**ELK- Standard Operating procedure**

Version 0.1

**Document History:**

**Document Amendment History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Description** | **Date** | **By** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Document Review History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document Version** | **Date Reviewed** | **Reviewed By** | **Role** | **Review Status** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Document Approval**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document Version** | **Date Approved** | **Approved By** | **Role** | **Document Status** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

[**1. Overview 5**](#_Toc465174756)

[**Introduction** 5](#_Toc465174757)

[**What is ELK?** 5](#_Toc465174758)

[**Why ELK?** 5](#_Toc465174761)

[**2.** **Elastic search** 6](#_Toc465174762)

[**2.1** **How elastic search Works?** 7](#_Toc465174764)

[**2.2** **Installation & Configuration** 8](#_Toc465174768)

[**2.3** **Exploring Your Cluster** 9](#_Toc465174771)

[**The REST API** 9](#_Toc465174772)

[**2.4** **Logstash** 15](#_Toc465174773)

[**2.4.1 The Power of Logstash***:* 15](#_Toc465174774)

[**2.4.2 Getting Started with Logstash** 16](#_Toc465174776)

[**Inputs***:* 17](#_Toc465174777)

[**Filter*s*** 18](#_Toc465174778)

[**Outputs** 18](#_Toc465174779)

[**2.4.3 Starting Logstash** 19](#_Toc465174780)

[**2.5** **Kibana** 22](#_Toc465174782)

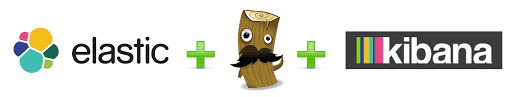
[**2.5.1** **Installation & Configuration** 22](#_Toc465174783)

1. **Overview**

**Introduction**

This document illustrates the steps that are followed to insert and visualize data using ELK.

**What is ELK?**

****

ELK stands for Elasticsearch, Logstash and Kibana.

* Elasticsearch for deep search and data analytics.

Elasticsearch is a search server based on Lucene. It provides a distributed, multitenant-capable full-text search engine with a RESTful web interface and schema-free JSON documents.

* Logstash for centralized logging, log enrichment and parsing.

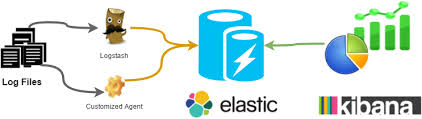
It is a tool for managing events and logs. You can use it to collect logs, parse them, and store them for later use (like, for searching). Speaking of searching, logstash comes with a web interface for searching and drilling into all of your logs. It is fully free and fully open source.

* Kibana for powerful and beautiful data visualizations.

Elasticsearch, along with Logstash and Kibana, provides a powerful platform for indexing, searching and analyzing your data.

**Why ELK?**

* The ELK stack provides a set of utilities and applications
* Each one serving a distinct purpose.
* It enables you to search all your logs in a single location.
* All these are Open source tools.
* Elasticsearch collects and stores logs, Logstash indexes the logs, and Kibana 4, a web interface, is used to view and search the logs already indexed.
* Easier to manage and monitor at scale
* Flexible architecture, integration with hadoop among others
* Less cost comparatively other alternatives
* Built by Devops for devops.



1. **Elastic search**

Elasticsearch is a search server based on Lucene. It provides a distributed, multitenant-capable full-text search engine with a RESTful web interface and schema-free JSON documents. Elasticsearch is developed in Java and is released as open source under the terms of the Apache License.

* It is a search engine not a search tool in a box.
* Open source search and analytics engine.
* Distributed and scalable search engine
* Designed for Horizontal scaling replication, fail over, Load balancing.
* Easy management & easy to use API
* Based on Lucene
* Hiding Lucene complexity by exposing all services
* HTTP, JSON, REST
* Works with all technologies.
* Fast search.
  1. **How elastic search Works?**

### Cluster

A cluster is a collection of one or more nodes (servers) that together holds your entire data and provides federated indexing and search capabilities across all nodes. A cluster is identified by a unique name which by default is "elasticsearch". This name is important because a node can only be part of a cluster if the node is set up to join the cluster by its name. It is good practice to explicitly set the cluster name in production, but it is fine to use the default for testing/development purposes.

### Node

A node is a single server that is part of your cluster, stores your data, and participates in the cluster’s indexing and search capabilities. Just like a cluster, a node is identified by a name which by default is a random Marvel character name that is assigned to the node at startup. You can define any node name you want if you do not want the default. This name is important for administration purposes where you want to identify which servers in your network correspond to which nodes in your Elasticsearch cluster.

A node can be configured to join a specific cluster by the cluster name. By default, each node is set up to join a cluster named elasticsearch.

In a single cluster, you can have as many nodes as you want. Furthermore, if there are no other Elasticsearch nodes currently running on your network, starting a single node will by default form a new single-node cluster named elasticsearch.

### Index

An index is a collection of documents that have somewhat similar characteristics. For example, you can have an index for customer data, another index for a product catalog, and yet another index for order data. An index is identified by a name (that must be all lowercase) and this name is used to refer to the index when performing indexing, search, update, and delete operations against the documents in it.

In a single cluster, you can define as many indexes as you want.

**Shards & Replica**

Elasticsearch provides the ability to subdivide your index into multiple pieces called shards.Single instance of Luceneon a node can be primary or replica

* 1. **Installation & Configuration**

Elasticsearch requires Java 7 and above. So before you install Elasticsearch, please check your Java version first by running (and then install/upgrade accordingly if needed):

java -version

echo $JAVA\_HOME

Once we have Java set up, we can then download and run Elasticsearch. From the below link.

[www.elastic.co/downloads](http://www.elastic.co/downloads).

For each release, you have a choice among a zip or tar archive,  (Windows-zip)(Linux-tar)

Then extract it as follows (Windows users should unzip the zip package):

**Elasticsearch-2.1.1.zip.**

It will then create a bunch of files and folders in your current directory. We then go into the bin directory as follows:

* cd elasticsearch-2.1.1.bat

[2016-03-23 16:27:13,442][WARN ][bootstrap ] jvm uses the client vm, make sure to run `java` with the server vm for best performance by adding `-server` to the command line

[2016-03-23 16:27:13,827][INFO ][node ] [Wyatt Wingfoot] version[2.2.0], pid[7748], build[8ff36d1/2016-01-27T13:32:39Z]

[2016-03-23 16:27:13,827][INFO ][node ] [Wyatt Wingfoot] initializing ...

[2016-03-23 16:27:14,426][INFO ][plugins ] [Wyatt Wingfoot] modules [lang-expression, lang-groovy], plugins [], sites []

[2016-03-23 16:27:14,468][INFO ][env ] [Wyatt Wingfoot] using [1] data paths, mounts [[Data (D:)]], net usable\_space [281.2gb], net total\_space [367.8gb], spins? [unknown], types [NTFS]

[2016-03-23 16:27:14,469][INFO ][env ] [Wyatt Wingfoot] heap size [989.8mb], compressed ordinary object pointers [unknown]

[2016-03-23 16:27:16,687][INFO ][node ] [Wyatt Wingfoot] initialized

[2016-03-23 16:27:16,688][INFO ][node ] [Wyatt Wingfoot] starting ...

[2016-03-23 16:27:17,260][INFO ][transport ] [Wyatt Wingfoot] publish\_address {127.0.0.1:9300}, bound\_addresses {127.0.0.1:9300}, {[::1]:9300}

[2016-03-23 16:27:17,270][INFO ][discovery ] [Wyatt Wingfoot] elasticsearch/6WjfAZBfTsaDtCBGQ2NQew

[2016-03-23 16:27:21,345][INFO ][cluster.service ] [Wyatt Wingfoot] new\_master {Wyatt Wingfoot}{6WjfAZBfTsaDtCBGQ2NQew}{127.0.0.1}{127.0.0.1:9300}, reason: zen-disco-join(elected\_as\_master, [0] joins received)

[2016-03-23 16:27:21,402][INFO ][gateway ] [Wyatt Wingfoot] recovered [0] indices into cluster\_state

[2016-03-23 16:27:21,810][INFO ][http ] [Wyatt Wingfoot] publish\_address {127.0.0.1:9200}, bound\_addresses {127.0.0.1:9200}, {[::1]:9200}

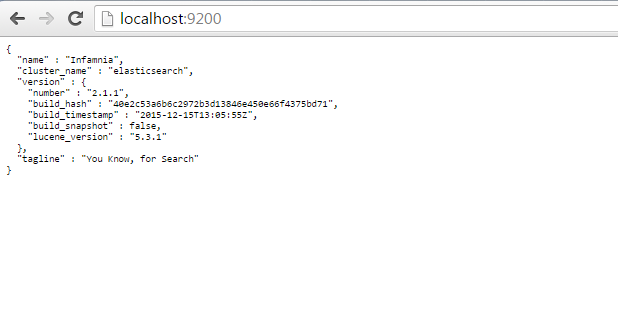
[2016-03-23 16:27:21,811][INFO ][node ] [Wyatt Wingfoot] started

Then extract it as follows (Windows users should unzip the zip package)

After installation. To check whether the elasticsearch is working.

Go to Browser type🡪 localhost:9200

You will see the below screen on the browser.



* 1. **Exploring Your Cluster**

**The REST API**

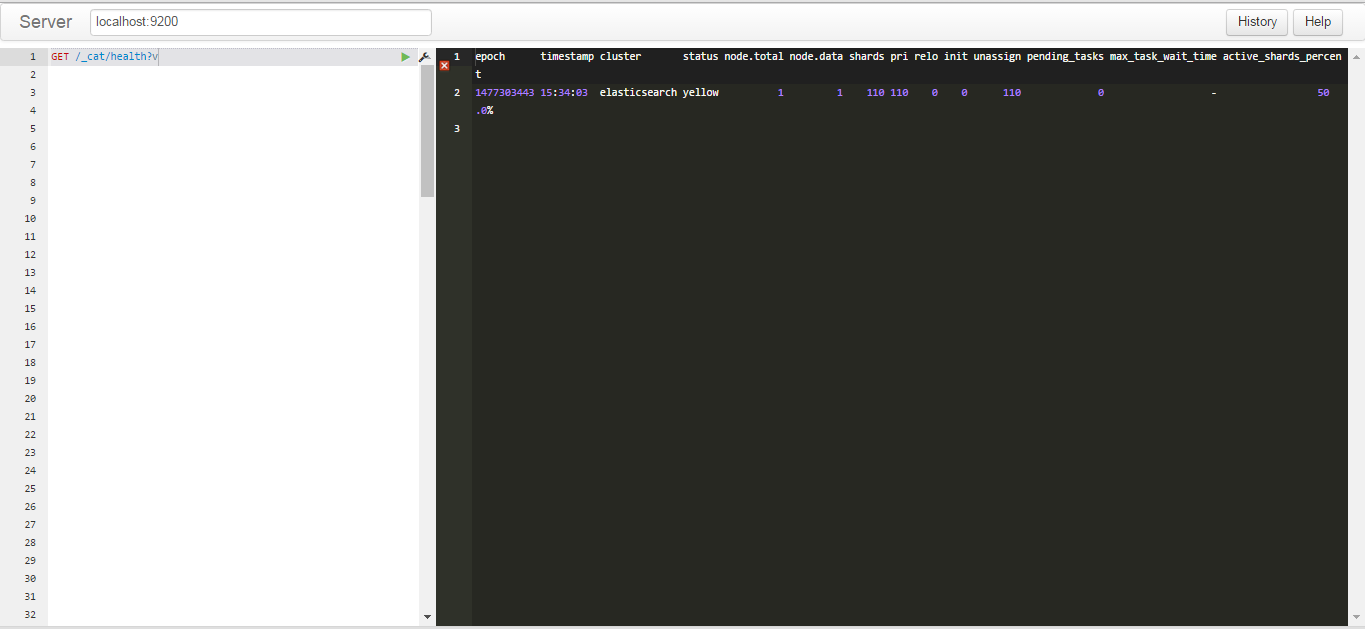
Now that we have our node (and cluster) up and running, the next step is to understand how to communicate with it. Fortunately, Elasticsearch provides a very comprehensive and powerful REST API that you can use to interact with your cluster. Check your cluster, node, and index health, status, and statistics.

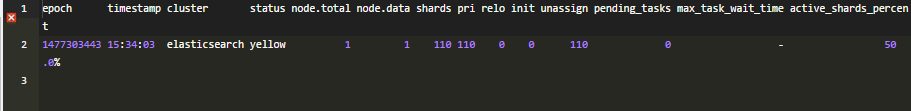
* We can use to see how our cluster is doing? By using curl Http/REST API calls.
* To do cluster health check by using “\_cat” API.

You can use CURL Commands or use SENSE plugin-chrome.

I used SENSE plugin which was user friendly.

GET /\_cat/health?v





**What is Document?**

It refers to the root object that is serialized into JSON and stored in Elasticsearch under a unique ID.

Document contains not only data, it contains Meta data.

* Index: An index like a database in a relational database. It stores data
* Type : The class of object that the document represents
* ID : Unique identifier of the Object

Sample data syntax:

curl -X<REST Verb> <Node>:<Port>/<Index>/<Type>/<ID>

**Index Operations**

**Creating index and inserting:**

<REST Verb>/<Index name>/<type>/<id>

PUT /customer/external/1

{

“name”:”moni”,

“age”:23

}

We can also create index without ID. The elastic search will automatically specify the index by itself.

**Update:**

POST customer/external/1/\_update?pretty' -d '

{

"doc": { "name": "Jane Doe" }

}

**DELETE:**

DELETE Customer

DELETE customer/external/1

**Search query:**

There are 2 basic ways to run searches:

* + - Search parameter through the REST request URI
    - Search through the REST Request body.

🡪POST /customer/\_search?pretty’\_d’

{

“query”:{match\_all:{}}

}

🡪POST /customer/\_search?pretty’\_d’

{

“query”:{match:{“name”: ”moni”}}

}

🡪POST /bank/\_search?pretty’\_d’

{

“query”:{

“query\_string”:{

“query”:{“moni”}

}

}}

🡪POST /bank/\_search?pretty’\_d’

{

“query”:{“bool”:{

“should”:[{“match\_phrase”:{“age”:40}}

{“match”:{“gender”:”m”}}

]}}}

**Batch Processing(BULK)**

It also provides the ability to perform all type of operations like index, update document in batches by using “\_bulk”

1. **we can add the data by adding records individualy.**

🡪POST /bank/\_bulk?pretty’\_d’

{

“update”:{“id”:1}}

{“doc”:{“name”:”john”}}

{“delete”:{“\_id”:”2”}

}

🡪PUT /employee/details/\_bulk?pretty' -d '

{"index":{"\_id":"21"}}

{"id":21,"address":{"area":"kochi","city":"kochi","country":"india","state":"kerala"},"name":"murugan","salary":23444}

{"index":{"\_id":"22"}}

{"id":22,"address":{"area":"kochi","city":"kochi","country":"india","state":"kerala"},"name":"james","salary":232444}

1. We can add whole file using Bulk API.

*🡪*POST /bank/account/\_bulk?pretty --data-binary @bank.json .

**There are different API available in elasticsearch.**

* + - Document API
    - Search API
    - Indices API,etc...

You can look API in depth at the below link.

https://www.elastic.co/guide/en/elasticsearch/reference/current/docs.html

******

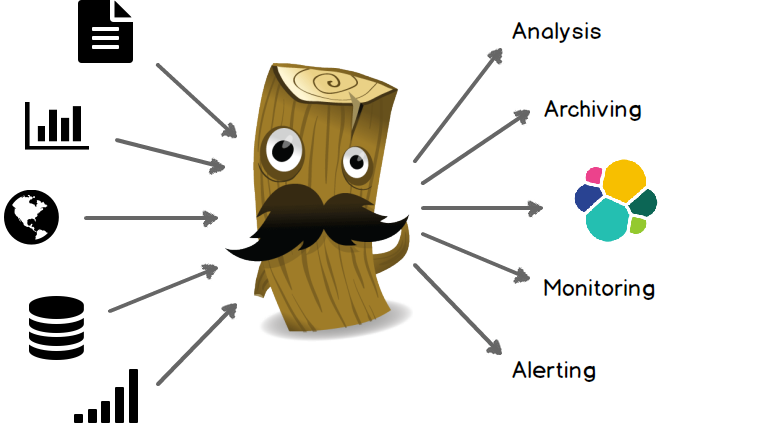
* 1. **Logstash**

Logstash is an open source data collection engine with real-time pipelining capabilities. Logstash can dynamically unify data from disparate sources and normalize the data into destinations of your choice.

* Logstash is a flexible, open source data collection, enrichment, and transportation pipeline.
* Logstash can dynamically unify data from disparate sources and normalize the data into destinations of your choice
* Logstash is designed to efficiently process a growing list of log, event, and unstructured data sources for distribution into a variety of outputs, including Elasticsearch.
* Developed by Jorden sissel.
* Jorden needed something that could handle a peak of 20000 messages per second

## *The Power of Logstash:*

* The ingestion workhorse for Elasticsearch and more.
* Horizontally scalable data processing pipeline with strong Elasticsearch and Kibana synergy
* Pluggable pipeline architecture
* Over 200 plugins available, plus the flexibility of creating and contributing your own



***Advantage of using Logstash:***

* It is easy to setup, scalable and easy to extend.
* Easy integration with connectors to common infrastructure.
* Data can collect from any source.

Ex: Mobile devices, social data, ecommerce, log data, News articles, Web trend data

* It handles all types of logging data

Ex: apache logs, syslog, windows event logs

* Transform HTTP Requests into events (Web)

Ex: Consume from web service firehoses like Twitter, Webhook support for GitHub, JIRA

* Data stores & Streams

Ex: Relational database or No SQL store with JDBC, streams from messaging queues like Kafka, RobiitMQ, ZeroMQ.

# Getting Started with Logstash

This section guides you through the process of installing Logstash and verifying that everything is running properly.

* [Installing Logstash](https://www.elastic.co/guide/en/logstash/current/installing-logstash.html)
* [Stashing Your First Event: Basic Logstash Example](https://www.elastic.co/guide/en/logstash/current/first-event.html)
* [Parsing Logs with Logstash](https://www.elastic.co/guide/en/logstash/current/advanced-pipeline.html)
* [Stalled Shutdown Detection](https://www.elastic.co/guide/en/logstash/current/stalled-shutdown.html)
* [Logstash Processing Pipeline](https://www.elastic.co/guide/en/logstash/current/pipeline.html)

[**Installing Logstash**](https://www.elastic.co/guide/en/logstash/current/installing-logstash.html)**:**

Download the logstash-2.1.1 version which is compatible with elasticsearch-2.1.1

🡪Windows(logstash-2.1.1.zip)-unzip it

We can install as service or we can run the logstash.bat file when we need to use logstash.

*1***. installing logstash using .bat file:**

Before run the batch file need to do configuration setting.

Configuration file should have

The Logstash event processing pipeline has three stages:

* Input – It consumes the data from a source & Inputs generate events.
* Filter – Modify the data as you specify. It is a optional plugin
* Output – Write the data to a destination or ship them else where

Input Plugin

Filter Plugin

Output Plugin

**Logstash Instance**

## *Inputs:*

Some of the more commonly-used inputs are:

* File - Reads from a file on the filesystem, much like the UNIX command tail -0F.
* Syslog - listens on the well-known port 514 for syslog messages and parses according to the RFC3164 format.
* Beats - Process events sent by Filebeat
* Redis - Reads from a redis server, using both redis channels and redis lists.
* Twitter - Reads events from the Twitter Streaming API
* JDBC - This plugin was created as a way to ingest data in any database with a JDBC interface into Logstash
* Stream - This input will read events from a Kafka topic

## *Filters*

Filters are intermediary processing devices in the Logstash pipeline:

* grok - Add information about geographical location of IP addresses.

parse unstructured log data into something structured and queryable.

* mutate - Perform general transformations on event fields.

We can rename, remove, replace, and modify fields in your events.

* drop - Drop an event completely

Ex: debug events

* clone - Make a copy of an event, possibly adding or removing fields.
* geoip - Reads events from the Twitter Streaming API

Note : Combine filters with conditionals to perform an action on an event if it meets certain criteria.

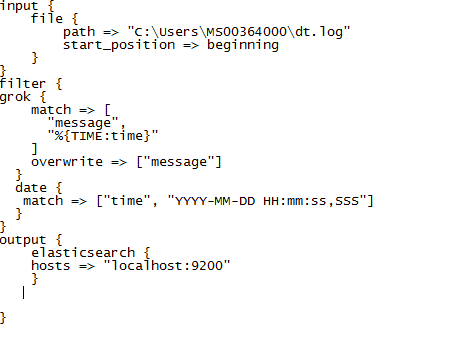
## *Outputs*

Outputs are the final phase of the Logstash pipeline. An event can pass through multiple outputs, but once all output processing is complete, the event has finished its execution.

Commonly used outputs:

* File - Write event data to a file on disk.
* elasticsearch - Send event data to Elasticsearch.
* graphite - Send event data to graphite.
* statsd - Send event data to statsd, a service that "listens for statistics, like counters and timers.

***Logstash Configuration:***



In this configuration file the input plugin we are specifying the log file path whrer the data is stored in filter we are parsing the data and in output plugin we mention the destination location where here it is elasticsearch.

Save it as pipeline.conf file in bin of the logstash.

## *Starting Logstash*

Now you can start Logstash. Use the command that works with your system. If you installed Logstash, make sure the config file is in the expected directory.and sun the command.

bin\logstash.bat -f pipeline.conf –configtest

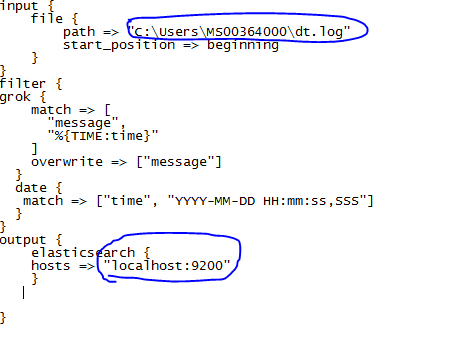
The logstash will read the configuration file pipeline.conf

Input plugin🡪read the data from the log file(dt-log) at the specified path”

“C:\USERS\MS00364000\dt.log”

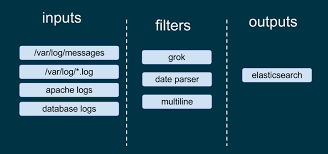
Filter plugin🡪parse(filters)the data and convert to json.

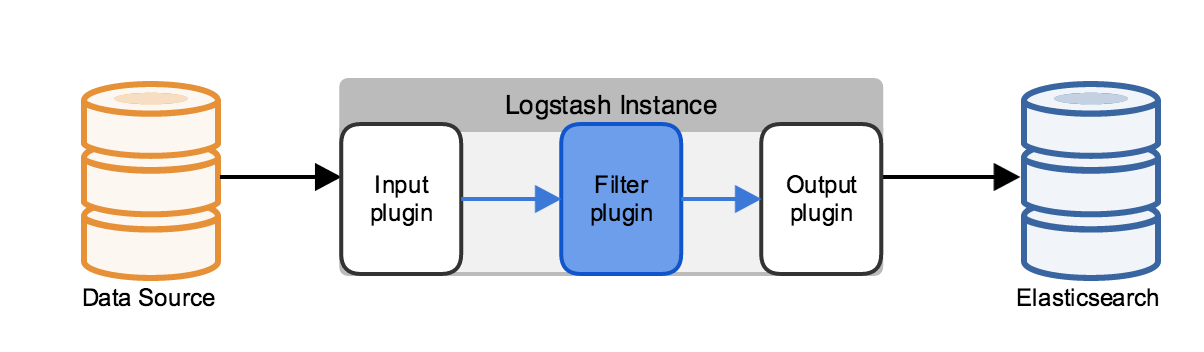
Output plugin🡪Store the data to elasticsearch.



The data will be added to the index as logstash-date(insert date).eg. logstash-2016.09.30

**Logstash Deployment**

****

****

****

* 1. **Kibana**
* It is an open source analytics and visualization platform designed to work with Elasticsearch.
* You can easily perform advanced data analysis and visualize your data in a variety of charts, tables, and maps
* Makes it easy to understand large volumes of data.
* Kibana makes it easy to understand large volumes of data.
* It enables to create and share dynamic dashboards that display changes to Elasticsearch queris.
* No code, No infrastructure required.
  + 1. **Installation & Configuration**

You can install Kibana and start exploring your Elasticsearch indices in minutes. All you need is:

Elasticsearch 2.4 or later.

**To get Kibana up and running:**

To install and start Kibana:

1. Download the [Kibana 4 binary package](https://www.elastic.co/downloads/kibana" \t "_top) for your platform.
2. Extract the .zip or tar.gz archive file.
3. If you’re upgrading, migrate any configuration changes from the previous kibana.yml to the new version.
4. Install Kibana plugins (optional).
5. Run Kibana from the install directory: bin/kibana (Linux/MacOSX) or bin\kibana.bat(Windows).

That’s it! Kibana is now running on port 5601.

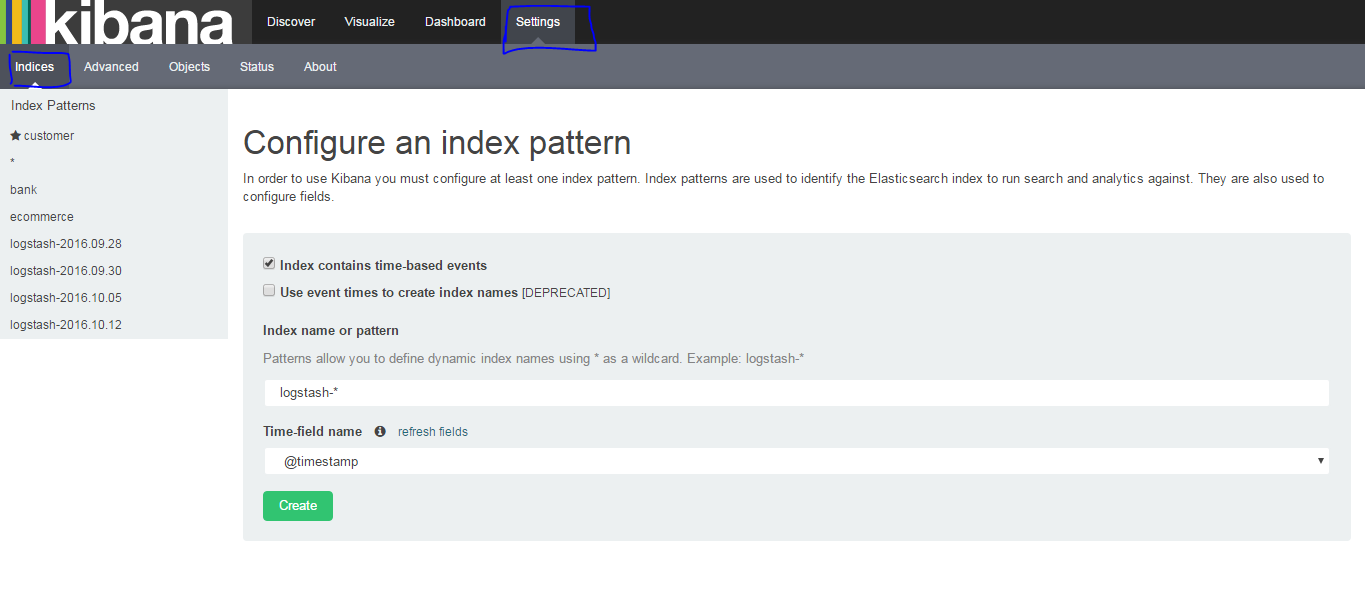
* + 1. **Connect Kibana with Elasticsearch**

Before you can start using Kibana, you need to tell it which Elasticsearch indices you want to explore. The first time you access Kibana, you are prompted to define an *index pattern* that matches the name of one or more of your indices. That’s it. That’s all you need to configure to start using Kibana. You can add index patterns at any time from the [Settings tab](https://www.elastic.co/guide/en/kibana/4.6/settings.html#settings-create-pattern).

By default, Kibana connects to the Elasticsearch instance running on localhost.To connect to a different Elasticsearch instance, modify the Elasticsearch URL in the kibana.yml configuration file and restart Kibana. For information about using Kibana with your production nodes, see Using Kibana in a Production Environment.

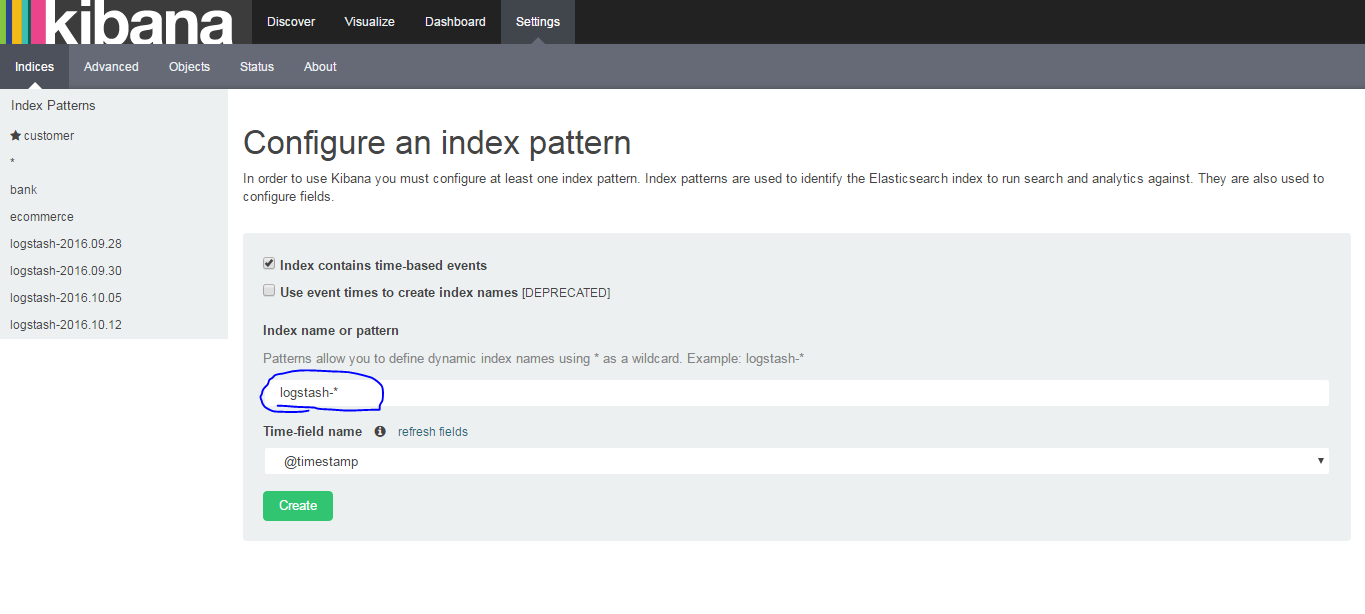
To configure the Elasticsearch indices you want to access with Kibana:

1. Point your browser at port 5601 to access the Kibana UI. For example, localhost:5601 orhttp://YOURDOMAIN.com:5601



You will see the window as above.

Now need to configure the index to kibana.



Give the index name to integrate to kibana.

1. Specify an index pattern that matches the name of one or more of your Elasticsearch indices. By default, Kibana guesses that you’re working with data being fed into Elasticsearch by Logstash. If that’s the case, you can use the default logstash-\* as your index pattern.
2. Select the index field that contains the timestamp that you want to use to perform time-based comparisons. Kibana reads the index mapping to list all of the fields that contain a timestamp. If your index doesn’t have time-based data, disable the **Index contains time-based events** option.
3. Click Create to add the index pattern. This first pattern is automatically configured as the default. When you have more than one index pattern, you can designate which one to use as the default from Settings > Indices.
   * 1. **Kibana and Elasticsearch Dynamic Mapping**

By default, Elasticsearch enables [dynamic mapping](http://www.elastic.co/guide/en/elasticsearch/reference/2.4/dynamic-mapping.html) for fields. Kibana needs dynamic mapping to use fields in visualizations correctly, as well as to manage the .kibana index where saved searches, visualizations, and dashboards are stored.

If the dynamic mapping is disabled.You also need to manually enable dynamic mapping for the .kibana index.

The following procedure assumes that the .kibana index does not already exist in Elasticsearch and that the index.mapper.dynamic setting in elasticsearch.yml is set to false:

1. Start Elasticsearch.
2. Create the .kibana index with dynamic mapping enabled just for that index:

PUT .kibana

{

"index.mapper.dynamic": true

}

Create Dashboard by 5 steps

* Loaded sample data set into your Elasticsearch installation.
* Define index pattern by using Settings tab
* Use the Discover functionality to explore your data
* Set up some Visualizations to graphically represent your data
* Assembled visualizations into a Dashboard

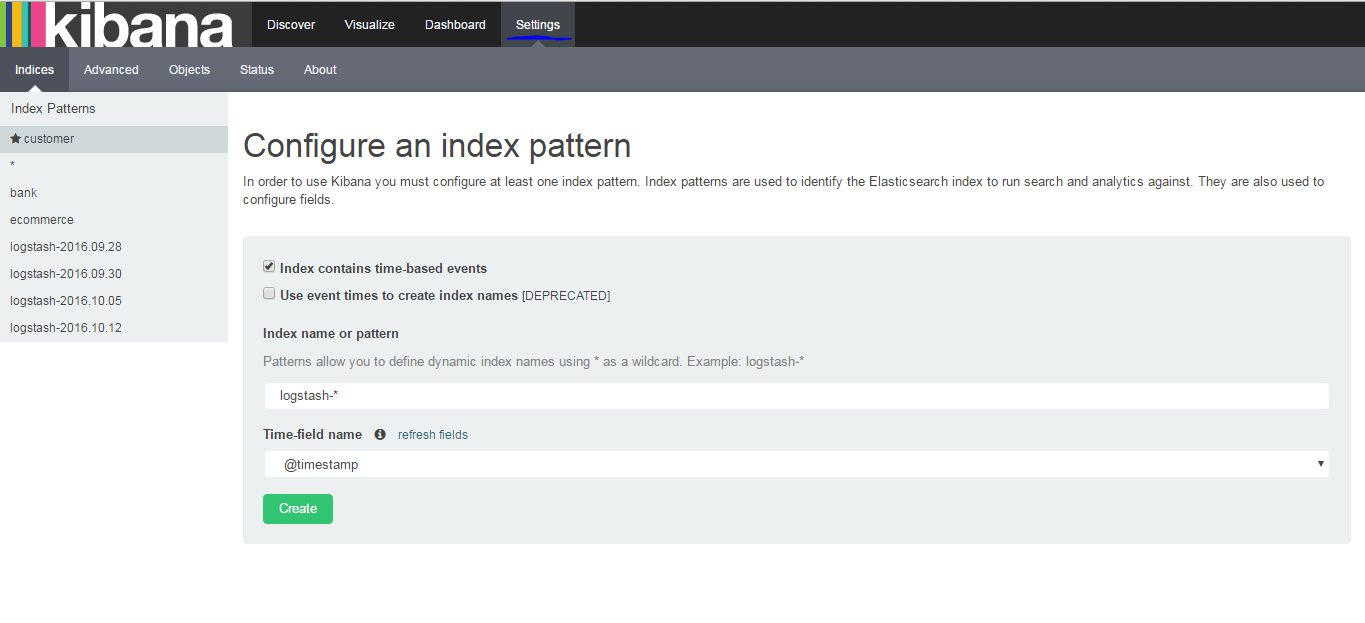
1. **Loaded sample data set into your Elasticsearch installation:**

We can load the data to elasticsearch using bulk API or logstash.that we have seen in the elasticsearch.

1. Define index pattern by using Settings tab

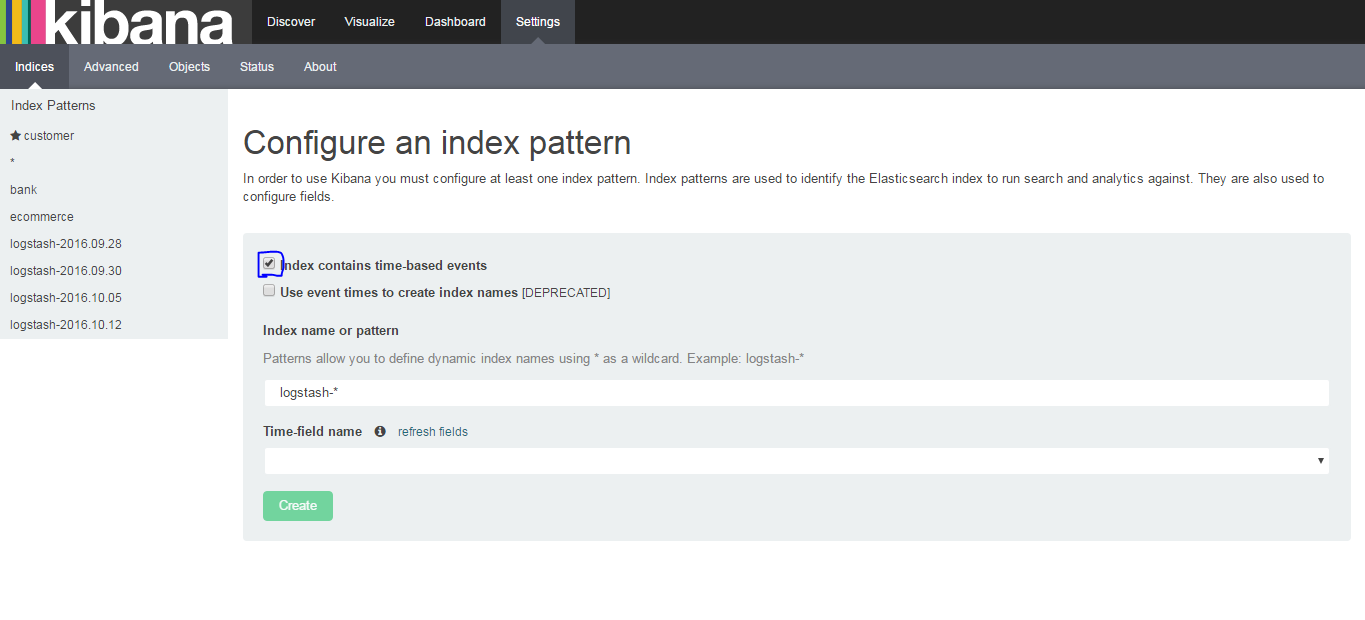
**Step:1**

We need to set the index or add the index to the kibana to visualize.so for that we need to add the index which is in elasticsearch.



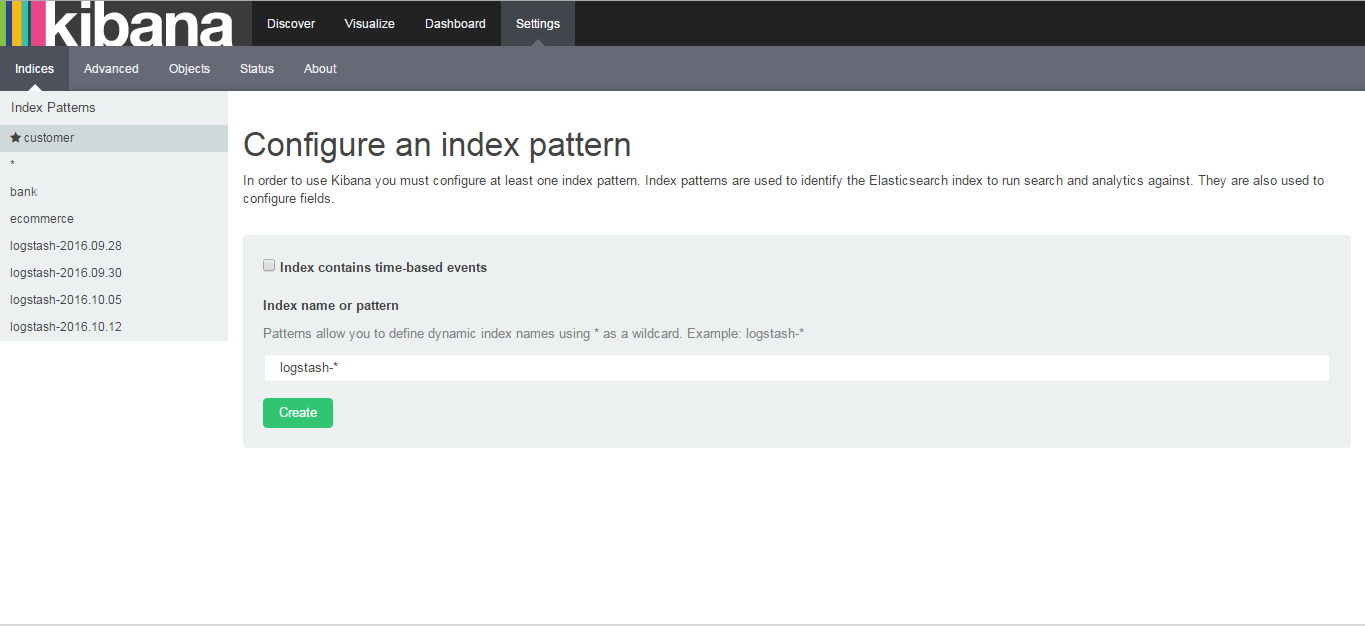
**Step:2**

If the index contain the time base data select the check box (index contain time-basec event) if not uncheck it.

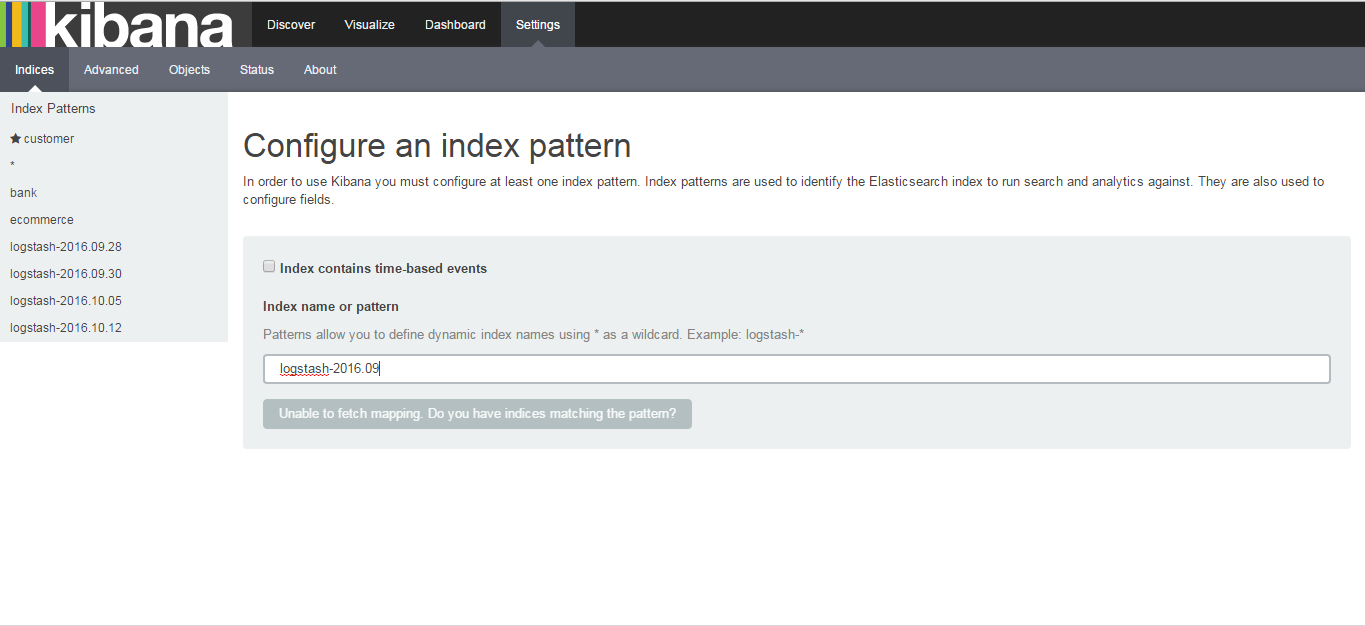


Step:3

Now type the index name which is in elasticsearch and which you need to integrate to kibana.

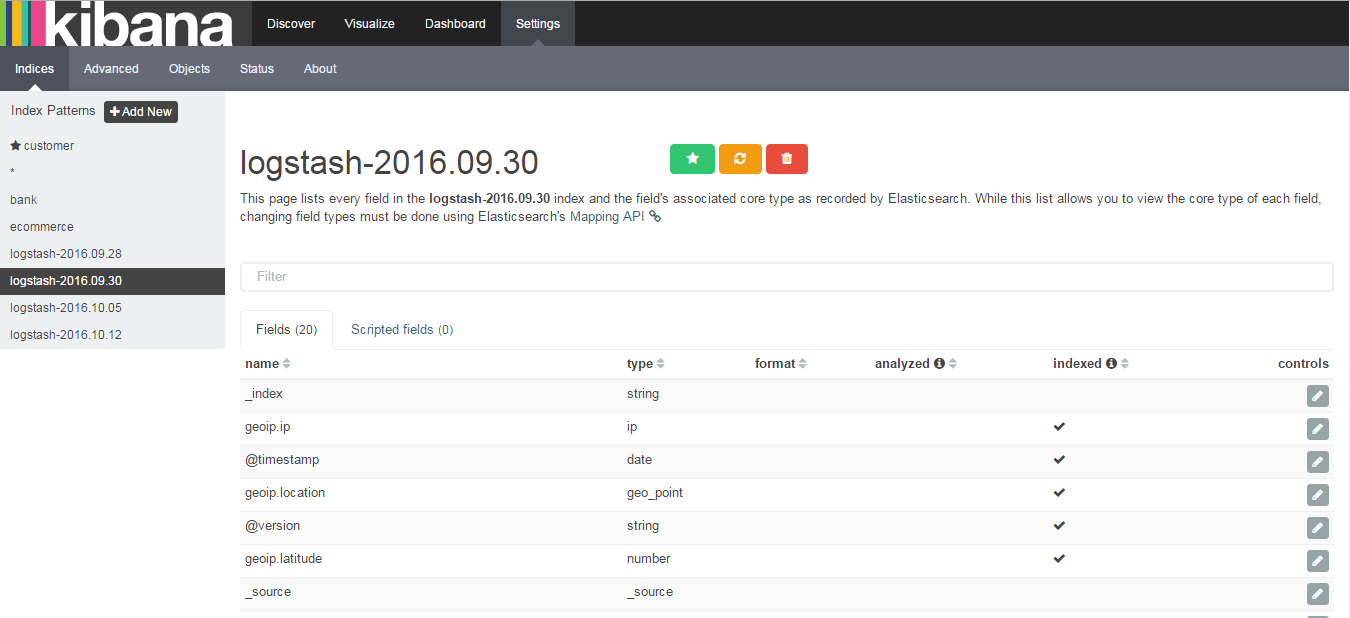


If you did not give the correct name then you cannot create the index in kibana.as shown below



Step:4

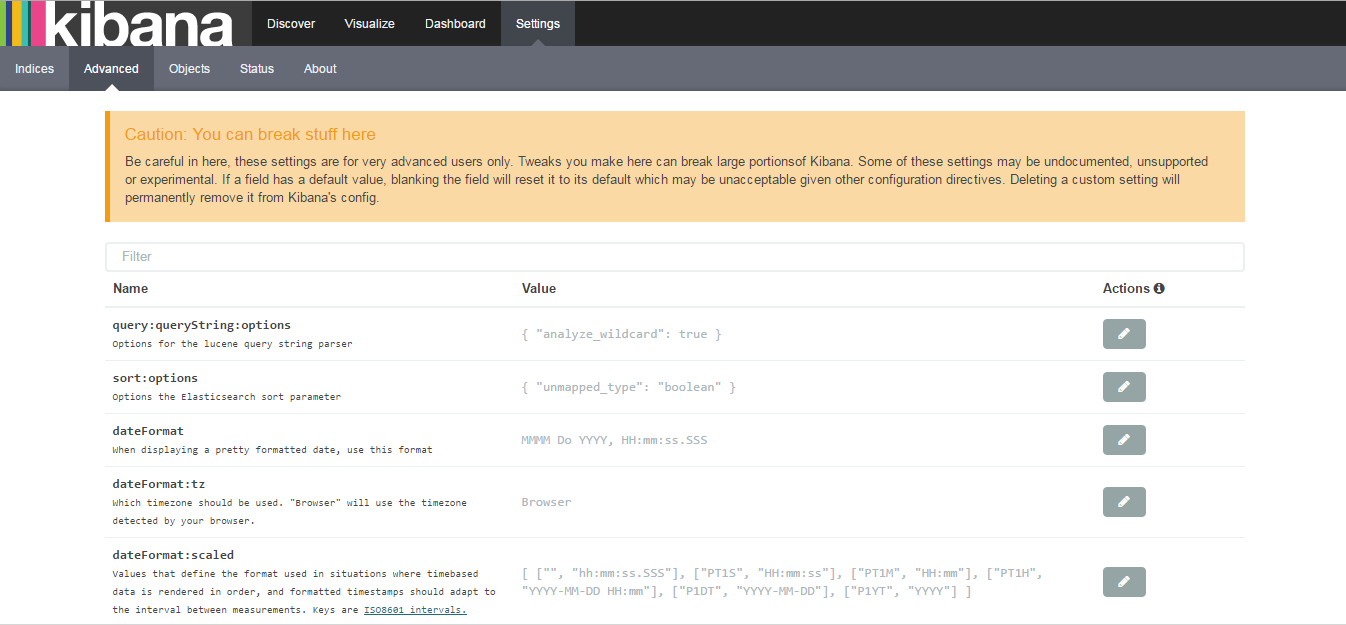
Give the correct index name and create the index.after creating you will see the below window.



So the index is added to kibana with the displayed field name.as shown above.

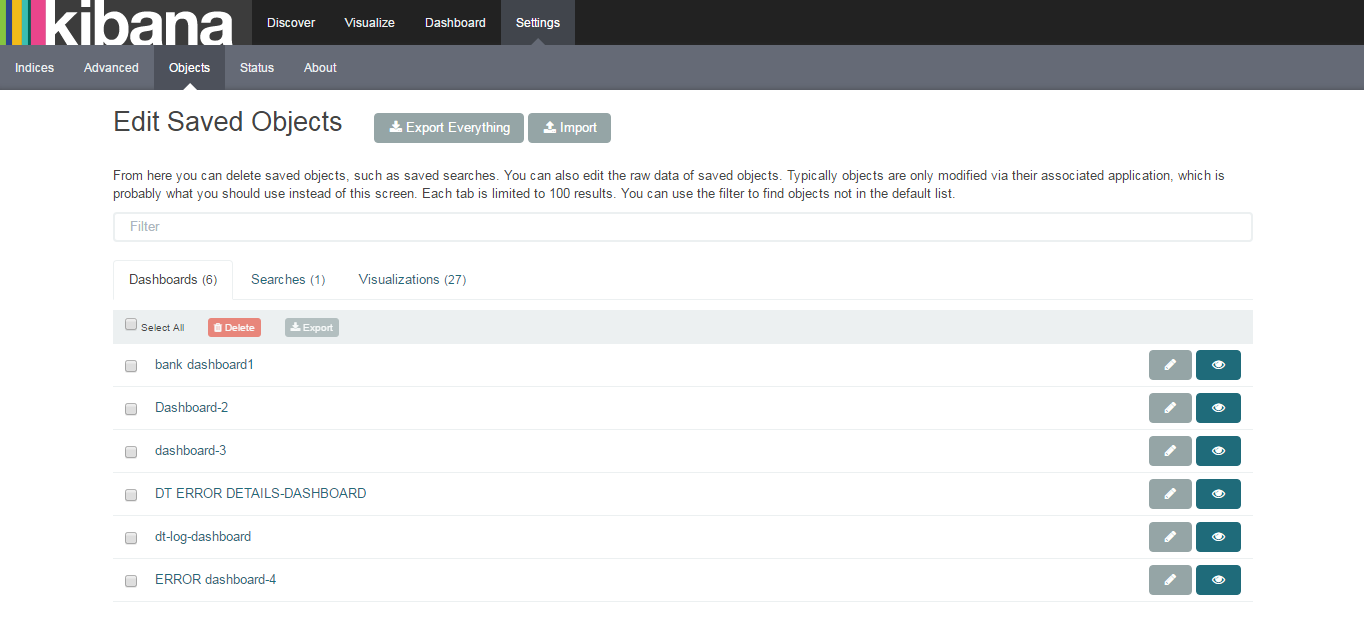
Step:5

Don’t do anything unless you are an expert in the Advanced tab.



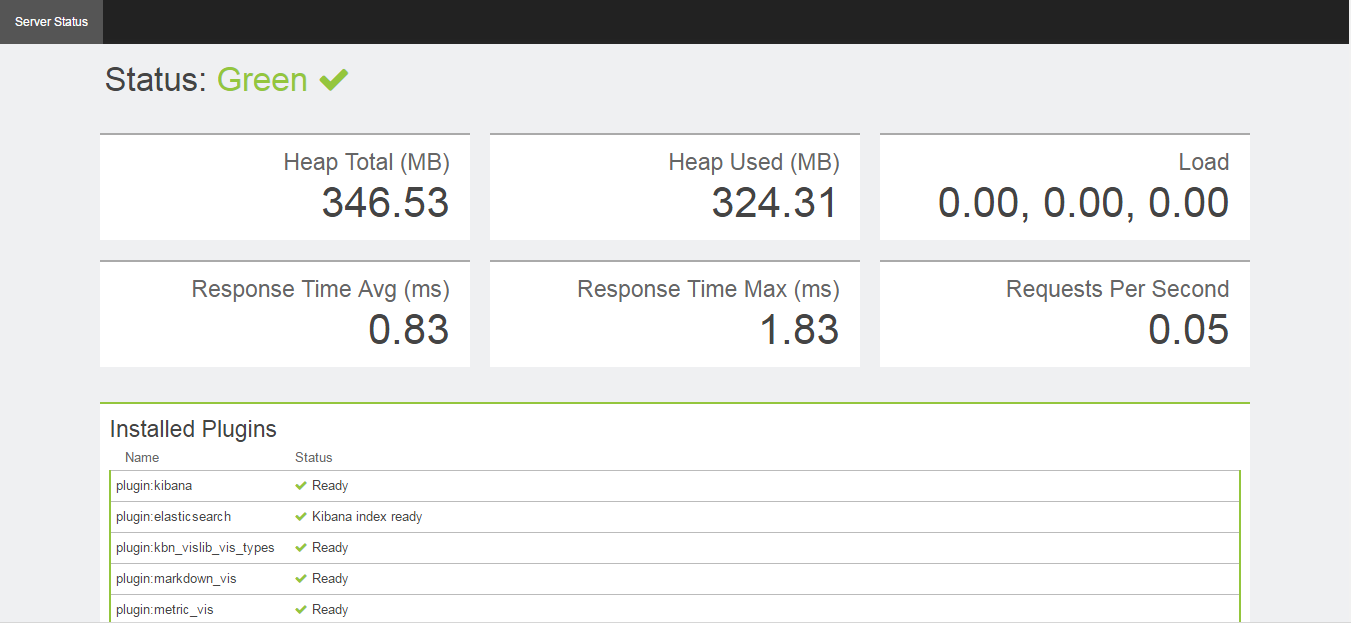
Step:6

In options tab you can see the dashboard, visualization and search which you have created for the all the index. As below.



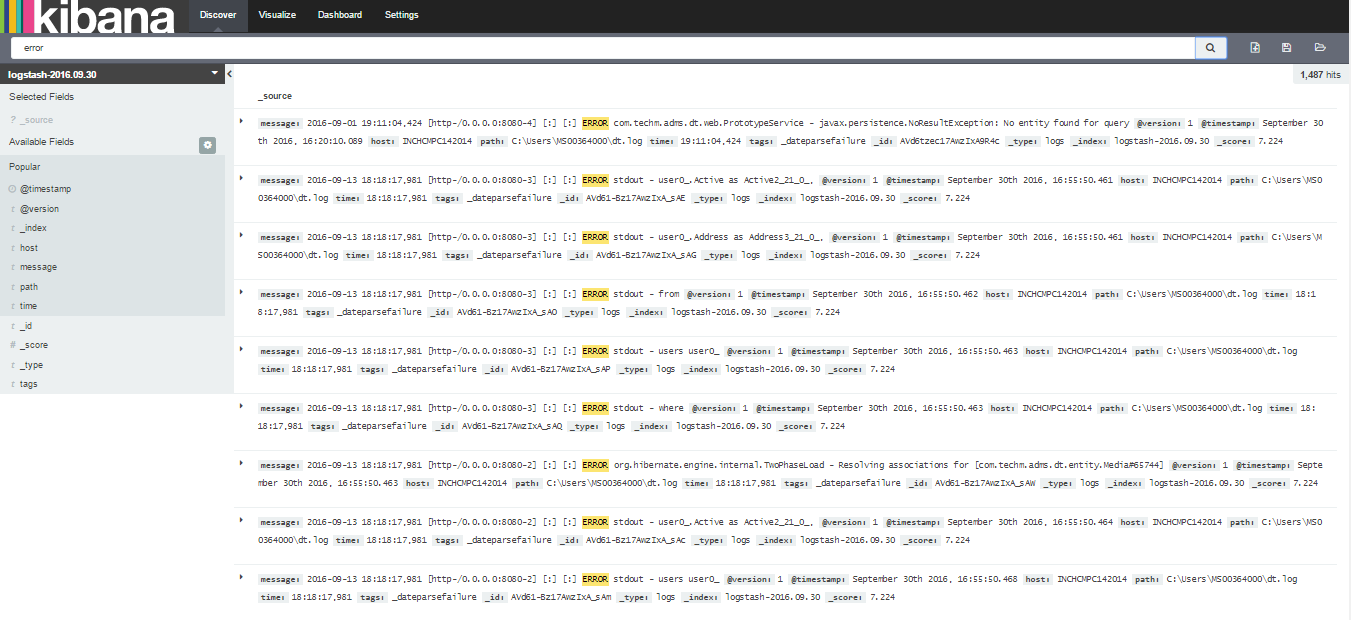
Step:7

On status tab you will be able to see the status of the kibana like working condition,memory.



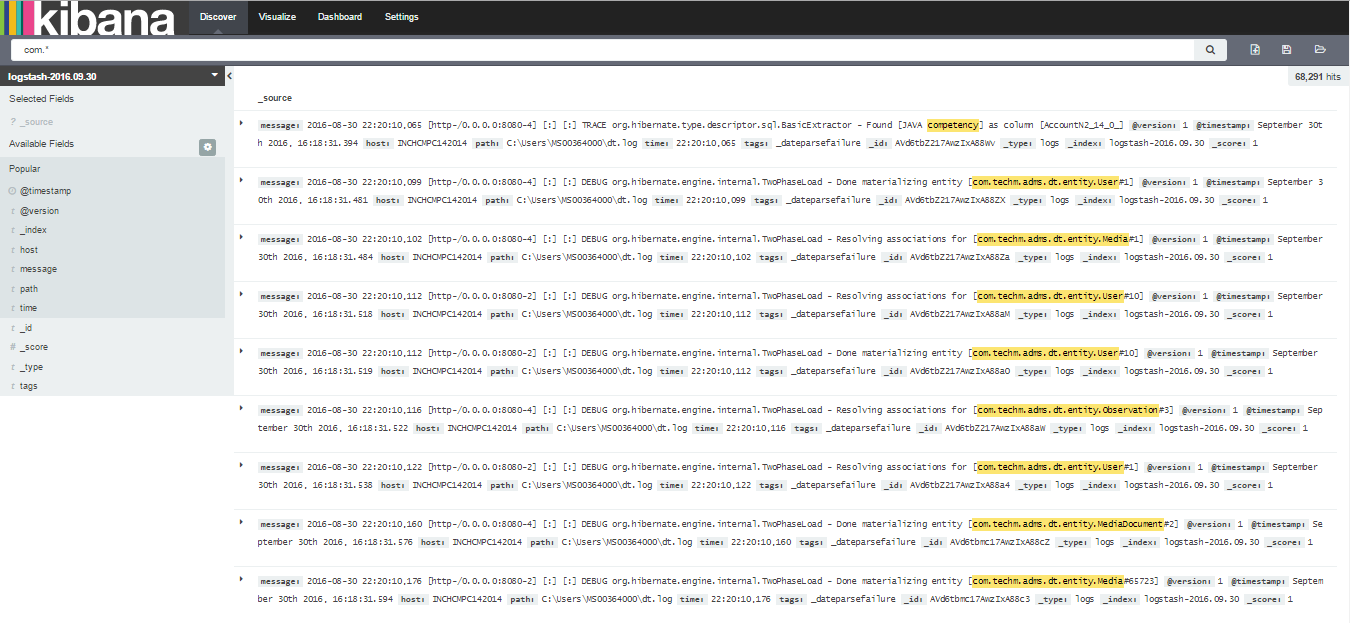
1. Use the **Discover** functionality to explore your data

We can search data using the discover tab. In search tab we can type the text which we want to search in the index.



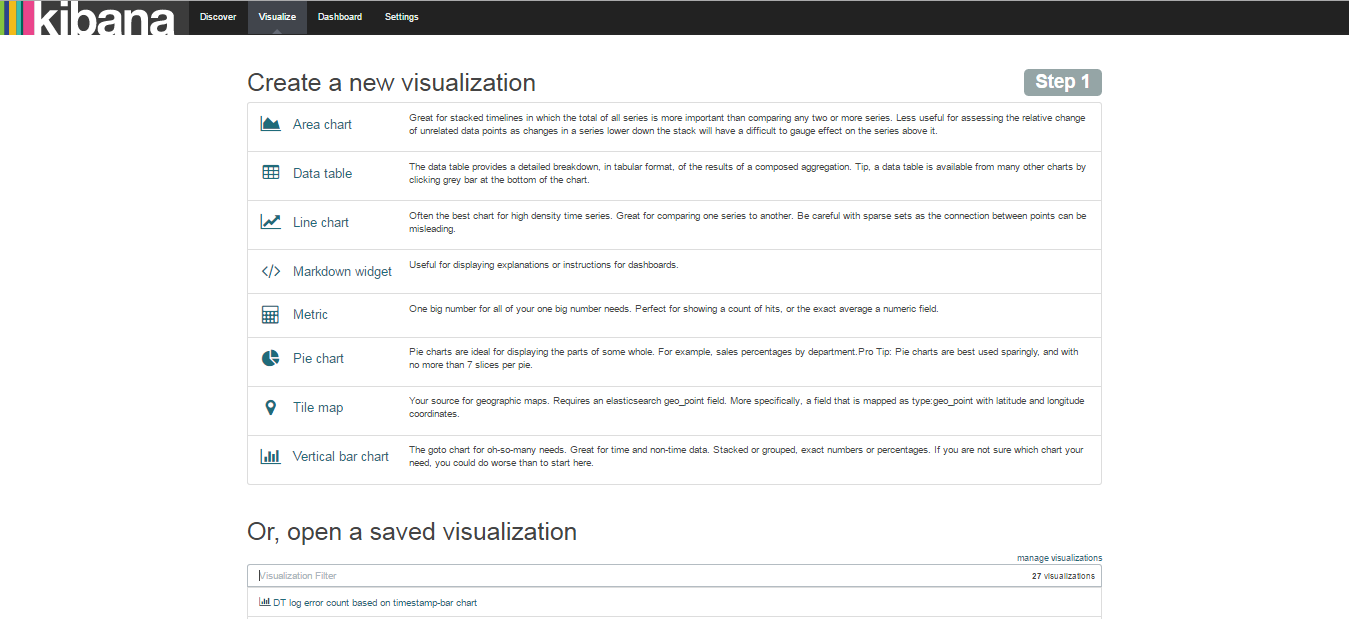
Example: error (need to search the log contain error) as above.

We can also search as below like (com.\*) so it will search the log file which has the text com\*



1. Set up some Visualizations to graphically represent your data

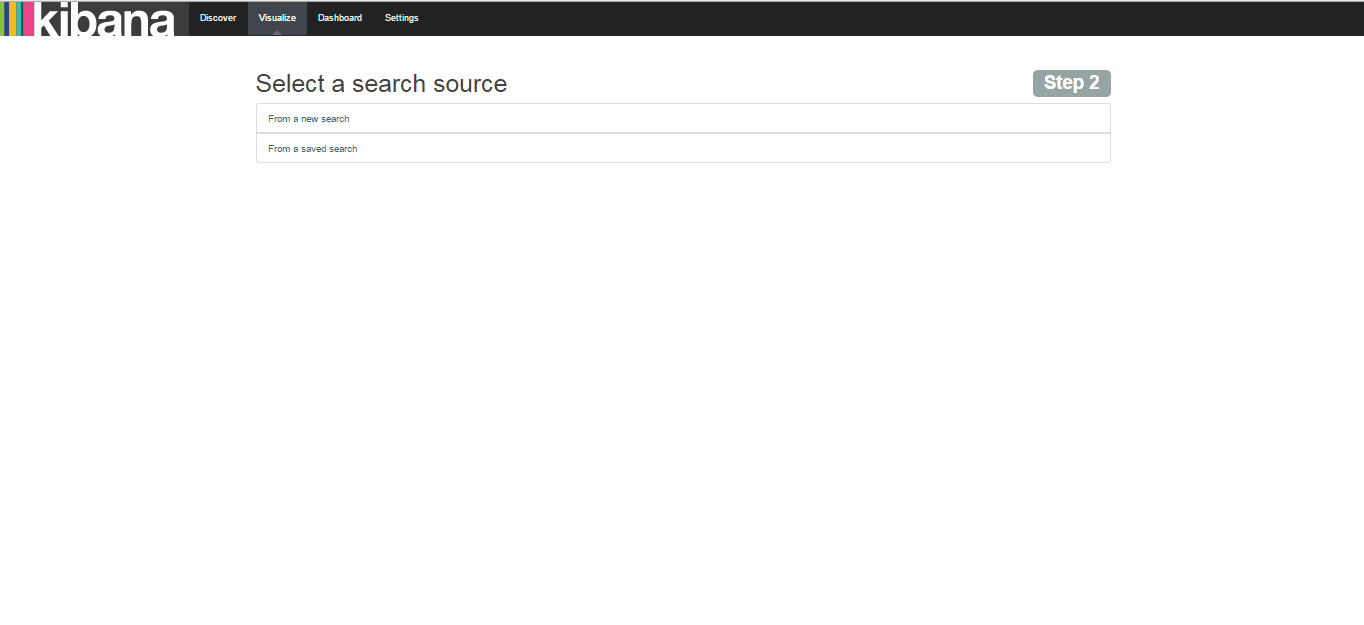
We can create visualization to the data by clicking the the visualize tab.



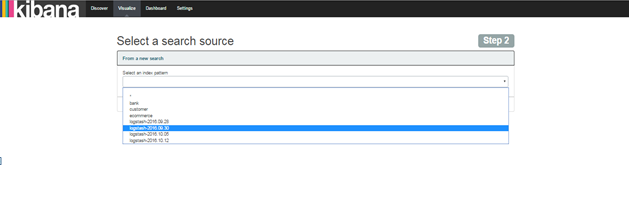
There are number of visualization that we can create for the data like line, pie, bar, etc...

Step:1(pie chart)

Now we will see how to create the pie chart first. So select the pie chart icon on create new visualization.then you will see the below window.

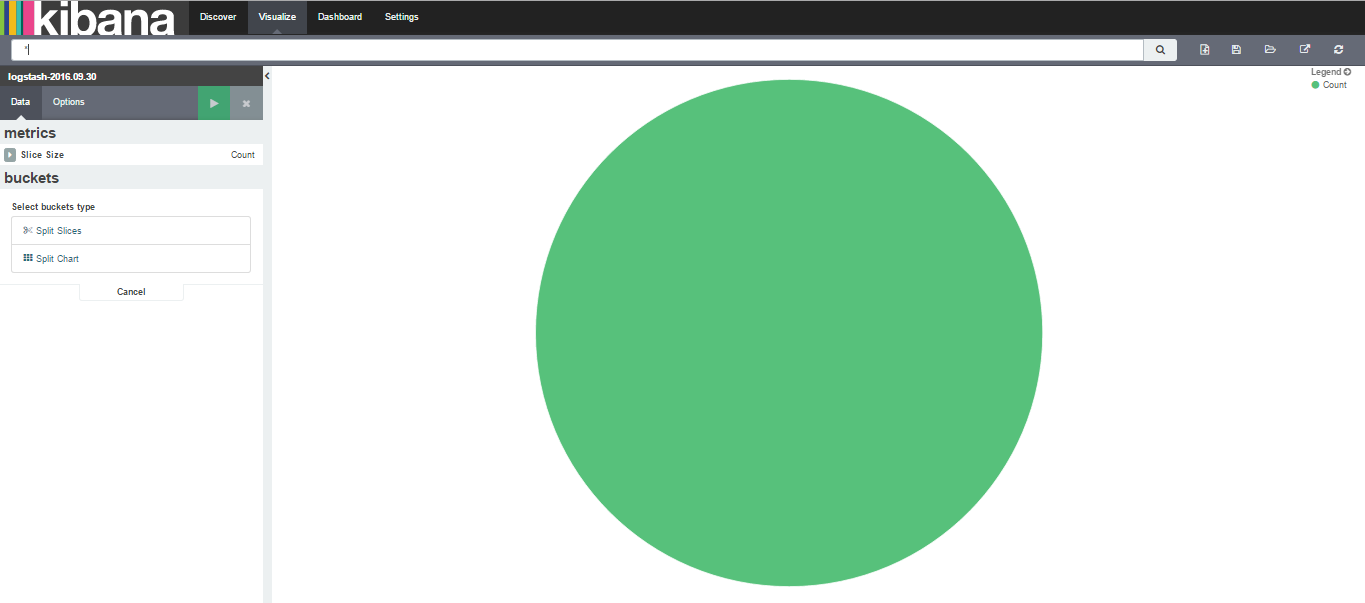


Select (from a new search) for new visualization. Then you will see the below window.

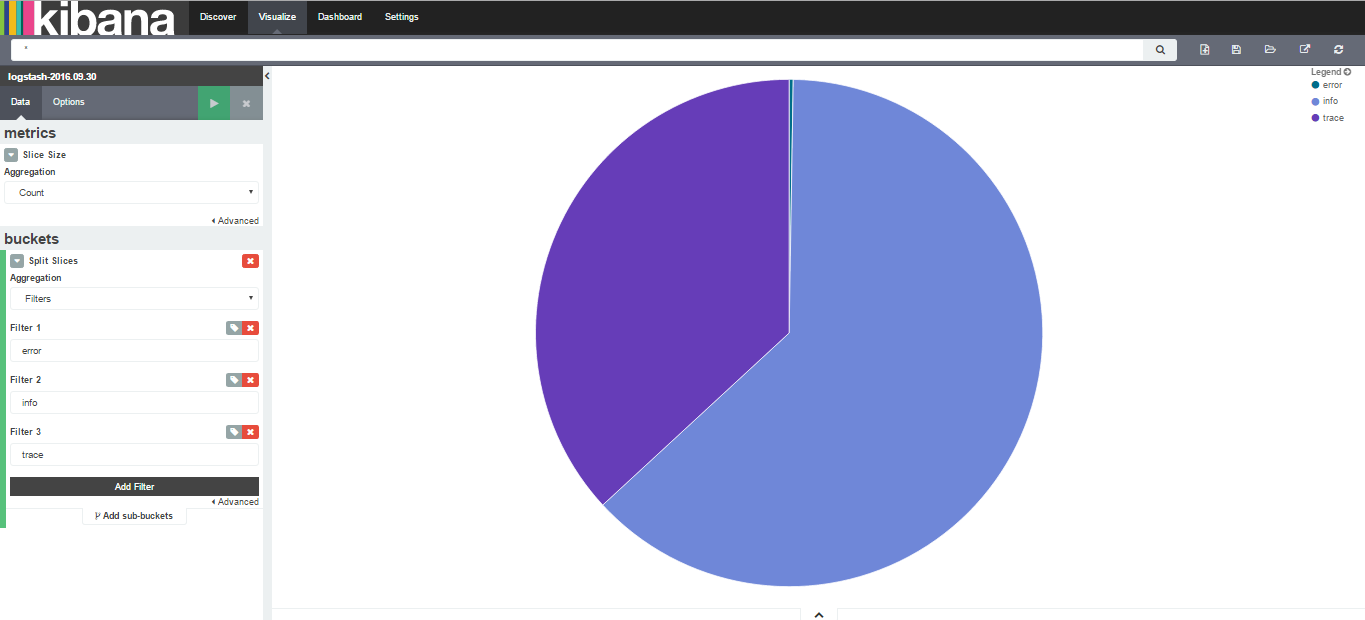


Select the index for which you need to create visualization.

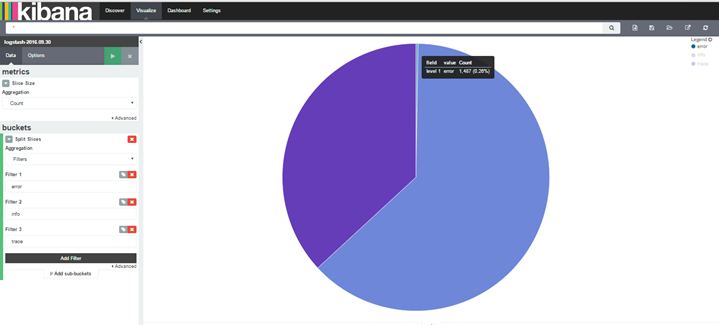
Now you will see the



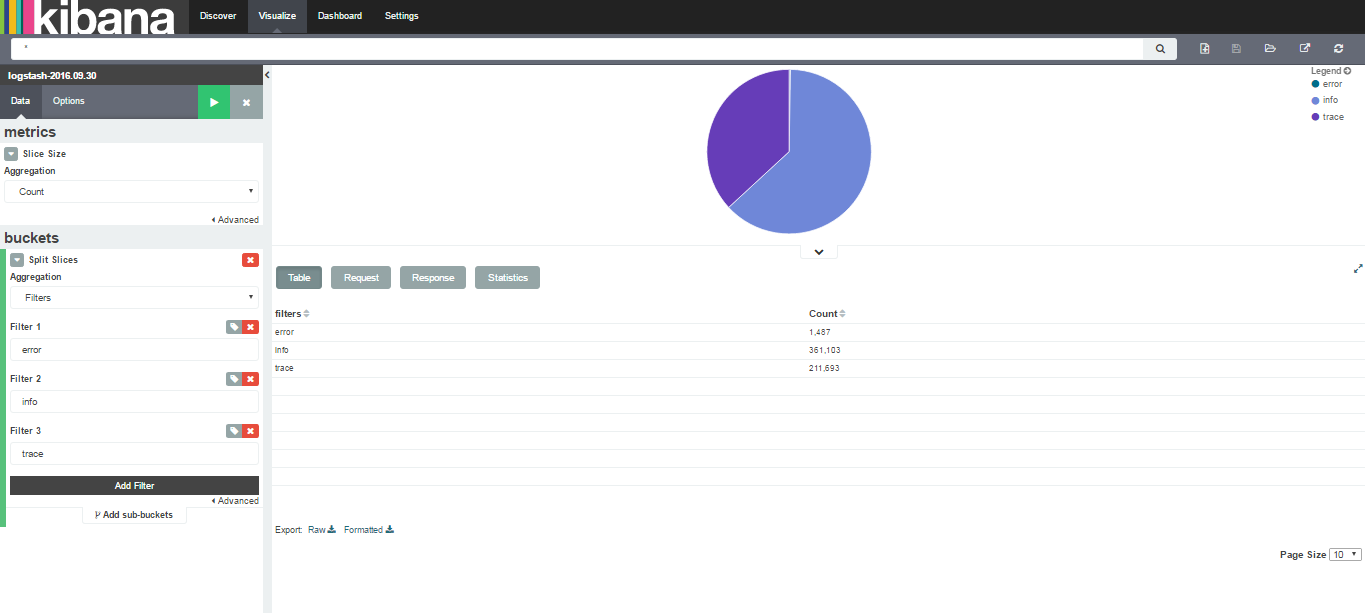
Now you can do the changes to get the pie chart you require by split slices. Here I have did for count on error, trace and info for the log data.by selecting the filter from the aggregation drop down.



We can see the count of error when we move over the pic chart as shown below.

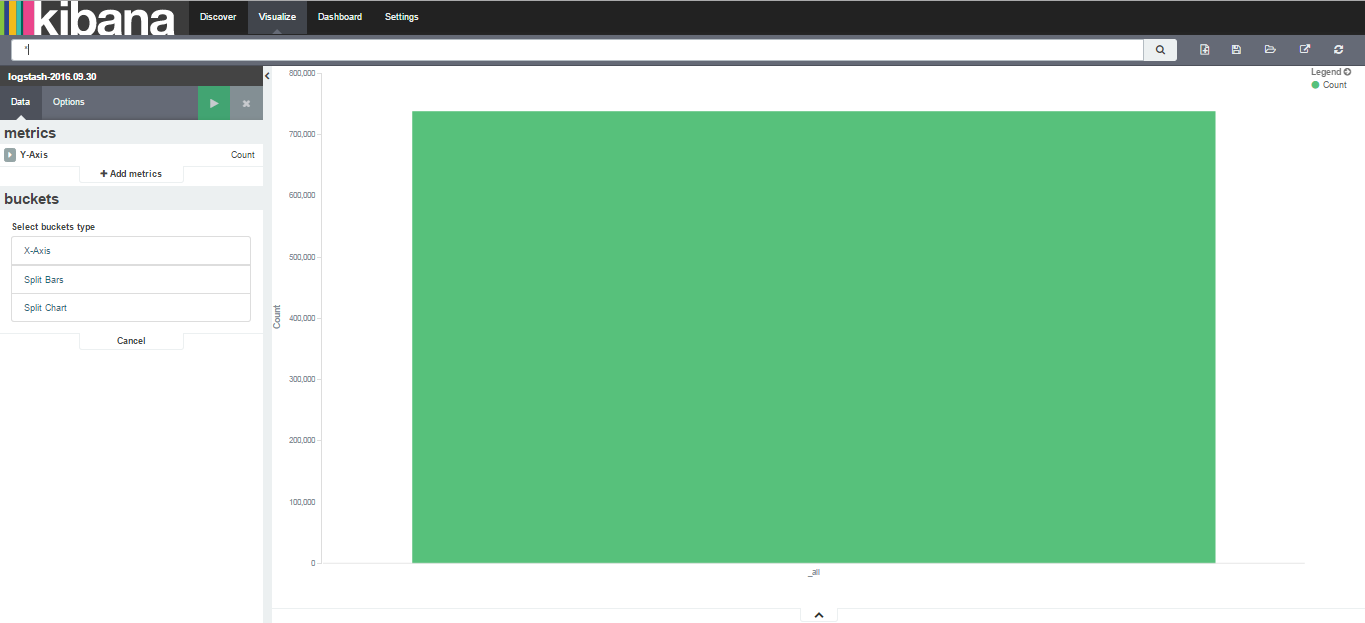


The count number can be seen by clicking the down arrow below the pie chart.



Step: 2(bar chart)

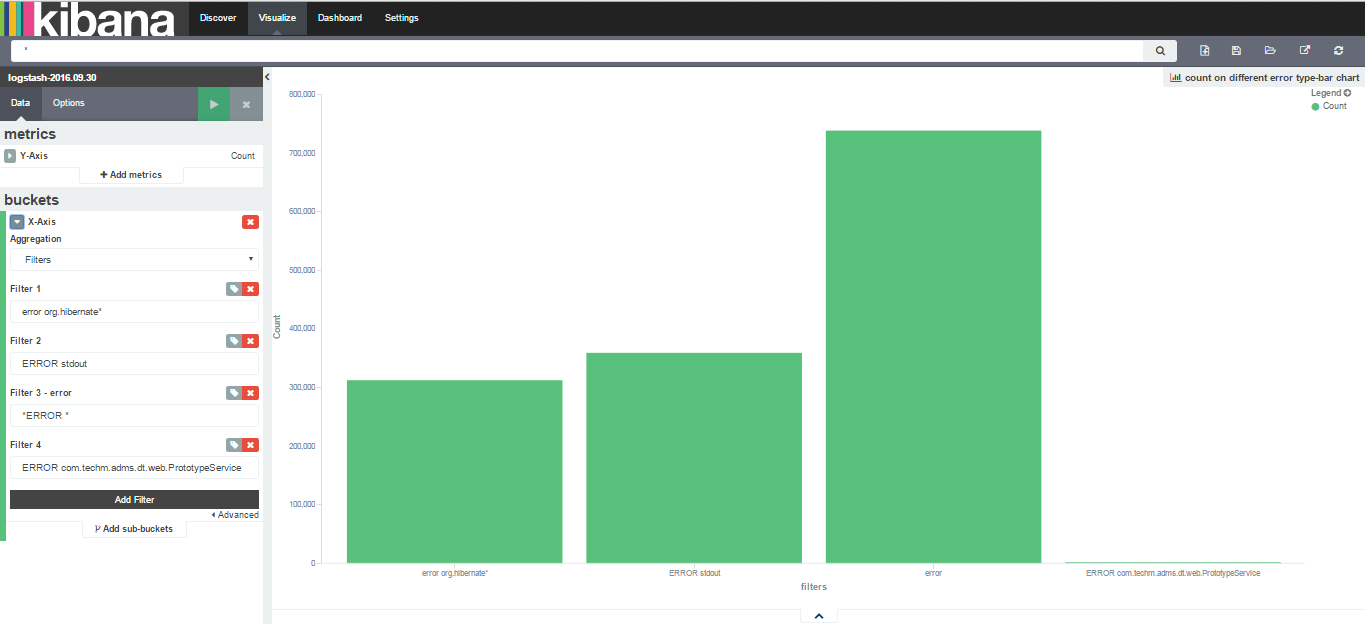
We can also do bar chart on the similar fashion. Select he bar chart from the create visualization from visualize tab and select the index for which you want to create visualization.



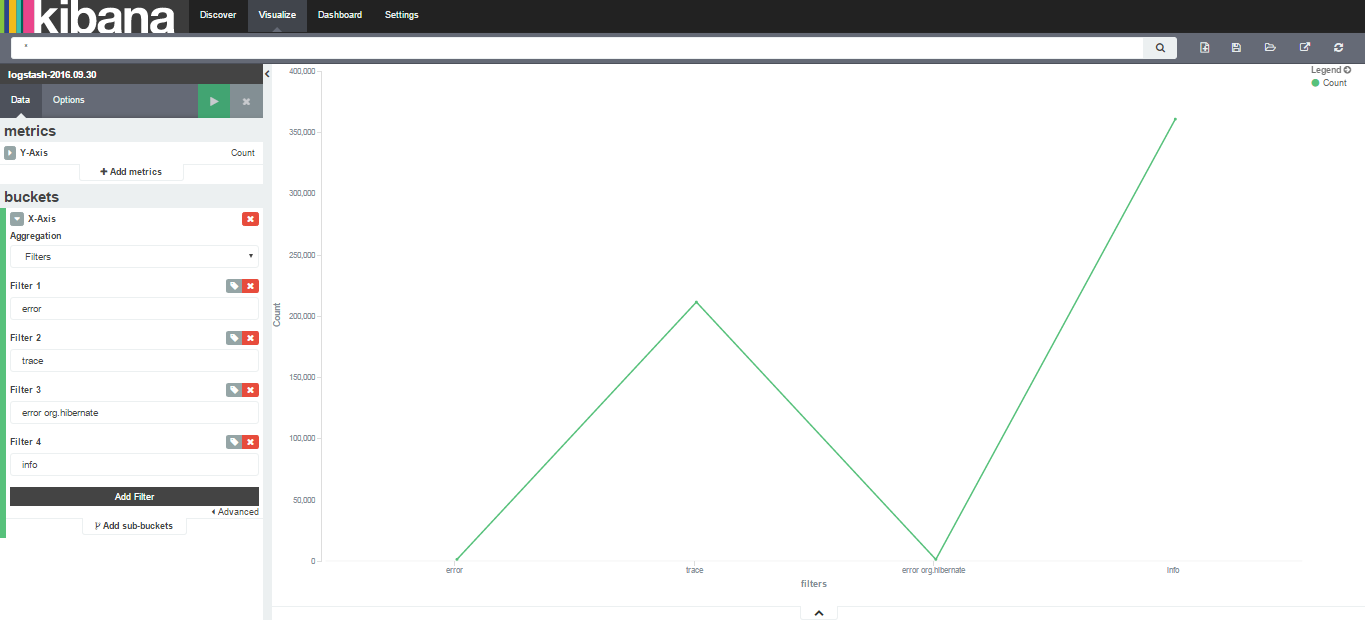
Then you can filter the data by doing the following

X axis🡪Aggregation🡪filter then type the data which you can to filter

Eg: error, trace, hibernate error etc…

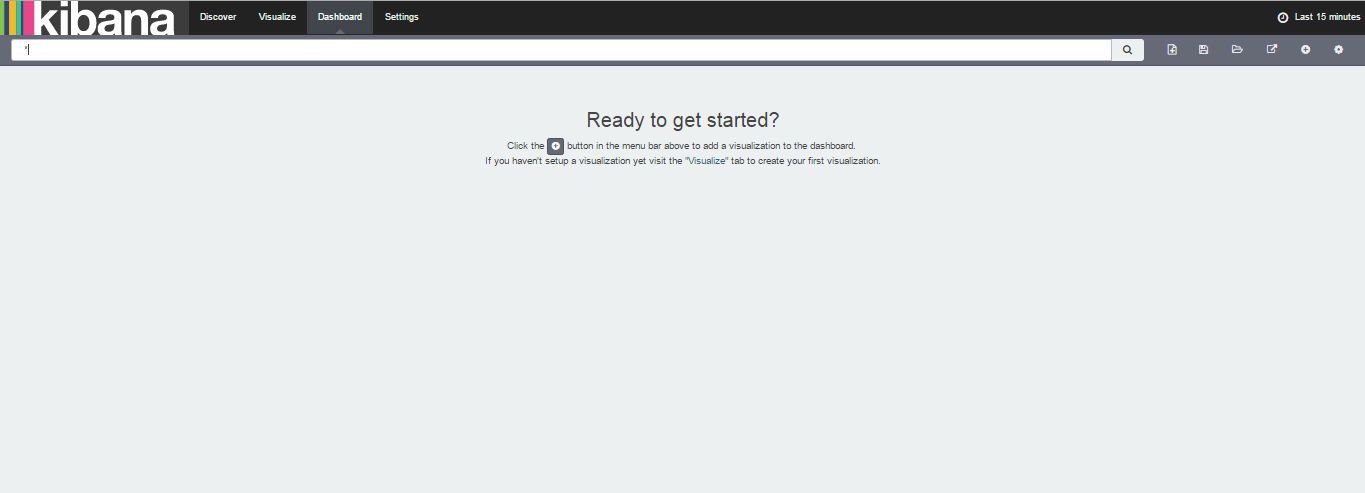


Similar way you can create line chart, etc...

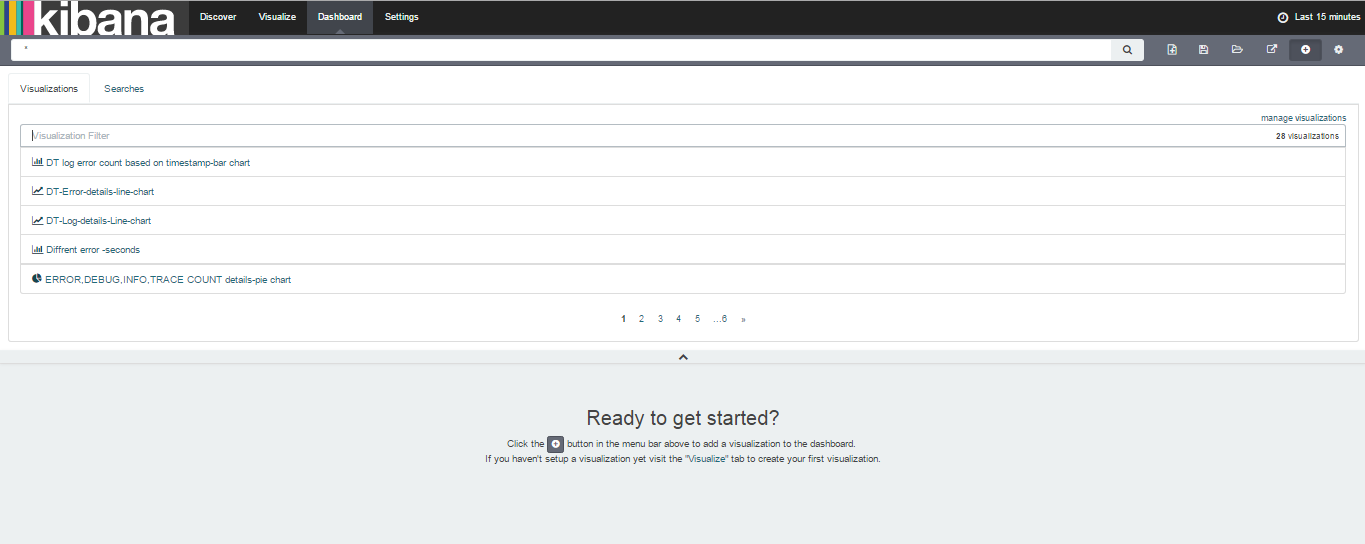


1. **Assembled visualizations into a Dashboard**

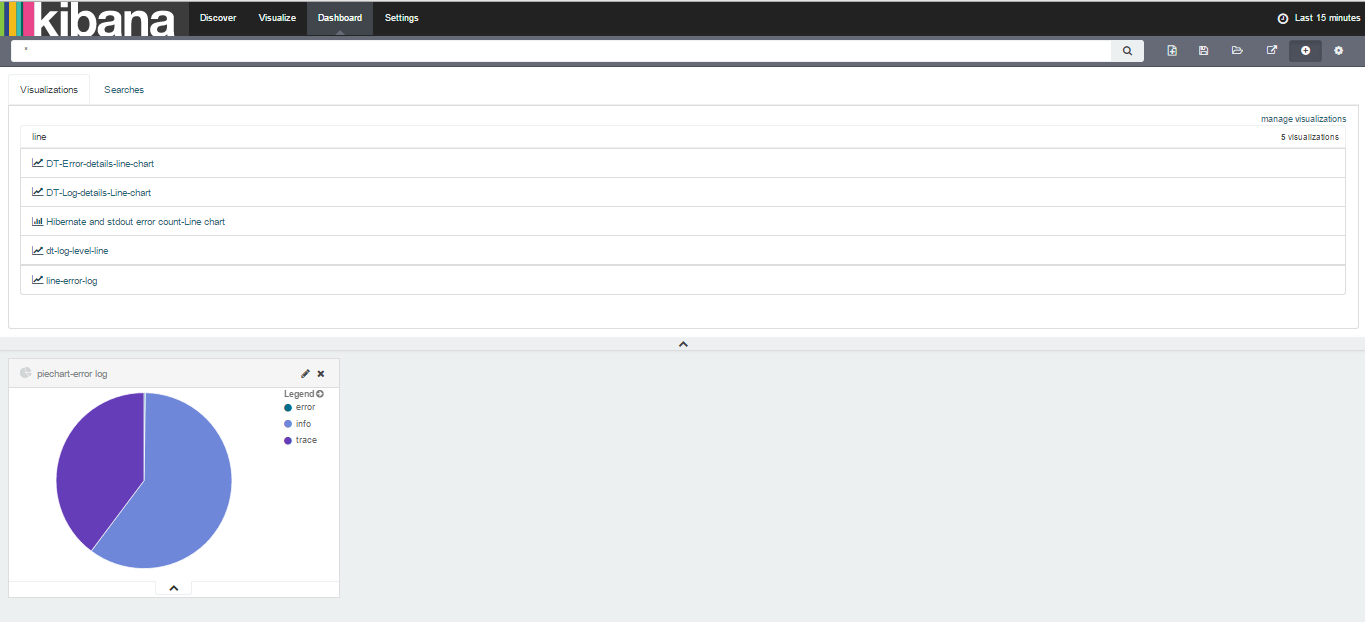
Now we can create dashboard from the created visualization. select dashboard tab



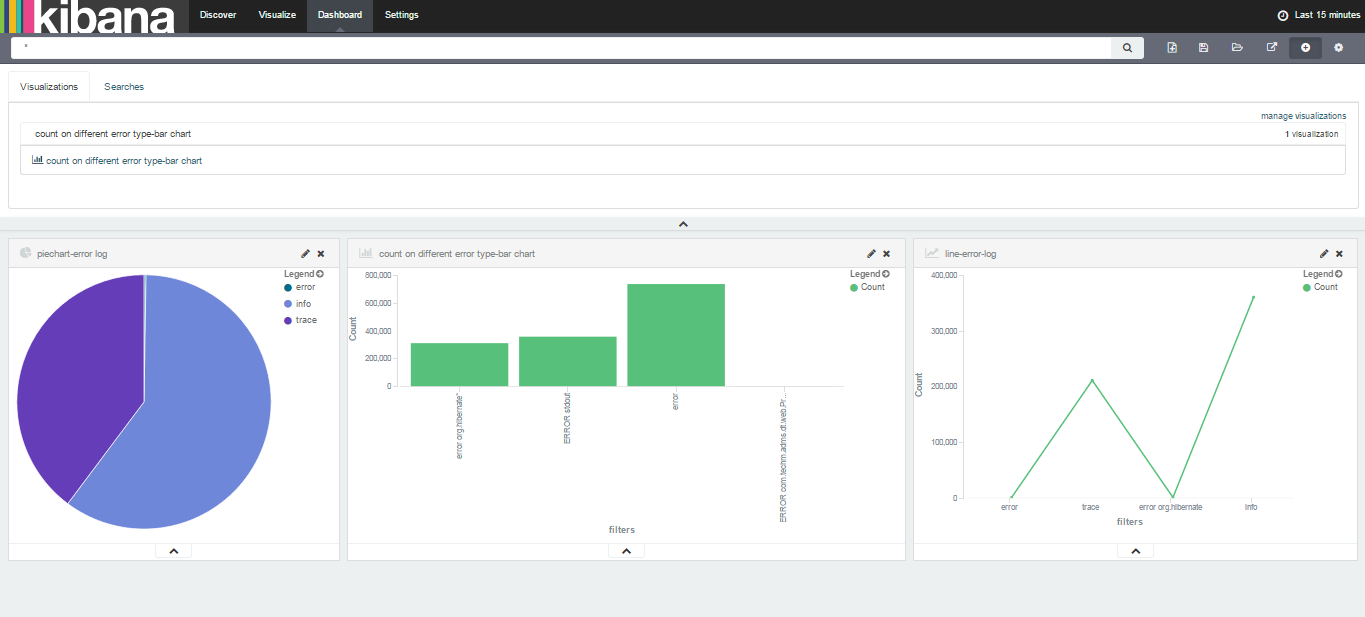
Dashboard🡪click the + icon to create the new dashboard. Then you will see the below window



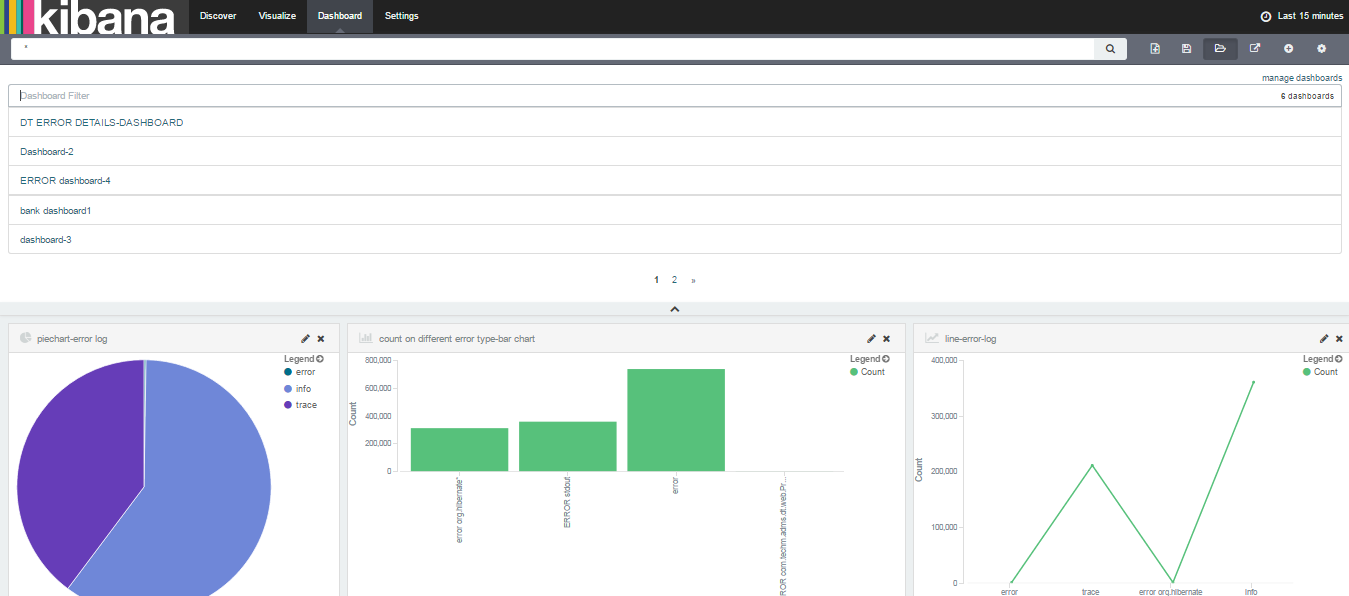
Select the visualization that you want to insert to the dashboard. Then it will be added to the dashboard below



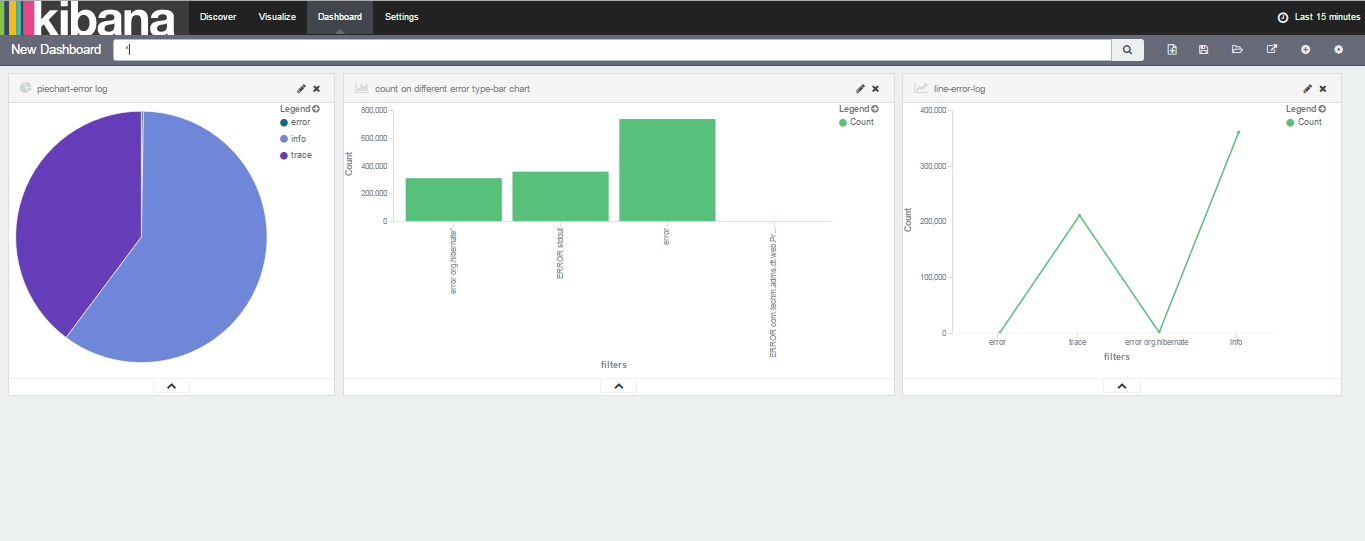
Like this you can add number of visualization to a dashboard.



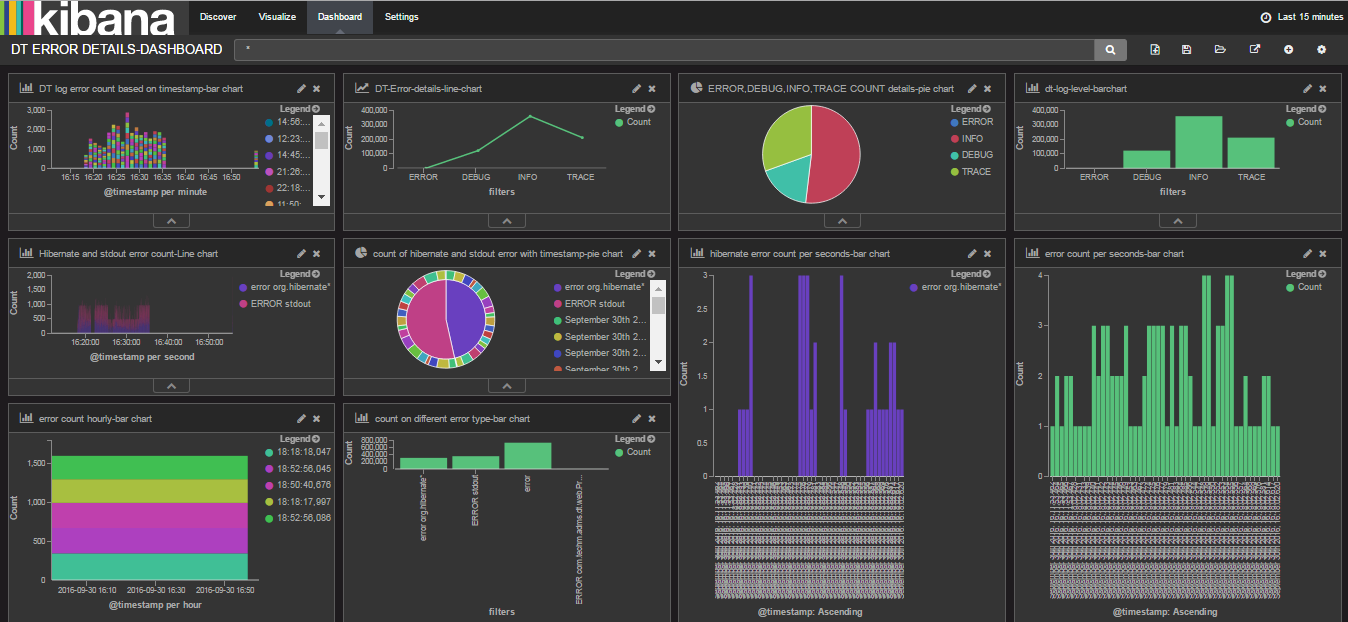
Save the dashboard by clicking the save icon on the left corner of the window.save with the name.



The dashboard will look like below.



We can do number of dashboard as required. some sample dashboards are.



This is how you will create the dashboard for the data using kibana.