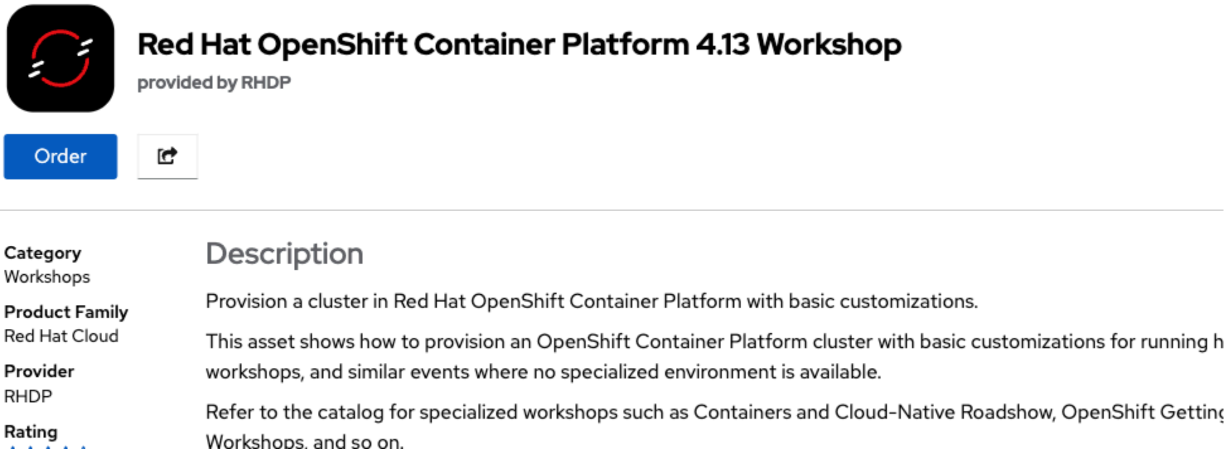
RHODS Demo for Object Detection

First step deploy OCP 4.13 workshop in demo.redhat.com

Select the Practice/ Enablement and purpose as ‘conduct internal training’



Red Hat OpenShift Container Platform 4.13 Workshop

provided by RHDP

Category
Workshops

Product Family
Red Hat Cloud

Provider
RHDP

Rating

Description

Provision a cluster in Red Hat OpenShift Container Platform with basic customizations.

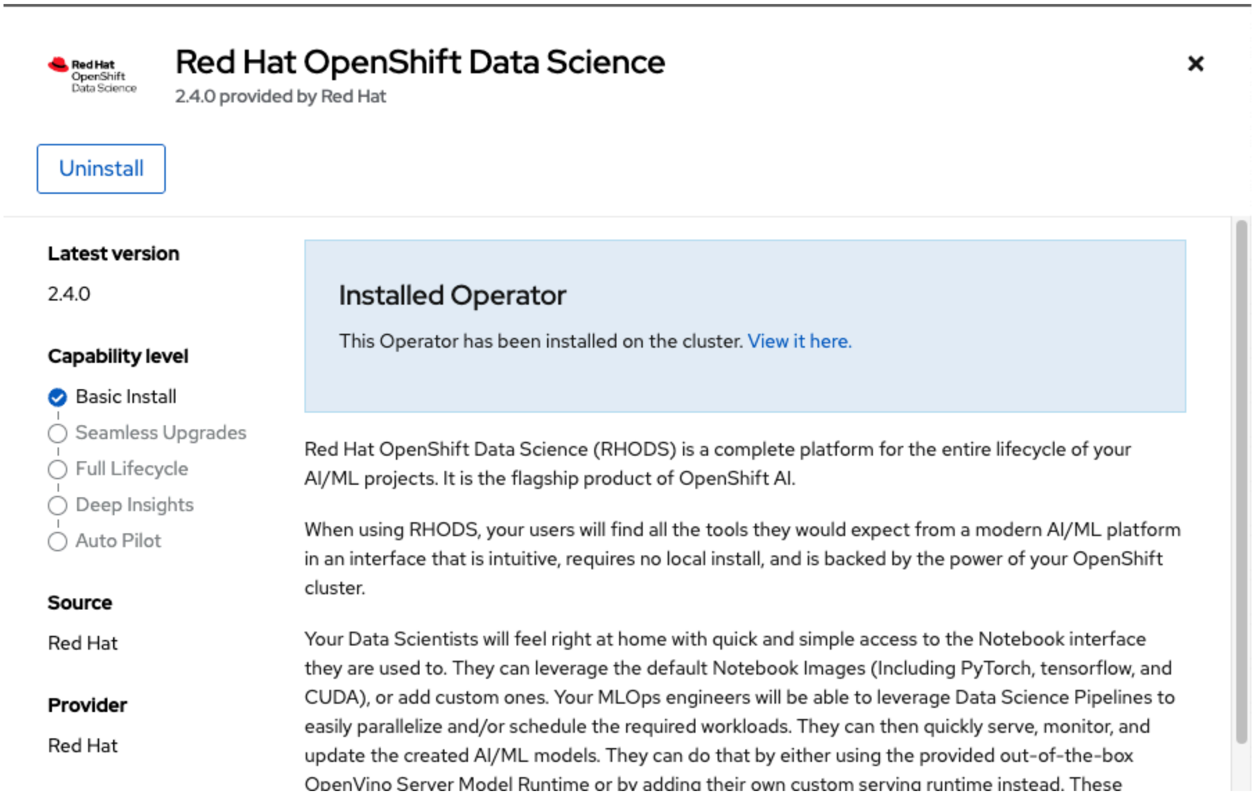
This asset shows how to provision an OpenShift Container Platform cluster with basic customizations for running h workshops, and similar events where no specialized environment is available.

Refer to the catalog for specialized workshops such as Containers and Cloud-Native Roadshow, OpenShift Gettinc Workshops, and so on.

After approx 90 mins you will get an email with login details as example below.
OpenShift Console: https://console-openshift-console.apps.cluster-5vn9m.dynamic.opentlc.com
OpenShift API for command line 'oc' client: https://api.cluster-5vn9m.dynamic.opentlc.com:6443
Download oc client from
http://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.13.1/openshift-client-linux-4.13.1.tar.gz
Authentication via htpasswd is enabled on this cluster.
User admin with password AHRy6Bnu0ROxVUTD is cluster admin.
Normal user user1 created with password PJZteLHosOukvFn2
You can access your bastion via SSH:
ssh lab-user@ssh.ocpv01.dal10.infra.demo.redhat.com -p 30960
Enter ssh password when prompted: piscM9vplQYO

Click on the OCP Console link and use ‘admin’ and password given in the email but you can also see details in the demo.redhat.portal.

We will also need to ssh on to the bastion server to run some commands so be familiar with this process. When on the OCP console install operators for RHODS and Pipelines , you should see below when completed



Red Hat OpenShift Data Science

2.4.0 provided by Red Hat

Latest version

2.4.0

Capability level

- ☑ Basic Install
- ○ Seamless Upgrades
- ○ Full Lifecycle
- ○ Deep Insights
- ○ Auto Pilot

Source

Red Hat

Provider

Red Hat

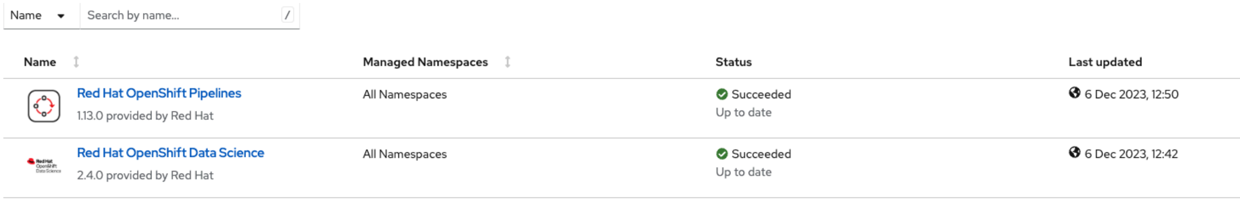
Installed Operator

This Operator has been installed on the cluster. View it here.

Red Hat OpenShift Data Science (RHODS) is a complete platform for the entire lifecycle of your AI/ML projects. It is the flagship product of OpenShift AI.

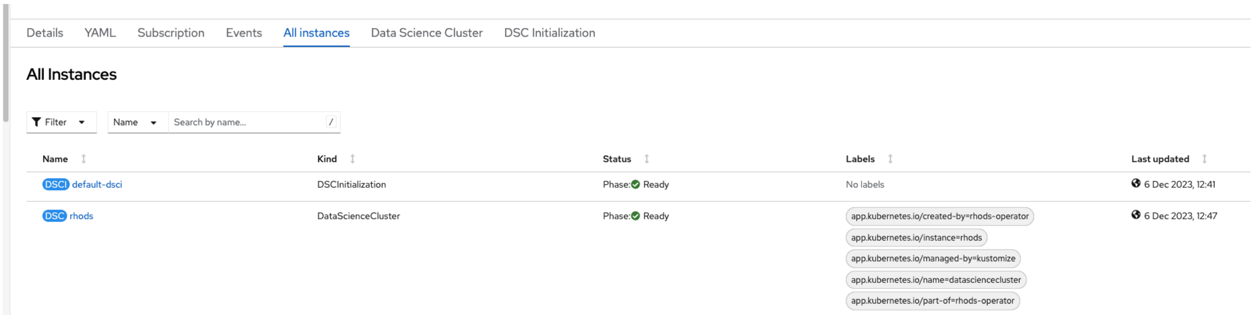
When using RHODS, your users will find all the tools they would expect from a modern AI/ML platform in an interface that is intuitive, requires no local install, and is backed by the power of your OpenShift cluster.

Your Data Scientists will feel right at home with quick and simple access to the Notebook interface they are used to. They can leverage the default Notebook Images (Including PyTorch, tensorflow, and CUDA), or add custom ones. Your MLOps engineers will be able to leverage Data Science Pipelines to easily parallelize and/or schedule the required workloads. They can then quickly serve, monitor, and update the created AI/ML models. They can do that by either using the provided out-of-the-box OpenVino Server Model Runtime or by adding their own custom serving runtime instead. These



Name	Managed Namespaces	Status	Last updated
Red Hat OpenShift Pipelines 1.13.0 provided by Red Hat	All Namespaces	✓ Succeeded Up to date	6 Dec 2023, 12:50
Red Hat OpenShift Data Science 2.4.0 provided by Red Hat	All Namespaces	✓ Succeeded Up to date	6 Dec 2023, 12:42

With 2.4 version you have to create the RHODS cluster, just follow the instructions and after a few minutes you will see below.



All Instances

Name	Kind	Status	Labels	Last updated
DSCI default-dsci	DSCInitialization	Phase: ✓ Ready	No labels	6 Dec 2023, 12:41
DSC rhods	DataScienceCluster	Phase: ✓ Ready	app.kubernetes.io/created-by=rhods-operator app.kubernetes.io/instance=rhods app.kubernetes.io/managed-by=kustomize app.kubernetes.io/name=datasciencecluster app.kubernetes.io/part-of=rhods-operator	6 Dec 2023, 12:47

Next step will be use terminal if you have a Mac by entering the command in the email you were sent of from the demo.redhat.com, in my case I did this with command on the terminal

ssh lab-user@ssh.ocpv01.dal10.infra.demo.redhat.com -p 30960   (this will be different for your setup)

Enter ssh password when prompted: piscM9vplQYO   (this will be different for your setup)

Then you will do first command in bold, change to the os-mlops/manifest directory after the git clone completes, then follow the next two command in bold

Note: Deploy the OpenShift Data Science operator on your OpenShift cluster. The demo pack has been tested with RHOAI 2.5.0.

For reference this is Max git repo https://github.com/mamurak/os-mlops

```
[lab-user@bastion ~]$ git clone https://github.com/mamurak/os-mlops
Cloning into 'os-mlops'...
remote: Enumerating objects: 3016, done.
remote: Counting objects: 100% (634/634), done.
remote: Compressing objects: 100% (267/267), done.
remote: Total 3016 (delta 433), reused 551 (delta 362), pack-reused 2382
Receiving objects: 100% (3016/3016), 15.86 MiB | 32.68 MiB/s, done.
Resolving deltas: 100% (1833/1833), done.
[lab-user@bastion ~]$ ls -lrt
total 4
drwxr-xr-x. 6 lab-user users 4096 Dec  6 07:52 os-mlops
[lab-user@bastion ~]$ cd os-mlops/
[lab-user@bastion os-mlops]$ ls
container-images   gpu-enablement.md  notebooks           odh-pachyderm-pipelines.md    README.md
rhods-kfp-modelmesh.md            rhods-workflow.png
elyra-runtime.png  manifests          odh-kfp-modelmesh.md  odh-rhods-troubleshooting.md
rhods-data-science-pipelines.md  rhods-od-workshop-instructions.md  visual-inspection-on-ocp.md
[lab-user@bastion os-mlops]$ cd manifests/
[lab-user@bastion manifests]$ oc apply -f dependencies.yaml  >>Wait until the
DataScienceCluster CR has been deployed.
project.project.openshift.io/fraud-detection created
project.project.openshift.io/object-detection created
project.project.openshift.io/ray-demo created
project.project.openshift.io/spark-demo created
project.project.openshift.io/minio created
namespace/production-project created
namespace/ci-cd created
[lab-user@bastion manifests]$ oc apply -k .
role.rbac.authorization.k8s.io/spark-role created
rolebinding.rbac.authorization.k8s.io/spark-serviceaccount_name created
secret/aws-connection-fraud-detection created
secret/aws-connection-pipelines created
secret/minio-secret created
secret/aws-connection-object-detection created
secret/aws-connection-pipelines created
secret/aws-connection-pipelines created
secret/starburst-data-connection created
service/minio-service created
persistentvolumeclaim/development created
persistentvolumeclaim/fraud-detection-offline-scoring-pvc created
persistentvolumeclaim/fraud-detection-training-pvc created
persistentvolumeclaim/minio-pvc created
```

```
persistentvolumeclaim/development created
persistentvolumeclaim/object-detection-training-pvc created
persistentvolumeclaim/offline-scoring-pvc created
persistentvolumeclaim/development created
deployment.apps/minio created
datasciencepipelinesapplication.datasciencepipelinesapplications.opendatahub.io/pipelines-definition
created
datasciencepipelinesapplication.datasciencepipelinesapplications.opendatahub.io/pipelines-definition
created
datasciencepipelinesapplication.datasciencepipelinesapplications.opendatahub.io/pipelines-definition
created
imagestream.image.openshift.io/code-server-cuda-data-science created
imagestream.image.openshift.io/codeflare created
imagestream.image.openshift.io/face-recognition created
imagestream.image.openshift.io/fraud-detection created
imagestream.image.openshift.io/langchain created
imagestream.image.openshift.io/langflow created
imagestream.image.openshift.io/monai created
imagestream.image.openshift.io/object-detection created
imagestream.image.openshift.io/optapy created
imagestream.image.openshift.io/rstudio-cuda-data-science created
imagestream.image.openshift.io/spark created
imagestream.image.openshift.io/trino created
imagestream.image.openshift.io/vscode-kfp created
networkpolicy.networking.k8s.io/allow-from-all-namespaces created
networkpolicy.networking.k8s.io/allow-from-ingress-namespace created
networkpolicy.networking.k8s.io/allow-from-all-namespaces created
networkpolicy.networking.k8s.io/allow-from-ingress-namespace created
networkpolicy.networking.k8s.io/allow-from-all-namespaces created
networkpolicy.networking.k8s.io/allow-from-ingress-namespace created
route.route.openshift.io/minio-api created
route.route.openshift.io/minio-ui created
servingruntime.serving.kserve.io/ovms created
servingruntime.serving.kserve.io/triton created
template.template.openshift.io/mlserver created
template.template.openshift.io/torchserve created
template.template.openshift.io/triton created
```

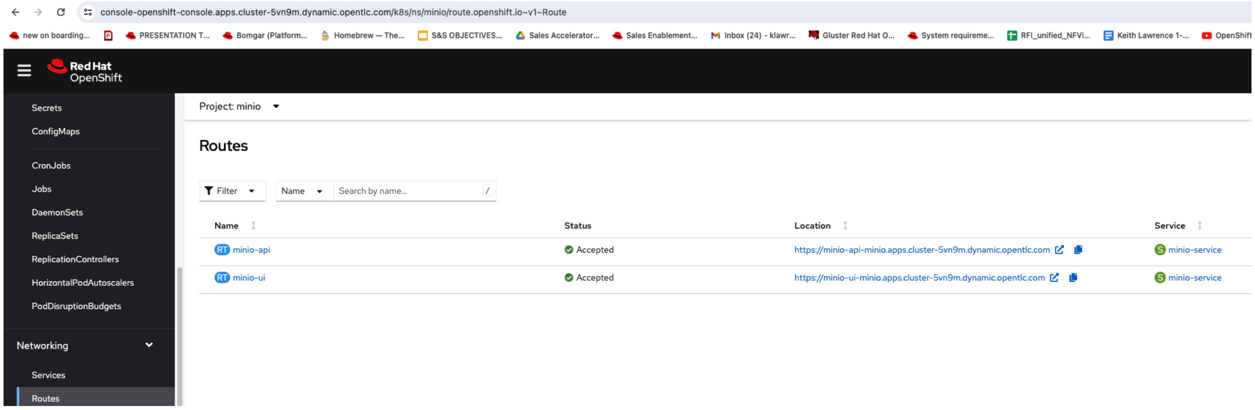
You can leave the terminal now or leave it open as we will switch to the OCP console.

Make sure you have a tab / window open for the workshop from Codrin that will step through the object demo.
https://codrinbucur.github.io/rhods-od-workshop/rhods-od-workshop/1-01-project-setup.html

This is the link you will need when you are at the stage to git clone the demo but it is in the instructions above https://github.com/mamurak/object-detection-rest.git

OK…..now you have executed the commands to build the Data Science projects when you were on the terminal it also set up a project for Minio which is used for S3 object storage, we need to do some manual work here..

In the OCP console we need to find the route to Minio so this is very easy and you can see from screen shot below how to achieve this, make sure you are in the Project for minio

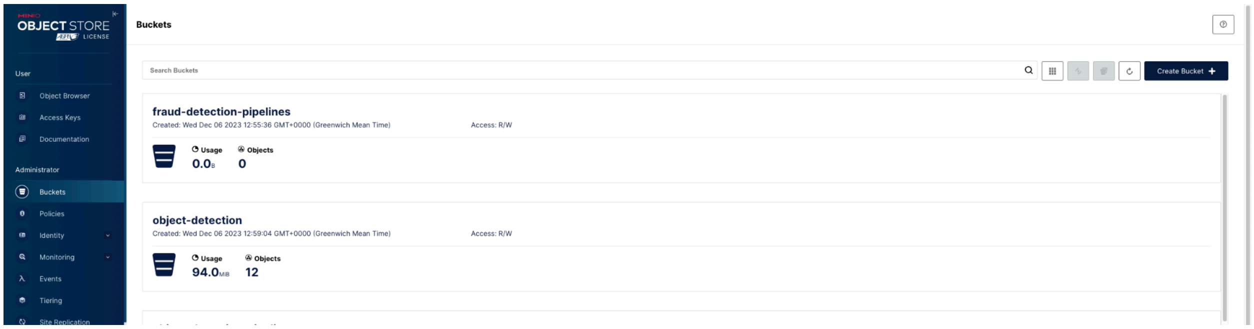


Routes

Name	Status	Location	Service
RT minio-api	Accepted	https://minio-api-minio.apps.cluster-5vn9m.dynamic.opentlc.com	S minio-service
RT minio-ui	Accepted	https://minio-ui-minio.apps.cluster-5vn9m.dynamic.opentlc.com	S minio-service

Click on the location for the minio-ui and a new tab will open, user = minio and the password = minio123

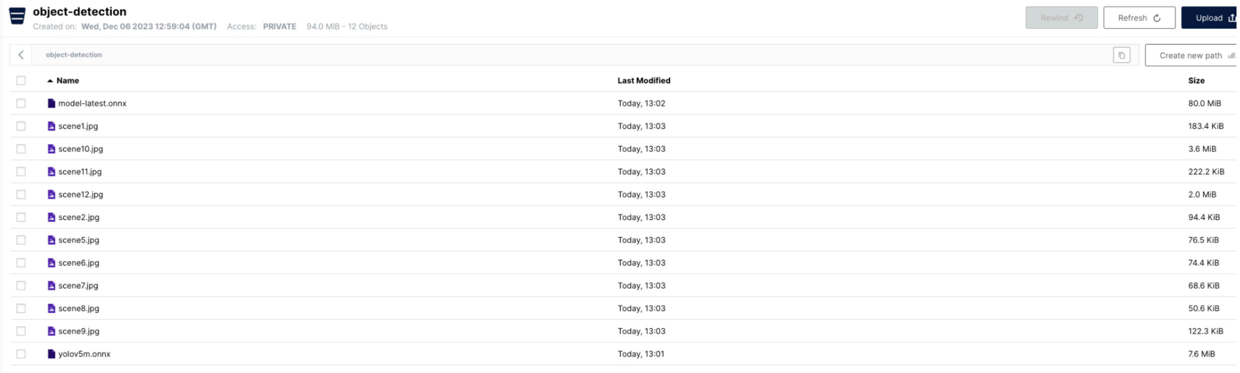
Go to the Buckets tab under Administrator, you will see some buckets but we need to create a new bucket called ‘object-detection’ like below



fraud-detection-pipelines
Created: Wed Dec 06 2023 12:55:36 GMT+0000 (Greenwich Mean Time)
Access: R/W
Usage 0.0B
Objects 0

object-detection
Created: Wed Dec 06 2023 12:59:04 GMT+0000 (Greenwich Mean Time)
Access: R/W
Usage 94.0MiB
Objects 12

Nearly finished in this UI, we need to now go to the Object Browser under the User tab as we need to upload some files to the object-detection bucket we created.
Upload the following files I have dropped in the shared folder I will create on my google drive….https://drive.google.com/drive/folders/1yv3HzDNsMDMV0FCrv6tNi3CkcsrLuK2c?usp=sharing



object-detection
Created on: Wed, Dec 06 2023 12:59:04 (GMT) Access: PRIVATE 94.0 MiB - 12 Objects

	Name	Last Modified	Size
□	model-latest.onnx	Today, 13:02	80.0 MiB
□	scene1.jpg	Today, 13:03	183.4 KiB
□	scene10.jpg	Today, 13:03	3.6 MiB
□	scene11.jpg	Today, 13:03	222.2 KiB
□	scene12.jpg	Today, 13:03	2.0 MiB
□	scene2.jpg	Today, 13:03	94.4 KiB
□	scene5.jpg	Today, 13:03	76.5 KiB
□	scene6.jpg	Today, 13:03	74.4 KiB
□	scene7.jpg	Today, 13:03	68.6 KiB
□	scene8.jpg	Today, 13:03	50.6 KiB
□	scene9.jpg	Today, 13:03	122.3 KiB
□	yolov5m.onnx	Today, 13:01	7.6 MiB

Now we are ready to just follow the workshop instructions you have in another tab… https://codrinbucur.github.io/rhods-od-workshop/rhods-od-workshop/1-01-project-setup.html