1. AWS service chosen and reasonings:

S3
Used as the central data lake for storing all raw and processed data assets such as images, audio, and Parquet files. It provides scalable, durable, and cost-effective object storage.

EC2 / EKS
Hosts Fluentd and Snowplow Collector as containerized applications deployed in pods.

- EC2/EKS is preferred over AWS Lambda to avoid cold start latency and unpredictable cost scaling under high-throughput workloads.
- Fluentd collects structured logs/events; Snowplow Collector ingests behavioral tracking data from mobile/web SDKs.

MSK (Managed Kafka)
Acts as the message bus for real-time ingestion.

- Buffers events between producers (Fluentd, Snowplow) and consumers (Apache Flink, Spark), enabling decoupled, scalable streaming.

EMR
Runs Apache Flink and Spark for big data processing.

- Flink handles real-time stream processing, while Spark is used for batch ETL.
- EMR provides flexible scaling, integration with S3, and easier debugging/logging compared to AWS Glue jobs.

MWAA (Managed Airflow)
Serves as the workflow orchestrator, responsible for scheduling and monitoring batch jobs.

- Integrates seamlessly with AWS services and supports complex ETL pipelines with retry, alerting, and dependency management.

Glue Data Catalog
Manages metadata (schema, partition info) for data stored in S3.

- Acts as a unified catalog for Apache Iceberg and Parquet datasets, enabling interoperability with engines like Athena and EMR.

Athena
Serverless interactive query engine for ad hoc analysis.

- Queries Iceberg/Parquet tables directly from S3 using SQL, leveraging the Glue Data Catalog for schema discovery.

QuickSight
Used for dashboarding and visual analytics: enables business users to explore data through visuals and KPIs built on top of Athena queries or other data sources.

2. Data Flow Description

Real-Time Pipeline (Snowplow & Fluentd)

1. Fluentd (Optional): Used for collecting structured log or event data from IoT devices or edge sources. Fluentd receives event payloads and forwards them directly to a Kafka broker for real-time processing.
2. Snowplow Collector: Deployed as pods in Kubernetes, this HTTP endpoint receives structured metadata events pushed from Snowplow SDKs embedded in mobile or web applications. Each event is serialized in Thrift base64-encoded format and published to a Kafka topic.
3. Kafka: Acts as a central event buffer for both Fluentd and Snowplow sources, allowing decoupled, scalable ingestion and delivery.
4. EMR (Flink): Reads data from Kafka, transforms and enriches the metadata, then serializes it into Parquet format and stores it in the Silver zone in S3, partitioned by time or source.

Batch Pipeline (Airflow + Spark)

1. MWAA (Airflow) orchestrates a DAG that runs at configurable intervals (e.g., hourly, daily).
2. The DAG uses a custom Python operator that leverages aiohttp, an asynchronous HTTP client, to concurrently fetch large volumes of data based on pagination from one or more external APIs.
3. Each fetch task writes raw JSON responses to the Bronze layer in S3, organized in subfolders by interval (e.g., date/hour).
4. Once data is staged, the DAG triggers an EMR Spark job to:
    - Load raw JSON from Bronze staging folders
    - Parse and flatten nested structures, cleanse nulls, normalize date formats => pre-clean
    - Enforce schema and enrich with batch metadata (e.g., batch_id, ingestion_time)
    - Convert the cleaned data into Parquet with Iceberg table format and store in the Silver zone
5. After that, based on business requirements, MWAA will trigger DAGs to transform (join, filter) mutiples parquets files into corresponding business-based parquet files, which are stored in the S3 gold layer.

3. Handling of Image Metadata

- Metadata includes fields like image_id, inspection_id, timestamp, gps_location, image_type, device_id, etc.
- Events are ideal for JSON/Parquet storage.

- In the **Bronze layer**, metadata is initially stored in raw JSON format inside structured subfolders.
- **EMR Spark** transforms raw JSON into normalized Parquet format in the **Silver layer**, applying schema enforcement and data quality rules.
- The **Gold layer** optionally contains aggregated metrics, filtered subsets, or join results with external data for business analysis.

4. Scalability Considerations

Migration Considerations for Cost Optimization & Flexibility

To improve cost-efficiency and maintain scalability as the business expands:

- **EMR (Spark) → Spark on Kubernetes via Spark Operator**: Reduce dependency on EMR-managed EC2 instances and improve autoscaling granularity.
- **EMR (Flink) → Flink Kubernetes Operator**: Achieve better control over streaming job deployments with pod-level autoscaling.
- **MSK (Managed Kafka) → Kafka Strimzi on EKS**: Lower operational cost and maintain full control over Kafka configuration.
- **MWAA (Airflow) → Airflow deployed on EKS**: Improve customization and reduce long-term costs.
- **Athena → StarRocks or Trino**: For lower latency, better support for joins and complex analytics without paying for S3 scan volume.