Design Phase Document for APM Ingestion Service

# 1. High-Level Architecture Diagram

The architecture consists of the following main components:  
  
Monitored Applications send APM data (logs and metrics) to the APM Ingestion Service via HTTP POST requests.  
The APM Ingestion Service (Node.js + Express) validates and processes this data.  
Data is stored temporarily in the Mock Database Module (in-memory JS objects).  
Metrics and debug endpoints allow retrieval and management of stored data.  
Clients or dashboards consume these APIs to display or analyze data.

## Architecture Diagram

Monitored Apps (e.g. user-service) → POST /ingest → APM Ingestion Service → Save data → Mock Database Module → Metrics & Debug APIs → Clients/Dashboard  
  
Visual Flow:  
Monitored Apps --> APM Ingestion Service --> Mock Database Module --> Metrics & Debug APIs --> Clients/Dashboard

# 2. Component Descriptions

- Monitored Applications:  
 Simulated apps generating APM data such as logs and performance metrics, sending JSON payloads to `/ingest`.  
  
- APM Ingestion Service:  
 Node.js backend with Express that exposes REST API endpoints to receive, validate, and process data. It logs received data and forwards it to the mock database.  
  
- Mock Database Module:  
 Simple in-memory JavaScript objects/arrays to store logs and metrics temporarily during runtime. Supports saving, retrieval, and clearing of data.  
  
- Metrics & Debug APIs:  
 Endpoints to aggregate metrics (e.g., count of errors, average response times) and view or clear all stored data.

# 3. Technology Stack

| Layer | Technology/Library |  
|--------------|----------------------------|  
| Backend | Node.js, Express |  
| Middleware | body-parser |  
| Data Storage | In-memory JavaScript objects|  
| API Type | RESTful APIs |  
| Deployment | Render (Cloud Platform) |

# 4. Data Flow

1. Client apps send HTTP POST requests with JSON payloads describing logs or metrics to the ingestion service at `/ingest`.  
2. The APM Ingestion Service validates the payload for required fields and data type.  
3. Valid data is saved to the mock database module.  
4. Clients can query `/metrics` endpoint to receive aggregated statistics.  
5. Debug endpoints `/view-all-data` and `/clear-all-data` allow inspection and reset of the stored data.

# 5. Sequence Diagram (Example)

- Client → POST `/ingest` → APM Ingestion Service  
- APM Ingestion Service → validate data → save to Mock DB  
- Client → GET `/metrics` → APM Ingestion Service → query Mock DB → return summary

# 6. Assumptions & Constraints

- Uses a mock database (non-persistent, in-memory storage).  
- Designed for demonstration and learning, not production use.  
- Handles only two data types: logs and metrics.  
- No authentication or security measures implemented yet.

# 7. Possible Future Enhancements

- Integrate a persistent database (e.g., MongoDB, PostgreSQL).  
- Add user authentication & authorization.  
- Build a frontend dashboard for live metrics visualization.  
- Implement alerting mechanisms based on error thresholds.  
- Support more diverse data types and complex queries.