

# Modern application Observability with Grafana & Quickwit





# Who am I?

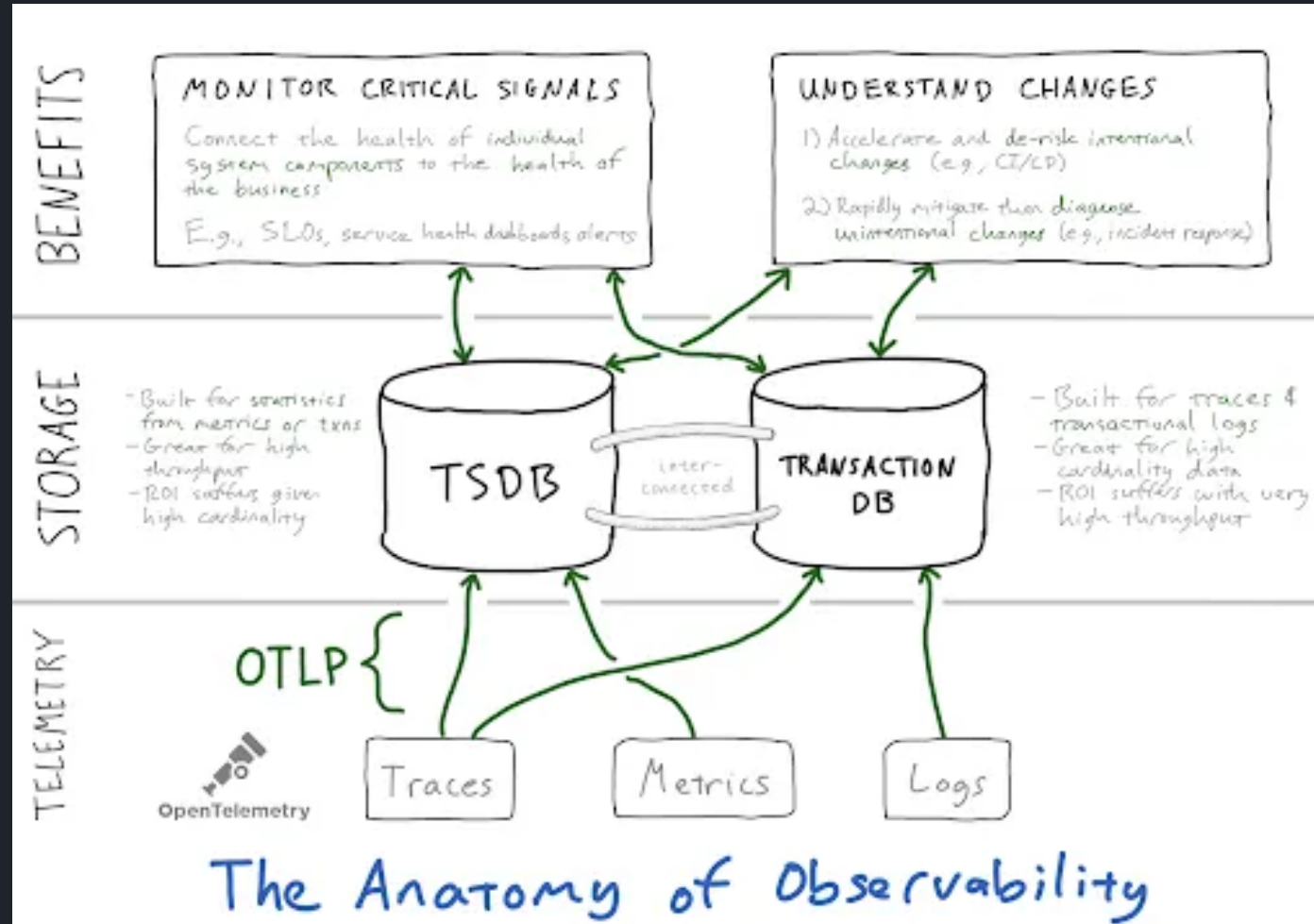
- François Massot @francoismassot on Twitter / Fosstodon
- Core dev on Quickwit engine (Rust) and the Grafana datasource plugin
- Cofounder of Quickwit



# Agenda

1. Anatomy of observability
2. The cardinality curse
3. Logs and traces storage engine: Quickwit
4. Demo time

# Anatomy of observability





# The cardinality curse

Distributed systems can fail for a large number of reasons.

What if we have metrics with labels: `version`, `host`, `customer_id`, `service`, ...

Cardinality =  $10 * 1k * 100k * 10 = 10$  billion

=> We should control the cardinality for metrics.

=> Let's keep all the attributes in traces.



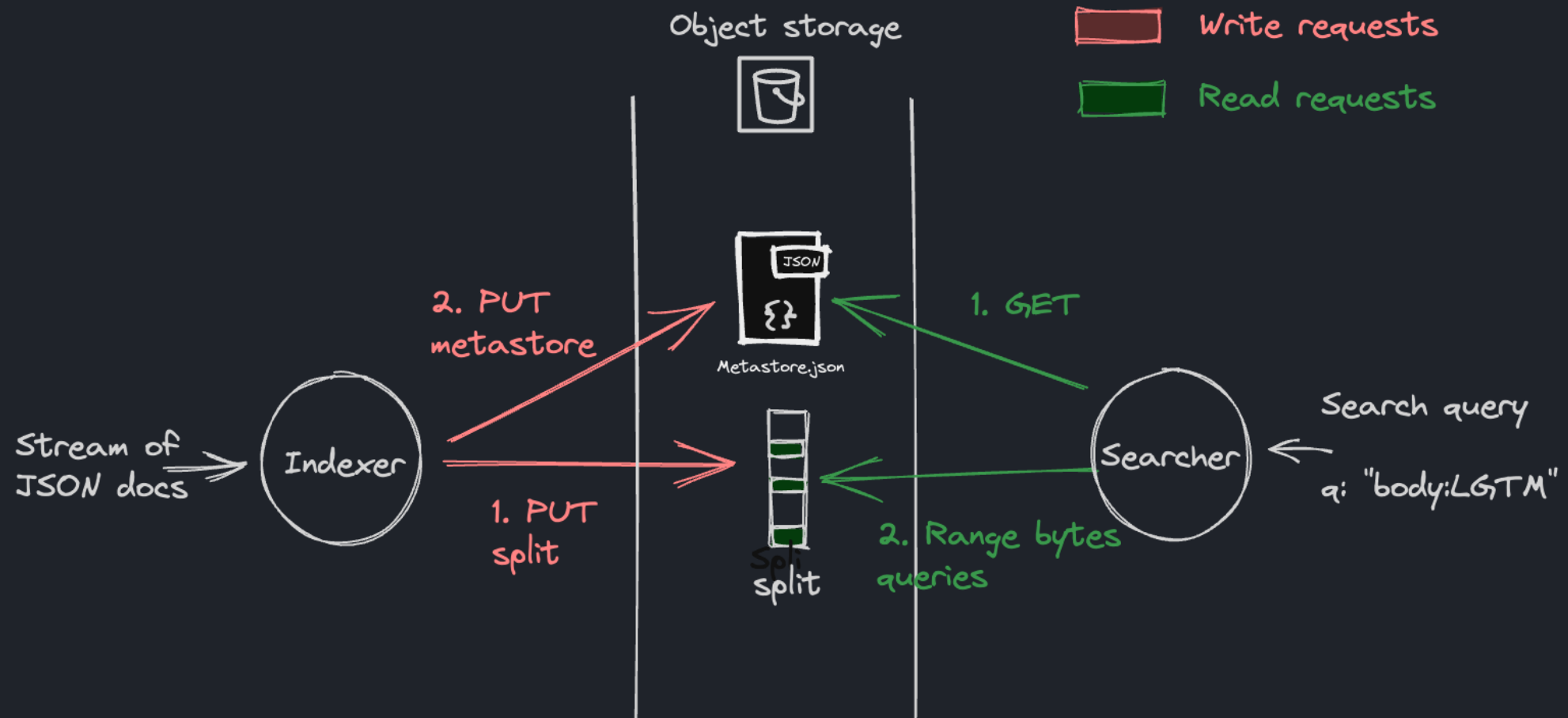
# Logs and traces storage engine: Quickwit

- Distributed search engine for logs and traces
- Decoupled compute & storage (like Loki/Tempo).
- Optimized (sub-second) for query on object storage.

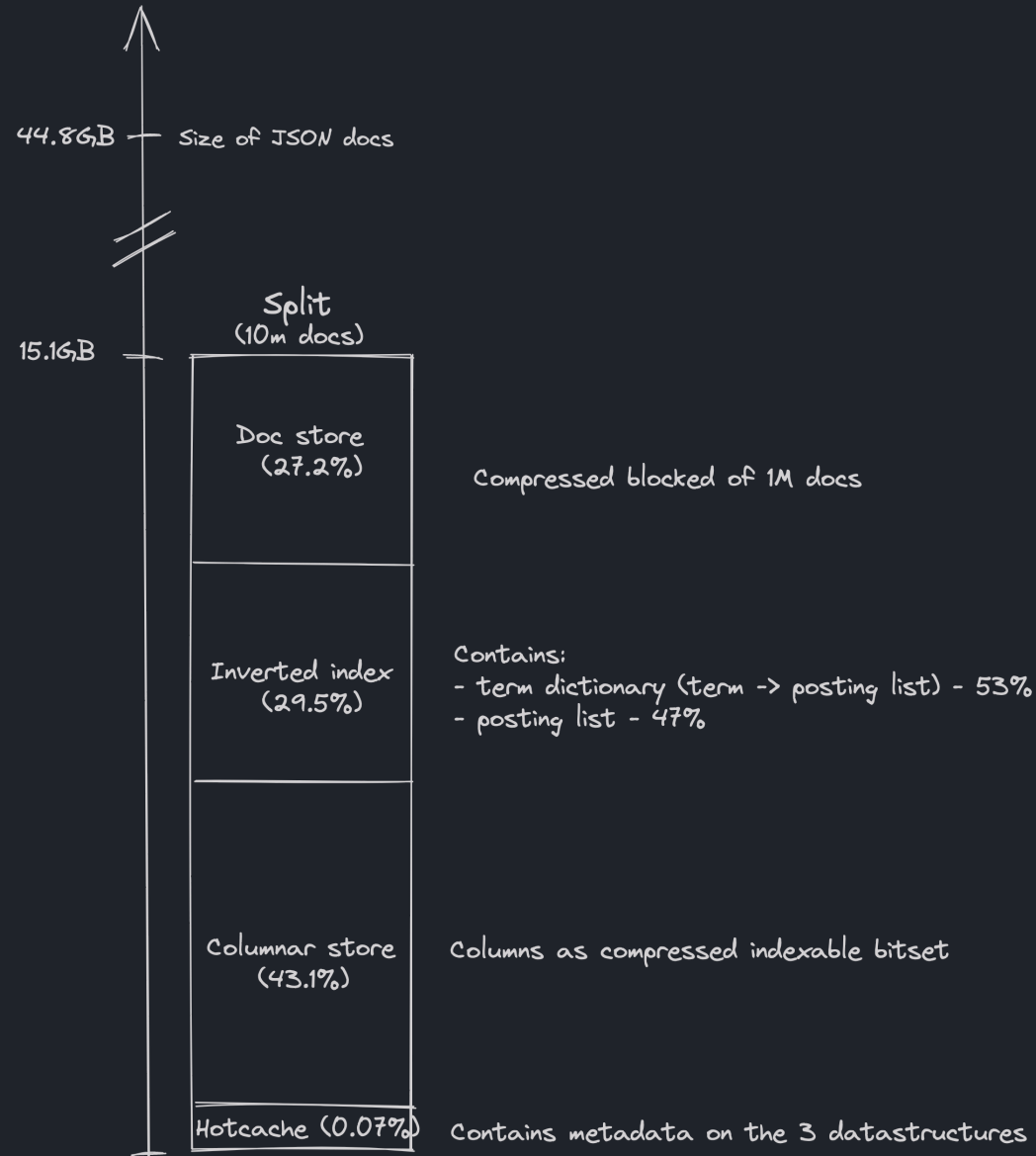
# Engine architecture



Architecture in its simplest form.



# Anatomy of a split





# Span data model in Quickwit



Based on the OTEL data model. `resource_attributes` and `span_attributes` are schemaless fields.

```
{
  "trace_id": "b31ab9dda41afde6b4ac992ea56afc89",
  "parent_span_id": "7505182209aa02d1",
  "span_id": "6bf90d1ffe7b9b5c",
  "span_kind": 2, // SERVER
  "service_name": "postgres",
  "span_name": "query-articles",
  "span_start_timestamp_nanos":1706984714411104000,
  "span_duration_millis":164,
  "span_attributes": {
    "net.sock.host.addr":"192.168.13.102",
    "k6.OK4pWn6reJlwtVH": "PwWJNACBiMOKvww2RMuykgsx0AySXI",
    ...
  },
  "resource_attributes": {...}
}
```



# Demo time!

Setup:

- xk6 tracing jobs to generate traces
- Quickwit deployed on a Kubernetes cluster
- Grafana instance with Quickwit datasource



# Generating and sending traces to Quickwit

- **tracegen**: simple but spans are too simple
- **xk6 tracing**: YES!

```
const traceTemplate =
  {
    spans: [
      {service: "shop-backend", name: "list-articles", duration: {min: 200, max: 900}},
      {service: "article-service", name: "select-articles", attributeSemantics: tracing.SEMANTICS_DB},
      {service: "postgres", name: "query-articles", attributeSemantics: tracing.SEMANTICS_DB, randomAttributes: {count: 5}},
      ...
    ]
  };

const client = new tracing.Client({
  endpoint,
  exporter: tracing.EXPORTER_OTLP
});

export default function () {
  const gen = new tracing.TemplatedGenerator(traceTemplate);
  client.push(gen.traces())
}
```

# Demo: Quickwit traces index



**QUICKWIT** quickwit-demo-quickwit Docs

Discover

`</>` **Query editor**

Admin

- Indexes
- Cluster
- Node info
- `</>` API

Indexes / otel-traces-v0\_7

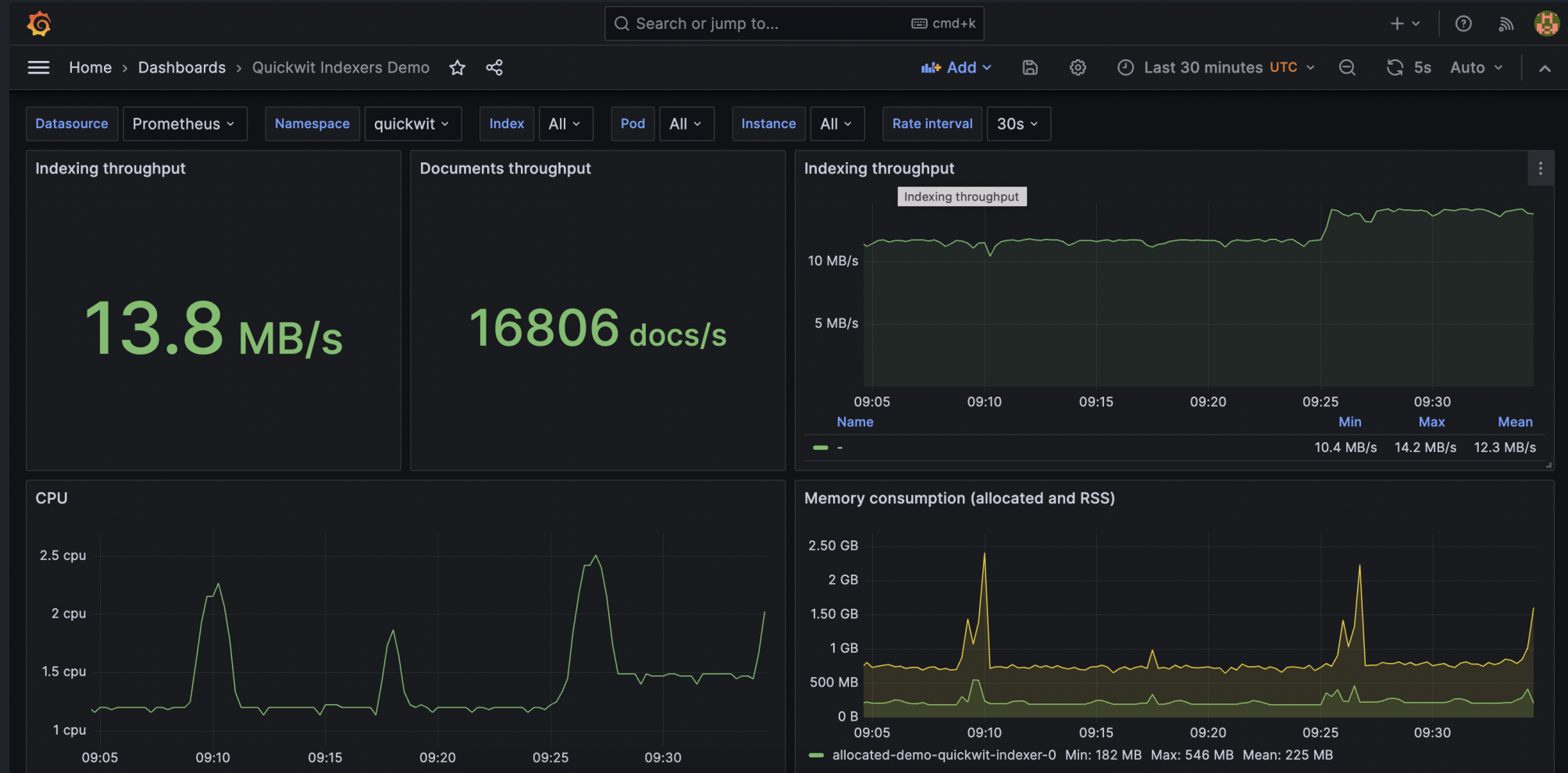
**SUMMARY** SOURCES DOC MAPPING INDEXING SETTINGS SEARCH SETTINGS RETENTION SETTINGS SPLITS

Created at:	2024/02/02 17:20
URI:	s3://quickwit-indexes/otel-traces-v0_7
Number of published documents:	128,347,652
Size of published documents (uncompressed):	105,680.32 MB
Number of published splits:	21
Size of published splits:	15,215.47 MB
Number of staged splits:	1
Number of splits marked for deletion:	514

API URL: [http://localhost:7280/api/v1/indexes/otel-traces-v0\\_7](http://localhost:7280/api/v1/indexes/otel-traces-v0_7)



# Demo: Quickwit indexer dashboard





# Demo: Exploring traces in Grafana

The screenshot displays the Grafana interface for exploring traces. At the top, there is a search bar with the text "Q Search or jump to..." and a "cmd+k" shortcut. Below the search bar, the navigation menu shows "Home > Explore". The main content area is titled "Quickwit Traces + Jaeger" and includes a "Run query" button. The query editor shows the following Lucene Query: `service_name:shop-backend AND span_name:article-to-cart AND span_attributes.k6.mM11MBRDxWEbj4S:"F6EC921QcPfMUmHSYKsF1stTFateVW"`. Below the query editor, there is a "Logs volume" bar chart showing the number of logs over time. The chart has a y-axis from 0 to 1 and an x-axis from 08:40 to 09:35. The bars represent the volume of logs, with a peak around 09:20. The bottom section shows the "Logs" table with various filters and options, including "Time", "Unique labels", "Wrap lines", "Prettify JSON", "Deduplication", and "Display results".

Query type: Metrics **Logs** Raw Data

Lucene Query: `service_name:shop-backend AND span_name:article-to-cart AND span_attributes.k6.mM11MBRDxWEbj4S:"F6EC921QcPfMUmHSYKsF1stTFateVW"`

Logs > Options

+ Add query Query history Query inspector

Logs volume

Quickwit Traces + Jaeger

logs

Logs Table

Time  Unique labels  Wrap lines  Prettify JSON  Deduplication **None** Exact Numbers Signature

Display results **Newest first** Oldest first

Line limit: 100 (11 returned) Download



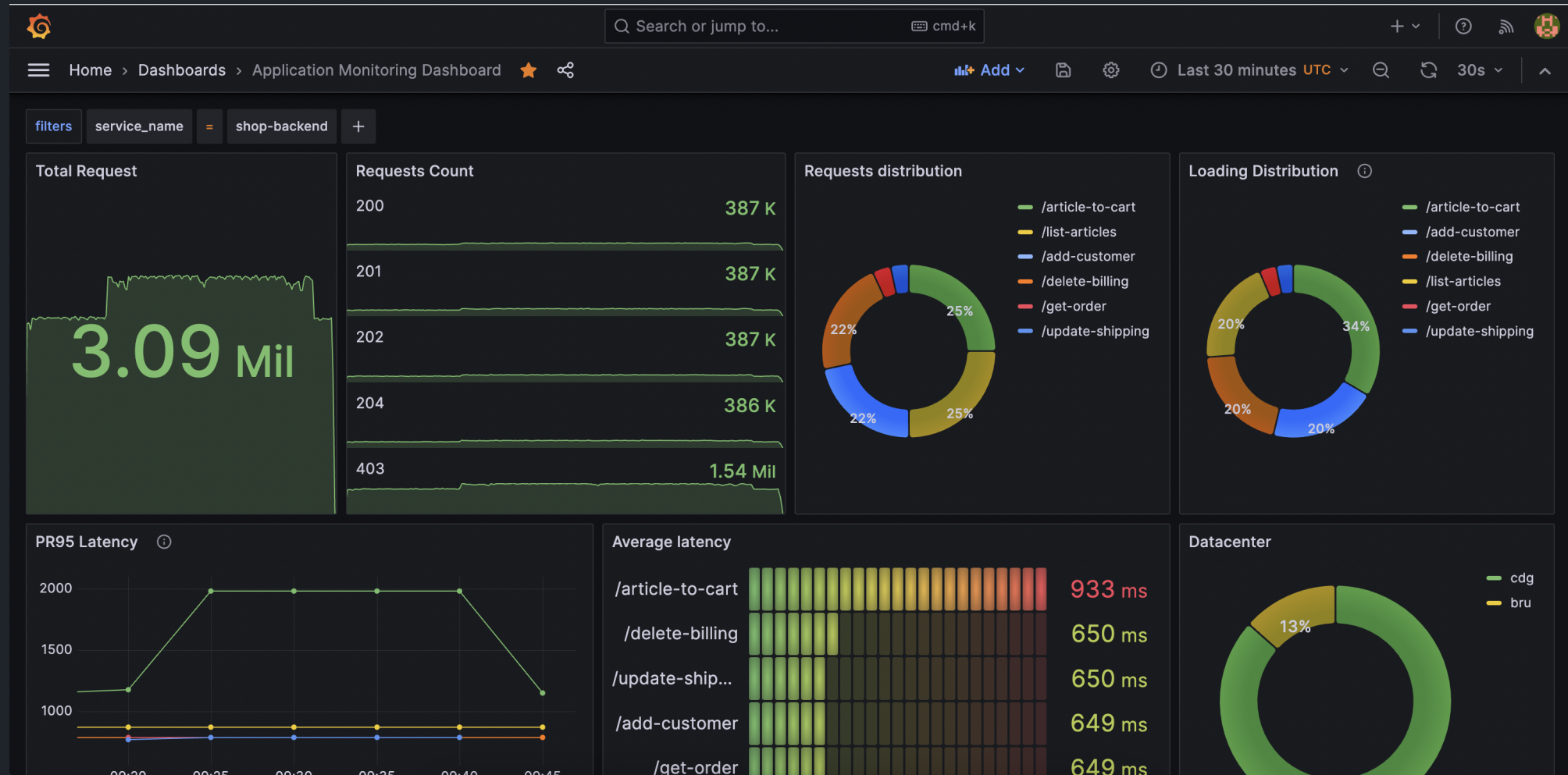
# Demo: Zoom on a specific trace

The screenshot displays the Jaeger web interface. On the left, a search panel shows a Lucene query: `service_name:shop-backend AND span_name:article-to-cart AND span_attributes.k6.mM11MBRDxWEbj4S:"F6EC921QcPmUmHSYKsF1stTFateVW"`. Below the search panel is a 'Logs volume' bar chart showing log activity over time from 08:40 to 09:30. At the bottom left, there are toggle switches for 'Time', 'Unique labels', 'Wrap lines', and 'Prettify JSON'.

The main right-hand panel shows the details of a selected trace: `shop-backend: article-to-cart` with a duration of 677.66ms. The trace ID is `2024-02-04 09:19:21.234` and the status is `PATCH 200 /article-to-cart`. Below this, a 'Span Filters' section indicates 10 spans. A horizontal bar chart visualizes the span durations. The 'Service & Operation' table below provides a detailed breakdown of the spans:

Service & Operation	Duration
shop-backend article-to-cart (677.66ms)	677.66ms
place-articles (444.45ms)	444.45ms
cart-service place-articles (376.51ms)	376.51ms
persist-cart (316.85ms)	316.85ms
shop-backend get-article (420.71ms)	420.71ms
article-service get-article (298.88ms)	298.88ms
select-articles (168.49ms)	168.49ms
postgres query-articles (197.31ms)	197.31ms
shop-backend authenticate (151.53ms)	151.53ms
auth-service authenticate (112.18ms)	112.18ms

# Demo: APM dashboard in Grafana







## Future work

- More aggregations (cardinality, rate) (Q2)
- Pipe-based query language (Q3)
- Metrics support (Q4)

# Thank you!

francois @ quickwit.io | @francoismassot on  
Twitter/Fosstodon

