



**Lab:**

**Performing Rolling Updates and  
Blue/Green Deployments**

# 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`
- Verify the tunnel and port forward from the last lab is stopped:
  - In the terminal window running the port forward, press **CTRL+C** to stop it
  - Run the following command to stop the tunnel:  
`pkill -f "minikube tunnel"`

# Verify Events App Is Running

- 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
```

# Increasing the Replicas

- To make the rolling update more interesting, you will increase the number of replicas of your events-web pods:
  - Modify the web-deployment.yaml to have four replicas
  - Apply the web-deployment.yaml file
  - Get the pods and verify four are running

```
$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
events-api-b7c5f7ccf-vg95r	1/1	Running	0	38m
events-web-5d5485cd7d-4ptd2	1/1	Running	0	6s
events-web-5d5485cd7d-4snn2	1/1	Running	0	6s
events-web-5d5485cd7d-68cv1	1/1	Running	0	6s
events-web-5d5485cd7d-72t5c	1/1	Running	0	23m

- Create a tunnel (we do not need a port forward here)
  - Once the tunnel starts press **Enter** to get the command prompt back  
`minikube tunnel &`

# Rolling Updates

- View the service
  - `kubectl get service`
  - And Record the External IP of the **events-web-svc** service
- You will now upgrade to Version 2.0 of the events-web
  - Modify the `web-deployment.yaml` and change the version of the container image from `v1.0` to `v2.0`

```
- image: your-docker-hub-id/events-website:v2.0
```

- You don't have to worry about changing any of the labels at this point
- Do not apply the yaml yet

# Rolling Updates (continued)

- Open a new terminal window and run the following command to request the app's page in a loop and display the welcome message from the page:  
`while true; do curl -s http://SERVICE-IP-HERE/ | grep -s Welcome; sleep 1; done;`
- Open a new terminal window (you should have 3 now)
  - Run the following command to watch the pods:  
`kubectl get pods -w`
- Switch back to the first terminal, run the following command and then quickly switch to the terminal that is running the get pods watch command:  
`kubectl apply -f web-deployment.yaml`
- You should see the pods being updated 25% (which is 1 pod) at a time

# Rolling Updates (continued)

- Switch to the terminal running the while loop
  - The welcome message will include Version 2.0 when the app is updated
- Because this is a rolling update, you may see the messages toggle back and forth until all version 1 pods are terminated
  - After a few seconds, it should only return Version 2.0

```
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
```

# Rollback

- You will now roll back to Version 1.0 of the events-web
  - Verify in one terminal you are still running the command to watch the pods:  
`kubectl get pods -w`
  - In the other terminal, run the following command:  
`kubectl rollout undo deployments/events-web`
  - Switch to the terminal running the watch command
    - You should see the pods being rolled back 25% at a time
  - Switch to the terminal running the loop
    - You should see the original version is now running
- Feel free to perform the update and roll back again if you want to see it again
  - Press **CTRL+C** in the terminal running the pod watch command



# Blue/Green Deployments

- Ensure the events-web is back on Version 1.0:
  - Modify the web-deployment.yaml and change the version of the container image back to v1.0
  - Apply the web-deployment.yaml file
- You will now make another copy of the web-deployment.yaml file
  - `cp web-deployment.yaml web-deployment-v2.yaml`
  - Edit the new web-deployment-v2.yaml file
  - Modify line 6 from "name: events-web" to "name: events-web-v2"
  - Modify the "ver: v1.0" labels to "ver: v2.0" (two places)
  - Modify the container image URL from v1.0 to v2.0
  - Leave the app: events-web labels as is

# The web-deployment-v2.yaml

- Your file should look like this (your dockerhub-id will be different):

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: events-web
  name: events-web-v2
spec:
  replicas: 4
  selector:
    matchLabels:
      app: events-web
      ver: v2.0
  template:
    metadata:
      labels:
        app: events-web
        ver: v2.0
```

```
spec:
  containers:
    - image: dockerhub-id/events-website:v2.0
      name: events-web
      imagePullPolicy: "Always"
      ports:
        - containerPort: 8080
      env:
        - name: SERVER
          value: "http://events-api-svc:8082"
```

# Create the events-web-2.0 Deployment

- Apply the new web-deployment-v2.yaml

```
kubectl apply -f web-deployment-v2.yaml
```

- Get the deployments and pods

```
kubectl get deployments
```

```
kubectl get pods
```

- In the terminal running the while Loop, you should see version 1

```
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
```

```
$ kubectl get deployments
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
events-api	1/1	1	1	88m
events-web	4/4	4	4	88m
events-web-v2	4/4	4	4	9m43s

```
$
```

```
$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
events-api-b7c5f7ccf-vg95r	1/1	Running	0	89m
events-web-5d5485cd7d-29wjr	1/1	Running	0	37m
events-web-5d5485cd7d-7k728	1/1	Running	0	36m
events-web-5d5485cd7d-82lpl	1/1	Running	0	37m
events-web-5d5485cd7d-w7vvh	1/1	Running	0	37m
events-web-v2-7cb5697d84-ggl8c	1/1	Running	0	10m
events-web-v2-7cb5697d84-hsqdc	1/1	Running	0	10m
events-web-v2-7cb5697d84-qwb4q	1/1	Running	0	10m
events-web-v2-7cb5697d84-vgwrq	1/1	Running	0	10m

# Switch the Load Balancer Labels

- Perform the Blue/Green deployment
  - Edit the `web-service.yaml`
  - In the selector, change "`ver: v1.0`" to "`ver: v2.0`"
  - Apply the `web-service.yaml` and switch to the terminal running the loop
  - It should quickly switch all requests to Version 2
- Rollback
  - Edit the `web-service.yaml`
  - In the selector, change "`ver: v2.0`" to "`ver: v1.0`"
  - Apply the `web-service.yaml` and switch to the terminal running the loop
  - It should quickly switch all requests to Version 1

# Canary Release

- Edit the web-deployment-v2.yaml file
  - Modify the replicas from 4 to 1
  - Apply the web-deployment-v2.yaml file
- Edit the web-service.yaml
  - In the selector, remove the entire “`ver: v1.0`” line so the entire selector is:

```
selector:  
  app: events-web
```
  - This will cause the load balancer to select all pods not matter the version
  - Apply the web-service.yaml file

# Testing the Canary Release

- Switch to the terminal running the loop
  - You should see both version 1 and version 2
- The app is now load balancing between four copies of Version 1.0 and one copy of Version 2.0
  - So you should see version 1 more often than version 2


```
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application</h1>
<h1>Welcome to Steve's application **Version 2.0**</h1>
```

# Clean Up

- Edit the web-service.yaml
  - In the selector, put the “`ver: v1.0`” line back in so the selector looks like:
  - Apply the web-service.yaml file
- Edit the web-deployment.yaml
  - Set the replicas to 2
  - Apply the web-deployment.yaml file
- Delete the Version 2 deployment with:  
`kubectl delete deployment events-web-v2`
- Verify the terminal running the while loop now only displays version 1 welcome
- Use **CTRL+C** to cancel the while loop and the pod watch
- Stop the tunnel:  
`pkill -f "minikube tunnel"`

```
selector:  
  app: events-web  
  ver: v1.0
```

# 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 updated Kubernetes workloads
  - Performed a rolling update of your pods
  - Implemented a Blue/Green deployment
  - Created a simple canary release