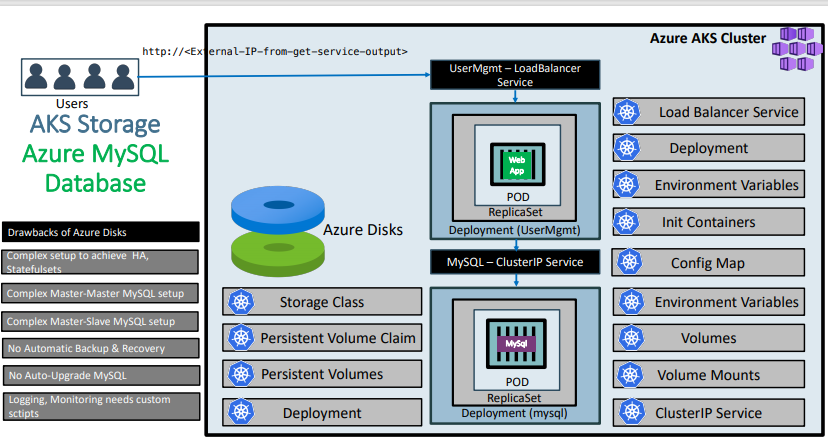
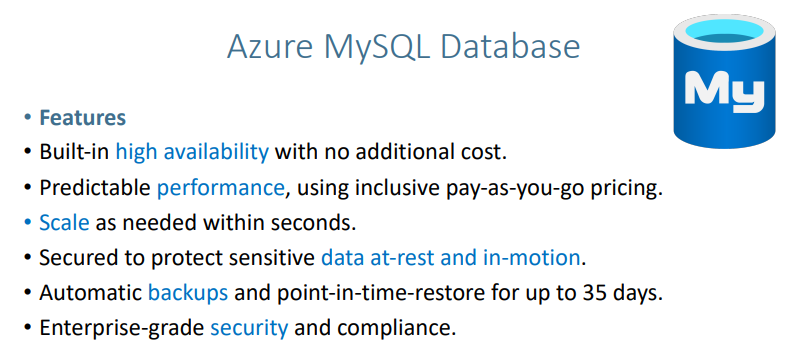
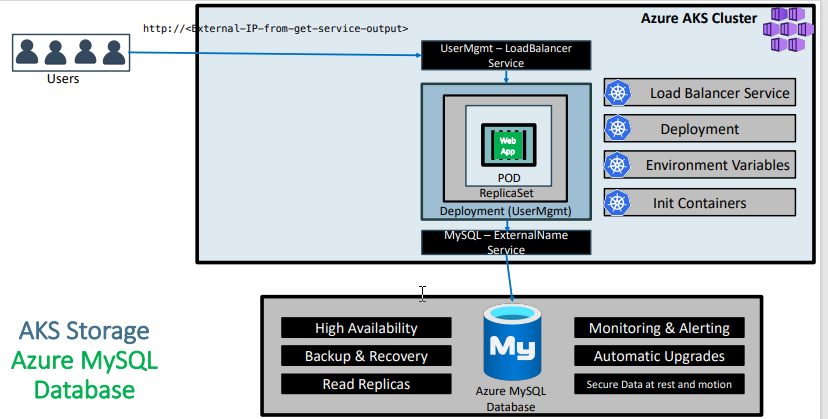
Kubernetes

AKS Storage – Azure MySQL Database azure my sql database as the underline ware

In previous we have implemented My sql database using azure disks here why we need to use azure my sql database instead of azure disks because one pod can connect to one disk at a time the high availability with single pod cannot work







53. Step-02: Create Azure MYSQL Database

Create Azure Database for MySQL servers

Go to Service Azure Database for MySQL servers

Click on Add

Basics

Project details

Subscription: Free Trial

Resource Group: aks-rg1

Server Details

Server name: akswebappdb (This name is based on availability - in your case it might be something else and it is unique)

Data source: none

Location: (US) East US

Version: 5.7 (default)

Compute + Storage

Pricing Tier: Basic

VCore: 1

Storage: 5GB

Storage Auto Growth: Yes

Backup Retention: 7 days

Locally Redundant: Yes

Administrative Account

Admin username: dbadmin

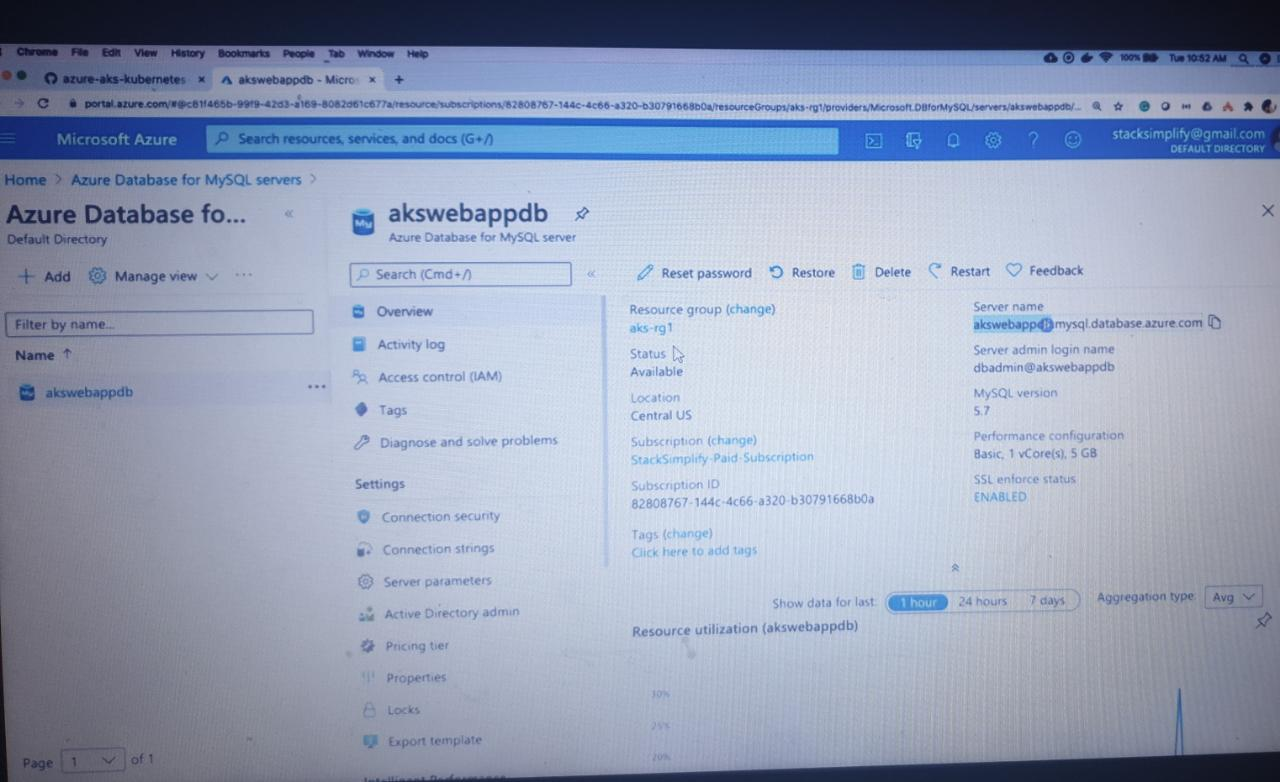
Password: Redhat1449

Confirm password: Redhat1449

Review + Create

It will take close to 15 minutes to create the database.

We see the unique server name below as mini highlighted



**Update Security Settings for Database**

* Go to **Azure Database for MySQL Servers** -> **akswebappdb**
* **Settings -> Connection Security**
  + **Very Important**: Enable **Allow Access to Azure Services**
  + Update Firewall rules to allow from local desktop (Add current client IP Address)
  + **SSL Settings**: Disabled
  + Click on **Save**
* It will take close to 15 minutes for changes to take place.

If we have my sql database locally then we can execute the below commands

# Template

mysql --host=mydemoserver.mysql.database.azure.com --user=myadmin@mydemoserver -p

#

mysql --host=akswebappdb.mysql.database.azure.com --user=dbadmin@akswebappdb -p

**Create Kubernetes externalName service Manifest and Deploy**

* Create mysql externalName Service
* **01-MySQL-externalName-Service.yml**

apiVersion: v1

kind: Service

metadata:

name: mysql

spec:

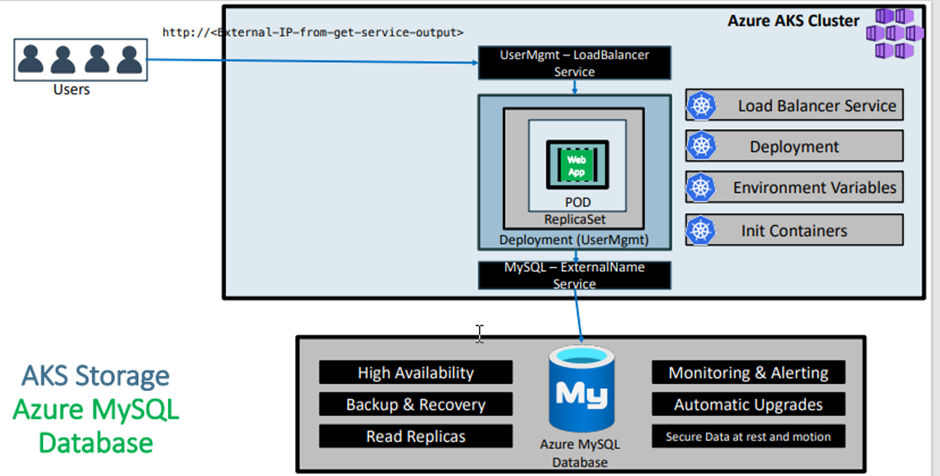
type: ExternalName

externalName: akswebappdb.mysql.database.azure.com

* **Deploy Manifest**

kubectl apply -f kube-manifests/01-MySQL-externalName-Service.yml

then we can say there will be connectivity between AKS cluster using MYSQL-External name to Azure My SQL database



# Template

kubectl run -it --rm --image=mysql:5.7.22 --restart=Never mysql-client -- mysql -h <AZURE-MYSQ-DB-HOSTNAME> -u <USER\_NAME> -p<PASSWORD>

# Replace Host Name of Azure MySQL Database and Username and Password

kubectl run -it --rm --image=mysql:5.7.22 --restart=Never mysql-client -- mysql -h akswebappdb.mysql.database.azure.com -u dbadmin@akswebappdb -pRedhat1449

mysql> show schemas;

mysql> create database webappdb;

mysql> show schemas;

mysql> exit

**In User Management WebApp deployment file change username from root to dbadmin@akswebappdb**

* **02-UserMgmtWebApp-Deployment.yml**

# Change From

- name: DB\_USERNAME

value: "root"

- name: DB\_PASSWORD

value: "dbpassword11"

# Change To dbadmin@<YOUR-Azure-MYSQL-DB-NAME>

- name: DB\_USERNAME

value: "dbadmin@akswebappdb"

- name: DB\_PASSWORD

value: "Redhat1449"

**Step-06: Deploy User Management WebApp and Test**

# Deploy all Manifests

kubectl apply -f kube-manifests/

# List Pods

kubectl get pods

# Stream pod logs to verify DB Connection is successful from SpringBoot Application

kubectl logs -f <pod-name>

**Step-07: Access Application**

# Get Public IP

kubectl get svc

# Access Application

http://<External-IP-from-get-service-output>

Username: admin101

Password: password101

**Step-08: Clean Up**

# Delete all Objects created

kubectl delete -f kube-manifests/

# Verify current Kubernetes Objects

kubectl get all

# Delete Azure MySQL Database

**Kubernetes - Secrets**

**Step-01: Introduction**

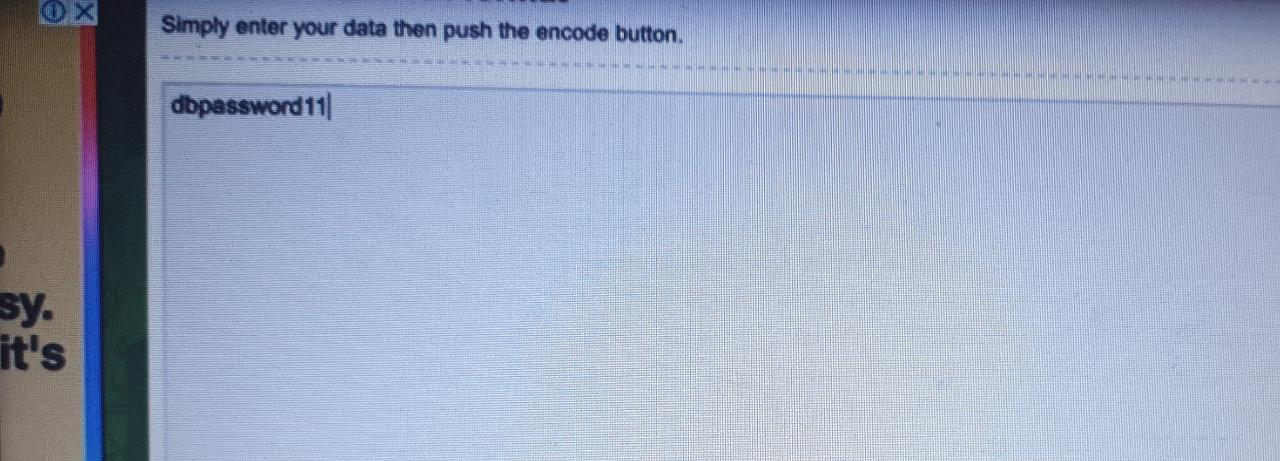
* Kubernetes Secrets let you store and manage sensitive information, such as passwords, OAuth tokens, and ssh keys.
* Storing confidential information in a Secret is safer and more flexible than putting it directly in a Pod definition or in a container image.

**Step-02: Create Secret for MySQL DB Password**

# Mac

echo -n 'dbpassword11' | base64

# URL: https://www.base64encode.org



click on encode below

generate the secret

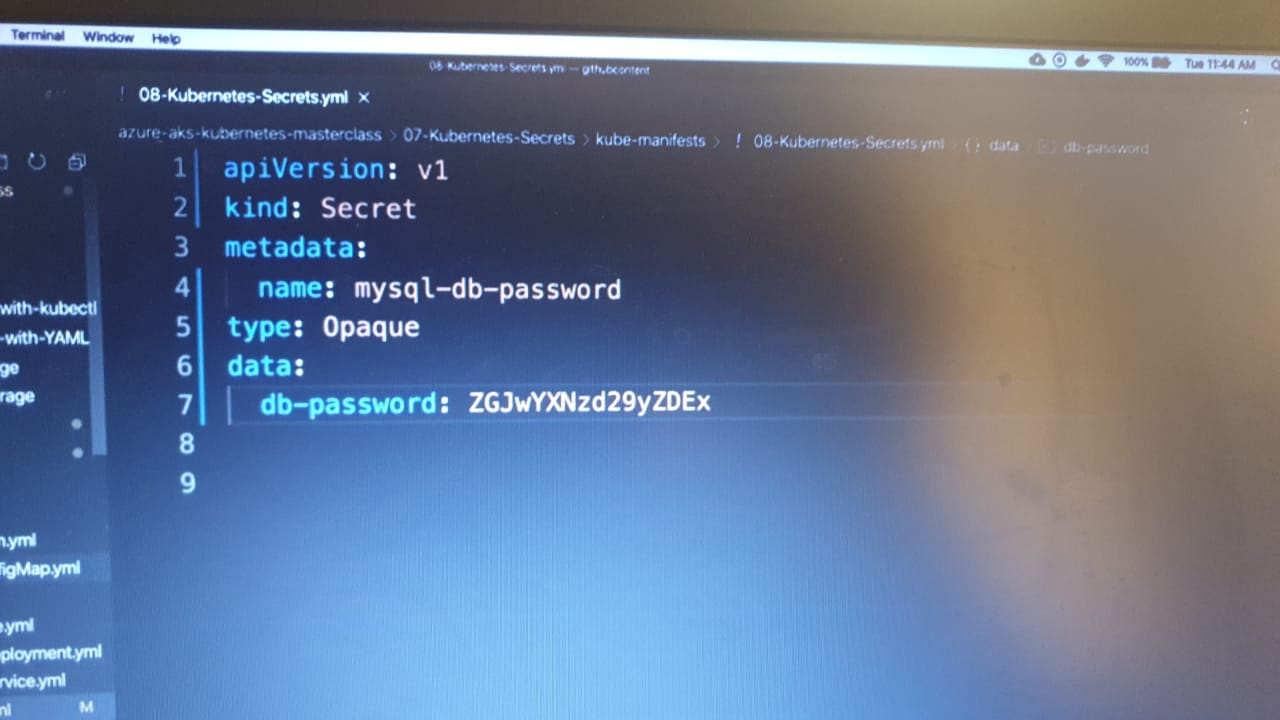
**Create Kubernetes Secrets manifest**

apiVersion: v1

kind: Secret

metadata:

name: mysql-db-password



#type: Opaque means that from kubernetes's point of view the contents of this Secret is unstructured.

#It can contain arbitrary key-value pairs.

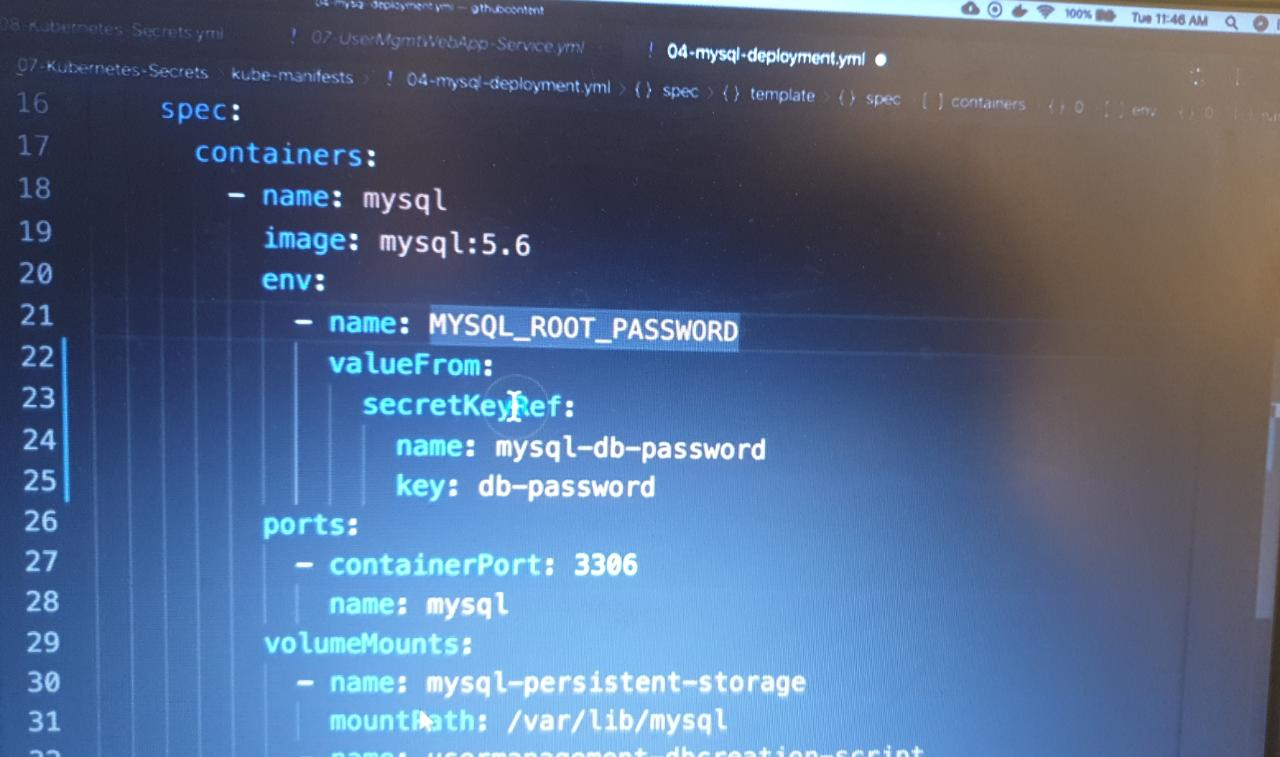
type: Opaque

data:

# Output of echo -n 'Redhat1449' | base64

db-password: ZGJwYXNzd29yZDEx

**Step-03: Update secret in MySQL Deployment for DB Password**



env:

- name: MYSQL\_ROOT\_PASSWORD

valueFrom:

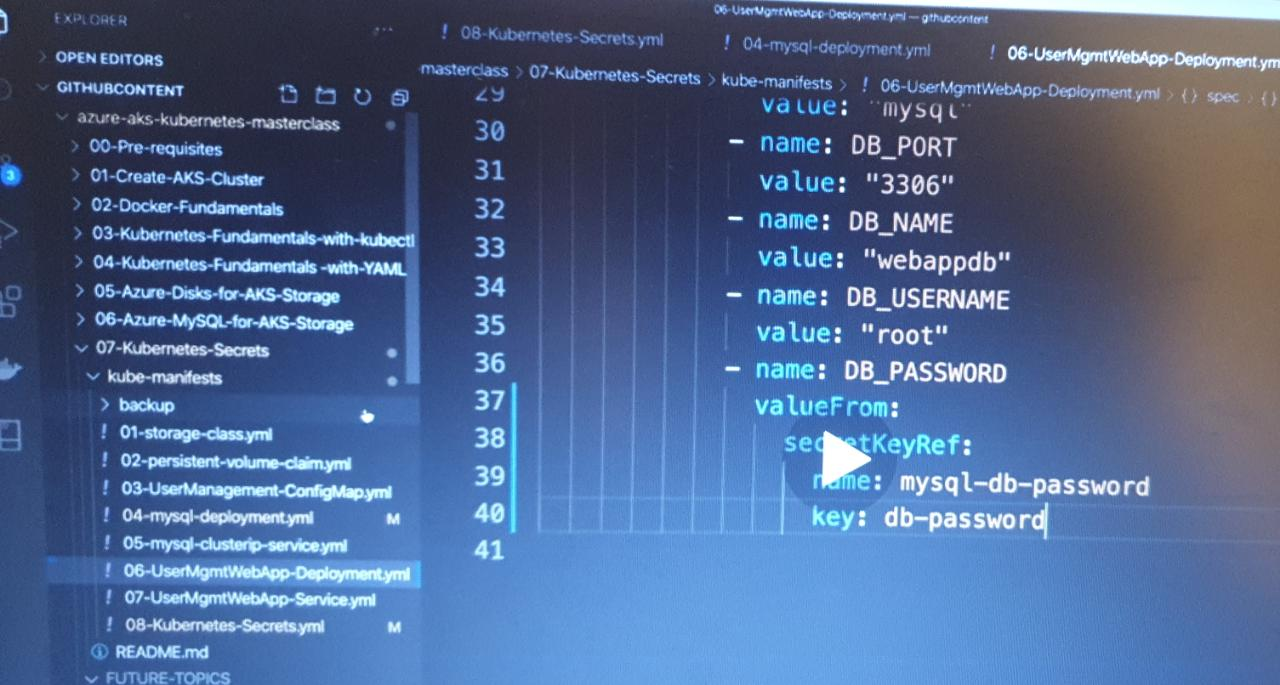
secretKeyRef:

name: mysql-db-password

key: db-password

**Step-04: Update secret in UWA Deployment**

* UMS means User Management Microservice



- name: DB\_PASSWORD

valueFrom:

secretKeyRef:

name: mysql-db-password

key: db-password

**Step-05: Create & Test**

# Create All Objects

kubectl apply -f kube-manifests/

# List Pods

kubectl get pods

# Get Public IP of Application

kubectl get svc

# Access Application

http://<External-IP-from-get-service-output>

Username: admin101

Password: password101

**Step-06: Clean-Up**

* Delete all k8s objects created as part of this section

# Delete All

kubectl delete -f kube-manifests/

# List Pods

kubectl get pods

# Verify sc, pvc, pv

kubectl get sc,pvc,pv