

# Observable JS Apps

@eanakashima



# about://

Emily Nakashima  
product engineering manager  
@ honeycomb.io

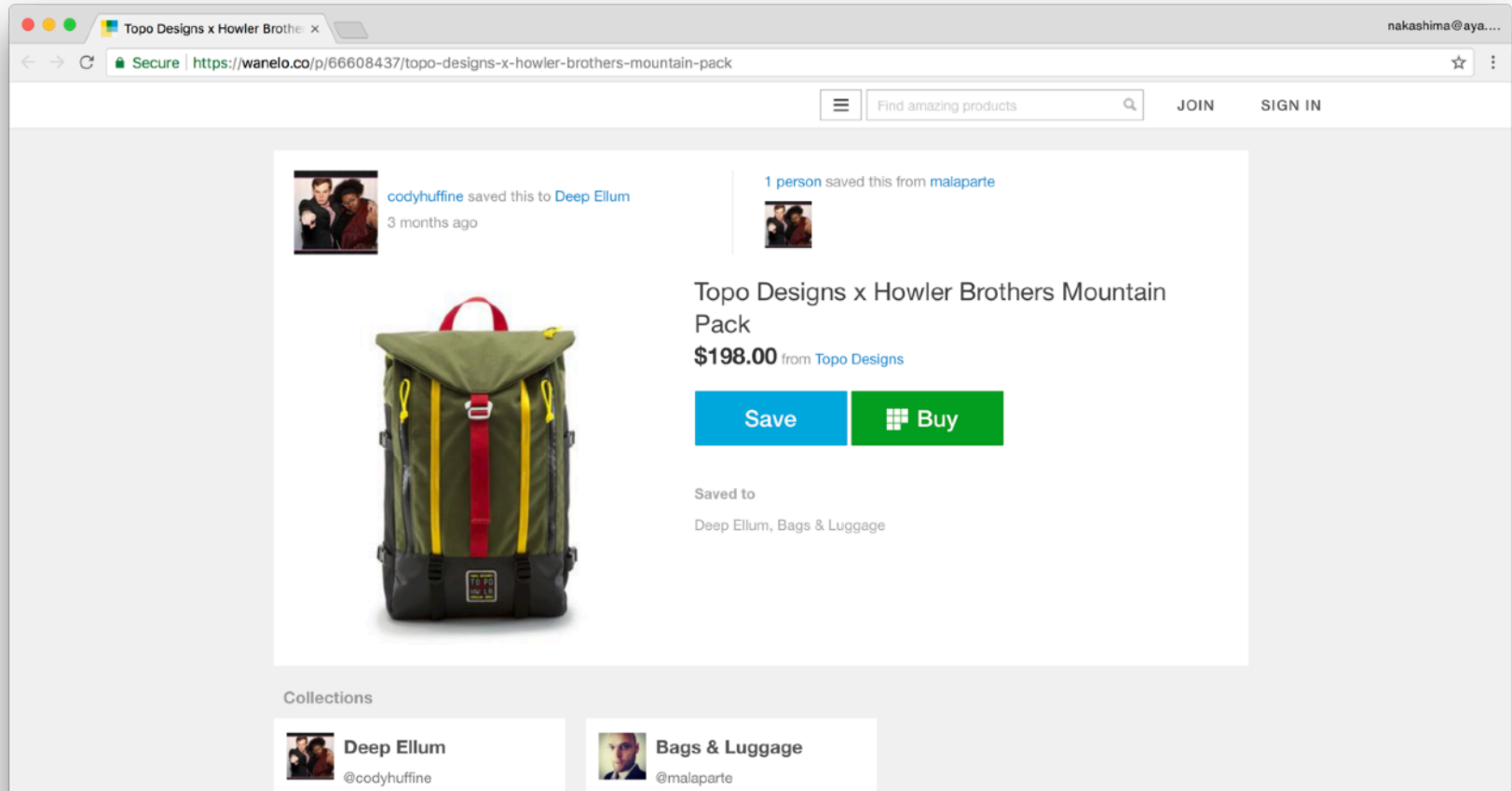




**GitHub**



# The one where we DDOS'd ourselves



The screenshot shows a web browser window with the URL <https://wanelo.co/p/66608437/topo-designs-x-howler-brothers-mountain-pack>. The page features a green and yellow backpack. The product title is "Topo Designs x Howler Brothers Mountain Pack" with a price of "\$198.00 from Topo Designs". There are "Save" and "Buy" buttons. The page also shows social proof: "codyhuffine saved this to Deep Ellum 3 months ago" and "1 person saved this from malaparte". At the bottom, there are collection cards for "Deep Ellum" and "Bags & Luggage".

Topo Designs x Howler Brothers Mountain Pack


Secure | <https://wanelo.co/p/66608437/topo-designs-x-howler-brothers-mountain-pack>

Find amazing products

JOIN SIGN IN

codyhuffine saved this to [Deep Ellum](#)  
3 months ago

1 person saved this from [malaparte](#)




Topo Designs x Howler Brothers Mountain Pack


\$198.00 from [Topo Designs](#)

Save Buy

Saved to  
Deep Ellum, Bags & Luggage

Collections

 **Deep Ellum**  
@codyhuffine

 **Bags & Luggage**  
@malaparte



# Epilogue

The screenshot shows a web browser window with the URL 'Topo Designs' and a user email 'nakashima@aya...'. The page features a navigation bar with a search bar containing the text 'Find amazing products', and links for 'JOIN' and 'SIGN IN'. The main content is a grid of ten product cards, each with an image, a title, and a price.

Product Name	Price
Heavyweight Pocket Tee	\$45.00
Heavyweight Long Sleeve Pocket Tee	\$59.00
Mountain Shirt - Plaid Flannel	\$129.00
Women's Coverall	\$189.00
Topo Designs x Howler Brothers Mountain Pack	\$198.00
Work Pants - Natural Canvas	\$149.00
Breaker Shirt Jacket	\$149.00
3-Day Briefcase	\$189.00
Commuter Briefcase	\$189.00
Climb Pants	\$129.00



```
// if the page is long enough to scroll
if (document.body.clientHeight > window.innerHeight) {

    // add a scroll event listener
    document.addEventListener('scroll', function(e) {

        // if within 100px of the bottom of the page
        if (window.innerHeight + window.scrollY + 100 >
            document.body.clientHeight)
            fetchNextPage();

    });

// else fetch another page of results immediately
} else {
    fetchNextPage();
}
```



# We had *lots* of production data

1. **Product Analytics**
2. **Metrics**
3. **Client-side metrics**
4. **Error monitoring**
5. **Logs**



# What data (or graphs) would have helped?

- Total traffic count broken down into buckets by screen size.
- Average number of API requests triggered by a particular page type.





# Frontend complexity is only increasing



**Yehuda Katz** ✓

@wycats

Follow



Front end software development is:

- real-time (instant load, 60fps)
- distributed, incremental (synchronize remote data as needed)
- asynchronous
- reactive (react to user actions in realtime)

Front end is the hardest kind of dev I do. The folks who do it every day are heroes.

7:53 AM - 14 Nov 2017

1,774 Retweets 4,726 Likes



# Observability

Instrument your code so that you can:

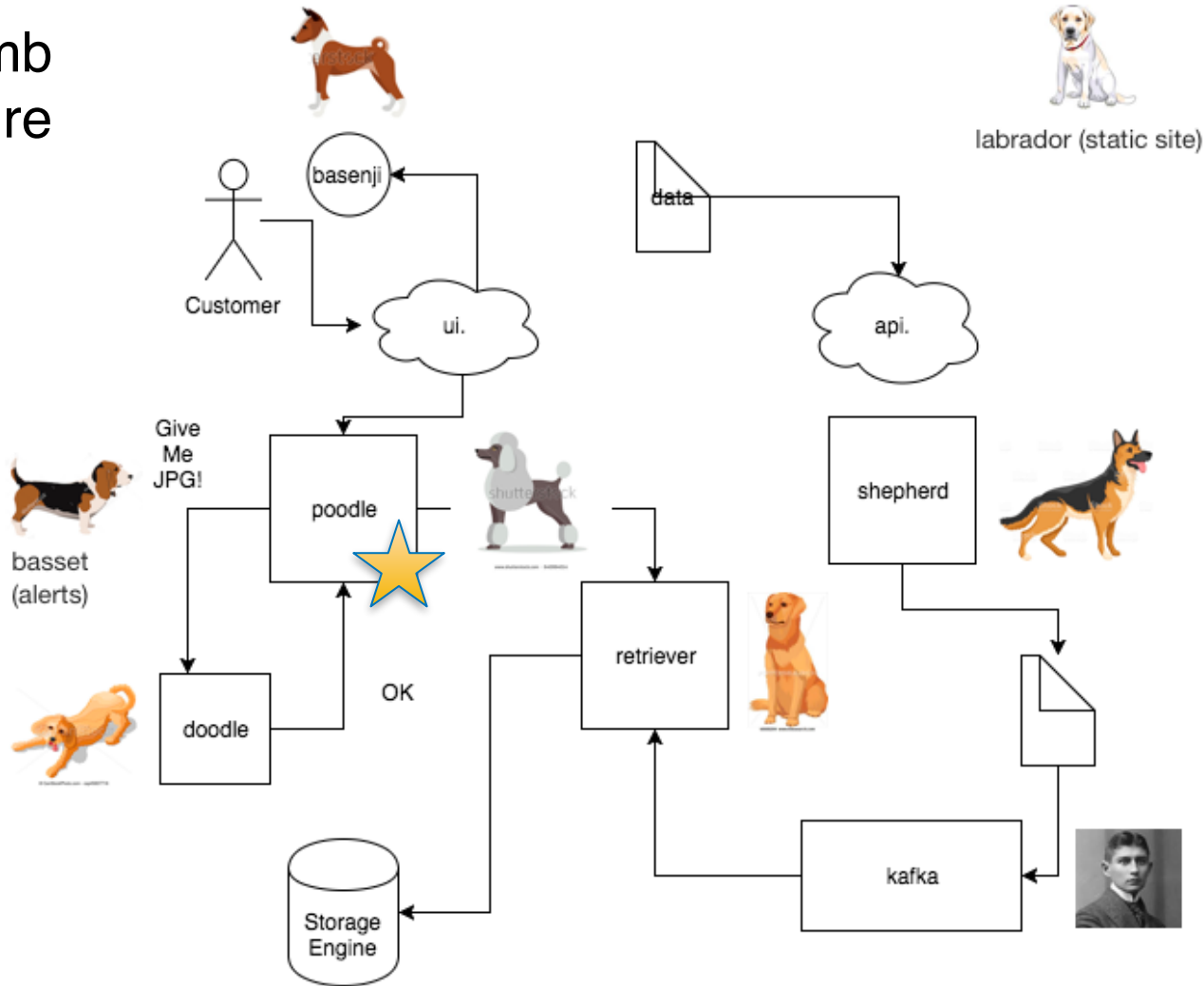
- ask any question, whether you anticipated it or not
- deeply understand the state of your system by observing its outputs



# Backend Instrumentation



# Honeycomb Architecture



# Types of production data

## 1. **Metrics**

## 2. **Events:** sent to Honeycomb

- Plus a few other tools that are “event-y”
  1. **Logs** (structured, no log aggregator)
  2. **Traces** (sometimes)
  3. **Error monitoring**

## 3. **Product Analytics**



# What's in an event?

```
{  
  "GojiPattern": "\/user event\/:event type",  
  "Header.Content-Type": "[\"application\/json\"]",  
  "Header.Cookie": "[\"_ga=GA1.2.2033006133.1516389900;\",  
  "Header.User-Agent": "[\"Mozilla\/5.0 (Macintosh; Intel Mac OS X 10_13_1)...\"]",  
  "Host": "127.0.0.1:8080",  
  "IsXHR": true,  
  "Method": "POST",  
  "RequestURI": "\/user_event\/page-unload",  
  "ResponseContentLength": 443,  
  "ResponseHttpStatus": 200,  
  "ResponseTime ms": 123,  
  "Timestamp": "2018-03-02T06:14:57.206349701Z",  
  "UserEmail": "nathan@honeycomb.io",  
  "UserID": 18,  
  "availability_zone": "us-east-1b",  
  "build_id": "6552",  
  "env": "dogfood",  
  "infra_type": "aws_instance",  
  "instance_type": "t2.micro",  
  "memory_inuse": 15450056,  
  "num_goroutines": 56,  
  "request_id": "poodle-a38f5e39\/5fIUGkX5D1-001814",  
  "server_hostname": "poodle-a38f5e39",  
  "type": "request"  
},
```



# High cardinality

Fields that may have many unique values

Common examples:

- email address
- username / user id / team id
- server hostname
- IP address
- user agent string
- build id
- request url
- feature flags / flag combinations



# What's in an event?

```
{
  "GojiPattern": "\/user_event\/:event_type",
  "Header.Content-Type": "[\"application\/json\"]",
  "Header.Cookie": "[\"_ga=GA1.2.2033006133.1516389900;\"",
  "Header.User-Agent": "[\"Mozilla\/5.0 (Macintosh; Intel Mac OS X 10_13_1)...\"]",
  "Host": "127.0.0.1:8080",
  "IsXHR": true,
  "Method": "POST",
  "RequestURI": "\/user_event\/page-unload",
  "ResponseContentLength": 443,
  "ResponseHttpStatus": 200,
  "ResponseTime_ms": 123,
  "Timestamp": "2018-03-02T06:14:57.206349701Z",
  "UserEmail": "nathan@honeycomb.io",
  "UserID": 18,
  "availability_zone": "us-east-1b",
  "build_id": "6552",
  "env": "dogfood",
  "infra_type": "aws_instance",
  "instance_type": "t2.micro",
  "memory_inuse": 15450056,
  "num_goroutines": 56,
  "request_id": "poodle-a38f5e39\/5fIUGkX5D1-001814",
  "server_hostname": "poodle-a38f5e39",
  "type": "request"
},
```

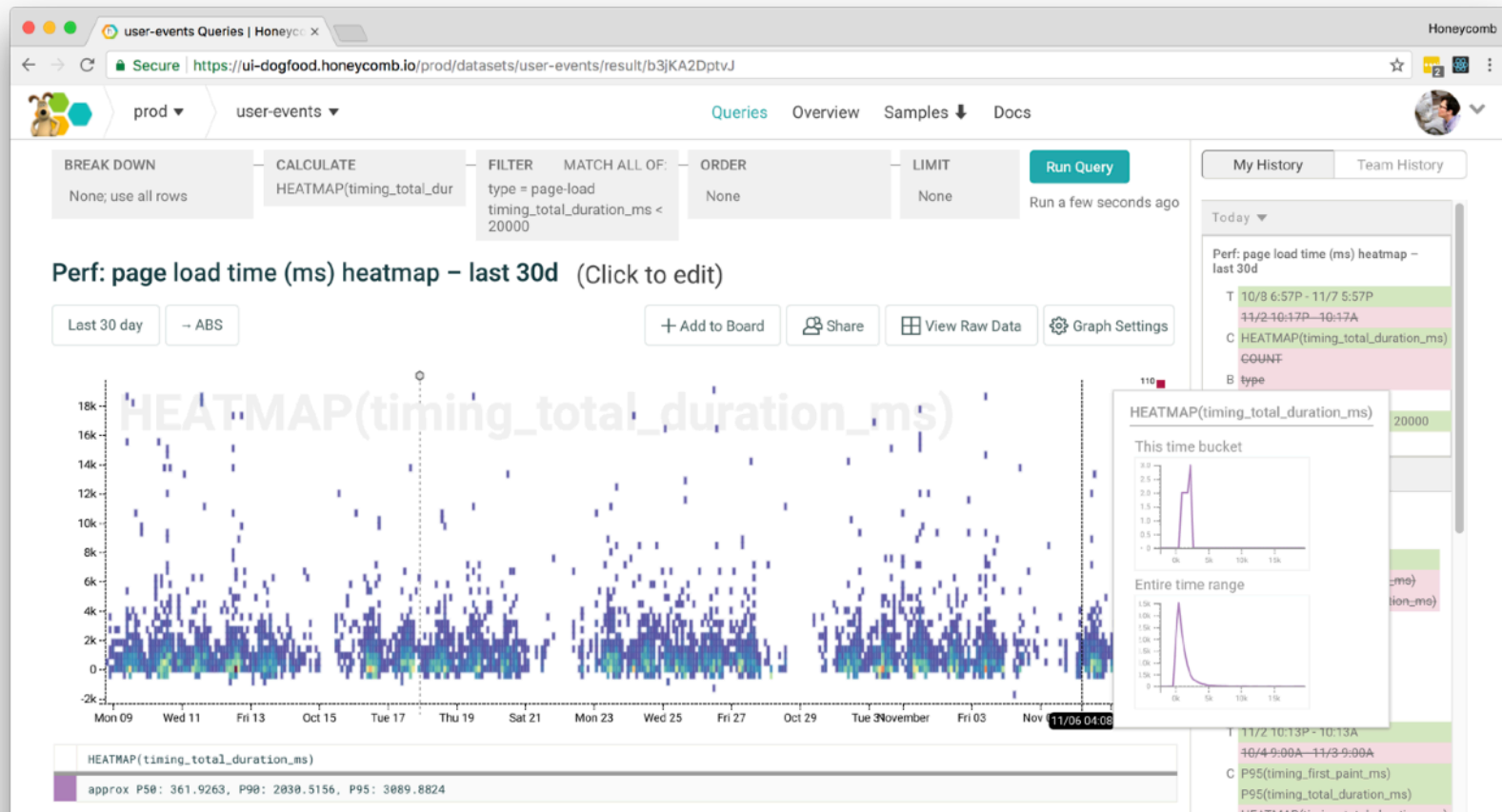




# Browser Instrumentation



# Honeycomb Query Sandbox: React, SCSS, go templates, and lots of data

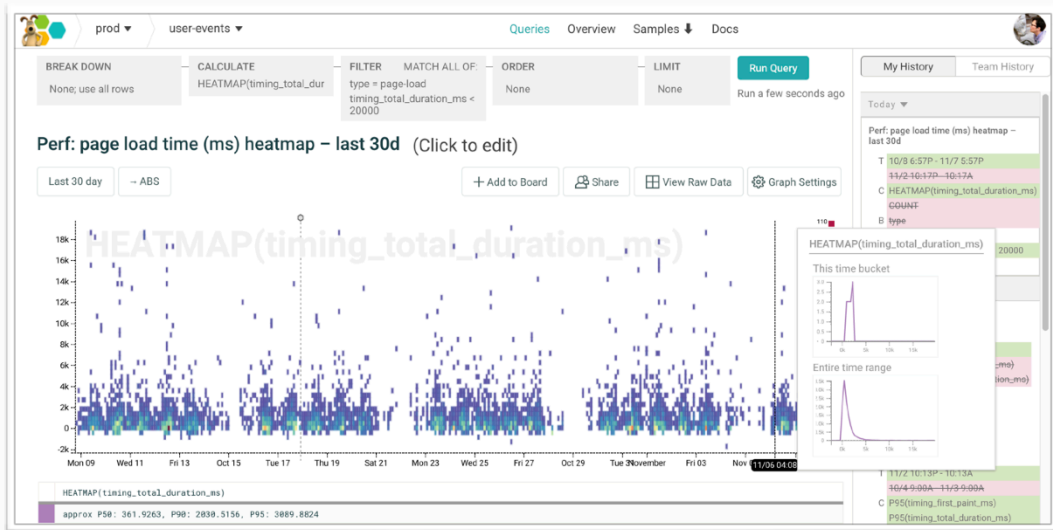


# Instrumentation toolkit

## I. Performance

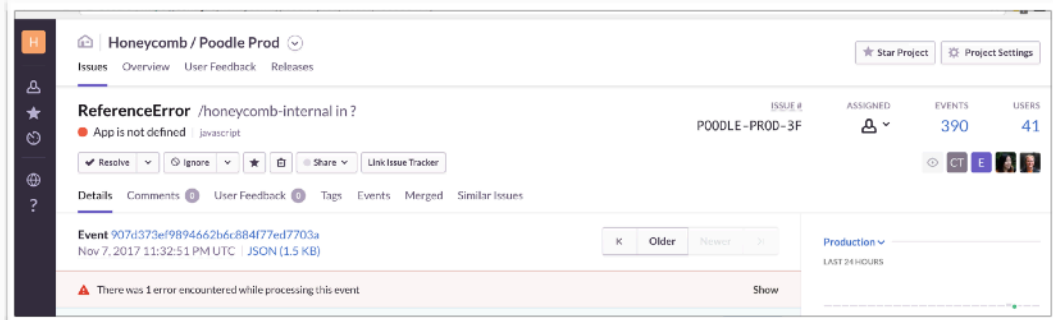
RAIL model

Loading metrics: page load time, resource load time, first paint.



## II. Errors

An event per client-side javascript error, with metadata like stack trace & event breadcrumb trail



# RAIL performance model

<https://developers.google.com/web/fundamentals/performance/rail>



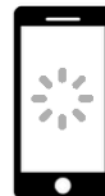
**Response**



**Animation**



**Idle**



**Load**

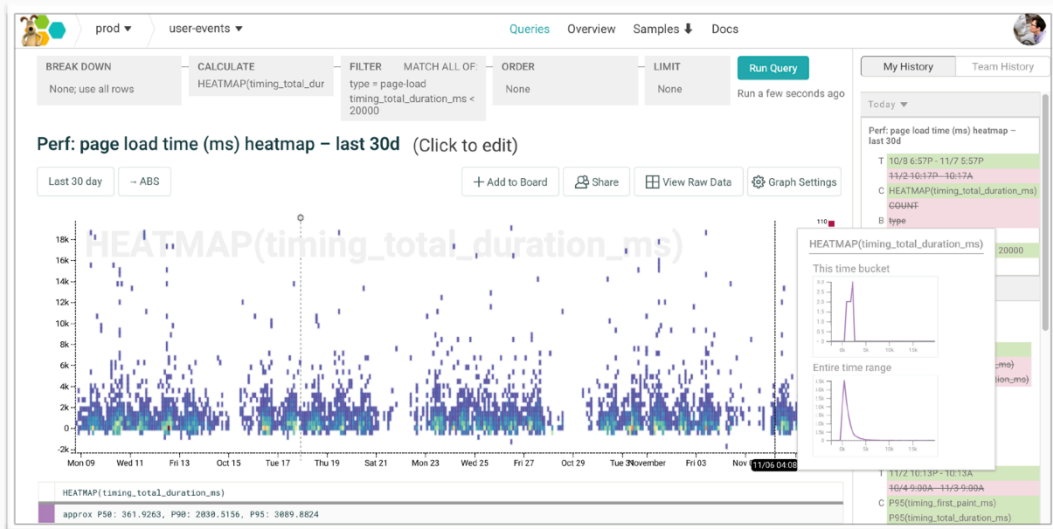


# Instrumentation toolkit

## I. Performance

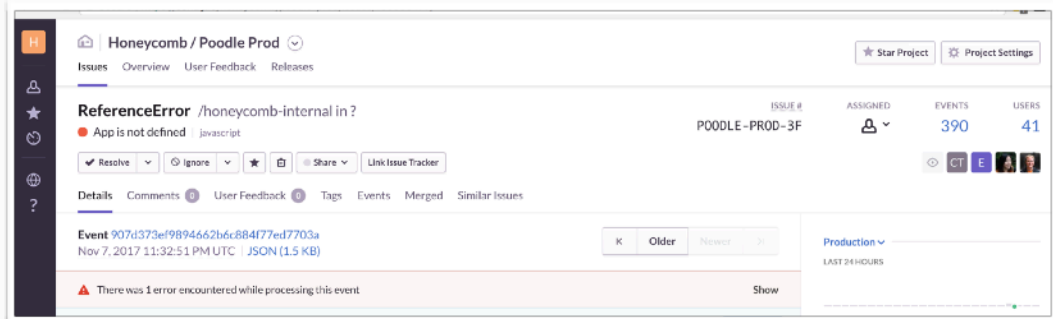
RAIL model

Loading metrics: page load time, resource load time, first paint.



## II. Errors

An event per client-side javascript error, with metadata like stack trace & event breadcrumb trail

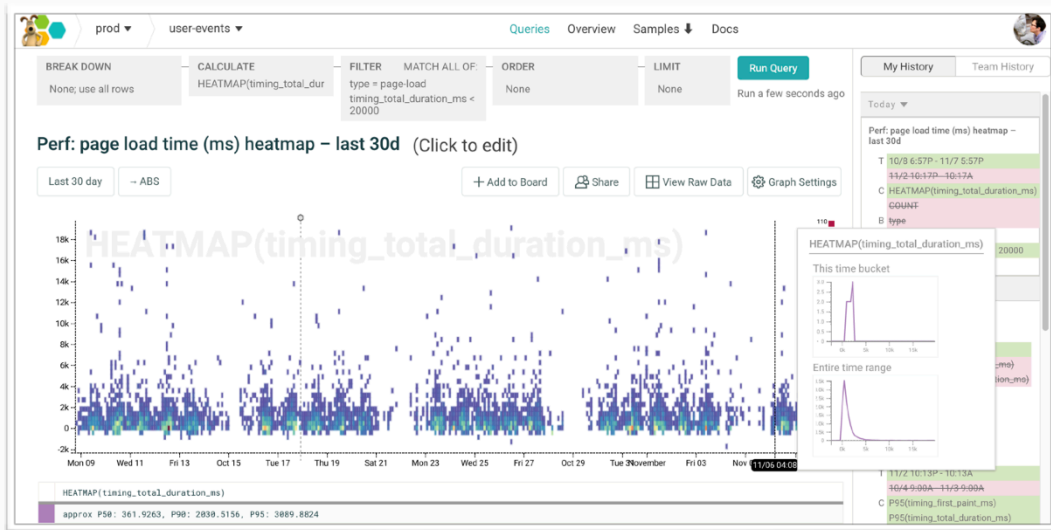


# Instrumentation code

## I. Performance

Write a custom thing

But actually, use Boomerang

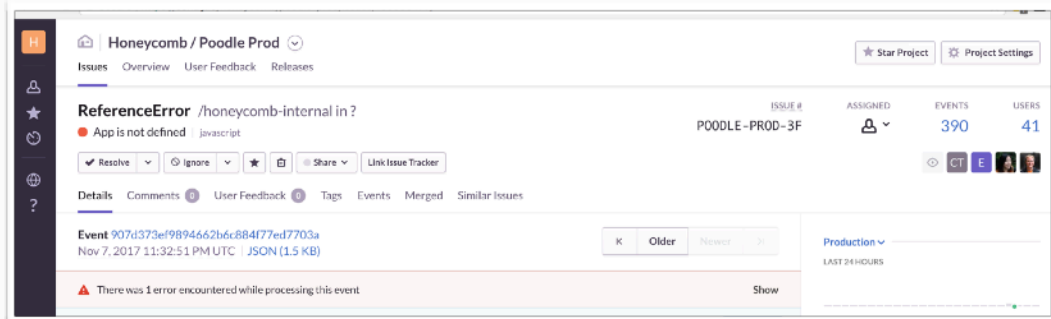


## II. Errors

Sentry's Raven JS is o/s

So is Bugsnag's

... or write a custom thing (no)



# When we fire events

1. **On page load**
2. **On SPA navigation**
3. **On significant user actions**
4. **On page unload**



## Sample page load event

```
{  
  // App-specific  
  type: "page-navigation",  
  page_type: "/:team/datasets/:dataset",  
  user_id: 123,  
  ab_groups:{ touch_ui: true, multi_team_chat: false }  
  
  // Performance / Environment  
  page_load_time_ms: 2145 // plus all navigation timing metrics  
  resource_count: 21  
  asset_version: "1.232.90"  
  canary: false  
  request_id: 123456,  
  
  // Capabilities  
  user_agent: "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/123.45"  
  window_height: 822,  
  window_width: 1145,  
  screen_height: 800,  
  screen_width: 1245,  
  feature_support_emoji: true,  
  feature_support_service_worker: false,  
}
```





# More browser context

- Installed fonts
- Screen dimensions & color depth
- Browser language
- Online/offline status
- Page visibility (backgrounded?)
- Connection type
- Support for emerging browser APIs
- Geographical location (using a library\*)



## Sample SPA navigation event

```
{  
  // Usage  
  type: "react-router-navigation",  
  page_type: "/:team/datasets/:dataset",  
  user_id: 123,  
  ab_group_touch_ui: true,  
  ab_group_multi_team_chat: false,  
  request_id: 123456,  
  
  // Performance / Regression  
  api_request_duration_ms: 2145,  
  api_response_parse_duration_ms: 12,  
  component_render_duration_ms: 42,  
}
```



## Sample user action event

```
{  
  // Usage  
  type: "user-derived-column-add",  
  page_type: "/:team/datasets/:dataset",  
  user_id: 123,  
  ab_groups: { touch_ui: true, multi_team_chat: false, }  
  request_id: 123456,  
  
  feature_column_type: "number",  
}
```

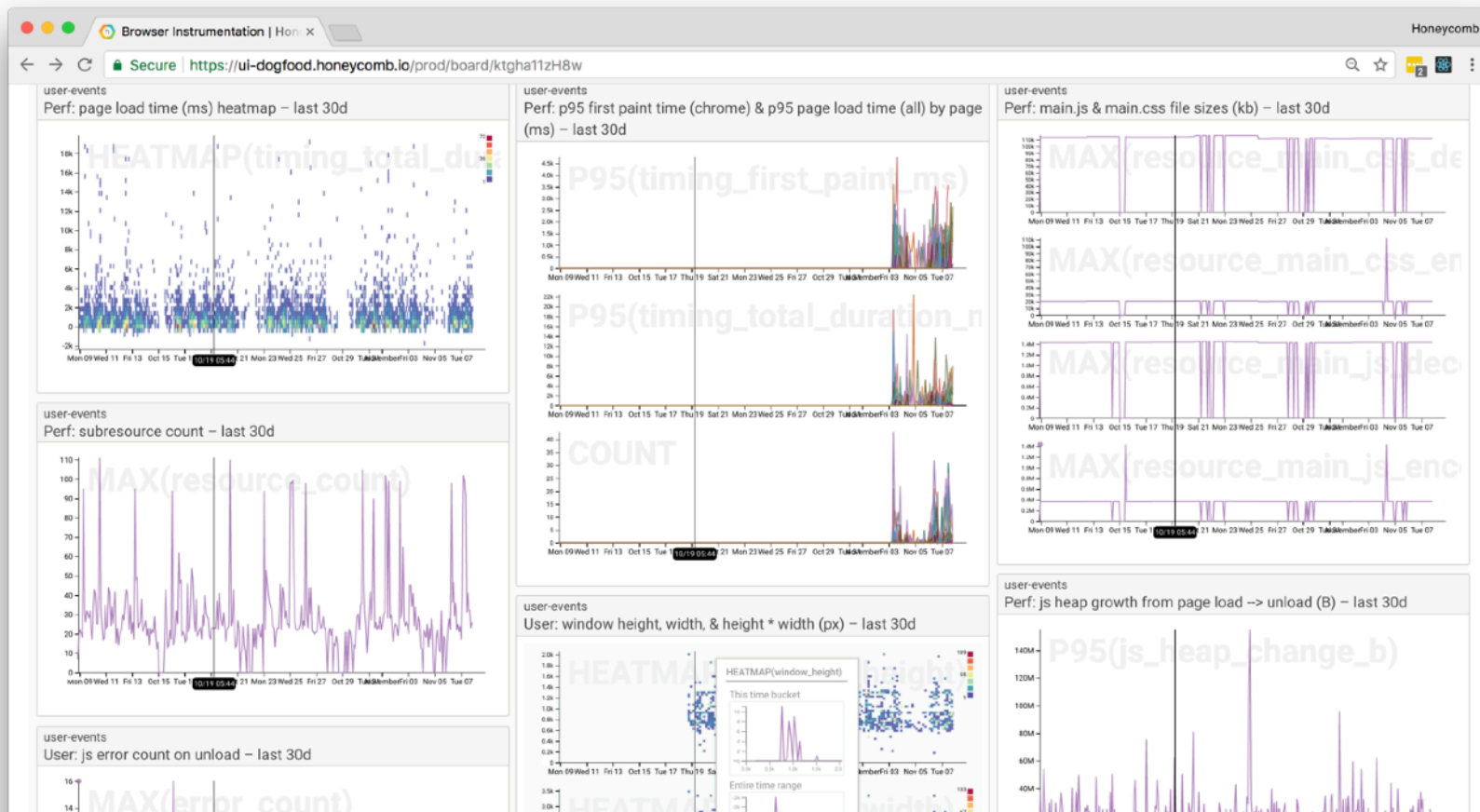


## Sample page unload event

```
{  
  // Usage  
  type: "react-router-navigation",  
  page_type: "/:team/datasets/:dataset",  
  user_id: 123,  
  ab_group_touch_ui: true,  
  ab_group_multi_team_chat: false,  
  
  // Performance / Regression  
  request_id: 123456,  
  js_error_count: 0,  
  window_open_duration_s: 45003,  
  
  // Memory info (Chrome) – also send this on load so we can compare heap size  
  // and understand how much memory we're using as the user interacts with the page.  
  js_heap_size_used_b: 123455,  
  js_heap_change_b: 20000,  
}
```



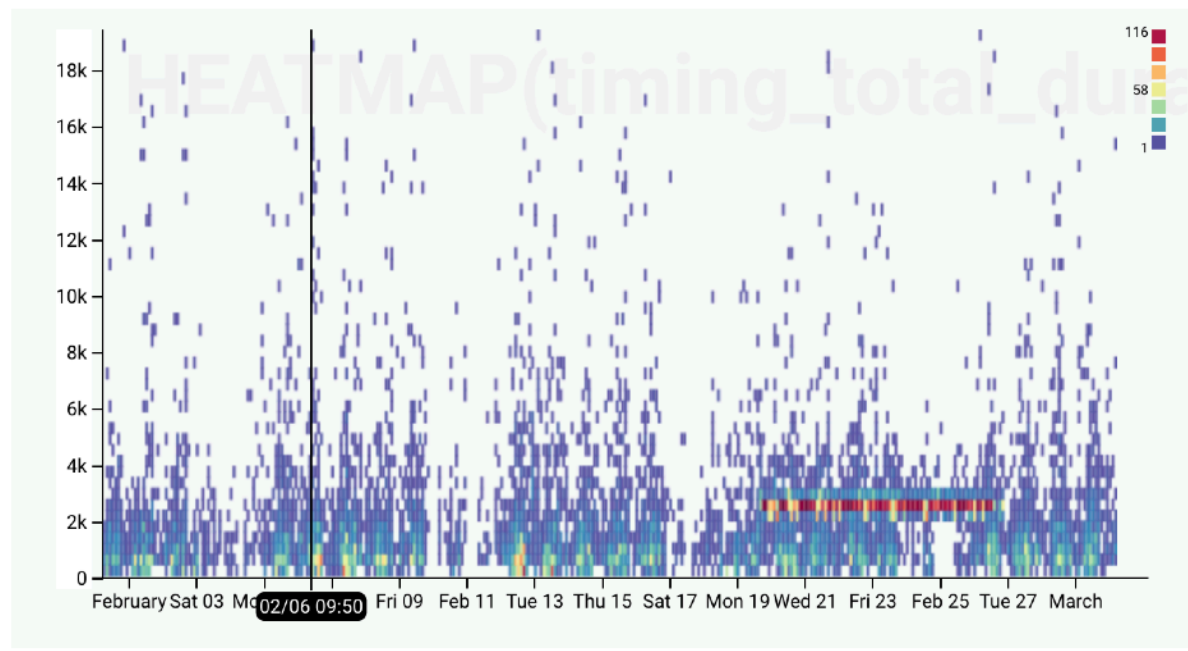
# Honeycomb Query Sandbox: what we graph



# Honeycomb Query Sandbox: what we graph

user-events

Perf: page load time (ms) heatmap – last 30d



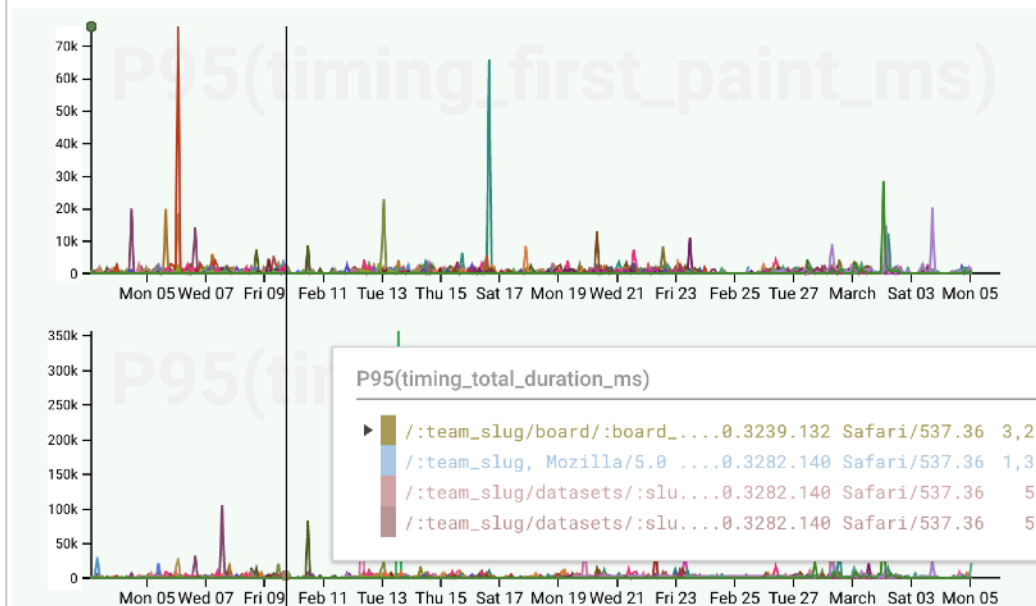
# Honeycomb Query Sandbox: what we graph



# Honeycomb Query Sandbox: what we graph

user-events

Perf: p95 first paint time (chrome) broken down by browser & page type – last 30d





# Tool Choice



# Tool choice: where to send events?

Places to send events (if you don't use Honeycomb)

1. Log aggregator
2. Metrics tools with support for high cardinality labels/tags
3. Error monitoring tool (maybe)



# Debugging performance



# Understanding Normal

How Honeycomb Uses Honeycomb, Part 7: Measure twice, cut once: How we made our queries 50% faster...with data

OCT 19, 2017  
CHRIS TOSHOK

SEARCH THIS BLOG

Search...  
SEARCH

KEEP IN TOUCH

Your email address  
JOIN

RECENT POSTS

Announcing Secondary Storage and the Fast Query Window  
MARCH 1, 2018

Sam Stokes talks about data infrastructure on the Data Engineering Podcast  
FEBRUARY 27, 2018

17 Shares

f  
t  
e  
in  
r  
G+

Tags: [DOGFOODING](#), [PERFORMANCE](#)

*This post continues our dogfooding series from [How Honeycomb Uses Honeycomb, Part 6: Instrumenting a Production Service](#).*

The entire value proposition of Honeycomb's columnar store is speed of queries, for instance:

```
elapsed query time: 330.657ms # results: 50 rows examined: 1,262,255 pct of nodes reporting: 100%
```

Examining over 1.2 million rows and returning 50 time series, aggregating results into 240 (1 minute granularity over 4 hours for this query) buckets per series, in 330ms. Nice.

# Understanding Normal

The screenshot shows the Honeycomb interface for a query named 'shepherd' in the 'prod' environment. The browser address bar shows the URL: `https://ui-dogfood.honeycomb.io/prod/datasets/shepherd/result/wyeE9w3rq5i`. The top navigation bar includes 'Queries', 'Overview', 'Samples', and 'Docs'. Below the navigation, there are controls for 'BREAK DOWN', 'CALCULATE COUNT', 'FILTER', 'ORDER', and 'LIMIT', all currently set to 'None'. A 'Run Query' button is visible on the right.

The main content area displays 'Results (Click to edit)' with a 'Last 7 day' filter and a '- ABS' button. Below these are buttons for '+ Add to Board', 'Share', 'View Raw Data', and 'Graph Settings'. A large teal loading spinner is centered on the page.

The right sidebar shows a 'My History' section with a dropdown menu for 'TODAY'. It contains three time-series charts for 'COUNT' over different periods: '7 days' (10:48 am), '7 days' (10:30 am), and '14 days' (10:30 am). Each chart includes a 'No breakdowns', 'No filters', and 'None' filter option. Below these is a section for 'COUNT OF EACH ROW' with a '2 hours' time range and another 'COUNT OF EACH ROW' section. At the bottom of the sidebar, there is a 'YESTERDAY' dropdown.

The Honeycomb logo is visible in the bottom right corner of the interface.

# Understanding Normal

	P10(query_and_persist_dur_ms)	P25(query_and_persist_dur_ms)	P50(query_and_persist_dur_ms)	P75(query_and_persist_dur_ms)	P90(query_and_persist_dur_ms)
	108.475	198.75	411.4375	1,347	4,060.4



# Understanding Normal

poll_iterations_required	COUNT ▼
1	4,840
2	155
3	109
4	55



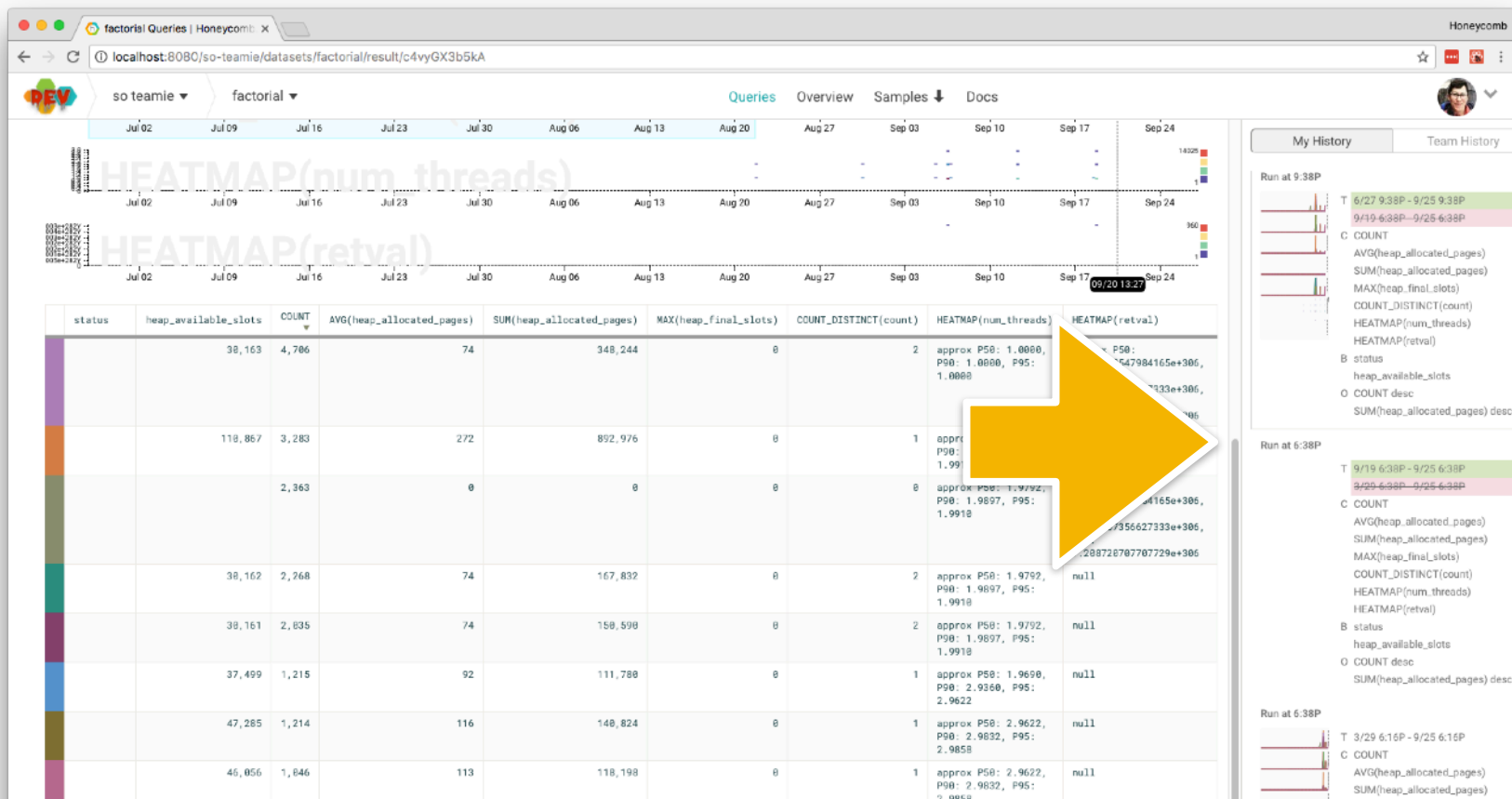
# Looking ahead

Using observability to drive design





# Using observability to drive design



# Using observability to drive design



# Using observability to drive design

Screen percentage used (all endpoints, last 30d) (Click to edit)

Last 30 day

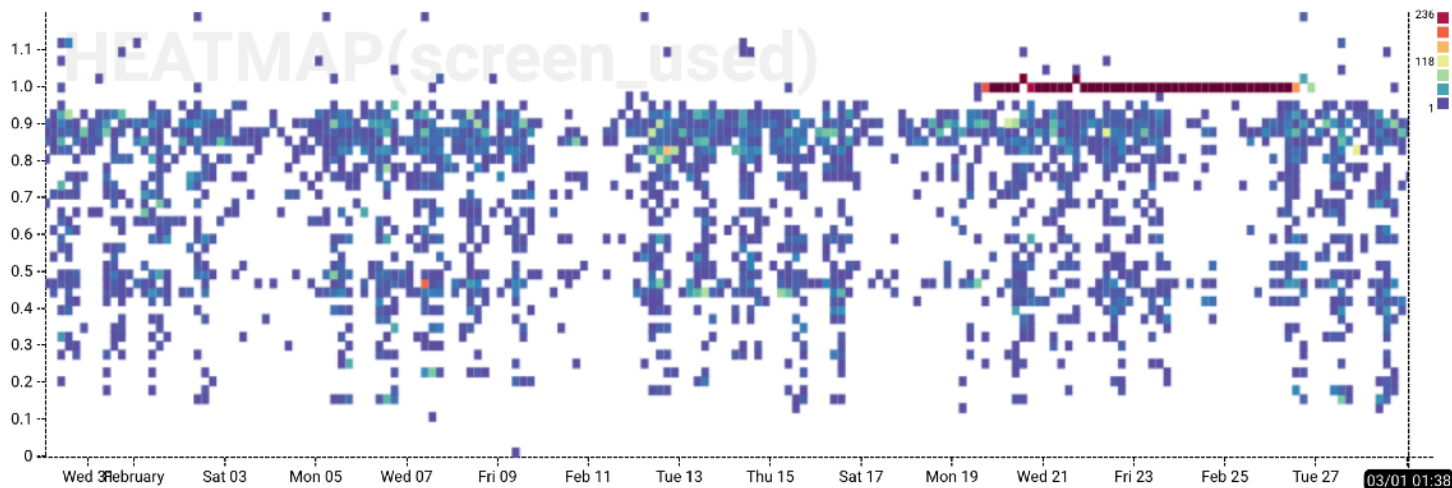
→ ABS

+ Add to Board

Share

View Raw Data

Graph Settings

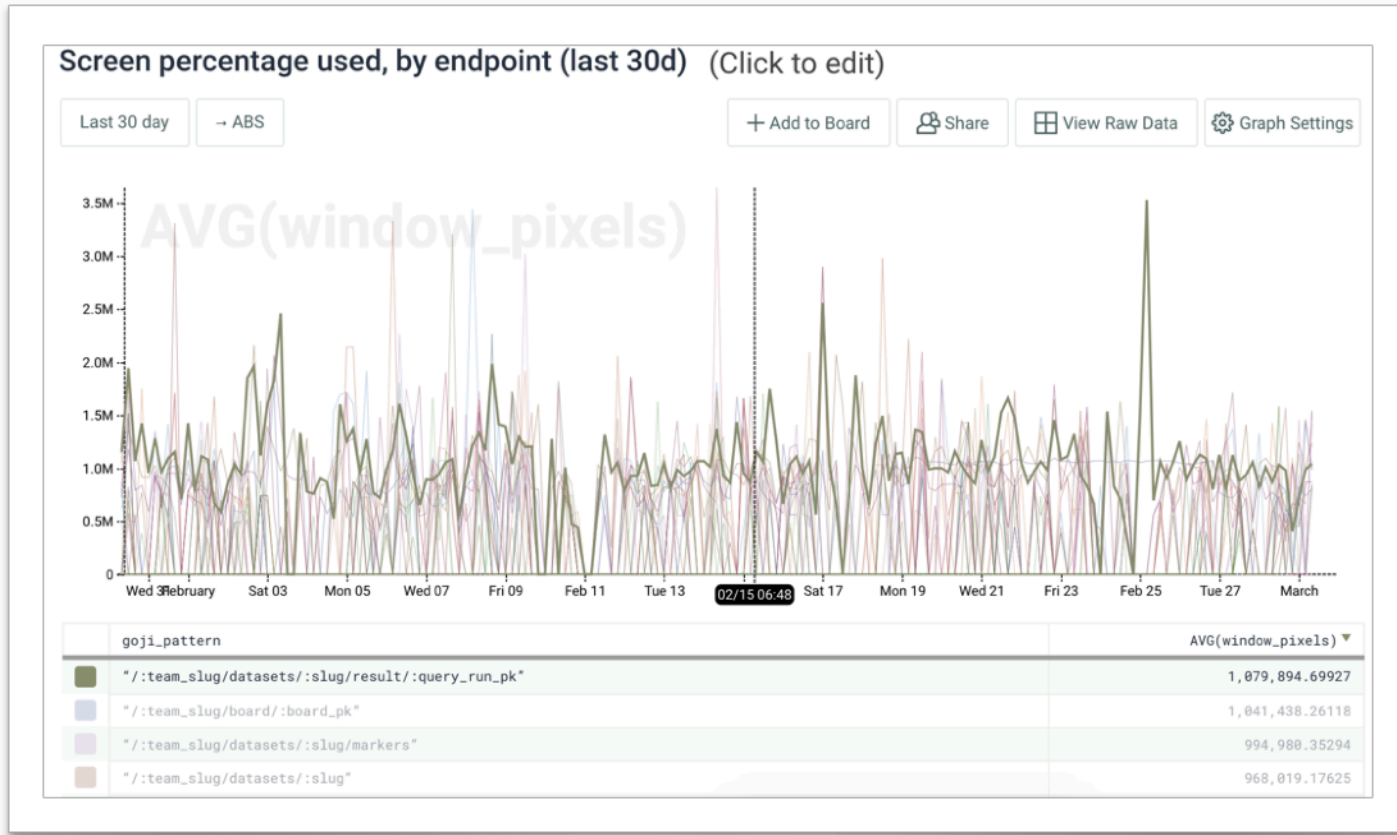


HEATMAP(screen\_used)

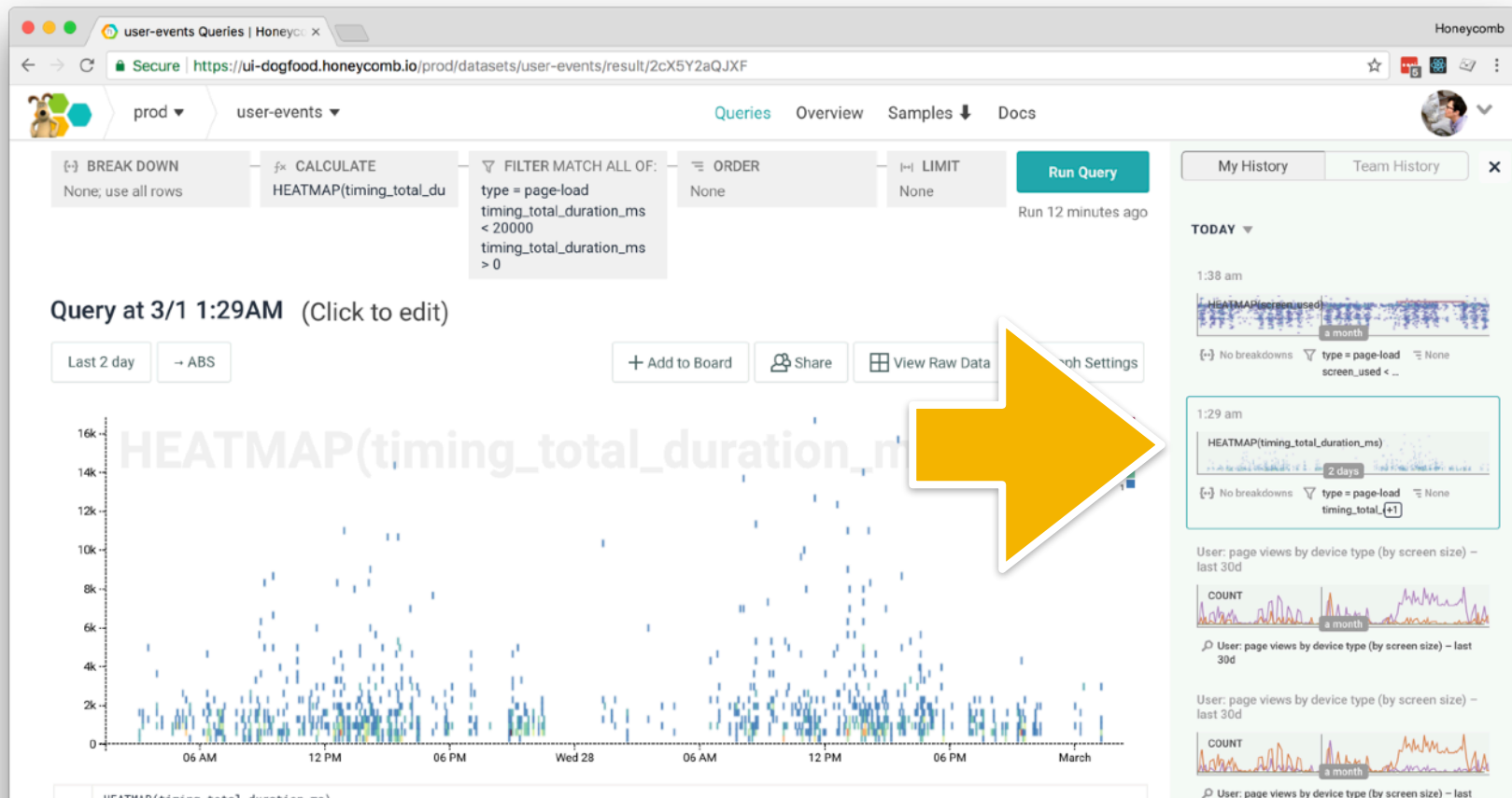
approx P50: 0.8588, P90: 0.9761, P95: 0.9793



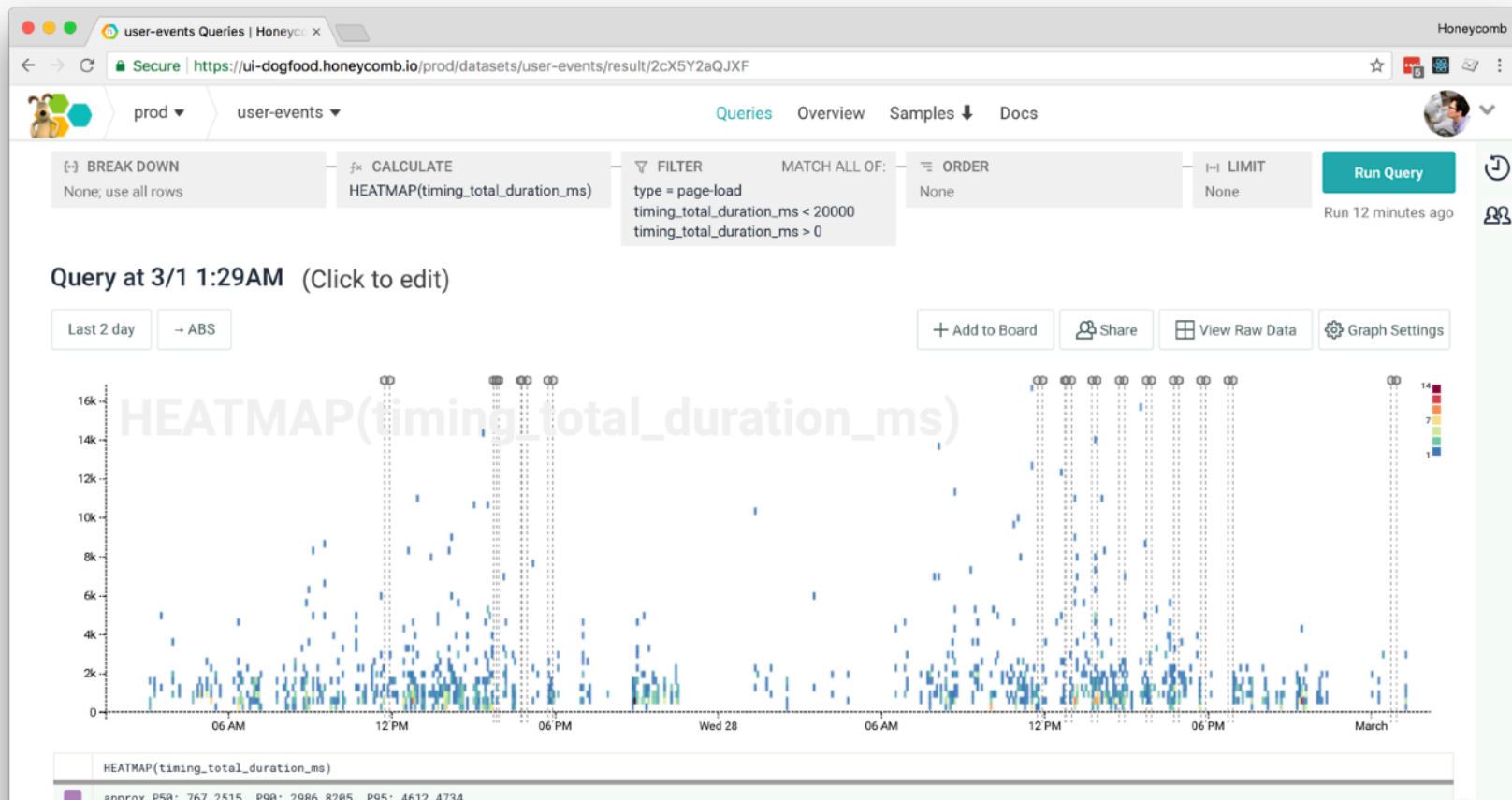
# Using observability to drive design



# Using observability to drive design



# Using observability to drive design



# Understanding Normal

Server-side rendering vs. client side rendering



# Understanding normal

The screenshot displays the Honeycomb Team Datasets interface in a browser window. The browser's address bar shows the URL `localhost:8080/honeycomb-team`. The interface is organized into a sidebar on the left and a main content area on the right.

**Sidebar (Left):**

- Nest**: Settings & Overview, 3mo ago 3mo ago
- poodle**: Settings & Overview, 4mo ago 19d ago
- poodle csp violations**: Settings & Overview, 4mo ago 3mo ago
- retriever-grpc-conn**: Settings & Overview, 4mo ago 20d ago (highlighted with a blue box)
- Test**: Settings & Overview, 3mo ago 3mo ago
- test js**: Settings & Overview, 3mo ago 3mo ago
- user-events**: Settings & Overview

**Main Content Area (Right):**

**Recent Team Activity**

- Emily • 1 hour ago • Ran 10+ queries in user-events**
  - 5:05 pm: P50(datasetRenderDuration) - 30 minutes
  - 3:14 pm: P50(datasetRenderDuration) - 30 minutes
  - 8+ more
- Emily • 6 hours ago • Ran 10+ queries in poodle**
  - 12:09 pm: COUNT, AVG(ResponseTime\_ms) - 14 days
  - 12:09 pm: COUNT, AVG(ResponseTime\_ms) - 2 days
  - 8+ more

The interface includes various interactive elements such as dropdown menus for filters (e.g., "type = dataset...", "No filters") and "No breakdowns" options. The charts show data trends over time, with some showing spikes or sustained activity.





## Page load event with server timings

```
{
  type: "page-navigation",
  page_type: "[:team/datasets/:dataset",

  // Performance / Environment
  page_load_time_ms: 2145 // plus all navigation timing metrics
  resource_count: 21
  asset_version: "1.232.90"
  canary: false

  // Already have this from window.performance navigation
  server_request_dur_ms: performance.timing.responseEnd - performance.timing.navigationStart,

  // New timings
  template_server_render_dur_ms: 12,
  time_to_component_rendered_ms: { dataset_usage_viz: 156 },

  request_id: 123456,
}
```



```
// capture time-to-component-rendered with react
```

```
class DatasetUsageViz extends React.Component {
```

```
  componentDidMount() {
```

```
    // updatePageLoadEventPayload will merge this payload with  
    // our global event context so these fields appear on the  
    // page-load or page-unload event.
```

```
    updatePageLoadEventPayload({
```

```
      time_to_component_rendered_ms: {
```

```
        dataset_usage_viz: window.performance.now()
```

```
      },
```

```
    });
```

```
  };
```

```
  // ...
```

```
}
```



```
// capture time-to-component-rendered with MutationObserver
// only run if there is feature support (older clients won't send data)
if (window.MutationObserver) {
  const container = document.querySelector('dataset-container');

  const observer = new MutationObserver(function(mutations) {
    mutations.forEach(function(mutation) {

      if ( /* check if mutated dom is in expected "done" state */ ) {
        // updatePageLoadEventPayload will merge this payload with
        // our global event context so these fields appear on the
        // page-load or page-unload event.
        updatePageLoadEventPayload({
          time_to_component_rendered_ms: {
            dataset_usage_viz: window.performance.now()
          },
        });
      }
    });
  });

  // Start observing! Pass in the target node as well as the observer config
  observer.observe(container, { childList: true });
}
```



# Understanding normal

The screenshot displays the Honeycomb Team Datasets interface in a browser window. The address bar shows `localhost:8080/honeycomb-team`. On the left, a sidebar lists several datasets: **Nest**, **poodle**, **poodle csp violations**, **retriever-grpc-conn**, **Test**, **test js**, and **user-events**. Each dataset has a 'Settings & Overview' link and a progress bar.

The main content area is titled **Recent Team Activity**. It features two activity entries by user **Emily**:

- Emily • 1 hour ago • Ran 10+ queries in user-events**: This entry includes two line charts. The first chart, titled **P50(datasetRenderDuration)**, shows a spike at 5:05 pm with a 30-minute time range. The second chart, also titled **P50(datasetRenderDuration)**, shows a spike at 3:14 pm with a 30-minute time range. A link for **8+ more** is visible to the right of the second chart.
- Emily • 6 hours ago • Ran 10+ queries in poodle**: This entry includes two line charts. The first chart shows **COUNT** and **AVG(ResponseTime\_ms)** metrics with a spike at 12:09 pm over a 14-day period. The second chart shows **COUNT** and **AVG(ResponseTime\_ms)** metrics with a spike at 12:09 pm over a 2-day period. A link for **8+ more** is visible to the right of the second chart.

Each chart includes a legend for 'No breakdowns', a filter dropdown (e.g., 'type = dataset...'), and a 'None' option.



# Privacy



# Privacy: no new data, but new problems

1. Pipe events through your own infrastructure
2. Consider whether a field will truly be helpful for debugging
  - Is there a less-revealing field value that would be as helpful?
3. Check the data retention policies & practices of your SaaS products



# High-performance instrumentation



# Send events in a non-blocking way

High-performance browser instrumentation code:

1. Use the Beacon API to send events in a non-blocking way
2. Use `requestIdleCallback` or `setTimeout` to schedule slow calculations
3. Wrap rendering timers in a `requestAnimationFrame`





# Sampling

Select & send representative events instead of sending *all* events.

1. Watch Ben's talk from LISA17: Sample Your Traffic but Keep the Good Stuff
2. Can be either client-side or server-side
3. Use the `dynsampler-js` library or write your own code
4. Be sure to set the ``sample_rate`` field on your events



# Epilogue



# Epilogue

The screenshot shows a web browser window with the URL 'Topo Designs' and a user email 'nakashima@aya...'. The page features a navigation bar with a search bar containing the text 'Find amazing products', and links for 'JOIN' and 'SIGN IN'. The main content area displays a grid of ten product cards, each with an image, a title, and a price.

Product Name	Price
Heavyweight Pocket Tee	\$45.00
Heavyweight Long Sleeve Pocket Tee	\$59.00
Mountain Shirt - Plaid Flannel	\$129.00
Women's Coverall	\$189.00
Topo Designs x Howler Brothers Mountain Pack	\$198.00
Work Pants - Natural Canvas	\$149.00
Breaker Shirt Jacket	\$149.00
3-Day Briefcase	\$189.00
Commuter Briefcase	\$189.00
Climb Pants	\$129.00



# Epilogue

## Results (Click to edit)

Last 30 day

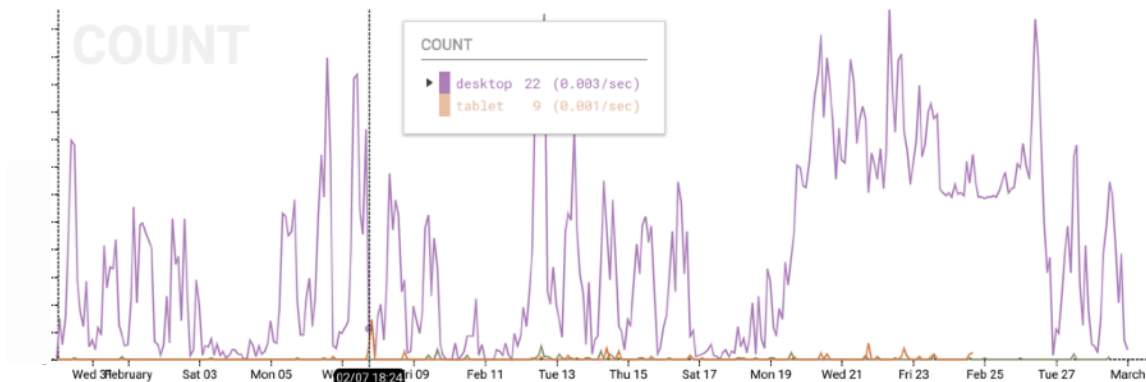
→ ABS

+ Add to Board

Share

View Raw Data

Graph Settings



device	COUNT
*desktop*	26,036
*tablet*	125
*phone*	94

elapsed query time: 189.586ms rows examined: 80,544 pct of nodes reporting: 100%

TODAY

1:01 am

COUNT

device

type = page-load

COUNT

User: page views by goji pattern - last 30d

COUNT

User: page views by goji pattern - last 30d

YESTERDAY

10:47 pm

HEATMAP(timing\_total\_duration\_ms)

No breakdowns

type = page-load

None

timing\_total [+1]

10:47 pm

HEATMAP(timing\_total\_duration\_ms)

No breakdowns

type = page-load

None

timing\_total [+1]

FRIDAY 2/23

11:13 am



# Epilogue

## User: subrequests by endpoint – last 30d (Click to edit)

Last 30 day

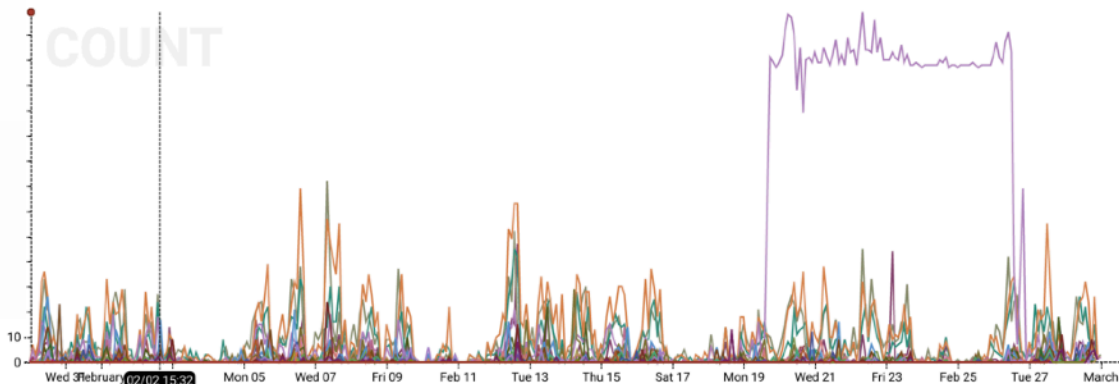
→ ABS

+ Add to Board

Share

View Raw Data

Graph Settings



goji_pattern	COUNT
"/:team_slug/board/:board_pk"	11,197
"/:team_slug"	4,284
"/:team_slug/datasets/:slug/result/:query_run_pk"	3,284
"/:team_slug/datasets/:slug"	2,655
"/:team_slug/datasets/:slug/overview"	846

### User: subrequests by endpoint – last 30d



User: subrequests by endpoint – last 30d

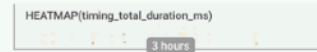
### Perf: subresource count – last 30d



Perf: subresource count – last 30d

### YESTERDAY

10:47 pm



No breakdowns type = page-load timing\_total [+1]

10:47 pm



No breakdowns type = page-load timing\_total [+1]

### FRIDAY 2/23

11:13 am



Thank you!

@eanakashima

honeycomb.io

