Visualizing Cluster Logs with Kibana

Objectives

After completing this section, you should be able to visualize Kibana charts and understand their purpose.

Creating Visualizations

Graphical representations of cluster logs and running applications help administrators to effectively analyze any possible issues within the cluster.

You can create charts from the Visualize menu in Kibana to graphically represent any kind of information present in the logs.

There are different types of visualizations, including Area Charts, Data Tables, Heat Maps, Time Series, and Other.

Hover over a visualization type to view a brief description:

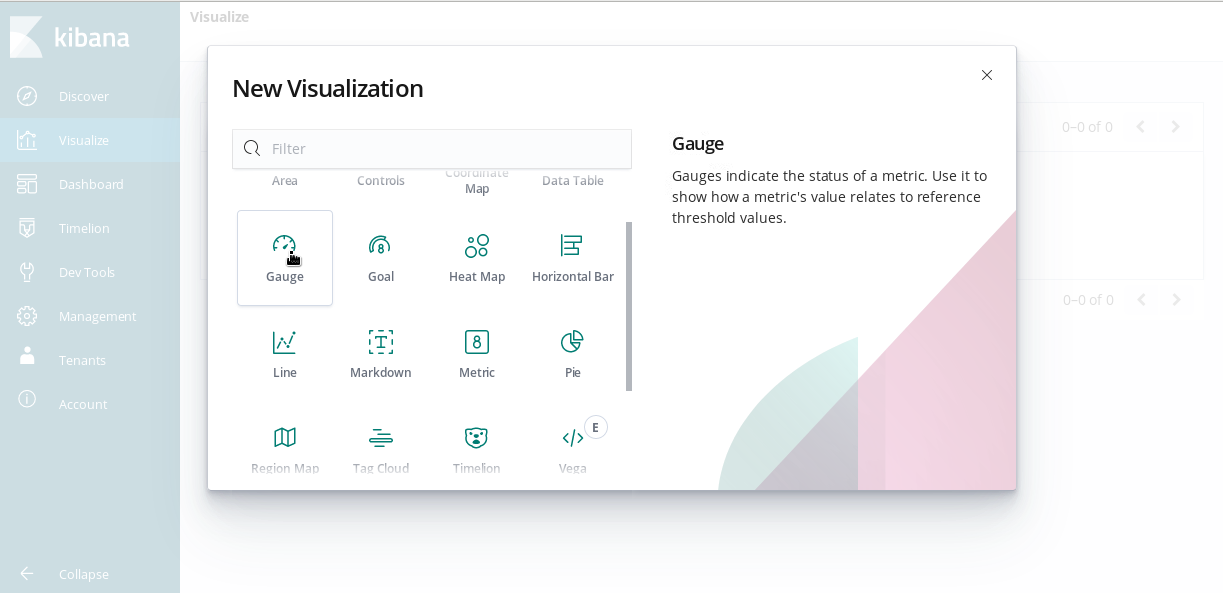


Figure 10.3: Visualization Types and a brief description.

For most visualizations, after selecting the visualization type you must provide the search source by selecting an index or a saved query.

Describing a Pie Chart

The following image represents a pie chart showing different types of HTTP response codes present in the logs of an application. On the left, you can create different Filters (HTTP response codes in this example). The chart on the right is a graphical representation of the Filters that you create on the left.

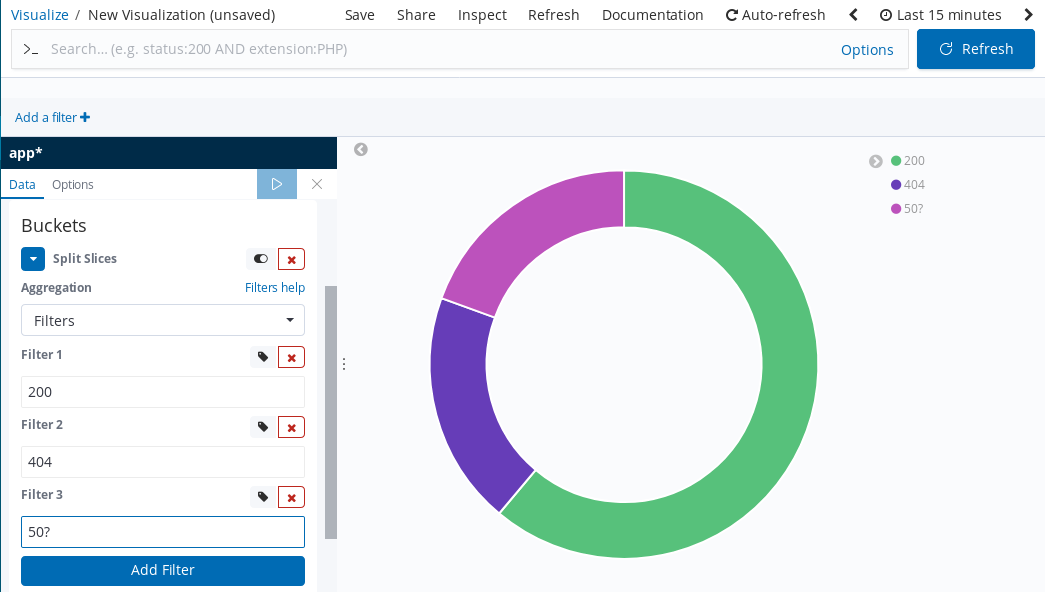


Figure 10.4: The Kibana Visualize screen with a Pie Chart.

The previous image shows three Filters. Each one represents a slice of the pie chart:

* The Filter with value 200 is represented in **green**.
* The Filter with value 404 is represented in **purple**.
* The Filter with value 50? is represented in **pink**.

**NOTE**

The value 50? matches any type of 500 HTTP error code, such as 500 (Internal server error) or 503 (Bad Gateway).

Hover over the pie chart to see the number and percentage of logs from each slice, as shown in the following image:

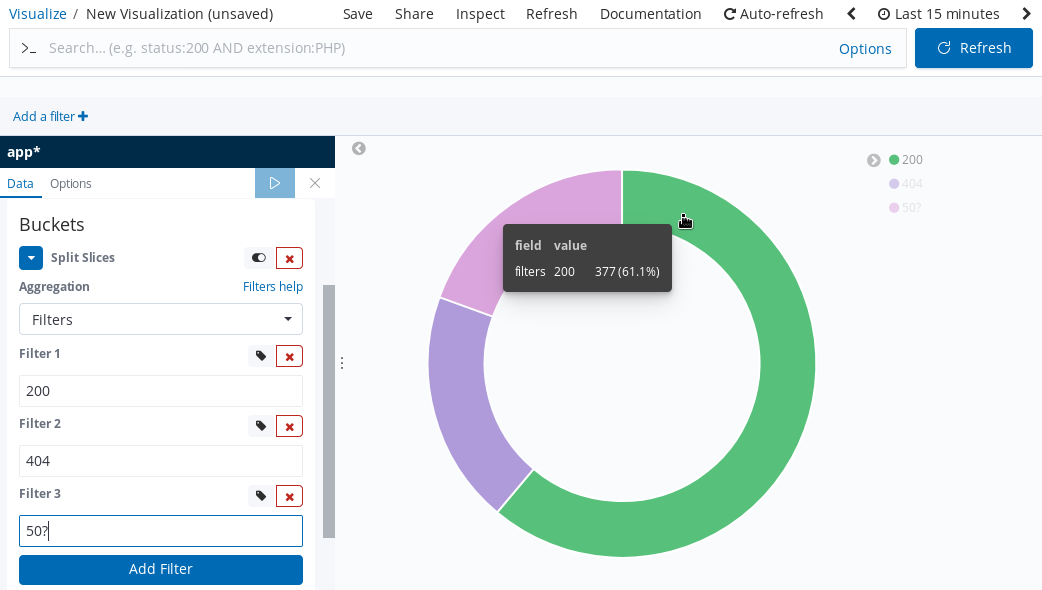
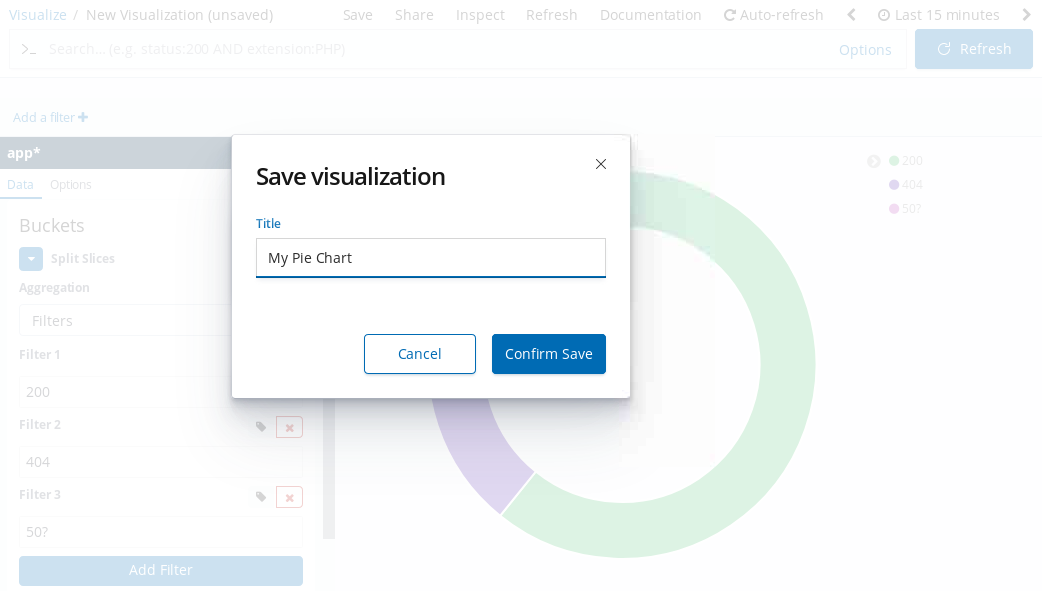
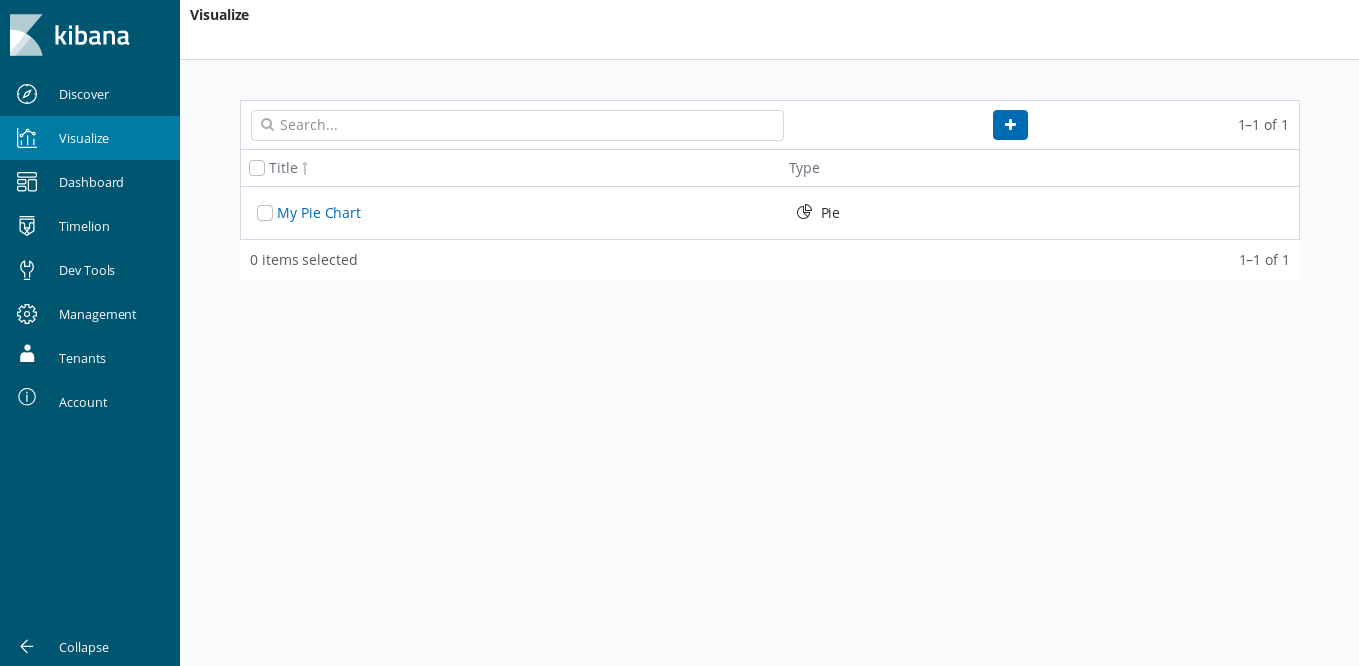


Figure 10.5: A Pie Chart showing information of the Slice.

Select Save on the main menu, type a name in the Title field, and then click **Confirm Save**.



After saving, the chart is available from the Visualize option on the main menu:



Visualizing Time Series with Timelion

Timelion uses time series data as an input to visualize data over time. Timelion provides functions to pull data from multiple data sources and to transform the data before representing it graphically. To access the Timelion documentation, click the **Jump to the function reference** link on the Timelion welcome page.

Describing a Sample Time Series

**IMPORTANT**

Timelion queries should be typed on a single line. The queries are presented here as multiline statements for readability.

You can visualize a new Timelion graphical series from the Timelion menu in Kibana.

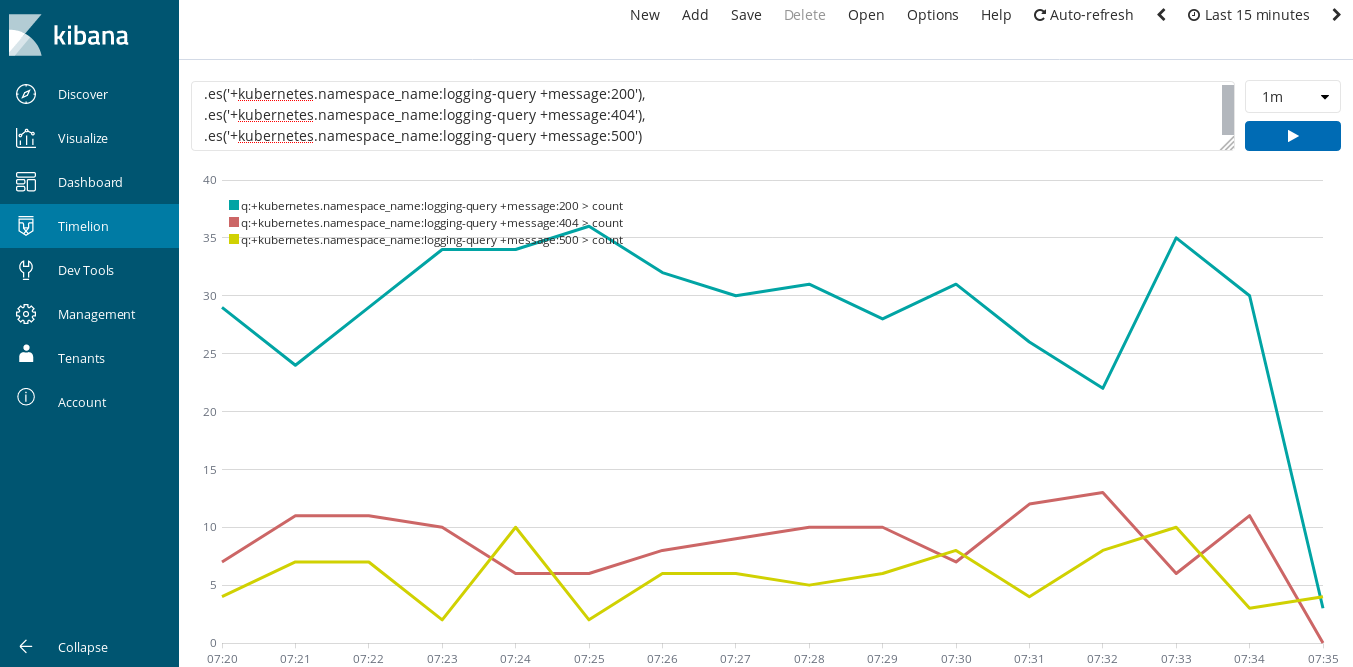
In the text box, you must indicate a valid Timelion Expression. Then, click **Play** to see the Time Series graph. The default expression is .es(\*), which uses the Elasticsearch function (.es()) to pass the \* Lucene query to Elasticsearch. The default query gathers all the data from the default index pattern. You can edit this expression to add a sample expression, such as:

.es('+kubernetes.namespace\_name:logging-query +message:200'),

.es('+kubernetes.namespace\_name:logging-query +message:404'),

.es('+kubernetes.namespace\_name:logging-query +message:500')

This expression represents the count of logs from the logging-query namespace that have either 200, 404, or 500 in them. Click **Apply Changes** to display the time series graph:



**NOTE**

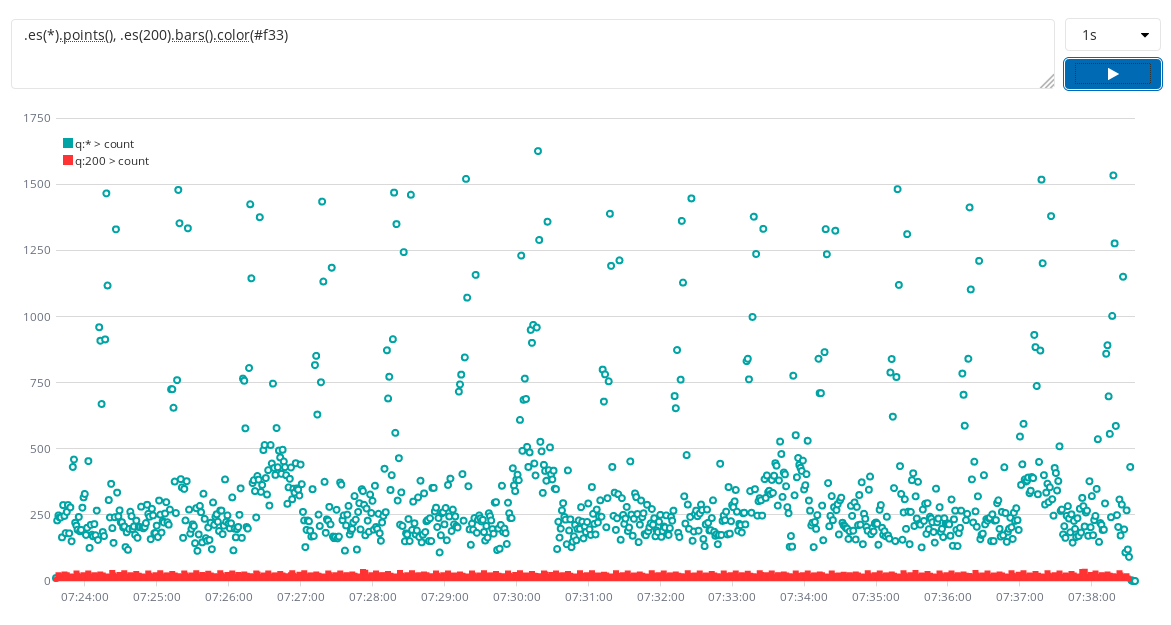
For better visualization, click on the "Full screen" icon on the generated graph.

You can also choose the resolution of the graph using the list located above the **Play** button. Adjust the resolution to refine time periods where there have been issues with your applications.

You can also click and drag the mouse over the graph to zoom in and view fine-grained details to find the exact time when an issue occurred.

Also, you can change the display of the chart. The following expression shows the count of all logs as points, and the logs that contain 200 as bright red bars.

.es(\*).points(), .es(200).bars().color(#f33)



As another example, the following expression adds labels and offset data for 10 minutes. The expression also labels the query for admin as current and the query for cart as previous.

.es(index=project.\*, q='admin').label(current),

.es(index=project.\*, q=cart, offset=-10m).label('previous')

Finally, you can save the charts as either a Timelion sheet or as a Kibana dashboard panel. Click **Save** on the Kibana toolbar, choose a save option, provide a name, and then click **Save**.