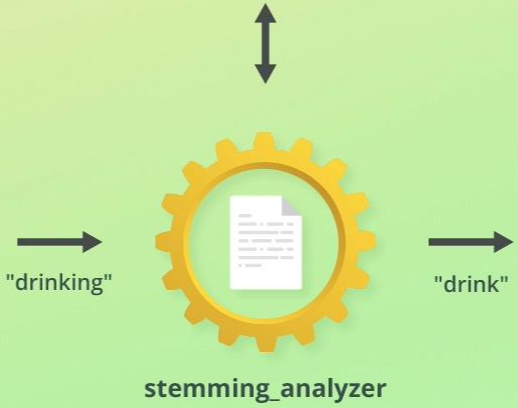
Stemming Analyzer

```
{
  "properties": {
    "description": {
      "type": "text",
      "analyzer": "stemming_analyzer"
    }
  }
}
```

```
GET /stemming_test/_search
{
  "query": {
    "match": {
      "description": "drinking"
    }
  }
}
```



TERM	DOCUMENT #1	...
i	X	
love	X	
drink	X	
bottl	X	
of	X	
wine	X	
on	X	
last	X	
year	X	
vacat	X	

Built-in analyzers

Standard Analyzer

- Splits text at word boundaries and removes punctuation
  - Done by the `standard` tokenizer
- Lowercases letters with the `lowercase` token filter
- Contains the `stop` token filter (disabled by default)

Standard Analyzer

Simple Analyzer

- Similar to the `standard` analyzer
  - Splits into tokens when encountering *anything else* than letters
- Lowercases letters with the `lowercase` tokenizer
  - Unusual and a performance hack

Simple Analyzer

Whitespace Analyzer

- Splits text into tokens by whitespace
- Does *not* lowercase letters

```
"Is that Peter's cute-looking dog?"
```

```
["Is", "that", "Peter's", "cute-looking", "dog?"]
```

Keyword Analyzer

- No-op analyzer that leaves the input text intact
  - It simply outputs it as a single token
- Used for `keyword` fields by default
  - Used for exact matching

Keyword Analyzer

Pattern Analyzer

- A regular expression is used to match token separators
  - It should match whatever should split the text into tokens
- This analyzer is very flexible
- The default pattern matches all non-word characters (`\W+`)
- Lowercases letters by default

Pattern Analyzer(Default Configuration)

Analyzers

Elasticsearch Reference [master] » Text analysis » **Built-in analyzer reference**

« Specify an analyzer | Fingerprint Analyzer »

Built-in analyzer reference

edit

Elasticsearch ships with a wide range of built-in analyzers, which can be used in any index without further configuration:

Standard Analyzer
: The `standard` analyzer divides text into terms on word boundaries, as defined by the Unicode Text Segmentation algorithm. It removes most punctuation, lowercases terms, and supports removing stop words.

Simple Analyzer
: The `simple` analyzer divides text into terms whenever it encounters a character which is not a letter. It lowercases all terms.

Whitespace Analyzer
: The `whitespace` analyzer divides text into terms whenever it encounters any whitespace character. It does not lowercase terms.

Stop Analyzer
: The `stop` analyzer is like the `simple` analyzer, but also supports removal of stop words.

Keyword Analyzer
: The `keyword` analyzer is a "noop" analyzer that accepts whatever text it is given and outputs the exact same text as a single term.

Pattern Analyzer
: The `pattern` analyzer uses a regular expression to split the text into terms. It supports lower-casing and stop words.

Language Analyzers
: Elasticsearch provides many language-specific analyzers like `english` or `french`.

Fingerprint Analyzer
: The `fingerprint` analyzer is a specialist analyzer which creates a fingerprint which can be used for duplicate detection.

Custom analyzers

edit

If you do not find an analyzer suitable for your needs, you can create a `custom` analyzer which combines the appropriate character filters, tokenizer, and token filters.

Elasticsearch Video

Optimizing Elastic for Search at McQueen Solutions.

Watch now

+ Elasticsearch Reference: master
+ Elasticsearch introduction
+ Getting started with Elasticsearch
+ Set up Elasticsearch
+ Upgrade Elasticsearch
+ Aggregations
+ Query DSL
  Search across clusters
  Long-running searches
+ Scripting
+ Mapping
- Text analysis
  - Overview
  + Concepts
  + Configure text analysis
  - **Built-in analyzer reference**
    - Fingerprint Analyzer
    - Keyword Analyzer
    - Language Analyzers
    - Pattern Analyzer
    - Simple Analyzer

Analyzers

```
PUT /english_example
{
  "settings": {
    "analysis": {
      "filter": {
        "english_stop": {
          "type":       "stop",
          "stopwords":  "_english_" (1)
        },
        "english_keywords": {
          "type":       "keyword_marker",
          "keywords":   ["example"] (2)
        },
        "english_stemmer": {
          "type":       "stemmer",
          "language":   "english"
        },
        "english_possessive_stemmer": {
          "type":       "stemmer",
          "language":   "possessive_english"
        }
      },
      "analyzer": {
        "rebuilt_english": {
          "tokenizer":  "standard",
          "filter": [
            "english_possessive_stemmer",
            "lowercase",
            "english_stop",
            "english_keywords",
            "english_stemmer"
          ]
        }
      }
    }
  }
}
```

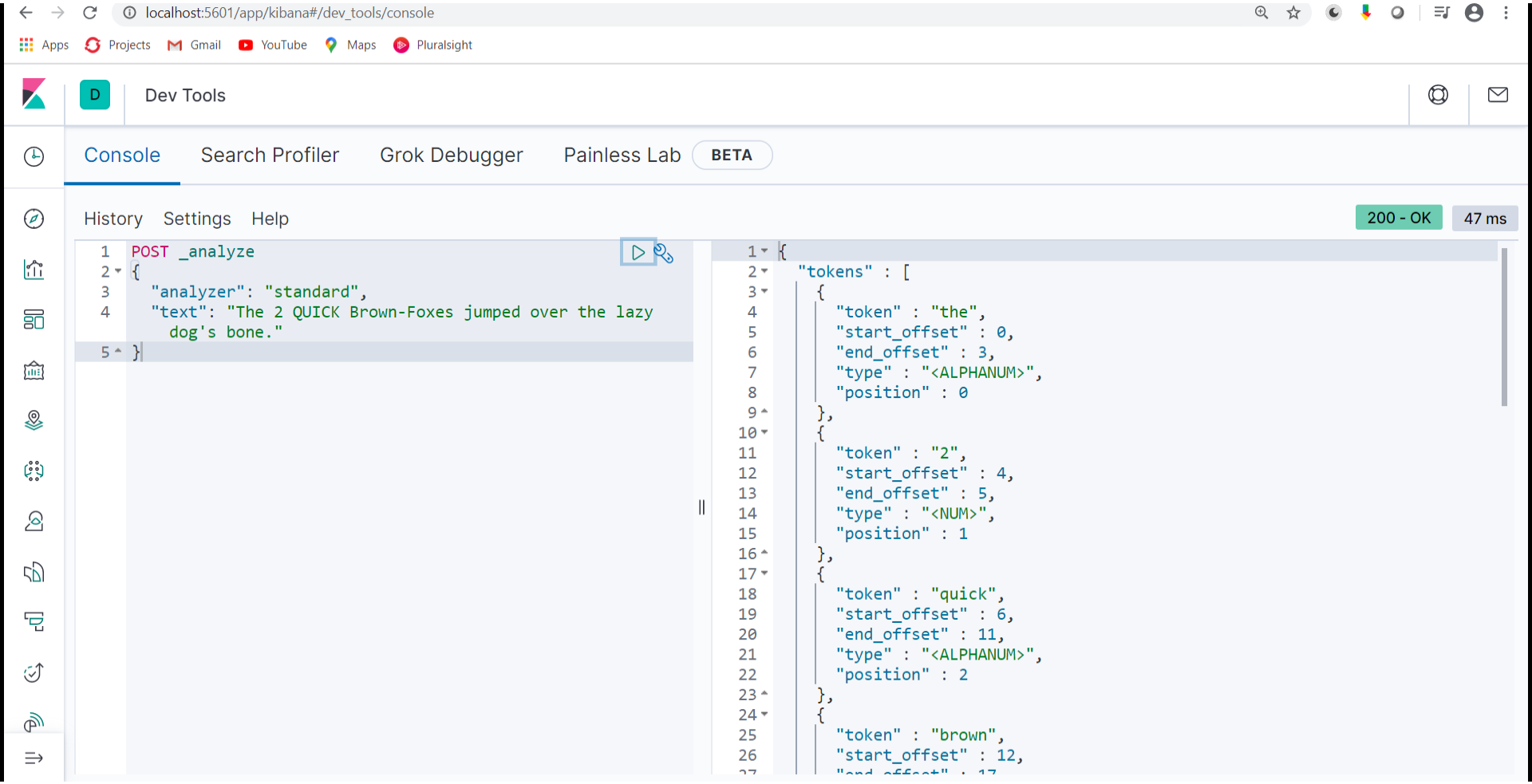
Analyzers

```
PUT /products
{
  "mappings": {
    "properties": {
      "description": {
        "type": "text",
        "analyzer": "english"
      }
    }
  }
}
```

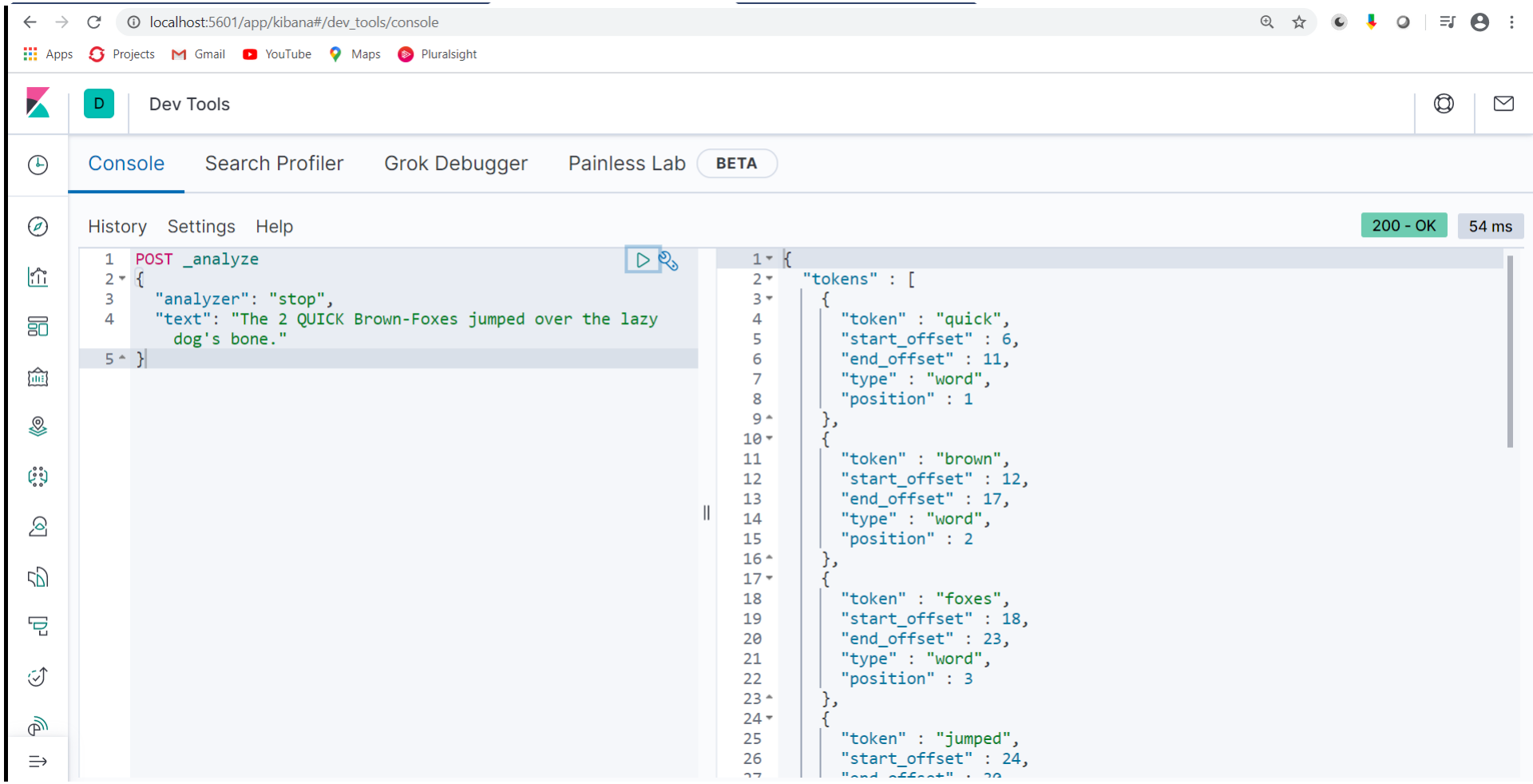
```
POST /products/_doc
{
  "description": "Is that
   Peter's cute-looking dog?"
}
```

```
["peter", "cute", "look", "dog"]
```

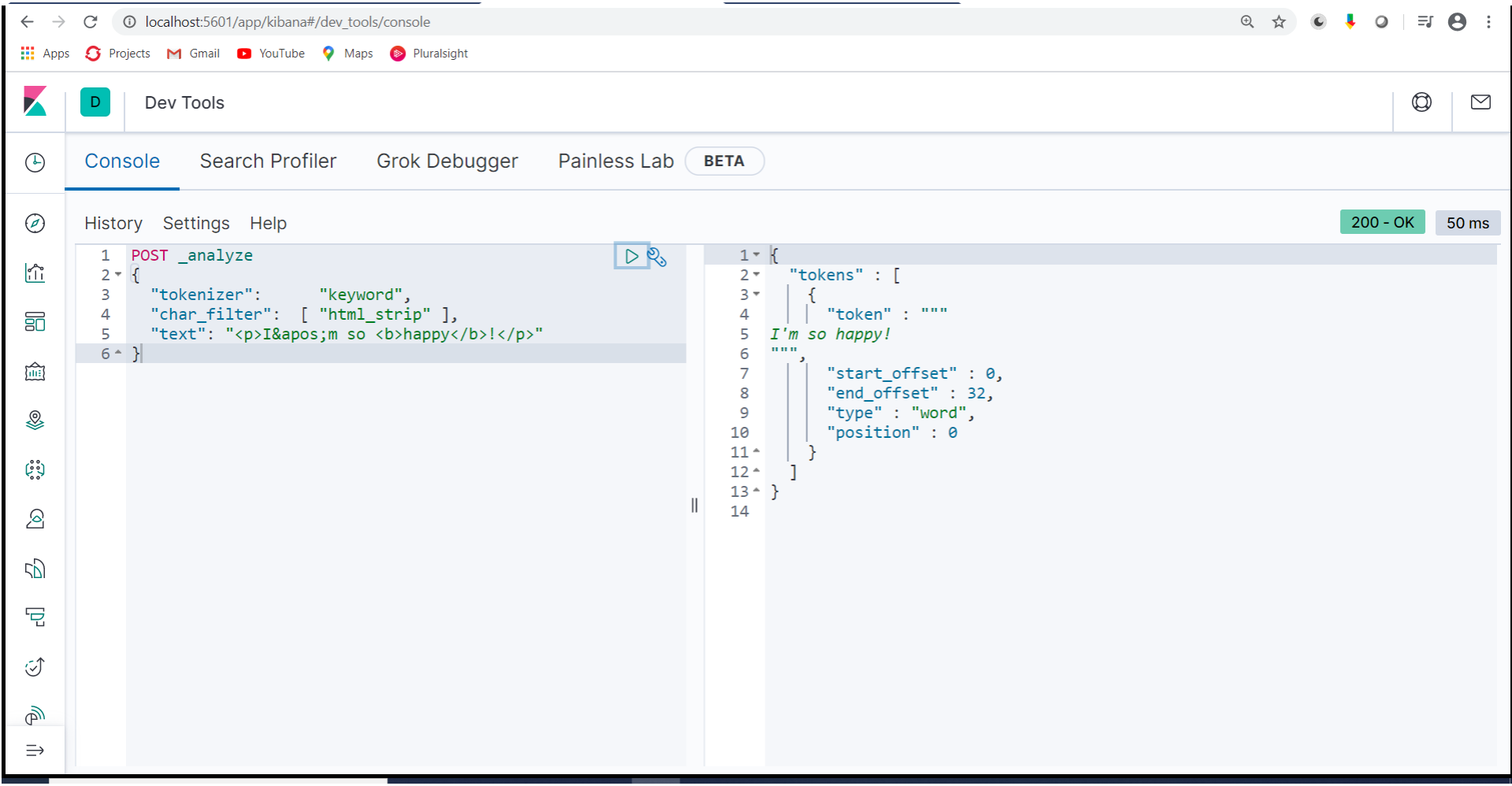
Standard Analyzer



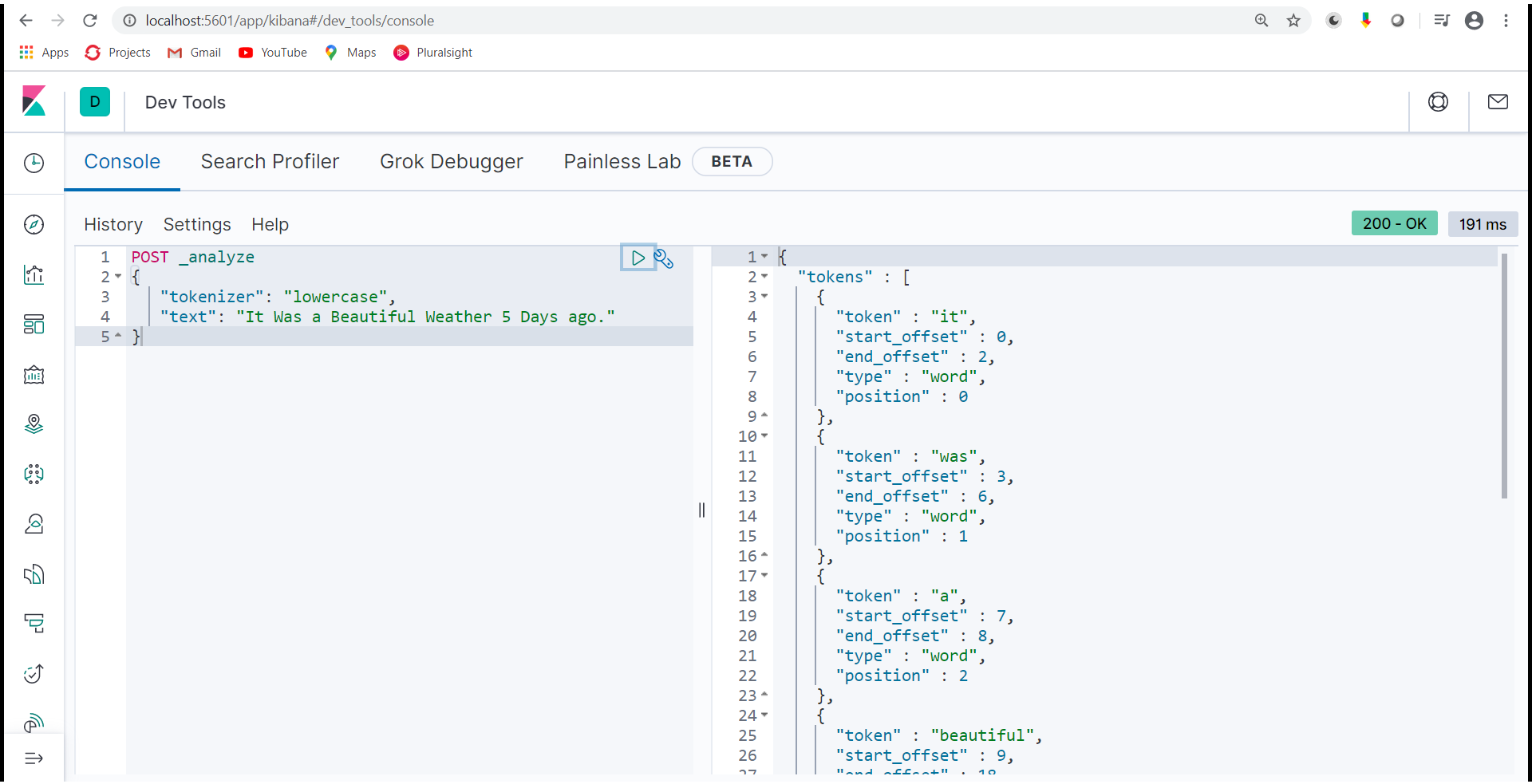
Stop Analyzer



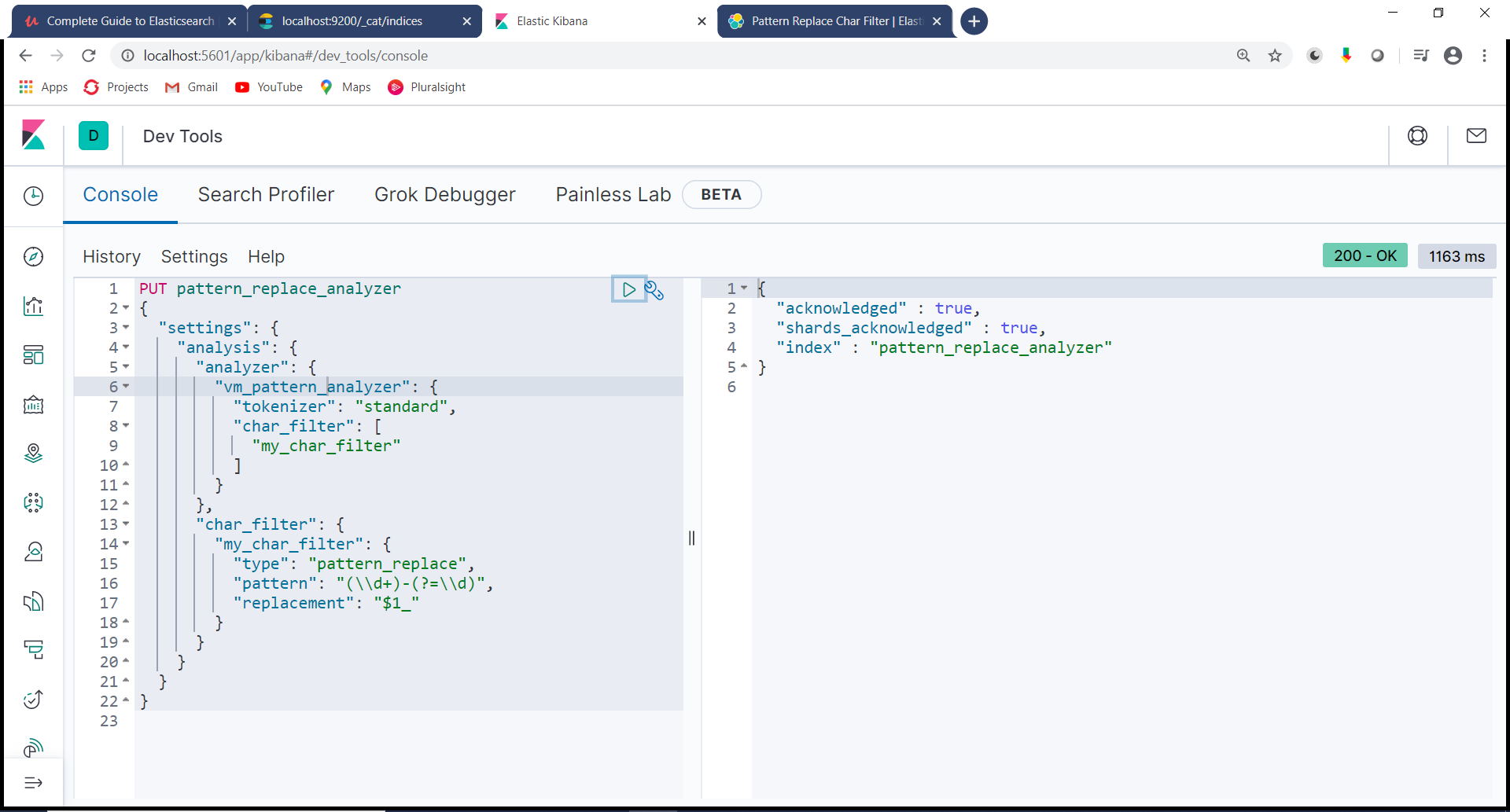
Html_strip Analyzer

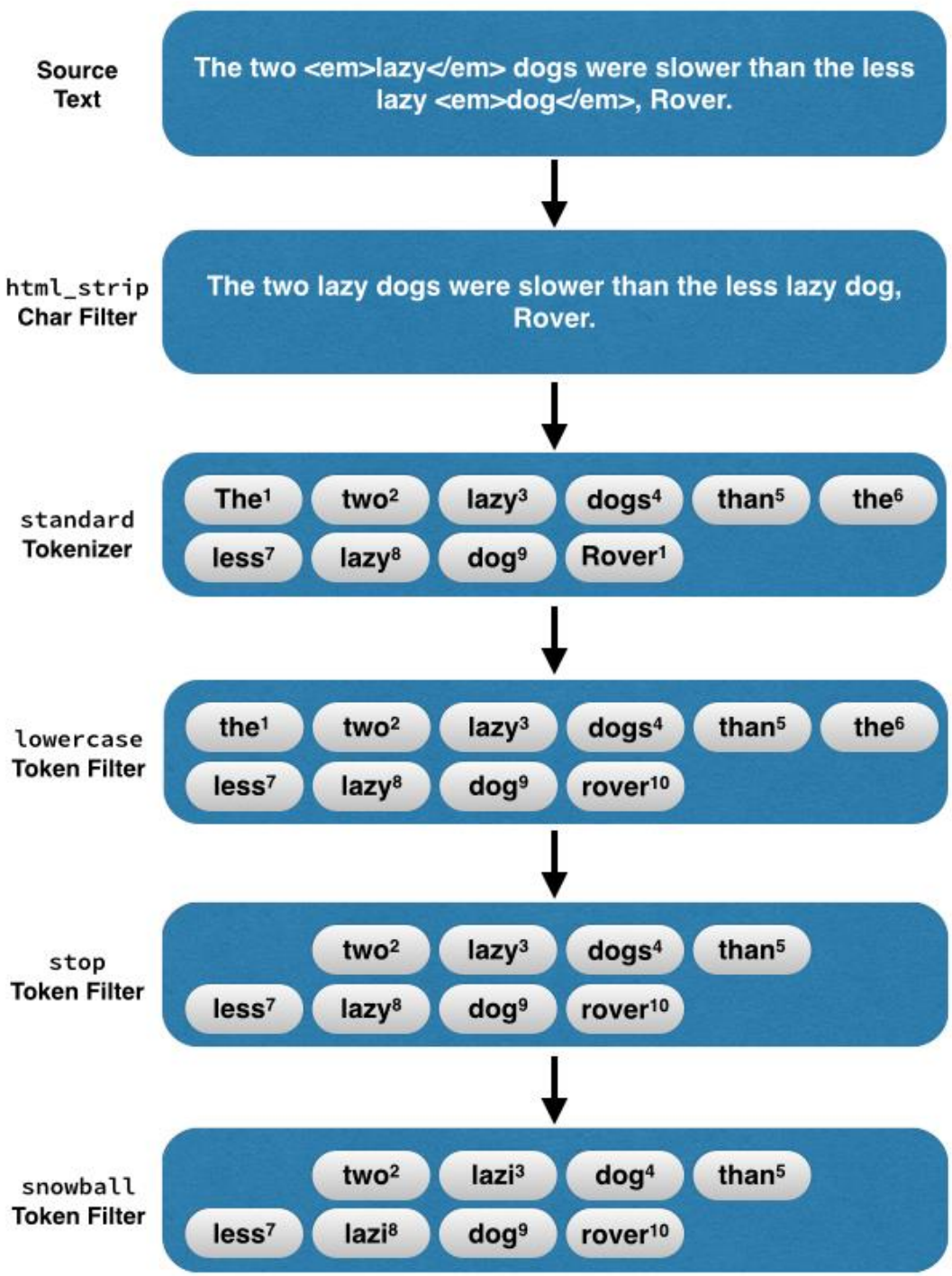


Lowercase Tokenizer

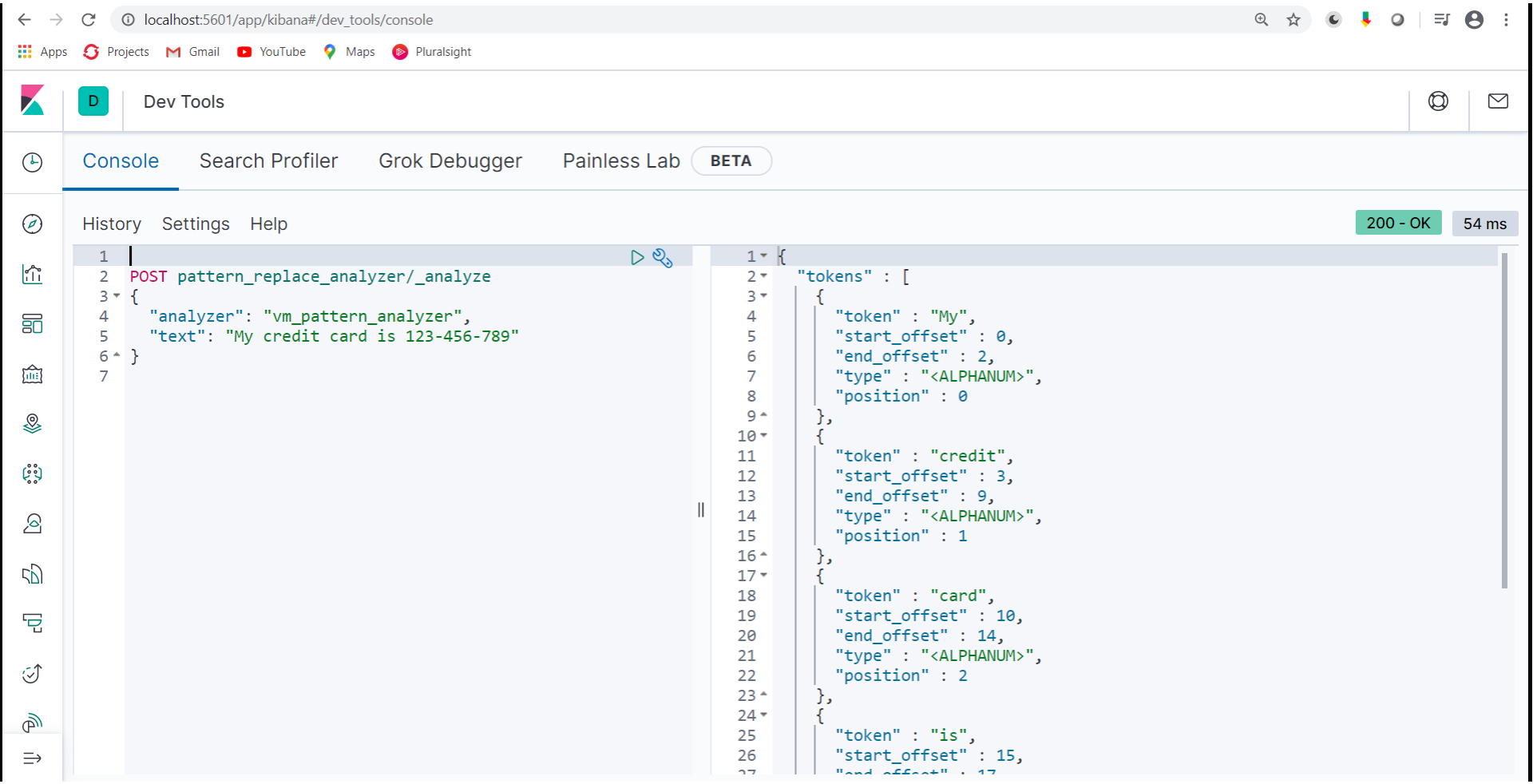


Pattern Replace Tokenizer

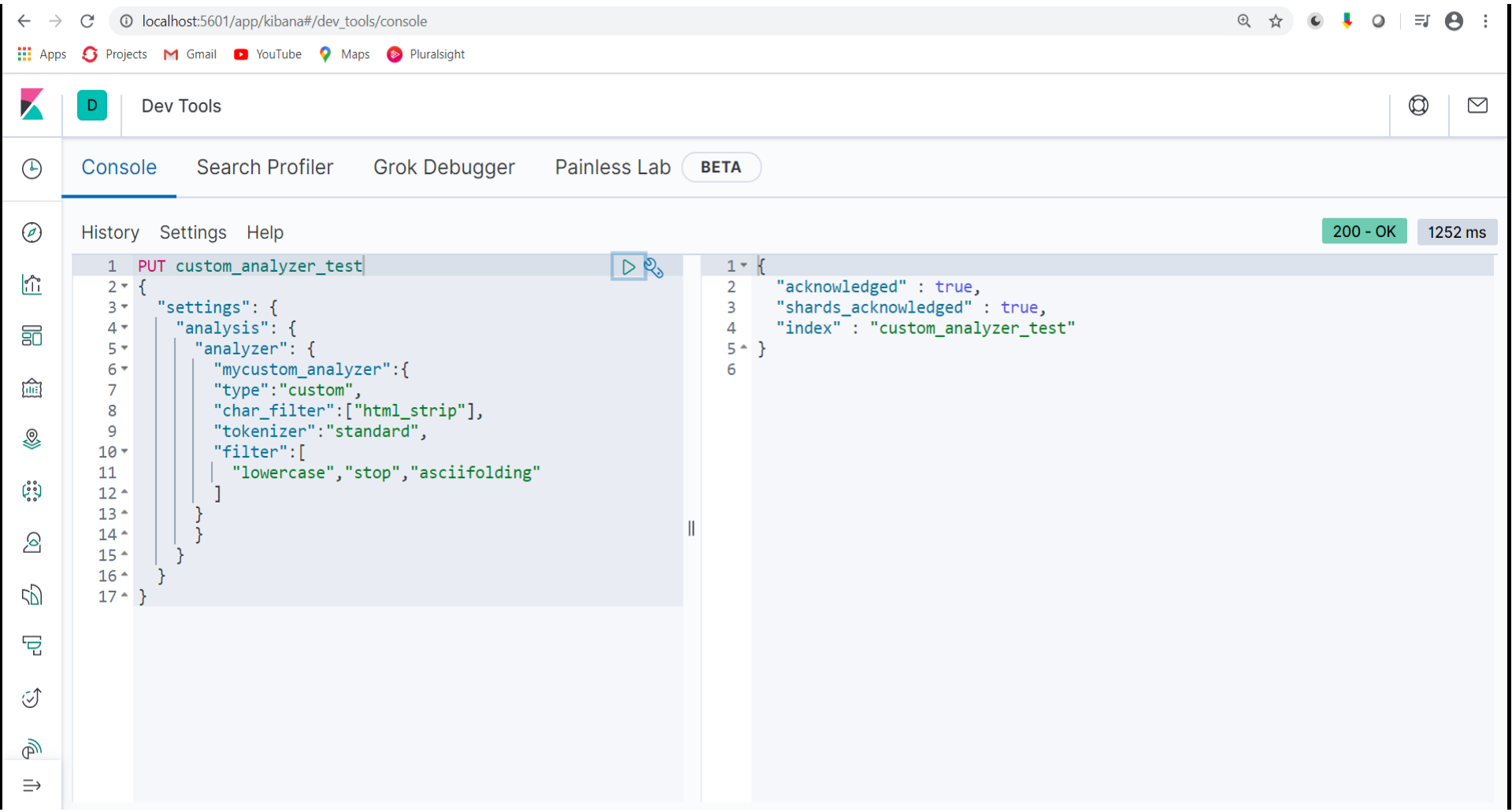



Source Text
The two lazy dogs were slower than the less lazy dog, Rover.
html_strip Char Filter
The two lazy dogs were slower than the less lazy dog, Rover.
standard Tokenizer
The1 two2 lazy3 dogs4 than5 the6 less7 lazy8 dog9 Rover1
lowercase Token Filter
the1 two2 lazy3 dogs4 than5 the6 less7 lazy8 dog9 rover10
stop Token Filter
two2 lazy3 dogs4 than5 less7 lazy8 dog9 rover10
snowball Token Filter
two2 lazi3 dog4 than5 less7 lazi8 dog9 rover10

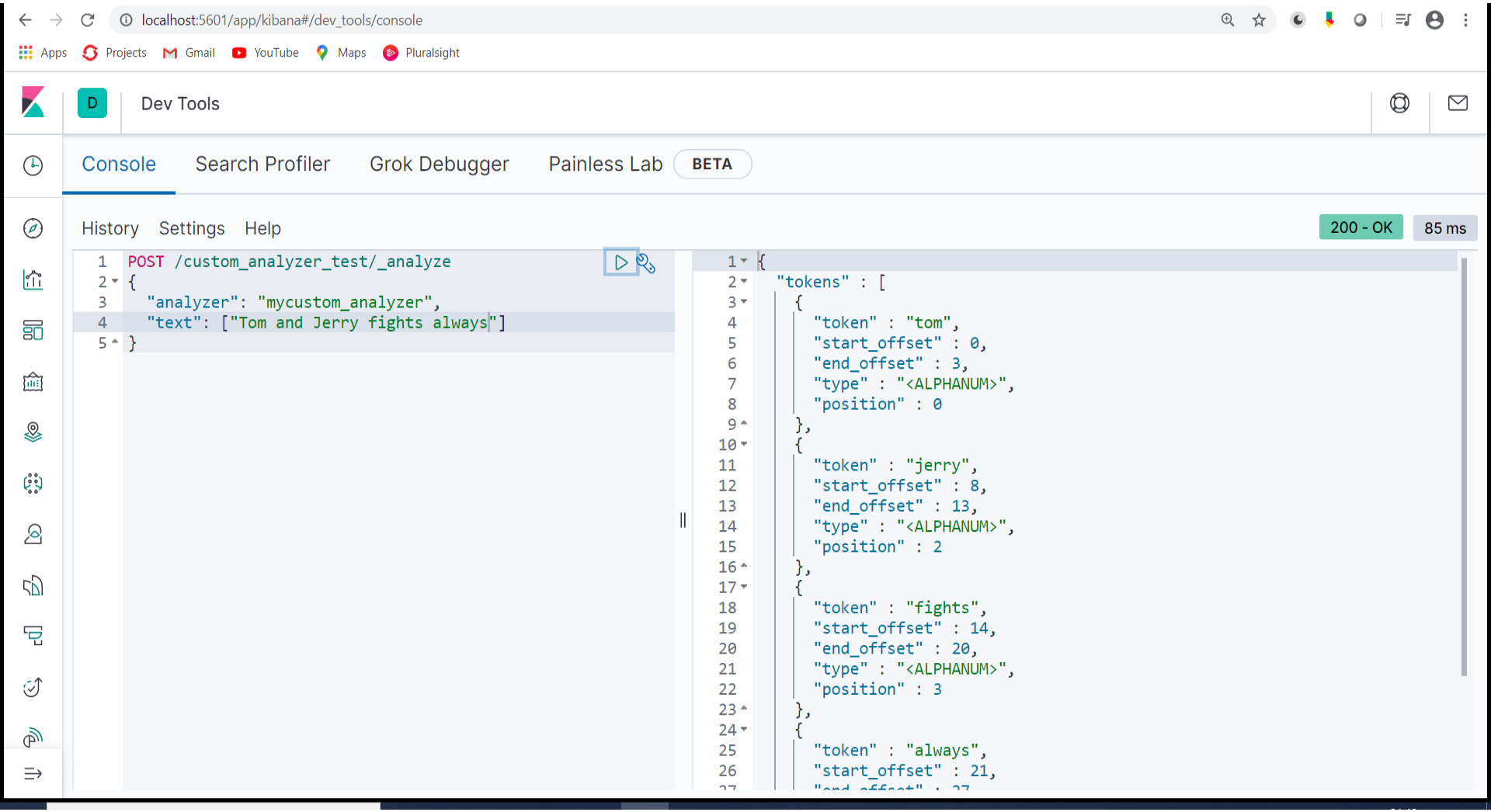
Pattern Replace Tokenizer



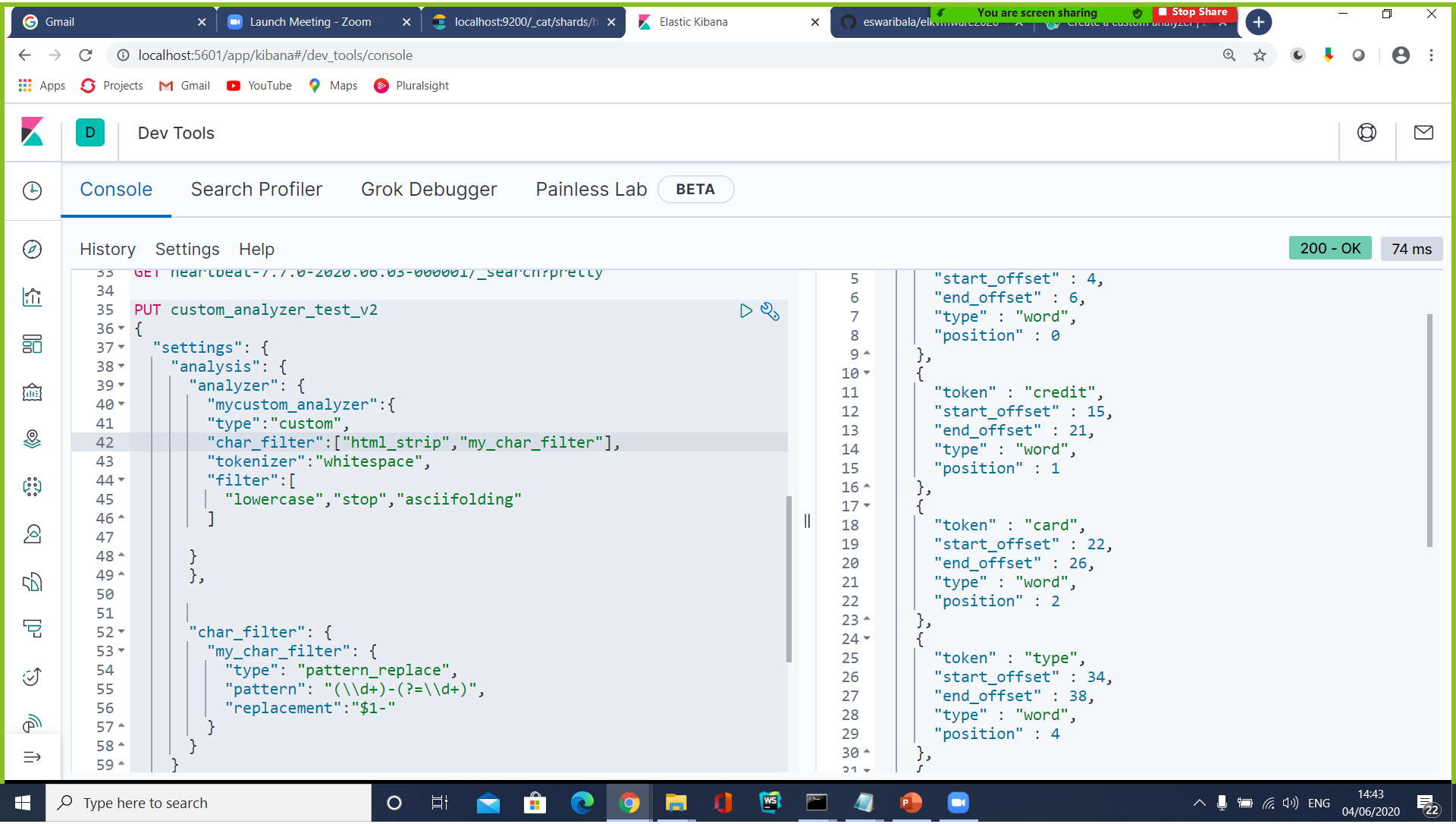
Custom Analyzer



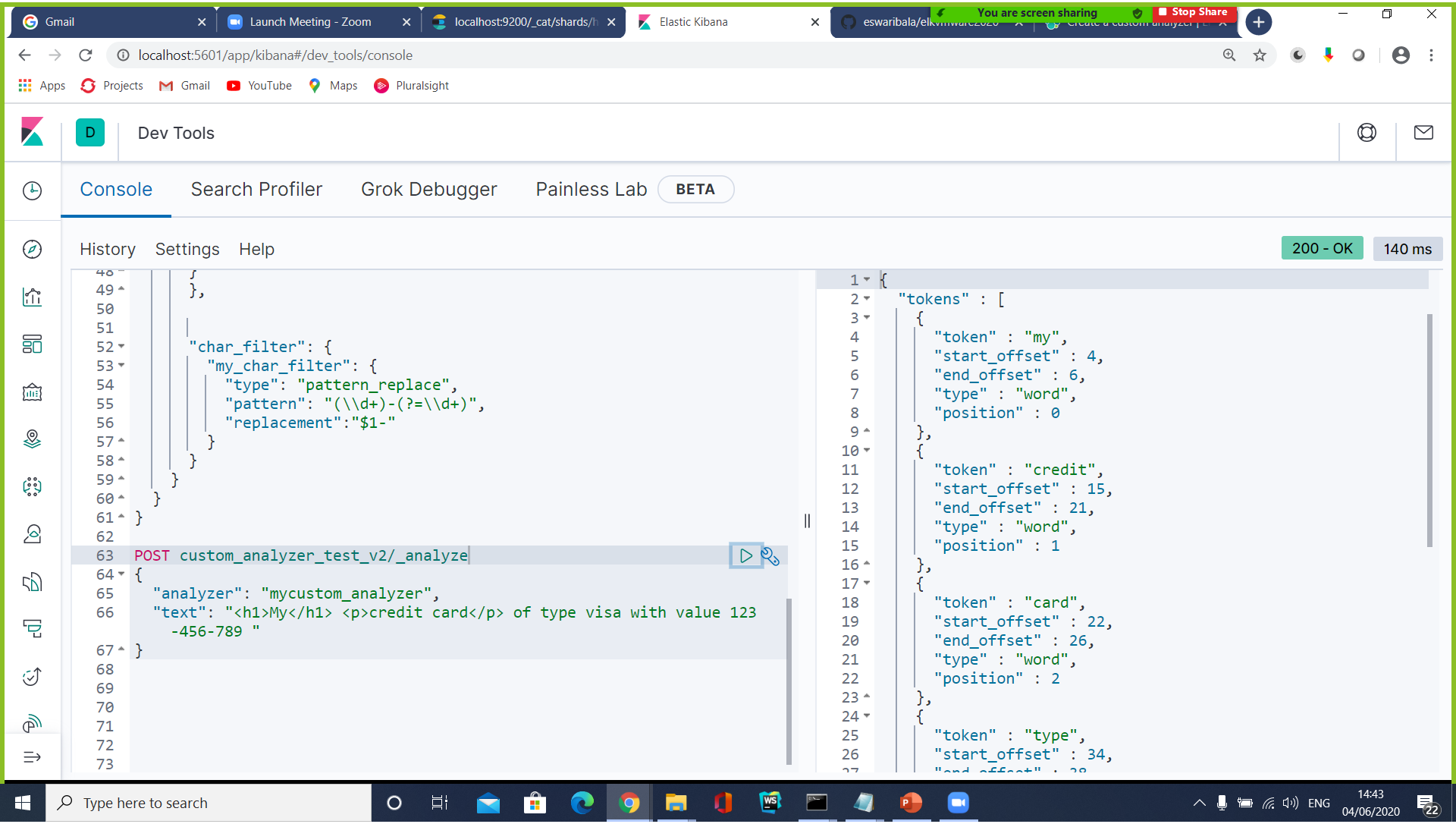
Custom Analyzer



Custom Analyzer



Custom Analyzer



Inverted Index

Term	Document #1	Document #2
best	X	
carbonara		X
delicious		X
pasta	X	X
pesto	X	
recipe	X	X
the	X	
with	X	

Inverted Index

Term	Document #1	Document #2
best	X	
carbonara		X
delicious		X
pasta	X	X
pesto	X	
recipe	X	X
the	X	
with	X	

Inverted Index

Term	Document #1	Document #2
best	X	
carbonara		X
delicious		X
pasta	X	X
pesto	X	
recipe	X	X
the	X	
with	X	

Character Filters – 3 Types

Mapping Character Filter (mapping)

Replaces values based on a map of keys and values.

"I broke my leg _sad_ But luckily I have some red wine around _happy_"

"I broke my leg :-(But luckily I have some red wine around :-)"

Character Filters – 3 Types

Pattern Replace `(pattern_replace)`

Uses a regular expression to match characters and replaces them with the specified replacement. Capture groups may be used.

Pattern: `([a-zA-Z0-9]+)(-?)`

Replacement: `$1`

"9c559866-d0a5-4dee-84fe-54d878fe1800"

"9c559866d0a54dee84fe54d878fe1800"

Tokenizers

1. Word Oriented Tokenizers
2. Partial Word Tokenizers
3. Structured Text Tokenizers

Tokenizers

Word oriented Tokenizers

Typically used for tokenizing full text into individual words.

Tokenizers

Standard Tokenizer `(standard)`

Divides text into terms on word boundaries and removes most symbols. Usually the best choice.

"I'm in the mood for drinking semi-dry red wine!"

[I'm, in, the, mood, for, drinking, semi, dry, red, wine]

Tokenizers

Letter Tokenizer `(letter)`

Divides text into terms when encountering a character that is not a letter.

"I'm in the mood for drinking semi-dry red wine!"

[I, m, in, the, mood, for, drinking, semi, dry, red, wine]

Tokenizers

Lowercase Tokenizer `(lowercase)`

Works like the `letter` tokenizer, but also lowercases all terms.

"I'm in the mood for drinking semi-dry red wine!"

[i, m, in, the, mood, for, drinking, semi, dry, red, wine]

Tokenizers

Whitespace Tokenizer `(whitespace)`

Divides text into terms when encountering whitespace characters.

"I'm in the mood for drinking semi-dry red wine!"

[I'm, in, the, mood, for, drinking, semi-dry, red, wine!]

Tokenizers

UAX URL Email Tokenizer (`uax_url_email`)

Like the `standard` tokenizer, but treats URLs and e-mail addresses as single tokens.

"Contact us at info@codingexplained.com or visit https://codingexplained.com"

[Contact, us, at, info@codingexplained.com, or, visit, https://codingexplained.com]

Tokenizers

Partial Word Tokenizers

Breaks up text or words into small fragments. Used for partial word matching.

Tokenizers

N-Gram Tokenizer (`ngram`)

Breaks text into words when encountering certain characters and then emits N-grams of the specified length.

"Red wine"

[Re, Red, ed, wi, win, wine, in, ine, ne]

Tokenizers

Edge N-Gram Tokenizer (`edge_ngram`)

Breaks text into words when encountering certain characters and then emits N-grams of each word beginning from the start of the word.

"Red wine"

[Re, Red, wi, win, wine]

Tokenizers

Structured Text Tokenizers

Used for structured text such as e-mail addresses, zip codes, identifiers, etc.

Tokenizers

Keyword Tokenizer (`keyword`)

No-op tokenizer which outputs the exact same text as a single term.

"I'm in the mood for drinking semi-dry red wine!"

[I'm in the mood for drinking semi-dry red wine!]

Tokenizers

Pattern Tokenizer `(pattern)`

Uses a regular expression to split text into terms when matching a word separator. Alternatively captures matched text as terms.

"I, like, red, wine!"

[I, like, red, wine!]

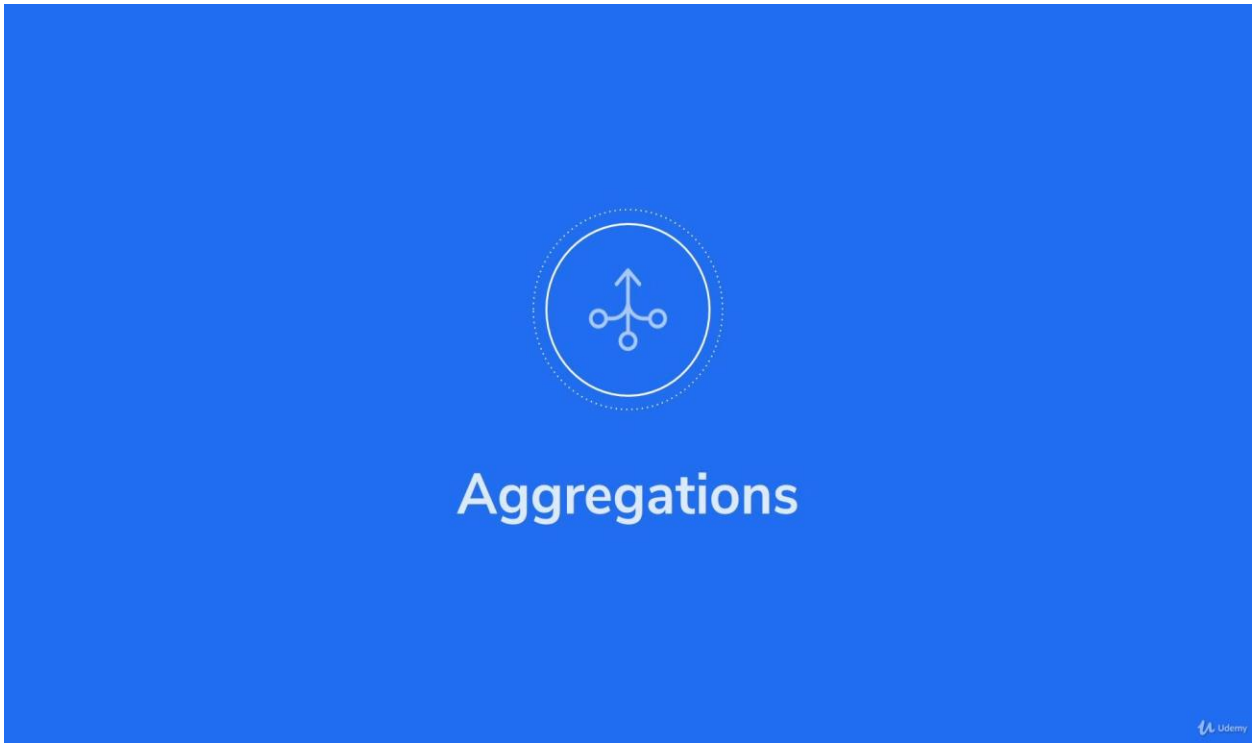
Tokenizers

Path Tokenizer (path_hierarchy)

Splits hierarchical values (e.g. file system paths) and emits a term for each component in the tree.

/path/to/some/directory

[/path, /path/to, /path/to/some, /path/to/some/directory]


Aggregations
Udemy

Aggregations

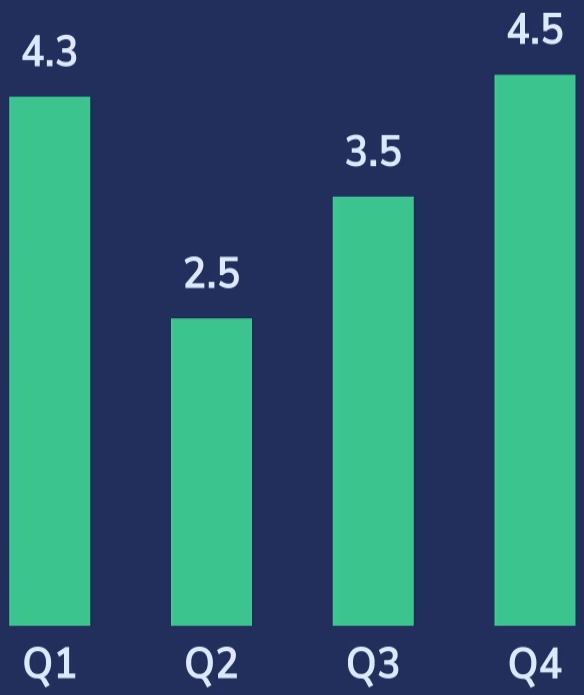
It's Not Just For Search Anymore

Metrics

Average, stats, min/max, percentiles, etc.

Buckets

Histograms, ranges, distances, significant terms, etc.



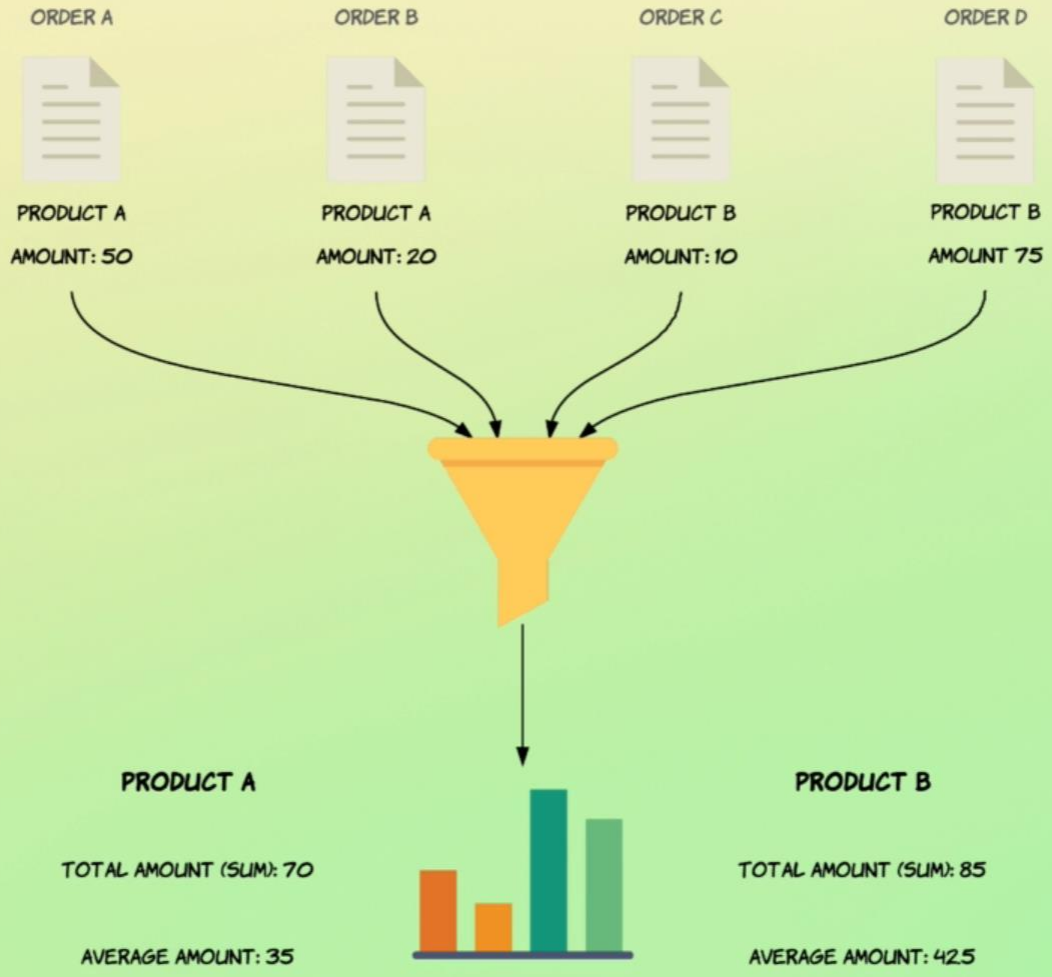
Pipelines

Moving average, average bucket, cumulative sum, etc.

Matrix

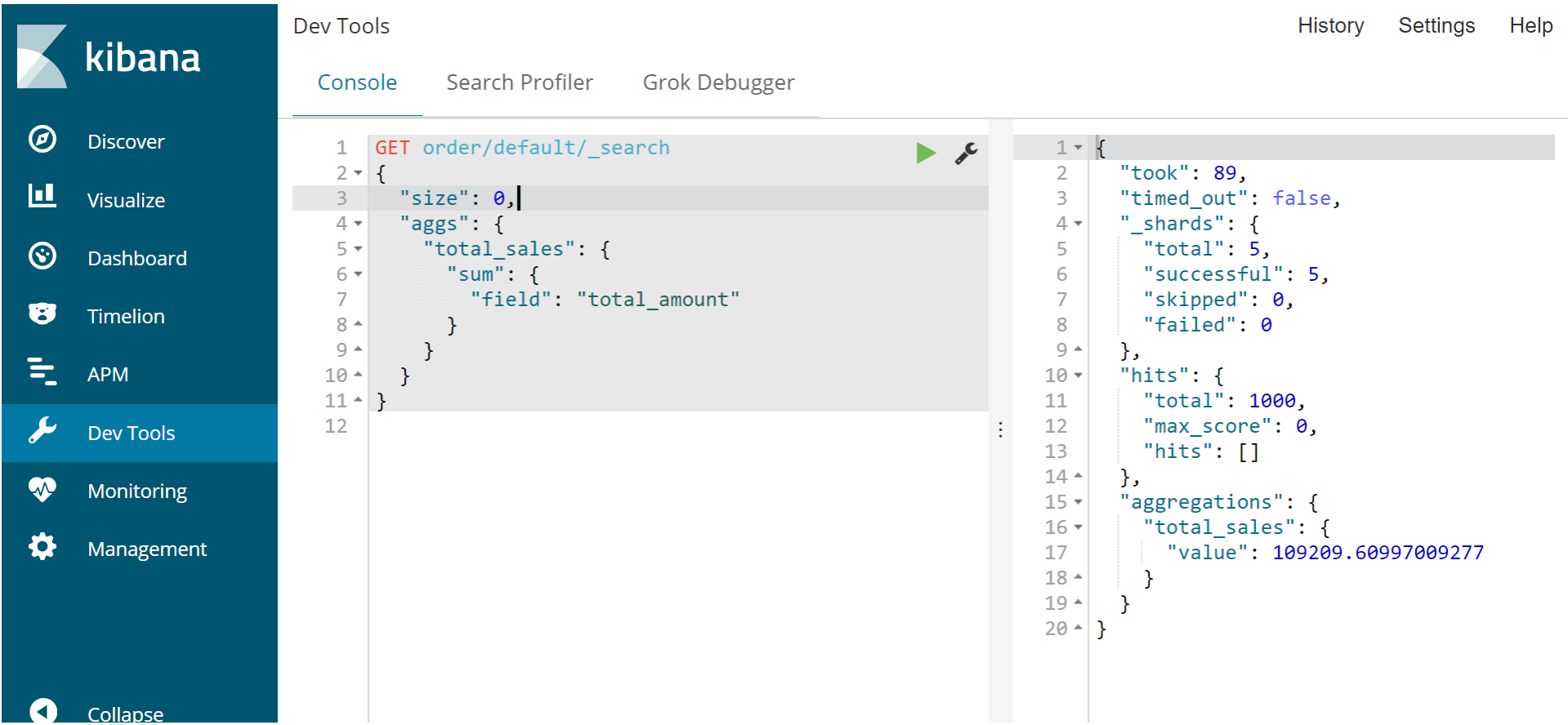
Matrix stats

Aggregations

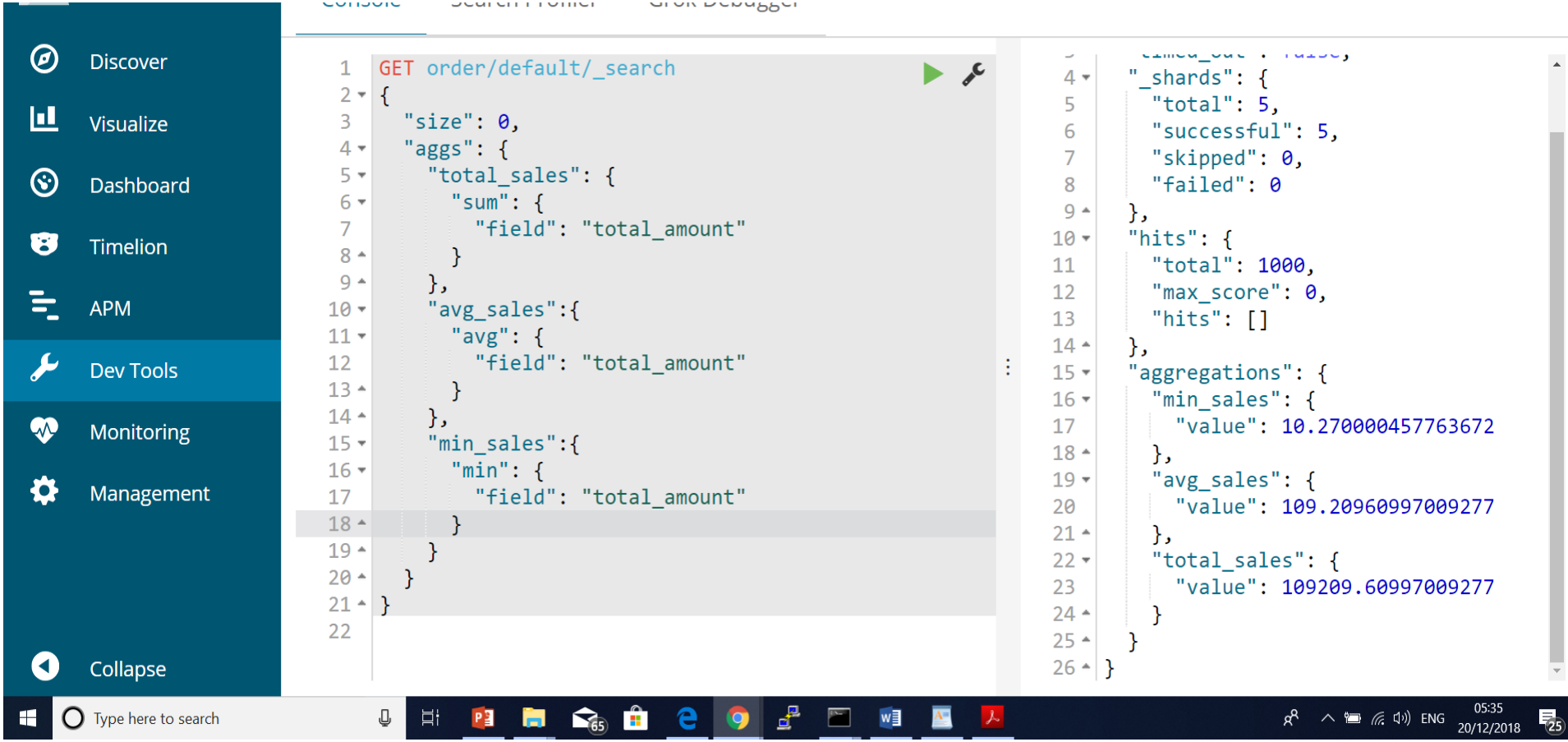


Aggregations

There are many occasions when aggregations are required but search hits are not. For these cases the hits can be ignored by setting size=0.



Aggregations



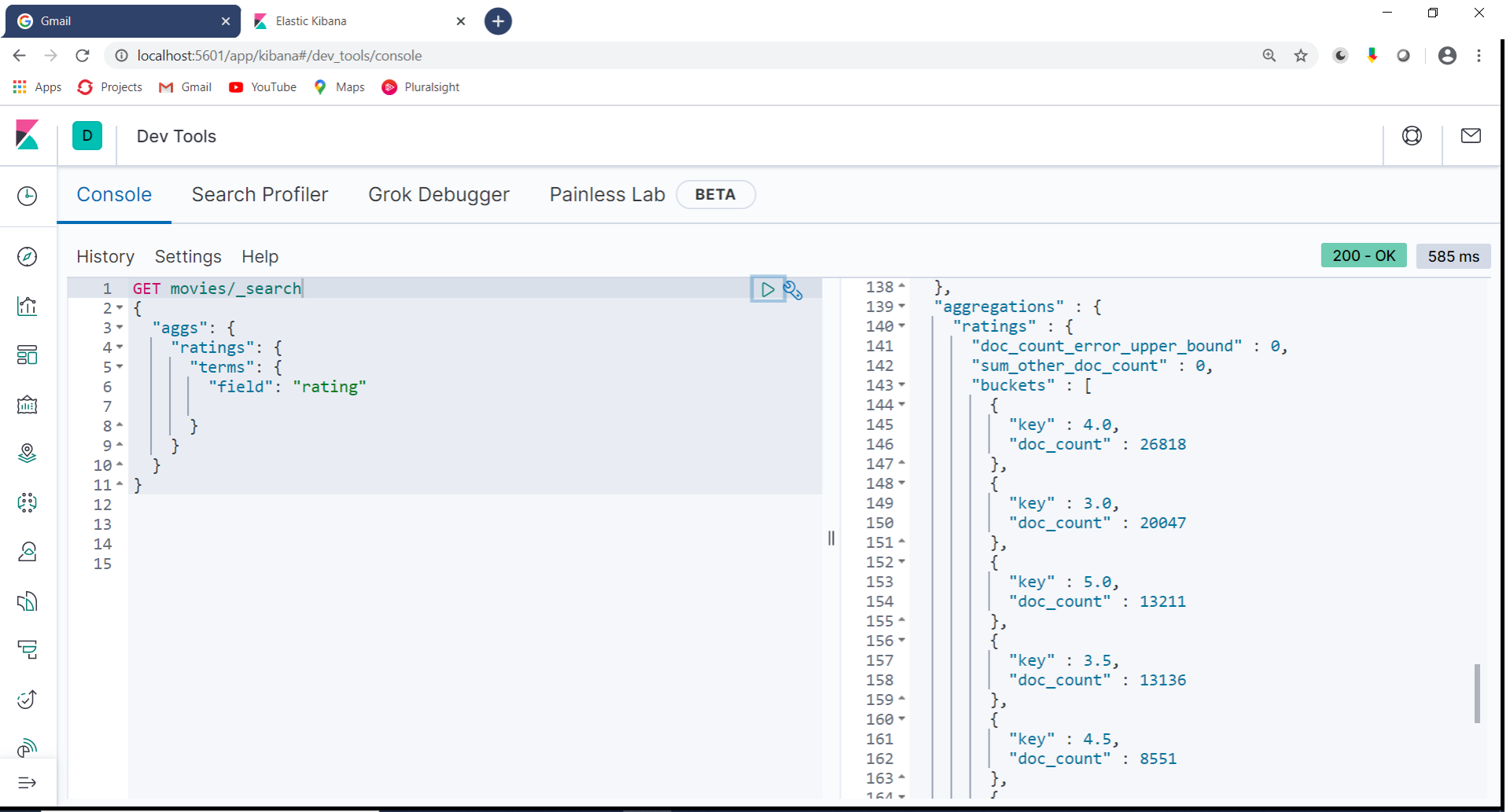
Aggregations

Let's Learn By Example

Bucket by rating value:

```
curl -XGET '127.0.0.1:9200/ratings/_search?size=0&pretty'
-d '
{
    "aggs": {
        "ratings": {
            "terms": {
                "field": "rating"
            }
        }
    }
}'
```

Aggregations



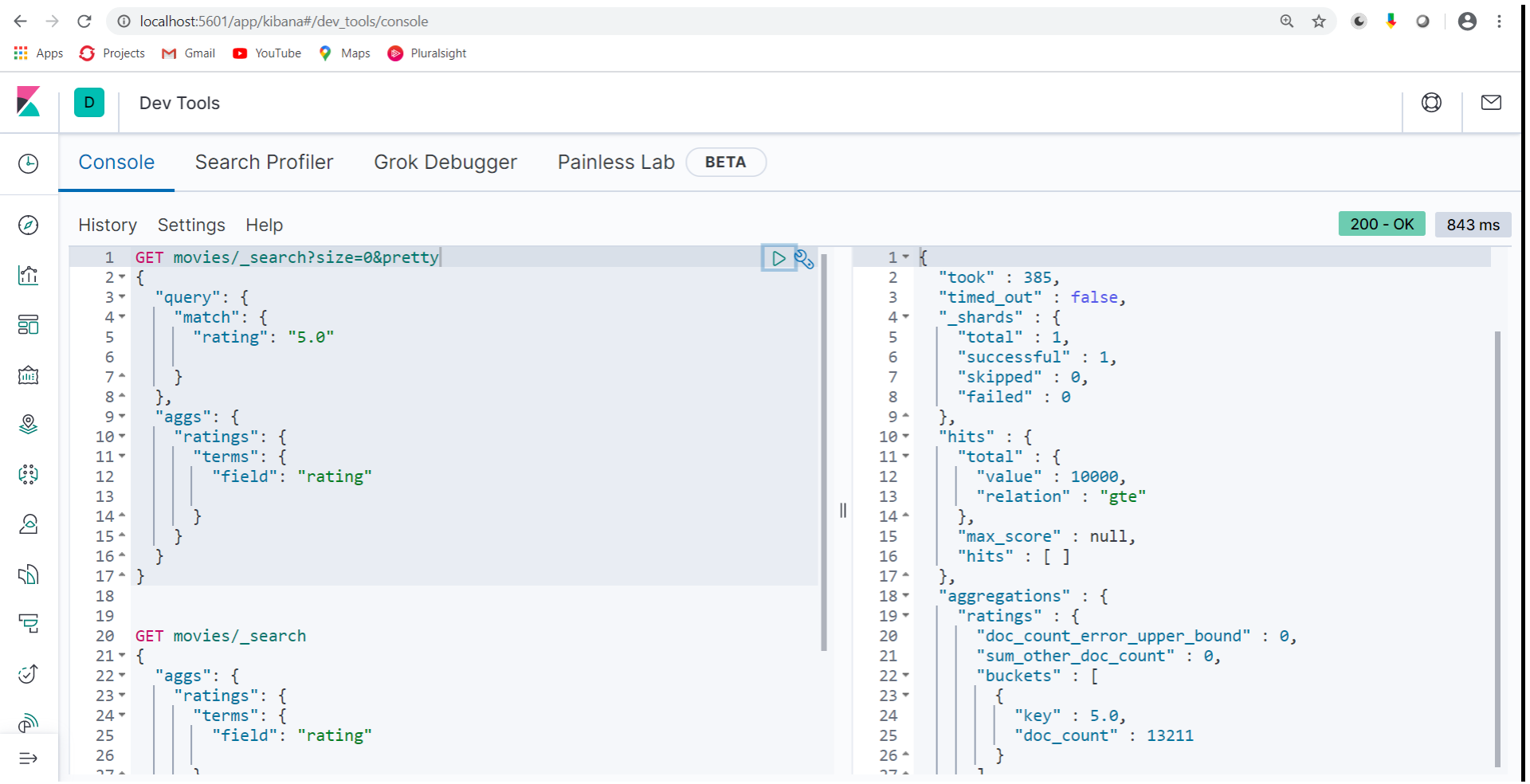
Aggregations

Let's Learn By Example

Count only 5-star ratings:

```
curl -XGET '127.0.0.1:9200/ratings/_search?size=0&pretty' -d
'
{
    "query": {
        "match": {
            "rating": 5.0
        }
    },
    "aggs" : {
        "ratings": {
            "terms": {
                "field" : "rating"
            }
        }
    }
}'
```

Aggregations



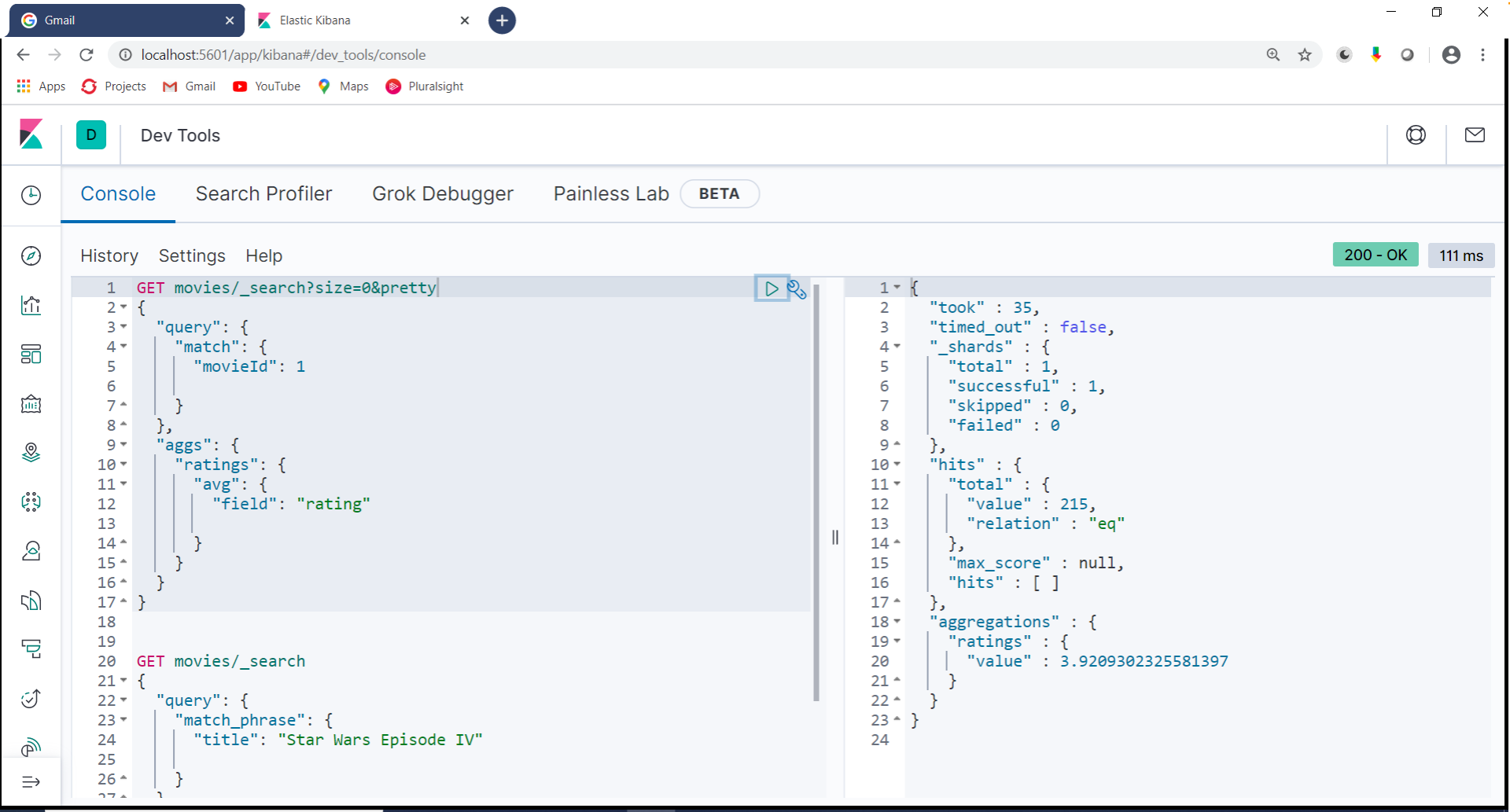
Aggregations

Let’s Learn By Example

Average rating for Star Wars:

```
curl -XGET '127.0.0.1:9200/ratings/_search?size=0&pretty' -d '
{
    "query": {
        "match_phrase": {
            "title": "Star Wars Episode IV"
        }
    },
    "aggs" : {
        "avg_rating": {
            "avg": {
                "field" : "rating"
            }
        }
    }
}'
```

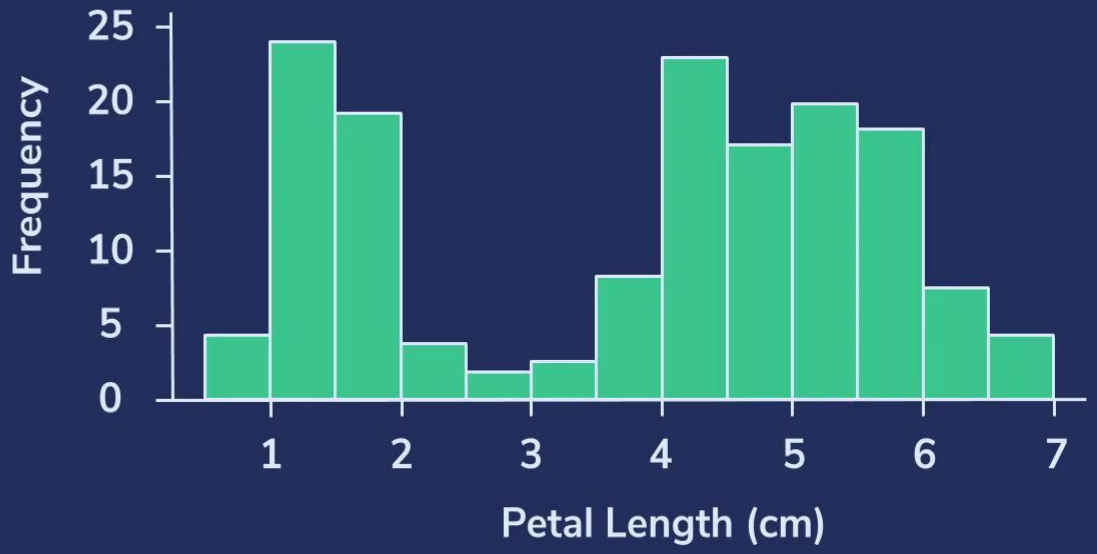
Aggregations



Aggregations

What Is A Histogram

Display totals of documents bucketed by some interval range

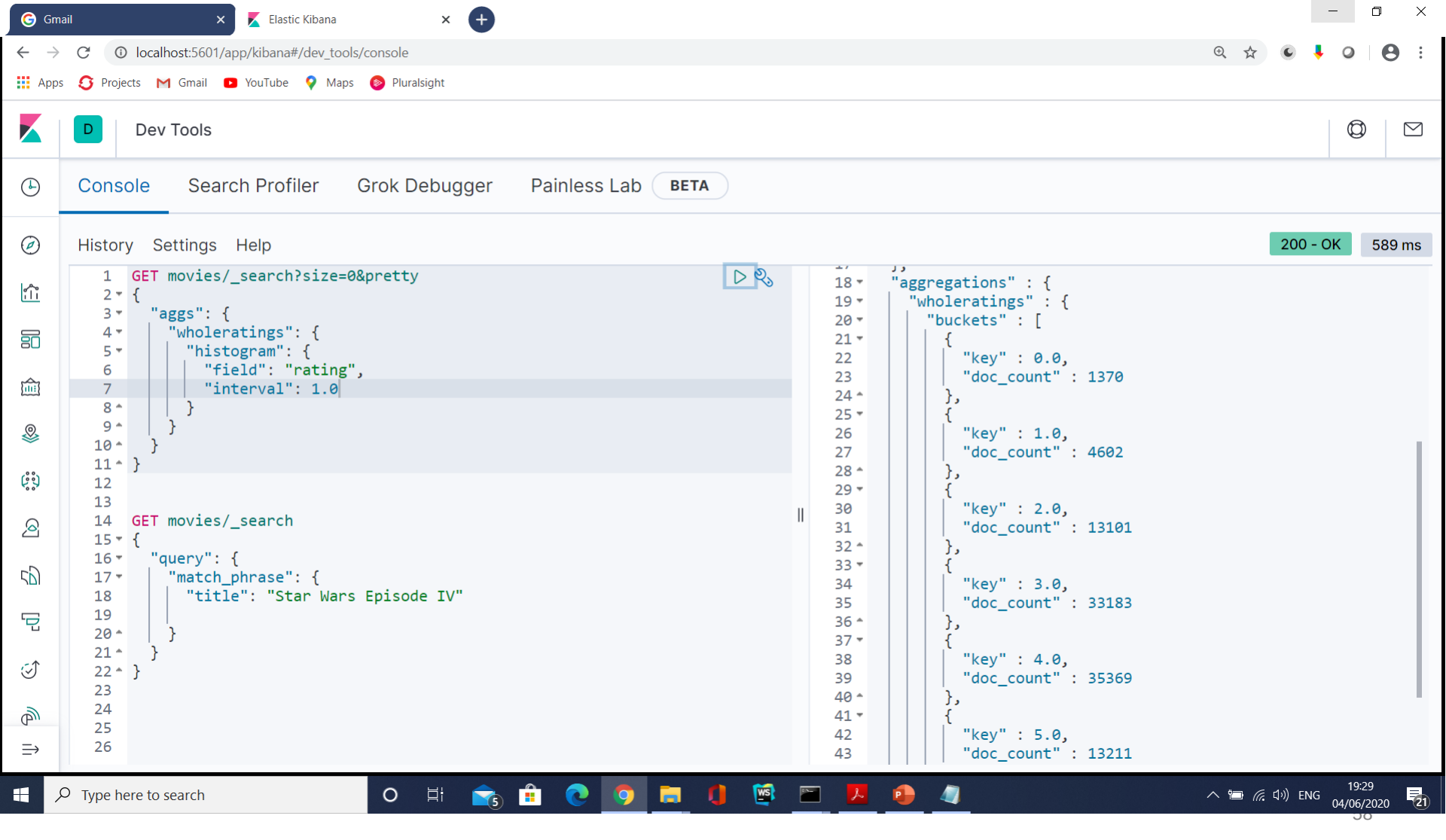


Aggregations

Display Ratings By 1.0-rating Intervals

```
curl -XGET '127.0.0.1:9200/ratings/_search?size=0&pretty' -d '
{
    "aggs" : {
        "whole_ratings": {
            "histogram": {
                "field": "rating",
                "interval": 1.0
            }
        }
    }
}'
```

Aggregations



Suggesters

- Elasticsearch offers three types of suggesters:
  - Term suggesters
  - Phrase suggesters
  - Completion suggesters (autocomplete)

Suggesters

- Term suggester
  - The term suggester can be run at the same time as a query, and can be used to suggest “did you mean …?” alternatives, particularly in the case where a user has misspelled a word.
  - The term suggester, in its simplest form, will search a field for a given set of words.
  - If the suggester finds very similarly spelled words in the index, which are more common than the terms used in the query, then the suggester will suggest replacing these terms with alternatives.

Suggesters

In its simplest form, the term suggester will work without a specific mapping on any “text” field.

```
POST content_programmes_v4/_search
{
  "suggest": {
    "my-suggestion" : {
      "text" : "elasticsearch powered search engyne",
      "term" : {
        "field" : "content.body",
        "suggest_mode" : "missing",

      }
    }
  }
}
```

Suggesters

- Phrase suggester
- The phrase suggester is similar to a term suggester but is more sophisticated.
- It looks at the position of the words in the text and can try to propose an improved phrase which is more likely to give relevant results.
- The phrase suggester requires you to implement a specific analyzer (trigram analyzer) to enable it to find relevant results.

Suggesters

- Completion suggester
- A completion suggester is quite different to the term and phrase suggesters.
- Rather than suggesting an improved query, it provides “search as you type” results.
- The completion suggester is optimized for speed.

Highlighling

- Elastic Search’s highlighting feature gives us the ability to highlight a portion of text in a document that matches the search term.
- Highlighting is very useful and one of these uses is to make it easier to find a term in long texts, which can help the user in his search.

Highlighling Parameters

- fields: Specifies the fields to retrieve highlights for.
- fragmenter: Specifies how text should be broken up in highlight snippets: simple or span.
- fragment_size: The size of the highlighted fragment in characters
- number_of_fragments: The maximum number of fragments to return. If the number of fragments is set to 0, no fragments are returned. Instead, the entire field contents are highlighted and returned.

Highlighling Parameters

- pre_tags: Use in conjunction with post_tags to define the HTML tags to use for the highlighted text.
- post_tags: Use in conjunction with pre_tags to define the HTML tags to use for the highlighted text.
- require_field_match: By default, only fields that contain a query match are highlighted.

Highlighling Parameters

```
GET idx_movies/_search
{
  "_source": ["title", "description"],
  "query": {
    "multi_match": {
      "query": "Aliens and predator",
      "fields": [
        "title",
        "description"
      ]
    }
  },
  "highlight": {
    "number_of_fragments": 5,
    "fragment_size": 100,
    "require_field_match": "true",
    "fields": [
      {
        "title": {
          "number_of_fragments":0
        }
      },
      {
        "description": {
          "fragment_size": 50,
          "number_of_fragments":4}
      }
    ]
  }
}
```

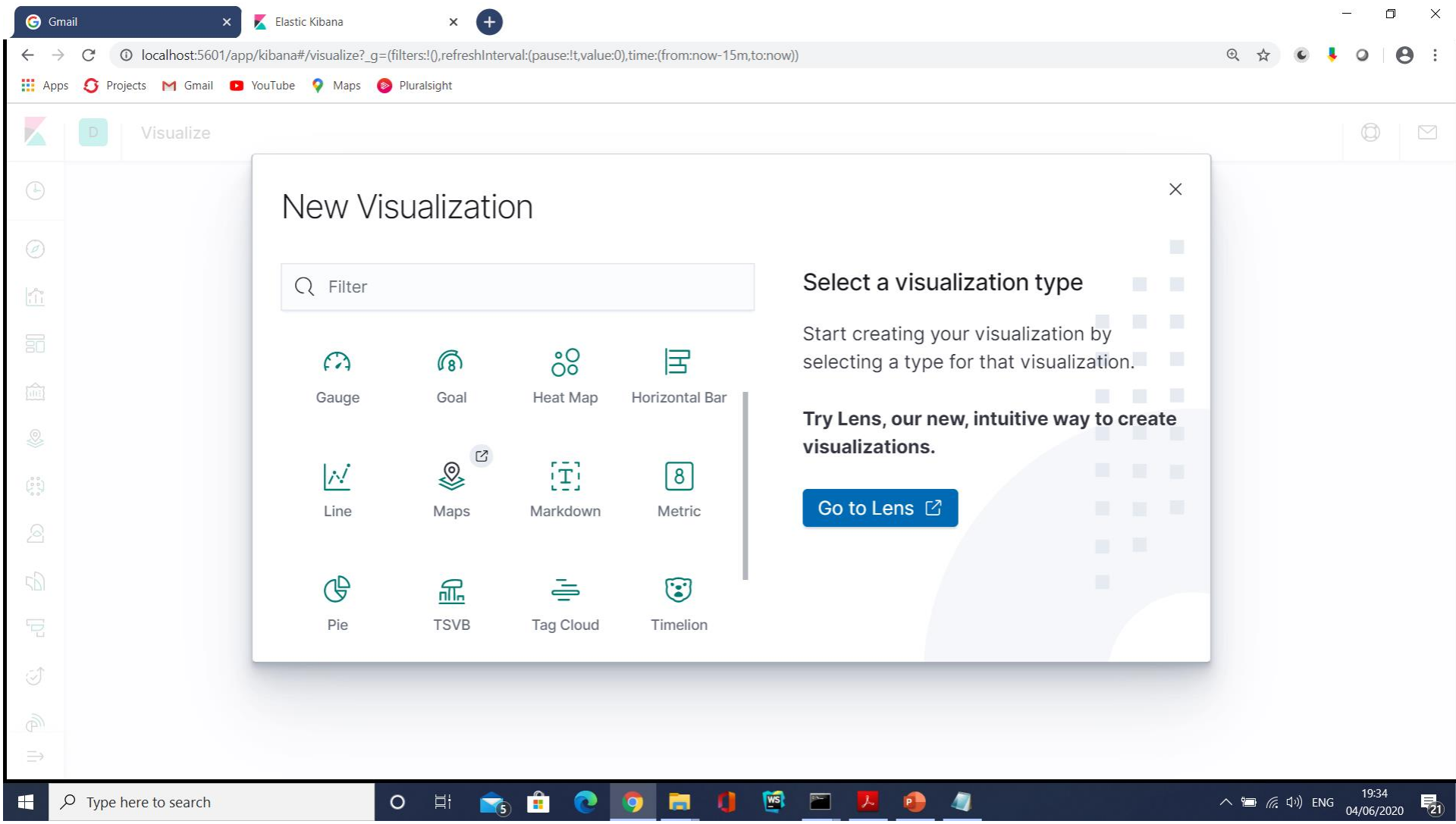
Highlighter types

Highlighter	Method of obtaining offsets
`unified`	Term vectors if `term_vector` is set to `with_positions_offsets`, postings if `index_options` is set to `offsets`, text reanalysis otherwise.
`fvh`	Term vectors.
`plain`	Text reanalysis.

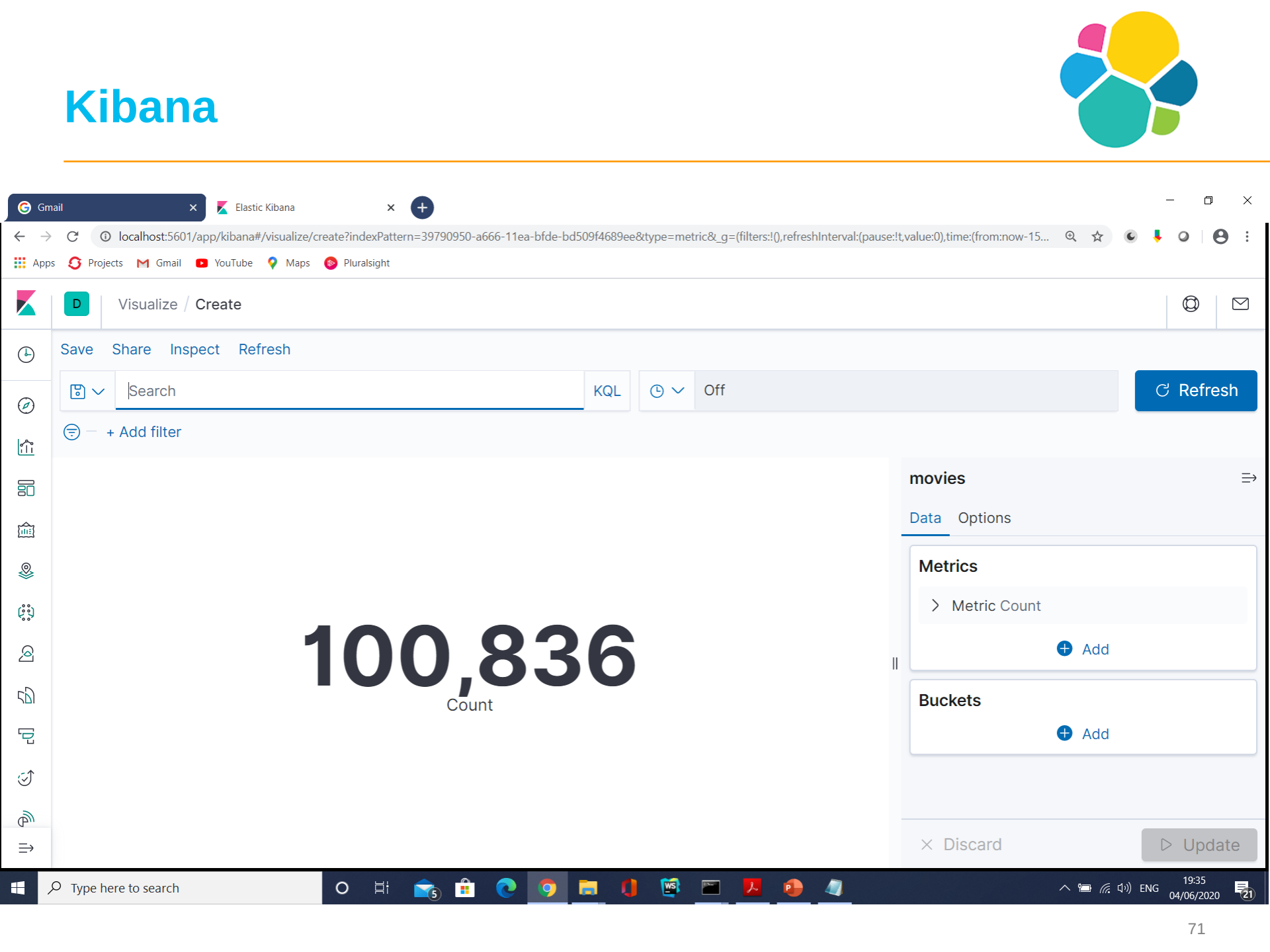
Fast Vector Highlighter

- The fvh highlighter is based on the Lucene Fast Vector Highlighter.
- To use this highlighter, we need to store term vectors with positions offsets, which increases the index size.
- The fvh highlighter can combine matched terms from multiple fields into one result.
- It can also assign weights to matches depending on their positions.
- We can sort phrase matches above term matches when highlighting a query that boosts phrase matches over term matches.
- Additionally, we can configure the fvh highlighter to select the boundaries of a returned text fragment.
- We can highlight multiple words with different tags.

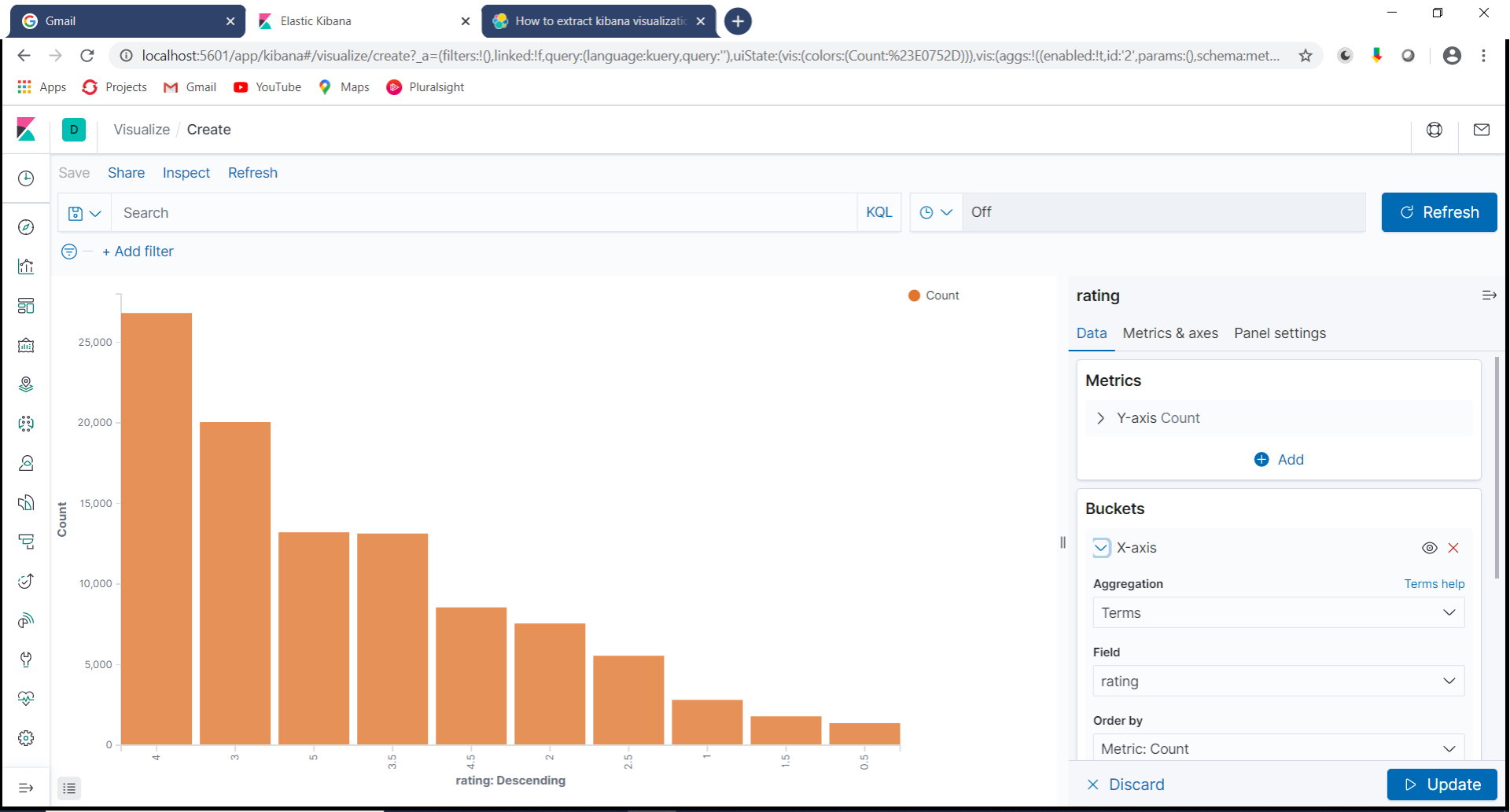
Kibana


Gmail
Elastic Kibana
localhost:5601/app/kibana#/visualize?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,to:now))
Apps
Projects
Gmail
YouTube
Maps
Pluralsight
Visualize
New Visualization
Filter
Gauge
Goal
Heat Map
Horizontal Bar
Line
Maps
Markdown
Metric
Pie
TSVB
Tag Cloud
Timelion
Select a visualization type
Start creating your visualization by selecting a type for that visualization.
Try Lens, our new, intuitive way to create visualizations.
Go to Lens
Type here to search
ENG
19:34
04/06/2020

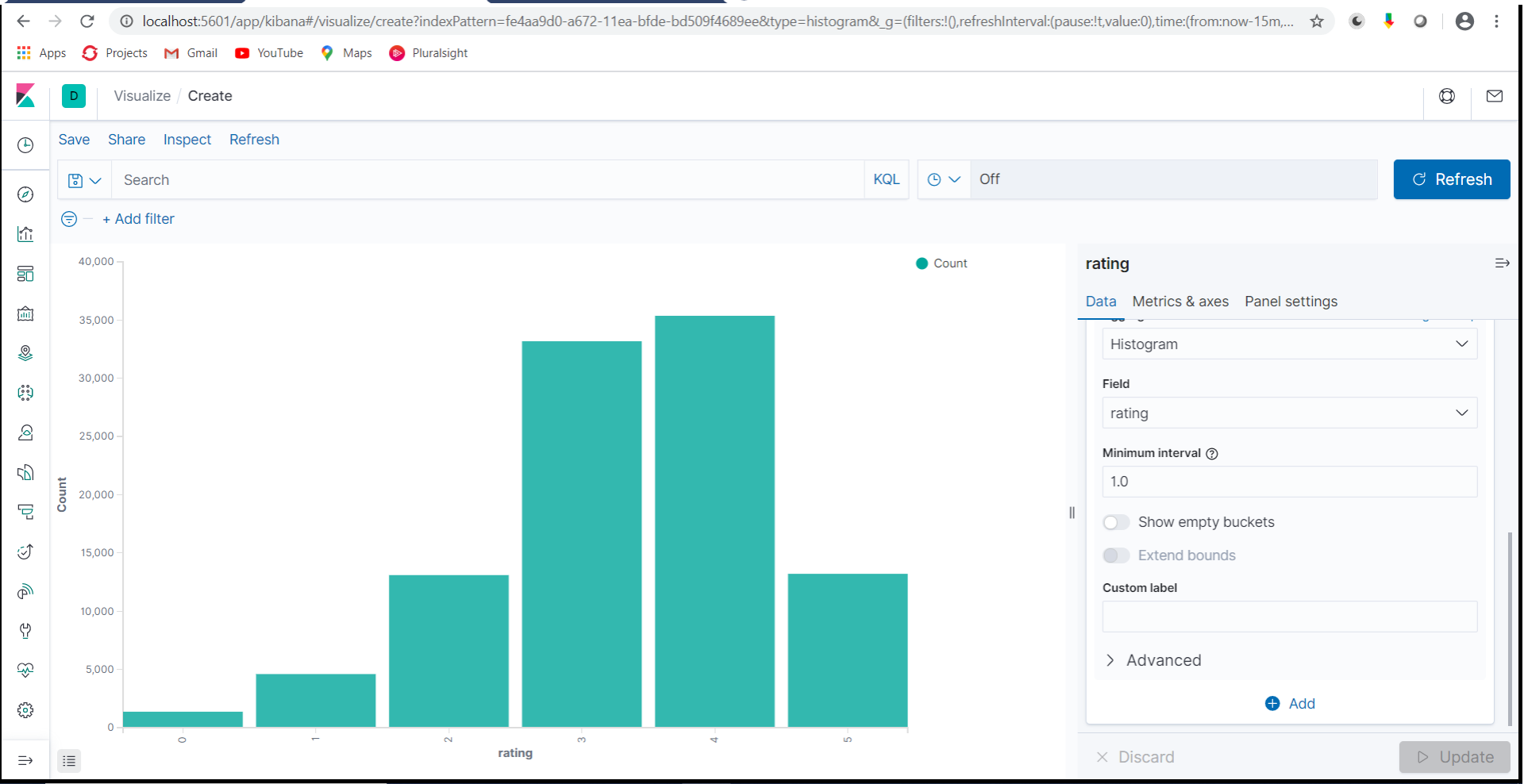
Kibana


Gmail
Elastic Kibana
localhost:5601/app/kibana#/visualize/create?indexPattern=39790950-a666-11ea-bfde-bd509f4689ee&type=metric&_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15...
Apps
Projects
Gmail
YouTube
Maps
Pluralsight
D
Visualize / Create
Save
Share
Inspect
Refresh
Search
KQL
Off
Refresh
+ Add filter
100,836
Count
movies
Data
Options
Metrics
Metric Count
Add
Buckets
Add
Discard
Update
Type here to search
ENG
19:35
04/06/2020

Kibana


Gmail
Elastic Kibana
How to extract kibana visualizatic
localhost:5601/app/kibana#/visualize/create?_a=(filters:!(),linked:!f,query:(language:kuery,query:''),uiState:(vis:(colors:(Count:%23E0752D))),vis:(aggs:!((enabled:!t,id:'2',params:(),schema:met...
Apps
Projects
Gmail
YouTube
Maps
Pluralsight
Visualize / Create
Save
Share
Inspect
Refresh
Search
KQL
Off
Refresh
+ Add filter
Count
25,000
20,000
15,000
10,000
5,000
0
Count
4
3
5
3.5
4.5
2
2.5
1
1.5
0.5
rating: Descending
rating
Data
Metrics & axes
Panel settings
Metrics
Y-axis Count
Add
Buckets
X-axis
Aggregation
Terms help
Terms
Field
rating
Order by
Metric: Count
Discard
Update
Type here to search
ENG
20:33
04/06/2020

Kibana


localhost:5601/app/kibana#/visualize/create?indexPattern=fe4aa9d0-a672-11ea-bfde-bd509f4689ee&type=histogram&_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,...
Apps
Projects
Gmail
YouTube
Maps
Pluralsight
Visualize / Create
Save
Share
Inspect
Refresh
Search
KQL
Off
Refresh
+ Add filter
Count
40,000
35,000
30,000
25,000
20,000
15,000
10,000
5,000
0
Count
0
1
2
3
4
5
rating
rating
Data
Metrics & axes
Panel settings
Histogram
Field
rating
Minimum interval
1.0
Show empty buckets
Extend bounds
Custom label
Advanced
Add
Discard
Update
Type here to search
ENG
20:45
04/06/2020

Dealing With Time

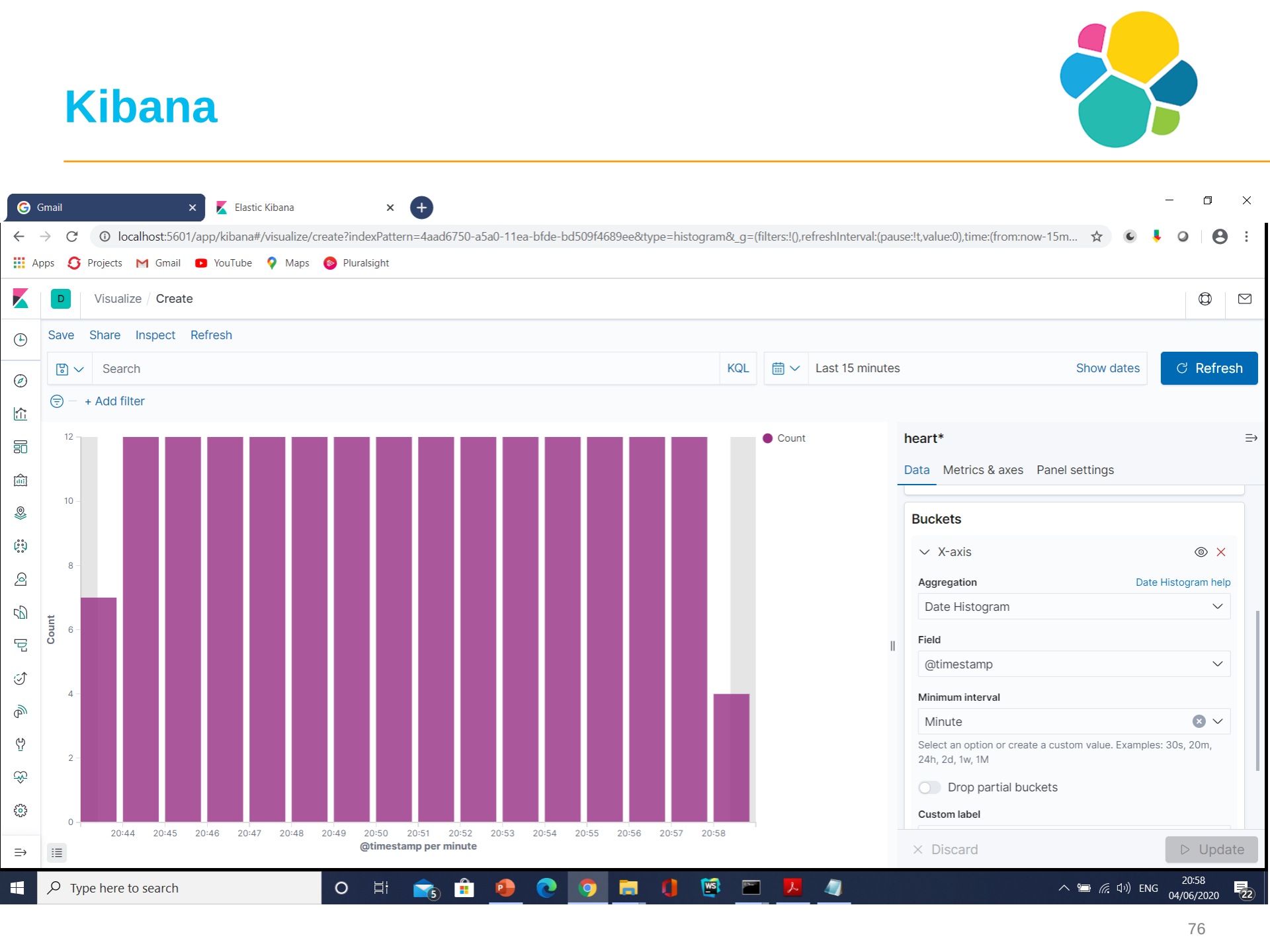
Elasticsearch can bucket and aggregate fields that contain time and dates properly. You can aggregate by “year” or “month” and it knows about calendar rules.

Kibana

Break Down Website Hits By Hour:

```
curl -XGET '127.0.0.1:9200/kafka-logs/_search?size=0&pretty' -d '
{
    "aggs" : {
        "timestamp": {
            "date_histogram": {
                "field": "@timestamp",
                "interval": "hour"
            }
        }
    }
}'
```

Kibana


Gmail
Elastic Kibana
localhost:5601/app/kibana#/visualize/create?indexPattern=4aad6750-a5a0-11ea-bfde-bd509f4689ee&type=histogram&_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-15m...
Apps
Projects
Gmail
YouTube
Maps
Pluralsight
Visualize / Create
Save
Share
Inspect
Refresh
Search
KQL
Last 15 minutes
Show dates
Refresh
+ Add filter
Count
@timestamp per minute
heart*
Data
Metrics & axes
Panel settings
Buckets
X-axis
Aggregation
Date Histogram help
Date Histogram
Field
@timestamp
Minimum interval
Minute
Select an option or create a custom value. Examples: 30s, 20m, 24h, 2d, 1w, 1M
Drop partial buckets
Custom label
Discard
Update
Type here to search
ENG
20:58
04/06/2020

Kibana

When Does Google Scrape Me?

```
curl -XGET
'127.0.0.1:9200/kafka-logs/_search?size=0&pretty' -d ‘
{
    "query" : {
        "match": {
            "agent": "Googlebot"
        }
    },
    "aggs" : {
        “timestamp": {
            "date_histogram": {
                "field": "@timestamp",
                "interval": "hour"
            }
        }
    }
}'
```

Installing Elasticsearch Plugins

- Regardless of what functionalities they add, Elasticsearch plugins belong to either of the following two categories: core plugins or community plugins.
- The former is supplied as part of the Elasticsearch package and are maintained by the Elastic team.
- The latter is developed by the community and are thus separate entities with their own versioning and development cycles.

Installing Elasticsearch Plugins

- Installing core plugins is simple.
- In the example below, I'm going to install the X-Pack plugin.
- X-Pack extends Elasticsearch by plugging in Shield, Watcher, and Marvel — three plugins that prior to Elasticsearch 5.x were separate entities, each requiring separate installation and setup.

Installing Elasticsearch Plugins

- Plugins must be installed on every node in the cluster, and each node must be restarted after installation.

- Community plugins are a bit different as each of them has different installation instructions.

Elastic Snapshot and Restore

- A snapshot is a backup of a running Elasticsearch cluster.
- You can use snapshots to:
  - Regularly back up a cluster with no downtime
  - Recover data after deletion or a hardware failure
  - Transfer data between clusters
  - Reduce your storage costs by using searchable snapshots in the cold and frozen data tiers

Snapshot Workflow

- Elasticsearch stores snapshots in an off-cluster storage location called a snapshot repository.
- Before you can take or restore snapshots, you must register a snapshot repository on the cluster.
- Elasticsearch supports several repository types with cloud storage options, including:
  - AWS S3
  - Google Cloud Storage (GCS)
  - Microsoft Azure
- After you register a snapshot repository, you can use snapshot lifecycle management (SLM) to automatically take and manage snapshots.
- You can then restore a snapshot to recover or transfer its data.

Snapshots Contents

- By default, a snapshot of a cluster contains the cluster state, all regular data streams, and all regular indices. The cluster state includes:
  - Persistent cluster settings
  - Index templates
  - Legacy index templates
  - Ingest pipelines
  - ILM policies

Snapshots Contents

- You can also take snapshots of only specific data streams or indices in the cluster.
- A snapshot that includes a data stream or index automatically includes its aliases.
- When you restore a snapshot, you can choose whether to restore these aliases.
- Snapshots don’t contain or back up:
  - Transient cluster settings
  - Registered snapshot repositories
  - Node configuration files
  - Security configuration files

Create Snapshot

```
PUT /_snapshot/backup_repo
{
  "type": "fs",
  "settings": {
    "location": "/home/root/backups",
    "compress": true
  }
}
```

Create Snapshot

- curl -XPUT "http://localhost:9200/_snapshot/backup_repo" -H 'Content-Type: application/json' -d'{ "type": "fs", "settings": {   "location": "/home/root/backups", "compress": true }}'

How to View the Snapshot Repository

- To confirm the successful creation of the snapshot repository, use the GET request with the _snapshot endpoint as:
- GET /_snapshot/backup_repo
- You can also use the following cURL command:
- curl -XGET "http://localhost:9200/_snapshot/backup_repo"

How to Create an Elasticsearch Snapshot

- Creating an Elasticsearch snapshot for a specific snapshot repository is handled by the create snapshot API.
- The API requires the snapshot repository name and the name of the snapshot.
- NOTE: A single snapshot repository can have more than one snapshot of the same clusters as long as they have unique identities/names.
- Consider the following request to add a snapshot called snapshot_2021 to the backup_repo repository.
  - PUT /_snapshot/backup_repo/snapshot_2021
- To use cURL, use the command:
  - curl -XPUT "http://localhost:9200/_snapshot/backup_repo/snapshot_2021"

How to Create an Elasticsearch Snapshot

- Consider the following request that backups the .kibana index (a system index) and specifies which user authorized the snapshot and the reason.

- PUT /_snapshot/backup_repo/snapshot_2
- {
- "indices": ".kibana",
- "ignore_unavailable": true,
- "include_global_state": true,
- "metadata": {
- "taken_by": "elasticadmin",
- “taken_because”: “Daily Backup”
- }
- }

How to View Snapshots

- All you need to pass in the request is the snapshot repository and the name of the snapshot you wish to view the details.
- The snapshot should respond with details about a specified snapshot. These details include:
  - Start and end time values
  - The version of Elasticsearch that created the snapshot
  - List of included indices
  - The snapshot's current state
  - List of failures that occurred during the snapshot

How to View Snapshots

- GET /_snapshot/backup_repo/snapshot_3

- To use cURL, use the command below:

- [cc lang="text" width="100%" height="100%" escaped="true" theme="blackboard" nowrap="0"]
- curl -XGET “http://localhost:9200/_snapshot/backup_repo/snapshot_3”

Restore Snapshot

- To restore the whole snapshot:

- POST /_snapshot/my_backup/snapshot-01-11-2019/_restore

- To restore an individual index:

- POST /_snapshot/my_backup/snapshot-01-11-2019/_restore
- {
- "indices": "my_index"
- }

Improve Performance

- Size parameter
- Assigning a huge value to size parameter causes Elasticsearch to compute vast amounts of hits, which causes severe performance issues. Instead of setting a huge size, you should batch requests in small sizes.

- Shards and replicas
- Optimize necessary index settings that play a crucial role in Elasticsearch performance, like the number of shards and replicas. In many cases having more replicas helps improve search performance.

Improve Performance

- Deleted documents
- Having many deleted documents in the Elasticsearch index also causes search performance issues.
- Force merge API can be used to remove many deleted documents and optimize the shards.
- Search filters
- Effective use of filters in Elasticsearch queries can improve search performance dramatically as the filter clauses are 1) cached, and 2) able to reduce the target documents to be searched in the query clause.

Improve Performance

- Wildcard queries
  - Avoid wildcard, especially leading wildcard queries, which causes the entire Elasticsearch index to be scanned.
  - Regex and parent-child
  - Note that Regex queries and parent-child can cause search latency.
- Implementing features
  - There are multiple ways to implement a specific feature in Elasticsearch.
  - For example, Autocomplete can be implemented in various styles.
  - Opster's blog gives a 360-degree view of both functional and non-functional features (especially performance).

Improve Performance

- Multitude of small shards
  - Having many small shards could cause a lot of network calls and threads, which severely impact search performance;
- Heavy aggregations
  - Avoid heavy aggregations that involve unique IDs. Refer to Opster's slow logs guide to identify such search slow logs effectively.
- Timeout and terminate
  - Timeout param and terminate after param can be useful when executing heavy searches, or when result data is vast.

Improve Performance

- Search templates
  - Use search templates to achieve better abstraction, meaning without exposing your query syntax to your users.
  - Search templates also help you transfer less data over the network, which is particularly useful when you have large Elasticsearch queries.
- Multi search API
  - Use msearch whenever possible. In most of the applications it’s required to query multiple Elasticsearch indices for a single transaction, and sometimes users do so in a serial order even when it’s not required.
  - In both cases, when you need to query multiple indices for the same transaction and when the result of these queries are independent, you should always use msearch to execute the queries in parallel in Elasticsearch.

Improve Performance

- Term queries
  - Use term query when you need an exact match and on keywords fields.
  - By default, Elasticsearch generates both text and keyword fields for every field that consists of a string value if explicit mapping is not supplied.
  - Users tend to use the match query even on keyword data types like product-ids, which is costly as match query goes through an analysis operation.

Improve Performance

- Source filtering
  - _source filtering is a great way to improve the performance of Elasticsearch queries when retrieving many documents or documents of large sizes.
  - By default, Elasticsearch returns the complete source of matching documents.
  - If you don’t need _source at all or need only values of specific fields, you can achieve this with _source filtering.

Mixing exact search with stemming

- When building a search application, stemming is often a must as it is desirable for a query on skiing to match documents that contain ski or skis.
- Example stemming.txt

Consistent Scoring

- The fact that Elasticsearch operates with shards and replicas adds challenges when it comes to having good scoring.
- Say the same user runs the same request twice in a row and documents do not come back in the same order both times.
- Unfortunately, this is something that can happen if you have replicas (index.number_of_replicas is greater than 0).
- The reason is that Elasticsearch selects the shards that the query should go to in a round-robin fashion.
- So, it is quite likely if you run the same query twice in a row that it will go to different copies of the same shard.

Consistent Scoring

- Index statistics are an important part of the score.
- And these index statistics may be different across copies of the same shard due to deleted documents.
- When documents are deleted or updated, the old document is not immediately removed from the index.
- It is just marked as deleted and it will only be removed from disk on the next time that the segment this old document belongs to is merged.
- Those deleted documents are taken into account for index statistics.
- So, imagine that the primary shard just finished a large merge that removed lots of deleted documents, then it might have index statistics that are sufficiently different from the replica (which still have plenty of deleted documents) so that scores are different too.

Consistent Scoring

- The recommended way to work around this issue is to use a string that identifies the user that is logged in (a user id or session id for instance) as a preference.
- This ensures that all queries of a given user are always going to hit the same shards, so scores remain more consistent across queries.
- This work around has another benefit: when two documents have the same score, they will be sorted by their internal Lucene doc id (which is unrelated to the _id) by default.
- However, these doc ids could be different across copies of the same shard.
- So, by always hitting the same shard, we would get more consistent ordering of documents that have the same scores.

Tune Indexing Speed

- Use bulk requests
- Bulk requests will yield much better performance than single-document index requests.
- In order to know the optimal size of a bulk request, you should run a benchmark on a single node with a single shard.
- First try to index 100 documents at once, then 200, then 400, etc. doubling the number of documents in a bulk request in every benchmark run.

Tune Indexing Speed

- Use bulk requests
- When the indexing speed starts to plateau then you know you reached the optimal size of a bulk request for your data.
- In case of tie, it is better to err in the direction of too few rather than too many documents.

Tune Indexing Speed

- Use multiple workers/threads to send data to Elasticsearchedit
- A single thread sending bulk requests is unlikely to be able to max out the indexing capacity of an Elasticsearch cluster.
- In order to use all resources of the cluster, you should send data from multiple threads or processes.
- In addition to making better use of the resources of the cluster, this should help reduce the cost of each fsync.

Tune Indexing Speed

- Unset or increase the refresh interval
- The operation that consists of making changes visible to search - called a refresh - is costly, and calling it often while there is ongoing indexing activity can hurt indexing speed.
- By default, Elasticsearch periodically refreshes indices every second, but only on indices that have received one search request or more in the last 30 seconds.

Tune Indexing Speed

- Disable replicas for initial loads
- If you have a large amount of data that you want to load all at once into Elasticsearch, it may be beneficial to set index.number_of_replicas to 0 in order to speed up indexing.
- Having no replicas means that losing a single node may incur data loss, so it is important that the data lives elsewhere so that this initial load can be retried in case of an issue.
- Once the initial load is finished, you can set index.number_of_replicas back to its original value.

Tune Indexing Speed

- Disable swapping
- You should make sure that the operating system is not swapping out the java process by disabling

Tune Indexing Speed

- Give memory to the filesystem cacheedit
- The filesystem cache will be used in order to buffer I/O operations.
- You should make sure to give at least half the memory of the machine running Elasticsearch to the filesystem cache.

Tune Indexing Speed

- Use auto-generated ids
- When indexing a document that has an explicit id, Elasticsearch needs to check whether a document with the same id already exists within the same shard, which is a costly operation and gets even more costly as the index grows.
- By using auto-generated ids, Elasticsearch can skip this check, which makes indexing faster.

Tune Indexing Speed

- Use faster hardware
- If indexing is I/O-bound, consider increasing the size of the filesystem cache (see above) or using faster storage.
- Elasticsearch generally creates individual files with sequential writes.
- However, indexing involves writing multiple files concurrently, and a mix of random and sequential reads too, so SSD drives tend to perform better than spinning disks.

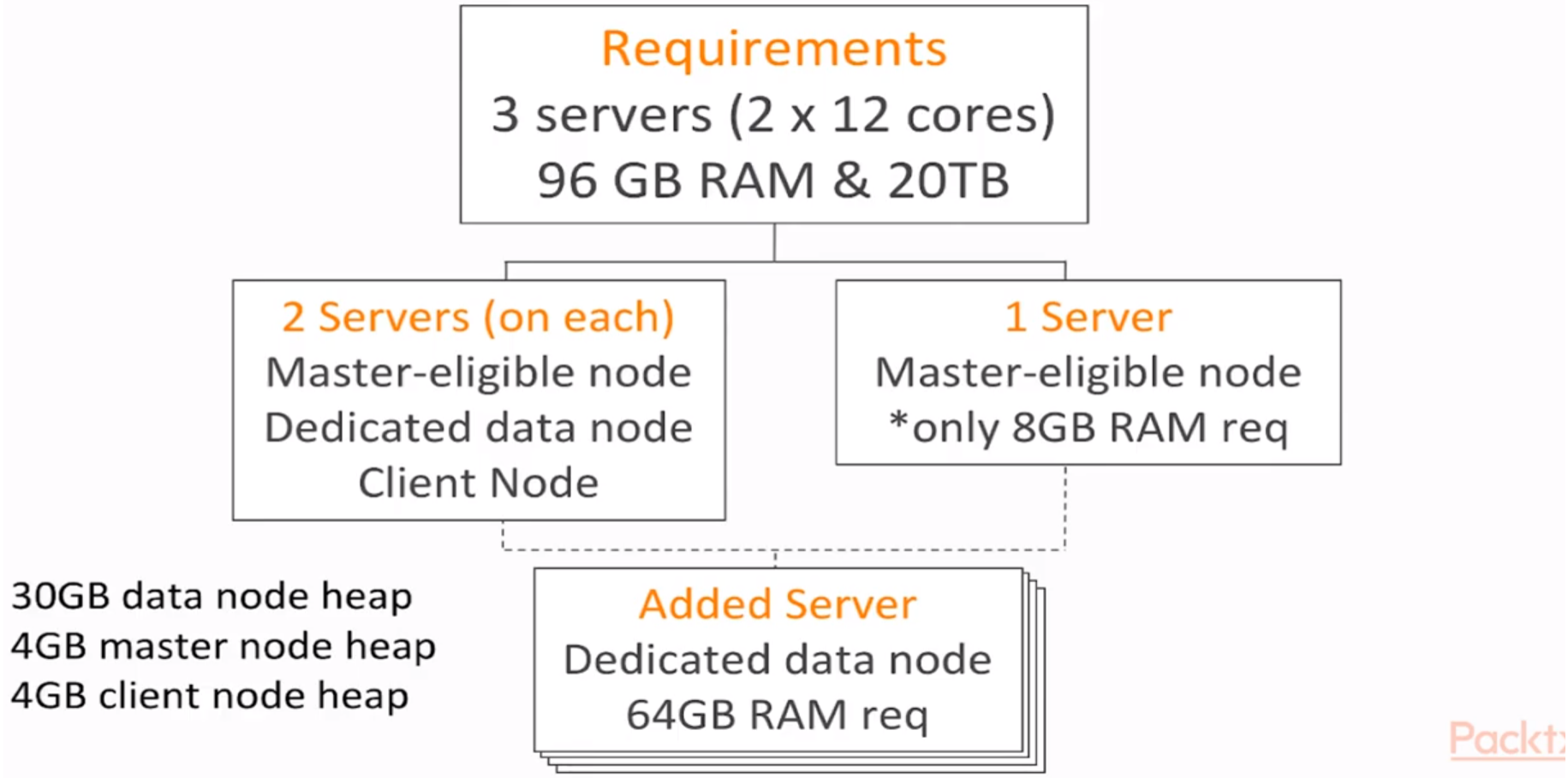
Tune Indexing Speed

- Directly-attached (local) storage generally performs better than remote storage because it is simpler to configure well and avoids communications overheads.
- With careful tuning it is sometimes possible to achieve acceptable performance using remote storage too.
- Benchmark your system with a realistic workload to determine the effects of any tuning parameters.
- If you cannot achieve the performance you expect, work with the vendor of your storage system to identify the problem.

Tune Indexing Speed

- Indexing buffer size
- If your node is doing only heavy indexing, be sure indices.memory.index_buffer_size is large enough to give at most 512 MB indexing buffer per shard doing heavy indexing (beyond that indexing performance does not typically improve).
- Elasticsearch takes that setting (a percentage of the java heap or an absolute byte-size) and uses it as a shared buffer across all active shards.
- Very active shards will naturally use this buffer more than shards that are performing lightweight indexing.
- The default is 10% which is often plenty: for example, if you give the JVM 10GB of memory, it will give 1GB to the index buffer, which is enough to host two shards that are heavily indexing.

Production Requirements



Split API on Index

- The split index API allows you to split an existing index into a new index, where each original primary shard is split into two or more primary shards in the new index.
- The number of times the index can be split (and the number of shards that each original shard can be split into) is determined by the index.number_of_routing_shards setting.
- The number of routing shards specifies the hashing space that is used internally to distribute documents across shards with consistent hashing.
- For instance, a 5 shard index with number_of_routing_shards set to 30 (5 x 2 x 3) could be split by a factor of 2 or 3.
- In other words, it could be split as follows:
- 5 → 10 → 30 (split by 2, then by 3)
- 5 → 15 → 30 (split by 3, then by 2)
- 5 → 30 (split by 6)

How Splitting Works

- A split operation:
- Creates a new target index with the same definition as the source index, but with a larger number of primary shards.
- Hard-links segments from the source index into the target index.
- Hashes all documents again, after low level files are created, to delete documents that belong to a different shard.
- Recovers the target index as though it were a closed index which had just been re-opened.

How Splitting Works

- PUT /dest1
- {
- "settings": {
- "index": {
- "number_of_routing_shards": 24,
- "number_of_shards":2
- }
- }
- }

Production Requirements

- number_of_routing_shards
  - The number of routing shards.
  - This parameter defines the number of times the original index can be split or the numbers of primary shards that can be obtained after the split.
  - When we create an index, we must make sure that the number of primary shards configured for the index is a factor of the value of this parameter.

Production Requirements

```
POST /dest1/_doc/_bulk
{"index":{}}
{"productName":"Product A","annual_rate":"3.2200%","describe":"A product that allows you to select whether to receive push messages for returns."}
{"index":{}}
{"productName":"Product B","annual_rate":"3.1100%","describe":"A product that daily pushes messages for returns credited to your account."}
{"index":{}}
{"productName":"Product C","annual_rate":"3.3500%","describe":"A product that daily pushes messages for returns immediately credited to your account."}
```

Production Requirements

Disable data write operations for the index.

```
PUT /dest1/_settings
{
  "settings": {
    "index.blocks.write": true
  }
}
```

Production Requirements

- Split the original index into a new index with more primary shards and enable data write operations for the new index.

```
POST dest1/_split/dest3
{
  "settings": {
    "index.number_of_shards": 12,
    "index.blocks.write": null
  }
}
```

Production Requirements

- Avoid split brain:
  - In a cluster of multiple master eligible nodes, we always have a concern that if a network becomes partitioned or unstable then the cluster would accidently elect more than one master, which is referred to as the “split brain” scenario.
  - So, to avoid such a scenario we need at least minimum master nodes votes to win an election for master node.
  - To determine the minimum number of master eligible nodes for this election, we use the N/2 + 1 formula where N is the number of master eligible nodes.
  - In case of production clusters of 3 master eligible nodes, just add the below in your elasticsearch.yml file.
  - “discovery.zen.minimum_master_nodes: 2”

Production Requirements

- Infrastructure cost reduction:
  - Pay less and get the maximum benefits.
  - While handling massive amounts of data, we worry about the huge infrastructure costs.
  - Elasticsearch provides Index Lifecycle Management (ILM) policies to automatically manage indices.
  - We can shift our data to the compute, storage or memory optimized systems based upon our performance, resiliency, and retention requirements.
  - We can use Hot-warm-cold architecture in Elasticsearch which can be implemented using Kibana GUI.

Production Requirements

- JVM heap size settings:
  - As Elasticsearch and Lucene are written in Java, we need to adjust the maximum heap space and JVM stats.
  - It is important to note that the more heap available to Elasticsearch, the more memory it can use for filtering, caching and other processes to increase query performance.
  - Also, too much heap space can lead to a large garbage collection.
  - By default, Elasticsearch automatically sets the JVM heap size based on a node's role and total memory.
  - Using a default sizing is recommended for most production environments.

Production Requirements

- JVM heap size settings:
  - In case you are overriding the default heap size then note that JVM should be set up to 50% of your RAM, but no more than 32GB (due to Java pointer inefficiency in larger heaps) and both the minimum and maximum heap size settings must be identical.
  - These values can be configured using the Xmx and Xms settings in the jvm.options file.

Production Requirements

- Disable swapping:
  - Operating systems try to use as much memory as possible for file system caches and eagerly swap out unused application memory.
  - Elasticsearch performance can drastically suffer when the OS decides so, as it can even swap out Elasticsearch executable pages out of the disk.
  - Disabling OS level swapping and enabling memory lock can help us to avoid such scenarios.
  - Just add the below in your elasticsearch.yml file.
  - Set bootstrap.memory_lock: true

Production Requirements

- Virtual memory adjustment:
  - Elasticsearch uses mmaps directory by default to store its indices.
  - The default operating system limits on mmap counts and is likely to be too low, which may result in out of memory exceptions.
  - So, to avoid running out of virtual memory, increase the number of limits on mmap counts in /etc/sysctl.conf file.
  - Set vm.max_map_count=262144
  - On DEB/RPM, this setting is configured automatically. No further configuration is required.

Production Requirements

- Increase Open file descriptor limit:
  - Elasticsearch uses a lot of file descriptors or file handlers.
  - Running out of file descriptors can be disastrous and will most probably lead to loss of data.
  - Make sure to increase the limit on the number of open files descriptors for the user running Elasticsearch to 65,536 or higher.
  - On DEB/RPM the default settings are already configured to suit this requirement, but we can fine tune it in the /etc/security/limits.conf file.
  - Just set nofile to 65536.

Production Requirements

- Disable wildcard:
  - As the data deleted from Elasticsearch cluster cannot be retrieved so to ensure that someone does not issue a DELETE operation on all indexes (* or _all) disable deleting all the indices by wildcard query.
  - Set action.destructive_requires_name to true

Production Requirements

- Cluster sizing:
  - Since Elasticsearch is a resource intensive setup, there are multiple factors which we should consider before setting up our infrastructure sizing for elastic clusters.
  - Like the size of data, frequency of data ingestion, retention requirement, number of active users, HA requirement for Kibana or Logstash, number of replicas required, data searching and reporting requirements and many more.

Production Requirements

- Cluster sizing:
  - For our ease, we can categorize this into two factors: volume and throughput requirements.
  - Volume for estimating the storage and memory resources required to store the expected amount of data and shards for each tier of the cluster.
  - Throughput for estimating the memory, compute, and network resources required to process the expected operations for each tier of the cluster.
  - There is a simple math related to Elasticsearch volume sizing.
  - Example actual space requirement for 100 GB of raw log data for the execution is 100GB * Json conversion factor (1.2) * compression (0.7) * replica set factor

Production Requirements

- Cluster sizing:
  - For throughput in case of standard log processing then we may require less CPU like 8 to 16 cores with 8x memory factor.
  - And if there is a need for more analytics computation in the form of DSL queries then we may need more CPU cores.
  - Local SSD drives such as NVMe or Index based disk with high IOPS should be used rather than remote file systems such as NFS or SMB.
  - Ultimately, the goal is to prevent the thread pool cues from growing faster than they are consumed.
  - With insufficient computing resources, there is always a risk of search requests getting dropped.

Production Requirements

- Cluster sizing:
  - To get the adequate amount of memory requirement, we should observe our cluster memory usage from the beginning.
  - We can scale in and out our cluster based upon our actual need.

Production Requirements

- Field mapping:
- When we ingest data into Elasticsearch cluster it will automatically create a mapping with specific type for your field values.
- This seems quite easy and straightforward but based upon the data we should define a proper field mapping data type as if the wrong field type is chosen, we will get an indexing error.

Production Requirements

- Example: In case the first document is indexed like this where elasticsearch has marked the payload as date type:
- POST index-sample/_doc
- {
- “Payment”: “In process”,
- “Payload”: “2021-10-20”
- }

Production Requirements

- And the next document is like below
- POST index-sample/_doc
- {
- “Payment”: “In process”,
- “Payload”: “user_locked”
- }

Production Requirements

- And the next document is like below
- POST index-sample/_doc
- {
- “Payment”: “In process”,
- “Payload”: “user_locked”
- }

Production Requirements

- Where the payload is not date type, elasticsearch may pop-up an error as it has already saved the payload field as a date.
- To avoid the above error, we may set dynamic date detection as false.

- PUT index-sample
- {
- “mappings”: {
- “date_detection”: false
- }
- }
-

Production Requirements

- PUT index-sample/_doc/1
- {
- ” Payload “: “2021-10-20”
- }

Production Requirements

- Speeding indexing:
  - To increase the indexing performance of elastic clusters, consider the below points.
  - Use of bulk request: Rather than using a single document insertion method we should use bulk requests in elastic clusters.
  - Larger requests seem to perform better but we should always do a benchmark test to check the breaking point.
  - This can be identified by trying out multiple values.
  - Cluster can also be in memory pressure when the bulk request is too large going beyond a couple tens of megabytes per request.

Production Requirements

- Speeding indexing:
  - Need for bulk ingestion can occur when we write our own code to ingest custom data in bulk.
  - Or this can be useful when we need to index data streams that can be queued up and indexed in batches of hundreds or thousands, such as logs.

Production Requirements

- Speeding indexing:
  - Use of multiple threads/workers to ingest data into Elasticsearch cluster:
  - To use our resources to the fullest we should use multiple threads to send data to the cluster.
  - If it is not able to keep up with current indexing rate elasticsearch will indicate the same by TOO_MANY_REQUESTS (429) response codes (EsRejectedExecutionException with the Java client).
  - Testing with various values can tell us about the optimal number like the above scenario.

Production Requirements

- Speeding indexing:
  - Increase refresh interval: In use cases where there is quite less or no search, increasing the refresh interval can increase our indexing process efficiency.
  - We need to explicitly set the refresh interval to choose this behavior.

Exploring Kibana

- Kibana is the front end for Elasticsearch and provides delightful visualizations for data.
- Kibana is an open source analytics engine that can be used to search, view, and analyze data.
- Various kinds of visualizations are available to illustrate data in the form of tables, charts, histograms, maps, etc.
- There is a web-based interface to handle large volumes of data.
- Creating a dashboard is quite seamless and queries data in real time. Essentially, a dashboard is nothing but a way for analyzing JSON documents.
- You can save them, make them as templates, or simply export them.
- The ease of setup and use will help you cut through the complexities of stored data in minutes.

Exploring Kibana

- Kibana comes as an Apache-licensed product.
- Its flexible interface comes out as a great combination for Elasticsearch's searching capabilities.
- You only need a web server and any modern web browser to start working with Kibana.
- It leverages the REST APIs of Elasticsearch.
- Data can be visualized in real time by using dashboards.
- This helps in getting real-time insights.

Exploring Kibana

- **Emphasis on search terms**: The list of documents returned as the response to a search contains the search terms as highlighted.
- **Aggregations**: In-depth use of Elasticsearch aggregations is done to facilitate visualizations. Buckets return a set of documents meeting criteria like terms, range, histogram, etc. Metrics, on the other hand, compute statistics like min, max, sum, or average for a set of documents.
- **Scripted fields**: Scripted fields help with computations in real time on indexed data. For example, for a certain field you want to add 500 before showing it. This can be saved as a scripted field. There is a caveat, though: scripted fields cannot be searched.

Exploring Kibana

Dynamic dashboards: Kibana dashboards are flexible and dynamic, allowing for individual visualizations to be conveniently arranged. Moreover, they enable automatic refreshing of data.

Kibana Features

Feature	Description
Visualization	Core Kibana features classic graphing interfaces: pie charts, histograms, line graphs, etc.
Dashboard	Piecing together various visualization on one dashboard pane creates a more straightforward data overview.
Generating and Sharing Reports	Generating CSV tables, embedding visualizations, and sharing via URL.
Searching and Filtering	Data filtering and queries using the intuitive Kibana Query Language (KQL).
Plugins	Additional visualization and UI tools, such as 3D graphs, calendar visualization, and Prometheus exporter are available through plugins.
Geospatial Analysis	Visualizing spatial data provides a realistic location view.
Time Series Analysis	Visual builder for time series data analysis with aggregation.
Canvas	Fully customizable colors, shapes, texts, and queries for dynamic presentations.

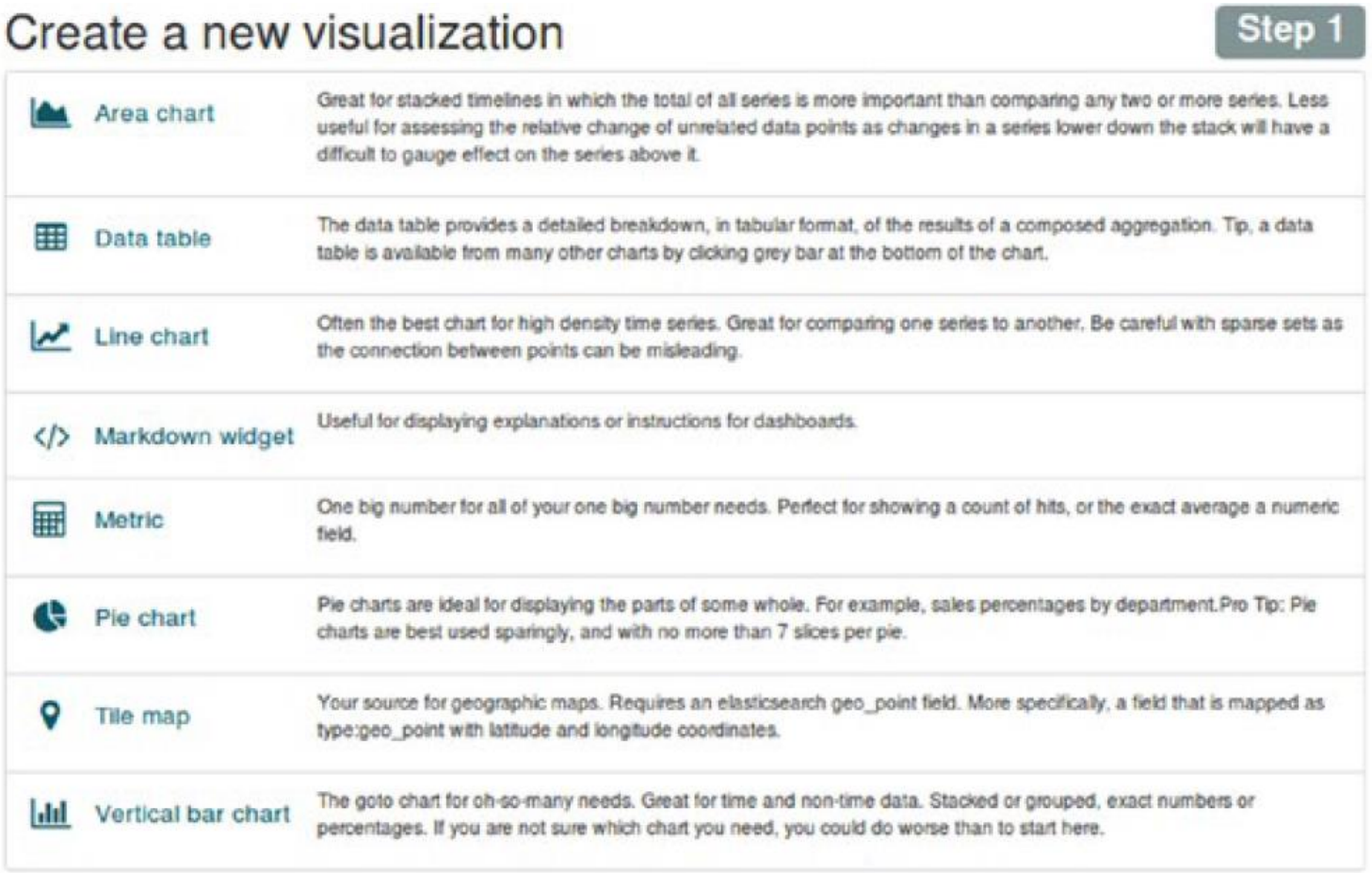
Kibana User Interface

- The Kibana user interface consists of four main tabs:
- 1. **Discover**: This page is used for search based on free form text, fields, or ranges.
- 2. **Visualize**: As the name suggests, this page provides the facility for creating multiple visualizations like bar charts, pie charts, line charts, etc. These charts can be saved for use in dashboards later.

Kibana User Interface

- **Dashboard**: This page is a set of collections of multiple visualizations and searches. It allows for easy application of filters based on click interaction, which helps in drawing conclusions based on multiple data aggregations.
- 4. **Settings**: The settings page is used to configure kibana operational parameters. Some of the examples are index patterns, scripted fields, data types of fields, etc.

Kibana - Data Visualization



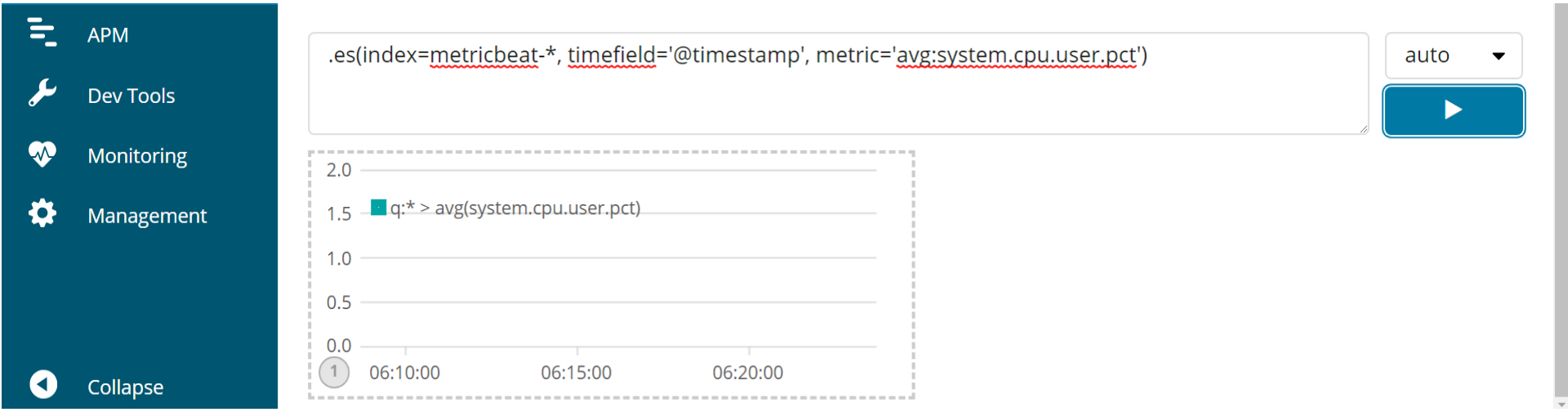
Timelion

- Timelion is a time series data visualizer that enables you to combine totally independent data sources within a single visualization.
- It’s driven by a simple expression language you use to retrieve time series data, perform calculations to tease out the answers to complex questions, and visualize the results.

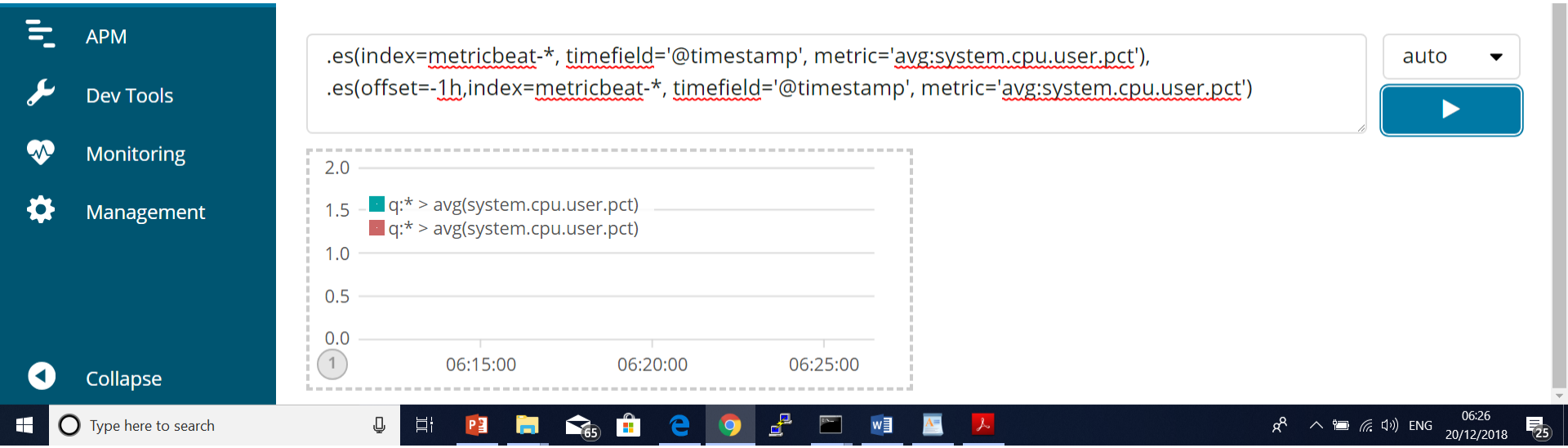
Timelion

- How many pages does each unique user view over time?
- What's the difference in traffic volume between this Friday and last Friday?
- What percent of Japan's population came to my site today?
- What's the 10-day moving average of the S&P 500?
- What's the cumulative sum of all search requests received in the last 2 years?

Timelion


APM
Dev Tools
Monitoring
Management
Collapse
.es(index=metricbeat-*, timefield='@timestamp', metric='avg:system.cpu.user.pct')
auto
2.0
1.5
1.0
0.5
0.0
q:* > avg(system.cpu.user.pct)
06:10:00
06:15:00
06:20:00

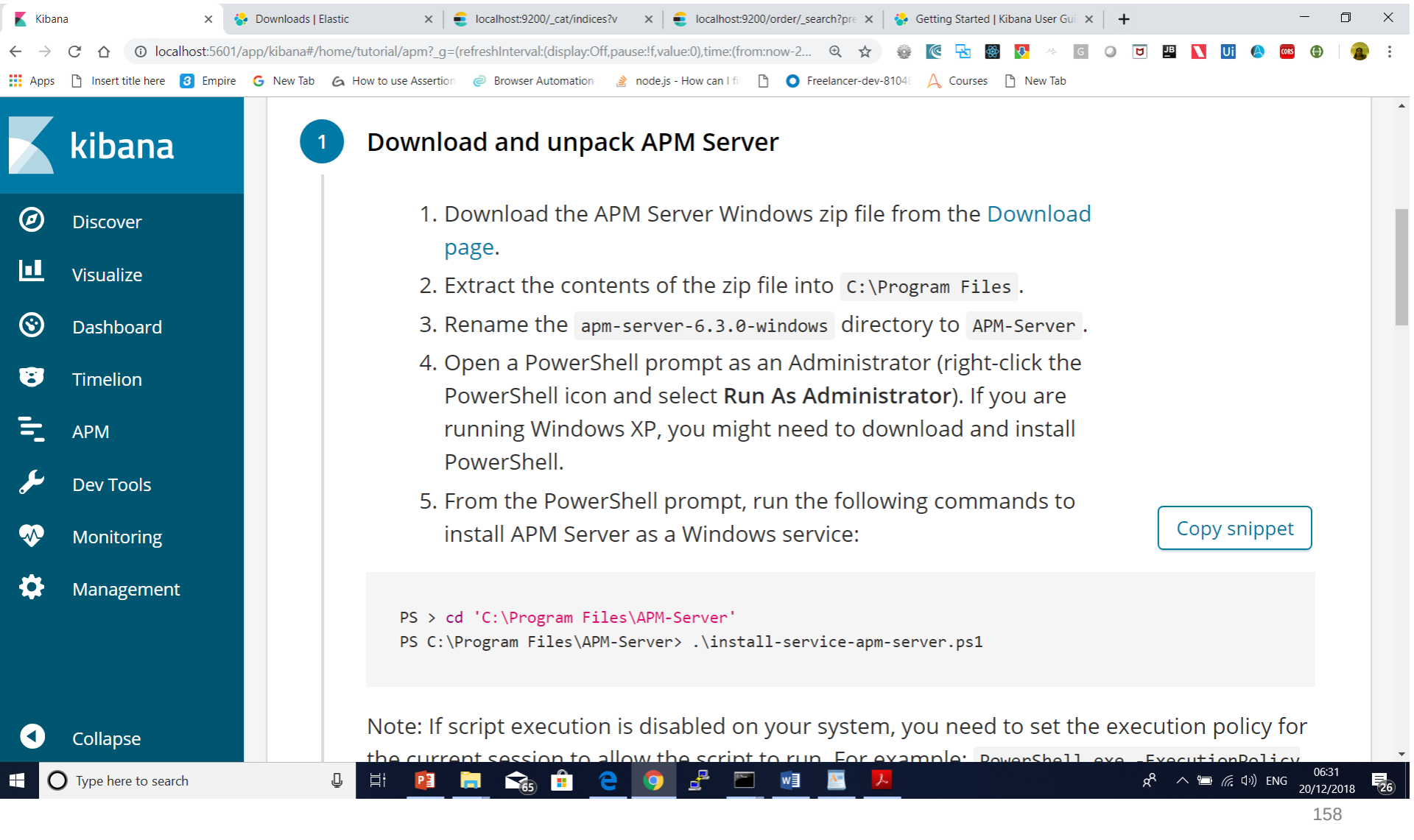
Timelion


APM
Dev Tools
Monitoring
Management
Collapse
.es(index=metricbeat-*, timefield='@timestamp', metric='avg:system.cpu.user.pct'),
.es(offset=-1h,index=metricbeat-*, timefield='@timestamp', metric='avg:system.cpu.user.pct')
auto
2.0
1.5
1.0
0.5
0.0
q:* > avg(system.cpu.user.pct)
q:* > avg(system.cpu.user.pct)
06:15:00
06:20:00
06:25:00
Type here to search
ENG
06:26
20/12/2018

APM

- Elastic Application Performance Monitoring (APM) automatically collects in-depth performance metrics and errors from inside your applications.
- The **APM** page in Kibana is provided with the X-Pack basic license. It enables developers to drill down into the performance data for their applications and quickly locate the performance bottlenecks.

APM

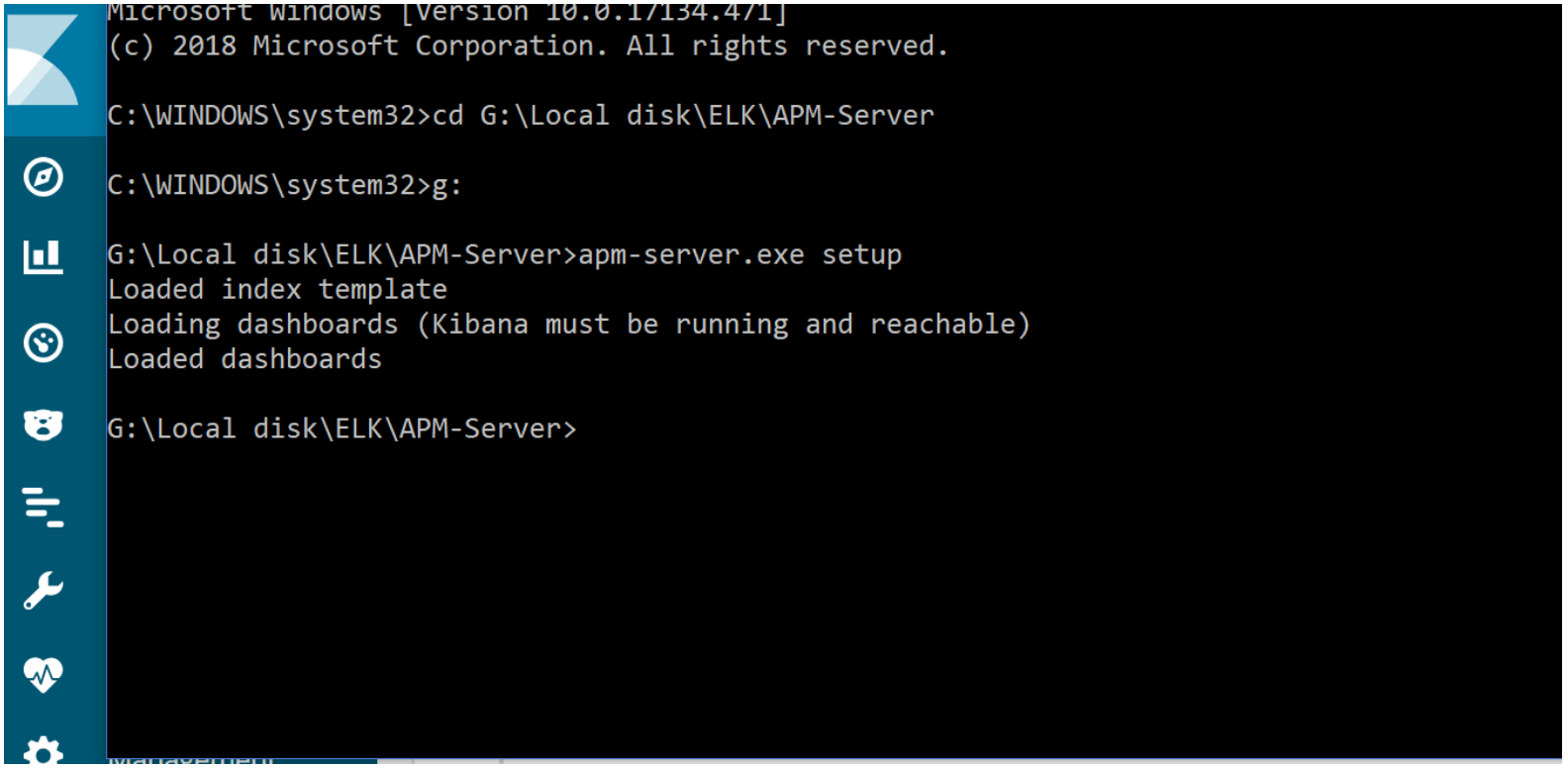

Kibana
Downloads | Elastic
localhost:9200/_cat/indices?v
localhost:9200/order/_search?pre
Getting Started | Kibana User Gui
localhost:5601/app/kibana#/home/tutorial/apm?_g=(refreshInterval:(display:Off,pause:!f,value:0),time:(from:now-2...
Apps
Insert title here
Empire
New Tab
How to use Assertion
Browser Automation
node.js - How can I fi
Freelancer-dev-8104
Courses
New Tab
kibana
Discover
Visualize
Dashboard
Timelion
APM
Dev Tools
Monitoring
Management
Collapse
1
Download and unpack APM Server
1. Download the APM Server Windows zip file from the Download page.
2. Extract the contents of the zip file into C:\Program Files.
3. Rename the apm-server-6.3.0-windows directory to APM-Server.
4. Open a PowerShell prompt as an Administrator (right-click the PowerShell icon and select Run As Administrator). If you are running Windows XP, you might need to download and install PowerShell.
5. From the PowerShell prompt, run the following commands to install APM Server as a Windows service:
Copy snippet
PS > cd 'C:\Program Files\APM-Server'
PS C:\Program Files\APM-Server> .\install-service-apm-server.ps1
Note: If script execution is disabled on your system, you need to set the execution policy for
Type here to search
ENG
06:31
20/12/2018

APM

Set-ExecutionPolicy -ExecutionPolicy bypass

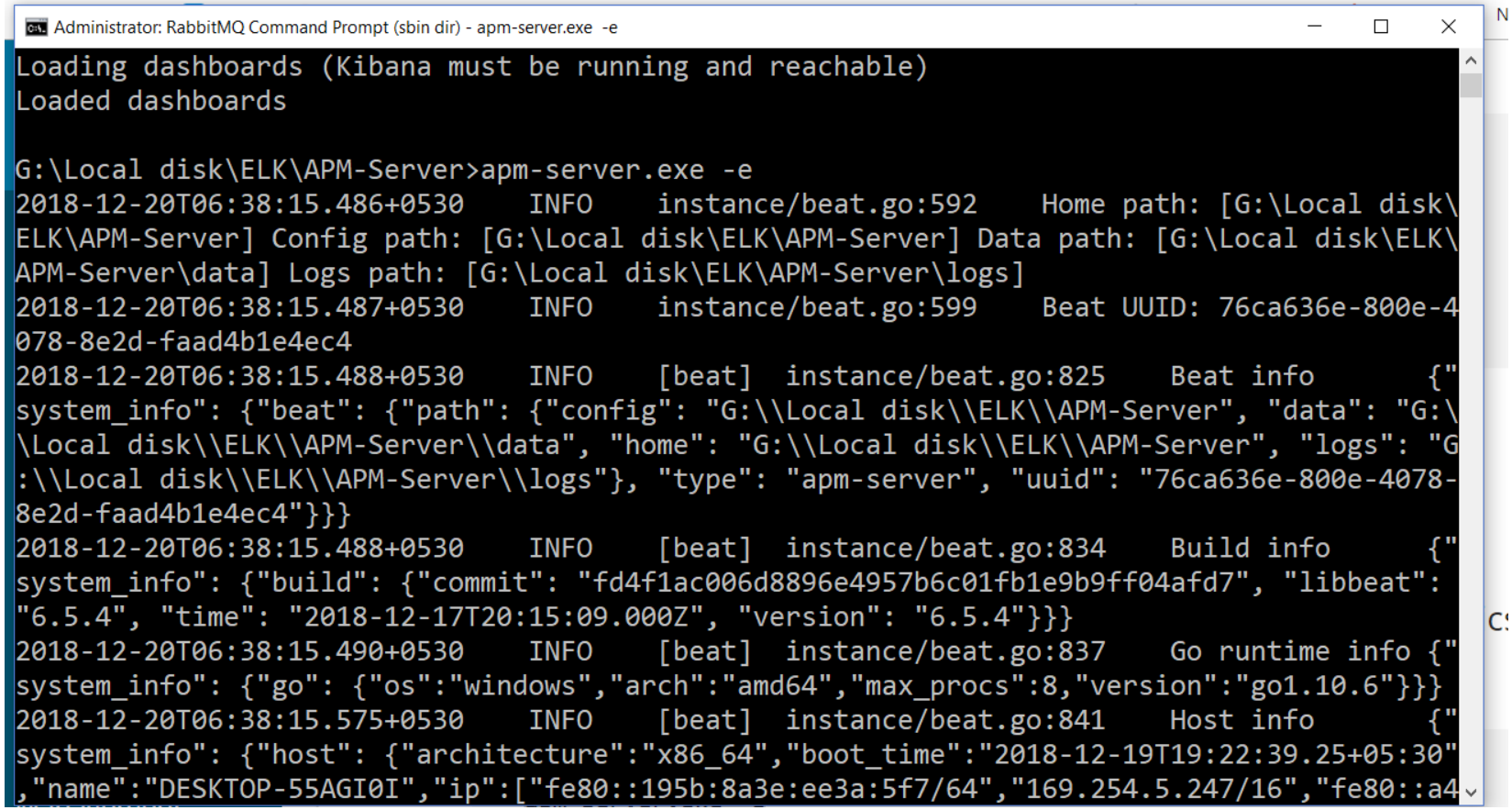
```
PS G:\Local disk\ELK\APM-Server> .\install-service-apm-server.ps1


__GENUS          : 2
__CLASS          : __PARAMETERS
__SUPERCLASS     :
__DYNASTY        : __PARAMETERS
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
ReturnValue      : 5
PSComputerName   :

__GENUS          : 2
__CLASS          : __PARAMETERS
__SUPERCLASS     :
__DYNASTY        : __PARAMETERS
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
ReturnValue      : 0
```

APM


Microsoft Windows [Version 10.0.17134.471]
(c) 2018 Microsoft Corporation. All rights reserved.
C:\WINDOWS\system32>cd G:\Local disk\ELK\APM-Server
C:\WINDOWS\system32>g:
G:\Local disk\ELK\APM-Server>apm-server.exe setup
Loaded index template
Loading dashboards (Kibana must be running and reachable)
Loaded dashboards
G:\Local disk\ELK\APM-Server>

APM

```
./apm-server -e -E output.elasticsearch.hosts=host.docker.internal:9200 -E apm-server.host=localhost:8200
```



APM java

- refer ericmsjuly2021
- ==========

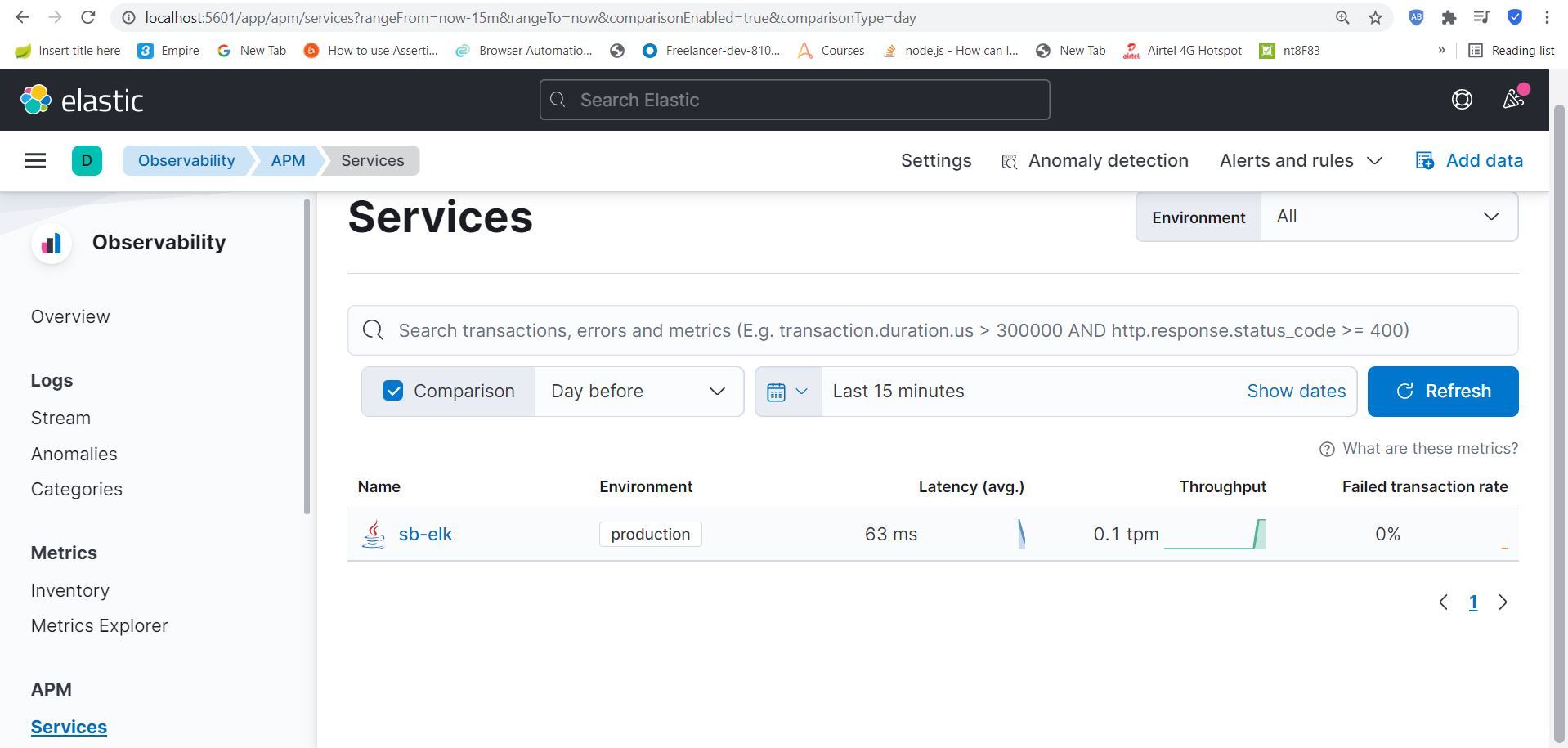
- java -javaagent:"D:\ELK\apm-server-7.15.0-windows-x86_64\elastic-apm-agent-1.26.0.jar" \
- -Delastic.apm.service_name=sb-elk \
- -Delastic.apm.server_urls=http://localhost:8200 \
- -Delastic.apm.secret_token= \
- -Delastic.apm.environment=production \
- -Delastic.apm.application_packages=com.dell \
- -jar spring-boot-elk-0.0.1-SNAPSHOT.jar

APM java

Administrator: Command Prompt - java -javaagent:"D:\ELK\apm-server-7.15.0-windows-x86_64\elastic-apm-agent-1.26.0.jar" -Delastic.apm.service_name=sb-elk -Delastic.apm.server_urls=http://localhost:8200 -Delastic.apm.secret_token= -Delastic.apm.envir...

```
F:\ericjulyms2021\spring-boot-elk\target>java -javaagent:elastic-apm-agent-1.26.0.jar -Delastic.apm.service_name=sb-elk -Delastic.apm.s
erver_urls=http://localhost:8200 -Delastic.apm.secret_token= -Delastic.apm.environment=production -Delastic.apm.application_packages=co
m.dell -jar spring-boot-elk-0.0.1-SNAPSHOT.jar
Error opening zip file or JAR manifest missing : elastic-apm-agent-1.26.0.jar
Error occurred during initialization of VM
agent library failed to init: instrument

F:\ericjulyms2021\spring-boot-elk\target>java -javaagent:"D:\ELK\apm-server-7.15.0-windows-x86_64\elastic-apm-agent-1.26.0.jar" -Delast
ic.apm.service_name=sb-elk -Delastic.apm.server_urls=http://localhost:8200 -Delastic.apm.secret_token= -Delastic.apm.environment=produc
tion -Delastic.apm.application_packages=com.dell -jar spring-boot-elk-0.0.1-SNAPSHOT.jar
2021-09-29 20:24:24,363 [main] INFO  co.elastic.apm.agent.util.JmxUtils - Found JVM-specific OperatingSystemMXBean interface: com.sun.m
anagement.OperatingSystemMXBean
2021-09-29 20:24:24,377 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - Starting Elastic APM 1.26.0 as sb-elk on Java 1.8
.0_251 Runtime version: 1.8.0_251-b08 VM version: 25.251-b08 (Oracle Corporation) Windows 10 10.0
2021-09-29 20:24:24,377 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - service_name: 'sb-elk' (source: Java System Prope
rties)
2021-09-29 20:24:24,378 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - environment: 'production' (source: Java System Pr
operties)
2021-09-29 20:24:24,378 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - secret_token: 'XXXX' (source: Java System Propert
ies)
2021-09-29 20:24:24,378 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - server_urls: 'http://localhost:8200' (source: Jav
a System Properties)
2021-09-29 20:24:24,379 [main] INFO  co.elastic.apm.agent.configuration.StartupInfo - application_packages: 'com.dell' (source: Java Sy
stem Properties)
2021-09-29 20:24:28,459 [elastic-apm-server-healthcheck] INFO  co.elastic.apm.agent.report.ApmServerHealthChecker - Elastic APM server
is available: {  "build_date": "2021-09-16T01:57:32Z",  "build_sha": "a183f675ecd03fca4a897cbe85fda3511bc3ca43",  "version": "7.15.0"}
2021-09-29 20:24:28,785 [main] INFO  co.elastic.apm.agent.impl.ElasticApmTracer - Tracer switched to RUNNING state


  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
```

APM java


localhost:5601/app/apm/services?rangeFrom=now-15m&rangeTo=now&comparisonEnabled=true&comparisonType=day
Insert title here
Empire
New Tab
How to use Asserti...
Browser Automatio...
Freelancer-dev-810...
Courses
node.js - How can I...
New Tab
Airtel 4G Hotspot
nt8F83
Reading list
elastic
Search Elastic
D
Observability
APM
Services
Settings
Anomaly detection
Alerts and rules
Add data
Observability
Overview
Logs
Stream
Anomalies
Categories
Metrics
Inventory
Metrics Explorer
APM
Services
Services
Environment
All
Search transactions, errors and metrics (E.g. transaction.duration.us > 300000 AND http.response.status_code >= 400)
Comparison
Day before
Last 15 minutes
Show dates
Refresh
What are these metrics?
Name
Environment
Latency (avg.)
Throughput
Failed transaction rate
sb-elk
production
63 ms
0.1 tpm
0%
1

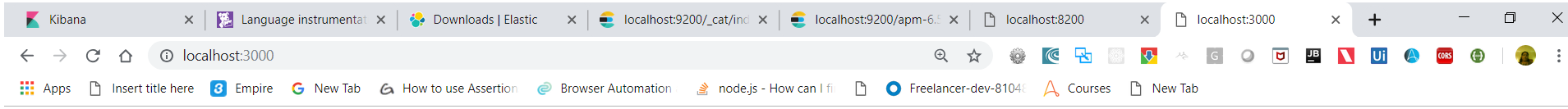
APM

http://localhost:9200/apm-6.5.4-onboarding-2018.12.20/_search?pretty

APM Nodejs

- For apm server 6.3
- npm install elastic-apm-node@1.14.3 --save

APM



Hello World!

APM

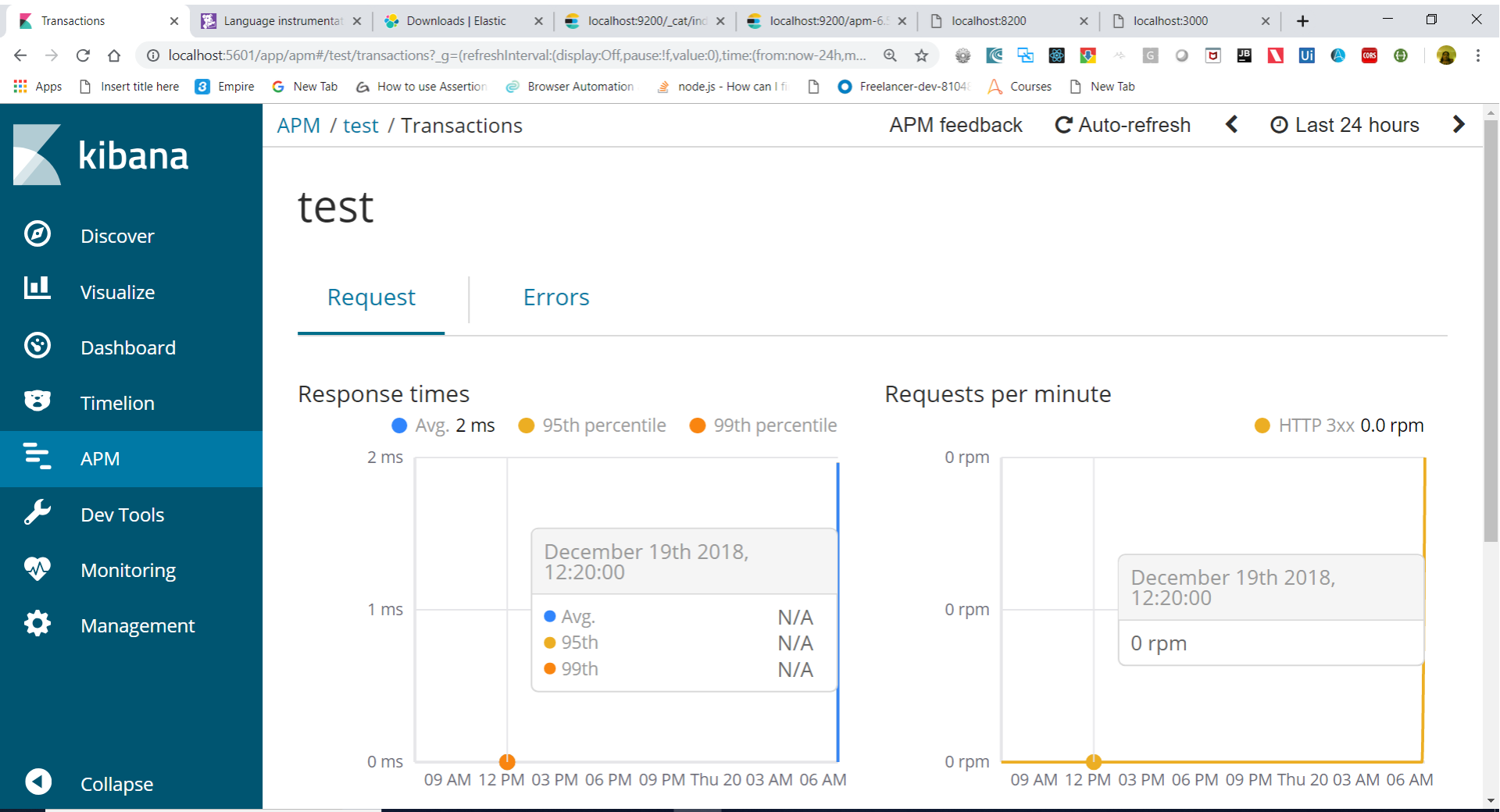
Administrator: Node.js command prompt

```
Your environment has been set up for using Node.js 9.3.0 (x64) and npm.

C:\Windows\System32>npm -g install elastic-apm-node --save
+ elastic-apm-node@2.0.6
added 84 packages in 26.307s

C:\Windows\System32>
```

Type here to search

ENG 06:42 20/12/2018

APM


Transactions
Language instrumentat
Downloads | Elastic
localhost:9200/_cat/ind
localhost:9200/apm-6.5
localhost:8200
localhost:3000
localhost:5601/app/apm#/test/transactions?_g=(refreshInterval:(display:Off,pause:!f,value:0),time:(from:now-24h,m...
Apps
Insert title here
Empire
New Tab
How to use Assertion
Browser Automation
node.js - How can I fi
Freelancer-dev-8104
Courses
New Tab
kibana
Discover
Visualize
Dashboard
Timelion
APM
Dev Tools
Monitoring
Management
Collapse
APM / test / Transactions
APM feedback
Auto-refresh
Last 24 hours
test
Request
Errors
Response times
Avg. 2 ms
95th percentile
99th percentile
2 ms
1 ms
0 ms
December 19th 2018, 12:20:00
Avg. N/A
95th N/A
99th N/A
09 AM 12 PM 03 PM 06 PM 09 PM Thu 20 03 AM 06 AM
Requests per minute
HTTP 3xx 0.0 rpm
0 rpm
0 rpm
0 rpm
December 19th 2018, 12:20:00
0 rpm
09 AM 12 PM 03 PM 06 PM 09 PM Thu 20 03 AM 06 AM
Type here to search
ENG
07:00
20/12/2018

Elastic SQL

- select * from rating where rating=4.0;

Docker

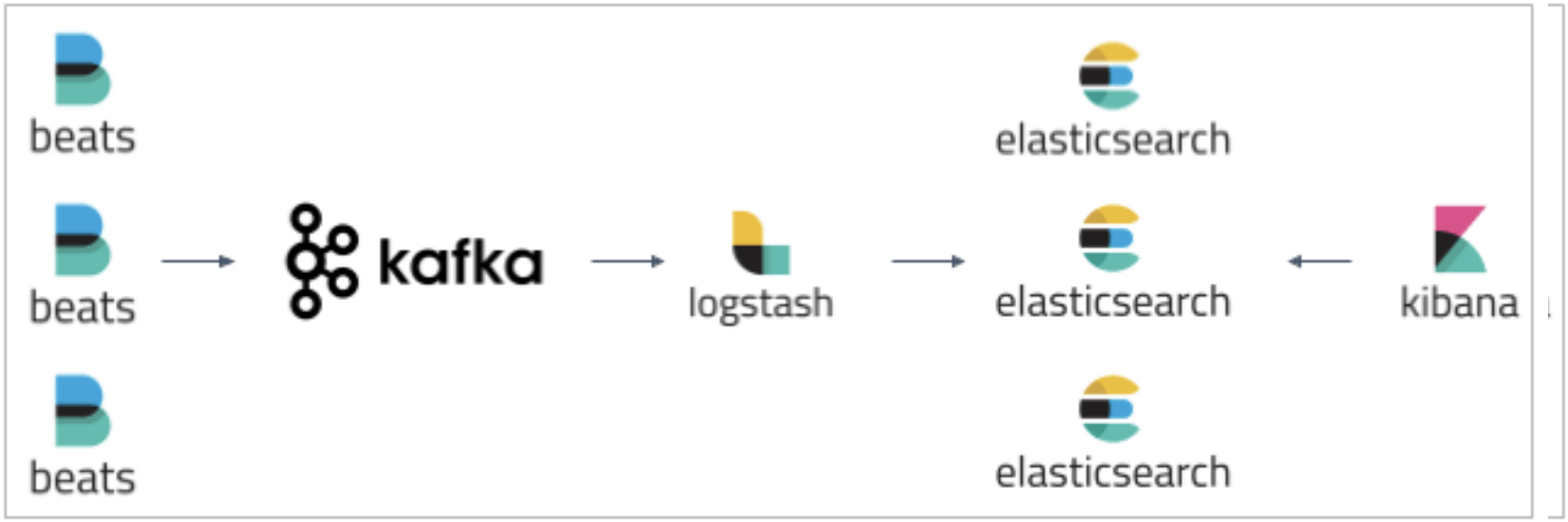
MINGW64:/d/Program Files/Docker Toolbox

```
                        ##         .
                  ## ## ##        ==
               ## ## ## ## ##    ===
           /"""""""""""""""""\___/ ===
      ~~~ {~~ ~~~~ ~~~ ~~~~ ~~~ ~ /  ===- ~~~
           \______ o           __/
             \    \         __/
              \____\_______/

docker is configured to use the default machine with IP 192.168.99.103
For help getting started, check out the docs at https://docs.docker.com


Start interactive shell

Balasubramaniam@DESKTOP-55AGI0I MINGW64 /d/Program Files/Docker Toolbox
$ docker pull docker.elastic.co/elasticsearch/elasticsearch:7.7.1
7.7.1: Pulling from elasticsearch/elasticsearch
86dbb57a3083: Downloading [>                                                  ]  1.051MB/85.34MB
2b5d7fba0568: Download complete
3f15a7e37bf2: Downloading [==>                                                ]  1.119MB/26.86MB
c11eec231e4f: Waiting
3c87444dd504: Waiting
95c144f0b3e7: Waiting
027c17d4d678: Waiting
6ea63efa2205: Waiting
e9b3616d9c02: Waiting
```

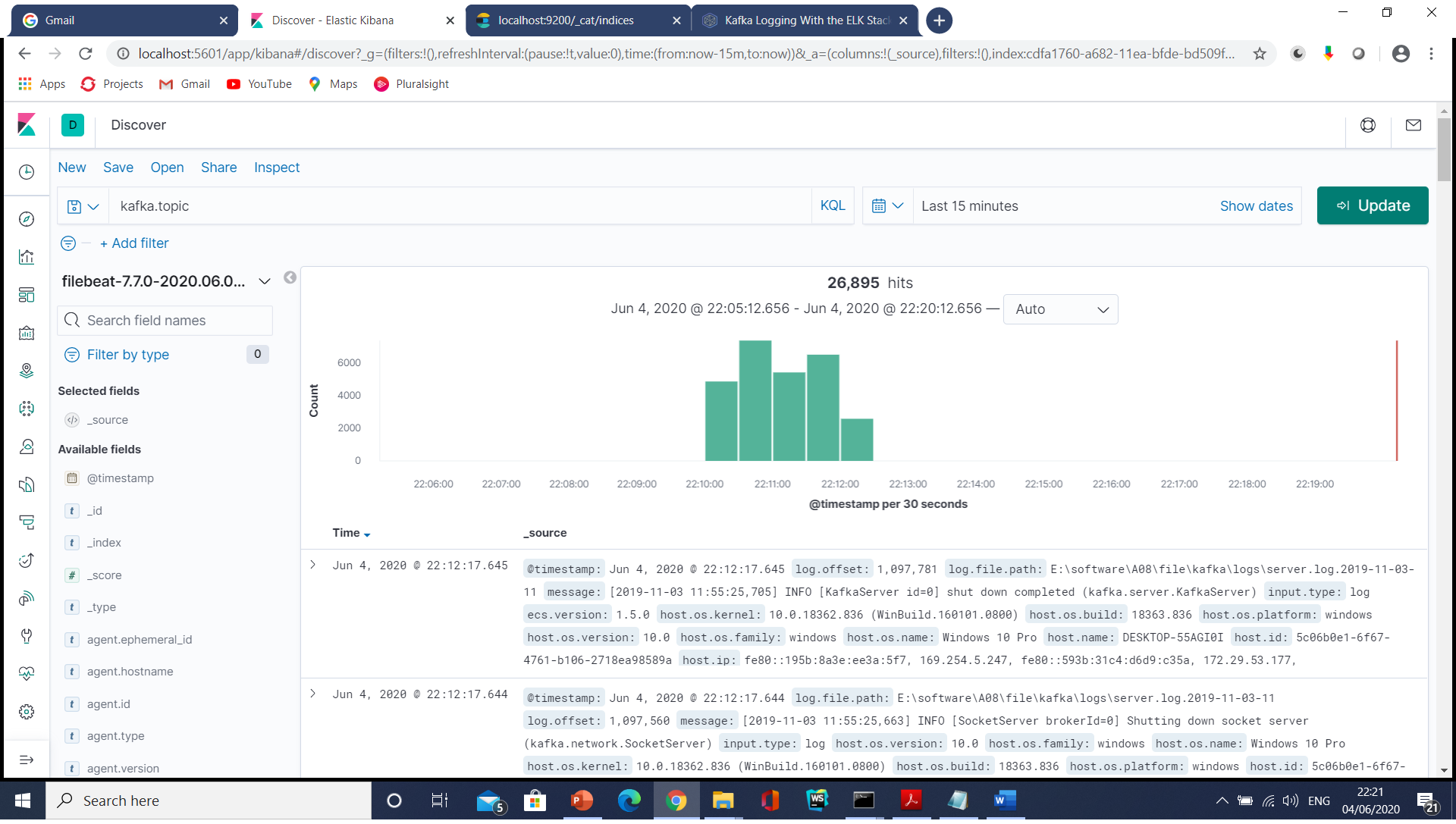
Docker

- docker run -h 192.168.99.103 -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node" docker.elastic.co/elasticsearch/elasticsearch:7.7.1
- Run Below command on Host machine it will resolve the issue
- $ sysctl -w vm.max_map_count=262144

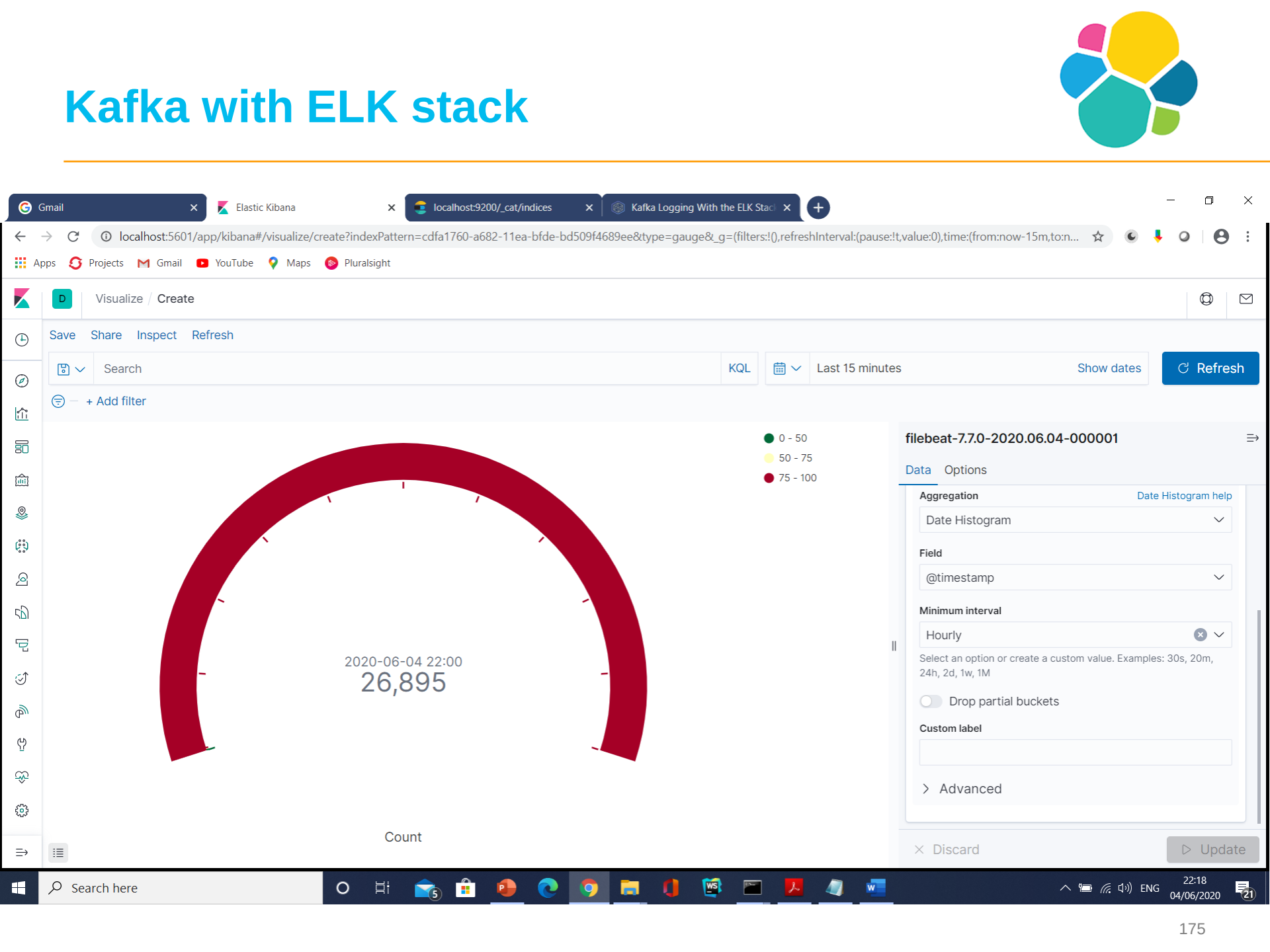
Kafka with ELK stack



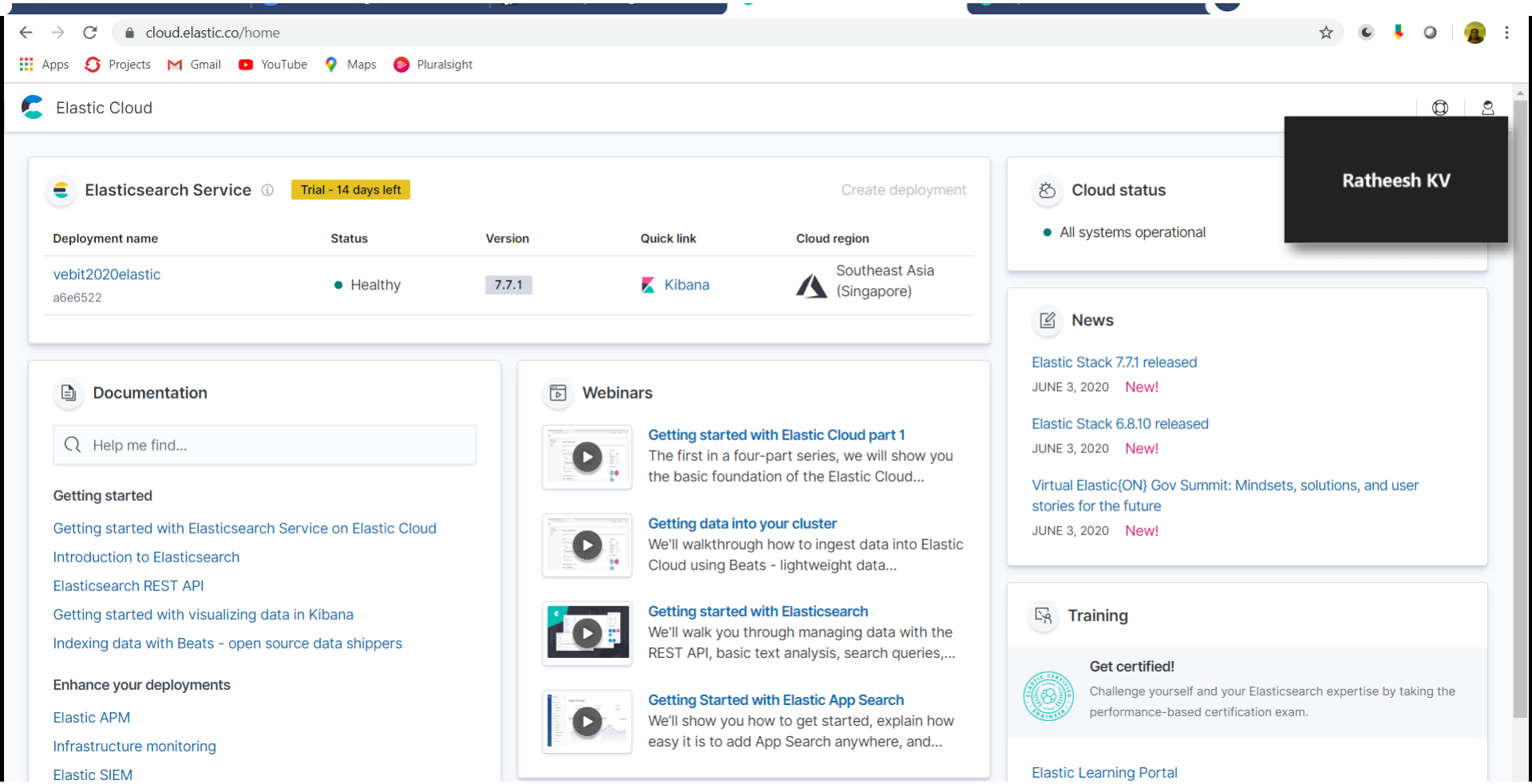
Kafka with ELK stack



Kafka with ELK stack



Elastic Cloud



Xpack Security

- Get license higher than basic
- xpack.security.enabled: true
- Discovery.type:single-node
- There are built-in users that you can use for specific administrative purposes: apm_system, beats_system, elastic, kibana, logstash_system, and remote_monitoring_user.
- Restart Elasticsearch before you set users

Xpack Security

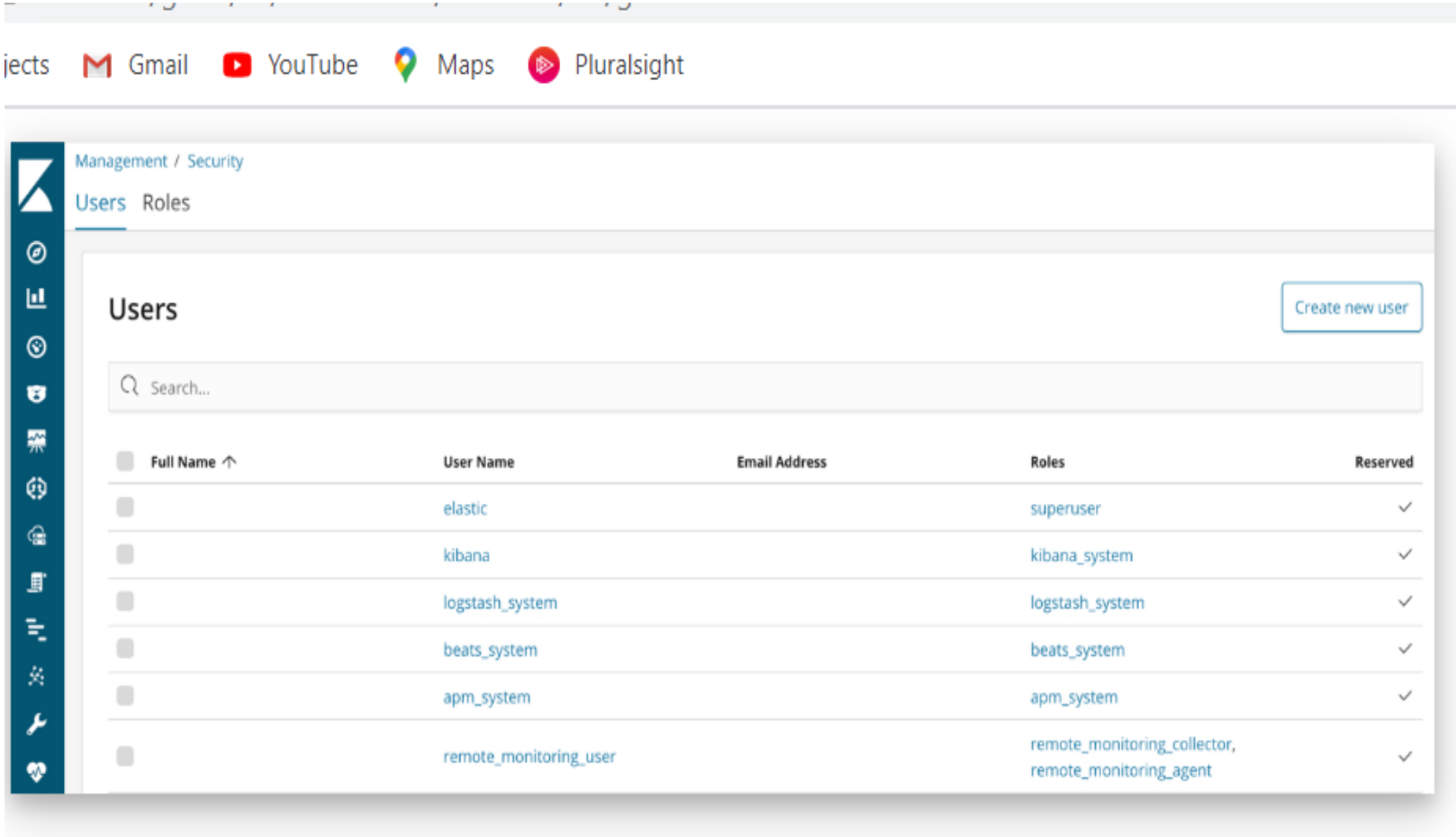
- Set the built-in users' passwords. Run the following command from the Elasticsearch directory:
- ./bin/elasticsearch-setup-passwords interactive
- Use kibana.yml
- elasticsearch.username: "kibana"
- elasticsearch.password: "your_password/bin/kibana-keystore create
- ./bin/kibana-keystore add elasticsearch.username
- ./bin/kibana-keystore add elasticsearch.password"

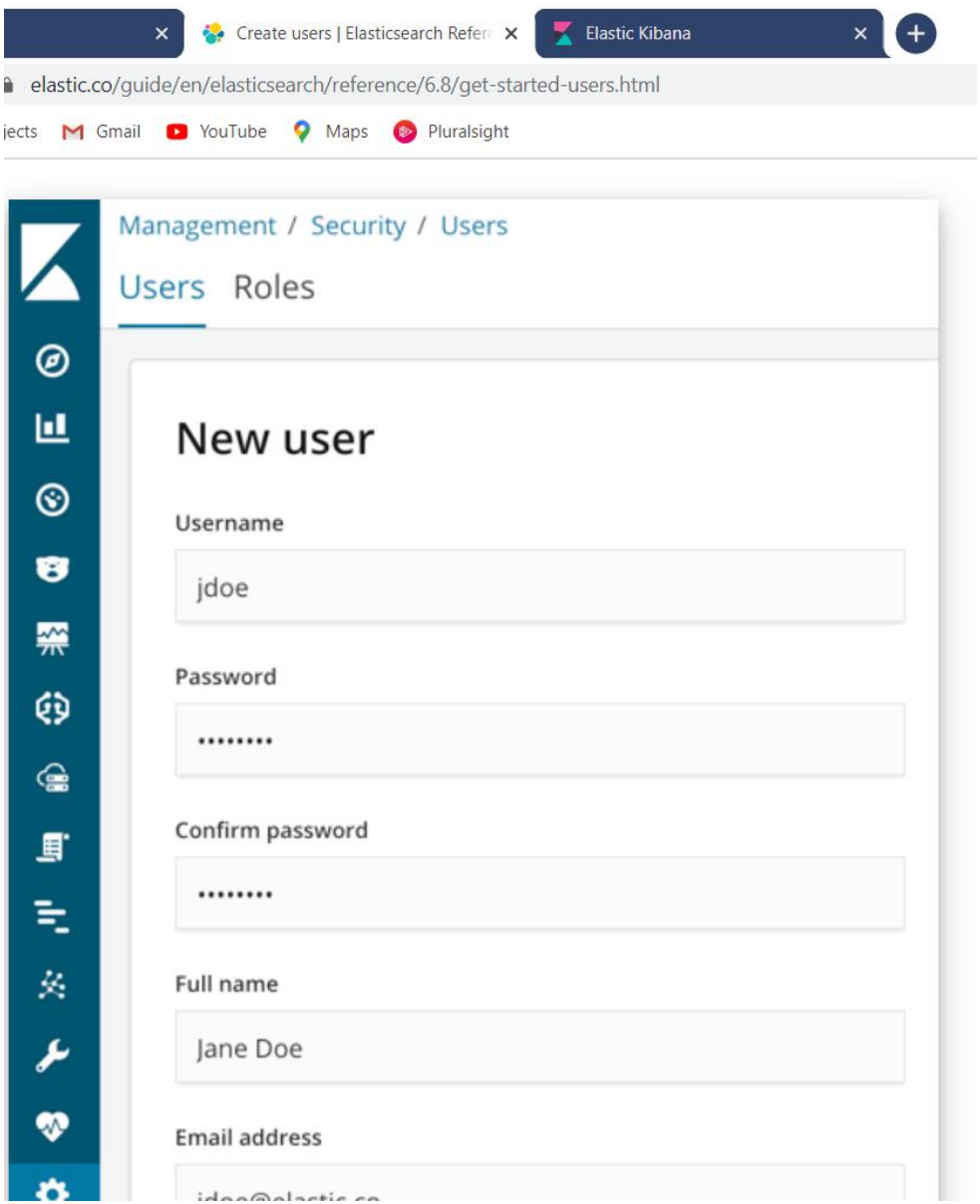
Xpack Security

- Restart kibana

- Let’s create two users in the native realm.

- Log in to Kibana with the elastic built-in user.
- Go to the Management / Security / Users page:

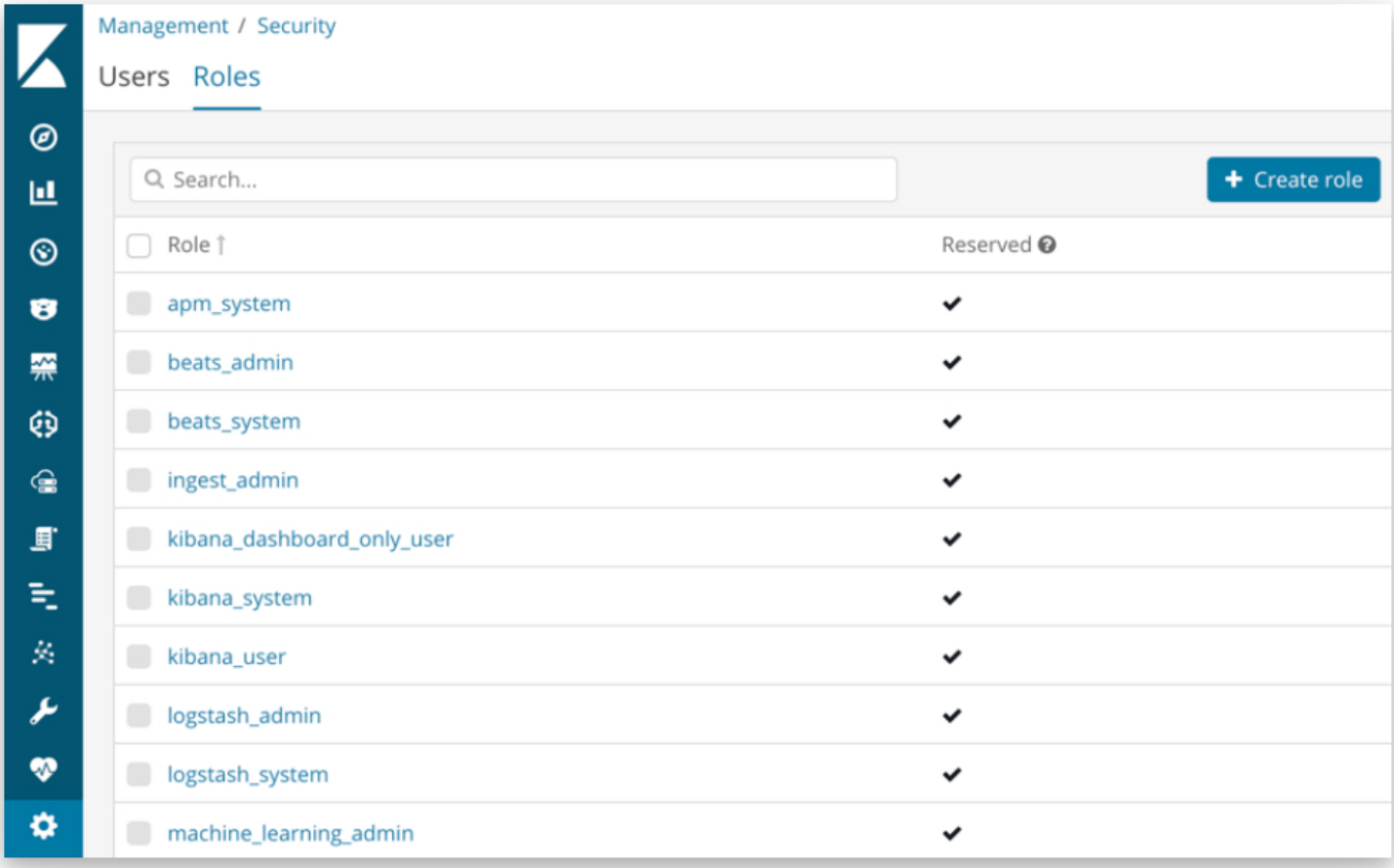

jects
Gmail
YouTube
Maps
Pluralsight
Management / Security
Users
Roles
Users
Create new user
Search...
Full Name
User Name
Email Address
Roles
Reserved
elastic
superuser
kibana
kibana_system
logstash_system
logstash_system
beats_system
beats_system
apm_system
apm_system
remote_monitoring_user
remote_monitoring_collector,
remote_monitoring_agent

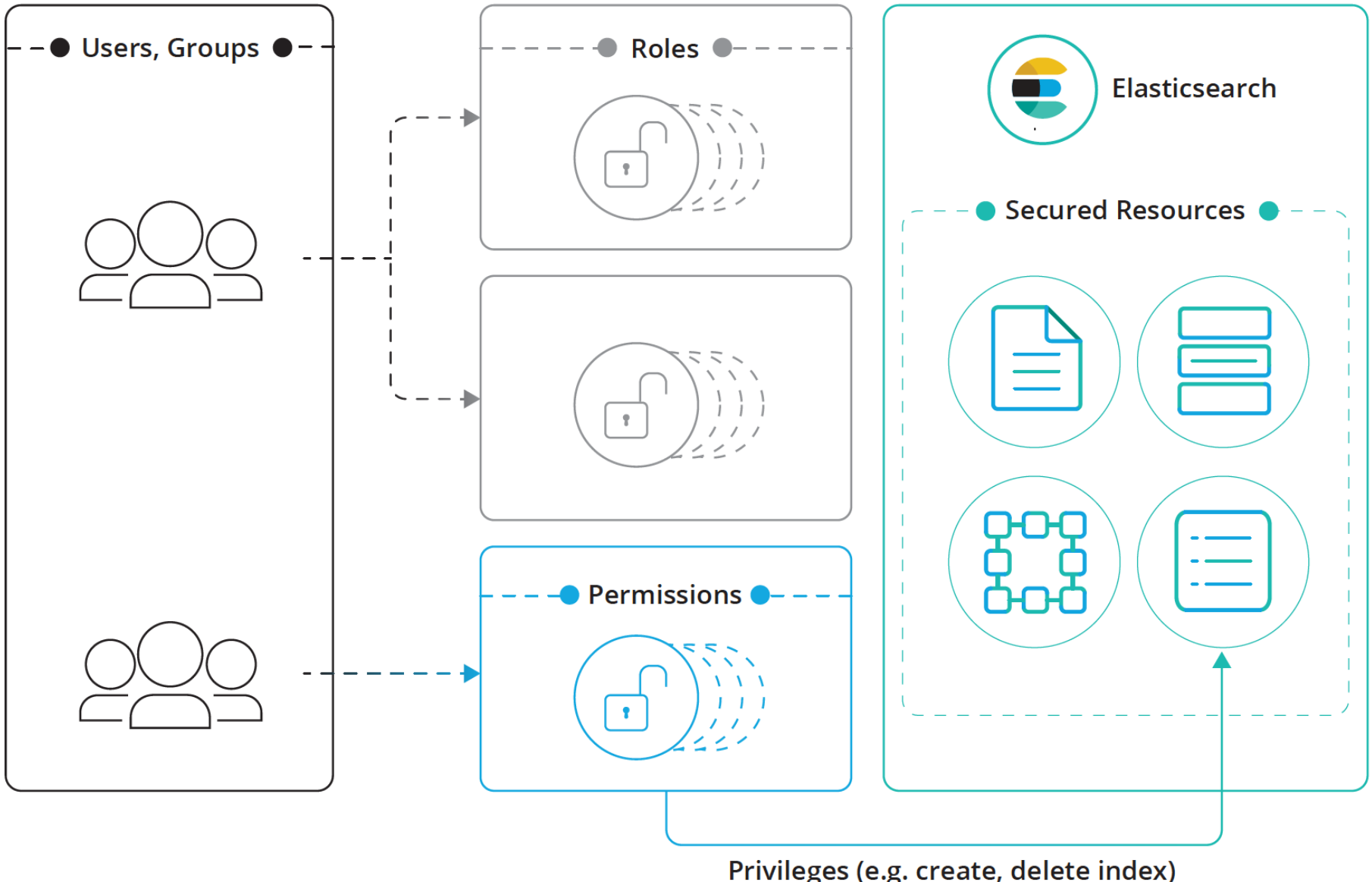

Create users | Elasticsearch Refer
Elastic Kibana
elastic.co/guide/en/elasticsearch/reference/6.8/get-started-users.html
jects
Gmail
YouTube
Maps
Pluralsight
Management / Security / Users
Users
Roles
New user
Username
jdoe
Password
Confirm password
Full name
Jane Doe
Email address

elastic.co/guide/en/elasticsearch/reference/6.8/get-started-roles.html

Apps Projects Gmail YouTube Maps Pluralsight

Go to the **Management / Security / Roles** page to see them:




Users, Groups
Roles
Permissions
Elasticsearch
Secured Resources
Privileges (e.g. create, delete index)

Built in roles

- ingest_admin
- Grants access to manage all index templates and all ingest pipeline configurations.
- This role does not provide the ability to create indices; those privileges must be defined in a separate role.
- kibana_dashboard_only_user
- Grants access to the Kibana Dashboard and read-only permissions to Kibana. This role does not have access to editing tools in Kibana. For more information, see Kibana Dashboard Only Mode.
- kibana_system
- Grants access necessary for the Kibana system user to read from and write to the Kibana indices, manage index templates and tokens, and check the availability of the Elasticsearch cluster. This role grants read access to the .monitoring-* indices and read and write access to the .reporting-* indices.

Built in roles

- apm_system
- Grants access necessary for the APM system user to send system-level data (such as monitoring) to Elasticsearch.
- beats_admin
- Grants access to the .management-beats index, which contains configuration information for the Beats.
- beats_system
- Grants access necessary for the Beats system user to send system-level data (such as monitoring) to Elasticsearch.

Role

- click_admins:
- run_as: ['clicks_watcher_1']
- cluster: ['monitor']
- indices:
- - names: ['events-*']
- privileges: ['read']
- field_security:
- grant: ['category', '@timestamp', 'message']
- query: '{"match": {"category": "click"}}'

Document Level Security

```
POST /_xpack/security/role/click_role
{
  "indices": [
    {
      "names": [ "events-*" ],
      "privileges": [ "read" ],
      "query": "{\"match\": {\"category\": \"click\"}}"
    }
  ]
}
```

Document Level Security

```
POST /_xpack/security/role/dept_role
{
  "indices" : [
    {
      "names" : [ "*" ],
      "privileges" : [ "read" ],
      "query" : {
        "term" : { "department_id" : 12 }
      }
    }
  ]
}
```

Field Level Security

```
POST /_xpack/security/role/test_role1
{
  "indices": [
    {
      "names": [ "events-*" ],
      "privileges": [ "read" ],
      "field_security" : {
        "grant" : [ "category", "@timestamp", "message" ]
      }
    }
  ]
}
```

Field Level Security

```
POST /_xpack/security/role/test_role4
{
  "indices" : [
    {
      "names" : [ "*" ],
      "privileges" : [ "read" ],
      "field_security" : {
        "grant" : [ "customer.*" ]
      }
    }
  ]
}
```

Grant index level privileges

- POST /_aliases
- {
- "actions" : [
- { "add" : { "index" : "2015", "alias" : "current_year" } }
-]
- }

Manage users and Groups

- monitoring:
- - "cn=admins,dc=example,dc=com"
- user:
- - "cn=John Doe,cn=contractors,dc=example,dc=com"
- - "cn=users,dc=example,dc=com"
- - "cn=admins,dc=example,dc=com"

Manage users and Groups

- PUT _xpack/security/role_mapping/admin_user
- {
- "roles" : ["monitoring"],
- "rules" : { "field" : { "dn" : "cn=Admin,ou=example,o=com" } },
- "enabled": true
- }

Questions

Module Summary

- Spring Integration Framework.
- Message, Channel and Adapter
- Understood the different Component Integration
- Understood the Event-Driven Architecture