Lab2 Infrastructure preview

project ID is globally unique, meaning that no one else can be using that project ID.

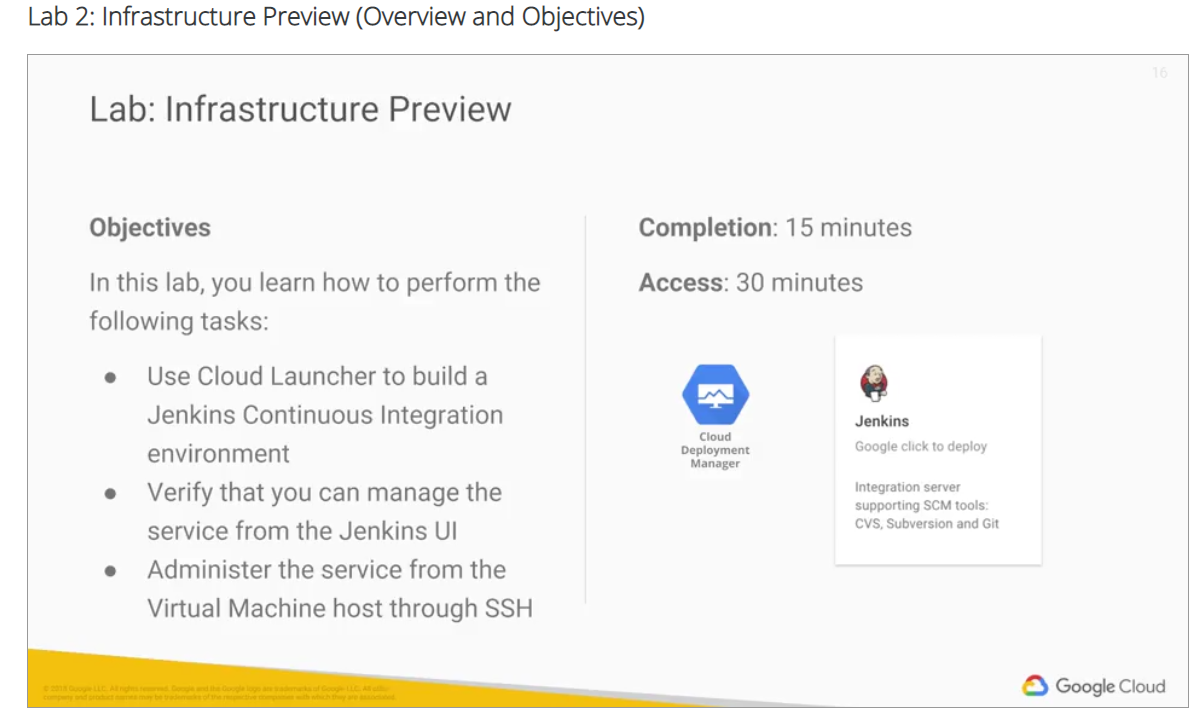
project number is also a globally unique identifier and cannot be changed either once the project is created.

$gcloud config list

Another way to do this is you can actually just switch the project directly within the command line. A good strategy is always to store some of the IDs and project numbers that you have within a shell variables so let's go ahead and do that. I'm going to create a variable called PROJECT\_1\_ID and I'm going to get that project ID. Just copy it right here and paste it directly into Cloud Shell. So now, I have a variable that is holding that. Now, the command to actually change the project is

$gcloud config set project ," and I'm just going to use that variable then. So, Project\_1\_ ID, and this says it updated it now. We can see that because the command line here has updated, and it can also again query for that project, and we can see that now my console is at that project and my Cloud Shell is also changed. So, in this demo, I showed you how projects are created and deleted, how to switch context between projects, and how to use Cloud Shell with projects.

Most importantly, the console's current project is independent from the Cloud Shell's current projects. But as you saw, you can use the Cloud Shell command prompt as an indicator and you can issue a command to change the current project in Cloud Shell.



# Task 1: Use Marketplace to build a deployment

## Navigate to Marketplace

1. In the GCP Console, on the **Navigation menu** ( 7a91d354499ac9f1.png), click **Marketplace**.
2. Locate the Jenkins deployment by searching for **Jenkins Certified by Bitnami**.
3. Click on the deployment and read about the service provided by the software.

Jenkins is an open-source continuous integration environment. You can define jobs in Jenkins that can perform tasks such as running a scheduled build of software and backing up data. Notice the software that is installed as part of Jenkins shown in the left side of the description.

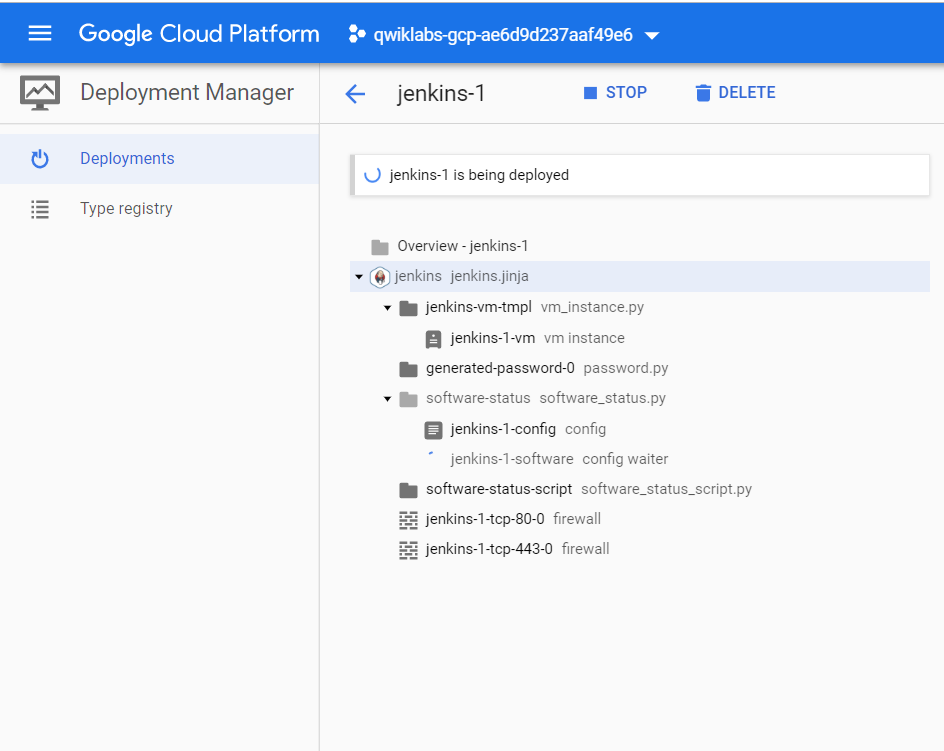
The service you are using, Marketplace, is part of Google Cloud Platform. The Jenkins template is developed and maintained by an ecosystem partner named Bitnami. Notice on the left side a field that says "Last updated." How recently was this template updated? 6/12/19,12:34am

The template system is part of another GCP service called Deployment Manager. Later in this class you learn how templates such as this one can be built. That service is available to you. You can create templates like the one you are about to use.

In a class that was previously offered, students would set up a Jenkins environment similar to the one you are about to launch. It took about two days of labs to build the infrastructure that you will achieve in the next few minutes.

## Launch Jenkins

1. Click **Launch on Compute Engine**.
2. Verify the deployment, accept the terms and services and click **Deploy**.
3. Click **Close** on the Welcome to Deployment Manager window.



It will take a minute or two for Deployment Manager to set up the deployment. You can watch the status as tasks are being performed. Deployment Manager is acquiring a virtual machine instance and installing and configuring software for you. You will see **jenkins-1 has been deployed** when the process is complete.

Deployment Manager is a GCP service that uses templates written in a combination of YAML, python, and Jinja2 to automate the allocation of GCP resources and perform setup tasks.

Behind the scenes a virtual machine has been created. A startup script was used to install and configure software, and network Firewall Rules were created to allow traffic to the service.

# Task 2: Examine the deployment

In this section, you examine what was built in GCP.

## View installed software and login to Jenkins

1. In the right pane, click **More about the software** to view additional software details. Look at all the software that was installed.
2. Copy the **Admin user** and **Admin password** values to a text editor.
3. Click **Visit the site** to view the site in another browser tab. If you get an error, you might have to reload the page a couple of times.
4. Log in with the **Admin user** and **Admin password** values.
5. After logging in, you will be asked to Customize Jenkins. Click **Install suggested plugins**, and then click **Restart** after the installation is complete. The restart will take a couple of minutes.

Note: If you are getting an installation error, **retry** the installation and if it fails again, **continue** past the error and **save and finish** before restarting. The code of this solution is managed and supported by Bitnami.

## Explore Jenkins

1. In the Jenkins interface, in the left pane, click **Manage Jenkins**. Look at all of the actions available. You are now prepared to manage Jenkins. The focus of this lab is GCP infrastructure, not Jenkins management, so seeing that this menu is available is the purpose of this step.
2. Leave the browser window open to the Jenkins service. You will use it in the next task.

Now you have seen that the software is installed and working properly. In the next task you will open an SSH terminal session to the VM where the service is hosted, and verify that you have administrative control over the service.

# Task 3: Administer the service

## View the deployment and SSH to the VM

1. In the GCP Console, on the **Navigation menu**( 7a91d354499ac9f1.png), click **Deployment Manager**.
2. Click **jenkins-1**.
3. Click **SSH** to connect to the Jenkins server.

The Console interface is performing several tasks for you transparently. For example, it has transferred keys to the virtual machine that is hosting the Jenkins software so that you can connect securely to the machine using SSH.

## Shut down and restart the services

1. In the SSH window, enter the following command to shut down all the running services:

sudo /opt/bitnami/ctlscript.sh stop

1. Refresh the browser window for the Jenkins UI. You will no longer see the Jenkins interface because the service was shut down.
2. In the SSH window, enter the following command to restart the services:

sudo /opt/bitnami/ctlscript.sh restart

1. Return to the browser window for the Jenkins UI and refresh it. You may have to do it a couple of times before the service is reachable.
2. In the SSH window, type exit to close the SSH terminal session.

# Task 4: Review

In a few minutes you were able to launch a complete Continuous Integration solution. You demonstrated that you had user access through the Jenkins UI, and you demonstrated that you had administrative control over Jenkins by using SSH to connect to the VM where the service is hosted and by stopping and then restarting the services. Many of the activities that occurred in this lab, were nearly transparent and they use resources and methods that you'll learn about in the rest of the specialization. For example, the acquisition and configuration of a network IP address, the provisioning of a virtual machine instance along with the installation of software on that machine, and the passing of default state information from the environment during the setup process.