ellucian

Banner Web Services
Installation Guide

Release 8.1.4, Revision 1
May 2014

Prepared by: Ellucian
4375 Fair Lakes Court
Fairfax, Virginia 22033
United States of America

Revision History

Publication Date	Summary
February 2014	New version that supports Banner Web Services 8.1.4 software.
May 2014	Revised the location of the logging file.

Banner Web Services 8.1.4
Installation Guide

Contents

1 Introduction

This chapter introduces web services and describes how Banner® uses web services.

What is a web service?

A web service exposes an application's processing logic to support a service-oriented architecture and to facilitate integration with external systems. A web service allows an external system or business process to invoke the application's logic without having to understand the application's internal structure.

Web services are based on open, Internet-based standards. This makes them relevant to application integration within an organization and with external organizations. Standards such as XML, SOAP, WSDL, and UDDI provide cross-platform compatibility that does not depend on a single programming language or network transport.

How do Banner web services work?

Java-based adapters expose Banner functions as web services. This exposure makes the Banner functions available to external systems using the SOAP protocol over HTTP/HTTPS. External systems interact with the web services, which in turn are supported by Banner APIs. This layered approach provides an insulating buffer between external systems and Banner. External systems do not interact with Banner directly, but rather exchange XML messages with the exposed web services.

The Banner Web Services Adapters support the synchronous, request/reply message exchange pattern as follows:

1. The external system requests a service of Banner by sending an XML message to the web service endpoint that is exposed by the adapter. The message contains the information required for Banner to service the request.

2. The Banner Web Services Adapter invokes the appropriate Banner API.

3. The Banner API performs the necessary Banner processing logic.

4. One of the following occurs:

    4.1. If the action is completed successfully, the API provides a response message, which the adapter forwards to the external system.

    4.2. If the action is not completed successfully, the adapter sends an error message (called a SOAP fault) to the external system.

Each web service is independent. A business process, however, can invoke several web services during the process.

The Banner Cardholder Event Publisher also uses web services, in a different way, to integrate Banner with external systems. Rather than receive requests from external systems, the Banner Cardholder Event Publisher pushes data from Banner to external systems when specified data elements change. The Publisher provides near real-time publication of data for which Banner is authoritative.

Example uses of Banner web services

Banner web services connect external systems with Banner. These connections enforce vital business processes and allow data to be centrally maintained. Data from one system can be used as input to the other. Data updates in one system can initiate data update in the other.

Banner web services support integration with campus card systems and housing systems. The available services, however, are not limited to these systems and can be used for other business needs.

Integration with campus card systems

Campus cards are electronic or magnetic cards that are issued to students, staff, faculty, and other constituents. Institutions use campus card systems to identify individuals, allow physical access to institution facilities, provide purchasing capabilities at campus point-of-purchase venues, and provide access to other institutional offerings such as libraries.

Web services can be used to provide real-time interaction between third-party campus card systems and Banner. Banner is the authoritative source for constituent information. Campus card systems are the authoritative source for card-access related data.

Integration with housing systems

Institutions use housing systems to accept applications for housing, assign applicants to residence halls and related services, and manage campus residence facilities.

Web services can be used to provide real-time interaction between third-party housing systems and Banner. Banner is the authoritative source for constituent information. Housing systems are the authoritative source for housing and resident-related data.

Supporting components

The following components work together to provide web services-based integration with Banner:

- Banner Translation Service
- Banner Web Services Adapters
- Banner Cardholder Event Publisher

Banner Translation Service

Data in Banner is often constrained by lists of valid values. These valid values are stored in support tables and can be configured by your institution. As a result, the data might not be appropriate or usable by external systems. Banner Translation Service converts institution-specific data values in Banner to standard values that external systems can recognize and use.

Refer to the *Banner Translation Service Installation and Administration Guide* for more details.

Banner Web Services Adapters

The Banner Web Services Adapters expose Banner functions as web services. An adapter can be configured to expose any number of defined web services. Two sample configurations are provided as J2EE compatible enterprise archive files. One configuration exposes web services that external campus card systems need to integrate with Banner. Another configuration exposes web services that external housing systems need to integrate with Banner. These adapters are referred to as the Banner Web Services Adapter for Campus Card Systems and the Banner Web Services Adapter for Housing Systems, respectively.

The adapters refer to an XML document for configuration information. This in-memory singleton class is built from a preconfigured XML file (`process-config.xml`). Configuration information includes the following elements:

- Data source that identifies the Banner database for performing database transactions
- Banner Translation Service lookup, regular expression, and delimiters that perform static and dynamic value translations

- Mappings for message types identified by the incoming root XML element to a PL/SQL packaged procedure to process the request
- List of XSL transformations used to convert UDC schema document instances to Banner schema document instances, and vice versa

Banner Cardholder Event Publisher

The Banner Cardholder Event Publisher publishes data from Banner to external systems when cardholder data changes in Banner tables. The following processing occurs:

1. Oracle Streams captures the table changes and publishes a corresponding Banner Identity event to the Campus Card Event Topic.
2. The Banner Cardholder Event Publisher reads events posted to this topic, retrieves Banner cardholder data, transforms the retrieved data to the proper format, and publishes SyncEligibleCardholder messages to a campus card system's exposed web service endpoint that supports the SyncEligibleCardholder interface and SOAP binding.

The Banner Cardholder Event Publisher is delivered as a J2EE compatible enterprise archive file and works with the Banner Web Services Adapter for Campus Card Systems.

Installation overview

Your institution's specific integration requirements influence which components you install. Use the following steps to guide your installation.

1. Review the *Banner Web Services Handbook* for an overview of available Banner web services and associated configuration requirements.
2. Decide what components need to be installed, based on your institution's needs.
3. Install the Banner Translation Service. Refer to the *Banner Translation Service Installation and Administration Guide* for details.
4. If needed, install and configure the appropriate Banner Web Services Adapters (see Chapter 2, "Install Banner Web Service Adapters").
5. If Banner Web Services Adapters are installed, verify the configuration (see Chapter 3, "Verify the Configuration").
6. Import translations for each deployed Banner Web Services Adapter into the Banner Translation Service (see Chapter 4, "Import Translations").

7. Customize the exposed web services for each deployed Banner Web Services Adapter (see Chapter 5, "Customize Web Service Responses" for an overview and the *Banner Web Services Handbook* for more details).

8. (Recommended) Test the installed web services (see Chapter 6, "Test Banner Web Services").

9. If needed, install the Banner Cardholder Event Publisher (see Chapter 7, "Install Banner Cardholder Event Publisher").

10. If the Banner Cardholder Event Publisher is installed, test the deployment (see Chapter 8, "Test Banner Cardholder Event Publisher").

Related documentation

The following documents provide more information about Banner web services:

- The *Banner Web Services Handbook* describes the messages, message mapping to Banner, intended usage, setup requirements, and translations for the Banner web services that support integration with campus card systems and housing systems.
- The *Banner Translation Service Installation and Administration Guide* provides information on installing and administering the Banner Translation Service, a prerequisite component for exposing the Banner web services.

2 Install Banner Web Service Adapters

The Banner® Web Services Adapters expose Banner functions as web services. Two sample configurations are provided as J2EE compatible enterprise archive files:

- One configuration exposes web services that external *campus card systems* need to integrate with Banner.
- Another configuration exposes web services that external *housing systems* need to integrate with Banner.

This chapter gives instructions for installing the adapters on Oracle WebLogic Server 11g. This chapter also lists the URLs that expose the WSDL (Web Services Description Language) files that define the web services exposed by the Banner Web Services Adapters.

Requirements

The Banner Web Services Adapters have the following requirements.

Oracle application server and Java

The Banner Web Services Adapters expose Banner functions as web services. These adapters are certified on Oracle WebLogic Server 11g with Java 1.7.

Oracle database

The Oracle Database 11g is required.

Banner Translation Service

You *must* install the Banner Translation Service *before* you deploy the Banner Web Services Adapters. Refer to the *Banner Translation Service Installation and Administration Guide* for details.

Banner products

The following Banner products support the Banner Web Services Adapters and must be installed:

Banner Web Services Adapter for Campus Card Systems

Product	Minimum Version
Banner General	8.0 plus patch p1-46c8mj_gen80100

Banner Web Services Adapter for Housing Systems

Product	Minimum Version
Banner General	8.0 plus patch p1-46c8mj_gen80100
Banner Student	8.0
Banner Accounts Receivable	8.0

Recommended configuration

The adapters must be installed in an Oracle WebLogic Basic Domain. They must not be installed using any other Oracle WebLogic template, especially the Oracle WebLogic Classic Domain that supports Oracle Forms and Reports.

The recommended configuration is to establish a separate physical or virtual domain for the adapters and other middle-tier components. This domain would run a separate installation of Oracle WebLogic Server, configured using the Basic Domain template (not the Classic Domain template) that is provided by Oracle.

The Oracle WebLogic Server domain should consist of the default Admin Server and at least one Managed Server for the deployment of the adapters and the Banner Translation Service. The adapters must be installed on the same Managed Server as the Banner Translation Service.

If a domain based on the Basic Domain template already exists for middle-tier applications, the adapters can be installed with the Banner Translation Service in a separate Managed Server in that domain.

Installation steps

The adapters are packaged as J2EE compatible enterprise archive files. Each file must be deployed and configured separately:

- `CampusCardIntegration_v8.1.4.ear` exposes web services that external campus card systems need to integrate with Banner.
- `HousingIntegration_v8.1.4.ear` exposes web services that external housing systems need to integrate with Banner.

Use the following steps to install each adapter on Oracle WebLogic Server 11g:

- Step 1, “Configure the Oracle WebLogic Server”
- Step 2, “Configure logging (optional)”
- Step 3, “Define the data source”
- Step 4, “Install the adapter”
- Step 5, “Configure the security group and user”
- Step 6, “Enable schema validation (optional)”

Step 1 Configure the Oracle WebLogic Server

The Oracle WebLogic Server must be configured to use the *Advanced* security model instead of the default *DD only* option. This step pertains to the realm configuration. It applies to the entire domain. (Although you can create a totally new realm for the domain, only one realm can be active at a time for the entire domain.) This security configuration protects all server resources for the domain.

Note

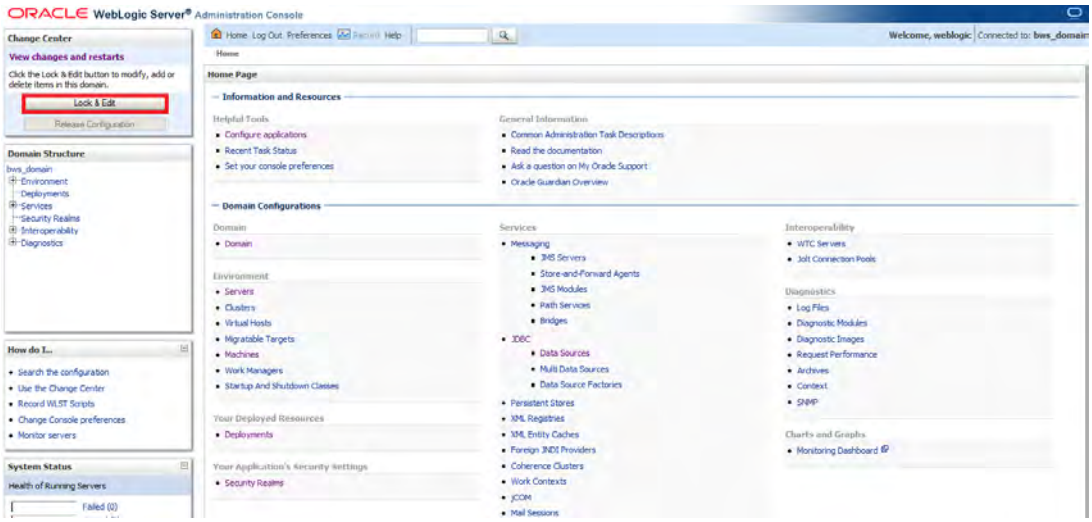
The Oracle WebLogic Server needs to be configured only once. If the server was previously configured, you can skip this step.

Use the following steps to configure the server.

1. Connect to the Oracle WebLogic Server administration console:

   `http://<host>:<port>/console`

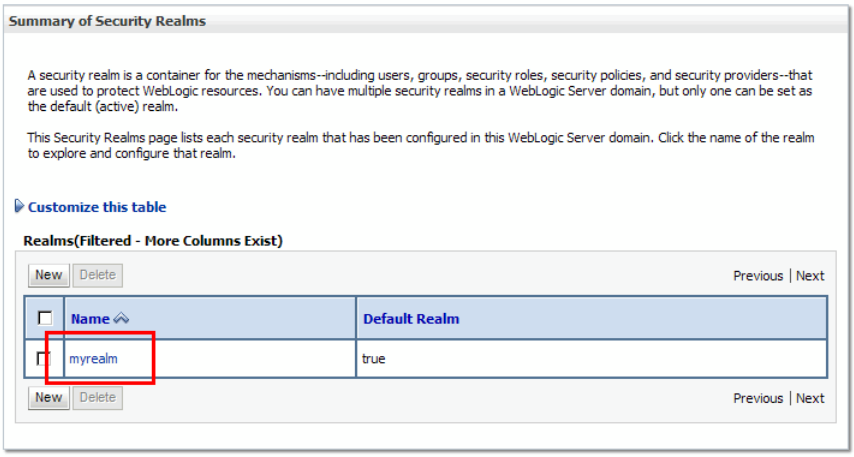
   The Home Page is displayed.



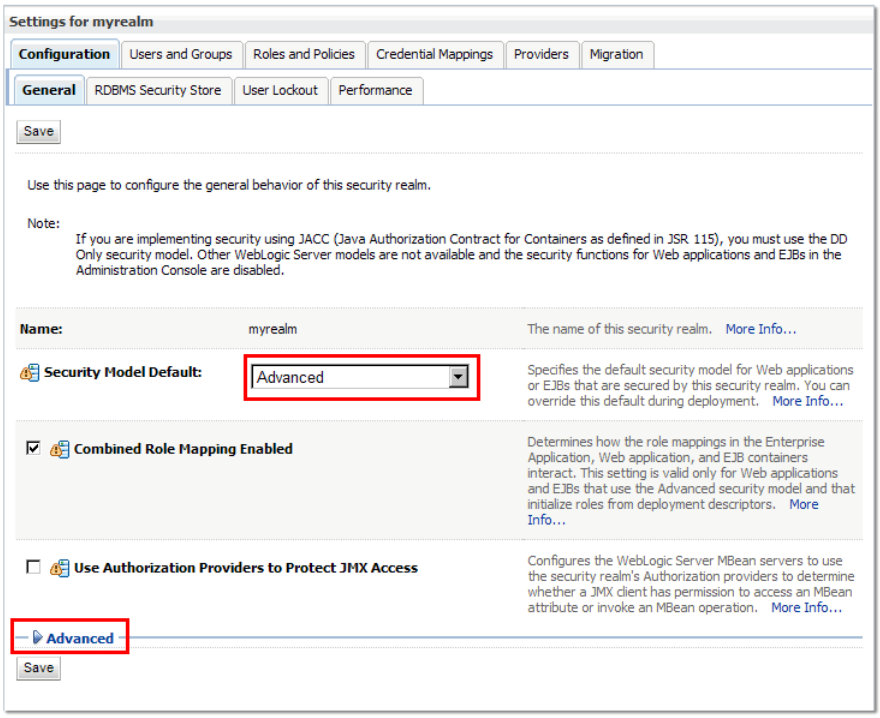
2. In the Change Center pane, click **Lock & Edit**.

3. In the Domain Structure pane, click **Security Realms**.

The Summary of Security Realms page is displayed.

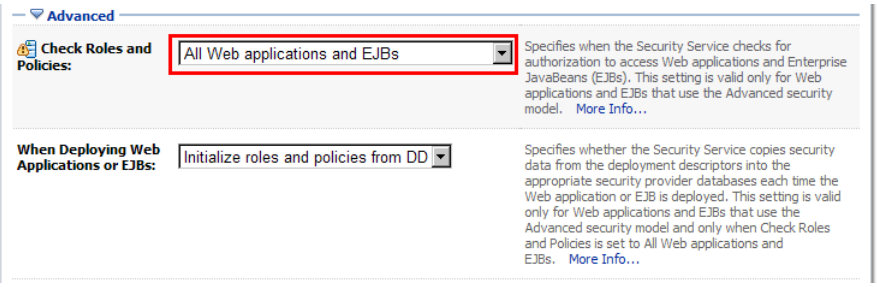


4. Click **myrealm**. The Settings for myrealm page is displayed.



5. Select *Advanced* in the **Security Model Default** drop-down list.

6. Click the **Advanced** link to display the advanced options.



7. Select *All Web Applications and EJBs* in the **Check Roles and Policies** drop-down list.

8. Click **Save**.

9. Click **Activate Changes** and wait for a console message before proceeding.

Step 2 Configure logging (optional)

The Banner Web Services Adapters use Apache's log4j to log the activities performed by the applications at runtime. The log file is located at the following location:

```
Oracle\Middleware\user_projects\domains\<domain_name>\servers\
<Managed_Server_name>\logs
```

where `<domain_name>` is the name of the domain where the Banner Web Services Adapters will be installed. This location cannot be changed.

A property in the `log4j.properties` file determines the logging level. The default logging level is *DEBUG*, resulting in a large amount of information (INFO, WARNING, ERROR, and FATAL level statements) being stored in log files. The logging level should be *DEBUG* for initial operations only. To provide more limited logging after initial operations, you should change the logging level to *INFO*.

Use the following steps to modify the logging level if you want less detailed logging.

1. Extract `BANNER_WEB_SERVICES_814.zip`. The extract directory is referred to as `<ZIP_HOME>`.

2. Configure logging for the Banner Web Services Adapter for Campus Card Systems as follows:

   2.1. Navigate to `<ZIP_HOME>/Deployables/Weblogic/campus_card_adapter/ear/CampusCardIntegration_v8.1.4.ear`.

   2.2. Copy `CampusCardIntegration_v8.1.4.ear` to a temporary location. This location is referred to as `<EAR_HOME>`.

2.3. Navigate to <EAR_HOME> and execute the following command.

```
jar xvf CampusCardIntegration_v8.1.4.ear
```

The extract contains CampusCardIntegration_web.war.

2.4. Create a folder under <EAR_HOME> and name it war_home.

2.5. Navigate to war_home and execute the following command.

2.6. `jar xvf <EAR_HOME>/CampusCardIntegration_web.war`

2.7. Open war_home/WEB-INF/classes/log4j.properties.

2.8. Edit the log4j.category.com.sungardsct property as follows:

Original value:	*DEBUG*
New value:	*INFO*

2.9. Save the change.

2.10. From war_home execute the following command to rebuild the .war file.

```
jar cvf <EAR_HOME>/CampusCardIntegration_web.war META-INF/
* WEB-INF/* ui/* index.jsp
```

2.11. From <EAR_HOME> execute the following command to rebuild the .ear file.

```
jar cvf CampusCardIntegration_v8.1.4.ear *.war META-INF/*
legal/* APP-INF/*
```

The rebuilt CampusCardIntegration_v8.1.4.ear is used for installation.

3. Configure logging for the Banner Web Services Adapter for Housing System as follows:

3.1. Navigate to <ZIP_HOME>/Deployables/Weblogic/housing_adapter/ear/HousingIntegration_v8.1.4.ear.

3.2. Copy HousingIntegration_v8.1.4.ear to a temporary location. This location is referred to as <EAR_HOME>.

3.3. Navigate to <EAR_HOME> and execute the following command.

```
jar xvf HousingIntegration_v8.1.4.ear
```

The extract contains HousingIntegration_web.war.

3.4. Create a folder under <EAR_HOME> and name it war_home.

3.5. Navigate to war_home and execute the following command.

```
jar xvf <EAR_HOME>/HousingIntegration_web.war
```

3.6. Open `war_home/WEB-INF/classes/log4j.properties.`

3.7. Edit the `log4j.category.com.sungardsct` property as follows:

Original value:	*DEBUG*
New value:	*INFO*

3.8. Save the change.

3.9. From `war_home` execute the following command to rebuild the .war file.

```
jar cvf <EAR_HOME>/HousingIntegration_web.war META-INF/*
WEB-INF/* ui/* index.jsp
```

3.10. From `<EAR_HOME>` execute the following command to rebuild the .ear file.

```
jar cvf HousingIntegration_v8.1.4.ear *.war META-INF/*
legal/* APP-INF/*
```

The rebuilt `HousingIntegration_v8.1.4.ear` is used for installation.

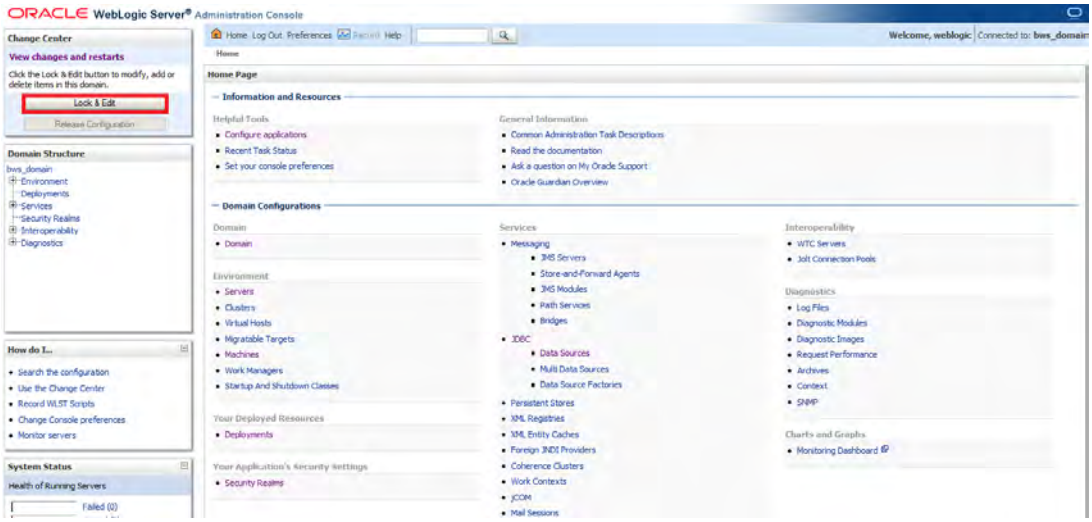
Step 3 Define the data source

A data source provides the connection properties to the Banner database. By default, the adapter needs a data source with lookup name `jdbc/bannerws`. If you previously installed a Banner Web Services Adapter in the instance, then you can skip this step. Otherwise, use the following steps to define the data source.

1. Connect to the Oracle WebLogic Server administration console:

```
http://<host>:<port>/console
```

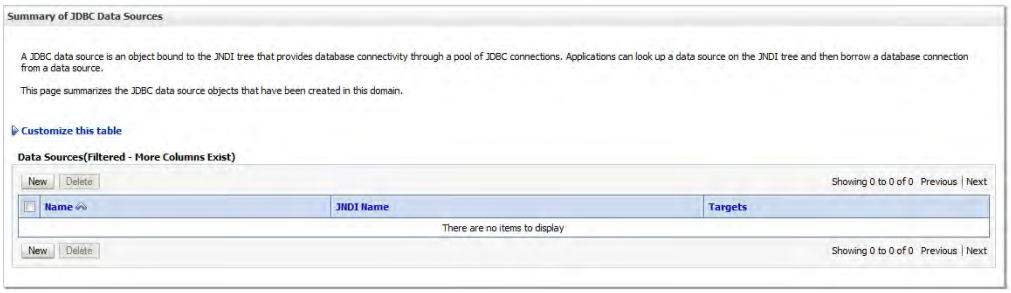
The Home Page is displayed.



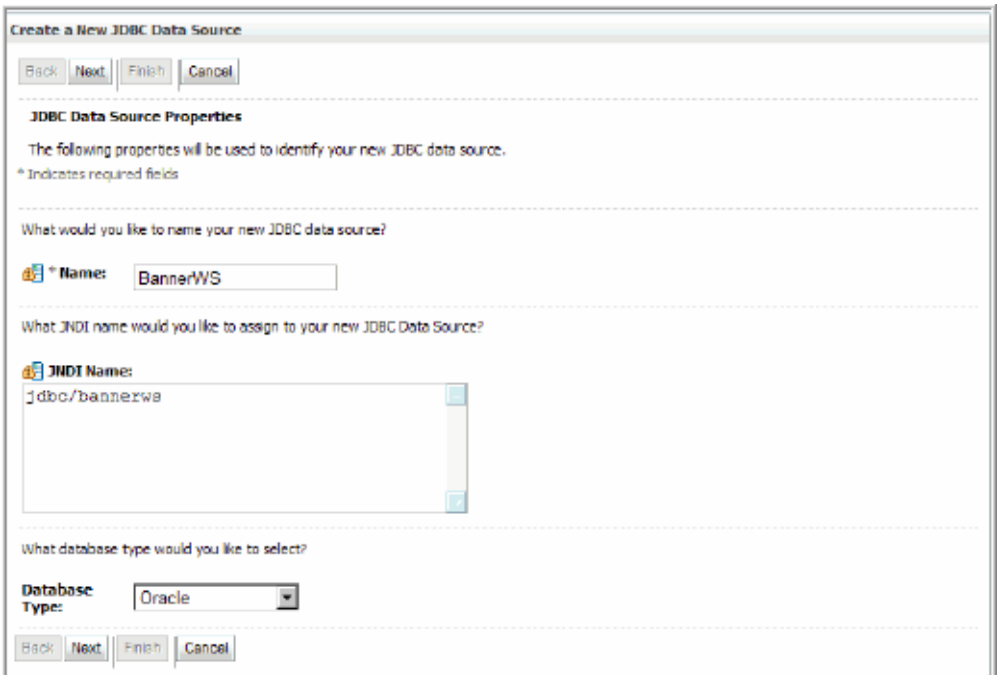
2. In the Change Center pane, click **Lock & Edit**.

3. In the Domain Structure pane, expand and click **Services > JDBC > Data Sources**.

The Summary of JDBC Data Sources page is displayed.



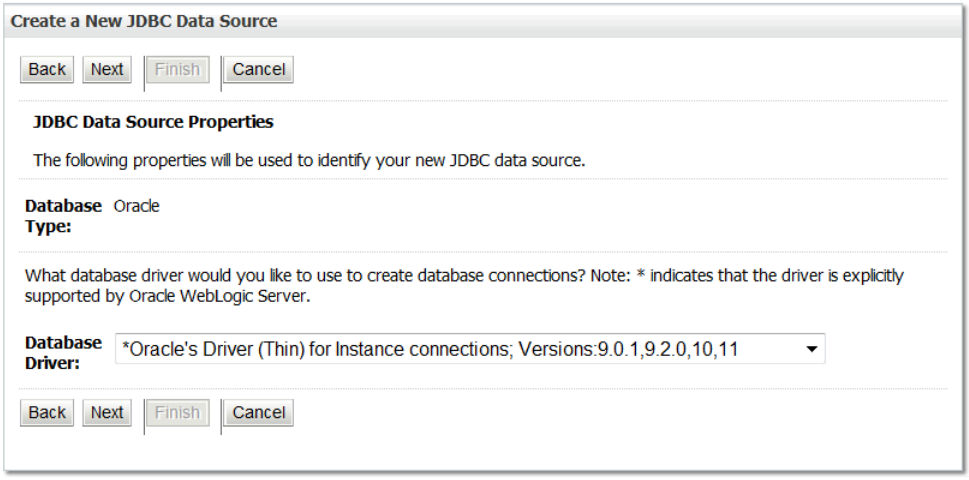
4. Click **New**. The Create a New JDBC Data Source page is displayed.



5. Enter the following data source properties:

Name	*BannerWS*
JNDI Name	*jdbc/bannerws*
Database Type	*Oracle*

6. Click **Next**. The next page is displayed.



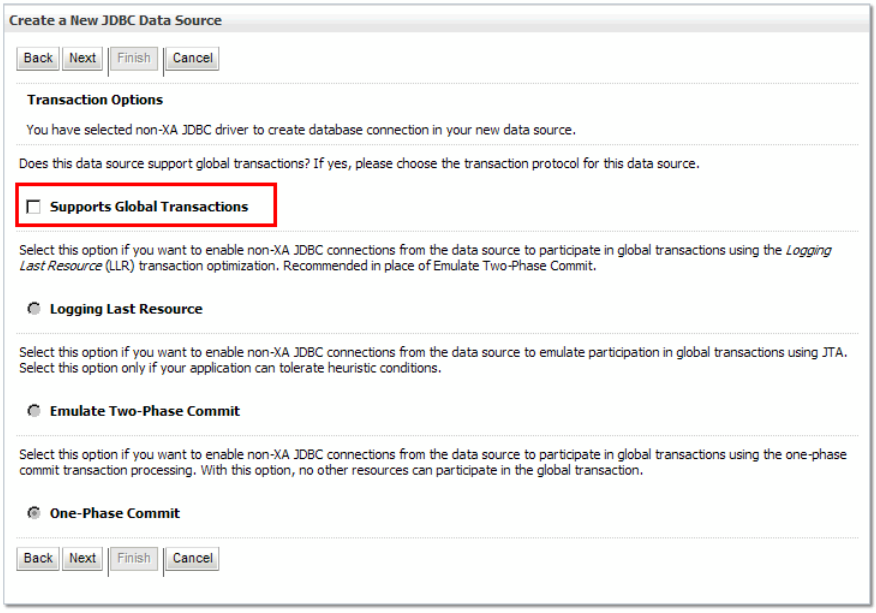
7. Select the appropriate database driver that is used to create database connections:

   - If your database is RAC-based, select **Oracle's Driver (Thin) for RAC Service-Instance connections; Versions:10,11*.
   - Otherwise, select **Oracle's Driver (Thin) for Instance connections; Versions:9.0.1,9.2.0,10,11*.

8. Click **Next**. The next page is displayed.

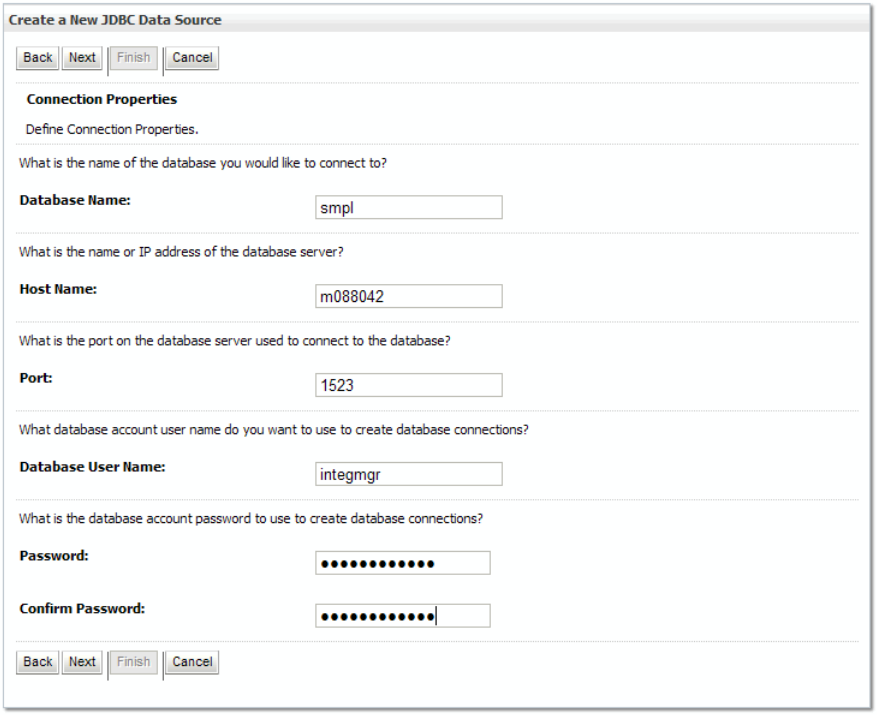
   If the Connection Properties page is displayed, go directly to step 9.

   If the Transaction Options page is displayed, clear the **Supports Global Transactions** check box and click **Next**. Then go to step 9.

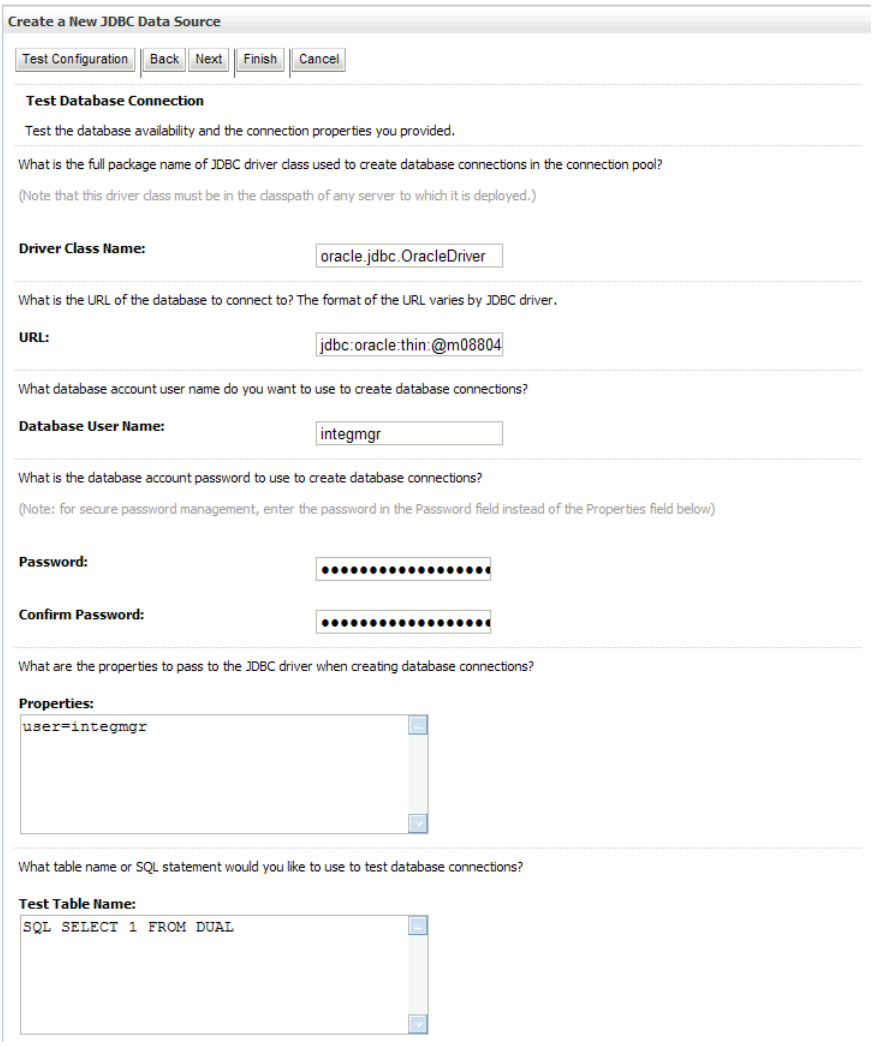


9. Enter the following properties on the Connection Properties page:

Service Name	Service name of the database to which you are connecting. **Note:** This field is displayed and is required if you selected **Oracle's Driver (Thin) for RAC Service-Instance connections; Versions:10,11* as the database driver.
Database Name	Name of the database to which you are connecting
Host Name	IP address or name of the database server
Port	Port on the database server that is used to connect to the database
Database User Name	*integmgr*
Password	Password for the `integmgr` user
Confirm Password	Confirmation of the password



10. Click **Next**. The next page is displayed with the properties that you entered.



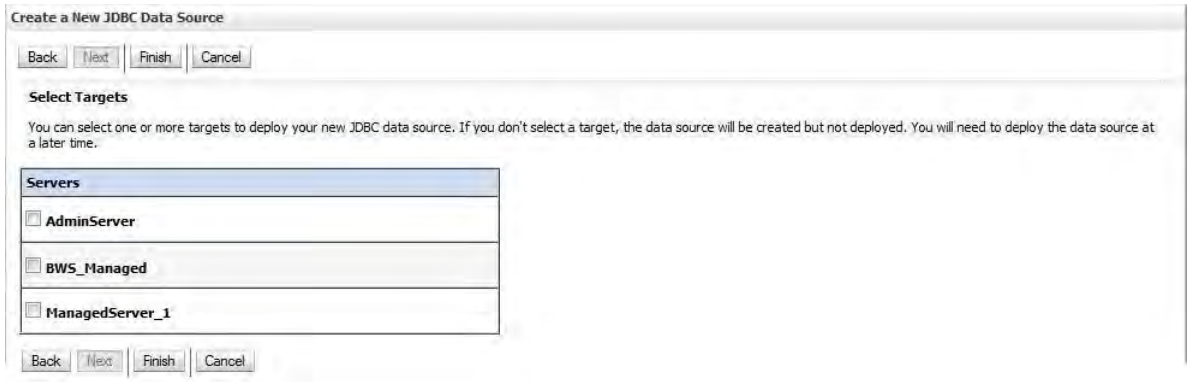
11. Verify the property values.

12. Click **Test Configuration**. The page is redisplayed with a success or failure message.

    12.1. If the test succeeds, continue with the next step.

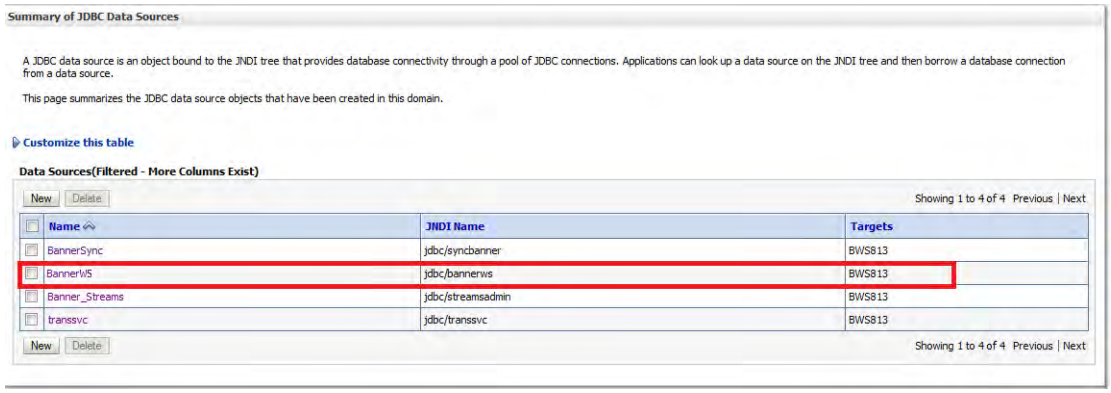
    12.2. If the test fails, ensure that the connection URL and credentials are correct. Continue testing until the connection is successful.

13. Click **Next**. The Select Targets page is displayed.



14. Select the server(s) where you want to deploy the new data source.

15. Click **Finish**.The Summary of JDBC Data Sources page is displayed with the new data source.



16. Verify that the new data source is associated with the server.

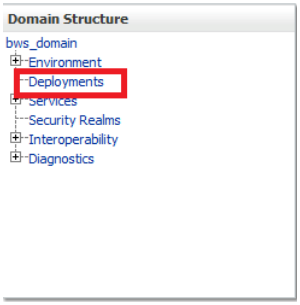
17. In the Change Center pane, click **Activate Change**.

Step 4 Install the adapter

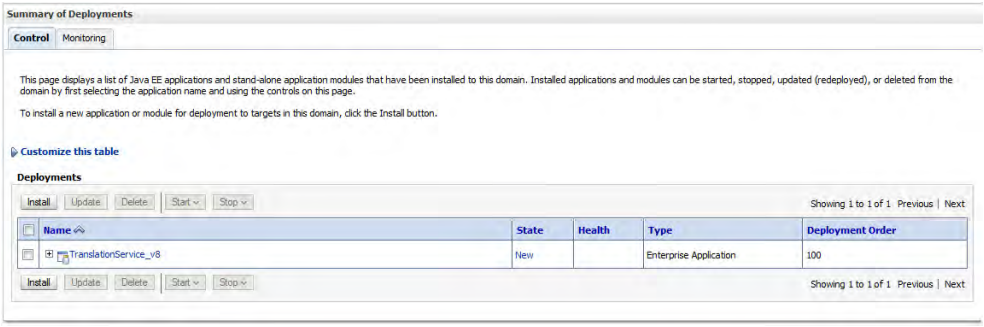
Use the following steps to install the adapter to the Oracle WebLogic Server.

1. In the Change Center pane, click **Lock & Edit**.

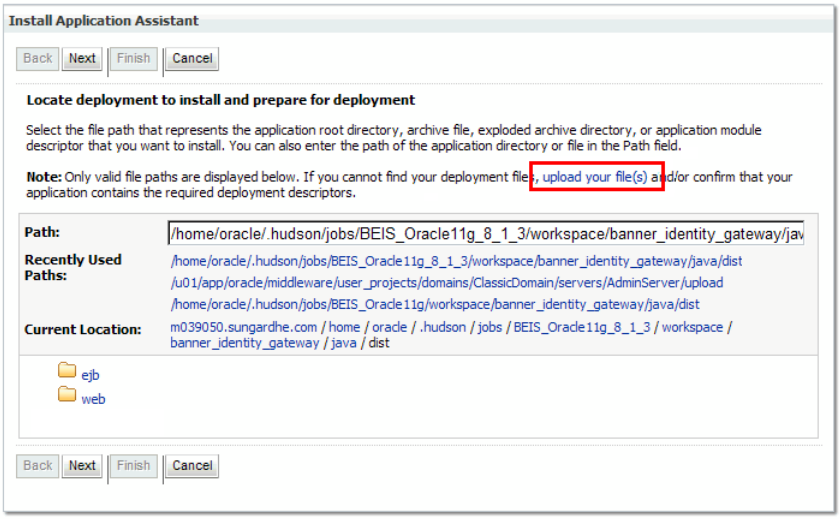
2. In the Domain Structure pane, click **Deployments**.



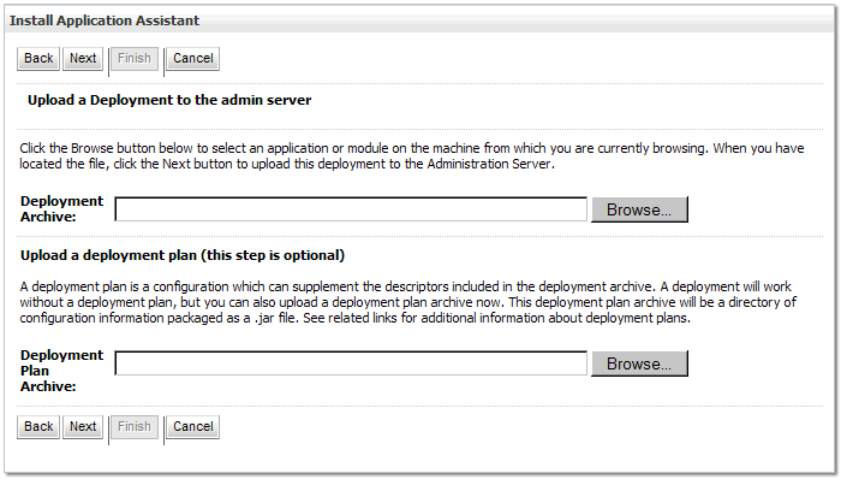
The Summary of Deployments page is displayed.



3. Click **Install**. The Install Application Assistant page is displayed.



4. Click **upload your file(s)**. The next installation page is displayed.



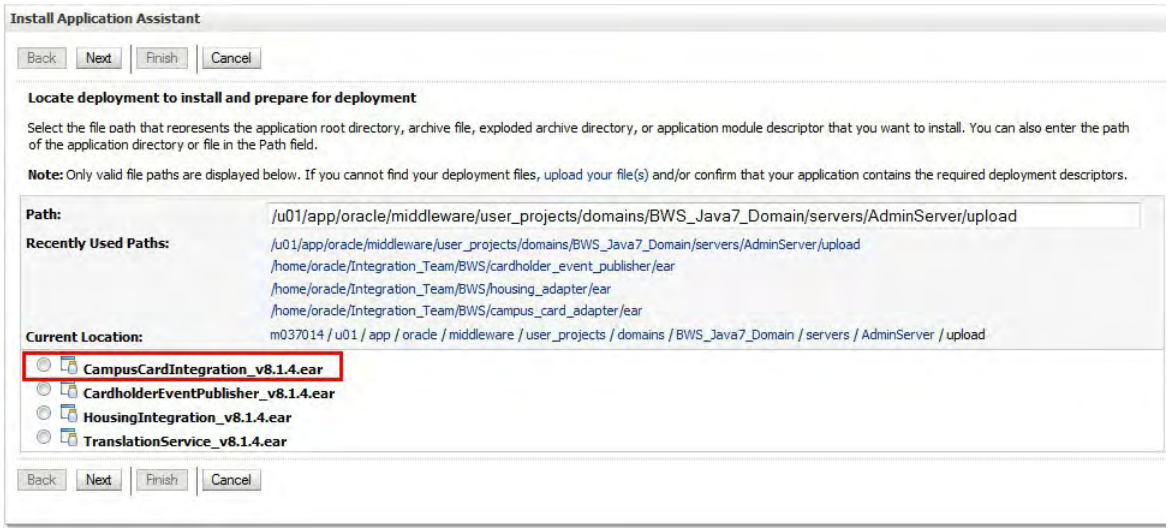
5. Select the file to be uploaded:

   5.1. In the **Deployment Archive** field, click **Browse** and navigate to the appropriate ear file:

   `CampusCardIntegration_v8.1.4.ear`
   or
   `HousingIntegration_v8.1.4.ear`

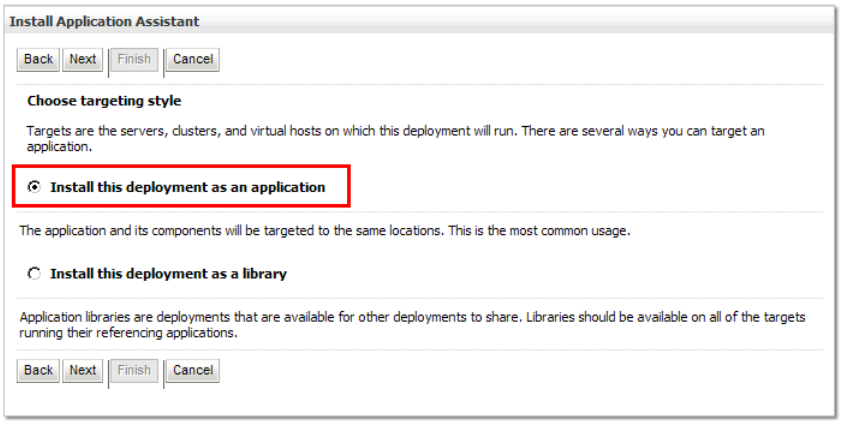
   5.1. Select the file and click **Open**.

6. Click **Next**. The next installation page is displayed.



7. Select the adapter ear file from the list.

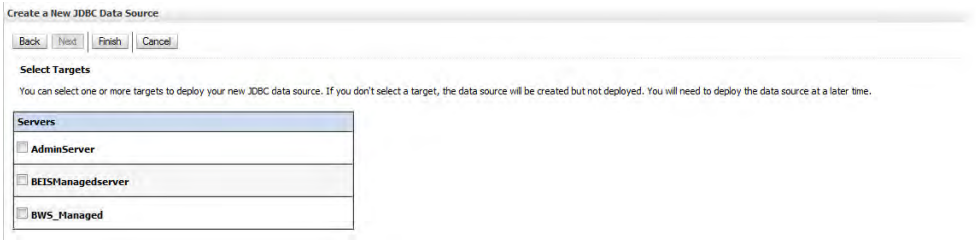
8. Click **Next**. The next installation page is displayed.



9. Select **Install this deployment as an application**.

10. Click **Next**. The Select Targets page may or may not be displayed, depending on the domain.

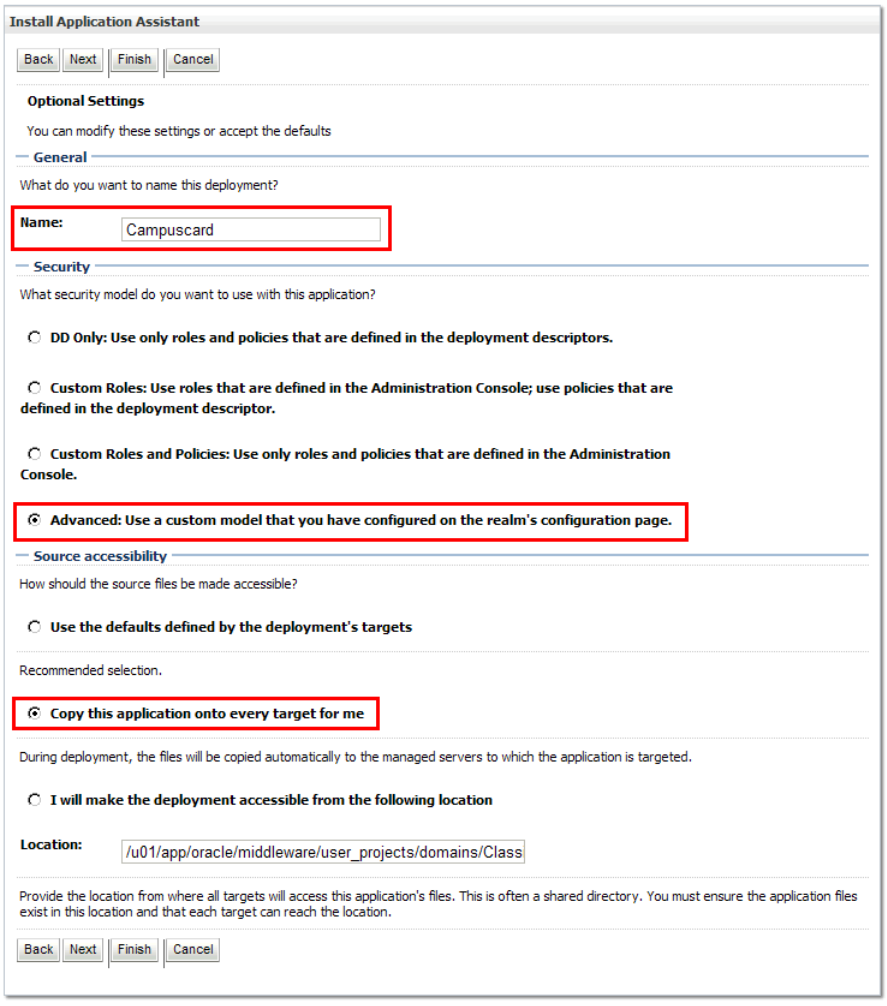
    10.1. If the Select Targets page *is* displayed, select the server where the adapter should be deployed. The adapter must be installed in the instance where the Banner Translation Service is installed. Then click **Next** to display the Optional Settings page.



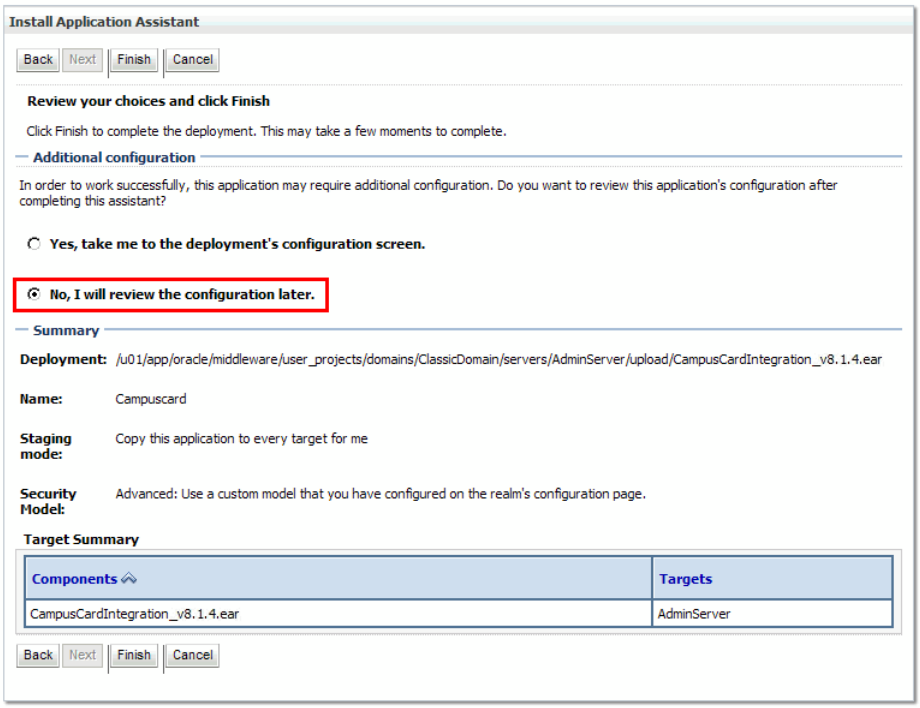
    10.2. If the Select Targets page *is not* displayed (rather, the Optional Settings page is displayed), check your WebLogic server configuration to ensure that a Managed Server is available for deployment of applications. If a Managed Server is not available, the adapter will be deployed to the Admin Server, which is not a recommended configuration. For more information, consult the Oracle WebLogic Server Documentation Library.

11. Enter the following information on the Optional Settings page:

Name	Name for the application (for example, *Campuscard*)
Advanced: Use a custom model that you have configured on the realm's configuration page	Select the radio button.
Copy this application onto every target for me	Select the radio button.

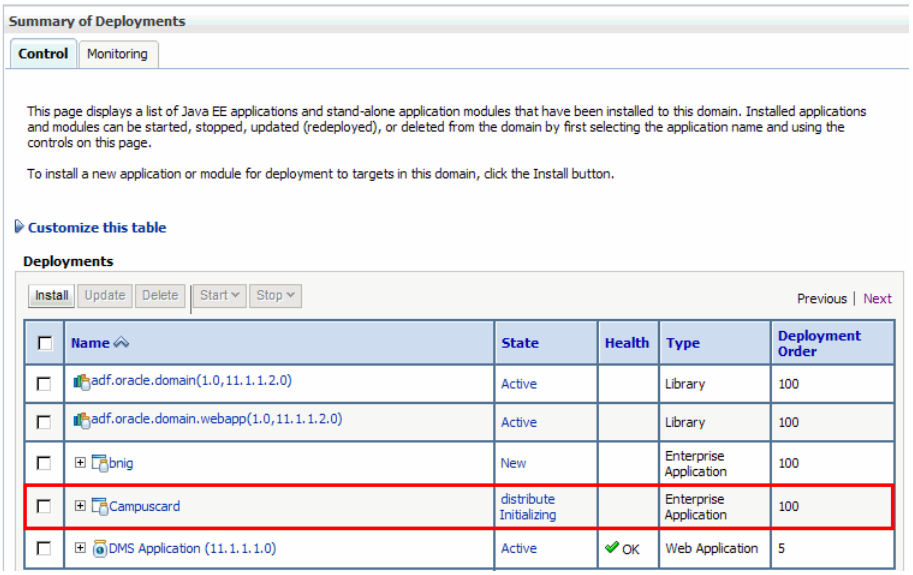


12. Click **Next**. The next installation page is displayed.



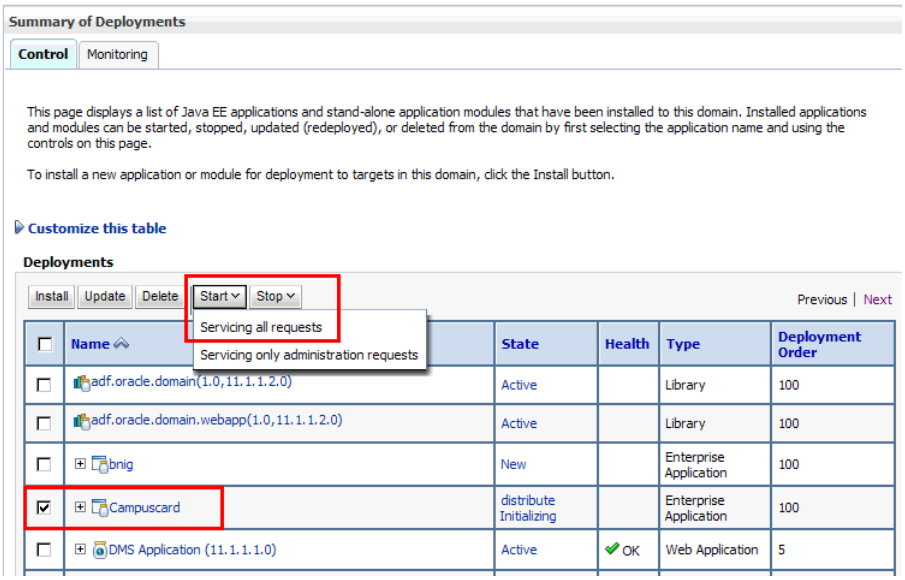
13. Select **No, I will review the configuration later**.

14. Click **Finish** to start the deployment. When deployment is completed, the Summary of Deployments page is redisplayed with the newly deployed adapter.



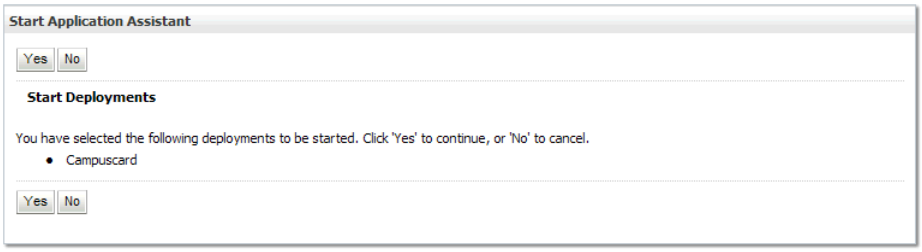
15. In the Change Center pane, click **Activate Changes**.

16. Start the newly deployed application as follows:



16.1. Select the newly deployed adapter.

16.1. Click **Start > Servicing all requests**. The Start Application Assistant page is displayed.



16.2. Click **Yes**.

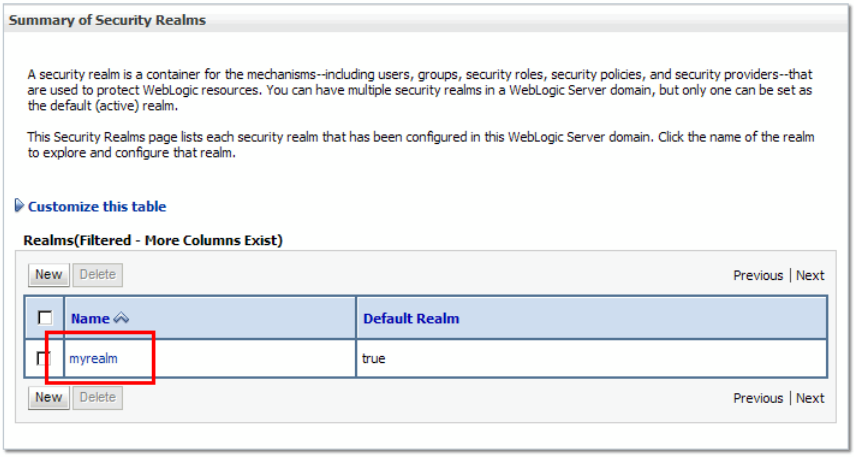
Step 5 Configure the security group and user

Use the following steps to add the `bannerwsGroup` group and an administrative user to the adapter. This group and user protect the defined endpoint.

1. In the Change Center pane, click **Lock & Edit**.

2. In the Domain Structure pane, click **Security Realms**.

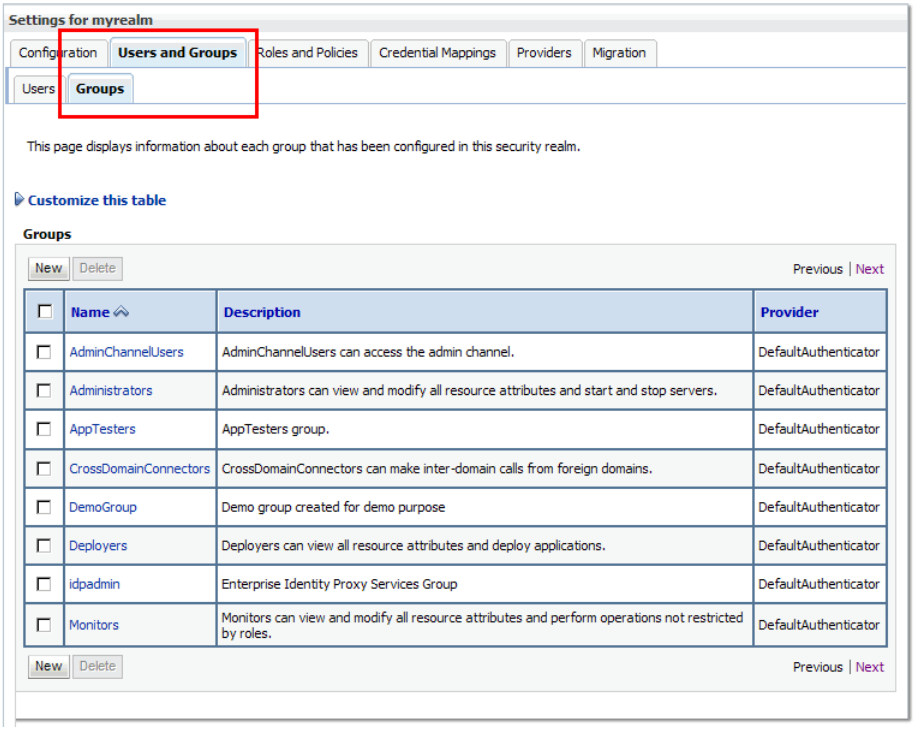
The Summary of Security Realms page is displayed.



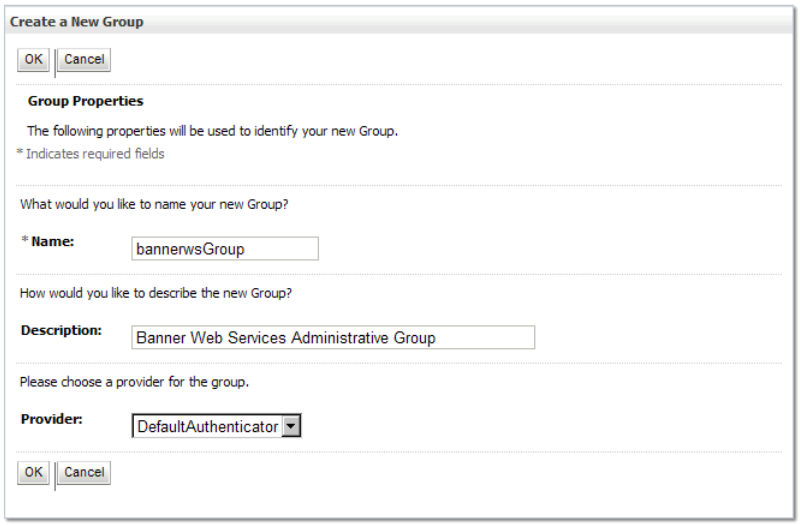
3. Click **myrealm**. The Settings for myrealm page is displayed.

4. Select the **Users and Groups** tab.

5. Select the **Groups** sub-tab. A table of existing groups is displayed.



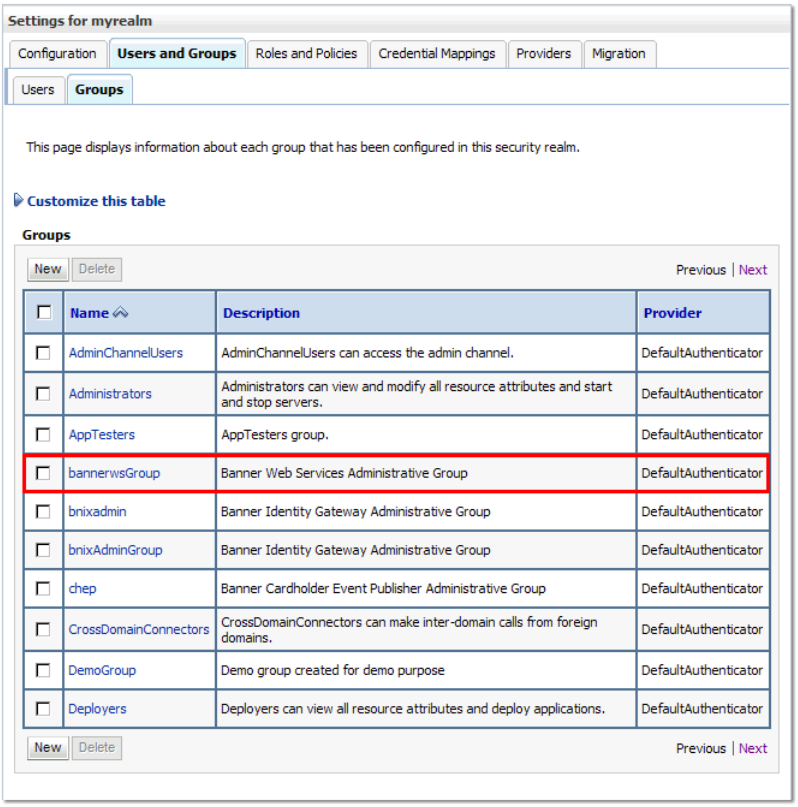
6. Click **New**. The Create a New Group page is displayed.



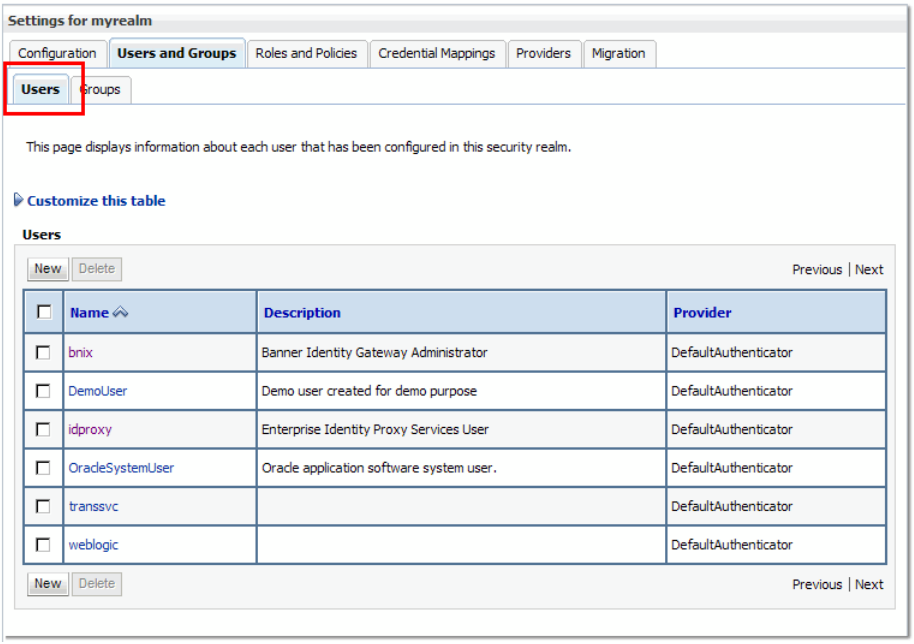
7. Enter the following information to create a group:

Name	*bannerwsGroup*
Description	*Banner Web Services Administrative Group*
Provider	*DefaultAuthenticator*

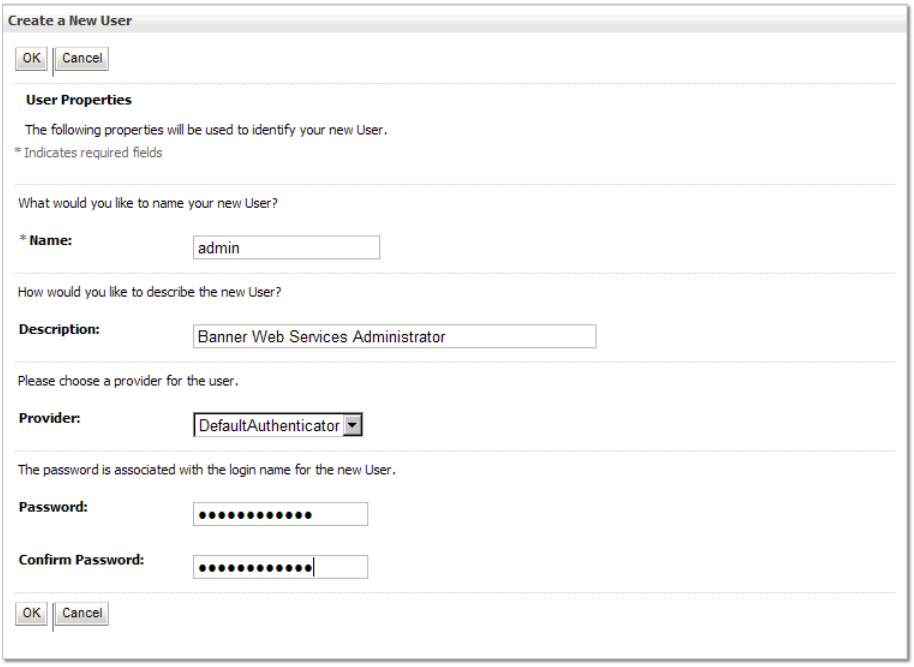
8. Click **OK**. The table of groups is redisplayed with the new group.



9. Select the **Users** sub-tab. A table of existing users is displayed.



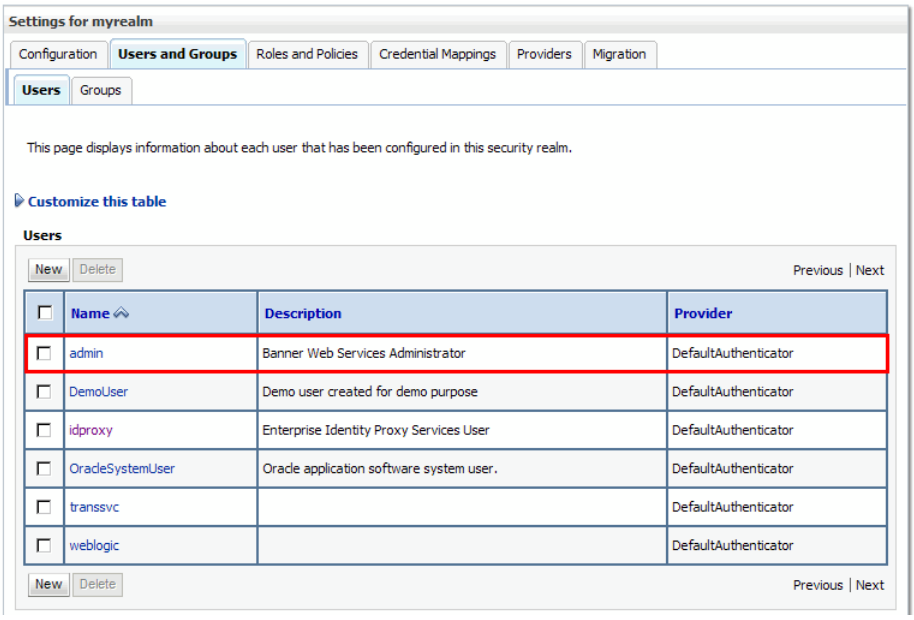
10. Click **New**. The Create a New User page is displayed.



11. Enter the following information to create a user:

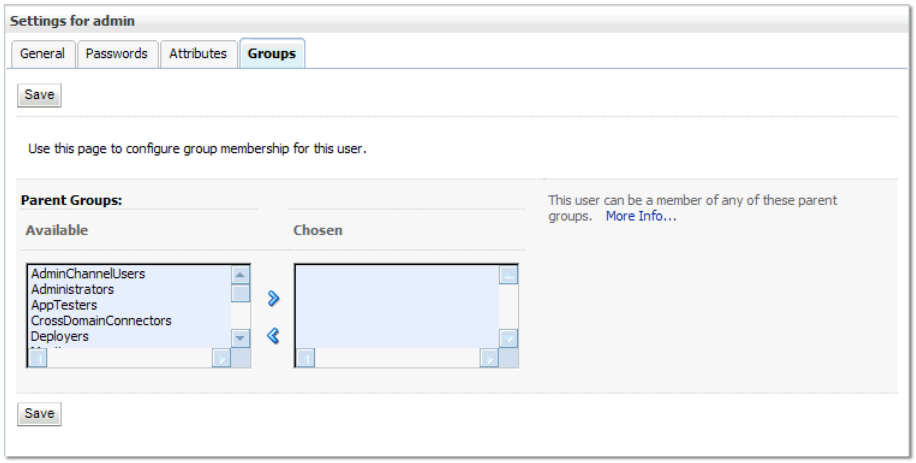
Name	*admin* (This is an example. Enter the name of your choice.)
Description	*Banner Web Services Administrator*
Provider	*DefaultAuthenticator*
Password	Password for the user being created
Confirm Password	Confirmation of the password

12. Click **OK**. The table of users is redisplayed with the new user.

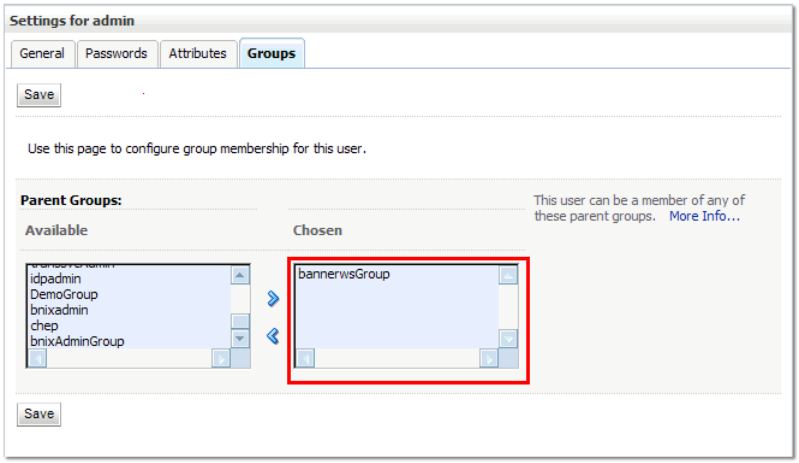


13. Click the name of the user you just created. The Settings page for the user is displayed.

14. Select the **Groups** tab.



15. In the Parent Groups section, select *bannerwsGroup* in the **Available** list and move it to the **Chosen** list.



16. Click **Save**.

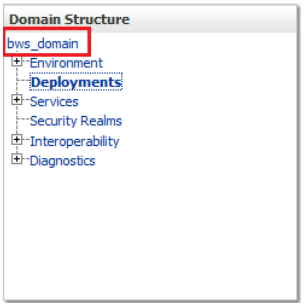
17. In the Change Center pane, click **Activate Changes**.

Step 6 Enable schema validation (optional)

Validating XML request and response messages for each web service invocation degrades system performance. For this reason, schema validation is turned off by default. To enable schema validation, you must set system property `BANNERWS_SCHEMA_VALIDATION` with

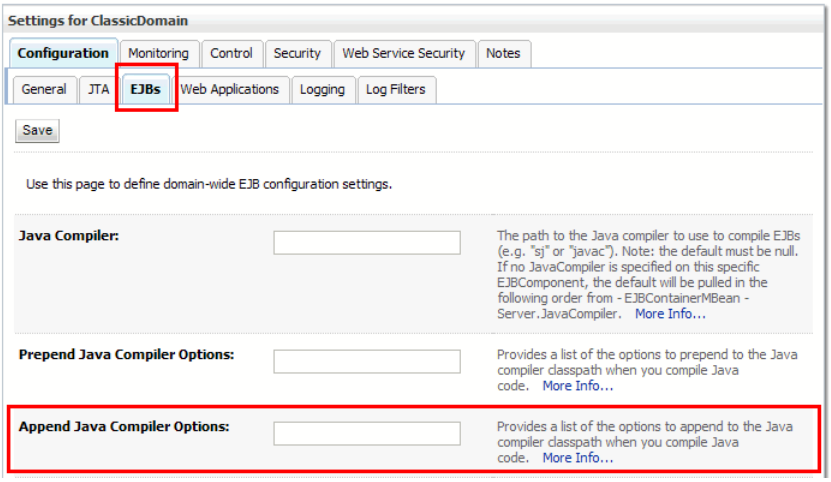
a value of *true* for the instance where the adapter is installed. Use the following steps to enable schema validation.

1. In the Domain Structure pane, click the name of the domain.



   The Settings page is displayed.

2. Select the **EJBs** tab.



3. Add the following value in the **Append Java Compiler Options** field:

   ```
   -DBANNERWS_SCHEMA_VALIDATION=true
   ```

4. Click **Save**.

5. Restart the server for the changes to take effect.

Step 7 Verify the deployment

Use the following steps to verify that the adapter is successfully deployed.

1. Use a web browser to access the URL mapped in following URL:

   ```
   <http or https>://<host>:<port>/<context root>/
   ```

   This URL is used to access an information page, not the endpoint. The context root is *campuscard* or *housing*.

2. Log in with the name and password configured in Step 5, “Configure the security group and user”. An information page for the adapter is displayed. This page shows the version of the adapter and provides a link to the web service’s WSDL.

3. (Optional) If you need to determine the URL that is being used to listen for messages, click the link on the information page to display the associated WSDL. The `<soap:address location>` attribute under the `<service>` tag at the bottom of the WSDL identifies the URL that you should always use to invoke the web service.

WSDL definitions

The following URLs expose the WSDL (Web Services Description Language) files that define the web services exposed by the Banner Web Services Adapters. The protocol, host, port (if used), and URL string reflect the location of the associated adapter.

WSDLs for campus card web services

```
http://<host>:<port>/campuscard/eligibleCardholderService.wsdl
http://<host>:<port>/campuscard/personIdentityService.wsdl
```

WSDLs for housing web services

```
http://<host>:<port>/housing/academicPeriodService.wsdl
http://<host>:<port>/housing/entityAddressService.wsdl
http://<host>:<port>/housing/housingApplicantService.wsdl
http://<host>:<port>/housing/personIdentityService.wsdl
http://<host>:<port>/housing/studentAccountService.wsdl
http://<host>:<port>/housing/studentDepositService.wsdl
```

3 Verify the Configuration

The open source soapUI tool (www.soapui.org) can be used to test exposed web services. A soapUI project is delivered with each Banner® Web Services Adapter to perform the following functions:

- Verify that the Banner Web Services Adapter is deployed and configured correctly.
- Verify that Banner web services have access to the data sources and the Banner Translation Service.
- Detail problems with an incorrectly configured data source (for example, a bad user ID or password for connecting to the Banner database).

These functions are achieved by sending a test SOAP request to a corresponding application that is deployed with the Banner Web Services Adapter, but has a different endpoint URL.

This chapter gives instructions for using soapUI to verify your Banner web services configuration.

Verification steps

Use the following steps to verify the Banner web services configuration:

- Step 1, “Download and install soapUI”
- Step 2, “Open the testing workspace”
- Step 3, “Import the soapUI project”

Step 1 Download and install soapUI

Download soapUI (www.soapui.org) and install it.

Step 2 Open the testing workspace

In soapUI, workspaces contain projects. Projects contain web services definitions. The Banner web services testing workspace is located in the unzipped Banner Web Services download. Use the following steps to open the testing workspace.

1. Open soapUI. A default workspace is displayed in the Navigator.
2. Select Switch Workspace from the File menu. The Switch Workspace window is displayed.

3. Navigate to `BANNER_WEB_SERVICES_814\Deployables\Weblogic\Banner Web Services 8.1.4 Testing-workspace.xml` in the unzipped Banner Web Services download.

4. Click **Open**. The testing workspace is displayed.

Step 3 Import the soapUI project

Use the following steps to import the soapUI project.

1. Select Import Project from the File menu. The Select soapui project file window is displayed.

2. Navigate to the soapUI project in the unzipped Banner Web Services download:

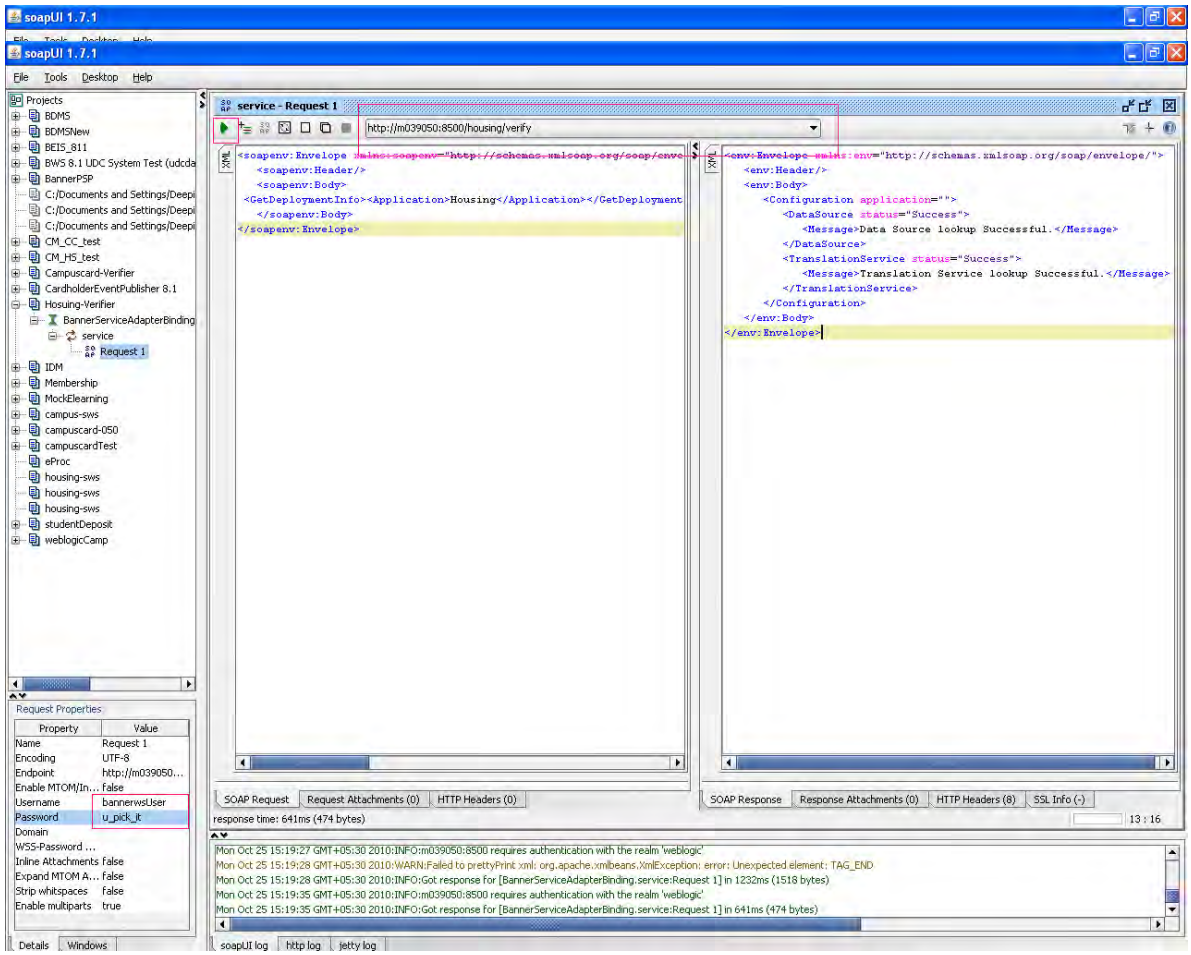
   `BANNER_WEB_SERVICES_814\Deployables\Weblogic\campus_card_adapter\Campuscard-Verifier-soapui-project.xml`
   -or-
   `BANNER_WEB_SERVICES_814\Deployables\Weblogic\housing_adapter\Housing-Verifier-soapui-project.xml`

3. Click **Open**. The workspace for the project is displayed with a single node, **BannerServiceAdapterBinding**.

4. Expand the **BannerServiceAdapter Binding** node and double-click the request tag. A service Request 1 window is displayed.

5. Edit the endpoint URL information and the username/password used to access the URL.

6. Click **Run**. The right pane displays the test results.



Test results

The following sections describe the results that you can get when the configuration is tested with the soapUI tool.

No errors

If the Banner Web Services Adapter for Housing Systems is installed and configured correctly, you get the following response message:

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/
envelope/">
   <env:Header/>
   <env:Body>
```

```
        <Configuration application="Housing">
           <DataSource status="Success">
              <Message>Data Source lookup Successful.</Message>
           </DataSource>
           <TranslationService status="Success">
              <Message>Translation Service lookup Successful.
              </Message>
           </TranslationService>
        </Configuration>
     </env:Body>
  </env:Envelope>
```

If the Banner Web Services Adapter for Campus Card Systems is installed and configured correctly, you get the following response message:

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/
envelope/">
   <env:Header/>
   <env:Body>
      <Configuration application="Campuscard">
         <DataSource status="Success">
            <Message>Data Source lookup Successful.</Message>
         </DataSource>
         <TranslationService status="Success">
            <Message>Translation Service lookup Successful.
            </Message>
         </TranslationService>
      </Configuration>
   </env:Body>
</env:Envelope>
```

Data source configuration errors

If the data source has errors (JNDI lookup is not found or is misconfigured), then the `DataSource status` attribute of the XML response embedded in the response shows `"Failure"`. The following are examples:

```
<Configuration application="Housing">
   <DataSource status="Failure">
      <Message>Error getting Connection from data
       source:DataSource lookup failed. jdbc/bannerws not
      found
      </Message>
```

```
   </DataSource>
   <TranslationService status="Success">
      <Message>Translation Service lookup Successful.</Message>
   </TranslationService>
</Configuration>

<Configuration application="CampusCard">
   <DataSource status="Failure">
      <Message>Error getting Connection from data
       source:DataSource lookup failed. jdbc/bannerws not
       found
      </Message>
   </DataSource>
   <TranslationService status="Success">
      <Message>Translation Service lookup Successful.</Message>
   </TranslationService>
</Configuration>
```

Banner Translation Service configuration errors

If the Banner Translation Service is not found, an error is displayed in the response and the `TranslationService status` attribute shows `"Failure"`. The following are examples:

```
<Configuration application="Housing">
   <DataSource status="Success">
      <Message>Data Source lookup Successful.</Message>
   </DataSource>
   <TranslationService status="Failure">
      <Message>Translation Service lookup failed. Error
      instantiating web-app JNDI-context: No location specified
      and no suitable instance of the type
      com.sct.translation.ejb.local.
      TranslationServiceLocal' found for the ejb-local-ref
      TranslationServiceBean
      </Message>
   </TranslationService>
</Configuration>

<Configuration application="CampusCard">
   <DataSource status="Success">
      <Message>Data Source lookup Successful.</Message>
   </DataSource>
```

```
      <TranslationService status="Failure">
         <Message>Translation Service lookup failed. Error
         instantiating web-app JNDI-context: No location specified
         and no suitable instance of the
         typecom.sct.translation.ejb.local.
         TranslationServiceLocal' found for the ejb-local-ref
         TranslationServiceBean
         </Message>
      </TranslationService>
   </Configuration>
```

To correct this error, undeploy and redeploy the adapter. The Banner Translation Service must be deployed before the adapter is deployed. If installed properly, the Banner Translation Service lookup should not fail.

4 Import Translations

For Banner® web services to function, the Banner Translation Service must be seeded with translations that map Banner codes to enterprise values. This is achieved by importing the translations using a specific XML format.

Delivered translation files

Two types of translation files are delivered with Banner web services:

- .xml files contain enterprise value translations for Banner codes. These translations are unlikely to change and can be imported directly into the Banner Translation Service. These files should *not* be edited before importing.
- .sql files are used to extract Banner support table values for use in translation. These files should be run from SQL*Plus against your Banner instance. Each script spools an output .xml file that contains institution-specific translation values that can be imported into the Banner Translation Service.

Files for campus card systems

The following translation files are located in the `campus_card_adapter/scripts` directory:

```
rco_cardholder_gender.xml
rco_country_code.sql
```

The .sql file is used to create the following .xml file:

.

SQL File Name	Enterprise Value	Banner Value	Output File Name
`rco_country_code.sql`	`STVNATN_NATION`	`STVNATN_CODE`	`rco_country_code.xml`

Files for housing systems

The following translation files are located in the `housing_adapter/scripts` directory:

```
rco_category.xml
rco_gender.xml
dar_deposit_amount_currency.sql
```

```
dar_transaction_amount_currency.sql
dpa_country_code.sql
rar_card_type.sql
rco_country_code.sql
```

The .sql files are used to create the following .xml files:

SQL File Name	Enterprise Value	Banner Value	Output File Name
dar_deposit_amount_currency.sql	GTVCURR_TITLE	GTVCURR_CURR_CODE	dar_deposit_amount_currency_code.xml
dar_transaction_amount_currency.sql	GTVCURR_TITLE	GTVCURR_CURR_CODE	dar_transaction_amount_currency.xml
dpa_country_code.sql	STVNATN_NATION	STVNATN_CODE	dpa_country_code.xml
rar_card_type.sql	GTVCCRD_DESC	GTVCCRD_CODE	rar_card_type.xml
rco_country_code.sql	STVNATN_NATION	STVNATN_CODE	rco_country_code.xml

Import steps

Use the following steps to import enterprise value translations into the Banner Translation Service:

- Step 1, “Extract Banner-specific translation values”.
- Step 2, “Import translation values”.

Step 1 Extract Banner-specific translation values

Run the .sql files from SQL*Plus against your Banner instance. The scripts should be run by an Oracle user with select permissions on the tables where the data values are selected. Each script spools an output .xml file that contains the translation values needed for your Banner instance.

Step 2 Import translation values

Use the following steps to import translation values from each .xml file into the Banner Translation Service. You must import values from the .xml files delivered with Banner web services as well as values from the .xml files created by the delivered .sql scripts.

1. Access the following URL:

   `http://<host>:<port>/transsvc`

   where `<host>` is the server name and `<port>` is the http port number of the server where the Banner Translation Service is installed.

2. Click **LOGIN**.

3. Enter the appropriate username and password.

4. Click **OK**.

5. Click **List** under the Enterprise Field menu.

6. Click **Import** in the Enterprise Fields section. The Load Enterprise Field/s page is displayed.

7. Click **Browse**.

8. Navigate to the file to be imported and select it.

9. Click **Open**. The Load Enterprise Field/s page is redisplayed.

10. Click **Load**. The Import Completed page is displayed.

11. Click **Continue**. An updated list of enterprise field names is displayed.

12. Repeat steps 6 through 11 to import each translation file.

5 Customize Web Service Responses

Underlying APIs contain the business logic for the Banner® web services and provide configuration rules and options. These rules and options are controlled by entries on the following forms:

- Business Rules (GORRSQL) form
- Crosswalk Validation (GTVSDAX) form

Entries on these forms are loaded during the Banner installation and contain data that may or may not be valid for your institution. Entries on GTVSDAX, in particular, must be scrutinized because they are delivered with a generic entry of *UPDATE_ME* instead of values that are valid for your institution.

This chapter provides instructions for validating the GORRSQL and GTVSDAX settings and summarizes the settings that are used by Banner web services. Refer to the *Banner Web Services Handbook* for more details on the settings.

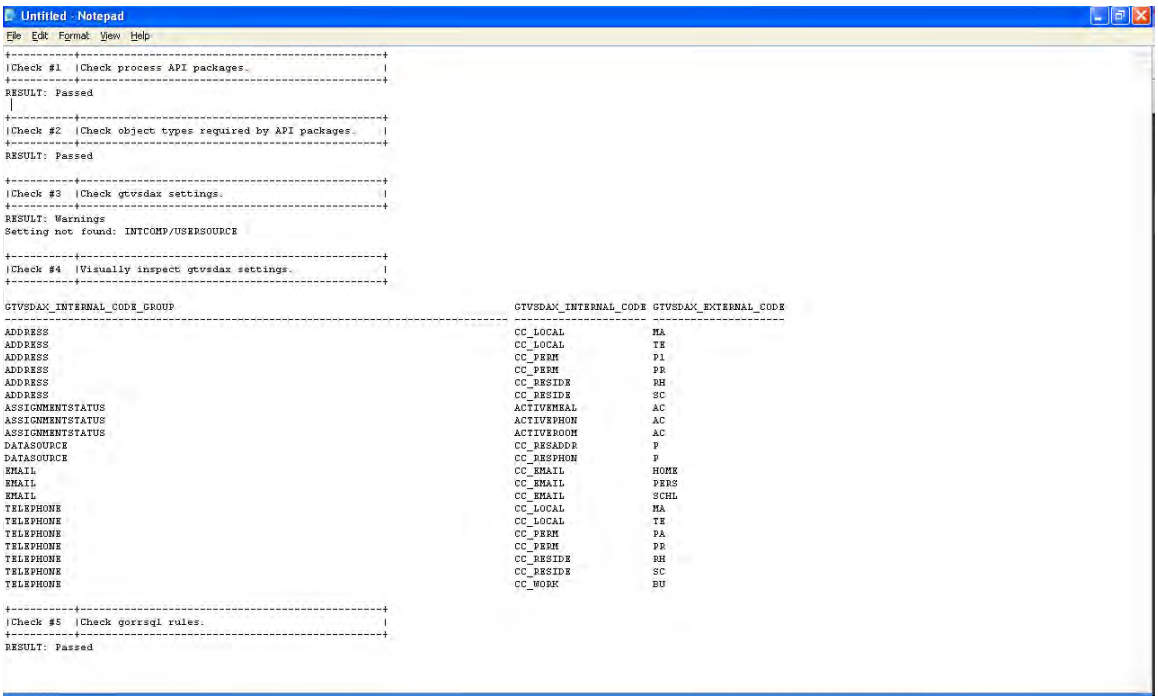
Scripts that check configuration settings

The following SQL scripts in the Banner Web Services download provide a quick glimpse of the configuration parameters required by the Banner Web Services Adapters:

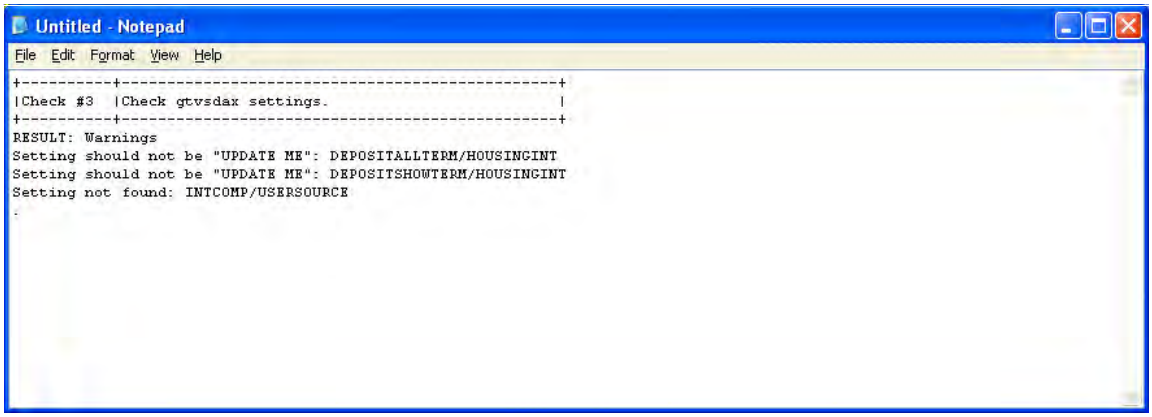
Adapter	Script Name	Script Location
Campus Card Systems	`campuscard_check.sql`	`\campus_card_adapter\scripts`
Housing Systems	`housing_check.sql`	`\housing_adapter\scripts`

The scripts validate the GORRSQL and GTVSDAX settings. The scripts also verify the existence of the required API packages and object types.

Execute the scripts via an SQL*Plus session to the database. The following figure displays a successful result for the web services associated with campus card system integration.



The script displays warnings if settings are missing or are not updated, as shown in the following figure.



GORRSQL rules

The following rules are created on the Business Rules (GORRSQL) form. Refer to the *Banner Web Services Handbook* for more details.

Banner Web Service	GORRSQL Process Code	Description
GetEligibleCardholder and SyncEligibleCardholder	CARDHOLDER_ROLES	Defines the criteria for assigning institution-defined cardholder roles to extracted `Eligible Cardholder` XML objects. Campus card systems can use roles to assign card privileges to specific cardholders.
GetHousingApplicant Eligibility	HOUSING_ELIGIBILITY	Defines the criteria for assigning institution-defined housing applicant roles to extracted `HousingApplicant` XML objects. Housing systems can use roles to determine if an applicant is eligible for specific types of housing (for example, athletic, honors).

GTVSDAX settings

The following settings are entered on the Crosswalk Validation (GTVSDAX) form. Refer to the *Banner Web Services Handbook* for more details.

Banner Web Service	GTVSDAX Internal Code	GTVSDAX Internal Group	Description
AddEntityAddress	INTEG	ADDRTYPE	Address type codes used to create new addresses in Banner
	INTEG	ADDRSRCE	Address source codes used to create new addresses in Banner
GetEligibleCardholder and SyncEligibleCardholder	CC_EMAIL	EMAIL	E-mail address type codes used to select cardholder e-mail addresses
	CC_PERM	ADDRESS	Address type codes used to select cardholder permanent mailing addresses

Banner Web Service	GTVSDAX Internal Code	GTVSDAX Internal Group	Description
	CC_PERM	TELEPHONE	Telephone type codes used to select cardholder permanent telephone numbers
	CC_LOCAL	ADDRESS	Address type codes used to select cardholder local mailing addresses
	CC_LOCAL	TELEPHONE	Telephone type codes used to select cardholder local telephone numbers
	CC_RESADDR	DATASOURCE	Source (SPRADDR or SLRRASG) used to get campus residence locations
	CC_RESIDE	ADDRESS	Address type codes used to select cardholder campus residence addresses (if SPRADDR is used to get campus residence locations)
	ACTIVEROOM	ASSIGNMENTSTATUS	Room assignment status codes used to select active room assignments (if SLRRASG is used to get campus residence locations)
	CC_RESPHON	DATASOURCE	Source (SPRTELE or SLRPASG) used to get campus residence location telephone numbers
	CC_RESIDE	TELEPHONE	Telephone type codes used to select cardholder campus residence telephone numbers (if SPRTELE is used to get campus residence location telephone numbers)
	ACTIVEPHON	ASSIGNMENTSTATUS	Phone assignment status codes used to select active telephone assignments (if SLRPASG is used to get campus residence location telephone numbers)

Banner Web Service	GTVSDAX Internal Code	GTVSDAX Internal Group	Description
	CC_WORK	TELEPHONE	Telephone type codes used to select cardholder work telephone numbers
	ACTIVEMEAL	ASSIGNMENTSTATUS	Meal plan assignment status codes used to identify active meal plan assignments for cardholders
GetHousingApplicant Eligibility	HOUSINGINT	DEPOSITALLTERM	Deposit type codes used to select deposit records for summarization, if the term is not included in the summary
	HOUSINGINT	DEPOSITSHOWTERM	Deposit type codes used to select deposit records for summarization, if the term is included in the summary
	HOUSINGINT	DEPOSITSPECIFICTERM	Deposit type codes used to select deposit records for summarization, if a specific term is specified
	HOUSINGINT	FEEALLTERM	Detail codes used to select housing-related fee records for summarization, if the term is not included in the summary
	HOUSINGINT	FEESHOWTERM	Detail codes used to select housing-related fee records for summarization, if the term is included in the summary
	HOUSINGINT	FEESPECIFICTERM	Detail codes used to select housing-related fee records for summarization, if a specific term is specified
GetHousingApplicant Profile	HOUSINGINT	ADDRESS	Address type codes used to select applicant mailing addresses

Banner Web Service	GTVSDAX Internal Code	GTVSDAX Internal Group	Description
	HOUSINGINT	TELEPHONE	Telephone type codes used to select applicant telephone numbers
	HOUSINGINT	EMAIL	E-mail address type codes used to select applicant e-mail addresses
	HOUSINGINT	SPORT	Activity codes used to select applicant athletic participation information
	HOUSINGINT	ACTIVITY	Activity codes used to select applicant extracurricular activity participation information
GetPersonIdentity	INTEG	CM_SOURCE_CODE	Set of Common Matching rules used to locate persons in Banner if user INTEGMGR (user ID that connects integration software with the Banner database) does not have a default Common Matching source code defined on the Common Matching User Setup (GORCMUS) form

6 Test Banner Web Services

Once the Banner® Web Services Adapters are installed and the appropriate WSDL files are modified, you should test the exposed Banner web services. This step is optional but highly recommended.

The degree of testing depends on the data and tools that your institution uses. For this reason, this section provides testing guidelines rather than specific steps.

Test method

You should use of an interactive testing tool that provides visibility into web service request and response messages. Such a tool helps functional and technical users understand the options available for specific Banner web services and their effect on the content of response messages. Users can create schema-compliant XML messages for their specific environment, quickly inspect the results, modify web service settings in Banner (if necessary), and execute the same request to see the resulting differences.

Several commercial and open source web service testing tools provide these capabilities. One open source tool is eviware soapUI (www.soapui.org). This tool uses a WSDL document and associated XML schema as input to generate compliant request message stubs to which data can be added for calling the associated service. Response messages are displayed in an adjacent window for easy visibility. The tool also allows individual service requests to be tied together to form larger test suites.

Manually creating web service requests and visually inspecting responses provide practical insight into the data returned by the web services under specific conditions. In addition, issues with the underlying Banner configuration for each web service are noticed more readily.

Whatever tool is used, be sure to reference the *Banner Web Services Handbook* to understand the messages, valid input data, and configuration options for each Banner web service that is deployed in your environment.

Web services for campus card systems

The Banner Web Services Adapter for Campus Card Systems supports two Banner web services:

- GetPersonIdentity
- GetEligibleCardholder

The following tests can help you understand these web services. These tests are examples only. They do not reflect actual data in your database.

GetPersonIdentity

This Banner web service finds the unique identifier of a person in Banner for use in subsequent web service calls. It allows for various combinations of person data to limit the result set.

SPRIDEN ID

Test GetPersonIdentity using a known SPRIDEN ID.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetPersonIdentity>
         <urn:UnidentifiedPerson>
            <urn1:PersonIdentifier>
               <urn2:IdValue name="BannerUID">977111370
               </urn2:IdValue>
            </urn1:PersonIdentifier>
         </urn:UnidentifiedPerson>
      </urn:GetPersonIdentity>
   </soapenv:Body>
</soapenv:Envelope>
```

Last name

Test GetPersonIdentity using a known last name.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetPersonIdentity>
         <urn:UnidentifiedPerson>
            <urn1:PersonName>
               <urn2:FamilyName>Mitchell</urn2:FamilyName>
            </urn1:PersonName>
         </urn:UnidentifiedPerson>
      </urn:GetPersonIdentity>
   </soapenv:Body>
</soapenv:Envelope>
```

First name and qualifying data

Test GetPersonIdentity with a first name and qualifying data, such as a phone number.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetPersonIdentity>
         <urn:UnidentifiedPerson>
            <urn1:PersonName>
               <urn2:GivenName>James</urn2:GivenName>
               <urn2:FamilyName>Mitchell</urn2:FamilyName>
            </urn1:PersonName>
            <urn1:ContactPhone>
               <urn2:SubscriberNumber>5551212
               </urn2:SubscriberNumber>
            </urn1:ContactPhone>
         </urn:UnidentifiedPerson>
```

```
      </urn:GetPersonIdentity>
   </soapenv:Body>
</soapenv:Envelope>
```

GetEligibleCardholder

This Banner web service returns all known “eligible cardholder” data for a person based on a given Banner identifier. For a definition of the EligibleCardholder structure and the customization options, refer to the *Banner Web Services Handbook*.

The response messages in the following examples might vary, but the request message is similar, if not the same:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetEligibleCardholder>
         <urn:CardholderIdentifier>
            <urn1:IdValue name="BannerUID">977111370
            </urn1:IdValue>
         </urn:CardholderIdentifier>
      </urn:GetEligibleCardholder>
   </soapenv:Body>
</soapenv:Envelope>
```

SPRIDEN ID

Test GetEligibleCardholder for a basic person using a known SPRIDEN ID.

Delivered roles

Test delivered roles:

1. Ensure that delivered cardholder roles are enabled on the Business Rules (GORRSQL) form.
2. Test GetEligibleCardholder for a person who should have one of the roles.
3. Ensure that the ShowEligibleCardholder message includes the role.

New cardholder role

Test a new cardholder role:

1. Create a new cardholder role on GORRSQL for criteria that can be easily applied to a person (for example, a specific e-mail address domain or a specific address).
2. Test GetEligibleCardholder for a person that matches the cardholder role criteria.
3. Ensure that the ShowEligibleCardholder message includes the role.

Changed GTVSDAX settings

Test changed GTVSDAX settings:

1. Change settings on the Crosswalk Validation (GTVSDAX) form for cardholder contact details, residence location, work location, or meal plan assignment status.
2. Retest GetEligibleCardholder for a known person.
3. Ensure that the ShowEligibleCardholder messages match expected results.

Web services for housing systems

The Banner Web Services Adapter for Housing Systems supports the following Banner web services:

- GetPersonIdentity
- GetHousingApplicantEligibility
- GetHousingApplicantProfile
- GetAcademicPeriods
- AddEntityAddress
- ExpireEntityAddress
- AddStudentDeposit
- ReleaseStudentDeposit
- AddStudentAccountTransaction

The following tests can help you understand the web services. These tests are examples only. They do not reflect actual data in your database.

GetPersonIdentity

This Banner web service finds the unique identifier of a person in Banner for use in subsequent web service calls. It allows for various combinations of person data to limit the result set.

GetPersonIdentity is exposed by both the Banner Web Services Adapter for Campus Card Systems and the Banner Web Services Adapter for Housing Systems. Therefore, the tests are identical. See "GetPersonIdentity" on page 58 for testing guidelines.

GetHousingApplicantEligibility

This Banner web service returns "eligibility" data for a person that might be applying for housing at the institution. For a definition of the HousingApplicantEligibility structure and the customization options, refer to the *Banner Web Services Handbook*.

The response messages in the following examples might vary, but the request message is similar, if not the same:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:residentiallife:1.0
" xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetHousingApplicantEligibility>
         <urn:SearchCriteria>
            <urn1:HousingApplicantIdentifier>
               <urn2:IdValue name="BannerUID">A00012649
               </urn2:IdValue>
            </urn1:HousingApplicantIdentifier>
            <urn1:AcademicPeriodIdentifier type="Term"
            id="200711"/>
         </urn:SearchCriteria>
      </urn:GetHousingApplicantEligibility>
   </soapenv:Body>
</soapenv:Envelope>
```

SPRIDEN ID and term code

Test GetHousingApplicantEligibility for a student using a known SPRIDEN ID and term code.

Delivered roles

Test delivered roles:

1. Ensure that delivered housing applicant roles are enabled on the Business Rules (GORRSQL) form.
2. Test GetHousingApplicantEligibility for a person who should have one of the roles.
3. Ensure that the ShowHousingApplicantEligibility message includes the role.

Deposit information

Test deposit information:

1. Review the section on Deposit Information in chapter 8 of the *Banner Web Services Handbook*.
2. Update settings on the Crosswalk Validation (GTVSDAX) form for the service with valid deposit type codes from the Deposit Type Code Validation (TTVDTYP) form for each deposit summarization.
3. Create deposits for a student or find a student with deposits.
4. Retest GetHousingApplicantEligibility using the student's SPRIDEN ID.
5. Ensure that the ShowHousingApplicantEligibility message includes the appropriate deposit summarization.

Fee information

Test fee information:

1. Review the section on Fee Information in chapter 8 of the *Banner Web Services Handbook*.
2. Update settings on GTVSDAX for the service with valid detail codes from the Student Account Detail (TSADETL) form for each fee summarization.
3. Create fees for a student or find a student with fees.
4. Retest GetHousingApplicantEligibility using the student's SPRIDEN ID.
5. Ensure that the ShowHousingApplicantEligibility message includes the appropriate fee summarization.

Holds

Test holds:

1. Create holds for a student.
2. Retest GetHousingApplicantEligibility using the student's SPRIDEN ID.
3. Ensure that the ShowHousingApplicantEligibility message includes the holds.

GetHousingApplicantProfile

This Banner web service returns detailed information about a housing applicant. For a definition of the `HousingApplicantProfile` structure and the customization options, refer to the *Banner Web Services Handbook*.

The response messages in the following examples might vary, but the request message is similar, if not the same:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:residentiallife:1.0
" xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetHousingApplicantProfile>
         <urn:SearchCriteria>
            <urn1:HousingApplicantIdentifier>
               <urn2:IdValue name="BannerUID">977111370
               </urn2:IdValue>
            </urn1:HousingApplicantIdentifier>
            <urn1:AcademicPeriodIdentifier type="Term"
            id="200711"/>
         </urn:SearchCriteria>
      </urn:GetHousingApplicantProfile>
   </soapenv:Body>
</soapenv:Envelope>
```

SPRIDEN ID and term code

Test GetHousingApplicantProfile for a student using a known SPRIDEN ID and term code.

Expanded profile

Add information to the student to expand the profile and retest. Information that can be added and verified includes privacy information, biographic data, demographic data, military service, driver's license information, mailing addresses, medical records, athletic participation, and extracurricular activities. Refer to the *Banner Web Services Handbook* for the HousingApplicantProfile structure and more information on its content.

Changed GTVSDAX settings

Test changed GTVSDAX settings:

1. Change settings on the Crosswalk Validation (GTVSDAX) form for housing applicant contact details and participation information.
2. Retest GetHousingApplicantProfile for a known person.
3. Ensure that response messages match expected results.

GetAcademicPeriods

This Banner web service returns information about a Banner term. This information is helpful for applications that need to create student account charges or search for information based on a term.

All terms

Test GetAcademicPeriods for all terms.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:student:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetAcademicPeriods>
         <urn:SearchCriteria>
            <urn1:TypeOfAcademicPeriod>Term
            </urn1:TypeOfAcademicPeriod>
         </urn:SearchCriteria>
      </urn:GetAcademicPeriods>
   </soapenv:Body>
</soapenv:Envelope>
```

Terms for a specific date

Test GetAcademicPeriods for terms for a specific date.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:student:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:GetAcademicPeriods>
         <urn:SearchCriteria>
            <urn1:TypeOfAcademicPeriod>Term
            </urn1:TypeOfAcademicPeriod>
            <urn1:SearchDate>2005-04-15</urn1:SearchDate>
         </urn:SearchCriteria>
      </urn:GetAcademicPeriods>
   </soapenv:Body>
</soapenv:Envelope>
```

AddEntityAddress

This Banner web service creates a SPRADDR record in Banner. The `EntityAddress` structure follows an industry standard specification; not all elements are supported by Banner. Refer to the *Banner Web Services Handbook* for a list of supported elements.

Known person

Test creation of an address for a known person in Banner.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:AddEntityAddress>
         <urn:AddressSource>SOAPUI</urn:AddressSource>
         <urn:EntityAddress type="RESIADDRESS">
            <urn1:AddressIdentifier>
               <urn2:IdValue name="BannerUID">A00012649
               </urn2:IdValue>
```

```
            </urn1:AddressIdentifier>
            <urn1:PostalCode>68502</urn1:PostalCode>
            <urn1:Region>NE</urn1:Region>
            <urn1:Municipality>Lincoln</urn1:Municipality>
            <urn1:DeliveryAddress>
               <urn2:AddressLine>Johnson Tower
               </urn2:AddressLine>
               <urn2:AddressLine>Suite 432</urn2:AddressLine>
            </urn1:DeliveryAddress>
         </urn:EntityAddress>
      </urn:AddEntityAddress>
   </soapenv:Body>
</soapenv:Envelope>
```

Temporary address

Test creation of a temporary address with effective and expiration dates.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:AddEntityAddress>
         <urn:AddressSource>SOAPUI</urn:AddressSource>
         <urn:EntityAddress type="RESIADDRESS"
         validFrom="2007-12-01"
         validTo="2008-05-31">
            <urn1:AddressIdentifier>
               <urn2:IdValue name="BannerUID">A00012720
               </urn2:IdValue>
            </urn1:AddressIdentifier>
            <urn1:PostalCode>29208</urn1:PostalCode>
            <urn1:Region>SC</urn1:Region>
            <urn1:Municipality>Columbia</urn1:Municipality>
            <urn1:DeliveryAddress>
               <urn2:AddressLine>Bates Tower</urn2:AddressLine>
               <urn2:AddressLine>Suite 432</urn2:AddressLine>
            </urn1:DeliveryAddress>
         </urn:EntityAddress>
      </urn:AddEntityAddress>
```

```
   </soapenv:Body>
</soapenv:Envelope>
```

ExpireEntityAddress

This Banner web service expires a SPRADDR record in Banner by updating the address end date. The web service requires the unique ID of the address, which is returned by the AddEntityAddress web service.

Known person

Test expiration of an address for a known person in Banner.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:ExpireEntityAddress>
         <urn:EntityAddressExpirationRequest>
            <urn1:EntityAddressIdentifier>
               <urn1:AddressIdentifier>
                  <!--1 or more repetitions:-->
                  <urn2:IdValue name="BannerUID">A00012649
                  </urn2:IdValue>
                  <urn2:IdValue name="AddressType">SC
                  </urn2:IdValue>
                  <urn2:IdValue name="SequenceNumber">1
                  </urn2:IdValue>
               </urn1:AddressIdentifier>
            </urn1:EntityAddressIdentifier>
            <urn1:ExpirationDate>2007-12-13
            </urn1:ExpirationDate>
         </urn:EntityAddressExpirationRequest>
      </urn:ExpireEntityAddress>
   </soapenv:Body>
</soapenv:Envelope>
```

Start date and expiration date are the same

Test expiration of an address with the same start date and expiration date.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:party:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:ExpireEntityAddress>
         <urn:EntityAddressExpirationRequest>
            <urn1:EntityAddressIdentifier>
               <urn1:AddressIdentifier>
                  <!--1 or more repetitions:-->
                  <urn2:IdValue name="BannerUID">A00012720
                  </urn2:IdValue>
                  <urn2:IdValue name="AddressType">SC
                  </urn2:IdValue>
                  <urn2:IdValue name="SequenceNumber">2
                  </urn2:IdValue>
               </urn1:AddressIdentifier>
            </urn1:EntityAddressIdentifier>
            <urn1:ExpirationDate>2007-12-01
            </urn1:ExpirationDate>
         </urn:EntityAddressExpirationRequest>
      </urn:ExpireEntityAddress>
   </soapenv:Body>
</soapenv:Envelope>
```

AddStudentDeposit

This Banner web service creates deposits in Banner for a given student. Deposits are reflected in the GetHousingApplicantEligibility web service. Therefore, GetHousingApplicantEligibility can be used to validate the creation of deposits via AddStudentDeposit. For details on the service's options, refer to the *Banner Web Services Handbook*.

New deposit

Test the creation of a deposit for a student.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:ar:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0"
xmlns:urn3="urn:sungardhe:enterprise:resources:ar:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:AddStudentDeposit>
         <urn:StudentDeposit>
            <urn1:AccountIdentifier>
               <urn2:IdValue name="BannerUID">IT121405G
               </urn2:IdValue>
            </urn1:AccountIdentifier>
            <urn1:AcademicPeriodIdentifier type="Term"
            id="200510"/>
            <urn1:DepositTransactionType id="GDD1"/>
            <urn1:TransactionDescription>Housing Reservation
            Deposit
            </urn1:TransactionDescription>
            <urn1:DepositPaymentTransactionType id="GDP1"/>
            <urn1:DepositAmount>500.00</urn1:DepositAmount>
            <urn1:EffectiveDate>2007-04-16</urn1:EffectiveDate>
            <urn1:ReleaseInstructions>
               <urn3:AutoRelease>Yes</urn3:AutoRelease>
               <urn3:DepositReleaseDate>2007-08-29
               </urn3:DepositReleaseDate>
               <urn3:DepositBalanceMinimumAmount>250.00
               </urn3:DepositBalanceMinimumAmount>
               <urn3:DepositExpirationDate>2007-08-29
               </urn3:DepositExpirationDate>
            </urn1:ReleaseInstructions>
            <urn1:CashierSession>
               <urn3:CashierId>WSTEST</urn3:CashierId>
               <urn3:SessionNumber>0</urn3:SessionNumber>
            </urn1:CashierSession>
            <urn1:DepositTransactionSource>
               <urn3:SourceSystem>soapUI</urn3:SourceSystem>
            </urn1:DepositTransactionSource>
            <urn1:DocumentNumber>MMB001</urn1:DocumentNumber>
```

```
            <urn1:OverrideHoldChecking>Yes
            </urn1:OverrideHoldChecking>
         </urn:StudentDeposit>
      </urn:AddStudentDeposit>
   </soapenv:Body>
</soapenv:Envelope>
```

Additional deposits

Create additional deposits to test various release options.

ReleaseStudentDeposit

This Banner web service is the counterpart to AddStudentDeposit. It allows external systems to request Banner to release funds held on deposit on a student's account. These funds might be released to cover damages charged to a student's account, or they might be remitted to the student at the end of a specified period. This service can be tested in conjunction with AddStudentDeposit and GetHousingApplicantEligibility. For details on the service's options, refer to the *Banner Web Services Handbook*.

Release of deposits

Request release of deposits of a specific type within a specific term.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:ar:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0"
xmlns:urn3="urn:sungardhe:enterprise:resources:ar:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:ReleaseStudentDeposit>
         <urn:StudentDepositReleaseRequest>
            <urn1:AccountIdentifier>
               <urn2:IdValue name="BannerUID">A00012649
               </urn2:IdValue>
            </urn1:AccountIdentifier>
            <urn1:CashierSession>
               <urn3:CashierId>SYSTEST24</urn3:CashierId>
               <urn3:SessionNumber>0</urn3:SessionNumber>
            </urn1:CashierSession>
            <urn3:ForceRelease>No</urn3:ForceRelease>
```

```
                <urn3:DepositReleaseTransactionType>%
                </urn3:DepositReleaseTransactionType>
                <urn3:AcademicPeriodIdentifier type="Term"
                id="200711"/>
                <urn3:DepositType>HOU</urn3:DepositType>
             </urn:StudentDepositReleaseRequest>
          </urn:ReleaseStudentDeposit>
   </soapenv:Body>
</soapenv:Envelope>
```

Forced release

Force release of all deposits.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:ar:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0"
xmlns:urn3="urn:sungardhe:enterprise:resources:ar:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:ReleaseStudentDeposit>
         <urn:StudentDepositReleaseRequest>
            <urn1:AccountIdentifier>
               <urn2:IdValue name="BannerUID">IT121405G
               </urn2:IdValue>
            </urn1:AccountIdentifier>
            <urn1:CashierSession>
               <urn3:CashierId>WSTEST</urn3:CashierId>
               <urn3:SessionNumber>0</urn3:SessionNumber>
            </urn1:CashierSession>
            <urn3:ForceRelease>Yes</urn3:ForceRelease>
            <urn3:DepositReleaseTransactionType>%
            </urn3:DepositReleaseTransactionType>
            <urn3:AcademicPeriodIdentifier id="Term"/>
            <urn3:DepositType>AP2</urn3:DepositType>
         </urn:StudentDepositReleaseRequest>
      </urn:ReleaseStudentDeposit>
   </soapenv:Body>
</soapenv:Envelope>
```

AddStudentAccountTransaction

This Banner web service and its siblings, AddStudentAcctTransSource and AddStudentAcctTransSystem, allow external systems to request that a charge or payment be recorded on a student account in Banner. Refer to the *Banner Web Services Handbook* for an explanation of differences among these services.

New charge

Test the creation of a charge, and verify the creation of the TBRACCD record in Banner.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:ar:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0"
xmlns:urn3="urn:sungardhe:enterprise:resources:ar:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:AddStudentAccountTransaction>
         <urn:StudentAccountTransaction>
            <urn1:AccountIdentifier>
               <urn2:IdValue name="BannerUID">IT121405G
               </urn2:IdValue>
            </urn1:AccountIdentifier>
            <urn1:AcademicPeriodIdentifier type="Term"
            id="200510"/>
            <urn1:TransactionType id="TECH"/>
            <urn1:TransactionSource>
               <urn3:SourceSystem>MMB001</urn3:SourceSystem>
            </urn1:TransactionSource>
            <urn1:TransactionAmount>50.00
            </urn1:TransactionAmount>
            <urn1:TransactionDescription>Technology Fee
            </urn1:TransactionDescription>
            <urn1:EffectiveDate>2007-10-16</urn1:EffectiveDate>
            <urn1:CashierSession>
               <urn3:CashierId>WSTEST</urn3:CashierId>
               <urn3:SessionNumber>0</urn3:SessionNumber>
            </urn1:CashierSession>
         </urn:StudentAccountTransaction>
      </urn:AddStudentAccountTransaction>
   </soapenv:Body>
</soapenv:Envelope>
```

New payment

Test the creation of a payment and verify the creation of the TBRACCD record in Banner.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:ws:messages:1.0"
xmlns:urn1="urn:sungardhe:enterprise:domain:ar:1.0"
xmlns:urn2="urn:sungardhe:enterprise:resources:common:1.0"
xmlns:urn3="urn:sungardhe:enterprise:resources:ar:1.0">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:AddStudentAccountTransaction>
         <urn:StudentAccountTransaction>
            <urn1:AccountIdentifier>
               <urn2:IdValue name="BannerUID">IT121405G
               </urn2:IdValue>
            </urn1:AccountIdentifier>
            <urn1:AcademicPeriodIdentifier type="Term"
            id="200510"/>
            <urn1:TransactionType id="GZP1"/>
            <urn1:TransactionSource>
               <urn3:SourceSystem>MMB001</urn3:SourceSystem>
            </urn1:TransactionSource>
            <urn1:TransactionAmount>125.56
            </urn1:TransactionAmount>
            <urn1:TransactionDescription>Payment by Check
            </urn1:TransactionDescription>
            <urn1:CashierSession>
               <urn3:CashierId>WSTEST</urn3:CashierId>
               <urn3:SessionNumber>0</urn3:SessionNumber>
            </urn1:CashierSession>
            <urn1:PaymentDetails>
               <urn3:PaymentID>206</urn3:PaymentID>
            </urn1:PaymentDetails>
         </urn:StudentAccountTransaction>
      </urn:AddStudentAccountTransaction>
   </soapenv:Body>
</soapenv:Envelope>
```

7 Install Banner Cardholder Event Publisher

The Banner® Cardholder Event Publisher pushes cardholder data changes to a defined external campus card system. The Publisher works with defined Oracle Streams capture and apply processes and Oracle Advanced Queuing to recognize cardholder data changes in Banner and to publish corresponding SyncEligibleCardholder messages. The Publisher must be installed and configured for the external system to receive messages.

This chapter gives instructions for installing the Banner Cardholder Event Publisher on Oracle WebLogic Server 11g.

Requirements

The Banner Cardholder Event Publisher has the following requirements.

External campus card system

The external card system that receives cardholder data changes from the Publisher must meet the following criteria:

- Implement the SyncEligibleCardholder web service interface (see the *Banner Web Services Handbook*)
- Implement SOAP binding
- Expose a compliant endpoint

Oracle application server and Java

The Banner Cardholder Event Publisher is certified on Oracle WebLogic Server 11g with Java 1.7.

Oracle database

The Oracle Database 11g is required.

Banner Translation Service

You *must* install the Banner Translation Service before you deploy the Banner Cardholder Event Publisher. Refer to the *Banner Translation Service Installation and Administration Guide* for details.

Banner dependency

The Banner Cardholder Event Publisher capture rules should already be loaded in the database as a result of seed data scripts for the GTVSQRUI, GTVSQPRI, GORCTAB, GORCRUL, and GORCCOL tables. Likewise, the cardholder apply handler API should already be applied as a result of the creation of the `gp_cardholder` package. In addition, patch p1-46c8mj_gen80100 must be installed. This patch addresses issues with the publication of international data and the mass publication of messages for initial and periodic synchronization of cardholder data.

Recommended configuration

A new Managed Server was created when the Banner Translation Service was installed. The Publisher must be deployed in this same Managed Server with the Banner Translation Service and the Banner Web Service Adapter for Campus Card Systems.

Installation steps

The Banner Cardholder Event Publisher is packaged as a J2EE compatible enterprise archive file named `CardholderEventPublisher_v8.1.4.ear`. Use the following steps to install the Publisher on Oracle WebLogic Server 11g:

- Step 1, “Verify the capture process rules”.
- Step 2, “Create, configure, and start the Oracle Streams processes”.
- Step 3, “Configure the Oracle WebLogic Server”
- Step 4, “Define the data source for Oracle Advanced Queuing”
- Step 5, “Define the data source for the bulk load process”
- Step 6, “Define the data source for the Oracle Streams administrator”
- Step 7, “Install the Publisher”
- Step 8, “Configure the security group and user”

Step 1 Verify the capture process rules

The capture rules for cardholder data should already be loaded in the database. Capture rules are initially provided in Banner General seed data scripts, as described in "Banner dependency" on page 76. Additional capture rules are provided with Banner Web Services download in the following directory:

```
\cardholder_event_publisher\scripts\capture_rules
```

Use the steps in Appendix B, "Using Oracle Streams," of the *Banner Web Services Handbook* to verify that the rules are loaded in the database.

Step 2 Create, configure, and start the Oracle Streams processes

Use the steps in Appendix B, "Using Oracle Streams," of the *Banner Web Services Handbook* to create, configure, and start the Oracle Streams capture and apply processes. These steps accomplish the following:

- Create buffered queues and queue tables to manage events.
- Create supplemental, primary key, and unique key log groups for the configured tables.
- Configure the DML callback handler for the apply process.
- Set the instantiation SCN for the tables in the apply process.

Note

These steps should be performed as the `streamsadmin` Oracle user only. ■

Refer to the *Banner Web Services Handbook* for more information about using Oracle Streams.

Step 3 Configure the Oracle WebLogic Server

The Oracle WebLogic Server must be configured to use the *Advanced* security model instead of the default *DD only* option.

Note

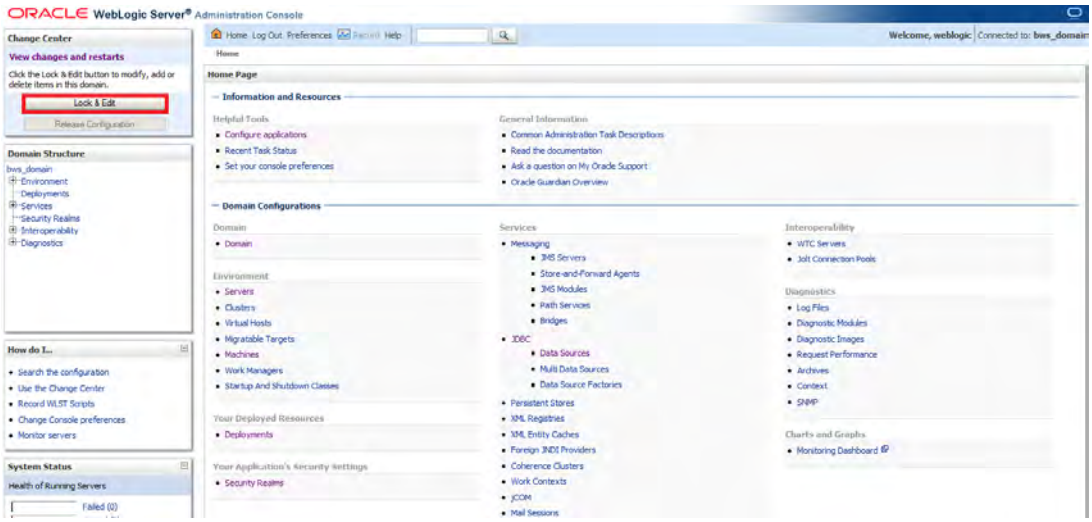
The Oracle WebLogic Server needs to be configured only once. If the server was previously configured, you can skip this step. ■

Use the following steps to configure the server.

1. Connect to the Oracle WebLogic Server Administration Console:

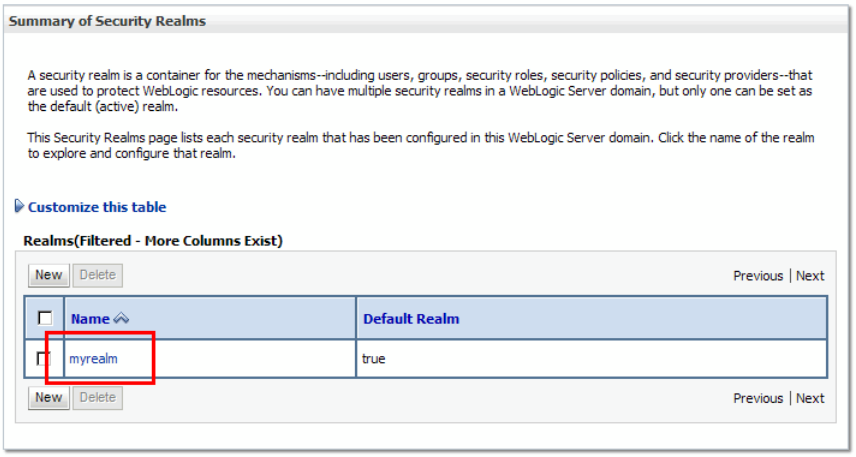
   `http://<host>:<port>/console`

   The Home Page is displayed.

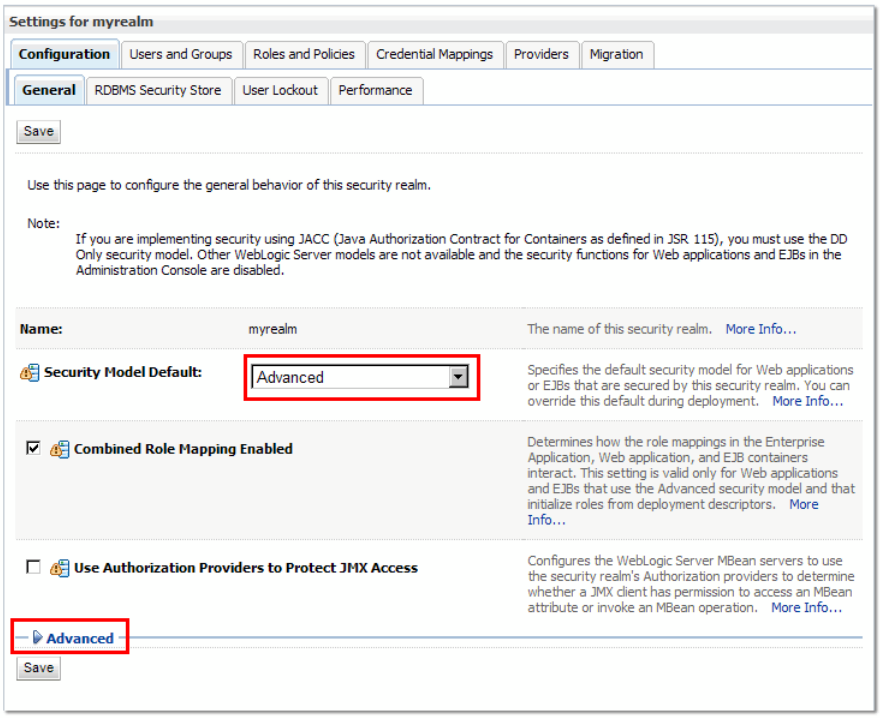


2. In the Domain Structure pane, click **Security Realms**.

The Summary of Security Realms page is displayed.

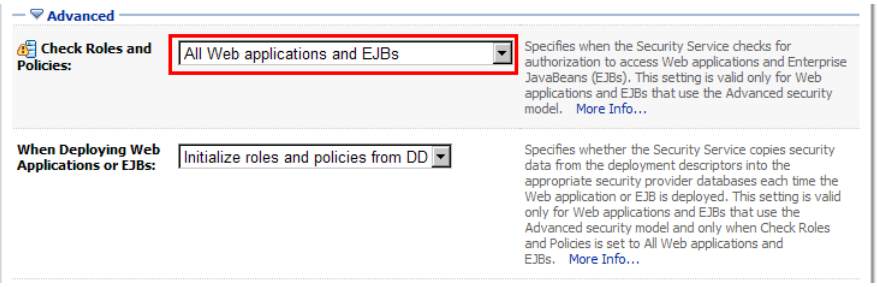


3. Click **myrealm**. The Settings for myrealm page is displayed.



4. Select *Advanced* in the **Security Model Default** drop-down list.

5. Click the **Advanced** link to display the advanced options.



6. Select *All Web Applications and EJBs* in the **Check Roles and Policies** drop-down list.

7. Click **Save**.

8. Restart the server for the changes to take effect.

Step 4 Define the data source for Oracle Advanced Queuing

A data source provides the connection properties to the Banner database. A data source must be defined for connecting to Oracle Advanced Queueing to consume Banner identity messages.

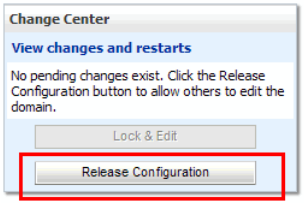
If the Publisher will be deployed in the same instance with the other Banner Web Services Adapters, the Publisher can use the same data source that was previously defined for the adapters. This is the recommended installation.

If the Publisher will be deployed in a different instance, use the steps in "Define the data source" on page 20 to define the data source for Oracle Advanced Queueing.

Step 5 Define the data source for the bulk load process

Use the following steps to define the data source for accessing the database schema that has access to invoke the bulk load process.

1. In the Change Center pane, click **Release Configuration**.

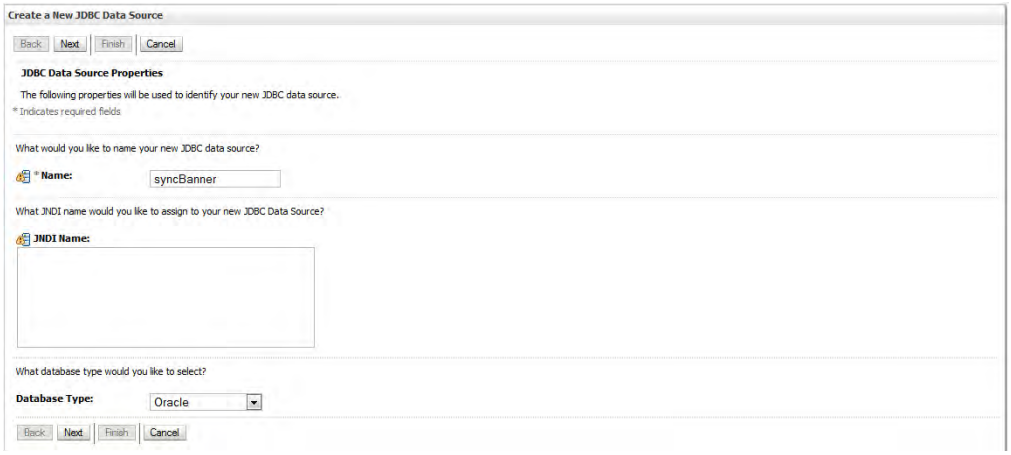


2. In the Domain Structure pane, expand and click **Services > JDBC > Data Sources**.

The Summary of JDBC Data Sources page is displayed.

3. In the Change Center pane, click **Lock & Edit**.

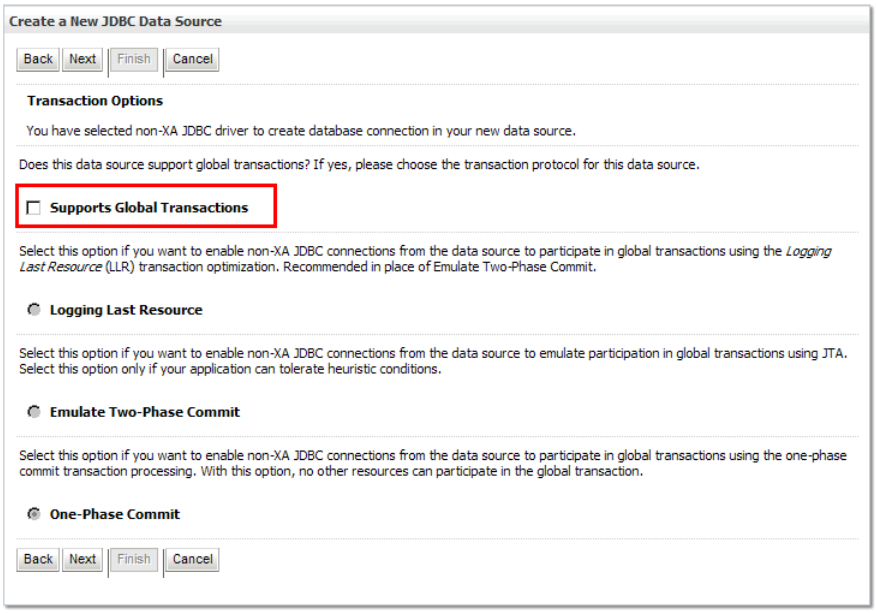
4. On the Summary of JDBC Data Sources page, click **New**. The Create a New JDBC Data Source page is displayed.



5. Enter the following data source properties:

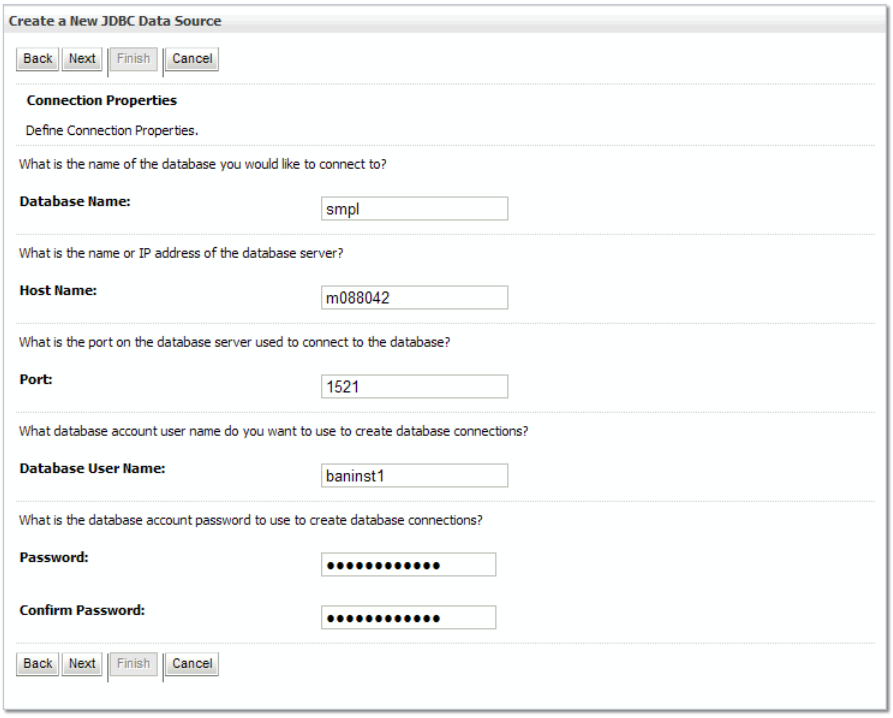
Name	*syncBanner*
JNDI Name	*jdbc/syncbanner*
Database Type	*Oracle*
Database Driver	Appropriate database driver that is used to create database connections Select *Oracle's Driver (Thin) for Instance connections; Versions:9.0.1, 9.2.0,10,11*

6. Click **Next**. The next page is displayed.



7. Clear the **Supports Global Transactions** check box.

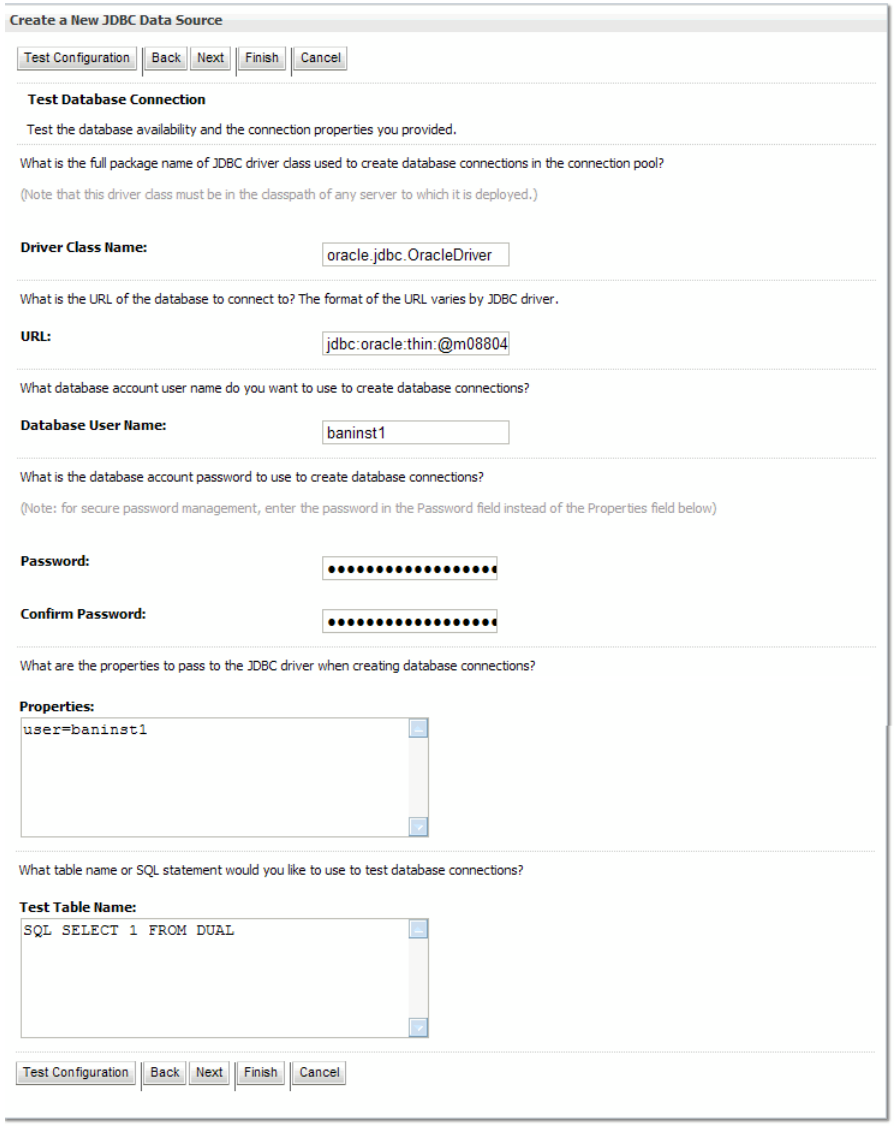
8. Click **Next**. The next page is displayed.



9. Enter the following connection properties:

Database Name	Name of the database to which you are connecting
Host Name	IP address of the database server
Port	Port on the database server that is used to connect to the database
Database User Name	*baninst1*
Password	Password for the `baninst1` user
Confirm Password	Confirmation of the password

10. Click **Next**. The next page is displayed with the properties that you entered.



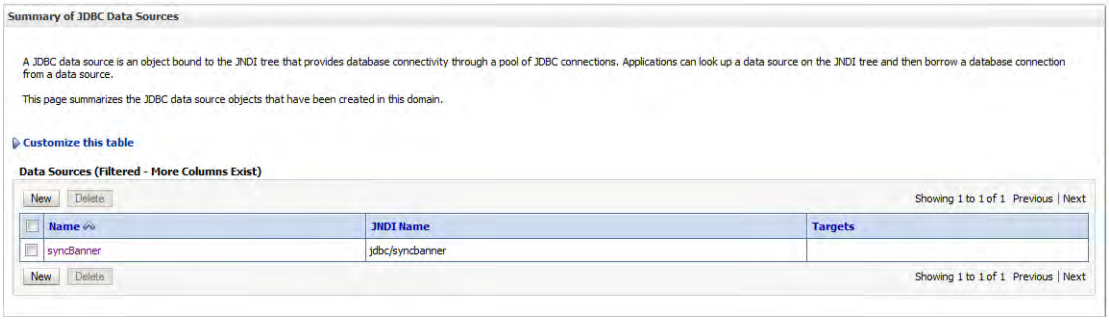
11. Verify the property values.

12. Click **Test Configuration**. The page is redisplayed with a success or failure message.

    12.1. If the test succeeds, continue with the next step.

    12.2. If the test fails, ensure that the connection URL and credentials are correct. Continue testing until the connection is successful.

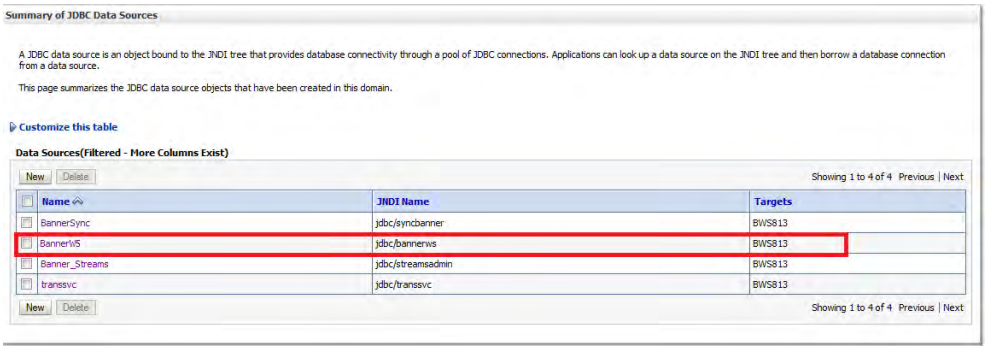
13. Click **Finish**. The Summary of JDBC Data Sources page is displayed with the new data source.



14. In the Change Center pane, click **Activate Changes**.

15. On the Summary of JDBC Data Sources page, click the name of the new data source. The Settings for syncBanner page is displayed.

16. Select the Targets tab.



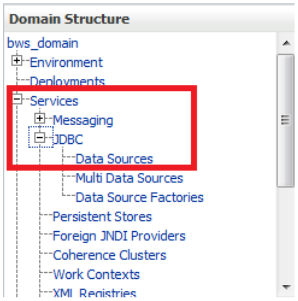
17. In the Change Center pane, click **Lock & Edit**.

18. On the Settings for syncBanner page, select the server where the data source should be deployed.

19. Click **Save**.

20. In the Change Center pane, click **Activate Changes.**

21. In the Domain Structure pane, expand and click **Services** > **JDBC** > **Data Sources**.



The Summary of JDBC Data Sources page is displayed.

22. Verify that the new data source is associated with the server.

Step 6 Define the data source for the Oracle Streams administrator

Use the following steps to define the data source for connecting to the Oracle database for administering Oracle Streams.

1. In the Change Center pane, click **Lock & Edit**.

2. Ensure that the Summary of JDBC Data Sources page is displayed. (If it is not displayed, expand and click **Services** > **JDBC** > **Data Sources** in the Domain Structure pane.)

3. Click **New** on the Summary of JDBC Data Sources page. The Create a New JDBC Data Source page is displayed.

4. Enter the following data source properties:

Name	*Banner_streamsadmin*
JNDI Name	*jdbc/streamsadmin*
Database Type	*Oracle*
Database Driver	Appropriate database driver that is used to create database connections

5. Click **Next**. The next page is displayed.

6. Clear the **Supports Global Transactions** check box.

7. Click **Next**. The next page is displayed.

8. Enter the following connection properties:

Database Name	Name of the database to which you are connecting
Host Name	IP address of the database server
Port	Port on the database server that is used to connect to the database
Database User Name	*streamsadmin*
Password	Password for the `streamsadmin` user
Confirm Password	Confirmation of the password

9. Click **Next**. The next page is displayed with the properties that you entered.
10. Verify the property values.
11. Click **Test Configuration**. The page is redisplayed with a success or failure message.
    - 11.1. If the test succeeds, continue with the next step.
    - 11.2. If the test fails, ensure that the connection URL and credentials are correct. Continue testing until the connection is successful.
12. Click **Finish**. The Summary of JDBC Data Sources page is displayed with the new data source.
13. In the Change Center pane, click **Activate Changes**.
14. On the Summary of JDBC Data Sources page, click the name of the new data source. The Settings for streamsadmin page is displayed.
15. Select the Targets tab.
16. In the Change Center pane, click **Lock & Edit**.
17. On the Settings for streamsadmin page, select the server where the data source should be deployed.
18. Click **Save**.
19. In the Change Center pane, click **Activate Changes.**

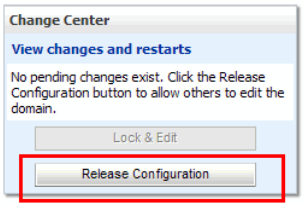
20. In the Domain Structure pane, click **Services > JDBC > Data Sources**. The Summary of JDBC Data Sources page is displayed.

21. Verify that the new data source is associated with the server.

Step 7 Install the Publisher

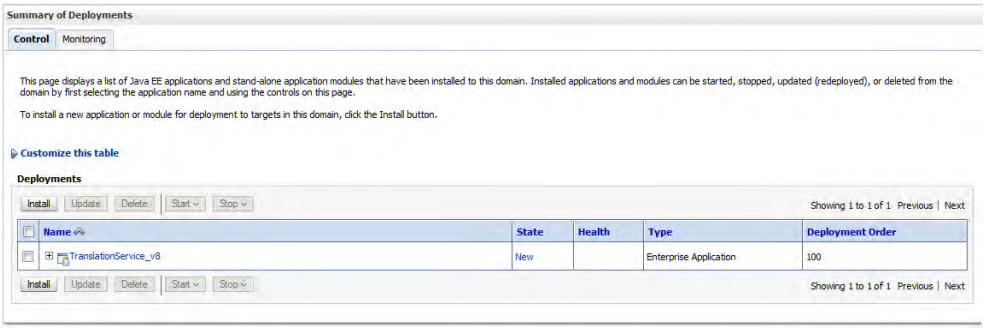
Use the following steps to install the Banner Cardholder Event Publisher to the Oracle WebLogic Server.

1. In the Change Center pane, click **Release Configuration**.



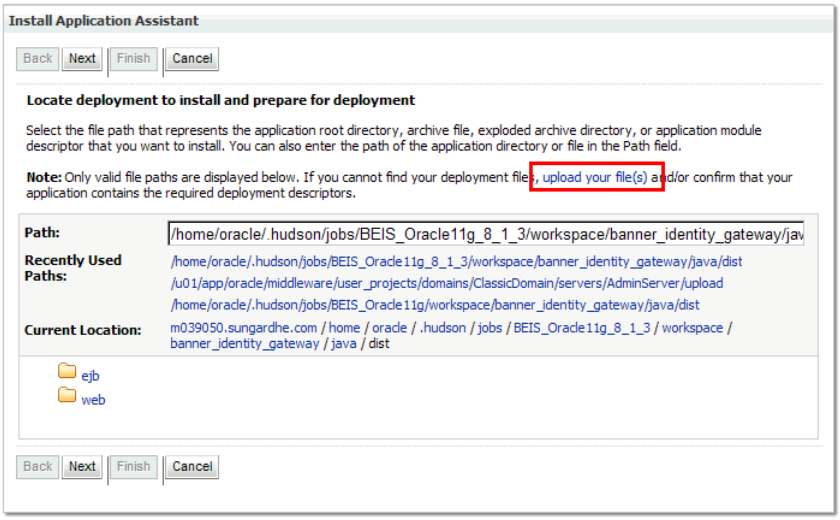
2. In the Domain Structure pane, click **Deployments**.

The Summary of Deployments page is displayed.

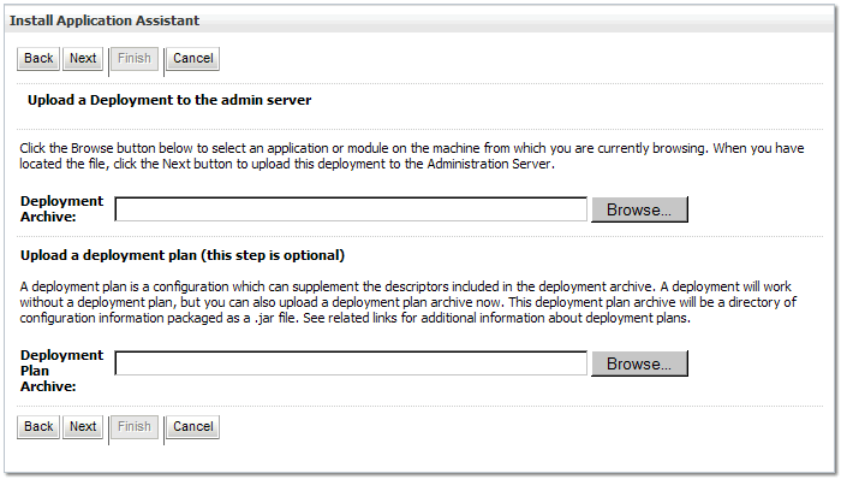


3. In the Change Center pane, click **Lock & Edit**.

4. In the Summary of Deployments page, click **Install**. The Install Application Assistant page is displayed.



5. Click **upload your file(s)**. The next installation page is displayed.

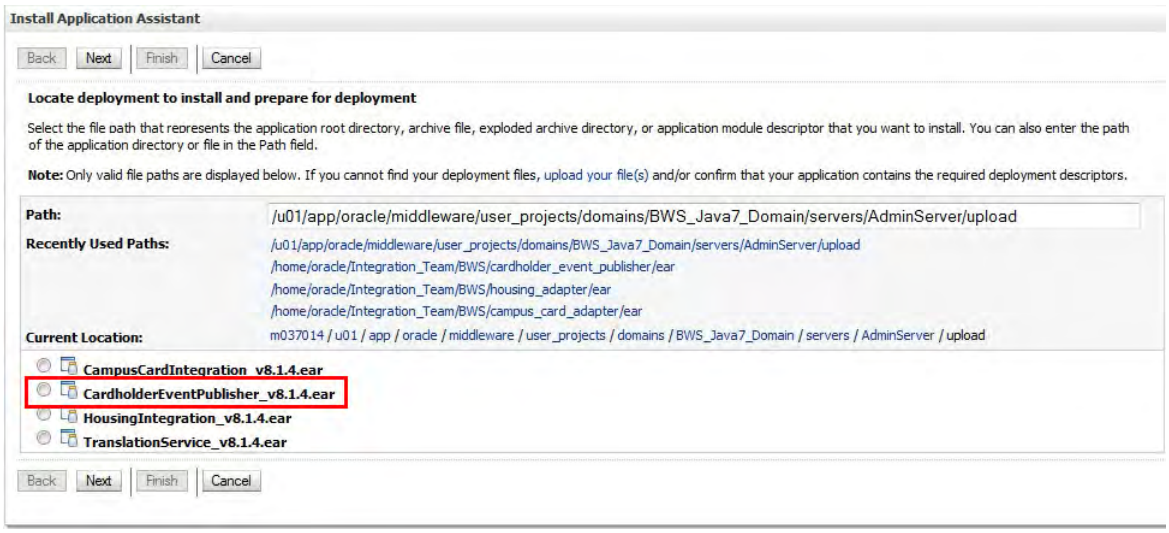


6. Select the file to be uploaded:

   6.1. In the **Deployment Archive** field, click **Browse** and navigate to the `CardholderEventPublisher_v8.1.4.ear` file.

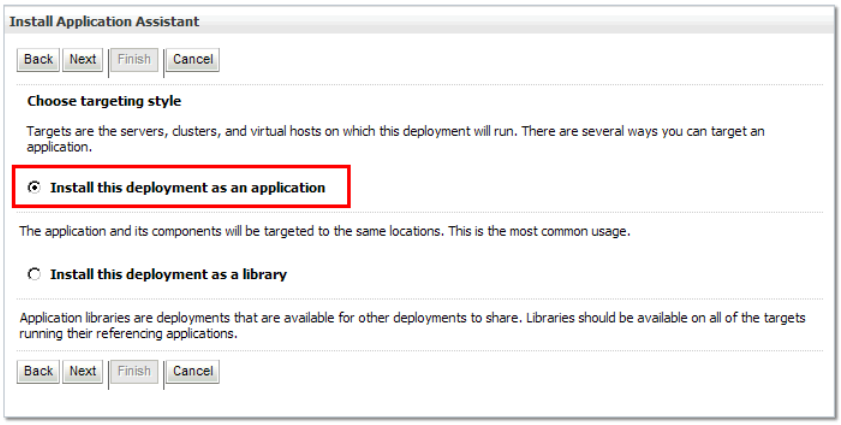
   6.1. Select the file and click **Open**.

7. Click **Next**. The next installation page is displayed.



8. Select the `CardholderEventPublisher_v8.1.4.ear` file from the list.

9. Click **Next**. The next installation page is displayed.

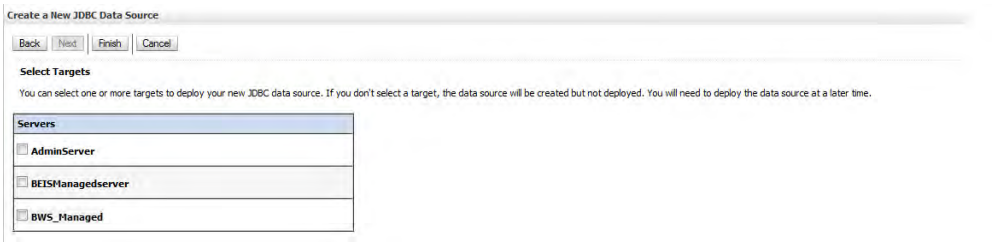


10. Select **Install this deployment as an application**.

11. Click **Next**. The next installation page is displayed.

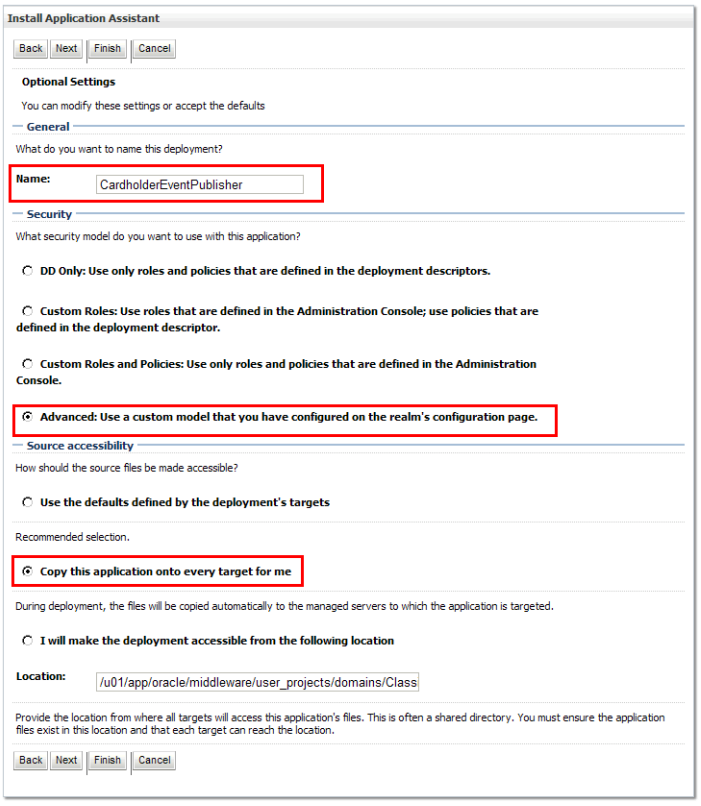
    11.1. In some domains, the following page is displayed. Select the server where the Publisher should be deployed and go to step 12.

    The Publisher must be installed in the instance where the Banner Translation Service and the Banner Web Service Adapter for Campus Card Systems are installed.



    11.2. In some domains, the preceding page is skipped. Go directly to step 12.

12. Click **Next**. The next installation page is displayed.

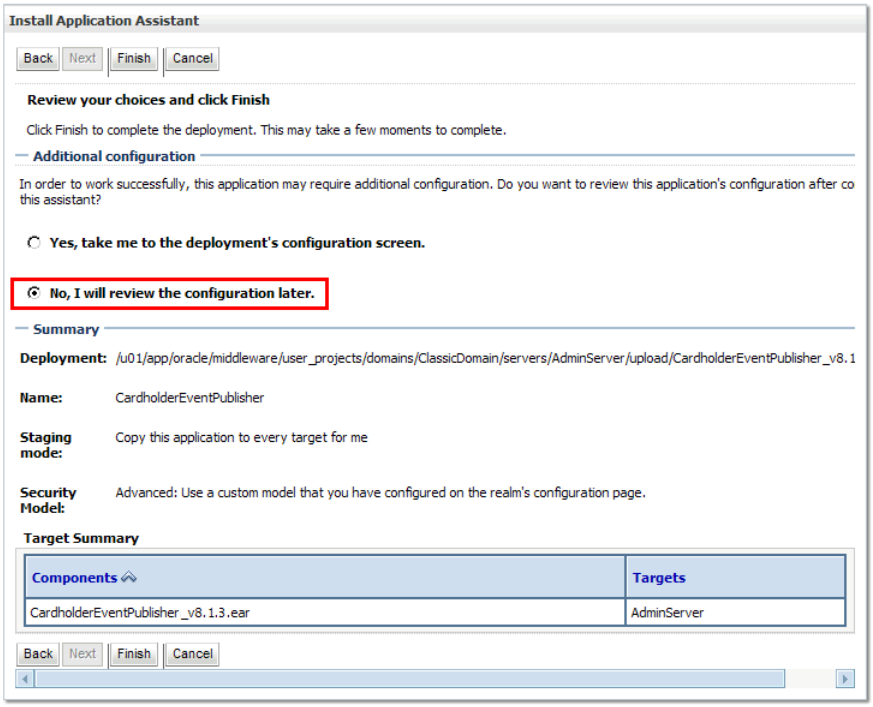


13. Enter a name for the application (for example, *CardholderEventPublisher*) in the **Name** field.

14. Select **Advanced: Use a custom model that you have configured on the realm's configuration page**.

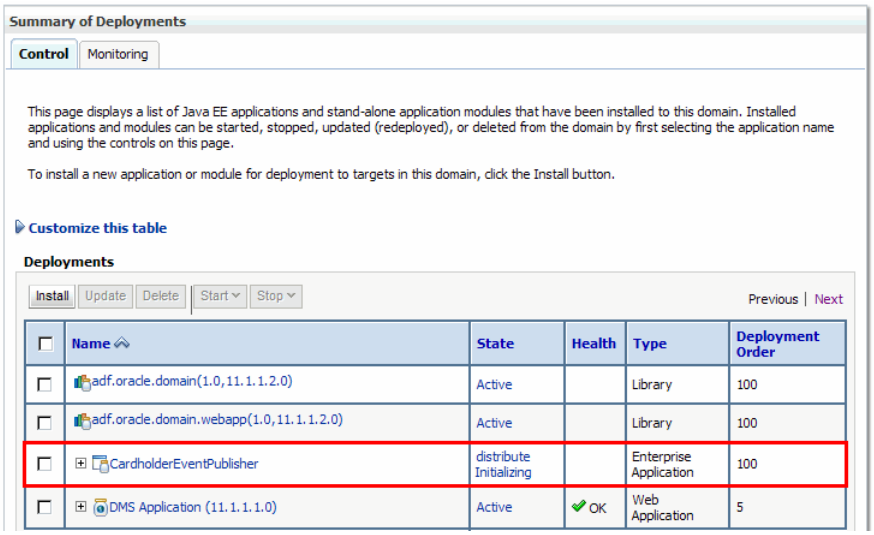
15. Select **Copy this application onto every target for me**.

16. Click **Next**. The next installation page is displayed.



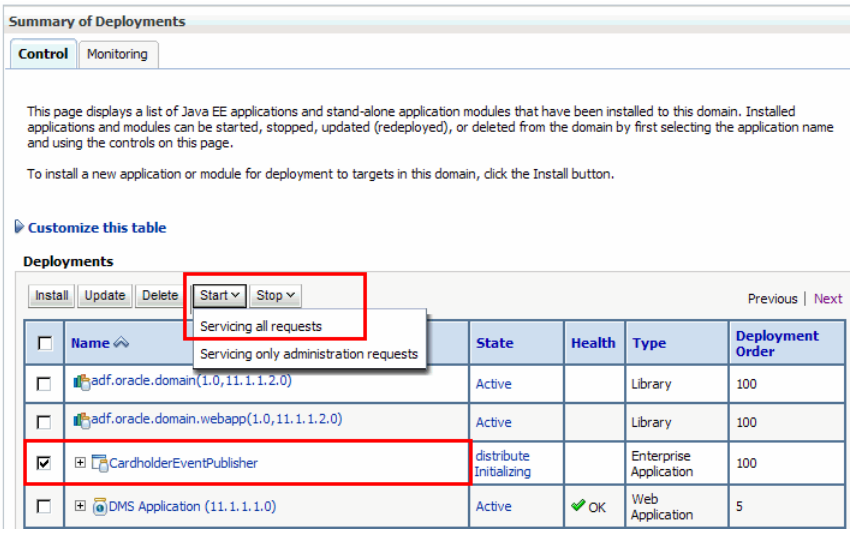
17. Select **No, I will review the configuration later**.

18. Click **Finish** to start the deployment. When deployment is completed, the Summary of Deployments page is redisplayed with the newly deployed Publisher.



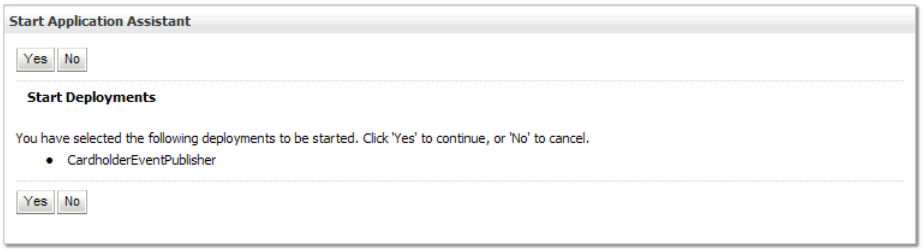
19. In the Change Center pane, click **Activate Changes**.

20. Start the newly deployed application as follows:



20.1. Select the newly deployed Publisher.

20.1. Click **Start > Servicing all requests**. The Start Application Assistant page is displayed.



20.2. Click **Yes**.

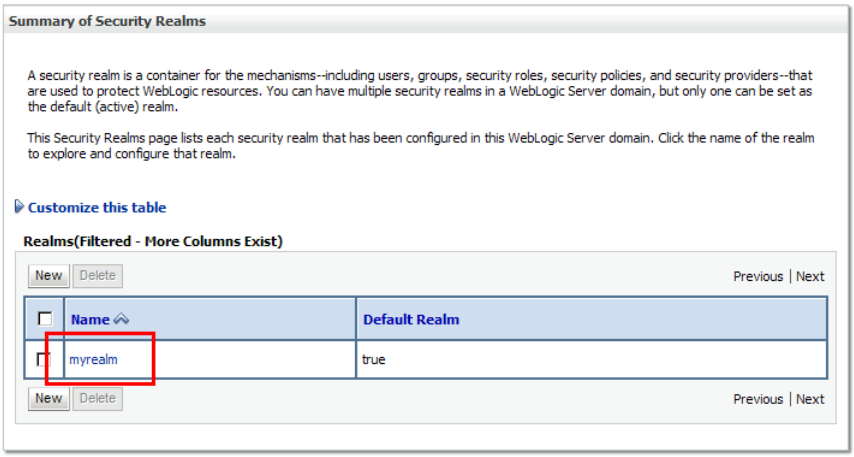
Step 8 Configure the security group and user

Before beginning this step, refer to the security configuration for your version of the Oracle WebLogic Server.

Use the following steps to add the `chepAdminGroup` group and an administrative user to the Banner Cardholder Event Publisher application. This group and user are required for accessing the Banner Cardholder Event Publisher administrative interface.

1. In the Domain Structure pane, click **Security Realms**.

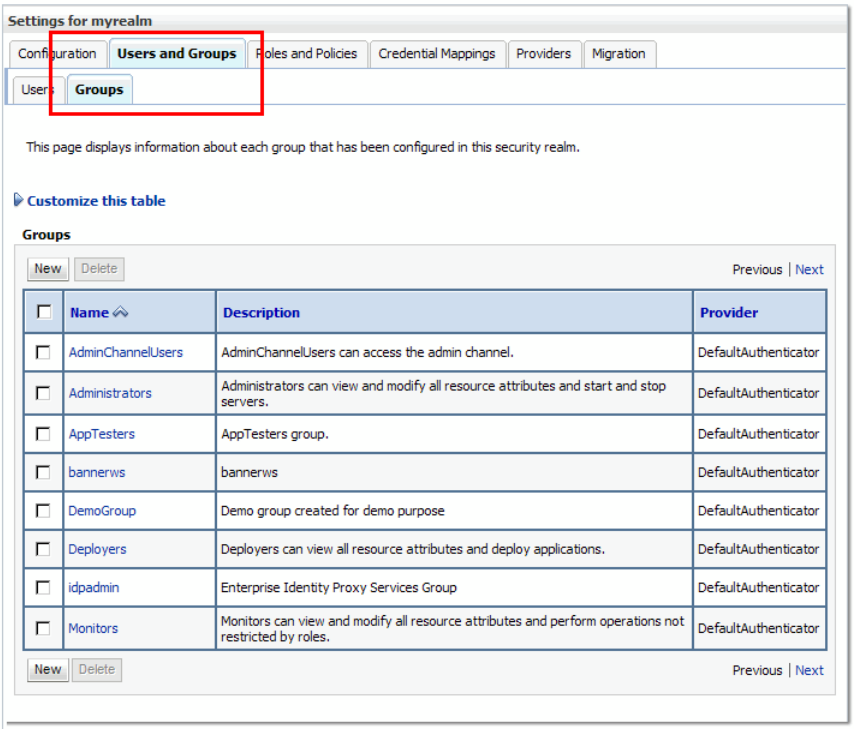
The Summary of Security Realms page is displayed.



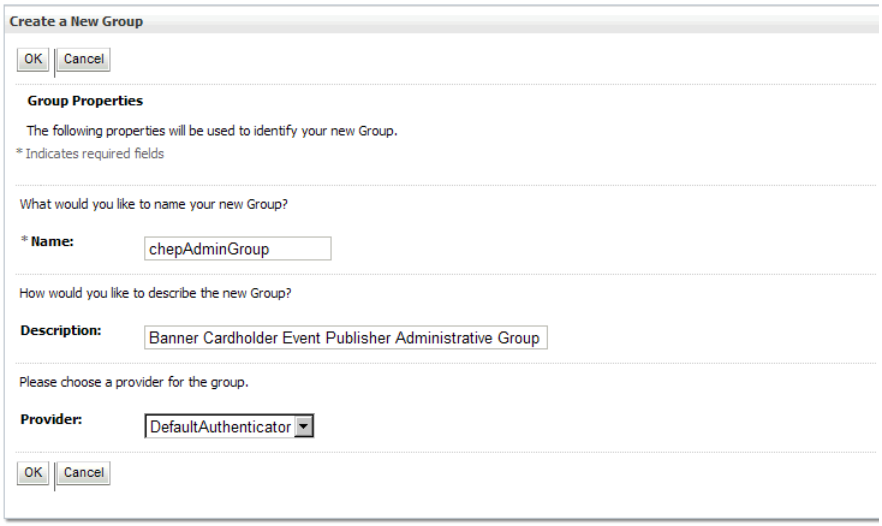
2. Click **myrealm**. The Settings for myrealm page is displayed.

3. Select the **Users and Groups** tab.

4. Select the **Groups** sub-tab. A table of existing groups is displayed.



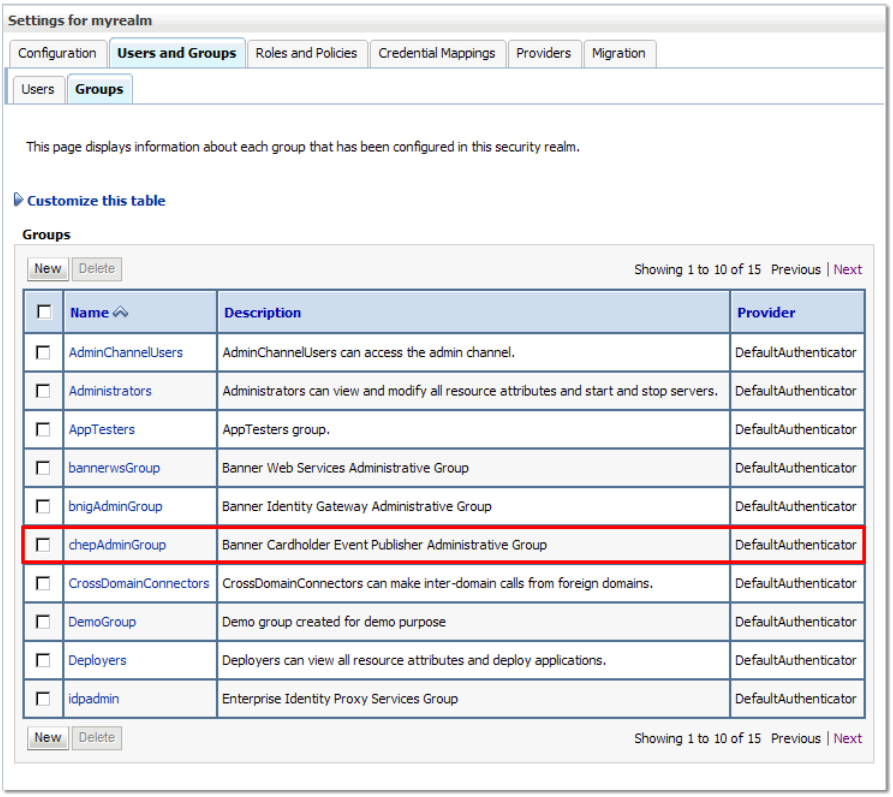
5. Click **New**. The Create a New Group page is displayed.



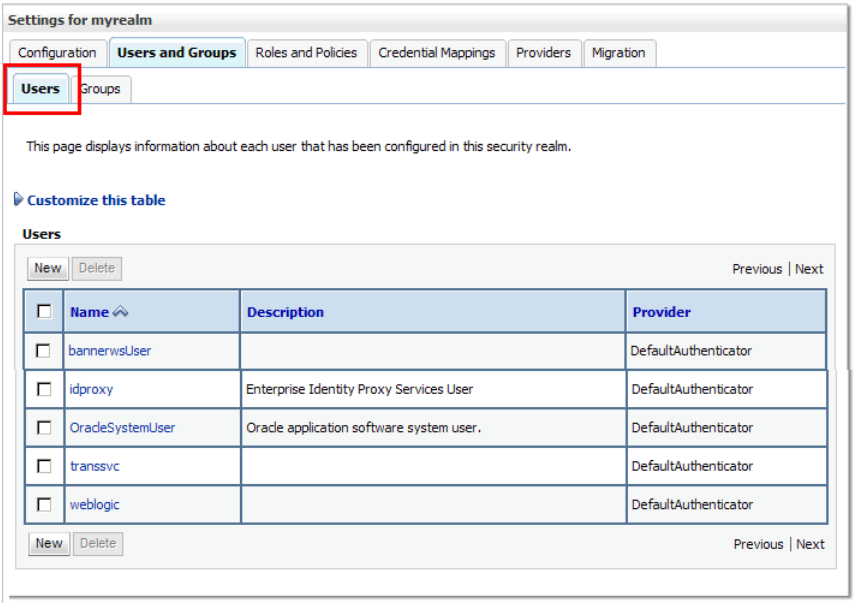
6. Enter the following information to create a group:

Name	*chepAdminGroup*
Description	*Banner Cardholder Event Publisher Administrative Group*
Provider	*DefaultAuthenticator*

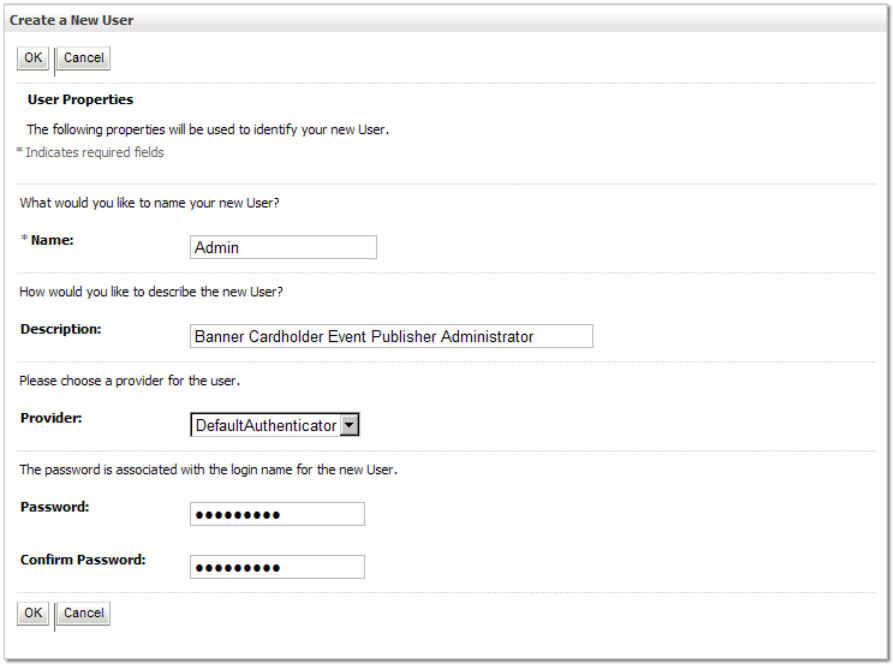
7. Click **OK**. The table of groups is redisplayed with the new group.



8. Select the **Users** sub-tab. A table of existing users is displayed.



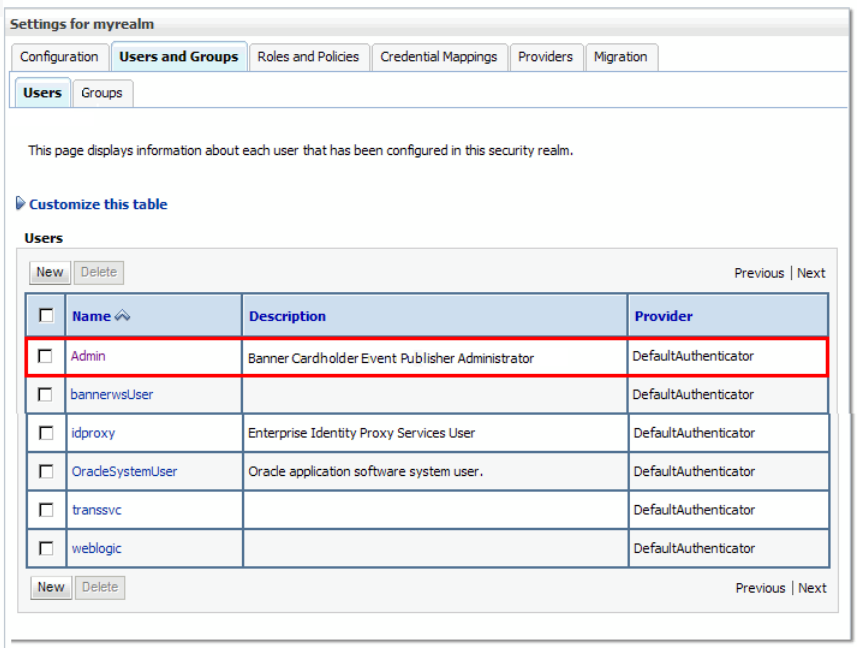
9. Click **New**. The Create a New User page is displayed.



10. Enter the following information to create a user:

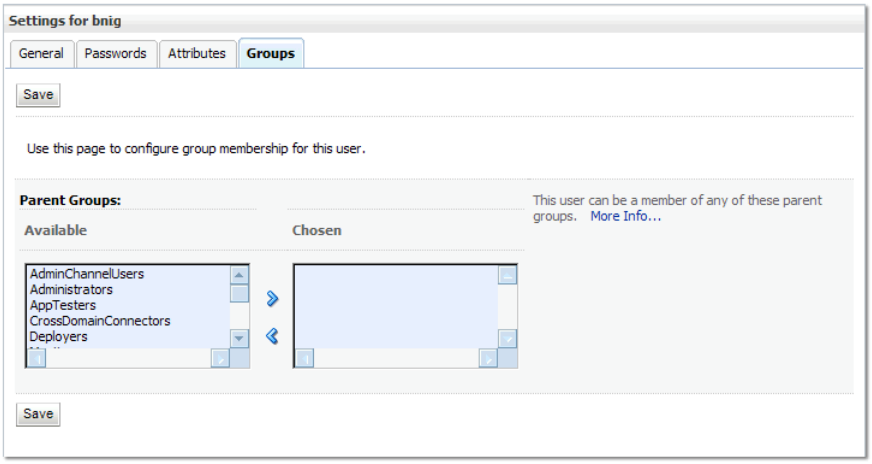
Name	*Admin* (This is an example. Enter the name of your choice.)
Description	*Banner Cardholder Event Publisher Administrator*
Provider	*DefaultAuthenticator*
Password	Password used to log in to the Banner Cardholder Event Publisher administrative interface
Confirm Password	Confirmation of the password

11. Click **OK**. The table of users is redisplayed with the new user.

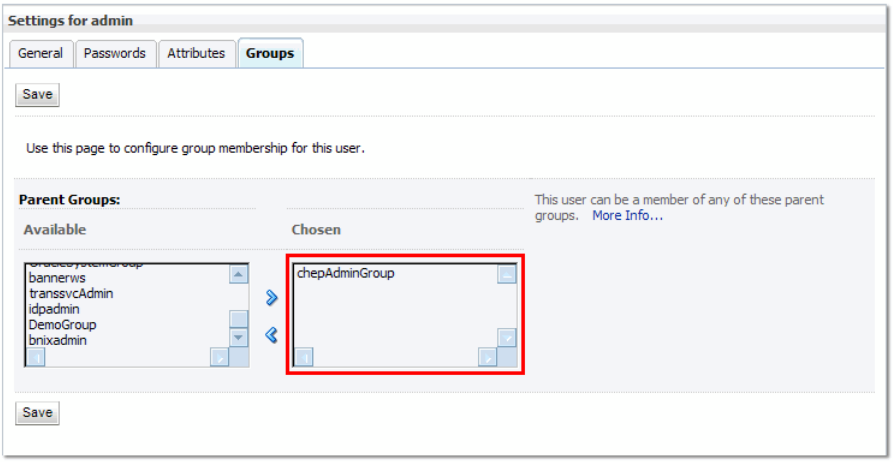


12. Click the name of the user you just created. The Settings page for the user is displayed.

13. Select the **Groups** tab.



14. In the Parent Groups section, select *chepAdminGroup* in the **Available** list and move it to the **Chosen** list.



15. Click **Save**.

16. Restart the server for the changes to take effect.

Configuration

The Banner Cardholder Event Publisher must be configured before it can start publishing events. See “Appendix A, Administering Banner Cardholder Event Publisher” in the *Banner Web Services Handbook* for information on configuring the Publisher.

8 Test Banner Cardholder Event Publisher

Testing the Banner® Cardholder Event Publisher before integrating with a third-party campus card systems is highly recommended, but not required. Testing ensures proper configuration and provides experience publishing SyncEligibleCardholder messages.

To test the publication of SyncEligibleCardholder messages by the Banner Cardholder Event Publisher, a test application must be able to perform the following:

- Expose an endpoint that can receive SyncEligibleCardholder messages via SOAP over HTTP.
- Preferably, display the SyncEligibleCardholder messages it receives.

This chapter describes soapUI, an open source tool that meets these criteria. Other testing means are available, including direct testing with your institution's chosen campus card system.

Setup and use of soapUI

The open source soapUI tool, available from eviware (www.soapui.org), can be used to test exposed web services. It can also be used to generate the mock implementation of a web service based on a WSDL definition. Mock implementations are simulations of a web service that can receive a SOAP message and reply with a predefined response.

A soapUI project is delivered with the Banner Web Services download. This project provides a quick start for exposing a SyncEligibleCardholder endpoint for testing the Banner Cardholder Event Publisher.

Note

Refer to the soapUI documentation for more details on MockService.

Use the following steps to set up and use the soapUI tool:

- Step 1, "Download and install soapUI"
- Step 2, "Open the testing workspace"
- Step 3, "Import the soapUI project"
- Step 4, "Start the MockService"
- Step 5, "Send a test message"
- Step 6, "Add accessible URL for the MockService"

- Step 7, “Reconfigure the Banner Cardholder Event Publisher”
- Step 8, “Test the Banner Cardholder Event Publisher”

Step 1 Download and install soapUI

Download soapUI from eviware (www.soapui.org) and install it.

Step 2 Open the testing workspace

In soapUI, workspaces contain projects. Projects contain web services definitions. The Banner web services testing workspace is located in the unzipped Banner Web Services download. Use the following steps to open the testing workspace.

1. Open soapUI. A default workspace is displayed in the Navigator.
2. Select Switch Workspace from the File menu. The Switch Workspace window is displayed.
3. Navigate to `BANNER_WEB_SERVICES_814\Deployables\Weblogic\Banner Web Services 8.1.4 Testing-workspace.xml` in the unzipped Banner Web Services download.
4. Click **Open**. The testing workspace is displayed.

Step 3 Import the soapUI project

Use the following steps to import the soapUI project.

1. Select Import Project from the File menu. The Select soapui project file window is displayed.
2. Navigate to `BANNER_WEB_SERVICES_814\Deployables\Weblogic\cardholder_event_publisher\testing\CardholderEventPublisher 8.1.4-soapui-project.xml` in the unzipped Banner Web Services download.
3. Click **Open**. The workspace for the project is displayed with two nodes:
   - **SyncEligibleCardholderSoapBinding**
   - **SyncEligibleCardholderSoapBinding MockService**

Step 4 Start the MockService

Use the following steps to start the MockService.

1. Select the **SyncEligibleCardholderSoapBinding MockService** node in the project.

2. Press Enter. The MockService Editor is displayed with three sections:

   - Toolbar
   - Operations pane, which displays the web service operations supported by the MockService
   - Message Log pane, which displays messages as they are received

Note

Tools and options for the Operations pane and the Message Log pane are not described in this document. ■

3. From the toolbar of the MockService Editor, press the green arrow icon to start the MockService. The following changes indicate that the MockService Editor is running:

   - The following message is visible in the toolbar: *running on port 8088*.
   - The red stop icon is displayed.
   - The green arrow icon is dimmed.

Step 5 Send a test message

The MockService can now receive SyncEligibleCardholder messages. A message can be sent from soapUI itself. A test request message is provided in the project for this purpose. Use the following steps to send a test message.

1. Expand the SyncEligibleCardholderSoapBinding node, and then expand the SyncEligibleCardholder node to see Test Request 1.

2. Double-click Test Request 1. The Request Editor is displayed with three sections:

   - Toolbar - The URL field displays the default endpoint for the MockService, which uses the computer's loopback IP address and default HTTP port rather than a specific hostname and port (`http://127.0.0.1:8088/mockSyncEligible CardholderSoapBinding`). This URL can be used as a quick test to see how the MockService works.
   - Request pane with a sample SOAP SyncEligibleCardholder message
   - Response pane (initially empty)

Note

To use the MockService as a destination for messages from the Banner Cardholder Event Publisher, you must determine the appropriate hostname and port number for your computer. Additionally, you might need to disable VPN software and/or open the default HTTP port (8088) using Windows Firewall. ■

3. Click the green arrow icon in the toolbar of the Request Editor to send the SyncEligibleCardholder message to the MockService.

   A `Message Received` message is displayed in the Response pane of the Request Editor.

   A new entry is displayed in the Message Log pane of the MockService Editor.

4. Double-click the message in the Message Log pane. A Message Viewer displays the SyncEligibleCardholder message that was received.

Note

If the complete message is not visible, right-click in the Message Viewer and select Format XML from the context menu. This reformats the message to a displayable format. ■

Step 6 Add accessible URL for the MockService

Use the following steps to determine and add the accessible URL for the MockService so that other computers can communicate with it.

1. Determine the hostname component of the URL by opening the Windows System Properties application from the Windows Control Panel. The value in the **Full computer name** field on the **Computer Name** tab can be used as the hostname component of the URL.

2. Determine the port number component of the URL. It can be kept as *8088*, or it can be changed using the soapUI MockService Editor.

3. Add an exception in Windows Firewall that opens the port for inbound requests.

4. Using the same test request (Test Request 1) or a clone of it, open the endpoint drop-down list from the toolbar of the Request Editor and select the **[add new endpoint]** option.

5. On the Add new endpoint window, edit the endpoint to reflect the hostname and port number (if changed).

6. Click **OK**.

7. Click the green arrow icon in the toolbar of the Request Editor to send the SyncEligibleCardholder message to the MockService.

   A `Message Received` message should be displayed in the Response pane of the Request Editor, and a new entry should be displayed in the Message Log pane of the MockService Editor. If you do not see these results, verify that any installed VPN software is disabled. Additional troubleshooting might be required.

Step 7 Reconfigure the Banner Cardholder Event Publisher

When the MockService can be reached using your computer's hostname, reconfigure the Banner Cardholder Event Publisher. See "Appendix A, Administering Banner Cardholder Event Publisher" in the *Banner Web Services Handbook* for information on configuring the Publisher.

Step 8 Test the Banner Cardholder Event Publisher

Test event publication by changing data in Banner and watching for messages in the Message Log pane of the MockService Editor.

Test cases

Once the Publisher is deployed to the Oracle WebLogic Server 11g and configured to post messages to the exposed endpoint, testing can begin. This entails creating data in Banner and monitoring the results. Typical test cases include the following:

- Create a person
- Update a person's name
- Add an address for a person
- Add a telephone number for a person
- Add an e-mail address for a person
- Enroll a person as a student
- Add a student's expected graduation year, term, and/or date
- Enroll a student in an active meal plan
- Assign an employee to a primary job that has an associated job location

Some of these test cases can have multiple contexts. For example, residence location addresses can be derived from information entered on the General Person Identification (SPAIDEN) form and stored in the SPRADDR table or from information entered on the Room Assignment (SLARASG) form and stored in the SLRRASG table. Be sure to conduct tests that are appropriate for your environment.