

Cheat Sheet: Modifying the Events App to Use the Database

Events-Internal Environment Variables

- The Events Internal service was written to look for several environment variables
 - DBHOST, DBUSER, DBPASSWORD, DBDATABASE
 - These variables tell it how to connect to a database
 - If these variables do not exist, the event data is stored in a local array
 - That is what has been happening so far in the class
- Now that we have MariaDB running, we just need to set the environment variables for the database
 - The app will then start storing the event data in the database

Modify the internaldeployment.yaml

- Edit your internaldeployment.yaml file
 - At the end of the file, add the highlighted lines shown here
 - That sets four environment variables
 - You can ignore warnings about duplicate keys
 - Be sure to note the indentation
- Then reapply the file:

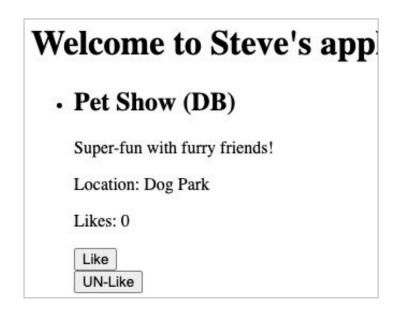
kubectl apply -f internaldeployment.yaml

 This should cause the pods for this deployment to be replaced

```
spec:
containers:
 - name: demo-api
  image: coursedemos/internal:v1.0
   env:
   name: SERVICE PORT
    value: "8082"
  ports:
   - containerPort: 8082
  env:
   name: DBHOST
    value: "database-server-mariadb"
  - name: DBUSER
    value: "root"
   name: DBPASSWORD
    valueFrom:
      secretKeyRef:
        name: database-server-mariadb
        key: mariadb-root-password
   name: DBDATABASE
    value: "events db"
```

Testing the Database

- List the pods and verify the internal pod(s) were replaced:
 kubectl get pods
- Test the application by viewing the service external address in a browser
 - If you need the address again:
 kubectl get service
- The DB initialization code added two rows to the database that have "(DB)" in their title
 - This is just so you can verify the DB is working
- Try adding some new events



Experiment with Replicas

- Scale the external pods:
 - Modify the externaldeployment.yaml to have three replicas
 - Apply the file and test the application
 - kubectl apply -f externaldeployment.yaml
 - Everything should still work fine
- Scale the internal pods:
 - Modify the internal deployment to have three replicas
 - Apply the file and test the application
 - kubectl apply -f internaldeployment.yaml
 - Everything should still work fine
 - This is because the internal service is now storing state in a database

Success

 Congratulations! You have successfully added a database to the Kubernetes services