API Logs Platform

Introduction

The API Logs Platform for Product Metrics is designed to help Walmart teams monitor, troubleshoot, and optimize APIs that power Walmart's products. It is scalable, provides clarity, and is intended for developers, data scientists, and operations managers to gain insights into API performance and product data.

This document explains how this platform works to help monitor, troubleshoot, and optimize APIs that power Walmart's products. Built for scalability and clarity, it helps identify issues, track performance, and improve the overall product lifecycle.

Key Features

- Real-Time Monitoring: Captures live API data for instant analysis.
- Scalability: Handles high volumes of API requests across Walmart's infrastructure.
- Error Analysis: Helps pinpoint root causes of API failures.
- Custom Metrics: Extracts business-relevant metrics (e.g., product availability, pricing, sales trends).
- Compliance Support: Provides detailed audit trails for regulatory purposes.

Use Cases

- Debugging API issues
- Optimizing API performance
- Analyzing product metrics
- Conducting security audits

Prerequisites

To effectively utilize the **API Logs for Product Metrics** platform, ensure the following prerequisites are met:

1. **Onboard to the API Audit Logging Service**
   To collect product metrics for your service, your application must be integrated with the **audit-logging-service**. Follow the step-by-step instructions in the [**Audit Logging Service Integration Guide**] to complete the onboarding process.

2. **Data Discovery Access for Visualization**
   API log records are stored in a GCS (Google Cloud Storage) bucket, with a corresponding Hive table created on top of this data. To visualize this data, you need access to **Data Discovery**.

   Before proceeding, confirm that you have access to the Data Discovery tool. If access is not granted, follow the necessary steps to obtain it.

3. **Active Directory (AD) Group Access**
   Access to the API logs requires membership in the appropriate AD groups. Ensure your AD access includes the following:

Required AD Groups and Access Details

GCP-DATA-DISCOVERY-PROD-ROLE-LOGIN	Access to Data Discovery	Service Now	**Link** - Data Discovery

GCP-DL-DV-LUMINATE-DEV-READ	Access to test data in Data Discovery	Service Now	**Schema** - us_dv_audit_log_dev **Table** - api_logs
GCP-DL-DV-LUMINATE-PROD-READ	Access to production data in Data Discovery	Service Now	**Schema** - us_dv_audit_log_prod **Table** - api_logs

4. **Explore Sample Queries**
Review the provided sample queries for guidance on how to interact with the API log data effectively. Additional queries may assist in debugging or refining your product metrics.

```
1. Query by Service Name, Endpoint Name, and Response Code:

SELECT * FROM us_dv_audit_log_prod.api_logs WHERE service_name = 'NRT' AND endpoint_name = 'inventoryActions' AND
response_code = 400 LIMIT 10;
```

5. **Read Documentation**:

- **ADT** - API Logs for Product Metrics - ADT
- **GCS Sink Service** - Kafka Connect GCS Sink Service

By ensuring these prerequisites are in place, you'll be set up to successfully integrate, access, and visualize API logs for actionable insights.

How the API Log Service Works

The API Log Service is a critical component of the technology backbone that enables seamless monitoring and tracking of API interactions within Walmart's ecosystem. By gathering and organizing API logs, this service ensures that product metrics and other data can be accurately analyzed for insights. Below is an explanation of how the API Log Service works, broken into key components and processes.

1. Overview of API Log Service
The API Log Service acts as a processing pipeline for logging API requests and responses. It allows teams to:

- Monitor API usage.
- Debug failures or errors in real-time.
- Collect product metrics and performance data.
- Provide audit trails for security and compliance.

This service ensures that API-related data is recorded in a structured and accessible way to enable better decision-making.

2. Core Workflow of API Log Service
Here is an overview of how data flows through the API Log Service:

Step 1: API Call Execution
Whenever a system or service calls an API, the request and response are automatically logged. The log data typically includes:

- API endpoint URL.
- HTTP method (GET, POST, PUT, DELETE).
- Request payload (if applicable).
- Response status code and payload.
- API latency (time taken to process the request).
- Timestamp of the interaction.
- Authentication or session details (if logged).

Step 2: Log Data Capture
Logs are captured at various points in the API lifecycle, such as:

- WCNP app logging.

API logs platform captures log from application running in WCNP ecosystem.

Step 3: Data Processing
Once the log data is collected, it undergoes processing to standardize the format and extract key metrics relevant to the API usage. This step may involve:

Data Parsing: Splitting raw logs into structured fields (e.g., timestamp, endpoint, error codes).

Filtering: Removing unnecessary or duplicate entries to reduce noise.

Enrichment: Adding metadata like service names, geographical information, or user IDs.

Step 4: Centralized Storage
Processed log data is sent to a centralized storage system, which could be:

- Cloud-based logging services (e.g., Splunk, ELK stack).
- Distributed databases for large-scale storage.
- Internal proprietary logging platforms.

The storage system ensures long-term data retention for historical analyses and compliance purposes.

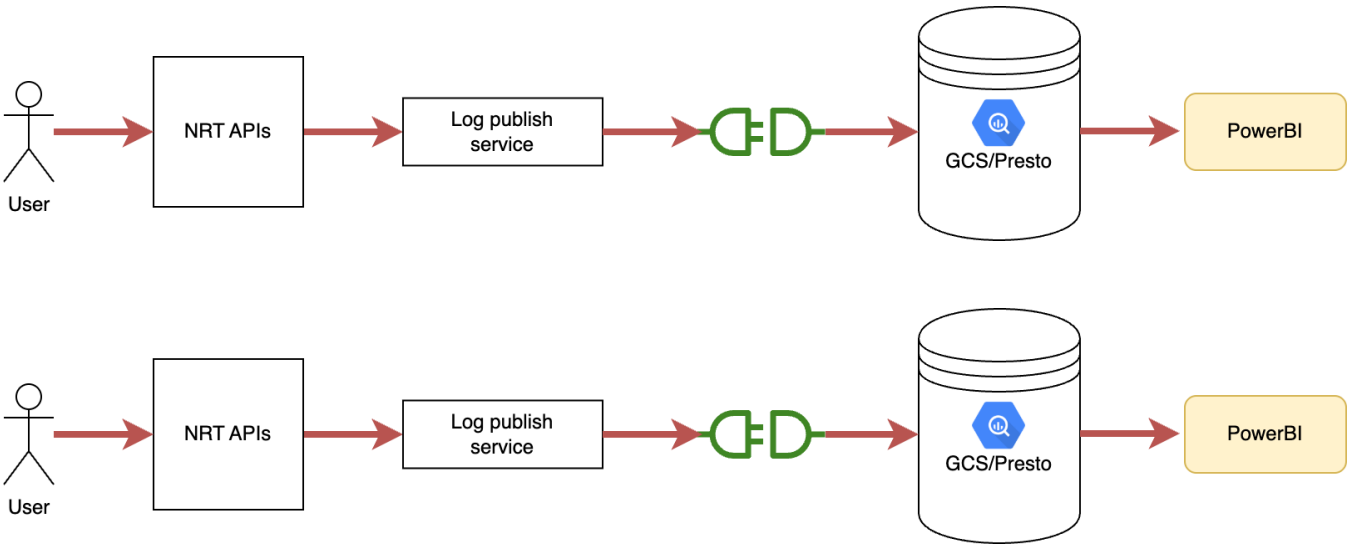
Step 5: Visualization & Reporting
Once the logs are organized and stored, the API Log Service interfaces with reporting tools and dashboards to provide insights. Teams can use these tools to:

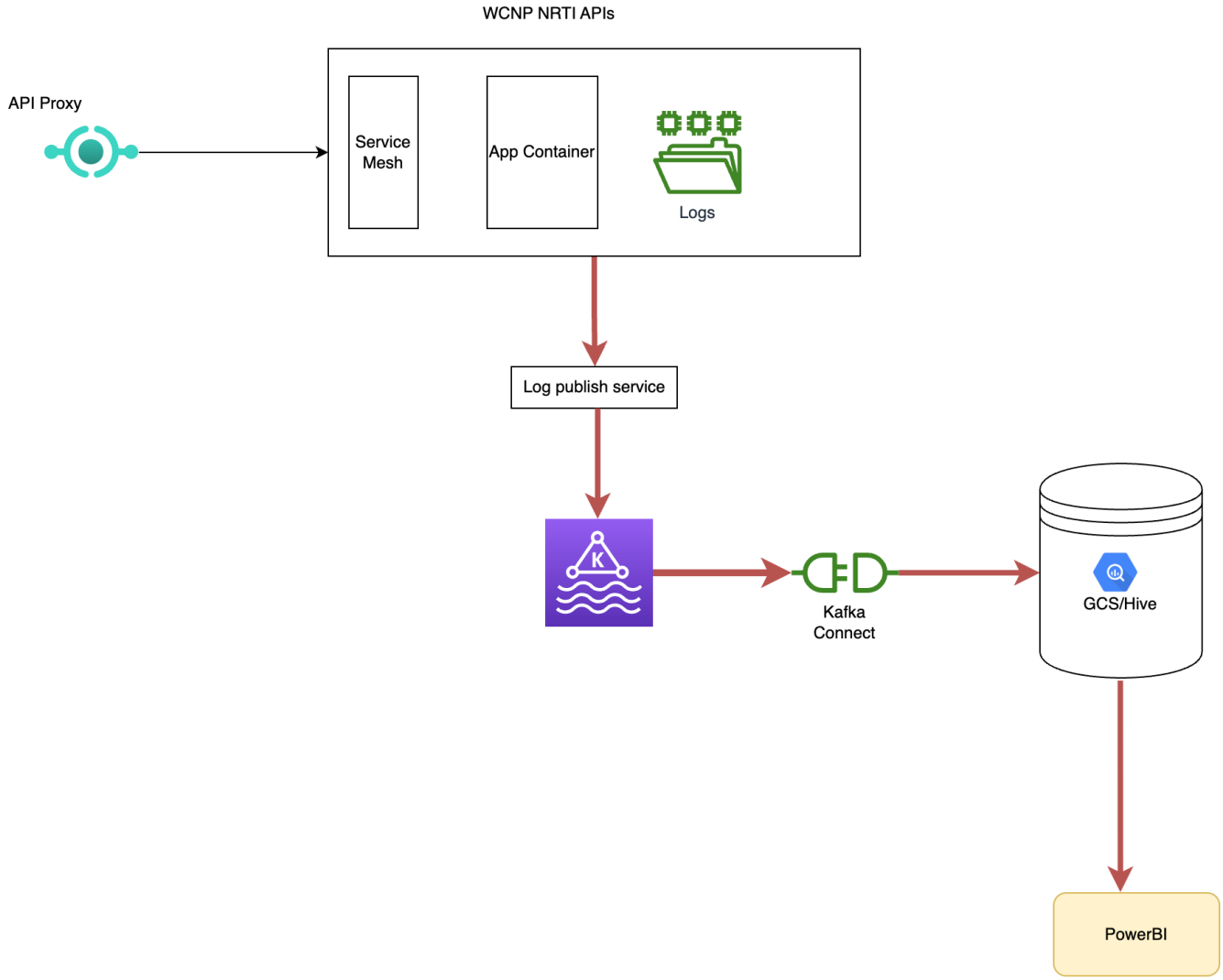
- Monitor API performance metrics (e.g., response times, throughput, error rates).
- Detect failures or anomalies.
- Generate reports for business stakeholders.
- Walmart's internal dashboards and visualization systems likely deliver live feeds and historical reports for deeper analysis.

Step 6: Alerts and Notifications
The API Log Service can be configured to trigger alerts or notifications based on predefined thresholds or rules. Examples include:

- Notifying engineers if the error rate for an API crosses 5%.
- Sending an alert when an unauthorized access attempt is logged.

This feature ensures proactive monitoring and enables faster resolution of issues.





3. Key Features of the API Log Service

- **Real-Time Monitoring**: Captures and delivers live data for instant analysis.
- **Scalability**: Handles high volumes of API requests across Walmart's extensive infrastructure.
- **Error Analysis**: Pinpoints root causes of API failures to streamline the debugging process.
- **Custom Metrics**: Extracts business-relevant metrics, such as product availability, pricing, and sales trends.
- **Compliance Support**: Logs provide a detailed audit trail for regulatory purposes.

4. Use Cases
The API Log Service supports various use cases for Walmart's teams:

- **Debugging Issues**: Developers can analyze logs to identify and correct API errors.
- **Optimizing Performance**: Logs help teams optimize API response times and reduce latency.
- **Product Metrics Analysis**: By mining log data, product teams can derive insights into inventory, pricing, and user behavior.
- **Security Audits**: Logs help in tracking unauthorized access attempts or vulnerabilities.

5. Collaboration with Other Systems
The API Log Service integrates with:

- **Monitoring Systems (e.g., Prometheus, Grafana)**: For real-time metrics visualization.
- **CI/CD Pipelines**: To test and analyze API performance during deployments.
- **Data Analytics Tools**: Power BI Dashboard etc.

Integration Steps

The Audit Logging Service enables Walmart teams to capture, persist, and analyze HTTP request and response data for APIs. This facilitates product metrics, operational insights, and compliance reporting. The service operates via an API-based model, accepting POST requests and returning a 204 status code upon successful log ingestion. Currently, Kafka is supported as the log target.

1. Onboard Your Application

Raise a ticket with the Audit Logging Service team to onboard your application and obtain a unique consumer ID for API access.

2. Add Dependency

For Java Applications:
Add the Audit Logging JAR dependency to your pom.xml.
Choose the appropriate version for your JDK (11 or 17).
It is strongly recommended to specify a version, even though the latest JAR is backward compatible.

3. Configure Spring Boot Application

In your main Spring Boot application class, specify the packages to scan for JAR utilities:
@SpringBootApplication(scanBasePackages = {"your.package", "audit.log.package"})

4. Build the Project

Run the following Maven command to include the JAR in your external dependencies:
mvn clean install

5. Configure CCM Properties and Feature Flags

a. Feature Flag Configuration

Add the following configuration in your application's CCM:
isAuditLogEnabled: true # Master switch for audit logging

true: Audit logging is active.
false: Audit logging is disabled.
Default is true.

b. Audit Logging Configuration

Add detailed configuration in CCM:
wmConsumerId: <your-unique-consumer-id>
auditLogURI: <audit-log-service-url>
enabledEndpoints: nrt_transactionHistory=^\\/store\\/\\d+\\/gtin\\/\\d+\\/transactionHistory$
isResponseLoggingEnabled: true
keyVersion: <encryption-key-version>
auditPrivateKeyPath: <path-to-private-key>
serviceApplication: <your-application-name>

wmConsumerId: Unique identifier for your application.
auditLogURI: URL of the Audit Log Service.
enabledEndpoints: List of endpoints and regex patterns to be audited.
isResponseLoggingEnabled: Whether to log response bodies (use with caution).
keyVersion: Encryption key version.
auditPrivateKeyPath: Secure path to your private key (do not store directly in the app).
serviceApplication: Name of your application.

6. Secure Your Private Key

Store the RSA private key securely (e.g., in AKeyless) and mount it at runtime. Never store the private key directly in the application code or repository.

7. Validate Integration

Send a test log entry and verify a 204 response.
Check that logs are being published to Kafka and persisted in storage.
Confirm that logs appear in downstream reporting or dashboards.

Best Practices

Endpoint Regex: Ensure regular expressions accurately match only the intended URLs.
Sensitive Data: Avoid logging sensitive or PII data. If necessary, implement masking logic before sending logs.
Performance: Enable response logging only if required, as it increases log volume.
Security: Protect API keys, use HTTPS, and follow Walmart security guidelines.

Troubleshooting

401 Unauthorized: Ensure your authentication token and consumer ID are valid.
Payload Validation Errors: Confirm your payload matches the expected schema.
No Logs Appearing: Check endpoint regex, feature flag, and CCM configurations.

Example cURL Request

Note - The request_body, response_body, headers to be updated with respective application's details. It is reference for the data structure

```
curl --location 'http://audit-logs-svc-stage.data-ventures-luminate-cperf.eus2-stage-a4.cluster.k8s.us.walmart.net/v1/logs/api-requests' \
--header 'WM_SVC.NAME: AUDIT-API-LOGS-SRV' \
--header 'WM_SVC.ENV: stg:1.0.0' \
--header 'WM_SVC.VERSION: 1.1.0' \
--header 'WM_CONSUMER.ID: fda7cddb-b0ea-451e-9d2a-b090a08290ae' \
--header 'Content-Type: application/json' \
--header 'WM_SEC.AUTH_SIGNATURE;' \
--header 'WM_CONSUMER.INTIMESTAMP: 1751905746570' \
--data '{
    "request_id": "99ff9f73-cef8-4696-855c-c57714ebc906",
    "service_name": "NRT",
    "endpoint_name": "transactionHistory",
    "version": "v1",
    "path": "/store/1998/gtin/00030772056301/transactionHistory",
    "method": "GET",
    "request_body": {
        "messageId": "dr-test-post-dr-1",
        "eventType": "ARRIVAL",
        "storeNbr": 45,
        "lineInfo": [
            {
                "gtin": "00044444444446",
                "secondaryItemIdentifier": {
                    "type": "ItemNbr",
                    "value": "553352632"
                },
                "destinationLocation": {
                    "locationArea": "STORE",
                    "location": "A142-002",
                    "lpn": "1234567890"
                },
                "quantity": 1,
                "expiryDate": "2023-09-30"
            }
        ],
        "documentInfo": [
            {
                "docType": "INVOICE",
                "docNbr": "10077",
                "docDate": 1676902620356
            }
        ],
        "userId": "testuser",
        "reasonDetails": [
            {
                "reasonCode": "xyz",
                "reasonDesc": "xyz"
            }
        ],
        "vendorNbr": "1234567",
        "eventCreationTime": 1676902620356
    },
    "response_body": {
        "errors": [
            {
                "instance": "/store/100/gtin/00000000040112/transactionHistory",
                "message": "Data not found for the given request parameters.",
                "status": "NOT_FOUND",
                "timestamp": "2025-04-01T04:56:15"
            }
        ]
    },
    "error_reason": "Data not found for the given request parameters.",
    "response_code": 200,
    "request_ts": 1744000188677,
    "supplier_company": "luminate_company_id",
    "response_ts": 1744000188678,
    "request_size_bytes": 2000,
    "response_size_bytes": 16,
    "created_ts": 1744000188677,
```

```
    "trace_id": "182589fa7f75585af5b05b1ff556f213",
    "headers": {
        "wm-site-id": "1694066566785477000",
        "content-type": "application/json",
        "WM_CONSUMER.ID": "fda7cddb-b0ea-451e-9d2a-b090a08290ae",
        "WM_SVC.NAME": "channelperformance-nrti",
        "WM_SVC.ENV": "stg",
        "WM_SVC.VERSION": "1.0.0",
        "Authorization": "load-test-2"
    }
}'
```

Refer for more details - Audit Logging Service Integration Guide

Visualizing Product Metrics

Use Walmart's approved BI tools (e.g., Tableau, Power BI) for dashboarding and reporting.
Connect to the Hive tables (us_dv_audit_log_dev.api_logs and us_dv_audit_log_prod.api_logs) as data sources for non-prod and prod api logs.

Recommended Visualizations

API Usage Trends: Track volume and types of API calls over time.
Error Rates: Visualize failed API calls and error types.
User Activity: Analyze which users or services are generating the most audit events.
Data Freshness: Monitor the timeliness of data refreshes via workflow status dashboards.

Required AD Groups and Access Details

GCP-DATA-DISCOVERY-PROD-ROLE-LOGIN	Access to Data Discovery	Service Now	**Link** - Data Discovery
GCP-DL-DV-LUMINATE-DEV-READ	Access to test data in Data Discovery	Service Now	**Schema** - us_dv_audit_log_dev **Table** - api_logs
GCP-DL-DV-LUMINATE-PROD-READ	Access to production data in Data Discovery	Service Now	**Schema** - us_dv_audit_log_prod **Table** - api_logs

Hive Workflow Automation

Airflow Workflow

- **Workflow Name:** Prod_Metrics_Hive_Tbl_Refresh_WF
- **Purpose:** Refreshes Hive tables for Product Metrics audit logs.
- **Airflow Links:**
  - Non-Prod: Airflow Non-Prod
  - Prod: Airflow Prod

Automic Workflow

- **Job Name:** JOBS.DVCHNLPERF.AUDIT.LOGGING.TBL.REFRESH.JOB
- **Automic Link:** Automic Job

Summary

The API Logs Platform is a robust, scalable solution for monitoring and analyzing API activity within Walmart. It supports operational excellence, compliance, and business insights, provided users follow the onboarding, access, and best practice guidelines.

References

API Logs for Product Metrics - ADT

Audit Logging Service Integration Guide

Visualising Product Metrics