

Deploy Video on Demand PoC Demo on Red Hat OpenShift Container Platform 4.11 on Ampere Altra Platform

In this tutorial, we explain how to deploy Video-on-Demand PoC demo on Red Hat OpenShift Container Platform 4.11 on Ampere Altra Platform with block storage, shared file systems, and object storage managed by Rook Ceph Operator. Estimated time to complete this tutorial: 30 minutes.

Overview

Ampere is pleased to showcase open-source Video-on-Demand Services on Red Hat OpenShift Container Platform, a hybrid cloud platform as a service built around Linux containers orchestrated and managed by Kubernetes, to exhibit the cloud native characteristics of the Ampere platform for video services:

- Red Hat OpenShift Container Platform - an enterprise-ready Kubernetes container platform with full-stack automated operations. It's a certified Kubernetes distribution with add-on functions/services.
 - A 3-node cluster is the minimum required for a highly available (HA) cluster
 - A Single Node OpenShift for Edge cloud or 5G Cell towers use cases
- Rook Ceph Operator - offers Block Storage, which is the key element for keeping metadata, videos (mp4, mov, or avi), subtitles (e.g., WebVTT, etc.), generated playlists & segment files for VOD services
- HAProxy - a Load Balancer for Kubernetes with an external Load Balancer which is an appliance acting as a reverse proxy and distributing network or application traffic across several servers.
- 2 Pods -
 - nginx-vod-module-container - An NGNIX-based container with VOD module serving as the backend for video streaming service
 - nginx-hello-container - An NGNIX Webserver container with HTML and resource files serving as a YouTube-like front-end web application

Prerequisites

- A DNS service like bind (named) supporting the Full Qualified Domain Name (FQDN, such as ocp4.hhii.amp or sno.ocp4.ampere)

- A HAProxy service as the external load balancer on the bastion node for multi-node OpenShift Cluster

Instructions

The following is the step-by-step to deploy Video on Demand PoC demo on OpenShift Container Platform 4.11

1. Login to OpenShift Web Console

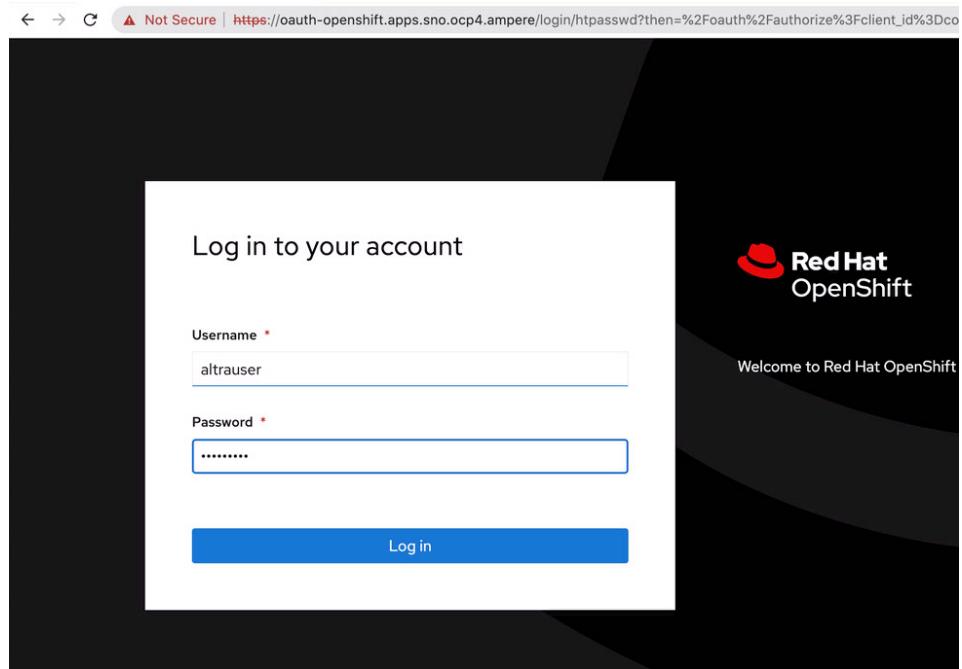


Figure A-1

2. The console will show the dashboards for activities, cluster details, and status

The screenshot shows the Red Hat OpenShift console interface. The left sidebar is titled "Administrator". The main content area is titled "Overview" and includes sections for "Cluster", "Status", and "Activity".

- Cluster:**
 - Details: Cluster API address https://api.sno.ocp4.ampere:6443, Cluster ID c0b32227-b7a4-4ffc-84f4-e89f4e3d4f6, OpenShift Cluster Manager None.
 - Infrastructure provider: None.
 - OpenShift version: 4.11.
 - Service Level Agreement (SLA): Self-support, 60 day trial, 56 days remaining. Manage subscription settings.
 - Update channel: stable-4.11.
 - Control plane high availability: No (single master).
- Status:**
 - Cluster: Green checkmark.
 - Control Plane: Single master.
 - Operators: Green checkmark.
 - Insights: Yellow warning icon, 2 issues found.
 - Dynamic Plugins: Green checkmark.
- Activity:**
 - Ongoing: There are no ongoing activities.
 - Recent events (list):
 - 6:30 PM Job completed
 - 6:30 PM Deleted job
 - 6:30 PM Saw complete
 - 6:30 PM Created container
 - 6:30 PM Started container
 - 6:30 PM Container image
 - 6:30 PM Add eth0 (IO)
 - 6:30 PM Successfully assigned
 - 6:30 PM Created job
 - 6:30 PM Created pod

Figure A-2

3. Switch to “Developer” prospect by clicking on “Administrator” pull down menu

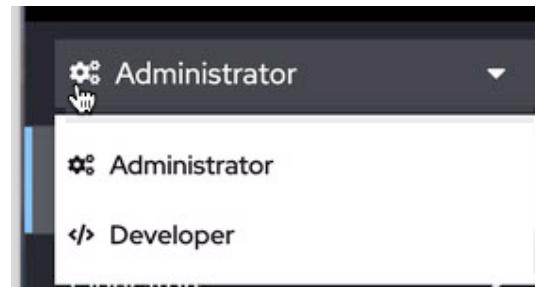


Figure A-3

4. Click “Create a project” to create a “vod-poc” project

The screenshot shows the Red Hat OpenShift console interface. The left sidebar is titled "Developer". The main content area is titled "Project: All Projects" and includes a "Topology" section and a search bar.

- Project: All Projects:**
 - Topology: Select a Project to view the topology or [create a Project](#).
- Search:**
 - Name: Name dropdown and search input field.

Figure A-4

5. Click “+Add” to start the VOD PoC demo deployment
6. Click “Import YAML” to create Persistent Volume Claims for NGNIX Web app and VOD service



Figure A-5

7. The console shows an online editor for drag-n-drop YAML or JSON file , or just simply enter the content of file and use “---” to separate each definition.

The screenshot shows the "Import YAML" editor interface. At the top, it says "Project: vod-poc". Below that is the title "Import YAML" and a sub-instruction: "Drag and drop YAML or JSON files into the editor, or manually enter files and use --- to separate each definition.". The main area contains a code editor with the following YAML content:

```

1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    name: nginx-vod-app-pvc
5    labels:
6      app: nginx-vod-app
7  spec:
8    accessModes:
9      - ReadWriteOnce
10   resources:
11     # This is the request for storage. Should be available in the cluster.
12     requests:
13       storage: 10Gi
14     storageClassName: rook-ceph-block

```

At the bottom of the editor are two buttons: "Create" and "Cancel".

Figure A-6

8. Import the YAML content for nginx-vod-app PVC

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nginx-vod-app-pvc
  labels:

```

```

app: nginx-vod-app
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    # This is the request for storage. Should be available in the cluster.
    requests:
      storage: 100Gi
  storageClassName: rook-ceph-block

```

9. Import is the YAML content for nginx-front-app PVC

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nginx-front-app-pvc
  labels:
    app: nginx-front-app
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    # This is the request for storage. Should be available in the cluster.
    requests:
      storage: 100Gi
  storageClassName: rook-ceph-block

```

10. Back to “Add” page, we deploy the NGNIX WebApp and VOD service with 2 GitHub repos:

- <https://github.com/AmpereComputing/nginx-hello-container>
- <https://github.com/AmpereComputing/nginx-vod-module-container>

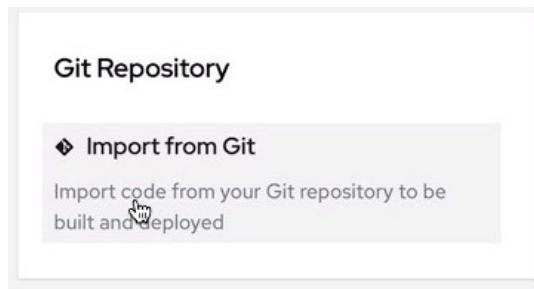


Figure A-7

11. Enter “<https://github.com/AmpereComputing/nginx-hello-container>” to Git Repo URL.

Project: vod-poc ▾ Application: All applications ▾

Import from Git

Git

Git Repo URL *

`https://github.com/AmpereComputing/nginx-hello-container`

Validated

▶ Show advanced Git options

✓ Multiple import strategies detected
The Dockerfile at Dockerfile is recommended.

 Dockerfile 

General

Application name

nginx-front-app

A unique name given to the application grouping to label your resources.

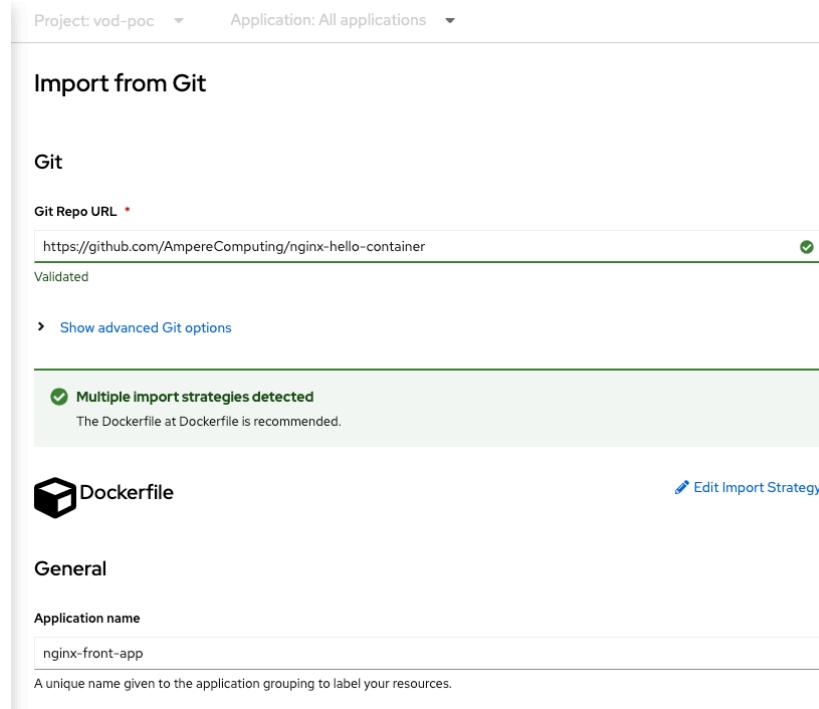


Figure A-8

12. Scroll down to enter “nginx-front-app” in “Application” and “Name” fields, then click “Create”

General

Application name

nginx-front-app

A unique name given to the application grouping to label your resources.

Name *

nginx-front-app

A unique name given to the component that will be used to name associated resources.

Resources

Select the resource type to generate

Deployment
apps/Deployment
A Deployment enables declarative updates for Pods and ReplicaSets.

DeploymentConfig

Create **Cancel**

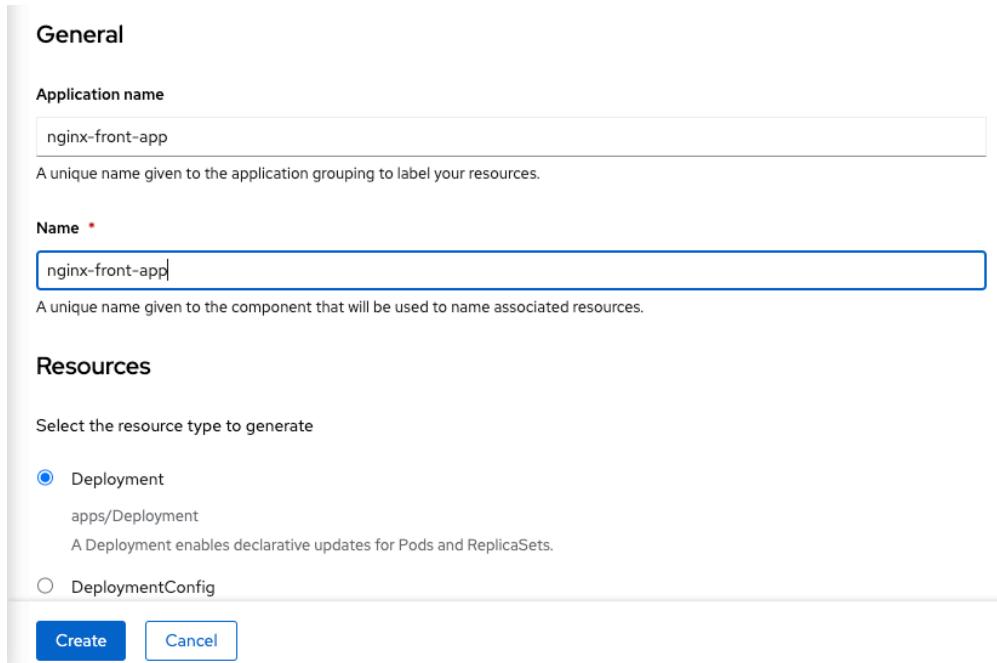
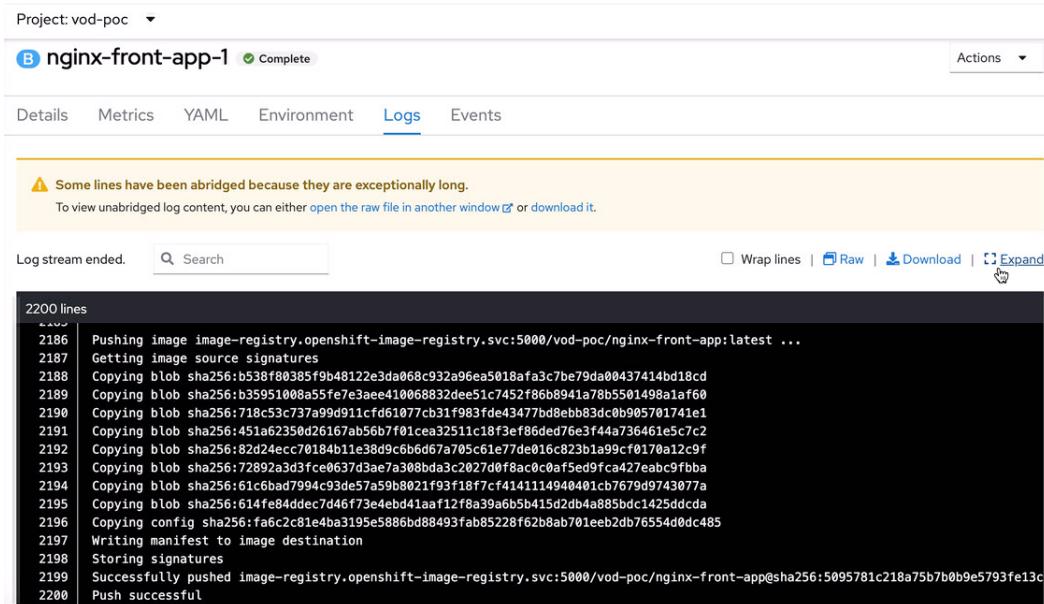


Figure A-9

13. The Source-to-Image (S2I) workflow will start to build the container image with the source code from GitHub repo



The screenshot shows the OpenShift web interface for the 'vod-poc' project. A deployment named 'nginx-front-app-1' is selected, indicated by a blue icon and the text 'Complete'. The 'Logs' tab is active. A warning message at the top states: '⚠ Some lines have been abridged because they are exceptionally long. To view unabridged log content, you can either [open the raw file in another window](#) or [download it](#)'. Below this, a search bar and several download/export options are available: 'Wrap lines', 'Raw', 'Download', and 'Expand' (which is being clicked). The log stream itself shows 2200 lines of command-line output from the build process, starting with pushing the image to the registry and ending with a successful push.

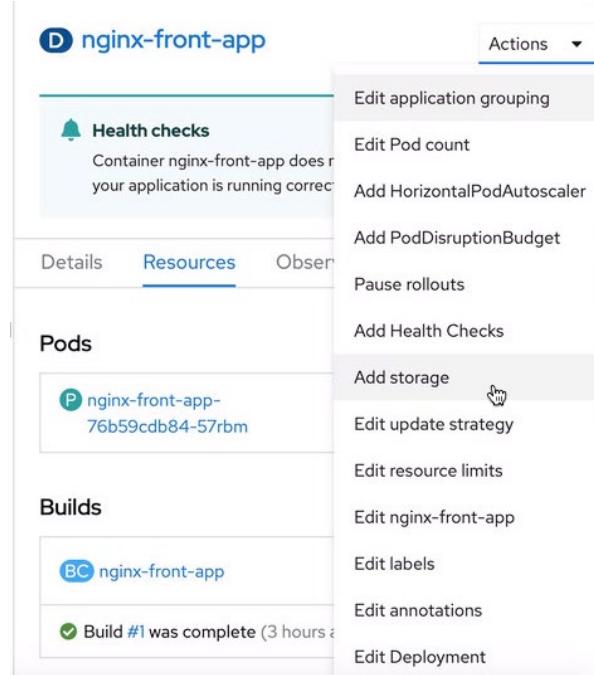
```

2286 Pushing image image-registry.openshift-image-registry.svc:5000/vod-poc/nginx-front-app:latest ...
2187 Getting image source signatures
2188 Copying blob sha256:b538f80385f9b48122e3da068c932a96ea5018afa3c7be79da00437414bd18cd
2189 Copying blob sha256:b35951008a55fe03aeed10068832dee51c7452f86b8941a7bb5501498a1af60
2190 Copying blob sha256:718c53c737a99d911cfdd61077cb31f983fde43477bd8eb83dc0b905701741e1
2191 Copying blob sha256:451a62350d26167ab56b7f01cea32511c18f3ef86ded76e3f44a736461e5c7c2
2192 Copying blob sha256:82d24ec70184b11e38d9c6b6d67a705c61e77de016c823b1a99cf0170a12c9f
2193 Copying blob sha256:72892a3d3fce0637d3ae7a308bd43c2027df8ac0c0af5ed9fc4a27eabc9fbba
2194 Copying blob sha256:f61c6bad7994c93de57a59b021f93f187fc414114940401cb7679d743077a
2195 Copying blob sha256:614fe84ddcc7d46f73e4ebd41aaaf12f8a39a6b5b415d2db4a885bd1425ddcda
2196 Copying config sha256:f46c2c81e4ba3195e5886bd88493fab85228f62b8ab701eeb2db76554d0dc485
2197 Writing manifest to image destination
2198 Storing signatures
2199 Successfully pushed image-registry.openshift-image-registry.svc:5000/vod-poc/nginx-front-app@sha256:5095781c218a75b7b0b9e5793fe13c
2200 Push successful

```

Figure A-10

14. Click “Actions” of nginx-front-app deployment, then “Add Storage” to mount the PVC, nginx-front-app-pvc, to the Pod



The screenshot shows the OpenShift web interface for the 'nginx-front-app' deployment. The 'Actions' dropdown menu is open, listing various management options. The 'Add storage' option is highlighted with a mouse cursor. Other visible options include 'Edit application grouping', 'Edit Pod count', 'Add HorizontalPodAutoscaler', 'Add PodDisruptionBudget', 'Pause rollouts', 'Add Health Checks', 'Edit update strategy', 'Edit resource limits', 'Edit nginx-front-app', 'Edit labels', 'Edit annotations', and 'Edit Deployment'. The main panel displays health checks, details, resources, and builds for the deployment.

Figure A-11

15. On “Add Storage” page for nginx-front-app, click “Use existing claim” for PersistentVolumeClaim , choose “nginx-front-app-pvc”, then enter “/usr/share/nginx/html” to the field under “Mount path” and click “save” button to complete the action.

Project: vod-poc ▾

Add Storage to D nginx-front-app

PersistentVolumeClaim *

Use existing claim
 Create new claim

Mount path *

/usr/share/nginx/htm|

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.

Figure A-12

16. Enter “<https://github.com/AmpereComputing/nginx-vod-module-container>” into “Git Repo URL”.

Project: vod-poc ▾ Application: All applications ▾

Import from Git

Git

Git Repo URL *

`https://github.com/AmpereComputing/nginx-vod-module-container` ✓

Validated

➤ Show advanced Git options

✓ **Multiple import strategies detected**
The Dockerfile at Dockerfile is recommended.

 Dockerfile 

General

Application

Create application ▾

Select an Application to group this component.

Figure A-13

17. Scroll down to enter “nginx-vod-app” in “Application” and “Name” fields, then click “Create”

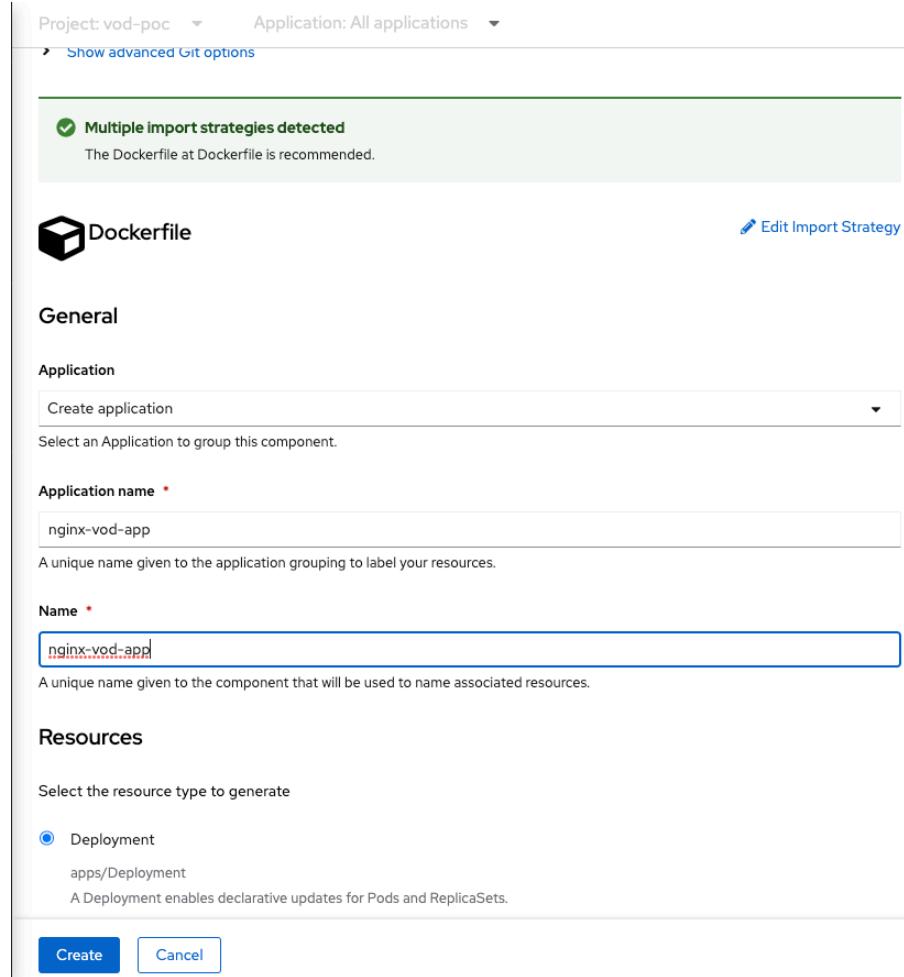


Figure A-14

18. S2I workflow also builds the container image for nginx-vod-app

```

Project: vod-poc ▾ Application: All applications ▾
Builds > Build details
B nginx-vod-app-1 ⚡ Complete Actions ▾
Details Metrics YAML Environment Logs Events
Log stream ended. Search Wrap lines | Raw | Download | Expand
1351 lines
1337 1337 [root@172-31-1-10 ~]# oc get build nginx-vod-app-1 -n vod-poc -o yaml --export=yaml
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351

```

Figure A-15

19. Once its S2I workflow is done, click “Actions” of nginx-vod-app deployment, then “Add Storage” to mount the PVC, nginx-vod-app-pvc, to the Pod

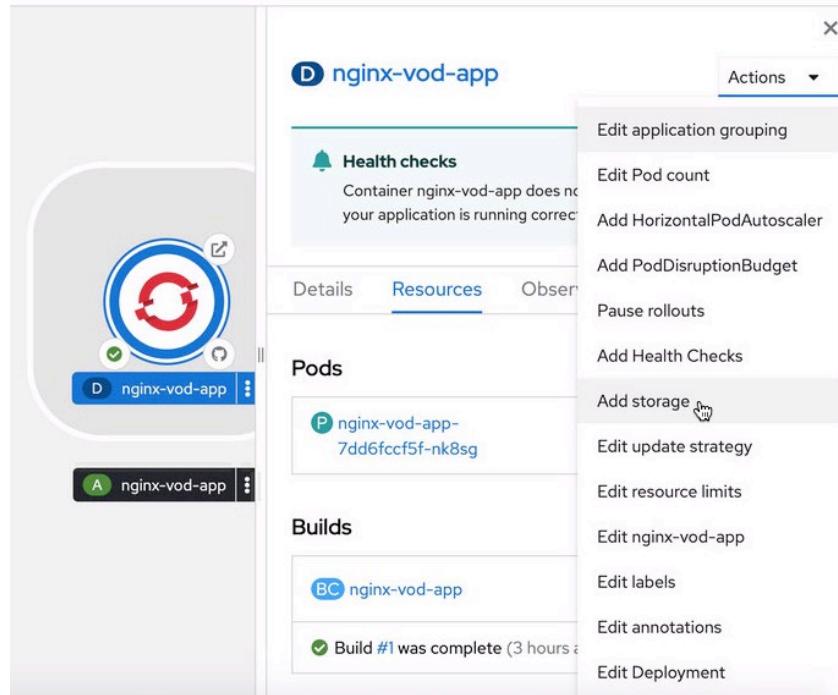


Figure A-16

20. On “Add Storage” page for nginx-vod-app, click “Use existing claim” for PersistentVolumeClaim, choose “nginx-vod-app-pvc”, then enter “/opt/static/videos” to the field under “Mount path” and click “save” button to complete the action.

This screenshot shows the 'Add Storage' configuration page for the 'nginx-vod-app' deployment. At the top, it says 'Project: vod-poc'. The main section is titled 'Add Storage to D nginx-vod-app'. Under 'PersistentVolumeClaim *', there are two options: 'Use existing claim' (which is selected) and 'Create new claim'. A dropdown menu shows a single item: 'PVC nginx-vod-app-pvc'. Below that is the 'Mount path *' field, which contains the value '/opt/static/videos'. There is also a note: 'Mount path for the volume inside the container.' and a checkbox for 'Mount as read-only'. At the bottom, there is a 'Subpath' field and a note: 'Optional path within the volume from which it will be mounted into the container. Defaults to the root of the volume.'

Figure A-17

21. “Topology” page shows two Pods, nginx-front-app and nginx-vod-app

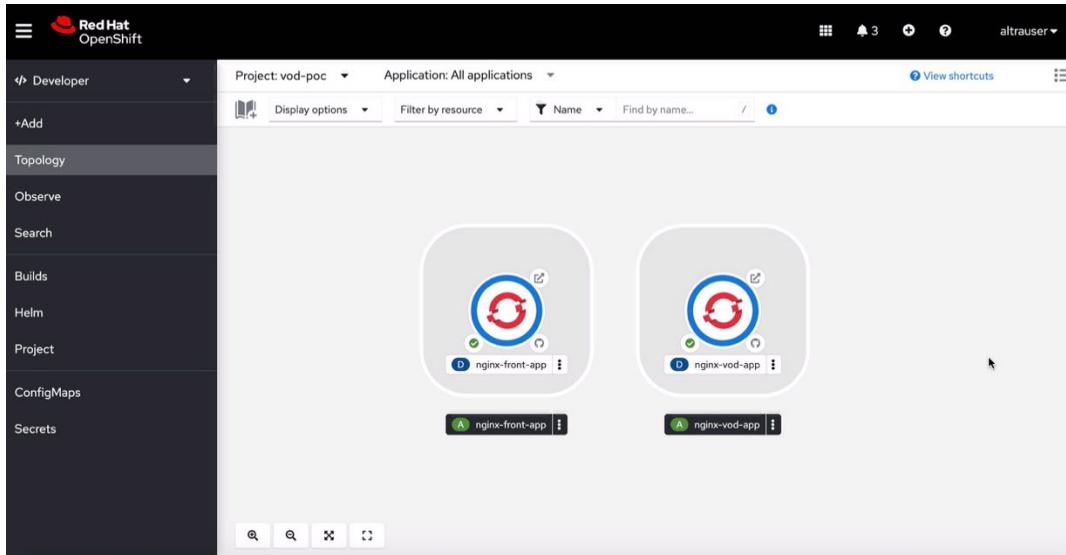


Figure A-18

22. OpenShift console also allows developers to access the Pod via the terminal tab

- Click the link for nginx-front-app-[xyz-123] under Pods, then it will prompt the Pod details page

A screenshot of the Red Hat OpenShift web console showing the details for the 'nginx-front-app' pod. On the left, the topology view shows the same two pods as Figure A-18. On the right, a detailed view for 'nginx-front-app' is open. It shows a summary section with a bell icon for 'Health checks' (noting no health checks are defined) and tabs for 'Details', 'Resources', and 'Observe'. Under the 'Pods' section, there is a table with one row for 'nginx-front-app' (status: Running, ID: 76b59cdb84-57rbm). There is a 'View logs' button next to this row. Below the table is a 'Builds' section with a table showing one completed build (Build #1 was complete 4 hours ago) and a 'Start Build' button. Navigation icons are at the bottom of the page.

Figure A-19

- Click “Terminal” Tab, it will connect to nginx-front-app container

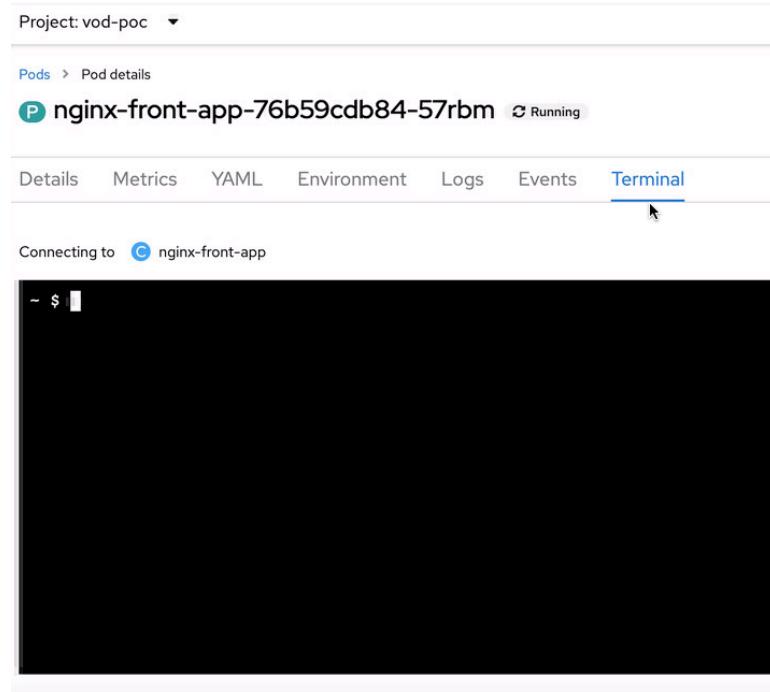


Figure A-20

- When the console is ready, execute the following commands

```
$ cd /usr/share/nginx/html/  
$ wget http://[A link of bucket on Object Storage]/hello-demo.tgz  
$ tar zxvf hello-demo.tgz  
$ mv hello-demo/* .  
$ rm -rf hello-demo
```

- Also update the URL (to <https://nginx-vod-app-vod-poc.apps.sno.ocp4.ampere>) in each video player HTML files and fix a typo

```
sed -i "s,http://\[vod-demo\]\",https://nginx-vod-app-vod-poc.apps.sno.ocp4.ampere/,g" *.html  
sed -i "s,Pytorch,PyTorch,g" *.html
```

- Continue to work on nginx-vod-app Pod and click the link for nginx-vod-app-[xyz-123] under Pods, then it will prompt the Pod details page

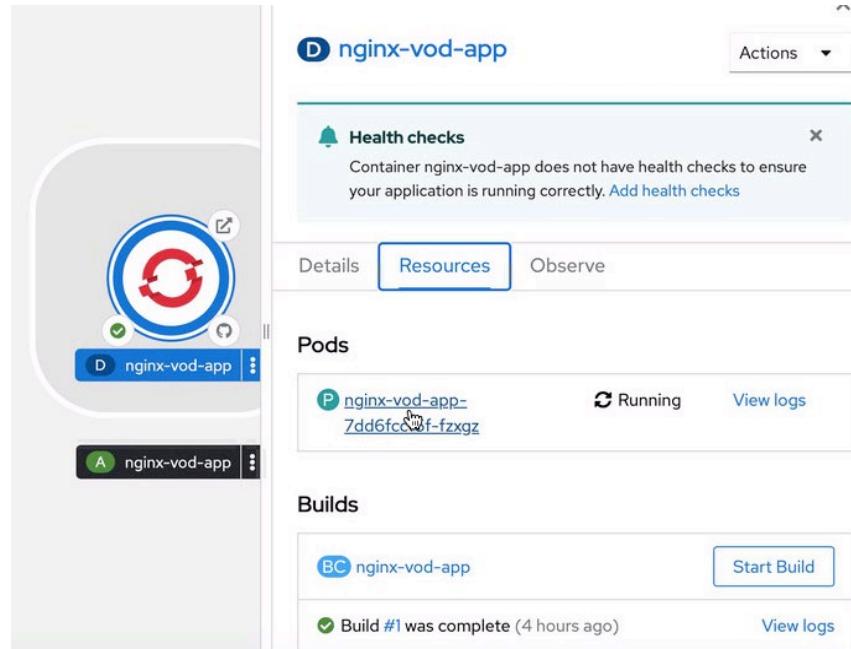


Figure A-21

- Click “Terminal” Tab, it will connect to nginx-vod-app container

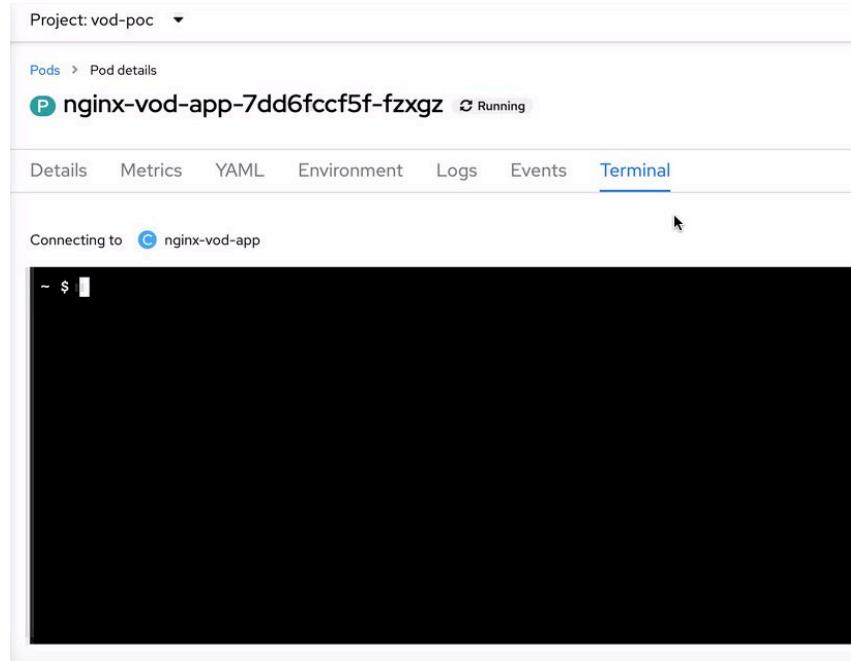


Figure A-22

- When the VoD container console is ready, execute the following commands to retrieve the tarball file included those pre-encoded videos

```
$ cd /opt/static/videos/
$ wget http://[A link of bucket on Object Storage]/vod-demo.tgz
$ tar zxvf vod-demo.tgz
$ mv vod-demo/* .
$ rm -rf vod-demo
$ sed -i "s,Pytorch,PyTorch,g" *.vtt
```

23. Since OpenShift will serve all web applications and services in HTTPS protocol, it's needed to configure NGNIX VoD module to generate master playlists to serve media playlists with HTTPS as well. Otherwise, the video player on HTML5 browsers won't be able to play the streaming videos.

- On bastion node, edit a nginx.conf file with the modified URL (<https://nginx-vod-app-vod-poc.apps.sno.ocp4.ampere>) for "vod_base_url" to create a ConfigMap for VOD service (nginx-vod-app)

```
worker_processes auto;

events {
    use epoll;
}

http {
    log_format main '$remote_addr $remote_user [$time_local] "$request" '
                   '$status "$http_referer" "$http_user_agent"';

    access_log /dev/stdout main;
    error_log stderr debug;

    default_type application/octet-stream;
    include      /var/cache/nginx/conf/mime.types;

    sendfile   on;
    tcp_nopush on;
    tcp_nodelay on;

    vod_base_url           https://nginx-vod-app-vod-
poc.apps.sno.ocp4.ampere;
    vod_mode                local;
    vod_metadata_cache      metadata_cache 16m;
    vod_response_cache      response_cache 512m;
    vod_last_modified_types  *;
    vod_segment_duration     9000;
    vod_align_segments_to_key_frames on;
    vod_dash_fragment_file_name_prefix "segment";
    vod_hls_segment_file_name_prefix "segment";

    vod_manifest_segment_durations_mode accurate;

    open_file_cache          max=1000 inactive=5m;
    open_file_cache_valid     2m;
    open_file_cache_min_uses  1;
    open_file_cache_errors    on;

    aio on;
```

```

server {
    listen 8080;
    server_name localhost;
    root /opt/static;

    location ~ ^/videos/.+$ {
        autoindex on;
    }

    location /hls/ {
        vod hls;
        alias /opt/static/videos/;
        add_header Access-Control-Allow-Headers '*';
        add_header Access-Control-Allow-Origin '*';
        add_header Access-Control-Allow-Methods 'GET, HEAD, OPTIONS';
    }

    location /thumb/ {
        vod thumb;
        alias /opt/static/videos/;
        add_header Access-Control-Allow-Headers '*';
        add_header Access-Control-Allow-Origin '*';
        add_header Access-Control-Allow-Methods 'GET, HEAD, OPTIONS';
    }

    location /dash/ {
        vod dash;
        alias /opt/static/videos/;
        add_header Access-Control-Allow-Headers '*';
        add_header Access-Control-Allow-Origin '*';
        add_header Access-Control-Allow-Methods 'GET, HEAD, OPTIONS';
    }
}
}

```

b. create configMap with nginx.conf

```
$ oc project vod-poc
$ oc create configmap nginx-conf --from-file=./nginx.conf
```

- c. Back to OpenShift Web console
- d. Edit the deployment for nginx-vod-app by adding the portions of “nginx-config-vol” under the sections of **volumes** and **volumeMounts**

```

...
spec:
  volumes:
    - name: nginx-vod-app-pvc
      persistentVolumeClaim:
        claimName: nginx-vod-app-pvc
    - name: nginx-config-vol
      configMap:
        name: nginx-conf
        defaultMode: 420
...

```

```

volumeMounts:
  - name: nginx-vod-app-pvc
    mountPath: /opt/static/videos
  - name: nginx-config-vol
    readOnly: true
    mountPath: /var/cache/nginx/conf/nginx.conf
    subPath: nginx.conf

```

- e. The nginx-vod-app will be re-deployed with the new nginx.conf to let NGNIX VoD module serve playlists, segments, and subtitles on HTTPS protocol.

24. Click the link icon on the nginx-front-app Pod

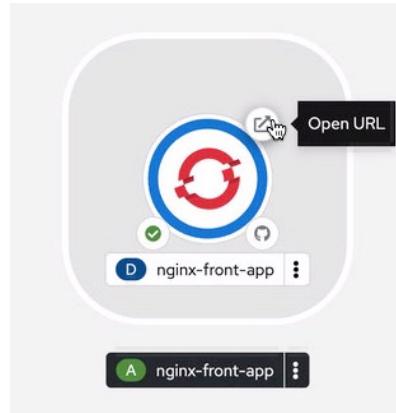


Figure A-23

25. It will trigger a new browser tab to access NGNIX Web App <https://nginx-front-app-vod-poc.apps.sno.ocp4.ampere>

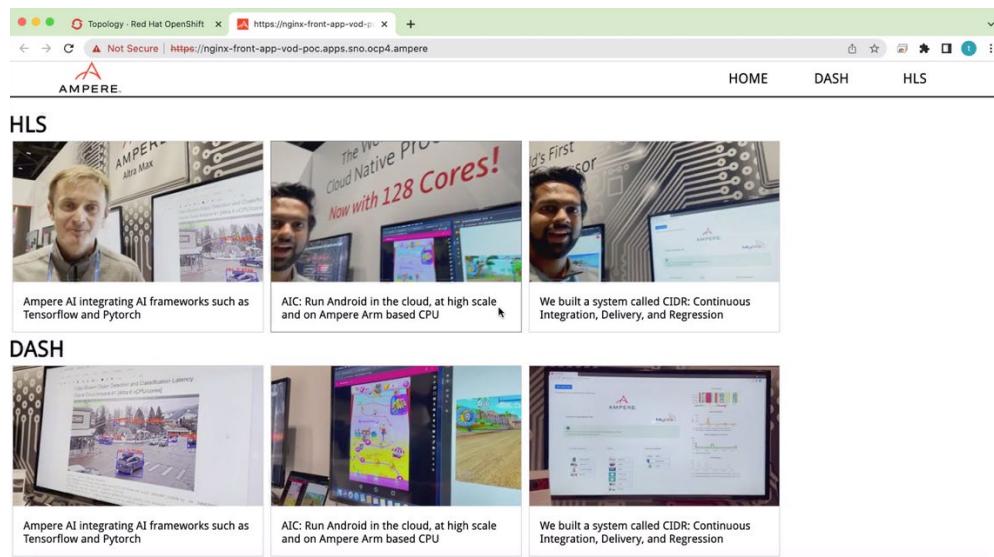


Figure A-24

26. The demo is ready for serving Video-on-Demand Streaming to audience

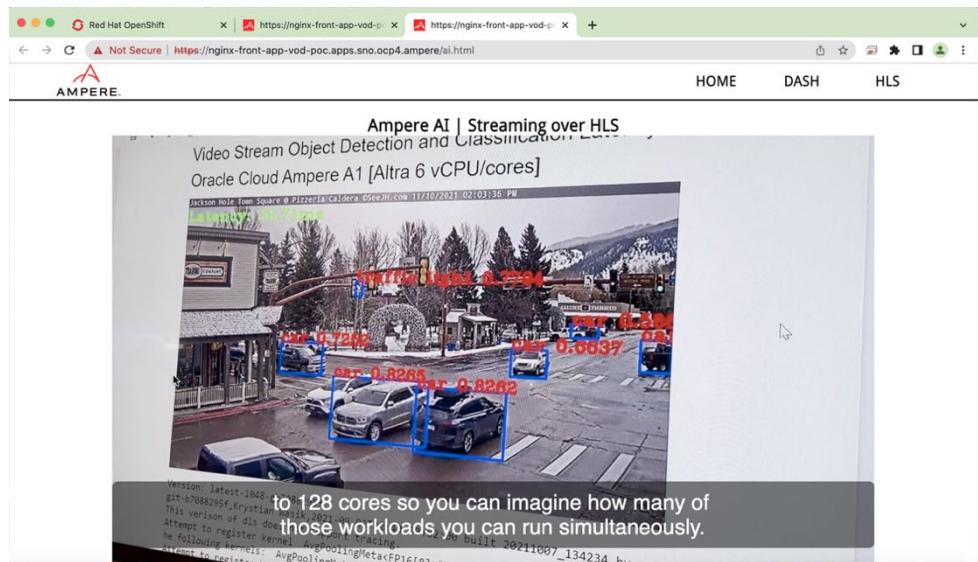


Figure 1A-25