



ROI Training

Lab:

**Modifying the Events-API App
to Use the Database**

Verify Minikube Is Running

- From a codespaces terminal, check if Minikube is still running with:
`minikube status`
- If it is stopped, start it again with:
`minikube start`

Kubernetes config Folder

- Remember, anytime you are applying Kubernetes YAML files, you must be in the folder containing the YAML files
 - In your codespaces terminal, change into the correct folder:
`cd /workspaces/eventsapp/kubernetes-config`
- Verify the events app is still running, apply the yaml files if it is not:
`kubectl get deploy`
`kubectl get svc`

```
$ kubectl get deploy
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
events-api    1/1     1             1           115s
events-web    1/1     1             1           89s
$ kubectl get svc
NAME          TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
events-api-svc ClusterIP      10.108.179.92 <none>        8082/TCP         113s
events-web-svc LoadBalancer  10.105.3.238  <pending>     80:31607/TCP     93s
kubernetes    ClusterIP      10.96.0.1     <none>        443/TCP          3m19s
```

Events-API Environment Variables

- The Events-API app 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 a 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

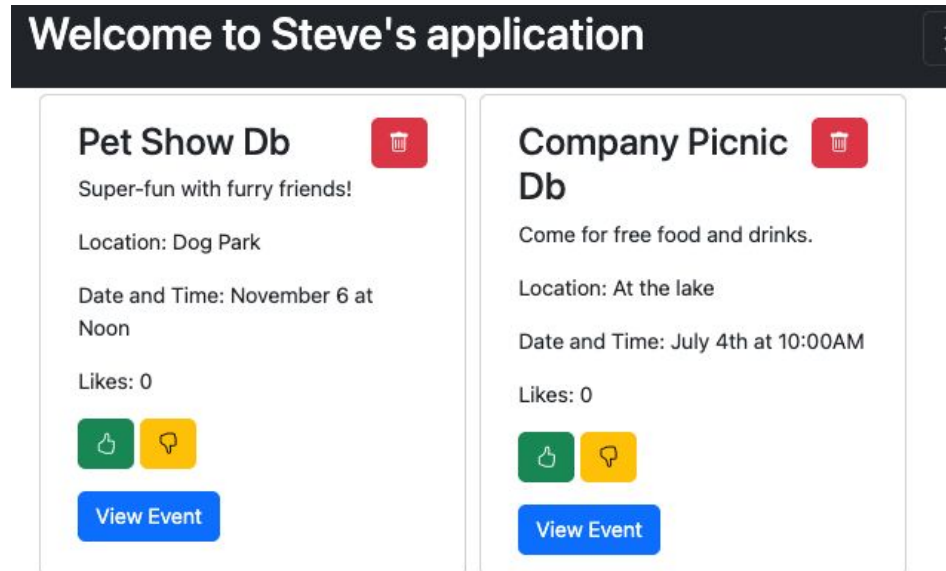
Modify the api-deployment.yaml

- Edit your `api-deployment.yaml` file
 - At the end of the file, add the highlighted lines shown here
 - That sets four environment variables
 - Be sure to note the indentation
- Then reapply the file:
`kubectl apply -f api-deployment.yaml`
- This should cause the pods for this deployment to be replaced

```
.....
spec:
  containers:
  - image: EVENTS-API-IMAGE-URL-HERE
    name: events-api
    ports:
    - containerPort: 8082
    env:
    - name: DBHOST
      value: "database-server-mariadb.default"
    - 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 events-api pod(s) were replaced:
`kubectl get pods`
- Open a new terminal (+ button) and create the port forward and tunnel:
`minikube tunnel & kubectl port-forward service/events-web-svc 8080:80`
 - Click the **Open in Browser** button
- The DB initialization job 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 events-web service:
 - Modify the `web-deployment.yaml` to have 3 replicas
 - Apply the file and test the application
 - Everything should still work fine
- Scale the events-api service:
 - Modify the `api-deployment.yaml` to have 3 replicas
 - Apply the file and test the application
 - Everything should still work fine
 - This is because the events-api service is now storing state in a database

Clean Up

- Stop the port forward:
 - In the terminal window running the port forward, press **CTRL+C** to stop it
 - Then run the following command to stop the tunnel:
`pkill -f "minikube tunnel"`
- There is no need to clean up the deployments, services, or pods
- Leave them running on your cluster

Syncing the Changes to Git

- Commit these changes to your Git repository
 - On the left side, click the **Source Control** button 
 - Be sure ALL changes are staged by clicking in the + button
 - Type a commit message of: **Added Events app start code** and click the **Commit** button
 - Press the **Sync Changes** button and press **OK** to push the changes
- The code has now been saved to your **eventsapp** Git repository created earlier

Success

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