Joget on OpenShift

- Deploy Joget using the OpenShift Web Console
    - Step 1: Create OpenShift Project
    - Step 2: Deploy MySQL Database
    - Step 3: Deploy Joget Certified Container Image
    - Step 4: Configure Persistent Storage
    - Step 5: Configure for Clustering and Licensing
- Deploy Joget using the OpenShift CLI
- Setup Database

Red Hat OpenShift is a container application platform that brings Docker and Kubernetes to the enterprise. As a cloud-native computing platform, OpenShift allows teams to automate the build, deployment, and management of applications. They can focus on developing apps without worrying about the underlying infrastructure, whether on physical/virtual servers or on public/ private/hybrid cloud environments.

As an open source no-code/low-code platform to visually build enterprise web apps for coders and non-coders, Joget is an ideal complement to OpenShift to fill the gap. The Joget platform is now available as a Red Hat Certified Container, so it is trusted, secure and commercially supported on OpenShift. This article describes the steps in deploying the certified Joget container image running with the MySQL database.

blocked URL blocked URL

Deploy Joget using the OpenShift Web Console

Step 1: Create OpenShift Project

Access the OpenShift Web Console and login

blocked URL

Create a project using the **Create Project** button from the Projects dropdown and key in the desired **Name**, **Display Name** and **Description**.

blocked URL

Step 2: Deploy MySQL Database

Under the selected project, select +**Add > Database** and select **MySQL**, then **Instantiate Template**.

blocked URL

Key in the appropriate settings and click on **Create** e.g.

Namespace	openshift
Database Service Name	jogetdb
MySQL Connection Username	joget
MySQL Connection Password	joget
MySQL Database Name	jwdb

blocked URL

Step 3: Deploy Joget Certified Container Image

To access the Red Hat Container Catalog, a valid username and password that is used to log in to the **Red Hat Customer Portal** are required.

If you do not have an account, you can acquire one by registering for one of the following options:

- Red Hat Developer Program. This account gives you access to developer tools and programs.
- 30-day Trial Subscription. This account gives you a 30-day trial subscription with access to select Red Hat software products.

Under the selected project, select +**Add > Container Image** and click on **create an image pull secret link**. In the ensuing popup, key in the Red Hat login details for the registry.

Secret Name	registry.connect.redhat.com
Authentication Type	Image Registry Credentials
Image Registry Server Address	registry.connect.redhat.com
Username	Red Hat account username
Password	Red Hat account password
Email	Red Hat account email

blocked URL

Once the secret has been created, select the **Image Name** option and key in **registry.connect.redhat.com/joget/joget-dx8-eap7.** Key in the desired **Application Name** and **Name.**

Image Name	registry.connect.redhat.com/joget/joget-dx8-eap7
Name	joget-dx8-eap7

blocked URL

Note

Do ensure that the exposed service route is configured to:

Path /jw

Port 8080

Step 4: Configure Persistent Storage

The next step is to add persistent storage to the container for storing configuration files and persistent file uploads. Under **Topology**, select the Deployment. Select **Add Storage** under the **Actions** menu.

blocked URL

In the **Add Storage** page under **Persistent Volume Claim**, select **Create new claim** and fill in desired values then **Save**.

Name	joget-dx8-eap7-claim
Access Mode	Shared Access (RWX)
Size	10GB (or as required)
Mount Path	/home/jboss/wflow

blocked URL

Step 5: Configure for Clustering and Licensing

Using the OpenShift command line interface (CLI), run the following commands to enable clustering and licensing.

```
    export PROJECT_NAME=demo # modify this to suit your project name
export APP_NAME=joget-dx8-eap7 # modify this to suit your app name

echo === configure jboss clustering ===
oc set env deployment/${APP_NAME} JGROUPS_PING_PROTOCOL=openshift.DNS_PING -e
OPENSHIFT_DNS_PING_SERVICE_NAME=${APP_NAME}-ping -e OPENSHIFT_DNS_PING_SERVICE_PORT=8888 -e CACHE_NAME=http-
session-cache
oc expose deployment/${APP_NAME} --port=8888 --name=${APP_NAME}-ping --cluster-ip=None

echo === assign cluster role view permission for the project service account (to read deployment info for
licensing) ===
oc create clusterrolebinding default-view --clusterrole=view --serviceaccount=$PROJECT_NAME:default --
namespace=$PROJECT_NAME
```

Note: If you already have an existing cluster and have deployed images before, it is likely that "**clusterrolebinding**" name of "**default-view**" may already be used. In this case, simply give it another meaningful name (**e.g.**: default-view-new)

Once the pods in the deployment have finished starting up, access the **Deployment** under **Topology** to see the running pods. You will also see a **Route** created for it, so click on the **Location** URL to access Joget.

blocked URL

Deploy Joget using the OpenShift CLI

Alternatively, the following is a Linux script to accomplish a similar Joget platform deployment such as above using the OpenShift command line interface (CLI). Change the environment variables in the script accordingly, at least the four values below:

PROJECT_NAME	The desired project name
REGISTRY_USERNAME	Red Hat account username
REGISTRY_PASSWORD	Red Hat account password
REGISTRY_EMAIL	Red Hat account email

```
    #!/bin/sh
```

```
export PROJECT_NAME=joget-openshift
export REGISTRY_USERNAME=email@domain
export REGISTRY_PASSWORD=password
export REGISTRY_EMAIL=email@domain
export REGISTRY_SERVER=registry.connect.redhat.com
export IMAGE_NAMESPACE=joget
export IMAGE_NAME=joget-dx8-eap7
export IMAGE_TAG=latest
export APP_NAME=joget-dx8-eap7
export DB_APP_NAME=jogetdb
export STORAGE_NAME=joget-data
export MYSQL_DATABASE=jwdb
export MYSQL_USER=joget
export MYSQL_PASSWORD=joget

echo === deploy Joget on OpenShift ===
echo PROJECT_NAME: $PROJECT_NAME
echo REGISTRY_SERVER: $REGISTRY_SERVER
echo REGISTRY_USERNAME: $REGISTRY_USERNAME
echo REGISTRY_EMAIL: $REGISTRY_EMAIL
echo IMAGE_NAMESPACE: $IMAGE_NAMESPACE
echo IMAGE_NAME: $IMAGE_NAME
echo IMAGE_TAG $IMAGE_TAG
echo IMAGE_NAME: $IMAGE_NAME
echo APP_NAME: $APP_NAME
echo DB_APP_NAME: $DB_APP_NAME
echo STORAGE_NAME: $STORAGE_NAME
echo MYSQL_DATABASE: $MYSQL_DATABASE
echo MYSQL_USER: $MYSQL_USER
echo MYSQL_PASSWORD: $MYSQL_PASSWORD

echo === create project ===
oc new-project $PROJECT_NAME

echo === deploy MySQL ===
oc new-app openshift/mysql:8.0 --name $DB_APP_NAME -e MYSQL_USER=$MYSQL_USER -e MYSQL_PASSWORD=$MYSQL_PASSWORD -
e MYSQL_DATABASE=$MYSQL_DATABASE

echo === create and bind secret to pull Joget image ===
oc create secret docker-registry $REGISTRY_SERVER --docker-server=$REGISTRY_SERVER --docker-
username=$REGISTRY_USERNAME --docker-password=$REGISTRY_PASSWORD --docker-email=$REGISTRY_EMAIL
oc secrets link default $REGISTRY_SERVER --for=pull

echo === assign cluster role view permission for the project service account to read deployment info for
licensing ===
oc create clusterrolebinding default-view --clusterrole=view --serviceaccount=$PROJECT_NAME:default --
namespace=$PROJECT_NAME

echo === create joget deployment, service and persistent volume claim ===
cat <<EOF > joget.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: $APP_NAME-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
---
apiVersion: v1
kind: Service
metadata:
  name: $APP_NAME
  labels:
    app: $APP_NAME
spec:
  ports:
  - port: 8080
  selector:
```

```
    app: $APP_NAME
  type: NodePort
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: $APP_NAME
spec:
  selector:
    matchLabels:
      app: $APP_NAME
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: $APP_NAME
    spec:
      containers:
      - image: registry.connect.redhat.com/joget/joget-dx8-eap7:latest
        name: $APP_NAME
        env:
        - name: JGROUPS_PING_PROTOCOL
          value: "openshift.DNS_PING"
        - name: OPENSHIFT_DNS_PING_SERVICE_NAME
          value: "$APP_NAME-ping"
        - name: OPENSHIFT_DNS_PING_SERVICE_PORT
          value: "8888"
        - name: CACHE_NAME
          value: "http-session-cache"
        ports:
        - containerPort: 8080
          name: $APP_NAME
        volumeMounts:
        - name: $APP_NAME-persistent-storage
          mountPath: /home/jboss/wflow
      volumes:
      - name: $APP_NAME-persistent-storage
        persistentVolumeClaim:
          claimName: $APP_NAME-pvc
---
kind: Service
apiVersion: v1
metadata:
  name: $APP_NAME-ping
  labels:
    app: $APP_NAME
spec:
  clusterIP: None
  ports:
    - name: $APP_NAME-ping
      port: 8888
  selector:
    app: $APP_NAME
---
kind: Route
apiVersion: route.openshift.io/v1
metadata:
  name: $APP_NAME
  labels:
    app: $APP_NAME
  annotations:
    haproxy.router.openshift.io/timeout: 600s
    openshift.io/host.generated: 'true'
spec:
  path: /jw
  to:
    kind: Service
    name: $APP_NAME
    weight: 100
  port:
```

```
    targetPort: 8080
  wildcardPolicy: None
EOF
oc apply -f joget.yaml
```

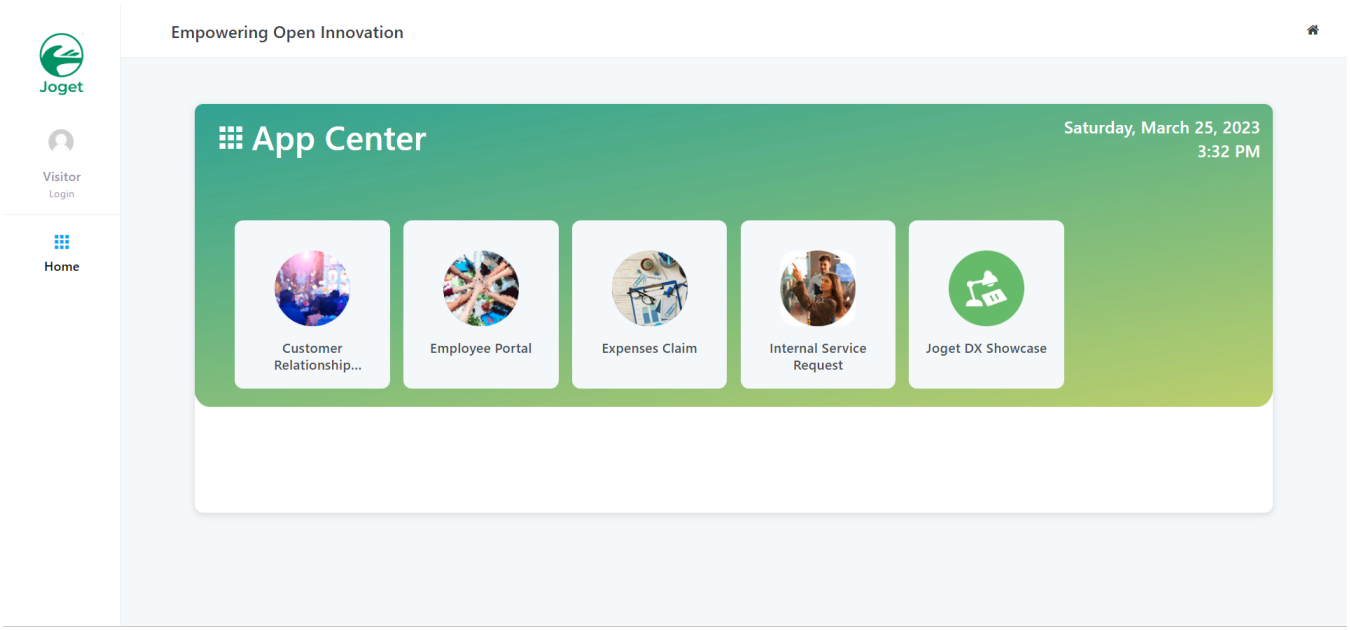
Setup Database

The first time the Joget platform is accessed, the Database Setup page will be displayed. Configure the database settings using the values defined when deploying the MySQL database previously e.g.

Database Host	jogetdb
Database Port	3306
Database Name	jwdb
Database User	joget
Database Password	joget

blocked URL

Upon successful configuration, the Joget App Center will be loaded.



NOTE: If you encounter a **504 Gateway Timeout** during the database setup, it is caused by the database initialization taking longer than the default OpenShift Route timeout. You can actually ignore the error and wait a couple of minutes before accessing the Application URL from the **Overview** page again.

You can also increase the route timeout using the OpenShift CLI i.e.

oc annotate route $APP_NAME --overwrite haproxy.router.openshift.io/timeout=60s