Cisco CloudCenter Suite 5.0 with ACI 4.0 v1

Last Updated: 10-April-2019

About This Demonstration

This guide for the preconfigured demonstration includes:

About This Demonstration

Requirements

About This Solution

Topology

Get Started

Scenario 1. User Role - Application Deployment

Scenario 2. Architect Role – Editing Application Profiles

Scenario 3. Admin Role – Create Constructs Within CloudCenter

Scenario 4. CI/CD Solution using GitLab, Jenkins, Artifactory and CloudCenter

Limitations

- Certain features of Cisco APIC 4.0 are outside the scope of this demonstration, because the demonstration uses a simulated fabric rather than a physical fabric:
  - All configuration will be lost after a reboot of the APIC simulator
  - No traffic will pass between devices connected to the simulated fabric for this reason we have 2 network interfaces on the CloudCenter deployed application.
  - Screen refresh may take slightly longer than expected
- Explanation of the features of Jenkins, Artifactory, and Git Bash are outside the scope of this demonstration.

Requirements

The table below outlines the requirements for this preconfigured demonstration.

Required	Optional
Laptop	Cisco AnyConnect®

About This Solution

CloudCenter provides customers with a single, intuitive platform that helps them manage the entire application lifecycle across simple or complex hybrid IT environments. The CloudCenter platform provides a compelling solution for modern IT organizations whether they are moving their first applications to the cloud, implementing self-service IT, or wanting to gain visibility and control across a vast portfolio of clouds, applications and users.

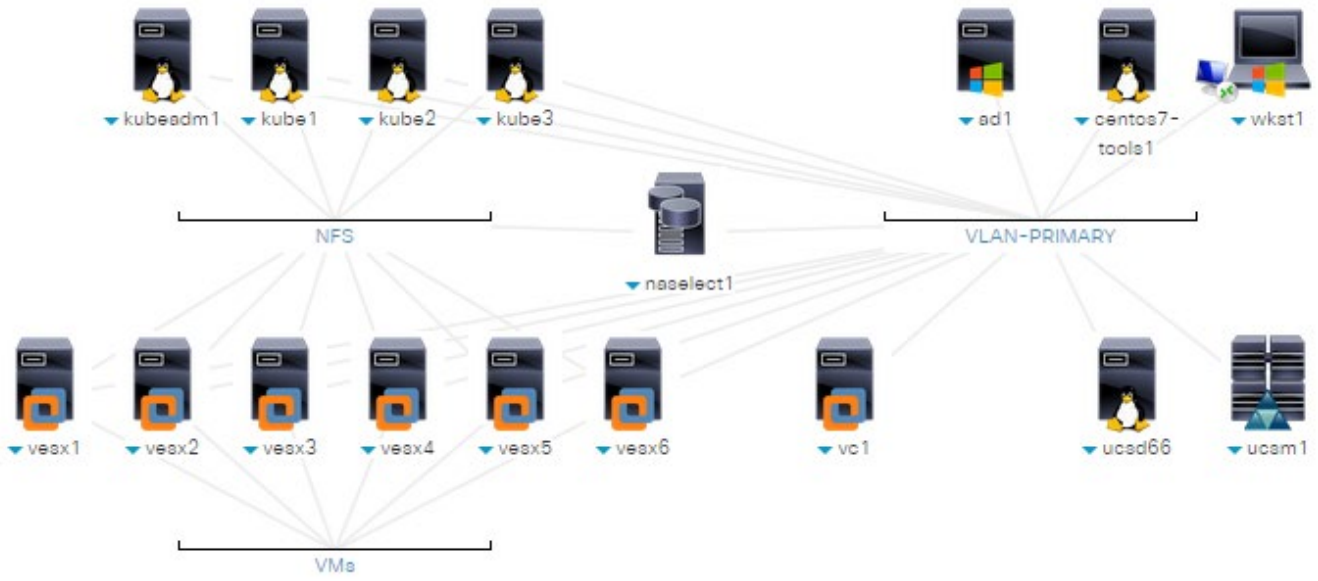
The solution provides customers with several key benefits:

- Global profile deployment: CloudCenters solution allows customers to create a single, cloud agnostic application profile once and then deploy it across any data center, public or private cloud.
- Consistent policy application: CloudCenter automatically applies a customers access control and security policies to an application, and then ensures that those policies move with the application.
- Application optimization across hybrid cloud environments: CloudCenter will measure both price and performance of applications on any cloud environment, helping users to make informed decisions about the best place for their application on any data center or cloud.
- One-click management: CloudCenter provides a single management interface to give customers complete visibility and control across applications, cloud environments and users.

Topology

This content includes preconfigured users and components to illustrate the scripted scenarios and features of the solution. Most components are fully configurable with predefined administrative user accounts. You can see the IP address and user account credentials to use to access a component by clicking the component icon in the Topology menu of your active session and in the scenario steps that require their use.

Figure 1. dCloud Topology



Get Started

BEFORE PRESENTING

Cisco dCloud strongly recommends that you perform the tasks in this document with an active session before presenting in front of a live audience. This will allow you to become familiar with the structure of the document and content.

It may be necessary to schedule a new session after following this guide in order to reset the environment to its original configuration.

PREPARATION IS KEY TO A SUCCESSFUL PRESENTATION.

Follow the steps to schedule a session of the content and configure your presentation environment.

1. Initiate your dCloud session. [Show Me How]

NOTE: It may take up to 10 minutes for your session to become active.

2. For best performance, connect to the workstation with **Cisco AnyConnect VPN** [Show Me How] and the **local RDP client** on your laptop [Show Me How]
   - **Workstation** 1: 198.18.133.36, **Username**: DCLOUD\demouser, **Password**: C1sco12345

NOTE: You can also connect to the workstation using the Cisco dCloud Remote Desktop client [Show Me How]. The dCloud Remote Desktop client works best for accessing an active session with minimal interaction.

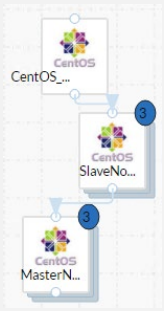
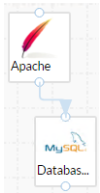
Scenario 1. User Role - Application Deployment

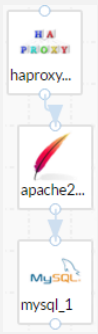
The purpose of this scenario is to deploy an application that can be monitored and managed by Cisco CloudCenter.

NOTE: If demonstrating to a customer, it is recommended that a deployment of an application be completed before demonstrating to the customer. This will decrease waiting time.

Any application in the Application Profiles list can be used for this scenario, depending on customer interest. The table below shows the applications that exist by default in the Application Profiles list.

Pre-installed Application Profiles

APPLICATION NAME	TOPOLOGY	NUMBER/TYPE OF VMs	APPROXIMATE DEPLOYMENT TIME IN MINUTES	NOTES
Applications (with ACI)				
Apache Mesos		1 – CentOS_6 Load Balancer VM 3 – CentOS_6 Master Nodes 3 – CentOS_6 Slave Nodes	10 minutes	Complex application, longer deployment time, confirmation page with application connectivity
Magento		1 – Nginx Load Balancer VM 1 – Apache2 Web VM 1 – MySQL Database VM	8 minutes	
MySQL		1 – CentOS_6 MySQL Node	4 minutes	
OpenCart		1 – Apache Web VM 1 – MySQL Database VM	7 minutes	

APPLICATION NAME	TOPOLOGY	NUMBER/TYPE OF VMs	APPROXIMATE DEPLOYMENT TIME IN MINUTES	NOTES
WordPress		1 – HAproxy Load Balancing VM 1 – Apache2 Web VM 1 – MySQL Database VM	8 minutes	Complex application, medium deployment time
Applications (Containerized)				
Artifactory OSS		1 – Artifactory OSS container	2 minutes	
GitLab CE		1 – GitLab CE container	2 minutes	
Joomla		1 – Joomla container 1 – MySQL container	3 minutes	
Odoo		1 – Odoo container 1 – PostgreSQL container	3 minutes	
OpenCart		1 – OpenCart container 1 – MySQL container	3 minutes	

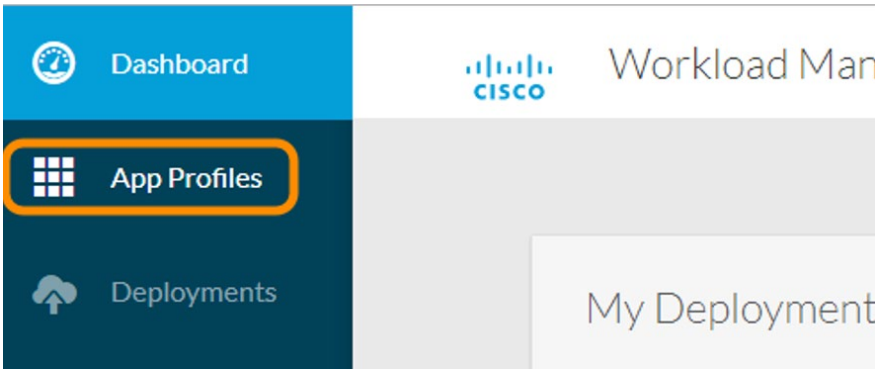
APPLICATION NAME	TOPOLOGY	NUMBER/TYPE OF VMs	APPROXIMATE DEPLOYMENT TIME IN MINUTES	NOTES
WordPress		1 – WordPress container 1 – MySQL container	3 minutes	
SINGLE VM WITH OS INSTALLED (visible by Admin only)				
CentOS 6		1 – CentOS 6 VM	3 minutes	
CentOS 7		1 – CentOS 7 VM	3 minutes	

Steps

Deploy a VM-Based Application in Production (with ACI)

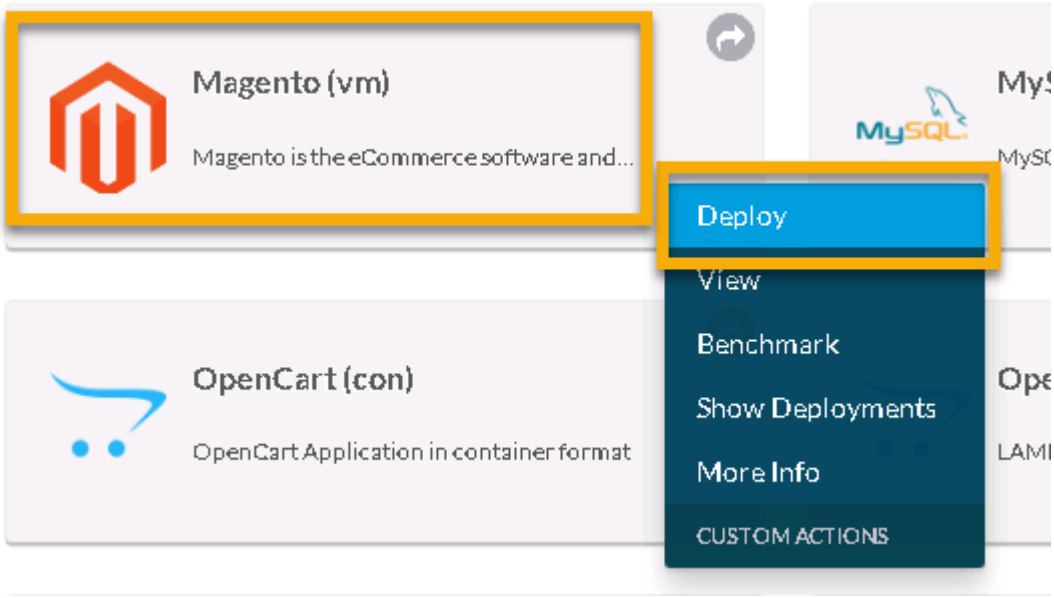
1. Open a Chrome browser and perform the following actions:
    a. Open a new browser tab and click the CloudCenter Suite bookmark. Log in using the credentials **user@dcloud.cisco.com** / **C1sco12345**, tenant id: **demo**.
    b. Open a second browser tab and click the **vCenter (HTML)** bookmark. Select the **Use Windows session authentication** and click on **Login** to login as demouser.
    c. Open a third browser tab and click the **Cisco APIC** bookmark. Log in using the credentials **admin / C1sco12345**.
2. Switch to the **CloudCenter** browser tab.
3. If not at the Workload Manager screen, click on the **Dashboard icon**.

4. From the side menu, click **App Profiles** to display the available Application Profiles.

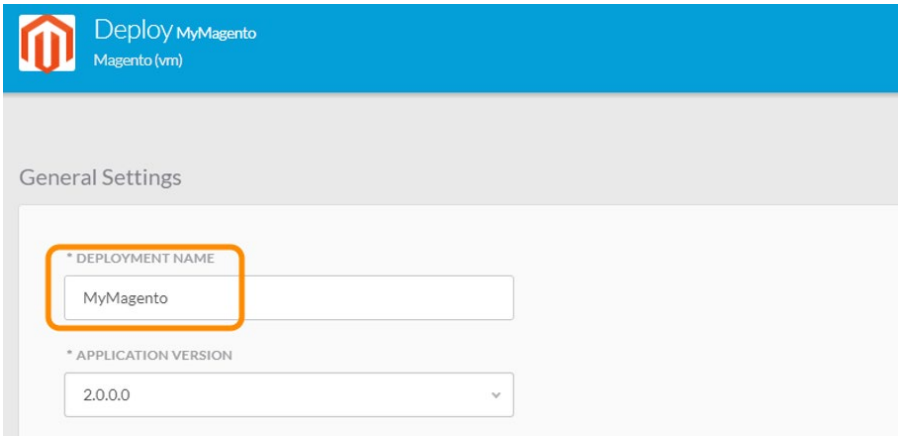


NOTE: Cisco CloudCenter provides role-based and object-based access control, so that when a user logs in, the content provided to them is based on their profile.

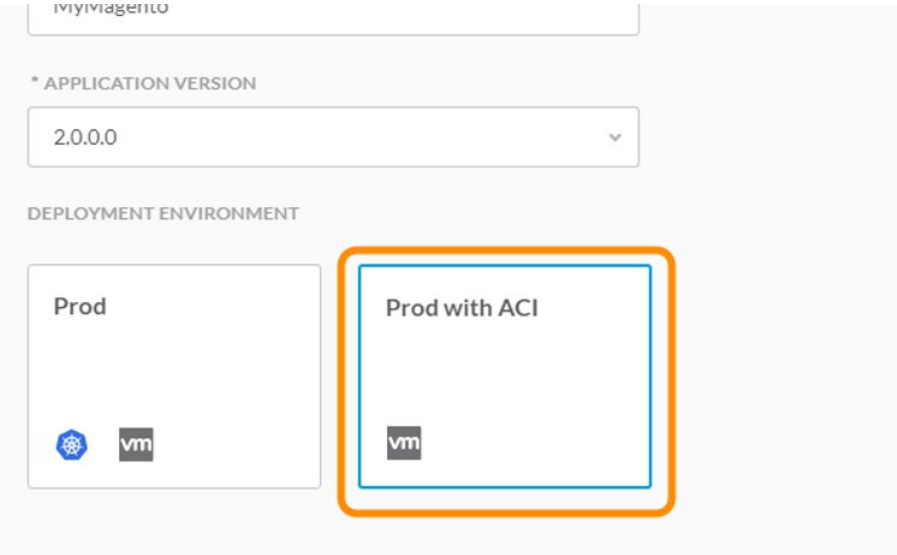
5. Mouse over the **Magento (vm) Application Profile** and click **Deploy** to deploy a Magento application.



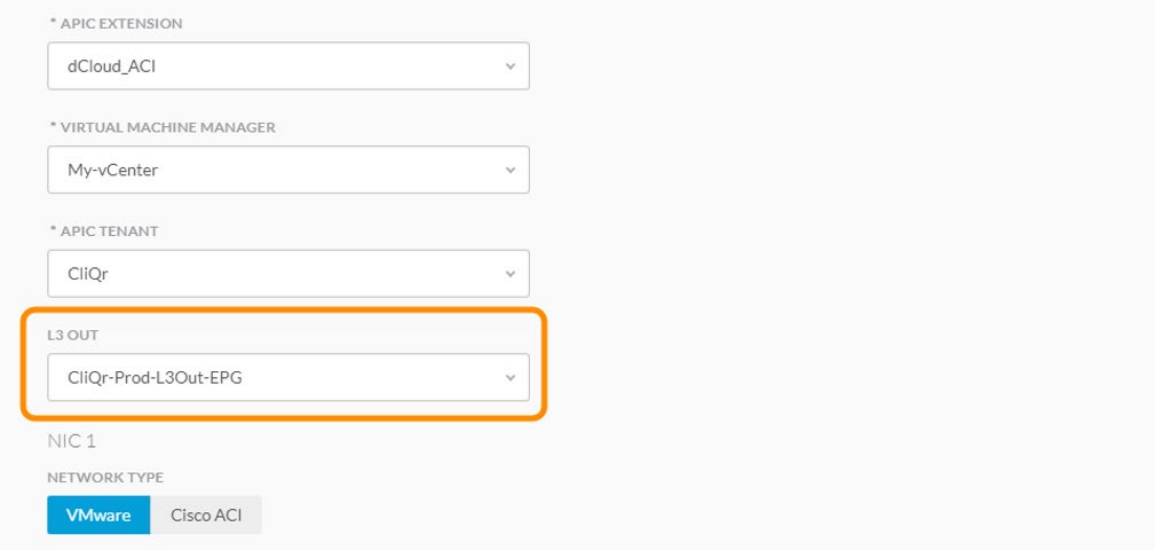
6. Enter **MyMagento** at the Deployment Name.



7. Choose **Prod with ACI** at the Deployment Environment.

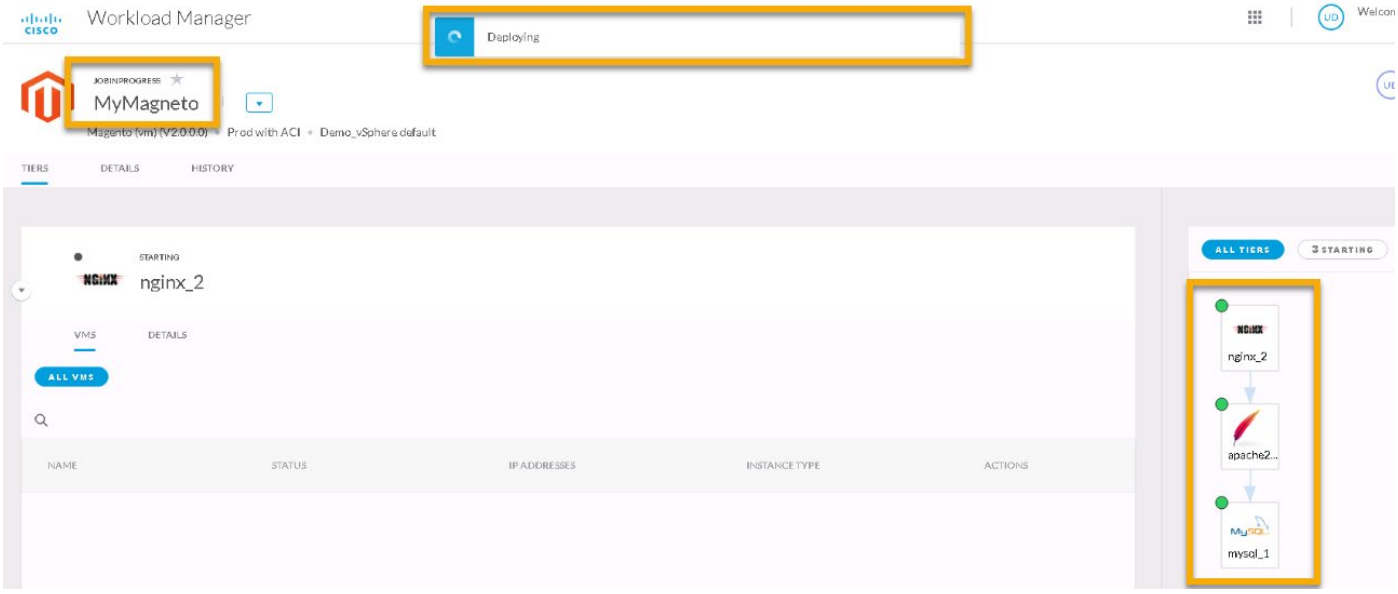


8. Leave the other fields to their default values and click **Next** at the bottom of the form.
9. At the next form, explore the **default selected options** (Cloud, Tiers, Instance Types, Network settings).
10. At the nginx_2 tier, scroll to the network section and at the **L3 OUT** field, choose **CliQr-Prod-L3Out-EPG**.



11. At the bottom of the form, click **Deploy**.

12. The screen should change to the deployment form, where each tier will be deployed in sequence. Once the application is fully deployed, an **Access Application Button** will appear at the top of the form.



13. Wait for the application to fully deploy and proceed with the verification steps below. To verify the application using the Access Application button, skip to Step 3 in the Verify Application Deployment section

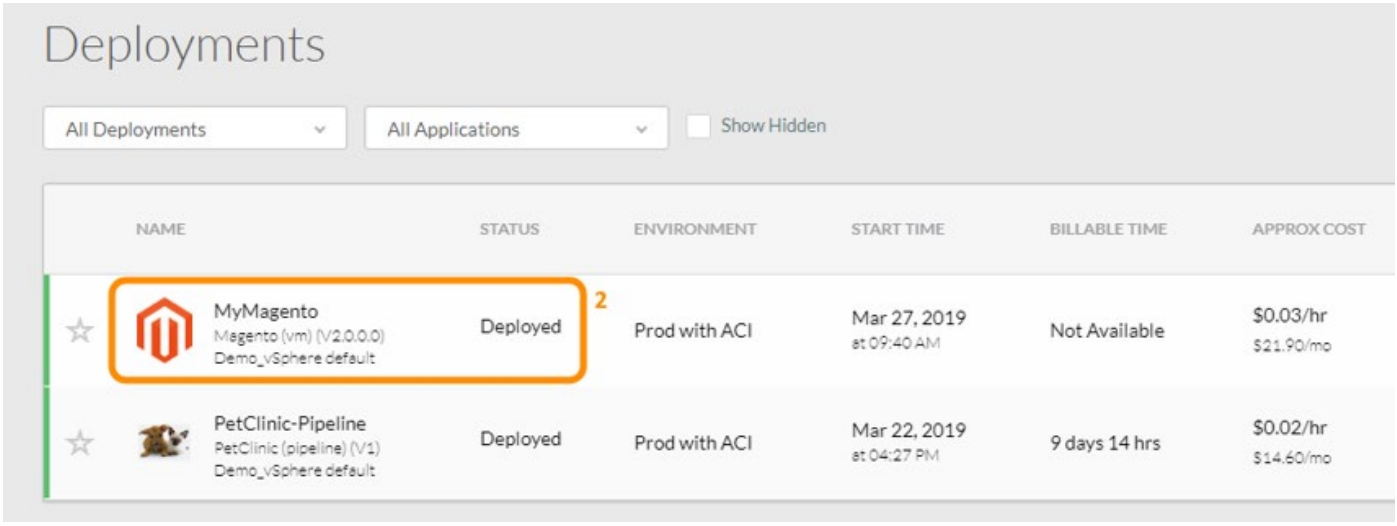
NOTE: If you pre-configured an application prior to the demonstration, you can proceed immediately with the following steps. If not, it may take 5-10 minutes for the application to fully deploy and show the **Access Application** button.

Verify Application Deployment

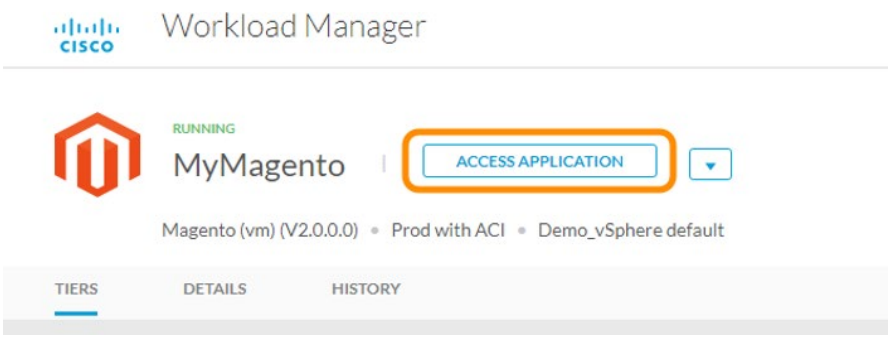
1. On the left menu, click **Deployments**. 

2. Click **MyMagento** on the previously deployed application (the rest of this scenario is based on MyMagento).

NOTE: Wait until the applications Status is Deployed before continuing.

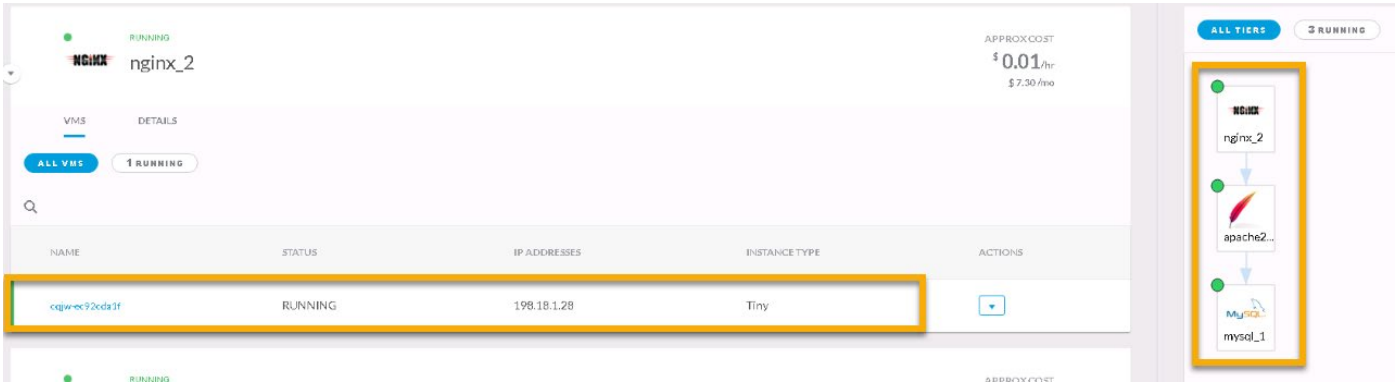


14. On the resulting screen, click the **Access Application** button to proceed to the application.

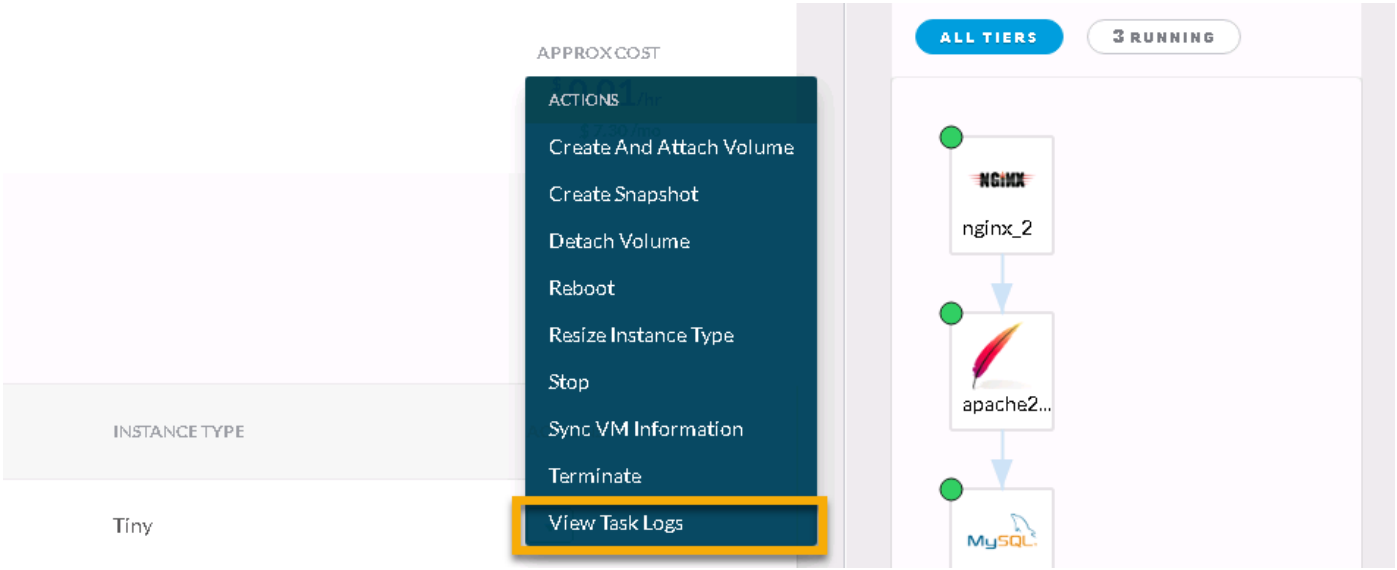


15. When the Magento page is displayed in the Chrome tab, **close the tab**. (Magento configuration is outside the scope of this demonstration.)

16. Return to the **Cisco CloudCenter tab** and observe the green circles indicating the health of the deployed services. Then observe the allocated VM names and IP Addresses. Finally, observe the current status of the VMs.

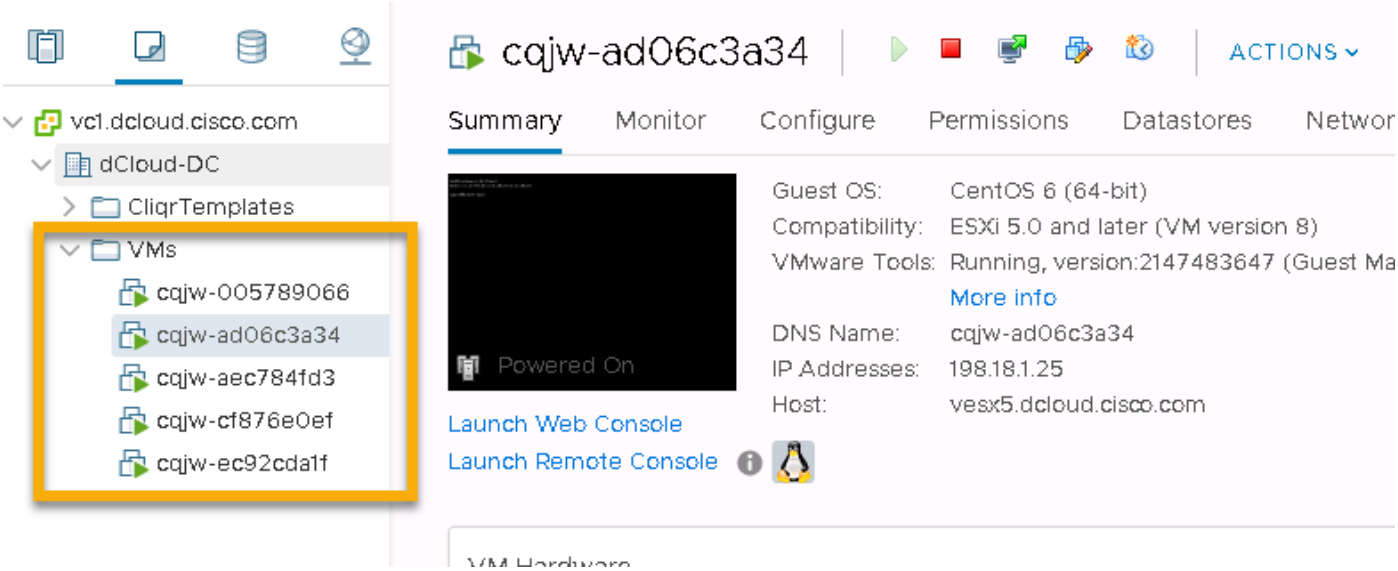


17. Next, click at the ▾ icon underneath the **Actions** to view the available actions for this server. Click on the **View Task Logs** option to view the actions performed against this server during deployment.

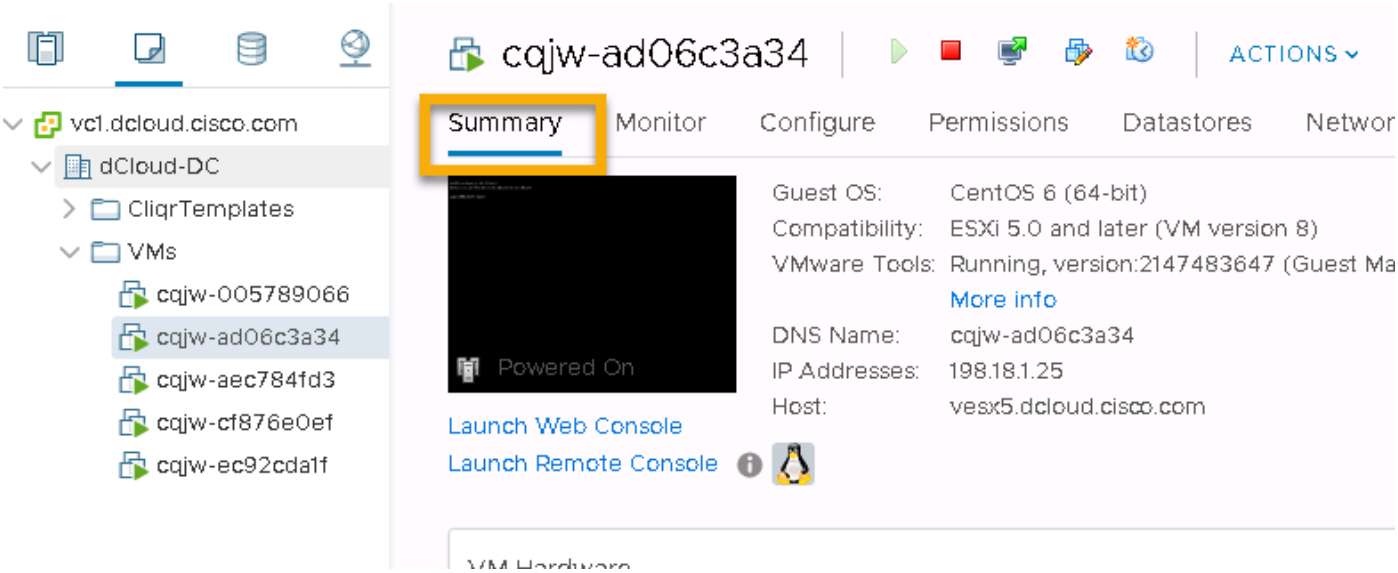


18. Switch to the **vSphere browser** tab. Log on again if the session has been timed out.
19. Click the **VMs and Templates** icon.

20. Expand the **dCloud-DC data center tree** and open the **VMs folder**. The VMs from the deployment should be in the folder vm list.

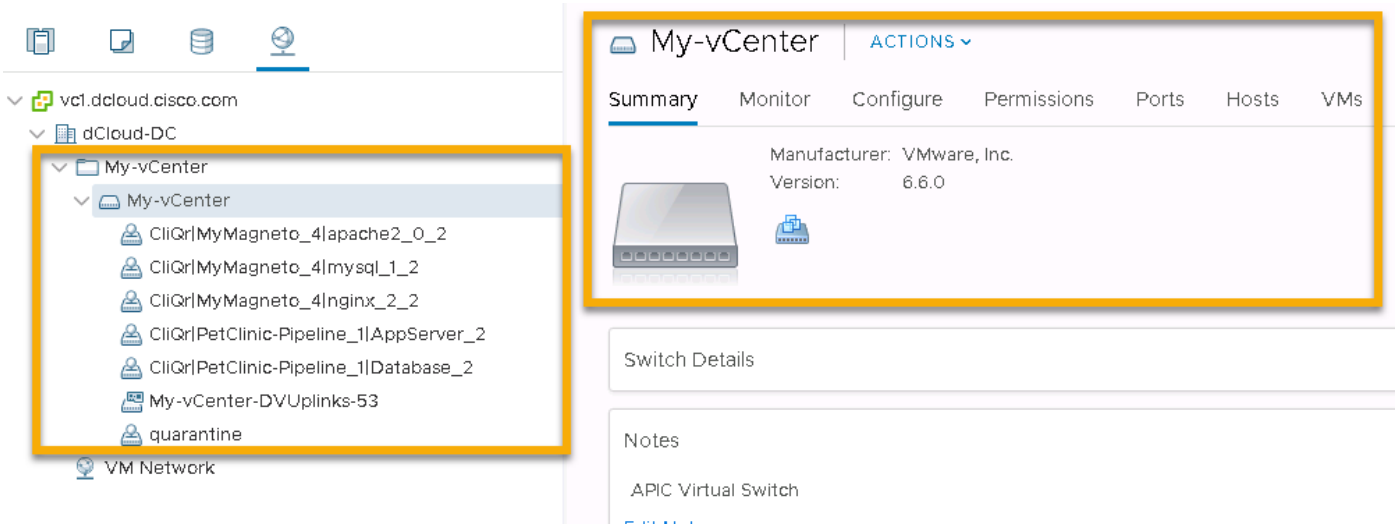


21. Click on one of the **deployment VMs** and click the **Summary** tab.

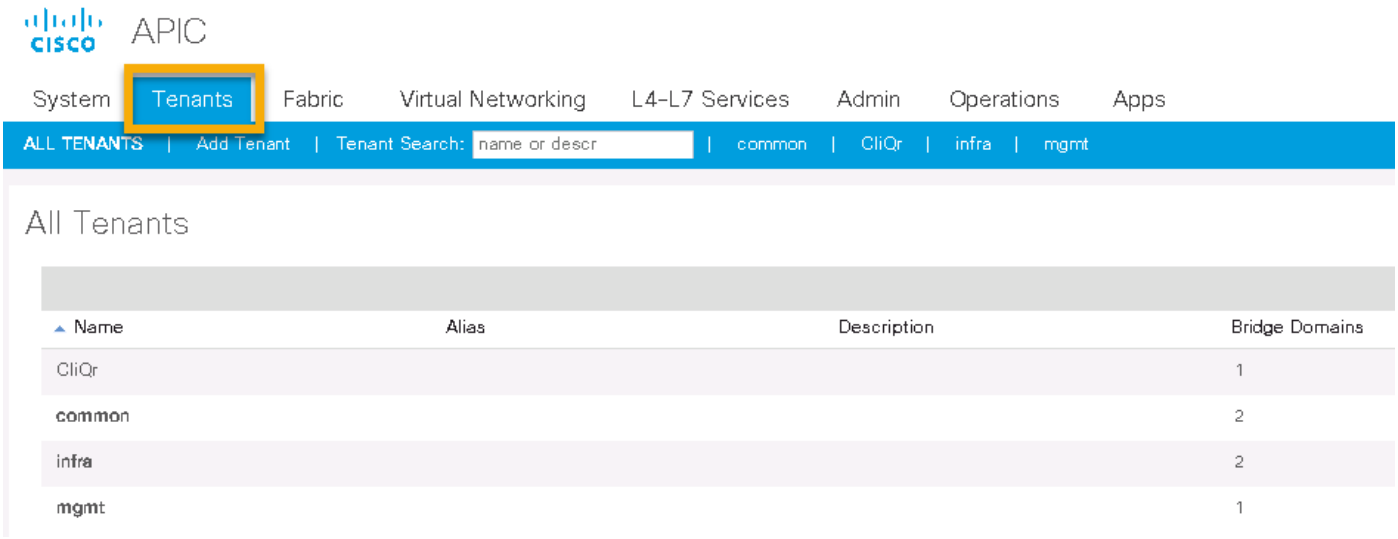


22. Click on one **Networking** icon to inspect the current network configuration

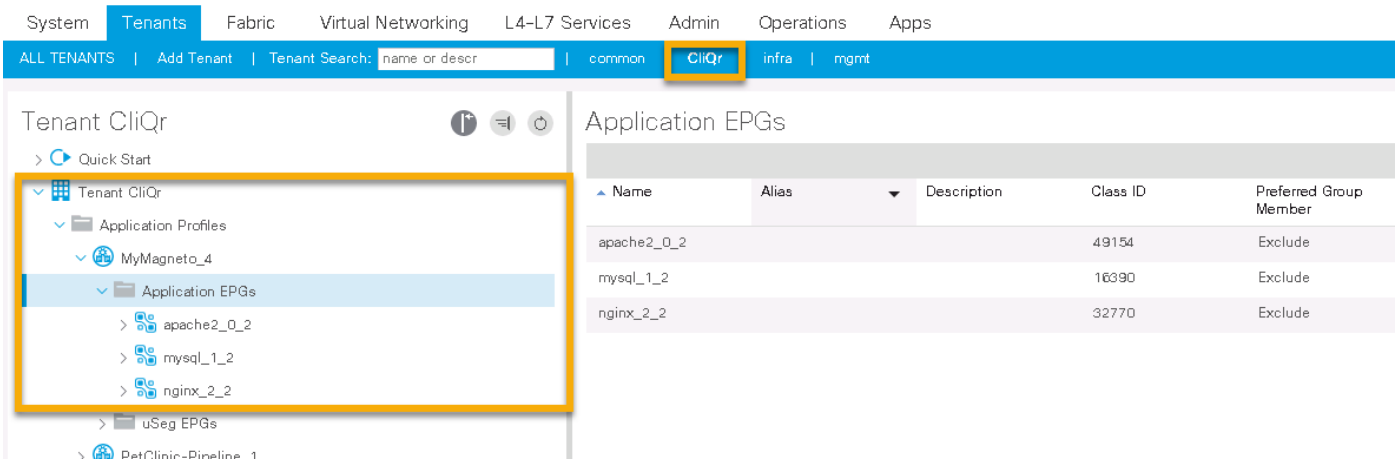
23. Expand the dCloud-DC data center tree and open the **My-vCenter folder**, then open the **My-vCenter switch**. Notice the ACI configuration that has been deployed for this application (CliQr|MyMagento_XXX|{tier name}).



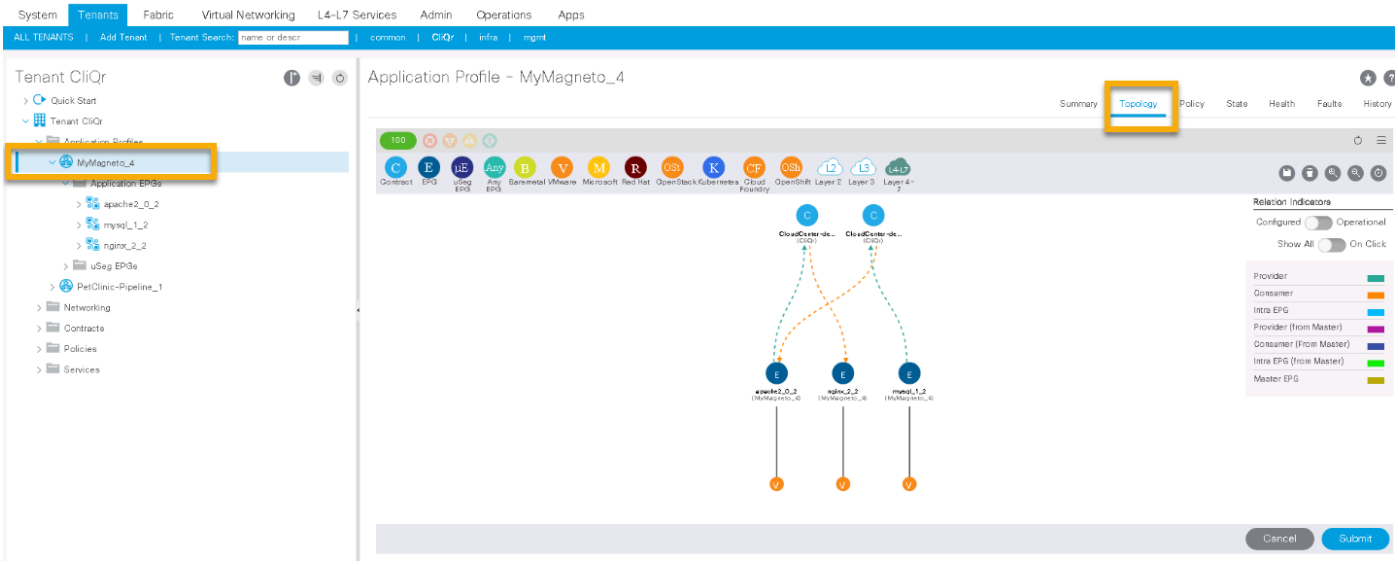
24. Next, switch to the Cisco APIC browser tab (logon again if the session has been timed out) and click on **Tenants**.



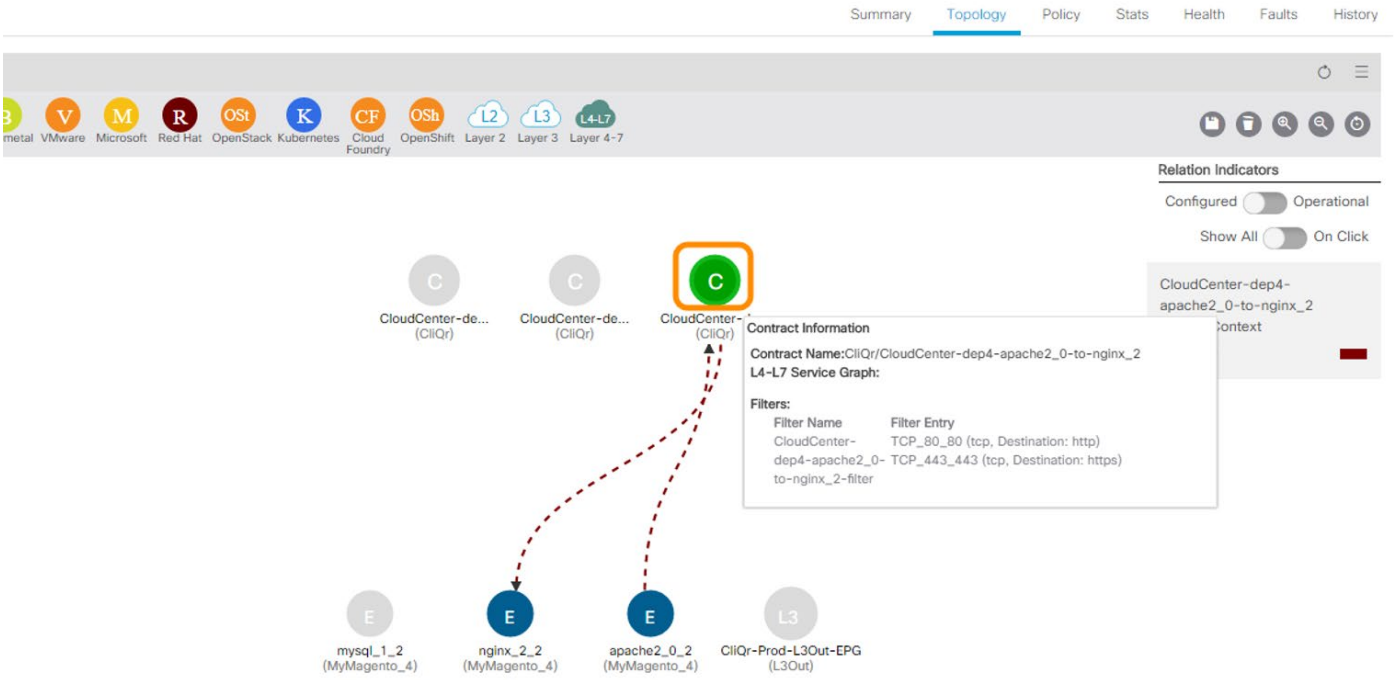
25. Click on the **CliQr Tenant**.
26. Under Tenant CliQr, open the **Application Profiles folder**, then the **MyMagento_XXX app profile** and then the **Application EPGs folder.**
27. Notice the newly created EPGs that correspond to the Application Tiers deployed from CloudCenter.



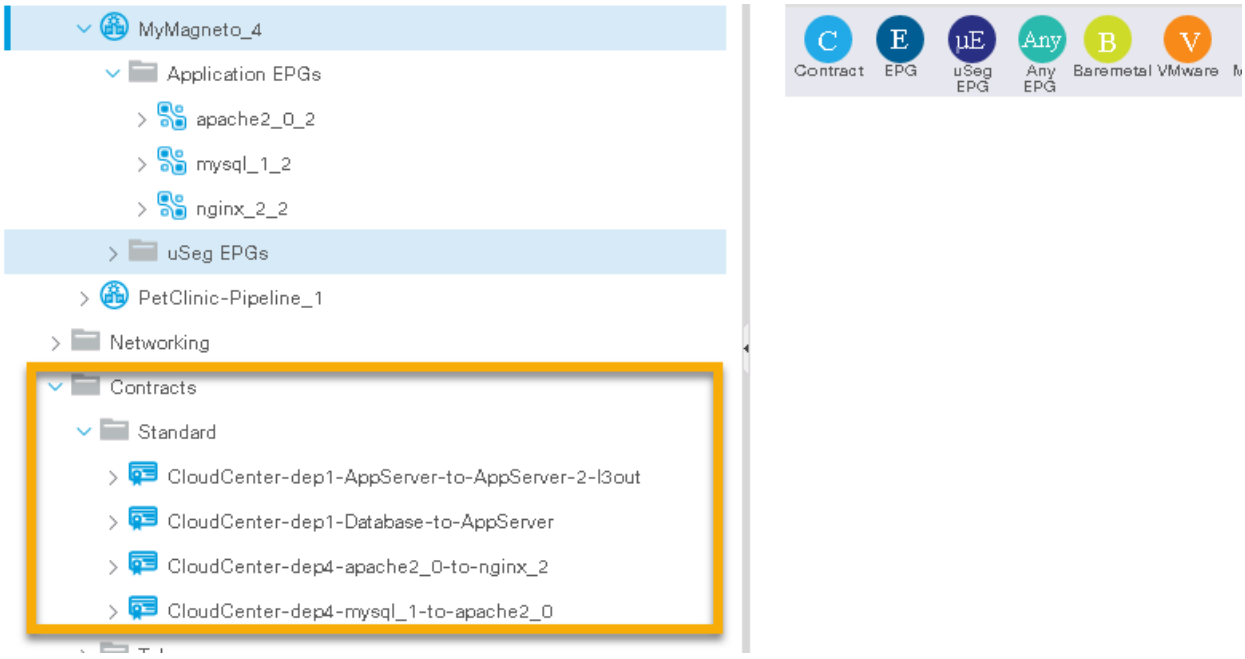
28. To view the currently configured ACI policy for communication between the EPGs, click on the **MyMagento_XXX** app profile and then click **Topology** at the right-hand side of the window.



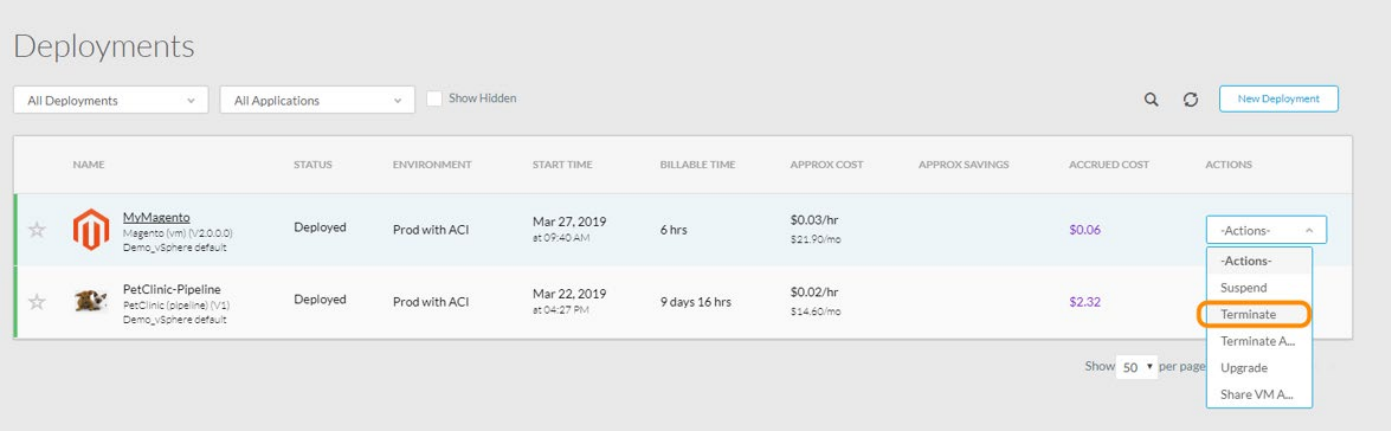
29. Click on one of the contracts the view more details on the traffic policy being applied by the specific contract.



30. Next, expand the **Contracts** folder to view more details about the contracts configured on the ACI, including the contracts created by the CloudCenter deployment.

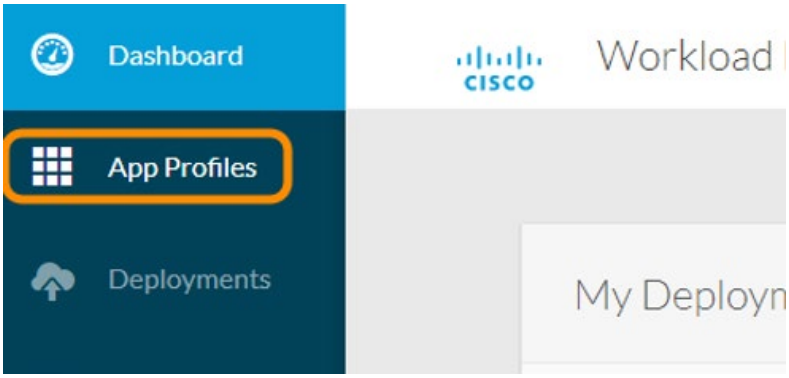


31. Optionally, return to the CloudCenter deployments and terminate the MyMagento deployment (it may take a few minutes for the deployment to be terminated). Then switch back to the Cisco APIC web page and observe the ACI objects related to the CloudCenter deployment having been removed.



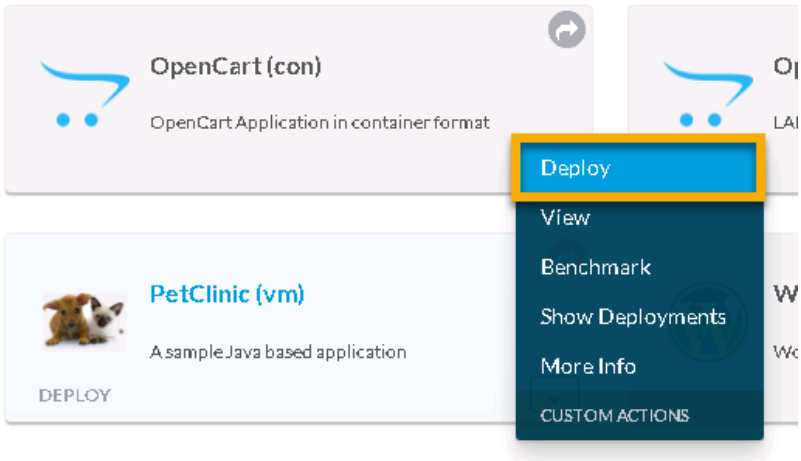
Deploy a Container-Based Application in Production

1. Open a Chrome browser and perform the following actions:
    a. Open a new browser tab and click the **CloudCenter Suite** bookmark. Log in using the credentials user@dcloud.cisco.com / C1sco12345, tenant id: demo.
    b. Open a second browser tab and click the **Kubernetes Dashboard** bookmark. Select the Token authentication method. Locate the file **Demo Kubernetes Cluster - clusteradmin Token.txt** saved on the desktop and copy/paste its contents to the **Token** field, then click **Sign In**.
2. Switch to the **CloudCenter** browser tab.
3. If not already open to **Workload Manager** screen, click on the **Dashboard** module.
4. From the side menu, click **App Profiles** to display the available Application Profiles.

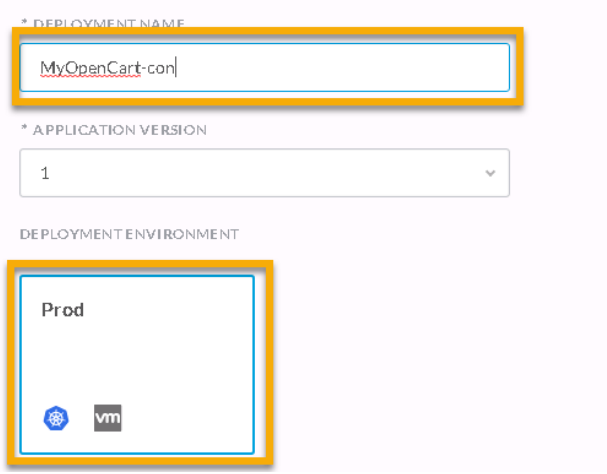


NOTE: Cisco CloudCenter provides role-based and object-based access control, so that when a user logs in the content provided to them is based on their profile.

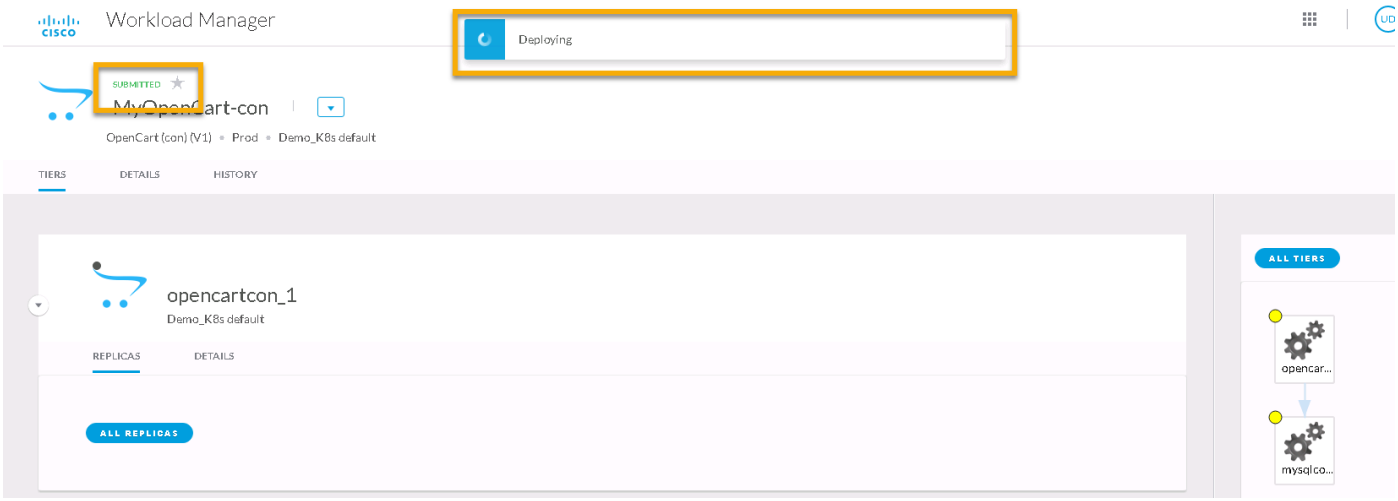
5. Mouse over the **OpenCart (con) Application Profile** and click **Deploy** to deploy an OpenCart application.



6. Enter **MyOpencart-con** at the Deployment Name field.
7. Choose **Prod** at the Deployment Environment.



8. Leave the other fields to their default values and click **Next** at the bottom of the form.
9. At the next form, explore the default selected options (Cloud, 2 Tiers, Namespace, Instance Types) and then click **Deploy**.
10. The screen displays the deployment form, where each tier will be deployed in sequence. Once the application is fully deployed, an **Access Application Button** should appear at the top of the form.



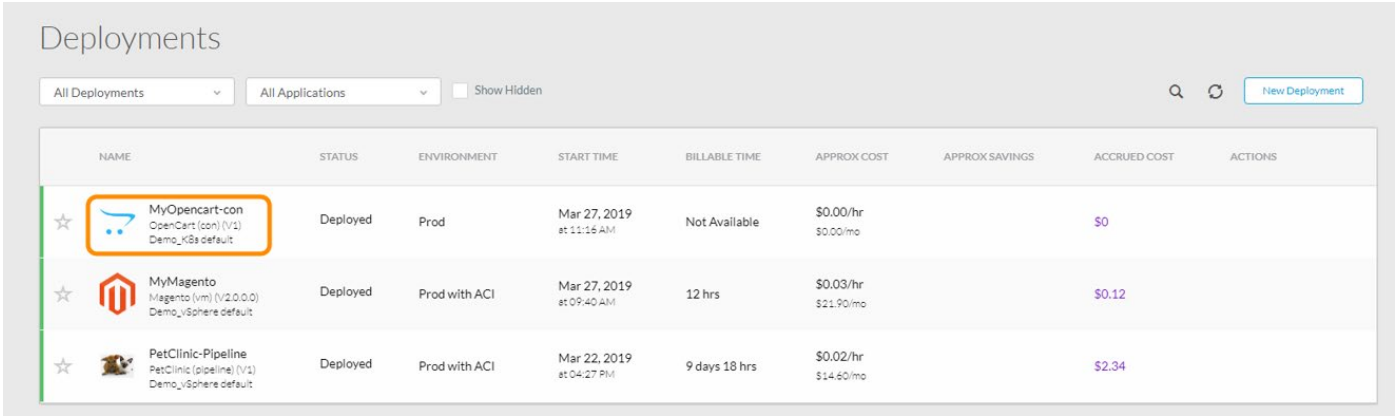
11. Wait for the application to fully deploy and proceed with the verification steps below. To verify the application using the Access Application button, skip to Step 3 in the Verify Application Deployment section

NOTE: If you pre-configured an application prior to the demonstration, you can proceed immediately with the following steps. If not, it may take up to 5 minutes for the application to fully deploy and show the **Access Application** button.

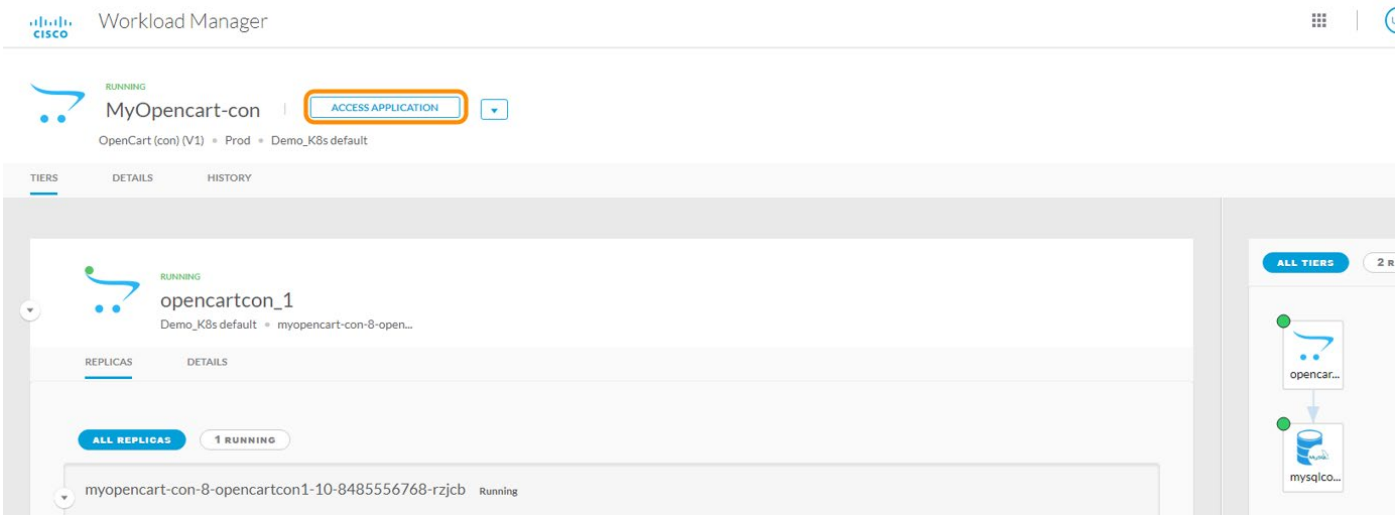
Verify Application Deployment

1. On the left menu, click **Deployments**.
2. Click **MyOpencart-con** on the previously deployed application (the rest of this scenario is based on MyOpencart-con).

NOTE: Wait until the applications Status is Deployed before continuing.

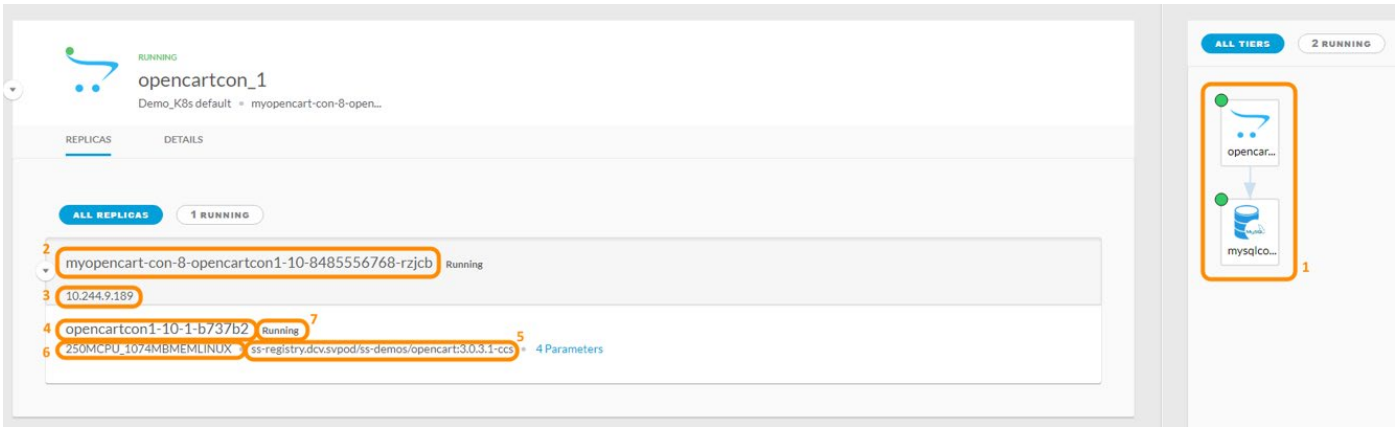


3. On the resulting screen, click the **Access Application** button to proceed to the application



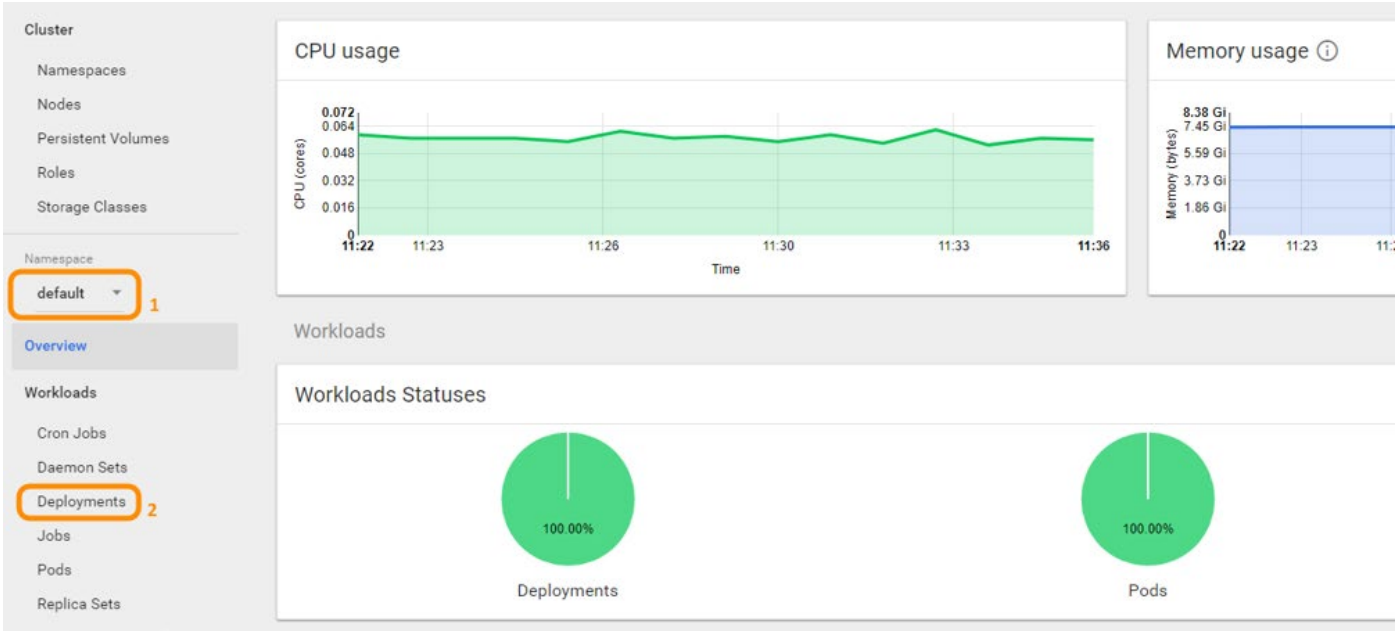
4. When the **Opencart** page is displayed in the Chrome tab, close the tab. (Opencart configuration is outside the scope of this demonstration.)

5. Return to the **Cisco CloudCenter** tab and observe the green circles indicating the health of the deployed services . Then observe the allocated container names, IP Addresses, deployment names, and source container registry . Finally, observe the container resource requests and current status .

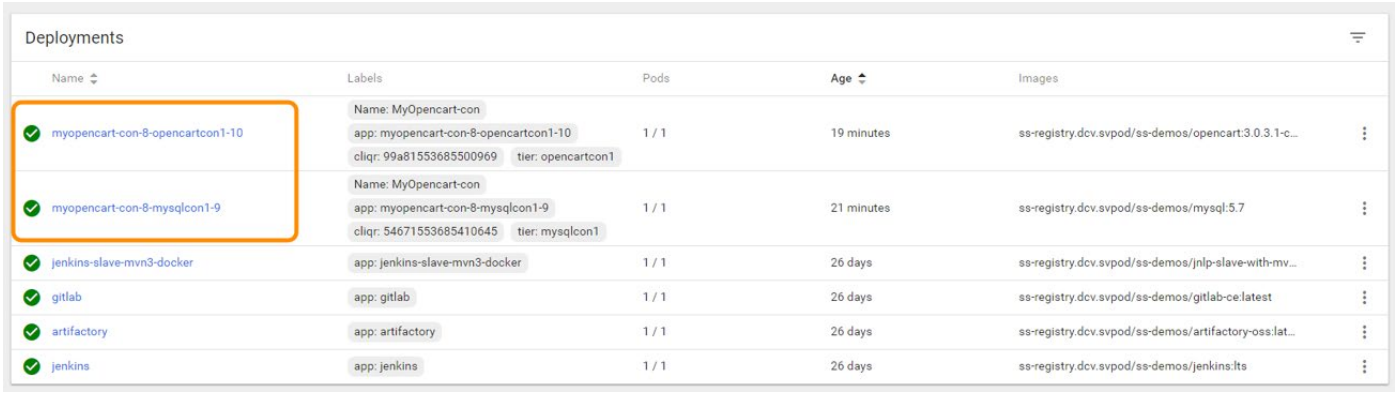


6. Note down the container names used for this deployment and switch to the **Kubernetes Dashboard** browser tab.

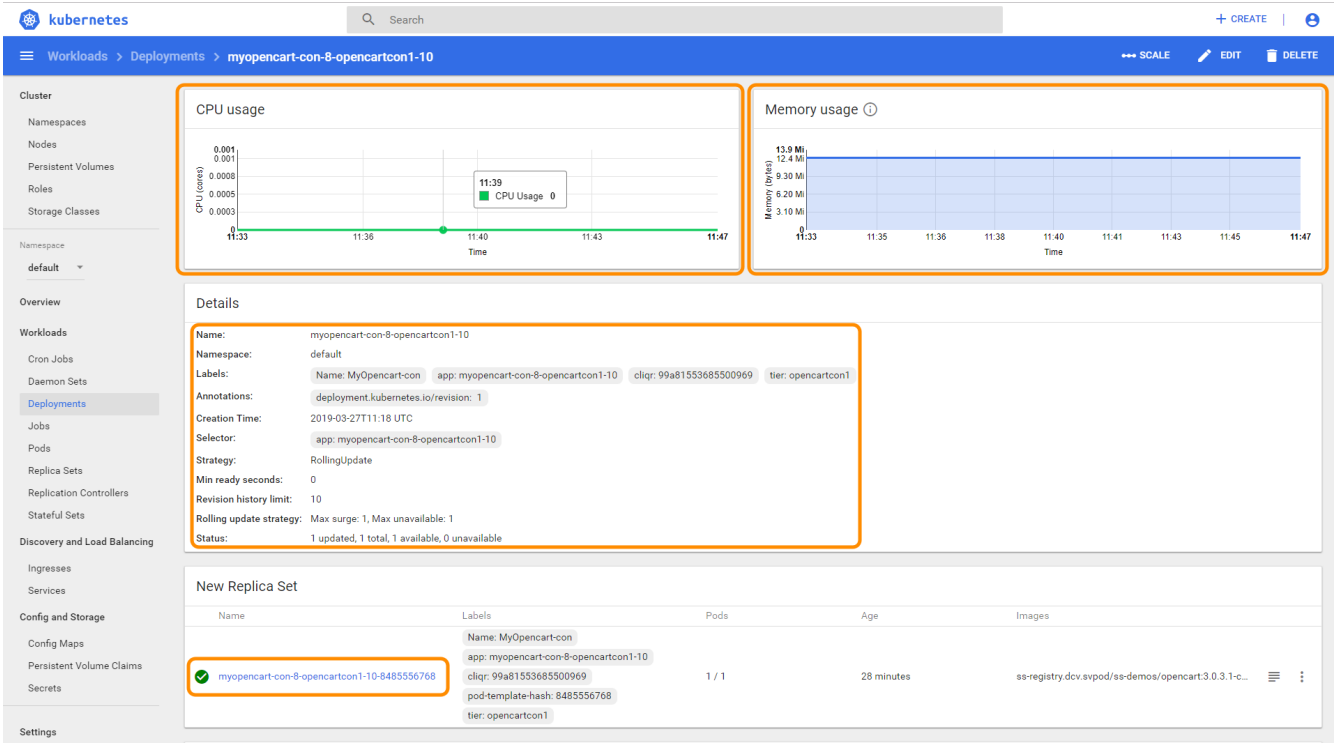
7. Ensure the select **Namespace** is **Default** and click on **Deployments**.



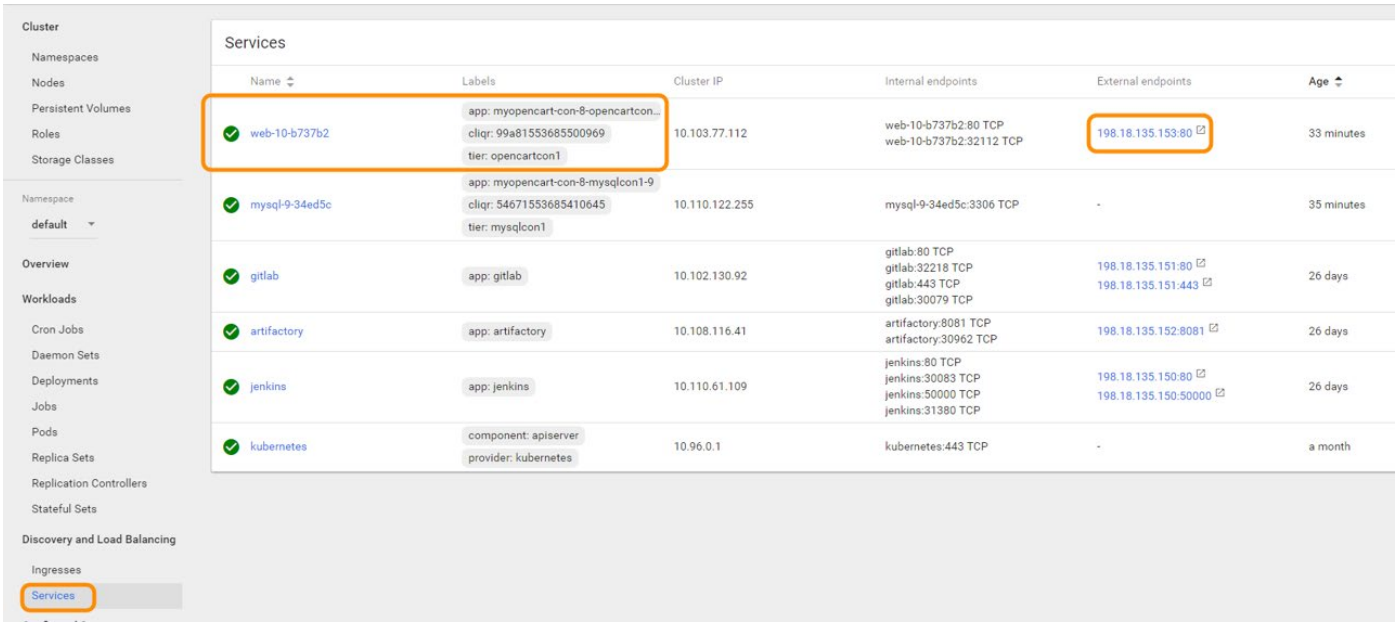
8. Observe the two new deployments present on the list, the **myopencart-con-XX-opencartcon1-YY** and **myopencart-con-XX-mysqlcon1-ZZ** (the XX, YY and ZZ numbers can vary).



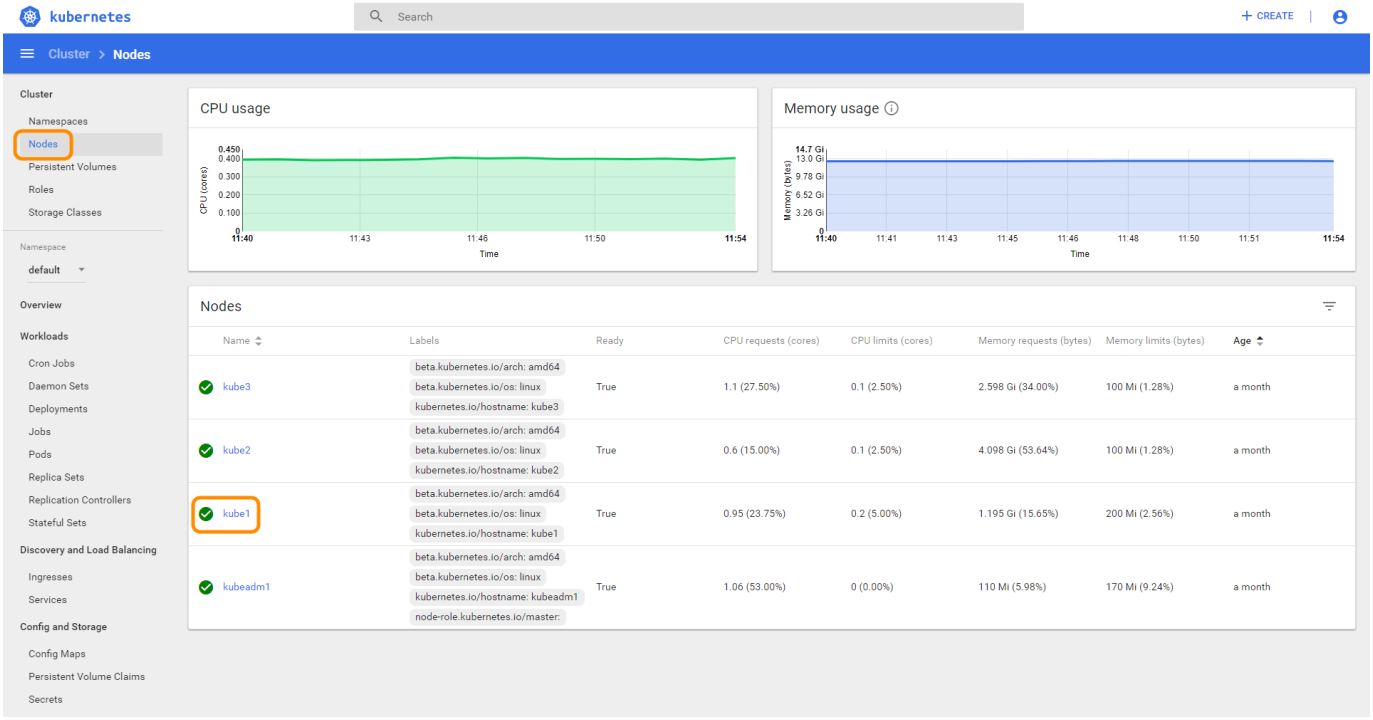
9. To obtain container-level information, **click on one of the deployments**, e.g. myopencart-con-XX-opencartcon1-YY. Observe the current CPU and Memory usage for the container, as well as the configured settings (Labels, Annotations, Strategy, etc.)

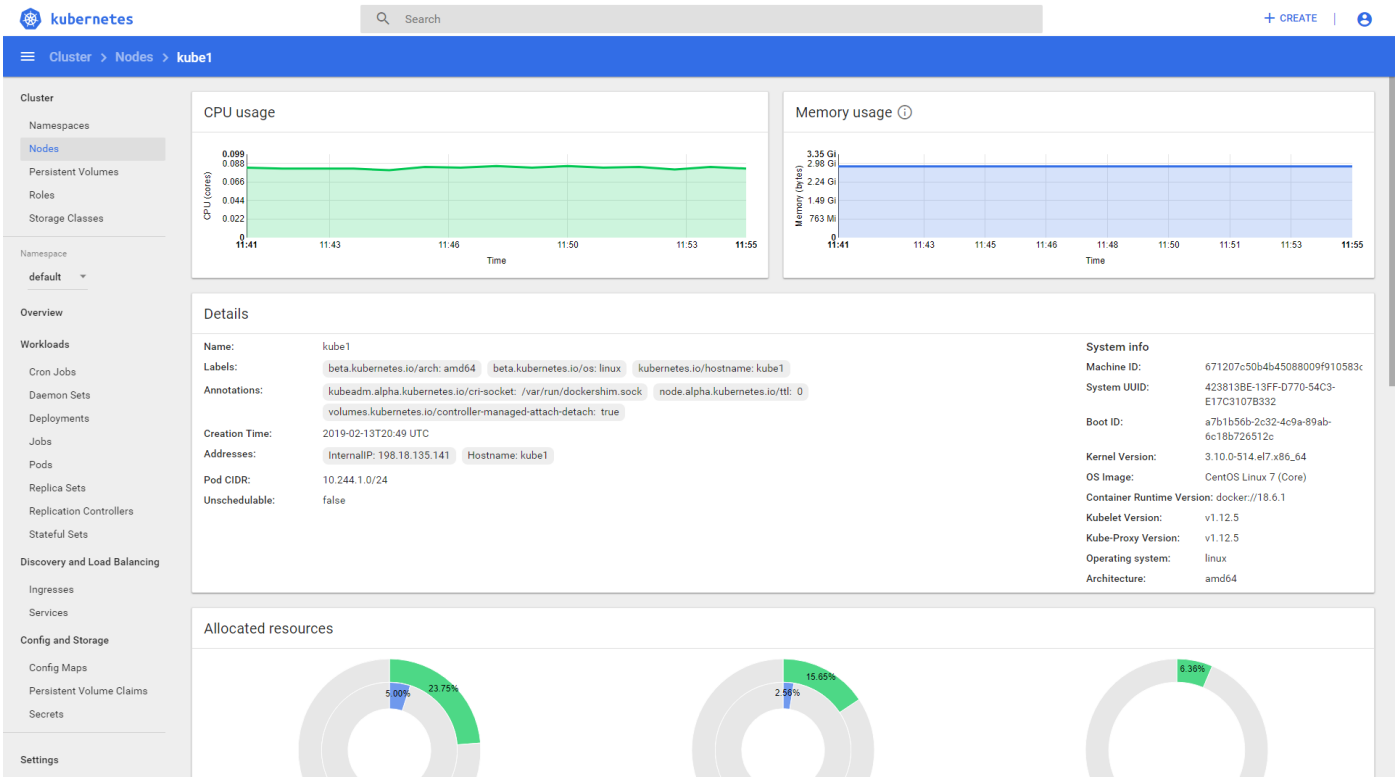


10. To review the allcated service IP to the **MyOpencart-con container deployment**, click on **Services**. To locate the service that links to the deployment, find the service that contains the labels app:myopencart-con-XX-opencartcon1-YY and tier:opencartcon1. The External endpoints column should list the same IP address as the **Access Application** in the **CloudCenter Deployment**.

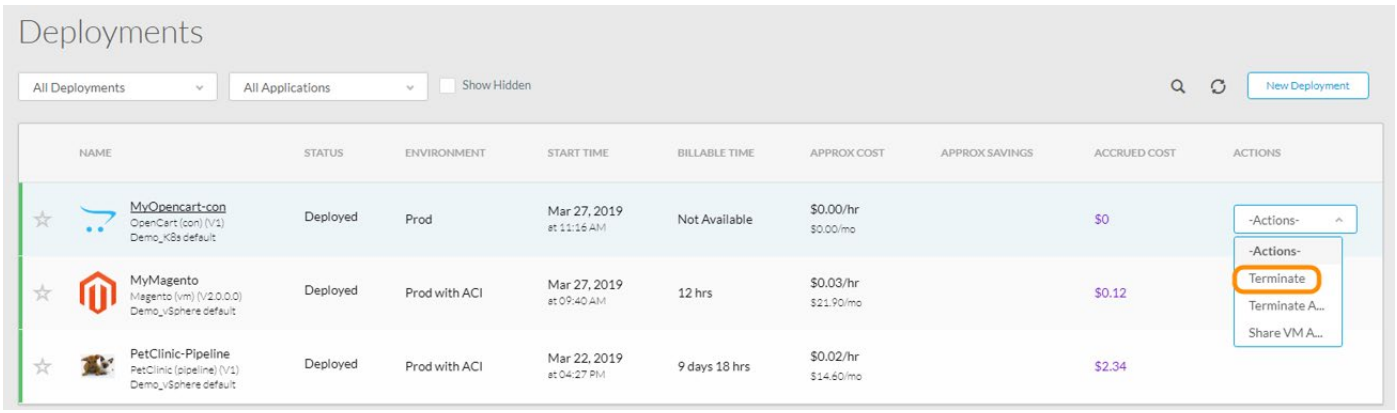


11. To observe the overall cluster CPU and Memory usage, click on **Nodes**. To drill down on the resource usage per node, click on each node name (e.g. kube1)





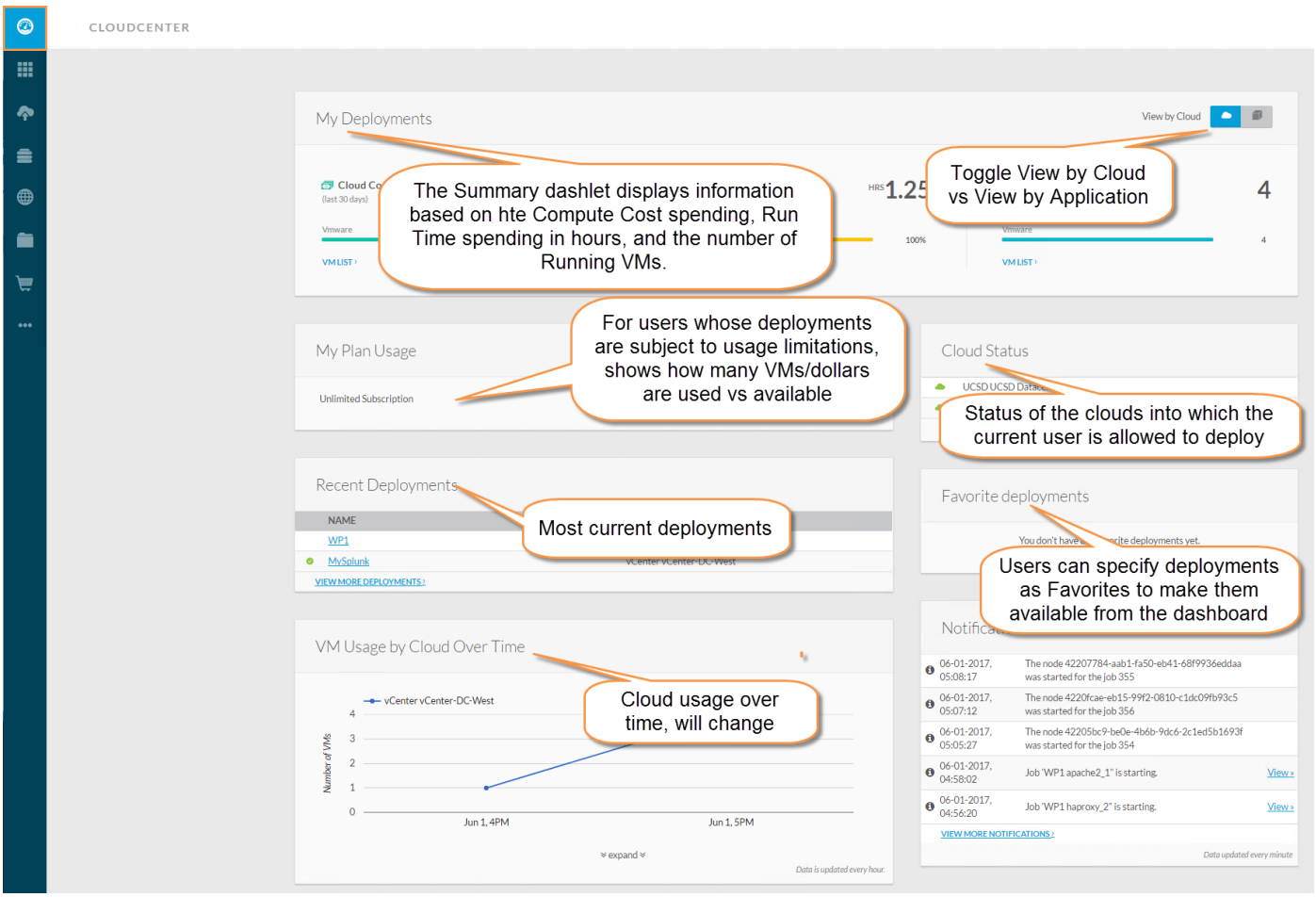
12. Optionally, return to the CloudCenter deployments and terminate the MyOpencart-con deployment (it may take a few minutes for the deployment to be terminated). Then switch back to the Kubernetes Dashboard web page and observe the deployment, pod and service objects related to the CloudCenter deployment having been removed.



Dashboard Overview

The purpose of this section is to review the dashboard now that more than one application is deployed.

1. Return to CloudCenter and click **Dashboard** on the side menu. If the Magento application is not reflected in the graphs yet, Refresh until it is. It may take up to ten minutes until the Magento application is fully deployed and displayed on the Dashboard.
2. Review and explain the graphs.

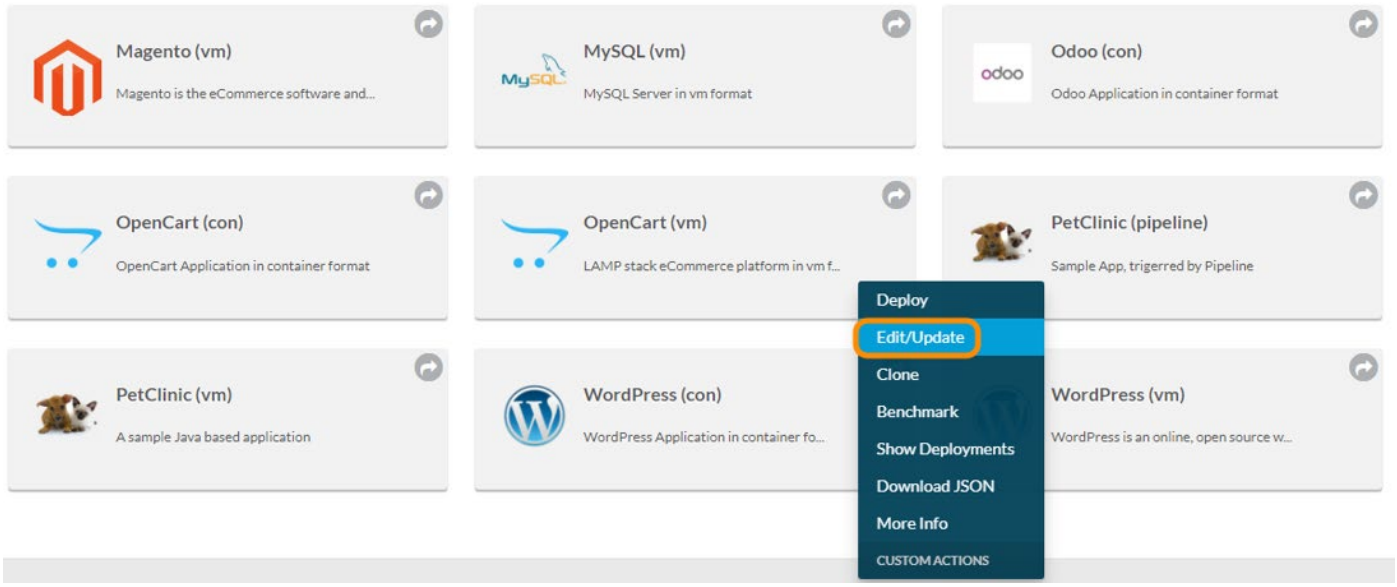


Scenario 2. Architect Role – Editing Application Profiles

The purpose of this section is to use the Edit/Update option to walk through an existing Application Profile, showing the different sections of an Application Profile and their role, as well as to demonstrate a typical scenario of reconfiguring an application for scalability. The Edit/Update option is only available to Architects/Power Users.

Steps

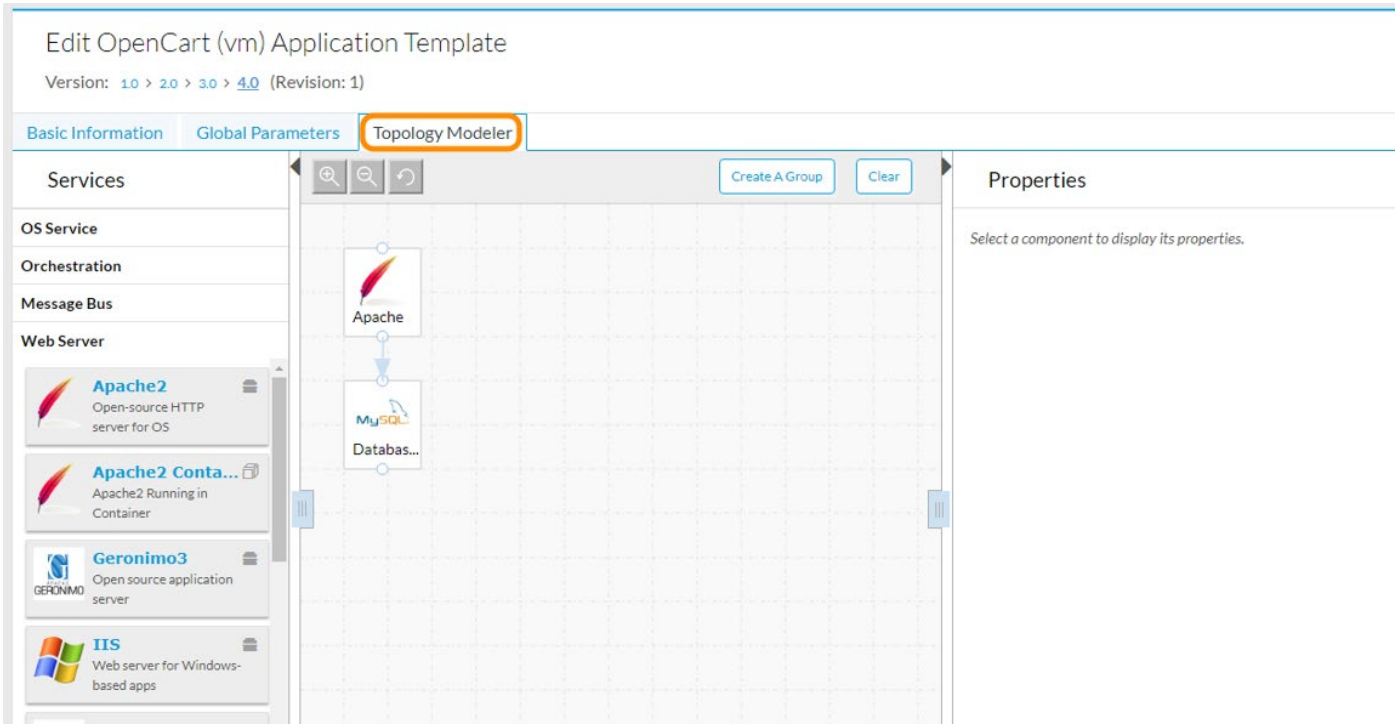
1. Navigate to the **CloudCenter** UI, login as **poweruser@dcloud.cisco.com / C1sco12345**, tenant: **demo**
2. Select **Workload Manager**.
3. Click on **App Profiles**.
4. Mouse over **Opencart (vm)** and choose **Edit/Update.**



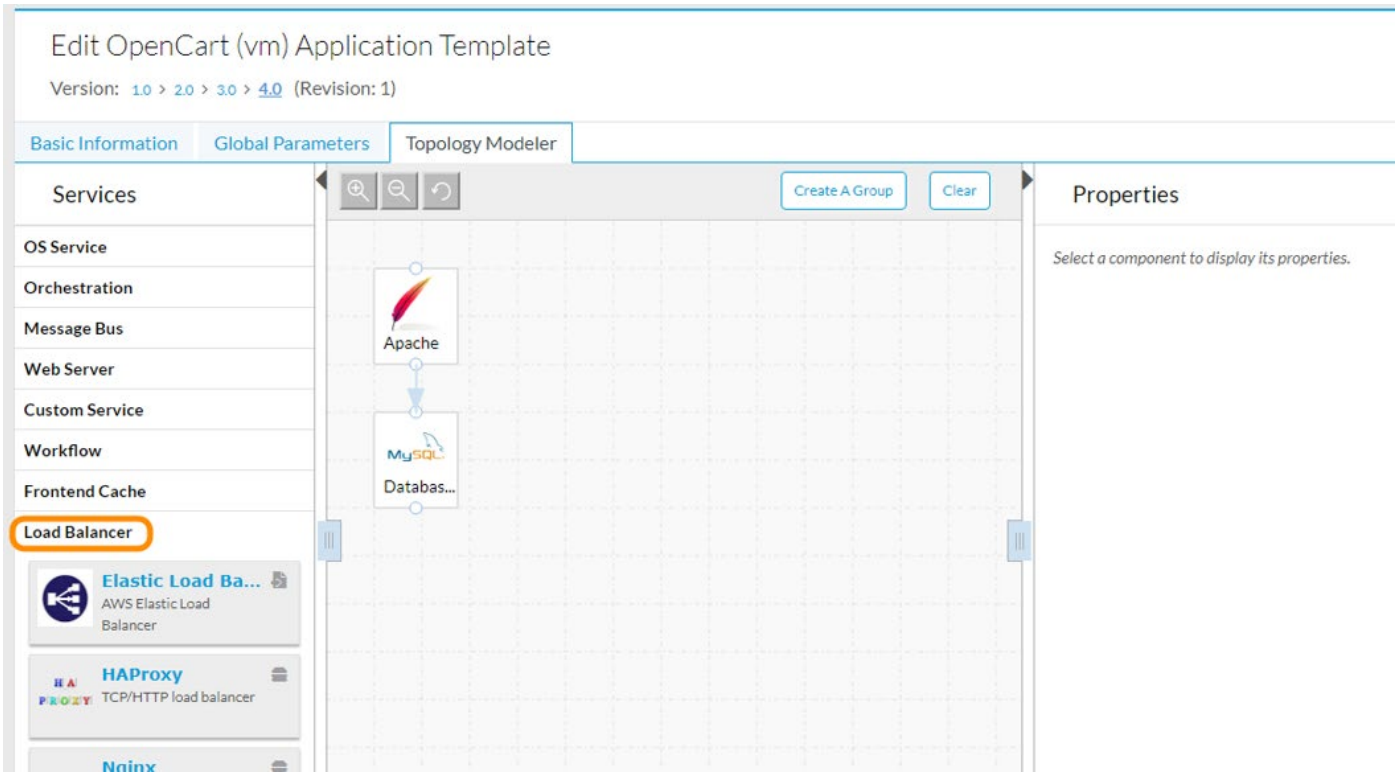
The Edit window is divided into three tabs:

- **Basic Information** – Allows the architect to set parameters such as name, version, application category, tags, URL, and micro segmentation, as well as choose the application icon that will appear with the application in the Application Profile catalog.
- **Global Parameters** – Allows the Architect to configure a list of information that the user will supply during application deployment.
- **Topology Modeler** – Allows the Architect to select a list of services that will be dynamically deployed and configured in when the application profile is deployed.

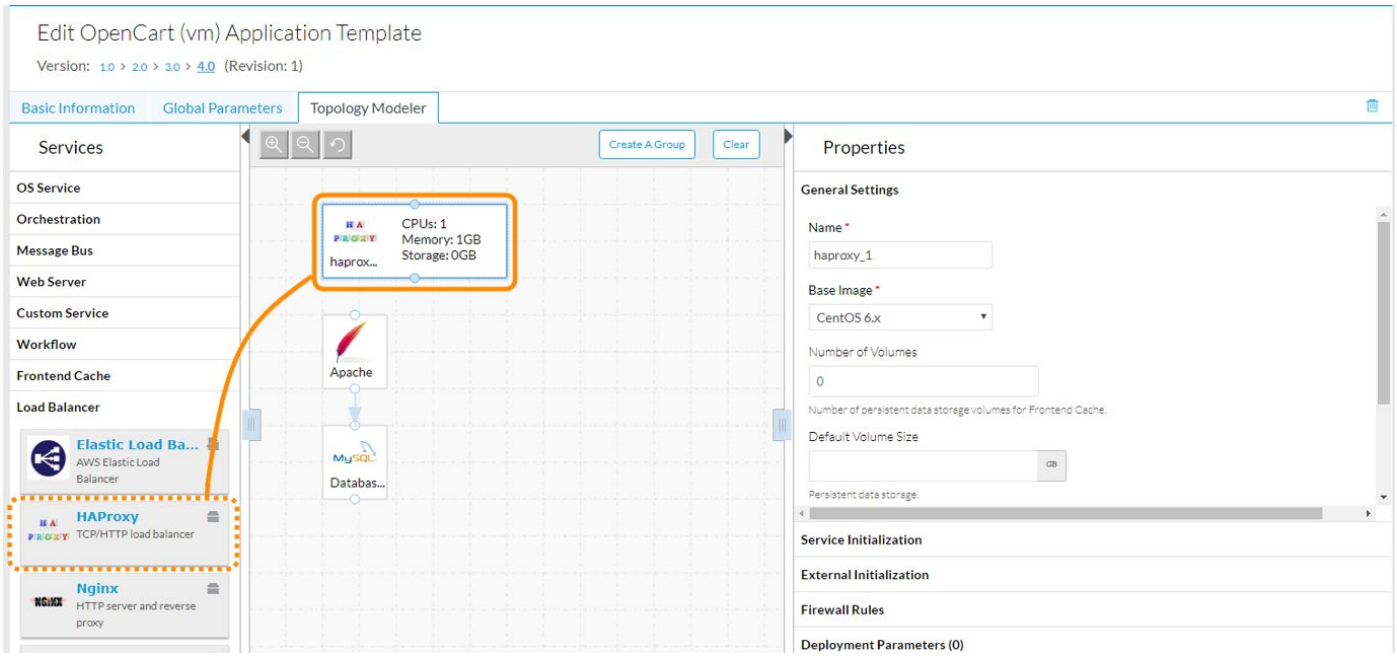
5. Navigate to the **Topology Modeler** tab.



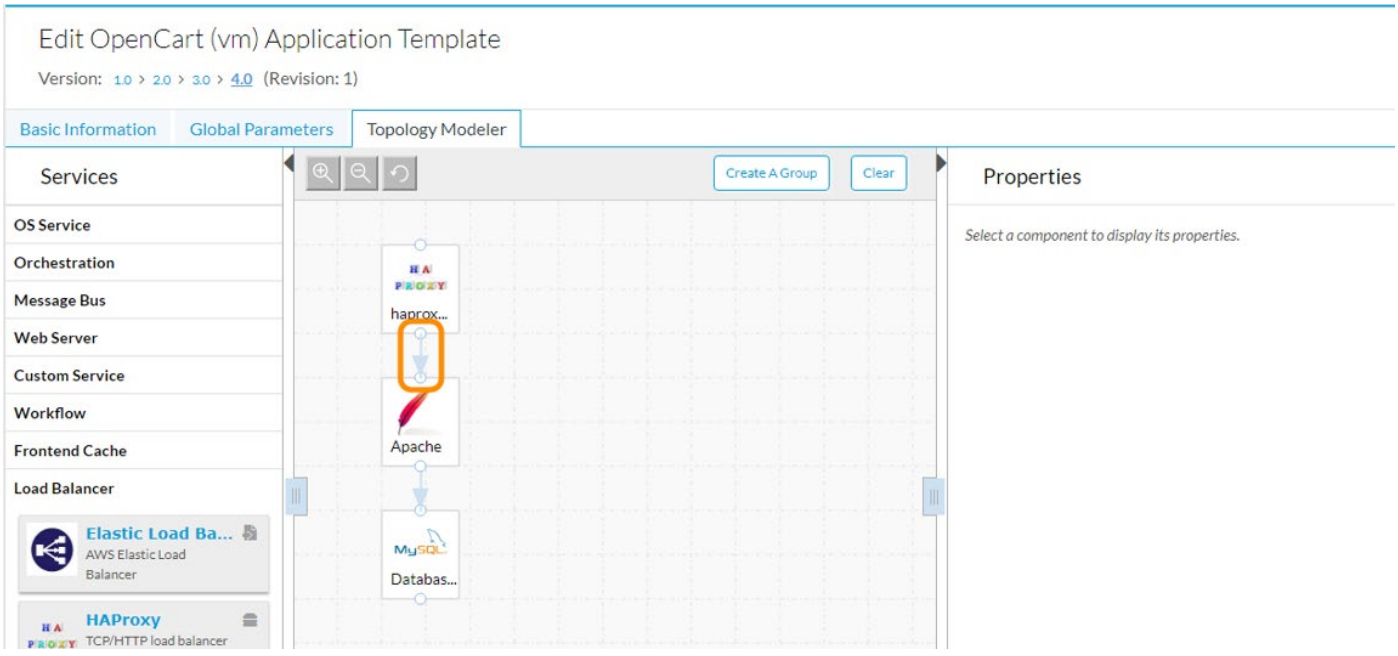
6. At the left-hand side of the form, under **Services**, expand **Load Balancer**.



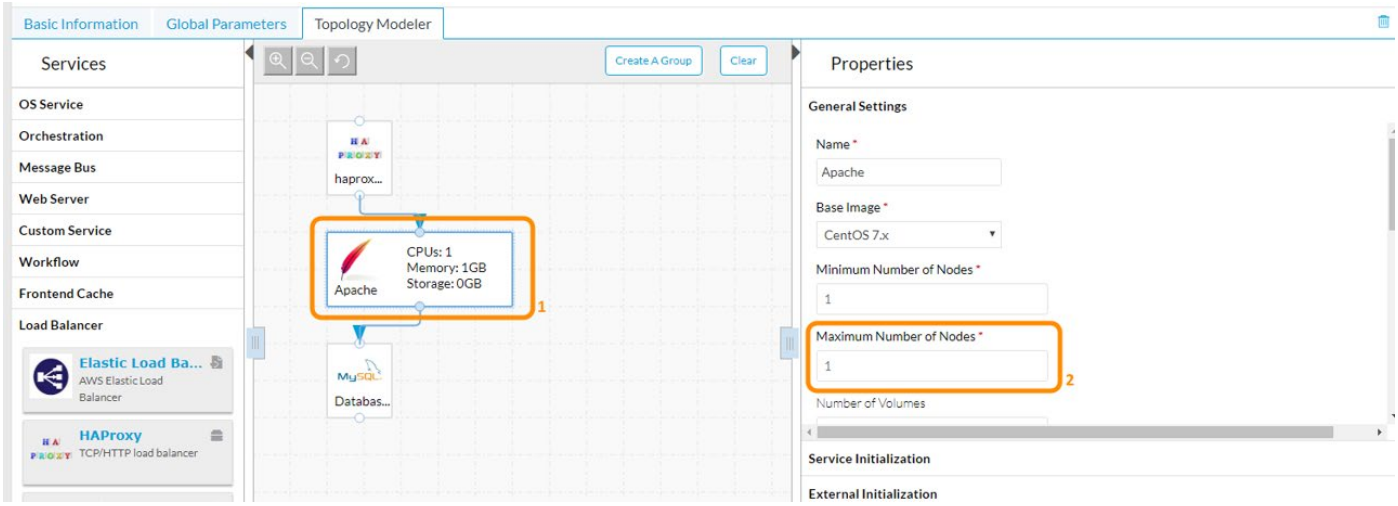
7. Drag and drop **HAProxy** to the **Modeler** section.



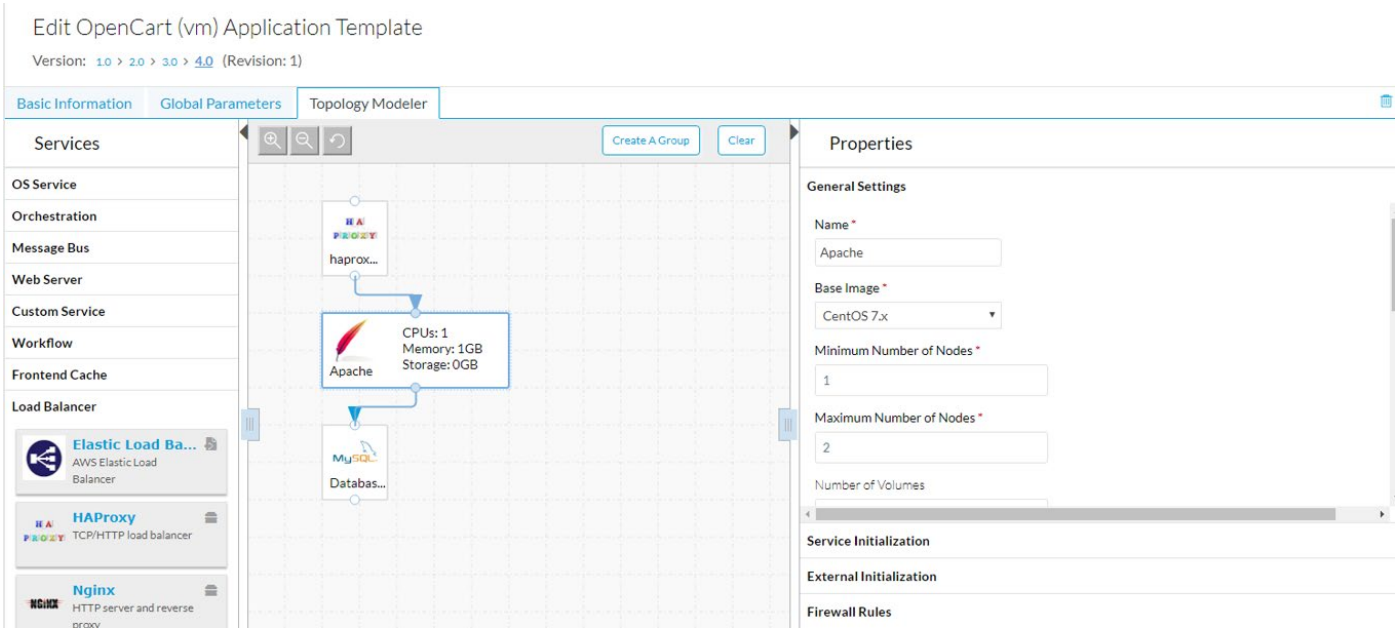
8. Link the **HAProxy service** to the **Apache service** (drag the botton circle of the haproxy service to the top circle of the Apache service, an arrow should form).



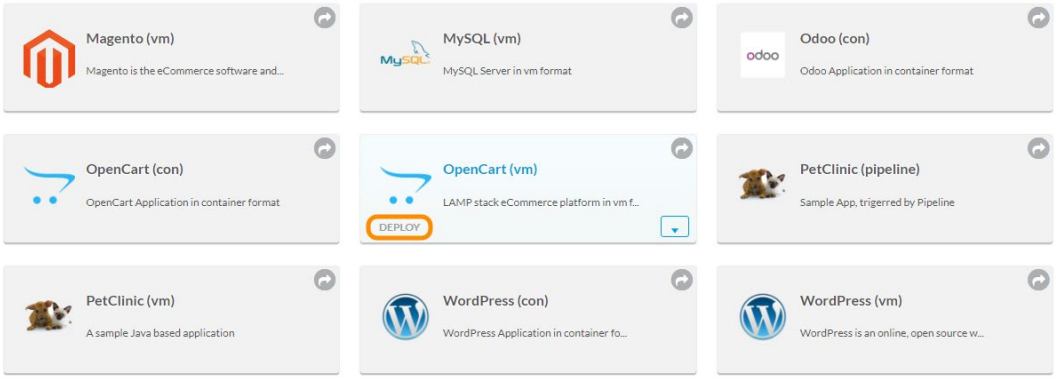
9. Click on the **Apache** service. At the right-hand side, under **Properties**, click in the **Maximum Number of Nodes** field. Change the value from 1 to 2.



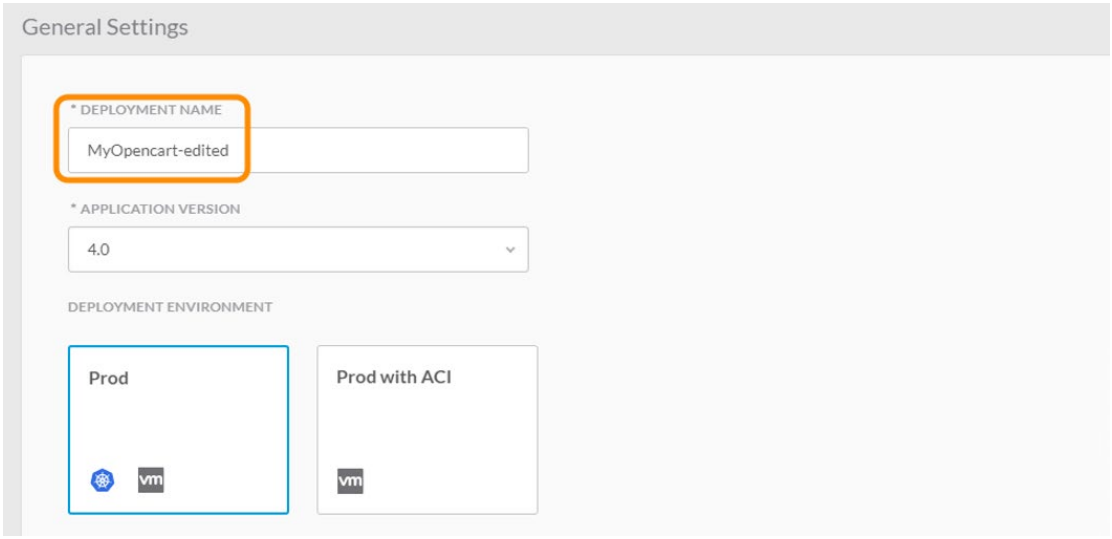
10. Click **Save App**.



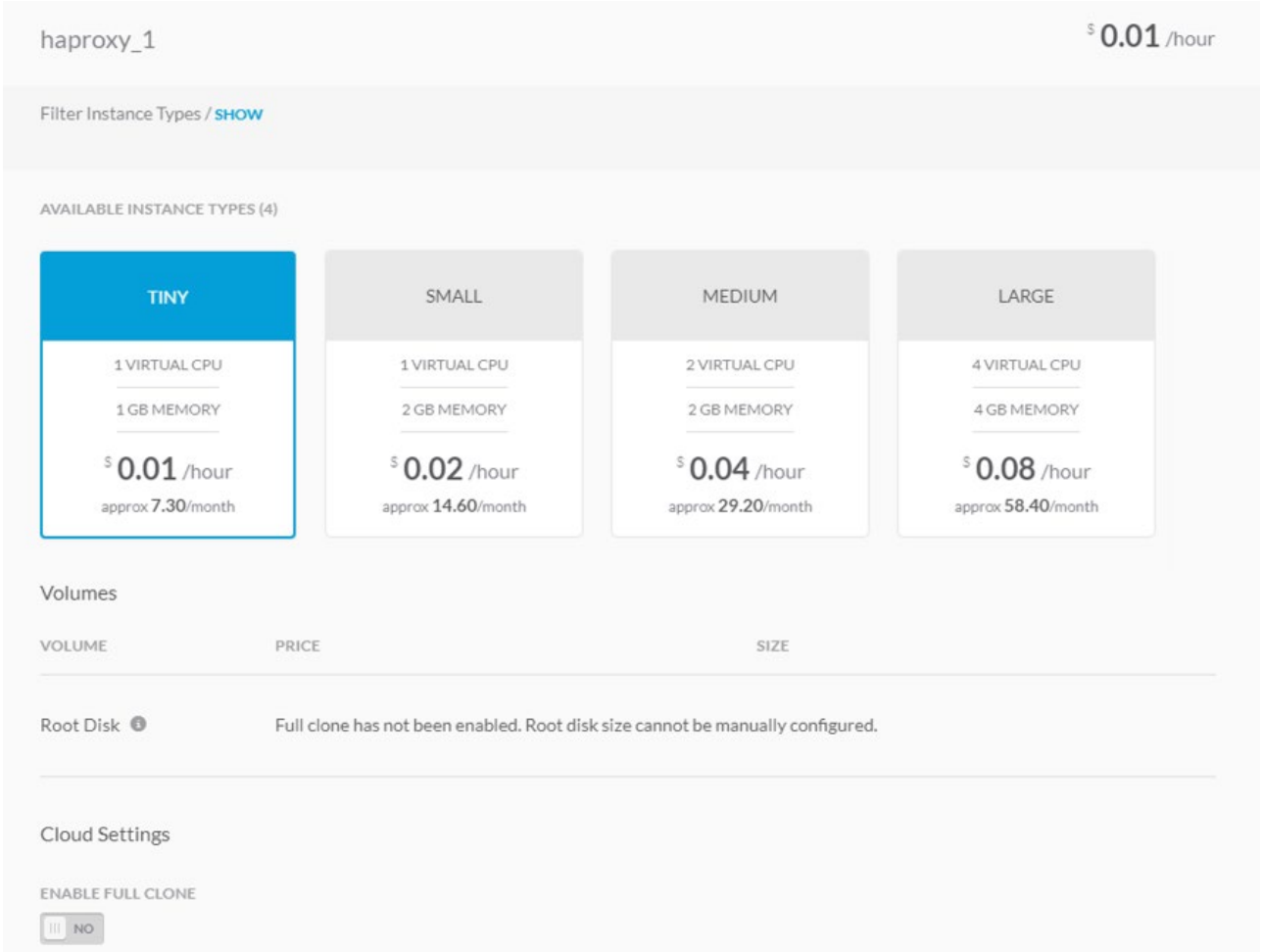
11. Click on **Opencart (VM)** to deploy a new application.



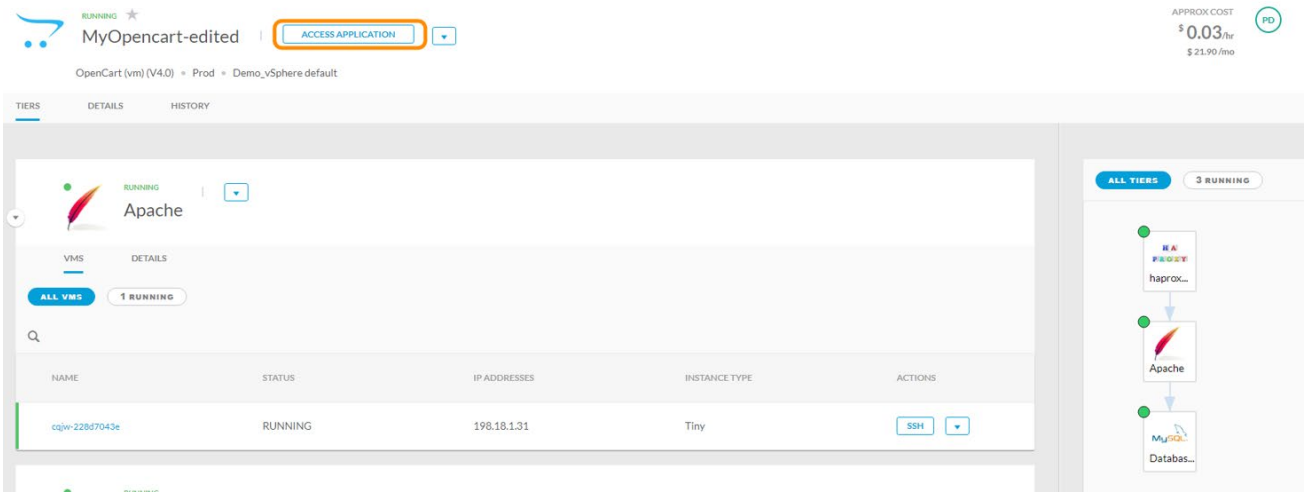
12. Enter **MyOpencart-edited** at the Deployment Name field.



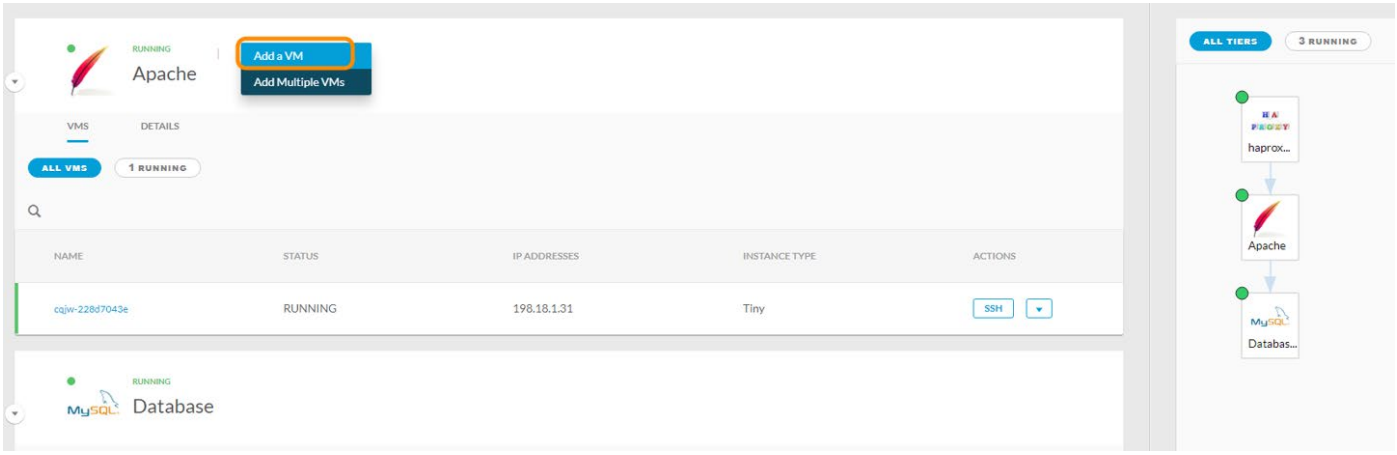
13. Leave the other fields to their default values and click **Next**.
14. Leave all the fields at their default values and click **Deploy**.



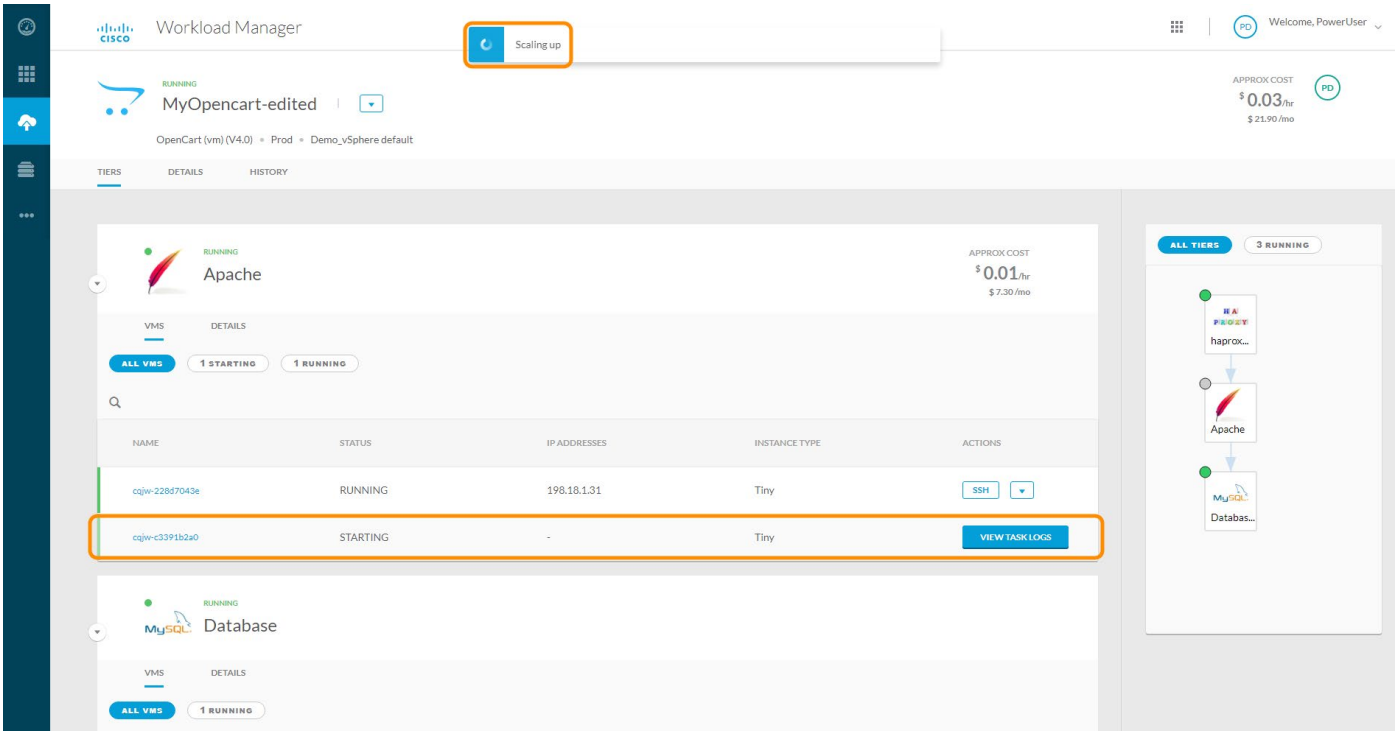
15. Once the application is deployed, to verify its proper operation, click on the **Access Application** at the top of the form to verify the application is working properly. Close the application browser tab and return to the CloudCenter tab.



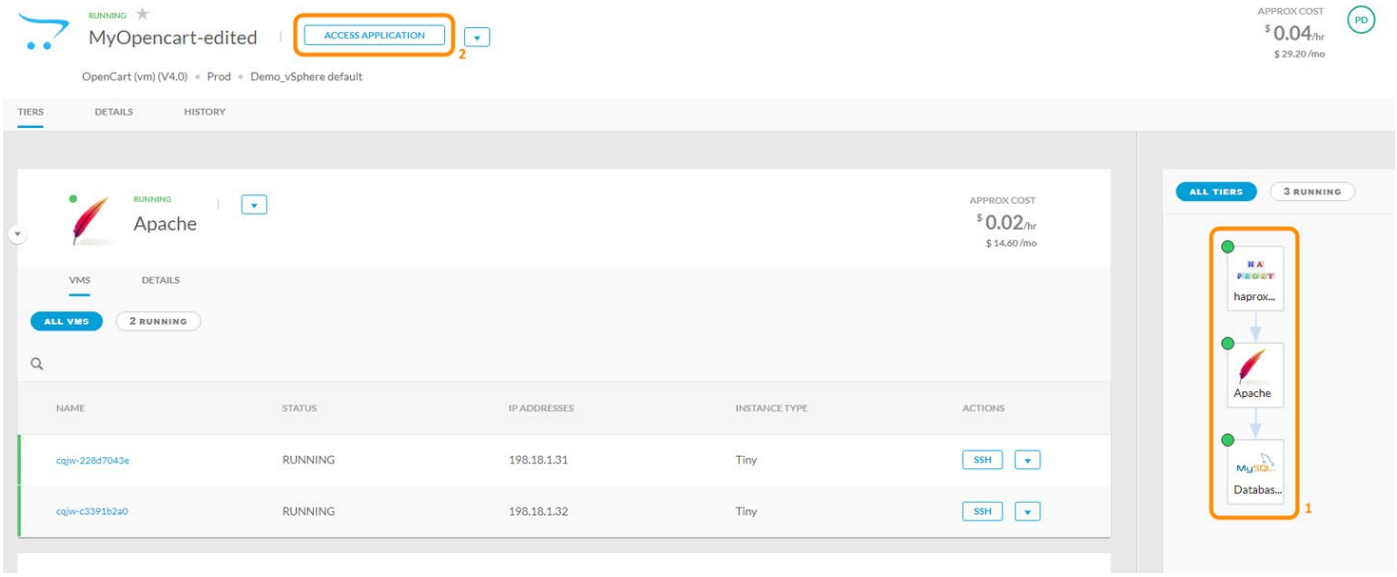
16. Locate the **Apache tier** and click at the down arrow next to it. Select **Add a VM.**



17. The status of the deployment should change to Scaling Up. Wait for the deployment to automatically deploy the new VM and configure it, followed by reconfiguring the higher tier (haproxy) to accommodate for the change.



18. Once the deployment has finished adding and reconfiguring the new resources, click **Access Application** at the top of the form to verify the Application is working properly.



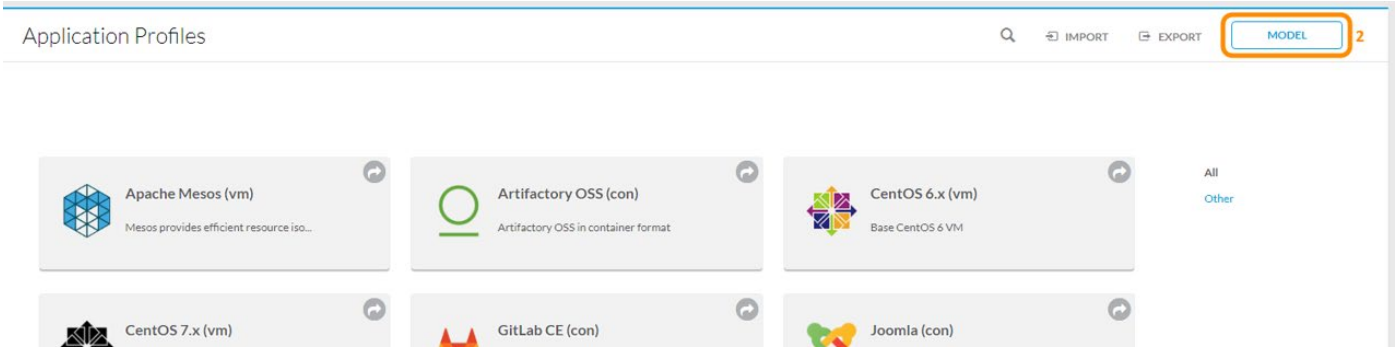
Scenario 3. Admin Role – Create Constructs Within CloudCenter

The purpose of this section is to demonstrate some of the functions of the Admin role:

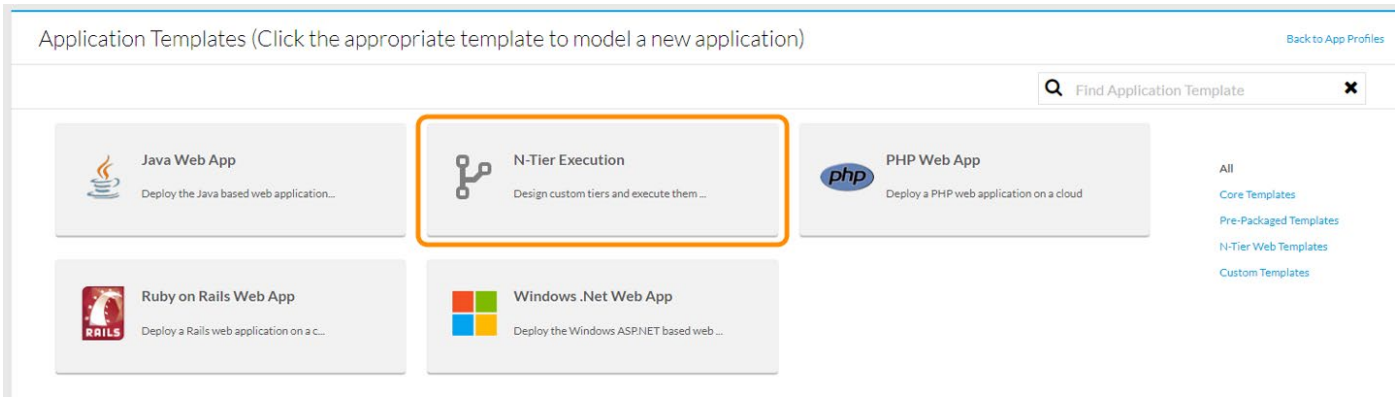
- Creating a VM-Based Application Profile
- Creating a Container-Based Application Profile
- Creating a Usage Plan
- Creating a Bundle

Creating a VM-Based Application Profile

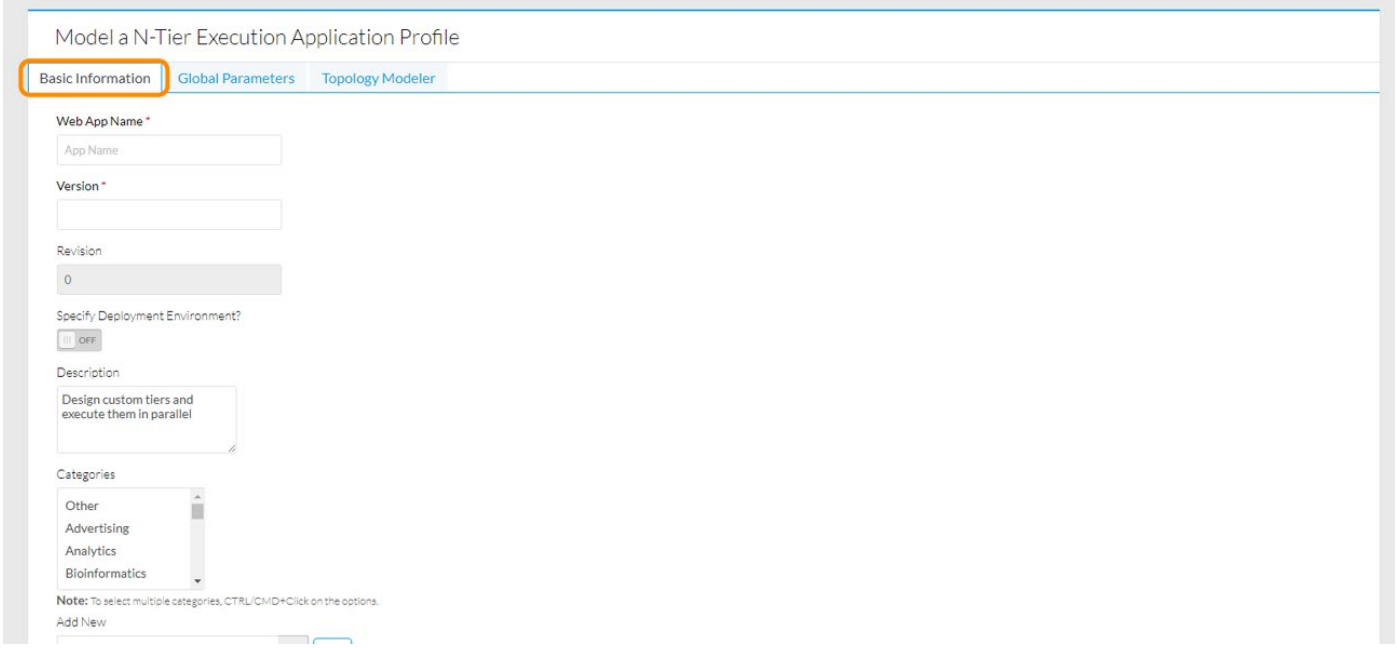
1. Navigate to the CloudCenter UI, login as **admin@dcloud.cisco.com / C1sco12345,** tenant: **demo**.
2. Select **Workload Manager**.
3. Click on **App Profiles**, then click on **MODEL**.



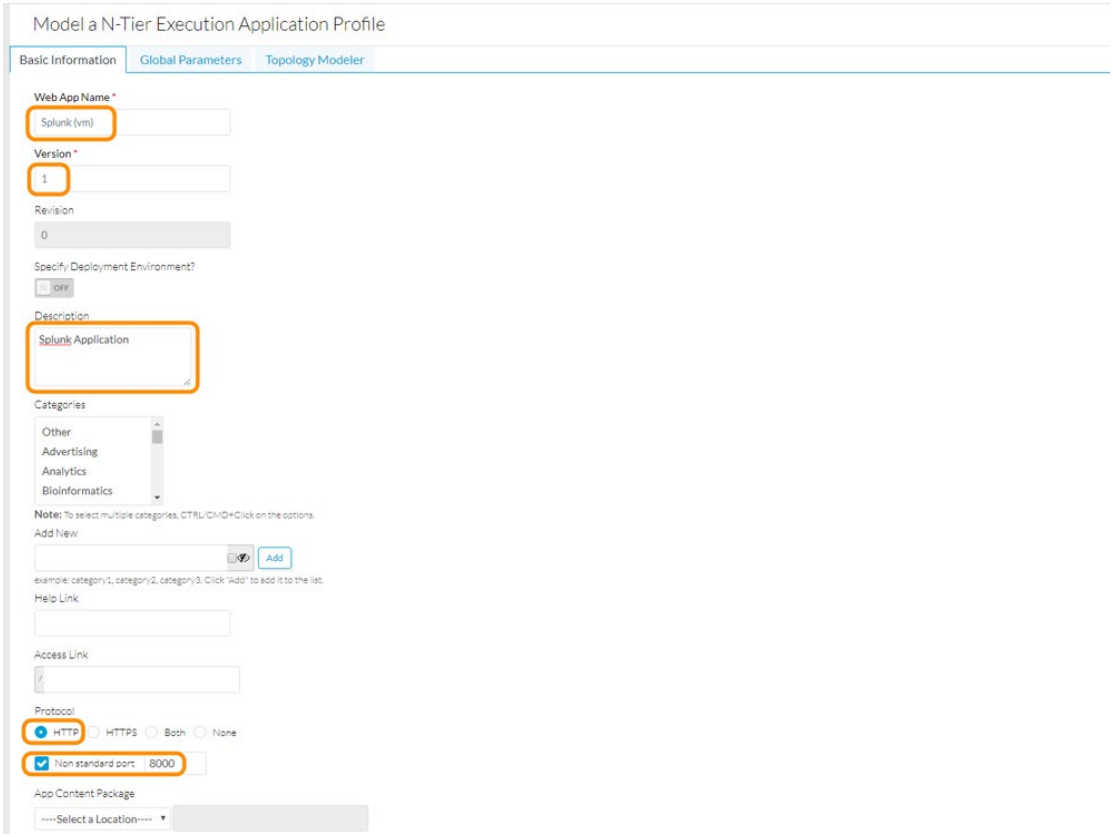
4. At the **Application Templates**, click on **N-Tier Execution.**



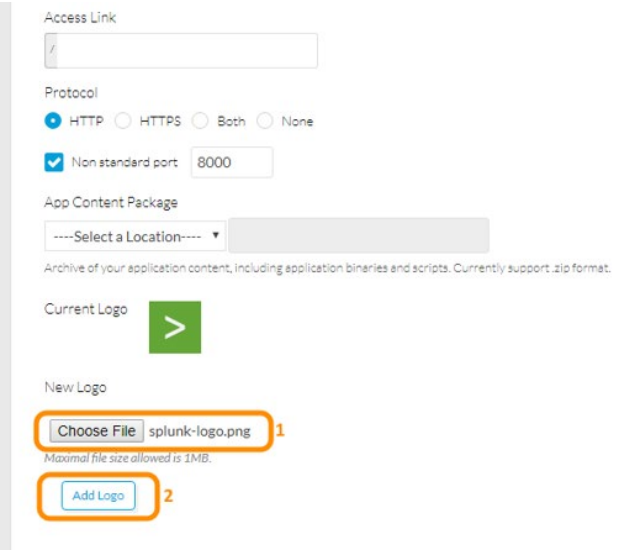
5. Click on the **Basic Information** tab.



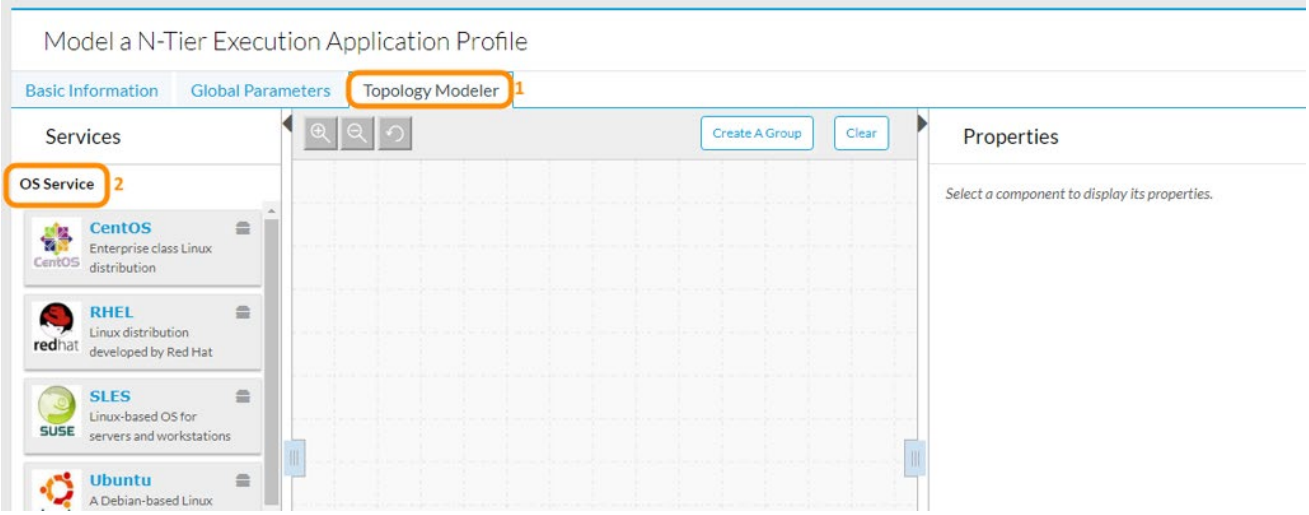
6. Enter **Splunk (vm)** at the **Web App Name** field, **1** at the **Version** field and **Splunk Application** at the **Description**. Then, at the Protocol Section **ensure HTTP is selected** and **enable Non standard port**. Enter the value **8000** as the non-standard port.



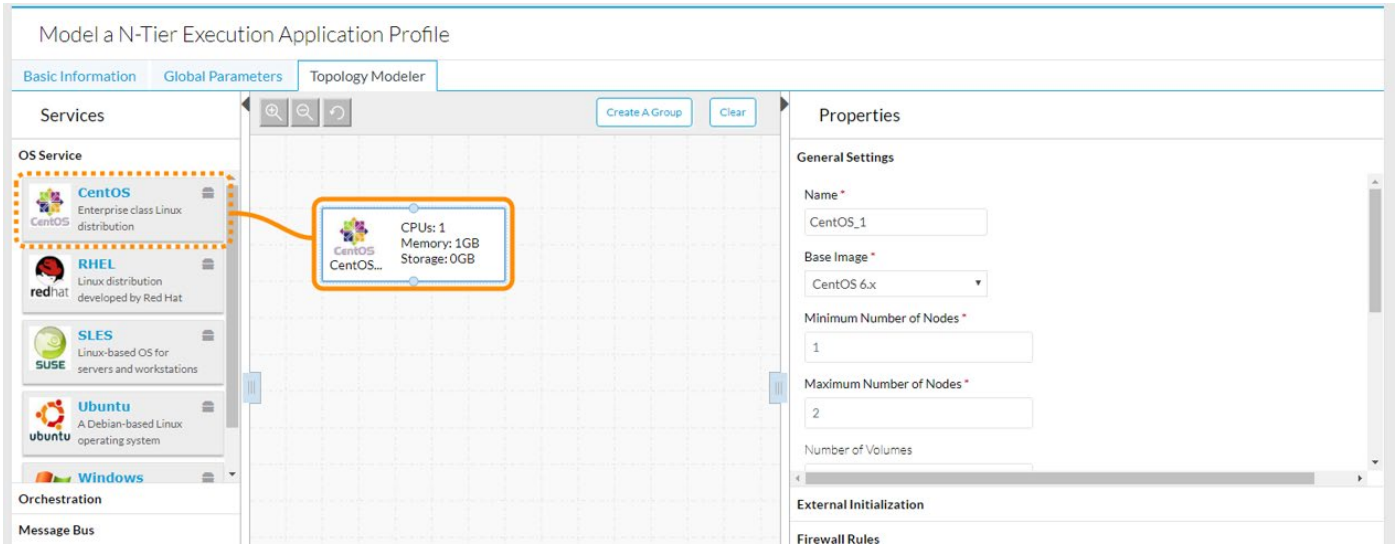
7. Scroll down and locate the **New Logo** section. Click on **Choose File**. Navigate to **Pictures** and select the splunk-logo.png file, then click **Open**. Next, click **Add Logo**. The logo should show up.



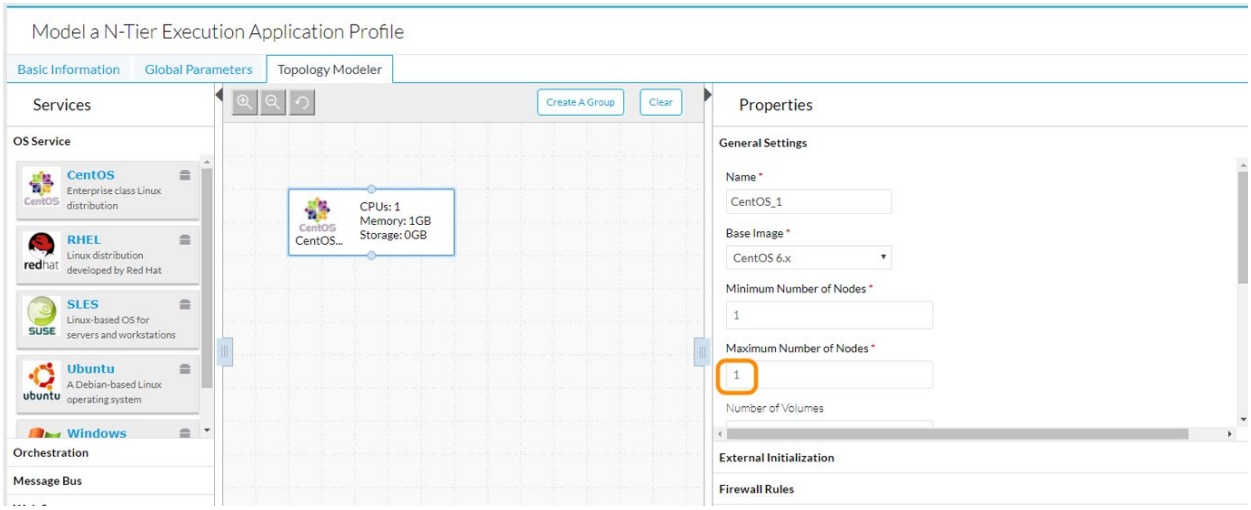
8. Next, navigate to the **Topology Modeler** at the top of the form. At the **Services** section on the left-hand side, expand the OS Service section.



9. Locate the **CentOS service** and drag and drop it to the modeler.

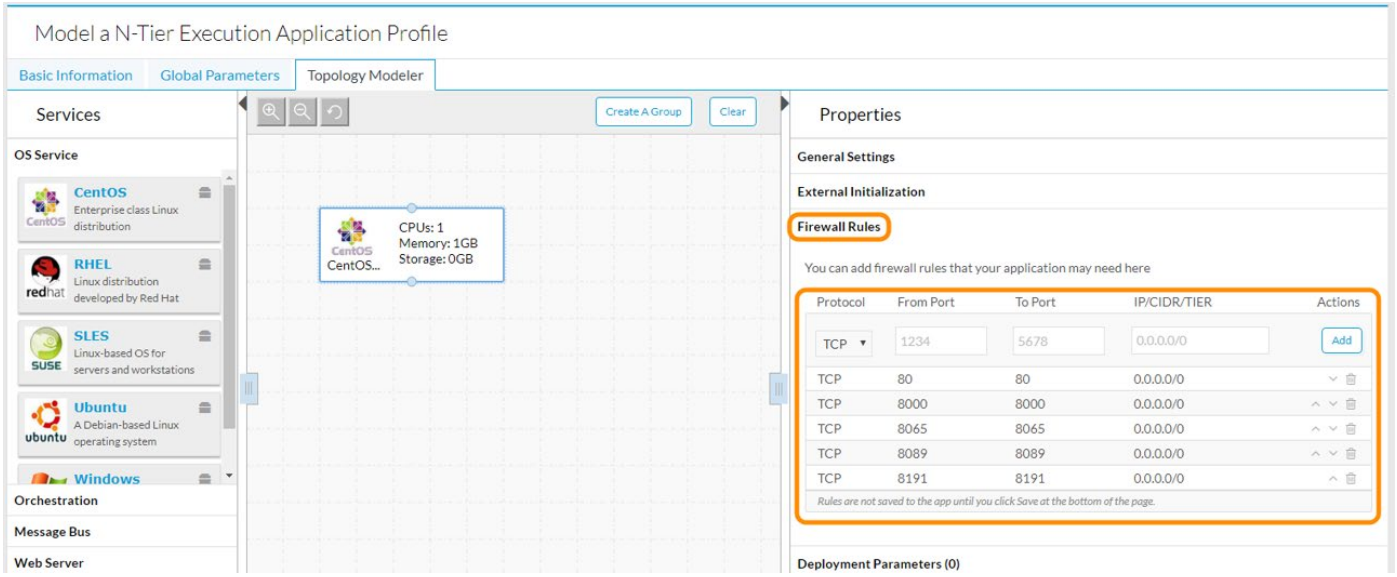


10. Click on the new object and at the Properties section on the right-hand side, under **General Settings**, change the value of Maximum Number of Replicas from 2 to 1.

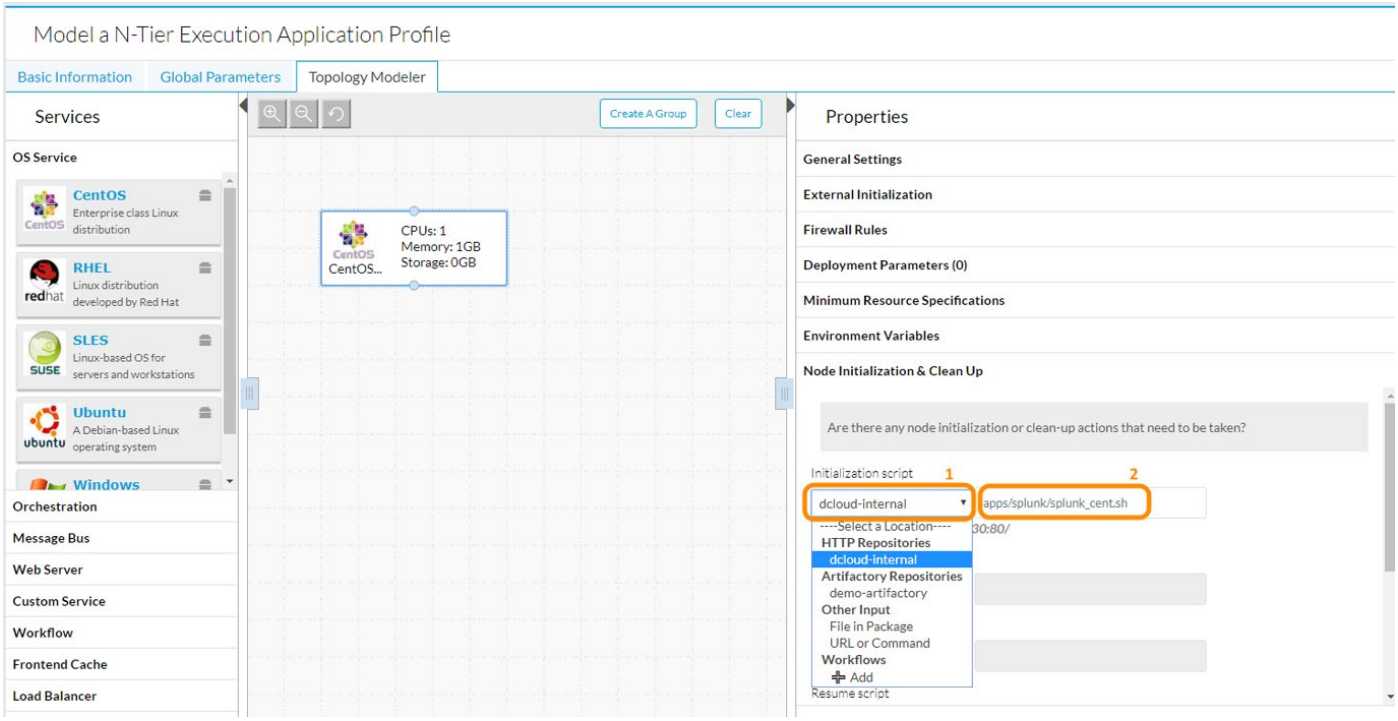


11. At the **Firewall Rules Section**, create the following records:

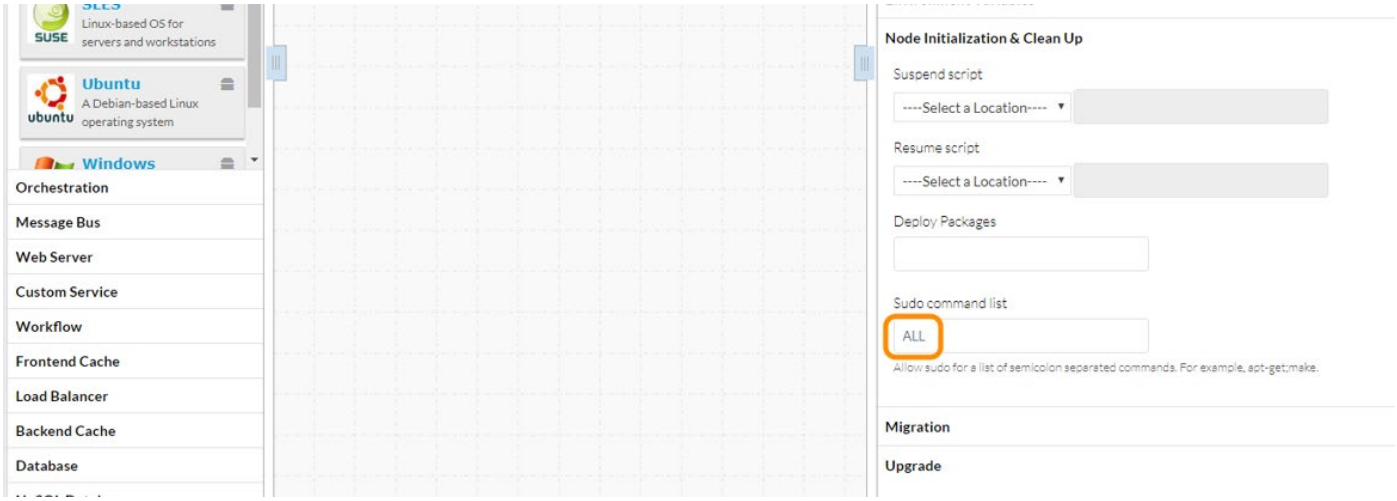
Protocol	For Port	To Port	IP/CIDR/TIER
TCP	80	80	0.0.0.0/0
TCP	8000	8000	0.0.0.0/0
TCP	8065	8065	0.0.0.0/0
TCP	8089	8089	0.0.0.0/0
TCP	8191	8191	0.0.0.0/0



12. At the **Node Initialization & Clean Up** section, at the Initialization Script, choose dcloud-internal and then type in apps/splunk/splunk_cent.sh to use a pre-configured script that configures Splunk.

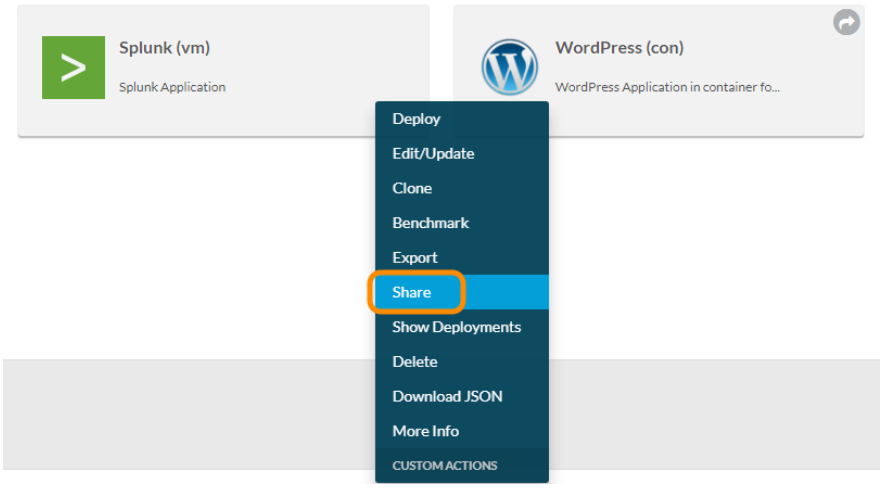


13. In the same section, at the Sudo command list text box, enter the value **ALL** (all capitals). This is to ensure the script has adequate permissions to perform admin-level tasks during the deployment.

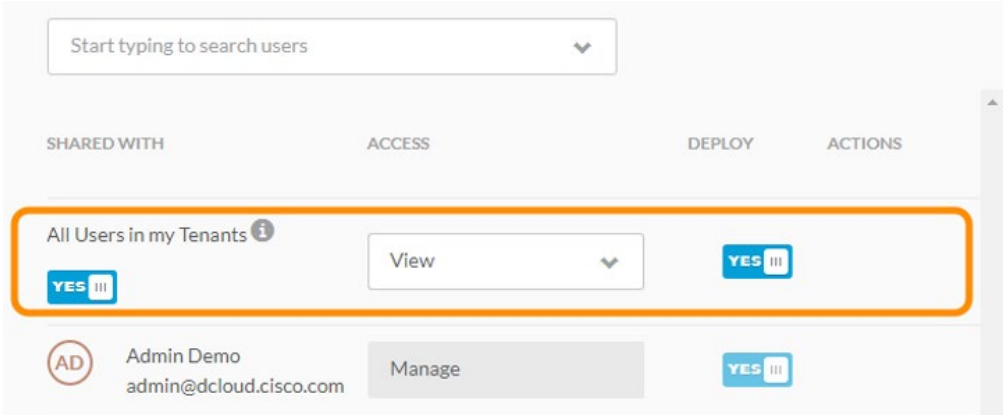


14. Click **Save As App** at the bottom of the form to save the new App Profile.

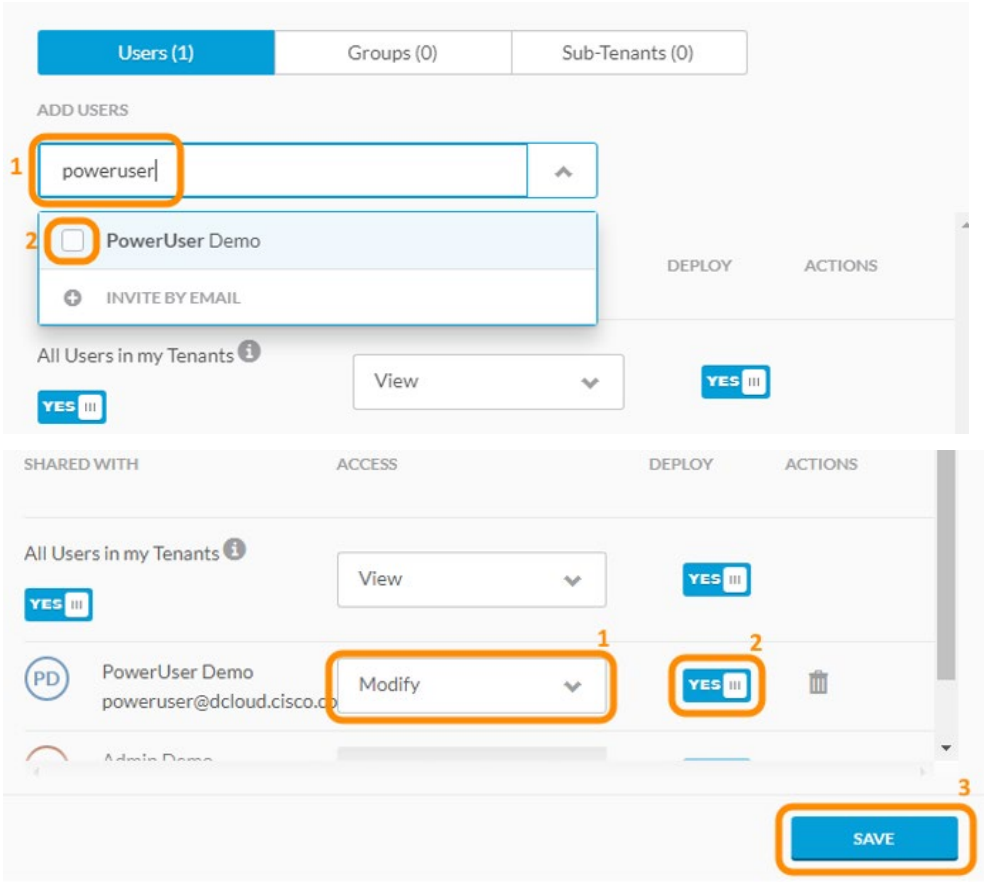
15. To Share this App Profile so other users can use it, click on the icon and select **Share**.



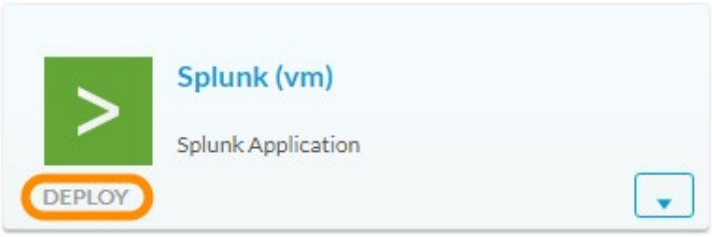
16. At the next form, under **All Users in my Tenants**, set the switch from **No** to **Yes**, leave the **Access to View** and under Deploy set the switch from **No** to **Yes**.



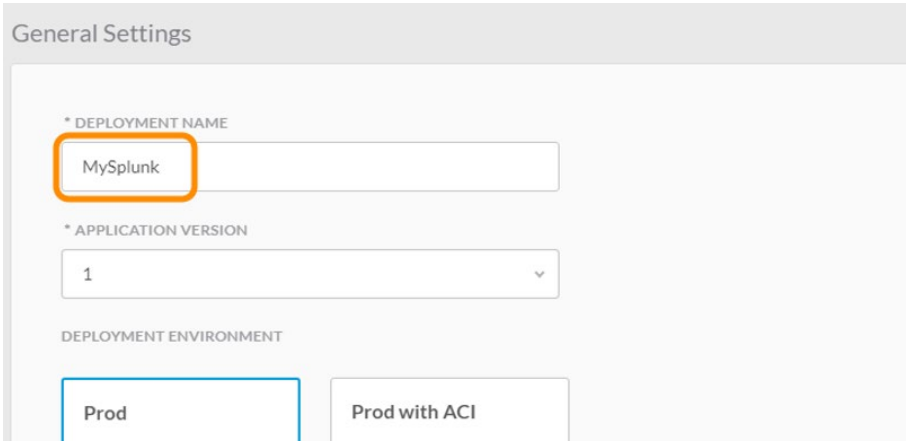
17. Next, at the Add Users text box, type **PowerUser Demo** and select the user from the list. Set the Access to **Modify** and the Deploy to **Yes** for this user, then click **Save**.



18. Once back at the App Profiles form, click on the **Splunk (vm) App Profile** to initiate a deployment.

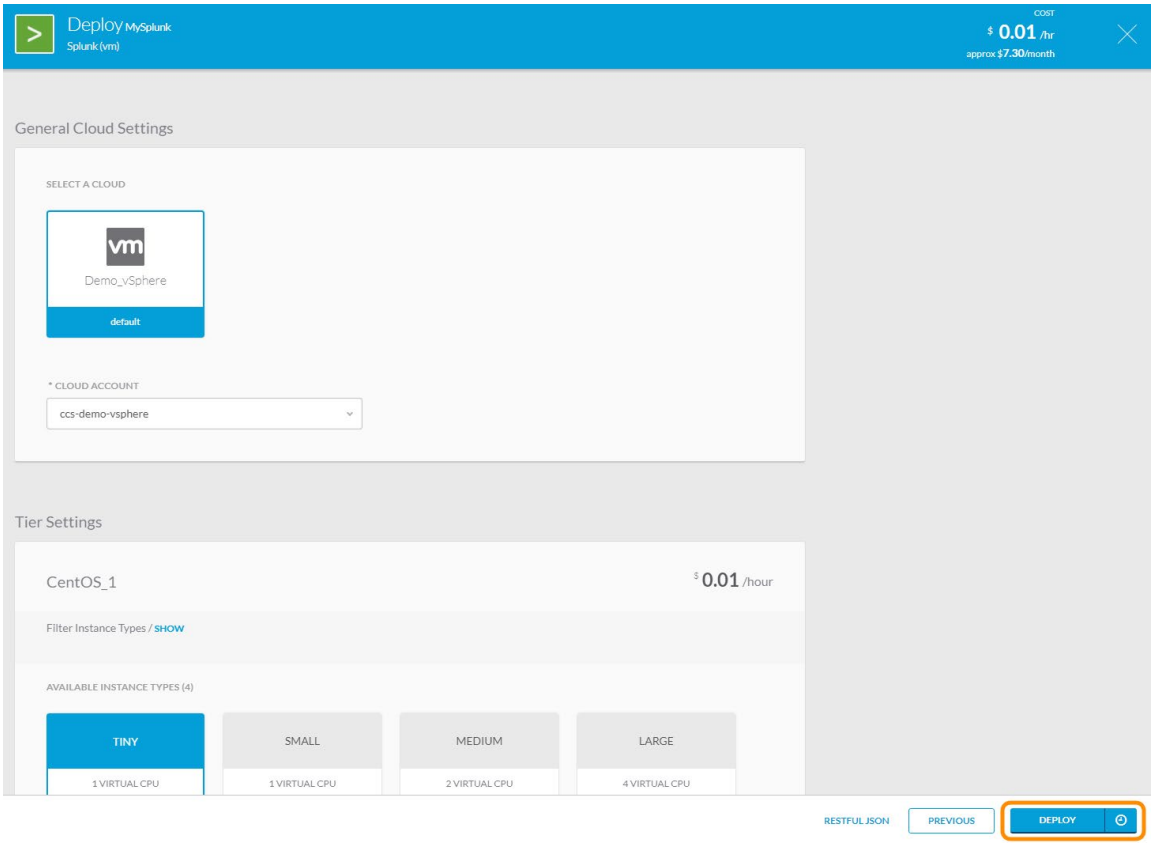


19. Enter **MySplunk** at the **Deployment Name**.

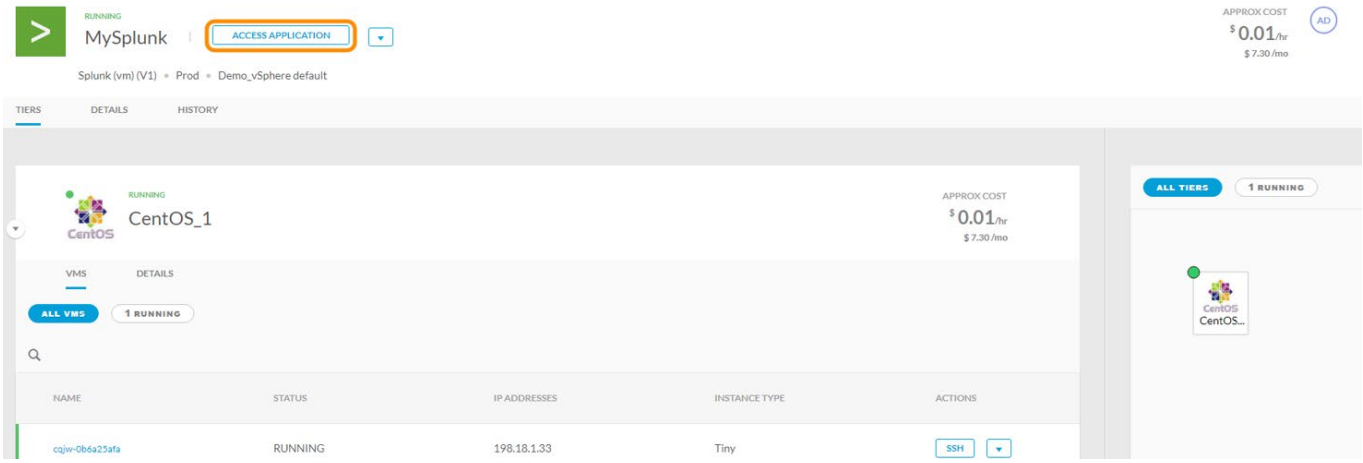


20. Leave the rest of the fields to their default values and click **Next** at the bottom of the form.

21. At the next form, leave the defaults and click **Deploy**.



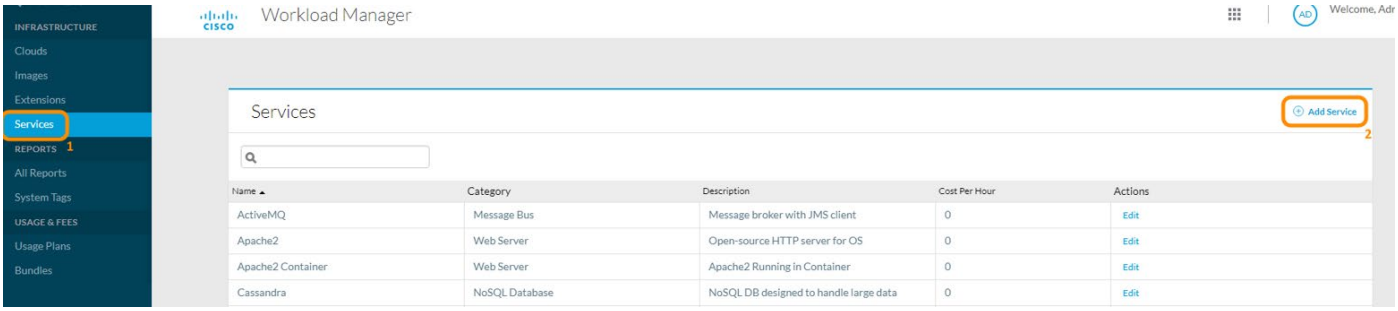
22. Once the status of the deployment changes to Running, click on the **Access Application** to launch a browser connection to the new application.



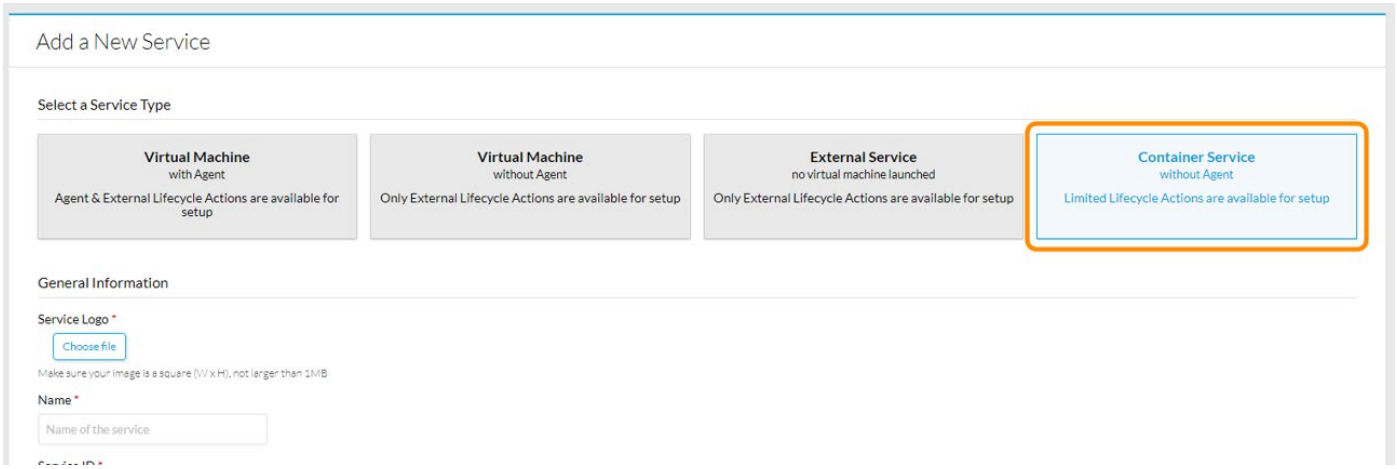
NOTE: Should the user wish to further experiment with the specific application, the admin credentials for this application are **admin / changeme.**

Creating a Container-Based Application Profile

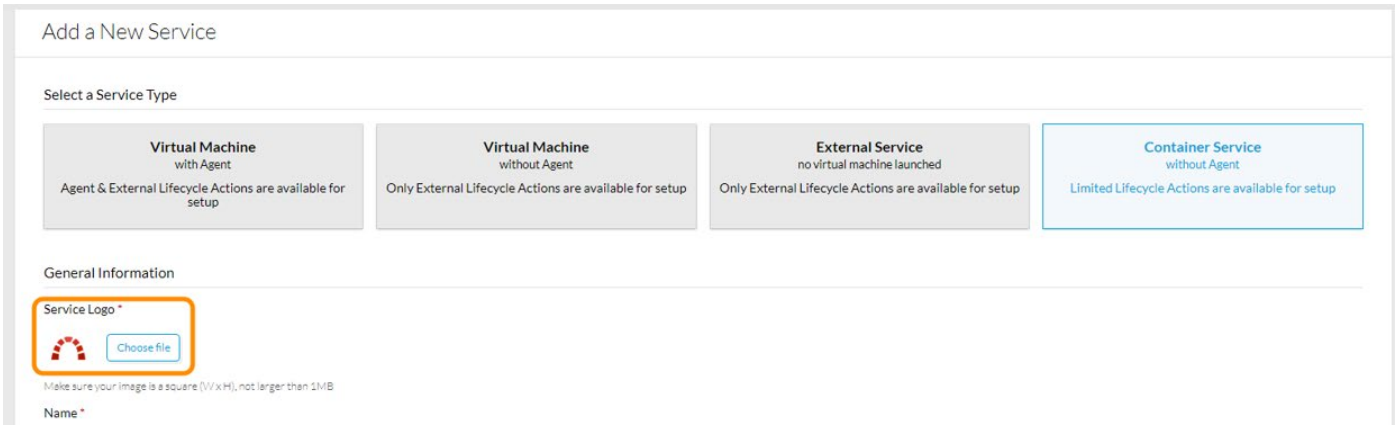
1. Navigate to the CloudCenter UI, login as **admin@dcloud.cisco.com / C1sco12345**, tenant: **demo**.
2. Select **Workload Manager**.
3. Click on **Admin**. 
4. Click on **Services**, then click on **Add Service**.



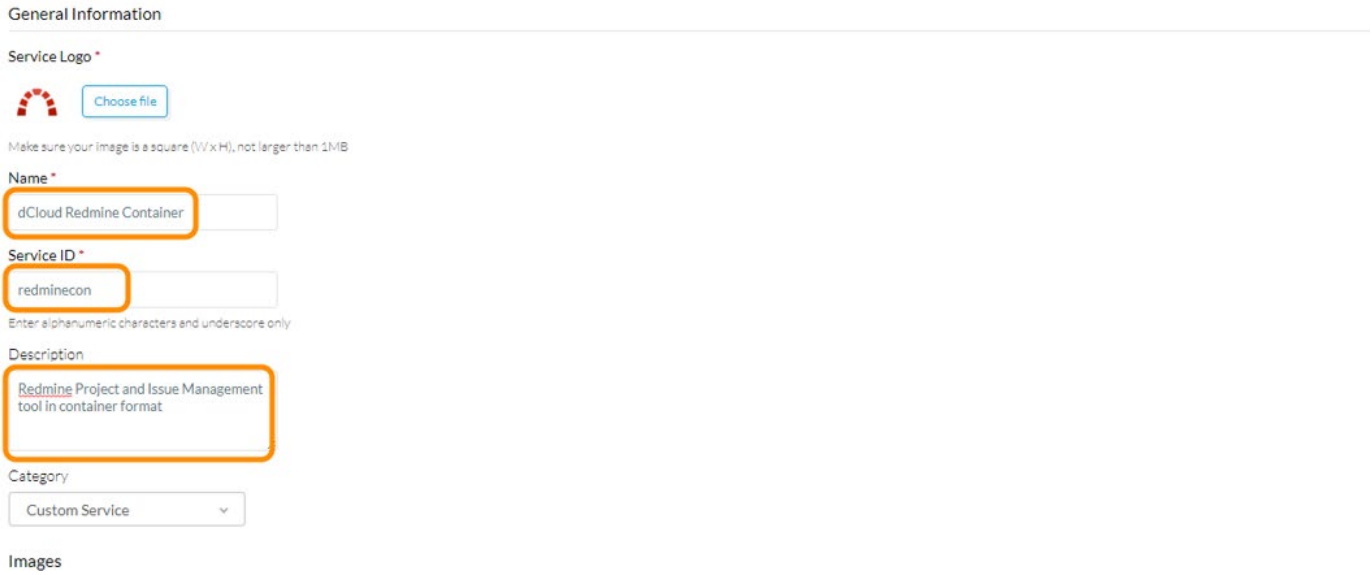
5. Select **Container Service** at the Service Type



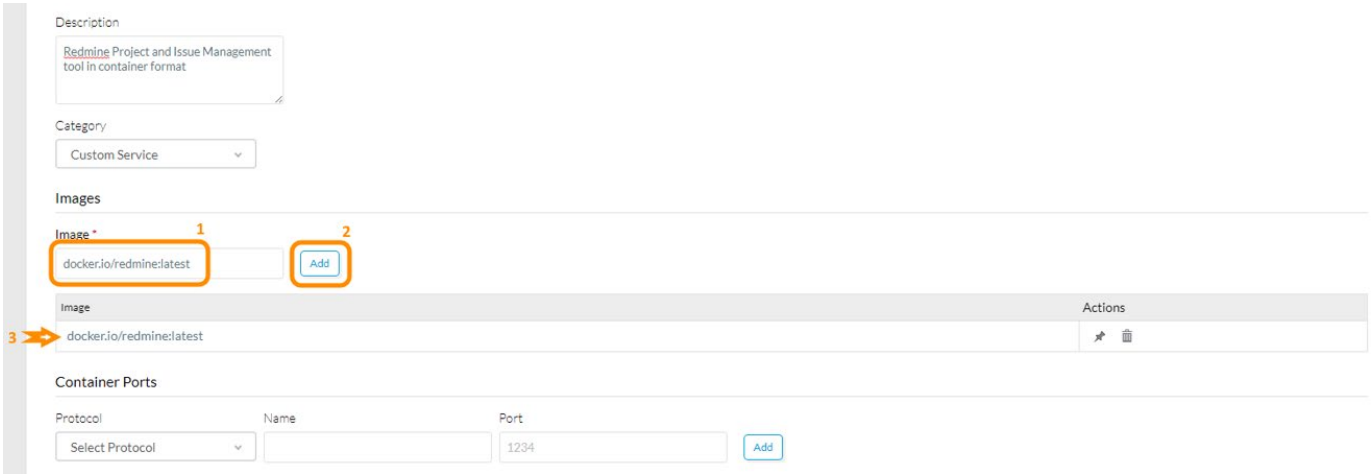
6. Click **Choose File** at the Service Logo. Then, navigate to **Pictures** and locate the file **redmine-logo.png**. Select the file and click **Open.** Ensure the application logo shows up.



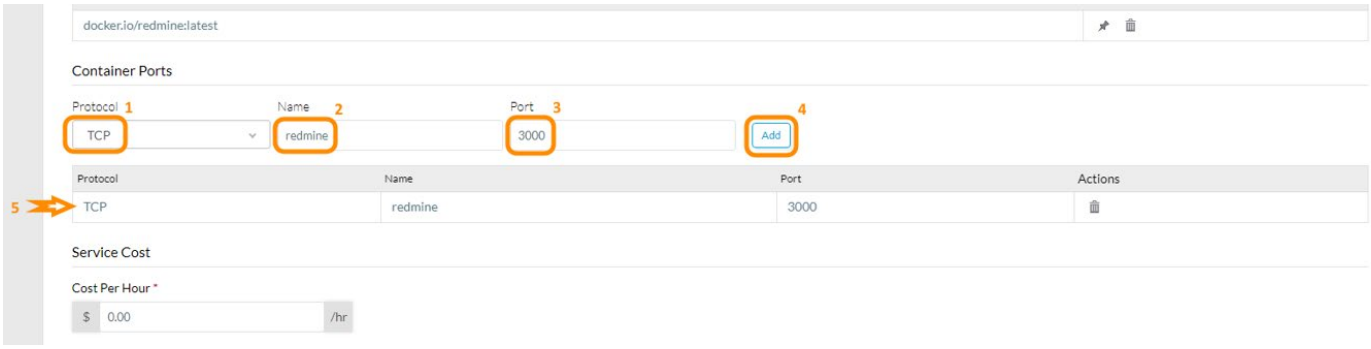
7. Enter **dCloud Redmine Container** at the Name, then enter **redminecon** at the Service ID. Next, enter **Redmine Project and Issue Management tool in container format** at the Description.



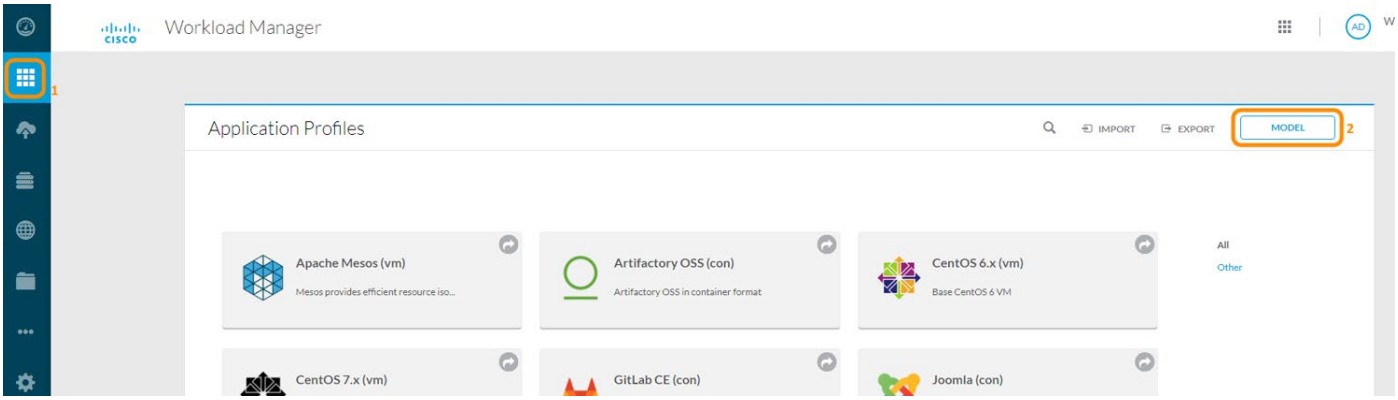
8. Enter **docker.io/redmine:latest** at the Image and click **Add**. The image should show up on the list below.



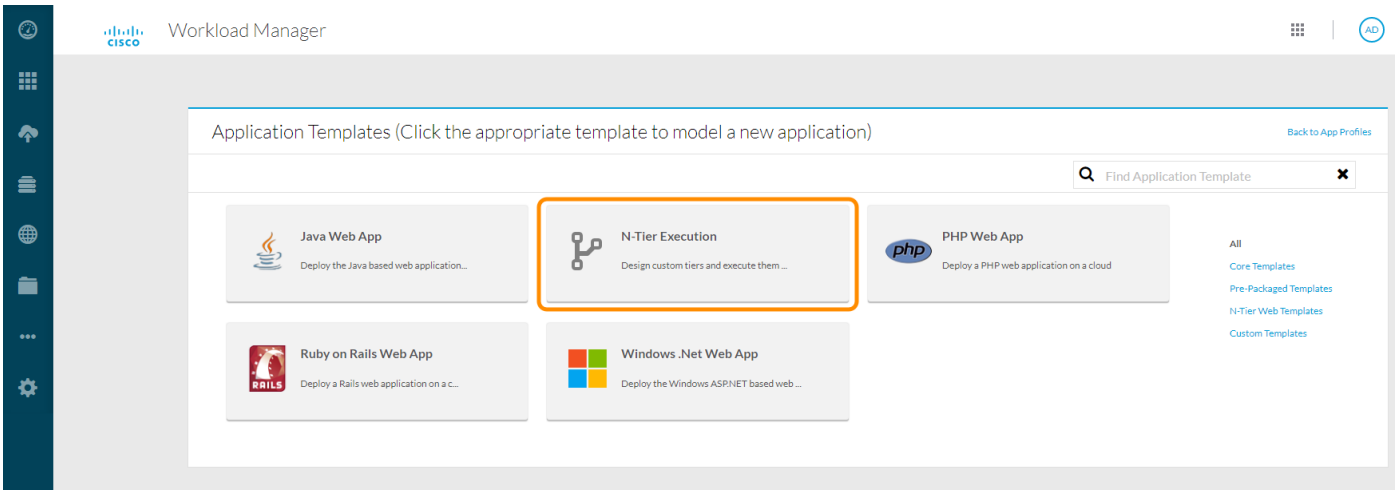
9. At the Container Ports, enter **TCP** for Protocol, **redmine** for Name and **3000** for Port, then click **Add**. The mapping should show up at a list below the fields.



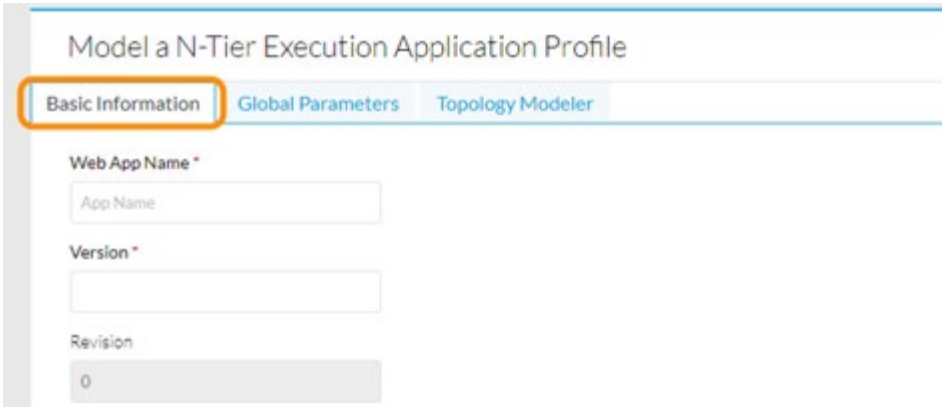
10. Leave the rest of the fields at their default values and click **Save**.
11. At the left-hand side, click on **MAIN MENU** to return to the main choices.
12. Click on **App Profiles**, then click on **MODEL**.



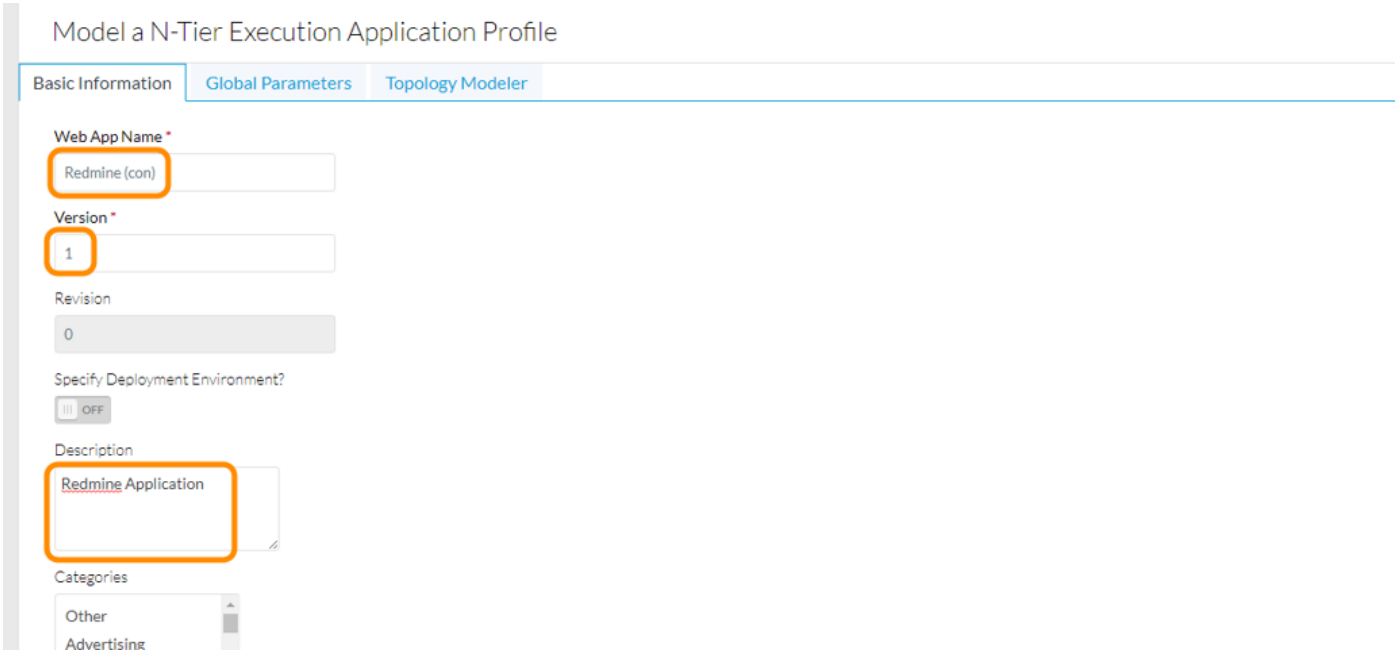
13. At the Application Templates, click on **N-Tier Execution**.



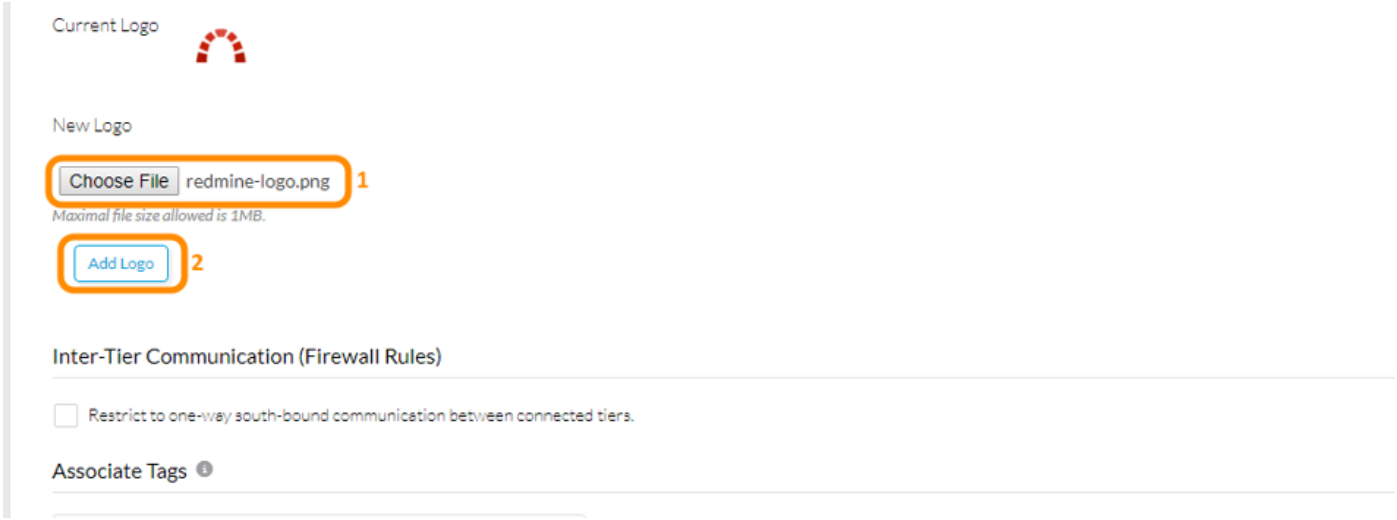
14. Click on the **Basic Information** tab.



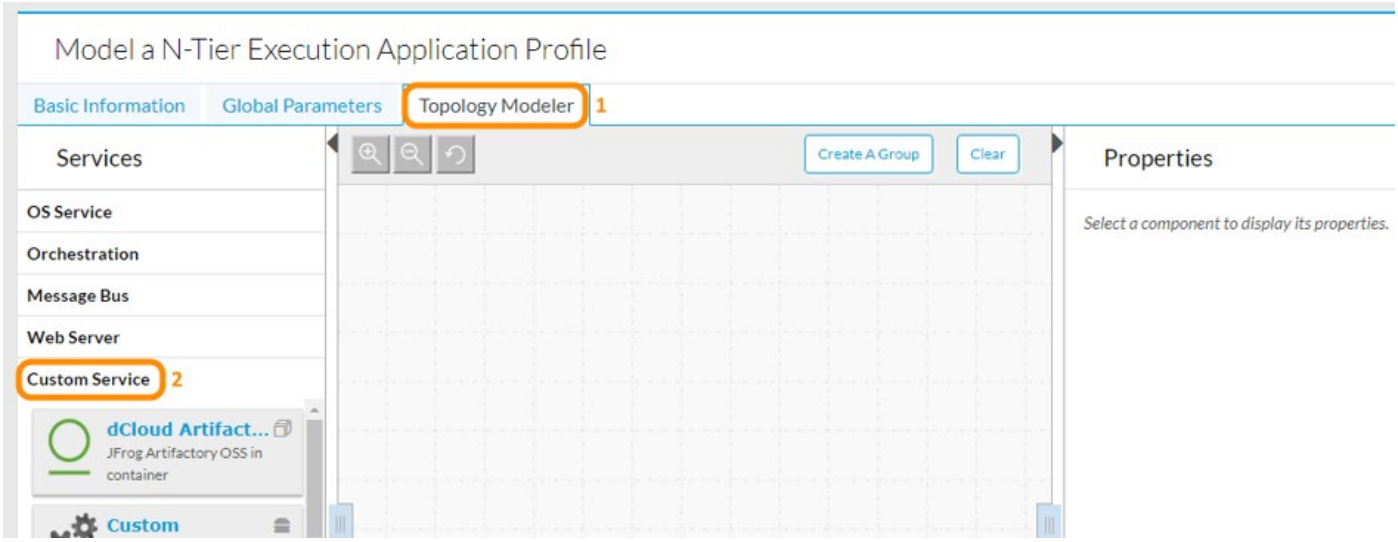
15. Enter **Redmine (con)** at the **Web App Name** field, **1** at the Version field and **Redmine Application** at the Description.



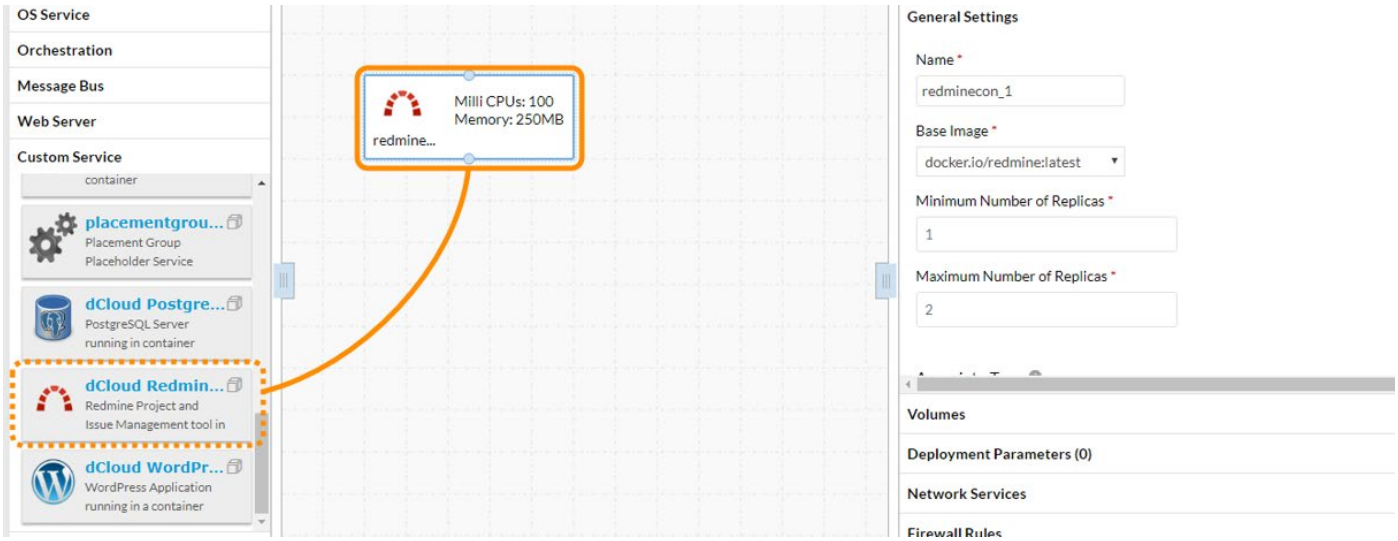
16. Scroll down and locate the New Logo section. Click on **Choose File**. Navigate to Pictures and select the **redmine-logo.png** file, then click **Open**. Next, click **Add Logo**. The logo should show up.



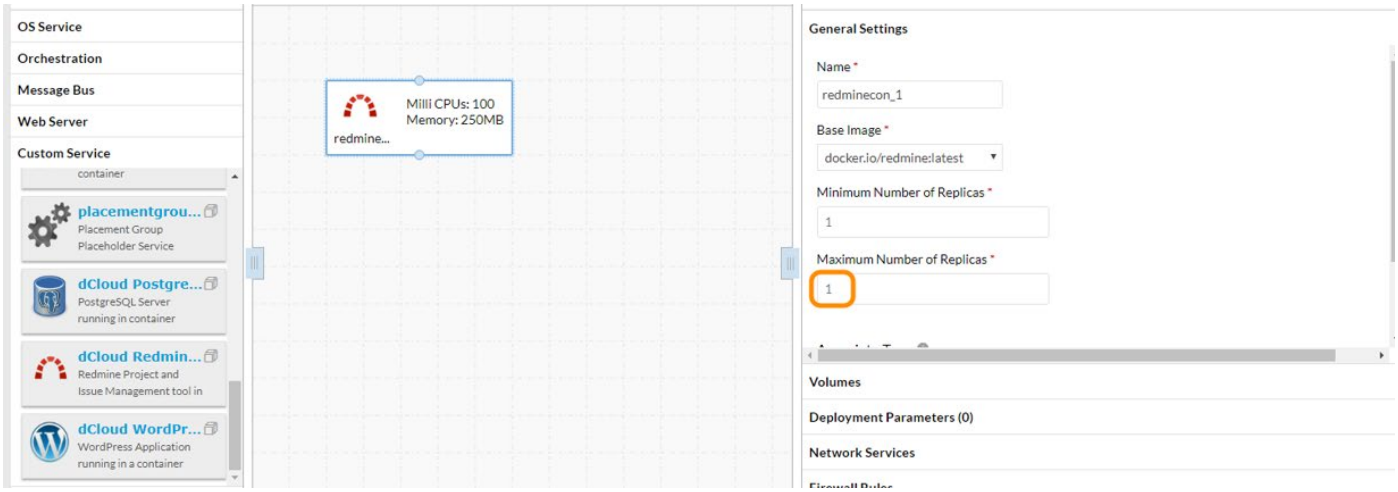
17. Next, navigate to the **Topology Modeler** at the top of the form. At the Services section on the left-hand side, expand the **Custom Service** section.



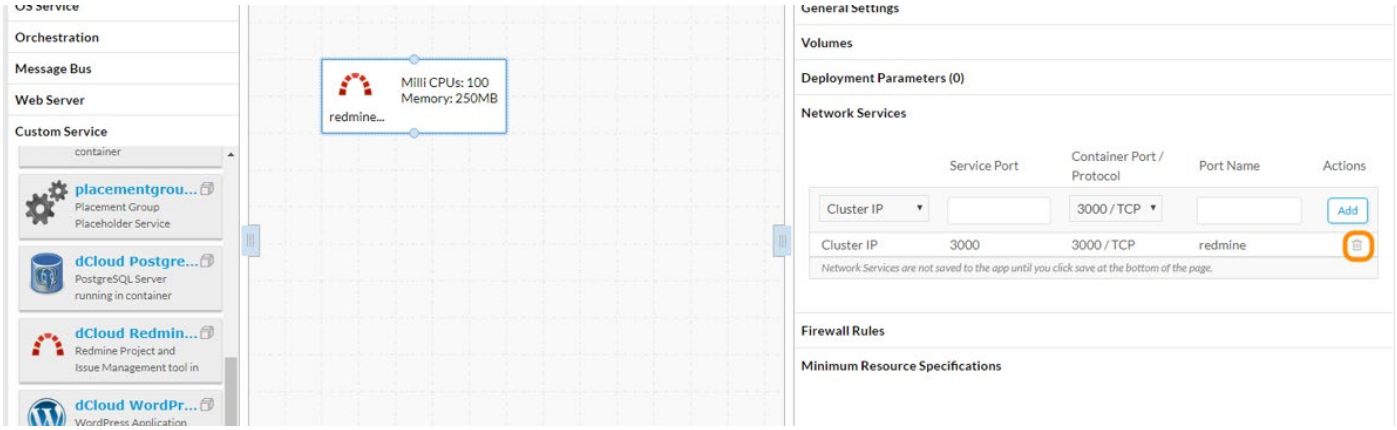
18. Locate the **dCloud Redmine Container** service and drag and drop it to the modeler.

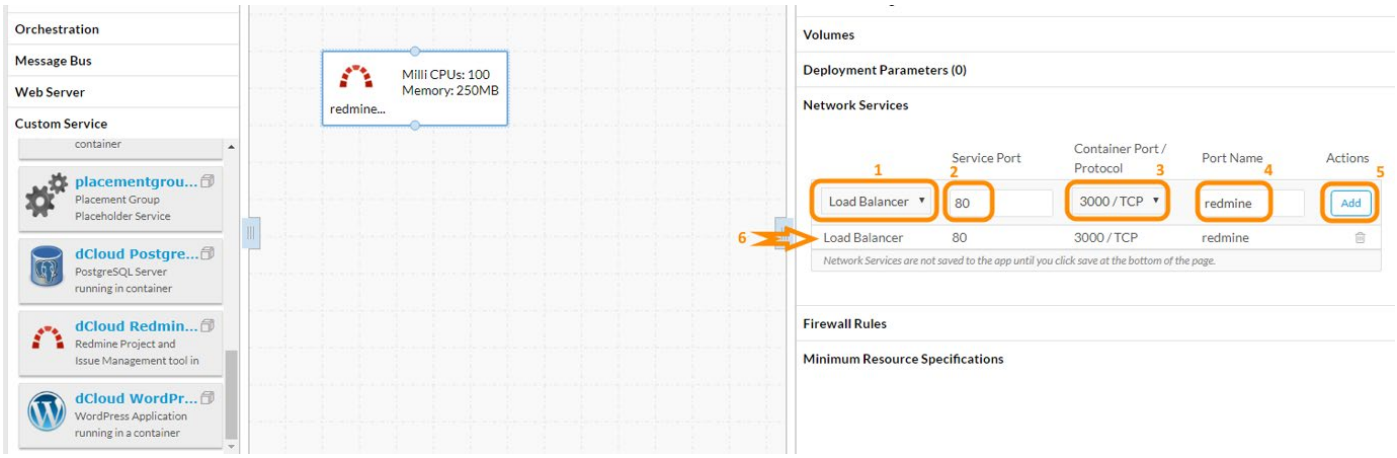


19. Click on the new object and at the Properties section on the right-hand side, under **General Settings**, change the value of Maximum Number of Replicas from **2 to 1**.

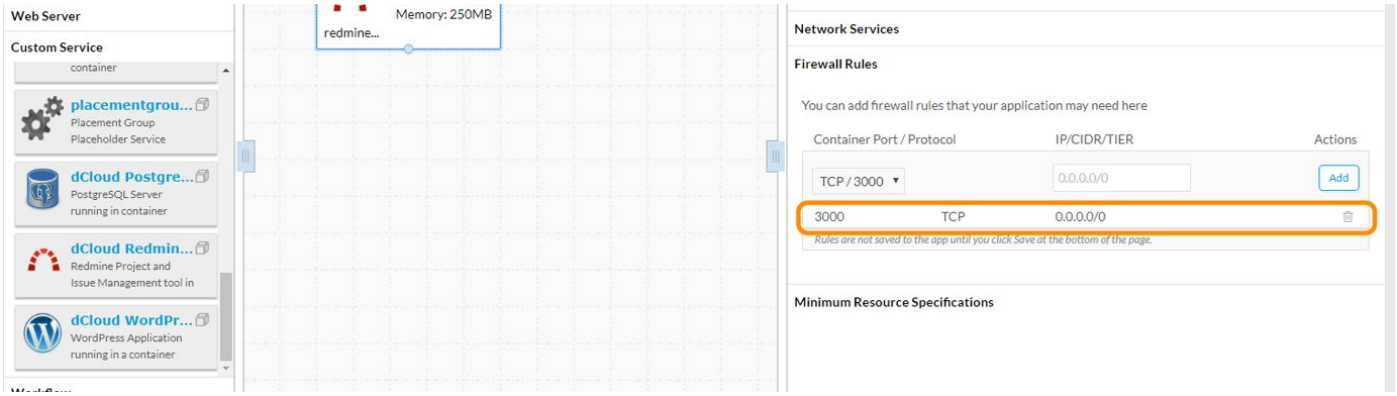


20. Select the **Network Services** section and remove all the records starting with Cluster IP by clicking on the icon at the right-hand side. Then, create a new record of type Load Balancer, enter **80** as Service Port, **3000/TCP** as Container Port/Protocol and **redmine** as Port Name, then click **Add**. Verify that a new record has been added to the list.

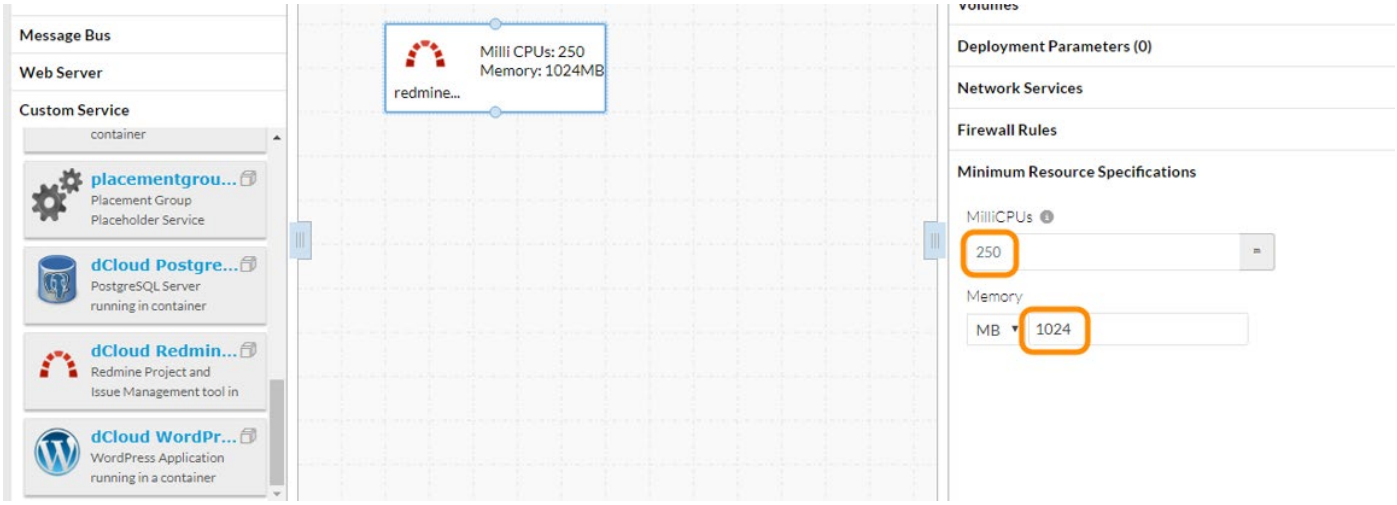




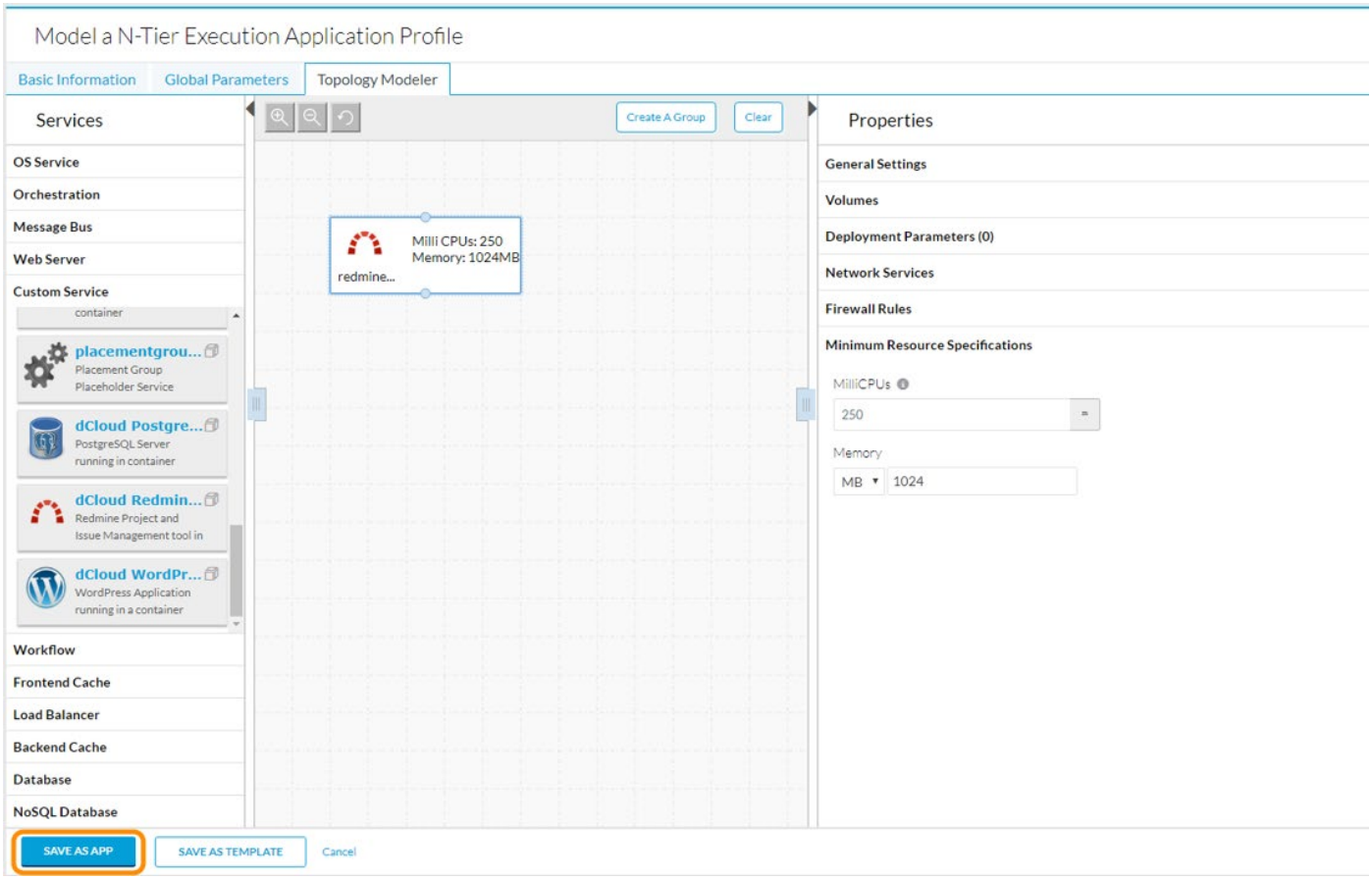
21. At the Firewall Rules section, ensure a record with the values **3000, TCP, 0.0.0.0/0** exists.



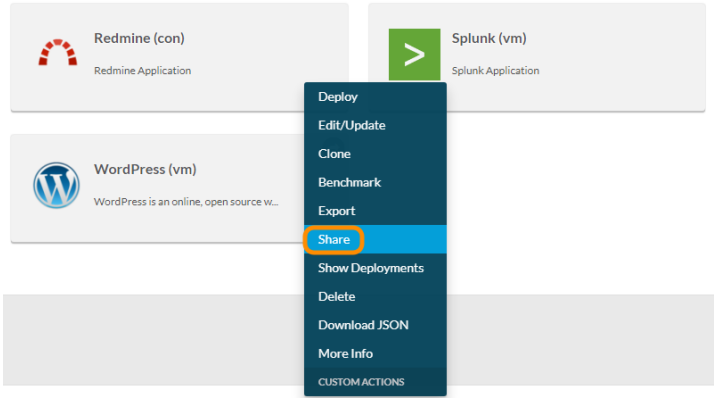
22. Then, at the Minimum Resource Specifications section, change the value at **MilliCPUs** from **100 to 250** and the Memory from **250 to 1024 MB**.



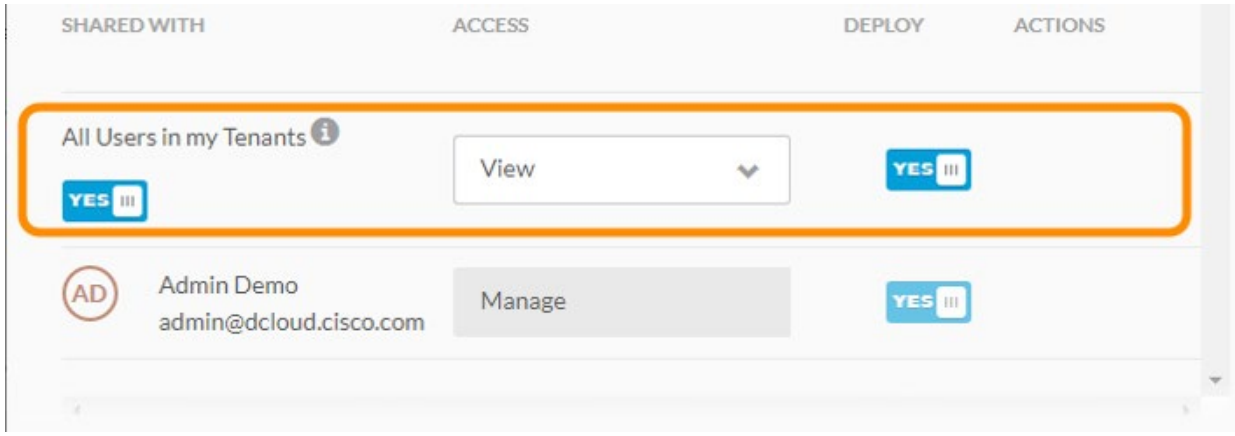
23. Click **Save As App** at the bottom of the form to save the new App Profile.



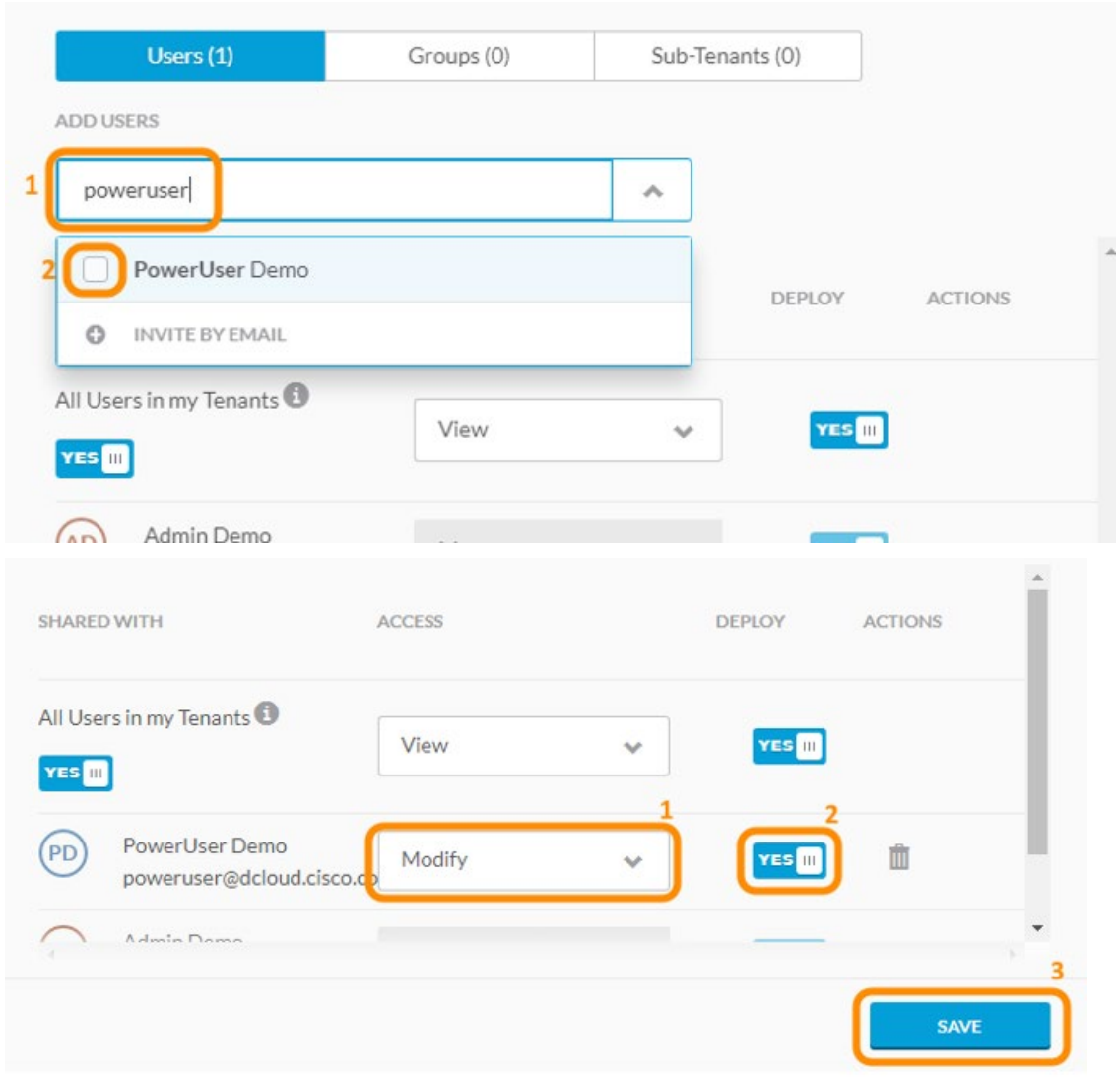
24. To Share this App Profile so other users can use it, click on **Share**.



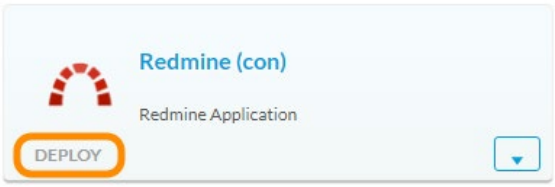
25. At the next form, under All Users in my Tenants, set the switch from **No** to **Yes**, leave the Access to View and under Deploy set the switch from **No** to **Yes**.



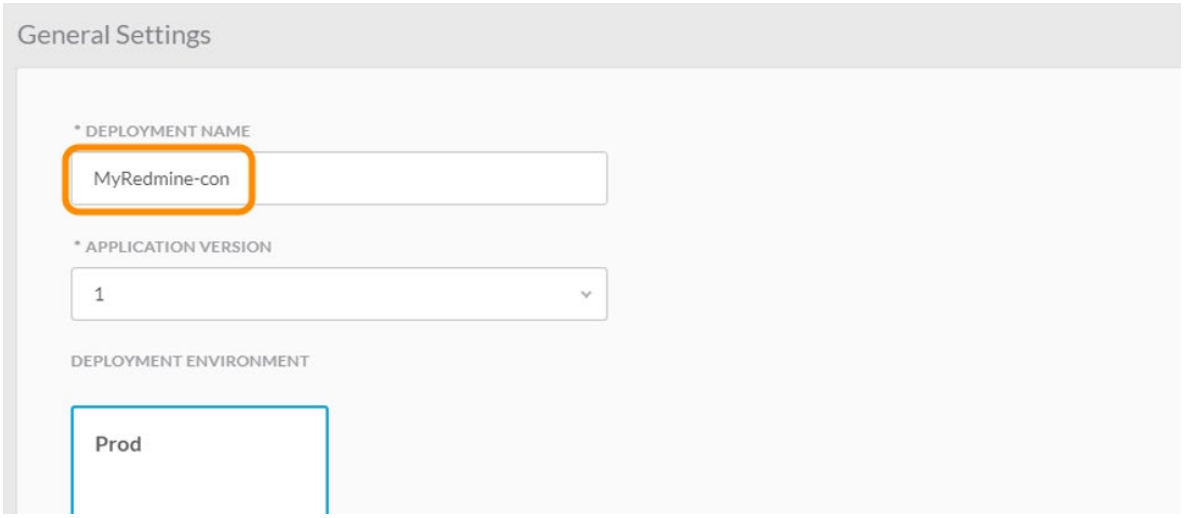
26. Next, at the Add Users text box, type **PowerUser Demo** and select the user from the list. Set the Access to **Modify** and the Deploy to **Yes** for this user, then click **Save**.



27. Once back at the App Profiles form, click on the **Redmine (con) App Profile** to initiate a deployment.

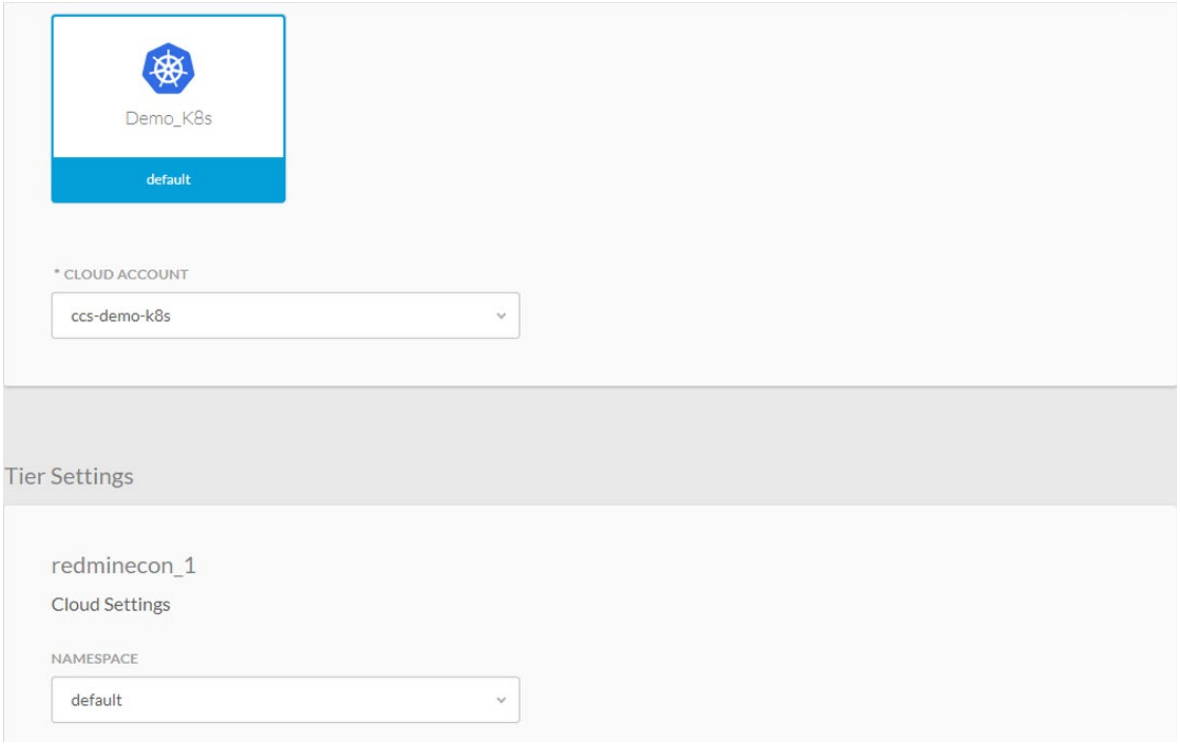


28. Enter **MyRedmine-con** at the Deployment Name.



29. Leave the rest of the fields to their default values and click **Next** at the bottom of the form.

30. At the next form, leave the defaults and click **Deploy**.

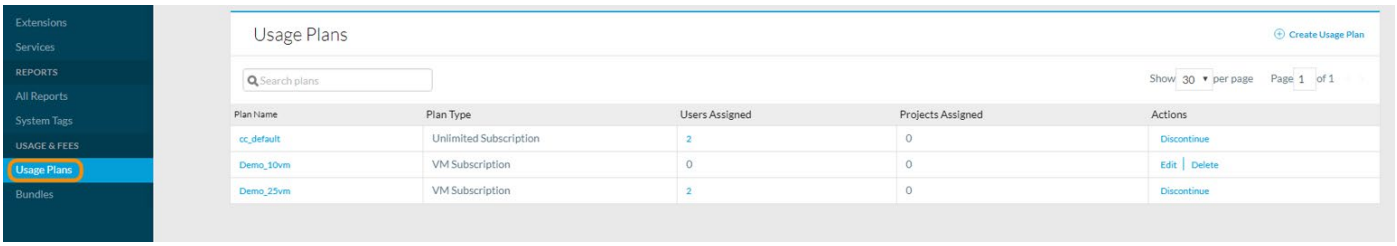


31. Once the status of the deployment changes to **Running**, click on the **Access Application** to launch a browser connection to the new application.

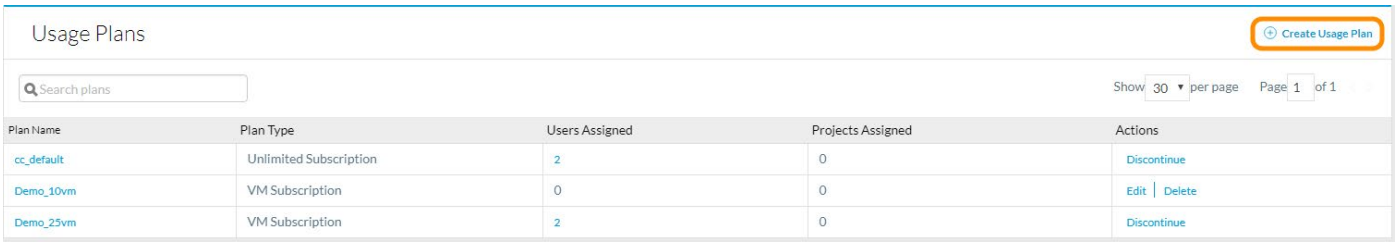
NOTE: Should the user wish to further experiment with the specific application, the admin credentials for this application are **admin / admin.**

Creating a Usage Plan

1. Log in to Cisco CloudCenter as admin (**admin@dcloud.cisco.com / C1sco12345**, tenant id: **demo**).
2. Click **Admin** on the side menu.
3. At the admin menu, click on **Usage Plans**.



4. Click the **Create Usage Plan** link.



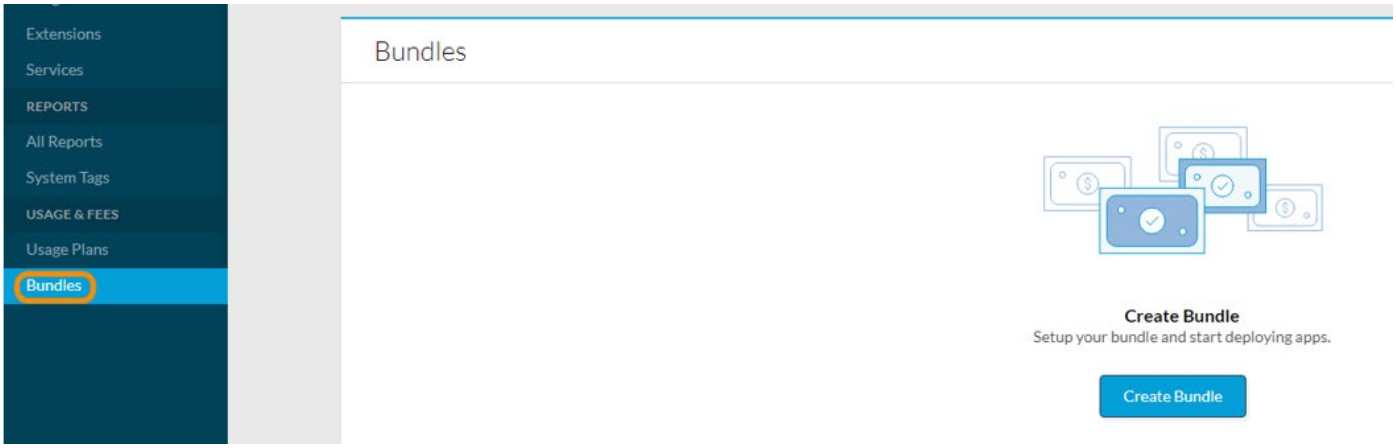
5. Explain the parameters of a usage plan and the possible values, as follows:

Parameter	Description	Values
Plan Name and Description		
Plan Type	Determines the type of bundle on which this usage plan will be based – users of the system will have a plan associated to their profile, and will be able to create applications according to the terms of their plan.	VM-hour Subscription VM Subscription Prepaid VM-hour Business Prepaid Budget Bundle Unlimited Subscription
Monthly VM Hours	The number of VM-hours that will be allowed under this plan – only required for plans that are hour-based	Determined by company policy
Usage Increment Units	Controls the increment steps. e.g.: if a 10 min increment unit is configured and the usage is 11 mins, the metering will be evaluated as 10 + 10 = 20 mins	Determined by company policy
Only Visible to Tenant Admin	Determines whether this usage plan will only be visible by the Tenant Admin or all admins for this module.	True / False

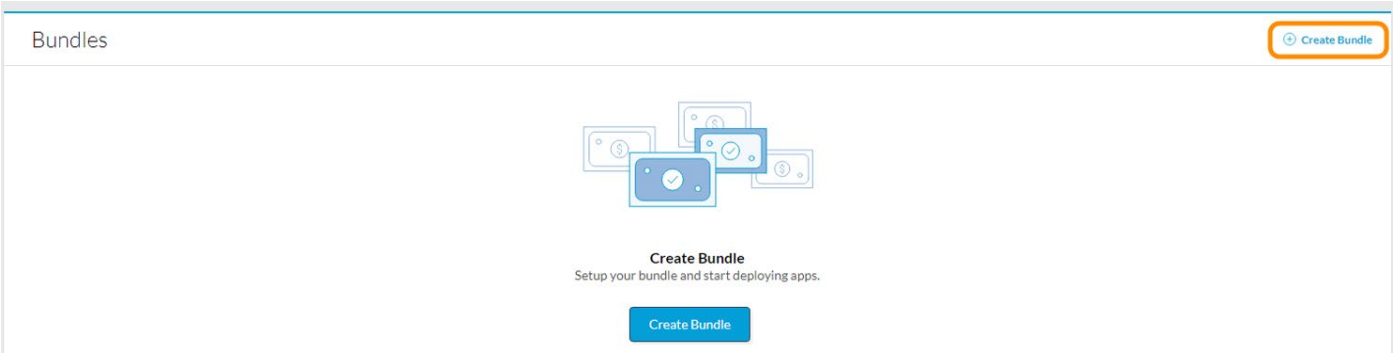
6. Click **Cancel** without creating a usage profile.

Creating a Bundle

1. Click **Bundles** on the side menu.



2. Click the **Create Bundle** link or **Create Bundle** button to show the process for creating a bundle.



3. Explain the parameters of a bundle and the possible values, as follows:

Parameter	Description	Values
Plan Name and Description		
Type	Determines whether a usage plan that uses this bundle will be based on hourly usage or a set number of dollars	Hour Based Budget Based
VM-hour Limit	Number of VM-hours allowed in an Hour Based plan – only required for Hour Based plans	
Budget Limit	The number of dollars per month that users (each user, or the user community as a whole?) can spend on VMs - only required for Budget Based plans	
Expiration Type		
Expiration Date	If the Expiration Type is Fixed Date, the date on which the bundle will expire	
Only Visible to Admin	Determines whether users can see the details of the usage plan	

4. Click **Cancel** without creating a Bundle.

Scenario 4. CI/CD Solution using GitLab, Jenkins, Artifactory and CloudCenter

The purpose of scenario is to demonstrate how developers can move code along the Continuous Integration / Continuous Delivery pipeline using CloudCenter, Jenkins, Artifactory and GitLab. This workflow is often referred to by those pursuing DevOps methodologies. The integration simplifies the movement of code along the tools in the tool chain and exhibits how a critical system variable – such as the BUILD_NUMBER – can be exchanged between the different platforms.

The first part of the scenario is a walk-through of the environment, showing how the application is configured in CloudCenter, Artifactory, and Jenkins.

The second part of the scenario is to clone a Git branch, check out the code, modify one of the files and observe Jenkins perform two builds:

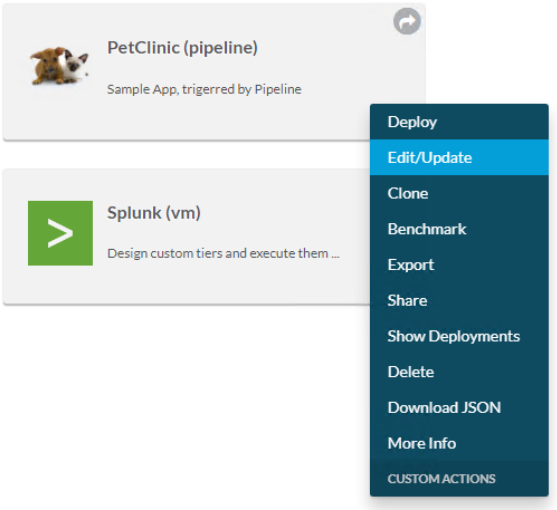
- The PetClinic.Build build takes the newest code to produce artifacts and store them into Artifactory
- The PetClinic.Deploy build deploys the application profile with the newly stored artifacts, leveraging the plugin for CloudCenter

Steps

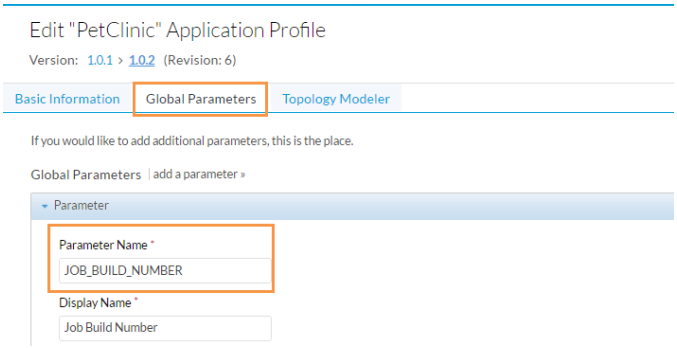
Environment Walkthrough

The purpose of this section is to become familiar with the CloudCenter, Artifactory, and Jenkins environments of the PetClinic application.

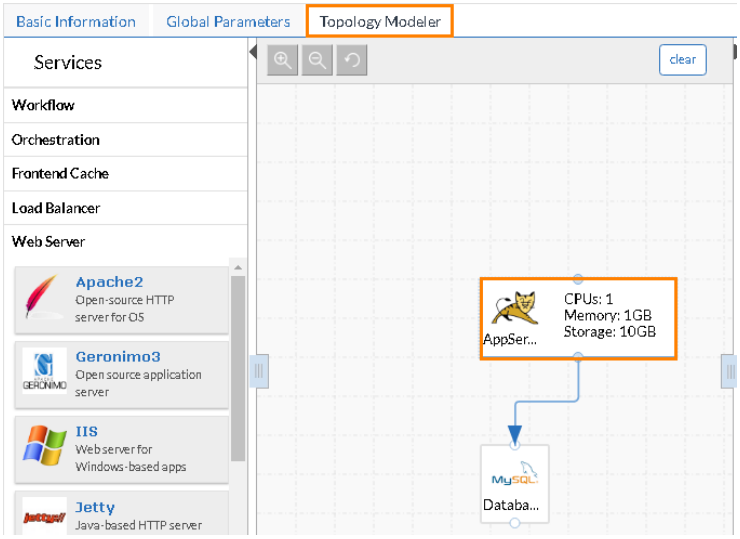
1. Log in to Cisco CloudCenter (**admin@dcloud.cisco.com / C1sco12345 / demo**).
2. Select **Workload Manager.**
3. Click **App Profiles** in the side menu.
4. Mouseover the **PetClinic (pipeline)** application. Click the down arrow and select **Edit/Update** from the resulting menu to review the application parameters.



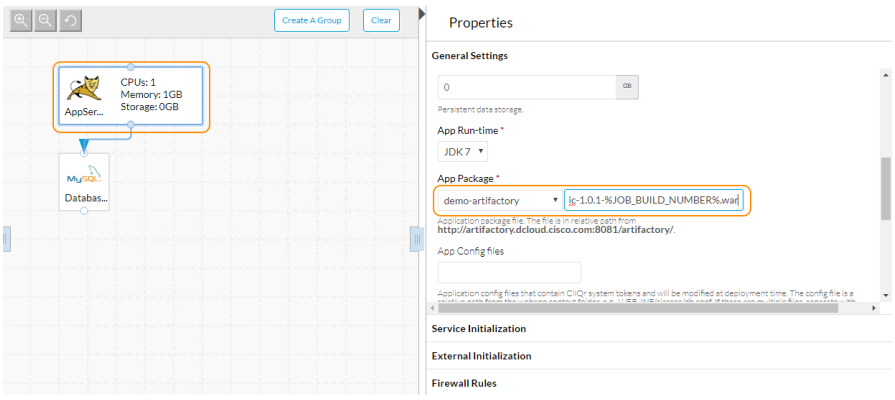
5. Click the **Global Parameters** tab.
6. Note that the Parameter Name is **JOB_BUILD_NUMBER**, a variable. The build number is passed from CloudCenter to Artifactory to Jenkins.



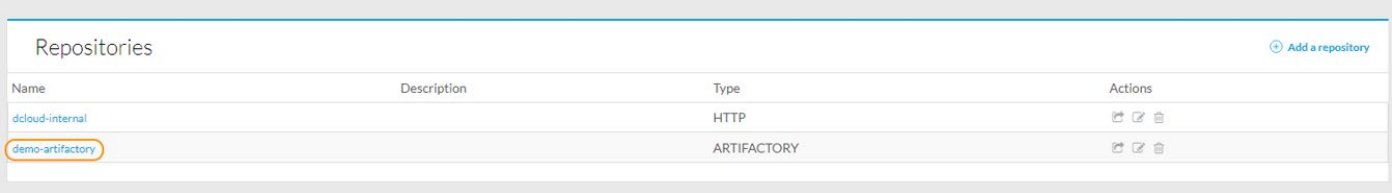
7. Click the **Topology Modeler** tab.
8. Click the **Apache Server**.



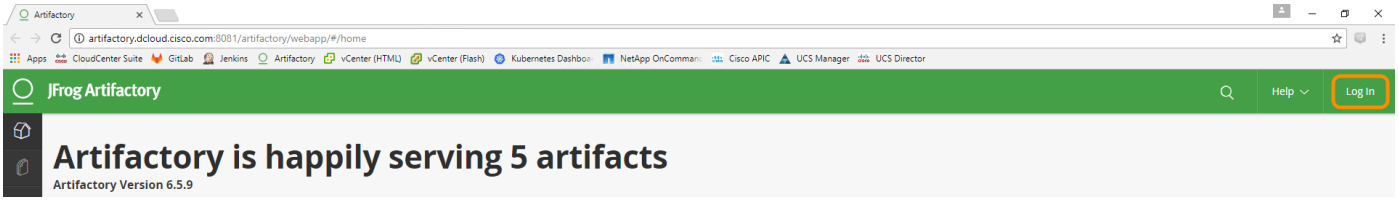
9. Scroll down the **Properties** list to the **App Package**.
10. Note that the application package is an **Artifactory** package.
11. Double-click in the Package File field and scroll over to show that the **JOB_BUILD_NUMBER** is part of the package name.



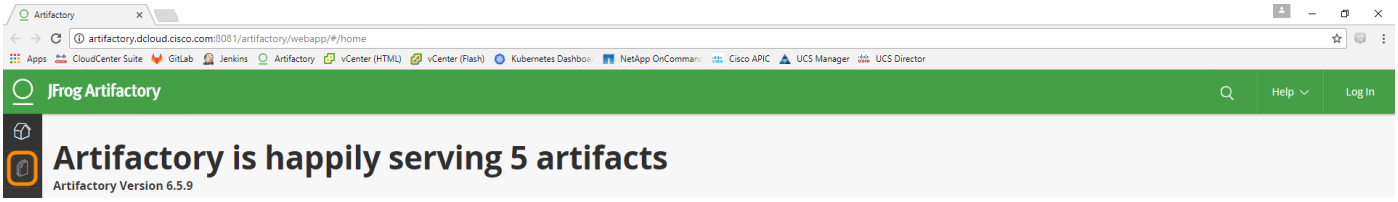
12. Click **More > Repositories** in the side menu and note that **Artifactory (demo-artifactory)** is one of the repositories.
13. If desired, click **demo-artifactory** to examine the settings.



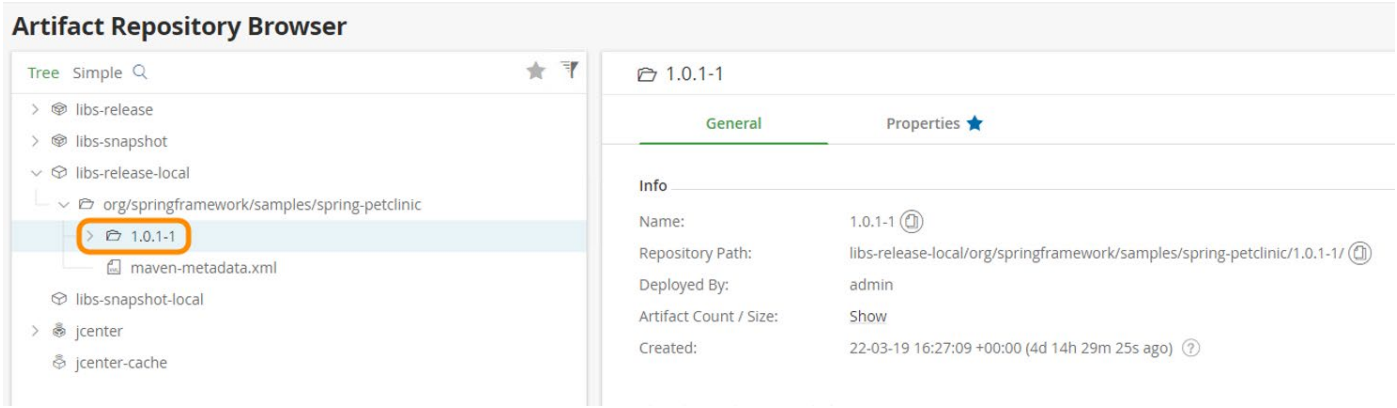
14. Open a Chrome browser and click the **Artifactory** Artifactory bookmark.
15. Click **Log In** and log in with the credentials **admin / C1sco12345.**



16. Click the **Artifacts** icon in the side menu.

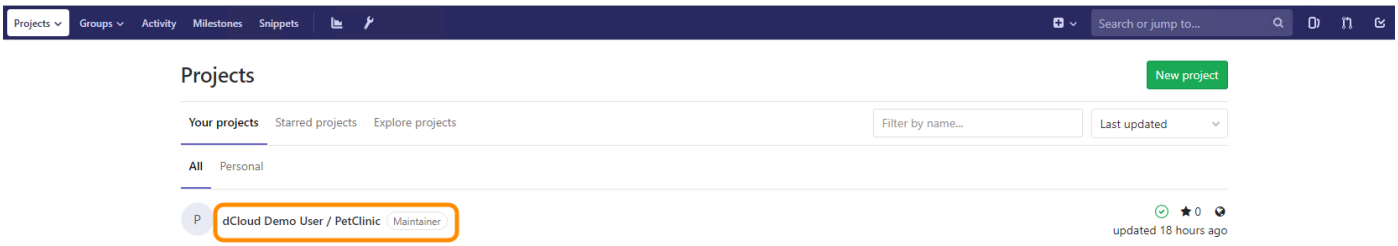


17. Expand the libs-release-local **repository**, then the org/springframework/samples/spring-petclinic directory and click the latest release number, 1.0.1.1. Show the parameters.

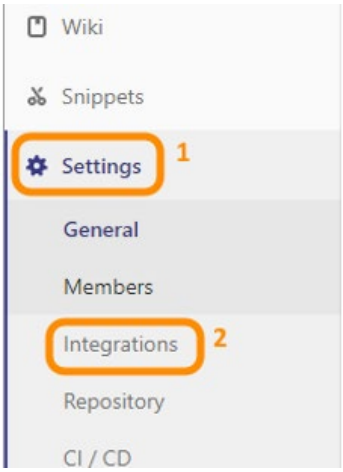


18. Open a new browser tab and click the GitLab bookmark  to open GitLab.

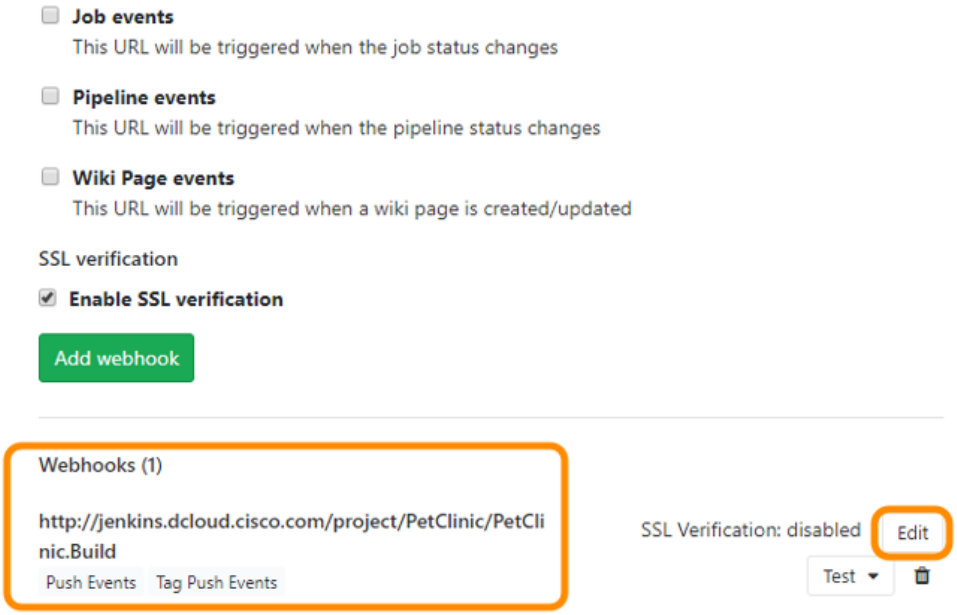
19. Sign in with the default login credentials (**demouser@dcloud.cisco.com** / **C1sco12345**).
20. Click **dCloud Demo User / PetClinic.**



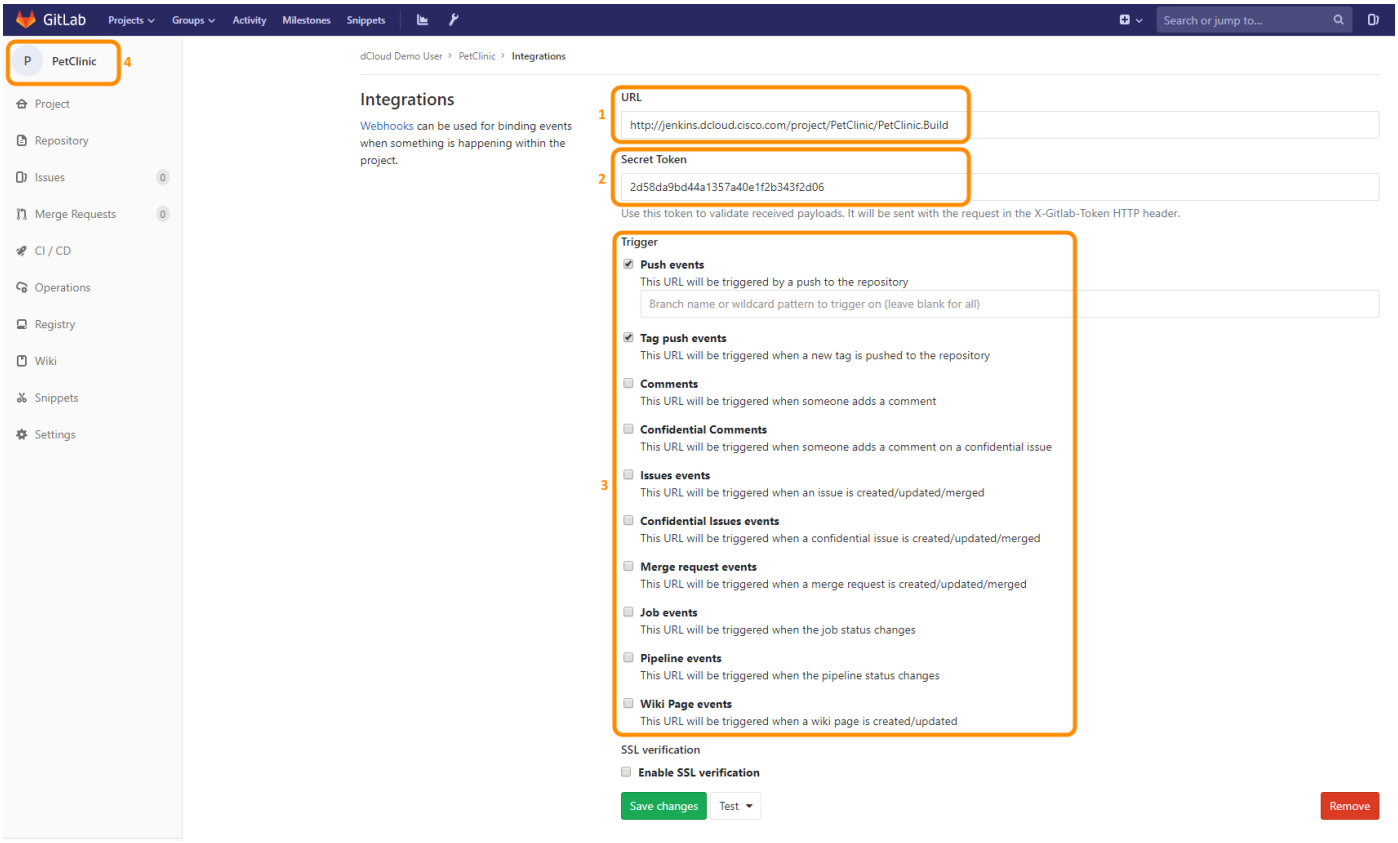
21. Click **Repository** in the top menu to view the repository files.
22. Click the **Settings** icon and then select **Integrations** from the resulting menu.



23. Scroll down to the middle of the form to locate the existing integration and click **Edit**.

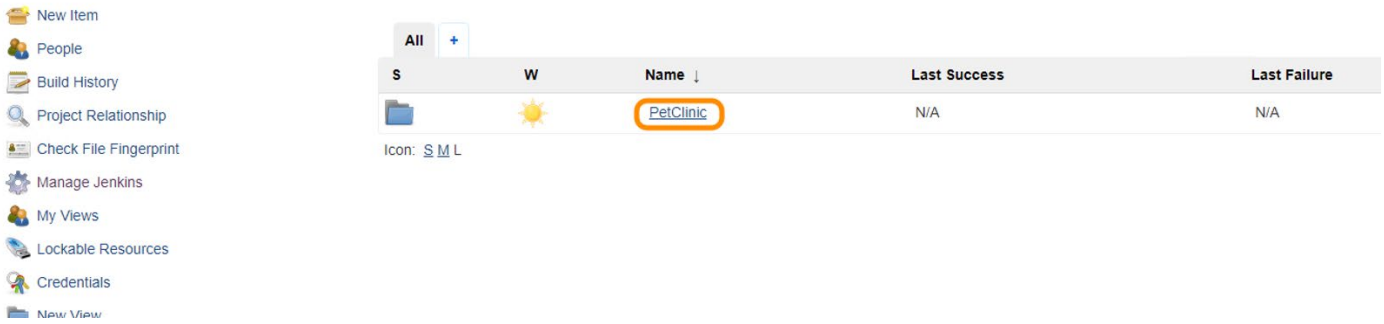


24. Observe the configured settings for this webhook, the **Change Notification URL**, the Notification Authentication Token and the **Notification Event Filters,** then click at the **PetClinic** link to exit without saving changes.

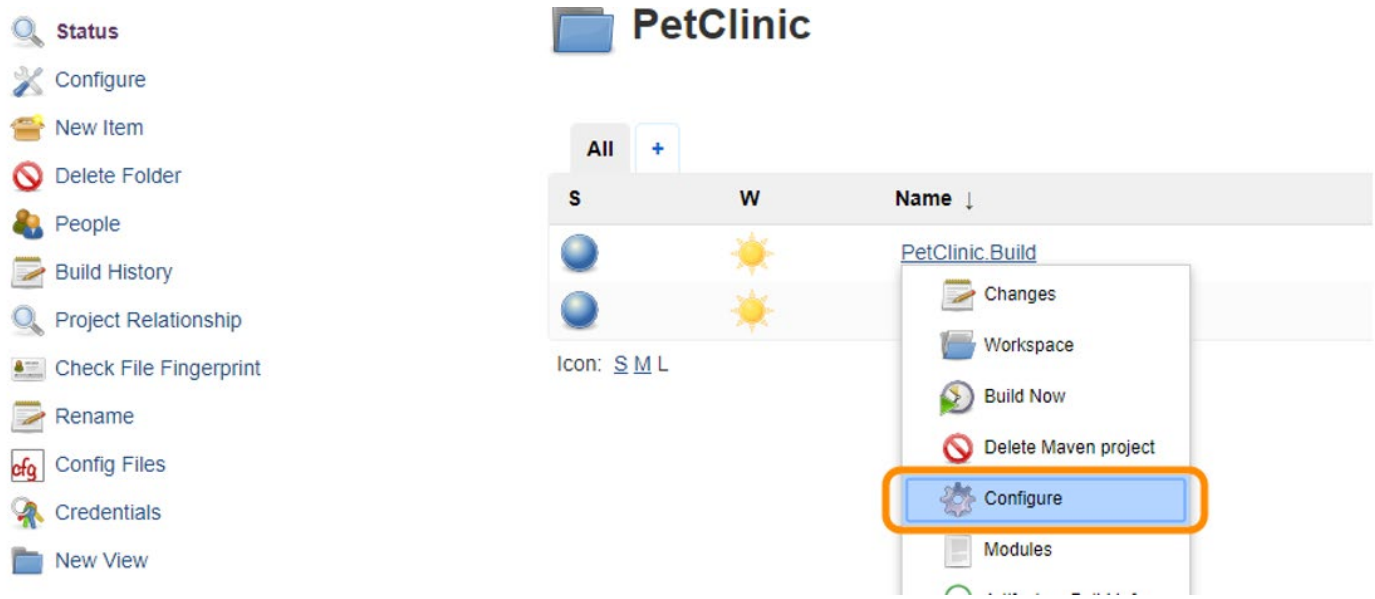


25. Open a new browser tab and click the **Jenkins** Jenkins [Dashboard] bookmark. Log in with the default credentials (**admin/C1sco12345**). The dashboard shows the **PetClinic** folder containing the two builds for the PetClinic application – the artifact build (PetClinic.Build) and the application deployment build (PetClinic.Deploy).

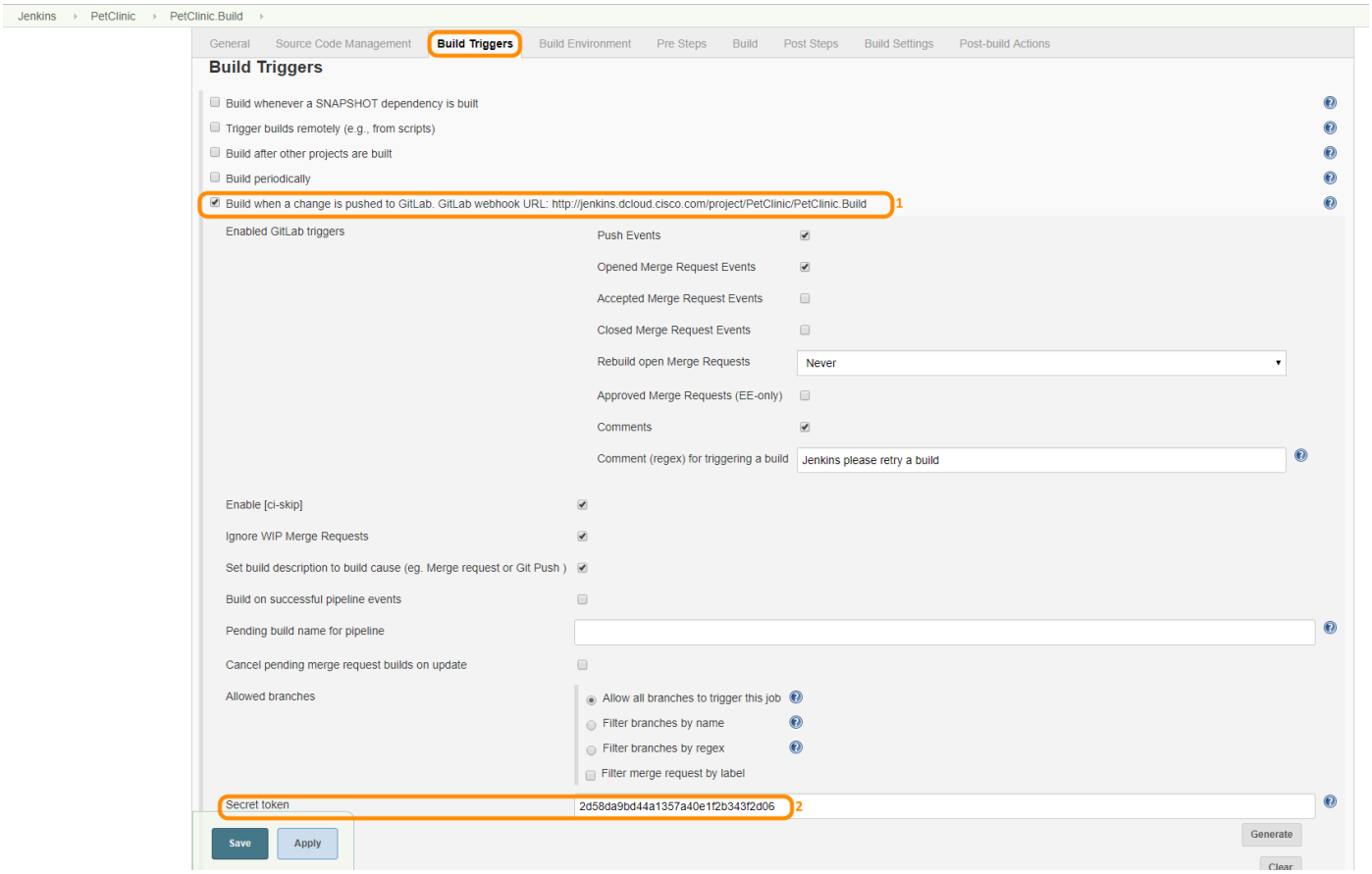
26. Click on the **PetClinic** folder to view the two jobs related to PetClinic.



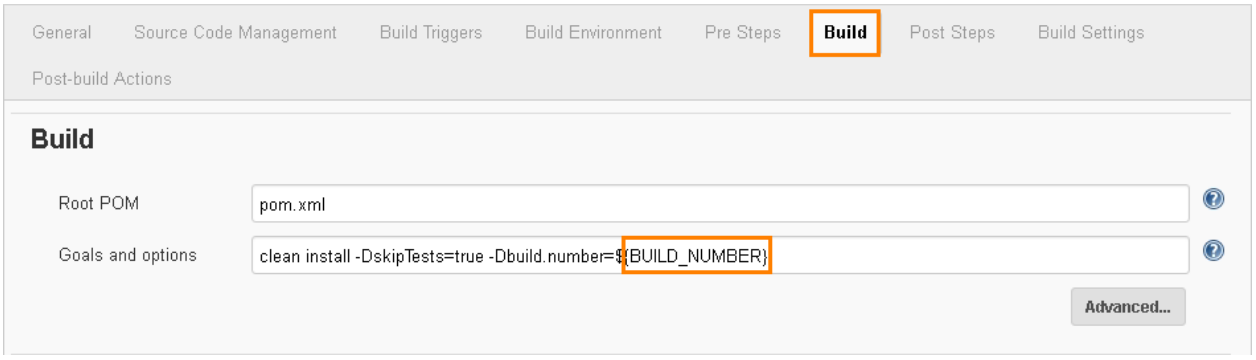
27. Mouseover **PetClinic.Build** to display the down arrow. Click the down arrow and select **Configure** from the menu.



28. Click **Build Triggers** to observe the configuration used for receiving build notifications. Notice the configured setting for enabling GitLab integration and the Secret token used to authenticate incoming requests .

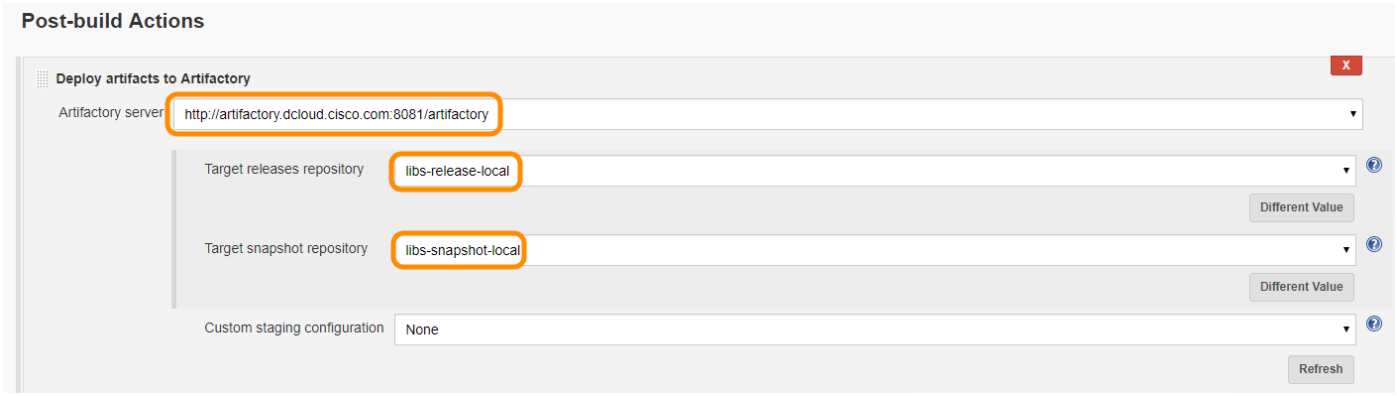


29. Click **Build** and show that the **BUILD_NUMBER** parameter is passed in Jenkins.

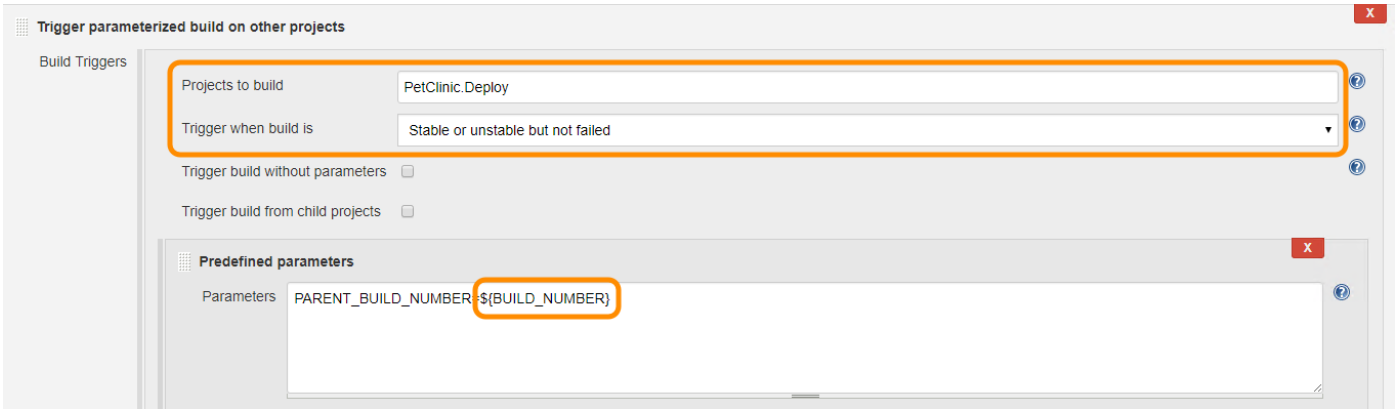


30. If desired, click **Post Steps** in the top menu to show the settings.

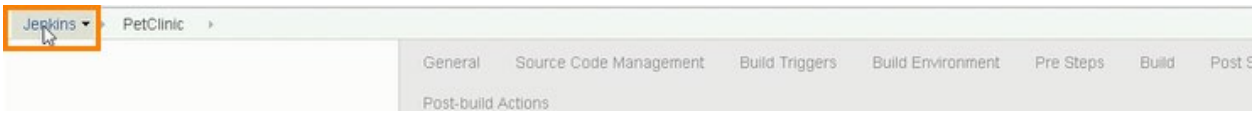
31. Click **Build Settings** in the top menu. Scroll down and show the **Deploy artifacts to Artifactory** settings. Note the Artifactory server, the Target releases repository, and the Target snapshot repository. These parameters set the repositories on the Artifactory server that will house the new artifacts and code from the Jenkins builds.



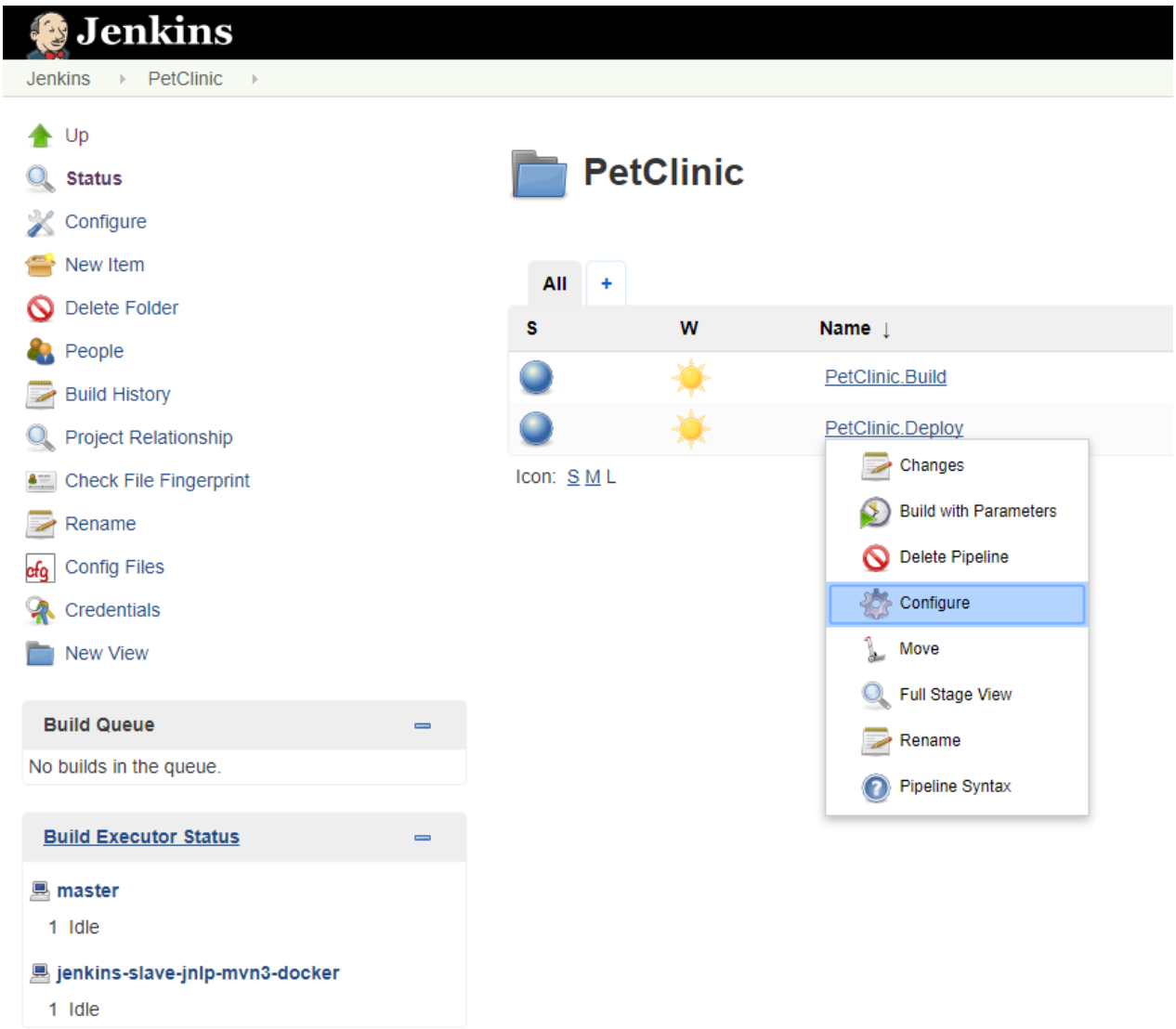
32. Scroll down to the **Trigger** parameterized build on other projects section and show that the Deploy_PetClinic project triggers when a build (from the PetClinic job) is Stable or Unstable but not failed. Also, note that the BUILD_NUMBER parameter is passed here.



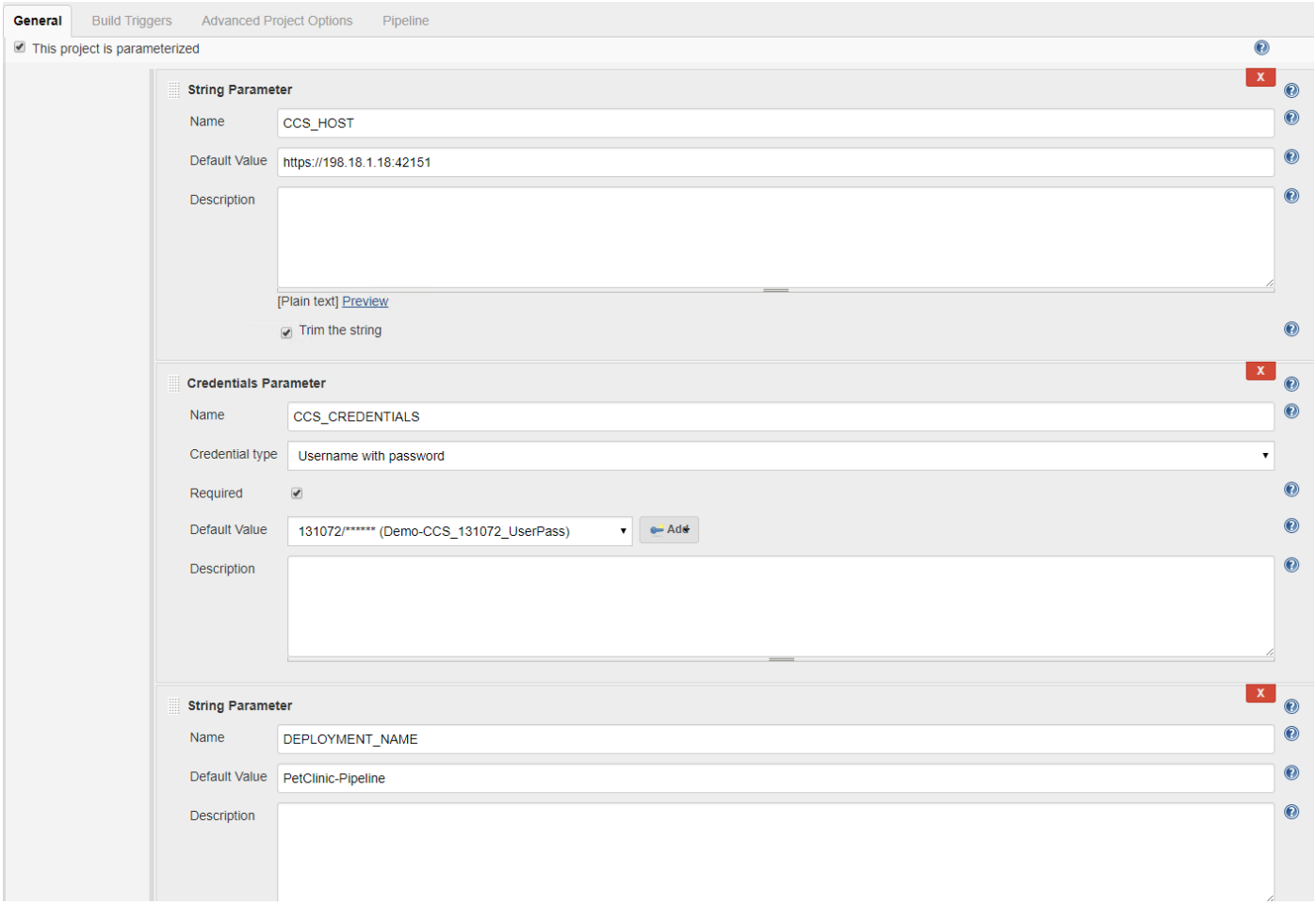
33. Click the **Jenkins** breadcrumb at the top of the screen to return to the dashboard.



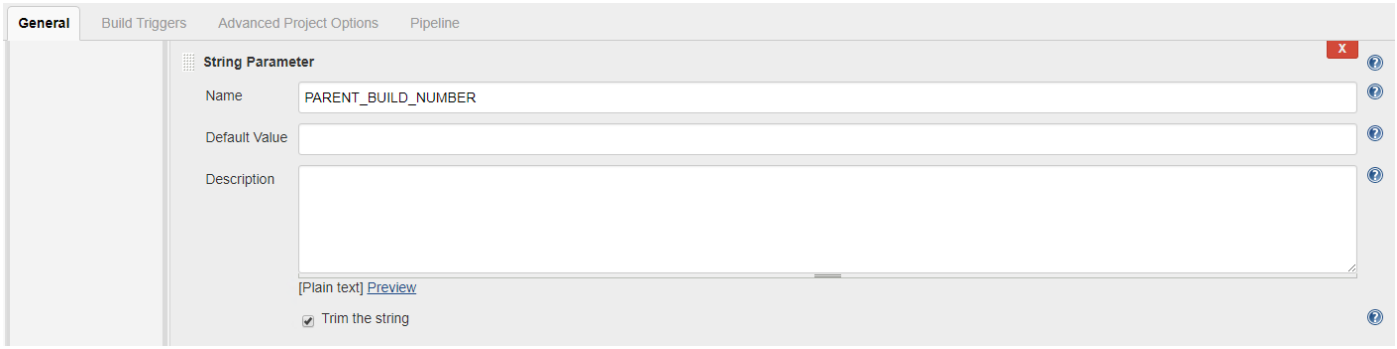
34. Mouseover **PetClinic.Deploy** and click the down arrow. Select **Configure** from the resulting menu.



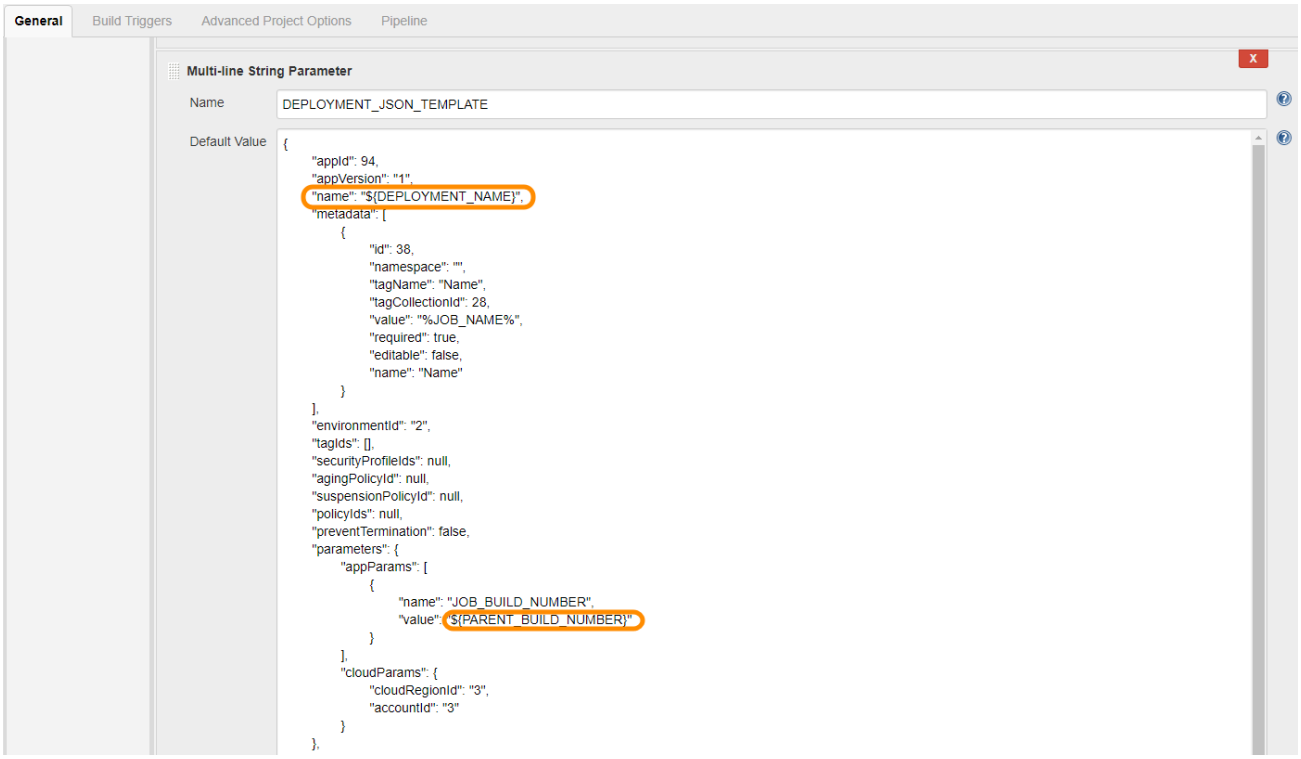
35. Scroll to the This project is parameterized section to observe the parameters used to connect to CloudCenter, namely **CCS_HOST** and **CCS_CREDENTIALS**, as well as the **DEPLOYMENT_NAME** to define the name of the deployment in CloudCenter.



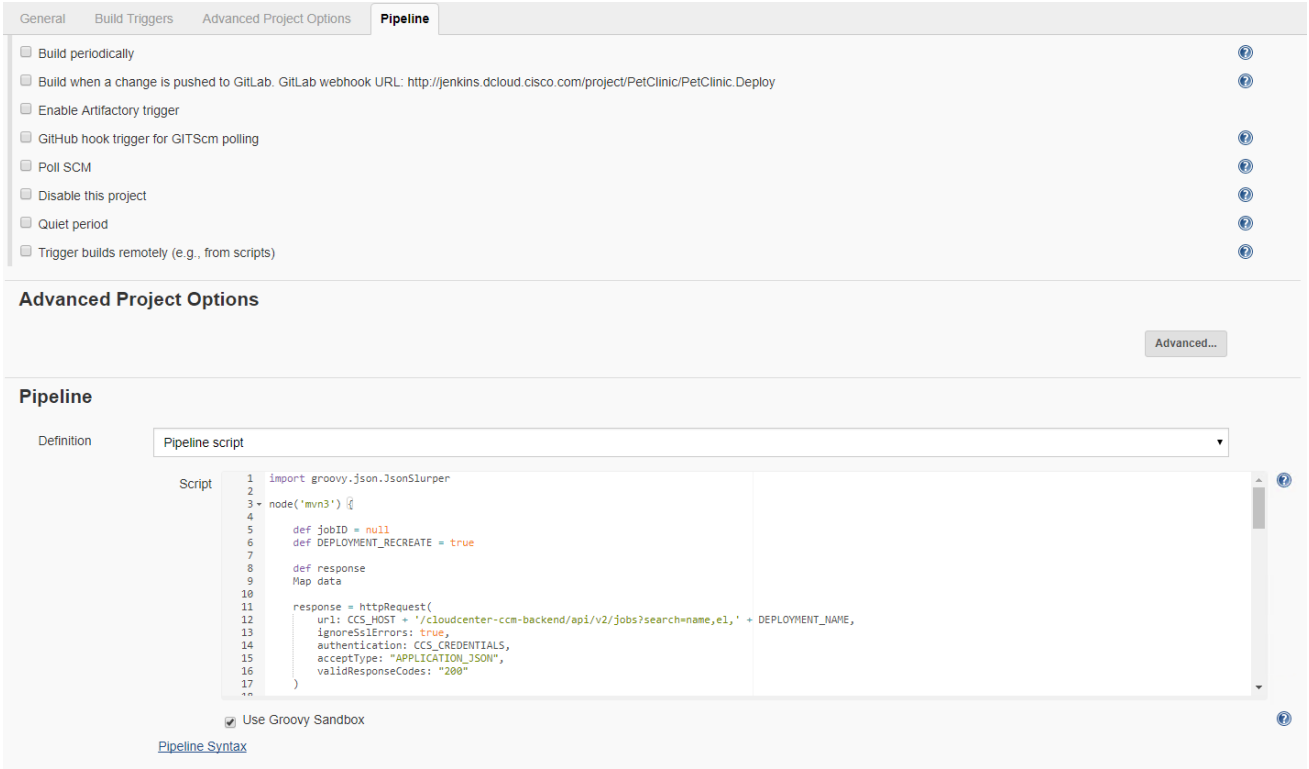
36. Further down the parameters list, observe the **PARENT_BUILD_NUMBER** used to receive the index to the latest compiled java binary.



37. To observe the API template used to initiate the deployment against CloudCenter, scroll to the **DEPLOYMENT_JSON_TEMPLATE** parameter.

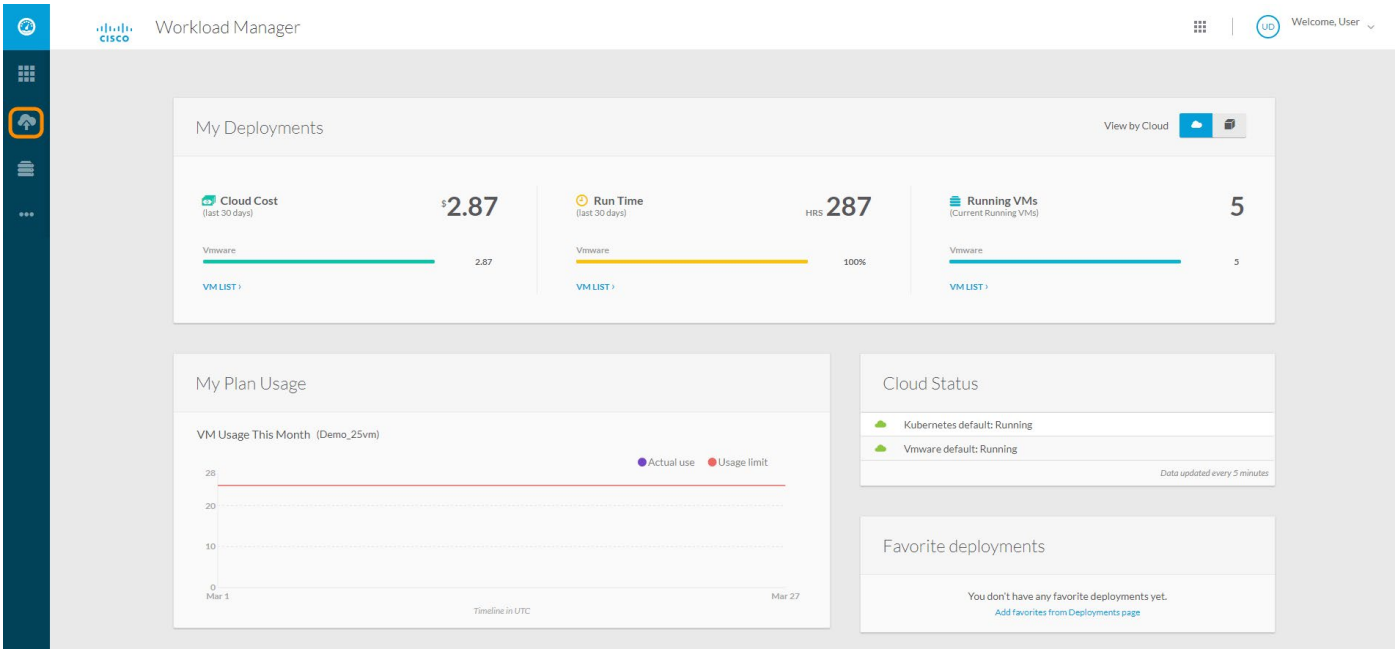


38. To observe the pipeline script used to interface with CloudCenter and manage the deployments, scroll down to the Pipeline section.

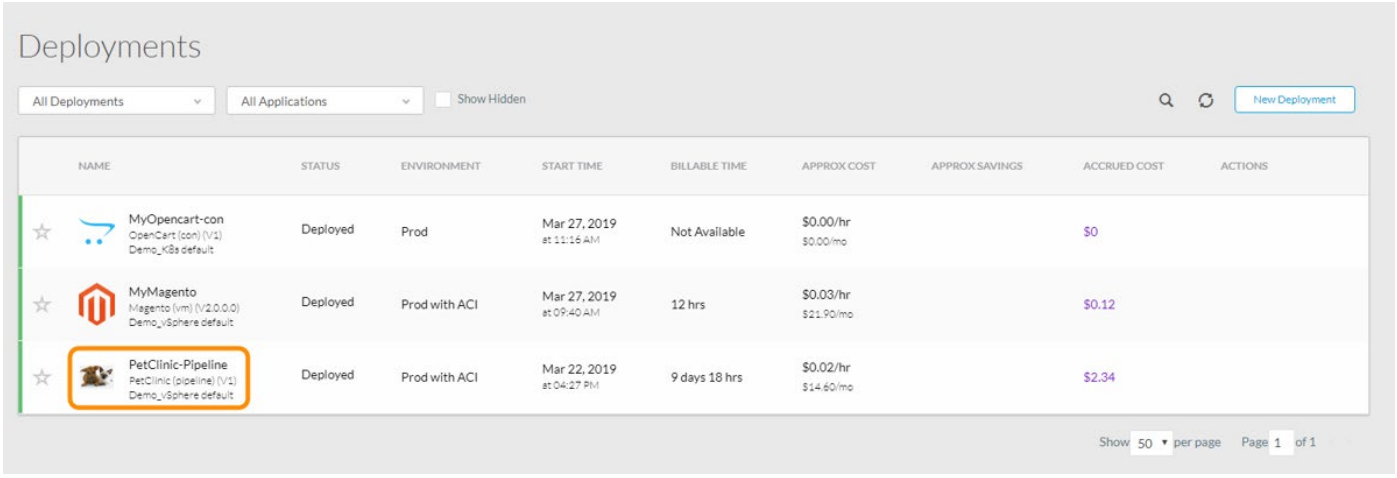


Interact with Application Code

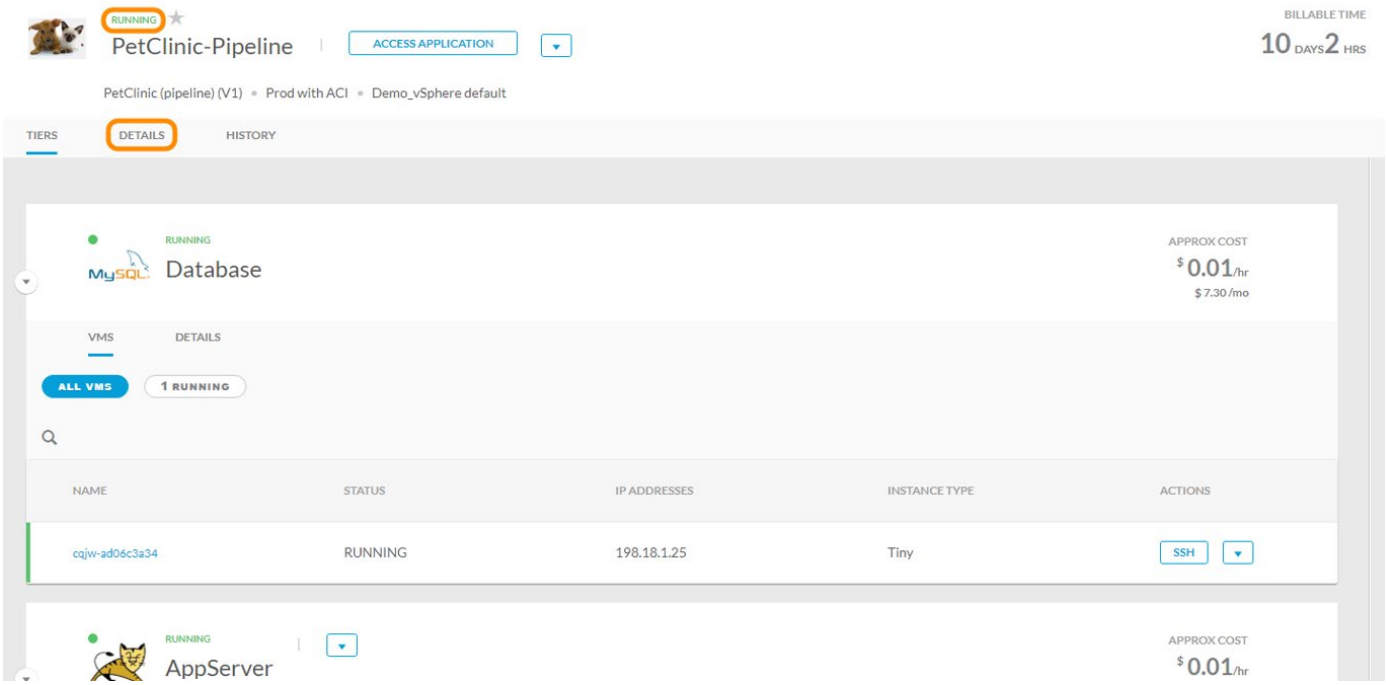
1. Open a **Chrome** browser and perform the following:
    a. Open a new browser tab and click the CloudCenter Suite bookmark. Log in using the credentials **user@dcloud.cisco.com / C1sco12345**, tenant id: **demo**.
    b. Open a second browser tab and click the **GitLab bookmark**. Log in using the credentials **demouser / C1sco12345**.
    c. Open a third browser tab and click the **Jenkins** bookmark. Log in using the credentials **admin / C1sco12345.**
    d. Open a forth browser tab and click the **Artifactory** bookmark. Log in using the credentials **admin / C1sco12345**.
2. Switch to the **CloudCenter** browser tab
3. Select the **Workload Manager** module.
4. Click on **Deployments**.



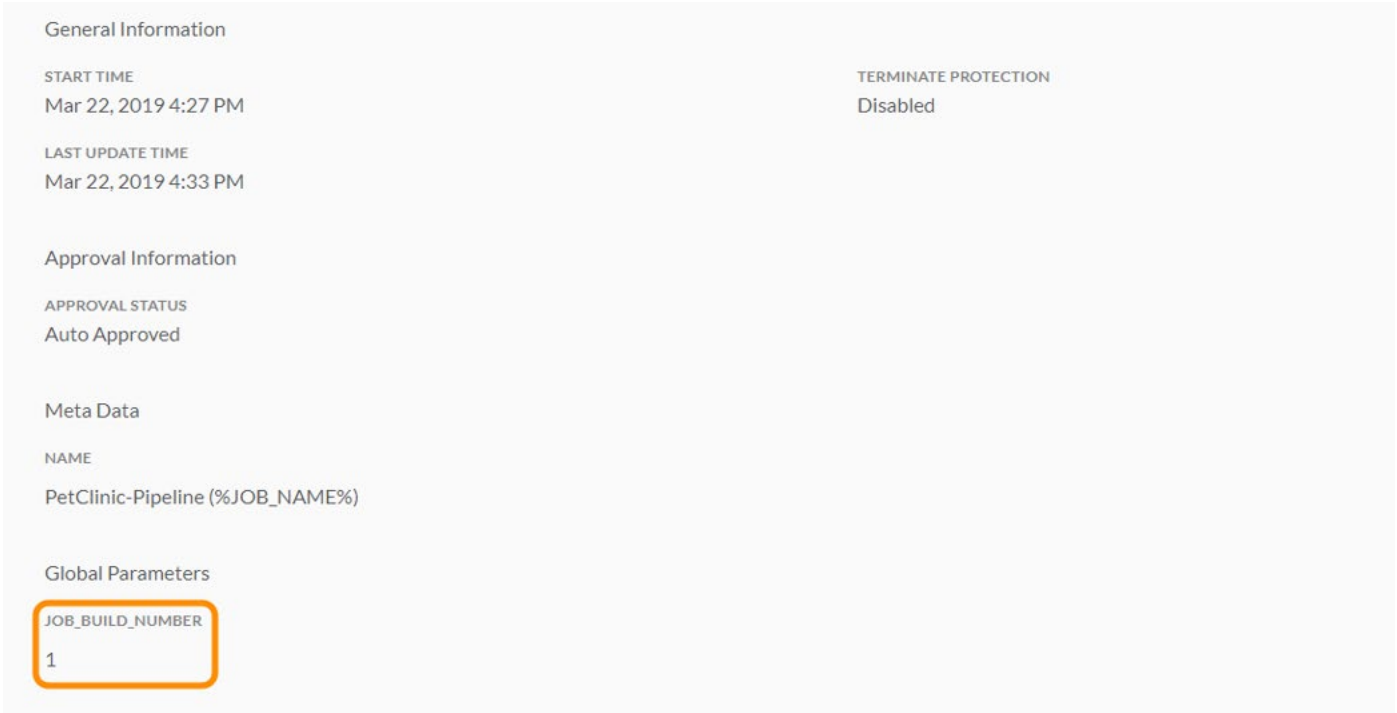
5. From the list of running deployments, locate and click on the **PetClinic-Pipeline** deployment.



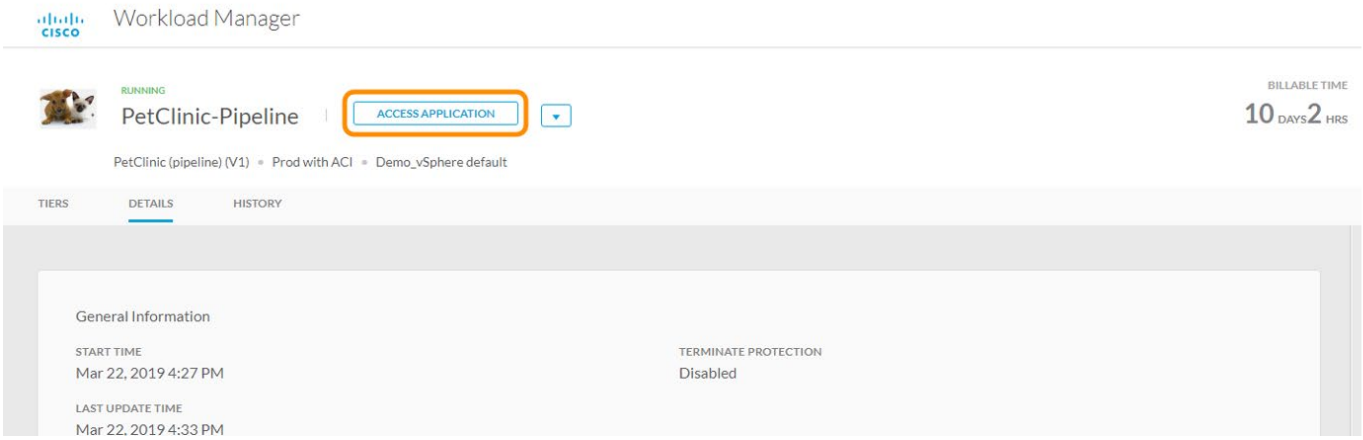
6. Notice the Application already deployed and currently in **Running** state. Click on **Details** at the top of the form



7. Note the **JOB_BUILD_NUMBER** which has the value of 1, pointing to the first compiled artifact having been produced by the CI/CD pipeline and already deployed



8. To access the already deployed application, click on the **Access Application** button at the top of the form

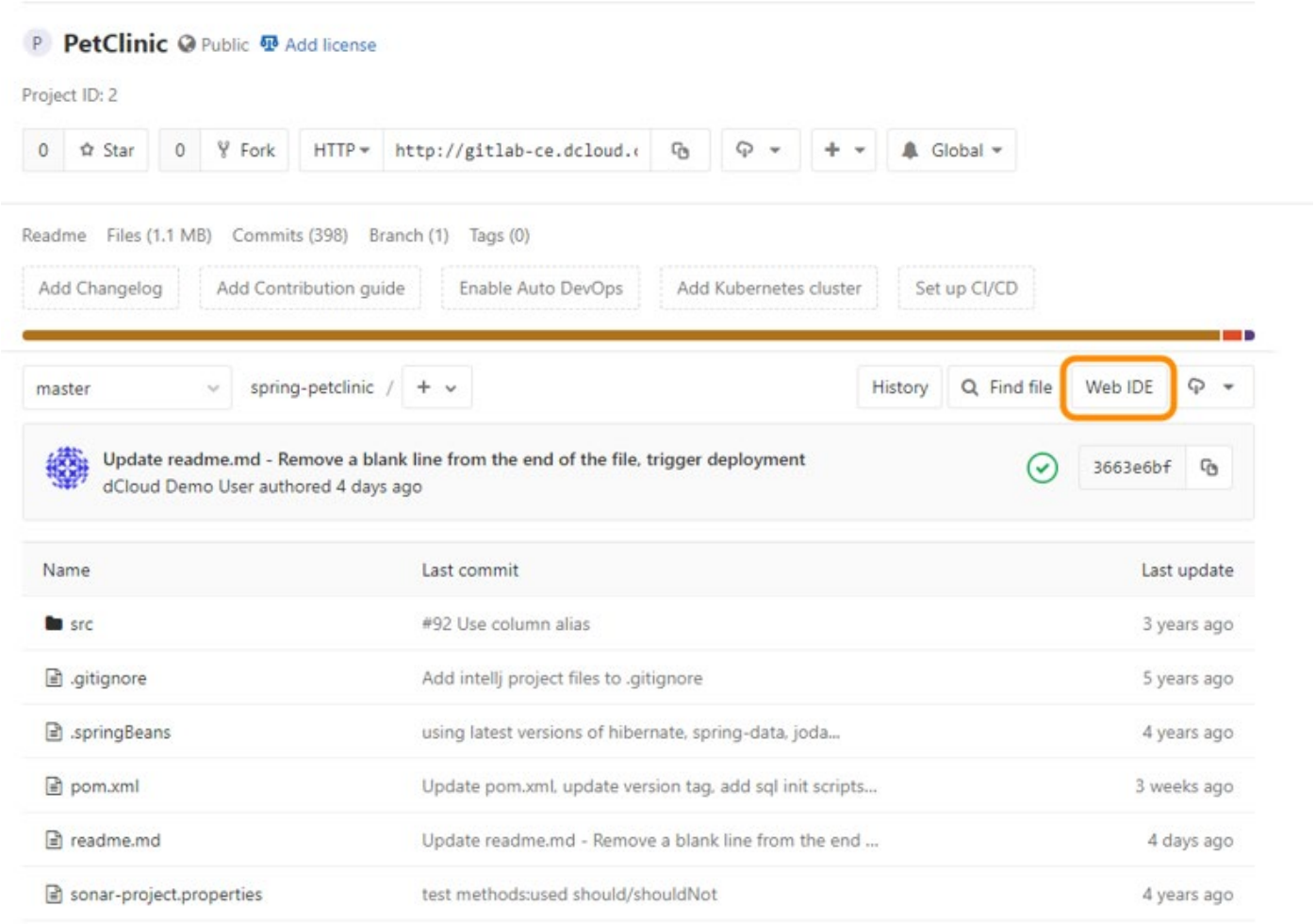


9. Note the Welcome message, which is going to be changed in the following steps by editing the application source code.

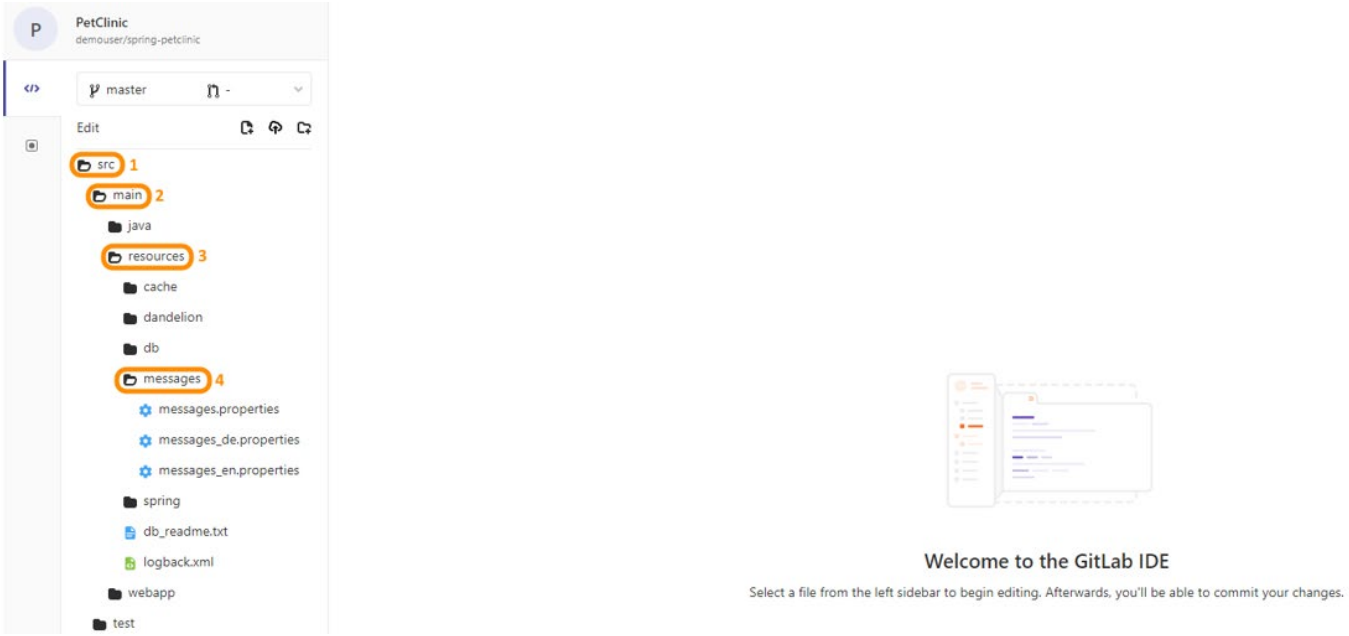
10. Switch to the **GitLab browser** tab

11. Click on the **dCloud Demo User / PetClinic** project.

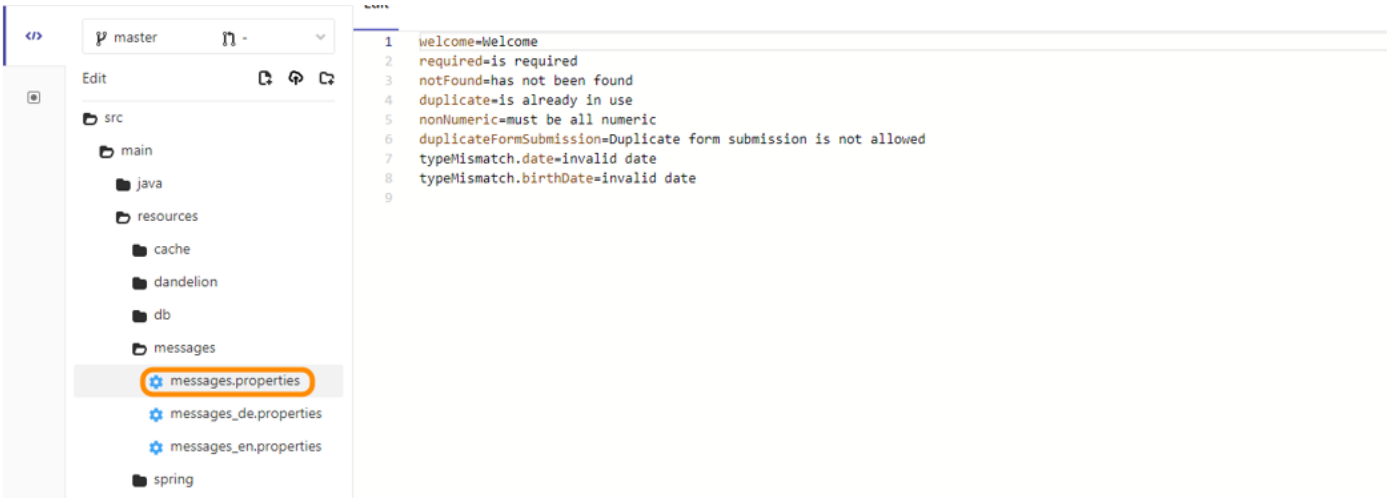
12. To start editing the source code in this git repository using an IDE, click on the **Web IDE** button at the top-right corner.



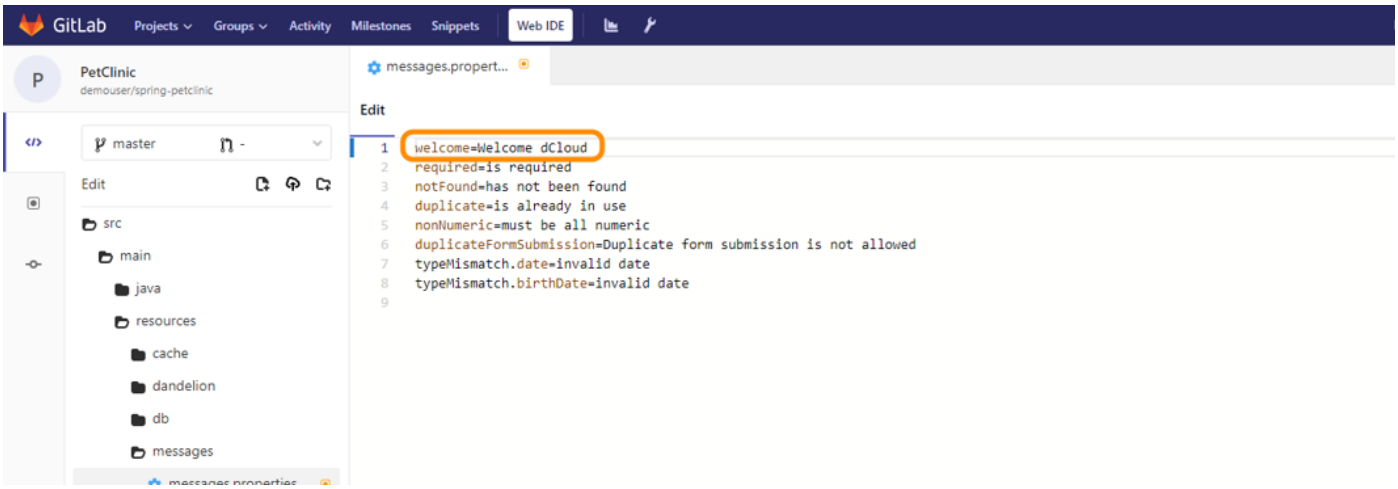
13. Once the Web IDE is up, navigate to **src / main / resources / messages**.



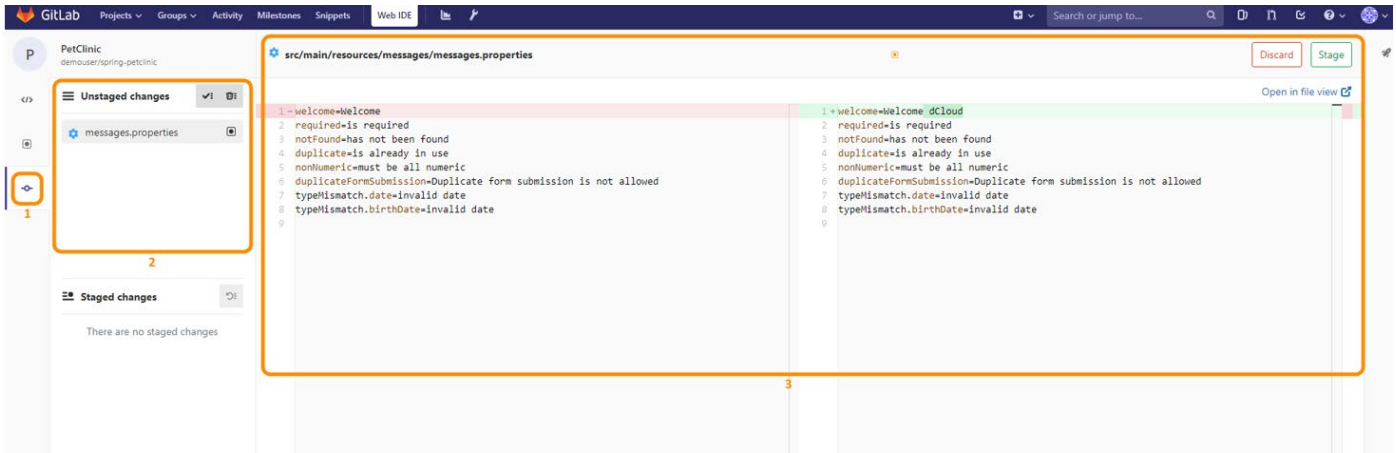
14. Click on the **messages.properties** file. Notice that a file editor has been loaded with the contents of this file.



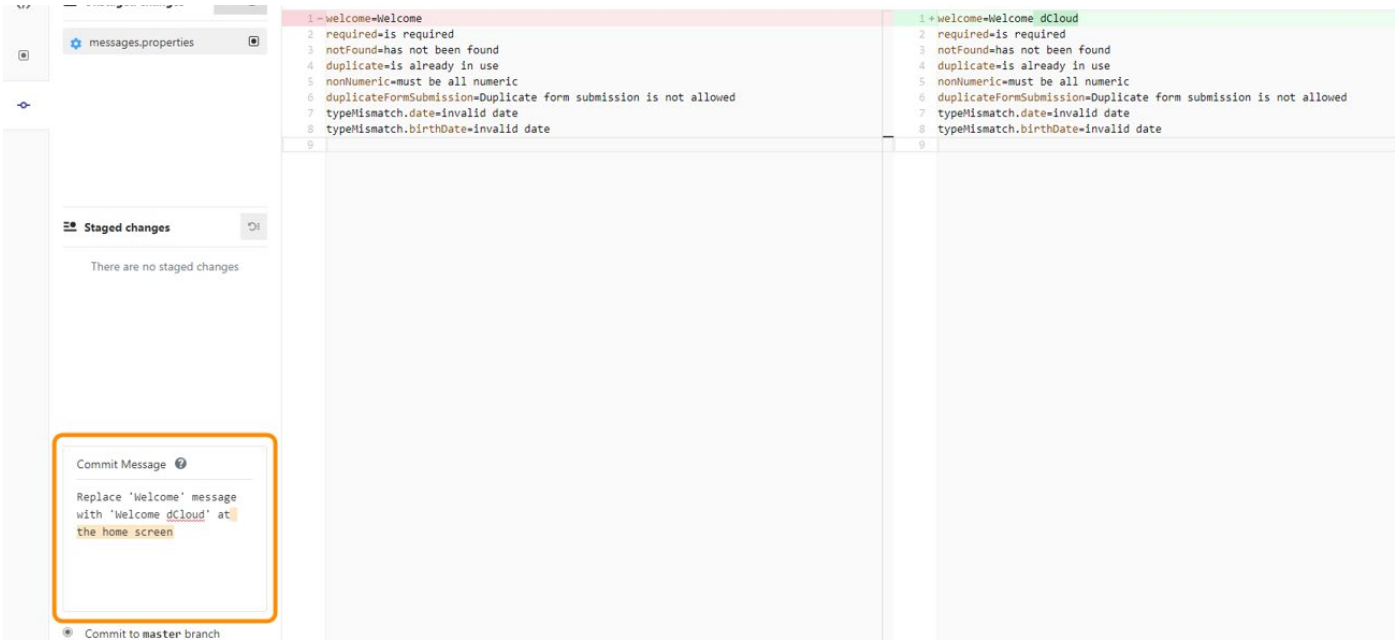
15. Change the **Welcome=Welcome** string to **Welcome=Welcome dCloud**. Notice that the Commit... button at the bottom-left corner has been enabled, indicating there are changes pending in this commit. Do <u>not</u> press the Commit... button at this stage.



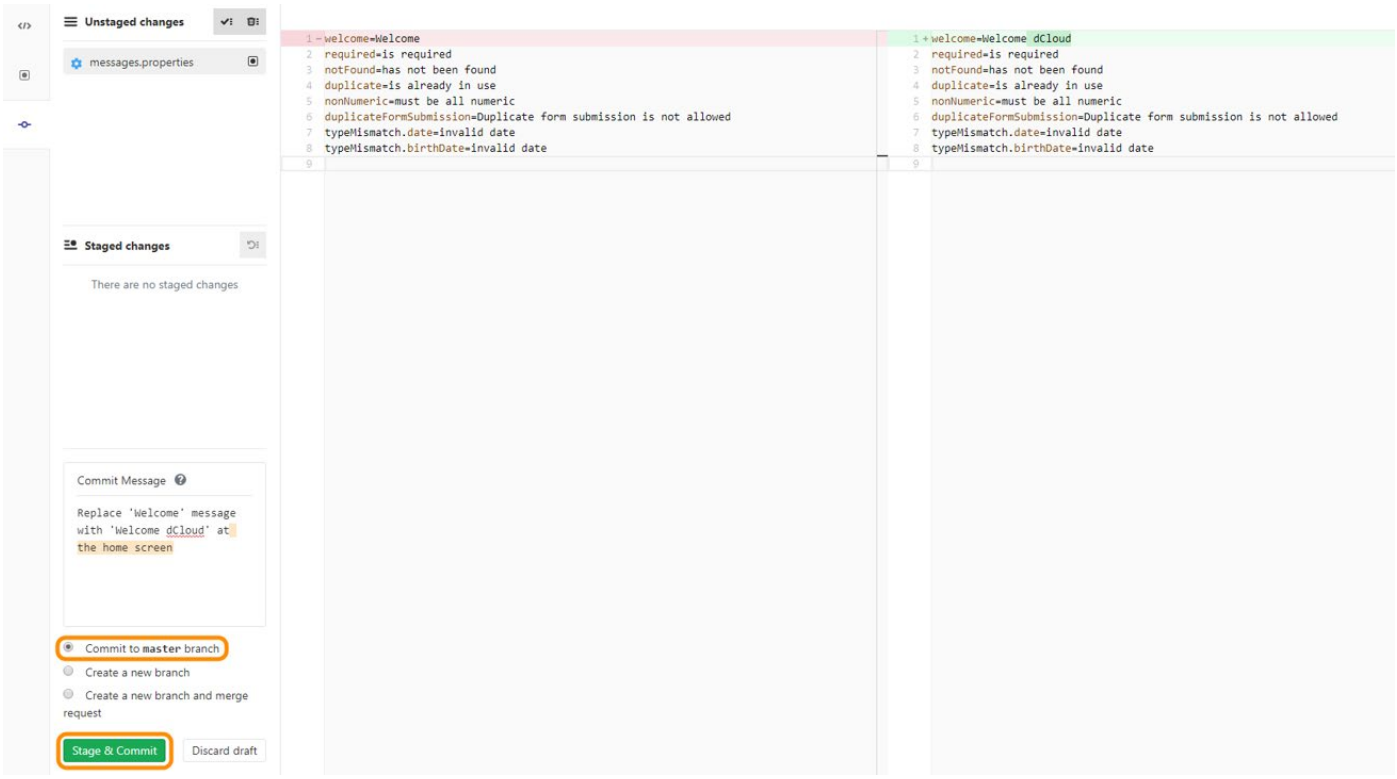
16. At the left-hand side, click on the **Commit** icon to view the changed files in this commit and review the changes for each file in this commit. In this case, only one file has changed, the **messages.properties** file.



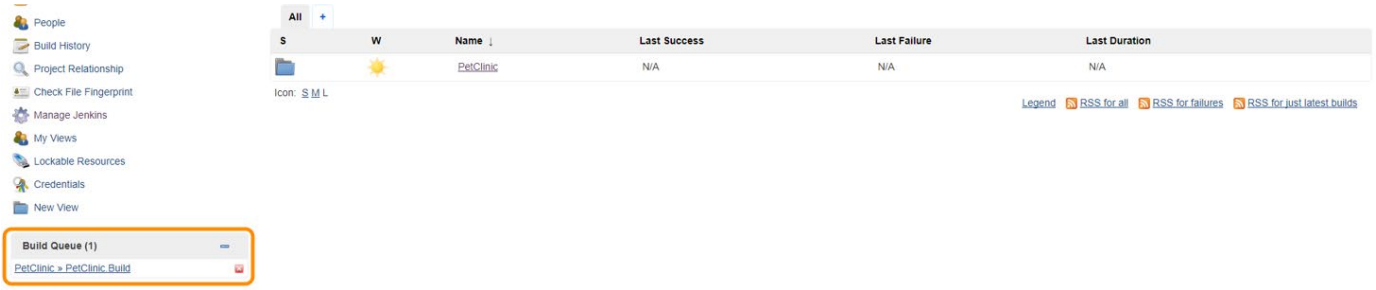
17. At the **Commit Message** text box, enter the text **Replace Welcome message** with **Welcome dCloud** at the home screen.



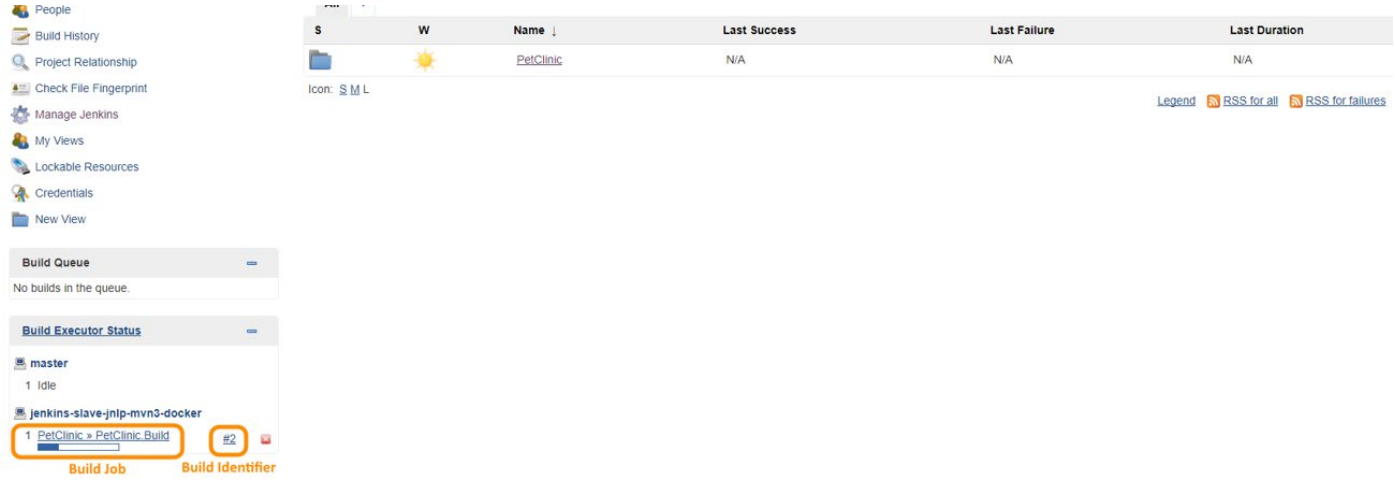
18. Next, leave the default setting to **Commit** to master branch and click on the **Stage & Commit** button at the bottom of the page. At the stage, GitLab will notify Jenkins there is a new build pending.



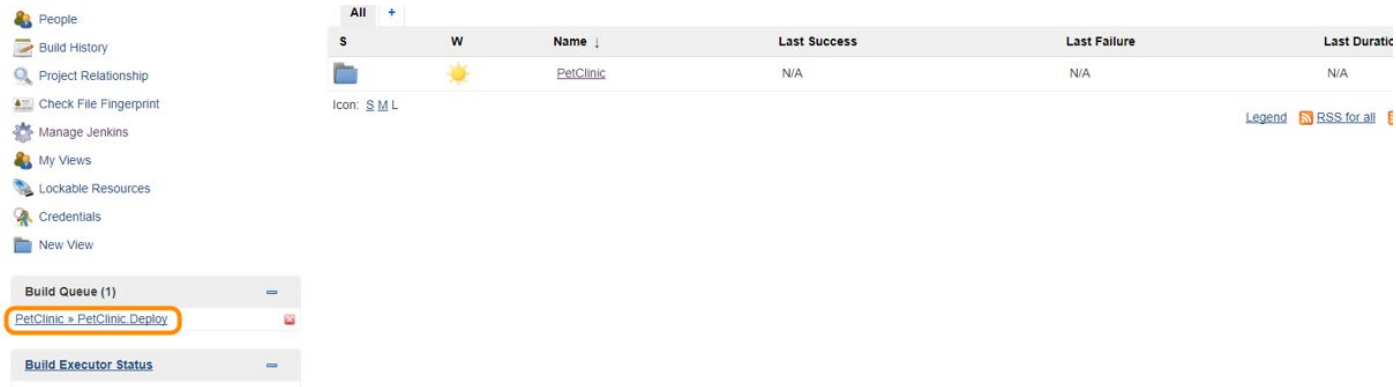
19. Switch to the **Jenkins browser** tab to observe the relevant jobs being executed. First, the **PetClinic/PetClinic.Build** job will be queued



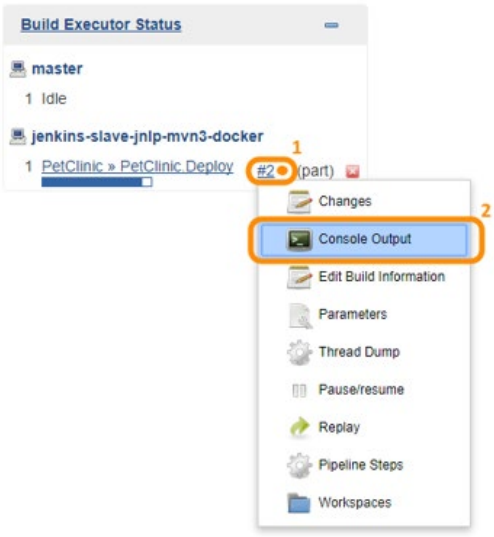
20. *[Observe-only step]* After a few seconds, the **PetClinic/PetClinic.Build** job will be executed at the available Jenkins slave node. Please note that the Build number next to the build job represents the identifier that is passed to the subsequent tasks (only if the build is successful), which points to the new java artifact(s) to be deployed.



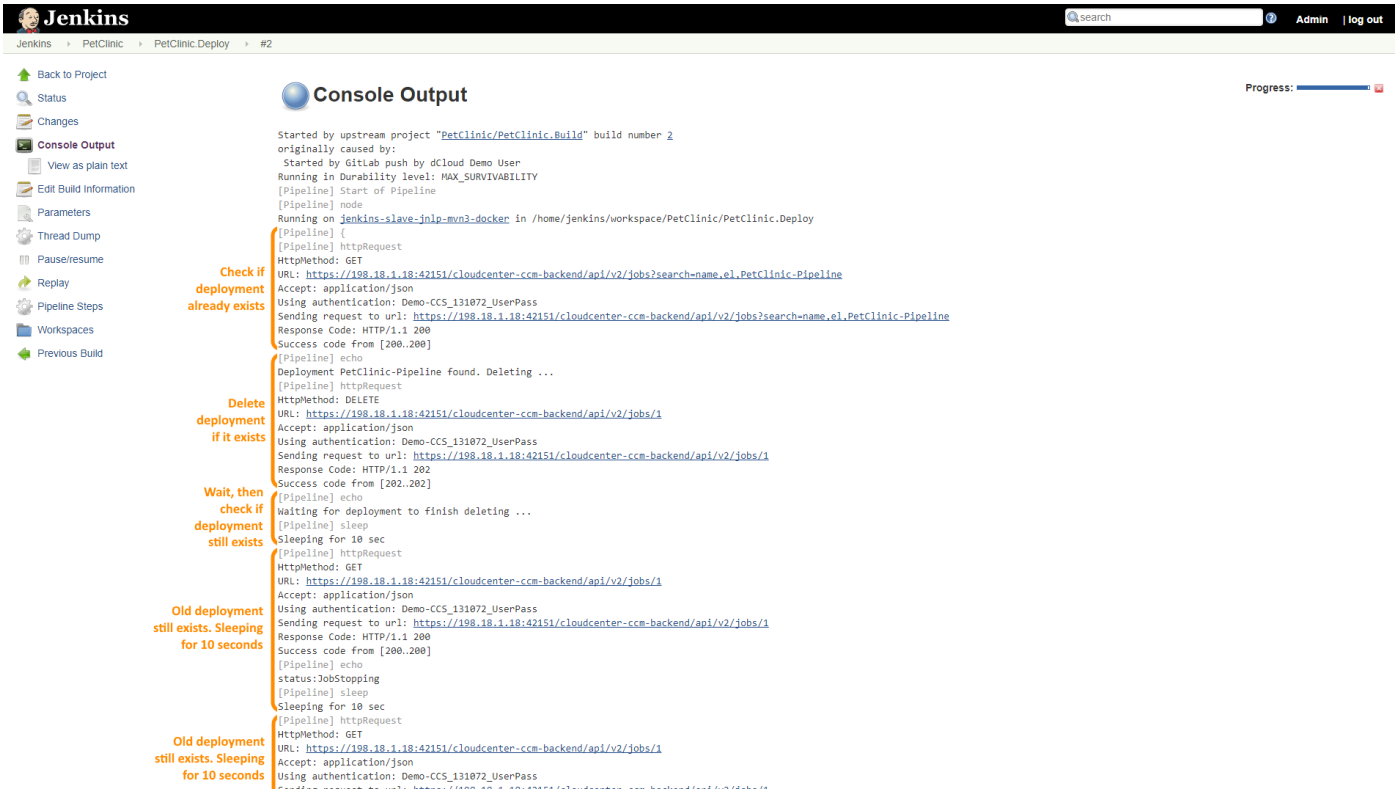
21. *[Observe-only step]* Once the PetClinic/PetClinic.Build job is complete (if successful), the PetClinic/PetClinic.Deploy job is queued for execution.

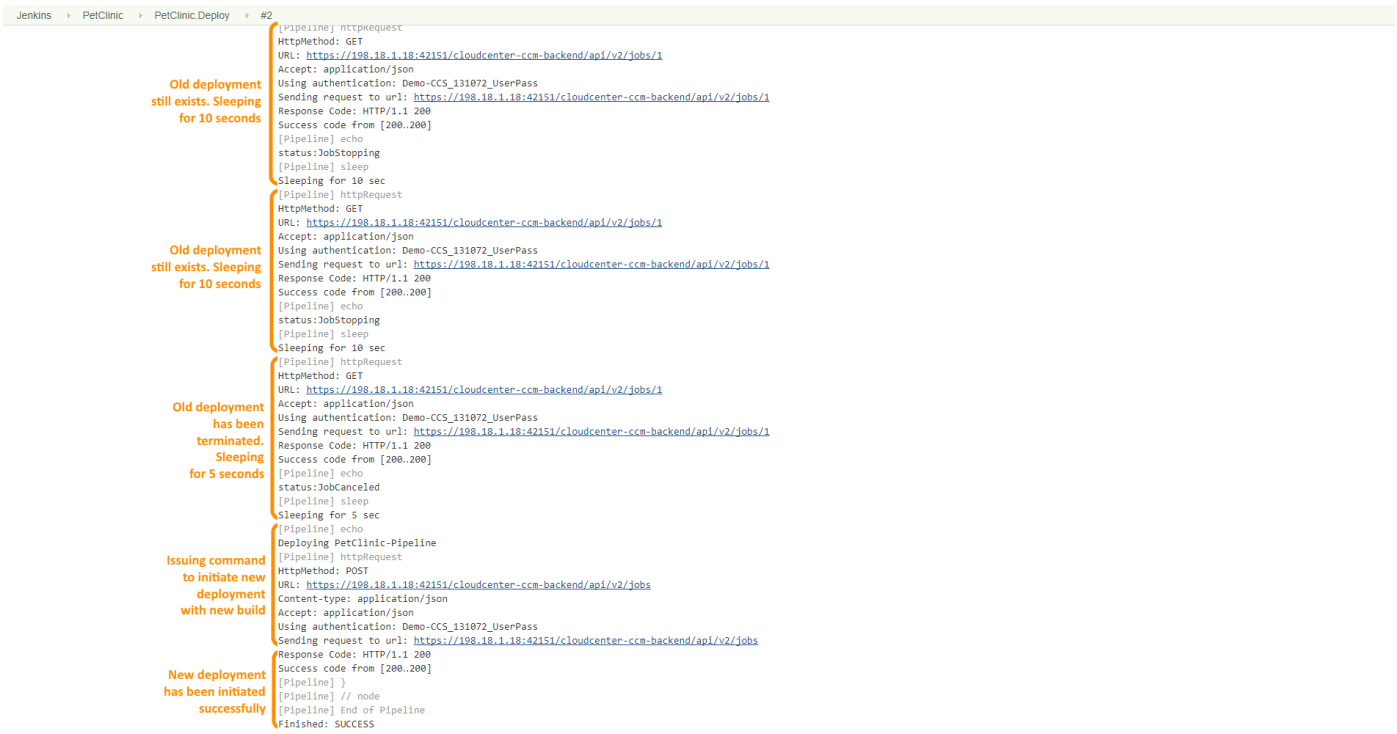


22. After a few seconds, the **PetClinic/PetClinic.Deploy** job will be executed at the available Jenkins slave node. Once the job enters the execution phase, **click on the small down arrow** next to the job number and click on **Console Output** to observe the progress of the job.



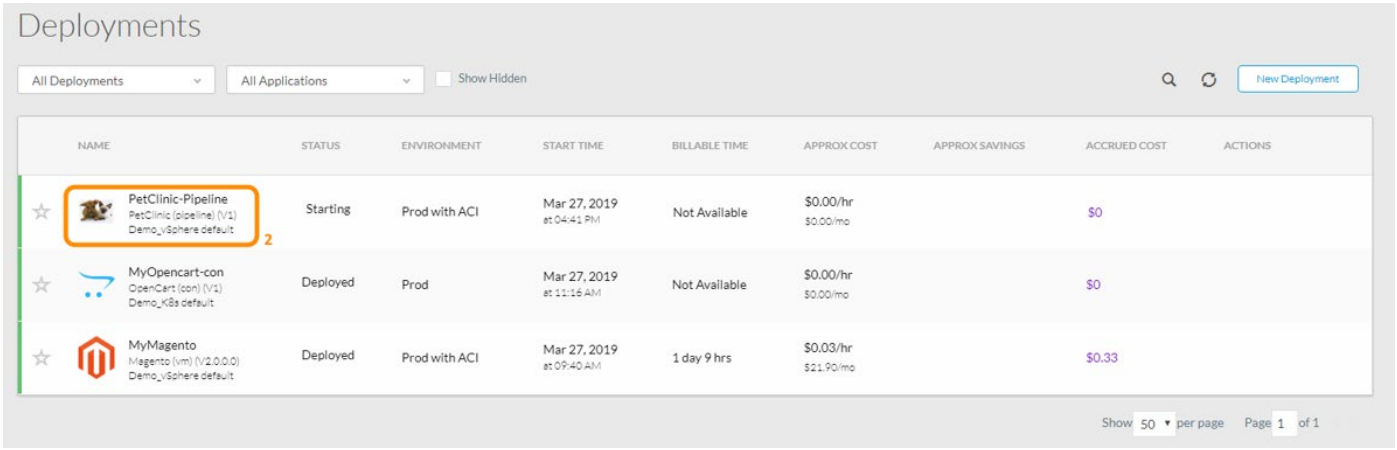
23. Observe the output logs of this job while executed (comments have been added to the output logs in order to explain the tasks being executed).

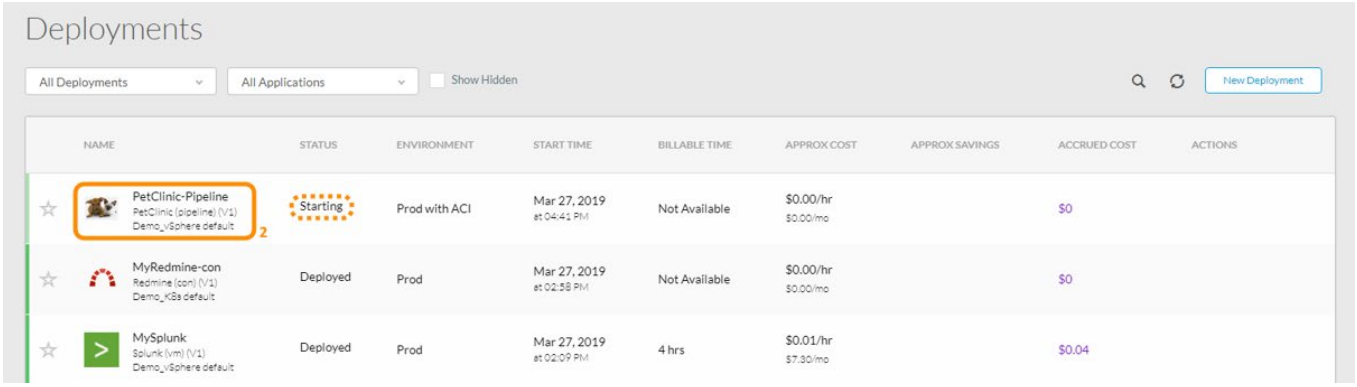




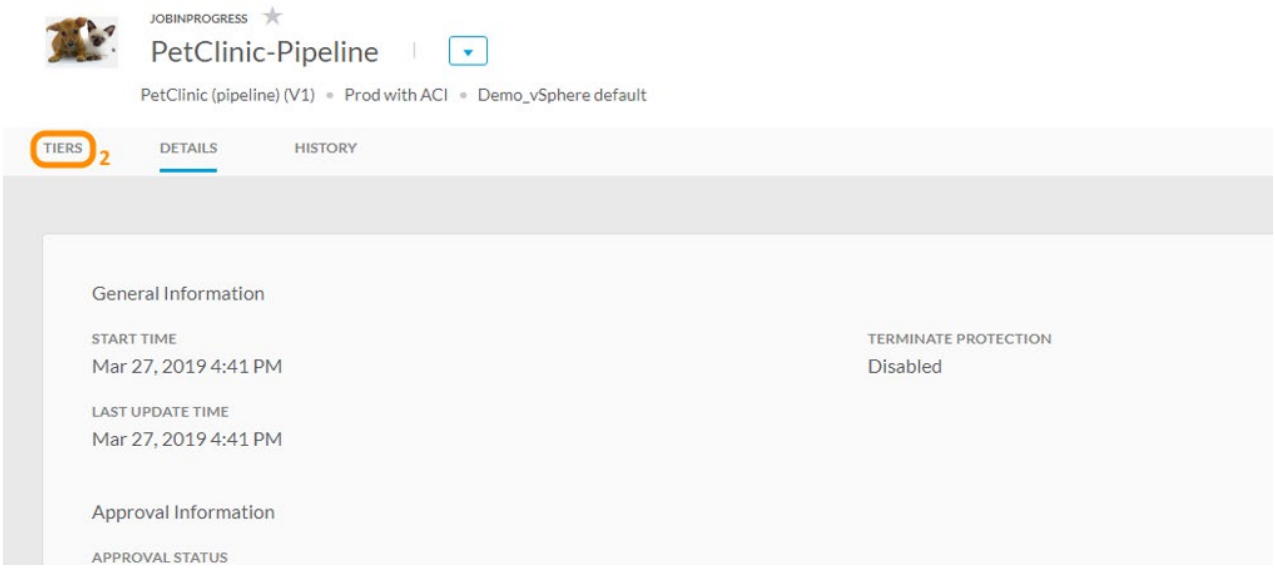
24. Once the new deployment request has been submitted successfully to CloudCenter, switch to the CloudCenter browser tab.

25. At the left-hand side, click on **Deployments** to observe the new deployment being rolled out. Notice the new PetClinic-Pipeline deployment being rolled out. Click on the newly created deployment to view the deployment details.

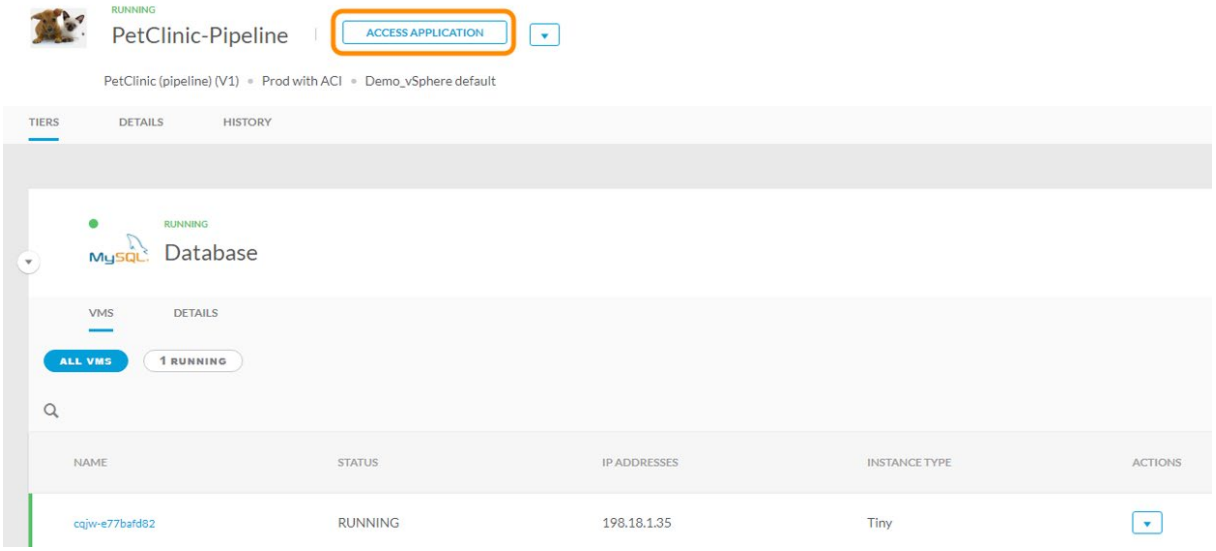




26. Click on the **Details** tab and observe the **JOB_BUILD_NUMBER** having now changed to **2**, indicating the new build being used for roll-out. Click on the **Tiers** tab to return to the previous screen. Wait for the new deployment to fully deploy and then proceed to the next step (5-10min).

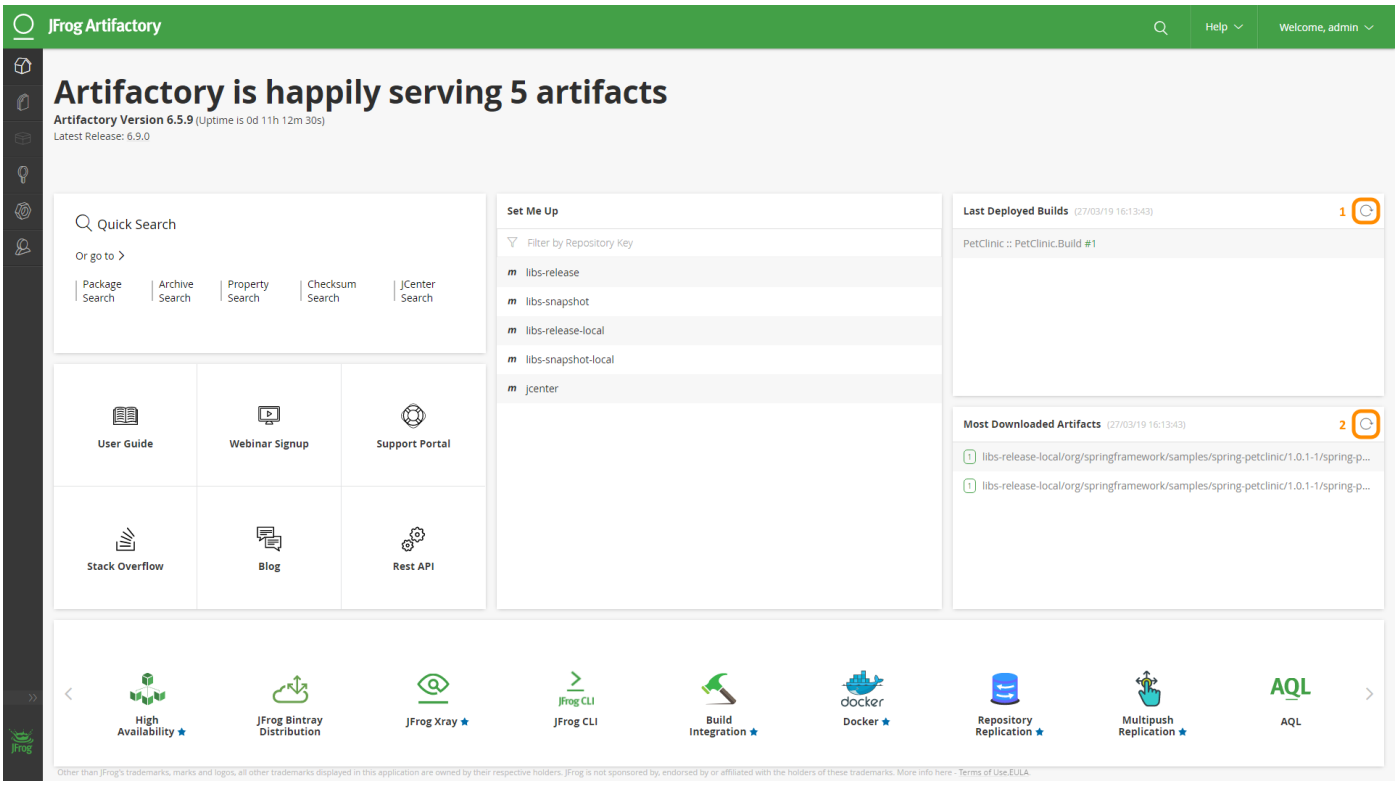


27. Once the deployment completes, click on the **Access Application** button.



28. You should witness the Welcome message at the home screen having changed to Welcome dCloud.

29. To observe the newly compiled artifacts and their repository location, switch to the Artifactory browser tab. The Web GUI should still be showing the old artifacts. Click on the **refresh buttons** next to Last Deployed Builds and Most Downloaded Artifacts.



30. After refreshing the two frames, the new artifacts should become visible at the web page.