neueda

futureproof your workforce

DATA PERSISTENCE

NEUEDA.COM

Objectives

- Data Persistence in OpenShift
- Persistent Volumes using NFS
- Persistent Volume Claims
- OpenShift Volumes
- Data Persistence Using Databases
- OpenShift Databases

Data Persistence in OpenShift

- In our previous lab we were able to deploy an application, but any data we put in was lost when the app updated.
- Actually, by design Kubernetes Pods are *ephemeral*.
    - Any given Pod could be replaced at any time.
- However persistent state is a critical aspect of many applications.

neueda
futureproof your workforce

Example: Pods are Ephemeral

- Let’s look at the Image Uploader deployment again
- Upload an image and see how it displays in the website
- Now scale the application down to 0 pods, then scale back up to 1 pod
- Open the URL again - no image!
- What happened? Pods (and their memory) are ephemeral
    - When the Pod is deleted, all its memory goes with it.

neueda futureproof your workforce

Handling Permanent Data

- OpenShift makes persistent storage available for
  - Data that is shared between Pods
  - Data that needs to persist past the lifetime of any particular Pod
- Permanent storage in pods is handled using **Persistent Volumes** (PVs)
- This can be supplied from a variety of storage solutions
  - NFS, HostPath (local directories), AWS EBS, Azure Disk, etc

Persistent Volumes using NFS

- In OpenShift the process of creating persistent file storage has three steps:

1 Create a Persistent Volume (these are cluster-wide resources)

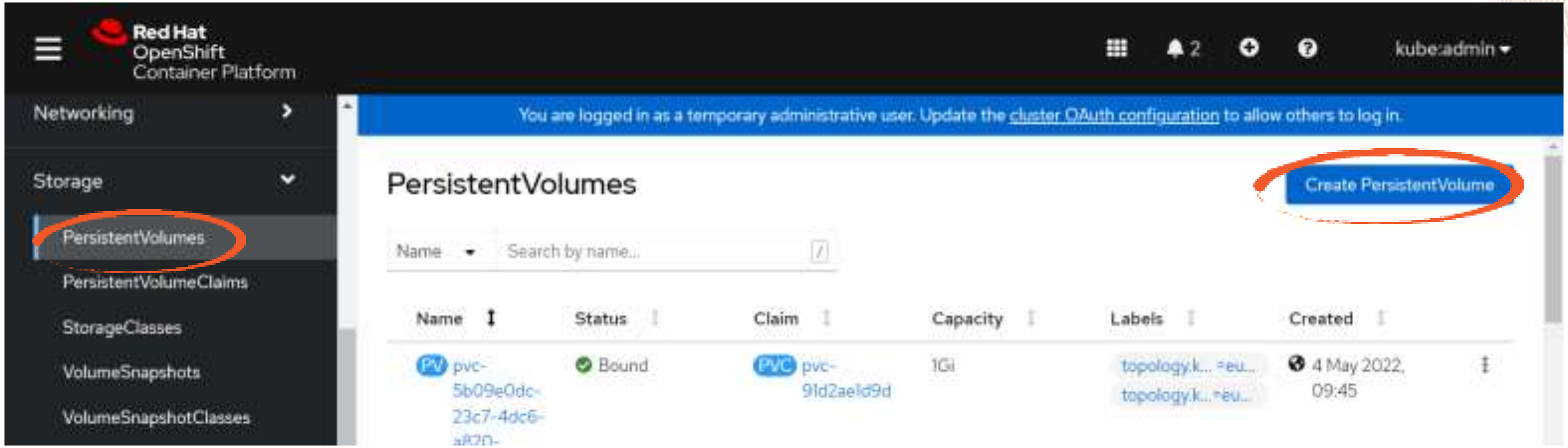
2 Create a **Persistent Volume Claim** (PVC) in your project.

3 Mount the persistent storage as a **Volume** into your application’s Pods

Creating a Persistent Volume

- Persistent Volumes are cluster-wide resources and really should only be allocated by the Cluster Administrator
- Go to the Administrator perspective in your Web Console and click on the Storage item.
    - See there is no “PersistentVolumes” option.
- In the Cluster Admin console you can see that option is available.

Creating a Persistent Volume



Red Hat OpenShift Container Platform

2

kube:admin

Networking

Storage

PersistentVolumes

PersistentVolumeClaims

StorageClasses

VolumeSnapshots

VolumeSnapshotClasses

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

PersistentVolumes

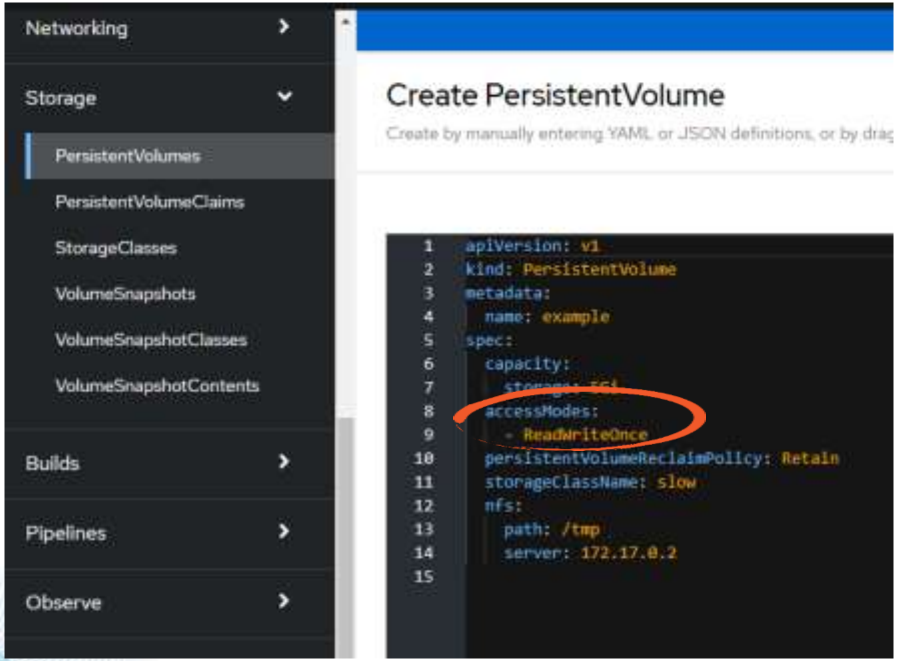
Create PersistentVolume

Name | Search by name... | /

Name	Status	Claim	Capacity	Labels	Created
PV pvc-5b09e0dc-23c7-4dc6-a820-	Bound	PVC pvc-91d2ae1d9d	1Gi	topology.k... =eu... topology.k... =eu...	4 May 2022, 09:45

neueda
futureproof your workforce

Configuring a PersistentVolume: accessMode



Create PersistentVolume

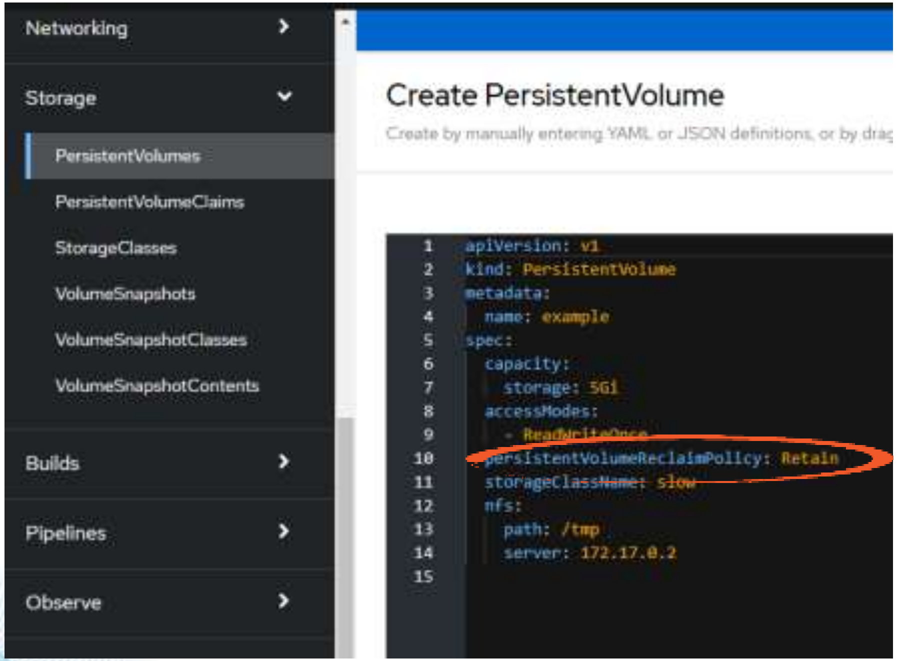
Create by manually entering YAML or JSON definitions, or by drag

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: example
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: slow
  nfs:
    path: /tmp
    server: 172.17.0.2
```

- The **accessMode** can be
  - Read/Write once (RWO) - Can be mounted as read/write by a single node in the cluster
  - Read-only many (ROX) - Can be mounted as read-only by many nodes
  - Read/Write many (RWX) - Can be mounted as read/write by multiple nodes in the cluster.

neueda
Futureproof your workforce

Configuring a PersistentVolume: Reclaim Policy



Create PersistentVolume

Create by manually entering YAML or JSON definitions, or by drag

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: example
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: slow
  nfs:
    path: /tmp
    server: 172.17.0.2
```

- The reclaim policy dictates how a PV handles reclaiming space after a storage claim on the PV is no longer required. It can be
  - Retain - all data is retaining in the PV, you have to reclaim space manually
  - Recycle - all data is automatically removed when the claim is deleted.

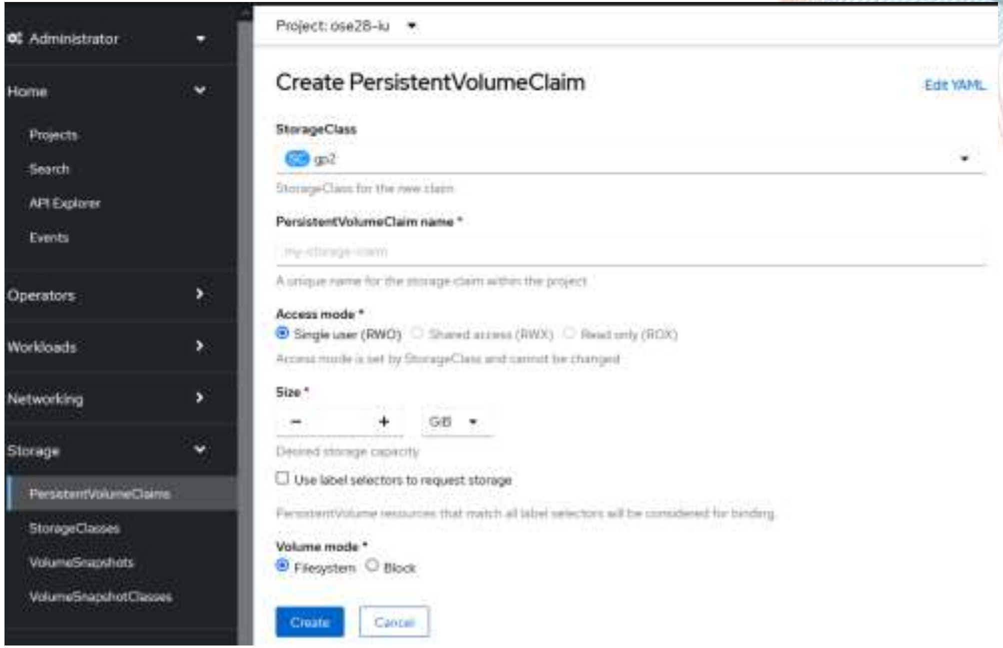
Using Persistent Storage

- To take advantage of a PV, you need to make a claim on that PV.
- PVs represent available storage, and PVCs represent an application’s need for that storage.
- When you create a PVC, OpenShift looks for the best fit among the available PVs and reserves it for use by the PVC.
    - PV size vs PVC need - use the smallest available PV
    - Access Mode - use a PV with the same or greater access privileges than that required by the PVC

neueda
futureproof your workforce

Adding a Persistent Volume Claim

- PersistentVolumeClaims can be created by accessing the PersistentVolumeClaims item under storage in the Administrator perspective



Project: ose28-lu

Create PersistentVolumeClaim

Edit YAML

StorageClass

SC gp2

StorageClass for the new claim

PersistentVolumeClaim name *

my-storage-claim

A unique name for the storage claim within the project

Access mode *

[✓ Single user (RWO)] Shared access (RWX) Read only (ROX)

Access mode is set by StorageClass and cannot be changed

Size *

− + GiB

Desired storage capacity

☐ Use label selectors to request storage

PersistentVolume resources that match all label selectors will be considered for binding

Volume mode *

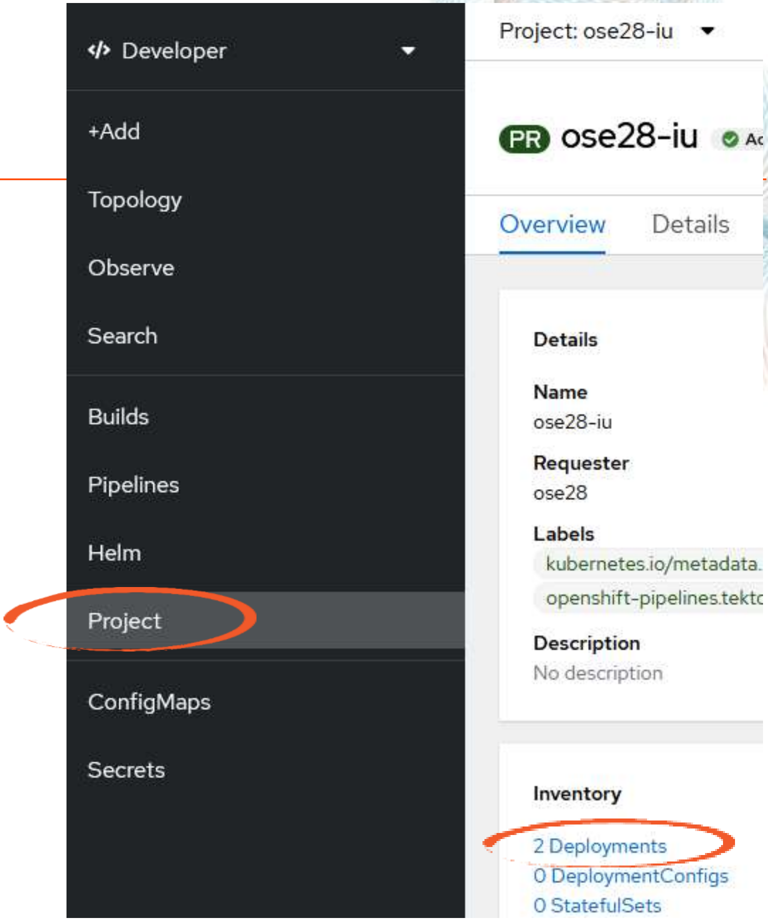
[✓ Filesystem] Block

Create Cancel

neueda
futureproof your workforce

OpenShift Volumes

- A *volume* is any filesystem, file or data mounted into an application's Pods to provide persistent data.
- To attach a PVC as a volume in our app using the Web Console you need to get to the project overview page and select the Deployment.



Details

Name
ose28-iu

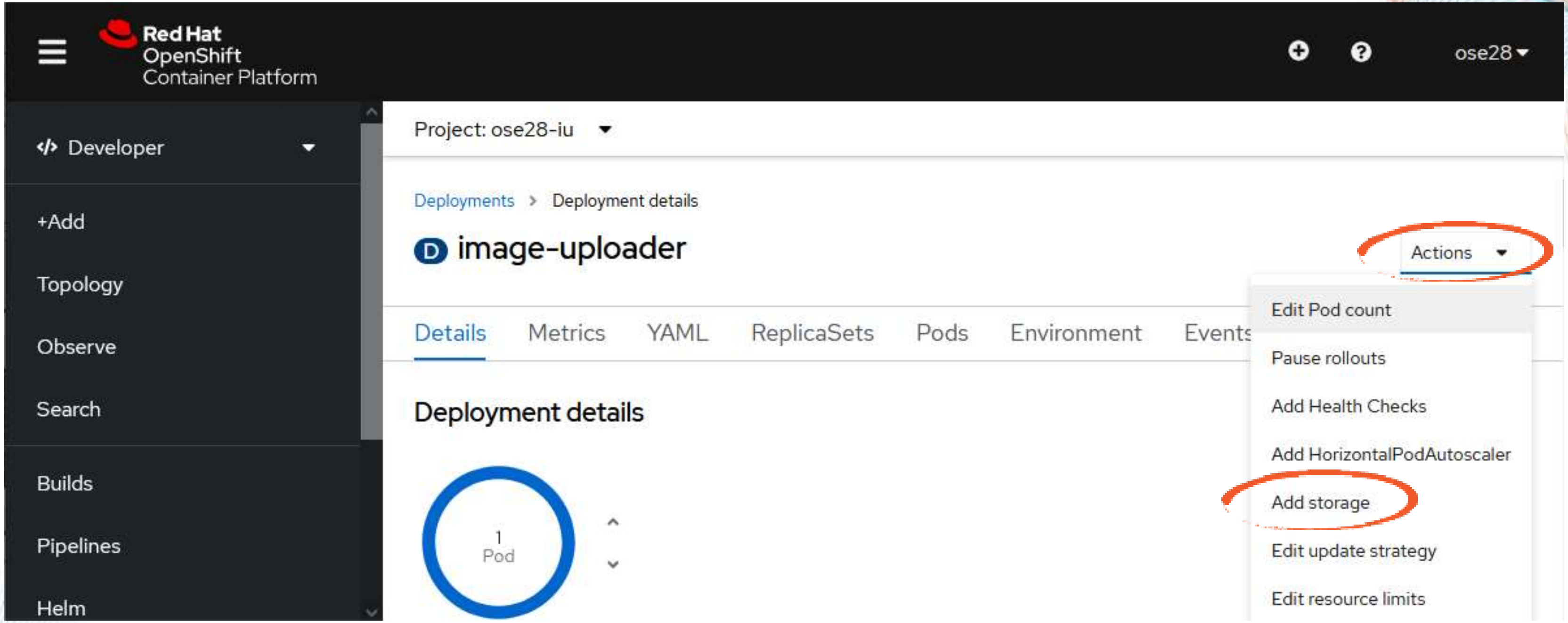
Requester
ose28

Labels
kubernetes.io/metadata.
openshift-pipelines.tekt

Description
No description

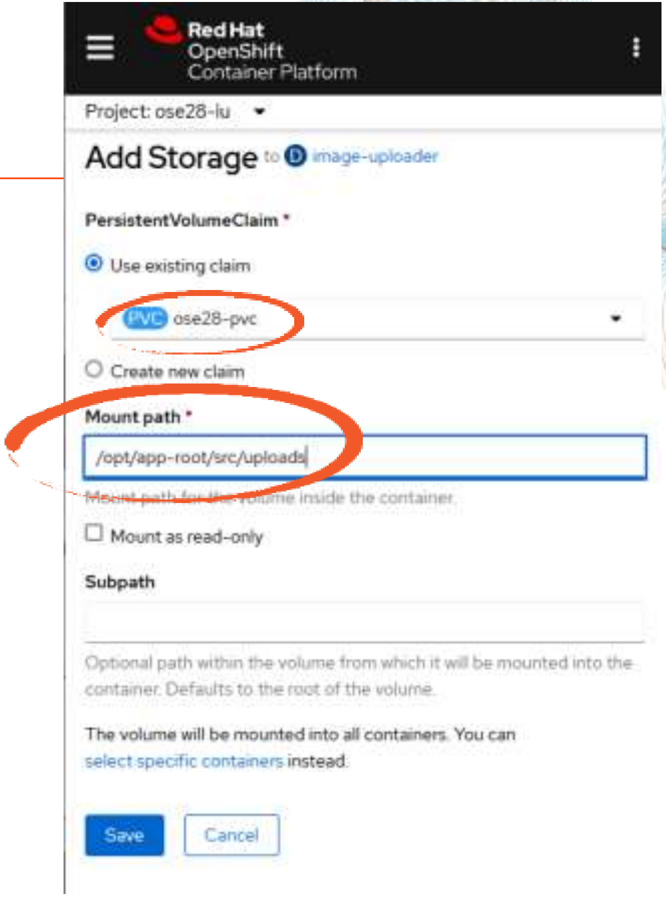
Inventory
2 Deployments
0 DeploymentConfigs
0 StatefulSets

Creating a Volume



Creating a Volume

- Just select your existing PVC
- The Mount Path is the path within each pod at which the persistent volume should be available - it is application dependent.
- On “Save” the volume is attached.



Project: ose28-lu

Add Storage to D image-uploader

PersistentVolumeClaim *

Use existing claim [✓]

PVC ose28-pvc

Create new claim

Mount path *

/opt/app-root/src/uploads

Mount path for the volume inside the container

Mount as read-only

Subpath

Optional path within the volume from which it will be mounted into the container. Defaults to the root of the volume.

The volume will be mounted into all containers. You can select specific containers instead.

neueda
futureproof your workforce

Lab 3: Add Persistent Data to the Image Uploader

- This is a quick lab to implement what we have just seen in your own project.
- Go to /labs/3-image-uploader-persistent-storage.md

neueda
futureproof your workforce

Volumes Separate Data

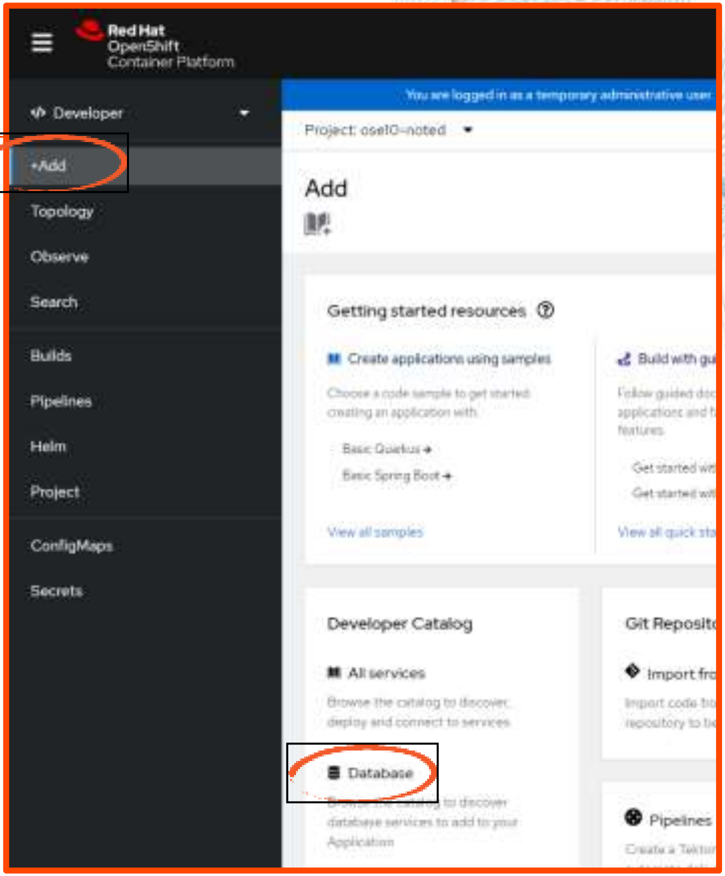
- Suppose everyone set up Volumes on our cluster - why don’t you see everyone else’s data?
    - Each application deployment uses its own NFS volume to store data.
    - Each NFS volume is then mounted into its own application’s mount space.
    - So the data is always separated on the cluster.

Data Persistence Using Databases

- The Persistent Volume technology is only one way to data persistence.
- You can also hook your application to a database
  - o If the database is not running on your cluster, then it supplies its own data persistence
  - o If the database is running on your cluster, then data persistence is achieved behind the scenes.

OpenShift Databases

- OpenShift has pre-configured MySQL, MariaDB and PostgreSQL databases available in the Developer Catalogue



You are logged in as a temporary administrative user

Choose a code sample to get started creating an application with

Follow guided doc applications and f features

Browse the catalog to discover, deploy and connect to services

Import code fro repository to be

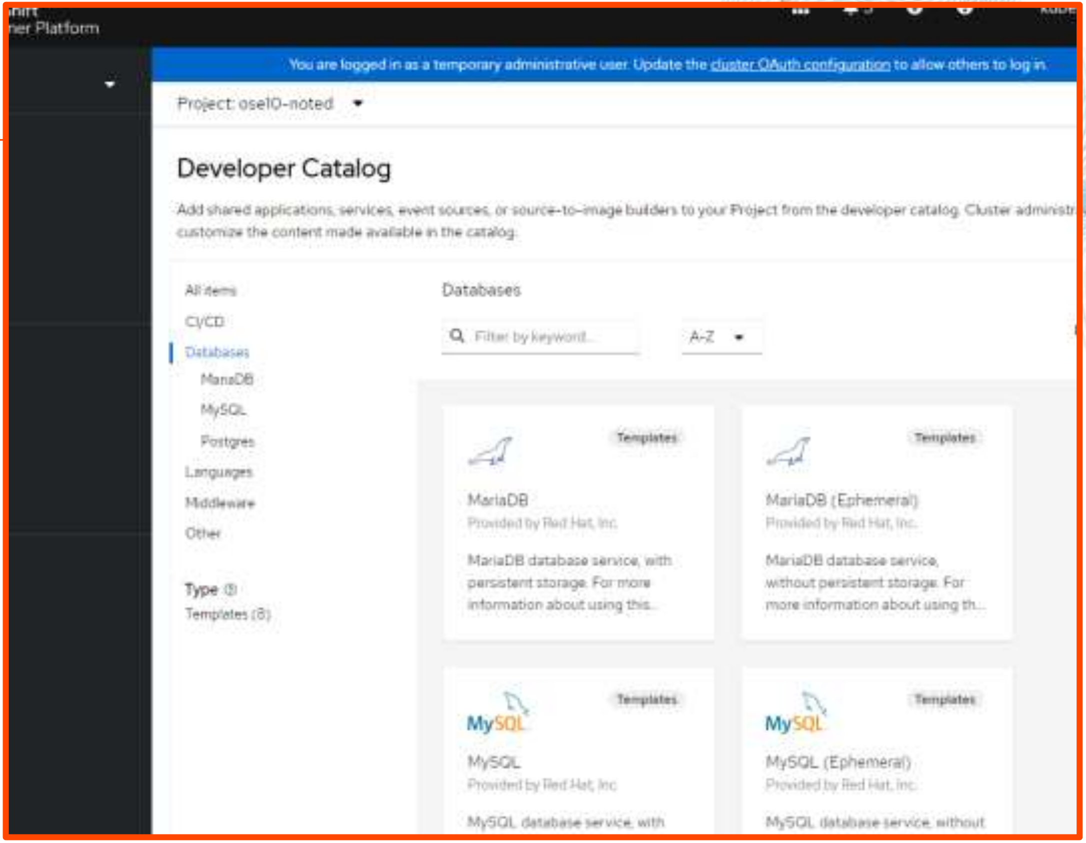
Browse the catalog to discover database services to add to your Application

Create a Tekton

neueda
futureproof your workforce

OpenShift Databases

- The databases all offer ephemeral (for testing) and persistent options.



ner Platform

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: ose10-noted

Developer Catalog

Add shared applications, services, event sources, or source-to-image builders to your Project from the developer catalog. Cluster administr customize the content made available in the catalog.

MariaDB
Provided by Red Hat, Inc.
MariaDB database service, with persistent storage. For more information about using this...

MariaDB (Ephemeral)
Provided by Red Hat, Inc.
MariaDB database service, without persistent storage. For more information about using th...

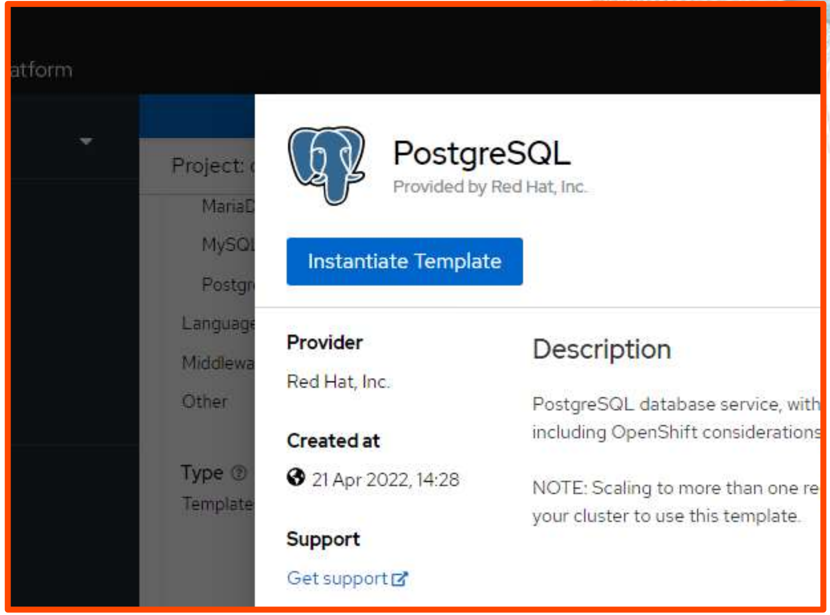
MySQL
Provided by Red Hat, Inc.
MySQL database service, with

MySQL (Ephemeral)
Provided by Red Hat, Inc.
MySQL database service, without

neueda
futureproof your workforce

OpenShift Databases

- During Instantiation you can supply a username, password and other configuration details.
- If you use a template you will need to manually configure your app to inject the environment variables needed to connect.



PostgreSQL
Provided by Red Hat, Inc.

Provider
Red Hat, Inc.

Created at
21 Apr 2022, 14:28

Support
Get support

Description
PostgreSQL database service, with
including OpenShift considerations

NOTE: Scaling to more than one re
your cluster to use this template.

neueda
futureproof your workforce

Demo: Petclinic

- In this demo we will deploy a Spring Boot app and a MySQL database on the same cluster and link the two.

Connecting to Databases

- The database deployment we saw is quite manual, OpenShift has a number of more automatic connection methods.
- One of the simplest of these is using a Secrets object.
  - Secrets objects store one or more values that are intended to be obscured (like passwords, or certificate files).
  - Secrets objects can be made available to Pods in a namespace without divulging the contents to users.
  - We will see Secrets for connecting a database later on.

Connecting the Databases: Service Binding

- Another method for connecting to databases is through the **Service Binding Operator**
- This is one of the built-in operators your cluster admin can install
- The Service Binding Operator (SBO) allows you to quickly bind an instance of a database to an application deployed on OpenShift
  - No need to distribute secrets, config maps, usernames, passwords.
  - But the database needs to be deployed via an operator or a Helm chart configured to what the SBO needs.

neueda
futureproof your workforce

Deploying a DB for the SBO

- There are PostgreSQL Operators in the OpenShift OperatorHub, but once again you would have to configure the how the quarkus-backend connects to it manually
- We will install a *new* Operator repository (or OperatorSource)
  - This provides the OpenShift Operator Lifecycle Manager the means to install the PostgreSQL Operator we need
- This is a cluster administrator level task, but worth seeing in a demo!

neueda
futureproof your workforce

Service Binding Operator Usage

- There isn’t an interface in the console for creating a Service Binding Object, so you have to supply the YAML configuration directly.
- You can either do this through the command line, or using the “+” in the top right of your console.
- This will then make the secrets in the services specified available directly to the application.

Demo: Notes App with PostgreSQL

- Let’s go back to our demo notes app and see how we can add a database in OpenShift.

Summary

- Data Persistence in OpenShift
- Persistent Volumes using NFS
- Persistent Volume Claims
- OpenShift Volumes
- Data Persistence Using Databases
- OpenShift Databases

Questions and Comments?