HUSTEF

HUNGARIAN SOFTWARE TESTING FORUM

Observability: What, why and how (on a shoestring budget)

Abby Bangser (She/her)
abby@paintedwavelimited.com
@a_bangser
@a_bangser.bsky.social
hachyderm.io/@abangser

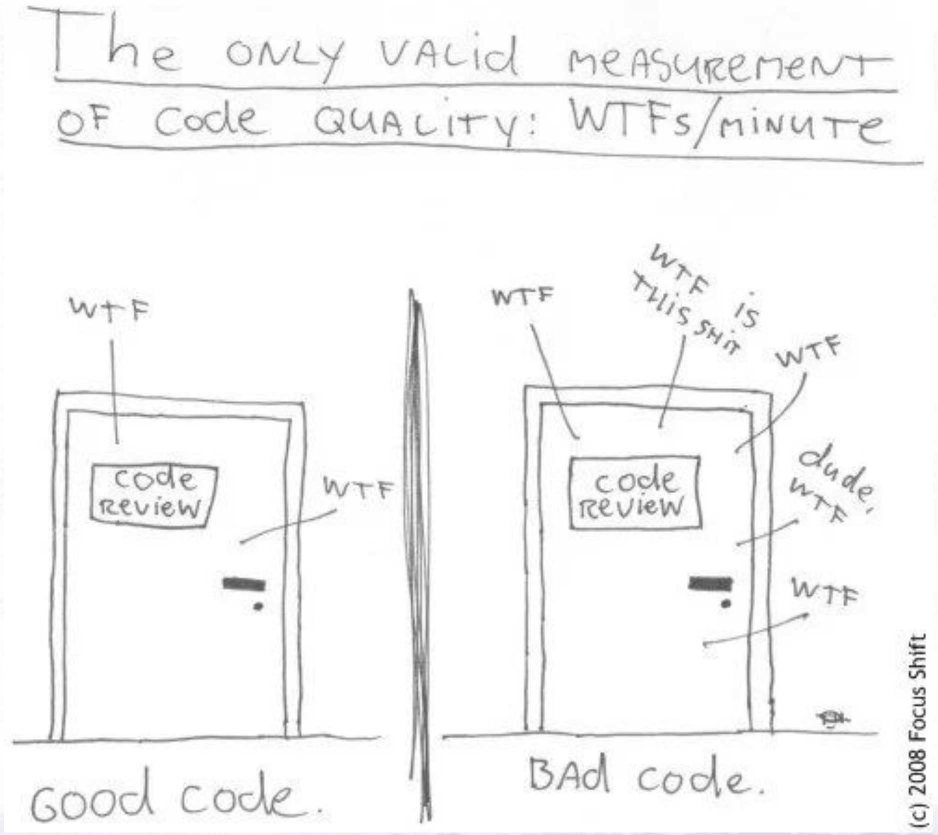
Observability

In control theory, observability is the measure of how well internal states of a system can be inferred from knowledge of its external outputs.

Observability

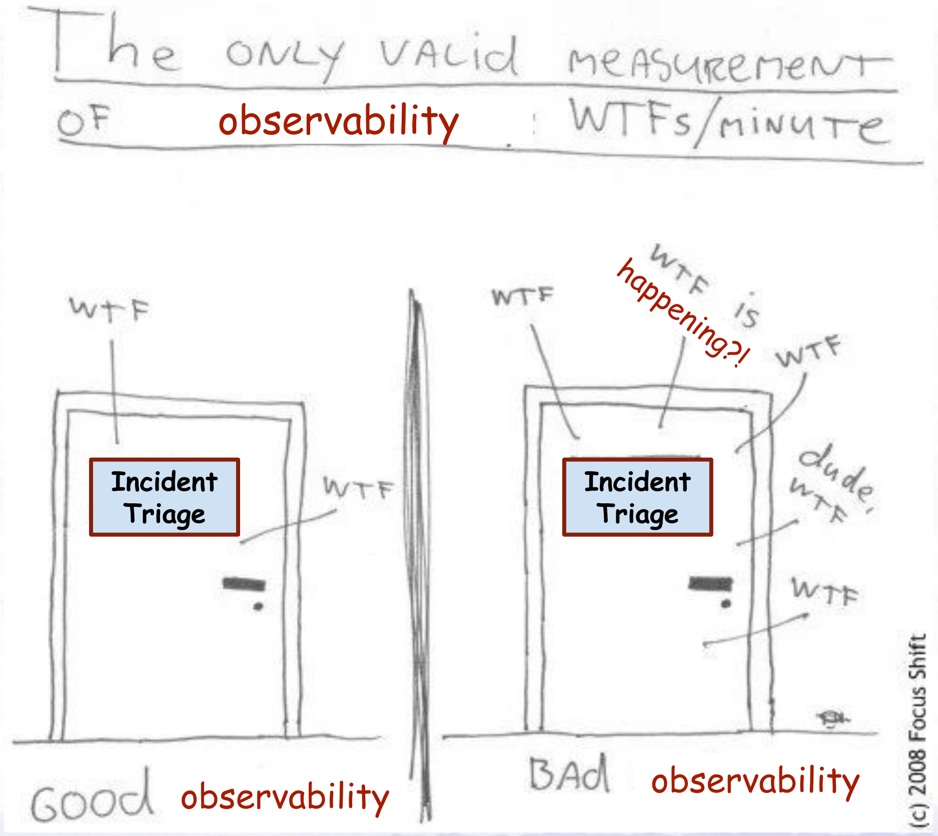
In control theory, observability is the <u>measure of how well</u> internal states of a system can be inferred from knowledge of its external outputs.

"Measure of how well" means it's a scale



@A_BANGSER

"Measure of how well" means it's a scale

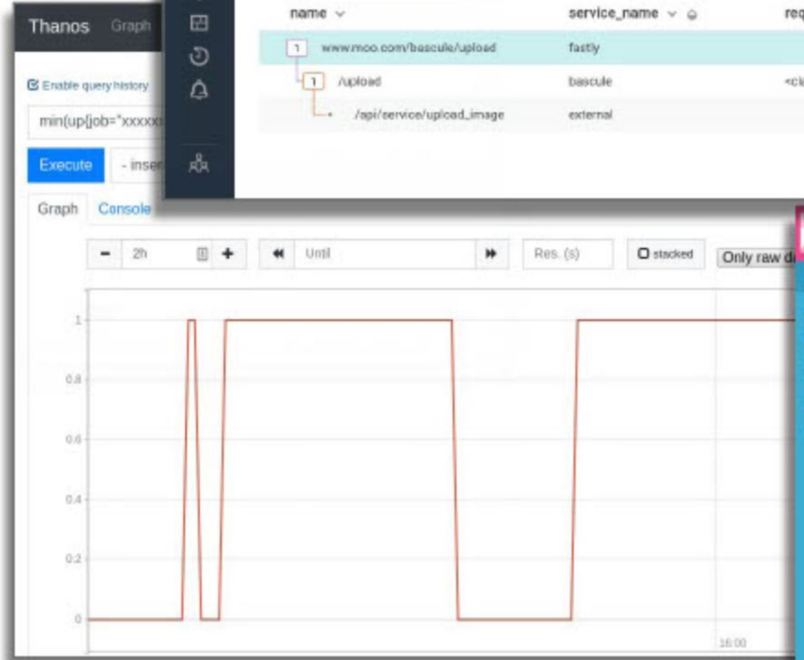


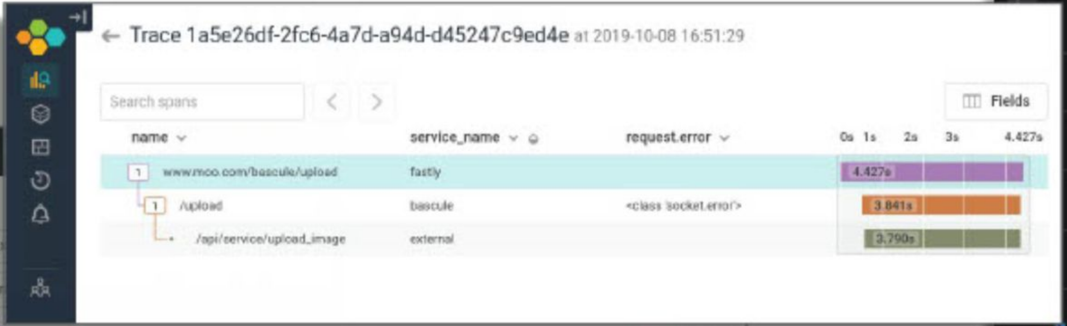
@A_BANGSER

Observability

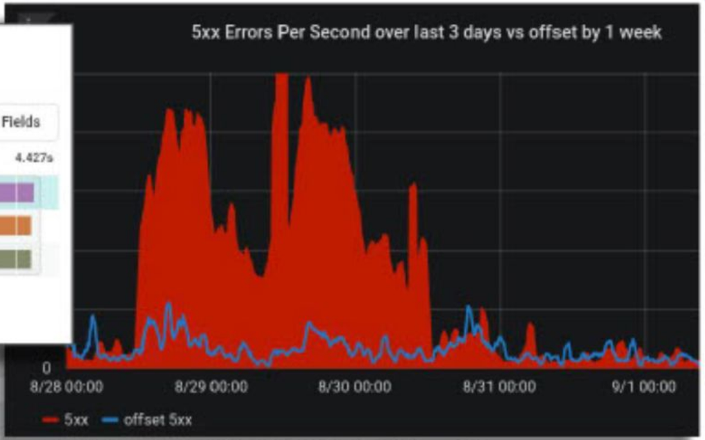
In control theory, observability is the measure of how well internal states of a system can be <u>*inferred from knowledge of its external outputs.*</u>

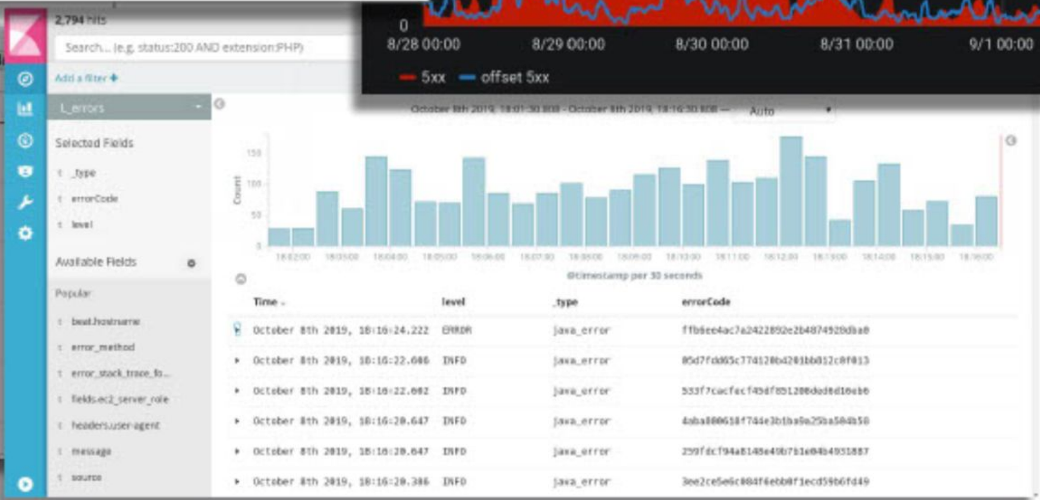
External outputs come in different forms




Trace 1a5e26df-2fc6-4a7d-a94d-d45247c9ed4e at 2019-10-08 16:51:29

name	service_name	request.error	0s 1s 2s 3s 4.427s
www.moo.com/bascule/upload	fastly		4.427s
/upload	bascule	<class 'socket.error'>	3.841s
/api/service/upload_image	external		3.790s




2,794 hits

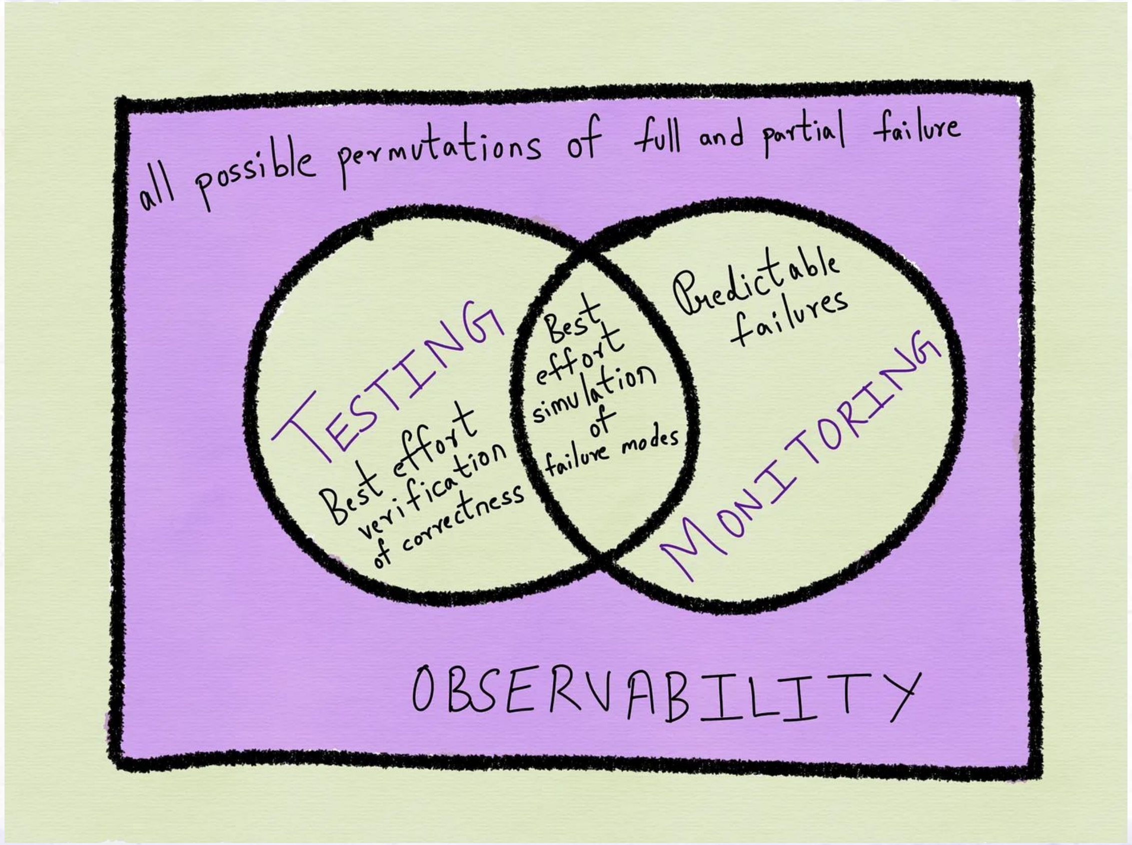
```
Time ▾                             level  _type       errorCode
October 8th 2019, 18:16:24.222     ERROR  java_error  ffb6ee4ac7a2422892e2b4874920dba0
October 8th 2019, 18:16:22.606     INFO   java_error  05d7fdd65c774120b4201bb012c0f013
October 8th 2019, 18:16:22.002     INFO   java_error  533f7cacfecf49df851200dad0d10eb6
October 8th 2019, 18:16:20.647     INFO   java_error  4aba800618f744e3b1ba9a25ba50db50
October 8th 2019, 18:16:20.647     INFO   java_error  299fdcf94a8148e49b7b1e04b4931887
October 8th 2019, 18:16:20.386     INFO   java_error  3ee2ce5e6c084f6ebb0f1ecd59b6fd49
```

@A_BANGSER

Observability

In control theory, observability is the measure of how well internal states of a system can be inferred from knowledge of its external outputs.

Observability enables understanding the "other"

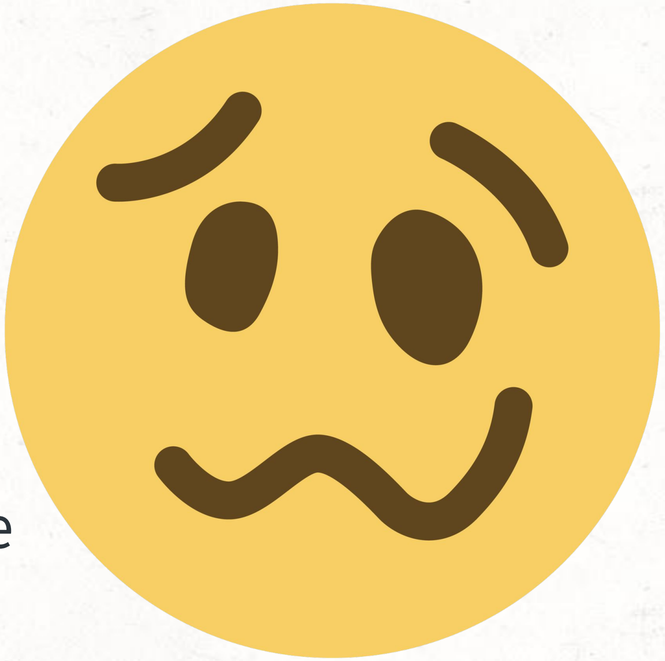

all possible permutations of full and partial failure
TESTING
Best effort verification of correctness
Best effort simulation of failure modes
Predictable failures
MONITORING
OBSERVABILITY

Cindy Sridharan

11 min read

Characteristics of valuable outputs

- raw events
- no pre-aggregation
- structured data
- arbitrarily wide events
- schema-less-ness
- high cardinality dimensions
- oriented around request lifecycle
- batched up context
- exploration over static dashboards



@A_BANGSER

Characteristics of valuable outputs

- raw events
- **no pre-aggregation**
- structured data
- **arbitrarily wide events**
- schema-less-ness
- high cardinality dimensions
- oriented around request lifecycle
- batched up context
- **exploration over static dashboards**

@A_BANGSER

Characteristics of valuable outputs

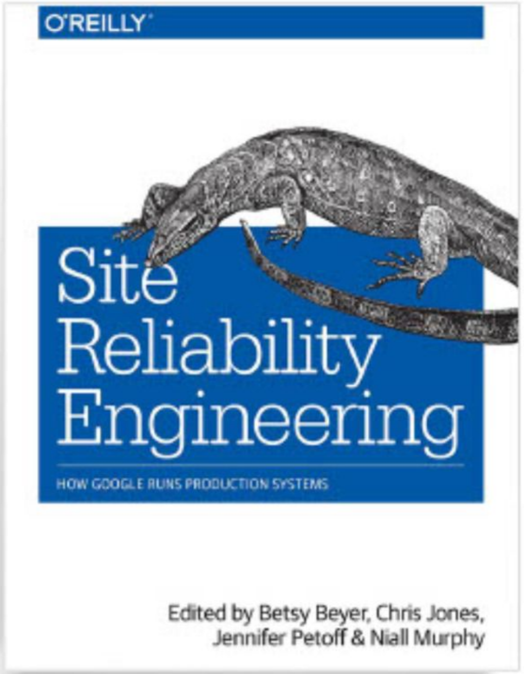
- → raw events
- → **no pre-aggregation**
- → structured data
- → **arbitrarily wide events**
- → schema-less-ness
- → high cardinality dimensions
- → oriented around request lifecycle
- → batched up context
- → **exploration over static dashboards**

@A_BANGSER

The promise of monitoring vs my reality

My rollercoaster journey with understanding metrics and pre-aggregation

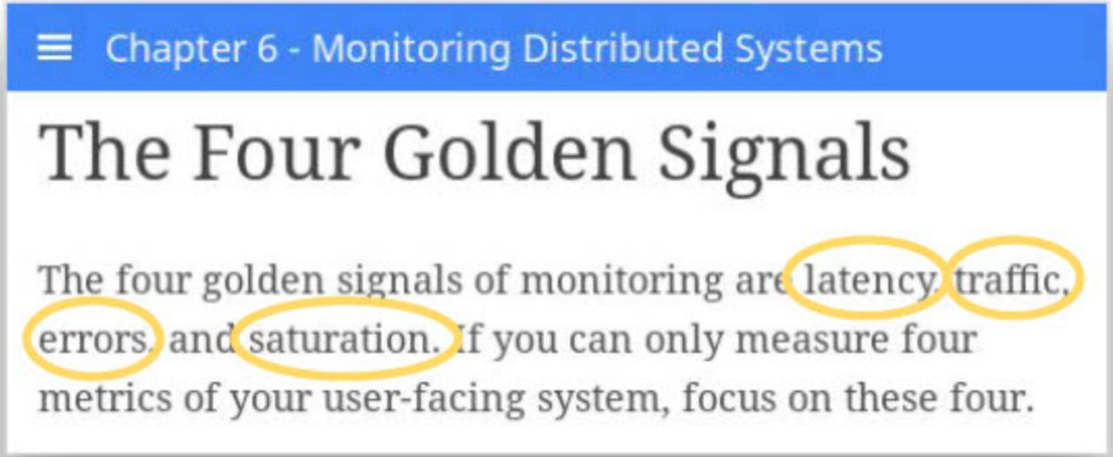
Metrics as signal for success (or failure)


O'REILLY

Site Reliability Engineering

HOW GOOGLE RUNS PRODUCTION SYSTEMS

Edited by Betsy Beyer, Chris Jones, Jennifer Petoff & Niall Murphy

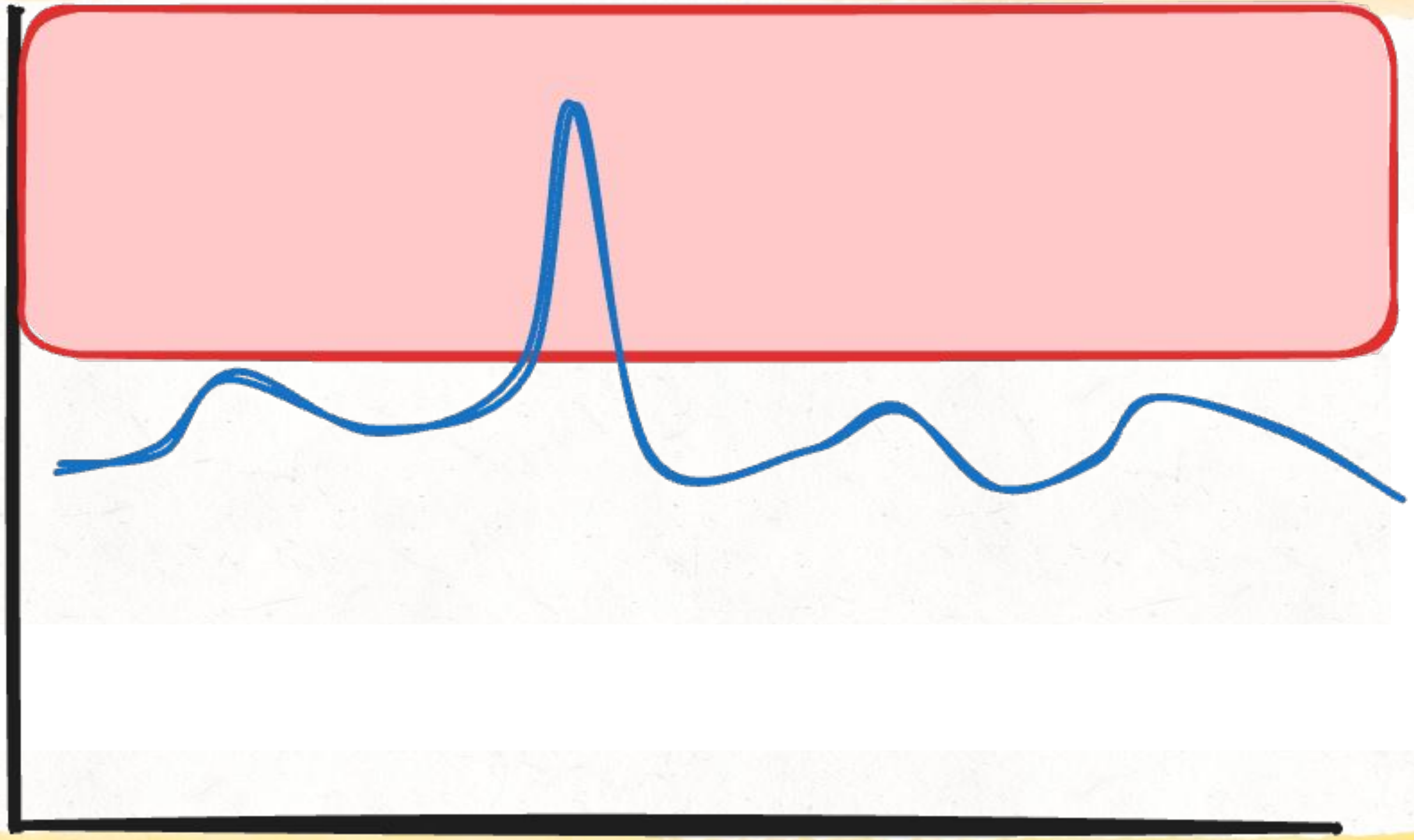

Chapter 6 - Monitoring Distributed Systems

The Four Golden Signals

The four golden signals of monitoring are latency, traffic, errors, and saturation. If you can only measure four metrics of your user-facing system, focus on these four.

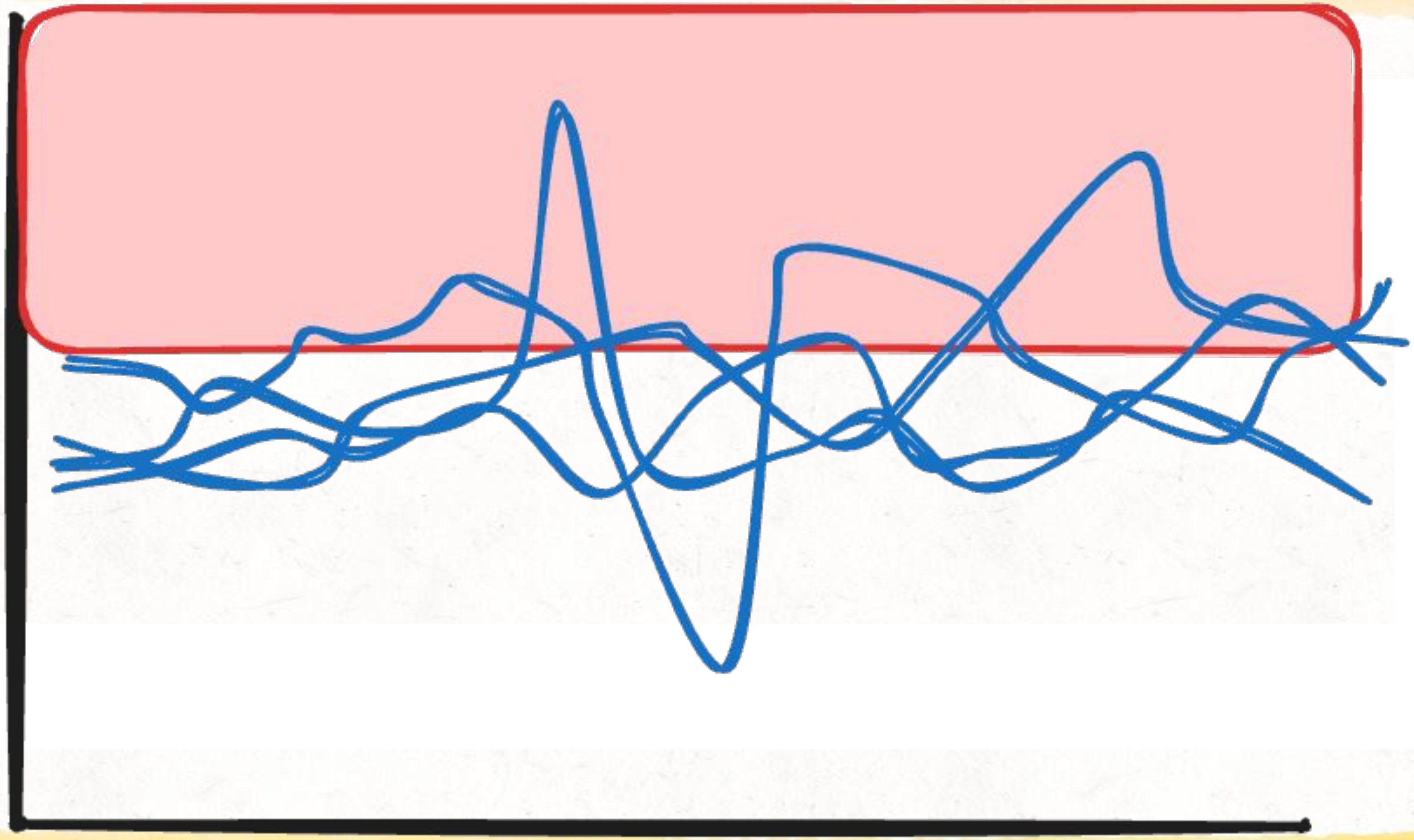
@A_BANGSER

How I imagined metrics & alerts would work



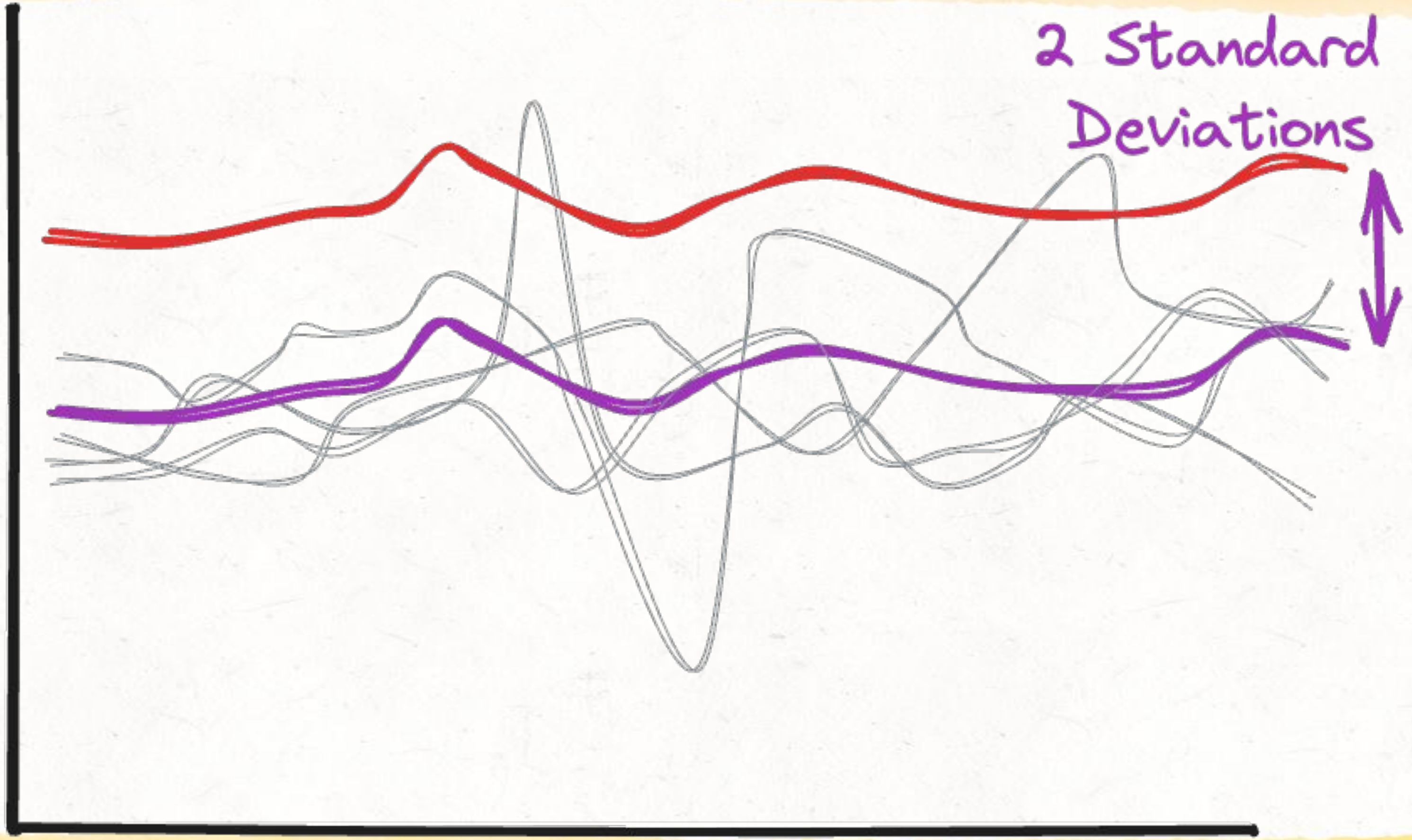
@a_bangser

What metrics actually looked like



@A_BANGSER

But trends can be helpful



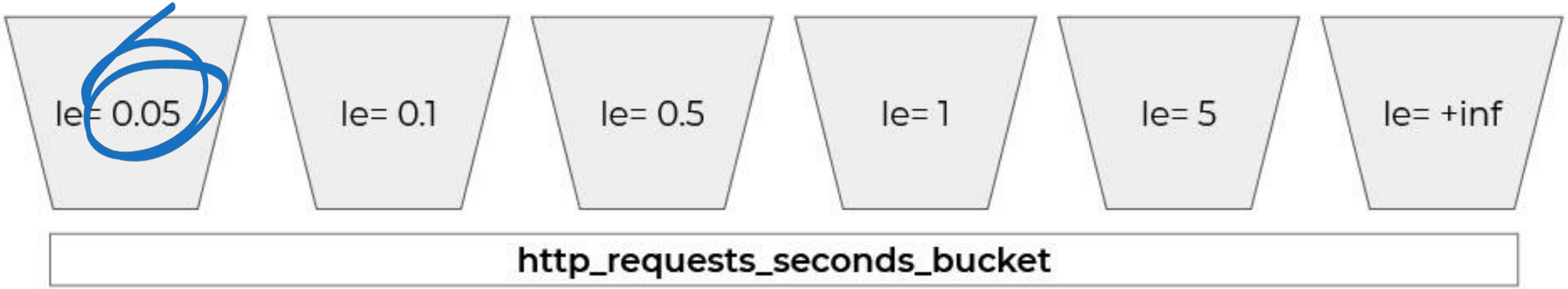
@A_BANGSER

The plan: Standardise metrics

The way metrics are stored means we had to pre-define two items:

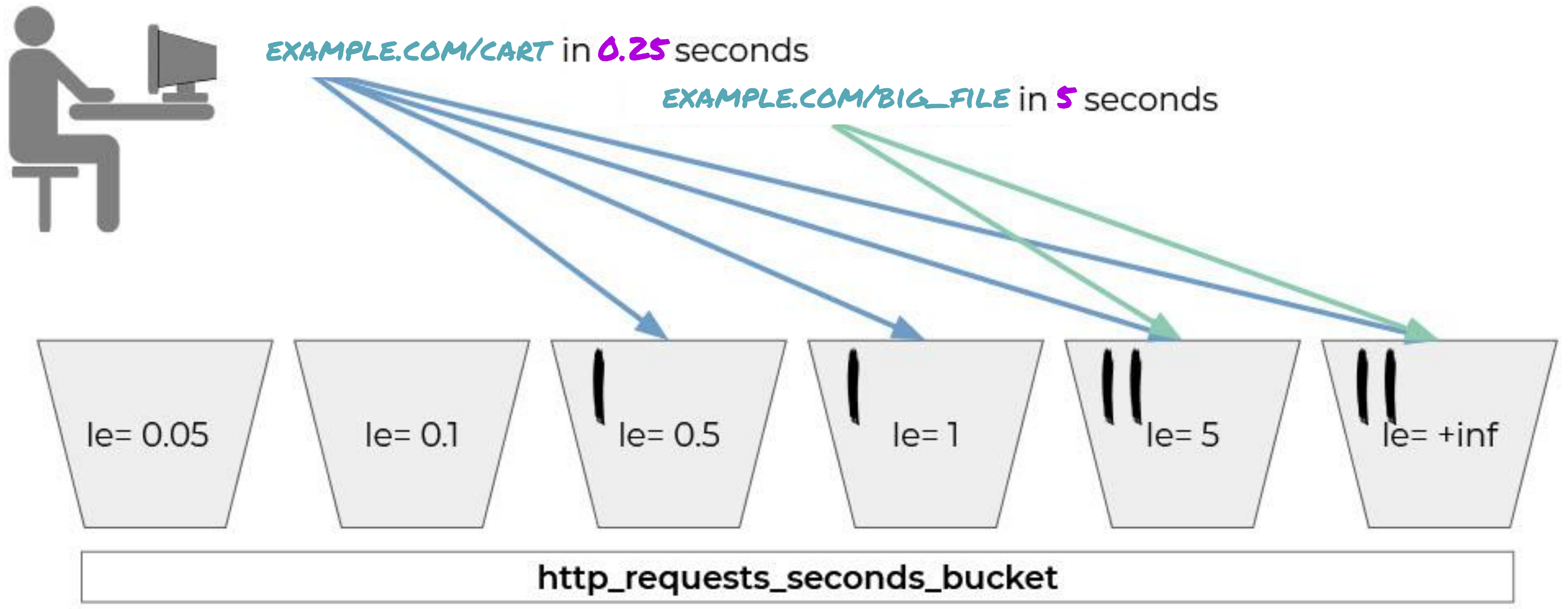
1. Buckets
2. Windows

Buckets: Aggregation of data for storage



* "le" stands for "less than or equal to"

Buckets: And then tallied


example.com/cart in 0.25 seconds
example.com/big_file in 5 seconds
le= 0.05
le= 0.1
le= 0.5
le= 1
le= 5
le= +inf
http_requests_seconds_bucket

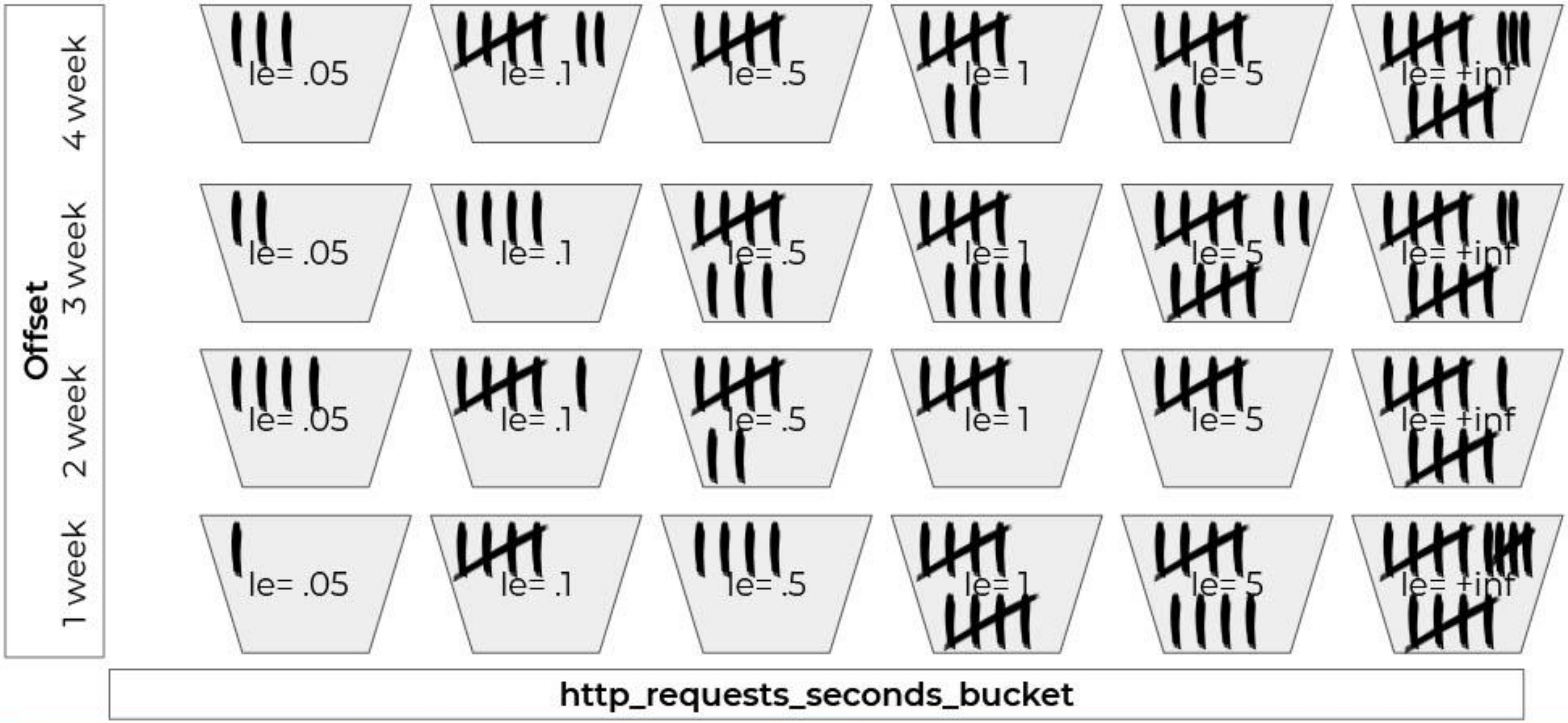
* “le” stands for “less than or equal to”

Windows: Define when the data is reviewed

Offset
1 week
2 week
3 week
4 week
http_requests_seconds_bucket

@A_BANGSER

We collected requests per bucket, per window



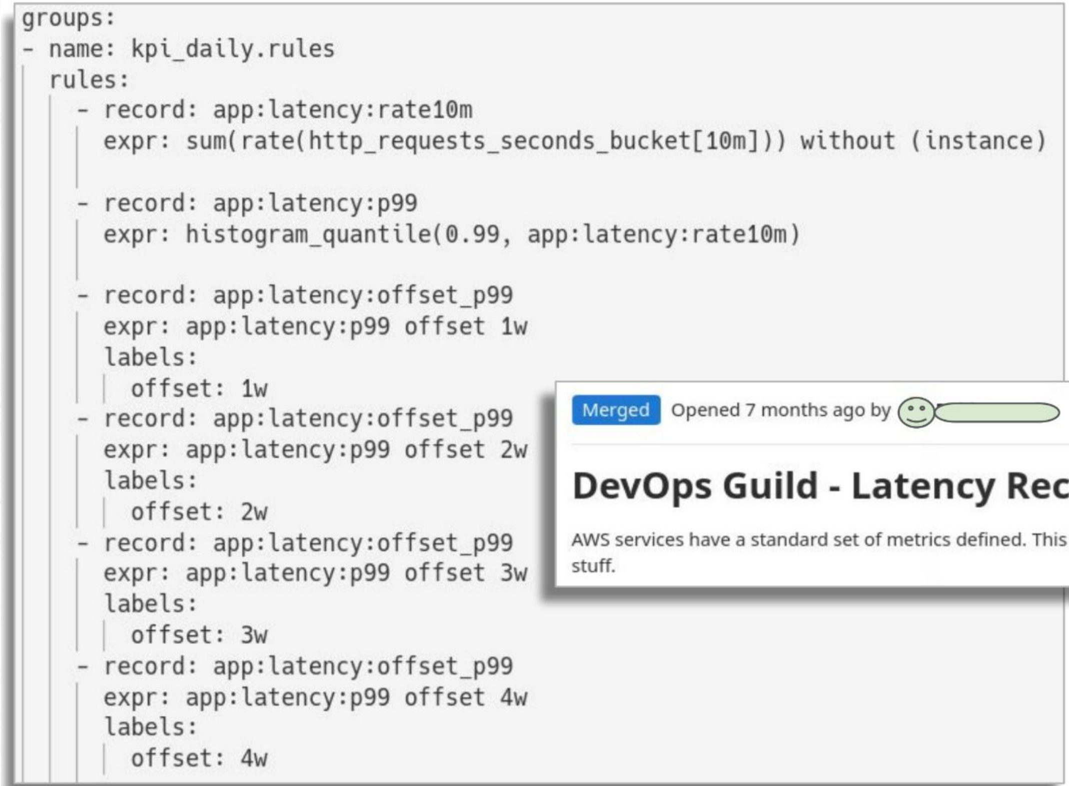
@A_BANGSER

ROLLING THIS OUT TOOK A NUMBER OF CHANGES

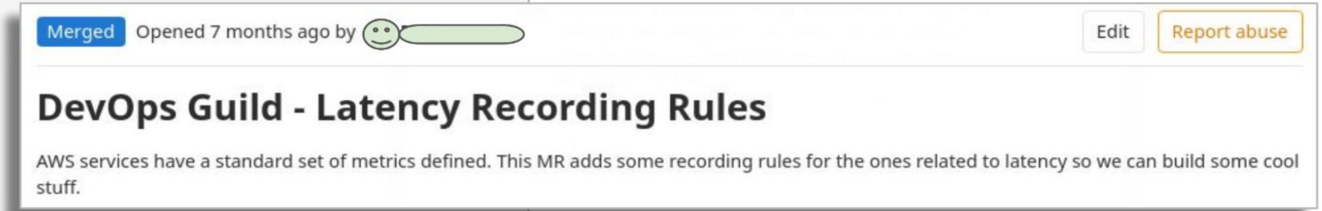
- → 40 services
- → 4 core languages
- → 3 architectural eras
- → 2 transport protocols (http and gRPC)

…and a partridge in a pear tree

We were ready to build some cool stuff



```
groups:
- name: kpi_daily.rules
  rules:
    - record: app:latency:rate10m
      expr: sum(rate(http_requests_seconds_bucket[10m])) without (instance)

    - record: app:latency:p99
      expr: histogram_quantile(0.99, app:latency:rate10m)

    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 1w
      labels:
        offset: 1w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 2w
      labels:
        offset: 2w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 3w
      labels:
        offset: 3w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 4w
      labels:
        offset: 4w
```



Merged Opened 7 months ago by [redacted]

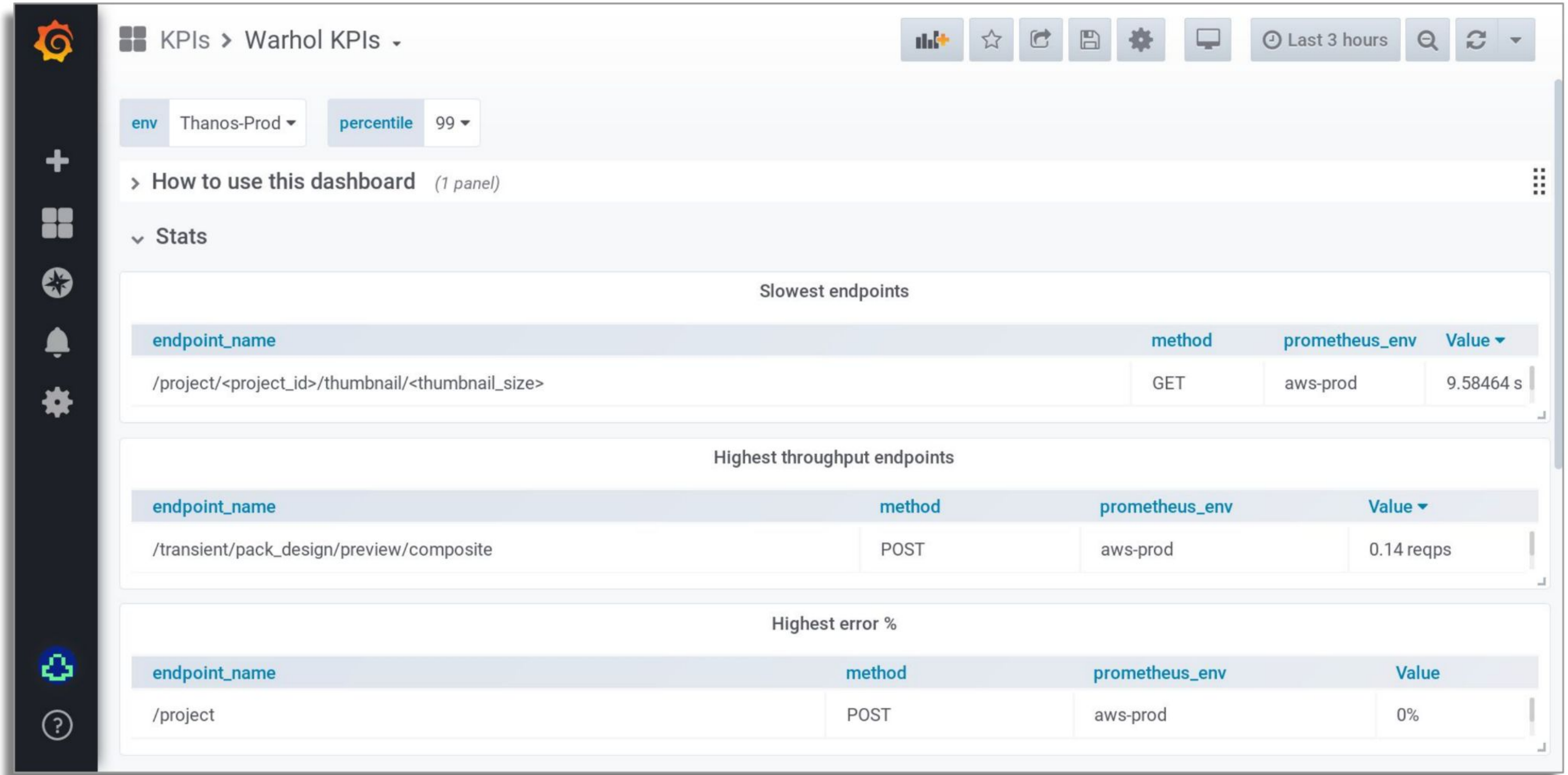
Edit Report abuse

DevOps Guild - Latency Recording Rules

AWS services have a standard set of metrics defined. This MR adds some recording rules for the ones related to latency so we can build some cool stuff.

@A_BANGSER

Consistency generated a ton of learning



KPIs > Warhol KPIs

Last 3 hours

env Thanos-Prod

percentile 99

How to use this dashboard (1 panel)

Stats

Slowest endpoints

endpoint_name	method	prometheus_env	Value
/project/<project_id>/thumbnail/<thumbnail_size>	GET	aws-prod	9.58464 s

Highest throughput endpoints

endpoint_name	method	prometheus_env	Value
/transient/pack_design/preview/composite	POST	aws-prod	0.14 reqps

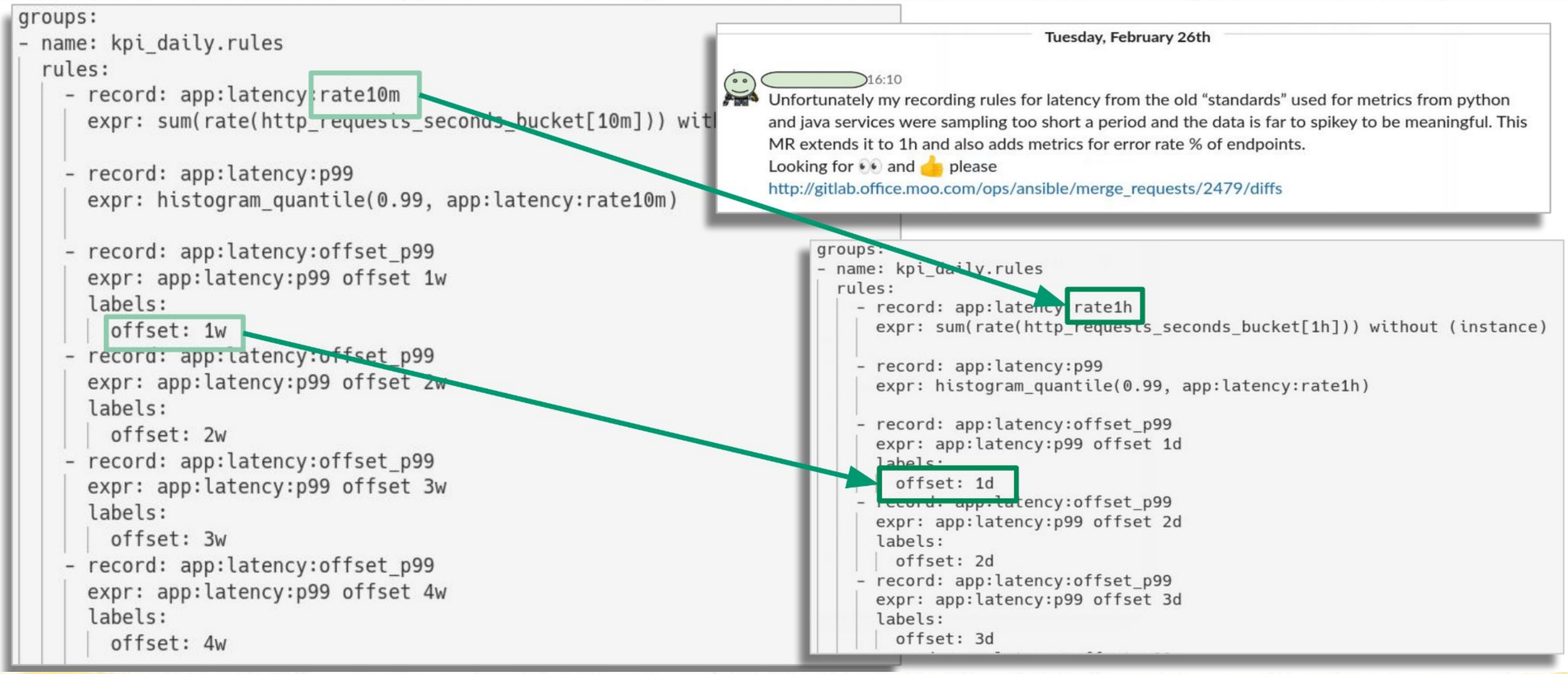
Highest error %

endpoint_name	method	prometheus_env	Value
/project	POST	aws-prod	0%

@A_BANGSER

But...

Data collection required assumptions. And we weren't always correct.



```
groups:
- name: kpi_daily.rules
  rules:
    - record: app:latency:rate10m
      expr: sum(rate(http_requests_seconds_bucket[10m])) wit

    - record: app:latency:p99
      expr: histogram_quantile(0.99, app:latency:rate10m)

    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 1w
      labels:
        offset: 1w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 2w
      labels:
        offset: 2w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 3w
      labels:
        offset: 3w
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 4w
      labels:
        offset: 4w
```

Tuesday, February 26th

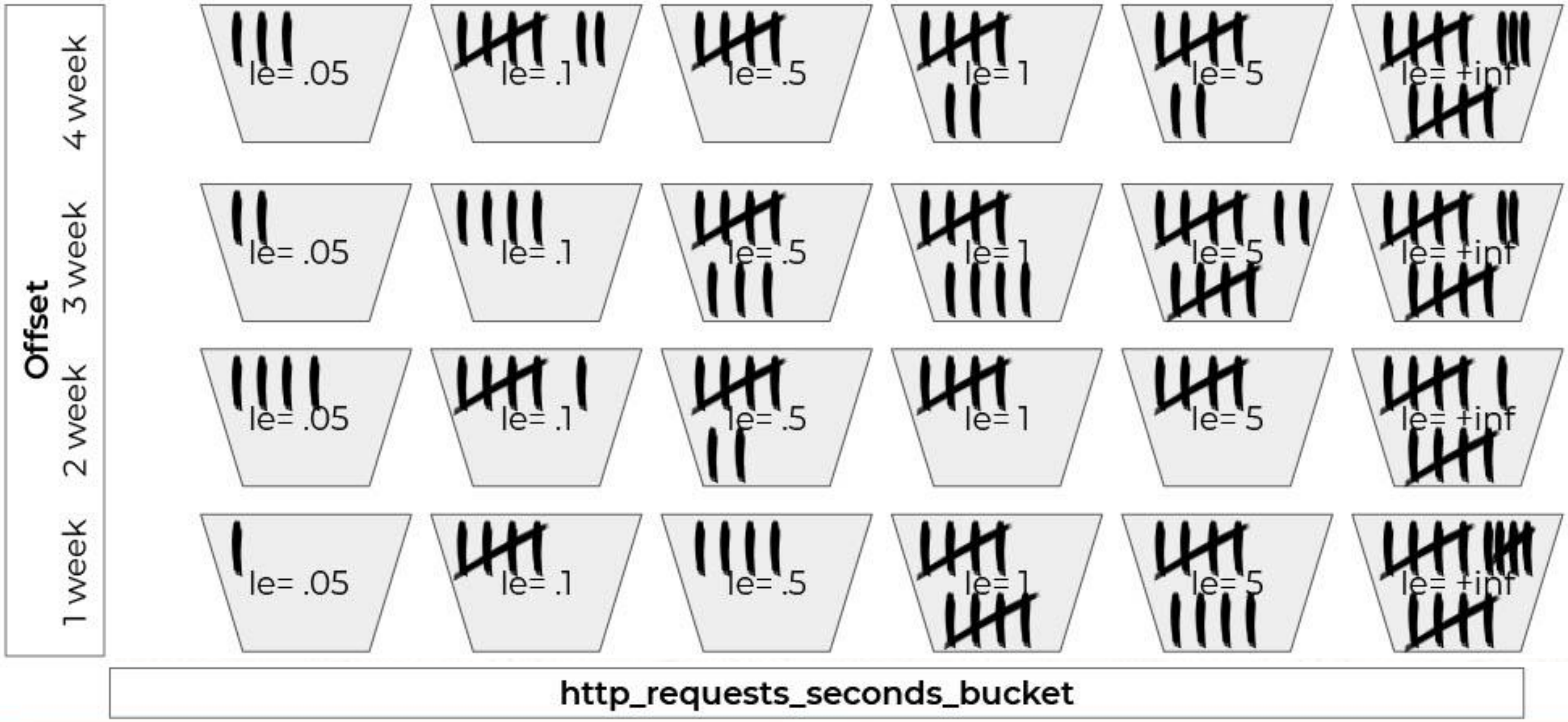
[redacted] 16:10

Unfortunately my recording rules for latency from the old "standards" used for metrics from python and java services were sampling too short a period and the data is far to spikey to be meaningful. This MR extends it to 1h and also adds metrics for error rate % of endpoints.
Looking for 👀 and 👍 please
http://gitlab.office.moo.com/ops/ansible/merge_requests/2479/diffs

```
groups:
- name: kpi_daily.rules
  rules:
    - record: app:latency:rate1h
      expr: sum(rate(http_requests_seconds_bucket[1h])) without (instance)

    - record: app:latency:p99
      expr: histogram_quantile(0.99, app:latency:rate1h)

    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 1d
      labels:
        offset: 1d
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 2d
      labels:
        offset: 2d
    - record: app:latency:offset_p99
      expr: app:latency:p99 offset 3d
      labels:
        offset: 3d
```

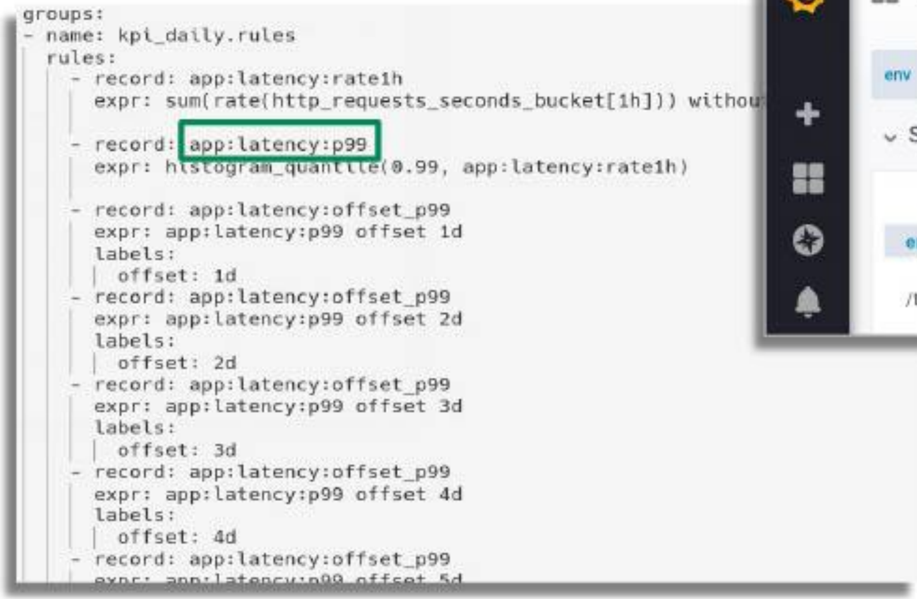
@A_BANGSER

And we ended up throwing it all away

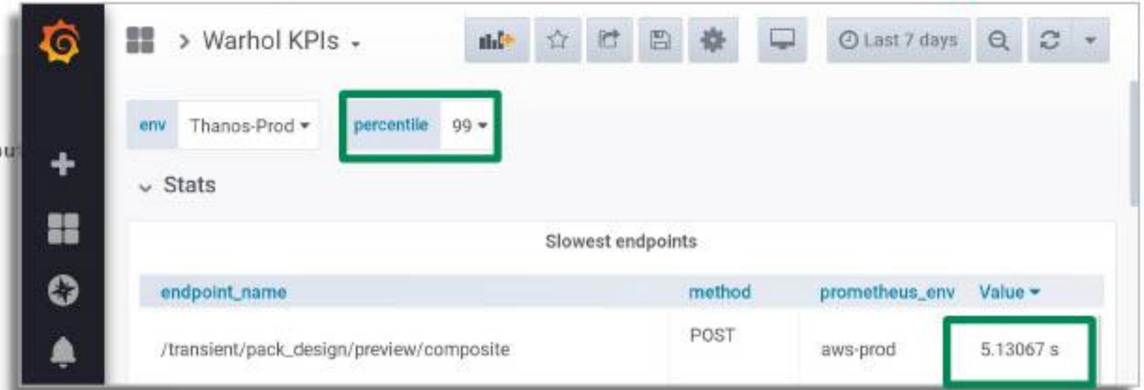


@A_BANGSER

At least once updated, we are set right?



```
groups:
- name: kpi_daily.rules
  rules:
  - record: app:latency:rate1h
    expr: sum(rate(http_requests_seconds_bucket[1h])) withou

  - record: app:latency:p99
    expr: histogram_quantile(0.99, app:latency:rate1h)

  - record: app:latency:offset_p99
    expr: app:latency:p99 offset 1d
    labels:
      offset: 1d
  - record: app:latency:offset_p99
    expr: app:latency:p99 offset 2d
    labels:
      offset: 2d
  - record: app:latency:offset_p99
    expr: app:latency:p99 offset 3d
    labels:
      offset: 3d
  - record: app:latency:offset_p99
    expr: app:latency:p99 offset 4d
    labels:
      offset: 4d
  - record: app:latency:offset_p99
    expr: app:latency:p99 offset 5d
```



env: Thanos-Prod
percentile: 99

Stats

Slowest endpoints

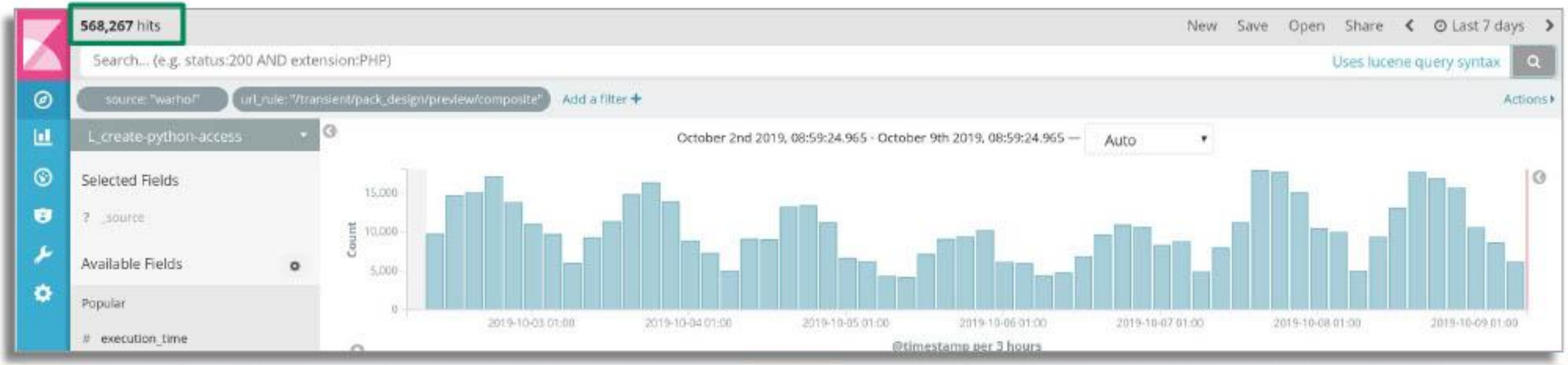
endpoint_name	method	prometheus_env	Value
/transient/pack_design/preview/composite	POST	aws-prod	5.13067 s

That depends...
are you ready for the truth?
We weren't.

But we found it.
We asked ourselves...

"what is the user impact of the 99th percentile"

1% is small right? Nope! 56k users!



568,267 hits

New Save Open Share Last 7 days

Search... (e.g. status:200 AND extension:PHP)

Uses lucene query syntax

source: "warhol"

url_rule: "/transient/pack_design/preview/composite"

Add a filter +

Actions

L_create-python-access

October 2nd 2019, 08:59:24.965 - October 9th 2019, 08:59:24.965 — Auto

Selected Fields

? _source

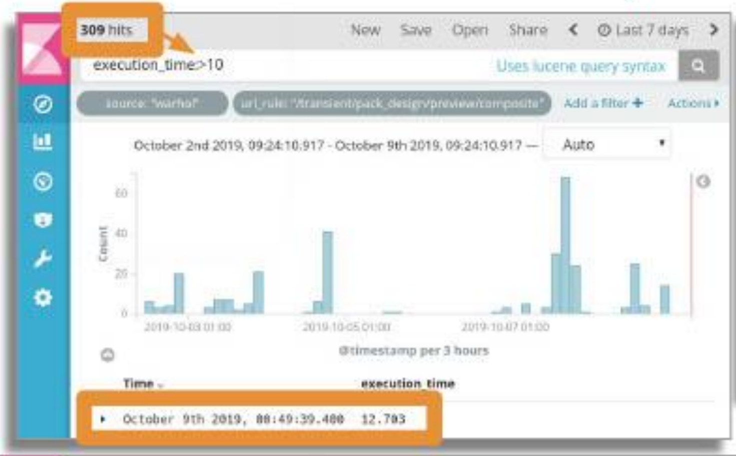
Available Fields

Popular

execution_time

@A_BANGSER

But 5s isn't <u>so</u> bad. At least it isn't like...10s!



309 hits

New Save Open Share ‹ Last 7 days ›

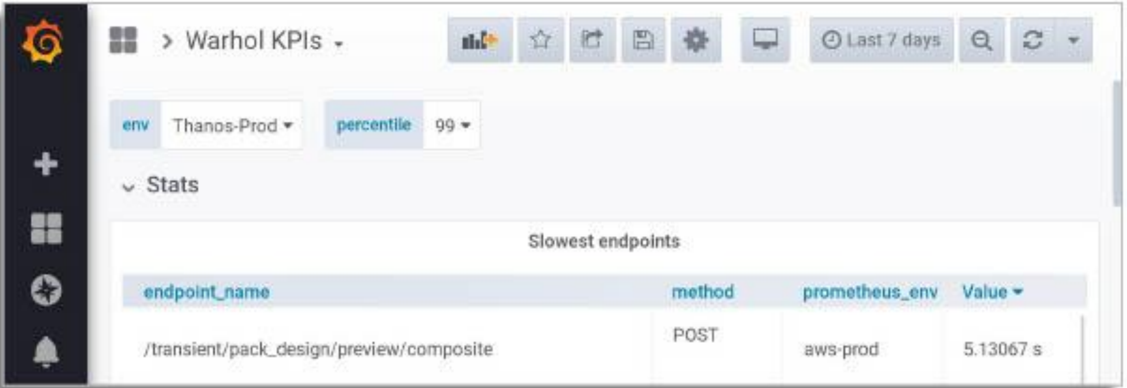
execution_time>10

Uses lucene query syntax

source: "warhol" url_rule: "/transient/pack_design/preview/composite" Add a filter + Actions ›

October 2nd 2019, 09:24:10.917 - October 9th 2019, 09:24:10.917 — Auto

Time	execution_time
October 9th 2019, 08:49:39.400	12.703



Warhol KPIs

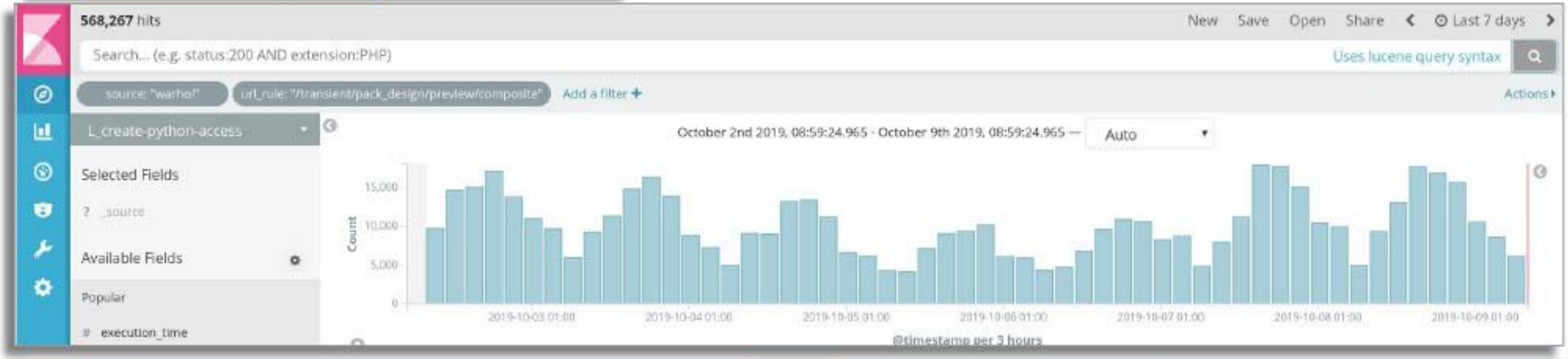
Last 7 days

env Thanos-Prod   percentile 99

Stats

Slowest endpoints

endpoint_name	method	prometheus_env	Value
/transient/pack_design/preview/composite	POST	aws-prod	5.13067 s



568,267 hits

New Save Open Share ‹ Last 7 days ›

Search... (e.g. status:200 AND extension:PHP)

Uses lucene query syntax

source: "warhol" url_rule: "/transient/pack_design/preview/composite" Add a filter + Actions ›

L_create-python-access

Selected Fields

? _source

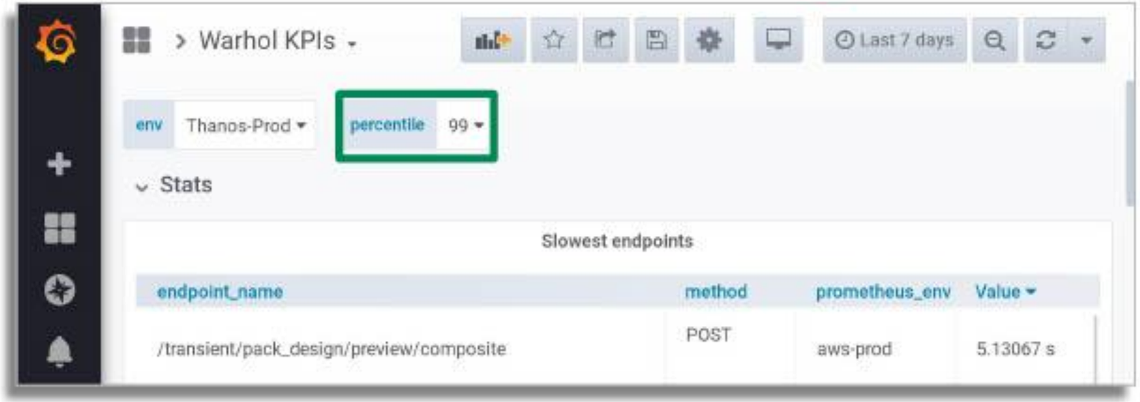
Available Fields

Popular

execution_time

October 2nd 2019, 08:59:24.965 - October 9th 2019, 08:59:24.965 — Auto

Turns out, metrics sometimes have to guess



Warhol KPIs

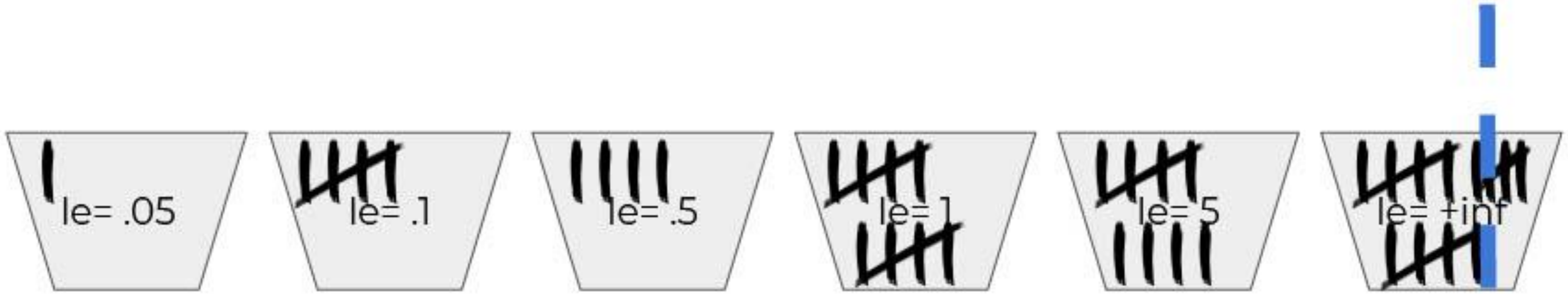
env Thanos-Prod

percentile 99

Stats

Slowest endpoints

endpoint_name	method	prometheus_env	Value
/transient/pack_design/preview/composite	POST	aws-prod	5.13067 s



While consistent metrics provided a step forward with trending...

In retrospect, this was not mature <u>observability</u>

@a_bangser

Why avoid pre-aggregation?

You can never regain original context and detail. You will only ever answer predetermined questions.

@A_BANGSER

Characteristics of valuable outputs

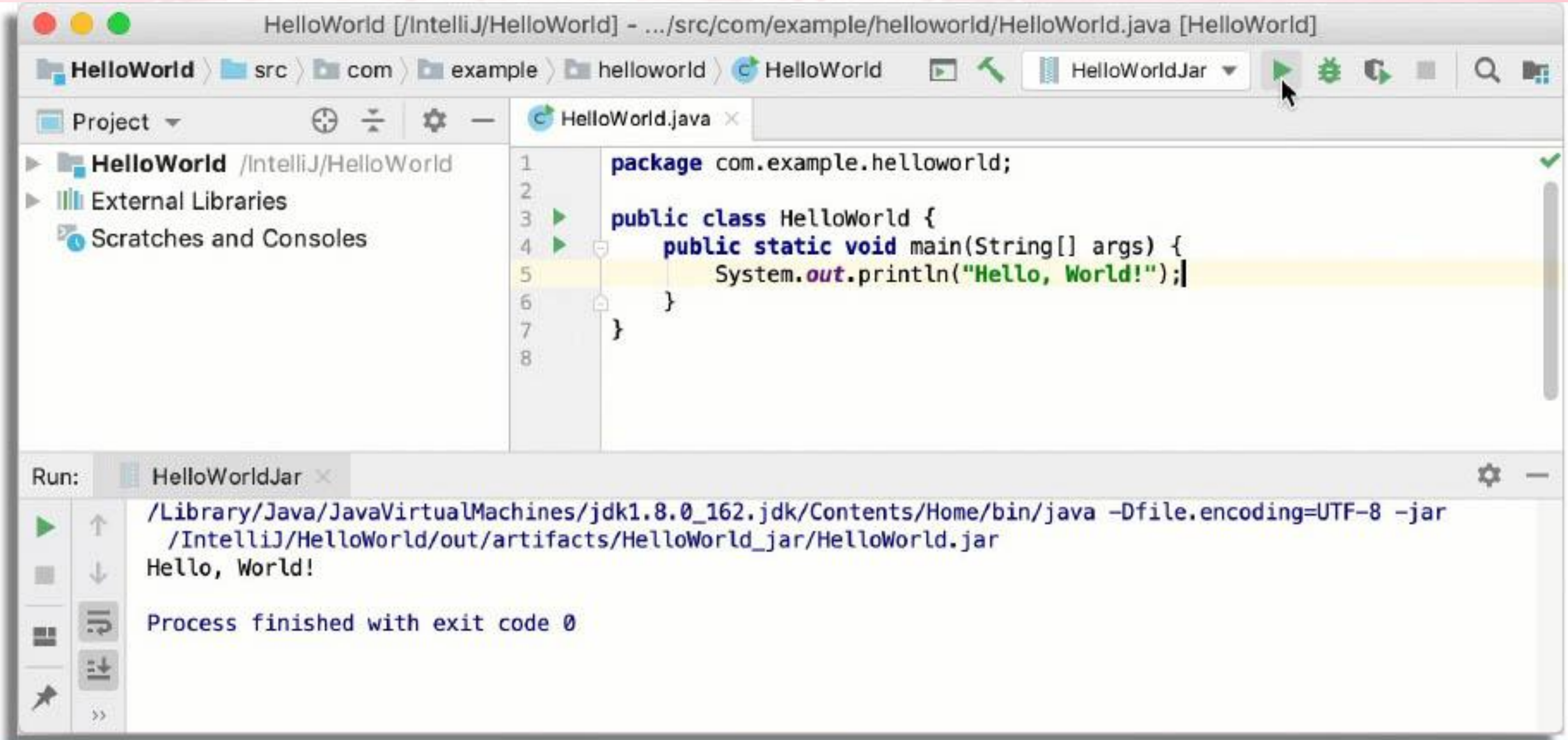
- → raw events
- → **no pre-aggregation**
- → structured data
- → **arbitrarily wide events**
- → schema-less-ness
- → high cardinality dimensions
- → oriented around request lifecycle
- → batched up context
- → **exploration over static dashboards**

@A_BANGSER

Data is not the same as information

When collecting data, think first about how you will turn that into useful information through queries

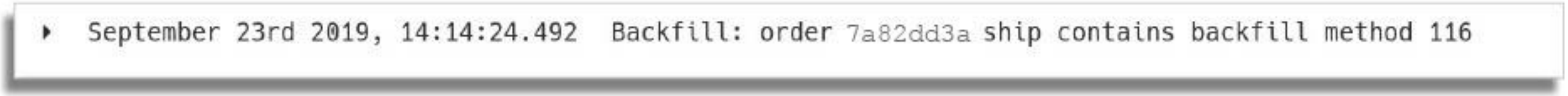
Humans have always logged



```java
package com.example.helloworld;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_162.jdk/Contents/Home/bin/java -Dfile.encoding=UTF-8 -jar
  /IntelliJ/HelloWorld/out/artifacts/HelloWorld_jar/HelloWorld.jar
Hello, World!

Process finished with exit code 0
```

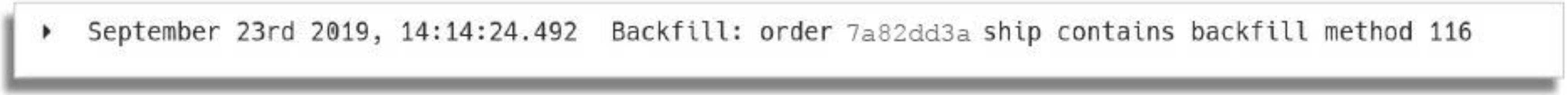
We have also always wanted more



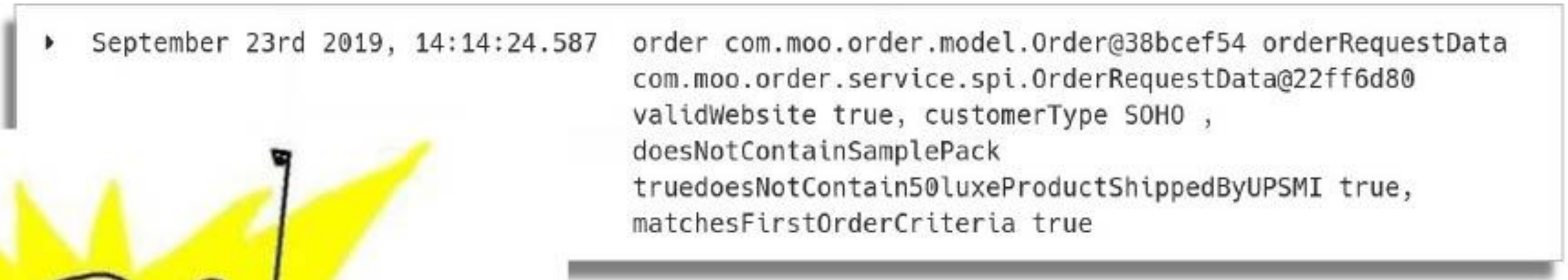
```
▸  September 23rd 2019, 14:14:24.492  Backfill: order 7a82dd3a ship contains backfill method 116
```

@A_BANGSER

... AND MORE



```
▸  September 23rd 2019, 14:14:24.492  Backfill: order 7a82dd3a ship contains backfill method 116
```

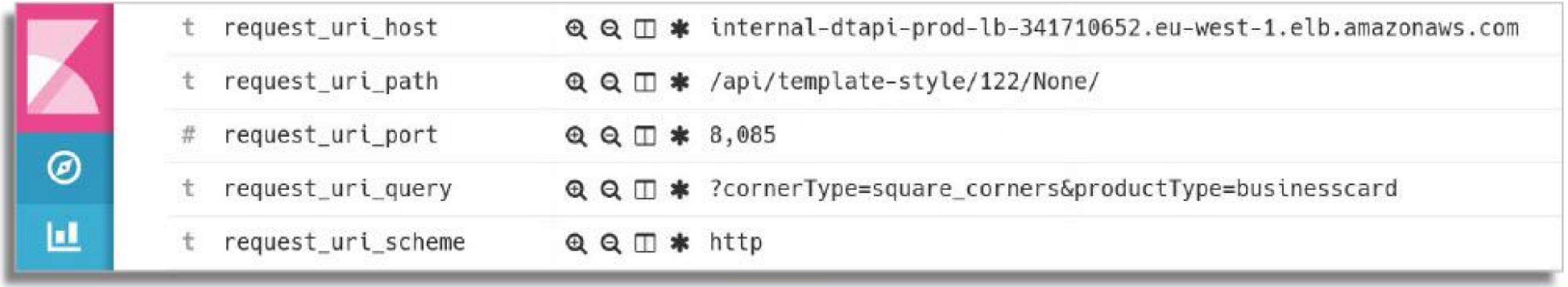


```
▸  September 23rd 2019, 14:14:24.587  order com.moo.order.model.Order@38bcef54 orderRequestData
                                      com.moo.order.service.spi.OrderRequestData@22ff6d80
                                      validWebsite true, customerType SOHO ,
                                      doesNotContainSamplePack
                                      truedoesNotContain50luxeProductShippedByUPSMI true,
                                      matchesFirstOrderCriteria true
```

Structure came later

logstash

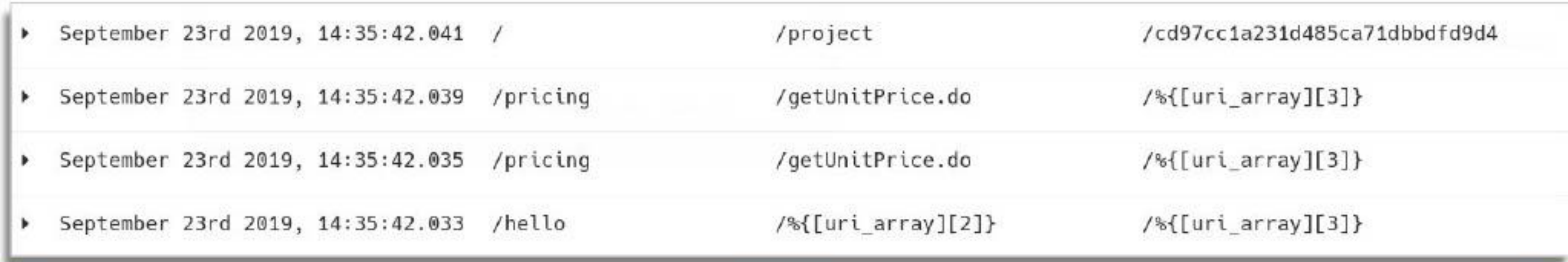
```
grok {
  match => [
   "Request",
    "%{URIPROTO:request_uri_scheme}://
    %{HOSTNAME:request_uri_host}(?::%{POSINT:request_uri_port})
    ?%{URIPATH:request_uri_path}(?:%{URIPARAM:request_uri_query})?"
  ]}
}
```



```
t  request_uri_host     internal-dtapi-prod-lb-341710652.eu-west-1.elb.amazonaws.com
t  request_uri_path     /api/template-style/122/None/
#  request_uri_port     8,085
t  request_uri_query    ?cornerType=square_corners&productType=businesscard
t  request_uri_scheme   http
```

...AND OF COURSE WE WANTED MORE

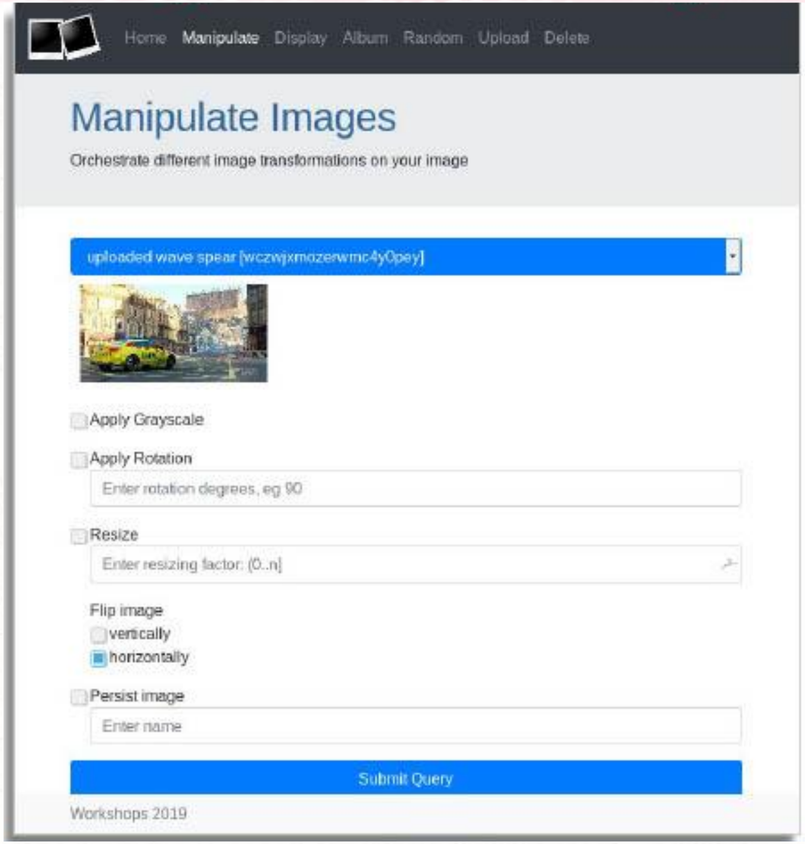
```
mutate {
  split => { "uri_array" => "/"}
  add_field => {
  "uri_root" => ["/%{[uri_array][1]}"]
  "uri_first" => ["/%{[uri_array][2]}"]
  "uri_second" => ["/%{[uri_array][3]}"]
  "uri_root_first" => "%{uri_root}%{uri_first}"
  "uri_root_second" => "%{uri_root}%{uri_first}%{uri_second}"
 }
```



```
▸  September 23rd 2019, 14:35:42.041  /                   /project                 /cd97cc1a231d485ca71dbbdfd9d4
▸  September 23rd 2019, 14:35:42.039  /pricing            /getUnitPrice.do         /%{[uri_array][3]}
▸  September 23rd 2019, 14:35:42.035  /pricing            /getUnitPrice.do         /%{[uri_array][3]}
▸  September 23rd 2019, 14:35:42.033  /hello              /%{[uri_array][2]}       /%{[uri_array][3]}
```

Wait a second...
What even is logging?

Imagine an image manipulation app



Manipulate Images

Orchestrate different image transformations on your image

uploaded wave spear [wczwjxmozerwmc4y0pey]

Apply Grayscale

Apply Rotation

Enter rotation degrees, eg 90

Resize

Enter resizing factor: (0..n]

Flip image

vertically

[✓ horizontally]

Persist image

Enter name

Submit Query

Workshops 2019

@A_BANGSER

A DEEP DIVE ON HOW LOGS ARE WRITTEN

```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                  @RequestParam(value="vertical") Boolean vertical,
                                  @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flip.", file.getContentType());
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

A DEEP DIVE ON HOW LOGS ARE WRITTEN

```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                 @RequestParam(value="vertical") Boolean vertical,
                                 @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flLOGGER.info("Receiving image to flip.", file.getC
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

A DEEP DIVE ON HOW LOGS ARE WRITTEN

```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                 @RequestParam(value="vertical") Boolean vertical,
                                 @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flip.", file.getContentType());
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

A DEEP DIVE ON HOW LOGS ARE WRITTEN

```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                @RequestParam(value="vertical") Boolean vertical,
                                @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flip.", file.getContentType());
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

A DEEP DIVE ON HOW LOGS ARE WRITTEN

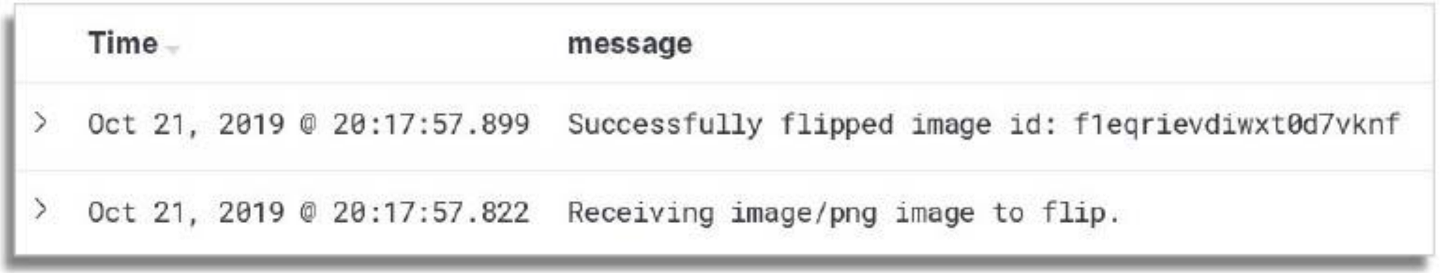
```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                 @RequestParam(value="vertical") Boolean vertical,
                                 @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flip.", file.getContentType());
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped iLOGGER.info("Successfully flipped image id: {}",
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

A DEEP DIVE ON HOW LOGS ARE WRITTEN

```
@PostMapping ("flip")
public ResponseEntity flipImage (@RequestParam("image") MultipartFile file,
                                 @RequestParam(value="vertical") Boolean vertical,
                                 @RequestParam(value="horizontal") Boolean horizontal
{
  LOGGER.info("Receiving image to flip.", file.getContentType());
  byte[] flippedImage = imageService.flip (file, vertical, horizontal);
  if (flippedImage == null) {
    new ResponseEntity<>("Failed to flip image", HttpStatus.INTERNAL_SERVER_ERROR);
  }
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  return new ResponseEntity<> (flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

Resulting log outputs



```
     Time              message
>    Oct 21, 2019 @ 20:17:57.899   Successfully flipped image id: f1eqrievdiwxt0d7vknf
>    Oct 21, 2019 @ 20:17:57.822   Receiving image/png image to flip.
```

@A_BANGSER

Sure,
but what else is there?

In contrast, how events are written

```
@PostMapping("flip")
public ResponseEntity flipImage(...) {
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
...
  LOGGER.info("Receiving {} image to flip.", file.getContentType () );
  byte[] flippedImage imageService.flip(file, vertical, horizontal);
...
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  EVENT.addField("action.success", "true");
  return new ResponseEntity<>(flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

IN CONTRAST, HOW EVENTS ARE WRITTEN

```
@PostMapping("flip")
public ResponseEntity flipImage(...) {
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
...
  LOGGER.info("Receiving {} image to flip.", fileLOGGER.info("Receiving {} image to flip.", file.get
  byte[] flippedImage imageService.flip(file, vertical, horizontal);
...
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  EVENT.addField("action.success", "true");
  return new ResponseEntity<>(flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

IN CONTRAST, HOW EVENTS ARE WRITTEN

```
@PostMapping("flip")
public ResponseEntity flipImage(...) {
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
...
  LOGGER.info("Receiving {} image to flip.", file.getContentType () );
  byte[] flippedImage imageService.flip(file, vertical, horizontal);
...
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  EVENT.addField("action.success", "true");
  return new ResponseEntity<>(flippedImage, headers, HttpStatus.OK);
}
```

@A_BANGSER

IN CONTRAST, HOW EVENTS ARE WRITTEN

```
@PostMapping("flip")
public ResponseEntity flipImage(...) {
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
...
  LOGGER.info("Receiving {} image to flip.", file.getContentType () );
  byte[] flippedImage imageService.flip(file, vertical, horizontal);
...
  LOGGER.info("Successfully flipped image id: {}",LOGGER.info("Successfully flipped image id: {}", f
  EVENT.addField("action.success", "true");
  return new ResponseEntity<>(flippedImage, headers, HttpStatus.OK);
}
```

@A_BA ER

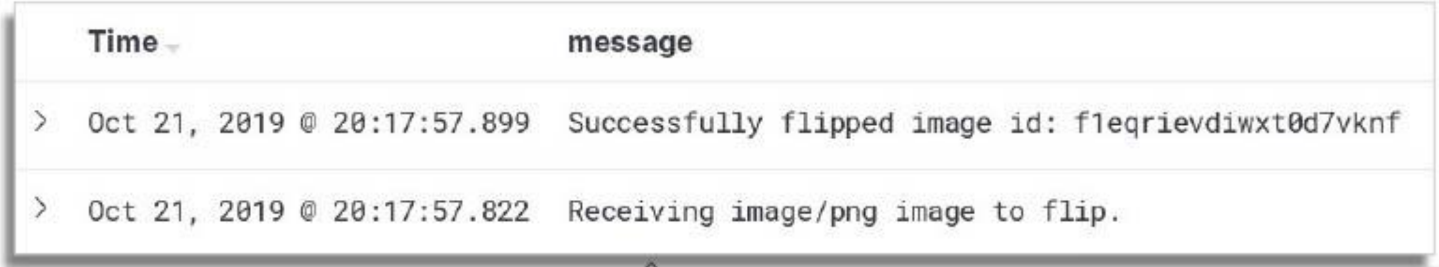
In contrast, how events are written

```java
@PostMapping("flip")
public ResponseEntity flipImage(...) {
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
...
  LOGGER.info("Receiving {} image to flip.", file.getContentType () );
  byte[] flippedImage imageService.flip(file, vertical, horizontal);
...
  LOGGER.info("Successfully flipped image id: {}", file.getId());
  EVENT.addField("action.success", "true");
                                   edImage, headers, HttpStatus.OK);
}
```

```java
  EVENT.addField("content.type", file.getContentType() );
  EVENT.addField("action", "flip");
  EVENT.addField("image_id", file.getId());
  EVENT.addField("flip_vertical", vertical);
  EVENT.addField("flip_horizontal", horizontal);
  EVENT.addField("action.success", "true");
```

@A_BANGSER

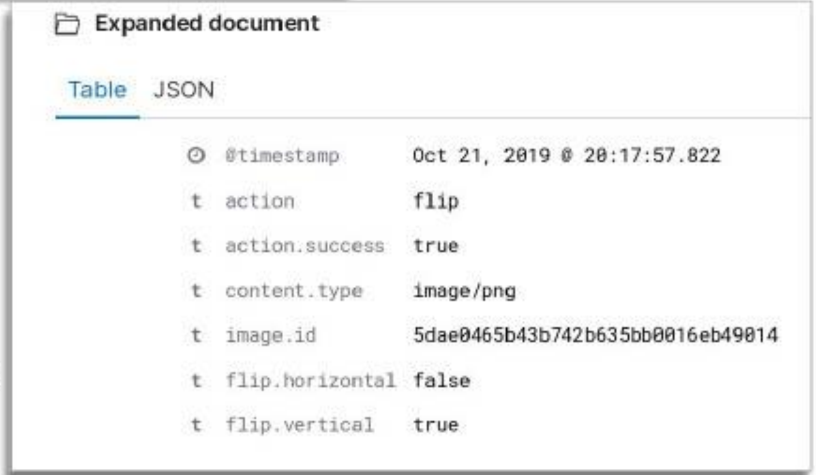
Comparing log and event output



```
Time        message
> Oct 21, 2019 @ 20:17:57.899   Successfully flipped image id: f1eqrievdiwxt0d7vknf
> Oct 21, 2019 @ 20:17:57.822   Receiving image/png image to flip.
```

Multiple logs

A single event

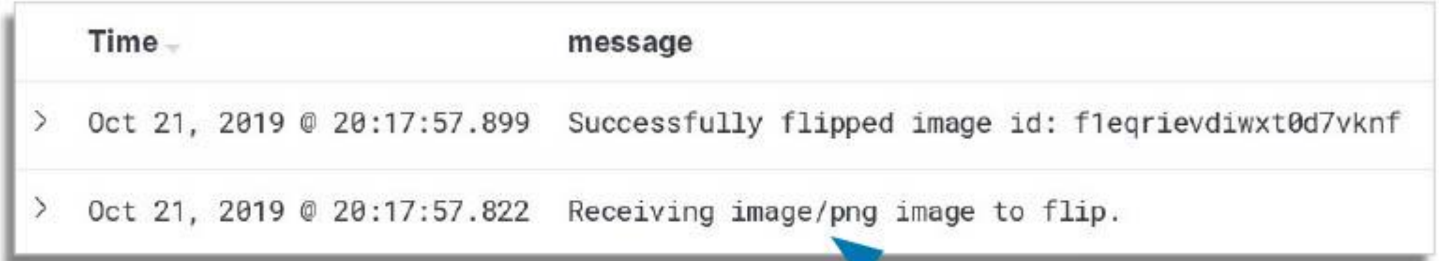


Expanded document

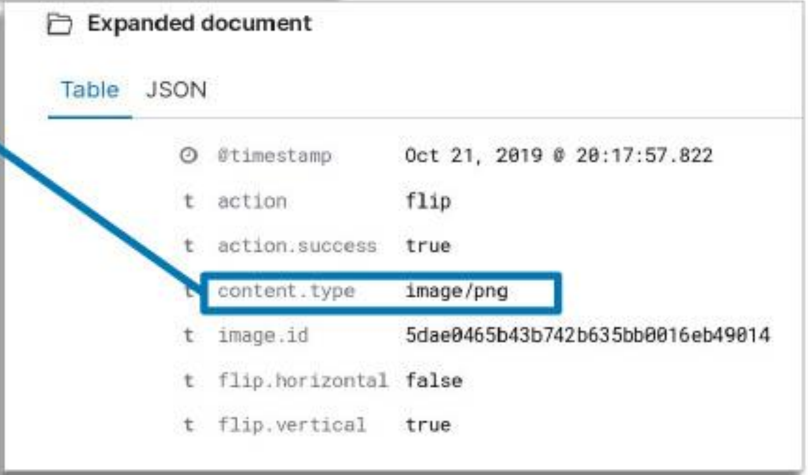
Table JSON

```
@timestamp        Oct 21, 2019 @ 20:17:57.822
t action          flip
t action.success  true
t content.type    image/png
t image.id        5dae0465b43b742b635bb0016eb49014
t flip.horizontal false
t flip.vertical   true
```

Key:Value makes data more accessible



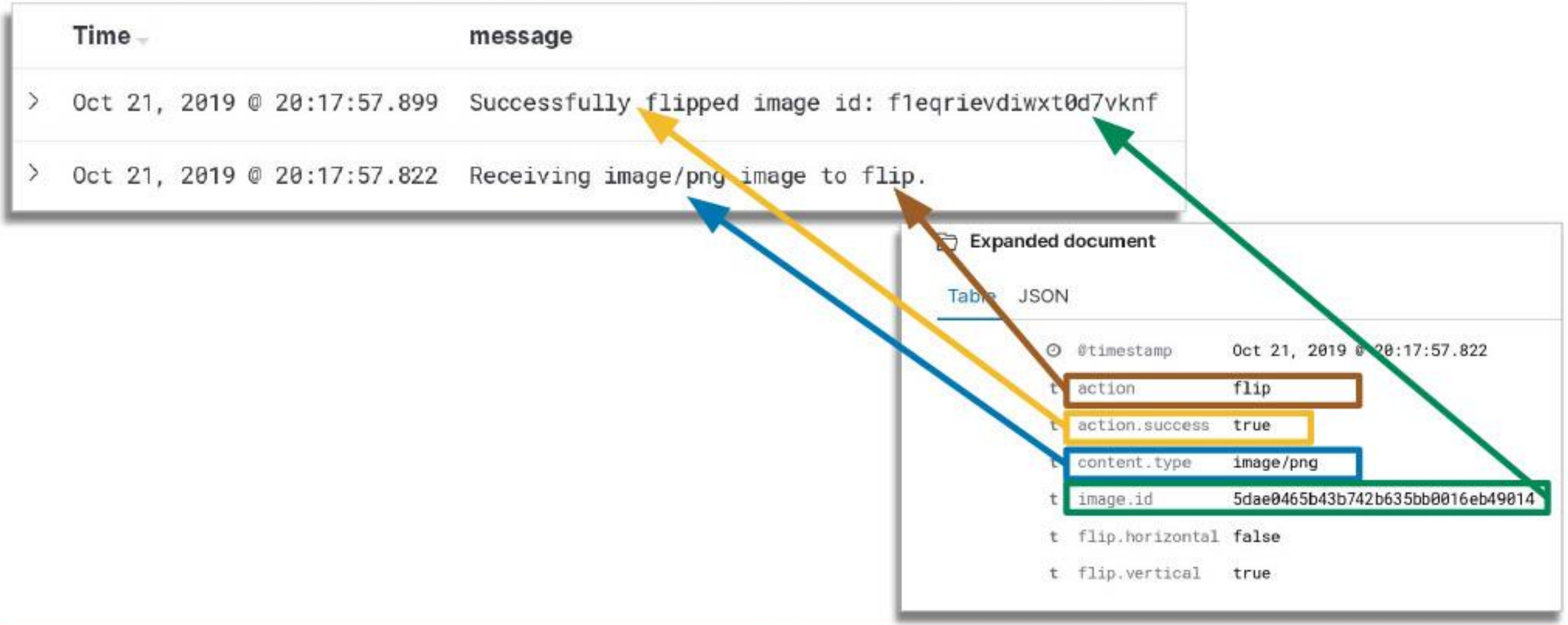
	Time	message
>	Oct 21, 2019 @ 20:17:57.899	Successfully flipped image id: f1eqrievdiwxt0d7vknf
>	Oct 21, 2019 @ 20:17:57.822	Receiving image/png image to flip.



Expanded document

	@timestamp	Oct 21, 2019 @ 20:17:57.822
t	action	flip
t	action.success	true
t	content.type	image/png
t	image.id	5dae0465b43b742b635bb0016eb49014
t	flip.horizontal	false
t	flip.vertical	true

...AND ALL WITHIN THE SAME CONTEXT



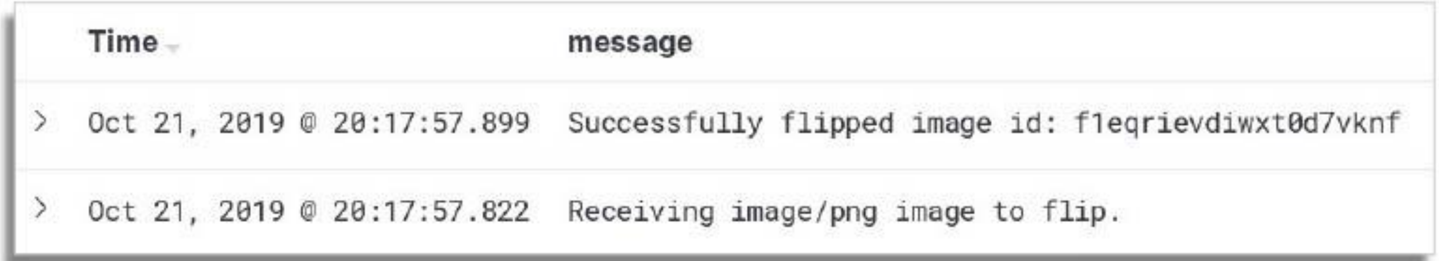
	Time	message
>	Oct 21, 2019 @ 20:17:57.899	Successfully flipped image id: f1eqrievdiwxt0d7vknf
>	Oct 21, 2019 @ 20:17:57.822	Receiving image/png image to flip.

Expanded document

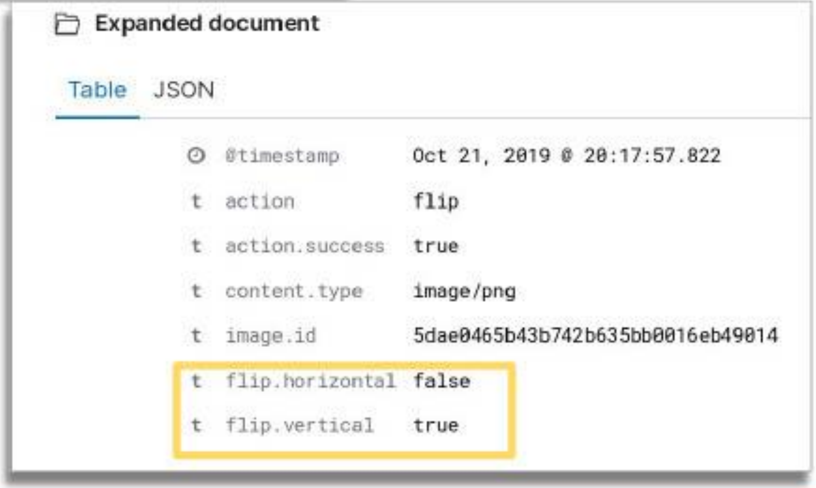
Table JSON

	@timestamp	Oct 21, 2019 @ 20:17:57.822
t	action	flip
t	action.success	true
t	content.type	image/png
t	image.id	5dae0465b43b742b635bb0016eb49014
t	flip.horizontal	false
t	flip.vertical	true

...AND EASY TO ADD MORE!



	Time	message
>	Oct 21, 2019 @ 20:17:57.899	Successfully flipped image id: f1eqrievdiwxt0d7vknf
>	Oct 21, 2019 @ 20:17:57.822	Receiving image/png image to flip.



Expanded document

Table JSON

⏲	@timestamp	Oct 21, 2019 @ 20:17:57.822
t	action	flip
t	action.success	true
t	content.type	image/png
t	image.id	5dae0465b43b742b635bb0016eb49014
t	flip.horizontal	false
t	flip.vertical	true

Better data structures supports more democratised debugging

Complex systems require a low friction way to add fields for added context and searchability and a way to combine technical context with business context

Pssst...
You heard of tracing?

That is just events with some extra IDs thrown in!

Characteristics of valuable outputs

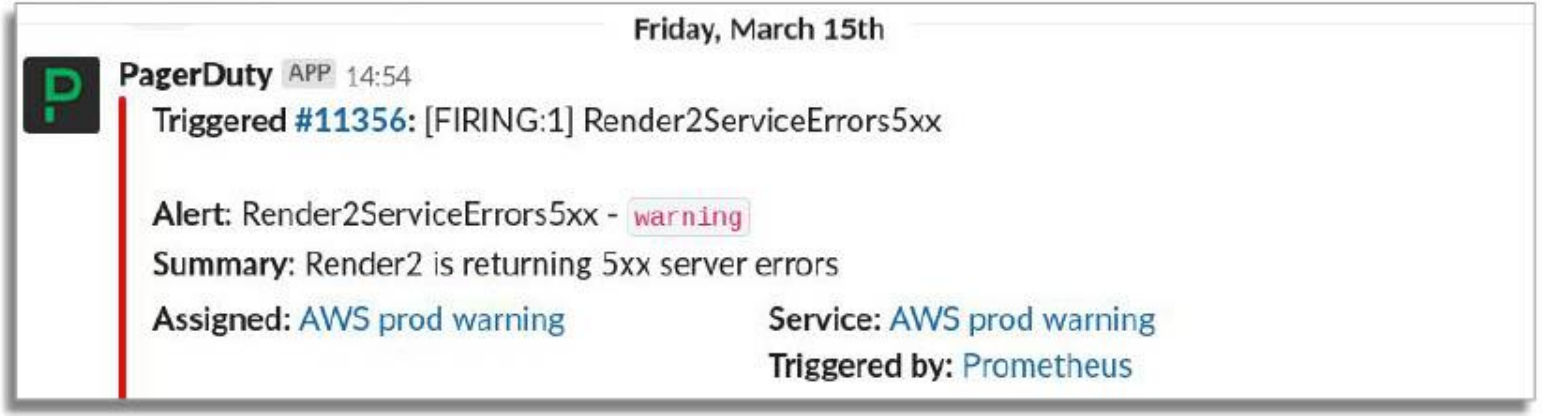
- → raw events
- → **no pre-aggregation**
- → structured data
- → **arbitrarily wide events**
- → schema-less-ness
- → high cardinality dimensions
- → oriented around request lifecycle
- → batched up context
- → **exploration over static dashboards**

@A_BANGSER

Debugging distributed systems is difficult

Especially when business impact is on the line. Let's talk incident response.

Hmmm, an automated alert



Friday, March 15th

PagerDuty APP 14:54

Triggered #11356: [FIRING:1] Render2ServiceErrors5xx

Alert: Render2ServiceErrors5xx - `warning`

Summary: Render2 is returning 5xx server errors

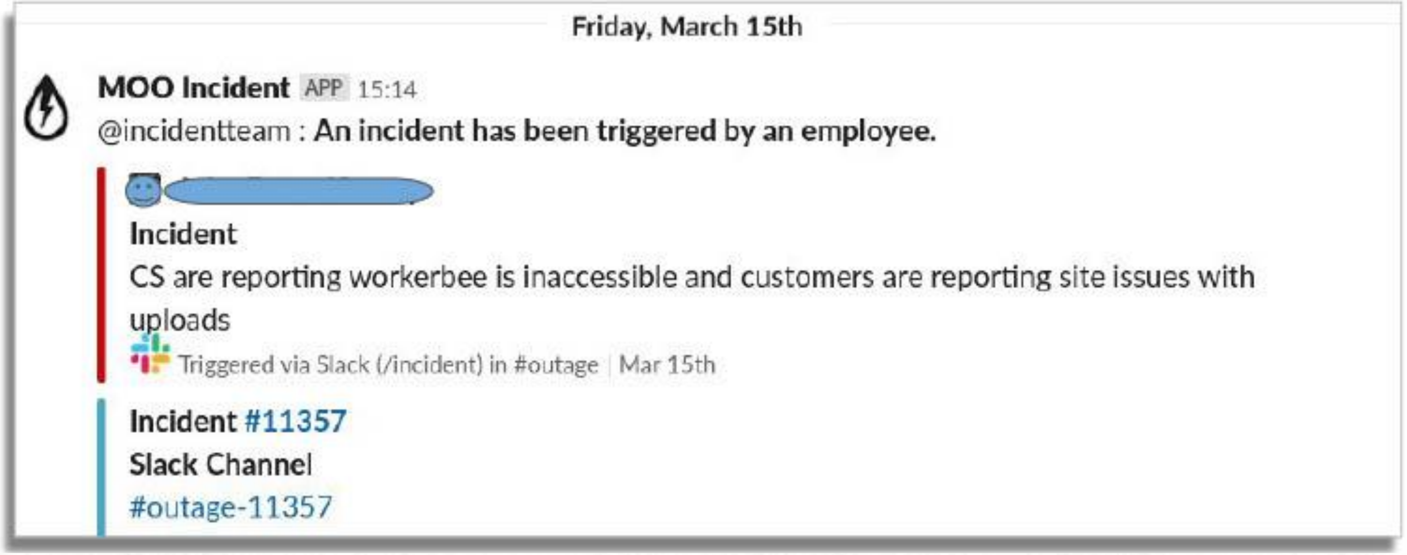
Assigned: AWS prod warning

Service: AWS prod warning

Triggered by: Prometheus

@A_BANGSI

Yup, definitely an issue!


Friday, March 15th

MOO Incident APP 15:14

@incidentteam : **An incident has been triggered by an employee.**

[redacted]

Incident

CS are reporting workerbee is inaccessible and customers are reporting site issues with uploads

Triggered via Slack (/incident) in #outage | Mar 15th

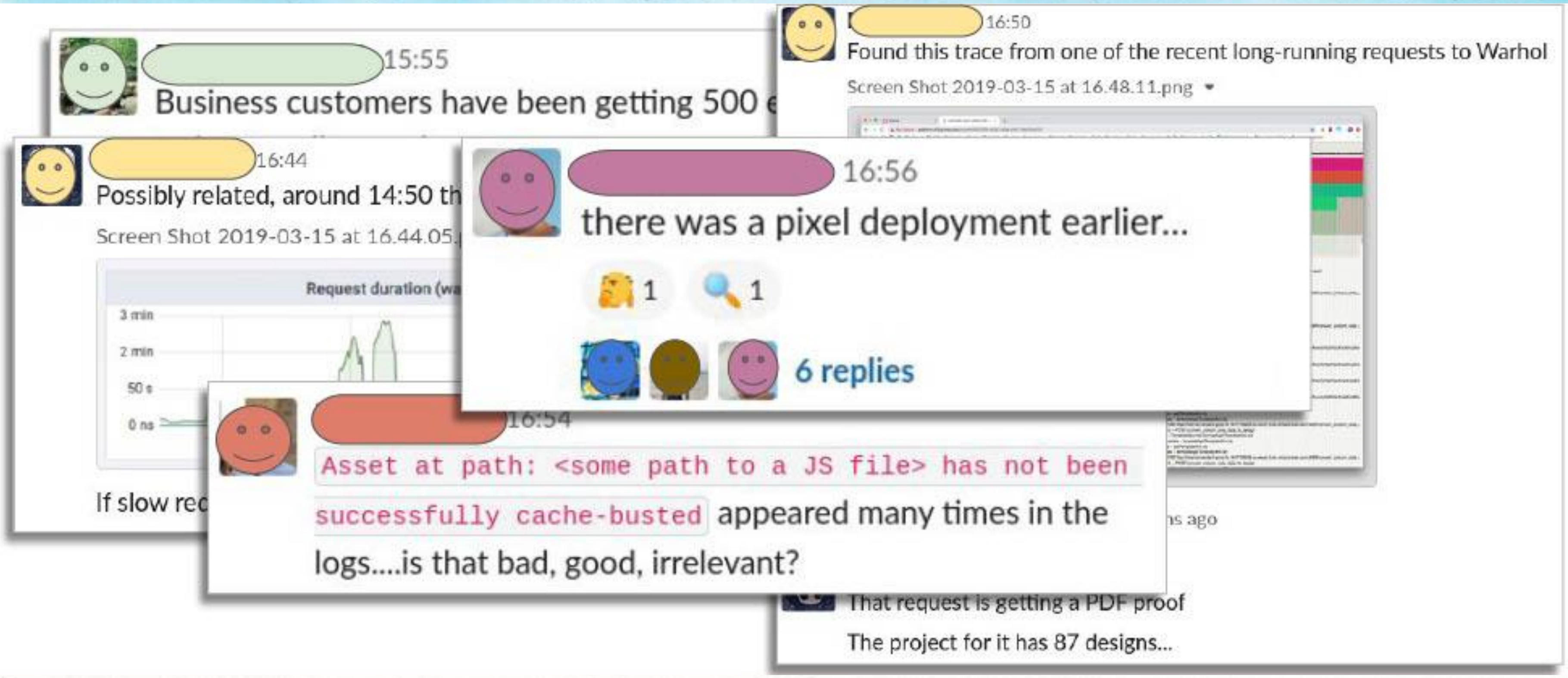
Incident **#11357**

Slack Channel

#outage-11357

@A_

All hands on deck, what is happening...and why?



[redacted] 15:55
Business customers have been getting 500 e

[redacted] 16:44
Possibly related, around 14:50 th
Screen Shot 2019-03-15 at 16.44.05.p

If slow rec

[redacted] 16:50
Found this trace from one of the recent long-running requests to Warhol
Screen Shot 2019-03-15 at 16.48.11.png ▾

ns ago

That request is getting a PDF proof
The project for it has 87 designs...

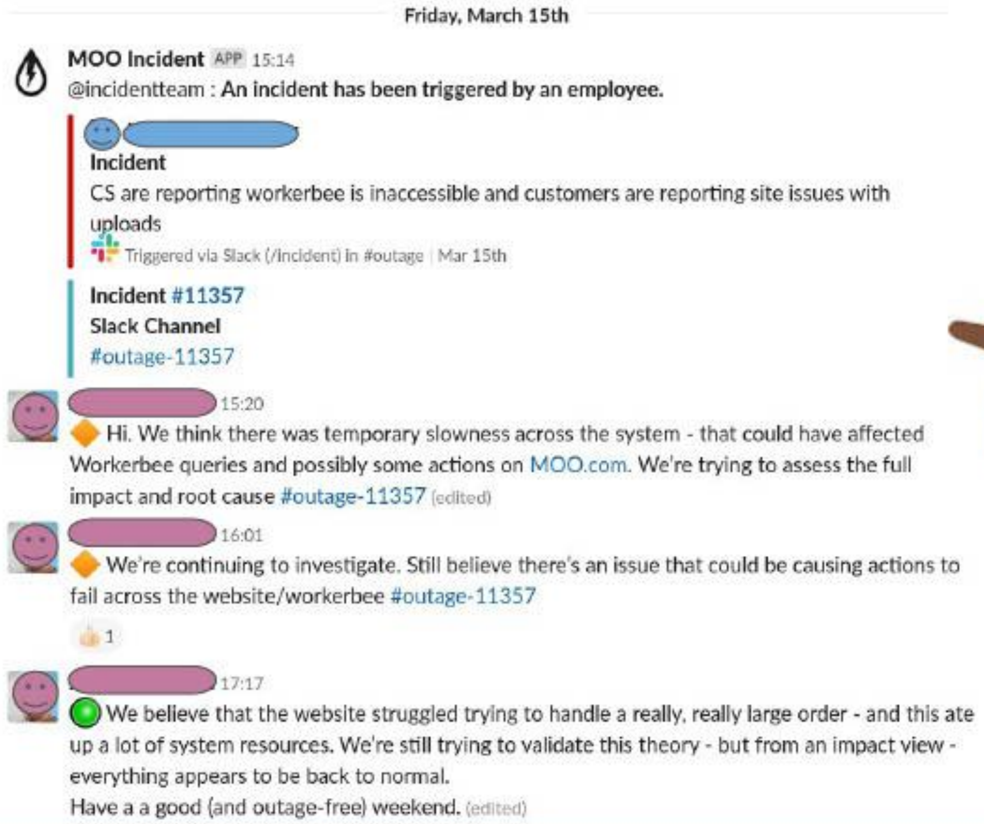
[redacted] 16:54
`Asset at path: <some path to a JS file> has not been successfully cache-busted` appeared many times in the logs....is that bad, good, irrelevant?

[redacted] 16:56
there was a pixel deployment earlier...

1 1

6 replies

2+ hrs later and still no idea!



Friday, March 15th

MOO Incident APP 15:14
@incidentteam : **An incident has been triggered by an employee.**

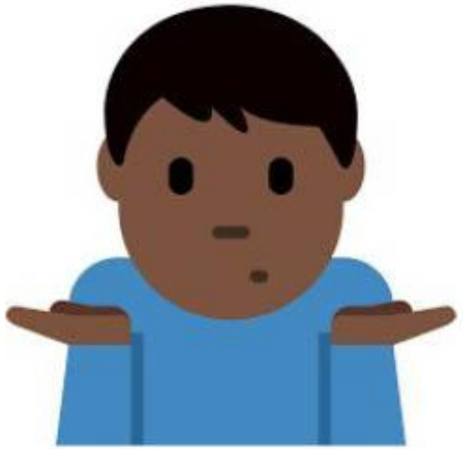
[redacted]
Incident
CS are reporting workerbee is inaccessible and customers are reporting site issues with uploads
Triggered via Slack (/incident) in #outage | Mar 15th

Incident #11357
Slack Channel
#outage-11357

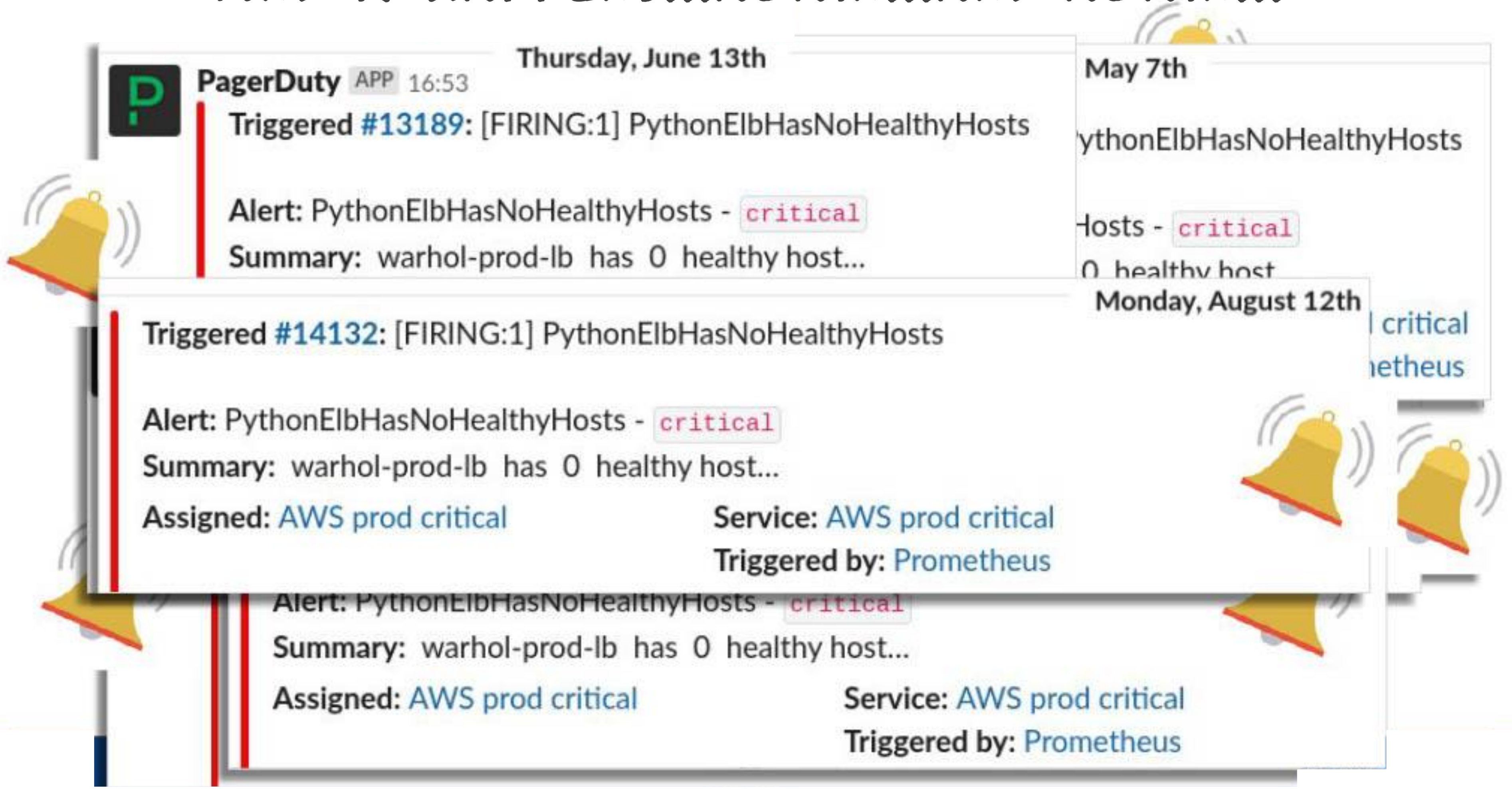
[redacted] 15:20
🔶 Hi. We think there was temporary slowness across the system - that could have affected Workerbee queries and possibly some actions on MOO.com. We're trying to assess the full impact and root cause #outage-11357 (edited)

[redacted] 16:01
🔶 We're continuing to investigate. Still believe there's an issue that could be causing actions to fail across the website/workerbee #outage-11357
👍 1

[redacted] 17:17
🟢 We believe that the website struggled trying to handle a really, really large order - and this ate up a lot of system resources. We're still trying to validate this theory - but from an impact view - everything appears to be back to normal.
Have a a good (and outage-free) weekend. (edited)



And it happens...again...and again...



Thursday, June 13th

PagerDuty APP 16:53

Triggered #13189: [FIRING:1] PythonElbHasNoHealthyHosts

Alert: PythonElbHasNoHealthyHosts - `critical`

Summary: warhol-prod-lb has 0 healthy host...

May 7th

ythonElbHasNoHealthyHosts

Hosts - `critical`

0 healthy host

critical

etheus

Monday, August 12th

Triggered #14132: [FIRING:1] PythonElbHasNoHealthyHosts

Alert: PythonElbHasNoHealthyHosts - `critical`

Summary: warhol-prod-lb has 0 healthy host...

Assigned: AWS prod critical

Service: AWS prod critical

Triggered by: Prometheus

Alert: PythonElbHasNoHealthyHosts - `critical`

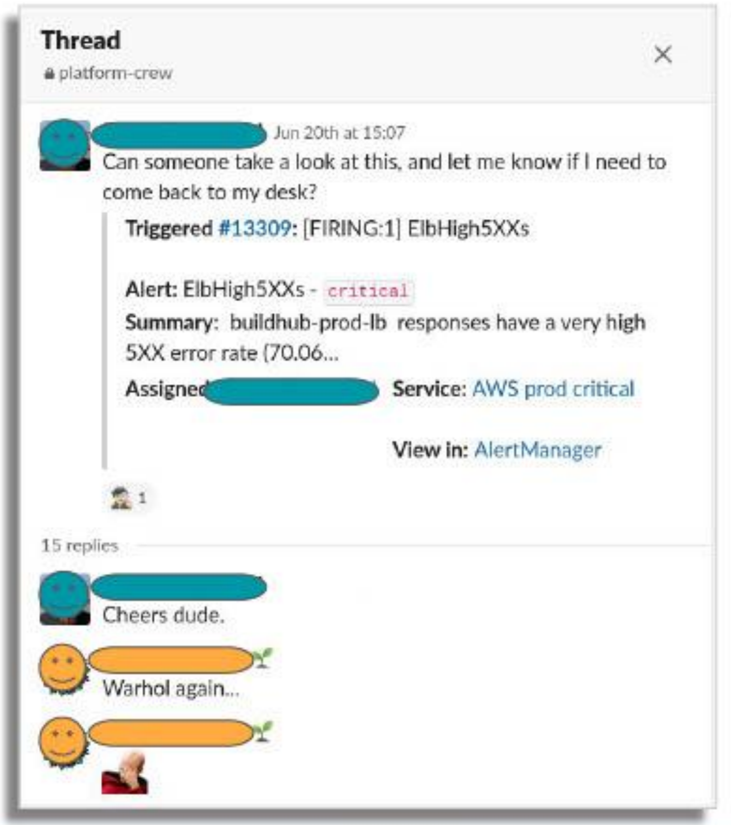
Summary: warhol-prod-lb has 0 healthy host...

Assigned: AWS prod critical

Service: AWS prod critical

Triggered by: Prometheus

On call engineers are not amused



Thread
platform-crew

[redacted] Jun 20th at 15:07
Can someone take a look at this, and let me know if I need to come back to my desk?

Triggered #13309: [FIRING:1] ElbHigh5XXs

Alert: ElbHigh5XXs - `critical`
Summary: buildhub-prod-lb responses have a very high 5XX error rate (70.06...
Assigned [redacted] **Service:** AWS prod critical

View in: AlertManager

1

15 replies

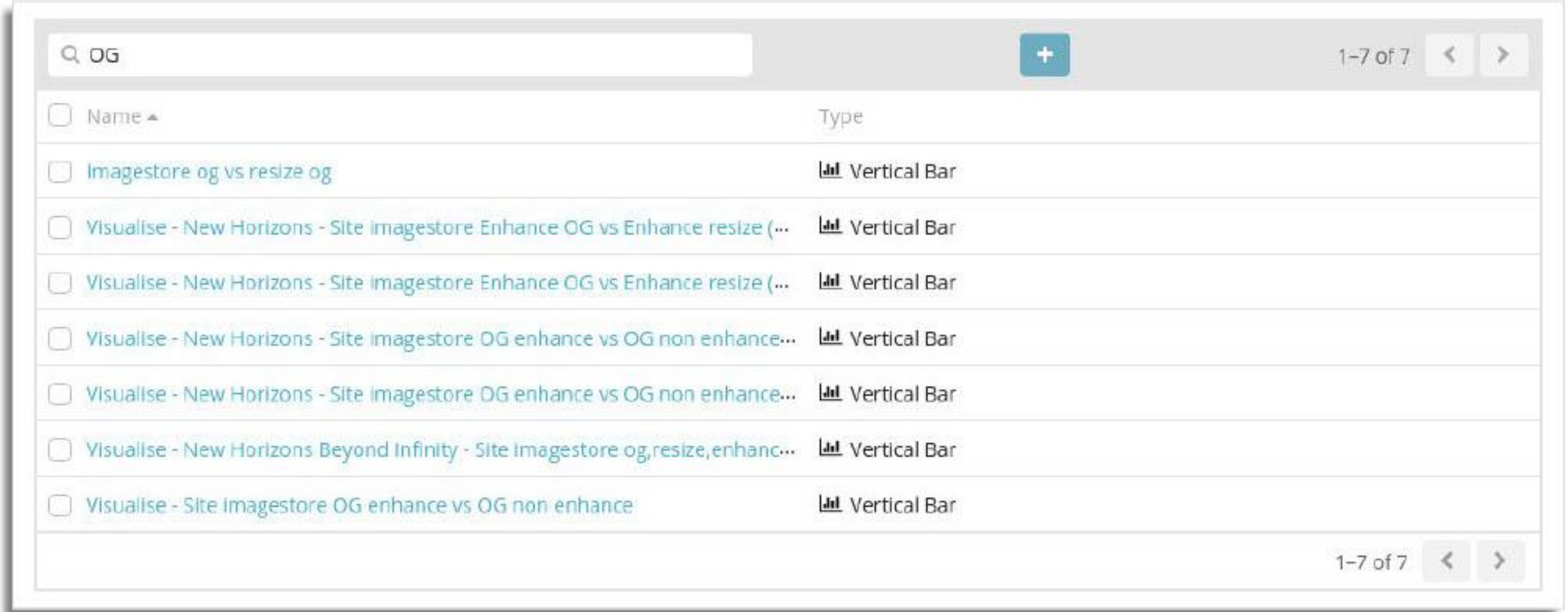
[redacted]
Cheers dude.

[redacted]
Warhol again...

[redacted]

@A_BANGSER

But service owners were working hard!



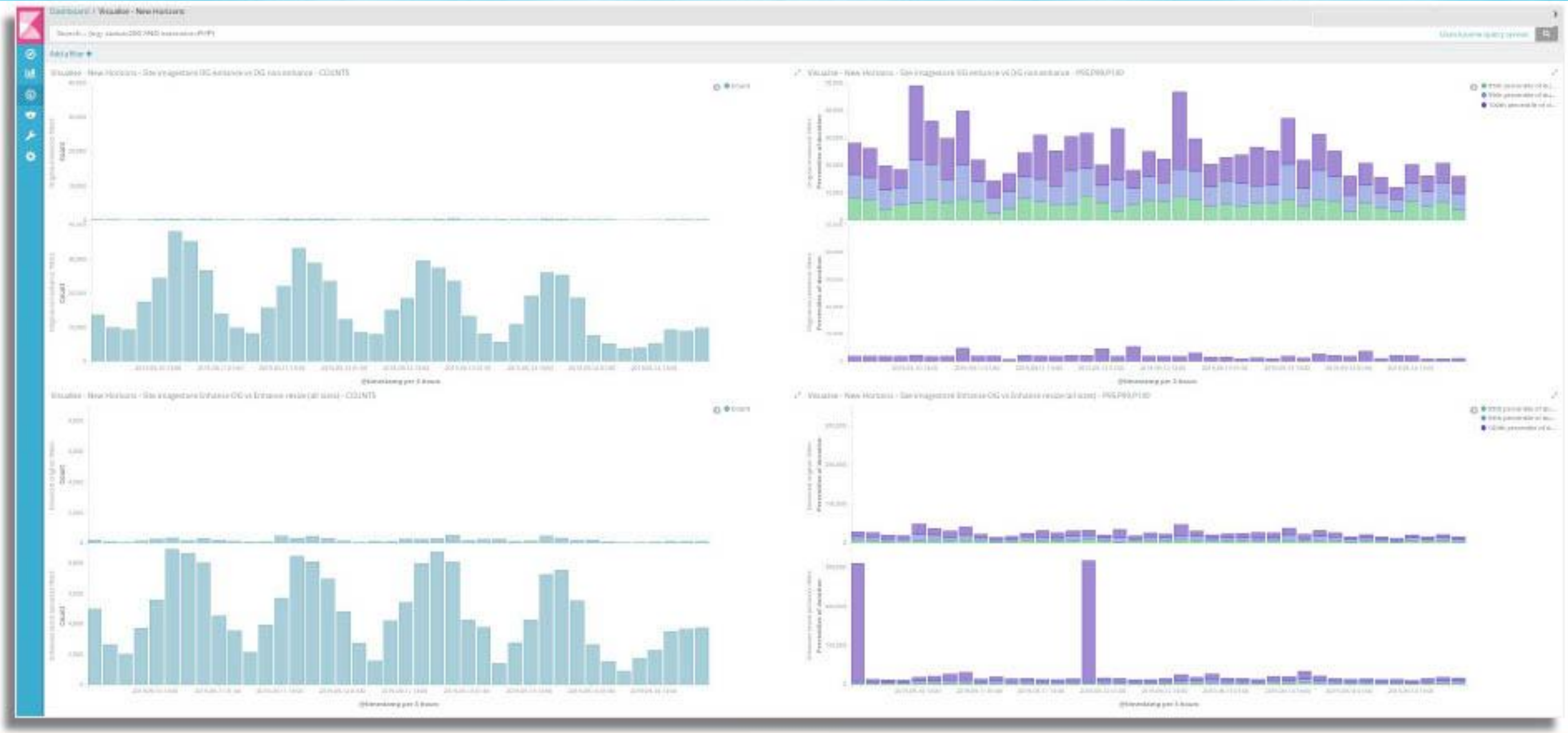
OG

1–7 of 7

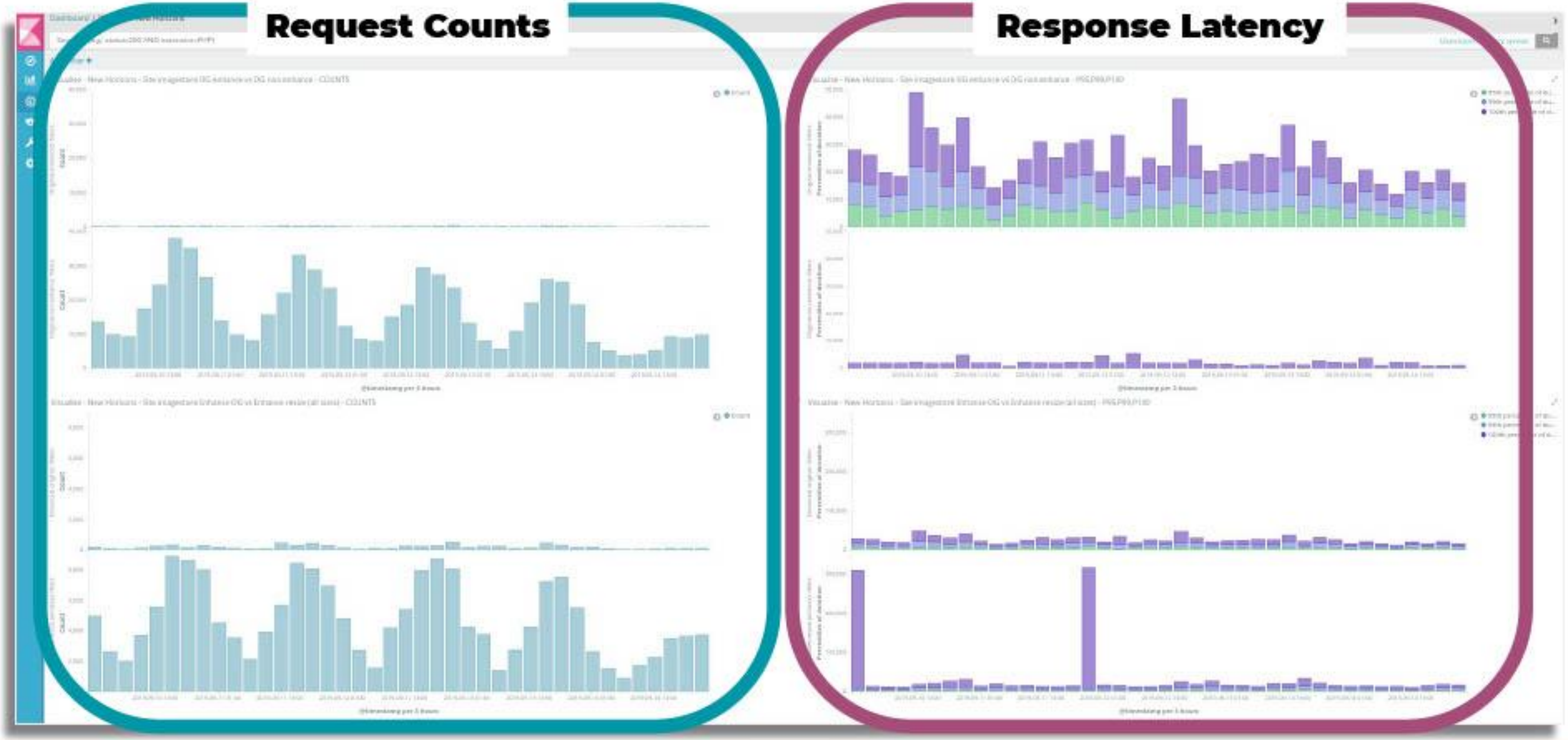
Name	Type
Imagestore og vs resize og	Vertical Bar
Visualise - New Horizons - Site imagestore Enhance OG vs Enhance resize (...	Vertical Bar
Visualise - New Horizons - Site imagestore Enhance OG vs Enhance resize (...	Vertical Bar
Visualise - New Horizons - Site imagestore OG enhance vs OG non enhance...	Vertical Bar
Visualise - New Horizons - Site imagestore OG enhance vs OG non enhance...	Vertical Bar
Visualise - New Horizons Beyond Infinity - Site imagestore og,resize,enhanc...	Vertical Bar
Visualise - Site imagestore OG enhance vs OG non enhance	Vertical Bar

1–7 of 7

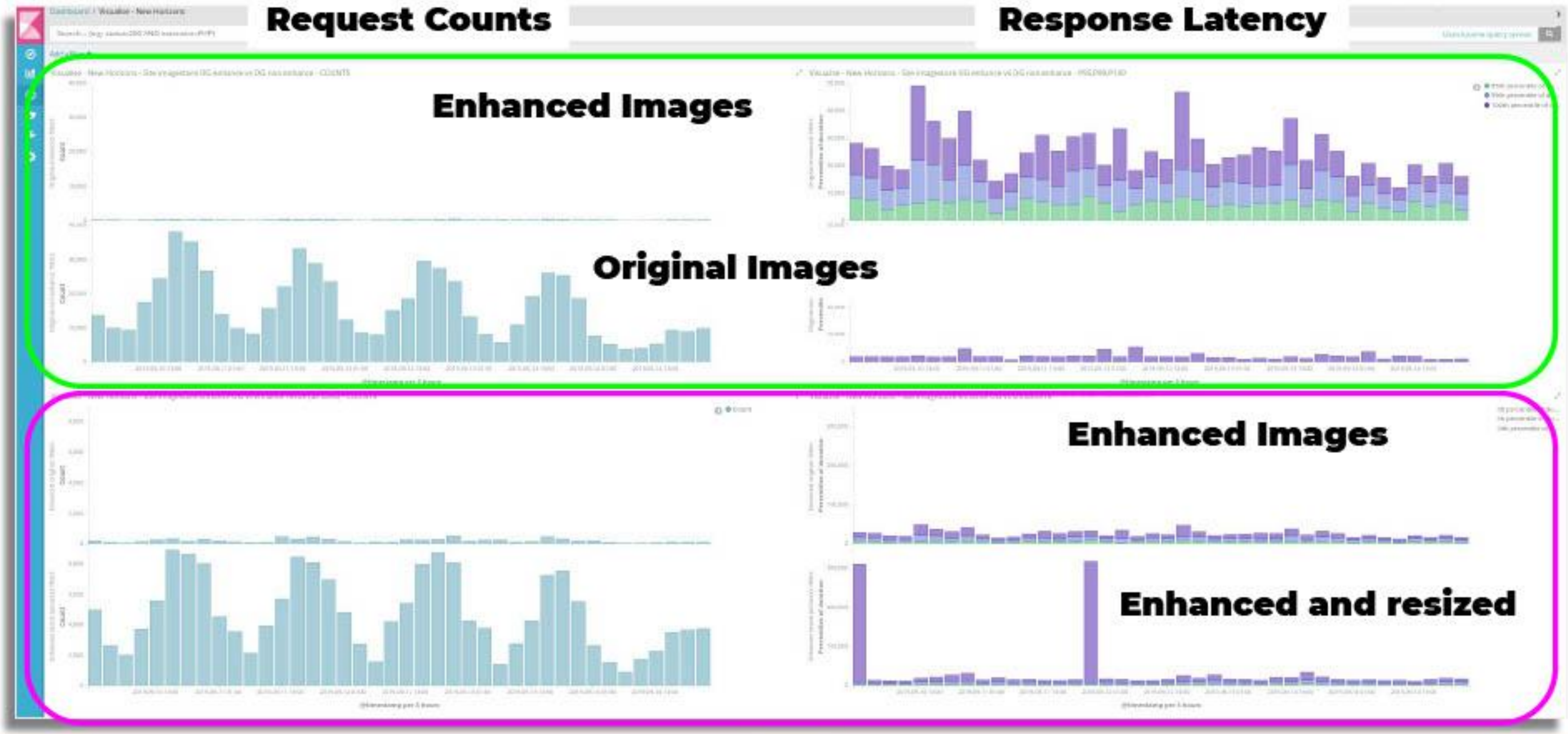
These were some awesome dashboards



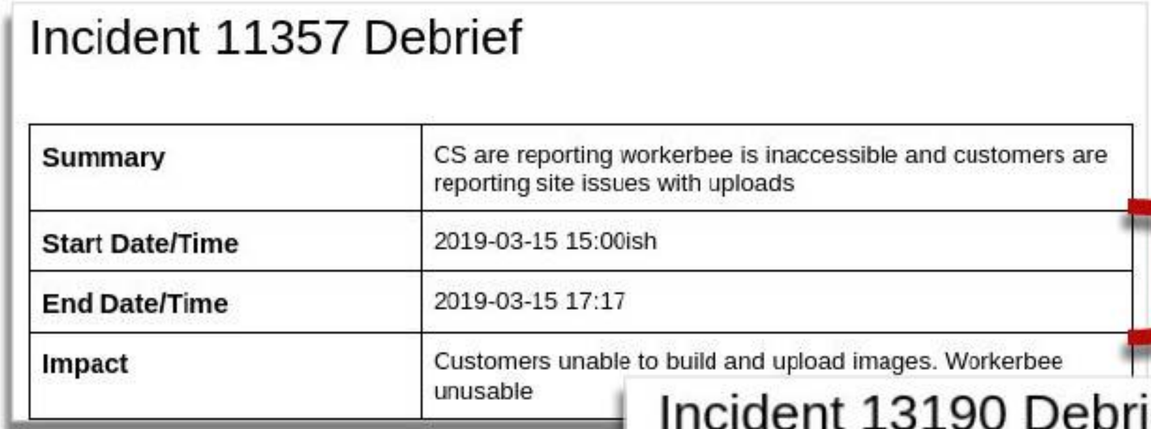
The dashboards showed a lot of detail



And broke down different parameters



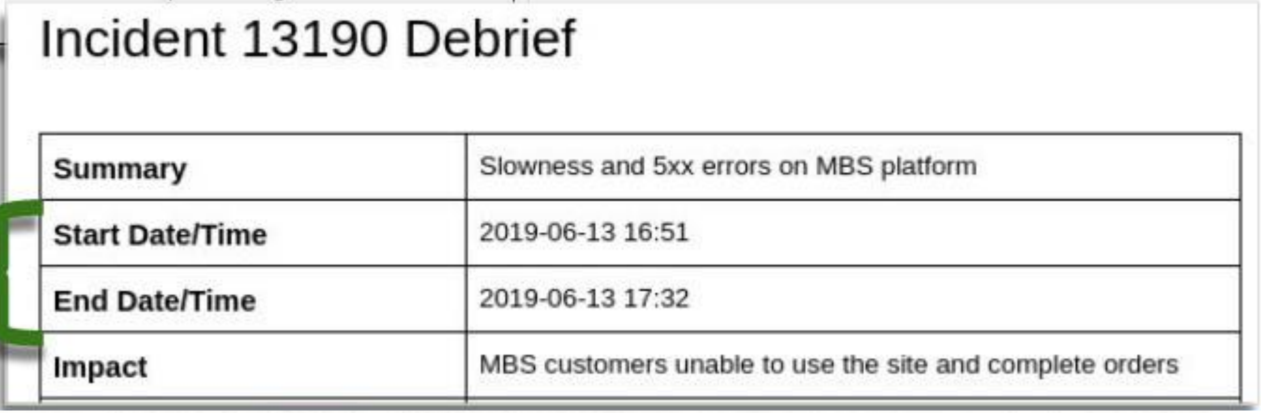
They even helped reduce incident impact



Incident 11357 Debrief

Summary	CS are reporting workerbee is inaccessible and customers are reporting site issues with uploads
Start Date/Time	2019-03-15 15:00ish
End Date/Time	2019-03-15 17:17
Impact	Customers unable to build and upload images. Workerbee unusable

~3 hours

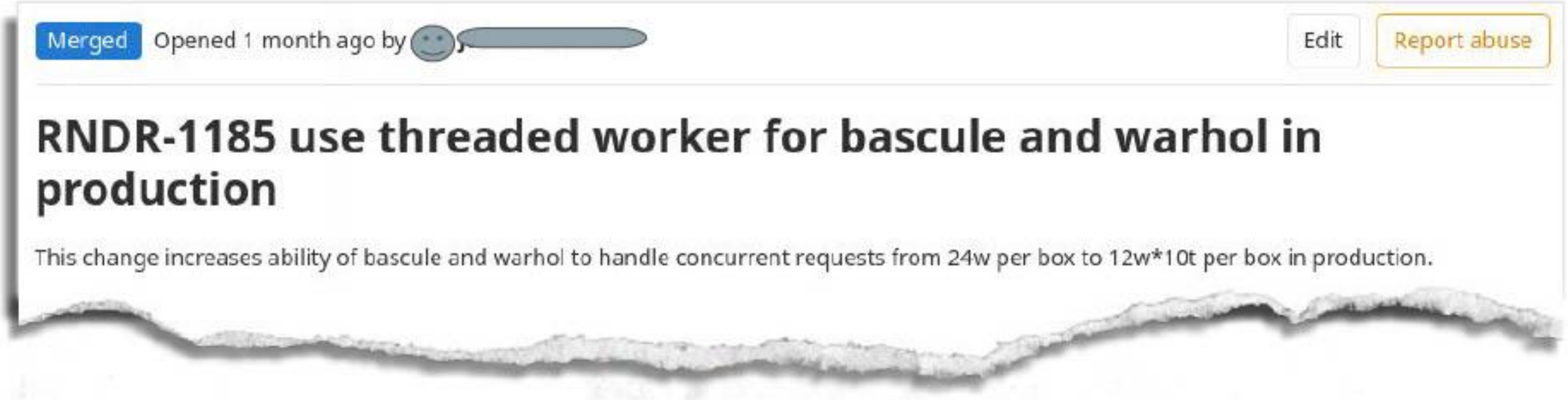


Incident 13190 Debrief

Summary	Slowness and 5xx errors on MBS platform
Start Date/Time	2019-06-13 16:51
End Date/Time	2019-06-13 17:32
Impact	MBS customers unable to use the site and complete orders

40 min

But human pattern matchers solved it


Merged Opened 1 month ago by [redacted]

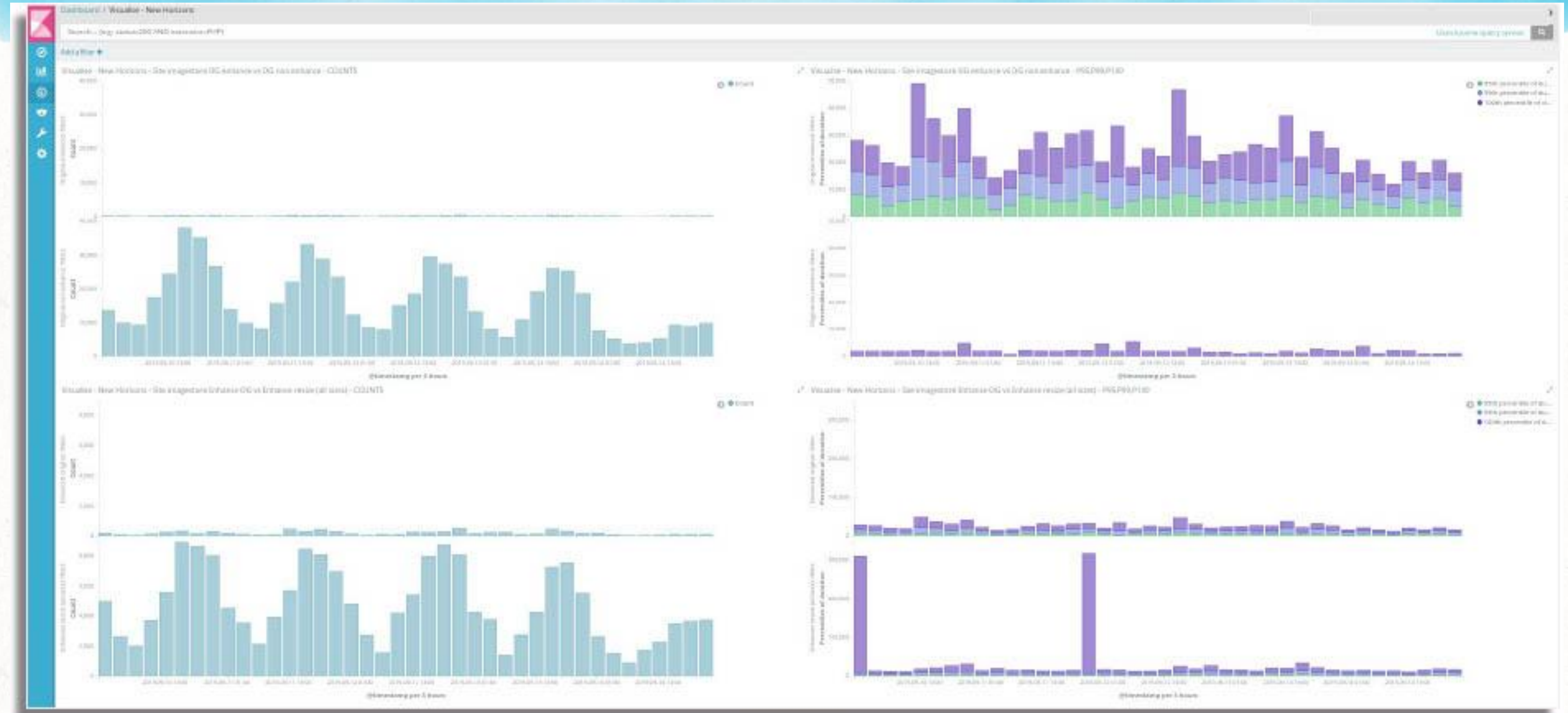
Edit Report abuse

RNDR-1185 use threaded worker for bascule and warhol in production

This change increases ability of bascule and warhol to handle concurrent requests from 24w per box to 12w*10t per box in production.

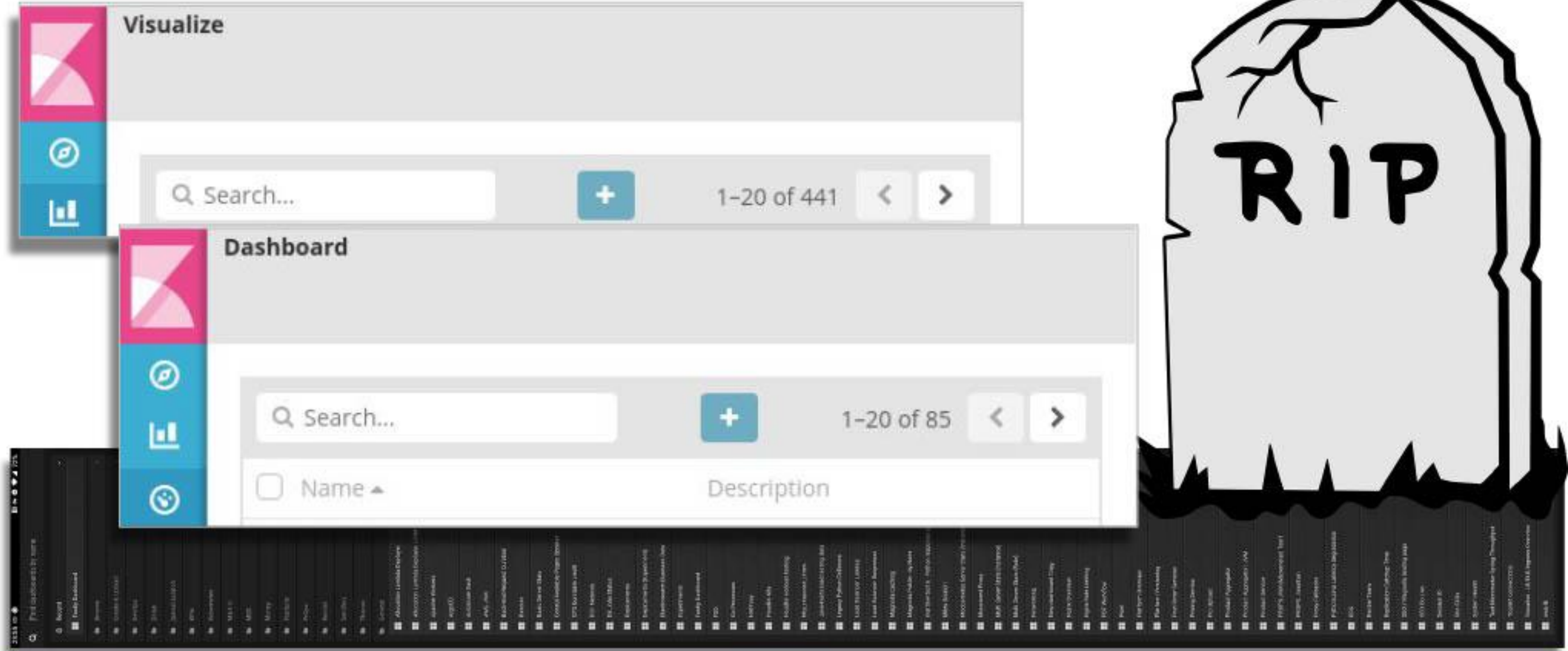
@A_BANGSER

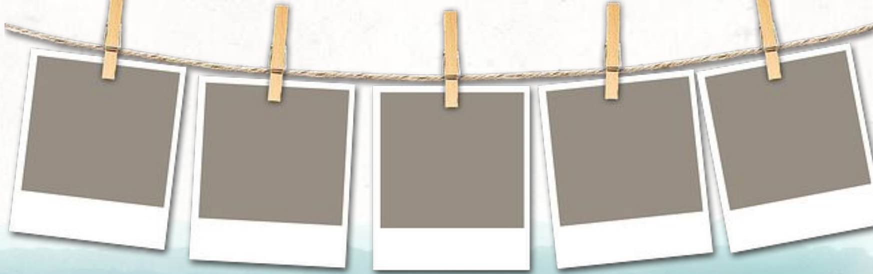
So what happened to the new dashboards?



@a_bangser

They were sent to the farm... with lots of friends

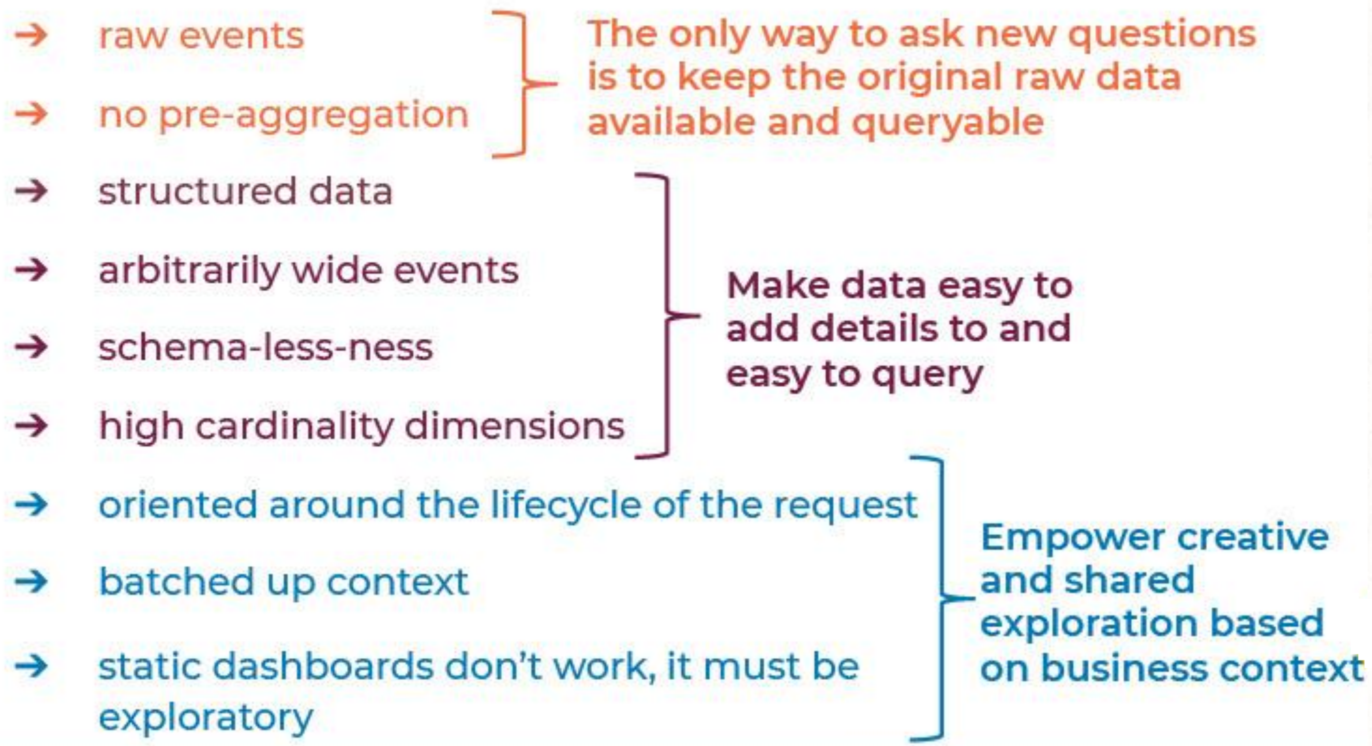




Why prioritise exploration?

Dashboards are the scar tissue of past incidents. Focus on learning about new behaviours and issues.

These characteristics drive outcomes



- → raw events
- → no pre-aggregation

The only way to ask new questions is to keep the original raw data available and queryable

- → structured data
- → arbitrarily wide events
- → schema-less-ness
- → high cardinality dimensions

Make data easy to add details to and easy to query

- → oriented around the lifecycle of the request
- → batched up context
- → static dashboards don't work, it must be exploratory

Empower creative and shared exploration based on business context

@A_BANGSER

So HOW can I get started?

It depends on your context

- Remove pressure from your metrics by understanding their use case better
- Introduce more context to your log/event data?
- Shift from logs to events where it makes sense?
- Enable easier exploration instead of fancier dashboards?

@A_BANGSER

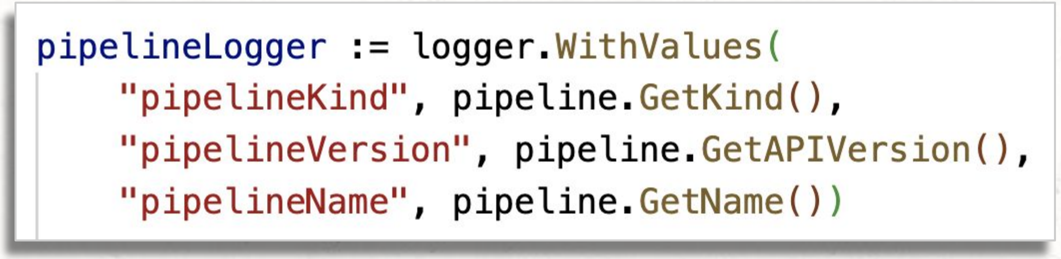
Focus on outcomes.
Our desired outcomes are:

- → Iterating quickly on feature development
- → Debugging user issues
- → Understanding usage patterns without putting user privacy at risk

@A_BANGSER

Iterate quickly on feature development

➔ Provide rich contextual information in our logs

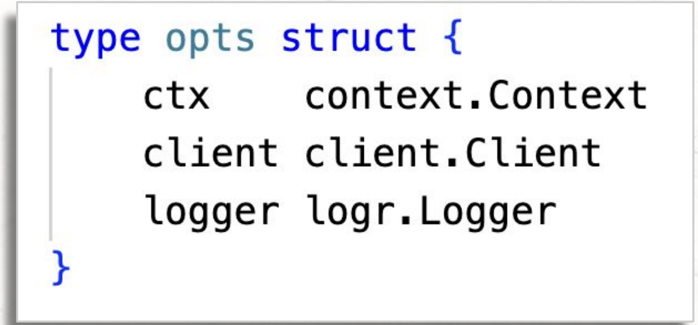


```
pipelineLogger := logger.WithValues(
    "pipelineKind", pipeline.GetKind(),
    "pipelineVersion", pipeline.GetAPIVersion(),
    "pipelineName", pipeline.GetName())
```

@A_BANGSER

Iterate quickly on feature development

- Provide rich contextual information in our logs
- Default access to a logger

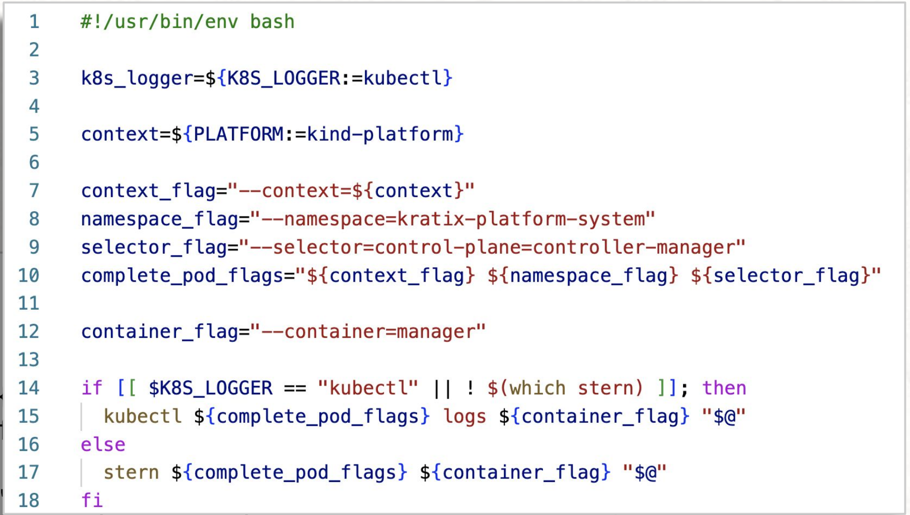


```go
type opts struct {
	ctx     context.Context
	client client.Client
	logger logr.Logger
}
```

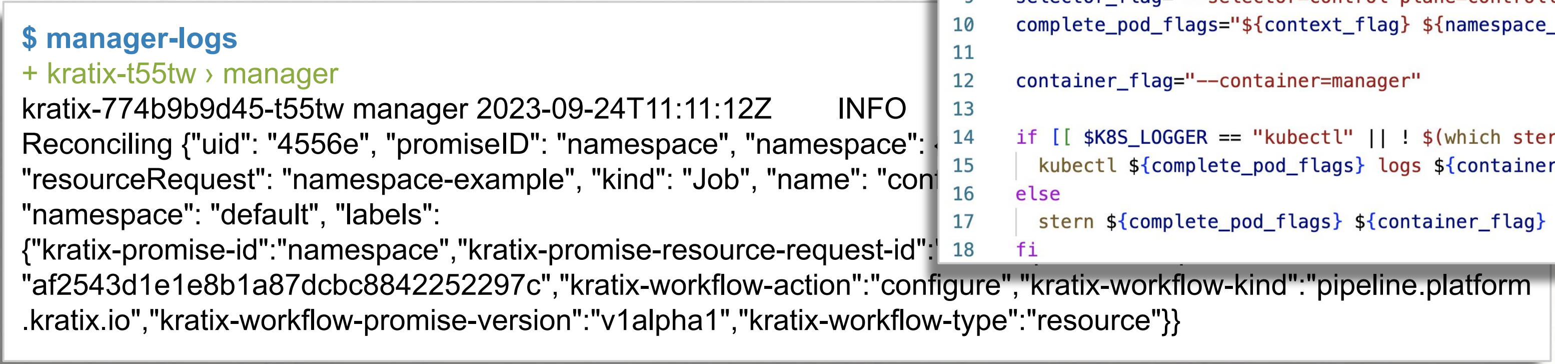
@A_BANGSER

Debug user issues

→ Create a way to see / share logs easily



```bash
#!/usr/bin/env bash

k8s_logger=${K8S_LOGGER:=kubectl}

context=${PLATFORM:=kind-platform}

context_flag="--context=${context}"
namespace_flag="--namespace=kratix-platform-system"
selector_flag="--selector=control-plane=controller-manager"
complete_pod_flags="${context_flag} ${namespace_flag} ${selector_flag}"

container_flag="--container=manager"

if [[ $K8S_LOGGER == "kubectl" || ! $(which stern) ]]; then
  kubectl ${complete_pod_flags} logs ${container_flag} "$@"
else
  stern ${complete_pod_flags} ${container_flag} "$@"
fi
```

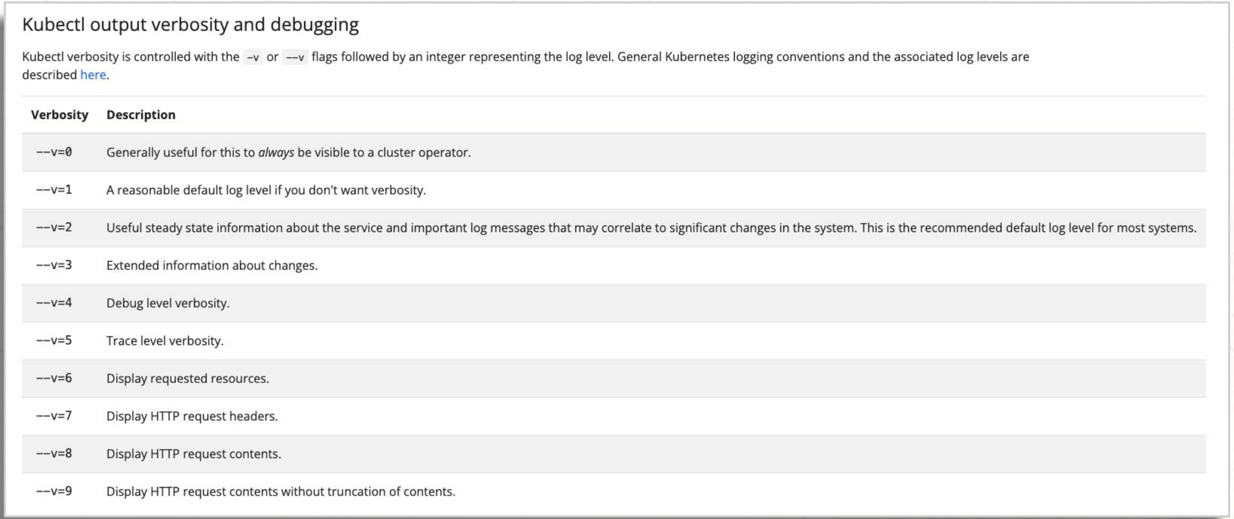


```
$ manager-logs
+ kratix-t55tw › manager
kratix-774b9b9d45-t55tw manager 2023-09-24T11:11:12Z        INFO
Reconciling {"uid": "4556e", "promiseID": "namespace", "namespace":
"resourceRequest": "namespace-example", "kind": "Job", "name": "con
"namespace": "default", "labels":
{"kratix-promise-id":"namespace","kratix-promise-resource-request-id":
"af2543d1e1e8b1a87dcbc8842252297c","kratix-workflow-action":"configure","kratix-workflow-kind":"pipeline.platform
.kratix.io","kratix-workflow-promise-version":"v1alpha1","kratix-workflow-type":"resource"}}
```

@A_BANGSER

Debug user issues

- Create a way to see / share logs easily
- Be intentional about “noise”



Kubectl output verbosity and debugging

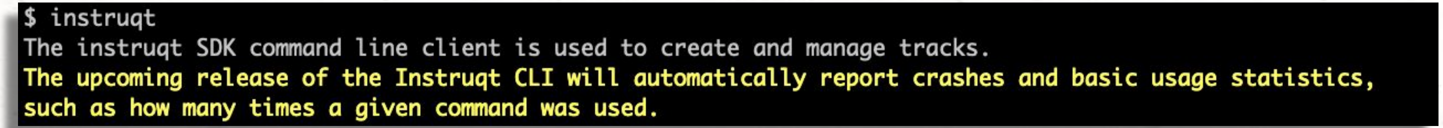
Kubectl verbosity is controlled with the `-v` or `--v` flags followed by an integer representing the log level. General Kubernetes logging conventions and the associated log levels are described here.

Verbosity	Description
`--v=0`	Generally useful for this to *always* be visible to a cluster operator.
`--v=1`	A reasonable default log level if you don't want verbosity.
`--v=2`	Useful steady state information about the service and important log messages that may correlate to significant changes in the system. This is the recommended default log level for most systems.
`--v=3`	Extended information about changes.
`--v=4`	Debug level verbosity.
`--v=5`	Trace level verbosity.
`--v=6`	Display requested resources.
`--v=7`	Display HTTP request headers.
`--v=8`	Display HTTP request contents.
`--v=9`	Display HTTP request contents without truncation of contents.

@A_BANGSER

Understand usage patterns without putting user privacy at risk

- ➔ Leverage the low cardinality of metrics

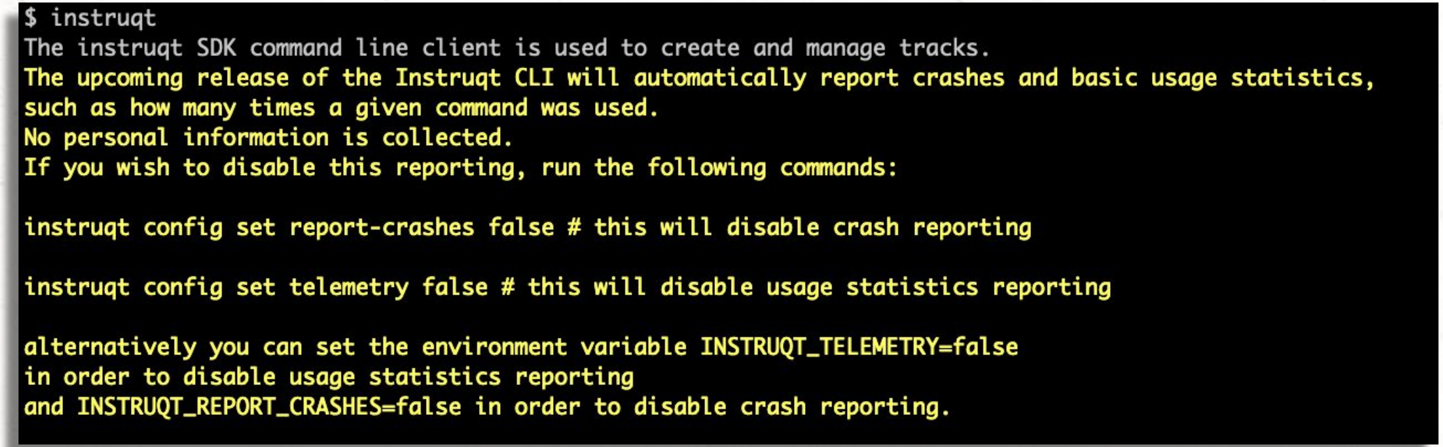


```
$ instruqt
The instruqt SDK command line client is used to create and manage tracks.
The upcoming release of the Instruqt CLI will automatically report crashes and basic usage statistics,
such as how many times a given command was used.
```

@A_BANGSER

Understand usage patterns without putting user privacy at risk

- Leverage the low cardinality of metrics
- Allow users to turn this off



```
$ instruqt
The instruqt SDK command line client is used to create and manage tracks.
The upcoming release of the Instruqt CLI will automatically report crashes and basic usage statistics,
such as how many times a given command was used.
No personal information is collected.
If you wish to disable this reporting, run the following commands:

instruqt config set report-crashes false # this will disable crash reporting

instruqt config set telemetry false # this will disable usage statistics reporting

alternatively you can set the environment variable INSTRUQT_TELEMETRY=false
in order to disable usage statistics reporting
and INSTRUQT_REPORT_CRASHES=false in order to disable crash reporting.
```

@A_BANGSER

At the end of the day...

- Observability is a tool and each technique has its use cases and challenges.
- Data collection is not the goal and is not magic.
- Focus on outcomes and use observability to achieve them.

HUSTEF

HUNGARIAN SOFTWARE TESTING FORUM

eventee

Please rate this session

(for a chance at one of these)

Thank you!

Abby Bangser (She/her)
abby@paintedwavelimited.com
@a_bangser
@abangser.bsky.social
hachyderm.io/@abangser