Configuration and Deployment Guide
Oracle Banking Electronic Data Exchange for Corporates
Patchset Release 14.7.2.0.0

Part No. F89453-01

November 2023

ORACLE

Configuration and Deployment Guide

November 2023

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax:+91 22 6718 3001

www.oracle.com/financialservices/

Table of Contents

1. Configuration and Deployment

1.1 Introduction

This guide is a supporting document for the installation of Oracle Banking Microservices Architecture applications. You can find the reference in the respective installation guides.

1.2 Placeholder Update for Oracle Banking Microservices Architecture Services

The Placeholder update can be performed in the following methods:

- Method 1 – Via **setUserOverrides.sh** file
- Method 2 – Via passing the **-D params** in the Server start argument
- Method 3 – Using **env** files and **setUserOverrides.sh** file
- Method 4 – Via Workflow creation in Plato O

1.2.1 Method 1 – Via setUserOverrides.sh file

Perform the following steps:

1. Create a file called **setUserOverrides.sh** inside the Weblogic bin location.
2. The following formats of the **setUserOverrides.sh** file and the list of parameters that need to be passed in order to run Oracle Banking Microservices Architecture services properly.

Note: Below are the list of **-D params** (ENV Variables) which needs to be set for all the individual services. Set a single **–Dparam** as follows:

1) JAVA_OPTIONS="${JAVA_OPTIONS} -DParam =<ParamValue>"
2) export JAVA_OPTIONS

//Common Properties

-Dplato.services.config.port= <CONFIG_SERVICE_PORT>

-Dplato.services.config.uri=
http://<CONFIG_SERVICE_HOSTNAME>:<CONFIG_SERVICE_PORT>

-Deureka.client.serviceUrl.defaultZone=
http://<DISCOVERY_SERVICE_HOST>:<DISCOVERY_SERVICE_PORT>/plato-discovery-service/eureka

-Dplato.services.entityservices.port= <PLATO_ORCH_SERVICE_PORT>

-Dplato.service.logging.path= <LOGGING PATH>

-Dspring.cloud.loadbalancer.ribbon.enabled=false

-Dspring.main.allow-circular-references=true

//Flyway Common Placeholders

-Dflyway.domain.placeHolders.eureka.host= <DISCOVERY_SERVICE_HOST>

-Dflyway.domain.placeHolders.eureka.port= <DISCOVERY_SERVICE_PORT>

-Dflyway.domain.placeHolders.plato-api-gateway.server.port=
<API_GATEWAY_PORT>

-Dflyway.domain.placeHolders.zipkin.host= <ZIPKIN_HOSTNAME>

-Dflyway.domain.placeHolders.zipkin.port= <ZIPKIN_PORT>

//SMS - Needed for other services also

-Dflyway.domain.placeHolders.sms.username= <SMS_SCHEMA_USERNAME>

-Dflyway.domain.placeHolders.sms.password= <SMS_SCHEMA_PASSWORD>

-Dflyway.domain.placeHolders.sms.jdbcUrl= <SMS_SCHEMA_URL>

-Dflyway.domain.placeHolders.sms.schemas= <SMS_SCHEMA_NAME>

//Plato Config Service - Needed for other services also

-Dflyway.domain.placeHolders.plato-config.username=
<PLATO_DB_USERNAME>

-Dflyway.domain.placeHolders.plato-config.password=
<PLATO_DB_PASSWORD>"

-Dflyway.domain.placeHolders.plato-config.jdbcUrl= <PLATO_DB_URL>

-Dflyway.domain.placeHolders.driver.className= oracle.jdbc.driver.OracleDriver

-Dflyway.domain.placeHolders.plato-config.schemas=
<PLATO_DB_SCHEMANAME>

//Plato Api Gateway - Needed for other services also

-Dflyway.domain.placeHolders.api-gateway.username=
<SECURITY_DB_USERNAME>

-Dflyway.domain.placeHolders.api-gateway.password=
<SECURITY_DB_PASSWORD>

-Dflyway.domain.placeHolders.api-gateway.jdbcUrl= <SECURITY_DB_URL>

-Dflyway.domain.placeHolders.api-gateway.schemas=
<SECURITY_SCHEMANAME>

-Dflyway.domain.placeHolders.apigateway.host=
<APIGATEWAY_ROUTER_HOSTNAME>

-Dflyway.domain.placeHolders.apigateway.port= <APIGATEWAY_ ROUTER _PORT>

-Dflyway.domain.placeHolders.USER.STORE= <USER.STORE>

-Dflyway.domain.placeHolders.LDAP.CORS.allowed.origin= <LDAP_CORS>

-Dflyway.domain.placeHolders.LDAP.credential.SALT=
<LDAP_CREDENTIALS_SALT>

-Dflyway.domain.placeHolders.JWT.EXPIRY.seconds=
<JWT_EXPIRY_SECONDS>

-Dflyway.domain.placeHolders.LDAP.url = <LDAP_SERVER_URL >

-Dflyway.domain.placeHolders.LDAP.userId = <LDAP_SERVER_USER>

-Dflyway.domain.placeHolders.LDAP.server.base = <LDAP_SERVER_BASE>

-Dflyway.domain.placeHolders.LDAP.server.credential =
<LDAP_CREDENTIALS>

-Dflyway.domain.placeHolders.LDAP.usersearch.base = <LDAP_USER_BASE>

-Dflyway.domain.placeHolders LDAP.user.prefix = <LDAP_USER_PREFIX>

-Dflyway.domain.placeHolders.LDAP.provider = <LDAP_PROVIDER>

-Dflyway.domain.placeHolders.TOKEN.autoregenerate =
<TOKEN_AUTOREGENERATION>

-Dflyway.domain.placeHolders.SSO.enabled = <SSO_ENABLED>

-Dflyway.domain.placeHolders.TOKEN.regeneration.enabled =
<TOKEN_ALWAYSNEW_GENERATION >

-Dplato-api-gateway.enableAudit=true

//Plato Discovery Service

-Dflyway.domain.placeHolders.plato-discovery-
service.server.port=<DISCOVERY_SERVICE_PORT>

//Plato UI-Config Services

-Dflyway.domain.placeHolders.plato-ui-config-services.server.port=<UICONFIG_SERVICE_PORT>

-Dflyway.domain.placeHolders.plato-ui-config.username=<UICONFIG_SCHEMA_USERNAME>

-Dflyway.domain.placeHolders.plato-ui-config.password=<UICONFIG_SCHEMA_PASSWORD>

-Dflyway.domain.placeHolders.plato-ui-config.jdbcUrl=<UICONFIG_SCHEMA_URL>

-Dflyway.domain.placeHolders.plato-ui-config.schemas=<UICONFIG_SCHEMA_NAME>

//Plato Apigateway Router Service

-Dflyway.domain.placeHolders.plato-apigateway-router.server.port=<APIGATEWAY_ROUTER_PORT>

-Dflyway.domain.placeHolders.plato-apigateway router.router.protocol=<ROUTER_PROTOCOL>

-Dflyway.domain.placeHolders.plato-apigateway-router.router.meadmin.port=<ROUTER_PORT>

//Plato Feed Services

-Dflyway.domain.placeHolders.plato-feed-services.feed.upload.directory=<FEED_SERVICE_UPLOAD_PATH>

-Dflyway.domain.placeHolders.plato-feed-services.server.port=<FEED_SERVICE_PORT>

-Dflyway.domain.placeHolders.plato-feed-services.username=<FEED_DB_USERNAME>

-Dflyway.domain.placeHolders.plato-feed-services.password=<FEED_DB_PASSWORD>

-Dflyway.domain.placeHolders.plato-feed-services.jdbcUrl=<FEED_DB_URL>

-Dflyway.domain.placeHolders.plato-feed-services.schemas=<FEED_SCHEMA_NAME>

//Plato Batch Server

-Dflyway.domain.placeHolders.plato-batch-server.server.port=<BATCH_SERVER_PORT>

-Dflyway.domain.placeHolders.plato-batch-server.plato.eventhub.kafka.brokers=<EVETNHUB_KAFKA_BROKERS>

-Dflyway.domain.placeHolders.plato-batch-server.plato.eventhub.zk.nodes=<ZK_NODES>

-Dflyway.domain.placeHolders.plato-batch-server.username=<BATCH_SCHEMA_USERNAME>

-Dflyway.domain.placeHolders.plato-batch-server.password=<BATCH_SCHEMA_PASSWORD>

-Dflyway.domain.placeHolders.plato-batch-server.jdbcUrl=<BATCH_SCHEMA_URL>

-Dflyway.domain.placeHolders.plato-batch-server.schemas=<BATCH_SCHEMA_NAME>

// Plato-Alerts-Management-Services

-Dflyway.domain.placeHolders.plato-alerts-management-services.server.port=<ALERTS-MANAGEMENT-SERVER-PORT>

-Dflyway.domain.placeHolders.plato-alerts-management-services.plato.eventhub.kafka.brokers=<EVETNHUB_KAFKA_BROKERS>

-Dflyway.domain.placeHolders.plato-alerts-management-services.plato.eventhub.zk.nodes=<ZK_NODES>

-Dflyway.domain.placeHolders.plato-alerts-management-services.username=<ALERTS_SCHEMA_USERNAME>

-Dflyway.domain.placeHolders.plato-alerts-management-services.password=<ALERTS_SCHEMA_PASSWORD>

-Dflyway.domain.placeHolders.plato-alerts-management-services.jdbcUrl=<ALERTS_SCHEMA_URL>

-Dflyway.domain.placeHolders.plato-alerts-management-services.schemas=<ALERTS_SCHEMA_NAME>

//Plato Orch Service

-Dflyway.domain.placeHolders.plato-orch-service.server.port=<ORCH_SERVICE_PORT>

-Dflyway.domain.placeHolders. plato-orchestrator.hostname=<CONDUCTOR-EUREKA-HOSTNAME >

//Conductor

-Dconductor.properties=<CONDUCTOR_CONFIG_FILE_PATH>

//Plato Regional Configurator Service

-Dflyway.domain.placeHolders.plato-regional-configurator-services.server.port=<REGIONAL_CONFIGURATOR_SERVICE_PORT>

-Dflyway.domain.placeHolders.plato-regional-configurator-services.locations =

- “db/migration/domain/obrc” (By default)

- “db/migration/domain/obrc, db/migration/domain<YOUR DOMAINS>”

-Dflyway.domain.placeHolders.plato-regional-configurator-services.schemas = “OBRC”

-Dflyway.domain.placeHolders.plato-regional-configurator-services.db.jndi = “jdbc/OBRC”

-Dflyway.domain.placeHolders. plato-regional-configurator-services.username=<OBRC_SCHEMA_USERNAME>

-Dflyway.domain.placeHolders. plato-regional-configurator-services.password=<OBRC_SCHEMA_PASSWORD>

//Common Core Interest Rate Services

-Dflyway.domain.placeHolders.cmc-interest-rate-services.server.port=8020"

-Dflyway.domain.placeHolders.cmc-interest-rate-services.schemas=CMNCORE"

-Dflyway.domain.placeHolders.cmc-interest-rate-services.coherence.enabled=false"

//**Common core NLP services**

-Dflyway.domain.placeholders.cmc-nlp-annotator-services.server.port=<CMC_NLP_ANNOTATOR_SERVICES_PORT>
-Dflyway.domain.placeholders.cmc-nlp-dashboard-widget-services.server.port=<CMC_NLP_DASHBOARD_SERVICES_PORT>
-Dflyway.domain.placeholders.cmc-nlp-model-mngmnt-services.server.port=<CMC_NLP_MODEL_MANGEMENT_PORT>
-Dflyway.domain.placeholders.cmc-nlp-online-processing-services.server.port=<CMC_NLP_ONLINE_PROCESSING_PORT>
-Dflyway.domain.placeholders.cmc-nlp-tag-maint-services.server.port=<CMC_NLP_TAG_MAINTENANCE_PORT>
-Dflyway.domain.placeholders.cmc-nlp-text-extraction-services.server.port=<CMC_NLP_TEXT_EXTRACTION_PORT>
-Dflyway.domain.placeholders.cmc-nlp-txn-log-services.server.port=<CMC_NLP_TXN_LOG_SERVICES_PORT>
-Dflyway.domain.placeholders.cmc-nlp-util-services.server.port=<CMC_NLP_UTIL_SERVICES_PORT>

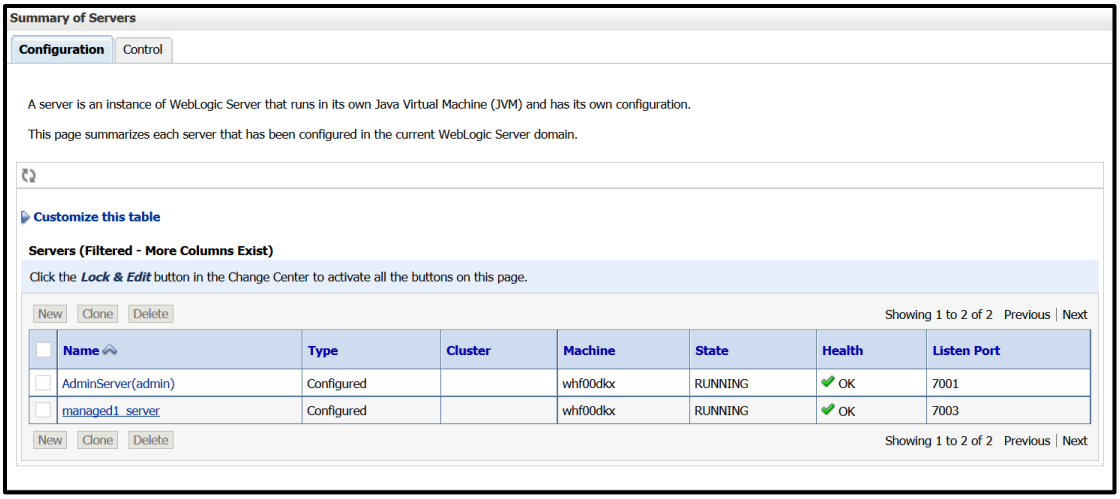
// **Common core NLP Poller service**

-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.port=<Server_Port>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.postingPath=<Posting_Path>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.pollingPath=<Polling_Path>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.pollingEmail=<Polling_Email>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.emailServerHost=<Email_Server_Host>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.emailServerPort=<Email_Server_PORT>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.pollingFrequency=<Polling_Frequency>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.pollerInitialDelay=<Poller_Initial_Delay>
-Dflyway.domain.placeholders.cmc-fc-ai-ml-services.server.emailPassword=<Poller_Email_Password>

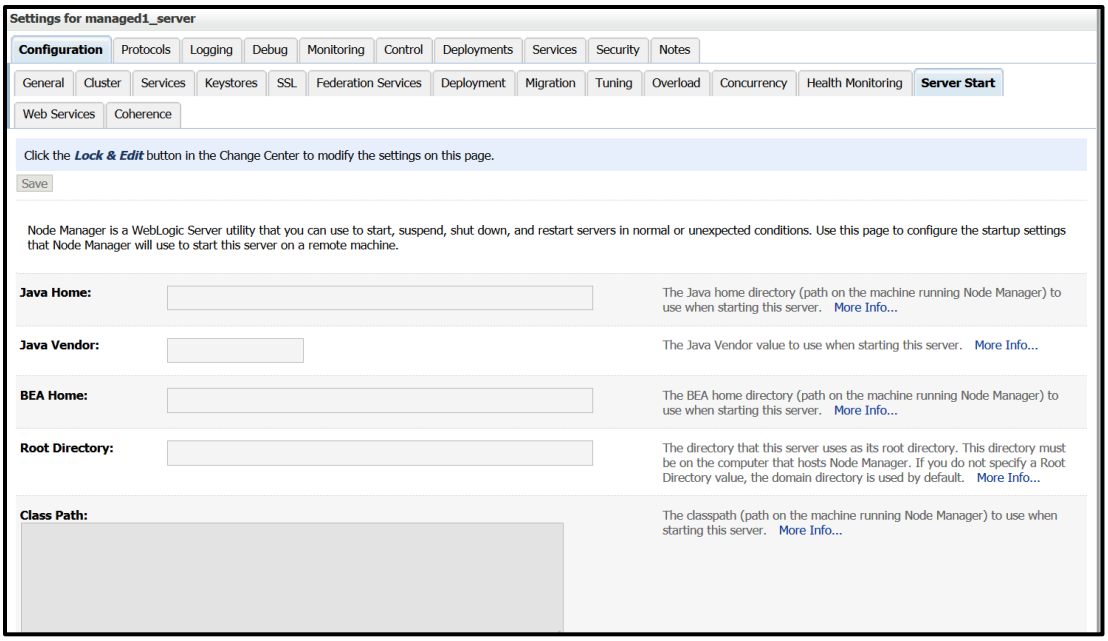
1.2.2 Method 2 – Via passing the -D params in the Server start argument

All the above mentioned -D parameters can be passed through the Server start argument in respective managed server. Perform the following steps:

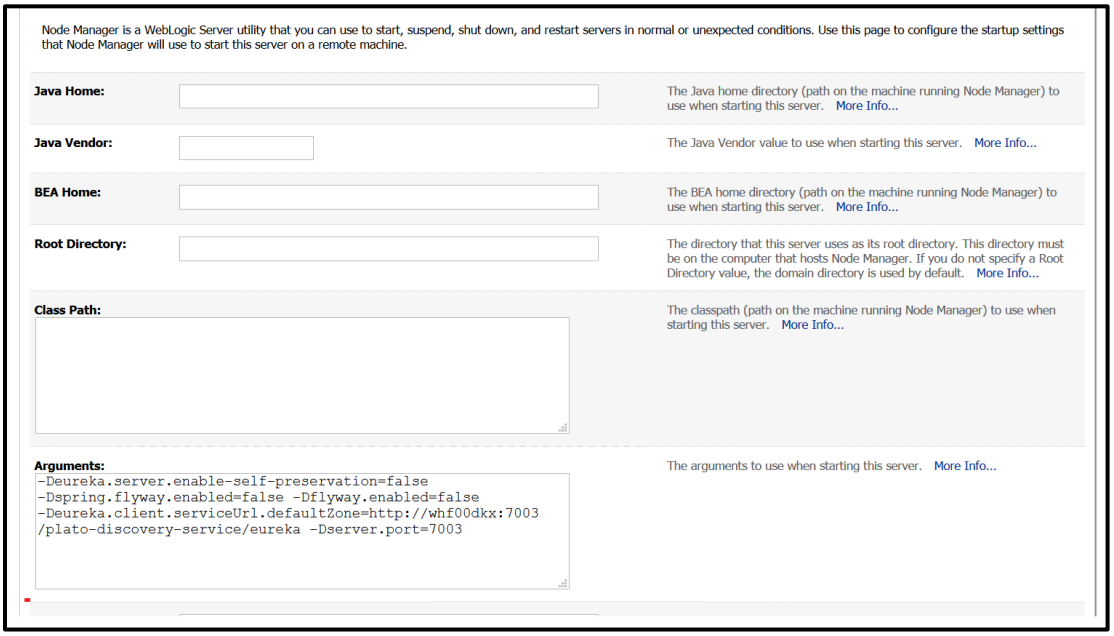
1. Navigate to the Server **Configuration** tab and click managed server to which you want to pass the values.



2. Select **Server Start** tab in the next screen.



3. Edit the **Arguments** field and pass all the environment parameters required for the service to run.



4. Save the configuration and restart the managed server. After you restart, the service can be started or deployed properly.

1.2.3 Method 3 – Using env files and setUserOverrides.sh file

Perform the following steps:

1. Copy the **setUserOverrides.sh** file to each of the <domain>/bin folder. The example of the file is given below:

#!/bin/bash

```
# shellcheck disable=SC1090

# Common functions

set -e -x

config_file=""

PLATO_CONFIG_MANAGED_SERVER_NAME=""

# This file is used only for PLATO-CONFIG service

plato_config_file="${DOMAIN_HOME}/bin/plato-config-deploy.env"

# This file is used for rest of the services

domain_config_file="${DOMAIN_HOME}/bin/domain-config-deploy.env"

if [ -f "$plato_config_file" ] ; then

PLATO_CONFIG_MANAGED_SERVER_NAME=`cat ${DOMAIN_HOME}/bin/plato-config-deploy.env | grep "PLATO_CONFIG_MANAGED_SERVER_NAME" | cut -d'=' -f2`

fi

if [ "${SERVER_NAME}" = "${PLATO_CONFIG_MANAGED_SERVER_NAME}" ] ; then

# This will get executed only for Plato-config service entries

config_file="${plato_config_file}"

else

# This will get executed for all other services

config_file="${domain_config_file}"

fi

if [ -f "$config_file" ]

then

while read -r prop || [ -n "$prop" ]

do

case "$prop" in \#*) continue ;; esac

if [ -z "${prop}" ]; then

continue

else
```

```
PLACEHOLDERS=${PLACEHOLDERS}" "$(echo -D$prop)

PLACEHOLDERS="${PLACEHOLDERS}"

fi

done < "$config_file"

else

echo "$config_file not found. please provide the property file to set -D parameter"

exit 1

fi

PLACEHOLDERS="${PLACEHOLDERS}"

JAVA_OPTIONS="${JAVA_OPTIONS}${PLACEHOLDERS}"

export JAVA_OPTIONS

echo "${JAVA_OPTIONS}"
```

2. Place the **env** files containing all the key value pairs of the **-D params** in the respective <domain>/env folder.

Note: The plato-config-deploy.env file contains all the key value pairs specific only to the plato-config-service and need to be placed in the bin folder of the plato-domain. The domain-config-deploy.env file contains the key-value pairs for the rest of the services and should be placed in each <domain>/bin folder.

The sample for each of the files are given below:

plato-config-deploy.env

```
### Managed server name of plato-config service ###

PLATO_CONFIG_MANAGED_SERVER_NAME=

### plato config flyway connection entries ###

flywayTask=migrate

flyway.enabled=true

spring.flyway.enabled=false

plato-config.flyway.domain.db.username=

plato-config.flyway.domain.db.password=

plato-config.flyway.domain.db.jdbcUrl=

plato-config.flyway.domain.schemas=
```

plato-config.flyway.domain.locations=db/migration/domain/plato,db/migration/domain/sms,db/migration/domain/cmc,db/migration/domain/obvam

Kafka properties for all services

flyway.domain.placeholders.plato.eventhub.broker.hosts=

flyway.domain.placeholders.plato.eventhub.zookeper.hosts=

Kafka Security for all services

flyway.domain.placeholders.plato.eventhub.broker.hosts=

flyway.domain.placeholders.plato.eventhub.zookeper.hosts=

flyway.domain.placeholders.kafka.ssl.truststore.location=

flyway.domain.placeholders.kafka.ssl.truststore.password=

flyway.domain.placeholders.kafka.broker.username=

flyway.domain.placeholders.kafka.broker.password=

common entries for all services

flyway.domain.placeholders.driver.className=oracle.jdbc.driver.OracleDriver

spring.cloud.loadbalancer.ribbon.enabled=false

spring.main.allow-circular-references=true

eureka entries for all services

flyway.domain.placeholders.eureka.host=

flyway.domain.placeholders.eureka.port=

zipkin entries for all services

flyway.domain.placeholders.zipkin.host=

flyway.domain.placeholders.zipkin.port=

plato config flyway placeholder entries

flyway.domain.placeholders.plato-config.username=

flyway.domain.placeholders.plato-config.password=

flyway.domain.placeholders.plato-config.jdbcUrl=

flyway.domain.placeholders.plato-config.schemas=

flyway.domain.placeholders.plato-config.sessionIdleTimeout=

flyway.domain.placeholders.plato-config.sessionIdleWarningTime=

flyway.domain.placeholders.plato-config.environment=

plato api-gateway flyway placeholder entries

flyway.domain.placeholders.api-gateway.host=

flyway.domain.placeholders.api-gateway.username=

flyway.domain.placeholders.api-gateway.password=

flyway.domain.placeholders.api-gateway.jdbcUrl=

flyway.domain.placeholders.api-gateway.schemas=

flyway.domain.placeholders.plato-api-gateway.server.port=

plato api-gateway LDAP flyway placeholder entries

flyway.domain.placeholders.USER.STORE=

flyway.domain.placeholders.LDAP.CORS.allowed.origin=

flyway.domain.placeholders.LDAP.credential.SALT=

flyway.domain.placeholders.JWT.EXPIRY.seconds=

flyway.domain.placeholders.LDAP.url=

flyway.domain.placeholders.LDAP.userId=

flyway.domain.placeholders.LDAP.server.base=

flyway.domain.placeholders.LDAP.server.credential=

flyway.domain.placeholders.LDAP.usersearch.base=

flyway.domain.placeholders.LDAP.user.prefix=

Allowed values for LDAP provider are: EMBEDDED_WEBLOGIC, PLATO

If LDAP is running in weblogic then value should be EMBEDDED_WEBLOGIC

If spring based LDAP(which is run through a jar provided) is used, then the value should be PLATO

flyway.domain.placeholders.LDAP.provider=

flyway.domain.placeholders.TOKEN.autoregenerate=

flyway.domain.placeholders.SSO.enabled=

flyway.domain.placeholders.TOKEN.regeneration.enabled=

plato-ui-config flyway placeholder entries

flyway.domain.placeholders.plato-ui-config.username=

flyway.domain.placeholders.plato-ui-config.password=

flyway.domain.placeholders.plato-ui-config.jdbcUrl=

flyway.domain.placeholders.plato-ui-config.schemas=

flyway.domain.placeholders.plato-ui-config-services.server.port=

flyway.domain.placeholders.apigateway.host=

flyway.domain.placeholders.apigateway.port=

plato-discovery flyway placeholder entries

flyway.domain.placeholders.plato-discovery-service.server.port=

plato-apigateway-router flyway placeholder entries

flyway.domain.placeHolders.plato-apigateway-router.server.port=

flyway.domain.placeHolders.plato-apigateway-router.router.protocol=

flyway.domain.placeHolders.plato-apigateway-router.router.meadmin.port=

plato-orch flyway placeholder entries

flyway.domain.placeholders.plato-orch-service.server.port=

flyway.domain.placeholders.plato-orchestrator.hostname=

plato-feed flyway placeholder entries

flyway.domain.placeholders.plato-feed-services.username=

flyway.domain.placeholders.plato-feed-services.password=

flyway.domain.placeholders.plato-feed-services.jdbcUrl=

flyway.domain.placeholders.plato-feed-services.jndi=jdbc/PLATOFEED

flyway.domain.placeholders.plato-feed-services.schemas=

flyway.domain.placeholders.plato-feed-services.feed.upload.directory=

flyway.domain.placeholders.plato-feed-services.server.port=

plato-batch flyway placeholder entries

flyway.domain.placeholders.plato-batch-server.username=

flyway.domain.placeholders.plato-batch-server.password=

flyway.domain.placeholders.plato-batch-server.jdbcUrl=

flyway.domain.placeholders.plato-batch-server.schemas=

flyway.domain.placeholders.plato-batch-server.server.port=

flyway.domain.placeholders.plato-batch-server.plato.eventhub.kafka.brokers=

flyway.domain.placeholders.plato-batch-server.plato.eventhub.zk.nodes=

flyway.domain.placeholders.plato-batch-server.jndi=jdbc/PLATOBATCH

plato-alerts-management flyway placeholder entries

flyway.domain.placeholders.plato-alerts-management-services.username=

flyway.domain.placeholders.plato-alerts-management-services.password=

flyway.domain.placeholders.plato-alerts-management-services.jdbcUrl=

flyway.domain.placeholders.plato-alerts-management-services.schemas=

flyway.domain.placeholders.plato-alerts-management-services.server.port=

sms flyway placeholder entries

flyway.domain.placeholders.sms-core-services.server.port=

flyway.domain.placeholders.sms.username=

flyway.domain.placeholders.sms.password=

flyway.domain.placeholders.sms.jdbcUrl=

flyway.domain.placeholders.sms.schemas=

cmncore flyway placeholder entries

flyway.domain.placeholders.cmncore.username=

flyway.domain.placeholders.cmncore.password=

flyway.domain.placeholders.cmncore.jdbcUrl=

flyway.domain.placeholders.cmncore.schemas=

flyway.domain.placeholders.cmc-corebanking-adapter-service.server.port=

flyway.domain.placeholders.cmc-currency-services.server.port=

flyway.domain.placeholders.cmc-account-services.server.port=

flyway.domain.placeholders.cmc-base-services.server.port=

flyway.domain.placeholders.cmc-external-virtual-account-services.server.port=

flyway.domain.placeholders.cmc-branch-services.server.port=

flyway.domain.placeholders.cmc-customer-services.server.port=

flyway.domain.placeholders.cmc-external-chart-account-services.server.port=

flyway.domain.placeholders.cmc-external-system-services.server.port=

flyway.domain.placeholders.cmc-advice-services.server.port=

flyway.domain.placeholders.cmc-facilities-services.server.port=

flyway.domain.placeholders.cmc-txn-code-services.server.port=

flyway.domain.placeholders.cmc-settlement-services.server.port=

flyway.domain.placeholders.cmc-businessoverrides-services.server.port=

flyway.domain.placeholders.cmc-resource-segment-orchestrator-service.server.port=

flyway.domain.placeholders.cmc-screenclass-services.server.port=

flyway.domain.placeholders.cmc-datasegment-services.server.port=

flyway.domain.placeholders.cmc-settlements-services.server.port=

flyway.domain.placeholders.cmc-transactioncontroller-services.server.port=

flyway.domain.placeholders.cmc-report-services.server.port=

flyway.domain.placeholders.cmc-nlp-annotator-services.server.port=

flyway.domain.placeholders.cmc-nlp-dashboard-widget-services.server.port=

flyway.domain.placeholders.cmc-nlp-model-mngmnt-services.server.port=

flyway.domain.placeholders.cmc-nlp-online-processing-services.server.port=

flyway.domain.placeholders.cmc-nlp-tag-maint-services.server.port=

flyway.domain.placeholders.cmc-nlp-text-extraction-services.server.port=

flyway.domain.placeholders.cmc-nlp-txn-log-services.server.port=

flyway.domain.placeholders.cmc-nlp-util-services.server.port=

flyway.domain.placeholders.cmc-batch-services.server.port=

flyway.domain.placeholders.cmc-fc-ai-ml-services.server.port=

flyway.domain.placeholders.cmc-fc-ai-ml-services.postingPath=

flyway.domain.placeholders.cmc-fc-ai-ml-services.pollingEmail=

flyway.domain.placeholders.cmc-fc-ai-ml-services.emailServerPort=

flyway.domain.placeholders.cmc-fc-ai-ml-services.emailServerHost=

flyway.domain.placeholders.cmc-fc-ai-ml-services.pollingFrequency=

flyway.domain.placeholders.cmc-fc-ai-ml-services.pollerInitialDelay=

flyway.domain.placeholders.cmc-fc-ai-ml-services.emailPassword=

flyway.domain.placeholders.cmc-fc-ai-ml-services.pollingPath=

biPublisher related cmc-report-service entries

flyway.domain.placeholders.weblogic.userid=

flyway.domain.placeholders.weblogic.password=

flyway.domain.placeholders.biPublisher.host=

flyway.domain.placeholders.biPublisher.port=

flyway.domain.placeholders.runReportTemplate=

flyway.domain.placeholders.emailTemplate=

flyway.domain.placeholders.dms.host=

flyway.domain.placeholders.dms.port=

flyway jndi connection details for shared services placeholder entries

flyway.domain.placeholders.plato.jndi=jdbc/PLATO

flyway.domain.placeholders.plato-config.jndi=jdbc/PLATO

flyway.domain.placeholders.plato-sec.jndi=jdbc/PLATO_SECURITY

flyway.domain.placeholders.plato-ui-config.jndi=jdbc/PLATO_UI_CONFIG

flyway.domain.placeholders.sms.jndi=jdbc/sms

flyway.domain.placeholders.cmncore.jndi=jdbc/CMNCORE

flyway jndi connection details for obvam services placeholder entries

flyway.domain.placeholders.eie.jndi=jdbc/EIE

flyway.domain.placeholders.eie.server.port=

flyway.domain.placeholders.eie.schemas=

flyway.domain.placeholders.elm.jndi=jdbc/ELM

flyway.domain.placeholders.elm.server.port=

flyway.domain.placeholders.elm.schemas=

flyway.domain.placeholders.vam.jndi=jdbc/VAM

flyway.domain.placeholders.vam.server.port=

flyway.domain.placeholders.vam.schemas=

flyway.domain.placeholders.vac.jndi=jdbc/VAC

flyway.domain.placeholders.vac.server.port=

flyway.domain.placeholders.vac.schemas=

flyway.domain.placeholders.vab.jndi=jdbc/VAB

flyway.domain.placeholders.vab.server.port=

flyway.domain.placeholders.vab.schemas=

flyway.domain.placeholders.vae.jndi=jdbc/VAE

flyway.domain.placeholders.vae.server.port=

flyway.domain.placeholders.vae.schemas=

flyway.domain.placeholders.eda.jndi=jdbc/EDA

flyway.domain.placeholders.eda.server.port=

flyway.domain.placeholders.eda.schemas=

flyway.domain.placeholders.vai.jndi=jdbc/VAI

flyway.domain.placeholders.vai.server.port=

flyway.domain.placeholders.vai.schemas=

flyway.domain.placeholders.van.jndi=jdbc/VAN

flyway.domain.placeholders.van.server.port=

flyway.domain.placeholders.van.schemas=

flyway.domain.placeholders.vap.jndi=jdbc/VAP

flyway.domain.placeholders.vap.server.port=

flyway.domain.placeholders.vap.schemas=

flyway.domain.placeholders.vas.jndi=jdbc/VAS

flyway.domain.placeholders.vas.server.port=

flyway.domain.placeholders.vas.schemas=

flyway.domain.placeholders.vat.jndi=jdbc/VAT

flyway.domain.placeholders.vat.server.port=

flyway.domain.placeholders.vat.schemas=

flyway.domain.placeholders.vaj.server.port=

flyway.domain.placeholders.platoorch.domain.jndi=jdbc/PLATO-O

flyway.domain.placeholders.platoorch.domain.schemas=

flyway.domain.placeholders.plato.alerts.email.userId=

flyway.domain.placeholders.plato.alerts.email.password=

flyway.domain.placeholders.plato.alerts.cmc.userId=

flyway.domain.placeholders.plato.alerts.cmc.branchCode=

flyway.domain.placeholders.plato.alerts.cmc.appId=

flyway.domain.placeholders.plato-rule.hostname=

flyway.domain.placeholders.plato-rule-service.server.port=

flyway.domain.placeholders.platorule.domain.jndi=

flyway.domain.placeholders.platorule.domain.schemas=

flyway.domain.placeholders.obrh.import.data.disable-modify=

flyway.domain.placeholders.cmc-obrh-services.kafka.server.path=

flyway.domain.placeholders.cmc-obrh-services.zookeeper.server.path=

flyway.domain.placeholders.cmc.schemas=

flyway.domain.placeholders.cmc-nlp-opennlp-services.server.port=

flyway.domain.placeholders.cmc-nlp-maintenance-services.server.port=

flyway.domain.placeholders.cmc-nlp-pipeline-services.server.port=

flyway.domain.placeholders.cmc-nlp-docview-services.server.port=

flyway.domain.placeholders.cmc-ml-indb-services.server.port=

flyway.domain.placeholders.cmc-obrh-services.kafka.enabled=

flyway.domain.placeholders.cmc-sla-services.server.port=

flyway.domain.placeholders.cmc-obcbs-services.schemas=

flyway.domain.placeholders.obcbs.server.port=

flyway.domain.placeholders.orch.cmc.brn=

flyway.domain.placeholders.orch.cmc.user=

flyway.domain.placeholders.orch.enableDynamicAllocation=

flyway.domain.placeholders.orch.enableSLA=

flyway.domain.placeholders.report-service.server.port=

flyway.domain.placeholders.report-service.hostname=

flyway.domain.placeholders.report-service.domain.jndi=jdbc/PLATOREPORT

flyway.domain.placeholders.report-service.template-metadata-directory=

flyway.domain.placeholders.report-service.output-directory=

flyway.domain.placeholders.report-service.fop-config-file=

generic entries for all services

spring.cloud.config.uri=

apigateway.url=

service.logging.environment=

service.logging.path=

domain-config-deploy.env

domain config flyway connection entries

flywayTask=migrate

flyway.enabled=true

spring.flyway.enabled=false

generic entries for all services

spring.cloud.config.uri=

apigateway.url=

service.logging.environment=

service.logging.path=

1.2.4 Method 4 – Workflow Configuration

Follow the below steps to create a workflow:

1. Metadata of the workflow creation. The sample DSL for workflow creation is given below:

   {

```
"name": "initialTest",

"description": "Test workflow",

"version": 4,

"tasks": [

{

"name": "TEST",

"taskReferenceName": "TESTING3",

"description": "TESTING2",

"inputParameters": {

"FUNCTIONAL_CODE": "TEST_FA_ILS_REGTN2",

"processRefNo":
"${workflow.input.transactionModel.txnIdentification.processRefNo}",

"processName": "Testing Process2",

"processCode":
"${workflow.input.transactionModel.txnIdentification.processName}",

"transactionModel": "${workflow.input.transactionModel}",

"stage": "TESTING2",

"priority":
"${workflow.input.transactionModel.transactionData.moduleData.taskPrioirty}",

"applicationDate":
"${workflow.input.transactionModel.txnIdentification.applicationDate}",

"applicationNumber":
"${workflow.input.transactionModel.txnIdentification.processRefNo}",

"processRefNumber":

"${workflow.input.transactionModel.txnIdentification.processRefNo}",

"branch": "${workflow.input.transactionModel.txnIdentification.branchCode}",

"user": "${workflow.input.transactionModel.txnIdentification.currentUser}",

"customerNumber":
"${workflow.input.transactionModel.transactionData.moduleData.customerId}",
```

```
            "amount":
        "${workflow.input.transactionModel.transactionData.moduleData.amount}",

            "currencyCode":
    "${workflow.input.transactionModel.transactionData.moduleData.currency}",

            "TASK_OUTCOMES": [

            "PROCEED"

            ],

            "moduleCode": "OBTFPM",

            "customFilter": [

            {

            "key": "contractRefNo",

            "label": "Back Office Reference"

            },

            {

            "key": "otherRefNo",

            "label": "External Reference"

            }

        ]

        },

        "type": "WAIT",

        "startDelay": 0,

        "optional": false,

        "asyncComplete": false

    }

    ],

    "outputParameters": {

    "stage": "CLMO_FA_SNPOAR_APPEN",
```

"taskOutcome": "PROCEED_WITH_PARTICIPANT"

},

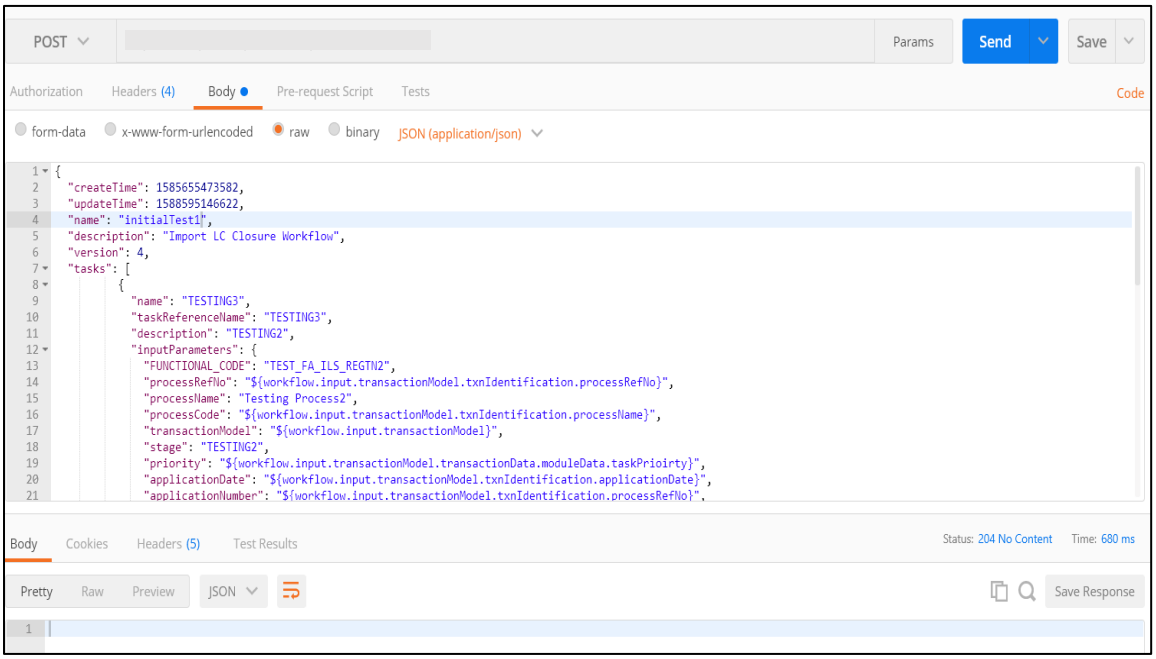
"schemaVersion": 2,

"restartable": true,

"workflowStatusListenerEnabled": false

}

Call the API (/api/metadata/workflow) and pass the DSL in body. The following screen depicts the sample workflow:



2. Workflow Creation

Call the API (/api/workflow) to create the workflow. This API provides the information to the workflow metadata which we have created using previous call.

Body:

{

"name": "initialTest",

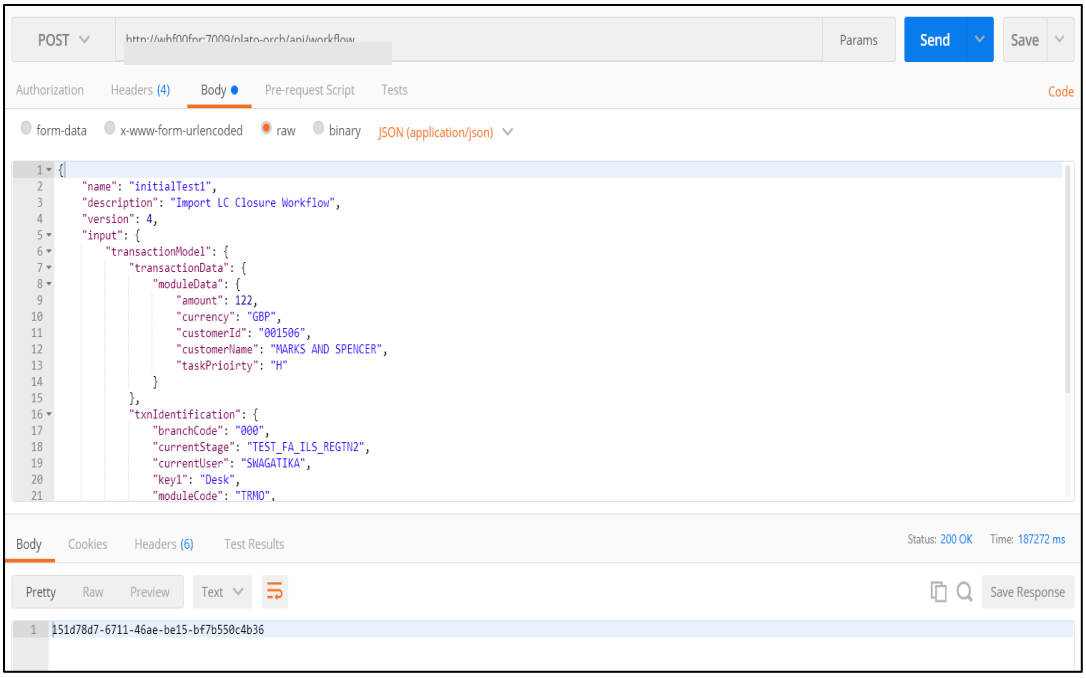
"description": "initialTest",

"version": 4,

"input": {

```
"transactionModel": {
"transactionData": {
"moduleData": {
"amount": 122,
"currency": "GBP",
"customerId": "001506",
"customerName": "MARKS AND SPENCER",
"taskPrioirty": "H"
 }
},
"txnIdentification": {
"branchCode": "000",
"currentStage": "TEST_FA_ILS_REGTN2",
"currentUser": "SWAGATIKA",
"key1": "Desk",
"moduleCode": "TRMO",
"processName": "Testing Process2",
"processRefNo": "300ILCI012260",
"applicationDate": 1588582461960,
"taskOutcome": "PROCEED",
"taskPrioirty": "H"
 }
      }
   }
}
```


The following screen depicts the sample workflow:



1.3 How to Create Domain and Cluster Configuration

This section contains the following sub-sections:

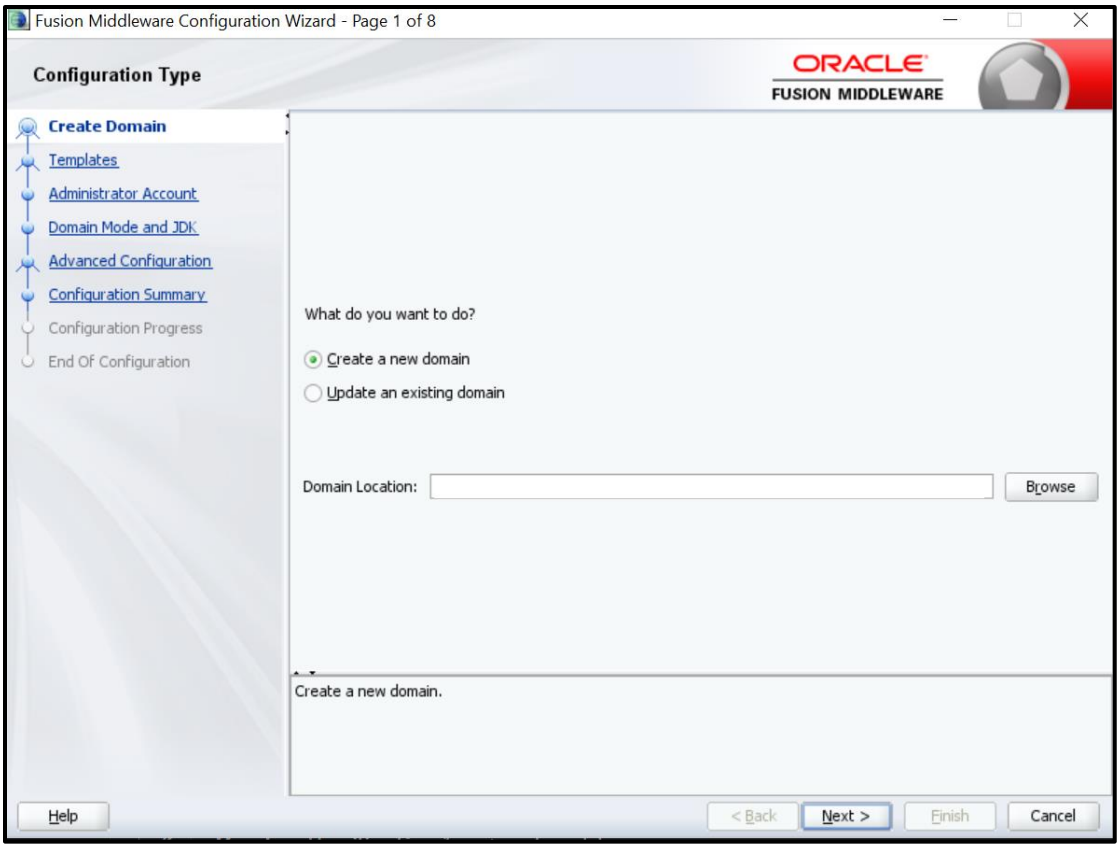
- Domain Creation Configuration
- Post Domain Creation Configurations

1.3.1 Domain Creation Configuration

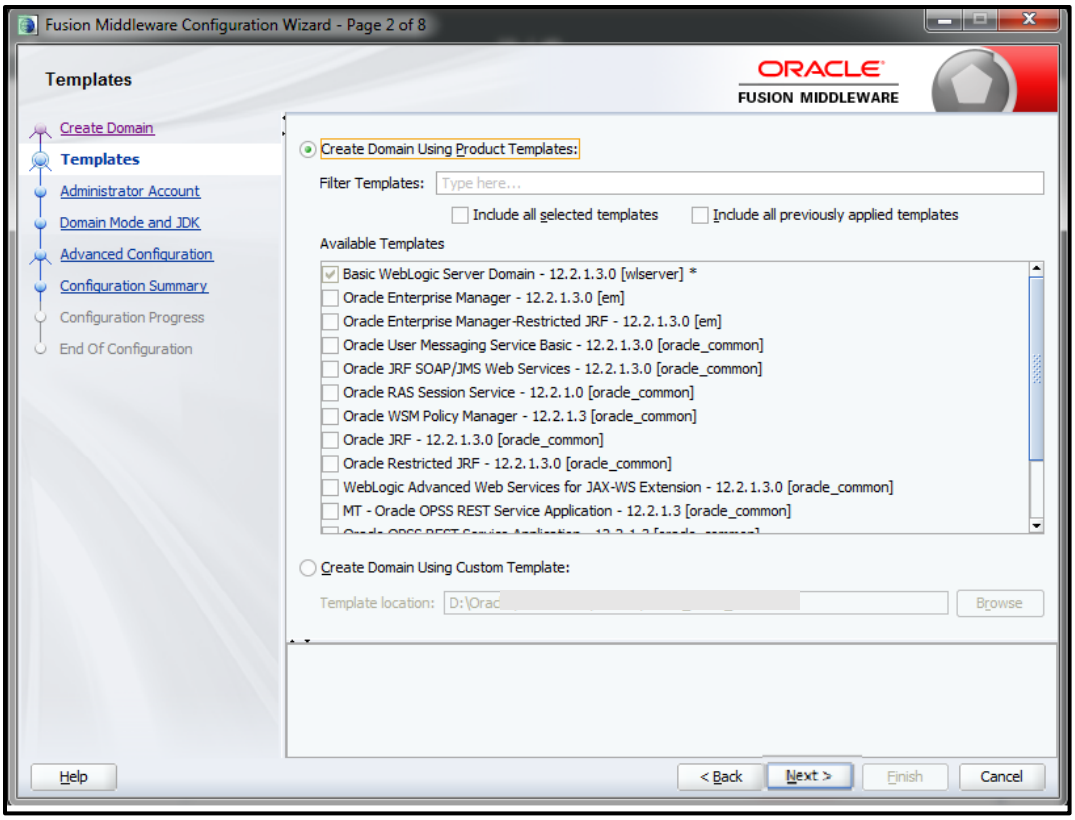
Perform the following steps for domain and cluster configuration:

Note: Name need not to be same as provided in Screenshot.

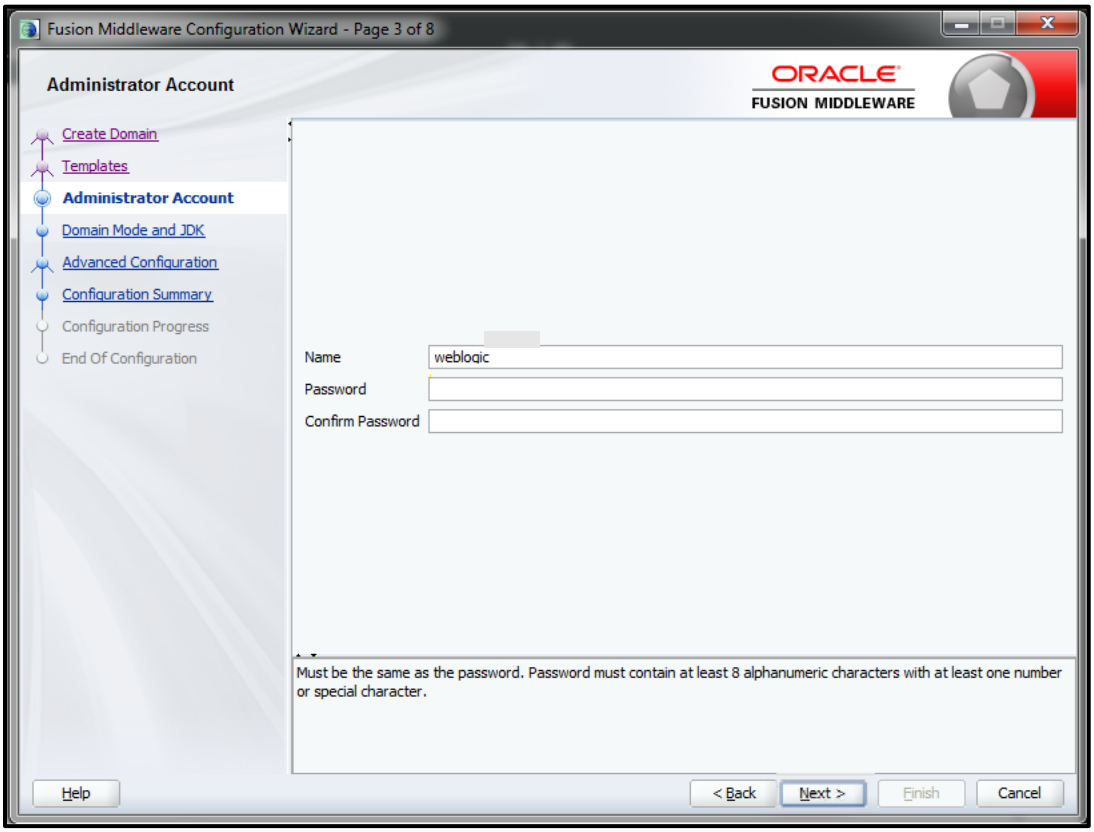
1. Open **/oracle_common/common/bin** and run **config.cmd** (or **.sh** if operating system is linux). Create domain with required cluster and server configurations. Refer to the screenshots below.
2. Select **Create a new domain** and provide domain name. For example, **platoinfra_domain**.



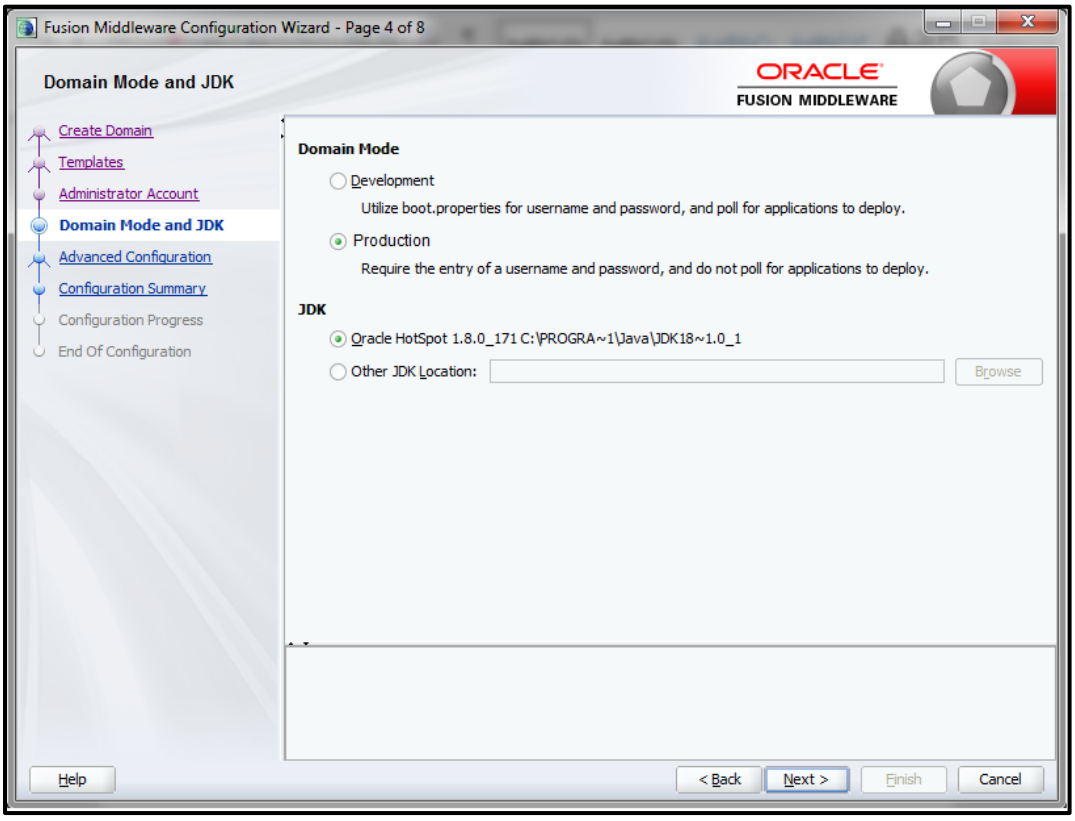
3. Click **Next** to create simple domain with default templates.



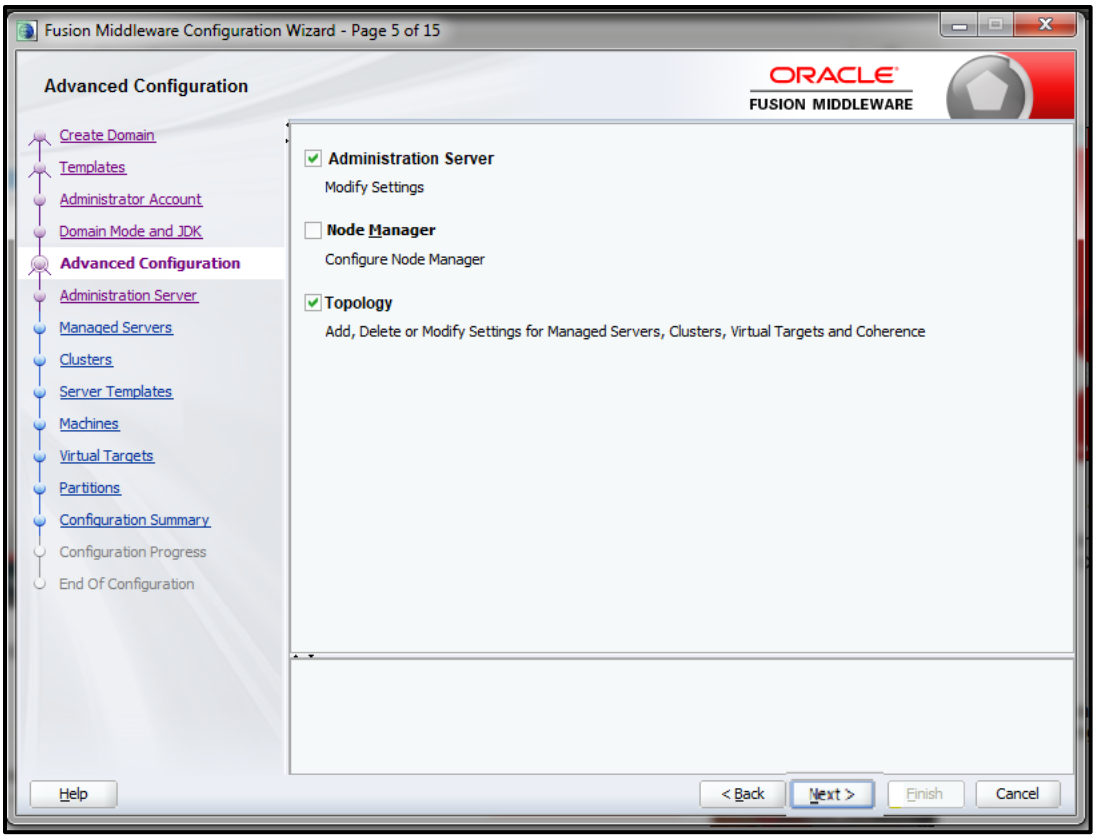
4. Set password and confirm, click **Next** to proceed.



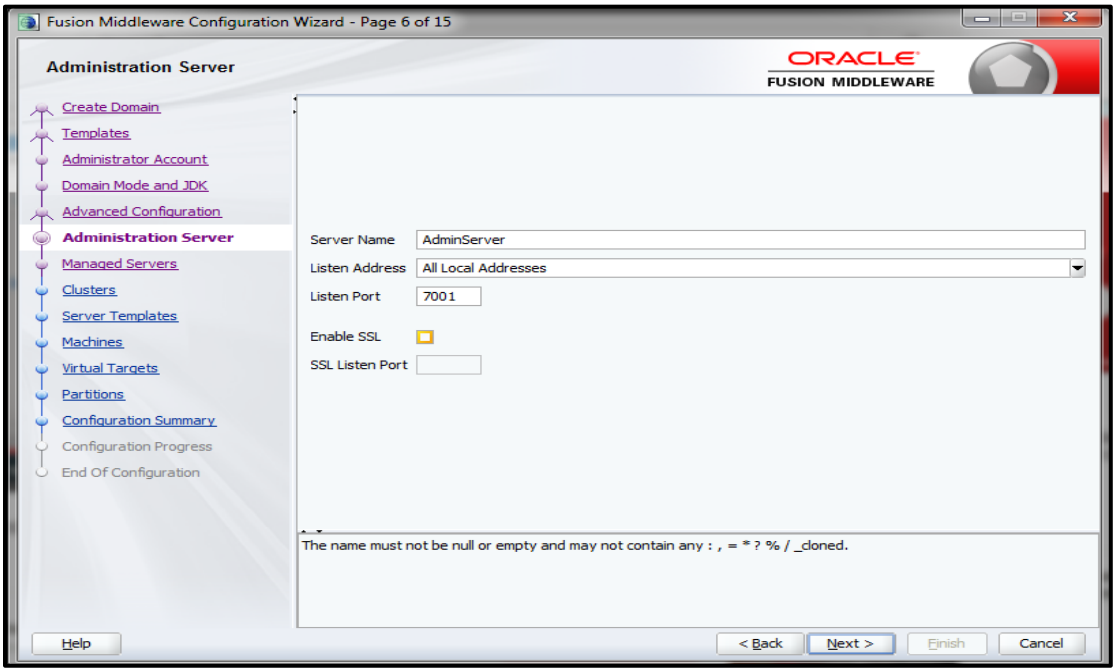
5. Select **Domain Mode** as **Production** and select **JDK**.



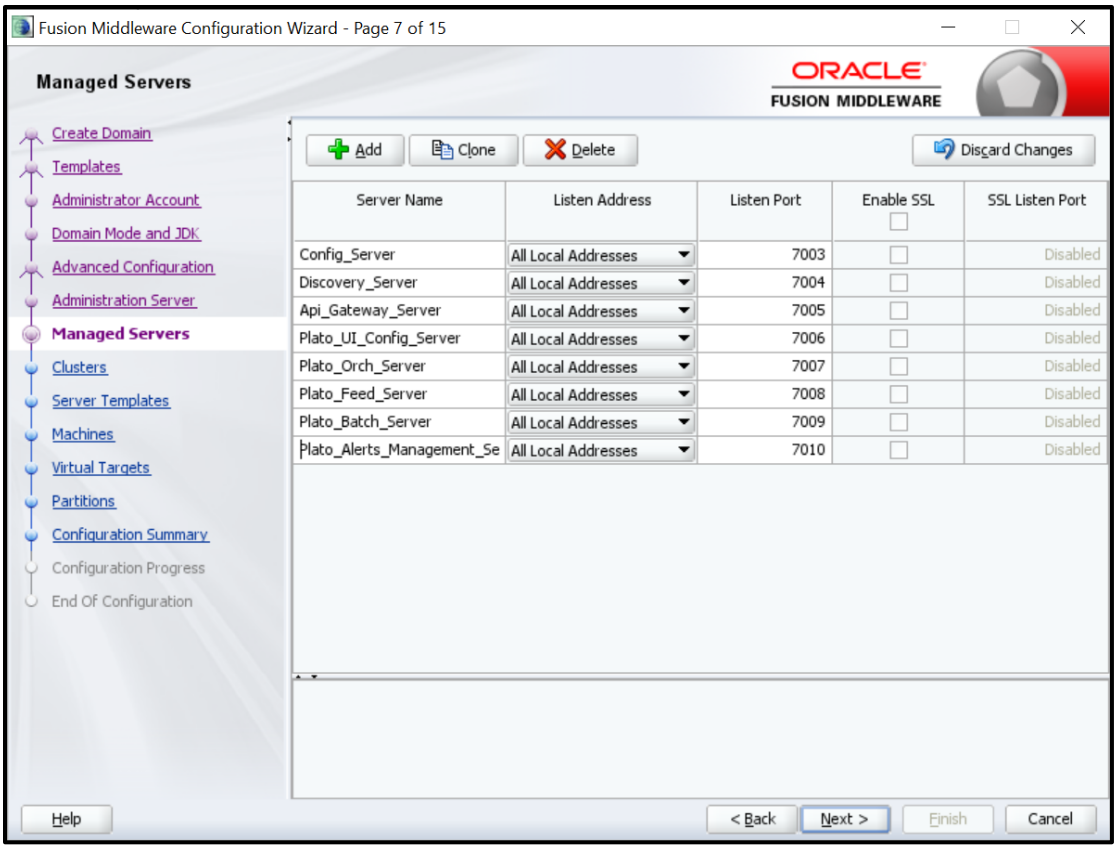
6. Select **Administration Server** and **Topology** in advanced configurations.



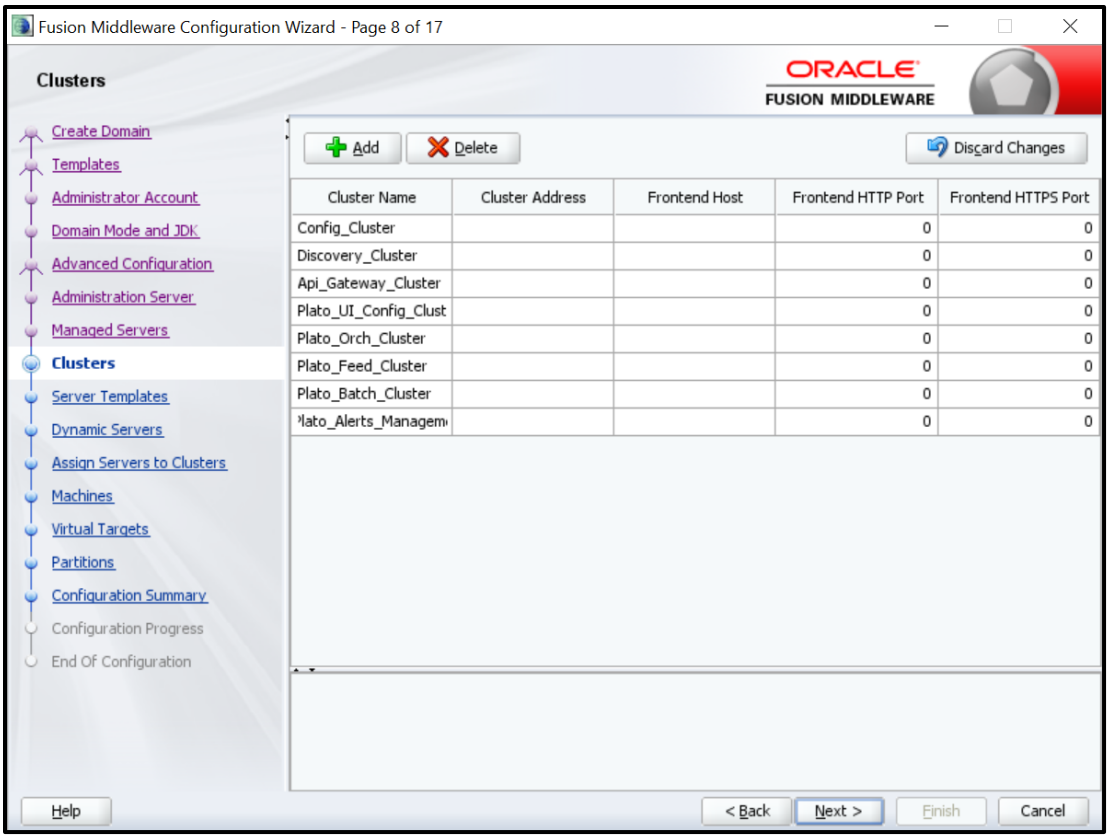
7. Edit the port and host configurations as required and click **Next**.



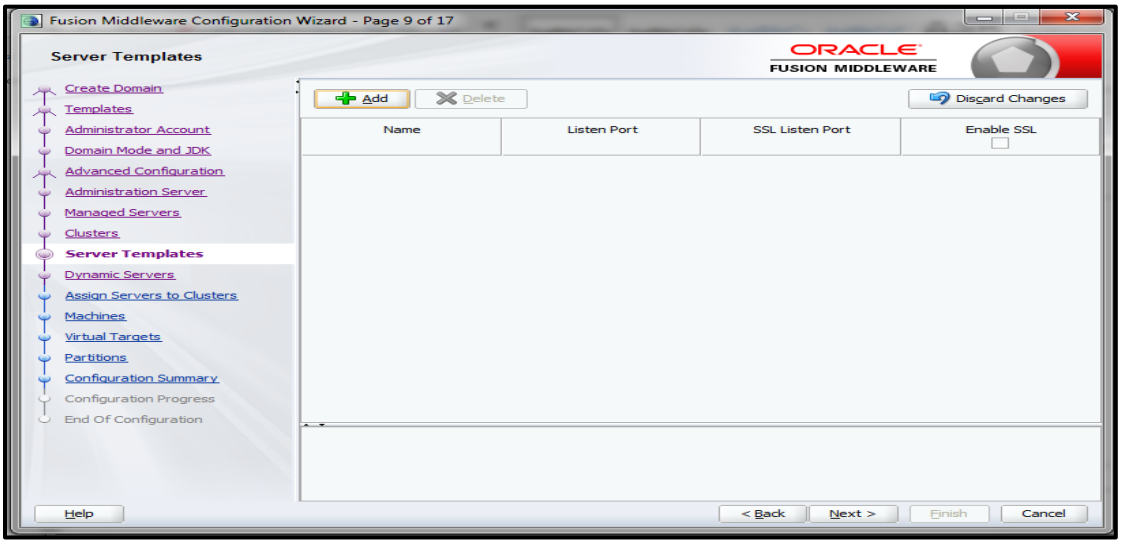
8. Add managed servers and provide meaningful **Server Name**, edit listen address and port as required.



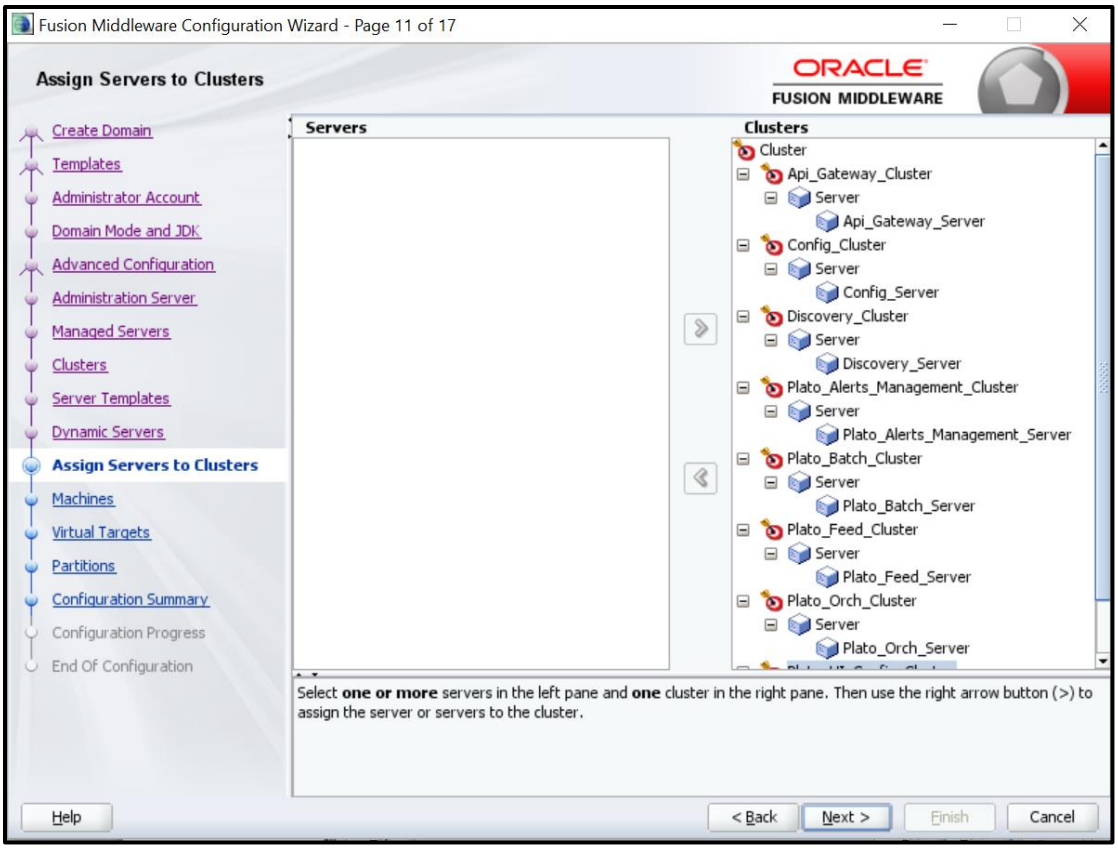
9. Add clusters one for each **managed servers**.



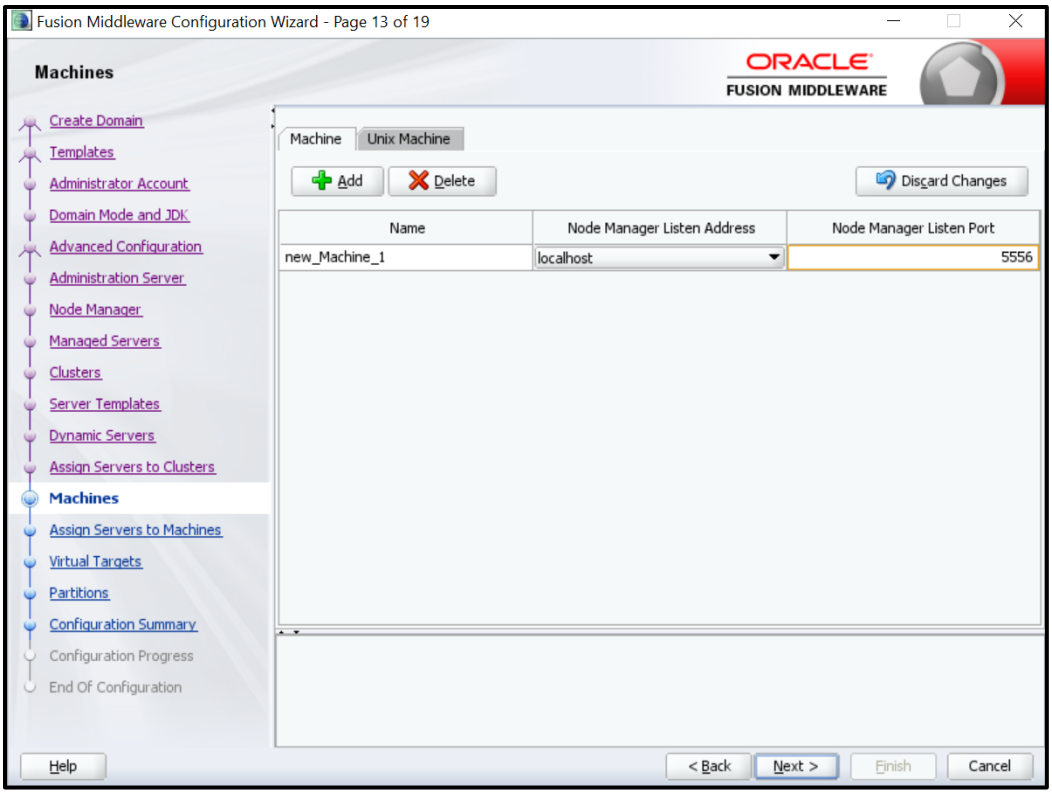
10. Skip **Server Templates** and **Dynamic Servers**.



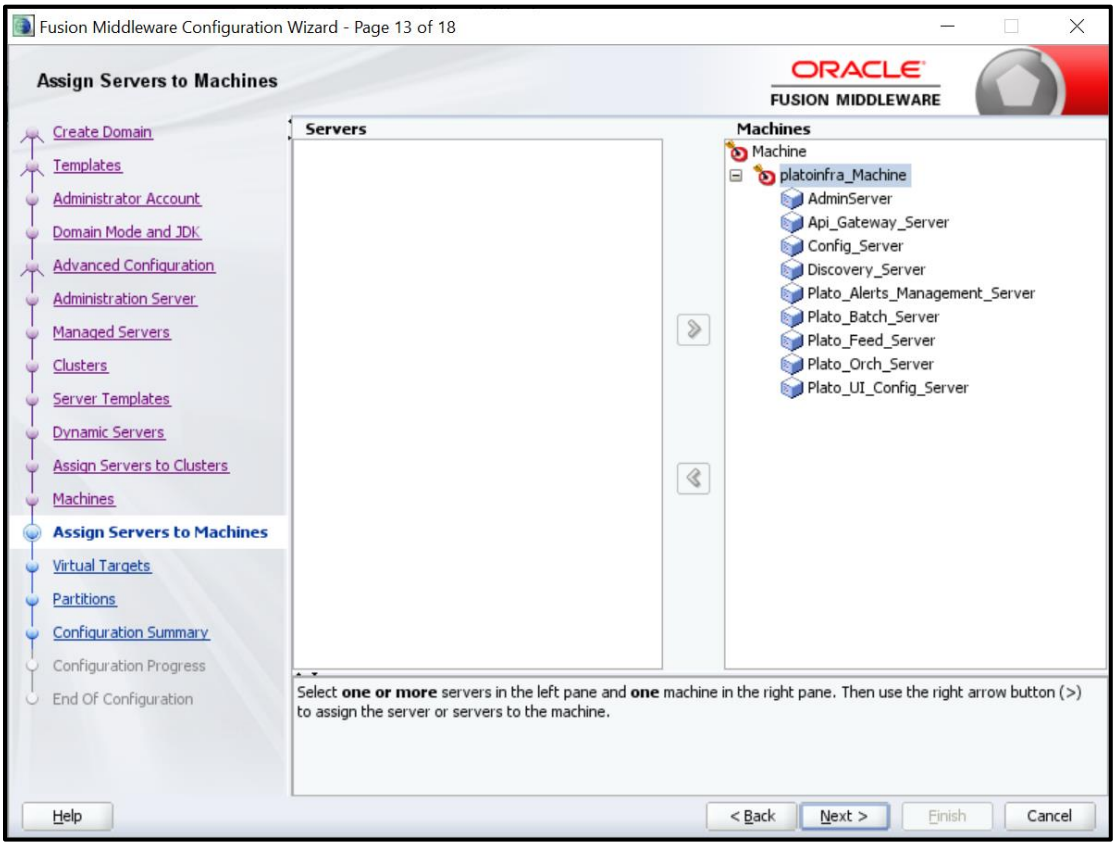
11. Assign clusters with servers.



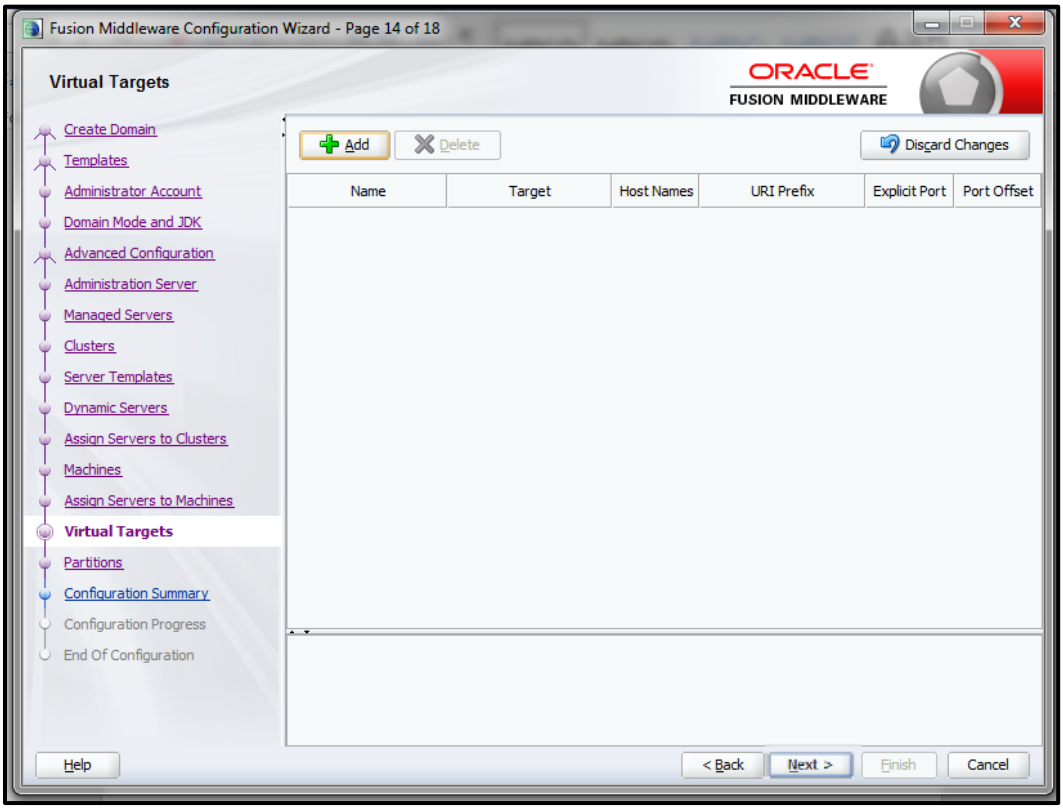
12. Add **Machine/Unix Machine** based on operating system and configure **Name**, **Node Manager Listen Address** and **Node Manager Listen Port** as required.

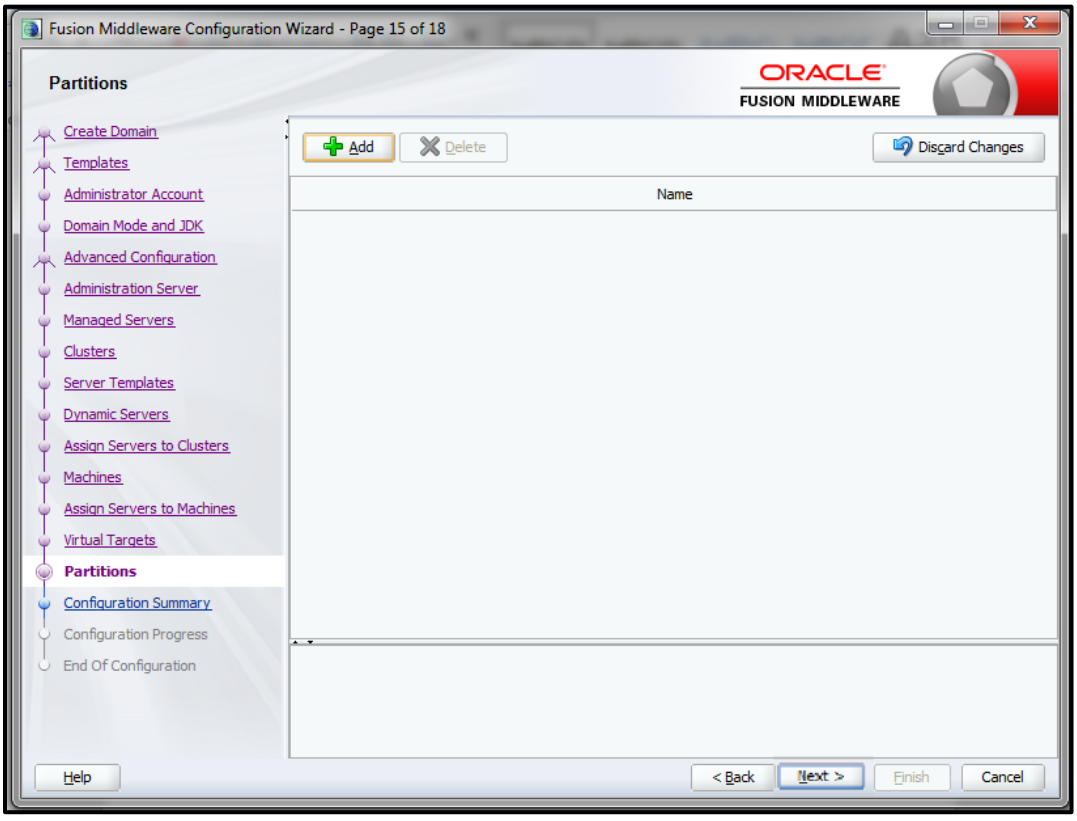


13. Map all managed servers under the machine created.

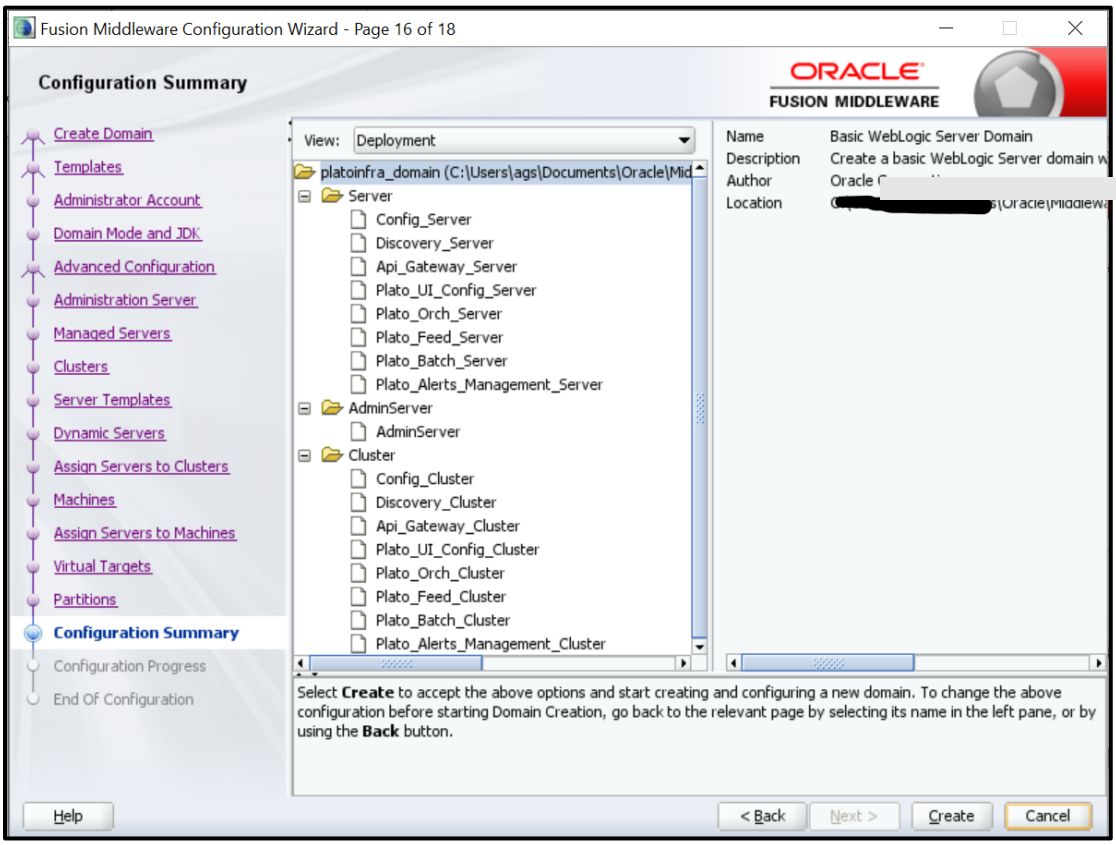


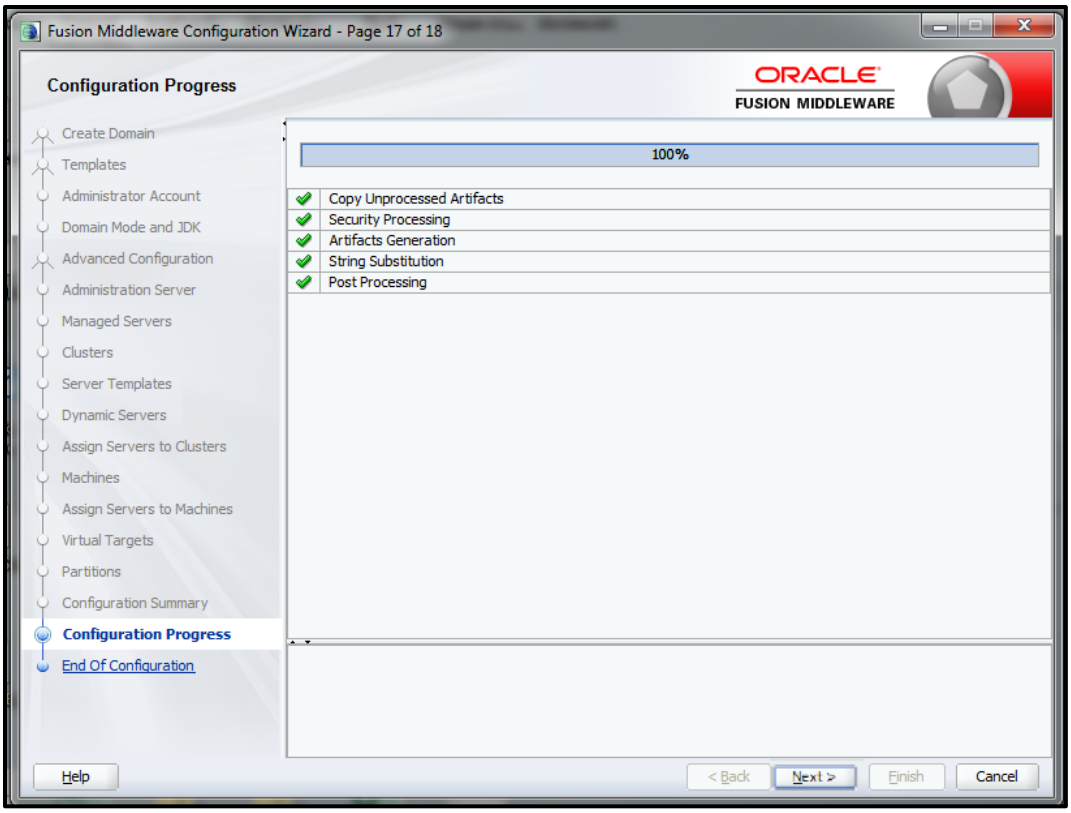
14. Skip or configure **Virtual Targets** and **Partitions** as required.



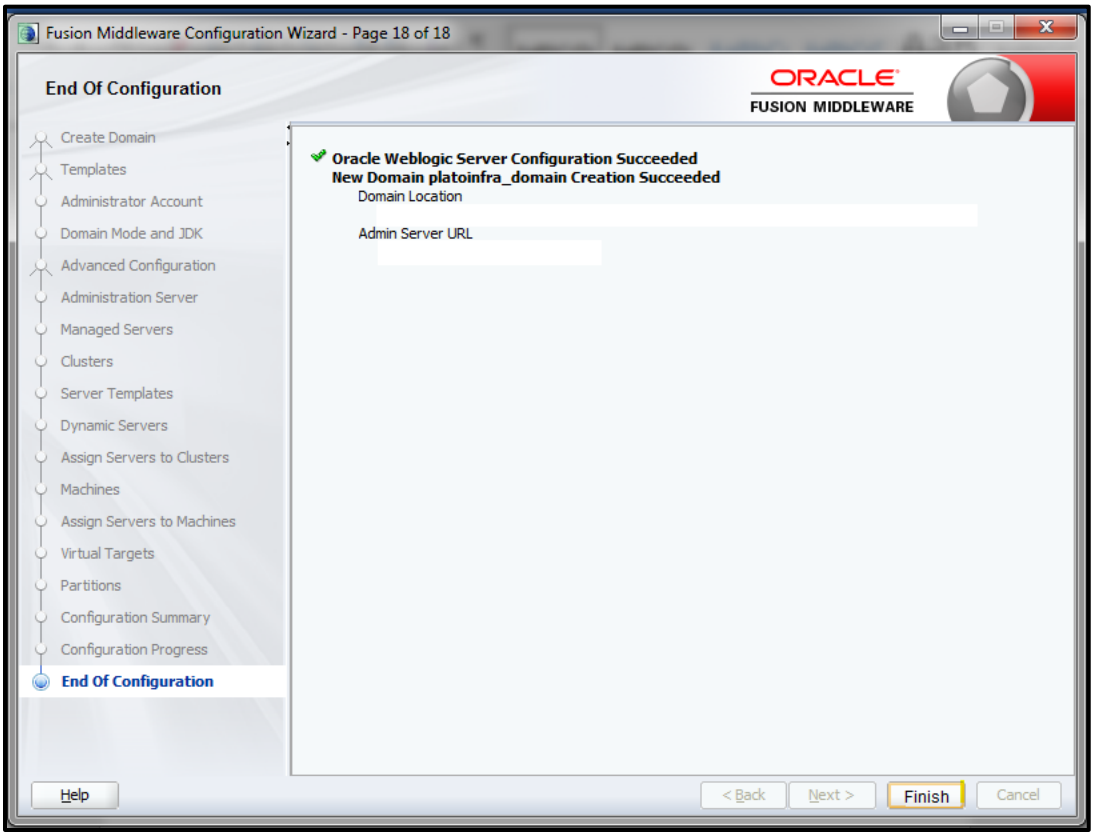


15. Check the **Configuration Summary** and confirm creating domain.





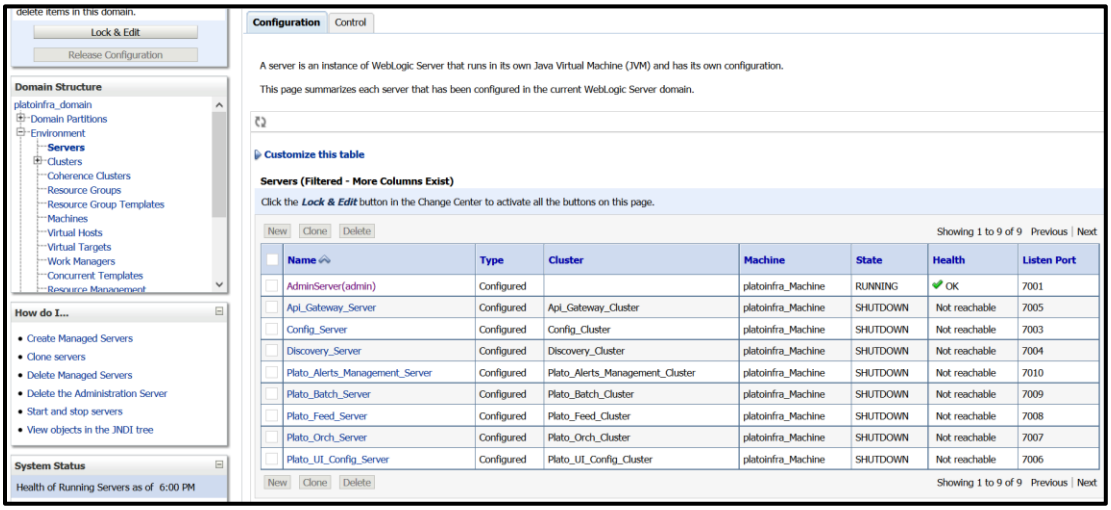
16. Click **Finish** to complete the procedure.

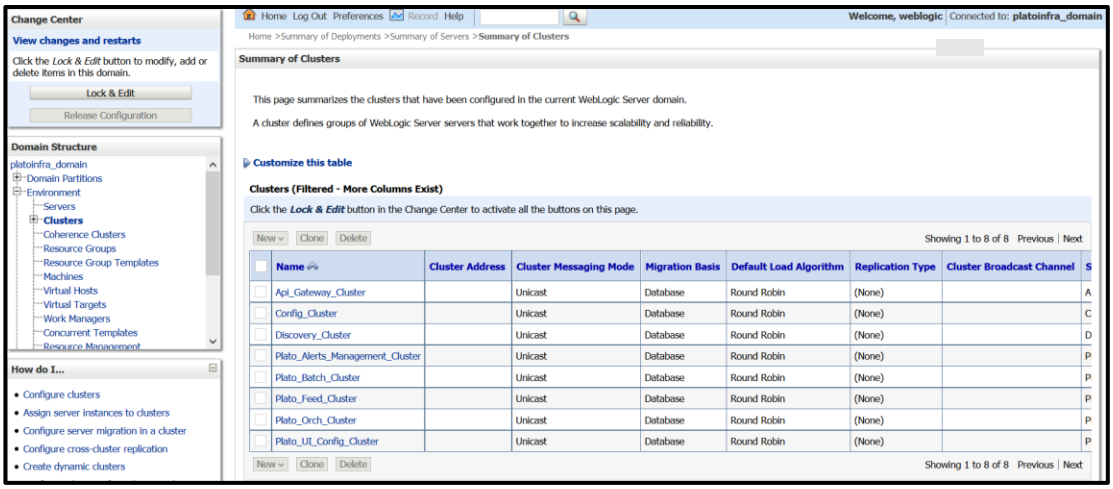


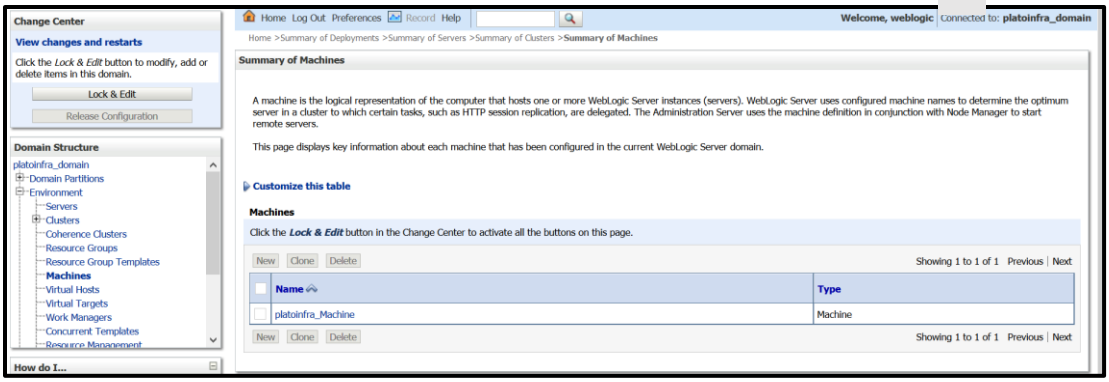
1.3.2 Post Domain Creation Configurations

Once finished, refer oracle fusion middleware documents for more details on how to start admin server, node manager and managed servers.

1. Open **/user_projects/domain/platoinfra_domain/bin**
2. Perform all the Environment Setup steps such as setting -D parameters, Embedded Weblogic Setup and changes required for OAuth.
3. Run **startWeblogic.cmd** (or **.sh** if operating system is linux).
4. Open **/user_projects/domains/platoinfra_domain/bin**.
5. Run **setNMJavaHome.cmd** (or **.sh** if operating system is linux).
6. Open **/user_projects/domains/platoinfra_domain/nodemanager**.
7. Edit **nodemanager.properties** as required (securelistner = false if ssl and keystore is not given).
8. In admin console, select the following options in sequential order:
    a. **Machines**
    b. **platoinfra_Machine**
    c. **Node Manager**
    d. **Type**
    e. **Plain**
    f. **Save**
9. Open **/user_projects/domains/platoinfra_domain/bin**.
10. Run **startNodeManager.cmd** (or **.sh** if operating system is linux)
11. Start all managed servers.
12. Login to console and verify servers and clusters. Refer to the screenshots below:



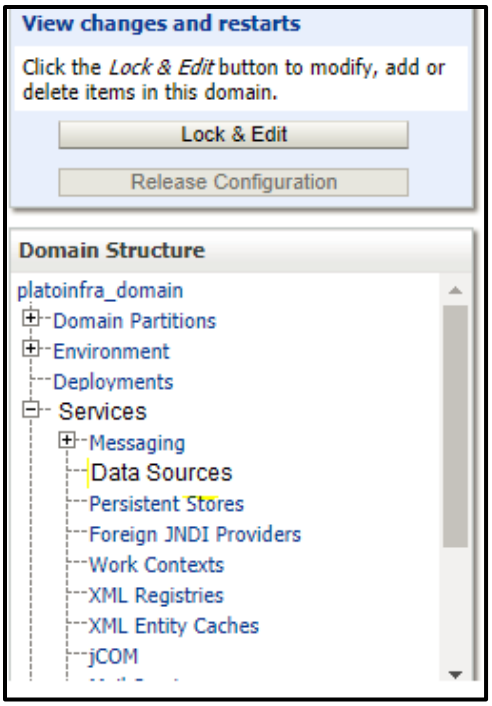


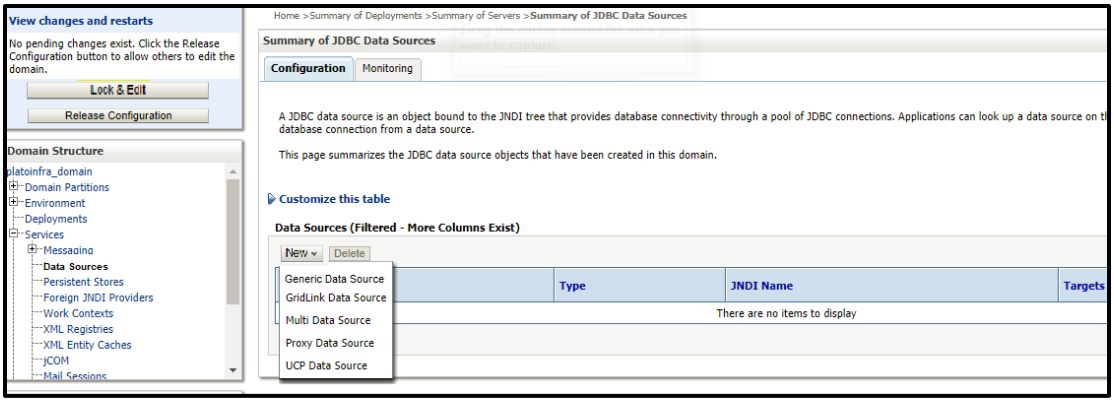


1.4 How to Create Datasource

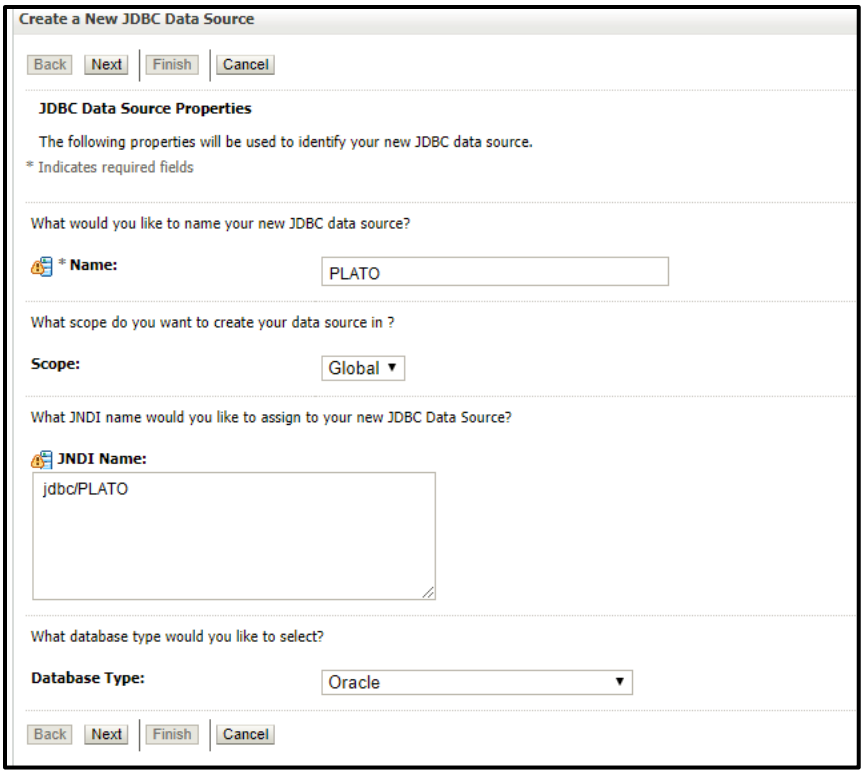
Perform the following steps to create data source:

1. Start **AdminServer**, **Node Manager** and make sure all the **managed servers** (targets) are in running mode.
2. Select the following options in sequential order:
   a. **Services**
   b. **Datasources**
   c. **New**
   d. **Generic Datasource**

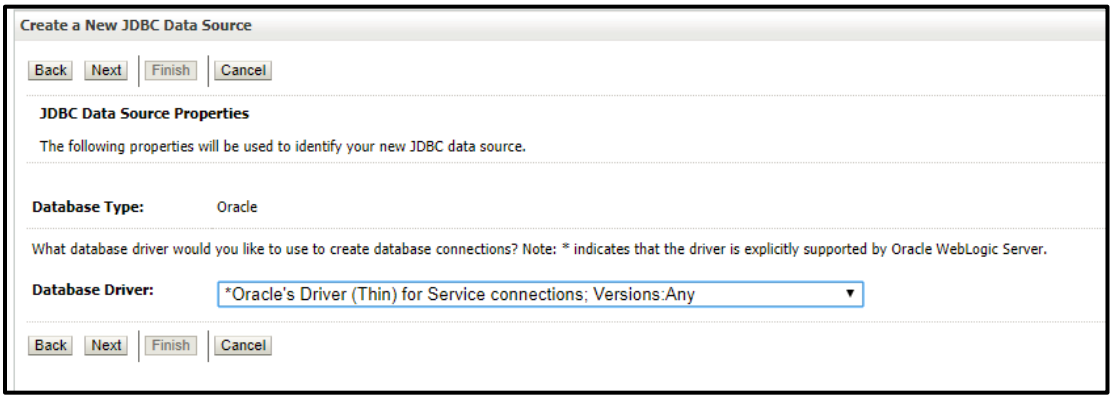




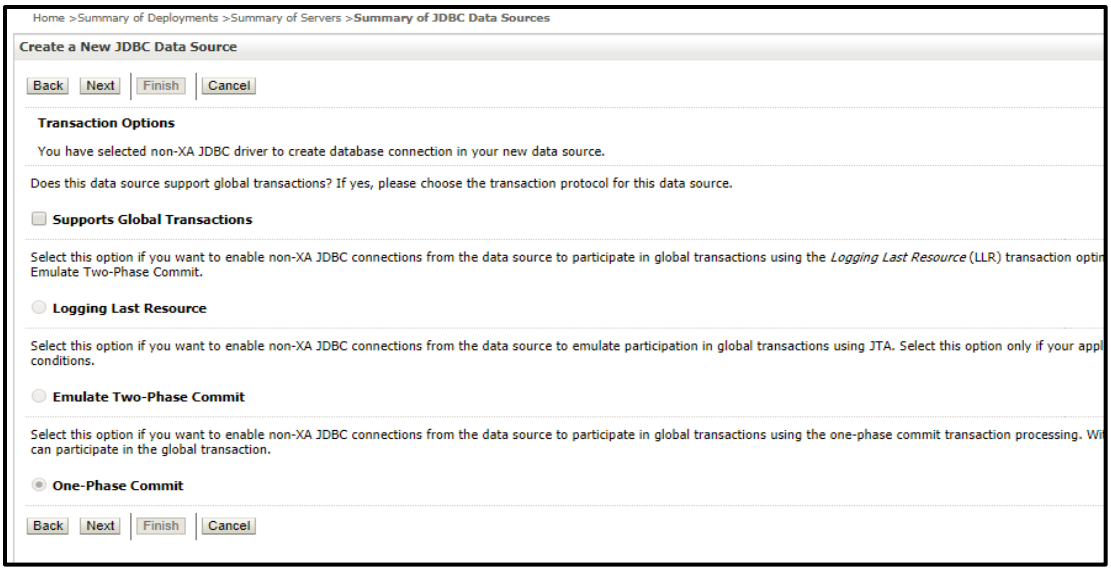
3. Give datasource **Name** and **JNDI Name**, and click **Next**.



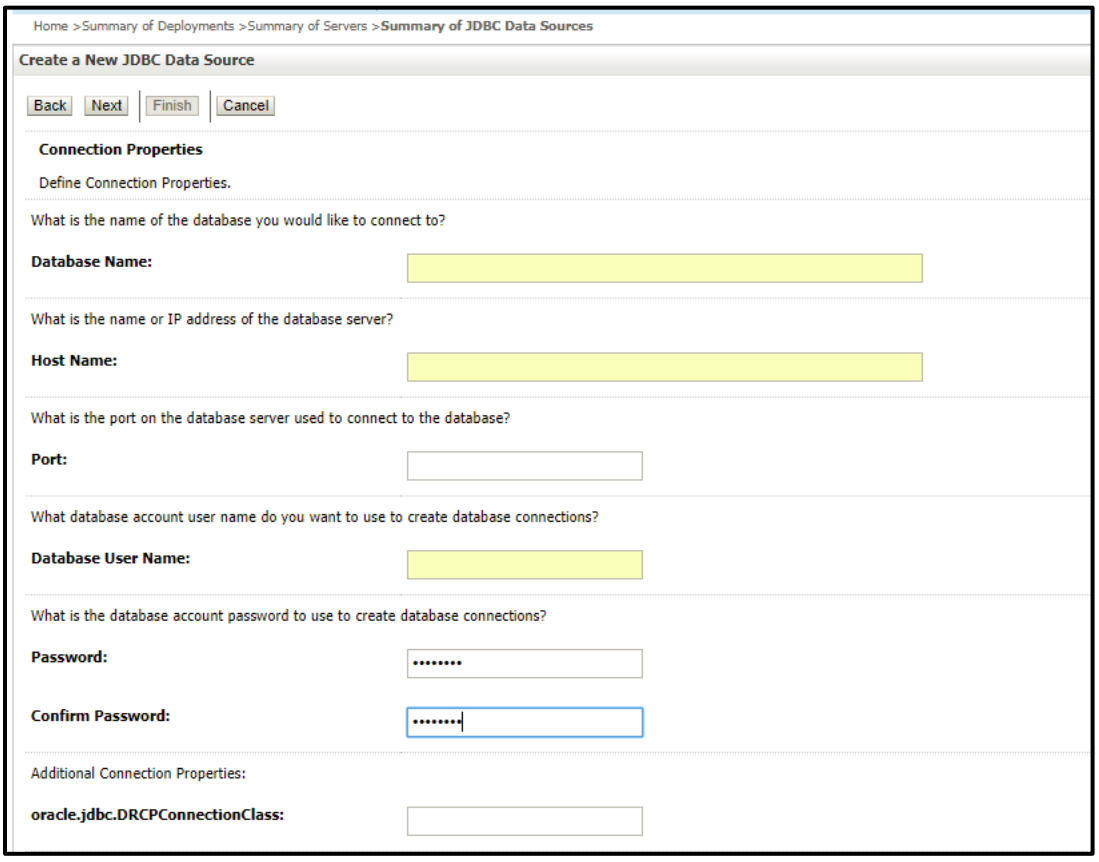
4. Select **Thin for Service Connections** (Instant) and click N**ext**.

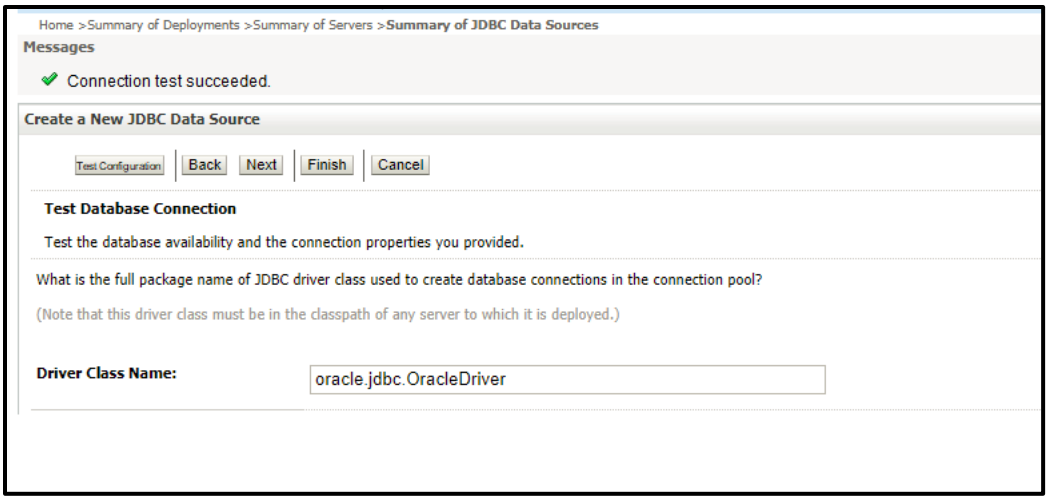


5. Uncheck support for Global Transactions.

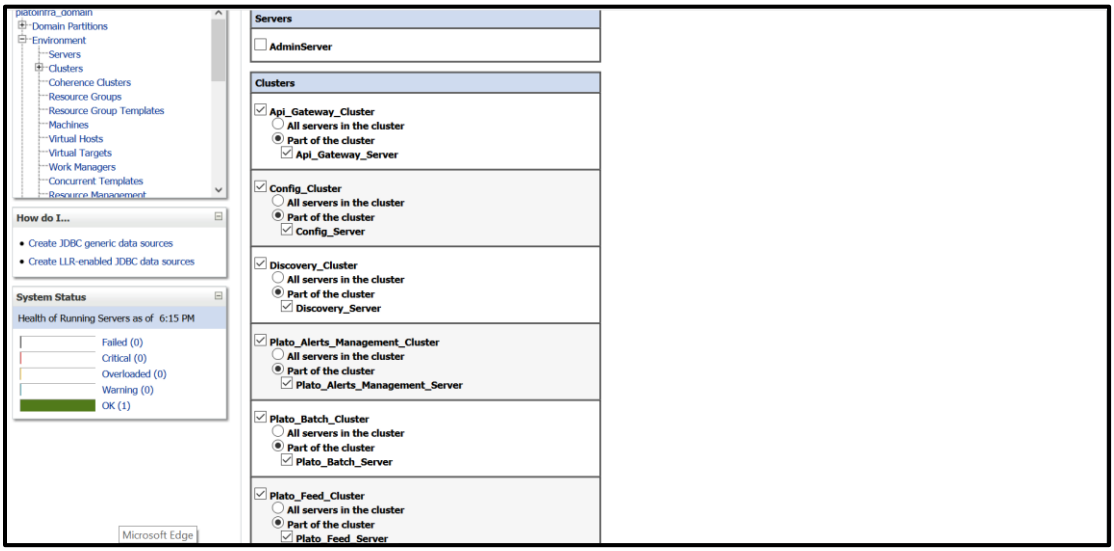


6. Give database connection details and click **Next** to test connection.

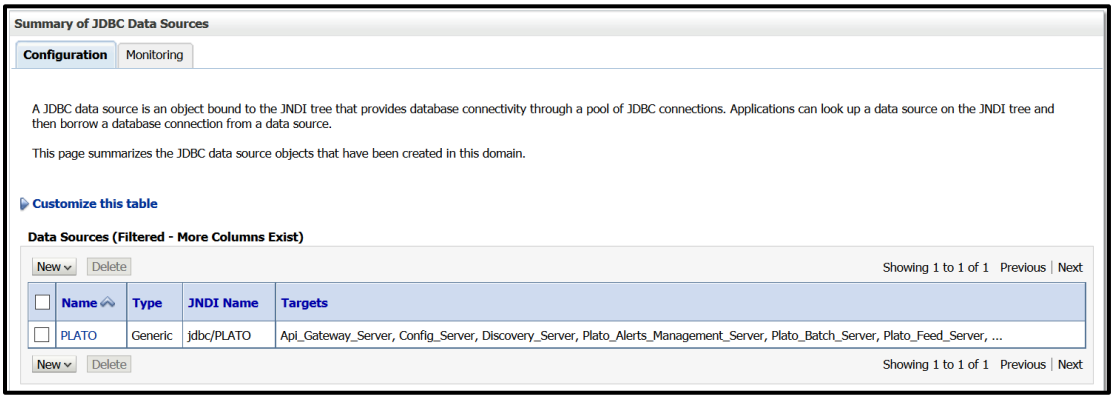




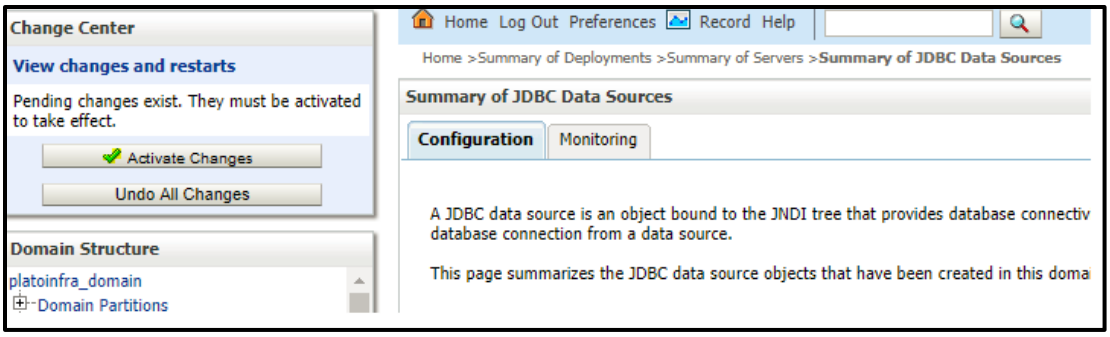
7. Select targets to deploy data source.

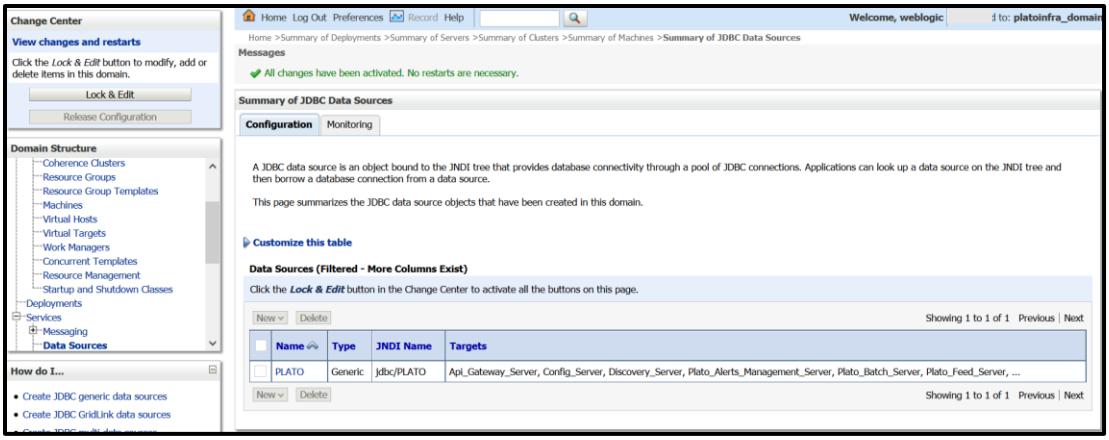


8. View created **Data Sources,** verify **JNDI Name** and **Targets**.

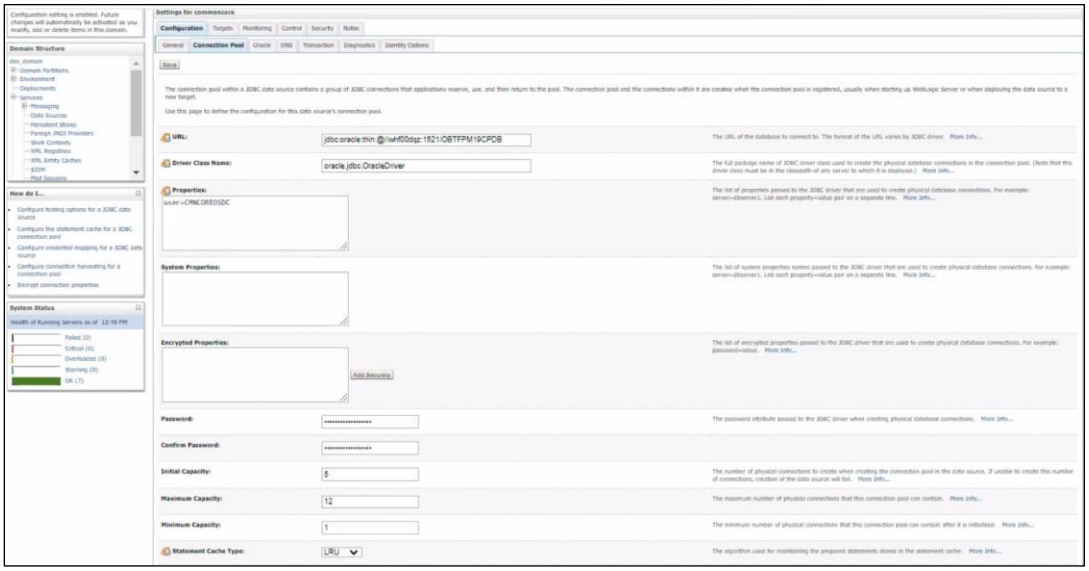


9. Click **Activate Changes** after confirming details.





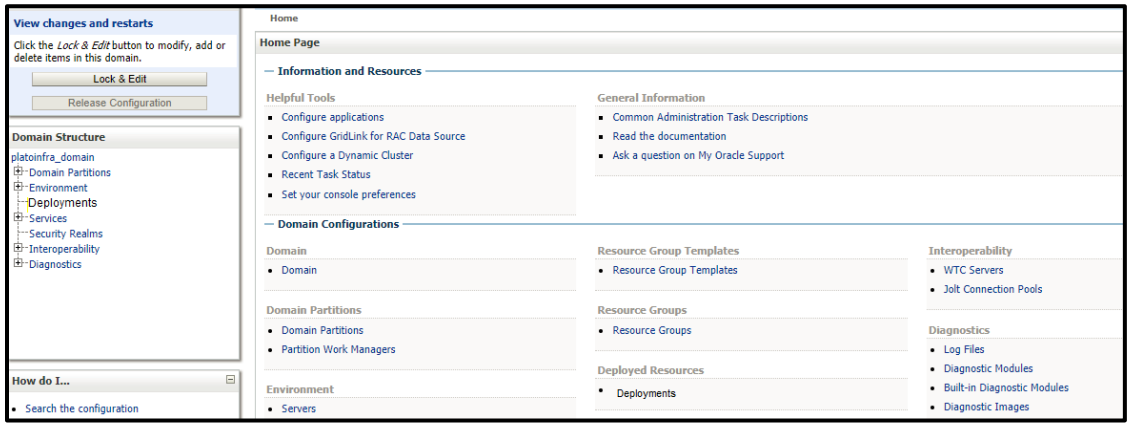
10. Post creating Data Source, click **Data source** under **Services** menu.
11. Select the specific Data sources one by one.
12. Navigate to **Connection Pool** tab under **Configuration** section and update the initial capacity, Minimum capacity and Max capacity as explained in the below screenshot



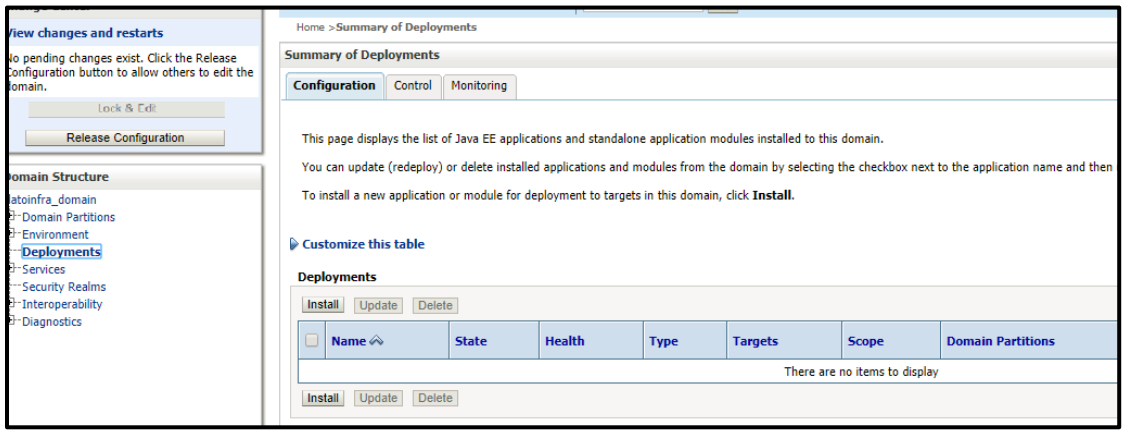
1.5 How to Deploy Application

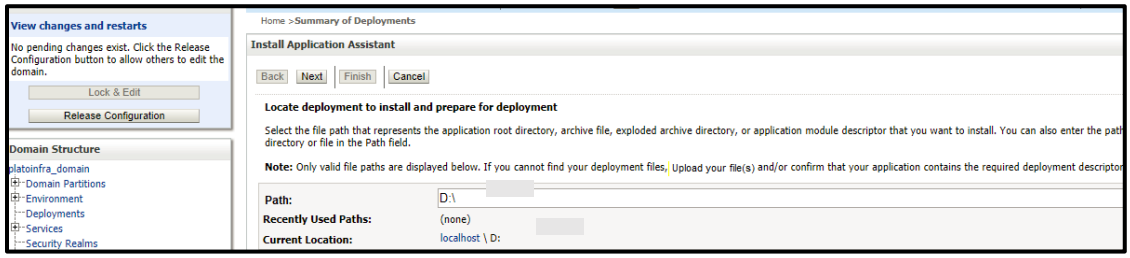
Steps to Deploy archives as application on weblogic is same for all the above except for managed server and domain, where we deploy will differ. Perform the following steps to see how deployment of archive as application is done on weblogic:

1. Navigate to left menu and select **Deployments**.

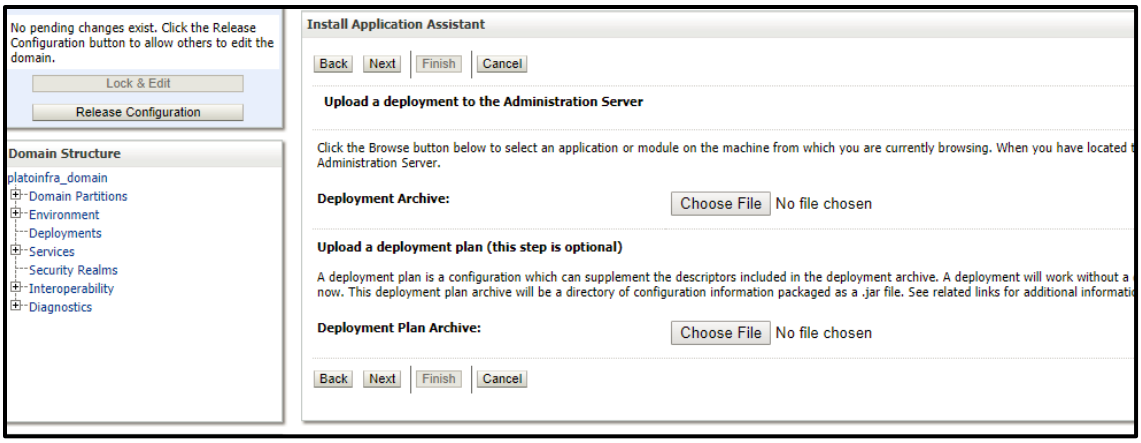


2. Click on **Lock and Edit** and then click **Install**.

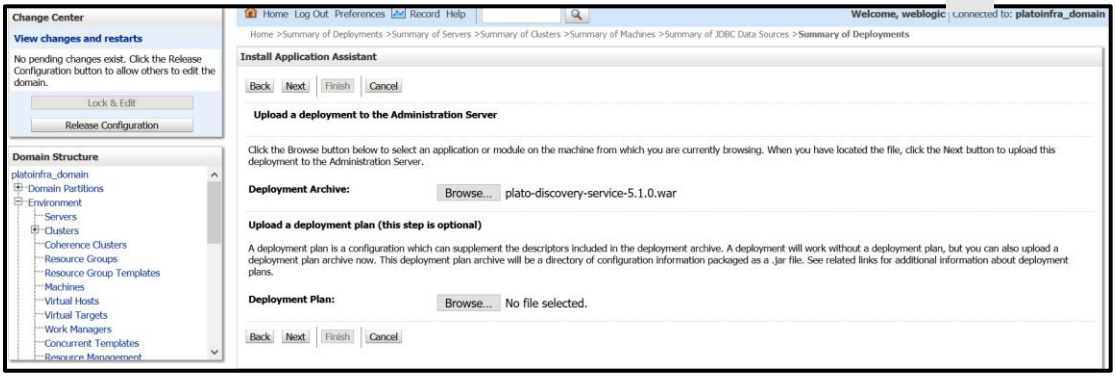


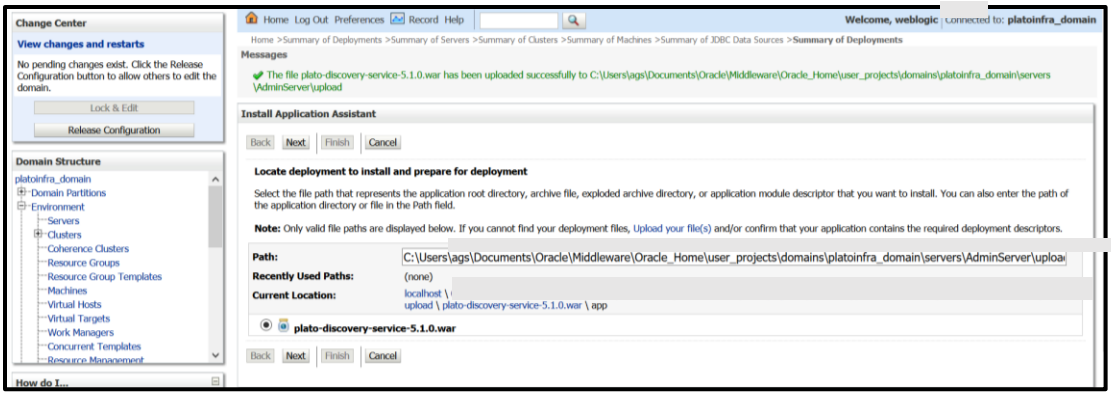


3. Click **Upload your file(s)** to select archive, **Choose File** and click **Next**.

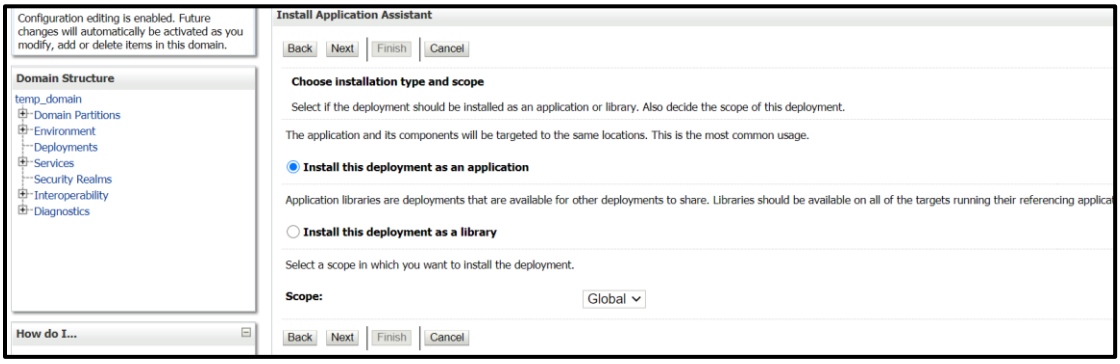


4. After archive is uploaded, click **Next**.

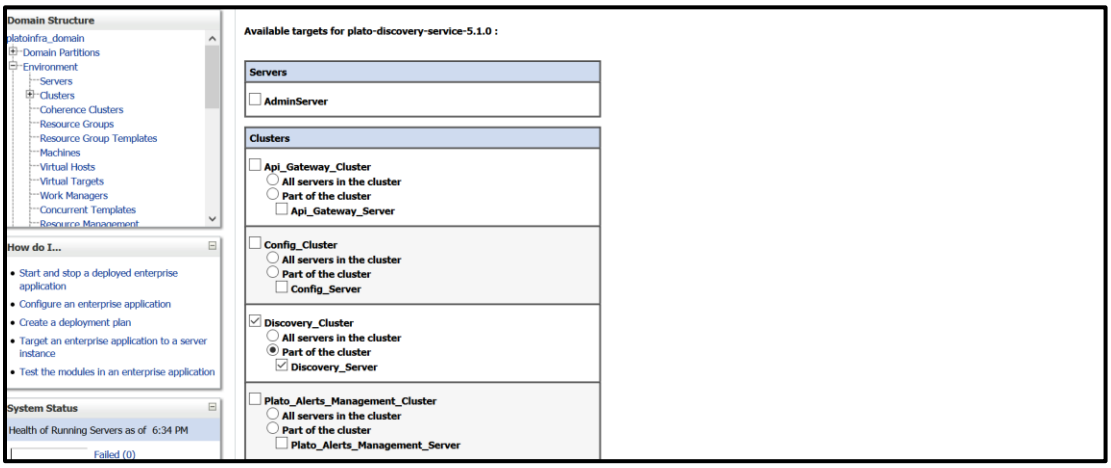


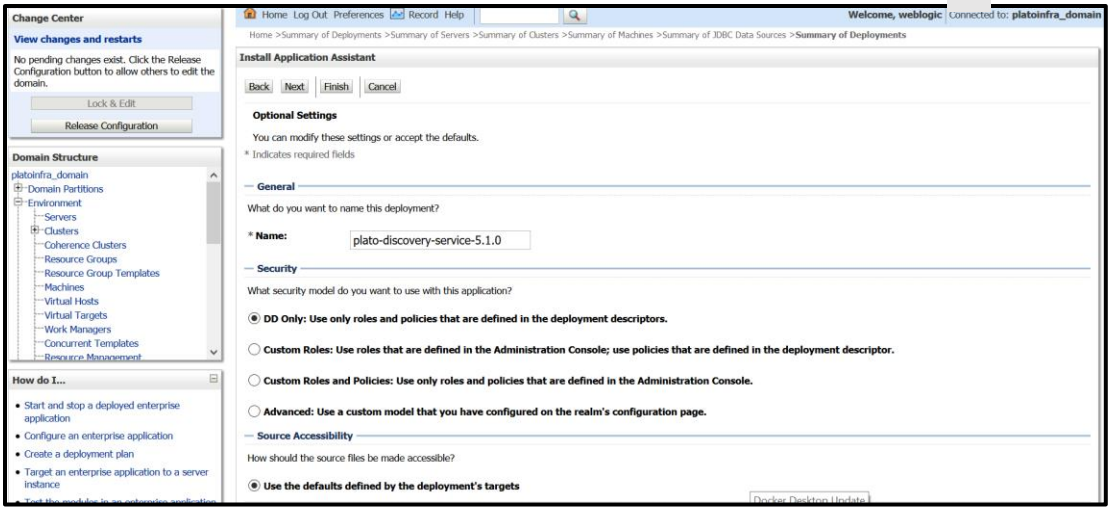


5. Select option **Install this deployment as an application** and click **Next**.

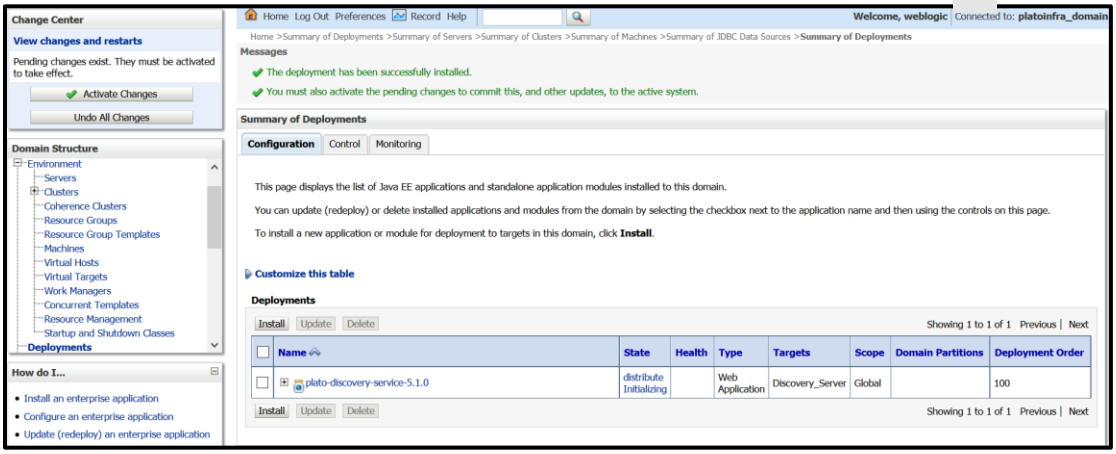


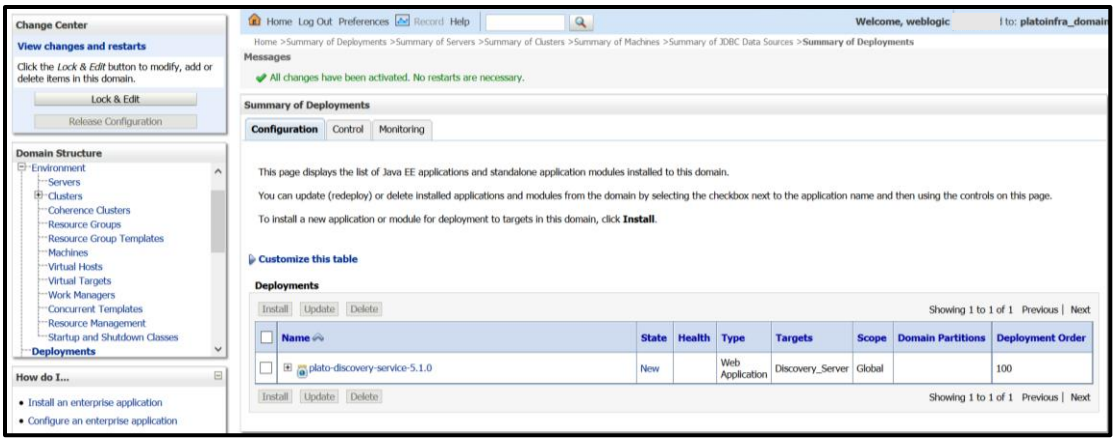
6. Select target servers/clusters on which application has to be deployed and the **Next**.



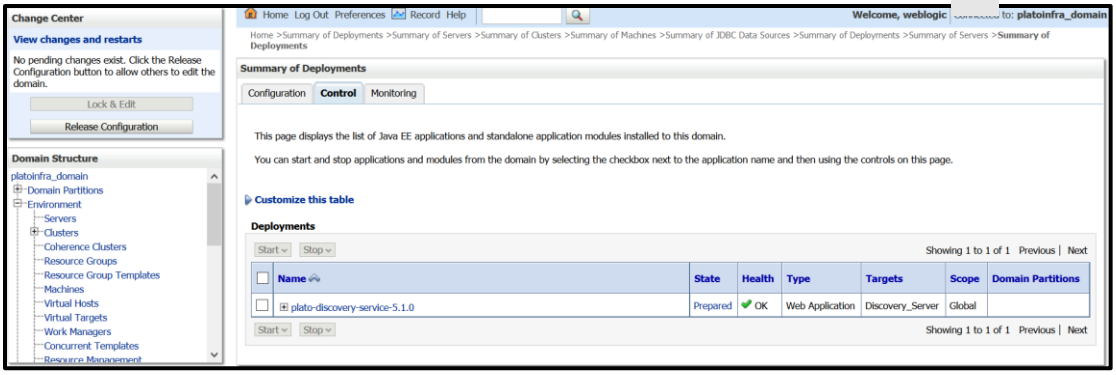


7. Click **Finish** and then click **Save and Activate Changes**.

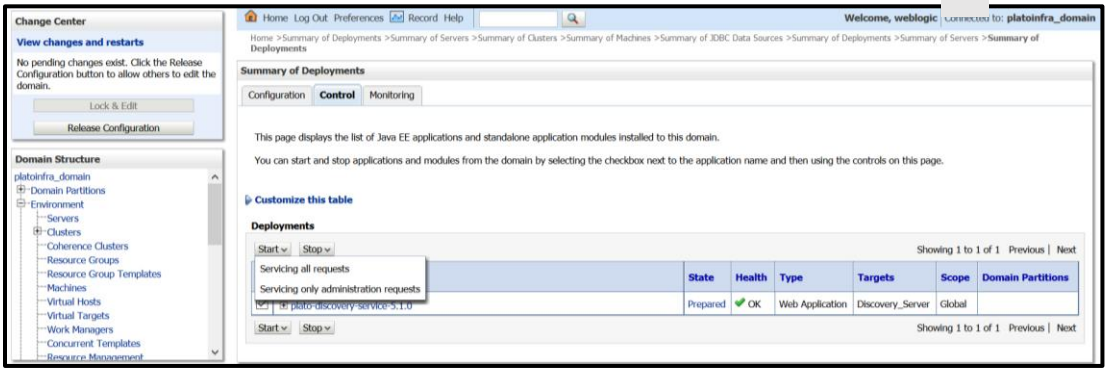




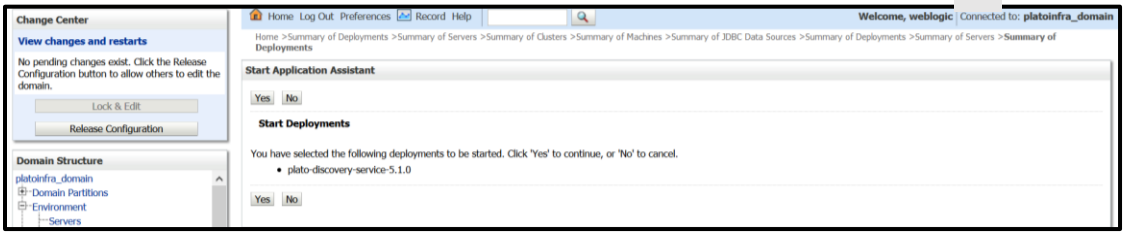
8. Click **Deployments** and then **Control** to changes the state of application from prepared to active status.



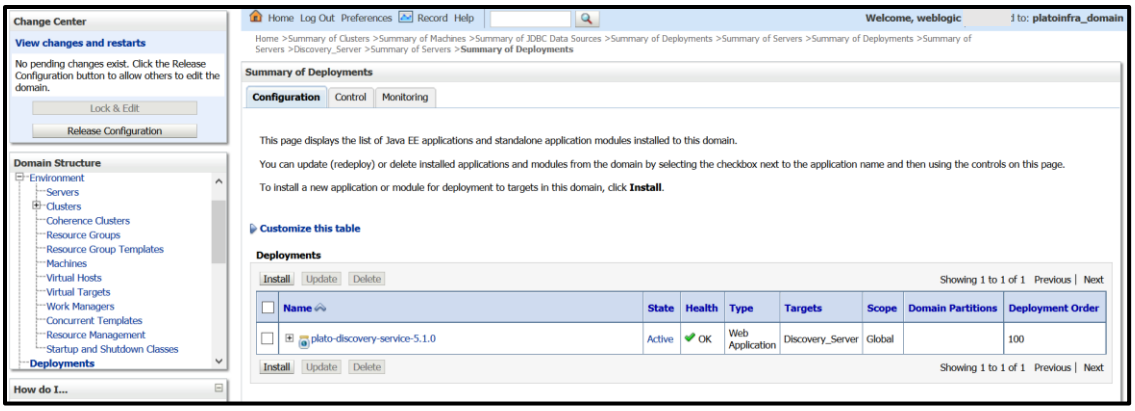
9. Under **Deployment**, click **Start** dropdown and select **Start all requests**.



10. Click **Yes.**



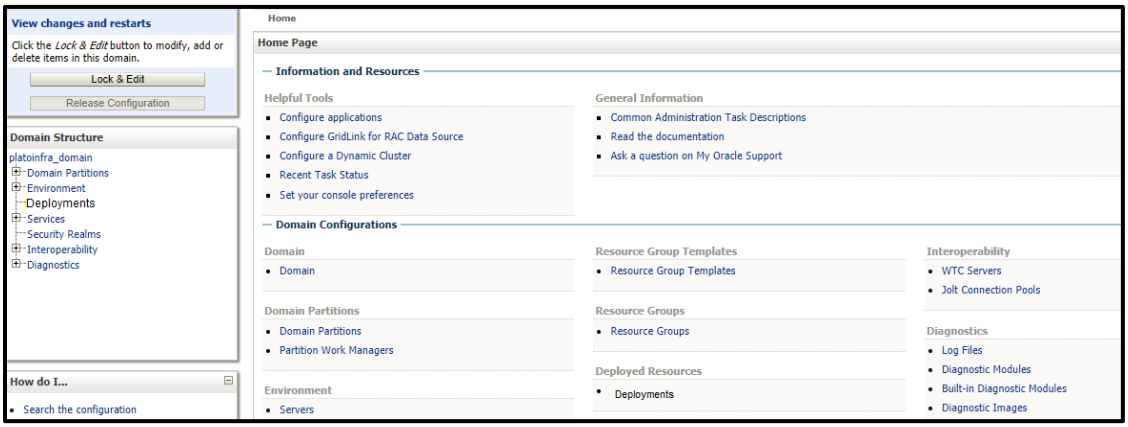
11. The status is displayed as **Active in the state column.**



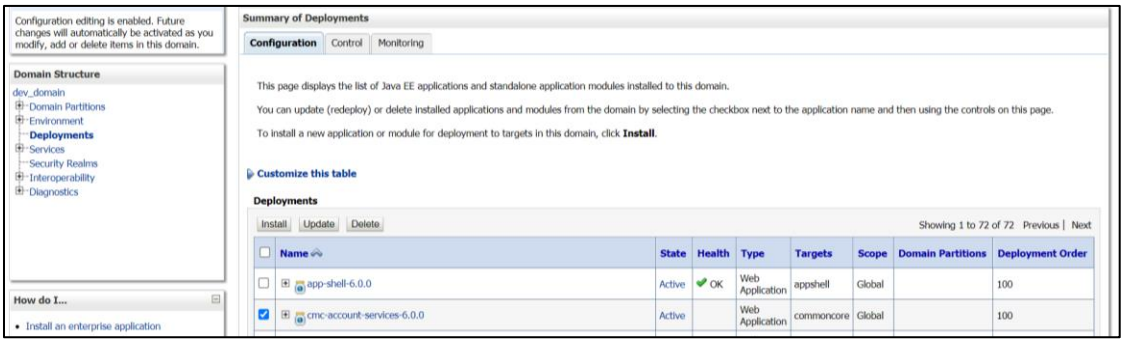
1.6 How to Undeploy Application

Login into weblogic server with the proper credentials.

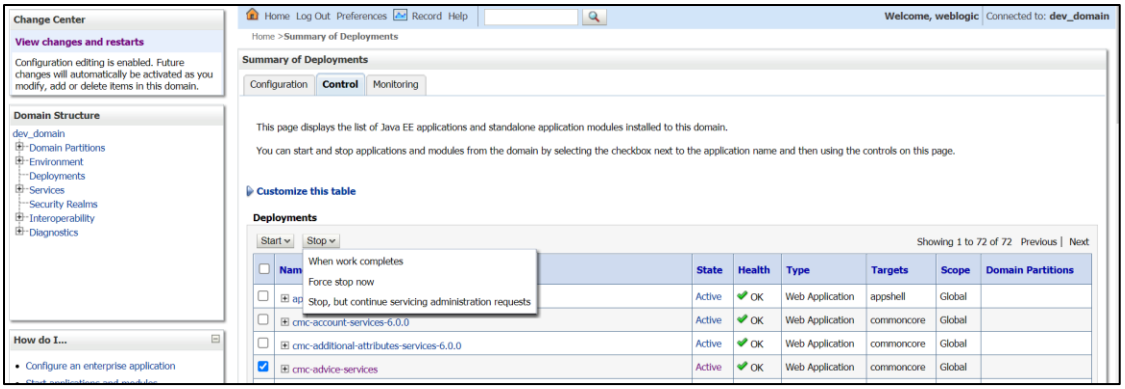
1. Navigate to left menu and select **Deployments**.



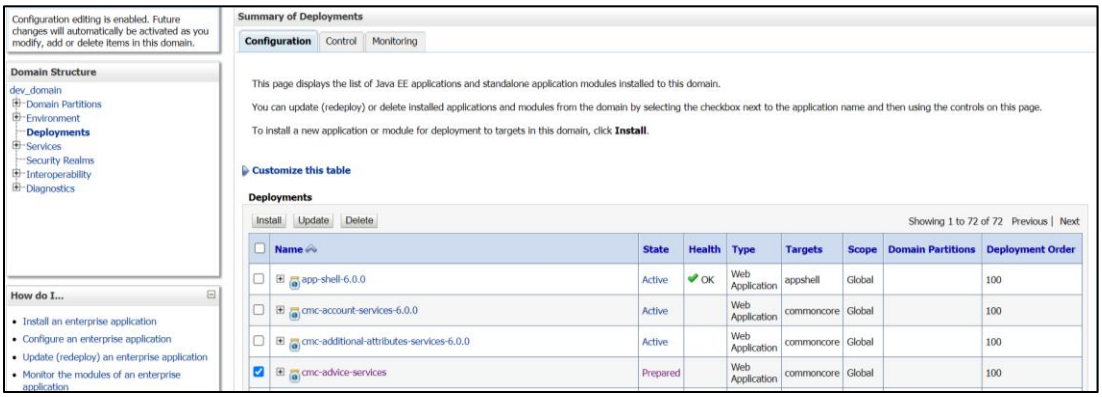
2. Click **Lock and Edit** and then select the service that needs to be undeployed in Deployments.



3. On **Control** tab, click **Stop**, and select **Force stop now** from the dropdown list.



4. Once it changes to prepared state, click **Configuration** tab.

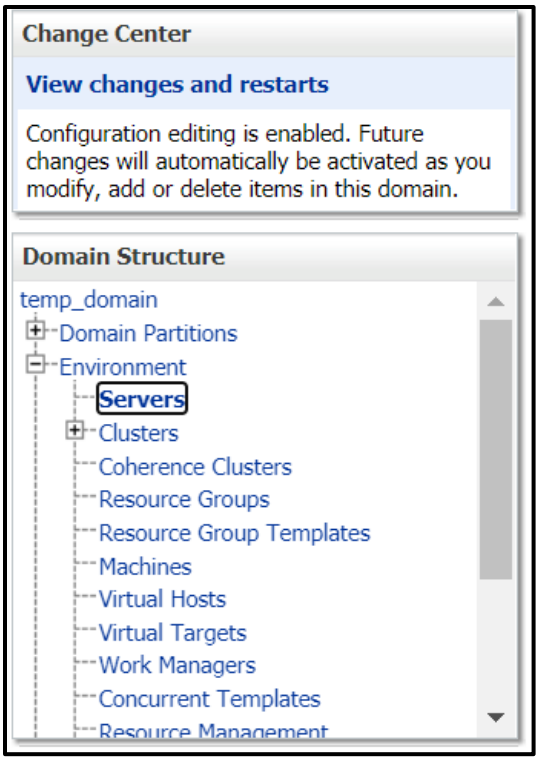


5. Select the service again and click on Delete to undeploy the service.

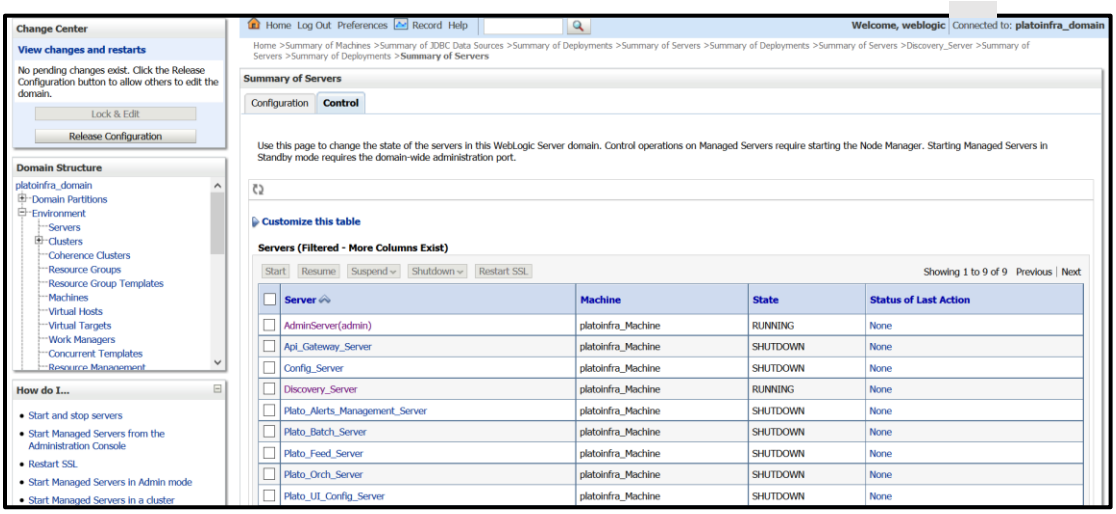
1.7 How to Restart Servers

Perform the following steps to restart servers:

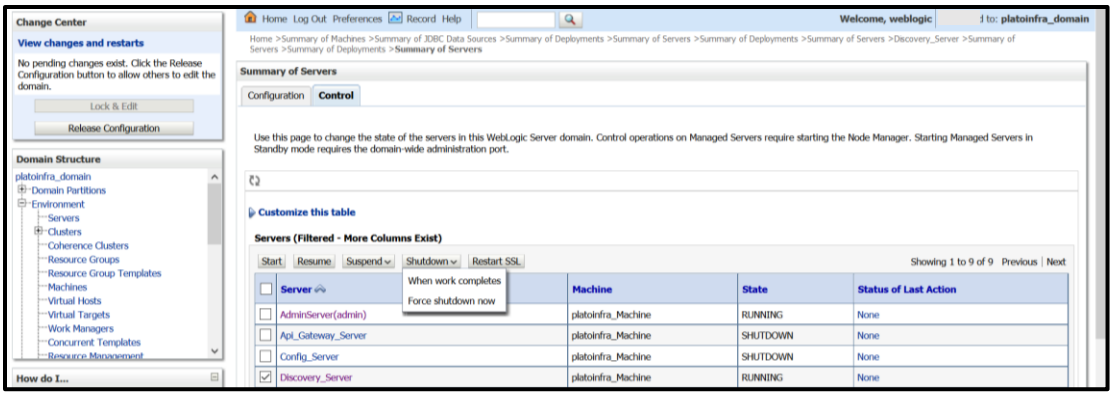
1. Navigate to left menu and select **Environment**, and then click **Servers**.



2. Click **Control** tab.

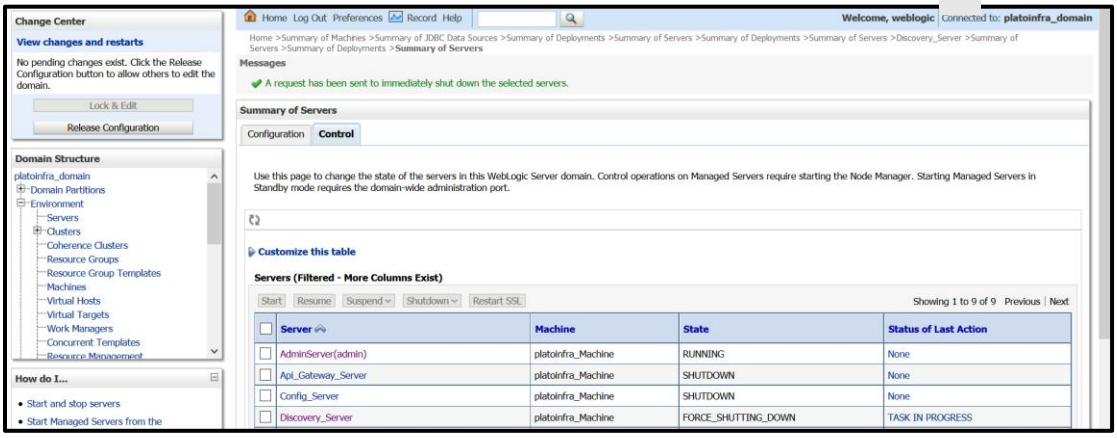


3. Select servers to **Shutdown**

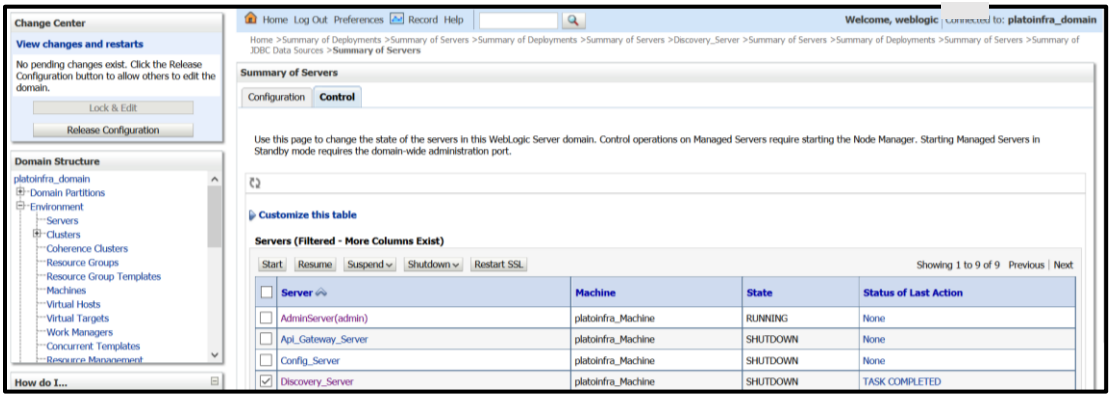


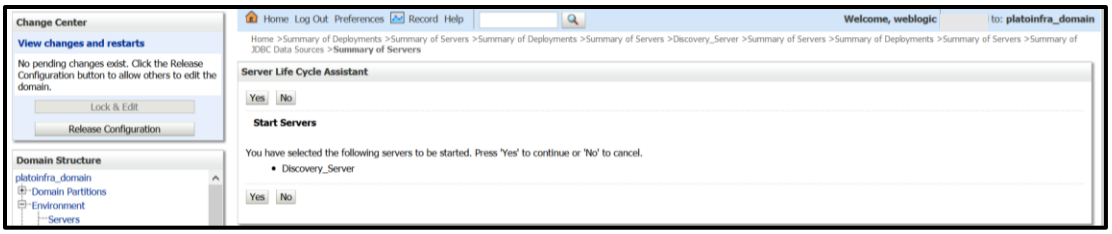
4. Click **Yes** to confirm shutdown.

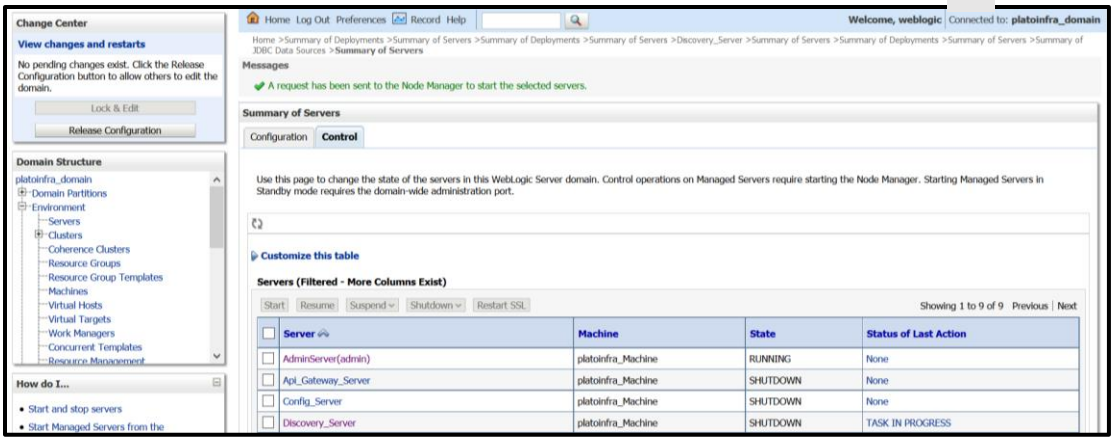
5. The status displayed as shown below:



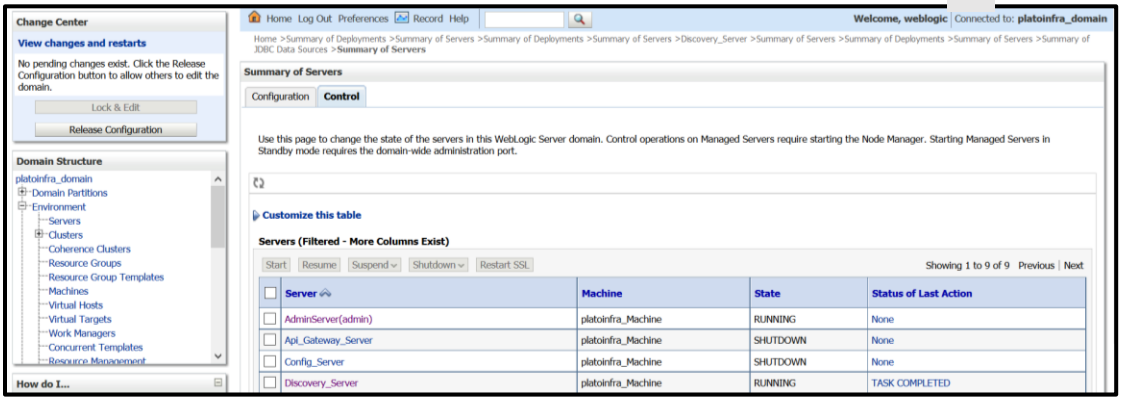
6. Once shutdown is completed, navigate to **Control**, select the servers to **Start**, and click **Yes** to confirm action.

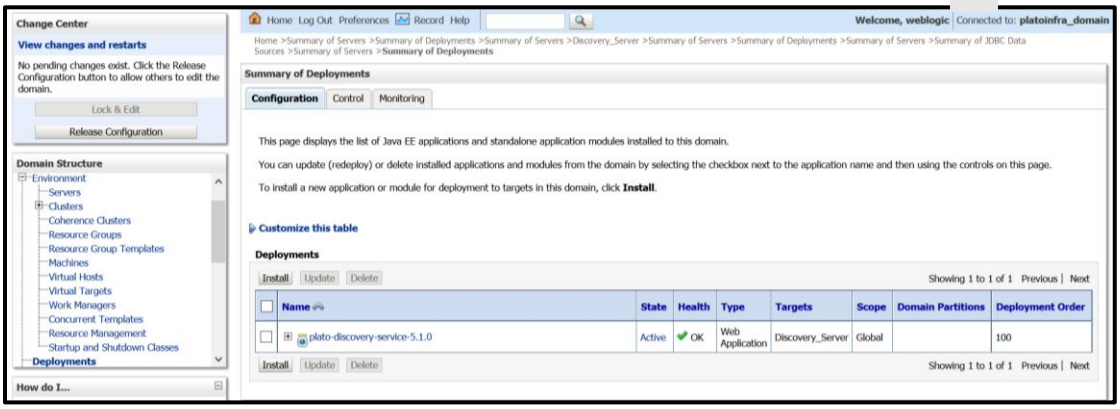






7. When all requested servers are running, navigate to **Summary of Deployments,** and check if deployments are in active state.

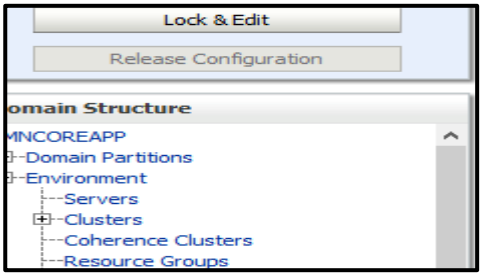




1.8 How to Check Port Number

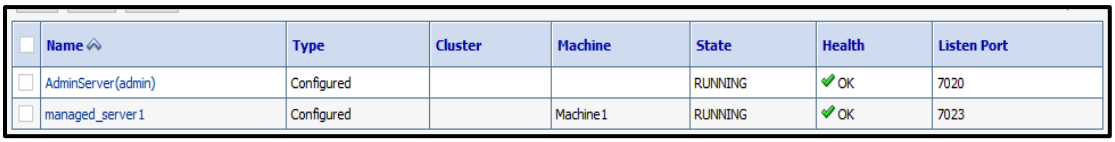
Perform the following steps to check port numbers:

1. Specify the **User id** and **Password**, and login to **WebLogic console**.
2. Click **Environment** and then click **Server**.



1.

3. Under Servers (Filtered - More Columns Exist) section, you will be able to see all the server listed.



Name	Type	Cluster	Machine	State	Health	Listen Port
AdminServer(admin)	Configured			RUNNING	OK	7020
managed_server1	Configured		Machine1	RUNNING	OK	7023

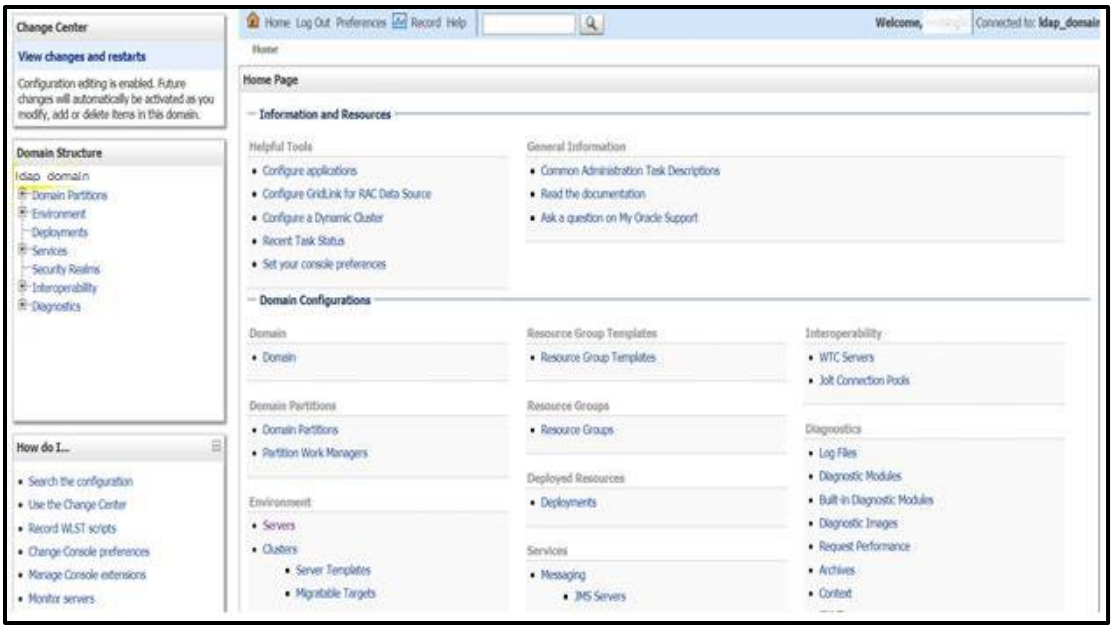
1.9 Weblogic Embedded LDAP Setup

The following changes are to be made for configuring the Weblogic-Embedded LDAP server for Oracle Banking Microservices Architecture:

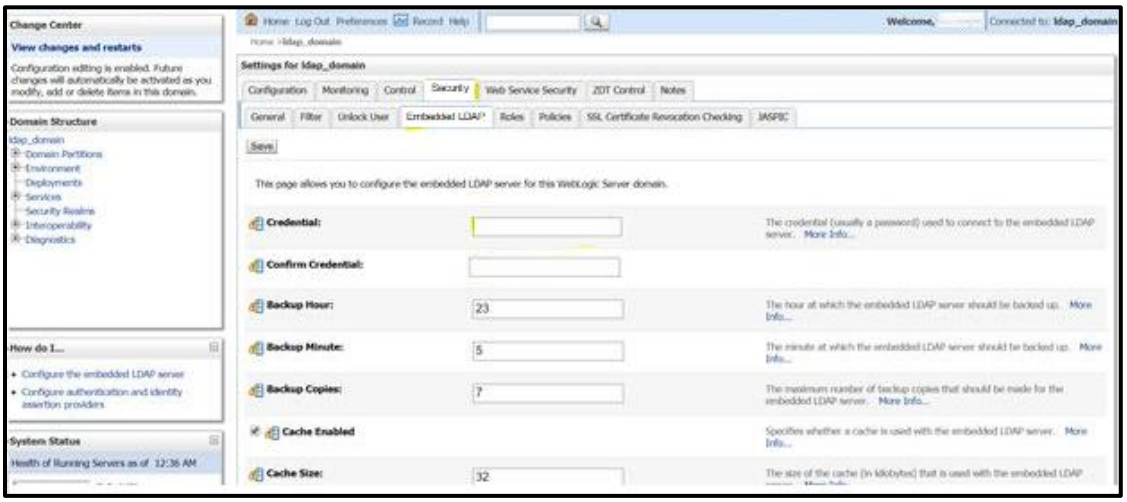
- Configuration of Weblogic LDAP
- Creation of Users
- Plato Security Config Table Entries

1.9.1 Configuration of Weblogic LDAP

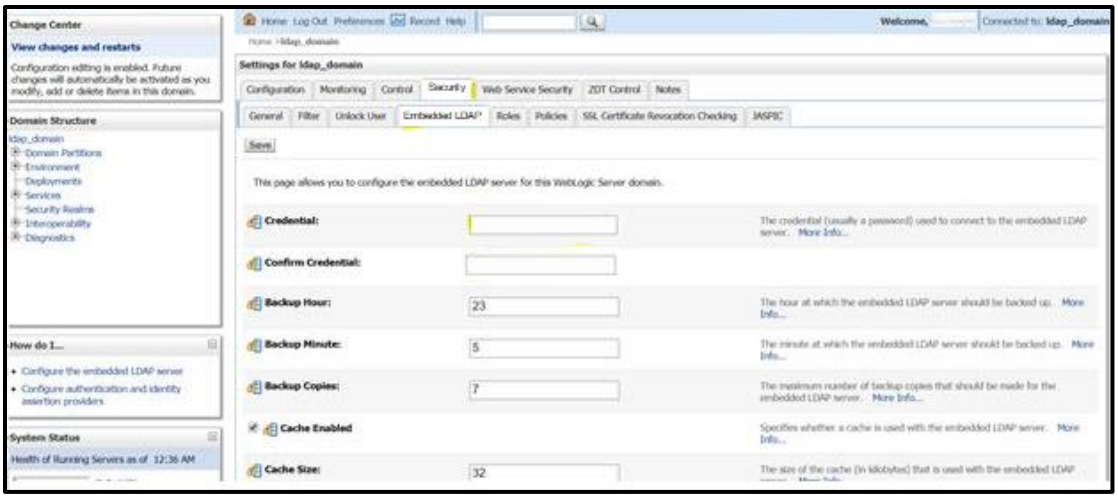
1. Open the **Administration Console** for the **Weblogic**, and click domain name in left panel.



2. Under Settings for ldap_domain, click **Security** tab, and then click **Embedded LDAP** tab.

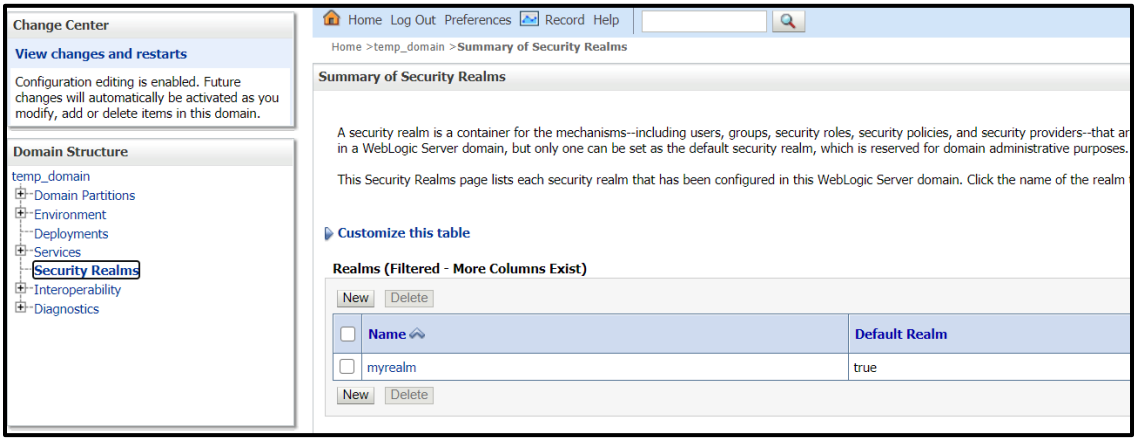


3. Set the **Credential** for Weblogic Embedded LDAP store. This is needed in the **Security Config** table.

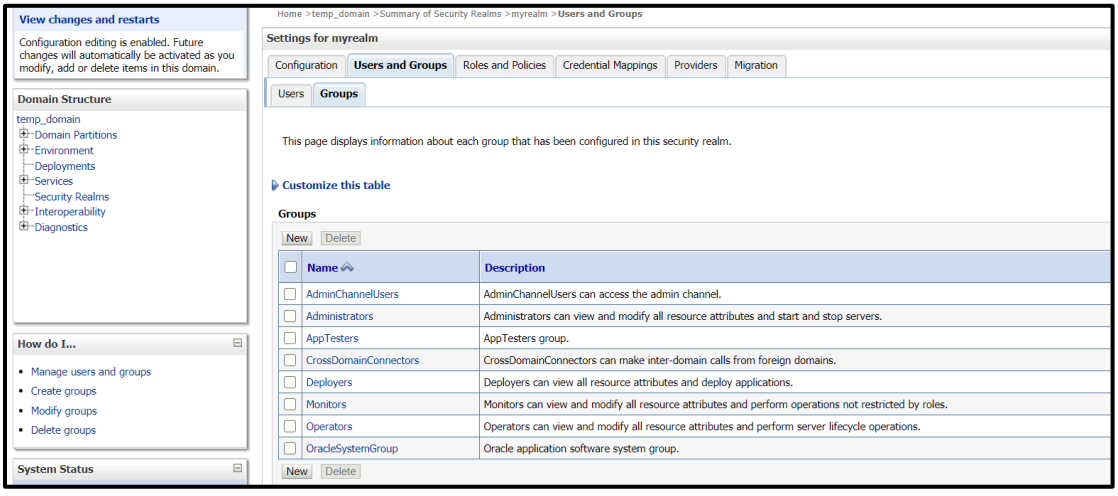


1.9.2 Creation of Users

1. Navigate to left menu, and click **Security Realms**.
2. In the **Summary of Security Realms** window, click **myrealm**.

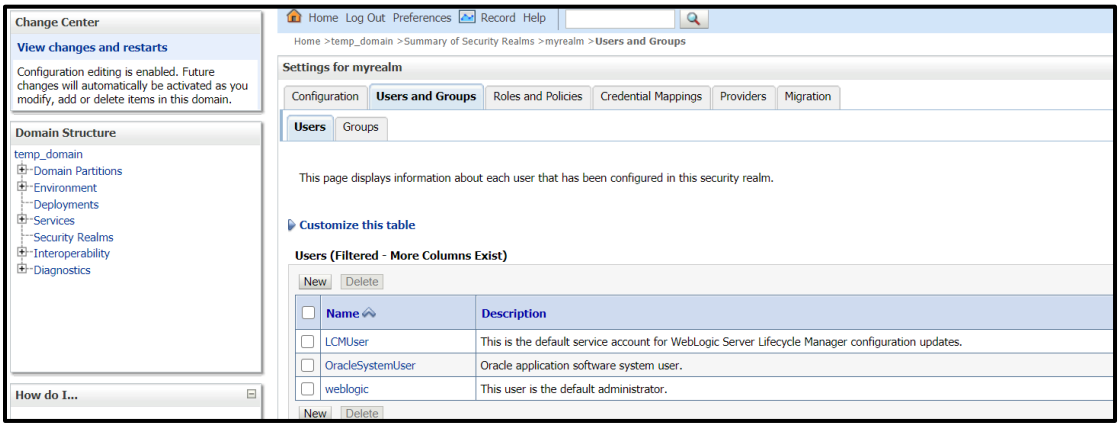


3. Under **Settings for myrealm**, click **Users and Groups**.
4. Click **Groups** tab. Click **New** to make a new group.

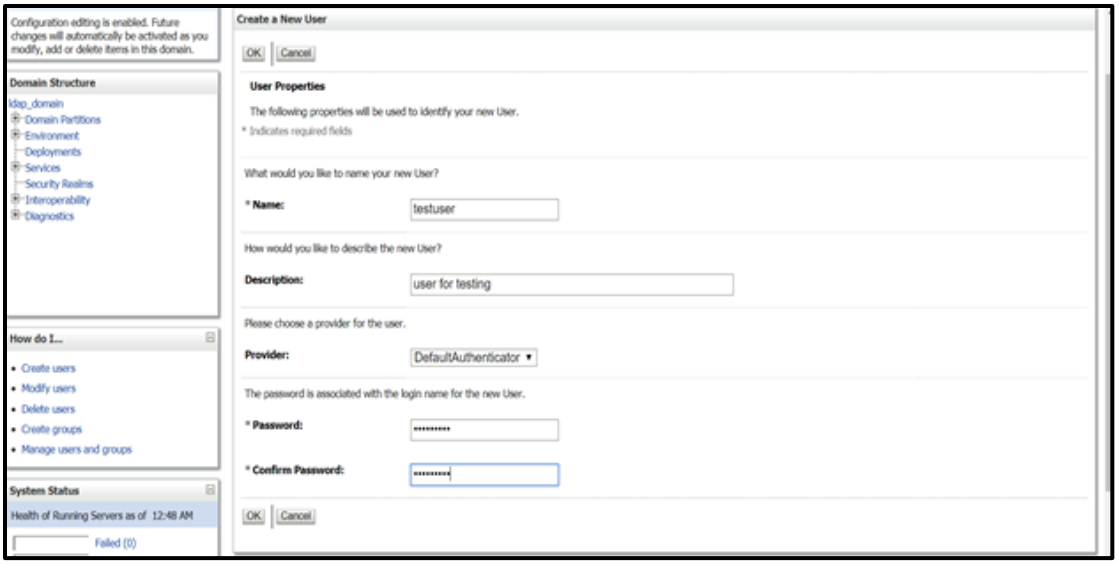


5. Add the relevant details and click **OK**. The new group will be created.

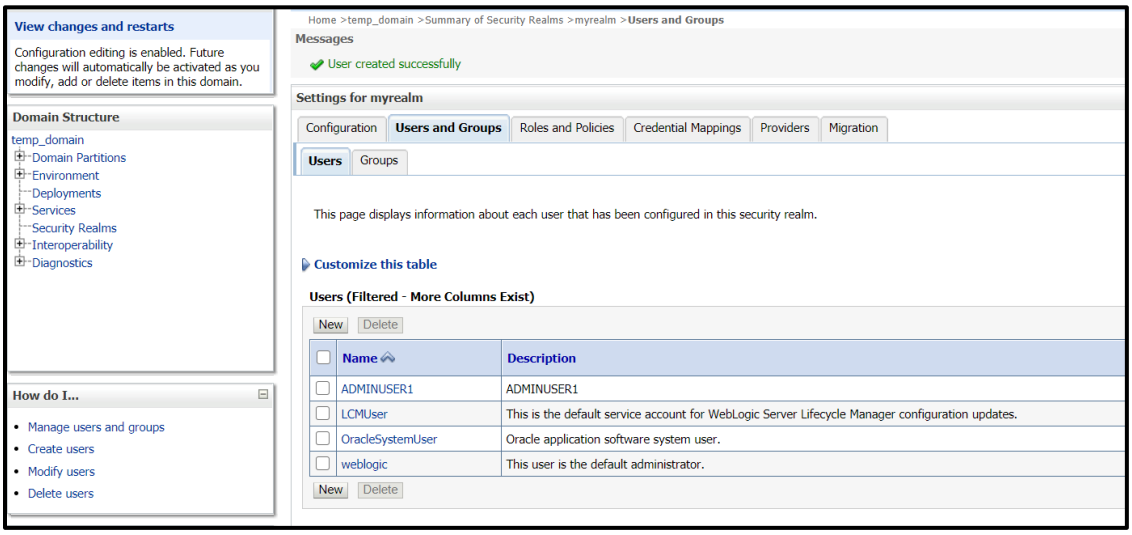
6. Click **Users** tab, and click **New** to create user.



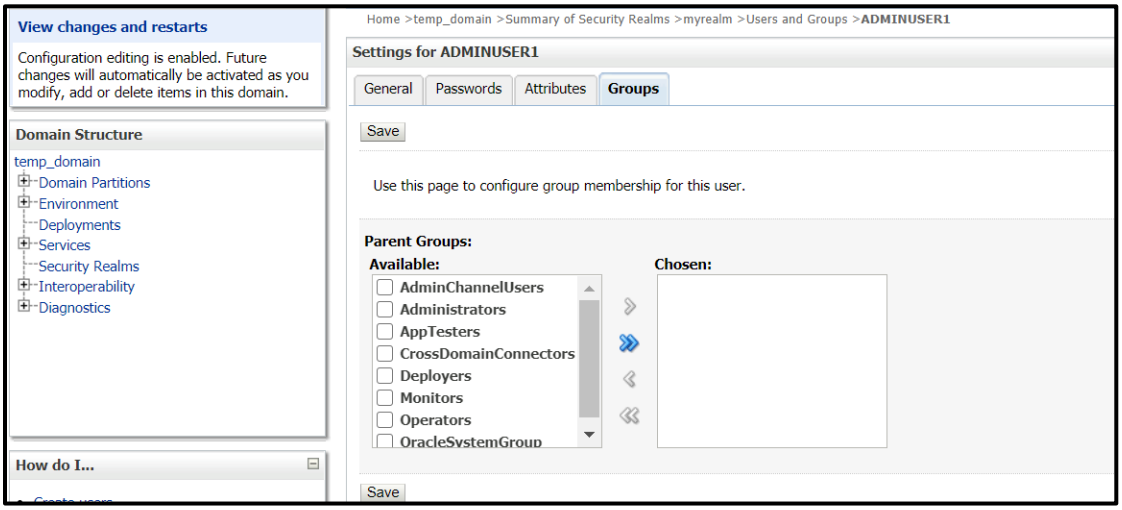
7. Enter the required details for the user. After completing click **OK**. The user will be created.



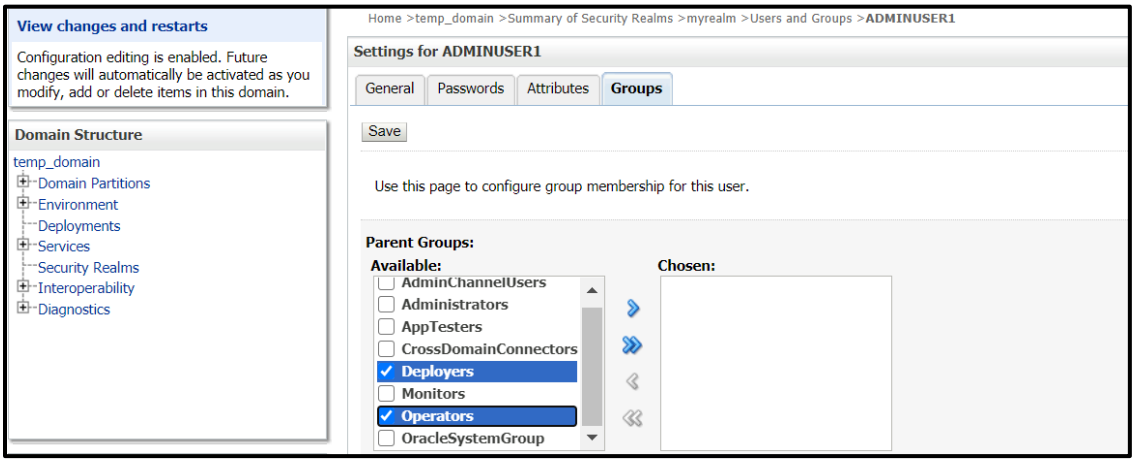
8. Click on the newly created user to assign the newly created user to some group.



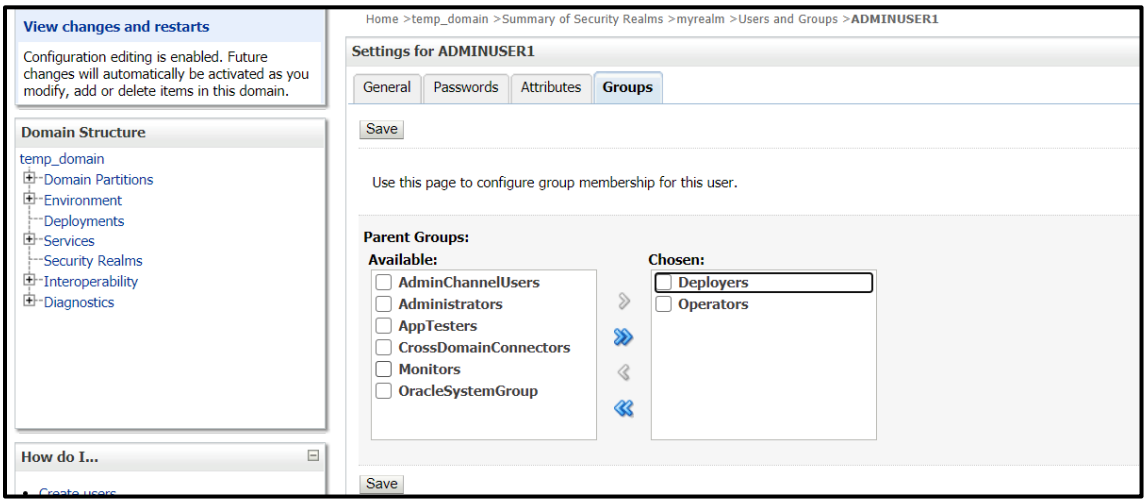
9. Under **Setting for ADMINUSER1** (or whatever your user's name is) window, click **Groups** tab.



10. Select the groups you want to assign to the user and click single right button as shown below.



11. Click **Save**.



1.9.3 Oracle Banking Microservices Architecture Security Config Table Entries

Connection details for the embedded LDAP of weblogic (assuming the admin server is running on 10.99.99.10:7001) are given below:

Connection Details:

URL: ldap:// 10.99.99.10:7001

Server Base: dc={DOMAIN_NAME} (in our case it would be dc=ldap_domain)

User Search Base: ou=people,ou=myrealm

Server User: cn=admin

Server Credentials: As setup in step Point 3 under 1.8.1

Security Config Table Entries:

ID	VALUE	Description
LDAP_URL	ldap:// 10.99.99.10:7001	Valid LDAP Server address with port.
LDAP_SERVER_USER	cn=admin	LDAP server login username
LDAP_SERVER_BASE	dc=ldap_domain	LDAP Server Base
LDAP_SERVER_CREDENTIAL	yIksiMFfjVbfcpA7Qheh8Q==	LDAP server credentials in encrypted form(For Encryption steps, refer to Encrypted Utility section below)
LDAP_USER_SEARCH_BASE	ou=people,ou=myrealm	LDAP User Search Base

ID	VALUE	Description
LDAP_PROVIDER	EMBEDDED_WEBLOGIC	Which LDAP Provider to be used. Also, if this row is not present in this table, then In-House Spring Plato LDAP will be used.

1.10 Oracle Analytic Server Setup

This section contains the following sub-sections:

- Prerequisite
- Start BI Server
- Upload BI Reports
- Test BI Reports

1.10.1 Prerequisite

Perform the following steps:

- Machine should have Java JDK1.8.0_271 has installed
- Oracle Analytics Server 5.5.0

1.10.2 Start BI Server

Perform the following steps to start BI server:

1. Start the weblogic server and analytics server.
2. Check the weblogic console whether analytics server is running.

1.10.3 Upload BI Reports

Perform the following steps to upload BI reports:

1. Login to the Analytics server console.
2. Open the OSDC and check for the report Catalog object **\{unzip folder}\REP\{reportfilename}.xdrz** or any other Catalog objects listed below:

Catalog Object	Extensions Supported
Data Model	.xdmz
Folder	.xdrz
Report	.xdoz
Style Template	.xssz
Subtemplate	.xsbz

3. Upload the catalog object to Analytics Server.

1.10.4 Test BI Reports

Perform the following steps to generate BI reports:

1. Open the application, and go the **Reports** section of the application.
2. Choose the report generation criteria. For example, **Start Date** or **End Date**.
3. Choose the format of the report.
4. Generate the report.

Note: If the format of the report selected is PDF, a PDF report will be generated.

1.11 How to deploy Plato-Apigateway Router

1.11.1 Router deployment steps

Follow services must be deployment in below order to setup router service:

1. **Deploy plato-config-service**
   a. Set placeholder -Dflyway.domain.placeholders.plato-apigateway-router.server.port=<new server port for plato-apigateway-router>
2. **Deploy plato-ui-config-service**
   a. set -Dflyway.domain.placeholders.apigateway.port=<new server port for plato-apigateway-router>
   b. set -Dflyway.domain.placeholders.apigateway.host=<server host for plato-apigateway-router>
3. **Deploy plato-api-gateway**
   a. Migrate existing OAuth users:

      API for migration - /api-gateway/migrateOauthUsers

      Example: http://hostname:8080/api-gateway/migrateOauthUsers

      Authorization - jwtToken

      Headers:

      appId,userId,entityId

      Body (Json): ["client1", "client2"] - Migrate selected list of clients

      or

      Body (Json): ["ALL"] -Migrate all clients.
4. Deploy **plato-apigateway-router**

java -jar plato-apigateway-router.jar --
plato.services.config.uri=http://hostname:8001 --
plato.service.logging.path=/logfilePath

--plato.services.config.uri - Config server URI which is referred by all other services.

--plato.service.logging.path - Path where log file(plato-apigateway-router.log) must be created. Specify the same path as that of other services.

We can enable SSL for plato-apigateway-router by providing:

--server.ssl.enabled=true

--server.ssl.key-store=C:/Users/KEYS/keytool/keystore.jks

--key-store-password=xxxx

--server.ssl.trust-store=C:/Users/KEYS/keytool/truststore.jks

--trust-store-password=xxxxx

--salt=xxxxx

Note: Passwords and salt must be encrypted value generated using respective toolkits.

Provide ssl certs of plato-api-gateway required for validation call when plato-api-gateway is deployed in different server.:

--apigateway.useServerSSLKeys=false

--apigateway.ssl.key-store=C:/Users/KEYS/keytool/keystore.jks

--apigateway.ssl.key-store-password=xxxx

--apigateway.ssl.trust-store=C:/Users/KEYS/keytool/truststore.jks

--apigateway.ssl.trust-store-password=xxxxx

Note: Above certificates can be different than that of plato-apigateway-route

we must also provide trust certificates as

--spring.cloud.gateway.httpclient.ssl.trusted-x509-certificates=C:/Users/KEYS/keytool/keystore1.pem, C:/Users/KEYS/keytool/keystore2.pem

Note: Run this service with nohup command to that process will run on background

App-shell must point to plato-apigateway-router service. Update 'apigateway.url' by correcting it to "http://hostname:8080" - here 8080 is the port is configured for plato-apigateway-router.

1.11.2 Generation pem file and encryption of secrets:

Use plato-security-toolkit to encrypt secrets ---key-store-password, --trust-store-password, --apigateway.ssl.key-store-password, --apigateway.ssl.trust-store-password and these encrypted values must be passed to router service.

Encryption of secrets:

To encrypt the passwords as per Oracle Standards, we recommend toolkit - plato-security-toolkit

Usage: java -jar plato-security-toolkit-9.1.0.jar

Enter pass phrase: Test123

Enter Salt: 0.9412345671234567

Encrypted Password: m4Q1rbtegkWse2s7D2jKfw==

Encryption of salt: to encrpt –salt value used while generating encrypted secret. This encrypt salt must be passed to router service.

To encrypt the salt as per Oracle Standards, we recommend toolkit - plato-security-salt-encryption-toolkit

Usage: java -jar plato-security-salt-encryption-toolkit-9.1.0.jar

Enter Salt: 0.9412345671234567

Encrypted Password:
VmtjMWQxTnJOVlpPV0VaWFZrVndUMWxYTVU1bFJsSlpZMFZLYTFaVVZrWldWbWgzVkRGS1JsWnFVVDA9

PEM file from keystore

keytool -exportcert -alias localhost -keystore keystore.jks -rfc -file keystore.pem

1.11.3 Timeout parameters

These parameters are similar to earlier ribbon timeout params

spring.cloud.gateway.httpclient.connect-timeout= 3000 //seconds

spring.cloud.gateway.httpclient.response-timeout= 360s

spring.cloud.gateway.httpclient.pool.acquire-timeout=6000 //milliseconds

spring.cloud.gateway.httpclient.pool.max-connections=10000

#Properties used webclient call is made to plato-api-gateway for validation

webclient.http.max.connections=1000

webclient.http.acquire.timeout.millisec=5000

webclient.http.connection.timeout.millisec=20000

webclient.http.read.timeout.seconds=20000

webclient.http.write.timeout.seconds=20000