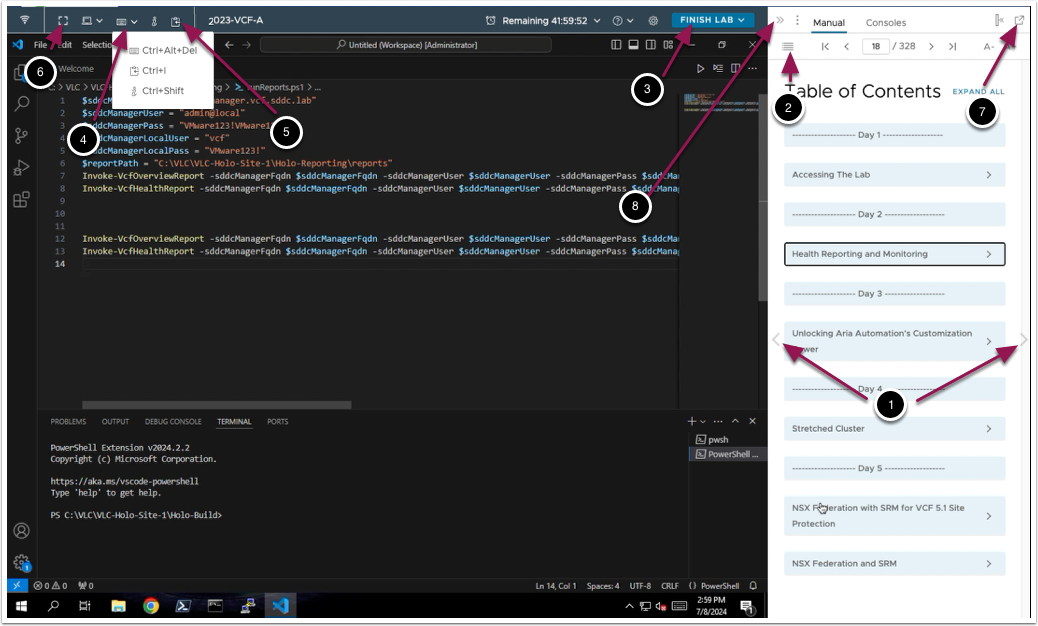
Lab Manual

## Default chapter

* Lab Guidance

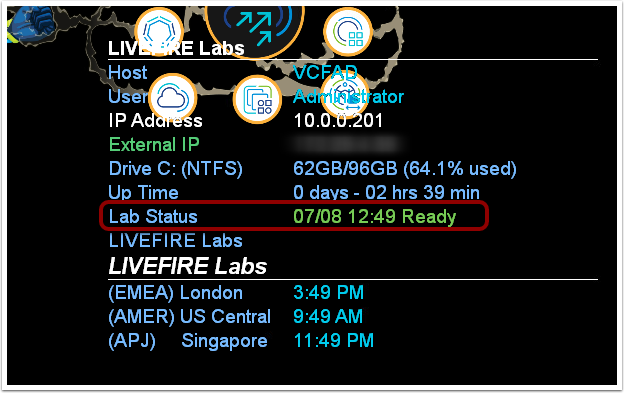
## Lab Guidance



From here you can:

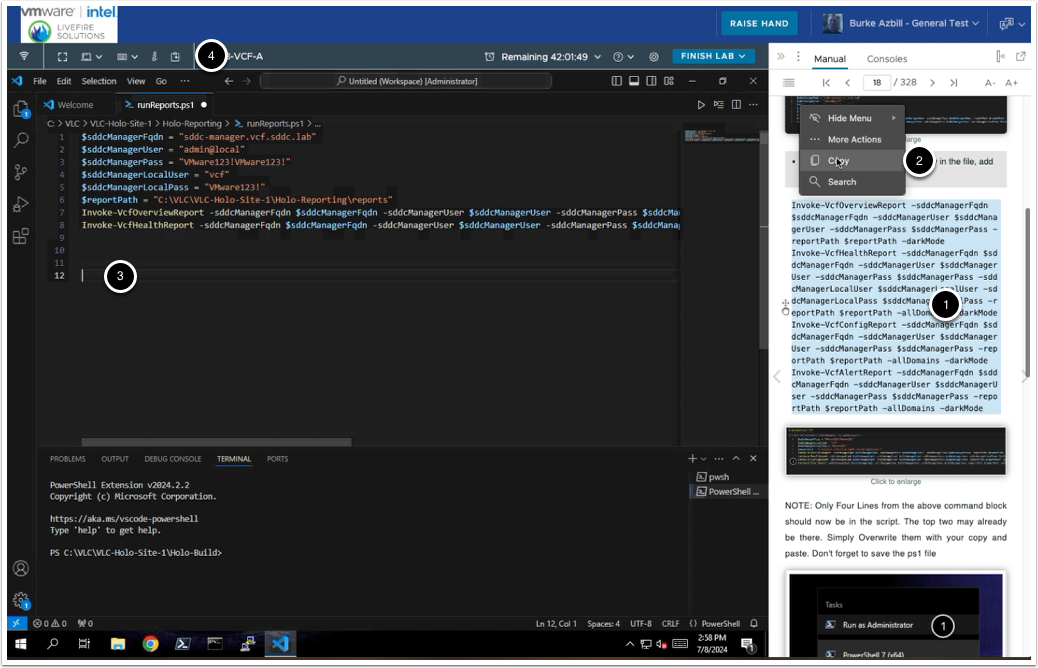
1. Click to advance to the next page and continue with the next lab module
2. Open the TABLE OF CONTENTS to jump to any module or lesson in this lab manual
3. Exit your lab and come back and resume it again in the future
4. Send Ctrl+Alt+Del to the console
5. Send Text into (Paste) the console
6. Enter Full-Screen
7. Detach the Lab Manual
8. Collapse the Lab Manual

## You are ready....is your lab?



The lab console will indicate when your lab has finished all the startup routines and is ready for you to start. If you see anything other than "Ready", please wait for the status to update.  If after 5 minutes your lab has not changed to "Ready", please ask for assistance.

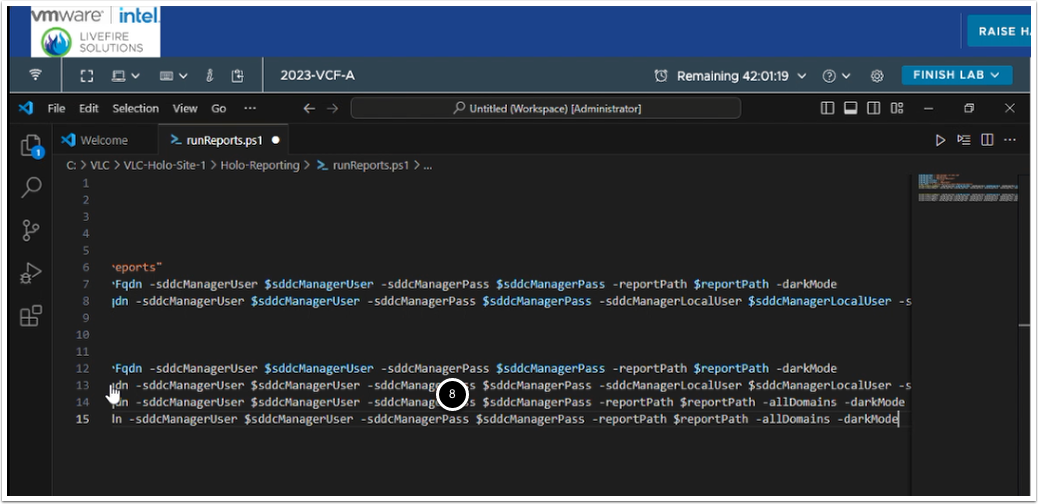
## How to Copy from manual and Paste into Console



1. Use your mouse to select the text you wish to copy
2. When you release the mouse button to complete your selection, a context menu shows up - Click CopyNOTE: Right click on selected text DOES NOT bring up a copy option and CTRL+C/CMD+C does not work
3. Place your cursor/prompt at the location you wish to paste inside the console
4. Click the Clipboard icon to bring up the "Send text to console" window



1. Paste the contents of your clipboard into the window using Right-Click -> Paste, or your OS Keyboard combination of either CTRL+V or CMD+V
2. Click the "SEND" button to send the text to the console
3. Click the "CANCEL" button when you are done sending text to the console. This will close the popup window



1. Confirm that your pasted text is at the desired location

## Cloud Services Access

* Cloud Services Access

## Cloud Services Access

You are now ready to dive into the labs!  READY, SET, GO!

## Getting Started with Cloud Assembler

* Configure Cloud Services
* 01. Setting up the Blueprints

## Configure Cloud Services

## 1. Add Folders for Resource Pools

Let's begin by opening a browser window to https://vcenter-mgmt.vcf.sddc.lab/ and logging in.

Enter the following to log in:

Username: administrator@vsphere.local

Password:  VMware123!VMware123!



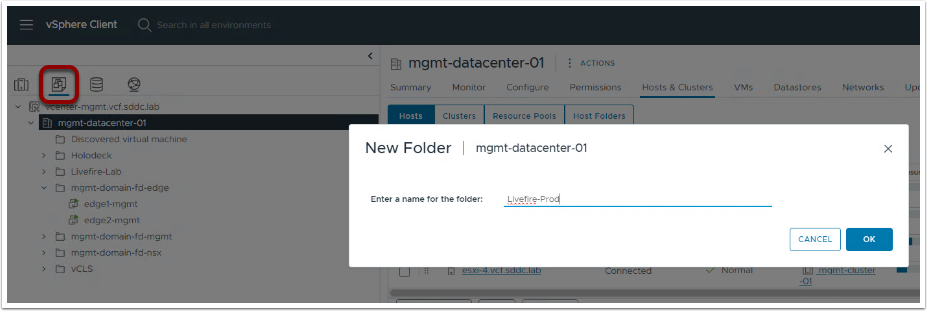
1. Under the VM and folders tab , under vcenter-mgmt.vcf.sdd.lab, click on the cluster - mgmt-datacenter-01-01 and right click

2. Click on New Folder > New VM and Template Folder

3. Enter Livefire-Prod

4. Click on OK to save

5. Repeat for Livefire-Dev and Livefire-Test

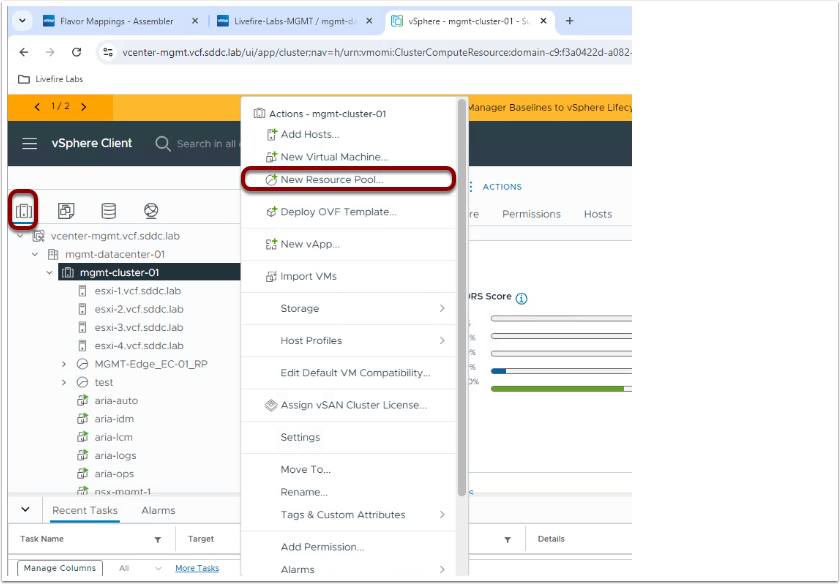


## 1.1 Create Resource Pools

1. On the Server Tab

2. Under vcenter-mgmt.vcf.sddc.lab > mgmt-datatcenter-01, click on the cluster - mgmt-cluster-01 and right click

2. Click on New Resource Pool



21.  Enter the following:

Name:  Name:  Livefire-Test-RP

CPU

Shares:  Normal

Reservation:  12000 MHz CPU Shares MHz

Reservation Type: Check Expandable

Limit: Unlimited MHz

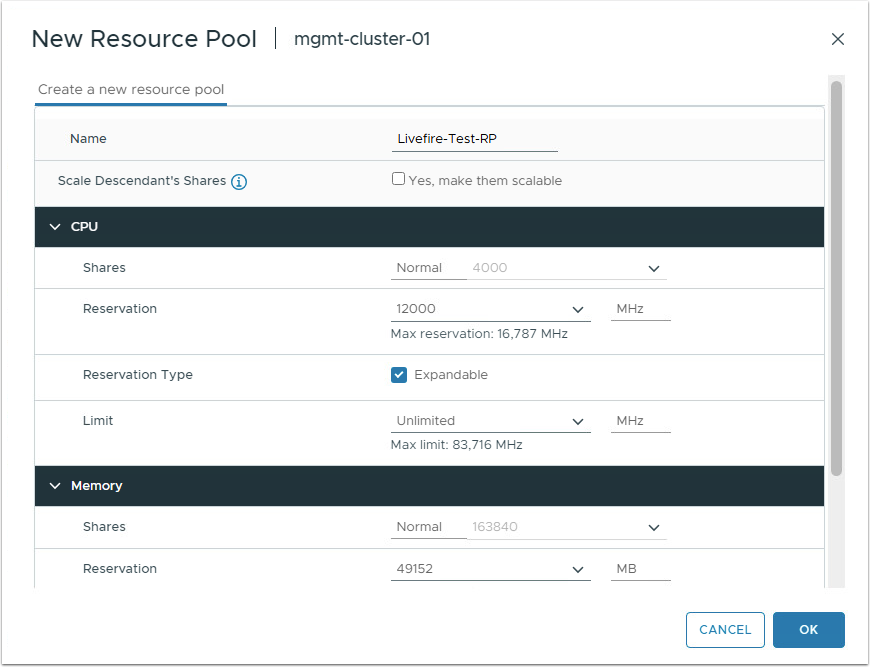
Memory

Shares:  Normal

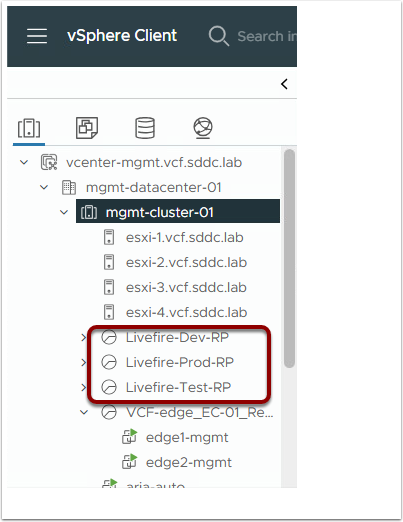
Reservation:  49152 MB

Limit: Unlimited MB

Reservation Type: Check Expandable



Confirm that you see the newly created resource pool on the left side of the vSphere Client window



Repeat steps and create an additional resource group for Dev each using the following:

Name:  Livefire-Dev-RP

CPU

Shares:  Normal

Reservation:  12000 MHz CPU Shares MHz

Reservation Type: Check Expandable

Limit: Unlimited MHz

Memory

Shares:  Normal

Reservation:  49152 MB

Limit: Unlimited MB

Reservation Type: Check Expandable

Repeat steps and create an additional resource group for Prod each using the following:

Name:  Livefire-Prod-RP

CPU

Shares:  Normal

Reservation:  12000 MHz CPU Shares MHz

Reservation Type: Check Expandable

Limit: Unlimited MHz

Memory

Shares:  Normal

Reservation:  49152 MB

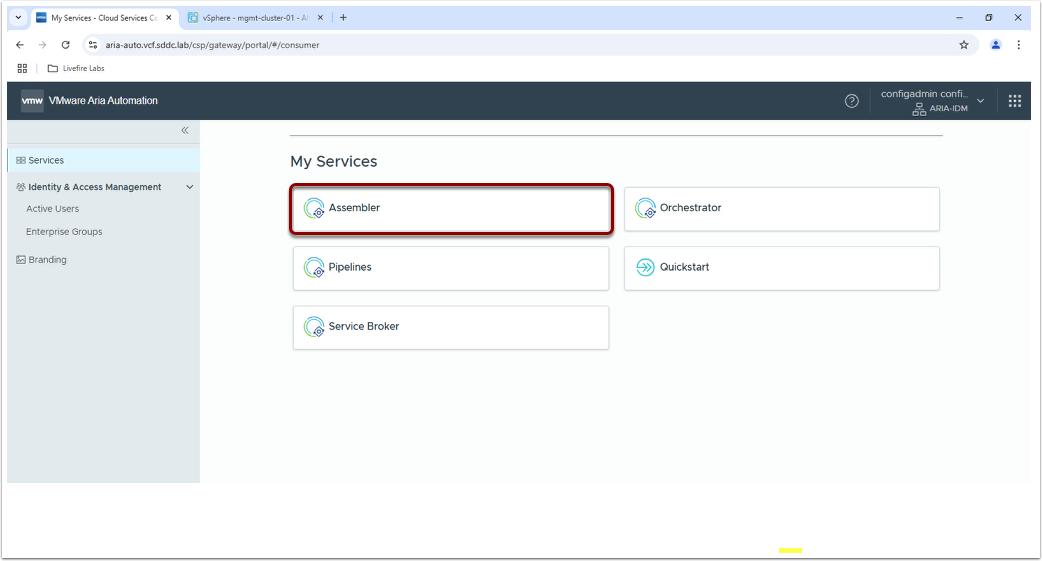
Limit: Unlimited MB

Reservation Type: Check Expandable

## 2. Setting Up the Cloud Assembly Configuration

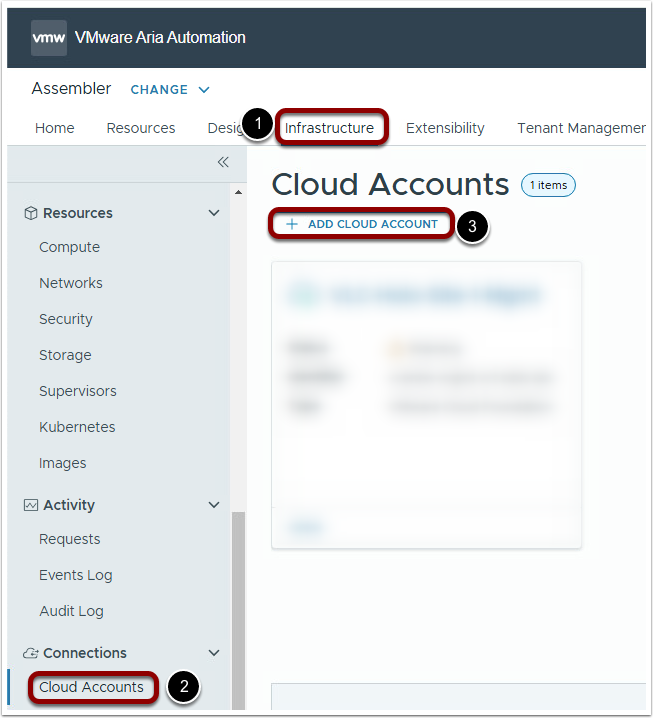
Let's begin by opening a browser window to https://aria-auto.vcf.sddc.lab/ and logging in.

1. Start by opening Assembler under My Services.
2. Enter the following to log in:
3. Username: configadmin
4. Password:  VMware123!

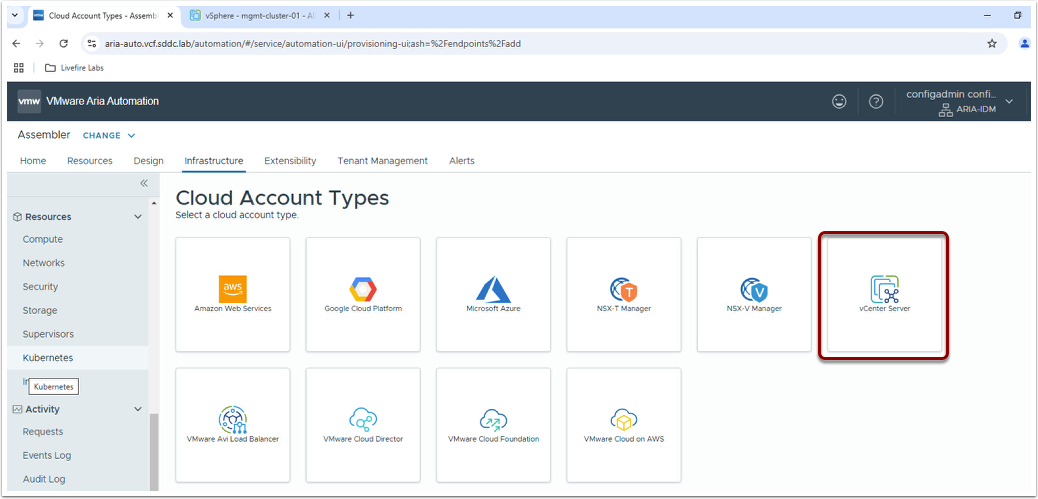


## 3. Setup Cloud Accounts

1. Click on Infrastructure
2. Click on Cloud Accounts
3. Click ADD CLOUD ACCOUNT



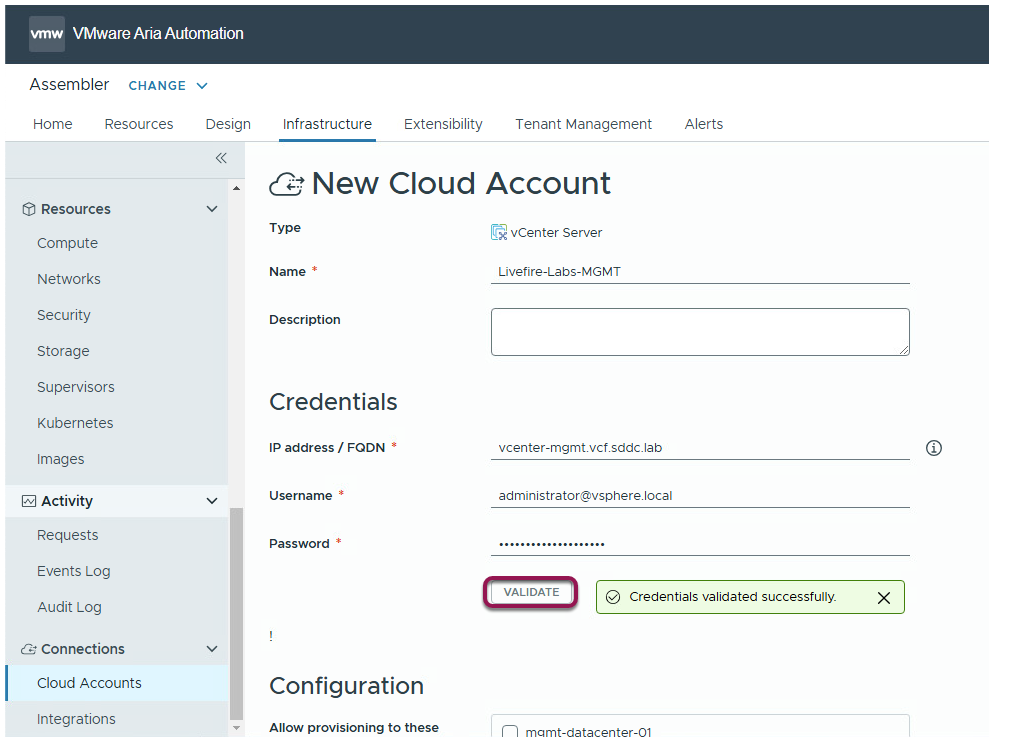
4. Click on vCenter Server



5. Enter the following:

1. Name:  Livefire-Labs-MGMT
2. IP address / FQDN:  vcenter-mgmt.vcf.sddc.lab
3. Username:  administrator@vsphere.local
4. Password: VMware123!VMware123!

6. Click VALIDATE



7. Proceed to the Configuration section

8. Once validated, more configuration options will appear

9. Under the Configuration section, enter the following:

    a. Allow Provisioning to these datacenters: Check the mgmt-datacenter-01

    b.  Do NOT select the "Create a cloud zone for the selected datacenters"...we will do this manually

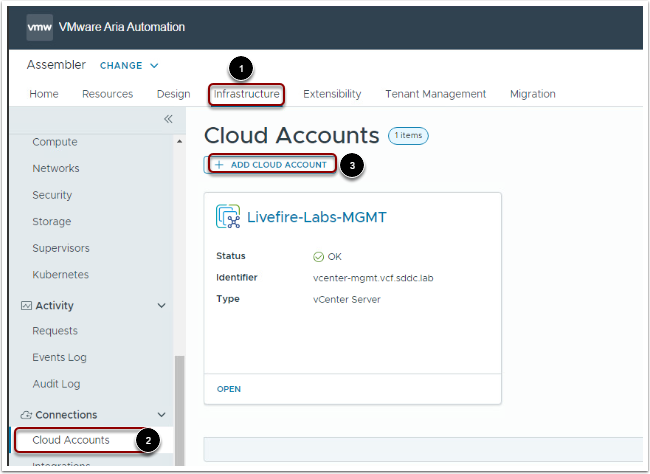


10. Click ADD add the bottom to create the new Cloud Account Configuration

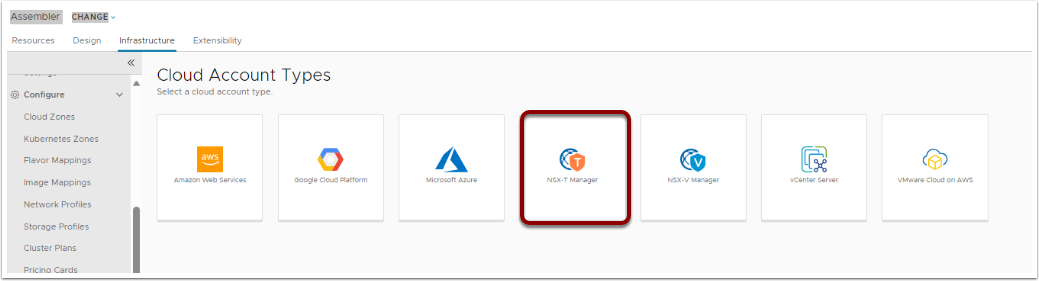


## 3.1 Integrate NSX-T into Cloud Services

1. Click on Infrastructure
2. Click on Cloud Accounts
3. Click ADD CLOUD ACCOUNT

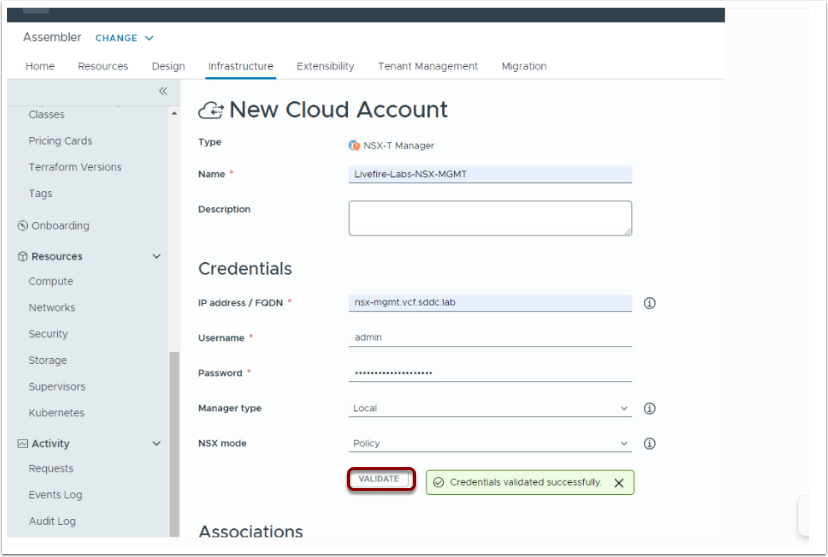


4. Click on NSX-T Manager



5. Enter the following:

1. Name:  Livefire-Labs-NSX-MGMT
2. IP address / FQDN: nsx-mgmt.vcf.sddc.lab
3. Username:  admin
4. Password: VMware123!VMware123!
5. Manager type: Local
6. NSX mode: Policy
7. Click on Validate

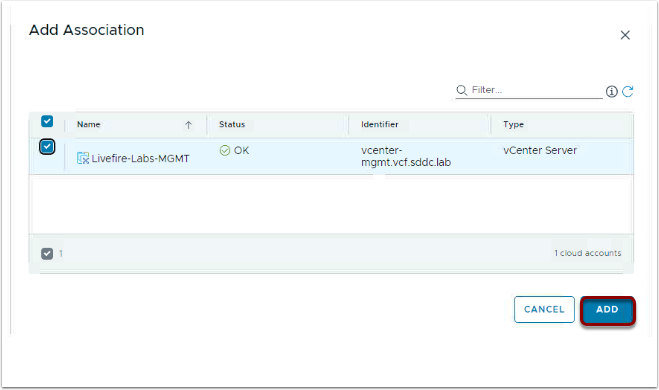


7. Add the associated Cloud account by clicking ADD

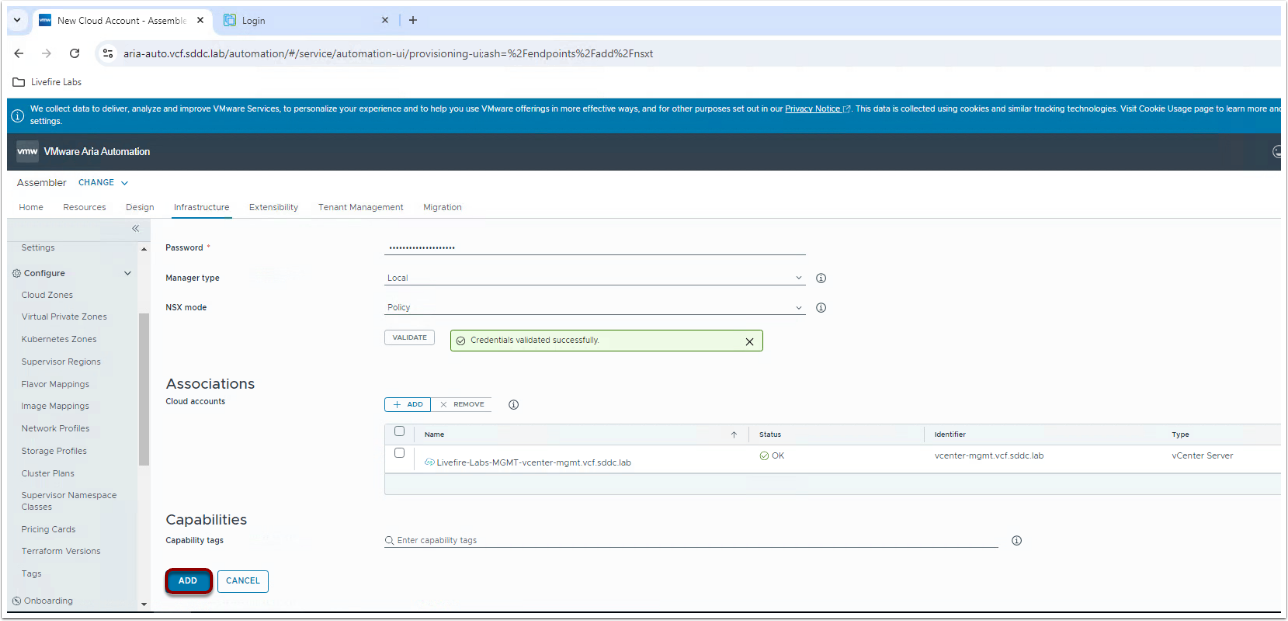


8. Select your Livefire-Labs-MGMT cloud account.

9. Click ADD



Click ADD to save the new cloud account



## 4. Add Tags to Resource Pools

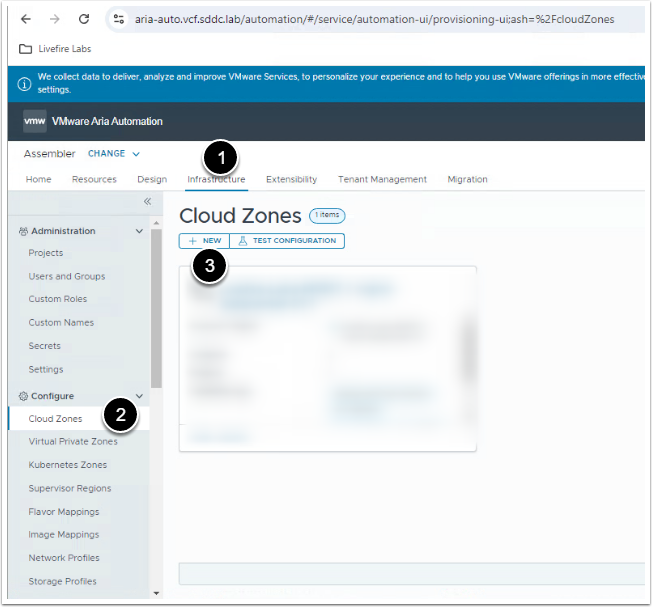
While in Assembler under My Services, go back to the Infrastructure tab

1. Click on Compute
2. Open the mgmt-cluster-01/Livefire-Prod-RP resource pool
3. Add the following tag - Environment:Production
4. Repeat for the mgmt-cluster-01/Livefire-Test-RP and add the Environment:Test tag
5. Repeat for the mgmt-cluster-01/Livefire-Dev-RP and add the Environment:Development tag

## 5. Create New Cloud Zones

While in Assembler under My Services.

1. Go to Infrastructure
2. Click on Cloud Zones
3. If there any existing Cloud Zones, please delete before continuing
4. Click on NEW



4. Enter the following:

    Account/ region: Livefire-Labs-MGMT / mgmt-datacenter-01

    Name: Livefire-Prod

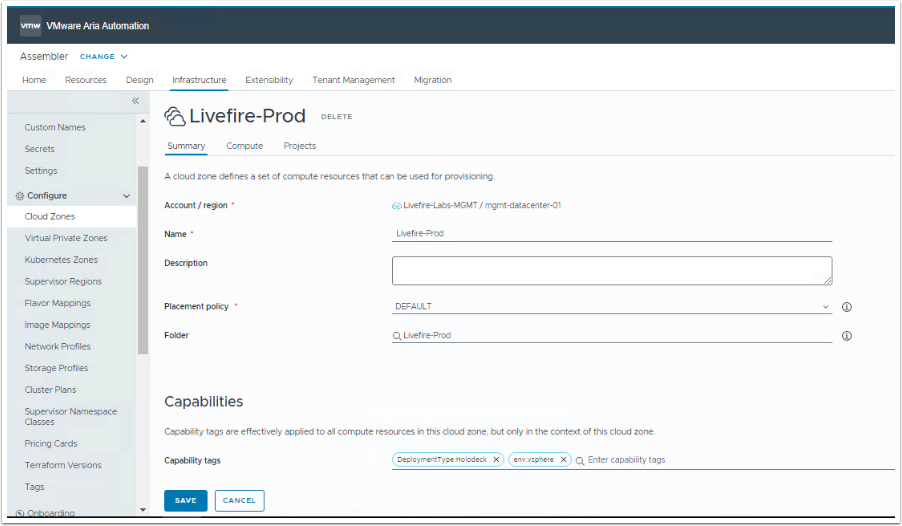
    Placement Policy: DEFAULT

    Folder: Livefire-Prod

5. Add the following Capability Tags:

    DeploymentType:Holodeck

    env:vsphere



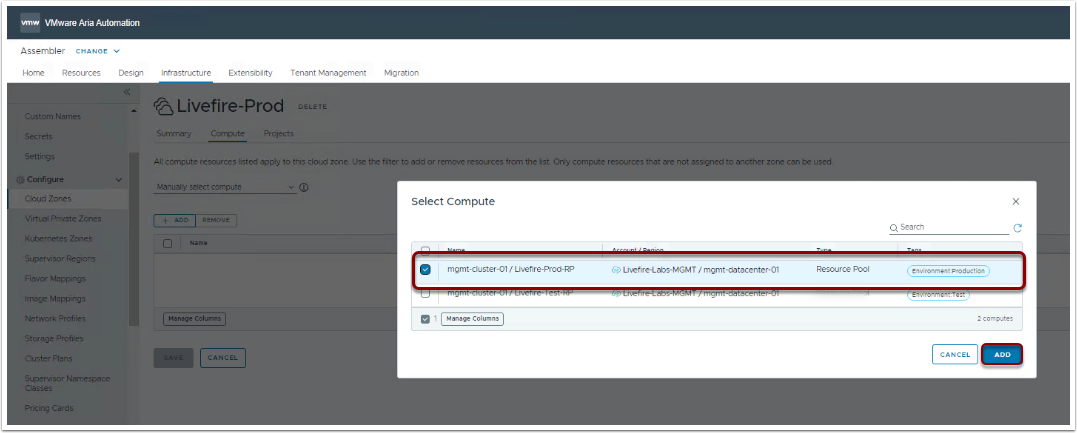
6. On the Compute Tab, add the previously created Resource Pools

 - Select Manually select compute from the dropdown

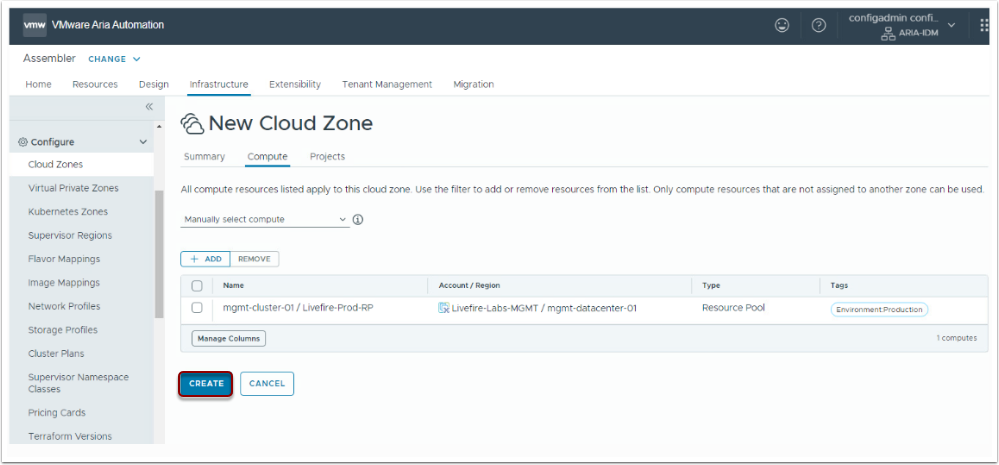
  - Select ADD to see the available Compute

7. Select the mgmt-cluster-01 / Livefire-Prod-RP that was created in the previous steps

8. Click on ADD



9. Click on CREATE to save the new resource



Repeat for both both Dev and Test Cloud Zones using the following:

Account/ region: Livefire-Labs-MGMT / mgmt-datacenter-01

    Name: Livefire-Test

    Placement Policy: DEFAULT

    Folder: Livefire-Test

    Compute: mgmt-cluster-01 / Livefire-Test-RP

    Capability Tags: DeploymentType:Holodeck     env:vsphere

Account/ region: Livefire-Labs-MGMT / mgmt-datacenter-01

    Name: Livefire-Dev

    Placement Policy: DEFAULT

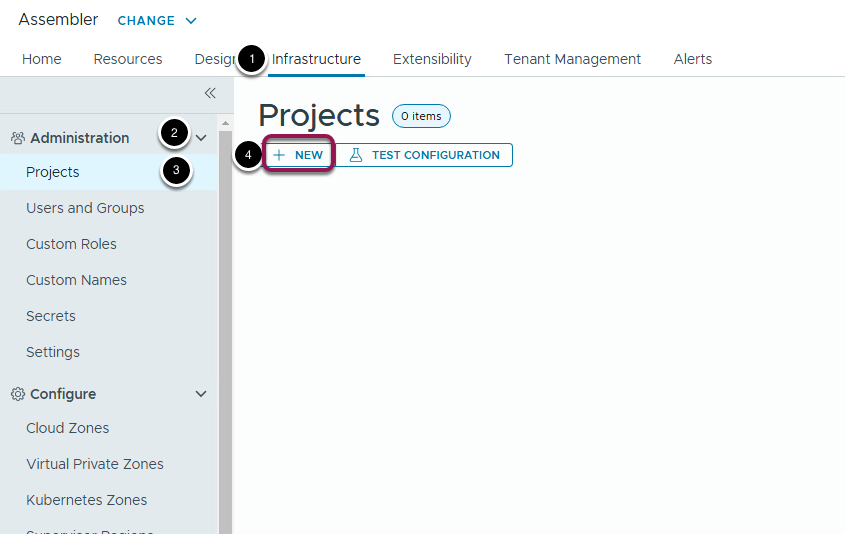
    Folder: Livefire-Dev

    Compute: mgmt-cluster-01 / Livefire-Dev-RP

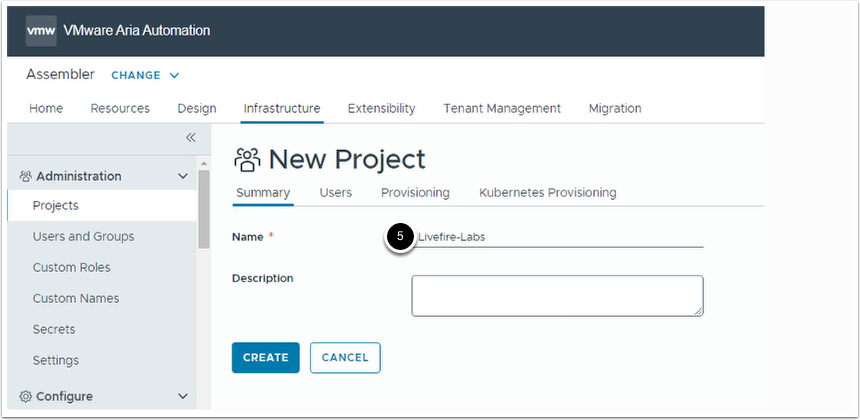
    Capability Tags: DeploymentType:Holodeck     env:vsphere

## 3. Set Up Your Project

1. Click on Infrastructure
2. Click on Administration
3. Click on Projects
4. Click on NEW



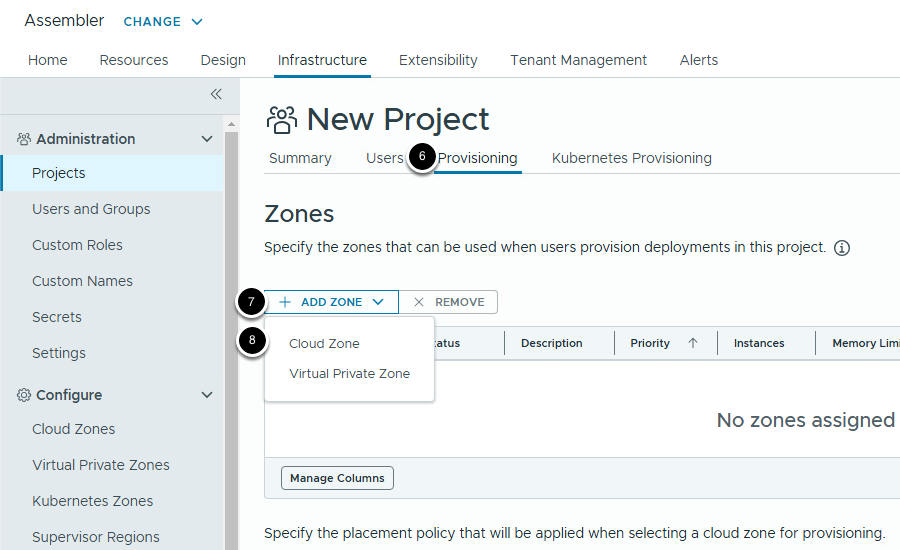
5. Enter a project name: Livefire-Labs



6. Click on the Provisioning tab

7. Click on the ADD ZONE option

8. Click on Cloud Zone



9. Select your Livefire Cloud Zone - Livefire-Prod

10. Enter 1 for Provisioning Priority

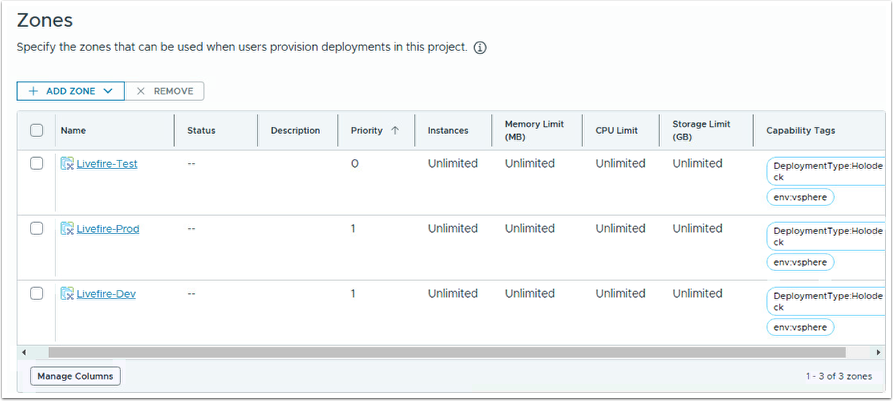
11. Accept the remaining defaults

10. Click ADD



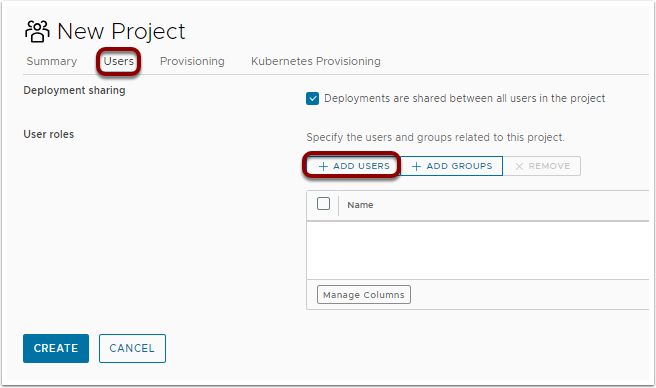
Repeat for the Livefire-Test (Provisioning priority - 0) and Livefire Dev (Provisioning priority - 1)

You should see the following:



11. Click on the Users tab

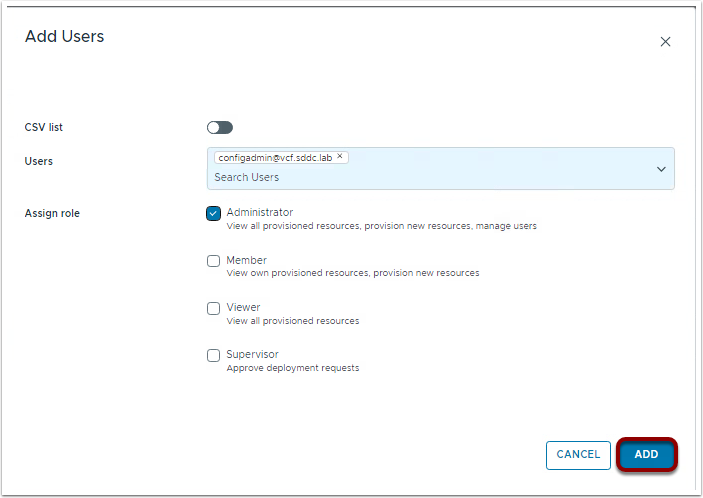
12. Click on the ADD USERS option



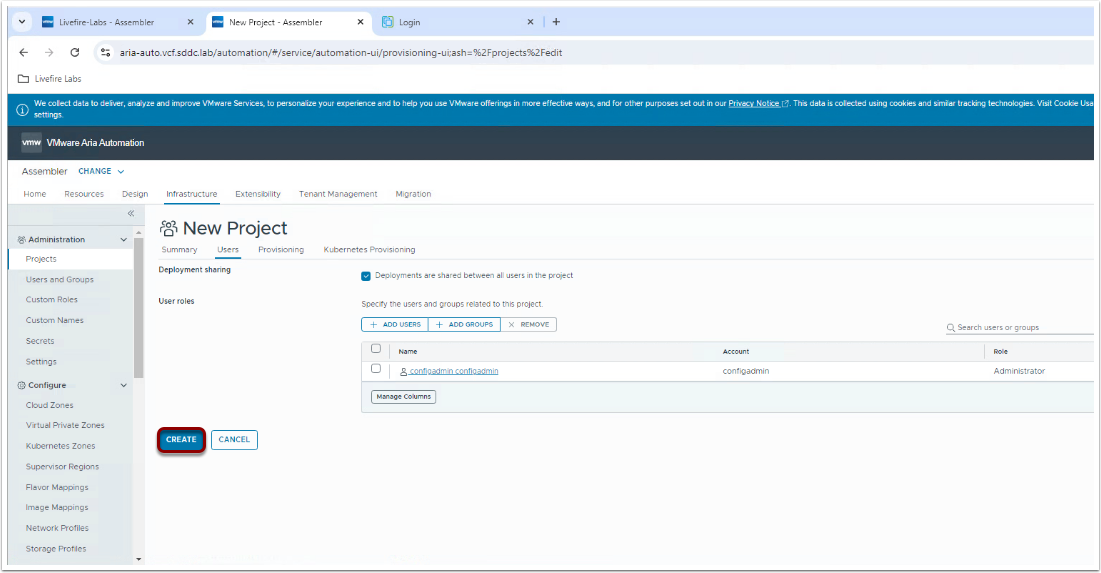
13. Search for configadmin

14. Click on Administrator for the role

15. Click on ADD



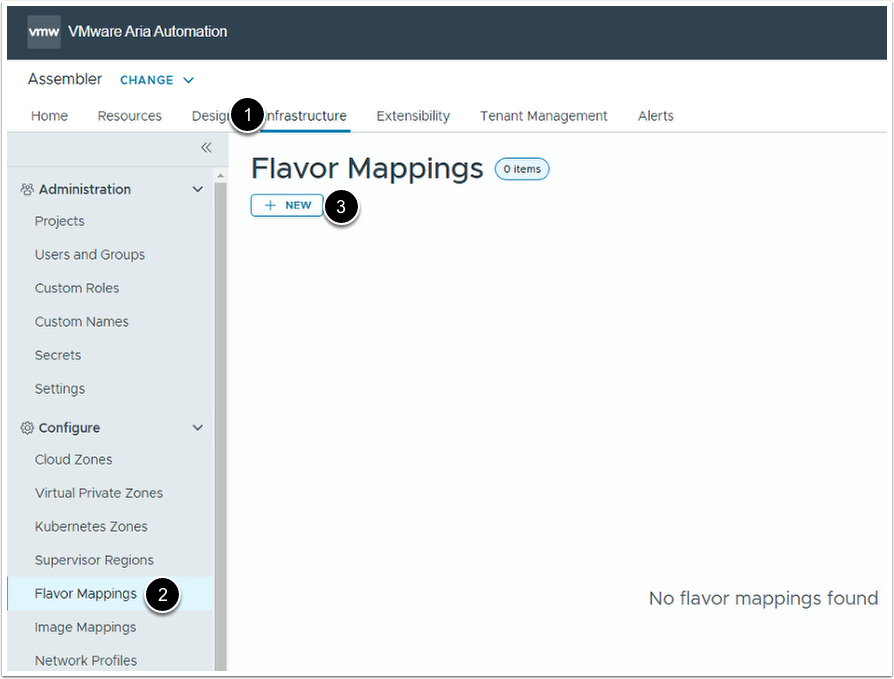
16. Click on CREATE



## 6. Add Flavor Mapping

While in Assembler:

1. Go to Infrastructure
2. Click on Flavor Mappings
3. Click on NEW



Enter the following:

1. Flavor name: Large

2. Account / Region: Livefire-Labs-MGMT / mgmt-datacenter-01

Value:

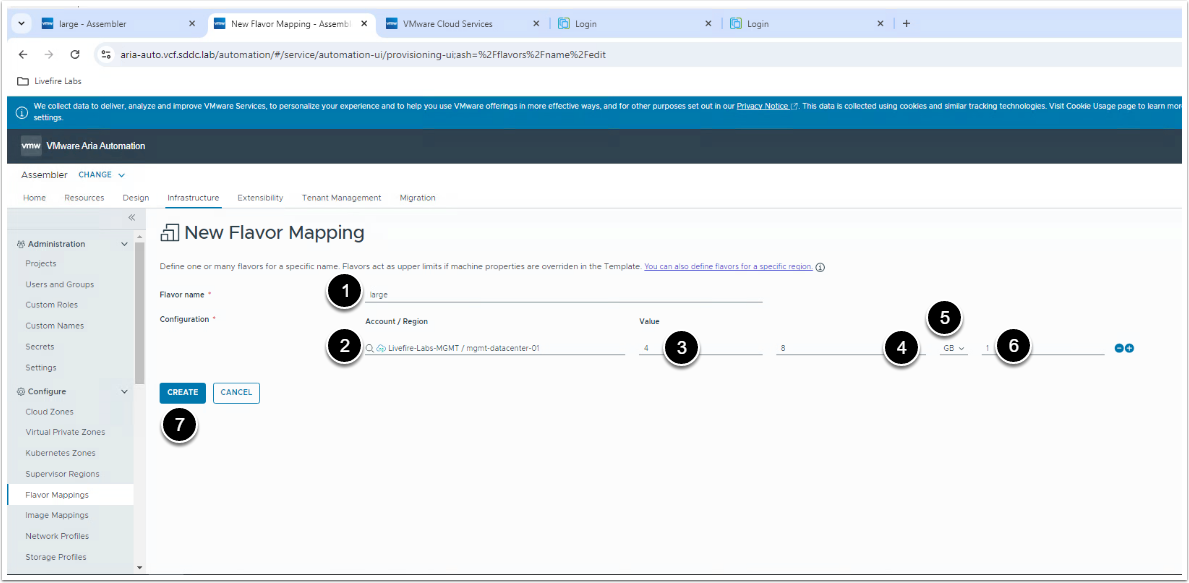
3. 4 (CPUs)

4. 8 (Memory)

5. GB

6. 1 (Core per socket)

7. Click on CREATE to save



Repeat the same for medium and small flavor mappings using the following:

1. Flavor name: Medium

2. Account / Region: Livefire-Labs-MGMT / mgmt-datacenter-01

Value:

3. 2 (CPUs)

4. 4 (Memory)

5. GB

6. 1 (Cores per socket)

1. Flavor name: Small

2. Account / Region: Livefire-Labs-MGMT / mgmt-datacenter-01

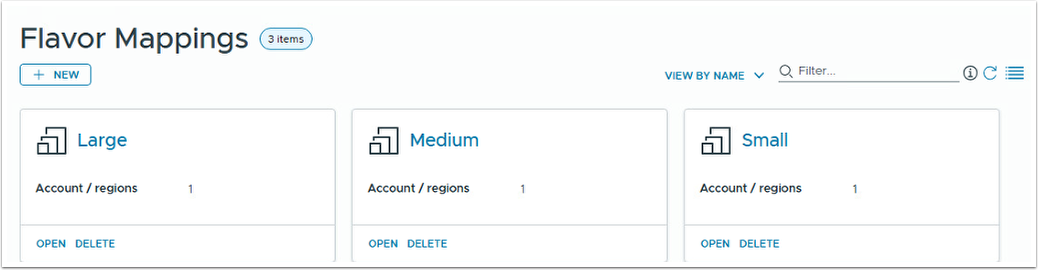
Value:

3. 1 (CPUs)

4. 2 (Memory)

5. GB

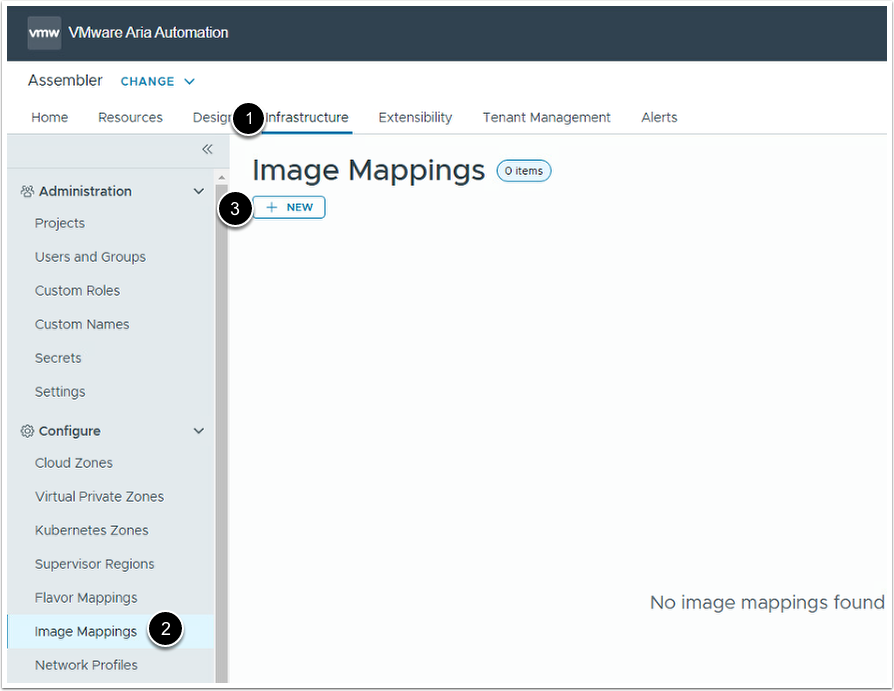
6. 1 (Cores per socket)



## 7. Add Image Mapping

While in Assembler:

1. Go to Infrastructure
2. Click on Image Mappings
3. Click on NEW



Enter the following:

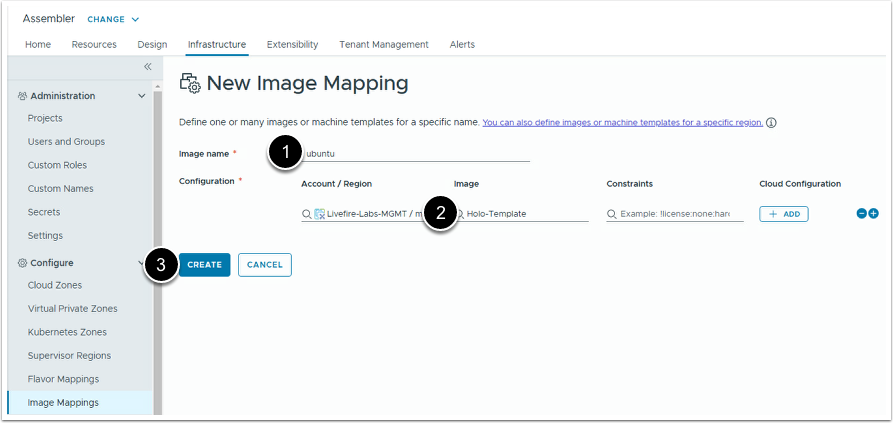
1.  Image name: ubuntu

2.  Configuration:

       Account / Region: Livefire-Labs-MGMT / mgmt-datacenter-01

      Image: Holo-Template

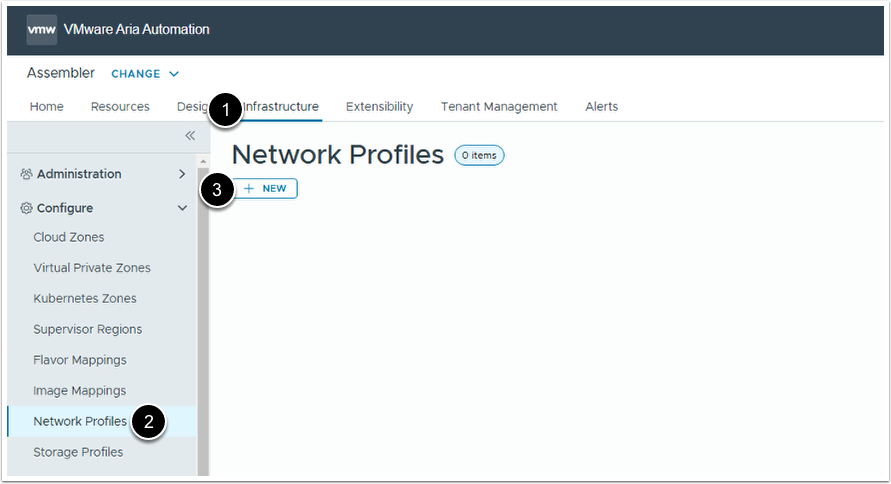
3. Click on CREATE to save



## 8. Add Network Profiles

While in Assembler:

1. Go to Infrastructure
2. Click on Network Profiles
3. Click on NEW



Enter the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: OC-Auto-Seg (Dev)

3.  Capability tags:

   DeploymentType:Holodeck

   Environment:Development

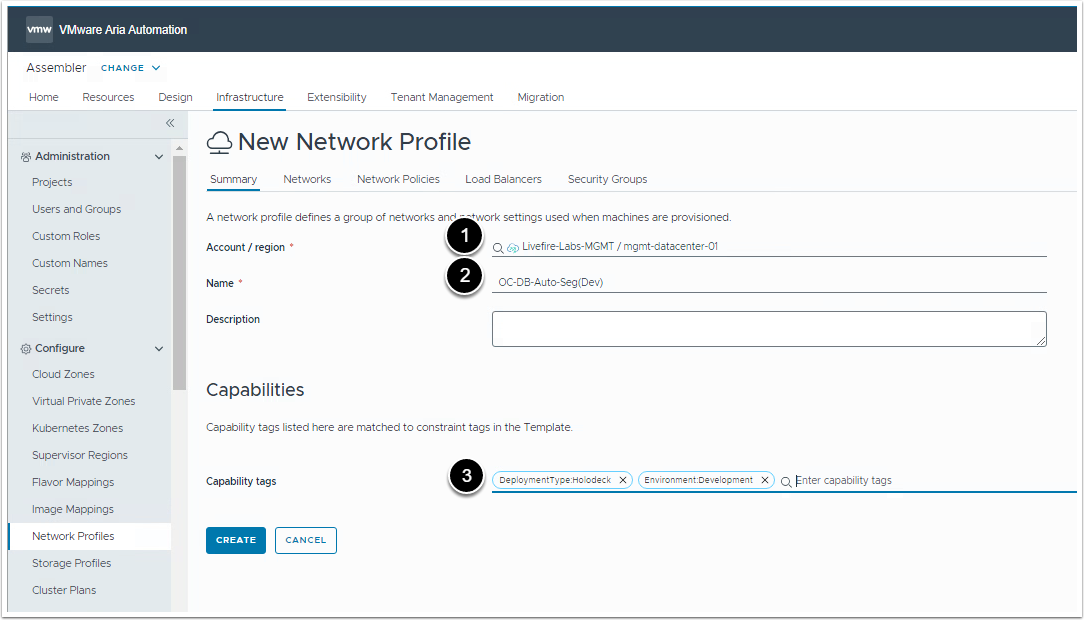
4. Click on the Networks Tab

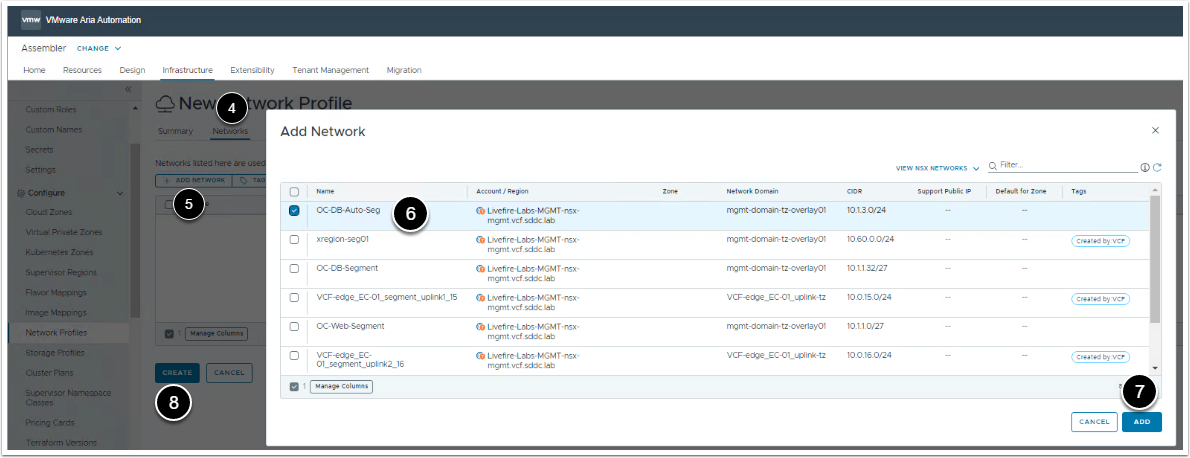
5. Click on ADD NETWORK

6. Select OC-DB-Auto-Seg

7. Click ADD to save

8. Click CREATE to create the network profile





Repeat for the OC-Auto-Seg(Test) network profile and enter the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: OC-Auto-Seg(Test)

3.  Capability tags:

   DeploymentType:Holodeck

   Environment:Test

4. Add the OC-Web-Auto-Seg network on the Networks tab

5. Click on CREATE to save

Repeat again for the OC-AUTO-Seg(Prod) network profile and enter the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: OC-Auto-Seg(Prod)

3.  Capability tags:

   DeploymentType:Holodeck

   Environment:Production

   network:dmz

4. Add the xregion-seg01 network on the Networks tab

5. Click on CREATE to save

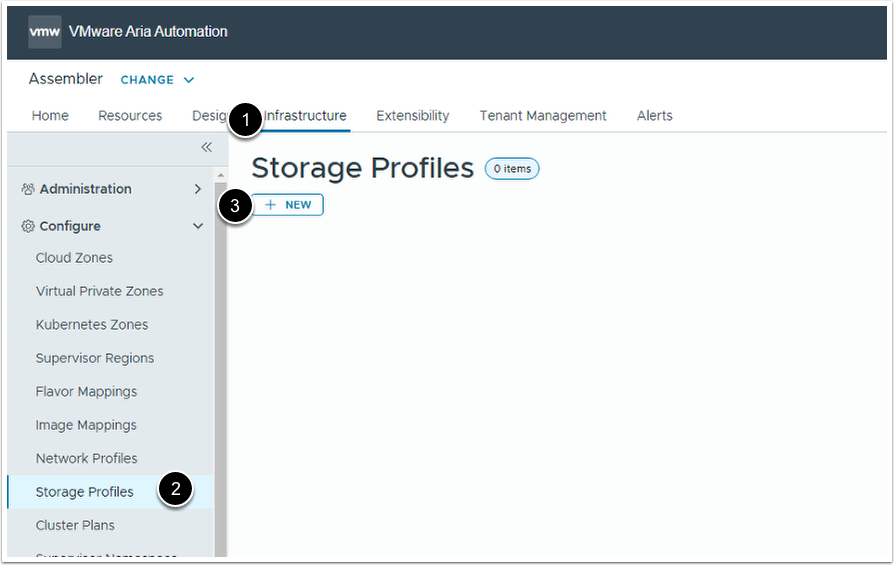
## Add Storage Profiles

In Aria Automation, a storage profile is a set of predefined settings and configurations for managing storage resources. It simplifies and standardizes how storage is allocated and configured for virtual machines (VMs) and other resources.

Also note that we are associating each Storage Profile to its corresponding Resource Pool to support the partitioned environment.

While in Assembler:

1. Go to Infrastructure
2. Click on Storage Profiles
3. Click on NEW



Enter the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: Livefire-standard

3. Disk type - Standard disk

4. Compute - mgmt-cluster-01 / Livefire-Dev-RP

Accept defaults for all remaining entries excluding Priority and Capability tags

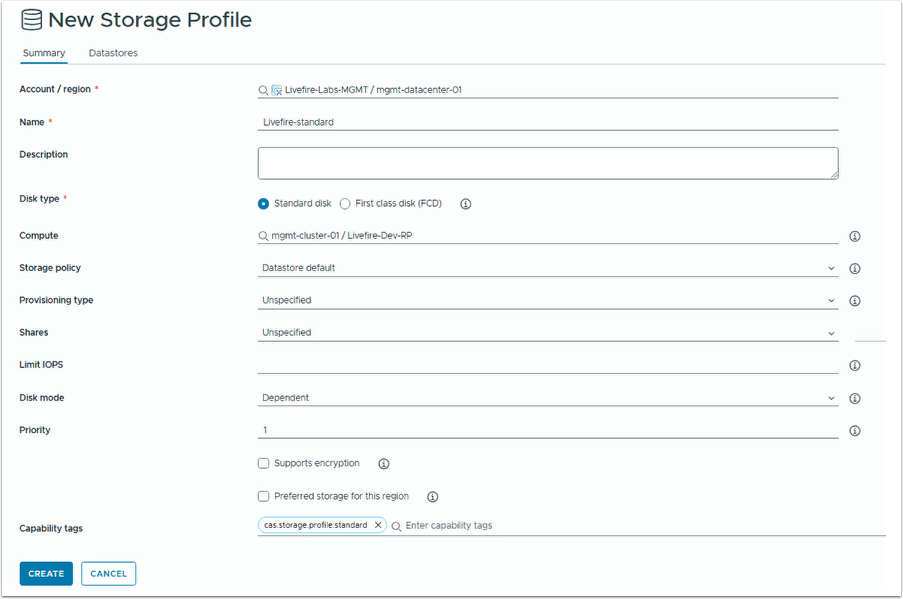
5. Priority: 1

6. Check the Preferred storage for this region box

7. Capability tags:

    cas.storage.profile:standard

8. Click on CREATE to save



Repeat steps to add the Livefire-PCI storage profile using the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: Livefire-PCI

3. Disk type: Standard disk

4. Compute:

    mgmt-cluster-01 / Livefire-Test-RP

5. Storage policy:

    Datastore default

Accept defaults for Provisioning type, Shares, Limit IOPs and Disk mode

6. Priority:

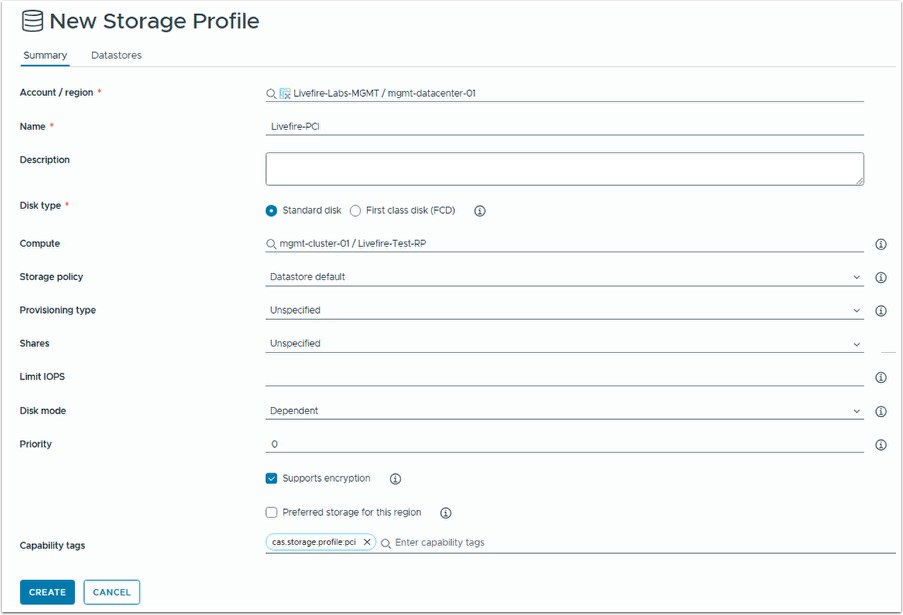
    Priority 0

7. Check the Supports Encryption box

8. Capability tags:

    cas.storage.profile:pci

9. Click on CREATE to save



Repeat steps to add the Livefire-PCI-2 storage profile using the following:

1.  Account / Region:  Livefire-Labs-MGMT / mgmt-datacenter-01

2.  Name: Livefire-PCI-2

3. Disk type: Standard disk

4. Compute:

    mgmt-cluster-01 / Livefire-Prod-RP

5. Storage policy:

    Datastore default

Accept defaults for Provisioning type, Shares, Limit IOPs and Disk mode

6. Priority:

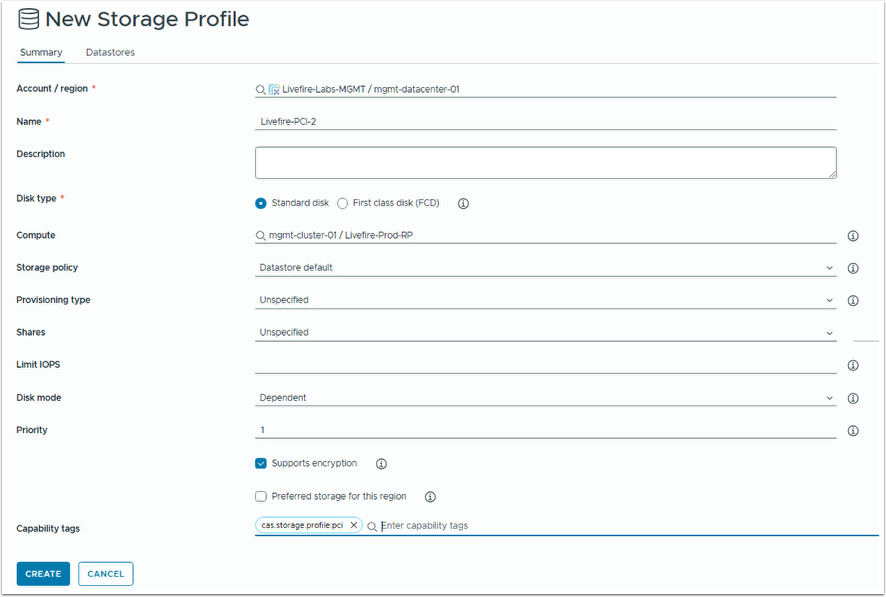
    Priority 1

7. Check the Supports Encryption box

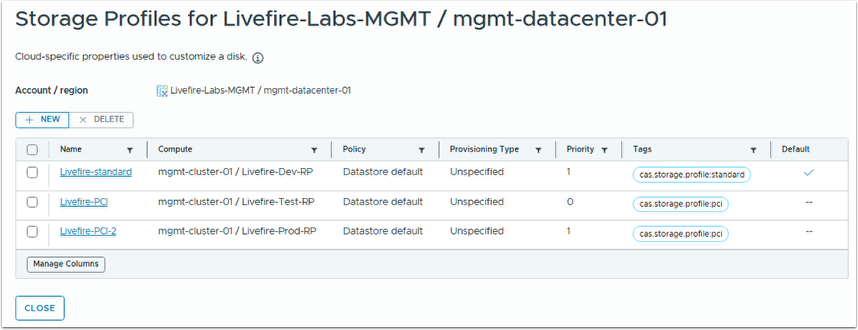
8. Capability tags:

    cas.storage.profile:pci

9. Click on CREATE to save



Your Storage Profile should look like the following:

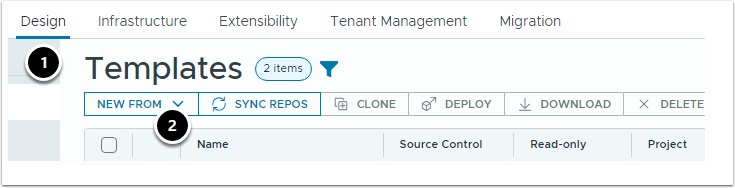


## 01. Setting up the Blueprints

## Importing the Blueprints

1. In Cloud Assembly navigate to the Design tab

2. Select NEW FROM from Upload



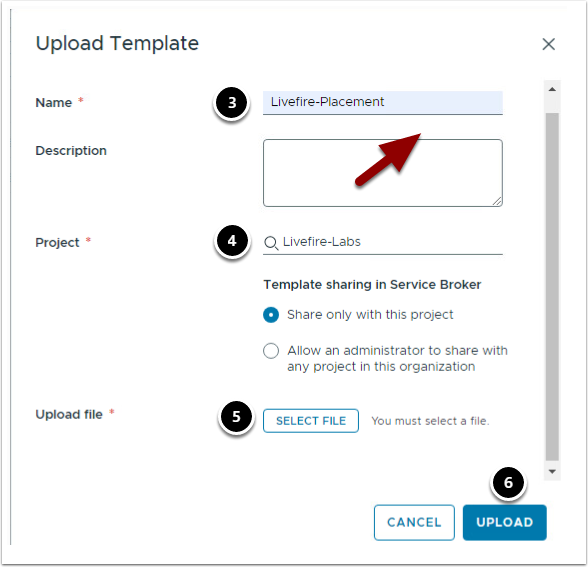
3. Name your blueprint Livefire-Placement.

4. Select Livefire-Labs for the project

5. Select the template from the  C:\Users\Administrator\Document\Livefire\_Labs directory

6. Click on UPLOAD

7. Repeat for the Livefire-Placement-2 template and name the file  Livefire-Placement-2



## Understanding Placement Decisions

* 00. Lab Introduction
* 01. Adding a PCI Resource Constraint Tag
* 02. Understanding the Placement Decision (PCI)
* 03. Adding a DMZ Network Constraint Tag

## 00. Lab Introduction

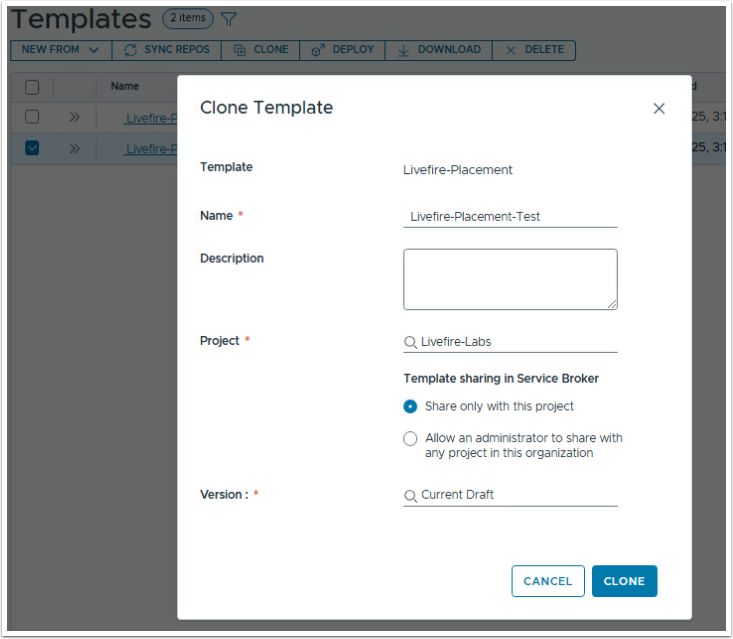
## 01. Adding a PCI Resource Constraint Tag

## Fixing the Failing Deployment

Open the Design , then Templates tab in Assembler.

Clone the Livefire-Placement template and name it Livefire-Placement-Test.

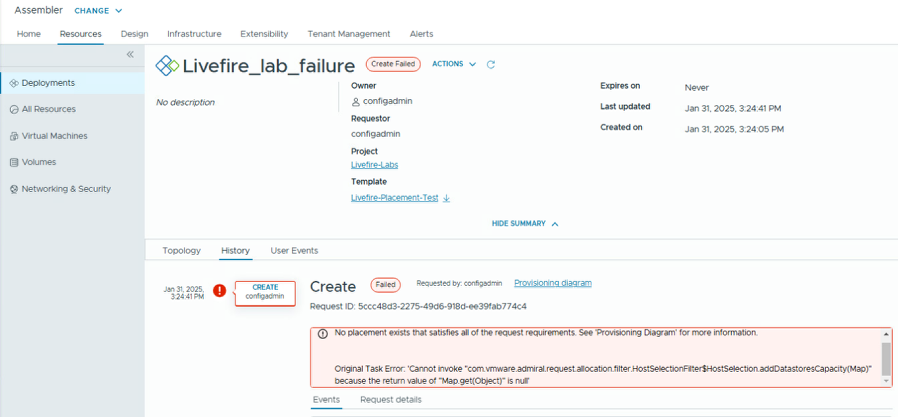
Select Livefire-Labs for the Project and Current Draft for the version.



Open the new template and examine the blueprint. Note that we only have size (flavor) and environment inputs.

Also note that the our machine resources must comply with PCI regulations so we've put a constraint on the disk: 'cas.storage.profile:pci'

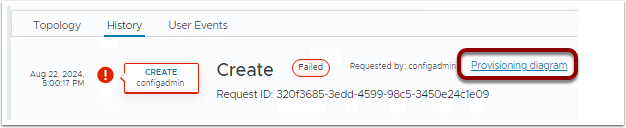
Deploy the blueprint to the Production environment and select small  as the flavor. Note that you will receive an error as shown below.



Why is the deployment failing?

Examine the deployment request details from the Provisioning diagram for more info.

How would you fix the broken deployment?



## 02. Understanding the Placement Decision (PCI)

## Understanding the Placement Decision

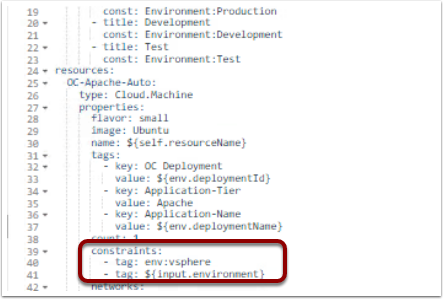
Examine again the deployment request details from the previous lab. Examine the resource request details.

Your deployment ended up in the Livefire-Prod-RP.



Now, revisit the template and note that there is also a env:vsphere hard constraint tag in the template which is not listed as a capability constraint on the Cloud Zone Computes.  And...the deployment didn't fail but why?

Why didn't the deployment fail when neither of the Resources Pools contained the hard constraint tag?



## 03. Adding a DMZ Network Constraint Tag

To test, create a deployment using the original Livefire-Placement template.

1. Deploy the template and select the environment of your choice

2. Now review the placement...you'll find that deployment is now assigned to the same Network Profile environment and Cloud Zone environment.

## Adding DMZ Network Capability Constraint

1. Clone the Livefire-Placement-2 template.

2. Name the new template Livefire-Placement-DMZ

3. Create a new deployment and name it Pre-DMZ...accept the defaults.

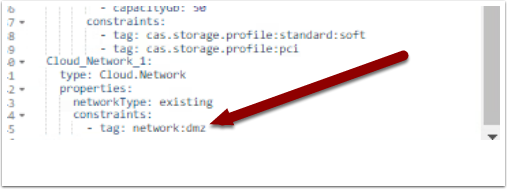
4. Make note of which Cloud Zone the deployment was placed in.  Why do think the deployment was placed in Livefire-Test this time when both Livefire-Test and Livefire-Prod have the PCI constraint tag?

1. Now update the Livefire-Placement-2 template with the DMZ constraint tag

2. Add the following Constraint Tag for the Cloud Network:

      constraints:

         -tag:  network:dmz



Deploy using the defaults inputs.

Examine the deployment request details from the Requests tab for more info.

But before you do, where do you think the deployment was placed?

OK, now check...were you correct?

Why did the deployment engine decide to place the deployment in (your guess)?

## Deployment Placement with Inputs

Since there's no OOB method of connecting the network and cloud machine resources, let's create one. It's actually already configured in the Livefire-Placement template using inputs. By passing the constraints received from the Cloud Machine input parameters to the network settings we are able to assign the same constraints to both.

## Building a Complex YAML Blueprint

* Lab Introduction
* Setting up the Blueprint
* Template versioning
* Deploying template and updating existing deployment
* Adding expressions
* Input based resource allocation
* Adding vCenter to Orchestrator inventory
* Importing Orchestrator package
* Adding day 2 activity
* Testing template with multiple inputs
* Creating custom resource
* Creating a subscription
* Creating custom property for resources in template
* Pop Quiz
* Testing the template
* Bonus- using property Groups as template input
* Cleanup (optional)

## Lab Introduction

## Setting up the Blueprint

## Template versioning

## Deploying template and updating existing deployment

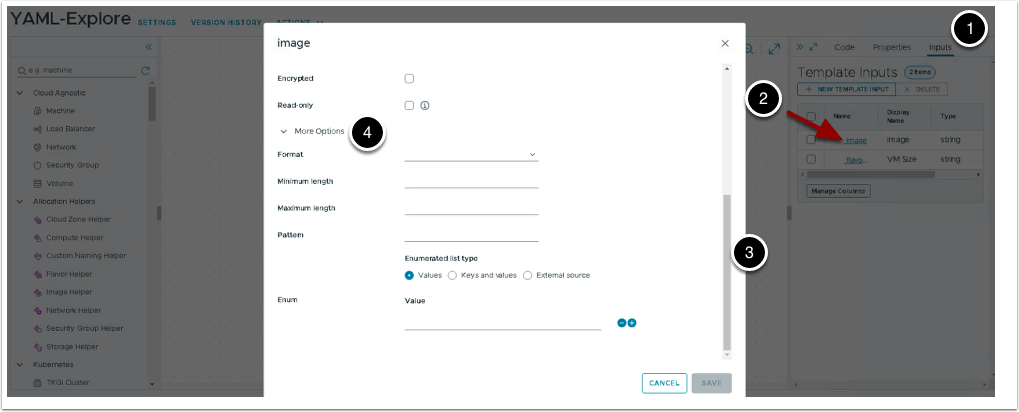
## Adding expressions

## Input based resource allocation

## Adding inputs to template

In current template we already have 2 inputs defined. Our task in this section is to add more inputs and make inputs more user-friendly

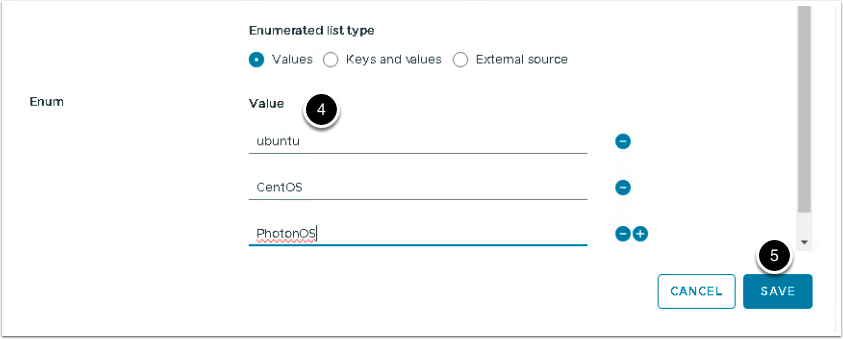
1. Select Inputs section,
2. then click on image input.
3. In new window select scroll bar
4. Then expand More Options.



1. Under Enum add 3 values,

* ubuntu
* CentOS
* PhotonOS

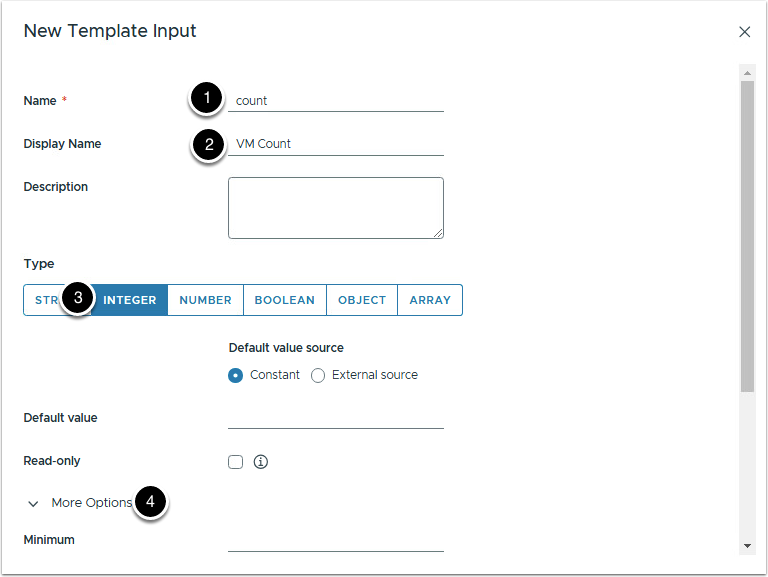
1. Select default value as ubuntu and click Save.



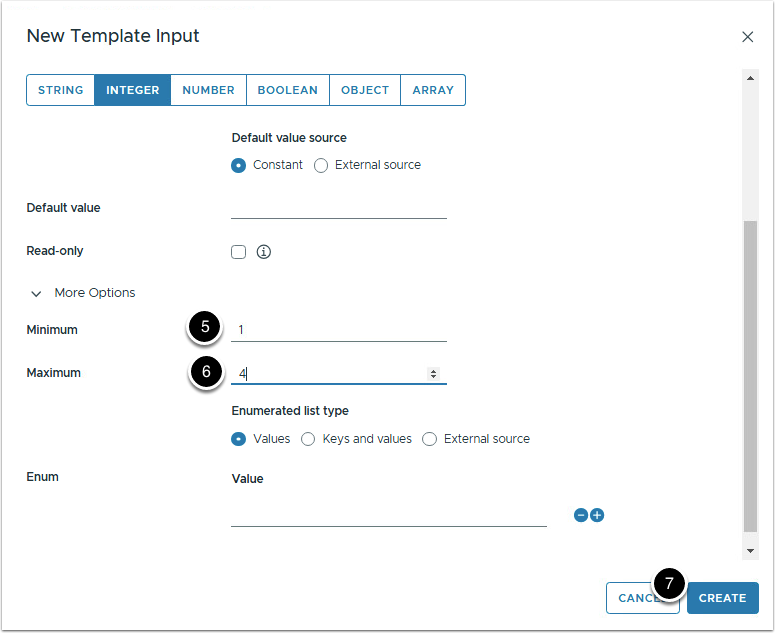
Similarly under flavour input add small, medium and large as input options, with small as default.

Create new input with

1. Name = count
2. display name = VM Count
3. type = Integer
4. Expand More Options.



1. Minimum = 1
2. Maximum = 4
3. Click Create.



We will be using count input to set the number of VMs. Set count property value in web-tier resource to ${input.count}

Now your input section of the code should look like.

## Lets add inputs with key value pair.

Add new input with below values.

* Name: location
* Display Name: Location
* Type: String
* For values add Enumerated List types as Key and values pair with default value being 'Adding multiple disks to web-tier resource.

At this point input section of code should look like.

We will not use this input as of now.

Add another property with key value pair input options

* Name: env
* Display Name: Environment
* Type: String
* For values add Enumerated List types as key and value pair with default value being Environment:Production.

We will also be not using this input now.

## Adding object type input property

We will now add array type input to get user input on number of additional disks to be added to web-tier resource. Input will include name of the disk and size of disk, there is will be limit on how many disks to be added and maximum and minimum size of each disk.

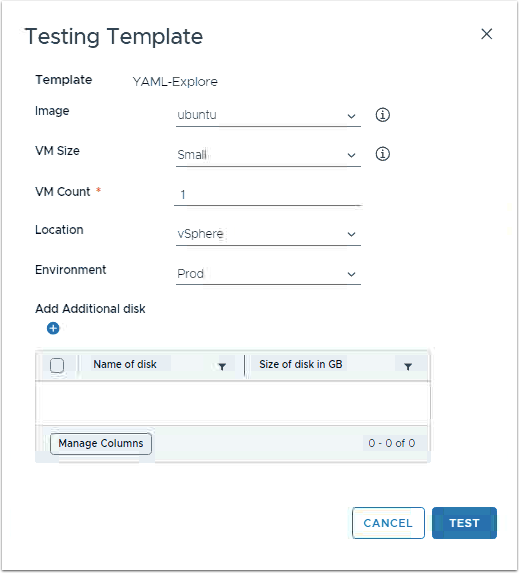
Create new input with below details and then click Create.

* Name: disk\_config
* Display Name: Add Additional disk
* Type: Array
* Min Items = 0 and Max Items = 3

Return to code section and add below code to disk\_config input. Copy the code below and replace the disk\_config input section.

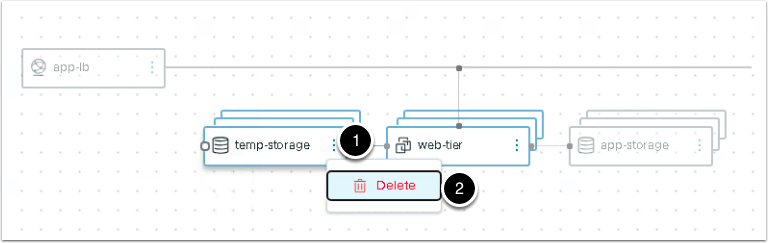
So what we are doing here, within the disk\_config array input we are creating items of type object (which is another input type). An object can have multiple properties. We are creating disk\_name and disk\_size as inputs.

At this point if we click on test to check template inputs, it should look like the image below.



## Using inputs to set resource values

Since we are getting input for all volumes from our object array input, temp-storage resource is no longer needed. Delete the resource, click vertical ellipsis > Delete.



Now lets update app-storage resource to create all disks from input and update web-tier resource to add correct disks to each web-tier resource. Our code now will look like this.

Lets break down the complexity for app-storage resource.

* count property for the volume is straight forward enough, number of disks provided as input multiplied by the number of VMs.
* count.index is the position of the current volume in the resource array. As we iterate through the array, this will be 0 and then 1 and so on.
* To get capacityGB and name we used a mathematical formula. Turns out if we divide 0 by any number remainder is always 0, similarly if we divide 1 by any number remainder is always 1 and so on. Using this formula we can get the index value from array of disks.

Let break down complexity for attached disks in web-tier resource.

* count.index is the position of the current VM in the resource array. As we iterate through the array, this will be 0 and then 1 and so on.
* map\_to\_object will create a key value pair, in our case it will create a list of key value pair with source being the key and volume.id being the value.
* slice expression will help us get the correct volumeID.
* resource['app-storage'][\*]['id'] is an array of volume IDs.
* length(input.disk\_config) \* count.index, this will equate to 0 for 1st web-app resource, n (if n is number of disks) for 2nd resource and n+n for 3rd resource and so on.
* length(input.disk\_config)\*(count.index + 1), this will equate to n for 1st web-app resource, n+n+1 for 2nd resource and so on

## Adding vCenter to Orchestrator inventory

## Importing Orchestrator package

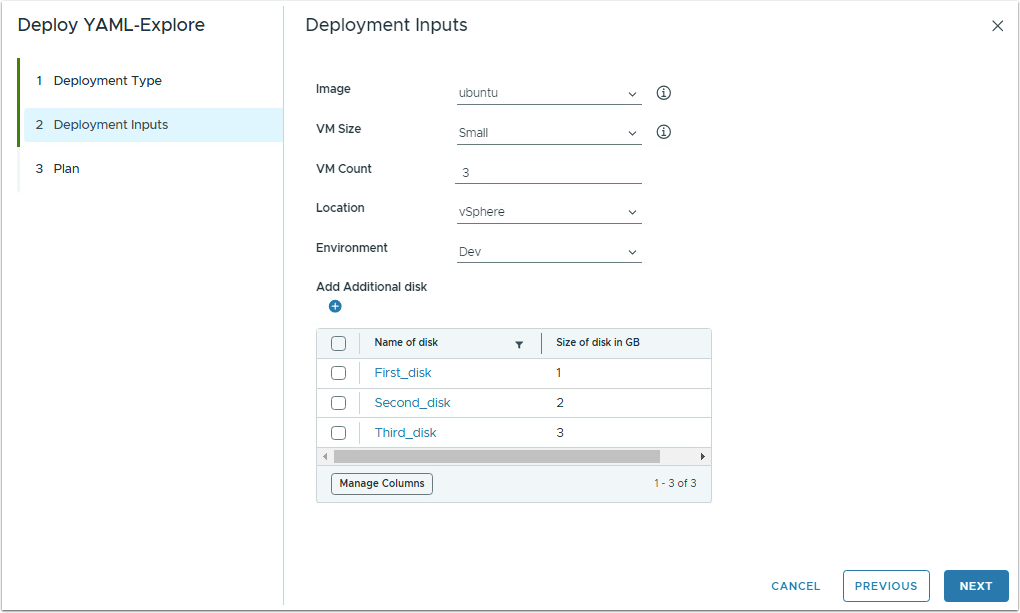
## Adding day 2 activity

## Testing template with multiple inputs

Now lets test our template against existing deployment.

Click Deploy > Update existing deployment > select multi-stage-deployment > Next. Under Deployment Inputs set VM Count to 3 and add 3 additional disks.  After giving much thought we have carefully selected these names and size inputs.

Deployment inputs should look like the image below.



Click Next and observe under Plan section

* 9 Volumes to be created.
* 1 VM to be created and 2 VMs to be updated.

Click Deploy to update the deployment.

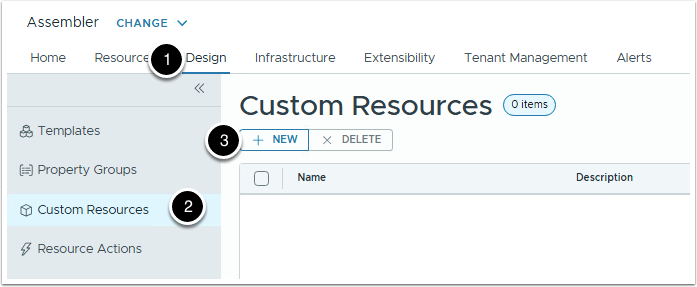
Create new template version after successful deployment.

Version: 5

Description: Inputs of type string, integer, array and object used

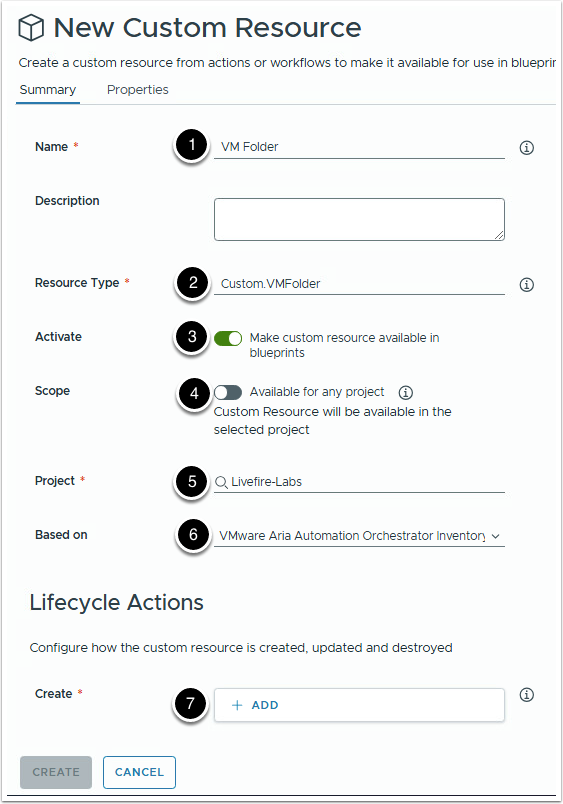
## Creating custom resource

To create custom template go to Design tab > Custom Resources > New.

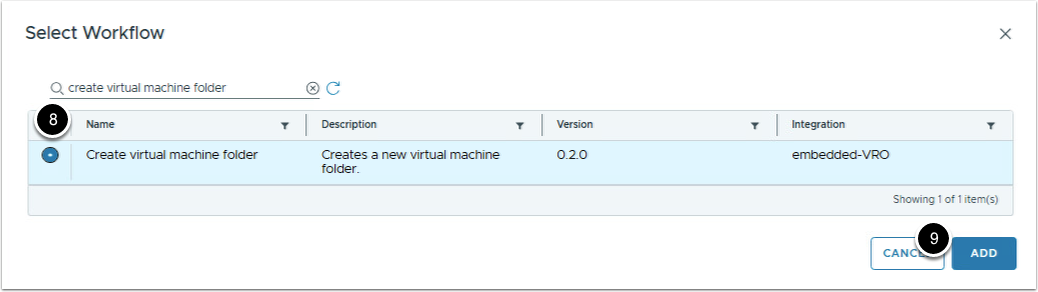


Create new custom resource with

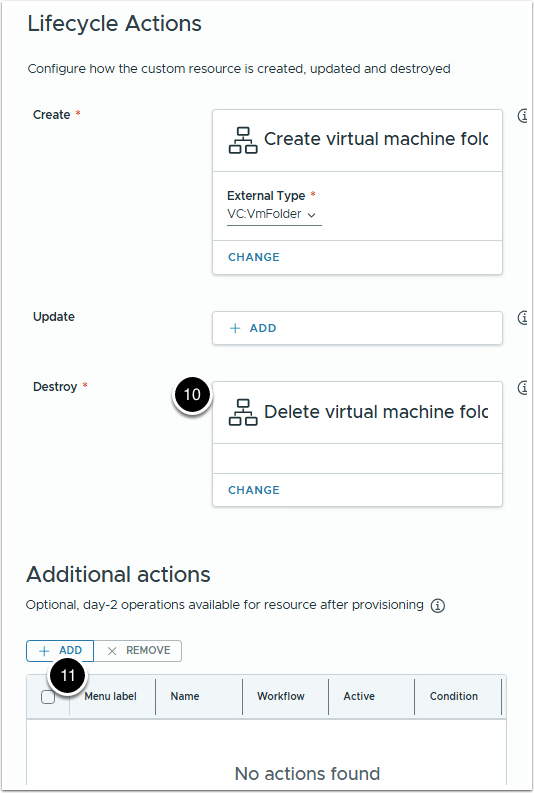
1. Name as VM Folder
2. Resource Type as Custom.VMFolder
3. Click on the slider to activate the custom resource
4. Since we want the custom resource to be only available to our project. Click on slider to disable Scope
5. Select Livefire-Labs as Project
6. Custom resource can be based on Orchestrator workflow or ABX actions. We are using Orchestrator workflow
7. Under Lifecycle Actions we will add Create and Delete action. Click + ADD to link a Orchestrator workflow



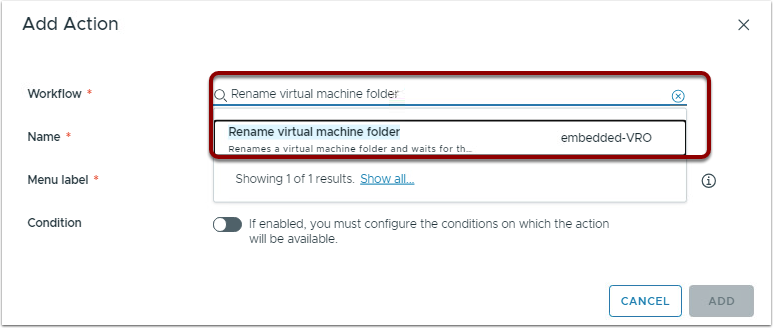
1. Select workflow by name Create virtual machine folder
2. Click ADD



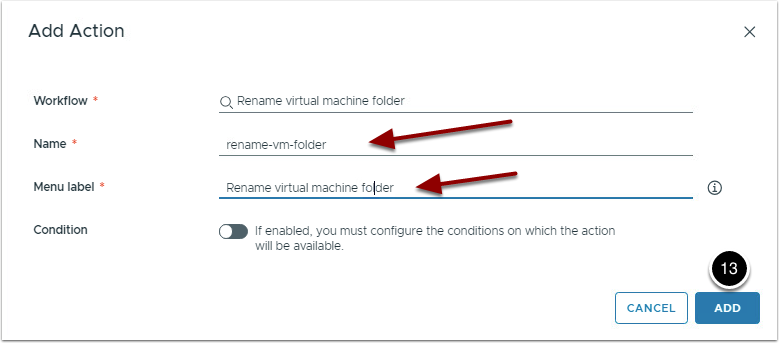
1. Similarly, for Destroy action click +ADD and select workflow Delete virtual machine folder
2. For Additional actions, click +ADD



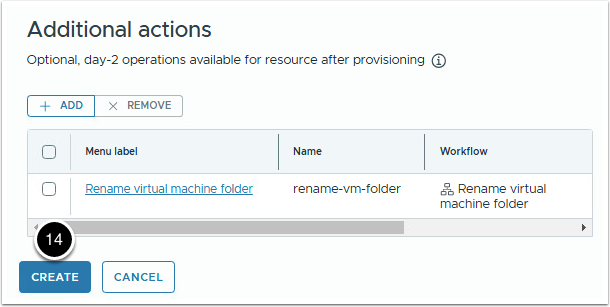
1. Within the Add Action tab update below inputs  
   Workflow = Rename virtual machine folder  
     
   Name = rename-vm-folder  
     
   Menu label = Rename virtual machine folder
2. Workflow = Rename virtual machine folder
3. Name = rename-vm-folder
4. Menu label = Rename virtual machine folder



1. click ADD



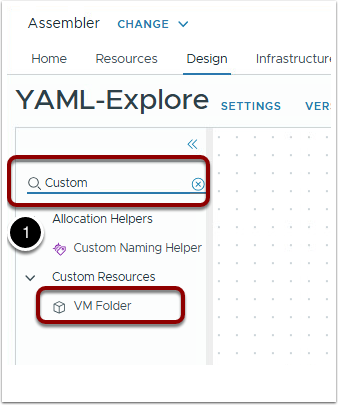
1. Finally click Create.



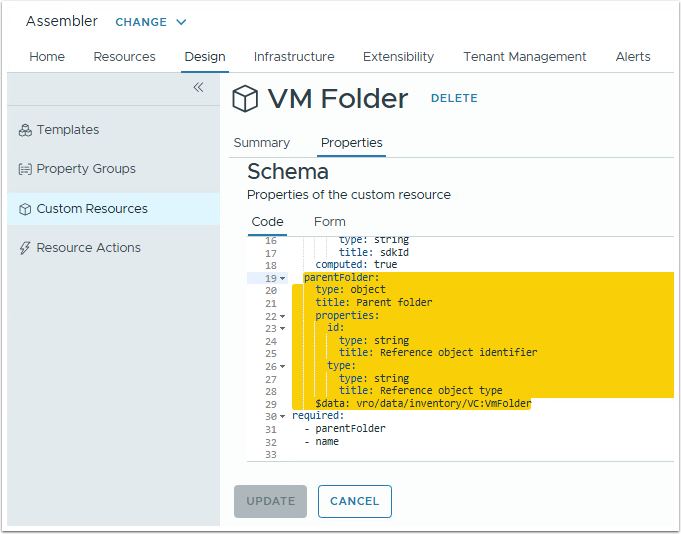
## Using custom resource in template

Go back to the template,

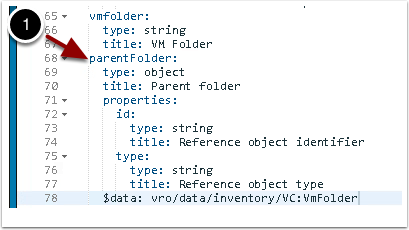
1. from the list of resources search for custom resource. Custom resource with name VM Folder is available for consumption



1. Drag and drop the custom resource in the canvas. Rename the Custom\_VMFolder\_1 to web-app-folder
2. Create new input to get name of the VM folder  
   Name = vmfolder  
     
   Display Name = VM Folder  
     
   Type = String
3. Name = vmfolder
4. Display Name = VM Folder
5. Type = String
6. To create parent folder input:  
   Go back to our VM Folder custom resource  
   Select properties tab  
   Code section of custom resource VM Folder is already selected. Copy the content from line 19 to 29
7. Go back to our VM Folder custom resource
8. Select properties tab
9. Code section of custom resource VM Folder is already selected. Copy the content from line 19 to 29



1. Paste the copied content in the template, below vmfolder input.



1. We now need to bind inputs to the custom resource. Bind name property to input vmfolder and parentFolder to the input parentFolder.

Click Test to get a glimpse of how template input looks like.

## Creating a subscription

## Creating custom property for resources in template

In past few sections we created custom resource and subscription, to bind all these together we will create a custom property which will be used by orchestrator workflow (Livefire-Move-VM-To-Folder).

Now lets add the custom property in web-tier resource. We are naming the property folderIDProp.

Name of the property tells us about what will be the value of this property.

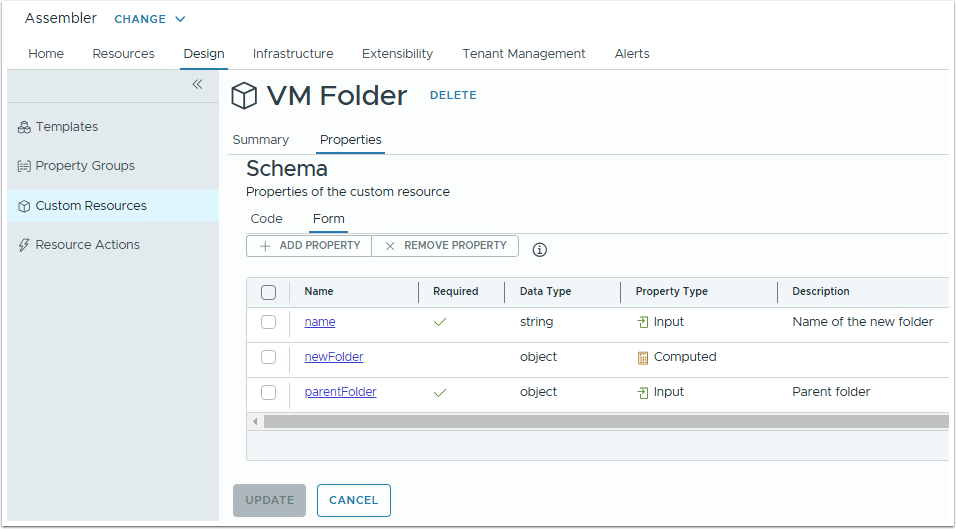
* folderID => this property will hold the ID of the vmfolder where VMs are to be moved.
* Prop => this is our way of signifying that this is a custom property.

Now lets add the property to our web-tier resource.

How did we arrive at this value for our property. Lets go back to our custom resource and explore the property section.

Go to Design > Custom Resources > VM Folder > Properties > Form

In the form newFolder object is listed as the output property. That means newFolder variable holds all the details of the vmfolder(created during the deployment) and we need id property from it.



With these changes code for our web-tier resource should look like:

Lets now create new template version:

Version: 6

Description: Add custom resource, custom property and subscription to the template

## Pop Quiz

Till now the cloud zone used by our deployment was not under our control. Aria automation was deploying resources on the priority of cloud zones within the project. Your mission should you choose to accept is to force the deployment to cloud and environment of your choosing.

I would strongly suggest not to directly jump to solution and come up with your own. Your solution could very well be different from ours and thats perfectly fine.

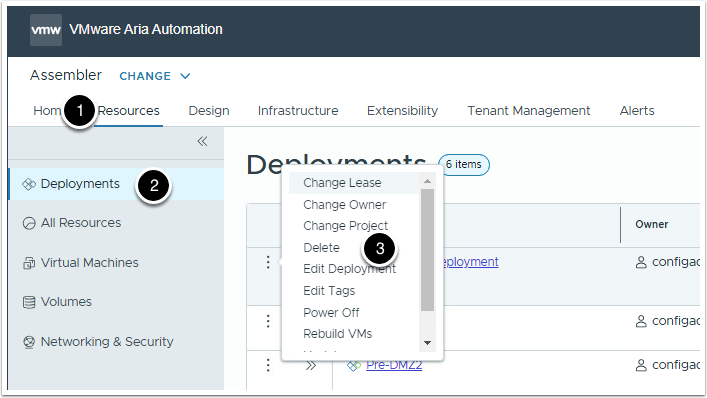
## Testing the template

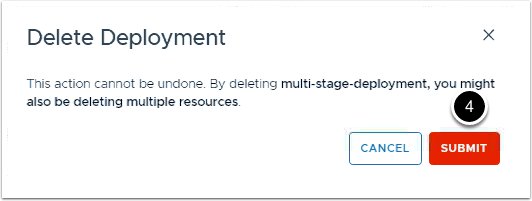
We want you to test all aspects of a template.

* Maximums and minimums (in terms of size and numbers). See if template throws error if you exceed the limit.
* Use the subscription to force VM to a different vmfolder then default.
* Modify the template, try something new.
* Break things and make things.

## Cleanup

Delete all deployments you created at the end of the lab.

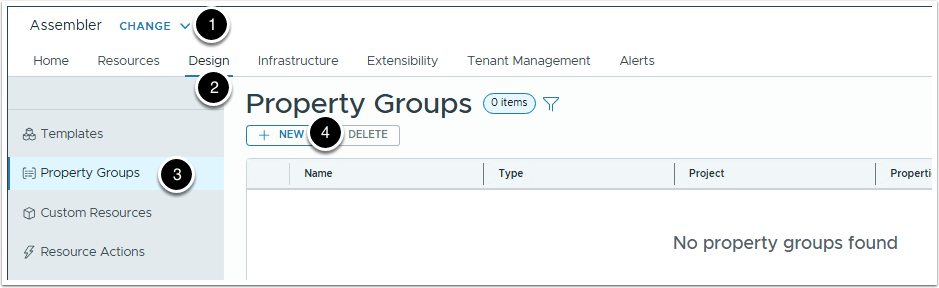




## Bonus- using property Groups as template input

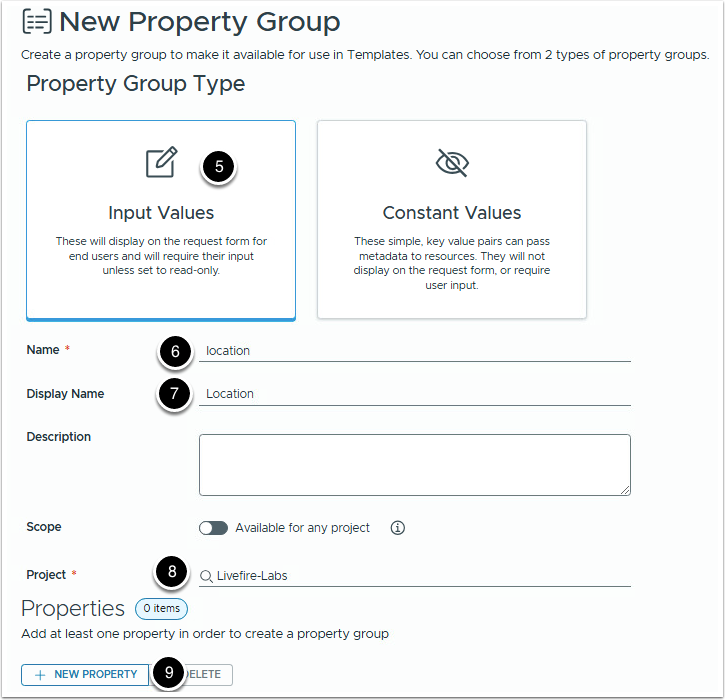
We are going to use an input property group to reference location and environment as input.

1. Go to Assembler.
2. Open Design tab.
3. Select Property Groups
4. Click +NEW to create new property group.



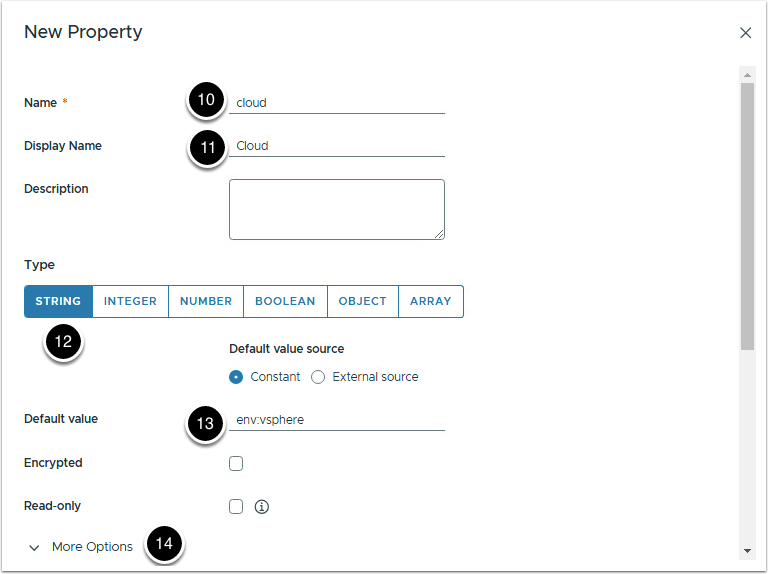
Create new property group of type Input Values with,

1. name: location
2. Display Name: Location
3. Project: Livefire-Labs
4. Click + NEW PROPERTY to add input property.

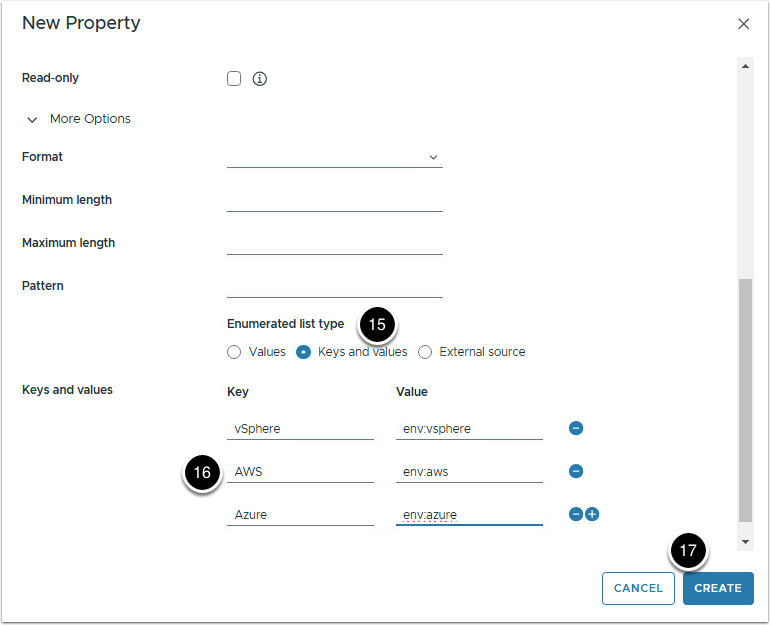


Create new input property with:

1. Name: cloud
2. Display Name: Cloud
3. Type: String
4. Default value: env:vsphere
5. Then click > More Options



1. Select Enumerated list type as Keys and Values
2. Enter below key value pairs:
3. Click Create



Create another property with,

Name: environment

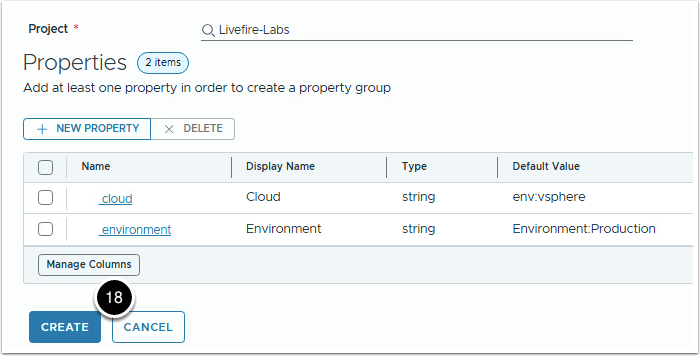
Display name: Environment

Type: String

Default value: Environment:Production

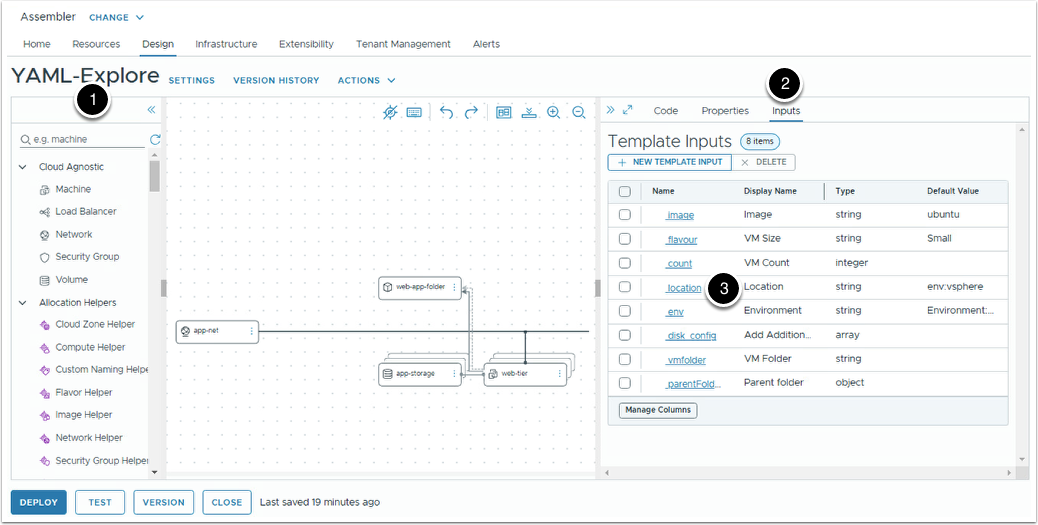
Key and value pairs:

1. Finally click Create to add the new property group



## Adding property group in template input.

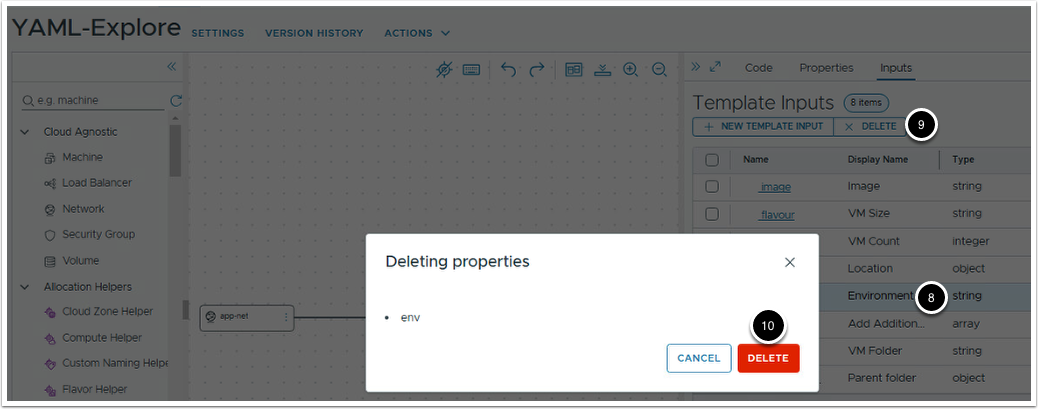
1. Navigate back to the template YAML-Explore
2. Select inputs section
3. Open location input



1. Change type from String to Object
2. Select object type to Property Groups
3. Select location property group
4. Click Save



1. Under inputs select env input checkbox
2. Click Delete
3. Click Delete again to confirm.



## Referencing property group in YAML code

Using property group in template is identical to using any input. We start with input namespace and then the input name

Then reference the exact property within the property group.

Using this principal lets change the input for both our constraint tags. And now the code for our resources should look like this.

## Cleanup (optional)

## Cleanup

Delete all deployments you created at the end of the lab.

