Article: Using Kubernetes on the Google Cloud Platform

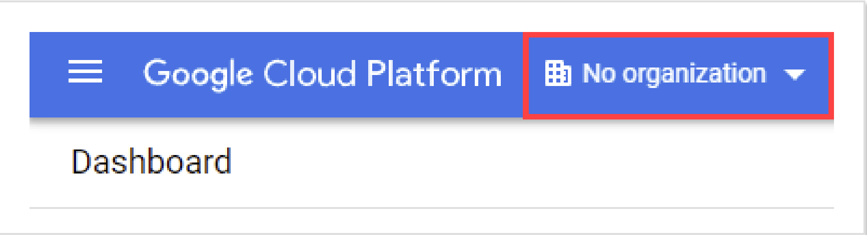
Using Kubernetes on the Google Cloud Platform

As part of your studies this trimester, you have been granted access to the Google Cloud Platform (GCP) and provided with a small budget. Information on how to login to Google Cloud has been/will be sent to your Deakin email address at the start of trimester or shortly after you enrolled. The instructions provided in this document will show you how to (1) create a three-node Kubernetes cluster in GCP, and (2) create and use a private container registry (for creating and using your own containers, such as for the labs).

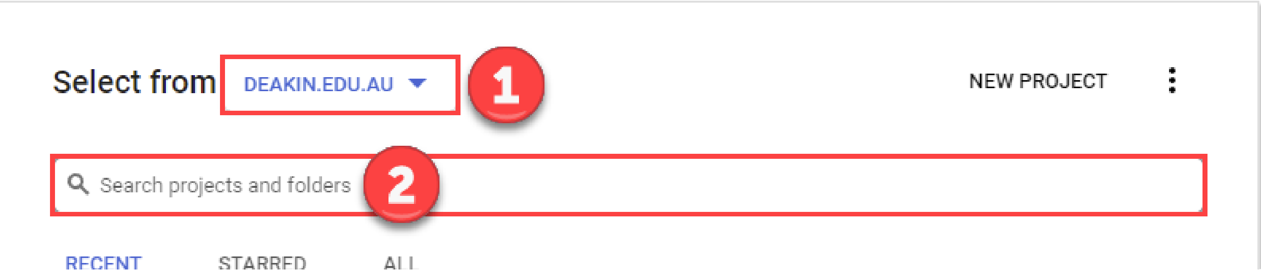
Step 1. Select Your Project in GCP

When you first access to GCP, it is critical to begin by selecting the project that has been created for you by Deakin. If you don't select the project correctly, Google will prompt you to begin a trial of GCP which, if used incorrectly, may eventually result in credit card charges. The Deakin project does not require you to provide any credit card information. The name of the project you will want to use for this unit will be included in the email sent to you. If you are enrolled in multiple units using GCP, look for the one that includes the correct unit code (SIT727).

To select your project, start by clicking on the organisation/project selector at the top left of the browser window, as shown:



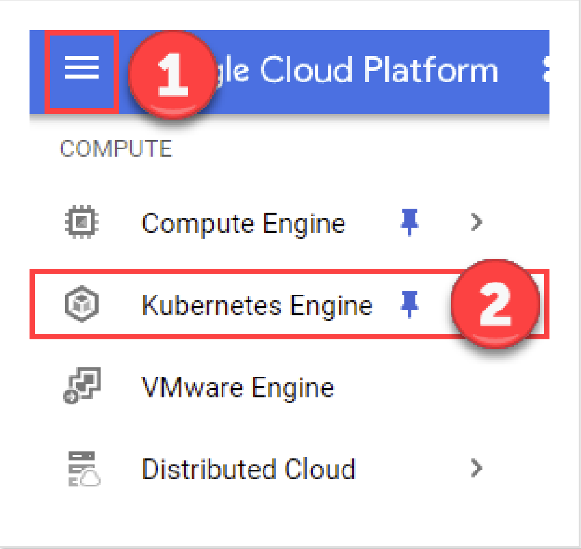
A new dialog will now pop-up allowing you to select your project. At the top of this dialog is the organisation selector (#1), make sure this is set to DEAKIN.EDU.AU. Then, using the Search box (#2) search for the project name sent to you in email and select that project, as shown:



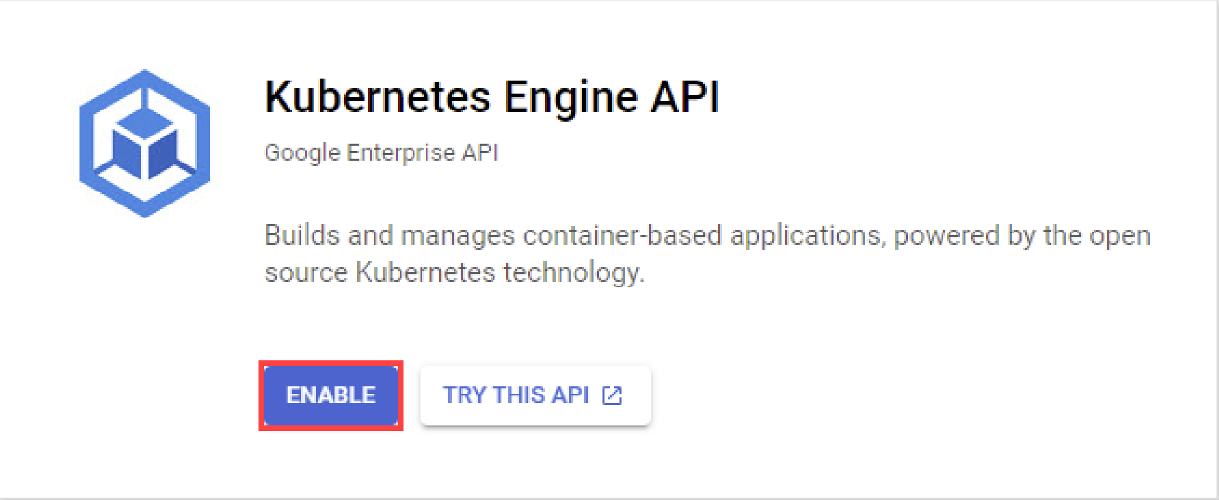
Finally, click on the **OPEN** button at the bottom of the project selector dialog.

Step 2. Enabling Kubernetes for your GCP Project

Before you can use any cloud services provided by GCP, you must first activate the service by enabling the relevant API. To do so, we begin by browsing to the Kubernetes service by clicking on the "hamburger icon" (icon consisting of three bars) at the top-left of your browser window (#1), then scrolling down the list of services and clicking on the Kubernetes Engine service (#2), as shown:



The browser will now display a page providing a summary of the Kubernetes Engine service. Click on the ENABLE button to activate this service, as shown:



Kubernetes Engine API

Google Enterprise API

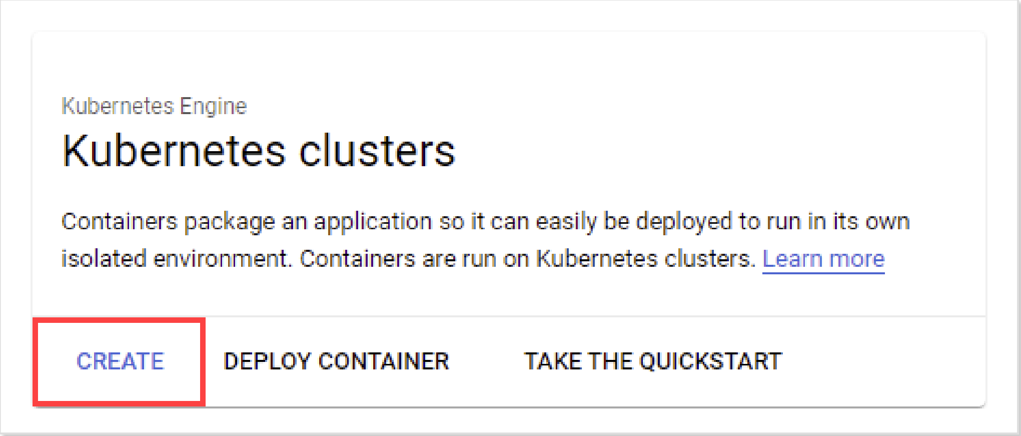
Builds and manages container-based applications, powered by the open source Kubernetes technology.

After a short delay, the page will refresh and you can now begin working with Kubernetes.

Step 3. Creating a Kubernetes Cluster

For the work we do in this unit, we will create a standard Kubernetes cluster. GCP also provides for managed Kubernetes services, however the managed services also have some features disabled (reflecting the features that are managed by Google).

Begin by making sure that you are on the page for managing Kubernetes clusters. If you just enabled the Kubernetes Engine service in Step 2 above, this is the default landing page. If not, make sure that you are currently viewing the Kubernetes Engine page, and then click on the Clusters link on the left hand side. You can now click on the CREATE link which will allow you to create a Kubernetes cluster, as shown:

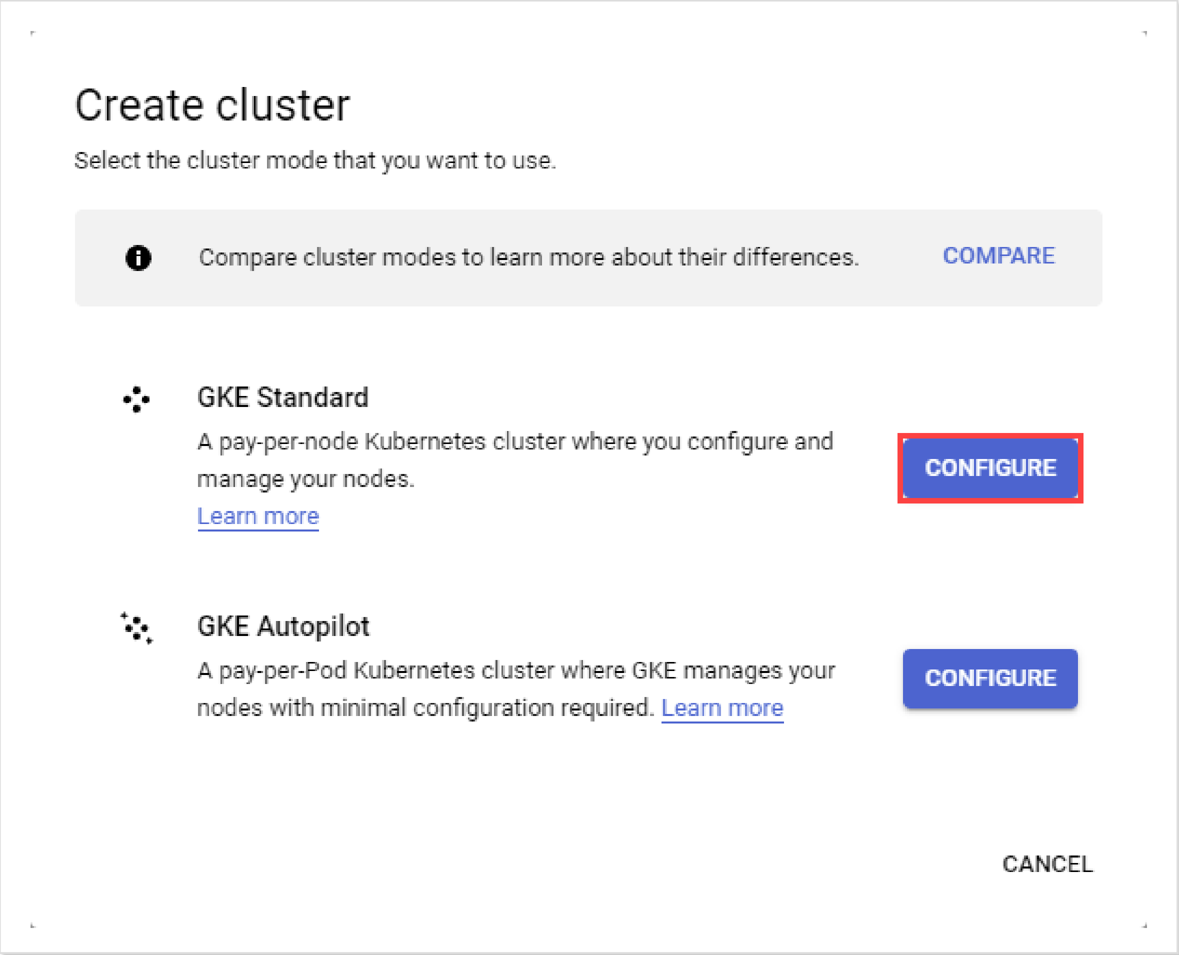


Kubernetes Engine

Kubernetes clusters

Containers package an application so it can easily be deployed to run in its own isolated environment. Containers are run on Kubernetes clusters. Learn more

You will now be prompted to select the type of cluster that you wish to create, for which we are using a GKE Standard cluster, so click on the CONFIGURE button matching this option, as shown:



Create cluster

Select the cluster mode that you want to use.

Compare cluster modes to learn more about their differences.

GKE Standard

A pay-per-node Kubernetes cluster where you configure and manage your nodes. Learn more

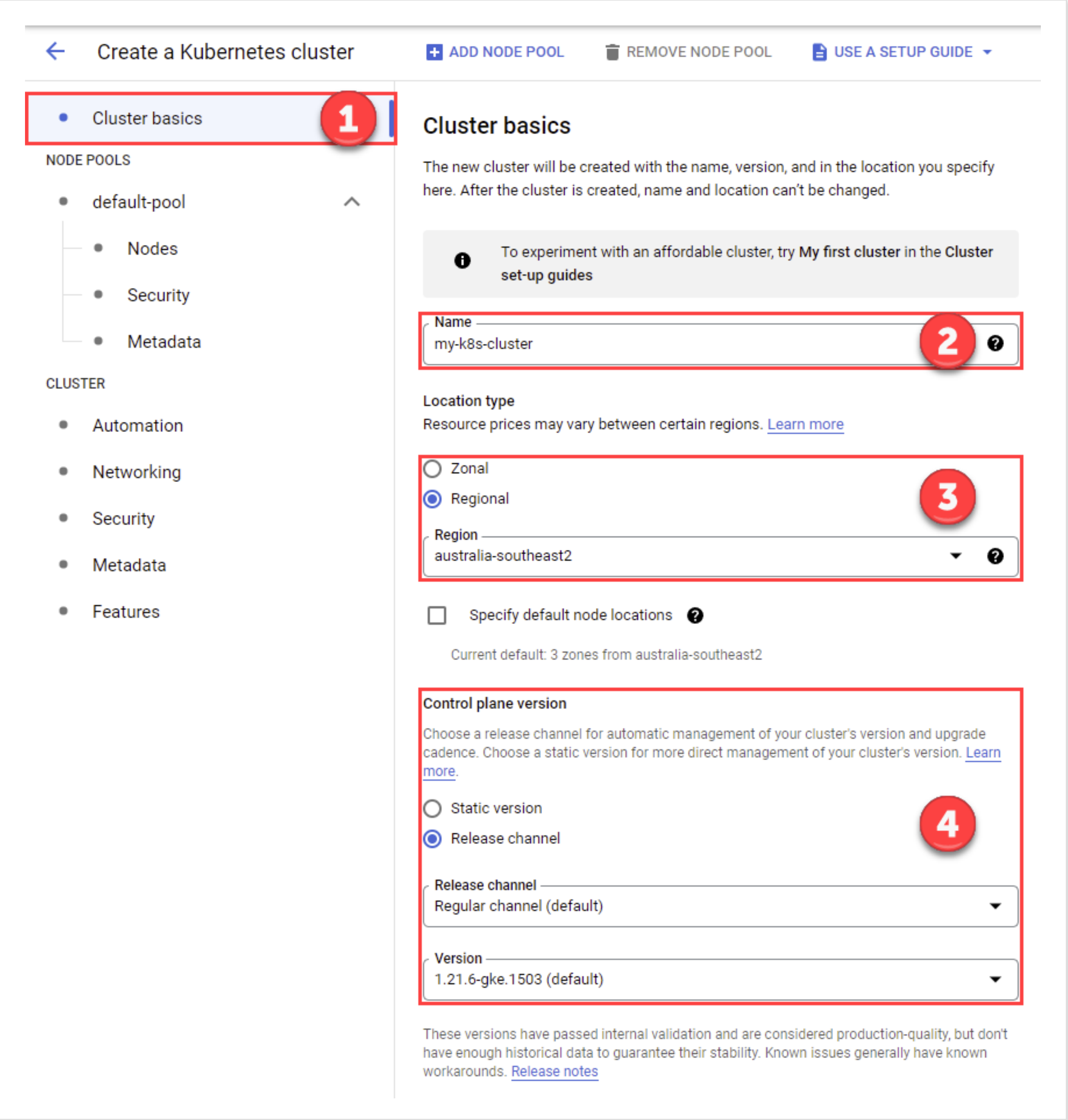
GKE Autopilot

A pay-per-Pod Kubernetes cluster where GKE manages your nodes with minimal configuration required. Learn more

A new dialog will now be displayed allowing us to configure the features of our new cluster. Make sure that you take the time to read through the various options presented on these pages and consider how it relates to what you learn in the unit, as you follow along with these instructions. For our purposes, we will be creating a very small and low-powered cluster.

The first page that needs to be configured is the "Cluster basics" page (#1). On this page you need to give your cluster a name (#2), for which you can use any name you wish (it's scoped to your project), and then choose an appropriate location for your cluster. In the example shown here, the location type has been changed to Regional and selected "australia-southeast2", which represents the GCP datacentres located in Melbourne (pay close attention to how you configure this here). You can also choose which version of Kubernetes should be deployed (#3), for which the defaults should usually be adequate.

Complete the Cluster basics page as shown:



Cluster basics

The new cluster will be created with the name, version, and in the location you specify here. After the cluster is created, name and location can't be changed.

To experiment with an affordable cluster, try **My first cluster** in the **Cluster set-up guides**

Name: my-k8s-cluster

Location type

Resource prices may vary between certain regions. Learn more

○ Zonal
● Regional

Region: australia-southeast2

□ Specify default node locations

Current default: 3 zones from australia-southeast2

Control plane version

Choose a release channel for automatic management of your cluster's version and upgrade cadence. Choose a static version for more direct management of your cluster's version. Learn more.

○ Static version
● Release channel

Release channel: Regular channel (default)

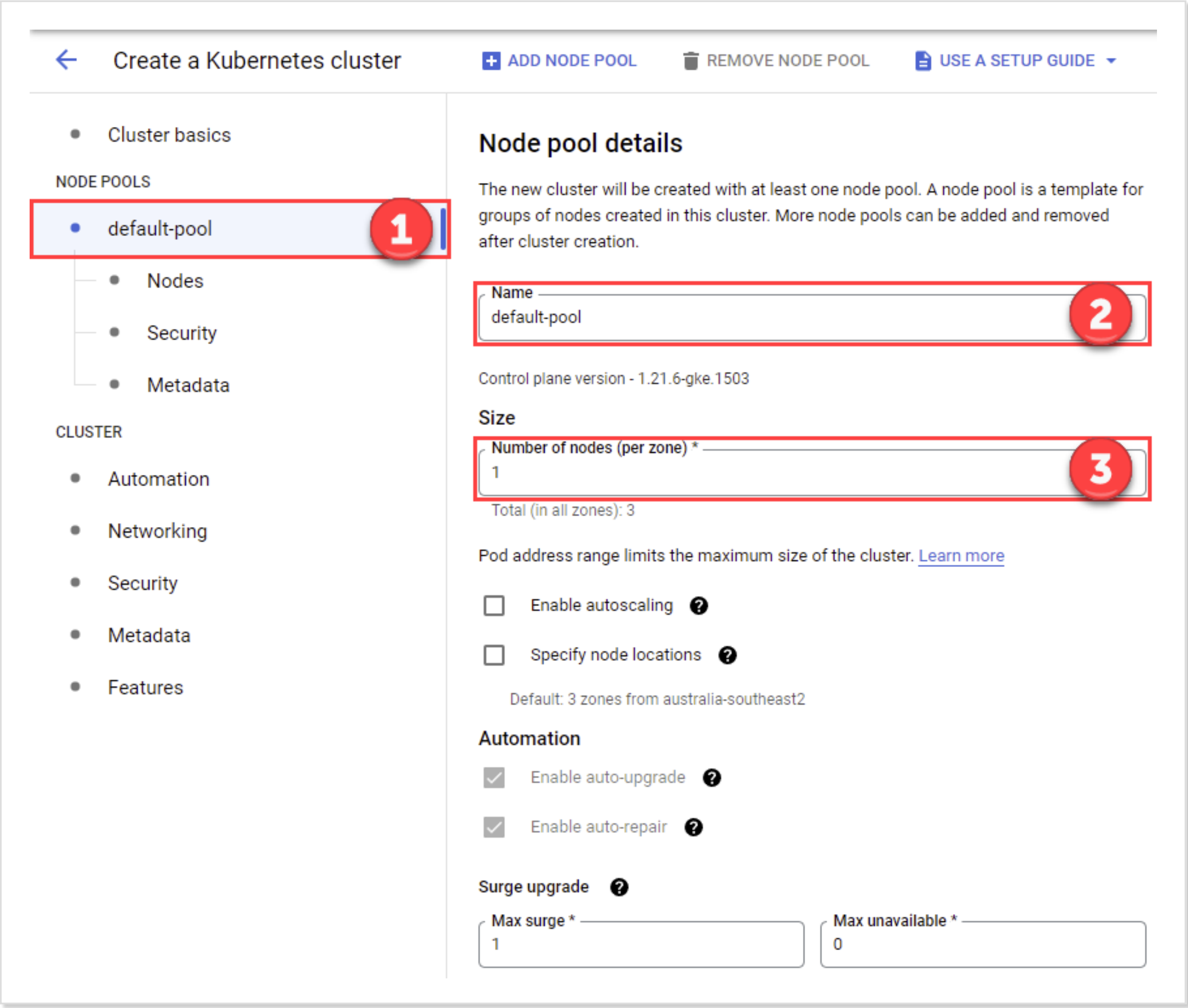
Version: 1.21.6-gke.1503 (default)

These versions have passed internal validation and are considered production-quality, but don't have enough historical data to guarantee their stability. Known issues generally have known workarounds. Release notes

Now switch to the 'default-pool' page under the NODE POOLS section (#1) to configure up how many nodes we require. Here we can give our node pool a name (#2), which again can be anything you wish as it is scoped to your own project, and then indicate how many nodes that we want in our cluster (#3).

When configuring the number of nodes you require, you need to be aware of the Location Type you selected on the previous screen. If you selected a Regional type as shown in the screenshot above, then on the default-pool screen we indicate that we want only 1 node per zone (GCP will deploy one node to each of three zones, i.e., 1x3=3 nodes). If you didn't follow the screen above and selected the Zonal type, then you instead need to indicate 3 nodes for the number of nodes (all three nodes will be deployed to the same zone/datacentre).

Complete the default-pool page as shown:



Node pool details

The new cluster will be created with at least one node pool. A node pool is a template for groups of nodes created in this cluster. More node pools can be added and removed after cluster creation.

Name: default-pool

Control plane version - 1.21.6-gke.1503

Size

Number of nodes (per zone) *: 1

Total (in all zones): 3

Pod address range limits the maximum size of the cluster. Learn more

□ Enable autoscaling

□ Specify node locations

Default: 3 zones from australia-southeast2

Automation

☑ Enable auto-upgrade

☑ Enable auto-repair

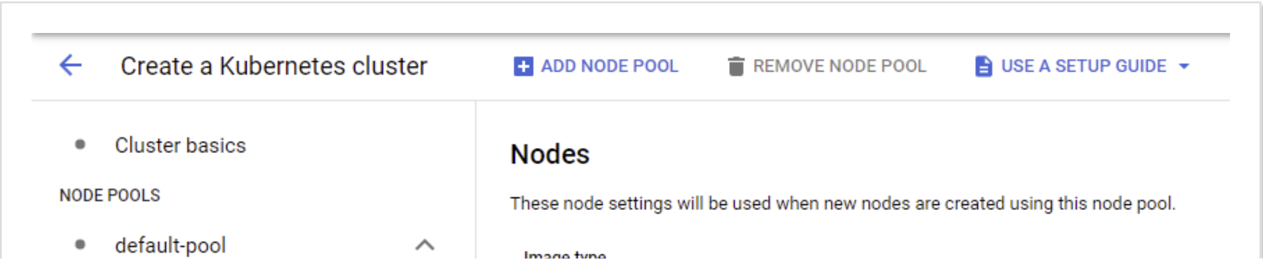
Surge upgrade

Max surge *: 1

Max unavailable *: 0

The last page that we need to modify is the Nodes page (#1), listed inside the default-pool configuration. This page allows us to specify what type of VM should be used for each node, for which we want ensure we use the smallest node that would work for us. For this purpose, we select the GENERAL PURPOSE (#2) category of VMs, the N1 series (#3), and the g1-small (#4) VM type. Finally, we don't need all the disk space shown here (#5), so reduce that down to 32GB to a more reasonable size.

Complete the Nodes page as shown:



Nodes

These node settings will be used when new nodes are created using this node pool.

Image type

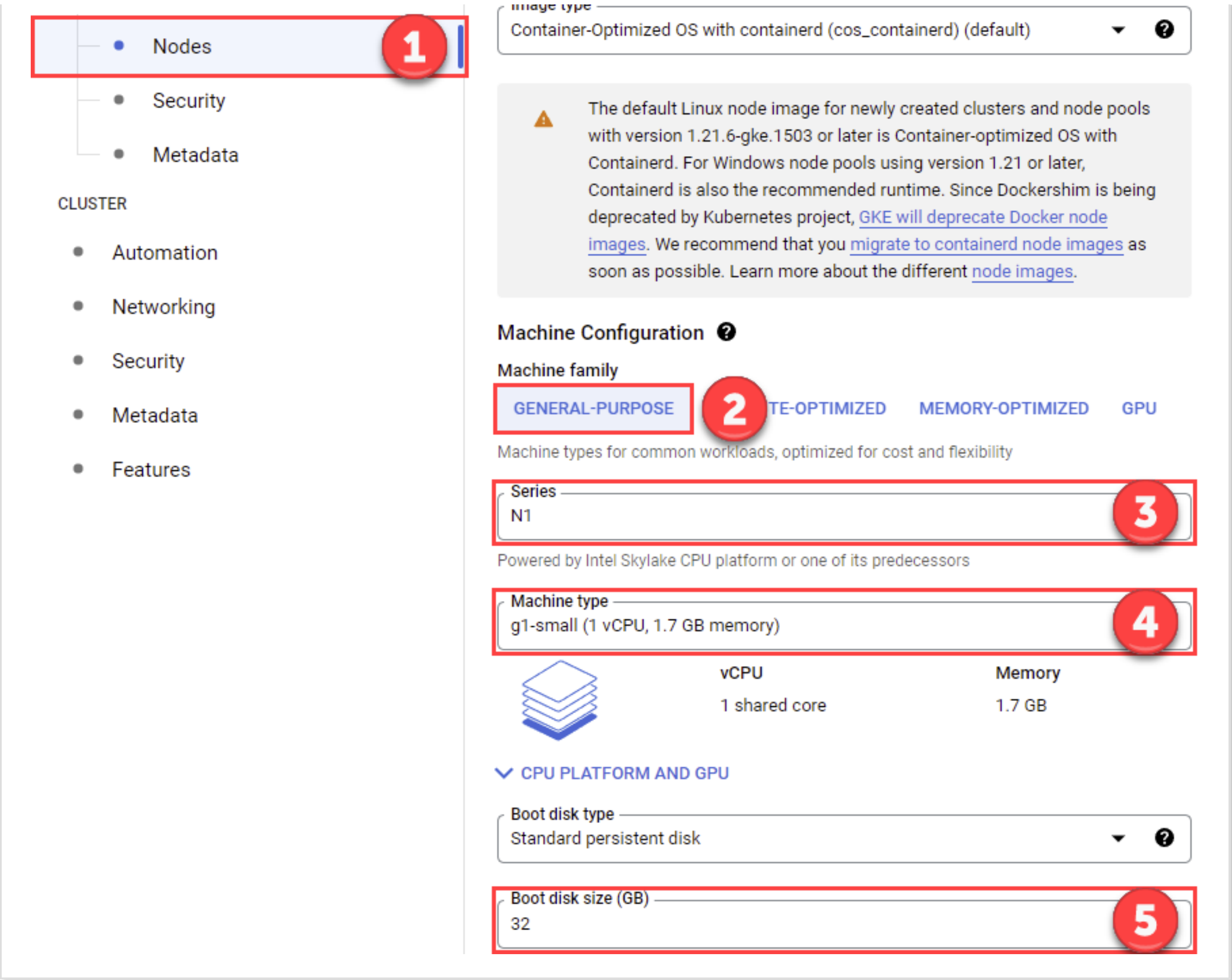


Image type
Container-Optimized OS with containerd (cos_containerd) (default)

The default Linux node image for newly created clusters and node pools with version 1.21.6-gke.1503 or later is Container-optimized OS with Containerd. For Windows node pools using version 1.21 or later, Containerd is also the recommended runtime. Since Dockershim is being deprecated by Kubernetes project, GKE will deprecate Docker node images. We recommend that you migrate to containerd node images as soon as possible. Learn more about the different node images.

Machine Configuration

Machine family
GENERAL-PURPOSE | TE-OPTIMIZED | MEMORY-OPTIMIZED | GPU

Machine types for common workloads, optimized for cost and flexibility

Series
N1

Powered by Intel Skylake CPU platform or one of its predecessors

Machine type
g1-small (1 vCPU, 1.7 GB memory)

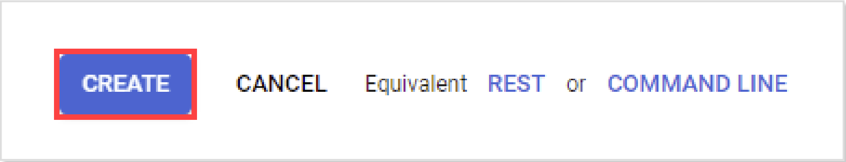
vCPU	Memory
1 shared core	1.7 GB

CPU PLATFORM AND GPU

Boot disk type
Standard persistent disk

Boot disk size (GB)
32

Finally, click on the CREATE button at the bottom of the page:

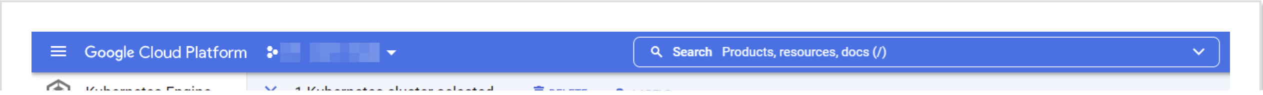


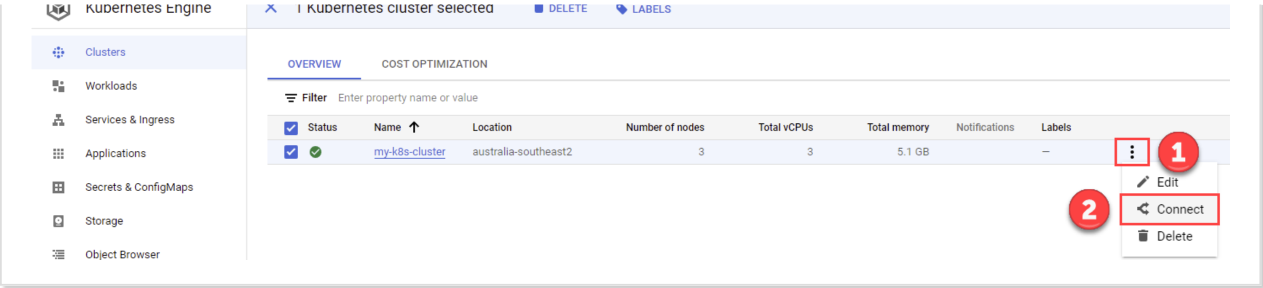
GCP will now go ahead and create the new cluster. This will take a short while, so feel free to take break for 10 minutes or so.

Step 4. Using your Kubernetes Cluster

Having completed the first three steps, our Kubernetes cluster is now ready to use, however we need to gain access to the command prompt to use it. The simplest way to achieve this is to use the built-in "cloud shell", which provides a BASH prompt via your web browser with all the required software already installed, we just need to configure its access to our cluster. GCP manages this process for us as well. After completing Step 3, you should now see a listing of all the Kubernetes clusters you have configured (should be just one). Check that the Status column shows your new cluster has been completed (a green tick indicates completion).

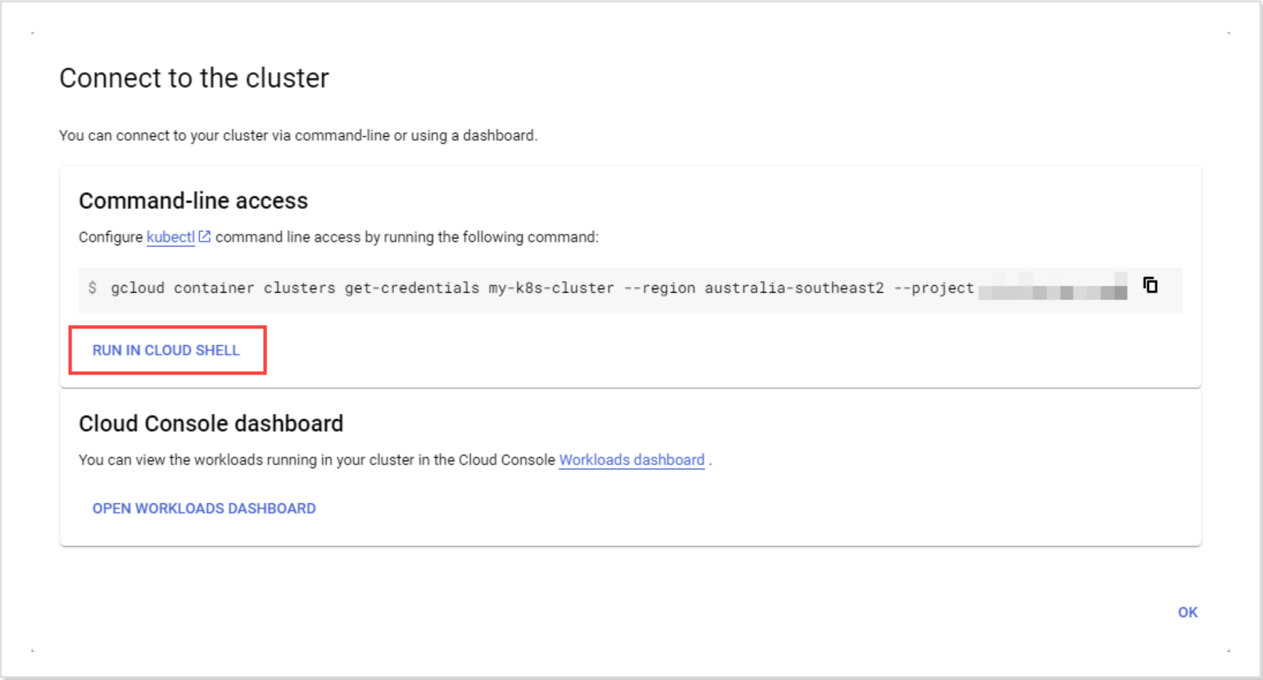
To connect to your Kubernetes cluster, click on the vertical elipsis (...) at the right of your cluster entry (#1) and select the the Connect option in the menu that is displayed (#2), as shown:





☑	Status	Name ↑	Location	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
☑	✓	my-k8s-cluster	australia-southeast2	3	3	5.1 GB		–

A new dialog will now be displayed advising you how to configure the kubectl program for your cluster. Underneath this instruction is a button to 'RUN IN CLOUD SHELL', i.e., to send this command to the cloud shell to configure its connectivity to your Kubernetes cluster. Click on the 'RUN IN CLOUD SHELL' button to proceed, as shown:



Connect to the cluster

You can connect to your cluster via command-line or using a dashboard.

Command-line access

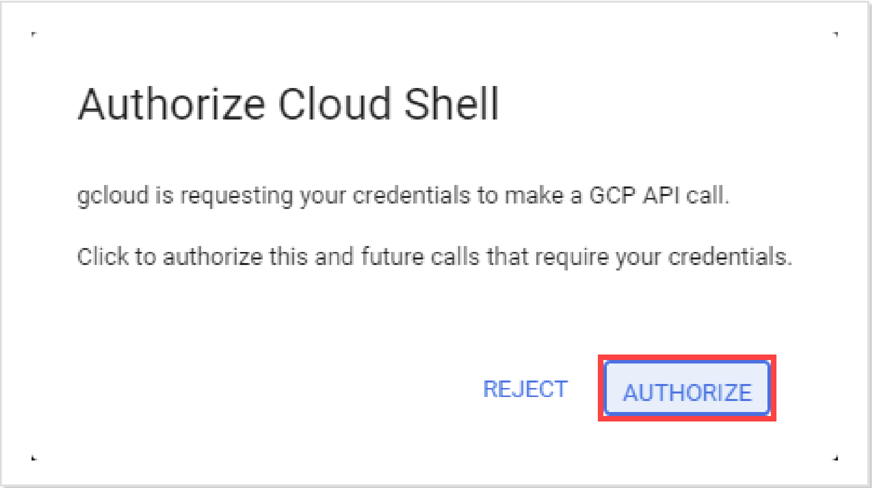
Configure kubectl command line access by running the following command:

```
$ gcloud container clusters get-credentials my-k8s-cluster --region australia-southeast2 --project [redacted]
```

Cloud Console dashboard

You can view the workloads running in your cluster in the Cloud Console Workloads dashboard .

GCP will now prepare the cloud shell to work for your project which will take a minute or two, and then it will automatically paste this command into the shell. Once this is displayed, just press the ENTER key without modifying this command. GCP will now prompt you to authorize the cloud shell to access the Kubernetes cluster. Click on the AUTHORIZE button to do, as shown:



Authorize Cloud Shell

gcloud is requesting your credentials to make a GCP API call.

Click to authorize this and future calls that require your credentials.

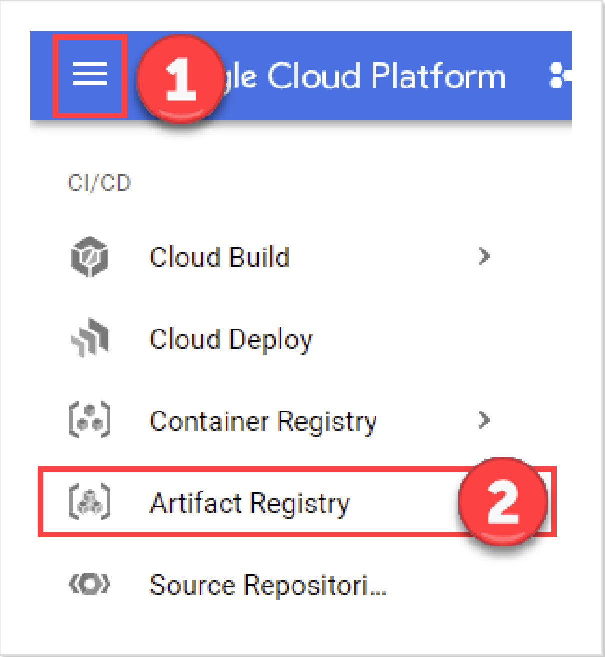
At this point you can now use the kubectl command from within the cloud shell and can use this shell in the same way we show throughout the unit.

Note: If you are too slow clicking the AUTHORIZE button in this step, you may be presented with an error in the cloud shell indicating that 'You do not currently have an active account selected'. If you receive this error, press the up-arrow key on your keyboard and then press ENTER (to repeat the command to authorise the cloud shell) and it will usually go through.

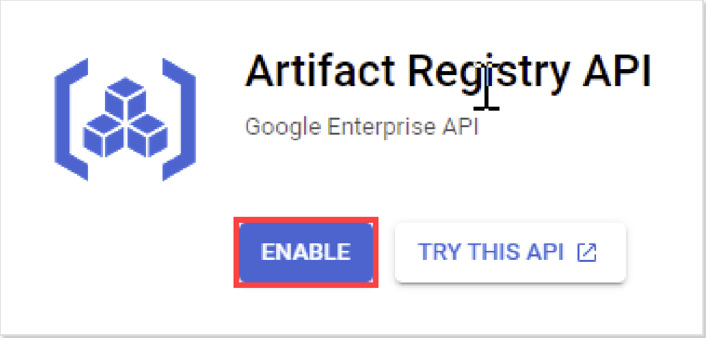
Step 5. Setting up a Container Registry (Optional)

This step is marked as optional because it will depend on what you want to do with your Kubernetes cluster in GCP. If you want to build your own containers, either for the instructions in this unit or for your own project/experiments, then you will need a registry which Kubernetes can pull from. You could use Docker Hub of course, which is publicly available and free for simple purposes, but here we provide instructions for configuring a registry in GCP. GCP offers two solutions: a Container Registry (deprecated), or the more general Artifact Registry, which we will configure here.

As for Kubernetes, we first need to activate the service. To do so, click on the hamburger icon and then select the Artifact Registry (in the CI/CD category), as shown:

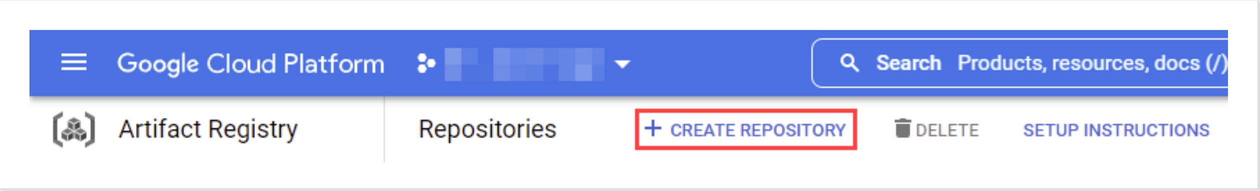


The browser will now display a page providing a summary of the Artifact Registry service. Click on the ENABLE button to activate this service, as shown:



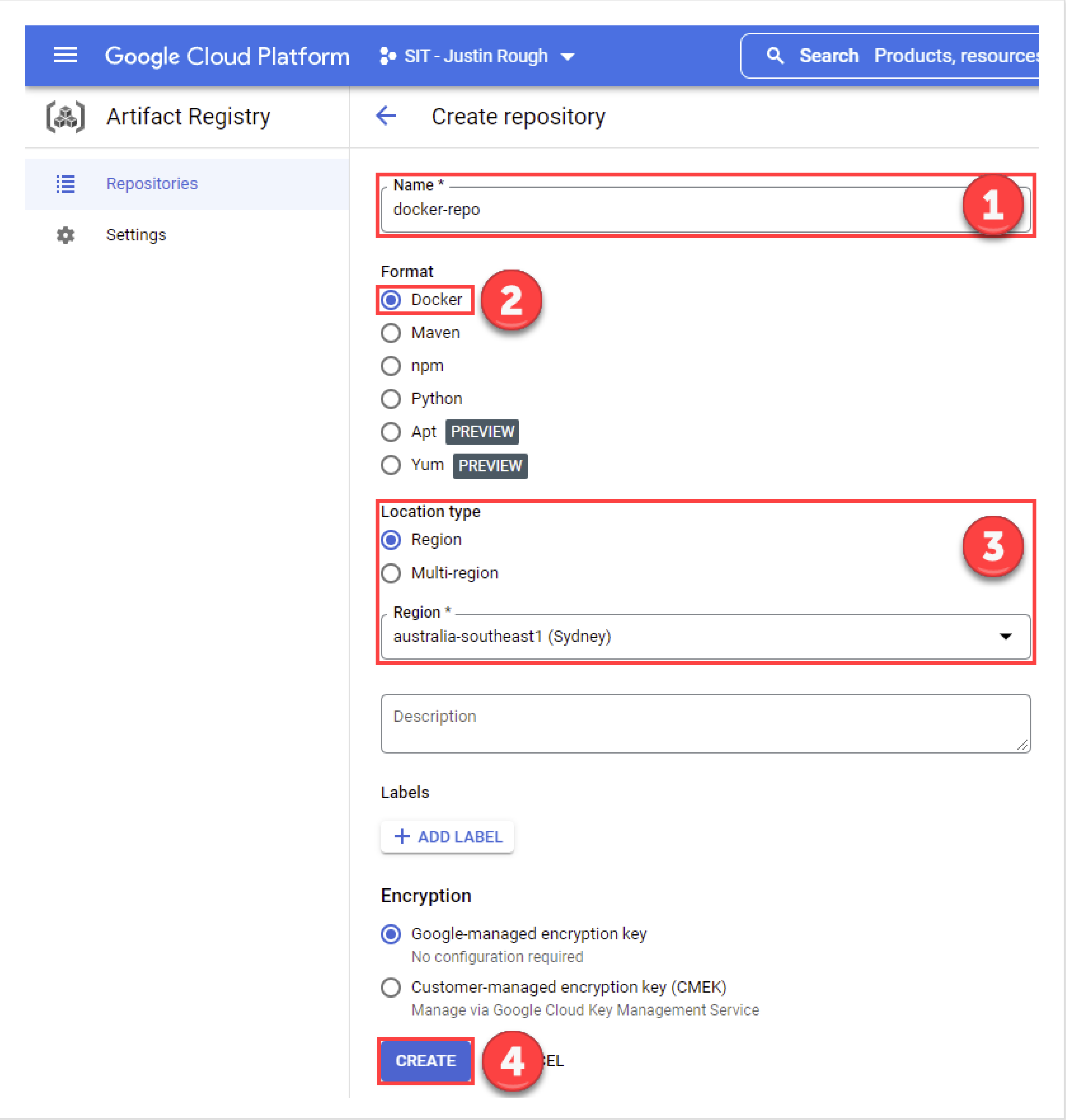
After a short while, GCP will refresh the page showing a summary of your use of the Artifact Registry service (nothing initially). We now need to create the container registry, which we do by

Registry service (nothing initially). We now need to create the container registry, which we do by clicking on the CREATE REPOSITORY button at the top of the browser screen, as shown:



Now we just need to provide the details of the repository, as follows. First, we need to provide a name for our repository (#1), which as before can be any name as it is scoped to your project. Second, make sure that the Docker format (#2) is selected, specifying we will be using this repository for containers. Third, select a region that is appropriate for you (#3) - in this case I have chosen australia-southeast1 (Sydney) to differ the Kubernetes Cluster for demonstration purposes, but normally you would select the same region. Finally, click on the CREATE button at the bottom of the screen (#4).

Complete the repository details as shown:



Name * docker-repo

Format
● Docker
○ Maven
○ npm
○ Python
○ Apt PREVIEW
○ Yum PREVIEW

Location type
● Region
○ Multi-region

Region * australia-southeast1 (Sydney)

Description

Encryption
● Google-managed encryption key
No configuration required
○ Customer-managed encryption key (CMEK)
Manage via Google Cloud Key Management Service

Critical Note: ***Although GCP returns quickly after you create the container registry at this point, it can take a while to propagate through Google's systems. This is a good point to take a break for 15 minutes or so to ensure this operation has fully completed. If you still encounter any errors when working with the container registry soon after its creation, just pause for a few minutes and then try again.***

Step 6. Using your Container Registry (Optional)

If you've setup a container registery in Step 5, using the registry requires: (1) authorise your cloud shell to use the registry, and (2) understand the path required to tag/push/pull containers with the registry.

Authorising your cloud shell to use the registry is completed using the using the following command:

```
gcloud auth configure-docker REGION-docker.pkg.dev
```

In this command, the *REGION* keyword needs to be replaced with the region that the registry was deployed to. For the example used in Step 5, the region is "australia-southeast1", so this command becomes:

```
gcloud auth configure-docker australia-southeast1-docker.pkg.dev
```

At this point you can now work with Docker to create container images, push them to your registry, and then use them from your Kubernetes cluster. For this purpose, you need to know how to write the tags however, which is as follows:

REGION-docker.pkg.dev/*PROJECT*/*REPOSITORY*/*IMAGENAME*:*TAG*

In this tag, *IMAGENAME*:*TAG* is the same as we learn in the unit (including a default *TAG* of 'latest' if omitted). Using the registry example from Step 5, this gives us the value for *REGION* ("australia-southeast1" as before) and also the *REPOSITORY* which refers to the name we gave our repository (#1). Finally the *PROJECT* is the name of your project, e.g., "sit727-22t1-example-abcd123". We can combine this into an example, for which we will use the Docker commands from the [Week 2 Lab]() on Docker Basics. In that lab, we used two Docker commands:

```
student@student-vm:~/my-container$ docker build -t localhost:5000/node-web .
```

```
student@student-vm:~/my-container$ docker push localhost:5000/node-web
```

Using our example container registry in GCP, these commands would become:

```
student@student-vm:~/my-container$ docker build -t australia-southeast1-
docker.pkg.dev/sit727-22t1-example-abcd123/docker-repo/node-web .
```

```
student@student-vm:~/my-container$ docker push australia-southeast1-
docker.pkg.dev/sit727-22t1-example-abcd123/docker-repo/node-web
```

‹ Previous Next ›

Edit HTML Reflect in ePortfolio Download Print

Alternative formats

‹ ›

Activity Details | Learning Objectives | Completion Summary

Visibility

✓ **Required: Automatic**

View this topic to complete the activity

Options

Reflecting in ePortfolio is enabled

Article Using Kubernetes on the Google Cloud Platform.html - Last Modified 08 April, 2022 10:23 AM