

# Lab: Manage Application Updates

Update two live applications to their latest releases as identified by non-floating tags.

## Outcomes

You should be able to configure Deployment objects with images and triggers, and configure image stream tags and aliases.

As the student user on the workstation machine, use the `lab` command to prepare your system for this exercise.

This command ensures that all resources are available for this exercise. It also creates the `updates-review` project and deploys two applications, `app1` and `app2`, in that project.

The command creates the `/home/student/D0180/labs/updates-review/resources.txt` file. The `resources.txt` file contains the name of the images that you use during the exercise. You can use the file to copy and paste these image names.

```
[student@workstation ~]$ lab start updates-review
```

## Instructions

The API URL of your OpenShift cluster is `https://api.ocp4.example.com:6443`, and the `oc` command is already installed on your workstation machine.

Log in to the OpenShift cluster as the developer user with the developer password. Use the `updates-review` project for your work.

1. Your team created the `app1` deployment in the `updates-review` project from the `registry.ocp4.example.com:8443/redhattraining/php-ssl:latest` container image. Recently, a developer in your organization pushed a new version of the image and then reassigned the `latest` tag to that version.

Reconfigure the `app1` deployment to use the `1-222` static tag instead of the `latest` floating tag, to prevent accidental redeployment of your application with untested image versions that your developers can publish at any time.

Log in to the OpenShift cluster.

```
[student@workstation ~]$ oc login -u developer -p developer \
  https://api.ocp4.example.com:6443
Login successful.
...output omitted...
```

Set the `updates-review` project as the active project.

```
[student@workstation ~]$ oc project updates-review
...output omitted...
```

Verify that the app1 deployment uses the latest tag. Retrieve the container name.

```
[student@workstation ~]$ oc get deployment/app1 -o wide
NAME    READY  ...  CONTAINERS  IMAGES ...
app1   1/1    ...  php-ssl      registry...:8443/redhatraining/php-ssl:latest ...
```

In the Deployment object, change the image to registry.ocp4.example.com:8443/redhat

```
[student@workstation ~]$ oc set image deployment/app1 \
  php-ssl=registry.ocp4.example.com:8443/redhatraining/php-ssl:1-222
deployment.apps/app1 image updated
```

Verify your work.

```
[student@workstation ~]$ oc get deployment/app1 -o wide
NAME    READY  ...  CONTAINERS  IMAGES ...
app1   1/1    ...  php-ssl      registry...:8443/redhatraining/php-ssl:1-222 ...
```

2. The app2 deployment is using the php-ssl:1 image stream tag, which is an alias for the php-ssl:1-222 image stream tag.

Enable image triggering for the app2 deployment, so that whenever the php-ssl:1 image stream tag changes, OpenShift rolls out the application. You test your configuration in a later step, when you reassign the php-ssl:1 alias to a new image stream tag.

Retrieve the container name from the Deployment object.

```
[student@workstation ~]$ oc get deployment/app2 -o wide
NAME    READY  UP-TO-DATE  AVAILABLE  AGE    CONTAINERS  ...
app2   1/1     1          1          21m    php-ssl     ...
```

Add the image trigger to the Deployment object.

```
[student@workstation ~]$ oc set triggers deployment/app2 \
  --from-image php-ssl:1 --containers php-ssl
deployment.apps/app2 triggers updated
```

Verify your work.

```
[student@workstation ~]$ oc set triggers deployment/app2
NAME          TYPE    VALUE        AUTO
deployments/app2 config           true
deployments/app2 image  php-ssl:1 (php-ssl) true
```

3. A new image version, `registry.ocp4.example.com:8443/redhattraining/php-ssl:1-234`, is available in the container registry. Your QA team tested and approved that version. It is ready for production.

Create the `php-ssl:1-234` image stream tag that points to the new image. Move the `php-ssl:1` image stream tag alias to the new `php-ssl:1-234` image stream tag. Verify that the `app2` application redeploys.

Create the `php-ssl:1-234` image stream tag.

```
[student@workstation ~]$ oc create istag php-ssl:1-234 \
--from-image registry.ocp4.example.com:8443/redhattraining/php-ssl:1-234
imagestreamtag.image.openshift.io/php-ssl:1-234 created
```

Move the `php-ssl:1` alias to the new `php-ssl:1-234` image stream tag.

```
[student@workstation ~]$ oc tag --alias php-ssl:1-234 php-ssl:1
Tag php-ssl:1 set up to track php-ssl:1-234.
```

Verify that the `app2` application rolls out. The names of the replica sets on your system p

```
[student@workstation ~]$ oc describe deployment/app2
Name:                      app2
Namespace:                  updates-review
...output omitted...
Events:
  Type    ...  Age            Message
  ----  ...
  Normal  ...  33m           ...
  Normal  ...  3m30s         ...
  Normal  ...  3m28s         ...

  Normal  ...  33m           Scaled up replica set app2-7dd589f6d5 to 1
  Normal  ...  3m30s         Scaled up replica set app2-7bf5b7787 to 1
  Normal  ...  3m28s         Scaled down replica set app2-7dd589f6d5 to 0 from 1
```

## Evaluation

As the student user on the workstation machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[student@workstation ~]$ lab grade updates-review
```

## Finish

As the student user on the workstation machine, use the `lab` command to complete this exercise. This step is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish updates-review
```