ClickHouse

Building an Events Lakehouse that supports dynamic schema

Siddarth Jain

Bengaluru, March 2024

DoctorDroid

Table of Contents

Doctor Droid is a tooling company

Dynamic Playbooks	Define dynamic playbooks that can auto-investigate metrics & logs data from different platforms.
Alert Insights Slack Bot	A tool that assess your existing alerting setup and provides recommendations on how to improve them.
Kenobi	Events Lakehouse that supports dynamic schema

Introductions

Siddarth Jain

– CEO & Co-Founder

– Data Science at Shadowfax

– Investment Analyst at Accel

Introductions

Dipesh Mittal

– CTO & Co-Founder

– VP-Engg at Shadowfax

– Engineer at PayPal

Siddarth Jain

– CEO & Co-Founder

– Data Science at Shadowfax

– Investment Analyst at Accel

ClickHouse

DoctorDroid

Introductions

Dipesh Mittal

– CTO & Co-Founder

– VP-Engg at Shadowfax

– Engineer at PayPal

Siddarth Jain

– CEO & Co-Founder

– Data Science at Shadowfax

– Investment Analyst at Accel

Mohit Goyal

– Founding Engineer

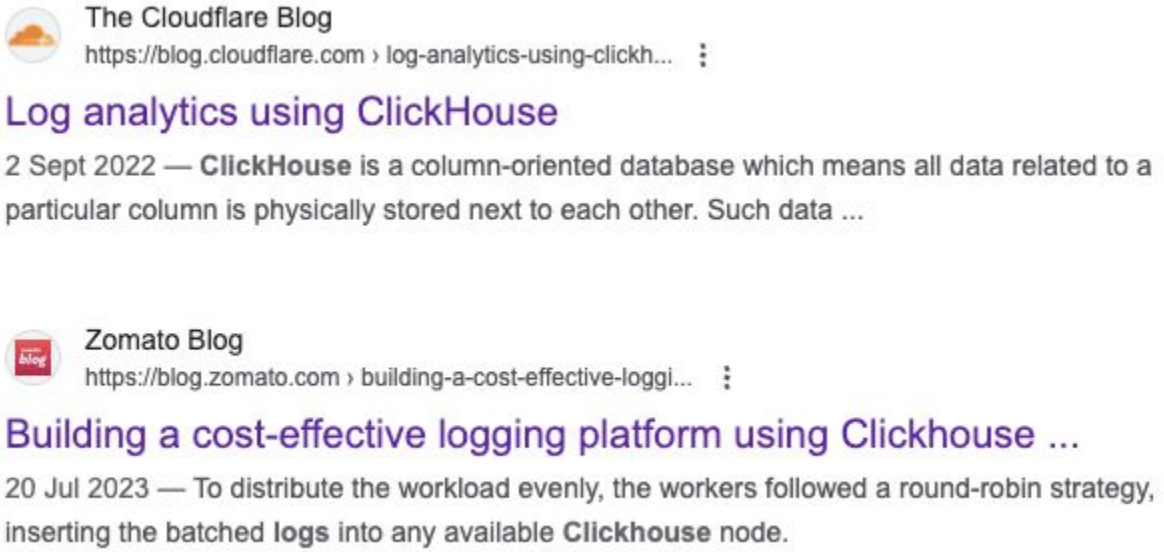
– Ex-Swiggy, BITS Pilani

Jayesh Sadhwani

– MERN Instructor at AccioJob

– DTU

Lot of discussion on Logging



Very Limited Discussion on Lakehouse

Create **a high-performing events storage** with support for dynamic schema and real-time analytics

1. Search
2. Aggregations on events & event_joins
3. Condition based alerts

Dynamic Schema

Java

```
{
 "event_name": "OrderCreated",
 "timestamp": 1677751161120,
 "payload": {
  "order_id": "ekpsHJ9GhQd7OU",
  "city": "Bengaluru",
  "store_code": "PZHT0056",
  "order_value": 435,
  "promised_eta": 23
 }
}
```

Common Use-cases:

- Asynchronous Data pipelines
- E-commerce order states
- Consumer Experience

Current Status

- We have processed **2B+ total events** till date in our platform.
- 150M events/day prod traffic tested
    - Sub 5-seconds search on this data with TTL of 15 days

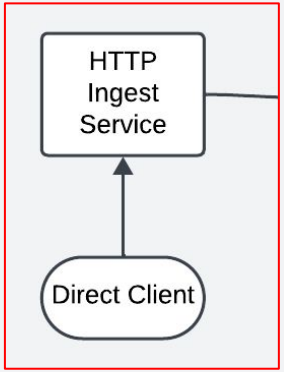
We have clients sending us events via **Segment, SDK, CloudWatch Logs**

- Unique Event Types: 2203
- Unique Keys Processed: ~270 keys

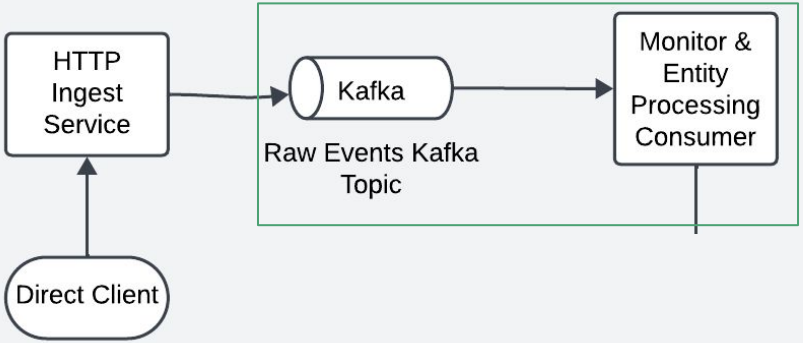
Requirements

Integrations	Ingest	Events Storage	User Features
Segment	Event drop guarantees	Source of Truth	Search
AWS Kinesis Firehose	Real-time processing	Fast-analytics	Metrics
SDK / API	Asynchronous processing	High performance on aggregation & search	Condition based alerts

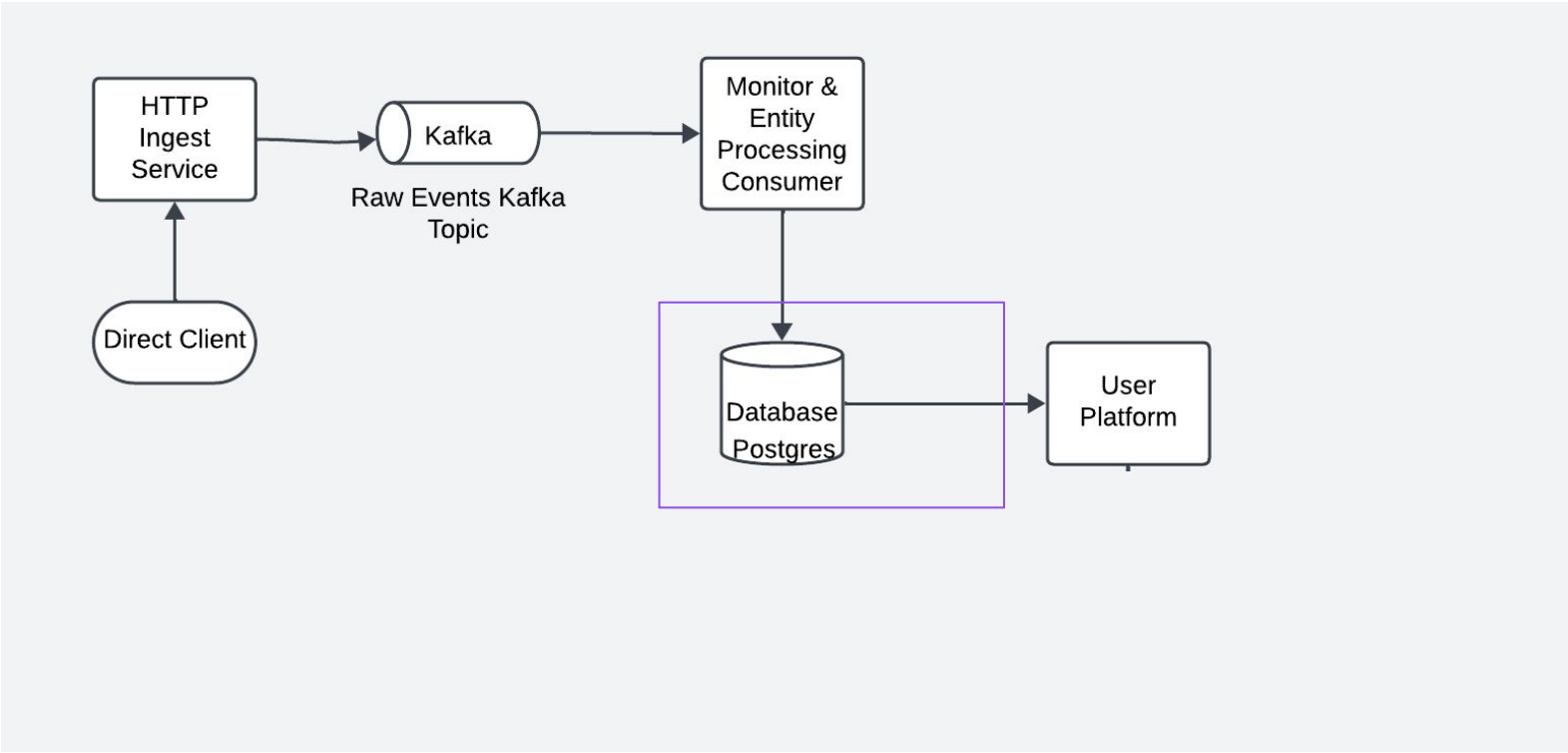
Kenobi v0 Architecture



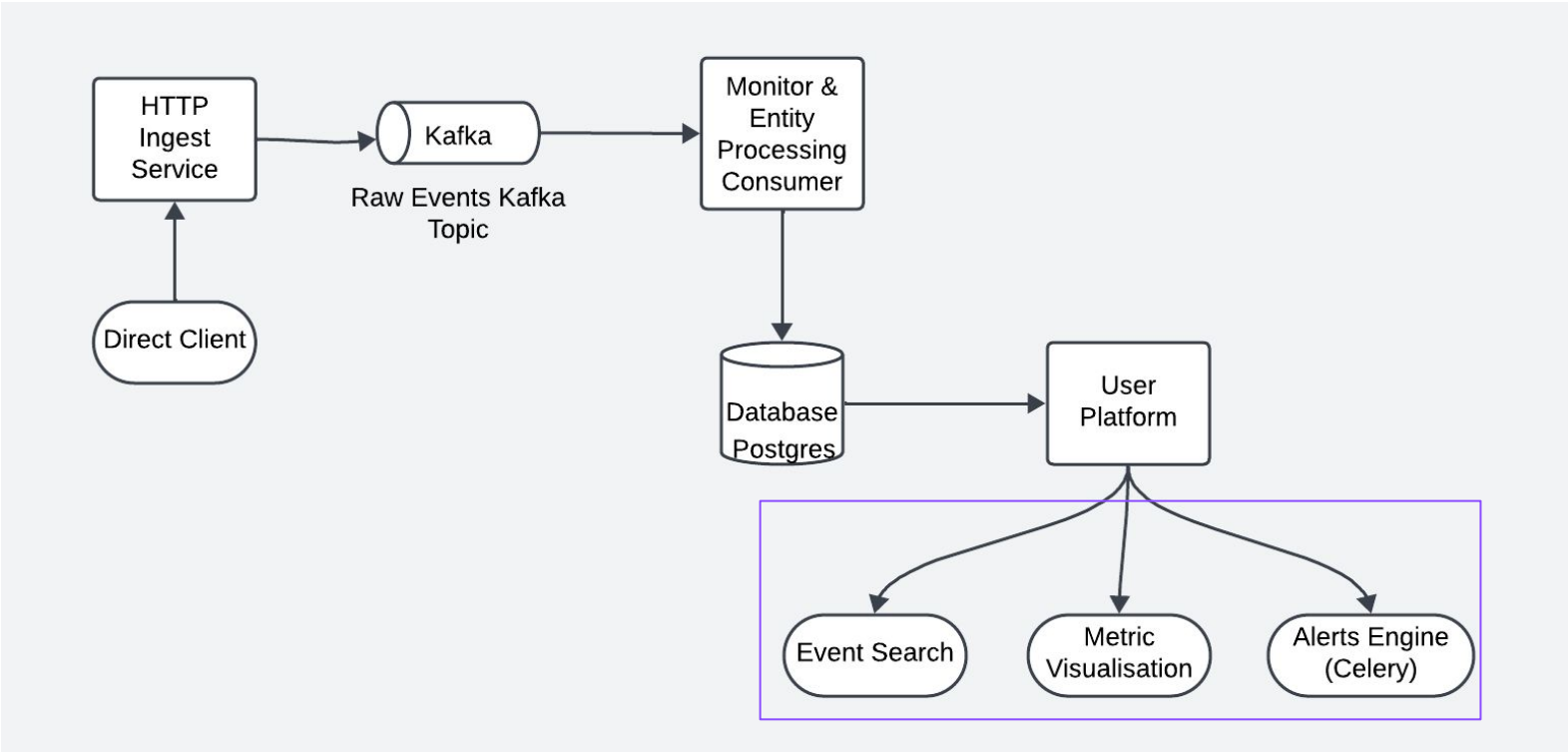
Kenobi v0 Architecture



Kenobi v0 Architecture



Kenobi v0 Architecture



Performance on PostgreSQL

Events Volume:	25-50k events/day
Aggregation API:	
Search API	Asynchronous

Metric	Avg
p50:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	1.78 s
p75:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	4.80 s
p90:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	9.55 s
p95:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	11.84 s
p99:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	13.03 s
p99.9:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	13.46 s

Quiz 1

Open www.DrDroid.io in one browser.

Send your results via Intercom chat.

2:30 timer.

30 seconds buffer to send screenshot.

Quiz 1

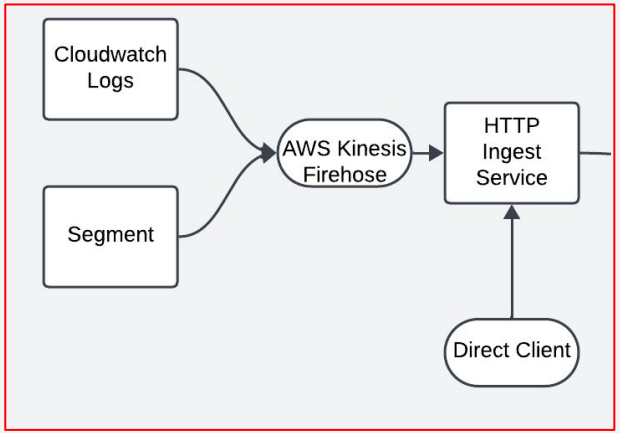
BUFFER SLIDE

Quiz 1

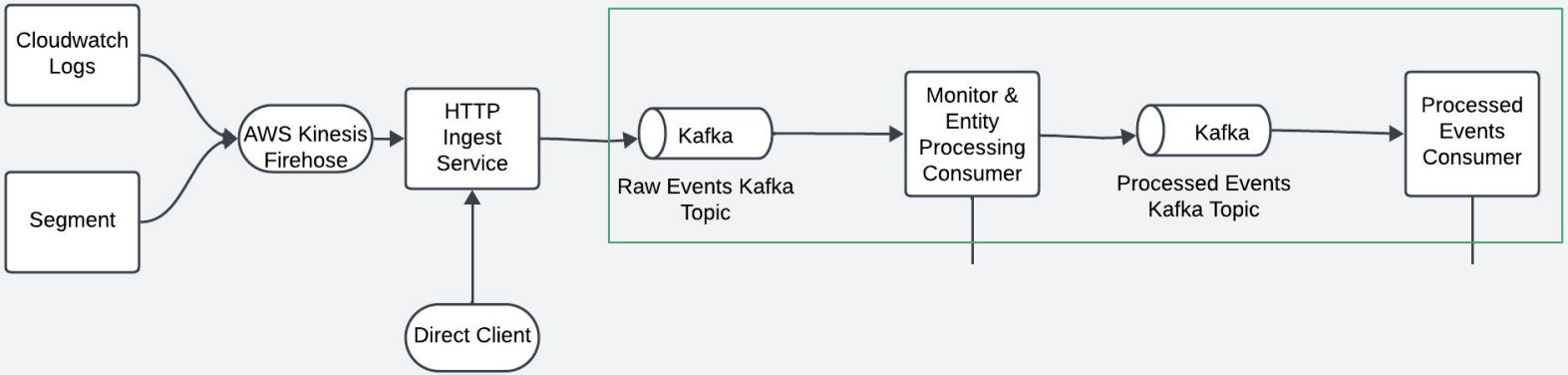
Tool Evaluations

	ClickHouse	**Pinot**	**ElasticSearch**
Storage	Columnar with parts optimised for querying across the dataset	Columnar with chunks with custom index, optimised for max data freshness	Inverted index storage, optimizing for quick full-text search and aggregations.
Maintenance	Single component deployment	Extensive fine-tuning & manual configuration, multiple components (Controller, Zookeeper, Broker, Server)	Difficult with non-strict schema – due to mapping explosion / multi-tenancy issues
Schema Flexibility	High	Low – with manual definition and updation	

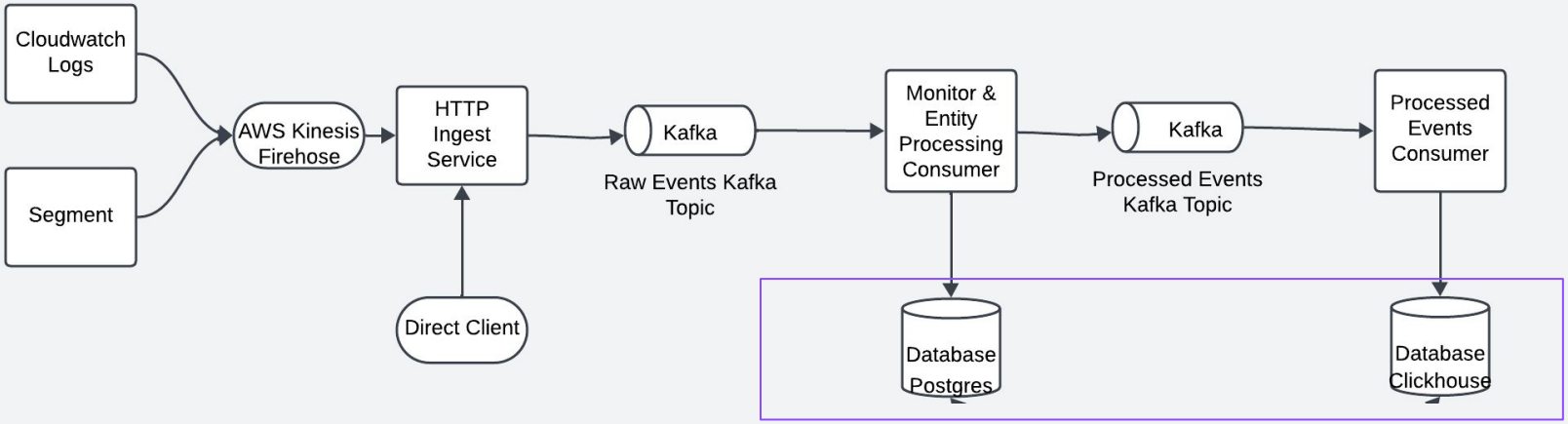
Kenobi v1 – Integration



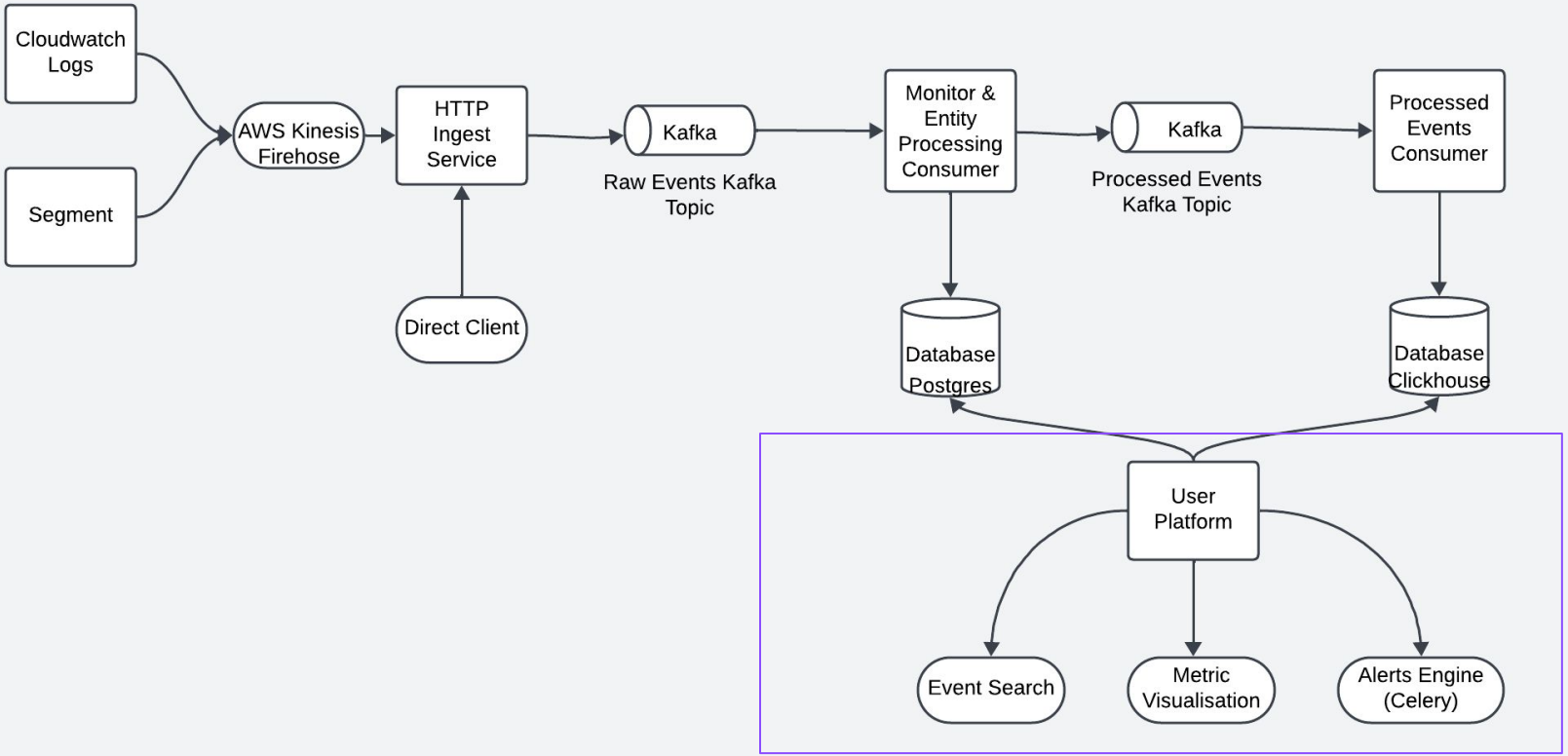
Kenobi v1 – Ingestion



Kenobi v1


Cloudwatch Logs
Segment
AWS Kinesis Firehose
HTTP Ingest Service
Direct Client
Kafka
Raw Events Kafka Topic
Monitor & Entity Processing Consumer
Kafka
Processed Events Kafka Topic
Processed Events Consumer
Database Postgres
Database Clickhouse

Kenobi v1


Cloudwatch Logs
Segment
AWS Kinesis Firehose
HTTP Ingest Service
Direct Client
Kafka
Raw Events Kafka Topic
Monitor & Entity Processing Consumer
Kafka
Processed Events Kafka Topic
Processed Events Consumer
Database Postgres
Database Clickhouse
User Platform
Event Search
Metric Visualisation
Alerts Engine (Celery)

Specs of our current stack

Databases:

- Postgres DB (Main + Replica) : db.m6i.large (2 vCPUs, 8GB Memory)
- Clickhouse Cloud DB : (4 vCPUs, 16GB Memory) X 3 Replicas

Kafka:

- 3 Broker Cluster (Each having 2 vCPUs, 8GB Memory, 200GB Disk)
- Same cluster manages all topics

Platform:

- Python, ReactJs
- Deployed on AWS Kubernetes Engine

Scales we have tested:

- 150M+ logs/day with this config with a TTL of 15 days

JSON Auto-inference:

Given any JSON, it automatically identifies the key/value pairs

- Fast querying
- No column changes (to the eyes)

```
{
  "name": "test_name"
  "timestamp": 1677751261100,
  "payload": {
    "key_1": "new_one",
    "key_2": "44",
    "key_3": "cc"
}
```


```
{
  "name": "test_name"
  "timestamp": 1677751261100,
  "payload": {
    "key_1": "new_one",
    "key_2": "44",
    "key_3": "cc"
}
```

ClickHouse Implementation

```
set allow_experimental_object_type = 1;
create table table_name (
   id UInt64,
   created_at DateTime64(3, 'UTC') default now(),
   timestamp DateTime64(3, 'UTC') default now(),
   event_type_id UInt64,
   event_type_name VARCHAR(256),
   event_source UInt16,
   processed_kvs JSON,
)
engine = MergeTree()
PARTITION BY toDate(timestamp)
primary key (event_type_id, timestamp)
order by (event_type_id, timestamp)
TTL toDateTime(created_at) + INTERVAL 10 DAY DELETE;
```

POST e/api/metrics

Aggregation API

Metric	Avg
p50:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	434 ms
p75:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	622 ms
p90:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	862 ms
p95:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	942 ms
p99:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	1 055 ms
p99.9:trace.django.request{env:prod,resource:98a42b80e2901126,service:prototype}	1 055 ms

Search API

Metric	Avg
p50:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	1.95 s
p75:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	2.20 s
p90:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	2.24 s
p95:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	2.24 s
p99:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	2.24 s
p99.9:trace.django.request{env:prod,resource:ff7f34ad56155191,service:prototype}	2.24 s

Quiz 2

– Open DrDroid.io – first person to respond correctly on intercom wins.

Quiz 2

– *There are 4 events being streamed to ClickHouse. Each event has a JSON field.*

– *Maximum key count in any row in JSON column in CH table?*

Quiz 2 – Max key count in JSON column in CH table?


```
{
  "name": "OrderCreated",
  "timestamp": 1677751161120,
  "payload": {
    "order_id": "ekpsHJ9GhQd7OU",
    "city": "Bengaluru",
    "store_code": "PZHT0056",
    "order_value": 435,
    "promised_eta": 23
  }
}
```

```
{
  "name": "Payment_Started"
  "timestamp": 1677751161100,
  "payload": {
    "transaction_id": "si_LULhDM2vvvt7yZ",
    "amount": 54.8,
    "currency": "usd",
    "card_type": "amex",
    "invoice_id": "in_1KnN0G589O8KAxCGfVSpD0Pj"
}
```

```
{
  "name": "Transfer_Prepared"
  "timestamp": 1677751161100,
  "payload": {
    "recipient_id": "tWaIb3aK3vlHn9",
    "amount": 10323,
    "currency": "inr",
    "recipient_bank": "citi",
    "source_bank": "hdfc"
}
```

```
{
  "name": "Transfer_Completed"
  "timestamp": 1677751461100,
  "payload": {
    "recipient_id": "tWaIb3aK3vlHn9",
    "transaction_id": "98363ixIwXnh07L5DuLvQE6c",
    "transaction_status": "success"
}
```

How the JSON column looks like:

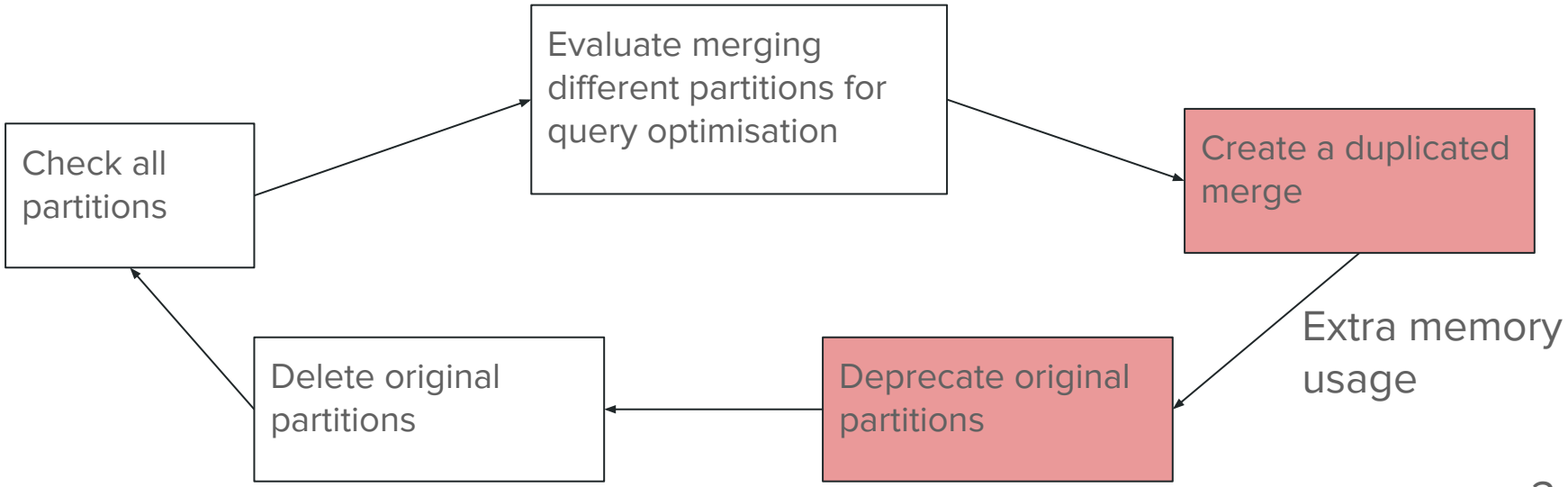
Open Source Setup

We wanted to make it easy for companies to host it in their environment:

- Docker compose based – without any config change, 1Mn events/day. With a few config +machine upgrades – should be able to scale.
    - `Prototype.docker-compose.yml`
- Link

How is CH able to query so fast with dynamic data?

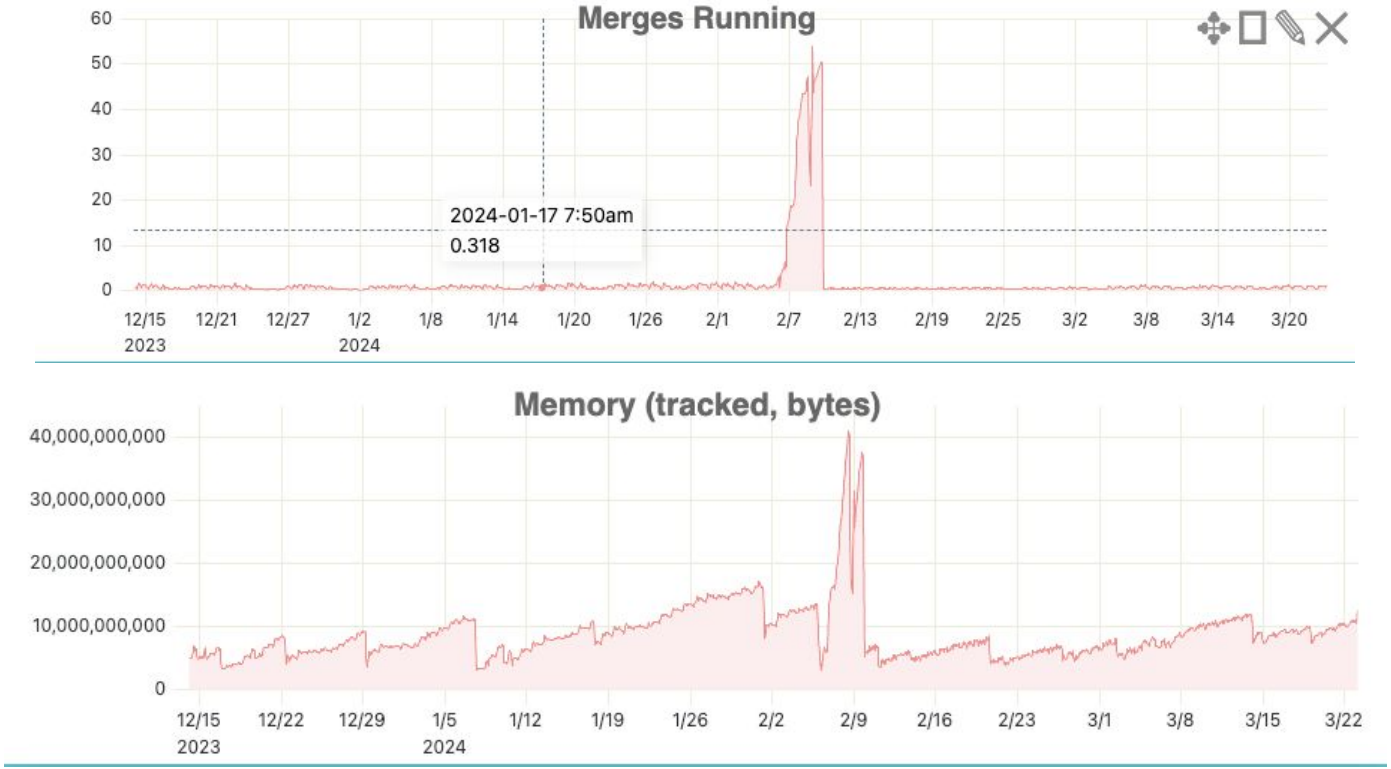
MergeTree – Background Operation to optimise it's partitions:

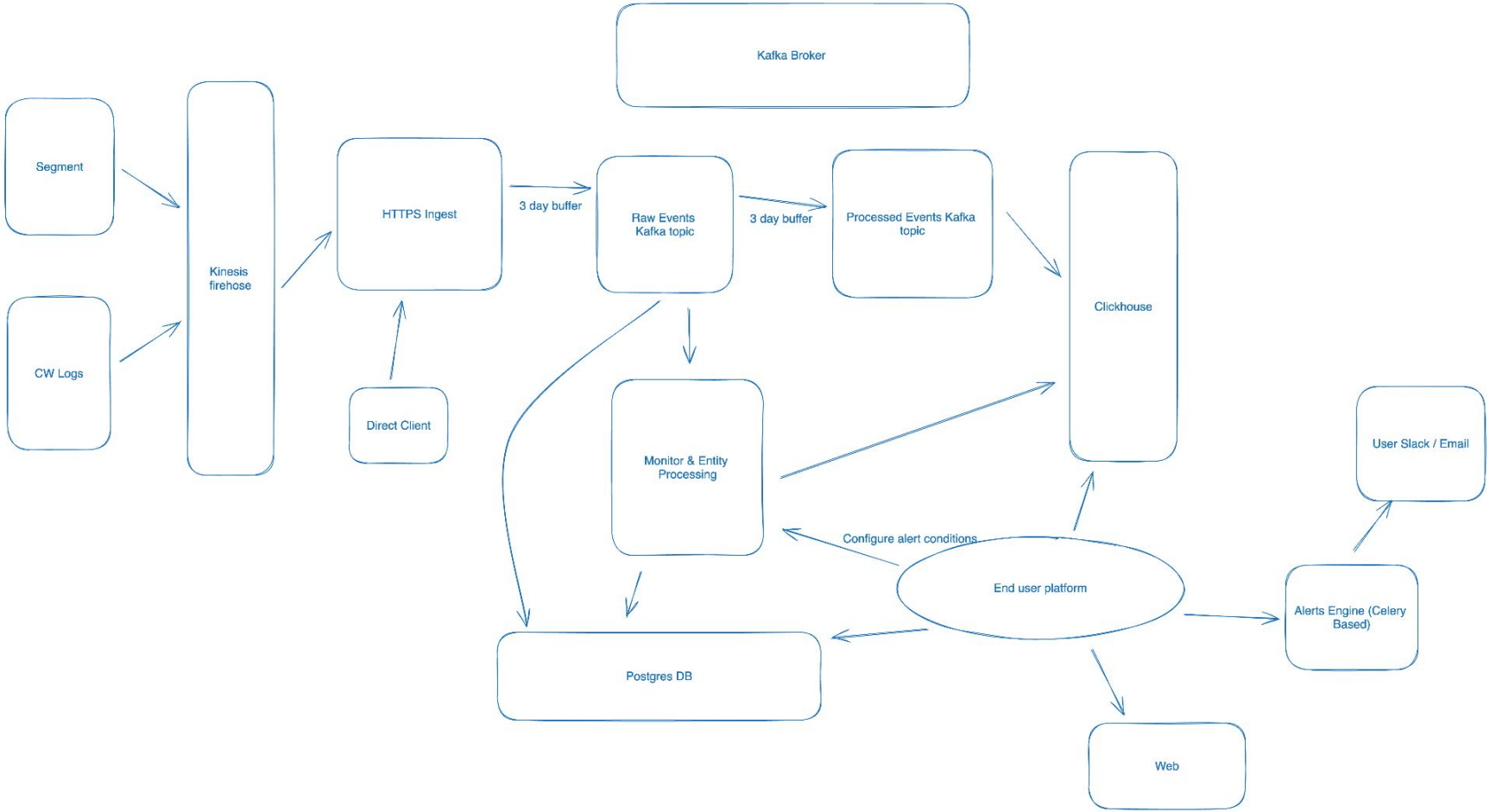


Constraints with JSON experimental feature

- KV Pair limit: 1000 per table
- Risk of breaking in production
- Lot of normal setups break (E.g. CDC Postgres to CH)

Constraints with JSON experimental feature




Kafka Broker
Segment
CW Logs
Kinesis firehose
HTTPS Ingest
3 day buffer
Raw Events Kafka topic
3 day buffer
Processed Events Kafka topic
Clickhouse
Direct Client
Monitor & Entity Processing
User Slack / Email
Configure alert conditions
End user platform
Alerts Engine (Celery Based)
Postgres DB
Web

Quiz 1

BUFFER SLIDE

Quiz 3

How many metro stations in Bangalore as per 2025 plan?

ClickHouse

DoctorDroid

Thank You

Linkedin: https://www.linkedin.com/in/siddarth-jain227/ (Siddarth Jain Doctor Droid)

Twitter: TheBengaluruGuy

Indexing

Table Name	Compression Ratio	Volume	Keys
t_1	25.8	1M rows	150
t_2	19.9	10M rows	160
t_3	8.3	200M rows	50

Appendix:

Pgtuning – worked on a read optimised tuning to maximise performance.