

**Name:** Arthur M. Artugue

**Subject:** Elective 3

**Section:** IV-BCSAD

**Date:** November 26, 2025

# Home Lab Activity

## Hello Minikube

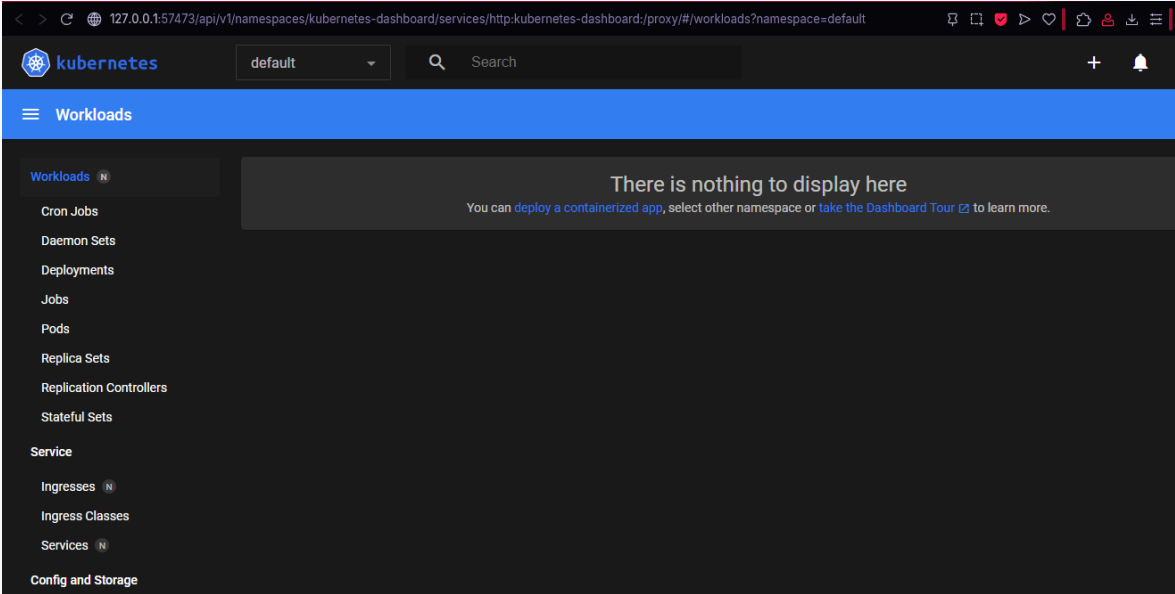
### Create minikube cluster

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 ...   &gt; minikube-v1.37.0-amd64.iso....: 65 B / 65 B [-----] 100.00% ? p/s 0s   &gt; 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 ...   &gt; preloaded-images-k8s-v18-v1...: 337.07 MiB / 337.07 MiB 100.00% 4.11 MiB p/ * 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
<p style="text-align: center;"><b>Output</b></p> <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/h 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
<pre>kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.53 -- /agnhost netexec --http-port=8080</pre>

Output
<pre>PS C:\WINDOWS\system32&gt; 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</pre>

## 2. View The Deployment

Command
<b>kubectl get deployments</b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl get deployments NAME          READY   UP-TO-DATE   AVAILABLE   AGE hello-node    1/1     1             1           60s</pre>

## 3. View The Pod

Command
<b>kubectl get pods</b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl get pods NAME                                READY   STATUS    RESTARTS   AGE hello-node-6c9b5f4b59-dp8xx        1/1     Running   0           3m12s</pre>

## 4. View Cluster Events

Command
<b>kubectl get events</b>
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
<b>kubectl config view</b>
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
<code>kubectl logs hello-node-6c9b5f4b59-dp8xx</code>
Output
<pre> PS C:\WINDOWS\system32&gt; 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 </pre>

## Create Service

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

Command
---------

<b>kubectl expose deployment hello-node --type=LoadBalancer --port=8080</b>
<b>Output</b>
<pre>PS C:\WINDOWS\system32&gt; kubectl expose deployment hello-node --type=LoadBalancer --port=8080 service/hello-node exposed</pre>

2. View the Service you created:

Command																		
kubectl get services																		
Output																		
<pre>PS C:\WINDOWS\system32&gt; kubectl get services</pre> <table><tr><th>NAME</th><th>TYPE</th><th>CLUSTER-IP</th><th>EXTERNAL-IP</th><th>PORT(S)</th><th>AGE</th></tr><tr><td>hello-node</td><td>LoadBalancer</td><td>10.102.188.173</td><td>&lt;pending&gt;</td><td>8080:30439/TCP</td><td>44s</td></tr><tr><td>kubernetes</td><td>ClusterIP</td><td>10.96.0.1</td><td>&lt;none&gt;</td><td>443/TCP</td><td>36m</td></tr></table>	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
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								
<pre>PS C:\WINDOWS\system32&gt; minikube service hello-node</pre> <table><tr><th>NAMESPACE</th><th>NAME</th><th>TARGET PORT</th><th>URL</th></tr><tr><td>default</td><td>hello-node</td><td>8080</td><td>http://172.26.139.136:30439</td></tr></table> <pre>* Opening service default/hello-node in default browser...</pre> <div><span>&lt;</span> <span>&gt;</span> <span>↻</span> <span>⚠ Not secure</span> 172.26.139.136:30439</div> <pre>NOW: 2025-11-25 05:29:06.875479131 +0000 UTC m=+976.562100184</pre>	NAMESPACE	NAME	TARGET PORT	URL	default	hello-node	8080	http://172.26.139.136:30439
NAMESPACE	NAME	TARGET PORT	URL					
default	hello-node	8080	http://172.26.139.136:30439					

## 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			
<pre>PS C:\WINDOWS\system32&gt; minikube addons list</pre>			
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 (MetalLB)
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
<b>minikube addons enable metrics-server</b>
Output
<pre>PS C:\WINDOWS\system32&gt; 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</pre>

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

Command
<b>kubectl get pod,svc -n kube-system</b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl get pod,svc -n kube-system NAME                                READY   STATUS    RESTARTS   AGE pod/coredns-66bc5c9577-zjvcx       1/1     Running   0           47m pod/etcd-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      &lt;none&gt;            53/UDP,53/TCP,9153/TCP 47m service/metrics-server              ClusterIP      10.102.165.83   &lt;none&gt;            443/TCP           55s</pre>

4. Check the output from metrics-server:

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

## 5. Disable metrics-server:

Command
<b>minikube addons disable metrics-server</b>
Output
<pre>PS C:\WINDOWS\system32&gt; minikube addons disable metrics-server * "The 'metrics-server' addon is disabled"</pre>

## *Clean up*

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

Command
<b>kubectl delete service hello-node</b> <b>kubectl delete deployment hello-node</b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl delete service hello-node &gt;&gt; kubectl delete deployment hello-node service "hello-node" deleted deployment.apps "hello-node" deleted</pre>

Command
<b>minikube stop</b>
Output
<pre>PS C:\WINDOWS\system32&gt; minikube stop * Stopping node "minikube" ... * Powering off "minikube" via SSH ... * 1 node stopped.</pre>

Command
<b>minikube delete # Optional</b>
Output
<pre>PS C:\WINDOWS\system32&gt; minikube stop * Stopping node "minikube" ... * Powering off "minikube" via SSH ... * 1 node stopped.</pre>

# 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
<b>kubectl apply -f <a href="https://k8s.io/examples/application/shell-demo.yaml">https://k8s.io/examples/application/shell-demo.yaml</a></b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl apply -f https://k8s.io/examples/application/shell-demo.yaml pod/shell-demo created</pre>

Verify that the container is running

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

Get a shell to the running container:

Command
<b>kubectl exec --stdin --tty shell-demo -- /bin/bash</b>
Output
<pre>PS C:\WINDOWS\system32&gt; kubectl exec --stdin --tty shell-demo -- /bin/bash root@minikube:/#</pre>

In your shell, list the root directory:

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

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

Command
<b>apt-get update</b>
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
<b>apt-get install curl</b>
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
<b>curl http://localhost/</b>
Output
<pre>root@minikube:/# curl http://localhost/ Hello shell demo root@minikube:/#</pre>

When you are finished with your shell, enter exit.

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

## ***Running individual commands in a container***

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

Command
<b>kubectl exec shell-demo -- env</b>
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 &lt;&lt;EOF &gt;./kustomization.yaml secretGenerator: - name: mysql-pass   literals:   - password=YOUR_PASSWORD EOF </pre>
<p>But Use This Command instead if in Windows CMD or Powershell</p> <pre> @" secretGenerator: - name: mysql-pass   literals:   - password=password123 resources: - mysql-deployment.yaml </pre>

```
- 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&gt;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  --:--:-- --:--:-- --:--:--   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&gt;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  --:--:-- --:--:-- --:--:--   723 100 1341  100 1341    0    0  2082      0  --:--:-- --:--:-- --:--:--  2082</pre>

3. Add them to kustomization.yaml file.

Command
<pre>cat &lt;&lt;EOF &gt;&gt;./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&gt; kubectl get pvc NAME                                STATUS    VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTR mysql-pv-claim                      Bound     pvc-cbe79c12-c1c7-4eea-90f1-e959c7590280  20Gi       RWO            standard       &lt;unset&gt; wp-pv-claim                         Bound     pvc-5c8ff6da-ca85-4cd9-97d5-788d8b8ee324  20Gi       RWO            standard       &lt;unset&gt;</pre>							

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

Command					
kubectl get pods					
Output					
<pre>PS C:\WINDOWS\system32&gt; 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					
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
wordpress	LoadBalancer	10.102.160.1	<pending>	80:30396/TCP	22m

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 displays the WordPress installation interface. The top section, titled 'Welcome', features the WordPress logo and a message: 'Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.' Below this is the 'Information needed' section, which instructs the user to provide the following information, noting that settings can be changed later:

- Site Title:** A text input field.
- Username:** A text input field. A note below states: 'Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.'
- Password:** A text input field with a strength indicator showing 'Strong' and a 'Show' button.
- Your Email:** A text input field. A note below states: 'Double-check your email address before continuing.'

## ***Cleaning up***

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

Command

kubect! delete -k ./

Output

```
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.
```