

Name: Arthur M. Artugue

Subject: Elective 3

Section: IV-BCSAD

Date: November 26, 2025

Home Lab Activity

Hello Minikube

Create minikube cluster

Command
minikube start
Output
<pre>* minikube v1.37.0 on Microsoft Windows 11 Pro 10.0.26100.7171 Build 26100.7171 * Automatically selected the hyperv driver * Downloading VM boot image ... > minikube-v1.37.0-amd64.iso....: 65 B / 65 B [-----] 100.00% ? p/s 0s > minikube-v1.37.0-amd64.iso: 370.78 MiB / 370.78 MiB 100.00% 4.16 MiB p/ * Starting "minikube" primary control-plane node in "minikube" cluster * Downloading Kubernetes v1.34.0 preload ... > preloaded-images-k8s-v18-v1...: 337.07 MiB / 337.07 MiB 100.00% 4.11 Mi * Creating hyperv VM (CPUs=2, Memory=4000MB, Disk=20000MB) ... ! Failing to connect to https://registry.k8s.io/ from inside the minikube VM * To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/ * Preparing Kubernetes v1.34.0 on Docker 28.4.0 ... * Configuring bridge CNI (Container Networking Interface) ... * Verifying Kubernetes components... - Using image gcr.io/k8s-minikube/storage-provisioner:v5 * Enabled addons: storage-provisioner, default-storageclass ! C:\Program Files\Docker\Docker\resources\bin\kubectl.exe is version 1.32.2, which may have incompatibilities with Kubernetes 1.34.0. - Want kubectl v1.34.0? Try 'minikube kubectl -- get pods -A' * Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default</pre>

Open Dashboard

Open the Kubernetes dashboard. You can do this two different ways:

Command

minikube dashboard	
Output	
<pre>* Enabling dashboard ... - Using image docker.io/kubernetesui/dashboard:v2.7.0 - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8 * Some dashboard features require the metrics-server addon. To enable all features please run: minikube addons enable metrics-server * Verifying dashboard health ... * Launching proxy ... * Verifying proxy health ... * Opening http://127.0.0.1:57473/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...</pre>	

Create a deployment

A Kubernetes Pod is a group of one or more Containers, tied together for the purposes of administration and networking.

1. Use the kubectl create command to create a Deployment that manages a Pod. The Pod runs a Container based on the provided Docker image.

Command
kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.53 -- /agnhost netexec --http-port=8080

Output

```
PS C:\WINDOWS\system32> kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.53 -- /agnhost netexec --http-port=8080
deployment.apps/hello-node created
```

2. View The Deployment

Command

```
kubectl get deployments
```

Output

```
PS C:\WINDOWS\system32> kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
hello-node  1/1       1           1          60s
```

3. View The Pod

Command

```
kubectl get pods
```

Output

```
PS C:\WINDOWS\system32> kubectl get pods
NAME        READY   STATUS    RESTARTS   AGE
hello-node-6c9b5f4b59-dp8xx  1/1     Running   0          3m12s
```

4. View Cluster Events

Command

```
kubectl get events
```

Output

```

PS C:\WINDOWS\system32> kubectl get events
LAST SEEN    TYPE      REASON          OBJECT
MESSAGE
4m56s       Normal    Scheduled        pod/hello-node-6c9b5f4b59-dp8xx
Successfully assigned default/hello-node-6c9b5f4b59-dp8xx to minikube
4m55s       Normal    Pulling         pod/hello-node-6c9b5f4b59-dp8xx
Pulling image "registry.k8s.io/e2e-test-images/agnhost:2.53"
4m41s       Normal    Pulled          pod/hello-node-6c9b5f4b59-dp8xx
Successfully pulled image "registry.k8s.io/e2e-test-images/agnhost:2.53" in 14.025
s (14.025s including waiting). Image size: 139374622 bytes.
4m41s       Normal    Created         pod/hello-node-6c9b5f4b59-dp8xx
Created container: agnhost
4m41s       Normal    Started         pod/hello-node-6c9b5f4b59-dp8xx
Started container agnhost
4m56s       Normal    SuccessfulCreate replicaset/hello-node-6c9b5f4b59
Created pod: hello-node-6c9b5f4b59-dp8xx
4m56s       Normal    ScalingReplicaSet deployment/hello-node
Scaled up replica set hello-node-6c9b5f4b59 from 0 to 1
26m        Normal    Starting        node/minikube
Starting kubelet.
26m        Normal    NodeAllocatableEnforced node/minikube
Updated Node Allocatable limit across pods
26m        Normal    NodeHasSufficientMemory node/minikube
Node minikube status is now: NodeHasSufficientMemory
26m        Normal    NodeHasNoDiskPressure   node/minikube
Node minikube status is now: NodeHasNoDiskPressure
26m        Normal    NodeHasSufficientPID   node/minikube
Node minikube status is now: NodeHasSufficientPID
26m        Normal    NodeReady          node/minikube
Node minikube status is now: NodeReady
26m        Normal    RegisteredNode     node/minikube
Node minikube event: Registered Node minikube in Controller
26m        Normal    CIDRAssignmentFailed node/minikube
Node minikube status is now: CIDRAssignmentFailed
26m        Normal    Starting         node/minikube

```

5. View The kubectl configuration

Command
kubectl config view
Output

```
PS C:\WINDOWS\system32> kubectl config view
apiVersion: v1
clusters:
- cluster:
  certificate-authority: C:\Users\PC\.minikube\ca.crt
  extensions:
  - extension:
    last-update: Tue, 25 Nov 2025 12:51:09 CST
    provider: minikube.sigs.k8s.io
    version: v1.37.0
    name: cluster_info
    server: https://172.26.139.136:8443
    name: minikube
contexts:
- context:
  cluster: minikube
  extensions:
  - extension:
    last-update: Tue, 25 Nov 2025 12:51:09 CST
    provider: minikube.sigs.k8s.io
    version: v1.37.0
    name: context_info
  namespace: default
  user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
```

6. View application logs for a container in a pod (replace pod name with the one you got from kubectl get pods).

Command
kubectl logs hello-node-6c9b5f4b59-dp8xx
Output
PS C:\WINDOWS\system32> kubectl logs hello-node-6c9b5f4b59-dp8xx I1125 05:12:50.323623 1 log.go:245] Started HTTP server on port 8080 I1125 05:12:50.323903 1 log.go:245] Started UDP server on port 8081

Create Service

1. Expose the Pod to the public internet using the kubectl expose command:

Command

```
kubectl expose deployment hello-node --type=LoadBalancer --port=8080
```

Output

```
PS C:\WINDOWS\system32> kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed
```

2. View the Service you created:

Command

```
kubectl get services
```

Output

```
PS C:\WINDOWS\system32> kubectl get services
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
hello-node  LoadBalancer  10.102.188.173  <pending>      8080:30439/TCP  44s
kubernetes  ClusterIP   10.96.0.1       <none>        443/TCP     36m
```

3. Run the following command: This opens up a browser window that serves your app and shows the app's response.

Command

```
minikube service hello-node
```

Output

```
PS C:\WINDOWS\system32> minikube service hello-node
```

NAMESPACE	NAME	TARGET PORT	URL
default	hello-node	8080	http://172.26.139.136:30439

```
* Opening service default/hello-node in default browser...
```

```
< > C ⚠ Not secure 172.26.139.136:30439
```

```
NOW: 2025-11-25 05:29:06.875479131 +0000 UTC m=+976.562100184
```

Enable Addons

The minikube tool includes a set of built-in addons that can be enabled, disabled and opened in the local Kubernetes environment.

1. List the currently supported addons:

Command			
minikube addons list			
Output			
PS C:\WINDOWS\system32> minikube addons list			
ADDON NAME	PROFILE	STATUS	MAINTAINER
ambassador	minikube	disabled	3rd party (Ambassador)
amd-gpu-device-plugin	minikube	disabled	3rd party (AMD)
auto-pause	minikube	disabled	minikube
cloud-spanner	minikube	disabled	Google
csi-hostpath-driver	minikube	disabled	Kubernetes
dashboard	minikube	enabled <input checked="" type="checkbox"/>	Kubernetes
default-storageclass	minikube	enabled <input checked="" type="checkbox"/>	Kubernetes
efk	minikube	disabled	3rd party (Elastic)
freshpod	minikube	disabled	Google
gcp-auth	minikube	disabled	Google
gvisor	minikube	disabled	minikube
headlamp	minikube	disabled	3rd party (kinvolk.io)
inaccel	minikube	disabled	3rd party (InAccel [info@inaccel.com])
ingress	minikube	disabled	Kubernetes
ingress-dns	minikube	disabled	minikube
inspektor-gadget	minikube	disabled	3rd party (inspektor-gadget.io)
istio	minikube	disabled	3rd party (Istio)
istio-provisioner	minikube	disabled	3rd party (Istio)
kong	minikube	disabled	3rd party (Kong HQ)
kubeflow	minikube	disabled	3rd party
kubetail	minikube	disabled	3rd party (kubetail.com)
kubevirt	minikube	disabled	3rd party (KubeVirt)
logviewer	minikube	disabled	3rd party (unknown)
metallb	minikube	disabled	3rd party (MetallLB)
metrics-server	minikube	disabled	Kubernetes
nvidia-device-plugin	minikube	disabled	3rd party (NVIDIA)
nvidia-driver-installer	minikube	disabled	3rd party (NVIDIA)
nvidia-gpu-device-plugin	minikube	disabled	3rd party (NVIDIA)
olm	minikube	disabled	3rd party (Operator Framework)
pod-security-policy	minikube	disabled	3rd party (unknown)
portainer	minikube	disabled	3rd party (Portainer.io)
registry	minikube	disabled	minikube
registry-aliases	minikube	disabled	3rd party (unknown)
registry-creds	minikube	disabled	3rd party (UPMC Enterprises)
storage-provisioner	minikube	enabled <input checked="" type="checkbox"/>	minikube
storage-provisioner-gluster	minikube	disabled	3rd party (Gluster)
storage-provisioner-rancher	minikube	disabled	3rd party (Rancher)
volcano	minikube	disabled	third-party (volcano)
volumesnapshots	minikube	disabled	Kubernetes
yakd	minikube	disabled	3rd party (marcnuri.com)

2. Enable an addon, for example, metrics-server:

Command
minikube addons enable metrics-server
Output
PS C:\WINDOWS\system32> minikube addons enable metrics-server * metrics-server is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub. You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS - Using image registry.k8s.io/metrics-server/metrics-server:v0.8.0 * The 'metrics-server' addon is enabled

3. Enable an addon, for example, metrics-server:

Command
kubectl get pod,svc -n kube-system
Output
PS C:\WINDOWS\system32> kubectl get pod,svc -n kube-system NAME READY STATUS RESTARTS AGE pod/coredns-66bc5c9577-zjvcx 1/1 Running 0 47m pod/etcfd-minikube 1/1 Running 0 47m pod/kube-apiserver-minikube 1/1 Running 0 47m pod/kube-controller-manager-minikube 1/1 Running 0 47m pod/kube-proxy-mb4dd 1/1 Running 0 47m pod/kube-scheduler-minikube 1/1 Running 0 47m pod/metrics-server-85b7d694d7-mbz4s 0/1 Running 0 55s pod/storage-provisioner 1/1 Running 1 (46m ago) 47m NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE service/kube-dns ClusterIP 10.96.0.10 <none> 53/UDP,53/TCP,9153/TCP 47m service/metrics-server ClusterIP 10.102.165.83 <none> 443/TCP 55s

4. Check the output from metrics-server:

Command
kubectl top pods
Output
PS C:\WINDOWS\system32> kubectl top pods NAME CPU(cores) MEMORY(bytes) hello-node-6c9b5f4b59-dp8xx 1m 5Mi

5. Disable metrics-server:

Command
minikube addons disable metrics-server
Output
PS C:\WINDOWS\system32> minikube addons disable metrics-server * "The 'metrics-server' addon is disabled

Clean up

Now you can clean up the resources you created in your cluster:

Command
kubectl delete service hello-node kubectl delete deployment hello-node
Output
PS C:\WINDOWS\system32> kubectl delete service hello-node >> kubectl delete deployment hello-node service "hello-node" deleted deployment.apps "hello-node" deleted

Command
minikube stop
Output
PS C:\WINDOWS\system32> minikube stop * Stopping node "minikube" ... * Powering off "minikube" via SSH ... * 1 node stopped.

Command
minikube delete # Optional
Output
PS C:\WINDOWS\system32> minikube stop * Stopping node "minikube" ... * Powering off "minikube" via SSH ... * 1 node stopped.

Get a shell to a running container

In this exercise, you create a Pod that has one container. The container runs the nginx image.

Create the pod

Command
kubectl apply -f https://k8s.io/examples/application/shell-demo.yaml
Output
PS C:\WINDOWS\system32> kubectl apply -f https://k8s.io/examples/application/shell-demo.yaml pod/shell-demo created

Verify that the container is running

Command
kubectl get pod shell-demo
Output
PS C:\WINDOWS\system32> kubectl get pod shell-demo >> NAME READY STATUS RESTARTS AGE shell-demo 1/1 Running 0 32m

Get a shell to the running container:

Command
kubectl exec --stdin --tty shell-demo -- /bin/bash
Output
PS C:\WINDOWS\system32> kubectl exec --stdin --tty shell-demo -- /bin/bash root@minikube:/#

In your shell, list the root directory:

Command
ls /
Output

```
root@minikube:/# ls /
bin  dev          docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot docker-entrypoint.d  etc            lib   media  opt  root  sbin  sys  usr
root@minikube:/#
```

Writing the root page for nginx

Look again at the configuration file for your Pod. The Pod has an emptyDir volume, and the container mounts the volume at /usr/share/nginx/html.

In your shell, create an index.html file in the /usr/share/nginx/html directory:

Command
echo 'Hello shell demo' > /usr/share/nginx/html/index.html
Output
<pre>root@minikube:/# echo 'Hello shell demo' > /usr/share/nginx/html/index.html root@minikube:/#</pre>

In your shell, send a GET request to the nginx server:

Command
apt-get update
Output
<pre>root@minikube:/# apt-get update Get:1 http://deb.debian.org/debian trixie InRelease [140 kB] Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB] Get:3 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB] Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [9670 kB] Get:5 http://deb.debian.org/debian trixie-updates/main amd64 Packages [5412 B] Get:6 http://deb.debian.org/debian-security trixie-security/main amd64 Packages [71.8 kB] Fetched 9978 kB in 1s (8126 kB/s) Reading package lists... Done root@minikube:/#</pre>

Command
apt-get install curl
Output

```
root@minikube:/# apt-get install curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (8.14.1-2+deb13u2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@minikube:/#
```

Command
curl http://localhost/
Output
<pre>root@minikube:/# curl http://localhost/ Hello shell demo root@minikube:/#</pre>

When you are finished with your shell, enter exit.

Command
exit # To quit the shell in the container
Output
<pre>root@minikube:/# exit exit PS C:\WINDOWS\system32></pre>

Running individual commands in a container

In an ordinary command window, not your shell, list the environment variables in the running container:

Command
kubectl exec shell-demo -- env
Output

```

PS C:\WINDOWS\system32> kubectl exec shell-demo -- env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=minikube
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
NGINX_VERSION=1.29.3
NJS_VERSION=0.9.4
NJS_RELEASE=1~trixie
PKG_RELEASE=1~trixie
DYNPKG_RELEASE=1~trixie
HOME=/root

```

Deploying wordpress and Mysql with persistent volumes

Create customization yaml

Add a secret generator

Add a Secret generator in kustomization.yaml from the following command. You will need to replace YOUR_PASSWORD with the password you want to use.

Command
<pre> cat <<EOF >/kustomization.yaml secretGenerator: - name: mysql-pass literals: - password=YOUR_PASSWORD EOF </pre>

But Use This Command instead if in Windows CMD or Powershell

```

@"
secretGenerator:
- name: mysql-pass
literals:
- password=password123
resources:
- mysql-deployment.yaml

```

```
- wordpress-deployment.yaml  
"@ | Set-Content -Encoding utf8 ./kustomization.yaml
```

Add resource configs for MySQL and WordPress

1. Download the MySQL deployment configuration file.

Command
<pre>curl -LO https://k8s.io/examples/application/wordpress/mysql-deployment.yaml</pre>
Output
<pre>C:\Windows\System32>curl -LO https://k8s.io/examples/application/wordpress/mysql-deployment.yaml % Total % Received % Xferd Average Speed Time Time Time Current Dload Upload Total Spent Left Speed 100 162 100 162 0 0 325 0 --:--:-- 0:00:01 --:--:-- 326 100 1442 100 1442 0 0 1372 0 0:00:01 0:00:01 --:--:-- 3755</pre>

2. Download the WordPress configuration file.

Command
<pre>curl -LO https://k8s.io/examples/application/wordpress-wordpress-deployment.yaml</pre>
Output
<pre>C:\Windows\System32>curl -LO https://k8s.io/examples/application/wordpress-wordpress-deployment.yaml % Total % Received % Xferd Average Speed Time Time Time Current Dload Upload Total Spent Left Speed 100 162 100 162 0 0 720 0 --:--:-- 0:00:01 --:--:-- 723 100 1341 100 1341 0 0 2082 0 --:--:-- 0:00:01 --:--:-- 2082</pre>

3. Add them to kustomization.yaml file.

Command
<pre>cat <<EOF >>./kustomization.yaml resources: - mysql-deployment.yaml - wordpress-deployment.yaml EOF</pre>

Note: changed syntax because it wont work on windows powershell or cmd

Output

```
PS C:\WINDOWS\system32> @"
>> resources:
>>   - mysql-deployment.yaml
>>   - wordpress-deployment.yaml
>> "@ | Out-File -Encoding utf8 ./kustomization.yaml
>>
PS C:\WINDOWS\system32>
```

Apply and Verify

The kustomization.yaml contains all the resources for deploying a WordPress site and a MySQL database. You can apply the directory by

Command

```
kubectl apply -k ./
```

Output

```
ps C:\WINDOWS\system32> kubectl apply -k ./
>>
secret/mysql-pass-2g227htkh5 created
service/wordpress unchanged
service/wordpress-mysql unchanged
persistentvolumeclaim/mysql-pv-claim unchanged
persistentvolumeclaim/wp-pv-claim unchanged
deployment.apps/wordpress configured
deployment.apps/wordpress-mysql configured
```

Now you can verify that all objects exist.

1. Now you can verify that all objects exist.

Command

```
kubectl get secrets
```

Output

```
PS C:\WINDOWS\system32> kubectl get secrets
NAME          TYPE      DATA  AGE
mysql-pass-2g227htkh5  Opaque    1    17s
```

2. Verify that a PersistentVolume got dynamically provisioned.

Command
kubectl get pvc
Output
<pre>PS C:\WINDOWS\system32> kubectl get pvc NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS VOLUMEATTR mysql-pv-claim Bound pvc-cbe79c12-c1c7-4eea-90f1-e959c7590280 20Gi RWO standard <unset> wp-pv-claim Bound pvc-5c8ff6da-ca85-4cd9-97d5-788d8b8ee324 20Gi RWO standard <unset> 20m</pre>

3. Verify that the Pod is running by running the following command:

Command
kubectl get pods
Output
<pre>PS C:\WINDOWS\system32> kubectl get pods NAME READY STATUS RESTARTS AGE wordpress-69468fcd7f-r8p9h 1/1 Running 0 11m wordpress-mysql-7d4c886d98-zrgm7 1/1 Running 0 11m</pre>

4. Verify that the Service is running by running the following command:

Command
kubectl get services wordpress
Output
<pre>NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE wordpress LoadBalancer 10.102.160.1 <pending> 80:30396/TCP 22m</pre>

5. Run the following command to get the IP Address for the WordPress Service:

Command
minikube service wordpress --url

Output

```
PS C:\WINDOWS\system32> minikube service wordpress --url  
http://172.26.181.11:30396
```

6. Copy the IP address, and load the page in your browser to view your site.

You should see the WordPress set up page similar to the following screenshot.

Output

The screenshot shows the WordPress setup process. At the top, a language selection dropdown is open, listing various languages including English (United States), Afrikaans, বাংলা, Aragonés, العربية, العربية المغربية, অসমীয়া, گوئىچى ئۇرپىزان, Azerbaijani, Belarusian, Български, বাংলা, ດັວໂນ, Bosanski, Català, Cebuano, Čeština, Cymraeg, Dansk, Deutsch (Schweiz), Deutsch (Österreich), and Deutsch (Deutschland). A 'Continue' button is visible at the bottom of the dropdown. Below this, the main 'Welcome' screen is shown with the title 'Information needed'. It asks for Site Title, Username, Password (which is marked as 'Strong'), and Your Email. A note says 'Important: You will need this password to log in. Please store it in a secure location.' and 'Double-check your email address before continuing.'

Cleaning up

5. Run the following command to delete your Secret, Deployments, Services and PersistentVolumeClaims:

Command
kubectl delete -k ./
Output
<pre>PS C:\WINDOWS\system32> minikube profile list +-----+-----+-----+-----+-----+-----+-----+-----+-----+ PROFILE DRIVER RUNTIME IP VERSION STATUS NODES ACTIVE PROFILE ACTIVE KUBECONTEXT +-----+-----+-----+-----+-----+-----+-----+-----+-----+ minikube hyperv docker 172.26.181.11 v1.34.0 OK 1 * * +-----+-----+-----+-----+-----+-----+-----+-----+-----+ PS C:\WINDOWS\system32> PS C:\WINDOWS\system32> minikube stop * Stopping node "minikube" ... * Powering off "minikube" via SSH ... * 1 node stopped.</pre>