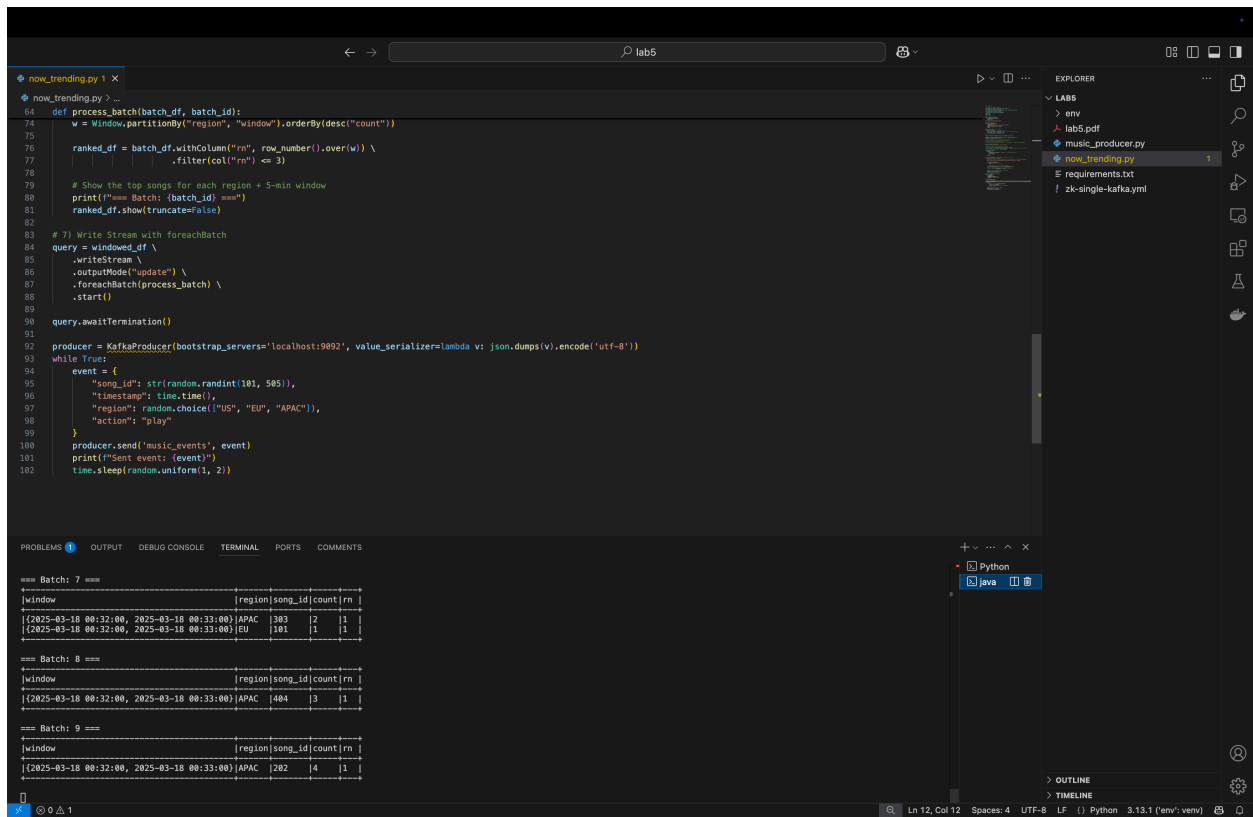


ADD/ SKIP Extension



```
now_trending.py > ...
64 def process_batch(batch_df, batch_id):
65     w = Window.partitionBy("region", "window").orderBy(desc("count"))
66     ranked_df = batch_df.withColumn("rn", row_number().over(w)) \
67         .filter(col("rn") <= 3)
68     # Show the top songs for each region + 5-min window
69     print(f"=== Batch: {batch_id} ===")
70     ranked_df.show(truncate=False)
71
72 # 7) Write Stream with foreachBatch
73 query = windowed_df \
74     .writeStream \
75     .outputMode("update") \
76     .foreachBatch(process_batch) \
77     .start()
78
79 query.awaitTermination()
80
81 producer = KafkaProducer(bootstrap_servers='localhost:9092', value_serializer=lambda v: json.dumps(v).encode('utf-8'))
82 while True:
83     event = {
84         "song_id": str(random.randint(101, 505)),
85         "timestamp": time.time(),
86         "region": random.choice(["US", "EU", "APAC"]),
87         "action": "play"
88     }
89     producer.send('music_events', event)
90     print(f"Sent event: {event}")
91     time.sleep(random.uniform(1, 2))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
=== Batch: 7 ===
+-----+
|window|region|song_id|count|rn|
+-----+-----+-----+-----+-----+
|[2025-03-18 00:32:00, 2025-03-18 00:33:00]|APAC|383|12|1|
|[2025-03-18 00:32:00, 2025-03-18 00:33:00]|EU|101|11|1|
+-----+-----+-----+-----+-----+

=== Batch: 8 ===
+-----+
|window|region|song_id|count|rn|
+-----+-----+-----+-----+-----+
|[2025-03-18 00:32:00, 2025-03-18 00:33:00]|APAC|404|13|1|
+-----+-----+-----+-----+-----+

=== Batch: 9 ===
+-----+
|window|region|song_id|count|rn|
+-----+-----+-----+-----+-----+
|[2025-03-18 00:32:00, 2025-03-18 00:33:00]|APAC|202|14|1|
+-----+-----+-----+-----+-----+
```

Ln 12, Col 12 Spaces: 4 UTF-8 LF 1 Python 3.13.1 (env: venv)

1 Minute Extension

The screenshot shows a JupyterLab environment with a Python script named `now_trending.py` and its execution output in the terminal.

Script Details:

- Imports:** `from pyspark.sql.functions import current_timestamp`
- Initial Setup:** The script filters for "play" events and groups them by region and song_id.
- Windowing:** A processing-time window of 1 minute is used, starting from `current_timestamp()`.
- Rank-based Logic:** The `process_batch` function uses `Window.partitionBy("region", "window").orderBy(desc("count"))` to rank songs by count within each region and window.
- Output:** The top 3 songs (ranked 1, 2, and 3) are selected for each batch.

Terminal Output:

```
==== Batch: 18 ====
|window|region|song_id|count|rn|
|-----|-----|-----|-----|---|
|{2025-03-18 00:27:00, 2025-03-18 00:28:00}|APAC|585|13|1|

==== Batch: 19 ====
|window|region|song_id|count|rn|
|-----|-----|-----|-----|---|
|{2025-03-18 00:27:00, 2025-03-18 00:28:00}|US|383|12|1|

==== Batch: 20 ====
|window|region|song_id|count|rn|
|-----|-----|-----|-----|---|
|{2025-03-18 00:27:00, 2025-03-18 00:28:00}|US|484|14|1|
|{2025-03-18 00:27:00, 2025-03-18 00:28:00}|EU|383|11|1|
```

5 Minutes Extension

The screenshot displays a JupyterLab environment with a Python script named `now_trending.py` and its output in the terminal.

Script Details:

- Imports:** `from pyspark.sql.functions import current_timestamp`
- Filtering:** The script filters for "play" events: `plays_df = events_df.filter(col("action") == "play")`
- Windowing:** A processing-time window of 5 minutes is defined: `window(current_timestamp(), "5 minutes")`. The data is grouped by region and song_id.
- Micro-batch Processing:** The `process_batch` function is called for each micro-batch. It partitions the data by region and window, then ranks songs by count within each partition.
- Output:** The script prints the top 3 songs in each region for each batch.

Terminal Output:

```
Batch: 51
window
+-----+-----+-----+-----+
|region|song_id|count|rn|
+-----+-----+-----+-----+
|{2025-03-18 00:25:00, 2025-03-18 00:30:00}|EU|505|2|
|{2025-03-18 00:25:00, 2025-03-18 00:30:00}|EU|484|1|
+-----+-----+-----+-----+

Batch: 52
window
+-----+-----+-----+-----+
|region|song_id|count|rn|
+-----+-----+-----+-----+
|{2025-03-18 00:25:00, 2025-03-18 00:30:00}|US|202|1|
+-----+-----+-----+-----+

Batch: 53
window
+-----+-----+-----+-----+
|region|song_id|count|rn|
+-----+-----+-----+-----+
|{2025-03-18 00:25:00, 2025-03-18 00:30:00}|EU|505|3|
|{2025-03-18 00:25:00, 2025-03-18 00:30:00}|US|484|1|
+-----+-----+-----+-----+
```