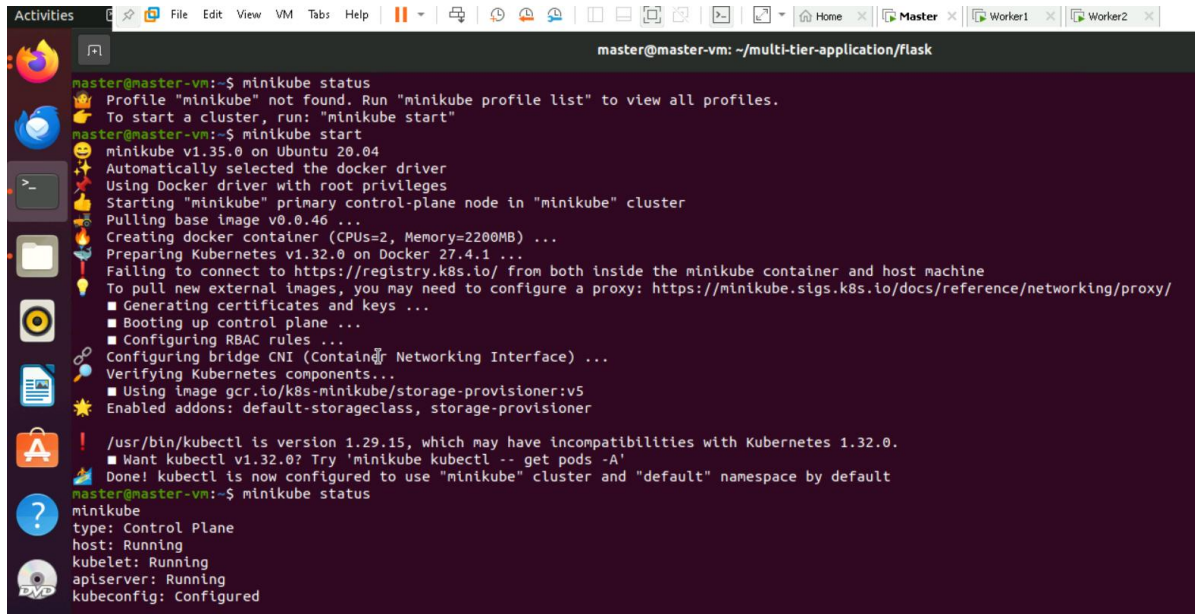


DEPLOY A MULTI-TIER WEB APPLICATION ON KUBERNETES

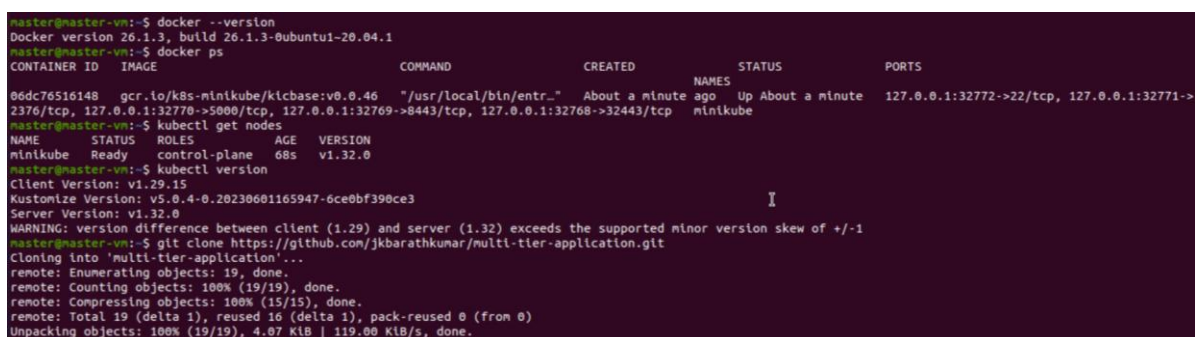
Step 1: First check the minikube is there or not, if not there install



A terminal window titled 'master@master-vm: ~/multi-tier-application/flask' showing the execution of minikube commands. The user first runs 'minikube status', which reports that the 'minikube' profile is not found and provides instructions to start it. Then, the user runs 'minikube start', which initiates the setup of a Kubernetes cluster on Ubuntu 20.04 using Docker. The process includes pulling the base image, creating a Docker container, preparing Kubernetes v1.32.0, and configuring various components like certificates, RBAC rules, and the bridge CNI. A warning message indicates that the installed kubectl version (1.29.15) is incompatible with the Kubernetes version (1.32.0) and suggests updating it. Finally, the user runs 'minikube status' again, which shows that the cluster is now running with the control plane, kubelet, apiserver, and kubeconfig all configured.

```
master@master-vm:~/multi-tier-application/flask$ minikube status
🚧 Profile "minikube" not found. Run "minikube profile list" to view all profiles.
To start a cluster, run: "minikube start"
master@master-vm:~/multi-tier-application/flask$ minikube start
minikube v1.35.0 on Ubuntu 20.04
👉 Automatically selected the docker driver
👉 Using Docker driver with root privileges
👉 Starting "minikube" primary control-plane node in "minikube" cluster
👉 Pulling base image v0.0.46 ...
👉 Creating docker container (CPUs=2, Memory=2200MB) ...
👉 Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
🚧 Failing to connect to https://registry.k8s.io/ from both inside the minikube container and host machine
👉 To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
👉 Generating certificates and keys ...
👉 Booting up control plane ...
👉 Configuring RBAC rules ...
👉 Configuring bridge CNI (Container Networking Interface) ...
👉 Verifying Kubernetes components...
👉 Using image gcr.io/k8s-minikube/storage-provisioner:v5
👉 Enabled addons: default-storageclass, storage-provisioner
🚧 /usr/bin/kubectl is version 1.29.15, which may have incompatibilities with Kubernetes 1.32.0.
👉 Want kubectl v1.32.0? Try 'minikube kubectl -- get pods -A'
👉 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
master@master-vm:~/multi-tier-application/flask$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

Step 2: Check docker install and node is connected or not



A terminal window showing the execution of docker and kubectl commands. The user first runs 'docker --version', which shows Docker version 26.1.3. Then, the user runs 'docker ps', which shows a list of running containers, including the minikube control plane. Next, the user runs 'kubectl get nodes', which shows the minikube node is ready. Finally, the user runs 'kubectl version', which shows the client version is v1.29.15 and the server version is v1.32.0. A warning message indicates that the version difference between the client and server exceeds the supported minor version skew of +/-1. The user then runs 'git clone https://github.com/jkbarathkumar/multi-tier-application.git', which successfully clones the repository.

```
master@master-vm:~/multi-tier-application/flask$ docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1-20.04.1
master@master-vm:~/multi-tier-application/flask$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS
06dc76516148   gcr.io/k8s-minikube/kicbase:v0.0.46 "/usr/local/bin/entr..." About a minute ago Up About a minute 127.0.0.1:32772->22/tcp, 127.0.0.1:32771->2376/tcp, 127.0.0.1:32770->5000/tcp, 127.0.0.1:32769->8443/tcp, 127.0.0.1:32768->32443/tcp minikube
master@master-vm:~/multi-tier-application/flask$ kubectl get nodes
NAME          STATUS    ROLES    AGE   VERSION
minikube      Ready    control-plane  68s   v1.32.0
master@master-vm:~/multi-tier-application/flask$ kubectl version
Client Version: v1.29.15
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
Server Version: v1.32.0
WARNING: version difference between client (1.29) and server (1.32) exceeds the supported minor version skew of +/-1
master@master-vm:~/multi-tier-application/flask$ git clone https://github.com/jkbarathkumar/multi-tier-application.git
Cloning into 'multi-tier-application'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 19 (delta 1), reused 16 (delta 1), pack-reused 0 (from 0)
Unpacking objects: 100% (19/19), 4.07 KiB | 119.00 KiB/s, done.
```

Step 3: Make directory and add all the files with code

```
master@master-vm:~$ cd multi-tier-application/
master@master-vm:~/multi-tier-application$ ls
flask  mysql  nginx  README.md
master@master-vm:~/multi-tier-application$ cd flask/
master@master-vm:~/multi-tier-application/flask$ ls
app.py  Dockerfile  flask-deployment.yaml  flask-service.yaml  requirements.txt
master@master-vm:~/multi-tier-application/flask$ nano Dockerfile
```

Step 4: Build the docker image and push to the docker hub

```
master@master-vm:~/multi-tier-application/flask$ docker build -t juhichoudhary/flaskapp
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

"docker build" requires exactly 1 argument.
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile
master@master-vm:~/multi-tier-application/flask$ docker push juhichoudhary/flaskapp
Using default tag: latest
The push refers to repository [docker.io/juhichoudhary/flaskapp]
An image does not exist locally with the tag: juhichoudhary/flaskapp
master@master-vm:~/multi-tier-application/flask$ docker build -t juhichoudhary/flaskapp .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  8.192kB
Step 1/6 : FROM python:3.8
3.8: Pulling from library/python
cdd62bf39133: Pull complete
a47c7f7f31e9: Pull complete
a173f2aee8e9: Pull complete
01272fe8adba: Extracting [=====> ] 210.6MB/211.3MB
01272fe8adba: Pull complete
cddc73e4e6c7: Pull complete
cc48f13b5f0f: Pull complete
5a98c896c047: Pull complete
Digest: sha256:d411270700143fa2683cc8264d9fa5d3279fd3b6afff62ae81ea2f9d070e390c
```

Step 5: login to the Docker and push the image

```
Successfully built 3439a3e04d5e
Successfully tagged juhichoudhary/flaskapp:latest
master@master-vm:~/multi-tier-application/flask$ docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/master/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
master@master-vm:~/multi-tier-application/flask$ docker push juhichoudhary/flaskapp
Using default tag: latest
The push refers to repository [docker.io/juhichoudhary/flaskapp]
b005f8114e16: Pushing [>] 1.083MB/73.82MB
3c41ae1085d1: Pushing [=====>] 2.56kB
5245d34565af: Pushing [=====>] 4.096kB
a4bcd1c9ec6a: Pushing 1.536kB
32ee710ca3c7: Preparing
1767e4d52b5a: Waiting
45b98afd69b3: Waiting
b005f8114e16: Pushed
3c41ae1085d1: Pushed
5245d34565af: Pushed
a4bcd1c9ec6a: Pushed
32ee710ca3c7: Mounted from library/python
1767e4d52b5a: Mounted from library/python
45b98afd69b3: Mounted from library/python
2bce433c3a29: Mounted from library/python
f91dc7a486d9: Mounted from library/python
3e14a6961052: Mounted from library/python
d50132f2fe78: Mounted from library/python
latest: digest: sha256:eb87883880e4c8522d4d1a94aad94cf906212b50be6aeb000781bffc6c03c78 size: 2628
master@master-vm:~/multi-tier-application/flask$ kubectl apply -f flask-deployment.yaml
```

Step 6: Apply the configuration

```
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/flask/flask-deployment.yaml
deployment.apps/flask-app configured
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/flask/flask-service.yaml
service/flask-service unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/mysql/mysql-deployment.yaml
deployment.apps/mysql unchanged
service/mysql unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/mysql/mysql-pv.yaml
persistentvolume/mysql-pv unchanged
persistentvolumeclaim/mysql-pvc unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/mysql/mysql-secret.yaml
secret/mysql-secret unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/nginx/nginx-configmap.yaml
configmap/nginx-config unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/nginx/nginx-deployment.yaml
deployment.apps/nginx unchanged
master@master-vm:~/multi-tier-application/nginx$ kubectl apply -f ~/multi-tier-application/nginx/nginx-service.yaml
service/nginx-service unchanged
```

Step 7: get all pods

```
master@master-vm:~/multi-tier-application/nginx$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE       NOMINATED NODE   READINESS GATES
flask-app-5b4dfc7c78-27mhf          1/1     Running   0           15h   10.244.0.19     minikube   <none>            <none>
flask-app-5b4dfc7c78-74fm4          1/1     Running   0           15h   10.244.0.17     minikube   <none>            <none>
flask-app-5b4dfc7c78-9scrk          1/1     Running   0           15h   10.244.0.16     minikube   <none>            <none>
flask-app-5b4dfc7c78-qdzkm          1/1     Running   0           15h   10.244.0.18     minikube   <none>            <none>
mysql-5f797c4f79-rzxrs              1/1     Running   0           15h   10.244.0.20     minikube   <none>            <none>
nginx-766c76446-6pf2g               1/1     Running   0           34s   10.244.0.22     minikube   <none>            <none>
master@master-vm:~/multi-tier-application/nginx$ minikube ip
192.168.49.2
master@master-vm:~/multi-tier-application/nginx$ kubectl get services nginx-service
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
nginx-service  NodePort    10.97.49.102 <none>        80:30007/TCP     15h
```

Step 8: Database created and outputs

```
master@master-vm:~/multi-tier-application/mysql$ kubectl exec -it mysql-5f797c4f79-rzxrs -- mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 5.7.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

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 mydb;
ERROR 1007 (HY000): Can't create database 'mydb'; database exists
mysql> USE mydb;
Database changed
mysql> SHOW TABLES;
Empty set (0.01 sec)

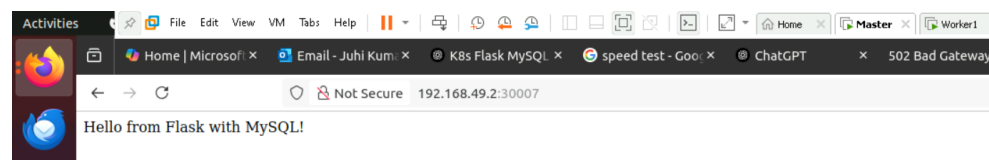
mysql> CREATE TABLE users (
    ->     id INT AUTO INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     email VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.16 sec)

mysql> INSERT INTO users (name, email) VALUES ('Alice', 'alice@example.com');
Query OK, 1 row affected (0.05 sec)

mysql> INSERT INTO users (name, email) VALUES ('Bob', 'bob@example.com');
Query OK, 1 row affected (0.03 sec)

mysql> SELECT * FROM users;
+-----+-----+-----+
| id | name | email |
+-----+-----+-----+
| 1 | Alice | alice@example.com |
| 2 | Bob | bob@example.com |
+-----+-----+-----+
```

Step 9: open browser and run ip:30007



Step10: Database output table created

