

Assignment #5 - Kubernetes Home Lab Activity

1. Hello Minikube

● Creating a Minikube Cluster

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.26100.7171]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>minikube start
* minikube v1.37.0 on Microsoft Windows 11 Pro 10.0.26100.7171 Build 26100.7171
* Using the hyperv driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Restarting existing hyperv VM for "minikube" ...
! To restart 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 docker.io/kubernetesui/dashboard:v2.7.0
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
  - 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

* Enabled addons: storage-provisioner, default-storageclass, dashboard

! C:\Program Files\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

C:\Windows\System32>
```

● Opening the Kubernetes Dashboard

The screenshot shows a dual-monitor setup. On the left monitor, a command prompt window titled 'Administrator: Command Prompt - minikube dashboard' displays the command 'minikube dashboard' and its execution. On the right monitor, a web browser window titled 'Hello Minikube' shows the Kubernetes Dashboard. The dashboard has a dark theme with a sidebar on the left containing icons for Workloads, Services, Ingresses, and Deployments. The main area is titled 'Workloads' and shows a single green circle representing a running deployment with the status 'Running: 1'. Below the main area, there are sections for 'Service', 'Ingresses', and 'Deployments'.

- Creating a Deployment

```
C:\Windows\System32>kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.53 -- /agnhost  
deployment.apps/hello-node created
```

- View the Deployments

```
C:\Windows\System32>kubectl get deployments  
NAME      READY   UP-TO-DATE   AVAILABLE   AGE  
hello-node 1/1     1           1           88s
```

```
C:\Windows\System32>
```

- View the Pods

```
C:\Windows\System32>kubectl get pods  
NAME          READY   STATUS    RESTARTS   AGE  
hello-node-6c9b5f4b59-2f2gf  1/1     Running   0          2m6s
```

```
C:\Windows\System32>
```

- View Cluster Events

```
C:\Windows\System32>kubectl get events  
LAST SEEN   TYPE      REASON          OBJECT  
MESSAGE  
2m27s       Normal    Scheduled        pod/hello-node-6c9b5f4b59-2f2gf  
Successfully assigned default/hello-node-6c9b5f4b59-2f2gf to minikube  
2m26s       Normal    Pulling          pod/hello-node-6c9b5f4b59-2f2gf  
Pulling image "registry.k8s.io/e2e-test-images/agnhost:2.53"  
2m16s       Normal    Pulled           pod/hello-node-6c9b5f4b59-2f2gf  
Successfully pulled image "registry.k8s.io/e2e-test-images/agnhost:2.53" in 9.815s  
(9.815s including waiting). Image size: 139374622 bytes.  
2m16s       Normal    Created          pod/hello-node-6c9b5f4b59-2f2gf  
Created container: agnhost  
2m16s       Normal    Started          pod/hello-node-6c9b5f4b59-2f2gf  
Started container agnhost  
2m27s       Normal    SuccessfulCreate   replicaset/hello-node-6c9b5f4b59  
Created pod: hello-node-6c9b5f4b59-2f2gf  
2m27s       Normal    ScalingReplicaSet  deployment/hello-node  
Scaled up replica set hello-node-6c9b5f4b59 from 0 to 1  
3m27s       Normal    Starting         node/minikube  
Starting kubelet.  
3m27s       Normal    NodeAllocatableEnforced  node/minikube  
Updated Node Allocatable limit across pods  
3m26s       Normal    NodeHasSufficientMemory  node/minikube  
Node minikube status is now: NodeHasSufficientMemory  
3m26s       Normal    NodeHasNoDiskPressure   node/minikube  
Node minikube status is now: NodeHasNoDiskPressure  
3m26s       Normal    NodeHasSufficientPID    node/minikube  
Node minikube status is now: NodeHasSufficientPID  
3m23s       Normal    NodeReady          node/minikube  
Node minikube status is now: NodeReady  
3m23s       Normal    RegisteredNode    node/minikube  
Node minikube event: Registered Node minikube in Controller  
3m20s       Normal    Starting          node/minikube
```

- View the kubectl configuration

```
C:\Windows\System32>kubectl config view
apiVersion: v1
clusters:
- cluster:
  certificate-authority: C:\Users\admin_pc\.minikube\ca.crt
  extensions:
  - extension:
    last-update: Mon, 01 Dec 2025 20:50:38 +08
    provider: minikube.sigs.k8s.io
    version: v1.37.0
    name: cluster_info
  server: https://172.29.106.254:8443
  name: minikube
contexts:
- context:
  cluster: minikube
  extensions:
  - extension:
    last-update: Mon, 01 Dec 2025 20:50:38 +08
    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:
    client-certificate: C:\Users\admin_pc\.minikube\profiles\minikube\client.crt
    client-key: C:\Users\admin_pc\.minikube\profiles\minikube\client.key
```

- View Application Logs from Containers

```
C:\Windows\System32>kubectl logs hello-node-6c9b5f4b59-2f2gf
I1201 12:50:45.766007      1 log.go:245] Started HTTP server on port 8080
I1201 12:50:45.766364      1 log.go:245] Started UDP server on port 8081
C:\Windows\System32>
```

- Exposing the Pod to the public internet

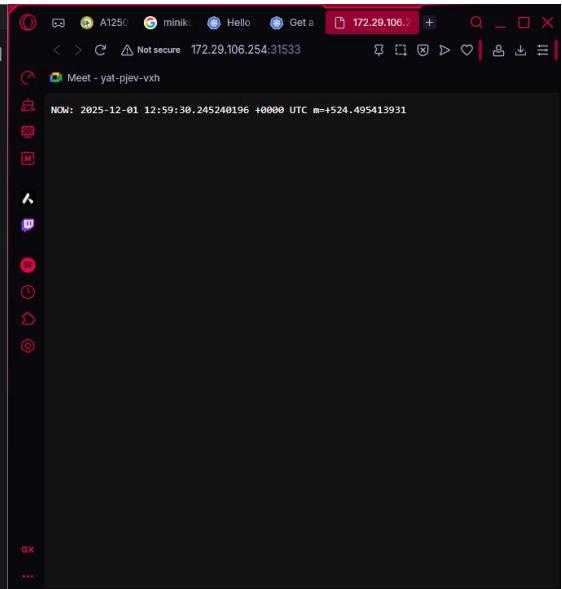
```
C:\Windows\System32>kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed
C:\Windows\System32>
```

- View the Created Services

```
C:\Windows\System32>kubectl get services
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
hello-node  LoadBalancer  10.105.211.242  <pending>      8080:31533/TCP  35s
kubernetes  ClusterIP  10.96.0.1      <none>        443/TCP      7m52s

C:\Windows\System32>
```

- Opening up a browser window that serves your app and shows the app's response.



The screenshot illustrates the process of deploying a simple application (Hello World) to a local Kubernetes cluster (minikube) and then viewing its output in a web browser.

Command Prompt (Left):

```
C:\Windows\System32>kubectl get services
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
hello-node  LoadBalancer  10.105.211.242  <pending>      8080:31533/TCP  35s
kubernetes  ClusterIP  10.96.0.1      <none>        443/TCP      7m52s

C:\Windows\System32>kubectl logs hello-node-5f76cf6ccf-br9b5
I1201 12:50:45.766007 1 log.go:245] Started HTTP server on port 8080
I1201 12:50:45.766364 1 log.go:245] Started UDP server on port 8081

C:\Windows\System32>kubectl expose deployment hello-node --type=LoadBalancer --port=8080
service/hello-node exposed

C:\Windows\System32>kubectl get services
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
hello-node  LoadBalancer  10.105.211.242  <pending>      8080:31533/TCP  35s
kubernetes  ClusterIP  10.96.0.1      <none>        443/TCP      7m52s

C:\Windows\System32>minikube service hello-node
  NAME        TARGET PORT | URL
  default     8080          http://172.29.106.254:31533
* Opening service default/hello-node in default browser...

C:\Windows\System32>minikube service hello-node
  NAME        TARGET PORT | URL
  default     8080          http://172.29.106.254:31533
* Opening service default/hello-node in default browser...
```

Browser Window (Right):

A Microsoft Edge browser window is open, showing a "Meet" page from Google Meet. The URL in the address bar is `http://172.29.106.254:31533`. The page displays a video feed and basic meeting controls.

- Viewing currently supported addons

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	disabled	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 naccel.com])	minikube	disabled	3rd party (InAccel [info@i naccel.com])
ingress	minikube	disabled	Kubernetes
ingress-dns	minikube	disabled	minikube
inspekto-r-gadget t.io)	minikube	disabled	3rd party (inspekto-r-gadge t.io)
istio	minikube	disabled	3rd party (Istio)
istio-provisioner	minikube	disabled	3rd party (Istio)

- Enabling an addon (Metrics Services)

```
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

C:\Windows\System32>
```

- View the Pod Services your created with the addon installed

```
C:\Windows\System32>kubectl get pod,svc -n kube-system
NAME                               READY   STATUS    RESTARTS   AGE
pod/coredns-66bc5c9577-blkd7      1/1     Running   0          12m
pod/etcd-minikube                 1/1     Running   0          12m
pod/kube-apiserver-minikube      1/1     Running   0          12m
pod/kube-controller-manager-minikube 1/1     Running   0          12m
pod/kube-proxy-c557t              1/1     Running   0          12m
pod/kube-scheduler-minikube      1/1     Running   0          12m
pod/metrics-server-85b7d694d7-m52f9 0/1     Running   0          54s
pod/storage-provisioner           1/1     Running   0          12m

NAME                TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)
                   AGE
service/kube-dns      ClusterIP   10.96.0.10    <none>         53/UDP,53/TCP,91
53/TCP   12m
service/metrics-server ClusterIP   10.98.74.242   <none>         443/TCP
54s

C:\Windows\System32>
```

- Check the output from metrics-server

```
C:\Windows\System32>kubectl top pods
NAME                  CPU(cores)   MEMORY(bytes)
hello-node-6c9b5f4b59-2f2gf   1m          5Mi

C:\Windows\System32>
```

- Disable the metrics-server addon

```
C:\Windows\System32>minikube addons disable metrics-server
* "The 'metrics-server' addon is disabled

C:\Windows\System32>
```

- Cleaning Up

```
C:\Windows\System32>kubectl delete service hello-node
service "hello-node" deleted

C:\Windows\System32>kubectl delete deployment hello-node
deployment.apps "hello-node" deleted

C:\Windows\System32>
```

- Stopping the Minikube

2. Get A Shell to a Running Container

- Creating the Pod and Verifying if it's running

```
C:\Windows\System32>kubectl apply -f https://k8s.io/examples/application/shell-demo.yaml
pod/shell-demo created

C:\Windows\System32>kubectl get pod shell-demo
NAME      READY   STATUS        RESTARTS   AGE
shell-demo  0/1     ContainerCreating   0          8s

C:\Windows\System32>
```

- Get a Shell to the Running Container and list the root directory

```
C:\Windows\System32>kubectl exec --stdin --tty shell-demo -- /bin/bash
root@minikube:/# ls /
bin  docker-entrypoint.d  home  media  proc  sbin  tmp
boot docker-entrypoint.sh  lib    mnt    root  srv  usr
dev   etc                lib64  opt    run   sys  var
```

- Writing the Root Page for Nginx

```
root@minikube:/# echo 'Hello shell demo' > /usr/share/nginx/html/index.html
```

- Sending a GET request to the nginx server

```
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 [76.0 kB]
Fetched 9982 kB in 2s (5904 kB/s)
Reading package lists... Done
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:/# curl http://localhost/
Hello shell demo
root@minikube:/#
```

- Exiting the Shell

```
root@minikube:/# exit
exit

C:\Windows\System32>
```

3. Deploying WordPress and MySQL with Persistent Volumes

- Downloading the MySQL and Wordpress Deployment YAMLS.

```
PS C:\Users\Admin> # Download MySQL YAML
>> Invoke-WebRequest -Uri "https://k8s.io/examples/application/wordpress/mysql-deployment.yaml" -OutFile "mysql-deployment.yaml"
>>
>> # Download WordPress YAML
>> Invoke-WebRequest -Uri "https://k8s.io/examples/application/wordpress/wordpress-deployment.yaml" -OutFile "wordpress-deployment.yaml"
>>
PS C:\Users\Admin>
```

- Creating a Kustomization.yaml with a secret generator.

```
PS C:\Users\Admin> @"
>> secretGenerator:
>>   - name: mysql-pass
>>     literals:
>>       - password=GEEWONII
>>     "@ > kustomization.yaml
>>
```

- Append the MySQL and WordPress YAML files to kustomization.yaml

```
PS C:\Users\Admin> @@
>> resources:
>>   - mysql-deployment.yaml
>>   - wordpress-deployment.yaml
>> "@ >> kustomization.yaml
>>
PS C:\Users\Admin>
```

- Deploying all resources to the cluster

```
PS C:\Users\Admin> kubectl apply -k .\
>>
secret/mysql-pass-fkgkd7hf27 created
service/wordpress created
Warning: spec.SessionAffinity is ignored for headless services
service/wordpress-mysql created
persistentvolumeclaim/mysql-pv-claim created
persistentvolumeclaim/wp-pv-claim created
deployment.apps/wordpress created
deployment.apps/wordpress-mysql created
PS C:\Users\Admin>
```

- Check if the Secret exists

```
PS C:\Users\Admin> kubectl get secrets
>>
NAME          TYPE      DATA  AGE
mysql-pass-fkgkd7hf27  Opaque   1    42s
PS C:\Users\Admin>
```

- Check that PersistentVolumeClaims are bound

```
PS C:\Users\Admin> kubectl get secrets
>>
NAME          TYPE      DATA   AGE
mysql-pass-fkgkd7hf27  Opaque   1    42s
PS C:\Users\Admin> kubectl get pvc
>>
NAME        STATUS    VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
mysql-pv-claim  Bound   pvc-3d9e93f8-4518-42f6-928c-a1f67c06cedb  20Gi       RWO          standard      <unset>          95s
wp-pv-claim   Bound   pvc-e51a71d7-4452-4a2a-bf35-05cf0a6cfaca  20Gi       RWO          standard      <unset>          95s
PS C:\Users\Admin>
```

- Check that Pods are running:

```
PS C:\Users\Admin> kubectl get pods
>>
NAME                  READY   STATUS    RESTARTS   AGE
hello-minikube-bbcb89c6c-876t1  1/1     Running   0          54m
nginx-66686b6766-47x5v         1/1     Running   0          26m
shell-demo                1/1     Running   0          23m
wordpress-598b5b87c4-h9n49     1/1     Running   0          2m30s
wordpress-mysql-869ff64b4d-vvgdt 1/1     Running   0          2m30s
PS C:\Users\Admin>
```

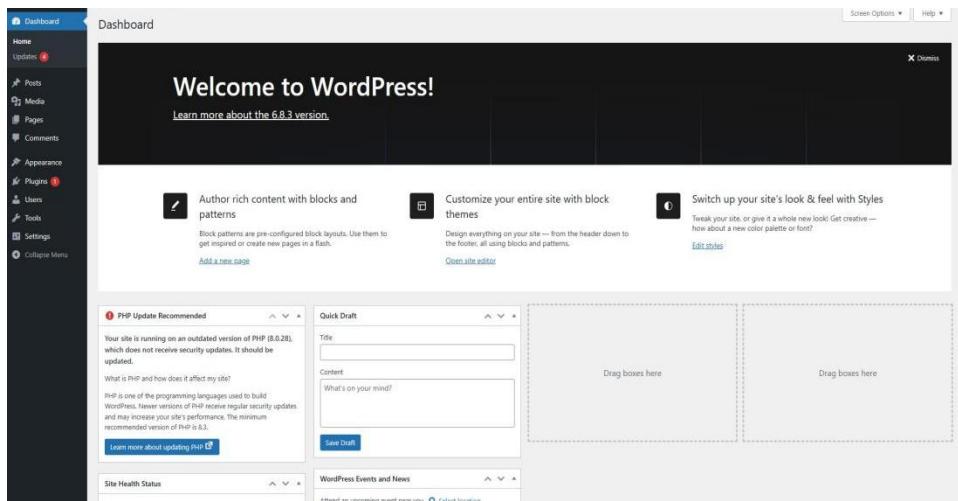
- Check that the WordPress Service exists

```
PS C:\Users\Admin> kubectl get services wordpress
>>
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
wordpress  LoadBalancer  10.104.40.72  <pending>      80:30965/TCP  2m53s
PS C:\Users\Admin>
```

- Get the WordPress URL

```
PS C:\Users\Admin> minikube service wordpress --url
>>
http://192.168.59.100:30965
PS C:\Users\Admin>
```

- Setup Wordpress



- Clean Up

```
PS C:\Users\Admin> kubectl delete -k .\  
>>  
secret "mysql-pass-fkgkd7hf27" deleted from default namespace  
service "wordpress" deleted from default namespace  
service "wordpress-mysql" deleted from default namespace  
persistentvolumeclaim "mysql-pv-claim" deleted from default namespace  
persistentvolumeclaim "wp-pv-claim" deleted from default namespace  
deployment.apps "wordpress" deleted from default namespace  
deployment.apps "wordpress-mysql" deleted from default namespace
```

