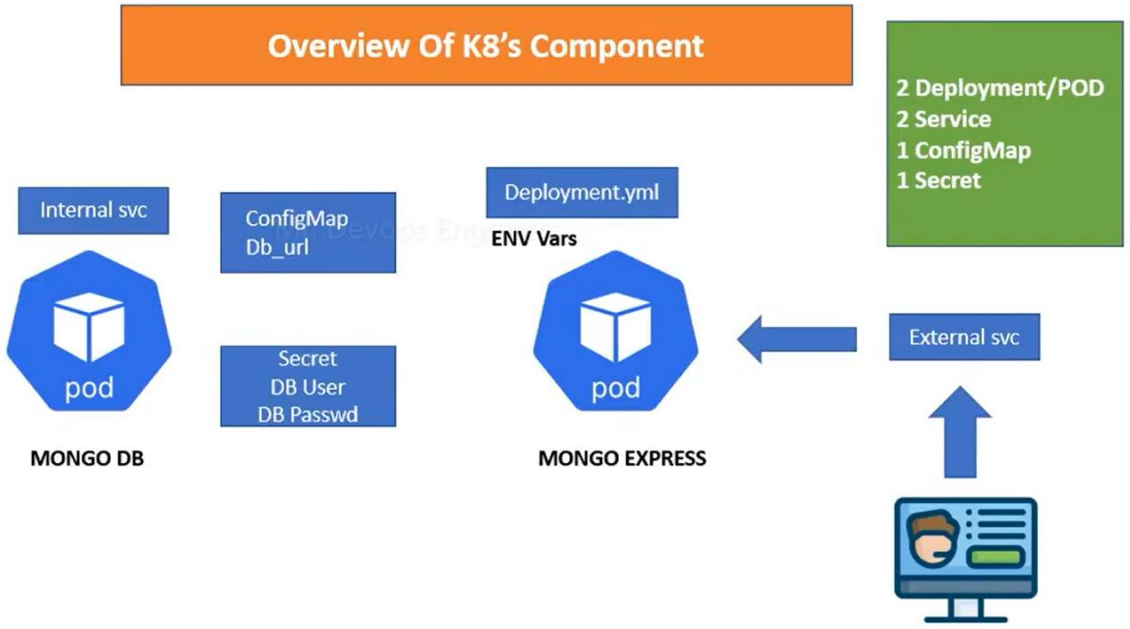
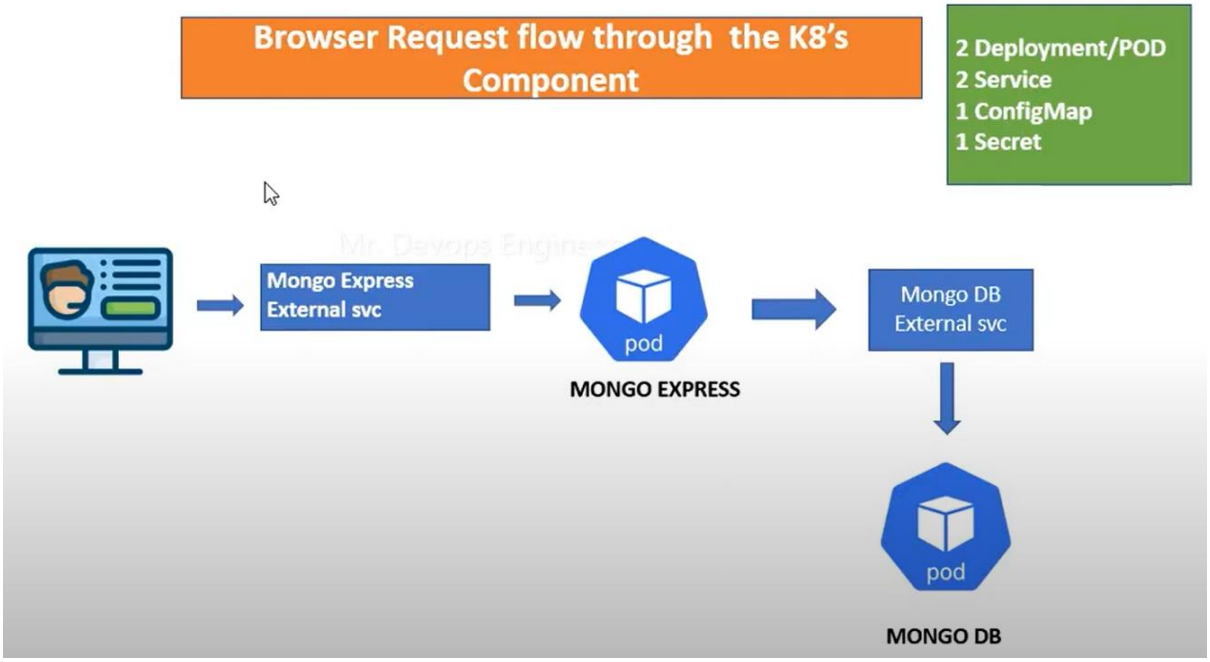


We are deploying 2 apps , mongo db and mongo express .
We will have mongodb pod .
To access mongo pod we need service – and since this is db we cant expose outside world . So we will create internal service so that internal cluster can connect .
We will have mongo express deployment , we need mongo db url so that we can access db from webapp .
We also need creds username and db password to authenticate .We will use configmap and secret to pass username and password .
In configmap we will store db url and in secret we will define usename password .
We want mongodb express from internet , we need external service for this .



Request flow ?
When request come from browser -> externak service of mongodb express (we will have ip along with service port)-> mondo express pod -> pod will be connected to internal service of mongp db . internal service will be mongo db url . -> request will go to mongo db pod .

We need minikube cluster .

Lets do mongo db deployment .
We need mongo_db.yaml manifest file .

In configuration we will add username and password referring to secret.yaml file .
So we need mongodb-secret.yaml manifest file also .

Also in secret file we need to mention username and password converted to base64 .

```
ubuntu@ip-172-31-47-135:~$ echo -n 'mongodb' | base64
bW9uZ29kYg==
ubuntu@ip-172-31-47-135:~$
```

Lets use same username and password .

We have to mention above base64 in secret and deployment file both wherever username and password required .

Lets apply secret in Kubernetes .

```
ubuntu@ip-172-31-47-135:~/projects$ vi secrects.yml
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f secrects.yml
secret/mongodb-secret created
ubuntu@ip-172-31-47-135:~/projects$ kubectl get all
NAME                                      READY   STATUS    RESTARTS      AGE
pod/myrelase-hello-world-798b49845c-vdw6d   1/1     Running   1 (13m ago)   60m

NAME                              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
service/kubernetes                ClusterIP   10.96.0.1       <none>        443/TCP    149m
service/myrelase-hello-world   ClusterIP   10.109.139.30   <none>        80/TCP     60m

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/myrelase-hello-world   1/1     1            1           60m

NAME                                               DESIRED   CURRENT   READY   AGE
replicaset.apps/myrelase-hello-world-798b49845c   1         1         1       60m
ubuntu@ip-172-31-47-135:~/projects$ kubectl get secrets
NAME                           TYPE                 DATA   AGE
mongodb-secret                 Opaque               2      30s
sh.helm.release.v1.myrelase.v1   helm.sh/release.v1   1      61m
sh.helm.release.v1.myrelase.v2   helm.sh/release.v1   1      57m
sh.helm.release.v1.myrelase.v3   helm.sh/release.v1   1      43m
ubuntu@ip-172-31-47-135:~/projects$
```

As we can see secret is created .

```
sh.helm.release.v1.myrelase.v3   helm.sh/release.v1   1      43m
ubuntu@ip-172-31-47-135:~/projects$ kubectl describe secret mongodb-secret
Name:         mongodb-secret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
mongodb-root-username:  7 bytes
mongodb-root-password:  7 bytes
ubuntu@ip-172-31-47-135:~/projects$
```

We have set the type as opaque if we don't do this then our password will be visible .

Lets run mongo db deployment .

```
              key: mongodb-root-password
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f mongo_db.yml
deployment.apps/mongodb-deployment created
ubuntu@ip-172-31-47-135:~/projects$ kubectl get deployments
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
mongodb-deployment     1/1     1            1           47s
myrelase-hello-world   1/1     1            1           67m
ubuntu@ip-172-31-47-135:~/projects$
```

Our deployment is created ,

To access our db we need internal service . so that component inside cluster can access db .
Lets create mongodb service also .
If service type is not mentioned then by default clusterip will be created .

```
ubuntu@ip-172-31-47-135:~/projects$ vi mongo_db.yml
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f mongo_db.yml
deployment.apps/mongodb-deployment unchanged
service/mongodb-service created
ubuntu@ip-172-31-47-135:~/projects$
```

Service also created .

Now Lets create mongo db express .

We will pass db url using configmap .
We have to map service of mongo db with

Lets create configmap .

```
ubuntu@ip-172-31-47-135:~/projects$ vi configmal.yml
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f configmal.yml
configmap/mongodb-configmap created
ubuntu@ip-172-31-47-135:~/projects$ kubectl get configs
error: the server doesn't have a resource type "configs"
ubuntu@ip-172-31-47-135:~/projects$ kubectl get configmap
NAME                DATA   AGE
kube-root-ca.crt    1      3h2m
mongodb-configmap   1      57s
ubuntu@ip-172-31-47-135:~/projects$
```

Configmap is created .

```
mongodb-configmap   1      57s
ubuntu@ip-172-31-47-135:~/projects$ kubectl describe configmap mongodb-configmap
Name:         mongodb-configmap
Namespace:    default
Labels:       <none>
Annotations:  <none>

Data
====
database_url:
----
mongodb-service

BinaryData
====

Events:  <none>
ubuntu@ip-172-31-47-135:~/projects$
```

Lets create mongo express .

```
Events:  <none>
ubuntu@ip-172-31-47-135:~/projects$ vi mango_express.yml
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f mango_express.yml
deployment.apps/mongo-express created
ubuntu@ip-172-31-47-135:~/projects$
```

Now expose deployment of mongodb express so that we can access the application .

We will use service type as load balancer .

```
ubuntu@ip-172-31-47-135:~/projects$ vi mango_express.yml
ubuntu@ip-172-31-47-135:~/projects$ kubectl apply -f mango_express.yml
deployment.apps/mongo-express unchanged
service/mongo-express-service created
ubuntu@ip-172-31-47-135:~/projects$ kubectl get svc
NAME                    TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes              ClusterIP      10.96.0.1        <none>        443/TCP          3h11m
mongo-express-service   LoadBalancer   10.107.237.189   <pending>     8081:30000/TCP   14s
mongodb-service         ClusterIP      10.108.143.243   <none>        27017/TCP        25m
myrelase-hello-world    ClusterIP      10.109.139.30    <none>        80/TCP           102m
ubuntu@ip-172-31-47-135:~/projects$
```

Service deployment is done .

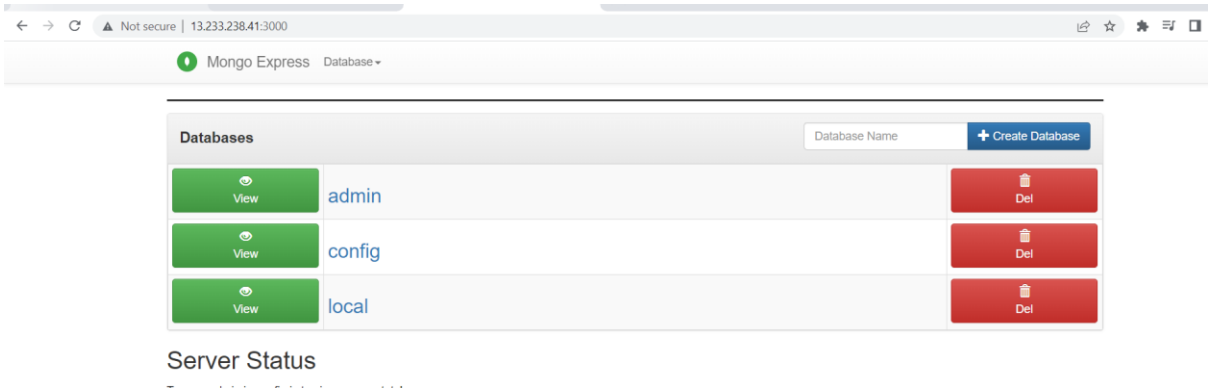
Start the service .

```
myrelase-hello-world    ClusterIP      10.109.139.30    <none>        80/TCP
ubuntu@ip-172-31-47-135:~/projects$ minikube service mongo-express-service
|-----------|-----------------------|-------------|---------------------------|
| NAMESPACE |         NAME          | TARGET PORT |            URL            |
|-----------|-----------------------|-------------|---------------------------|
| default   | mongo-express-service |        8081 | http://192.168.49.2:30000 |
|-----------|-----------------------|-------------|---------------------------|
* Opening service default/mongo-express-service in default browser...
  http://192.168.49.2:30000
ubuntu@ip-172-31-47-135:~/projects$
```

Try accessing .

Lets do port forwarding .

```
ubuntu@ip-172-31-47-135:~/projects$ kubectl get svc
NAME                    TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes              ClusterIP      10.96.0.1        <none>        443/TCP          3h11m
mongo-express-service   LoadBalancer   10.107.237.189   <pending>     8081:30000/TCP   14s
mongodb-service         ClusterIP      10.108.143.243   <none>        27017/TCP        25m
myrelase-hello-world    ClusterIP      10.109.139.30    <none>        80/TCP           102m
ubuntu@ip-172-31-47-135:~/projects$ minikube service mongo-express-service
|-----------|-----------------------|-------------|---------------------------|
| NAMESPACE |         NAME          | TARGET PORT |            URL            |
|-----------|-----------------------|-------------|---------------------------|
| default   | mongo-express-service |        8081 | http://192.168.49.2:30000 |
|-----------|-----------------------|-------------|---------------------------|
* Opening service default/mongo-express-service in default browser...
  http://192.168.49.2:30000
ubuntu@ip-172-31-47-135:~/projects$ kubectl port-forward svc/mongo-express-service 3000:30000 --address 0.0.0.0
error: Service mongo-express-service does not have a service port 30000
ubuntu@ip-172-31-47-135:~/projects$ kubectl port-forward svc/mongo-express-service 3000:8081 --address 0.0.0.0
Forwarding from 0.0.0.0:3000 -> 8081
```



As we can see our application is accessible after port forwarding .

When we create database it will forward traffic to mongodb .
So we have deployed web and db server using Kubernetes .