Oracle® Banking Microservices Architecture

Configuration and Deployment Guide

Release 14.7.2.0.0
F88492-01
November 2023

Oracle Banking Microservices Architecture Configuration and Deployment Guide, Release 14.7.2.0.0

F88492-01

Contents

Preface

Purpose

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

Audience

This guide is intended for WebLogic admin or ops-web team who are responsible for installing OFSS Banking Products.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

For more information on any related features, refer to the following documents

- Product Installation Guide

Acronyms and Abbreviations

The list of the acronyms and abbreviations used in this guide are as follows:

Table 1 Acronyms and Abbreviations

Abbreviation	Description
LDAP	Lightweight Directory Access Protocol

1

Placeholder Update for Oracle Banking Microservices Architecture Services

This topic provides the information about the various methods to perform the placeholder update for Oracle Banking Microservices Architecture services.

- Method 1 – Via setUserOverrides.sh file
  This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services through setUserOverrides.sh file.
- Method 2 – Via Passing the -D params in the Server Start Argument
  This topic provides the systematic instructions to run the Oracle Banking Microservices Architecture services by passing the -D params in the Server start argument.
- Method 3 – Using env files and setUserOverrides.sh file
  This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services by using env files and setUserOverrides.sh file.
- Method 4 – Workflow Configuration
  This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services through workflow configuration.

1.1 Method 1 – Via setUserOverrides.sh file

This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services through setUserOverrides.sh file.

1. Create a file named **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 to run Oracle Banking Microservices Architecture Services properly.

> **Note:**
>
> Below is the list of **-D params** (ENV variables) that must be set for all individual services. Set a single **–Dparam** as follows:
> JAVA_OPTIONS="${JAVA_OPTIONS} -DParam =<ParamValue>” 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 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>

// Common Core Interest Rate Service

-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>

1.2 Method 2 – Via Passing the -D params in the Server Start Argument

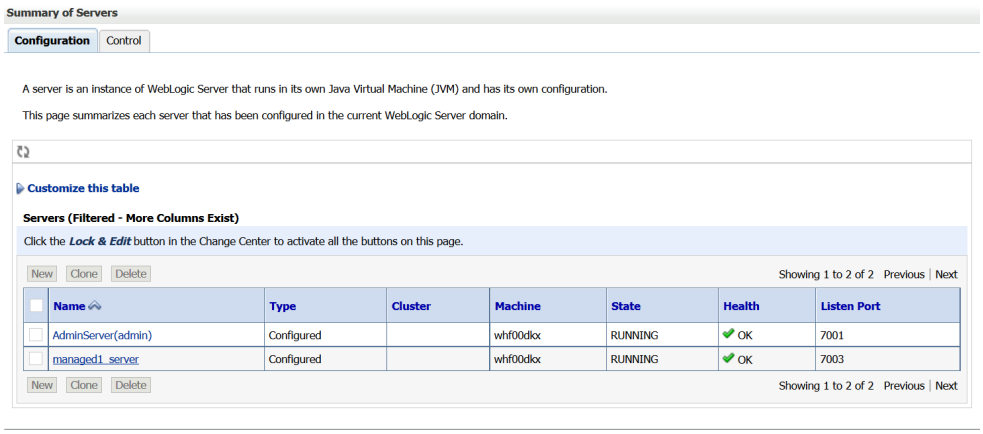
This topic provides the systematic instructions to run the Oracle Banking Microservices Architecture services by passing the -D params in the Server start argument.

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

1. On **Domain Structure**, click **Environment**. Under **Environment**, click **Servers**.

   The **Summary of Servers** screen displays.

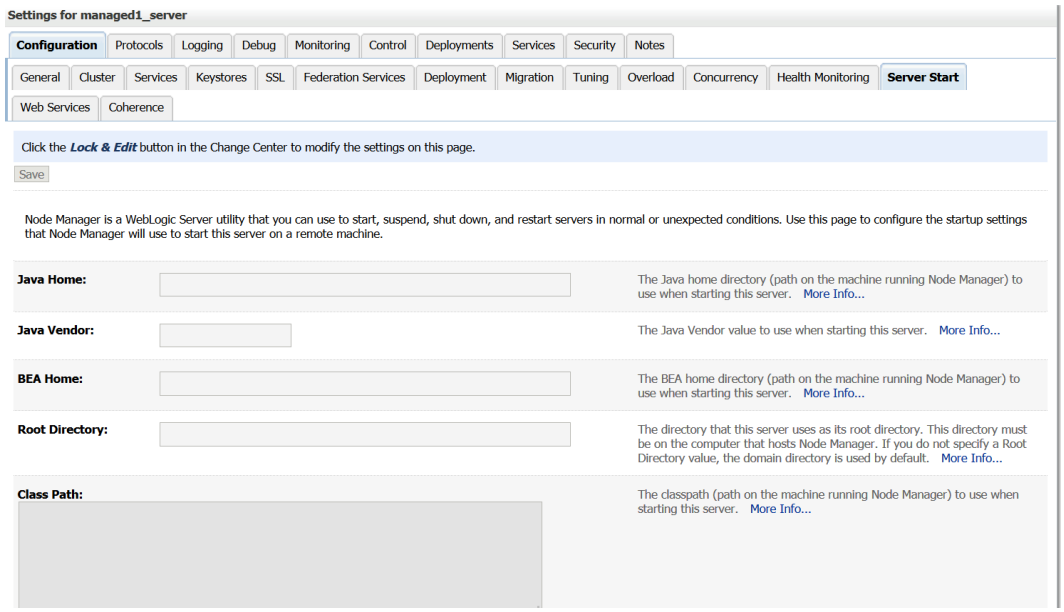
Figure 1-1 Summary of Servers



2. On the **Servers (Filtered - More Columns Exist)** table, click managed server to pass the values.

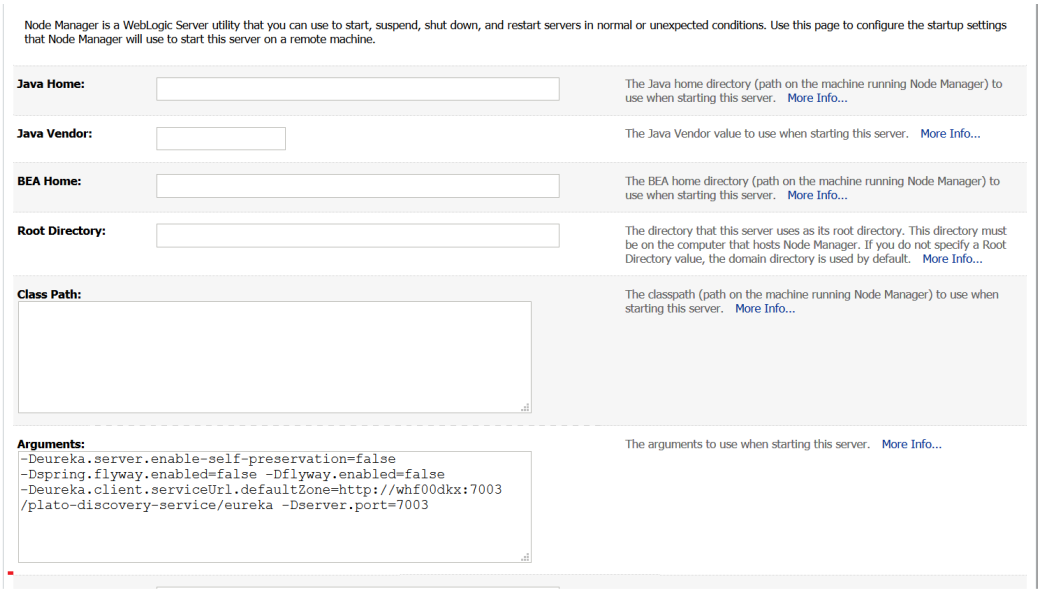
   The **Settings for managed server** tab displays.

Figure 1-2 Settings for managed server



3. On **Settings for managed server** tab, select **Server Start** tab.
4. Edit the **Arguments** field and pass all the environment parameters required for the service to run.

Figure 1-3 Arguments



5. Save the configuration and restart the managed server.

   After restart, the service can be started or deployed properly.

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

This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services by using env files and setUserOverrides.sh file.

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}"
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.

   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.

   Sample file for **plato-config-deploy.env** is given below:

```
### 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.OracleDrive
r
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=
```

Sample file for **domain-config-deploy.env** is given below:

```
### 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=
```

1.4 Method 4 – Workflow Configuration

This topic provides the systematic instructions to run Oracle Banking Microservices Architecture services through workflow configuration.

1. Create Metadata of the workflow.

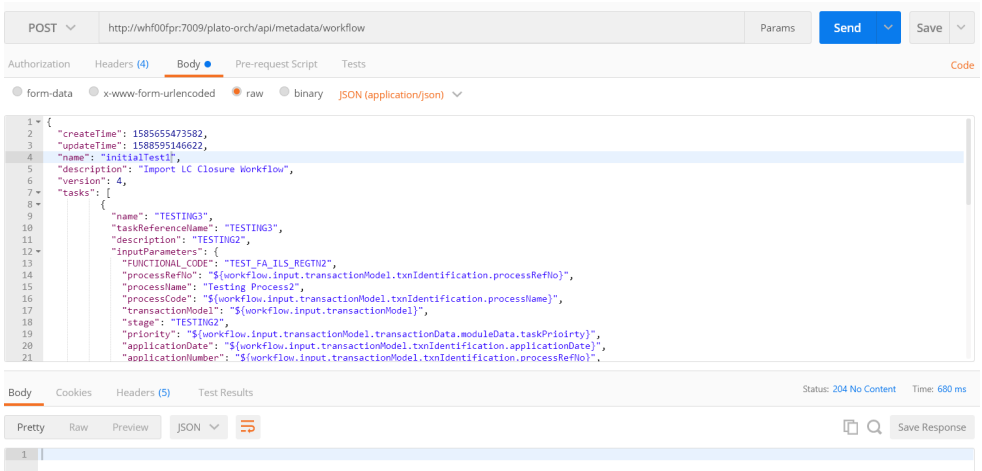
   The sample DSL for workflow creation is given below:

```
{
     "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 below screen depicts the sample workflow

Figure 1-4 Body



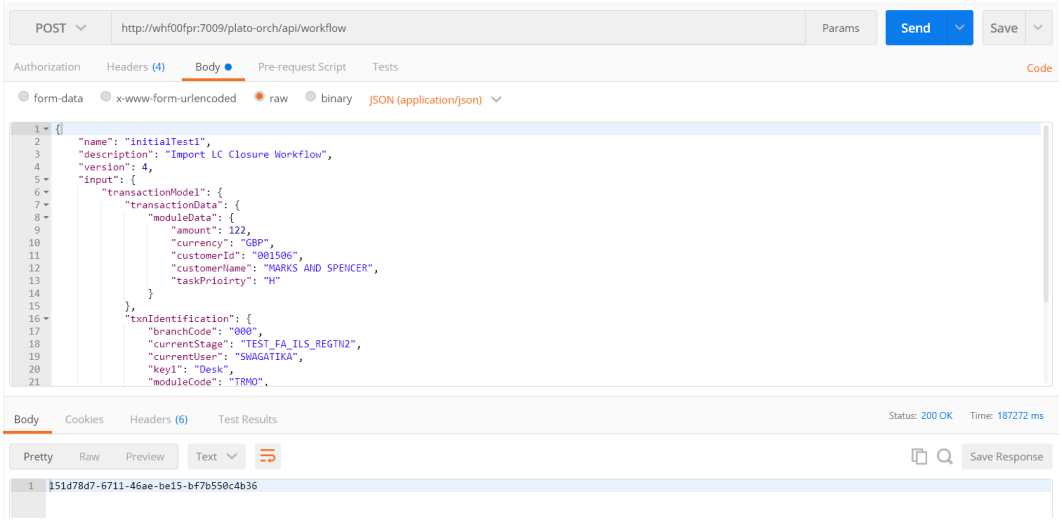
2. For workflow creation, call the API (/api/workflow) to create the workflow. This API provides the information to the workflow metadata which is created using previous call.

```
{
        "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"
      }
    }
  }
}
```

Pass the DSL in body. The below screen depicts the sample workflow:

Figure 1-5 Body



2

Domain Creation and Cluster Configuration

This topic provides the information about the domain creation instructions, cluster configuration, and post domain creation configurations.

- Create Domain and Cluster Configuration
  This topic provides the systematic instructions to create domain and cluster configuration.
- Post Domain Creation Configurations
  This topic provides the systematic instructions for the configurations to be performed once the domain is created.

2.1 Create Domain and Cluster Configuration

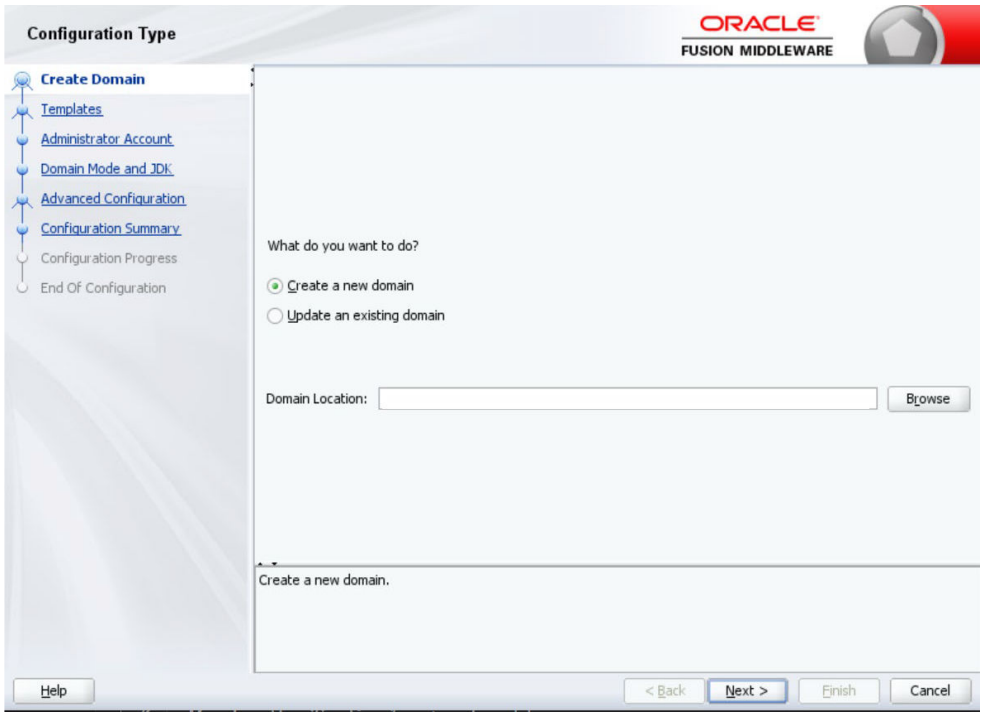
This topic provides the systematic instructions to create domain and cluster configuration.

> **Note:**
>
> Names need not to be same as provided in the screenshot.

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

   The **Create Domain** screen displays.

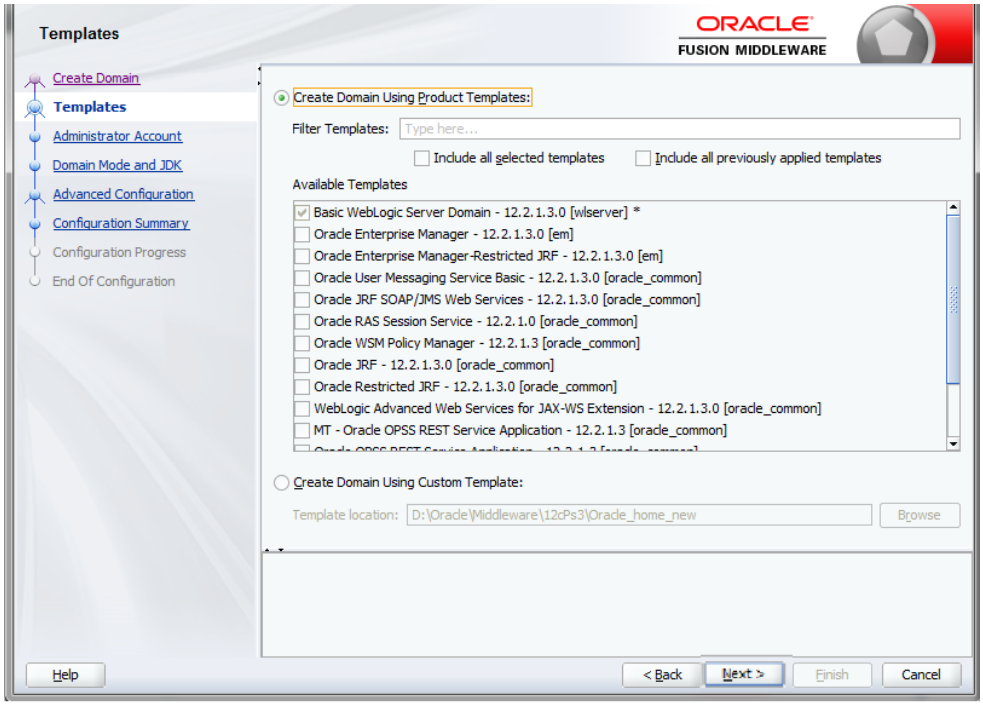
Figure 2-1 Create Domain



4. Click **Next**.

   The **Templates** screen displays.

Figure 2-2 Templates



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

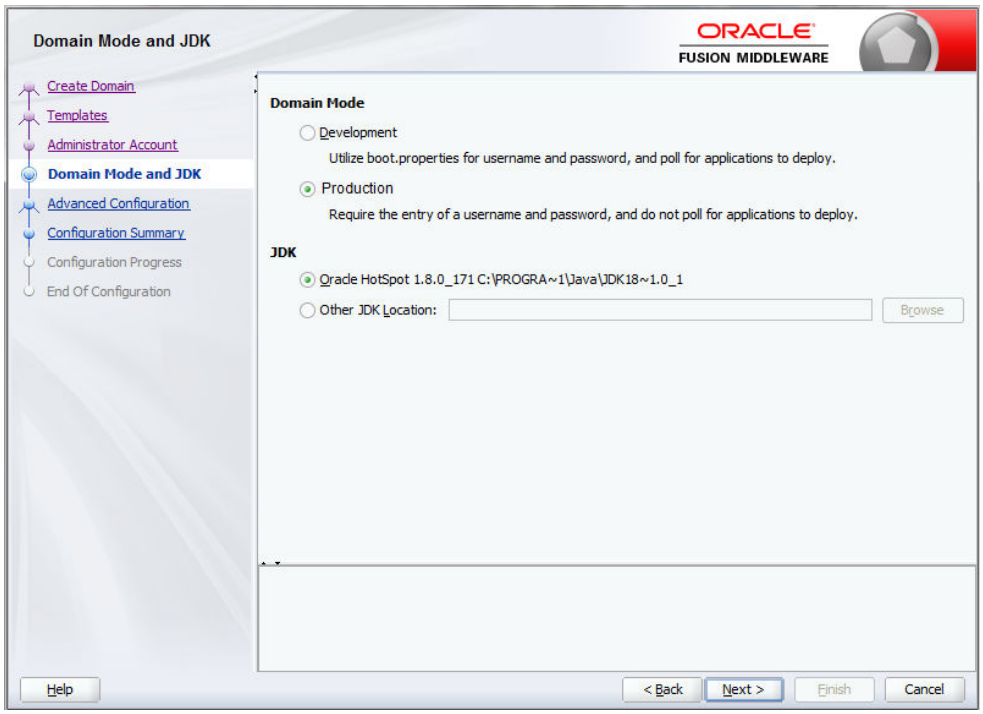
The **Administrator Account** screen displays.

Figure 2-3 Administrator Account

6. Fill the fields **Password** and **Confirm Password** to set the password, and click **Next** to proceed.

   The **Domain Mode and JDK** screen displays.

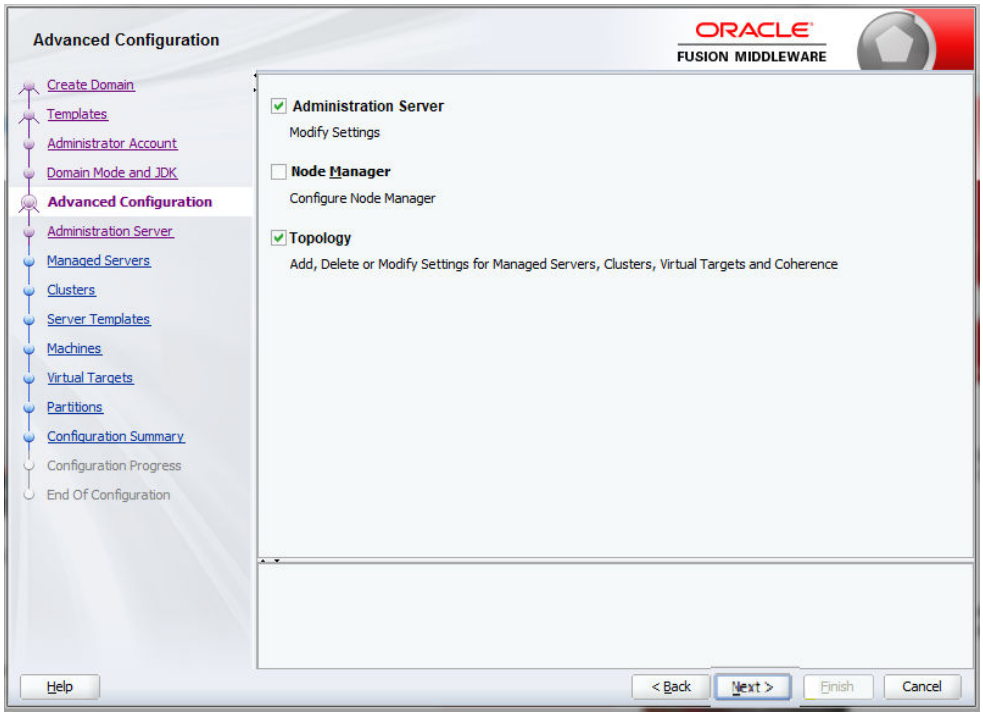
Figure 2-4 Domain Mode and JDK



7. Select **Domain Mode** as **Production**.
8. Select **JDK**, and click **Next**.

   The **Advanced Configuration** screen displays.

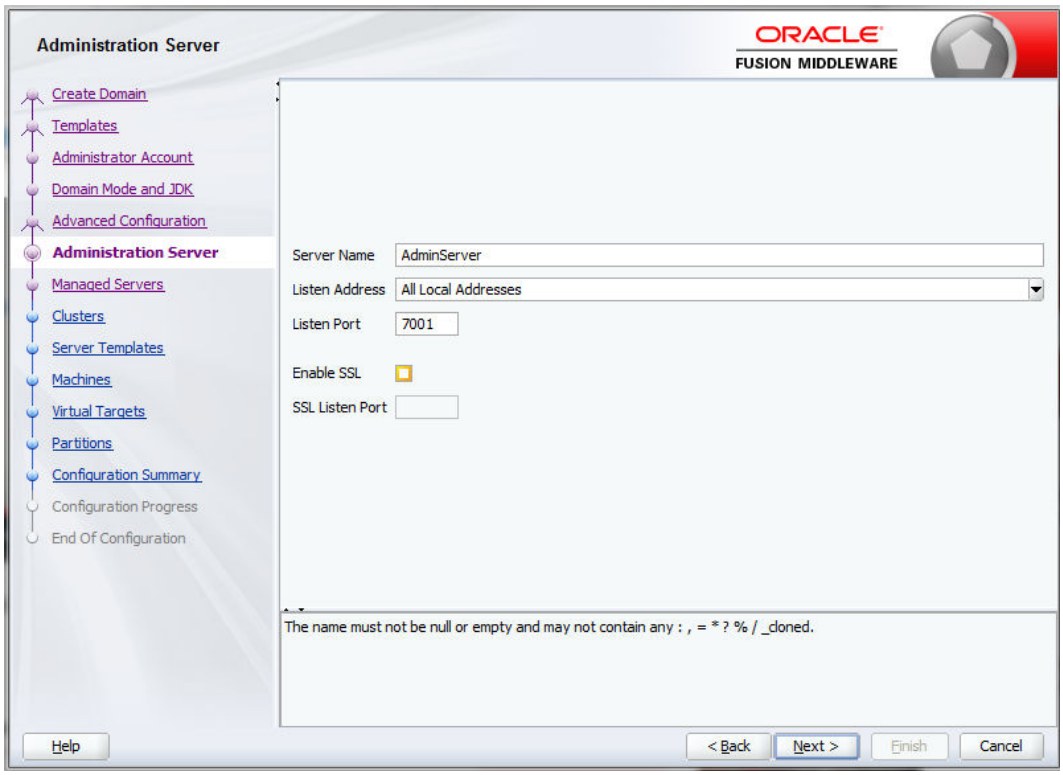
Figure 2-5 Advanced Configuration



9. Select **Administration Server** and **Topology**, and click **Next**.

   The **Administration Server** screen displays.

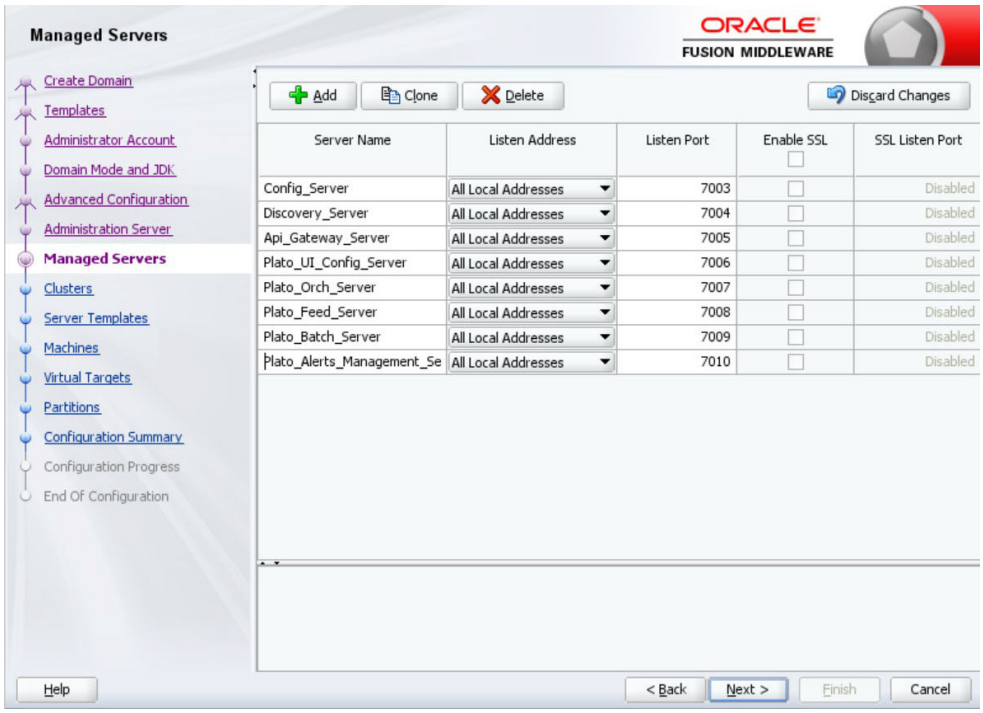
Figure 2-6 Administration Server



10. Edit the fields **Listen Port** and host configurations as required, and click **Next**.

    The **Managed Servers** screen displays.

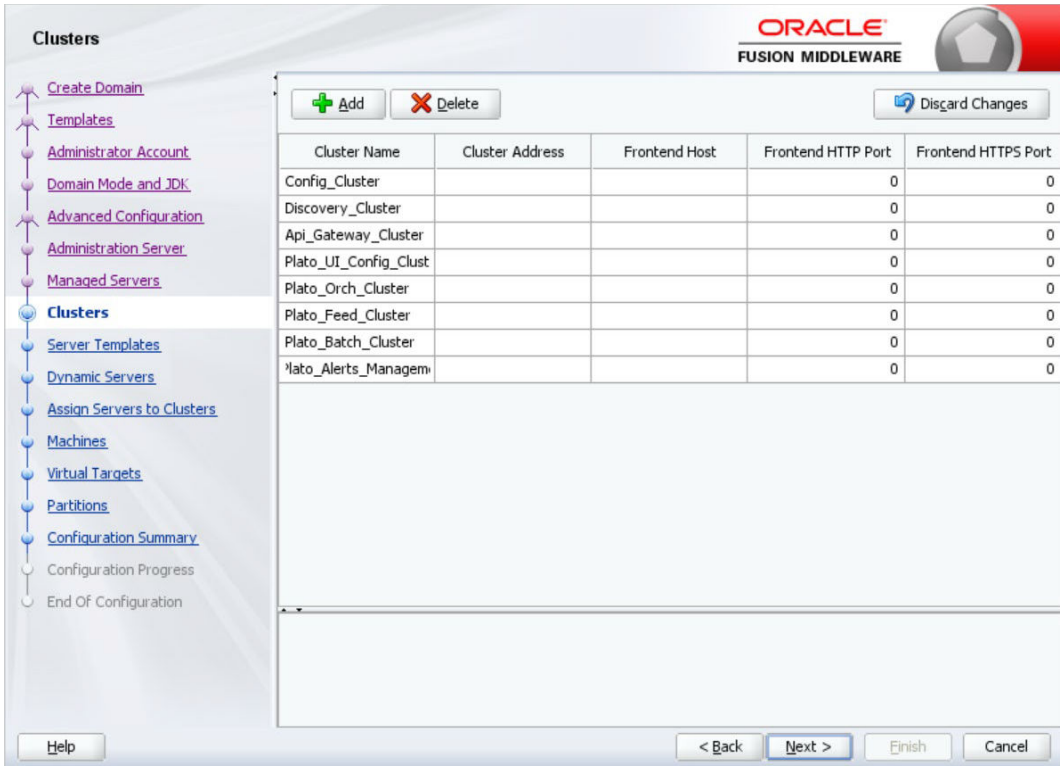
Figure 2-7 Managed Servers



11. Click **Add** to add managed servers and edit the fields as required.
    a. Specify the name in **Server Name** filed.
    b. Edit the address in **Listen Address** filed.
    c. Edit the port in **Listen Port** filed.
12. Click **Next**.

    The **Cluster** screen displays.

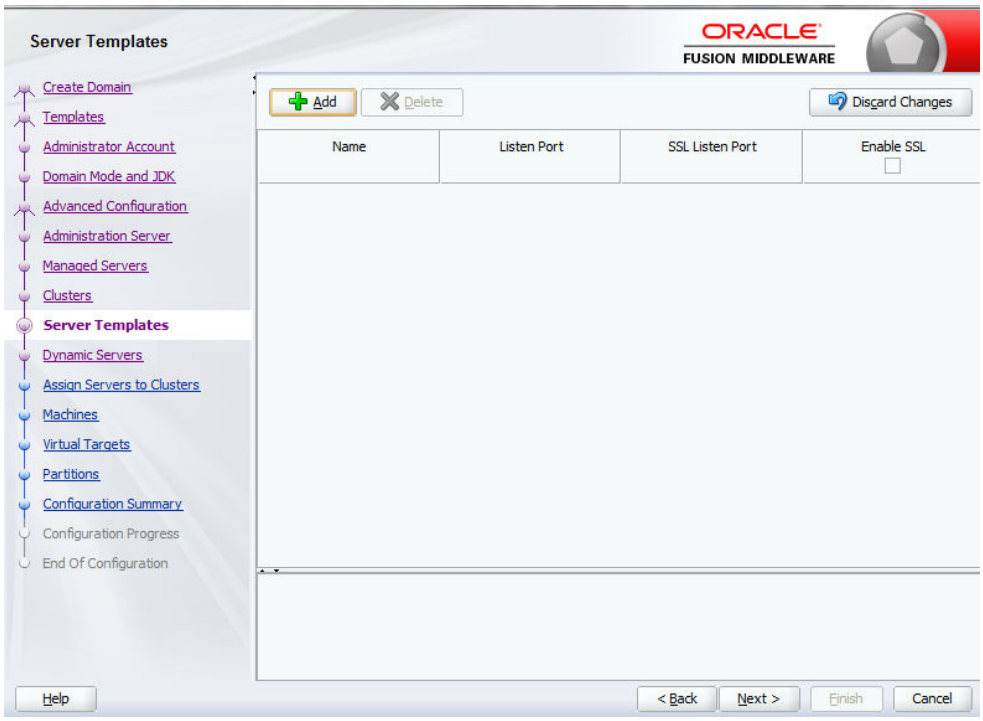
Figure 2-8 Cluster



13. Click **Add** to add clusters one for each **Managed Servers**, and click **Next**.

    The **Server Template** screen displays.

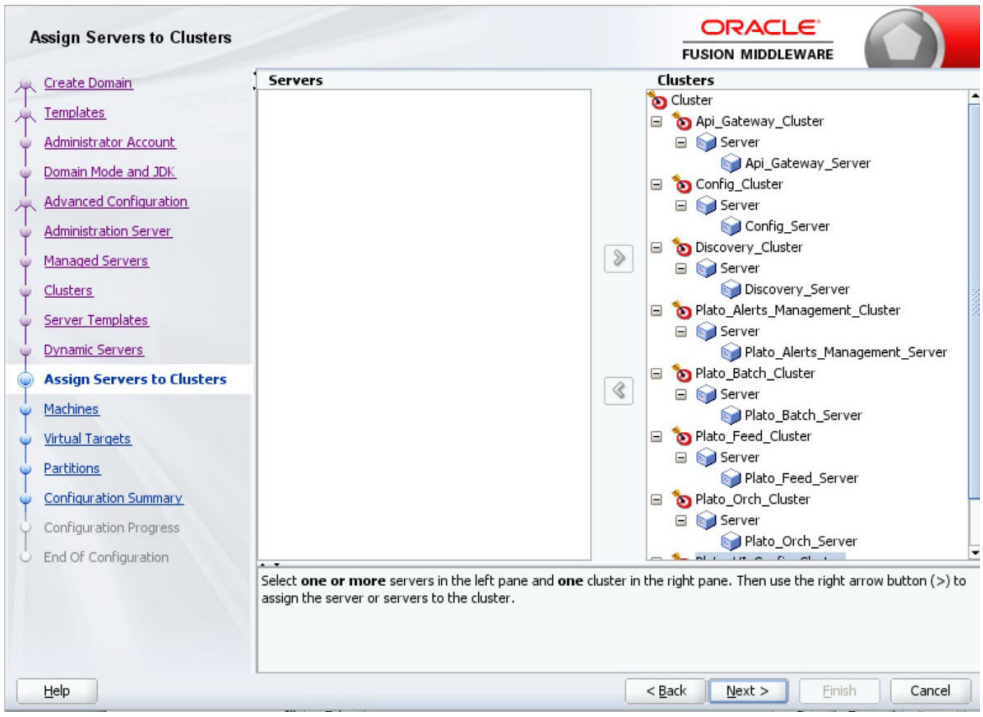
Figure 2-9 Server Template



14. Skip the **Server Templates** and **Dynamic Servers**, and click **Next**.

    The **Assign Servers to Clusters** screen displays.

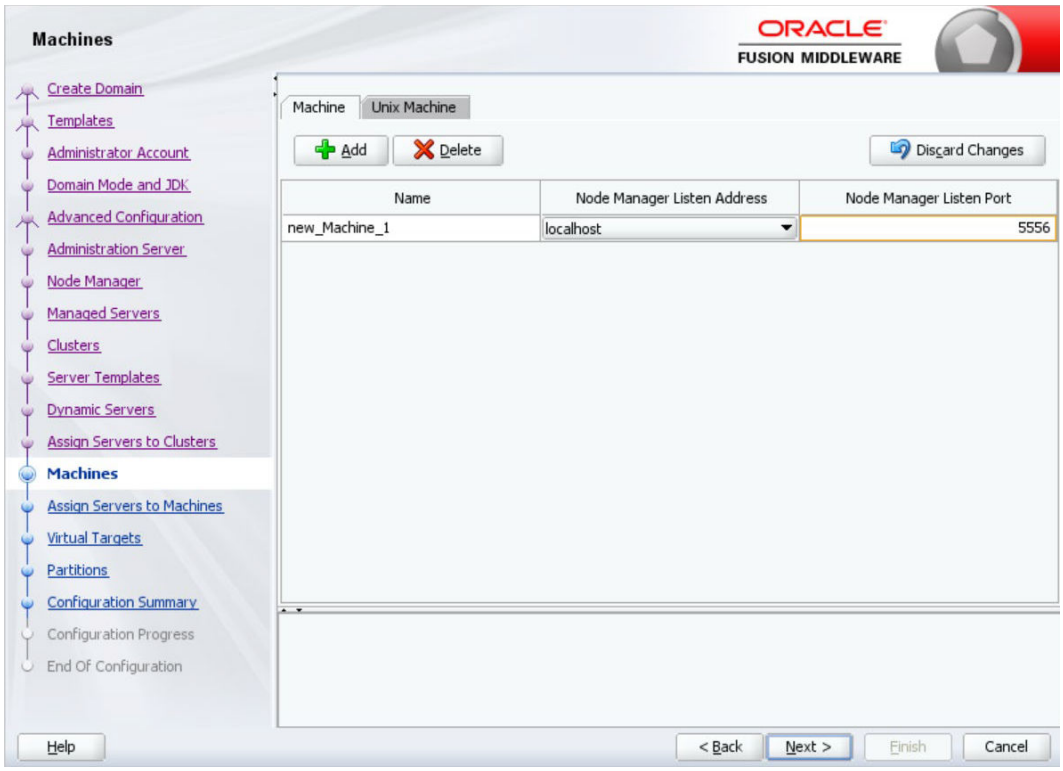
Figure 2-10 Assign Servers to Clusters



15. Assign clusters with servers, and click **Next**.

    The **Machines** screen displays.

Figure 2-11 Machines



16. Click **Add** to add **Machine** or **Unix Machine** based on the operating system and configure **Name**, **Node Manager Listen Address**, and **Node Manager Listen Port** as required.
17. Click **Next**.

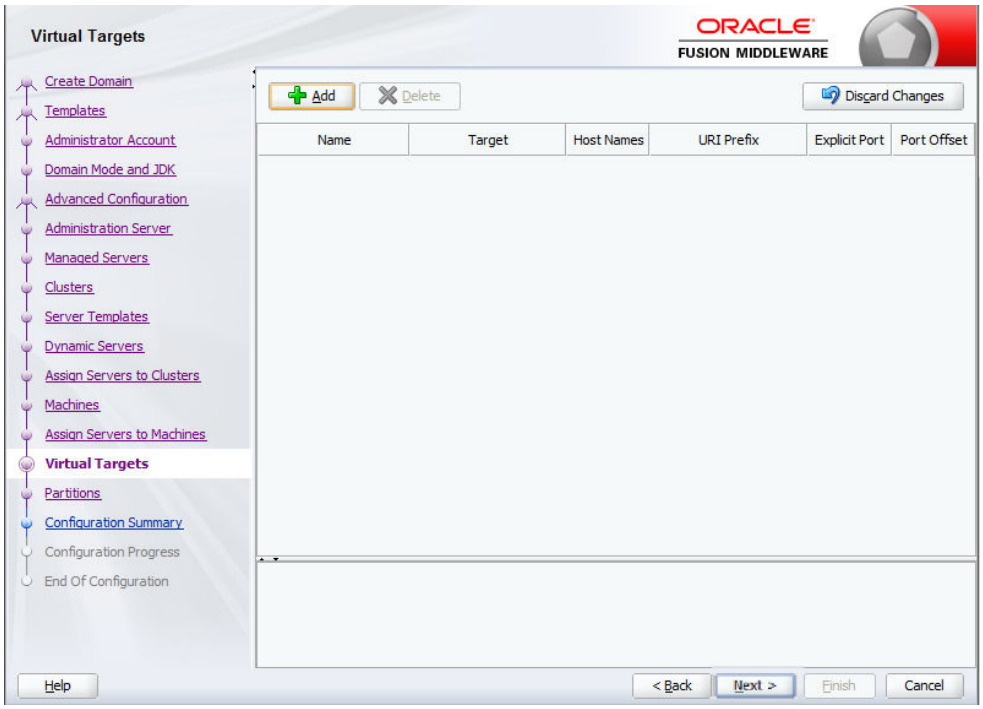
    The **Assign Servers to Machines** screen displays.

Figure 2-12 Assign Servers to Machines

18. Map all the managed servers under the machine created, and click **Next**.

    The **Virtual Targets** screen displays.

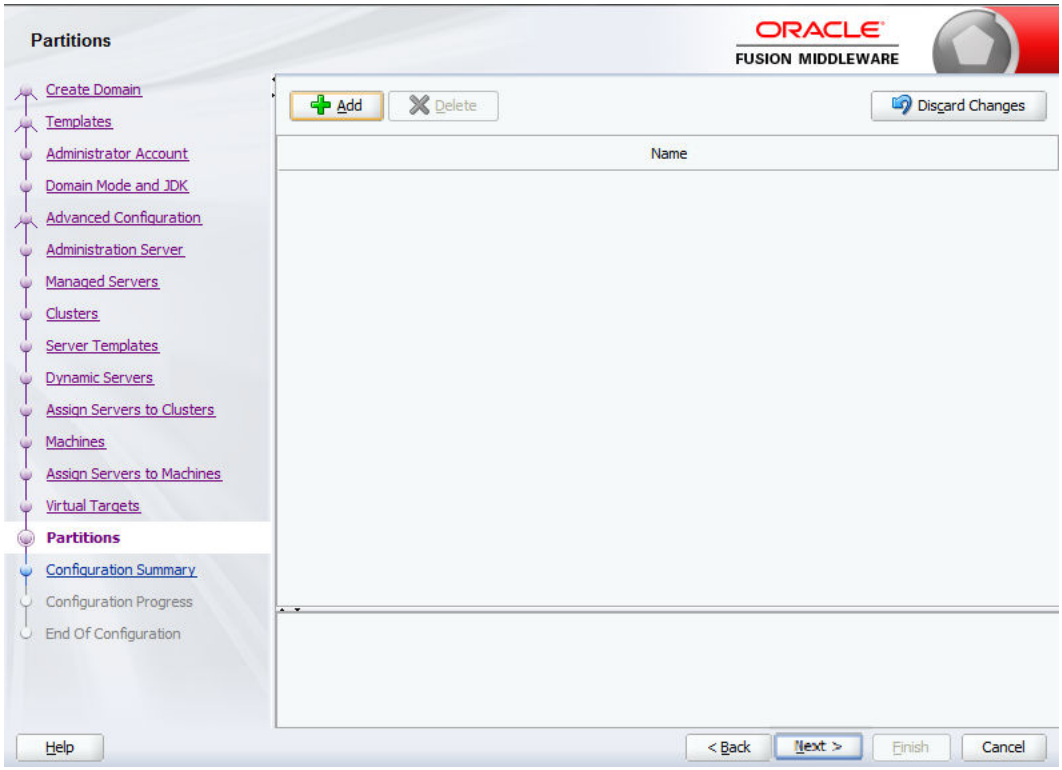
Figure 2-13 Virtual Targets



19. Skip or configure **Virtual Targets**, and click **Next**.

The **Partitions** screen displays.

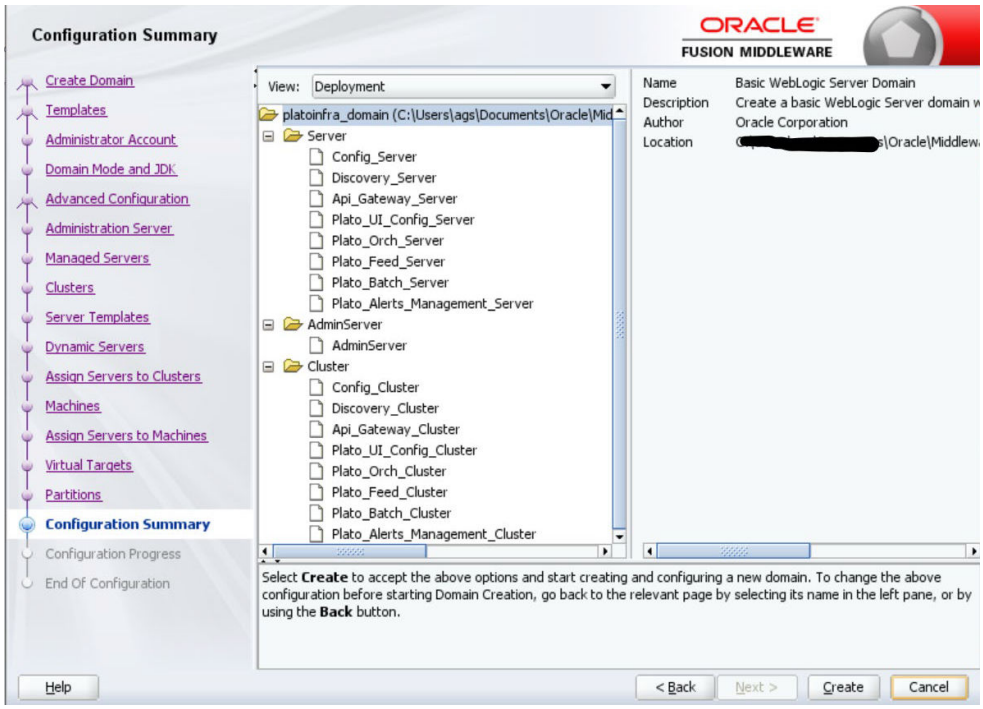
Figure 2-14 Partitions



20. Skip or configure **Partitions**, and click **Next**.

    The **Configuration Summary** screen displays.

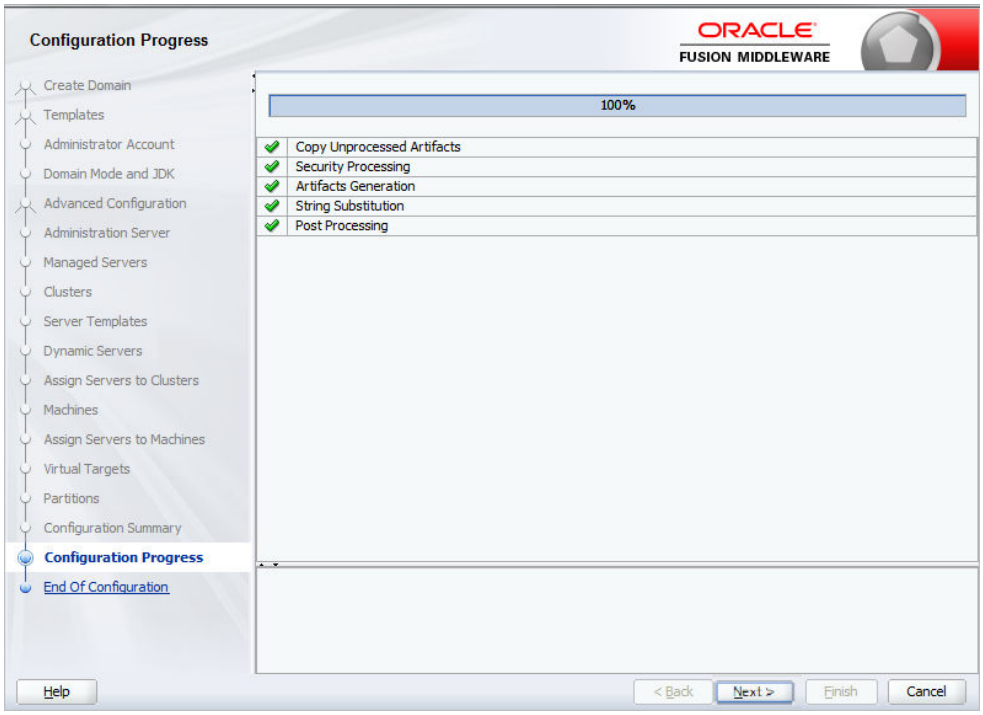
Figure 2-15 Configuration Summary



21. Check the **Configuration Summary** and confirm creating the domain.
22. Click **Next**.

    The **Configuration Progress** screen displays.

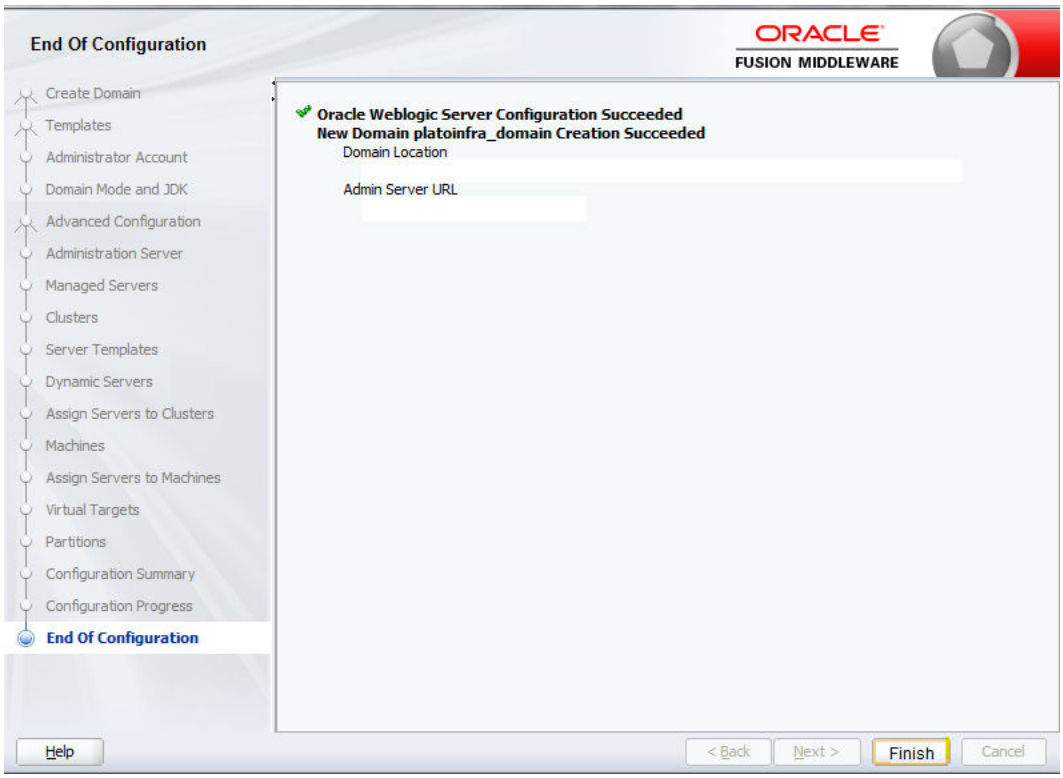
Figure 2-16 Configuration Progress



23. Check the **Configuration Progress** screen, and click **Next**.

    The **End of Configuration** screen displays.

Figure 2-17 End of Configuration



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

    The configuration is completed successfully.

2.2 Post Domain Creation Configurations

This topic provides the systematic instructions for the configurations to be performed once the domain is created.

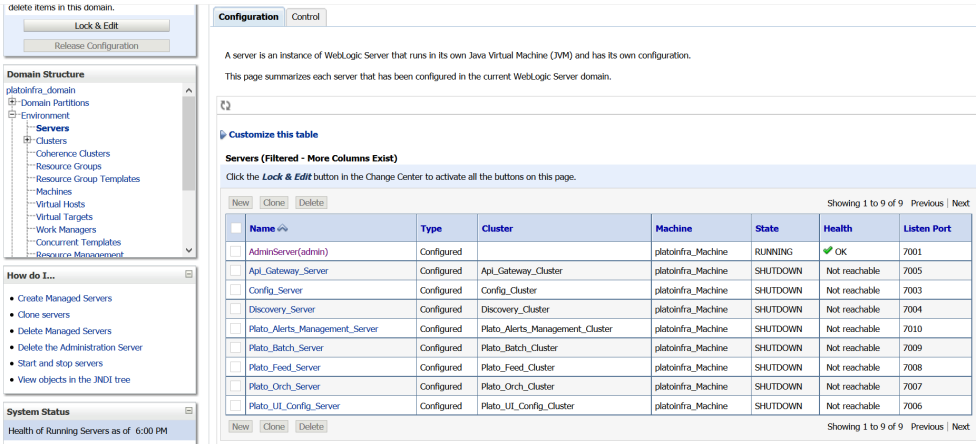
> **Note:**
>
> Once domain creation and cluster configuration is done, refer to Oracle Fusion Middleware documents for more details on how to start an admin server, node manager, and managed servers.

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`. Run `.sh` if operating system is linux.
4. Open `/user_projects/domains/platoinfra_domain/bin`.

5. Run `setNMJavaHome.cmd`. Run `.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:Under **Machine**, select **platoinfra_Machine**.
   a. **Machine**
   b. **platoinfra_Machine**
   c. **Node Manager**
   d. **Type**
   e. **Plain**
   f. **Save**
9. Open `/user_projects/domains/platoinfra_domain/bin`.
10. Run `startNodeManager.cmd`. Run `.sh` if operating system is linux.
11. Start all managed servers.
12. Access the **Oracle WebLogic Administration Console**.
13. From **Domain Structure**, Click **Services** and verify the **Servers (Filtered - More Coloumns Exist)** table.

    The **Summary of Services** screen displays.

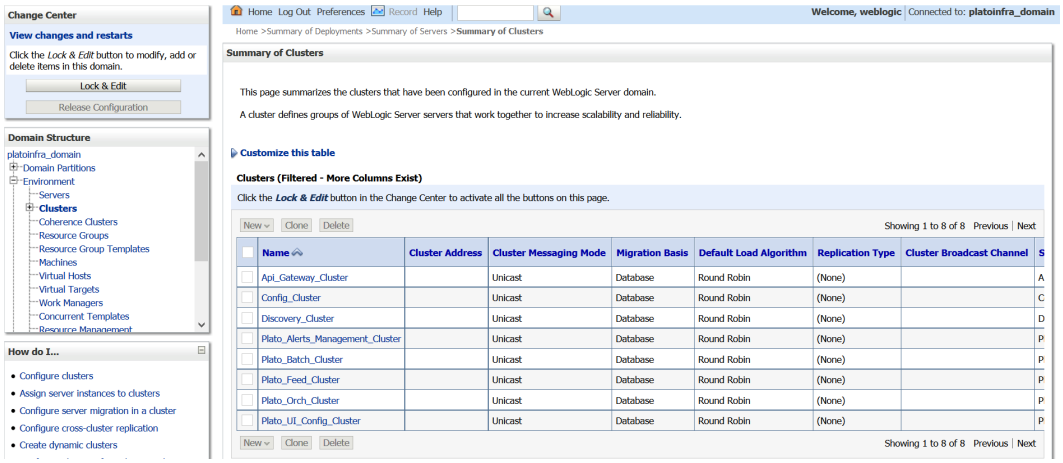
Figure 2-18 Summary of Services



14. Click **Clusters** and verify the **Clusters (Filtered - More Coloumns Exist)** table.

    The **Summary of Clusters** screen displays.

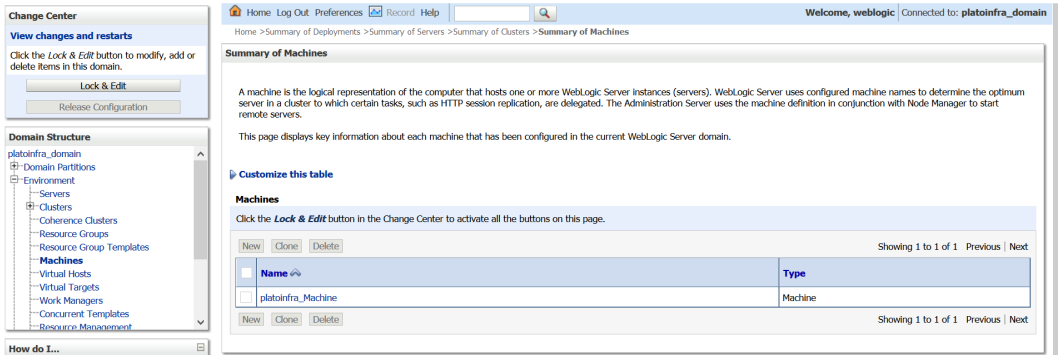
Figure 2-19 Summary of Clusters



15. Click **Machines** and verify the **Machines (Filtered - More Coloumns Exist)** table.

    The **Summary of Machines** screen displays.

Figure 2-20 Summary of Machines



3
Datasource Creation

This topic provides the systematic instruction to create datasource.

- Create Datasource

3.1 Create Datasource

Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

Perform the following steps to create data source:

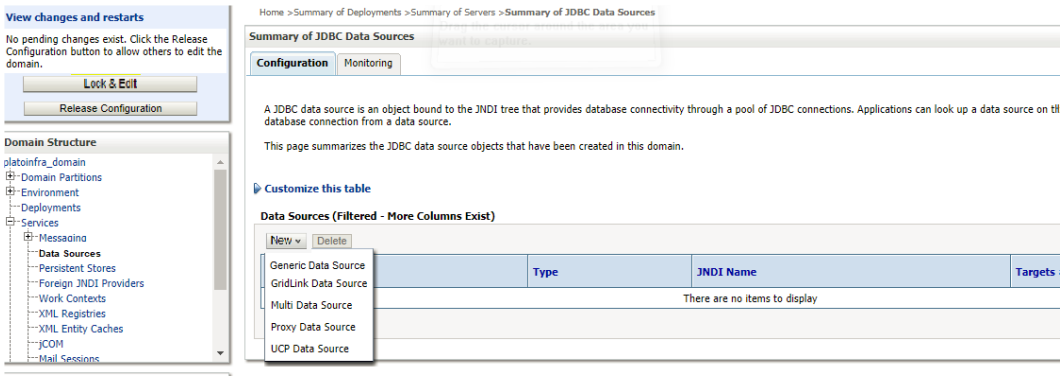
1. Start **AdminServer** and **Node Manager**.

> **Note:**
>
> Make sure that all the managed servers (targets) are in running mode.

2. On **Domain Structure**, click **Services**. Under **Services**, click **Data Sources**.
3. On the **Data Sources (Filtered - More Columns Exist)** table, click **New** and select **Generic Data Source** from drop-down list.

   The **Summary of JDBC Data Sources** screen displays.

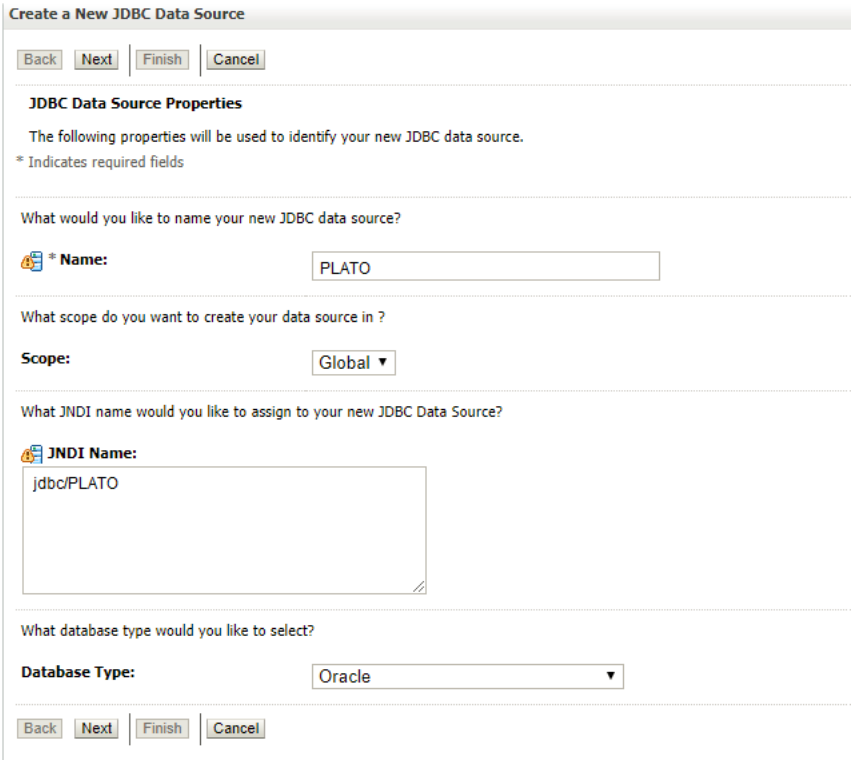
Figure 3-1 Summary of JDBC Data Source



4. Specify the **Name** and **JNDI Name** on **Create a New JDBC Data Source** and click **Next**.

   The **Create a New JDBC Data Source** screen displays.

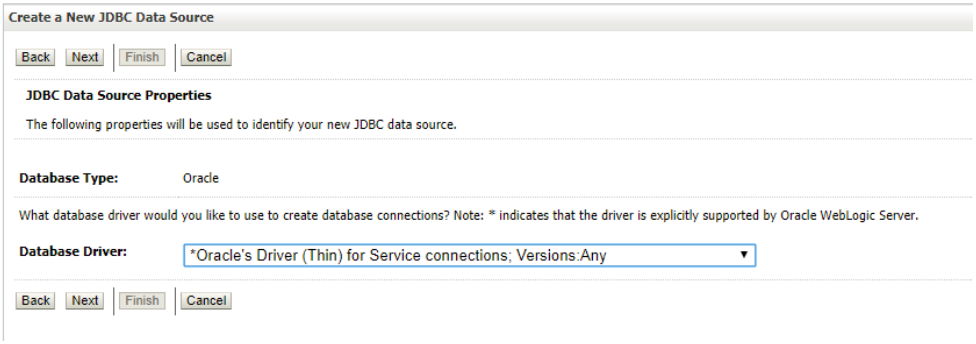
Figure 3-2 Create a New JDBC Data Source



5. On **Database Driver**, select **Thin for Service Connections (Instant)** from drop-down list and click **Next**.

   The **Create a New JDBC Data Source - Database Driver** screen displays.

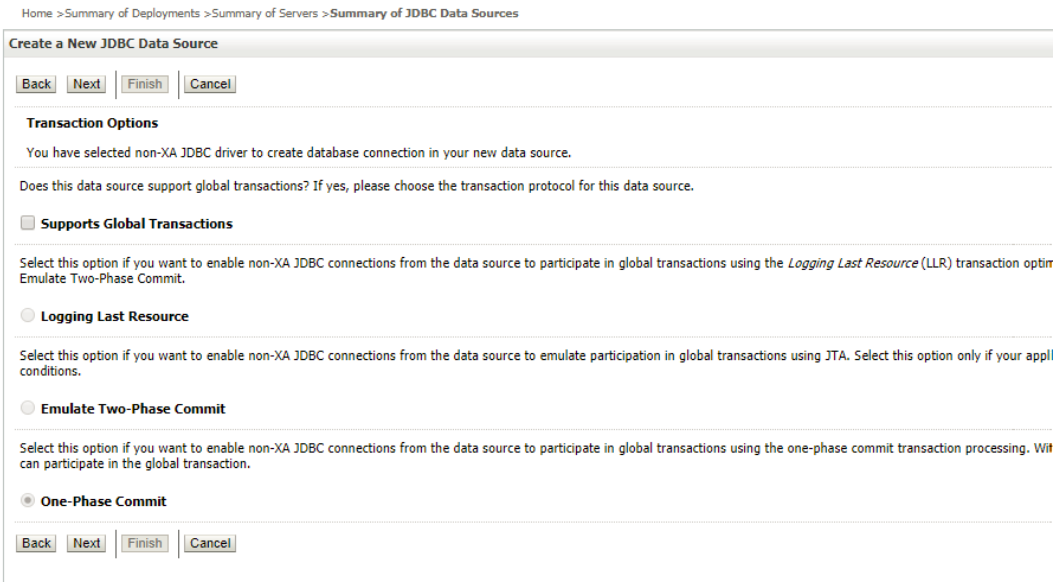
Figure 3-3 Create a New JDBC Data Source - Database Driver



6. Uncheck the **Supports Global Transactions** and click **Next**.

   The **Create a New JDBC Data Source - Transaction Options** screen displays.

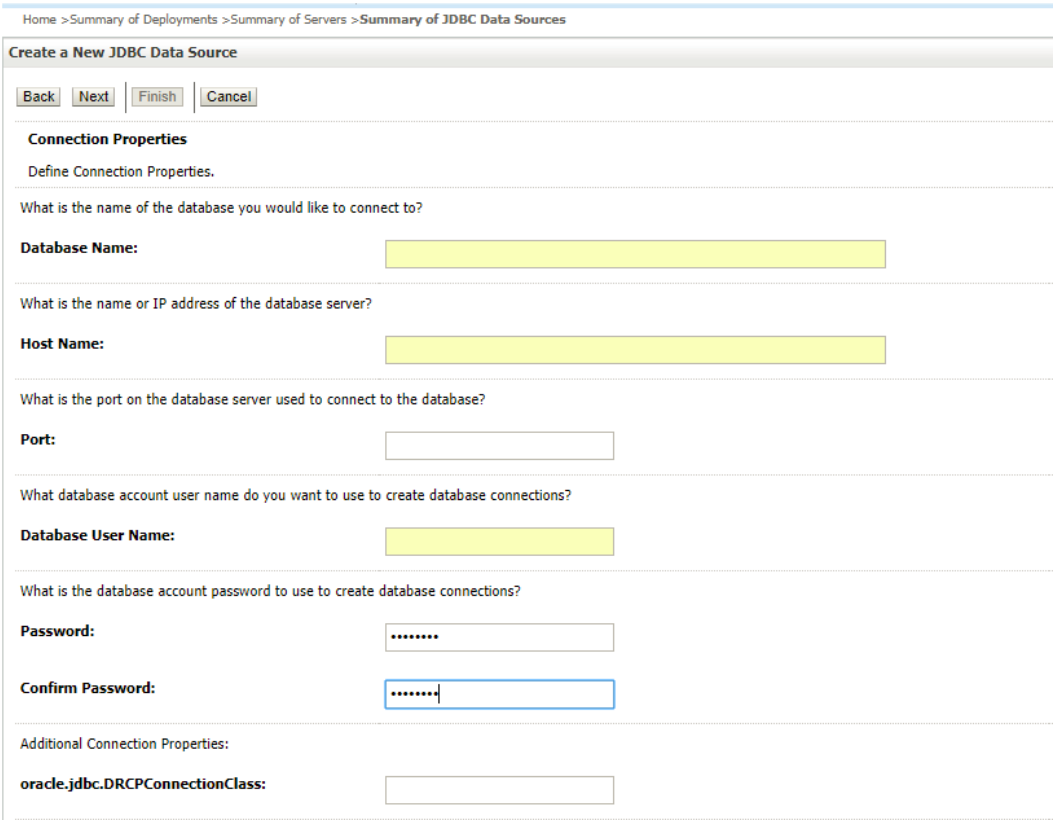
Figure 3-4 Create a New JDBC Data Source - Transaction Options



7. Specify the required fields on **Connection Properties**.

   The **Create a New JDBC Data Source - Connection Properties** screen displays.

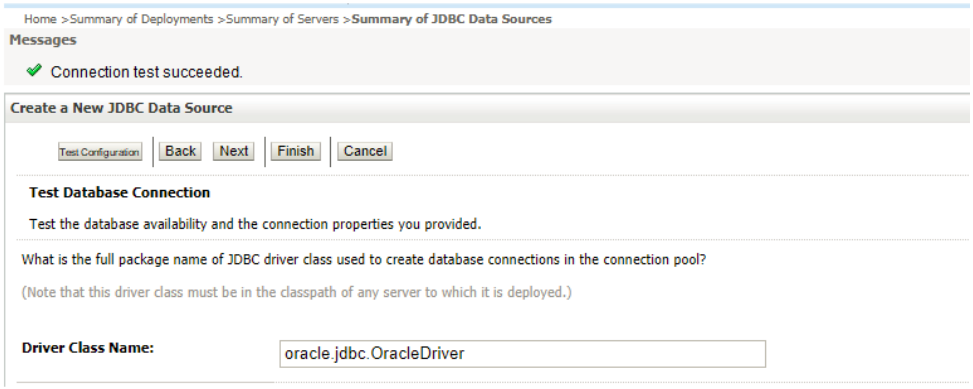
Figure 3-5 Create a New JDBC Data Source - Connection Properties



8. Click **Next** to test connection.

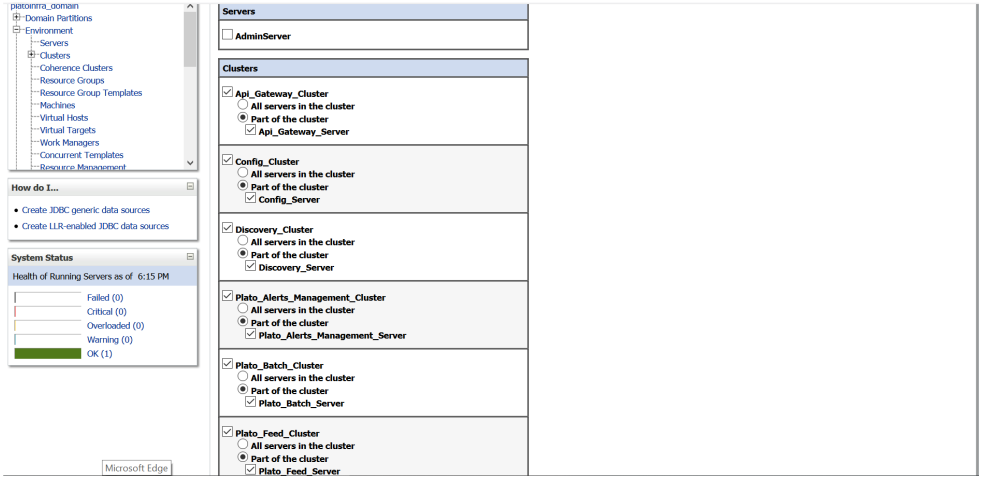
   The **Connection test succeeded** message displays.

Figure 3-6 Connection test succeeded - Message



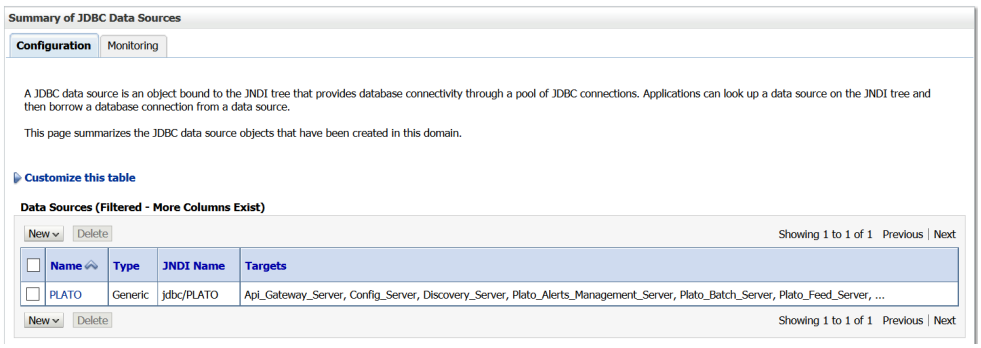
9. Select the target **Servers** and **Clusters** to deploy the data source and click **Next**.

Figure 3-7 Servers and Clusters



10. Go to **Summary of JDBC Data Sources**.
11. On the **Data Sources (Filtered - More Columns Exist)** table, verify the **JNDI Name** and **Targets**.

Figure 3-8 Summary of JDBC Data Source - Configuration



12. On the **Change Center**, click **Activate Changes** once the details are confirmed.

    All the changes are activated.

Figure 3-9 Change Center

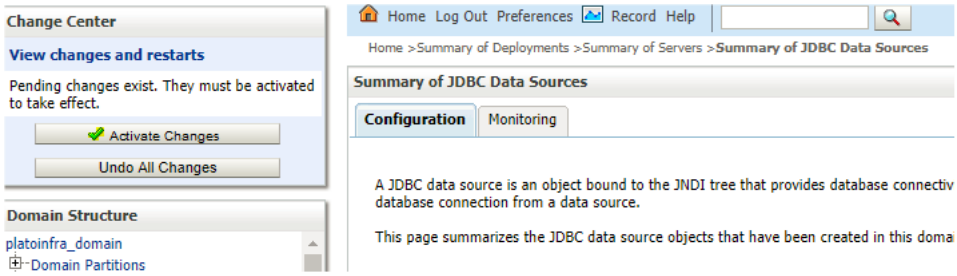
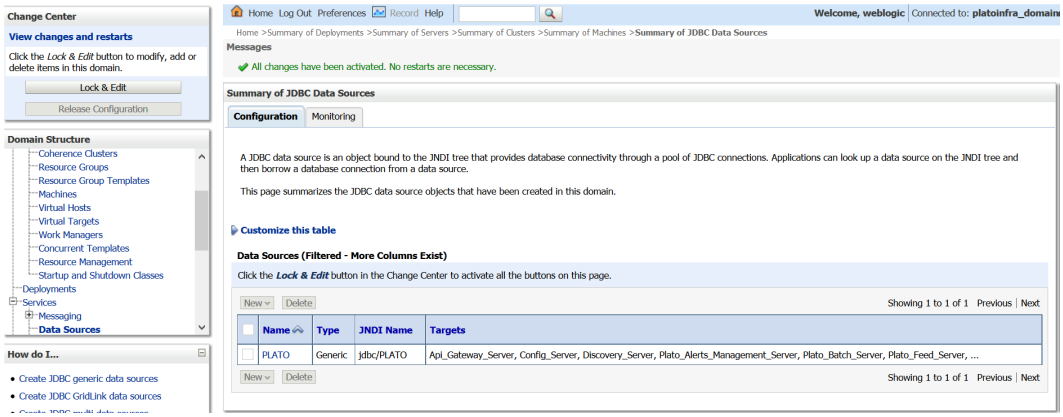


Figure 3-10 Changes - Success Message



4
Deploy Application

This topic provides the systematic instructions to deploy the application.

- Deploy Application

4.1 Deploy Application

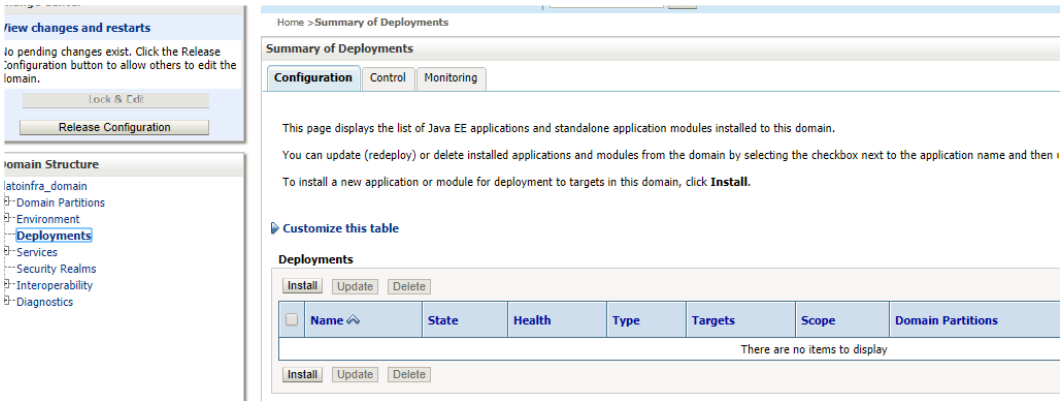
Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

The steps for deploying archives as an application in WebLogic are the same for all of the above except the managed server and the domain, where we deploy differs.

1. Navigate to left menu and click **Domain Structure**.
2. On **Domain Structure**, click **Deployments**.

   The **Summary of Deployments** screen displays.

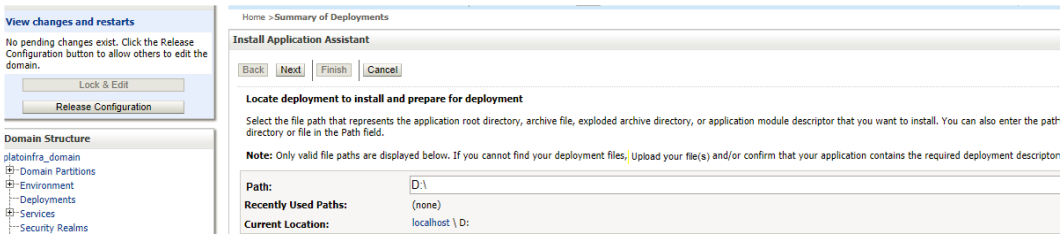
Figure 4-1 Summary of Deployments



3. On the **Change Center**, click **Lock and Edit**.
4. On the **Deployments** table, click **Install**.

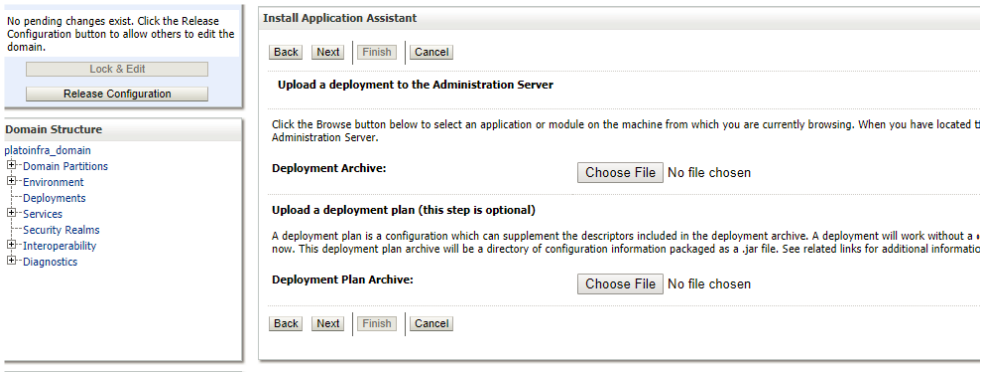
   The **Install Application Assistant** screen displays.

Figure 4-2 Install Application Assistant



5. Click **Upload your file(s)** to select archive. On **Delpoyment Archive**, select **Choose File**.

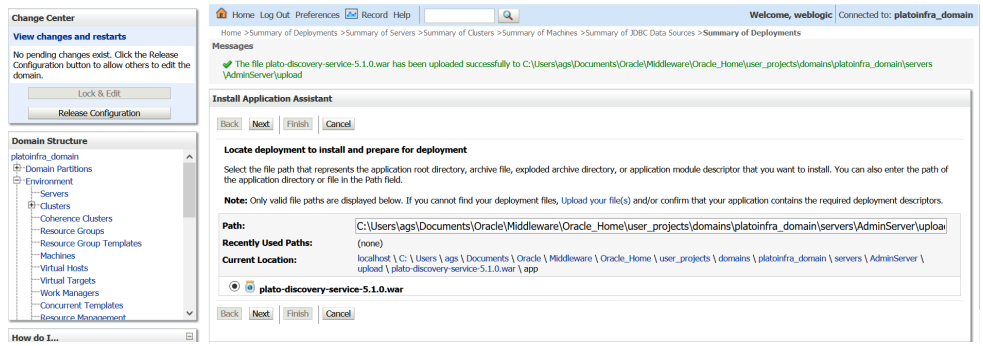
Figure 4-3 Install Application Assistant



6. Once the Archive is uploaded, click **Next**.

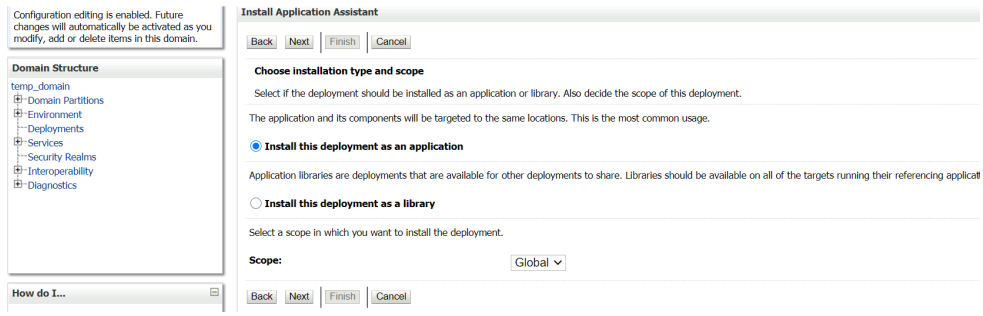
   The file is uploaded successfully.

Figure 4-4 File Upload - Success Message



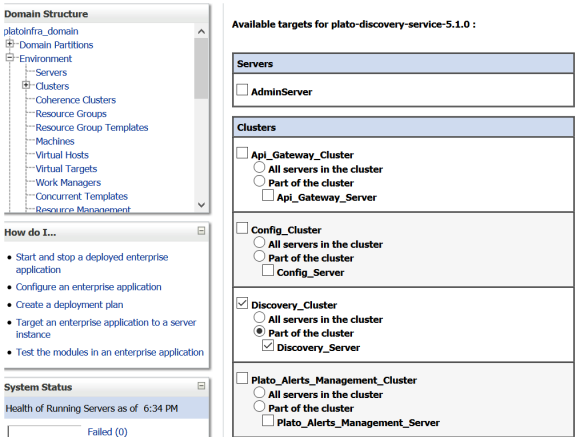
7. Click **Next**.
8. Select **Install this deployment as an application** and click **Next**.

Figure 4-5 Install Application Assistant



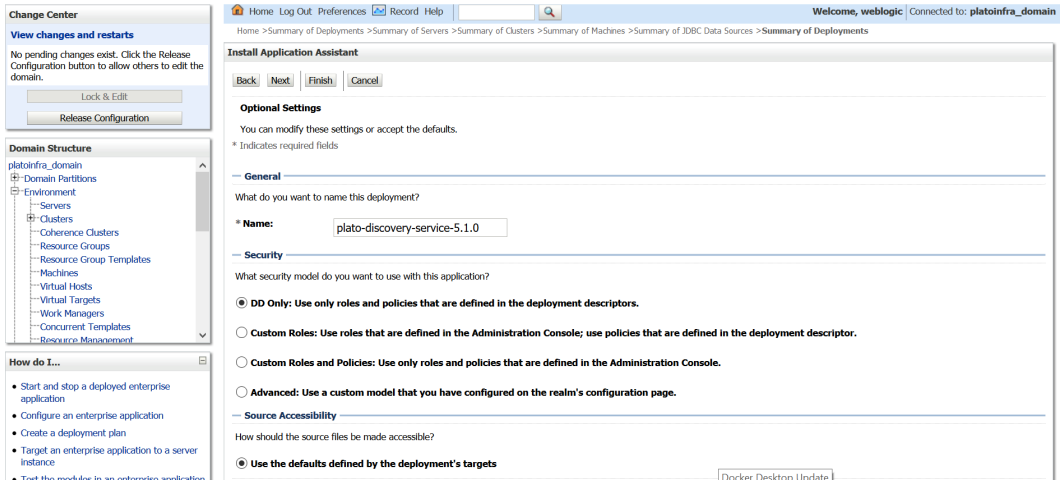
9. Select the target **Servers** and **Clusters** to deploy.

Figure 4-6 Available targets



10. Click **Next**.

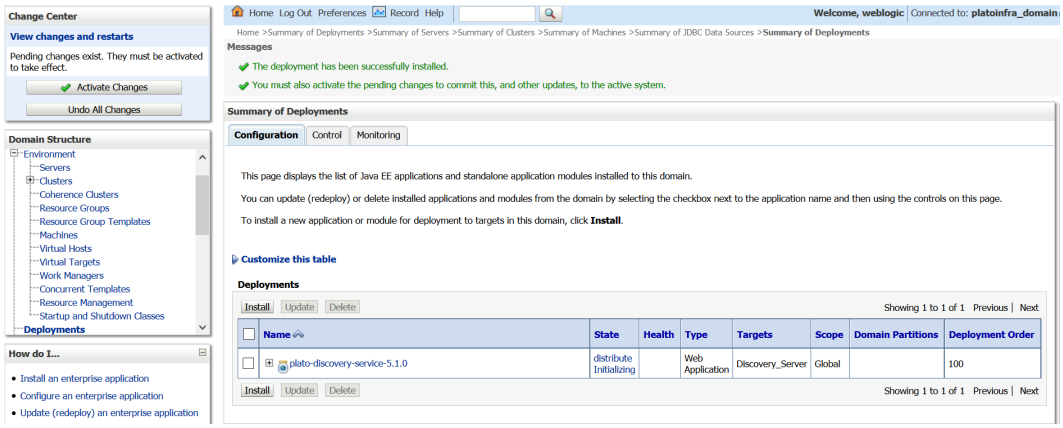
Figure 4-7 Install Application Assistant



11. Click **Finish**.

The deployment is successfully completed.

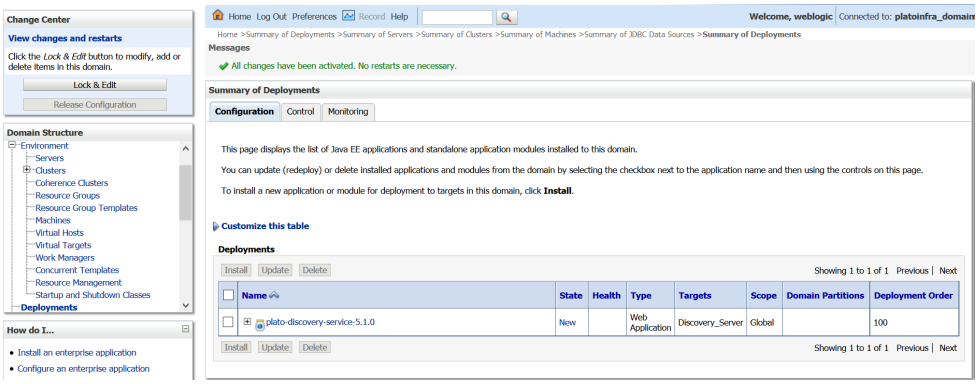
Figure 4-8 Deployment - Success Message



12. On **Change Center**, click **Activate Changes**.

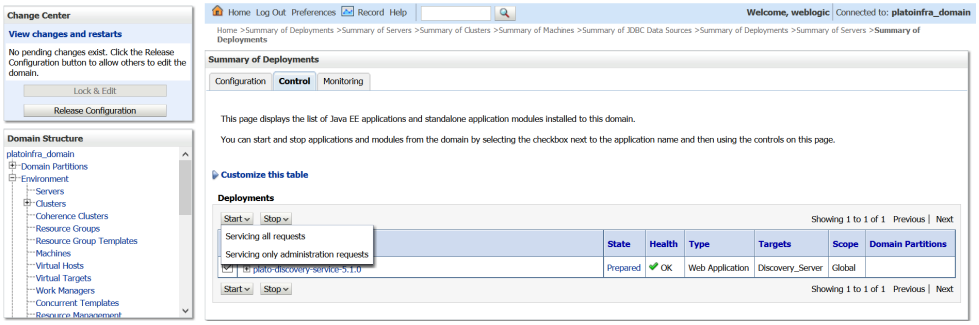
    All the changes are activated.

Figure 4-9 Activate Changes - Success Message



13. On the **Summary of Deployments**, click **Control** to change the status from prepared to active.

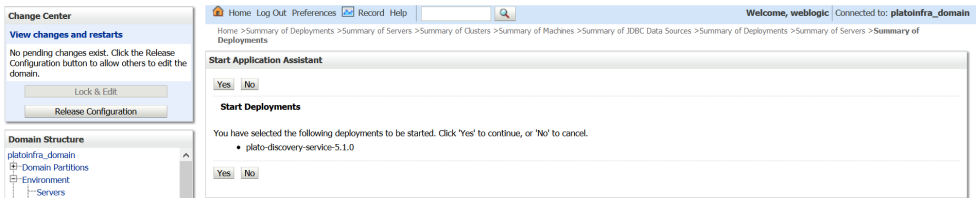
Figure 4-10 Summary of Deployments - Control



14. On **Deployments** table, select the server. Click **Start** and select **Servicing all requests** from the drop-down list.

    Start Application Assistant screen displays.

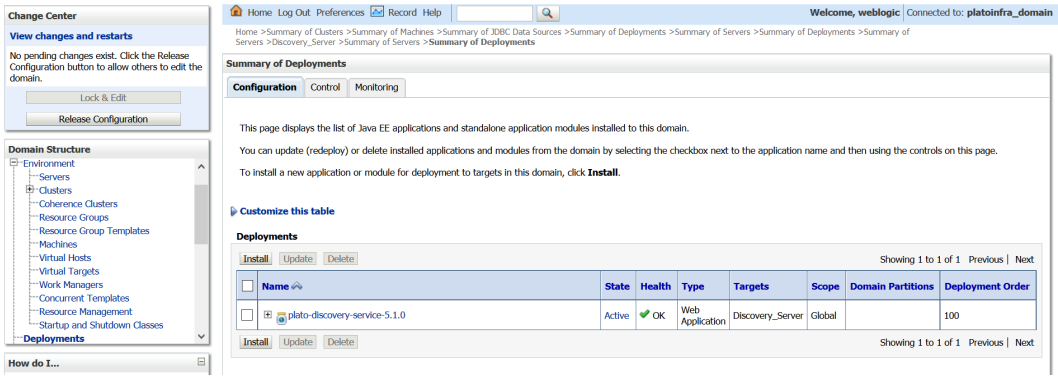
Figure 4-11 Start Application Assistant



15. Click **Yes**.

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

Figure 4-12 Summary of Deployments - Configuration



5

Undeploy Application

This topic provides the systematic instructions to undeploy the application.

- Undeploy Application

5.1 Undeploy Application

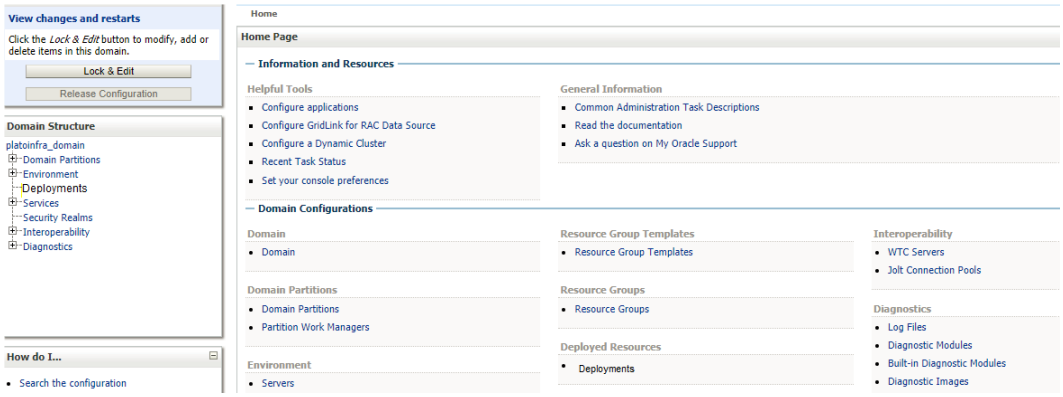
Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

Perform the following steps to undeploy the application:

1. On **Domain Structure**, click **Deployments**.

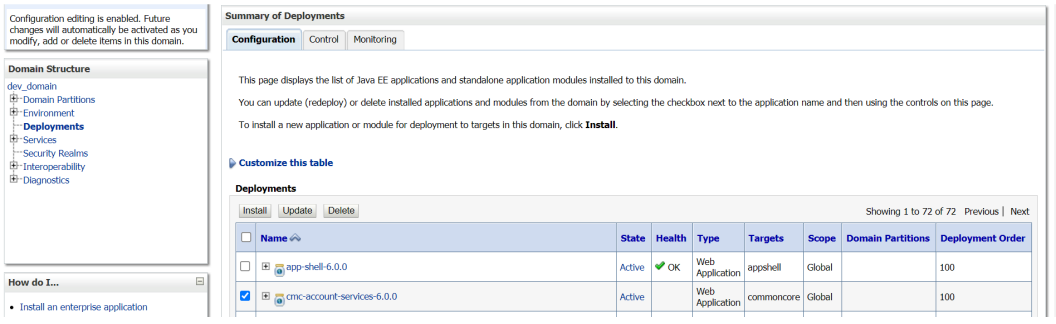
   The **Summary of Deployments** screen displays.

Figure 5-1 Summary of Deployments



2. Click **Lock and Edit**.

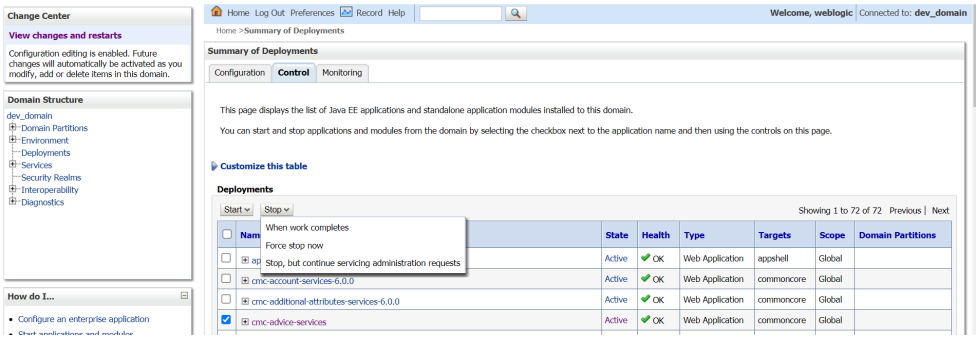
Figure 5-2 Lock and Edit



3. On the **Deployments** table, select the service that needs to be undeployed in Deployments.

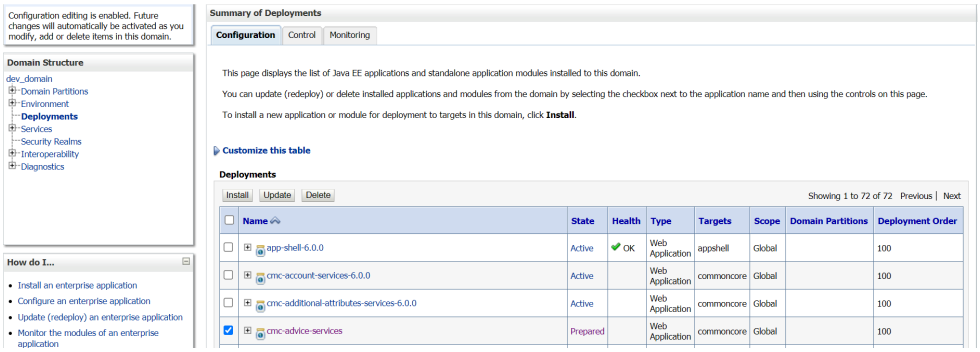
4. Go to **Control**. On the **Deployments** table, click **Stop** and select **Force stop now** from the drop-down list.

Figure 5-3 Summary of Deployments - Control



5. Once the status is changed to **Prepared** state, go to **Configuration**.

Figure 5-4 Summary of Deployments



6. Select the service again and click **Delete** to undeploy the service.

6
Restart Servers

This topic provides the systematic instruction to restart the server.

- Restart Servers

6.1 Restart Servers

Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

Perform the following steps to restart the server:

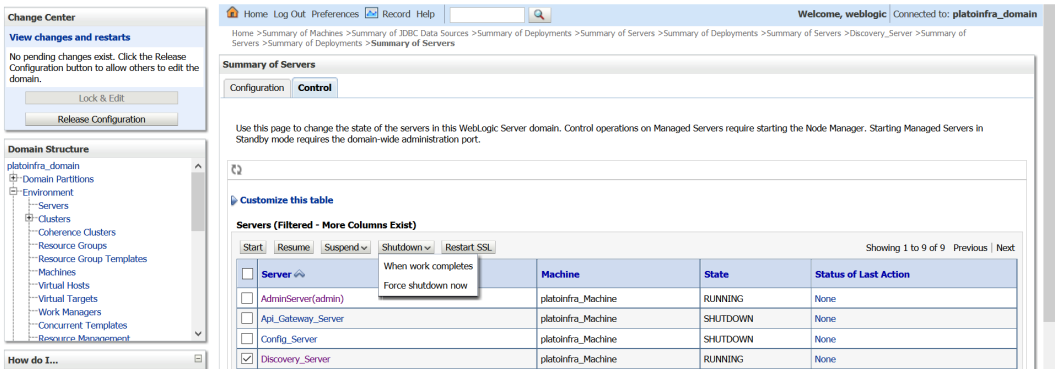
1. On Domain Structure, click **Environment**. Under **Environment**, click **Servers**.

   The **Summary of Servers** screen displays.

2. On the **Summary of Servers** screen, click **Control**.

   The **Summary of Servers - Control** screen displays.

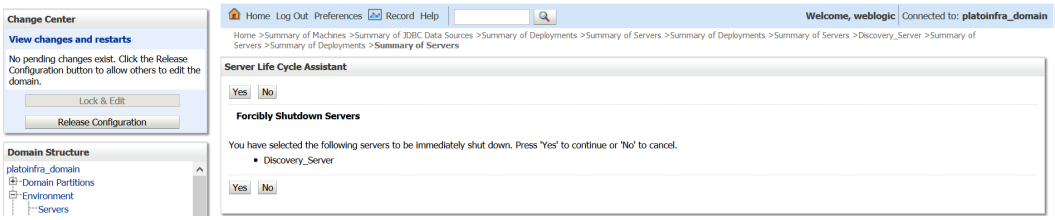
Figure 6-1 Summary of Servers - Control



3. On the **Servers (Filtered - More Columns Exist)** table, select the server. Click **Shutdown** and select the required option from the drop-down list.

   The **Server Life Cycle Assistant** screen displays.

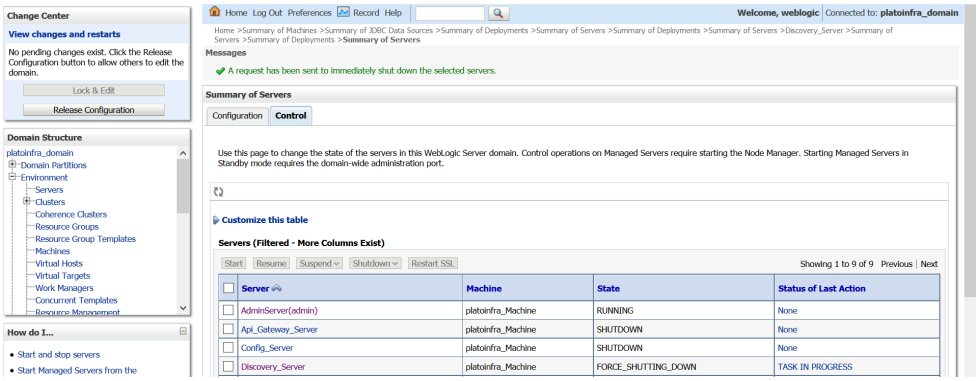
Figure 6-2 Server Life Cycle Assistant



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

   A request is sent to immediately shut down the selected server.

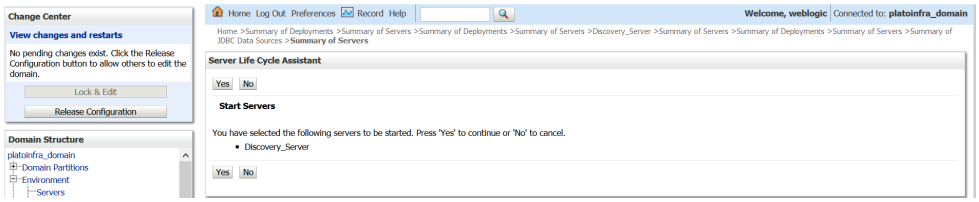
Figure 6-3 Request Sent - Success Message



5. On the **Servers (Filtered - More Columns Exist)** table, select the server and click **Start**.

   The **Server Life Cycle Assistant** screen displays.

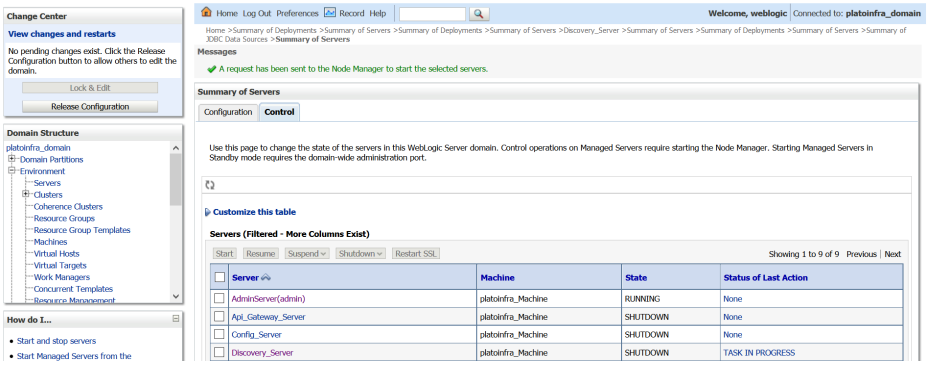
Figure 6-4 Server Life Cycle Assistant



6. Click **Yes** to confirm the action.

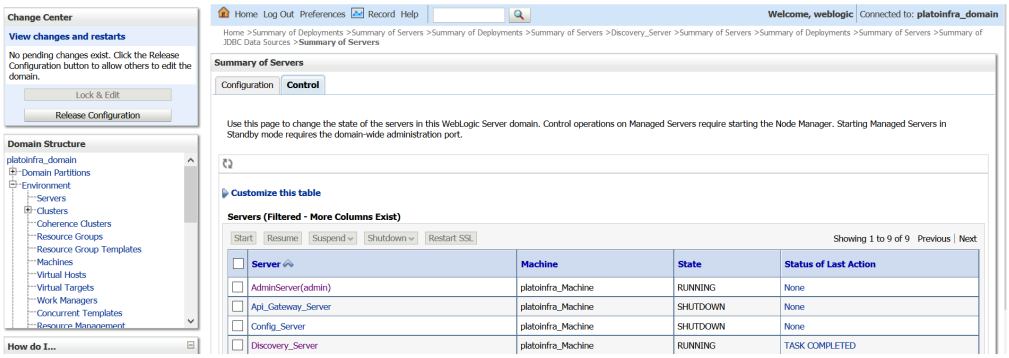
   - A request is sent to the node manager to start the selected servers.

Figure 6-5 Request Sent - Success Message



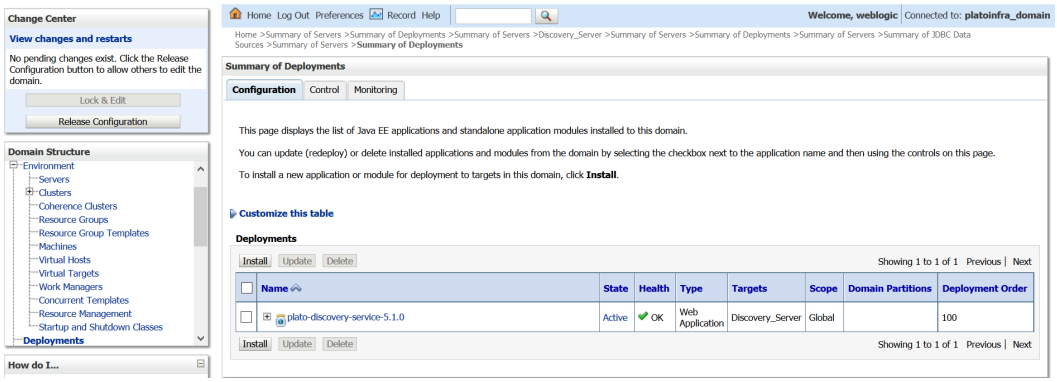
   - All the requested servers are running in the state column.

Figure 6-6 Summary of Servers - Control



7. When all requested servers are running, go to **Summary of Deployments** and check if deployments status is active.

Figure 6-7 Summary of Deployments



7

Check Port Number

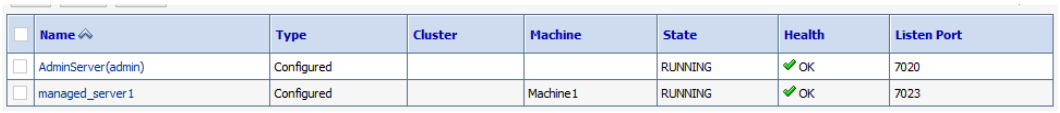
This topic describes the systematic instructions to check the port number.

Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

1. On **Domain Structure**, click **Environment**. Under **Environment**, click **Servers**.

   The **Summary of Servers** screen displays.

Figure 7-1 Summary of Servers



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

2. On the **Servers (Filtered - More Columns Exist)** table, check all the listed servers.

8
WebLogic Embedded LDAP Setup

This topic provides the information to configure the Weblogic Embedded LDAP server for Oracle Banking Microservices Architecture

- Configure WebLogic LDAP
  This topic provides systematic instructions to configure WebLogic LDAP Setup.
- Create Users
  This topic provides systematic instructions to create users.
- Oracle Banking Microservices Architecture Security Config Table Entries
  This topic describes about Oracle Banking Microservices Architecture Security Config Table Entries.

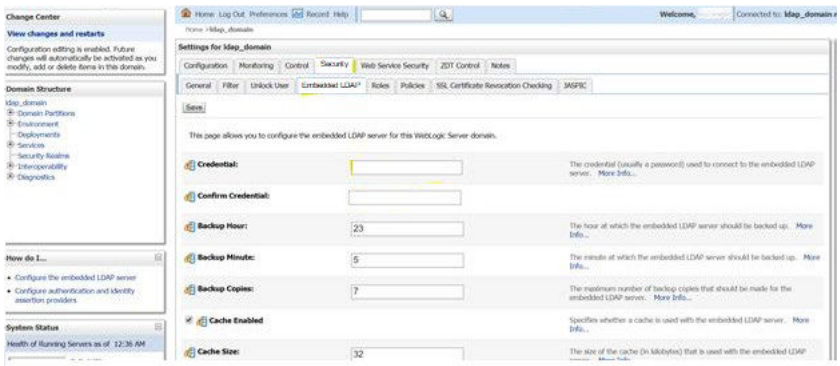
8.1 Configure WebLogic LDAP

This topic provides systematic instructions to configure WebLogic LDAP Setup.

Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

1. Navigate to left panel and click domain name.
2. Under **Settings for ldap_domain**, click **Security** and **Embedded LDAP**.

Figure 8-1 Settings for ldap_domain



3. Set the **Credential** for WebLogic Embedded LDAP store.

> **Note:**
>
> This is needed in the Security Config table.

8.2 Create Users

This topic provides systematic instructions to create users.

Specify **User ID** and **Password** to login to **Oracle WebLogic Administration Console**.

1. On **Domain Structure**, click **Security Realms**.

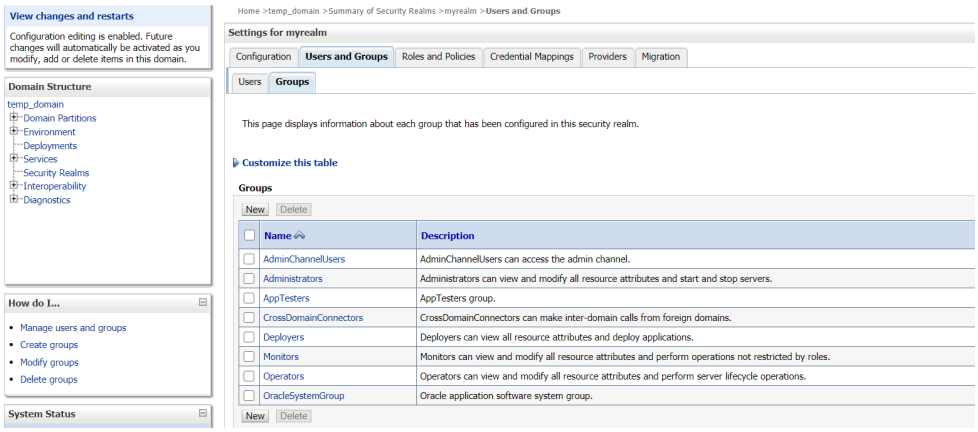
   The **Summary of Security Realms** screen displays.

Figure 8-2 Summary of Security Realms

2. On the **Realms (Filtered - More Columns Exist)** table, click **myrealm**.

   The **Setting of myrealm** screen displays.

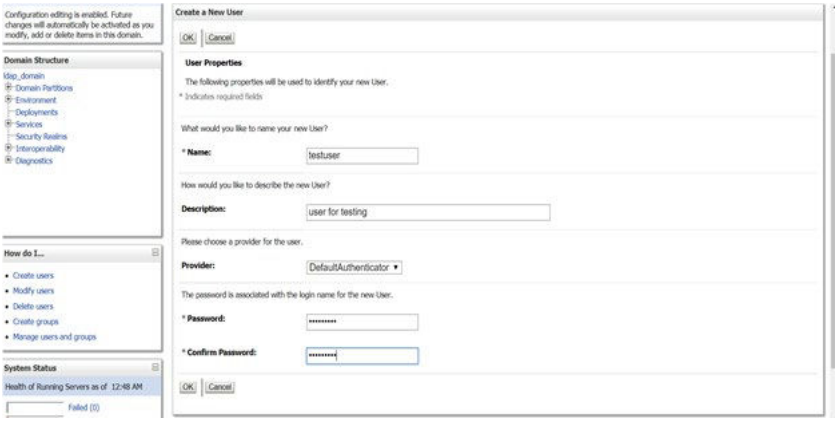
Figure 8-3 Settings for myrealm



3. Under **Settings for myrealm**, click **Users and Groups**
4. Click **Groups**. On the **Groups** table, Click **New**.

   The **Create a New User** screen displays.

Figure 8-4 Create a New User

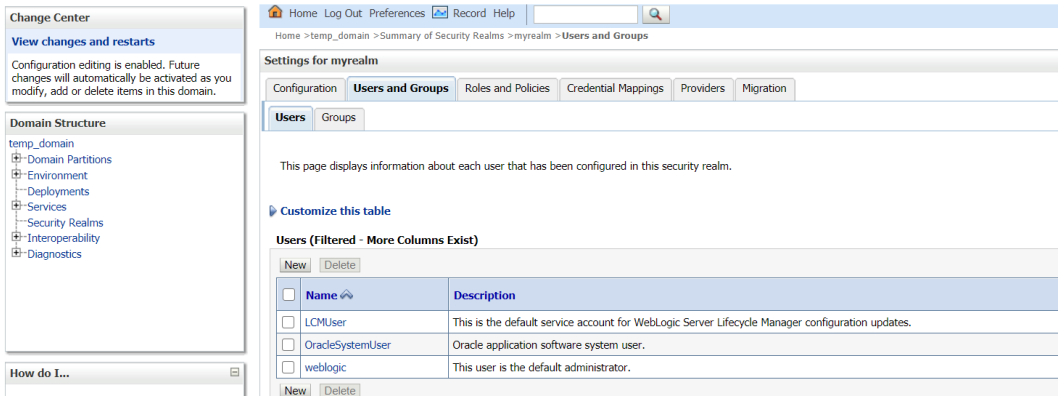


5. Specify all the required details and click **OK**.

   The new group is created.

6. Go to **Settings for myrealm**, click **Users**.

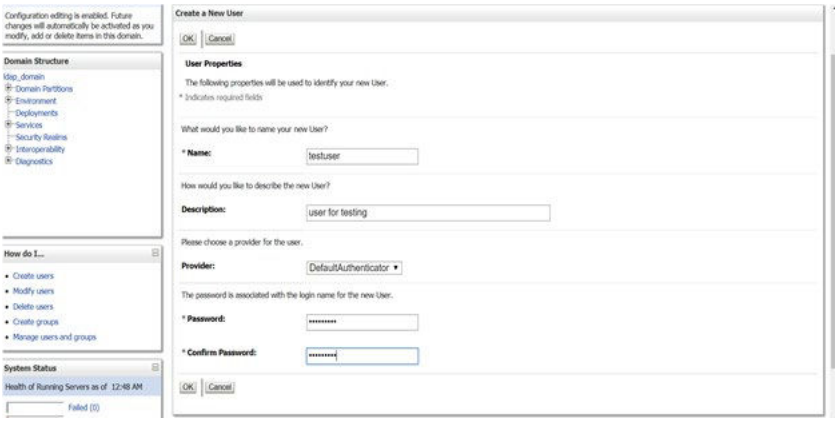
Figure 8-5 Settings for myrealm



7. On the **Users** table, Click **New**.

   The **Create a New User** screen displays.

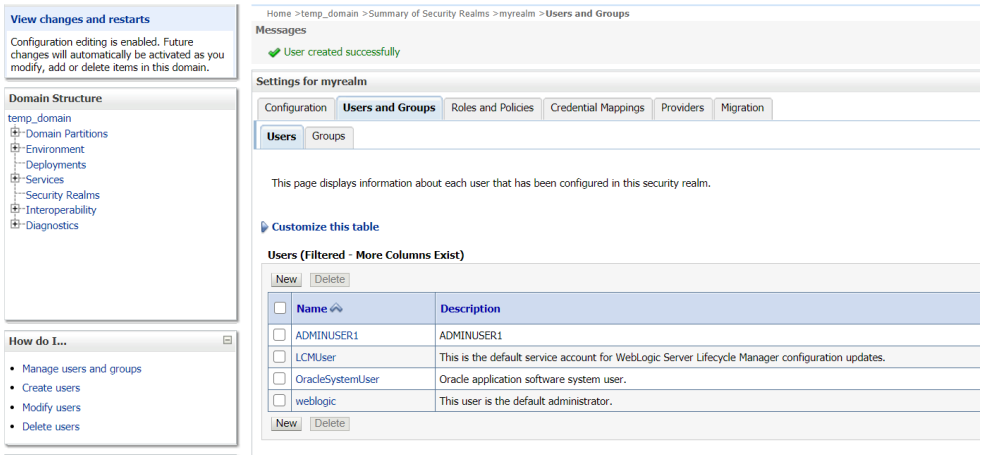
Figure 8-6 Create a New User



8. Specify all the required details and click **OK**.

   The new user is created.

Figure 8-7 User Creation- Success Message

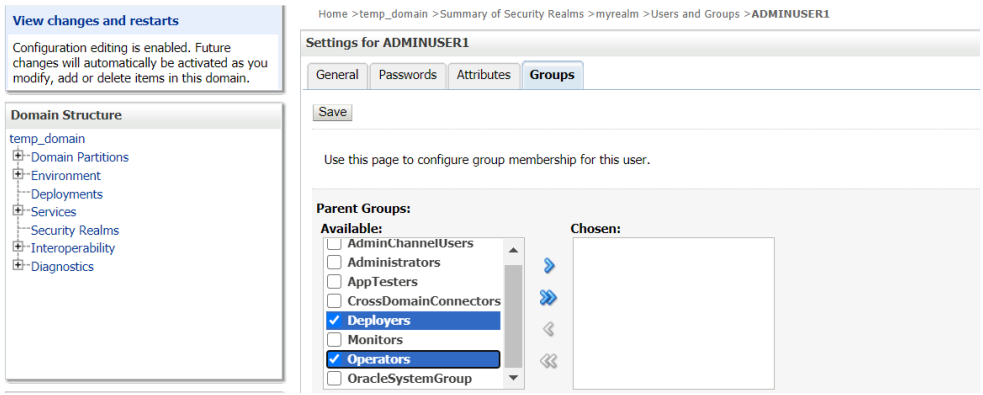


9. On the **Users (Filtered - More Columns Exist)** table, click on the newly created user to assign to some group.

   The **Setting for ADMINUSER1** screen displays.

10. On **Setting for ADMINUSER1** (or whatever your user name is) window, click **Groups** and select the required groups to assign to the user and click single right button as shown below.

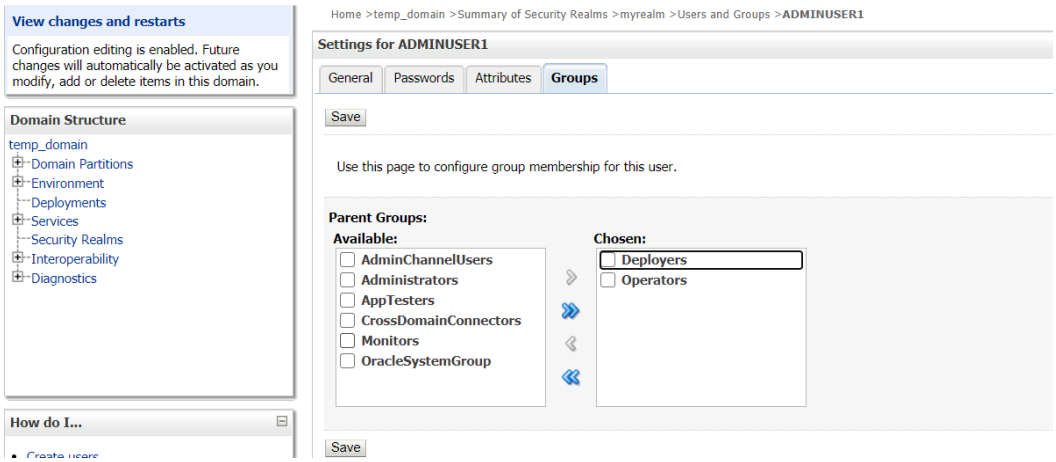
Figure 8-8 Setting for ADMINUSER1



11. Click **Save**.

    The selected groups displays on **Chosen**.

Figure 8-9 Setting for ADMINUSER1



8.3 Oracle Banking Microservices Architecture Security Config Table Entries

This topic describes about 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:

Table 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

Table 8-1 (Cont.) Security Config Table Entries:

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.

9
Oracle Analytic Server Setup

This topic provides the information to configure the Oracle Analytic Server for Oracle Banking Microservices Architecture.

- Prerequisites
  This topic describes about prerequisites for Oracle analytic server setup.
- Start BI Server
  This topic provides systematic instructions to start BI server.
- Upload BI Reports
  This topic provides systematic instructions to upload BI reports.
- Test BI Reports
  This topic provides systematic instructions to test BI reports.

9.1 Prerequisites

This topic describes about prerequisites for Oracle analytic server setup.

- Make sure that the machine is installed with Java JDK.

> **Note:**
>
> For the exact version to be installed, refer to **Software Prerequisites** section in **Release Notes**.

- Oracle Analytics Server 5.5.0

9.2 Start BI Server

This topic provides systematic instructions to start BI server.

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

9.3 Upload BI Reports

This topic provides systematic instructions 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:

Table 9-1 Upload BI Reports

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

3. Upload the catalog object to Analytics Server.

9.4 Test BI Reports

This topic provides systematic instructions to test 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 is generated.

10

How to deploy Plato-Apigateway Router

This topic provides the systematic instructions to deploy the plato-apigateway router.

Router deployment steps

The following 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.

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

Encrypted Password: m4Q1rbtegkWse2s7D2jKfw==

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

Enter pass phrase: Test123

Enter Salt: 0.9412345671234567

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

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
```

Index