Set up and Configure a Cloud Environment in Google Cloud Challenge Lab

Author: Vedant Kakde | **GitHub Profile:** github.com/vedant-kakde | **LinkedIn Profile:** linkedin.com/in/vedant-kakde/

Task 1: Create development VPC manually

Make sure you create all resources in the `us-east1` region and `us-east1-b` zone.

1. In the Google Cloud Console, navigate to **VPC network** > **VPC networks**
2. Click on **Create VPC network**.
3. Enter `griffin-dev-vpc` to the **Name** field.
4. Select **Custom** for the Subnet creation mode.
5. Add `griffin-dev-wp` subnet with the following parameters:

Field	**Value**
Name:	`griffin-dev-wp`
Region:	`us-east1`
IP address range:	`192.168.16.0/20`

6. Click **+ Add subnet** and add `griffin-dev-mgmt` subnet with the following parameters

Field	**Value**
Name:	`griffin-dev-mgmt`
Region:	`us-east1`
IP address range:	`192.168.32.0/20`

7. Click **Create**.

Task 2: Create production VPC using Deployment Manager

1. Copy the Deployment Manager configuration files to Cloud Shell using the following command:

```
gsutil cp -r gs://cloud-training/gsp321/dm ~/
```

2. Edit `prod-network.yaml` configuration file

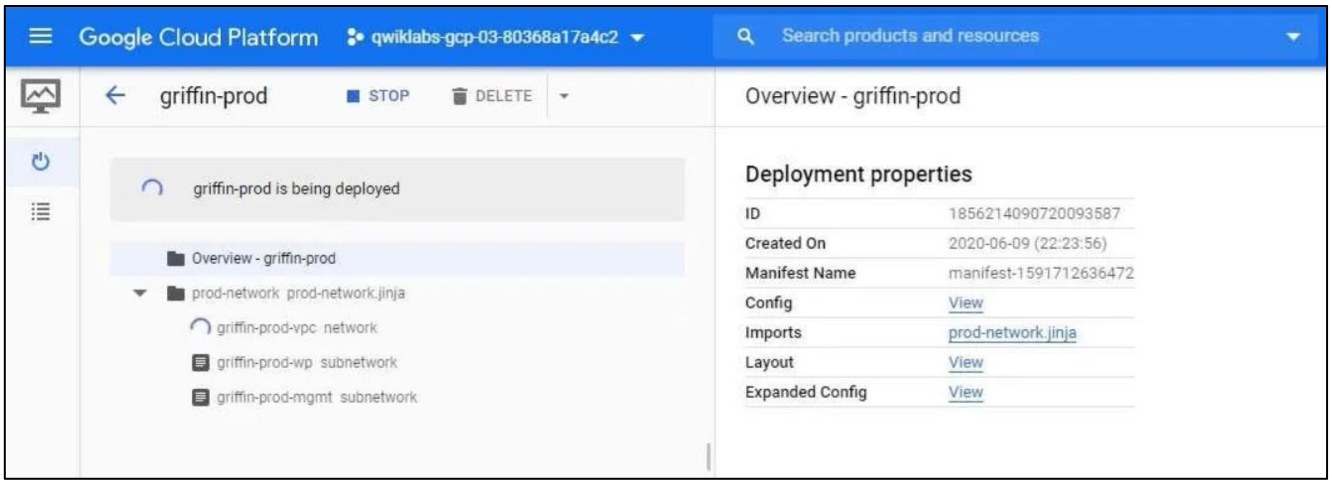
```
cd dm

edit prod-network.yaml
```

3. Replace `SET_REGION` to `us-east1` in the editor, and then save the change.
4. Go back to the Cloud Shell, use the following command to create the production VPC network with the configuration files:

```
gcloud deployment-manager deployments create griffin-prod --config prod-network.yaml
```

5. Go back to the Cloud Console, navigate to **Deployment Manager** to confirm the deployment.



griffin-prod is being deployed

- Overview - griffin-prod
- prod-network prod-network.jinja
  - griffin-prod-vpc network
  - griffin-prod-wp subnetwork
  - griffin-prod-mgmt subnetwork

Overview - griffin-prod

Deployment properties

ID	1856214090720093587
Created On	2020-06-09 (22:23:56)
Manifest Name	manifest-1591712636472
Config	View
Imports	prod-network.jinja
Layout	View
Expanded Config	View

Task 3: Create bastion host

1. In the Cloud Console, navigate to **Compute Engine** > **VM instances**.
2. Click **Create**.
3. Use the following parameters to create the bastion host:

Field	Value
Name:	`griffin-dev-db`
Region:	`us-east1`

4. Expand the **Management, security, disks, networking, sole tenancy** section.
5. In the **Networking** tab, add `bastion` to the Network tags.
6. Click **Add network interface**, make sure that you set up two Network interfaces,
   - `griffin-dev-mgmt`
   - `griffin-prod-mgmt`
7. Click **Create**.
8. Navigate to **VPC network** > **Firewall**.
9. Click **CREATE FIREWALL RULE**.
10. Configure the rule with the following parameters:

Field	Value
Name:	`allow-bastion-dev-ssh`
Network:	`griffin-dev-vpc`
Targets:	`bastion`
Source IP ranges:	`192.168.32.0/20`
Protocols and ports:	tcp: `22`

11. Click **CREATE**.
12. Click **CREATE FIREWALL RULE** again.
13. Configure another rule with the following parameters:

Field	Value
Name:	`allow-bastion-prod-ssh`
Network:	`griffin-prod-vpc`
Targets:	`bastion`
Source IP ranges:	`192.168.48.0/20`
Protocols and ports:	tcp: `22`

14. Click **CREATE**.

Task 4: Create and configure Cloud SQL Instance

1. In the Cloud Console, navigate to **SQL**.
2. Click **CREATE INSTANCE**.
3. Click **Choose MySQL**.
4. Use the following parameters to create the instance:

Field	Value
Name:	`griffin-dev-db`
Region:	`us-east1`
Zone:	`us-east1-b`
Root password:	e.g. `12345678`

5. **Note**: In real practice, you must set a strong password.
6. Click **Create**.
7. Click the `griffin-dev-db` in the SQL pane after it has been created.
8. Under **Connect to this instance**, click on **Connect using Cloud Shell**.
9. Go back to the Cloud Shell, run:

```
gcloud sql connect griffin-dev-db --user=root --quiet
```

10. Enter the **Root password** generated in Step 4.
11. In the SQL console, run the following query to create the wordpress database:

```
CREATE DATABASE wordpress;

GRANT ALL PRIVILEGES ON wordpress.* TO "wp_user"@"%" IDENTIFIED BY "stormwind_rules";

FLUSH PRIVILEGES;
```

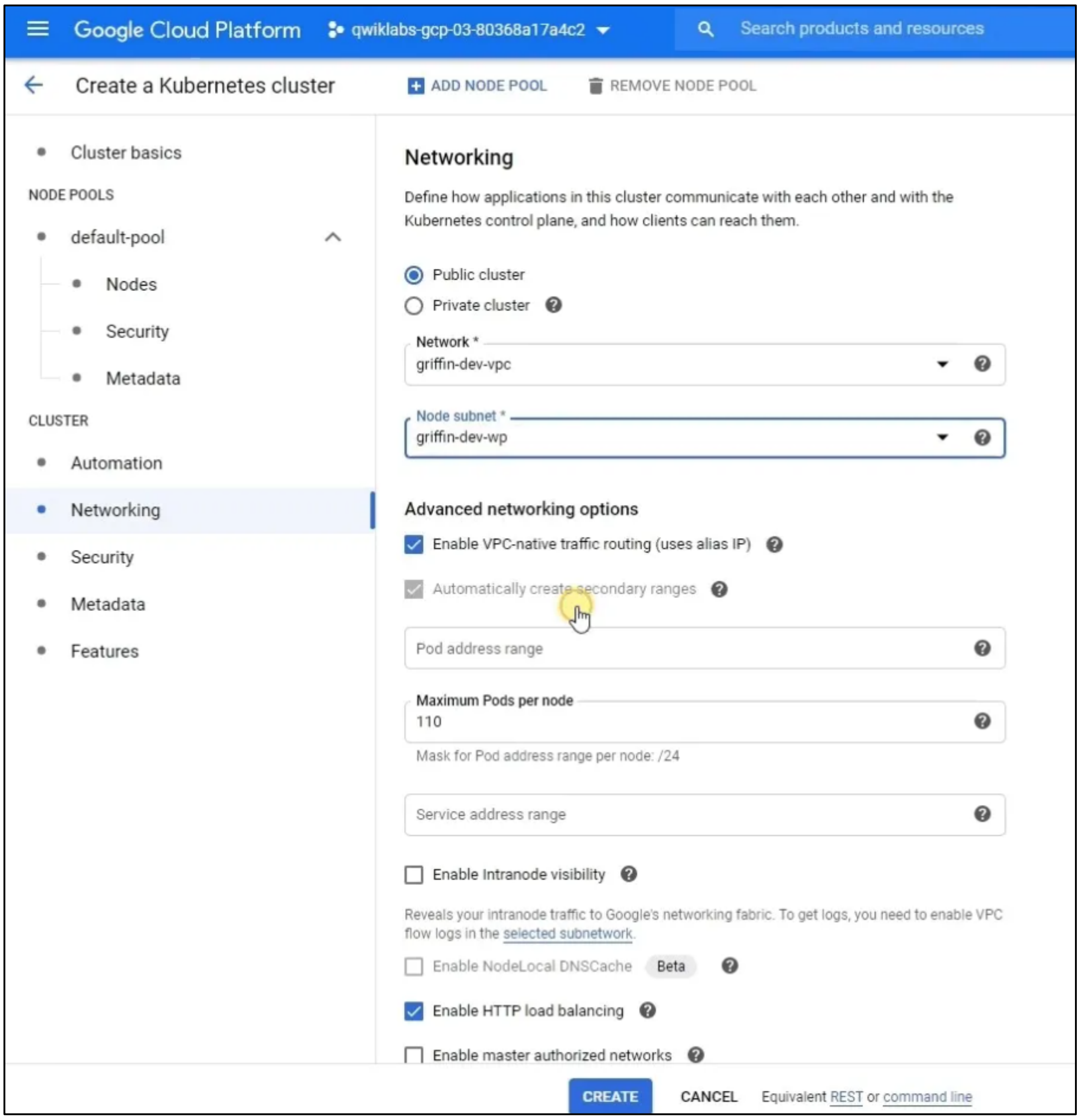
12. Enter `exit` to quit the SQL shell.

Task 5: Create Kubernetes cluster

Create a 2 node cluster (n1-standard-4) called `griffin-dev`, in the `griffin-dev-wp` subnet, and in the zone `us-east1-b`.

1. In the Cloud Console, navigate to **Kubernetes Engine** > **Clusters**.

2. Click **Create cluster**.
3. In the Cluster basics tab, configure:
   - Name: `griffin-dev`
   - Zone: `us-east1-b`
4. In the left pane, click **default-pool** under **NODE POOLS** and set
   - Number of nodes: `2`
5. Click **Nodes** Under **default-pool**, and set
   - Machine type: `n1-standard-4`
6. Go to the **Network** tab, set
   - Network: `griffin-dev-vpc`
   - Node subnet: `griffin-dev-wp`



Networking

Define how applications in this cluster communicate with each other and with the Kubernetes control plane, and how clients can reach them.

[✓ Public cluster] Private cluster

Network *: griffin-dev-vpc

Node subnet *: griffin-dev-wp

Advanced networking options

[✓] Enable VPC-native traffic routing (uses alias IP)

[✓] Automatically create secondary ranges

Pod address range

Maximum Pods per node: 110

Mask for Pod address range per node: /24

Service address range

[ ] Enable Intranode visibility

Reveals your intranode traffic to Google's networking fabric. To get logs, you need to enable VPC flow logs in the selected subnetwork.

[ ] Enable NodeLocal DNSCache Beta

[✓] Enable HTTP load balancing

[ ] Enable master authorized networks

7. Click **CREATE**.

Task 6: Prepare the Kubernetes cluster

1. In the Cloud Shell, use the following command to copy the files for the Kubernetes:

```
gsutil cp -r gs://cloud-training/gsp321/wp-k8s ~/
```

2. Open `wp-k8s/wp-env.yaml` with the Cloud Shell Editor.

```
cd ~/wp-k8s

edit wp-env.yaml
```

3. Replace `username_goes_here` and `password_goes_here` to `wp_user` and `stormwind_rules`, respectively.
4. Save the file change.
5. After the Kubernetes cluster has been created, click on the **Connect** button.
6. Run the following command to connect the cluster:

```
gcloud container clusters get-credentials griffin-dev --zone=us-east1
```

7. Deploy the configuration to the cluster using:

```
kubectl apply -f wp-env.yaml
```

8. Use the command below to create the key, and then add the key to the Kubernetes environment:

```
gcloud iam service-accounts keys create key.json \

    --iam-account=cloud-sql-proxy@$GOOGLE_CLOUD_PROJECT.iam.gserviceaccount.com

kubectl create secret generic cloudsql-instance-credentials \

    --from-file key.json
```

Task 7: Create a WordPress deployment

1. Open `wp-k8s/wp-deployment.yaml` with the Cloud Shell Editor

```
cd ~/wp-k8s

edit wp-deployment.yaml
```

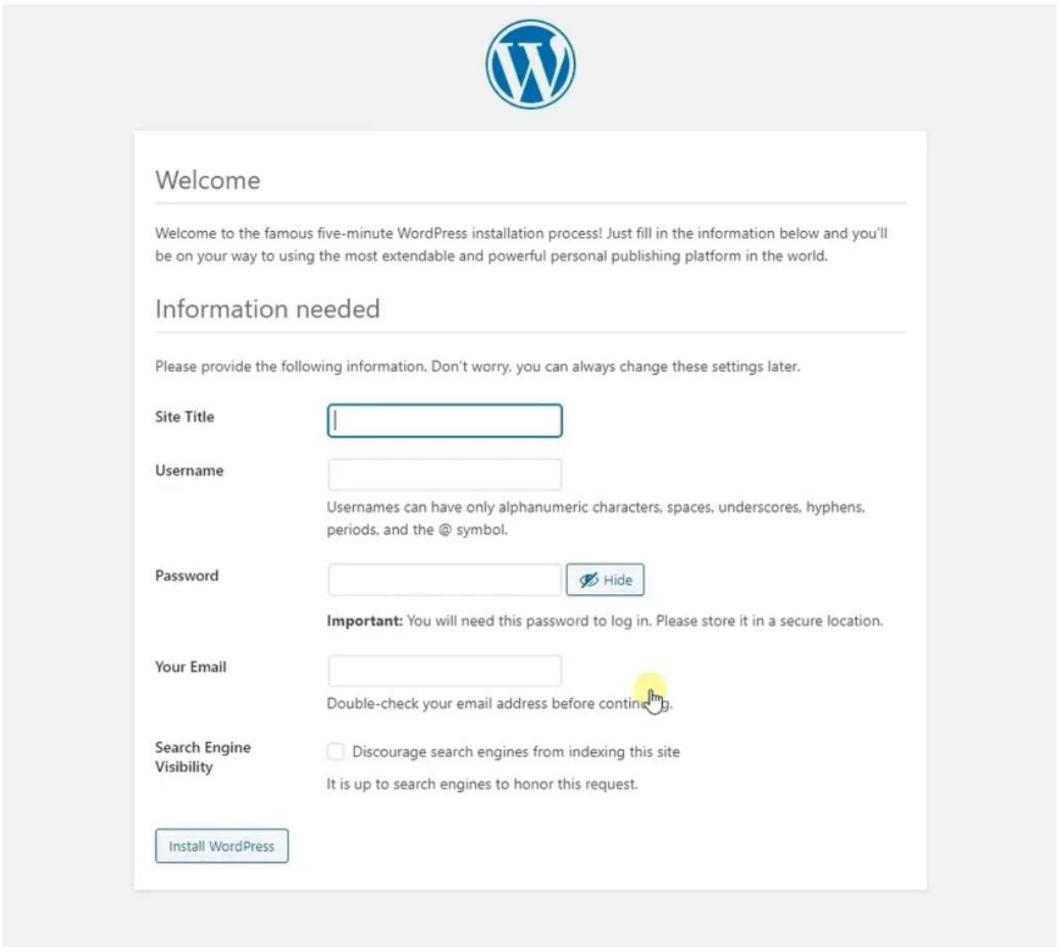
2. Replace `YOUR_SQL_INSTANCE` with `griffin-dev-db`'s Instance connection name.

3. Save the file change.
4. Go back to the Cloud Shell, run the following commands:

```
kubectl create -f wp-deployment.yaml

kubectl create -f wp-service.yaml
```

5. Copy the External endpoints of the deployed **wordpress** service and open it in your browser.



Welcome

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.

Information needed

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

Site Title

Username

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

Password

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

Your Email

Double-check your email address before continuing.

Search Engine Visibility

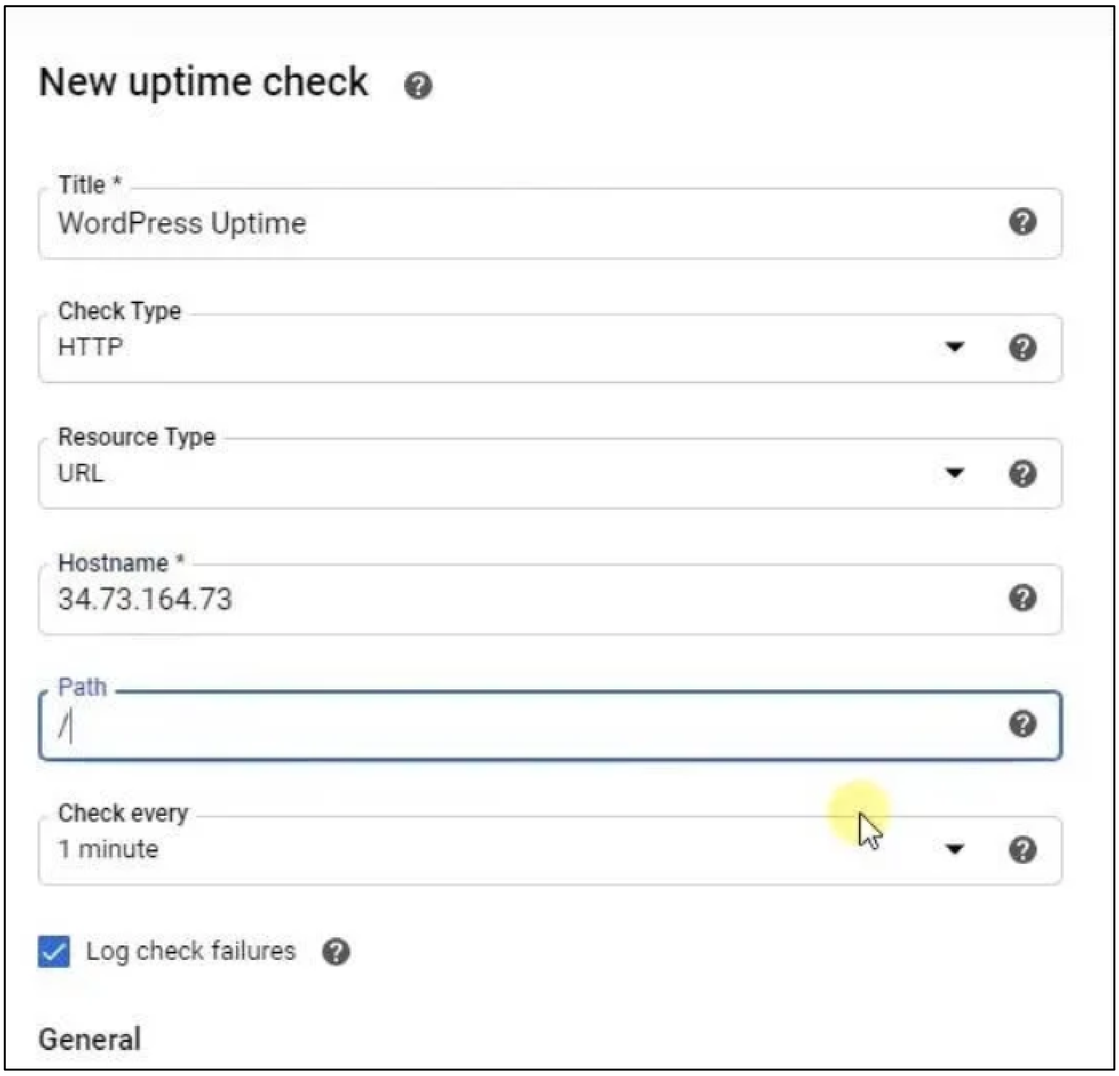
Discourage search engines from indexing this site

It is up to search engines to honor this request.

Task 8: Enable monitoring

1. Go back to the Cloud Console, and navigate to **Monitoring**.
2. In the Monitoring console, click **Uptime checks** in the left pane.
3. Click **CREATE UPTIME CHECK**.
4. Configure using the following parameters:

Field	Value
Title	`WordPress Uptime`
Check Type	`HTTP`
Resource Type	`URL`
Hostname	`YOUR-WORDPRESS_ENDPOINT`
Path	`/`


New uptime check

Title *: WordPress Uptime

Check Type: HTTP

Resource Type: URL

Hostname *: 34.73.164.73

Path: /

Check every: 1 minute

☑ Log check failures

General

5. Click **TEST**.
6. Click **SAVE** if there is no error.

Task 9: Provide access for an additional engineer

1. In the Cloud Console, navigate to **IAM & Admin** > **IAM**.
2. Click **+ADD**.
3. In the Add members to … pane, copy and paste the **second user account** for the lab to the **New members** field.
4. In the Role dropdown, select **Project** > **Editor**.
5. Click **SAVE**.

Congratulations! You completed this challenge lab.