

CBSA Liberty UI application deployment

Assumptions:

- This document utilises WAZI for deployment, and therefore the screenshots are WAZI based.
- At the time of writing these instructions, we were utilising CICS TS 5.5 and therefore all directory names etc used are based around that, they will need to be amended accordingly for difference versions of CICS.
- Unless you are utilising this in a supplied (zTrial) environment, chances are that the hostname, port number , userid and CICS TS version may be different in your own environment. We have highlighted, within these instructions, where such things may need to be amended accordingly.

Updating the CICSUSER Userid:

It may be necessary to update the USERID being utilised in your environment to allow the USERID running the CICS region to have access to everything it needs. In this example the USER running the CICS region is called CICSUSER. It needs to have access to the “HOME” directory.

1. Issue the commands from the TSO Shell in WAZI Developer for Eclipse.

LISTUSER CICSUSER OMVS NORACF shows:

```
USER=CICSUSER

OMVS INFORMATION
-----
UID= 0000990018
HOME= /u
PROGRAM= /bin/sh
CPUTIMEMAX= NONE
ASSIZEMAX= NONE
FILEPROCMA= NONE
PROCUSERMA= NONE
THREADSMA= NONE
MMAPAREAMA= NONE
```

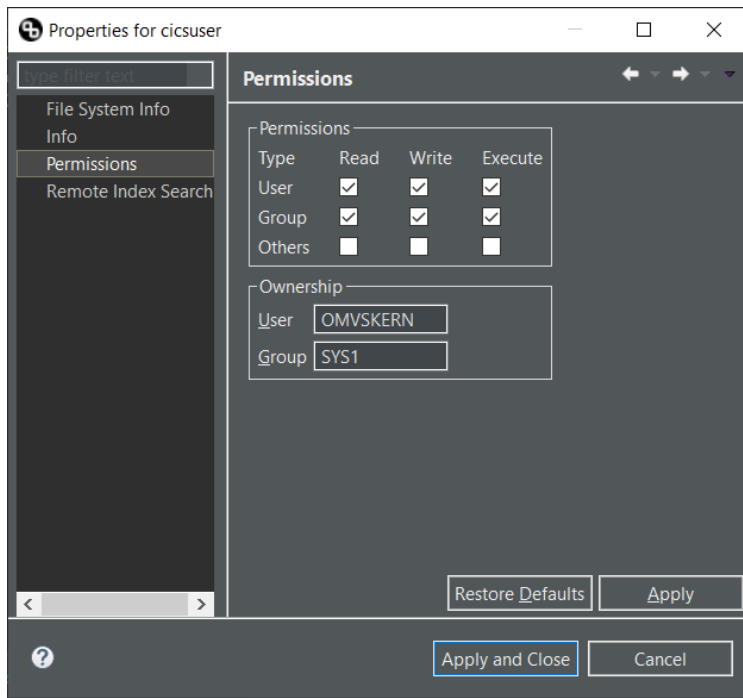
2. The “HOME” directory is set to /u and the USERID running the CICS region (CICSUSER) may not have access to that. You fix that by:

```
ALTUSER CICSUSER OMVS(HOME(/u/cicsuser))
```

This fixes it, but that directory doesn’t exist yet.

3. Create a new folder for /u/cicsuser in Remote Systems Explorer.

This may cause permissions problems. So amend the permissions as follows:



And then change the owner:

```
chown CICSUSER .
```

Create a JVM server:

We need a JVMSERVER resource.

1. Create a new directory called

```
/var/cics/JVMProfiles/
```

2. Copy the CICS supplied JVM profile called DFHWLP in to this new directory. In this case

```
/usr/lpp/cicsts/cicsts55/JVMProfiles/DFHWLP.jvmprofile
```

into this new directory.

3. To have a JVMSERVER resource, we need a JVMPROFILEDIR. Specify the following in the CICS region SIT parameters

```
DFH550.SYSIN(DFH$SIP1)
JVMPROFILEDIR=/var/cics/JVMProfiles/
```

4. DFHWLP.jvmprofile should already exist. It needs to be edited.

5. Ensure that JAVA_HOME is set to

```
JAVA_HOME=/usr/lpp/java/J8.0_64/
```

6. Ensure that autoconfigure is set to true

```
-Dcom.ibm.cics.jvmserver.wlp.autoconfigure=true
```

7. You need to set the timeout to a large value. Please note that if the execution machine is not very powerful that Java may take a significant time to start.

```
-Dcom.ibm.cics.jvmserver.controller.timeout=900000
```

8. Add the following to the JVM profile to add support for DB2

```
-Dcom.ibm.cics.jvmserver.wlp.jdbc.driver.location=/usr/lpp/db2c10/jdbc/
```

```
-Dcom.ibm.cics.jvmserver.wlp.server.http.port=19080
```

(Although port 19080 is used in this example, you may need to use a different port number in your own environment, the above needs to reflect your choice of port number.

9. WORK_DIR must be set to a directory that the CICS region userid "CICSUSER" has access to. For example:

```
/u/cicsuser/
```

CICS Resources:

1. CSD GROUP CBSAWLP contains a JVMSERVER called CBSAWLP

```
CEDA DEFINE GROUP(CBSAWLP) JVMSERVER(CBSAWLP) JVMPROFILE(DFHWLP)
```

This is JVMSERVER DFHWLP from DFHWLP copied and renamed.

```
CEDA DEFINE GROUP(CBSAWLP) URIMAP(CBSAWLP) USAGE(JVMSERVER) HOST(*)  
PORT(19080) PATH(*) USERID(CICSUSER)
```

2. Add the group CBSAWLP to a list installed on a cold start, for example CICSTS55.

3. Add the group DFH\$WU to a list installed on a cold start.

4. Restart the CICS region with a cold or initial start.

5. Create and install this and it will create a JVM server. It may take a few minutes to become enabled.

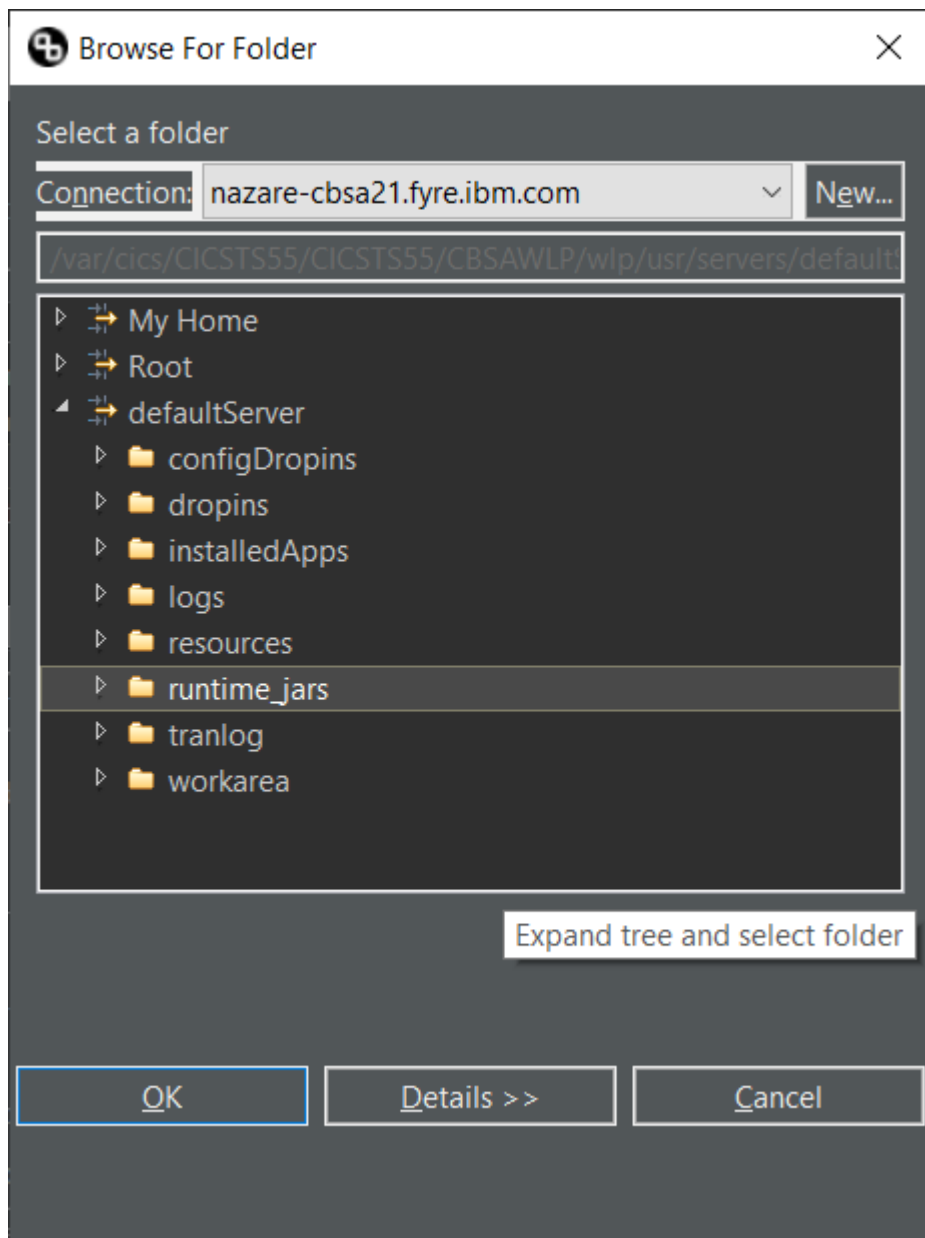
This will create a server.xml file in

```
/u/cicsuser/CICSTS55/CICSTS55/CBSAWLP/wlp/usr/servers/defaultServer/
```

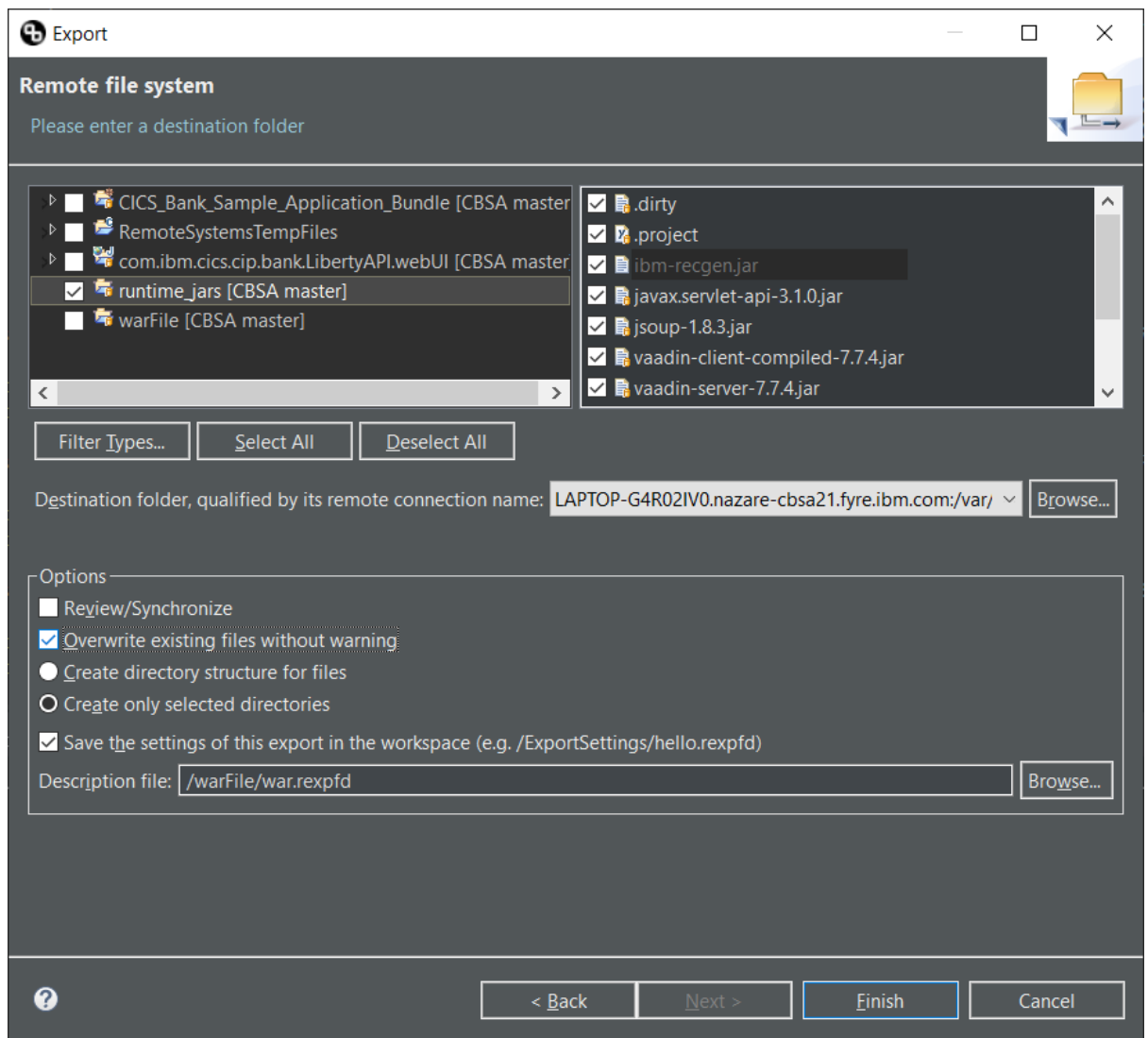
Create a runtime_jars directory:

1. In UNIX Systems Services create a directory called "runtime_jars" in the Liberty workspace.

```
/u/cicsuser/CICSTS55/CICSTS55/CBSAWLP/wlp/usr/servers/defaultServer/runtime_jars
```



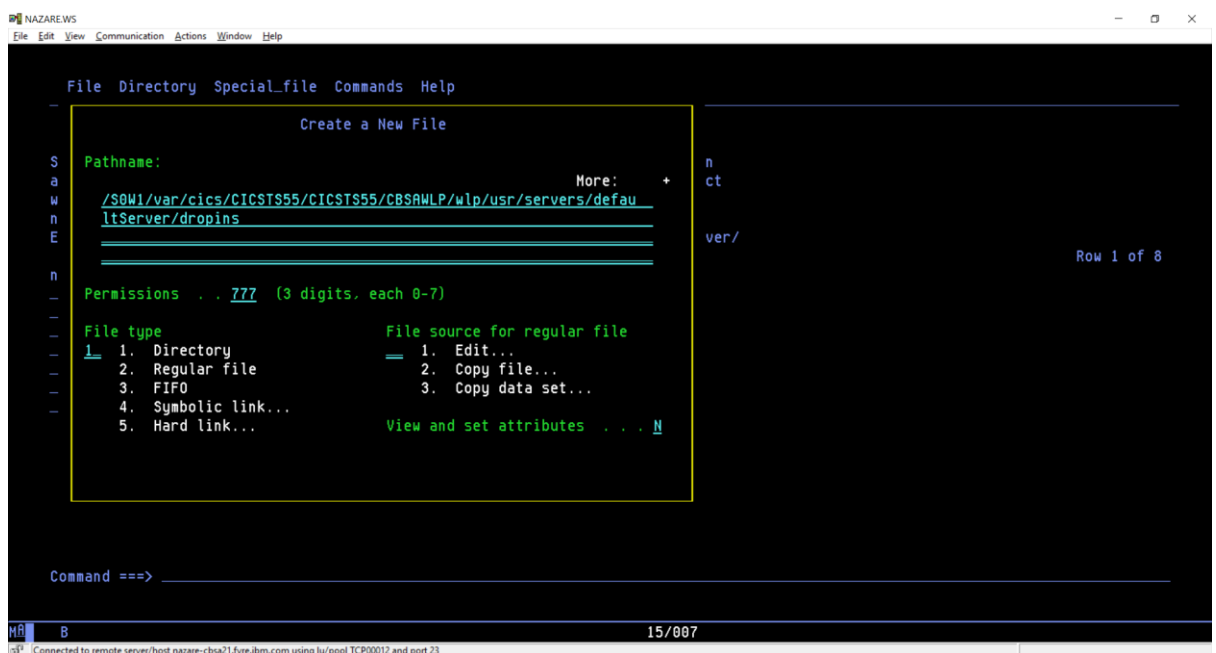
2. Right click on the jars in “runtime_jars” in WAZI and select export to remote file system.



Create a “dropins” directory:

1. In UNIX Systems Services create a directory called “dropins” in the Liberty workspace.

/u/cicsuser/CICSTS55/CICSTS55/CBSAWLP/wlp/usr/servers/defaultServer/defaultServer/dropins



Edit server.xml:

1. Edit server.xml so that the attribute dropinsEnabled is set to true.

```
<applicationMonitor dropins="dropins" dropinsEnabled="true"
pollingRate="5s" updateTrigger="disabled"/>
```

2. Add the following attribute to the **properties.db2.jcc** element:

```
currentSchema="IBMUSER"
```

So it now looks like this

```
<properties.db2.jcc driverType="2" currentSchema="IBMUSER"/>
```

Note – “IBMUSER” is utilised in this example and relates to the user id assigned to the Db2 environment setup. This will need to reflect your chosen user id if different.

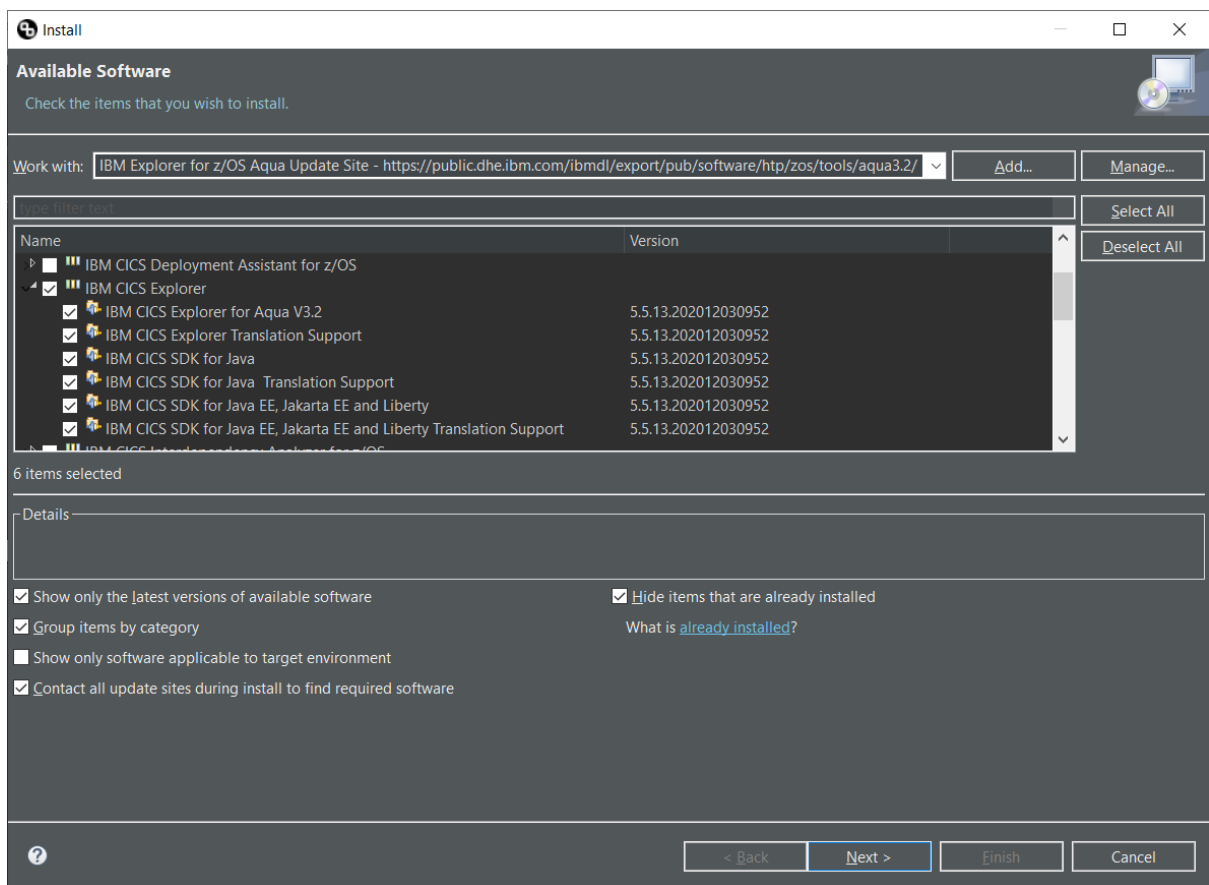
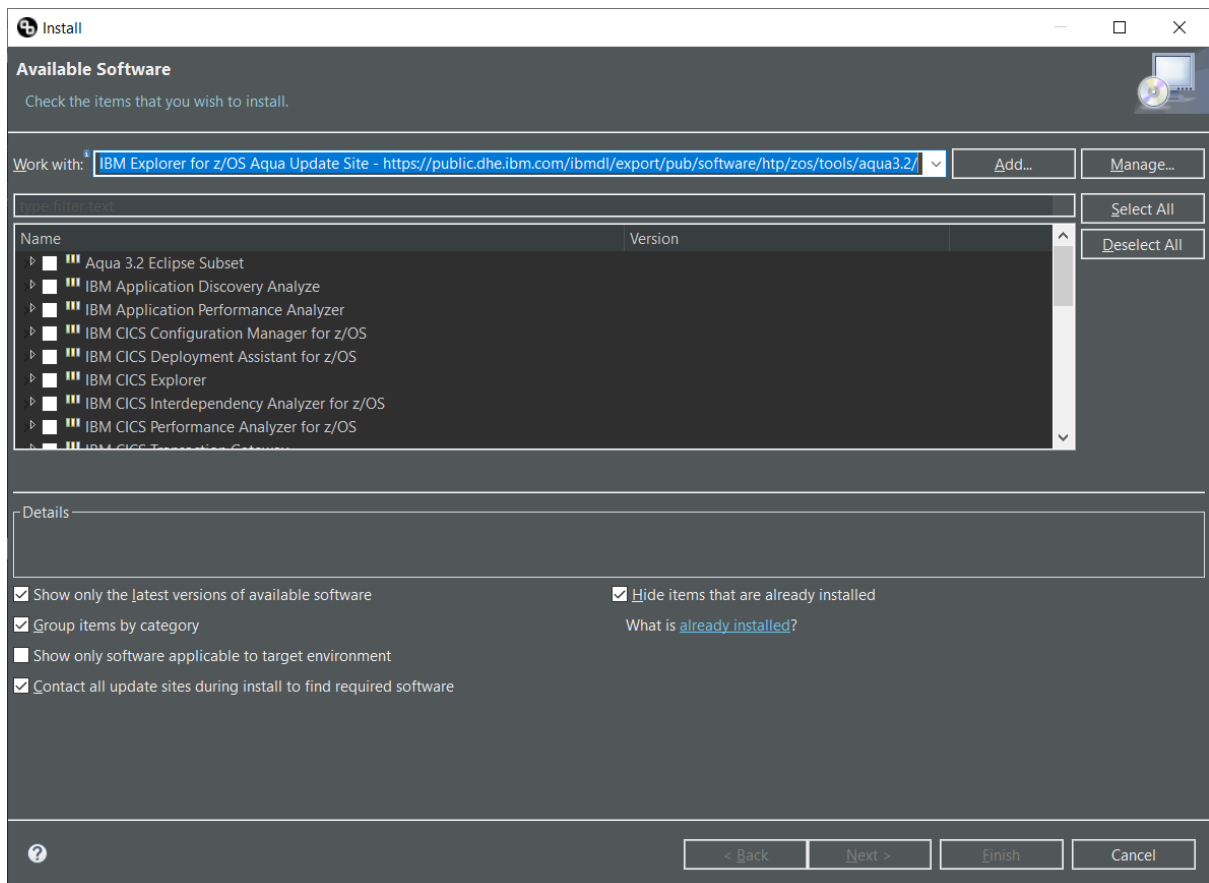
3. Add the following:

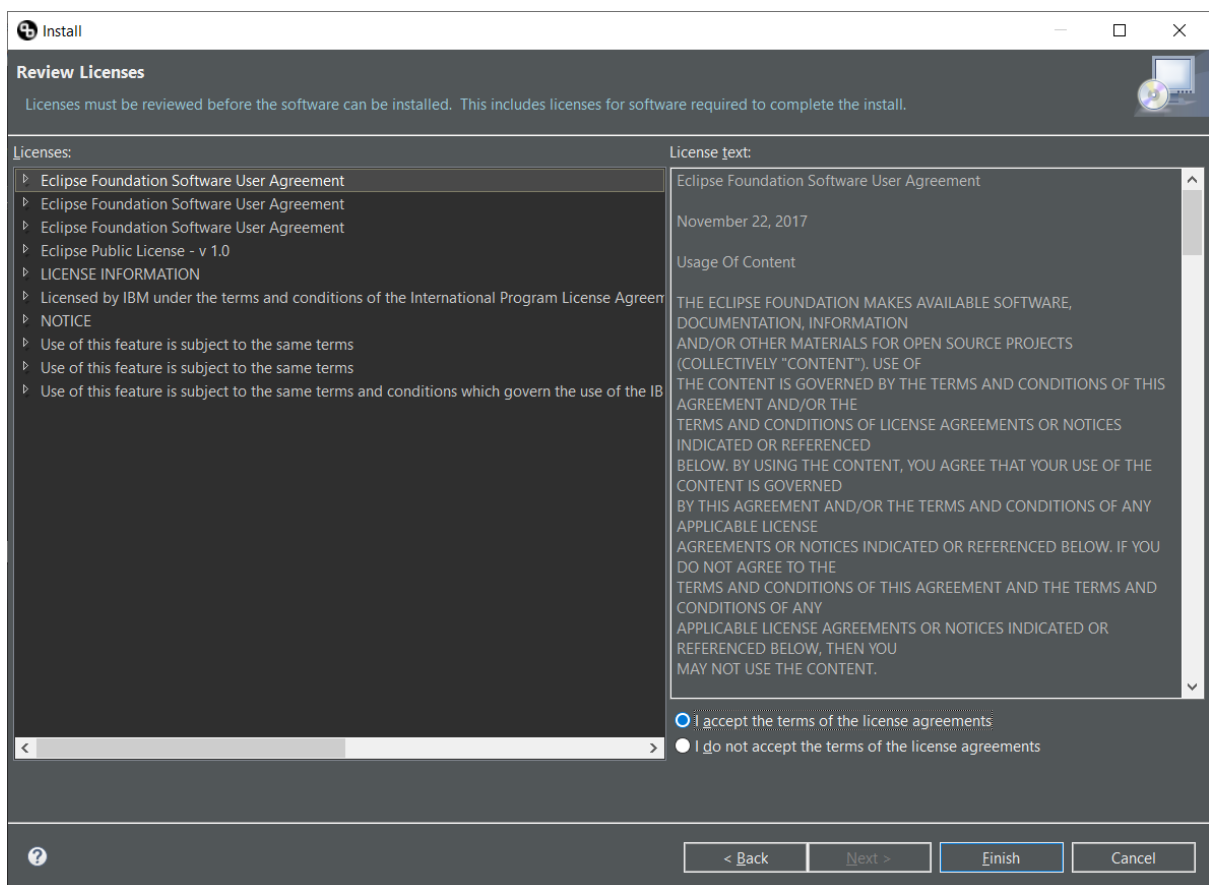
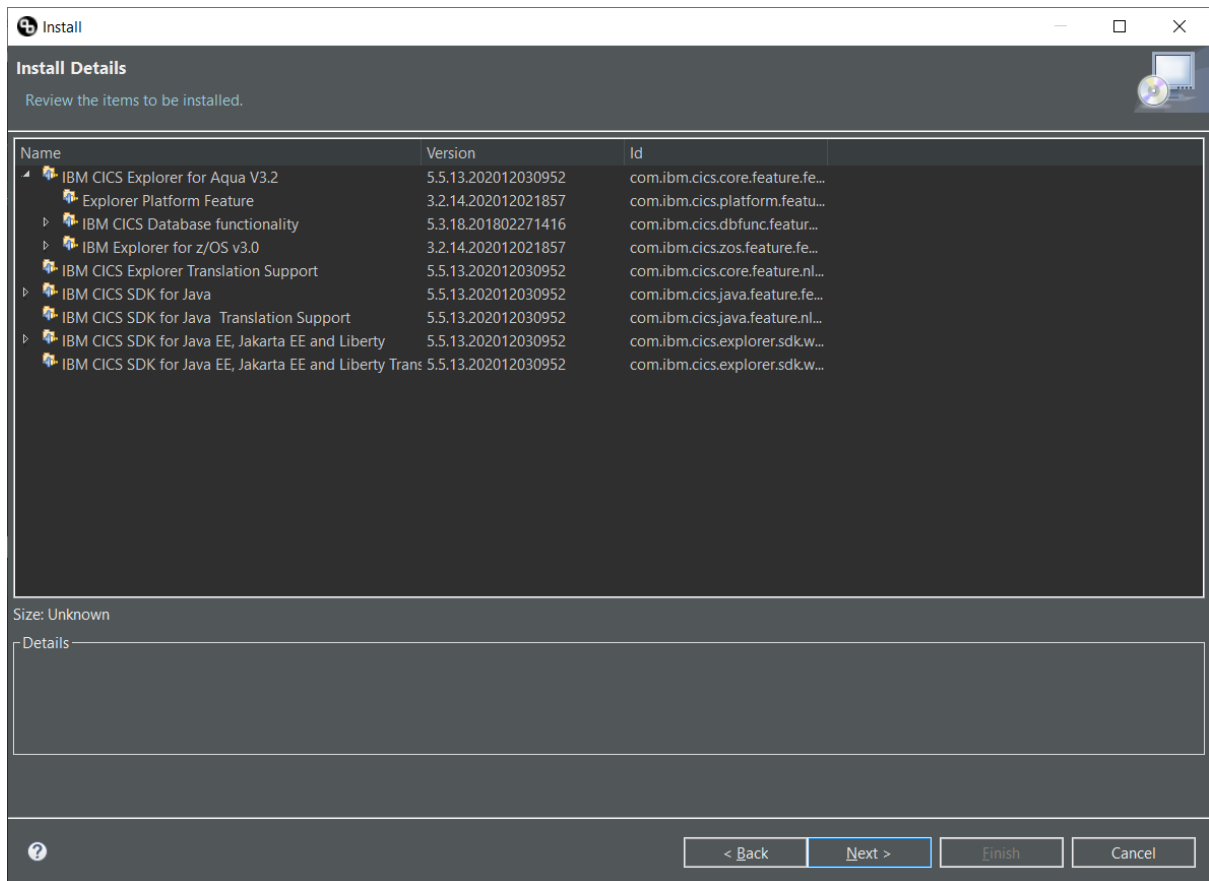
```
<library id="global">
  <fileset dir="/u/cicsuser/CICSTS55/CICSTS55/CBSAWLP/wlp/usr/servers/defaultServer/
runtime_jars" include="*.jar"/>
</library>
```

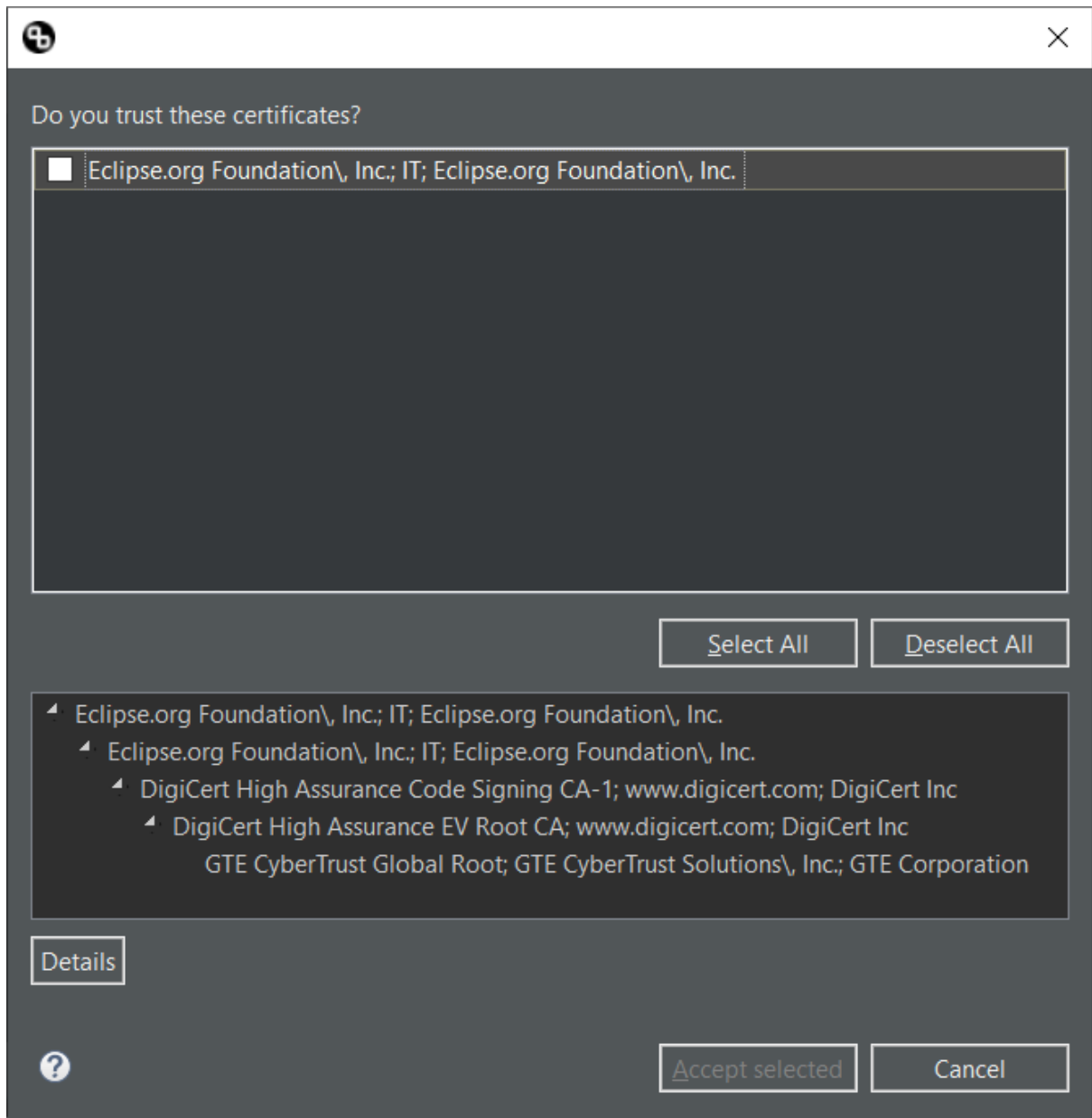
4. Within the <featureManager> tag, add the following lines:

```
<feature>jaxrs-1.1</feature>
<feature>json-1.0</feature>
```

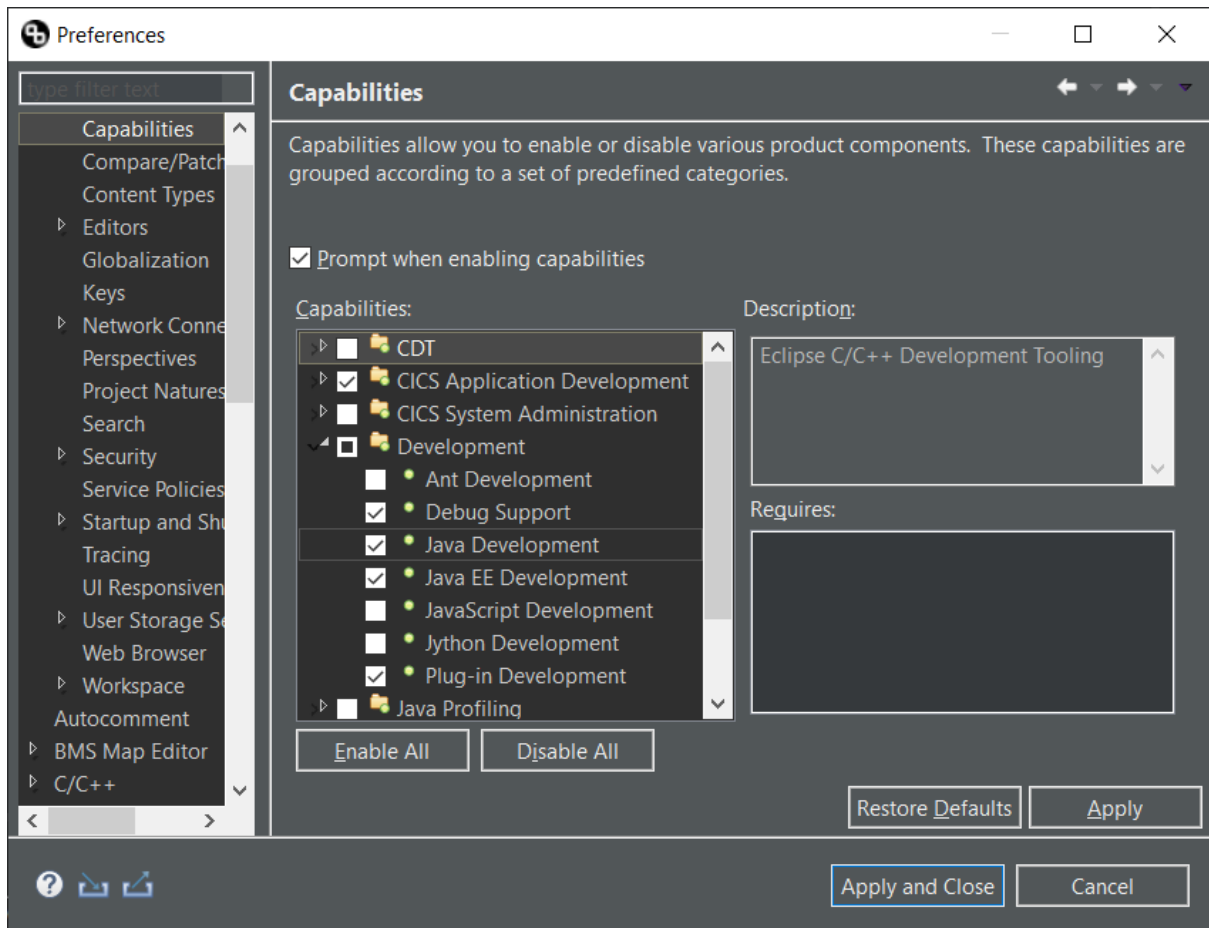
Updating WAZI Developer for Eclipse:







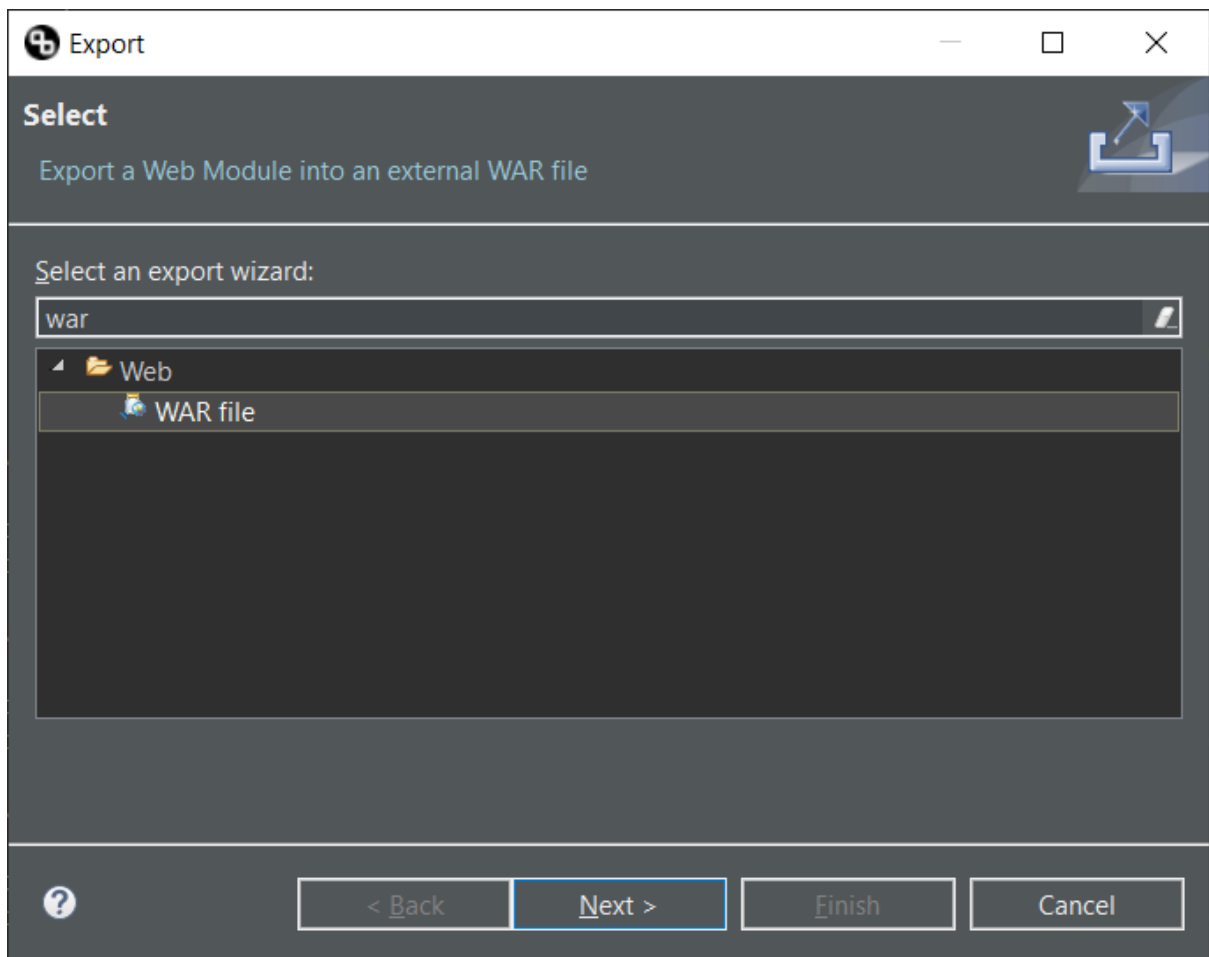
1. Update Wazi capabilities to add Java EE Development



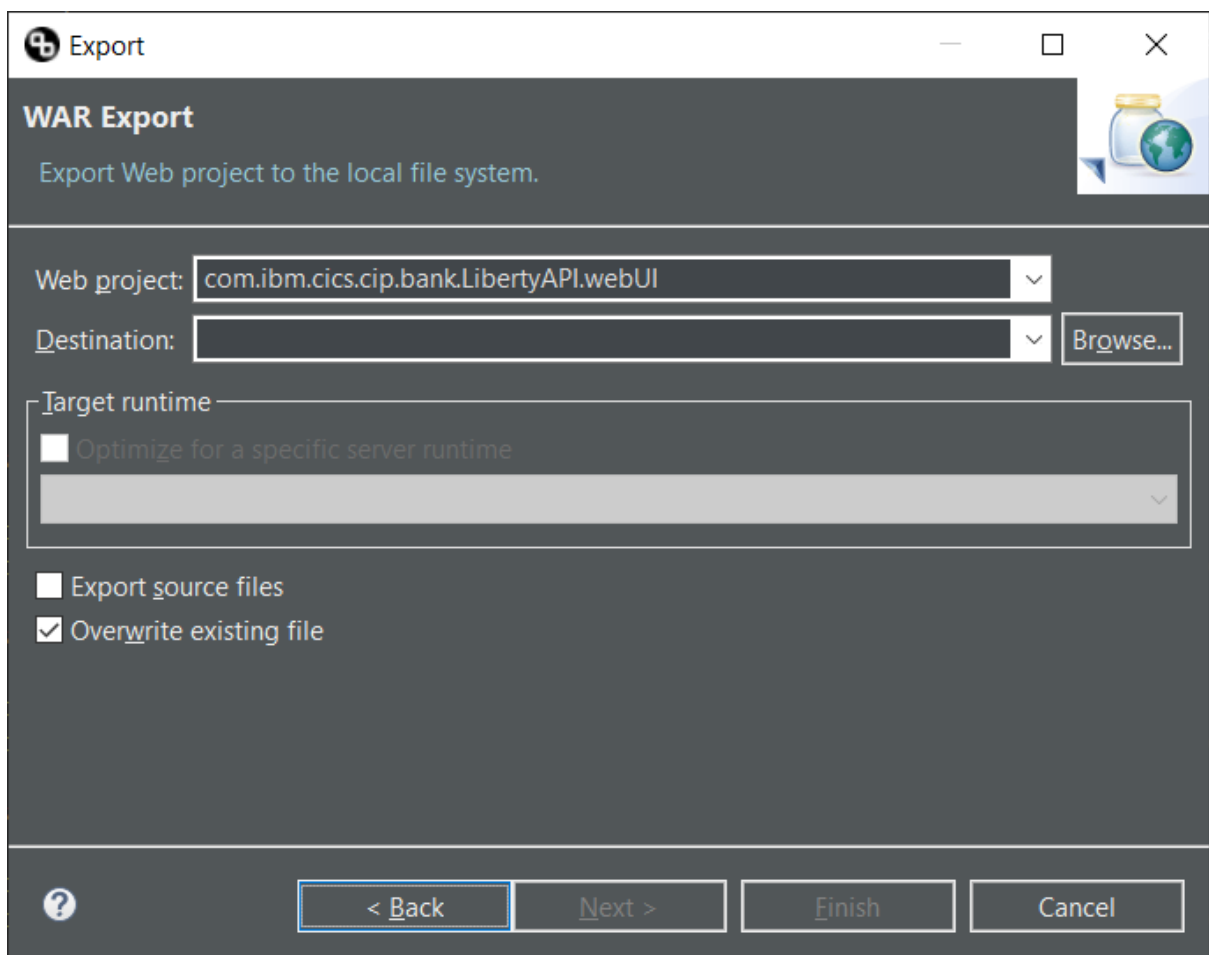
2. RESTART WAZI Developer for Eclipse.

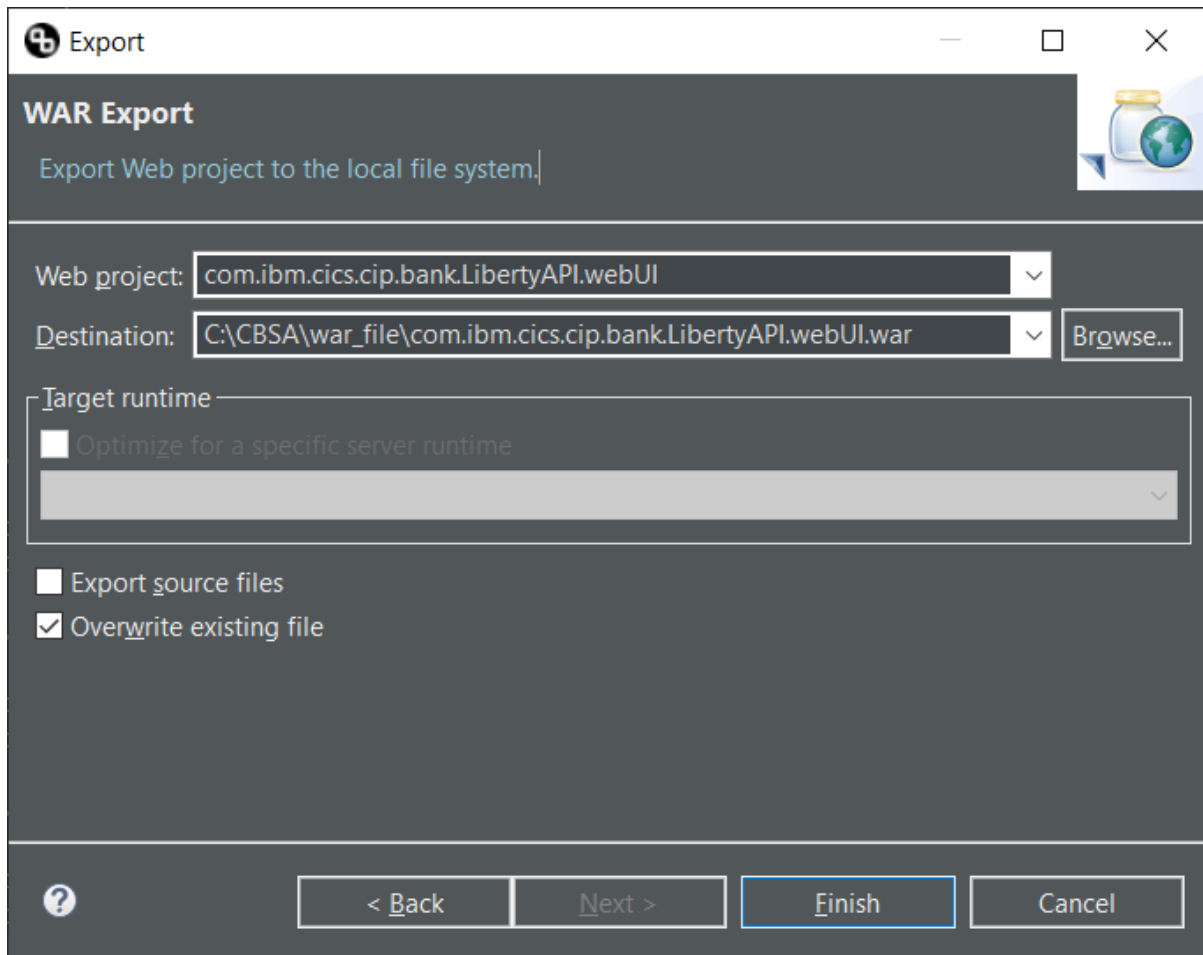
Deploying the application:

1. Right-click the com.ibm.cics.cip.bank.LibertyAPI.webUI project and select EXPORT.
2. We are going to Export as a WAR FILE. If this is not seen as an option immediately, expand Export and search for it. It is under WEB.

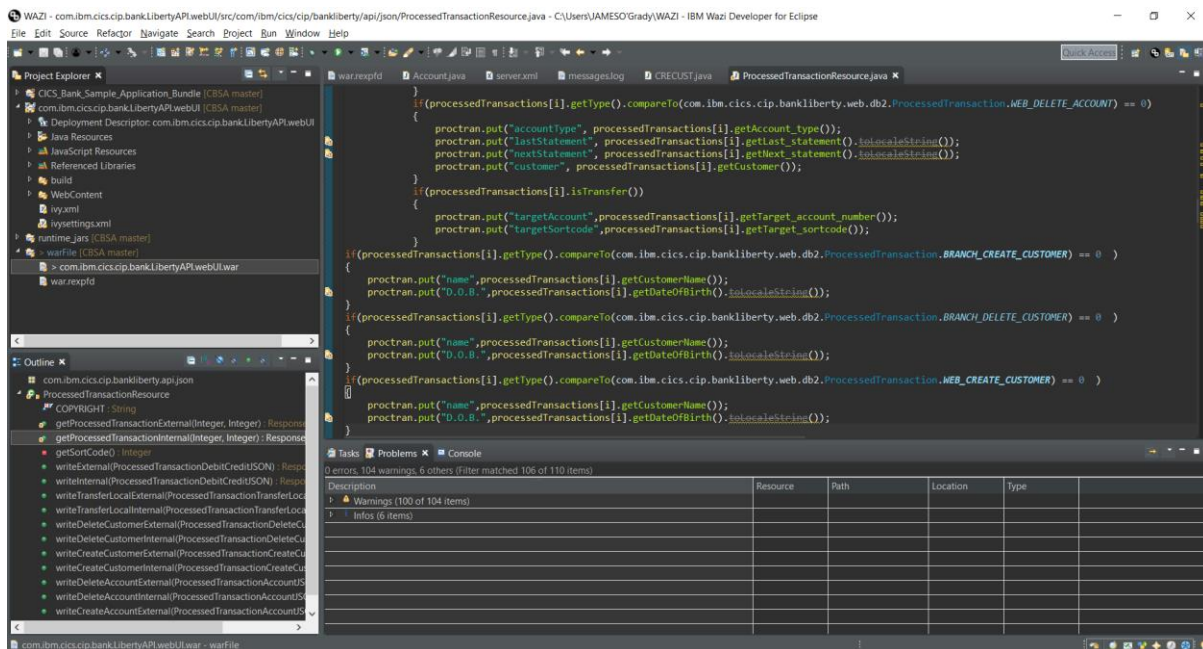


We have to export locally.






This creates a WAR file in a folder called War_file which is also a project in your workspace.



3. Right click THAT and select EXPORT.

This time we are going to export to a remote file system, on the mainframe. In this example the host is called NAZARE-CBSA21.FYRE.IBM.COM, yours may be different.

 Enter Password ×

System type: z/OS

Host name: NAZARE-CBSA21.FYRE.IBM.COM



Connection name: nazare-cbsa21.fyre.ibm.com


User ID:


Password:

☒ Save user ID

☐ Save password

 Enter password 

 RSEC2315 ×



Connection NAZARE-CBSA21.FYRE.IBM.COM has not been secured using SSL/TLS. Proceed anyway?

☐ Do not show this message again



Browse For Folder



Select a folder

Connection: nazare-cbsa21.fyre.ibm.com

New...

/var/cics/CICSTS55/CICSTS55/CBSAWLP/wlp/usr/servers/default

- ▶ ➔ My Home
- ▶ ➔ Root
- ▶ ➔ defaultServer
 - ▶ 📁 configDropins
 - ▶ 📁 dropins
 - ▶ 📁 installedApps
 - ▶ 📁 logs
 - ▶ 📁 resources
 - ▶ 📁 runtime_jars
 - ▶ 📁 tranlog
 - ▶ 📁 workarea

OK

Details >>

Cancel



Remote file system

Please enter a destination folder



- ▶ CICS_Bank_Sample_Application_Bundle [CBSA master]
- ▶ RemoteSystemsTempFiles
- ▶ com.ibm.cics.cip.bank.LibertyAPI.webUI [CBSA master]
- runtime_jars [CBSA master]
- ☐ > warFile [CBSA master]

- ☐ .project
- ☒ > com.ibm.cics.cip.bank.LibertyAPI.webUI.war
- ☐ war.rexpdf

Filter Types...

Select All

Deselect All

Destination folder, qualified by its remote connection name:

LAPTOP-G4R02IV0.nazare-cbsa21.fyre.ibm.com:/u/cicsuser/v

Browse...

Options

- ☐ Review/Synchronize
- ☐ Overwrite existing files without warning
- ☐ Create directory structure for files
- ☐ Create only selected directories
- ☒ Save the settings of this export in the workspace (e.g. /ExportSettings/hello.rexpdf)

Description file: /warFile/war.rexpdf

Browse...

Please enter a destination folder

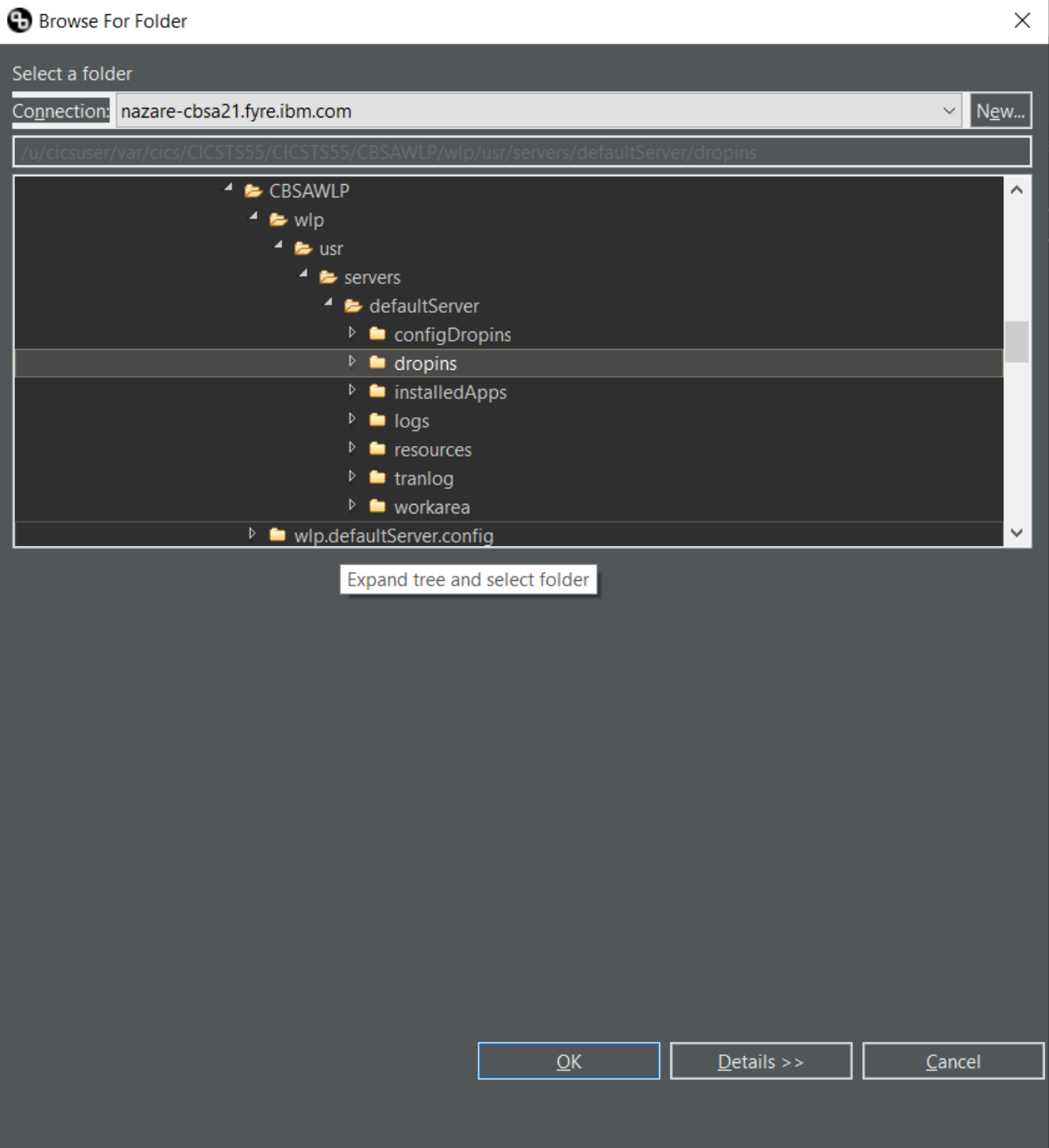


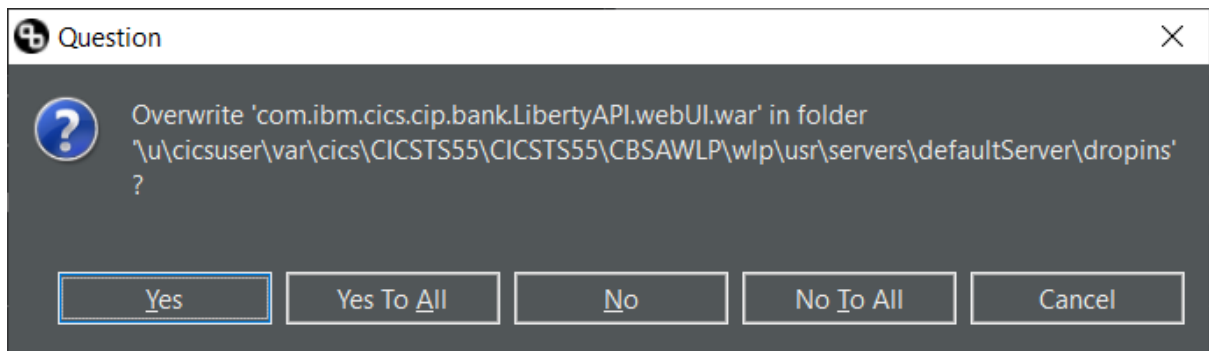
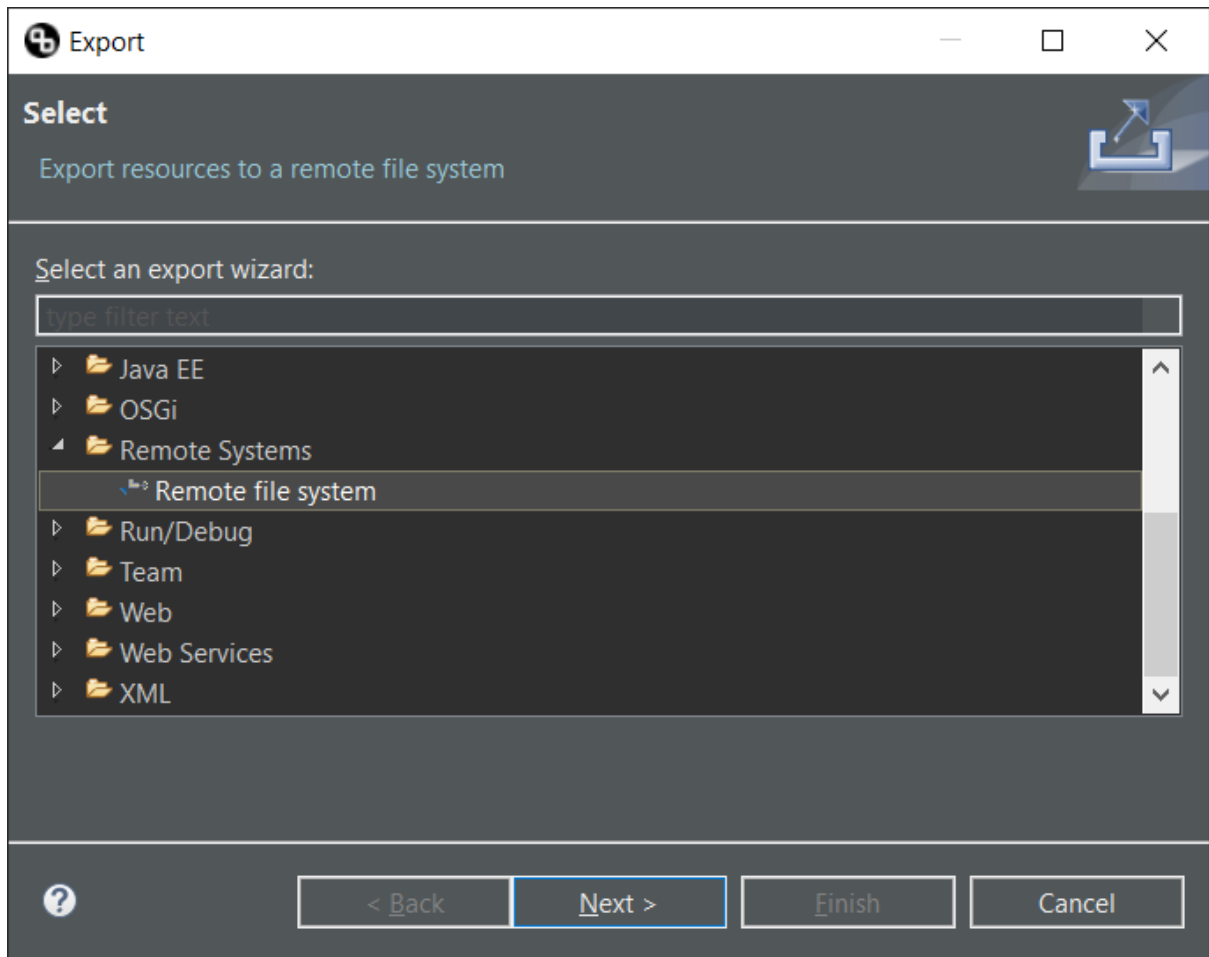
< Back

Next >

Finish

Cancel





4. Then change the permissions of the file to rwx/rwx/---
5. If you experience a "file not found".
Chmod runtimeJars to add rx for everyone.

Checking the Liberty UI:

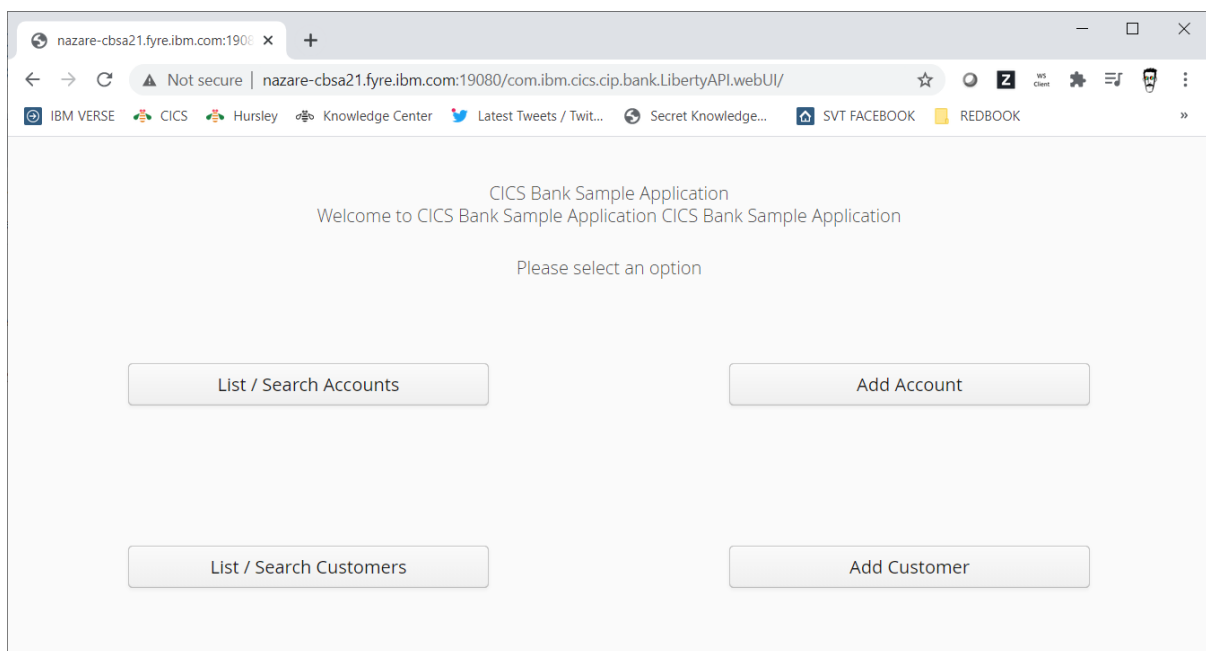
Access the Liberty UI using the following URL (please note that this example utilises a host called Nazare-cbsa.fyre.ibm.com on port number 19080. You should utilise the hostname and port number that you have assigned).

<http://nazare-cbsa21.fyre.ibm.com:19080/com.ibm.cics.cip.bank.LibertyAPI.webUI/>

Please note that on lesser powered machine it may take up to 5 minutes for the JVM server to restart.

1. Once the JVM server has restarted you should be presented with the CBSA Liberty UI main menu.

CICS Bank Sample Application Main Menu



2. Please refer to the “CBSA Liberty UI User Guide” pdf in the **/doc** folder on the repo for a detailed description of the functionality provided via this interface.