NewgenONE OmniDocs RMS

Configuration and Deployment Guide for OpenShift

Version: 4.0 SP1

Newgen Software Technologies Ltd.

Table of Contents

1 Preface

This guide describes the deployment and configuration of OmniDocs suite (Record Management System (RMS) 4.0 and OmniDocs 11.0 SP2) deliverables like OmniDocs Container images and their required configuration files on the OpenShift Container Platform.

1.1 Revision history

Revision Date	Description
April 2024	Initial publication

1.2 Intended audience

This guide is intended for System Administrators, developers, and all other users who are seeking information on the deployment of OmniDocs and RMS 4.0 containers on OpenShift Container Platform. The reader must be comfortable in understanding the computer terminology.

1.3 Documentation feedback

To provide feedback or any improvement suggestions on technical documentation, write an email to docs.feedback@newgensoft.com.

To help capture your feedback effectively, share the following information in your email.

- Document Name
- Version
- Chapter, Topic, or Section
- Feedback or Suggestions

1.4 Third-party information

This guide contains third-party product information about configuring OpenShift Container Platform and Jenkins CICD Pipeline for Container Deployment on OpenShift. Newgen Software Technologies Ltd does not claim any ownership of such third-party content. This information is shared in this guide only for the convenience of our users and could be an excerpt from the OpenShift documentation. For the latest information on configuring the OpenShift Container Platform and Jenkins CICD Pipeline refer to the respective official documentation.

2 Deploying the OmniDocs and RMS Containers

This section describes the prerequisites, deliverables and other changes required to deploy the OmniDocs and RMS Containers.

2.1 Prerequisites

Following are the prerequisites:

- OpenShift Container Platform is already configured, and its Worker nodes are in Ready state.
- RedHat Quay Registry is already configured to store container images.
- NAS server is already configured, and a mount path is already created.

2.2 Deliverables

Newgen has isolated the product suite into multiple Container containers to enable the independent scalability of each container image. This separation is done based on the product's usability. At a broad level, Web components and EJB components are isolated for deployment in separate container instances. Web components get deployed on the underlying web server JBoss WebServer 5.7.x. EJB components gets deployed on the underlying application server *JBoss EAP 7.4.x*. Newgen releases multiple Container images for the different product suites along with some configuration files for data persistence, YAML files for deployment, and some documentation for end-to-end configurations and deployments.

Newgen product team delivers the following:

- Docker images
- Configuration files
- YAML files

2.2.1 Docker images

The following Docker images are delivered for the initial product deployment:

- OmniDocs+RMS Web Components
- OmniDocs Web Service Components
- OmniDocs+RMS EJB Components
- OmniDocs Add-on Services (Wrapper, AlarmMailer, Scheduler, ThumbnailManager and LDAP)
- EasySearch (Apache Manifold only)
- Text Extraction Manager or Full-Text Search (TEM/FTS)
- OmniScan Web Components

- RMS SharePoint Adapter
- WOPI
- OmniDocs SMS (Storage Management System) Service

OmniDocs SMS service is required to store the PN files. PN files are encrypted files that contain all the added/uploaded/scanned documents by Newgen products.

NOTE:

These Container images can be delivered to a private Container repository like Azure ACR (Azure Container Registry), AWS ECR (Elastic Container Registry), RedHat Quay Registry or in the form of compressed files that can be shared over the FTP or similar kind of media.

2.2.2 Configuration files

Configuration files are dynamic in nature and data is written at runtime. Database details in configuration files such as *Server.xml* and *standalone.xml* are written at runtime. These types of files must be kept outside the container to persist the data. Here NAS (Network-attached Storage) server will be used to persists the configuration files.

The following configuration files are shared for OmniDocs and RMS Docker images:

- OmniDocs11.0Web
- OmniDocs11.0EJB
- OD11.0Services
- EasySearch11.0
- TEM11.0
- OmniScanWeb6.0
- OD11.0SMS
- OmniDocs11.0Wopi
- RMSSharePointAdapter

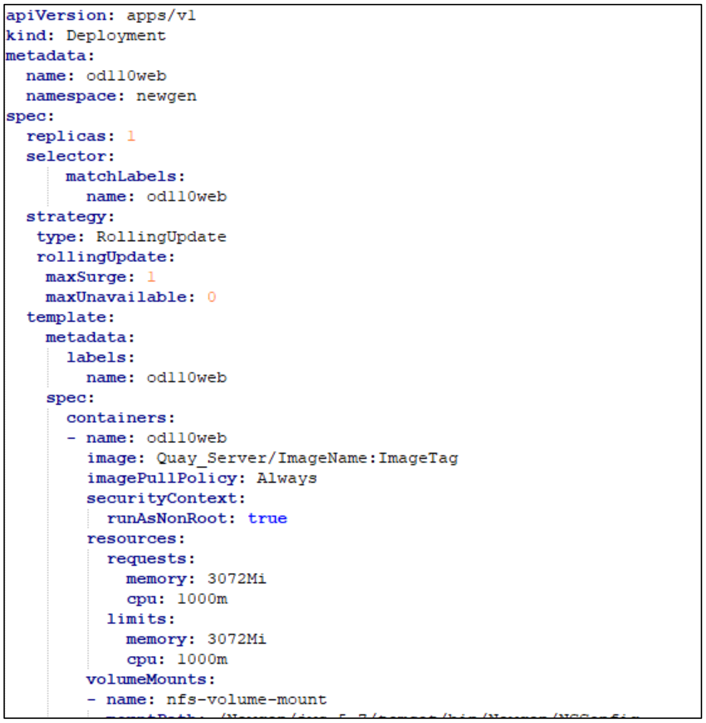
2.2.3 YAML files

YAML files stands for “YAML Ain’t Markup Language”. It is a human-readable object configuration file that is used to deploy and manage the objects on the OpenShift cluster. In other words, it is a manifest file that contains the deployment descriptor of OpenShift containers. YAML files can be executed using *oc apply –f <YAMLFile>* or we can use these files in Jenkins Pipeline to deploy the containers.

The following configuration files are shared for OmniDocs and RMS Container images:

- OmniDocs11.0Web.yml
- OmniDocs11.0Web_Services.yml
- OmniDocs11.0EJB.yml
- OmniDocs11.0Services.yml
- EasySearch11.0_ApacheOnly.yml
- TEM11.0.yml
- OmniScanWeb6.0.yml
- OmniDocs11.0SMS.yml
- OmniDocs11.0Wopi.yml
- RMSSharePointAdapter.yml
- quay-secret.yml
- Storage_PV_PVC.yml
- IngressController.yml

Here’s an example of a YAML file:



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: od110web
  namespace: newgen
spec:
  replicas: 1
  selector:
      matchLabels:
        name: od110web
  strategy:
   type: RollingUpdate
   rollingUpdate:
    maxSurge: 1
    maxUnavailable: 0
  template:
    metadata:
      labels:
        name: od110web
    spec:
      containers:
      - name: od110web
        image: Quay_Server/ImageName:ImageTag
        imagePullPolicy: Always
        securityContext:
          runAsNonRoot: true
        resources:
          requests:
            memory: 3072Mi
            cpu: 1000m
          limits:
            memory: 3072Mi
            cpu: 1000m
        volumeMounts:
        - name: nfs-volume-mount
          mountPath: /Newgen/jws-5.7/tomcat/bin/Newgen/NGConfig
```

Figure 2.1

Quay-secret.yml file is used to create image pull secret from RedHat Quay Registry.

Storage_PV_PVC.yml file is used for Persistent Volume and Persistent Volume Claim.

Persistent Volume (PV) is a piece of storage in the cluster that has been provisioned using Storage Classes. It contains the **NFS** Server details along with **mount path** that will be used to store product’s configuration files.

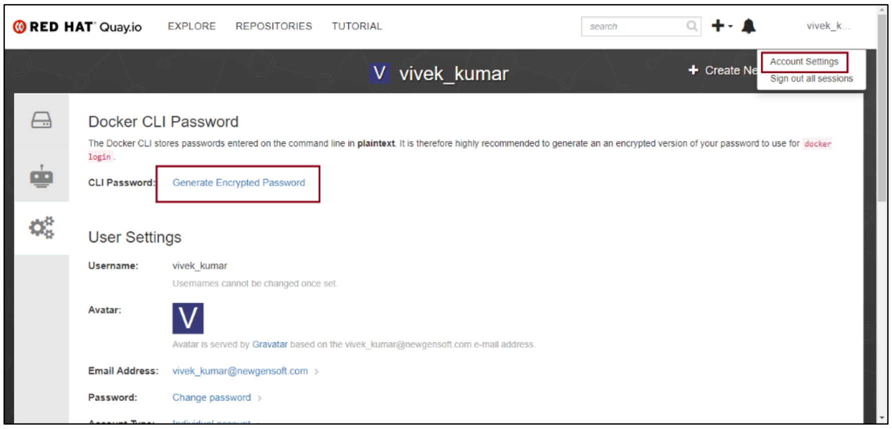
A PersistentVolumeClaim (PVC) is a request for storage by a user. It is similar to a Pod. Pods consume node resources and PVCs consume PV resources. Pods can request specific levels of resources (CPU and Memory). Claims can request specific size and access modes (For example, it can be mounted ReadWriteOnce, ReadOnlyMany or ReadWriteMany).

IngressController.yaml is used for the ingress controller. An ingress controller is an object running inside the OpenShift cluster that is used to manage the host-based routing rules. For example, set the host-based routing rules like if the URL is OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net then the ingress controller redirects the user request to OmniDocsWEB containers.

2.3 Changes in Quay-secret.yml file

RedHat Quay Registry can be used to store product's container images.

- To get the Kubernetes pull secret YAML file, login to the https://quay.io/ , go to the 'Account Settings' then click on 'Generate Encrypted Password'.
  For example,



Docker CLI Password

The Docker CLI stores passwords entered on the command line in **plaintext**. It is therefore highly recommended to generate an an encrypted version of your password to use for `docker login`.

CLI Password: Generate Encrypted Password

User Settings

Username: vivek_kumar
Usernames cannot be changed once set.

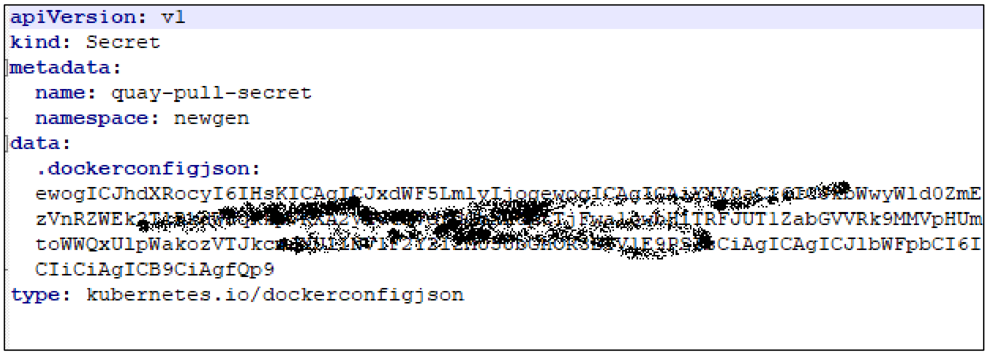
Avatar: V
Avatar is served by Gravatar based on the vivek_kumar@newgensoft.com e-mail address.

Email Address: vivek_kumar@newgensoft.com

Password: Change password

Figure 2.2

- Here, Kubernetes pull secret file can be downloaded.
- Rename the downloaded file as Quay-secret.yml so that the same name can be used in Jenkins pipeline.
- Open this file in edit mode and add **namespace** spec under the **metadata**. For example,



```
apiVersion: v1
kind: Secret
metadata:
  name: quay-pull-secret
  namespace: newgen
data:
  .dockerconfigjson:
  ewogICJhdXRocyI6IHsKICAgICJxdWF5LmlyIjogewogICAg[redacted]WwyWld0ZmE
  zVnRZWEk[redacted]TRFJUT1ZabGVVRk9MMVpHUm
  toWWQxUlpWakozVTJkc[redacted]CiAgICAgICJ1bWFpbCI6I
  CIiCiAgICB9CiAgfQp9
type: kubernetes.io/dockerconfigjson
```

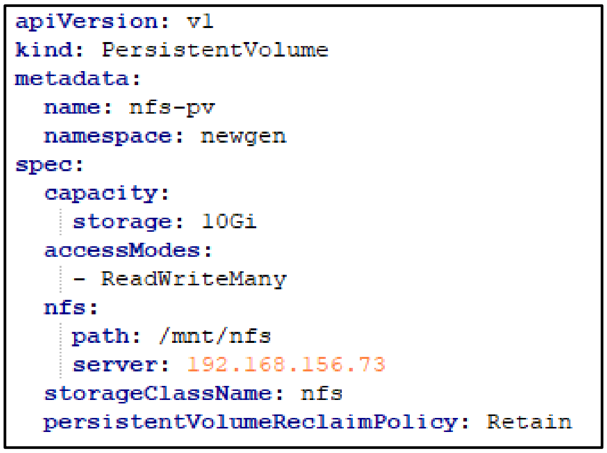
Figure 2.3

2.4 Storage_PV_PVC.yml file changes

Container-based applications often need to access and persist data in an external data volume. All the files created inside a container are stored on a writable container layer. This means that the data doesn't persist when that container no longer exists, and it can be difficult to get the data out of the container if another process needs it. Thus, NAS server can be used to persist these types of

data. NAS server persists the Newgen Product’s configuration files as well as Newgen Product’s logs.
This YAML file is used to create a persistent volume and a persistence volume claim with NAS server in OpenShift Container Platform.

Persistent Volume: Persistent Volume is a piece of storage in the cluster that has been provisioned using Storage Classes. It contains the NAS server details like NAS server IP and mount path that is already created. For example,

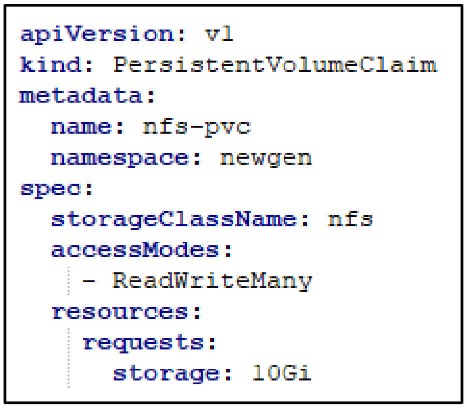


```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-pv
  namespace: newgen
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  nfs:
    path: /mnt/nfs
    server: 192.168.156.73
  storageClassName: nfs
  persistentVolumeReclaimPolicy: Retain
```

Figure 2.4

Here,
newgen: is the name of OpenShift project or namespace in which persistence volume will be created.
192.168.156.73: is the IP address of NAS server.
/mnt/nfs: is the mount path created to NAS server.

Persistent Volume Claim: A PersistentVolumeClaim (PVC) is a request for storage by a user. It is similar to a Pod. Pods consume node resources and PVCs consume PV resources. Pods can request specific levels of resources (CPU and Memory). Claims can request specific size and access modes means they can be mounted ReadWriteOnce, ReadOnlyMany or ReadWriteMany).



```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfs-pvc
  namespace: newgen
spec:
  storageClassName: nfs
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
```

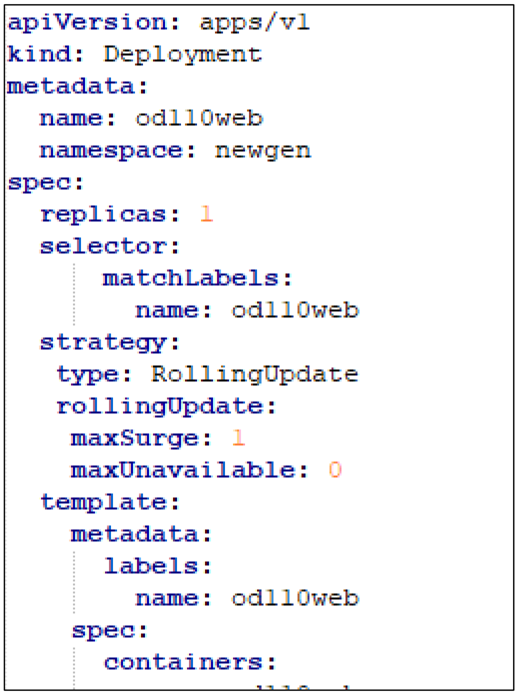
Figure 2.5

Here, **newgen**: is the name of OpenShift project or namespace in which persistence volume claim will be created.

2.5 Product's YAML files changes

The changes in the Product’s YAML files are as follows:

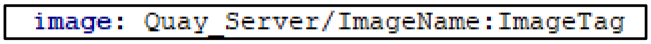
- **Name:** In the *OmniDocs11.0Web.yml* file, **od110web** is given as the default name of Kubernetes objects - deployment, replica-set, container, and service. You can change this name as per your requirement. While changing the name, ensure that this name is not more than 13 letters in length and must contain small letters only.
- **Namespace**: In the YAML files, default namespace is given as **newgen.** You can change this name as per your requirement.



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: od110web
  namespace: newgen
spec:
  replicas: 1
  selector:
      matchLabels:
        name: od110web
  strategy:
   type: RollingUpdate
   rollingUpdate:
    maxSurge: 1
    maxUnavailable: 0
  template:
    metadata:
      labels:
        name: od110web
    spec:
      containers:
```

Figure 2.6

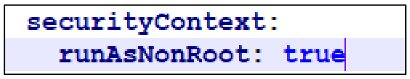
- **Replica:** In the *OmniDocs11.0Web.yml* file, the default replica is given as **1**. Thus, one container gets created after the deployment. You can increase this number as per your requirement.
- **Image:** In the *OmniDocs11.0Web.yml* file, update the **image** location. By default, the below value is given:



```
  image: Quay_Server/ImageName:ImageTag
```

Here, Quay_Server, ImageName, and ImageTag values will be updated at runtime by Jenkins Pipeline during deployment.

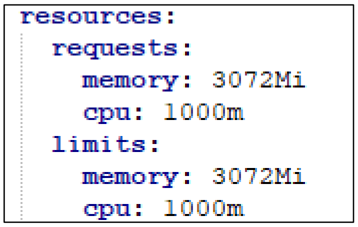
- **SecurityContext:** In the *OmniDocs11.0Web.yml* file, **SecurityContext [runAsNonRoot: true]** is defined. It means the OmniDocs WEB container can never be run with root privileges. If any container tries to run with the root user, then Kubernetes stops its deployments.



```
  securityContext:
    runAsNonRoot: true
```

- **Resource request and limit:** In the *OmniDocs11.0Web.yml* file, resource request and resource limit parameters are defined. The **request** parameter specifies the minimum required resources to run the container and the **limit** parameter specifies the maximum resource limit that a container can use. In other words, a running container is not allowed to use more than the resource limit you set.

Here, 1000m CPU = 1 Core CPU

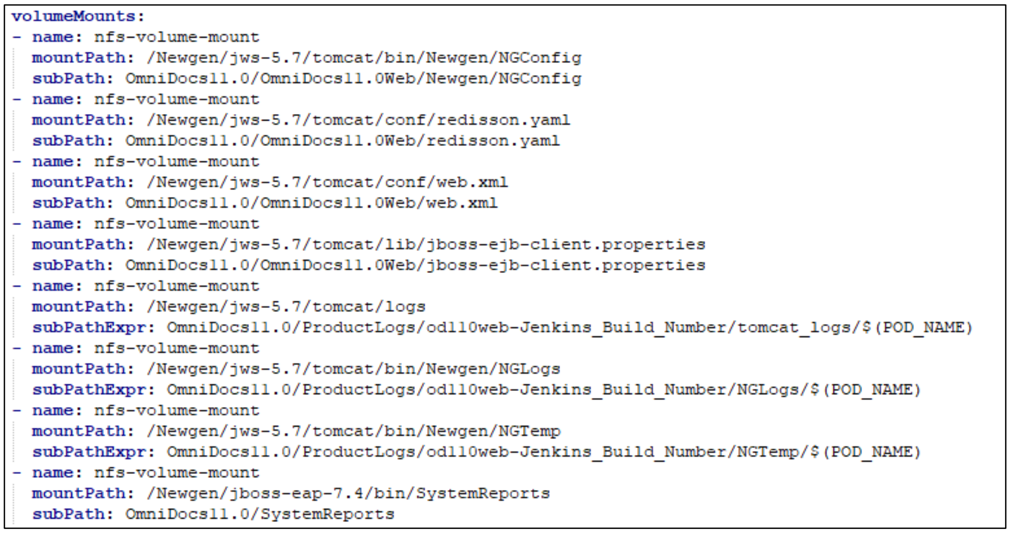


```
resources:
  requests:
    memory: 3072Mi
    cpu: 1000m
  limits:
    memory: 3072Mi
    cpu: 1000m
```

Figure 2.7

The above-specified limit is the minimum required resource to run a container. If users are increasing, then you must increase the limit range accordingly.

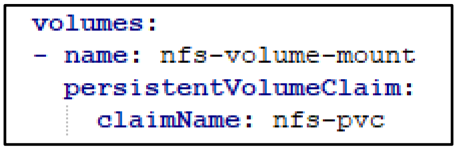
- **VolumeMounts and Volume:** Volume mounts and volumes are used to persist the data outside the container so that whenever the container terminates due to any reason our data is always persisted. In the *OmniDocs11.0Web.yml* file, there's a persisted configuration files or folders and log files.



```
volumeMounts:
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/bin/Newgen/NGConfig
  subPath: OmniDocs11.0/OmniDocs11.0Web/Newgen/NGConfig
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/conf/redisson.yaml
  subPath: OmniDocs11.0/OmniDocs11.0Web/redisson.yaml
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/conf/web.xml
  subPath: OmniDocs11.0/OmniDocs11.0Web/web.xml
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/lib/jboss-ejb-client.properties
  subPath: OmniDocs11.0/OmniDocs11.0Web/jboss-ejb-client.properties
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/logs
  subPathExpr: OmniDocs11.0/ProductLogs/od110web-Jenkins_Build_Number/tomcat_logs/$(POD_NAME)
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/bin/Newgen/NGLogs
  subPathExpr: OmniDocs11.0/ProductLogs/od110web-Jenkins_Build_Number/NGLogs/$(POD_NAME)
- name: nfs-volume-mount
  mountPath: /Newgen/jws-5.7/tomcat/bin/Newgen/NGTemp
  subPathExpr: OmniDocs11.0/ProductLogs/od110web-Jenkins_Build_Number/NGTemp/$(POD_NAME)
- name: nfs-volume-mount
  mountPath: /Newgen/jboss-eap-7.4/bin/SystemReports
  subPath: OmniDocs11.0/SystemReports
```

Figure 2.8

In volumeMounts, **mountPath** is a path inside the container that is being mounted. Here, mountPath cannot be changed as this structure is predefined in a container image. **subPath** works as a relative path that is appended to the attached persistent volume's mount path. **subPathExpr** is used to segregate the product logs container wise. And the **name** is a user-defined name that must be matched with the name specified in volumes.

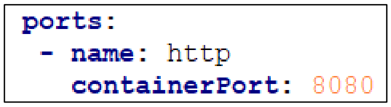


```
volumes:
- name: nfs-volume-mount
  persistentVolumeClaim:
    claimName: nfs-pvc
```

Figure 2.9

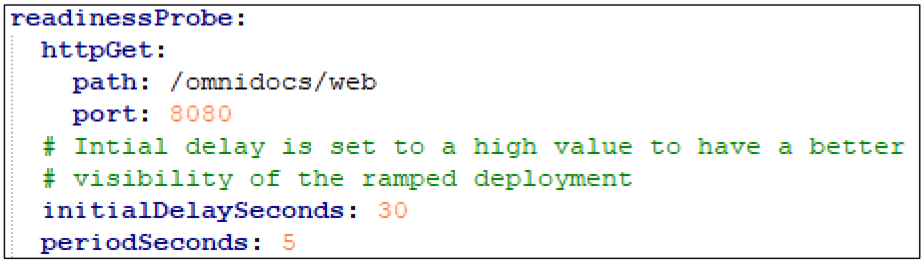
In volumes, **nfs-pvc** is the persistent volume claim name.

- **Ports:** In the *OmniDocs11.0Web.yml* file, containerPort **8080** is specified that means only 8080 port gets exposed outside the container and no other port is accessible from outside.



```
ports:
  - name: http
    containerPort: 8080
```

- **ReadinessProbe:** The kubelet used the readiness probe to know when a container is ready to start accepting traffic. Until unless the readiness probe is not succeeded the container does not serve the user requests.

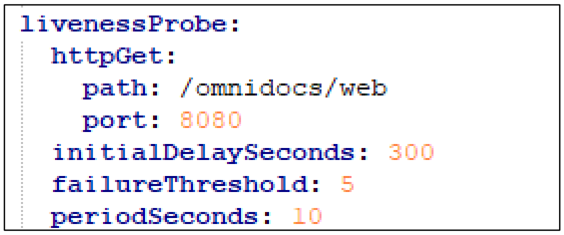


```
readinessProbe:
  httpGet:
    path: /omnidocs/web
    port: 8080
  # Intial delay is set to a high value to have a better
  # visibility of the ramped deployment
  initialDelaySeconds: 30
  periodSeconds: 5
```

Figure 2.10

Here, until unless ip:port/OmniDocs/web is not accessible, the container not accepts the user request.

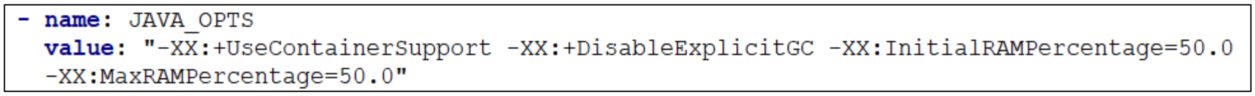
- **LivenessProbe**: Docker containers have healing power, if an application running inside the container gets down due to any reason or becomes unresponsive then Kubernetes auto-restarts that application inside the container. This feature is known as **LivenessProbe** in Kubernetes.



```
livenessProbe:
  httpGet:
    path: /omnidocs/web
    port: 8080
  initialDelaySeconds: 300
  failureThreshold: 5
  periodSeconds: 10
```

Figure 2.11

- **Environment variable:** In the *OmniDocs11.0Web.yml* file, the **JAVA_OPTS** parameter is defined that is used to set the heap size in the WEB container dynamically.

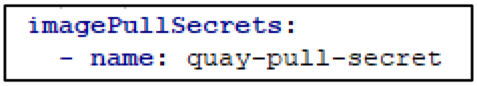


```
- name: JAVA_OPTS
  value: "-XX:+UseContainerSupport -XX:+DisableExplicitGC -XX:InitialRAMPercentage=50.0
  -XX:MaxRAMPercentage=50.0"
```

NOTE:

XX:MaxRAMPercentage is a parameter that tells the JVM how much available memory to use as a max heap size. In the above example, 50% of total memory is allocated as heap size. You can use the above guidelines to update other YAML files.

- **ImagePullSecret:** ImagePullSecret is a secret value that is used to pull an image from a private container repository like RedHat Quay Registry.



```
imagePullSecrets:
  - name: quay-pull-secret
```

This imagePullSecrets will be created using *Quay-secret.yml* file**.**

NOTE:

Follow the above steps to update the below YAML files:

- OmniDocs11.0Web_Services.yml
- OmniDocs11.0EJB.yml
- OmniDocs11.0Services.yml
- EasySearch11.0_ApacheOnly.yml
- TEM11.0.yml
- OmniScanWeb6.0.yml
- OmniDocs11.0SMS.yml
- RMSSharePointAdapter.yml
- OmniDocs11.0Wopi.yml

2.6 IngressController.yml file changes

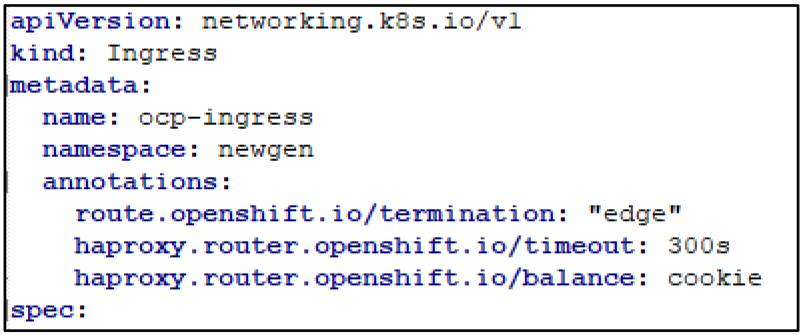
The changes are as follows:

- Along with the product's YAML file, Ingress Controller's YAML file **IngressController.yml** will also be shared. Using an ingress controller and ingress rules, a single IP address can be used to route traffic to multiple services in an OpenShift cluster. The Ingress Controller creates a separate OpenShift route for each ingress rule which routes the incoming requests to the target OpenShift services according to the host-based routing rules.

For example, set the rules as below:

- ➢ If URL is *OmniDocs.newgendocker.com,* then redirect to the OmniDocs+RMS Web container.
- ➢ If URL is *omniscan.newgendocker.com,* then redirect to the OmniScanWeb container.

- In the **IngressController.yml** file, default namespace is given as **newgen.** You can change this name as per your requirement. For example,



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ocp-ingress
  namespace: newgen
  annotations:
    route.openshift.io/termination: "edge"
    haproxy.router.openshift.io/timeout: 300s
    haproxy.router.openshift.io/balance: cookie
spec:
```

Figure 2.12

- In the *IngressController.yml* file, there are multiple host-based rules defined. A few of them are listed below:

➢ **OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net**

If the host URL is *'OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net'*, then it redirects the user request to the **od110web** container's service which is running on port 8080. Here, OmniDocs web is the name of the OmniDocs+RMS Web container and *newgenopenshiftcluster* is name of OpenShift cluster.

➢ **OmniDocswebservices.apps.newgenopenshiftcluster.newgensoftware.net**

If the host URL is *OmniDocswebservices*.apps.newgenopenshiftcluster.newgensoftware.net, then it redirects the user request to the **od110websvc** container's service which is running on port 8080. Here, od110websvc is the name of the OmniDocs Web Service container and *newgenopenshiftcluster* is name of OpenShift cluster.

➢ **OmniDocsconsole.apps.newgenopenshiftcluster.newgensoftware.net**

If the host URL is *OmniDocsconsole*.apps.newgenopenshiftcluster.newgensoftware.net, then it redirects the user request to the **od110ejb** container's service which is running on port 9990. Here, od110ejb is the name of the OmniDocs+RMS EJB container and *newgenopenshiftcluster* is name of OpenShift cluster.

➢ **apachemanifold.apps.newgenopenshiftcluster.newgensoftware.net**

If the host URL is *apachemanifold*.apps.newgenopenshiftcluster.newgensoftware.net, then it redirects the user request to the **easysearch11** container's service which is running on port 8345. Here, easysearch11 is the name of the EasySearch container and *newgenopenshiftcluster* is name of OpenShift cluster.

- ➢ **omniscan.apps.newgenopenshiftcluster.newgensoftware.net**

  If the host URL is *omniscan*.apps.newgenopenshiftcluster.newgensoftware.net, then it redirects the user request to the **omniscanweb** container's service which is running on port 8080. Here, omniscanweb is the name of the omniscanweb container and *newgenopenshiftcluster* is name of OpenShift cluster.

- In this YAML file, change the host URL, ServiceName, ServicePort, and the name "**name: appgw-ingress**" as per our choice.
- After making the required changes, deploy the Ingress controller by executing this YAML file using the below command:

```
kubectl apply -f IngressController.yml
```

2.7 Configuration files changes

The section contains the prerequisites and other changes required in configuration files.

2.7.1 Prerequisites

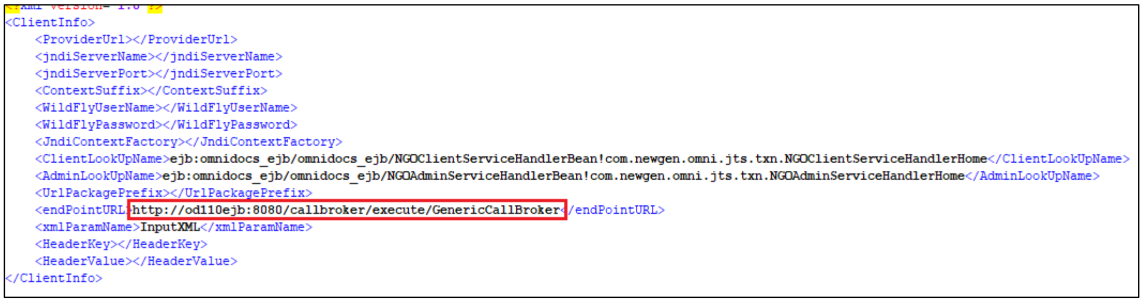
The prerequisites are as follows:

- All the configuration files and folders must be uploaded to the NAS server at mount path as defined in the YAML file *Storage_PV_PVC.yml.*
- The Redis Cache server must be configured.
- A valid wildcard certificate and the domain are already configured.
- SSL/TLS must be configured with the Ingress Controller or Load Balancer.

2.7.2 OmniDocs+RMS Web changes

The changes in OmniDocsWeb are as follows:

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *NGOClientData.xml, RMAppConfig.xml and RMClientData.xml* files in between the *<endPointURL></endPointURL>* tags located inside the *OmniDocs11.0Web\Newgen\NGConfig\ngdbini* folder kept inside the mount path on NAS server.

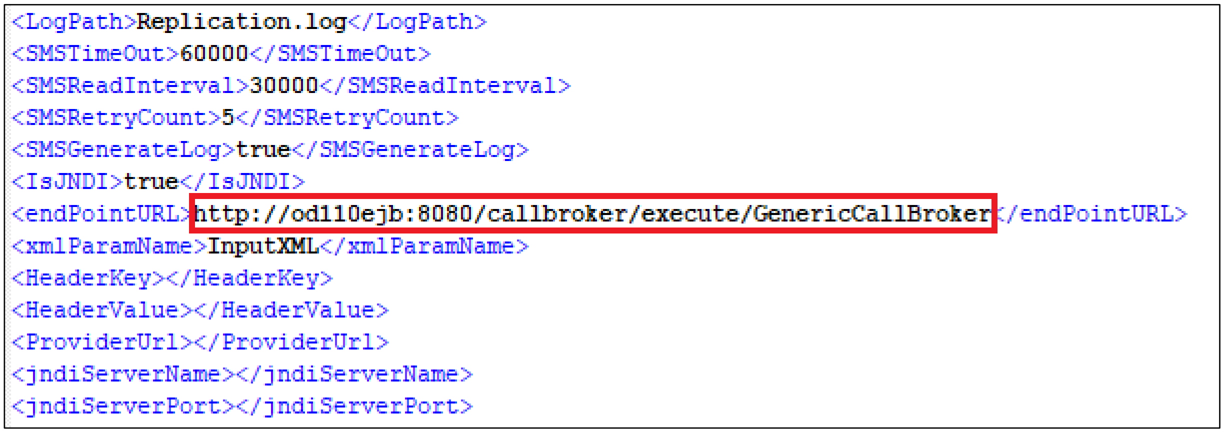


```
<?xml version="1.0" ?>
<ClientInfo>
    <ProviderUrl></ProviderUrl>
    <jndiServerName></jndiServerName>
    <jndiServerPort></jndiServerPort>
    <ContextSuffix></ContextSuffix>
    <WildFlyUserName></WildFlyUserName>
    <WildFlyPassword></WildFlyPassword>
    <JndiContextFactory></JndiContextFactory>
    <ClientLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOClientServiceHandlerBean!com.newgen.omni.jts.txn.NGOClientServiceHandlerHome</ClientLookUpName>
    <AdminLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOAdminServiceHandlerBean!com.newgen.omni.jts.txn.NGOAdminServiceHandlerHome</AdminLookUpName>
    <UrlPackagePrefix></UrlPackagePrefix>
    <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
    <xmlParamName>InputXML</xmlParamName>
    <HeaderKey></HeaderKey>
    <HeaderValue></HeaderValue>
</ClientInfo>
```

Figure 2.13

Here, **od110ejb** is the name of the OmniDocs+RMS EJB container.

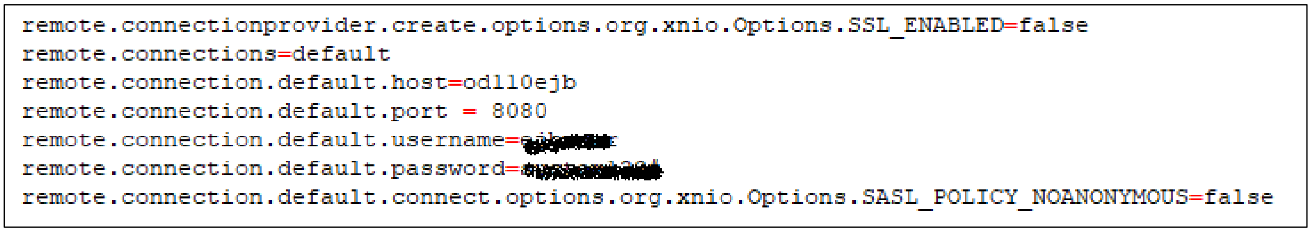
- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *IS.ini* file in between the *<endPointURL ></endPointURL >* tags located inside the *OmniDocs11.0Web\Newgen\NGConfig* folder kept inside the mount path on NAS server.
  For example,



```
<LogPath>Replication.log</LogPath>
<SMSTimeOut>60000</SMSTimeOut>
<SMSReadInterval>30000</SMSReadInterval>
<SMSRetryCount>5</SMSRetryCount>
<SMSGenerateLog>true</SMSGenerateLog>
<IsJNDI>true</IsJNDI>
<endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
<xmlParamName>InputXML</xmlParamName>
<HeaderKey></HeaderKey>
<HeaderValue></HeaderValue>
<ProviderUrl></ProviderUrl>
<jndiServerName></jndiServerName>
<jndiServerPort></jndiServerPort>
```

Figure 2.14

- Update the OmniDocs+RMS EJB container name [Defined in OmniDocs11.0EJB.yml file] in *jboss-ejb-client.properties* file located inside the OmniDocs11.0Web folder kept inside the mount path on NAS server.
  For example,

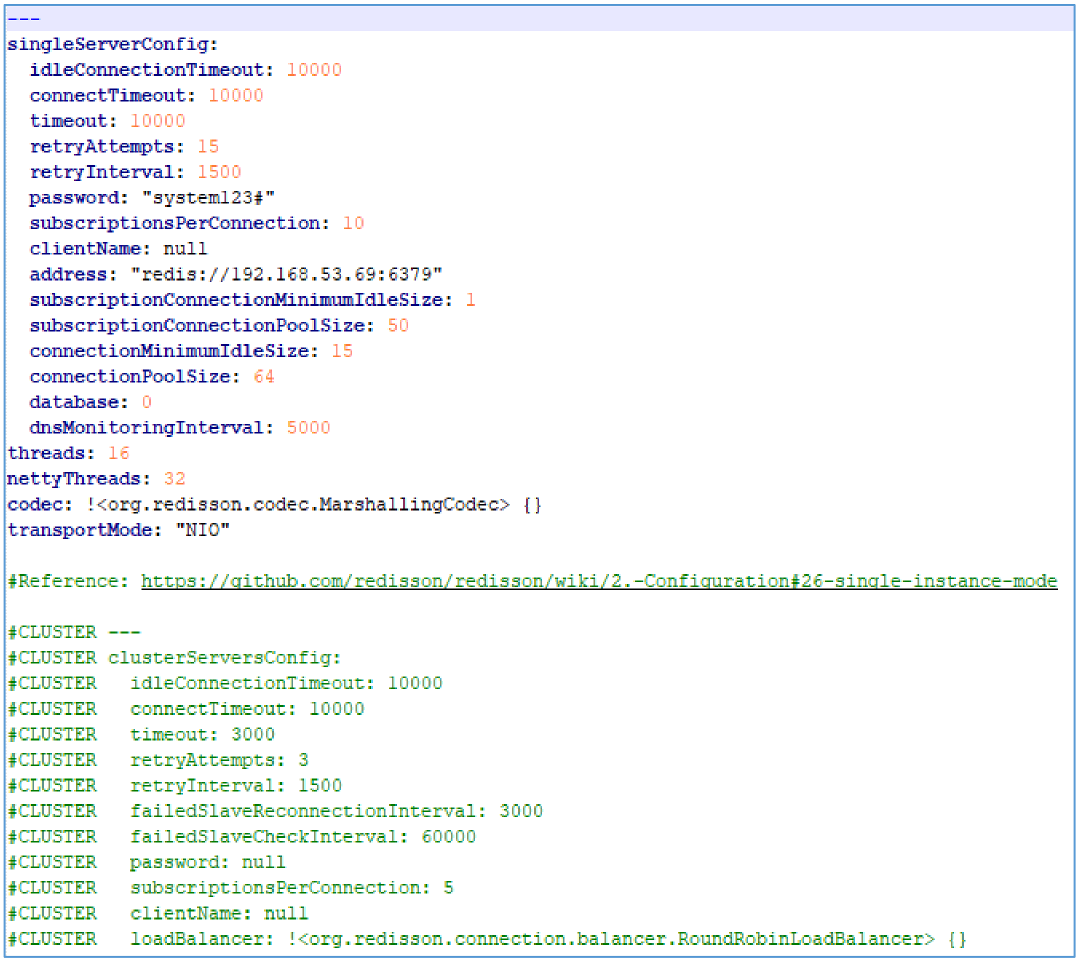


```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=false
remote.connections=default
remote.connection.default.host=od110ejb
remote.connection.default.port = 8080
remote.connection.default.username=[redacted]
remote.connection.default.password=[redacted]
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_NOANONYMOUS=false
```

Figure 2.15

Here, **od110ejb** is the name of the OmniDocs+*RMS* EJB container.

- Update the **RMSEnabled=Y** in *eworkstyle.ini* file located at *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/odwebini* folder at mapped location.
- Update the Redis cache's configuration endpoint in **redisson.yaml** file against the singleServerConfig or clusterServersConfig. If redis cache is SSL enabled then use rediss://<endpoint url>:port and if SSL is not enabled then use redis://<endpoint url>:port. This file *redisson.yaml* is located inside the OmniDocs11.0Web folder kept inside the mount path on NAS server.

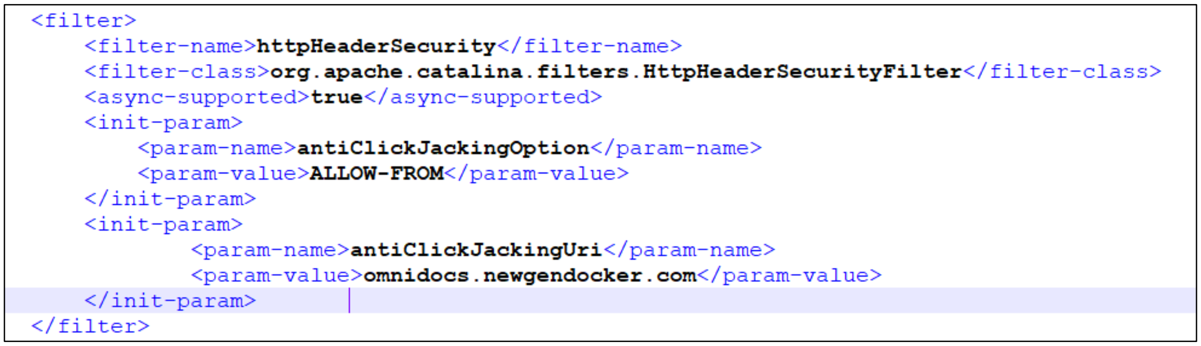


```
---
singleServerConfig:
  idleConnectionTimeout: 10000
  connectTimeout: 10000
  timeout: 10000
  retryAttempts: 15
  retryInterval: 1500
  password: "system123#"
  subscriptionsPerConnection: 10
  clientName: null
  address: "redis://192.168.53.69:6379"
  subscriptionConnectionMinimumIdleSize: 1
  subscriptionConnectionPoolSize: 50
  connectionMinimumIdleSize: 15
  connectionPoolSize: 64
  database: 0
  dnsMonitoringInterval: 5000
threads: 16
nettyThreads: 32
codec: !<org.redisson.codec.MarshallingCodec> {}
transportMode: "NIO"

#Reference: https://github.com/redisson/redisson/wiki/2.-Configuration#26-single-instance-mode

#CLUSTER ---
#CLUSTER clusterServersConfig:
#CLUSTER   idleConnectionTimeout: 10000
#CLUSTER   connectTimeout: 10000
#CLUSTER   timeout: 3000
#CLUSTER   retryAttempts: 3
#CLUSTER   retryInterval: 1500
#CLUSTER   failedSlaveReconnectionInterval: 3000
#CLUSTER   failedSlaveCheckInterval: 60000
#CLUSTER   password: null
#CLUSTER   subscriptionsPerConnection: 5
#CLUSTER   clientName: null
#CLUSTER   loadBalancer: !<org.redisson.connection.balancer.RoundRobinLoadBalancer> {}
```

Figure 2.16

- Open the *web.xml* file in edit mode located inside the OmniDocs11.0Web folder kept inside the mount path on NAS server.

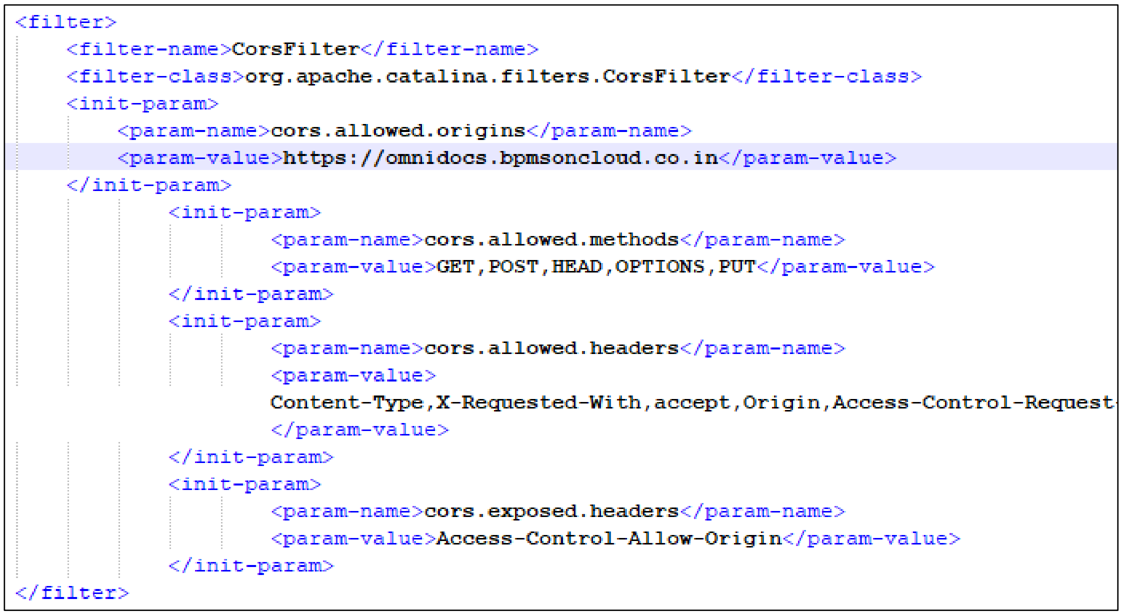
- Search for filter **httpHeaderSecurity** and update the *<param-value></param-value>* tag’s value with OmniDocs URL without context name against <param-name> *antiClickJackingUri</param-name>.*



```
<filter>
    <filter-name>httpHeaderSecurity</filter-name>
    <filter-class>org.apache.catalina.filters.HttpHeaderSecurityFilter</filter-class>
    <async-supported>true</async-supported>
    <init-param>
        <param-name>antiClickJackingOption</param-name>
        <param-value>ALLOW-FROM</param-value>
    </init-param>
    <init-param>
            <param-name>antiClickJackingUri</param-name>
            <param-value>omnidocs.newgendocker.com</param-value>
    </init-param>
</filter>
```

Figure 2.17

- Open the web_svc.xml file in edit mode located inside the OmniDocs11.0Web folder kept inside the mount path on NAS server.
- Search for filter-class "<filter-class>org.apache.catalina.filters.CorsFilter</filter-class>" and update the <param-value></param-value> tag’s value with OmniDocs URL with protocol against <param-name> antiClickJackingUri</param-name>.



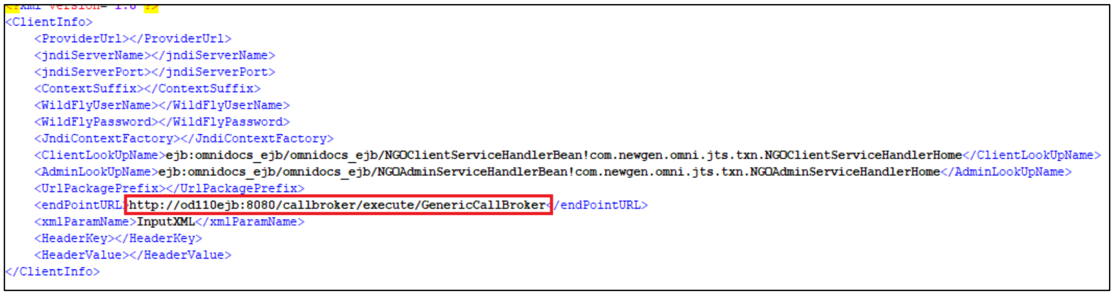
```
<filter>
    <filter-name>CorsFilter</filter-name>
    <filter-class>org.apache.catalina.filters.CorsFilter</filter-class>
    <init-param>
        <param-name>cors.allowed.origins</param-name>
        <param-value>https://omnidocs.bpmsoncloud.co.in</param-value>
    </init-param>
            <init-param>
                <param-name>cors.allowed.methods</param-name>
                <param-value>GET,POST,HEAD,OPTIONS,PUT</param-value>
            </init-param>
            <init-param>
                <param-name>cors.allowed.headers</param-name>
                <param-value>
                Content-Type,X-Requested-With,accept,Origin,Access-Control-Request-
                </param-value>
            </init-param>
            <init-param>
                <param-name>cors.exposed.headers</param-name>
                <param-value>Access-Control-Allow-Origin</param-value>
            </init-param>
</filter>
```

Figure 2.18

2.7.3 Wrapper changes

The changes are as follows:

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in NGOClientData.xml in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/Wrapper/ngdbini* folder kept inside the mount path on NAS server.



```
<?xml version="1.0"?>
<ClientInfo>
    <ProviderUrl></ProviderUrl>
    <jndiServerName></jndiServerName>
    <jndiServerPort></jndiServerPort>
    <ContextSuffix></ContextSuffix>
    <WildFlyUserName></WildFlyUserName>
    <WildFlyPassword></WildFlyPassword>
    <JndiContextFactory></JndiContextFactory>
    <ClientLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOClientServiceHandlerBean!com.newgen.omni.jts.txn.NGOClientServiceHandlerHome</ClientLookUpName>
    <AdminLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOAdminServiceHandlerBean!com.newgen.omni.jts.txn.NGOAdminServiceHandlerHome</AdminLookUpName>
    <UrlPackagePrefix></UrlPackagePrefix>
    <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
    <xmlParamName>InputXML</xmlParamName>
    <HeaderKey></HeaderKey>
    <HeaderValue></HeaderValue>
</ClientInfo>
```

Figure 2.19

Here, **od110ejb** is the name of the OmniDocsEJB container.

2.7.4 AlarmMailer changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

Make the changes in AlarmMailer that are as follows:

1. Update the OmniDocs+RMS EJB container name [Defined in OmniDocs11.0EJB.yml file] in *IS.ini* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services* or *AlarmMailer* folder kept inside the mount path on NAS server.
   For example,
   <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
   Here, **od110ejb** is the name of the OmniDocsEJB container.
2. Update the OmniDocsEJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/AlarmMailer/ngdbini* folder kept inside the mount path on NAS server.
   For example,
   <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
   Here, **od110ejb** is the name of the OmniDocsEJB container.
3. Update the below settings in the *Alarm.ini* file located inside the *OD11.0Services/AlarmMailer* folder kept inside the mount path on NAS server.
   i. Update the OmniDocs URL without context name in between the *<webservername></webservername>* tag.

For example,
<webservername>OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net
</webservername>
Here, *OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net* is the host path defined in the *AppGateway-IngressController.yml* file.

ii. Leave the WebServerPort as blank if OmniDocsWEB URL does not contain a port.
For example, <webserverport></webserverport>

iii. Update the OmniDocs cabinet name in between **<cabinetname></cabinetname>** tag.
For example, <cabinetname>ecmsuite</cabinetname>
Here, **ecmsuite** is the OmniDocs cabinet name gets created.

iv. Update the OmniDocs supervisor group's user in between the **<user></user>** tag.
For example, <user>supervisor</user>

v. Update the OmniDocs supervisor group's user password in between the **<password></password>** tag. Ensure that this password must be in an encrypted format.
For example, <password>:X-D;U:T-C;P-C;p5-C;b:d:</password>

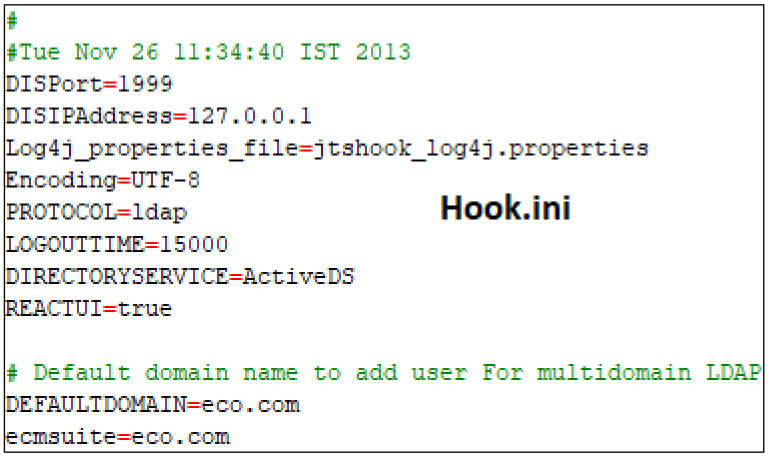
2.7.5 LDAP changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to the Cabinet and Data Source Creation section.

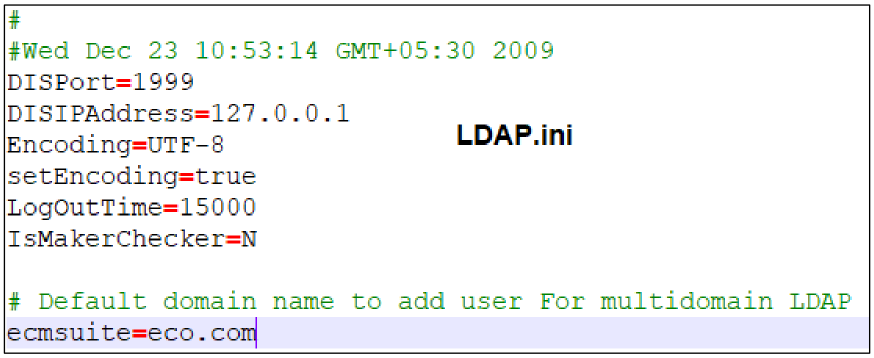
The changes in LDAP are as follows: (For On_Prem Active Directory)

- Ensure that the LDAP Domain server is configured, and a private tunnel is created between the Kubernetes worker nodes and the LDAP Domain server.
- Update the OmniDocs+RMS EJB container name [Defined in OmniDocs11.0EJB.yml file] in *NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/ODAuthMgr/ngdbini* folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocsEJB container.
- Update the cabinet name and domain name in the ldap.ini and Hook.ini file located inside the *OD11.0Services/ODAuthMgr* folder kept inside the mount path on NAS server.



```
#
#Tue Nov 26 11:34:40 IST 2013
DISPort=1999
DISIPAddress=127.0.0.1
Log4j_properties_file=jtshook_log4j.properties
Encoding=UTF-8
PROTOCOL=ldap
LOGOUTTIME=15000
DIRECTORYSERVICE=ActiveDS
REACTUI=true

# Default domain name to add user For multidomain LDAP
DEFAULTDOMAIN=eco.com
ecmsuite=eco.com
```

Figure 2.20

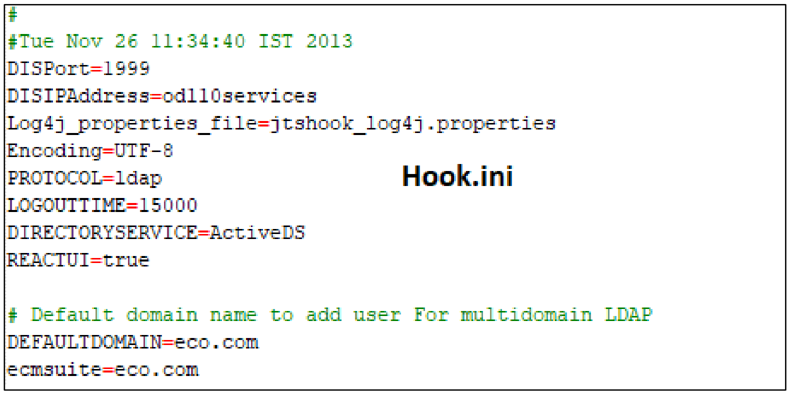


```
#
#Wed Dec 23 10:53:14 GMT+05:30 2009
DISPort=1999
DISIPAddress=127.0.0.1
Encoding=UTF-8
setEncoding=true
LogOutTime=15000
IsMakerChecker=N

# Default domain name to add user For multidomain LDAP
ecmsuite=eco.com
```

Figure 2.21

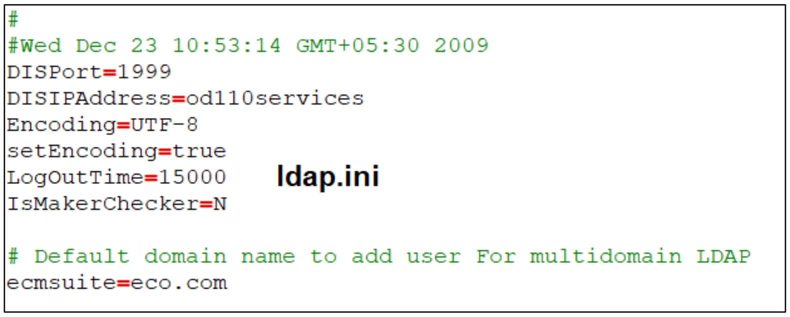
Here, **ecmsuite** is the cabinet name and *eco.com* is the domain name.

- Update the same cabinet name and domain name in the *ldap.ini* and *Hook.ini* file located inside the *OmniDocs11.0Web\Newgen\NGConfig* folder kept inside the mount path on NAS server.
- Update the OD11.0Services container's service name [Defined in respective YAML file] in *ldap.ini* and *Hook.ini* file located inside the *OmniDocs11.0Web\Newgen\NGConfig* folder kept inside the mount path on NAS server.



```
#
#Tue Nov 26 11:34:40 IST 2013
DISPort=1999
DISIPAddress=od110services
Log4j_properties_file=jtshook_log4j.properties
Encoding=UTF-8
PROTOCOL=ldap
LOGOUTTIME=15000
DIRECTORYSERVICE=ActiveDS
REACTUI=true

# Default domain name to add user For multidomain LDAP
DEFAULTDOMAIN=eco.com
ecmsuite=eco.com
```

Figure 2.22

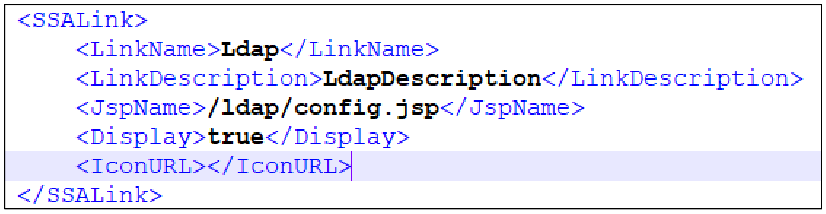


```
#
#Wed Dec 23 10:53:14 GMT+05:30 2009
DISPort=1999
DISIPAddress=od110services
Encoding=UTF-8
setEncoding=true
LogOutTime=15000
IsMakerChecker=N

# Default domain name to add user For multidomain LDAP
ecmsuite=eco.com
```

Figure 2.23

Here, **od110services** is the service name of the OD11.0Services container.

- Set **<Display>** as true for LDAP in *AdminMenuOptions.xml* located inside *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/Custom/CABINETNAME* folder kept inside the mount path on NAS server.

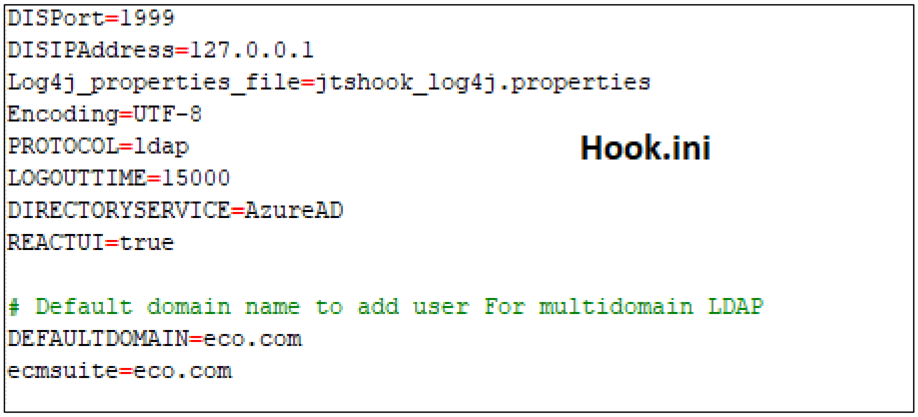


```
<SSALink>
    <LinkName>Ldap</LinkName>
    <LinkDescription>LdapDescription</LinkDescription>
    <JspName>/ldap/config.jsp</JspName>
    <Display>true</Display>
    <IconURL></IconURL>
</SSALink>
```

Figure 2.24

The changes in LDAP are as follows: (For Azure Active Directory)

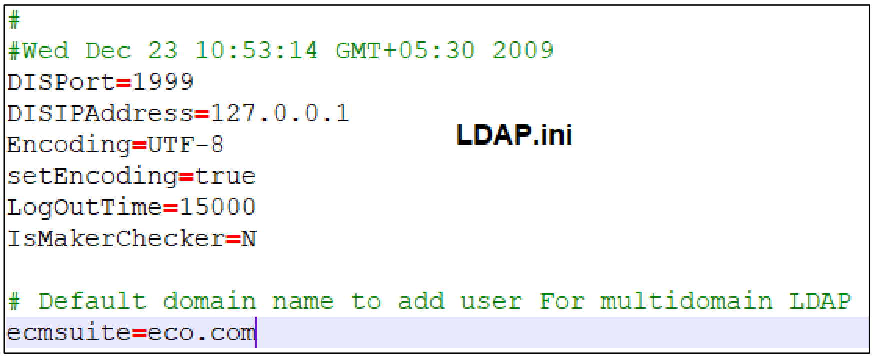
- Update the OmniDocs+RMS EJB container name [Defined in OmniDocs11.0EJB.yml file] in *NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/ODAuthMgr/ngdbini* folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocsEJB container.
- Update the cabinet name, domain name, and directory service as **AzureAD** in the Hook.ini file located inside the *OD11.0Services/ODAuthMgr* folder kept inside the mount path on NAS server.



```
DISPort=1999
DISIPAddress=127.0.0.1
Log4j_properties_file=jtshook_log4j.properties
Encoding=UTF-8
PROTOCOL=ldap
LOGOUTTIME=15000
DIRECTORYSERVICE=AzureAD
REACTUI=true

# Default domain name to add user For multidomain LDAP
DEFAULTDOMAIN=eco.com
ecmsuite=eco.com
```

Figure 2.25

- Update the cabinet name and domain name in the *ldap.ini* file located inside the *OD11.0Services* or *ODAuthMgr* folder kept inside the mount path on NAS server.

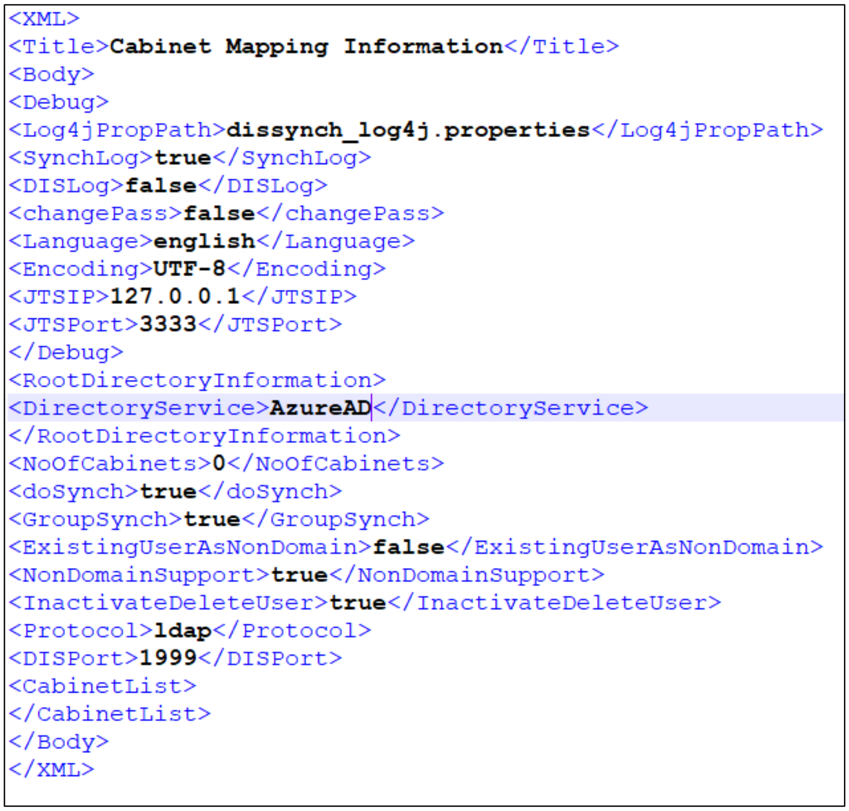


```
#
#Wed Dec 23 10:53:14 GMT+05:30 2009
DISPort=1999
DISIPAddress=127.0.0.1
Encoding=UTF-8
setEncoding=true
LogOutTime=15000
IsMakerChecker=N

# Default domain name to add user For multidomain LDAP
ecmsuite=eco.com
```

Figure 2.26

Here, **ecmsuite** is the cabinet name and *eco.com* is the domain name.

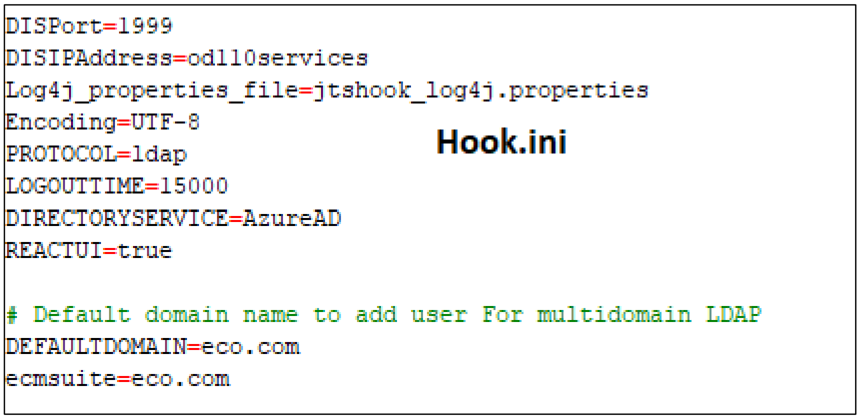
- Update the directory service as **AzureAD** in the DIS.xml file located inside the **OD11.0Services** or **ODAuthMgr** folder kept inside the mount path on NAS server.



```
<XML>
<Title>Cabinet Mapping Information</Title>
<Body>
<Debug>
<Log4jPropPath>dissynch_log4j.properties</Log4jPropPath>
<SynchLog>true</SynchLog>
<DISLog>false</DISLog>
<changePass>false</changePass>
<Language>english</Language>
<Encoding>UTF-8</Encoding>
<JTSIP>127.0.0.1</JTSIP>
<JTSPort>3333</JTSPort>
</Debug>
<RootDirectoryInformation>
<DirectoryService>AzureAD</DirectoryService>
</RootDirectoryInformation>
<NoOfCabinets>0</NoOfCabinets>
<doSynch>true</doSynch>
<GroupSynch>true</GroupSynch>
<ExistingUserAsNonDomain>false</ExistingUserAsNonDomain>
<NonDomainSupport>true</NonDomainSupport>
<InactivateDeleteUser>true</InactivateDeleteUser>
<Protocol>ldap</Protocol>
<DISPort>1999</DISPort>
<CabinetList>
</CabinetList>
</Body>
</XML>
```

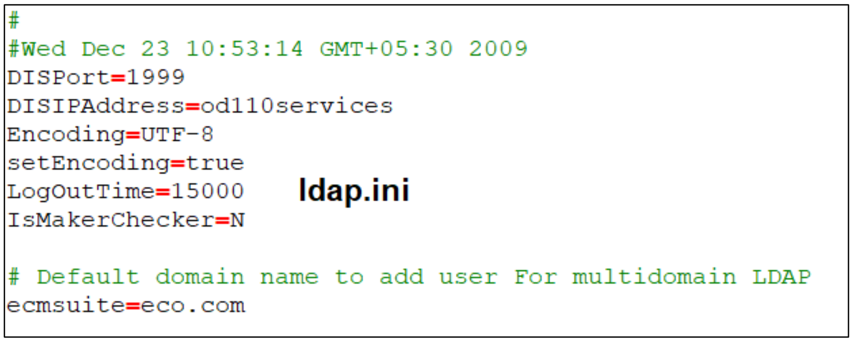
Figure 2.27

- Update the same cabinet name and domain name in the *ldap.ini* and *Hook.ini* file located inside the **OmniDocs11.0Web\Newgen\NGConfig** folder kept inside the mount path on NAS server.
- Update the OD11.0Services container's service name [Defined in respective YAML file] in ldap.ini and Hook.ini file located inside the **OmniDocs11.0Web\Newgen\NGConfig** folder kept inside the mount path on NAS server.
- Update the directory service as **AzureAD** in Hook.ini and config.ini located inside the **OmniDocs11.0Web\Newgen\NGConfig** folder kept inside the mount path on NAS server.



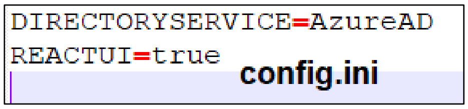
```
DISPort=1999
DISIPAddress=od110services
Log4j_properties_file=jtshook_log4j.properties
Encoding=UTF-8
PROTOCOL=ldap
LOGOUTTIME=15000
DIRECTORYSERVICE=AzureAD
REACTUI=true

# Default domain name to add user For multidomain LDAP
DEFAULTDOMAIN=eco.com
ecmsuite=eco.com
```

Figure 2.28



```
#
#Wed Dec 23 10:53:14 GMT+05:30 2009
DISPort=1999
DISIPAddress=od110services
Encoding=UTF-8
setEncoding=true
LogOutTime=15000
IsMakerChecker=N

# Default domain name to add user For multidomain LDAP
ecmsuite=eco.com
```

Figure 2.29



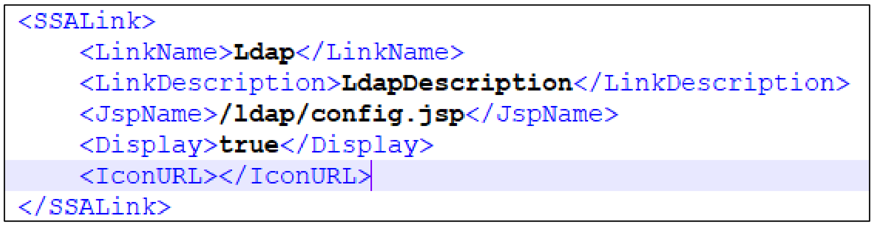
```
DIRECTORYSERVICE=AzureAD
REACTUI=true
```

Figure 2.30

Here, **od110services** is the service name of the OD11.0Services container.

- Set **<Display>** as true for ldap in *AdminMenuOptions.xml* located inside *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/Custom/CABINETNAME* folder kept inside the mount path on NAS server.

  For example,



```
<SSALink>
    <LinkName>Ldap</LinkName>
    <LinkDescription>LdapDescription</LinkDescription>
    <JspName>/ldap/config.jsp</JspName>
    <Display>true</Display>
    <IconURL></IconURL>
</SSALink>
```

Figure 2.31

2.7.6 SSO changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

The changes in SSO are as follows:

- Update the *<Host-Path URL of OmniDocsWeb container>* at the place of *ibps5aurora.newgendocker.com* in *mapping.xml* file located inside the *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/SSOConFig* folder.
- Update the **CabinetName** in *mapping.xml* file located inside the *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/SSOConFig* folder.

- Configure the CabinetName=DomainName in *sso.ini* file located inside the *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/SSOConFig* folder.

ecmsuite=eco.com

2.7.7 Scheduler changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

The changes in Scheduler are as follows:

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *IS.ini* in between the *<endPointURL></endPointURL>* tags file located inside the **OD11.0Services** or **Scheduler** folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocs_RMS EJB container.
- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/Scheduler/ngdbini* folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocs+RMS EJB container.
- Update the OD11.0Services container's service name [Defined in respective YAML file] in *SchedulerConf.ini* file located at **OD11.0Services** or **Scheduler** folder kept inside the mount path on NAS server.
  For example: **schedulerIpAddress=od110services**
- Update the OD11.0Services container's service name [Defined in respective YAML file] in eworkstyle.ini file located at *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/Custom/<CABINETNAME>* folder kept inside the mount path on NAS server.
  For example: **schedularLocation=od110services**

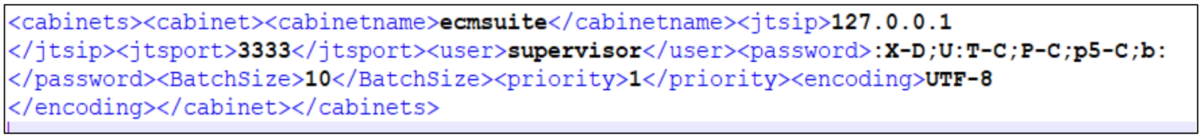
2.7.8 ThumbnailManager changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

The changes in ThumbnailManager are as follows:

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *IS.ini* in between the *<endPointURL></endPointURL>* tags file located inside the **OD11.0Services** or **ThumbnailManager** folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocs+RMS EJB container.
- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the *OD11.0Services/ThumbnailManager/ngdbini* folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocs+RMS EJB container.
- Update the cabinet name, supervisor group's user name, and password in *ThumnailConfig.xml* located inside the **OD11.0Services** or **ThumbnailManager** folder kept inside the mount path on NAS server.



```
<cabinets><cabinet><cabinetname>ecmsuite</cabinetname><jtsip>127.0.0.1
</jtsip><jtsport>3333</jtsport><user>supervisor</user><password>:X-D;U:T-C;P-C;p5-C;b:
</password><BatchSize>10</BatchSize><priority>1</priority><encoding>UTF-8
</encoding></cabinet></cabinets>
```

Figure 2.32

2.7.9 TEM changes

Prerequisite:

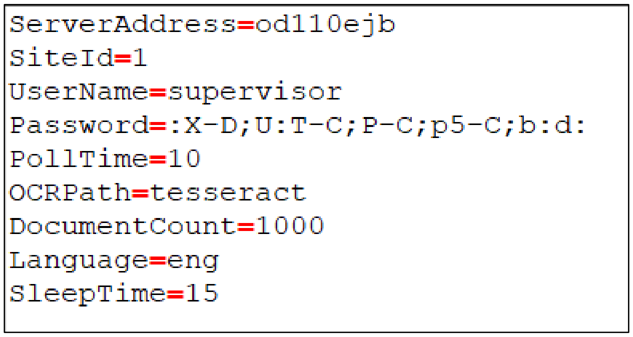
The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

The changes in TEM are as follows:

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *IS.ini and NGOClientData.xml* in between the *<endPointURL></endPointURL>* tags file located inside the **TEM11.0** folder kept inside the mount path on NAS server.
  For example,
  <endPointURL>http://od110ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
  Here, **od110ejb** is the name of the OmniDocs+RMS EJB container.
- Update the cabinet name in filename FTSServer-CABINETNAME-1.properties.

For example: FTSServer-**ecmsuite**-1.properties [ecmsuite is the cabinet name].

- Update the OmniDocs+RMS EJB container name [Defined in OmniDocs11.0EJB.yml file] in FTSServer-**ecmsuite**-1.properties renamed earlier.
- Update the OmniDocs supervisor group's user name.
- Update the OmniDocs supervisor group's user password. Ensure this password must be in an encrypted format.



```
ServerAddress=od110ejb
SiteId=1
UserName=supervisor
Password=:X-D;U:T-C;P-C;p5-C;b:d:
PollTime=10
OCRPath=tesseract
DocumentCount=1000
Language=eng
SleepTime=15
```

Figure 2.33

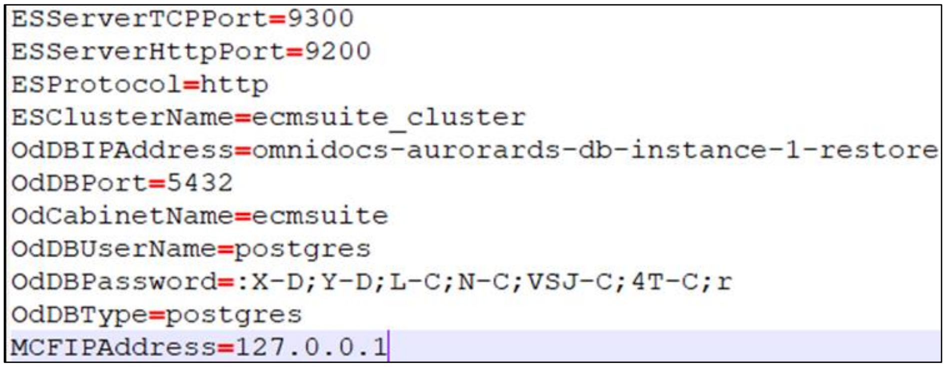
2.7.10 EasySearch changes

Prerequisite:

The cabinet is created and associated with the running containers. If the cabinet is not created, then refer to Cabinet and Data Source Creation section.

The changes in EasySearch (Apache Manifold only) are as follows:

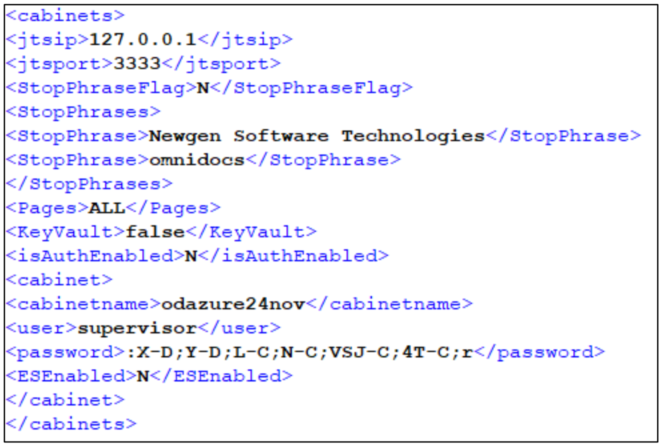
- Update the Database details in the *ESconfig.ini* file located inside the *EasySearch11.0\ESConfigurator\conf* folder kept inside the mount path on NAS server.
  - ➢ ESClusterName=CABINETNAME_cluster
  - ➢ OdDBIPAddress=DBIP
  - ➢ OdDBPort=DBPORT
  - ➢ OdCabinetName=CABINETNAME
  - ➢ OdDBUserName=DBUSER
  - ➢ OdDBPassword=DBPASSWORD in encrypted format
  - ➢ OdDBType=sqlserver | oracle | postgres



```
ESServerTCPPort=9300
ESServerHttpPort=9200
ESProtocol=http
ESClusterName=ecmsuite_cluster
OdDBIPAddress=omnidocs-aurorards-db-instance-1-restore
OdDBPort=5432
OdCabinetName=ecmsuite
OdDBUserName=postgres
OdDBPassword=:X-D;Y-D;L-C;N-C;VSJ-C;4T-C;r
OdDBType=postgres
MCFIPAddress=127.0.0.1
```

Figure 2.34

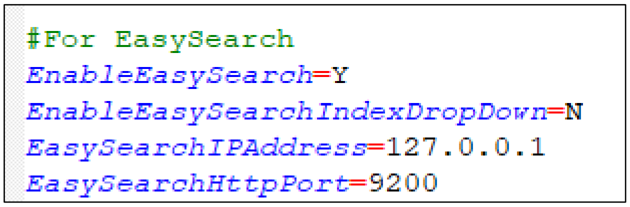
- Update **AppToBeConfigured=ApacheManifold** in the *ESconfig.ini* file located inside the *EasySearch11.0\ESConfigurator\conf* folder kept inside the mount path on NAS server.
- Update the cabinet name in the **CrawlerConfig.xml** file located inside the **EasySearch11.0\apache-manifoldcf-2.19\example** folder kept inside the mount path on NAS server.
- Update the OmniDocs supervisor group's user name.
- Update the OmniDocs supervisor group's user password. Ensure this password must be in an encrypted format.



```
<cabinets>
<jtsip>127.0.0.1</jtsip>
<jtsport>3333</jtsport>
<StopPhraseFlag>N</StopPhraseFlag>
<StopPhrases>
<StopPhrase>Newgen Software Technologies</StopPhrase>
<StopPhrase>omnidocs</StopPhrase>
</StopPhrases>
<Pages>ALL</Pages>
<KeyVault>false</KeyVault>
<isAuthEnabled>N</isAuthEnabled>
<cabinet>
<cabinetname>odazure24nov</cabinetname>
<user>supervisor</user>
<password>:X-D;Y-D;L-C;N-C;VSJ-C;4T-C;r</password>
<ESEnabled>N</ESEnabled>
</cabinet>
</cabinets>
```

Figure 2.35

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in ***NGOClientData.xml** and **RMClientData.xml*** in between the *<endPointURL></endPointURL>* tags file located inside the EasySearch11.0/apache-manifoldcf-2.19/example/Newgen/NGConfig/ngdbini folder kept inside the mount path on NAS server.
- Update the EnableEasySearch=Y in the *eworkstyle.ini* file located inside the *OmniDocs11.0Web\Newgen\NGConfig\ngdbini\Custom\CABINET_NAME* folder kept inside the mount path on NAS server.

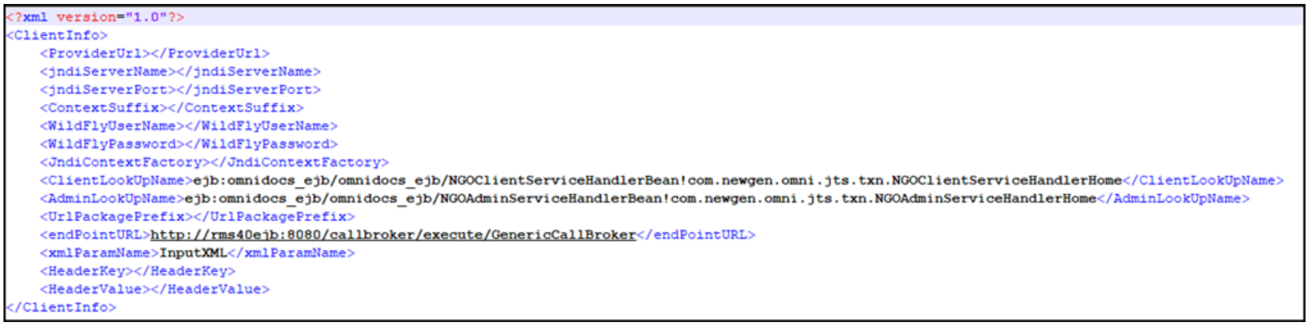


```
#For EasySearch
EnableEasySearch=Y
EnableEasySearchIndexDropDown=N
EasySearchIPAddress=127.0.0.1
EasySearchHttpPort=9200
```

Figure 2.36

2.7.11 WOPI Changes

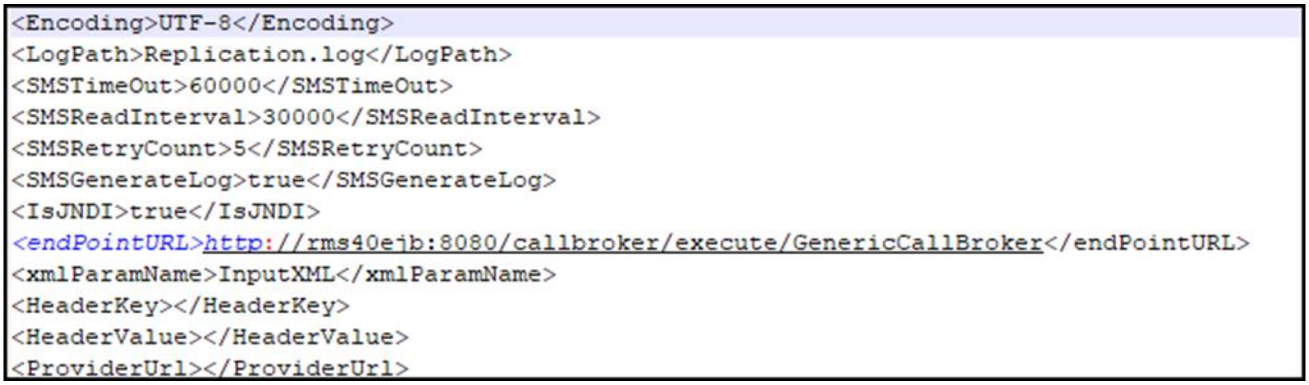
- Update the OmniDocsEJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *NGOClientData.xml* file in between the *<endPointURL></endPointURL>* tags located inside the *OmniDocs_WOPI\Newgen\NGConfig\ngdbini* folder at the mapped location on the Worker node.



```
<?xml version="1.0"?>
<ClientInfo>
    <ProviderUrl></ProviderUrl>
    <jndiServerName></jndiServerName>
    <jndiServerPort></jndiServerPort>
    <ContextSuffix></ContextSuffix>
    <WildFlyUserName></WildFlyUserName>
    <WildFlyPassword></WildFlyPassword>
    <JndiContextFactory></JndiContextFactory>
    <ClientLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOClientServiceHandlerBean!com.newgen.omni.jts.txn.NGOClientServiceHandlerHome</ClientLookUpName>
    <AdminLookUpName>ejb:omnidocs_ejb/omnidocs_ejb/NGOAdminServiceHandlerBean!com.newgen.omni.jts.txn.NGOAdminServiceHandlerHome</AdminLookUpName>
    <UrlPackagePrefix></UrlPackagePrefix>
    <endPointURL>http://rms40ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
    <xmlParamName>InputXML</xmlParamName>
    <HeaderKey></HeaderKey>
    <HeaderValue></HeaderValue>
</ClientInfo>
```

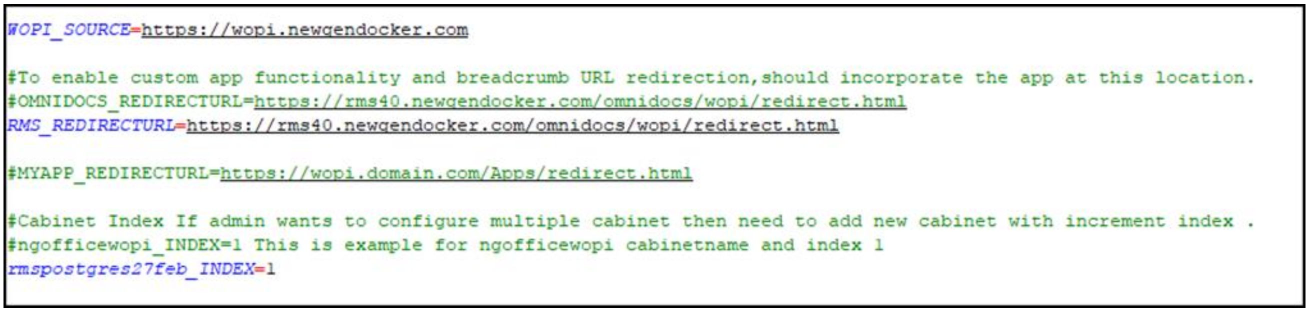
Here, **rms40ejb** is the name of the OmniDocs+RMS EJB container.

- Update the OmniDocs+RMS EJB container name [Defined in *OmniDocs11.0EJB.yml* file] in *IS.ini* file in between the *<endPointURL ></endPointURL >* tags located inside the *OmniDocs_WOPI\Newgen\NGConfig* folder at the mapped location on the Worker node.
  For example,



```
<Encoding>UTF-8</Encoding>
<LogPath>Replication.log</LogPath>
<SMSTimeOut>60000</SMSTimeOut>
<SMSReadInterval>30000</SMSReadInterval>
<SMSRetryCount>5</SMSRetryCount>
<SMSGenerateLog>true</SMSGenerateLog>
<IsJNDI>true</IsJNDI>
<endPointURL>http://rms40ejb:8080/callbroker/execute/GenericCallBroker</endPointURL>
<xmlParamName>InputXML</xmlParamName>
<HeaderKey></HeaderKey>
<HeaderValue></HeaderValue>
<ProviderUrl></ProviderUrl>
```

- Update the WOPI_SOURCE, RMS_REDIRECTURL and CABINETNAME in *WOPIConfiguration.ini* file located inside the *OmniDocs_WOPI\Newgen\NGConfig\AddInsConfig* folder at the mapped location on the Worker node.

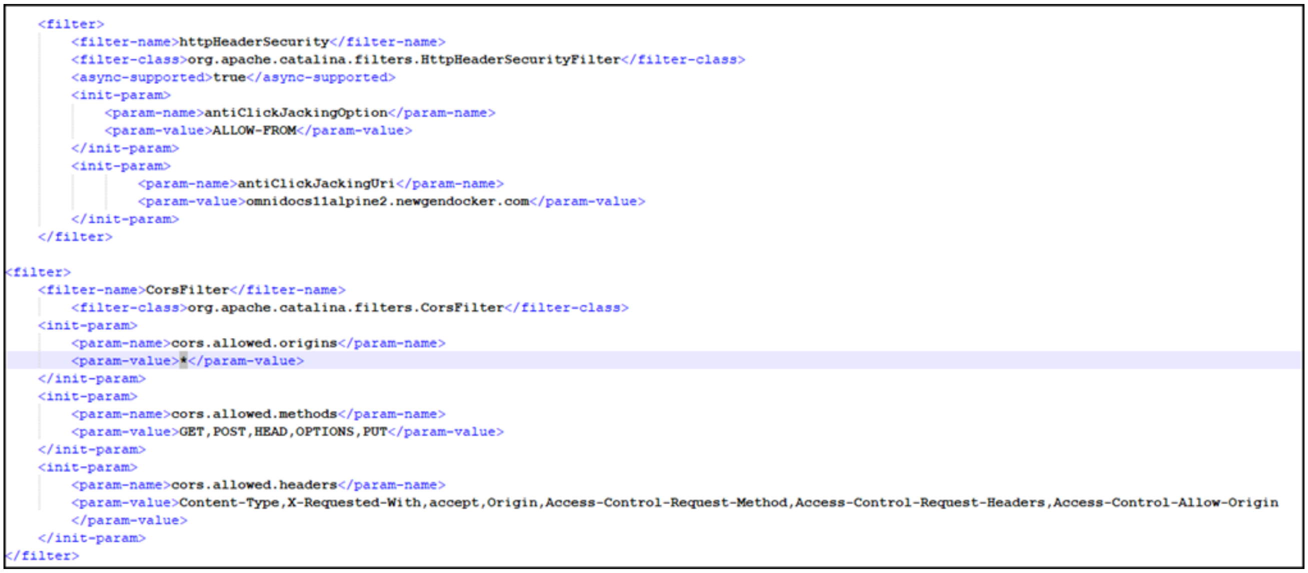


```
WOPI_SOURCE=https://wopi.newgendocker.com

#To enable custom app functionality and breadcrumb URL redirection,should incorporate the app at this location.
#OMNIDOCS_REDIRECTURL=https://rms40.newgendocker.com/omnidocs/wopi/redirect.html
RMS_REDIRECTURL=https://rms40.newgendocker.com/omnidocs/wopi/redirect.html

#MYAPP_REDIRECTURL=https://wopi.domain.com/Apps/redirect.html

#Cabinet Index If admin wants to configure multiple cabinet then need to add new cabinet with increment index .
#ngofficewopi_INDEX=1 This is example for ngofficewopi cabinetname and index 1
rmspostgres27feb_INDEX=1
```

Where,

https://wopi.newgendocker.com is host URL of WOPI container.
https://rms40.newgendocker.com is Host URL of RMS WEB container.
rmspostgres27feb is cabinet name.

- Open the *web.xml* file in edit mode located inside the *OmniDocs_WOPI* folder at the mapped location on the Worker node.
- Search for filter-class *<filter-class>org.apache.catalina.filters.CorsFilter</filter-class>* and update the *<param-value></param-value>* tag's value with OmniDocs URL against *<param-name> antiClickJackingUri</param-name> and * against <param-name>cors.allowed.origins</param-name>*



```
    <filter>
        <filter-name>httpHeaderSecurity</filter-name>
        <filter-class>org.apache.catalina.filters.HttpHeaderSecurityFilter</filter-class>
        <async-supported>true</async-supported>
        <init-param>
            <param-name>antiClickJackingOption</param-name>
            <param-value>ALLOW-FROM</param-value>
        </init-param>
        <init-param>
                <param-name>antiClickJackingUri</param-name>
                <param-value>omnidocs11alpine2.newgendocker.com</param-value>
        </init-param>
    </filter>

<filter>
    <filter-name>CorsFilter</filter-name>
        <filter-class>org.apache.catalina.filters.CorsFilter</filter-class>
    <init-param>
        <param-name>cors.allowed.origins</param-name>
        <param-value>*</param-value>
    </init-param>
    <init-param>
        <param-name>cors.allowed.methods</param-name>
        <param-value>GET,POST,HEAD,OPTIONS,PUT</param-value>
    </init-param>
    <init-param>
        <param-name>cors.allowed.headers</param-name>
        <param-value>Content-Type,X-Requested-With,accept,Origin,Access-Control-Request-Method,Access-Control-Request-Headers,Access-Control-Allow-Origin
        </param-value>
    </init-param>
</filter>
```

- Add the CSPHeaderAllowedDomains tag in the *eworkstyle.ini* file located inside the *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/odwebini* folder at the mapped location on the Worker node.

```
CSPHeaderAllowedDomains=default-src * data: 'unsafe-inline' 'unsafe-eval';
```

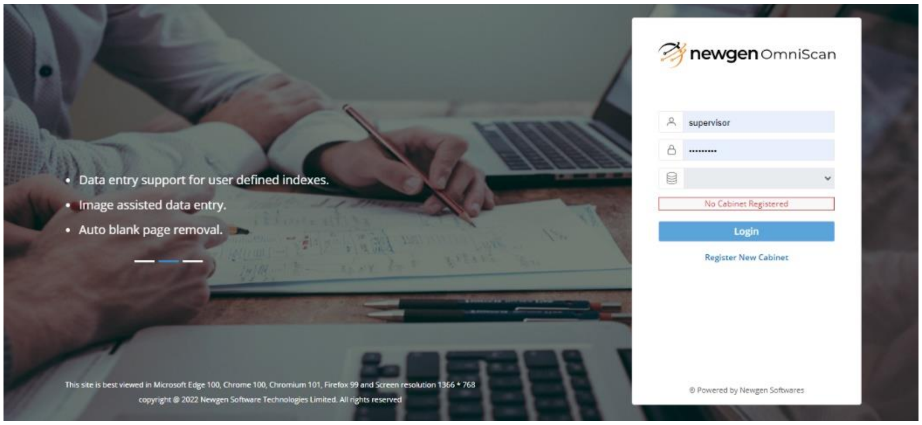
- Add the WOPIOfficeExtensionSuppport and WOPIOfficeExtensionSuppportURL tag in the *eworkstyle.ini* file located inside the *OmniDocs11.0Web/Newgen/NGConfig/ngdbini/Custom/CABINET_NAME* folder at the mapped location on the Worker node.

```
WOPIOfficeExtensionSuppport=doc,docx,DOCX,DOC,xls,xlsx,XLSX,XLS,ppt,pptx,PPTX,PPT,wopitest,WOPITEST,wopitestx,WOPITESTX
WOPIOfficeExtensionSuppportURL=https://wopi.newgendocker.com
```

2.7.12 OmniScanWeb changes

Perform the below steps to register the cabinet in OmniScanWeb:

1. Open the OmniScanWeb using the following URL:
   http://<Host-Path URL of OmniScanWeb container>/omniscanweb
   For example,
   https://omniscan.apps.newgenopenshiftcluster.newgensoftware.net/omniscanweb
2. Click **Register New Cabinet** link on the OmniScan Web login screen.



- Data entry support for user defined indexes.
- Image assisted data entry.
- Auto blank page removal.

supervisor

....

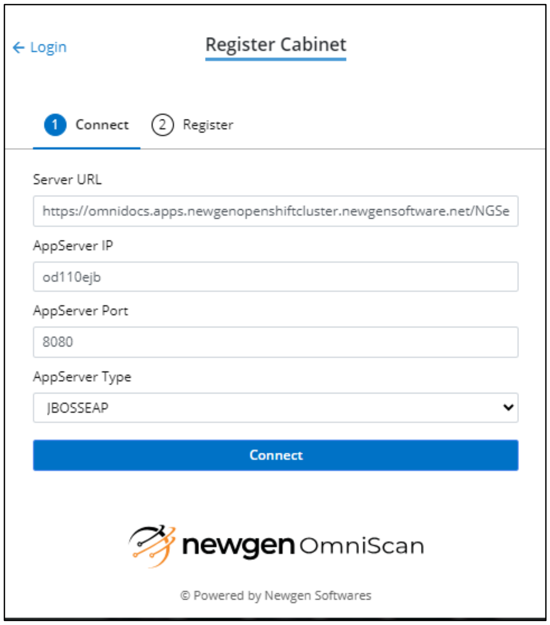
No Cabinet Registered

This site is best viewed in Microsoft Edge 100, Chrome 100, Chromium 101, Firefox 99 and Screen resolution 1366 * 768
copyright @ 2022 Newgen Software Technologies Limited. All rights reserved

Figure 2.37

3. Specify the Server URL as given below:
   http://<Host-Path URL of OmniDocsWeb container>/NGServlet/servlet/ExternalServlet
   For example,
   https://OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net/NGServlet/servlet/ExternalServlet

4. Specify the **OmniDocs+RMS EJB** container name for AppServer IP or Server URL, 8080 for AppServer Port, and JBOSSEAP for AppServer Type.



Server URL
https://omnidocs.apps.newgenopenshiftcluster.newgensoftware.net/NGSe

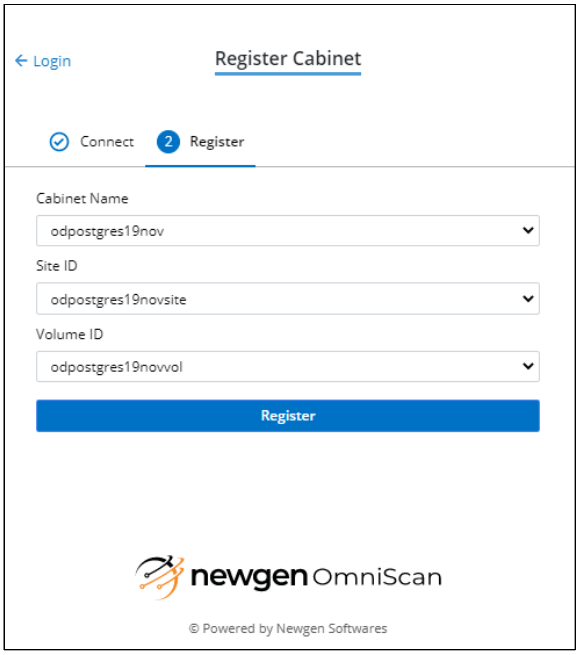
AppServer IP
od110ejb

AppServer Port
8080

AppServer Type
JBOSSEAP

Figure 2.38

5. Click **Connect**.
6. Select the **Cabinet Name**, **Site ID**, and **Volume ID** from the list.



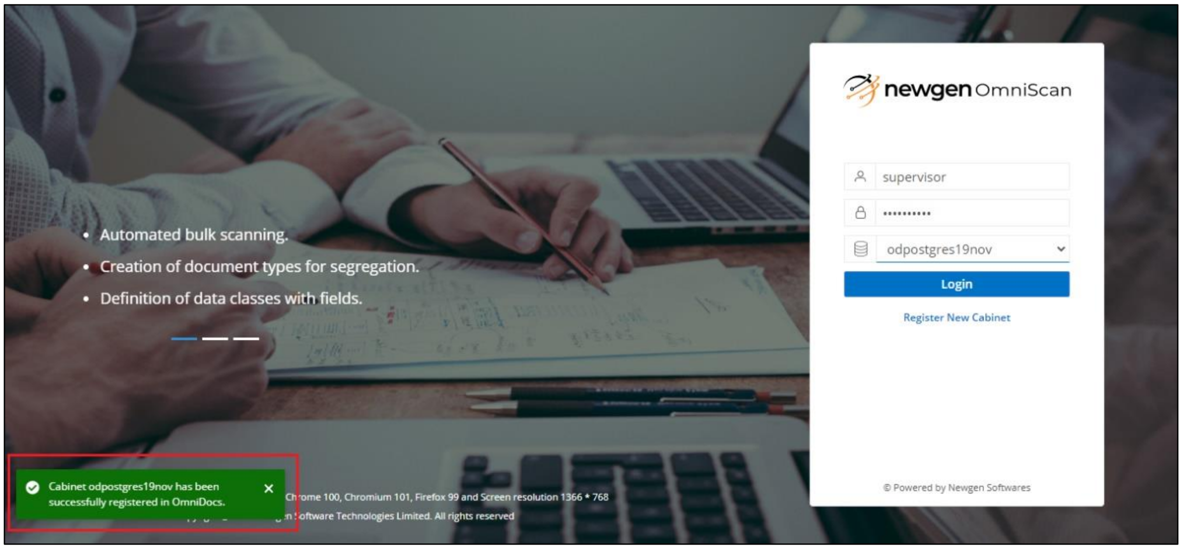
Cabinet Name
odpostgres19nov

Site ID
odpostgres19novsite

Volume ID
odpostgres19novvol

Figure 2.39

7. Click **Register**.



- Automated bulk scanning.
- Creation of document types for segregation.
- Definition of data classes with fields.

supervisor

••••••••••

odpostgres19nov

Cabinet odpostgres19nov has been successfully registered in OmniDocs.

newgen OmniScan
Login
Register New Cabinet
© Powered by Newgen Softwares
Chrome 100, Chromium 101, Firefox 99 and Screen resolution 1366 * 768
Software Technologies Limited. All rights reserved

Figure 2.40

The registered cabinet appears in the **Cabinet Name** list on the login screen. Now you can log into OmniScan Web.

NOTE:
Ensure that the **OmniScan_Template_Repository** folder is already created in OmniDocs before logging into OmniScan Web.

2.7.13 RMS SharePoint Adapter Chnages

Prerequisite:

The cabinet must be created and associated with the running containers. To create a cabinet, refer to the Creating cabinet and data source section.

The changes in RMS SharePoint Adapter are as follows:

1. Update the cabinet name in filename RMS-SPServer-CABINETNAME-1.properties located inside SharePointAdapter/properties folder at the mapped location on the worker node. For example, RMS-SPServer-ecmsuite-1.properties [ecmsuite is the cabinet name].
2. Change the required changes in RMS-SPServer-CABINETNAME-1.properties file.

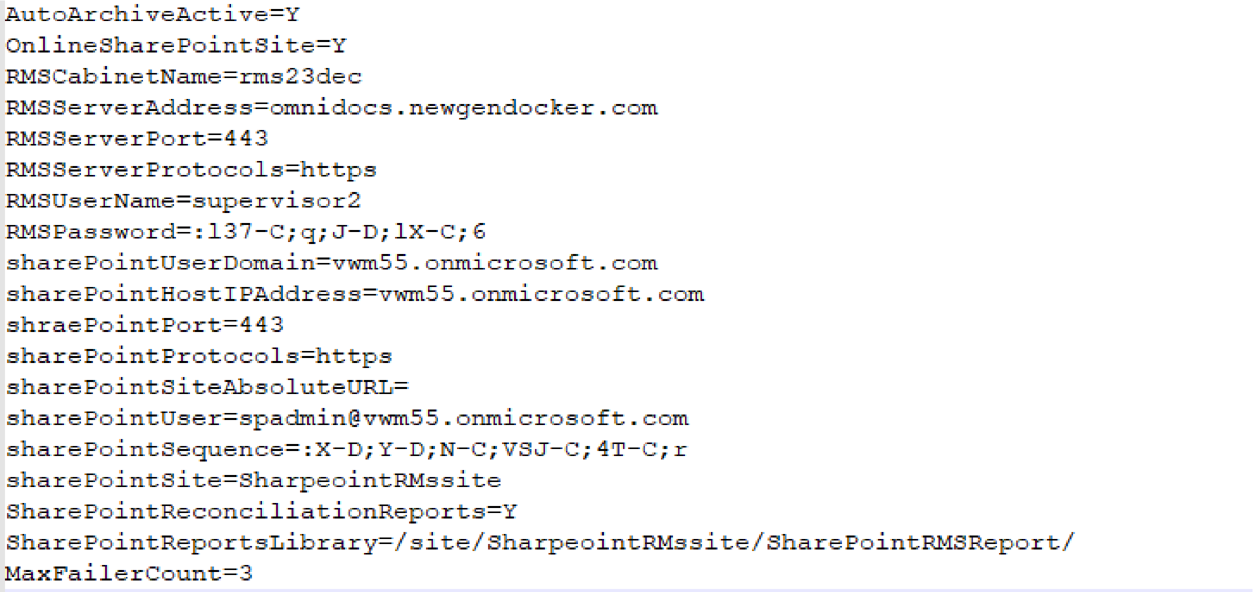
   Following is the sample of the Properties file:

```
AutoArchiveActive=Y
OnlineSharePointSite=Y
RMSCabinetName =spcab2 RMSServerAddress=omnidocs.newgendocker.com
RMSServerPort=443
RMSServerProtocols=https
RMSUserName=supervisor2
RMSPassword=:X-D;U:T-C;P-C;p5-C;b:d:u:SJDE
sharePointUserDomain=vwm55.onmicrosoft.com
```

```
sharePointHostIPAddress=vwm55.onmicrosoft.com
shraePointPort=443
sharePointProtocols=https
sharePointSiteAbsoluteURL=
sharePointUser=spadmin@vwm55.onmicrosoft.com
sharePointSequence=:F:0s-C;6-C;G-C;a:8:y-D;W:Z-C;4P-C;a-C;k:0
sharePointSite=newSite
SharePointReconciliationReports=Y
SharePointReportsLibrary=/sites/newSite/SharePointRMSReport/
MaxFailerCount=3
```

Description of the Properties keys:

Keys	Description
AutoArchiveActive	Y signifies this properties file is in used whereas N signifies this file is not in use.
OnlineSharePointSite	Y – Signifies online method is working. N – Signifies offline method is working.
RMSCabinetName	Name of the RMS cabinet.
RMSServerAddress	RMS web container host URL.
RMSServerPort	RMS web container port that is, 443 = if SSL enabled or 80 = if SSL Disabled.
RMSServerProtocols	RMS server protocol that is, https = if SSL enabled or http = if SSL Disabled.
RMSUserName	RMS supervisor's group user name.
RMSPassword	RMS supervisor's group user encrypt password from alarm mailer.
sharePointUserDomain	SharePoint domain name.
sharePointHostIPAddress	SharePoint site server address.
shraePointPort	SharePoint site server port.
sharePointProtocols	SharePoint site server protocols.
sharePointSiteAbsoluteURL	SharePoint site absolute URL.
sharePointUser	SharePoint admin user name.
sharePointSequence	SharePoint encrypt password from alarm mailer.
sharePointSite	SharePoint site name.
SharePointReconciliationReports	Y or N.
SharePointReportsLibrary	SharePoint Reconciliation Reports absolute path.
MaxFailerCount	Maximum failure count.

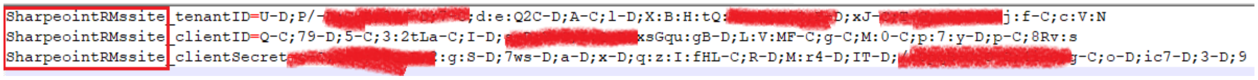


```
AutoArchiveActive=Y
OnlineSharePointSite=Y
RMSCabinetName=rms23dec
RMSServerAddress=omnidocs.newgendocker.com
RMSServerPort=443
RMSServerProtocols=https
RMSUserName=supervisor2
RMSPassword=:137-C;q;J-D;lX-C;6
sharePointUserDomain=vwm55.onmicrosoft.com
sharePointHostIPAddress=vwm55.onmicrosoft.com
shraePointPort=443
sharePointProtocols=https
sharePointSiteAbsoluteURL=
sharePointUser=spadmin@vwm55.onmicrosoft.com
sharePointSequence=:X-D;Y-D;N-C;VSJ-C;4T-C;r
sharePointSite=SharpeointRMssite
SharePointReconciliationReports=Y
SharePointReportsLibrary=/site/SharpeointRMssite/SharePointRMSReport/
MaxFailerCount=3
```

Figure 3.45

3. Update **site name**, **tenantID**, **ClientID** and **clientSecret** in file SharePointConfig.ini located inside the *OmniDocs11.0Web\Newgen\NGConfig\AddInsConfig* folder at the mapped location on the Worker node.

   For Example,



```
SharpeointRMssite_tenantID=U-D;P/-[redacted];d:e:Q2C-D;A-C;1-D;X:B:H:tQ:[redacted]D;xJ-[redacted]j:f-C;c:V:N
SharpeointRMssite_clientID=Q-C;79-D;5-C;3:2tLa-C;I-D;[redacted]xsGqu:gB-D;L:V:MF-C;g-C;M:0-C;p:7:y-D;p-C;8Rv:s
SharpeointRMssite_clientSecret[redacted]:g:S-D;7ws-D;a-D;x-D;q:z:I:fHL-C;R-D;M:r4-D;IT-D;[redacted]g-C;o-D;ic7-D;3-D;9
```

2.8 Deploying containers

This section explains deployment of containers.

- Deployment of containers on OpenShift Container Platform can be done from the local machine or bastion server by executing the below command or deploy them using Jenkins pipeline. It is recommended to deploy the containers using Jenkins pipeline for better traceability.

  ```
  oc apply -f <YAML_File>
  ```

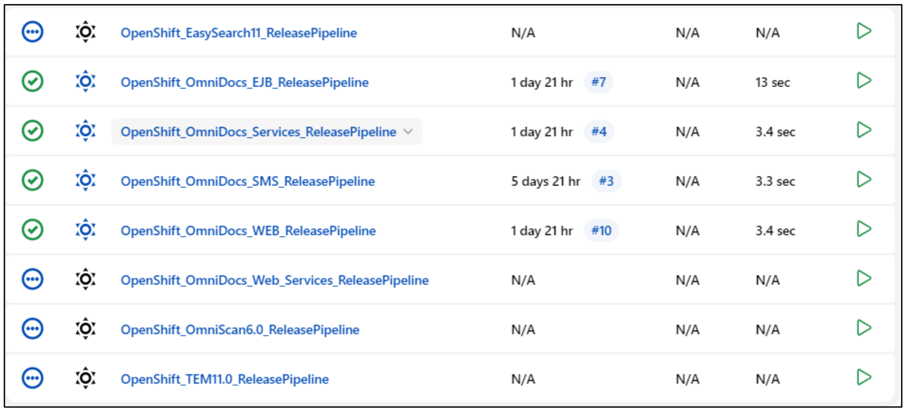
  For example**,**

  ```
  oc apply -f OmniDocs11.0Web.yml
  ```

NOTE:

- To execute the above command, OpenShift CLI (oc) must be configured on your local server or bastion server.
- To deploy the containers using Jenkins pipeline, refer to the chapter "**3 Configuration of Jenkins Release Pipeline**".

- In Jenkins, a separate Release pipeline gets created for each Container images like:
  - OmniDocs Web
  - OmniDocs EJB
  - OmniDocs Web Services
  - OmniDocs Services
  - EasySearch
  - TEM
  - OmniScan
  - OmniDocs SMS
  - RMSSharePointAdapter
  - Wopi



⋯	OpenShift_EasySearch11_ReleasePipeline	N/A		N/A	N/A	▷
✓	OpenShift_OmniDocs_EJB_ReleasePipeline	1 day 21 hr	#7	N/A	13 sec	▷
✓	OpenShift_OmniDocs_Services_ReleasePipeline	1 day 21 hr	#4	N/A	3.4 sec	▷
✓	OpenShift_OmniDocs_SMS_ReleasePipeline	5 days 21 hr	#3	N/A	3.3 sec	▷
✓	OpenShift_OmniDocs_WEB_ReleasePipeline	1 day 21 hr	#10	N/A	3.4 sec	▷
⋯	OpenShift_OmniDocs_Web_Services_ReleasePipeline	N/A		N/A	N/A	▷
⋯	OpenShift_OmniScan6.0_ReleasePipeline	N/A		N/A	N/A	▷
⋯	OpenShift_TEM11.0_ReleasePipeline	N/A		N/A	N/A	▷

Figure 2.41

- Trigger the Release Pipeline to deploy the required Container containers.
- Once the deployment is done, deployed containers can be visible from the OpenShift Console.
- In any case, to restart the container then there are two options either redeploy the container from Jenkins Release Pipeline which launches the new container by following up the rolling update feature of Kubernetes or execute the restart command from Kubernetes' pod's shell.
- The restart command is different for each container.

Container Name	Restart Command
OmniDocs11.0Web, OmniDocs11.0WebService	restartjws.sh
OmniDocs11.0EJB	restartjboss.sh

Container Name	Restart Command
OmniDocs11.0Services	restartalarm.sh, restartauthmgr.sh, restartscheduler.sh,restartthumbnail.sh, restartwrapper.sh
EasySearch11.0	restartapache.sh
TEM11.0	restarttem.sh
OmniDocs11.0SMS	restartsms.sh
OmniScanWeb6.0	restartjws.sh
RMSSharePointAdapter	restartsharepointadapter.sh
Wopi	restartjws.sh

- Once the EasySearch11 container is deployed, execute the below command in Kubernetes pod's shell for the 1st time to configure the Apache Manifold jobs. After that in subsequent deployments, this execution is not required.

```
runESConfigurator.sh
```

2.9 Creating a cabinet and data source

Prerequisites:

- OmniDocs+RMS Web, OmniDocs+RMS EJB, and OmniDocsServices are already deployed.
- Ingress Controller is already configured and deployed using the *IngressController.yml* file.

Once the above prerequisites are fulfilled, follow the below steps to create the Cabinet and Data Source.

- Getting started with OSA
- Register JTS Server
- Connecting OSA to the JTS Server
- Creating a Cabinet
- Associating the Cabinet
- Creating a Data Source
- Registration of the Cabinet
- Registering a cabinet in RMS
- Creating Site and Volume

2.9.1 Getting started with OSA

Perform the below steps to start the OSA:

1. Since the container is a CLI-based deployment you can't launch any GUI-based application inside the container. But you must use the OSA to create a cabinet that is a GUI-based application. In

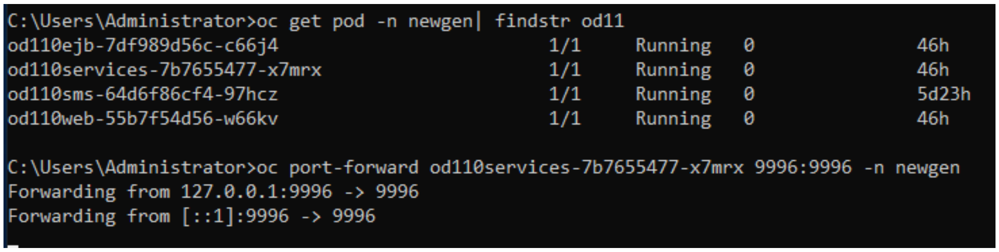
such a case, deploy OSA to some GUI-based machine either on a local server or bastion server along with OpenShift CLI (oc) installation on that bastion server.

2. Once OpenShift CLI is configured on the bastion server and it gets authenticated to access the POD details, execute the below command to get local IP address and port number of the OmniDocs11.0 Services container.

   oc port-forward <ODServices_POD_NAME> 9996:9996 –n <namespace>

   for example,

   oc port-forward od110services-7b7655477-x7mrx 9996:9996 –n newgen



```
C:\Users\Administrator>oc get pod -n newgen| findstr od11
od110ejb-7df989d56c-c66j4                        1/1     Running   0              46h
od110services-7b7655477-x7mrx                    1/1     Running   0              46h
od110sms-64d6f86cf4-97hcz                        1/1     Running   0              5d23h
od110web-55b7f54d56-w66kv                        1/1     Running   0              46h

C:\Users\Administrator>oc port-forward od110services-7b7655477-x7mrx 9996:9996 -n newgen
Forwarding from 127.0.0.1:9996 -> 9996
Forwarding from [::1]:9996 -> 9996
```

Figure 2.42

3. Once OSA is deployed on a machine, navigate to the OSA folder on that machine and double click on RunAdmin.bat (For Windows) or RunAdmin.sh (For Linux) to start OSA.
4. When the application is launched. The Login dialog appears.

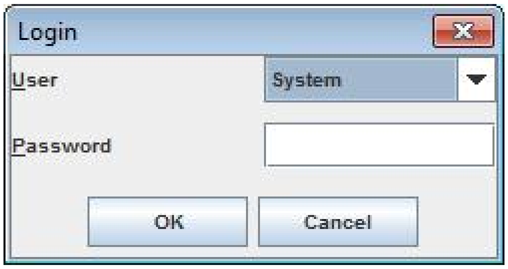
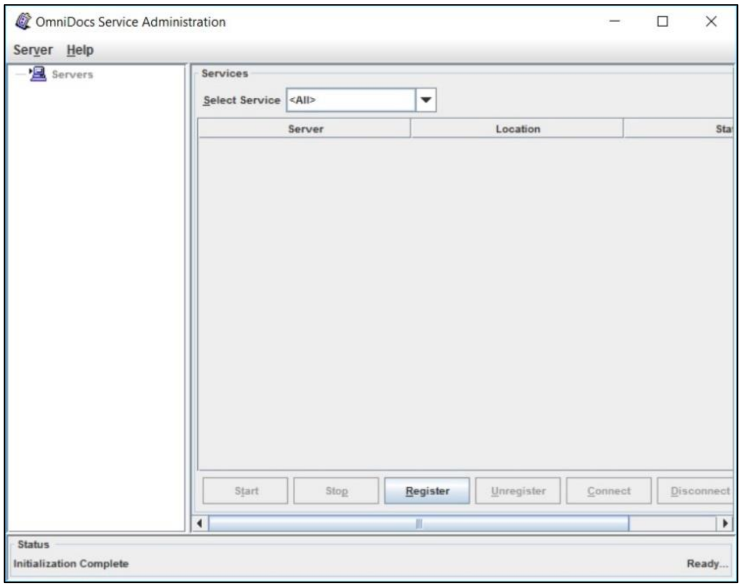


Figure 2.43

5. Select the user as **System** and specify the password as **system**.
6. Click **OK** to log in. After the successful login, the OSA screen appears displaying the list of registered services.



Initialization Complete

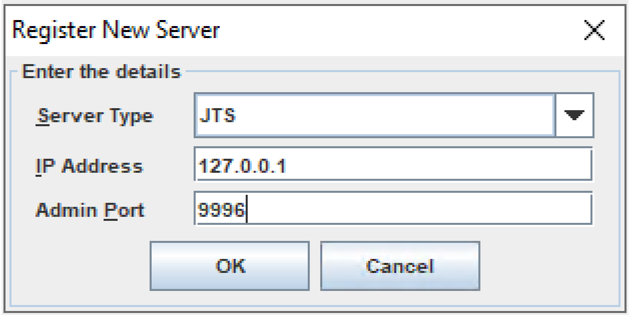
Ready...

Figure 2.44

2.9.2 Registering JTS server

Perform the below steps to register the JTS Server:

1. To register the JTS server, click **Register** button. The **Register New Server** dialog appears.
2. Specify the following details to register the SMS service.
   - **Server Type**: JTS
   - **IP Address**: 127.0.0.1
   - **Admin Port**: 9996



Register New Server

Enter the details

Server Type: JTS

IP Address: 127.0.0.1

Admin Port: 9996

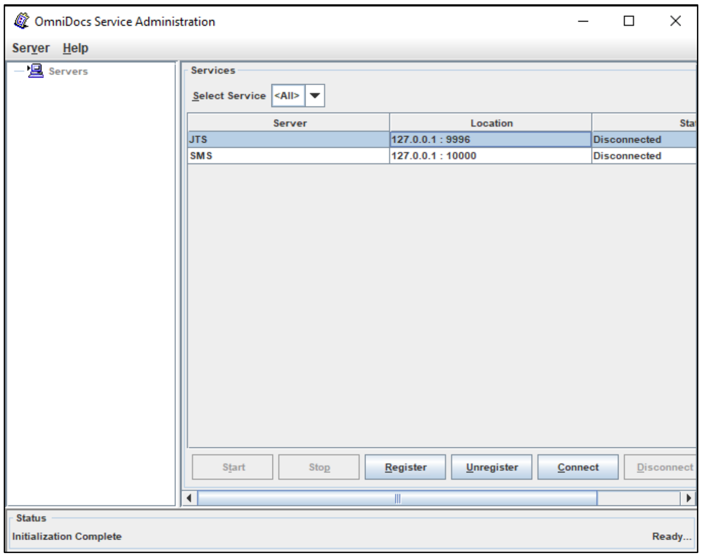
Figure 2.45

3. Click **OK** to register the JTS Server.

2.9.3 Connecting OSA to the JTS Server

Perform the below steps to connect the OSA to the JTS Server:

1. Once the JTS Server is registered, it is displayed in the list in a disconnected state.

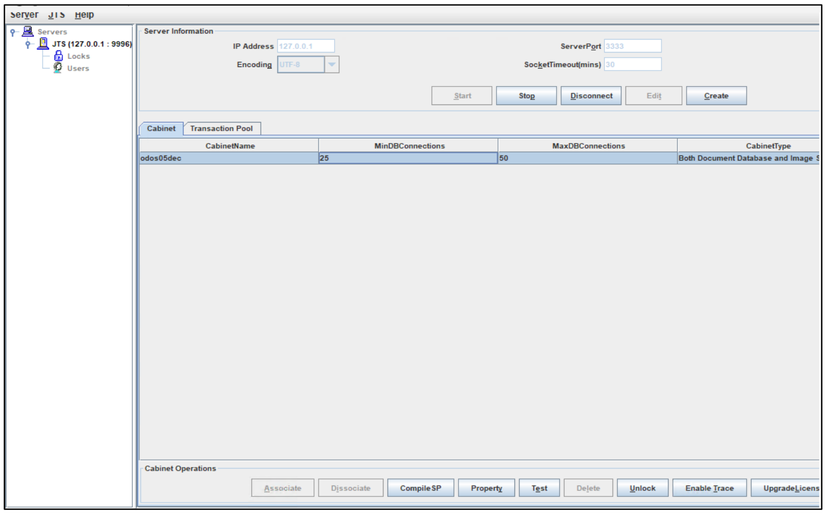


Server	Location	Sta
JTS	127.0.0.1 : 9996	Disconnected
SMS	127.0.0.1 : 10000	Disconnected

Initialization Complete Ready...

Figure 2.46

2. Select the registered JTS Server and click **Connect**. Once JTS is connected, the **Manage** button gets enabled.
3. Click **Manage** button, after clicking on the Manage button, an entry of the connected JTS server along with its IP Address is displayed on the upper-left panel in the repository view.
4. Select the JTS from the repository view. The list of already created and associated cabinets, appears.



IP Address 127.0.0.1
Encoding UTF-8
ServerPort 3333
SocketTimeout(mins) 30

CabinetName	MinDBConnections	MaxDBConnections	CabinetType
odos05dec	25	50	Both Document Database and Image S

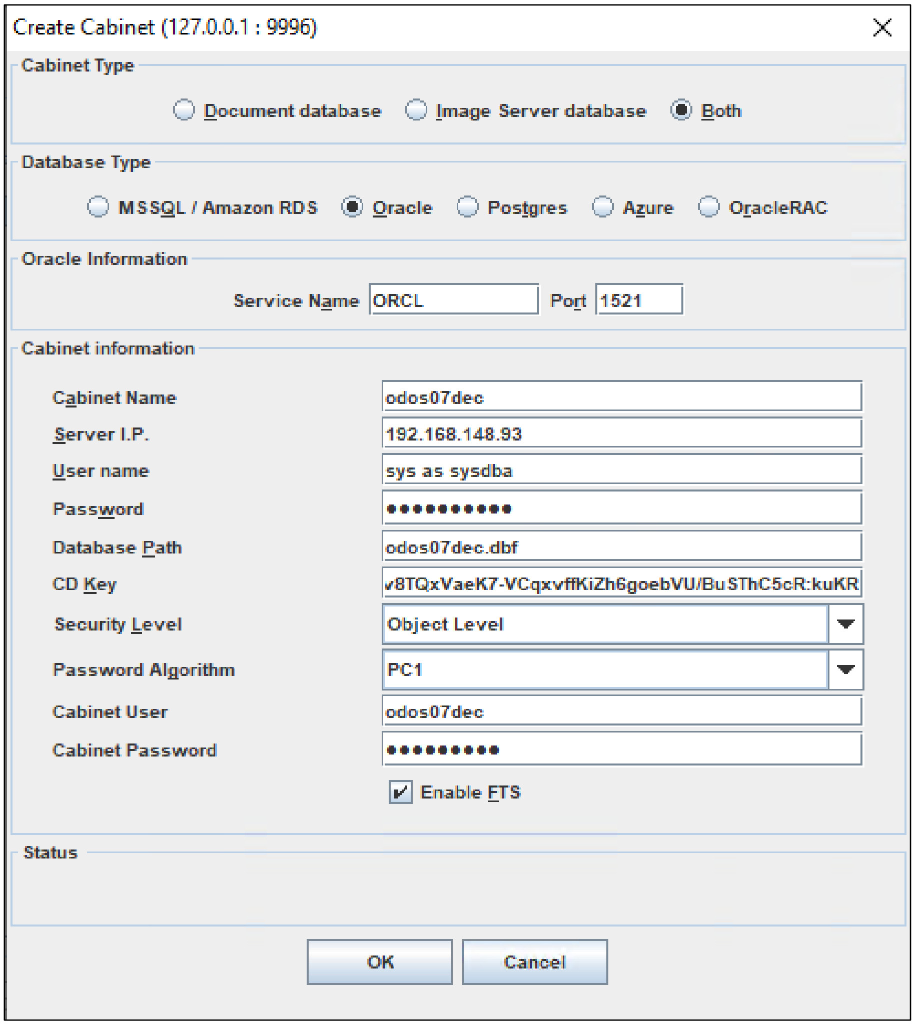
Figure 2.47

2.9.4 Creating a cabinet

Perform the below steps to create a cabinet:

For Oracle:

1. Click **Create**. The Create Cabinet dialog appears.



Create Cabinet (127.0.0.1 : 9996)

Cabinet Type

○ Document database ○ Image Server database ● Both

Database Type

○ MSSQL / Amazon RDS ● Oracle ○ Postgres ○ Azure ○ OracleRAC

Oracle Information

Service Name ORCL Port 1521

Cabinet information

Cabinet Name	odos07dec
Server I.P.	192.168.148.93
User name	sys as sysdba
Password	••••••••••
Database Path	odos07dec.dbf
CD Key	v8TQxVaeK7-VCqxvffKiZh6goebVU/BuSThC5cR:kuKR
Security Level	Object Level
Password Algorithm	PC1
Cabinet User	odos07dec
Cabinet Password	•••••••••

☑ Enable FTS

Status

Figure 2.48

2. Select the cabinet type that needs to be created from the Cabinet Type area. The Cabinet can be a **Document database**, **an Image server database**, or both.
3. Select the database option from the Database Type section.
4. **Oracle Information:** Specify the **Port number** of the Oracle Database Server.
5. Specify the following cabinet information:
   - Specify the cabinet name in the **Cabinet Name** textbox.
   - Specify the server name (name of the machine where the Oracle server is running) in the **Server I.P.** textbox.

- Specify the Oracle server username in the **User name** textbox.
- Specify the Oracle server password in the **Password** textbox.
- Specify the CD key in the **CD Key** textbox.
- Specify the **Cabinet User** and **Cabinet Password** for further processing as once the cabinet is created, that cabinet user and cabinet password will be used for all communications.
- Select the **Enable FTS** checkbox.

6. Click **OK** to create the cabinet. The Cabinet created successfully dialog appears.

2.9.5 Associating a cabinet

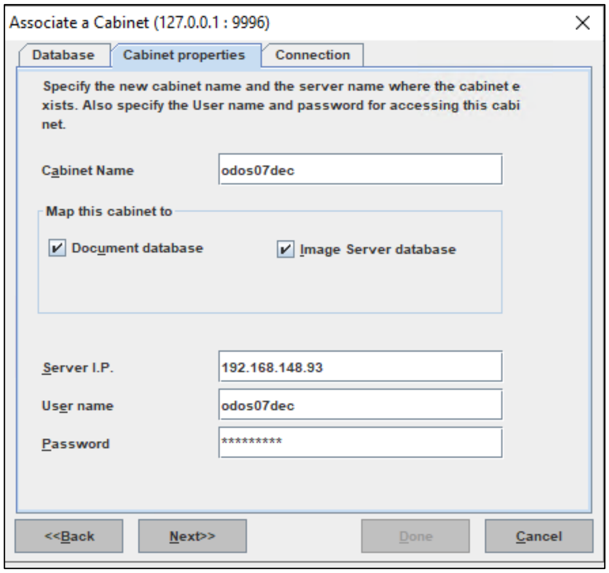
Perform the below steps to associate the cabinet:

For Oracle:

1. Click **Stop** to enable the Associate button.
2. Click **Associate**. The Associate a Cabinet dialog appears with the following tabs:

   i. **Database tab:** Select the database type and specify Oracle server port and service name.

   ii. **Cabinet properties tab:** Specify the cabinet details that you have specified during cabinet creation.



Specify the new cabinet name and the server name where the cabinet exists. Also specify the User name and password for accessing this cabinet.

Cabinet Name: odos07dec

Map this cabinet to

☑ Document database ☑ Image Server database

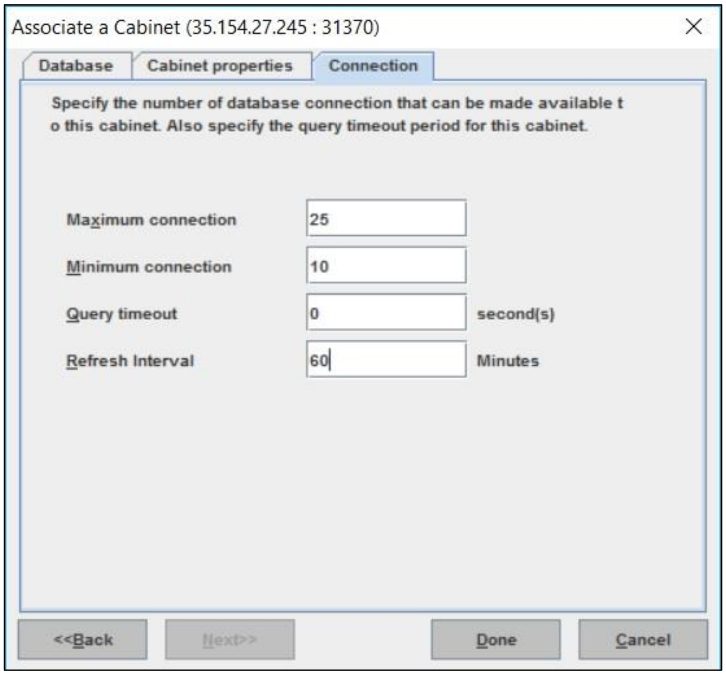
Server I.P.: 192.168.148.93

User name: odos07dec

Password: *********

Figure 2.49

   iii. **Connection tab:** Specify the **maximum** and the **minimum** number of connections that the JTS must maintain with the database, specify the **query time** out for the selected cabinet in the Query timeout text box and specify the **refresh interval** time for connection.

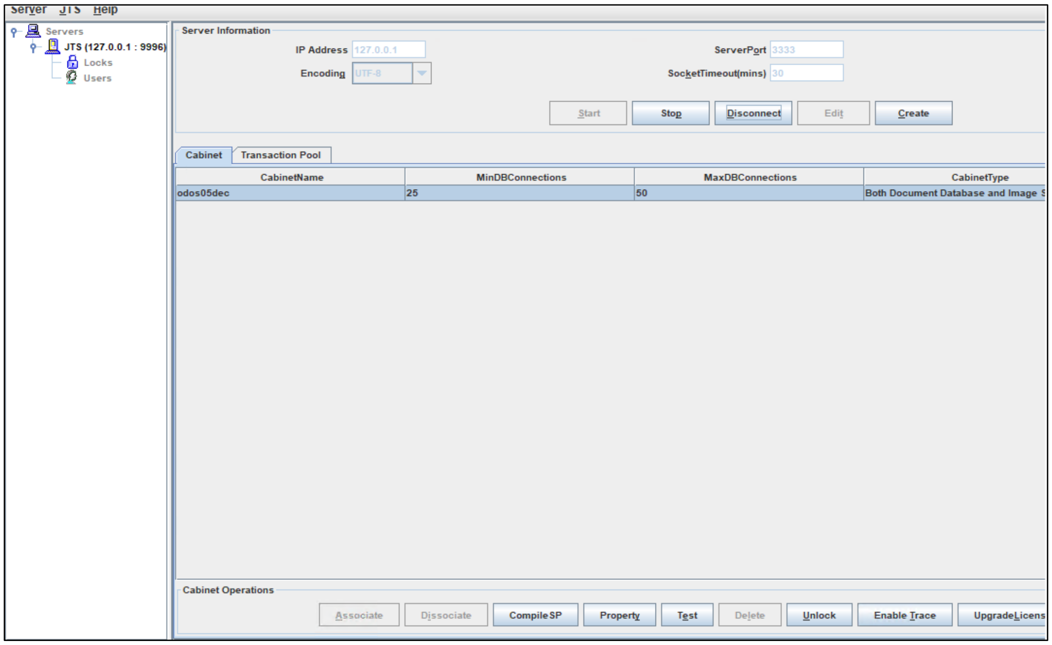


Specify the number of database connection that can be made available t o this cabinet. Also specify the query timeout period for this cabinet.

Maximum connection	25	
Minimum connection	10	
Query timeout	0	second(s)
Refresh Interval	60	Minutes

Figure 2.50

3. Click **Done** to associate the selected cabinet. Once the cabinet is associated successfully, it appears with the list.



IP Address 127.0.0.1

ServerPort 3333

Encoding UTF-8

SocketTimeout(mins) 30

CabinetName	MinDBConnections	MaxDBConnections	CabinetType
odos05dec	25	50	Both Document Database and Image S

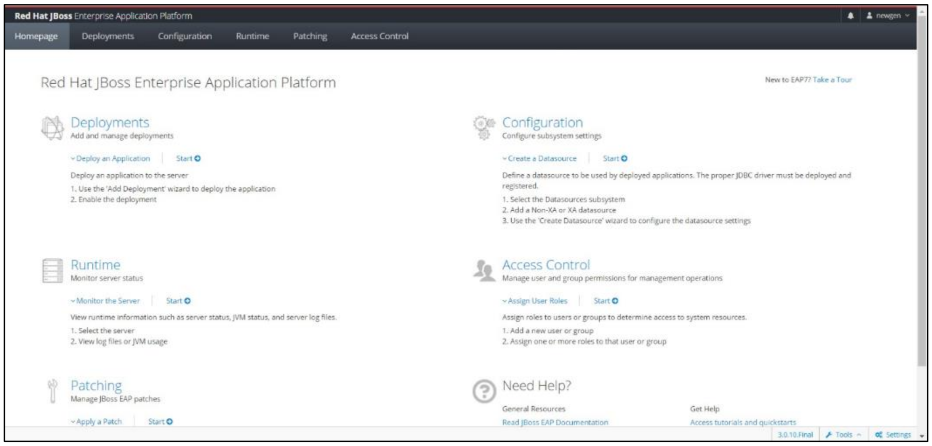
Figure 2.51

2.9.6 Creating a data source

Perform the below steps to create the data source:

For Oracle:

1. Open the<Host-Path URL of OmniDocs+RMS EJB container> like *http://OmniDocsconsole*.apps.newgenopenshiftcluster.newgensoftware.net as defined in the *IngressController.yml* file. It automatically redirects to the JBoss EAP 7.4 Admin console.
2. Enter the newgen as username and password system123# respectively to login to the Admin console. After a successful login, the Red Hat JBoss Enterprise Application Platform screen appears.



Deploy an application to the server

1. Use the 'Add Deployment' wizard to deploy the application
2. Enable the deployment

Define a datasource to be used by deployed applications. The proper JDBC driver must be deployed and registered.

1. Select the Datasources subsystem
2. Add a Non-XA or XA datasource
3. Use the 'Create Datasource' wizard to configure the datasource settings

View runtime information such as server status, JVM status, and server log files.

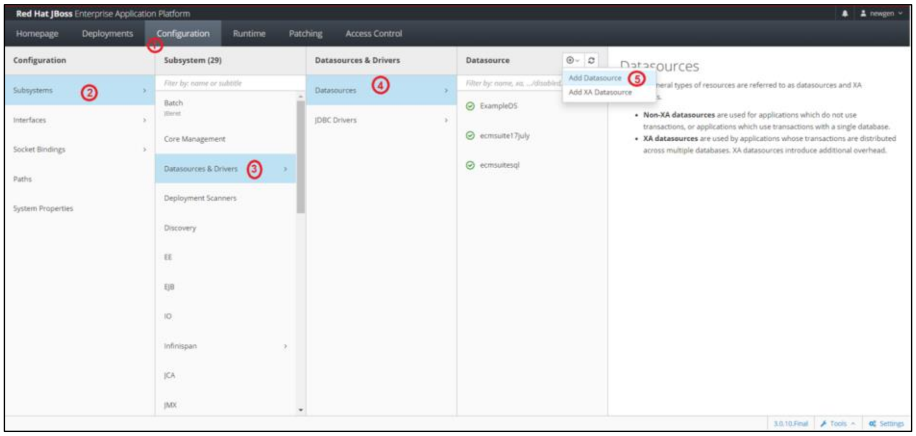
1. Select the server
2. View log files or JVM usage

Assign roles to users or groups to determine access to system resources.

1. Add a new user or group
2. Assign one or more roles to that user or group

Figure 2.52

3. Go to the **Subsystems** in the Configuration tab.
4. Go to the **Datasources & Drivers**. Then, click Datasources.

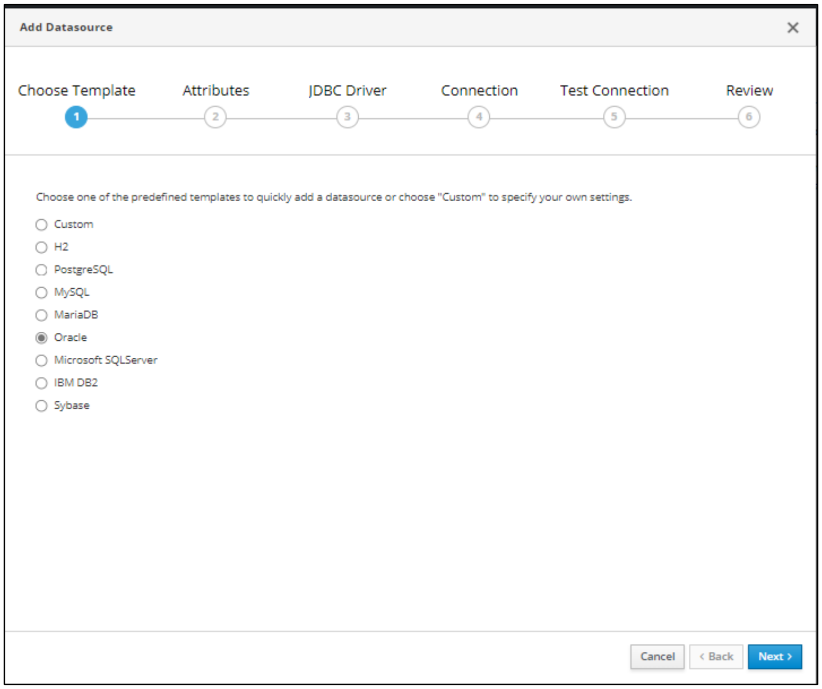


...neral types of resources are referred to as datasources and XA ...s.

- **Non-XA datasources** are used for applications which do not use transactions, or applications which use transactions with a single database.
- **XA datasources** are used by applications whose transactions are distributed across multiple databases. XA datasources introduce additional overhead.

Figure 2.53

5. Click Plus **+** icon and select **Add Datasource**. The Add Datasource dialog appears.
6. For Oracle Database Server, select **Oracle** and click **Next**.

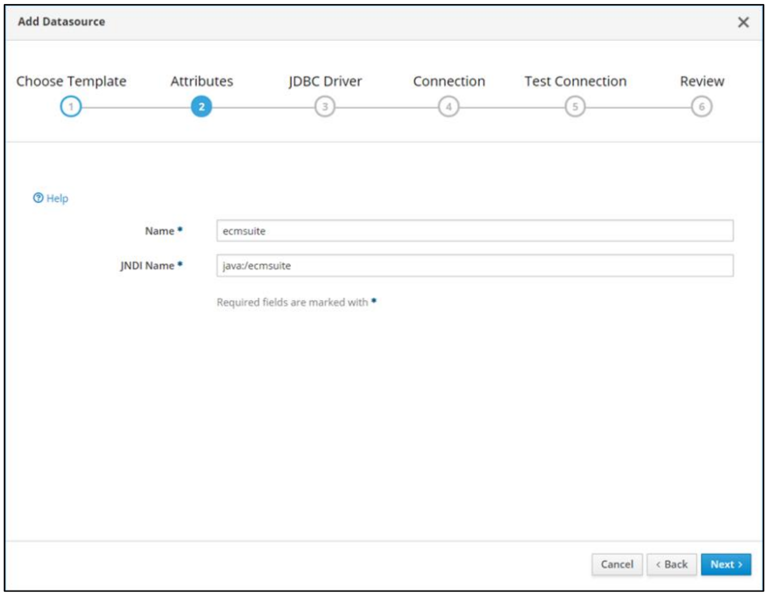


Choose one of the predefined templates to quickly add a datasource or choose "Custom" to specify your own settings.

○ Custom
○ H2
○ PostgreSQL
○ MySQL
○ MariaDB
● Oracle
○ Microsoft SQLServer
○ IBM DB2
○ Sybase

Figure 2.54

7. Provide a DataSource Name and JNDI Name.
   - **Name:** Enter the OmniDocs cabinet name that is cabinet name.
   - **JNDI Name:** java:/same as OmniDocs cabinet name
8. Click **Next**.



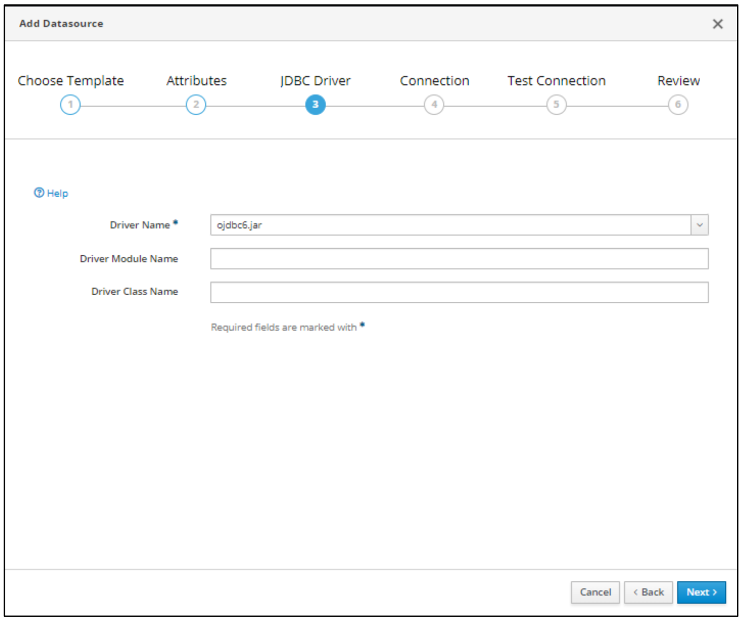
Name * ecmsuite

JNDI Name * java:/ecmsuite

Required fields are marked with *

Figure 2.55

9. Select JDBC Driver Name.
10. For Oracle, select **ojdbc6.jar**.
11. Clear **Drive Module Name** and **Driver Class Name** textboxes.
12. Click **Next**.



Add Datasource

Driver Name * ojdbc6.jar

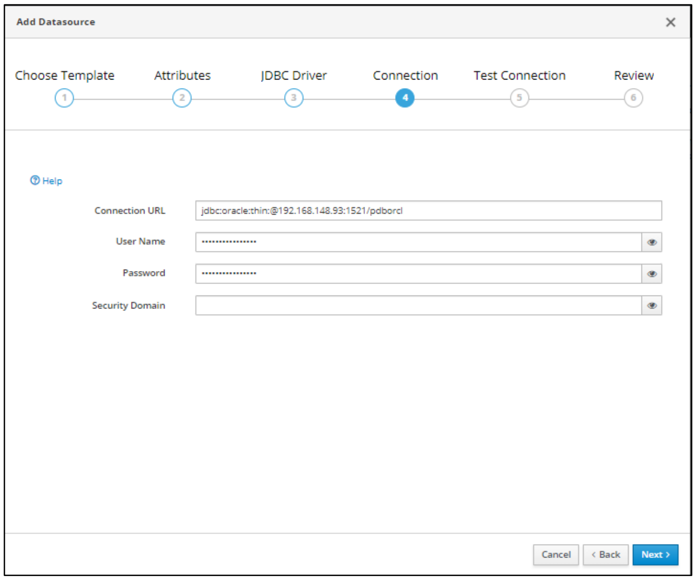
Driver Module Name

Driver Class Name

Required fields are marked with *

Figure 2.56

13. Provide the following Connection Setting details and click **Next**:
- **Connection URL:** jdbc:oracle:thin:@Oracle_Server_IP:Oracle_Server_Port/Oracle_Servicename
- **UserName:** Oracle DB User Name
- **Password:** Oracle DB Password
- **Security Domain:** Keep this blank.



Connection URL: jdbc:oracle:thin:@192.168.148.93:1521/pdborcl

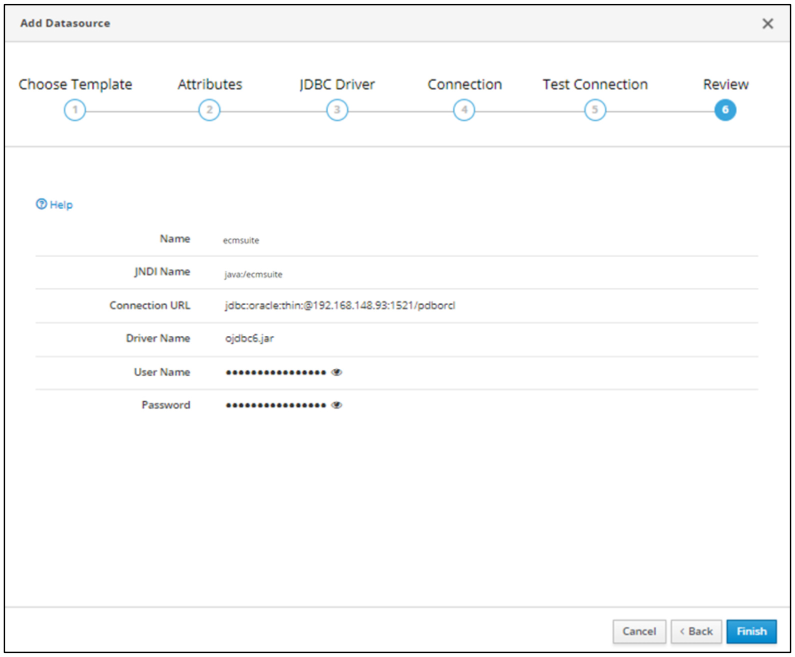
User Name: ••••••••••••••••

Password: ••••••••••••••••

Security Domain:

Figure 2.57

14. Click **Next** on the **Test Connection** page.
15. Click **Finish.** After the creation of the datasource, a success message appears.

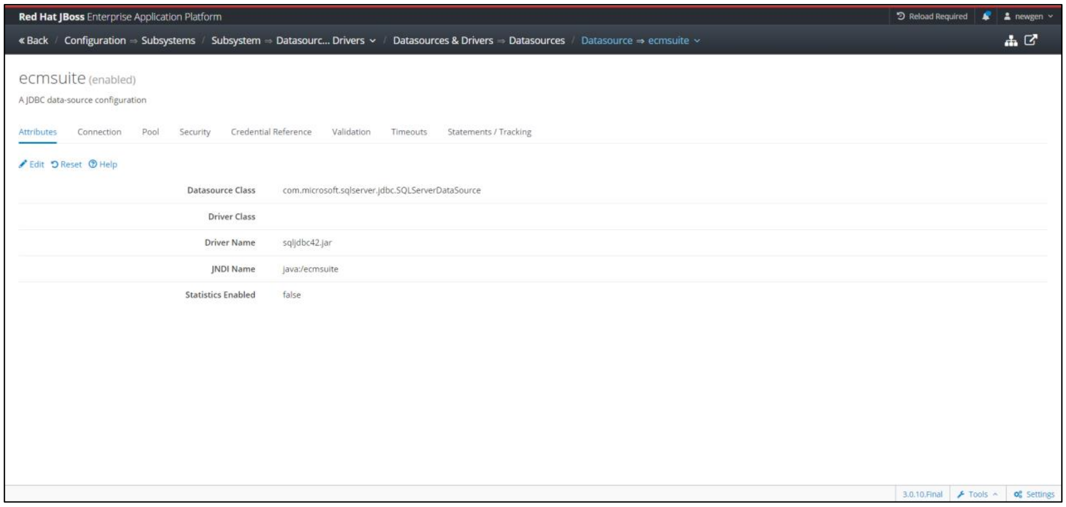


Name	ecmsuite
JNDI Name	java:/ecmsuite
Connection URL	jdbc:oracle:thin:@192.168.148.93:1521/pdborcl
Driver Name	ojdbc6.jar
User Name	••••••••••••••••
Password	••••••••••••••••

Figure 2.58

16. Click V**iew Datasource** to view the created datasource. The created datasource appears in the list of **Datasource**.
17. Click **View** against the datasource. A screen appears with the attributes of the datasource appears.

18. Click **Edit** link.



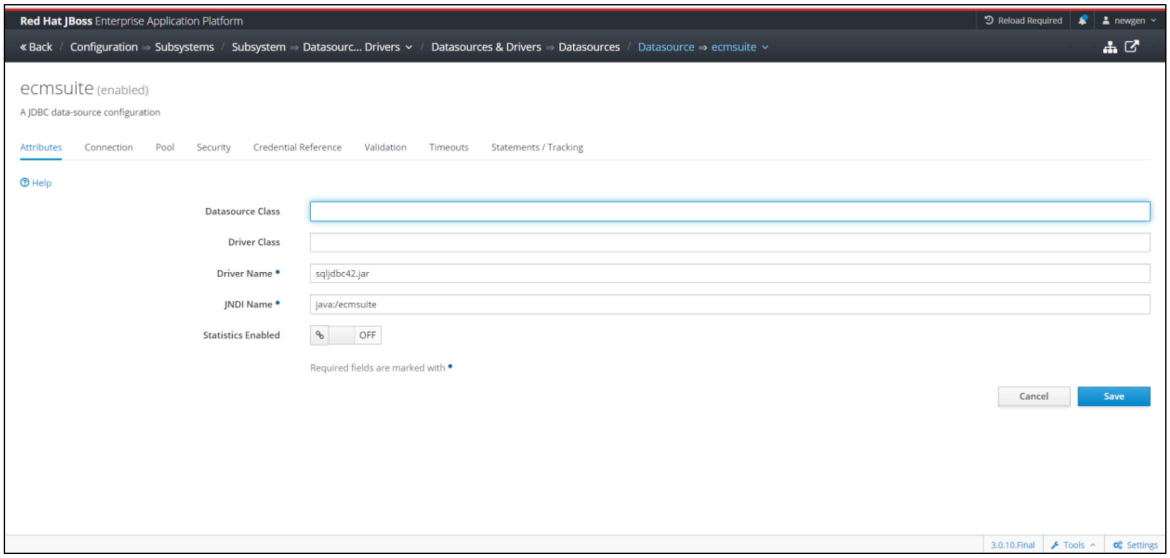
ecmsuite (enabled)

A JDBC data-source configuration

Datasource Class	com.microsoft.sqlserver.jdbc.SQLServerDataSource
Driver Class	
Driver Name	sqljdbc42.jar
JNDI Name	java:/ecmsuite
Statistics Enabled	false

Figure 2.59

19. Clear the **Datasource Class** textbox and click **Save**.



ecmsuite (enabled)

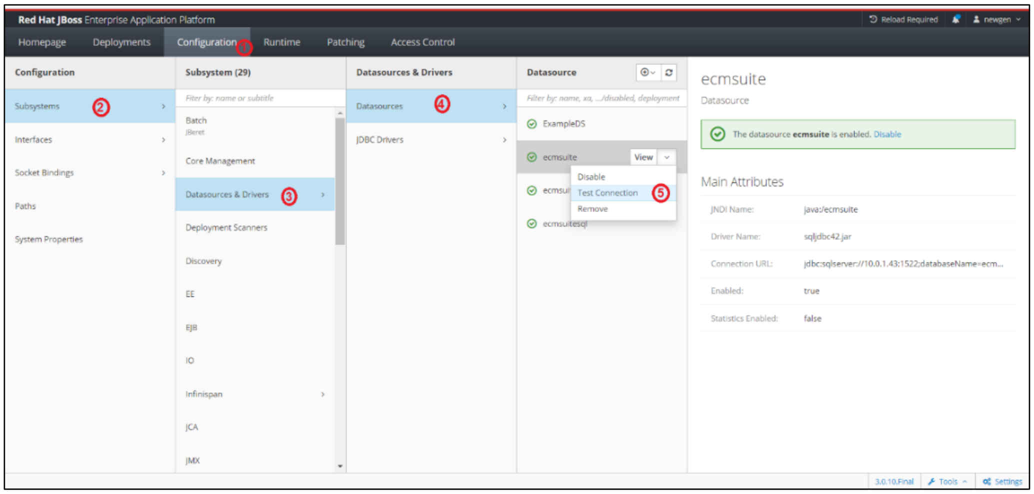
A JDBC data-source configuration

Datasource Class	
Driver Class	
Driver Name *	sqljdbc42.jar
JNDI Name *	java:/ecmsuite
Statistics Enabled	OFF

Required fields are marked with *

Figure 2.60

20. After that restart the OmniDocs+RMS EJB container.
21. Once the OmniDocs+RMS EJB container is restarted, open the JBossEAP Admin console once again.
22. Go to the **Subsystems** in the Configuration tab.
23. Go to the **Datasources & Drivers**. Then, click Datasources.
24. Select the created data source and click **Test connection** from the dropdown list. On the successful data connection, a success message appears.



The datasource **ecmsuite** is enabled. Disable

Main Attributes

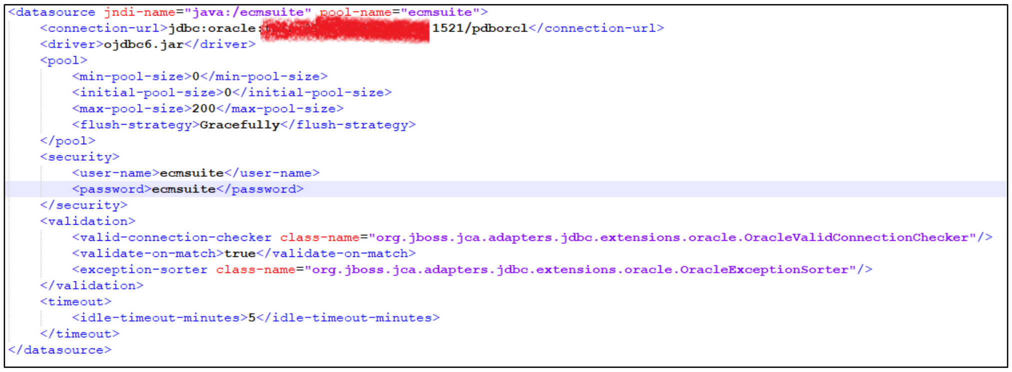
JNDI Name:	java:/ecmsuite
Driver Name:	sqljdbc42.jar
Connection URL:	jdbc:sqlserver://10.0.1.43:1522;databaseName=ecm...
Enabled:	true
Statistics Enabled:	false

Figure 2.61

25. Add the below connection pool setting and idle-connection-timeout setting inside the created DataSource in *standalone.xml* file located inside the **OmniDocs11.0Ejb** or **configuration** folder kept inside the mount path on NAS server.

```
<pool>
      <min-pool-size>100</min-pool-size>
      <initial-pool-size>100</initial-pool-size>
      <max-pool-size>600</max-pool-size>
      <flush-strategy>Gracefully</flush-strategy>
</pool>
<timeout>
      <idle-timeout-minutes>5</idle-timeout-minutes>
</timeout>
```

For example,



```
<datasource jndi-name="java:/ecmsuite" pool-name="ecmsuite">
    <connection-url>jdbc:oracle:[redacted]1521/pdborcl</connection-url>
    <driver>ojdbc6.jar</driver>
    <pool>
        <min-pool-size>0</min-pool-size>
        <initial-pool-size>0</initial-pool-size>
        <max-pool-size>200</max-pool-size>
        <flush-strategy>Gracefully</flush-strategy>
    </pool>
    <security>
        <user-name>ecmsuite</user-name>
        <password>ecmsuite</password>
    </security>
    <validation>
        <valid-connection-checker class-name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker"/>
        <validate-on-match>true</validate-on-match>
        <exception-sorter class-name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter"/>
    </validation>
    <timeout>
        <idle-timeout-minutes>5</idle-timeout-minutes>
    </timeout>
</datasource>
```

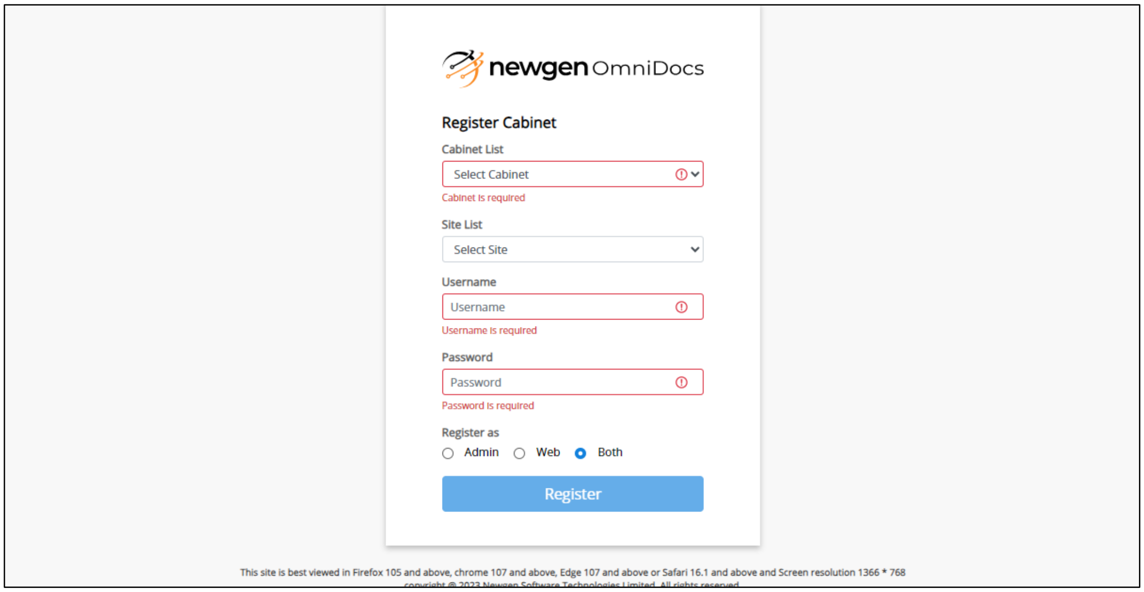
Figure 2.62

26. Restart the **OmniDocs+RMS EJB** container once again.

2.9.7 Registering a cabinet in OmniDocs

Perform the below steps to register a cabinet:

1. Register the cabinet for OmniDocs Admin using the following URL:
   http:*//<Host-Path URL of* OmniDocsWeb container>/OmniDocs/register
   For example,
   http://ecmsuite.newgendocker.com /OmniDocs/register



Cabinet is required

Username is required

Password is required

Register as: Both

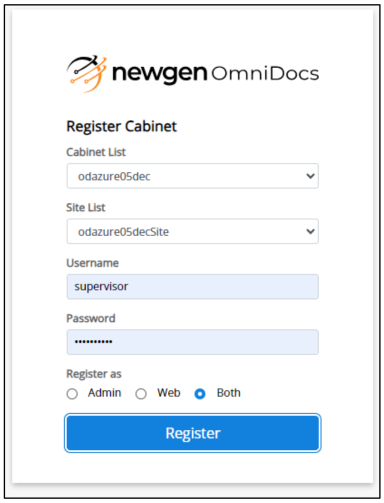
This site is best viewed in Firefox 105 and above, chrome 107 and above, Edge 107 and above or Safari 16.1 and above and Screen resolution 1366 * 768

copyright @ 2023 Newgen Software Technologies Limited. All rights reserved.

Figure 2.63

   All the created cabinets get auto populated in the **Cabinet List** dropdown list.

2. Select the required cabinet, select the associated site, and specify the **Username** and **Password**.
3. Select the Register as **Both** and click **Register.** After successful registration, a confirmation message appears.



Register Cabinet

Cabinet List: odazure05dec

Site List: odazure05decSite

Username: supervisor

Password: ••••••••••

Register as ○ Admin ○ Web ● Both

Figure 2.64

2.9.8 Registering a cabinet in RMS

Perform the below steps to register a cabinet in RMS:

6. Configure the cabinet for RMS using the following URL:

   [http://<Host-Path]() URL of OmniDocsWeb container>/rms/config

   For example, *[http://ecmsuite.aws.co.in/rms/config]()*

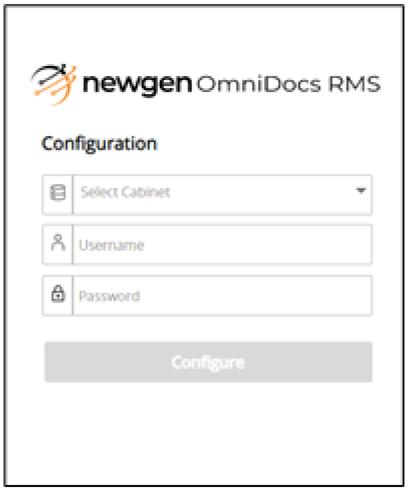
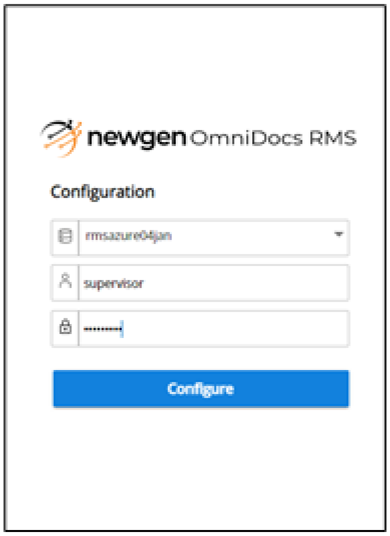


Figure 3.93

All the created cabinets get auto populated in the Cabinet List dropdown list.

7. Select the required cabinet, specify the **Username** and **Password** and click **Configure**.



Configuration

rmsazure04jan

supervisor

••••••••

Figure 3.94

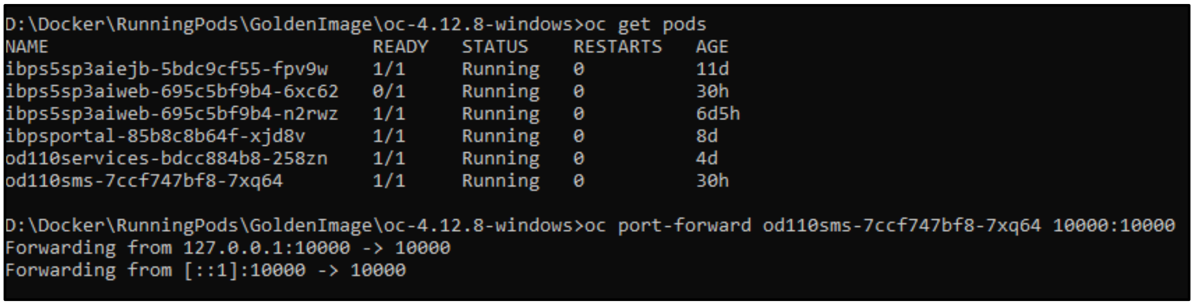
After successful registration of cabinet, a confirmation message appears.

2.9.9 Creating site and volume

To create site and volume, follow the below steps:

Create Label:

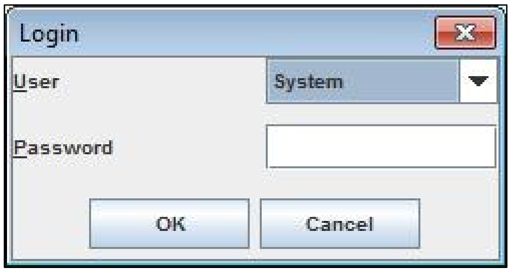
1. SMS labels can be created using **OSA** (OmniDocs Service Administration) which is a GUI based application and container is a CLI-based deployment you can't launch any GUI-based application inside the container. But you must use the OSA to create an SMS label that is a GUI-based application. In such a case, deploy OSA to some GUI-based machine either on a local server or bastion server along with **OpenShift CLI** (oc) installation on that bastion server.
2. Once OpenShift CLI is configured on the bastion server and it gets authenticated to access the POD details, execute the below command to get local IP address and port number of the SMS container.
   oc port-forward <SMS_POD_NAME> 10000:10000
   for example,
   oc port-forward od110sms-7ccf747bf8-7xq64 10000:10000



```
D:\Docker\RunningPods\GoldenImage\oc-4.12.8-windows>oc get pods
NAME                              READY   STATUS    RESTARTS   AGE
ibps5sp3aiejb-5bdc9cf55-fpv9w     1/1     Running   0          11d
ibps5sp3aiweb-695c5bf9b4-6xc62    0/1     Running   0          30h
ibps5sp3aiweb-695c5bf9b4-n2rwz    1/1     Running   0          6d5h
ibpsportal-85b8c8b64f-xjd8v       1/1     Running   0          8d
od110services-bdcc884b8-258zn     1/1     Running   0          4d
od110sms-7ccf747bf8-7xq64         1/1     Running   0          30h

D:\Docker\RunningPods\GoldenImage\oc-4.12.8-windows>oc port-forward od110sms-7ccf747bf8-7xq64 10000:10000
Forwarding from 127.0.0.1:10000 -> 10000
Forwarding from [::1]:10000 -> 10000
```

Figure 2.65

3. Once OSA is copied on bastion server, navigate to the OSA folder on that machine and double click on RunAdmin.bat (For Windows) or RunAdmin.sh (For Linux) to start OSA.
4. When the application is launched. The Login dialog appears.

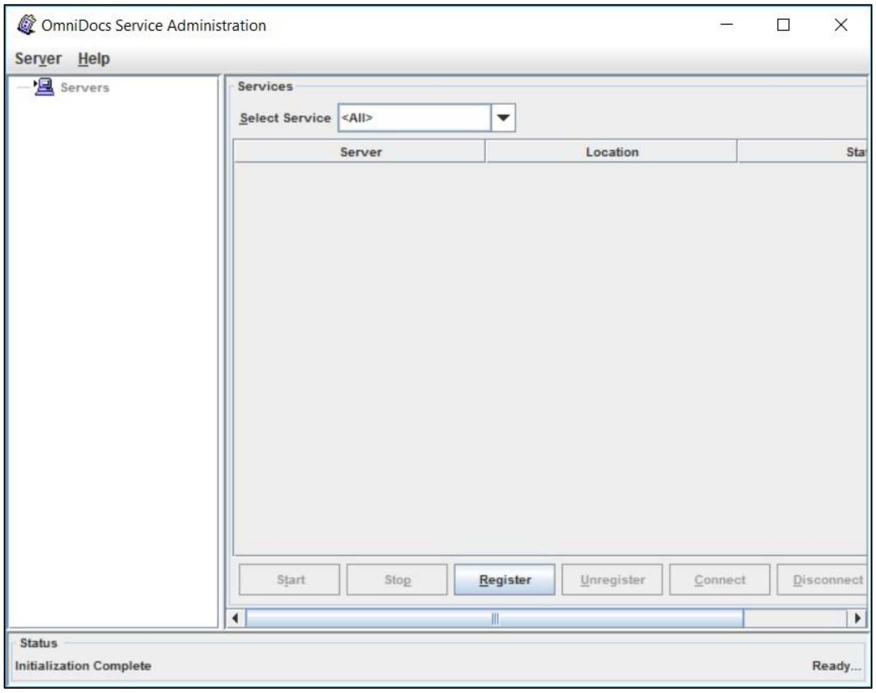


User System

Password

Figure 2.66

5. Select the user as **system** and specify the password as **system**.
6. Click **OK** to log in. After the successful login, the OSA screen appears displaying the list of registered services.

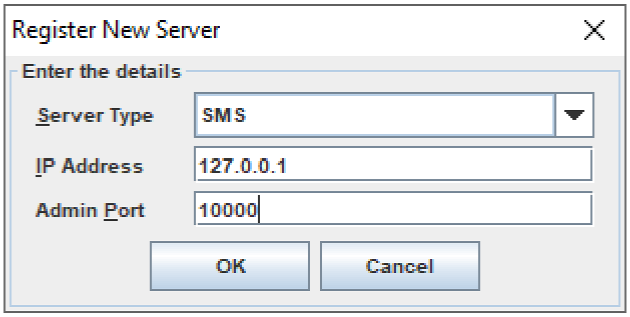


Initialization Complete

Ready...

Figure 2.67

7. Click **Register** button. The **Register New Server** dialog appears.
8. Specify the following details to register the SMS service.
   - **Server Type**: SMS
   - **IP Address**: 127.0.0.1
   - **Admin Port**: 10000
9. Click **OK**.



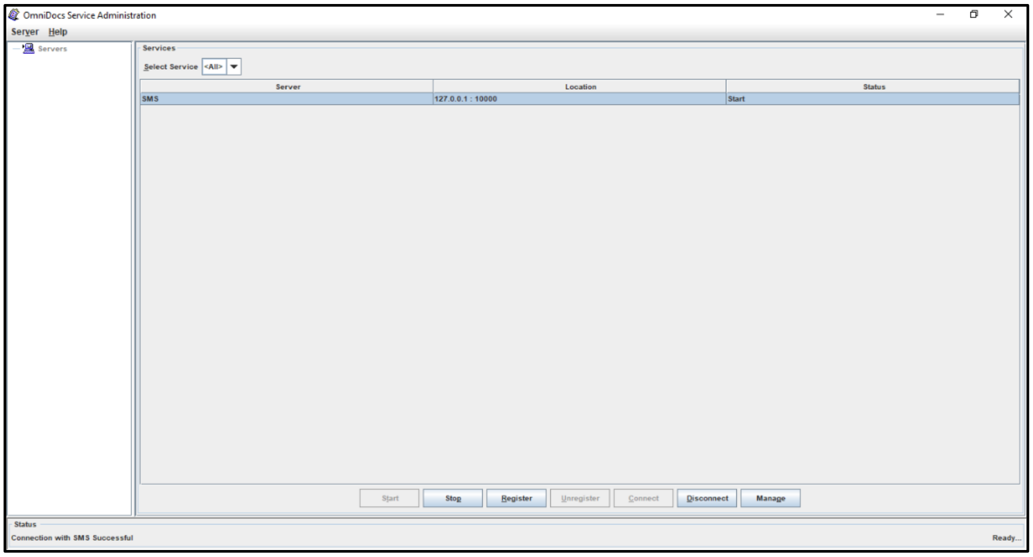
Server Type: SMS

IP Address: 127.0.0.1

Admin Port: 10000

Figure 2.68

10. Select the registered SMS service and click Connect.
11. On successful connection, ‘Manage’ button will be enabled.
12. Click on **Manage** button.



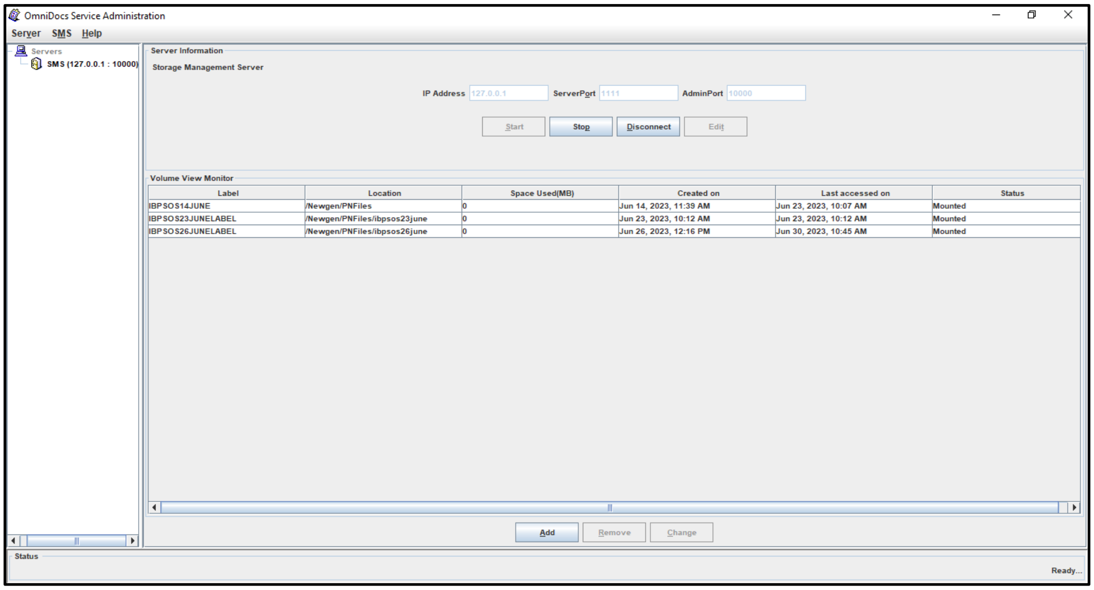
Select Service <All>

Server	Location	Status
SMS	127.0.0.1 : 10000	Start

Status
Connection with SMS Successful
Ready...

Figure 2.69

13. After click on Manage SMS service will be list out under Servers tree in Top-Left corner.
14. Click on SMS Service.
15. Click **Add** button to add a new label.



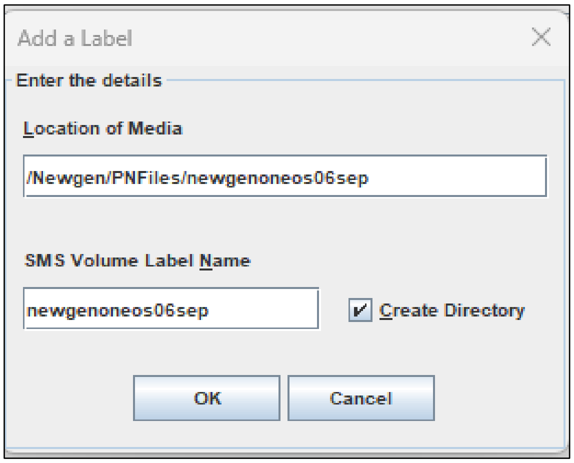
IP Address 127.0.0.1
ServerPort 1111
AdminPort 10000

Label	Location	Space Used(MB)	Created on	Last accessed on	Status
IBPSOS14JUNE	/Newgen/PNFiles	0	Jun 14, 2023, 11:39 AM	Jun 23, 2023, 10:07 AM	Mounted
IBPSOS23JUNELABEL	/Newgen/PNFiles/ibpsos23june	0	Jun 23, 2023, 10:12 AM	Jun 23, 2023, 10:12 AM	Mounted
IBPSOS26JUNELABEL	/Newgen/PNFiles/ibpsos26june	0	Jun 26, 2023, 12:16 PM	Jun 30, 2023, 10:45 AM	Mounted

Ready...

Figure 2.70

16. Specify the following details to add a new label:
    - **Location of Media:** /Newgen/PNFiles/<iBPS Cabinet name>
    - **SMS Volume Label Name**: User defined name
    - Checked the checkbox '**Create Directory'.**
17. Click **OK.**



Location of Media: /Newgen/PNFiles/newgenoneos06sep

SMS Volume Label Name: newgenoneos06sep

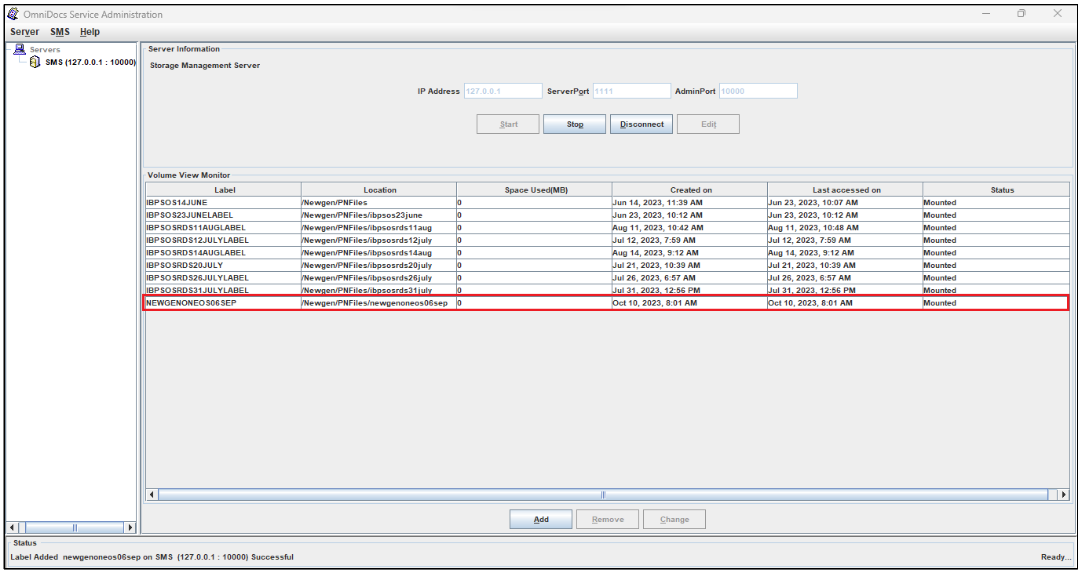
☑ Create Directory

Figure 2.71

NOTE:

Make sure the location of media must always start with **/Newgen/PNFiles** path as this path is mentioned as a mounted path in OmniDocs11.0SMS YAML file that is, 'OmniDocs11.0SMS.yml'.

18. After successfully addition of SMS label, the below screen appears.



IP Address 127.0.0.1 ServerPort 1111 AdminPort 10000

Label	Location	Space Used(MB)	Created on	Last accessed on	Status
IBPSOS14JUNE	/Newgen/PNFiles	0	Jun 14, 2023, 11:39 AM	Jun 23, 2023, 10:07 AM	Mounted
IBPSOS23JUNELABEL	/Newgen/PNFiles/ibpsos23june	0	Jun 23, 2023, 10:12 AM	Jun 23, 2023, 10:12 AM	Mounted
IBPSOSRDS11AUGLABEL	/Newgen/PNFiles/ibpsosrds11aug	0	Aug 11, 2023, 10:42 AM	Aug 11, 2023, 10:48 AM	Mounted
IBPSOSRDS12JULYLABEL	/Newgen/PNFiles/ibpsosrds12july	0	Jul 12, 2023, 7:59 AM	Jul 12, 2023, 7:59 AM	Mounted
IBPSOSRDS14AUGLABEL	/Newgen/PNFiles/ibpsosrds14aug	0	Aug 14, 2023, 9:12 AM	Aug 14, 2023, 9:12 AM	Mounted
IBPSOSRDS20JULY	/Newgen/PNFiles/ibpsosrds20july	0	Jul 21, 2023, 10:39 AM	Jul 21, 2023, 10:39 AM	Mounted
IBPSOSRDS26JULYLABEL	/Newgen/PNFiles/ibpsosrds26july	0	Jul 26, 2023, 6:57 AM	Jul 26, 2023, 6:57 AM	Mounted
IBPSOSRDS31JULYLABEL	/Newgen/PNFiles/ibpsosrds31july	0	Jul 31, 2023, 12:56 PM	Jul 31, 2023, 12:56 PM	Mounted
NEWGENONEOS06SEP	/Newgen/PNFiles/newgenoneos06sep	0	Oct 10, 2023, 8:01 AM	Oct 10, 2023, 8:01 AM	Mounted

Label Added newgenoneos06sep on SMS (127.0.0.1 : 10000) Successful

Figure 2.72

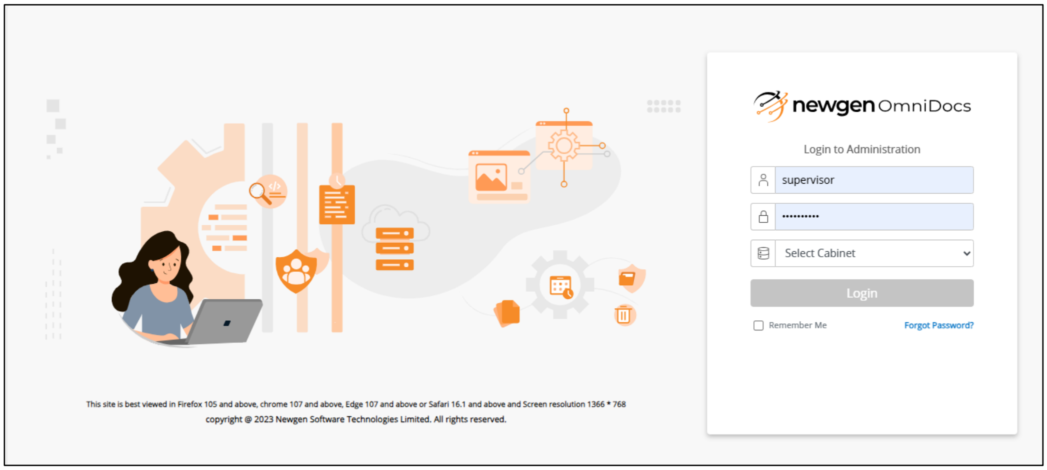
Create Site & Volume:

1. Login to the OmniDocs Admin using the following URL:

http:*//<Host-Path URL of OmniDocsWeb container>/OmniDocs/admin*

For example,

http:// OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net/OmniDocs/admin



supervisor

••••••••••

Select Cabinet

This site is best viewed in Firefox 105 and above, chrome 107 and above, Edge 107 and above or Safari 16.1 and above and Screen resolution 1366 * 768

copyright @ 2023 Newgen Software Technologies Limited. All rights reserved.

Figure 2.73

2. After a successful login, click **Sites** link under **Administration**.

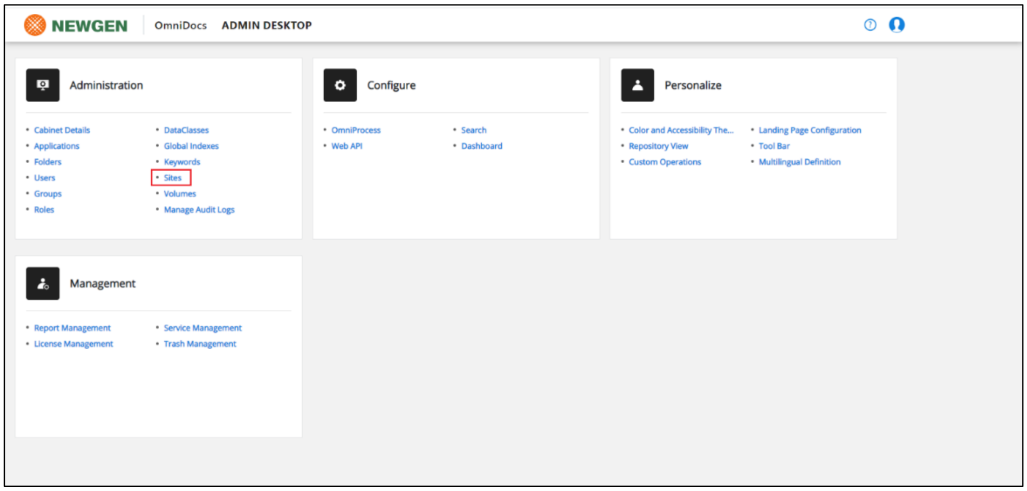


Figure 2.74

3. Click **+Add**. The Add Site dialog appears.

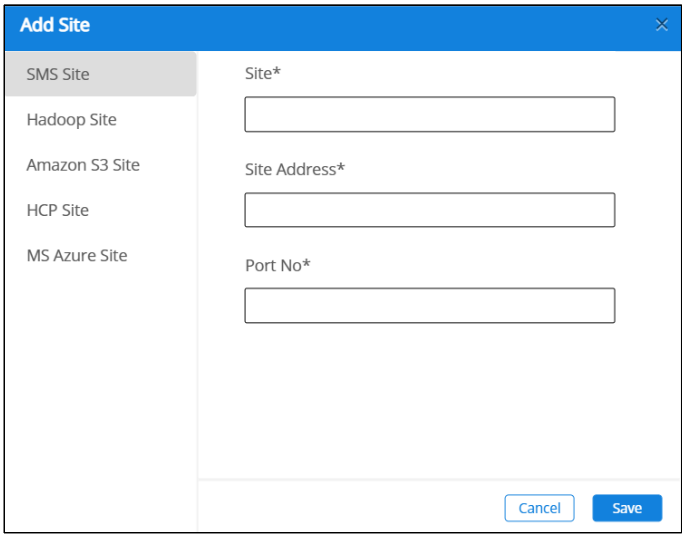
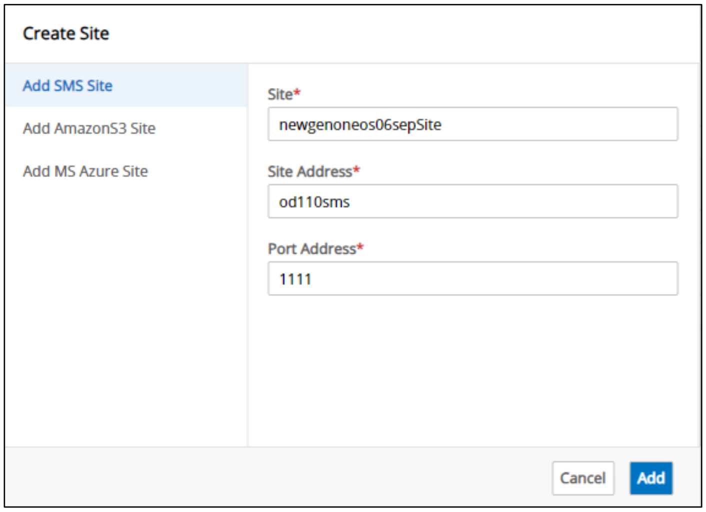


Figure 2.75

4. Specify the following details:
   - **Site:** Specify the user defined name.
   - **Site Address:** Specify the OD11.0SMS container’s service name [Defined in respective YAML file].
   - **Port Address:** Specify the SMS client port that is, 1111
5. Click **Save**.



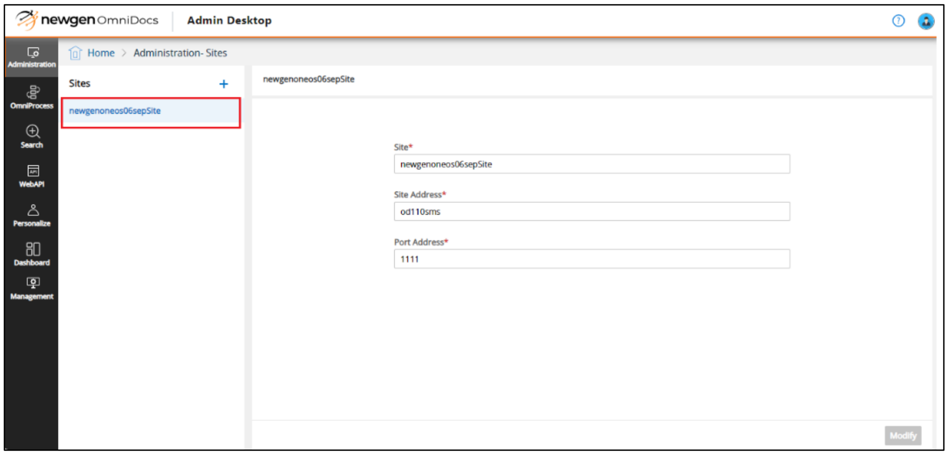
Site*: newgenoneos06sepSite

Site Address*: od110sms

Port Address*: 1111

Figure 2.76

6. The added Site appears under Sites in the left pane.



newgenoneos06sepSite

Site* newgenoneos06sepSite

Site Address* od110sms

Port Address* 1111

Figure 2.77

7. Go back to the **Home** page.

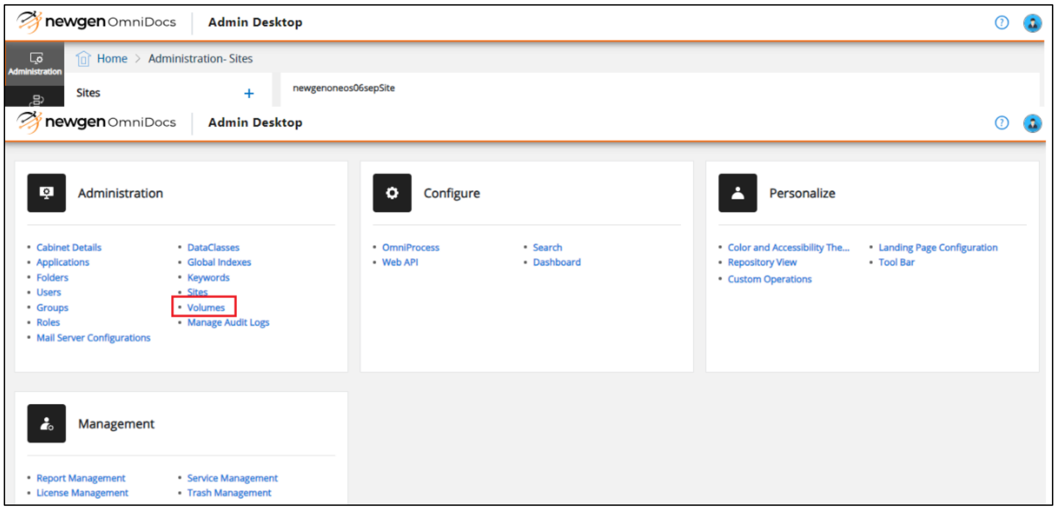
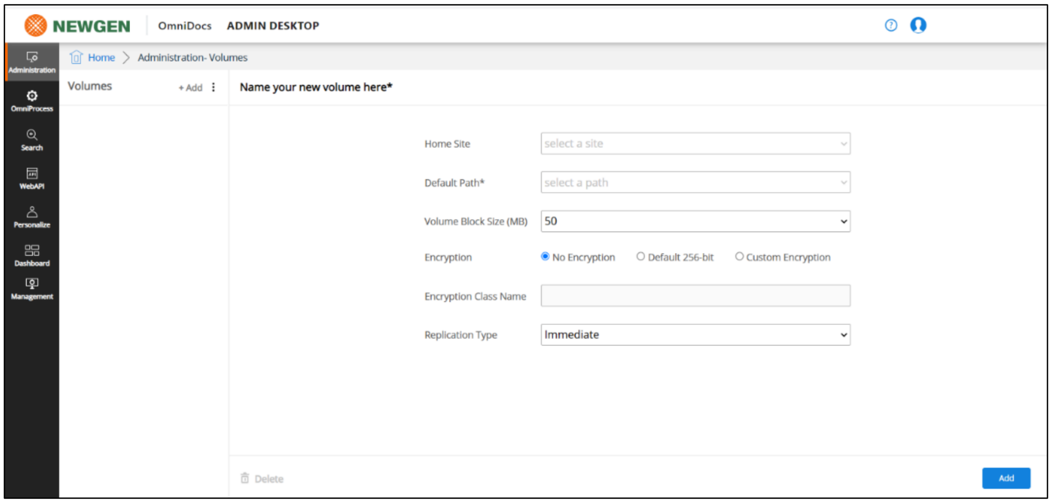


Figure 2.78

8. Select **Volumes**. The Volumes screen appears.



Name your new volume here*

Home Site: select a site

Default Path*: select a path

Volume Block Size (MB): 50

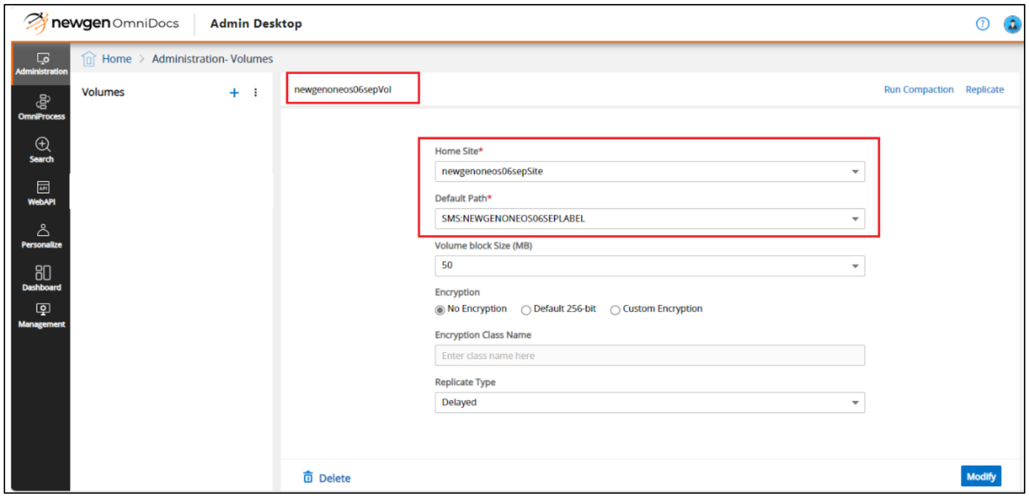
Encryption: ● No Encryption ○ Default 256-bit ○ Custom Encryption

Encryption Class Name:

Replication Type: Immediate

Figure 2.79

9. Specify the following details:
   - **Volume Name**: Specify the user-defined volume name.
   - **Home Site**: Select the newly created Site name.
   - **Default Path:** Select the created SMS label in which you want to store PN files.
10. Click **Add**.



newgenoneos06sepVol

Home Site*: newgenoneos06sepSite

Default Path*: SMS:NEWGENONEOS06SEPLABEL

Volume block Size (MB): 50

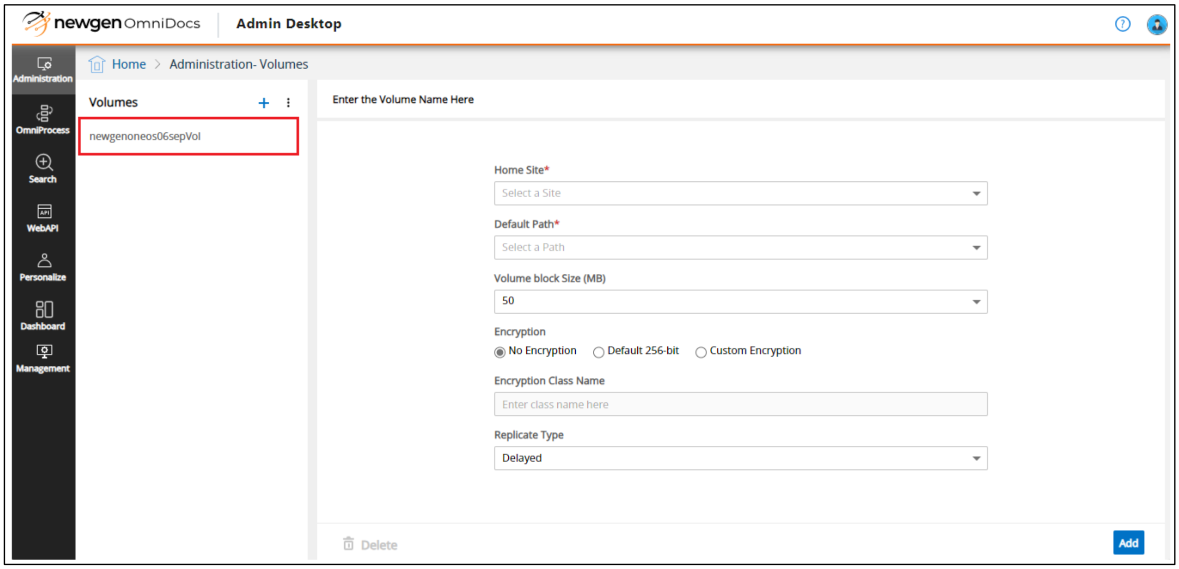
Encryption: ● No Encryption ○ Default 256-bit ○ Custom Encryption

Encryption Class Name: Enter class name here

Replicate Type: Delayed

Figure 2.80

The added volume appears under **Image Volumes** in the left panel.



newgenoneos06sepVol

Enter the Volume Name Here

Home Site*
Select a Site

Default Path*
Select a Path

Volume block Size (MB)
50

Encryption
● No Encryption ○ Default 256-bit ○ Custom Encryption

Encryption Class Name
Enter class name here

Replicate Type
Delayed

Figure 2.81

11. Go back to the **Home** screen and click **Cabinet Details**

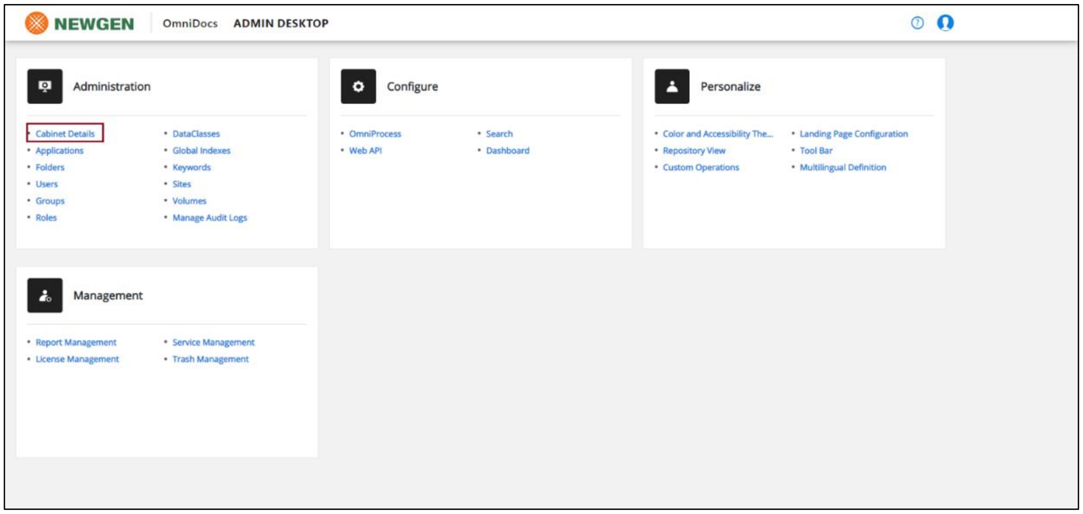
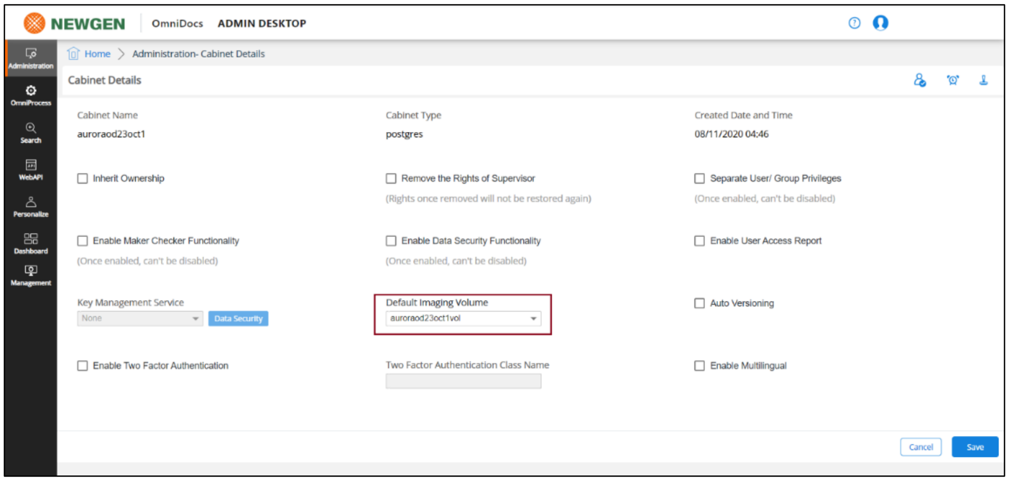


Figure 2.82

12. Click **Cabinet Details**.
13. Select the added volume from the **Default Image Volume** using the dropdown.
14. Click **Save**. The Site and Volume are now created successfully.



Cabinet Name: auroraod23oct1

Cabinet Type: postgres

Created Date and Time: 08/11/2020 04:46

□ Inherit Ownership

□ Remove the Rights of Supervisor (Rights once removed will not be restored again)

□ Separate User/ Group Privileges (Once enabled, can't be disabled)

□ Enable Maker Checker Functionality (Once enabled, can't be disabled)

□ Enable Data Security Functionality (Once enabled, can't be disabled)

□ Enable User Access Report

Key Management Service: None [Data Security]

Default Imaging Volume: auroraod23oct1vol

□ Auto Versioning

□ Enable Two Factor Authentication

Two Factor Authentication Class Name:

□ Enable Multilingual

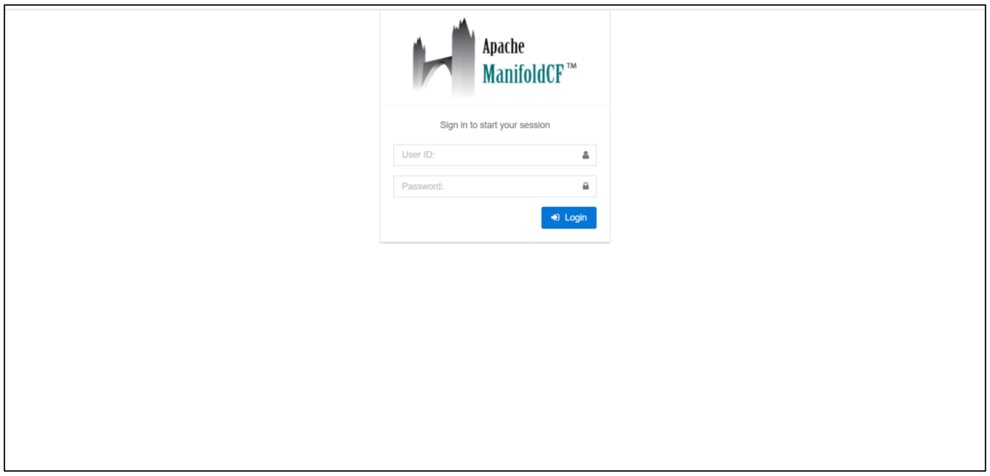
Figure 2.83

15. Log in to the OmniDocs Web using the below URL to start.
    http://<Host-Path URL of OmniDocsWeb container>/OmniDocs/web
    For example: *http:// OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net/OmniDocs/web*

2.10 EasySearch post-deployment changes

Perform the below steps to do EasySearch post-deployment changes:

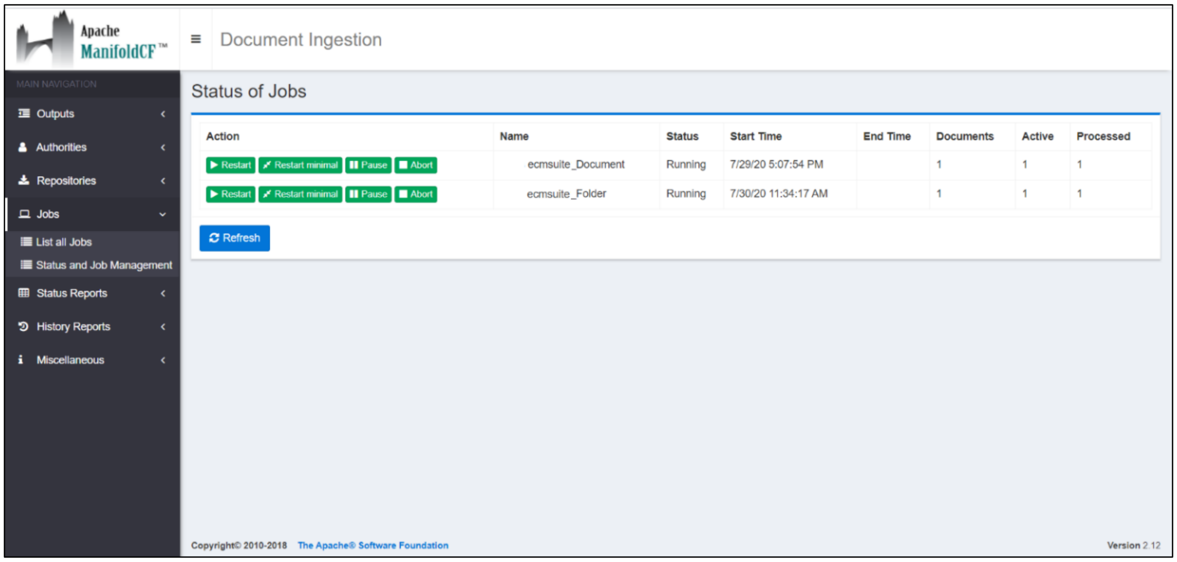
1. Login to the ApacheManifold Admin using the following URL:
   <Host-Path URL of ApacheManifold>/mcf-crawler-ui/login.jsp
   For example,
   http://apachemanifold .apps.newgenopenshiftcluster.newgensoftware.net/mcf-crawler-ui/login.jsp



Sign in to start your session

Figure 2.84

2. Log in with the following credentials:
    - **User ID:** admin
    - **Password**: admin
3. After a successful login, click **Jobs** tree showing in the left panel.
4. Click **Status and Job Management**. The below job list appears:
    - <CABINET_NAME>_Document
    - <CABINET_NAME>_Folder
5. Start both the jobs.
6. Once both the jobs started, the Job's status appears as **Running.**



Status of Jobs

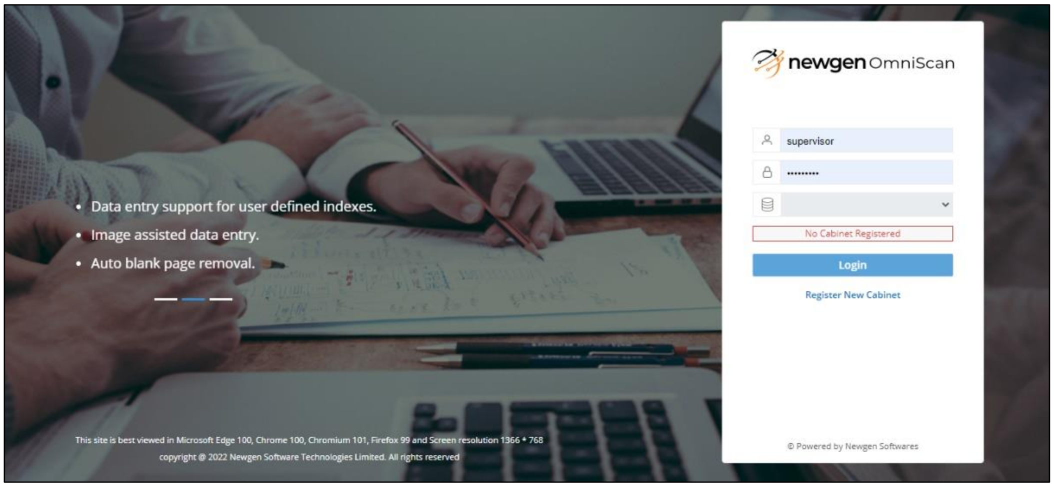
Action	Name	Status	Start Time	End Time	Documents	Active	Processed
▶ Restart ⤢ Restart minimal ❚❚ Pause ■ Abort	ecmsuite_Document	Running	7/29/20 5:07:54 PM		1	1	1
▶ Restart ⤢ Restart minimal ❚❚ Pause ■ Abort	ecmsuite_Folder	Running	7/30/20 11:34:17 AM		1	1	1

Figure 2.85

2.11 Registering a cabinet in OmniScanWeb

Perform the below steps to register the cabinet in OmniScanWeb:

1. Open the OmniScanWeb using the following URL:
    http://<Host-Path URL of OmniScanWeb container>/omniscanweb
    For example,
    Error! Hyperlink reference not valid.
2. Click **Register New Cabinet** link on the OmniScan Web login screen.



- Data entry support for user defined indexes.
- Image assisted data entry.
- Auto blank page removal.

This site is best viewed in Microsoft Edge 100, Chrome 100, Chromium 101, Firefox 99 and Screen resolution 1366 * 768

copyright @ 2022 Newgen Software Technologies Limited. All rights reserved

newgen OmniScan

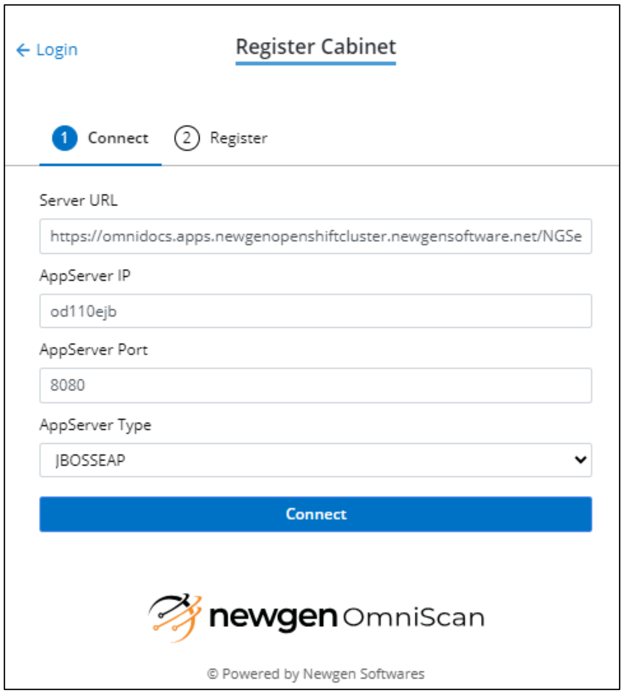
supervisor

•••••••••

No Cabinet Registered

Figure 2.86

3. Specify the Server URL as given below:
   http://<Host-Path URL of OmniDocsWeb container>/NGServlet/servlet/ExternalServlet
   For example,
   https://OmniDocs.apps.newgenopenshiftcluster.newgensoftware.net/NGServlet/servlet/ExternalServlet
4. Specify the **OmniDocs EJB** container name for AppServer IP or Server URL, 8080 for AppServer Port, and JBOSSEAP for AppServer Type.



Server URL

https://omnidocs.apps.newgenopenshiftcluster.newgensoftware.net/NGSe

AppServer IP

od110ejb

AppServer Port

8080

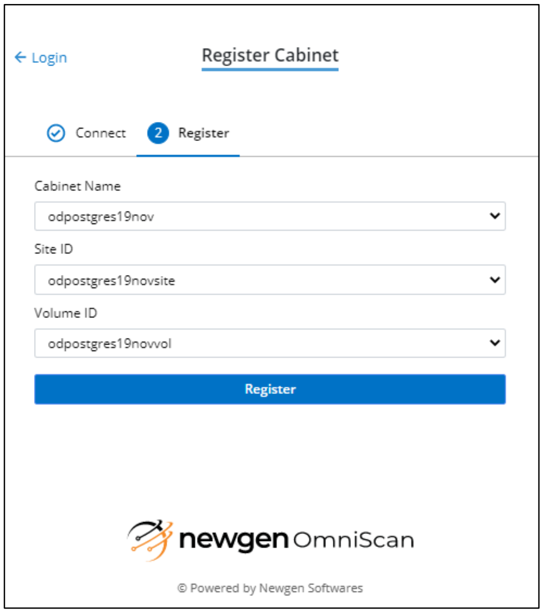
AppServer Type

JBOSSEAP

Figure 2.87

5. Click **Connect**.

6. Select the **Cabinet Name**, **Site ID**, and **Volume ID** from the list.



Cabinet Name: odpostgres19nov

Site ID: odpostgres19novsite

Volume ID: odpostgres19novvol

Figure 2.88

7. Click **Register**.

   The registered cabinet appears in the **Cabinet Name** list on the login screen. Now you can log into OmniScan Web.

NOTE:

Ensure that the **OmniScan_Template_Repository** folder is already created in OmniDocs before logging into OmniScan Web.

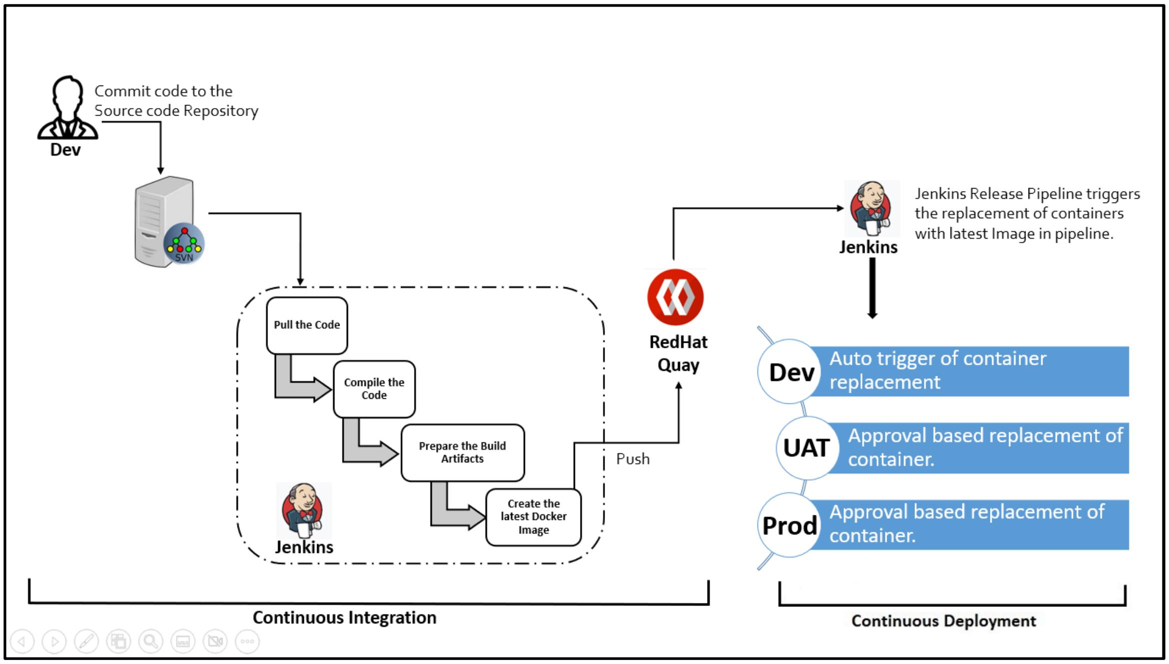
3 Configuring Jenkins release pipeline

This chapter describes the configuration of Jenkins Release Pipeline, follow the below sections for procedural details.

3.1 Overview

The Build Pipeline (aka Continuous Integration) and Release Pipeline (aka Continuous Deployment) are separated into two parts. Build Pipeline and Release Pipeline both are done through the Jenkins server which can be installed on an on-premises machine or a bastion server. In this architecture, three stages are created, Dev, UAT, and Production, and in each stage, deployment is quite different. You can have some more stages depending on the requirements. This document describes the configuration of the Jenkins Release Pipeline for container deployment on the OpenShift Container Platform.

3.2 CICD pipeline architecture



Commit code to the Source code Repository
Dev
SVN
Pull the Code
Compile the Code
Prepare the Build Artifacts
Create the latest Docker Image
Jenkins
Push
RedHat Quay
Jenkins
Jenkins Release Pipeline triggers the replacement of containers with latest Image in pipeline.
Dev
Auto trigger of container replacement
UAT
Approval based replacement of container.
Prod
Approval based replacement of container.
Continuous Integration
Continuous Deployment

Figure 3.1

1. The Newgen representative builds the product's base container images on the company's on-premises servers using Jenkins.
2. As soon as the Dev team commits the code to the source code repository, the Jenkins pipeline gets triggered. It pulls the code then compiles them and prepares the build artifacts as well as

creates container images and pushes the newly created container images to the RedHat Quay Registry.

3. As soon as any container image is pushed to the RedHat Quay Registry, Jenkins Release Pipeline triggers the deployment to the Dev environment. Here, you can configure the performance testing as well as security testing of the application. In Addition, you can perform manual testing as required.
4. UAT and Production deployments are based on approval and are available on-demand. To deploy to the UAT/Production environment, you need to trigger the UAT/Production deployment. Upon deployment trigger, an approval mail is sent to the project manager or the concerned team. As soon as the project manager or concerned team approves the go-ahead, UAT/Production deployment gets started.

3.3 Configuring Jenkins release pipeline

This section describes the configuration of Jenkins for Build Pipeline.

3.3.1 Prerequisites

Following are the prerequisites:

- **Operating System**: Windows Server 2019 or above (Edition: Standard or Datacenter).
- **Java** 1.11 update 18 and above.
- **Docker Engine** 20.10.10 or later version must be installed.
- **OpenShift CLI** 4.12.8 or a later version compatible with OpenShift cluster.
- **Cygwin** utility must be installed. [This utility is used to execute the Linux commands on Windows].
- **Jenkins** 2.246.0 or a later version must be installed with default suggested plug-ins along with the following plug-ins.
  - Credentials Binding
  - Environment Injector
  - Parameterized Trigger

3.3.2 Configuration of Jenkins

Before creating any job, perform the following server-level configurations in the Jenkins.

1. Sign into the **Jenkins Server**.

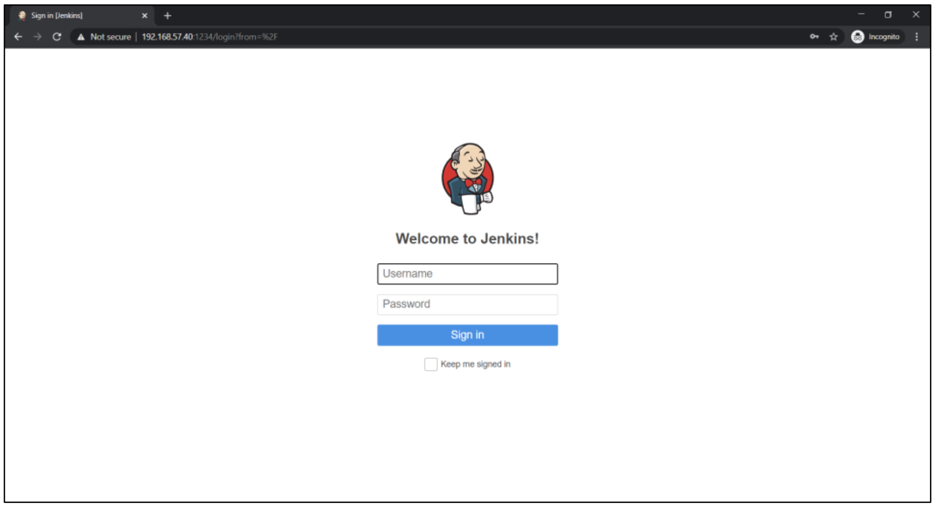


Figure 3.2

2. After the successful login, click **Manage Jenkins** link showing on the left panel.

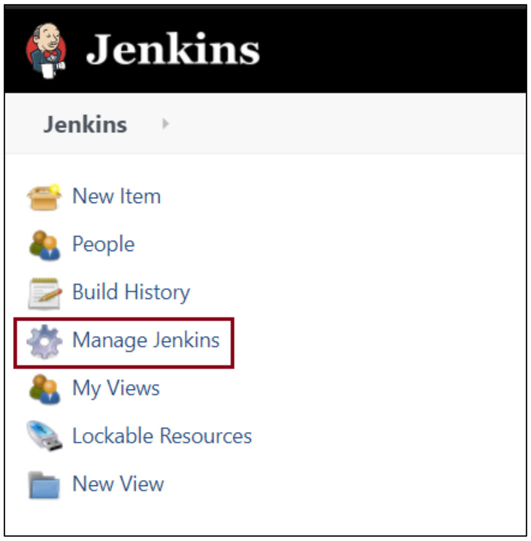
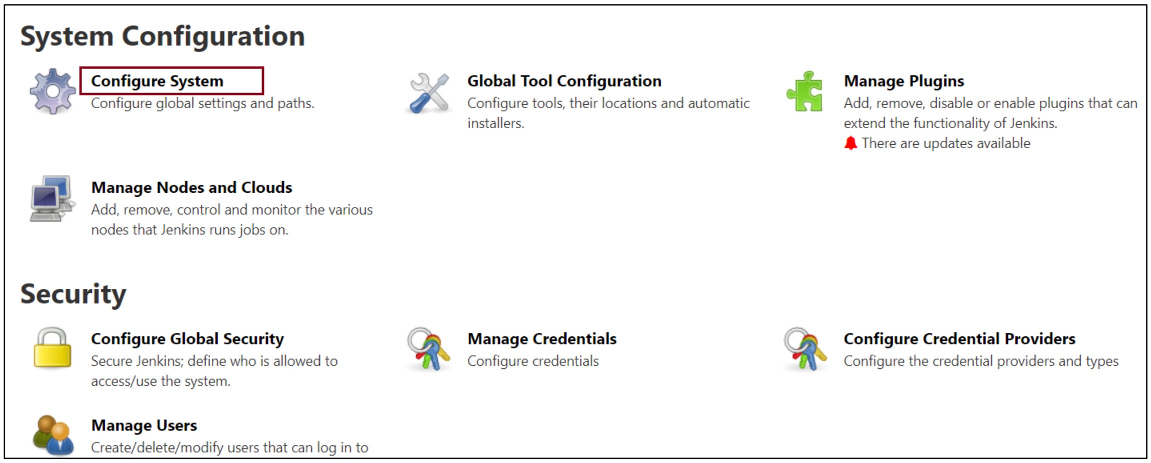


Figure 3.3

3. Click **Configure System** in the **System Configuration** section.



System Configuration

Configure System
Configure global settings and paths.

Global Tool Configuration
Configure tools, their locations and automatic installers.

Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
There are updates available

Manage Nodes and Clouds
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Security

Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.

Manage Credentials
Configure credentials

Configure Credential Providers
Configure the credential providers and types

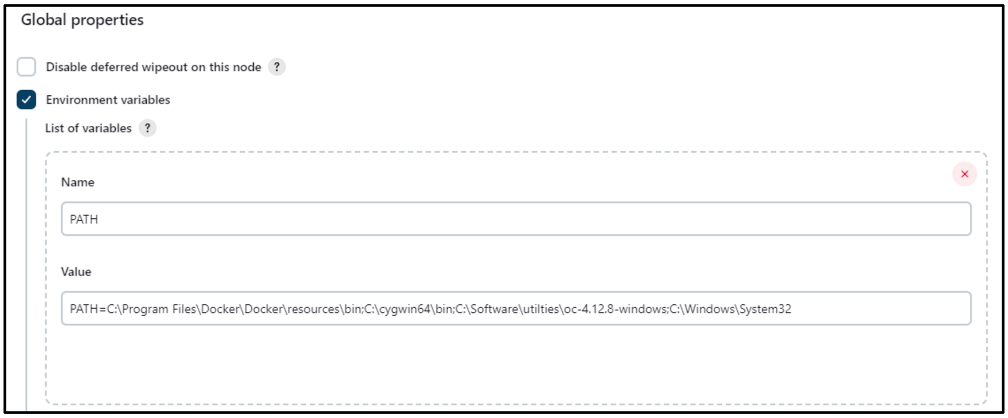
Manage Users
Create/delete/modify users that can log in to

Figure 3.4

4. Under the **Global properties**, define an environment variable **PATH** with the following values separated with a semi-colon:
   - Docker installation path for example, *C:\Program Files\Docker\Docker\resources\bin*
   - Cygwin installation path for example*, C:\cygwin64\bin*
   - OpenShift CLI installation path for example, *C:\Software\utilties\oc-4.12.8-windows*
   - Windows System32 path [C:\Windows\System32]

   For example,

```
PATH=C:\Program
Files\Docker\Docker\resources\bin;C:\cygwin64\bin;C:\Software\utilties\oc-
4.12.8-windows;C:\Windows\System32
```



Global properties

□ Disable deferred wipeout on this node ?

☑ Environment variables

List of variables ?

Name

PATH

Value

PATH=C:\Program Files\Docker\Docker\resources\bin;C:\cygwin64\bin;C:\Software\utilties\oc-4.12.8-windows;C:\Windows\System32

Figure 3.5

5. Click **Save** the changes.

3.3.3 Push and Pull Container Images to/from RedHat Quay

This section describes how to push and pull container images to/from RedHat Quay Registry.

Prerequisites - Ensure that you have installed the latest version of OpenShift **CLI** and **Docker**.

Following are the steps to push, and pull container images to/from RedHat Quay Registry:

- Authentication
- Push
- Pull

Authentication:

1. Before you push or pull the container images, you need to authenticate the Docker client to RedHat Quay.
2. Execute the below command to authenticate the Docker client to RedHat Quay:

```
docker login -u="Quay_User" -p="Quay_Password" quay.io
```

Push:

1. To push any local container images to a registry, it is mandatory to first tag that image.
2. Execute the below command to push the container images from your local machine to RedHat Quay registry:

```
docker tag <ImageName>:<ImageTag> <Quay_Server>/<ImageName>:<ImageTag>
docker push <Quay_Server>/<ImageName>:<ImageTag>
```

NOTE:

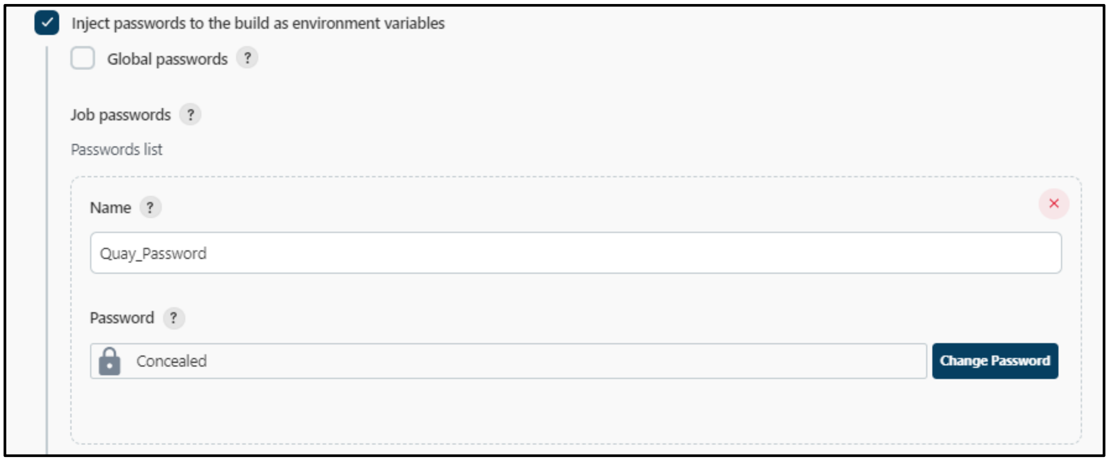
- Container images might be shared in the form of a compressed tar file. As compressed container images cannot be used directly, first you need to decompress them in a container image form, and then you can use it. In such a case, the client needs to perform the following:
  - ➢ Download the compressed container image file.
  - ➢ Convert the compressed file into a container image using the docker load command.
  - ➢ Example: docker load *–i C:\ContainerImages\ OmniDocs110EJB.tar*
  - ➢ Re-tag the images with your own registry and push them up.
- You can also configure these commands in Jenkins to execute them automatically.

3. Use the below **batch** scripts to configure the **push container images** to RedHat Quay in Jenkins:

```
@echo off
set Quay_Server=quay.io/newgen
set Quay_User=vivek_kumar
set Quay_Password=%Quay_Password%
set ImageName=OmniDocs11.0ejb
set ImageTag=sp1-jdk17-patch3
set BuildNumber=%ImageTag%-build-%BUILD_NUMBER%
```

```
docker login -u="%Quay_User%" -p="%Quay_Password%" quay.io
docker tag %ImageName%:%ImageTag% %Quay_Server%/%ImageName%:%ImageTag%
docker push %Quay_Server%/%ImageName%:%ImageTag%

docker tag %ImageName%:%ImageTag% %Quay_Server%/%ImageName%:%BuildNumber%
docker push %Quay_Server%/%ImageName%:%BuildNumber%
```

Here,
Quay_Server =quay.io/newgen, in this server detail, newgen is the name of the organization created inside quay.io server.
%Quay_Password% is encrypted using the Jenkins "*Inject passwords to the build as environment variables*" Build Environment options. For example,



☑ Inject passwords to the build as environment variables

□ Global passwords ?

Job passwords ?

Passwords list

Name ?

Quay_Password

Password ?

Concealed

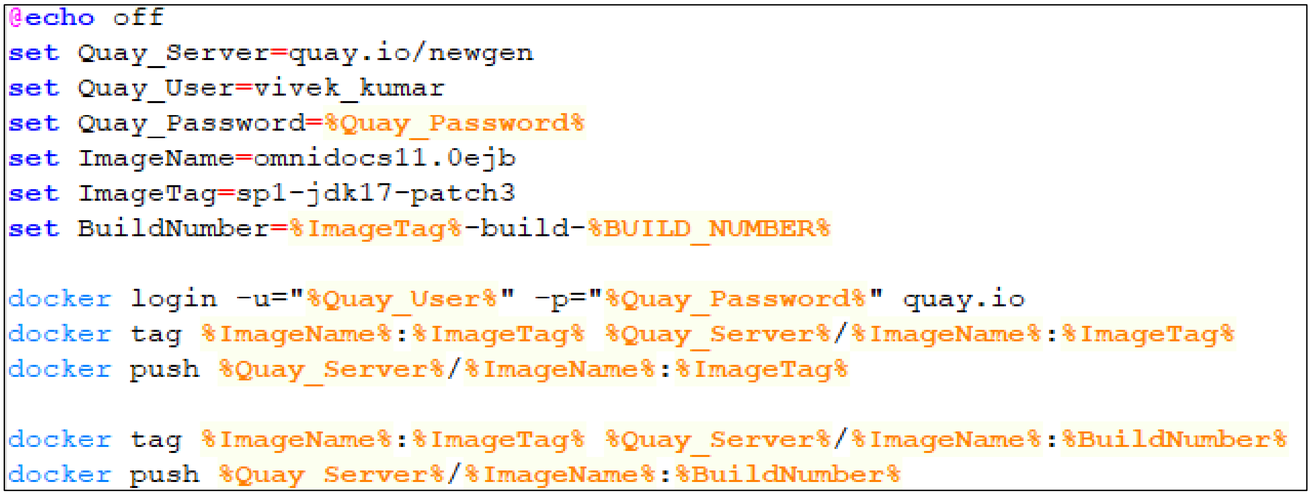
Figure 3.6

In the above scripts, images are tags and pushed 2 times, 1st time with default image tag and 2nd time with BuildNumber tag. As for container image management, two tags creation is recommended for each container image through Jenkins while pushing them to the container registry.
For Example,

- OmniDocs11.0ejb: sp1-jdk17-patch3
- OmniDocs11.0ejb: sp1-jdk17-patch3-build10 (where build-10 is the build pipeline number)

This means that you will always have a copy of the container image of each build pipeline in the container registry so that whenever we require to roll back the current deployment, we can use the previous build pipeline number tag – container Image for rolling back the deployments.

For example,



```
@echo off
set Quay_Server=quay.io/newgen
set Quay_User=vivek_kumar
set Quay_Password=%Quay_Password%
set ImageName=omnidocs11.0ejb
set ImageTag=sp1-jdk17-patch3
set BuildNumber=%ImageTag%-build-%BUILD_NUMBER%

docker login -u="%Quay_User%" -p="%Quay_Password%" quay.io
docker tag %ImageName%:%ImageTag% %Quay_Server%/%ImageName%:%ImageTag%
docker push %Quay_Server%/%ImageName%:%ImageTag%

docker tag %ImageName%:%ImageTag% %Quay_Server%/%ImageName%:%BuildNumber%
docker push %Quay_Server%/%ImageName%:%BuildNumber%
```

Figure 3.7

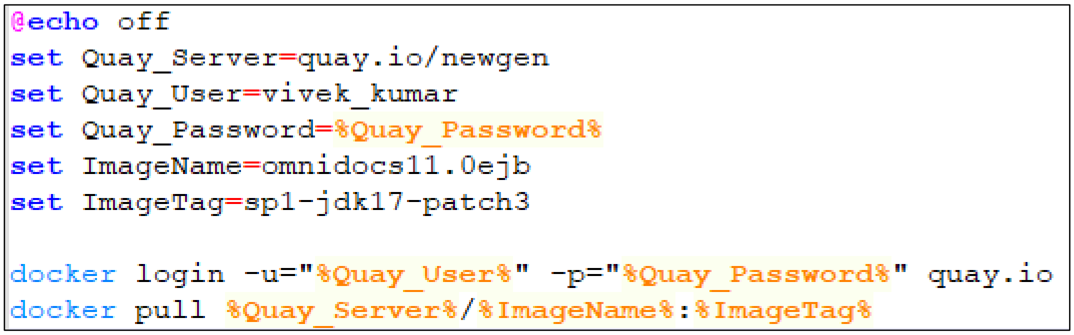
Pull:

1. Execute the below command to pull the Docker images from AWS ECR:

```
docker pull <Quay_Server>/<ImageName>:<ImageTag>
```

2. Use the below **batch** scripts to configure the **pull container images** from RedHat Quay in Jenkins:

```
@echo off
set Quay_Server=quay.io/newgen
set Quay_User=vivek_kumar
set Quay_Password=%Quay_Password%
set ImageName=OmniDocs11.0ejb
set ImageTag=sp1-jdk17-patch3

docker login -u="%Quay_User%" -p="%Quay_Password%" quay.io
docker pull %Quay_Server%/%ImageName%:%ImageTag%
```



```
@echo off
set Quay_Server=quay.io/newgen
set Quay_User=vivek_kumar
set Quay_Password=%Quay_Password%
set ImageName=omnidocs11.0ejb
set ImageTag=sp1-jdk17-patch3

docker login -u="%Quay_User%" -p="%Quay_Password%" quay.io
docker pull %Quay_Server%/%ImageName%:%ImageTag%
```

Figure 3.8

3.3.4 Configuring Dev stage

Perform the below steps to configure a Jenkins job for the Dev Stage Release Pipeline:

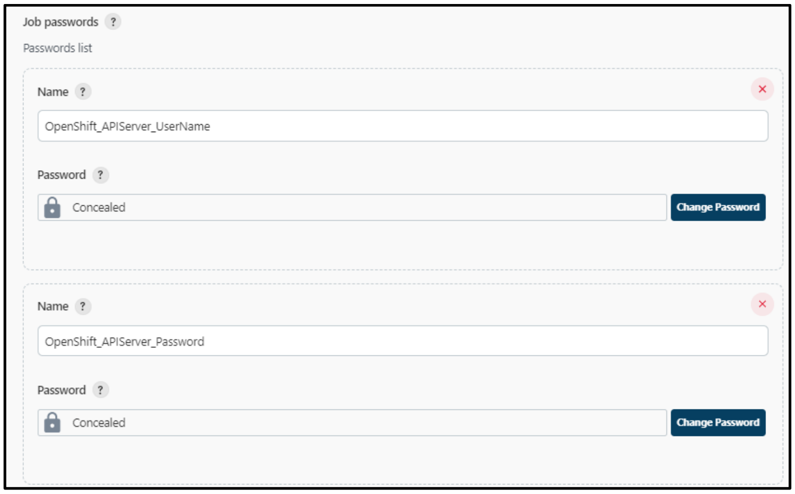
NOTE:

Refer the following steps as a reference to configure the Release Pipeline for the below container Images.

- OmniDocs11.0Web
- OmniDocs11.0Web_Services
- OmniDocs11.0Services
- EasySearch11.0_ApacheOnly
- TEM11.0
- OmniScanWeb6.0
- OmniDocs11.0SMS
- RMSSharePointAdapter
- Wopi

1. Click **New Item** link given on the left panel.
2. Specify the item name or job name and select the project type as **Freestyle project**.
3. You can specify the project description.
4. Select the checkbox **Inject passwords to the build as environment variables** given in the **Build Environment** section.
5. Specify 2 Job passwords: **OpenShift_APIServer_UserName** and **OpenShift_APIServer_Password** and provide the appropriate username and password to authenticate the OpenShift CLI to connect to the OpenShift API Server.

   For example,



Job passwords ?

Passwords list

Name ?

OpenShift_APIServer_UserName

Password ?

Concealed

Name ?

OpenShift_APIServer_Password

Password ?

Concealed

Figure 3.9

6. Add **'Execute Shell'** as a build step task under the **Build** section and specify the following shell script:

```
#Quay_Server=quay.io/newgen
Quay_Server=678035612169.dkr.ecr.ap-south-1.amazonaws.com
OpenShift_APIServer_URI=https://api.newgenopenshiftcluster.newgensoftware.net:
6443
ImageName= OmniDocs11.0ejb
ImageTag=sp1-jdk17-patch3
YAML_File_Location="D:\RunningPods\OpenShift\YAML_Files\OmniDocs"
YAML_File_Name= OmniDocs11.0EJB.yml

mkdir -p TempDir
cp -rf "$YAML_File_Location/$YAML_File_Name" TempDir/

grep -q Quay_Server TempDir/$YAML_File_Name
grep -q ImageName TempDir/$YAML_File_Name
grep -q ImageTag TempDir/$YAML_File_Name
grep -q Jenkins_Build_Number TempDir/$YAML_File_Name

sed -i s+Quay_Server+$Quay_Server+g TempDir/$YAML_File_Name
sed -i s+ImageName+$ImageName+g TempDir/$YAML_File_Name
sed -i s+ImageTag+$ImageTag+g TempDir/$YAML_File_Name
sed -i s+Jenkins_Build_Number+$BUILD_NUMBER+g TempDir/$YAML_File_Name

#oc login $OpenShift_APIServer_URI --username=$OpenShift_APIServer_UserName --
password=$OpenShift_APIServer_Password --insecure-skip-tls-verify

#oc apply -f "$YAML_File_Location\quay-secret.yml"
oc apply -f "$YAML_File_Location\Storage_PV_PVC.yml"
oc apply -f TempDir/$YAML_File_Name
oc apply -f "$YAML_File_Location\IngressController.yml"
```

   Here,
   Quay_Server =quay.io/newgen, in this server detail, newgen is the name of the organization created inside quay.io server where container images are stored. Jenkins will use this Quay server to pull the container images and deploy them as containers to the OpenShift container platform.
   OpenShift_APIServer_URI: Provide the OpenShift API Server URI which will be used by OpenShift CLI to connect to the OpenShift cluster.
   ImageName & ImageTag: The container image name and tag that you want to deploy.
   YAML_File_Location: Directory in which all products YAML file are kept.
   YAML_File_Name: Provide the YAML file name which will be used to deploy above mentioned container images. For example., to deploy ibps5admininstanceejb container image, iBPS5.0AdminInstanceEJB.yml will be used.

3.3.5 Configuring UAT or Production stage

In this Jenkins Release Pipeline, UAT and Production deployments are approval-based and are available on-demand. To deploy to the UAT or Production environment, you need to trigger the UAT or Production deployment. Upon deployment trigger, an approval mail is sent to the project manager or the concerned team. As soon as the project manager or concerned team approves the go-ahead, UAT/Production deployment gets started.

Therefore, for UAT or Production deployment, 2 Jenkins jobs will be created in Release Pipeline.

- Waiting for Approval
- Trigger the Deployment

NOTE:

Refer the following steps as a reference to configure the Release Pipeline for the below container Images.

- OmniDocs11.0Web
- OmniDocs11.0Web_Services
- OmniDocs11.0Services
- EasySearch11.0_ApacheOnly
- TEM11.0
- OmniScanWeb6.0
- OmniDocs11.0SMS
- RMSSharePointAdapter
- Wopi

<u>Create Jenkins Job: Waiting for Approval</u>

1. Click '**New Item'** link given on the left panel.
2. Specify the item name or job name and select the project type as **Pipeline**.
3. You can specify the project description.
4. Specify the following Groovy script under the Pipeline Definition:

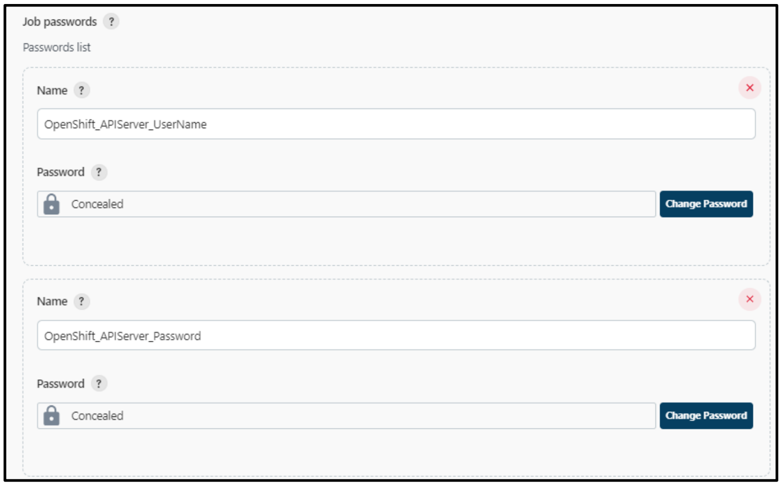
```
pipeline {
    agent any
    stages {
        stage('Send notification for approval') {
            steps {

                mail (to: 'amit.gaur@newgensoft.com', cc:
'vivek_kumar@newgensoft.com,vikash.k@newgensoft.com',
                subject: "Release Pipeline: OmniDocs EJB: Pipeline
#${env.BUILD_NUMBER} is waiting for approval",
                body: "Dear Sir/Madam,\n\n OmniDocs EJB Release Pipeline is
waiting for your approval to deploy to the UAT/Production
environment.\n\nPlease refer to the below link to allow or reject.\n
${env.BUILD_URL}console.\n\n Regards:\n DevOps Team");
                //echo 'Send Notification to Developer'
            }
        }
        stage('Waiting for approval') {
            steps {
                input 'Dear Sir/Madam,\nProceed to deploy to the
UAT/Production environment?'
            }
        }

    }
}
```

5. In the above script you can update the following items as per your business requirement:
   - To Recipient List
   - Cc Recipient List
   - Mail Subject
   - Mail Body

NOTE:
Make sure Email Notification is configured in Jenkins before executing this job that is, Waiting for Approval

<u>Create Jenkins Job: Trigger the Deployment</u>

1. Click '**New Item'** link given on the left panel.
2. Specify the item name or job name and select the project type as **Freestyle project**.
3. You can specify the project description.
4. Select the checkbox **Inject passwords to the build as environment variables** given in the **Build Environment** section.
5. Specify 2 Job passwords: **OpenShift_APIServer_UserName** and **OpenShift_APIServer_Password** and provide the appropriate username and password to authenticate the OpenShift CLI to connect to the OpenShift API Server.
   For example,



Job passwords ?

Passwords list

Name ?

OpenShift_APIServer_UserName

Password ?

Concealed

Name ?

OpenShift_APIServer_Password

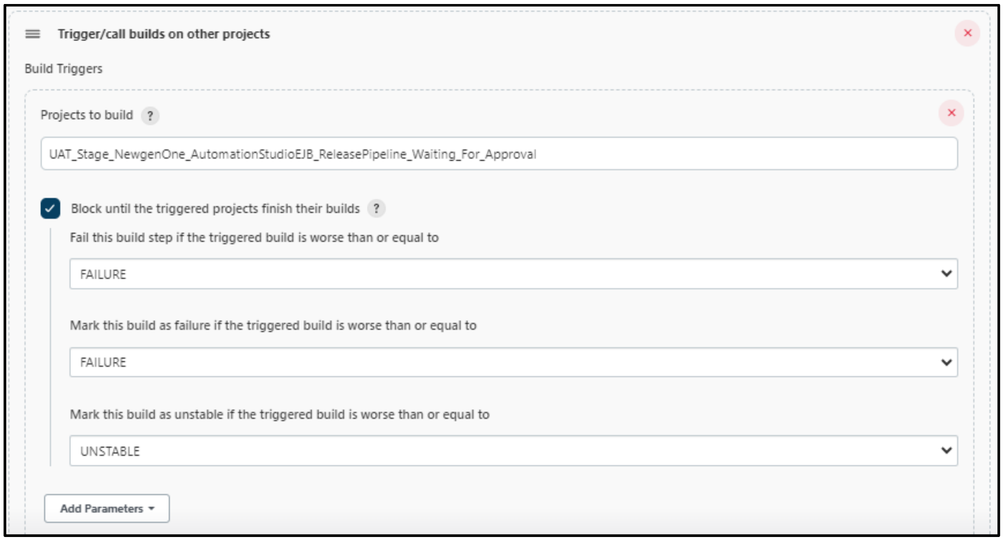
Password ?

Concealed

Figure 3.10

6. Add '**Trigger/call builds on other projects**' as a build step task under the **Build** section and provide the job name which is created for approval in the previous steps that is, **Waiting for Approval**.

   For example,



Trigger/call builds on other projects

Build Triggers

Projects to build ?

UAT_Stage_NewgenOne_AutomationStudioEJB_ReleasePipeline_Waiting_For_Approval

☑ Block until the triggered projects finish their builds ?

Fail this build step if the triggered build is worse than or equal to

FAILURE

Mark this build as failure if the triggered build is worse than or equal to

FAILURE

Mark this build as unstable if the triggered build is worse than or equal to

UNSTABLE

Figure 3.11

7. Add **'Execute Shell'** as a build step task under the **Build** section and specify the following shell script:

```
Quay_Server=quay.io/newgen
OpenShift_APIServer_URI=https://api.newgenoscluster.newgensoftware.net:6443
ImageName=OmniDocs11.0ejb
ImageTag=sp1-jdk17-patch3
YAML_File_Location="D:\RunningPods\OpenShift\YAML_Files\OmniDocs"
```

```
YAML_File_Name=OmniDocs11.0EJB.yml

mkdir -p TempDir
cp -rf "$YAML_File_Location/$YAML_File_Name" TempDir/

grep -q Quay_Server TempDir/$YAML_File_Name
grep -q ImageName TempDir/$YAML_File_Name
grep -q ImageTag TempDir/$YAML_File_Name
grep -q Jenkins_Build_Number TempDir/$YAML_File_Name

sed -i s+Quay_Server+$Quay_Server+g TempDir/$YAML_File_Name
sed -i s+ImageName+$ImageName+g TempDir/$YAML_File_Name
sed -i s+ImageTag+$ImageTag+g TempDir/$YAML_File_Name
sed -i s+Jenkins_Build_Number+$BUILD_NUMBER+g TempDir/$YAML_File_Name

oc login $OpenShift_APIServer_URI --username=$OpenShift_APIServer_UserName --
password=$OpenShift_APIServer_Password --insecure-skip-tls-verify

oc apply -f "D:\RunningPods\OpenShift\YAML_Files\OmniDocs\quay-secret.yml"
oc apply -f "D:\RunningPods\OpenShift\YAML_Files\OmniDocs\Storage_PV_PVC.yml"
oc apply -f TempDir/$YAML_File_Name
oc apply -f
"D:\RunningPods\OpenShift\YAML_Files\OmniDocs\IngressController.yml"
```

Here,
Quay_Server =quay.io/newgen, in this server detail, newgen is the name of the organization created inside quay.io server where container images are stored. Jenkins will use this Quay server to pull the container images and deploy them as containers to the OpenShift container platform.
OpenShift_APIServer_URI: Provide the OpenShift API Server URI which will be used by OpenShift CLI to connect to the OpenShift cluster.
ImageName & ImageTag: The container image name and tag that you want to deploy.
YAML_File_Location: Directory in which all products YAML file are kept.
YAML_File_Name: Provide the YAML file name which will be used to deploy above mentioned container images. For example, to deploy od110ejb container image, OmniDocs11.0EJB.yml will be used.