Manual Method the deploy Kubernetes using gcloud

Select the project and use same project id for this deployment

Login to gcloud console and click on kubernates Engine option and select cluster

Then click on create cluster and based on the requirement you can select the option

Standard and auto

By default, one cluster will have 3 nodes if you want you add additional nodes based on the requirement.

You can select option like scaling and auto scaling, resources and OS type by default E2

Then it will click on create it will take few minutes to complete and come online

Once it's up then you can connect to gcloud shell prompt and copy and paste the content line command line access

Once shell prompt is up and then u can paste the content and activate the shell

Commands to check

Kubectl get nodes--- by default 3 nodes

Kubectl get pods--- no pods will be available

Now deploy the WordPress on k8 with shell line

- Create a GKE cluster.
- Create a PV and a PVC backed by Persistent Disk.
- Create a Cloud SQL for MySQL instance.
- Deploy WordPress.
- Set up your WordPress blog.

In Cloud Shell, enable the GKE and Cloud SQL Admin APIs

```
gcloud services enable container.googleapis.com sqladmin.googleapis.com
```

Cloud Shell, set the default zone for the `gcloud` command-line tool:

```
gcloud config set compute/zone zone
```

Download the app manifest files

```
 git clone https://github.com/GoogleCloudPlatform/kubernetes-engine-samples
```

Change to the directory with the `wordpress-persistent-disks` file:

```
cd kubernetes-engine-samples/wordpress-persistent-disks
```

Set the `WORKING_DIR` environment variable:

```
WORKING_DIR=$(pwd)
```

In Cloud Shell, deploy the manifest file:

```
kubectl apply -f $WORKING_DIR/wordpress-volumeclaim.yaml
```

```
kubectl get persistentvolumeclaim
```

Creating a Cloud SQL for MySQL instance

```
INSTANCE_NAME=mysql-wordpress-instance
gcloud sql instances create $INSTANCE_NAME
```

```
export INSTANCE_CONNECTION_NAME=$(gcloud sql instances describe
$INSTANCE_NAME \
    --format='value(connectionName)')
```

```
gcloud sql databases create wordpress --instance $INSTANCE_NAME

CLOUD_SQL_PASSWORD=$(openssl rand -base64 18)
gcloud sql users create wordpress --host=% --instance $INSTANCE_NAME \
    --password $CLOUD_SQL_PASSWORD
```

Configure a service account and create secrets

```
SA_NAME=cloudsql-proxy
gcloud iam service-accounts create $SA_NAME --display-name $SA_NAME

SA_EMAIL=$(gcloud iam service-accounts list \
    --filter=displayName:$SA_NAME \
    --format='value(email)')

gcloud projects add-iam-policy-binding $PROJECT_ID \
    --role roles/cloudsql.client \
    --member serviceAccount:$SA_EMAIL

gcloud iam service-accounts keys create $WORKING_DIR/key.json \
    --iam-account $SA_EMAIL

kubectl create secret generic cloudsql-db-credentials \
    --from-literal username=wordpress \
    --from-literal password=$CLOUD_SQL_PASSWORD

kubectl create secret generic cloudsql-instance-credentials \
    --from-file $WORKING_DIR/key.json
```

Deploy WordPress

```
cat $WORKING_DIR/wordpress_cloudsql.yaml.template | envsubst > \
    $WORKING_DIR/wordpress_cloudsql.yaml

kubectl create -f $WORKING_DIR/wordpress_cloudsql.yaml
```

```
kubectl get pod -l app=wordpress --watch
```

Expose the WordPress service

```
kubectl create -f $WORKING_DIR/wordpress-service.yaml
```

```
kubectl get svc -l app=wordpress --watch
```

```
http://external-ip-address
```