Log Querying and Analytics

Throughout this lab, each section will be broken down into a series of steps. To navigate between sections, click each header to expand or collapse the sections.

Make sure you are logged into Datadog using the Datadog training account credentials provisioned for you. You can find that information by running `creds` in the lab terminal.

The Log Explorer

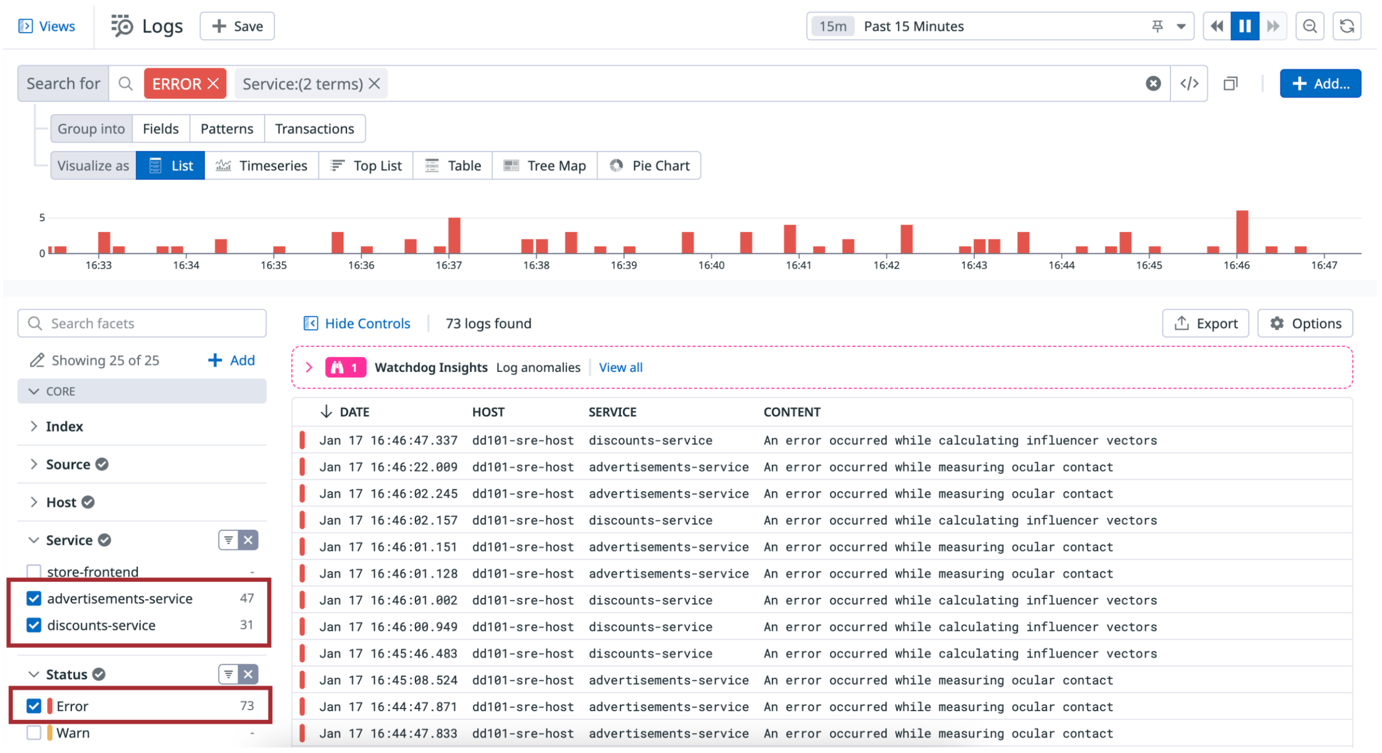
The Logs Search page displays indexed application logs matching a search query for a selected time range. A search query defines the criteria to “filter” the log entries to display. You can progressively filter the logs to home in on the specific types of log lines that you’re looking for.

The Datadog Agent is collecting many log lines from all of Storedog’s services and sending them to the Datadog to process.

1. Log in to Datadog using the trial credentials the lab created for you. You can run `creds` in the lab terminal whenever you need to retrieve your Datadog training account credentials.
2. Navigate to **Logs** > **Search**.

   You can build a search query using the text field at the top of the page and the Facets panel to the left of the search results.

   A search query can include assigned tags, like `env` and `service`; attributes extracted from the logs, like `@http.status_code`; and text strings from log messages.
3. In the Facets panel, under **Service**, select `advertisements-service` and `discounts-service`, and under **Status**, select `Error` to filter logs and display only error log lines from those two services:



```
DATE                   HOST            SERVICE                 CONTENT
Jan 17 16:46:47.337   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:46:22.009   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:46:02.245   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:46:02.157   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:46:01.151   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:46:01.128   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:46:01.002   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:46:00.949   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:45:46.483   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:45:08.524   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:44:47.871   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:44:47.833   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
```

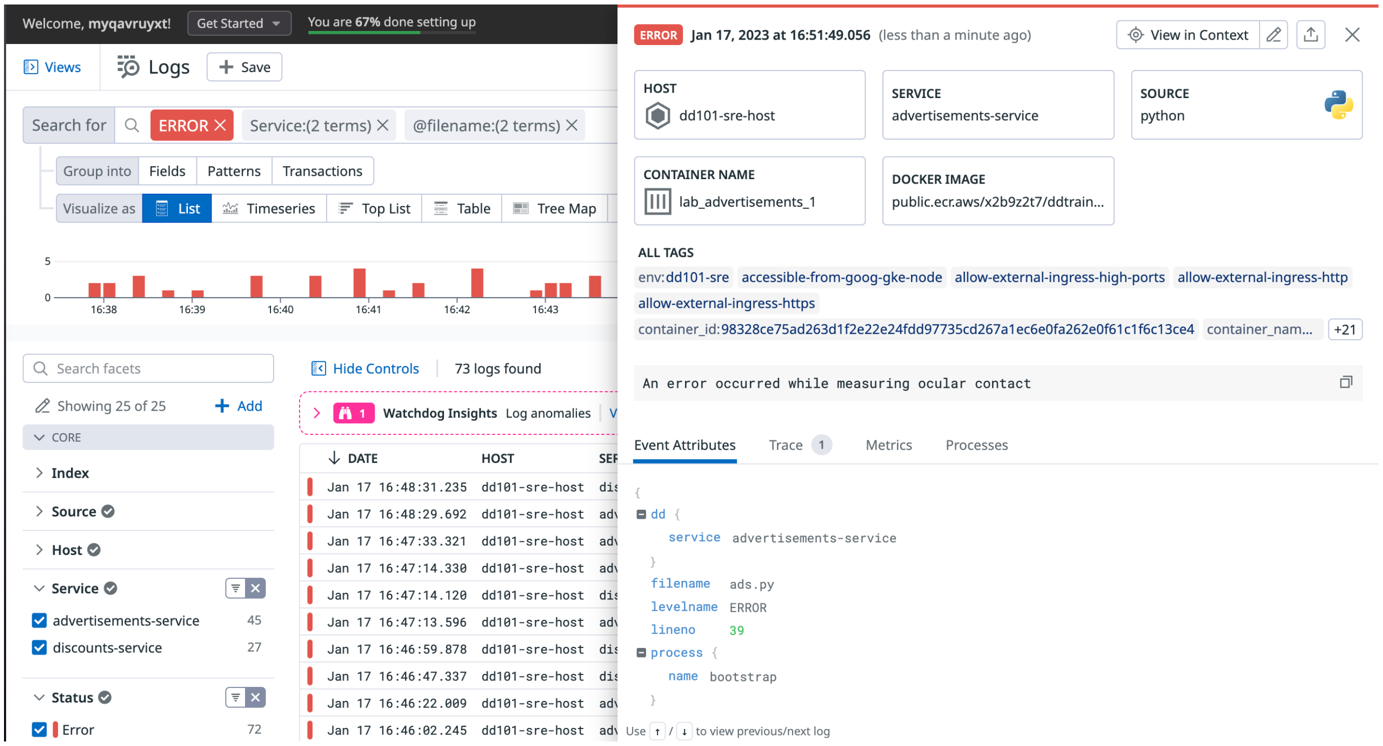
   Most of the log lines come from either `ads.py` or `discounts.py`, the main scripts for those services.

4. Further filter the results by adding the following to the search field:

   `@filename:(discounts.py OR ads.py)`

   This will filter log lines from those two files specifically.

5. Click on a log in the log list to open a side panel which displays details including assigned tags, the log message, any extracted attributes, related traces, and related infrastructure metrics.



```
Jan 17 16:48:31.235  dd101-sre-host  dis
Jan 17 16:48:29.692  dd101-sre-host  adv
Jan 17 16:47:33.321  dd101-sre-host  adv
Jan 17 16:47:14.330  dd101-sre-host  adv
Jan 17 16:47:14.120  dd101-sre-host  dis
Jan 17 16:47:13.596  dd101-sre-host  adv
Jan 17 16:46:59.878  dd101-sre-host  dis
Jan 17 16:46:47.337  dd101-sre-host  dis
Jan 17 16:46:22.009  dd101-sre-host  adv
Jan 17 16:46:02.245  dd101-sre-host  adv
```

ERROR Jan 17, 2023 at 16:51:49.056 (less than a minute ago)

HOST dd101-sre-host	SERVICE advertisements-service	SOURCE python
CONTAINER NAME lab_advertisements_1	DOCKER IMAGE public.ecr.aws/x2b9z2t7/ddtrain...	

ALL TAGS
env:dd101-sre accessible-from-goog-gke-node allow-external-ingress-high-ports allow-external-ingress-http allow-external-ingress-https container_id:98328ce75ad263d1f2e22e24fdd97735cd267a1ec6e0fa262e0f61c1f6c13ce4 container_nam... +21

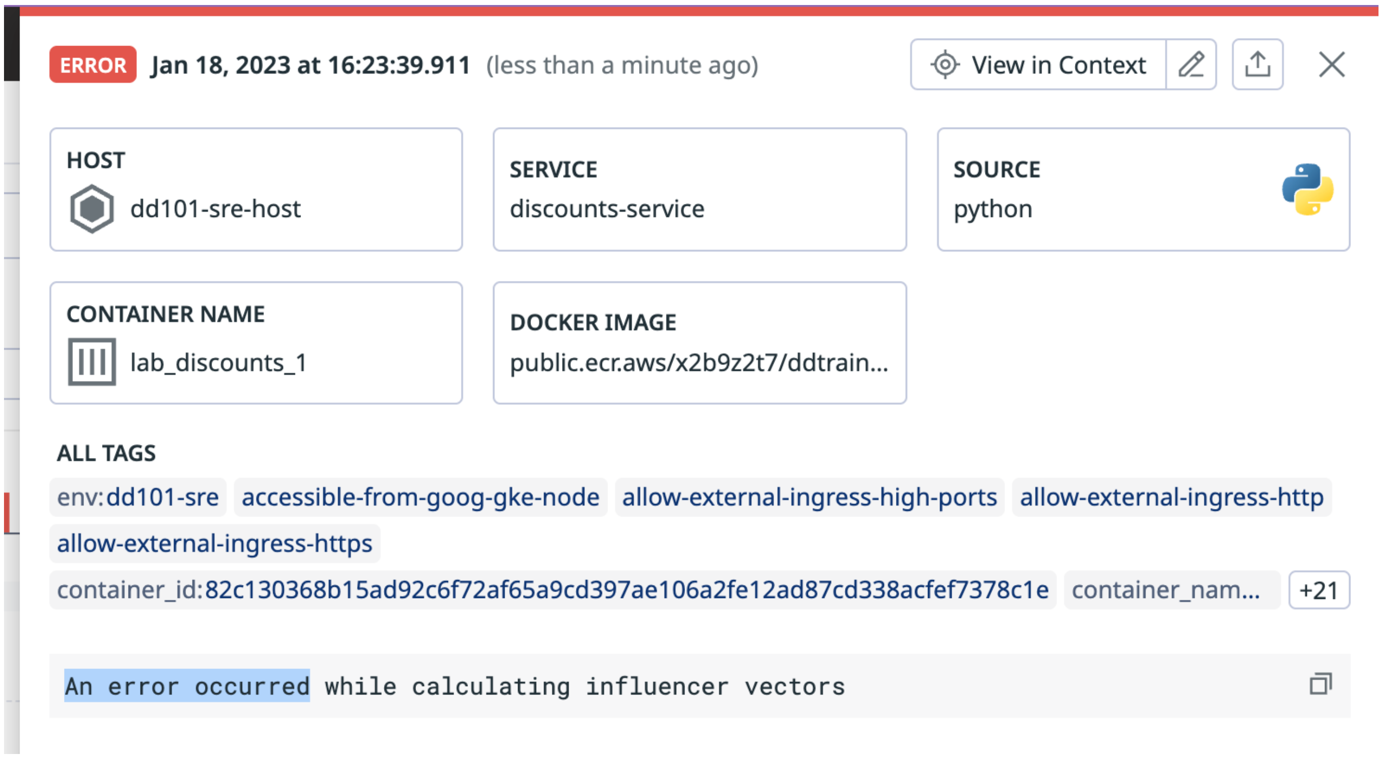
```
An error occurred while measuring ocular contact
```

```
{
⊟ dd {
    service  advertisements-service
  }
  filename   ads.py
  levelname  ERROR
  lineno     39
⊟ process {
    name  bootstrap
  }
```

6. Get familiar with the details provided by the log side panel:
   1. Click the **Trace** tab to view the associated trace.
   2. Click the **Metrics** tab to view the associated infrastructure metric.

7. Notice the log message beginning with "An error occurred…". Instead of using the `ERROR` status, use this string to filter error logs for these services.

   Copy the part of the log message that reads `An error occurred`, and close the log side panel.


ERROR Jan 18, 2023 at 16:23:39.911 (less than a minute ago)

HOST	SERVICE	SOURCE
dd101-sre-host	discounts-service	python

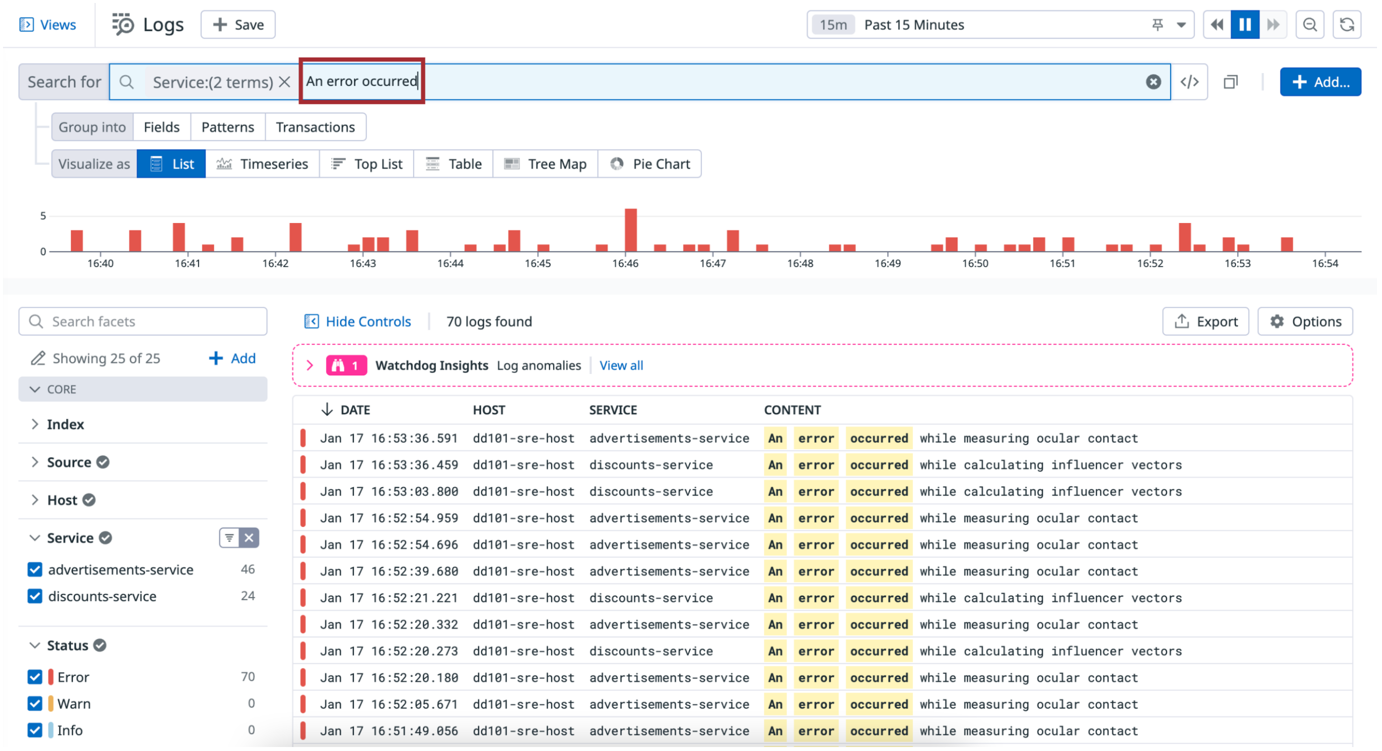
CONTAINER NAME	DOCKER IMAGE
lab_discounts_1	public.ecr.aws/x2b9z2t7/ddtrain...

ALL TAGS

env:dd101-sre accessible-from-goog-gke-node allow-external-ingress-high-ports allow-external-ingress-http allow-external-ingress-https container_id:82c130368b15ad92c6f72af65a9cd397ae106a2fe12ad87cd338acfef7378c1e container_nam... +21

```
An error occurred while calculating influencer vectors
```

8. In the search field, delete `ERROR` and `@filename:(discounts.py OR ads.py)` from the search field.
9. Paste the log message text `An error occurred` in the search field and press Enter. Notice the same list of logs appears.



Service	
✓ advertisements-service	46
✓ discounts-service	24

Status	
✓ Error	70
✓ Warn	0
✓ Info	0

70 logs found

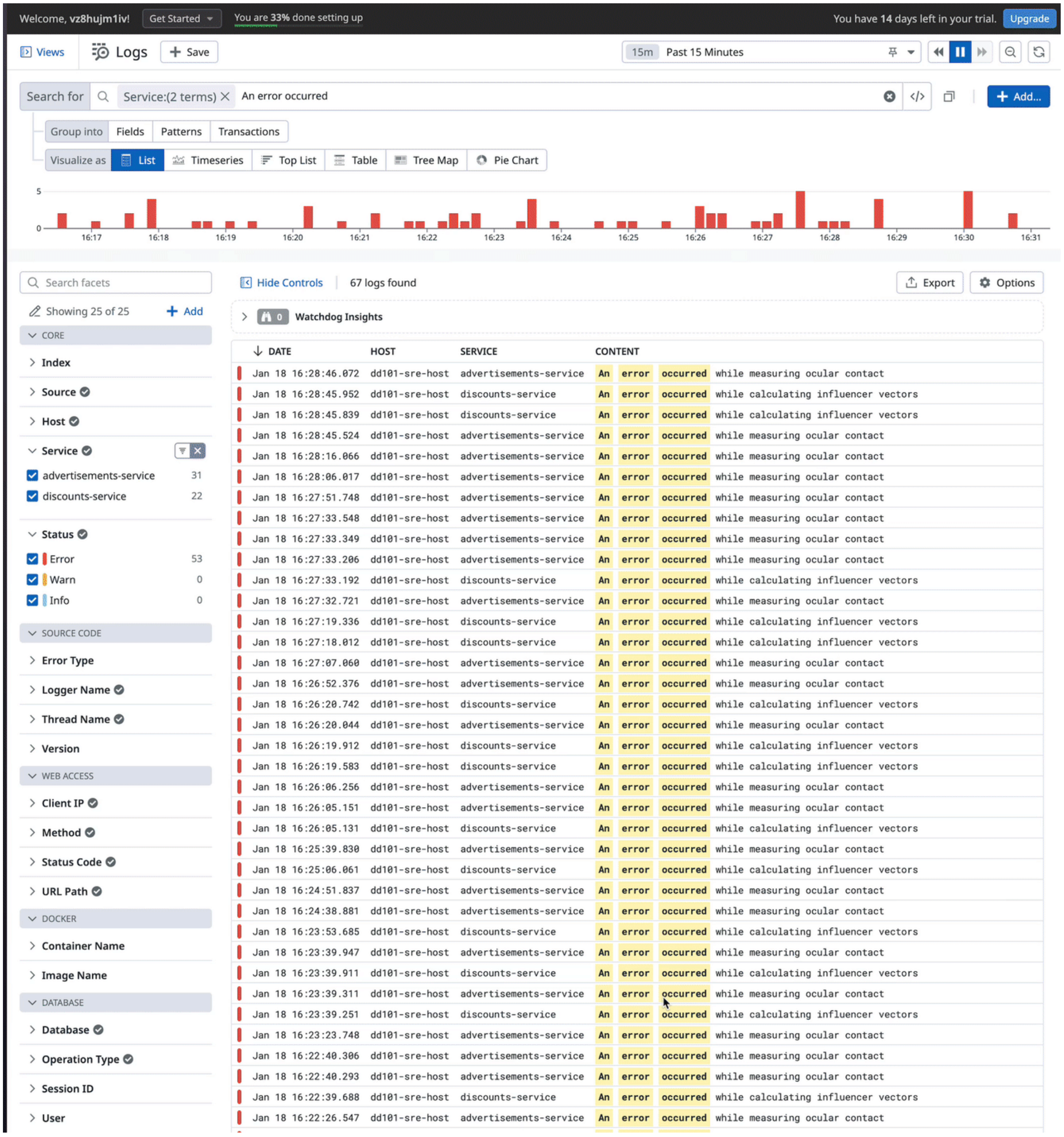
Watchdog Insights Log anomalies View all

```
DATE                  HOST            SERVICE                 CONTENT
Jan 17 16:53:36.591   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:53:36.459   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:53:03.800   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:52:54.959   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:52:54.696   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:52:39.680   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:52:21.221   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:52:20.332   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:52:20.273   dd101-sre-host  discounts-service       An error occurred while calculating influencer vectors
Jan 17 16:52:20.180   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:52:05.671   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
Jan 17 16:51:49.056   dd101-sre-host  advertisements-service  An error occurred while measuring ocular contact
```

Now you know how to build a search query using the facets panel as well as the search field.

Custom Facets

Common tags and attributes automatically appear in the Facets panel as Datadog parses logs. You can also add a tag to the Facets panel as a custom facet from the log details in the log side panel.

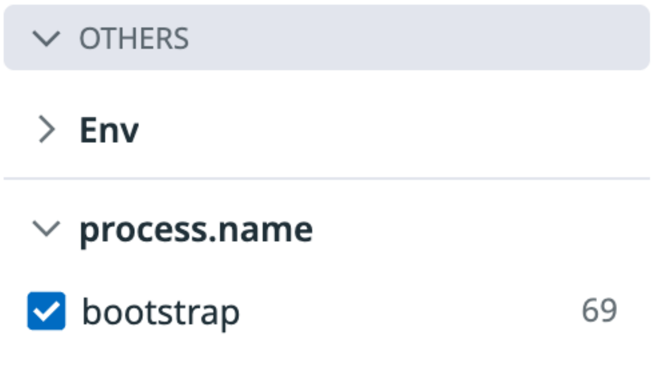


```
DATE                    HOST             SERVICE                 CONTENT
Jan 18 16:28:46.072     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:28:45.952     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:28:45.839     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:28:45.524     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:28:16.066     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:28:06.017     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:51.748     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:33.548     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:33.349     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:33.206     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:33.192     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:27:32.721     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:27:19.336     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:27:18.012     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:27:07.060     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:26:52.376     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:26:20.742     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:26:20.044     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:26:19.912     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:26:19.583     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:26:06.256     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:26:05.151     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:26:05.131     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:25:39.830     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:25:06.061     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:24:51.837     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:24:38.881     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:23:53.685     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:23:39.947     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:23:39.911     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:23:39.311     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:23:39.251     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:23:23.748     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:22:40.306     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:22:40.293     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
Jan 18 16:22:39.688     dd101-sre-host   discounts-service       An error occurred while calculating influencer vectors
Jan 18 16:22:26.547     dd101-sre-host   advertisements-service  An error occurred while measuring ocular contact
```

1. Click one of the log lines to open the log side panel. Scroll down to view the list of **Event Attributes**.
2. Under the `process` attribute, hover over `name` and click the gear icon that appears.
3. Select **Create facet for @process.name**.

   The **Add facet** window will appear. Expand **Advanced** to view the additional fields.
4. Click **Add**. You'll see a message confirming that the facet has been successfully added.
5. Close the logs side panel.
6. Clear the search field at the top of the page and select `Past 15 Minutes` from the timeframe dropdown in the upper-right.
7. Scroll to the bottom of the Facets list. Under the **OTHERS** facet group, expand the **process.name** facet.

You may need to wait for fresh logs to be collected and processed for the new facet to appear. Soon, you'll see the values of this attribute that was found in log entries:



8. Select `bootstrap` to filter the logs by lines where the `process.name` attribute value is `bootstrap`.

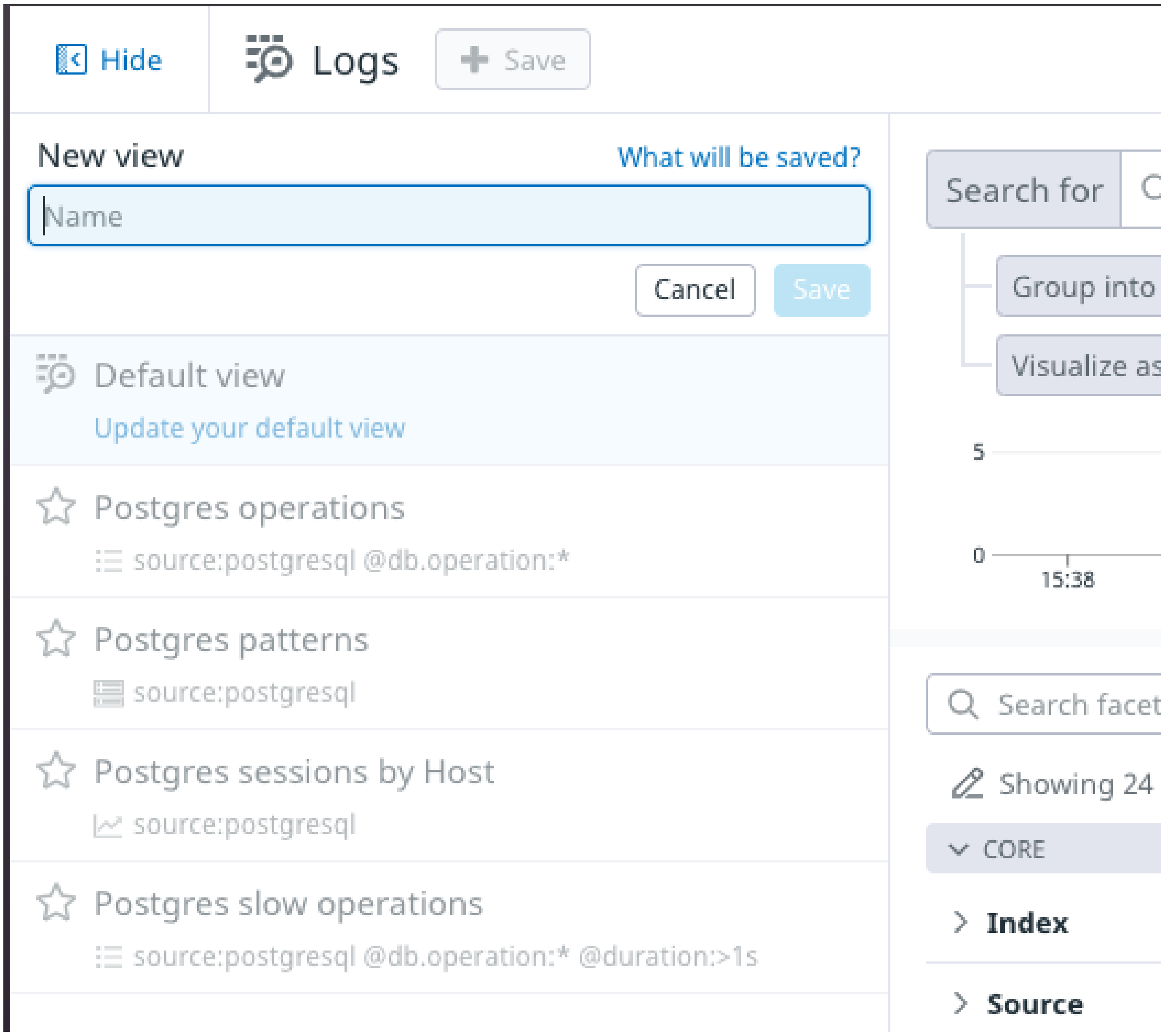
The tags and attributes available to you for creating search contexts depend on the tags you assign to the logs, and the attributes you extract from the logs.

Saved Views

You can save search queries as Saved Views to recall at any time:

1. Clear the search field.
2. In the Facets panel, under **Service**, select `advertisements-service` and `discounts-service`, and under **Status**, select `Error`.
3. Click the **+ Save** button above the search field to save this search query.

   The Views panel will open with a list of Saved Views. In addition to `Default view`, you will see predefined views provided by the PostgreSQL integration:



New view — What will be saved?

Name

Cancel · Save

Default view
Update your default view

Postgres operations
source:postgresql @db.operation:*

Postgres patterns
source:postgresql

Postgres sessions by Host
source:postgresql

Postgres slow operations
source:postgresql @db.operation:* @duration:>1s

4. Under **New view**, in the **Name** field, enter `Ads and Discounts Errors` and click **Save**.

   The new view will appear in the list.

5. Click the **Default view** saved view. This will clear the search query.

6. Click the new **Ads and Discounts Errors** view you just created. You'll see the search query populate with the saved view.

7. Click **Hide** above the filtered views to close the Views panel.

Live Tail

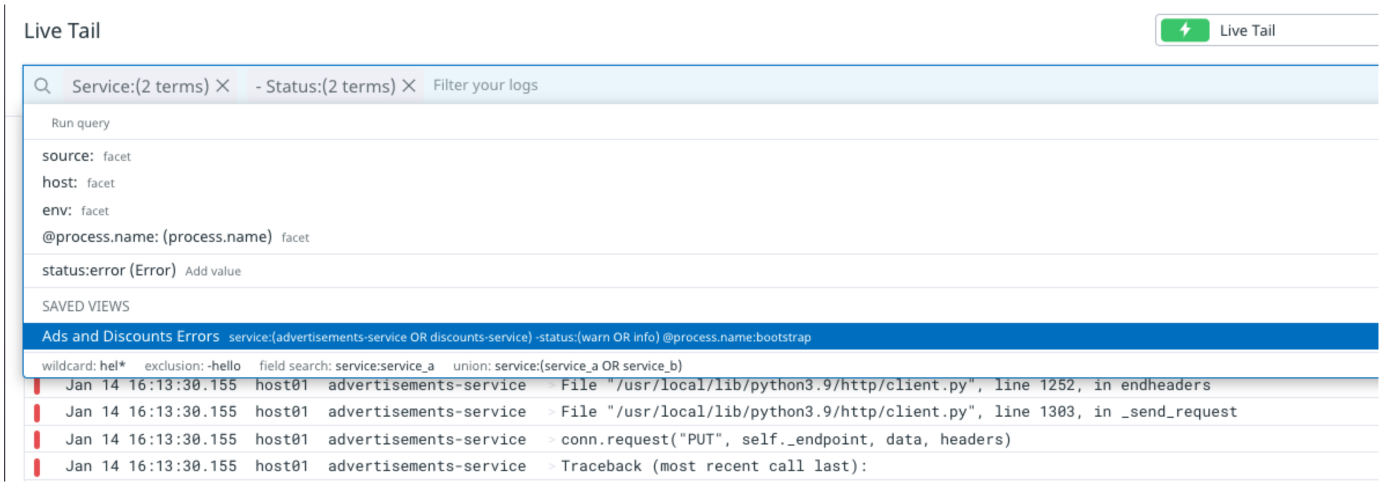
The Live Tail view displays all logs ingested by Datadog after processing, but before indexing (or archiving). This view is useful for seeing log lines as soon as possible. However, these logs do not persist in this list, so you can't see past logs that have been displayed.

1. In the timeframe dropdown above the Log List, select `Live Tail` and wait for the list to start populating.

   Notice that the logs displayed in the Live Tail logs list match the search query.

2. Clear the search query. Wait for new logs to be collected. Notice that the Live Tail list now displays *all* the logs that are being collected from the app.

While you don't have access to the filters available for indexed logs, you can update the query in the search field. You also have access to your custom facets and saved views in Live Tail:


Live Tail
Live Tail
Service:(2 terms)
- Status:(2 terms)
Filter your logs
Run query
source: facet
host: facet
env: facet
@process.name: (process.name) facet
status:error (Error) Add value
SAVED VIEWS
Ads and Discounts Errors service:(advertisements-service OR discounts-service) -status:(warn OR info) @process.name:bootstrap
wildcard: hel* exclusion: -hello field search: service:service_a union: service:(service_a OR service_b)

```
Jan 14 16:13:30.155  host01  advertisements-service  File "/usr/local/lib/python3.9/http/client.py", line 1252, in endheaders
Jan 14 16:13:30.155  host01  advertisements-service  File "/usr/local/lib/python3.9/http/client.py", line 1303, in _send_request
Jan 14 16:13:30.155  host01  advertisements-service  conn.request("PUT", self._endpoint, data, headers)
Jan 14 16:13:30.155  host01  advertisements-service  Traceback (most recent call last):
```

3. In the timeframe dropdown above the Log List, select `Past 15 minutes` to go back to Logs Search.

To learn more about the syntax of log search queries, see the Logs Search Syntax docs.

Now that you know how to search and filter logs, you'll can explore the different aggregation features for analyzing logs.

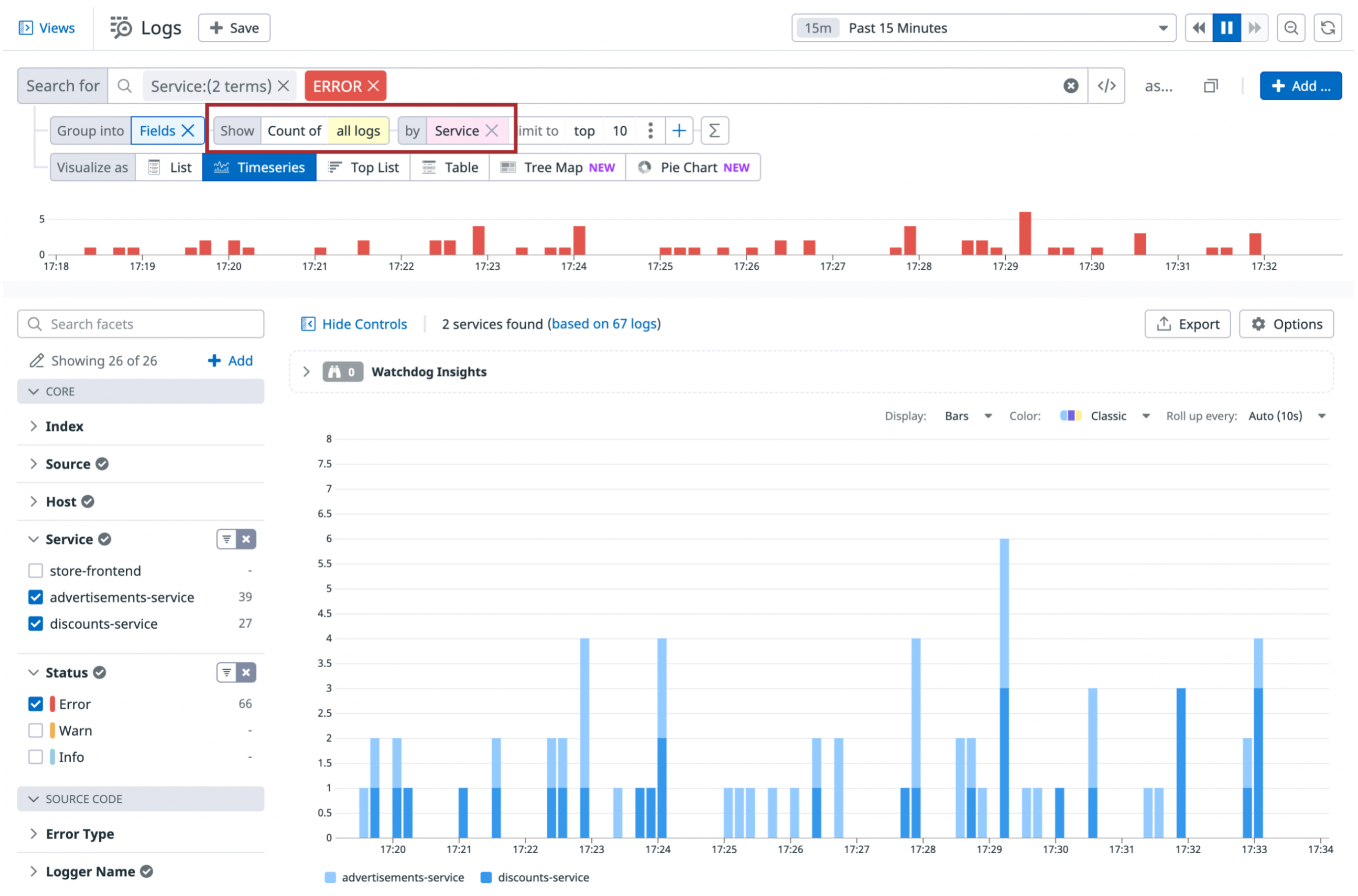
Log Aggregation

With Fields aggregation, all logs matching the query filter are aggregated into groups based on the value of a log facet. For these groups, you can extract counts of logs per group, a unique count of coded values for a facet per group, and statistical operations on numerical values of a facet per group.

Aggregating logs this way can help you see trends more clearly, visualize relationships between different types of log facets, and more.

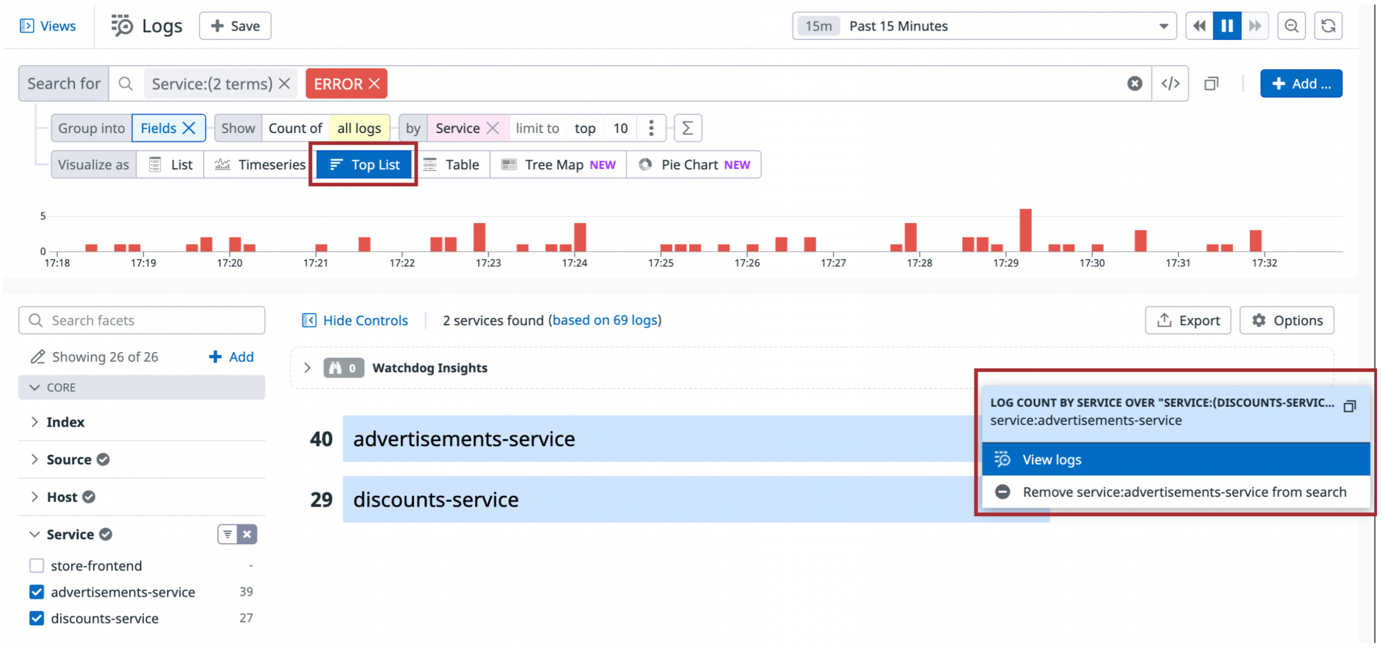
1. Navigate to **Logs** > **Search**.
2. Open the Views panel and select the saved view you created, **Ads and Discounts Errors**.
3. Below the search field, for **Group into**, select `Fields`.

   A graph visualization of the filtered logs will replace the Log List.
4. Group the fields by `service` and show a count of all logs:



5. For **Visualize as** above the graph, you'll see that **Timeseries** is selected by default. Select the other visualization options, like **Top List**, to see what they look like.

Note that the visualizations are interactive. You can click on rows or graphs to view a list of the aggregated logs in a side panel:

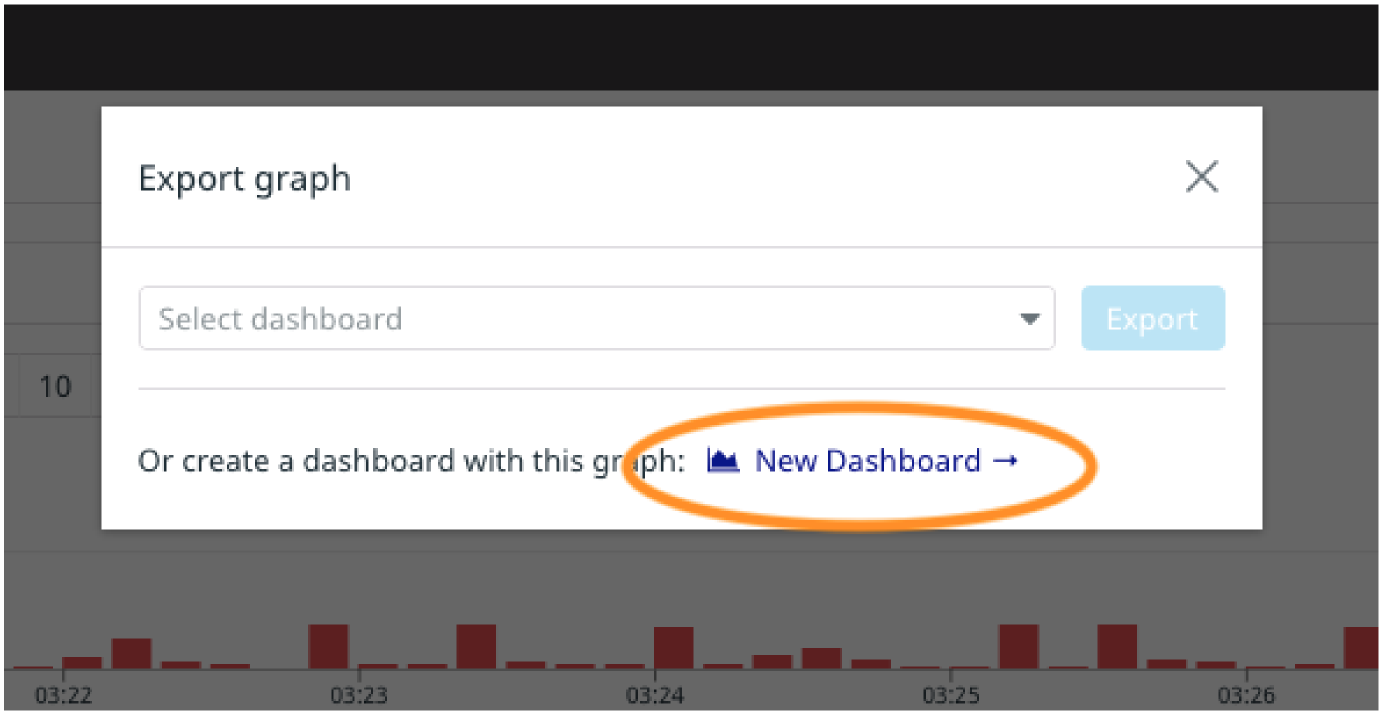


Exporting Graphs

You can export any logs visualization to other areas of the Datadog App, such as Monitors, Dashboards, and Notebooks. You can create a custom metric from logs, or download the aggregated data as a CSV.

1. For **Visualize as**, select **Timeseries**.

2. Click the **Export** button above the graph.
3. Click **Export to dashboard**.
4. In the export dialog, click the **New Dashboard** link.



   A notice at the top of the screen contains a link to the new dashboard. You can dismiss it, as you'll work with the dashboard later.
5. Under **Visualize as**, click **List** to return to the default log list view.

In the next section, you'll aggregate logs based on patterns that Datadog detects across all of your log contents.

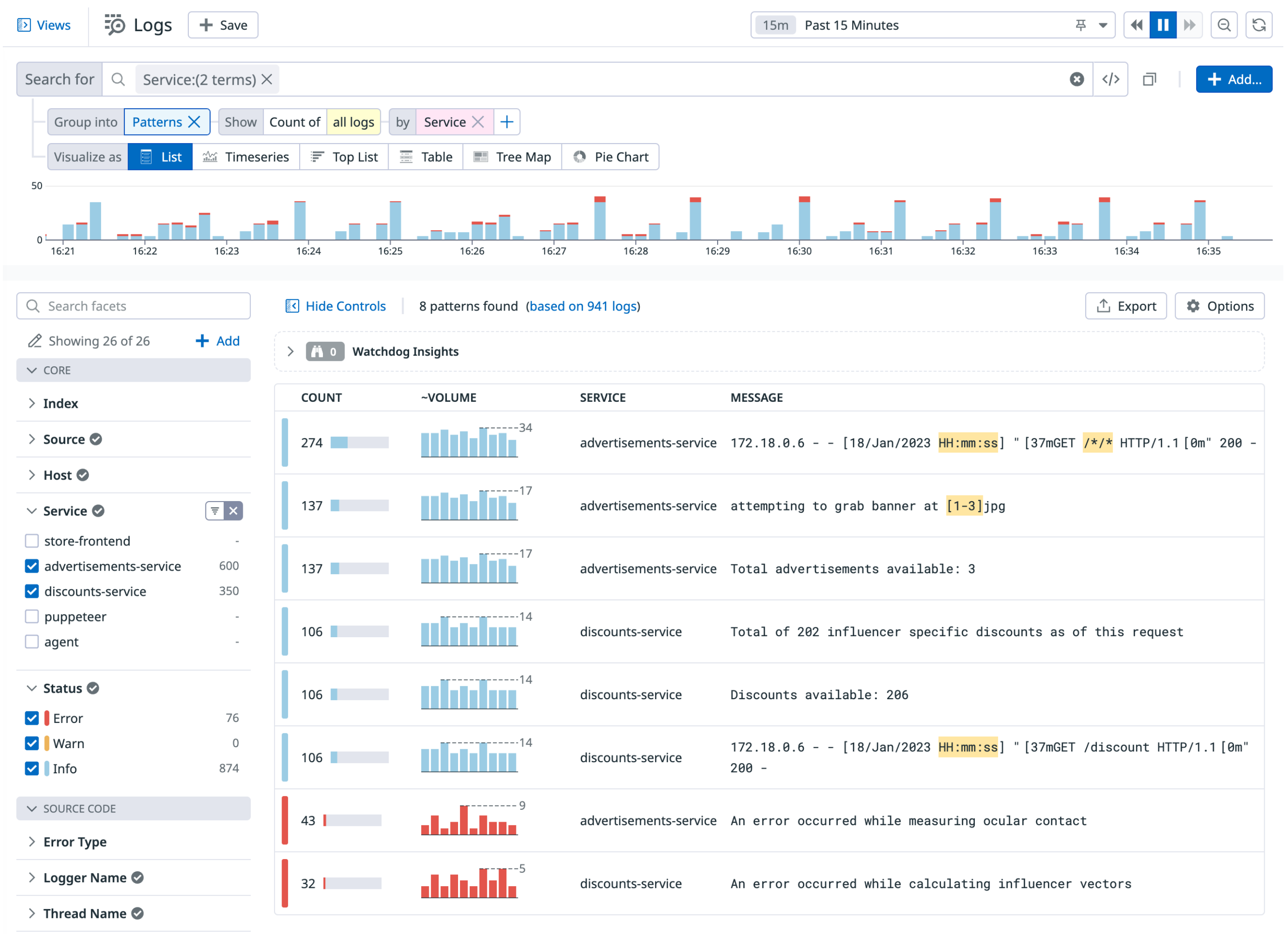
Pattern Aggregation

Organizing large volumes of logs from different sources can be cumbersome. Not all log lines will be nicely tagged, intelligently parsed, and easily referenced in search queries. Fortunately, logs have patterns that Datadog can automatically detect, enabling aggregation of similar log lines.

With pattern aggregation, logs that have a message with similar structures, belong to the same service, and have the same status are grouped together. The patterns view helps you detect and filter noisy error patterns that could cause you to miss other issues.

1. Navigate to **Logs > Search**.
2. Open the Views panel and select the saved view you created, **Ads and Discounts Errors**.
3. Remove the `ERROR` status from the search field.
4. For **Group into**, select `Patterns`.

   The detected log patterns are displayed, sorted from the most common to the least common. You can see the count, the facet by which they are grouped (`service`, in this case), and the log message.



8 patterns found (based on 941 logs)

COUNT	~VOLUME	SERVICE	MESSAGE
274	34	advertisements-service	`172.18.0.6 - - [18/Jan/2023 HH:mm:ss] " [37mGET /*/* HTTP/1.1 [0m" 200 -`
137	17	advertisements-service	`attempting to grab banner at [1-3]jpg`
137	17	advertisements-service	`Total advertisements available: 3`
106	14	discounts-service	`Total of 202 influencer specific discounts as of this request`
106	14	discounts-service	`Discounts available: 206`
106	14	discounts-service	`172.18.0.6 - - [18/Jan/2023 HH:mm:ss] " [37mGET /discount HTTP/1.1 [0m" 200 -`
43	9	advertisements-service	`An error occurred while measuring ocular contact`
32	5	discounts-service	`An error occurred while calculating influencer vectors`

CORE	
Index	
Source	
Host	
Service	
☐ store-frontend	-
☑ advertisements-service	600
☑ discounts-service	350
☐ puppeteer	-
☐ agent	-
Status	
☑ Error	76
☑ Warn	0
☑ Info	874
SOURCE CODE	
Error Type	
Logger Name	
Thread Name	

5. In the query editor below the search field, at the end of the **Group into** row, set **by** to `process.name`. Here you can see the patterns grouped by the custom facet you created earlier.
6. Change **by** to `status` to see the patterns grouped by status value.
7. The log list column headings are sortable. Click the **COUNT** header twice to sort from least common to most common.
8. Click one of the patterns to open the details side panel. The pattern and a list of Log Samples are displayed.
9. Click a log line to view the details panel. Close the details panel when you're done.
10. Back in the pattern details side panel, click **Show Parsing Rule** above the pattern. This is a regular expression dialect known as Grok, one of the parsers used in Datadog log pipelines.

    Parsing rules can be very powerful for creating a custom pipeline to parse semi-structured log lines into well-structured, taggable objects, in the same way that Datadog automatically parses JSON log lines.

    You can take a look at the pipelines Datadog set up automatically under **Logs > Configuration > Pipelines**

    Creating custom pipelines is out of scope for this lab, but make a note of the Pipelines docs when you'd like to go deeper into this topic.

Lab Conclusion

Great job! You now know how to search logs, aggregate them by fields and by patterns. You also got a brief introduction to creating custom log pipelines to parse semi-structured log lines.

As an SRE, you will spend a lot of time managing and analyzing logs, and you now have powerful tools to assist you.

You can go *much* deeper with Datadog Logs than what we covered in this lab. Datadog offers many powerful log features, such as the following:

- Transactions allow you to aggregate related log events into a single higher-level “transaction” event using a tag or an attribute from the logs. You can visualize complex interconnected systems through log events, identify transaction bottlenecks by comparing durations, event counts, and custom measures.
- Generate Metrics from ingested logs to summarize log data over very long periods of time.
- Indexes provide fine-grained control over your Log Management budget by allowing you to segment data into value groups for differing retention, quotas, usage monitoring, and billing.
- As touched upon in this lab, Log Pipelines can transform semi-structured log lines to valuable data sources. For more information about using Grok and other parsers, see the Parsing Docs.
- Visit the Logs docs to learn more about even more logs features and functions.

When you’re done, enter the following command in the terminal:

```
finish
```

Click the **Check** button in the lower right corner of the lab and wait for the lab to close down before moving on to the next lesson.