to raw metrics like memory usage or BIOS versions enables detailed performance analysis and issue diagnosis.

- **Reduced IT Dependency:** Custom reports that once took days now take hours, as product owners generate them independently.
- **Improved Decision-Making:** Interactive Power BI dashboards provide actionable insights, such as identifying underperforming devices or optimizing fleet configurations.

Data Analysis Data Analytics Spark Optimization Big Data

Written by Anurag Sharma

83 followers · 3 following

Edit profile

Data Engineering Specialist with 10+ exp. Passionate about optimizing pipelines, data lineage, and Spark performance and sharing insights to empower data pros!

No responses yet

Anurag Sharma him/he

What are your thoughts?