**Big Data Processing Pipeline Design**

# 1. Project Description

**Title:  
*Multi-Modal Big Data Processing System with Hybrid Storage Architecture***

**Key Problem Statement:  
"Processing diverse data types (text, images, videos) at scale while maintaining optimal performance across different access patterns and storage requirements"**

**Significance:**

* **Handles modern data variety (unstructured images/videos + structured metadata)**
* **Demonstrates hybrid storage patterns for different data dimensions**
* **Solves real-world challenges in content platforms, IoT systems, and media analytics**

**Core Capabilities:**

1. **Synthetic data generation (text, images, videos)**
2. **Stream processing with Kafka**
3. **Multi-modal storage optimization:**
   * Object storage for raw files (MinIO)
   * Document store for metadata (MongoDB)
   * Time-series for metrics (Cassandra)
   * Search index (Elasticsearch)
4. **Monitoring stack (Prometheus + Grafana)**

# 2. Design Challenges

**A. Data Variety Management**

**Challenge:**  
Handling 3 distinct data types with different:

* Size characteristics (KB text vs MB media)
* Access patterns (frequent metadata vs rare raw video access)
* Processing requirements (text analysis vs image thumbnailing)

**B. Storage Optimization**

**Challenge:**Avoiding "one-size-fits-all" storage anti-pattern while maintaining:

* Cost efficiency
* Query performance
* Scalability

**Solution Matrix:**

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Storage Choice** | **Rationale** |
| Image/Video | MinIO (S3-compatible) | Optimal for large binaries |
| Metadata | MongoDB | Flexible schema for varied formats |
| Metrics | Cassandra | High-write throughput for TS data |
| Search Index | Elasticsearch | Full-text search capabilities |

**C. Pipeline Reliability**

**Challenge:**  
Ensuring exactly-once processing semantics across:

* Stream ingestion (Kafka)
* Distributed storage systems
* Failure-prone network operations

**D. Resource Constraints**

**Challenge:**

Managing memory-intensive operations (image processing) in containerized environment

# 3. Big Data Architecture & Tools

## Architecture Diagram

Data Flow:

Generator -> Kafka -> Processor -> [MinIO(S3), MongoDB, Elasticsearch, Cassandra]

Monitoring:

[Prometheus] <- Metrics <- [Processor, Generator]

|

v

[Grafana]

## Tool Selection Rationale

**1. Apache Kafka**

* *Why:* Handles 10K+ msg/sec with 5MB payloads

**2. MongoDB**

* *Why:* Schema flexibility for evolving metadata

**3. Cassandra**

* *Why:* Write-optimized for metrics (10K+ writes/sec)

**4. MinIO**

* *Why:* S3 API compatibility for cloud portability

**5. Elasticsearch**

* *Why:* Combined full-text + geo queries