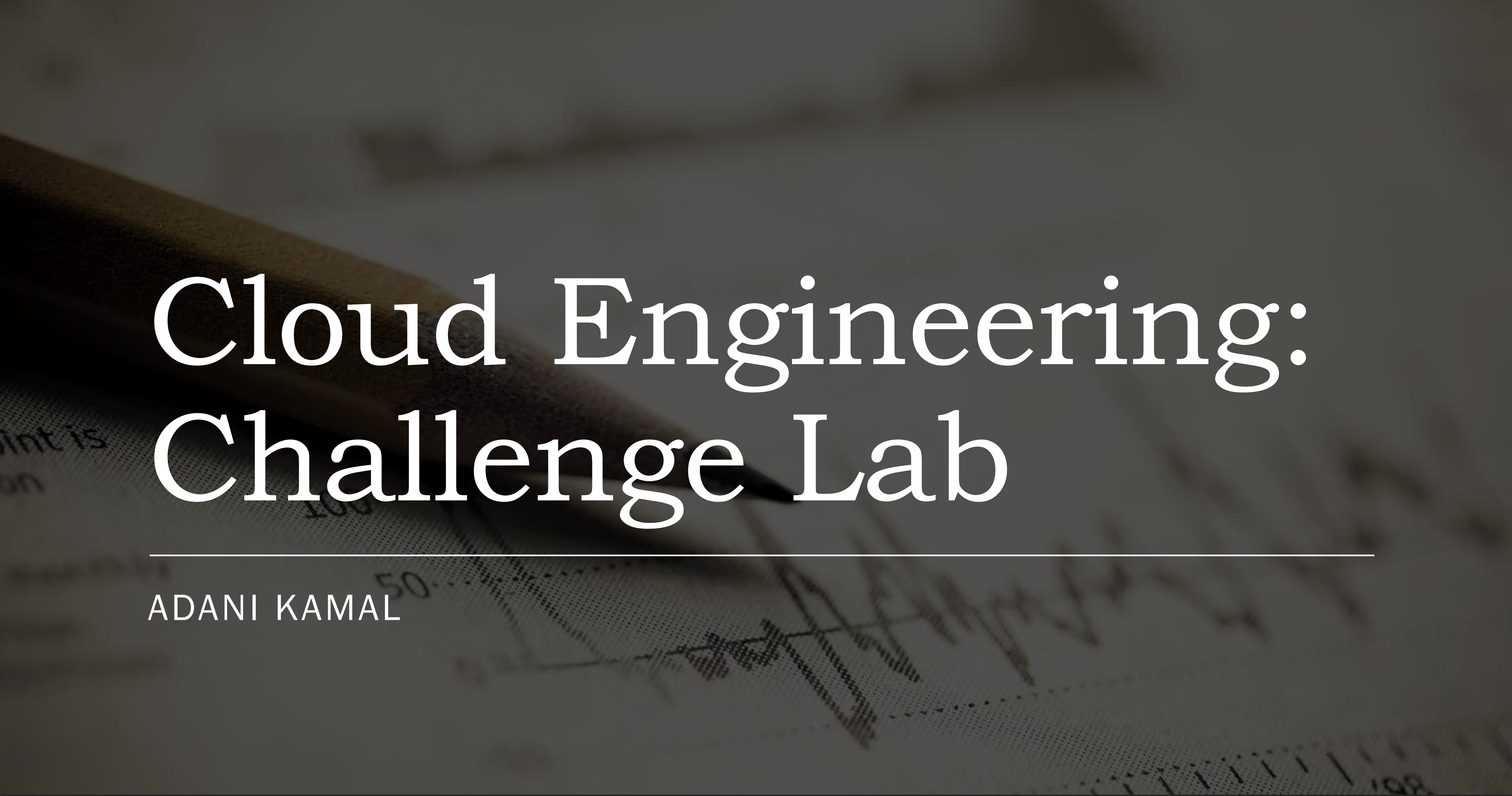


Cloud Engineering: Challenge Lab

ADANI KAMAL

Challenge scenario

As a cloud engineer in Jooli Inc. and recently trained with Google Cloud and Kubernetes you have been asked to help a new team (Griffin) set up their environment. The team has asked for your help and has done some work, but needs you to complete the work.

You are expected to have the skills and knowledge for these tasks so don't expect step-by-step guides.

You need to complete the following tasks:

- Create a development VPC with three subnets manually
- Create a production VPC with three subnets using a provided Deployment Manager configuration
- Create a bastion that is connected to both VPCs
- Create a development Cloud SQL Instance and connect and prepare the WordPress environment
- Create a Kubernetes cluster in the development VPC for WordPress
- Prepare the Kubernetes cluster for the WordPress environment
- Create a WordPress deployment using the supplied configuration
- Enable monitoring of the cluster via stackdriver
- Provide access for an additional engineer

Some Jooli Inc. standards you should follow:

- Create all resources in the `us-east1` region and `us-east1-b` zone, unless otherwise directed.
- Use the project VPCs.
- Naming is normally *team-resource*, e.g. an instance could be named **kraken-webserver1**.
- Allocate cost effective resource sizes. Projects are monitored and excessive resource use will result in the containing project's termination (and possibly yours), so beware. This is the guidance the monitoring team is willing to share: unless directed, use `n1-standard-1`.

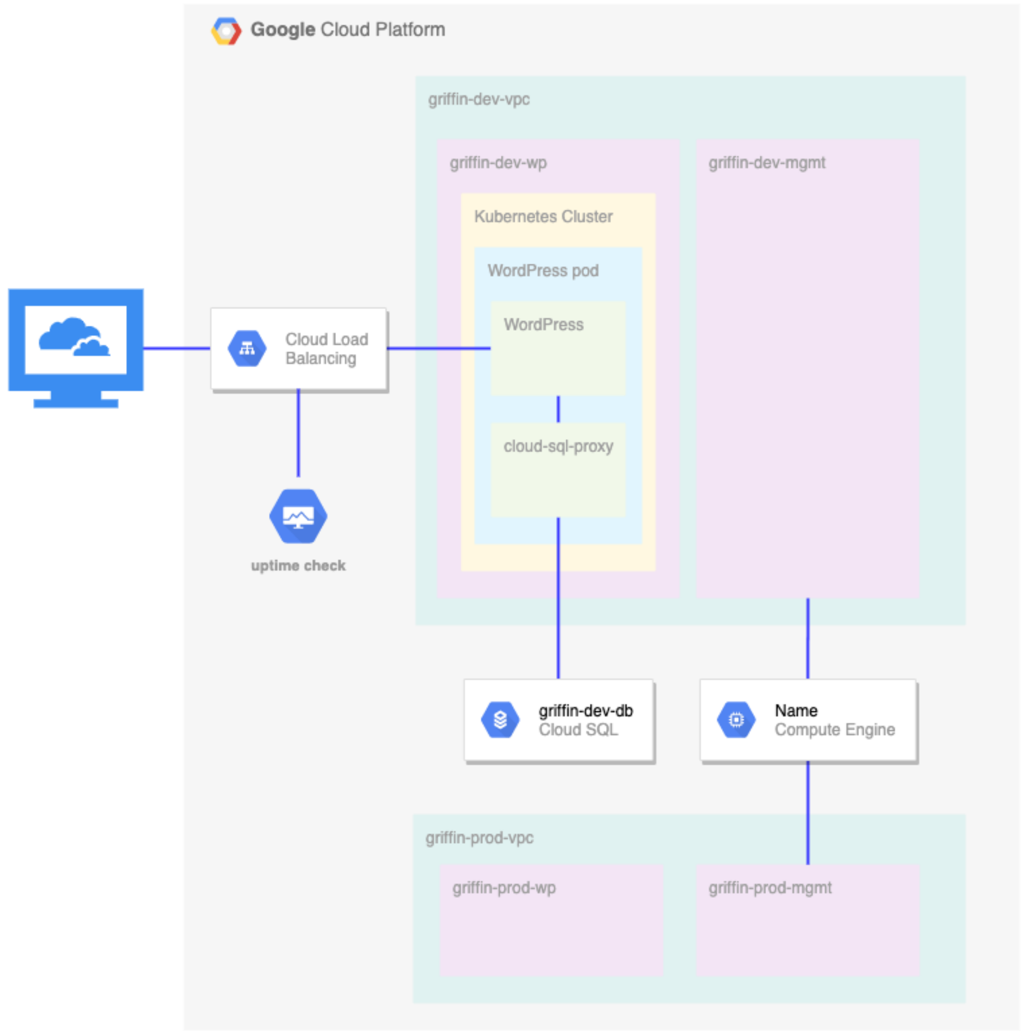
Your challenge

You need to help the team with some of their initial work on a new project. They plan to use WordPress and need you to set up a development environment. Some of the work was already done for you, but other parts require your expert skills.

As soon as you sit down at your desk and open your new laptop you receive the following request to complete these tasks. Good luck!

Environment

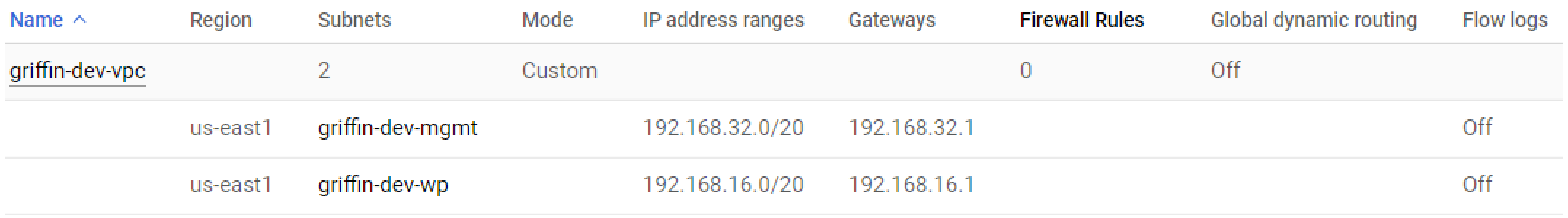
Team Griffin Infrastructure



Task 1: Create development VPC manually

Create a VPC called `griffin-dev-vpc` with the following subnets only:

- `griffin-dev-wp`
    - IP address block: `192.168.16.0/20`
- `griffin-dev-mgmt`
    - IP address block: `192.168.32.0/20`



Name ^	Region	Subnets	Mode	IP address ranges	Gateways	Firewall Rules	Global dynamic routing	Flow logs
griffin-dev-vpc		2	Custom			0	Off	
	us-east1	griffin-dev-mgmt		192.168.32.0/20	192.168.32.1			Off
	us-east1	griffin-dev-wp		192.168.16.0/20	192.168.16.1			Off

VPC Network > Create VPC Network

Name: griffin-dev-vpc

Subnet: griffin-dev-wp

Region: us-east1

Ip address: 192.168.16.0/20

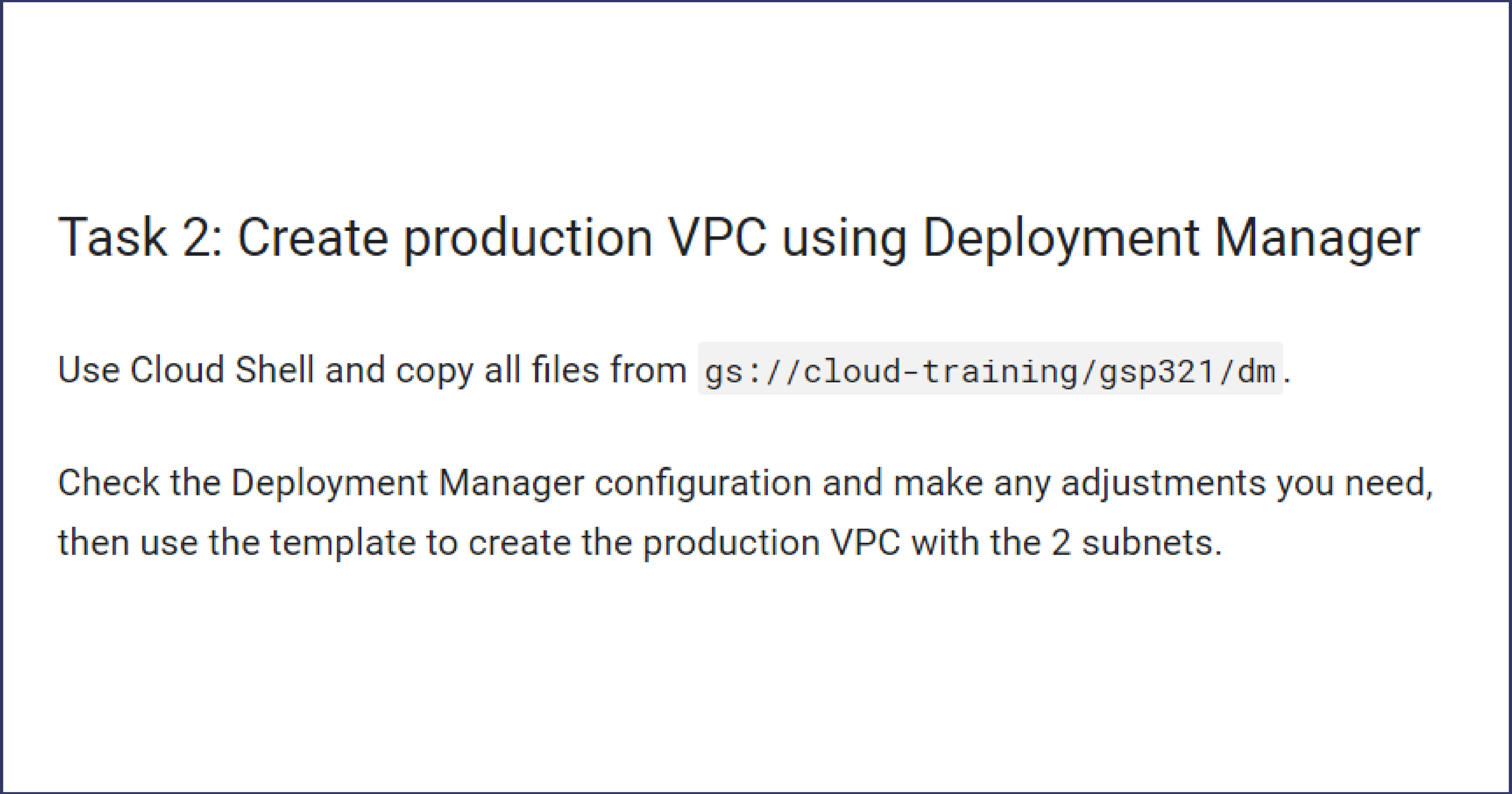
Add subnet

Subnet: griffin-dev-mgmt

Region: us-east1

Ip address: 192.168.32.0/20

Create

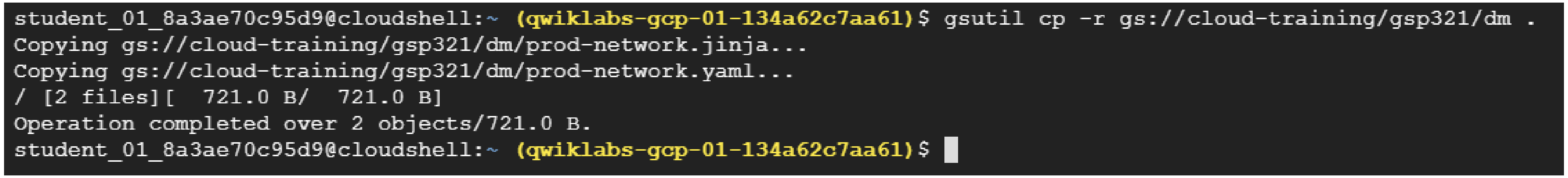


Task 2: Create production VPC using Deployment Manager

Use Cloud Shell and copy all files from `gs://cloud-training/gsp321/dm`.

Check the Deployment Manager configuration and make any adjustments you need, then use the template to create the production VPC with the 2 subnets.

gsutil cp –r gs://cloud-training/gsp321/dm .



```
student_01_8a3ae70c95d9@cloudshell:~ (qwiklabs-gcp-01-134a62c7aa61)$ gsutil cp -r gs://cloud-training/gsp321/dm .
Copying gs://cloud-training/gsp321/dm/prod-network.jinja...
Copying gs://cloud-training/gsp321/dm/prod-network.yaml...
/ [2 files][  721.0 B/  721.0 B]
Operation completed over 2 objects/721.0 B.
student_01_8a3ae70c95d9@cloudshell:~ (qwiklabs-gcp-01-134a62c7aa61)$
```

Cd dm

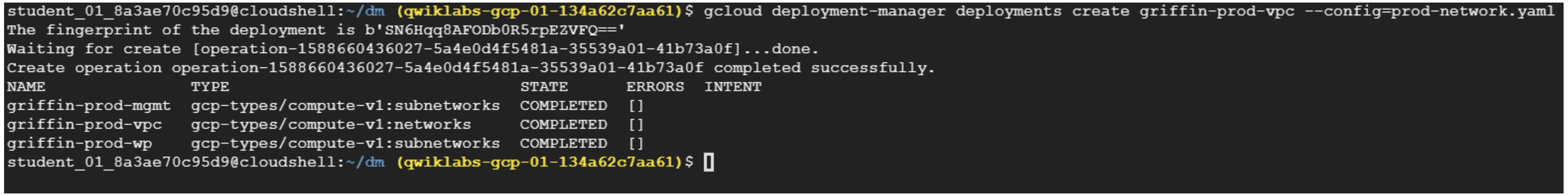
Ls

Open Editor

Dm> prod-network.yaml

Edit: “SET_REGION” to us-east1

gcloud deployment-manager deployments create griffin-prod-vpc --config=**prod-network**.yaml



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ gcloud deployment-manager deployments create griffin-prod-vpc --config=prod-network.yaml
The fingerprint of the deployment is b'SN6Hqq8AFODb0R5rpEZVFQ=='
Waiting for create [operation-1588660436027-5a4e0d4f5481a-35539a01-41b73a0f]...done.
Create operation operation-1588660436027-5a4e0d4f5481a-35539a01-41b73a0f completed successfully.
NAME                TYPE                                STATE      ERRORS  INTENT
griffin-prod-mgmt   gcp-types/compute-v1:subnetworks   COMPLETED  []
griffin-prod-vpc    gcp-types/compute-v1:networks      COMPLETED  []
griffin-prod-wp     gcp-types/compute-v1:subnetworks   COMPLETED  []
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

Firewall rules > Create

[Name]= griffin-dev-vpc-allow-http-ssh-rdp

[Network]= griffin-dev-vpc

[Target]= All instance in the network

[Source ip range]= 0.0.0.0/0

[tick- tcp]= 22,80,3389

[tick- other protocols]= icmp

CREATE

Firewall rules > Create

griffin-prod-vpc-allow-http-ssh-rdp

[Network]= griffin-prod-vpc

[Target]= All instance in the network

[Source ip range]= 0.0.0.0/0

[tick- tcp]= 22,80,3389

[tick- other protocols]= icmp

CREATE

Task 3: Create bastion host

Create a bastion host with two network interfaces, one connected to `griffin-dev-mgmt` and the other connected to `griffin-prod-mgmt`. Make sure you can SSH to the host.

Compute engine > VM Instance

Create

Name: bastion-host

Management, security disks, networking, sole tenancy

Networking

Network interfaces

Name: griffin-dev-vpc

Subnetwork: griffin-dev-mgmt

Done

Add Network Interfaces

Name: griffin-dev-vpc

Subnetwork: griffin-prod-mgmt

Done

Create

SSH

Task 4: Create and configure Cloud SQL Instance

Create a **MySQL Cloud SQL Instance** called `griffin-dev-db` in us-east1. Connect to the instance and run the following SQL commands to prepare the **WordPress** environment:

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

These SQL statements create the worpdress database and create a user with access to the wordpress dataase.

You will use the username and password in task 6.

SQL

Create Instance

MySQL

Name: griffin-dev-db

Pw: griffin

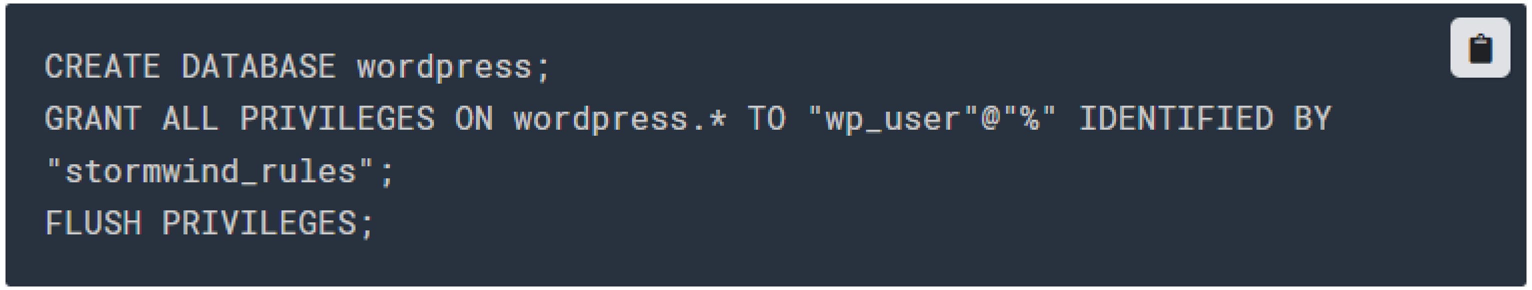
Region: us-east1

Zone: us-east1-b

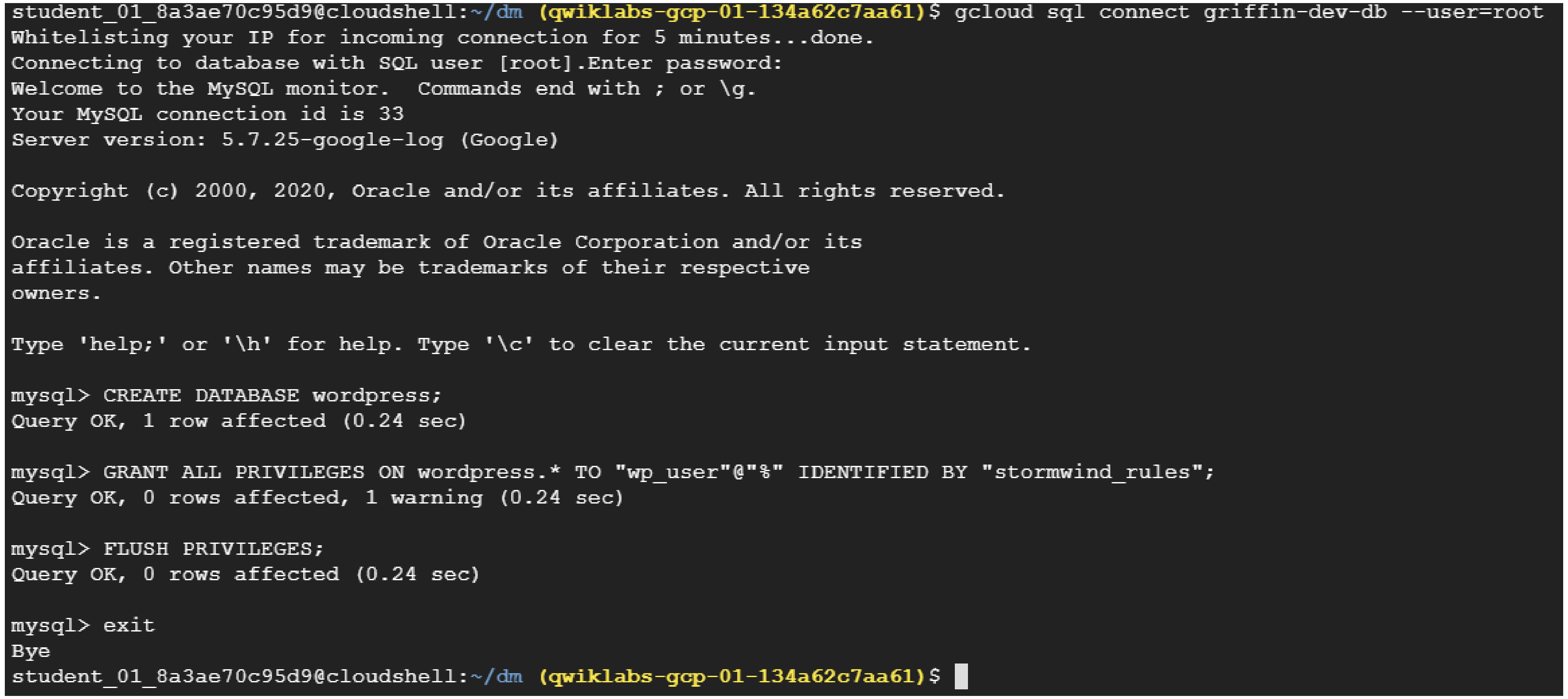
Create

Wait GREEN tick

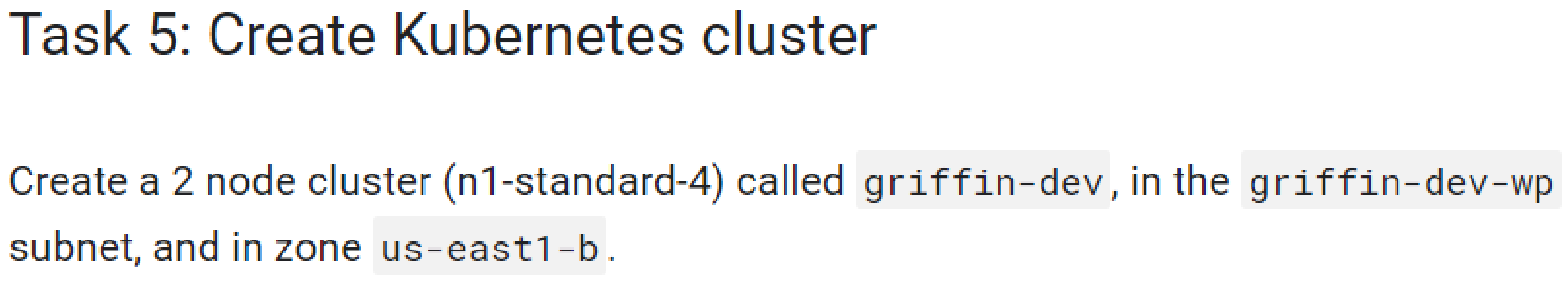
gcloud sql connect griffin-dev-db --user=root



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



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ gcloud sql connect griffin-dev-db --user=root
Whitelisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 33
Server version: 5.7.25-google-log (Google)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE wordpress;
Query OK, 1 row affected (0.24 sec)

mysql> GRANT ALL PRIVILEGES ON wordpress.* TO "wp_user"@"%" IDENTIFIED BY "stormwind_rules";
Query OK, 0 rows affected, 1 warning (0.24 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.24 sec)

mysql> exit
Bye
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```



Task 5: Create Kubernetes cluster

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

TASK 5

Kubernetes Engine

Create Cluster

Name: griffin-dev

Zone: us-east1-b

Click” default pools” (at the left side)

Number of nodes = 2

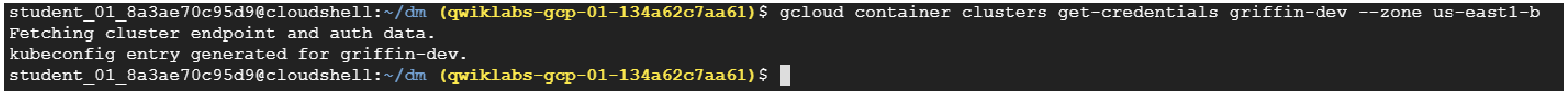
Click “nodes”

Machine type: n1-standard-4

Click “networking”

Chcek “Node Subnet” = griffin-dev-wp

gcloud container clusters get-credentials griffin-dev --zone us-east1-b



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ gcloud container clusters get-credentials griffin-dev --zone us-east1-b
Fetching cluster endpoint and auth data.
kubeconfig entry generated for griffin-dev.
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

Task 6: Prepare the Kubernetes cluster

Use Cloud Shell and copy all files from `gs://cloud-training/gsp321/wp-k8s`.

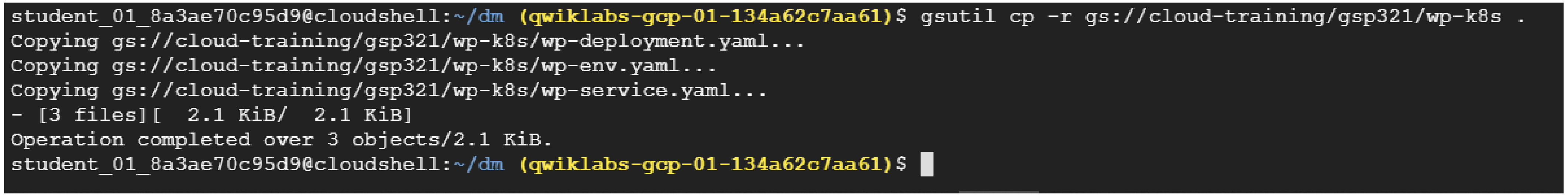
The **WordPress** server needs to access the MySQL database using the *username* and *password* you created in task 4. You do this by setting the values as secrets. **WordPress** also needs to store its working files outside the container, so you need to create a volume.

Add the following secrets and volume to the cluster using `wp-env.yaml`. Make sure you configure the *username* to `wp_user` and *password* to `stormwind_rules` before creating the configuration.

You also need to provide a key for a service account that was already set up. This service account provides access to the database for a sidecar container. 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
```

gsutil cp –r gs://cloud-training/gsp321/wp-k8s .



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ gsutil cp -r gs://cloud-training/gsp321/wp-k8s .
Copying gs://cloud-training/gsp321/wp-k8s/wp-deployment.yaml...
Copying gs://cloud-training/gsp321/wp-k8s/wp-env.yaml...
Copying gs://cloud-training/gsp321/wp-k8s/wp-service.yaml...
- [3 files][  2.1 KiB/  2.1 KiB]
Operation completed over 3 objects/2.1 KiB.
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

Cloud Shell Editor

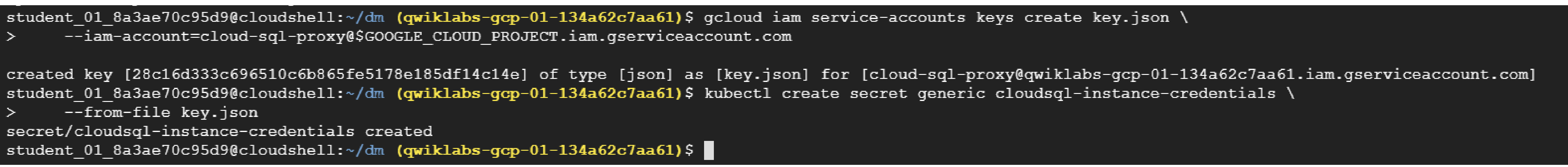
wp-k8s

wp-env.yaml

Username: wp_user

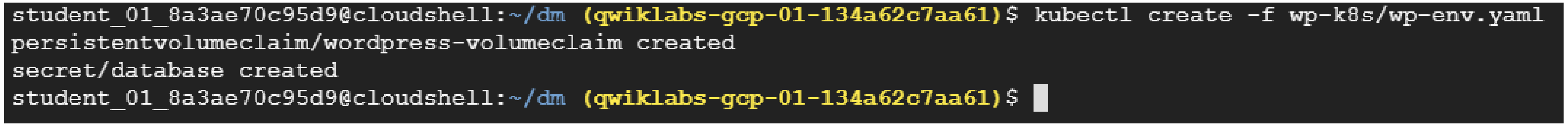
password: stormwind_rules

```
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
```



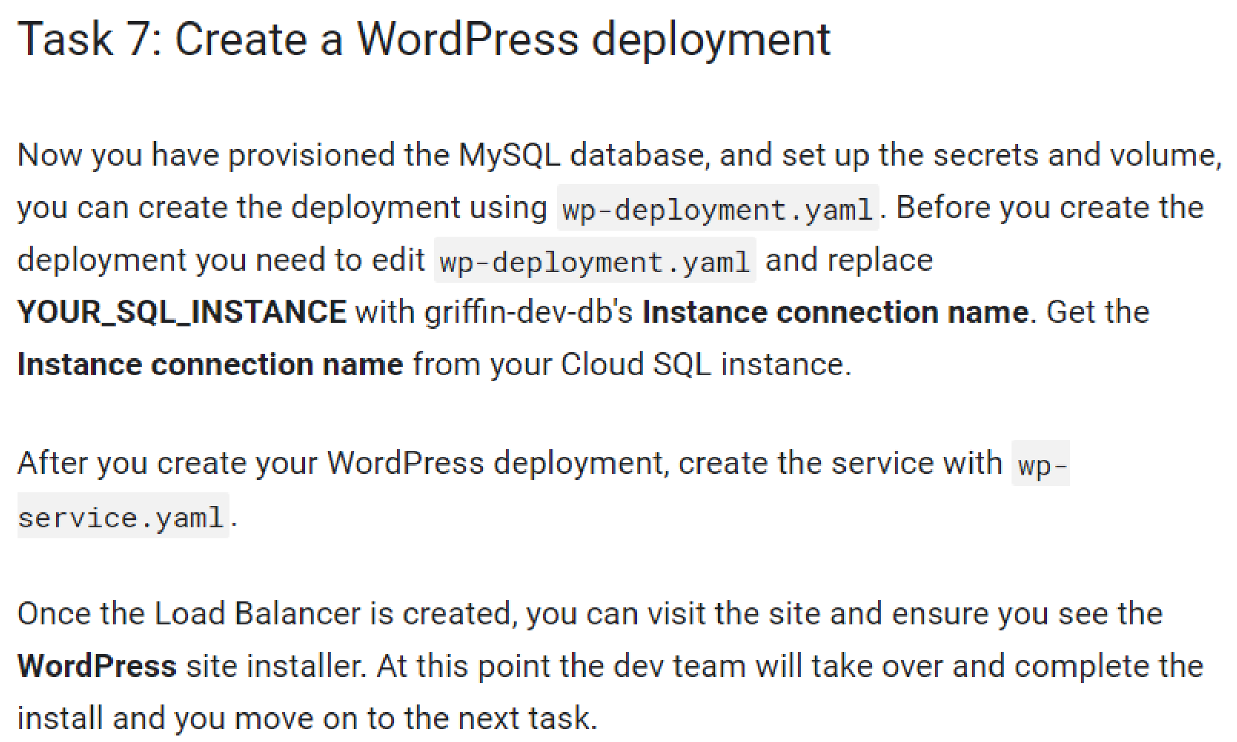
```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ gcloud iam service-accounts keys create key.json \
>     --iam-account=cloud-sql-proxy@$GOOGLE_CLOUD_PROJECT.iam.gserviceaccount.com

created key [28c16d333c696510c6b865fe5178e185df14c14e] of type [json] as [key.json] for [cloud-sql-proxy@qwiklabs-gcp-01-134a62c7aa61.iam.gserviceaccount.com]
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ kubectl create secret generic cloudsql-instance-credentials \
>     --from-file key.json
secret/cloudsql-instance-credentials created
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

kubectl create -f wp-k8s/wp-env.yaml



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ kubectl create -f wp-k8s/wp-env.yaml
persistentvolumeclaim/wordpress-volumeclaim created
secret/database created
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

TASK 7



Task 7: Create a WordPress deployment

Now you have provisioned the MySQL database, and set up the secrets and volume, you can create the deployment using `wp-deployment.yaml`. Before you create the deployment you need to edit `wp-deployment.yaml` and replace **YOUR_SQL_INSTANCE** with griffin-dev-db's **Instance connection name**. Get the **Instance connection name** from your Cloud SQL instance.

After you create your WordPress deployment, create the service with `wp-service.yaml`.

Once the Load Balancer is created, you can visit the site and ensure you see the **WordPress** site installer. At this point the dev team will take over and complete the install and you move on to the next task.

SQL

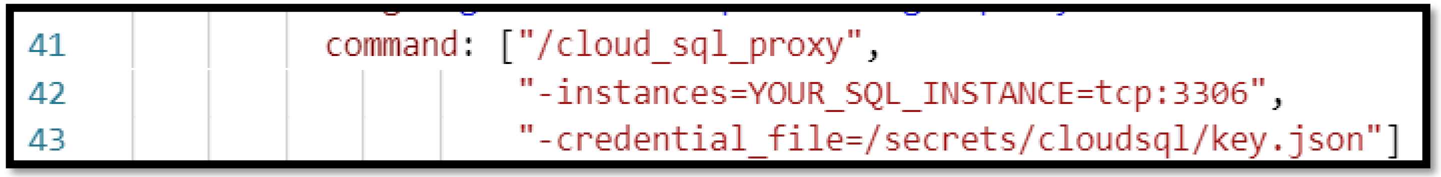
Copy “Instance connection name”

Editor

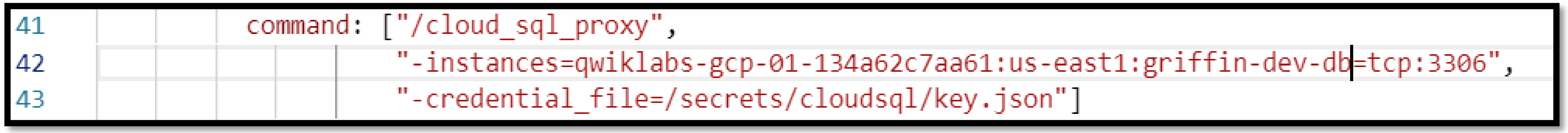
In wp-deployment.yaml

Paste at “YOUR_SQL_INSTANCE”

Save



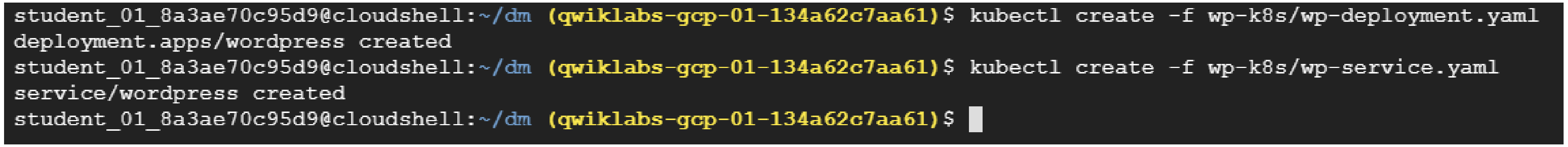
```
            command: ["/cloud_sql_proxy",
                      "-instances=YOUR_SQL_INSTANCE=tcp:3306",
                      "-credential_file=/secrets/cloudsql/key.json"]
```



```
            command: ["/cloud_sql_proxy",
                      "-instances=qwiklabs-gcp-01-134a62c7aa61:us-east1:griffin-dev-db=tcp:3306",
                      "-credential_file=/secrets/cloudsql/key.json"]
```

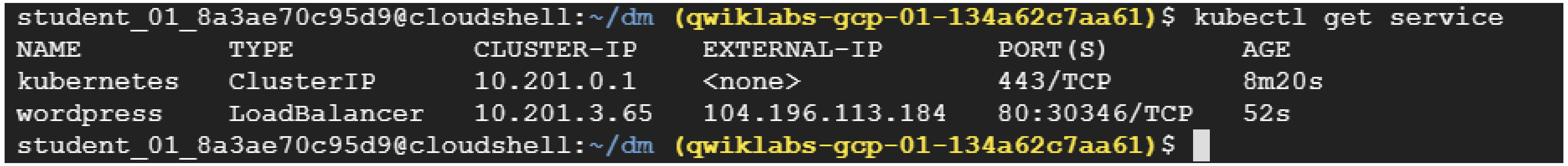
kubectl create -f wp-k8s/wp-deployment.yaml

kubectl create -f wp-k8s/wp-service.yaml



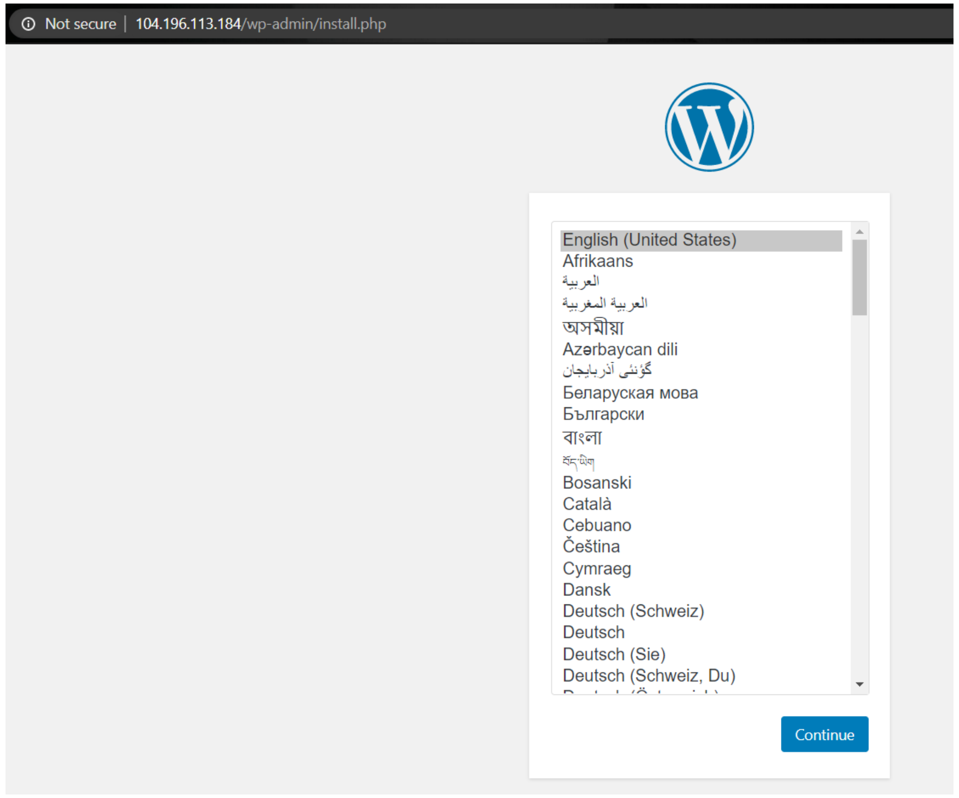
```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ kubectl create -f wp-k8s/wp-deployment.yaml
deployment.apps/wordpress created
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ kubectl create -f wp-k8s/wp-service.yaml
service/wordpress created
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

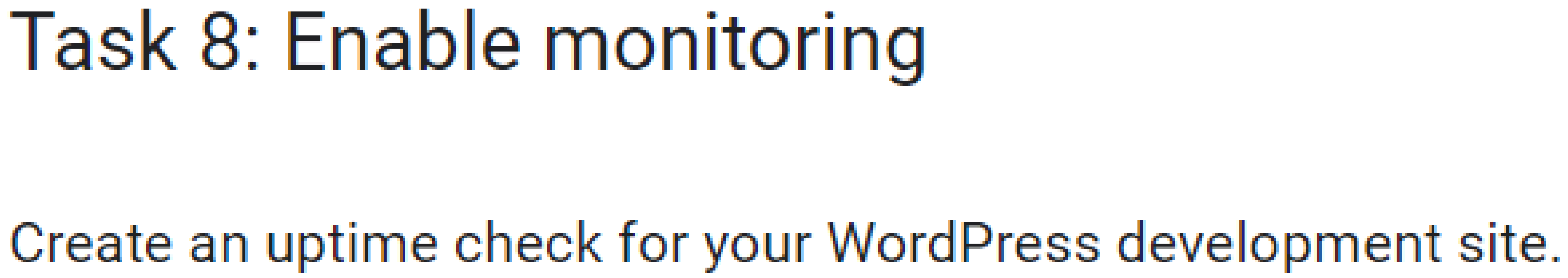
kubectl get service



```
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$ kubectl get service
NAME         TYPE           CLUSTER-IP    EXTERNAL-IP       PORT(S)        AGE
kubernetes   ClusterIP      10.201.0.1    <none>            443/TCP        8m20s
wordpress    LoadBalancer   10.201.3.65   104.196.113.184   80:30346/TCP   52s
student_01_8a3ae70c95d9@cloudshell:~/dm (qwiklabs-gcp-01-134a62c7aa61)$
```

Take “external ip”, open in browser





Task 8: Enable monitoring

Create an uptime check for your WordPress development site.

TASK 8

Monitoring

Create check

Title: wp-uptime-check

Host: EXTERNAL-IP

Click “TEST”

SAVE

Task 9: Provide access for an additional engineer

You have an additional engineer starting and you want to ensure they have access to the project, so please go ahead and grant them the editor role to the project.

The second user account for the lab represents the additional engineer.

IAM & ADMIN

On your student ID, Click Pencil icon.

Role “editor”, SAVE