

# Yugandhar Microservice Platform for JEE Container (JEEC) Web Server

## Development Environment Setup Guide

Yugandhar Microservice Platform - JEEC Release - V1.0.0

Date – 23/05/2018

+++++

Copyright [2017] [Yugandhar Microservice Platform]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

## Contents

About Yugandhar Project.....	4
About Yugandhar Microservice Platform .....	4
About this document.....	4
System Requirements .....	5
Software Download links.....	6
Eclipse.....	6
Red Hat JBoss Enterprise Application Platform .....	6
Apache maven.....	6
Database.....	6
Oracle.....	6
MariaDB .....	6
Java JDK .....	7
Hibernate Tools .....	7
Oracle SQL Developer.....	7
Heidi SQL.....	7
Database drivers .....	7
Setup Database.....	7
Oracle setup: .....	8
Oracle Install.....	8
Yugandhar MSP schema setup for oracle.....	8
MariaDB setup: .....	9
MariaDB Install.....	9
Yugandhar MSP schema setup for MariaDB.....	9
Setup Workspace.....	10
Setup up Network Connections.....	12
Eclipse network .....	12
Maven Settings.....	12
Set JDK Path .....	15
Configure RedHat JBOSS EAP.....	19
a. Set the JBoss server port.....	25

b. Add User to Access JBoss Management Console .....	25
c. Create JBoss Datasource and Active MQ Queues .....	27
Import Yugandhar msp java projects in the workspace .....	36
Properties file changes .....	39
application.properties .....	39
yugandhar-msp.properties .....	40
Running the application .....	40
TEST With SOAPUI .....	41
Test JSON message .....	41
Test XML message .....	43

## **About Yugandhar Project**

The Yugandhar Project is the umbrella project focused on building open source cloud ready solutions. The current offerings include Yugandhar Microservice Platform (MSP) and Yugandhar Open Master Data Management (MDM) Hub.

## **About Yugandhar Microservice Platform**

Yugandhar Microservice Platform (also referred as Yugandhar MSP) provides framework for rapid development of your Microservice. This is an architecturally proven Springboot based application having all the basic components needed for a Microservice application to work which just needs to be extended as per your requirements.

Yugandhar Microservice platform codes with code generation templates for rapid development. Just design your data model and generate the code using Yugandhar templates, your base table services will be ready. To create the composite services, you may have to write custom code which would take minimal efforts. Your new Microservice would be ready in just few hours.

## **About this document**

This document covers the system requirements for Yugandhar Microservice Platform.

## System Requirements

Below are the System Requirements for setting up Development Environment

1. OS – Windows 7 enterprise edition, Service Pack 1 or later
2. 8GB RAM and 100 GB Storage
3. Eclipse Java EE IDE for Web Developers. Oxygen.3a Release (4.7.3a) or later
4. JBoss EAP 7.1.0 full runtime or later
5. apache-maven-3.5.0 or later
6. Java jdk 1.8 (jdk1.8.0\_121) or later
7. Spring Boot 2.0.2.RELEASE
8. Hibernate Tools
9. Databases
  - a. Oracle Database 11g Release 11.2.0.2.0 or later
  - b. Oracle 12c
  - c. MariaDB v10.3.x
10. Oracle SQL Developer/HeidiSQL
11. SOAPUI /postman or any other tool to test REST services

**Note** - Internet connectivity is needed to setup workspace. If internet is not available because of any reason then all the software and Maven jars needs to be manually downloaded which would be tedious task.

**IMPORTANT NOTICE** - Please read the licensing terms of all the above listed software before using for commercial as well as non-commercial purpose. Yugandhar team would not be responsible for licensing violations if any.

## Software Download links

### Eclipse

Eclipse Java EE IDE for Web Developers.

**Version** - Oxygen.3a Release (4.7.3a) or later

**Download page** - <http://www.eclipse.org/downloads/>

### Red Hat JBoss Enterprise Application Platform

Red Hat JBoss Enterprise Application Platform

**Version** - JBoss EAP 7.1 full runtime or later

**Download page** - <https://developers.redhat.com/products/eap/download/>

### Apache maven

Apache Maven comes integrated with Eclipse Neon but if you want to install standalone maven then you may download it from below path

**Version** - apache-maven-3.5.0 or later

**Download Page** - <https://maven.apache.org/download.cgi>

### Database

Download the database as per your choice

#### Oracle

Download Oracle from Oracle download site

**Version** - Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit OR Oracle Database 12c

**Download link** - <https://www.oracle.com/database>

#### MariaDB

Version: MariaDB v10.3 or later

<https://mariadb.org/>

## Java JDK

Download Java jdk 1.8 (jdk1.8.0\_121) or later from below link

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

## Hibernate Tools

You may choose to download the Hibernate Tools (Now renamed as JBoss Tools) from the link below. To install the plugin from eclipse market place using eclipse Menu → Help > Eclipse Marketplace... option.

<http://tools.jboss.org/downloads/jbosstools/oxygen/4.5.1.Final.html#marketplace>

## Oracle SQL Developer

Download SQL developer to connect to database

<http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html>

## Heidi SQL

Heidi SQL comes packaged along with Maria DB installable; you may use the same to explore maria db.

## Database drivers

- Oracle JDBC drivers: download ojdbc14.jar from below link  
<http://www.oracle.com/technetwork/apps-tech/jdbc-10201-088211.html>
- Oracle JDBC drivers: Oracle 11g and 12c driver for JDK7 and JDK8 ojdbc7.jar  
<http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html>
- **MariaDB drivers:** Download mariadb-java-client-2.2.3.jar from below location <https://mariadb.com/downloads/connector>

<https://downloads.mariadb.com/Connectors/java/connector-java-2.2.3/mariadb-java-client-2.2.3.jar>

## Setup Database

Install either Oracle or mariaDB database as per requirement. Oracle 11g/12c/MariaDB 5.5.3 are supported. The step by step installation instructions for

installing the Oracle 11g, 12c database and Oracle SQL Developer is out of scope of this document.

By Default Yugandhar Microservice Platform uses the schema YUG\_MSP. If different user name (schema name) is needed then modify all the scripts with required schema name.

## **Oracle setup:**

### **Oracle Install**

Install the Oracle database using the instructions mentioned below.

**Oracle 11g:** [https://docs.oracle.com/cd/E11882\\_01/nav/portal\\_11.htm](https://docs.oracle.com/cd/E11882_01/nav/portal_11.htm)

### **Oracle 12c:**

[http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/12c/r1/Windows\\_DB\\_Install\\_OBE/Installing\\_Oracle\\_Db12c\\_Windows.html](http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/12c/r1/Windows_DB_Install_OBE/Installing_Oracle_Db12c_Windows.html)

### **Yugandhar MSP schema setup for oracle**

Download the scripts from github repository "resources\dbsetupscripts\oracle" location and execute the below scripts in sequence

Github repository link - <https://github.com/yugandharproject/yugandhar-microservices-platform>

Download the scripts from '<gitrepo>/resources/yugmsp-dbsetupscripts-oracle/'

1. "1.yugmsp\_ora\_create\_tablespace\_and\_user.sql" – Needs DBA access
2. "2.yugmsp\_ora\_create\_fullschema.sql" – execute through User YUG\_MSP

Verify the logs and check that all the steps are executed correctly and REF\_xxx as well as CONFIG\_xxx tables are loaded with sample data. In Summary, below mentioned objects to the TABLES, Sequence and INDEXES are created in database

### **Table spaces –**

MSP\_DATATS – used for data and reference tables

MSP\_INDXTS – used for Indexes

### **Profile:**

YUGMSP\_PROFILE - Used to create YUG\_MSP user

### **User Schema:**

YUG\_MSP – Default user Schema used by Yugandhar Microservice Platform.



## MariaDB setup:

### MariaDB Install

Install MariaDB as per instructions mentioned in below links

<https://mariadb.com/kb/en/library/getting-installing-and-upgrading-mariadb/>

<https://mariadb.com/products/get-started>

### Yugandhar MSP schema setup for MariaDB

Download the scripts from github repository resources\dbsetupscripts location and execute the below scripts in sequence

Github repository link - <https://github.com/yugandharproject/yugandhar-microservices-platform>

Download the sqls from '`<gitrepo>/resources/yugmsp-dbsetupscripts-mariadb/`' path of the Yugandhar-Microservice-platform

1. "1.yugmsp\_mariadb\_createuser-and-database.sql" – login through root or DBA

Verify the logs and check that all the steps are executed correctly. Also verify that REF\_XXX as well as CONFIG\_XXX tables are loaded with sample data. In Summary, below mentioned objects to the TABLES, Sequence and INDEXES are created in database

#### **Table spaces:**

MSP\_DATATS – used for data tables as well as reference tables

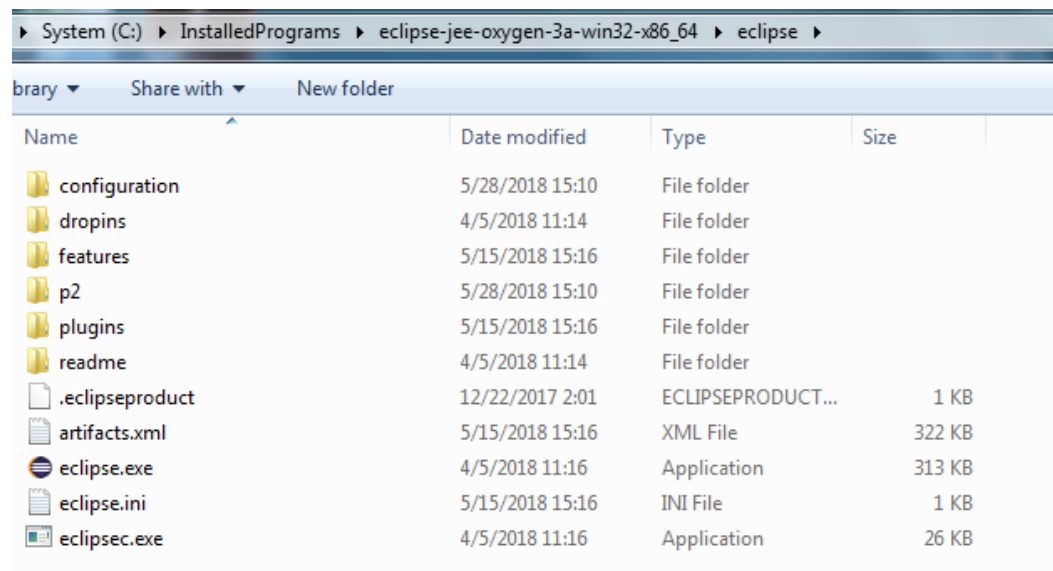
MSP\_INDXTS – used for Indexes

**DB User – YUG\_MSP:** Default user used by Yugandhar Microservice Platform to connect to database.

**Database:** yug\_msp database is created.

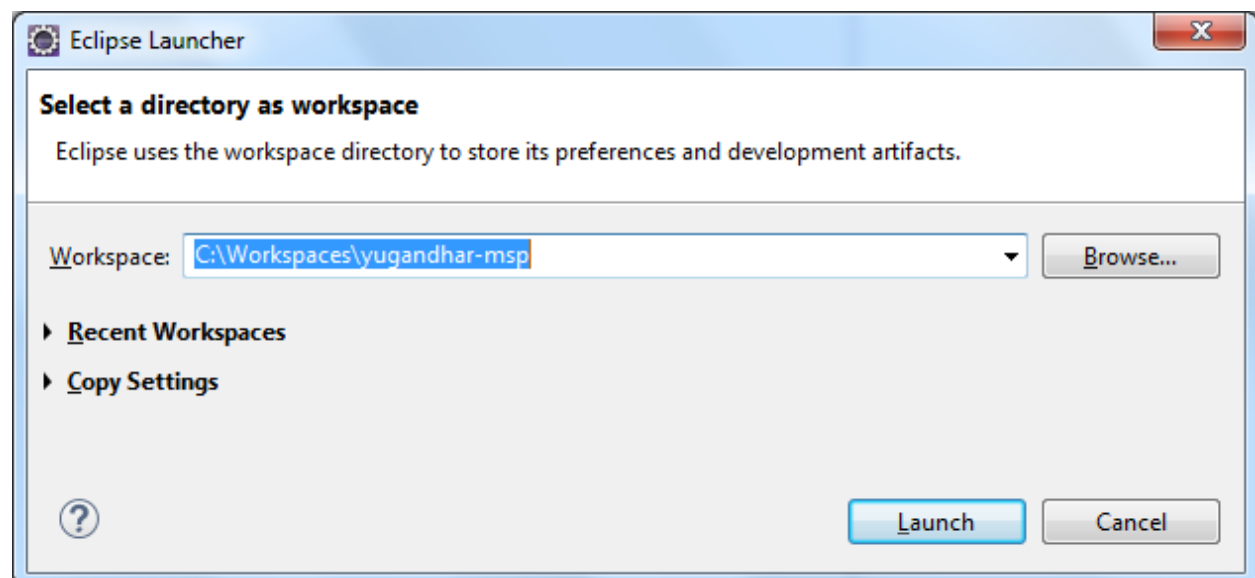
## Setup Workspace

Extract the downloaded eclipse archive in any of the folder of your choice.

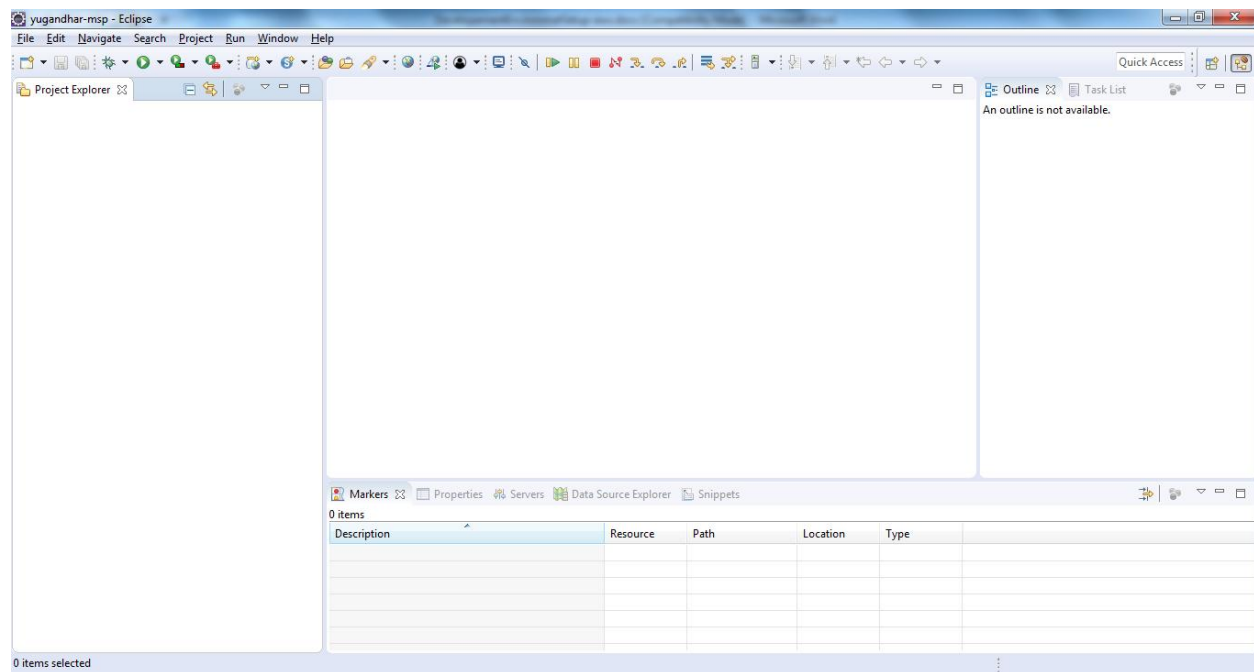


The extracted folder would have above files, click on eclipse.exe to start the eclipse IDE.

The next window would ask you for the workspace location, provide the location as per your choice, for this documentation purpose we are creating workspace in C:\workspaces\yugandhar-msp folder.



Your workspace is ready to be configured now.



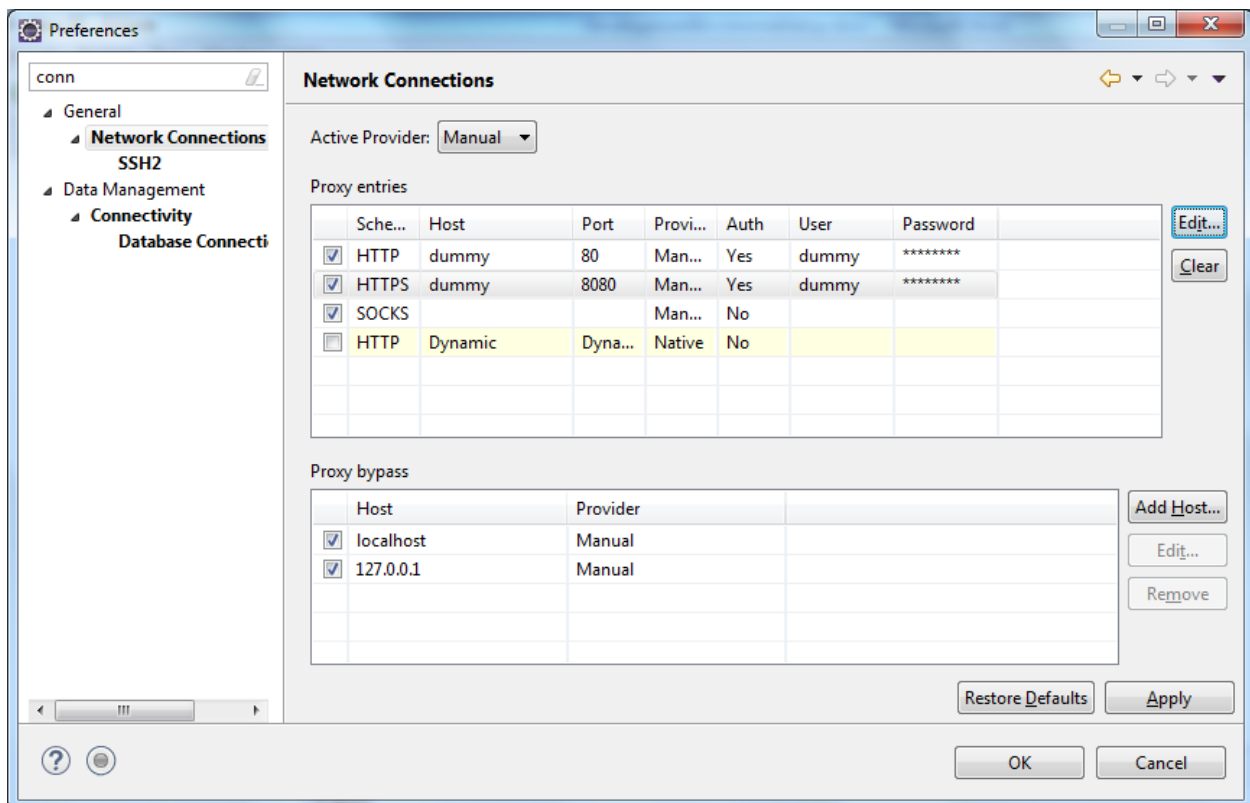
## Setup up Network Connections

If you are behind firewall then you need to perform below steps. Those who have direct internet connection (without proxy) can skip the eclipse Network and Maven Settings step.

### Eclipse network

Go to Windows -- > preferences → General → Network Connections

Change the Active Provider to manual and set the proxy host, port and user credentials as per your firewall settings. The below settings are dummy so should not be copied as-is. Click on apply and click OK.



### Maven Settings

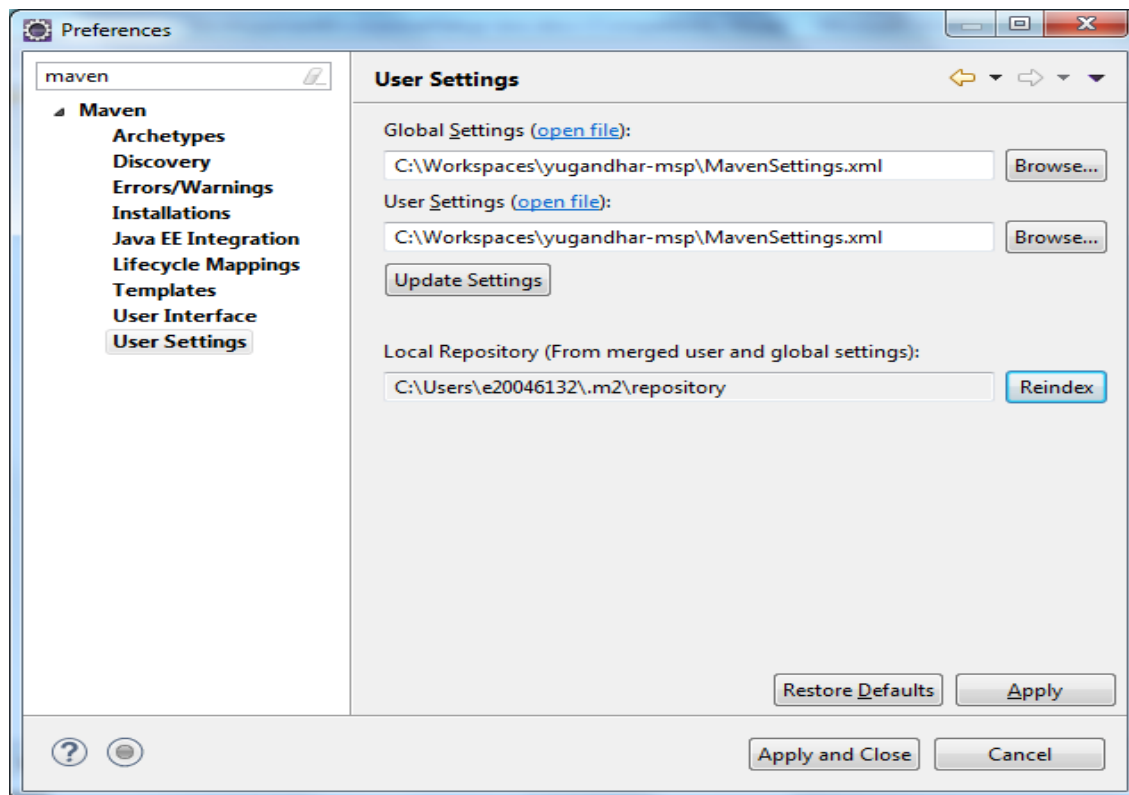
Create a simple text file named MavenSettings.xml in workspace having below content.

You may also download sample MavenSettings.xml from git hub resources/mavensettings folder.

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <localRepository>C:\Users\<UserName>\.m2\repository</localRepository>
  <interactiveMode/>
  <usePluginRegistry/>
  <offline/>
  <pluginGroups/>
  <servers/>
  <mirrors/>
  <proxies>
    <proxy>
      <id>myproxy</id>
      <active>true</active>
      <protocol>http</protocol>
      <host>dummy</host>
      <port> dummy </port>
      <username>dummy</username>
      <password>dummy</password>
      <nonProxyHosts>localhost,127.0.0.1</nonProxyHosts>
    </proxy>
  </proxies>
  <profiles/>
  <activeProfiles/>
</settings>
```

Note – Make sure to change the host, port, username and password as per your firewall settings. Also change the localRepository to users folder.

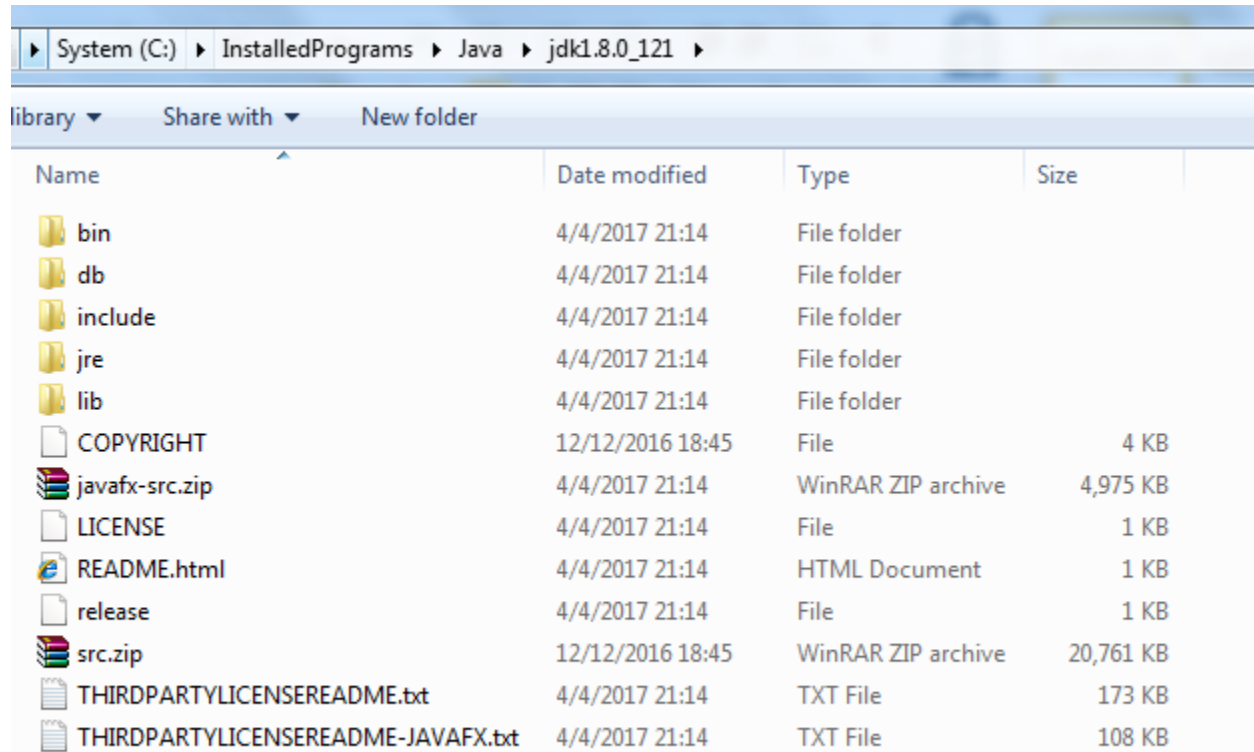
Now go to Eclipse Menu → windows → preferences → maven → User Settings → provide path of the file MavenSettings.xml in the local and global settings as shown in screenshot below



## Set JDK Path

Extract the JDK in a folder of your choice, for the document purpose the jdk directory is

C:\InstalledPrograms\Java\jdk1.8.0\_121. The directory should look something like below



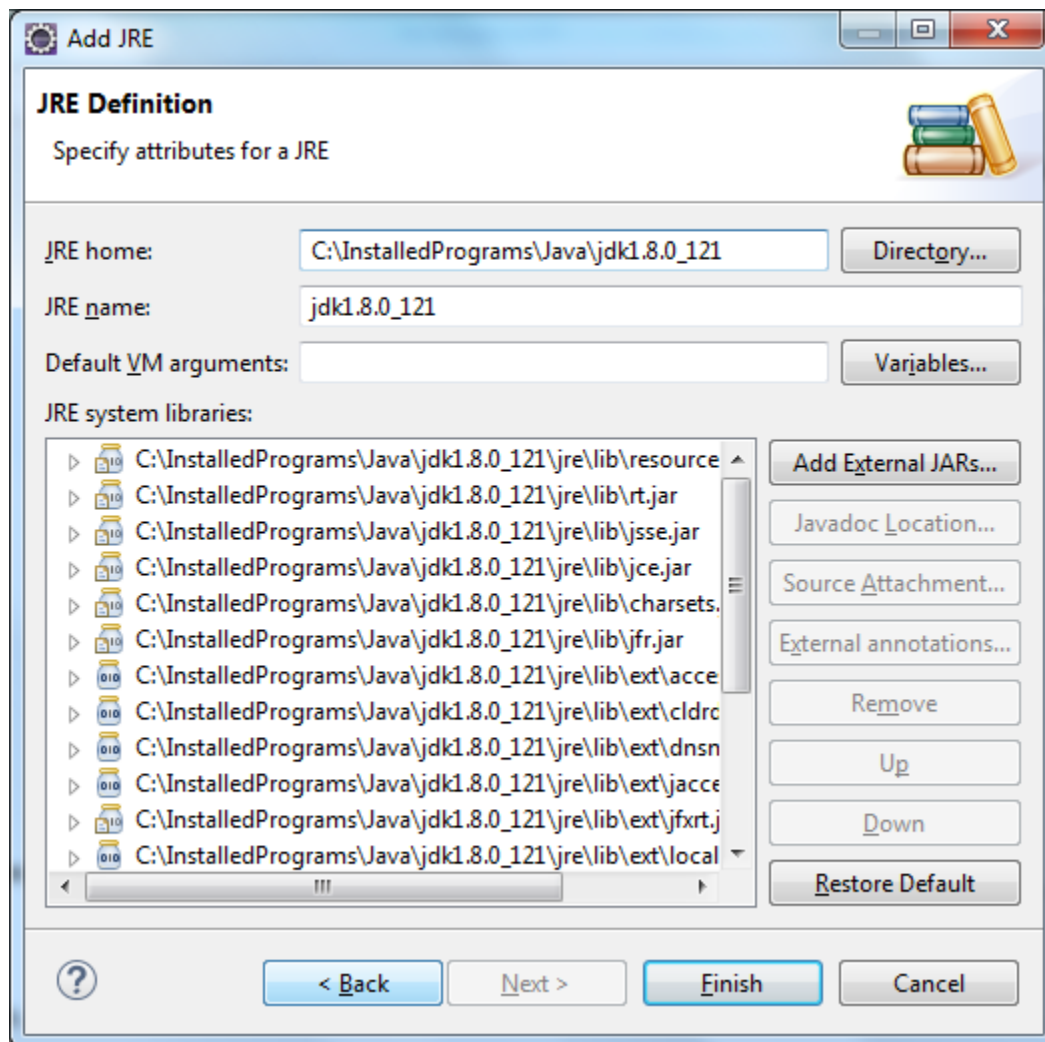
System (C:) > InstalledPrograms > Java > jdk1.8.0_121				
library Share with New folder				
Name	Date modified	Type	Size	
bin	4/4/2017 21:14	File folder		
db	4/4/2017 21:14	File folder		
include	4/4/2017 21:14	File folder		
jre	4/4/2017 21:14	File folder		
lib	4/4/2017 21:14	File folder		
COPYRIGHT	12/12/2016 18:45	File	4 KB	
javafx-src.zip	4/4/2017 21:14	WinRAR ZIP archive	4,975 KB	
LICENSE	4/4/2017 21:14	File	1 KB	
README.html	4/4/2017 21:14	HTML Document	1 KB	
release	4/4/2017 21:14	File	1 KB	
src.zip	12/12/2016 18:45	WinRAR ZIP archive	20,761 KB	
THIRDPARTYLICENSEREADME.txt	4/4/2017 21:14	TXT File	173 KB	
THIRDPARTYLICENSEREADME-JAVAFX.txt	4/4/2017 21:14	TXT File	108 KB	

Go to eclipse Menu → Windows → Preferences → Java → Installed JREs

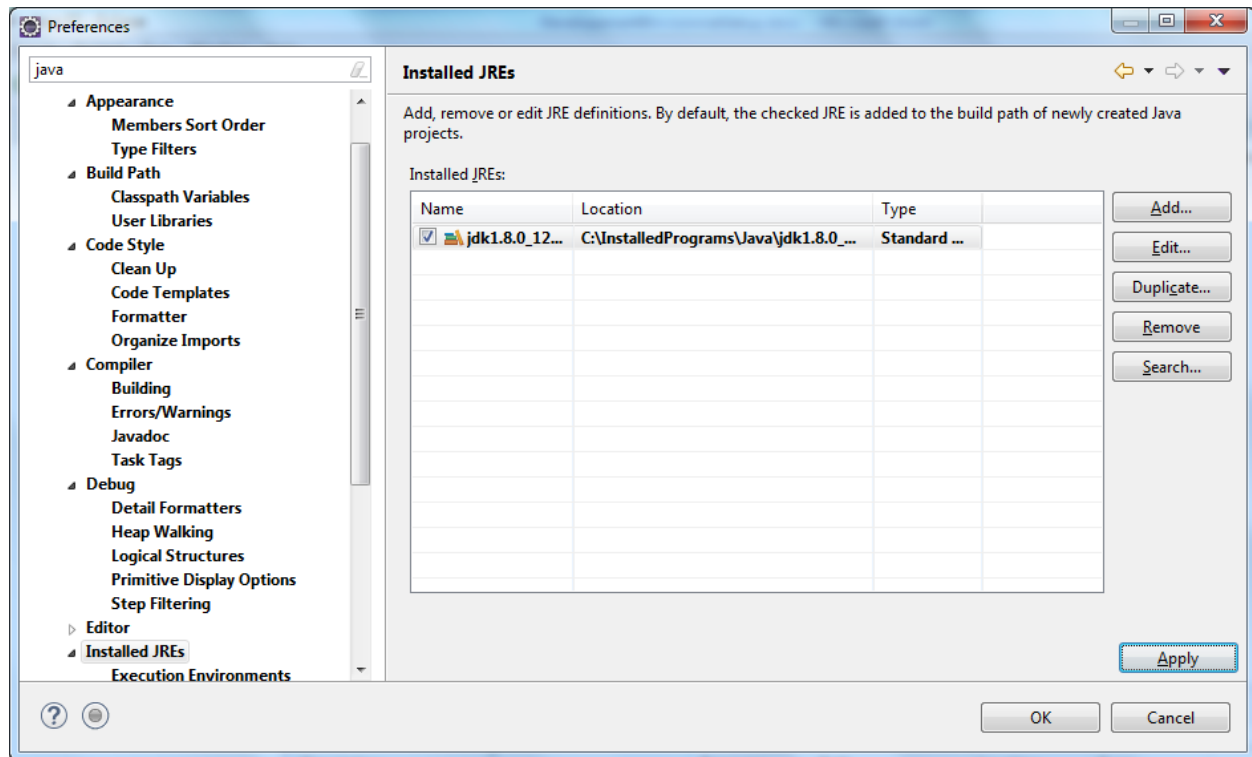




Provide the JDK folder name where JDK is extracted



Click finish

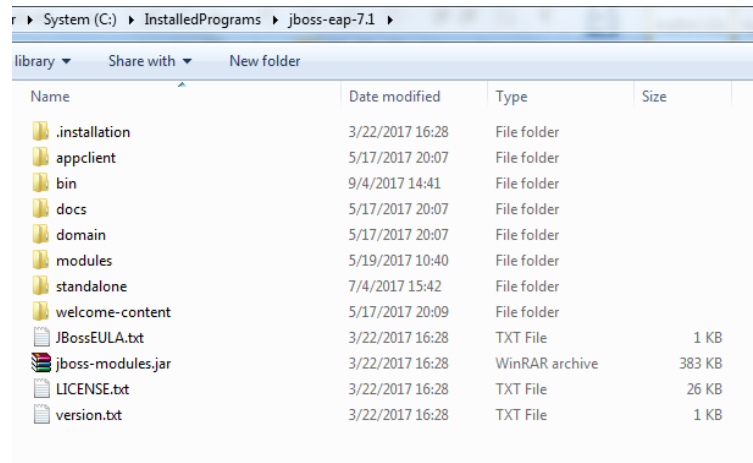


Check the textbox against recently added JDK, click Apply and OK.

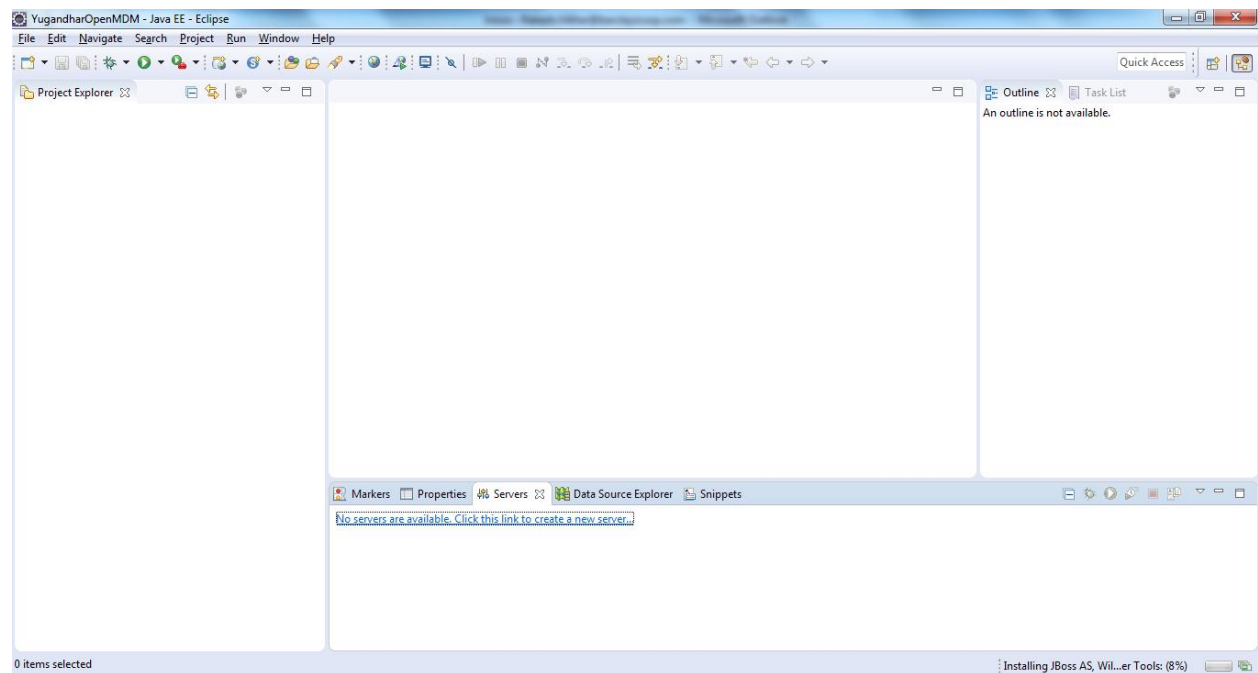
## Configure RedHat JBOSS EAP

Extract the downloaded RedHat Server to a directory of your choice like

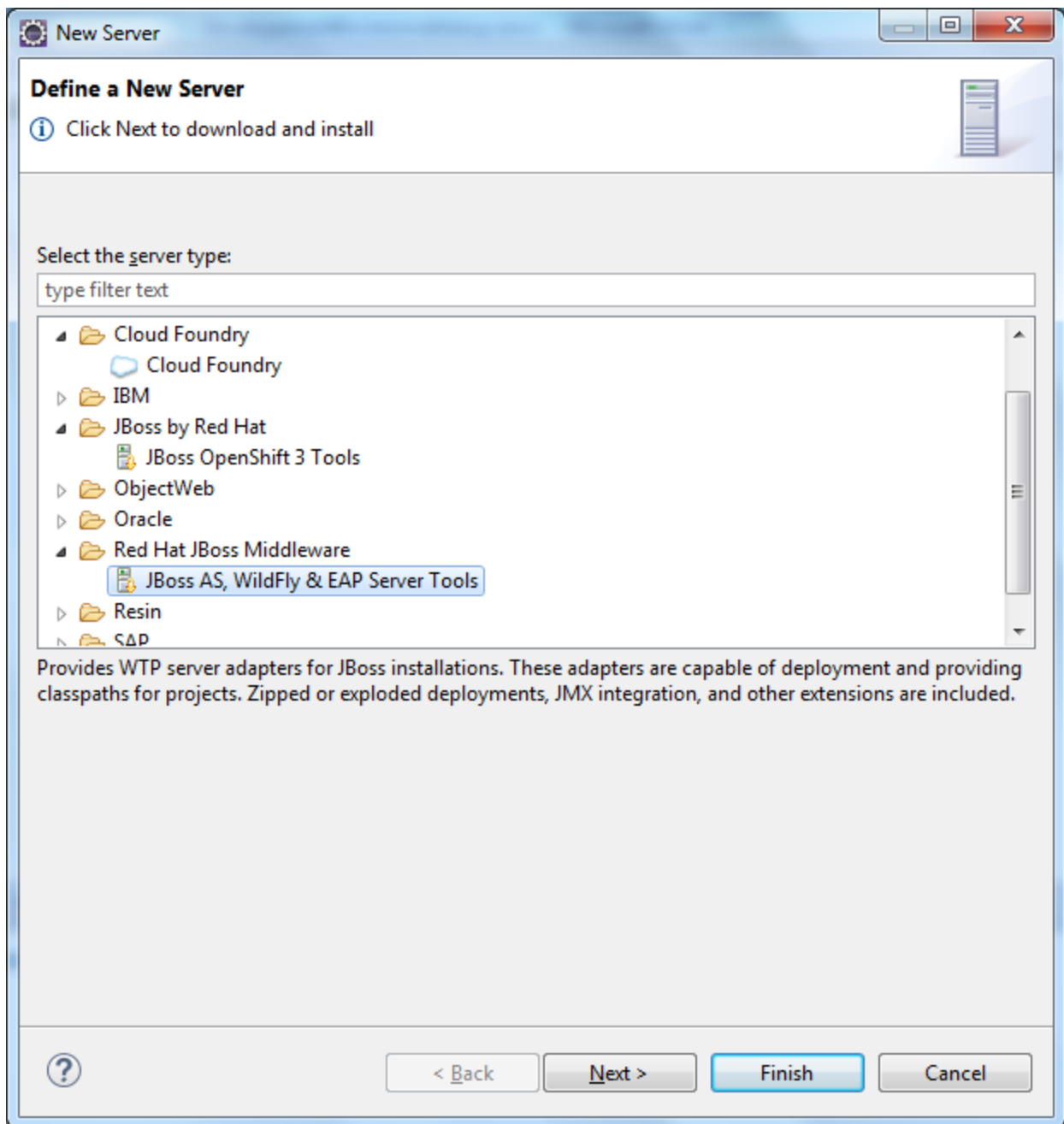
C:\InstalledPrograms\jboss-eap-7.1



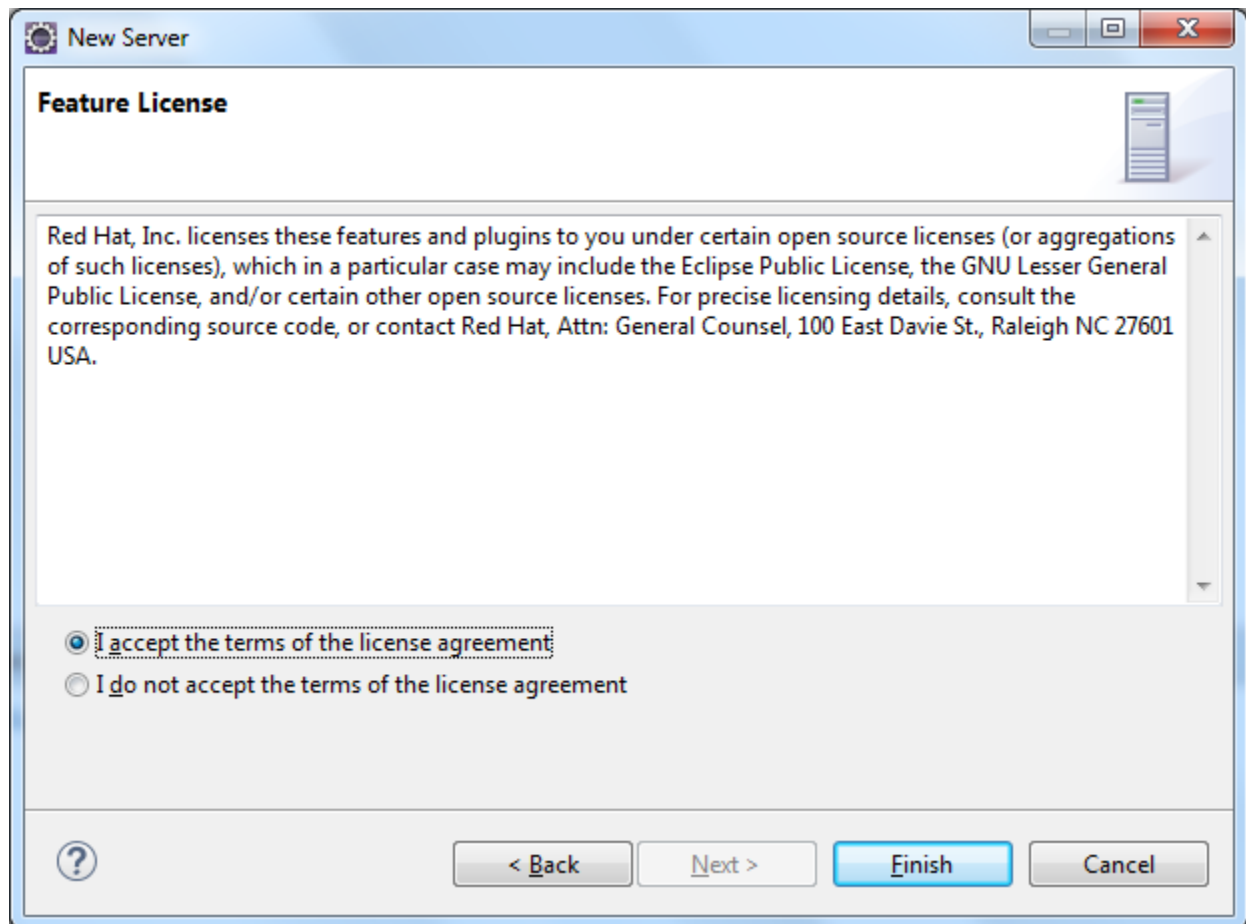
Go to eclipse 'Servers' tab and click on the link to create new Server.



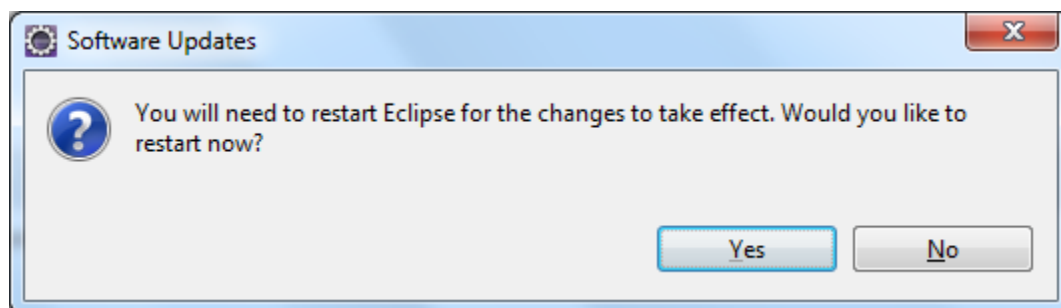
Click the Red Hat Jboss Middleware option as shown in below screenshot



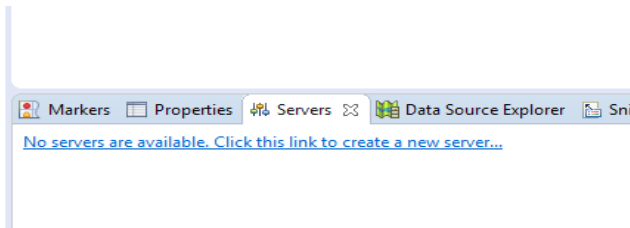
Click Next. It may take some time to process after which will ask to accept the licensing terms. Accept the license and click Finish.



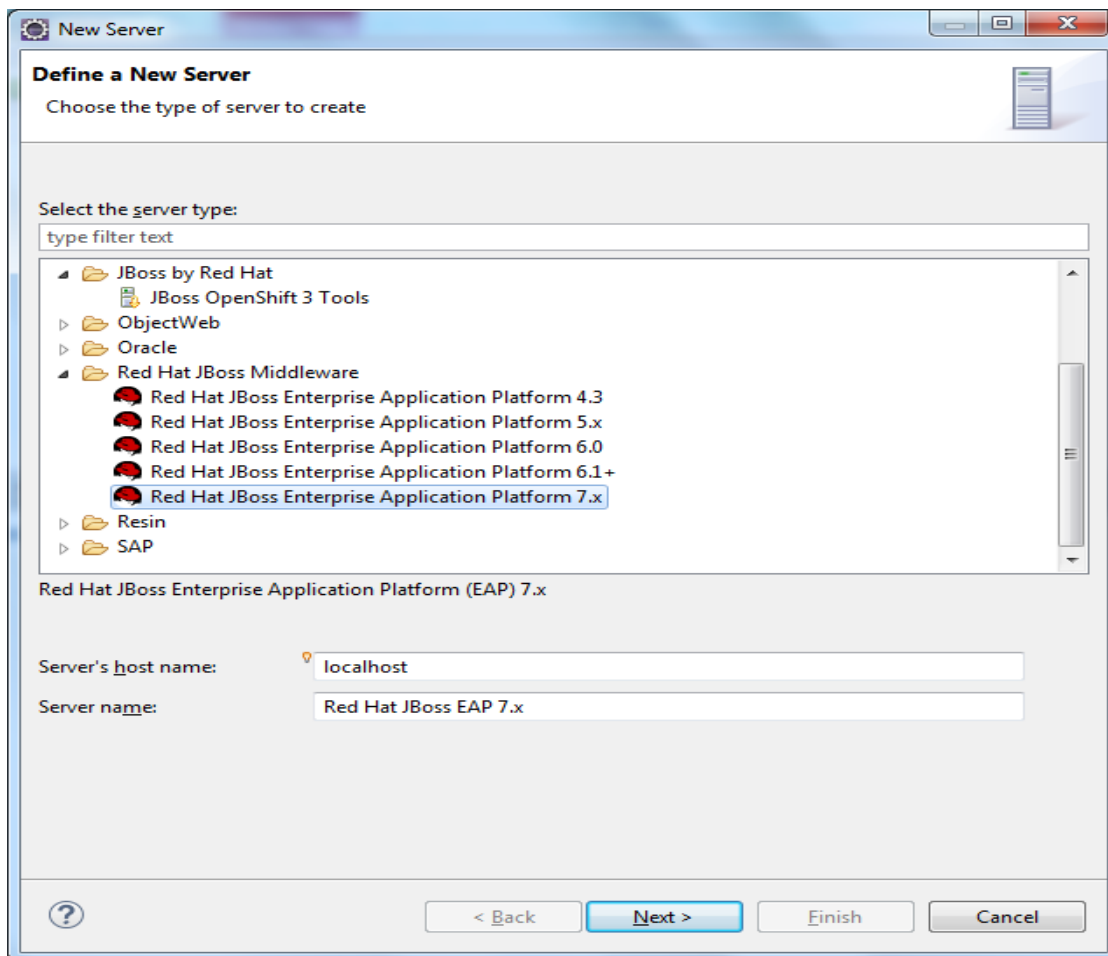
Eclipse will automatically download and install the JBoss Tools which may take some time based on the speed of your internet connection. The eclipse IDE needs to be restarted after this step so that JBoss EAP tools are available in eclipse.



Click on the Server Tab and Create new server link



You would now see Red Hat JBoss Enterprise Application Platform 7.x as an option now. Click on the same and click Next



Click Next

**New Server**

**Create a new Server Adapter**

Red Hat JBoss Enterprise Application Platform (EAP) 7.x

A Server Adapter manages starting and stopping instances of your server. It manages command line arguments and keeps track of which modules have been deployed.

The server is: ☒ Local ☐ Remote

Controlled by: ☒ Filesystem and shell operations ☐ Management Operations

☐ Server lifecycle is externally managed.

The selected profile requires a runtime.

☒ Assign a runtime to this server

Create new runtime (next page) ▼

**Runtime Details**

JRE:

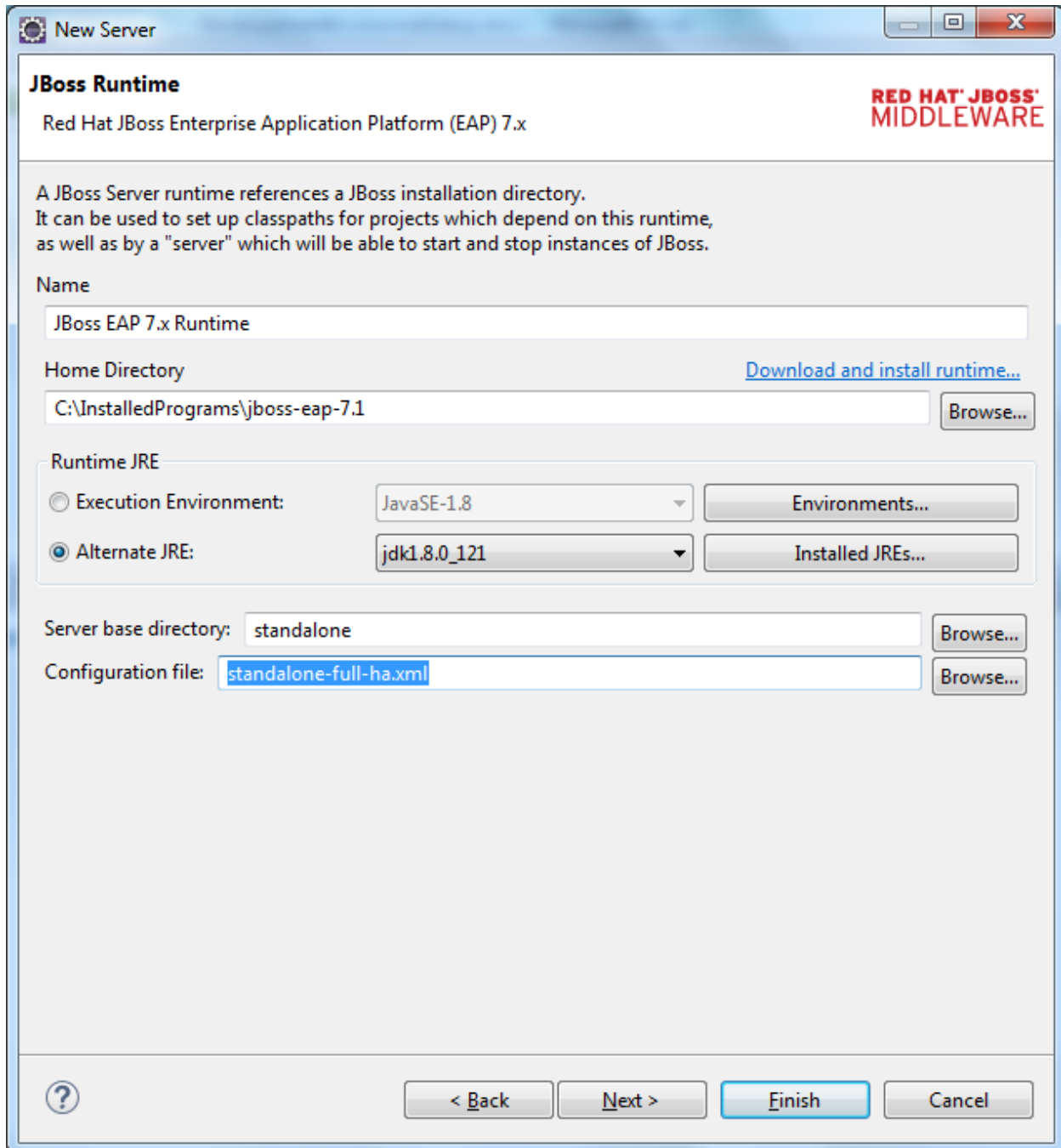
Home Directory:

Base Directory:

Configuration File:

? < Back Next > Finish Cancel

Provide the directory where we have extracted Jboss archive. Also provide the JRE we added in previous steps. Also choose the Configuration file as standalone-full-ha.xml.



**New Server**

**JBoss Runtime**  
Red Hat JBoss Enterprise Application Platform (EAP) 7.x

RED HAT JBOSS MIDDLEWARE

A JBoss Server runtime references a JBoss installation directory. It can be used to set up classpaths for projects which depend on this runtime, as well as by a "server" which will be able to start and stop instances of JBoss.

Name  
JBoss EAP 7.x Runtime

Home Directory  
C:\InstalledPrograms\jboss-eap-7.1 [Download and install runtime...](#) [Browse...](#)

Runtime JRE  
☐ Execution Environment: JavaSE-1.8 [Environments...](#)  
☒ Alternate JRE: jdk1.8.0\_121 [Installed JREs...](#)

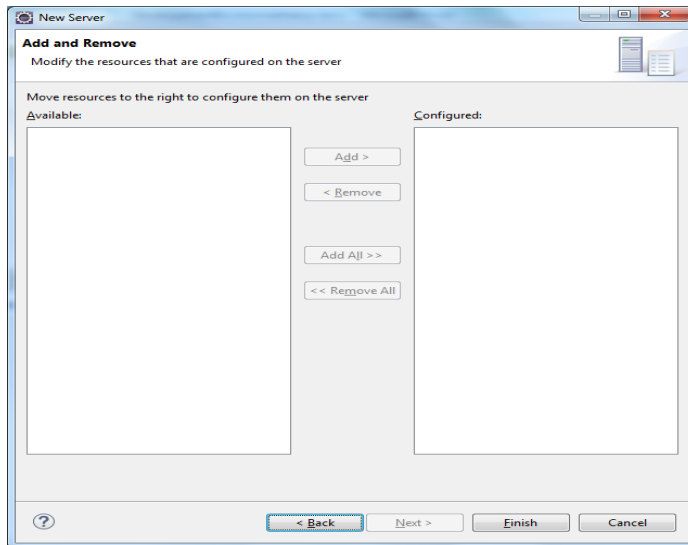
Server base directory: standalone [Browse...](#)

Configuration file: standalone-full-ha.xml [Browse...](#)

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)



Click Next and Finish



You would see the added server in the Servers Tab now. You may click on the start button and check that server is started without errors. Check the logs in Console View and after verification stop the server.

#### **a. Set the JBoss server port**

Modify the below property to set the desired port, for yugandhar msp default port is set to 8091

```
<socket-binding name="http" port="${jboss.http.port:8091}"/>
```

#### **b. Add User to Access JBoss Management Console**

Go to directory where jboss is installed e.g. C:\InstalledPrograms\jboss-eap-7.1.0\bin

Edit the add-user.bat file to set java home, add the jdk home path where you extracted the JDK

e.g. set JAVA\_HOME=C:\InstalledPrograms\Java\jdk1.8.0\_121

```
add-user.bat - Notepad
File Edit Format View Help
@echo off
set JAVA_HOME=C:\InstalledPrograms\Java\jdk1.8.0_121
rem -----
rem Add User script for windows
rem -----
```

Go to command prompt and add the user by running add-user.bat file

Provide mspuser / Testpwd\_123 as username and password (Password may be different of your choice)

```
Administrator: C:\Windows\system32\cmd.exe

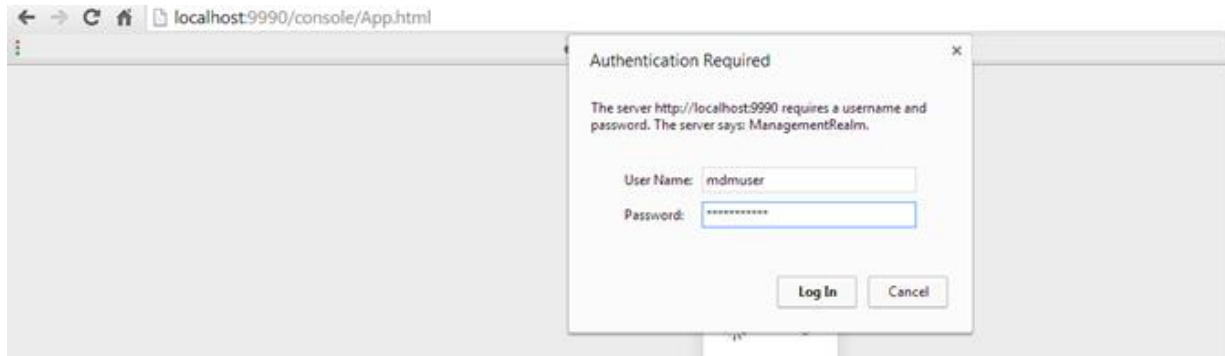
C:\InstalledPrograms\jboss-eap-7.1.0\bin>add-user

What type of user do you wish to add?
  a) Management User <mgmt-users.properties>
  b) Application User <application-users.properties>
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : mdmuser
Password recommendations are listed below. To modify these restrictions edit the
add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
Re-enter Password :
What groups do you want this user to belong to? <Please enter a comma separated list, or leave blank for none>[ ]:
About to add user 'mdmuser' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'mdmuser' to file 'C:\InstalledPrograms\jboss-eap-7.1.0\standalone\configuration\mgmt-users.properties'
Added user 'mdmuser' to file 'C:\InstalledPrograms\jboss-eap-7.1.0\domain\configuration\mgmt-users.properties'
Added user 'mdmuser' with groups to file 'C:\InstalledPrograms\jboss-eap-7.1.0\standalone\configuration\mgmt-groups.properties'
Added user 'mdmuser' with groups to file 'C:\InstalledPrograms\jboss-eap-7.1.0\domain\configuration\mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? yes
To represent the user add the following to the server-identities definition <secret value="UGVzdHB3ZF8xMjM=" />
Press any key to continue . . .

C:\InstalledPrograms\jboss-eap-7.1.0\bin>
```

Type the url `http://localhost:9990/` to login to console and provide username and password



### c. Create JBoss Datasource and Active MQ Queues

Yugandhar Microservice Platform needs JBoss based datasource to connected to Oracle database. The JNDI name of the datasource needs to be provided in `application.properties` file. We need to create either of the below Data source and uncomment the related data source in the properties

```
#spring.datasource.jndi-name=java:/YUGMSP_XAOracle11gDS
#spring.datasource.jndi-name=java:/YUGMSP_XAOracle12cDS
#spring.datasource.jndi-name=java:/YUGMSP_XAMariaDBDS
```

### **Configure Oracle drivers**

Note- Either configure Oracle or MariaDB, no need to configure both databases.

To create Oracle data source in JBoss, You need to manually make some changes on the file system. Also make sure that `ojdbc7.jar` is downloaded on your system from oracle distribution site.

You may take help from the jboss sample configuration file provided in the <github Repository/resources\jbossconfig folder

1. Create a new module for oracle driver.
  - i. Create a folder hierarchy with path `$JBOSS_HOME/modules`.
  - ii. `$JBOSS_HOME/modules/com/oracle/main`
  - iii. Copy `ojdbc7.jar` to `$JBOSS_HOME/modules/com/oracle/main` folder

System (C:) > InstalledPrograms > jboss-eap-7.1.0 > modules > com > oracle > main			
library ▾	Share with ▾	New folder	
Name	Date modified	Type	Size
module.xml	5/29/2018 17:02	XML File	1 KB
ojdbc7.jar	5/17/2018 15:31	WinRAR archive	3,613 KB

- iv. Create module.xml file.
- v. Add the following content:

```
<?xml version="1.0" ?>
<module xmlns="urn:jboss:module:1.1" name="com.oracle">
  <properties>
    <property name="jboss.api" value="unsupported" />
  </properties>
  <resources>
    <resource-root path="ojdbc7.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api" />
    <module name="javax.transaction.api" />
    <module name="javax.servlet.api" optional="true" />
  </dependencies>
</module>
```

### Configure MariaDB drivers

2. Create a new module for MariaDB driver.
  - i. Create a folder hierarchy with path `$JBOSS_HOME/modules/system/layers/base/org/mariadb`
  - ii. Copy mariadb-java-client-2.2.3.jar to `$JBOSS_HOME/modules/system/layers/base/org/mariadb` folder

System (C:) > InstalledPrograms > jboss-eap-7.1.0 > modules > system > layers > base > org > mariadb			
library ▾	Share with ▾	New folder	
Name	Date modified	Type	Size
mariadb-java-client-2.2.3.jar	4/6/2018 11:15	WinRAR archive	554 KB
module.xml	4/16/2018 18:08	XML File	1 KB

- iii. Create module.xml file.

iv. Add the following content:

```
<?xml version="1.0" ?>
<module xmlns="urn:jboss:module:1.1" name="org.mariadb">
  <properties>
    <property name="jboss.api" value="unsupported" />
  </properties>
  <resources>
    <resource-root path="mariadb-java-client-2.2.3.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
    <module name="javax.servlet.api" optional="true" />
  </dependencies>
</module>
```

### **Configure datasources in Jboss configuration file**

Configure data source in `$JBOSS_HOME/standalone/configuration/standalone-full-ha.xml`

Add only one of the below configuration in `<datasources>` tag as per your database type, change the host and port as per database configuration.

Database	Data Source property to enable	Data Source configuration
MariaDB	spring.datasource.jndi-name=java:/YUGMSP_XAMariaDBDS	<pre>&lt;xa-datasource jndi-name="java:/YUGMSP_XAMariaDBDS" pool-name="YUGMSP_XAMariaDBDS" enabled="true" use-ccm="false" stateless="true" enabled="true"&gt;   &lt;xa-datasource-property name="ServerName"&gt;     databasehostname   &lt;/xa-datasource-property&gt;   &lt;xa-datasource-property name="DatabaseName"&gt;     YUG_MSP   &lt;/xa-datasource-property&gt;   &lt;driver&gt;mariadb&lt;/driver&gt;   &lt;xa-pool&gt;     &lt;min-pool-size&gt;10&lt;/min-pool-size&gt;     &lt;initial-pool-size&gt;10&lt;/initial-pool-size&gt;     &lt;max-pool-size&gt;200&lt;/max-pool-size&gt;     &lt;allow-multiple-users&gt;false&lt;/allow-multiple-users&gt;   &lt;/xa-pool&gt;   &lt;security&gt;     &lt;security-domain&gt;YUGMSP_XAMariaDBDS_UserSecurityDomain&lt;/security-domain&gt;   &lt;/security&gt;   &lt;validation&gt;     &lt;valid-connection-checker class-name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLValidConnectionChecker"&gt;       &lt;validate-on-match&gt;true&lt;/validate-on-match&gt;       &lt;background-validation&gt;false&lt;/background-validation&gt;       &lt;exception-sorter class-name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLExceptionSorter"&gt;         &lt;exception-sorter&gt;           &lt;exception-sorter&gt;             &lt;exception-sorter&gt;               &lt;exception-sorter&gt;                 &lt;exception-sorter&gt;                   &lt;exception-sorter&gt;                     &lt;exception-sorter&gt;                       &lt;exception-sorter&gt;                         &lt;exception-sorter&gt;                           &lt;exception-sorter&gt;                         &lt;/exception-sorter&gt;                       &lt;/exception-sorter&gt;                     &lt;/exception-sorter&gt;                   &lt;/exception-sorter&gt;                 &lt;/exception-sorter&gt;               &lt;/exception-sorter&gt;             &lt;/exception-sorter&gt;           &lt;/exception-sorter&gt;         &lt;/exception-sorter&gt;       &lt;/exception-sorter&gt;     &lt;/valid-connection-checker&gt;   &lt;/validation&gt; &lt;/xa-datasource&gt;</pre>

		</validation> </xa-datasource>	
Oracle 11g	spring.datasource.jndi-name=java:/YUGMSP_XAOracle11gDS	<xa-datasource jndi-name="java:/YUGMSP_XAOracle11gDS" pool-name="YUGMSP_XAOracle11gDS" enabled="true" use-ccm="false" st enabled="true"> <xa-datasource-property name="URL"> jdbc:oracle:thin:@<hostname>:<postname>/service </xa-datasource-property> <driver>oracle</driver> <xa-pool> <min-pool-size>10</min-pool-size> <initial-pool-size>10</initial-pool-size> <max-pool-size>200</max-pool-size> <allow-multiple-users>false</allow-multiple-users> <is-same-rm-override>false</is-same-rm-override> <no-tx-separate-pools>true</no-tx-separate-pools> </xa-pool> <security> <security-domain>YUGMSP_XAOracle11gDS_UserSecurityDomain</security-do </security> <validation> <valid-connection-checker class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConn <background-validation>true</background-validatio <stale-connection-checker class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConn <exception-sorter class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleException </validation> </xa-datasource>	
Oracle 12c	spring.datasource.jndi-name=java:/YUGMSP_XAOracle12cDS	<xa-datasource jndi-name="java:/YUGMSP_XAOracle12cDS" pool-name="YUGMSP_XAOracle12cDS" enabled="true" use-ccm="false" st enabled="true"> <xa-datasource-property name="URL"> jdbc:oracle:thin:@ <hostname>:<postname>/service </xa-datasource-property> <driver>oracle</driver> <xa-pool> <min-pool-size>10</min-pool-size> <initial-pool-size>10</initial-pool-size> <max-pool-size>200</max-pool-size> <is-same-rm-override>false</is-same-rm-override> <no-tx-separate-pools>true</no-tx-separate-pools> </xa-pool> <security> <security-domain>YUGMSP_XAOracle12cDS_UserSecurityDomain</security-do </security> <validation> <valid-connection-checker class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConn <background-validation>true</background-validatio <stale-connection-checker class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConn <exception-sorter class- name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleException </validation>	

</xa-datasource>

Add the drivers in drivers section

Database type	Driver
MariaDB	<pre>&lt;driver name="oracle" module="com.oracle"&gt;   &lt;xa-datasource- class&gt;oracle.jdbc.xa.client.OracleXADataSource&lt;/xa-datasource-class&gt; &lt;/driver&gt;</pre>
Oracle11g/12c	<pre>&lt;driver name="mariadb" module="org.mariadb"&gt;   &lt;driver-class&gt;org.mariadb.jdbc.Driver&lt;/driver-class&gt;   &lt;xa-datasource-class&gt;org.mariadb.jdbc.MariaDbDataSource&lt;/xa- datasource-class&gt; &lt;/driver&gt;</pre>

Add one of the security domain as per your database configuration

Database	Security configuration
MariaDB	<pre>&lt;security-domain name="YUGMSP_XAMariaDBDS_UserSecurityDomain" cache- type="default"&gt;   &lt;authentication&gt;     &lt;login-module code="org.picketbox.datasource.security.SecureIdentityLoginModule" flag="required"&gt;       &lt;module-option name="username" value="YUG_MSP"/&gt;       &lt;module-option name="password" value="-46069be03ea6548a"/&gt;       &lt;module-option name="managedConnectionFactoryName" value="jboss.jca:service=LocalTxCM,name=YUGMSP_XAMariaDBDS"/&gt;     &lt;/login-module&gt;   &lt;/authentication&gt; &lt;/security-domain&gt;</pre>
Oracle 11g	<pre>&lt;security-domain name="YUGMSP_XAOracle11gDS_UserSecurityDomain" cache- type="default"&gt;   &lt;authentication&gt;     &lt;login-module code="org.picketbox.datasource.security.SecureIdentityLoginModule" flag="required"&gt;       &lt;module-option name="username" value="YUG_MSP"/&gt;       &lt;module-option name="password" value="-46069be03ea6548a"/&gt;       &lt;module-option name="managedConnectionFactoryName" value="jboss.jca:service=LocalTxCM,name=YUGMSP_XAOracle11gDS"/&gt;     &lt;/login-module&gt;   &lt;/authentication&gt; &lt;/security-domain&gt;</pre>
Oracle 12c	<pre>&lt;security-domain name="YUGMSP_XAOracle12cDS_UserSecurityDomain" cache- type="default"&gt;   &lt;authentication&gt;     &lt;login-module code="org.picketbox.datasource.security.SecureIdentityLoginModule" flag="required"&gt;       &lt;module-option name="username" value="YUG_MSP"/&gt;       &lt;module-option name="password" value="-46069be03ea6548a"/&gt;       &lt;module-option name="managedConnectionFactoryName" value="jboss.jca:service=LocalTxCM,name=YUGMSP_XAOracle12cDS"/&gt;     &lt;/login-module&gt;   &lt;/authentication&gt; &lt;/security-domain&gt;</pre>

```
</login-module>
</authentication>
</security-domain>
```

Encrypt the database password in the security domain is done through picket box. You may use the below command to encrypt the password.

```
java -cp
$JBOSS_HOME\modules\system\layers\base\org\picketbox\main\picketbox-
5.0.2.Final-redhat-1.jar
org.picketbox.datasource.security.SecureIdentityLoginModule <password String to
encrypt>
```

Change the jar name and version as per the jar available in your jboss modules.

### 3. Disable default JPA in jboss 7.1 by removing below entry altogether

```
<subsystem xmlns="urn:jboss:domain:jpa:1.1">
  <jpa default-datasource="" default-extended-persistence-inheritance="DEEP"/>
</subsystem>
```

### 4. Create ActiveMQ server and JMS Queues

Add the below entries in Active MQ subsystem inside below tag

```
<subsystem xmlns="urn:jboss:domain:messaging-activemq:2.0">
```

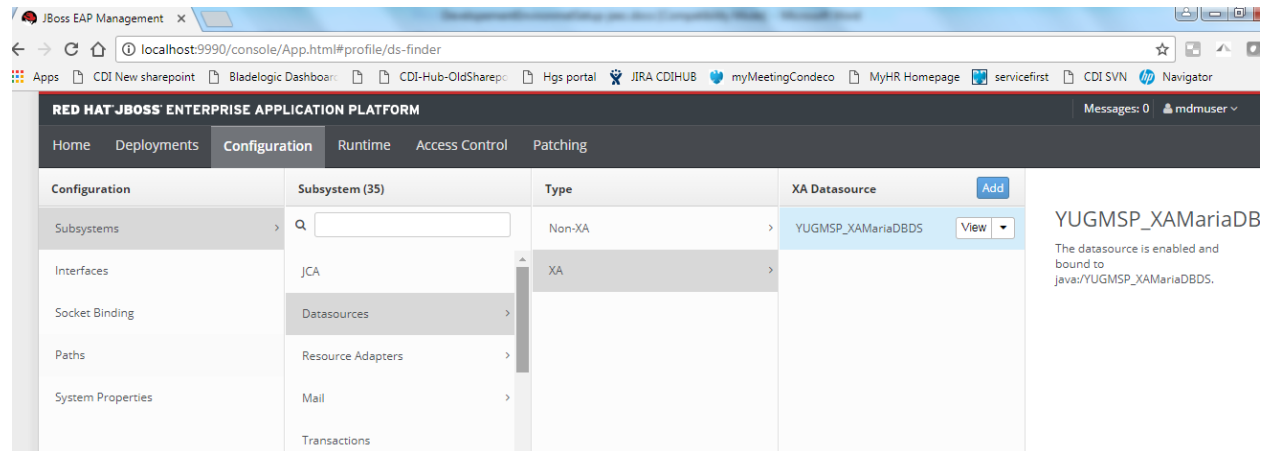
```
<server name="Yug">
  <security enabled="false"/>
  <management address="jms.queue.activemq.management1"/>
  <statistics enabled="true"/>
  <security-setting name="#">
    <role name="guest" send="true" consume="true" manage="true"/>
  </security-setting>
  <address-setting name="#" dead-letter-address="jms.queue.DLQ" expiry-
address="jms.queue.ExpiryQueue"/>
  <remote-connector name="yugConnectorRemote" socket-binding="http"/>
  <in-vm-connector name="yugConnectorInvm" server-id="0"/>
  <jms-queue name="YUG.DEFAULT.RESPONSE"
entries="java:jboss/com/yugandhar/default/responseQueue"/>
  <jms-queue name="YUG.DEFAULT.REQUEST"
entries="java:jboss/com/yugandhar/default/requestQueue"/>
  <pooled-connection-factory name="YugandharDefaultPooledConnectionFactory"
entries="java:jboss/com/yugandhar/DefaultPooledConnectionFactory"
connectors="yugConnectorInvm" statistics-enabled="true"/>
</server>
```



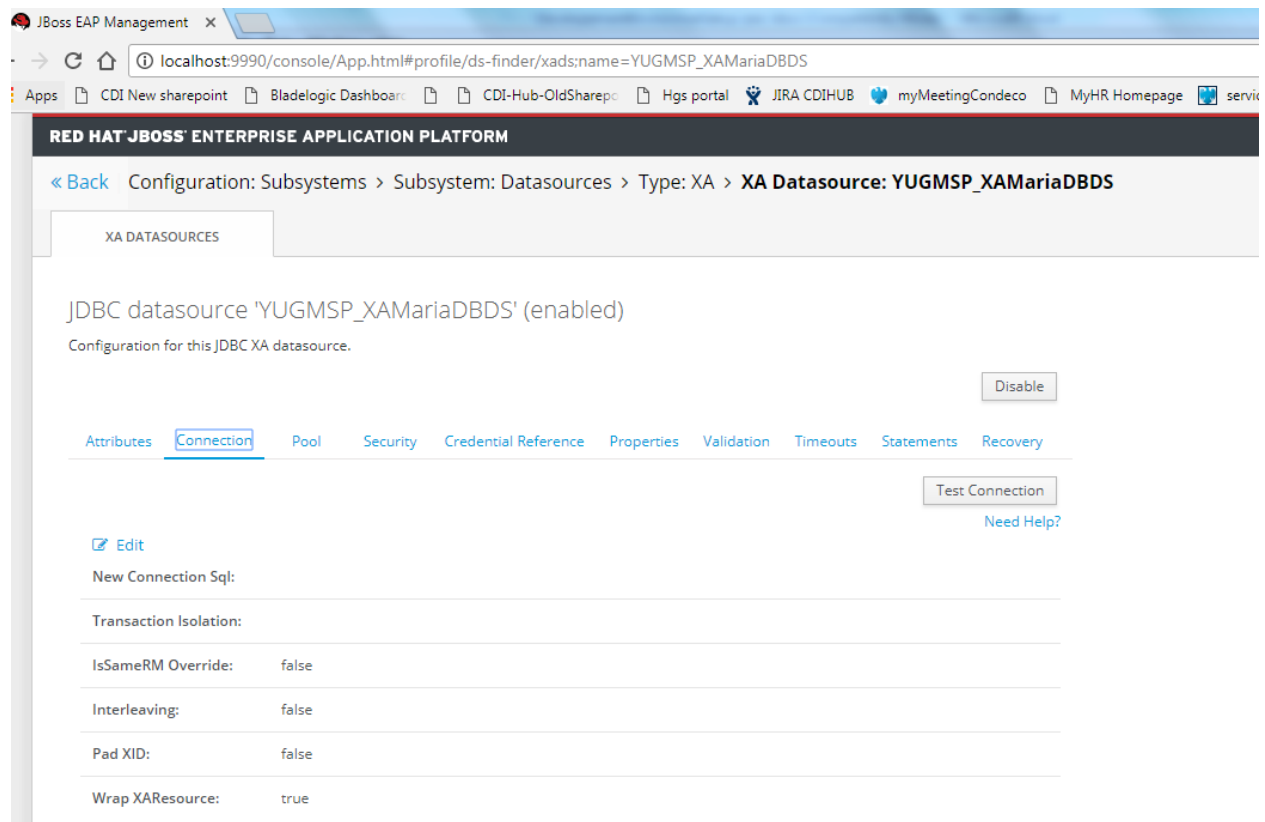
## Verify the Datasource

Login to jboss console using the user created in 'add User to Access JBoss Console' step.

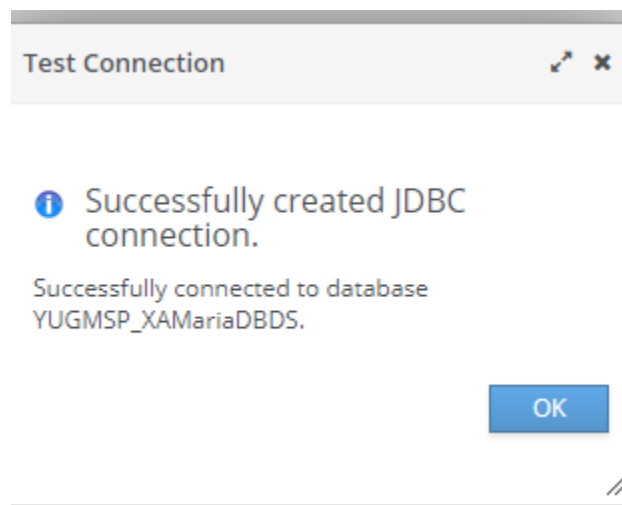
Navigate to datasources



Click View and then navigate to Connection tab.

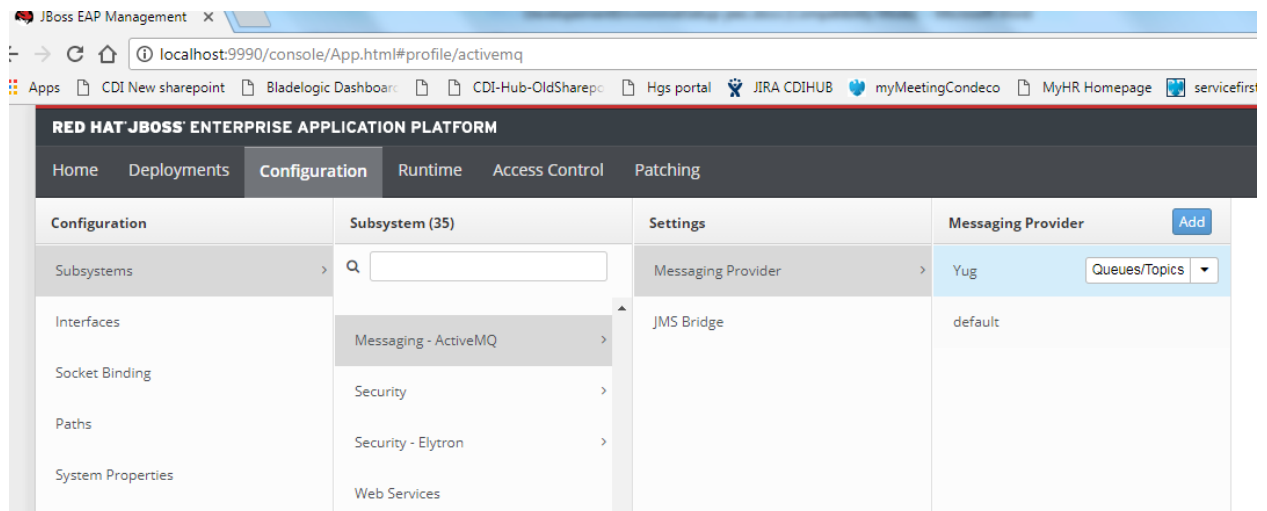


Click on the test connection and verify that the connection is successful

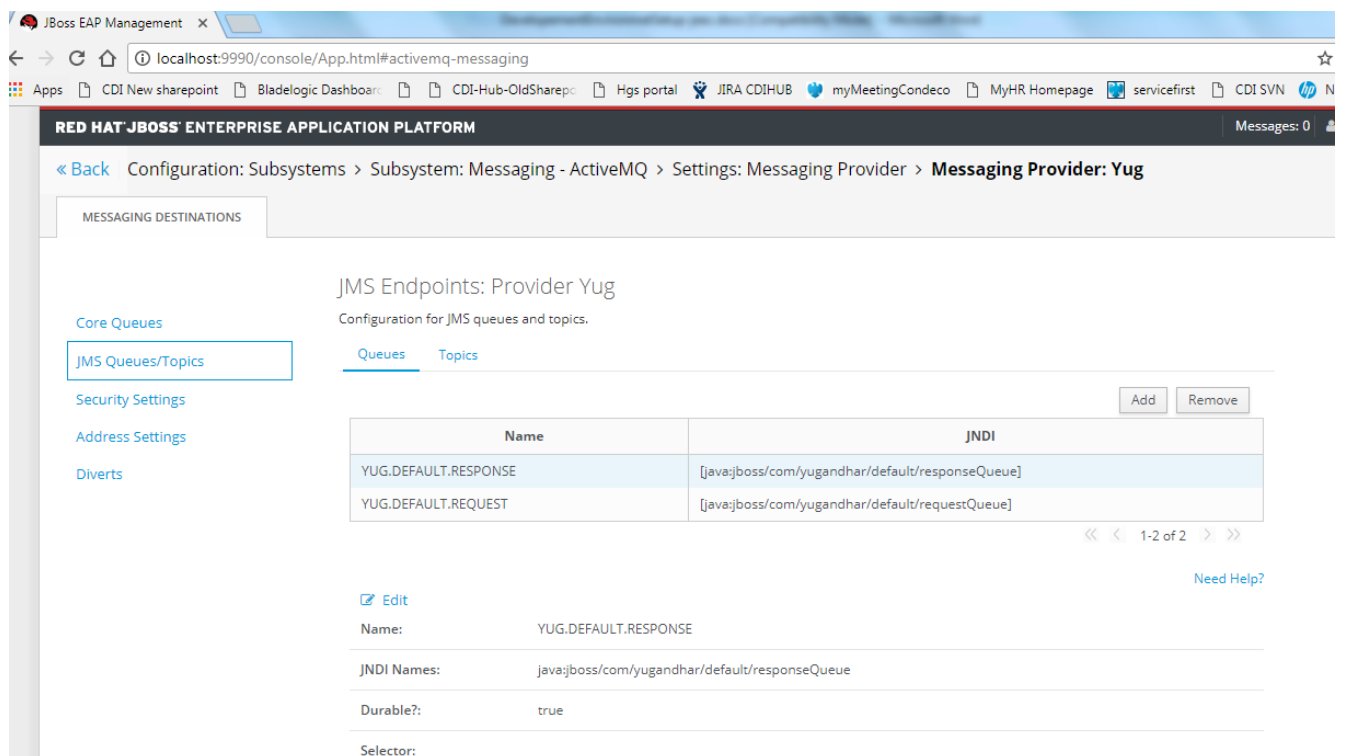


### **Verify ActiveMQ**

Navigate to ActiveMQ subsystem → Yug, click Queues/Topics



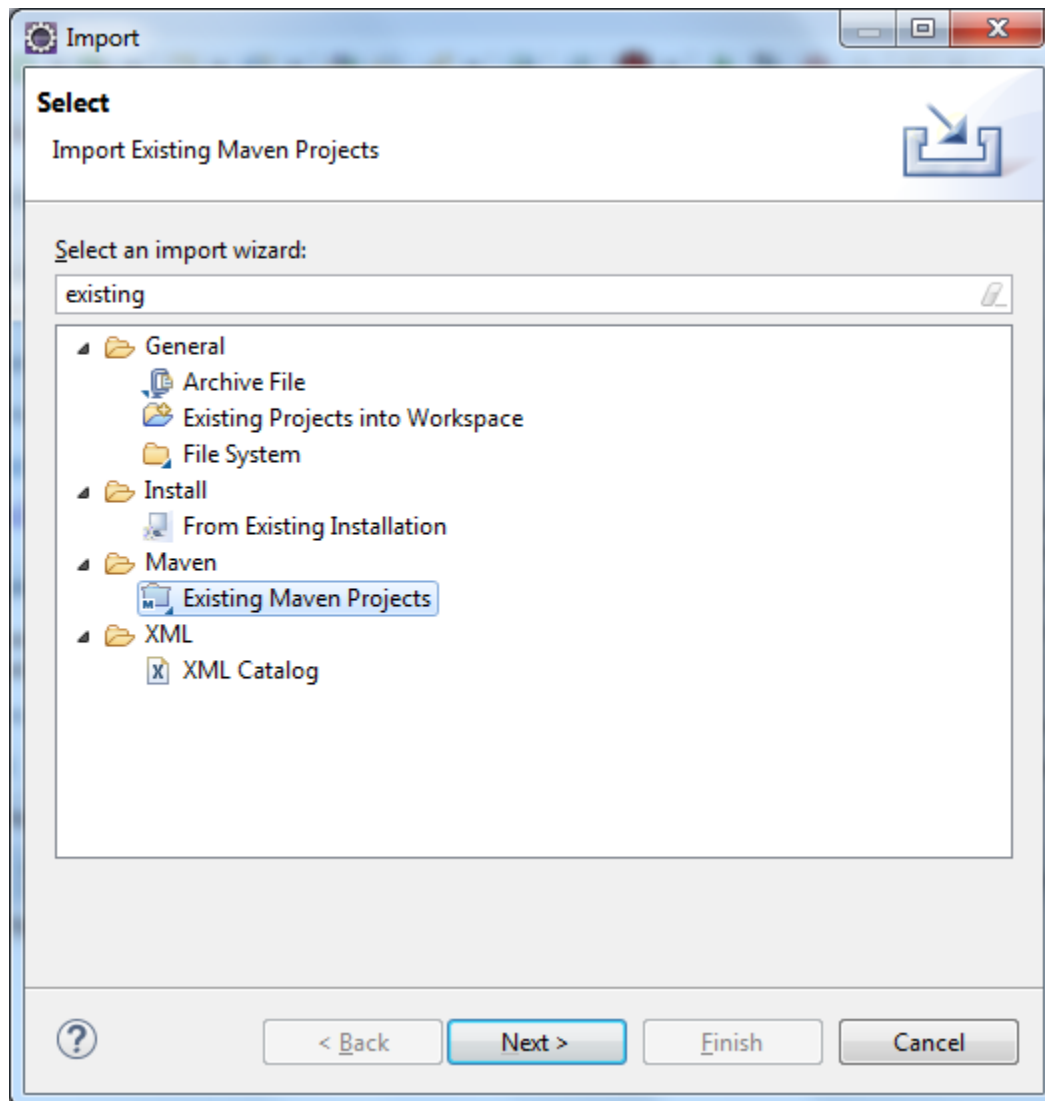
You should see the Active MQs as shown in below image



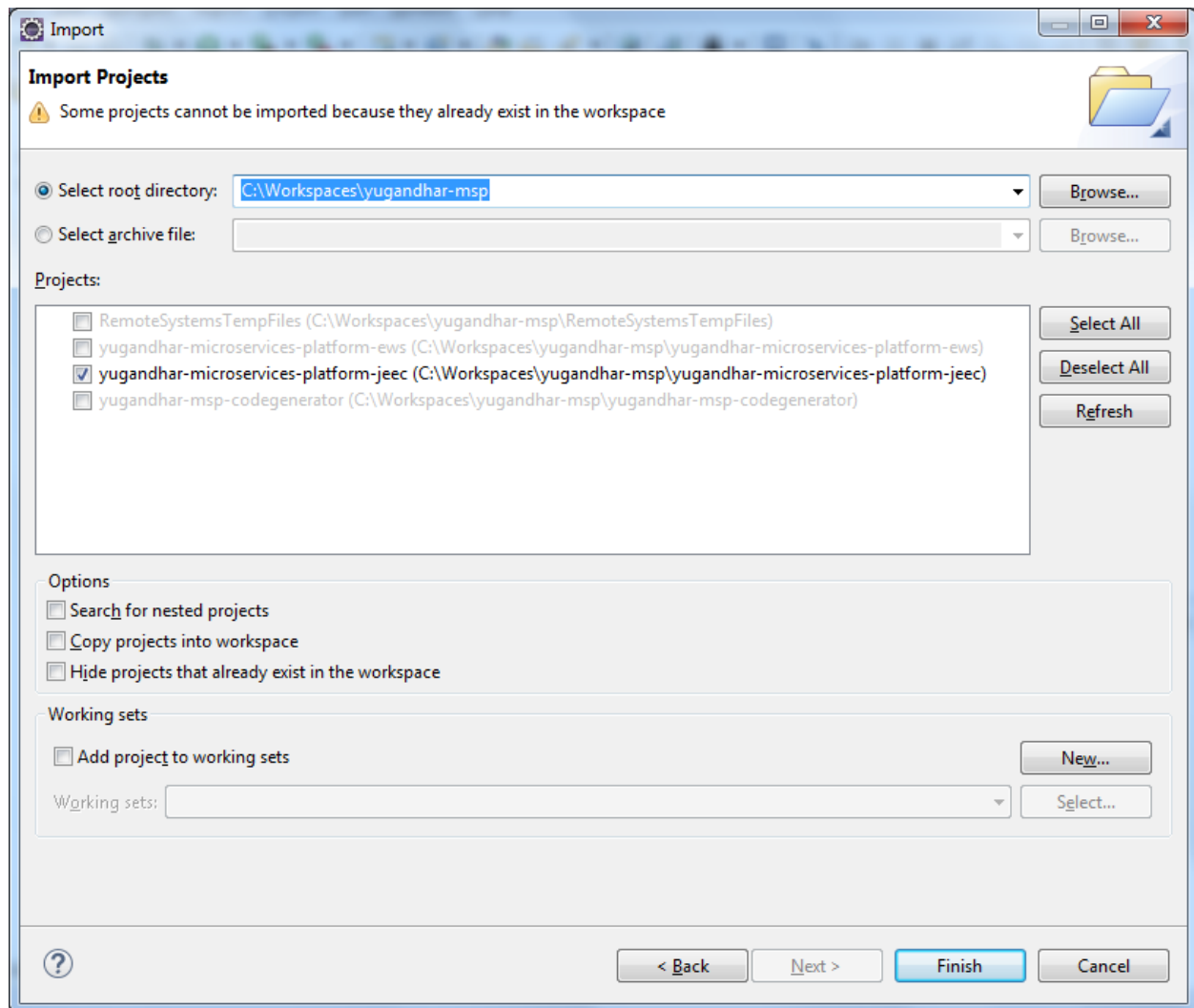
## Import Yugandhar msp java projects in the workspace

Download the yugandhar-Microservice-platform-jeec from the github repository, extract it in the workspace directory (e.g. C:\Workspaces\yugandhar-msp) and import in Workspace

Go to File → Import Menu → Existing Maven Projects →



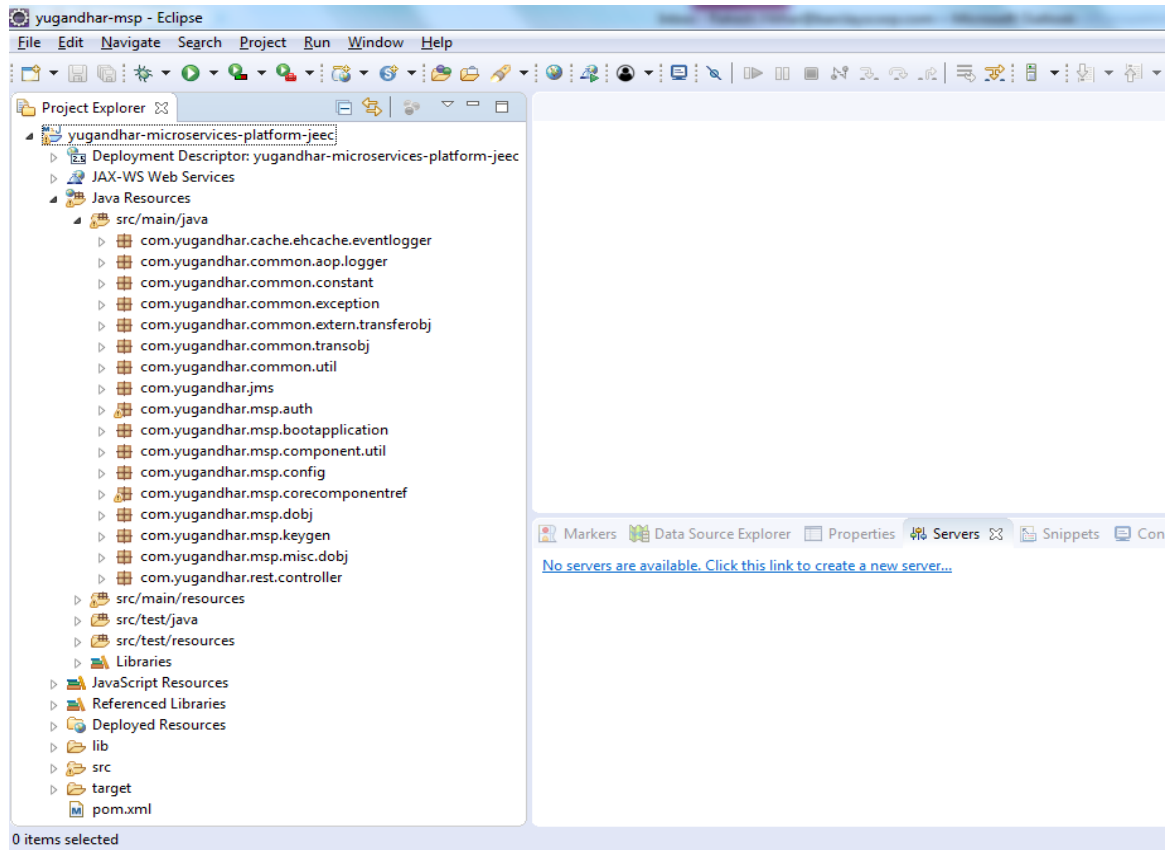
Click on Next Button



Select the yugandhar-Microservice-platform-jeeec project and click Finish.

It may take some time as eclipse will download the maven jars automatically and build the project. You may track the progress in 'Progress' tab.

Make sure that your workspace is error free



### ***Make datasource jndi name changes***

Set the spring.datasource.jndi-name property in application.properties as per your database. For sample here the mariaDB jndi is enabled.

```
#JNDI for the data sources
#spring.datasource.jndi-name=java:/YUGMSP_XAOracle11gDS
#spring.datasource.jndi-name=java:/YUGMSP_XAOracle12cDS
spring.datasource.jndi-name=java:/YUGMSP_XAMariaDBDS
```

## Make Logging changes

Microservice platform does the logging to default folder C:/Yugandhar/logs so create this folder or change the log directory to the directory of your choice in /yugandhar-Microservice-platform-jeec/src/main/resources/yugandhar\_logback.xml

```
<appender name="TIME_AND_SIZE_BASED_APPENDER" class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>C:/yugandhar/logs/YugandharCommon.log</file>
  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <fileNamePattern>C:/yugandhar/logs/YugandharCommon.%d{yyyy-MM-dd}.%i.log</fileNamePattern>
    <timeBasedFileNamingAndTriggeringPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
```

## Properties file changes

There are below two properties files in the Microservice platform

### application.properties

The application properties file covers the below properties

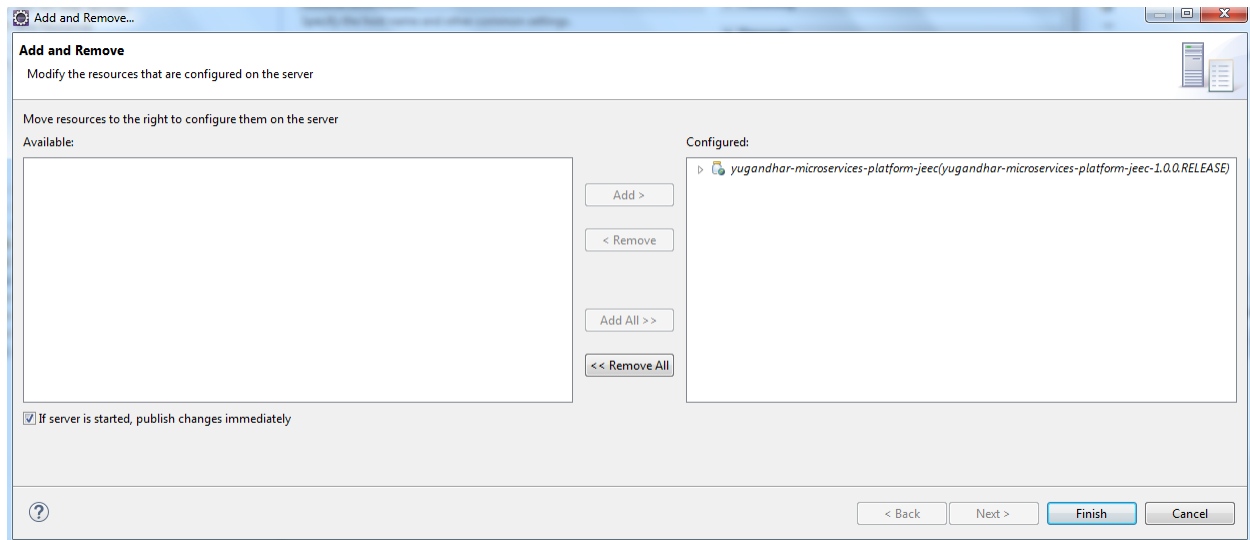
- **Springboot trace:** enable/disable the trace
- **Server port:** set the port number for the tomcat server
- **JPA:** If the generated ddl needs to be logged the enable the property spring.jpa.show-sql. By default this is enabled.
- **mariaDB specific settings:**  
#Enable both the below properties for mysql/MariaDB, if you are using oracle then comment both the properties.  
#spring.jpa.properties.hibernate.globally\_quoted\_identifiers=true  
#spring.jpa.database-platform=org.hibernate.dialect.MariaDB53Dialect  
Note- "spring.jpa.properties.hibernate.globally\_quoted\_identifiers= true" with Oracle database may result in errors.
- **Oracle specific settings:**  
Oracle Specific configuration, use 10g dialect for Oracle 11g database else use 12c  
#spring.jpa.database-platform=org.hibernate.dialect.Oracle10gDialect  
#spring.jpa.database-platform=org.hibernate.dialect.Oracle12cDialect
- **Logging:** Logback configuration
- **JTA:** Narayana JTA of JBoss is used
- **Ehcache:** ehcache properties
- **Json:** json parser related properties
- **Active mq:** By default the JBoss embedded Active MQ is used, modify the properties as needed.
- **Actuator:** spring boot actuator properties
- **Eureka integration:** Eureka integration properties, by default it's disabled.

## yugandhar-msp.properties

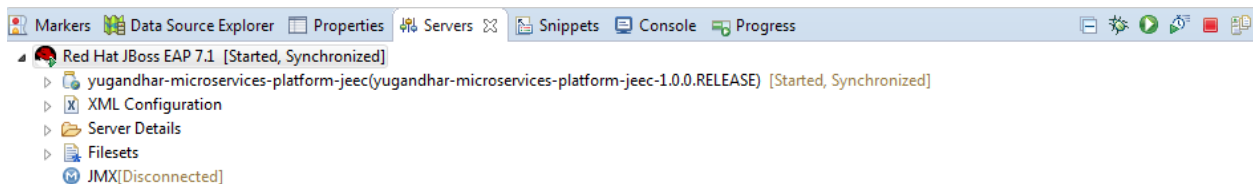
The yugandhar-msp.properties file is custom properties file not used by JEEC version.

## Running the application

Right click the jboss server in the Servers tab and add the MSP application, click finish.



Check the logs lines below to verify the application is started.





## TEST With SOAPUI

### Test JSON message

The rest url is as below.

<http://localhost:8091//yugandhar-Microservice-platform-jeec-1.0.0.RELEASE/rest/YugandharRequestProcessor>

Yugandhar MSP uses the port 8091 as default; you may change it through application.properties file.

### Sample json message

```
{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDO": {
      "configLanguageCodeKey": "1"
    }
  }
}
```

For the soap xml message, add the below headers for the request xmls

Header	Value	use
Accept	application/json	This header tells yugandhar rest controller that the response must be sent in json format.
Content-Type	application/json	This header tells yugandhar rest controller that the request message is of type json

Create 'New REST service from URI' in the soapui project, and execute it with attached json message

**Request 1**

Method: POST Endpoint: http://localhost:8091 Resource: /yugandhar-microservices-platform-jeeec-1.0.0.RELEASE/rest/YugandharRequestProcessor

Name	Value	Style	Level
Required: <input type="checkbox"/> Sets if parameter is required			

Media Type: application/ison ☐ Post QueryString

```
{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDO": {}
  }
}
```

Header	Value
Accept	application/json
Content-Type	application/json

Auth: Headers (2) Attachments (0) Representations (1) JMS Headers JMS Properties (0)

Check the response as SUCCESS

**Request 1**

Method: POST Endpoint: http://localhost:8091 Resource: /yugandhar-microservices-platform-jeeec-1.0.0.RELEASE/rest/YugandharRequestProcessor

Name	Value	Style	Level
Required: <input type="checkbox"/> Sets if parameter is required			

Media Type: application/ison ☐ Post QueryString

```
{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDOList": [
      {
        "idPk": "144",
        "version": 0,
        "createdTs": "2018-04-16T12.57.17.000+0000",
        "updatedTs": "2018-04-16T12.57.17.000+0000",
        "updatedByUser": "admin",
        "updatedByTxnId": "000000",
        "configLanguageCodeKey": "1",
        "txn": "5000"
      }
    ]
  }
}
```

Header	Value
Accept	application/json
Content-Type	application/json

Headers (7) Attachments (0) SSL Info Representations (14) Schema JMS (0)

## Test XML message

The rest url is as below.

<http://localhost:8091/rest/YugandharRequestProcessor>

Yugandhar MSP uses the port 8091 as default; you may change it through application.properties file.

For the soap xml message, add the below headers for the request xmls

Header	Value	use
Accept	application/xml	This header tells yugandhar rest controller that the response must be sent in xml format.
Content-Type	application/xml	This header tells yugandhar rest controller that the request message is of type xml

Sample XML message:

```
<TxnTransferObj>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>
  </txnHeader>
  <transactionServiceName>findAllRefCountryIsoByLanguageCodeBase</transactionServiceName>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
    <paginationPageSize>50</paginationPageSize>
    <refCountryIsoDO>
      <configLanguageCodeKey>1</configLanguageCodeKey>
    </refCountryIsoDO>
  </txnPayload>
</TxnTransferObj>
```

**Request 1**

Method: POST Endpoint: http://localhost:8091 Resource: /yugandhar-microservices-platform-jeec-1.0.0.RELEASE/rest/YugandharRequestProcessor

Name	Value	Style	Level
Required: <input type="checkbox"/> Sets if parameter is required			
Media Type	application/xml	<input type="checkbox"/> Post QueryString	

```
<transactionServiceName>findAllRefCountryIsoByLanguageCodeBase</transactionServiceName>
</txnHeader>
<txnPayload>
  <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
  <paginationPageSize>50</paginationPageSize>
  <refCountryIsoDO>
    <configLanguageCodeKey>1</configLanguageCodeKey>
  </refCountryIsoDO>
</txnPayload>
</TxnTransferObj>
```

Header	Value
Content-Type	application/xml
Accept	application/xml

Auth Headers (2) Attachments (0) Representations (1) JMS Headers JMS Property (0)

Check the success response.

The screenshot displays the REST Client interface for a request labeled "Request 1". The Method is POST, the Endpoint is http://localhost:8091, and the Resource is /yugandhar-microservices-platform-jeeec-1.0.0.RELEASE/rest/YugandharRequestProcessor. The Media Type is set to application/xml. The request body is an XML document with the following structure:

```
<TxnTransferObj>
  <transactionServiceName>findAllRefCountryIs
</txnHeader>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
    <paginationPageSize>50</paginationPageSize>
    <refCountryIsoDO>
      <configLanguageCodeKey>1</configLanguageCodeKey>
    </refCountryIsoDO>
  </txnPayload>
</TxnTransferObj>
```

The response body is an XML document with the following structure:

```
<TxnTransferObj>
  <responseCode>SUCCESS</responseCode>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>
    <transactionServiceName>findAllRefCountryIsoByLanguageCodeBase</transactionServiceName>
    <totalExecutionTimeMillies>118</totalExecutionTimeMillies>
  </txnHeader>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
    <paginationPageSize>50</paginationPageSize>
    <paginationInfoElementsOnCurrentSlice>50</paginationInfoElementsOnCurrentSlice>
    <paginationInfoTotalElements>247</paginationInfoTotalElements>
    <paginationInfoTotalPages>5</paginationInfoTotalPages>
    <refCountryIsoDOList>
      <refCountryIsoDOList>
        <idPk>144</idPk>
        <version>0</version>
        <createdTs>2018-04-16T12.57.17.000+0000</createdTs>
        <updatedTs>2018-04-16T12.57.17.000+0000</updatedTs>
        <updatedByUser>admin</updatedByUser>
        <updatedByTxnId>000000</updatedByTxnId>
        <configLanguageCodeKey>1</configLanguageCodeKey>
        <key>500</key>
        <value>MSR</value>
      </refCountryIsoDOList>
    </refCountryIsoDOList>
  </txnPayload>
</TxnTransferObj>
```

The interface also shows a table for headers:

Header	Value
Accept	application/xml
Content-Type	application/xml

At the bottom, there are tabs for Headers (7), Attachments (0), SSL Info, Representations (14), Schema (conflicts), and JMS (0).

This certifies your workspace. Go ahead with Development and customization guide, API Transaction reference guide as well as Code generation guide to understand more.