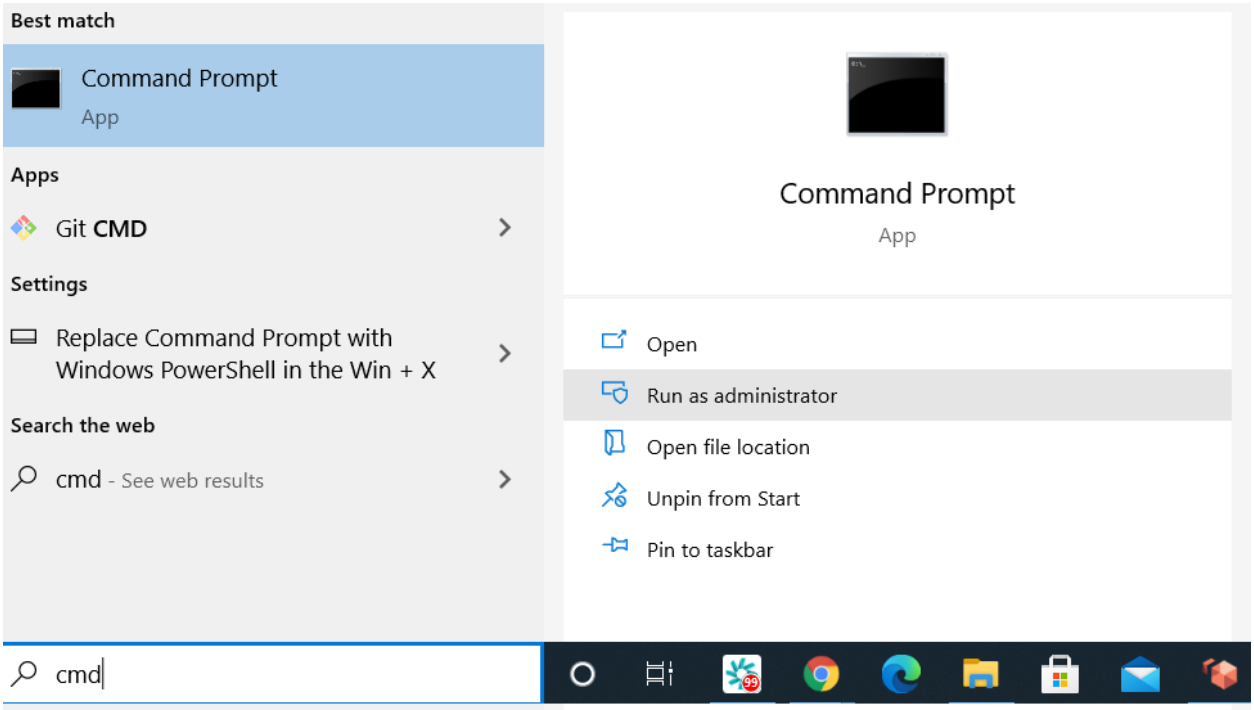
Experiment 02: Working With Kind

In the Foundation experiment/lab we installed Docker and KIND, validated our Docker installation, and now we'll move to using KIND. KiND/kind/KIND is a development tool for running a Kubernetes cluster as docker containers. This greatly simplifies working with cloud-native application modernization and microservices vs. running a full k8s cluster on your development environment. Because this runs in containers even a computer with 8GB of RAM can be used to effectively develop. KiND is one of two options that we'll use for running Kubernetes in our development environments, and is a tool developed by the Google team.

In Windows:

Open a Windows Command Prompt with "run as administrator"

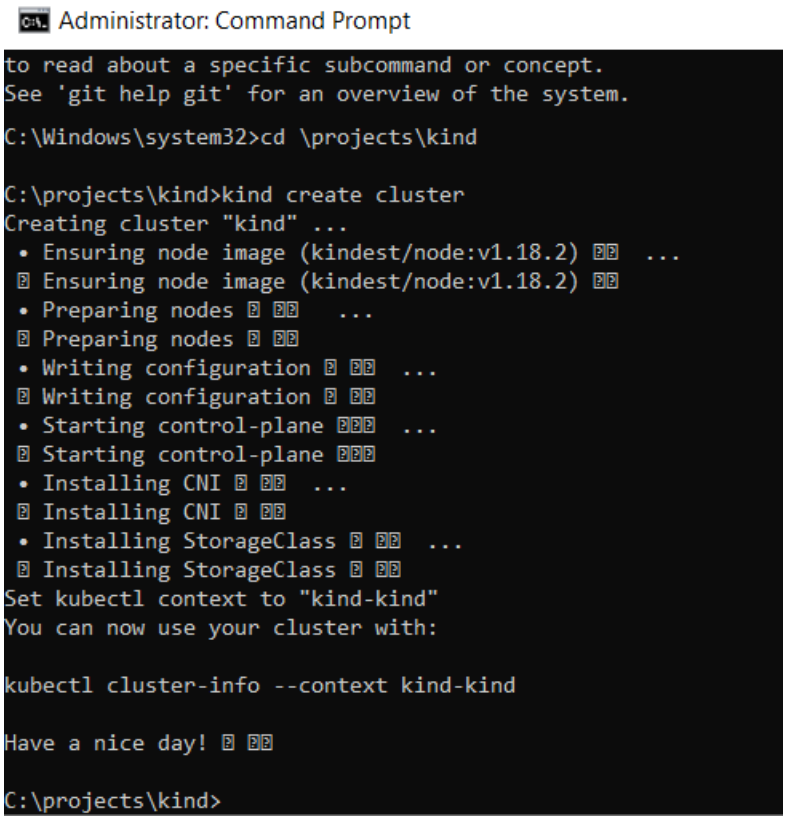


C:\> **mkdir c:\projects**

C:\> **mkdir c:\projects\kind**

C:\> **cd c:\projects\kind**

C:\projects\kind> **kind create cluster**



Super simple, execute “kind create cluster”, and go grab a refill on your favorite beverage while it spins for a few minutes depending on your network speed. Any delay in our initial cluster loading is in grabbing the images we need to build our cluster.

C:\projects\kind> **kind get clusters**

Kind

Check the kubectl client version
C:\>**kubectl version --client**
Client Version: version.Info{Major:"1", Minor:"19", GitVersion:"v1.19.0", GitCommit:"e19964183377d0ec2052d1f1fa930c4d7575bd50", GitTreeState:"clean", BuildDate:"2020-08-26T14:30:33Z", GoVersion:"go1.15", Compiler:"gc", Platform:"windows/amd64"}

Check the kind cluster we just created

C:\> **kubectl cluster-info --context kind-kind**
Kubernetes master is running at https://127.0.0.1:51089
KubeDNS is running at https://127.0.0.1:51089/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

Notice we need to use the --context or set our default context to kind-kind

The newest installation of KIND automagically creates the ~/.kube/config file targeted at kind-kind, so you won't have to create that file. If your ~/.kube/config file is not created you will have to do that.

C:\> **cd %USERPROFILE%**
C:\Users\wcsadmin> **mkdir .kube**
C:\Users\wcsadmin> **cd .kube**
C:\Users\wcsadmin\.kube> **notepad config**

The file should look similar to below, and of course the server port reference will be from the information that was returned from your "kubectl cluster-info"

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN5RENDQWJDZ0F3SUJBZ0lCQURB
TkJna3Foa2lHOXcwQkFRc0ZBREFWTVJNd0VRWURWUVFERXdwcmRXSmwKY201bGRHV
npNQjRYRFRJd01Ea3dPREF3TWpJME0xb1hEVE13TURrd05qQXdNakkwTTFvd0ZURVRNQk
VHQTFVRQpBeE1LYTNWaVpYSnVaWFJsY3pDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFB
RGdnRVBBRENDQVFvQ2dnRUJBTHJJCk9oYXpaZFJNamx1WFh5aTQ0cFRBbkxPZEZ5QlJV
b0FlRmZHUTlzRzd3UkJWMzNMaU5DUDRCMlNvUTZUb20zejMKK2pydUZsUWhZK0grQ1A3U
jByRGMwUmozbXdHM0dPckJYcnRBNzB6N2MrTUdMMm5QaEt0OTZ4Q0lNdVIvZ1NUbgpIMV
FuY0pkM3I1T3RGd1I2czdGbWxyb1RRNkFoVE5GZktBMTV2UHZ4Mmw2bEV3a0JGRlMrNjZ3
dERPT2dJZy9CCklwQkIzcWZ3bzYwbFRFUGorOEczbnBEYTdPNkVTNXNmZFlBZWJGTndIT2
1DL0cvdnN2Yks4MXV0anNNM1oyWmgKNUExNkxnUlpNOXk2RzI2blluYjNlekM2UWNMVG9Y
Vm94YU95d0l0MFJIaXBIVkN3U2hKRG9qcjU0ZTBTWXlncQo5MlFpeE1GWEphUndVc1M0eV
U4Q0F3RUFBYU1qTUNFd0RnWURWUjBQQVFIL0JBUURBZ0trTUE4R0ExVWRFd0VCCi93U
UZNQU1CQWY4d0RRWUpLb1pJaHZjTkFRUxCUUFEZ2dFQkFMVUxZMkNXTVlNSlJUb0dQ
ZGVWSmQ3MmZoL24KblFGWHRYT3dQbldmMElSR0grYTVwOVcyZytKRzQ4QjUwSUtZWFEr
VEZOTVZ0QUYrMDFWTFJRbzFQUEhudTM5NQppYlpHSnNybTBhWGVwOHFlSWF2Lzl0MUx
HcWk5dThkdFV6cjYvVEkwK2dUK2lEZlZ3bHRYSlZiVXN4Q2U1VE5UCmdQTW9GWjBOdDVC
THUzRkdrV0x3OXRNSlkyU0JNckNZZ0ZhZHg5bkVSMjd4L0NsdjEvR3picEhWRmJmYVlubmQ
KUFNZbVRiWncxYlgyZzJBdC9pQ1BSbDJGbXYzbXBTdENjaEppVDNkK0U5aEtpV2p4T0kvVD
FsN2E5UWxJTXdpdApldjZoMlJkUTQvZVdkeTFxY2hDYkZ6N0UyZEtvSXRONFFseFhQcVlMQ
1VnYzB4TEFGK2kxdkV1bDM3Zz0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo=
    server: https://127.0.0.1:51089
  name: kind-kind
contexts:
- context:
    cluster: kind-kind
    user: kind-kind
  name: kind-kind
current-context: kind-kind
kind: Config
preferences: {}
users:
- name: kind-kind
  user:
    client-certificate-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUM4akNDQWRxZ0F3SUJBZ0lJU3VyaX
```

QxdGJKNnN3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmla
WEp1WlhSbGN6QWVGdzB5TURBNU1EZ3dNREl5TkROYUZ3MHlNVEE1TURnd01ESXlORFJ
hTURReApGekFWQmdOVkJBb1REbk41YzNSbGJUcHRZWE4wWlhKek1Sa3dGd1lEVlFRREV
4QnJkV0psY201bGRHVnpMV0ZrCmJXbHVNSUlCSWpBTkJna3Foa2lHOXcwQkFRUZBQU9
DQVE4QU1JSUJDZ0tDQVFFQXo1K1d0L242RGtEbm5QblcKMFVmb3kxM1ArSldjQi9BMXdBT
TU3L2RSREpIZ1g4N1lqOFRibkRDdUNhcElxVFY0ZjBqdW9JdDRDZmNKQUxMVgpXTWxhYV
hncmljSXgxejdBdzlEUDVNRHM2VkdJcWRKaGc1Z0ROQWVYdTIzZUNuaUgxVDBnUFJxNjdK
WHY4YW9VCi9YblY3UFdXNDVGbDlvd0NwSEc5Vlp4MUNIaVVOd1RVb09pd29oYzhpWmh6Q
zNHdS9CZ3FRbDR5K3hMVU0wTDkKWXZEY0NBU2dYdEhCQkU3TWN2SWxIQWFFdmdKVV
k0c1MvMUhZMytZQmZVcCtzUzZjcHBKWFpveXhaTGdaMm51YgpTYmVhQ0J0Wk9wbGR3YT
gvV29SbGtrRkFpK0RURkt3cEZmTC9DYkxqOHd3QTVXdnNtWVdyWkFRZGE2aTBKQ2k5Cmd
aMzhCUUlEQVFBQm95Y3dKVEFPQmdOVkhROEJBZjhFQkFNQ0JhQXdFd1lEVlIwbEJBd3dD
Z1lJS3dZQkJRVUgKQXdJd0RRWUpLb1pJaHZjTkFRUxCUUFEZ2dFQkFCOS9FL3o1cnNBe
DNoaDVxVlo3ZUswTy95bUtES05SOFVDTApwc21kaVl2dm1qZEVkc0I2WFhWM2dMeWQ1Tm
F0SSt2WU5rbS9YODlqNTlURXNVeGhtS2V2SEorQmRZWXBPcEQ5CmY3Y3dBNnNIS2lEcGJ
UMVVuajRoeHJoZyt6VEFXN3hleHZEam8vdG5xazg1dDR1bzVLdFVrRjRFNEN1QXQ3TFUKUj
JaTlBta2UwdTRsUWgzaFU5Slg1bzY5N3FHb21EMkZTeDdvb212eXdDMS8rbFBlVkNURk95U2
pPbXEya001VApiZCtRbmhoeVVTRXJheC9sbnV5SDhkUlJYZkZWelk1WElLNS81RnBBMFFUU
kltRXJUVExSem5mcktNQXVaS3VTCjVFL0d4Z083MU1vZGN3ajM2NWZKWE9JT1c0UlZ4TXJa
bEpnQVZIa0lGUitQSUV0blErRT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo=
    client-key-data:
LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFcEFJQkFBS0NBUUVBejUrV3Q
vbjZEa0RublBuVzBVZm95MTNQK0pXY0IvQTF3QU01Ny9kUkRKSGdYODdZCmo4VGJuREN
1Q2FwSXFUVjRmMGp1b0l0NENmY0pBTExWV01sYWFYZ3JpY0l4MXo3QXc5RFA1TURzNlZ
HSXFkSmgKZzVnRE5BZVh1MjNlQ25pSDFUMGdQUnE2N0pYdjhhb1UvWG5WN1BXVzQ1Rm
w5b3dDcEhHOVZaeDFDSGlVTndUVQpvT2l3b2hjOGlaaHpDM0d1L0JncVFsNHkreExVTTBMO
Vl2RGNDQVNnWHRIQkJFN01jdklsSEFhRXZnSlVZNHNTCi8xSFkzK1lCZlVwK3NTNmNwcEp
YWm95eFpMZ1oybnViU2JlYUNCdFpPcGxkd2E4L1dvUmxra0ZBaStEVEZLd3AKRmZML0NiT
Go4d3dBNVd2c21ZV3JaQVFkYTZpMEpDaTlnWjM4QlFJREFRQUJBb0lCQUVzYmxFNWhvO
C9jTXUxYQpoQmVaUitHcHdqNVBBTzd1T3NPSFowSWoyYklPWTNqRlB4cGpRSDYwTFlGW
mxJZUJ6R0ZmWk5PM0lHbWFjQ3RNCmhsbGtIY3pocC81aHZkMzcyWWY4MWZnT3dxVjAxV
mQ1djhUM0ROR1puWTQwSklydEoxWkFrcFVJUW02cm51MXgKZGI1c2dMTUQ5TjNHRDNpd
EZaZWZmYnFtcXIreEYvN2tTQ0ZKSEpyR0JoMk9GT3JSTXdxUXREaFBJWS9DTzN6VQoybV
BOTXN2Rk5JQjBoK2lKVXEvWEpwOHAvd3hmYkYva3FuRXBEYTZhaTFXT21NdzhuQjFtRFZq
SjR3TWtZdEkwCjZrTWFTdHYvVVBBeVIvZzFvV0ZTeGFGcGs4OEh6UUp1ZnpXamZ6c2Z4bD
N4ZnNwY1FKNzNZUlZzb29kRFdWTEUKRkwxVFRBRUNnWUVBODVVWklMcHp3M0lSa21rQ
kJOT0VLYk5wWWNOZEYyOUNDOGNuYWJRMFdMNVlqUWEzdjBYMgo5YXdiVllQVXBCck1w
ZUY0VlNPalhSTWpIazkwZDIvM1dwRlkxdjJVUjBSTXdrYy9MVXNIVjJ6S2xUczZmanZFCi8vU0
VTRm9FS2ZmWmkzbjAzUm81R3gvdWZxK1YwUnl6L0pNMzlKeDI5N0o5NHVHdE1FSEVCY1
VDZ1lFQTJqVXoKK1VzSnRBQ2NwNkJ3TDNOV1h3U29aMk9nbzZnYjBOclpTUW9YbnNwdGt0
UmF0R3NER3JTWG1CeGJNRUloVklERQpKbDBNTGc3QnFEaDcwVklBbmNJWmQzQkhNd0
NYMmgzNDN6NmJZSTZ1RFBSU1k0amd4WHB0djBaVDdIV1ZDVFgwCk9TWnArYkpuMFpJb3
gwSzBHc0FIcDk2WThIZnRReUoyU2FOendVRUNnWUVBbytzWFVONEluMk01RGdVWnlXeX
EKQ1FJU2pkYlI0NzVjZk42VjJGMkx5Smkzc0pmdnVZbFV5emo1NEE5cVh0RW1IUTloVWpJO
GNwczVTK0YrYUEzeQpNSUdWZm9DQmM0QTBBNTI4bHpkaGhtVFE0NkpMRjc0VE1ZZ1VL
VGhpaXZlZTcyeXY2c2NGM1FvZERpWU5OUDhTCjVJc1I0Y3dWaDFVWHdFSE1zYWZnU1Yw
Q2dZRUFpUmY2RW5zeG1uVHo5MkVXZXNsMUQzZW1zbVptcTh2WHhnMXAKakxrWmcvdG
NnbTZHbW1uTlpuN2w2M3lOVWpHS0xVUkZlSERuVVJ5V1VURkRvWXhxdExNWk92QkEyMn
RZL0lIdQpOWnhwRkc0d0xoVm1tZ0NLYjZmdXdjald1MjVZZDVQOVg5YWhxRzZOU1o4Um5iZ
HIzbzZkZ0x1YXpTSWI0QjBMCndsSStUTUVDZ1lCSTFrek5EeHhWUFVucERleXNnWmxlS3Qw
cDltTzhKWjlGcU5UM044eXY1OEZnbitkSFU2WmYKWXZUNDdkK3FxdHRYREJEa1ZlWkJ4Kz

VSdW5kWlgxM3NKUkg2NUhVeE5QczBkdGVLa3RObnh5Zm9ubEFZTU13QgorWXBlMFVUT2
xaK3FOT3d1UzFkZHM2MU14SVZWRlJxTDVzVGlVZGdkU3VLenU3ZW41R2hIY3c9PQotLS0t
LUVORCBSU0EgUFJJVkFURSBLRVktLS0tLQo=

Create a folder under your projects or profile folder and switch directory to our new folder

C:\projects\kind> **mkdir kind-wp**

C:\projects\kind> **cd kind-wp**

Create kustomization.yaml with the following content

C:\projects\kind\kind-wp> **notepad kustomization.yaml**

In MacOS:

~: kubenerd $ **cd ~**

~: kubenerd $ **mkdir projects/kind**

~: kubenerd $ **cd projects/kind**

For both Windows and MacOS:

~: projects/kind $ **kind create cluster**

~: projects/kind $ **mkdir kind-wp**

~: projects/kind $ **cd kind-wp**

~/projects/kind/kind-wp $ **vi kustomization.yaml**

Paste the following content into our file and save or alternatively you can download this from the project GitHub repo here under labs folder or by cloning the repo locally. This uses a Secrets Generator. Since Kubernetes v1.14, kubectl supports managing objects using Kustomize. Kustomize provides resource Generators to create Secrets and ConfigMaps. The Kustomize generators should be specified in a **`kustomization.yaml`** file inside a directory. After generating the Secret, you can create the Secret on the API server in our Kubernetes cluster with **`kubectl apply`**

The following is an example that we'll use for using Kustomize to create a MySQL Database password.

```
secretGenerator:
- name: mysql-pass
  literals:
  - password=VerySecure2020
resources:
  - mysql-deployment.yaml
```

- wordpress-deployment.yaml

The formatting of a YAML file is key to the stanzas. The indenting in a YAML file is how it groups the information, similar to Ruby coding. This eliminates the need for curly braces that you see in JSON or similar types of groupings in popular programming languages.

Download the following two resource files to our (Windows) **c:\projects\kind\kind-wp** or (MacOS) **~/projects/kind/kind-wp** folders

If you don't have wget for your Windows environment, you can install with chocolatey

D:\projects\kind\kind-wp> **choco install wget**

Chocolatey v0.10.15

Installing the following packages:

wget

By installing you accept licenses for the packages.

Progress: Downloading Wget 1.21.1.20210323... 100%

Use our wget to pull the files to complete this experiment.

```
$ wget https://kubernetes.io/examples/application/wordpress/mysql-deployment.yaml
```

```
$ wget https://kubernetes.io/examples/application/wordpress/wordpress-deployment.yaml
```

Copy or move the two resource YAML files that we downloaded to the kind-wp folder that you created, if there were not saved there.

From the kind-wp folder run the kubectl apply

C:\projects\kind\kind-wp> **kubectl apply -k ./**

Run the following commands to inspect the deployment

C:\projects\kind\kind-wp> **kubectl get secrets**
C:\projects\kind\kind-wp> **kubectl get pvc**
C:\projects\kind\kind-wp> **kubectl get pods**

Once the "get pods" shows status as Running, rather than ContainerCreating we'll check the services for wordpress

C:\projects\kind\kind-wp> **kubectl get services wordpress**

Try accessing the normal default wordpress at the following port.

Open http://localhost:8080/ in your web browser

This will result in a 404, and we'll correct that defect in our next update

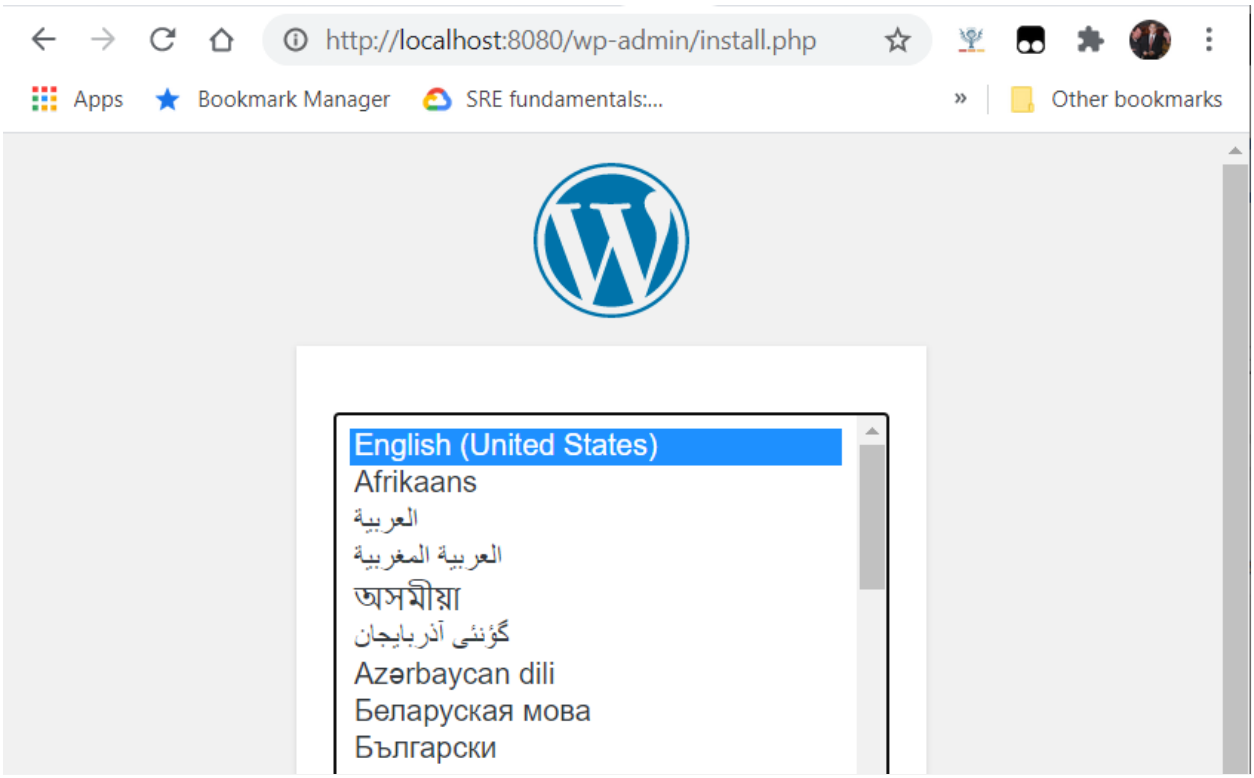
Note: We see that this is refused but recall that we see our service, but have not enabled a port forward to get to that WordPress instance

Create a port forward to allow us to access our Wordpress environment

~/projects/kind/kind-wp$ **kubectl port-forward svc/wordpress 8080:80**

Now reopen or refresh

http://localhost:8080/

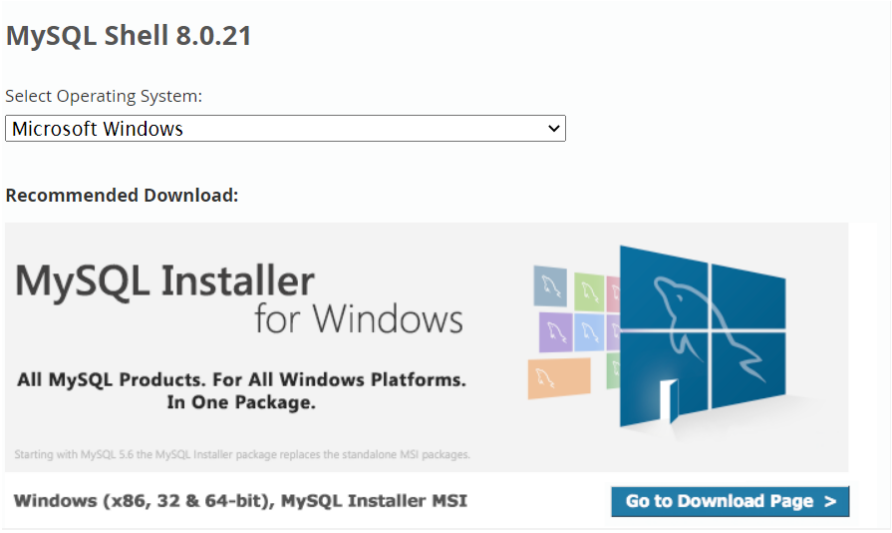


For MacOS:

$ brew install mysql-client

For Windows: Download the MySQL Shell if you don't already have it from the below URL

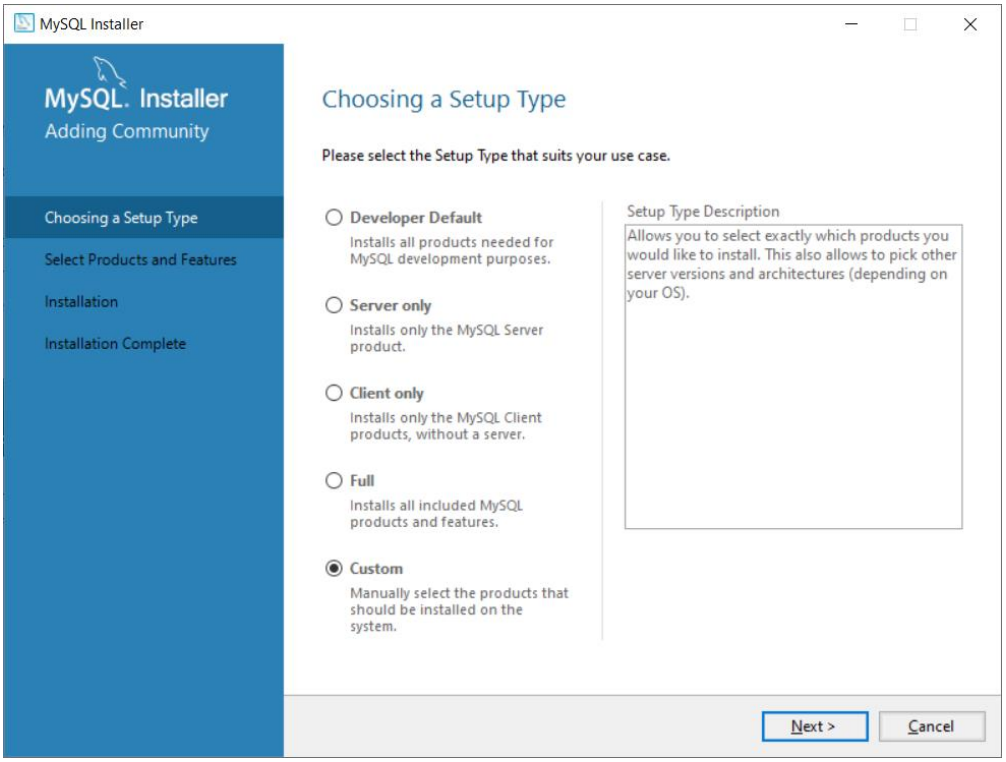
https://dev.mysql.com/downloads/shell/



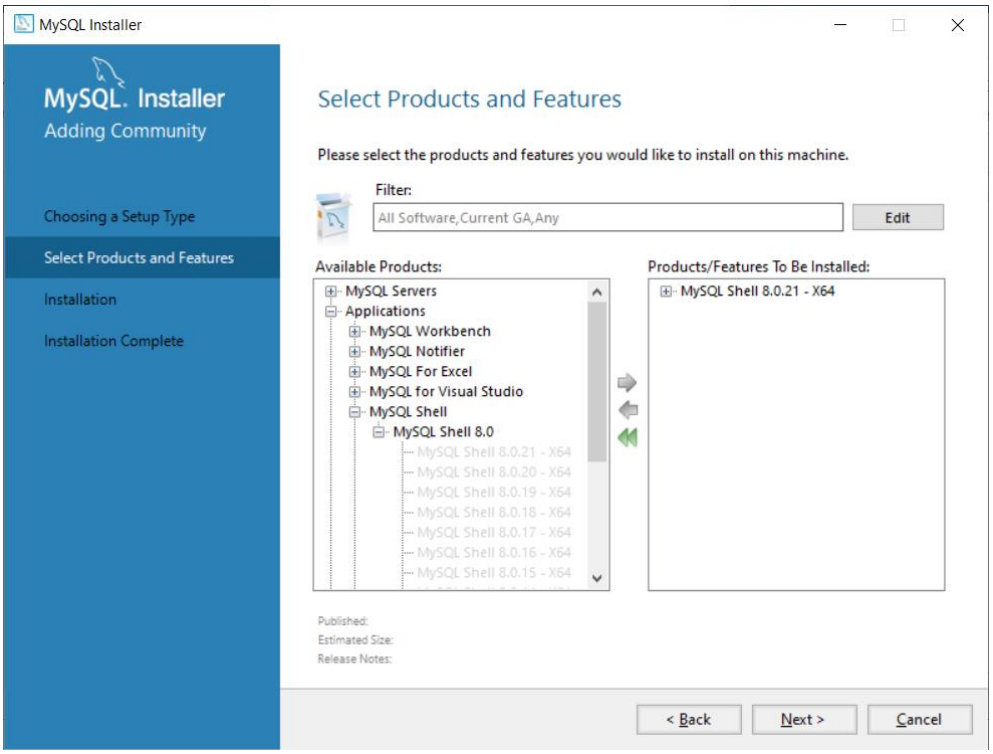
If you have an Oracle account, you can login or you can select "No thanks, just download"

Install the MySQL Shell, by running the downloaded installer

Select the "Custom" unless you prefer to install the server and such locally.

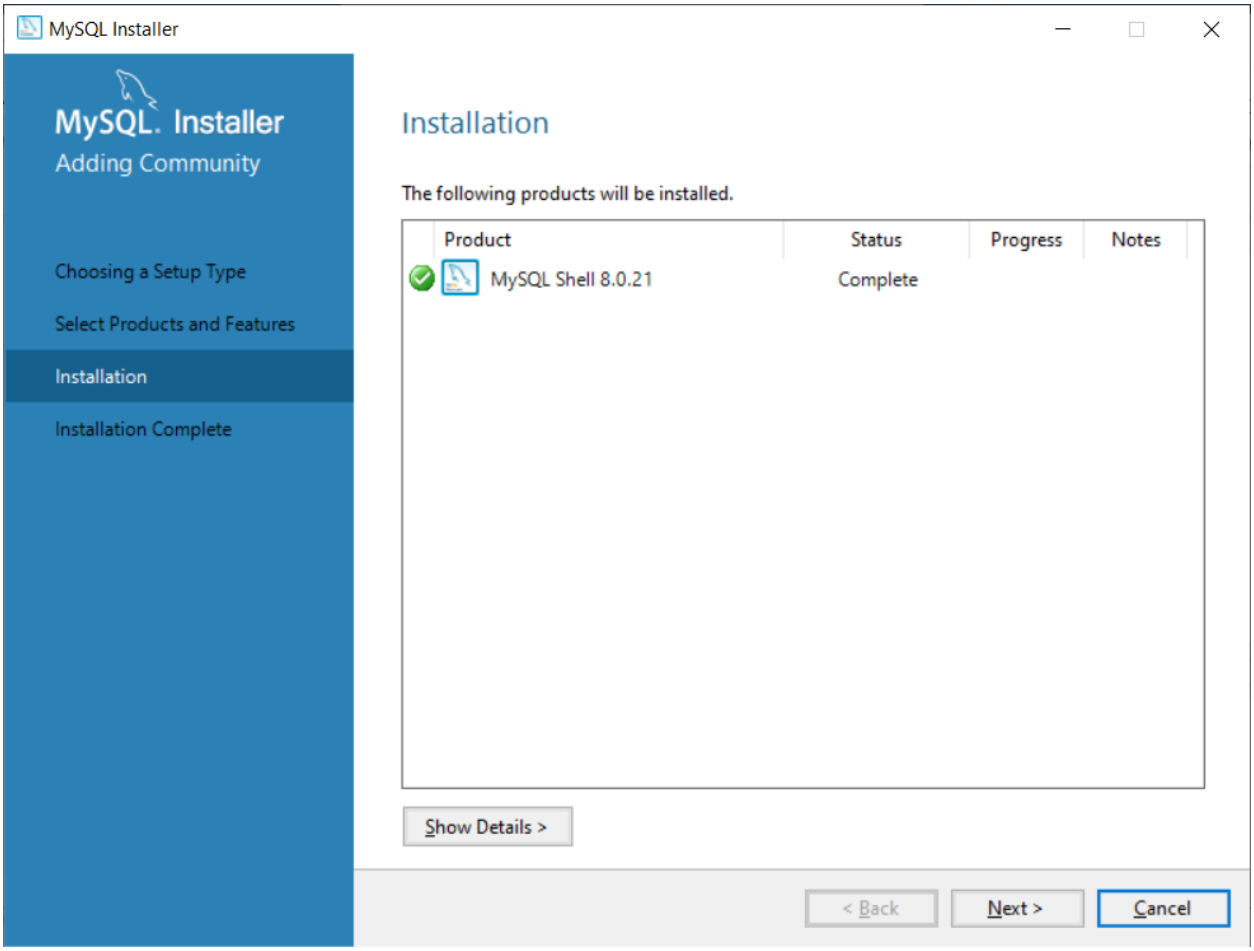


Select “Next”

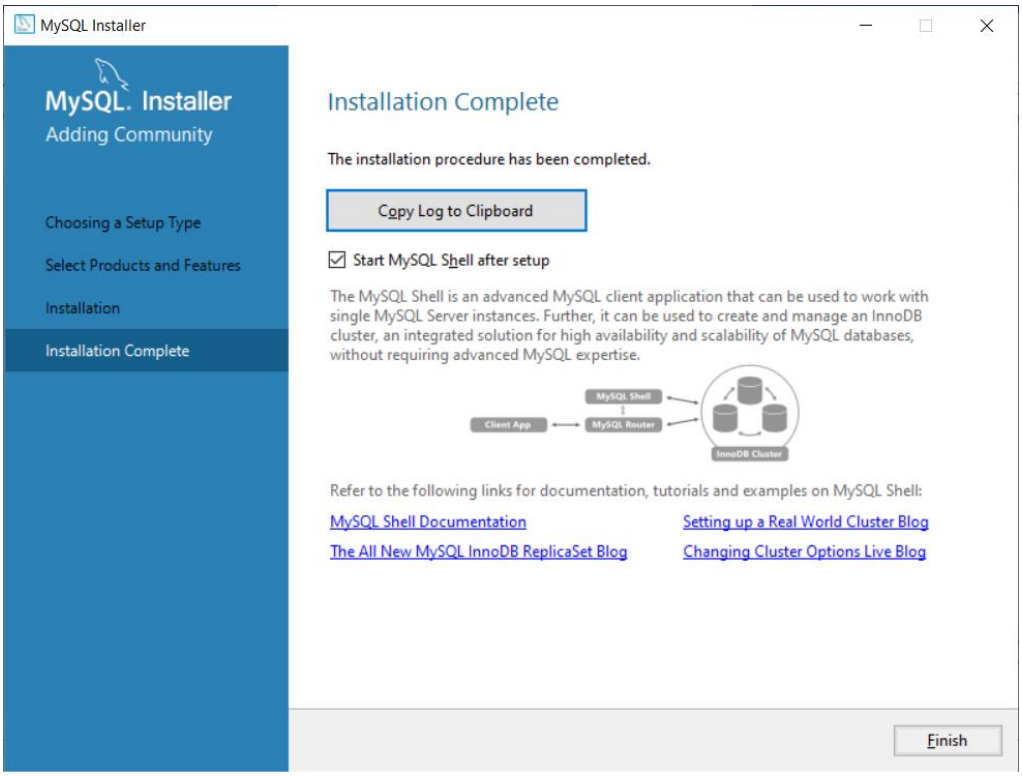


Expand the options under Applications -> MySQL Shell -> MySQL Shell 8.0 and then select MySQL Shell 8.0.21, selecting the top arrow to add that to the Products/Features To Be Installed list.

Select “Execute” to download the MySQL Shell


MySQL Installer
MySQL. Installer
Adding Community
Choosing a Setup Type
Select Products and Features
Installation
Installation Complete
Installation
The following products will be installed.
Product
Status
Progress
Notes
MySQL Shell 8.0.21
Complete
Show Details >
< Back
Next >
Cancel

Select “Next”



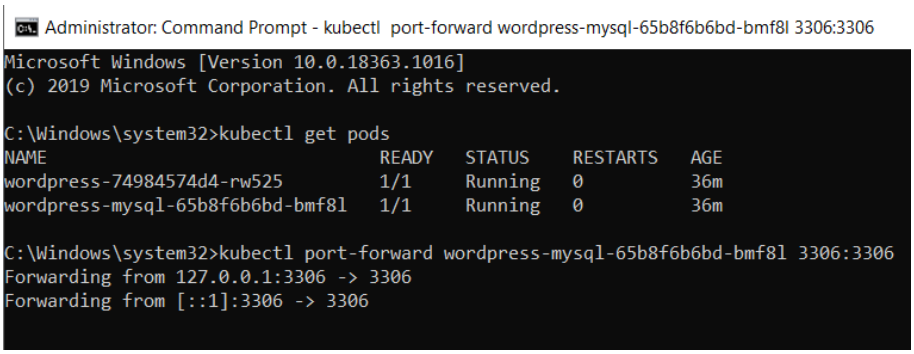
Select “Finish”

Connectivity into our containerized DB

In a command shell check your pods and create a port forward for MySQL to connect to port 3306

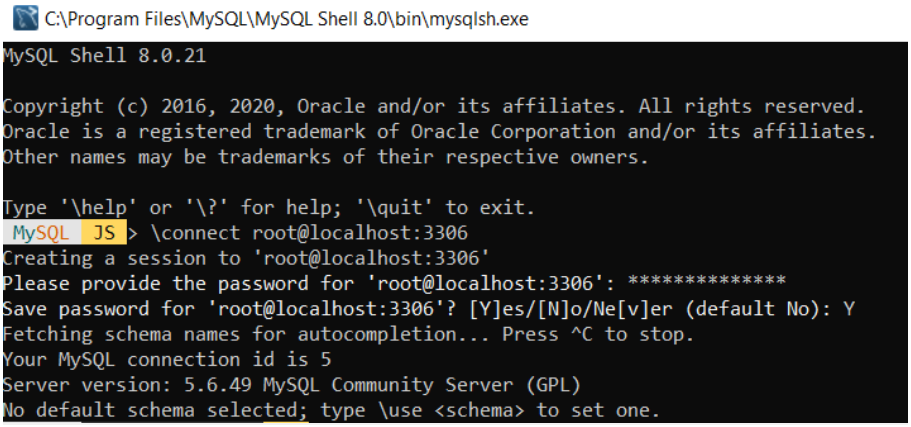
~/projects/kind/kind-wp $ **kubectl get pods**

C:\projects\kind\kind-wp> **kubectl port-forward wordpress-mysql-65b8f6b6bd-bmf81 3306:3306**

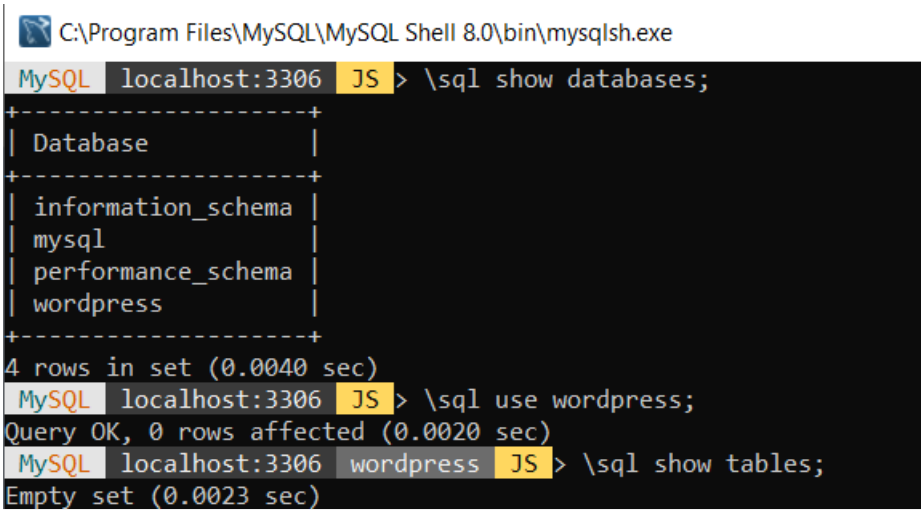


From your MySQL Shell you'll now be able to connect to the database running in the container for the WordPress application

Enter \connect root@localhost:3306



You can review your **kustomization.yaml** to find the MySQL password set or if you used the default noted in this lab, it will be **VerySecure2020**



Enter **\sql show databases;**

The list of databases in MySQL is listed, including our WP DB

Enter **\sql use wordpress;**

This selects the WordPress DB that we've created in our deployment

Enter **\sql show tables;**

You'll notice that no tables are created, since we've not created our WorkPress instance, as yet.

Go back to the WordPress admin screen, fill in the information and select "Install WordPress"

Please provide the following information. Don't worry, you can always change these settings later.

Site Title KIND-WP

Username georgeniece

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password VerySecure2020 Hide

Medium

Important: You will need this password to log in. Please store it in a secure location.

Your Email george.niece@digitaltransform

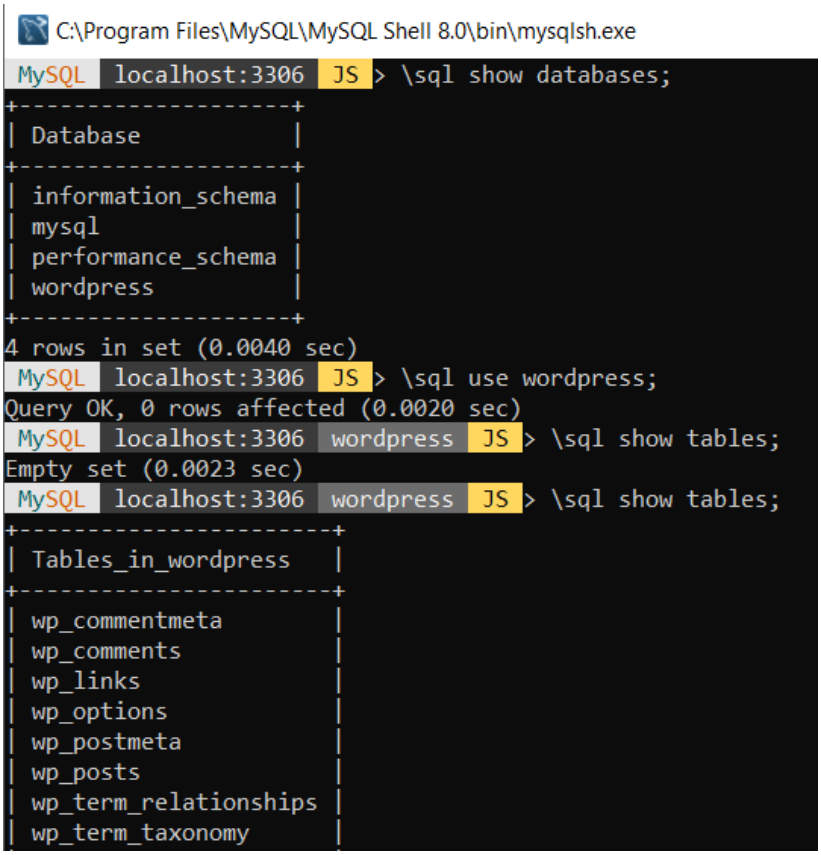
Double-check your email address before continuing.

Search Engine Visibility Discourage search engines from indexing this site

It is up to search engines to honor this request.

Install WordPress

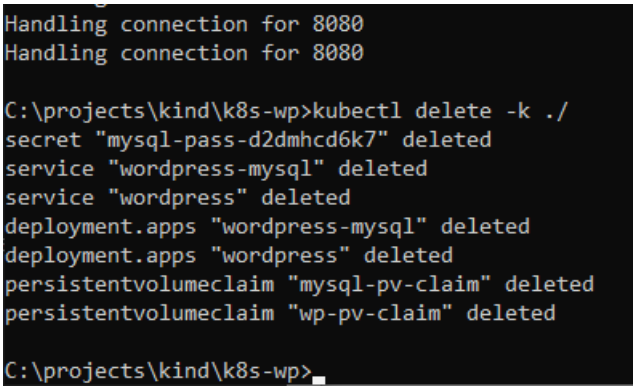
Return to your MySQL Shell and look at the tables again in our WordPress DB



Lab complete, kill the port-forwarding session by Ctrl-C

Run the following command to delete your Secret, Deployments, Services and PersistentVolumeClaims from our kind-wp folder in a command prompt (Windows) or terminal (MacOS)

```
C:\projects\kind\kind-wp> kubectl delete -k ./
```




```
C:\projects\kind\kind-wp> kind help delete
Deletes one of [cluster]

Usage:
  kind delete [command]

Available Commands:
  cluster     Deletes a cluster
  clusters    Deletes one or more clusters

Flags:
  -h, --help   help for delete

Global Flags:
      --loglevel string   DEPRECATED: see -v instead
  -q, --quiet             silence all stderr output
  -v, --verbosity int32   info log verbosity

Use "kind delete [command] --help" for more information about a command.

C:\projects\kind\kind-wp>kind delete cluster --help
Deletes a resource

Usage:
  kind delete cluster [flags]

Flags:
```

```
  -h, --help                help for cluster
      --kubeconfig string   sets kubeconfig path instead of $KUBECONFIG or
$HOME/.kube/config
      --name string         the cluster name (default "kind")

Global Flags:
      --loglevel string   DEPRECATED: see -v instead
  -q, --quiet             silence all stderr output
  -v, --verbosity int32   info log verbosity

C:\projects\kind\kind-wp> kind delete cluster --name Kind
Deleting cluster "Kind" ...
```

Now we can create a multi node cluster by inputting a config flag into our “kind create cluster” using the config flag and modifying the name with the name flag.

First we create a new configuration yaml file using your favorite text editor

$ **vi grogu.yaml**

Insert the following

```
# six nodes (three workers & three leaders) cluster config
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
- role: control-plane
- role: control-plane
- role: control-plane
- role: worker
- role: worker
- role: worker
#end of file
```

Save our file and now run that using the kind create with optional flags

$ **kind create cluster --name grogu --config grogu.yaml**

And not surprisingly the same in Windows

D:\projects\kind>**kind create cluster --name grogu --config grogu.yaml**

Creating cluster "grogu" ...
 • Ensuring node image (kindest/node:v1.21.1) 🖼 ...
 ✓ Ensuring node image (kindest/node:v1.21.1) 🖼
 • Preparing nodes 📦 📦 📦 📦 📦 📦 ...
 ✓ Preparing nodes 📦 📦 📦 📦 📦 📦
 • Configuring the external load balancer ⚖️ ...

✓ Configuring the external load balancer ⚖️
• Writing configuration 📜 ...
✓ Writing configuration 📜
• Starting control-plane 🕹️ ...
✓ Starting control-plane 🕹️
• Installing CNI 🔌 ...
✓ Installing CNI 🔌
• Installing StorageClass 💾 ...
✓ Installing StorageClass 💾
• Joining more control-plane nodes 🎮 ...
✓ Joining more control-plane nodes 🎮
• Joining worker nodes 🚜 ...
✓ Joining worker nodes 🚜
Set kubectl context to "kind-grogu"
You can now use your cluster with:

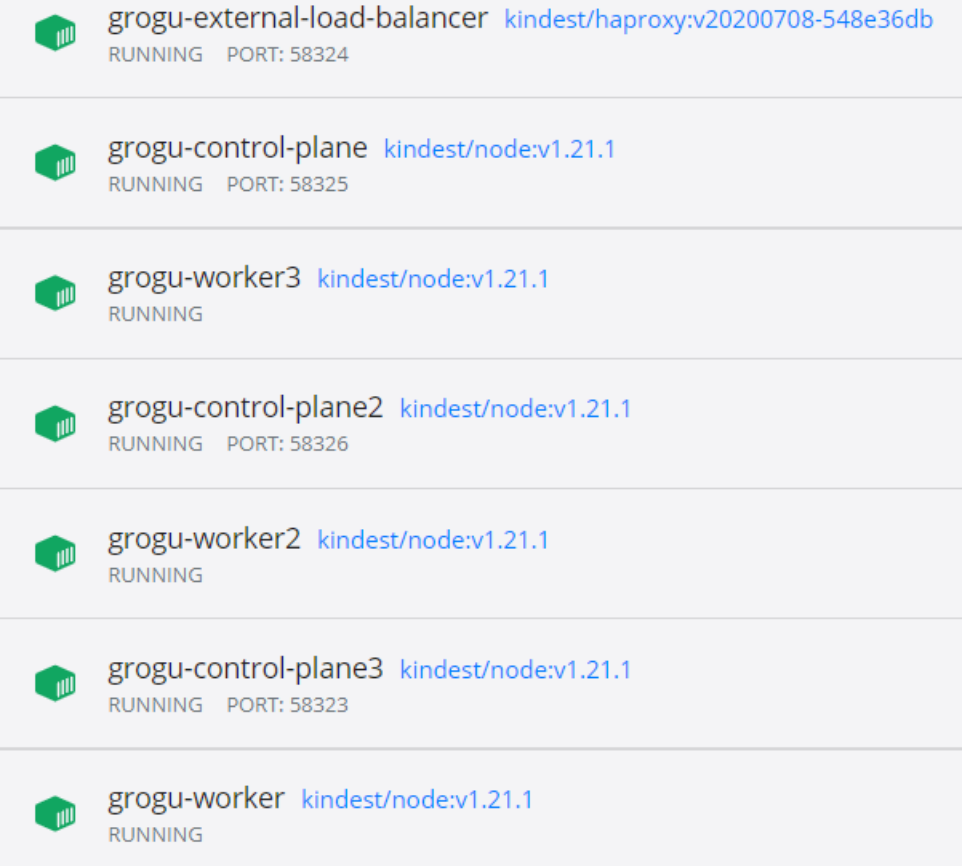
kubectl cluster-info --context kind-grogu

Have a nice day! 👋

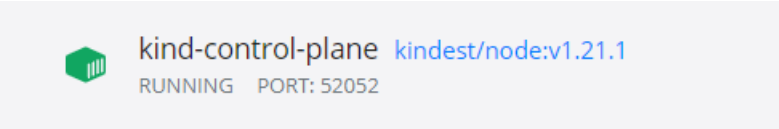
D:\projects\Microservices_On_Kubernetes\labs>kubectl get nodes

NAME	STATUS	ROLES	AGE	VERSION
grogu-control-plane	Ready	control-plane,master	3m20s	v1.21.1
grogu-control-plane2	Ready	control-plane,master	2m16s	v1.21.1
grogu-control-plane3	Ready	control-plane,master	77s	v1.21.1
grogu-worker	Ready	<none>	45s	v1.21.1
grogu-worker2	Ready	<none>	48s	v1.21.1
grogu-worker3	Ready	<none>	45s	v1.21.1

Looking at this environment in Docker Desktop we see



For the original single node cluster, created initially, we'd only see the single node

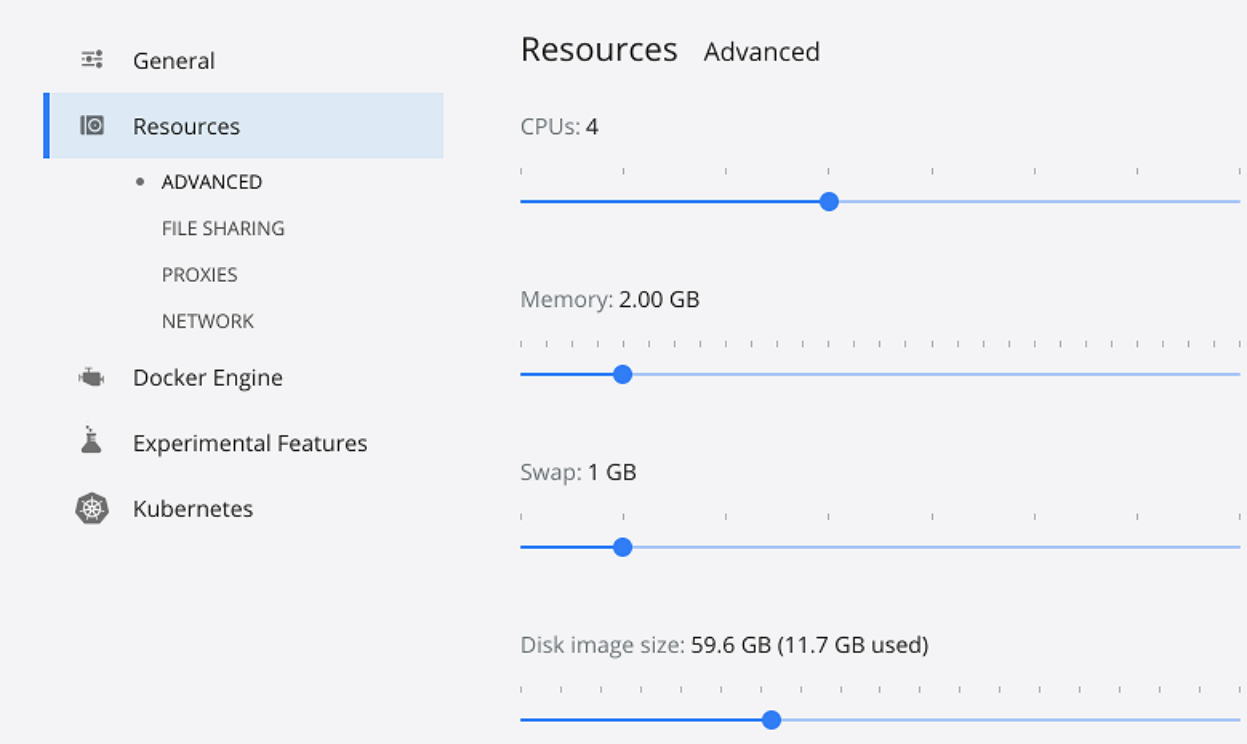


Now we'll delete the additional cluster with the following

D:\projects\kind> **kind delete cluster --name grogu**

Deleting cluster "grogu" ...

For updating our resource settings in Kind running in Docker Desktop for MacOS we'd go to the Settings -> Resources -> Advanced. These settings are not exposed in Docker Desktop for Windows and you to adjust those settings by setting priority on the Launch for Docker Desktop. For some task like building images the default settings won't work, i.e. building images typically requires 6GB RAM allocation.


General
Resources
ADVANCED
FILE SHARING
PROXIES
NETWORK
Docker Engine
Experimental Features
Kubernetes
Resources Advanced
CPUs: 4
Memory: 2.00 GB
Swap: 1 GB
Disk image size: 59.6 GB (11.7 GB used)