IBM

# Lab: Deploy a scalable WordPress Implementation on a Kubernetes Cluster

## Overview

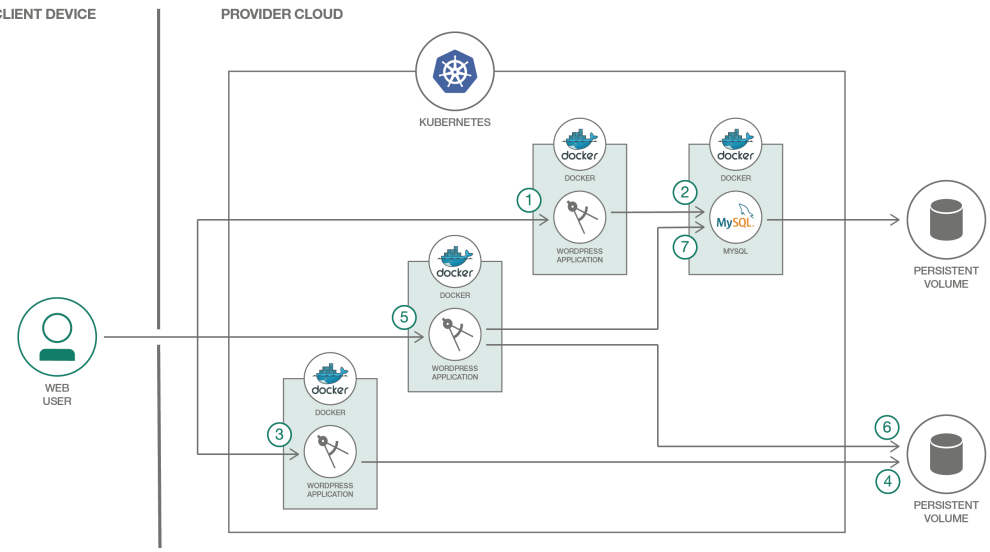
In this lab, you will deploy a WordPress implementation on a IBM cloud Kubernetes Cluster. WordPress is the world's most popular website management and blogging system, supporting more than 60 million websites. At its core, WordPress is built on one of the most common web programming languages, PHP, and uses MySQL as its back-end database. Kubernetes, the open-source container management system, is one of the top 10 GitHub projects based on number of unique developers contributing code. The challenge for developers is how to bring these two giant open source projects together to provide maximum benefits

## Prerequisites

Following are the prerequisites software for this lab:

- IBM Cloud CLI
- Kubernetes CLI
- Container Service Plugin
- A Pay-As-You-Go or Subscription IBM Cloud account

## Architecture




1. The user interacts with WordPress via the web interface. Each WordPress container will respond to its users via HTTP/HTTPS.
2. When a user posts to any WordPress container, WordPress will typically post the changes to the MySQL database. The MySQL database stores the post data into persistent disks to maintain security. In addition to a MySQL container, you can also use the Compose MySQL service from Bluemix. After authentication and authorization are complete, WordPress user information such as password (encrypted with MD5) and email address

are created and stored in MySQL. Website, blogs, tags, categories, and other data are also stored in MySQL.

3. The user can also upload themes, plugins, images, and documents. Non-textual data such as PDFs,videos, and MP3s, can also be uploaded.
4. Themes, plugins, PDFs, videos, MP3s, etc. are stored in a persistent volume attached to the WordPress pods.
5. The user accesses the WordPress website or blog. The WordPress core (that is, the WordPress "brain") calls the required PHP scripts, starting with index.php.
6. WordPress reaches out to the MySQL database to retrieve the website, blogs, tags, categories, and so on.
7. TheWordPress core then retrieves the themes, documents, images, etc. from the persistent volume, combines it with data retrieved from the database, and presents the page to the user.

### Steps to Create Service and deploy on to the Cluster

Step 1: Login to IBM Cloud
open the command terminal and execute the following command to login to IBM Cloud, when it prompts for email, provide a valid Email which is used to register for IBM Cloud and enter valid password.

```
bx login -a https://api.ng.bluemix.net
API endpoint: https://api.ng.bluemix.net
Email> <Enter valid a Email>
Password><Enter valid Password>
Select an account (or press enter to skip):
1. Demo Account
Enter a number>1

Execute "bx target –cf" to set the Organization and Space
bx target –cf
```

Step 2: Initialize Container Service Plugin

```
bx cs init
Using default API endpoint: https://containers.bluemix.net
OK
```

Step 3: Create Kubernete Cluster and initialize Kubernete Client Configuration

```
bx cs cluster-create --name <Cluster Name>
Creating cluster...
The machine-type flag was not specified. So, a lite cluster with default
parameters will be created. To customize the parameters, create a standard
cluster and include all required flags.
OK


Check the Cluster status before procced to next step, Cluster state should in
"normal" State
bx cs clusters

OK
Name         ID         State    Created    Workers   Datacenter   Version
```

```
cluster   695bfab1a   normal   27 minutes ago   1          hou02      1.8.6_1504
```

Initialize the Kubernete Client configurations.

```
bx cs cluster-config <Cluster Name>
OK
The configuration for mydemocluster was downloaded successfully. Export
environment variables to start using Kubernetes.

export KUBECONFIG=/Users/rameshpoomalai/.bluemix/plugins/container-
service/clusters/mydemocluster/kube-config-hou02-mydemocluster.yml
```

Set the Configuration to environment variable KUBECONFIG, use the Path location from previous command output, You execute the export command in the above results as well.

```
export KUBECONFIG=<Path location of previous command results>
```

Verify the Configuration by executing "**kubectl version**", should return the valid server version

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"7", GitVersion:"v1.7.3",
GitCommit:"2c2fe6e8278a5db2d15a013987b53968c743f2a1", GitTreeState:"clean",
BuildDate:"2017-08-03T07:00:21Z", GoVersion:"go1.8.3", Compiler:"gc",
Platform:"darwin/amd64"}
Server Version: version.Info{Major:"1", Minor:"8+", GitVersion:"v1.8.6-
4+e5b2250ba66db9", GitCommit:"e5b2250ba66db94bf5c6b60196aec6e577a005b1",
GitTreeState:"clean", BuildDate:"2018-01-08T08:11:01Z", GoVersion:"go1.8.3",
Compiler:"gc", Platform:"linux/amd64"}
```

## Step 4: Setup Mysql Secrets

Create password file

```
echo "changeme" >password.txt
tr -d '\n' <password.txt >.strippedpassword.txt && mv .strippedpassword.txt password.txt
kubectl create secret generic mysql-pass --from-file=password.txt
```

## Step 5: Create Local Persistent Volumes

Change your working directory to the git repository local folder, Spec files are available locally, also it can be downloaded from git repository as well.

```
kubectl create -f local-volumes.yaml
```

## Step 6: Create Services and deployments for WordPress and MySQL

```
kubectl create -f mysql-deployment.yaml
kubectl create -f wordpress-deployment.yaml
```

run the following commend to verify PODS, POD's status should be running, proceed to next step once its status is changed to Running.

```
kubectl get pods
NAME                               READY     STATUS    RESTARTS   AGE
wordpress-76b66d5644-qzc7t         1/1       Running   0          2m
wordpress-mysql-5cdbc78858-qvt5h   1/1       Running   0          2m
```

Step 7: Check the Deployment status

```
kubectl get deployments
NAME              DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
wordpress         1         1         1            1           23h
wordpress-mysql   1         1         1            1           23h
```

Step 8: Access the Application

Get the Public IP address, Execute the below commands and note down the public IP address

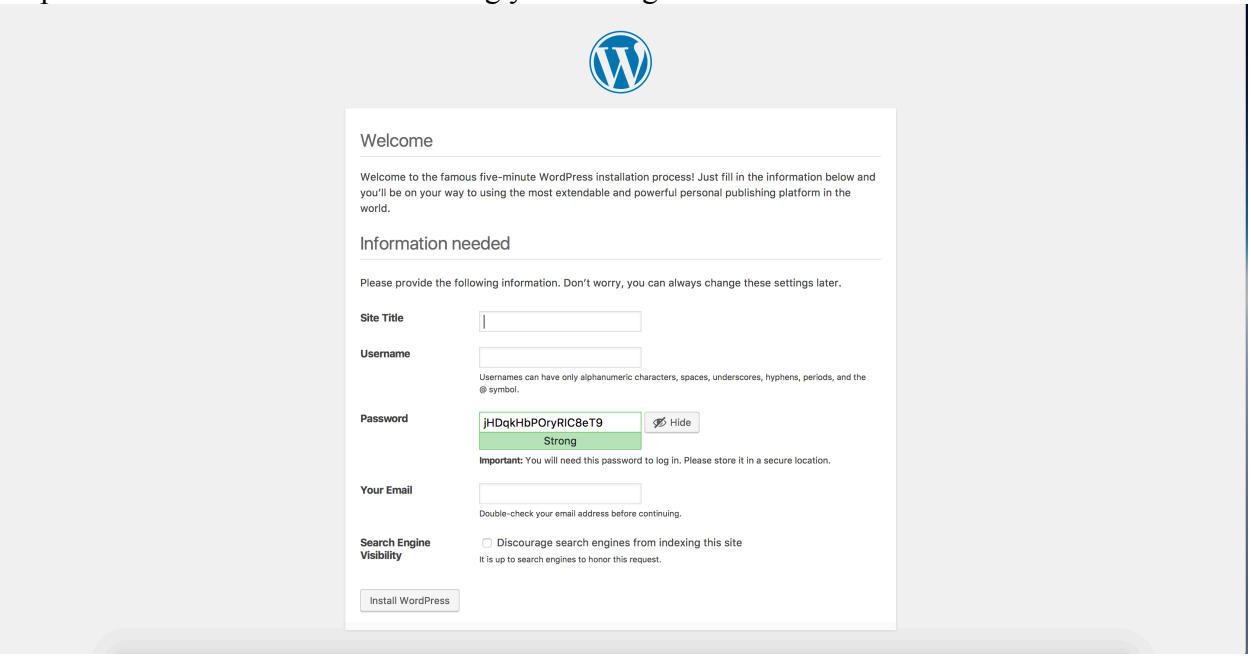
```
bx cs workers <your_cluster_name>
OK
ID          Public IP         Private IP     Machine Type   State          Status
Kube-w1     169.47.220.142    10.10.10.57    free           normal         Ready
```

Get the port details

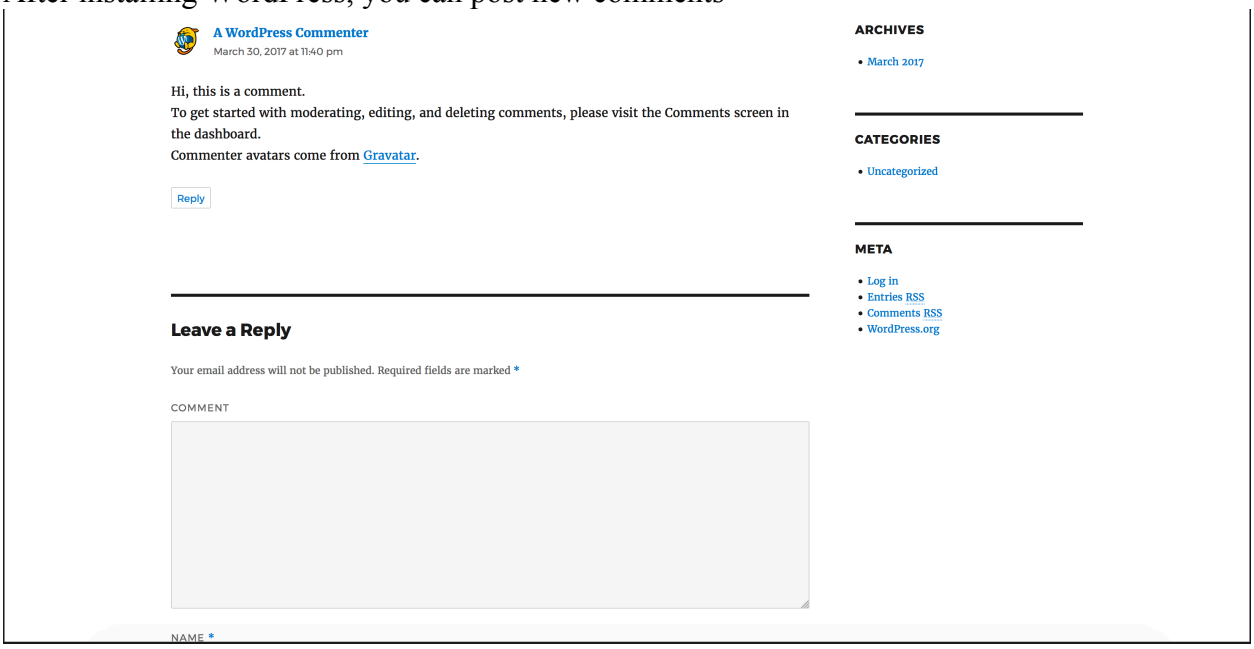
```
kubectl get svc wordpress

NAME         CLUSTER-IP     EXTERNAL-IP         PORT(S)              AGE
wordpress    10.10.10.57          <nodes>       80:30180/TCP         2m
```

Congratulation. Now you can use the link http://[Public IP]:[port number] to access your WordPress site.

Step 9: Now that WordPress is running you can register as a new user and install WordPress.



After installing WordPress, you can post new comments




## Scale up the application

Now, you can run the following commands to scale up for WordPress frontend.

```
kubectl scale deployments/wordpress --replicas=2
```

```
deployment "wordpress" scaled
```

```
kubectl get deployments
```

```
NAME              DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
wordpress         2         2         2            2           23h
wordpress-mysql   1         1         1            1           23h
```

## Resource Links

- Code Pattern URL: https://developer.ibm.com/code/patterns/scalable-wordpress-on-kubernetes/
- Github URL: https://github.com/IBM/scalable-wordpress-deployment-on-kubernetes?cm_sp=IBMCode-_-scalable-wordpress-on-kubernetes-_-Get-the-Code
- Docs: https://console.bluemix.net/docs/containers/container_index.html#container_index