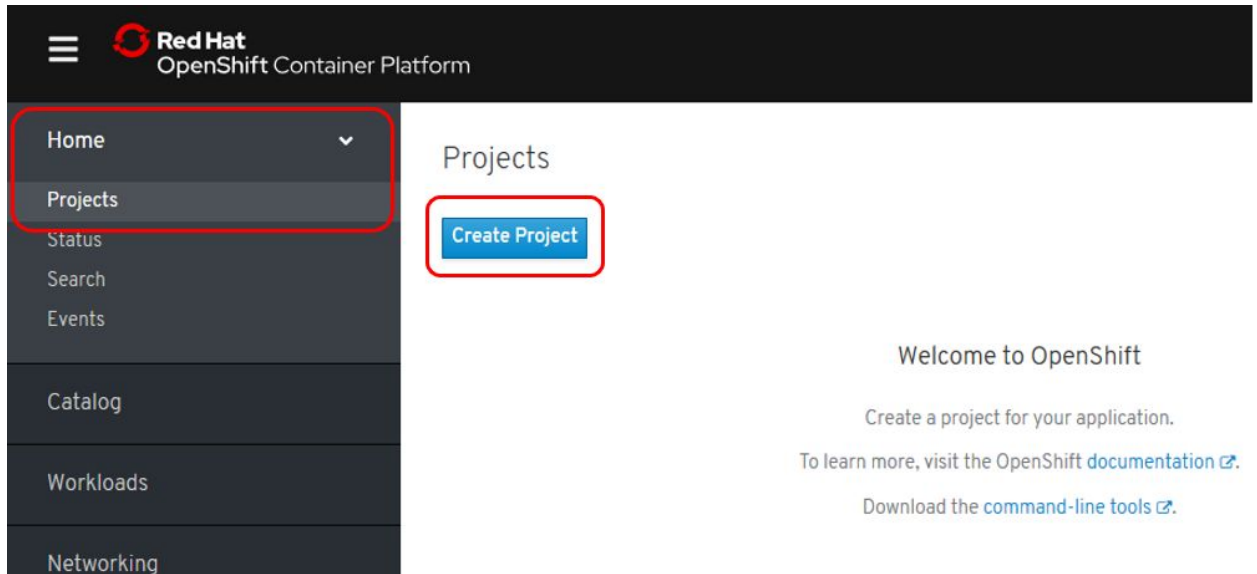


# Openshift 4 Workshop Jenkins Pipeline

## Deploy Jenkins Pipeline

In this workshop we will leverage the Openshift Console to create a new project, create a Jenkins deployment and then setup a build pipeline for a NodeJS application. Log into Openshift Console - the Instructor will provide the URL and your user. Login with user<assigned #> and password is **r3dh4t1!**

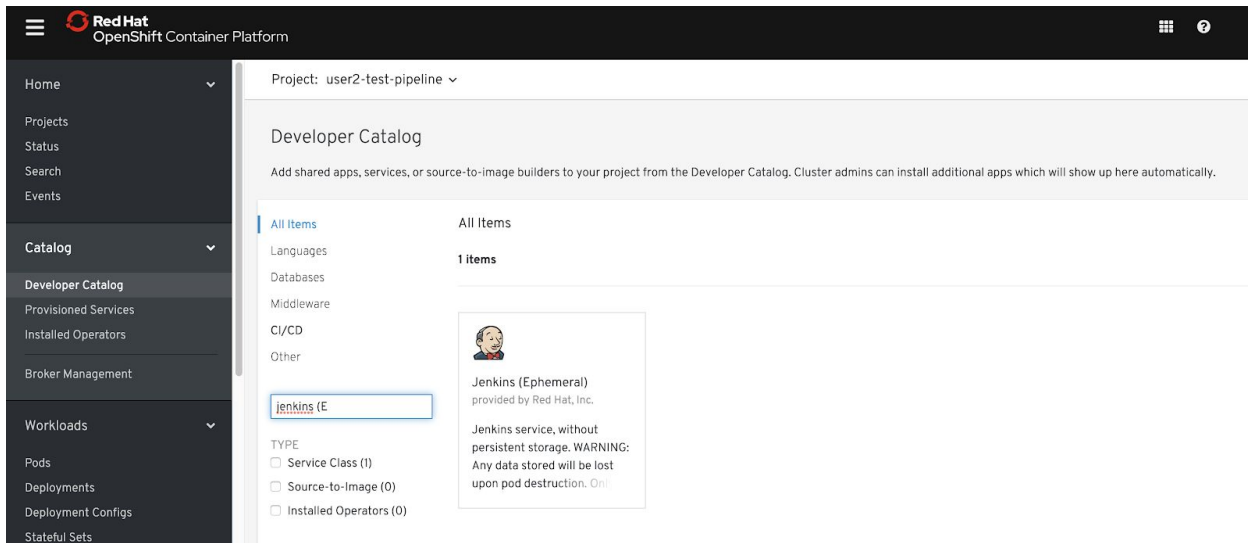
1. Click on the **Home->Projects** menu, then select the **Create Project** button.



2. Populate **Create Project** dialog with the following information - The project to create is user<assigned#>-test-pipeline

The screenshot shows the 'Create Project' dialog box. It has a title bar 'Create Project'. Below it, there are three input fields: 'Name \*' (containing 'user2-test-pipeline'), 'Display Name' (empty), and 'Description' (empty). At the bottom right, there are two buttons: 'Cancel' and 'Create'.

3. Now search for Jenkins in the catalog, click on the **Add->Browse Catalog** button. Select the Jenkins (Ephemeral).



4. Click on Jenkins and create the Service Instance.



# Jenkins (Ephemeral)

Provided by Red Hat, Inc.

## Create Service Instance

Jenkins service, without persistent storage.

### PROVIDER

Red Hat, Inc.

### SUPPORT

[Get Support](#)

### CREATED AT

Jul 23, 3:26 pm

WARNING: Any data stored will be lost upon pod destruction. Only use this template for testing.

This template deploys a Jenkins server capable of managing OpenShift Pipeline builds and supporting OpenShift-based oauth login. The Jenkins configuration is stored in non-persistent storage, so this configuration should be used for experimental purposes only.

### Documentation

[https://docs.okd.io/latest/using\\_images/other\\_images/jenkins.html](https://docs.okd.io/latest/using_images/other_images/jenkins.html)

### Service Plans

- Default plan

5. Create the Service Binding and accept the default, and click once again.

jenkins-ephemeral Actions

Overview YAML Events Service Bindings

### Create Service Binding

Service bindings create a secret containing the necessary information for a workload to use jenkins-ephemeral. Once the binding is ready, add the secret to your workload's environment variables or volumes.

Create Service Binding

Service Instance Overview

6. Wait until the Jenkins pod is up and Running. If the Container is Not Ready like this, then wait.

jenkins-1-svcv user2-test-pipeline deployment=jenkins-1 ip-10-0-142-221.ec2.internal Running ContainersNotReady

7. Now when We get this, We are ready to go.

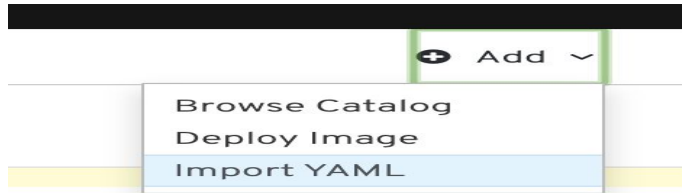
Pods

Create Pod Filter Pods by name...

1 Running 0 Pending 0 Terminating 0 CrashLoopBackOff 1 Completed 0 Failed 0 Unknown Select All Filters 1 of 2 Items

NAME	NAMESPACE	POD LABELS	NODE	STATUS	READINESS
jenkins-1-svcv	user2-test-pipeline	deployment=jenkins-1 deploymentco...=jenk... name=jenkins	ip-10-0-142-221.ec2.internal	Running	Ready

8. Select Add on the upper right and select the “import yaml”.



9. Copy the contents from <https://raw.githubusercontent.com/glamperi/OCP4WORKSHOP/master/nodejs-sample-pipeline.yaml> into the window. Make sure to change the namespace in the text to the namespace of the project. After the edit, click create.



10. You will see the pipeline build under builds now.

OpenShift Container Platform

Deployment Configs  
Stateful Sets  
Secrets  
Config Maps  
Cron Jobs  
Jobs  
Daemon Sets  
Replica Sets  
Replication Controllers  
Horizontal Pod Autoscalers  
Networking  
Storage  
Builds  
Build Configs  
Builds  
Image Streams

Project: user2-test-pipeline

### Builds

[Filter Builds](#)

0 New	0 Pending	1 Running	0 Complete	0 Failed	0 Error	0 Cancelled	Select All Filters
NAME ↑	NAMESPACE	STATUS	CREATED				
<a href="#">nodejs-pipeline-1</a>	NS user2-test-pipeline	Running	a minute ago				

11. Click into the nodejs-pipeline-1 and see the progress. Going to the Logs will bring you to the Jenkins Console, use the same login as to the Openshift Console. This may take about 10 minutes to build with the given CPU/Memory setup for this particular lab.

[nodejs-pipeline](#) > Build Details

[nodejs-pipeline-1](#) [Actions](#)

[Overview](#) [YAML](#) [Environment](#) [Logs](#) [Events](#)

### Build Overview

NAME	STATUS
<a href="#">Build 1</a> 4 minutes ago <a href="#">View Logs</a>	<div><div>preamble 3 minutes ago</div><div>cleanup 2 minutes ago</div><div>create 2 minutes ago</div><div>build 2 minutes ago</div></div>

12. Here is what the Jenkins Console will look like.

The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and user information (user6, log out). The breadcrumb trail indicates the current view is for the 'user2-test-pipeline/nodejs-pipeline' build #1. The left sidebar contains various controls like 'Back to Project', 'Status', 'Changes', 'Console Output' (selected), 'View as plain text', 'Edit Build Information', 'Open Blue Ocean', 'Thread Dump', 'Pause/resume', 'Replay', 'Pipeline Steps', and 'Workspaces'. The main area is titled 'Console Output' and displays the following log text:

```
OpenShift Build user2-test-pipeline/nodejs-pipeline-1
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Still waiting to schedule task
'nodejs-c0g1k' is offline
Agent nodejs-c0g1k is provisioned from template Kubernetes Pod Template
Agent specification [Kubernetes Pod Template] (nodejs):
* [jnl] image-registry.openshift-image-registry.svc:5000/openshift/jenkins-agent-nodejs:latest

Running on nodejs-c0g1k in /tmp/workspace/user2-test-pipeline/user2-test-pipeline-nodejs-pipeline
[Pipeline] {
[Pipeline] timeout
Timeout set to expire in 20 min
[Pipeline] {
[Pipeline] stage
[Pipeline] { (preamble)
[Pipeline] script
[Pipeline] {
[Pipeline] echo

[Pipeline] _OcContextInit
[Pipeline] _OcContextInit
[Pipeline] readFile
[Pipeline] echo
Using project: user2-test-pipeline
[Pipeline] }
```

13. Once the build is done and the backing mongodb is deployed as well. Find the NodeJS frontend route from Network->Routes.

## Routes

<a href="#">Create Route</a>					Filter Routes by name...
2 Accepted 0 Rejected 0 Pending Select All Filters					2 Items
NAME ↑	NAMESPACE	LOCATION	SERVICE	STATUS	
jenkins	user2-test-pipeline	<a href="https://jenkins-user2-test-pipeline.apps.cluster-vegas-c060.vegas-c060.openshiftworkshop.com">https://jenkins-user2-test-pipeline.apps.cluster-vegas-c060.vegas-c060.openshiftworkshop.com</a>	jenkins	Accepted	
nodejs-mongodb-example	user2-test-pipeline	<a href="http://nodejs-mongodb-example-user2-test-pipeline.apps.cluster-vegas-c060.vegas-c060.openshiftworkshop.com">http://nodejs-mongodb-example-user2-test-pipeline.apps.cluster-vegas-c060.vegas-c060.openshiftworkshop.com</a>	nodejs-mongodb-example	Accepted	

## 14. Click on the Route and you should see the following in the Web Browser.

### Welcome to your Node.js application on OpenShift

---

#### How to use this example application

For instructions on how to use this application with OpenShift, start by reading the [Developer Guide](#).

#### Deploying code changes

The source code for this application is available to be forked from the [OpenShift GitHub repository](#). You can configure a webhook in your repository to make OpenShift automatically start a build whenever you push your code:

1. From the Web Console homepage, navigate to your project
2. Click on Browse > Builds
3. Click the link with your BuildConfig name
4. Click the Configuration tab
5. Click the "Copy to clipboard" icon to the right of the "GitHub webhook URL" field
6. Navigate to your repository on GitHub and click on repository settings > webhooks > Add webhook
7. Paste your webhook URL provided by OpenShift in the "Payload URL" field
8. Change the "Content type" to 'application/json'
9. Leave the defaults for the remaining fields — that's it!

After you save your webhook, if you refresh your settings page you can see the status of the ping that Github sent to OpenShift to verify it can reach the server.

Note: adding a webhook requires your OpenShift server to be reachable from GitHub.

#### Working in your local Git repository

If you forked the application from the OpenShift GitHub example, you'll need to manually clone the repository to your local system. Copy the application's source code Git URL and then run:

```
$ git clone <git_url> <directory_to_create>

# Within your project directory
# Commit your changes and push to OpenShift

$ git commit -a -m 'Some commit message'
$ git push
```

After pushing changes, you'll need to manually trigger a build if you did not setup a webhook as described above.

#### Managing your application

Documentation on how to manage your application from the Web Console or Command Line is available at the [Developer Guide](#).

#### Web Console

You can use the Web Console to view the state of your application components and launch new builds.

#### Command Line

With the [OpenShift command line interface](#) (CLI), you can create applications and manage projects from a terminal.

#### Development Resources

- [OpenShift Documentation](#)
- [OpenShift Origin GitHub](#)
- [Source To Image GitHub](#)
- [Getting Started with Node.js on OpenShift](#)
- [Stack Overflow questions for OpenShift](#)
- [Git documentation](#)


#### Request information

Page view count: 49

#### DB Connection Info:

Type: MongoDB  
URL: mongodb://172.30.148.142:27017/sampledb

Built on

**OPENSIFT**  
by Red Hat