

Lab: EDGE COMPUTING LAB (23A39605)

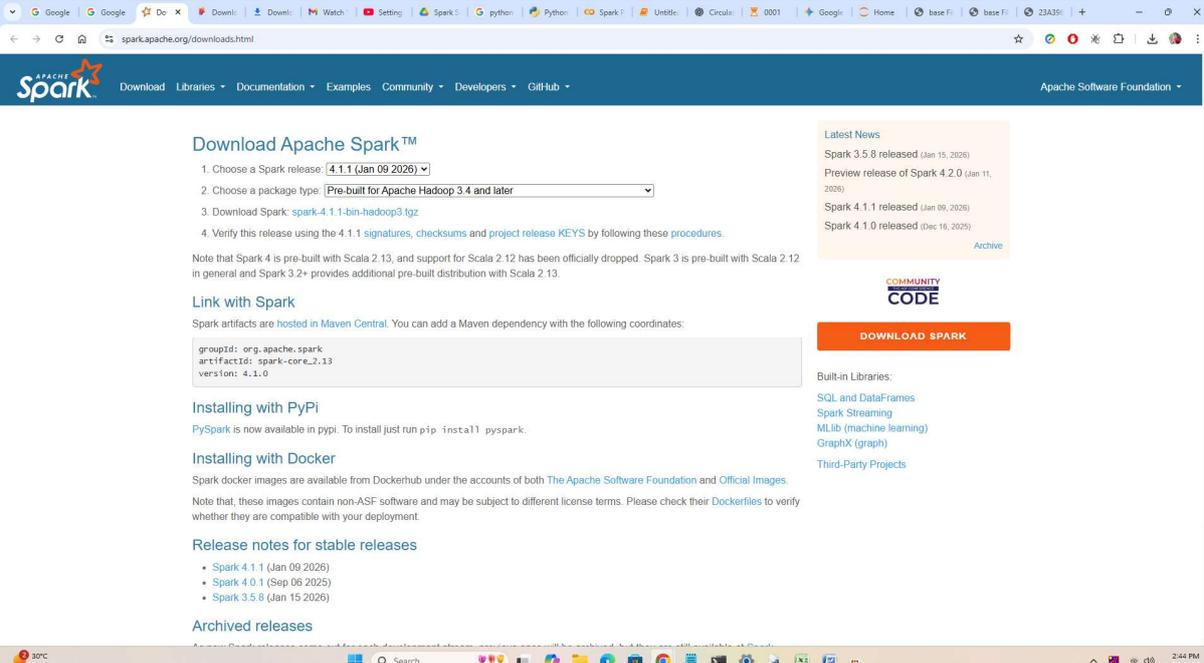
College: Andhra Engineering College

9. Streaming Data Analytics on Edge o Local aggregation and event processing with Kafka or lightweight alternatives

Download

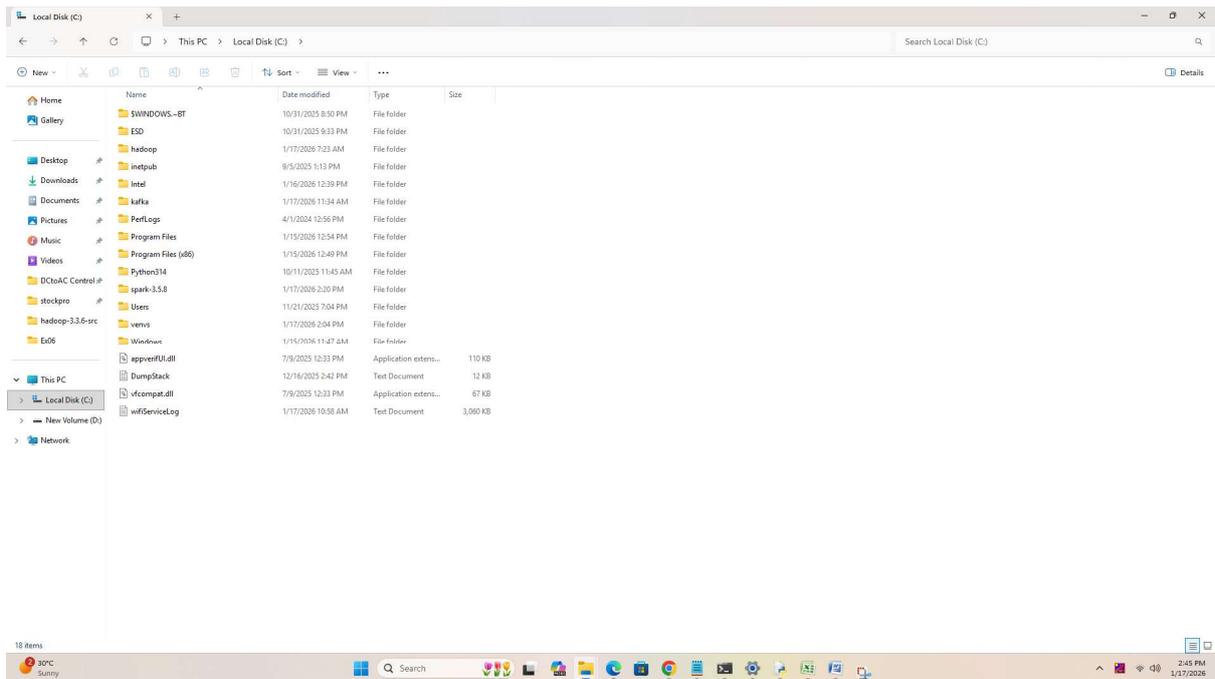
<https://spark.apache.org/downloads.html>

<https://kafka.apache.org/community/downloads/>



The screenshot shows the Apache Spark download page. The main heading is "Download Apache Spark™". Below it, there are four numbered steps: 1. Choose a Spark release (4.1.1 (Jan 09 2026)), 2. Choose a package type (Pre-built for Apache Hadoop 3.4 and later), 3. Download Spark (spark-4.1.1-bin-hadoop3.tgz), and 4. Verify this release using the 4.1.1 signatures, checksums and project release KEYS by following these procedures. A note states that Spark 4 is pre-built with Scala 2.13, and support for Scala 2.12 has been officially dropped. Below the steps, there is a section "Link with Spark" with Maven coordinates: groupId: org.apache.spark, artifactId: spark-core_2.13, version: 4.1.0. There are also sections for "Installing with PyPi" (PySpark is now available in pypi. To install just run pip install pyspark.), "Installing with Docker" (Spark docker images are available from Dockerhub under the accounts of both The Apache Software Foundation and Official Images. Note that, these images contain non-ASF software and may be subject to different license terms. Please check their Dockerfiles to verify whether they are compatible with your deployment.), "Release notes for stable releases" (Spark 4.1.1 (Jan 09 2026), Spark 4.0.1 (Sep 06 2025), Spark 3.5.8 (Jan 15 2026)), and "Archived releases". On the right side, there is a "Latest News" section with links to Spark 3.5.8 released (Jan 15, 2026), Preview release of Spark 4.2.0 (Jan 11, 2026), Spark 4.1.1 released (Jan 09, 2026), and Spark 4.1.0 released (Dec 16, 2025). Below this is the "COMMUNITY CODE" logo and a "DOWNLOAD SPARK" button. At the bottom, there is a "Built-in Libraries" section with links to SQL and DataFrames, Spark Streaming, MLlib (machine learning), GraphX (graph), and Third-Party Projects.

After Download Extract and Past on C:



Install Java 17

After

Set Env Variable on **system path**

```
JAVA_HOME      C:\Program Files\Java\jdk-17
HADOOP_HOME    C:\hadoop
SPARK_HOME     C:\spark-3.5.8
path           %HADOOP_HOME%\bin
              %SPARK_HOME%\bin
              %JAVA_HOME%\bin
              %KAFKA_HOME%\bin

KAFKA_HOME     C:\kafka
```

Next Go to C:

```
C:\kafka\config
CHANGE HERE server file
```

```
log.dirs=/tmp/kraft-combined-logs
to
log.dirs=C:/kafka/kraft-combined-logs
```

Production Point of view change Here

Server Basics

```
process.roles=broker,controller
node.id=1
```

```
controller.quorum.bootstrap.servers=10.208.87.140:9093
```

Socket Server Settings

```
# Bind to all interfaces
listeners=PLAINTEXT://0.0.0.0:9092,CONTROLLER://0.0.0.0:9093
```

```
# Only advertise broker listener (VERY IMPORTANT)
advertised.listeners=PLAINTEXT://10.208.87.140:9092
```

```
# Required for KRaft
controller.listener.names=CONTROLLER
inter.broker.listener.name=PLAINTEXT
```

```
listener.security.protocol.map=CONTROLLER:PLAINTEXT,PLAINTEXT:PLAINTEXT
```

Log Basics

```
log.dirs=C:/kafka/kraft-combined-logs
num.partitions=1
```

Internal Topics

```
offsets.topic.replication.factor=1
transaction.state.log.replication.factor=1
transaction.state.log.min.isr=1
```

Go to this Path C:\kafka

```
C:\kafka>. \bin\windows\kafka-storage.bat random-uuid
```

```
2026-01-17T05:45:46.755244Z main ERROR
Reconfiguration failed: No configuration found for
'266474c2' at 'null' in 'null'
RDowQE9sSI22VfkPmBWLlw
```

```
C:\kafka>. \bin\windows\kafka-storage.bat format --standalone -t RDowQE9sSI22VfkPmBWLlw
-c config\server.properties
```

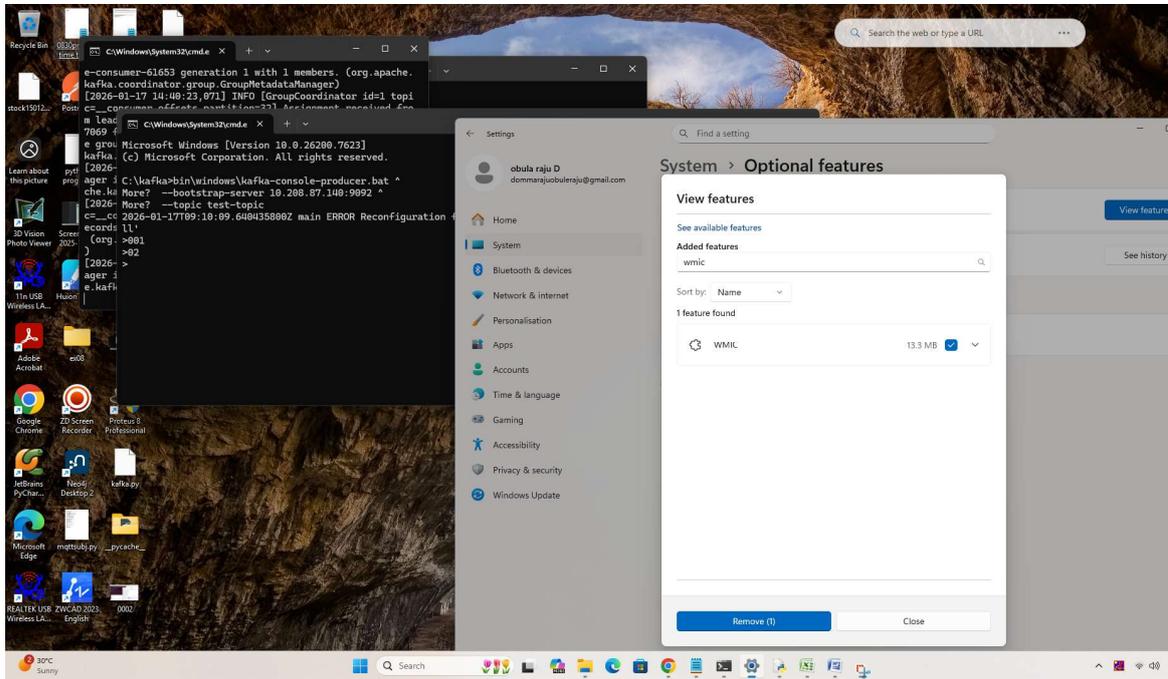
```
2026-01-17T05:47:33.696916700Z main ERROR Reconfiguration failed: No configuration found
for '266474c2' at 'null' in 'null'
Formatting dynamic metadata voter directory C:/kafka/kraft-combined-logs with
metadata.version 4.1-IV1.
```

command line utility function

settings->system->optional features ->view features

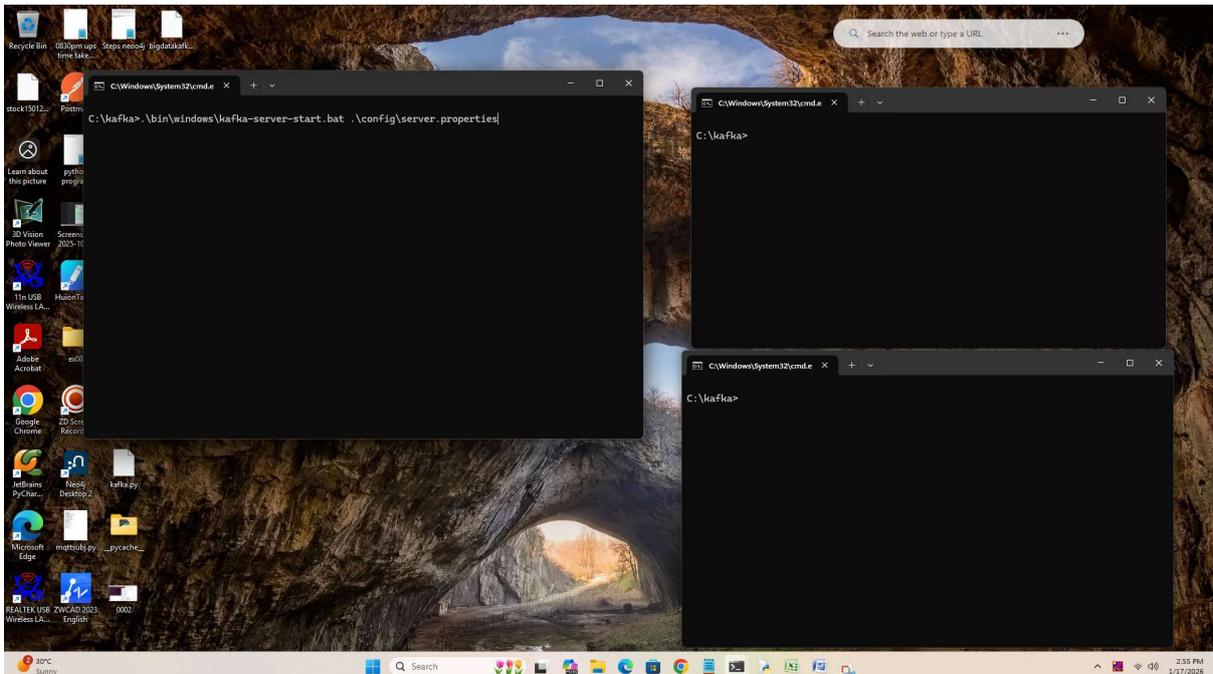
Search

wmic add it



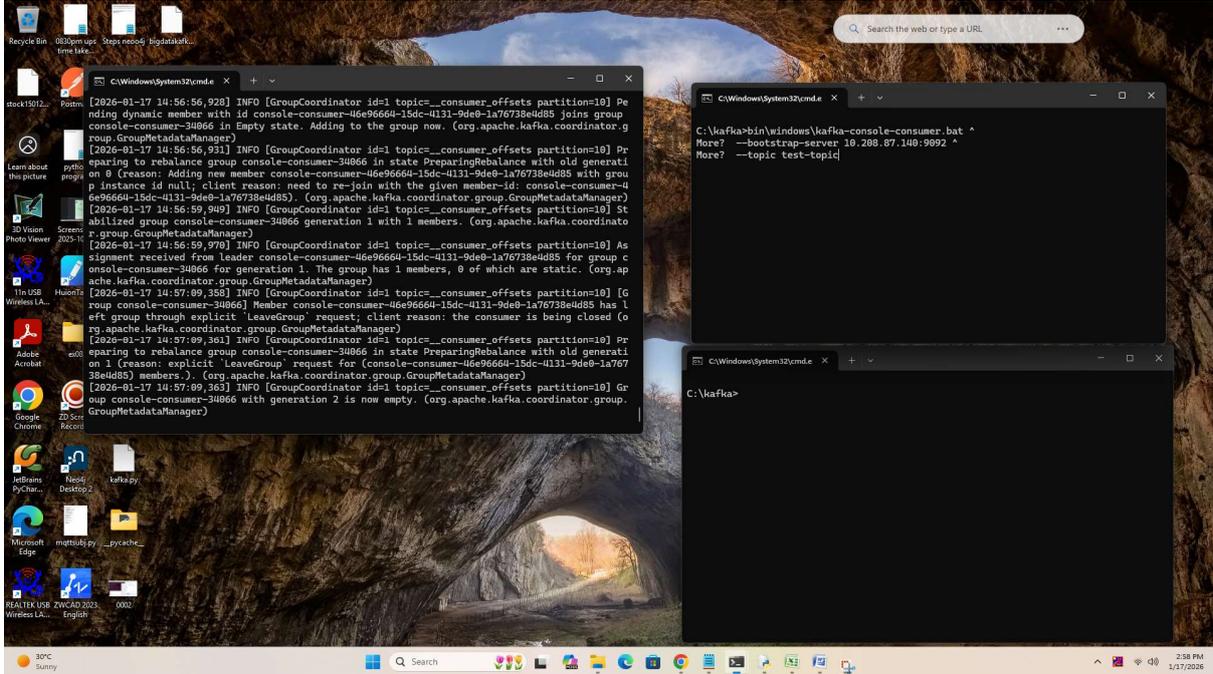
Start Kafka

C:\kafka>.\bin\windows\kafka-server-start.bat .\config\server.properties



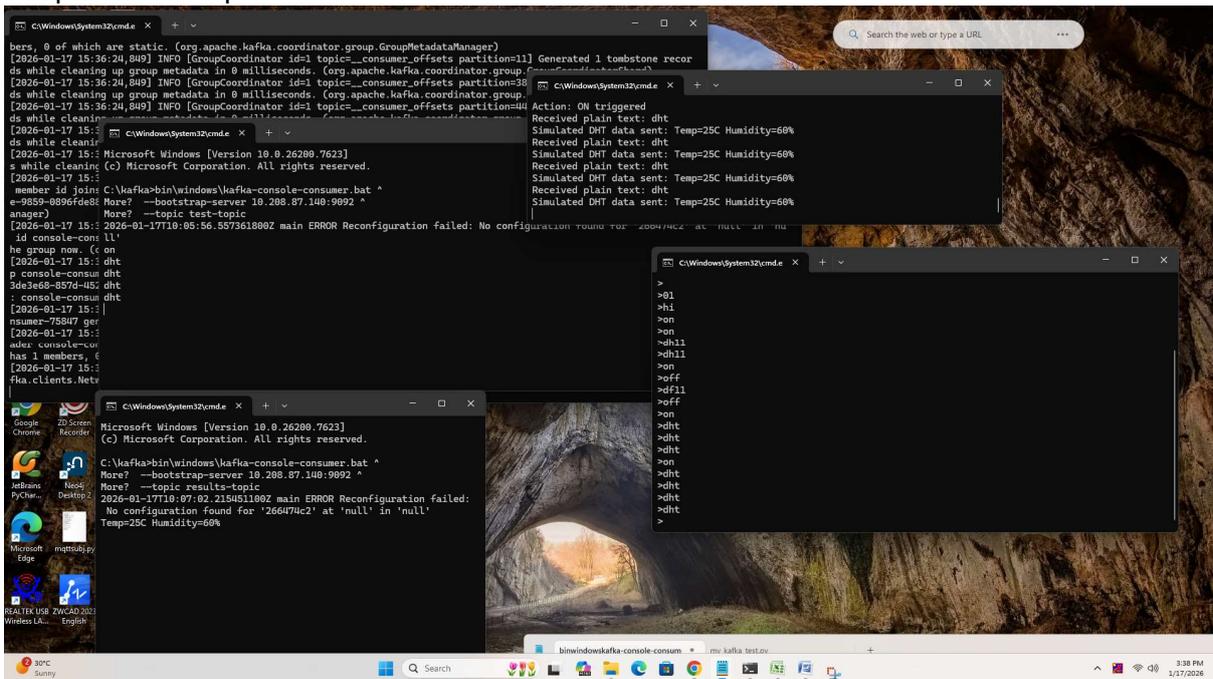
Consumer (new CMD window)

```
bin\windows\kafka-console-
consumer.bat ^
--bootstrap-server
10.208.87.140:9092 ^
--topic test-topic ^
--from-beginning
```



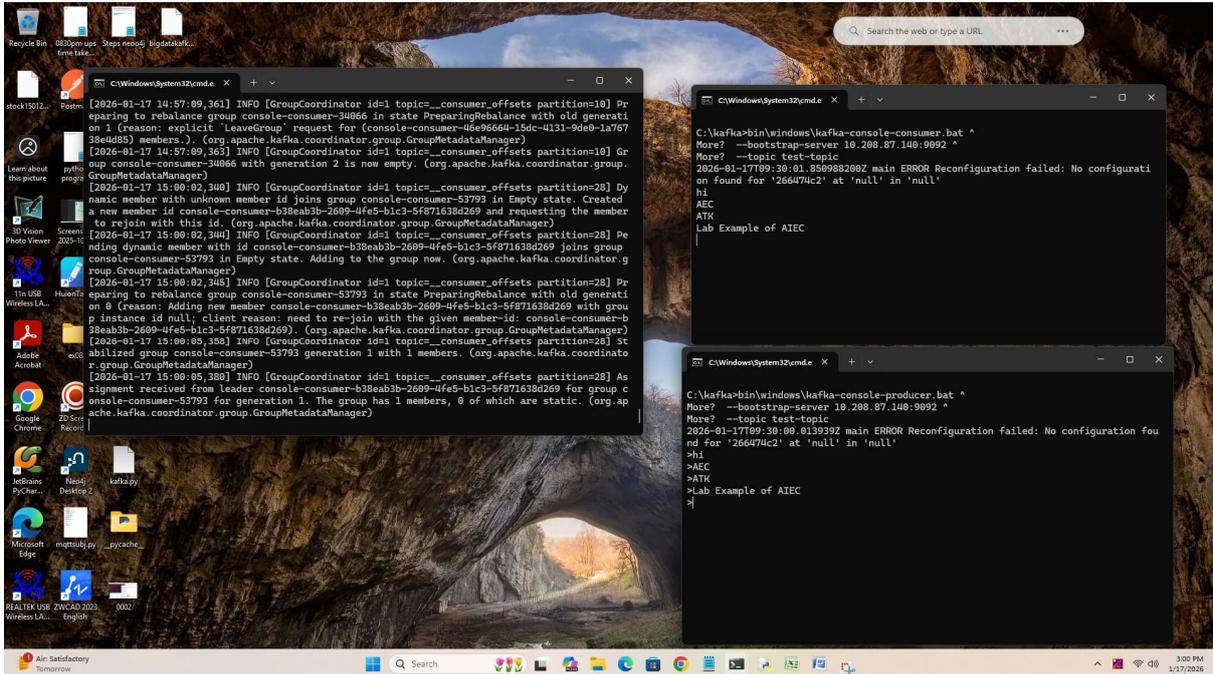
Consumer 02

```
bin\windows\kafka-console-consumer.bat ^
--bootstrap-server 10.208.87.140:9092 ^
--topic results-topic
```



Producer

```
bin\windows\kafka-console-producer.bat ^
--bootstrap-server 10.208.87.140:9092 ^
--topic test-topic
```



Using Edge End Control

```
C:\spark-3.5.8>pip install kafka-python
```

Collecting kafka-python

Downloading kafka_python-2.3.0-py2.py3-none-any.whl.metadata (10.0 kB)

Downloading kafka_python-2.3.0-py2.py3-none-any.whl (326 kB)

Installing collected packages: kafka-python

Successfully installed kafka-python-2.3.0

[notice] A new release of pip is available: 24.3.1 -> 25.3

[notice] To update, run: python.exe -m pip install --upgrade pip

Python Code

```
from kafka import KafkaConsumer, KafkaProducer
import json
```

```
# ----- CONFIG -----
```

```
KAFKA_BOOTSTRAP = "10.208.87.140:9092"
```

```
INPUT_TOPIC = "test-topic"
```

```
RESULT_TOPIC = "results-topic"
```

```
# ----- PRODUCER -----
```

```

producer = KafkaProducer(
    bootstrap_servers=KAFKA_BOOTSTRAP,
    value_serializer=lambda x: json.dumps(x).encode("utf-8")
        if isinstance(x, dict) else str(x).encode("utf-8"),
    acks="all",
    retries=3
)

# ----- SAFE PARSER -----
def parse_message(x):
    try:
        return json.loads(x.decode("utf-8"))
    except Exception:
        return x.decode("utf-8")

# ----- PROCESSING -----
def on_message(msg):

    # ----- JSON messages -----
    if isinstance(msg, dict):
        event = msg.get("event")

        if event == "predict":
            data = msg.get("data", [])
            if not data:
                return

            result = sum(data) / len(data)
            producer.send(
                RESULT_TOPIC,
                {"event": "prediction", "value": result}
            )
            print("JSON prediction:", result)

        else:
            print("Unknown JSON:", msg)

    # ----- Plain text messages -----
    else:
        payload = msg.strip().lower()
        print("Text received:", payload)

        if payload == "on":
            producer.send(RESULT_TOPIC, "Action: ON triggered")

        elif payload == "off":
            producer.send(RESULT_TOPIC, "Action: OFF triggered")

```

```

elif payload == "dht":
    temp, hum = 25, 60
    producer.send(
        RESULT_TOPIC,
        f"Temp={temp}C Humidity={hum}%"
    )

else:
    producer.send(
        RESULT_TOPIC,
        f"Message received: {payload}"
    )

producer.flush()

# ----- CONSUMER -----
consumer = KafkaConsumer(
    INPUT_TOPIC,
    bootstrap_servers=KAFKA_BOOTSTRAP,
    group_id="mixed-group-v2",
    value_deserializer=parse_message,
    auto_offset_reset="earliest",
    enable_auto_commit=True
)

# ----- RUN -----
print("Kafka consumer started...")
for msg in consumer:
    if msg.value is not None:
        on_message(msg.value)

```

