Fundamentals of Cloud Logging

GSP610

Overview

Cloud Logging is part of the Operations suite of products in Google Cloud. It includes storage for logs, a user interface called the Logs Viewer, and an API to manage logs programmatically. Use Cloud Logging to read and write log entries, search and filter your logs, export your logs, and create logs-based metrics.

In this hands-on lab, you learn how to use Cloud Logging to accumulate application logs in a single place, filter to reach the required log, understand how to create logs based metrics for advanced analysis, examine the audit logs use case, and export logs for compliance and/or advanced analysis needs.

What you'll do

- Launch an example Google App Engine application to generate logs.
- Use Cloud Logging console to interact with the logs generated by the application.
- Create log-based Cloud Monitoring metrics.
- Use Cloud Logging to dive deep into Audit Logging.
- Create an Export of logs into BigQuery.

Setup and requirements

Qwiklab setup

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

To complete this lab, you need:

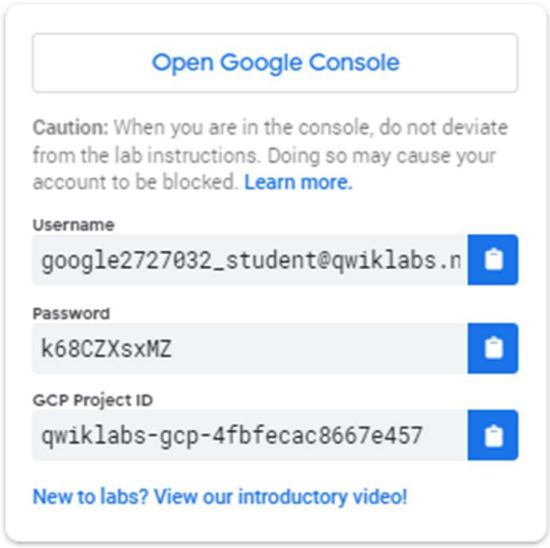
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.
  Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

Note: If you are using a Pixelbook, open an Incognito window to run this lab.

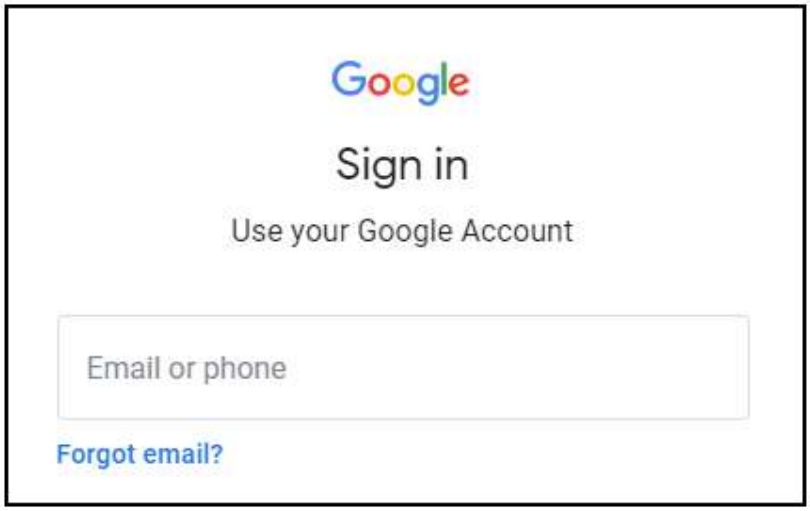
Cloud Console

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.

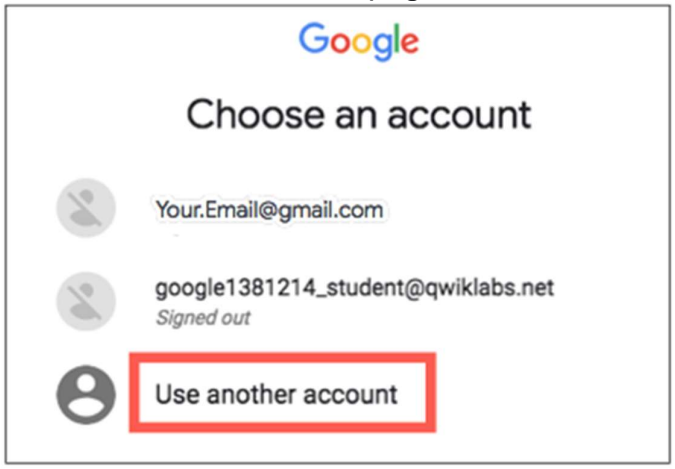


2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**.



3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

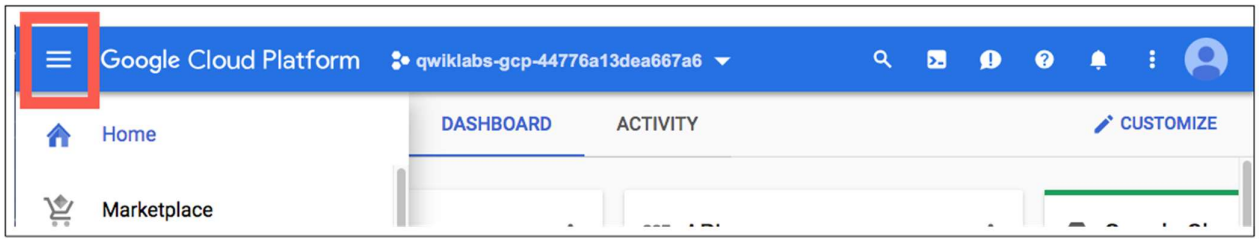
   Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:

   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

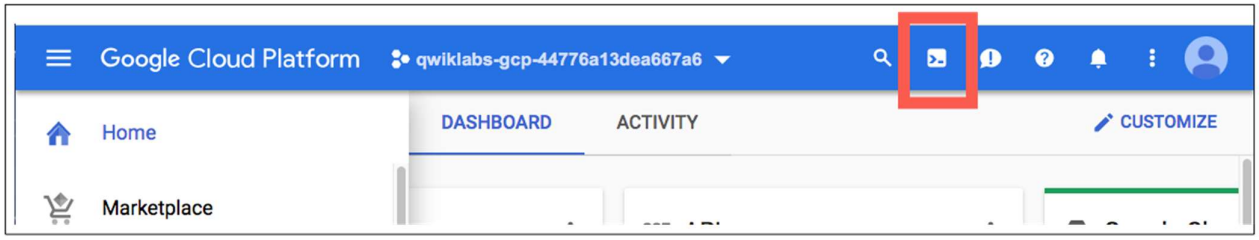
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-
left.



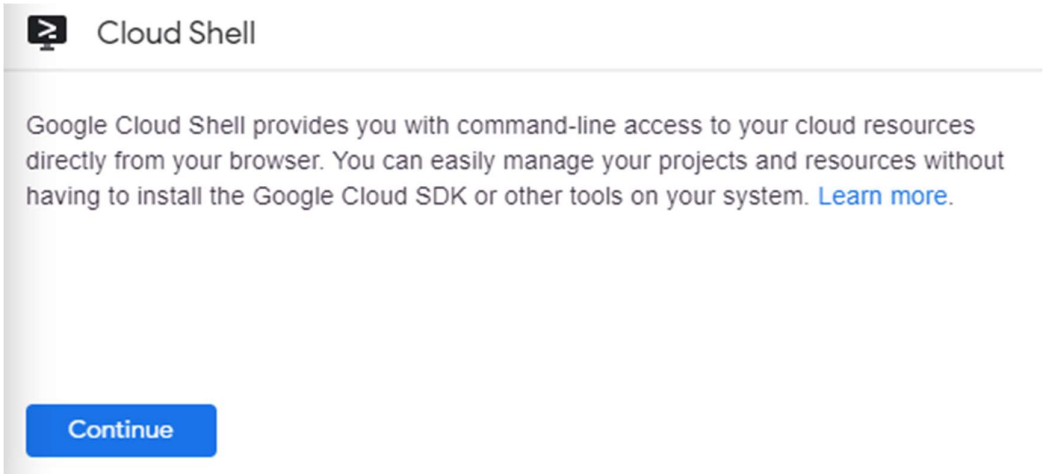
Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

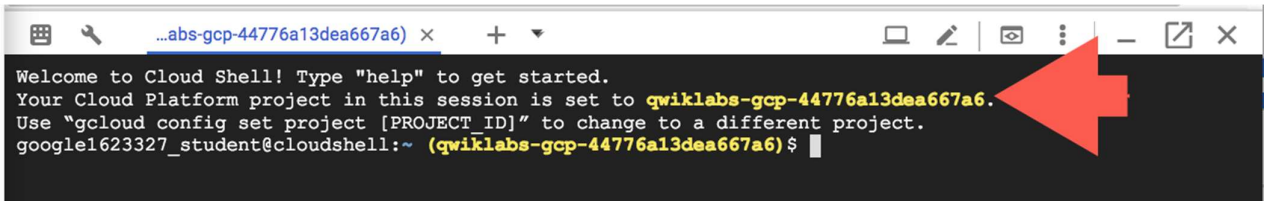
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT_ID*. For example:



`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

```
gcloud auth list
```

(Output)

```
Credentialed accounts:
 - <myaccount>@<mydomain>.com (active)
```

(Example output)

```
Credentialed accounts:
 - google1623327_student@qwiklabs.net
```

You can list the project ID with this command:

```
gcloud config list project
```

(Output)

```
[core]
project = <project_ID>
```

(Example output)

```
[core]
project = qwiklabs-gcp-44776a13dea667a6
```

For full documentation of `gcloud` see the gcloud command-line tool overview.

Deploy the application

Use Cloud Shell command line to deploy a sample Google App Engine application called bookshelf. This web application generates logs for us to examine.

1. Clone the source code repository for the bookshelf application:

```
git clone https://github.com/GoogleCloudPlatform/getting-started-python
```

2. Navigate to the cloned repo:

```
cd getting-started-python/bookshelf
```

3. Install dependencies with pip:

```
virtualenv -p python3 env

source env/bin/activate

pip3 install -r requirements.txt
```

4. Deploy the bookshelf application:

```
gcloud app deploy
```

5. Select a Region close to you:

```
Please choose the region where you want your App Engine application
located:

 [1] asia-east2    (supports standard and flexible and search_api)
 [2] asia-northeast1 (supports standard and flexible and search_api)
 [3] asia-northeast2 (supports standard and flexible and search_api)
 [4] asia-northeast3 (supports standard and flexible and search_api)
 [5] asia-south1   (supports standard and flexible and search_api)
 [6] asia-southeast2 (supports standard and flexible and search_api)
 [7] australia-southeast1 (supports standard and flexible and search_api)
 [8] europe-west   (supports standard and flexible and search_api)
 [9] europe-west2  (supports standard and flexible and search_api)
 [10] europe-west3  (supports standard and flexible and search_api)
 [11] europe-west6  (supports standard and flexible and search_api)
 [12] northamerica-northeast1 (supports standard and flexible and search_api)
 [13] southamerica-east1 (supports standard and flexible and search_api)
 [14] us-central    (supports standard and flexible and search_api)
 [15] us-east1      (supports standard and flexible and search_api)
 [16] us-east4      (supports standard and flexible and search_api)
 [17] us-west2      (supports standard and flexible and search_api)
 [18] us-west3      (supports standard and flexible and search_api)
 [19] us-west4      (supports standard and flexible and search_api)
 [20] cancel
```

6. Then hit "Y" to continue. After a few minutes, the app is fully deployed.

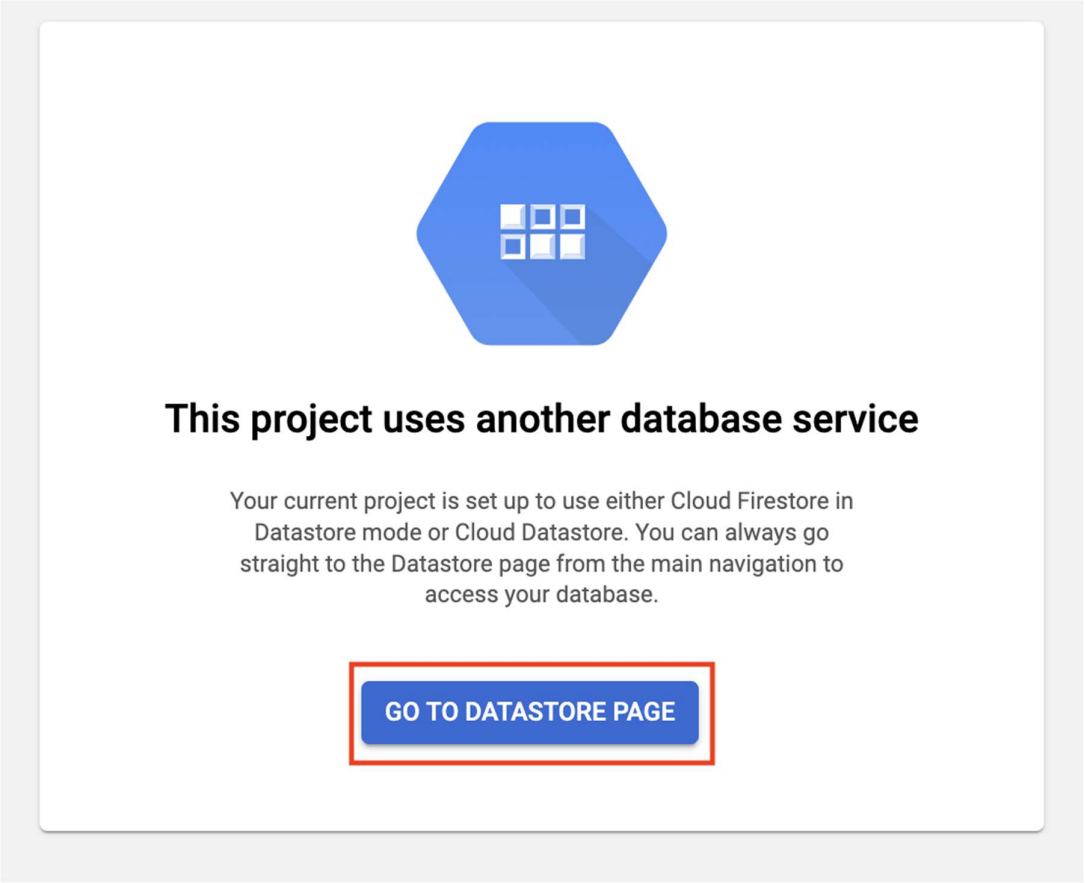
```
You can stream logs from the command line by running:
  $ gcloud app logs tail -s default

To view your application in the web browser run:
  $ gcloud app browse
(env) student_03_45b8883dfb70@cloudshell:~/getting-started-python/bookshelf (qwiklabs-gcp-03-9ad18feb1c46)$
```

Setup your database

1. Open the **Navigation menu** and select **Firestore** from the list of services.

2. Then select **Go to Datastore page**:



3. Click **SWITCH TO NATIVE MODE**, and click **SWITCH MODES** to confirm.

4. Your app is now ready to use Firestore as its database. Return to your Cloud Shell session and run the following command to redeploy your application:

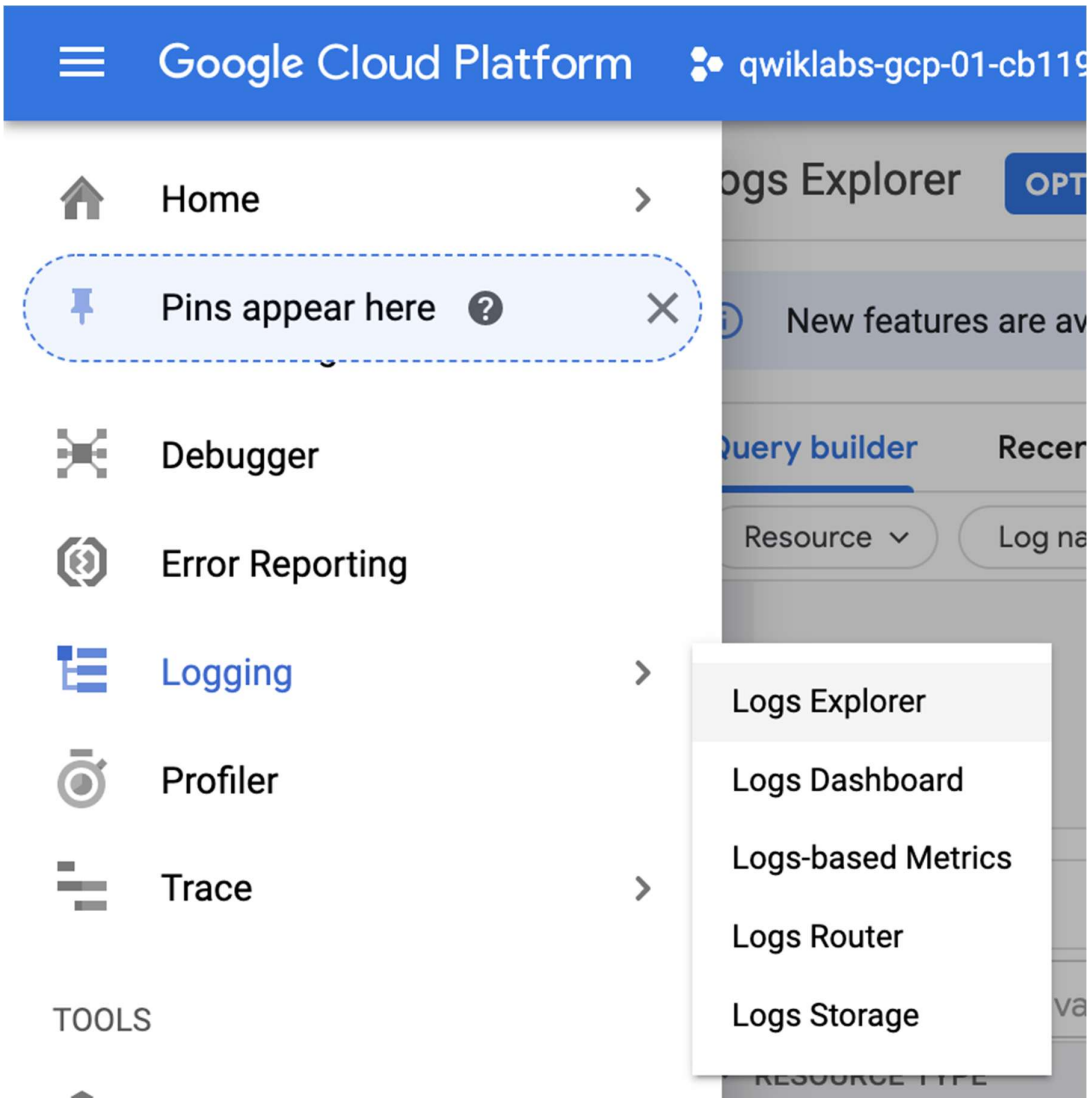
```
gcloud app deploy
```

5. Then hit "Y" to continue. After a few minutes, the app is fully deployed.

Viewing and searching logs

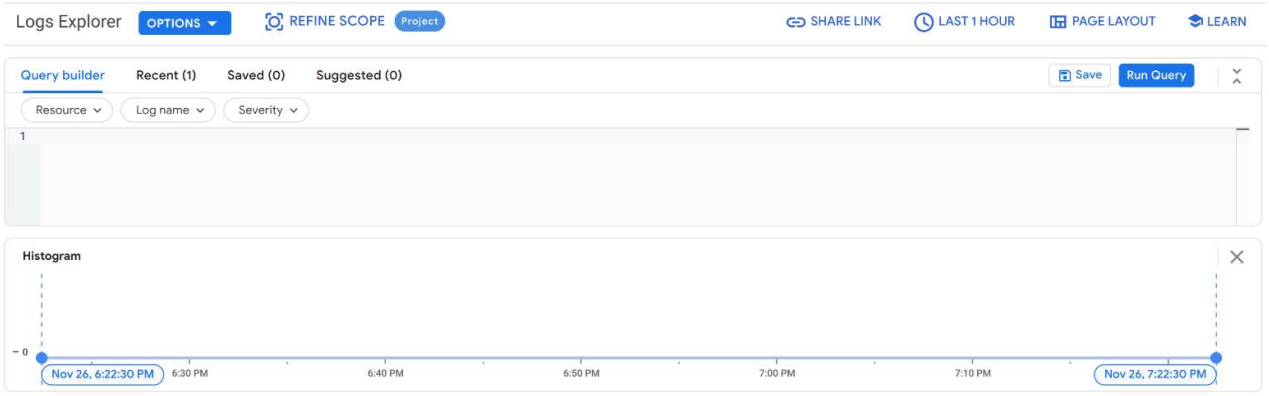
Navigate to Cloud Logs Explorer to configure which logs you view.

Select **Navigation menu** > **Logging** > **Logs Explorer**.



The Cloud Logging console has the following features:

- Resource selector: filters by resource types
- Log selector: filters to specific log types of the resources selected
- Severity selector: filters to specific log levels
- Histogram
- A search box for text, label, or regular expression search, advanced filter



Generate logs

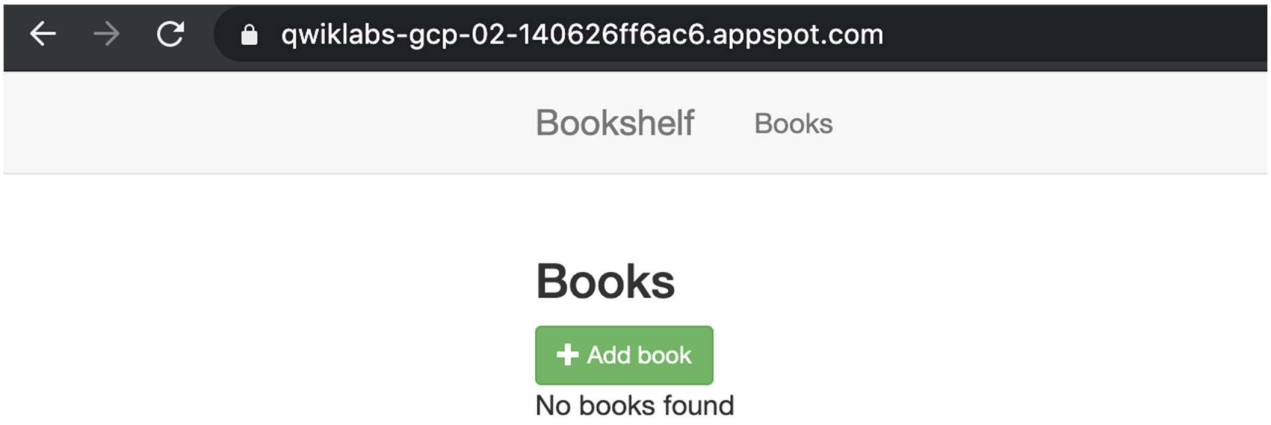
Generate logs to view by visiting the Google App Engine web app (bookshelf) you provisioned earlier.

1. In a new tab, launch the bookshelf application. The URL for your application is: `https://<PROJECT_ID>.appspot.com`

Replace **<PROJECT_ID>** with your **Project ID** in the left panel of this lab.

Optionally, you may copy and paste the complete URL from Cloud Shell (it was output as target url when you launched the App Engine app).
If you see an **Internal Server Error**, this is because the Datastore Index is not yet ready. Wait a minute and reload your browser.

Expected Result:



When you see the App Engine Bookshelf in your browser tab, your App Engine application is deployed and verified. Let's generate some logs!

Click *Check my progress* to verify the objective.

2. Refresh the browser and click **Add book**. Then fill out the form like the following and click **Save**:

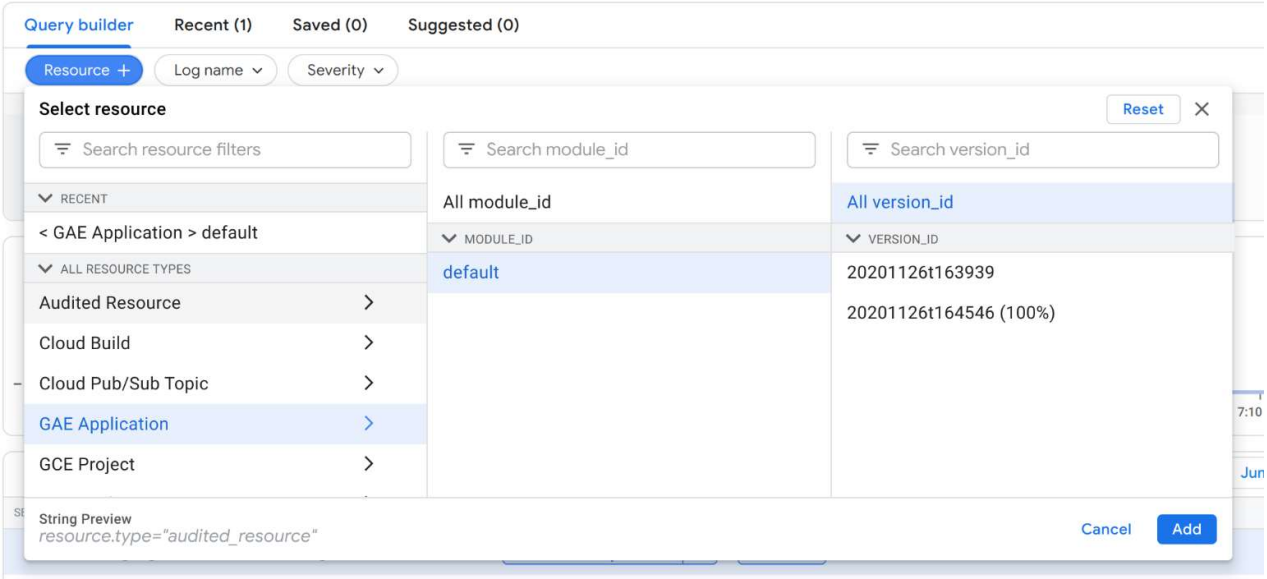
3. Return to the Cloud Logs Viewer.

Filters

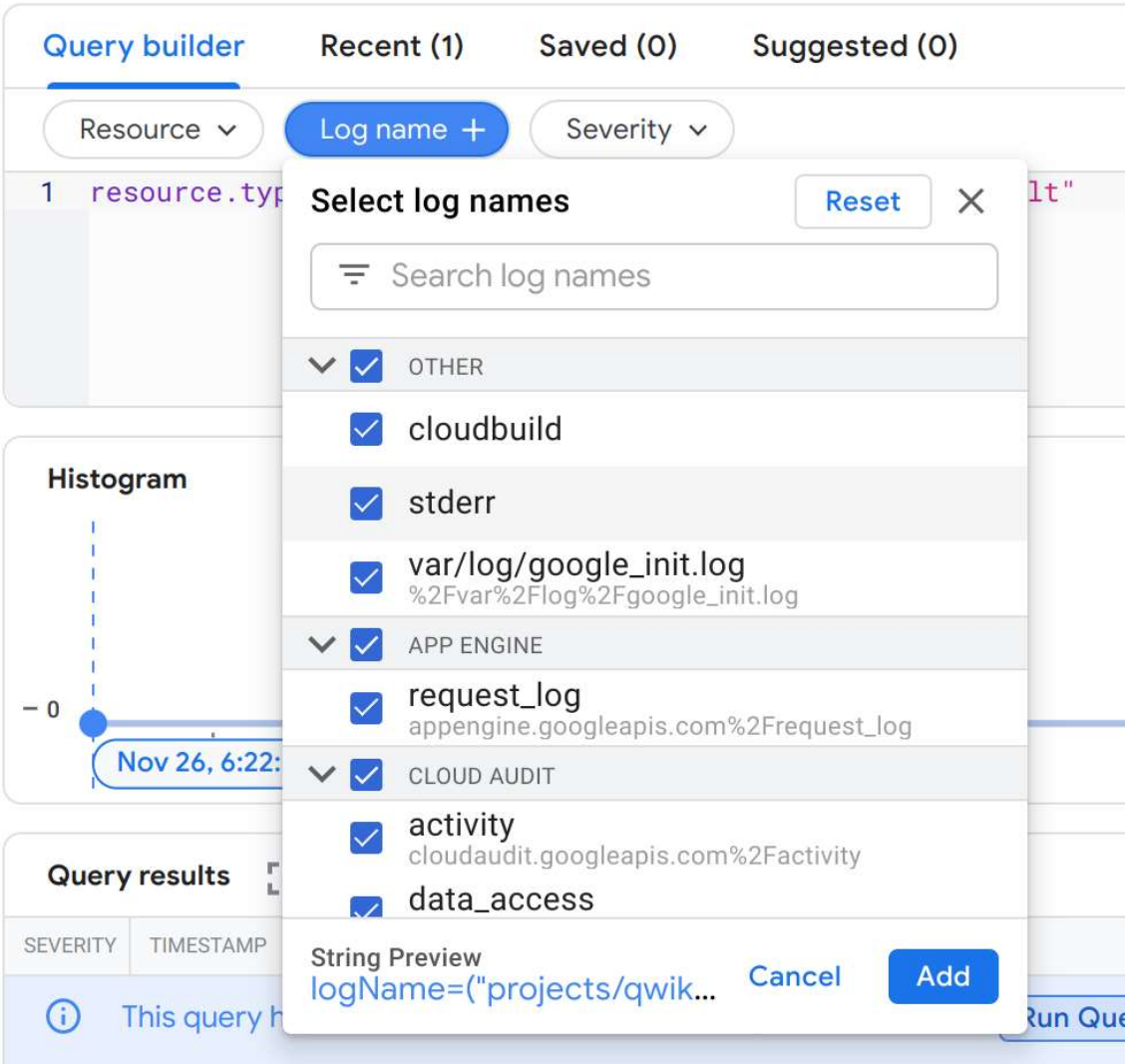
The Logs Explorer provides a variety of **basic filters** and **advanced filters** to tailor your search.

Basic filters

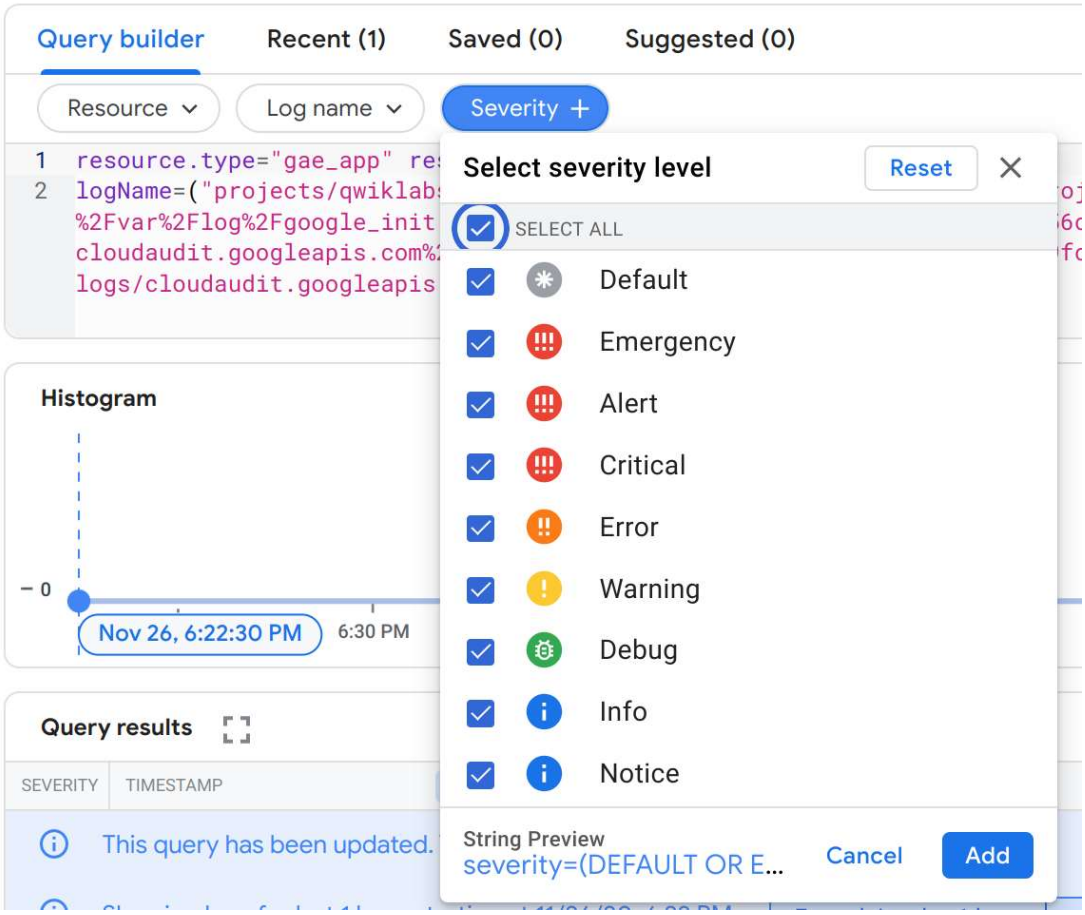
1. Still in the Logs Explorer, in the first (Resource selector) dropdown, select **GAE Application** > **default** > **All version_id** as the service for which you want to view the logs. Click **Add**. This displays all the logs for your bookshelf app.



2. In the next (Log name selector) dropdown, select all the log names and click **Add**.



3. In the next (Severity selector) dropdown, click on **SELECT ALL** and click **Add**.



4. Your Query builder should look like this:

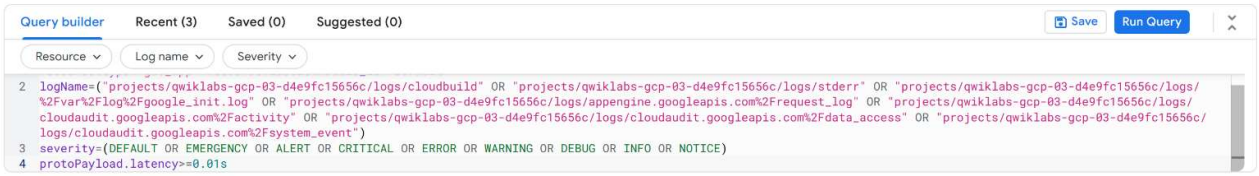
5. Click on **Run Query** button in the top right of the Query builder.

Advanced filters

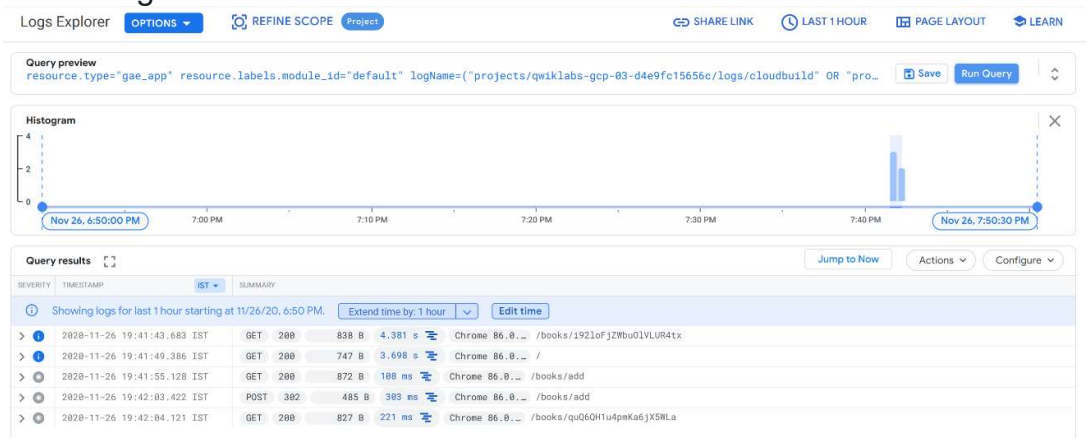
1. In the Query builder text area, add the following new line.

protoPayload.latency>=0.01s

This line displays all GAE app logs with latency of greater than or equal to 0.01seconds.



2. Click **Run Query** and review the updated list of log entries, which show page loads longer than 0.01s.



3. Remove the filters by clearing all the text from the Query builder text area and click **Run Query**.

Log based metrics

Log-based metrics are Cloud Monitoring metrics based on the content of log entries. Therefore, your logs don't just sit around and wait for someone to notice problems; Cloud Monitoring automatically monitors the logs for events and information you define in monitoring metrics. Log-based metrics are also a great way to achieve monitoring of your custom applications. If your application can write logs to a VM's filesystem, you can build monitoring on top of them!
Cloud Logging provides two kinds of user-defined logs-based metrics
- **Counter** and **Distribution**.

Counter metrics

Counter metrics count the number of log entries matching an advanced logs filter. For example, a metric that counts log entries representing certain types of errors from specific resources. Want to be alerted if a lot of your website visitors are receiving HTTP 500 errors? Counter metrics can help.

Distribution metrics

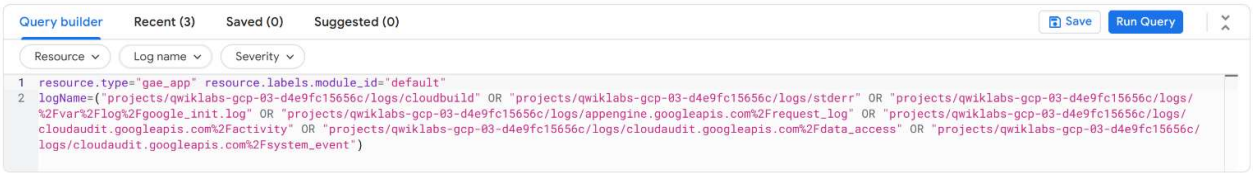
Distribution metrics accumulate numeric data from log entries matching a filter, and perform mathematical calculations against them. A common use for distribution metrics is to track latency patterns/trends over time. As each log entry is received, a latency value is extracted from the log entry and added to the distribution. At regular intervals, the accumulated distribution is written to Cloud Monitoring.

Create a counter metric

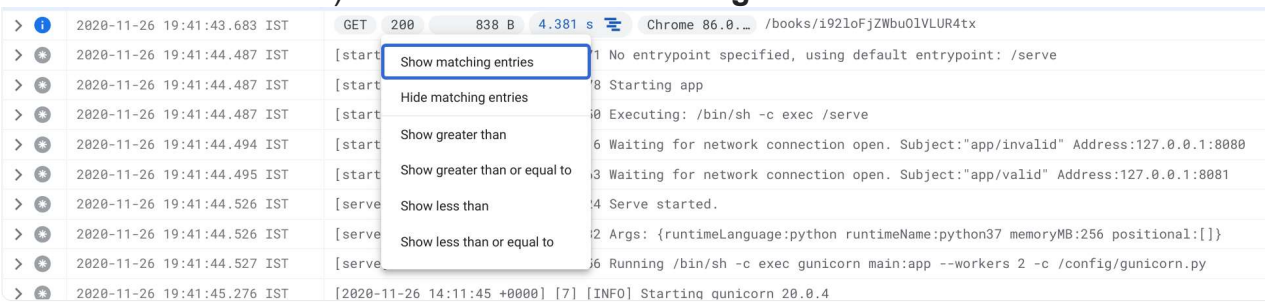
In this section, you create a counter metric to count the number of successful website hits - in this case, all logs with HTTP response status = 200.

1. Still in Logs Explorer, in the **Resource** selector dropdown, select **GAE Application** > **default** > **All version_id** and click **Add**.
2. In the **Log name** selector dropdown, select all the log names and click **Add**.

Your Query builder should look like this:



3. Click on **Run Query** button. In the Query results, click on the status "200" (in any row that has 200) and select **Show matching entries**:

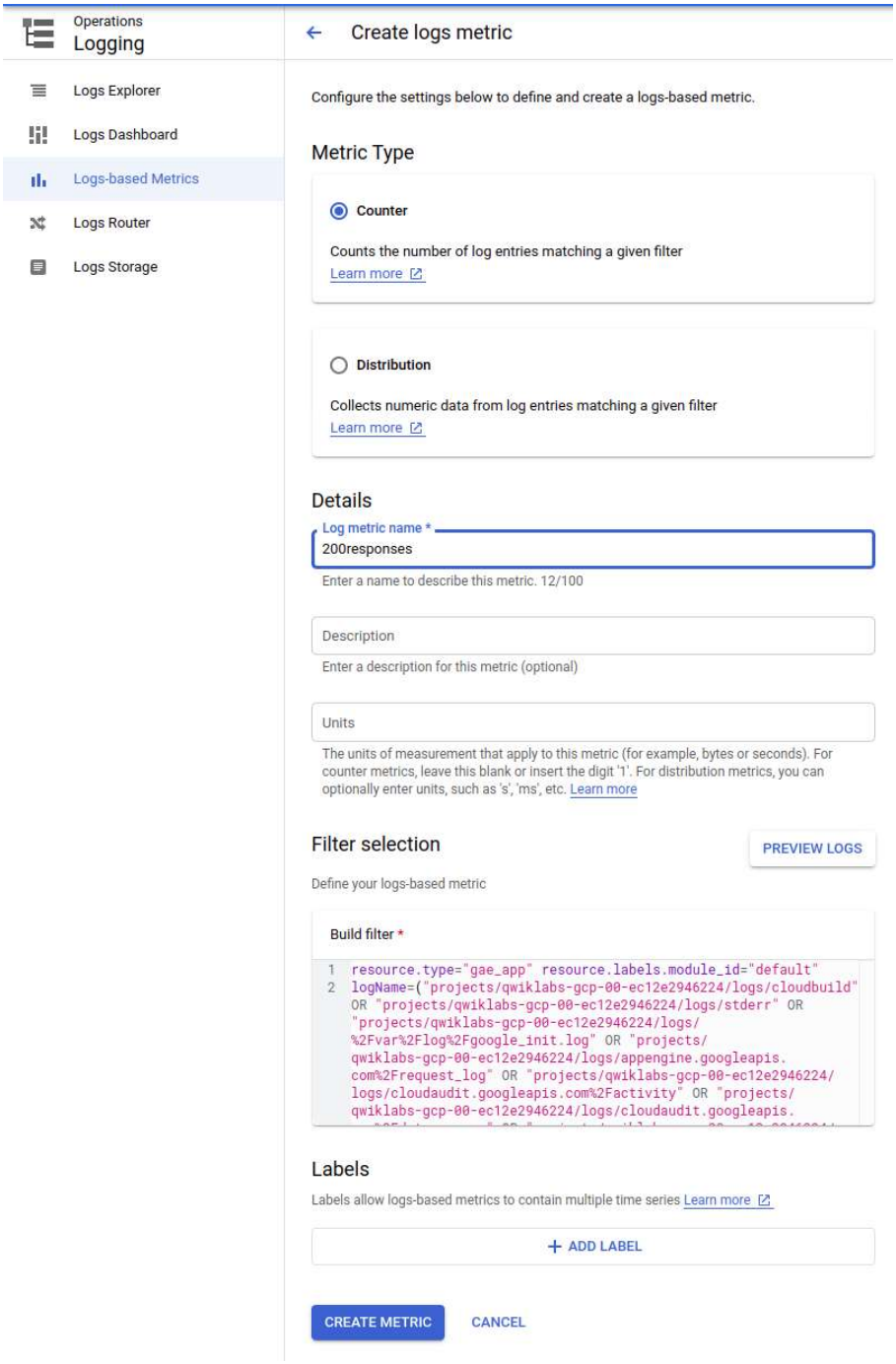


You'll see:

- The list displays only the logs with a 200 status.
- In the Query build text area, a filter is automatically created with the condition **(protoPayload.status=200 OR httpRequest.status=200)**.

Create a monitoring metric based on your filter:

1. Click on **Actions** dropdown in the top right of the Query results section.
2. Select **Create Metric** to create a monitoring metric based on this filter.
3. In the Metric Editor, set **Metric Type** as **Counter**.
4. Under the **Details** section, set the **Log metric name** to **200responses**. Leave all the other fields at their default.
5. Click **CREATE METRIC**.



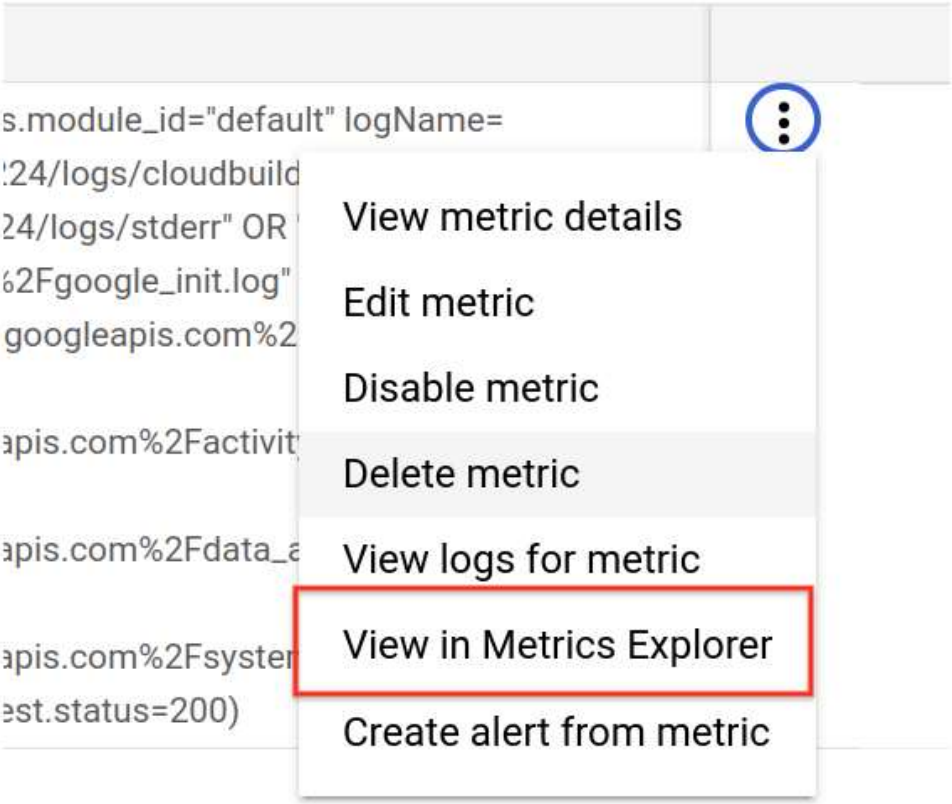
Click *Check my progress* to verify the objective.

Create a counter metric

Check my progress

6. The Logs-based metrics window opens and lists your new metric under the User-defined Metrics section.

7. View metric details by clicking the vertical ellipsis on the metric line > **View in Metrics Explorer**.



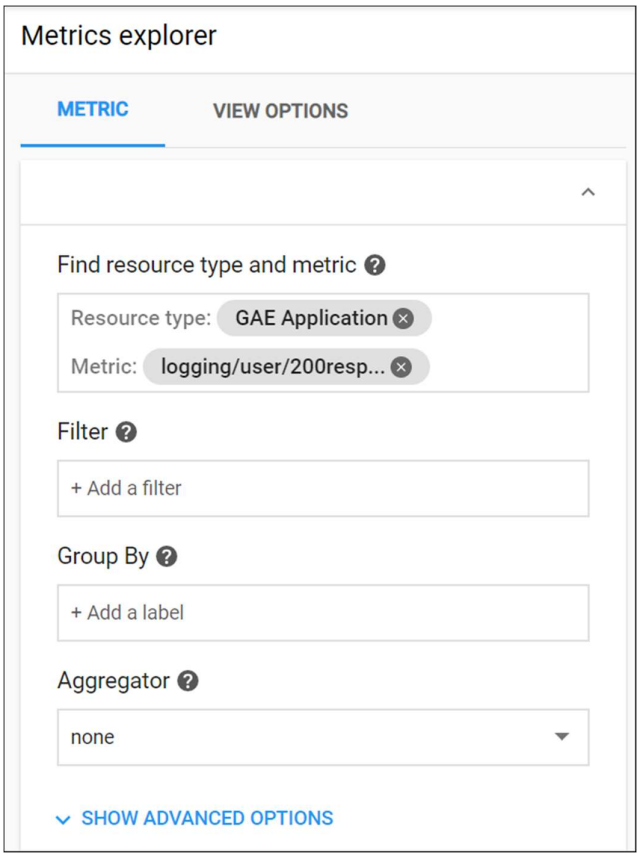
This opens Cloud Monitoring, wait for your Cloud Monitoring workspace to build.

When the Cloud Monitoring window opens, your workspace is ready.

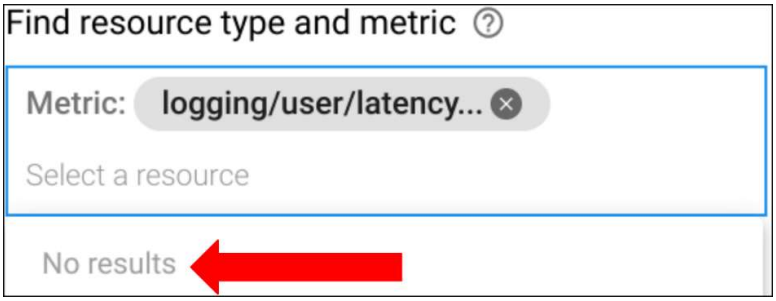
8. In the left menu, select **Metrics Explorer**.

9. In Metrics Explorer:

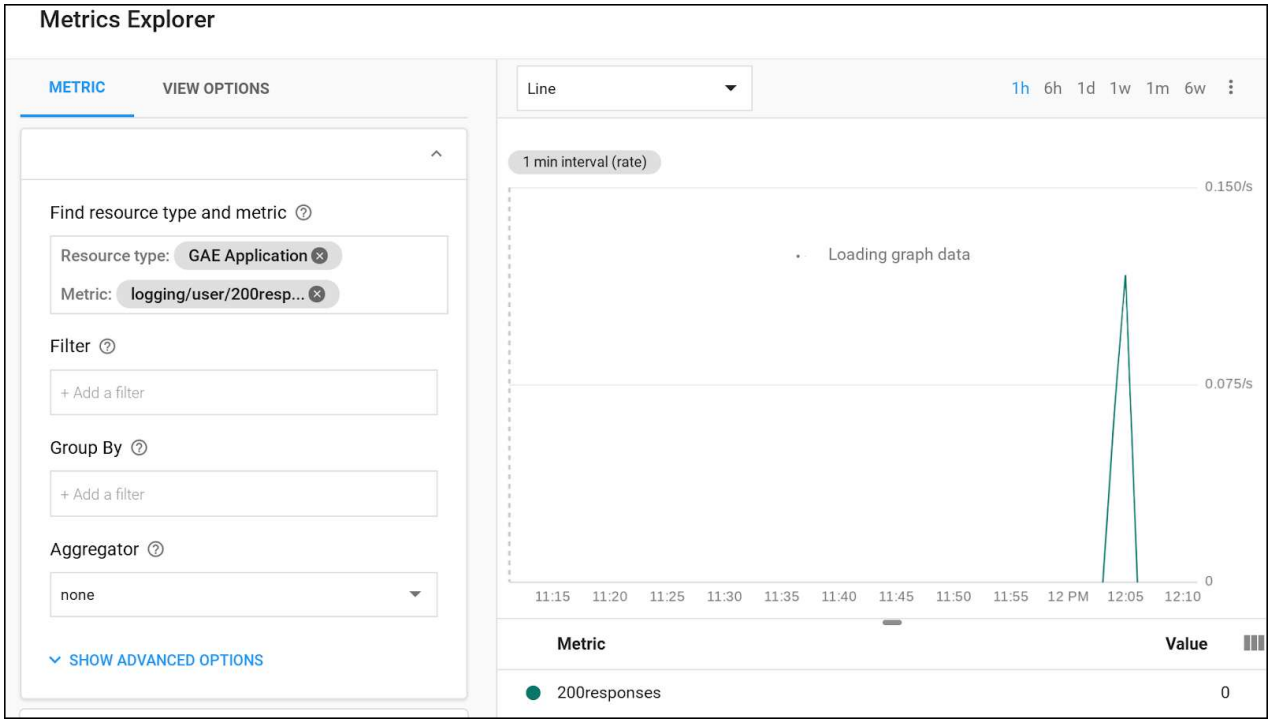
- Set Resource type to **GAE Application**.
- Select **logging/user/200responses** as the metric.



If you don't see GAE Application as a **Resource types** option:



- Close the Metrics Explorer window.
- Refresh your bookshelf app a few times to provide log entries
- Wait a minute for your log entries (after refreshing the App Engine app) to populate. It may take two or three minutes.
- Click the vertical elipse > **View in Metrics Explorer**
  Once the logs have been ingested, you'll see **GAE Application** in the **Resource types** selector.
  This metric is ready to monitor and analyze your application's behavior:



Note: Don't worry if your graph is currently empty—it will be populated as you continue with the lab.

10. Save your chart to a Dashboard to easily check during the lab.

- Click **Save Chart** in the upper right.
- Select **New Dashboard** under **Dashboard**.
- Name your Dashboard.
- Click **Save**.

Click **View Dashboard** to view the dashboard.

Create a distribution metric

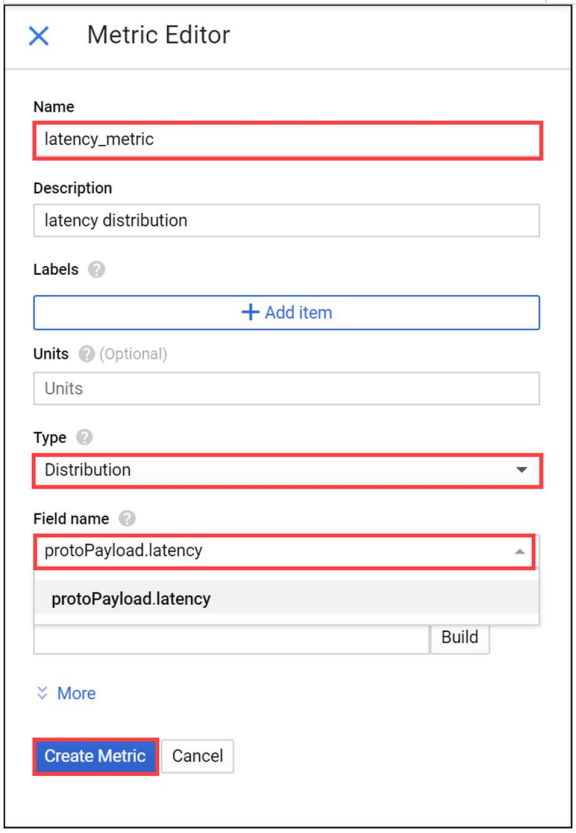
In this section, you create a distribution counter to monitor bookshelf application latency.

1. Return to the Cloud Logs Viewer (**Navigation menu** > **Logging** > **Logs Explorer**). Create a filter to select **GAE Application** > **Default Service** > **All version_id**, **All Logs**, and **All Severity** in the Query builder as shown below and click **Run query**.

2. Click on **Actions** dropdown in the top right of Query results section and select **Create Metric**.

3. In the Metric Editor panel, set the following fields to the values below:

Field	Value
Metric Type	Distribution
Log metric name	latency_metric
Description	latency distribution
Field name	protoPayload.latency



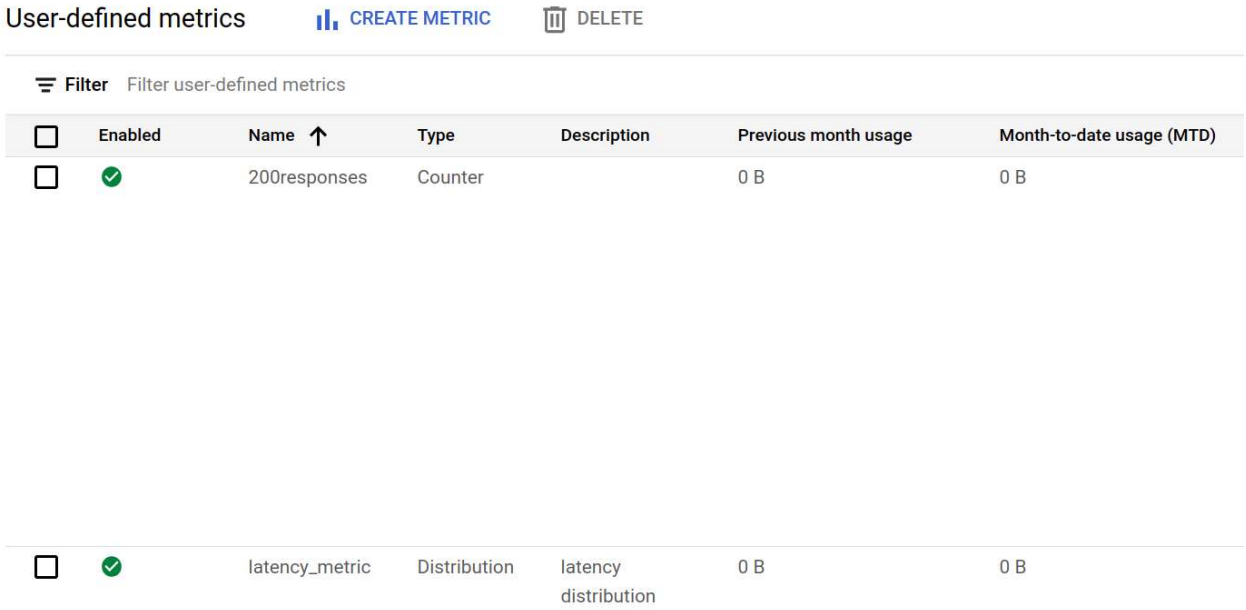
4. Click **Create Metric**.

Click *Check my progress* to verify the objective.

Create a distribution metric

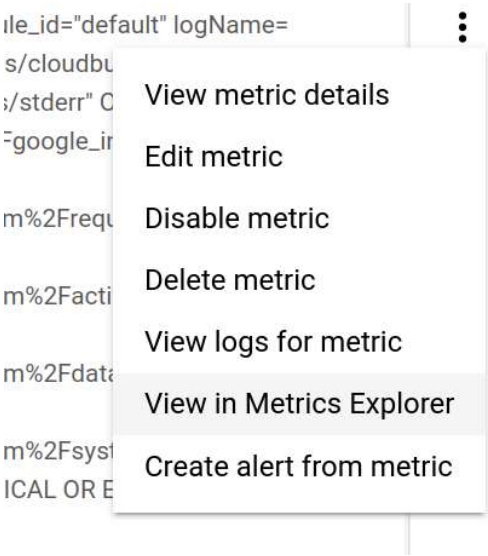
Check my progress

5. Verify the latency metric is created in User-defined Metrics:



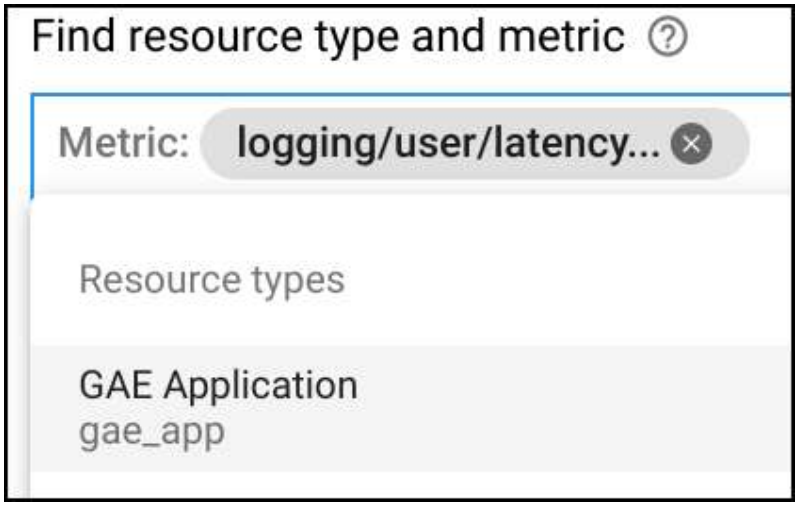
6. Generate more logs. Refresh the bookshelf application multiple times and add another book. Give the metric a minute or two to catch up and accumulate the data points.

7. Click on **View in Metrics Explorer** by selecting the option in the vertical ellipsis menu against the metric:



8. In the left menu, select **Metrics Explorer**.

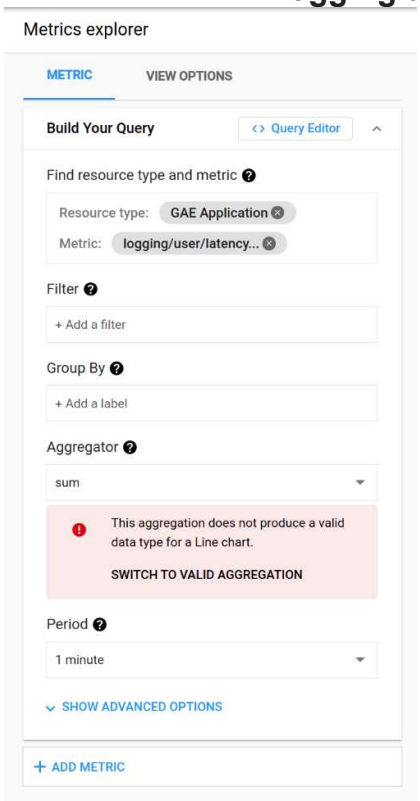
9. As before, click **GAE Application** as the resource type, and make sure the Metric is the one you just created (latency_metric).



If **GAE Application** or the logging/user/latency_metric does not auto-populate:

- Reload the Metrics explorer-Monitoring browser tab.
- Refresh your bookshelf app a few times to provide log entries
- Wait a minute for your log entries (after refreshing the App Engine app) to populate. It may take two or three minutes.
  Once the logs have been ingested, you'll see **GAE Application** in the **Resource types** selector.

10. Once you see **GAE Application** as a **Resource type** option and have selected the **logging/user/latency_metric** metric, you'll see the following message:



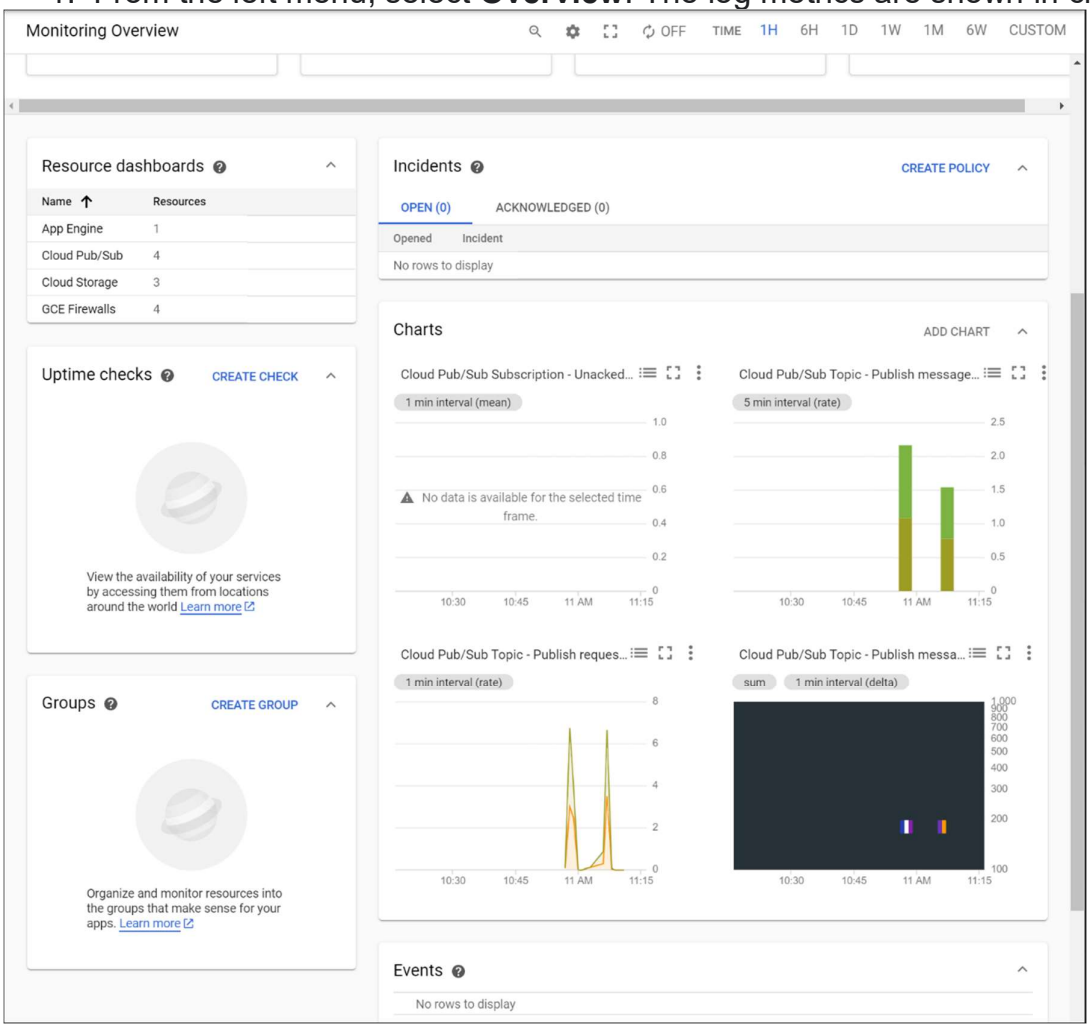
11. Click **SWITCH TO VALID AGGREGATION** to switch to the appropriate aligner and aggregator. You should see latency_metric data in your chart.

12. Optional: Save this chart to your Dashboard and/or check out the Dashboard to see if the chart you previously saved shows data for 200 responses.

View log metrics in the Cloud Monitoring Console

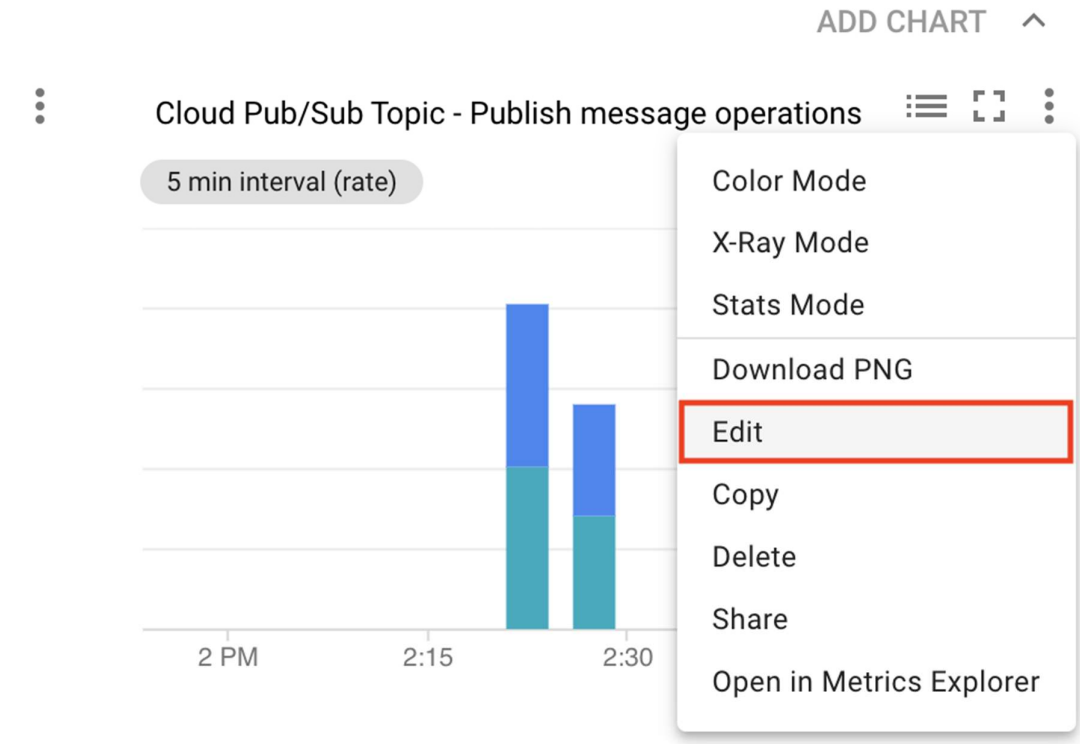
The Cloud Monitoring Overview window provides a monitoring resource overview.

1. From the left menu, select **Overview**. The log metrics are shown in charts.

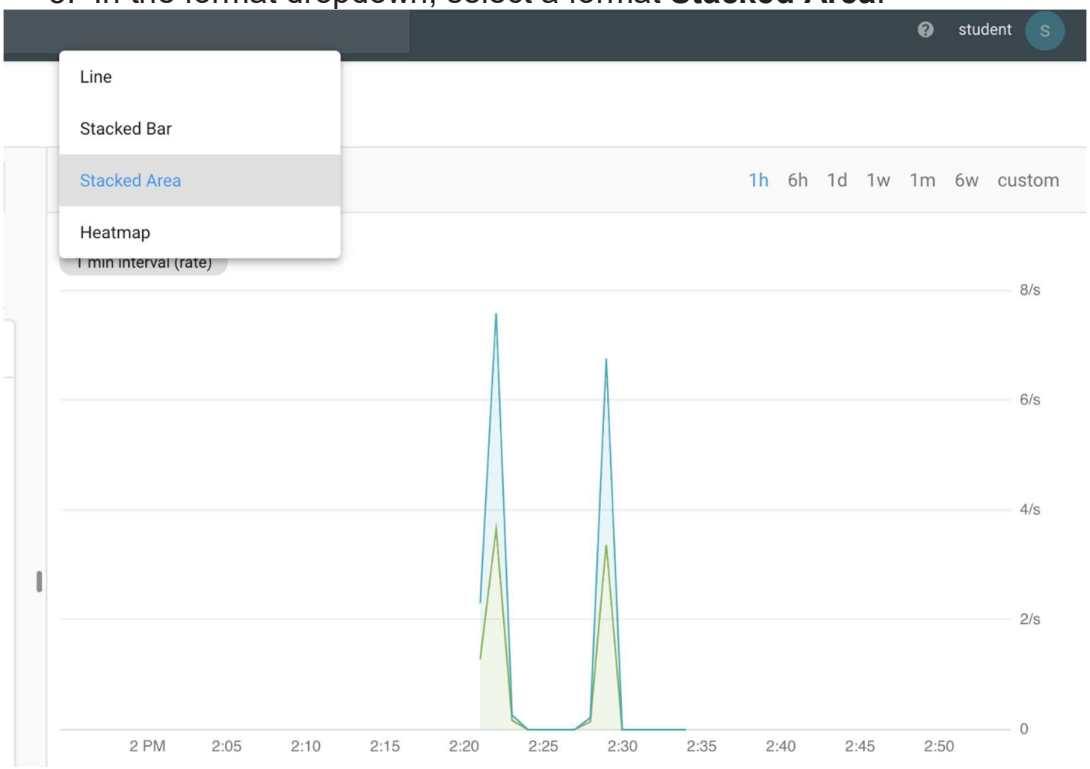


Cloud Monitoring displays the chart data in one of four formats: Line, Stacked Bar, Stacked Area or Heatmap. To specify the format:

2. Click the vertical ellipse for one of the charts then click **Edit**.



3. In the format dropdown, select a format **Stacked Area**.



4. Click **Save**.

Try each of the four views to see which one best represents your latency metric.

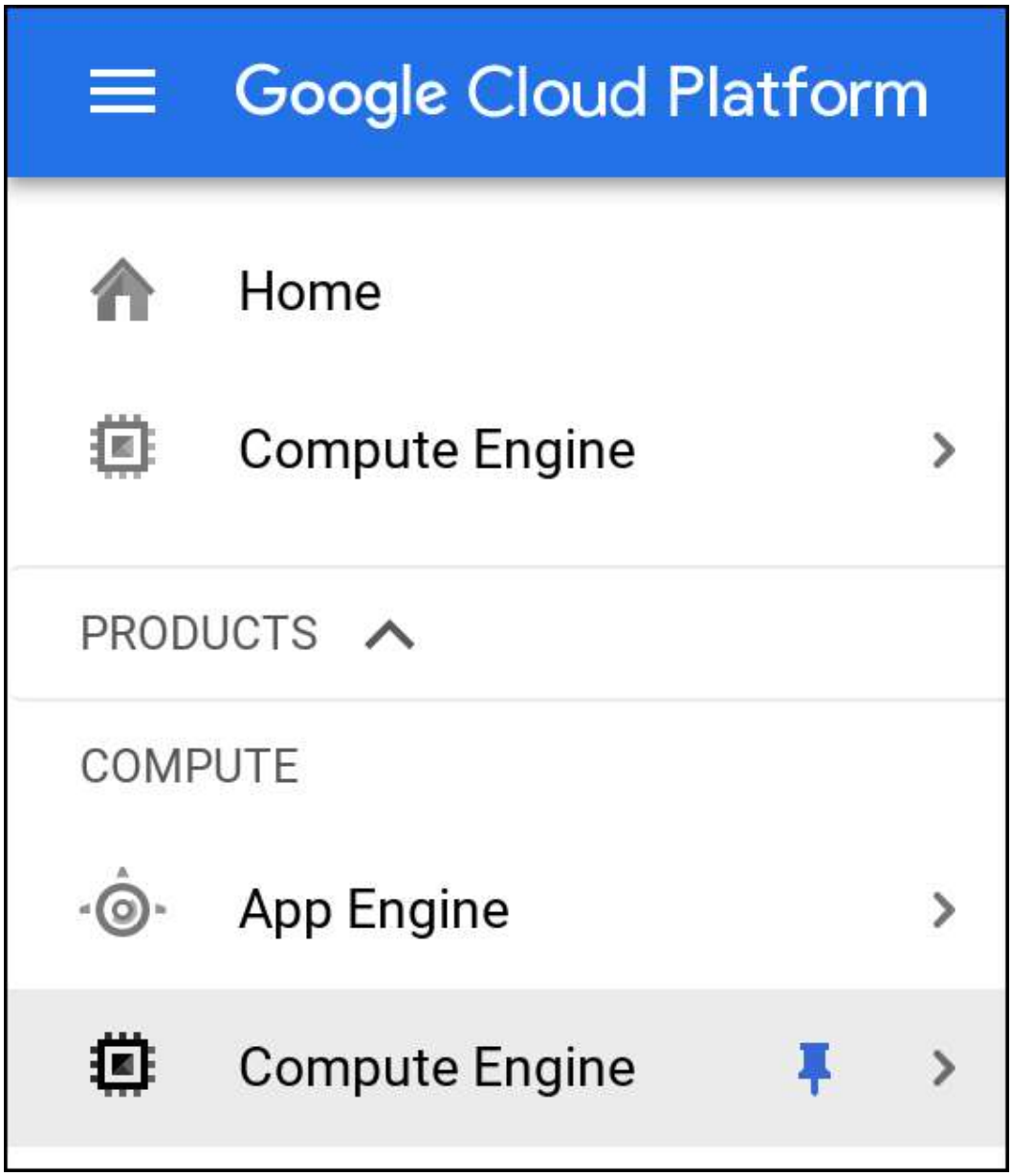
5. Experiment with the other charts. Challenge: can you edit or add one of the charts you made in Metric Explorer?

Audit logging

Google Cloud provides Auditing of all Google Cloud resources by default. The audit logs answer the question "Who did what, when?" Let's look at Audit Logging, starting by creating a new Compute Engine (Compute Engine) virtual machine (VM). Launching a VM is an example of an audited privileged activity, so it generates logs.

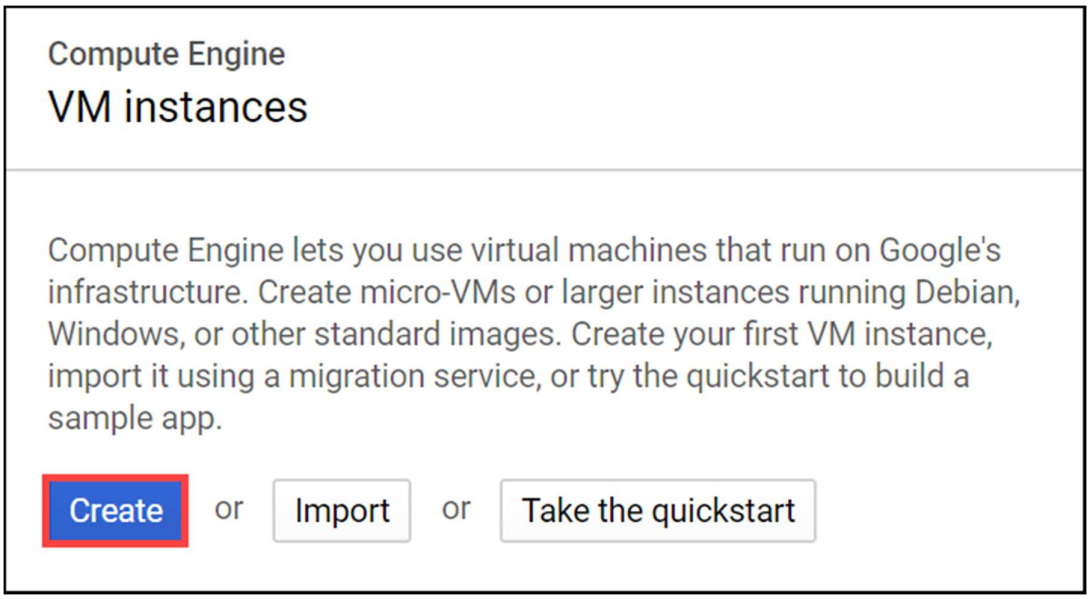
Launch a VM to generate audit log events

1. In the Cloud Console, select **Navigation menu** > **Compute Engine** > **VM instances**.



Wait for the Compute Engine service to initialize.

2. Click **Create**:



3. Set the following field to the values below, leave all others at default.

Field	Value
Series	N1
Machine Type	g1-small (1 vCPU, 1.7 GB memory)
Firewall	check Allow HTTP traffic

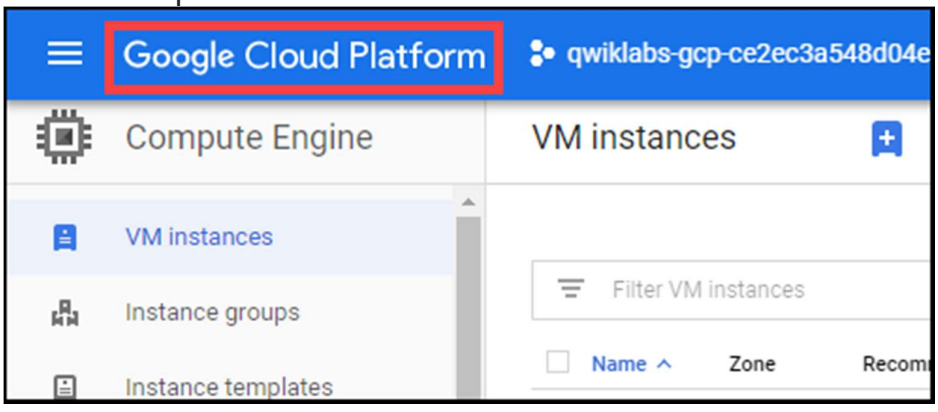
4. Click **Create**.

Click *Check my progress* to verify the objective.

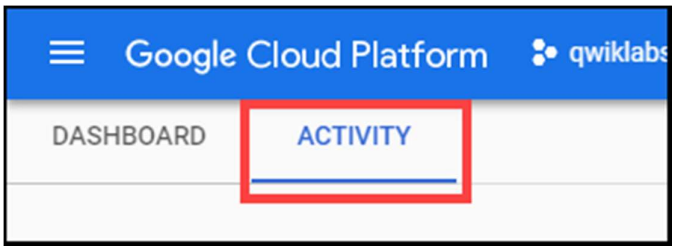
Viewing audit logs in Activity Viewer

Activity Viewer, on the main Google Cloud Dashboard, provides a quick view into audit logs.

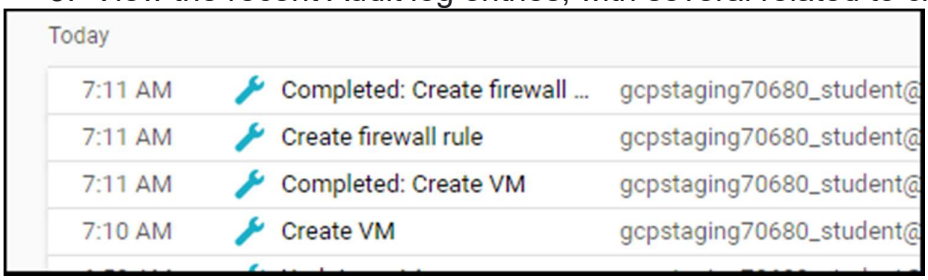
1. Return to the Google Cloud Dashboard by clicking **Google Cloud** in the banner at the top of the console window:



2. Switch to the *Activity* tab. You may have to click **Navigation menu** to close the menu to see the *Activity* tab.



3. View the recent Audit log entries, with several related to creating a VM at the top.



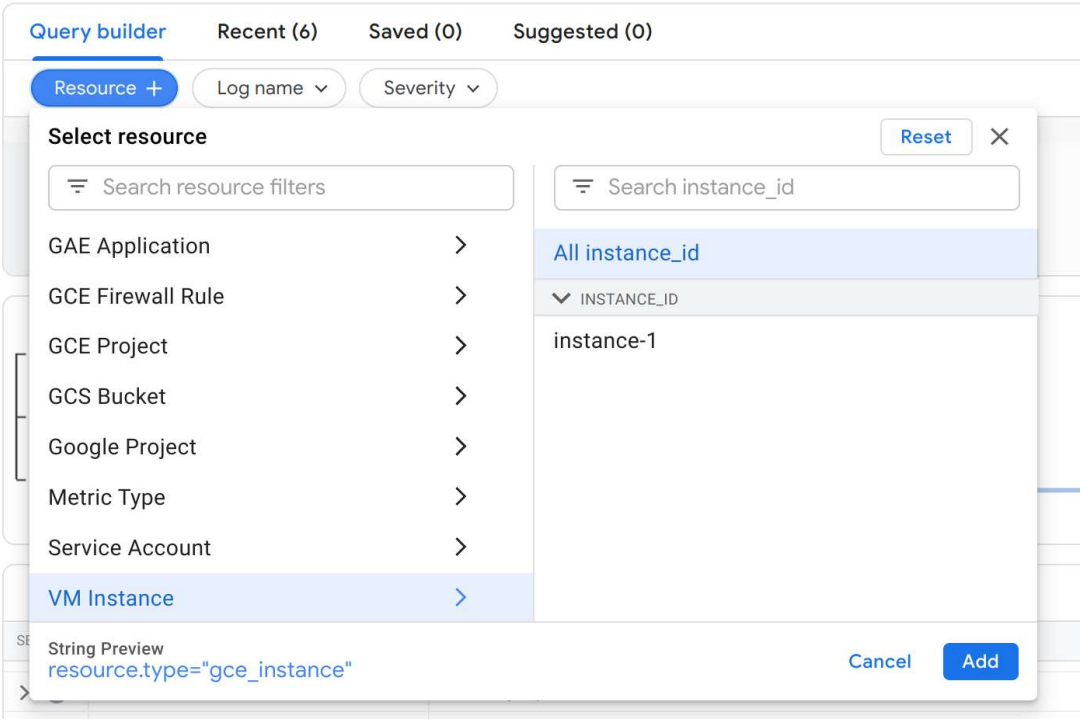
In the screenshot above, notice the four log entries documenting your creation of the VM and the HTTP firewall rule you associated with it.

4. Click on different rows for a few minutes to see what they tell you. Do you recognize any of the previous actions you have taken during this lab?

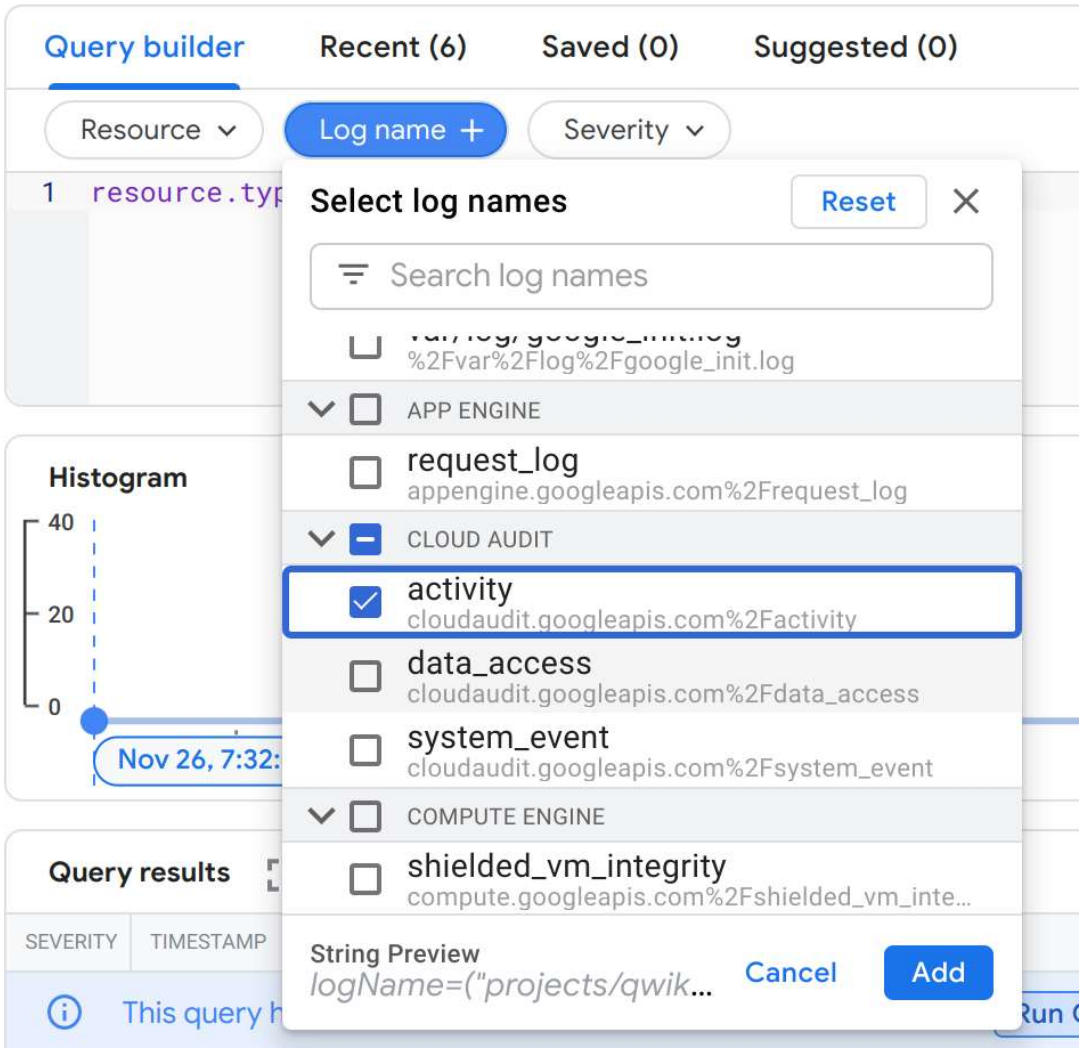
Viewing audit logs in Cloud Logs Viewer

In Cloud Logs Viewer, you can view the same Audit log entries as in Activity Viewer. Logs Viewer is much more versatile, allowing advanced filters and other log management functionality.

1. From the Cloud Console, return to Cloud Logs Viewer (**Navigation menu** > **Logging** > **Logs Explorer**).
2. In the **Resource** selector, select **VM Instance** > **All instance_id** and click **Add**:



3. In the **Log Name** selector dropdown, select **activity** under **CLOUD AUDIT**, and click **Add**:

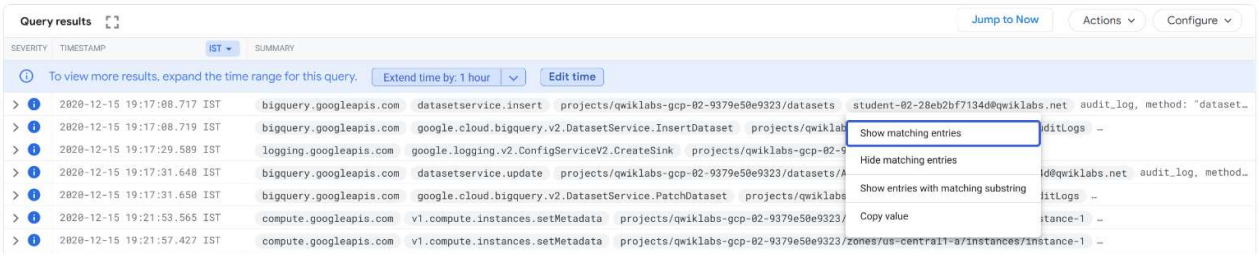


4. Click on **Run Query** button in the top right of the Query builder and view the two Audit log entries that correspond to the **Create VM** and **Completed: Create VM** entries you saw in the Activity Viewer.

5. Expand the **Query Preview** to look at all audit logs for all Google Cloud services. Remove line 1 to remove the gce_instance filter, then click **Run Query**:

View all activities performed in Google Cloud by any user.

7. In any log row, click your Qwiklabs Username (email) and click **Show matching entries**:



This adds a new filter row to the Advanced Filter, and limits the result set to actions performed by you:

```
1 logName="projects/qwiklabs-gcp-02-9379e50e9323/logs/cloudaudit.googleapis.com%2Factivity"
2 protoPayload.authenticationInfo.principalEmail="student-02-28eb2bf7134d@qwiklabs.net"
```

Exporting logs

Cloud Logging retains logs for 30 days, after which they are deleted. To retain logs longer, you should export them to another storage system, or "sink", such as BigQuery. Cloud Logging allows you to set up automated exporting jobs so that all logs will automatically be exported. Logs may then be further analyzed with the features of your chosen sink.

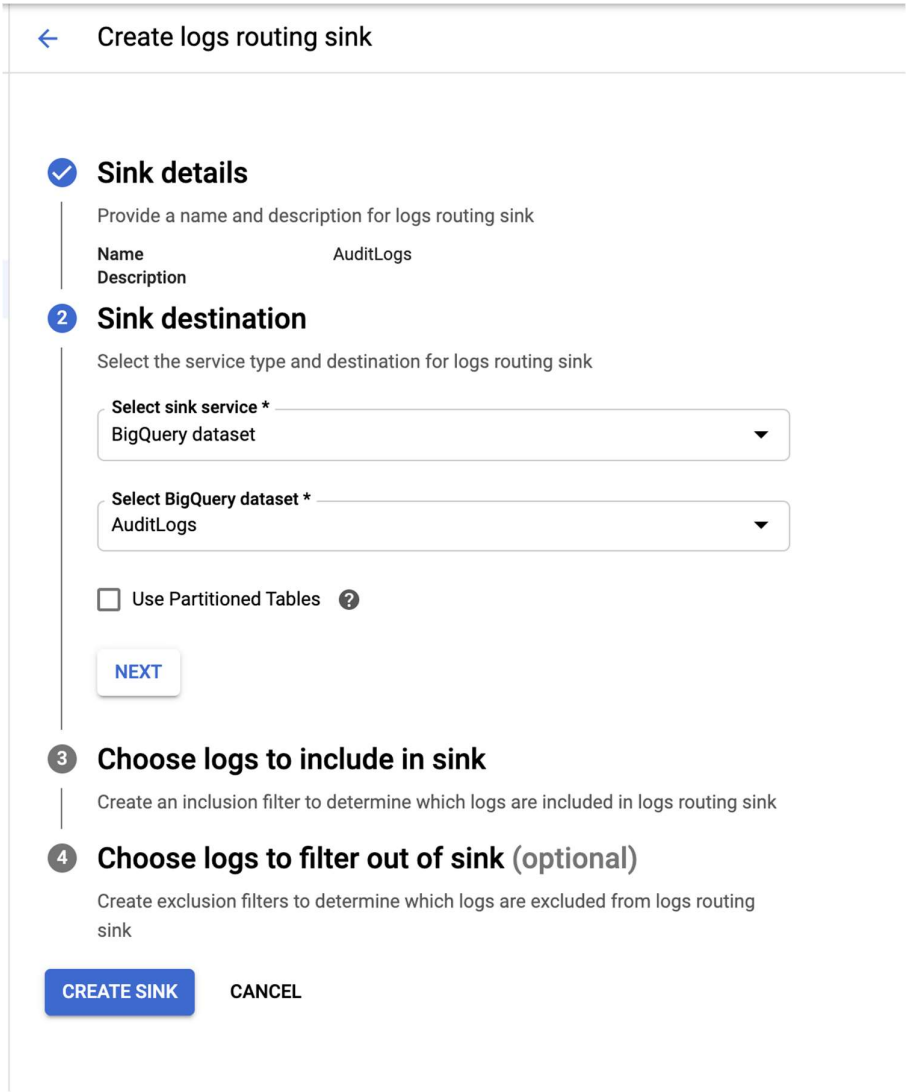
Creating an export job

Set up an export job to send all audit logs to BigQuery for long-term storage and analysis.

1. In the Cloud Logs Explorer window, remove line 2 from the Query builder and click **Run Query**, so that you are viewing all audit logs for your Google Cloud Project:

2. Click on **Actions** dropdown in the top right of Query results section. Click **Create Sink** and set the following fields to the values below.

Field	**Value**
Sink Name	AuditLogs then click Next.
Select Sink Service	BigQuery dataset
Select BigQuery dataset	Create new BigQuery dataset and then name the new BigQuery dataset "AuditLogs" and click **Create**.

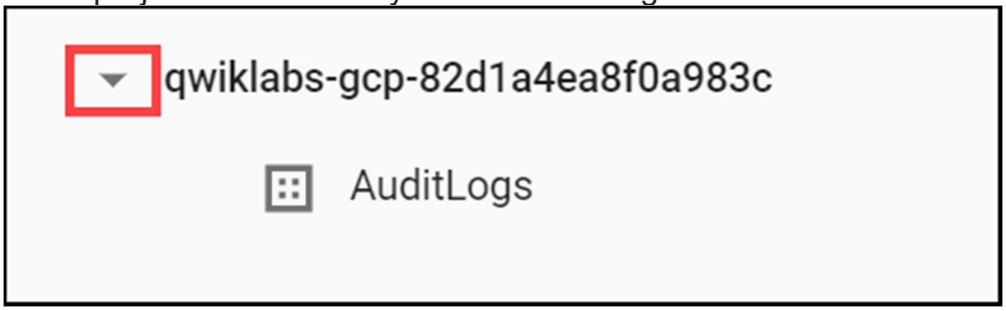


3. Click **Create Sink**. Click **Close** to exit the Sink Created confirmation.

Click *Check my progress* to verify the objective.

Viewing audit logs in BigQuery

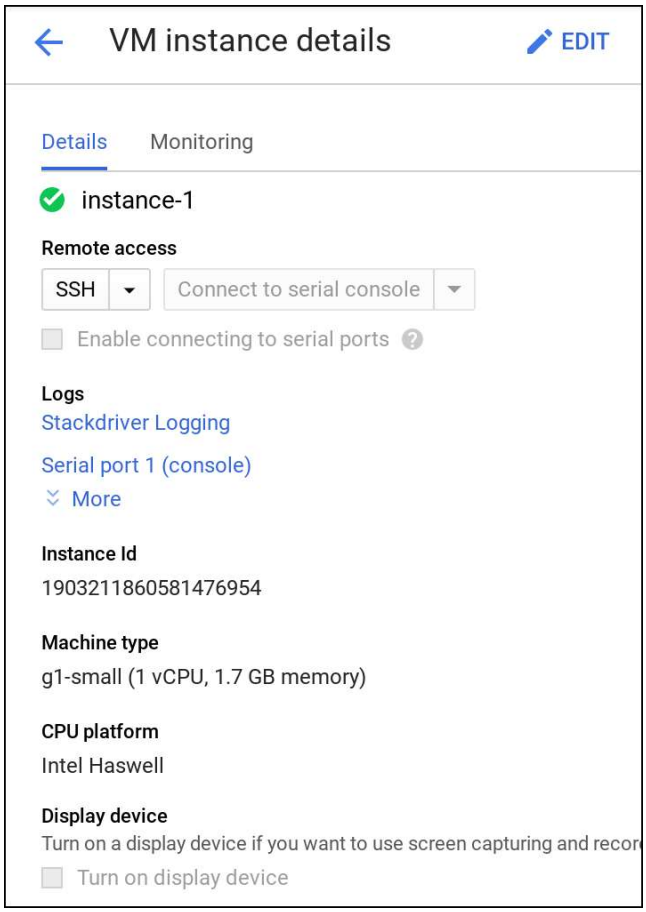
1. Launch BigQuery. Select **Navigation menu** > **BigQuery**.
2. Click **Done** to close the Welcome banner.
3. In the left menu, click the arrow next to your Project ID to expand your Google Cloud project name to see your new AuditLogs dataset:



Notice that AuditLogs has no tables under it yet. Log Exporting begins to send data to the sink after the export job is created. Generate some audit log entries, which will create a table in the sink and add rows to it.

4. Return to the VM instances window (**Navigation menu** > **Compute Engine** > **VM instances**).

5. Click on your Compute Engine VM instance to view it's details.



6. Click **Edit** at the top to make two small changes to the VM:

- Check the checkbox for **Enable connecting to serial ports**.
- Scroll down and check the checkbox to **Allow HTTPS traffic**.

7. Click **Save**.

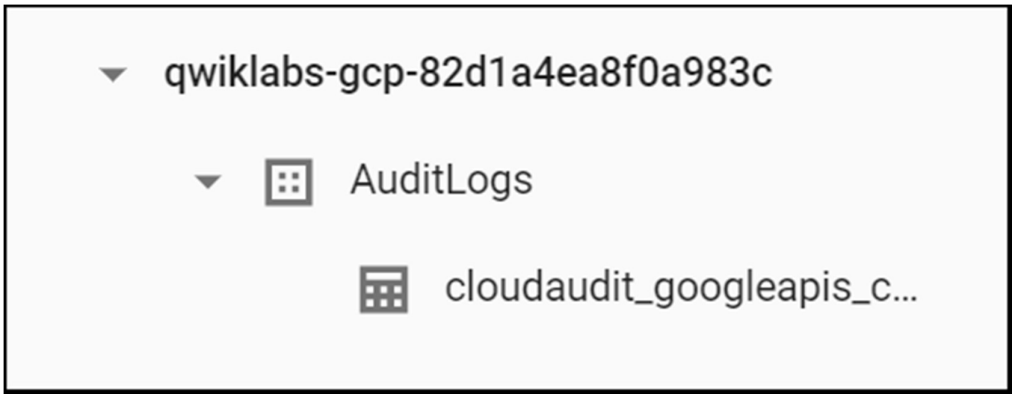
Go to the *Activity* tab on the main Google Cloud Dashboard. You should see several Audit Log entries, including one named "Set metadata on VM", another named "Create firewall rule" and others related to your VM changes. You'll also see an event named "Create Table" indicating that the BigQuery sink was created.

After a minute or so (you may need to refresh the page) you'll see Audit Log entries indicating that the BigQuery table has been updated with the new Audit Log entries you

just generated by editing the VM. Look at the timestamps to recognize all of the log entries related to your VM edit action and the associated BigQuery capture.

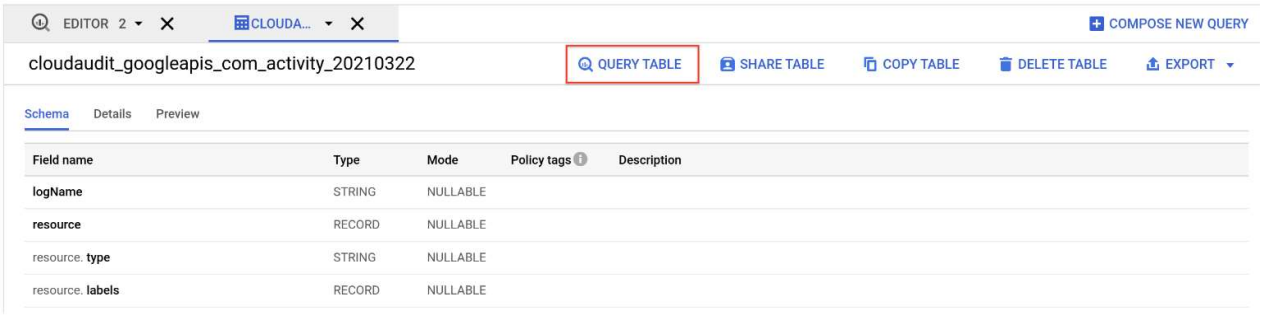
Click *Check my progress* to verify the objective.

8. Return to the BigQuery console (**Navigation menu** > **BigQuery**) and expand the AuditLogs dataset. You might need to refresh the page. You should see that a new **cloudaudit** table has been created in the dataset. Click the new table:



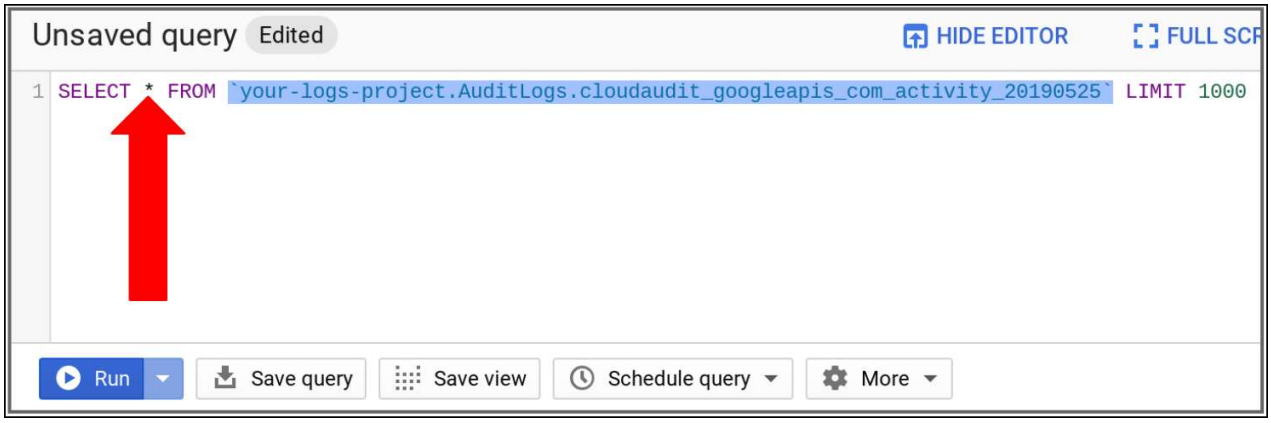
Use BigQuery to explore the audit logs.

1. Click the new **cloudaudit** table, then click on **QUERY TABLE**:



The Query Editor text box is pre-populated with a partial SQL query, with the cursor between "SELECT" and "FROM":

2. Type an asterisk (*) between "SELECT" and "FROM", then click **Run**:



After a few seconds the query completes and you see the Audit Log entries in the bottom Results pane. There are many columns, some of which are nested.

3. Click anywhere in the Results pane, then use the arrow keys to scroll right and left. Look around a bit—audit logs are very detailed!

Now define a narrower query to view just a summary of each audit entry.

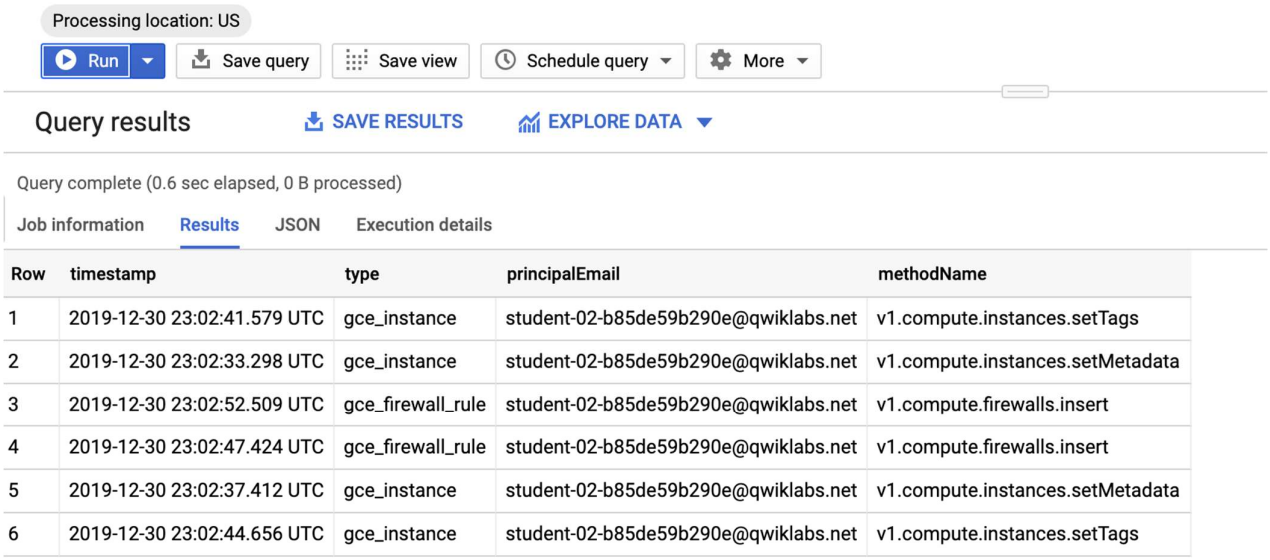
4. Copy the following code into the Query Editor:

```
SELECT
timestamp,
resource.type,
protopayload_auditlog.authenticationInfo.principalEmail,
protopayload_auditlog.methodName
FROM `<your-project-ID>.AuditLogs.<your_audit_log_table_name>`
WHERE protopayload_auditlog.authenticationInfo.principalEmail =
"<your_qwiklabs_username_email>"
LIMIT 1000
```

5. Replace the following parameters with the values below:

Parameter	Value
`<your-project-ID>`	Project ID (found in the left panel of the lab)
`<your_audit_log_table_name>`	audit log table name (found under AuditLogs in the BigQuery Resources section)
`<your_qwiklabs_username_email>`	Username (found in the left panel of the lab)

6. Click **Run** to run the query. You should see a smaller set of columns, limited to actions you performed in Google Cloud. Your results should be very similar to the below:



Row	timestamp	type	principalEmail	methodName
1	2019-12-30 23:02:41.579 UTC	gce_instance	student-02-b85de59b290e@qwiklabs.net	v1.compute.instances.setTags
2	2019-12-30 23:02:33.298 UTC	gce_instance	student-02-b85de59b290e@qwiklabs.net	v1.compute.instances.setMetadata
3	2019-12-30 23:02:52.509 UTC	gce_firewall_rule	student-02-b85de59b290e@qwiklabs.net	v1.compute.firewalls.insert
4	2019-12-30 23:02:47.424 UTC	gce_firewall_rule	student-02-b85de59b290e@qwiklabs.net	v1.compute.firewalls.insert
5	2019-12-30 23:02:37.412 UTC	gce_instance	student-02-b85de59b290e@qwiklabs.net	v1.compute.instances.setMetadata
6	2019-12-30 23:02:44.656 UTC	gce_instance	student-02-b85de59b290e@qwiklabs.net	v1.compute.instances.setTags

This simple query is just one example of using BigQuery to generate custom log output. You can construct any number of SQL queries to analyze your audit logs as you see fit.

Congratulations!

This concluded this hands-on lab, Fundamentals of Cloud Logging. You learned what Cloud Logging is and how to use it.

Finish your Quest

This self-paced lab is part of the Qwiklabs Cloud Logging Quest. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge public and link to them in your online resume or social media account. Enroll in this Quest and get immediate completion credit if you've taken this lab. See other available Qwiklabs Quests.

Next steps / learn more

Questions about Cloud Monitoring? See Cloud Monitoring Documentation.
See what else you can do with BigQuery.
For more information on advanced filters and on the various fields that you can use within your filter criteria, see:

- Advanced logs filters
- LogEntry

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.
Manual Last Updated March 22, 2021
Lab Last Tested March 22, 2021