Build and Deploy a Docker Image to a Kubernetes Cluster

1 hour 30 minutes7 Credits

Rate Lab

**GSP304**



**Overview**

For this Challenge Lab you must complete a series of tasks within a limited time period. Instead of following step-by-step instructions, you'll be given a scenario and task - you figure out how to to complete it on your own! An automated scoring system (shown on this page) will provide feedback on whether you have completed your tasks correctly.

To score 100% you must complete all tasks within the time period!

When you take a Challenge Lab, you will not be taught GCP concepts. You'll need to use your advanced Google Compute Engine (GCE) skills to assess how to build the solution to the challenge presented. This lab is only recommended for students who have GCE skills. Are you up for the challenge?

**Topics tested**

* Build and tag a Docker Image of a sample application
* Push the tagged image to Google Container Registry
* Create a Kubernetes Cluster
* Deploy the application to the Kubernetes Cluster

**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 Cloud resources will be made available to you.

This Qwiklabs hand-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 the Google Cloud Platform 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 GCP account or project, do not use it for this lab.

**How to start your lab and sign in to the 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 you will see a panel populated with the temporary credentials that you must use for this lab.



1. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Choose an account** page.

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

1. On the Choose an account page, click **Use Another Account**.



1. The Sign in page opens. 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 GCP account, do not use it for this lab (avoids incurring charges).

1. 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 GCP console opens in this tab.

**Note:** You can view the menu with a list of GCP Products and Services by clicking the **Navigation menu** at the top-left, next to “Google Cloud Platform”. 

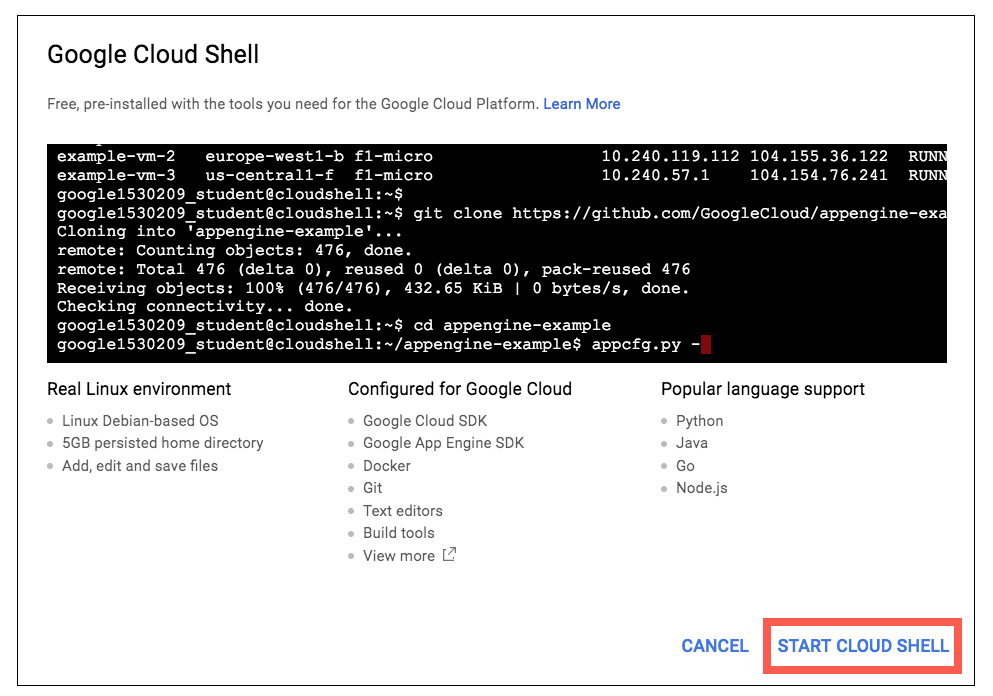
Activate Google Cloud Shell

Google 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. Google Cloud Shell provides command-line access to your GCP resources.

1. In GCP console, on the top right toolbar, click the Open Cloud Shell button.



1. In the dialog box that opens, click **START CLOUD SHELL**:



You can click "START CLOUD SHELL" immediately when the dialog box opens.

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 Platform. 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

Full documentation of **gcloud** is available on [Google Cloud gcloud Overview](https://cloud.google.com/sdk/gcloud).

**Challenge scenario**

Your development team is interested in adopting a containerized micro-services approach to application architecture. You need to test a sample application they have provided for you to make sure that that it can be deployed to a Google Cloud Platform Kubernetes container. The development group provided a simple Go application called echo-web with a Dockerfile and the associated context that allows you to build a Docker image immediately.

**Your challenge**

To test the deployment, you need to download the sample application and then build the Docker container image using a tag that allows it to be stored on the Google Container Registry. Once the image has been built, you'll push it out to the Google Container Registry before you can deploy it.

With the image prepared you can then create a Kubernetes cluster and then deploy the sample application to the cluster.

**Note:** In order to ensure accurate lab activity tracking you must use echo-appas the container repository image name, call your Kubernetes cluster echo-cluster, and use echo-web for the deployment name.

Build a tagged Docker Image

The sample application, including the Dockerfile and the application context files, are contained in an archive called echo-web.tar.gz. The archive has been copied to a Google Cloud storage bucket belonging to your lab project called gs://[PROJECT\_ID]. You must deploy this with a tag called v1.

Push the image to the Google Container Registry

Your organization has decided that it will always use the gcr.io Google Container Registry hostname for all projects. The sample application is a simple web application that reports some data describing the configuration of the system where the application is running. It is configured to use TCP port 8000 by default.

Create a Kubernetes Cluster

Your test environment is limited in capacity, so you should limit the test Kubernetes Cluster you are creating to just two N1-standard-2 instances. You must call your cluster echo-cluster.

**Deploy the application to the Kubernetes Cluster**

Even though the application is configured to respond to HTTP requests on port 8000, you must configure the service to respond to normal web requests on port 80. When configuring the cluster for your sample application, call your deployment echo-web.

**Troubleshooting**

**Receiving a 504, Gateway timeout error:** This might just indicate that the application hasn't quite initialized yet, but it could also be caused by a mismatch between the default port that is set in the Dockerfile (TCP port 8000) and the choice of application port you configured when deploying the application image, or when you configured external access.

**Congratulations!**

You have completed the challenge lab.