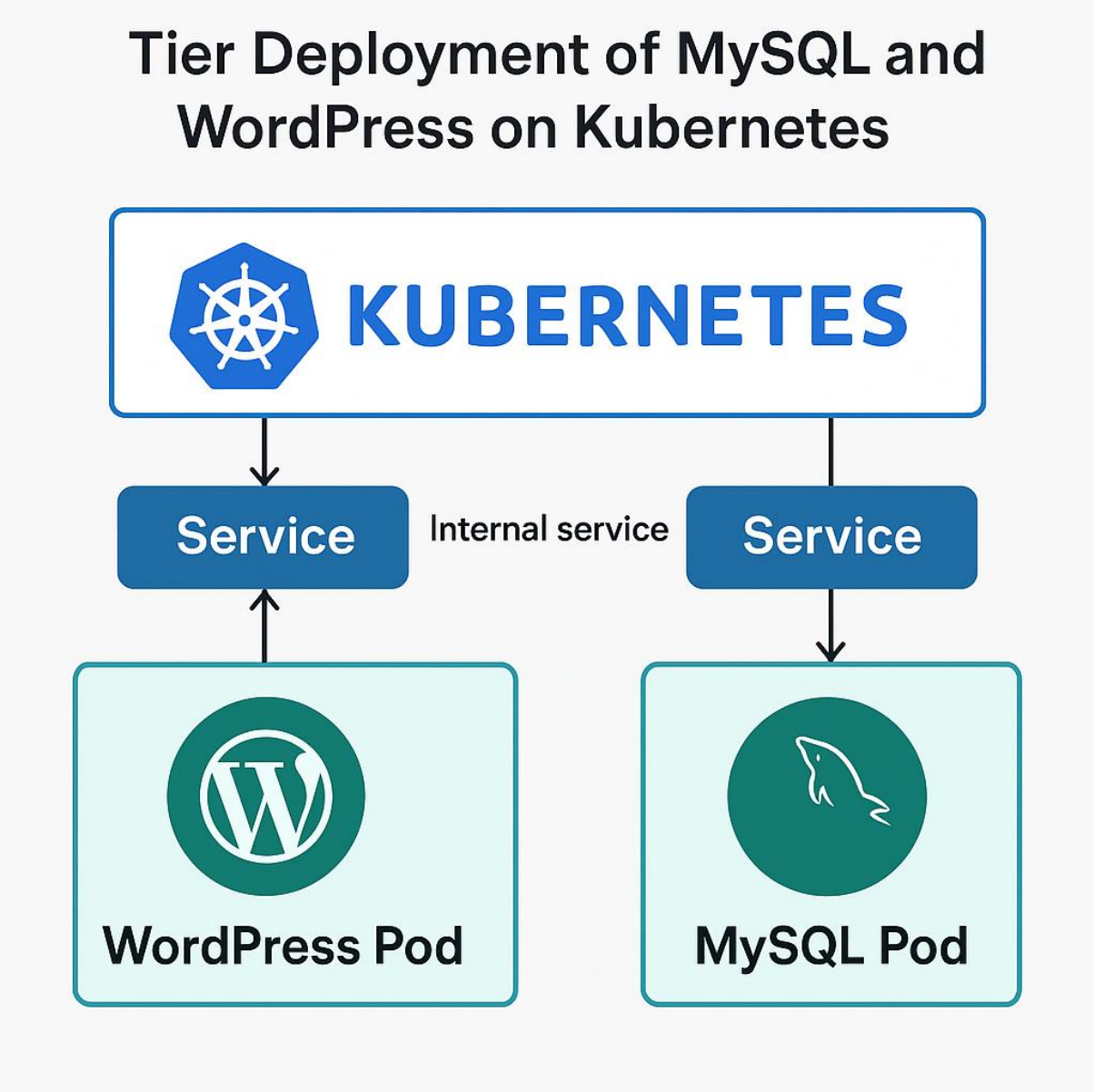
Deployment of MySQL and WordPress on Kubernetes

Objective

To deploy a 2-tier application on Kubernetes where:

- **Tier 1: MySQL acts as the database backend**
- **Tier 2: WordPress acts as the frontend PHP application**
- **Services are exposed using ClusterIP and NodePort**



1️⃣ MySQL Deployment (Database Tier)

✅ Step 1: Generate Deployment YAML

kubectl create deployment dbserver --image=mysql:latest --dry-run=client -o yaml > dbserver.yaml

```
controlplane:~$  kubectl create deployment dbserver --image mysql:latest --dry-run=client -o yaml > dbserver.yaml
controlplane:~$ ls
dbserver.yaml  filesystem
```

📝 Edit the YAML (dbserver.yaml)

```
env:
  - name: MYSQL_ROOT_PASSWORD
    value: mypass@123
  - name: MYSQL_DATABASE
    value: unnati
  - name: MYSQL_USER
    value: satyajit
  - name: MYSQL_PASSWORD
    value: mypass@12345
```

Apply the Deployment

kubectl apply -f dbserver.yaml

kubectl get deployments

kubectl get pods

```
controlplane:~$ kubectl get deployment
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
dbserver   1/1     1            1           62s
controlplane:~$ kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
dbserver-7fb7c7566f-dtqz8   1/1     Running   0          88s
controlplane:~$
```

For the confirmation you go to container and login in mysql database and using show databases you can check your database is created or not.

```
controlplane:~$ kubectl exec -it dbserver-7c87459794-n7lp9 -- bash
bash-5.1# mysql -usatyajit -pmypass@12345
```

```
mysql> show databses;
ERROR 1064 (42000): You have a
hat corresponds to your MySQL
r 'databses' at line 1
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| performance_schema |
| unnati             |
+--------------------+
3 rows in set (0.005 sec)

mysql>
```

🌐 **Expose MySQL as a Service**

kubectl expose deployment dbserver --port=3306
kubectl get svc

```
controlplane:~$ kubectl expose deployment dbserver --port 3306
service/dbserver exposed
controlplane:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
dbserver     ClusterIP   10.99.136.70   <none>        3306/TCP   3s
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP    18d
controlplane:~$
```

2️⃣ WordPress Deployment (Application Tier)

✅ Step 2: Generate Deployment YAML

kubectl create deployment mywordpress --image=wordpress:latest --dry-run=client -o yaml > wp.yaml

```
controlplane:~$ kubectl create deployment mywordpress --image wordpress:late
st --dry-run=client -o yaml > wp.yaml
```

📝 Edit the YAML (wp.yaml) add this in yaml file

```
env:
  - name: WORDPRESS_DB_HOST
    value: dbserver
  - name: WORDPRESS_DB_USER
    value: satyajit
  - name: WORDPRESS_DB_PASSWORD
    value: mypass@12345
  - name: WORDPRESS_DB_NAME
    value: unnati
resources: {}
```

📦 Apply the Deployment

kubectl apply -f wp.yaml

kubectl get deployments

kubectl get pods

```
controlplane:~$ kubectl apply -f wp.yaml
deployment.apps/mywordpress created
controlplane:~$ vim wp.yaml
```

```
controlplane:~$ kubectl get deployments
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
dbserver      1/1     1            1           18m
mywordpress   1/1     1            1           3m30s
controlplane:~$ kubectl get pods
NAME                           READY   STATUS    RESTARTS   AGE
dbserver-7c87459794-n7lp9      1/1     Running   0          13m
mywordpress-64698b7578-rdm5h   1/1     Running   0          3m42s
controlplane:~$
```

🌎 Expose WordPress with NodePort

kubectl expose deployment mywordpress --port=80 --type=NodePort

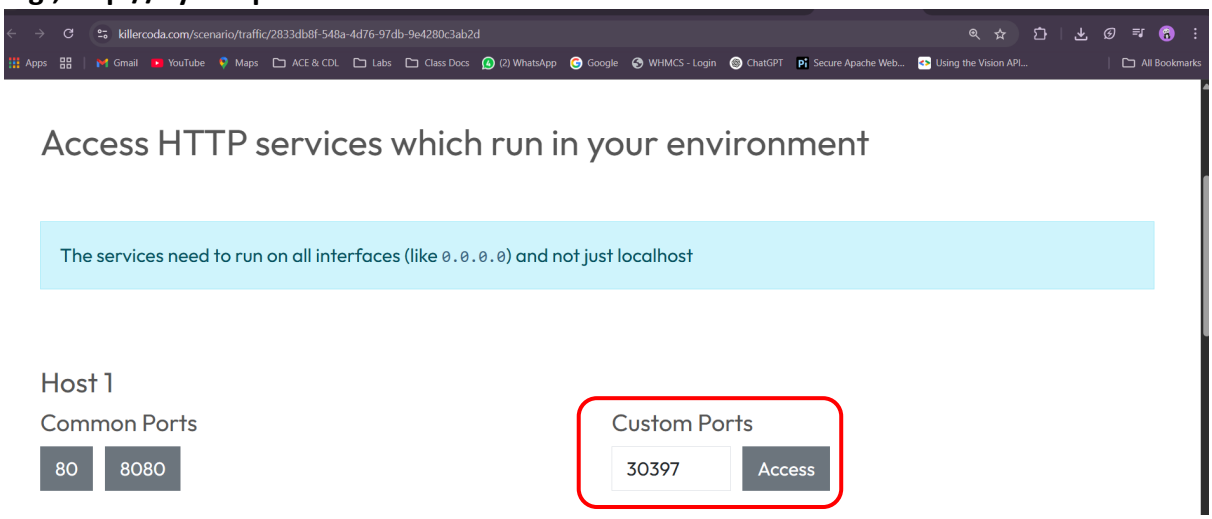
kubectl get svc

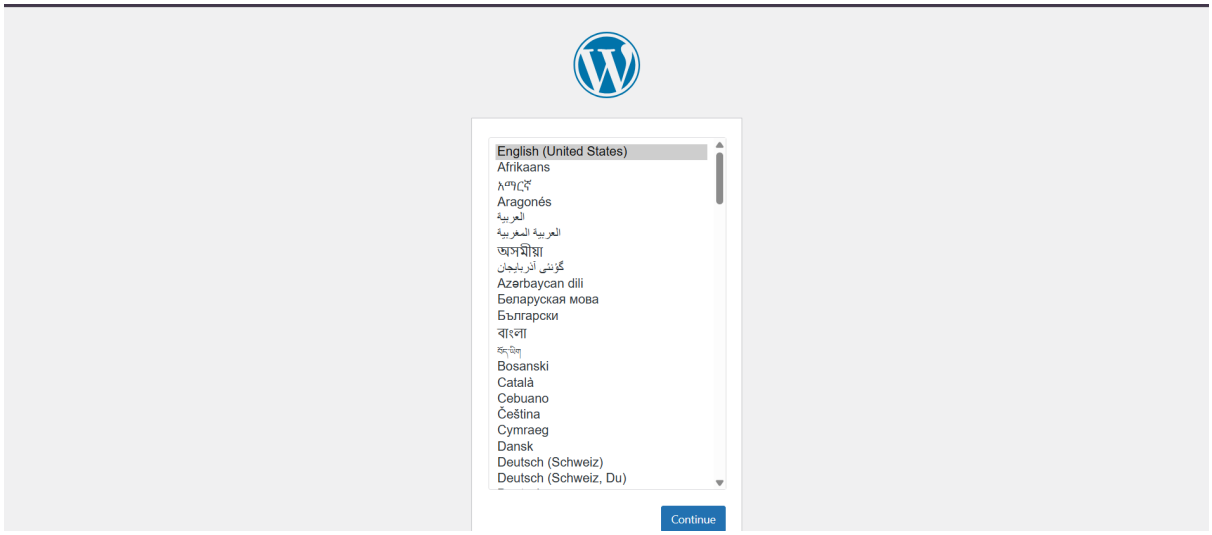
```
controlplane:~$ kubectl get svc
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
dbserver      ClusterIP   10.99.136.70    <none>        3306/TCP       12m
kubernetes    ClusterIP   10.96.0.1       <none>        443/TCP        18d
mywordpress   NodePort    10.98.227.119   <none>        80:30397/TCP   9s
controlplane:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION     CONTAINER
-RUNTIME
controlplane   Ready    control-plane   18d   v1.32.1   172.30.1.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   container
d://1.7.24
node01         Ready    <none>          18d   v1.32.1   172.30.2.2    <none>        Ubuntu 24.04.1 LTS   6.8.0-51-generic   container
d://1.7.24
controlplane:~$
```

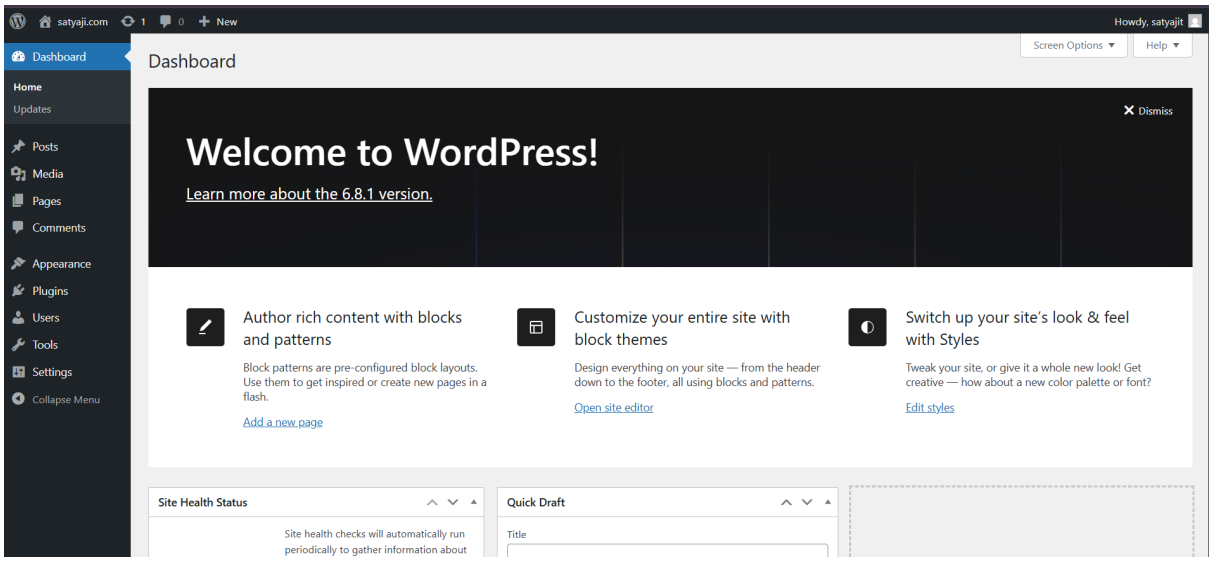
🌐 Accessing the WordPress Site

kubectl get nodes -o wide

Access via:
http://<Node-IP>:<NodePort>
e.g., http://<your ip >:31046






satyaji.com
New
Howdy, satyajit
Dashboard
Home
Updates
Posts
Media
Pages
Comments
Appearance
Plugins
Users
Tools
Settings
Collapse Menu
Dashboard
Screen Options
Help
Dismiss
Welcome to WordPress!
Learn more about the 6.8.1 version.
Author rich content with blocks and patterns
Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.
Add a new page
Customize your entire site with block themes
Design everything on your site — from the header down to the footer, all using blocks and patterns.
Open site editor
Switch up your site's look & feel with Styles
Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?
Edit styles
Site Health Status
Site health checks will automatically run periodically to gather information about
Quick Draft
Title

THANK YOU!