

HOL-2636-01-VCF-L -
ATE



Table of Contents

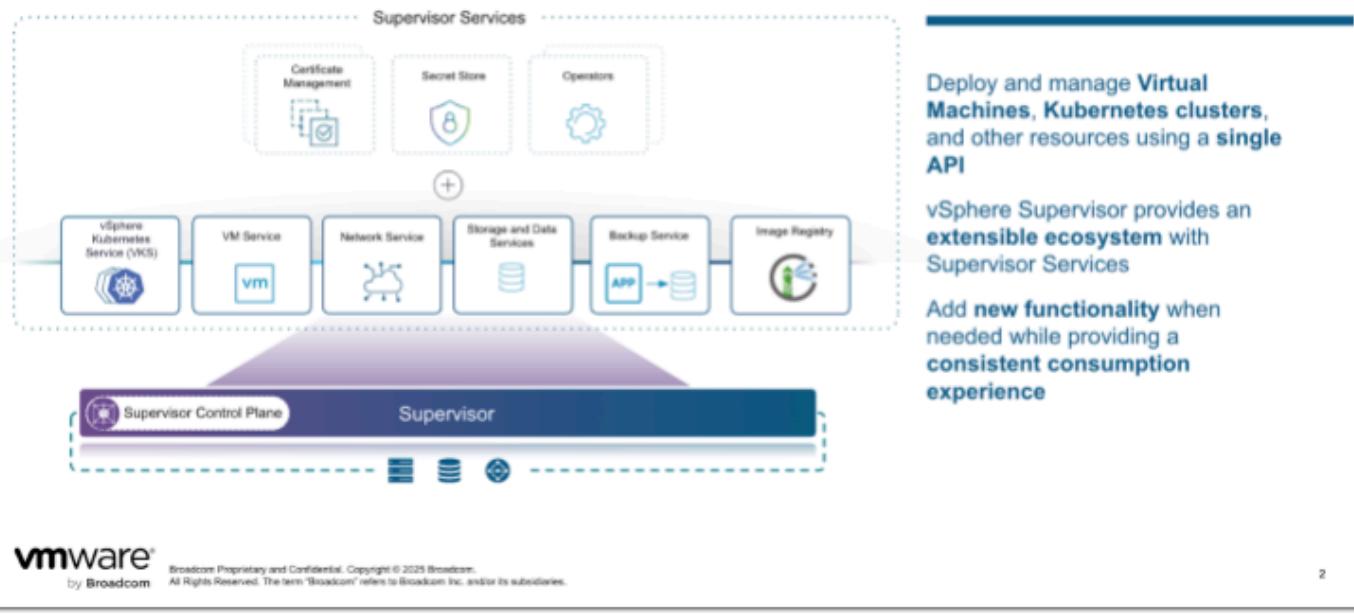
Lab Overview - HOL-2636-01-VLP-L: Managing Kubernetes and Cloud Services on VMware Cloud Foundation	4
Introduction.....	5
Module 1 - vSphere Supervisor concepts and components (15 minutes) Basic	8
Intro	9
vSphere Supervisor	10
vSphere Zone (Workloads)	11
vSphere Namespace	12
Kubernetes Namespace	13
Differences Between a vSphere Namespace and Kubernetes Namespace	14
Supervisor Networking	15
Supervisor Storage	16
Extensible through Supervisor Services.....	17
The control plane and declarative API.....	19
Module 2 -Enable and Configure vSphere Supervisor(30minutes) Basic	20
Enabling vSphere Supervisor	21
Deploy configure a namespace	34
Configure a Content Library.....	48
Create a VM	62
Deploy a Kubernetes Cluster (VKS)	75
Add services in a modular approach	84
Module 3 - Login to The Cluster and Deploy An Application (30 minutes) Basic	90
Module 3 - Login to The Cluster and Deploy An Application (30 minutes) Basic	91
Module 4 - Updating The Supervisor and VKS Services (15 minutes) Basic.....	101
Module 4 - Updating The Supervisor and VKS Services (15 minutes) Basic	102
Module 5 -Unlocking GitOps with ArgoCD Service(20 minutes) Advanced	107
Module 5 -Unlocking GitOps with ArgoCD Service(20 minutes) Advanced	108
Module 6 - Using ArgoCD to Manage VMs.....	125
Module 6 - Using ArgoCD to Manage VMs	126
Additional Information	167

Lab Overview - HOL-2636-01-VLP-L: Managing Kubernetes and Cloud Services on VMware Cloud Foundation

Introduction

Unlock Cloud Experience with vSphere Supervisor

Deliver a consistent Service consumption experience with an embedded Declarative API



- i** vSphere Supervisor delivers a consistent Service consumption experience with an embedded Declarative API. By using the integration between the Supervisor and VMware Cloud Foundation you can deploy and operate the compute, networking, and storage infrastructure for vSphere Supervisor (previously known as Workload Management). vSphere Supervisor transforms vSphere into a platform for running Kubernetes workloads natively on the hypervisor layer. When enabled on a vSphere cluster, the Supervisor provides the capability to deploy VMs using the VM Service, to run Kubernetes workloads directly on ESX hosts, and to create upstream Kubernetes clusters within dedicated resource pools.

Lab Guidance

Welcome! This lab is available for you to repeat as many times as you want. Use the Table of Contents in the upper right-hand corner of the Lab Manual to jump ahead to any module.

Module	Title	Length	Level
1	vSphere Supervisor concepts and components.	15 min	Beginner
2	Enable and configure the vSphere Supervisor	30 min	Beginner
3	Login to The Cluster and Deploy An Application	30 min	Basic
4	Updating The Supervisor And VKS Services	15 min	Basic
5	Unlocking Gitops with ArgoCD	20 min	Advanced

This lab has 2 Modules. The first module explains vSphere Supervisor concepts and how it interacts with vCenter and vSphere. In Module 2 you will interact with the Supervisor (previously known as Workload Management) by configuring the service and deploying both Kubernetes clusters and vSphere PODs with virtual machines.

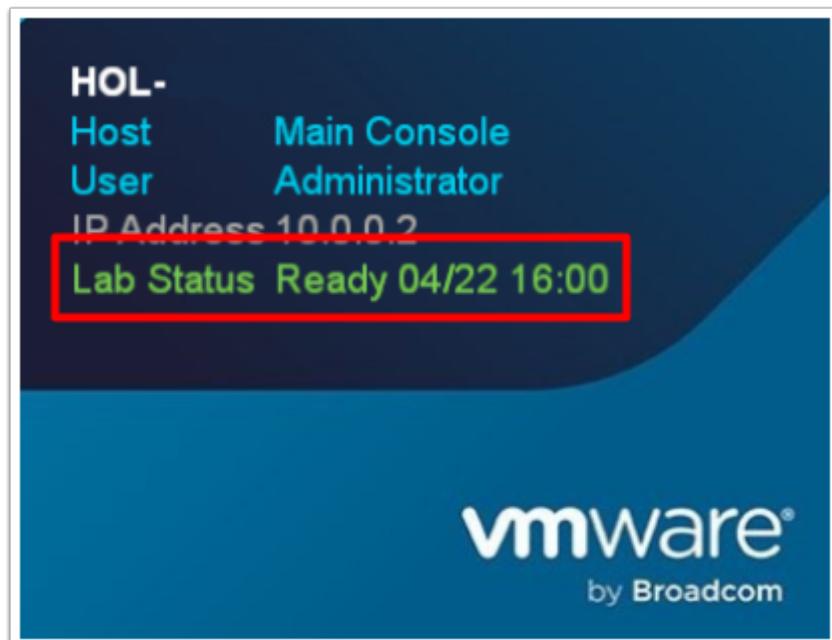
Lab Authors:

Principal: Kevin Brady

Captains: Katarina Brookfield, Randy Carson, Michael Fleisher, Pablo Iglesias

First Time Using Hands-on Labs?

If this is your first time taking a lab you can review the [VMware Learning Platform interface](#) before proceeding.



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.

Module 1 - vSphere Supervisor concepts and components (15 minutes) Basic

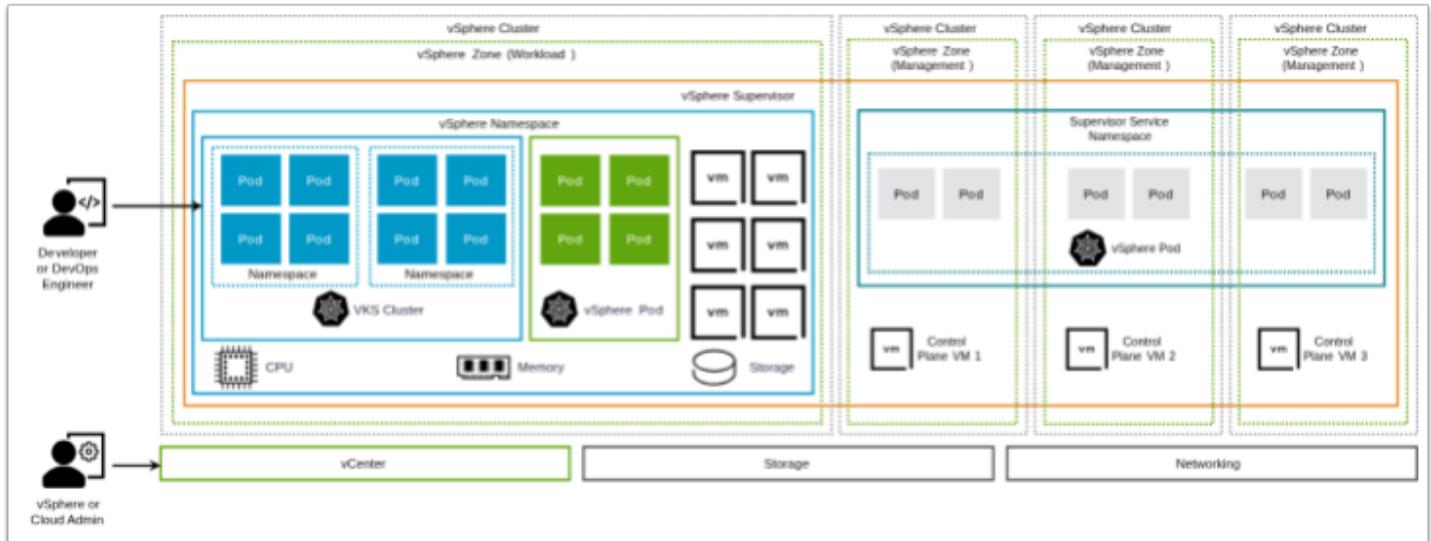
Intro

-  You can use vSphere Supervisor to enable a consistent declarative API in vSphere and create a platform for building a modern cloud experience with VCF Automation. When activated on vSphere clusters, the Supervisor provides the capability to run workloads in vSphere in declarative fashion, which includes creating and managing upstream Kubernetes clusters, virtual machines, and vSphere Pods . The resource boundaries where workloads run in the Supervisor are called vSphere Namespaces .

vSphere Supervisor supports Kubernetes 1.32, is bootstrapped with cluster API, and is decoupled from the vCenter lifecycle. That means you can upgrade Kubernetes based on your applications and not on vCenter upgrades.

vSphere Supervisor

vSphere Supervisor provides a declarative API and desired state by deploying a Kubernetes control plane directly on vSphere clusters. As a vSphere administrator, you activate existing vSphere clusters for the Supervisor, thus creating a Kubernetes layer within the clusters.



Let's define what some of the vSphere Supervisor constructs are. We start with a standard vSphere Cluster as a resource boundary for each Kubernetes environment needed for our business needs.

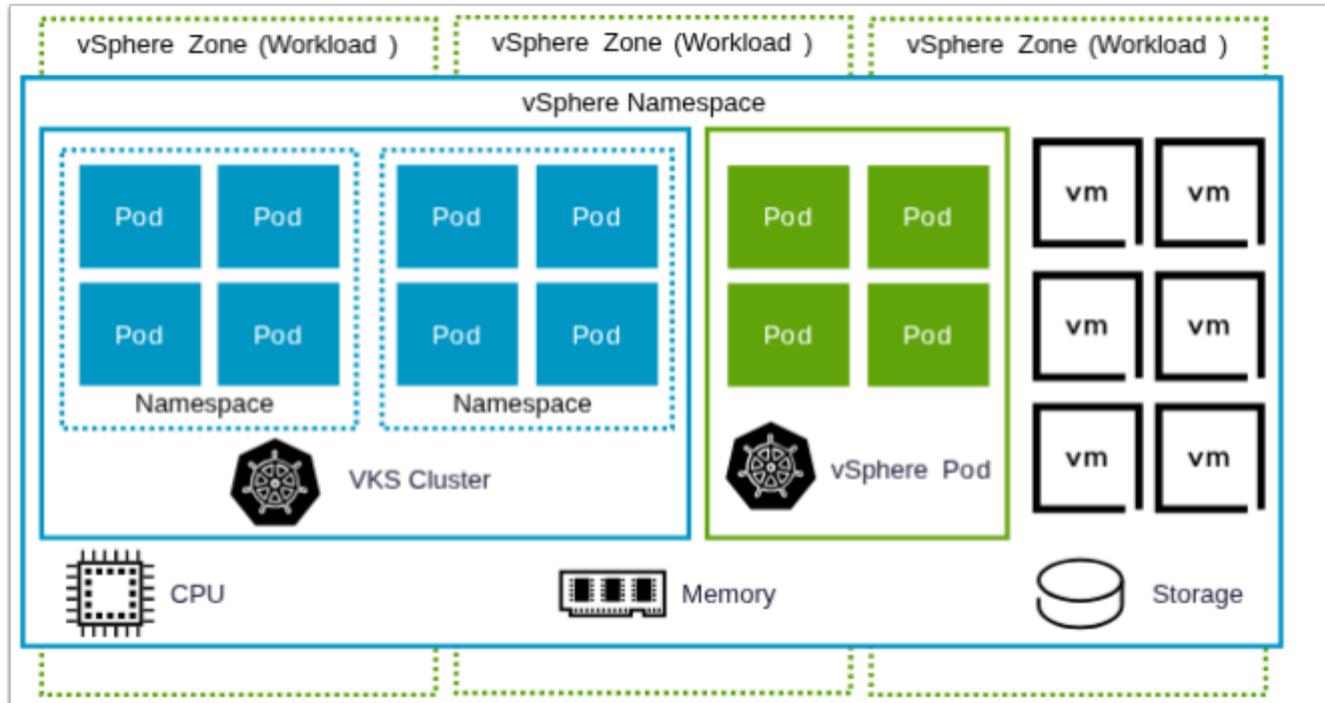
vSphere Zone (Workloads)

In the vSphere Supervisor, workloads are applications deployed in one of the following ways:

- Applications that consist of containers running inside vSphere Pods .
- Workloads provisioned through the VM service.
- Kubernetes clusters deployed by using vSphere Kubernetes Service (VKS)
- Applications that run inside VKS clusters.

vSphere Zones protect against cluster-level failure by providing high-availability to workloads deployed on the Supervisor . As a vSphere administrator, you create vSphere Zones in the vSphere Client and then map vSphere clusters to zones. There are 2 levels of high availability that you can provide with vSphere Zone. Creating a single vSphere Zone only gives you host level redundancy. To provide cluster level redundancy, you would create 3 vSphere Zones. This would spread the control plane VMs across these 3 zones giving you cluster level redundancy.

vSphere Namespace



Think of a vSphere Namespace as a Resource Pool that can contain vSphere Pods, VMs, and VKS clusters. A vSphere administrator can set limits for CPU, memory, storage, as well as the number of Kubernetes objects that can run within the vSphere Namespace.

When initially created, a vSphere Namespace has unlimited resources within the Supervisor. As a vSphere administrator, you can set limits for CPU, memory, storage, as well as the number of Kubernetes objects that can run within the vSphere Namespace. Storage limitations are represented as storage quotas in Kubernetes. A resource pool is created in vSphere per each vSphere Namespace on the Supervisor.

You can map up to three vSphere Zones to a vSphere Namespace for running workloads on them. The vSphere Namespace spreads across all vSphere clusters that are part of the vSphere Zones. The resources utilized on a vSphere Namespace are taken from all the underlying vSphere clusters on equal parts. For example, if you dedicate 300 MHz of CPU, 100 MHz are taken from each vSphere cluster.

Kubernetes Namespace

In Kubernetes, namespaces provide a mechanism for isolating groups of resources within a single cluster. Names of resources need to be unique within a namespace, but not across namespaces.

Namespaces are a way to organize clusters into virtual sub-clusters — they can be helpful when different teams or projects share a Kubernetes cluster. Any number of namespaces are supported within a cluster, each logically separated from others but with the ability to communicate with each other. Namespaces cannot be nested within each other.

Any resource that exists within Kubernetes exists either in the default namespace or a namespace that is created by the cluster operator. Only nodes and persistent storage volumes exist outside of the namespace; these low-level resources are always visible to every namespace in the cluster.

Differences Between a vSphere Namespace and Kubernetes Namespace

Although in its core a vSphere Namespace serves the same function as a Kubernetes namespace, a vSphere Namespace is specific to the vSphere Supervisor. You should not confuse a vSphere Namespace with a Kubernetes namespace.

A vSphere Namespace is implemented as an extension to a vSphere resource pool and its function is to provide resources to workloads running in the Supervisor. A vSphere Namespace has a direct mapping to a Kubernetes namespace through which object and storage quotas are enforced upon workloads.

Another difference with a regular Kubernetes namespace is that the vSphere administrator manages the user access to vSphere Namespaces, as mentioned from above. The vSphere administrator can also associate VM classes and Content Libraries containing VM templates that DevOps engineers can use to self-service VMs.

Supervisor Networking

A Supervisor can either use a vSphere networking stack or NSX to provide connectivity to Supervisor control plane VMs, services, and workloads. A Supervisor that is configured with the vSphere networking stack, all hosts from the Supervisor are connected to a VDS that provides connectivity to workloads and Supervisor control plane VMs. A Supervisor that uses the vSphere networking stack requires a load balancer on the vCenter management network to provide connectivity to DevOps users and external services. A Supervisor that is configured with NSX, uses virtualized networks of the solution and NSX Load Balancer or the Avi Load Balancer to provide connectivity to external services and DevOps users.

Supervisor Storage

A vSphere Zone is a fault domain that can fail independently from other zones. Supervisor supports different storage topologies for zones. These topologies define how datastores are attached to ESX hosts and how they behave during zone failures. Zonal datastore, a zonal datastore is local to a single vSphere Zone. It is attached to all the ESX hosts that are part of the underlying cluster. Cross-zonal datastore, a cross-zonal datastore spans multiple vSphere Zones. It is attached to ESX hosts from all the zones it covers.

Supervisor uses storage policies to integrate with storage available in your vSphere environment. The policies represent datastores and manage storage placement of such components and objects as control plane VMs, vSphere Pod ephemeral disks, and container images. You might also need policies for storage placement of persistent volumes and VM content libraries. If you use VKS clusters, the storage policies also dictate how the VKS cluster nodes are deployed. Storage policies support any shared datastores in your environment, such as VMFS, NFS, vSAN, and vSAN ESA. Depending on your vSphere storage environment and the needs of DevOps, you can create several storage policies for different classes of storage. When you enable a Supervisor and set up vSphere Namespaces, you can assign different storage policies to be used by various objects, components, and workloads.

Certain Kubernetes workloads that DevOps run on a namespace require persistent storage to store data permanently.

Persistent storage can be used by vSphere Pods , VKS clusters, VMs, and other workloads you run on the namespace. To make persistent storage available to the DevOps team, the vSphere administrator creates storage policies that describe different storage requirements and classes of services. The administrator then assigns storage policies and configures storage limits at a namespace level. To understand how the Supervisor works with persistent storage, be familiar with the such essential Kubernetes concepts as storage classes, persistent volumes, and persistent volume claims. For more information, see the Kubernetes documentation at <https://kubernetes.io/docs/home/>.

Extensible through Supervisor Services

The screenshot shows the 'Supervisor Management' interface in vSphere. The top navigation bar includes 'Namespaces', 'Supervisors', 'Services' (which is selected), 'Updates', and 'Content Distribution'. Below this, the 'Supervisor Services' section is displayed under the heading 'VC-WLD01-A-SITE-A.VCF.LAB'. A note states: 'Supervisor Services is a platform for managing core infrastructure components, such as virtual machines. Application teams are able to deploy instances of Supervisor Services within their own Namespaces using industry standard tools and practices. Discover and download available Supervisor Services here.' Below this, a 'Sort By:' dropdown is set to 'Recently added'. A message indicates: 'Below are the services registered to this vCenter Server system. You can manage services with multiple versions from the same service card.' The main area contains several service cards:

- VM Service**: This service allows developers to self-service VMs and allows you to set policies for VM deployment. It has a 'MANAGE' button.
- Local Consumption Interface**: Provides the Local Consumption Interface for N... Status: Active (Active Versions 1, Supervisors 1). It has an 'ACTIONS' dropdown.
- ArgoCD Service**: This service allows users to self-service ArgoCD ... Status: Active (Active Versions 1, Supervisors 1). It has an 'ACTIONS' dropdown.
- Velero vSphere Operator**: Helps users install Velero and the vSphere plugi... Status: Active (Core Service, Active Versions 2, Supervisors 1). It has an 'ACTIONS' dropdown.
- Kubernetes Service**: Cluster management Status: Active (Core Service, Active Versions 1, Supervisors 1). It has an 'ACTIONS' dropdown.

A sidebar on the left shows a 'harbor' service card with status 'Active' (Active Versions 1, Supervisors 1) and an 'OCI Registry' section. There are also 'ADD' and 'ACTIONS' buttons.

vSphere Supervisor has a catalog of Services that can be added once it is enabled. There are 3 services that are installed by default when you enable the Supervisor. Those services are VM Service, Kubernetes Service, and Velero vSphere Operator Service. These base services allow you to create vSphere PODs (for virtual machine deployments), Kubernetes clusters (VKS), and be able to backup your kubernetes environment with Velero.

Some of the other services that can be added to the Supervisor are Certificate Management, DNS, and vSAN Data Persistence Platform (vDPP) to mention a few. As you can see in the screen shot we have also added 2 services to the default install, Contour and Harbor. We added these 2 services so we could have an off-line image repository for use later in this lab. In Module 2 we will walk you through adding a Grafana Operator service to the vSphere Supervisor.

The screenshot shows the Service Catalog interface for a namespace named 'svc-tkg-domain-c10'. The top navigation bar includes links for Summary, Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The 'Resources' tab is selected. Below the navigation, there are four main resource categories displayed in cards:

- Virtual Machine**: Shows 0 Virtual Machines. Buttons: GO TO SERVICE, CREATE VM.
- Network**: Shows 21 Network Services. Buttons: GO TO SERVICE.
- Virtual Machine Image**: Shows 89 VM Images. Buttons: GO TO SERVICE.
- Kubernetes**: Shows 0 Kubernetes Clusters. Buttons: GO TO SERVICE, CREATE CLUSTER.

Below these cards, there are two additional sections:

- Volume**: Shows 0 Volumes. Buttons: GO TO SERVICE, CREATE VOLUME.

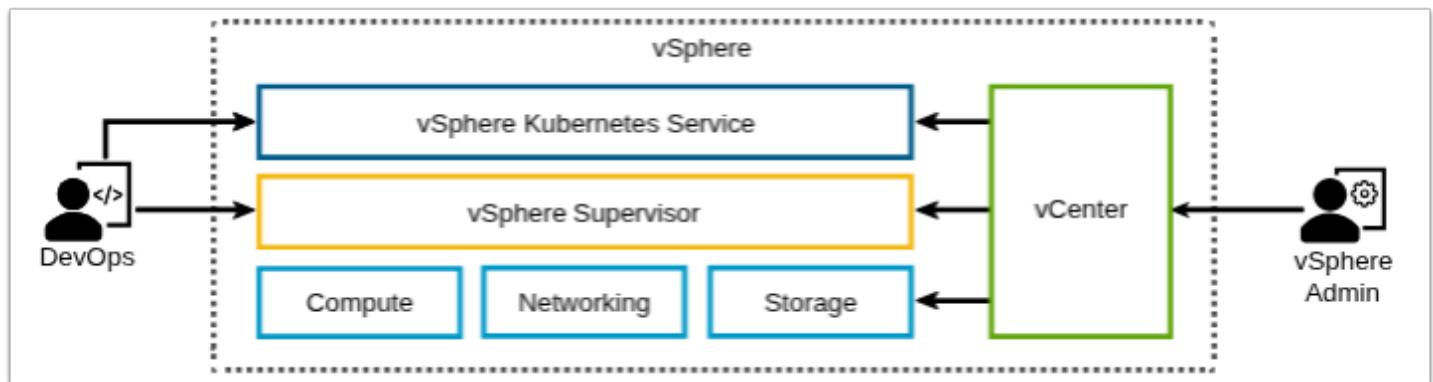
A very useful service to add to the Supervisor is the **Local Consumption Interface**. This service gives you a great picture into the resources being used in a namespace. You can also create **VMs, Kubernetes Clusters (VKS)**, and storage volumes as well as add to some existing service for that particular namespace.

As you can see in the screen shot we have added 2 services to the default install, **Contour** and **Harbor**. We added these 2 services so we could have an off-line image repository. We will walk you through adding the **Grafana Operator** services to this **Supervisor**.

The control plane and declarative API

The control plane and declarative API

When you activate a Supervisor on vSphere Zones, a Kubernetes control plane is created. This control plane enables the capability to run workloads in a declarative fashion on vSphere clusters.



A Supervisor runs on top of an SDDC layer that consists of ESX for compute, NSX or vSphere Distributed Switch networking, and vSAN or another shared storage solution. Shared storage is used for persistent volumes for vSphere Pods and VMs running inside the Supervisor, and pods in a VKS cluster. After a Supervisor is created, as a vSphere administrator you can create vSphere Namespaces within the Supervisor. As a DevOps engineer, you can run workloads consisting of containers running inside vSphere Pods, deploy VMs through the VM service, and create VKS clusters.

Module 2 -Enable and Configure vSphere Supervisor(30minutes) Basic

Enabling vSphere Supervisor

 We will not enable vSphere Supervisor in this section. We will walk through the steps but not actually create one. The creation process for the Supervisor takes longer than this lab allows

In this module, we will walk through the process of enabling the Supervisor on the Management vCenter. You can only enable 1 supervisor per cluster and in this lab the supervisor on the workload cluster `wld-01a` has been enabled for you to use in this lab.

A vSphere Zone Storage Policy are prerequisites for Enabling the Supervisor

vSphere Cluster

To activate a Supervisor, you must create at least one vSphere Zone that will act as the Management Zone of the Supervisor. To provide zone-level HA to the Supervisor control plane, you deploy the Supervisor on three Management Zones. In Supervisor with three Management Zones and HA for the control plane, meaning with three control plane VMs, each control plane VM is placed on a different Management Zone.

- Create one or three vSphere clusters with at least 3 hosts in each zone. For storage with vSAN, the cluster must have 4 hosts. For testing of POC environments, each cluster must have at least 1 host, and 2 hosts when using vSAN.
- Configure storage with vSAN or other shared storage solution for each cluster.
- Enable vSphere HA and vSphere DRS on Fully Automate or Partially Automate mode.
- All clusters must be connected to the same VDS and Management traffic network.

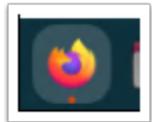
Storage Policy

Before you activate the Supervisor, create storage policies to be used in the Supervisor and vSphere Namespaces. The policies represent datastores and manage storage placement of such components and objects as Supervisor control plane VMs, vSphere Pod ephemeral disks, and container images. You might also need policies for storage placement of persistent volumes and VM content libraries. If you use VKS clusters, the storage policies also dictate how the VKS cluster nodes are deployed.

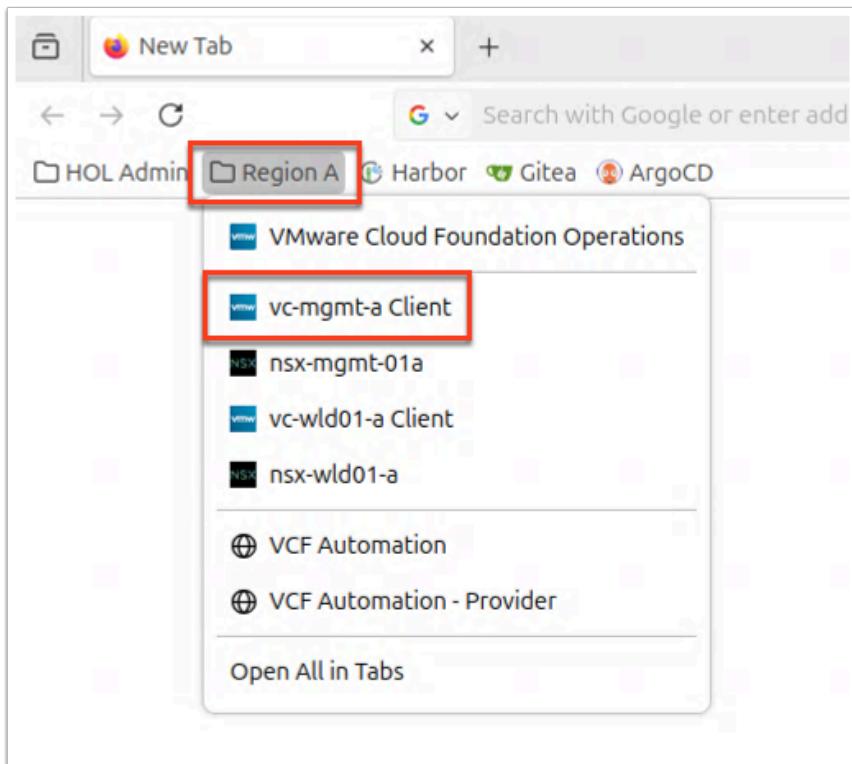
Depending on your vSphere storage environment and the needs of DevOps, you can create several storage policies for different classes of storage. For example, if your vSphere storage environment has three classes of datastores, Bronze, Silver, and Gold, you can create storage

policies for all datastore types. When you enable a Supervisor and set up namespaces, you can assign different storage policies to be used by various objects, components, and workloads.

Connect to the Management vCenter

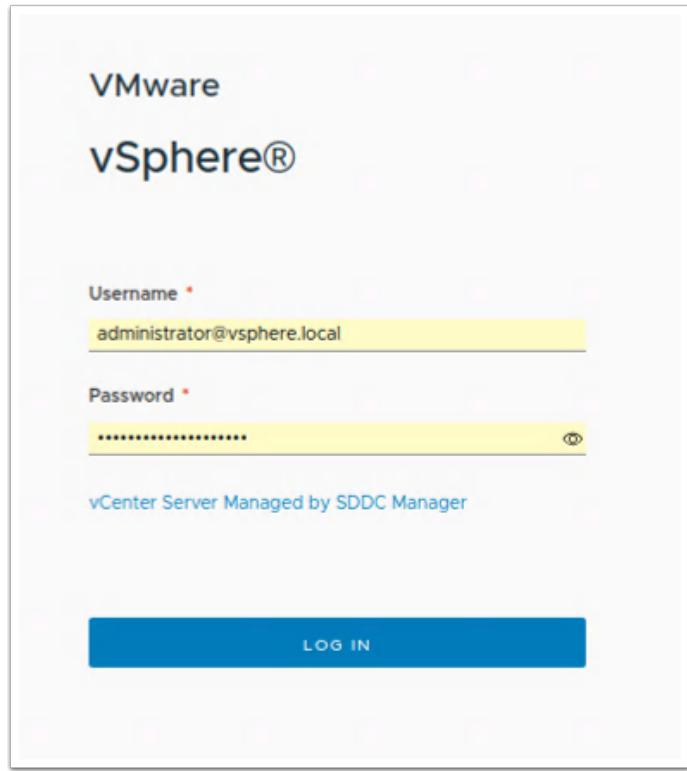


1. Open Firefox from the taskbar at the bottom of the screen



2. Select **Region A** tab
3. Select **vc-mgmt-a** Client from the **Region A** tab

Login to vSphere



Log in with:

Username:

Administrator@vsphere.local

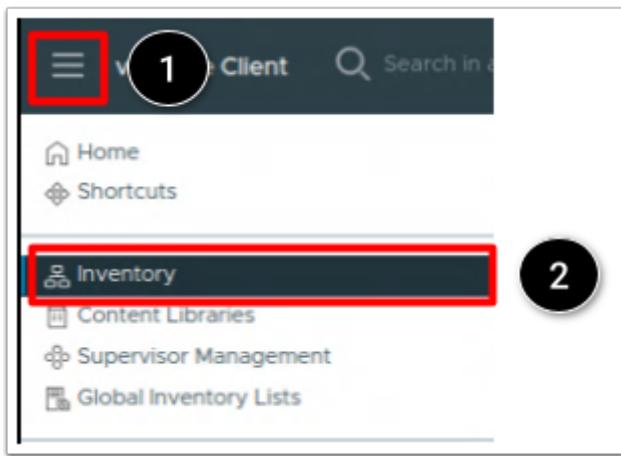
Click to copy

Password:

VMware123!VMware123!

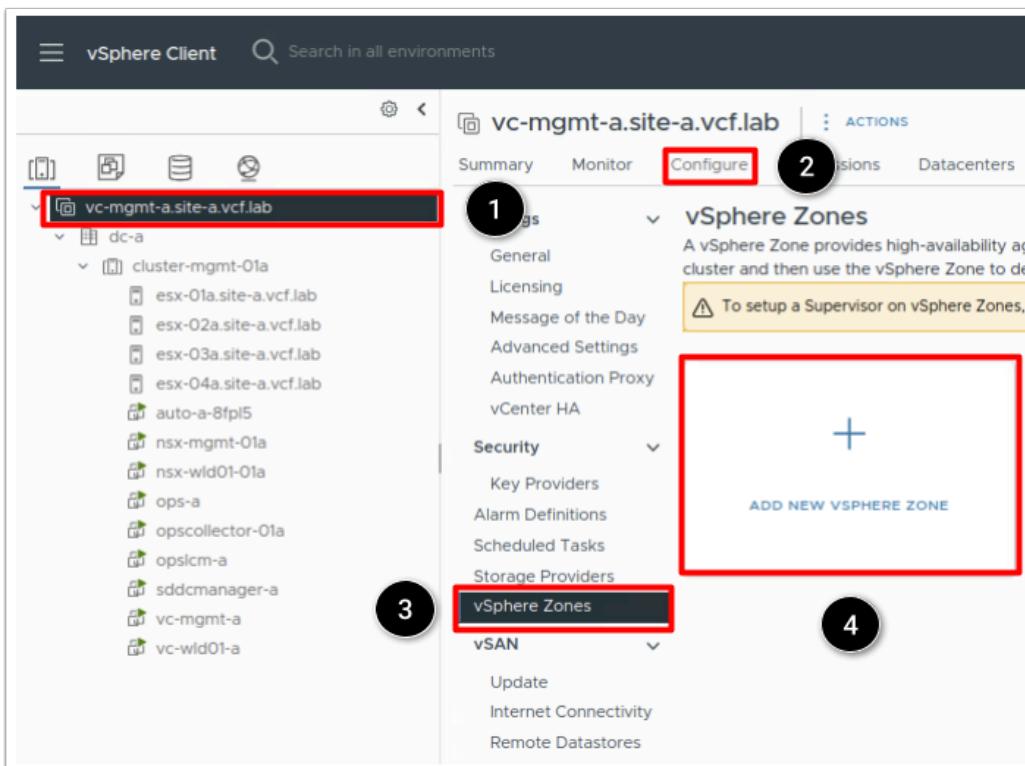
Click to copy

Create a vSphere Zone



1. Select the **Menu** in the upper left corner
2. Select **Inventory**

Create a New vSphere Zone



We need to set up at least 1 vSphere Zone to be able to activate the Supervisor.

1. Select the vCenter **vc-mgmt-a.site-a.vcf.lab**.
2. Select **Configure**
3. Select **vSphere Zones**
4. Select **ADD NEW VSphere ZONE**

Set Up the vSphere Zone

Set up vSphere Zones

1. vSphere Zone creation Enter name and description for the vSphere Zone

Fields marked with * are required

vSphere Zone name * 1

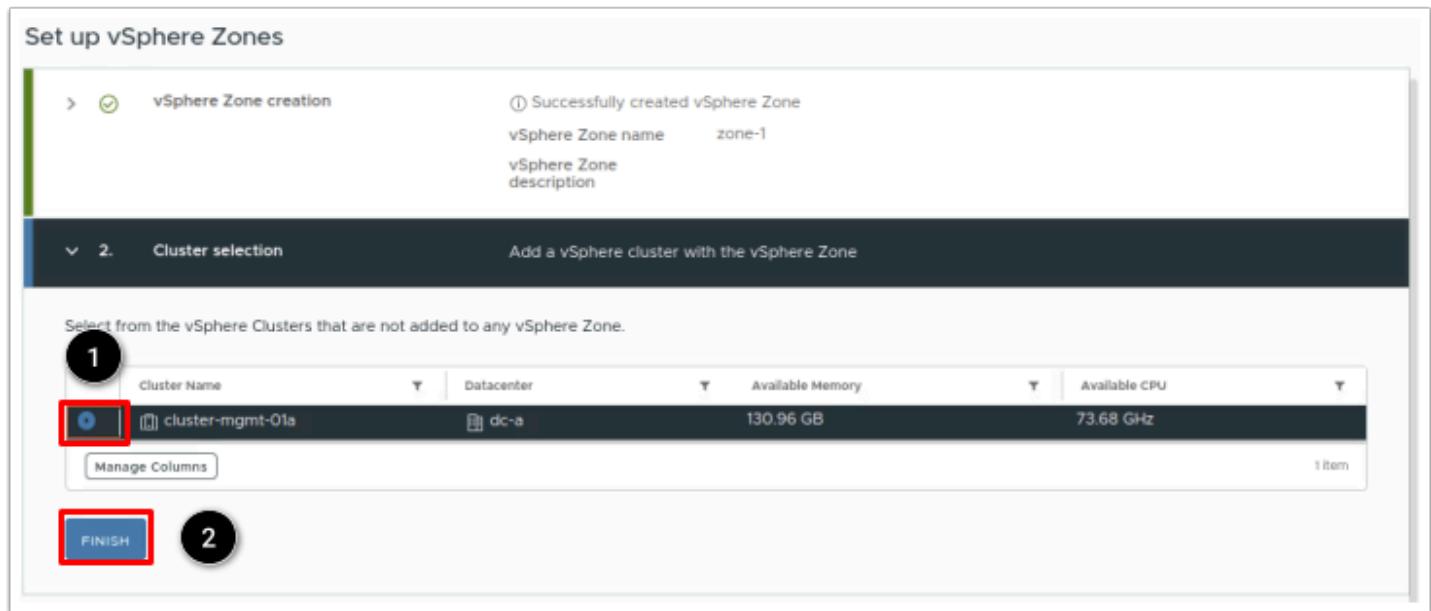
vSphere Zone description

NEXT 2

2. Cluster selection Add a vSphere cluster with the vSphere Zone

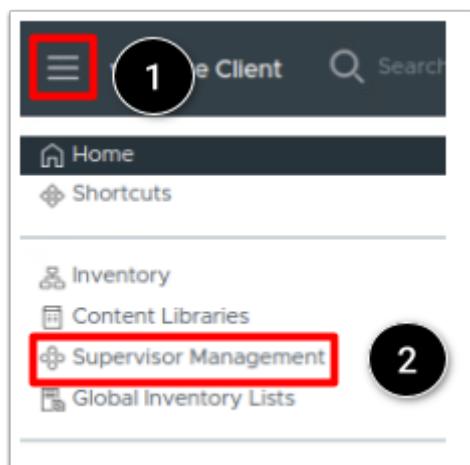
1. Type the name zone-1. (There are some rules to follow when naming your zone. To see those rules click the blue "i" next to vSphere Zone name.)
2. Select **Next**

Select the Clusters



1. Select **cluster-mgmt-01a**. You will want to pick the cluster that you want to use for the *VKS Cluster vSphere Pods*.
2. Select **Finish**

Start the Supervisor create wizard



1. Select the **Menu** in the upper left corner
2. Select **Supervisor Management**

Get Started in the Wizard

Supervisor Management

Supervisor Management enables deploying and managing Kubernetes workloads in vSphere. By using Supervisor Management, you can leverage both Kubernetes and vSphere functionality. Once you configure a vSphere cluster for Supervisor Management and it becomes a Supervisor, you can create namespaces that provide compute, networking, and storage resources for running your Kubernetes applications. You can also configure Supervisors with policies for resource consumption.

[Learn more about Supervisor Management](#)

[GET STARTED](#)

[GET STARTED WITH CONFIG](#)

We will be walking through the setup wizard. If you already had a JSON config file from a previous setup you could use that file to *declare* the state of the Supervisor deployment.

1. Click **GET STARTED**.

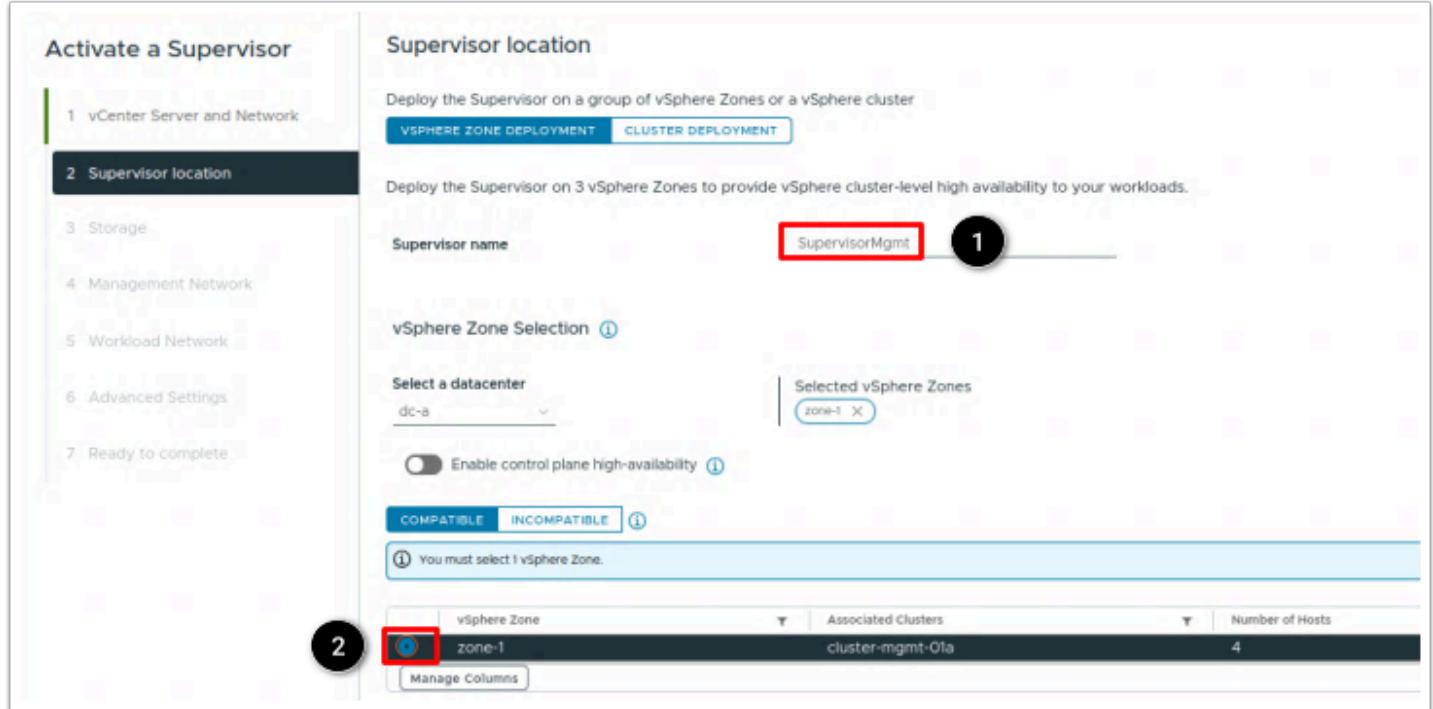
Select the Networking Stack

Activate a Supervisor 1 vCenter Server and Network 2 Supervisor location 3 Storage 4 Management Network 5 Advanced Settings 6 Ready to complete	vCenter Server and Network Select a vCenter Server system and a network to set up a Supervisor. Select a vCenter Server system (i) <input checked="" type="checkbox"/> VC-MGMT-A.SITE-A.VCF.LAB (SUPPORTS NSX) ▾ Select the networking stack that will provide connectivity to this Supervisor. Select a networking stack <input type="radio"/> VCF Networking with VPC <input checked="" type="radio"/> NSX Classic <input type="radio"/> vSphere Distributed Switch (VDS)
--	--

Select the best underlying networking for your supervisor to use when creating *VKS cluster* and *vSphere PODs*. For this module we will be using NSX networking

1. Select **NSX Classic**
2. Select **Next**(not shown)

Configure Supervisor Location



In this step, we are going to use the zone that we created in the previous step.

1. Enter the **Supervisor Name:** *SupervisorMgmt*
2. Select the available **vSphere Zone:** *zone-1*. This was previously created as one of the prerequisites.
3. Click **Next**(not shown).

Configure Storage

Activate a Supervisor

1 vCenter Server and Network
2 Supervisor location
3 Storage
4 Management Network
5 Workload Network

Storage

Select the storage policy for the control plane VMs on this Supervisor.

Select a storage policy to be used for datastore placement of Supervisor control plane VMs. The policy will be applied to zone-1.

Control Plane Storage Policy K8S Storage Policy 1 [VIEW DETAILS](#)
This policy will be applied to zone-1

Ephemeral Disks Storage Policy Select a policy [VIEW DETAILS](#)
This policy is disabled for zone based deployments

Image Cache Storage Policy Select a policy [VIEW DETAILS](#)
This policy is disabled for zone based deployments

We will only need to select the Control Plane Storage Policy.

1. Select **K8S Storage Policy** from the drop down menu. This was previously created as one of the prerequisites.
2. Select **Next(not shown)**

Configure Management Network

We need to enter the network values for the Management Network that contains the 3 control plan VMs.

These static IP addresses are specific to this HOL lab.

Activate a Supervisor

- 1 vCenter Server and Network
- 2 Supervisor location
- 3 Storage
- 4 Management Network**
- 5 Workload Network
- 6 Advanced Settings
- 7 Ready to complete

Management Network

A Supervisor contains up to 3 control plane VMs. The Management Network provides connectivity between the vCenter Server and the Supervisor.

Network Mode	1 Static
Network	2 mgmt-vds01-mgmt-01a
IP Addresses	3 10.1.1.90-10.1.1.95
Subnet Mask	4 255.255.255.0
Gateway	5 10.1.1.1
DNS Server(s)	6 10.1.1.1
DNS Search Domain(s)	7 site-a.vcf.lab
NTP Server(s)	10.1.1.1

1. Change the Network Mode to **Static**
2. Select the Network **mgmt-vds01-mgmt-01a**
3. Input the IP address for the control plane VMs, type in **10.1.1.90-10.1.1.95**
4. Enter the **Subnet Mask:** 255.255.255.0
5. Enter the **Gateway:** 10.1.1.1
6. Enter the **DNS Server:** 10.1.1.1
7. Enter the **DNS Search Domain:** site-a.vcf.lab
8. Select Next

Configure Workload Network

Activate a Supervisor

- 1 vCenter Server and Network
- 2 Supervisor location
- 3 Storage
- 4 Management Network
- 5 Workload Network**
- 6 Advanced Settings
- 7 Ready to complete

Workload Network

Workload Networks provide connectivity to applications running in vSphere Namespace: additional Workload Networks to vSphere Namespaces.

The Workload Network supports traffic to the Kubernetes API and to the Workloads/Ser

vSphere Distributed Switch (1)

DNS Server(s) (1) 10.1.1.1 (1)

Tier-0 Gateway (1)

NAT Mode (1) Enabled

Subnet Prefix (1) /28

Namespace Network (1) 10.244.0.0/20

Service CIDR (1) 10.96.0.0/23

Ingress CIDRs (1) 10.2.0.0/24 (2)

Egress CIDRs (1) 172.16.200.64/24 (3)

We need to enter the Network values for the Workload Network. These IP addresses will be used for the VKS clusters and vSphere PODs.

These static IP addresses are specific to this HOL lab.

1. Enter the **DNS Server:** 10.1.1.1
2. Enter the **Ingress CIDRs:** 10.2.0.0/24
3. Enter the **Egress CIDRs:** 172.16.200.64/24
4. Select **Next**(not shown)

Configure Advanced Settings

The screenshot shows the 'Activate a Supervisor' wizard with the following steps:

- 1 vCenter Server and Network
- 2 Supervisor location
- 3 Storage
- 4 Management Network
- 5 Workload Network
- 6 Advanced Settings (highlighted in dark blue)

In the 'Advanced Settings' step, there are two main configuration sections:

- Supervisor Control Plane Size**: A dropdown menu currently set to 'Small (CPUs: 4, Memory: 16 GB, Storage: 48 GB)'. A small circular callout with the number '1' is positioned above this section.
- API Server DNS Name(s)**: An optional input field with placeholder text 'E.g. server.yourdomainname.com'.

A red rectangular box highlights the **Export configuration** checkbox, which is checked. This checkbox is located below the API server DNS name input field.

For this lab module we will select the small size. If you select the drop down arrow, you can see the different control plane sizes. Be sure to select the size that fits best for your applications.

1. Select the **Export configuration** check box.
1. Select **Next(not shown)**

A json file will be downloaded. This file can be modified and imported on a future supervisor deployment

Review Summary

Activate a Supervisor <ul style="list-style-type: none"> 1 vCenter Server and Network 2 Supervisor location 3 Storage 4 Management Network 5 Workload Network 6 Advanced Settings <p>7 Ready to complete</p>	<p>Ready to complete</p> <p>Review your selections before finishing the wizard</p> <p>vCenter Server and Network</p> <table border="0"> <tr> <td>vCenter Server</td> <td>vc-mgmt-a.site-a.vcf.lab</td> </tr> <tr> <td>Network</td> <td>NSX Classic</td> </tr> </table> <p>Supervisor location</p> <table border="0"> <tr> <td>Supervisor Name</td> <td>SupervisorMgmt</td> </tr> <tr> <td>vSphere Zone Selection</td> <td>zone-1</td> </tr> <tr> <td>Control plane high-availability</td> <td>Disabled</td> </tr> </table> <p>Storage</p> <table border="0"> <tr> <td>Control Plane Storage Policy</td> <td>K8S Storage Policy</td> </tr> <tr> <td>Ephemeral Disks Storage Policy</td> <td>K8S Storage Policy</td> </tr> <tr> <td>Image Cache Storage Policy</td> <td>K8S Storage Policy</td> </tr> </table> <p>Management Network</p> <table border="0"> <tr> <td>Network Mode</td> <td>Static</td> </tr> <tr> <td>Network</td> <td>mgmt-vds01-mgmt-01a</td> </tr> <tr> <td>IP Addresses</td> <td>10.1.1.90 - 10.1.1.95</td> </tr> <tr> <td>Subnet Mask</td> <td>255.255.255.0</td> </tr> <tr> <td>Gateway</td> <td>10.1.1.1</td> </tr> <tr> <td>DNS Server(s)</td> <td>10.1.1.1</td> </tr> <tr> <td>DNS Search Domain(s)</td> <td>site-a.vcf.lab</td> </tr> <tr> <td>NTP Server(s)</td> <td>10.1.1.1</td> </tr> </table>	vCenter Server	vc-mgmt-a.site-a.vcf.lab	Network	NSX Classic	Supervisor Name	SupervisorMgmt	vSphere Zone Selection	zone-1	Control plane high-availability	Disabled	Control Plane Storage Policy	K8S Storage Policy	Ephemeral Disks Storage Policy	K8S Storage Policy	Image Cache Storage Policy	K8S Storage Policy	Network Mode	Static	Network	mgmt-vds01-mgmt-01a	IP Addresses	10.1.1.90 - 10.1.1.95	Subnet Mask	255.255.255.0	Gateway	10.1.1.1	DNS Server(s)	10.1.1.1	DNS Search Domain(s)	site-a.vcf.lab	NTP Server(s)	10.1.1.1
vCenter Server	vc-mgmt-a.site-a.vcf.lab																																
Network	NSX Classic																																
Supervisor Name	SupervisorMgmt																																
vSphere Zone Selection	zone-1																																
Control plane high-availability	Disabled																																
Control Plane Storage Policy	K8S Storage Policy																																
Ephemeral Disks Storage Policy	K8S Storage Policy																																
Image Cache Storage Policy	K8S Storage Policy																																
Network Mode	Static																																
Network	mgmt-vds01-mgmt-01a																																
IP Addresses	10.1.1.90 - 10.1.1.95																																
Subnet Mask	255.255.255.0																																
Gateway	10.1.1.1																																
DNS Server(s)	10.1.1.1																																
DNS Search Domain(s)	site-a.vcf.lab																																
NTP Server(s)	10.1.1.1																																

On the Summary page

1. Select **Cancel**(not shown)

We are **not finishing this** install. We will be using the Supervisor already configured on the **vc-wld01-a vCenter**

Close the firefox browser by closing the vSphere Client tab for **vc-mgmt-a.site-a.vcf.lab vCenter**.

Deploy configure a namespace

We will walk through the process of creating a vSphere Namespace on the Workload vCenter.

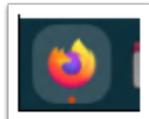
After creating a namespace, we will at least need to:

- Add VM Classes that will specify the size for the VMs we will be able to create (standalone or VKS VMs)
- Add Storage policies for volumes to be consumed by them and their internal components such as PODs

NOTE: We will not set specific permissions or create Resource and Object quotas or limits for our namespace in this section.

The next section will explain how to create and associate content libraries to our newly created namespace.

Open Firefox



1. Open Firefox from the taskbar at the bottom of the screen

Connect to the Workload vCenter (vc-wld01-a Client)

The screenshot shows a web-based interface for VMware Cloud Foundation Operations. At the top, there's a navigation bar with icons for Admin, Region A (which is highlighted with a red box), Harbor, Gitea, and ArgoCD. Below this is a search bar with the placeholder "Search with Google or enter URL". The main content area is a sidebar with several items listed:

- VMware Cloud Foundation Operations
- vc-mgmt-a Client
- nsx-mgmt-01a
- vc-wld01-a Client** (this item is also highlighted with a red box)
- nsx-wld01-a
- VCF Automation
- VCF Automation - Provider

At the bottom of the sidebar, there's a link "Open All in Tabs".

2. Select **Region A** tab
2. Select **vc-wld01-a Client** from the **Region A** tab

Login to vSphere

The screenshot shows the vSphere login page. The title "vSphere®" is displayed above the login fields. There are two input fields: "Username *" containing "administrator@wld.sso" and "Password *". Both fields are highlighted with a red box. Below the password field is a "Forgot Password?" link. At the bottom of the page is a large blue "LOG IN" button, which is also highlighted with a red box.

Log in with:

Username:

Administrator@wld.sso

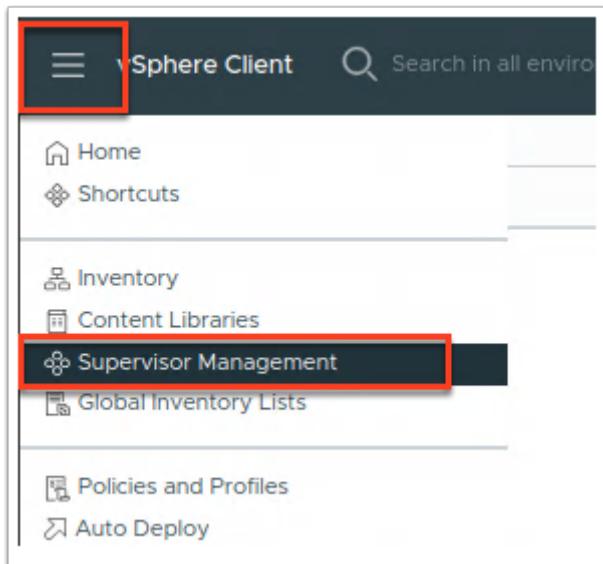
Click to copy

Password:

VMware123!VMware123!

Click to copy

Create a vSphere Namespace



1. Select the **Menu** in the upper left corner
2. Select **Supervisor Management**

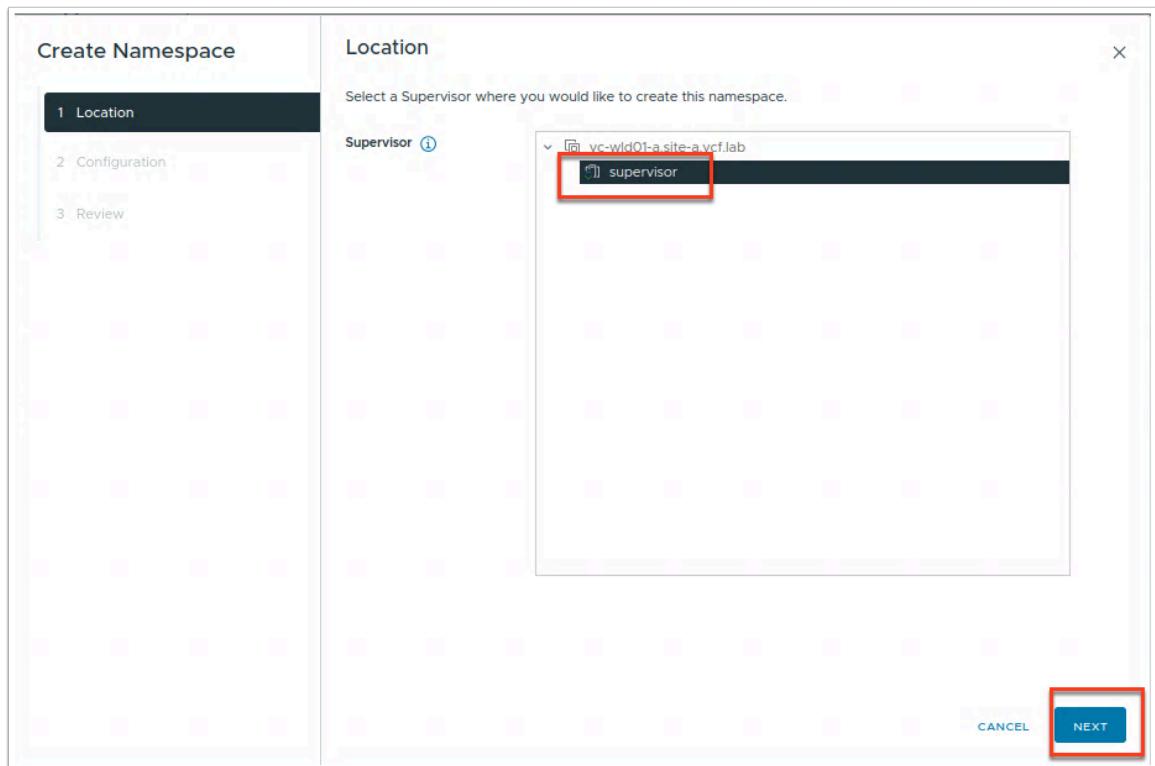
Select Namespace

The screenshot shows the Supervisor Management interface. On the left, there's a sidebar with a 'Namespaces' section containing several items: argocd, svc-argocd-service-domain-c10, svc-cci-ns-domain-c10, svc-harbor-domain-c10, svc-tkg-domain-c10, and svc-velero-domain-c10. The 'Namespaces' tab is highlighted with a red box. On the right, the main panel has tabs for Namespaces, Supervisors, Services, Updates, and Content Distribution. The 'Namespaces' tab is active. Below it is a 'NEW NAMESPACE' button, also highlighted with a red box. A table lists namespaces with their supervisors and status. The table has columns for Namespace Name, Supervisor, and Config Status. All listed namespaces are in a 'Running' state.

	Namespaces	Supervisor	Config Status
○	(≡) argocd	supervisor	✓ Running
○	(≡) svc-argocd-service-domain-c10	supervisor	✓ Running
○	(≡) svc-cci-ns-domain-c10	supervisor	✓ Running
○	(≡) svc-harbor-domain-c10	supervisor	✓ Running
○	(≡) svc-tkg-domain-c10	supervisor	✓ Running
○	(≡) svc-velero-domain-c10	supervisor	✓ Running

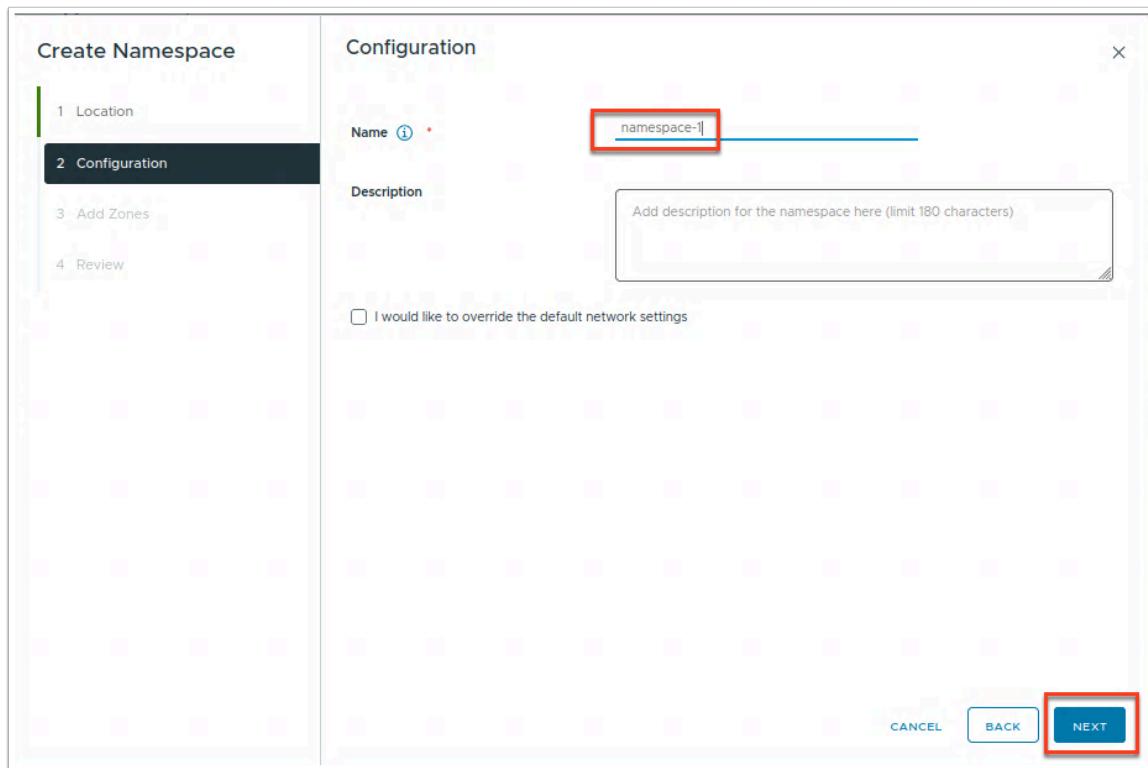
1. Select the main **Namespaces** element
2. Select the **Namespaces** tab
3. Click **NEW NAMESPACE**

Select the Supervisor For The Namespace



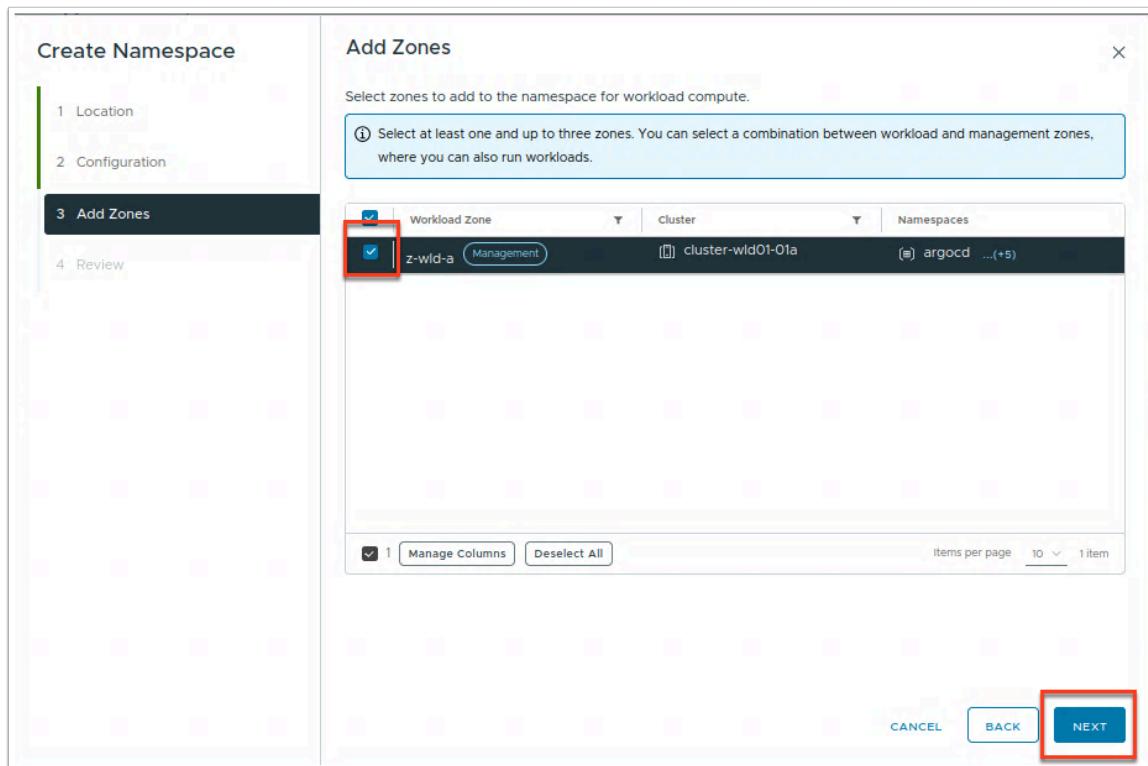
1. In the Create Namespace window, select **Supervisor**
2. Click **NEXT**

Create a Name For The New Namespace



1. Type the name of your namespace. In this case, please enter **namespace-1**. There are some rules to follow when naming your namespace. To see those rules click the blue "i" next toName.
2. Click **NEXT**

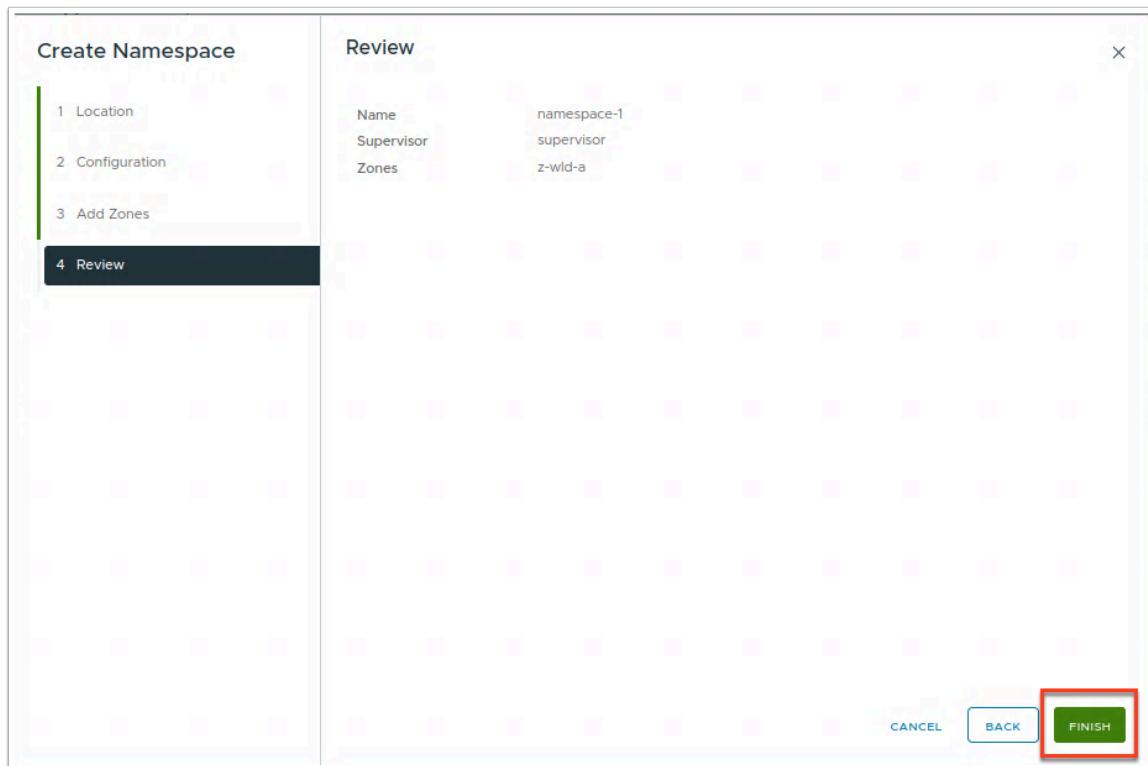
Add Zones



We need to set up at least 1 vSphere Zone to be able to create our Namespace.

1. Select Radio check box for **Workload Zone: z-wld-a**
2. Click **NEXT**

Review And Accept



1. Review the configuration and click **FINISH**

View The New Namespace

The screenshot shows the summary page for namespace-1. It includes tabs for Summary, Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The Summary tab is selected. On the left, there are sections for VM Service (0 Associated VM Classes, ADD VM CLASS) and Associated Content Libraries (0 Associated Content Libraries, ADD CONTENT LIBRARY). In the center, there is a 'Pods' section showing 0 pods. On the right, a detailed status box is highlighted with a red border, containing the following information:

- Status: Created 9/22/25
- Config Status: **Running**
- Kubernetes Status: **Active**
- Location: supervisor, vc-wld01-a.site-a.vcf.lab
- Link to CLI Tools: [Copy link](#) [Open](#)

1. In the new **namespace-1** Summary page, check that the Status is **Running and Active**. The status may show Creating for a minute or two.

Configure a vSphere Namespace

After having created our first namespace **namespace-1**, we will now associate VM Classes and Storage Policies.

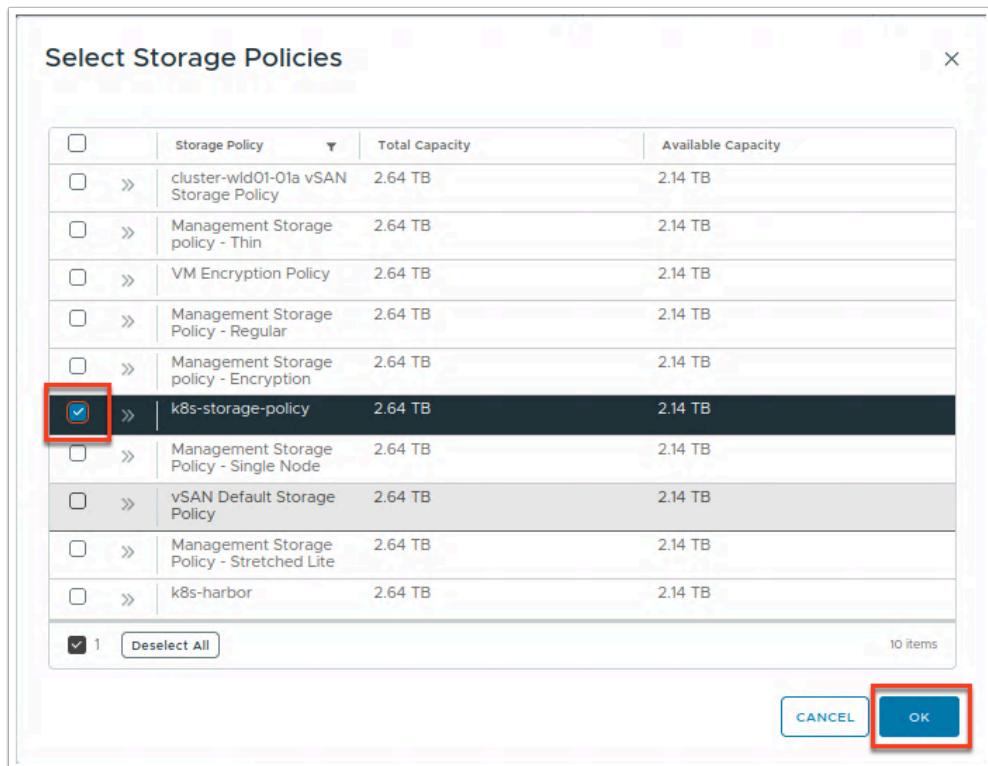
Add a Storage Policy

The screenshot shows the vSphere Namespace-1 Summary page. The top navigation bar includes tabs for Summary, Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The Summary tab is selected. The main content area is divided into several cards:

- VM Service**: Shows 0 Associated VM Classes and a link to ADD VM CLASS.
- Pods**: Shows 0 pods, with status indicators for Running, Pending, and Failed.
- Status**: Displays the namespace was Created on 9/22/25, is Config Status is Running (green checkmark), Kubernetes Status is Active (green checkmark), and is located at supervisor (vc-wld01-a.site-a.vcf.lab). It also provides links to Copy link and Open.
- Storage**: States that no storage policies have been added. It includes a database icon and a message: "You haven't added any storage policies for this namespace. Add some policies to let your devops team access persistent storage." A blue "ADD STORAGE" button is located at the bottom right of this card, which is highlighted with a red box.

1. In the Supervisor Management main Namespaces page, select **namespace-1** in the list
2. Click the **Summary** tab
3. Click the **ADD STORAGE** button

Select a Storage Policy



For the sake of simplicity, we will only select one storage policy, but you can select as many as you want to make them available for your workload volumes.

1. Select the **k8s-storage-policy** policy
2. Click **OK**

Review The Newly Created Storage Policy

Summary Monitor Configure Permissions Zones Compute Storage Network Resources

VM Service (i)

0 Associated VM Classes [ADD VM CLASS](#)

0 Associated Content Libraries [ADD CONTENT LIBRARY](#)

[GO TO VM SERVICE](#)

Pods

0

• Running • Pending • Failed ⚙

Status

Created 9/22/25

Config Status (i) Running

Kubernetes Status (i) Active

Location (i) [supervisor](#) (i) [vc-wld01-a.site-a.vcf.lab](#)

Link to CLI Tools [Copy link](#) [Open](#)

Storage

0 Persistent Volume Claims

Storage Policies (i) k8s-storage-policy | No limit

[EDIT STORAGE](#)

1. Now, you will just need to check if the selected Storage Policy, **k8s-storage-policy** in our case, is listed in theStorage block on the **namespace-1** summary page Note it may take a moment for the added storage to display in the Storage window.

Add VM Classes to The Namespace

The screenshot shows the vSphere Web Client interface. On the left, a sidebar lists namespaces: Namespaces, argocd, and namespace-1 (which is selected). The main area has tabs: Summary (highlighted with a red box), Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The Summary tab displays sections for VM Service, Pods, Kubernetes Service, and Zones. In the VM Service section, there is a button labeled 'ADD VM CLASS' which is also highlighted with a red box.

To add VM Classes that will specify the available sizes for our Namespace VMs, we will need to follow the next steps:

1. In the **Supervisor Management** main **Namespaces** page, select **namespace-1** in the list
2. Click the **Summary** tab
3. Click the **ADD VM CLASS** button. You may need to scroll the page down a bit to find it

Select The VM Classes For This Namespace

Add VM Class | namespace-1 X

Add a VM Class for your developers to self-service VMs on this Namespace. VM Classes shown here were created using VM Service.

MANAGE VM CLASSES

	VM Class Name	CPU	CPU Reservation	Memory	Memory Reservation	PCI Devices	Namespaces	VMs
<input checked="" type="checkbox"/>	best-effort-2xlarge	8 vCPUs	--	64 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-4xlarge	16 vCPUs	--	128 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-8xlarge	32 vCPUs	--	128 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-large	4 vCPUs	--	16 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-medium	2 vCPUs	--	8 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-small	2 vCPUs	--	4 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-xlarge	4 vCPUs	--	32 GB	--	No	1	0
<input checked="" type="checkbox"/>	best-effort-xsmall	2 vCPUs	--	2 GB	--	No	1	2
<input checked="" type="checkbox"/>	guaranteed-2xlarge	8 vCPUs	100%	64 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-4xlarge	16 vCPUs	100%	128 GB	100%	No	1	0

10 [Manage Columns](#) [Deselect All](#) Items per page: 10 | < | 1 / 2 | > | [CANCEL](#) [OK](#)

You can select as many VM Classes as you want. In our case, we will select all the available VM Classes:

1. Click on the check box on the top left corner of the **VM Classes** table
2. Click the right arrow icon in the bottom right corner of the table to go to the second page

Select The VM Classes For This Namespace (continued)

Add VM Class | namespace-1 X

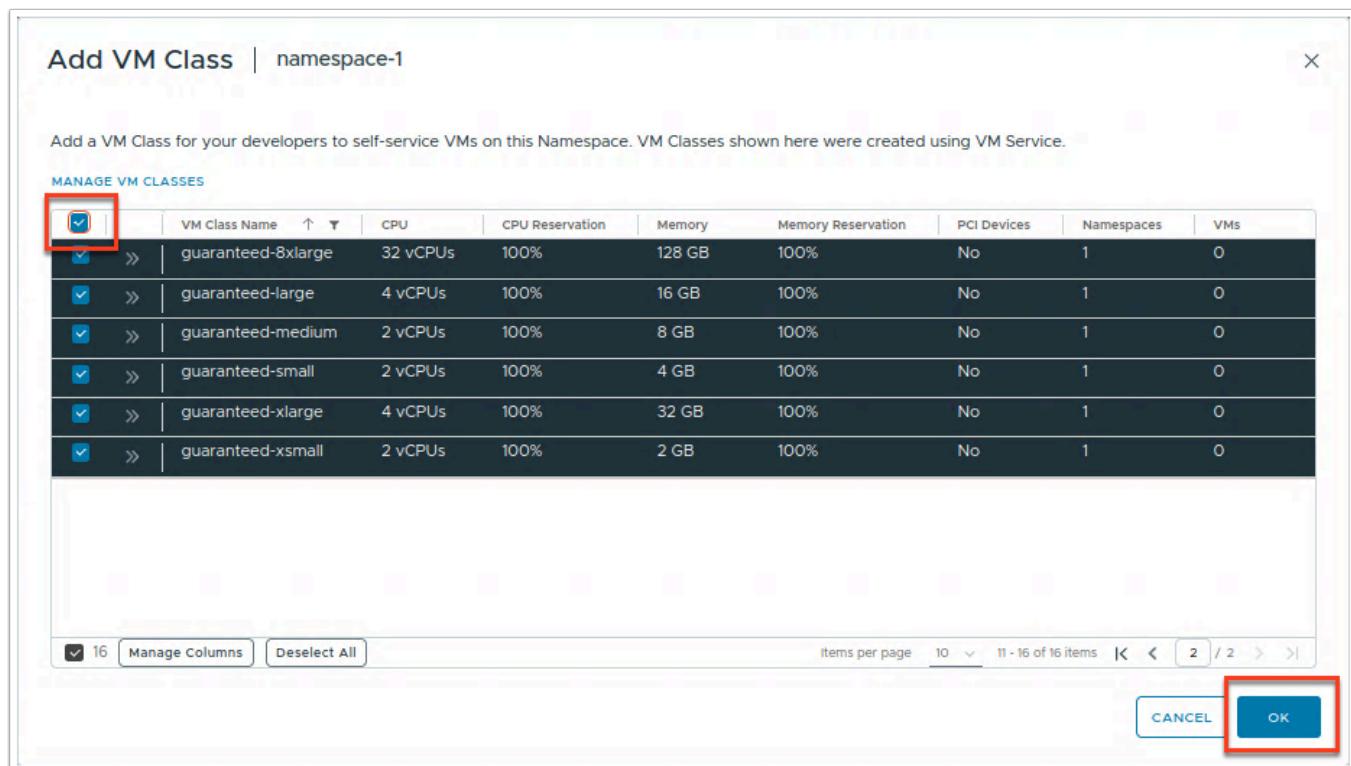
Add a VM Class for your developers to self-service VMs on this Namespace. VM Classes shown here were created using VM Service.

MANAGE VM CLASSES

<input checked="" type="checkbox"/>	VM Class Name	CPU	CPU Reservation	Memory	Memory Reservation	PCI Devices	Namespaces	VMs
<input checked="" type="checkbox"/>	guaranteed-8xlarge	32 vCPUs	100%	128 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-large	4 vCPUs	100%	16 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-medium	2 vCPUs	100%	8 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-small	2 vCPUs	100%	4 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-xlarge	4 vCPUs	100%	32 GB	100%	No	1	0
<input checked="" type="checkbox"/>	guaranteed-xsmall	2 vCPUs	100%	2 GB	100%	No	1	0

16 [Manage Columns](#) [Deselect All](#) Items per page: 10 11 - 16 of 16 items K < 2 / 2 > |

[CANCEL](#) [OK](#)



1. Click on the check box on the top left corner of the second page of **VM Classes** table again, to finish selecting the rest of available VM Classes
2. Click **OK**

View The VM Classes Added to The Namespace

(#) namespace-1 | : ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network Resources

VM Service ⓘ
16 Associated VM Classes
[MANAGE VM CLASSES](#)

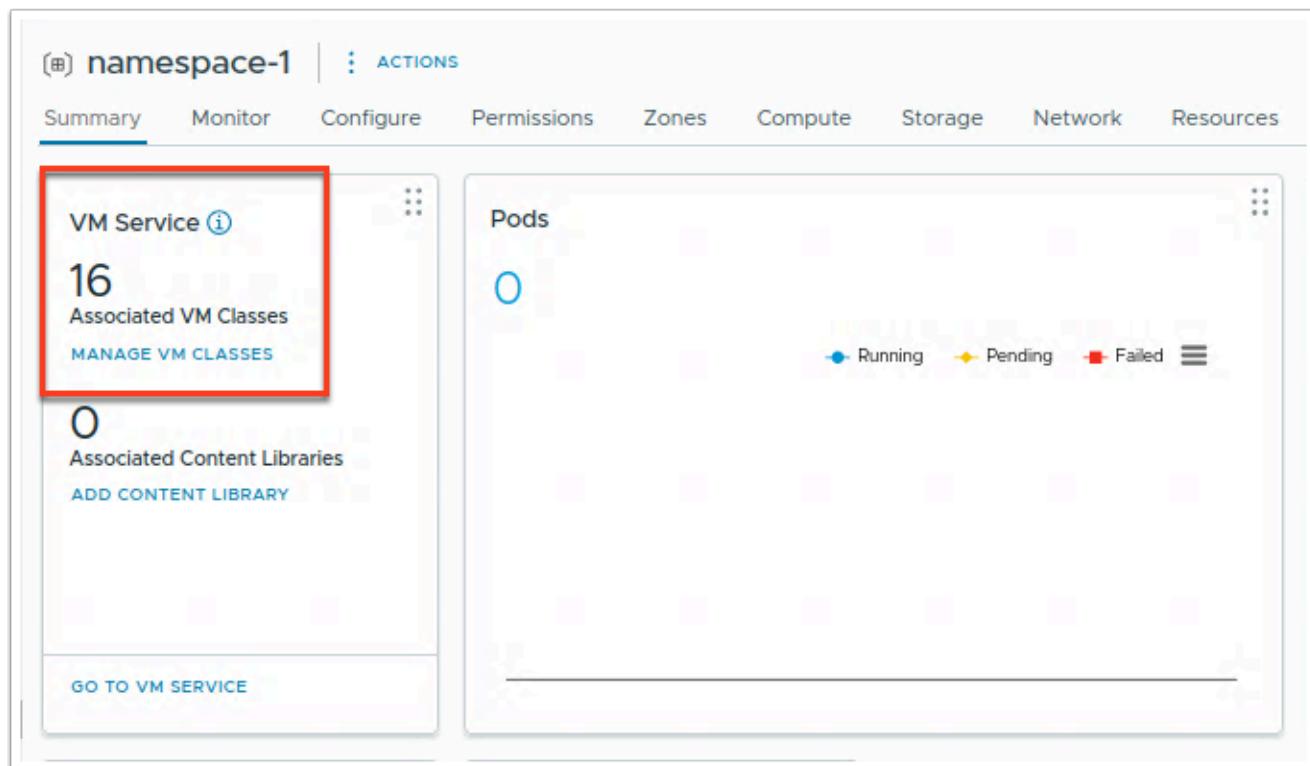
0 Associated Content Libraries
[ADD CONTENT LIBRARY](#)

[GO TO VM SERVICE](#)

Pods

0

Legend: ● Running ▲ Pending ■ Failed ⚡

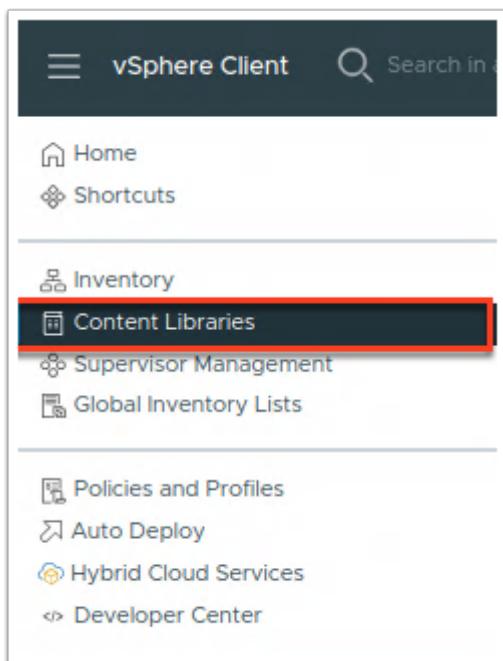


1. After finishing selecting the VM Classes, check in the **namespace-1 Summary** page the number of associated VM Classes. Again it may take a moment for the newly added VM Classes to appear in the VM Service tile.

Configure a Content Library

Next, we will create a new Content Library for deploying VMs through the VM Service. We will then associate that new Content Library with the Namespace created above.

Open Content Libraries



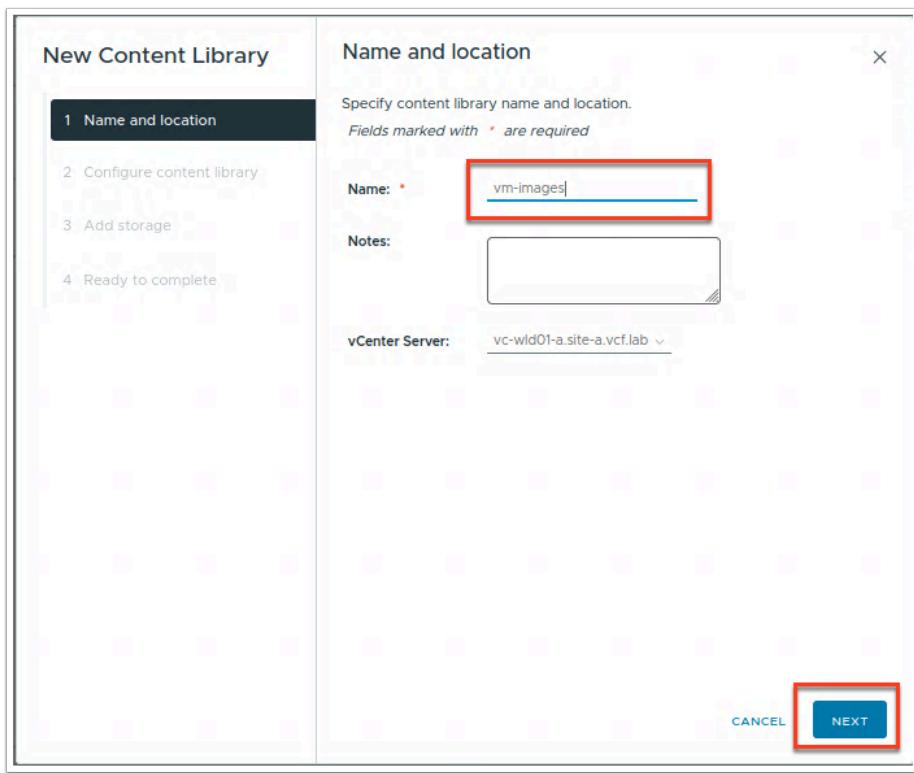
1. Select **Content Libraries** from the Menu drop down

Create a New Content Library

Content Libraries														
ADVANCED		CREATE												
		Name	Type	Publishing Enabled	Security Policy	Password Protected	Automatic Synchronization	vCenter Server	Templates	Other Library Items	Storage Used	Creation Date	Last Modified Date	Last Sync Date
<input type="checkbox"/>	Kubernetes Service Content Library	Subscribe d	No	Not applied	No	No	<input type="checkbox"/> vc-wl d01-a.site-a.vcf.lab	89	0	47.48 GB	05/20/2025, 5:16:08 PM	08/07/2025, 8:29:14 AM	08/07/2025, 12:00:01 AM	
<input type="checkbox"/>	Supervisor images	Subscribe d	No	Not applied	No	No	<input type="checkbox"/> vc-wl d01-a.site-a.vcf.lab	1	3	13.09 GB	07/29/2025, 3:47:05 AM	08/07/2025, 8:30:23 AM	08/07/2025, 12:00:00 AM	

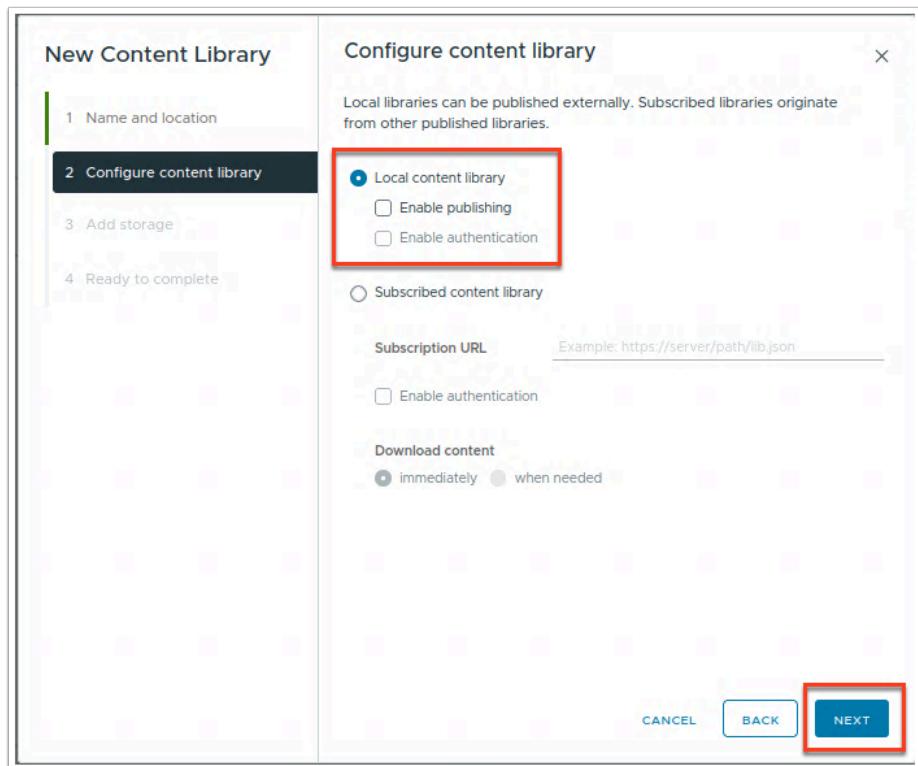
1. Click **CREATE**.

Specify the Name And Location



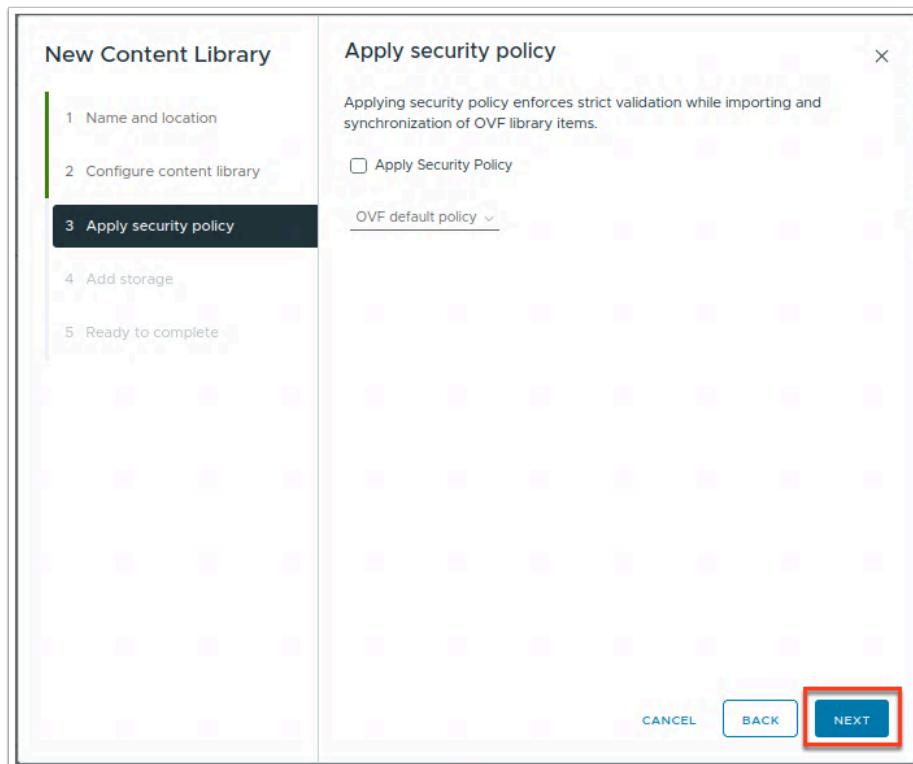
1. Name the content library, use **vm-images** for the name.

Specify as Local Content Library



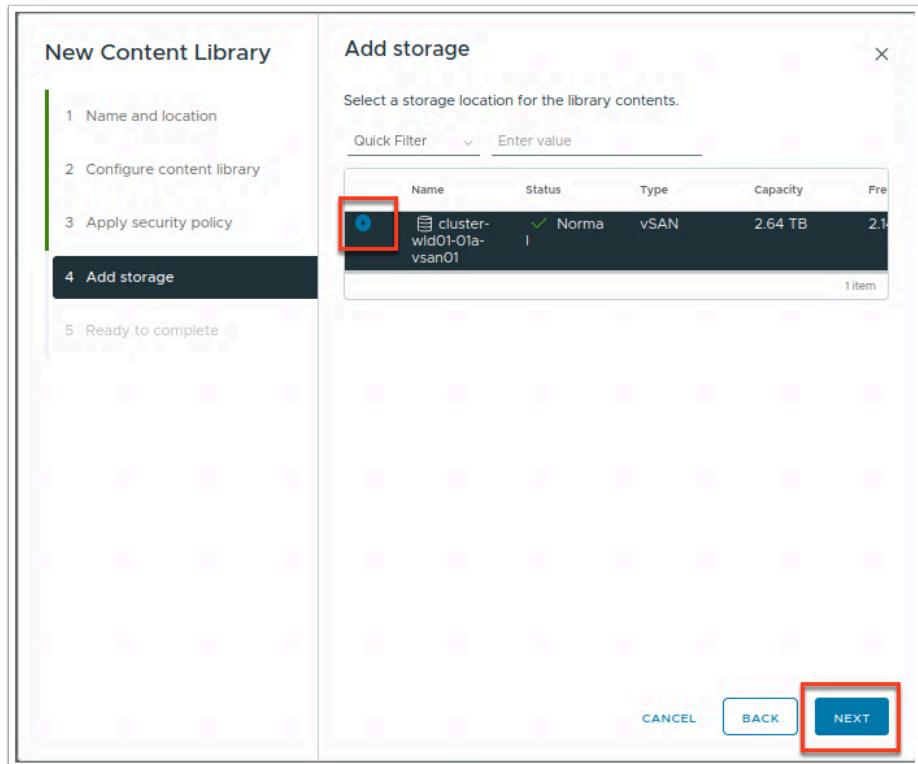
1. If not already selected, select use **Local content library** and hit **NEXT**.

Apply Security Policy



1. Accept the defaults and click NEXT.

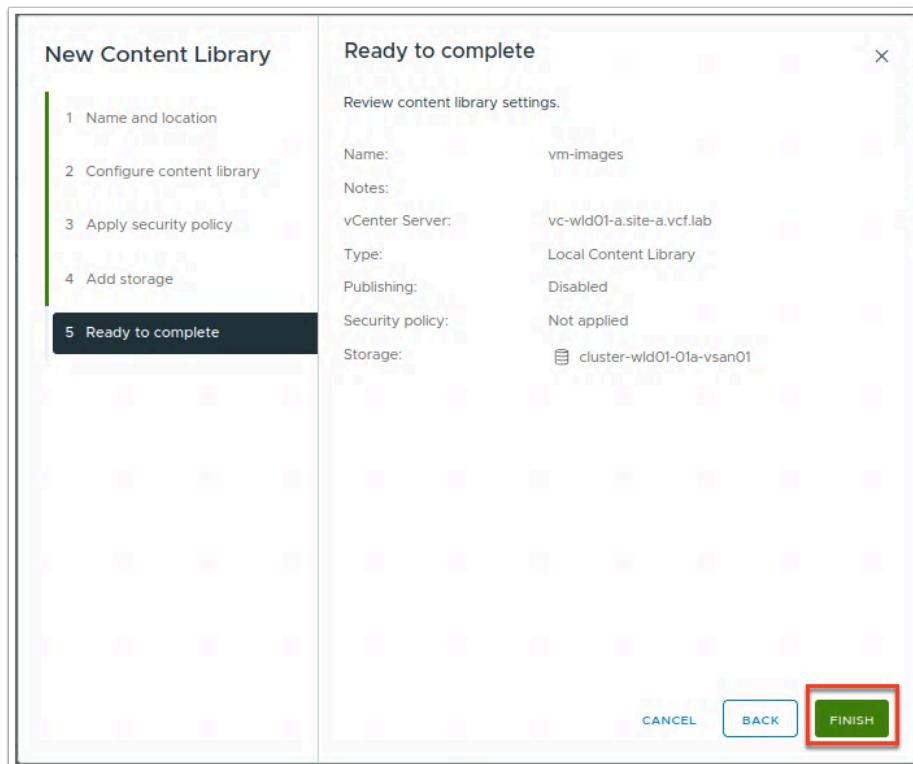
Add Storage For The Content Library



If it is not already selected, click the radio button to elect the **cluster-wkld-01a-vsanc01** storage policy .

1. click **NEXT**.

Complete Adding The Storage Library



1. Click **FINISH** to add the storage library to the supervisor cluster.

Upload new .ova for VM Deployment to Content Library

Content Libraries															
ADVANCED		CREATE													
		Quick Filter		Enter value											
<input type="checkbox"/>	Name ↑	Type	Publishing Enabled	Security Policy	Password Protected	Automatic Synchronization	vCenter Server	Templates	Other Library Items	Storage Used	Creation Date	Last Modified Date	Last Sync Date		
<input type="checkbox"/>	Kubernetes Service Content Library	Subscribe	No	Not applied	No	No	vc-wld01-a.site-a.vcf.lab	89	0	47.48 GB	05/20/2025, 5:16:08 PM	08/07/2025, 8:29:14 AM	08/07/2025, 12:00:01 AM		
<input type="checkbox"/>	Supervisor Images	Subscribe	No	Not applied	No	No	vc-wld01-a.site-a.vcf.lab	1	3	13.09 GB	07/29/2025, 3:47:05 AM	08/07/2025, 8:30:23 AM	08/07/2025, 12:00:00 AM		
<input type="checkbox"/>	vm-images	Local	No	Not applied	--	No	vc-wld01-a.site-a.vcf.lab	0	0	0 B	09/22/2025, 8:08:53 AM	09/22/2025, 8:08:53 AM	09/22/2025, 8:08:53 AM		

1. Click in the newly created content library, **vm-images**.

Upload an Image to The vm-images Content Library

The screenshot shows the 'vm-images' content library details. The 'Summary' tab is selected, indicated by a red box. Other tabs include 'Templates' and 'Other Types'. The 'Content Library Details' section lists basic information: Type (Local), Security policy (Not applied), Storage (Datastore), Size (0 B), Items (0), Streaming optimized (No). It also shows creation and modification dates (09/22/2025, 8:08:53 AM). The 'Storage' section shows a single datastore named 'cluster-wld01-01a-vsan01' with a capacity of 2.64 TB and free space of 2.14 TB. The 'Related Objects' section lists a vCenter Server named 'vc-wld01-a.site-a.vcf.lab'. The 'Tags' and 'Notes' sections both indicate 'No tags assigned' and 'No notes assigned'.

If the Summary tab is not selected, select the summary tab.

Select Actions

The screenshot shows the same 'vm-images' content library details, but the 'Actions' tab is selected, indicated by a red box. The other tabs ('Summary', 'Templates', 'Other Types') are visible but not selected. The 'Content Library Details' section contains the same information as the previous screenshot.

1. Click **Actions**.

Import a OVA File - Step 1

The screenshot shows the 'Content Library Details' page for 'vm-images'. The 'Import item' option in the 'Actions' dropdown menu is highlighted with a red box.

Content Library Details

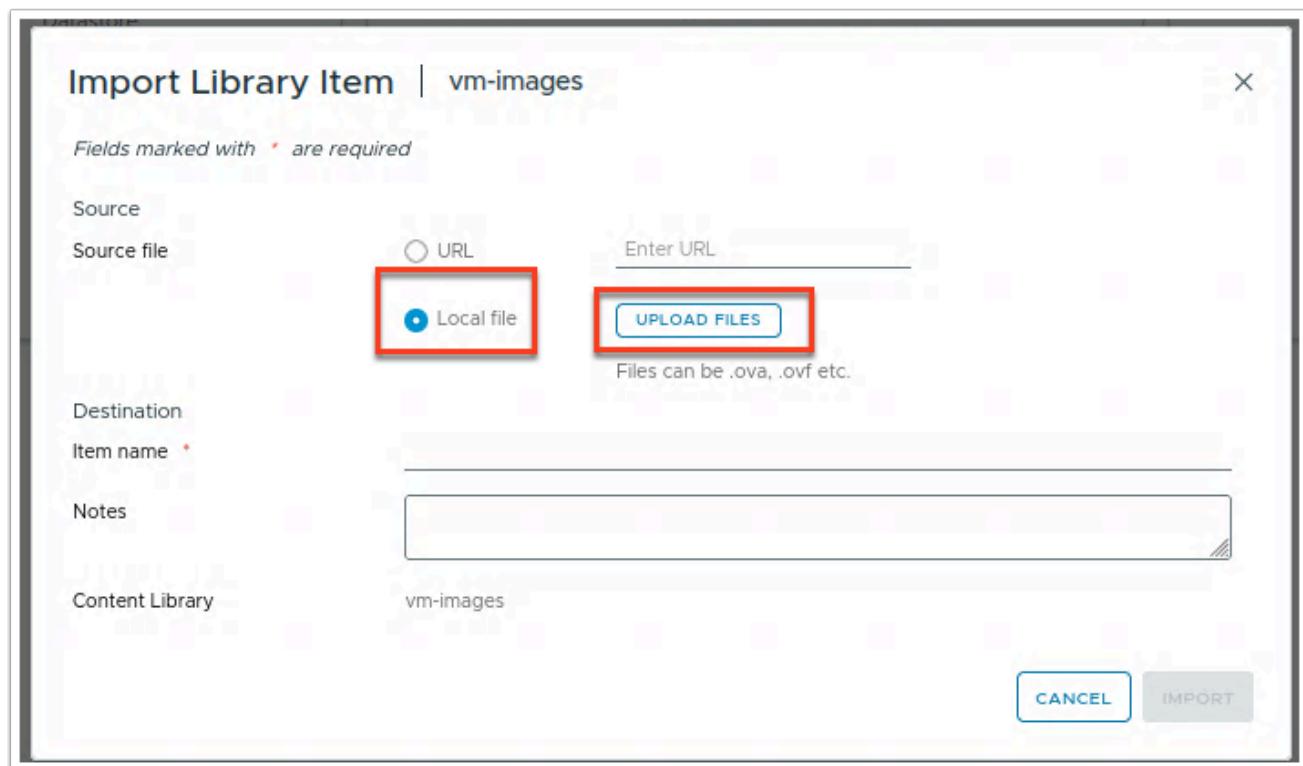
Actions

- Import item
- Migrate
- Edit Settings...
- Edit Notes...
- Rename...
- Tags >
- Delete Content Library

Created: 09/22/2025, 8:08:53 AM
Last modified: 09/22/2025, 8:08:53 AM
Last sync:

1. Select **Import Item**.

Select Local File



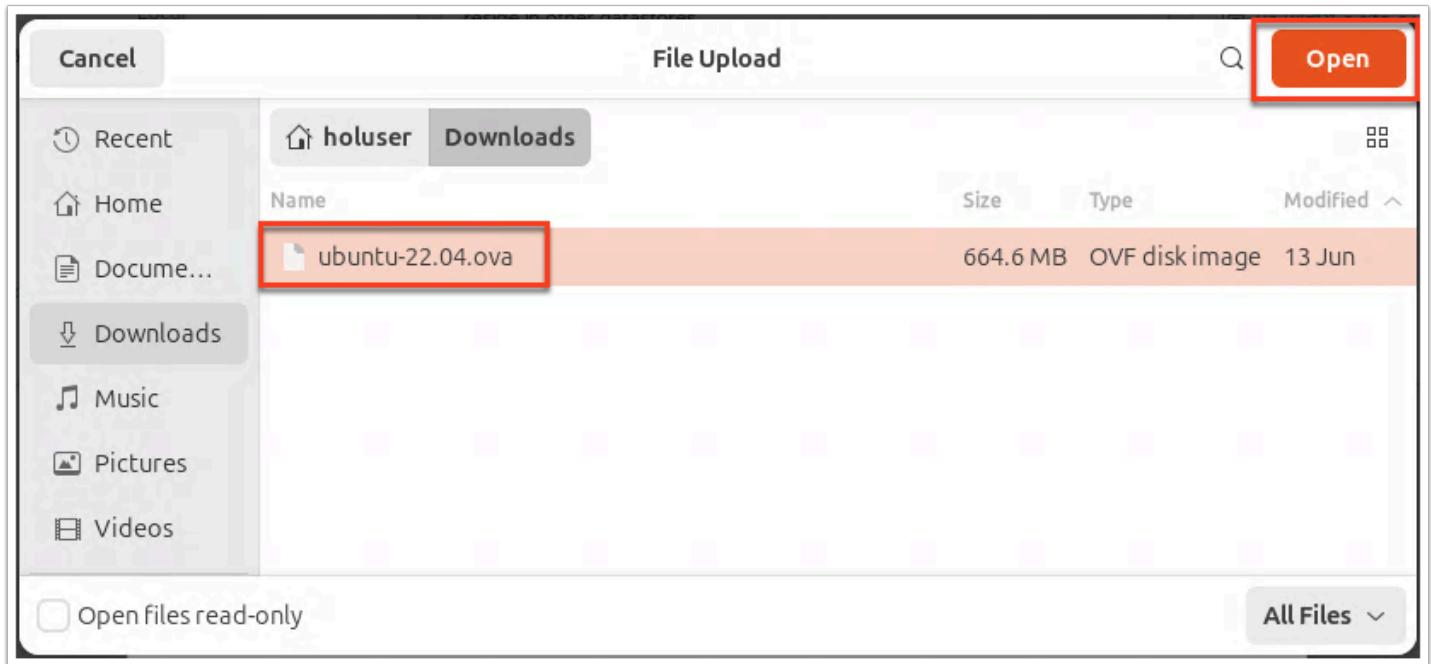
1. Click the radio button for Local File.
2. Click **UPLOAD FILES**.

Select File Image to Upload



1. Select the **Downloads** folder.

Select Ubuntu Image



1. Select the ***ubuntu.22.04.ova*** image.
1. Click **OPEN**.

Import The Ubuntu Image to The Content Library

Import Library Item | vm-images X

Fields marked with * are required

If the certificate or manifest file are not available at source during the import process, validations for each will be skipped.

Source

Source file URL
 Local file

Source file details
1 file ready to import
Files can be .ova, .ovf etc.

Destination
Item name *

Notes

Content Library

1. Click **IMPORT**.

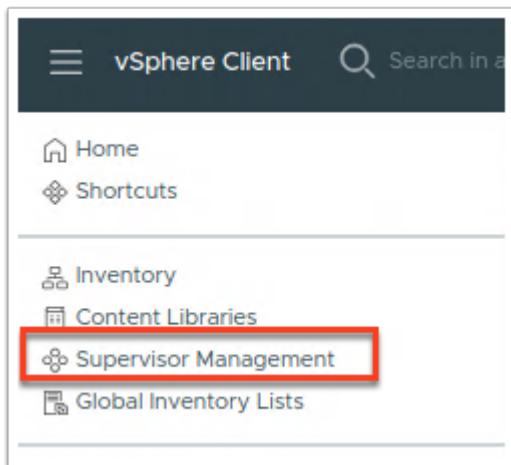
Monitor the Import

Recent Tasks			Alarms
Task Name	Target	Status	
Upload Files to a Library Item	ubuntu-22.04	✓ Completed	

Looking at the **Recent Tasks** section, after some time you will see that the image has been imported successfully.

i You can move forward in the lab as the ova uploads

Navigate to Supervisor Management



1. From the **vSphere Client** menu, select **Supervisor Management**.

Associate The Newly Created Content Library With A Namespace

A screenshot of the "Namespaces" tab within the Supervisor Management section. The tab has a list of namespaces: "argocd", "namespace-1" (which is highlighted with a red box), "svc-argocd-service-domain-c10", "svc-cci-ns-domain-c10", "svc-harbor-domain-c10", "svc-tkg-domain-c10", and "svc-velero-domain-c10".

Ensure that the Namespaces tab is selected.

1. Click on **namespace-1**.

Manage the Content Library

VM Service
16 Associated VM Classes
MANAGE VM CLASSES
0 Associated Content Libraries
ADD CONTENT LIBRARY
GO TO VM SERVICE

Pods
0 Running Pending Failed

Status
Created 9/22/25
Config Status Running
Kubernetes Status Active
Location supervisor
@ yc-wld01-a.site-a.vcf.lab
Link to CLI Tools Copy link Open

Storage
0 Persistent Volume Claims
Storage Policies k8s-storage-policy No limit
EDIT STORAGE

Capacity and Usage
CPU 0 MHz used No limit
Memory 0 MB used No limit
Storage 0 MB used No limit
EDIT LIMITS

Kubernetes Service
0 Kubernetes clusters
1 Associated Content Libraries
MANAGE
VIEW ALL

Zones
1 Zones
ADD ZONE
VIEW ALL

1. Click **ADD CONTENT LIBRARY** on the **VM Service** tile.

Select the Content Library

Add Content Library

Add a content library containing VM templates for your developers to self-service VMs from this Namespace. Kubernetes Service content libraries must be managed from the Kubernetes Service card.

CREATE NEW CONTENT LIBRARY

<input type="checkbox"/>	Name	Type	Templates	Storage Used	Last Modified Date
<input type="checkbox"/>	Kubernetes Service Content Library	Subscribed	89	47.48 GB	Aug 7, 2025, 3:29 PM
<input type="checkbox"/>	Supervisor images	Subscribed	1	13.09 GB	Aug 7, 2025, 3:30 PM
<input checked="" type="checkbox"/>	vm-images	Local	1	3.09 GB	Sep 22, 2025, 3:08 PM

1 Manage Columns Deselect All 3 items

CANCEL OK

1. Click the Radio Checkbox for **vm-images**.
2. Click **OK**.

It may take a few seconds but you will see the **vm-images** content library added to the namespace.

Return to Namespaces

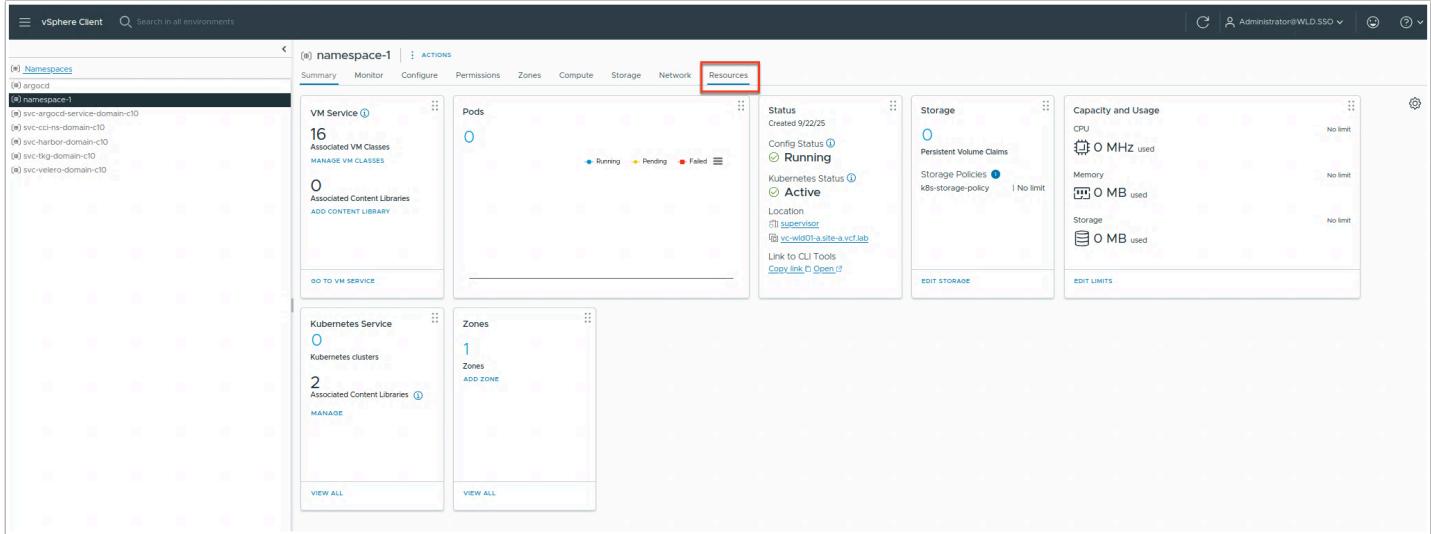
1. Click on the **Namespaces** Tab.

Select namespace-1

	Namespaces	Supervisor	Config Status	CPU (Used Limit)	Memory (Used Limit)	Storage (Used Limit)	Description	vCenter
(#)	argocd	supervisor	Running	1.837 GHz No Limit	2.35 GB No Limit	47 GB No Limit		vc-wld01-a-site-a.vcf.lab
(#)	namespace-1	supervisor	Running	0 No Limit	0 No Limit	0 No Limit		vc-wld01-a-site-a.vcf.lab
(#)	svc-argocd-service-domain-c10	supervisor	Running	21 MHz No Limit	374 MB No Limit	0 No Limit		vc-wld01-a-site-a.vcf.lab
(#)	svc-ci-ns-domain-c10	supervisor	Running	210 MHz No Limit	567 MB No Limit	0 No Limit		vc-wld01-a-site-a.vcf.lab
(#)	svc-harbor-domain-c10	supervisor	Running	420 MHz No Limit	1.22 GB No Limit	18 GB No Limit		vc-wld01-a-site-a.vcf.lab
(#)	svc-tkg-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit		vc-wld01-a-site-a.vcf.lab
(#)	svc-velero-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit		vc-wld01-a-site-a.vcf.lab

1. Click on **namespace-1**

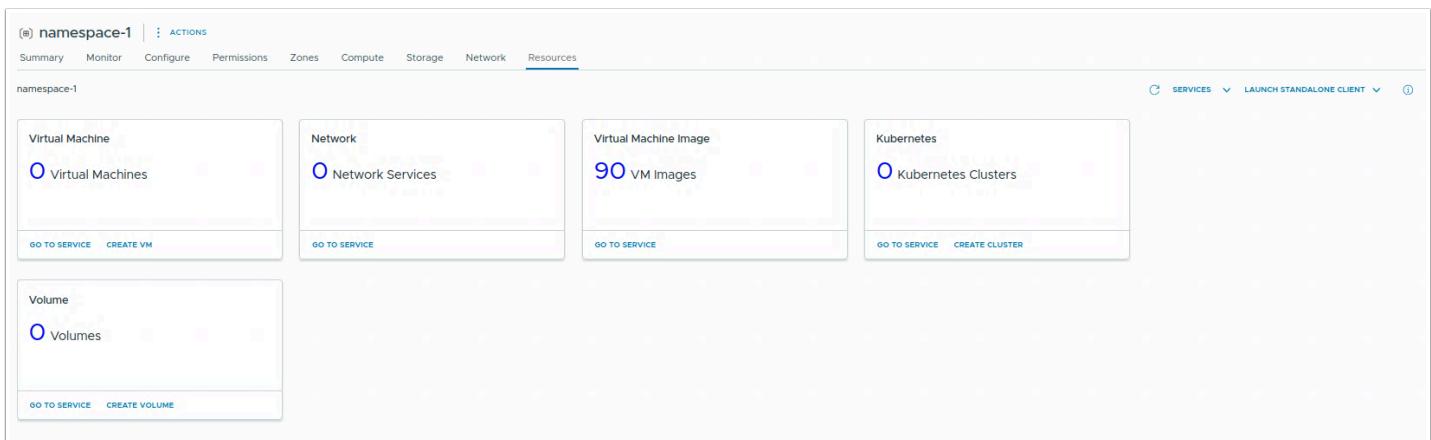
Create a VM



Next, let's deploy a VM via the VM Service through the Consumption Interface.

1. Click the **Resources** tab.

Select The Create Link



1. In the **Virtual Machine** tile, click **CREATE VM**

Select the OVF

The screenshot shows the vSphere Web Client interface. On the left, the 'New Virtual Machine' wizard is open with five steps: 1. Deploy VM from..., 2. Deploy a New VM, 3. Advanced Settings (Optional), 4. Network Configuration (Optional), and 5. Review and Confirm. Step 1 has 'Deploy from OVF' checked. Step 2 has a 'NEXT' button highlighted with a red box. To the right, a 'Kubernetes Resource YAML' panel displays the following code:

```
apiVersion: vmoperator.vmware.com/v1alpha3
kind: VirtualMachine
metadata:
  name: vm-515s
  namespace: namespace-1
  labels:
    vm-selector: vm-515s
spec:
  classLabel: ''
  imageName: ''
  storageClass: k8s-storage-policy
  powerState: PoweredOn
```

1. Leave **Deploy from OVF** checked.
2. Click **NEXT**.

i Keep an eye on the Kubernetes Resource YAML part of the screen. Take note of the **Copy to Clipboard** and **Download** buttons

Configure the VM

The screenshot shows the vSphere Web Client interface for creating a new virtual machine. The top navigation bar includes 'namespace-1', 'Actions', 'Summary', 'Monitor', 'Configure', 'Permissions', 'Zones', 'Compute', 'Storage', 'Network', and 'Resources'. The 'Resources' tab is selected. Below the navigation is a breadcrumb trail: 'namespace-1 > Virtual Machine service > create-vm'. A 'New Virtual Machine' button is present. The main area has two steps: '1. Deploy VM from...' (OVF) and '2. Deploy a New VM' (Select the VM Class and VM Image). Step 2 is active, showing a table of VM Images. A red box highlights the 'VM Name' field ('cli-vm'), the 'Zone' dropdown ('z-wld-a'), and the 'VM Image' table row ('ubuntu-22.04'). Another red box highlights the 'cli-vm' entry in the 'Virtual Machine' section of the 'Kubernetes Resource YAML' panel on the right.

```

apiVersion: vmoperator.vmware.com/v1alpha3
kind: VirtualMachine
metadata:
  name: cli-vm
  namespace: namespace-1
  labels:
    vm-selector: cli-vm
    topology.kubernetes.io/zone: z-wld-a
spec:
  className: ''
  # Friendly image name: ubuntu-22.04
  imageName: vmi-323ad24528727ec58
  storageClass: k8s-storage-policy
  powerState: PoweredOn

```

1. For **VM NAME:** *cli-vm*.
2. For **Zone:** *z-wld-a*.
3. For **VM Image Name:** *ubuntu-22.04*.

i This center window scrolls. You cannot see it when you are not scrolling up or down, but it is there.

Scroll down

Select the VM Class

VM Class

A VM Class is the virtual hardware specification of the VM including vCPU, memory and any additional devices that are used to create the VM. DevOps users can select from the VM Classes available to this namespace. Different Namespaces may have different VM Classes available to them. Additional VM Classes can be customized with custom hardware specification to target workloads such as web servers, database servers, AI/ML workers, etc., and may only be added to a Namespace by an Administrator.

Name	CPU	CPU Reservation	Memory	Memory Reservation
<input type="radio"/> best-effort-xsmall	2 vCPUs	No Reservation	2 GiB	No Reservation
<input type="radio"/> guaranteed-xsmall	2 vCPUs	100%	2 GiB	100%
<input checked="" type="radio"/> best-effort-small	2 vCPUs	No Reservation	4 GiB	No Reservation
<input type="radio"/> guaranteed-small	2 vCPUs	100%	4 GiB	100%
<input type="radio"/> best-effort-medium	2 vCPUs	No Reservation	8 GiB	No Reservation

Storage Class: k8s-storage-policy

Power State: Powered On

REVIEW AND CONFIRM **NEXT**

Kubernetes Resource YAML

```

apiVersion: vmoperator.vmware.com/v1alpha3
kind: VirtualMachine
metadata:
  name: cli-vm
  namespace: namespace-1
  labels:
    vm-selector: cli-vm
    topology.kubernetes.io/zone: z-wld-a
spec:
  className: best-effort-small
  # Friendly image name: ubuntu-22.04
  imageName: vmi-323ad24520727ec58
  storageClass: k8s-storage-policy
  powerState: PoweredOn

```

Now we will select the VM class, the VM class specifies the configuration for the VM that will be created. In this case we are requesting a best-effort-small, this will tell the service to TRY and create the VM with this specification. If a VM class is not available it will select the next available VM class. You may have to scroll down to see the VM classes.

1. Click the radio button for **best-effort-small**.
2. Click **NEXT**

Select the Persistent Volume

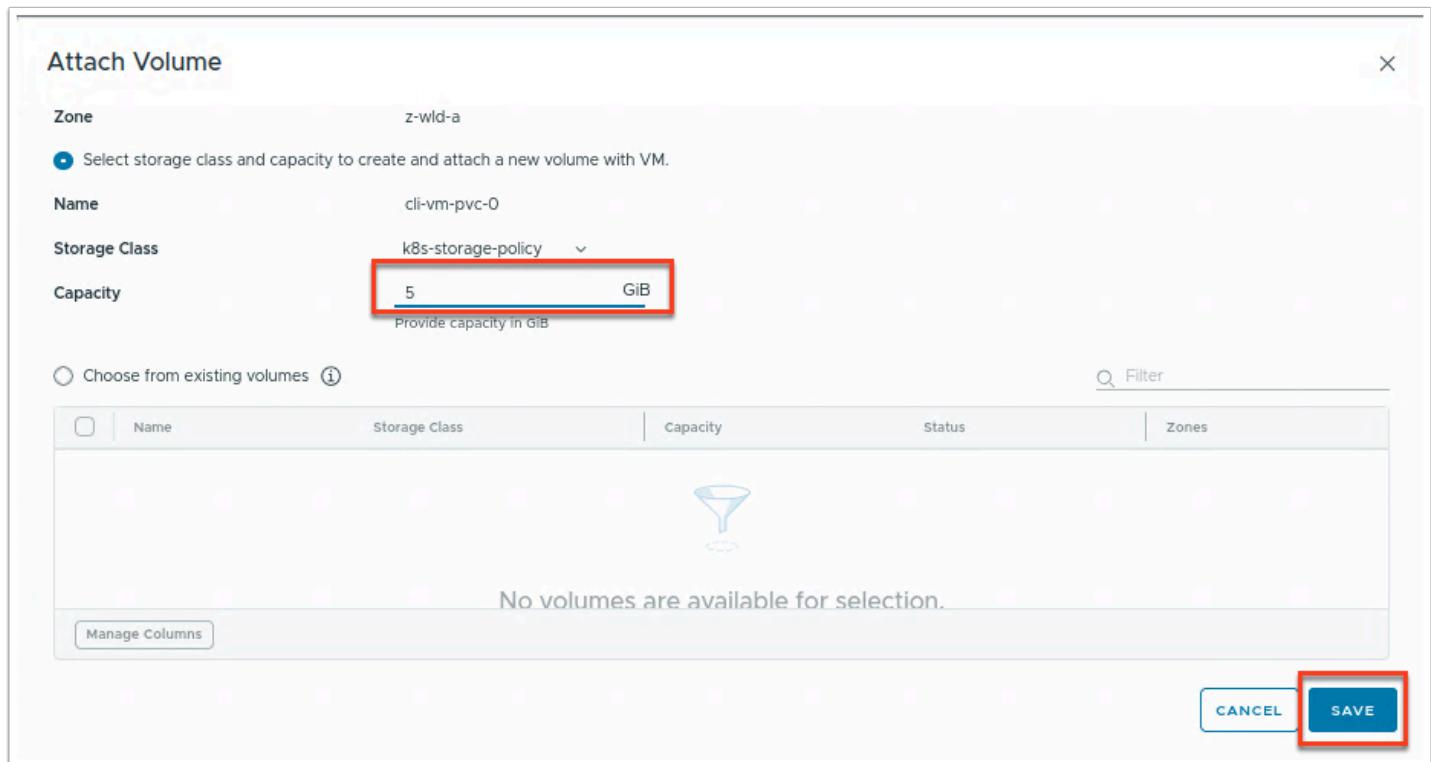
The screenshot shows the 'create-vm' wizard in the vSphere Web Client. The 'Persistent Volume' section is highlighted with a red box around the '+ ATTACH VOLUME' button. To the right, the 'Kubernetes Resource YAML' pane displays the generated YAML configuration for the VM.

```
apiVersion: vmoperator.vmware.com/v1alpha1
kind: VirtualMachine
metadata:
  name: cli-vm
  namespace: namespace-1
  labels:
    vm-selector: cli-vm
    topology.kubernetes.io/zone: z-wld-a
spec:
  className: best-effort-small
  # Friendly image name: ubuntu-22.04
  imageName: vml-323ad24528727ec58
  storageClass: k8s-storage-policy
  powerState: PoweredOn
```

Here we will assign a Persistent Volume to the VM we are creating. A Persistent Volume is storage assigned to the VM which will remain even if the VM is deleted.

1. Click **ATTACH VOLUME**.

Attach a Storage Volume



1. Enter **5GB** for **Capacity**.
2. Click **SAVE**.

Attach a Load Balancer

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: cli-vm-pvc-0
  namespace: namespace-1
  labels:
    vm-selector: cli-vm
  annotations:
    csi.vsphere.volume-requested-topology: '{["tier": "t1"]}'
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: k8s-storage-policy

```

Here we are going to attach a Load Balancer to our VM. A load balancer is a device or software that distributes network traffic across multiple servers, ensuring no single server is overwhelmed and maximizing resource utilization. It enhances application performance, reliability, and availability by intelligently routing client requests to healthy servers.

1. Scroll down to the Load Balancer section and click **ADD**.
2. Select **NEW**

Create a VM Load Balancer - Step 1

The screenshot shows the 'Create VM Load Balancer' dialog box. In the 'Selector' section, the identifier 'cli-vm-lb-fln0' and its corresponding selector 'cli-vm-lb-fln0: vm-lb-selector' are displayed. Below this, a new port is being configured with the name 'ssh', port '22', protocol 'TCP', and target '22'. The 'ADD' button is highlighted with a red box. The bottom part of the dialog shows a table with one row, featuring a funnel icon and the target value '22'. At the bottom right are 'CANCEL' and 'SAVE' buttons.

Use the Following Settings to Create a new Loadbalancer for this VM

- **Name:** Add **cli-** to the name and leave the unique identifier in tact
- **Create new Port:**
 - **Name:** **ssh**
 - **Port:** **22**
 - **Protocol:** select **TCP**
 - **Target:** **22**

1. Click **ADD**.

Create VM Load Balancer

Edit the specifications for the Virtual Machine Load Balancer.

Name	cli-vm-lb-fln0								
Selector	cli-vm-lb-fln0: vm-lb-selector								
Create new port									
Name									
Port									
Protocol	TCP								
Target									
<input type="button" value="ADD"/> <table border="1"> <thead> <tr> <th>Name</th> <th>Port</th> <th>Protocol</th> <th>Target</th> </tr> </thead> <tbody> <tr> <td>ssh</td> <td>22</td> <td>TCP</td> <td>22</td> </tr> </tbody> </table>		Name	Port	Protocol	Target	ssh	22	TCP	22
Name	Port	Protocol	Target						
ssh	22	TCP	22						
Ports per page <input type="button" value="10"/>									
<input type="button" value="CANCEL"/> <input type="button" value="SAVE"/>									

1. Select **SAVE**.

Guest Customization

namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network Resources

namespace-1 > Virtual Machine service > create-vm

Load Balancers per page 1 Load Balancer

Guest Customization
You can specify any data, configuration or scripts in the bootstrap format that will be used to initialize the VM. You can use this section to mount disks, configure additional users, run any custom scripts at boot time and much more.
 Cloud-Init Sysprep Linuxprep vAppConfig
Learn more about [Cloud-Init](#)

SSH Public Keys

GUIDED INPUTS **RAW CONFIGURATION**

Timezone

Enable Default User

SSH Password Authentication

Run Commands

NEXT

4. Network Configuration (Optional) Configure the network for this virtual machine

Kubernetes Resource YAML

```

1  apiVersion: vmoperator.vmware.com/v1alpha3
2  kind: VirtualMachineService
3  metadata:
4    name: cli-vm-lb-fln0
5    namespace: namespace-1
6  spec:
7    selector:
8      cli-vm-lb-fln0: vm-lb-selector
9    type: LoadBalancer
10   ports:
11     - name: ssh
12       protocol: TCP
13       port: 22
14       targetPort: 22
15

```

1. Scroll down and select **NEXT**.

Network Configuration

The screenshot shows the vSphere Network Configuration interface for creating a VM in namespace-1. The 'Network' tab is selected. Step 4, 'Network Configuration (Optional)', is shown with the sub-step 'Configure the network for this virtual machine'. A table lists one network interface: eth0, connected to the default network. Below this, 'Global Network Settings' include Host Name (my-vm) and Domain Name (domain.local). Under 'DNS Settings', there is a nameservers entry (192.0.2.1) and a search domains entry (my.domain.local). A 'NEXT' button is at the bottom left, highlighted with a red box. To the right, a 'Kubernetes Resource YAML' panel displays the generated YAML code:

```
1 apiVersion: vmoperator.vmware.com/v1alpha3
2 kind: VirtualMachineService
3 metadata:
4   name: cli-vm-lb-f1n0
5   namespace: namespace-1
6 spec:
7   selector:
8     cli-vm-lb-f1n0: vm-lb-selector
9   type: LoadBalancer
10  ports:
11    - name: ssh
12      protocol: TCP
13      port: 22
14      targetPort: 22
15
```

Accept the default values.

1. Scroll down and select **NEXT**.

Review the YAML

» Kubernetes Resource YAML

- ✓ Virtual Machine
 - cli-vm
- ✓ Load Balancer
 - cli-vm-lb-fIn0
- ✓ Persistent Volume Claim
 - cli-vm-pvc-0

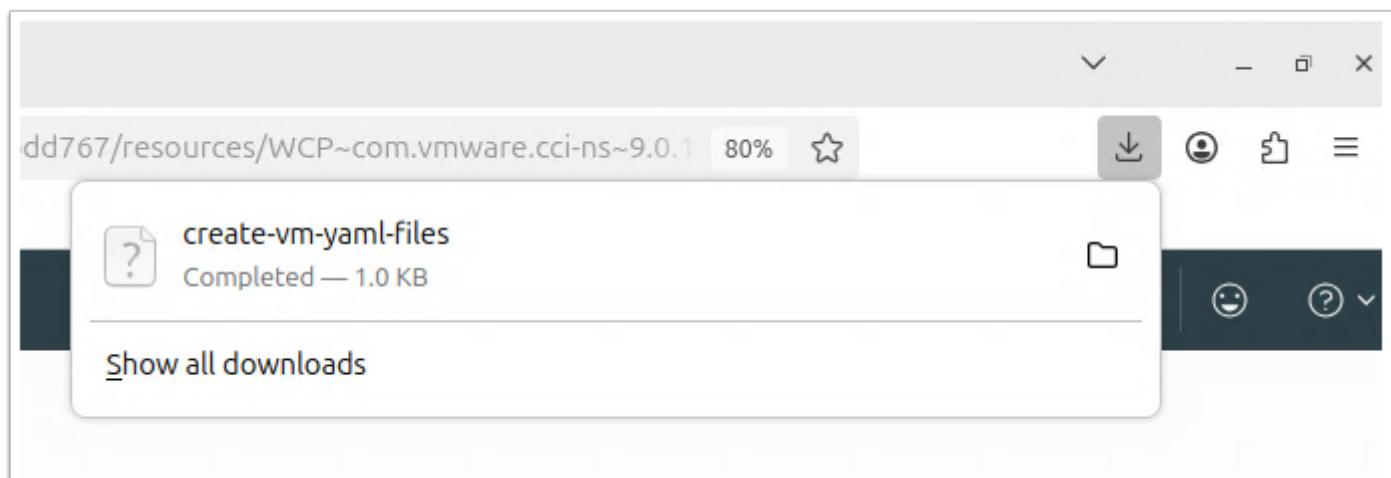
```
1 apiVersion: vmoperator.vmware.com/v1alpha3
2 kind: VirtualMachine
3 metadata:
4   name: cli-vm
5   namespace: namespace-1
6   labels:
7     vm-selector: cli-vm
8     cli-vm-lb-fIn0: vm-lb-selector
9     topology.kubernetes.io/zone: z-wld-a
10 spec:
11   className: best-effort-small
12   # Friendly image name: ubuntu-22.04
13   imageName: vmi-323ad24528727ec58
14   storageClass: k8s-storage-policy
15   powerState: PoweredOn
16   volumes:
17     - name: cli-vm-pvc-0
18       persistentVolumeClaim:
19         claimName: cli-vm-pvc-0
20
```

Download all YAMLS to a zip

Cancel 

Here you can review the YAML file that will be used to deploy the VM. You can copy or download the YAML file to use as a template for additional deployments.

Click the download icon in the RED box to download the YAML file.



Create a VM - Review and Complete.

The screenshot shows the 'New Virtual Machine' wizard in the VMware Cloud Foundation UI. The configuration steps are as follows:

- 1. Deploy VM from..: OVF
- 2. Deploy a New VM: VM Name (cli-vm), VM Image (ubuntu-22.04), VM Class (best-effort-small), Storage Class (k8s-storage-policy), Power State (Powered On)
- 3. Advanced Settings (Optional): Persistent Volume (Name: cli-vm-pvc-0, Size: 5 GiB (New)), Load Balancer (Configured), Cloud-Init (Validated)
- 4. Network Configuration (Optional): None
- 5. Review and Confirm: Summary of configuration

A note at the bottom states: "The Virtual Machine will have one network interface that will be connected to the default workload network for the Supervisor namespace." A red box highlights the "DEPLOY VM" button.

Kubernetes Resource YAML

```

apiVersion: vmoperator.vmware.com/v1alpha1
kind: VirtualMachine
metadata:
  name: cli-vm
  namespace: namespace-1
  labels:
    vm-selector: cli-vm
    cli-vm-lb-f1n0: vm-lb-selector
    topology.kubernetes.io/zone: z-wld-a
spec:
  className: best-effort-small
  # Friendly image name: ubuntu-22.04
  imageName: vml-323ad24528727ec58
  storageClass: k8s-storage-policy
  powerState: PoweredOn
  volumes:
    - name: cli-vm-pvc-0
      persistentVolumeClaim:
        claimName: cli-vm-pvc-0

```

Review the settings.

1. Select **DEPLOY VM** to deploy the VM .

Review The VM Load Balancer Deployment

The screenshot shows the 'Virtual Machine Service' page in the VMware Cloud Foundation UI. The table lists the available VMs:

Name	Status	Power State	Zone	Address	VM Image	VM Class	Age
cli-vm	? Unknown	--	--	--	ubuntu-22.04	best-effort-small	0 second

Click **namespace-1** in the breadcrumbs menu to go back to the root of the **Resources** tab

namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network **Resources**

namespace-1

Virtual Machine
1 Virtual Machine
[GO TO SERVICE](#) [CREATE VM](#)

Network
1 Network Service
[GO TO SERVICE](#)

Virtual Machine Image
90 VM Images
[GO TO SERVICE](#)

Kubernetes
0 Kubernetes Clusters
[GO TO SERVICE](#) [CREATE CLUSTER](#)

You should now see 1 **Virtual Machine**

Click **GO TO SERVICE** in the **Network** tile

(#) namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network **Resources**

namespace-1 > Network service

The Networking Service enables you to do self-service network configuration within a namespace and a VPC, such as subnets, static routes, and load balancers.

[Services](#) [VM Load Balancers](#) [VPCs](#)

	Name	Type	Cluster IP	External IP	Ports	Age
»	cli-vm-lb-fn0	Load Balancer	10.96.1.248	10.1.0.13	22/TCP	3 minutes

You will see the LoadBalancer is created

Click **namespace-1** in the breadcrumbs menu to go back to the root of the **Resources** tab. In the **Volume** tile click on **GO TO SERVICE**.

(#) namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network **Resources**

namespace-1 > Volume service

The Volume Service enables you to manage persistent storage for workloads by exposing storage capabilities and capacity.

[Volumes](#) [Volume Snapshots](#) [Storage Classes](#)

List of available persistent volume claims. You can create new volumes, increase their capacity, edit labels of existing volumes, or delete unused volumes.

+ CREATE	Name	Status	Capacity	Health	Zones	Storage Class	Access Modes	Age
»	cli-vm-pvc-0	Bound	5 GiB	Accessible	z-wld-a	k8s-storage-policy	Read Write Once	5 minutes

You will see the new **Persistent Volume** has been created. It also shows that it's current status is **Bound**. Meaning this PVC is currently in use.

Click on **cli-vm-pvc-0**

The screenshot shows the vSphere web interface for managing Kubernetes Persistent Volume Claims (PVCs). The top navigation bar includes 'namespace-1', 'Actions', 'Summary', 'Monitor', 'Configure', 'Permissions', 'Zones', 'Compute', 'Storage', 'Network', and 'Resources'. Below the navigation is a breadcrumb path: 'namespace-1 > Volume service > cli-vm-pvc-0'. The main content area displays the PVC details for 'cli-vm-pvc-0'. Key fields shown include:

- Status:** Bound
- Capacity:** 5 GiB
- Health:** Accessible (9/22/25, 9:42 AM)
- Zones:** z-wild-a
- Storage Class:** k8s-storage-policy
- Access Modes:** Read Write Once
- Volume Mode:** Filesystem
- Age:** 6 minutes
- Labels:** `{vm-selector: cli-vm}`
- Related VM:** cli-vm (highlighted with a red box)

The 'Events' section lists three provisioning events:

Type	Reason	Age	From	Message
Normal	ExternalProvisioning	7 minutes	persistentvolume-controller	Waiting for a volume to be created either by the external provisioner 'csi.vsphere.vmware.com' or manually by the system administrator. If volume creation is delayed, please verify that the provisioner is running and correctly registered.
Normal	Provisioning	7 minutes	csi.vsphere.vmware.com_420e27b584adda015479e5a9bf094a5_2502bbc6-47e3-44cc-a53d-3f5f1996b86d	External provisioner is provisioning volume for claim "namespace-1/cli-vm-pvc-0"
Normal	ProvisioningSucceed	7	csi.vsphere.vmware.com_420e27b584adda015479e5a9bf094a5_2502bbc6-47e3-44cc-a53d-3f5f1996b86d	Successfully provisioned volume pvc-lb38lc0d-410d-482e-b722-e1f590d184d5

At the bottom right of the events table, there are filters and pagination controls: 'Events per page' set to 25, and '3 events' listed.

Here you can see information on the PVC, including what VM it is bound to.

Deploy a Kubernetes Cluster (VKS)

i Switching from a Traditional VM, in the following section you will deploy a **Kubernetes** Cluster using the **vSphere Supervisor Service - VKS**

The screenshot shows the vSphere interface with the 'namespace-1' selected. The top navigation bar includes 'Summary', 'Monitor', 'Configure', 'Permissions', 'Zones', 'Compute', 'Storage', 'Network', and 'Resources'. The 'Resources' tab is active. Below the tabs, there are four main tiles: 'Virtual Machine' (1 Virtual Machine), 'Network' (1 Network Service), 'Virtual Machine Image' (90 VM Images), and 'Kubernetes' (0 Kubernetes Clusters). The 'Kubernetes' tile has a red box drawn around its 'CREATE CLUSTER' button. At the bottom of each tile are 'GO TO SERVICE' and 'CREATE' buttons.

Filling out the cluster configuration has 5 steps. Be sure to scroll to the bottom of each step even if you don't fill anything out. Pay attention to each option.

Return to **namespace-1** if you are not already there and click on the **Resources** tab

1. Click **CREATE CLUSTER** option within the **Resources** tab of the **Kubernetes** tile

Create a New Kubernetes Cluster

The screenshot shows the 'New Kubernetes Cluster' wizard in the VMware Cloud Foundation interface. The 'NEXT' button is highlighted with a red box. The 'Kubernetes Resource YAML' panel on the right shows the generated YAML code:

```
1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: kubernetes-cluster-t3ad
5   namespace: namespace-1
```

1. Verify that the **Cluster API** is selected
2. Select **Custom Configuration**
3. Select **NEXT**

i Once Again take note of the YAML as it gets created by your choices

Name the Cluster

Kubernetes Resource YAML

```

1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: namespace-1
6 spec:
7   clusterNetwork:
8     cidrBlocks:
9       - 192.168.156.0/20
10    services:
11      cidrBlocks:
12        - 10.96.0.0/12
13      serviceDomain: cluster.local
14    topology:
15      class: builtin-generic-v3.4.0
16      classNamespace: vmware-system-vks-public
17      version: v1.33.1---vmware.1-fips-vkr.2
18    variables:
19      - name: kubernetes
20        value:
21          certificateRotation:
22            enabled: true
23            renewalDaysBeforeExpiry: 90
24      - name: vmClass
25        value:
26          enabled: true
27          renewalDaysBeforeExpiry: 90

```

- 1. Cluster Name: vks-01**
2. Scroll to the bottom of the **General Settings** section
3. Click **NEXT**

Select Control Plane Node Size

Kubernetes Resource YAML

```

1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: namespace-1
6 spec:
7   clusterNetwork:
8     cidrBlocks:
9       - 192.168.156.0/20
10    services:
11      cidrBlocks:
12        - 10.96.0.0/12
13      serviceDomain: cluster.local
14    topology:
15      class: builtin-generic-v3.4.0
16      classNamespace: vmware-system-vks-public
17      version: v1.33.1---vmware.1-fips-vkr.2
18    variables:
19      - name: kubernetes
20        value:
21          certificateRotation:
22            enabled: true
23            renewalDaysBeforeExpiry: 90
24      - name: vmClass
25        value:
26          enabled: true
27          renewalDaysBeforeExpiry: 90

```

For VM Class

1. Select **best-effort-xsmall** for VM-Class

2. Scroll to the bottom of this section (Be sure to browse through each option).

The screenshot shows the 'create-cluster' step for a Kubernetes service in namespace-1. The interface has tabs for Summary, Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The Resources tab is active. The 'VM Classes per page' dropdown is set to 1-5 of 16 VM Classes. The 'Storage Class' dropdown is set to k8s-storage-policy. The 'OS Image' dropdown is set to Photon 5 - Kubernetes Service Content Library. The 'Volumes (Optional)' section shows a table with columns Name, Mount path, Storage Class, and Capacity. A message says 'No item found'. Below the table, it says '1 - 5 of 0 Volumes'. A 'NEXT' button is visible. To the right, a 'Kubernetes Resource YAML' panel shows the configuration for the cluster, including API version, kind, metadata, spec, and topology details.

```

1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: namespace-1
6 spec:
7   clusterNetwork:
8     pods:
9       cidrBlocks:
10      - 192.168.156.0/20
11   services:
12     cidrBlocks:
13      - 10.96.0.0/12
14   serviceDomain: cluster.local
15 topology:
16   class: builtin-generic-v3.4.0
17   classNamespace: vmware-system-vks-public
18   version: v1.33.1---vmware.1-fips-vkr.2
19   variables:
20     - name: kubernetes
21       value:
22         certificateRotation:
23           enabled: true
24           renewalDaysBeforeExpiry: 90
25     - name: vmClass

```

3. Click **NEXT**

Add a Nodepool

The screenshot shows the 'create-cluster' step with a nodepool added. The 'NEXT' button is highlighted with a red box. The interface includes sections for General Settings, Control plane, and Node pools. The Node pools section shows a table with columns Name, Zone, Replicas, VM Class, and Storage Class. A message says '1 - 1 of 1 Node pools'. A 'Manage Columns' button is available. To the right, a 'Kubernetes Resource YAML' panel shows the updated configuration for the cluster, including the added nodepool information.

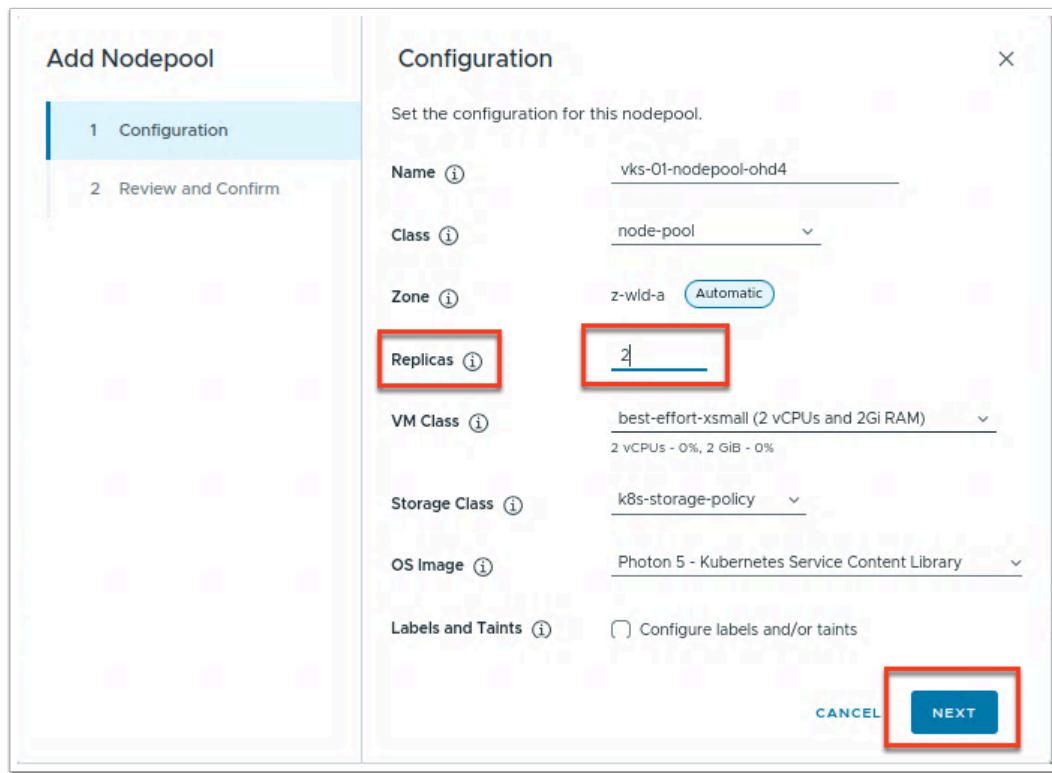
```

1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: namespace-1
6 spec:
7   clusterNetwork:
8     pods:
9       cidrBlocks:
10      - 192.168.156.0/20
11   services:
12     cidrBlocks:
13      - 10.96.0.0/12
14   serviceDomain: cluster.local
15 topology:
16   class: builtin-generic-v3.4.0
17   classNamespace: vmware-system-vks-public
18   version: v1.33.1---vmware.1-fips-vkr.2
19   variables:
20     - name: kubernetes
21       value:
22         certificateRotation:
23           enabled: true
24           renewalDaysBeforeExpiry: 90
25     - name: vmClass

```

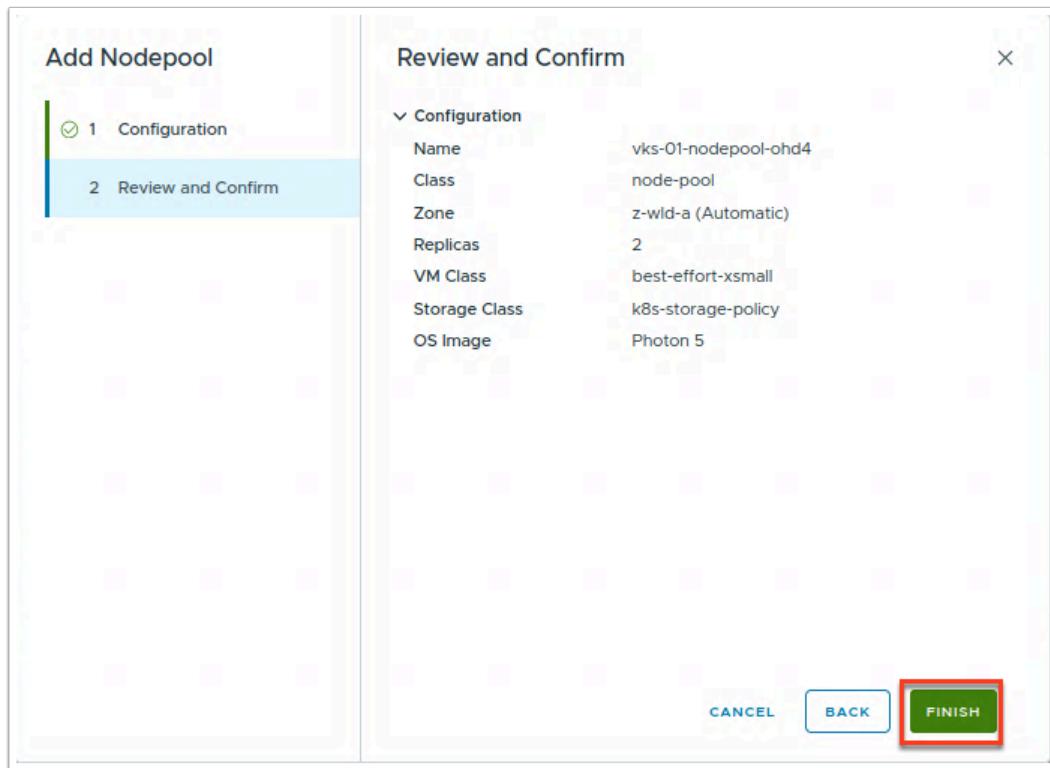
1. Click **+ ADD NODEPOOL**

Add Nodepool Configuration



1. Set **Replicas** to 2
2. Click **NEXT**

Review and Confirm Nodepool



1. Verify everything looks correct and click **FINISH**

Create Cluster Cont'd

```

apiVersion: cluster.x-k8s.io/v1beta1
kind: Cluster
metadata:
  name: vks-01
  namespace: namespace-1
spec:
  clusterNetwork:
    pods:
      cidrBlocks:
        - 192.168.156.0/20
      services:
        cidrBlocks:
          - 10.96.0.0/12
    serviceDomain: cluster.local
  topology:
    class: builtin-generic-v3.4.0
    classNamespace: vmware-system-vks-public
    version: v1.33.1...vmware.1-fips-vkr.2
    variables:
      - name: kubernetes
        value:
          certificateRotation:
            enabled: true
            renewalDaysBeforeExpiry: 90
      - name: vmClass

```

1. Click **NEXT**

Review and Confirm Cluster Create

The screenshot shows a software interface for creating a Kubernetes cluster. At the top, there's a header with a back arrow and the text "Kubernetes Resource YAML". Below this is a tree view under the "Cluster" node, with "vks-01" selected. The main area displays the YAML configuration for this cluster. A tooltip is visible over the "Download all YAMLS to a .zip" button, which is highlighted with a red box. The YAML code is as follows:

```
1 apiVersion: cluster.x-k8s.io/v1beta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: namespace-1
6 spec:
7   clusterNetwork:
8     pods:
9       cidrBlocks:
10      - 192.168.156.0/20
11     services:
12       cidrBlocks:
13      - 10.96.0.0/12
14     serviceDomain: cluster.local
15   topology:
16     class: builtin-generic-v3.4.0
17     classNamespace: vmware-system-vks-public
18     version: v1.33.1---vmware.1-fips-vkr.2
19     variables:
20       - name: kubernetes
21         value:
22       Download all certificateRotation:
23       YAMLs to a .zip
24       enabled: true
25       renewalDaysBeforeExpiry: 90
26       name: vmClass
```

At the bottom left, there are two buttons: a "Cancel" button with a red border and a "Create" button.

1. While reviewing the cluster create settings, you can hover over this button to download a YAML file of the entire cluster creation options to use later via CLI/API to deploy more clusters rapidly.

Review and Confirm Cluster Creation Cont'd

```

apiVersion: cluster.x-k8s.io/v1beta1
kind: Cluster
metadata:
  name: vks-01
  namespace: namespace-1
spec:
  clusterNetwork:
    cidrBlocks:
      - 192.168.156.0/20
    services:
      - cidrBlocks:
          - 10.96.0.0/12
        serviceDomain: cluster.local
  topology:
    class: builtin-generic-v3.4.0
    classNamespace: vmware-system-vks-public
    version: v1.33.1--vmware.1-fips-vkr.2
  variables:
    - name: kubernetes
      value:
        certificateRotation:
          enabled: true
          renewalDaysBeforeExpiry: 90
    - name: vmClass

```

1. Verify everything looks correct and click **FINISH**

Cluster Creation Complete

Name	Status	Kubernetes release	Age
vks-01	Not Ready	v1.33.1--vmware.1-fips-vkr.2	8 seconds

The VKS cluster will now be deployed. Note: this will take some time to complete.

1. Wait for the status to turn toReady
1. This could take up to **5 minutes**

Network Service

(a) namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network Resources

namespace-1

Virtual Machine 2 Virtual Machines

Network 2 Network Services

Virtual Machine Image 90 VM Images

Kubernetes 1 Kubernetes Cluster

Volume 1 Volume

GO TO SERVICE CREATE VM GO TO SERVICE CREATE VOLUME

GO TO SERVICE LAUNCH STANDALONE CLIENT ⓘ

1. Return to the main **Resources** tab page for **namespace-1**
2. In the resources tab of **namespace-1**, select **GO TO SERVICE** on the **Network** tile

Network Service

(a) namespace-1 | ACTIONS

Summary Monitor Configure Permissions Zones Compute Storage Network Resources

namespace-1 > Network service

Network Service

The Networking Service enables you to do self-service network configuration within a namespace and a VPC, such as subnets, static routes, and load balancers.

Services VM Load Balancers VPCs

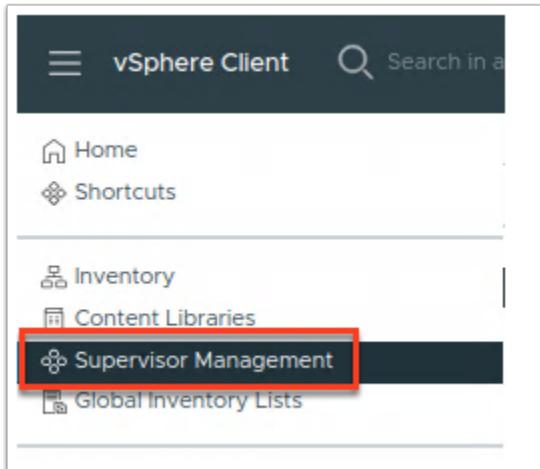
Filter

	Name	Type	Cluster IP	External IP	Ports	Age
»	vks-Q	Load Balancer	10.96.0.192	10.1.0.14	6443/TCP	1 minute
»	cli-vm-lb-fInQ	Load Balancer	10.96.1.248	10.1.0.13	22/TCP	3 hours

Here you can see the **External IP** that was given to your **VKS** cluster upon creation.

Add services in a modular approach

i Next, we will show you how quickly you can add different Services to a **Supervisor**. We will show you how to install the Grafana service, but will not actually deploy it in the lab.



1. On the upper-lefthand dropdown, click **Supervisor Management**

Supervisor Management Services

vSphere Kubernetes Service (VKS) Supervisor Services are components that run within the vSphere Supervisor and are used to provision and manage Kubernetes clusters. These services streamline the deployment and operation of modern applications alongside traditional workloads on vSphere. They offer features like automated lifecycle management, built-in security, and enterprise-grade scalability. Essentially, VKS Supervisor Services simplify the process of running Kubernetes on vSphere, making it more accessible and efficient for IT teams.

In this lab we have deployed a few supervisor services for you, the Consumption Interface which you have used to deploy resources in a namespace and ArgoCD which we will use later in this lab in Module 5.

Due to the limited internet access within this lab we will not be deploying a supervisor service, the following steps will walk you through how a supervisor service is deployed.

The screenshot shows the 'Supervisor Management' interface with the 'Services' tab selected. A red box highlights the 'Services' tab in the top navigation bar. Below it, there's a 'NEW NAMESPACE' input field and a 'Quick Filter' dropdown. The main area displays a table of services with columns: Namespaces, Supervisor, Config Status, CPU (Used | Limit), Memory (Used | Limit), Storage (Used | Limit), Description, and vCenter. The table lists several services, each with a status indicator (green circle with a checkmark) and a link to its details.

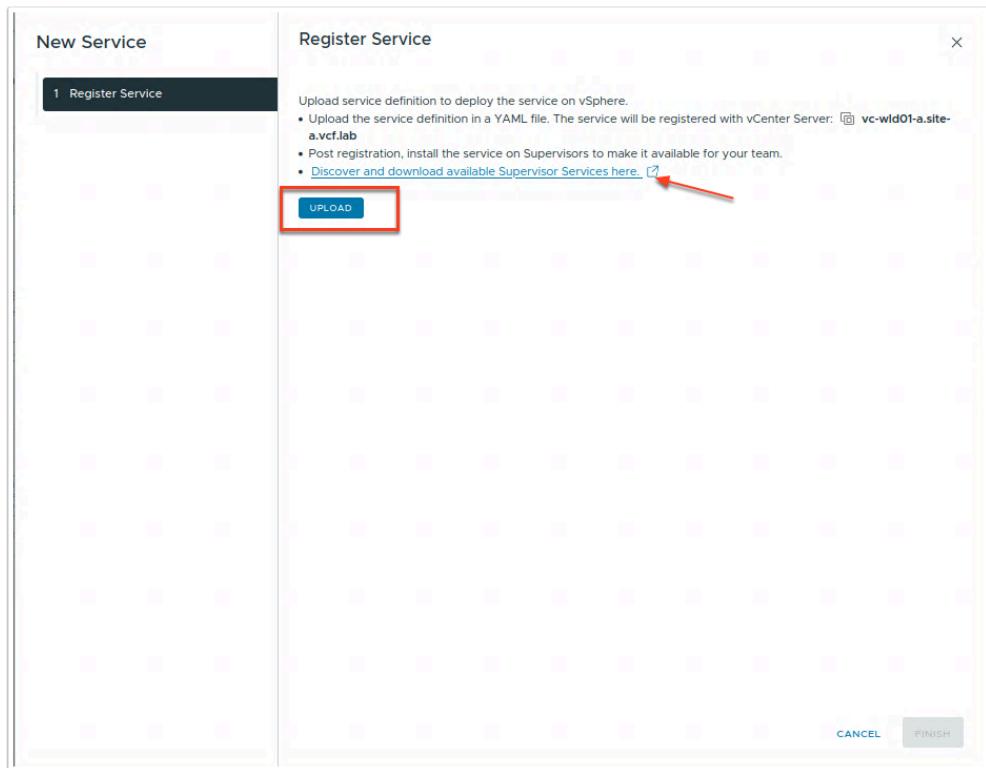
1. Click the **Services** tab

Add Service

The screenshot shows the 'Supervisor Management' interface with the 'Supervisor Services' tab selected. A red box highlights the 'Supervisor Services' tab in the top navigation bar. Below it, there's a 'Sort By' dropdown and a note about supervisor services. The main area displays a grid of service tiles. One tile, 'Add New Service', has a red box around its 'ADD' button. Other tiles include 'VM Service', 'Local Consumption Interface', 'ArgoCD Service', 'harbor', 'Velero vSphere Operator', and 'Kubernetes Service'.

1. Click **ADD** in the **Add New Service** tile

Register Service



1. You can get a list of the currently supported services by clicking the ***Discover and download available Supervisor Services here*** link.
2. Click **UPLOAD** to find a service file to add.

i NOTE : Since this lab sits behind a proxy and firewall you will not be able to access these links from this lab these are for illustrative purposes only. The next few steps are illustrated for you since this lab is behind a firewall you can not access the supervisor services repository.

Download The ArgoCD YAML File

This is an image showing the the Supervisor Services Catalog.

Supervisor Services Catalog

- [vSphere Kubernetes Service \(VKS\)](#)
 - [vSphere Kubernetes Service \(VKS\) Versions](#)
- [Local Consumption Interface](#)
 - [Local Consumption Interface Versions](#)
- [vSAN Data Persistence Platform \(vDPP\) Services:](#)
- [Backup & Recovery Service](#)
 - [Velero vSphere Operator CLI Versions](#)
 - [Velero Versions](#)
- [Certificate Management Service](#)
 - [CA Cluster Issuer Versions](#)
- [Cloud Native Registry Service](#)
 - [Harbor Versions](#)
- [Kubernetes Ingress Controller Service](#)
 - [Contour Versions](#)
- [External DNS Service](#)
 - [ExternalDNS Versions](#)
- [Supervisor Management Proxy](#)
 - [Supervisor Management Proxy Versions](#)

ArgoCD Service

Please refer to [How to find and install Supervisor Services](#) to find and install supervisor services.



Argo CD empowers teams to deliver applications with speed and precision by continuously synchronizing Git-defined desired state with live environments. Argo CD, a leading declarative GitOps continuous delivery tool, revolutionizes how teams deploy and manage applications. It champions a paradigm where the desired state of applications and infrastructure is explicitly defined in Git repositories. This "Git-defined desired state" serves as the single source of truth, offering unparalleled transparency, version control, and auditability for deployments. ArgoCD Service provides the entire lifecycle of Argo CD instance, including create, delete, upgrade ArgoCD and update its configurations. It gives both platform teams and developers access to automated, version-controlled delivery pipelines—whether they're managing vSphere Kubernetes Service (VKS) clusters, VMs, vSphere Pods on supervisor cluster or workloads on VKS clusters.

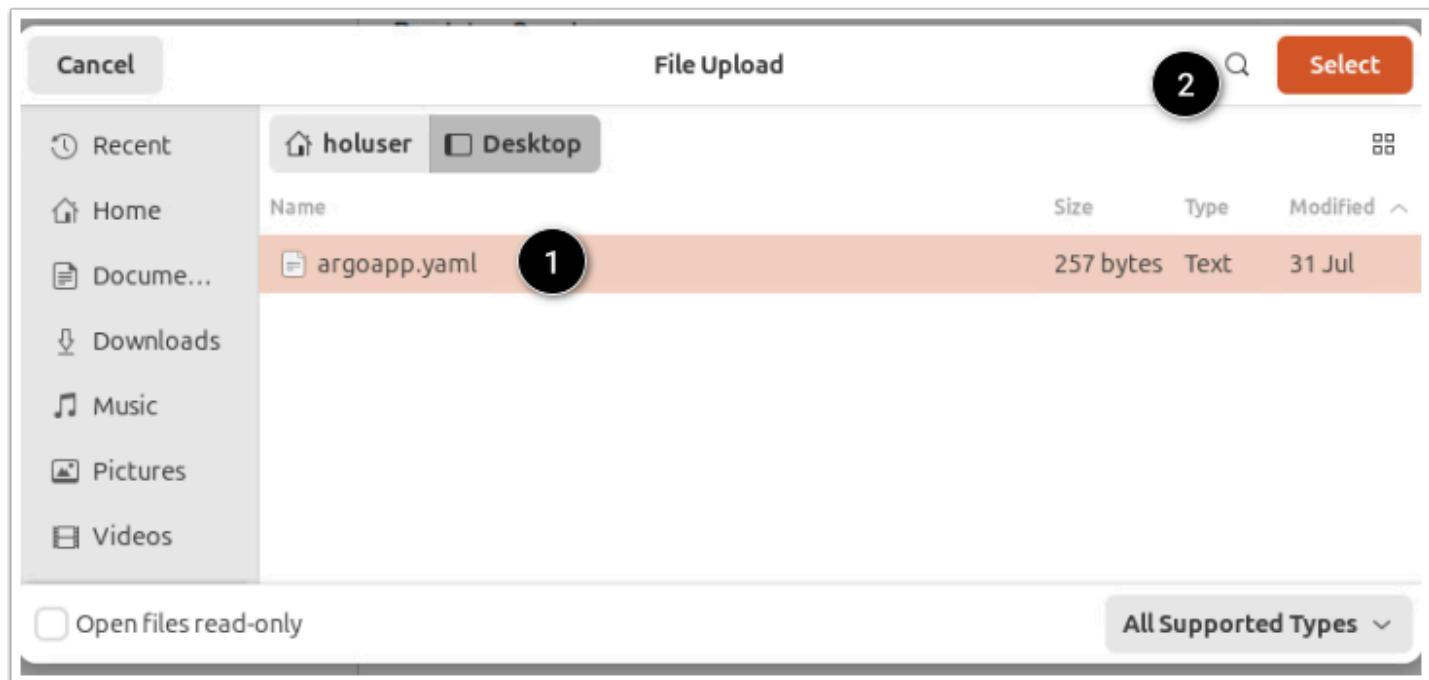
- Service Installation and Configuration [documentation](#)

ArgoCD Service Versions

- v1.0.0
 - [Release Notes](#)
- v1.0.1
 - [Release Notes](#)

Here you would scroll down until you see **ArgoCD** and you would select the version of the **ArgoCD Service yaml** and **cli** from the **Broadcom support portal**.

The ArgoCD YAML File

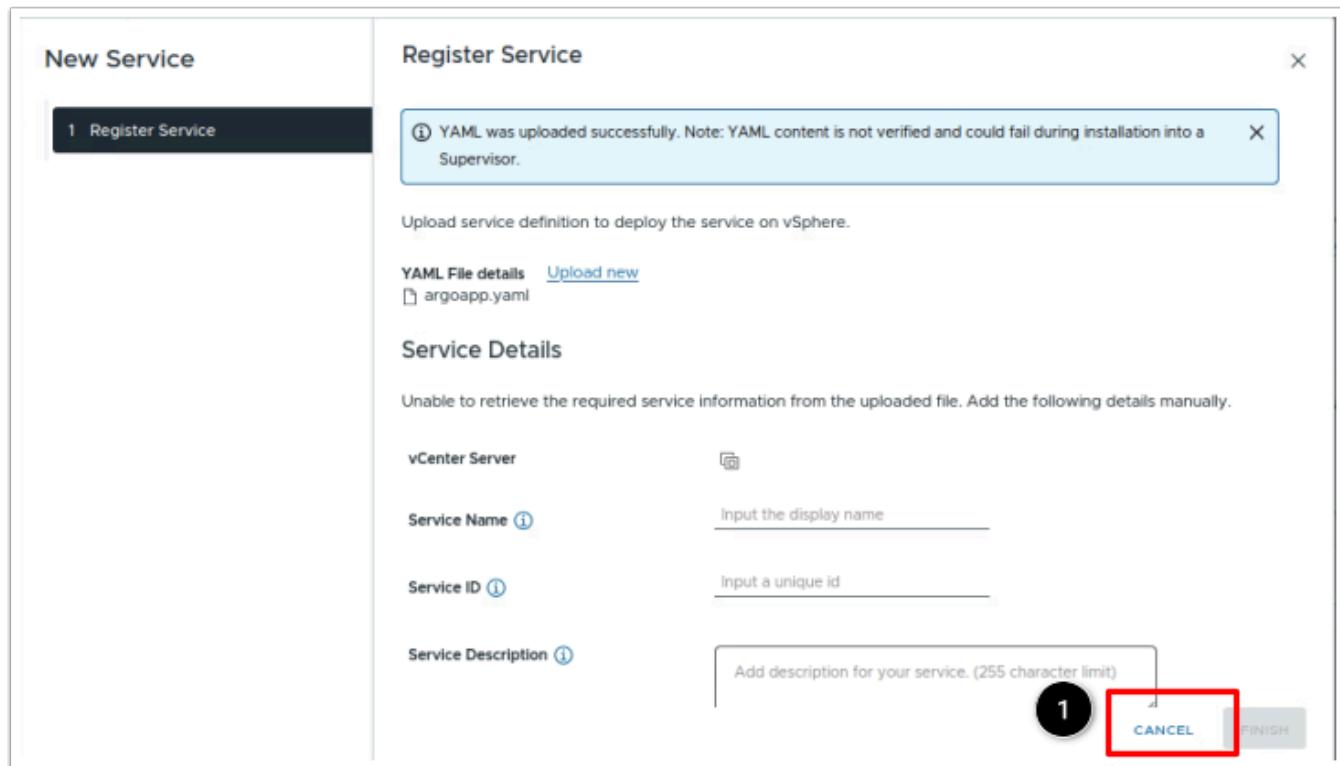


After downloading you would then:

1. Select the **argoapp.yaml** file.
2. Click **Select**.

i *NOTE : Since this lab sits behind a proxy and firewall you will not be able to access these links from this lab these are for illustrative purposes only.*

Register Service Cont'd



As you can see above, the YAML file has been decoded and shows that an ArgoCD service will be installed.

To proceed with the lab you must cancel out of the register service wizard.

1. Select **CANCEL** to close the wizard

Module 3 - Login to The Cluster and Deploy An Application (30 minutes)

Basic

Module 3 - Login to The Cluster and Deploy An Application (30 minutes) Basic

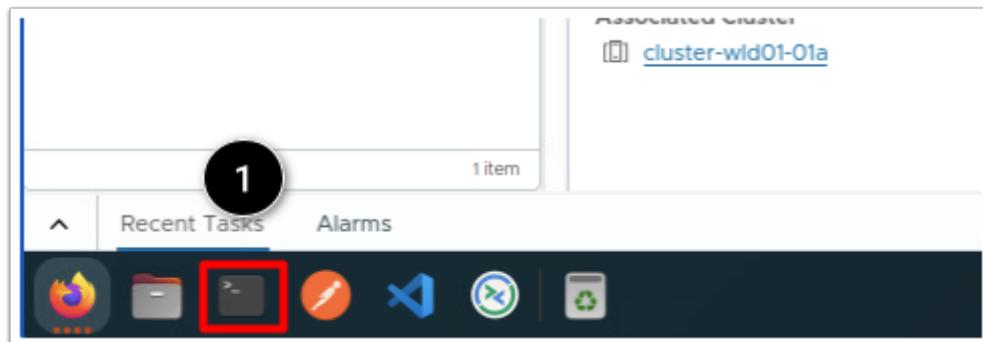
Deploy The nginx Application

- i** In this module we will walk through the steps to deploy the **nginx** application in the namespace and cluster we created previously.

Verify the version of the VCF Command Line Interface (VCF CLI)

! You need to complete Module 2 before continuing with Module 3. At a minimum, you need to create the VKS cluster

Please remember that you can copy and paste any of the following commands into the terminal window



1. Open a terminal window by clicking the terminal icon.

```
holuser@console:~$ kubectl cluster-info
Kubernetes control plane is running at https://10.1.0.2:443
KubeDNS is running at https://10.1.0.2:443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

Check that you are logged into the supervisor by typing the following command

```
kubectl cluster-info
```

Click to copy

If you do not see the response control plane is running to the previous command, log into the Supervisor by typing the command

```
kubectl vsphere login --vsphere-username administrator@wld.sso --server=https://10.1.0.2 --insecure-skip-tls-verify
```

Click to copy

```
holuser@console:~$ vcf version
version: v9.0.0
buildDate: 2025-03-27
sha: d503801
arch: amd64
```

Validate the VCF version by typing the following command

```
vcf version
```

Click to copy

Create a VCF Context for Supervisor

```
holuser@console:~$ vcf context create supervisor --endpoint https://10.1.0.2 -u administrator@wld.sso --insecure-skip-tls-verify --auth-type basic
[!] Refreshing plugin inventory cache for "projects.packages.broadcom.com/vcf-cli/plugins/plugin-inventory:latest", this will take a few seconds.
Provide Password:

Logged in successfully.

You have access to the following contexts:
supervisor
supervisor:argocd
supervisor:namespace-1
supervisor:svc-argocd-service-domain-c10
supervisor:svc-ccl-ns-domain-c10
supervisor:svc-harbor-domain-c10
supervisor:svc-tkg-domain-c10
supervisor:svc-velero-domain-c10

If the namespace context you wish to use is not in this list, you may need to refresh the context again, or contact your cluster administrator.

To change context, use 'vcf context use <context_name>'
[ok] successfully created context: supervisor
[ok] successfully created context: supervisor:svc-argocd-service-domain-c10
[ok] successfully created context: supervisor:argocd
[ok] successfully created context: supervisor:svc-harbor-domain-c10
[ok] successfully created context: supervisor:svc-ccl-ns-domain-c10
[ok] successfully created context: supervisor:svc-velero-domain-c10
[ok] successfully created context: supervisor:namespace-1
[ok] successfully created context: supervisor:svc-tkg-domain-c10
```

Create a VCF context for Supervisor by typing the following command

```
vcf context create supervisor --endpoint https://10.1.0.2 -u administrator@wld.sso --  
insecure-skip-tls-verify --auth-type basic
```

Click to copy

Password:

```
VMware123!VMware123!
```

Click to copy

Set Context to Namespace-1

```
holuser@console:~$ vcf context use supervisor:namespace-1  
[ok] Token is still active. Skipped the token refresh for context 'supervisor:namespace-1'  
[i] Successfully activated context 'supervisor:namespace-1' (Type: kubernetes)  
[i] Fetching recommended plugins for active context 'supervisor:namespace-1'...  
[ok] All recommended plugins are already installed and up-to-date.
```

1

Set your newly created context to use the namespace you created in Module 2 (**namespace-1**) by typing the following command:

```
vcf context use supervisor:namespace-1
```

Click to copy

Check Your Kubernetes Cluster

```
holuser@console:~$ kubectl get cluster  
NAME      CLUSTERCLASS      PHASE      AGE      VERSION  
vks-01    builtin-generic-v3.4.0  Provisioned  12m     v1.33.1+vmware.1-fips
```

1

Check your kubernetes cluster by typing the following command

```
kubectl get cluster
```

Click to copy

Create the Context for the VKS Cluster

```
holuser@console:~$ vcf context create vks-01 --endpoint https://10.1.0.2 -u administrator@wld.sso --workload-cluster-name vks-01 --workload-cluster-namespace namespace-1 --insecure-skip-tls-verify --auth-type basic
Provide Password:

[i] Logging in to Kubernetes cluster (vks-01) (namespace-1)
[i] Successfully logged in to Kubernetes cluster 10.1.0.12
You have access to the following contexts:
  vks-01
  vks-01:vks-01

If the namespace context you wish to use is not in this list, you may need to
refresh the context again, or contact your cluster administrator.

To change context, use 'vcf context use <context_name>'
[ok] successfully created context: vks-01
[ok] successfully created context: vks-01:vks-01
```

Create a VCF context for theVKS-01cluster by typing the following command

```
vcf context create vks-01 --endpoint https://10.1.0.2 -u administrator@wld.sso --
workload-cluster-name vks-01 --workload-cluster-namespace namespace-1 --insecure-skip-
tls-verify --auth-type basic
```

Click to copy

If prompted for a password use :

```
VMware123!VMware123!
```

Click to copy

Set the Context to Use the VKS-01 Cluster

```
holuser@console:~$ vcf context use vks-01:vks-01
1 [ok] Token is still active. Skipped the token refresh for context "vks-01:vks-01"
[i] Successfully activated context 'vks-01:vks-01' (Type: kubernetes)
[i] Fetching recommended plugins for active context 'vks-01:vks-01'...
[ok] No recommended plugins found.
```

Set the context to the VKS-01 Cluster by typing the following command

```
vcf context use vks-01:vks-01
```

Click to copy

NAME	CURRENT	TYPE
supervisor	false	kubernetes
supervisor:argocd	false	kubernetes
supervisor:namespace-1	false	kubernetes
supervisor:svc-argocd-service-domain-c10	false	kubernetes
supervisor:svc-cci-ns-domain-c10	false	kubernetes
supervisor:svc-harbor-domain-c10	false	kubernetes
supervisor:svc-tkg-domain-c10	false	kubernetes
supervisor:svc-velero-domain-c10	false	kubernetes
vks-01	false	kubernetes
vks-01:vks-01	true	kubernetes

To verify that you are using the correct context type the following command

```
vcf context list
```

Click to copy

Verify that the **context vks-01:vks-01** is set to **true**

NAME	STATUS	ROLES	AGE	VERSION
vks-01-hlrxz-cdf4d	Ready	control-plane	14m	v1.33.1+vmware.1-fips
vks-01-vks-01-nodepool_v35e-2mrn2-kmpv6-mc8zb	Ready	<none>	10m	v1.33.1+vmware.1-fips
vks-01-vks-01-nodepool_v35b-2mrn2-kmpv6-mc8zb	Ready	<none>	10m	v1.33.1+vmware.1-fips

You can also validate the context is correct by typing the following command

```
kubectl get nodes
```

Click to copy

Verify that you can see the **control-plane** and **nodepool nodes**

Let's Deploy An Application

```
holuser@console:~$ cd app  
holuser@console:~/app$ ls  
nginx-deploy.yaml
```

1

2

We'll deploy the **nginx** application in the **vks-01cluster**. Let's start by changing to the app directory, type the following command

```
cd ~/app
```

Click to copy

```
holuser@console:~/app$ cat nginx-deploy.yaml  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: nginx-deployment  
  namespace: app  
  labels:  
    app: nginx  
spec:  
  replicas: 2  
  selector:  
    matchLabels:  
      app: nginx  
  template:  
    metadata:  
      labels:  
        app: nginx  
    spec:  
      containers:  
        - name: nginx  
          image: harbor-01a.site-a.vcf.lab/vcf/nginx  
          ports:  
            - containerPort: 80  
          imagePullSecrets:  
            - name: harbor-creds  
...  
apiVersion: v1  
kind: Service  
metadata:  
  name: nginx-service  
  namespace: app  
spec:  
  selector:  
    app: nginx  
  ports:  
    - name: http  
      protocol: TCP  
      port: 80  
      targetPort: 80
```

Let's review the files for the application we are going to deploy, type the following commands:

```
cat nginx-deploy.yaml
```

Click to copy

Review the yaml file and move on. No changes to the file are necessary.

Create a Namespace for the Application on the VKS-01 Cluster

```
holuser@console:~/app$ 1 kubectl create ns app  
namespace/app created
```

Create a kubernetes namespace by typing the following command

```
kubectl create ns app
```

Click to copy

Set the Pod Security Policy for the Namespace

```
holuser@console:~/app$ 1 kubectl label --overwrite ns app pod-security.kubernetes.io/enforce=privileged  
namespace/app labeled
```

Set the pod security policy by typing the following command

```
kubectl label --overwrite ns app pod-security.kubernetes.io/enforce=privileged
```

Click to copy

Create A Pull Secret For Harbor

```
holuser@console:~/app$ kubectl create secret docker-registry harbor-creds --docker-server=10.1.0.7 --docker-username=admin --docker-password=VMware123! -n app  
secret/harbor-creds created  
holuser@console:~/app$
```

To create the pull secret for Harbor type the following command

```
kubectl create secret docker-registry harbor-creds --docker-server=10.1.0.7 --docker-username=admin --docker-password=VMware123! -n app
```

Click to copy

Deploy the nginx Application in the app Namespace

```
holuser@console:~/app$ kubectl apply -f nginx-deploy.yaml -n app
deployment.apps/nginx-deployment created
service/nginx-service created
holuser@console:~/app$
```

To deploy the **nginx** application type the following command

```
kubectl apply -f nginx-deploy.yaml -n app
```

Click to copy

You should see the message above that the service was created.

List All of the Resources Deployed for the Application In the Namespace

```
holuser@console:~/app$ kubectl get all -n app
NAME                                         READY   STATUS    RESTARTS   AGE
pod/nginx-deployment-64594fdd87-cw8n9     1/1    Running   0          73s
pod/nginx-deployment-64594fdd87-ksgv5      1/1    Running   0          73s

NAME           TYPE      CLUSTER-IP        EXTERNAL-IP       PORT(S)    AGE
service/nginx-service   LoadBalancer   10.100.120.178   10.1.0.15        80:31790/TCP   73s

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment   2/2     2           2           73s

NAME           DESIRED  CURRENT    READY   AGE
replicaset.apps/nginx-deployment-64594fdd87 2        2         2       73s
holuser@console:~/app$
```

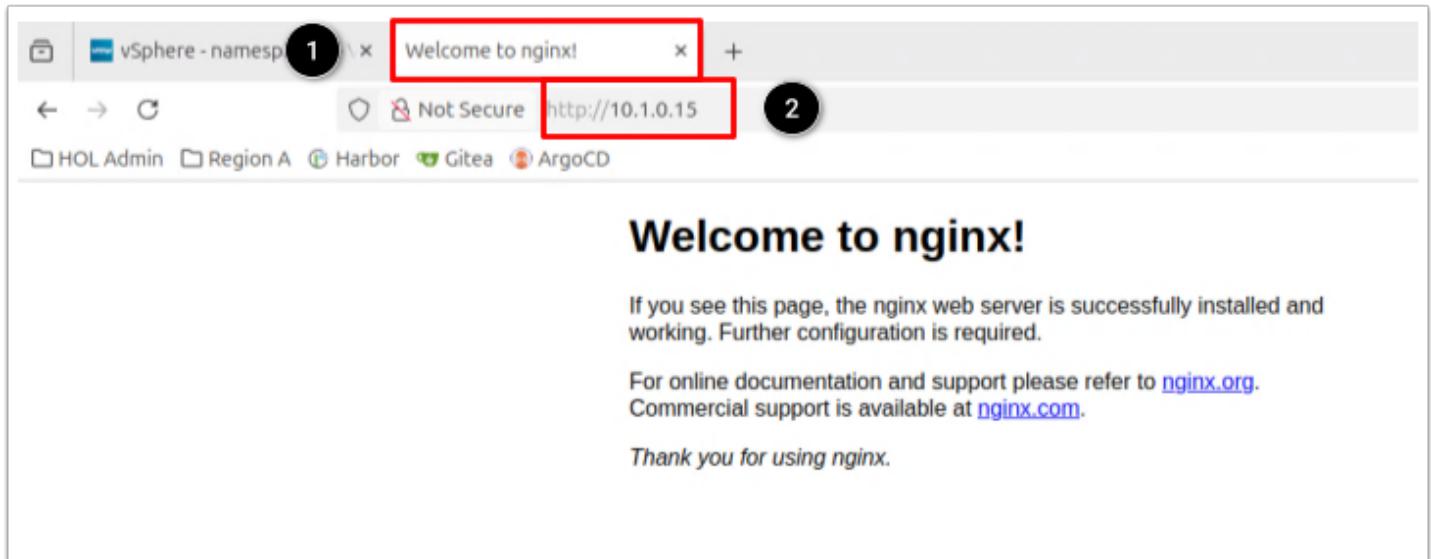
Check the details of the deployed **nginx** application. To see all the services deployed type the following command

Take note of the **external** IP address of the **nginx** application

```
kubectl get all -n app
```

Click to copy

Check that nginx Is Working



Check that **nginx** is running. Return to the browser

1. Open a new tab in the browser
2. Enter the external IP address gathered in the previous step

A screenshot of a terminal window. The prompt is 'holuser@console:~/app\$'. The user runs the command 'curl 10.1.0.15'. The terminal displays the full 'Welcome to nginx!' HTML page, including the title, styles, and body content. The entire output is shown in a monospaced font.

```
holuser@console:~/app$ curl 10.1.0.15
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
holuser@console:~/app$
```

You can also use the curl command to validate **nginx** is running by typing the following command

```
curl xx.xx.xx.xx
```

 Click to copy

Use the external IP address from the previous step

Module 4 - Updating The Supervisor and VKS Services (15 minutes)

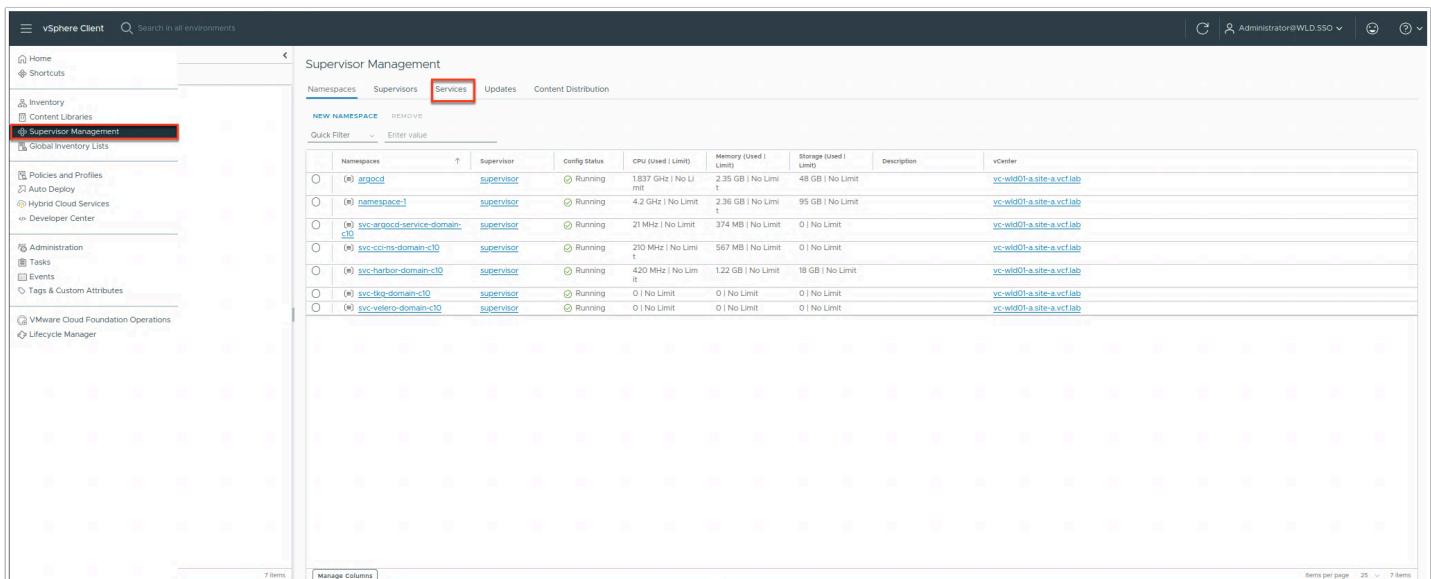
Basic

Module 4 - Updating The Supervisor and VKS Services (15 minutes) Basic

VKS Update

In Module 2 of this lab we deployed a version of the vSphere Kubernetes Service, we deployed the 1.33 version which at this time was the latest available. But, kubernetes versions change quickly so you will need to update the VKS service to stay in lockstep with the rapid pace of new kubernetes releases.

Update The VKS Service



The screenshot shows the vSphere Client interface with the 'Supervisor Management' section selected. The 'Services' tab is highlighted with a red box. A table lists several supervisor services across different namespaces:

Name	Type	Status	CPU (Used Limit)	Memory (Used Limit)	Storage (Used Limit)	Description	vCenter
argocd	supervisor	Running	1.87 GHz No Limit	3.35 GB No Limit	48 GB No Limit		vc-wld01-a-site-a-vctlab
nameSpace-1	supervisor	Running	4.2 GHz No Limit	2.36 GB No Limit	95 GB No Limit		vc-wld01-a-site-a-vctlab
svc-argocd-service-domain-c10	supervisor	Running	21 MHz No Limit	374 MB No Limit	0 No Limit		vc-wld01-a-site-a-vctlab
svc-eccns-domain-c10	supervisor	Running	210 MHz No Limit	567 MB No Limit	0 No Limit		vc-wld01-a-site-a-vctlab
svc-harbor-domain-c10	supervisor	Running	420 MHz No Limit	1.22 GB No Limit	18 GB No Limit		vc-wld01-a-site-a-vctlab
svc-tkg-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit		vc-wld01-a-site-a-vctlab
svc-velero-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit		vc-wld01-a-site-a-vctlab

At any given time there will be multiple versions of the VKS service available for download. You can review the available versions by executing the following steps:

1. Navigate to **Supervisor Management**
2. Click **Services**

Update The VKS Service(Contd.)

The screenshot shows the vSphere Client interface under the 'Supervisor Management' section, specifically the 'Services' tab. It lists several supervisor services:

- VM Service**: Status: Active. This service allows developers to self-service VMs and allows you to set policies for VM deployment.
- Local Consumption Interface**: Status: Active. Provides the Local Consumption Interface for N...
- ArgoCD Service**: Status: Active. This service allows users to self-service ArgoCD ...
- harbor**: Status: Active. OCI Registry
- Velero vSphere Operator**: Status: Active. Helps users install Velero and the vSphere plug... Actions: Manage
- Kubernetes Service**: Status: Active. Cluster management. Actions: Manage Service, Add New Version, Manage Versions, Edit, Delete.

Under Kubernetes Services:

1. Click **Manage Versions**

Manage Versions: Kubernetes Service
Service ID: tkg.vsphere.vmware.com

Find details about all versions of 'Kubernetes Service' in the below table.

- To delete a service version, first deactivate the version and then remove it from the Supervisors where it is installed.
- To delete a service, first deactivate the entire service and then remove all of its versions from the Supervisors where it is installed.

You cannot create instances on Supervisors with deactivated versions and services.

DEACTIVATE VERSION	DELETE		
<input type="radio"/> Kubernetes Service	3.4.0+v1.33	Active	1
<input type="radio"/> Kubernetes Service	3.3.3-embedded	Active	0
<input type="radio"/> Kubernetes Service	3.3.1-embedded	Active	0

Manage Columns 3 items

Deactivate entire service **CONFIRM**

You must deactivate a service before deleting it.

- By deactivating the service you deactivate all its service versions.
- You will be unable to add or change service versions.
- You will be unable to install service versions on Supervisors.

CLOSE

This is where Platform Admins will select the new version to install. The lifecycle of the various supervisor services are independent. Some services such as Velero are considered core/critical services and are installed by default while other such as ArgoCD which we will cover later are optional and installed by the Platform Administrator in either case, ,you have full flexibility to

update (or not) to a new version as soon as they are released, without having to update VCF/vCenter, the supervisor or other supervisor services.

1. Click **CLOSE** when you are done reviewing.

Supervisor Updates

The screenshot shows the vSphere Client interface with the 'Supervisor Management' tab selected. In the 'Updates' section, there are two main sections: 'Latest Supervisor Update' and 'Available Supervisor Updates'. The 'Latest Supervisor Update' section shows a single update entry: 'vSphere Namespace Update 9.0.0.0100-24845085, Jul 10, 2025' with a note about compatibility with vSphere 10.0. The 'Available Supervisor Updates' section lists two updates: 'v1.30.5+vmware.4-lfps-vsc9.0.0.0-24686447, Apr 7, 2025' and 'v1.29.14+vmware.1-lfps-vsc9.0.0.0100-24845085, Jul 10, 2025'. Below these sections is a table titled 'APPLY UPDATES' with columns for 'Supervisor Name', 'Current Version', 'Available Versions', and 'Last Updated Time'. A message at the bottom states 'No items found'.

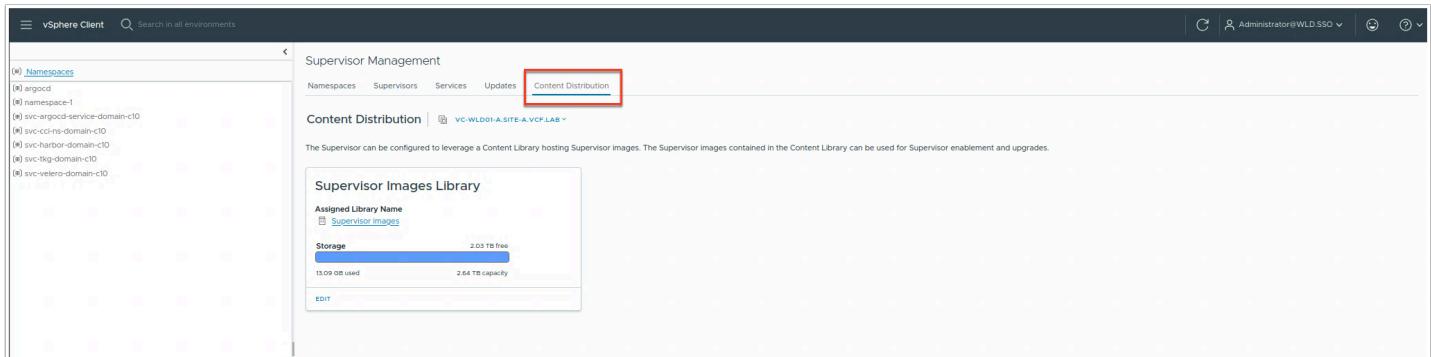
Navigate to Supervisor Management - Updates

This lab has been already updated for you but review the sequence to update the supervisor.

1. Always a good idea to review the compatibility matrix.
2. And always good to review the release notes.

The screenshot shows the Broadcom Product Interoperability Matrix tool. The 'Interoperability' tab is selected. On the left, there's a 'Compare' section with dropdown menus for 'Select a Solution' and 'Select a Version', and a 'With' dropdown. The main area is titled 'Interoperability Result' and displays a legend: 'Compatible' (green), 'Part of Previous Release' (orange), 'Part of Current Release' (blue), and 'Not Supp'd' (red). A note below the legend says 'Most interoperable products and versions are now displayed.' At the bottom right, there are 'Compatibility Order' and 'Download' buttons.

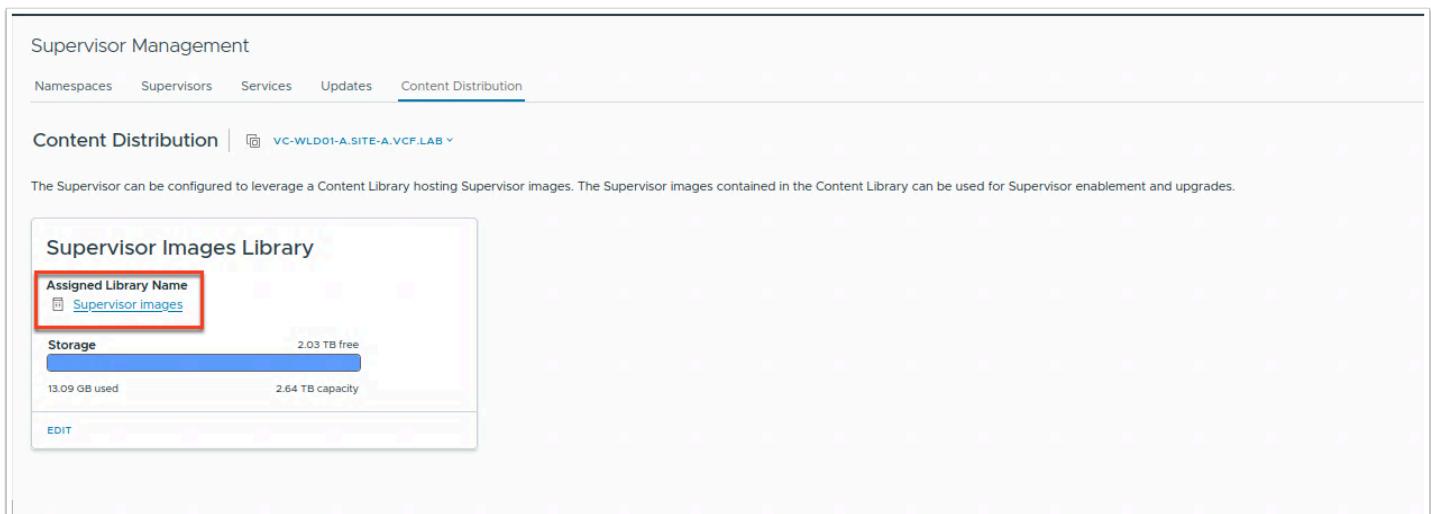
Content Distribution



1. Click **Content Distribution**.

Updates are distributed through a content library configured in Content Distribution tab.

Edit Settings



You can edit the settings for the supervisor services image.

1. Click **Supervisor images**.

This is important moving forward as new Supervisor releases bring not only new version of kubernetes for control plane, but also new functionality and services. This first one brought ArgoCD Service.

Edit The Supervisor Image

The screenshot shows the 'Supervisor images' section of the vSphere Web Client. On the left, there's a table listing content libraries, with one named 'super' selected. On the right, a detailed view of the selected library is shown, including its size (1 GB), last modified date (08/07/2020 25, 12:00:00 AM), last sync date (08/07/2020 25, 12:00:00 AM), content library ('Supervisor images'), UUID (urn:vapic:om.vmware.content.library.item:f1c6cd0-003b-4998-bea-3686e7bb930e:b3909cb7-999d-454a-bfa3-0faf86bdd767), content version (2), and description. A context menu is open over the 'super' library, with the 'Edit Settings...' option highlighted by a red box.

1. Click the **Actions** dropdown.
2. Click **Edit Settings**.

Edit The Supervisor Images

The screenshot shows the 'Edit Settings' dialog for the 'Supervisor images' content library. It contains several configuration options:

- Automatic synchronization:** An unchecked checkbox for enabling automatic synchronization with the external content library.
- Subscription URL:** A text input field containing the URL <https://wp-content.broadcom.com//supervisor/v1/latest/lib.json>.
- Authentication:** An unchecked checkbox for enabling user authentication.
- Library content:** A radio button group where the 'Download library content only when needed' option is selected. A note below explains that this saves storage space by storing only metadata for items, requiring synchronization of items or the whole library.
- Security policy:** A section with an unchecked checkbox for applying a security policy and a dropdown menu set to 'OVF default policy'.

At the bottom right of the dialog, there are two buttons: 'CANCEL' (highlighted with a red box) and 'OK'.

Notice that here you can enable automatic synchronization, downloads and security policy.

1. Click **CANCEL**.

Module 5 -Unlocking GitOps with ArgoCD Service(20 minutes)

Advanced

Module 5 -Unlocking GitOps with ArgoCD Service(20 minutes) Advanced

In this module we will be deploying an application using a combination of Gitea which is a free and open-source software forge for hosting Git repositories, providing features similar to GitHub but designed for self-hosting. And ArgoCD which is a declarative, GitOps continuous delivery tool for Kubernetes. It automates the deployment of applications to specified target environments by continuously comparing the running state of applications with the desired state defined in a Git repository. Argo CD then automatically synchronizes the applications to match the desired state, ensuring consistency and enabling automated deployments.

With VCF, ArgoCD is an optional supervisor service which would be deployed by the platform administrator in this lab the ArgoCD service has already been installed and an instance deployed for you.

Review the ArgoCD service

We'll start by checking the status of the service



1. Click the terminal icon in the taskbar and open a terminal window.

Switch context to argocd namespace

```
holuser@console:~/app$ vcf context use supervisor:argocd
[ok] Token is still active. Skipped the token refresh for context "supervisor:argocd"
[i] Successfully activated context 'supervisor:argocd' (Type: kubernetes)
[i] Fetching recommended plugins for active context 'supervisor:argocd'...
[ok] All recommended plugins are already installed and up-to-date.
holuser@console:~/app$
```

For this lab we have created the ArgoCD service in the namespace argocd. Switch context to use the argocd name space, type the command:

```
vcf context use supervisor:argocd
```

Click to copy

If prompted for a password use:

```
VMware123!VMware123!
```

Click to copy

Review the deployment

```
holuser@console:~/app$ kubectl get all
NAME                                         READY   STATUS    RESTARTS   AGE
pod/argocd-application-controller-0          1/1     Running   0          4d5h
pod/argocd-redis-74b45cdb48-l7kjb           1/1     Running   0          4d5h
pod/argocd-redis-secret-init-l9g65          0/1     Completed  0          40d
pod/argocd-repo-server-5c7db7b498-75sp2     1/1     Running   0          4d5h
pod/argocd-server-686d44f776-fldpn          1/1     Running   0          4d5h

NAME              TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
service/argocd-redis       ClusterIP   10.96.1.234    <none>           6379/TCP        40d
service/argocd-repo-server   ClusterIP   10.96.0.94     <none>           8081/TCP        40d
service/argocd-server        LoadBalancer 10.96.0.233   10.1.0.9        80:31705/TCP,443:32231/TCP  40d
service/vks-01            LoadBalancer 10.96.1.207   10.1.0.11       6443/TCP        40d

NAME             READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/argocd-redis      1/1      1           1           40d
deployment.apps/argocd-repo-server 1/1      1           1           40d
deployment.apps/argocd-server      1/1      1           1           40d

NAME          DESIRED   CURRENT   READY   AGE
replicaset.apps/argocd-redis-74b45cdb48  1         1         1         40d
replicaset.apps/argocd-repo-server-5c7db7b498 1         1         1         40d
replicaset.apps/argocd-server-686d44f776     1         1         1         40d

NAME          READY   AGE
statefulset.apps/argocd-application-controller 1/1     40d

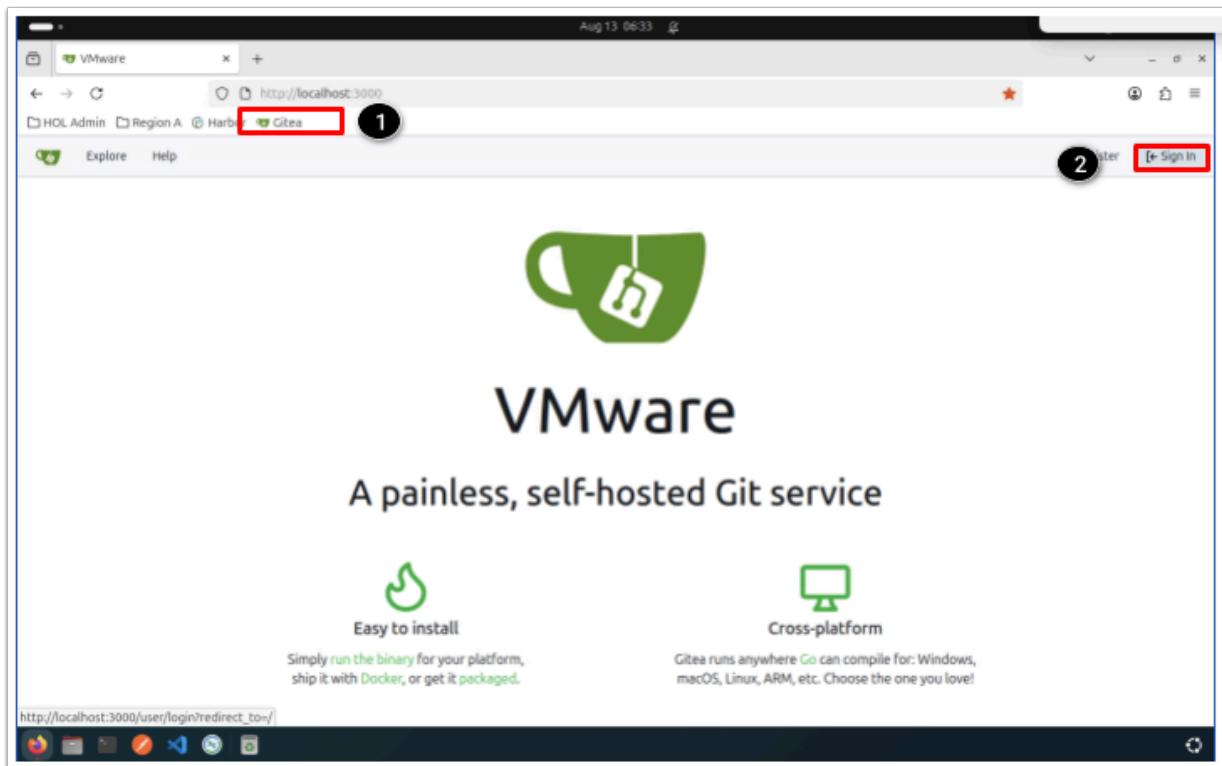
NAME          STATUS    COMPLETIONS   DURATION   AGE
job.batch/argocd-redis-secret-init   Complete  1/1          18s        40d
holuser@console:~/app$
```

Review the deployment by typing:

```
kubectl get all
```

Click to copy

Open A Browser Tab to Gitea



If you are not familiar with Gitea it is a lightweight, self-hosted Git service that provides a platform for software development and version control, similar to GitHub. We have already deployed an instance of Gitea for you

1. Open the **gitea** browser tab in a new tab (you will need to return to gitea later)
2. Click **Sign In**.

Login to Gitea

Sign In

Username or Email Address *

Password [Forgot password?](#)

Remember This Device

Sign In

or

 [Sign in with OpenID](#)

[Sign in with a passkey](#)
[Need an account? Register now.](#)

1. Login with : Username

holuser

Click to copy

2. Password:

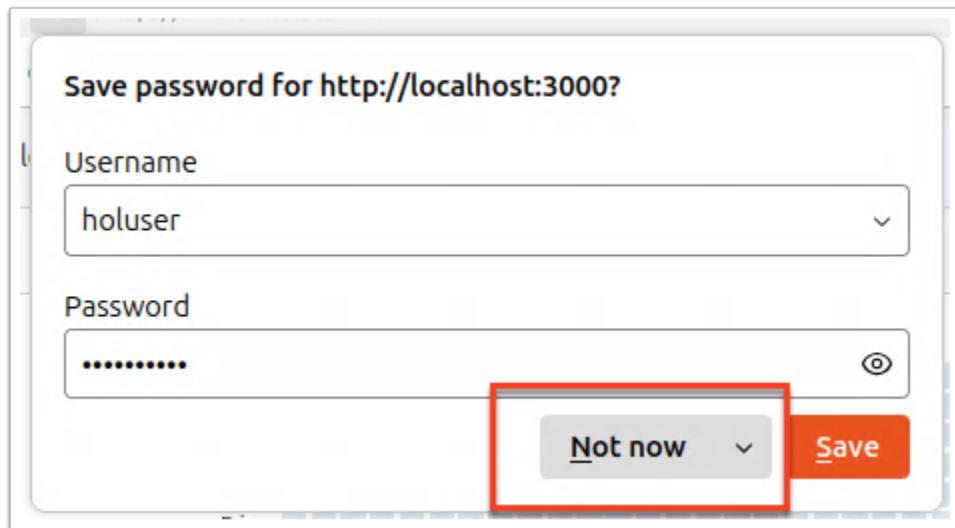
VMware123!

Click to copy

3. Click **Sign In**



Do not close the **gitea** tab as we will need it later in this module.



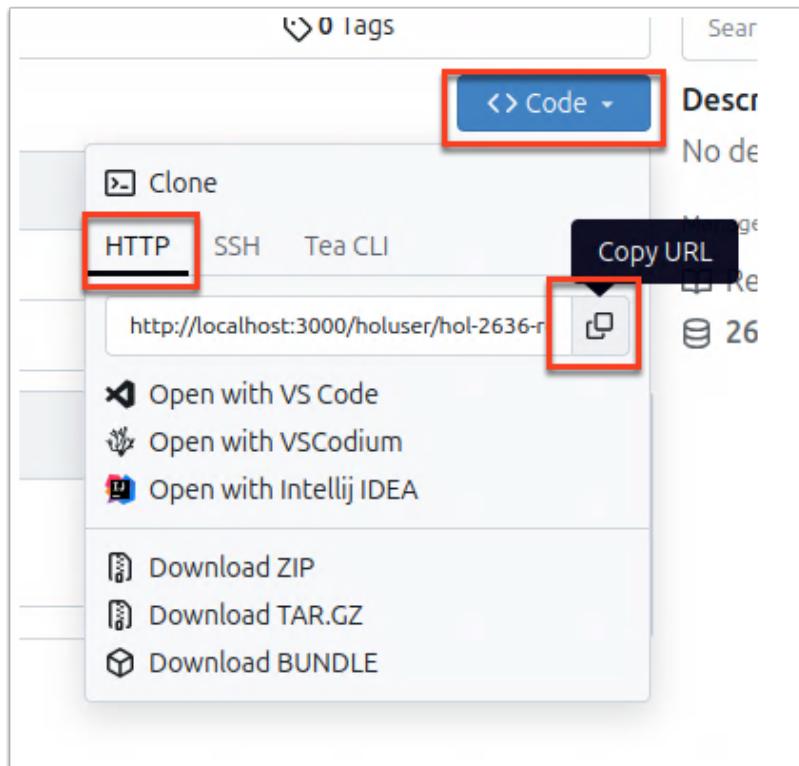
If this window pops up click **Not now**

Switch to the holuser repository

The building team, to save time, has created and populated a repository **holusser** for you.

1. Click on **holuser/2636-repo**

Click the Code dropdown

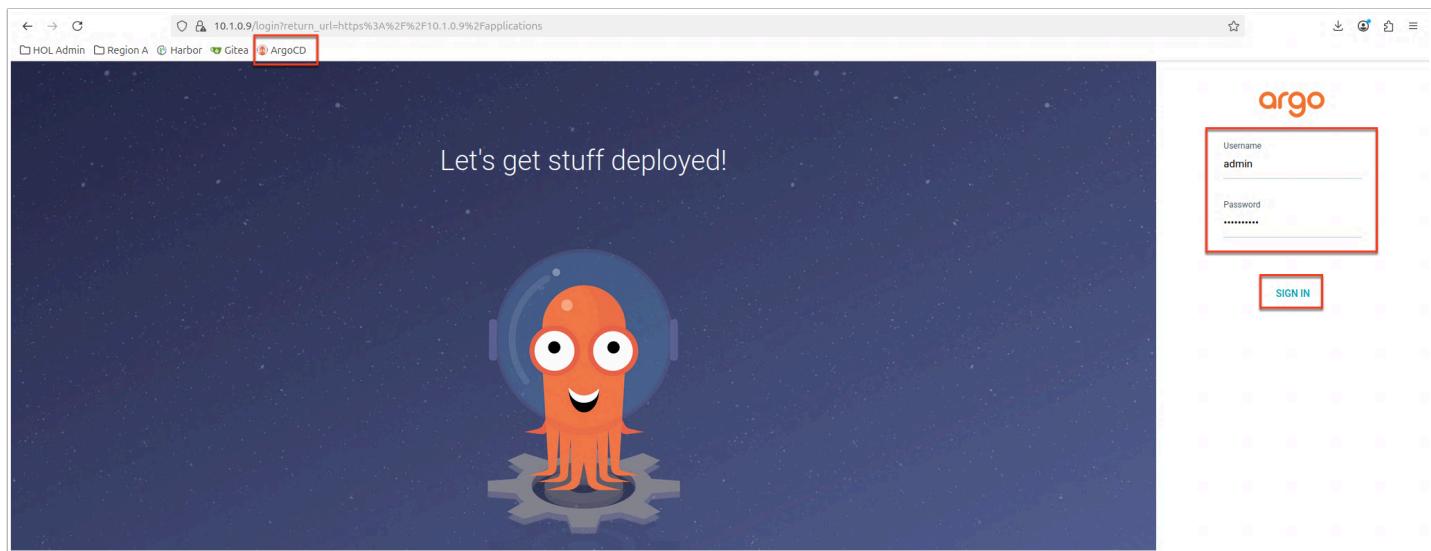


Here is where you would copy the URL for the repository which you would point ArgoCD to. In the YAML code which follows you will see we filled the repository URL in for you however, normally you would need to copy this URL to use when pointing ArgoCD to the code repository.

1. Click the **Code** dropdown
2. Copy the **URL for the holuser/2636-repo**.

i Note that we do not actually use this URL in the next steps; this is just an example of where/how you would obtain the URL for the repository for ArgoCD.

Login to ArgoCD



Now, let's login to the ArgoCD supervisor service we created for you for this lab. Remember that ArgoCD is an optional supervisor service which will not be installed by default.

We have created an app for you, VKS-1 which was created by pulling a file from the gitea repository.

1. Open the ArgoCD in a new browser tab.
2. Login to ArgoCD instance with:

Username:

holuser

Click to copy

Password:

VMware123 !

Click to copy

3. Click **Sign In**

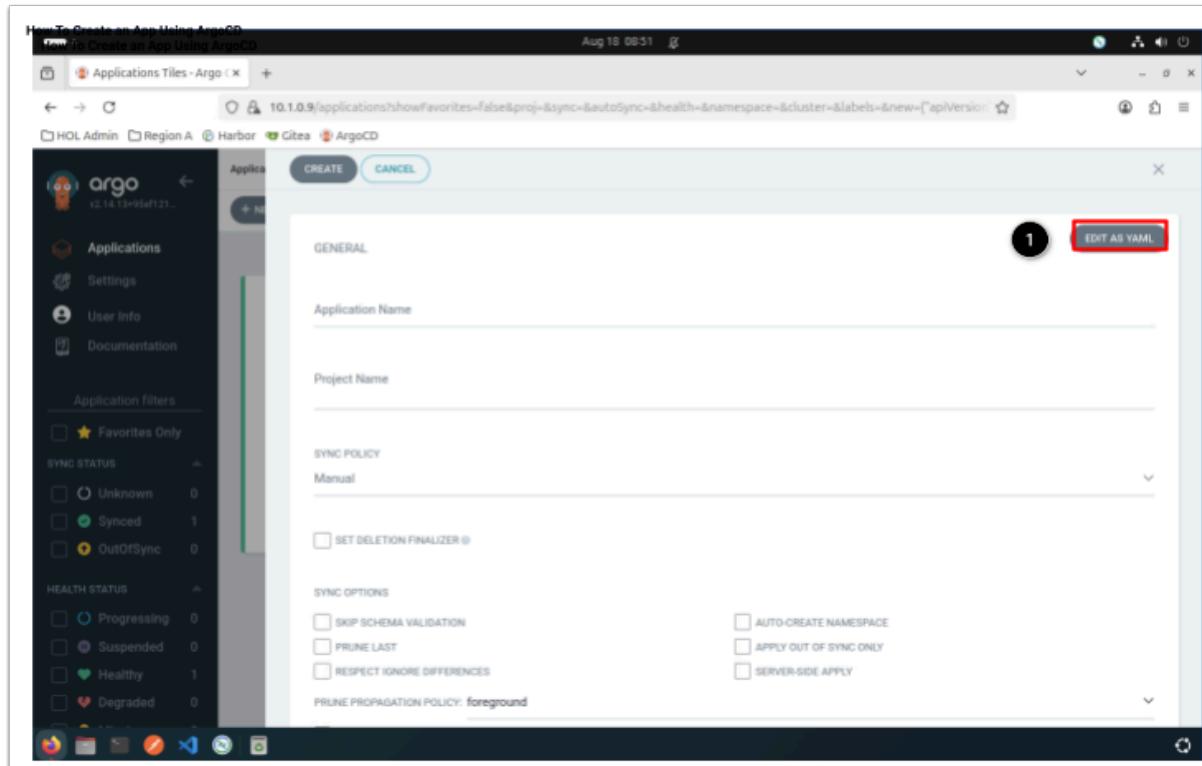
How To Create an App Using ArgoCD

We have already created an app for you, **vks-01** using code pulled from the repository gitea. In the next few steps we will walk through the process by which you would create an application.

Note these steps are for reference and would fail if you tried to deploy the actual application (since vks-01 already exists).

1. Click on **Applications** in the left **menu** (under argo)
2. Click **+New App**

How To Create an App Using ArgoCD (contd.)



To create a new application in Argo CD you can walk through a guided wizard or enter YAML code.

By default the wizard is deployed. You can directly edit YAML code by clicking "EDIT AS YAML"

Here is an example using the YAML we used to create the VKS-1 application (note the repoURL is filled in for you in the code block, normally you would have to provide this, the repoURL being the URL you pulled from gitea previously)

Example only you do not need to use this code

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: vks-01
spec:
  destination:
    namespace: argocd
    server: https://10.1.0.2:443
  source:
```

```
path: .
repoURL: http://10.1.10.130:3000/holuser/hol-2636-repo.git
targetRevision: HEAD
directory:
  recurse: true
sources: []
project: default
syncPolicy:
  automated:
    prune: false
    selfHeal: false
```

Click to copy

How To Create an App Using ArgoCD (contd.)



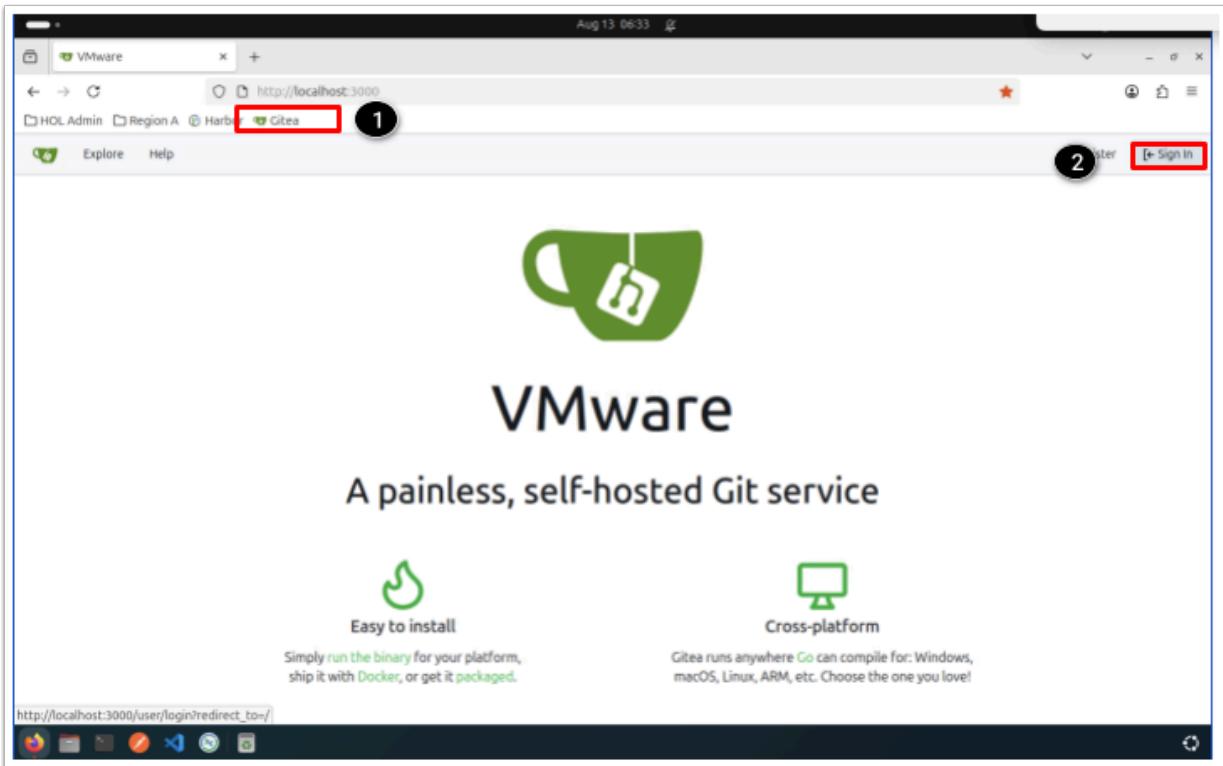
The screenshot shows a modal window titled "CREATE" with a "CANCEL" button at the top left and a "SAVE" button at the top right. A red box highlights the "CANCEL" button. The main area contains the following YAML code:

```
1 apiVersion: argoproj.io/v1alpha1
2 kind: Application
3 metadata:
4   name: vks-01
5   spec: null
6 destination:
7   namespace: argocd
8   server: 'https://10.1.0.2:443'
9 source:
10  path: .
11  repoURL: 'http://10.1.10.130:3000/holuser/hol-2636-repo.git'
12  targetRevision: HEAD
13  directory: null
14  recurse: true
15  sources: []
16  project: default
17  syncPolicy:
18    automated: null
19    prune: false
20    selfHeal: false
21
```

-  Note that the fields in the YAML code match the prompts in the wizard. Again note you are not deploying this.

Click **CLOSE** as this application already exists for you

Scaling The vks-01 Application



One of the largest benefits of containerized applications deployed via pipeline is the ability to rapidly iterate and scale. In the following steps we will walk through an example of scaling the **vks-01** application. In practice you would likely use an autoscaler but for informational purposes we will walk through the manual steps.

For example you determine that you need to deploy a second instance of the **vks-01** application.

First let's look at the **vks-01** YAML file which we used to deploy the application using ArgoCD.

If you have closed the tab for gitea, open a new tab and log back in

Scaling The vks-01 Application (Contd.)

holuser / hol-2636-repo

Code Issues Pull Requests Actions Packages Projects Releases Wiki Activity

2 Commits 1 Branch 0 Tags

Search code...

Description
No description provided

Manage Topics

Readme 26 KiB

vks-01.yaml

README.md Initial commit 2 months ago

vks-01.yaml Uploaded vks-01.yaml 2 months ago

README.md vks-01.yaml

hol-2636-repo

In gitea, in the previously opened repo **hol-2636-repo** there is deployment file for the application named **vks-01**. This app could just as likely be called something like **account-update**. Maybe it is a containerized portion of a larger application for handling customer orders in a Web Store that sells T-Shirts. Maybe using microservices development methodologies.

In this case it is VKS CLuster deployment file. So it represents an infrastructure deployment. This file could be just one part of a much larger deployment, where the entire test stack is deployed. Everything from the infrastructure, consisting of VKS (Kubernetes), Traditional Virtual Machines, Applications, Databases, test suites, etc. are deployed, logged and torn down. Any single part of this scenario and more can be completely automated with VCF and ArgoCD. This "application" **vks-01** is a simple example of how this level of Continuous Delivery can be handled.

holuser / hol-2636-repo

Code Issues Pull Requests Actions Packages Projects Releases Wiki Activity Settings

main · hol-2636-repo / vks-01.yaml

holuser · 503e4ad4b6 · Uploaded vks-01.yaml · 2 months ago

41 lines | 1.1 KiB | YAML

```
1 apiVersion: cluster.x-k8s.io/vibeta1
2 kind: Cluster
3 metadata:
4   name: vks-01
5   namespace: argocd
6 spec:
7   clusterNetwork:
8     pods:
9       cidrBlocks:
10      - 192.168.156.0/20
11   services:
12     cidrBlocks:
13      - 10.96.0.0/12
14   serviceDomain: cluster.local
15 topology:
16   class: builtin-generic-v3.3.0
17   version: v1.33.1---vmware.i-fips-vkr.2
18 variables:
19   - name: kubernetes
20     value:
21     certificateRotation:
22       enabled: true
23       renewalDaysBeforeExpiry: 90
24   - name: vmClass
25     value: best-effort-xsmall
26   - name: storageClass
27     value: cluster-wld01-01a-vs-san-storage-policy
28 controlPlane:
29   replicas: 1
30   metadata:
31   annotations:
32     run.tanzu.vmware.com/resolve-os-image: os-name=photon, content-library=cl-5e990aa753a07bc9e
```

Raw Permalink Blame History

1. Click on the **pencil** icon to edit the **vks-01** yaml.

Scaling The vks-01 Application (Contd.)

```
33 workers:
34   machineDeployments:
35     - class: node-pool
36       name: vks-01-nodepool-wa4n
37     replicas: 1
38   metadata:
39     annotations:
40       run.tanzu.vmware.com/resolve-os-image: os-name=photon, content-library=cl-5e990aa753a07bc9e
41
```

Commit Changes

Add an optional extended description...

Add a Signed-off-by trailer by the committer at the end of the commit log message.

Commit directly to the `main` branch.

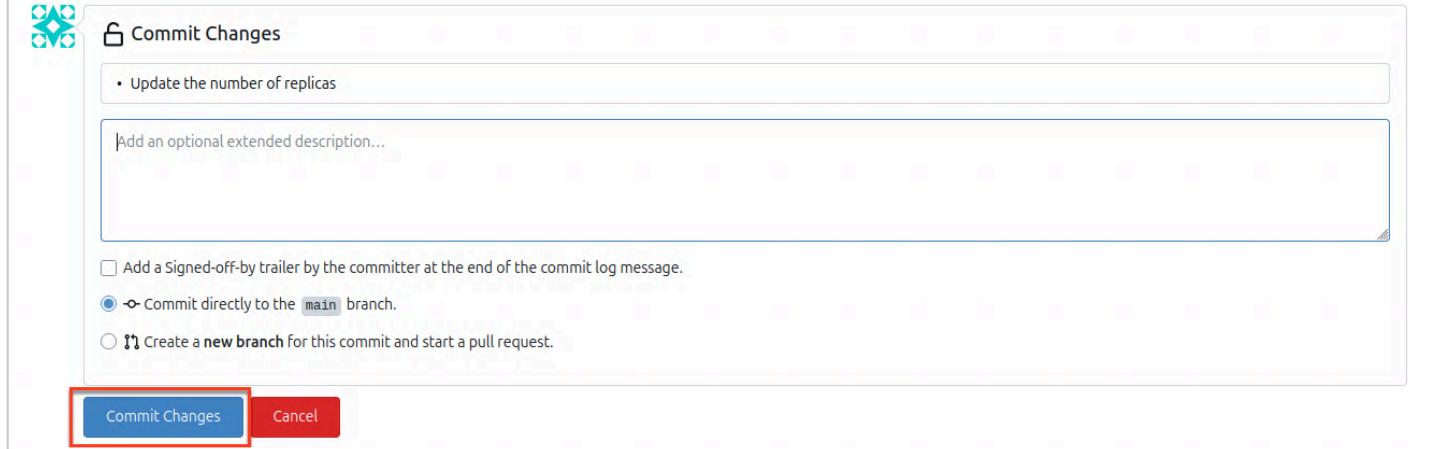
Create a `new branch` for this commit and start a pull request.

Commit Changes Cancel

1. Scroll down to the bottom of the **vks-01** yaml file until you see **workers**:
2. On the **replicas** line change the **1** to a **2**

Essentially as you might guess, we are going to deploy a new **vks-01 VKS** cluster to the **argocd** **vSphere Namespace**. A simple example of "*declarative end state configuration*". The desired state is a 2 worker node cluster in the namespace **argocd**. We **declare** this desire to the platform and it does it for us. More importantly, it will do its best to keep the system constantly based on the stated desired end state. We will demonstrate how the system responds when an "out of band" change is made a bit later.

```
33 workers:
34   machineDeployments:
35     - class: node-pool
36       name: vks-01-nodepool-wa4n
37       replicas: 2
38       metadata:
39         annotations:
40           run.tanzu.vmware.com/resolve-os-image: os-name=photon, content-library=cl-5e990aa753a07bc9e
41
```



After changing the 1 to a 2 scroll down again.

- Under Commit Changes type something in the top text box like “Update the number of replicas”
- Hit “Commit changes”

i Normally this would be done from a tool like VSCode, or some external system. For demo purposes it works fine.

Scaling The vks-01 Application (Contd.)

The screenshot shows the Gitea dashboard for the repository 'holuser/hol-2636-repo'. The sidebar on the left has a red box around the 'Issues' tab, with a black circle containing the number '1' above it. The main area displays a timeline of contributions over the last 12 months. The first contribution, made by 'holuser' at 1 minute ago with the commit hash '3622cafeas5' and the message 'Update the number of replicas', is highlighted with a red box and a black circle containing the number '2' above it. To the right, there is a sidebar titled 'Repository' showing the repository details.

1. The dashboard in gitea will show that the file was updated.
2. Return to the gitea Home Page To Review Changes by clicking the coffee cup.
3. You will see that the change you just made has been reflected here.

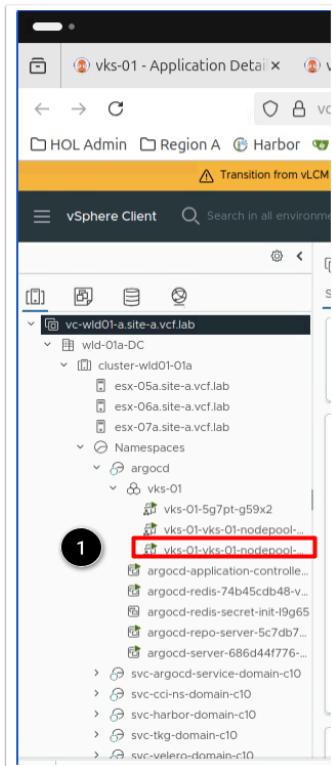
Return To vCenter

The screenshot shows the ArgoCD interface with the 'Files' tab selected. The breadcrumb navigation bar has a red box around the 'Region A' tab, with a black circle containing the number '1' above it. The main content area lists various Kubernetes resources, including 'vc-wld01-a Client', which is highlighted with a red box and a black circle containing the number '2' above it. The right side of the screen shows the YAML configuration for the selected file.

If you do not have the vCenter browser tab open:

1. Under the **Region A** tab
1. Select **vc-wld01-a Client** to open vCenter

Now ArgoCD is continuously synching with the gitea repo and will see the change to the vcf-01 file and upload it and re-run it. Which in turn results in another replica of the vcf-01 pod being deployed. If we open vCenter and look at the ArgoCD tab we'll notice that there is a new replica deployed.



You could scale vks-01 up or down using the same method.

For Example if you want to **play** a bit, wait for the vks cluster to finish deploying and in gitea change replica to 1 or 3. Don't go too crazy, as this is a lab with limited resources and you might just break something.

Module 6 - Using ArgoCD to Manage VMs

Module 6 - Using ArgoCD to Manage VMs

Declarative vs Imperative

A declarative end-state configuration is a model for system management that focuses on describing the **desired state** of a system, rather than the step-by-step instructions needed to achieve it. A "reconciliation engine" or "controller," is then responsible for analyzing the current state of the system and performing the necessary actions to converge it toward the declared end state. This approach is in direct contrast to an imperative configuration, which relies on a series of commands to detail **how-to** change the system state.

In this module we are going to step away from Kubernetes, microservices and modern apps, but continue to work with ArgoCD. You will use the VCF Consumption Platform to define and manage the **desired end-state** of, shall we say, a more Traditional Virtual Machine. As in the previous modules you will use the Local Consumption interface in vCenter to create the declarative **yaml** for a VM. You will then create a new repository the local git engine GitEA. You configure a new Application in ArgoCD. This new setup in GitEA and ArgoCD representing the tools gitOps teams will use to do their work. You will then use ArgoCD's continuous delivery in combination with the **VM Service** of the **vSphere Supervisor** to demonstrate how they work in cooperation to create components of the "reconciliation engine" or "controller"

Lastly you will see how this platform will work to enforce this "desired endstate" of the VM even when external efforts try to create undesired change of state.

Add Namespace-1 to ArgoCD Cluster Config

Before you can move on, you need to add the **vSphere Namespace** `namespace-1` to the ArgoCD Cluster Settings. ArgoCD was deployed and configured for you as part of the lab, but it was only configured to use a the **vSphere Namespace** it was deployed to. Your shiny new **vSphere Namespace** needs to be added before ArgoCD can manage deployments in it.

The steps are quite simple.

```
holuser@console:~$ argocd login 10.1.0.9
WARNING: server certificate had error: tls: failed to verify certificate: x509: cannot validate certificate for 10.1.0.9 because it doesn't contain any IP SANs. Proceed insecurely (y/n)? y
Username: admin
Password:
'admin:login' logged in successfully
Context '10.1.0.9' updated
holuser@console:~$ argocd cluster add supervisor --namespace argocd --namespace namespace-1 --upsert
WARNING: This will create a service account 'argocd-manager' on the cluster referenced by context 'supervisor' with full namespace level privileges. Do you want to continue [y/N]? y
supervisor cluster
WARNING: '--upsert' option will override the ArgoCD managed namespaces with the specified values. All previously configured namespaces will be removed.
Make sure to include all required namespaces. Do you want to continue [y/N]? y
--system-namespace is not supported when it is Supervisor Cluster, we will create serviceaccount argocd-manager in ArgoCD namespace argocd for kubernetes resource management
Start permission checking for managing namespace of Supervisor Cluster
INFO[0004] ServiceAccount "argocd-manager" already exists in namespace "argocd"
INFO[0004] RoleBinding "argocd/argocd-argocd-manager-role-binding" updated
INFO[0004] RoleBinding "namespace-1/argocd-argocd-manager-role-binding" created
Namespace argocd, namespace-1 from Cluster 'https://10.1.0.2:443' added
holuser@console:~$
```

As the ArgoCD admin you need to log in via the argocd cli. From a terminal window log into ArgoCD using the ArgoCD CLI

```
argocd login 10.1.0.9
```

Click to copy

Username: and **Password:**

```
admin - VMware123!
```

Click to copy

Then you run the command that will add, or in this case add the new **vSphere Namespace** to the **Cluster configuration** in the ArgoCD engine. The switch at the **-upsert** is what tells ArgoCD to add not replace.

```
argocd cluster add supervisor --namespace argocd --namespace namespace-1 --upsert
```

Click to copy

Answer **y** twice when you get asked. Once this is done, you can verify the change back in the ArgoCD UI --> Settings --> Clusters

The screenshot shows the Argo UI interface. On the left is a sidebar with icons for Applications, Settings (which is selected and highlighted with a red box), User Info, and Documentation. The main content area has a header with 'Settings / Clusters / https://10.1.0.2:443' and 'CLUSTERS' and 'Log out' buttons. Below the header is a 'C INVALIDATE CACHE' button. The main content is divided into two sections: 'GENERAL' and 'CONNECTION STATE'. The 'GENERAL' section contains fields for SERVER (https://10.1.0.2:443), CREDENTIALS TYPE (Token/Basic Auth), NAME (supervisor), NAMESPACES (argocd, namespace-1, highlighted with a red box), and ANNOTATIONS (argocd.ergoproj.io/refresh=2025-09-27T19:42:17Z). The 'CONNECTION STATE' section shows STATUS as Successful and VERSION as 1.30.

Add a new Storage policy to vSphere Namespace

OK, very sorry about the bit of house cleaning efforts, but it needed to be done. **Back to our regularly scheduled programming.**

The screenshot shows the vSphere Client interface. The left sidebar includes Home, Inventory (with Supervisor Management selected and highlighted with a red box), Policies and Profiles, Administration, and VMware Cloud Foundation Operations. The main content area shows a summary for 'namespace-1' with tabs for Summary (highlighted with a red box) and Actions. It displays VM Service (16 associated VM classes), Associated Content Libraries (1), Kubernetes Service (1 cluster, 2 libraries), and Zones (1 zone). The right side shows Pods (0), Storage (1 persistent volume claim, Storage Policies: k8s-storage-policy, VSAN Default Storage ...), and Capacity and Usage (CPU: 4.2 GHz used, Memory: 2.36 GB used, Storage: 95 GB used). A red box highlights the 'EDIT STORAGE' button under the Storage section.

This module's focus is on how you can bridge the management of traditional **vSphere Virtual Machines** with the management techniques used by the dev-ops or git-ops teams as they work on new "cloud native" applications, like those that have been refactored to microservices and run in containers. With that in mind the following steps should be very familiar.

1. Go back to the **vCenter client** for **wld01-a**
2. Click **Supervisor Management**
3. Click **namespace-1**

4. Click the **Summary** tab
5. Click **EDIT STORAGE** in the **Storage** tile

Add VM storage Policy

The screenshot shows a list of storage policies in a dialog box:

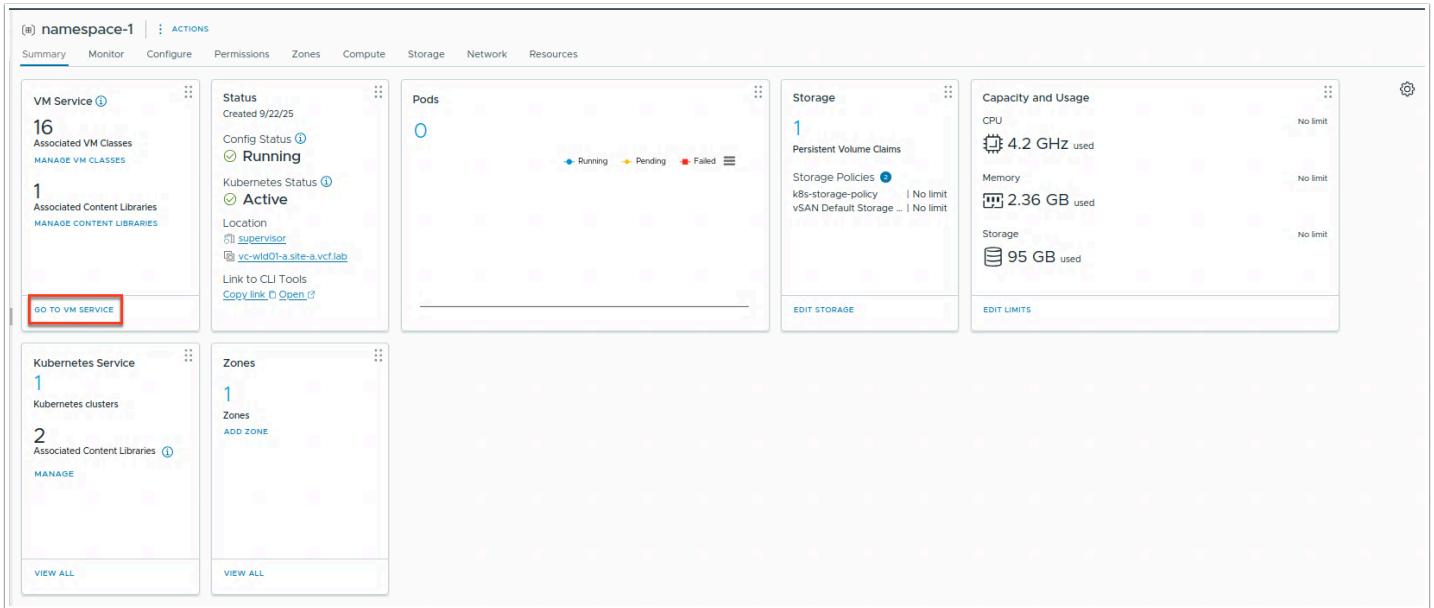
	Storage Policy	Total Capacity	Available Capacity
<input type="checkbox"/>	cluster-wld01-01a vSAN Storage Policy	2.64 TB	2.03 TB
<input type="checkbox"/>	Management Storage policy - Thin	2.64 TB	2.03 TB
<input type="checkbox"/>	VM Encryption Policy	2.64 TB	2.03 TB
<input type="checkbox"/>	Management Storage Policy - Regular	2.64 TB	2.03 TB
<input type="checkbox"/>	Management Storage policy - Encryption	2.64 TB	2.03 TB
<input checked="" type="checkbox"/>	k8s-storage-policy	2.64 TB	2.03 TB
<input type="checkbox"/>	Management Storage Policy - Single Node	2.64 TB	2.03 TB
<input checked="" type="checkbox"/>	vSAN Default Storage Policy	2.64 TB	2.03 TB
<input type="checkbox"/>	Management Storage Policy - Stretched Lite	2.64 TB	2.03 TB
<input type="checkbox"/>	k8s-harbor	2.64 TB	2.03 TB

At the bottom left, there is a checkbox labeled '2' and a 'Deselect All' button. At the bottom right, there are 'CANCEL' and 'OK' buttons, with the 'OK' button being highlighted by a red box.

In this example we will use the existing **vSan Default Storage Policy** to represent the desired storage policy for the Virtual machine. Most likely the storage handling needs for a VM will be different from the storage needs of an application running in containers.

1. Click the **Radio Check Box** for **vSan Deafault Storage Policy**
2. Leave the **Radio Check Box** for **k8s-storage-policy** checked
3. Click **OK**.

Close message in VM Service UI



The screenshot shows the 'namespace-1' summary page in the VM Service UI. The top navigation bar includes 'Summary', 'Monitor', 'Configure', 'Permissions', 'Zones', 'Compute', 'Storage', 'Network', and 'Resources'. The main area is divided into several tiles:

- VM Service**: Shows 16 Associated VM Classes and 1 Associated Content Library. A red box highlights the 'GO TO VM SERVICE' button.
- Status**: Created 9/22/25, Config Status Running, Kubernetes Status Active.
- Pods**: 0 pods listed.
- Storage**: 1 Persistent Volume Claims, Storage Policies (k8s-storage-policy, vSAN Default Storage), Edit Storage button.
- Capacity and Usage**: CPU (4.2 GHz used, No limit), Memory (2.36 GB used, No limit), Storage (95 GB used, No limit).
- Kubernetes Service**: 1 Kubernetes clusters and 2 Associated Content Libraries.
- Zones**: 1 Zone, Add Zone button.

1. Click **GO TO VM SERVICE** in the **VM SERVICE** tile



The screenshot shows the 'Getting Started with VM Service' modal. It contains the following text:

Getting Started with VM Service

Next steps:

- Customize your [VM Classes](#) to set guardrails for developers to self-service VMs.
- Create a content library [with templates](#).
- Add content libraries and VM classes to each Namespace for developers to self-service VMs.

At the bottom of the modal are two buttons: 'GOT IT' (highlighted with a red box) and 'Don't show me this again' (also highlighted with a red box). To the right of the buttons is a cartoon character holding balloons.

1. Close the **Getting Started with VM Service** tile if it is open by check the box for **Don't show me this again** and clicking **GOT IT**

Review VM Service Requirements

The screenshot shows the 'VM Service' section of the Supervisor Management interface. The 'VM Classes' tab is highlighted with a red box. Below it, the 'VM Class Summary' section displays statistics: 17 VM Classes, 2 Namespaces with VM Classes, and 7 VMs running VM Classes. Further down, the 'Content Libraries' section shows 1 Content Library added to Namespaces and 1 Namespace with added Content Libraries.

We can see here we already have 16 VM Classes defined the VM Services. This was done in earlier efforts when you added them to the VKS Service.

1. Click **VM Classes** tab

Continue Review

The screenshot shows the 'Available VM Classes' grid. It lists various VM class profiles with their resource requirements and reservation details. The 'best-effort-medium' class is highlighted with a red box. Other classes shown include best-effort-small, guaranteed-xsmall, guaranteed-medium, guaranteed-small, best-effort-xsmall, guaranteed-xlarge, best-effort-large, guaranteed-large, best-effort-xlarge, and best-effort-2xlarge.

VM Class	CPU	Memory	Notes
best-effort-medium	2 vCPUs No Reservation	8 GB No Reservation	(Namespaces 2) (VMs 8)
best-effort-small	2 vCPUs No Reservation	4 GB No Reservation	(Namespaces 2) (VMs 8)
guaranteed-xsmall	2 vCPUs 100% Reservation	2 GB 100% Reservation	(Namespaces 2) (VMs 8)
guaranteed-medium	2 vCPUs 100% Reservation	8 GB 100% Reservation	(Namespaces 2) (VMs 8)
guaranteed-small	2 vCPUs 100% Reservation	4 GB 100% Reservation	(Namespaces 2) (VMs 8)
best-effort-xsmall	2 vCPUs No Reservation	2 GB No Reservation	(Namespaces 2) (VMs 8)
guaranteed-xlarge	4 vCPUs 100% Reservation	32 GB 100% Reservation	(Namespaces 2) (VMs 8)
best-effort-large	4 vCPUs No Reservation	16 GB No Reservation	(Namespaces 2) (VMs 8)
guaranteed-large	4 vCPUs 100% Reservation	16 GB 100% Reservation	(Namespaces 2) (VMs 8)
best-effort-xlarge	4 vCPUs No Reservation	32 GB No Reservation	(Namespaces 2) (VMs 8)
best-effort-2xlarge	8 vCPUs No Reservation	64 GB No Reservation	(Namespaces 2) (VMs 8)

Here you can see the **VM Classes** that were added earlier. You can also create new classes from here.

1. Click the + above **CREATE VM CLASS**

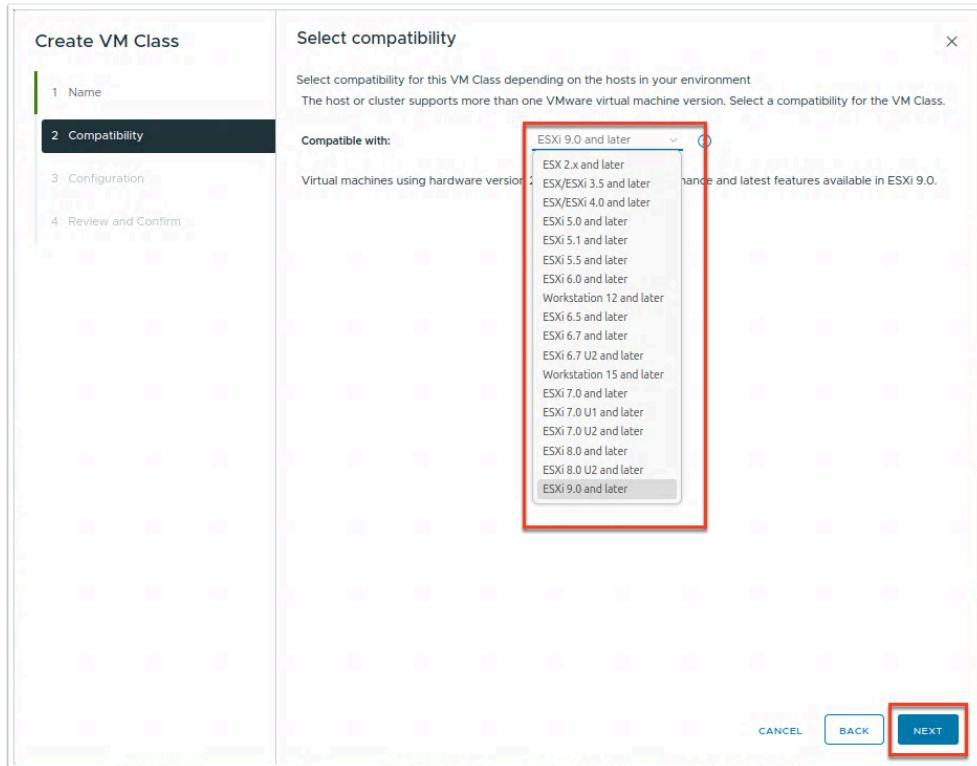
Create New VM Class - Name

The screenshot shows a 'Create VM Class' wizard with four steps: 1. Name (selected), 2. Compatibility, 3. Configuration, and 4. Review and Confirm. The 'Name' step is active, showing a 'Select a Name' dialog. The dialog has a title 'Select a Name', a subtitle 'Specify a VM Class name', and a note 'Fields marked with * are required'. A 'Name' field contains the value 'new-vm-class'. At the bottom of the dialog are 'CANCEL' and 'NEXT' buttons.

You give your new **VM Class** a name

1. Click **NEXT**

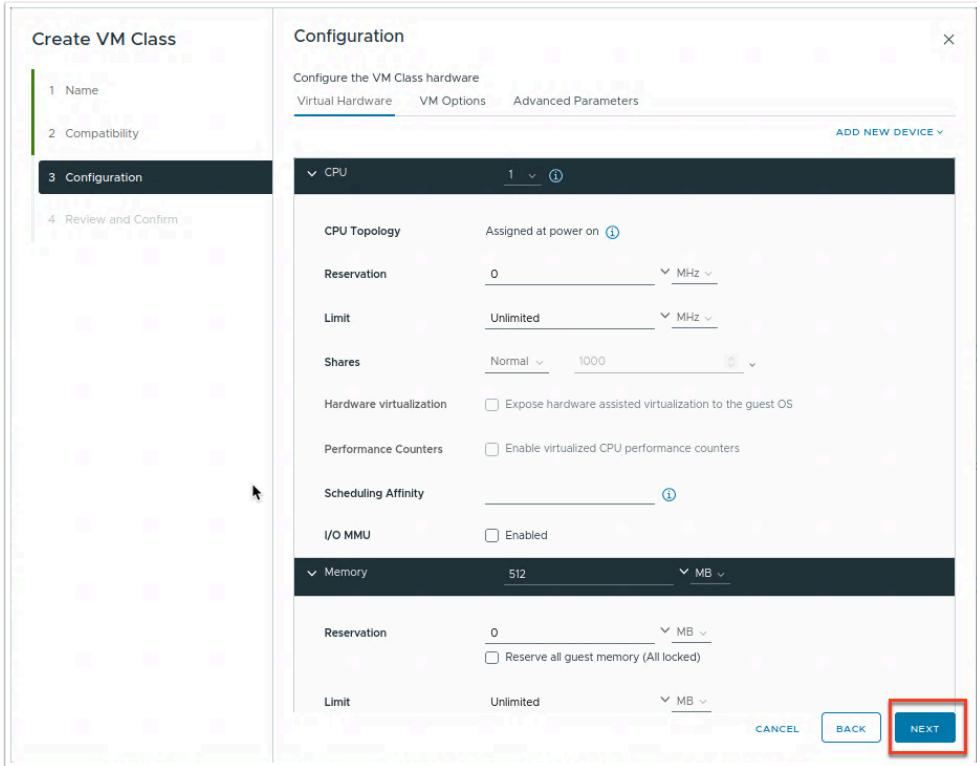
Create New VM Class - Compatibility



Choose the Hardware Version to use

1. Click **NEXT**

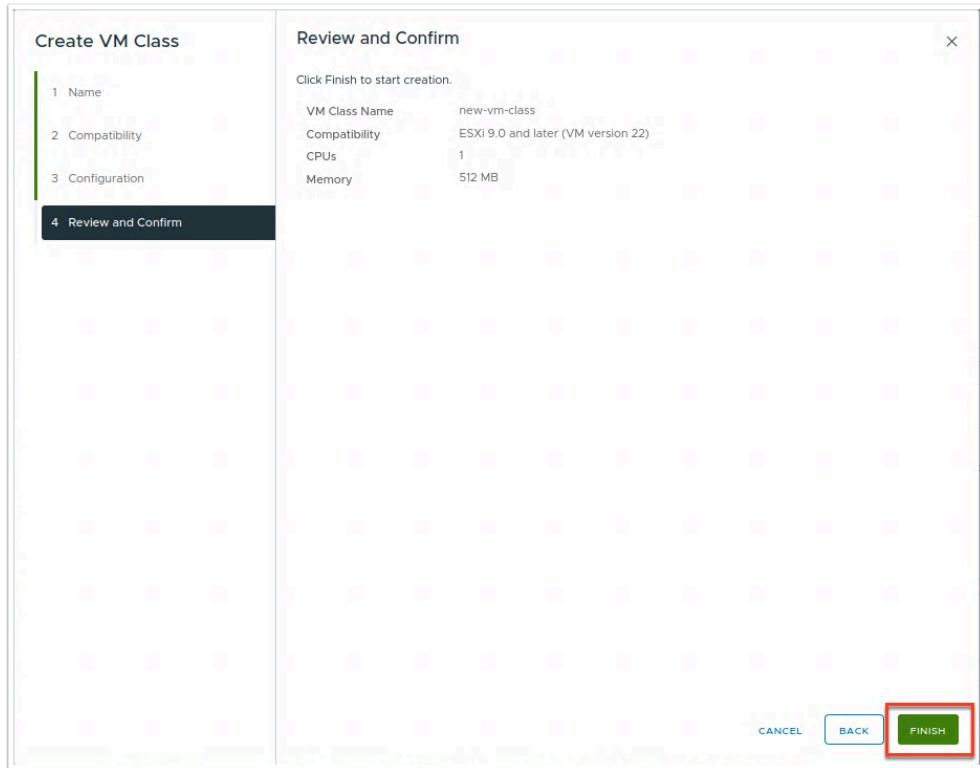
Create New VM Class - Configuration



Essentially here you have all that you expect to see under **VM Settings** in vCenter. You can click through the tabs to verify what you expect to see.

1. Click **NEXT**

Create New VM Class - Review and Confirm



Review and confirm your settings

1. Click **FINISH**

- i** We are going to use an existing **VM Class** so it is entirely up to you as to whether save this definition or not. Maybe use it do learn what can be defined and consider the possibilities.

Content Libraries

This screenshot shows the 'VM Service' section of the 'Content Libraries' tab in Supervisor Management. It displays a grid of available VM classes. A new class, 'new-vm-class', has been successfully created and is highlighted with a green message bar at the top right.

VM Class	CPU	Memory	Resource Type	Description
new-vm-class	1 vCPUs No Reservation	512 MB No Reservation	Namespaces (1) VMs (1)	
best-effort-medium	2 vCPUs No Reservation	8 GB No Reservation	Namespaces (1) VMs (1)	
best-effort-small	2 vCPUs No Reservation	4 GB No Reservation	Namespaces (1) VMs (1)	
guaranteed-xsmall	2 vCPUs 100% Reservation	2 GB 100% Reservation	Namespaces (1) VMs (1)	
guaranteed-medium	2 vCPUs 100% Reservation	8 GB 100% Reservation	Namespaces (1) VMs (1)	
guaranteed-small	2 vCPUs 100% Reservation	4 GB 100% Reservation	Namespaces (1) VMs (1)	
best-effort-xsmall	2 vCPUs No Reservation	2 GB No Reservation	Namespaces (1) VMs (1)	
guaranteed-xlarge	4 vCPUs 100% Reservation	32 GB 100% Reservation	Namespaces (1) VMs (1)	
best-effort-large	4 vCPUs No Reservation	16 GB No Reservation	Namespaces (1) VMs (1)	
guaranteed-large	4 vCPUs 100% Reservation	16 GB 100% Reservation	Namespaces (1) VMs (1)	
best-effort-xlarge	4 vCPUs No Reservation	32 GB No Reservation	Namespaces (1) VMs (1)	

1. Click CONTENT LIBRARIES

This screenshot shows the 'VM Service' section of the 'Content Libraries' tab in Supervisor Management. It displays a list of existing content libraries. One library, 'vm-images', is highlighted with a red box and a red arrow points to its '+ CREATE CONTENT LIBRARY' button.

Content Library	Templates	Namespaces
vm-images	1	1

This **VM Service** already has a **Content Library** assigned to it from **Module 2** so you will not need to add one. However if you needed to create one you can start it from here by clicking the + sign in the **CREATE CONTENT LIBRARY** tile.

Since you have what you need:

2. Click the **Namespaces** tab

Back to namespace-1 - Resources

The screenshot shows the 'Supervisor Management' interface with the 'Namespaces' tab selected. A table lists various namespaces, each with a status indicator, CPU and memory usage, storage usage, and a description. One row, '(#) namespace-1', has its entire row highlighted with a red box, and the 'Namespace' column header is also highlighted.

Namespaces	Supervisor	Config Status	CPU (Used Limit)	Memory (Used Limit)	Storage (Used Limit)	Description	vCenter
(#) argoocd	supervisor	Running	1.837 GHz No Limit	2.35 GB No Limit	49 GB No Limit	yc-wld01-a-site-a.vcf.lab	
(#) namespace-1	supervisor	Running	4.2 GHz No Limit	2.36 GB No Limit	95 GB No Limit	yc-wld01-a-site-a.vcf.lab	
(#) svc-argocd-service-domain-c10	supervisor	Running	21 MHz No Limit	374 MB No Limit	0 No Limit	yc-wld01-a-site-a.vcf.lab	
(#) svc-cci-ns-domain-c10	supervisor	Running	210 MHz No Limit	567 MB No Limit	0 No Limit	yc-wld01-a-site-a.vcf.lab	
(#) svc-harbor-domain-c10	supervisor	Running	420 MHz No Limit	1.22 GB No Limit	18 GB No Limit	yc-wld01-a-site-a.vcf.lab	
(#) svc-tkg-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit	yc-wld01-a-site-a.vcf.lab	
(#) svc-velero-domain-c10	supervisor	Running	0 No Limit	0 No Limit	0 No Limit	yc-wld01-a-site-a.vcf.lab	

1. Click **namespace-1**

The screenshot shows the 'Resources' tab for the 'namespace-1' page. It displays various resource management sections: VM Service, Storage, Capacity and Usage, Kubernetes Service, and Zones. The 'Resources' tab itself is highlighted with a red box.

VM Service
16 Associated VM Classes
MANAGE VM CLASSES

Storage
1 Persistent Volume Claims
Storage Policies: k8s-storage-policy | No limit, vSAN Default Storage... | No limit
EDIT STORAGE

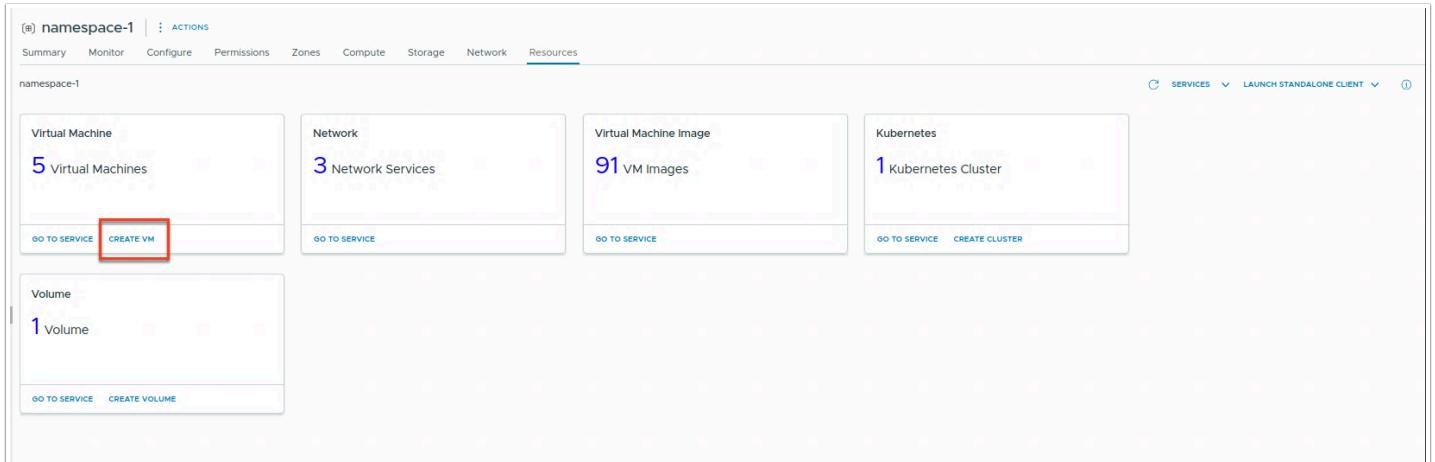
Capacity and Usage
CPU: 4.2 GHz used | No limit
Memory: 2.36 GB used | No limit
Storage: 95 GB used | No limit
EDIT LIMITS

Kubernetes Service
1 Kubernetes clusters
2 Associated Content Libraries
MANAGE

Zones
1 Zones
ADD ZONE
VIEW ALL

1. Click **Resources** tab

Create a basic Virtual Machine declaratively

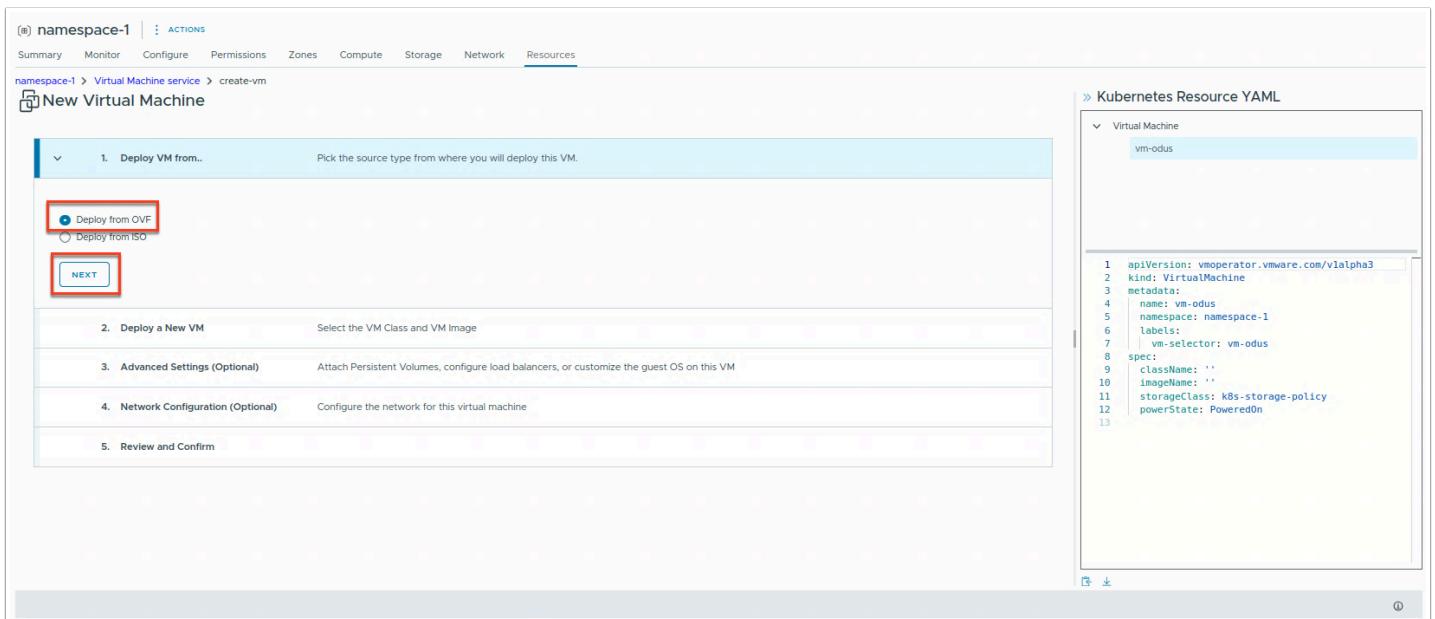


The screenshot shows the LCI interface for a namespace named 'namespace-1'. The top navigation bar includes 'Summary', 'Monitor', 'Configure', 'Permissions', 'Zones', 'Compute', 'Storage', 'Network', and 'Resources'. The 'Resources' tab is selected. Below the tabs, there are four main sections: 'Virtual Machine' (5 Virtual Machines), 'Network' (3 Network Services), 'Virtual Machine Image' (91 VM Images), and 'Kubernetes' (1 Kubernetes Cluster). In the 'Virtual Machine' section, there is a 'CREATE VM' button which is highlighted with a red box. Another red box highlights the 'CREATE VOLUME' button in the 'Volume' section.

Back in Local Consumption Interface the goal is to create a Virtual Machine deployment yaml

1. Click **CREATE VM**

New Virtual Machine



The screenshot shows the 'New Virtual Machine' wizard in the LCI. The first step, '1. Deploy VM from...', has 'Deploy from OVF' selected and the 'NEXT' button highlighted with a red box. The right side of the screen displays the generated Kubernetes Resource YAML:

```
1  apiVersion: vmoperator.vmware.com/v1alpha3
2  kind: VirtualMachine
3  metadata:
4    name: vm-odus
5    namespace: namespace-1
6    labels:
7      vm-selector: vm-odus
8  spec:
9    className: ''
10   imageName: ''
11   storageClass: k8s-storage-policy
12   powerState: PoweredOn
13
```

Once again we see in the LCI the UI Workflow to creating a VM on the left two thirds and the yaml being created dynamically on the right.

1. Leave **Deploy VM from OVF** selected
2. Click **NEXT**

Deploy a VM

The screenshot shows the VMware Cloud Foundation interface for deploying a new virtual machine. The process is divided into two steps: 1. Deploy VM from.. and 2. Deploy a New VM. Step 2 is currently active, showing the configuration for a new VM named "managed-vm". The VM is set to run in zone "z-wld-a". A table lists available VM images, with "ubuntu-22.04" selected. On the right, the "Kubernetes Resource YAML" pane displays the corresponding YAML configuration:

```
apiVersion: vmoperator.vmware.com/v1alpha1
kind: VirtualMachine
metadata:
  name: break-net
  namespace: namespace-1
  labels:
    vm-selector: break-net
    topology.kubernetes.io/zone: z-wld-a
spec:
  className: best-effort-xsmall
  # Friendly image name: ubuntu
  imageName: vmi-323ad24528727ec58
  osType: ubuntu64Guest
  libraryName: vm-images
  createdOn: 2025-09-25T12:37:02Z
```

1. Set the **VM Name** as `managed-vm`
2. Click the dropdown for **Zone**: select `z-wld-a`
3. Select the `ubuntu-22.04` VM Image

i All the settings, and more, that are being set now can be made into variables by the pipeline project at deploy time. We are simply creating the basic yaml deployment file.

Deploy a VM (contd)

Kubernetes Resource YAML

```

1 apiVersion: vmoperator.vmware.com/v1alpha3
2 kind: VirtualMachine
3 metadata:
4   name: vm-odus
5   namespace: namespace-1
6   labels:
7     vm-selector: vm-odus
8 spec:
9   className: best-effort-xsmall
10  # Friendly image name: ubuntu-22.04
11  imageName: vml-323ad24528727ec58
12  storageClass: vsan-default-storage-policy
13  powerState: PoweredOn
14

```

Scroll down in the workflow window to **VM Class**

1. Select the **best-effort-xsmall** **VM Class**.
2. Select the **vsan-default-storage-policy** from **Storage Class**
3. Leave the slider as **Powered On** for **Power State**
4. Click **NEXT**

Advanced Setting (optional)

Kubernetes Resource YAML

```

1 apiVersion: vmoperator.vmware.com/v1alpha3
2 kind: VirtualMachine
3 metadata:
4   name: vm-xhnr
5   namespace: namespace-1
6   labels:
7     vm-selector: vm-xhnr
8 spec:
9   className: best-effort-xsmall
10  # Friendly image name: ubuntu-22.04
11  imageName: vml-323ad24528727ec58
12  storageClass: vsan-default-storage-policy
13  powerState: PoweredOn
14  bootstrap:
15    cloudInit:
16      cloudConfig:
17        timezone: America/Los_Angeles
18

```

1. Make sure the **Cloud-Init** Radio button is selected
2. Enter **America/Los_Angeles**
3. Leave **Enable Default User** and **SSH Password Authentication** as they are
4. Click **CREATE NEW USER**

Create New User

Username * vmware

Password * VMware123!

Confirm Password * VMware123!

Default Sudo ⓘ Enable

Sudo ⓘ ALL=(ALL) NOPASSWD:ALL

SSH Authorized Keys ⓘ Add SSH Keys ADD CANCEL SAVE

1. Set **Username:** `vmware`
2. Set **Password:** `VMware123!`
3. Set **Deafult Sudo:** `Enable`
4. Click **SAVE**

Kubernetes Resource YAML

```

1  apiVersion: vmoperator.vmware.com/v1alpha3
2  kind: VirtualMachine
3  metadata:
4    name: vm-3uvp
5    namespace: namespace-1
6    labels:
7      vm-selector: vm-3uvp
8  spec:
9    className: best-effort-xsmall
10   # Friendly image name: ubuntu-22.04
11   imageName: vml-323ad24528727ec58
12   storageClass: vsan-default-storage-policy
13   powerState: PoweredOn
14   bootstrap:
15     cloudInit:
16       cloudConfig:
17         timezone: America/Los_Angeles
18         users:
19           - name: vmware
20             passwd:
21               name: vm-3uvp-bootstrap-secret
22               key: vmware-passwd
23             lock_passwd: false
24             sudo: ALL=(ALL) NOPASSWD:ALL
25

```

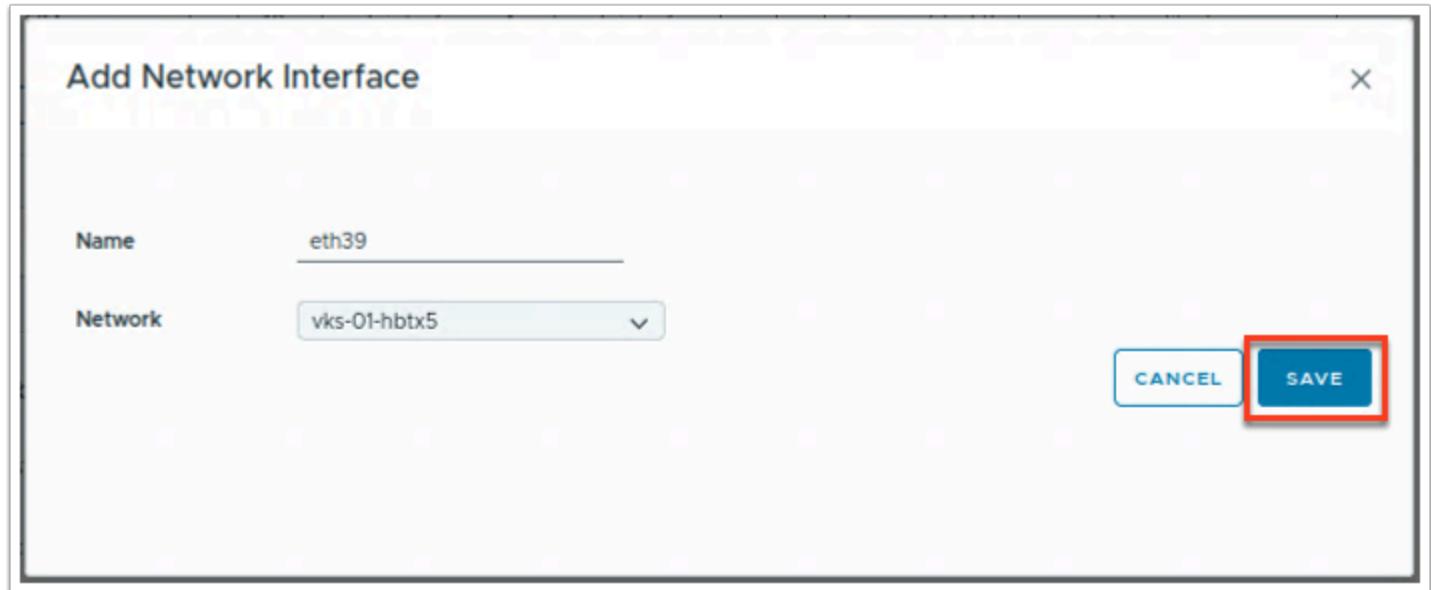
Network Configuration (Optional)

Kubernetes Resource YAML

```

1  apiVersion: vmoperator.vmware.com/v1alpha3
2  kind: VirtualMachine
3  metadata:
4    name: vm-3uvp
5    namespace: namespace-1
6    labels:
7      vm-selector: vm-3uvp
8  spec:
9    className: best-effort-xsmall
10   # Friendly image name: ubuntu-22.04
11   imageName: vml-323ad24528727ec58
12   storageClass: vsan-default-storage-policy
13   powerState: PoweredOn
14   networkInterface:
15     hostName: my-vm
16     domainName: vcf.lab
17     nameservers:
18       - 10.1.10.129
19     searchDomains:
20       - site-a.vcf.lab
21   bootstrap:
22     cloudInit:
23       cloudConfig:
24         timezone: America/Los_Angeles
25         users:
26           - name: vmware
27             passwd:

```



This lab does not have external internet access but we can do some simple network tests. Make the following configuration changes

1. Click + ADD NETWORK INTERFACE
2. Take the defaults and click **SAVE**
3. For **HostName**: - you can name it anything you want.
4. For **Domain name**: **vcf.lab**
5. For **DNS Settings** - > **Nameservers**: enter **10.1.10.129** - click **ADD**
6. For **Search Domains**: enter **site-a.vcf.lab** - click **ADD**
7. Scroll down and click **NEXT**

Download Deployment yaml

```

apiVersion: vmoperator.vmware.com/v1alpha3
kind: VirtualMachine
metadata:
  name: vm-odus
  namespace: namespace-1
  labels:
    vm-selector: vm-odus
spec:
  className: best-effort-xsmall
  # Friendly image name: ubuntu-22.04
  imageName: vml-323ad24528727ec58
  storageClass: vsan-default-storage-policy
  powerState: PoweredOn
  network:
    hostName: my-vn
    domainName: vcf.lab
    nameservers:
      - 10.1.10.129
    searchDomains:
      - site-a.vcf.lab

```

At this point you could click **DEPLOY VM**. First though click the **Download** button for the YAML that has been created from the UI Workflow.

1. Click **Download All YAMls to zip**
2. Click **DEPLOY VM**



We will use this zip file later.

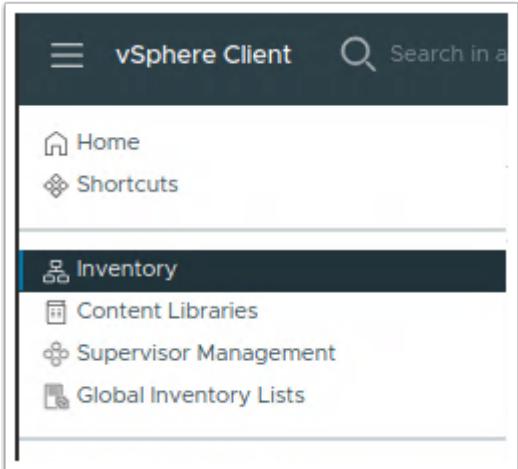
New VM Deploys

A screenshot of the VCF UI showing the "Virtual Machine Service" page. The page header includes tabs for Summary, Monitor, Configure, Permissions, Zones, Compute, Storage, Network, and Resources. The Resources tab is selected. The main content area displays a table of deployed VMs. The table columns are: Name, Status, Power State, Zone, Address, VM Image, VM Class, and Age. The table lists six VMs:

Name	Status	Power State	Zone	Address	VM Image	VM Class	Age
vm-odus	? Unknown	--	--	--	ubuntu-22.04	best-effort-xsmall	2 seconds
vks-01-vks-01-nodepool-ohd4-jchss-q2kqg-6p6xl	Ready	Powered On	z-wld-a	172.16.200.38	ob-24790750-photon-5-amd64-v1.33.1--vmware-1-fips-vkr.2	best-effort-xsmall	3 days
vks-01-vks-01-nodepool-ohd4-jchss-q2kqg-6xrk	Ready	Powered On	z-wld-a	172.16.200.37	ob-24790750-photon-5-amd64-v1.33.1--vmware-1-fips-vkr.2	best-effort-xsmall	3 days
vks-01-vks-01-np-jox6-mb8b7-45qbo-mcfos	Ready	Powered On	z-wld-a	172.16.200.36	ob-24790750-photon-5-amd64-v1.33.1--vmware-1-fips-vkr.2	guaranteed-xsmall	3 days
vks-01-v2jcc-vq1d	Ready	Powered On	z-wld-a	172.16.200.35	ob-24790750-photon-5-amd64-v1.33.1--vmware-1-fips-vkr.2	best-effort-xsmall	3 days
cli-vm	Ready	Powered On	z-wld-a	172.16.200.3	ubuntu-22.04	best-effort-small	3 days

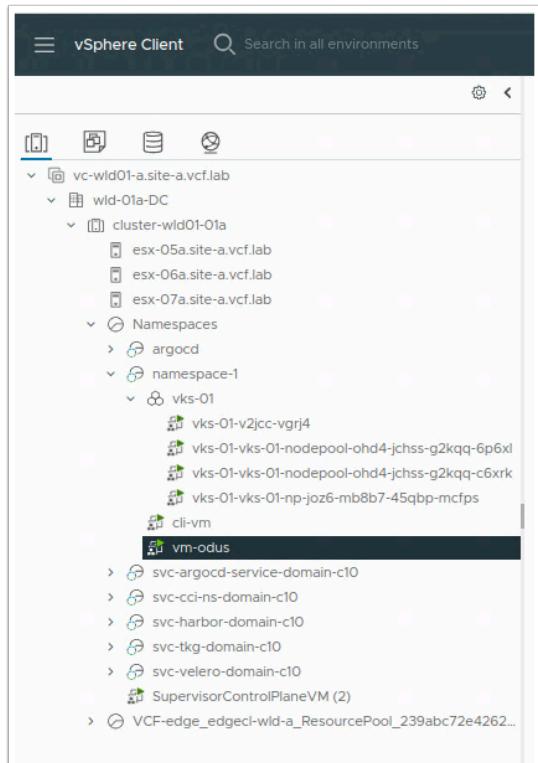
You should see your new VM show up in the Virtual Machines table.

vCenter Inventory



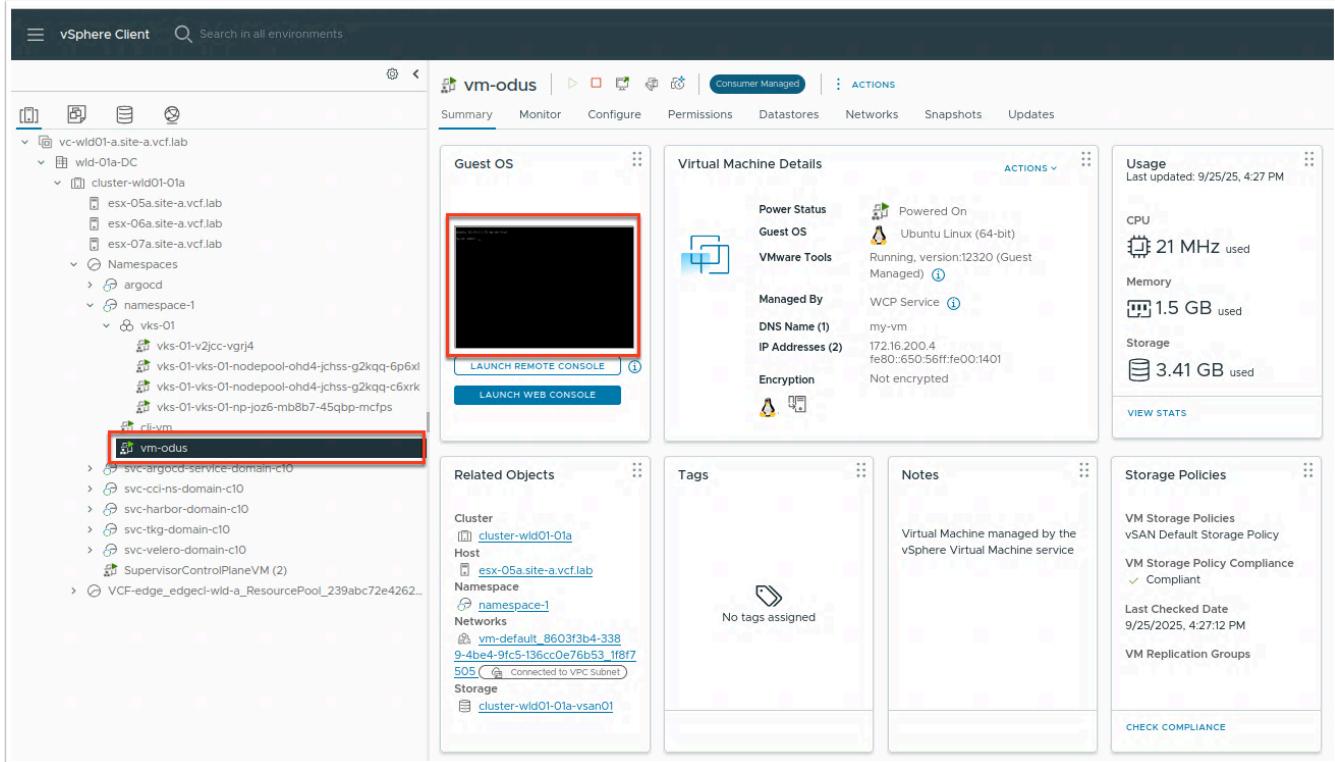
Open the console and log into the new machine.

1. Go to the **Menu** and open **Inventory**



Expand the **vCenter --> vDC --> Cluster --> Namespaces --> namespace-1**. And you will see your new VM coming online.

Open VM Console

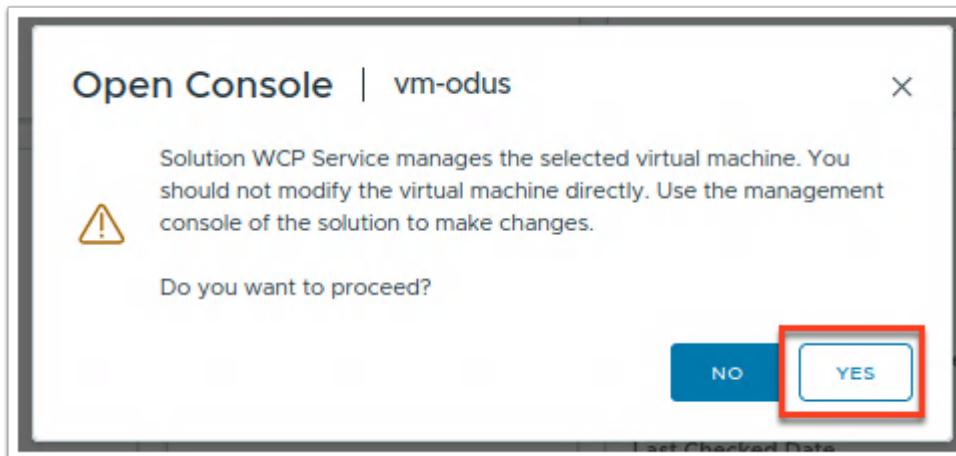


The screenshot shows the vSphere Client interface with the following details:

- Left Sidebar:** Shows the navigation tree with categories like vc-wld01-a.site-a.vcf.lab, wld-01a-DC, cluster-wld01-01a, Namespaces, argoCD, namespace-1, vks-01, and vcf-vm. The item "vm-odus" is highlighted with a red box.
- VM Details Panel:** Displays the following information:
 - Guest OS:** Ubuntu Linux (64-bit), Running, version:12320 (Guest Managed)
 - Power Status:** Powered On
 - VMware Tools:** Version 12.3.20 (Guest Managed)
 - Managed By:** WCP Service
 - DNS Name:** my-vm
 - IP Addresses:** 172.16.200.4, fe80::650:56ff:fe00:1401
 - Encryption:** Not encrypted
- Right Side Panels:** Show Usage (Last updated: 9/25/25, 4:27 PM), CPU (21 MHz used), Memory (1.5 GB used), Storage (3.41 GB used), and View Stats.

Once the VM is deployed and boots.

Click the **Console**



Answer Yes

```
Ubuntu 22.04.2 LTS my-vm tty1
```

```
Hint: Num Lock on
```

```
my-vm login: vmware
```

```
Password: _
```

1. Login: **vmware**
2. Password: **VMware123!**

```
* Introducing Expanded Security Maintenance for Applications.  
Receive updates to over 25,000 software packages with your  
Ubuntu Pro subscription. Free for personal use.
```

```
https://ubuntu.com/pro
```

```
Expanded Security Maintenance for Applications is not enabled.
```

```
0 updates can be applied immediately.
```

```
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status
```

```
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update
```

```
New release '24.04.3 LTS' available.
```

```
Run 'do-release-upgrade' to upgrade to it.
```

```
Last login: Fri Sep 26 15:42:34 PDT 2025 on tty1  
$
```

And you are in!

Open Gitea



You have deployed a **Virtual Machine** using the **vSphere Supervisor VM Service**. During the process of creating the **VM** you entered a series of values into the **UI** and the **LCI** created the **yaml** that represents your declared **endstate configuration**. You are now going to take that yaml and put into a repository in GitEA, then connect that repo to **ArgCD** and direct **ArgoCD** to maintain your desired end state. In cooperation with the **vSphere Supervisor** you will be creating an example of a "**reconciliation engine**" we spoke about earlier.

1. In a new browser tab open the **Gitea** app and **Sign In**

Sign In

Username or Email Address *

Forgot password?

Password

Remember This Device

Sign In

or

 Sign in with OpenID

[Sign in with a passkey](#)
[Need an account? Register now.](#)

As a reminder

1. Login with : Username

holuser

Click to copy

2. Password:

VMware123!

Click to copy

3. Click **Sign In**

Create a new Repo

The screenshot shows a user profile 'holuser' with a recent activity feed. The timeline highlights four contributions in the last 12 months. The search bar on the right shows 'holuser/hol-2636-repo'.

1. Click the + to create a new repo

The form allows creating a new repository. The owner is set to 'holuser'. The repository name is 'deploy-vm'. A note suggests using short, memorable, and unique names.

New Repository

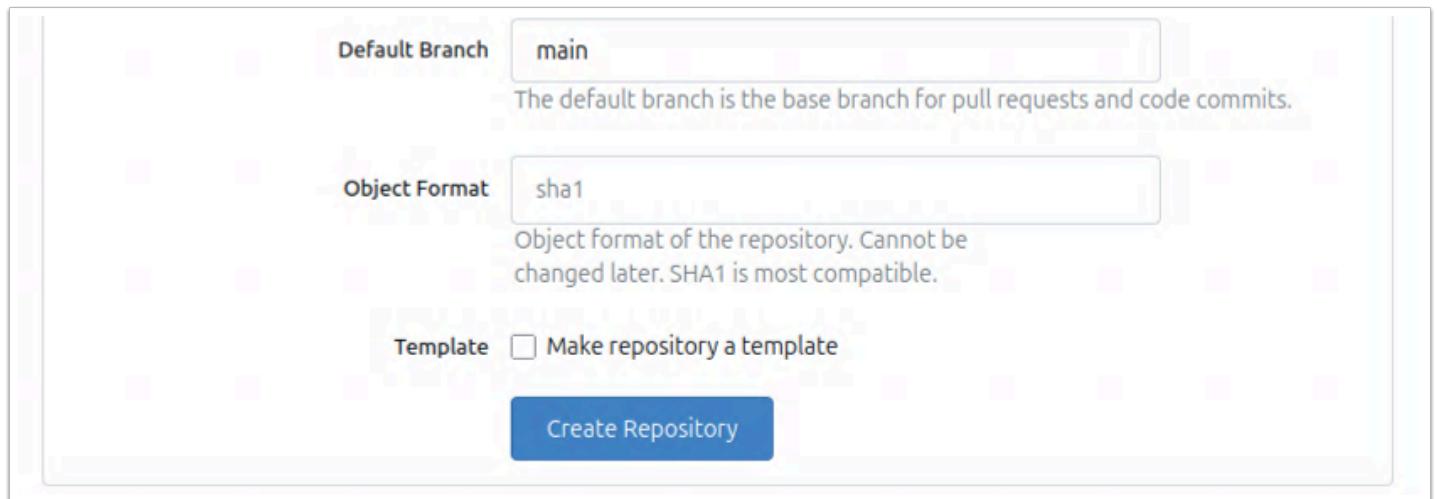
A repository contains all project files, including revision history. Already hosting one elsewhere? [Migrate repository](#).

Owner * holuser

Some organizations may not show up in the dropdown due to a maximum repository count limit.

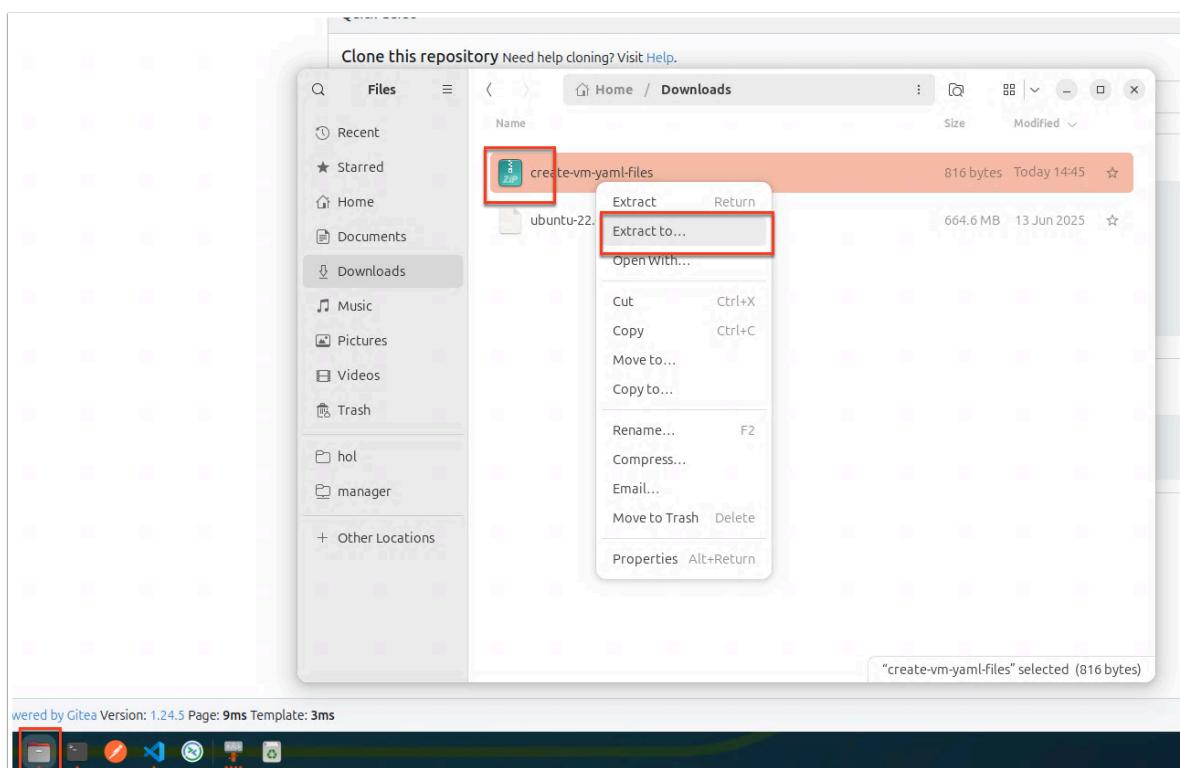
Repository Name * deploy-vm

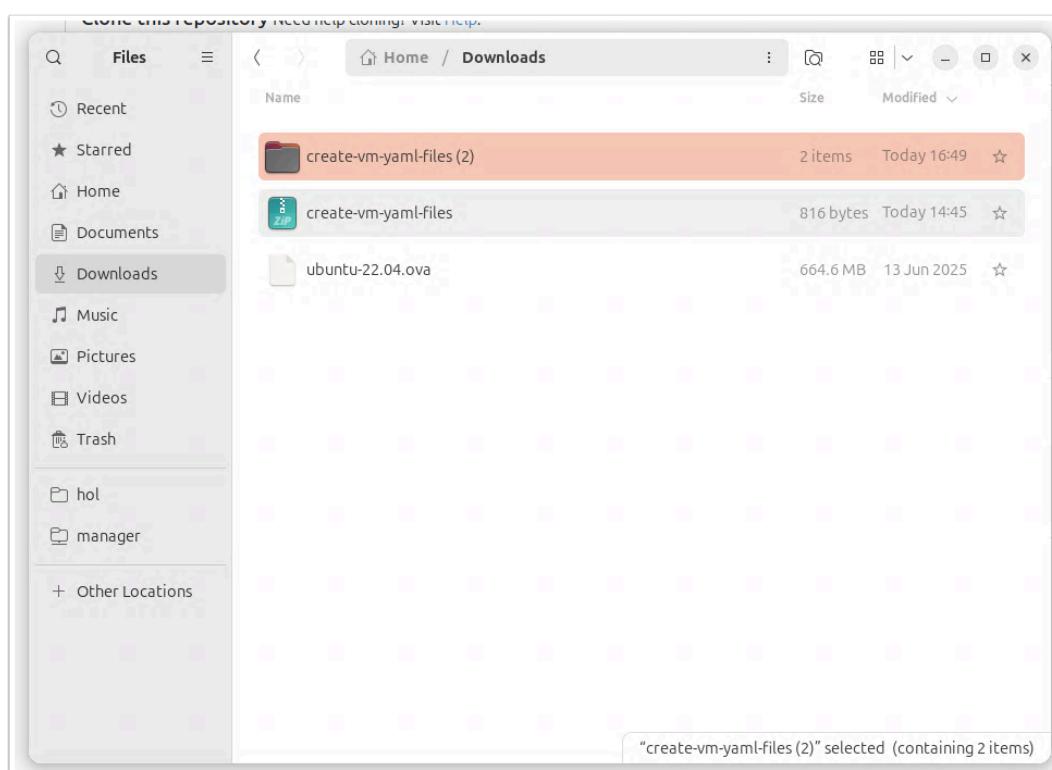
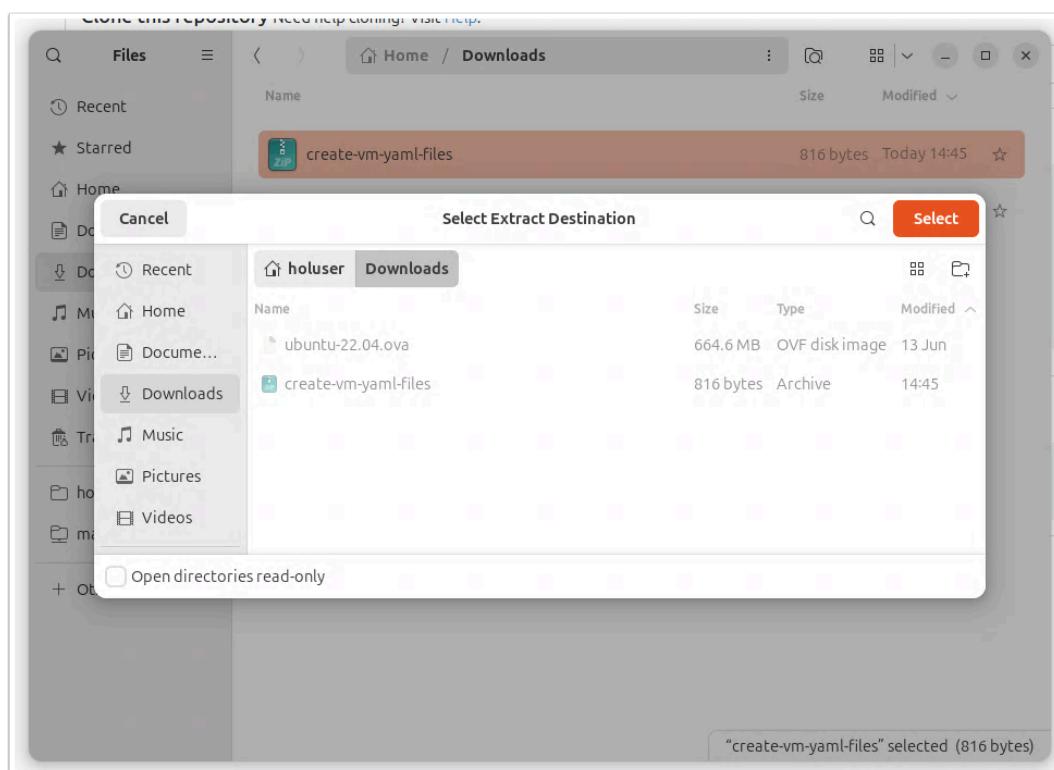
Good repository names use short, memorable and unique keywords. A repository named ".profile" or ".profile-private" could

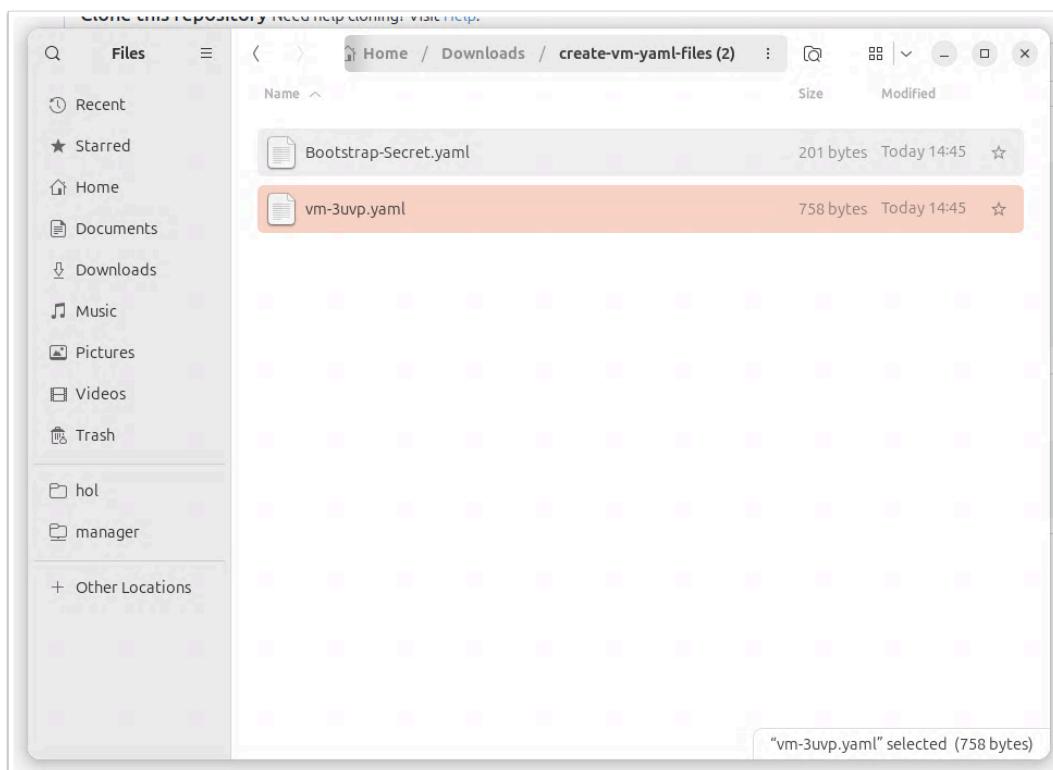


1. Name the repo `deploy-vm`
2. Click **Create Repository**

Unzip yaml so you can upload to your new repo



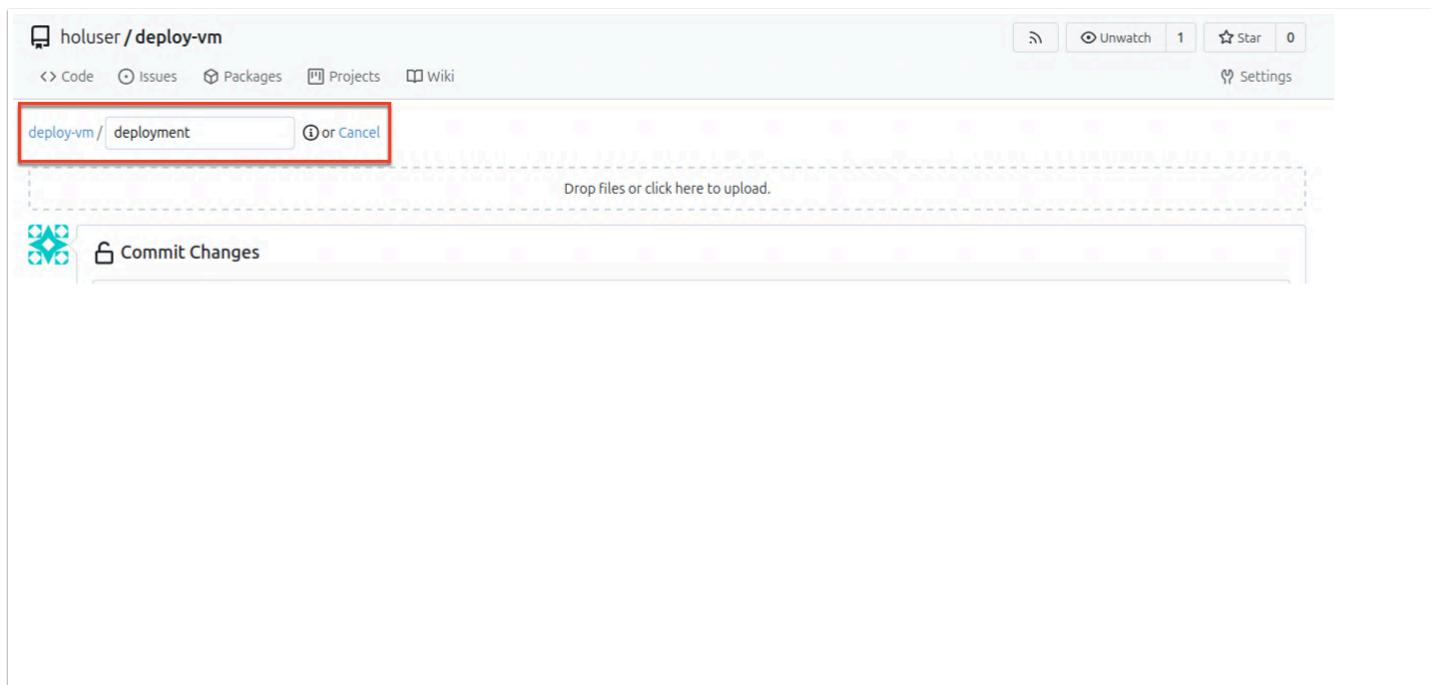




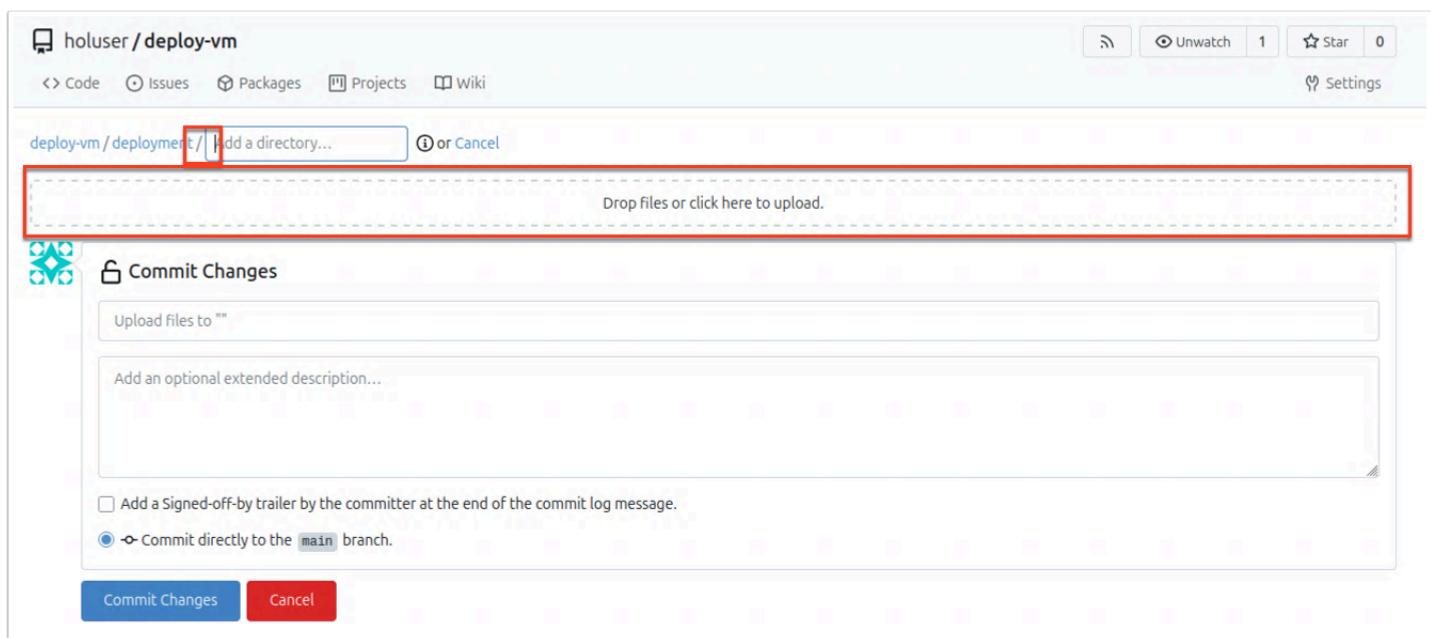
Upload yaml to new repo

A screenshot of a GitHub repository page for 'holuser / deploy-vm'. The top navigation bar includes links for 'Code', 'Issues', 'Packages', 'Projects', and 'Wiki'. On the right, there are buttons for 'Unwatch' (1), 'Star' (0), and 'Settings'. Below the navigation, a 'Quick Guide' section and a 'Clone this repository' link are visible. At the bottom of the page, there is a navigation bar with 'New File' and 'Upload File' buttons, an 'HTTP' link, an 'SSH' link, and a URL 'http://localhost:3000/holuser/deploy-vm.git'. The 'Upload File' button is highlighted with a red box.

Click **Upload File**

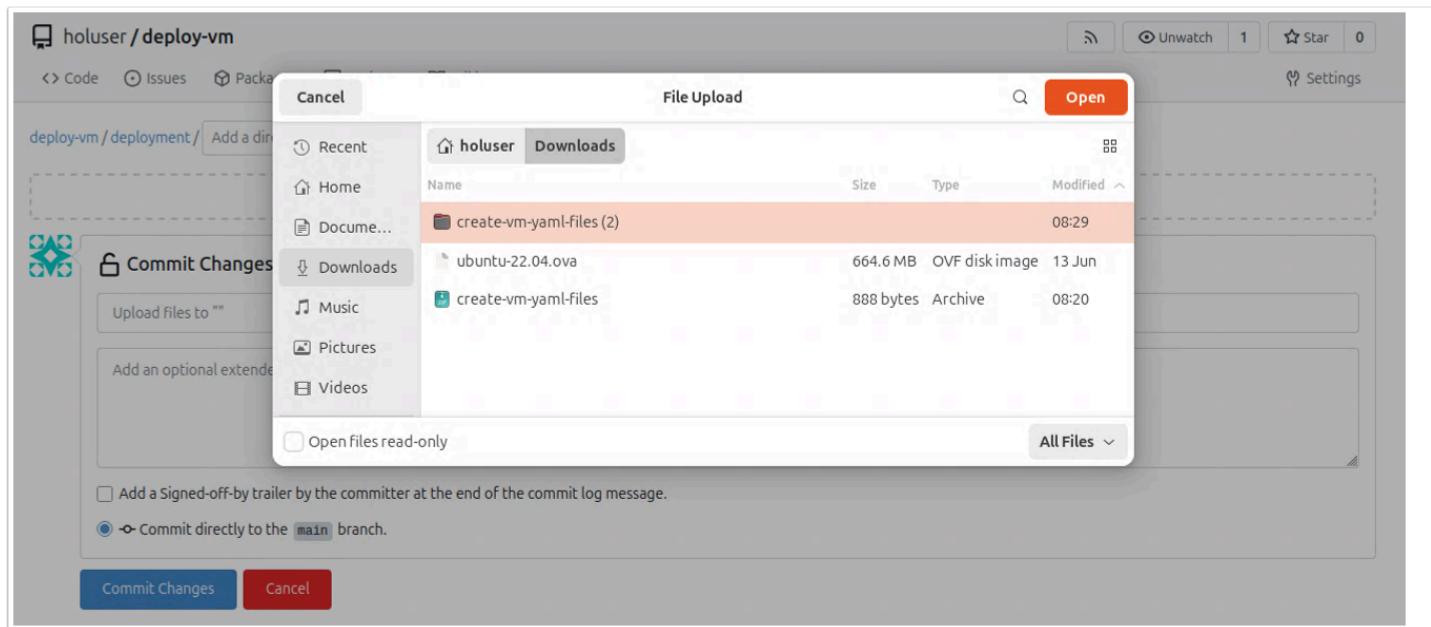


1. In the form at the top type **deployment** - this will be the folder name
2. Then add a **/** after the name



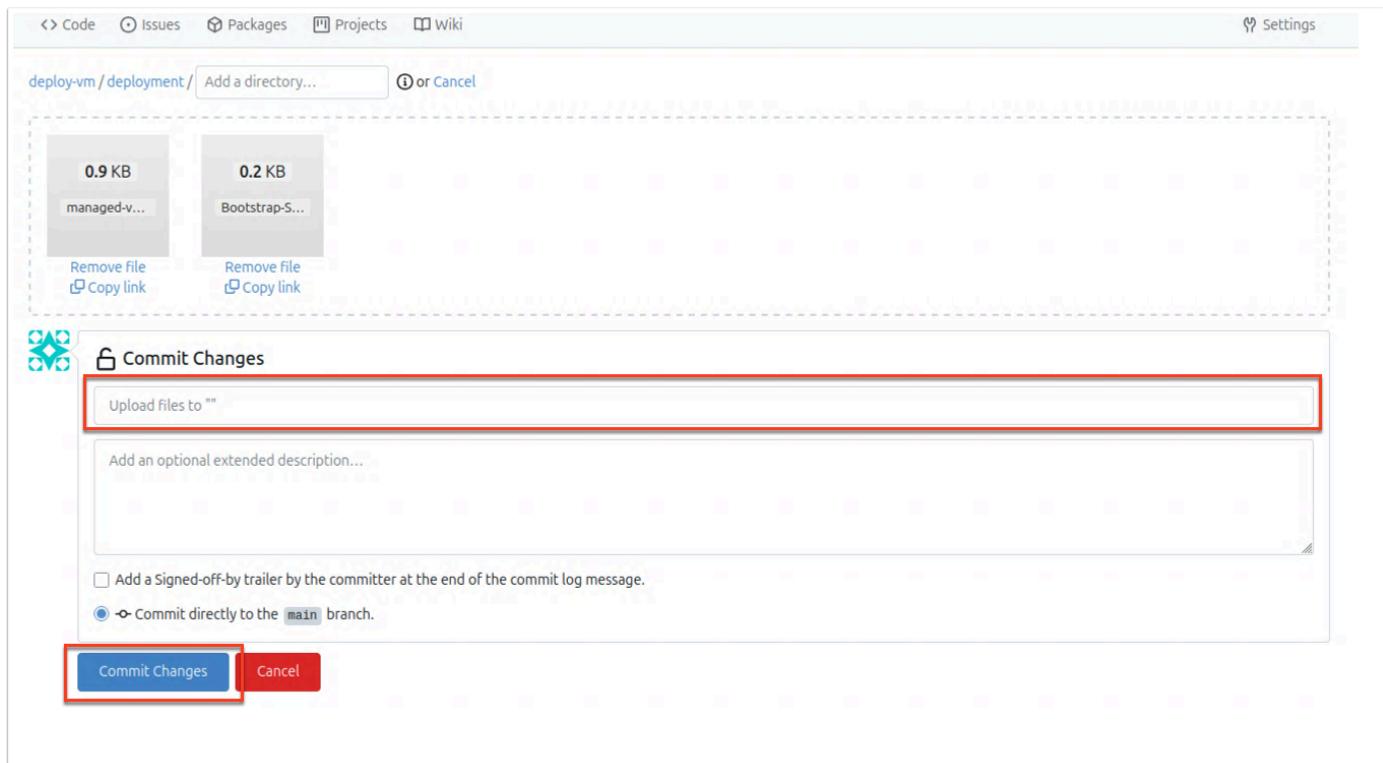
You will see that **vm-deploy** has become a directory.

1. Click into the **Drop files or click here to upload** form



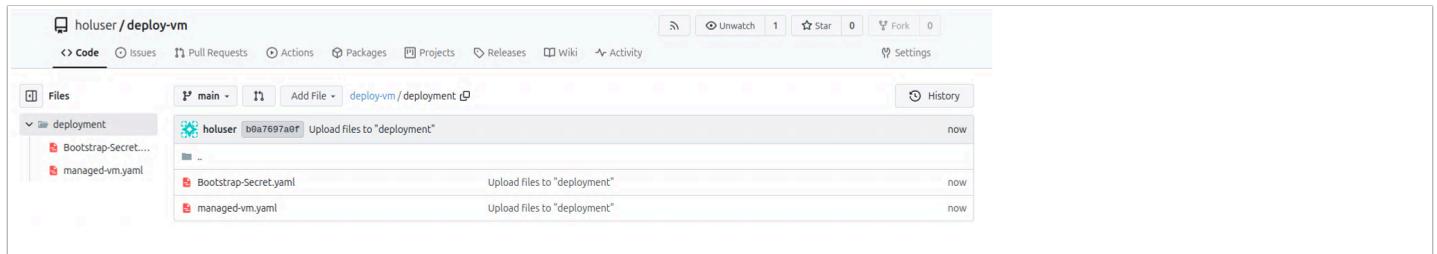
In the file Manager:

1. Click **Downloads**
2. Click the folder that extracting the downloaded zip create
3. Select both yaml files
4. Click **Open**

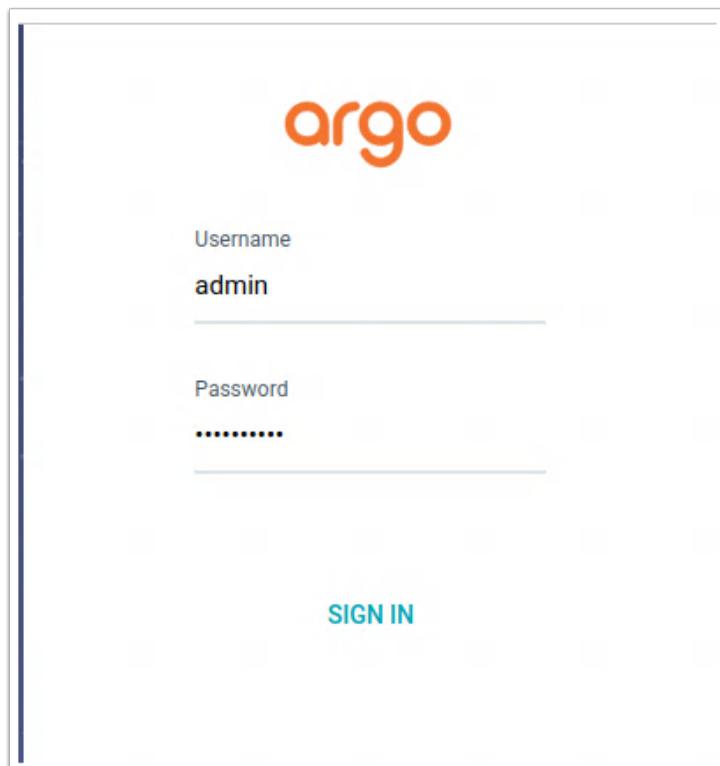


This part is standard git repo commit and sync behavior. Every time a change is committed to the repo the developer (you in this case) must document what is being done.

1. At a Minium you must add a single line to this top field, it can as simple as one word - **"Update"** or it can be much more. I leave this up to you.
2. The description can be left blank or you can be as verbose as you wish.
3. Click **Commit Changes**



Login To ArgoCD



Log Into ArgoCD

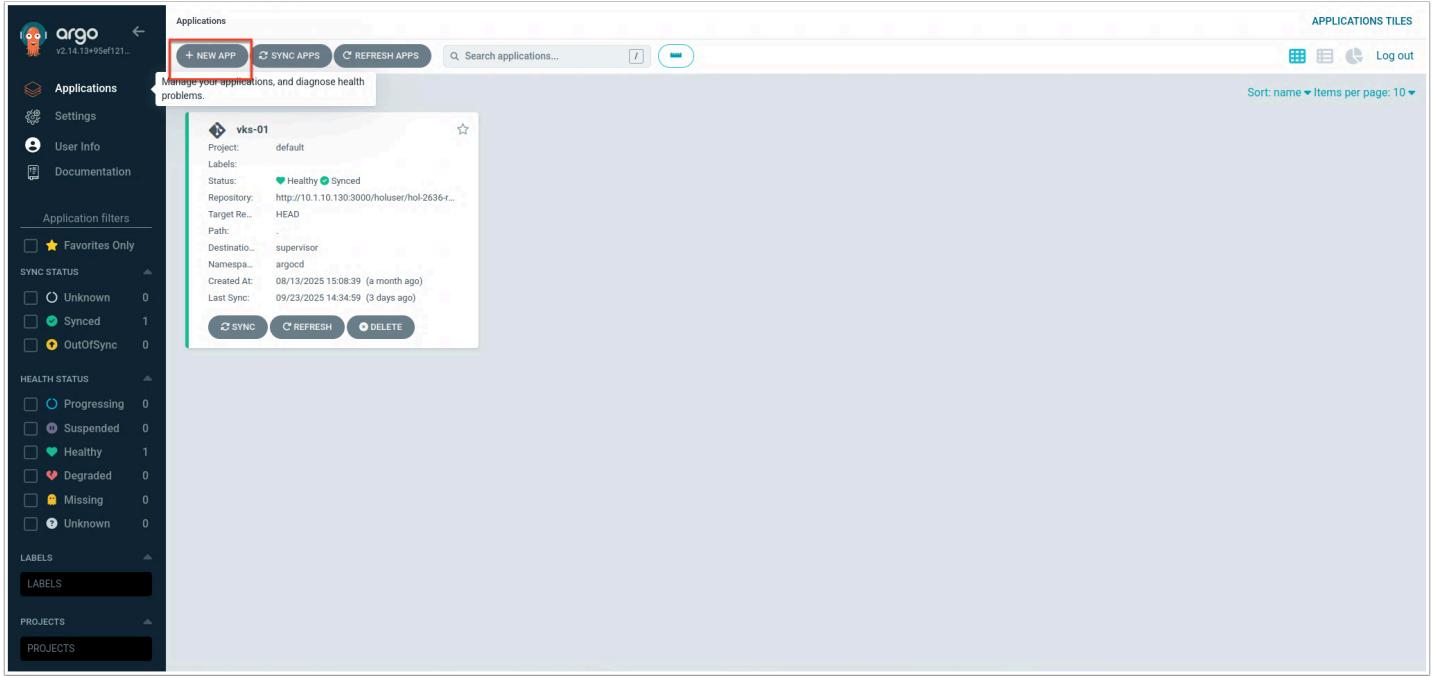
Username:

Click to copy

Password:

Click to copy

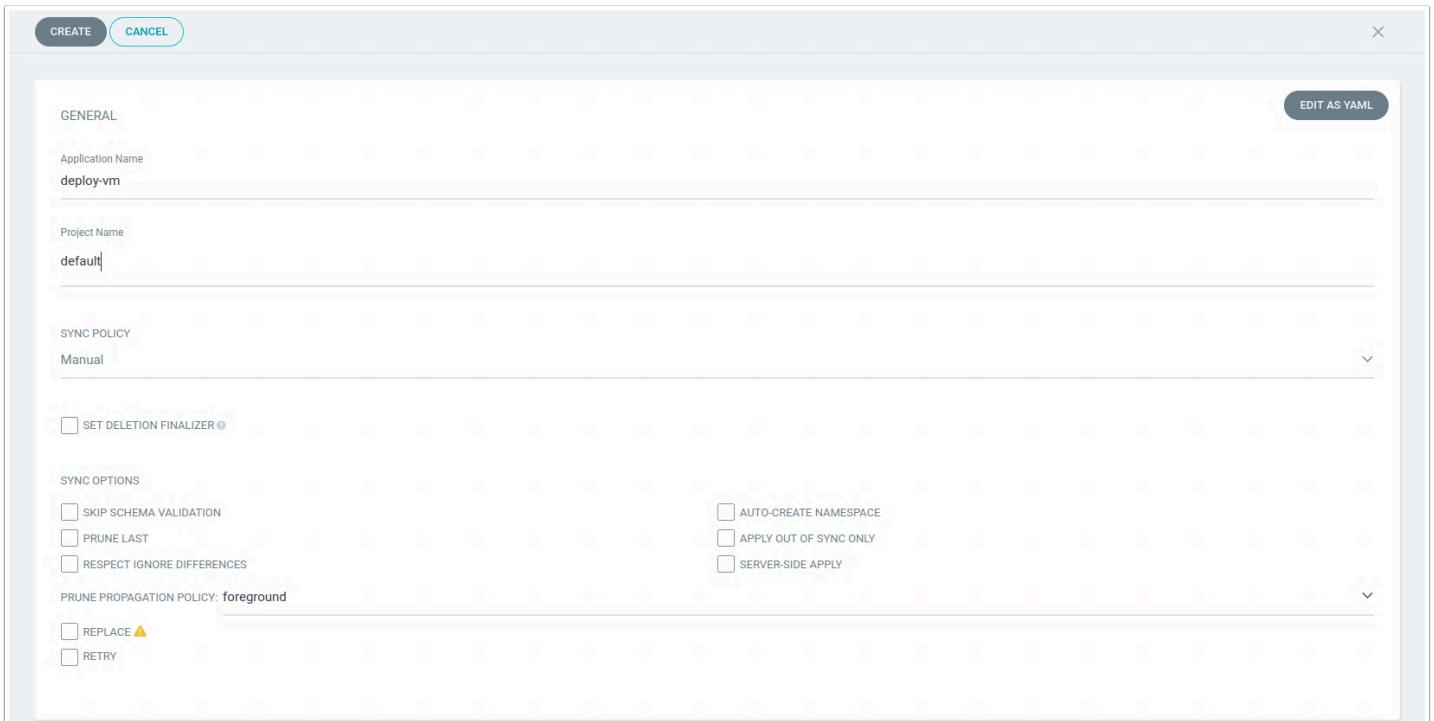
New ArgoCD App



The screenshot shows the ArgoCD application management interface. The left sidebar includes links for Applications, Settings, User Info, Documentation, Application filters (Favorites Only, Sync Status, Health Status, Labels), and Projects. The main area displays a list of existing applications, such as 'vks-01', with details like Project: default, Status: Healthy Synced, and Repository: http://10.1.10.130:3000/holuser/hol-2636-r... The top navigation bar features '+ NEW APP', 'SYNC APPS', 'REFRESH APPS', a search bar, and a log out link.

You are now going to create a new app. much like the existing App and see how ArgoCD keeps your VM in your **declared endstate configuration**.

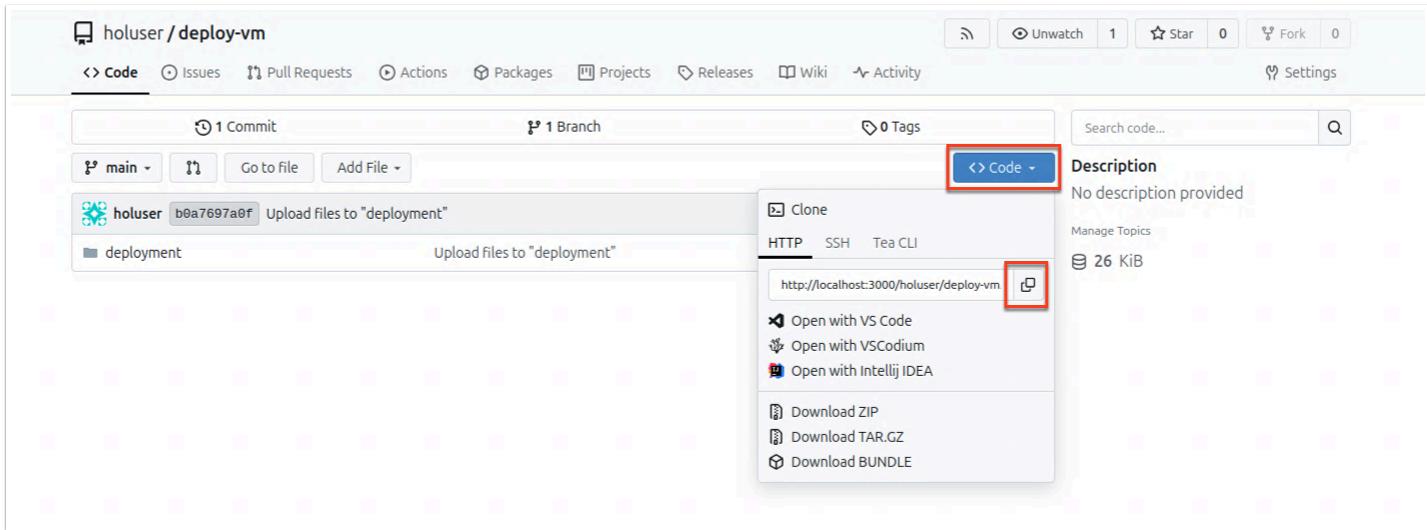
1. Click + NEW APP



The screenshot shows the 'CREATE' dialog for a new application. The 'GENERAL' tab is active, displaying fields for 'Application Name' (set to 'deploy-vm') and 'Project Name' (set to 'default'). Other tabs include 'SYNC POLICY' (set to 'Manual'), 'SYNC OPTIONS' (checkboxes for 'SKIP SCHEMA VALIDATION', 'PRUNE LAST', 'RESPECT IGNORE DIFFERENCES', 'AUTO-CREATE NAMESPACE', 'APPLY OUT OF SYNC ONLY', and 'SERVER-SIDE APPLY'), and 'PRUNE PROPAGATION POLICY' (set to 'foreground'). Buttons for 'CREATE', 'CANCEL', and 'EDIT AS YAML' are located at the top right of the dialog.

1. For Application Name: deploy-vm

2. For **Project Name**: click into field and select `default`



Go back to the **GitEA UI** and the **vm01** repo

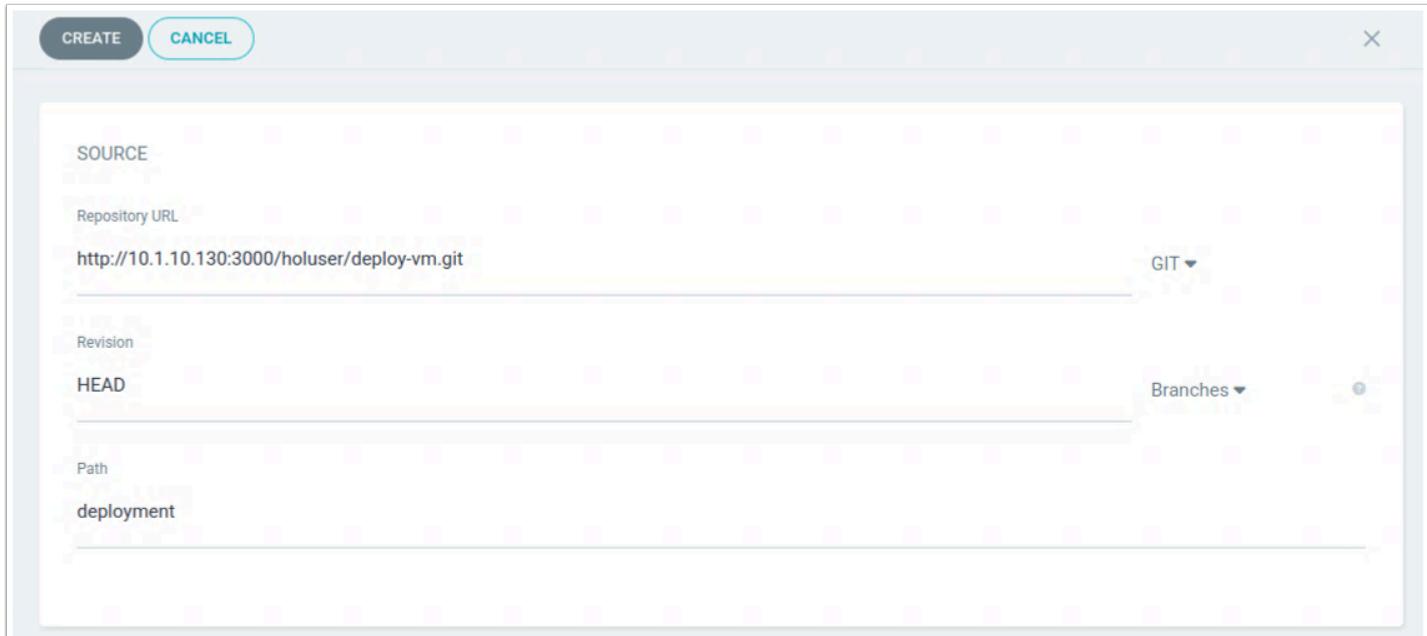
1. Click the **CODE** Dropdown
2. Click the **Copy to Clipboard** for the **HTTP Clone action**

The **URL** as it is will not work, it is handed to you as <http://localhost>, but the ArgoCD app needs the hostname or IP. So change `localhost` to `10.1.10.130` so it looks like this

`http://localhost:3000/holuser/deploy-vm.git`

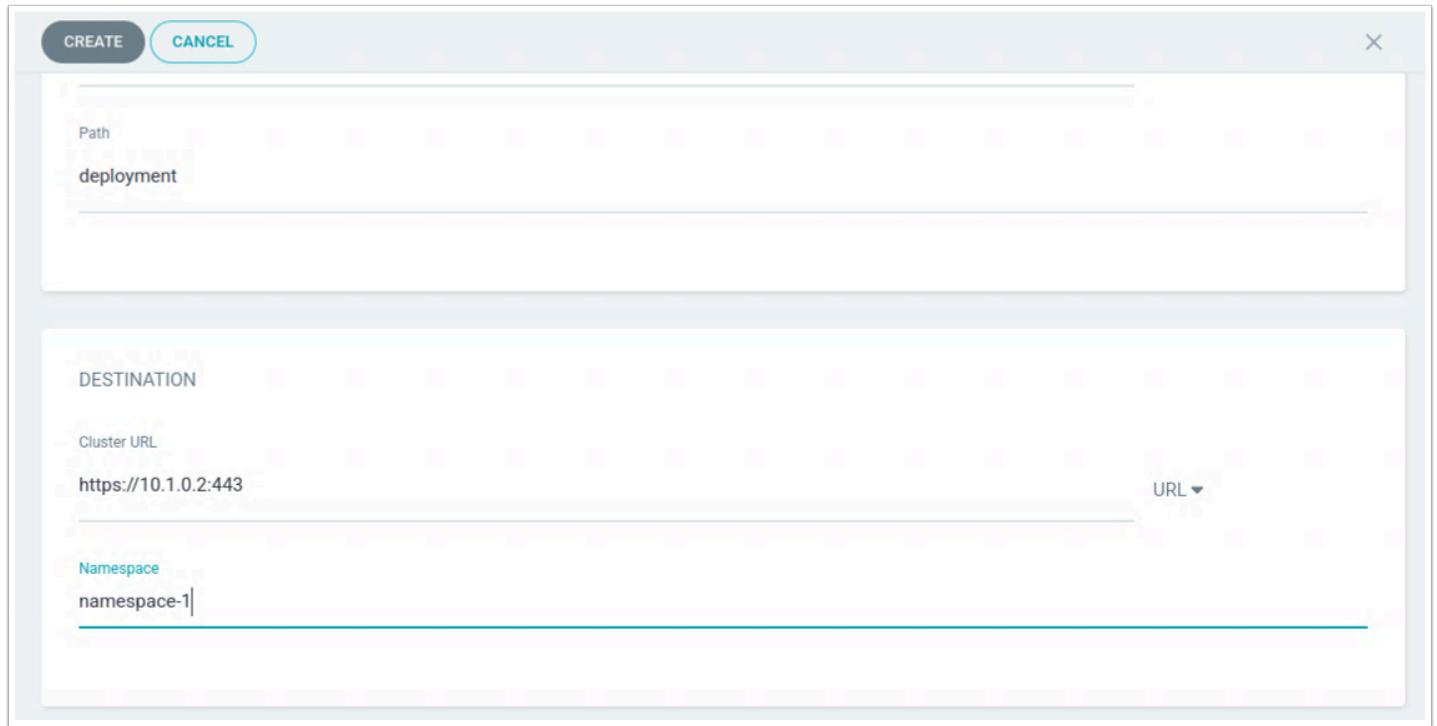
Click to copy

Copy the new **URL** into your clipboard



1. For **Repository URL**: paste your edited URL - Hit **ENTER** - this tells ArgoCD to check the URL

2. Click into **Revision**: select HEAD - **Note: it may already say HEAD, but do this anyway**
3. For **Path**: type `deployment` - the folder we put the deploy yaml into.



4. Click the **Cluster URL Dropdown** select the only entry - `https://10.1.0.2:443` - this the Supervisor IP. In a non lab environment this would be a most likley be the FQDN for a Load balanced, three node Supervisor
5. Enter `namespace-1` for **Namespace**
6. Click **CREATE**

Sync ArgoCD Application

Project: default

Labels:

Status: Healthy OutOfSync

Repository: <http://10.1.10.130:3000/holuser/deploy-vm.git>

Target Revision: HEAD

Path: deployment

Destination: supervisor

Namespace: namespace-1

Created At: 09/29/2025 09:00:52 (a few seconds ago)

SYNC REFRESH DELETE

1. Click White Space on the App tile

Applications / Q deploy-vm APPLICATION DETAILS TREE

DETAILS DIFF SYNC SYNC STATUS HISTORY AND ROLLBACK DELETE REFRESH ▾

APP HEALTH Healthy

SYNC STATUS OutOfSync from HEAD (b0a7697)

Auto sync is not enabled.

Author: holuser <holuser@site-a.vcf.lab> -
Comment: Upload files to "deployment"

APPLICATION DETAILS TREE

Log out

Deployment Tree:

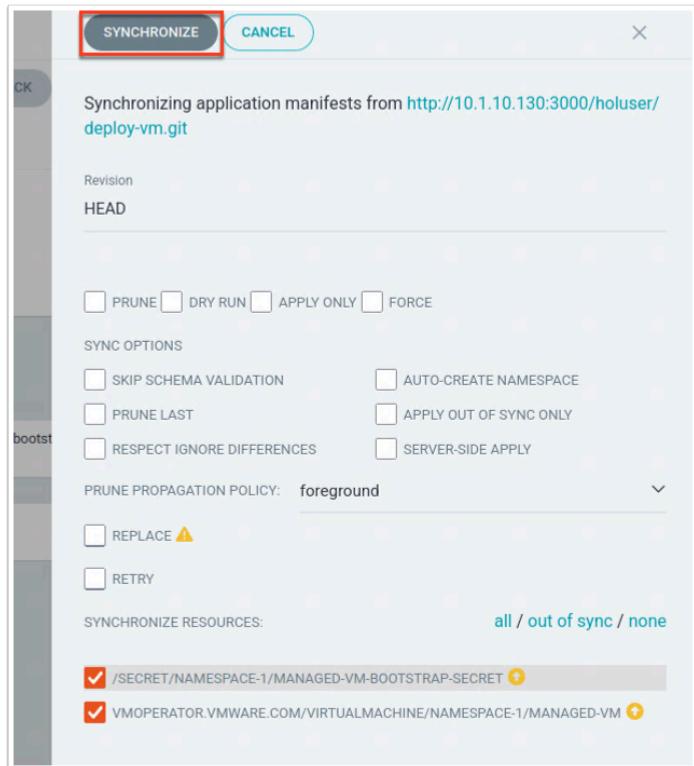
- deploy-vm (3 minutes) → managed-vm-bootstrapping-secret (secret, 8 minutes)
- managed-vm-bootstrapping-secret → managed-vm (virtualmachine, 6 minutes)
- managed-vm → managed-vm-vks-01-hbtv5-et... (subnetport, 6 minutes)

The ArgoCD UI is pretty simple. Each of the icons in this screen give you information on the state of the assets ArgoCD is managing. The heart, as you might have guessed references "HEALTH" and the Circle shows the status of Synchronization of the asset.

In this example the green heart is saying the VM has been found and seems to be healthy, BUT the yellow SYNC icon is saying it may be out of sync from the declared endstate.

So...lets synchronize it

1. Click **SYNC**



In the new window you are going to leave all the settings as is. We want to leave the two assets it has recognized as needed to be synchronized. Leave the bootstrap secrets file and the vm deployment yaml checked.

1. Click **SYNCRONIZE**

Applications / Q deploy-vm

APPLICATION DETAILS TREE

DETAILS **DIFF** **SYNC** **SYNC STATUS** **HISTORY AND ROLLBACK** **DELETE** **REFRESH**

APP HEALTH **Synced** to HEAD (b0a7697)

Auto sync is not enabled.
Author: holuser <holuser@site-a.vcf.lab> -
Comment: Upload files to "deployment"

LAST SYNC **Sync OK** to b0a7697

Succeeded a few seconds ago (Mon Sep 29 2025 09:04:01 GMT-0700)
Author: holuser <holuser@site-a.vcf.lab> -
Comment: Upload files to "deployment"

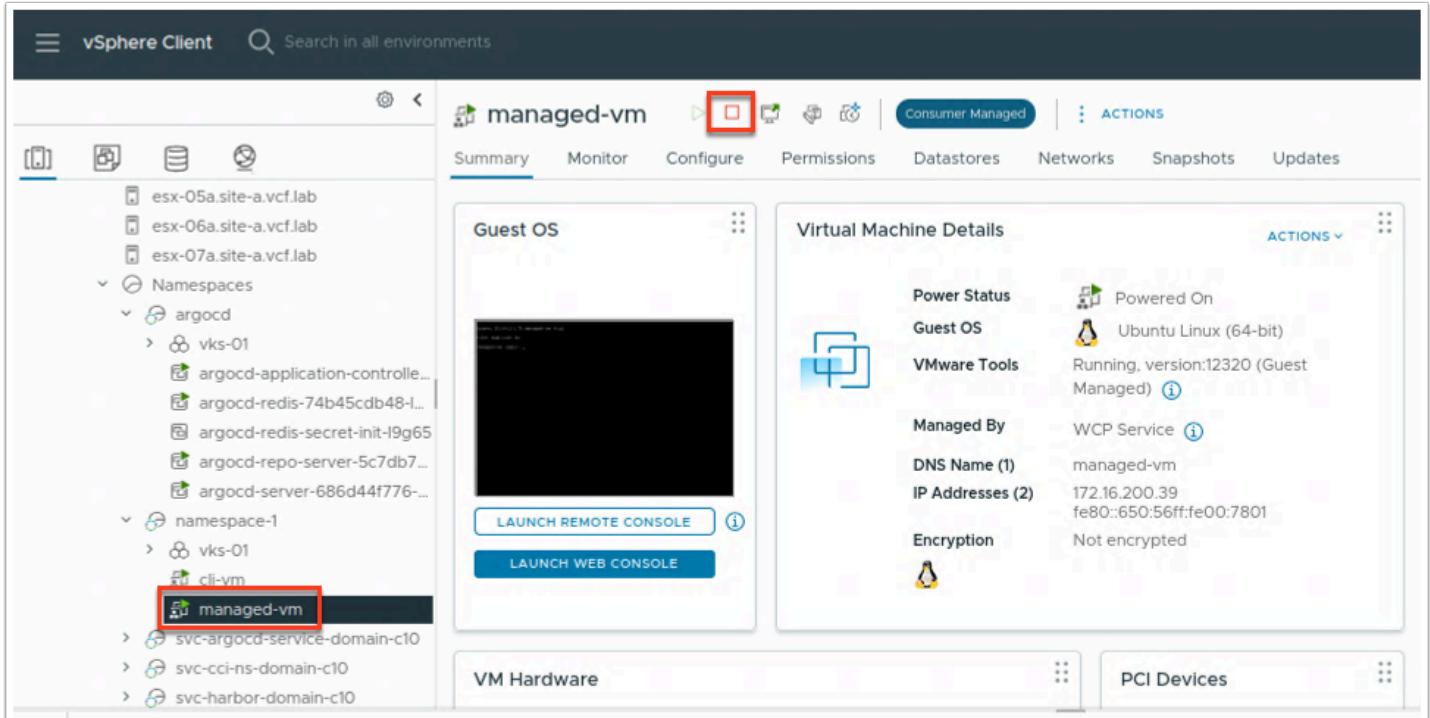
100%

The screenshot shows the 'APPLICATION DETAILS TREE' interface for the 'deploy-vm' application. At the top, there are tabs for 'DETAILS', 'DIFF', 'SYNC', 'SYNC STATUS', 'HISTORY AND ROLLBACK', 'DELETE', and 'REFRESH'. Below these are sections for 'APP HEALTH' (labeled 'Synced' to HEAD) and 'LAST SYNC' (labeled 'Sync OK'). A large central area displays a dependency graph. It starts with a 'deploy-vm' node (stack icon), which has a dependency on a 'managed-vm-bootstrap-secret' secret (key icon). This secret has a dependency on a 'managed-vm' virtual machine (VM icon). Finally, the 'managed-vm' has a dependency on a 'managed-vm-vks-01-hbt5-etc...' subnetport (SP icon). Each node has a green heart icon next to it, indicating it is healthy. Time stamps (3 minutes, 9 minutes, 6 minutes) are shown between the nodes.

In a few seconds you should see the status change to all green, healthy and synchronized. This means the managed vm is healthy running, in existence as expected, and it is currently in the expected endstate as declared.

SYNCRONIZED!!

Now Break The Managed VM

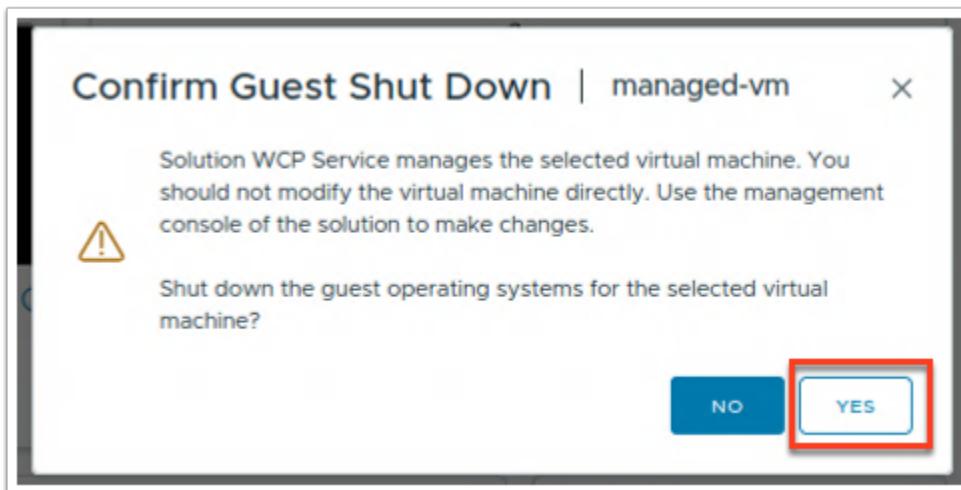


The screenshot shows the vSphere Client interface. In the left sidebar, under 'Namespaces', there is a section for 'namespace-1' which contains several entries, one of which is 'managed-vm'. This entry is highlighted with a red box. In the main content area, the 'Summary' tab for the 'managed-vm' VM is selected. The 'Virtual Machine Details' panel shows the following information:

- Power Status:** Powered On
- Guest OS:** Ubuntu Linux (64-bit)
- VMware Tools:** Running, version:12320 (Guest Managed)
- Managed By:** WCP Service
- DNS Name (1):** managed-vm
- IP Addresses (2):** 172.16.200.39, fe80::650:56ff:fe00:7801
- Encryption:** Not encrypted

Now we break it!

Put your pesky **know it** all **VI Admin** hat on. Log into the **wld01 vCenter**. Find your **VM** in **inventory** under **namespace-1** and click **power off**



"WAIT HOLD ON! "

"This VM is managed by another system?!"

"What the heck is the WCP Service?"

"I know better!"

1. Click YES

The screenshot shows the 'APPLICATION DETAILS TREE' interface for an application named 'deploy-vm'. The 'APP HEALTH' section indicates 'Progressing' (highlighted with a red box). The 'SYNC STATUS' section shows 'Synced to HEAD (b0a7697)' with a green checkmark. The 'LAST SYNC' section shows 'Sync OK to b0a7697' with a green checkmark, indicating a successful sync 2 minutes ago. Below these sections is a diagram illustrating the deployment flow: 'deploy-vm' (server icon) connects to 'managed-vm-bootstrap-secret' (secret icon), which then connects to 'managed-vm' (virtual machine icon), and finally to 'managed-vm-vks-01-hbt5-et...' (subnetport icon). Each connection has a timestamp: 5 minutes, 11 minutes, and 9 minutes respectively.

Swap back to the **ArgoCD UI**, and you will see the health gets put into **Progressing**.

The screenshot shows the ArgoCD UI with the 'Recent Tasks' tab selected. A task named 'Power On virtual machine' is highlighted with a red box. This task is associated with the 'managed-vm' target. Below it, another task 'Initiate guest OS shutdown' is also highlighted with a red box and is also associated with the 'managed-vm' target. Both tasks have a status of 'Corr' (correct).

Swap back to the **wld01 vCenter** and in tasks you will see the **VM Shutdown**, then **..PowerOn!!?**

Project: deploy-vm

Labels:

Status: Progressing Synced

Repository: http://10.1.10.130:3000/holuser/deploy-vm.git

Target Revision: HEAD

Path: deployment

Destination: supervisor

Namespace: namespace-1

Created At: 09/29/2025 09:00:52 (2 days ago)

Last Sync: 09/29/2025 10:57:50 (2 days ago)

SYNC **REFRESH** **DELETE**

Back to ArgoCD

Project: deploy-vm

Labels:

Status: Healthy Synced

Repository: http://10.1.10.130:3000/holuser/deploy-vm.git

Target Revision: HEAD

Path: deployment

Destination: supervisor

Namespace: namespace-1

Created At: 09/29/2025 09:00:52 (2 days ago)

Last Sync: 09/29/2025 10:57:50 (2 days ago)

SYNC **REFRESH** **DELETE**

And as soon as the VM reports back in the expected state, it goes healthy.

-  From here there are a bunch of other things we can try, like change the encoded password in the secrets file. Change storage policies. Some of them will work as expected some will take a bit more pre-work to get the full workflow interrupted and working. This is meant to be a technical intro. Explore you or your customers requirements, but a real use and go from there.

Additional Information

Additional Information

Thanks to all who took the time to take this lab and for your interest and support for the VCF/VKS service. For more information on VKS and Supervisor Services:

<https://blogs.vmware.com/cloud-foundation/?s=breakroom+chats>

The following two videos walk thru the updates of the Supervisor and VKS which were covered in Module 4.

NOTE : We cannot guarantee that headsets will be available in the Hands On Labs area so as a courtesy to those around you, if headsets are not available please watch and listen on a personal device or use your own headset.

Supervisor update and VKS update videos:

<https://youtu.be/KOeVCcr7aa0>

<https://youtu.be/bCUVPJ-SnMA>

VKS Tech Talk:

<https://www.youtube.com/watch?v=F3CdrmgjVN4>

Conclusion

In this module, we demonstrated how to use gitea and ArgoCD to demonstrate continuous deployment using the ArgoCD supervisor which we deployed for you.



End Of Lab manual