BILLING SYSTEM IN DISTRIBUTED COMPUTING USING APACHE KAFKA

Team TechTitans

Team Members

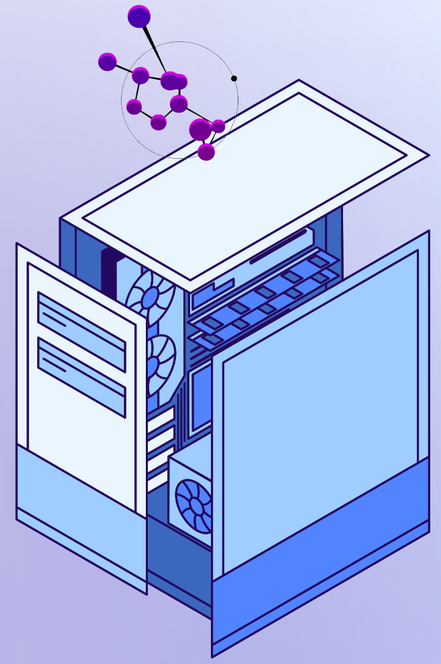
Mitali Manish Admuthe

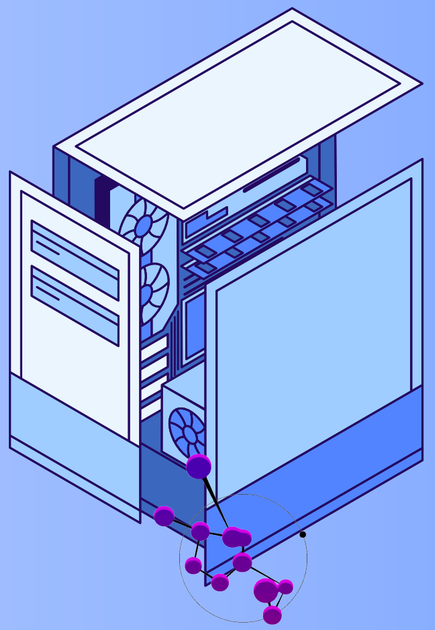
Joyli Marshal Rumao

Pooja Devendra Chavan

Sonal Bijitkar

IMPORTANCE OF BILLING IN DISTRIBUTED SYSTEMS

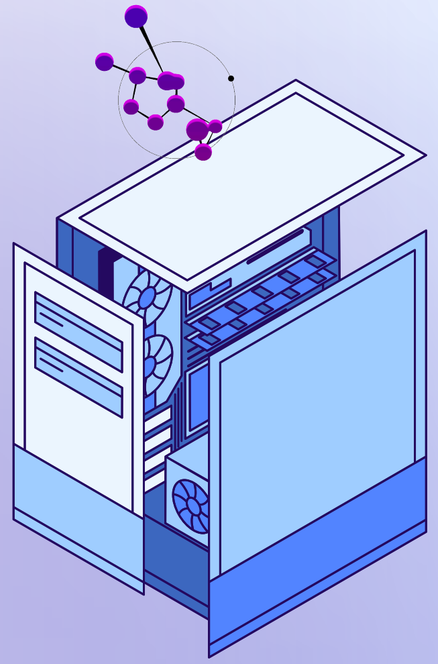


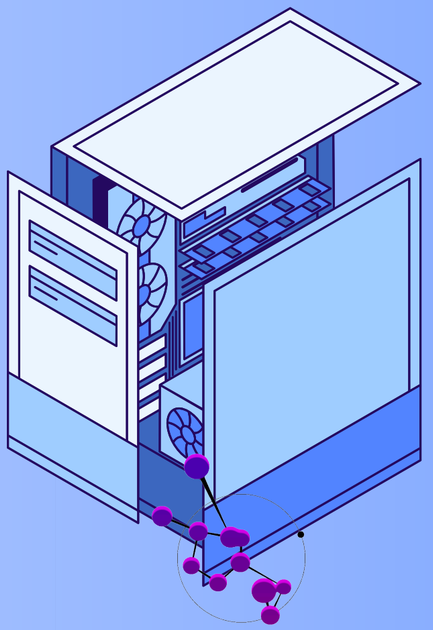


Why Billing Matters?

1.Resource Optimization
2.Cost Allocation
3.Transparency and Accountability
4.Budgeting and Forecasting

WHAT IS A CENTRALIZED BILLING SYSTEM?





- Data processing, calculations, and transaction handling occur on a single central server.
- All customer data, resource usage information, and billing activities are funneled into and managed from one main system.
- The system processes payments and updates records, ensuring that every transaction is stored centrally for easy access and tracking.

DRAWBACKS OF A CENTRALIZED SYSTEM

- Scalability
- Single Point of Failure
- High latency and low response
- Lack of Real time insights

PROBLEM STATEMENT

Our project implements a distributed system for monitoring resource usage and calculating billing costs based on CPU and memory usage across multiple machines. It uses Apache Kafka for distributed messaging, enabling producers to send resource usage data to consumers who perform billing calculations. The system demonstrates the scalability and efficiency of Kafka in distributed computing environments.

KEY FEATURES

- Real-Time Monitoring: Tracks CPU and memory usage continuously.

- Interactive Dashboard: Visualizes resource usage data with charts for easy understanding.

- Multi-Machine Support: Runs on multiple machines simultaneously.

- Configurable Update Interval: Default interval of 5 seconds for data fetch and update.

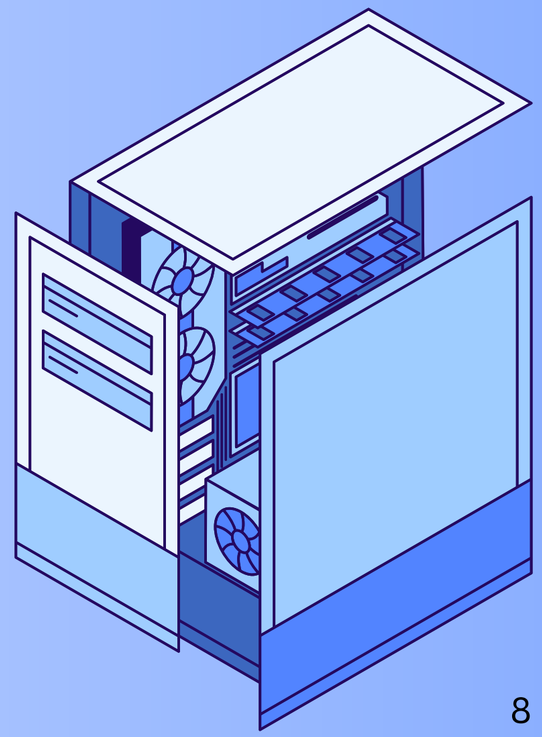
SYSTEM ARCHITECTURE OVERVIEW

Primary Components:

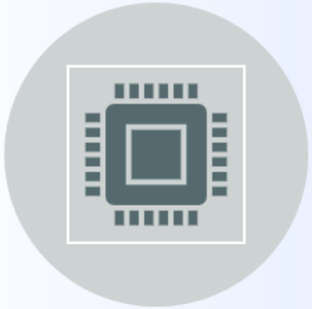
- **Kafka:** Acts as a reliable, distributed messaging system for efficient communication.
- **SQLite Database:** Stores data in a structured format, recording each machine's CPU and memory usage for billing purposes.
- **Distributed Nodes:** Individual machines or servers monitoring their usage and sending data to Kafka.
- **Worker Process:** The workers script runs on nodes, consuming data from Kafka, storing it in the database, and calculating billing costs.

What is Apache Kafka?

Apache Kafka is an open-source, distributed streaming platform that uses the publish and subscribe mechanism to stream the records in key-value pairs, that allows for the development of real-time event-driven applications, Specifically, It Allows Developers to that continuously Produce and Consume streams of data records. For Each Produce and Consume, there will be a specific Topic



Key features of Apache Kafka



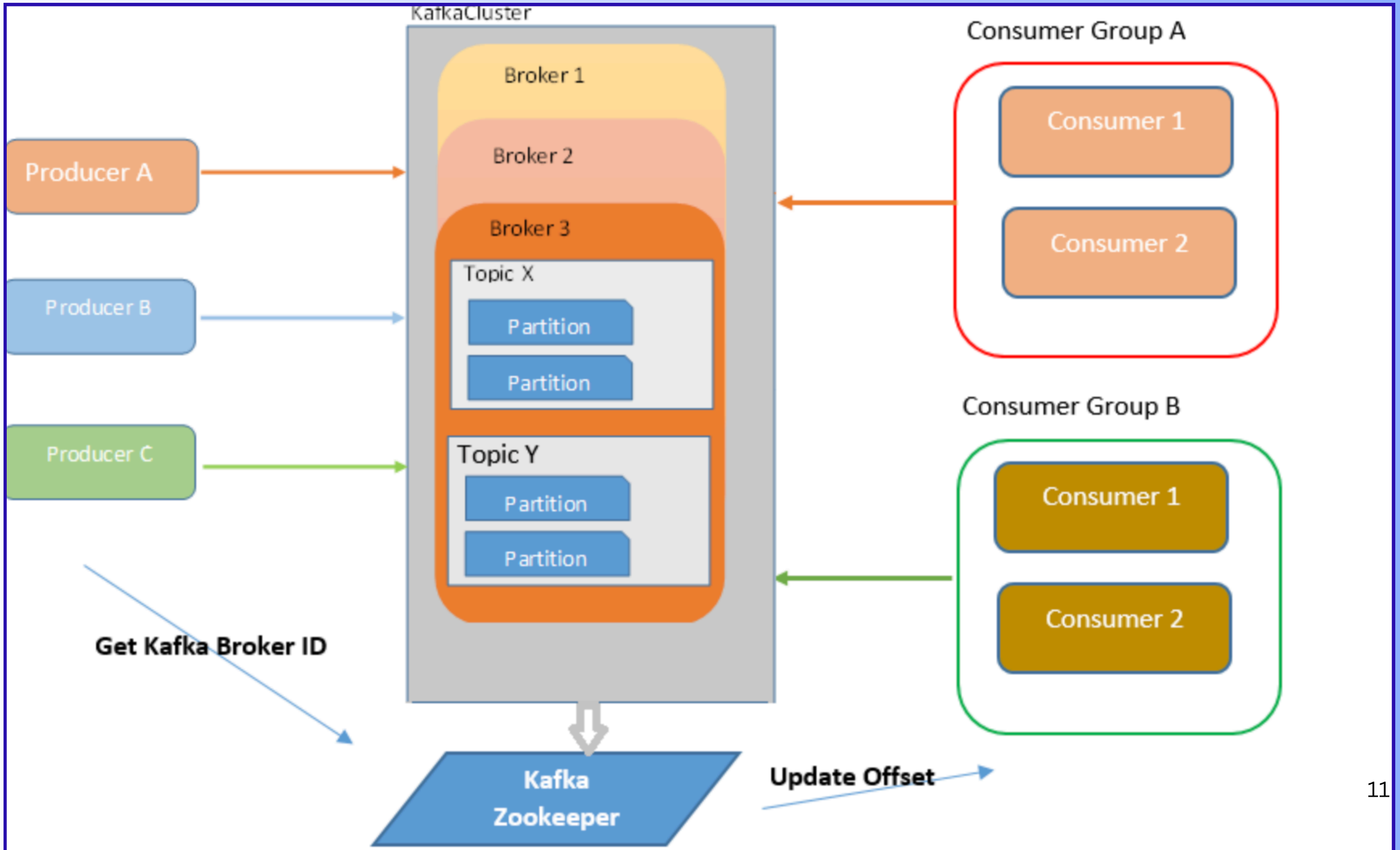
Distributed: Scalable across many machines

High Throughput: Handles millions of messages per secondLow

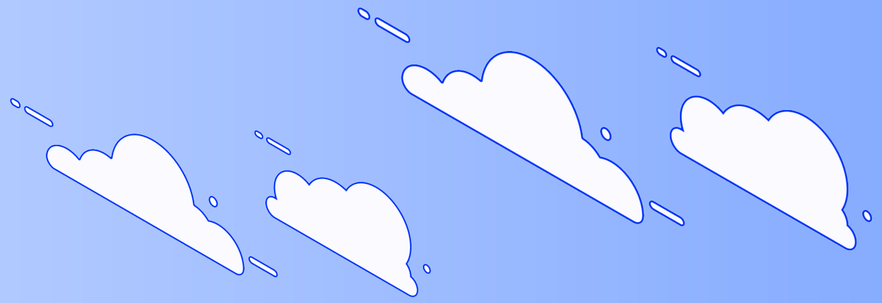
Latency: Real-time data streaming

Fault-Tolerant: Ensures data reliability with replication

KAFKA ARCHITECTURE


KafkaCluster
Broker 1
Broker 2
Broker 3
Topic X
Partition
Partition
Topic Y
Partition
Partition
Producer A
Producer B
Producer C
Consumer Group A
Consumer 1
Consumer 2
Consumer Group B
Consumer 1
Consumer 2
Get Kafka Broker ID
Kafka
Zookeeper
Update Offset

HOW IS KAFKA USED IN BILLING SYSTEM



- The billing system uses Kafka for real-time data transmission and storage of resource usage information from multiple machines, with cost calculation and visualization in real-time on a dashboard.
- Kafka uses Zookeeper to manage its brokers.
- Kafka here acts as a buffer and transport for real-time resource usage data from each machine to the billing and dashboard components.
- Kafka enables workers to consume real-time data as it’s published, making it possible to compute costs promptly and accurately.
- Kafka enables the dashboard to pull data continuously, allowing visualization to reflect real-time usage across machines.

IMPLEMENTATION

- **Command to start Zookeeper:**
  java cp"libs/*;config"org.apache.zookeeper.server.ZooKeeperServerMainconfig\zookeeper.properties

- **Command to start Kafka Server:**
  java -cp "libs/*;config" kafka.Kafka config\server.properties

- **Command to create topic :**
  bin/kafka-topics.sh --create --topic resource_usage --bootstrap-server localhost:9092 -partitions 3 --replication-factor 1

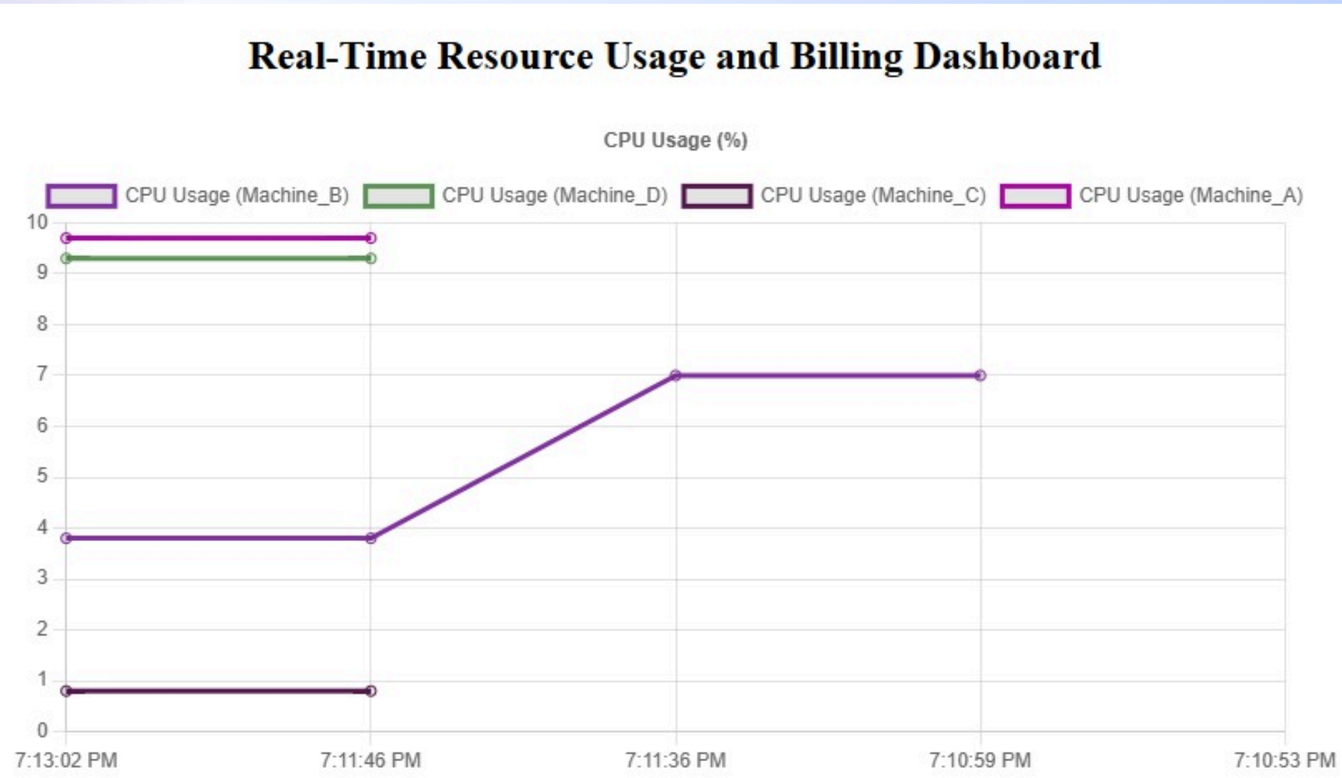
SYSTEM WORKFLOW

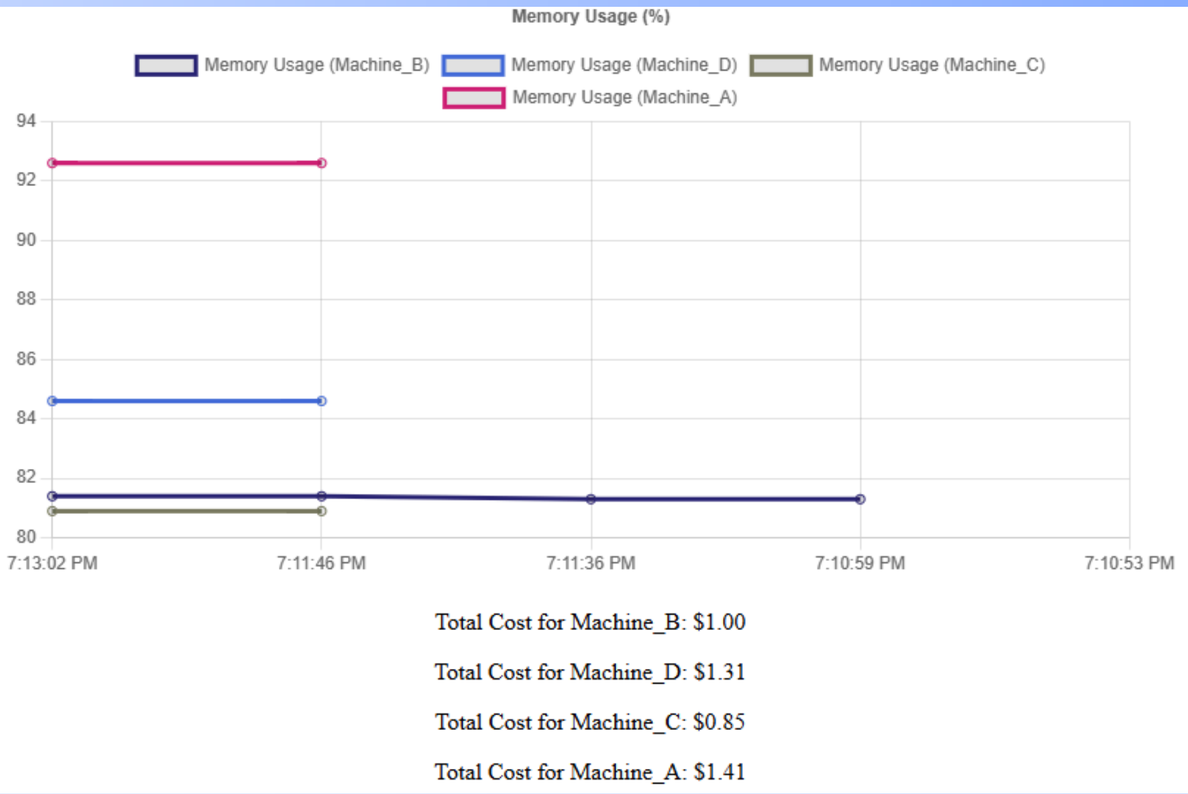
1.Data Collection

2.Data Transmission

3.Data Consumption

DASHBOARD REAL-TIME DATA





- Provides a live view of CPU usage data from multiple machines in the distributed system.
- Tracks CPU usage for each machine, updated in real-time as new data arrives.

BENEFITS OF KAFKA

- Scalability
- Fault Tolerance
- Real-Time Processing
- Decoupling of Components

SYSTEM

BENEFITS AND SCALABILITY

EFFICIENCY IN RESOURCE ALLOCATION

REDUCED SYSTEM OVERHEAD

ENHANCED RELIABILITY

IMPROVED USER CONTROL

FUTURE IMPROVEMENTS

PREDICTIVE BILLING WITH MACHINE LEARNING

CLOUD-BASED KAFKA INTEGRATION

MULTI-TIERED BILLING OPTIONS

ENHANCED DATA ANALYTICS AND REPORTING

THANK YOU!

Any Questions?