Oracle® Communications Digital Business Experience

Solution Deployment Guide

Release 1.0
G23290-01
May 2025

Oracle Communications Digital Business Experience Solution Deployment Guide, Release 1.0

G23290-01

Contents

Preface

This guide provides an overview of the ways to deploy the Oracle Communications Digital Business Experience Solution. This guide also describes the system requirements and procedures for deploying the solution and its components.

Audience

This document is intended for cloud operations administrators and other personnel who are responsible for deploying, configuring, managing, and maintaining the Oracle Communications Digital Business Experience solution.

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 customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

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.

1

Solution Overview and Architecture

The chapter provides an overview of Oracle Communications Digital Business Experience deployed in a cloud native environment.

You can deploy the solution on Oracle Cloud Infrastructure (OCI) and in other cloud native environments as well. When deployed on OCI, the cloud native solution uses OCI Container Engine for Kubernetes (OKE).

About the Solution

Digital Business Experience is a pre-integrated, end-to-end, digital business support system (BSS) for managing experiences and revenue at every stage of your journey. The solution enables you to:

- Use Launch to intuitively design multi-dimensional offers faster with an intelligence-enabled, GUI-based enterprise product catalog.
- Use Siebel CRM to capture customer and partner orders across assisted and unassisted channels.
- Capture, validate, and deliver orders across channels faster using Order and Service Management (OSM) to dynamically orchestrate the order fulfillment of both subscriber orders and service orders.
- Use Billing and Revenue Management (BRM) with Elastic Charging Engine (ECE) and Offline Mediation Controller (OCOMC) to mediate, charge and bill for multi-generational communications services (2G-5G mobile, fixed, satellite) at scale.
- Use Siebel CRM to provide intelligent, personalized, and proactive subscriber care across traditional, digital, assisted, and unassisted channels.

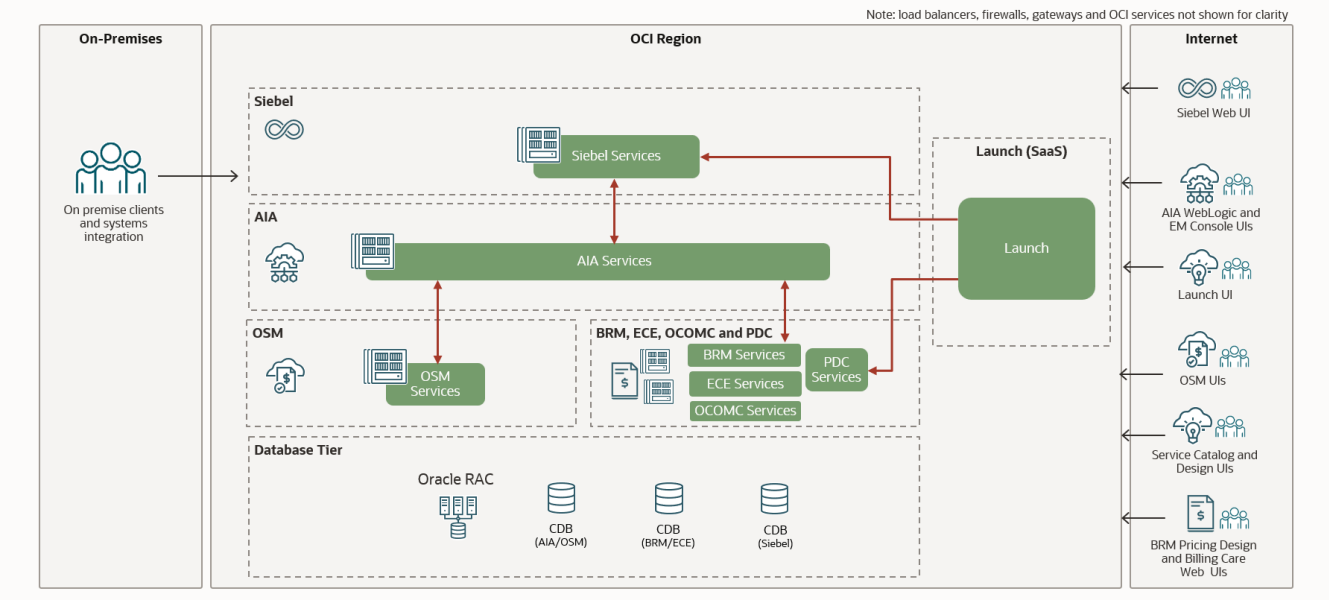
Oracle Application Integration Architecture (Oracle AIA) is the solution's integration framework that provides pre-built integrations and process flows between Siebel, OSM, and BRM using standard integration patterns, business processes, orchestration logic, and common objects and services to connect Oracle applications.

Digital Business Experience Architecture

This section describes and illustrates the deployment architecture and the deployment environment of the solution.

Figure 1-1 illustrates a functional deployment view of the solution within a single OCI region, showing the use of Oracle RAC as the underpinning data tier with container databases for AIA, OSM, BRM, ECE, and Siebel CRM.

Figure 1-1 Digital Business Experience Functional Architecture



> **Note:**
>
> The above diagram is a functional representation of the Digital Business Experience architecture and does not include infrastructure components such as load balancers, firewalls, gateways and OCI services for clarity.

In the above conceptual reference architecture, the solution is deployed on Oracle Cloud Infrastructure Container Engine for Kubernetes (OKE). The solution applications have dedicated clusters.

You can use this architectural blueprint as a starting point for an end-to-end Cash to Care communications application solution that leverages the benefits of OCI. What the actual deployment architecture looks like for your deployment would be dependent on your needs and may differ from this architecture.

> **Note:**
>
> While the conceptual reference architecture above shows the Digital Business Experience Solution deployed on OCI, you can deploy the solution on OCNE, CNCF, and other cloud environments as well.

The Oracle RAC database is privately accessible from the Kubernetes worker nodes. You can use Oracle Base Database Service or the Exadata Cloud Service. This architecture depicts a typical approach to distribute the solution applications across Container Databases (CDBs). For more information about the latest supported database versions, refer to the compatibility matrix of each application.

Oracle Data Guard or Active Data Guard can be used for replication to standby databases.

A bastion host can be configured in a public subnet to allow access to the solution's worker nodes from your network. For example, this can be done using SSH. The web clients for Siebel, BRM and OSM and external integrations connect to load balancers through the Internet gateway. Additional security rule enforcement can be provided by Oracle Cloud Infrastructure Web Application Firewall (WAF) for Internet traffic.

You can use an ingress controller behind an external load balancer to expose solution services outside of the Kubernetes cluster and use that to communicate with the solution application components. The ingress controller monitors the ingress objects and acts on the configuration embedded in these objects to expose HTTP and T3 services to the external network. The load balancer provides a highly reliable, single-point access into the services exposed by the Kubernetes cluster.

Database backups can use the Object Storage Service, and block storage is required for the worker node OS boot volumes and root filesystems. OKE Persistent Volumes (PVs) can use NFS-based persistence for shared storage. Block volumes can be used for all PVs that do not require shared storage.

> **Note:**
>
> You should share an external public load balancer with multiple backends across the solution component applications.

Considerations for Deploying the Solution

You need to consider the following for deploying the solution:

- Detailed deployment architecture and performance: Actual production deployment architectures and solution sizing will vary depending on many factors, which should be discussed with Oracle or your implementation partner prior to and during the deployment project design phase.
- Availability and resiliency: For simplicity, this architecture depicts a single availability domain. Advanced Kubernetes features (such as pod anti-affinity) enable deployments on OCI regions with multiple availability domains to maximize availability. In addition, deployments can be split across regions for geographic redundancy and disaster recovery scenarios. In such models, data replication across RAC instances can be provided using Active Data Guard. The above reference architecture does not consider individual application availability scenarios and these must be factored into your production deployment architecture.

2

Solution Requirements

This chapter describes the requirements for deploying the Digital Business Experience Solution.

Prerequisites for Deploying the Solution Components

Refer to the Digital Business Experience Compatibility Matrix for information about the required solution components.

For information on the prerequisites for your deployment of the solution components, refer to the following:

- Launch Overview, for Launch.
- About Configuring and Deploying Your BRM Cloud Native Environment, for BRM.
- The topic "Before You Begin", in the chapter Deploy Oracle Analytics Server on Oracle Cloud, for OAP.
- Planning and Validating Your Cloud Environment, for SCD.
- Planning and Validating Your Cloud Native Environment, for OSM.
- Generating the OSM Cloud Native Artifacts for the Order-to-Activate Solution, for O2A.
- Requirements for Installing and Configuring Siebel CRM, for Siebel.
- Preparing to Install and Configure Oracle Data Integrator, for ODI.
- Prerequisites for Your AIA Cloud Native Deployment, for AIA.

3

Deploying and Validating the Solution Components

This chapter provides information about deploying and validating the solution components.

Assumptions for Deploying the Solution

Before the deployment process begins, it is assumed that:

- The Solution Specialist can operate and run commands as well as support underlying cloud native competencies and technologies.
- The Solution Specialist can access, operate, and run commands across operating systems, such as Linux, Unix, and so on.
- The Solution Specialist is familiar with different resources required for operations, maintenance, and escalations of the service.
- The cloud native environment infrastructure requirements of all applications in the solution are fulfilled prior to performing the procedures.

See the following topics for details about the activities you perform to deploy and validate the solution components:

- Preparing the cluster (set up the solution infrastructure). See Preparing the Cluster for more information.
- Deploying the individual applications that are part of the solution. See Deploying the Solution Components for more information.
- Validating the solution components. See Validating the Solution Components for more information.

Preparing the Cluster

This section provides information about deploying a comprehensive cloud infrastructure for the Digital Business Experience solution that involves setting up various foundational and service-specific components, enabling secure, scalable, and efficient operations. It also outlines the steps and objectives for deploying key infrastructure elements, including a Virtual Cloud Network (VCN), a Public Bastion for secure access, a Kubernetes Cluster using Oracle Kubernetes Engine (OKE), and an Oracle Base Database Service. Each component serves a distinct purpose and together they form a robust cloud environment suitable for a range of enterprise applications.

Deploy the following key infrastructure elements in the sequence provided below:

- **VCN**: Helps establishing a secure, isolated, and customizable virtual network, which serves as the foundation for deploying various cloud resources. VCN provides full control over the network architecture, including IP address ranges, subnets, route tables, and security lists. See Deploying the Virtual Cloud Network for more information.
- **Public Bastion**: Allows secure access to resources within the VCN without exposing those resources to the public internet. Public Bastion acts as a gateway for administrators and

authorized users to manage resources in private subnets, enhancing the security posture of the environment. See Deploying the Public Bastion for more information.

- **OKE**: Provides a scalable platform for deploying, managing, and automating the operations of containerized applications, ensuring high availability and fault tolerance. See Deploying an Oracle Kubernetes Engine Environment for more information.
- **Oracle Base Database Service**: Provides reliable, high-performance data management, with configurations optimized for the specific needs of the application workloads. See Deploying the Oracle Base Database Service for more information.
- **Kubernetes cluster using Oracle Cloud Native Environment (OCNE)**: The Kubernetes cluster is set up using the OCNE Command Line Interface (CLI) using the `libvirt` provider. It is designed for running cloud native applications at scale, providing a standardized environment for managing microservices, containers, and workloads across various cloud providers. Its benefits include improved agility, scalability, and cost efficiency by automating infrastructure management and simplifying deployment and orchestration of containerized applications. See Deploying a Kubernetes Cluster using Oracle Cloud Native Environment for more information.
- **Cloud Native Computing Foundation (CNCF)**: CNCF is an open source software foundation that promotes the adoption of cloud-native computing. It is a subsidiary of the Linux Foundation that aims to establish a vendor-agnostic community of developers, end users, and IT technology and service providers to collaborate on Open Source projects. CNCF hosts and supports projects, such as Kubernetes, Prometheus, and Envoy, which are essential components of many modern cloud-native architectures. See Deploying a Cloud Native Computing Foundation Environment for more information.

The following sections provide detailed instructions for deploying the infrastructure elements mentioned above.

Deploying the Virtual Cloud Network

This section provides detailed instructions about the following:

- Creating the Virtual Cloud Network (VCN): Helps you to define the IP address space and create the VCN.
- Creating Subnets: Helps you to set up public and private subnets within the VCN, ensuring they are logically separated for different types of resources.
- Configuring Route Tables and Security Lists: Helps you to set up route tables for network traffic management and security lists for controlling ingress and egress traffic.

Creating a VCN

To create a VCN within Oracle Cloud Infrastructure (OCI):

1. Log in to Oracle Cloud Infrastructure using your credentials.
   The OCI home page opens.
2. From the **Navigation** pane, select **Networking**, and then click the **Virtual cloud networks** link.
   The **Create a Virtual Cloud Network** page opens. Provide the following details:
   - In the **Name** text box, enter the desired vcn name. For example, `dbe-vcn`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - In the **CIDR Block** field, enter the CIDR in the **0.0.0.0/0** format. For example, `10.0.0.0/28`.

3. Click **Create VCN**. The VCN is created.

Creating a Service Gateway, NAT Gateway, and Internet Gateway in the VCN

To create a Service Gateway:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, and then select your VCN. On the **Resources** pane, select **Service Gateways**, and then click the **Create Service Gateway** button.
3. In the **Create Service Gateway** page, provide the following details:
   - In the **Name** text box, enter the service gateway name. For example, `serviceGW`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - In the **Services** text box, enter the required services. For example, `All IAD Services in Oracle Services Network`.
   - Click **Create Service Gateway**.

To create a NAT Gateway:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, and then select your VCN. On the **Resources** pane, select **NAT Gateways**, and then click the **Create NAT Gateway** button.
3. In the **Create NAT Gateway** page, provide the following details:
   - In the **Name** text box, enter the NAT gateway name. For example, `NATGW`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - Select the **Ephemeral Public IP Address** option.
   - Click **Create NAT Gateway**.

To create an Internet Gateway:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, and then select your VCN. On the **Resources** pane, select **Internet Gateways**, and then click the **Create Internet Gateway** button.
3. In the **Create Internet Gateway** page, provide the following details:
   - In the **Name** text box, enter the internet gateway name. For example, `InternetGW`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - Click **Create Internet Gateway**.

Creating Public and Private Route Tables in the VCN

To create a public route table:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, and then select your VCN.

3. On the **Resources** pane, select **Route Tables**, and then click **Create Route Table**.
4. In the **Create Route Table** page, provide the following details:
   - In the **Name** text box, enter the public route table name. For example, `rt_publicsubnet`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - Click **Create Route Table**.
5. To attach an internet gateway to the created public route table:
   - From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select your public route table, and then click the **Add Route Rules** button.
     The **Add Route Rules** page opens.
   - From the **Target Type** drop-down list, select the **Internet Gateway** option.
     The **Destination CIDR Block** and **Target Internet Gateway in Compartment** fields are auto-populated.
   - Click **Add Route Rules**.

To create a private route table:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select **Route Tables**, and then click the **Create Route Table** button.
3. In the **Create Route Table** page, provide the following details:
   - In the **Name** text box, enter the public route table name. For example, `rt_privatesubnet`.
   - In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
   - Click **Create Route Table**.
4. To attach a NAT gateway to the created private route table:
   - From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select your private route table, and then click the **Add Route Rules** button.
     The **Add Route Rules** page opens.
   - From the **Target Type** drop-down list, select the **NAT Gateway** option.
     The **Destination CIDR Block** and **Target NAT Gateway in Compartment** fields are auto-populated.
   - Click **Add Route Rules**.
5. To attach a service gateway to the created private route table:
   - From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select your private route table, and then click the **Add Route Rules** button.
     The **Add Route Rules** page opens.
   - From the **Target Type** drop-down list, select the **Service Gateway** option.
     The **Destination Service** and **Target Service Gateway in Compartment** fields are auto-populated.

- Click **Add Route Rules**.

Creating Public, Private, and LB Subnets in the VCN

To create a public subnet:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select **Subnets**, and then click the **Create Subnet** button.
   The **Create Subnet** page opens.
3. In the **Name** text box, enter the public subnet name. For example, `Public subnet`.
4. In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
5. From the **Subnet Type** field, select **Regional (Recommended)**.
6. In the **IPv4 CIDR Block** text box, enter the CIDR in the **0.0.0.0/0** format. For example, `10.0.0.0/28`.
7. From the **Route Table Compartment** drop-down list, select the created public route table.
8. From the **Subnet Access** field, select **Public Subnet**.
9. Under the **DNS Resolution** field, select the **Use DNS hostnames in this Subnet** check box.
   The **DNS Label** and **DNS Domain Name** fields are auto-populated.
10. Click **Create Subnet**.

To create a private subnet:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select **Subnets**, and then click the **Create Subnet** button.
   The **Create Subnet** page opens.
3. In the **Name** text box, enter the private subnet name. For example, `Private subnet`.
4. In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
5. From the **Subnet Type** field, select **Regional (Recommended)**.
6. In the **IPv4 CIDR Block** text box, enter the CIDR in the **0.0.0.0/0** format. For example, `10.0.0.0/24`.
7. From the **Route Table Compartment** drop-down list, select the created private route table.
8. From the **Subnet Access** field, select **Private Subnet**.
9. Under the **DNS Resolution** field, select the **Use DNS hostnames in this Subnet** check box.
   The **DNS Label** and **DNS Domain Name** fields are auto-populated.
10. Click **Create Subnet**.

To create a private LB subnet:

1. Log in to OCI.

2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, then from the **Resources** pane, select **Subnets**, and then click the **Create Subnet** button.
   The **Create Subnet** page opens.
3. In the **Name** text box, enter the private subnet name. For example, `Private LB subnet`.
4. In the **Create In Compartment** text box, enter the compartment name. For example, `demo`.
5. From the **Subnet Type** field, select **Regional (Recommended)**.
6. In the **IPv4 CIDR Block** text box, enter the CIDR in the **0.0.0.0/0** format. For example, `10.0.2.0/28`.
7. From the **Route Table Compartment** drop-down list, select the created private route table.
8. From the **Subnet Access** field, select **Private Subnet**.
9. Under the **DNS Resolution** field, select the **Use DNS hostnames in this Subnet** check box.
   The **DNS Label** and **DNS Domain Name** fields are auto-populated.
10. Click **Create Subnet**.

Modifying the Security List in the VPN

You can add, modify, or terminate the ingress and egress rules of your VCN.

To modify the Security List of your VCN:

1. Log in to OCI.
2. From the **Navigation** pane, select **Networking**, then select **Virtual cloud networks**, then select your VCN, and then from the **Resources** pane, select **Security List Details**.
   The **Default Security List** page of your VCN opens.
3. To add an Ingress rule:
   - From the **Resources** pane, select **Ingress Rules**.
   - Click **Add Ingress Rules**.
   - Provide the required details and then click **Save**.
4. To add an Egress rule:
   - From the **Resources** pane, select **Egress Rules**.
   - Click **Add Egress Rules**.
   - Provide the required details and then click **Save**.
5. To modify the security list details, select any one rule from the ingress or egress rules list, and then click **Edit**.
6. Modify the required details, and then click **Save**.

Deploying the Public Bastion

This section provides detailed instructions about deploying the public bastion and verifying it.

To deploy the public bastion:

1. Log in to OCI.

2. From the **Navigation** pane, select **Compute**, then select **Instances**, and then click **Create instance**.
   The **Create compute instance** page opens, provide the following details:
   - Name. For example, `Public-bastion`.
   - Image and Shape
   - VCN
   - Boot volume
   - SSH public keys

   > **Note:**
   >
   > You must create your own public keys.

3. Click **Create**. The public bastion is created and a public IP is generated.

You can verify the public bastion access using the SSH key.

> **Note:**
>
> Ensure that you are connected to the OCNA VPN before starting the below procedure.

To verify the public bastion:

1. Open Git Bash on your local machine.
2. Run the following command:

   ```
   ssh -i /path/to/private-key opc@<bastion-public-ip>
   ```

   Replace `/path/to/private-key` with the path to your SSH private key.

   Replace `<bastion-public-ip>` with the bastion's public IP address.

If correct values are provided in the above command, you will get the access to the public bastion.

Deploying the Oracle Base Database Service

This section provides detailed instructions about provisioning the Oracle Base Database Service, integrating it with the VCN, configuring automated backups, and monitoring for the database to ensure data integrity and performance.

To deploy the Oracle Base Database Service:

1. Log in to OCI.
2. From the **Navigation** pane, select **Oracle Database**, then select **Oracle Base Database**, then select **DB Systems**, and then click the **Create DB System** button.
   A **Create DB System** page opens.
3. Click DB system information, and provide the following details:

a. From the **Select a compartment** drop-down list, select the appropriate compartment.
b. In the **Name your DB System** text box, enter the desired name. For example, `dbe-database`.
c. From the **Select an availability domain** list, select an appropriate domain.
d. In the **Configure shape** pane, attach the shape details.
e. From the **Configure storage** pane, select the appropriate storage management software type.
f. In the **Configure the DB System** field, enter the host name.
g. Click **Save**.

4. Click **Database information**, and configure the administrator credentials as follows:
   a. In the **Database name** text box, enter the desired name. For example, `DB0822`.
   b. Click the **Change database image** button in the **Database image** field, browse the appropriate image, and upload the image.
   c. In the **Create administrator credentials** pane, provide the following details:
      - In the **Password** text box, enter the desired password.

      > **Note:**
      > The **Username** field will be auto-populated and is a read-only field.

      - In the **Confirm Password** text box, enter the password again.
      - Select the **Use the administrator password for the TDE wallet** check box.
   d. From the **Configure database backups** pane, select the **Enable automatic backups** check box.
5. Click **Create DB System**. The Oracle Base Database is created.
   The **Database system information** page opens.
6. From the **Resources** pane, click **Nodes**, and save the Private IP address.

Verifying and Accessing the Database

To verify and access the Oracle Base Database:

1. Open the Command Line Interface (CLI).
2. Log in to public bastion using SSH. See Deploying the Public Bastion for more information.
3. Run the following command inside the public bastion:

   ```
   $ ssh opc@Private_IP_Address
   ```

   Replace `Private_IP_Address` with the Oracle Database private IP address generated after it was created.
4. Run the following commands to verify the access to Oracle Database:

   ```
   $ sudo su -oracle
   sqlplus / as sysdba
   ```

Creating a Pluggable Database within the Oracle Base Database

To create a pluggable database (PDB):

1. Log in to OCI.
2. From the **Navigation** pane, select **Oracle Database**, then select **Oracle Base Database Service**, then from the **Compartment** drop-down list, select your compartment, and then select your **Oracle Database System**.
3. From the **Resources** pane, click **Databases**, and then select the existing database from the table.
4. From the **Resources** pane, click **Pluggable Databases**, and then click the **Create pluggable database** button.
5. Enter the values for the following fields:
   - **PDB name**, select the **Unlock the PDB admin account** check box.
   - **PDB admin password**

     > **Note:**
     >
     > You must always refer to the OCI vault for the password.

   - **Confirm PDB admin password**
   - **TDE wallet password of database**
6. Click **Create pluggable database**.

Deploying an Oracle Kubernetes Engine Environment

This section provides detailed instructions about creating a Kubernetes cluster using Oracle Kubernetes Engine (OKE), configuring the node pools, setting up the Kubernetes access, and verifying the access.

To deploy the Kubernetes cluster using OKE:

1. Log in to OCI.
2. From the **Navigation** pane, select **Developer Services**, and then select **Kubernetes Cluster**.
3. Click the **Create cluster** button.
   A **Create cluster (custom)** page opens. Provide the following details:
   a. In the **Name** text box, enter the desired cluster name. For example, `dbe-cluster`.
   b. In the **Compartment** text box, enter the name of the container. For example, `demo`.
   c. In the **Kubernetes version** text box, enter the **Kubernetes** version. For example, `v1.30.1`.
   d. From the **Network type** field, select your VCN.
   e. In the **VCN in Compartment** text box, enter your VCN name. For example, `dbe-vcn`.
   f. From the **Kubernetes service LB subnets in compartment** field, select the LB subnet created in your VCN.

g. From the **Kubernetes API endpoint subnet in compartment** field, select the private subnet created in your VCN.
The node pool details will be auto-populated in the **Node pools** section.

h. Click the **Review** link from the left pane to review the details.

4. Click **Create cluster**.
The Kubernetes cluster is created.

Verifying and Accessing the Cluster

Prerequisites before verifying the access to the cluster:

- Download OCI CLI version 2.24.0 or later. See Installing the CLI for more information.
- Configure CLI on your local machine. See Configuring the CLI for more information.
- Navigate to the cluster details in OCI, click **Access Cluster**, and then select the **Local Access** option.

To verify the access to the cluster:

1. Open OCI CLI on your local machine.
2. Log in to the public bastion using SSH.
3. Navigate to the cluster details in OCI, then click **Access Cluster**, and then copy the commands from the **Access Your Cluster** page.
4. Run the commands in OCI CLI in the sequence mentioned in the **Access Your Cluster** page to download and configure the `kubectl` file.
5. Run the following commands in OCI CLI to verify the cluster access:

```
$kubectl get ns
$kubectl get nodes
```

Deploying a Kubernetes Cluster using Oracle Cloud Native Environment

This section provides detailed instructions about deploying a Kubernetes cluster using Oracle Cloud Native Environment (OCNE).

To deploy a Kubernetes cluster using OCNE:

1. Log in to OCI.
2. From the **Navigation** pane, select **Compute**, and then select **Instances**.
A **Create Instance** page opens.
3. Click **Create Instance**.
4. From the **Compartment** drop-down list, select the appropriate compartment for your instance. For example, **operations-staging**.
5. In the **Instance Name** field, enter the display name of your instance.
6. Under the **Image and Shape** field, provide the following details:
   a. In the **Image** field, browse and upload an appropriate image for your instance. For example, Oracle Linux, Ubuntu, or a custom image.
   b. In the **Shape** field, select an appropriate shape. For example, VM.Standard.E2.1.Micro.
7. Under the **Configure Networking** field, provide the following details:

a. From the **VCN** drop-down list, select an existing VCN.

> **Note:**
>
> You can also create a new VCN by clicking the **Create VCN** button.

b. From the **Subnet** drop-down list, select the subnet within the VCN selected in the above step.

8. Select either **Public IP address** or **Private IP address** option for the instance.
9. Click **Create Instance**. The instance is created.
10. After the instance is created, see Oracle Cloud Native Environment Quick Start Guide and follow the instructions in the sequence mentioned in this guide to complete the deployment of the Kubernetes cluster using OCNE.

Deploying a Cloud Native Computing Foundation Environment

This section provides detailed instructions about deploying a Cloud Native Computing Foundation (CNCF) environment, which involves creating a control plane and creating a worker node.

Prerequisites for Deploying a CNCF Environment

- Ensure that the system has at least four CPUs and 64 GB of memory.
- You must be on an Oracle Linux 8 operating system.
- You must disable the swap option on all nodes.
- You must have a good internet connection to download the packages.

Creating a Control Plane

To create a control plane:

1. Log in to the virtual machine (VM) control plane and run the following script on Oracle Linux 8 to configure a Kubernetes control plane:

```
#!/bin/bash
################# Adapted for Oracle Linux 8 ################
# This script configures a Kubernetes control plane on Oracle Linux 8.

# Set Kubernetes and crictl versions
export VER="v1.31.1"
export K8S_VER="1.31"
export K8S_PKG="v1.31"

# Check if the script has been run before
FILE=/k8scp_run
if [ -f "$FILE" ]; then
    echo "WARNING!"
    echo "$FILE exists. Script has already been run on control plane."
    echo
    exit 1
else
    echo "$FILE does not exist. Running script."
```

```
fi

# Prevent script from running twice
sudo touch /k8scp_run

# Update the system
sudo dnf update -y

# Install necessary software
sudo dnf install -y curl vim git wget gnupg2 socat \
    yum-utils device-mapper-persistent-data lvm2

# Add the Kubernetes repo
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/${K8S_PKG}/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/${K8S_PKG}/rpm/repodata/
repomd.xml.key
#exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

# Install Kubernetes components
sudo dnf install -y kubelet kubeadm kubectl
sudo dnf versionlock add kubelet kubeadm kubectl

# Ensure Kubelet is running
sudo systemctl enable --now kubelet

# Disable swap
sudo swapoff -a
sudo sed -i '/swap/d' /etc/fstab

# Load necessary kernel modules
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter

# Update networking settings
cat <<EOF | sudo tee /etc/sysctl.d/kubernetes.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system

# Install containerd
sudo dnf config-manager --add-repo https://download.docker.com/linux/
centos/docker-ce.repo
sudo dnf install -y containerd.io
```

```
# Configure containerd
sudo mkdir -p /etc/containerd
containerd config default | sudo tee /etc/containerd/config.toml
sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/
containerd/config.toml
sudo systemctl restart containerd
sudo systemctl enable containerd

# Install and configure crictl
wget https://github.com/kubernetes-sigs/cri-tools/releases/download/${VER}/
crictl-${VER}-linux-amd64.tar.gz
tar zxvf crictl-${VER}-linux-amd64.tar.gz
sudo mv crictl /usr/local/bin
sudo crictl config --set \
runtime-endpoint=unix:///run/containerd/containerd.sock \
--set image-endpoint=unix:///run/containerd/containerd.sock

# Initialize the Kubernetes cluster
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 | sudo tee /var/log/
kubeinit.log

# Configure kubectl for the current user
mkdir -p $HOME/.kube
sudo cp -f /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

# Install Cilium CLI
export CILIUM_CLI_VERSION=$(curl -s https://raw.githubusercontent.com/
cilium/cilium-cli/master/stable.txt)
export CLI_ARCH=amd64
if [ "$(uname -m)" = "aarch64" ]; then CLI_ARCH=arm64; fi
curl -L --fail --remote-name-all https://github.com/cilium/cilium-cli/
releases/download/${CILIUM_CLI_VERSION}/cilium-linux-$
{CLI_ARCH}.tar.gz{,.sha256sum}
sha256sum --check cilium-linux-${CLI_ARCH}.tar.gz.sha256sum
sudo tar xzvfC cilium-linux-${CLI_ARCH}.tar.gz /usr/local/bin
rm cilium-linux-${CLI_ARCH}.tar.gz{,.sha256sum}
cilium install --set ipam.mode=cluster-pool --set
ipam.operator.clusterPoolIPv4PodCIDRList=10.244.0.0/16 --set
ipam.operator.clusterPoolIPv4MaskSize=24

#Disable firewall
sudo systemctl stop firewalld
sudo systemctl disable firewalld
sudo systemctl mask --now firewalld

# Install Helm
wget https://get.helm.sh/helm-v3.16.2-linux-amd64.tar.gz
tar -xf helm-v3.16.2-linux-amd64.tar.gz
sudo mv linux-amd64/helm /usr/local/bin/

# Output the state of the cluster
kubectl get node

# release storage
sudo /usr/libexec/oci-growfs -y
```

```
echo "Setup complete. Proceed to the next step."
```

2. Check if the status of the control plane node is **Ready**. The following is a sample command and its output:

```
[opc@vanillak8sre-cp ~]$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
vanillak8sre-cp    Ready    control-plane   78m   v1.31.4
```

3. Run the following command to get the kubejoin details:

```
kubeadm token create --print-join-command
```

Creating a Worker Node

To create a worker node:

1. Log in to the VM worker node and run the following script on Oracle Linux 8 to set up a Kubernetes worker node:

> **Note:**
>
> You can also add multiple worker nodes based on your environment's requirement.

```
#!/bin/bash
################## Adapted for Oracle Linux 8 ################
# This script sets up a Kubernetes worker node on Oracle Linux 8.


export VER="v1.31.1"
export K8S_VER="1.31"
export K8S_PKG="v1.31"

# Check if the script has been run before. Exit if it has.
FILE=/k8scp_run
if [ -f "$FILE" ]; then
    echo "WARNING!"
    echo "$FILE exists. Script has already been run."
    echo "Do not run on the control plane. Run on a worker node."
    echo
    exit 1
else
    echo "$FILE does not exist. Running script."
fi

# Prevent script from being run multiple times
sudo touch /k8scp_run

# Update the system
sudo dnf update -y
```

```
# Install necessary software
sudo dnf install -y curl vim git wget gnupg2 socat \
    yum-utils device-mapper-persistent-data lvm2

# Add the Kubernetes repo
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/${K8S_PKG}/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/${K8S_PKG}/rpm/repodata/
repomd.xml.key
#exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

# Install Kubernetes components and lock their versions
sudo dnf install -y kubelet kubeadm kubectl
sudo dnf versionlock add kubelet kubeadm kubectl

# Ensure Kubelet is running
sudo systemctl enable --now kubelet

# Disable swap
sudo swapoff -a
sudo sed -i '/swap/d' /etc/fstab

# Load necessary kernel modules
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter

# Update networking settings
cat <<EOF | sudo tee /etc/sysctl.d/kubernetes.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sudo sysctl --system

# Install containerd
sudo dnf config-manager --add-repo https://download.docker.com/linux/
centos/docker-ce.repo
sudo dnf install -y containerd.io

# Configure containerd
sudo mkdir -p /etc/containerd
containerd config default | sudo tee /etc/containerd/config.toml
sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/
containerd/config.toml
sudo systemctl restart containerd
sudo systemctl enable containerd
```

```
# Install and configure crictl
wget https://github.com/kubernetes-sigs/cri-tools/releases/download/${VER}/
crictl-${VER}-linux-amd64.tar.gz
tar zxvf crictl-${VER}-linux-amd64.tar.gz
sudo mv crictl /usr/local/bin
sudo crictl config --set \
runtime-endpoint=unix:///run/containerd/containerd.sock \
--set image-endpoint=unix:///run/containerd/containerd.sock

#Disable firewall
sudo systemctl stop firewalld
sudo systemctl disable firewalld
sudo systemctl mask --now firewalld


# release storage
sudo /usr/libexec/oci-growfs -y


# Instructions for joining the worker node to the cluster
sleep 3
echo
echo
echo '**************************'
echo
echo "Continue to the next step"
echo
echo "Use sudo and copy the kubeadm join command from"
echo "the control plane node."
echo
echo '**************************'
echo
echo
```

2. Run the kubejoin command and check the node status. The following is a sample kubejoin command and its status:

```
[opc@abc8sre-cp ~]$ kubeadm join 10.0.5.248:6443 --token
ukztj1.sgd165r61xknz2qs --discovery-token-ca-cert-hash
sha256:7ab8dff2197aaa6c0701c35132006ab0934e95e8e259611b17d4710285dbfbc9
[opc@abc8sre-cp ~]$ kubectl get nodes
NAME              STATUS   ROLES           AGE   VERSION
abc8sre-cp        Ready    control-plane   78m   v1.31.4
abc8sre-wn1       Ready    <none>          67m   v1.31.4
```

Deploying the Solution Components

This section provides detailed instructions for deploying Digital Business Experience solution components.

To deploy the Digital Business Experience solution:

1. Request for an environment for the solution. Contact Oracle Support for assistance.

2. Prepare the cluster for the solution. See Preparing the Cluster for information about deploying the key cloud infrastructure elements.
3. Deploy Launch Cloud Service and CX Industries Framework. See *Oracle Communications Launch Implementation Guide* for detailed instructions about deploying Launch and CX Industries Framework applications.
4. Deploy BRM. See *Oracle Communications Billing and Revenue Management Cloud Native Deployment Guide* for detailed instructions about deploying BRM.
5. Deploy PDC. See *Oracle Communications Billing and Revenue Management PDC Installation Guide* for detailed instructions about deploying PDC.
6. Deploy ECE. See *Oracle Communications Billing and Revenue Management ECE Installation Guide* for detailed instructions about deploying ECE.
7. Deploy OCOMC. See *Oracle Communications Offline Mediation Controller Cloud Native Installation and Administration Guide* for detailed instructions about deploying OCOMC.
8. Deploy OAP. See Installing and Configuring Oracle Analytics Server for detailed instructions about deploying OAP.
9. Deploy SCD. See *Oracle Communications Service Catalog and Design Studio Installation Guide* for detailed instructions about deploying SCD.
10. Deploy OSM. See *Oracle Communications Order and Service Management Cloud Native Deployment Guide* for detailed instructions about deploying OSM.
    - Deploy Order to Activate (O2A). See *Order and Service Management Cartridges for Application Integration Architecture Cloud Native Deployment Guide* for detailed instructions about deploying O2A.
11. Deploy Siebel CRM. See Developing and Deploying Siebel CRM for detailed instructions about deploying Siebel CRM.
12. Deploy ODI. See Oracle Fusion Middleware Installing and Configuring Oracle Data Integrator Guide for detailed instructions about deploying ODI.
13. Deploy AIA. See *Oracle Communications Application Integration Architecture Cloud Native Deployment Guide* for detailed instructions about deploying AIA.

Validating the Solution Components

This section provides information about various validations required for all solution components. These validations are necessary for smooth and unrestricted functioning of the solution.

To validate the deployment of the solution:

1. Verify the password expiration of the cloud native applications: See Verifying the Password Expiration for detailed procedure for verifying the password expiration of the cloud native applications.
2. Validate the public certificates: See Validating the Public Certificates for detailed procedure about validating the various public certificates.
3. Validate the Launch and CXIF deployment: See Validating the Connection in *Oracle Communications Launch Cloud Service Integration Guide* and follow the procedure for validating the Launch and CXIF connection.
4. Validate the SCD deployment: See Validating the Solution Designer Instance in *Oracle Communications Service Catalog and Design Solution Designer Installation Guide* and follow the procedure for validating the SCD deployment.

5. Validate the AIA deployment: See Validating the AIA Cloud Native Deployment in *Oracle Communications Application Integration Architecture Cloud Native Deployment Guide* and follow the procedure for validating the AIA deployment.

> **Note:**
>
> If any of the above validation fails, stop the validation process, and contact Oracle Support.

Verifying the Password Expiration

To verify the password expiration of the Digital Business Experience cloud native applications, from the Oracle Database (DB):

1. Run the following command to access the sql client:
   ```
   sqlplus / as sysdba
   ```
   The sample output is as follows:

   ```
   [opc@vanillak8s-bastion .ssh]$ ssh -i vanillaK8sdb_private.key
   vanillak8sdb.sub05131336501.ops.oraclevcn.com
   [opc@vanillak8sdb ~]$ sudo su - oracle
   [oracle@vanillak8sdb]$ sqlplus / as sysdba
   SQL*Plus: Release 19.0.0.0.0 - Production on Wed Dec 11 11:46:04 2024
   Version 19.24.0.0.0

   Copyright (c) 1982, 2024, Oracle.  All rights reserved.


   Connected to:
   Oracle Database 19c EE High Perf Release 19.0.0.0.0 - Production
   Version 19.24.0.0.0

   SQL>
   ```

2. Run the following command to access the PDB:
   ```
   alter session set container=BRMCN15;
   ```
3. Run the following command to check the expiry of a DB user:
   ```
   select username, account_status, EXPIRY_DATE from dba_users;
   ```
4. Run the following command to set a DB user to no expiration:

   ```
   SQL> alter profile DEFAULT limit PASSWORD_REUSE_TIME unlimited;

   SQL> alter profile DEFAULT limit PASSWORD_LIFE_TIME  unlimited;
   ```

Validating the Public Certificates

This section provides information about validating various public certificates for the Digital Business Experience environment.

Validating the CXIF Certificate

To validate the CXIF certificate, run the following command:

```
$ openssl s_client -showcerts -connect rododsiebel.jetpen.com:31000
```

The following is a sample output:

```
$ openssl s_client -showcerts -connect rododsiebel.jetpen.com:31000
CONNECTED(00000003)
depth=2 C = US, O = Internet Security Research Group, CN = ISRG Root X1
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = R11
verify return:1
depth=0 CN = rododsiebel.jetpen.com
verify return:1
---
```

Validating the Siebel and PDC Certificates

Before validating the Siebel and PDC certificates, you must identify the Siebel and PDCRSM API endpoint URLs. For example, Siebel API Endpoint URL for Variant 1 is: `https://rododsiebel.jetpen.com:32401`

To validate the Siebel and PDC certificates, run the following command on a jump host with openssl:

```
$ echo -n Q | openssl s_client -connect rododsiebel.jetpen.com:32401 | openssl
x509 -noout -dates ---
```

The following is a sample output:

```
[mimatysk@test-overlay-t83z-hc ~]$ echo -n Q | openssl s_client -connect
rododsiebel.jetpen.com:32401 | openssl x509 -noout -dates ---
depth=2 C = US, O = Internet Security Research Group, CN = ISRG Root X1
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = R11
verify return:1
depth=0 CN = rododsiebel.jetpen.com
verify return:1
DONE
notBefore=Oct 24 21:39:58 2024 GMT
notAfter=Jan 22 21:39:57 2025 GMT
```

4

Performing Post-Deployment Tasks

This chapter describes the tasks you perform after deploying the Digital Business Experience solution.

Integrating Launch and CXIF

This section describes the tasks you perform to integrate Launch and CXIF applications with the Digital Business Experience solution.

See Integrate Launch with Digital Business Experience in *Oracle Communications Launch Cloud Service Implementation Guide* for information about Launch integration with Digital Business Experience solution and other applications.

Prerequisites

Before integrating Launch and CXIF applications, you must perform the following:

- Request for Launch environment. Contact Oracle Support for assistance.
- Request for CXIF environment. Contact Oracle Support for assistance.
- Request for Launch and CXIF integration. Contact Oracle Support for assistance.

Integrating Launch and CXIF with Siebel CRM and PDC

After the prerequisites are completed, perform the following:

- Integrate Launch and CXIF with Siebel CRM. See Launch Cloud Service Siebel CRM Integration in *Oracle Communications Launch Cloud Service Integration Guide* for more information.
- Integrate Launch and CXIF with PDC (BRM). See Launch Cloud Service PDC (BRM) Integration in *Oracle Communications Launch Cloud Service Integration Guide* for more information.

Integrating Siebel CRM on Containers with AIA Cloud Native

This section provides instructions for integrating Siebel CRM on Containers with AIA cloud native.

To integrate Siebel CRM on Containers with AIA cloud native:

1. Get the following JAR files:
    - From Siebel containers, get **siebel.jar** and **SiebelJI_enu.jar**.
    - From the WebLogic container, get **wlthint3client.jar**.
2. Log in to Siebel eCommunication Web UI as SADMIN user and update the JAVA64 profile parameters to include the three JAR files:

> **Note:**
>
> Ensure that the path **/sfs/aiacn/jms** is a persistent store so that the files are retained after the pod restarts.

```
/sfs/aiacn/jms:/sfs/aiacn/jms/Siebel.jar:/sfs/aiacn/jms/
SiebelJI_enu.jar:/sfs/aiacn/jms/wlthint3client.jar:.
```

3. Relocate the JAR files:
   a. Connect to the Siebel SES pod.
   b. Create a folder with the same name as what is listed in step 2 (/sfs/aiacn/jms), where /sfs is a shared persistent folder, and then copy the 3 JAR files into the folder.
4. In the same folder, create the **jndi.properties** file and copy the following text into it:

> **Note:**
>
> - Ensure that the AIACN t3 channel URL and user name and password are set with the correct values.
> - In the below example, Siebel and AIA are in the same cluster and different namespace.

```
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
java.naming.provider.url=t3://
soa_cluster_service_name.soa_namespace.svc.cluster.local:soa_service_cluste
r_port
java.naming.security.principal=soa_console_username
java.naming.security.credentials=soa_console_password
```

5. Copy the three JAR files into the Apache TOMCAT **/siebel/mde/applicationcontainer/lib** folder.
6. Restart Apache TOMCAT server inside Siebel SES pod.
7. Configure Siebel Web Service in Siebel DB.
   a. Create the "update_siebel_ws.sql" SQL script with the following content:

```
UPDATE S_WS_PORT SET PORT_ADDRESS='jms://jms/aia/
AIA_SALESORDERJMSQUEUE@jms/aia/COMMS_SUBMITORDER_CONSUMER',
PORT_TRANSPORT='JMS' WHERE NAME='SWISubmitOrderPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='jms://jms/aia/
AIA_SALESORDERJMSQUEUE@jms/aia/COMMS_SUBMITORDER_CONSUMER',
PORT_TRANSPORT='JMS' WHERE NAME='SWISubmitOrder_o2cPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='jms://jms/aia/
AIA_SALESORDERJMSQUEUE@jms/aia/COMMS_SUBMITORDER_CONSUMER',
PORT_TRANSPORT='JMS' WHERE NAME='SWISubmitQuote_o2cPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='jms://jms/aia/
AIA_SPECIALRATINGJMSQ@jms/aia/COMMS_SPECIALRATINGLIST_CONSUMER',
PORT_TRANSPORT='JMS' WHERE NAME='SWISpecialRatingListPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='jms://jms/aia/
```

```
AIA_CMUREQADJIOJMSQUEUE@jms/aia/COMMS_ADJUSTMENT_CONSUMER',
PORT_TRANSPORT='JMS' WHERE NAME='SWICreateAdjustmentPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/AccountBalanceSiebelCommsReqABCS/
AccountBalanceSiebelCommsReqABCS_ep' WHERE
NAME='_soap_AccountBalanceSiebelCommsReqABCS_AccountBalanceSiebelCommsRe
qABCS';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/AdjustmentSiebelCommsReqABCS/
AdjustmentSiebelCommsReqABCS_ep' WHERE
NAME='AdjustmentSiebelCommsReqABCSPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/InvoiceSiebelCommsReqABCS/
InvoiceSiebelCommsReqABCS_ep' WHERE
NAME='_soap_InvoiceSiebelCommsReqABCS_InvoiceSiebelCommsReqABCS';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/PaymentSiebelCommsReqABCS/
PaymentSiebelCommsReqABCS_ep' WHERE
NAME='PaymentSiebelCommsReqABCSPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/UnbilledUsageSiebelCommsReqABCS/
UnbilledUsageSiebelCommsReqABCS_ep' WHERE
NAME='_soap_UnbilledUsageSiebelCommsReqABCS_UnbilledUsageSiebelCommsReqA
BCS';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/SyncCustomerSiebelEventAggregator/Client'
WHERE NAME='SyncCustomerSiebelEventAggregatorPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='http://
aiacn_clusterService_name.aiacn_kubernetes_name:aiacn_clusterService_por
t/soa-infra/services/default/UpdateCreditAlertSiebelCommsReqABCSImpl/
UpdateCreditAlertSiebelCommsReqABCSImpl' WHERE
NAME='UpdateCreditAlertSiebelCommsReqABCSImplServicePort';
SET ESCAPE ON;
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWIOrderUpsert';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWI Product Attribute Import';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWI Product Class Import';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWIProductImport';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
```

```
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWIPromotionImport';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/eai/enu/start.swe?
SWEExtSource=SecureWebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='SWIUpsertQuote';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/app/eai/enu?
SWEExtSource=WebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='COMMSCancelOrderPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/app/eai/enu?
SWEExtSource=WebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='COMMSCustomServicesPort';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/app/eai/enu?
SWEExtSource=WebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='COMMSOrderUpsert';
UPDATE S_WS_PORT SET PORT_ADDRESS='https://Siebel_hostname:Siebel_port/
siebel/app/eai/enu?
SWEExtSource=WebService\&SWEExtCmd=Execute\&WSSOAP=1' WHERE
NAME='COMMSSubmitBillingOrder';
commit;
quit;
```

b. Connect to Siebel DB and run the SQL script with DB user name and the corresponding password.

8. Configure the Siebel repository in Siebel DB:

a. Create the "update_siebel_repository.sql" SQL script with the following contents:

```
UPDATE S_SYS_PREF SET VAL='TRUE' WHERE SYS_PREF_CD='Enable AIA Comms';
UPDATE S_SYS_PREF SET VAL='TRUE' WHERE SYS_PREF_CD='Enable AIA MDM';
UPDATE S_SYS_PREF SET VAL='TRUE' WHERE SYS_PREF_CD='Enable AIA Testing';
UPDATE S_SYS_PREF SET VAL='FALSE' WHERE SYS_PREF_CD='Enable AIA
Utility';
UPDATE S_SYS_PREF SET VAL='No' WHERE SYS_PREF_CD='Enable Promotion
Group';
UPDATE S_SYS_PREF SET VAL='/siebel/mde/siebsrvr/temp/OrderBackup/'
WHERE SYS_PREF_CD='AIA Order Backup Path';
UPDATE S_SYS_PREF SET VAL='Yes' WHERE SYS_PREF_CD='Enable Promotion
Group';
UPDATE S_SYS_PREF SET VAL='Y' WHERE SYS_PREF_CD='Promotion Group
Compatibility';
commit;
quit;
```

b. Connect to Siebel DB and run the SQL script with DB user name and the corresponding password.

9. Configure EAI File Transfer Folder:

a. Connect to the SES pod of the Siebel CRM on Containers instance.

b. Run the following two Siebel commands respectively and set "EAIFileTransportFolders" with the created "OrderBackup" sub-folder's full path as follows:

> **Note:**
>
> Ensure that the `/siebel/mde/siebsrvr/temp/OrderBackup` folder is created in the Siebel SES container. If not, then create the folder and proceed with below commands.

```
[aiacn_pod-0:/siebel/mde]#srvrmgr /g cgw-aiacn-0.ses-aiacn.siebel-
cn.svc.cluster.local:2320 /e aiacn /u username /p password /c "change
ent param EAIFileTransportFolders=/siebel/mde/siebsrvr/temp/OrderBackup"
```

```
srvrmgr> change ent param EAIFileTransportFolders=/siebel/mde/siebsrvr/
temp/OrderBackup
```

```
srvrmgr> change ent param EAIFileTransportFolders=/siebel/mde/siebsrvr/
temp/OrderBackup for server aiacn_pod-0
```

c. Restart the SES service using the kubectl command:

```
kubectl -n siebel_namespace, delete pod
Siebel_Enterprise_Server_pod_name-0
```

d. After the pod is recreated and you have verified that it is running, follow steps 5 and 6.

10. (Optional) Import products into Siebel CRM on Containers by using the Siebel eCommunication application. Refer to Siebel CRM documentation for instructions.
11. Import Siebel CRM on Containers SSL/TLS security certificates and configure AIA cloud native with the certificates. See Installing SSL Certificates for more details.
    a. Validate that keystore custom identity and custom trust are created successfully. To do this, log in to the Enterprise Manager Console for AIA and navigate to the Keystore section.
    b. Validate that Siebel trust certificate is available in the custom trust keystore in the Keystore section.
    c. Log in to the Weblogic Console of AIA cloud native. For each managed server, in the Keystore section, ensure the following:
       - **kss://system**/*custom_identity_keystorename* is displayed for Custom Identity Store.
       - **kss://system**/*custom_trust_keystorename* is displayed for Custom Trust Store.
    d. Validate **/u01/oracle/user_projects/domains/***domain_name***/bin/setDomainEnv.sh** with custom trust:
       i. Connect to any managed server pod.
       ii. Open the **/u01/oracle/user_projects/domains/***domain_name***/bin/setDomainEnv.sh** using the vi tool.

iii. Validate that **-Djavax.net.ssl.trustStore=kss://system/***custom_trust_keystorename* **-Djavax.net.ssl.trustStoreType=kss -Djavax.net.ssl.keyStorePassword**=*password* **-Djavax.net.ssl.trustStorePassword**=*password* is configured in EXTRA_JAVA_PROPERTIES.

12. Configure Siebel credentials in the Enterprise Console of AIA cloud native.
    a. Navigate to the Credentials section and edit **participatingapplications.siebel.server.eai.password** to specify the username and password.
    b. Repeat step a. for **participatingapplications.siebel.server.db.password** if you want to change the Siebel DB credentials.
    c. Restart the AIA cloud native services by using the domain-lifecycle scripts.
13. Configure AIA cloud native MetaData with Siebel Connection details, Business Unit ID, and PRICELIST.
    a. Get the latest **AIAConfigurationProperties.xml** and **PRICELIST.dvm** files:
        i. In the Enterprise Manager Console for AIA, navigate to the **MDS Configuration** section and export the zip file.
        ii. Copy the following files from the zip: **soa/configuration/default/AIAConfigurationProperties.xml** and **apps/AIAMetaData/dvm/PRICELIST.dvm**. For more information on MDS operations, refer to Managing the Metadata Repository in *Administering Oracle Fusion Middleware*.
    b. Update **Siebel connection** details:
        i. In the **AIAConfigurationProperties.xml** file, update all the values of the XML tag **SEBL_01.EndpointURI** with correct details if required.
    c. Update the **Business Unit ID** details:
        i. Log in to Siebel and get Siebel Business Unit ID by navigating to **Organizations**. Navigate to **About Record**, and copy the Row Number.
        ii. In the **AIAConfigurationProperties.xml** file, update all the values of the XML tag **Siebel.SEBL_01.BusinessUnit**, with the copied Default Organization Row Number.
    d. Update Pricelist details:
        i. **Pricelists**:
            i. Log in to Siebel and create or confirm pricelists as required on Siebel. Copy the Row Number of the required pricelist. For more information, refer to Siebel Price List in the *Application Services Interface Reference Guide*.
            ii. In the **PRICELIST.dvm** file, update or add the values for **SEBL_01 column** of **row** with the copied Row Number and with the other required values.

                For more information, refer to Working with the PRICELIST DVM in the *Order to Cash Implementation Guide* .
            iii. Repeat steps 1 and 2 for each pricelist.
        ii. **Default Pricelist**:
            i. In the **AIAConfigurationProperties.xml** file, update all the values of XML tag **Siebel.SEBL_01.PriceList.ID** with Row Number from Siebel for default pricelist.

> **Note:**
>
> Mention the default pricelist details in the **AIAConfigurationProperties.xml** file. In case of multi-pricelist configuration, do not mention the default pricelist section in the **PRICELIST.dvm** file.

e. Update AIA MDS with the updated **PRICELIST.dvm** and the **AIAConfigurationProperties.xml** files. For more information about updating files in AIA MDS, refer to Updating Files in AIA MDS in the *Application Integration Architecture Cloud Native Deployment Guide*.

f. Restart AIA cloud native using the domain-lifecycle scripts.

14. Enable the eai_enu application configuration using the Siebel Management Console. Refer to the Siebel Management Console documentation for instructions.

Integrating BRM Cloud Native with AIA Cloud Native

This section provides instructions for integrating BRM cloud native with AIA cloud native.

To integrate BRM cloud native with AIA cloud native:

1. Validate the BRM cloud native CM parameter by running the following command:

> **Note:**
>
> Ensure that the BRM CM service is configured with the dnsName of the cluster, so that the CM service can be connected using the dnsName in the cluster.

```
kubectl -n brmcn_namespace get deployment/cm -o yaml
```

A sample output is as follows:

```
- name: CM_DNS_NAME
  value: dns:<cm service>.<brm ns>.svc.cluster.local
```

2. Deploy the BRM JCA Adapter. See Deploying the BRM JCA Adapter in *Oracle Communications Application Integration Architecture Cloud Native Deployment Guide* for more information.
3. (Optional) Validate the connection between AIA cloud native and BRM cloud native by deploying the BRM JCA Adapter test client (Web application) and sending a test request with the test client Web UI. See Testing JCA Resource Adapter Configuration and BRM Connectivity in *Oracle Communications BRM JCA Resource Adapter Guide* for more information.
4. Enable notification and Product Sync in BRM cloud native.
   a. Ensure that the **fm_publish enable_publish** parameter is set to **1** in the CM pin.conf. You can set this by using the helm chart or by editing the Kubernetes config manager entry of cm `cm-pin-conf-config`.
   b. For EAI (**eai-java-server** container in the cm pod), ensure that payload for **Infranet.properties** is set to **payloadconfig_crm_sync.xml**. This payload contains

the required events (ProductInfoChange and DiscountInfo change) for generating the XML for EAI. Ensure that the DB is 0.0.0.0, which points to EAI. The DB entries mapping can be found in dm-oracle pin.conf. You can set this by using the helm chart or by editing the Kubernetes config manager entry of the EAI `eai-java-server-infranet-properties-config` and `eai-java-server-payload`.

> **Note:**
>
> Bounce the CM pod after the change to reflect the changes.
> `Publisher DB=\"0.0.0.0`

c. In the **dm-oracle-aq-event-map-config** config file, uncomment ALL in aq_event_map.

d. Ensure that notifications are enabled for the following:

```
/event/notification/price/products/modify
/event/notification/price/discounts/modify
/event/notification/price/sponsorships/modify
/event/customer/status
/event/notification/amt/AccountInfoChange
```

e. Run helm command from the BRM Helm charts path to publish Prod Sync data.

```
cd $BRM_CNTK/artifacts
$ helm upgrade --namespace <brm_namespace> brm-cn-apps helm-charts/ --
values profiles/deploy-oci.yaml --values profiles/pdc-publish.yaml
```

f. Restart the **cm**, **dm-oracle**, and **dm-ifw-sync** pods of the BRM cloud native instance.

Integrating OSM Cloud Native with AIA Cloud Native

This section provides instructions for integrating OSM cloud native with AIA cloud native.

To integrate OSM cloud native with AIA cloud native:

1. Create a t3 channel in the AIA cloud native instance:
   - For a single cluster scenario, where in AIA cloud native, Siebel CRM cloud native, BRM cloud native, and OSM cloud native are deployed in the same cluster, create a t3 channel in the WebLogic Admin Console for AIA cloud native.
     a. Log in to AIA cloud native Weblogic Console.
     b. Navigate to the Servers section in the Domain Structure pane and then select a managed server (for example, select **soa_server1**).
     c. Navigate to the Channels tab in Protocols and create a new channel (for example, T3Channel).
     d. Specify the following:
        – Listen Address as **soainfra-cluster-soa-cluster.*namespace*.svc.cluster.local**.
        – External Listen Port as ***soa-cluster-service-name.namespace*.svc.cluster.local**
     e. Ensure that ‘Tunneling Enabled’ is selected.

f. Repeat steps **b.** to **e.** for remaining managed servers.

- For multiple clusters, where in AIA cloud native and OSM cloud native are deployed in different clusters, do the following:

  a. Connect to the AIA cloud native cluster and create an ingressroute that includes route rules for all common names and the respective ports.

```
apiVersion: traefik.containo.us/v1alpha1
kind: IngressRoute
metadata:
  name: aia-ingress
  namespace: namespace
spec:
  entryPoints:
  - web
  routes:
  - kind: Rule
    match: Host(`soa.domain_name.namespace.aia.org`)
    services:
    - name: soa_cluster_service_name
      port: soa_ms_port
      sticky:
        cookie:
          httpOnly: true
  - kind: Rule
    match: Host(`t3.domain_name.namespace.aia.org`)
    services:
    - name: soa_cluster_service_name
      port: soa_cluster_service_port
      sticky:
        cookie:
          httpOnly: true
  - kind: Rule
    match: Host(`admin.domain_name.namespace.aia.org`)
    services:
    - name: soa_admin_server_service_name
      port: soa_admin_server_port
      sticky:
        cookie:
          httpOnly: true
```

  b. Apply the yaml file to create the ingressroute:

```
kubectl apply -f aia-ingress.yaml
```

  c. Edit the AIA cloud native domain configuration to specify the following settings for integrating with the OSM cloud native instance:

> **Note:**
>
> If the previous node is cordoned, deleted, or repaved, update the values for `hostAliases` and change the IP addresses to new IP addresses of a working node.

```
spec:
.........
.........
  serverPod:
.........
.........
    hostAliases:
    - hostnames:
      - t3.instance.project.osm.org
      - instance.project.osm.org
      - admin.instance.project.osm.org
      ip: osm_node_IP_address
    - hostnames:
      - soa.soainfra.soa_namespace.aia.org
      - t3.domain_name.namespace.aia.org
      - admin.domain_name.namespace.aia.org
      ip: soa_node_IP_address
```

d. Edit the OSM cloud native domain configuration to specify the same settings as described in step 1.b for integrating with the AIA cloud native instance.

> **Note:**
>
> Ensure that the AIA cloud native and OSM cloud native instance pods restart automatically after steps b and c. If they do not restart automatically, run the corresponding scripts to manually restart the AIA and OSM instances.

e. Create a t3 channel in the WebLogic Admin Console for AIA cloud native, ensuring that the External Listen Address is set as the hostname defined earlier. In addition, ensure that the **HTTP Enabled for This Protocol** option is selected.

2. Deploy the O2A cartridge into the OSM cloud native instance. See Creating Order-to-Activate Credentials and Accounts and Deploying the Sample Cartridge for instructions.
3. Set the AIA cloud native SAF t3 value which was created earlier, in the OSM project specification file as follows:
   - For single cluster, specify the following:

```
safConnectionConfig:
  - name: O2A_SAFImportedDestinations
    t3Url: t3://
soa_cluster_servicename.namespace.svc.cluster.local:soa_cluster_serviceport
    secretName: osm_project_instance__saf_credentials_aia_secret_name
```

> **Note:**
>
> *osm_project_instance__saf_credentials_aia_secret_name* is the secret you created while setting up and deploying OSM.

- For multiple clusters, specify the following:

```
safConnectionConfig:
  - name: O2A_SAFImportedDestinations
    t3Url: http://t3_hostname_in_ingressroute:t3_port_created
    secretName: osm_project_instance__saf_credentials_aia_secret_name
```

4. Restart the OSM instance to deploy the O2A cartridge with the AIA cloud native configuration. See Restarting the Instance in *Oracle Communications Order and Service Management Cloud Native Deployment Guide*.
5. Log in to the Weblogic Admin Console for OSM cloud native and verify that the SAF setting is configured with the expected t3 Url value.
6. Log in to the OSM Task Web client and verify that the O2A cartridge is deployed.
7. In WebLogic Admin Console for OSM cloud native, copy the t3 URL displayed for **T3ClustChannel** for single cluster or the value of **T3Channel (HTTP)** for multi-cluster channel.
8. In the WebLogic Admin Console for AIA cloud native, navigate to the JMS Modules page, for the **OSM** and **SOM** AIAJMSModules, set the URL test fields with the copied t3 URLs.

> **Note:**
>
> You must set a username and password for the WebLogic Admin Console for AIA cloud native.

9. Restart the AIA cloud native domain services by using the domain-lifecycle scripts.
10. In AIA WebLogic Console, navigate to the **Store and Forward Agents** page. In the **Remote Endpoints** tab for OSM_SAFAgent, ensure that for each Remote Endpoint, the t3 URL is displayed for single-cluster environment.
11. Log in to Oracle Enterprise Manager Fusion Middleware Control for AIA cloud native, and add the singleton property. See the AIA Installation Guide for instructions.
12. Configure the AIA queues to support JMS Priority. See the AIA Installation Guide for instructions.
13. Add the No Authentication security policy to the Product class service. See the AIA Installation Guide for instructions. Ensure that you select "QueryProductClassAndAttributesSCECommsReqABCSImpl" in the Service and References region.
14. Ensure that the JAVA_OPTIONS parameter **`-Dweblogic.rjvm.allowUnknownHost=true`** is added into AIA cloud native domain setting.
15. In Oracle Enterprise Manager Fusion Middleware Control, for SOA Infrastructure, set the **Callback Server URL** and **Server URL** common properties with the SOA infrastructure URL.

Integrating ODI with AIA Cloud Native

This section describes tasks to be performed to integrate ODI with AIA cloud native.

Prerequisites and Assumptions

The following are the prerequisites:

- The Master Repository and the Work Repository are created in Oracle Data Integrator (ODI).
- ODI is installed with template - J2EE Enterprise Agent and has the **encode.sh** script in **$ODI_HOME/bin** directory.
- ODI agent is available and agent wsdl is reachable inside the AIA cloud native installation pods.
- Verify Collections installation. Within the admin pod, verify that the following are present in the DVM path **/u01/shared/aia-comms/commsOracleHome/comms_home/source/soainfra/apps/AIAMetaData/dvm**:
  - CURRENCY_CODE.dvm
  - COLLECTION_STATUS.dvm
  - COLLECTION_ACTIONNAME.dvm
  - COLLECTION_PRIORITY.dvm
  - COLLECTION_SUBSTATUS.dvm
- Confirm the URL for composite definition for "SyncCollectionHeaderInfoBRMCommsReqImpl" composite is correct.
- Mount AIA PV on the machine where ODI is installed. Mount the PV at location "/u01/shared/".
- You must configure a unified collections action view and a unified collections scenario view in single-schema or multischema environment to be queried by Oracle AIA. See Configuring Views for Oracle Application Infrastructure Architecture for detailed instructions about configuring views for Oracle AIA.

Integrating ODI with AIA

To integrate ODI with AIA cloud native:

1. Update the mailing details in the **/u01/shared/aia-comms/commsOracleHome/comms_home/source/soainfra/apps/config/AIAConfigurationProperties.xml** file as required. Update the following properties for the module **CollectionsParameters** file:
   - **FromMailAddress**
   - **ToMailAddress**
   - **MailServer**
   - **MailSMTPPort**
2. Encode passwords for the following:
   - **AIA XREF**

To encode the XREF password, run the following command and note the encoded password:

```
$ODI_HOME/bin/encode.sh -INSTANCE=OdiInstanceName AIA_XREF_DB_Password
```

Update the encoded password in the connections file:

```
cd $AIA_PV/aia-comms/commsOracleHome/comms_home/source/soainfra/apps/
services/industry/Communications/BulkDataProcess/BRMToSiebel/
Collections/ODI/Oracle/V1/ODI_Master_Repository

# Replace fp.db.xref.password with value received from encode.sh
vi CONN_AIADS.xml
```

- **BRM**
  To encode the BRM password, run the following command and note the encoded password:

  ```
  $ODI_HOME/bin/encode.sh -INSTANCE=OdiInstanceName
  BRM_Database_Advanced_Queuing_user_password
  ```

  Update the encoded password in the connections file:

  ```
  cd $AIA_PV/aia-comms/commsOracleHome/comms_home/source/soainfra/apps/
  services/industry/Communications/BulkDataProcess/BRMToSiebel/
  Collections/ODI/Oracle/V1/ODI_Master_Repository

  # Replace participatingapplications.brm.db.password with value received
  from encode.sh
  vi CONN_OracleBRMDS.xml
  ```

- **Siebel**
  To encode the Siebel password, run the following command:

  ```
  $ODI_HOME/bin/encode.sh -INSTANCE=OdiInstanceName
  Siebel_database_user_password
  ```

  Update the encoded password in the connections file:

  ```
  cd $AIA_PV/aia-comms/commsOracleHome/comms_home/source/soainfra/apps/
  services/industry/Communications/BulkDataProcess/BRMToSiebel/
  Collections/ODI/Oracle/V1/ODI_Master_Repository

  # Replace participatingapplications.siebel.db.password with value
  received from encode.sh
  vi CONN_SiebelDS.xml
  ```

3. Import the AIA connections to ODI Studio. If the AABC installation is successful, then the **importAIAODIConnections.sh** script will be generated in the **AIA_PV** directory. Run the following command to import:

   ```
   sh importAIAODIConnections.sh \
       -a AIA_PV_Path \
   ```

```
-o ODI_DOMAIN_HOME_path \
-i ODI Instance Name. Example: OracleDIAgent \
-w ODI Workrep name. Example: WORKREP
```

Verifying the Integration

To verify the Siebel CRM on Containers - BRM integration deployment:

1. Login to ODI Studio with ODI user credentials.
2. Verify and confirm that the **AIA Collections Project** is imported into the **Designer** section.
3. Verify and confirm the connection details under **Topology** - **Physical Architecture** - **Oracle and XML**.
   - Oracle - AIA DS
   - Oracle - Oracle BRM DS
   - Oracle - Siebel DS
   - XML - AIA Config Properties DS
   - XML - Collection Action Name
   - XML - Collection Priority DS
   - XML - Collection Status Mapping
   - XML - Collection Substatus model
   - XML - Currency DS

Testing the Solution Deployment

This section provides information about testing the deployment of the solution to check if all applications are integrated correctly and the solution is working end-to-end as expected.

You must perform the following to test end-to-end deployment of the solution:

- Smoke test validation. See Testing the Solution using the Smoke Test Validation for more information.
- Functional testing. See Testing the Integrations using Functional Testing for more information.

Testing the Solution using the Smoke Test Validation

This section provides details of the smoke test validations to be performed to test the deployment of the solution. It includes validating your access to UIs of all the applications integrated with the Digital Business Experience solution and validating if the load balancer is working fine.

Validating the Access to the Application UIs

> **Note:**
>
> - A list of all the supported cloud native application URLs and their usernames and passwords will be shared to your Email ID.
> - Logging in to the VPN is mandatory.

To validate the access to all the application UIs:

1. Open the email consisting of all application URLs and their credentials.
2. Open the application URLs one-by-one and log in using your credentials.
3. Verify if you can access all the URLs.

> **Note:**
>
> If you have trouble logging in to any of the application URLs, contact Oracle Support.

Validating the Access to Load Balancer

To validate if the load balancer is working fine:

1. Open Postman on the environment you wish to test the loadbalancer from.
2. Click **New**, and then click **Request**.
3. From the **Request Type** drop-down list, select **GET**.
   The GET method page opens.
4. In the **GET URL** field, enter the load balancer URL.

> **Note:**
>
> Use the **public DNS name** of your load balancer.

Example load balancer URL for PDC:

```
https://<public_load_balancer_dns_name>:31200/pdc/config
```

Example load balancer URL for Siebel:

```
https://<public_load_balancer_dns_name>:32200/siebel/config
```

5. Under the **Authorization** tab, set the header for authorization.
6. Click **Send**.
7. Validate the response.

- If the status code is **200 OK**, the load balancer is working fine.
- If the status code is **503 Service Unavailable**, there might be a backend issue.

Testing the Integrations using Functional Testing

This section provides information about testing the Launch - PDC data sync and Launch - Siebel data sync, which is the functional testing of the integrations.

The functional testing includes the following:

- Setting up the destination and lifecycle status
- Importing and publishing an initiative
- Verifying the data sync in Siebel
- Verifying the data sync in PDC

Setting Up the Destination and Lifecycle Status

To set up the destination and lifecycle status:

1. Sign in to the Launch application.
2. From the taskbar available at the bottom of the Launch home page, select the **Administration** tab.
   The **Administration** page opens.
3. Click **Manage** on the **Lifecycle status** card.
   The **Entity lifecycle** page opens.
4. Click **View** on the **Destinations** card.
   Ensure that there are destinations created for Siebel and PDC.

> **Note:**
>
> Ensure that the value under the **Sequence** column for Siebel is **1** and for PDC is **2**.

5. Click **Back (<)** to go back to the **Entity lifecycle** page.
6. Click **Manage** on the **Lifecycle status** card.
   The **Lifecycle Status** page opens.
7. Click **Save as New Version**.
   A new version is created. For example, Version 2.0.
8. From the **Version** drop-down list available in the **Entity Lifecycle Status Configuration** section, select the version created from Step 7. For example, Version 2.0.
9. Under the **Actions** column, click the ellipsis button for the **Ready to publish** lifecycle status, and then click **Edit**.
   The **Ready to Publish** page opens.
10. In the **Destinations** field, add the destinations created for Siebel and PDC.
11. Click **Save**.
12. Click the ellipsis button available in the **Entity Lifecycle Status Configuration** section, and then click **Activate.**

Importing and Publishing an Initiative

To import and publish an initiative:

1. Sign in to the Launch application.
2. From the taskbar available at the bottom of the Launch home page, select the **Administration** tab.
   The **Administration** page opens.
3. Click **Manage** on the **Job management** card.
   The **Job Management** page opens.
4. Click **Create** on the **Import jobs** card.
   The **New Import Job** page opens.
5. Click **Select a File**, then browse the file (either .json or .zip only) you want to import, and then click **Import**. For example, `initiative_test.json` file.
   The **Import Jobs** page opens. Wait till the **Job Status** column for your job changes to **Succeeded**.
6. Navigate back to the **Administration** page after the import job is successful.
7. Click **View** on the **Initiatives** card.
   The **Initiatives** page opens.
8. Search for the initiative with your import file name. For example, initiative_test.
9. Under the **Actions** column for the initiative, click the ellipsis button, and then from the **Change status to** list, select the **Ready to publish** option.
10. Click **Publish** by clicking the ellipsis button for your initiative.
11. Track the status of your initiative till it changes to **Launched**.
12. Click the ellipsis button for your initiative, then click **View**, and then click the **Publish Tracker** tab.
    Verify if the status of all jobs is updated to **Success**.

Verifying the Launch - Siebel Data Sync

To verify if the Launch - Siebel data is synced correctly:

1. Sign in to Siebel using your credentials.
2. From the taskbar available at the bottom, click **Administration**, and then click **Product**.
   The **Products** page opens.
3. Search for a product with your initiative name. For example, **initiative_test_PO**.
4. Check if the Price Type is **One-Time**.
   The data is successfully synced to Siebel.

Verifying the Launch - PDC Data Sync

To verify if the Launch - PDC data is synced correctly:

1. Sign in to PDC using your credentials.
2. From the taskbar available at the bottom, click **Administration**, and then click **Charge Offer**.
   The **Charge Offer** page opens.
3. Search for a charge offer with your initiative name. For example, **initiative_test_PO**.

4. From **Charge Actions**, select **Charge Details**, and then select **Charge Category**. Ensure that the Charge Category is **One-Time**.
   The data is successfully synced to PDC.

5
Troubleshooting

This chapter provides information about issues that you may face while deploying the Digital Business Experience solution.

Troubleshooting for the Solution Components

This section provides troubleshooting instructions for the components of the Digital Business Experience solution.

To troubleshoot the deployment for the solution components, refer to the following:

- See Troubleshooting for detailed instructions about troubleshooting Launch and CX Industries Framework applications.
- See Troubleshooting Your BRM Cloud Native Deployment for detailed instructions about troubleshooting BRM.
- See Troubleshooting the ECE Installation for detailed instructions about troubleshooting ECE.
- See Troubleshooting OAP for detailed instructions about troubleshooting OAP.
- See Troubleshooting Issues with the Scripts for detailed instructions about troubleshooting SCD.
- See Debugging and Troubleshooting for detailed instructions about troubleshooting OSM.
- See Troubleshooting Order-to-Activate Cartridges for detailed instructions about troubleshooting O2A.
- See Troubleshooting Installation and Configuration for Siebel CRM for detailed instructions about troubleshooting Siebel.
- See Troubleshooting Oracle Data Integrator for detailed instructions about troubleshooting ODI.
- See Troubleshooting Issues for detailed instructions about troubleshooting AIA.

6

Downloading and Deploying the Reference Solution

This chapter provides information about downloading and deploying the Digital Business Experience Reference Solution (the reference solution).

Before you start to download and deploy the reference solution:

- Learn about the reference solution. See About the Reference Solution in *Oracle Communications Digital Business Experience Concepts* for more details.
- Learn about the reference product models and seed data available in the reference solution. See Reference Product Models and Seed Data in *Oracle Communications Digital Business Experience Concept to Market Guide* for more details.
- Learn about the order to cash reference library, which consists of a few sample preconfigured order-to-cash end-to-end features. See Using the Order to Cash Reference Library in *Oracle Communications Digital Business Experience Order to Cash Implementation Guide* for more details.
- Download and install Oracle Communications Solution Test Automation Platform (STAP) in your Digital Business Experience environment. See Installing STAP for more details.

> **Note:**
>
> The reference solution is packaged with STAP. Hence, you must install STAP first so that you can deploy the reference solution package.

- After installing STAP, set up STAP Design Experience (STAP DE). See Setting Up The STAP Design Experience for more information.

Prerequisites

Following are the prerequisites for installing the reference solution:

- Verify all the REST API end points for various applications and SSH keys, ensuring proper connectivity before running the automation.
- Ensure that you enabled the Rest calls for BCWS and TMF. See Configuring REST Services in *Oracle Communications Billing and Revenue Management Cloud Native Deployment Guide* for detailed instructions about configuring the rest services in BRM and PDC.
- Ensure to edit the config files for pre-automation. See Editing the config files for more information.
- Run the `run_publish.sh` script for pre-automation. See Running the script for more information.
- Verify if the seed data is created properly on Launch, PDC, and BRM. See About the Seed Data in *Oracle Communictions Digital Business Experience Concept to Market Implementation Guide* for more information.

- Check if the destinations for PDC and Siebel are created in Launch.
- Verify if the pdc-test destination has destination exclusion rule to avoid publishing bundles to PDC as it is not supported by Digital Business Experience.
- Ensure that the **Lifecycle Status** for the destinations on the Launch UI is **Ready to Publish**.

Deploying the Reference Solution Package

To deploy the reference solution package:

1. Download the reference solution pack from the Oracle software delivery website, located at:

   http://edelivery.oracle.com

2. Run the following command from the STAP environment to unzip the reference solution pack:

   ```
   unzip oc-stap-otc-testlib-version.zip
   ```

3. Run the following command to edit the **run_publish.sh** file:

   ```
   vi  oc-stap-otc-testlib-version/run_publish.sh
   Replace:
   export JAVA_HOME="/home/java/jdk-xx.x.x" with the correct path to JDK
   export STAP_HOME=/home/STAP-DE-PATH  with the correct path to STAP-DE jar
   For example:
   export JAVA_HOME="/home/user/java/jdk-21.0.6"
   export STAP_HOME="/home/user/STAP-DE"
   ```

4. Edit the config files for Publish-Automation.
   Set up the **tdaas** and **tes** environment files to point them to the STAP deployment to which you want to publish them:

   ```
   /E2E-Automation/Publish-Automation/config/environments/
   TdaasEnvironment.properties
   /E2E-Automation/Publish-Automation/config/environments/TaaS-TES-
   environment.properties
   ```

5. Run the following commands in stages to deploy and publish the reference solution pack to the Digital Business Experience environment:

   ```
   sh run_publish.sh -h
         Usage: sh publish.sh -w <STAP_WORKSPACE_NAME>
         Step 1 - Edit the run_publish.sh files to point to your STAP DE
   location.
         Step 2 - Edit the TaaS-TES-environment.properties and
   TdaasEnvironment.properties files to
         point to your STAP target deployment location.
         Step 3 - Edit the config.properties and environment files in your
   workspace.
         Step 4 - Run the run_publish.sh script with the workspace you want
   to publish and run on your
         STAP deployment.
   ```

```
      For example:
      /E2E-Automation/STAP_Workspace_Name/config/publish/
tdaasEnvironment.properties
      /E2E-Automation/STAP_Workspace_Name/config/publish/
environment.properties
      /E2E-Automation/STAP_Workspace_Name/config/publish/persistence-
volume-environment.properties
      /E2E-Automation/STAP_Workspace_Name/config/publish/publish-
automation.properties
      /E2E-Automation/STAP_Workspace_Name/config/environments/* --> all
files
```

> **Note:**
>
> - In the above command, you can replace STAP_WORKSPACE_NAME with Pre-Automation, SD-Automation, DT-Automation, or RT-Automation based on the workspace you want to publish.
> - You must publish the workspaces in the following order:
>   a. Edit the config files for Pre-Automation workspace, then run the `sh replace_token_paths.sh` command to update the scenario files, and then publish the Pre-Automation workspace.
>   b. Edit the config files for SD-Automation workspace, and then publish the SD-Automation workspace.
>   c. Edit the config files for DT-Automation workspace, and then publish the DT-Automation workspace.
>   d. Edit the config files for RT-Automation workspace, and then publish the RT-Automation workspace.

6. To implement all the run time scenarios, run the following command:

   ```
   sh run_rt_automation.sh --all
   ```

7. To implement a specific run time scenario, run the following command:

   ```
   sh run_rt_automation.sh --scenario_name
   For example: sh run_rt_automation.sh --BroadbandProductModels
   ```

8. To implement the bundled sample orders, run the following command:

   ```
   sh run_rt_automation.sh --scenario_name/subfolder_name
   For example: sh run_rt_automation.sh --MobileProductModels/
   Supremo5GUnlimited/SampleOrder
   ```

Troubleshooting the Reference Solution

This section provides guidelines to help you troubleshoot problems while deploying the reference solution.

Seed Data Issues

- If the loading of BRM data fails:
  - Load the following files from the BRM scenario data folder in the cm pod:
    - `customservices_proximity.podl`
    - `customservices_OTT_VOIP_dtv_dmusic.podl`
  - Sync the new events and services by restarting the cm, dm, and syncpdc pods.
- If the importing of PDC data fails, create the ServiceEvent map manually from the PDC UI. See BRM PDC Seed Data in *Oracle Communications Digital Business Experience Concept to Market Implementation Guide* for a list of supported PDC Services, Events, and ServiceEvent maps.