Event collection framework @ Blinkit

Why?

We are ‘event’ driven

How?

Aggregation of ‘events’

What?

Observability

Gotta catch ém all !
BACKEND EVENTS
FRONTEND EVENTS
imgflip.com

Frontend Events

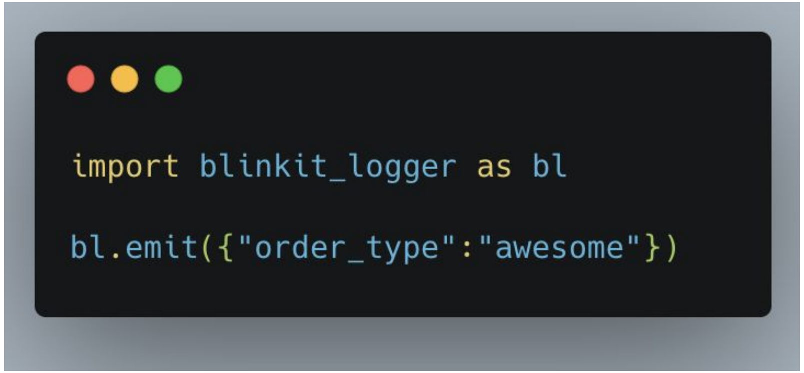
Clicks and Impressions

Backend Events

- Across Services (Pub-Sub)
- Localised to Services (Logs)

Logs -> Aggregator -> Kafka

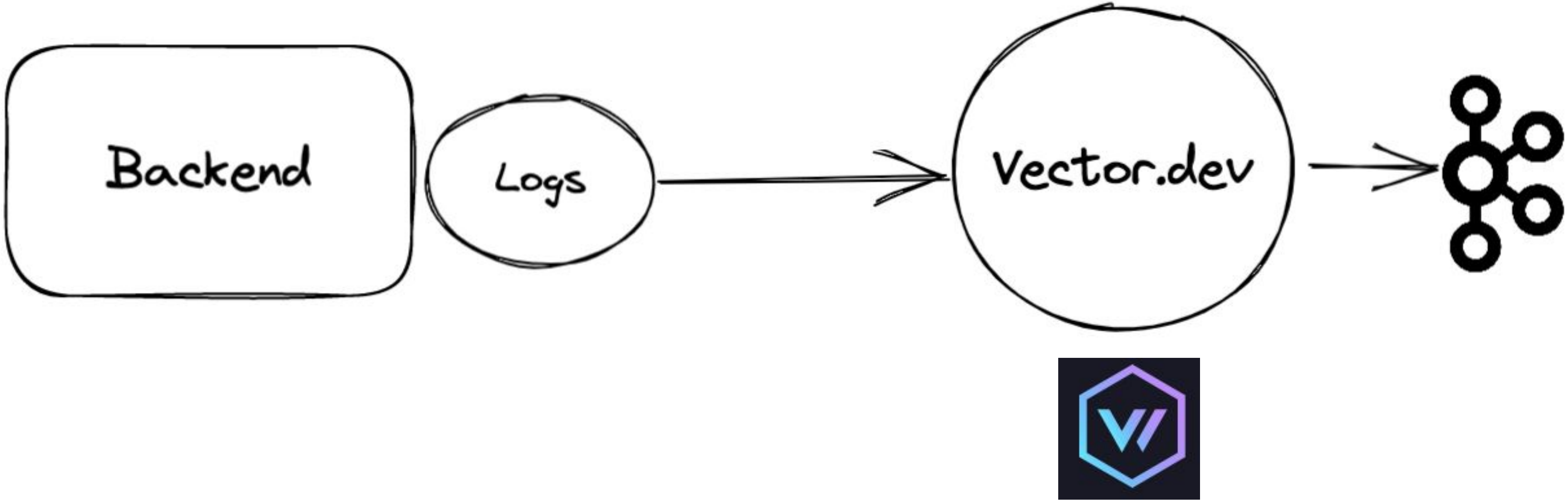
- Log events with our client



```
import blinkit_logger as bl

bl.emit({"order_type":"awesome"})
```

- Write to a file -> Stdout in a sidecar

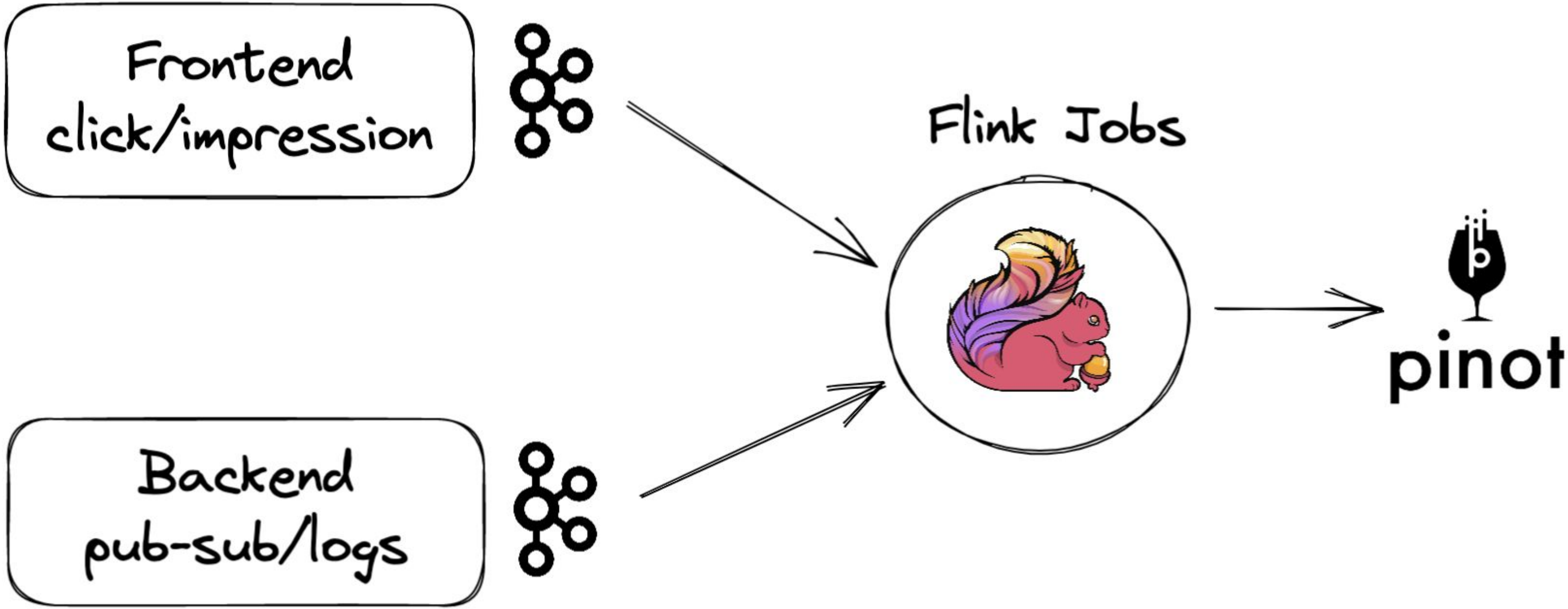

Backend
Logs
Vector.dev

Aggregation Layer

- To Pre-aggregate or Not To
- Apache Flink (Pre)
- Sink (Post)

Sink

Apache Pinot ✨



Frontend
click/impression

Backend
pub-sub/logs

Flink Jobs

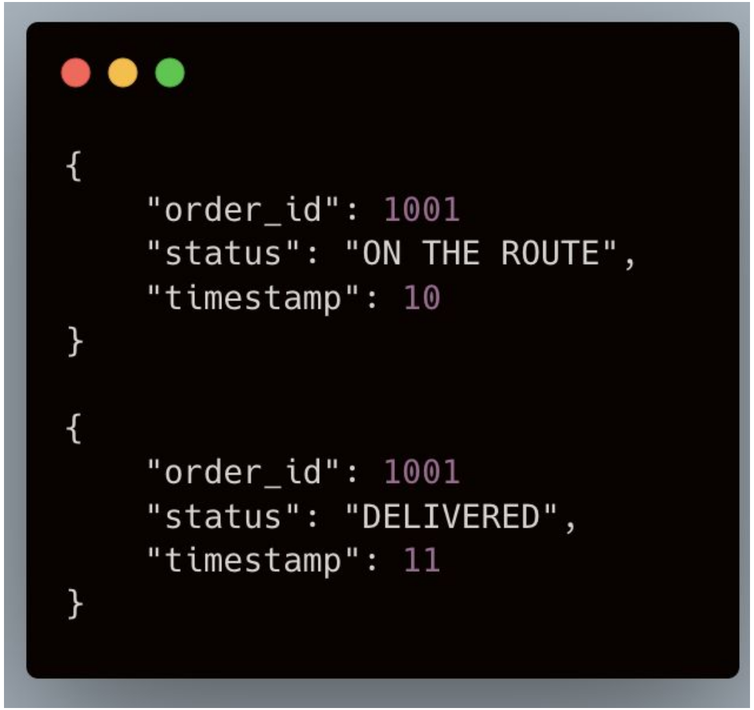
pinot

Observability Overlord

Sonar

What we need

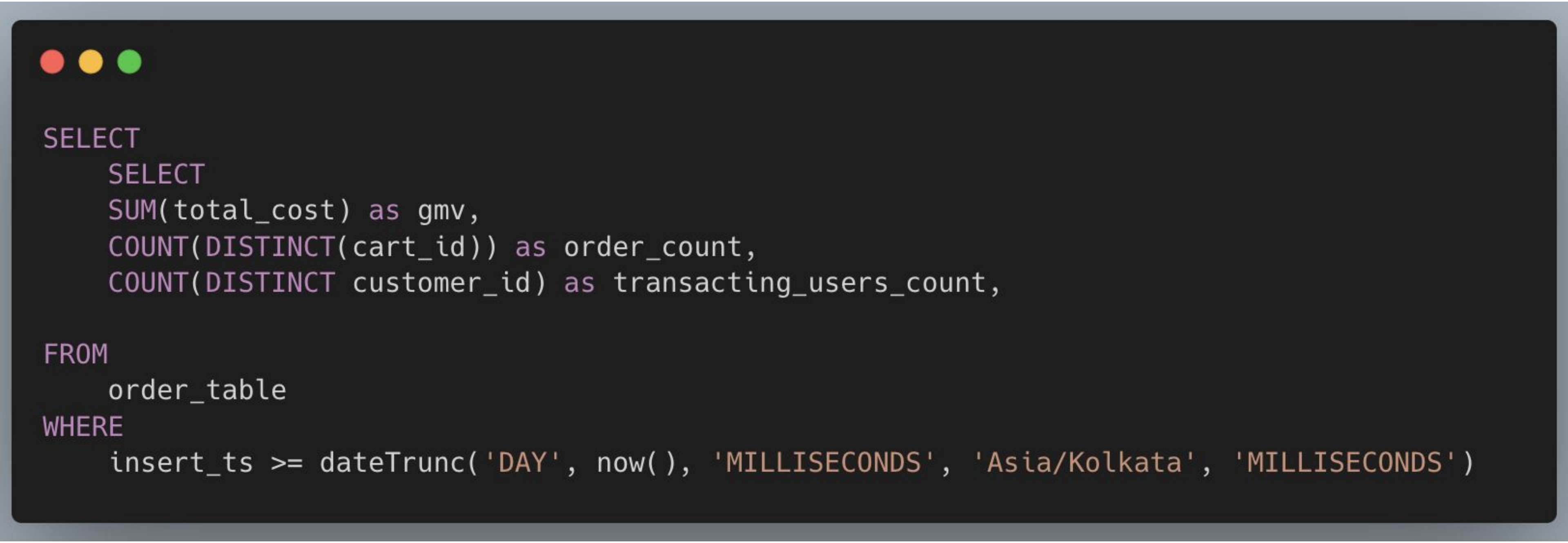
- Ingest from Kafka
- Upserts!
- Primary Key



```
{
    "order_id": 1001
    "status": "ON THE ROUTE",
    "timestamp": 10
}

{
    "order_id": 1001
    "status": "DELIVERED",
    "timestamp": 11
}
```

SQL-like Interface

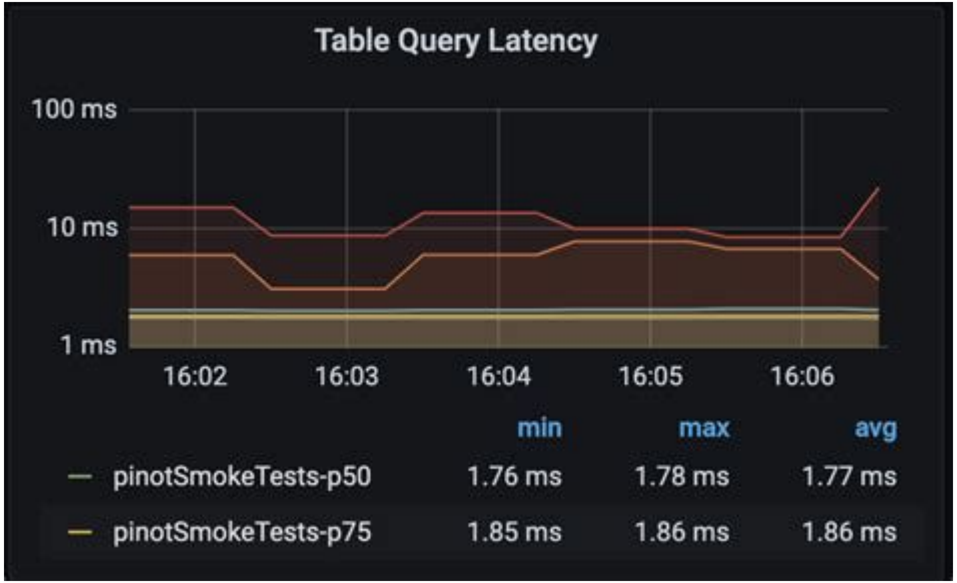


```sql
SELECT
    SELECT
    SUM(total_cost) as gmv,
    COUNT(DISTINCT(cart_id)) as order_count,
    COUNT(DISTINCT customer_id) as transacting_users_count,

FROM
    order_table
WHERE
    insert_ts >= dateTrunc('DAY', now(), 'MILLISECONDS', 'Asia/Kolkata', 'MILLISECONDS')
```

Performance

- Fast Queries
- Historical Data



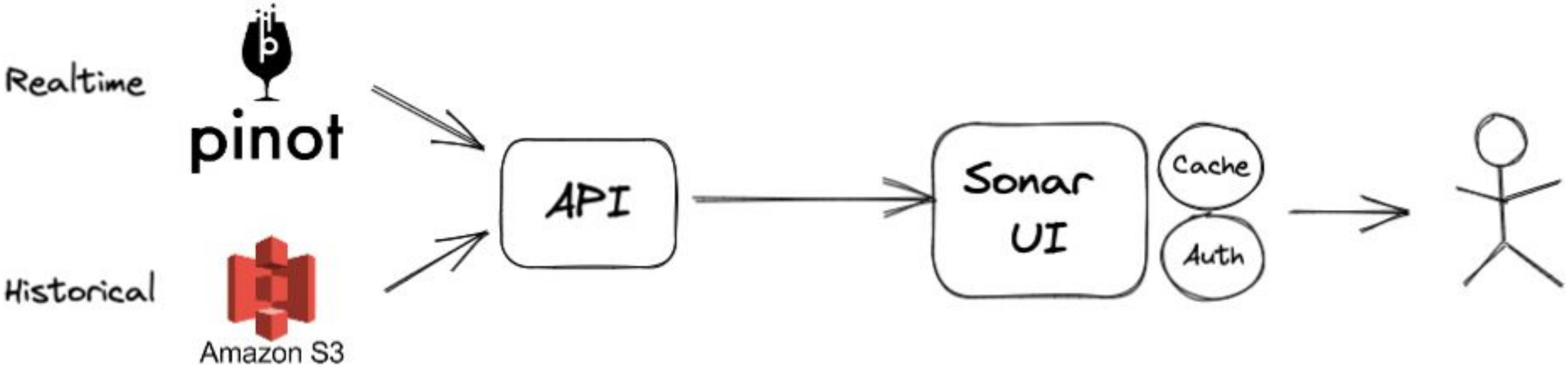
	min	max	avg
— pinotSmokeTests-p50	1.76 ms	1.78 ms	1.77 ms
— pinotSmokeTests-p75	1.85 ms	1.86 ms	1.86 ms

WE GOT IT!

- Pluggable indexing
- Real time ingestion
- SQL Interface
- Lookup Joins
- Hybrid tables
- Full and Partial Upserts

…

pinot


Realtime
pinot
Historical
Amazon S3
API
Sonar
UI
Cache
Auth

Ad Platform

- Performance feedback loop
- Powering campaigns
- Agg metrics for brand partners
- Rotate ads and ensure delivery rates

More where that came from...

- Performance monitoring
- App HTTP metrics
- etc...

Thank you 🙏