Autoscaling an Instance Group with Custom Cloud Monitoring Metrics

qwiklabs.com/focuses/611

GSP087

Overview

In this lab you will create a Compute Engine managed instance group that autoscales based on the value of a custom Cloud Monitoring metric.

Objectives

- Deploy an autoscaling Compute Engine instance group.
- Create a custom metric used to scale the instance group.
- Use the Cloud Console to visualize the custom metric and instance group size.

Application architecture

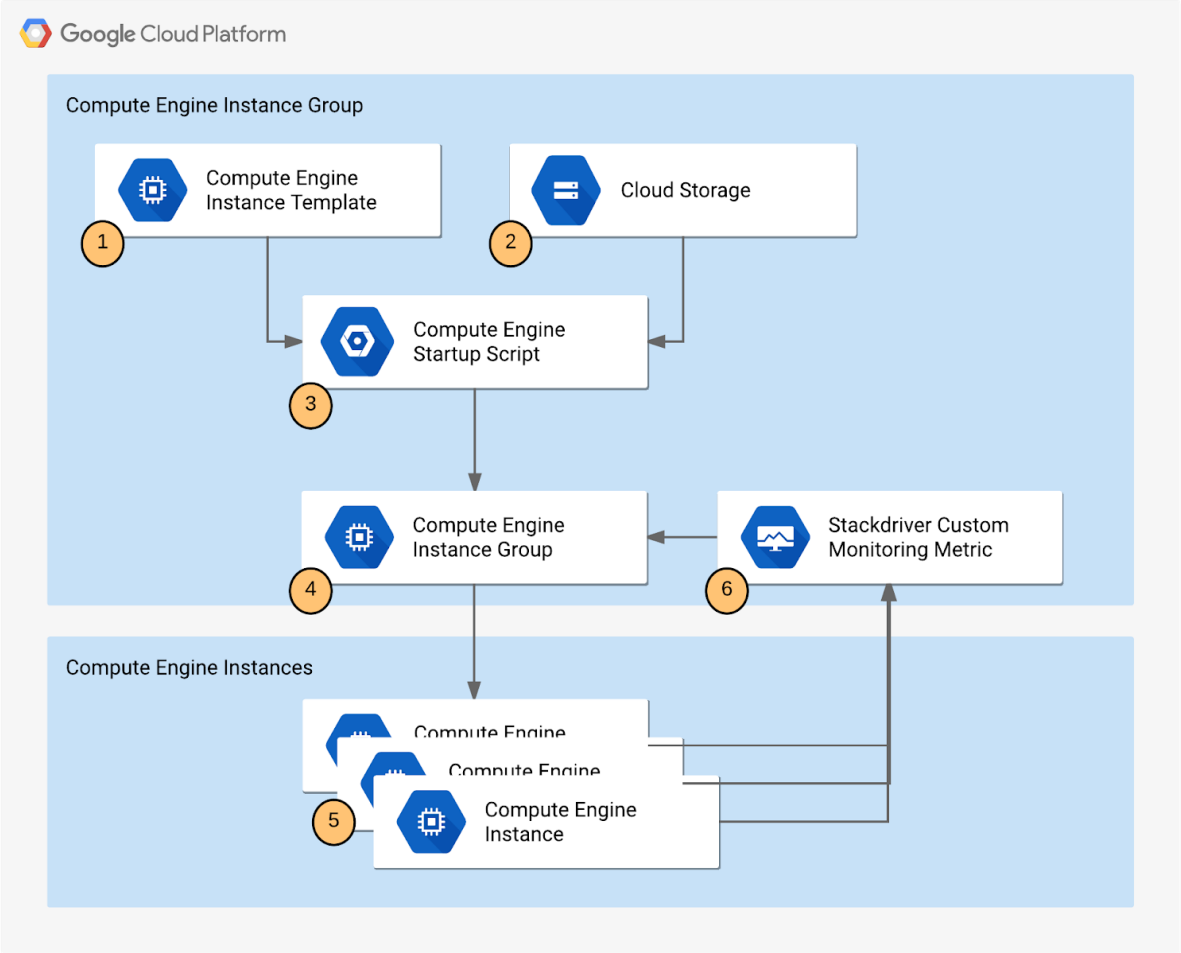
The autoscaling application uses a Node.js script installed on Compute Engine instances. The script reports a numeric value to a Cloud monitoring metric. You do not need to know Node.js or JavaScript for this lab. In response to the value of the metric, the application autoscales the Compute Engine instance group up or down as needed.

The Node.js script is used to seed a custom metric with values that the instance group can respond to. In a production environment, you would base autoscaling on a metric that is relevant to your use case.

The application includes the following components:

1. **Compute Engine instance template** - A template used to create each instance in the instance group.
2. **Cloud Storage** - A bucket used to host the startup script and other script files.
3. **Compute Engine startup script** - A startup script that installs the necessary code components on each instance. The startup script is installed and started automatically when an instance starts. When the startup script runs, it in turn installs and starts code on the instance that writes values to the Cloud monitoring custom metric.
4. **Compute Engine instance group** - An instance group that autoscales based on the Cloud monitoring metric values.
5. **Compute Engine instances** - A variable number of Compute Engine instances.

6. **Custom Cloud Monitoring metric** - A custom monitoring metric used as the input value for Compute Engine instance group autoscaling.



Setup and Requirements

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.

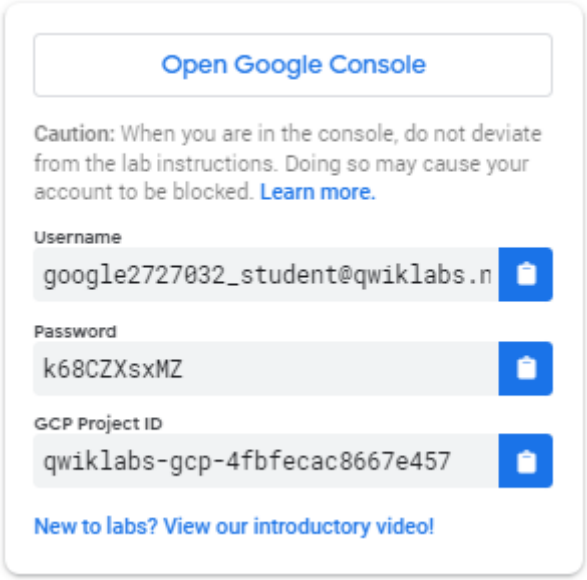
Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

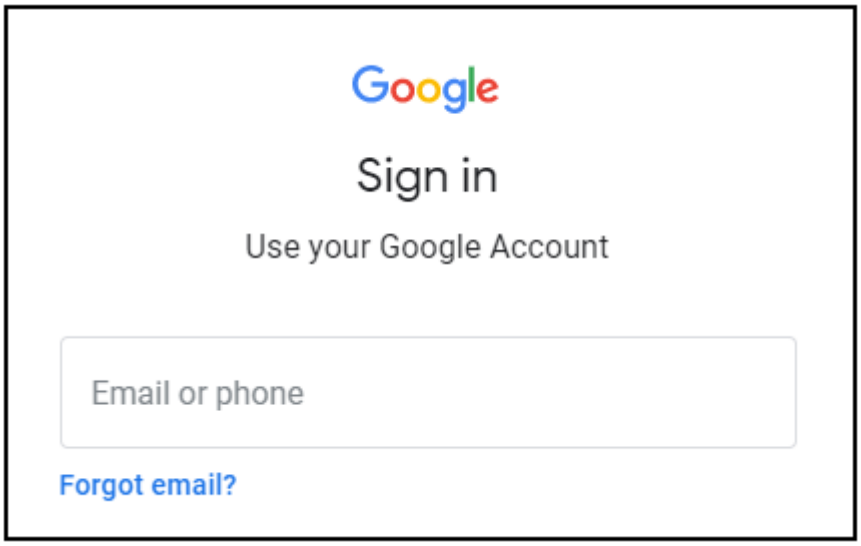
Note: If you are using a Pixelbook, open an Incognito window to run this lab.

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.



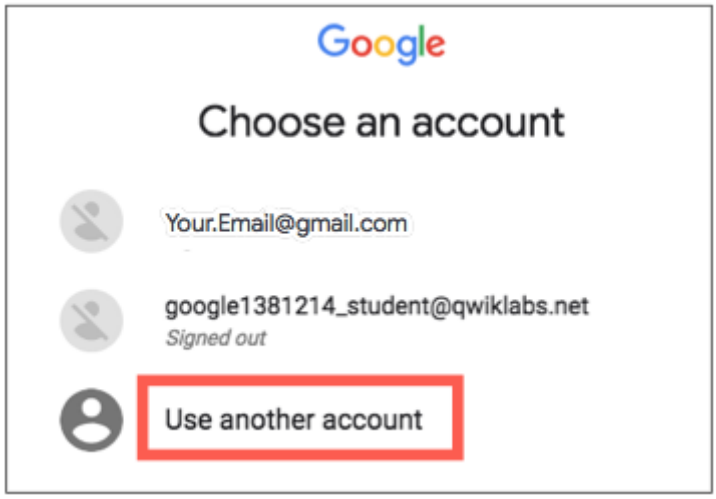


Tip: Open the tabs in separate windows, side-by-side.

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.

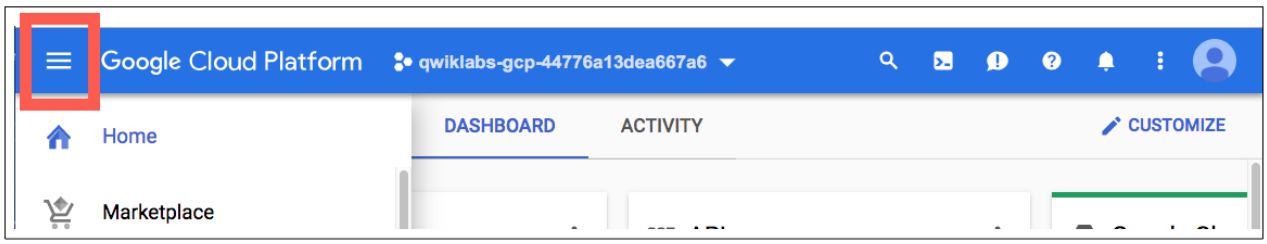
   Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).



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.

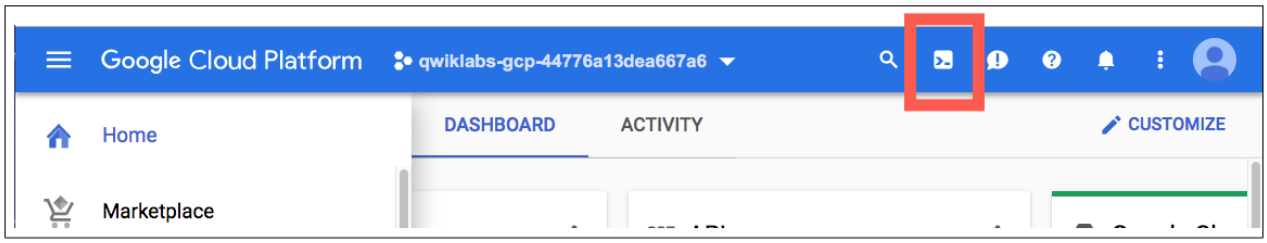
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



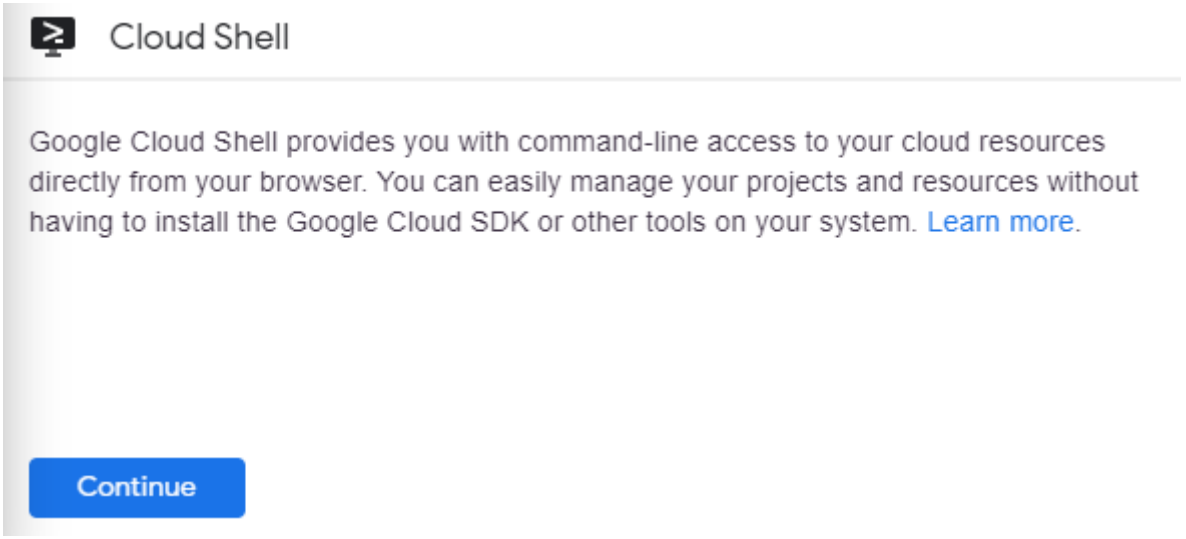
Activate Cloud Shell

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.

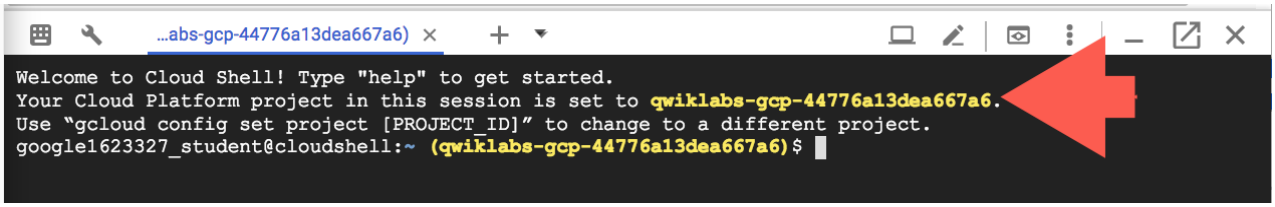
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



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.

Creating the application

Creating the autoscaling application requires downloading the necessary code components, creating a managed instance group, and configuring autoscaling for the managed instance group.

Uploading the script files to Cloud Storage

During autoscaling, the instance group will need to create new Compute Engine instances. When it does, it creates the instances based on an instance template. Each instance needs a startup script. Therefore, the template needs a way to reference the startup script. Compute Engine supports using Cloud Storage buckets as a source for your startup script. In this section, you will make a copy of the startup script and application files for a sample application used by this lab that pushes a pattern of data into a custom Cloud logging metric that you can then use to configure as the metric that controls the autoscaling behavior for an autoscaling group.

Note: There is a pre-existing instance template and group that has been created automatically by the lab that is already running. Autoscaling requires at least 30 minutes to demonstrate both scale-up and scale-down behavior, and you will examine this group later to see how scaling is controlled by the variations in the custom metric values generated by the custom metric scripts.

Create a bucket

In the Cloud Console, from the **Navigation menu** scroll down to **Cloud Storage**, then click **Create bucket**.

Give your bucket a unique name, but don't use a name you might want to use in another project. For details about how to name a bucket, see the bucket naming guidelines. This bucket will be referenced as `YOUR_BUCKET` throughout the lab.

Accept the default values then click **Create**.

When the bucket is created, the Bucket details window opens.

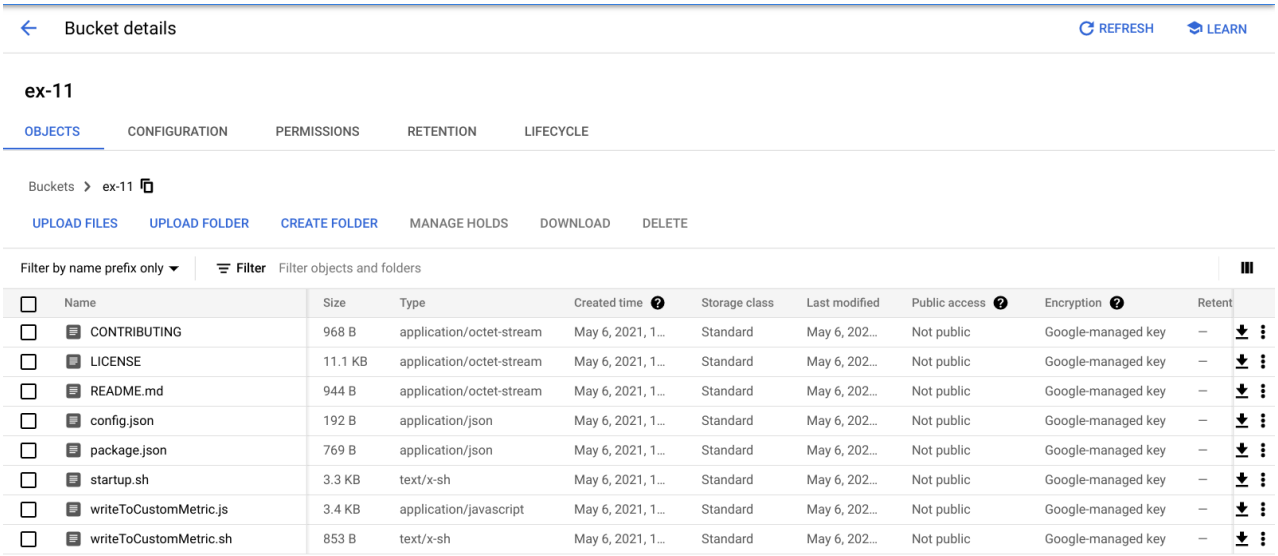
Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully created a Cloud Storage bucket, you will see an assessment score.

Create a Cloud Storage bucket
Next, run the following command in Cloud Shell to copy the startup script files from the lab default Cloud Storage bucket to your Cloud Storage bucket. Remember to replace `<YOUR BUCKET>` with the name of the bucket you just made.

```
gsutil cp -r gs://spls/gsp087/* gs://<YOUR BUCKET>
```

After you upload the scripts, click **Refresh** on the Bucket details page. Your bucket should list the added files.



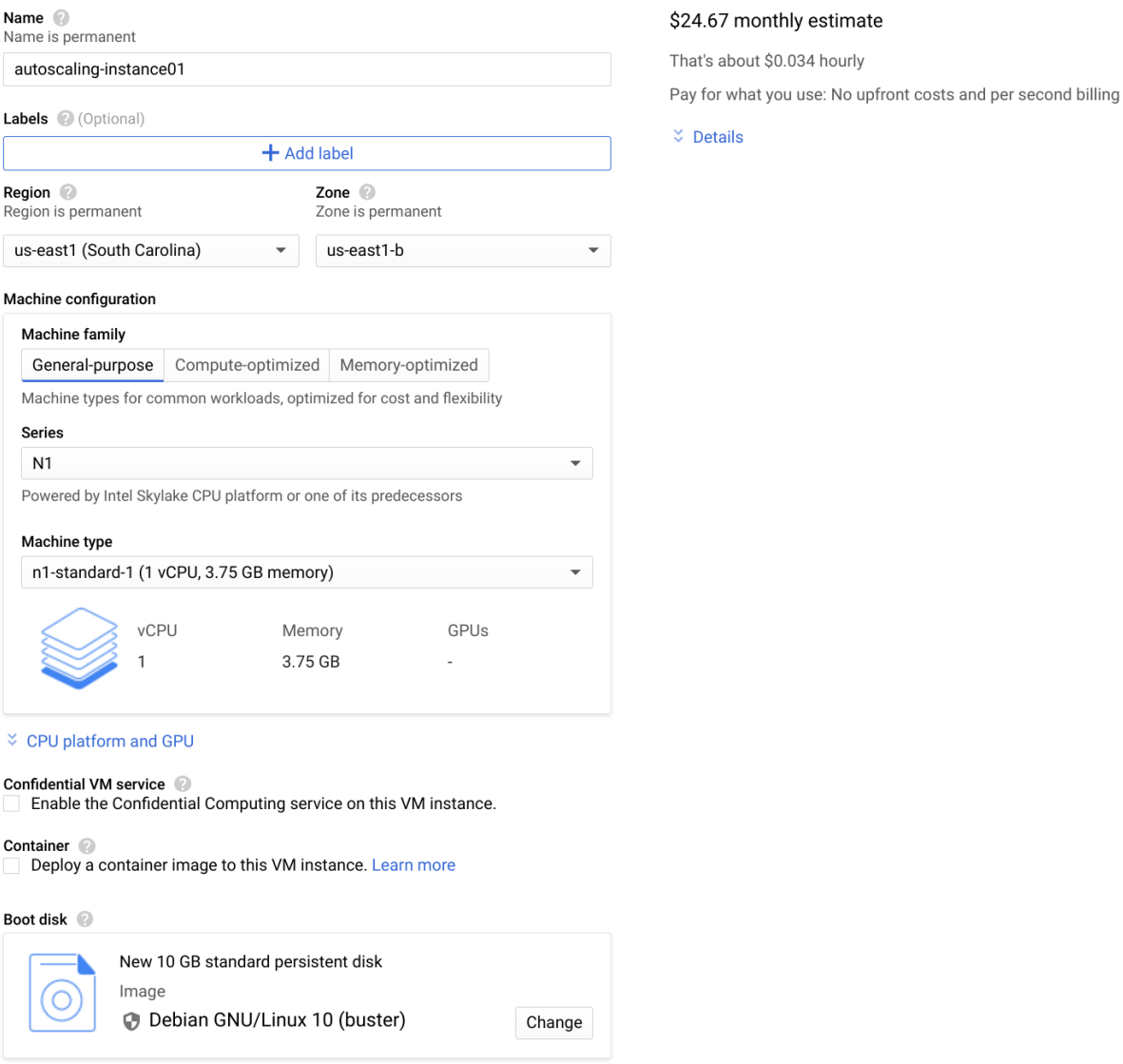
Understanding the code components

- `Startup.sh` - A shell script that installs the necessary components to each Compute Engine instance as the instance is added to the managed instance group.
- `writeToCustomMetric.js` - A Node.js snippet that creates a custom monitoring metric whose value triggers scaling. To emulate real-world metric values, this script varies the value over time. In a production deployment, you replace this script with custom code that reports the monitoring metric that you're interested in, such as a processing queue value.
- `Config.json` - A Node.js config file that specifies the values for the custom monitoring metric and used in `writeToCustomMetric.js`.
- `Package.json` - A Node.js package file that specifies standard installation and dependencies for `writeToCustomMetric.js`.
- `writeToCustomMetric.sh` - A shell script that continuously runs the `writeToCustomMetric.js` program on each Compute Engine instance.

Creating an instance template

Now create a template for the instances that are created in the instance group that will use autoscaling. As part of the template, you specify the location (in Cloud Storage) of the startup script that should run when the instance starts.

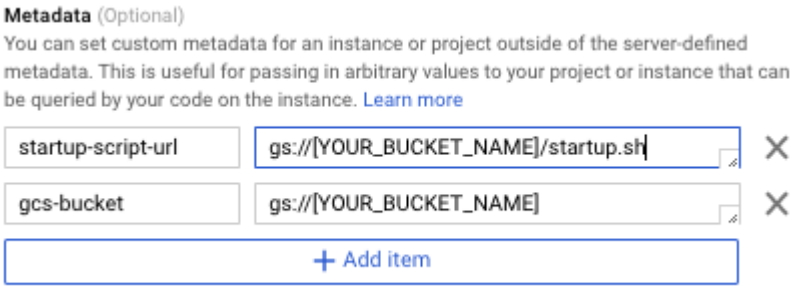
1. In the Cloud Console, go to **Navigation menu** > **Compute Engine** > **Instance templates**.
2. Click **Create Instance Template** at the top of the page.
3. Name the instance template `autoscaling-instance01`.



4. Scroll down, click **Management, security, disks, networking, sole tenancy** to expand the input options.
5. In the **Metadata** section of the **Management** tab, enter these metadata keys and values, clicking the **+ Add item** button to add each one. Remember to substitute your bucket name for the `[YOUR_BUCKET_NAME]` placeholder:

Key	Value

startup-script-url	`gs://[YOUR_BUCKET_NAME]/startup.sh`
gcs-bucket	`gs://[YOUR_BUCKET_NAME]`



6. Click **Create**.

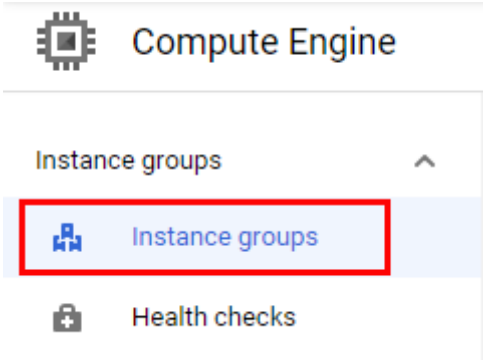
Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully created an instance template, you will see an assessment score.

Create an instance template

Creating the instance group

1. In the left pane, click **Instance groups**.
2. Click **Create instance group** at the top of the page.
3. **Name:** `autoscaling-instance-group-1`.
4. Under **Instance template**, select the instance template you just created.
5. Set **Autoscaling mode** to **Don't autoscale**.



You'll edit the autoscaling setting after the instance group has been created. Leave the other settings at their default values.

6. Click **Create**.

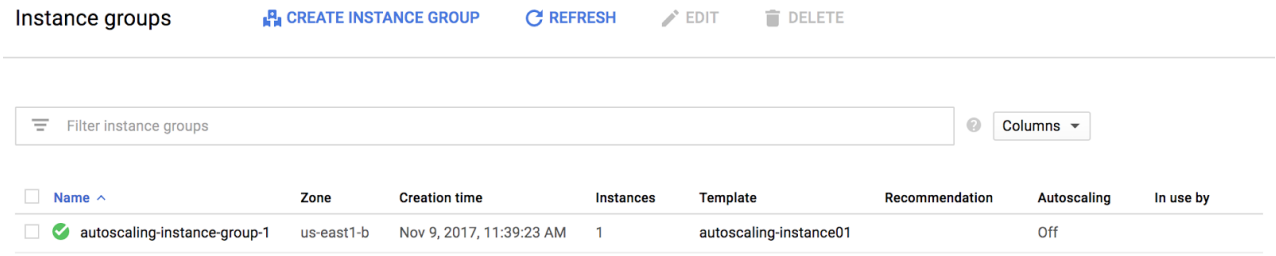
Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully created an instance group, you will see an assessment score.

Create an instance group

Verifying that the instance group has been created

Wait to see the green check mark next to the new instance group you just created. It might take the startup script several minutes to complete installation and begin reporting values. Click **Refresh** if it seems to be taking more than a few minutes.



Note: If you see a red icon next to the other instance group that was pre-created by the lab, you can ignore this warning. The instance group reports a warning for up to ten minutes as it is initializing. This is expected behavior.

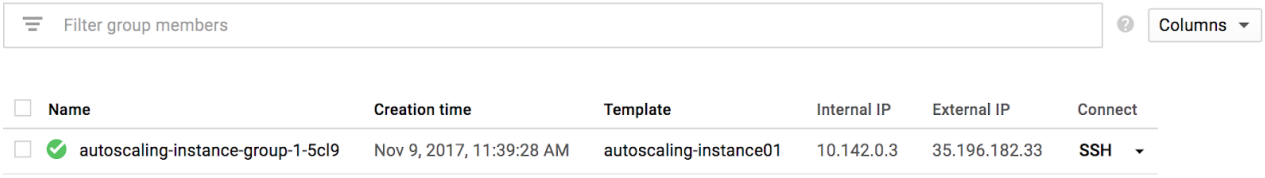
Verifying that the Node.js script is running

The custom metric `custom.googleapis.com/appdemo_queue_depth_01` isn't created until the first instance in the group is created and that instance begins reporting custom metric values.

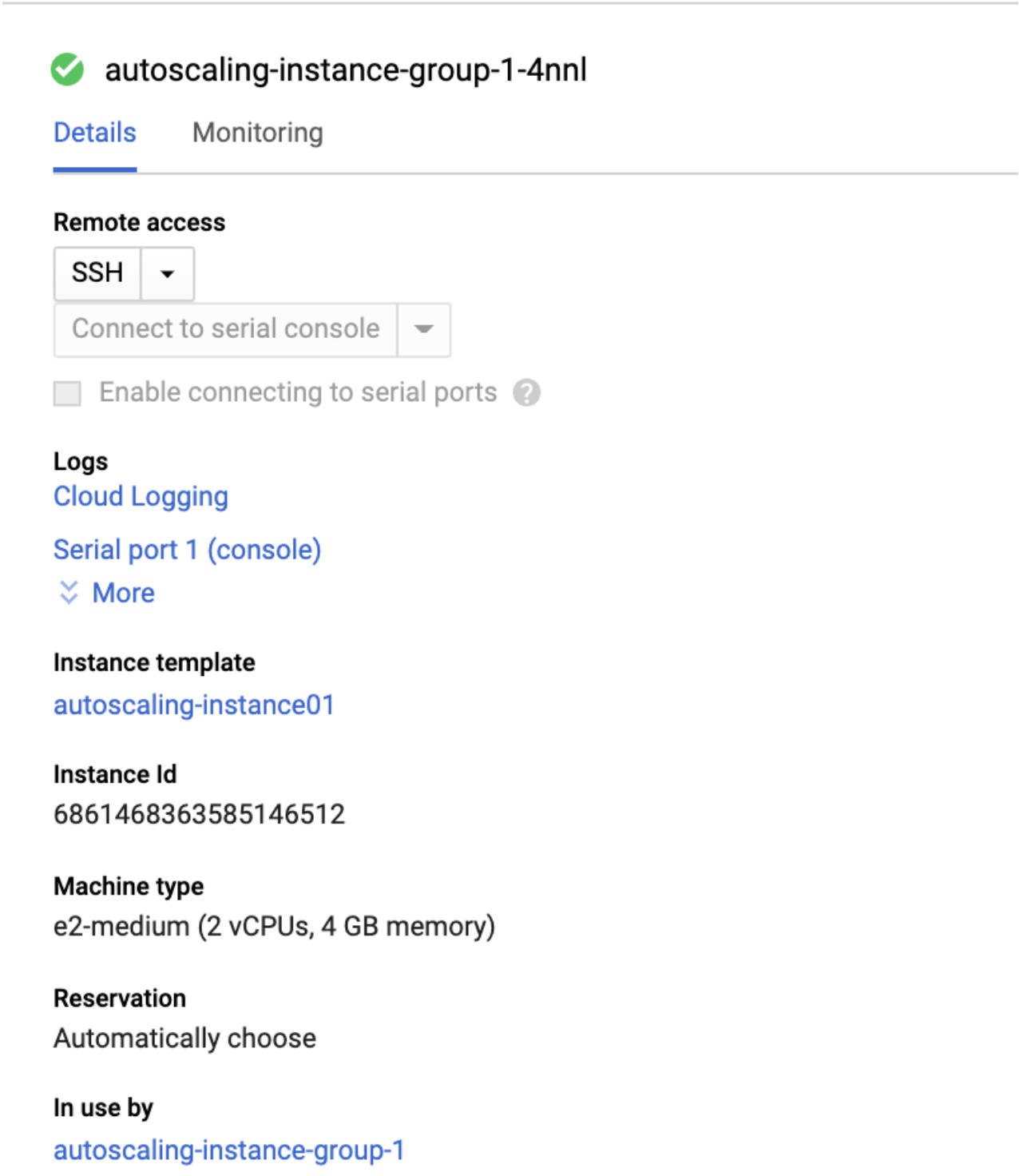
You can verify that the `writeToCustomMetric.js` script is running on the first instance in the instance group by checking whether the instance is logging custom metric values.

1. Still in the Compute Engine Instance groups window, click the name of the `autoscaling-instance-group-1` to display the instances that are running in the group.

2. Scroll down and click the instance name. Because autoscaling has not started additional instances, there is just a single instance running.



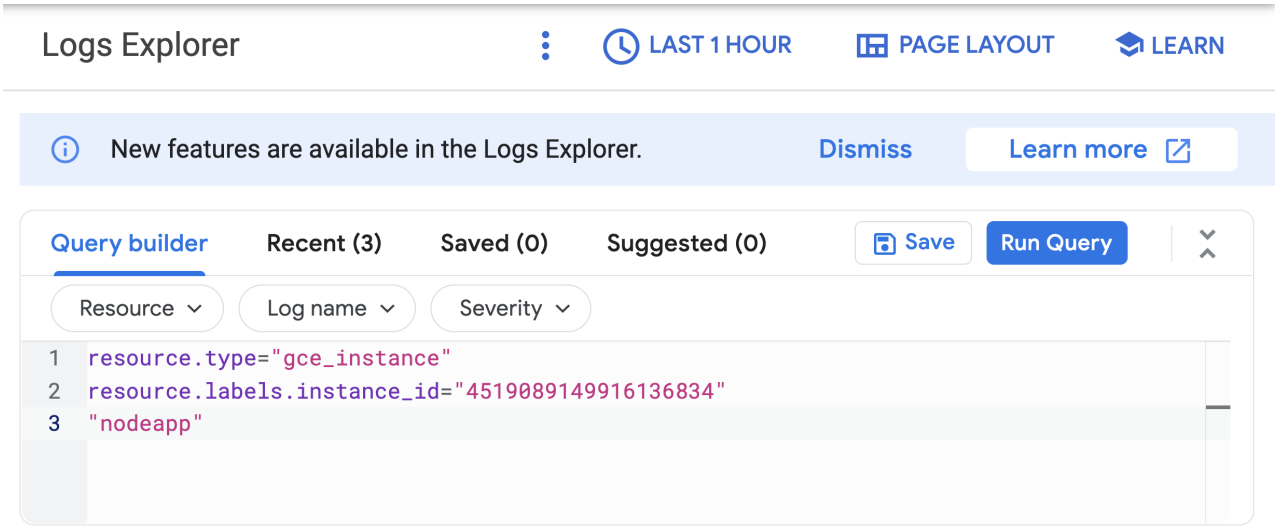
3. In the **Details** tab, click **Cloud Logging** link to view the logs for the VM instance.



4. Wait a minute or 2 to let some data accumulate. You will see `resource.type` and `resource.labels.instance_id` in the **Query preview** box.

5. Now click the drop-down arrow next to **Run Query** to open.

6. Add "nodeapp" as line 3, so the code looks similar to this:



7. Click **Run Query**.

If the `Node.js` script is being executed on the Compute Engine instance, a request is sent to the API, and log entries that say `Finished writing time series data` appear in the logs.

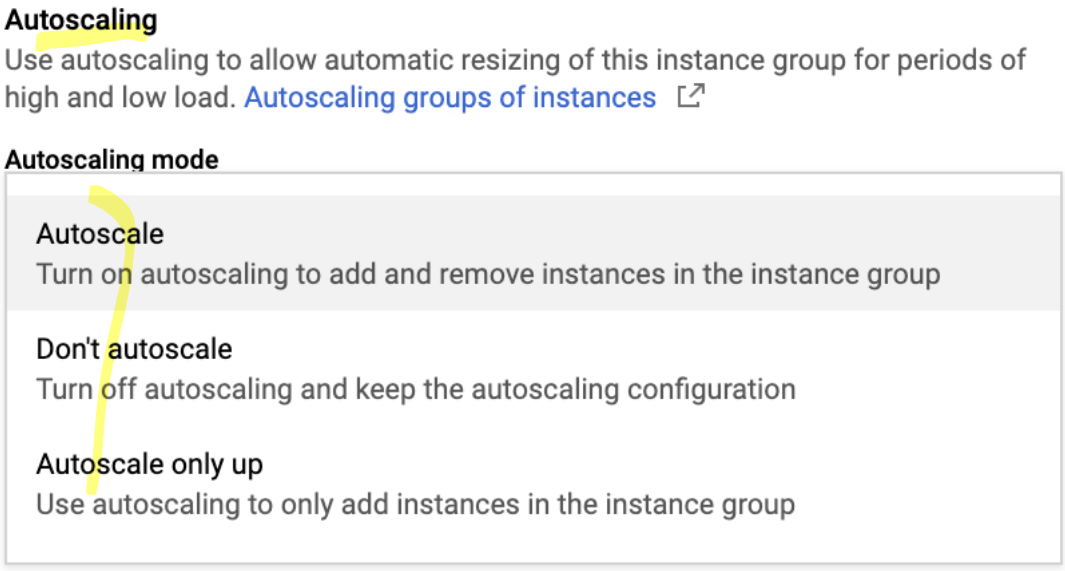
If you don't see this log entry, the Node.js script isn't reporting the custom metric values. Check that the metadata was entered correctly. If the metadata is incorrect, it might be easiest to restart the lab.

Configure autoscaling for the instance group

After you've verified that the custom metric is successfully reporting data from the first instance, the instance group can be configured to autoscale based on the value of the custom metric.

1. In the Cloud Console, go to **Compute Engine** > **Instance groups**.
2. Click the `autoscaling-instance-group-1` group.
3. Under Autoscaling click **Configure**.

4. Set **Autoscaling mode** to **Autoscale.**



5. Click **Autoscaling Configuration** click **Add new metric** to edit metric. Set the following fields, leave all others at the default value.

- **Metric Type**: `Stackdriver Monitoring metric`
- **Metric export scope**: `Time series per instance`
- **Metric identifier**: `custom.googleapis.com/appdemo_queue_depth_01`
- **Utilization target**: `150`

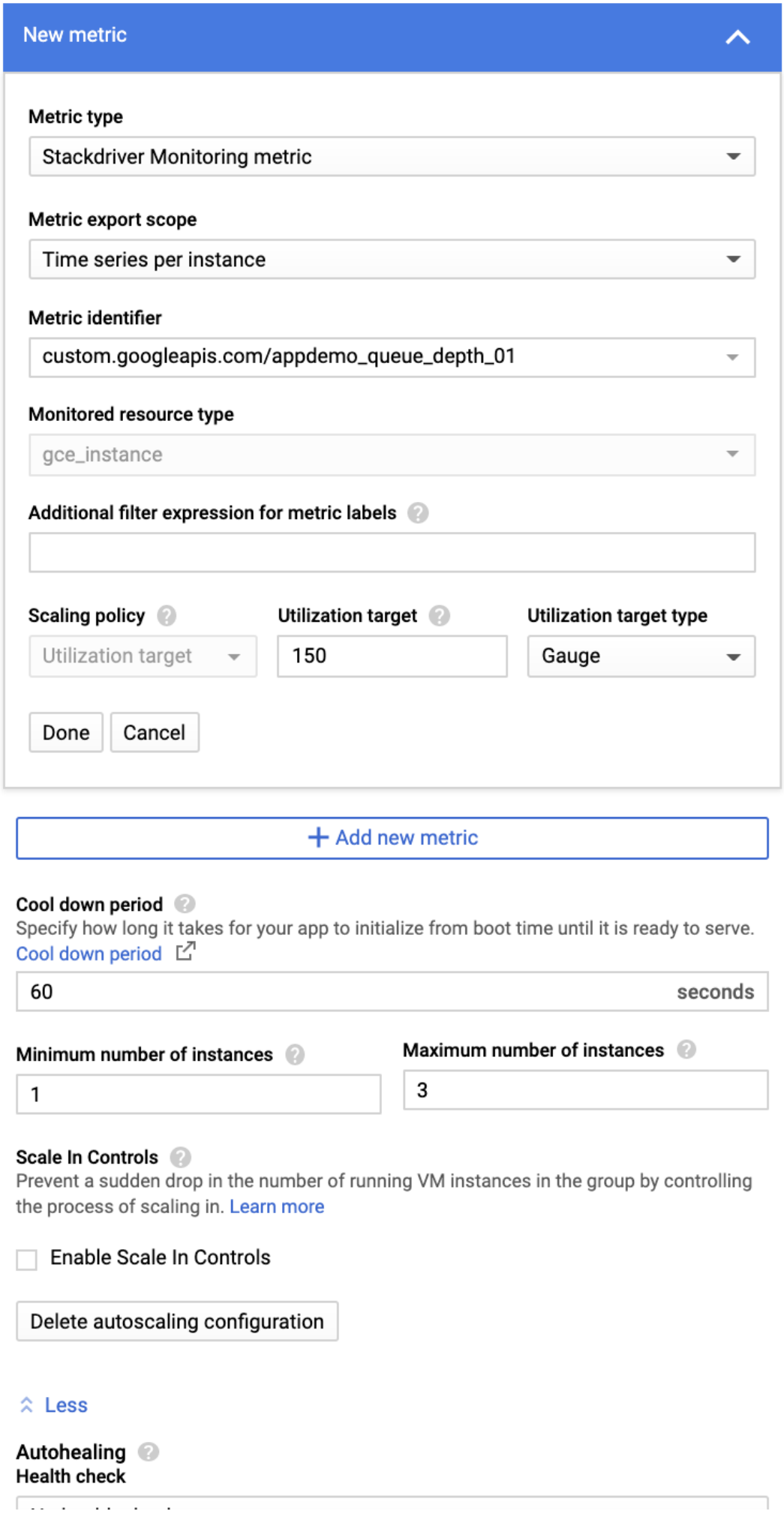
  When custom monitoring metric values are higher or lower than the **Target** value, the autoscaler scales the managed instance group, increasing or decreasing the number of instances. The target value can be any double value, but for this lab, the value 150 was chosen because it matches the values being reported by the custom monitoring metric.

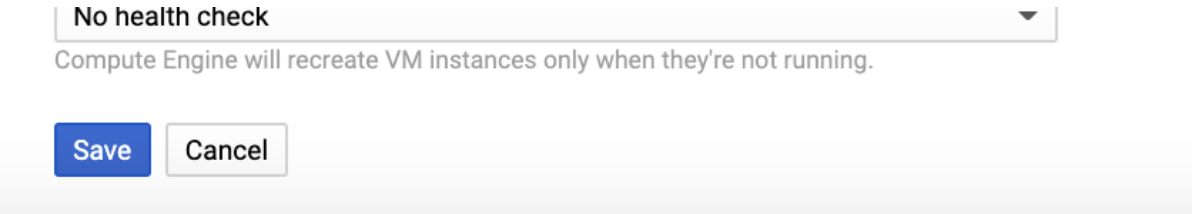
- **Utilization target type**: `Gauge`

  The **Gauge** setting specifies that the autoscaler should compute the average value of the data collected over the last few minutes and compare it to the target value. (By contrast, setting **Target mode** to **DELTA_PER_MINUTE** or **DELTA_PER_SECOND** autoscales based on the *observed* rate of change rather than an *average* value.)

- **Minimum number of instances**: `1`

- **Maximum number of instances**: `3`


New metric
Metric type
Stackdriver Monitoring metric
Metric export scope
Time series per instance
Metric identifier
custom.googleapis.com/appdemo_queue_depth_01
Monitored resource type
gce_instance
Additional filter expression for metric labels
Scaling policy
Utilization target
Utilization target
150
Utilization target type
Gauge
Done
Cancel
+ Add new metric
Cool down period
Specify how long it takes for your app to initialize from boot time until it is ready to serve.
Cool down period
60
seconds
Minimum number of instances
1
Maximum number of instances
3
Scale In Controls
Prevent a sudden drop in the number of running VM instances in the group by controlling the process of scaling in. Learn more
Enable Scale In Controls
Delete autoscaling configuration
Less
Autohealing
Health check



6. Click **Save**.

Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully configured autoscaling for the instance group, you will see an assessment score.

Configure autoscaling for the instance group

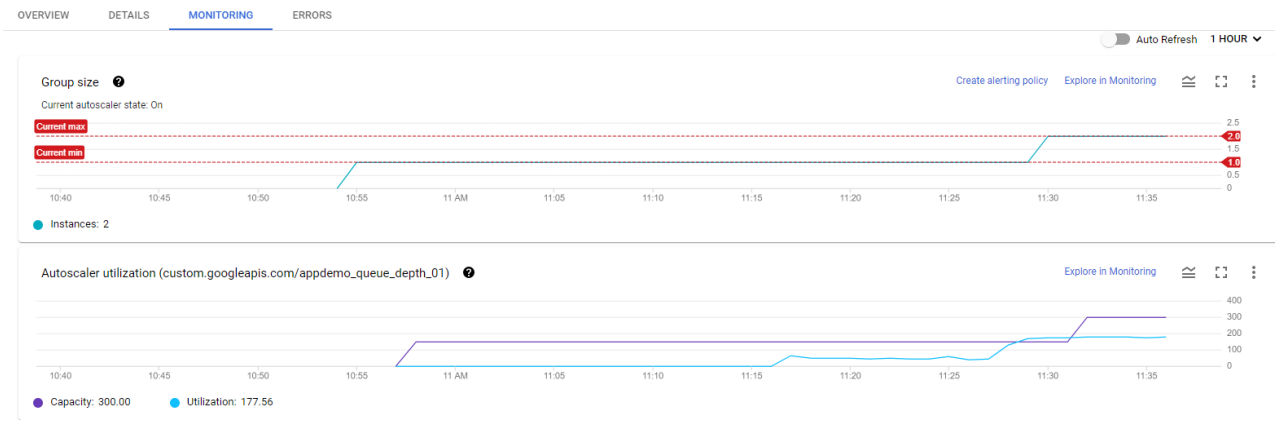
Watching the instance group perform autoscaling

The Node.js script varies the custom metric values it reports from each instance over time. As the value of the metric goes up, the instance group scales up by adding Compute Engine instances. If the value goes down, the instance group detects this and scales down by removing instances. As noted earlier, the script emulates a real-world metric whose value might similarly fluctuate up and down.

Next you will see how the instance group is scaling in response to the metric by clicking the **Monitoring** tab to view the **Autoscaled size** graph

1. In the left pane, click **Instance groups**.
2. Click the `builtin-igm` instance group in the list.
3. Click the **Monitoring** tab.
4. Enable **Auto Refresh**.

Since this group had a head start, you can see the autoscaling details about the instance group in the autoscaling graph. The autoscaler will take about five minutes to correctly recognize the custom metric and it can take up to ten minutes for the script to generate sufficient data to trigger the autoscaling behavior.



Hover your mouse over the graphs to see more details.

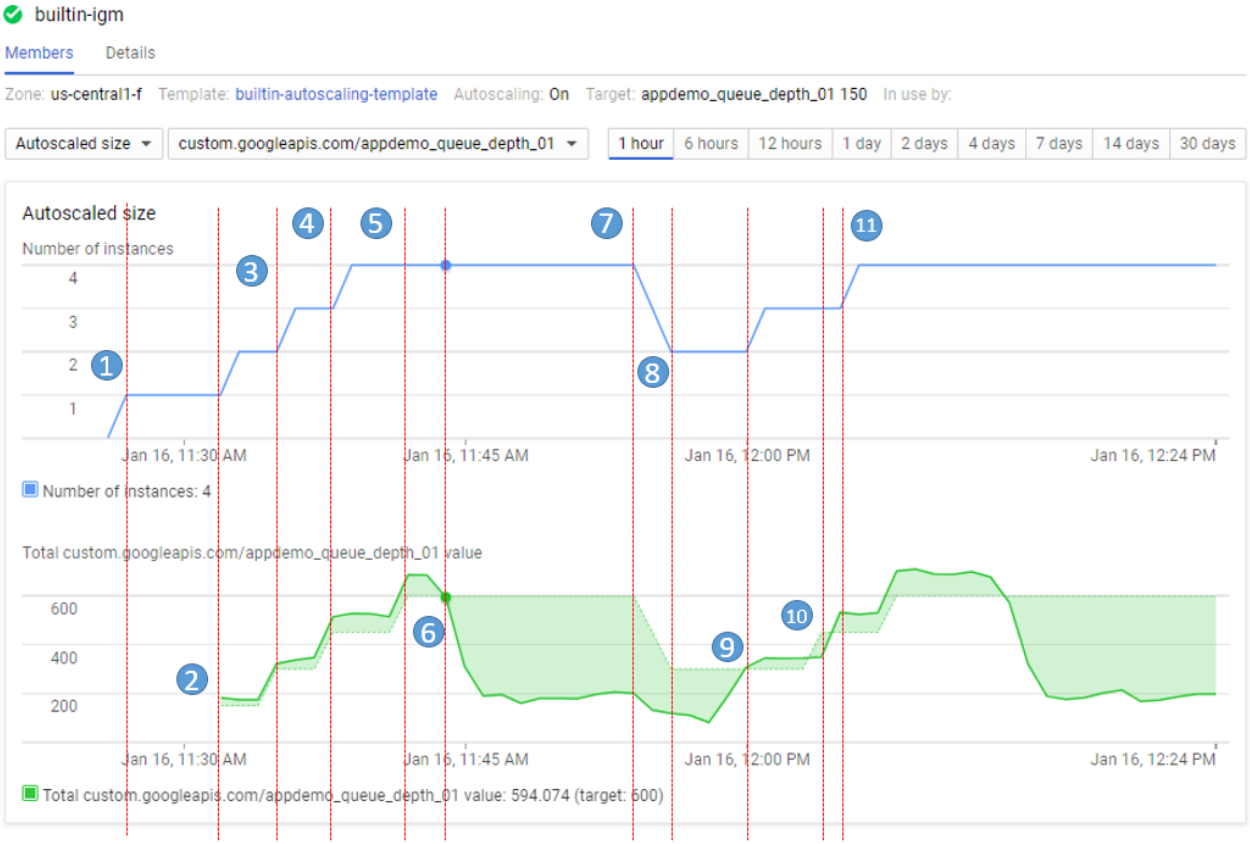
You can switch back to the instance group that you created to see how it's doing (there may not be enough time left in the lab to see any autoscaling on your instance group).

For the remainder of the time on your lab, you can watch the autoscaling graph move up and down as instances are added and removed.

Autoscaling example

Read through this autoscaling example to see how capacity and number of autoscaled instances can work in a larger environment.

The number of instances depicted in the top graph changes as a result of the varying aggregate level of the custom metric property values reported in the lower graph. There is a slight delay of up to five minutes after each instance starts up before that instance begins to report its custom metric values. While your autoscaling starts up, read through this graph to understand what will be happening:



The script starts by generating high values for approximately 15 minutes in order to trigger scale-up behavior.

- **11:27** Autoscaling Group starts with a single instance. The aggregate custom metric target is 150.
- **11:31** Initial metric data acquired. As the metric is greater than the target of 150 the autoscaling group starts a second instance.

- **11:33** Custom metric data from the second instance starts to be acquired. The aggregate target is now 300. As the metric value is above 300 the autoscaling group starts the third instance.
- **11:37** Custom metric data from the third instance starts to be acquired. The aggregate target is now 450. As the cumulative metric value is above 450 the autoscaling group starts the fourth instance.
- **11:42** Custom metric data from the fourth instance starts to be acquired. The aggregate target is now 600. The cumulative metric value is now above the new target level of 600 but since the autoscaling group size limit has been reached no additional scale-up actions occur.
- **11:44** The application script has moved into a low metric 15 minute period. Even though the cumulative metric value is below the target of 600 scale-down must wait for a ten minute built-in scale-down delay to pass before making any changes.
- **11:54** Custom metric data has now been below the aggregate target level of 600 for a four node cluster for over 10 minutes. Scale-down down now removes two instances in quick succession.
- **11:56** Custom metric data from the removed nodes is eliminated from the autoscaling calculation and the aggregate target is reduced to 300.
- **12:00** The application script has moved back into a high metric 15 minute period. The cumulative custom metric value has risen above the aggregate target level of 300 again so the autoscaling group starts a third instance.
- **12:03** Custom metric data from the new instance have been acquired but the cumulative values reported remain below the target of 450 so autoscaling makes no changes.
- **12:04** Cumulative custom metric values rise above the target of 450 so autoscaling starts the fourth instance.

Congratulations!

You have successfully created a managed instance group that autoscales based on the value of a custom metric.

Finish your Quest

This self-paced lab is part of the Qwiklabs Google Cloud's Operations Suite and Google Cloud Solutions I: Scaling Your Infrastructure Quests. 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 (or badges) public and link to them in your online resume or social media account. Enroll in a Quest and get immediate completion credit if you've taken this lab. See other available Qwiklabs Quests.

Take Your Next Lab

Continue your Quest with Monitor and Log with Google Cloud Operations Suite: Challenge Lab, or check out these suggestions:

Running Dedicated Game Servers in Google Kubernetes Engine

Next Steps / Learn More

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 June 9, 2021

Lab Last Tested May 6, 2021