Deployment on IBM WebSphere Application Server

Summary: This page describes how to deploy the CAST dashboards on **IBM WebSphere Application Server (WAS).**

Introduction

CAST recommends that the CAST dashboards are installed on **Apache Tomcat**, however it is also possible to use **IBM WebSphere Application Server (WAS)** instead. The following page describes how to perform this installation.

IBM WebSphere Application Server (WAS) is not supported for deployment of **CAST Dashboard 2.0**.

Prerequisites and assumptions

The following section lists all prerequisites and assumptions that have been made. Please ensure that your environment adheres to these:

	Item	Description
✓	**IBM WebSphere Application Server 8.5.x**	This document describes installation on **IBM WebSphere Application Server 8.5.x**. It is assumed that this is already installed and functioning.
✓	**Host Operating System**	This document assumes that **IBM WebSphere Application Server 8.5.x** is already installed on a **Linux Operating System**.
✓	**RDBMS for CAST AIP schemas**	This document assumes that **CAST Storage Service** will be used to host the CAST AIP schemas. It is also assumed that the CAST Storage Service is installed and configured already (see CAST Storage Service) either on the same machine used for **IBM WebSphere Application Server 8.5** or on a remote machine (however, instructions are not provided for CAST Storage Service hosted on a remote machine).
✓	**Java Development Kit (JDK)**	This document assumes that a **JDK 1.7.0** is already installed on the **Linux Operating System**.
✓	**Linux user used to run IBM WebSphere Application Server**	This document assumes that the **Linux user** used to run **IBM WebSphere Application Server** has root privileges.

Step 1 - Install the JDBC driver on the host Linux Operating System

This step describes the installation of a JDBC driver on the host Linux Operating System so that the CAST Storage Service can be accessed. The instructions below assume that the CAST Storage Service (i.e. the equivalent PostgreSQL installation) are installed on the local machine.

First check the existing PostgreSQL version installed on the machine. This shows an equivalent to a **CAST Storage Service 2**:

```
james@andromeda:~$ psql --version
psql (PostgreSQL) 9.2.14
```

Check the JDK version installed on the machine and if necessary download/install the correct version:

```
james@andromeda:~$ java -version
```

Download and move the PostgreSQL JDBC drivers for your PostgreSQL server to the appropriate **WAS** folder:

```
james@andromeda:~/download$ wget https://jdbc.postgresql.org/download/postgresql-42.1.4.jar
james@andromeda:~/download$ cp postgresql-42.1.4.jar /opt/IBM/WebSphere/AppServer/pgsql/postgresql-42.1.4.jar
```

Grant permissions to the user that is running the WAS to the target folder (this is not required if WAS is running with root privileges). In the example below, WAS runs under the "websphere" user/group:

```
james@andromeda:~$ chown -R websphere:websphere /opt/IBM/WebSphere/AppServer/pgsql/
```

Now check that the permissions have been granted:

```
james@andromeda:~$ ls -la /opt/IBM/WebSphere/AppServer/pgsql/
drwxr-xr-x 2 websphere websphere 4096 Oct 31 09:49 .
drwxr-xr-x 39 websphere websphere 4096 Oct 31 10:09 ..
-rw-r--r-- 1 websphere websphere 505233 Oct 31 09:49 postgresql-42.1.4.jar
```

Step 2 - Increase the JVM initial and max heap sizes for WAS

It is necessary to increase the JVM initial and max heap sizes for WAS either in the in WAS console GUI or else by editing the **server.xml** file through command line. To edit the server.xml file, use the following instructions.

First make a backup copy of the current server.xml file. Navigate to the **profile-root/config/cells/nodes/servers** folder. By default this will be located here: **/install_root/profiles/<profile_name>/config/cells/<cellname>/nodes/<nodename>/servers/server1/server.xml**

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/Dmgr01/config/cells/mopyz6160104Cell01/nodes
/mopyz6160104CellManager01/servers/dmgr/
james@andromeda:~$ cp server.xml backup_server.xml
```

Edit the server.xml file and search for the **<processDefinitions>** and **<jvmEntries>** tags at the end of the file. Update these entries as follows: **"initialHeapSize="1280" maximumHeapSize="2048"**:

```
james@andromeda:~$ vi server.xml
```

For example:

```
<jvmEntries xmi:id="JavaVirtualMachine_1183122130078" verboseModeClass="false" verboseModeGarbageCollection="
true" verboseModeJNI="false" initialHeapSize="1280" maximumHeapSize="2048" runHProf="false" hprofArguments=""
debugMode="false" debugArgs="-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=7777"
genericJvmArguments="-Xdisableexplicitgc -Djava.awt.headless=true -Xjit:{com/ibm/db2/jcc/*}
(disableIdiomRecognition)">
</jvmEntries>
```

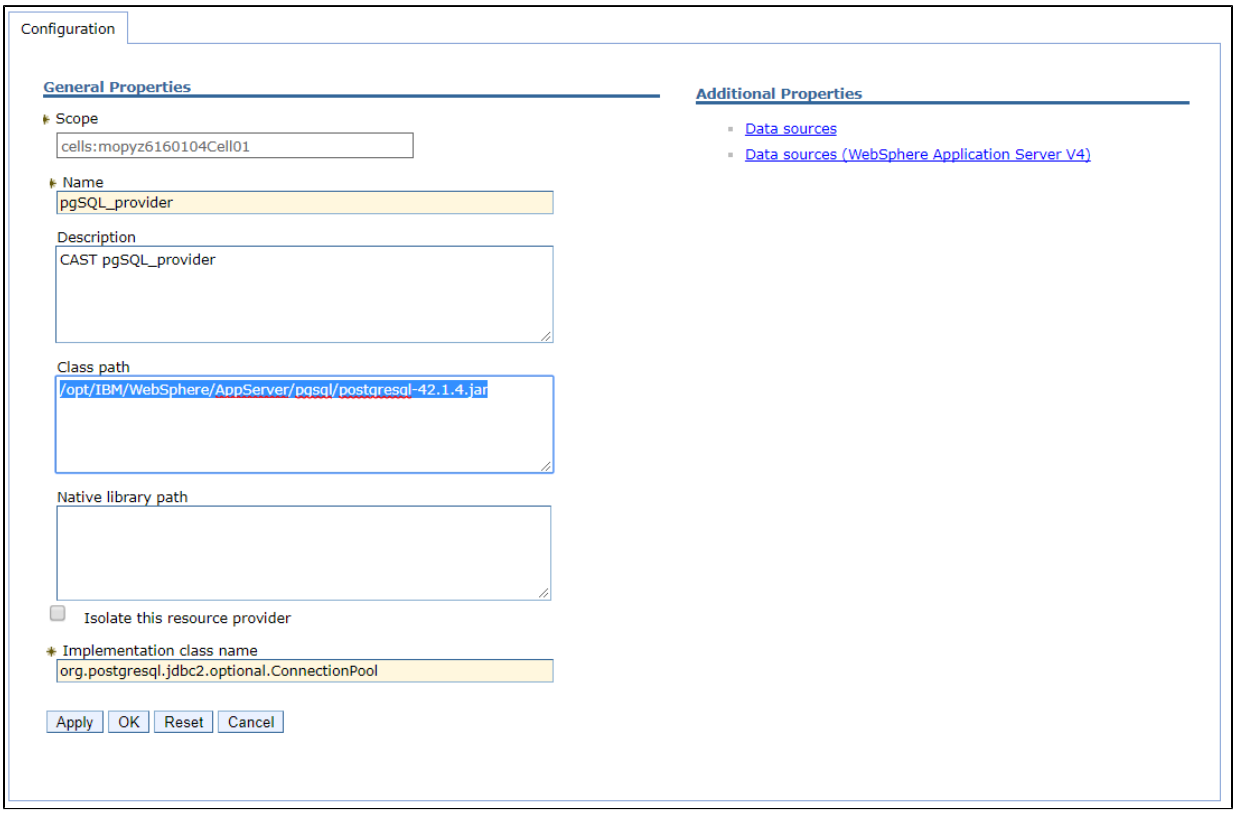
Save the server.xml file and quit the editor. Restart WAS to take into account the new heap sizes:

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/bin/
james@andromeda:~$ ./startManager.sh
```

Step 3 - Configure JDBC provider in WAS Console

- In the WAS console, go to Resources > JDBC > JDBC provider and click **New**.
- In the JDBC providers window, select the scope of the JDBC Provider, in our case we want it available across the entire Cell, and click **New**:
  - **Scope**: cells:mopyz6160104Cell01 (i.e. the equivalent in your environment)
  - **Databse type**: user-defined
  - **Implementation class name**: org.postgresql.jdbc2.optional.ConnectionPool
  - **Name**: PSQL_provider
  - **class path**: /opt/IBM/WebSphere/AppServer/pgsql/postgresql-42.1.4.jar (this is the path where you have copied the .jar file as described in Step 1)

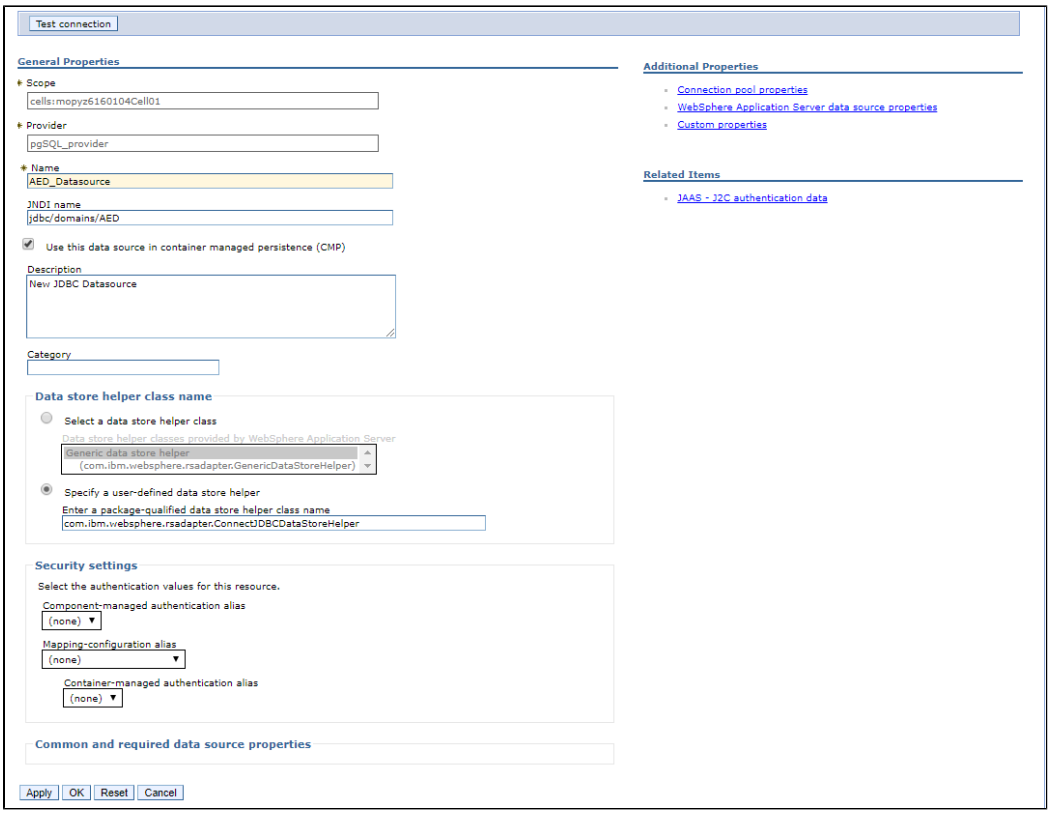
Click to enlarge:



Step 4 - Configure data sources

- In the WAS console, go to Resources > JDBC > Data sources choose the scope (in our case the Cell) > New
- Select a display name for the datasource name:
  - **Data source name**: pg_radocea_database
  - **JNDI name**: jdbc/domains/AED **or** jdbc/domains/AAD (depending on the WAR you are installing
- Click **Next** and move to "**Step 2: Select JDBC provider**".
- Click "**Select an existing JDBC provider and select PgSQL_provider**"
- Move to "**Step 3: Enter database specific properties for the data source**" and enter a "Data store helper class name: **com.ibm.websphere.rsadapter.ConnectJDBCDataStoreHelper**
- Go to the Summary section, click **Finish** and save the changes to the master configuration:

Click to enlarge:

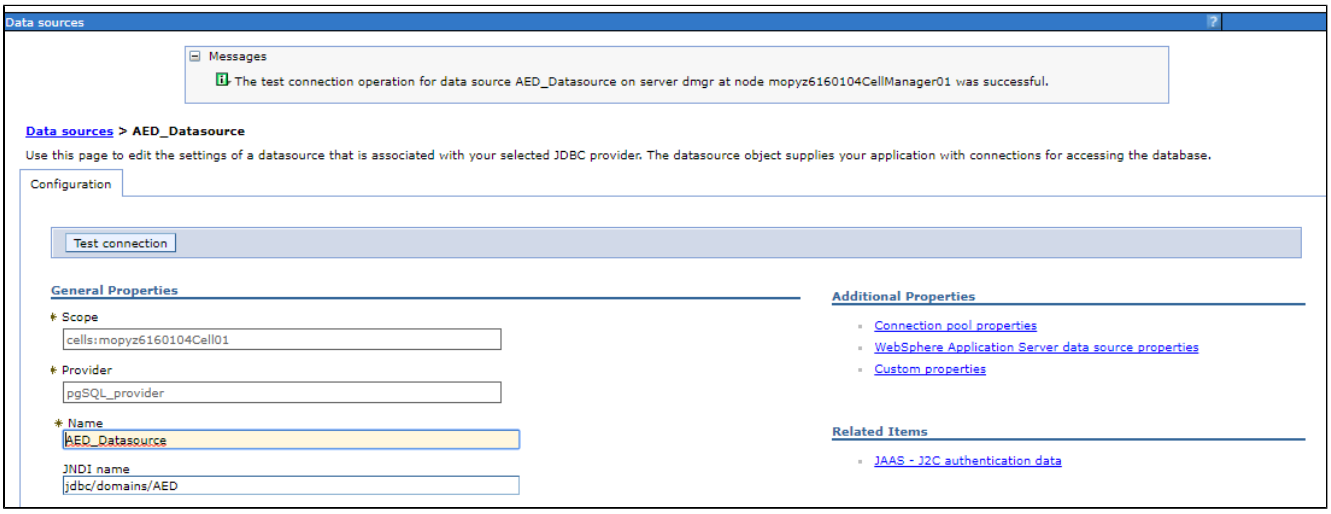


- Click on your newly added datasource name, go to the "**Custom properties**" section and configure the **schema**, **user**, **password** and **port number** for your CAST Storage Service (i.e. PostgreSQL instance). Make sure you set the **Variable Type** to "**java.lang.String**" for all properties:

```
databaseName: postgres ;currentSchema: 82_central; user:operator; password: CastAIP; portNumber: 2280
```

- Save the changes.
- Now click "**Test connection**" and ensure a "successful" result is returned:

Click to enlarge:

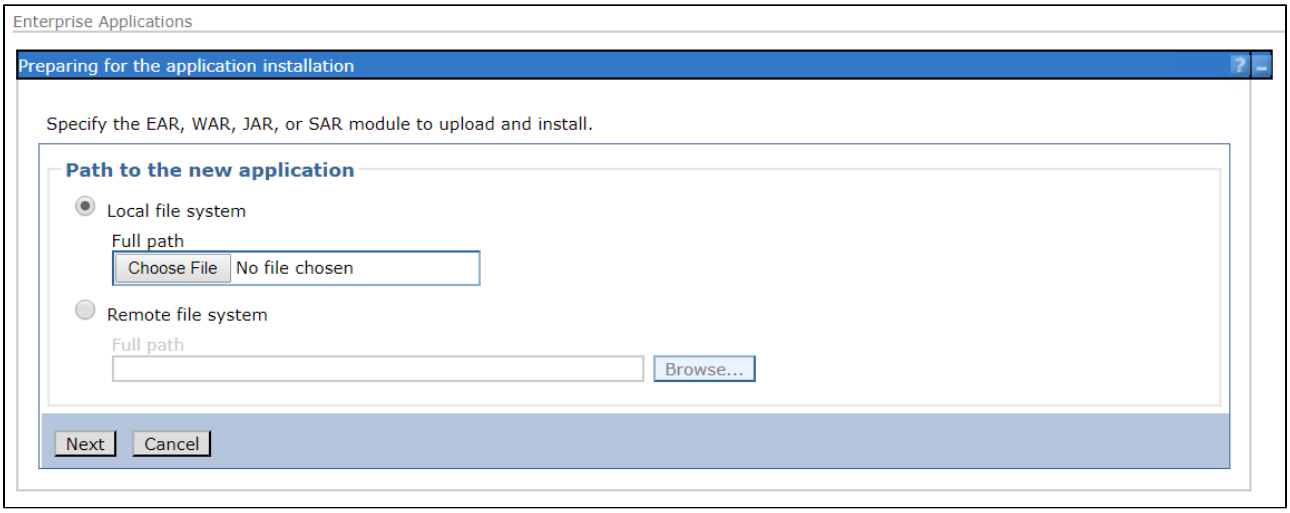


If you have any errors while testing the connection, please consult the following log to help debug the issue: **/opt/IBM/WebSphere/AppServer/profiles/Dmgr01/logs/dmgr/SystemOut.log**

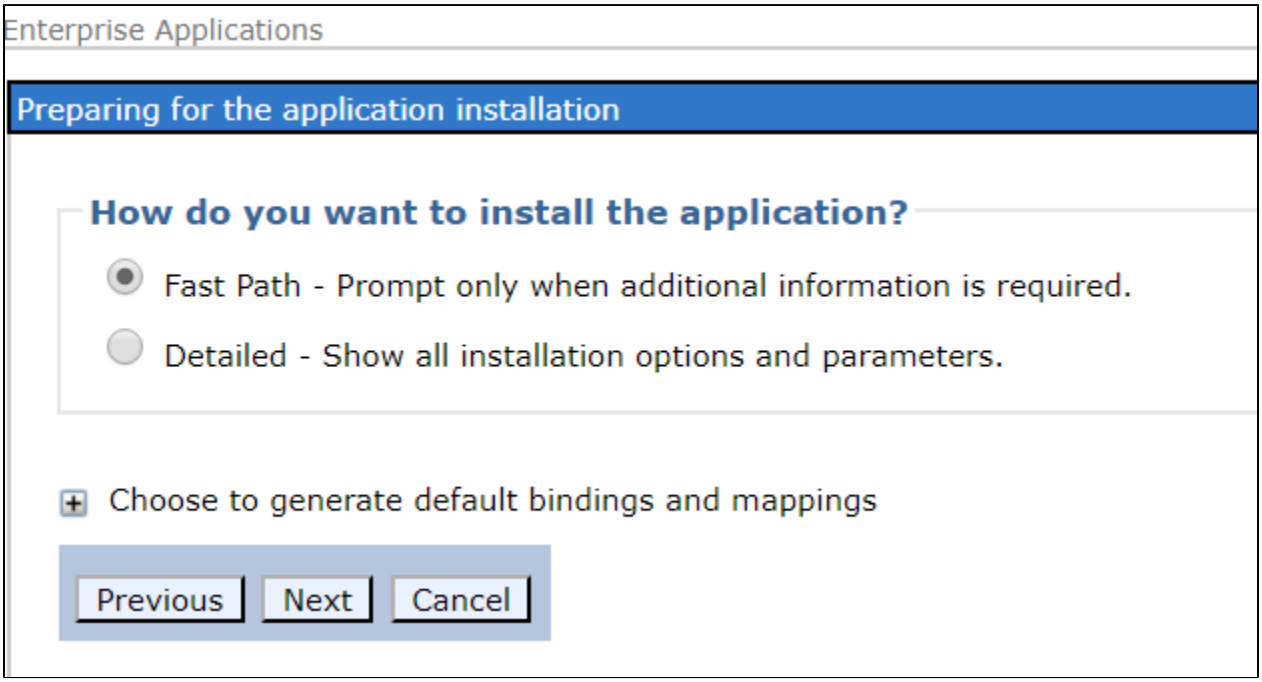
Step 5 - Deploy WAR files in WAS Console

Note that the WAR files you want to deploy must already be present on the local machine.

- In the WAS console click Applications > New Application > New Enterprise Application:

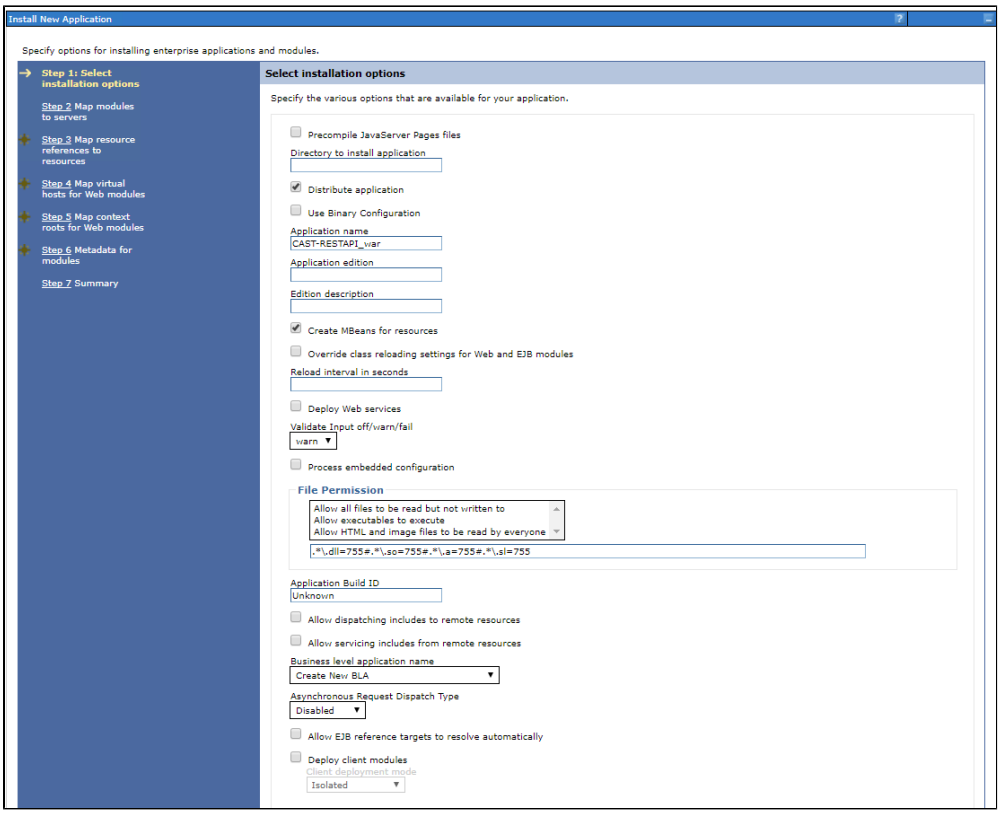


- Select **Local file system** option and then locate your war file (for example CAST-Health.war)
- Click **Next** and wait for a few minutes for the war deployment to complete.
- Select the "**Fast Path**" option to install the application and then click **Next**:



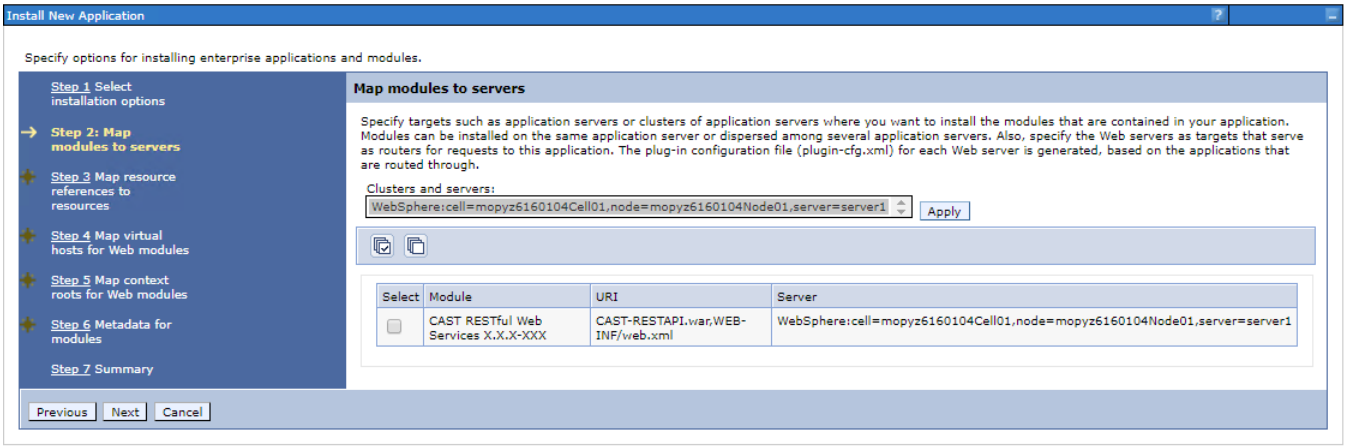
- The next page comprises a five step wizard to install the application:
- **Step 1: Select installation options** - fill in as shown in the image below. Click **Next** to continue:

Click to enlarge:



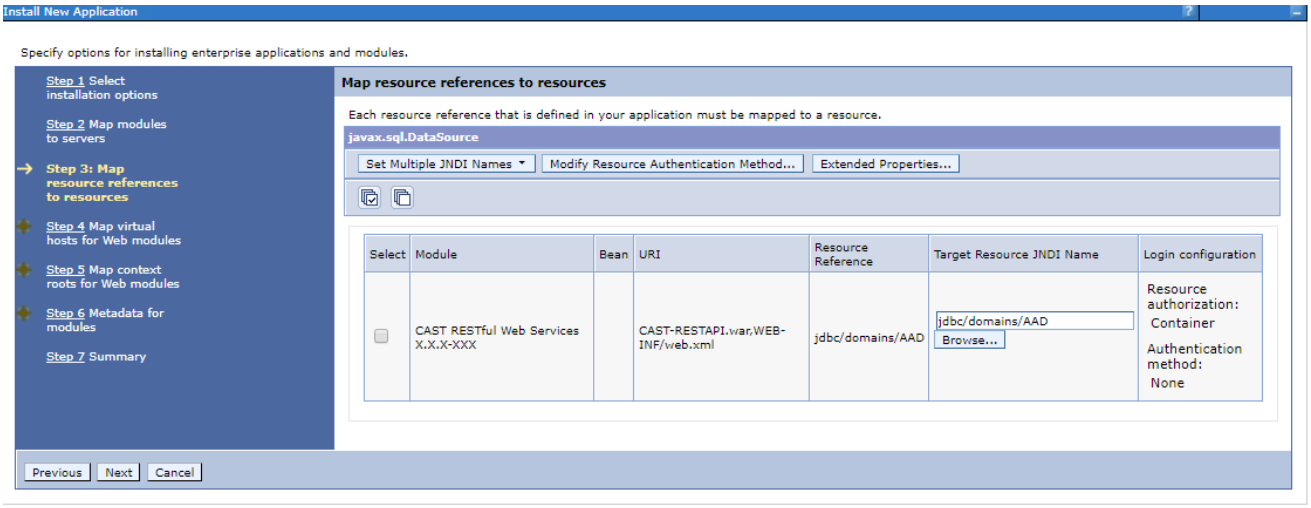
- **Step 2: Map modules to servers**: Make sure the "**Clusters and servers**" option is set correctly. Click **Next** to continue:

Click to enlarge:



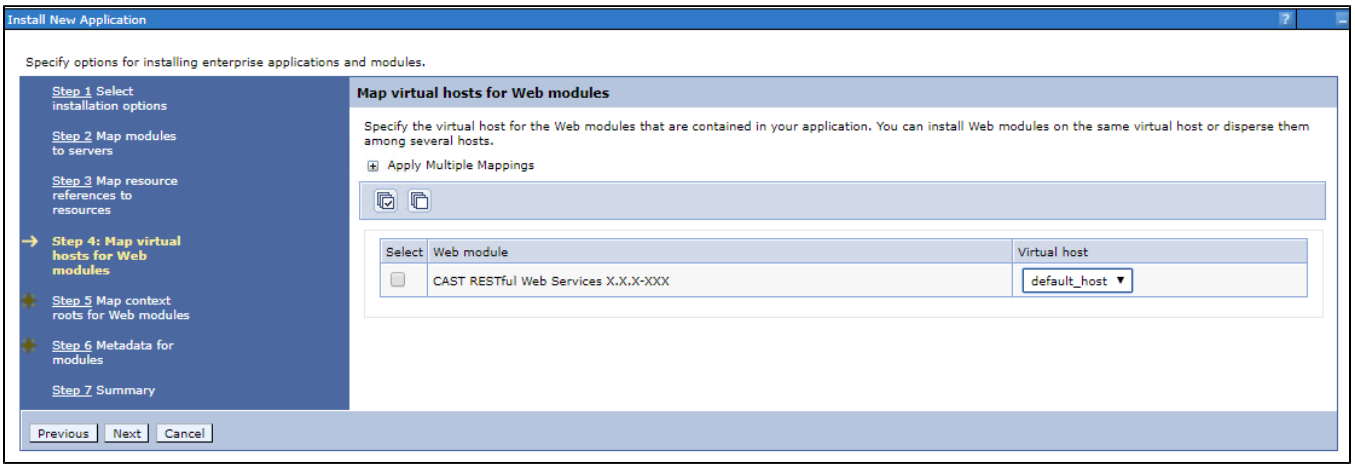
- **Step 3: Map resource references to resources**: Go to the **Target Resource JNDI Name** column and click **Browse** to select the JNDI name that matches the WAR you are deploying (**jdbc/domains/AAD** or **jdbc/domains/AED**). Your confiugration should be as below. Click **Next** to continue:

Click to enlarge:



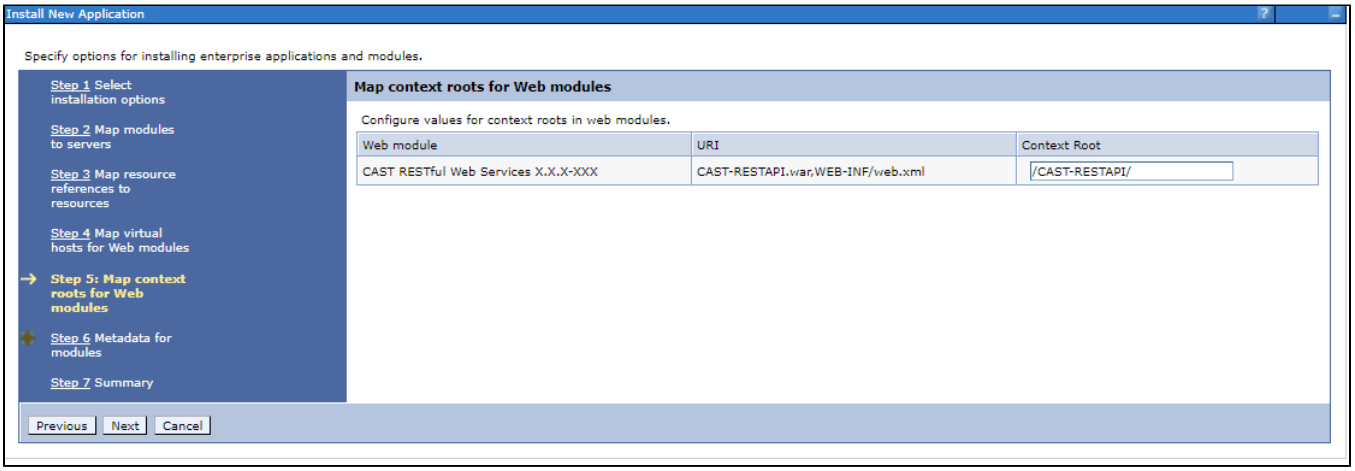
- **Step 4: Map virtual hosts for Web modules**: make sure the virtual host is set to "**default_host**". Click **Next** to continue:

Click to enlarge:



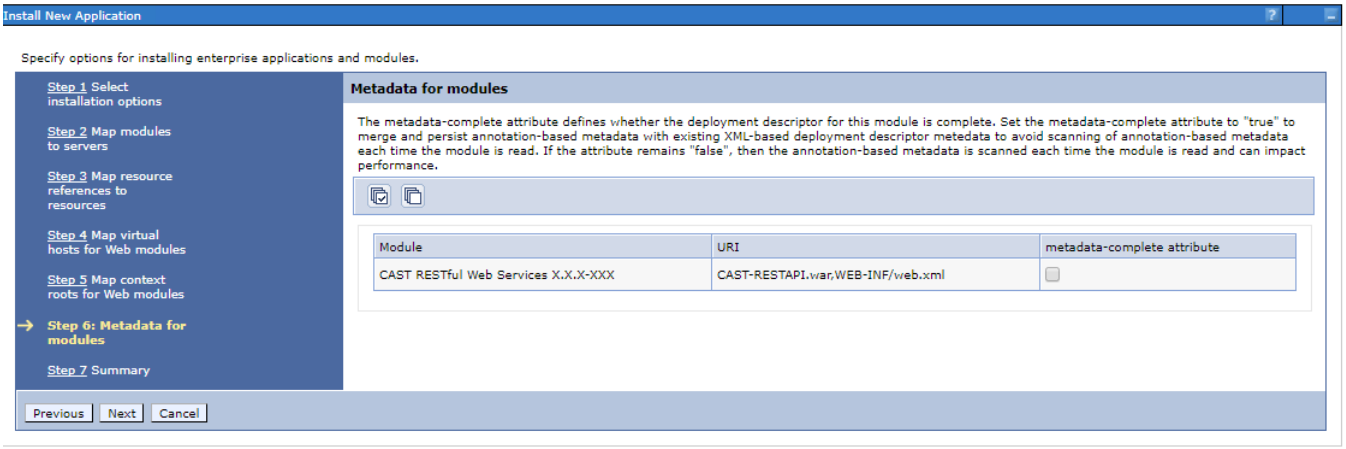
- **Step 5: Map context roots for Web modules**: enter your war file name in **Context Roo**t, for example: "/CAST-Health/" or "/CAST-Engineering /". Click **Next** to continue:

Click to enlarge:



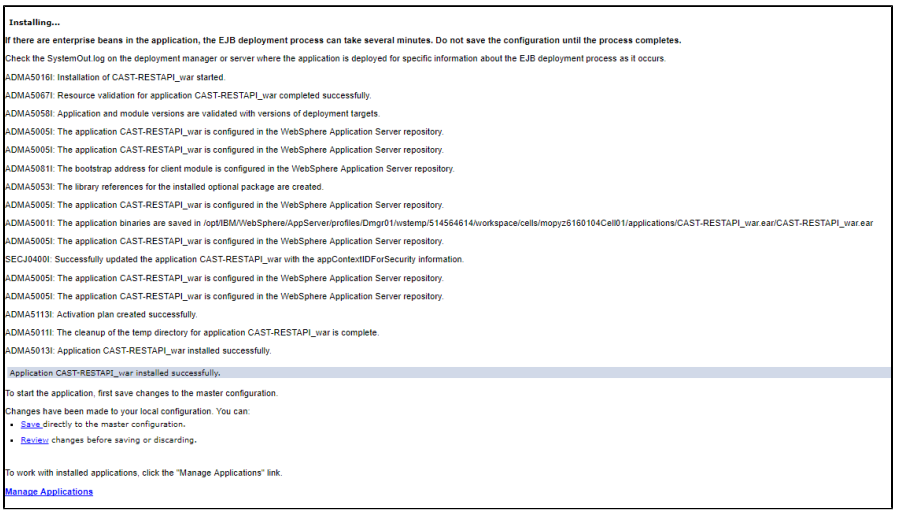
- **Step 6: Metadata for modules**: do not make any changes. Click **Next** to continue:

Click to enlarge:



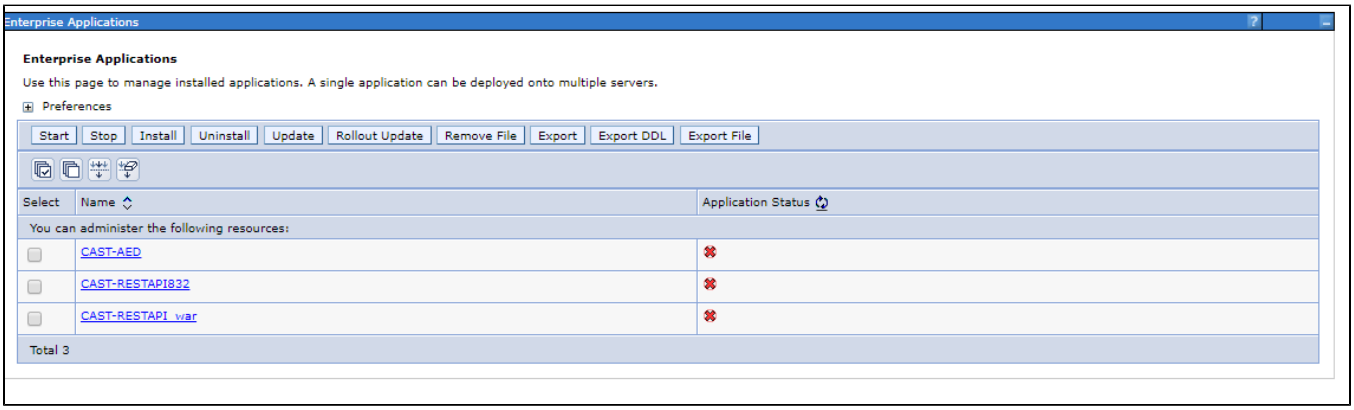
- **Step 7: Summary**: check the options listed in the summary and then click **Finish** (it will take a few minutes to complate). Make sure you shoud not get any errors during the installation:

Click to enlarge:



- Click the "Save directly to the master configuration" (assuming there were no errors listed).
- Now go to Applications > All Applications . You will be able to see the list of all installed WARs. Ensure your WAR is listed:

Click to enlarge:



- Select your application and click **Start**.

Step 6 - Configure dashboards

WARs are deployed in the following location:

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/installedApps/<cellname>/
james@andromeda:~$ ls
CAST-Engineering.war
```

Configure user authentication/authorization and license key

Navigate to the contents of the deployed WAR file and complete the configuration as follows. Ensure you save all modified configuration files and then restart your application and the server to ensure the new settings are taken into account:

- User authentication
- Data authorization
- Dashboard Service license key configuration

Configure log

By default, the log folder will be set to **$${web:rootDir}/logs** which makes it difficult to access the logs. Therefore CAST recommends configuring a custom location for the WAR log files. Please refer to HD-ED - Configuring the Log and Audit Trail for more information.

- Edit the **log4j2.xml** file in the deployed WAR folder and change the configuration: create a unique directory for each WAR, for example **CAST-Health/logs**)

```
<Properties>
        <Property name="logPath">CAST-Health.war/logs</Property>
                <!--
                Set the auditLevel property's value to enable/disable the audit trail:
                        - OFF -> audit trail is disabled
                        - ALL -> audit trail is enabled
                -->
                <Property name="auditLevel">OFF</Property>
</Properties>
```

- Save the **log4j2.xml** file.
- Restart your application and the server to ensure the new settings are taken into account.

Step 7 - modify Jersey settings

There may be some conflicts between the Jersey version embedded in the CAST WARs and the native Jersey version used in WebSphere, as such, CAST recommends disabling the Jersey version in WAS:

- In the WAS console, go to: Servers > All Servers > <Your Server> > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties > New and set the following settings:
  - **Set Name** = com.ibm.websphere.jaxrs.server.DisableIBMJAXRSEngine
  - **Set Value** = true

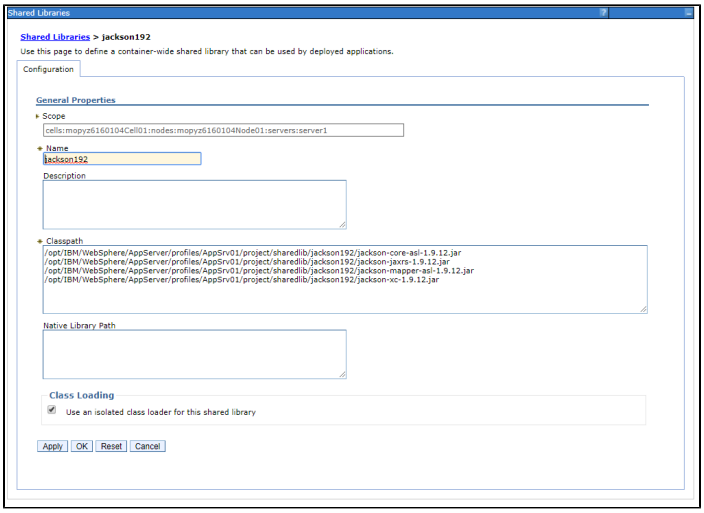
Step 8 - create shared library for Jackson .JAR files

CAST highly recommends creating a **shared library** for the Jackson related .JAR files that are delivered in the CAST WAR files:

- In the WAS console, define a shared library: Environment SharedLibraries New

- Define a shared library for jackson with the following classpath:
  - **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jackson192**
  - **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jackson192/jackson-xc-1.9.12.jar**
  - **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jackson192/jackson-core-asl-1.9.12.jar**
  - **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jackson192/jackson-jaxrs-1.9.12.jar**
  - **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jackson192/jackson-mapper-asl-1.9.12.jar**
- Ensure you tick "**Use an isolated class loader for this shared library**"

Click to enlarge:

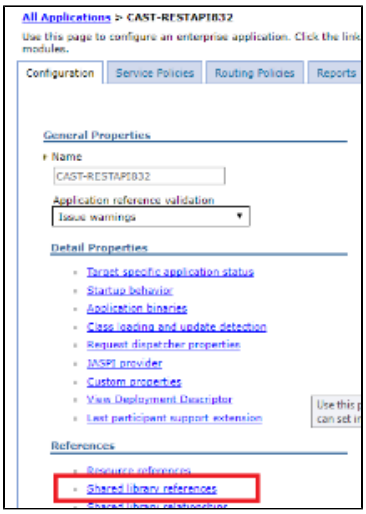


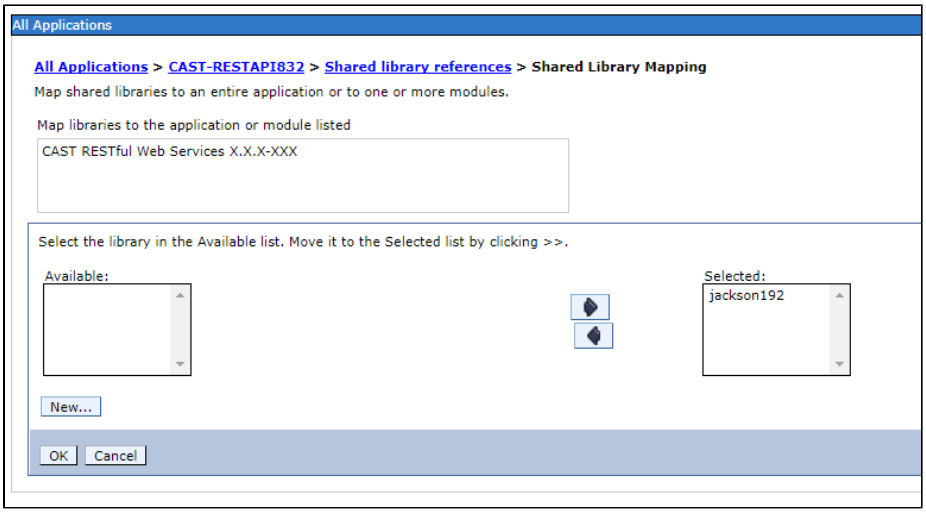
- Locate the **WEB-INF\lib** folder in the deployed WAR file

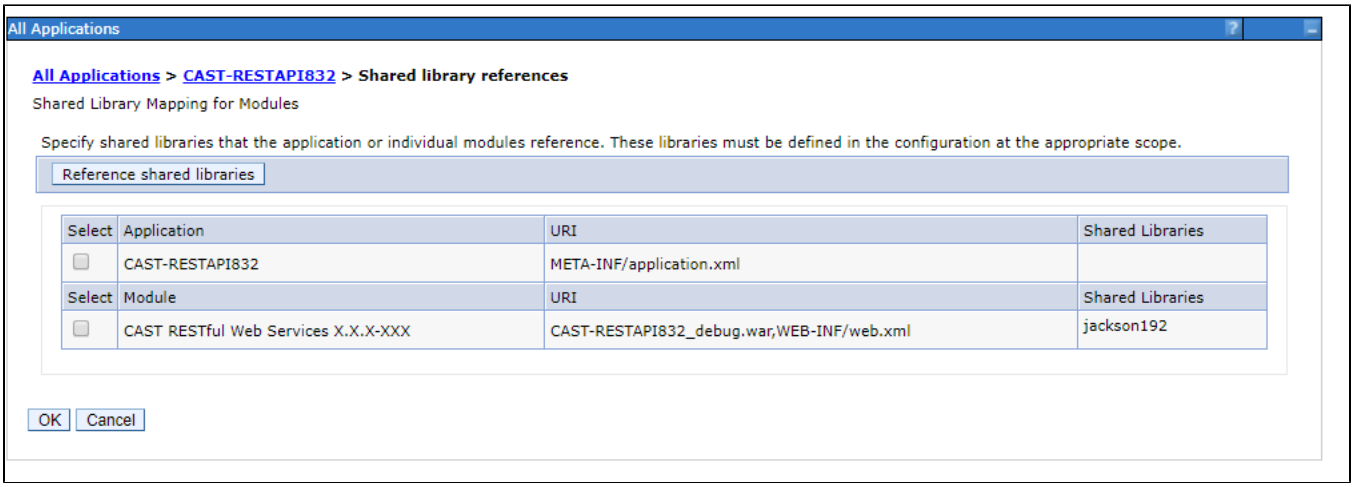
```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/installedApps/<cellname>/CAST-Engineering.war/WEB-INF/lib
```

- Copy all **jackson*.jar** files into **/opt/IBM/WebSphere/AppServer/profiles/<profile_name>/project/sharedlib/jackson192**
- Finally, in the WAS console, select your Application and click "Shared Library Reference", for Application and Module click on "Reference Shared Library" and add the Shared Library you just created:

Click to enlarge:







Step 9 - create shared library for JAXB .JAR files

CAST highly recommends creating a **shared library** for the JAXB related .JAR files that are delivered in the CAST WAR files.

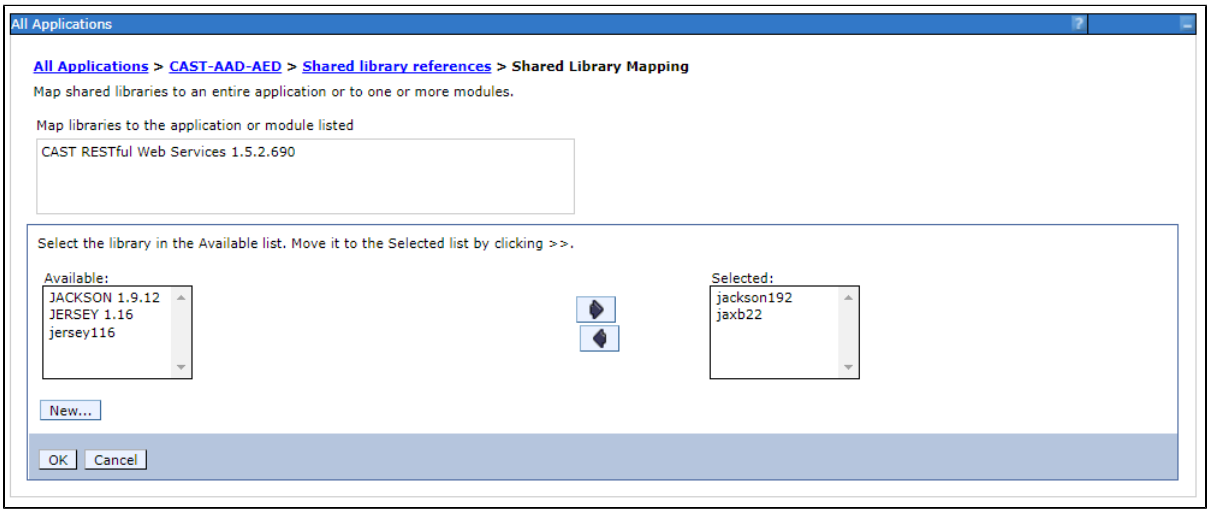
- Use the exact same process described in **Step 8** above and create a shared library under **/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/project/sharedlib/jaxb22**
- Locate the **WEB-INF\lib** folder in the deployed WAR file

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/installedApps/<cellname>/CAST-Engineering.war/WEB-INF/lib
```

- Copy all **jaxb*.jar** files into **/opt/IBM/WebSphere/AppServer/profiles/<profile_name>/project/sharedlib/jaxb22**

Assign these JAR to the shared library for the application:

Click to enlarge:



fsdf<sdf

Tips and tricks

Start the WAS console

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/bin/
james@andromeda:~$ ./startManager.sh
```

Stop the WAS console

```
james@andromeda:~$ cd /opt/IBM/WebSphere/AppServer/profiles/<profile_name>/bin/
james@andromeda:~$ ./stopManager.sh
```

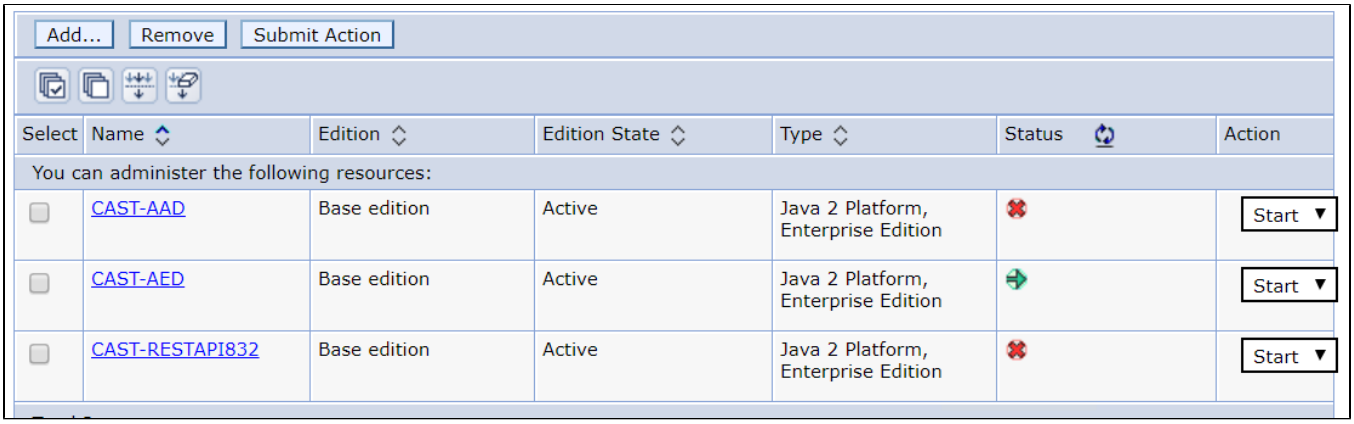
Restart Application Server

You can stop or start the application server from the WAS console. Restarting an application server involves stopping the server, then starting it.

- In the console navigate to Servers > All Server and select your server
- Click the **Stop** button
- You get a message "mopyz6160104Node01/server1 has been stopped" and the status icon will be green in color.
- To start the server, elect your server and click the **Start** button

Restart the Applications (dashboards)

- In the console navigate to Applications> All Applications and select your application (for example: "CAST-Engineering") and select the action "Start" from the dropdown options
- Click **Submit Action**
- To restart a running application, select the application you want to restart, click **Stop** and then click **Start**.


Add...
Remove
Submit Action
Select
Name
Edition
Edition State
Type
Status
Action
You can administer the following resources:
CAST-AAD
Base edition
Active
Java 2 Platform, Enterprise Edition
Start
CAST-AED
Base edition
Active
Java 2 Platform, Enterprise Edition
Start
CAST-RESTAPI832
Base edition
Active
Java 2 Platform, Enterprise Edition
Start