Top of Form

Bottom of Form

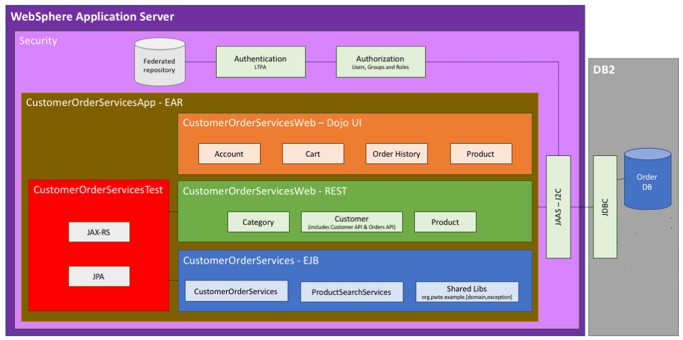
Top of Form

Bottom of Form

**Customer Order Services – Java EE Enterprise Application - WebSphere 855 Repository**

**Application Overview**

The application is a simple store-front shopping application, built during the early days of the Web 2.0 movement. As such, it is in major need of upgrades from both the technology and business point of view. Users interact directly with a browser-based interface and manage their cart to submit orders. This application is built using the traditional [3-Tier Architecture](http://www.tonymarston.net/php-mysql/3-tier-architecture.html) model, with an HTTP server, an application server, and a supporting database.



There are several components of the overall application architecture:

* Starting with the database, the application leverages two SQL-based databases running on [IBM DB2](https://www.ibm.com/analytics/us/en/technology/db2/).
* The application exposes its data model through an [Enterprise JavaBean](https://en.wikipedia.org/wiki/Enterprise_JavaBeans) layer, named **CustomerOrderServices**. This components leverages the [Java Persistence API](https://en.wikibooks.org/wiki/Java_Persistence/What_is_JPA%3F) to exposed the backend data model to calling services with minimal coding effort.
  + As of the [WebSphere Application Server](http://www-03.ibm.com/software/products/en/appserv-was) Version 855 build, the application is using **EJB 3.0** and **JPA 2.0** versions of the respective capabilities.
* The next tier of the application, named **CustomerOrderServicesWeb**, exposes the necessary business APIs via REST-based web services. This component leverages the [JAX-RS](https://en.wikipedia.org/wiki/Java_API_for_RESTful_Web_Services) libraries for creating Java-based REST services with minimal coding effort.
  + As of the [WebSphere Application Server](http://www-03.ibm.com/software/products/en/appserv-was) Version 855 build, the application is using **JAX-RS 1.1** version of the respective capability.
* The application's user interface is exposed through the **CustomerOrderServicesWeb** component as well, in the form of a [Dojo Toolkit](https://github.com/IBM/application-modernization-javaee-quarkus/blob/master/monolith-websphere-855/README.md#tbd)-based JavaScript application. Delivering the user interface and business APIs in the same component is one major inhibitor our migration strategy will help to alleviate in the long-term.
* Finally, there is an additional integration testing component, named **CustomerOrderServicesTest** that is built to quickly validate an application's build and deployment to a given application server. This test component contains both **JPA** and **JAX-RS**-based tests.

**Build and deploy the application on WebSphere Application Server 855**

**1. Prerequisites**

The following are prerequisites for deploying the original ASIS version of this application:

* [WebSphere Application Server Version 855](http://www-03.ibm.com/software/products/en/appserv-was)

**2. Getting the project repository**

You can clone the repository from its main GitHub repository page and checkout the appropriate branch for this version of the application.

1. git clone https://github.com/ibm-cloud-architecture/cloudpak-for-applications.git
2. cd cloudpak-for-applications
3. git checkout was855

**3. Running the Database and Creating the tables**

This project uses DB2 as its database. See the DB2 section of the git repository for the DB2 database configuration. The ORDERDB setup is in Docker, and runs on the local VM in the environment

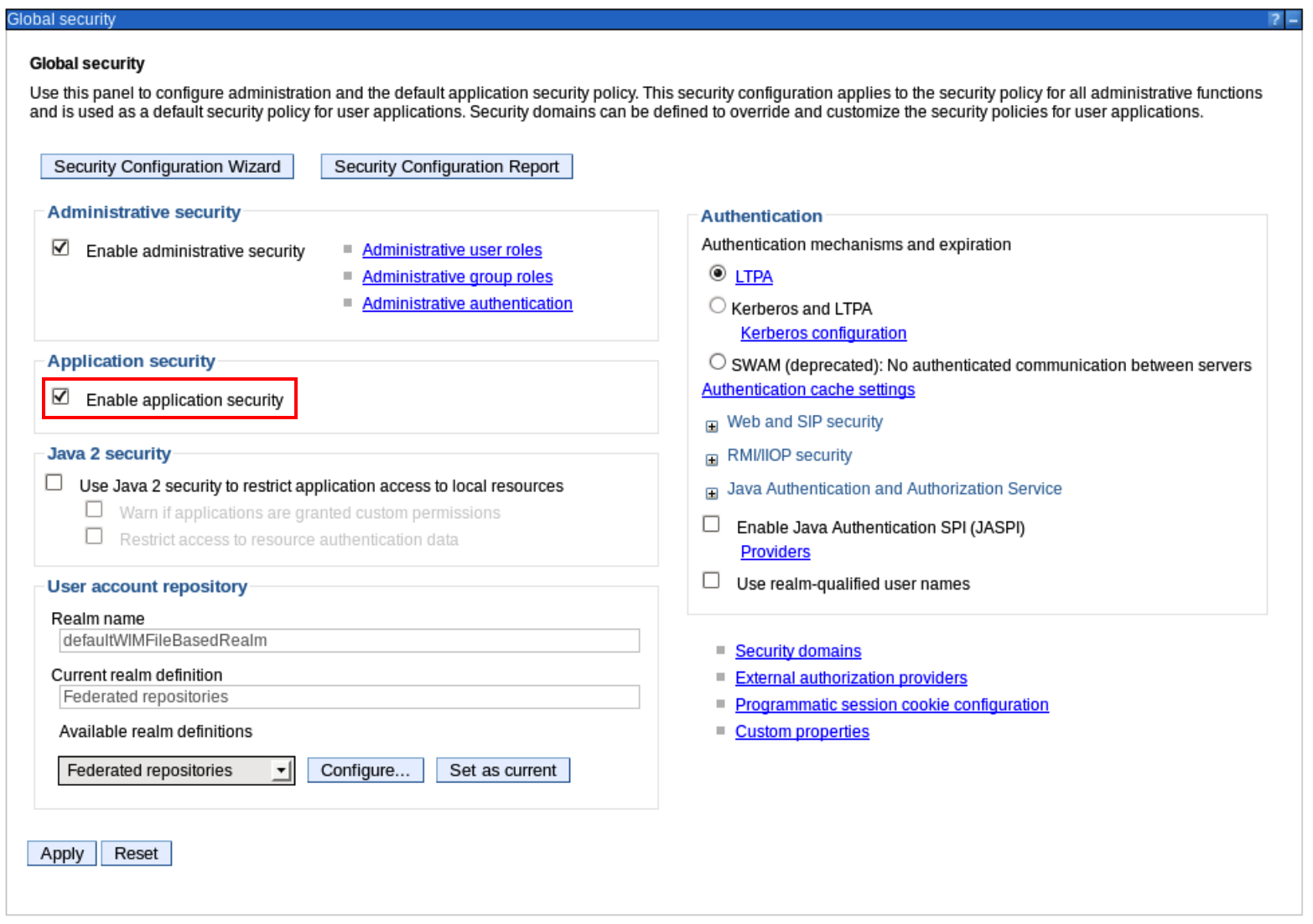
**4. Configuring the WebSphere v855 Environment with Security and Resources**

WebSphere environment configuration can be setup using the automation script or it can be done manually. You can choose from either ways based upon your convenience.

**Manual Setup**

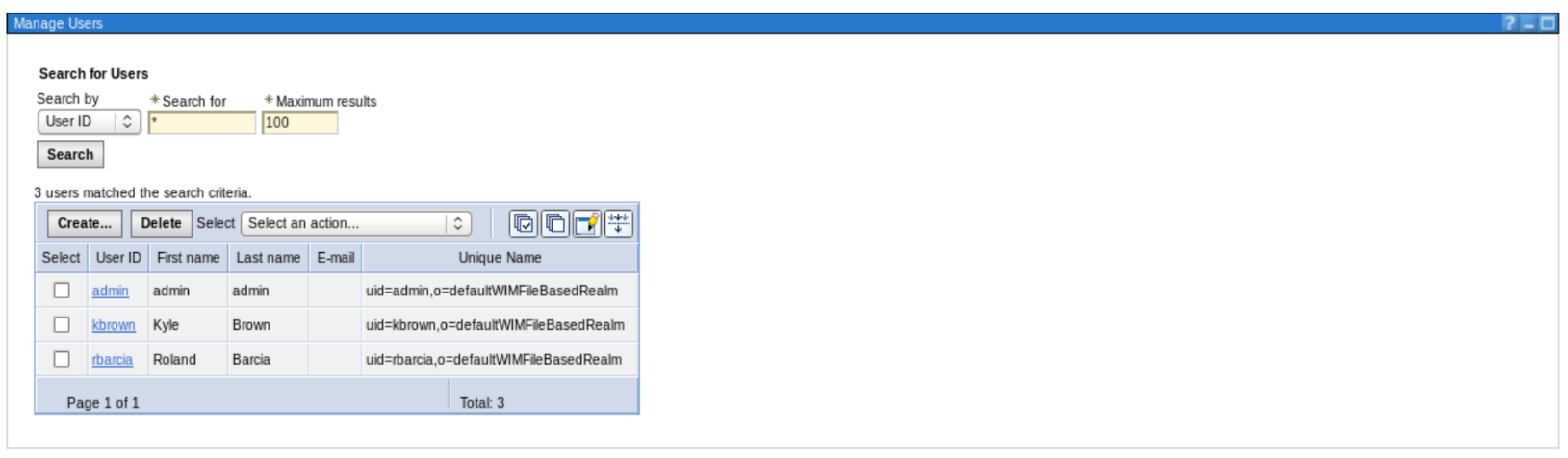
**Setting Up Security**

1. Log into the Admin Console via <http://localhost:9043/admin>.
2. In the Global security section, check **Enable application security** and click **Save**.

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme1.png)

1. In the **Users and Groups** section, select **Manage Users** and create the following users:

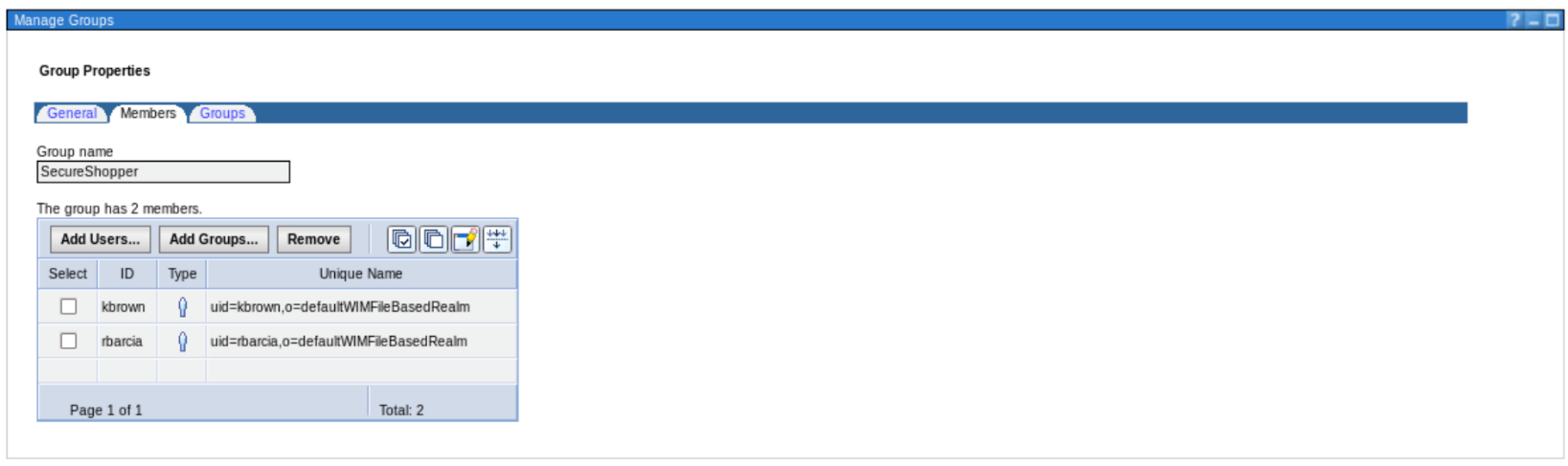
* username: **rbarcia** password: **passw0rd**
* username: **kbrown** password: **passw0rd**

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme2.png)

1. In the **Users and Groups** section, select **Manage Groups** and create the following group:

* group name: **SecureShopper**

This JEE application implements role-based security whereby only those users and groups with appropriate roles can execute certain actions. As a result, all users must belong to the SecureShopper group if they want to be able to access to the protected customer resources:

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme3.png)

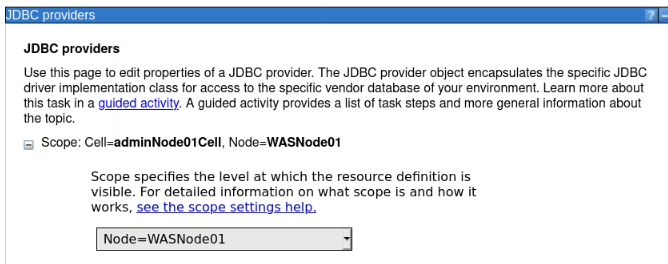
During deployment, you will need to map your desired users or groups to the **SecureShopper** role. By default, SecureShopper group gets mapped to the SecureShopper role.

Under **Global Security**, select **J2C authentication data**. Create a new user named **DBUser** using your db2 instance and password. (db2inst1 / db2inst1)

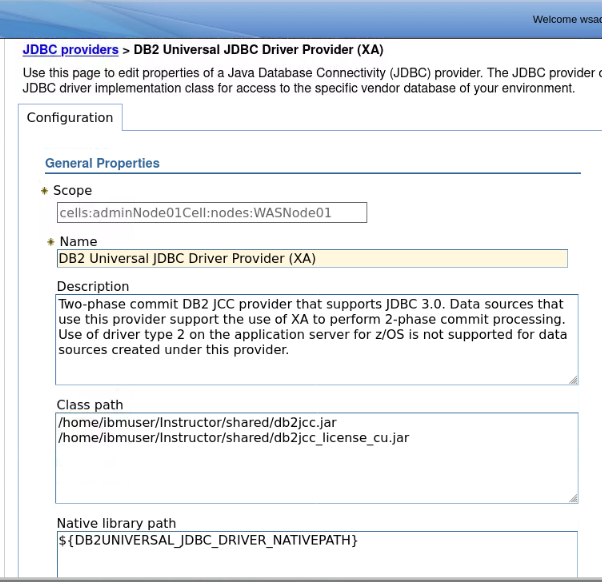
[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme4.png)

**Configuring JDBC Resources**

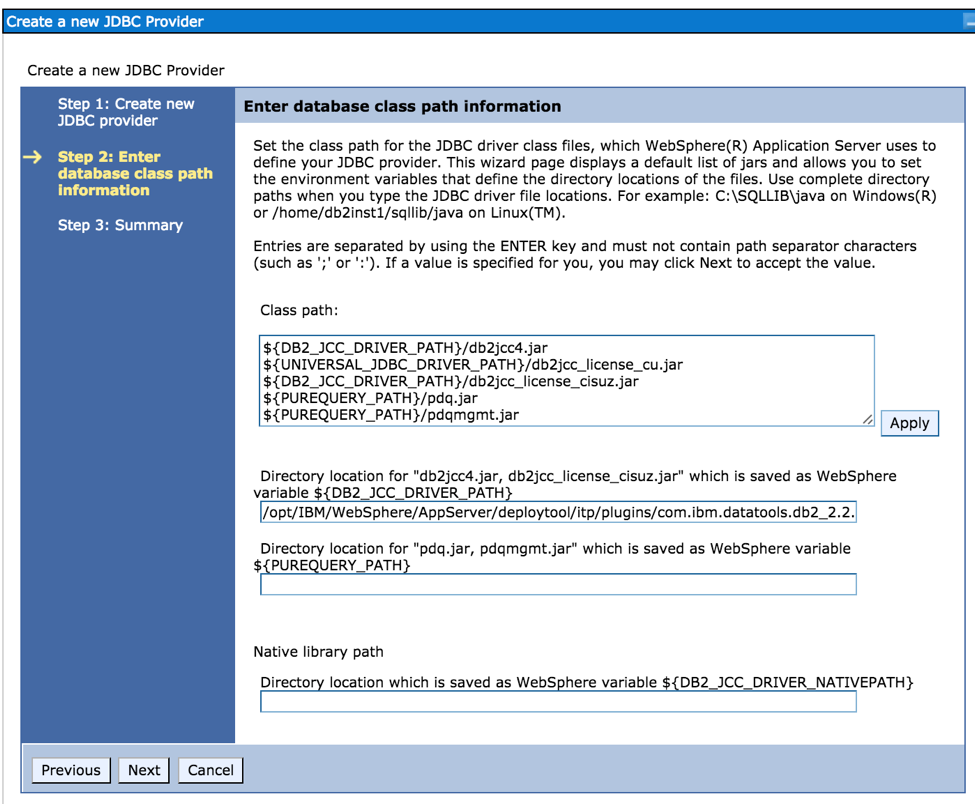
1. Go to the **Resources > JDBC > JDBC Providers** section and ensure that you are at the **Cell** scope.



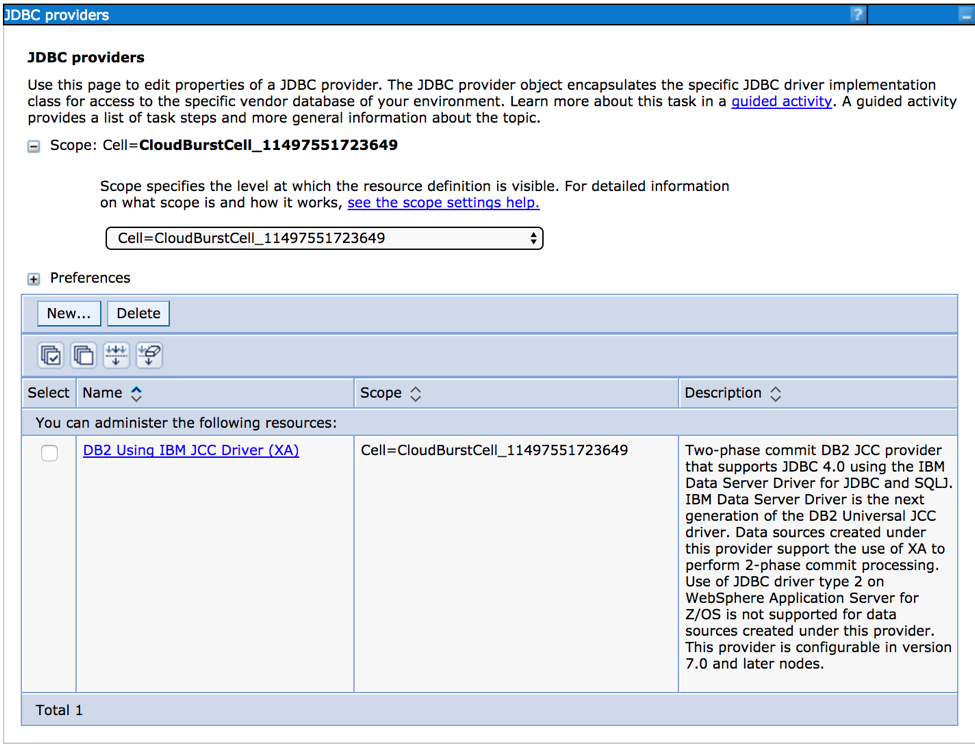
1. Click the New Button to create a new JDBC provider.
   * Database type : **DB2**
   * Provider type : **DB2 Using IBM JCC Driver**
   * Implementation type : **XA data source**

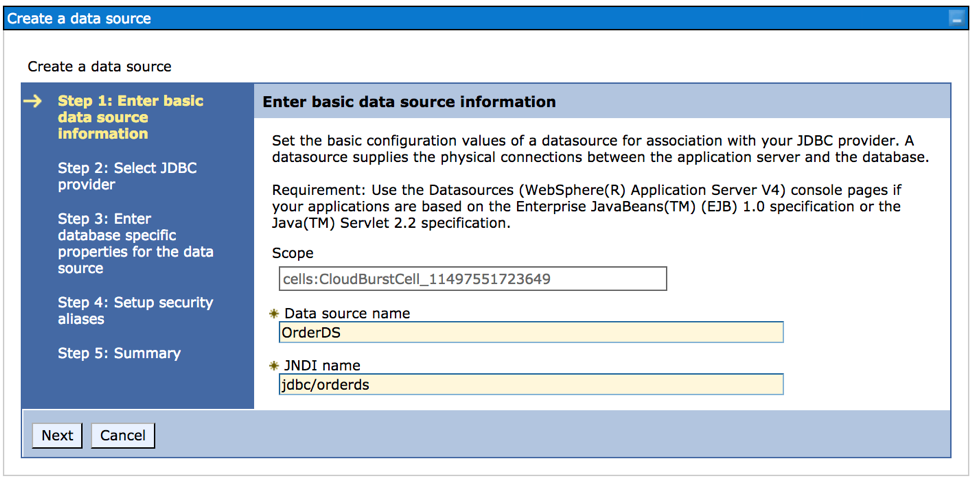
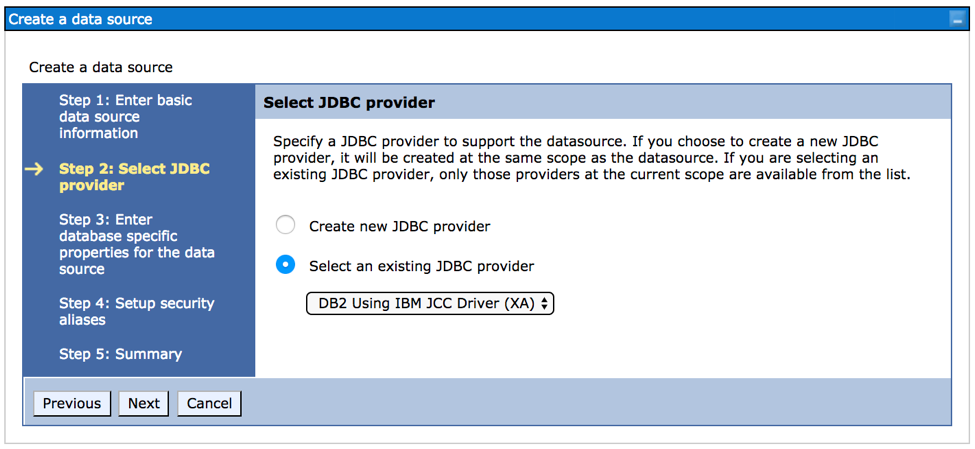
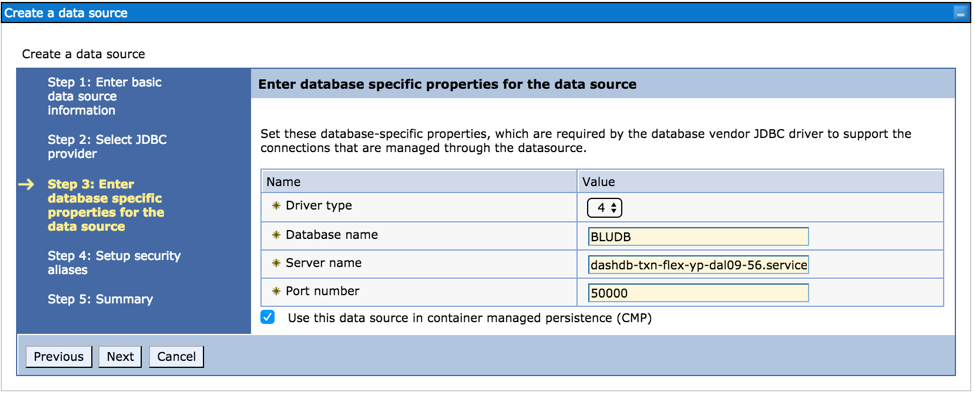
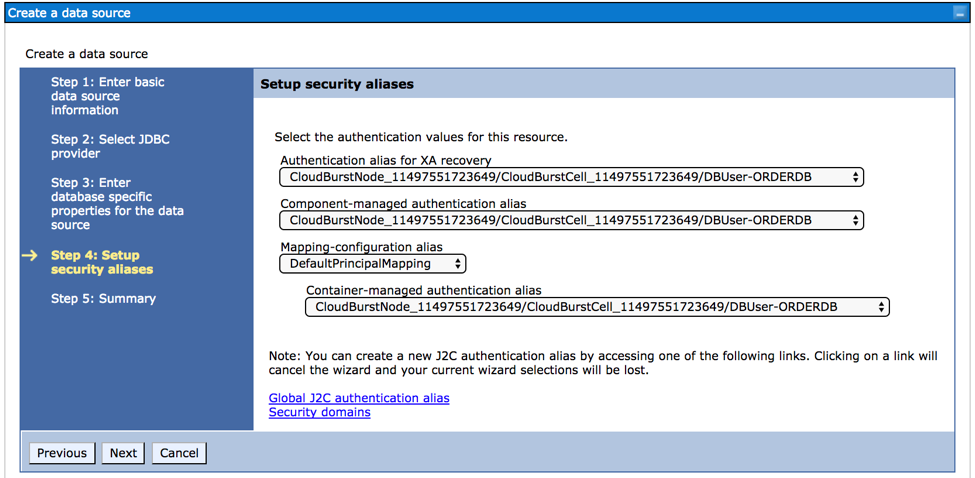


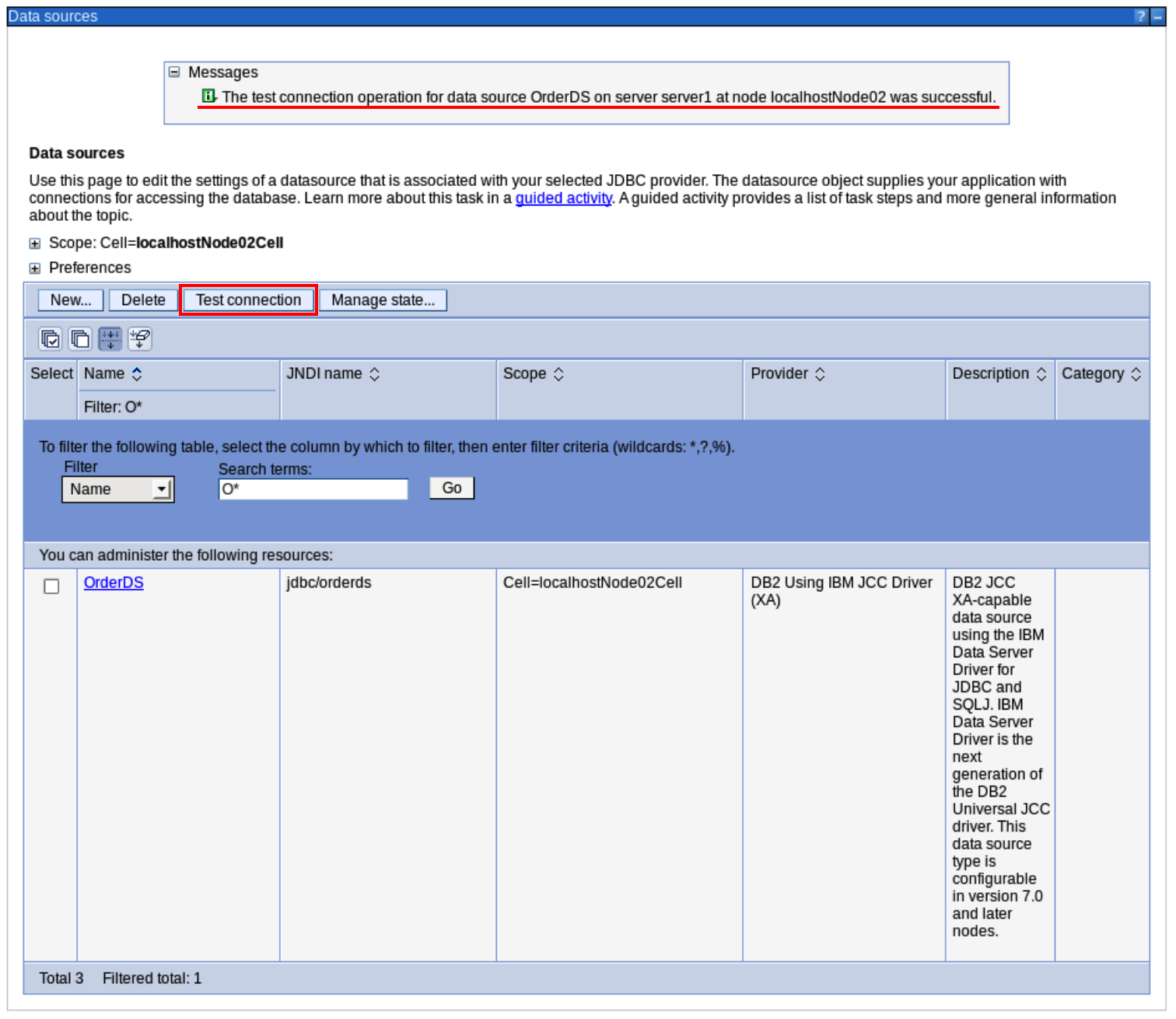
1. You need to enter the database class path information. Enter the directory where the DB2 Java is set.

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme7.png)

1. Press **Next** and then **Finish**. Save the Configuration.

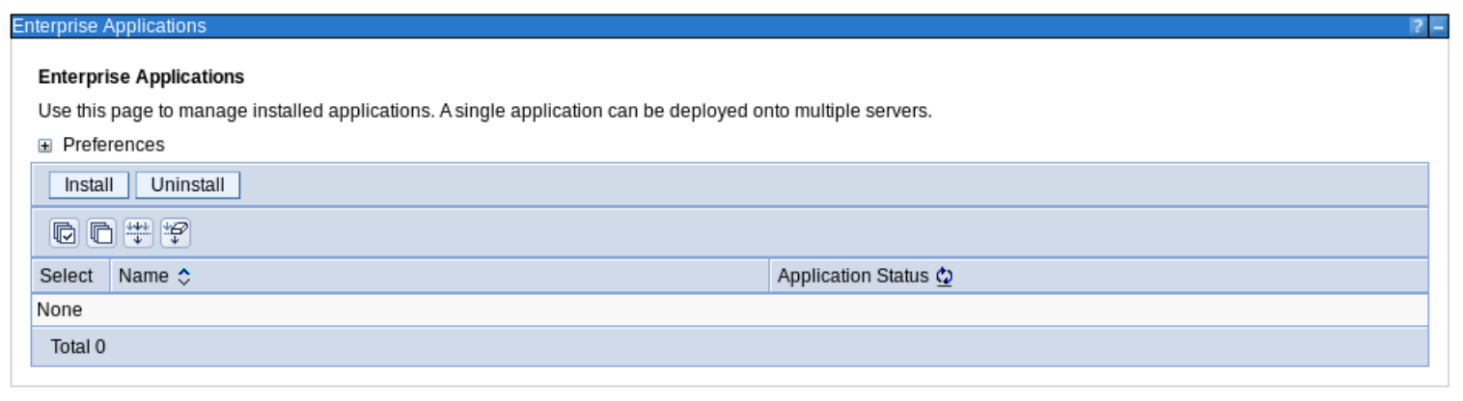
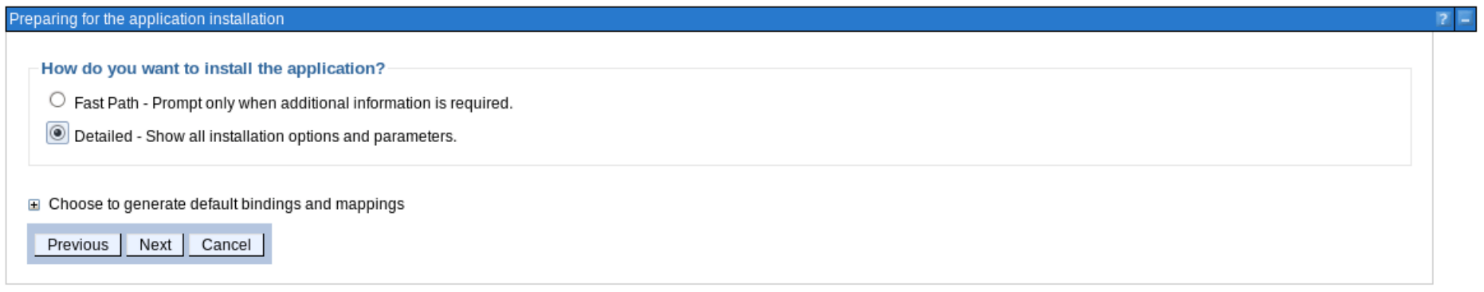
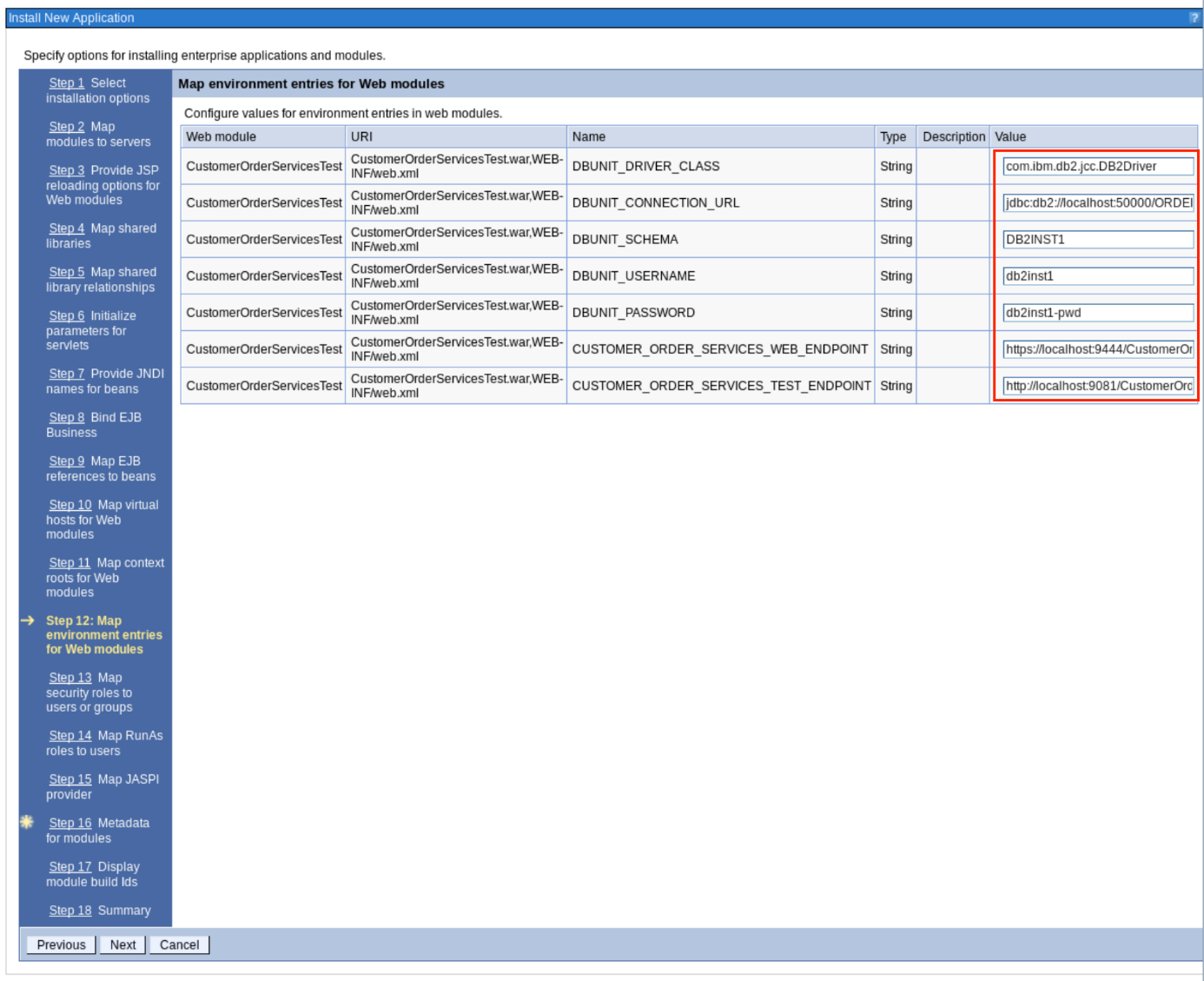
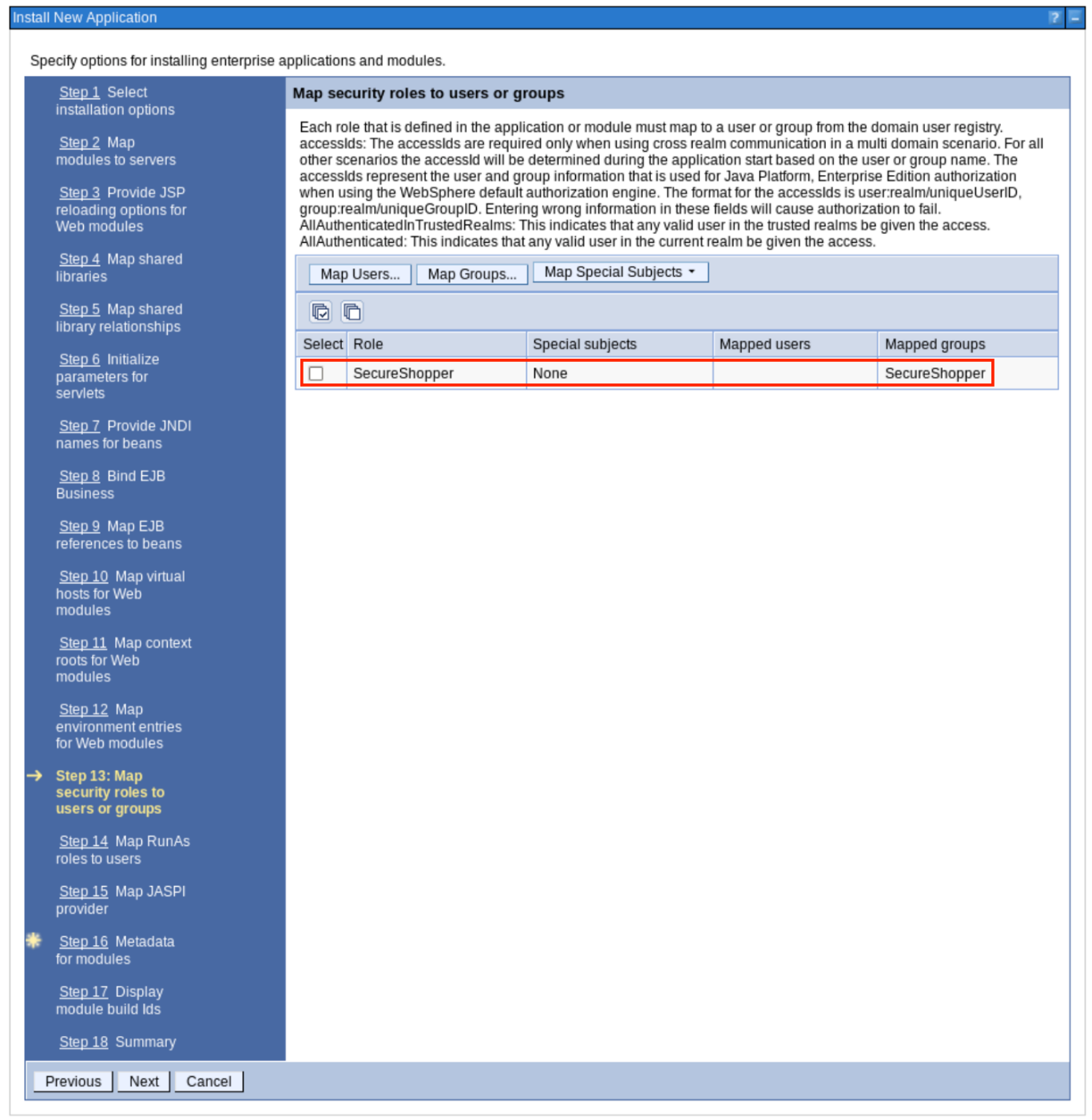
[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme8.png)

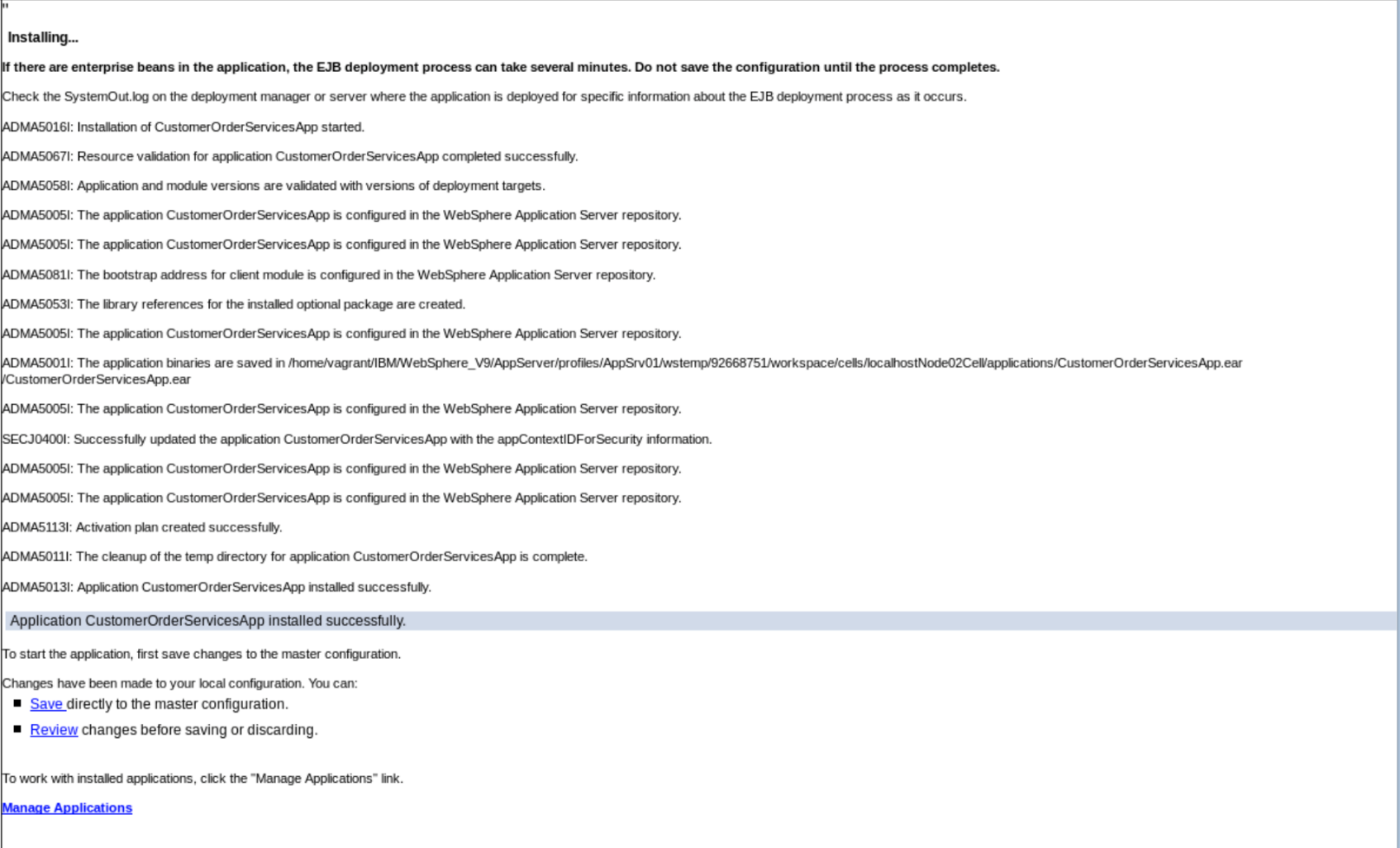
1. Go to the **Resources > JDBC > Data sources** section to create a new data source.
   1. Make sure that the scope is at **Cell** level and click **New**
   2. OrderDB - Step 1
      * Data source name: **OrderDS**
      * JNDI name: **jdbc/orderds**  
        [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme9.png)
   3. OrderDB - Step 2
      * Select an existing JDBC provider --> **DB2 Using IBM JCC Driver (XA)** [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme10.png)
   4. ORDERDB - Step 3
      * Driver Type: **4**
      * Database name: **ORDERDB**
      * Server name: **Your default DB2 host**
      * Port number: **Your default DB2 port** [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme11.png)
   5. OrderDB - Step 4
      * Authentication alias for XA recovery: **DB2User**
      * Component-managed authentication alias: **DB2User**
      * Mapping-configuration alias: **DefaultPrincipalMapping**
      * Container-managed authentication alias: **DB2User** [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme12.png)
2. Once this is done, under Preferences, there will be a new resource called **OrderDS**. Make sure that the resources got connected using **Test Connection** option. You will see a success message if the connection is established successfully.

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme13.png)

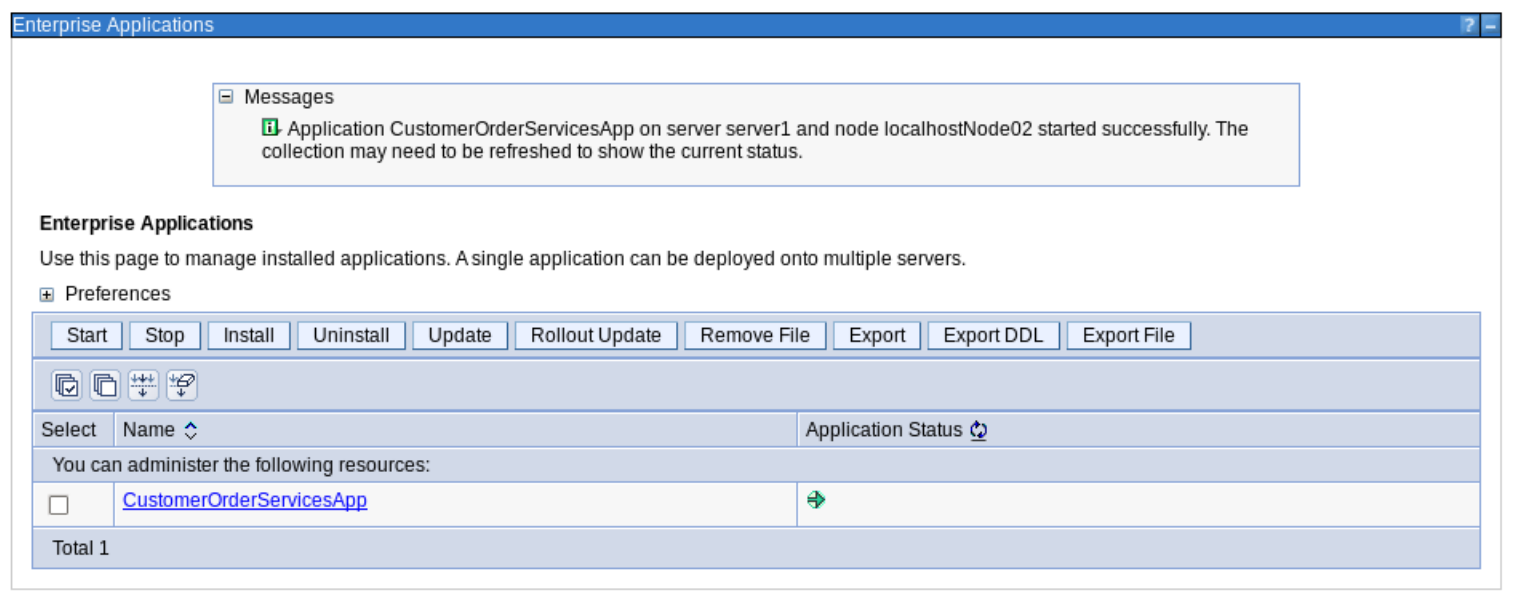
1. Check the Data source and select Test Connection to ensure you created the database correctly. If the connection fails, a few things to check are
   * Your database is started as we did in the beginning.
   * Your host and port number are correct.
   * The classpath for the Driver is set properly.
   * Check the WebSphere Variables. You may want to change them to point to your local DB2 install.

**5. Installing the Application in WAS855**

1. Build the EAR using Maven in CustomerOrderServicesProject.
   * Install Maven and run mvn -v to test your version
   * cd CustomerOrderServicesProject
   * mvn clean package
   * You will have an EAR built in the CustomerOrderServicesApp/target subdirectory, named CustomerOrderServicesApp-X.Y.Z-SNAPSHOT.ear.
2. Install the EAR to <http://localhost:9060/ibm/console>
   * Login to the Administrative Console.
   * Select **Applications > Application Types > WebSphere enterprise applications**  
     [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme14.png)
   * Choose **Install > Browse the EAR > Next > Choose Detailed**  
     [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme15.png)
   * Click on **Step 12**. Customize the environment variables for your system. This is most likely just going to be the **DBUNIT\_SCHEMA**, **DBUNIT\_USERNAME**, and **DBUNIT\_PASSWORD** fields. Those values need to be specific to your local DB2 installation. [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme17.png)
   * Click on **Step 13**. Verify the **SecureShopper** role is mapped to the **SecureShopper** group (or a corresponding group in your application server's user registry). [](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme18.png)
   * Click on **Summary** (Step 16) and click **Finish**.
   * Once you see Application CustomerOrderServicesApp installed successfully, click **Save** and now your application is ready.

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme19.png)

1. Go back to the Enterprise Applications list, select the application, and click **Start**.

[](https://github.com/ibm-cloud-architecture/refarch-jee/raw/master/static/imgs/Customer_README/Readme20.png)

**6. Running the application**

KLP: No need to do the following unless you want to add NEW users. The DB already has 4 users efined. Kbrown, rbarcia, dmulley, dvandepol

Note: Be sure you are NOT logged into the WAS ADMIN Console (wsadmin) when running the app. The wsadmin user will be used, and you will not be prompted to login to the app. And wsadmin is NOT in the ORDER DB, and is not a SecureShopper ID.

In order to create new users for the application, you need to have your application users defined in both your LDAP/Security registry and the application database. For adding users to your security registry please see the previous [security section](https://github.com/IBM/application-modernization-javaee-quarkus/blob/master/monolith-websphere-855/README.md#setting-up-security). The *ORDERDB* application database contains a table called *CUSTOMER* which will store the application users. Therefore, you also need to add your application users to this table:

1. Edit the [addBusinessCustomer.sql](https://github.com/ibm-cloud-architecture/refarch-jee-customerorder/blob/was90-dev/Common/addBusinessCustomer.sql) and/or [addResidentialCustomer.sql](https://github.com/ibm-cloud-architecture/refarch-jee-customerorder/blob/was90-dev/Common/addResidentialCustomer.sql) sql files you can find in the Common folder for this repo to define your users in there.
2. Connect to the ORDERDB database: db2 connect to ORDERDB
3. Execute the sql files: db2 -tf Common/addBusinessCustomer.sql and/or db2 -tf Common/addResidentialCustomer.sql

You should now be able to log into the Customer Order Services application with your newly created users.