



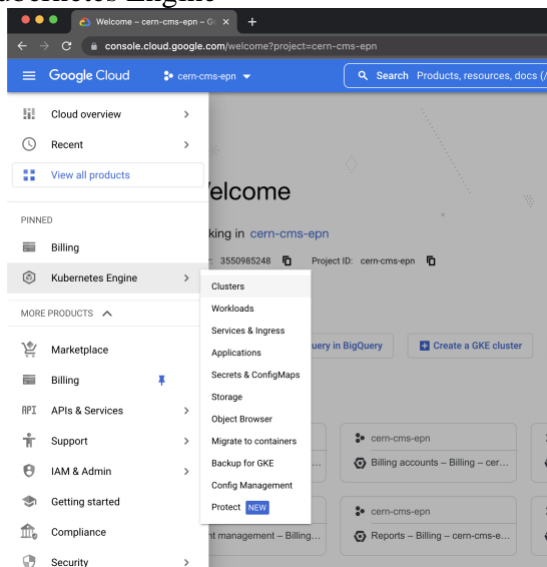
# CMS Workshop 2022 Cloud Computing

## Setup

Before you begin

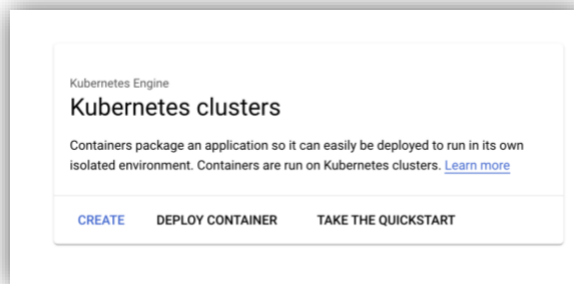
1. **Prep-work: Kubernetes Clusters (2021)**
2. **Demo: Creating a cluster (2021)**

Select Cluster in Kubernetes Engine



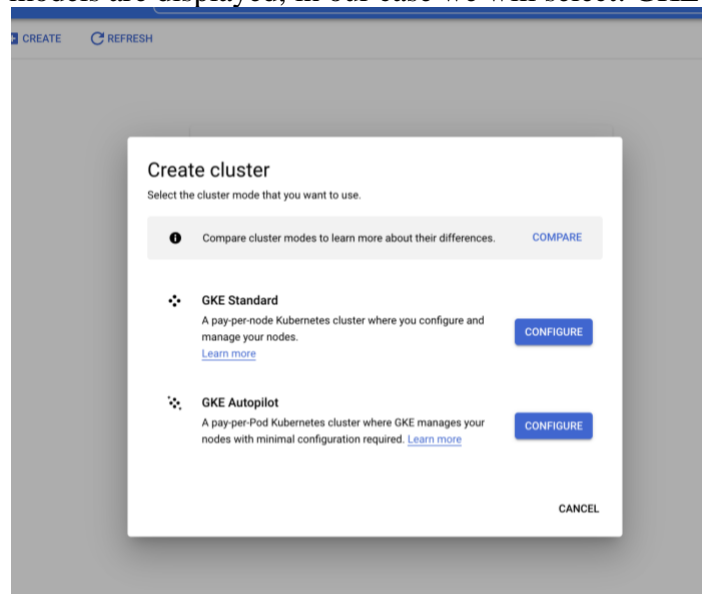
*Fig. 1 Select cluster*

This interface will appear and then we select CREATE



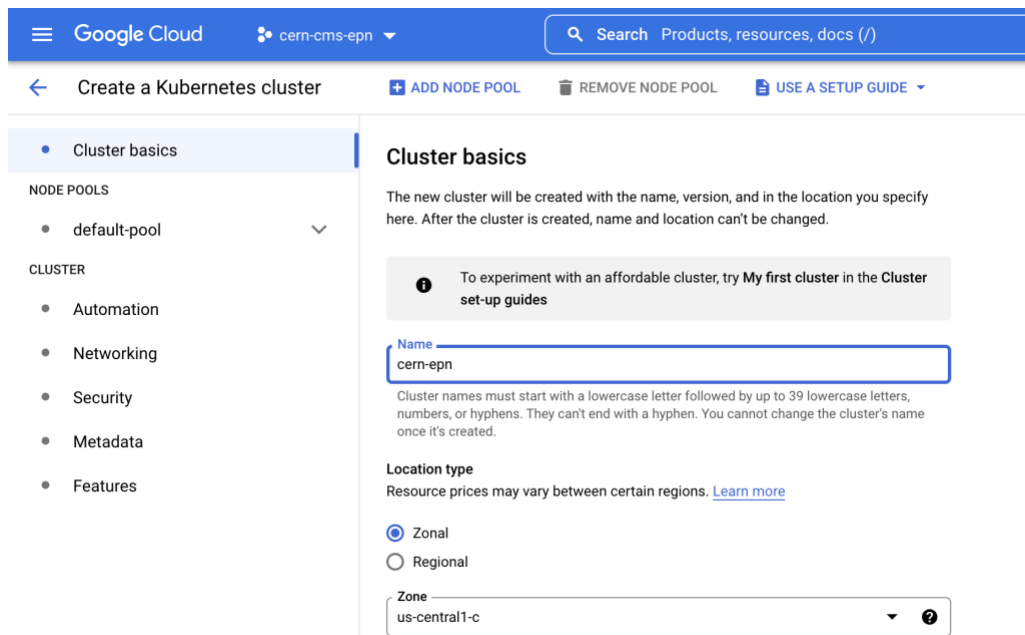
*Fig 2. Select Create*

Some cluster models are displayed, in our case we will select: GKE STANDARD



*Fig. 3 Select GKE Standard*

Give it a name



*Fig. 4 Basic Cluster Name*

- Many ways to configure the cluster, but let's try an efficient one with autoscaling
- Go to default pool
- Choose size: 1 node
- Autoscaling 0 to 4

← Create a Kubernetes cluster    + ADD NODE POOL    REMOVE NODE POOL    USE A SETUP GUIDE ▾

- Cluster basics
- NODE POOLS**
  - default-pool
    - Nodes
    - Security
    - Metadata
- CLUSTER
  - Automation
  - Networking
  - Security
  - Metadata
  - Features

### Node pool details

A node pool is a template for groups of nodes created in this cluster. The new cluster will be created with at least one node pool. More node pools can be added and removed after cluster creation. [Learn more](#)

Name: default-pool

Node pool names must start with a lowercase letter followed by up to 39 lowercase letters, numbers, or hyphens. They can't end with a hyphen. You cannot change the node pool's name once it's created.

Control plane version - 1.22.8-gke.202

Size

Number of nodes \* 1

Pod address range limits the maximum size of the cluster. [Learn more](#)

☒ Enable cluster autoscaler  
Cluster autoscaler automatically creates or deletes nodes based on workload needs. [Learn more](#)

Minimum number of nodes \* 0

Maximum number of nodes \* 4

☐ Specify node locations ?  
Default: us-central1-c

Fig. 5 Cluster details

- Go to Nodes
- Choose a machine e2-standar-4
- Leave the rest as it is
- Hit create

← Create a Kubernetes cluster    + ADD NODE POOL    REMOVE NODE POOL    USE A SETUP GUIDE ▾

- Cluster basics
- NODE POOLS**
  - default-pool
    - Nodes**
    - Security
    - Metadata
- CLUSTER
  - Automation
  - Networking
  - Security
  - Metadata
  - Features

containerd. For Windows node pools using version 1.21 or later, containerd is also the recommended runtime. Since Dockershim is being deprecated by Kubernetes project, [GKE will deprecate Docker node images](#). We recommend that you [migrate to containerd node images](#) as soon as possible. [Learn more](#).

### Machine configuration

Choose the machine family, type, and series that will best fit the resource needs of your cluster. You won't be able to change the machine type for this cluster once it's created. [Learn more](#)

Machine family

GENERAL-PURPOSE    COMPUTE-OPTIMIZED    MEMORY-OPTIMIZED    GPU

Machine types for common workloads, optimized for cost and flexibility

Series: E2

CPU platform selection based on availability

Machine type: Standard

- e2-standard-2  
2 vCPU, 8 GB memory
- e2-standard-4  
4 vCPU, 16 GB memory**
- e2-standard-8  
8 vCPU, 32 GB memory
- e2-standard-16  
16 vCPU, 64 GB memory
- e2-standard-32  
32 vCPU, 128 GB memory

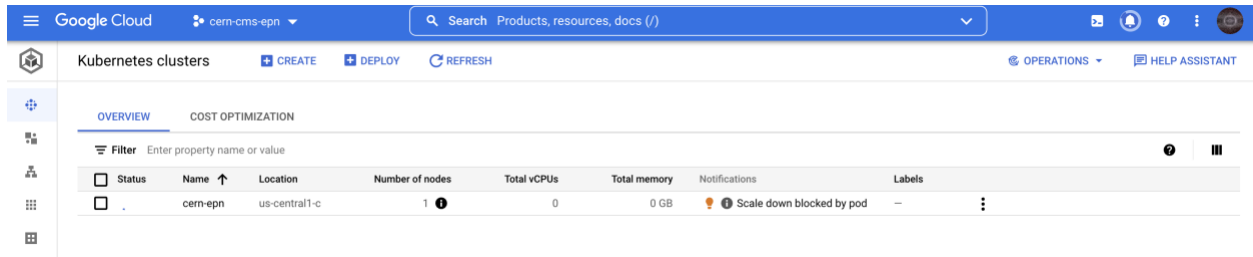
☐ Enable customer-managed encryption for boot disk ?

Local SSD disks ?

CREATE    CANCEL    Equivalent    REST    or    COMMAND LINE

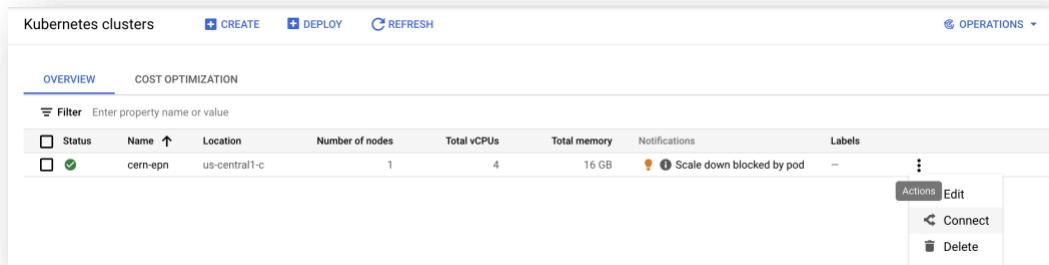
Fig. 6 Machine Configuration

Wait till it's ready (Around 5 min)



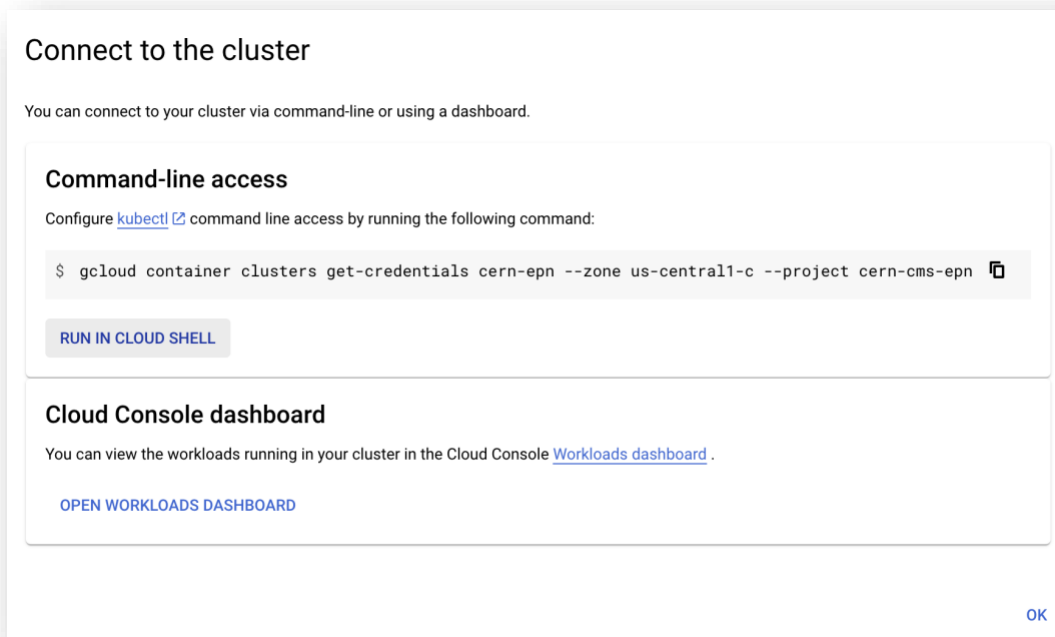
*Fig. 7 Creating*

Select the connect action



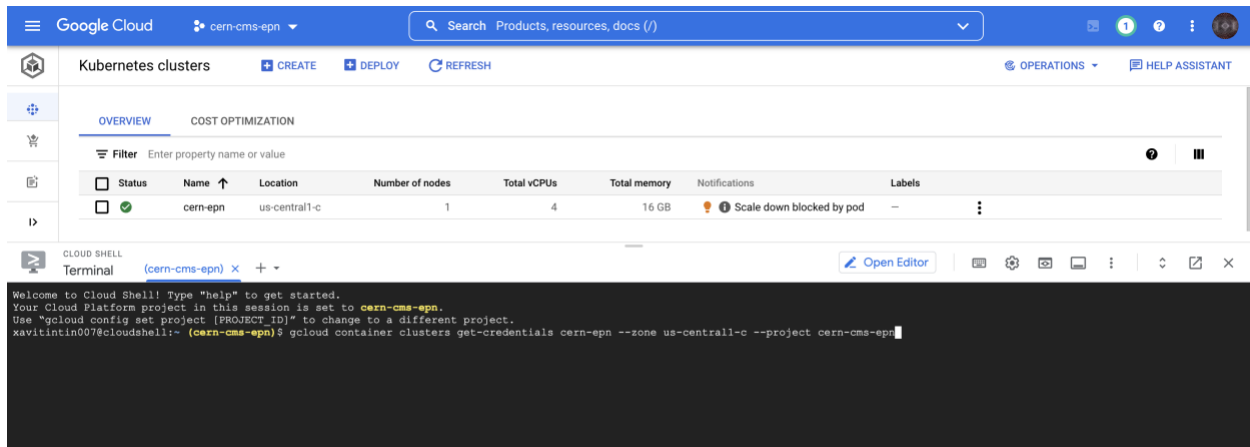
*Fig. 8 Connection*

Run cloud in shell



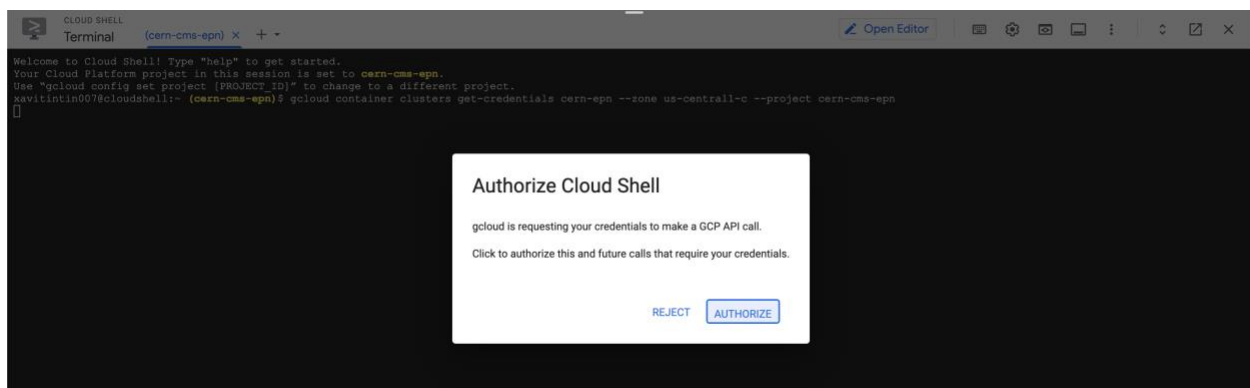
*Fig. 9 Command-line*

You've connected to your shell, now press enter to link to your GKE cluster



*Fig. 10 Load the command*

## Authorize Shell



*Fig. 11 Some Credentials*

## 3. Kubectl and additional tools and services (2021)

### Argo

While jobs can also be run manually, a workflow engine makes defining and submitting jobs easier. In this tutorial, we use [argo quick start](#) page to install it:

```
kubectl create ns argo
kubectl apply -n argo -f https://raw.githubusercontent.com/argoproj/argo-
workflows/master/manifests/quick-start-postgres.yaml
curl -sLO https://github.com/argoproj/argo/releases/download/v2.11.1/argo-
linux-amd64.gz
gunzip argo-linux-amd64.gz
chmod +x argo-linux-amd64
sudo mv ./argo-linux-amd64 /usr/local/bin/argo
```

```
kubectl create clusterrolebinding cern-cms-cluster-admin-binding --  
clusterrole=cluster-admin --user=cms-gXXX@arkivum.com
```

argo version

A terminal window titled 'Terminal' with a tab '(cern-cms-eqn)'. The prompt is 'xavitintin007@cloudshell:~ (cern-cms-eqn)'. The command 'argo version' has been executed, and the output is displayed. The output includes the argo version (v2.11.1), build date (2020-09-29T17:25:25Z), git commit (13b51d569d580ab9493e977fe2944889784d2a0a), git tree state (clean), git tag (v2.11.1), go version (go1.13.15), compiler (gc), and platform (linux/amd64).

```
xavitintin007@cloudshell:~ (cern-cms-eqn)$ argo version  
argo: v2.11.1  
BuildDate: 2020-09-29T17:25:25Z  
GitCommit: 13b51d569d580ab9493e977fe2944889784d2a0a  
GitTreeState: clean  
GitTag: v2.11.1  
GoVersion: go1.13.15  
Compiler: gc  
Platform: linux/amd64  
xavitintin007@cloudshell:~ (cern-cms-eqn)$
```

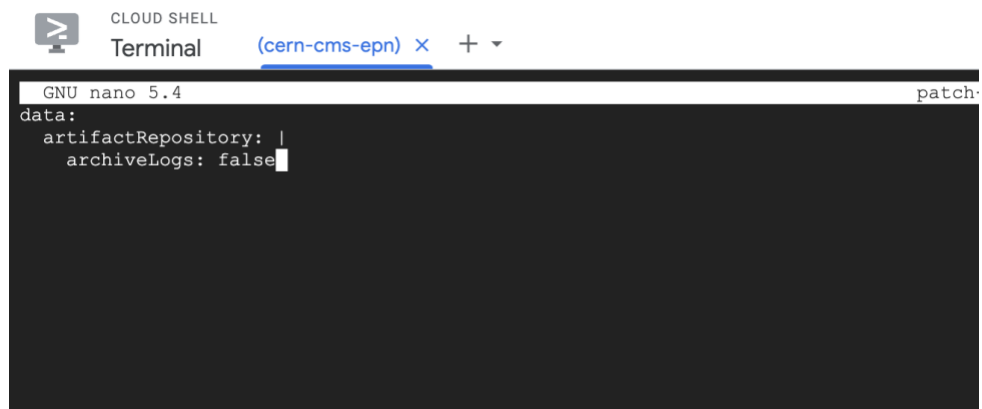
*Fig. 12 Check Version*

Patch a yaml file with the following command:

```
nano patch-workflow-controller-configmap.yaml
```

Paste this inside:

```
data:  
  artifactRepository: |  
    archiveLogs: false
```

A terminal window showing the nano text editor. The title bar says 'GNU nano 5.4' and the file name 'patch-workflow-controller-configmap.yaml' is visible on the right. The editor shows the following content:

```
data:  
  artifactRepository: |  
    archiveLogs: false
```

```
GNU nano 5.4 patch-workflow-controller-configmap.yaml  
data:  
  artifactRepository: |  
    archiveLogs: false
```

*Fig. 13 Inside NANO*

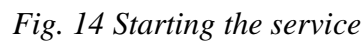
To exit press ^X Y Enter

Patch with the following command:

## Run a simple test workflow

To test the setup, run a simple test workflow with

Wait till the yellow light turns green



installed  
have the



To delete the latest argo workflow use:

```
argo delete -n argo @latest
```

#### 4. Storing workflow output on Google Kubernetes Engine (2020)

We will create the volume disk, in my example my <NUMBER>=1

```
gcloud compute disks create --size=100GB --zone=us-central1-c gce-nfs-disk-  
<NUMBER>
```

Set up an nfs server for this disk:

NOTE: If you did use a different <NUMBER> the you have to change this number in every yaml configuration file, you can do this with the following command:

*vim (file you want to change)*, for the first file it would look like this: *vim 001-nfs-server.yaml*

Press the letter “i” in your keyboard to edit text, and to exit press ^C and the text: “:wq”.

Remember to run the kubectl apply command afterwards.

```
wget https://cms-opendata-workshop.github.io/workshop2021-lesson-  
cloud/files/001-nfs-server.yaml  
kubectl apply -n argo -f 001-nfs-server.yaml
```

Set up a nfs service, so we can access the server:

```
wget https://cms-opendata-workshop.github.io/workshop2021-lesson-  
cloud/files/002-nfs-server-service.yaml  
kubectl apply -n argo -f 002-nfs-server-service.yaml
```

Let's find out the IP of the nfs server:

```
kubectl get -n argo svc nfs-server |grep ClusterIP | awk '{ print $3; }'
```

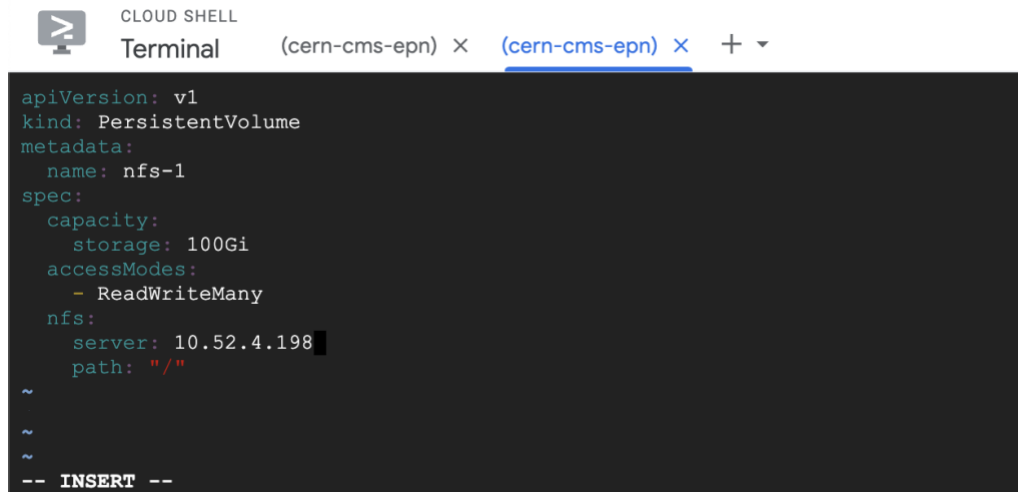
Save this ip address or open a new terminal window

We need to write that IP number above into the appropriate place in this file:

```
wget https://cms-opendata-workshop.github.io/workshop2021-lesson-  
cloud/files/003-pv.yaml  
vim 003-pv.yaml
```



To edit you have to press the letter “I” and to save and quit press ^C and enter the string: “:wq”



```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-1
spec:
  capacity:
    storage: 100Gi
  accessModes:
    - ReadWriteMany
  nfs:
    server: 10.52.4.198
    path: "/"
~
~
~
-- INSERT --
```

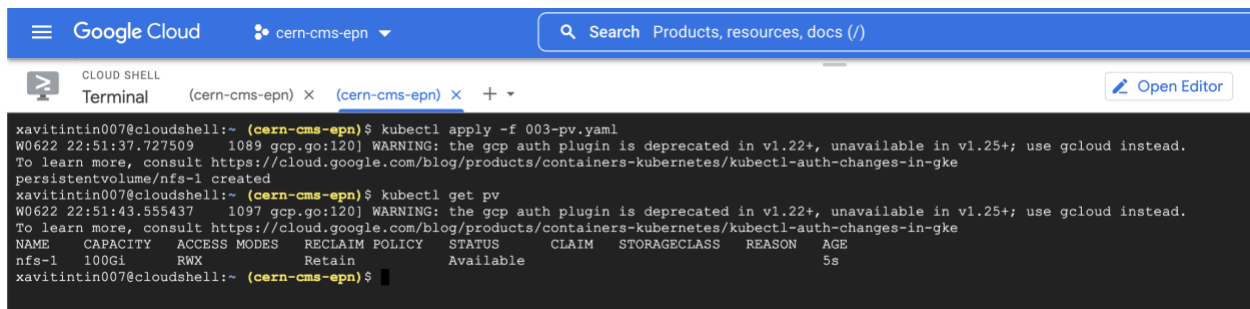
Fig. 16 Load our IP

## Deploy

```
kubectl apply -f 003-pv.yaml
```

## Check:

```
kubectl get pv
```



```
xavitintin007@cloudshell:~ (cern-cms-eprn)$ kubectl apply -f 003-pv.yaml
W0622 22:51:37.727509 1089 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
persistentvolume/nfs-1 created
xavitintin007@cloudshell:~ (cern-cms-eprn)$ kubectl get pv
W0622 22:51:43.555437 1097 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME      CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM  STORAGECLASS  REASON  AGE
nfs-1     100Gi     RWX           Retain          Available                 5s
xavitintin007@cloudshell:~ (cern-cms-eprn)$
```

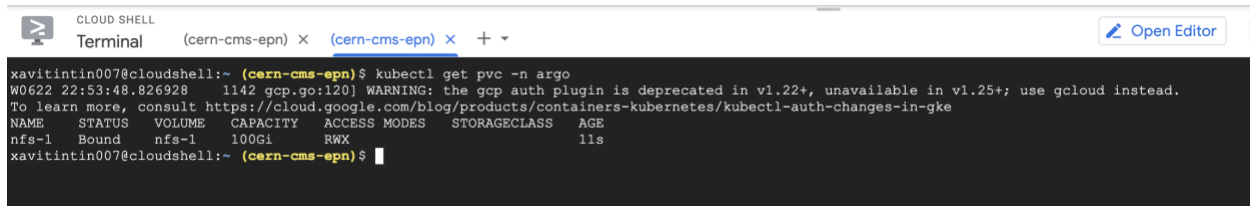
Fig. 17 Commands

Apps can claim persistent volumes through *persistent volume claims* (pvc). Let’s create a pvc:

```
wget https://cms-opendata-workshop.github.io/workshop2021-lesson-
cloud/files/003-pvc.yaml
kubectl apply -n argo -f 003-pvc.yaml
```

## Check:

```
kubectl get pvc -n argo
```



```
Cloud Shell
Terminal (cern-cms-epn) x (cern-cms-epn) x + v Open Editor

xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl get pvc -n argo
W0622 22:53:48.826928 1142 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubect1-auth-changes-in-gke
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
nfs-1     Bound    nfs-1    100Gi      RWX             nfs             11s
xavitintin007@cloudshell:~ (cern-cms-epn) $
```

*Fig. 18 Get Argo*

Now an argo workflow could claim and access this volume with a configuration like:

```
nano argo-wf-volume.yaml
```

Paste the following and change <NUMBER>, exit with ^X, Y, Enter

```
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: test-hostpath-
spec:
  entrypoint: test-hostpath
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: nfs-<NUMBER>
  templates:
    - name: test-hostpath
      script:
        image: alpine:latest
        command: [sh]
        source: |
          echo "This is the output" > /mnt/vol/test.txt
          echo ls -l /mnt/vol: `ls -l /mnt/vol`
      volumeMounts:
        - name: task-pv-storage
          mountPath: /mnt/vol
```

Submit and check this workflow with

```
argo submit -n argo argo-wf-volume.yaml
argo list -n argo
```

Take the name of the workflow from the output (replace XXXXX in the following command) and check the logs:

```
kubectl logs pod/test-hostpath-XXXXX -n argo main
```

Once the job is done, you will see something like:

```

xavitintin007@cloudshell:~ (cern-cms-epn) $ nano argo-wf-volume.yaml
xavitintin007@cloudshell:~ (cern-cms-epn) $ argo submit -n argo argo-wf-volume.yaml
Name:      test-hostpath-8vt2v
Namespace: argo
ServiceAccount: default
Status:    Pending
Created:   Wed Jun 22 23:10:38 +0000 (now)
xavitintin007@cloudshell:~ (cern-cms-epn) $ argo list -n argo
NAME                STATUS      AGE      DURATION    PRIORITY
test-hostpath-8vt2v  Succeeded   16s      10s         0
hello-world-pr8kh    Succeeded   40m       20s         0
xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl logs pod/test-hostpath-8vt2v -n argo main
W0622 23:11:27.290930 1336 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubernetes-auth-changes-in-gke
time="2022-06-22T23:10:41.632Z" level=info msg="capturing logs" argo=true
ls -l /mnt/vol: total 24 -rw-r--r-- 1 root root 16 Jun 22 22:38 index.html drwx----- 2 root root 16384 Jun 22 22:38 lost+found -rw-rw-rw- 1 root root 18 Jun 22 23:10 test.txt
time="2022-06-22T23:10:42.635Z" level=info msg="sub-process exited" argo=true error=""<nil>"
xavitintin007@cloudshell:~ (cern-cms-epn) $

```

*Fig. 19 Logs and status review*

## Run a CMS open data workflow

If the steps above are successful, we are now ready to run a workflow to process CMS open data.

Create a workflow file `argo-wf-cms.yaml` with the following content:

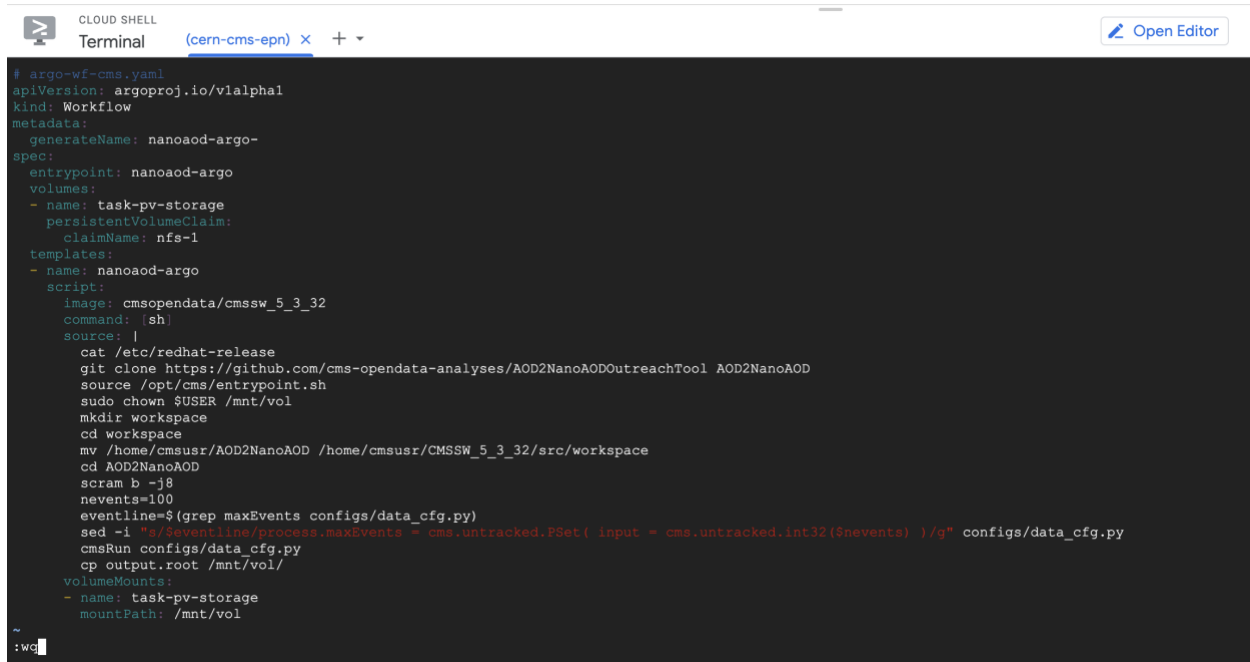
```
nano argo-wf-cms.yaml
```

```

apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: nanoaod-argo-
spec:
  entrypoint: nanoaod-argo
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: nfs-<NUMBER>
  templates:
    - name: nanoaod-argo
      script:
        image: cmsopendata/cmssw_5_3_32
        command: [sh]
        source: |
          cat /etc/redhat-release
          git clone https://github.com/cms-opendata-analyses/AOD2NanoAODOutreachTool AOD2NanoAOD
          source /opt/cms/entrypoint.sh
          sudo chown $USER /mnt/vol
          mkdir workspace
          cd workspace
          mv /home/cmsusr/AOD2NanoAOD home/cmsusr/CMSSW_5_3_32/src/workspace
          cd AOD2NanoAOD
          scram b -j8
          nevents=100
          eventline=$(grep maxEvents configs/data_cfg.py)
          sed -i "s/$eventline/process.maxEvents = cms.untracked.PSet( input = cms.untracked.int32($nevents) )/g"
          configs/data_cfg.py
          cmsRun configs/data_cfg.py
          cp output.root /mnt/vol/
        volumeMounts:
          - name: task-pv-storage

```

mountPath: /mnt/vol



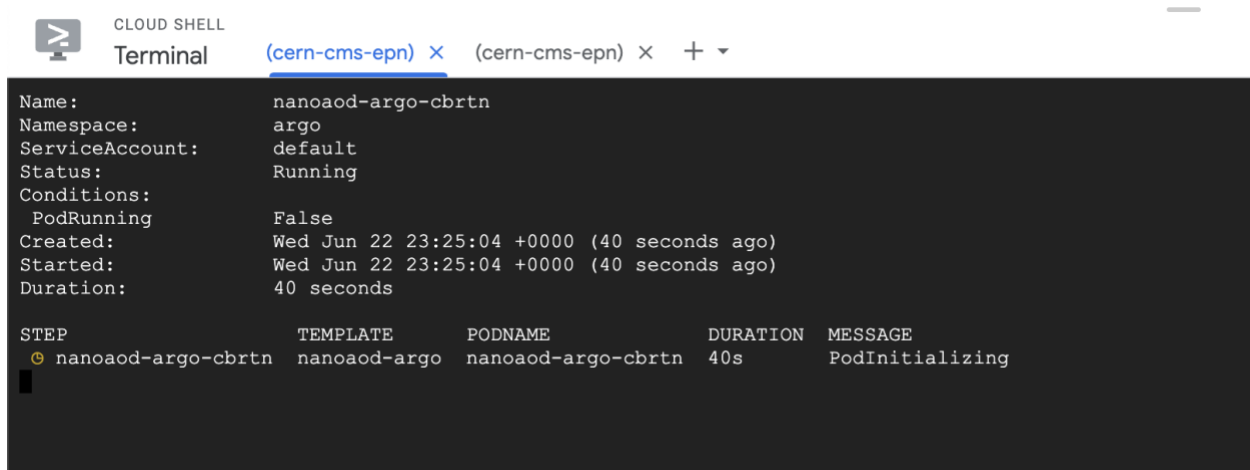
```
# argo-wf-cms.yaml
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: nanoaod-argo-
spec:
  entrypoint: nanoaod-argo
  volumes:
  - name: task-pv-storage
    persistentVolumeClaim:
      claimName: nfs-1
  templates:
  - name: nanoaod-argo
    script:
      image: cmsopendata/cmssw_5_3_32
      command: [sh]
      source: |
        cat /etc/redhat-release
        git clone https://github.com/cms-opendata-analyses/AOD2NanoAODOutreachTool AOD2NanoAOD
        source /opt/cms/entrypoint.sh
        sudo chown $USER /mnt/vol
        mkdir workspace
        cd workspace
        mv /home/cmsusr/AOD2NanoAOD /home/cmsusr/CMSSW_5_3_32/src/workspace
        cd AOD2NanoAOD
        scram b -j8
        nevents=100
        eventline=$(grep maxEvents configs/data_cfg.py)
        sed -i "s/seventline/process.maxEvents = cms.untracked.PSet( input = cms.untracked.int32($nevents) )/g" configs/data_cfg.py
        cmsRun configs/data_cfg.py
        cp output.root /mnt/vol/
      volumeMounts:
      - name: task-pv-storage
        mountPath: /mnt/vol
```

Fig. 20 Workflow file

Exit. Submit the job with:

```
argo submit -n argo argo-wf-cms.yaml --watch
```

Wait till the yellow light turns green so it is available to launch



```
Name: nanoaod-argo-cbrtn
Namespace: argo
ServiceAccount: default
Status: Running
Conditions:
  PodRunning: False
Created: Wed Jun 22 23:25:04 +0000 (40 seconds ago)
Started: Wed Jun 22 23:25:04 +0000 (40 seconds ago)
Duration: 40 seconds
```

STEP	TEMPLATE	PODNAME	DURATION	MESSAGE
🟡 nanoaod-argo-cbrtn	nanoaod-argo	nanoaod-argo-cbrtn	40s	PodInitializing

Fig. 21 Connecting

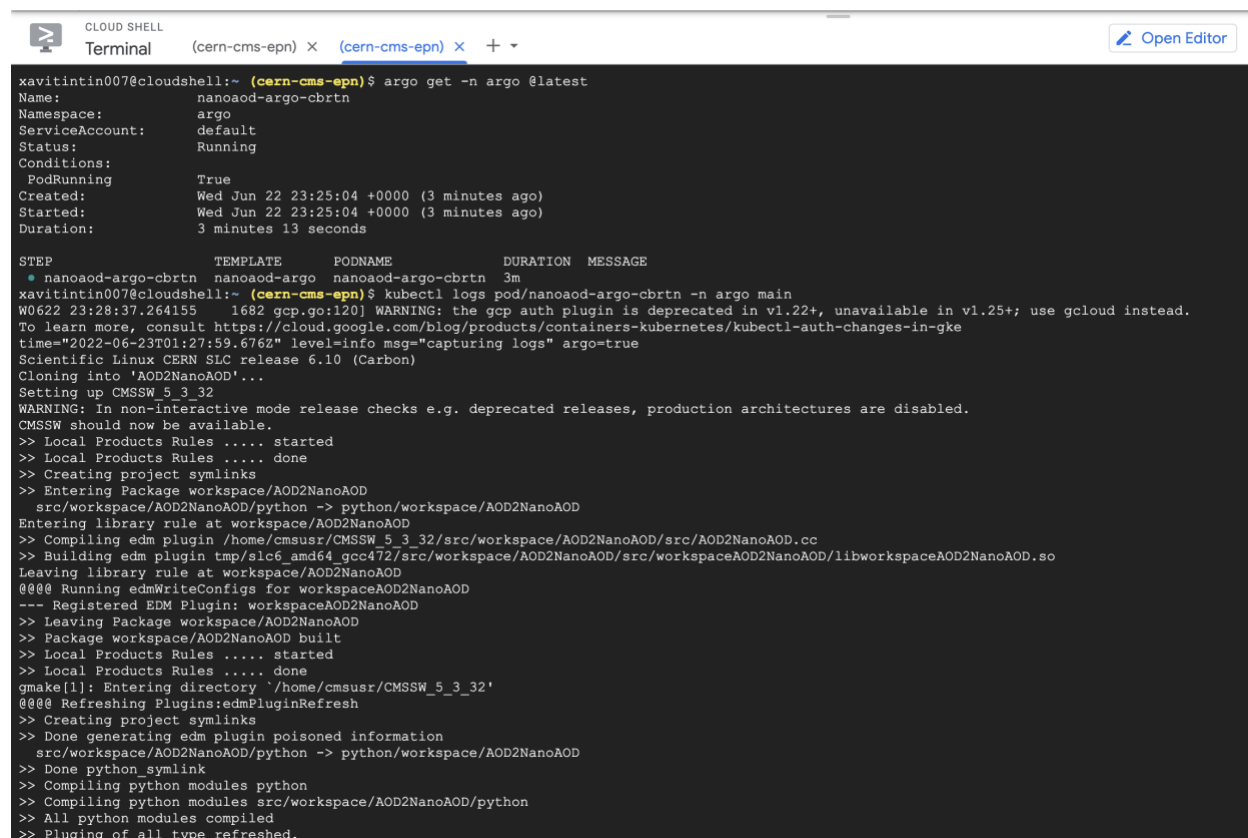
The option `--watch` gives a continuous follow-up of the progress. To get the logs of the job, use the process name (nanoaod-argo-XXXXXX) which you can also find with

```
argo get -n argo @latest
```

and follow the container logs with

```
kubectl logs pod/nanoaod-argo-XXXXX -n argo main
```

You will have to get the following:



```
CLOUD SHELL
Terminal (cern-cms-eprn) x (cern-cms-eprn) x + - Open Editor

xavitintin007@cloudshell:~ (cern-cms-eprn)$ argo get -n argo @latest
Name: nanoaod-argo-cbrtn
Namespace: argo
ServiceAccount: default
Status: Running
Conditions:
  PodRunning: True
Created: Wed Jun 22 23:25:04 +0000 (3 minutes ago)
Started: Wed Jun 22 23:25:04 +0000 (3 minutes ago)
Duration: 3 minutes 13 seconds

STEP      TEMPLATE      PODNAME      DURATION  MESSAGE
• nanoaod-argo-cbrtn nanoaod-argo nanoaod-argo-cbrtn 3m
xavitintin007@cloudshell:~ (cern-cms-eprn)$ kubectl logs pod/nanoaod-argo-cbrtn -n argo main
W0622 23:28:37.264155 1682 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectrl-auth-changes-in-gke
time="2022-06-23T01:27:59.676Z" level=info msg="capturing logs" argo=true
Scientific Linux CERN SLC release 6.10 (Carbon)
Cloning into 'AOD2NanoAOD'...
Setting up CMSSW_5_3_32
WARNING: In non-interactive mode release checks e.g. deprecated releases, production architectures are disabled.
CMSSW should now be available.
>> Local Products Rules ..... started
>> Local Products Rules ..... done
>> Creating project symlinks
>> Entering Package workspace/AOD2NanoAOD
src/workspace/AOD2NanoAOD/python -> python/workspace/AOD2NanoAOD
Entering library rule at workspace/AOD2NanoAOD
>> Compiling edm plugin /home/cmsusr/CMSSW_5_3_32/src/workspace/AOD2NanoAOD/src/AOD2NanoAOD.cc
>> Building edm plugin tmp/slc6_amd64_gcc472/src/workspace/AOD2NanoAOD/src/workspaceAOD2NanoAOD/libworkspaceAOD2NanoAOD.so
Leaving library rule at workspace/AOD2NanoAOD
@@@ Running edmWriteConfigs for workspaceAOD2NanoAOD
--- Registered EDM Plugin: workspaceAOD2NanoAOD
>> Leaving Package workspace/AOD2NanoAOD
>> Package workspace/AOD2NanoAOD built
>> Local Products Rules ..... started
>> Local Products Rules ..... done
gmake[1]: Entering directory '/home/cmsusr/CMSSW_5_3_32'
@@@ Refreshing Plugins:edmPluginRefresh
>> Creating project symlinks
>> Done generating edm plugin poisoned information
src/workspace/AOD2NanoAOD/python -> python/workspace/AOD2NanoAOD
>> Done python_symlink
>> Compiling python modules python
>> Compiling python modules src/workspace/AOD2NanoAOD/python
>> All python modules compiled
>> Plugging of all type refreshed.
```

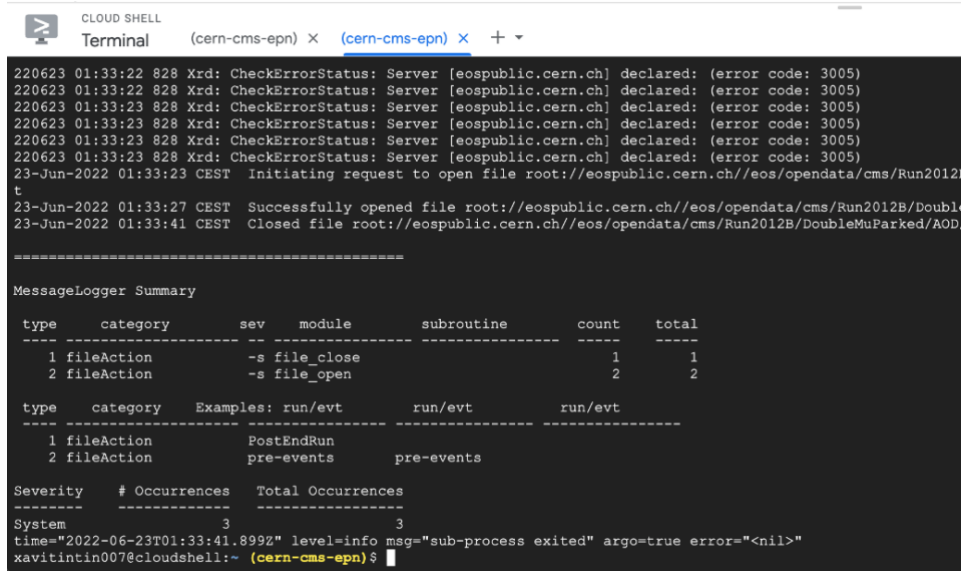
Fig. 21 Connecting

Note: There is a problem connecting with the CERN servers, it will take around 5 minutes for it to finally run.

To see the container logs re-enter the latest command:

```
kubectl logs pod/nanoaod-argo-XXXXX -n argo main
```

Finally, success!



```
CLOUD SHELL
Terminal (cern-cms-epn) x (cern-cms-epn) x + v

220623 01:33:22 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
220623 01:33:22 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
220623 01:33:23 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
220623 01:33:23 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
220623 01:33:23 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
220623 01:33:23 828 Xrd: CheckErrorStatus: Server [eospublic.cern.ch] declared: (error code: 3005)
23-Jun-2022 01:33:23 CEST Initiating request to open file root://eospublic.cern.ch/eos/opendata/cms/Run2012B/Doubl
t
23-Jun-2022 01:33:27 CEST Successfully opened file root://eospublic.cern.ch/eos/opendata/cms/Run2012B/Doubl
23-Jun-2022 01:33:41 CEST Closed file root://eospublic.cern.ch/eos/opendata/cms/Run2012B/DoubleMuParked/AOD

=====
MessageLogger Summary

type      category      sev      module      subroutine      count      total
-----
1 fileAction      -s file_close      1          1
2 fileAction      -s file_open       2          2

type      category      Examples: run/evt      run/evt      run/evt
-----
1 fileAction      PostEndRun
2 fileAction      pre-events

Severity      # Occurrences      Total Occurrences
-----
System              3                  3
time="2022-06-23T01:33:41.899Z" level=info msg="sub-process exited" argo=true error="<nil>"
xavitintin007@cloudshell:~ (cern-cms-epn) $
```

Fig. 22 Output File

## Get the output file

Create a file pv-pod.yaml with the following contents:

```
apiVersion: v1
kind: Pod
metadata:
  name: pv-pod
spec:
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: nfs-<NUMBER>
  containers:
    - name: pv-container
      image: busybox
      command: ["tail", "-f", "/dev/null"]
      volumeMounts:
        - mountPath: /mnt/data
          name: task-pv-storage
```

```
kubectl apply -f pv-pod.yaml -n argo
kubectl cp pv-pod:/mnt/data /tmp/poddata -n argo
```

and you will get the file created by the job in /tmp/poddata/test.txt.

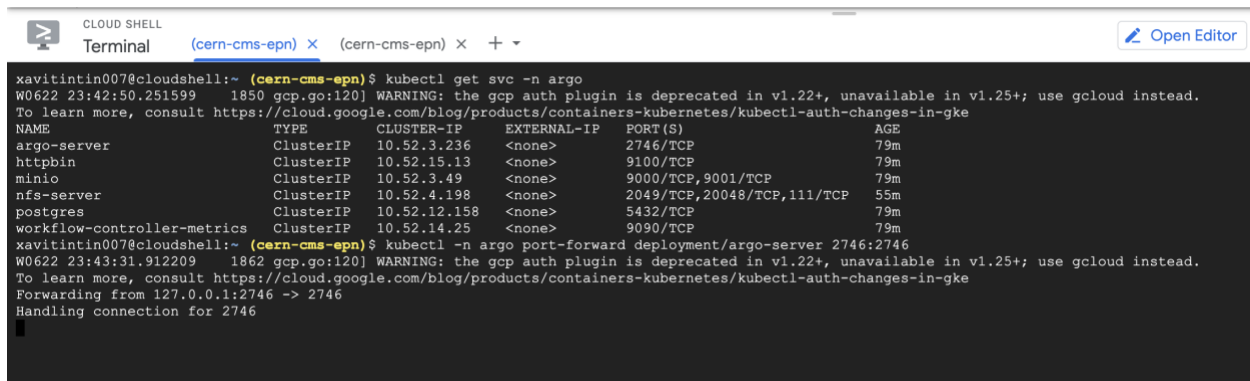
## ARGO

Check the services running and the associated IP addresses:

```
kubectl get svc -n argo
```

```
kubectl -n argo port-forward deployment/argo-server 2746:2746
```

It will start forwarding port, to not interrupt this open a new window, after a couple minutes it will be handling connection.



```
Cloud Shell
Terminal (cern-cms-epn) x (cern-cms-epn) x + v Open Editor

xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl get svc -n argo
W0622 23:42:50.251599 1850 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                TYPE                CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
argo-server          ClusterIP           10.52.3.236    <none>          2746/TCP          79m
httpbin              ClusterIP           10.52.15.13    <none>          9100/TCP          79m
minio                ClusterIP           10.52.3.49     <none>          9000/TCP,9001/TCP 79m
nfs-server            ClusterIP           10.52.4.198    <none>          2049/TCP,20048/TCP,111/TCP 55m
postgres             ClusterIP           10.52.12.158   <none>          5432/TCP          79m
workflow-controller-metrics ClusterIP           10.52.14.25    <none>          9090/TCP          79m
xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl -n argo port-forward deployment/argo-server 2746:2746
W0622 23:43:31.912209 1862 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
Forwarding from 127.0.0.1:2746 -> 2746
Handling connection for 2746
```

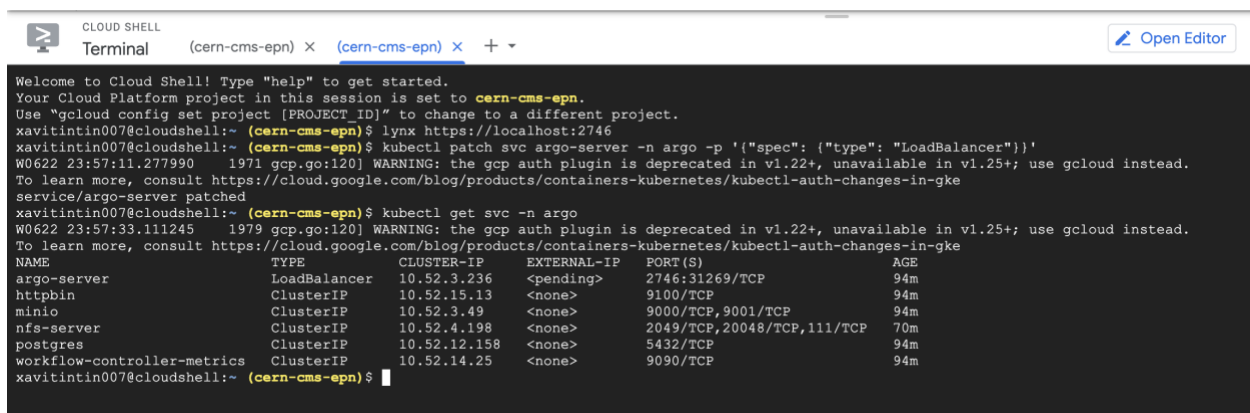
Fig. 23 Connection Stabilished

If you want to see the process in depth, in new terminal window run: `lynx https://localhost:2746`

```
kubectl patch svc argo-server -n argo -p '{"spec": {"type": "LoadBalancer"}}'
```

```
kubectl get svc -n argo
```

Creating an external ip, wait a couple minutes.

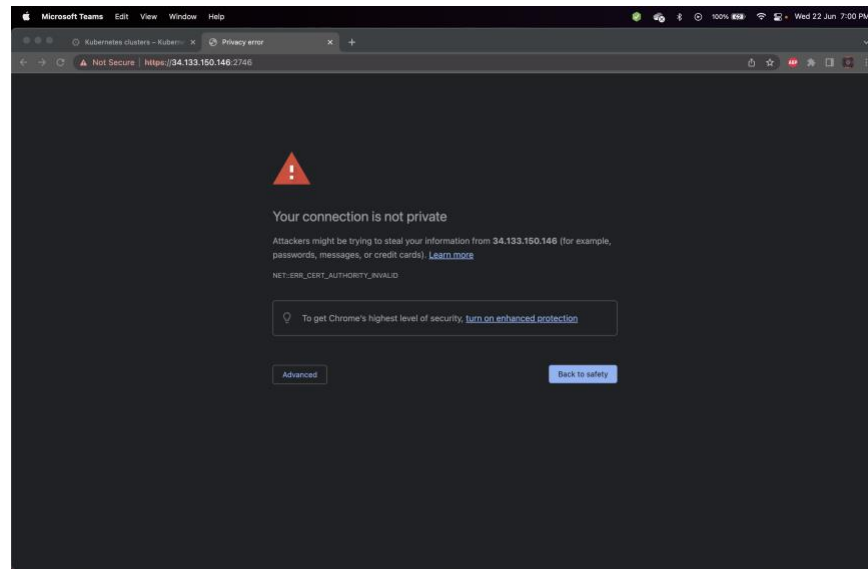


```
Cloud Shell
Terminal (cern-cms-epn) x (cern-cms-epn) x + v Open Editor

Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to cern-cms-epn.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
xavitintin007@cloudshell:~ (cern-cms-epn) $ lynx https://localhost:2746
xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl patch svc argo-server -n argo -p '{"spec": {"type": "LoadBalancer"}}'
W0622 23:57:11.277990 1971 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
service/argo-server patched
xavitintin007@cloudshell:~ (cern-cms-epn) $ kubectl get svc -n argo
W0622 23:57:33.111245 1979 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectl-auth-changes-in-gke
NAME                TYPE                CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
argo-server          LoadBalancer       10.52.3.236    31.269.111.245 2746:31269/TCP    94m
httpbin              ClusterIP           10.52.15.13    <none>          9100/TCP          94m
minio                ClusterIP           10.52.3.49     <none>          9000/TCP,9001/TCP 94m
nfs-server            ClusterIP           10.52.4.198    <none>          2049/TCP,20048/TCP,111/TCP 70m
postgres             ClusterIP           10.52.12.158   <none>          5432/TCP          94m
workflow-controller-metrics ClusterIP           10.52.14.25    <none>          9090/TCP          94m
xavitintin007@cloudshell:~ (cern-cms-epn) $
```

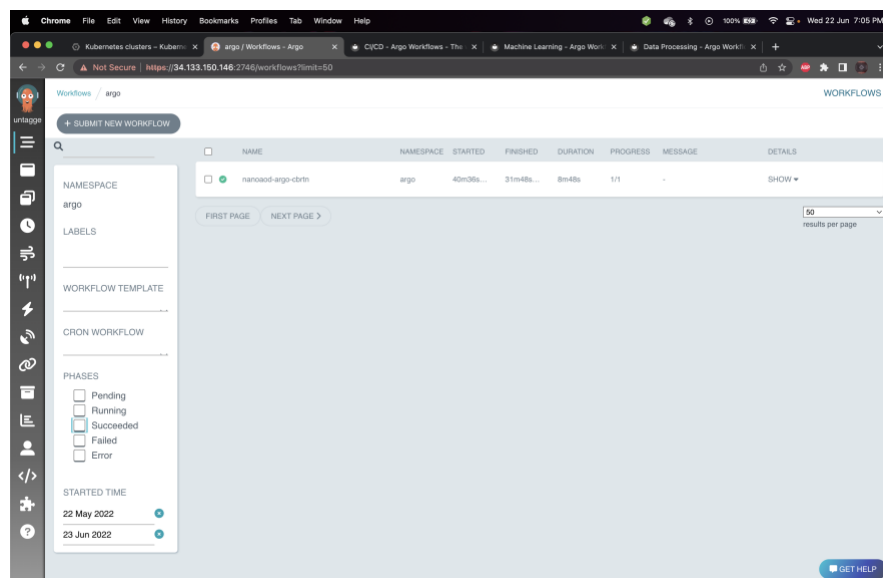
Fig. 24 Patch argo

Finally, you can access this address in your localhost. Do not forget to add “https://” and “:2746”



*Fig. 25 Enter Argo*

Click on Advanced, proceed to <ip>(unsafe) and voilà



*Fig. 26 Argo Interface*



## Accessing files via http

We first patch the config of the webserver to be created as follows:

```
mkdir conf.d
cd conf.d
curl -sLO https://raw.githubusercontent.com/cms-opendata-workshop/workshop-payload-kubernetes/master/conf.d/nginx-basic.conf
cd ..
kubectl create configmap basic-config --from-file=conf.d -n argo
```

Then prepare to deploy the fileserver by downloading the manifest:

```
curl -sLO https://github.com/cms-opendata-workshop/workshop-payload-kubernetes/raw/master/deployment-http-fileserver.yaml
```

Open this file and again adjust the <NUMBER> and the apiVersion to apps/v1 as follows:

```
vim deployment-http-fileserver.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    service: http-fileserver
    name: http-fileserver
spec:
  replicas: 1
  strategy: {}
  selector:
    matchLabels:
      service: http-fileserver
  template:
    metadata:
      labels:
        service: http-fileserver
    spec:
      volumes:
        - name: volume-output
          persistentVolumeClaim:
            claimName: nfs-<NUMBER>
        - name: basic-config
          configMap:
            name: basic-config
      containers:
        - name: file-storage-container
          image: nginx
```

```
ports:
- containerPort: 80
volumeMounts:
- mountPath: "/usr/share/nginx/html"
  name: volume-output
- name: basic-config
  mountPath: /etc/nginx/conf.d
```

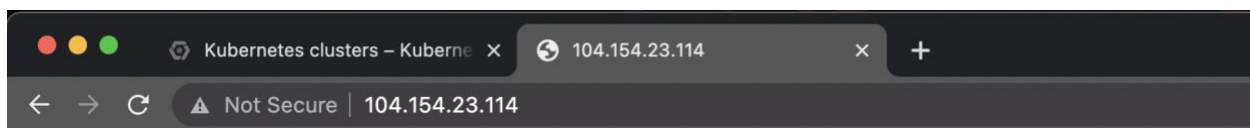
Apply and expose the port as a LoadBalancer:

```
kubectl create -n argo -f deployment-http-fileserver.yaml
kubectl expose deployment http-fileserver -n argo --type LoadBalancer --port
80 --target-port 80

kubectl get svc -n argo
```

```
CLOUD SHELL
Terminal (cern-cms-epn) x +
xavitintin007@cloudshell:~ (cern-cms-epn)$ kubectl get svc -n argo
W0623 00:17:24.207332 2267 gcp.go:120] WARNING: the gcp auth plugin is deprecated in v1.22+, unavailable in v1.25+; use gcloud instead.
To learn more, consult https://cloud.google.com/blog/products/containers-kubernetes/kubectyl-auth-changes-in-gke
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
argo-server    LoadBalancer 10.52.3.236    34.133.150.146 2746:31269/TCP    113m
http-fileserver LoadBalancer 10.52.13.154   104.154.23.114 80:30315/TCP      50s
httpbin        ClusterIP      10.52.15.13    <none>          9100/TCP          113m
minio          ClusterIP      10.52.3.49     <none>          9000/TCP,9001/TCP 113m
nfs-server     ClusterIP      10.52.4.198    <none>          2049/TCP,20048/TCP,111/TCP 90m
postgres       ClusterIP      10.52.12.158   <none>          5432/TCP          113m
workflow-controller-metrics ClusterIP 10.52.14.25    <none>          9090/TCP          113m
xavitintin007@cloudshell:~ (cern-cms-epn)$
```

Test it in your browser

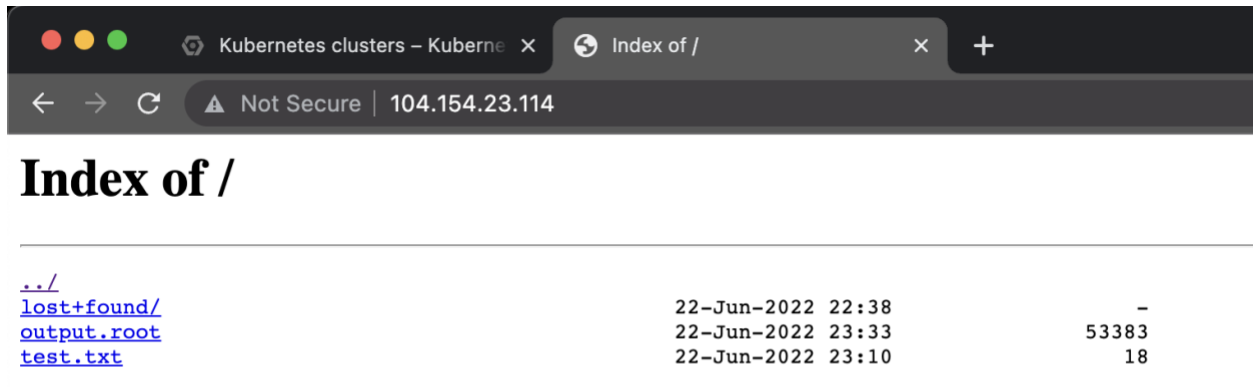


Hello from NFS!

The <pending> EXTERNAL-IP will update after a few minutes (run the command again to check). Once it's there, copy the IP and paste it into a new browser tab. This should welcome you with a "Hello from NFS" message. In order to enable file browsing, we need to delete the index.html file in the pod. Determine the pod name using the first command listed below and adjust the second command accordingly.

```
kubectl get pods -n argo
kubectl exec http-fileserver-XXXXXXXX-YYYYYY -n argo -- rm
/usr/share/nginx/html/index.html
```

Reload your localhost



<a href="#">../</a>		
<a href="#">lost+found/</a>	22-Jun-2022 22:38	-
<a href="#">output.root</a>	22-Jun-2022 23:33	53383
<a href="#">test.txt</a>	22-Jun-2022 23:10	18

**NOTE:** This IP is now accessible from anywhere in the world, and therefore also your files (mind: there are charges for outgoing bandwidth). Please delete the service again once you have finished downloading your files.

```
kubectl delete svc/http-fileserver -n argo
```

**Remember to delete your workflow again to avoid additional charges.**

```
argo delete -n argo @latest
```

## Get the output file

Create a file pv-pod.yaml with the following contents:

```
nano pv-pod.yaml
```

Paste configuration and change <NUMBER>:

```
apiVersion: v1
kind: Pod
metadata:
  name: pv-pod
spec:
  volumes:
  - name: task-pv-storage
    persistentVolumeClaim:
      claimName: nfs-<NUMBER>
```

```
containers:
- name: pv-container
  image: busybox
  command: ["tail", "-f", "/dev/null"]
  volumeMounts:
  - mountPath: /mnt/data
    name: task-pv-storage
```

Create the storage pod and copy the files from there

```
kubectl apply -f pv-pod.yaml -n argo
kubectl cp pv-pod:/mnt/data /tmp/poddata -n argo
```

and you will get the file created by the job in /tmp/poddata/test.txt.

## Deleting Workspace

Commands:

```
kubectl delete ns argo
rm *
gcloud compute disks list
```

Only delete gce-nfs-disk-..... with

```
xavitintin007@cloudshell:~ (cern-cms-epn) $ gcloud compute disks list
NAME: gce-nfs-disk-1
LOCATION: us-centrall-c
LOCATION_SCOPE: zone
SIZE_GB: 100
TYPE: pd-standard
STATUS: READY

NAME: gke-epn-cern-cms-default-pool-d23d8bc4-svf4
LOCATION: us-centrall-c
LOCATION_SCOPE: zone
SIZE_GB: 100
TYPE: pd-standard
STATUS: READY

NAME: gke-epn-cern-cms-default-pool-d23d8bc4-wtzj
LOCATION: us-centrall-c
LOCATION_SCOPE: zone
SIZE_GB: 100
TYPE: pd-standard
STATUS: READY
xavitintin007@cloudshell:~ (cern-cms-epn) $
```

```
gcloud compute disks delete gce-nfs-disk-1 --zone=us-centrall-c
```

```

xavitintin007@cloudshell:~ (cern-cms-epn)$ gcloud compute disks delete gce-nfs-disk-1 --zone=us-central1-c
The following disks will be deleted:
- [gce-nfs-disk-1] in [us-central1-c]

Do you want to continue (Y/n)? y

Deleted [https://www.googleapis.com/compute/v1/projects/cern-cms-epn/zones/us-central1-c/disks/gce-nfs-disk-1].
xavitintin007@cloudshell:~ (cern-cms-epn)$

```

## Delete Kubernetes Cluster:

### Delete epn-cern-cms

**⚠** This operation cannot be undone.

Deleting a cluster permanently removes all data from every pod within the cluster, as well as any containers running in each pod. [Learn more](#)

Do you want to delete cluster epn-cern-cms?

Confirm deletion by typing the cluster name below: epn-cern-cms

epn-cern-cms \*
  
epn-cern-cms

CANCEL
DELETE

Standby to see the complete deletion of the cluster

cern-cms-epn

Search Products, resources, docs (/)

Kubernetes clusters

CREATE

DEPLOY

REFRESH

OVERVIEW

COST OPTIMIZATION

Filter

Enter property name or value

Status	Name	Location	Number of nodes	Total vCPUs	Total memory
<div><div></div><div></div></div>	epn-cern-cms	us-central1-c	2	8	

Notifications

Delete Kubernetes Engine cluster "epn-cern-cms"
  
cern-cms-epn
  
Deleting cluster ...

✓

Create Kubernetes Engine cluster "epn-cern-cms"
  
cern-cms-epn
  
4 hours ago

✓

Delete Kubernetes Engine cluster "selenoidata"
  
cern-cms-epn
  
4 hours ago

Perfect you're ready to start over

**References:**

Workshop 2020: <https://cms-opendata-workshop.github.io/workshop-lesson-kubernetes/08-cleaning-up/index.html>

Workshop 2021: <https://cms-opendata-workshop.github.io/workshop2021-lesson-cloud/>

Argo Getting Started: [https://argo-cd.readthedocs.io/en/stable/getting\\_started/](https://argo-cd.readthedocs.io/en/stable/getting_started/)

CMS Repository: <https://github.com/cms-opendata-analyses/AOD2NanoAODOutreachTool>