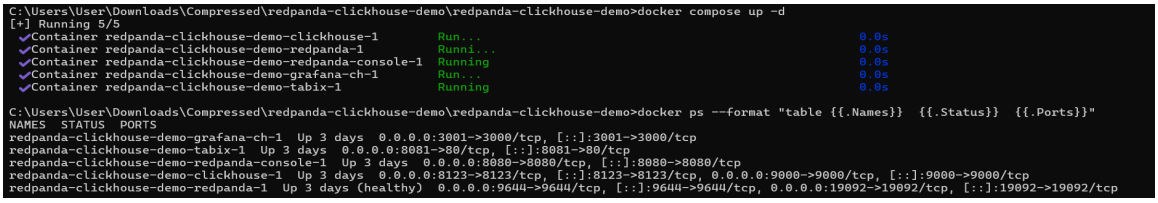
Redpanda + ClickHouse ile Gerçek Zamanlı Anomali Tespiti ve LLM Yorumlama

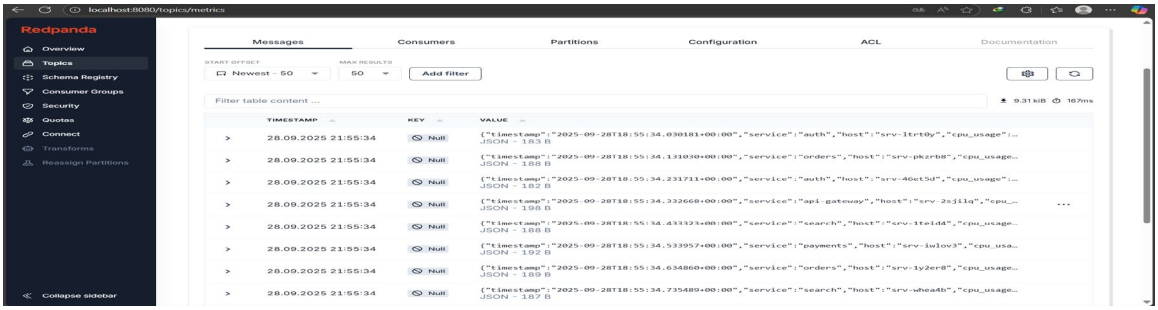
1) Hedef

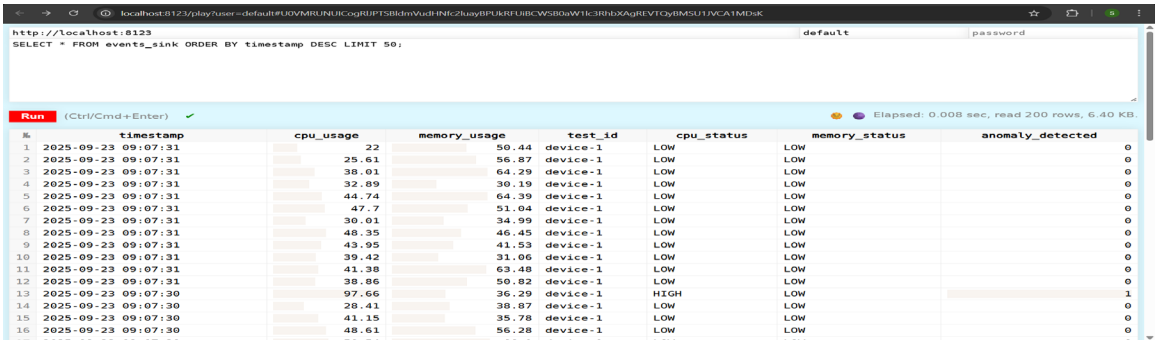
Redpanda (Kafka uyumlu) ve ClickHouse kullanarak gerçek zamanlı metrikleri almak, kalıcı hale getirmek, Grafana ile görselleştirmek; anomalies (eşik + istatistiksel) tespit edip bu olaylara bir LLM (Ollama) ile kısa açıklama/öneri ürettirmek ve sonuçları ClickHouse'ta alerts tablosunda saklayıp Grafana'da göstermek.

2) Ne Yapıldı

- Docker Compose ile Redpanda, ClickHouse, Tabix ve Grafana'yı ayağa kaldırdık.







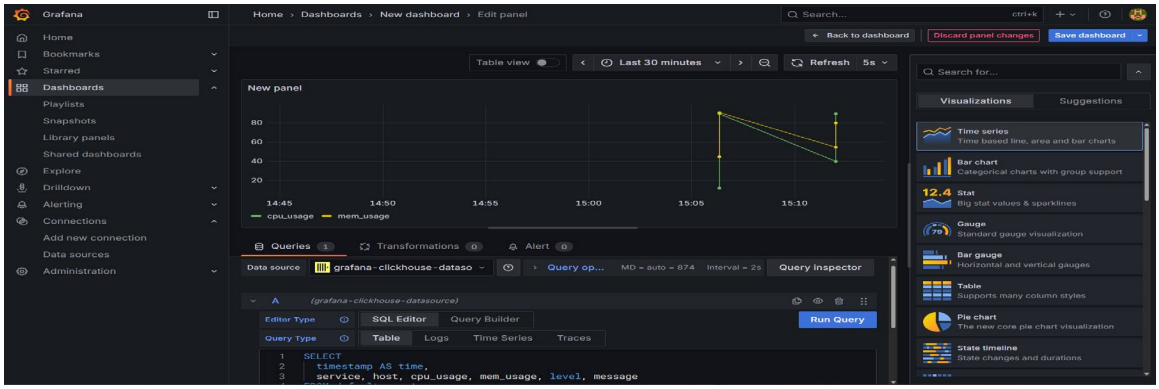
- ClickHouse'ta Kafka Engine → Materialized View ile akıştan 'events' tablosuna ingest ettik.
- Grafana'da CPU/Mem zaman serileri ve ham olay tabloları oluşturduk.
- Python 'anomaly_worker.py' ile threshold + z-score tabanlı anomalileri tespit ettik.

```
C:\Users\User\Downloads\Compressed\redpanda-clickhouse-demo\redpanda-clickhouse-demo>.\.venv\Scripts\python.exe -u .\anomaly_worker.py --brokers localhost:1
9092 --topic metrics --model "phi3:mini" --threshold 70 --z 2.5 --window 50 --clickhouse localhost:8123 --ch_user default --ch_pass chpass
[worker] consuming metrics from localhost:19092 model=phi3:mini
[worker] ollama warmed
[anomaly] cpu>70.0,mem>70.0 svc=demo cpu=96.0 mem=92.0
[worker] alerts flushed to ClickHouse
[anomaly] |z_cpu|>=2.5 svc=search cpu=38.09 mem=40.39
[worker] alerts flushed to ClickHouse
[anomaly] |z_mem|>=2.5 svc=orders cpu=37.06 mem=69.49
[worker] alerts flushed to ClickHouse
[anomaly] |z_cpu|>=2.5 svc=api-gateway cpu=10.39 mem=23.53
```

- Ollama (phi3:mini) üzerinden her anomali için kısa açıklama/öneri ürettik.

```
C:\Users\User\Downloads\Compressed\redpanda-clickhouse-demo\redpanda-clickhouse-demo>docker exec -it redpanda-clickhouse-demo-clickhouse-1 clickhouse-client
 -u default --password chpass -q "SELECT toString(toTimeZone(timestamp,'Europe/Istanbul')) AS time, service, rule, round(score,2) AS score, left(explanation
,200) AS explanation FROM default.alerts WHERE timestamp > now() - INTERVAL 15 MINUTE ORDER BY timestamp DESC LIMIT 20"
2025-09-28 21:57:35.652 orders  |z_mem|>=2.5    4.69    * The event is not concerning as the CPU and memory usage are within reasonable limits (37%-100%, wh
ere higher than usual does not necessarily indicate an issue).\n* Suggested Actions:\n    * Continue m
2025-09-28 21:57:14.301 search  |z_cpu|>=2.5    17.55   Concerning: No\n- The CPU and memory usage seem to be within an acceptable range given the context o
f processing a search request (assuming typical service behavior).\nActions Suggested:\n1. Monitor reso
2025-09-28 21:56:51.746 demo    cpu>70.0,mem>70.0       0       Concerning Event: High CPU and memory usage on \'demo\' host by manual trigger leading to an
 error event which violates the rule set for acceptable thresholds.\n- Investigate if there is unnecessary back
2025-09-28 21:47:19.722 api-gateway     cpu>70.0,mem>70.0,|z_cpu|>=2.5,|z_mem|>=2.5     6.62    - Event is concerning as it indicates that the API Gateway s
ervice on host \'srv-v7ooti\' has exceeded CPU and memory thresholds set in a rule indicating high anomaly potential for resource consumption.
2025-09-28 21:46:47.852 orders  cpu>70.0,mem>70.0,|z_cpu|>=2.5,|z_mem|>=2.5     7.18    - Concerning: Yes\n- Actions suggested:\n  1. Immediately start an i
nvestigation to understand the root cause of high CPU/MEM usage and implement a fix if possible (e.g., optimize queries or scale up re
2025-09-28 21:46:08.044 search  cpu>70.0,mem>70.0,|z_cpu|>=2.5  4.88    - This event is concerning as it indicates the service \'search\' on host \'srv-o8fj
p8\' has high CPU and memory usage exceeding set thresholds with abnormal z-score deviation for CPU load. Actions to tak
2025-09-28 21:44:58.086 demo    cpu>70.0,mem>70.0       0       - **Event is concerning**: The event shows a high CPU and memory usage that exceeds the pred
efined threshold of 70% each for both resources; this indicates an immediate performance issue which could i
```

- Sonuçları ClickHouse 'default.alerts' tablosuna yazdık; Grafana'da 'Anomali Listesi' ve 'Alerts per minute' panellerini gösterdik.



3) Mimari

Producer → Redpanda (topic: metrics)
→ ClickHouse (Kafka Engine + MV → events)
→ Grafana (Time series + Table panelleri)
→ Python Anomaly Worker (threshold + z-score)
→ Ollama (LLM açıklaması)
→ ClickHouse (alerts) → Grafana (Anomali Listesi & Alerts/min)

4) Kullanılan Teknolojiler

Bileşen	Amaç	Not
Redpanda	Kafka uyumlu event streaming broker	Düşük gecikme, basit kurulum
ClickHouse	OLAP veritabanı, Kafka ingest + analitik SQL	MergeTree, MV, hızlı sorgu
Grafana	Görselleştirme ve dashboard	Time series & Table panelleri
Tabix	ClickHouse'a basit web SQL istemcisi	Hızlı test/SQL
Python	Anomaly worker ve producer	kafka-python, requests
Ollama (phi3:mini)	Yerel LLM ile kısa açıklama/öneri	11434 REST API, hızlı ve lokal

5) Uygulama Adımları

1) Docker Compose ile Redpanda, ClickHouse, Tabix, Grafana çalıştırıldı; port çakışmaları çözüldü (9092/9644 vb).
2) ClickHouse'ta Kafka Engine kaynak tablo (raw_events_kafka) ve materialized view (mv_consume_metrics) ile events tablosuna insert akışı sağlandı.
3) Grafana ClickHouse datasource eklendi; Time series (CPU/Mem) ve Table (ham events) panelleri kuruldu.
4) anomaly_worker.py: metrics topiğini tüketip eşik ve z-score (rolling window) ile anomali tespiti yaptı.
5) Ollama REST API: her anomali için kısa açıklama/öneri üretildi; timeout ve retry ile güvenilirlik artırıldı.
6) ClickHouse alerts tablosuna JSONEachRow ile batch insert; 'best_effort' datetime parse ile uyumluluk sağlandı.
7) Grafana'da 'Anomali Listesi' ve 'Alerts per Minute' panelleri oluşturuldu.

6) anomaly_worker.py – Ne Yapıyor?

6.1 Giriş / Çıkış

Giriş: Redpanda'daki 'metrics' topiğinden JSON mesajlar (timestamp, service, host, cpu_usage, mem_usage, level, message).

Çıkış: Anomali tespit edilen olaylar için, LLM açıklaması ve skorla birlikte ClickHouse ‘default.alerts’ tablosuna kayıt.

6.2 Tespit Mantığı

• Threshold: cpu_usage > threshold veya mem_usage > threshold ise tetiklenir.
• Z-score (Rolling): Her servis için ayrı rolling pencere (örn. 100). Yeni değerin z-skoru |z| ≥ z_eşiği ise anomali.
• Kural birleştirme: Tetiklenen kurallar virgülle birleştirilir (örn. ‘cpu>70,|z_mem|≥2.5’).

6.3 LLM Entegrasyonu (Ollama)

• HTTP POST /api/generate: model=phi3:mini, prompt içinde olay JSON’u.
• İlk isteklerde model yükleme süresine karşı warm-up + retry + uzun read timeout kullanıldı.
• Yanıt ‘response’ alanından alınır ve explanation sütununa yazılır.

6.5 Örnek Kod Parçaları

Rolling z-score

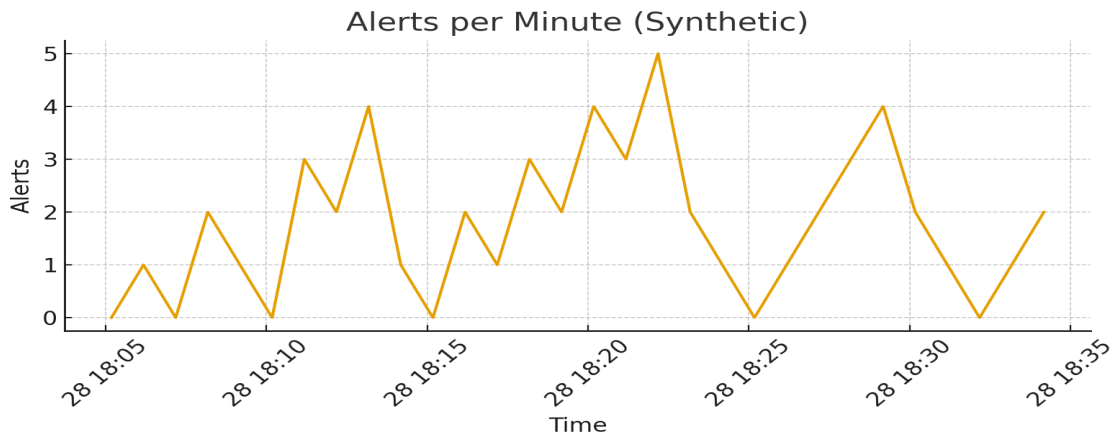
```
class Rolling:
    def __init__(self, maxlen=100):
        self.values = collections.deque(maxlen=maxlen)
    def add(self, x):
        self.values.append(float(x))
    def z(self, x):
        s = statistics.pstdev(self.values) if len(self.values)>1 else 0.0
        m = statistics.mean(self.values) if self.values else 0.0
        return 0.0 if s==0 else (float(x)-m)/s
```

LLM çağrısı (retry + timeout)

```
def call_ollama(model, payload):
    url = "http://localhost:11434/api/generate"
    body = {"model": model, "prompt": prompt_from(payload), "stream": False}
    for attempt in range(3):
        try:
            r = requests.post(url, json=body, timeout=(5,120))
            r.raise_for_status()
            return (r.json() or {}).get("response","").strip()
        except Exception as e:
            time.sleep(2*(attempt+1))
    return "(LLM unavailable)"
```

7) Görseller ve Örnek Çıktılar

Alerts per Minute



8) Kıyas: Apache Kafka + Flink vs Redpanda + ClickHouse

Kriter	Kafka + Flink (Artılar)	Redpanda + ClickHouse (Artılar)	Dikkat/Limitler
Gerçek zamanlı işleme	Event-time, watermark, CEP, stateful ops çok güçlü	SQL + MV ile hızlı agregasyon, düşük operasyon yükü	Flink yoksa karmaşık pattern'lar ek uygulama ister
Operasyon karmaşıklığı	Güçlü ama cluster yönetimi zor olabilir	Compose ile hızlı; broker+DB yeterli	Yine de prod için izleme/backup/TTL şart
Maliyet/Kaynak	Flink cluster ekstra kaynak ve bakım	Daha az bileşen ⇒ daha düşük operasyon maliyeti	Yüksek ingest hızı için CH tuning gerekebilir
Analitik	Harici OLAP/BI gerekir	CH çok hızlı analitik; Grafana/Tabix hazır	Gelişmiş ML/feature pipeline için ek araç gerekir
Kullanım örüntüsü	Karmaşık stream pipeline, CEP, join	Gözlemleme, telemetri, log/metric analitiği	State machine/CEP gereken yerlerde sınırlı

9) Neyi İyi Yaptık / Neleri İyileştirebiliriz

Güçlü Yönler:

- Hızlı kurulum ve uçtan uca akışın devreye alınması
- Anomali tespiti için hem eşik hem z-score kullanımı
- LLM entegrasyonunda warm-up + retry + uzun timeout ile dayanıklılık
- ClickHouse insertlerinde best_effort tarih parse ve batch yazım
- Grafana'da anlamlı paneller: liste + dakika başı adet

İyileştirme Alanları:

- İlk ısınmada LLM time-out yaşandı (çözüldü)
- Port çakışmaları ve Tabix bağlantı ayarlarında zaman kaybı
- Producer hızı ile worker eşikleri uyumlandırma ihtiyacı
- Kalıcı servisleştirme (Compose) henüz yapılmadı

Grafana SQL (alerts per minute)

```
-- Anomali / dakika grafiği
SELECT toStartOfMinute(timestamp) AS time, count() AS alerts_per_min
FROM default.alerts
WHERE $__timeFilter(timestamp)
GROUP BY time
ORDER BY time;
```

Grafana SQL (Anomali Listesi)

```
-- LLM açıklamalı liste
SELECT timestamp AS time, service, host, cpu_usage, mem_usage, rule,
round(score,2) AS score, explanation
FROM default.alerts
WHERE $__timeFilter(timestamp)
ORDER BY time DESC
LIMIT 200;
```

Komutlar (çalıştırma + tetikleme)

```
# Worker'ı çalıştırma (PowerShell)
.\.venv\Scripts\python.exe -u .\anomaly_worker.py --brokers localhost:19092 --
topic metrics --model "phi3:mini" --threshold 70 --z 2.5 --window 50 --
clickhouse localhost:8123 --ch_user default --ch_pass chpass
```

```
# Tetikleyici anomali gönderme
.\.venv\Scripts\python.exe -c "from kafka import KafkaProducer; import json;
p=KafkaProducer(bootstrap_servers='localhost:19092', value_serializer=lambda v:
json.dumps(v).encode()); p.send('metrics', {'timestamp':'2025-09-28
12:34:56.789','service':'demo','host':'manual','cpu_usage':96,'mem_usage':92,'l
evel':'ERROR','message':'manual spike'}); p.flush(); print('sent')"
```