Lab 05 Application Modernization using IBM Transformation Advisor

Contents

5.1 Introduction

This is "**Lab05 – IBM Cloud Pak for Applications - App Modernization using Transformation Advisor**" from an IBM Cloud Pak for Applications & App Modernization Proof of technology (PoT). The labs are not required to be executed in order. And, you may skip labs, and only perform the labs that suit your desired learning objectives.

This lab assumes basic familiarity with Docker for building images, running containers, and employing Kubernetes to deploy applications and route application traffic. This lab will introduce Operators which are the preferred mechanism in Red Hat OpenShift Container Platform (RHOCP) for application packaging, deployment, and management.

The full set of labs in the PoT are:

Lab01 - Getting started with Docker
Lab02 - Explore RedHat OpenShift Container Platform
Lab03 - Getting started with Kubernetes
Lab04 – Liberty application deployment using Operators
Lab05 – IBM Cloud Pak for Applications - App Modernization using Transformation Advisor
Lab06 – App Modernization with Java EE Microservices and Liberty
Lab07 – Using Tekton pipelines for CI/CD of microservices to RedHat OpenShift Container Platform

In this lab exercise, we explore the procedure to modernize existing, traditional applications running under traditional WebSphere Application Server (tWAS) and a simple method to bring them to RHOCP – which can run in your own data center or in any public cloud provider including IBM Cloud.

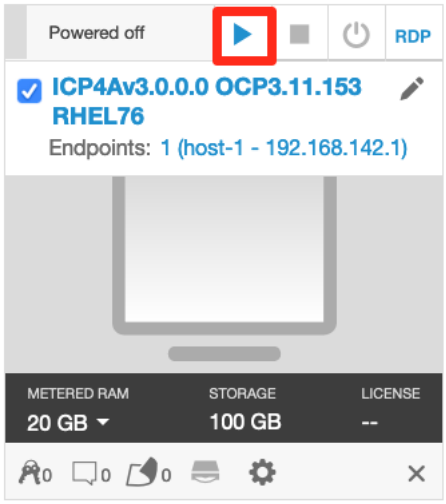
This lab assumes basic familiarity with Docker for building images, running containers, and employing Kubernetes to deploy applications and route application traffic.

The Red Hat OpenShift Container Platform (RHOCP) Kubernetes implementation provides additional features of high availability and enhanced security in their own data centers similar to, or better than, public cloud providers.

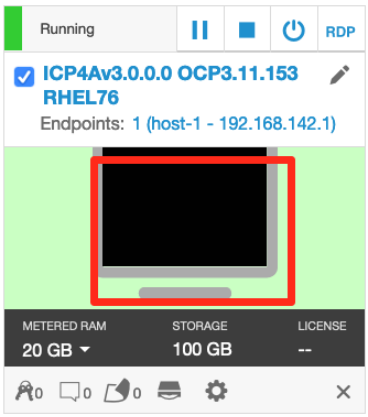
5.2 Access and run the Transformation Advisor

The Transformation Advisor is installed as part of IBM Cloud Pak for Applications

__1. Launch the Lab environment by clicking the **Run this VM** icon.

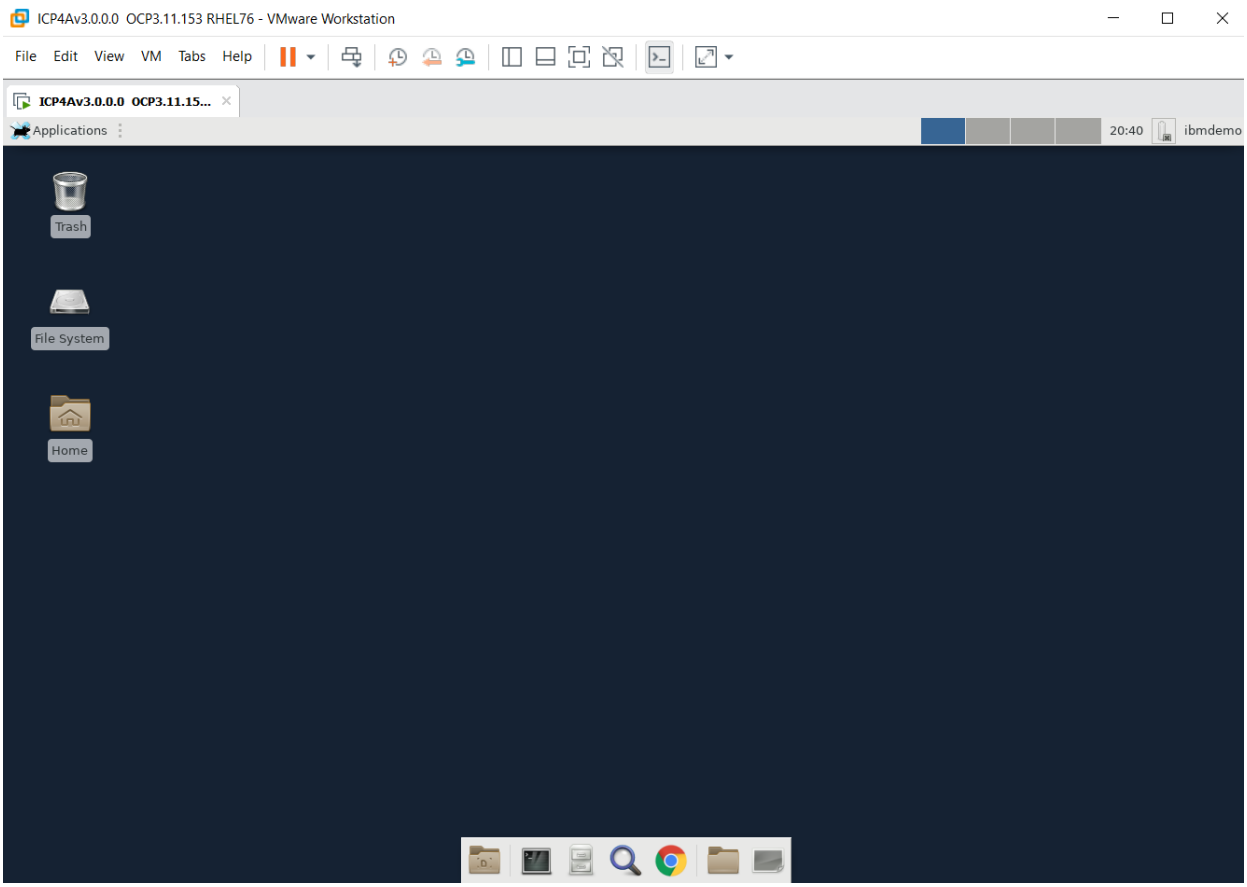


__2. After the VM is running, click its icon to access the VM's desktop.

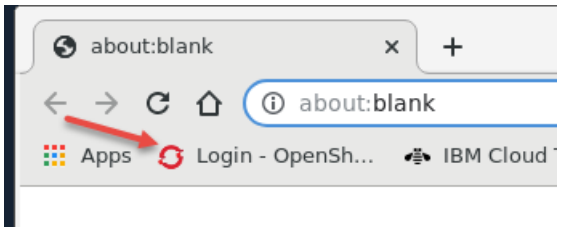


__3. After the VM machine powers on, log with the `ibmdemo` user using the password `passw0rd`

The `ICP4Av3.0.0.0 OCP3.11.153 RHEL76` virtual machine running and its Desktop is displayed in a web browser window.



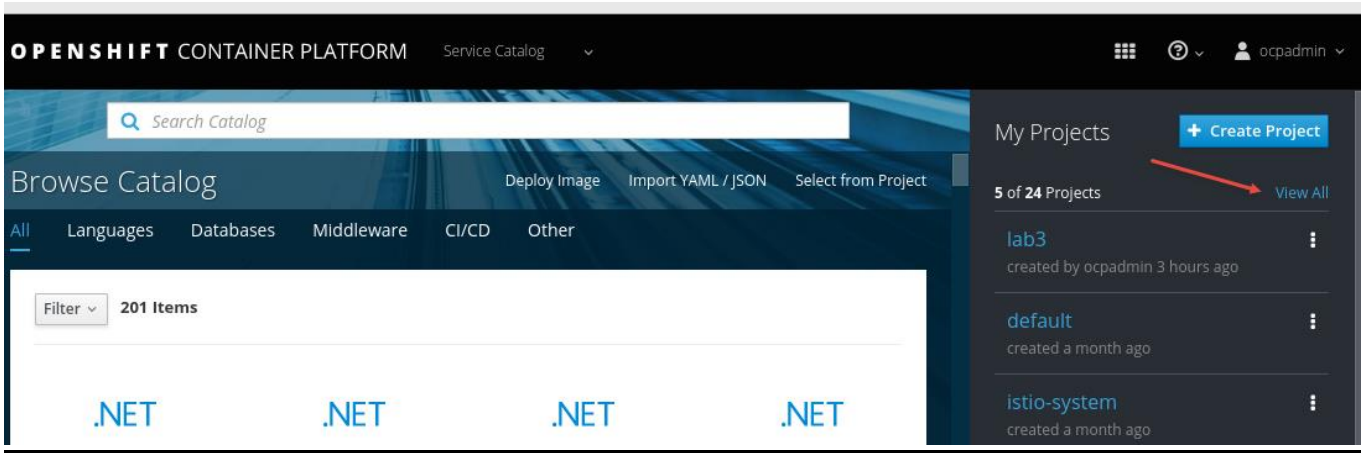
__4. Launch the `Chrome` browser and click on the `Login - OpenShift Container Platform` bookmark to login to OpenShift



__5. Scroll down to the `username` and `password` fields and type `ocpadmin` as he username and `ocpadm1n` (note the "1") as the password. Then click `Log in`

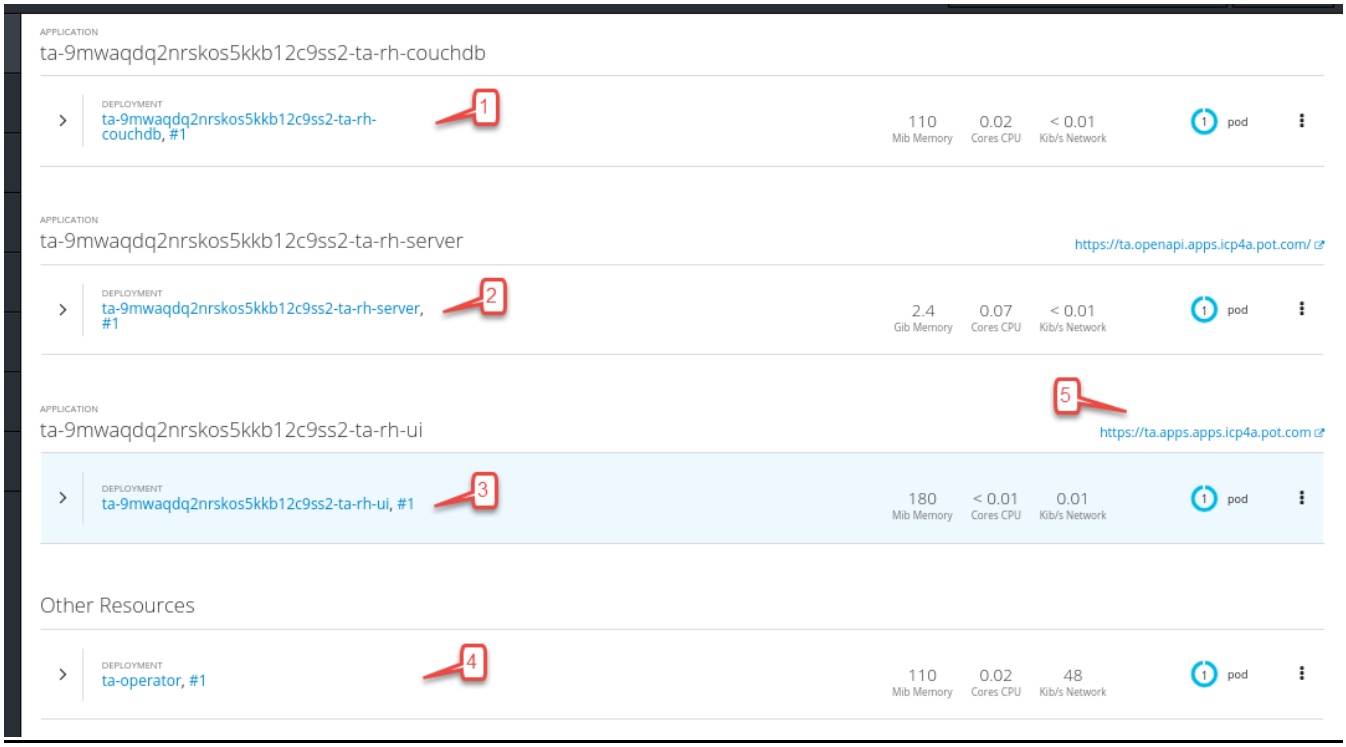
Note: If you receive a "Warning: Potential Security Risk Ahead" page, this is because OCP uses a self-signed certificate. Click `Advanced`, then scroll down and click `Accept the Risk and Continue` which will import the certificate into the browser

__6. After logging in to the OCP console opens click `View All` in `My Projects`



__7. Scroll to the bottom of the Projects list and click on the `ta` project

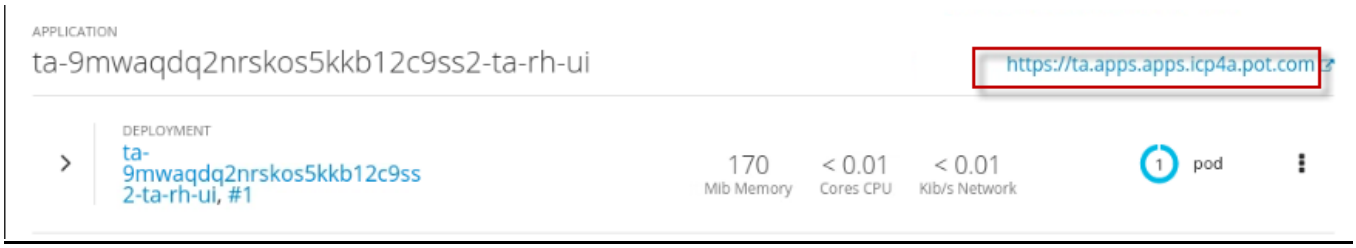
__8. The list of artifacts for the `ta` project is displayed



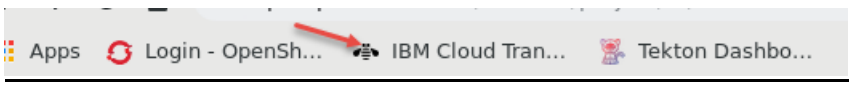
__9. The Transformation Advisor is microservice composed of 3 application pods and an operator pod.

a. Couch DB pod (1) used for storing results
b. Server pod (2) a Liberty server for analysis
c. UI pod (3) Node.js for the browser client
d. The TA operator (4)

__10. To open the TA, either click on the link (5) on the UI pod, or click on the `IBM Cloud Transformation Advisor` link in the bookmark's toolbar. Both shown below.



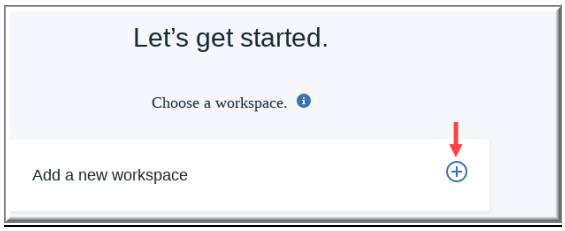
Or use the Bookmark in the Chrome browser to open the TA UI.



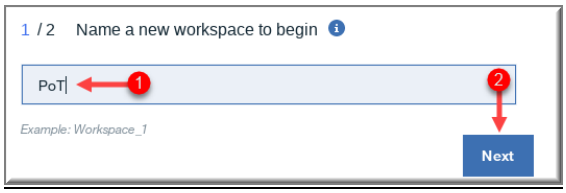
__11. The TA web UI will open in a browser tab, if prompted to login, if prompted to login use `ocpadmin` in the username and `ocpadm1n` (note the "1") in the password as you did previously

Note: If you receive a "Warning: Potential Security Risk Ahead" page, this is because TA uses a self-signed certificate. Click `Advanced`, then scroll down and click `Accept the Risk and Continue` which will import the certificate into the browser

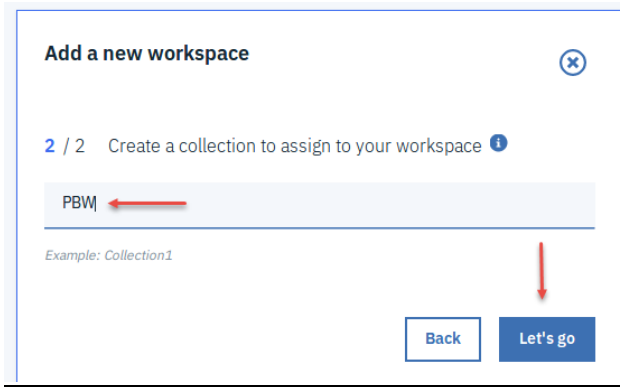
__12. Click `+` to `Add a new workspace`.



__13. Type `PoT` as the workspace name. Then click `Next`



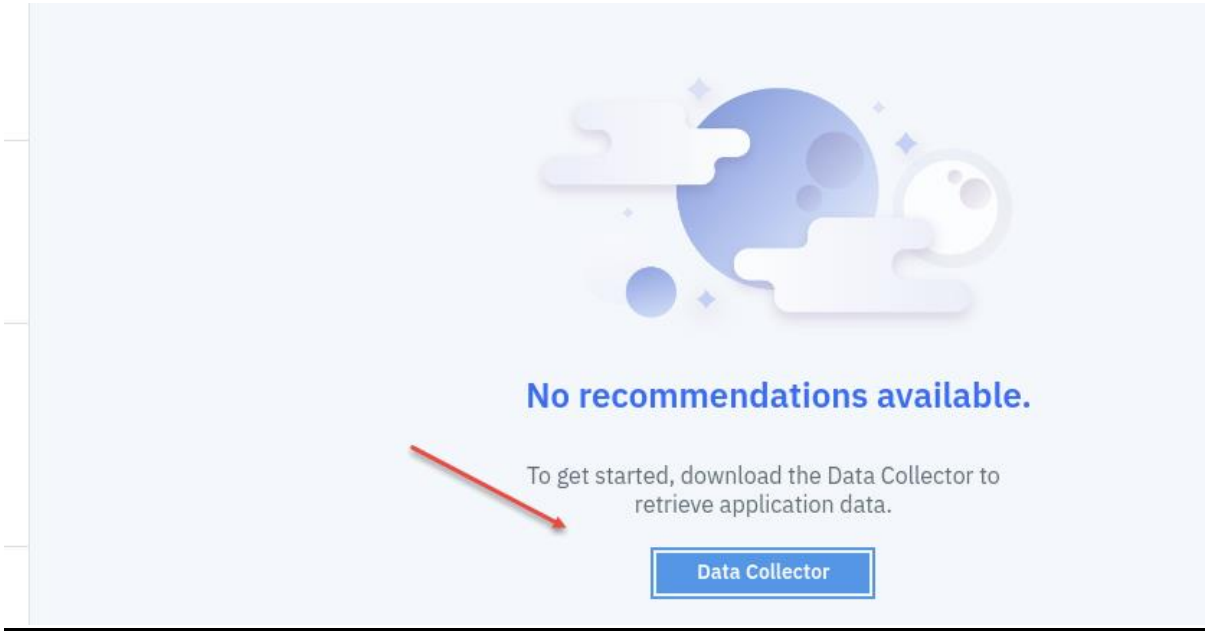
__14. Type `PBW` as collection name. Then click `Let's go`.



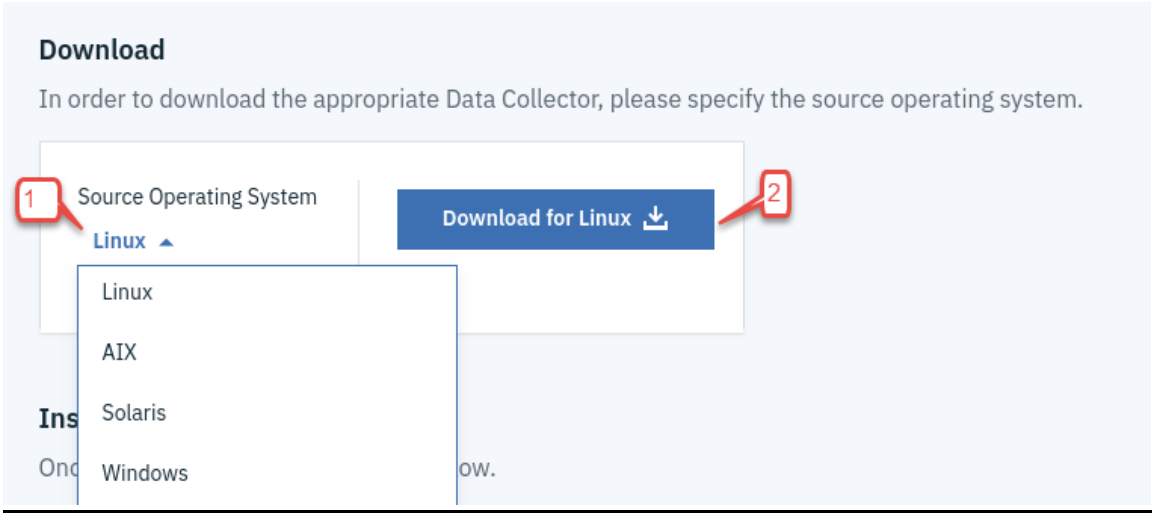
The Transformation Advisor has three steps.

- Download of the Data Collector and copying it to your analysis target server, where WAS, WLS, Tomcat or MQ are running to collect the data.
- Once the data collector tool is run on target platform, it can send the data back to the Transformation Advisor, if firewall configuration allows the data collector to send the data the results are automatically uploaded, otherwise a zip file with the results is moved and imported to the TA for analysis.
- Review of the Data Collector results

__15. Click on `Data Collector.`

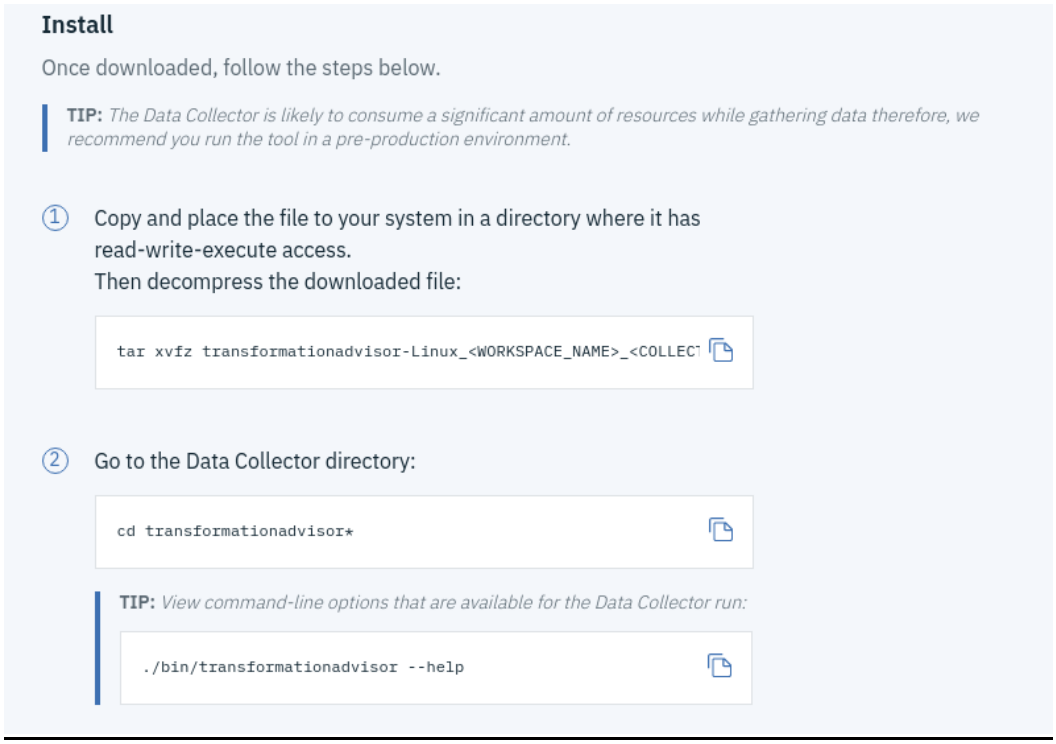


__a. To download the collector, select the `Source Operating System` (1) and then click on `Download` (2).

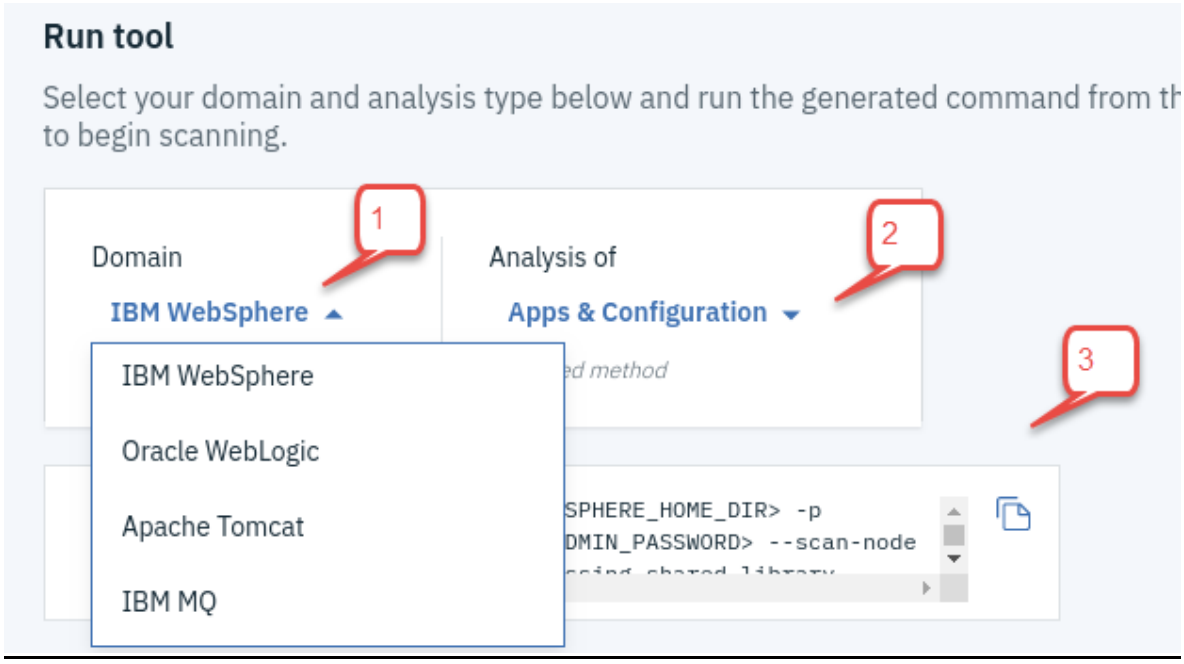


__b. Scroll down and review **but do not execute**, the `Install` steps

	**Note:** do not execute the install steps. These are for your reference only. We have already run the data collector on a WebSphere server for this lab. In subsequent steps, you will load the data collection provided into TA for analysis.



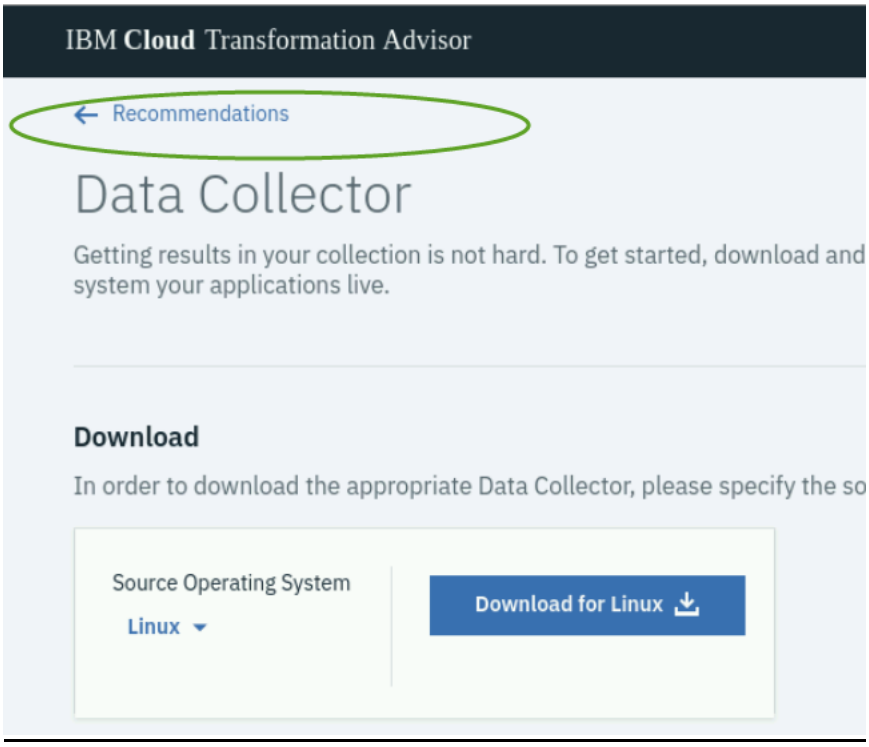
__c. Scroll down and review but do not execute the `Run` options. The `Run` options are based on the `Domain` (1) and the `Analysis` (2) options selected, the appropriate commands for execution are then displayed (3) and can be copied.



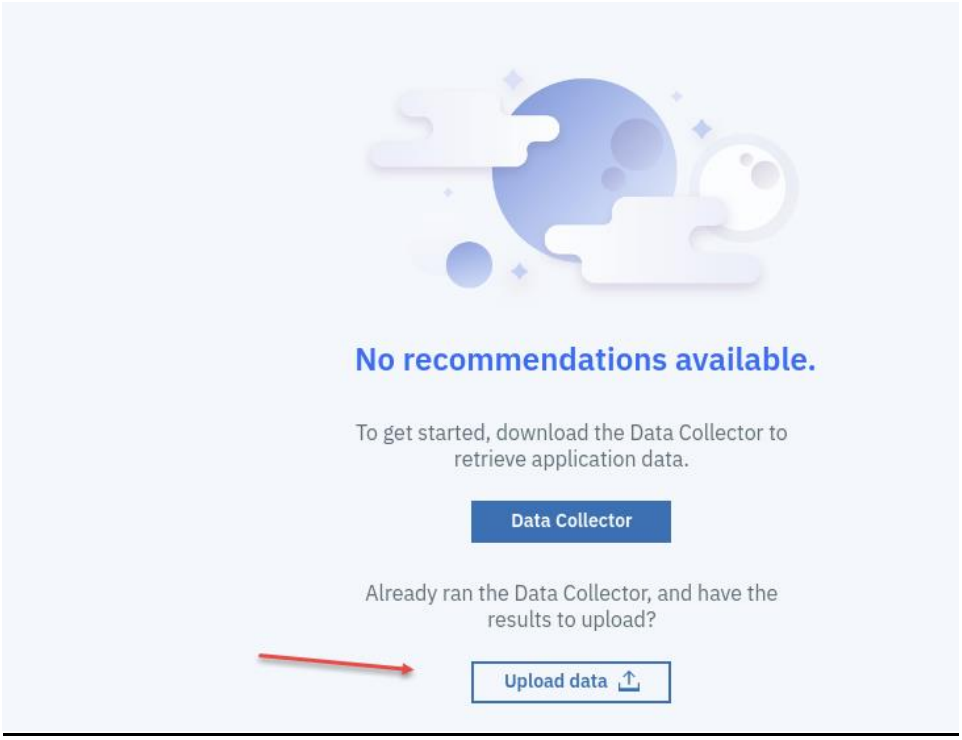
Here is a sample output from the Data Collector.

```
[wasadmin@RHEL73 transformationadvisor-2.0.1]$ ./bin/transformationadvisor -w /opt/IBM/WebSphere/AppServer/ -p Dmgr01 wsadmin wsadmin
Its no longer needed to run the data collector tool
LICENSE INFORMATION

The Programs listed below are licensed under the following License Information terms and conditions in addition to the Program license terms previously agreed to by Client and IBM. If Client does not have previously agreed to license terms in effect for the Program, the International License Agreement for Non-Warranted Programs (Z125-5589-05) applies.

<multiple lines of license omitted for brevity>

The following replaces Items 10.2b and 10.2c:
b. special, incidental, exemplary, or indirect damages or consequential damages; or
c. wasted management time or lost profits, business, revenue, goodwill, or anticipated savings.

Z125-5589-05 (07/2011)

1. I have read and agreed to the license agreements
2. Don't accept the license agreements

1
Licence Accepted



================================================================================
Status:  Failed
Configuration analysis: Completed
--------------------------------------------------------------------------------
Profile   Currently processing: 1/1       Profile name: Dmgr01
--------------------------------------------------------------------------------
Applications         Total: 1    Completed: 1
--------------------------------------------------------------------------------
Time     Elapsed time: 00:01:05          Time remaining: 00:00:00
--------------------------------------------------------------------------------
Progress             >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> 100%
--------------------------------------------------------------------------------
Current Operation:Error occurred:       Problem connecting with server            See log for details.
--------------------------------------------------------------------------------
```

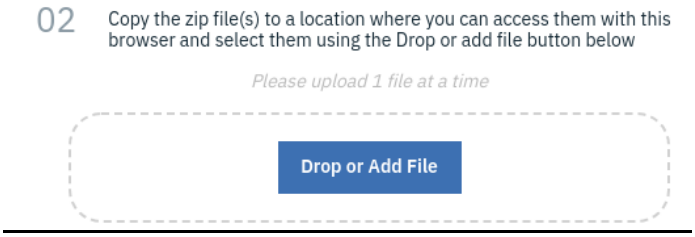
__16. Resume executing instructions; Click `Recommendations` on the top of Data Collector page



__17. Click `Upload Data`

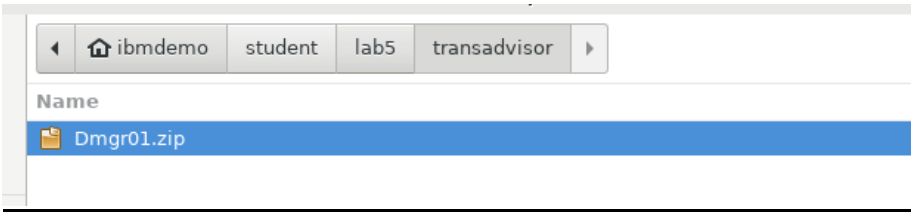


__18. Click the `Drop or Add File` button. (You may need to scroll down)



__19. Click `student -> lab5 ->transadvisor -> Dmgr01.zip`

Dmgr01.zip is the data collection zip file created for this lab. It was run on a WebSphere and provided for this lab for convenience.



__20. Click `Open` (lower right-hand corner).

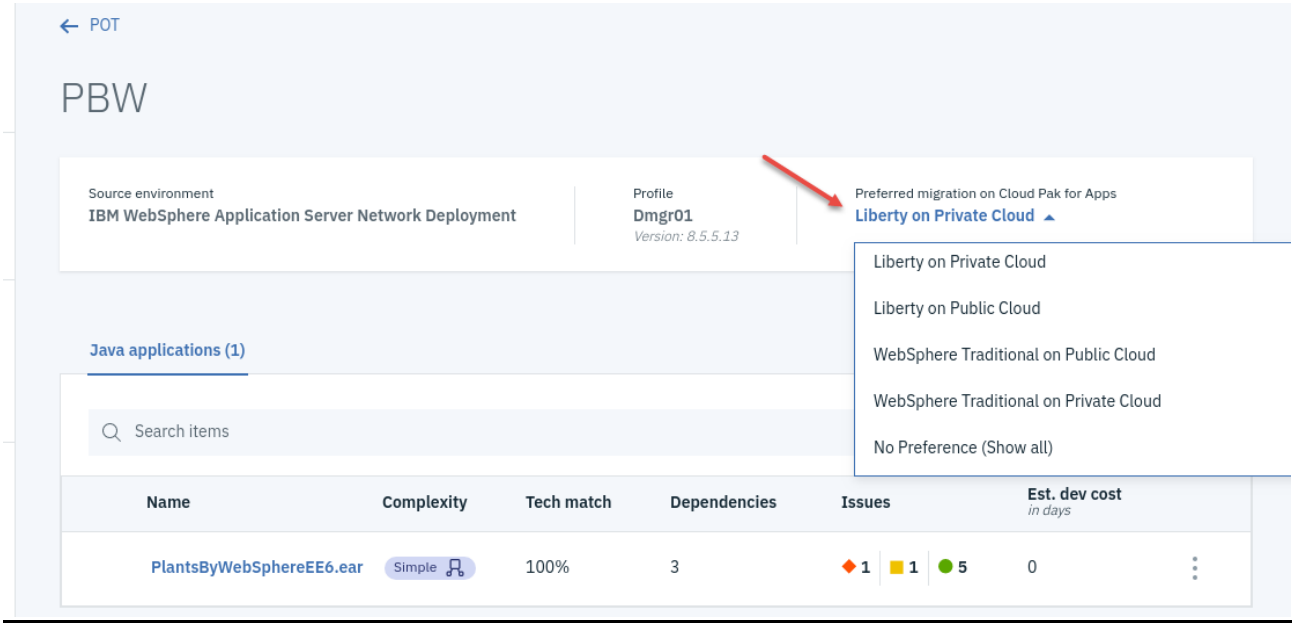
__21. Click `Upload`. (You may need to scroll, if it is not visible.)

__22. The TA analyzes the data collected from the app server and provides summary information on the overview page.

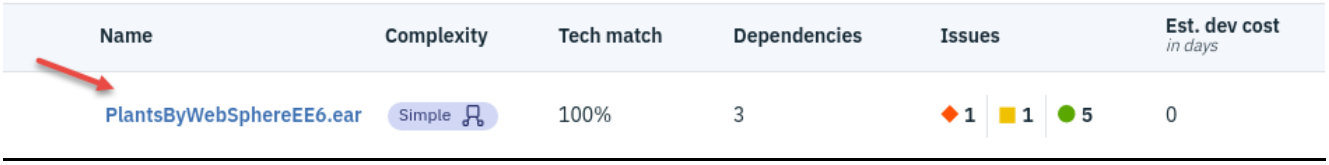
- Application that was analyzed on the WebSphere server
- complexity level
- technology match
- number of dependencies
- Number of issues discovered during analysis
- estimated effort to modernize the apps.

__23. The TA provides various cloud migration options with specific guidance and multiple reports for each of these cloud options.

_a. For the purposes of this PoT, select the `Liberty on Private Cloud` option which is the default option.



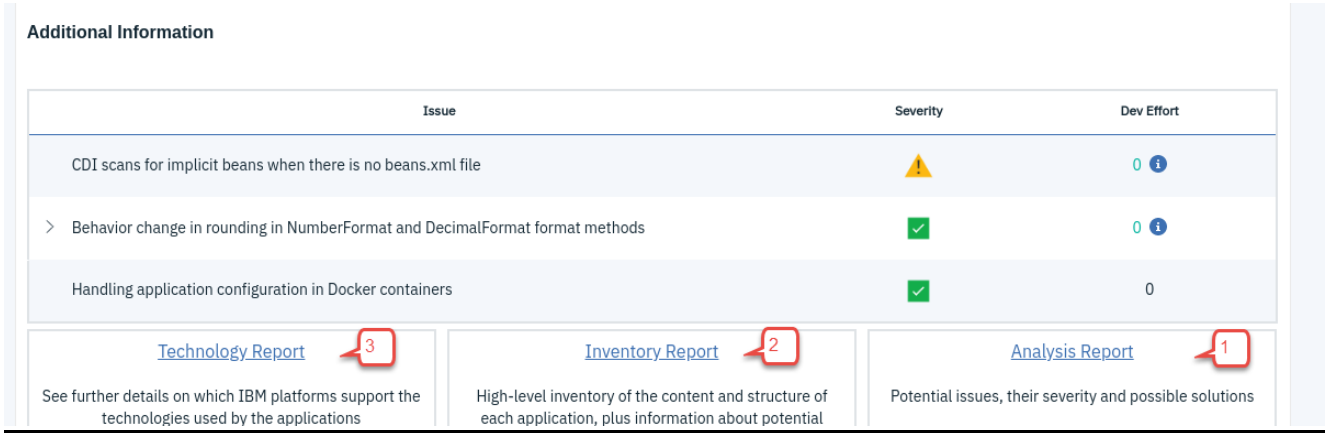
__24. Click on `PlantsByWebSphereEE6.ear` which will expand the analysis of the PlantsByWebSphere application



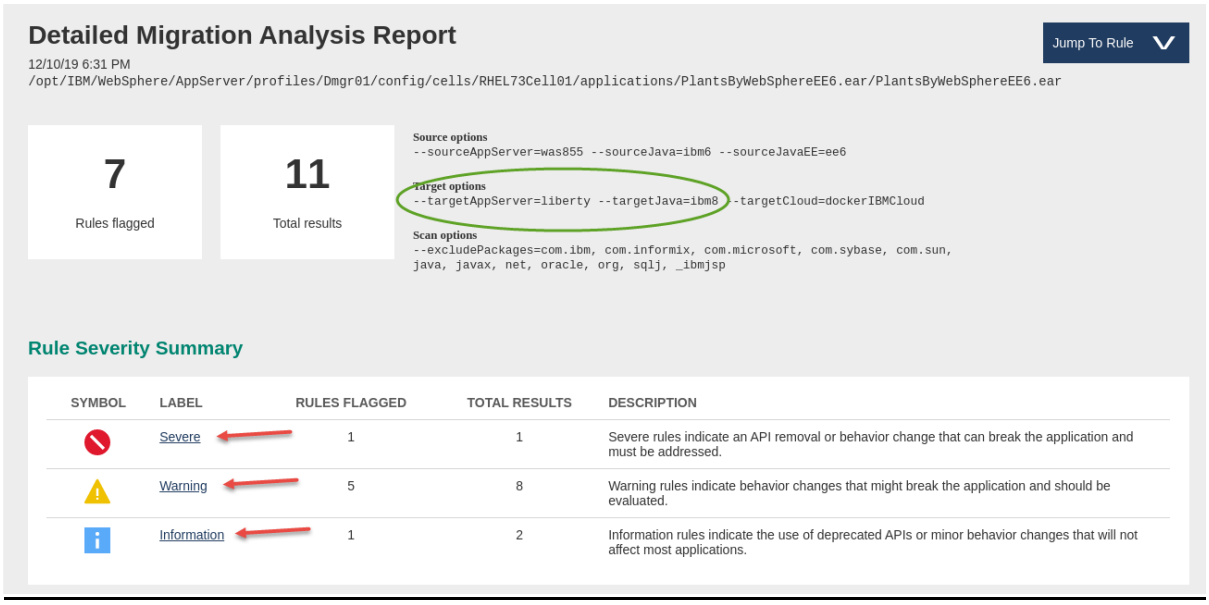
__25. Scroll down, noting the summaries for

- `Complexity`
- `Application Details`
- `Technology Issues`
- `External Dependencies`
- `Additional Information`

__26. Click `Analysis Report`, (1) for more detailed application information, which will open a new tab



The **Detailed Migration Analysis** report is displayed.



Note the `target:` `Because` the `source` was tWAS v8.5.5, which is Java EE 6, the target is also EE6.

Note: If the source was tWAS v9.x, both would be EE7.

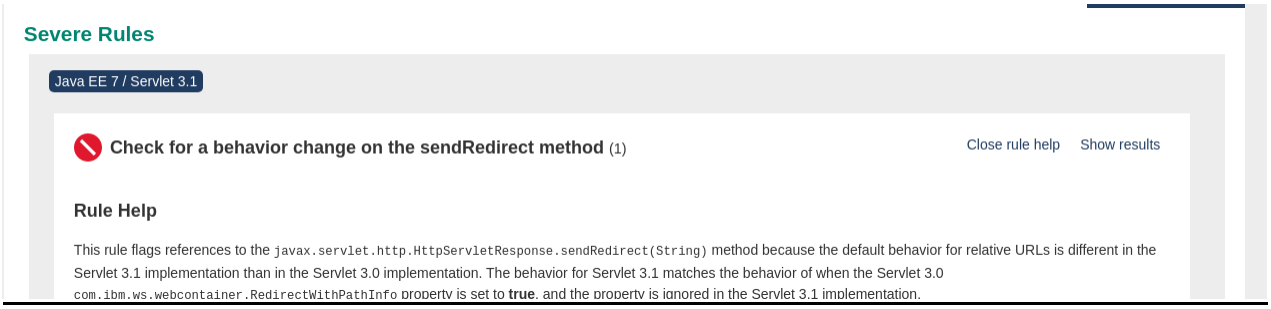
If Java EE 6 is appropriate for your application architecture, no further analysis is needed. However, to analyze an application for Java EE 7 or Java EE 8, you would need to override the defaults.

Options exist in the WebSphere Application Migration Toolkit (WAMT), as well as the TA for overriding defaults.

	A WAMT generated report for Java 8 EE migration: PlantsByWebSphereEE6.ear_MigrationReport.html Is in the /home/ibmdemo/student/lab5 folder and can be opened in a browser for comparison with the TA generated report Java 6 EE report. You will note that several application changes are required for Java 8 EE, though all these changes were accomplished in a couple of hours.

_a. In the report, click on the Severe label, to review the details of this item.

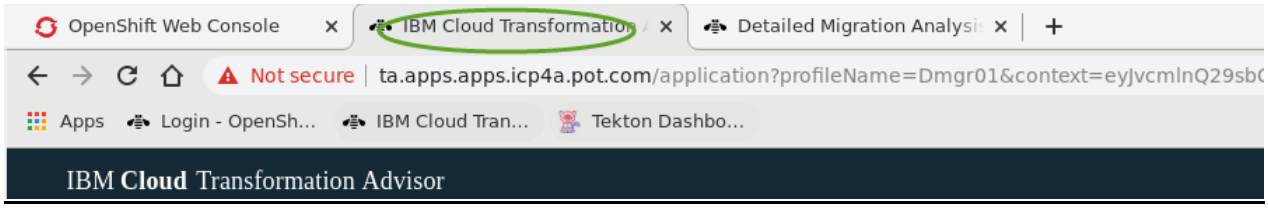
While the presence of a severe warning may seem ominous, in this case, it's easy for development to review the rule help and then examine the application code to determine if there's an impact



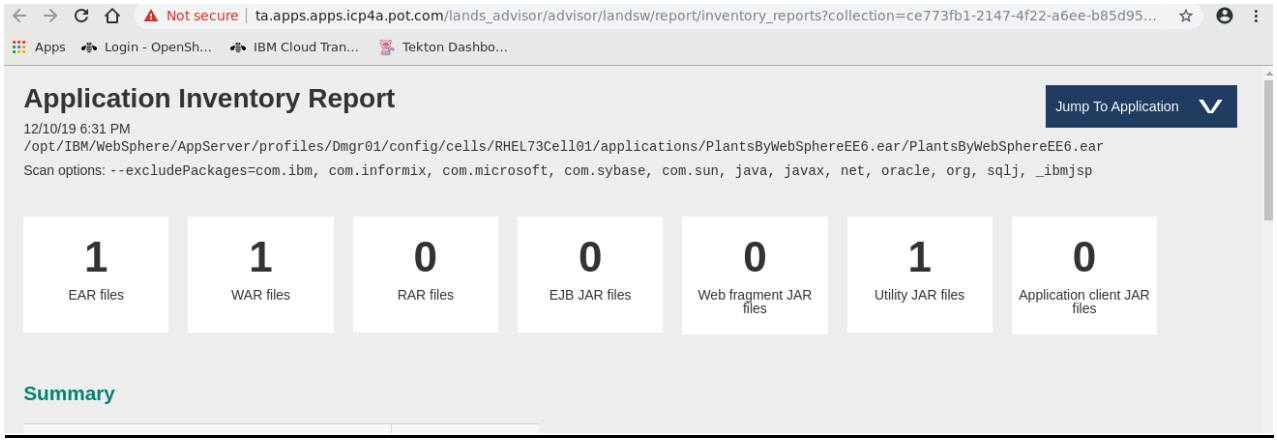
_b. In a similar manner, click on the Warning and Information labels and review the guidance

_c. In the case of this application with a Java 6 EE target, no code changes were needed, and the analysis took less than an hour.

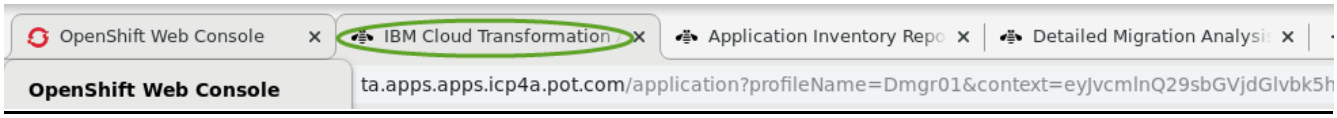
__27. Return to the Cloud Transformation Advisor browser tab and click on Inventory Report (2) from the screen shot above in step #26, which will open in a new tab



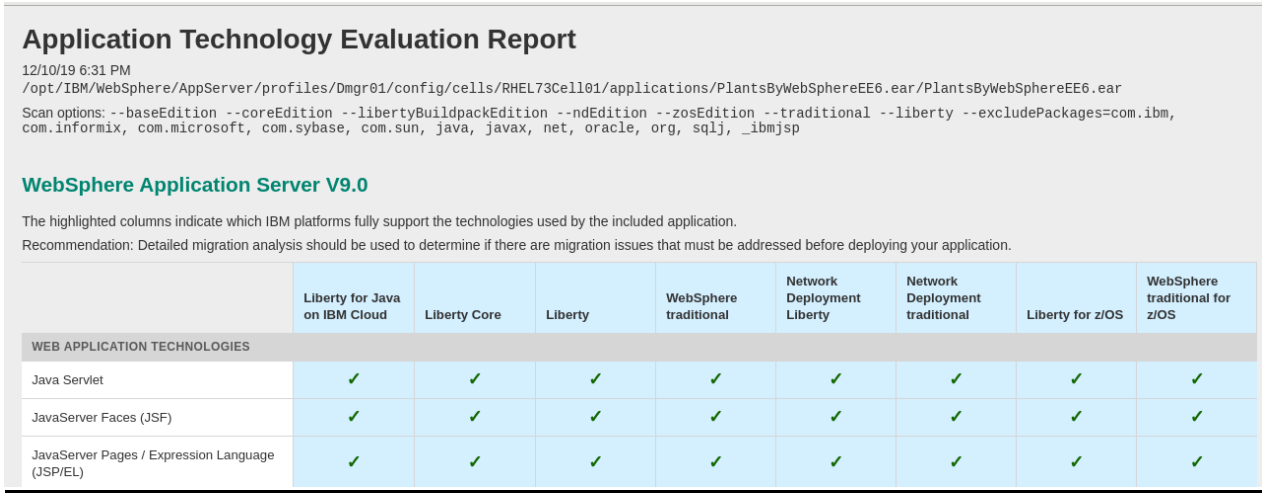
_a. Scroll down and **review** the `Inventory Report.` `This report` is especially useful in larger applications. It is helpful for both understanding the existing application components, as well as identifying potential microservices which can be exposed for application modernization.



__28. Return to the `Cloud Transformation Advisor` browser tab and click on `Technology Report` (3) from the screen shot above in #26, which will open in a new tab



_a. Scroll down `Technology Report,` which can quickly help assess applications which use APIs that are available in Liberty, as opposed to applications which might best remain in tWAS. This report can be used to analyze existing applications and their API use in potential microservices which can be exposed as part of application modernization.



Application Technology Evaluation Report

12/10/19 6:31 PM
/opt/IBM/WebSphere/AppServer/profiles/Dmgr01/config/cells/RHEL73Cell01/applications/PlantsByWebSphereEE6.ear/PlantsByWebSphereEE6.ear

Scan options: --baseEdition --coreEdition --libertyBuildpackEdition --ndEdition --zosEdition --traditional --liberty --excludePackages=com.ibm, com.informix, com.microsoft, com.sybase, com.sun, java, javax, net, oracle, org, sqlj, _ibmjsp

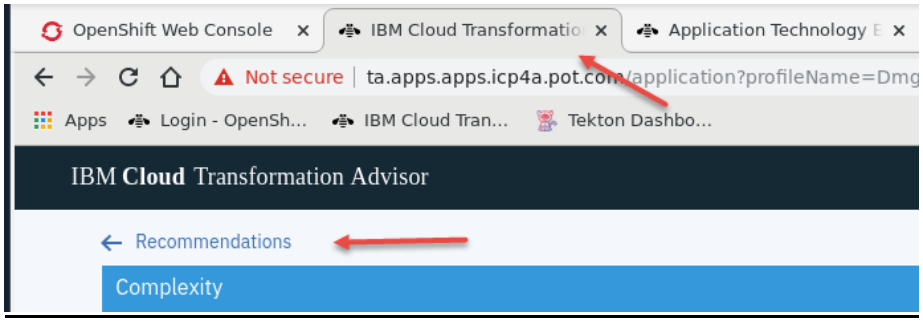
WebSphere Application Server V9.0

The highlighted columns indicate which IBM platforms fully support the technologies used by the included application.

Recommendation: Detailed migration analysis should be used to determine if there are migration issues that must be addressed before deploying your application.

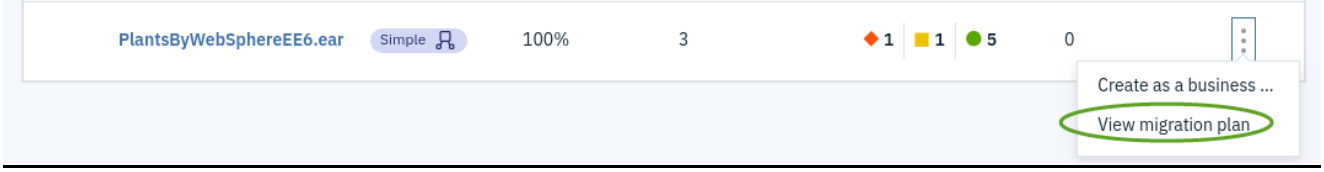
	Liberty for Java on IBM Cloud	Liberty Core	Liberty	WebSphere traditional	Network Deployment Liberty	Network Deployment traditional	Liberty for z/OS	WebSphere traditional for z/OS
WEB APPLICATION TECHNOLOGIES								
Java Servlet	✓	✓	✓	✓	✓	✓	✓	✓
JavaServer Faces (JSF)	✓	✓	✓	✓	✓	✓	✓	✓
JavaServer Pages / Expression Language (JSP/EL)	✓	✓	✓	✓	✓	✓	✓	✓

__29. Return to the `Cloud Transformation Advisor` browser tab and click on `Recommendations`



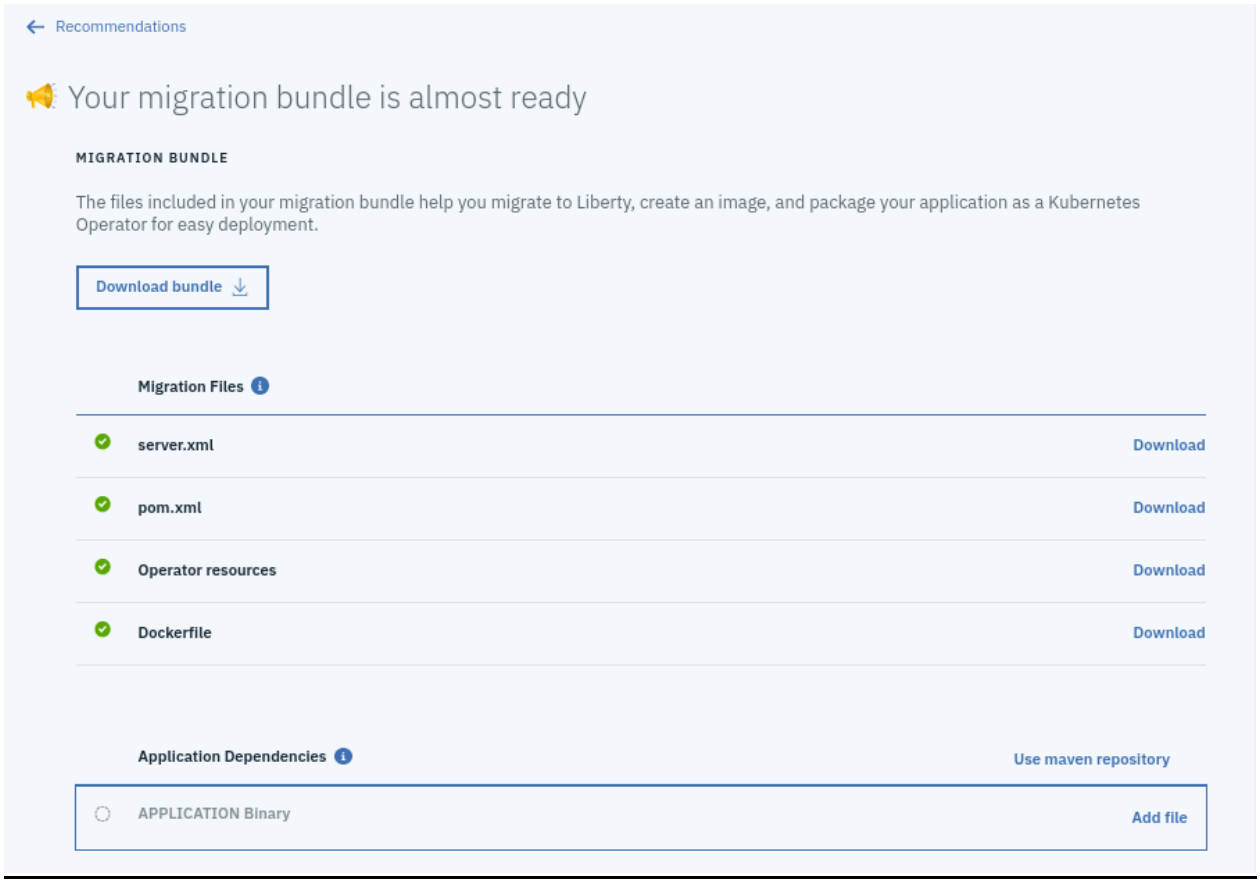
__30. Click the `3 vertical dots` located next to PlantsByWebSphereEE6.ear application

__31. Click `View Migration Plan`



The TA generates migration files necessary to build and deploy the application to RedHat OpenShift as you modernize the application for containers and cloud.

As illustrated below, the `server.xml`, `Dockerfile`, `Operator Resources,` and `pom.xml` files are generated for the application. The generated files assist you in moving your application to RHOSCP on your own cloud. You can download each of them individually, or as a bundle which can optionally include the application binaries.


Recommendations
Your migration bundle is almost ready
MIGRATION BUNDLE
The files included in your migration bundle help you migrate to Liberty, create an image, and package your application as a Kubernetes Operator for easy deployment.
Download bundle
Migration Files
server.xml
Download
pom.xml
Download
Operator resources
Download
Dockerfile
Download
Application Dependencies
Use maven repository
APPLICATION Binary
Add file

5.3 Application Modernization

In this section, you will use the files generated by Transformation Advisor's migration plan in order to build the application in a Docker container, and deploy the application to RHOSCP using the Liberty Operator.

__1. Click `Terminal` from the bottom of the desktop to open a command line terminal.

__1. Type `cd ~/student/lab5/appmodern` to change directory

```
[ibmdemo@icp4a student]$ cd ~/student/lab5/appmodern/

[ibmdemo@icp4a appmodern]$ $
```

__2. Login to OpenShift using the `oc login` command. Enter `ocpadmin` for the username and `ocpadm1n` (note the "1", not "i") for the password

```
ibmdemo@icp4a appmodern]$ oc login

Authentication required for https://icp4a.pot.com:8443 (openshift)
Username: ocpadmin
Password:
Login successful.

You have access to the following projects and can switch between them with 'oc
project <projectname>':

  * default
    istio-system
    kabanero
    knative-eventing
    knative-serving
    knative-sources
    kube-public
    kube-service-catalog
    kube-system
    lab3
    lab4
    management-infra
    openshift
    openshift-console
    openshift-infra
    openshift-logging
    openshift-metrics-server
    openshift-monitoring
    openshift-node
    openshift-node-problem-detector
    openshift-pipelines
    openshift-sdn
    openshift-web-console
    operator-lifecycle-manager
    ta
Using project "default".
```

__3. Type `ls` to review the contents of this directory

The directory includes the **migration bundle** generated by Transformation Advisor:

- **plantsbywebsphereee6_migrationBundle.zip**

The **migration bundle** has already been unzipped to this director, and in turn created many of the directories and some of the artifacts displayed.

```
[ibmdemo@icp4a appmodern]$ ls
01-buildDocker.sh              docs
02-createsecrets.sh            operator
03-createOperatorArtifacts.sh  plantsbywebsphereee6_migrationBundle.zip
04-deployApplication.sh        pom.xml
Dockerfile                     src
Dockerfile.bak                 target
```

__4. Type `cat Dockerfile` to review the Dockerfile that is used to build the docker image

```
[ibmdemo@icp4a appmodern]$ cat Dockefile

#If on slow network comment the line below

#FROM docker.io/ibmcom/websphere-liberty:19.0.0.6-kernel-ubi-min

# If on slow network uncomment the line below, image has Liberty server and features
FROM lab5:latest


COPY  src/main/liberty/config/server.xml /config/
COPY  target/plantsbywebsphereee6.ear /config/apps/
COPY  src/main/liberty/lib/DB2libs/db2jcc4.jar /config/resources/DB2libs/

USER root
RUN chown default:root -R /opt/ibm/wlp/usr/servers/defaultServer
USER 1001

RUN configure.sh
```

The Dockerfile Is based on the TA created artifact but the maven build commands have been removed along with the following modifications:

- the PULL which has been modified to use the **Liberty kernel image**, not the Liberty webProfile7 image (this because the lab uses EE6 APIs, not EE7 APIs)
- Multi-stage Docker builds are not available with versions of Docker lower than 17.0.5, additionally, so the COPY commands have been modified, and then the chown command is run separately.

- A COPY command was added to add the **DB2 JDBC driver** to the image
- Optionally you can review the TA generated Dockerfile using the command `cat Dockerfile.bak` which includes maven build commands as well as the multi-stage Docker build COPY commands

__5. The TA generated Liberty **server.xml** has also been modified.

_a. You can review the **TA generated** server.xml by typing:

```
cat src/main/liberty/config/server.xml.bak
```

_b. You can review the **modified** server.xml used for this lab by typing

```
cat src/main/liberty/config/server.xml
```

The primary difference between the TA generated server.xml and the modified server.xml are:

- the deletion of duplicated JDBC provider and datasource entries which result from the differences in the tWAS and Liberty configuration models
- adding environment variables for the datasource which are obtained from a Kubernetes secret

__6. Create a new OpenShift Project.

An **OpenShift project** is a Kubernetes namespace with some additional annotations which set the scope for the Objects, such as pods, services, replication controllers, etc.;

Policies which are rules for the allowed actions; Constraints (or quotas) for each kind of object, as well as Service Accounts for the project.

_a. Type `oc new-project lab5` to create the **lab5** project and switch your context to that project.

```
[ibmdemo@icp4a appmodern]$ oc new-project lab5

Now using project "lab5" on server "https://icp4a.pot.com:8443".

You can add applications to this project with the 'new-app' command. For
example, try:

    oc new-app centos/ruby-25-centos7~https://github.com/sclorg/ruby-ex.git

to build a new example application in Ruby.
[ibmdemo@icp4a ~]$
```

Note: For this lab, we created some scripts to automate the multiple manual command line entries needed to build and deploy the application

__7. Review the **01-buildDocker.sh** script by typing `cat 01-buildDocker.sh`

As you can see this script does the following:

- builds the Docker image
- tags it for RHOSCP internal registry
- authenticates with the RHOCP registry
- pushes the image to the RHOSCP internal registry

```
[ibmdemo@icp4a appmodern]$ cat 01-buildDocker.sh
IMAGENAME=plantsbywebsphereee6
PROJECT=lab5

echo ============================================================
echo Build Liberty Docker image for $IMAGENAME
echo ============================================================
echo

docker build -t $IMAGENAME .

echo ============================================================
echo Tag Liberty Docker image for $IMAGENAME
echo ============================================================
echo
docker tag $IMAGENAME:latest docker-registry.default.svc:5000/$PROJECT/$IMAGENAME:latest

docker login -u $(oc whoami) -p $(oc whoami -t) docker-registry.default.svc:5000

echo ============================================================
echo Push Liberty Docker image for $IMAGENAME
echo ============================================================
echo
docker push docker-registry.default.svc:5000/$PROJECT/$IMAGENAME:latest

[ibmdemo@icp4a appmodern]$
```

__8. Run the 01-buildDocker.sh script by typing: `./01-buildDocker.sh`

Note: The output from the script is illustrated below:

```
[ibmdemo@icp4a appmodern]$ ./01-buildDocker.sh
 ============================================================
Build Liberty Docker image for plantsbywebsphereee6
 ============================================================

Sending build context to Docker daemon 10.57 MB
Step 1/8 : FROM docker.io/ibmcom/websphere-liberty:19.0.0.6-kernel-ubi-min
 ---> 4b96d9ab9f54
Step 2/8 : COPY src/main/liberty/config/server.xml /config/
 ---> f04ab508e409
Removing intermediate container f7d57cb407d1
Step 3/8 : COPY target/plantsbywebsphereee6.ear /config/apps/
 ---> f456768edd13
Removing intermediate container 0b0f0ff496d3
Step 4/8 : COPY src/main/liberty/lib/DB2Libs/db2jcc4.jar /config/resources/DB2Libs/
 ---> aa235054165e
Removing intermediate container d2ef019a53d0
Step 5/8 : USER root
 ---> Running in e49f47c2d78a
 ---> daea68991b78
Removing intermediate container e49f47c2d78a
Step 6/8 : RUN chown default:root -R /opt/ibm/wlp/usr/servers/defaultServer
 ---> Running in 646a42a8f231

 ---> 00e1aaf11dfc
Removing intermediate container 646a42a8f231
Step 7/8 : USER 1001
 ---> Running in b1344378226f
 ---> 79a118c61a55
Removing intermediate container b1344378226f
Step 8/8 : RUN configure.sh
 ---> Running in 84c56cdf6296

+ WLP_INSTALL_DIR=/opt/ibm/wlp
+ SHARED_CONFIG_DIR=/opt/ibm/wlp/usr/shared/config
+ SHARED_RESOURCE_DIR=/opt/ibm/wlp/usr/shared/resources
+ SNIPPETS_SOURCE=/opt/ibm/helpers/build/configuration_snippets
+ SNIPPETS_TARGET=/config/configDropins/overrides
+ mkdir -p /config/configDropins/overrides
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ '[' '' == client ']'
+ '[' '' == embedded ']'
+ '[' '' == true ']'
+ '[' '' == true ']'
+ installUtility install --acceptLicense defaultServer
Checking for missing features required by the server ...
The server requires the following additional features: jsp-2.3 transportsecurity-1.0 jsf-2.2
servlet-3.1 jndi-1.0 cdi-1.2 javamail-1.5 beanvalidation-1.1 ejblite-3.2 jpa-2.0.  Installing
features from the repository ...
```

```
Establishing a connection to the configured repositories ...
This process might take several minutes to complete.

Successfully connected to all configured repositories.

Preparing assets for installation. This process might take several minutes to complete.

Additional Liberty features must be installed for this server.

To install the additional features, review and accept the feature license agreement:
The --acceptLicense argument was found. This indicates that you have
accepted the terms of the license agreement.


Step 1 of 32: Downloading el-3.0 ...
Step 2 of 32: Installing el-3.0 ...
Step 3 of 32: Downloading servlet-3.1 ...
Step 4 of 32: Installing servlet-3.1 ...
Step 5 of 32: Downloading jsp-2.3 ...
Step 6 of 32: Installing jsp-2.3 ...
Step 7 of 32: Downloading ssl-1.0 ...
Step 8 of 32: Installing ssl-1.0 ...
Step 9 of 32: Downloading transportSecurity-1.0 ...
Step 10 of 32: Installing transportSecurity-1.0 ...
Step 11 of 32: Downloading jsf-2.2 ...
Step 12 of 32: Installing jsf-2.2 ...
Step 13 of 32: Downloading jndi-1.0 ...
Step 14 of 32: Installing jndi-1.0 ...
Step 15 of 32: Downloading cdi-1.2 ...
Step 16 of 32: Installing cdi-1.2 ...
Step 17 of 32: Downloading javaMail-1.5 ...
Step 18 of 32: Installing javaMail-1.5 ...
Step 19 of 32: Downloading beanValidation-1.1 ...
Step 20 of 32: Installing beanValidation-1.1 ...
Step 21 of 32: Downloading ejbLite-3.2 ...
Step 22 of 32: Installing ejbLite-3.2 ...
Step 23 of 32: Downloading servlet-3.0 ...
Step 24 of 32: Installing servlet-3.0 ...
Step 25 of 32: Downloading beanValidation-1.0 ...
Step 26 of 32: Installing beanValidation-1.0 ...
Step 27 of 32: Downloading jdbc-4.0 ...
Step 28 of 32: Installing jdbc-4.0 ...
Step 29 of 32: Downloading jpa-2.0 ...
Step 30 of 32: Installing jpa-2.0 ...
Step 31 of 32: Validating installed fixes ...
Step 32 of 32: Cleaning up temporary files

All assets were successfully installed.

Start product validation...
Product validation completed successfully.
+ find /opt/ibm/fixes -type f -name '*.jar' -print0
+ sort -z
+ xargs -0 -n 1 -r -I '{}' java -jar '{}' --installLocation /opt/ibm/wlp
+ find /opt/ibm/wlp -perm -g=w -print0
+ xargs -0 -r chmod -R g+rw
```

```
Starting server defaultServer.
Server defaultServer started with process ID 115.
+ /opt/ibm/wlp/bin/server stop

Stopping server defaultServer.
Server defaultServer stopped.
+ rm -rf /output/resources/security/ /output/messaging /logs/console.log /logs/ffdc
/logs/messages.log /logs/messages_20.01.30_22.37.24.0.log /opt/ibm/wlp/output/.classCache
+ chmod -R g+rwx /opt/ibm/wlp/output/defaultServer
+ find /opt/ibm/wlp -type d -perm -g=x -print0
+ xargs -0 -r chmod -R g+rwx
 ---> c59b138f838d
Removing intermediate container 84c56cdf6296
Successfully built c59b138f838d
============================================================
Tag Liberty Docker image for plantsbywebsphereee6
============================================================

Login Succeeded
============================================================
Push Liberty Docker image for plantsbywebsphereee6
============================================================

The push refers to a repository [docker-registry.default.svc:5000/lab5/plantsbywebsphereee6]
b9606a2617bd: Pushed
b8186b4ed4ff: Pushed
68ce2e9bdedb: Pushed
09af79cc3edd: Pushed
7c98abc96bed: Pushed
be36d206af93: Mounted from lab4/simpleapp
ba04059ad9a3: Mounted from lab4/simpleapp
71532d3a56e4: Mounted from lab4/simpleapp
790bcf471d32: Mounted from lab4/simpleapp
fe274995fb89: Mounted from lab4/simpleapp
9649117d0875: Mounted from lab4/simpleapp
9e19e22c9a42: Mounted from lab4/simpleapp
e9417d2583e6: Mounted from lab4/simpleapp
481324a7ba6d: Mounted from lab4/simpleapp
26429bebe019: Mounted from lab4/simpleapp
latest: digest: sha256:d47a474c28e2ebc96be205fdd2f745a2a94d6f2fd6adcf105f53b48c1a5e02eb size:
3465
[ibmdemo@icp4a appmodern]$
```

__9. Type `cat 02-createSecret.sh` to review the file

The script creates a Kubernetes secret that contains the connection information to the PlantsByWebSphere DB2 database. The Kubernetes secret is encoded when stored.

```
[ibmdemo@icp4a appmodern]$ cat 02-createSecret.sh
#!/bin/bash
#
# IBM Cloud Pak for Applications  - Proof of Technology
#
# Purpose: Create secrets for PBW database access


PROJECT=lab5

echo ============================================================
echo Create secrets for PBW database access
echo ============================================================
echo

kubectl -n $PROJECT  delete secret db2-secret > /dev/null 2>&1

# clear test used below for illustration
# encoded values employed typically

# Db2 secret
kubectl -n $PROJECT  \
    create secret generic db2-secret \
    --from-literal=JDBC_ID=db2inst1 \
    --from-literal=JDBC_PASSWORD=db2inst1 \
    --from-literal=JDBC_HOST=192.168.142.130 \
    --from-literal=JDBC_PORT=50000 \
    --from-literal=JDBC_DB=PBW

[ibmdemo@icp4a appmodern]$
```

__10. Type `./02-createSecret.sh` to run a script to specify Db2 credentials so that the application can connect to the Db2 back end

```
[ibmdemo@icp4a appmodern]$ ./02-createSecret.sh
 ============================================================
 Create secrets for PBW database access
 ============================================================

 secret/db2 created
 [ibmdemo@icp4a appmodern]$
```

__11. **The 03-createOperatorArtifacts.sh** script has been created to consolidate the creation of the 3 custom resource definition (CRD's) for the v0.3.0 Open Liberty operator, the required ServiceAccount, Role, and RoleBinding Kubernetes resources for the Liberty Operator.

_a. Enter the command: `gedit 03-createOperatorArtifacts.sh` to edit the file. Ensure the three lines shown below are **UNCOMMENTED**.

_b. Remove the # from the three lines if they are commented out (This will uncomment them)

_c. Save and close the file

```
# Uncomment the lines below to create the CRD's if lab 4 is not completed

#echo ============================================================
#echo " Create the custom resource definitions (CRDs) "
#echo ============================================================
#oc apply -f operator/application/application-crd.yaml
#oc apply -f operator/application/traces-crd.yaml
#oc apply -f operator/application/dumps-crd.yaml

echo ============================================================
```

remove # from beginning of line

remove # from beginning of line

remove # from beginning of line

These commands create the Custom Resource Definitions (CRDs) for the Open Liberty Operator which only needs to be performed once in a K8s/RHOCP cluster

We modified the CR and security files so that the names and labels specify "*plantsbywebsphereee6-operator"* in order to create artifacts specific to this deployment

Using the following commands, you can review the CRD files which define all the operator required resources:

```
cat operator/application/application-crd.yaml
cat operator/application/dumps-crd.yaml
cat operator/application/traces-crd.yaml
cat operator/deploy/role_binding.yaml
cat operator/deploy/role.yaml
cat operator/deploy/service_account.yaml
```

__12. Review the **application-cr** used for this lab by typing

```
cat operator/application/application-cr.yaml
```

```
[ ibmdemo@icp4a appmodern]$ cat operator/application/application-cr.yaml
# Revised Application CR for Open Liberty Operator v0.3
apiVersion: openliberty.io/v1beta1
kind: OpenLibertyApplication
metadata:
  name: plantsbywebsphereee6
spec:
  replicas: 1
  applicationImage: docker-registry.default.svc:5000/lab5/plantsbywebsphereee6
  tag: latest
# Add readiness and liveness probes
  ports:
  readinessProbe:
      httpGet:
        path: /
        port: 9080
      initialDelaySeconds: 3
      periodSeconds: 5
  livenessProbe:
    httpGet:
      path: /
      port: 9080
    initialDelaySeconds: 40
    periodSeconds: 10
# Expose external route
  expose: true
#Environment entries from db2-secret to access PBW
  envFrom:
   - secretRef:
       name: db2-secret
[ibmdemo@icp4a appmodern]$
```

- The file creates an `OpenLibertyApplication` instance named `plantsbywebsphereee6.`
- It used the docker image in the RHOSCP registry, specified by the `applicationImage.`
- It created a `readinessProbe` and a `livenessProbe` to monitor the application.
- A `route` is created to expose the application for external access
- `environmental variables` (`envFrom`) are imported for use by the application from a `secret(db2-secret)` (which contains database information and credentials)

__13. Create the OpenLibertyApplication artifacts required for the operator by typing:

```
./03-createOperatorArtifacts.sh
```

```
[ibmdemo@icp4a appmodern]$ ./03-createOperatorArtifacts.sh
============================================================
create Liberty operator ServiceAccount, Role, and RoleBinding
============================================================
serviceaccount/plantsbywebsphereee6-operator created
role.rbac.authorization.k8s.io/plantsbywebsphereee6-operator created
rolebinding.rbac.authorization.k8s.io/plantsbywebsphereee6-operator created
============================================================
deploy Liberty operator pod
============================================================
deployment.apps/plantsbywebsphereee6-operator created
============================================
Run command "oc get pods "
wait until the plantsbywebsphereee6 operator pod is ready
before running next script
============================================================
[ibmdemo@icp4a appmodern]$
```

__14. As indicated at the end of the prior script type `oc get pods` until the operator pod `Status` is `Running` and `Ready (1/1)`

```
[ibmdemo@icp4a appmodern]$ oc get pods
NAME                                        READY     STATUS              RESTARTS   AGE
plantsbywebsphereee6-operator-f985d9796-vzgrx   0/1       ContainerCreating   0          5s
[ibmdemo@icp4a appmodern]$ oc get pods
NAME                                        READY     STATUS              RESTARTS   AGE
plantsbywebsphereee6-operator-f985d9796-vzgrx   1/1       Running             0          8s

[ibmdemo@icp4a appmodern]$
```

__15. Once the operator pod is running type `./04-deployApplication.sh` which will deploy the application using the parameters specified in the `application-cr.yaml` (from step 12 above)

```
[ibmdemo@icp4a appmodern]$ ./04-deployApplication.sh

============================================================
deploy the application
============================================================
openlibertyapplication.openliberty.io/plantsbywebsphereee6 created
```

__16. Type `oc get pods` until both the `plantsbywebsphereee6` and the `plantsbywebsphereee6-operator` pods are running and ready

```
[ibmdemo@icp4a appmodern]$ oc get pods
NAME                                        READY   STATUS              RESTARTS   AGE
plantsbywebsphereee6-c8b46b768-btt4r        0/1     ContainerCreating   0          3s
plantsbywebsphereee6-operator-f985d9796-5p5p8   1/1     Running             0          46s
[ibmdemo@icp4a appmodern]$ oc get pods
NAME                                        READY   STATUS    RESTARTS   AGE
plantsbywebsphereee6-c8b46b768-btt4r        0/1     Running   0          7s
plantsbywebsphereee6-operator-f985d9796-5p5p8   1/1     Running   0          50s
[[ibmdemo@icp4a appmodern]$ oc get pods
NAME                                        READY   STATUS    RESTARTS   AGE
plantsbywebsphereee6-c8b46b768-btt4r        1/1     Running   0          16s
plantsbywebsphereee6-operator-f985d9796-5p5p8   1/1     Running   0          59s
[ibmdemo@icp4a appmodern]$
```

__17. Type `oc get all` and note that Open Liberty operator has created the following resources:

- A `ClusterIP` service for the application
- A `route` for external access to the application
- A `pod`
- A `replicaset`
- A `deployment`

```
 [ibmdemo@icp4a appmodern]$ oc get all
NAME                                                READY     STATUS    RESTARTS   AGE
pod/plantsbywebsphereee6-c8b46b768-btt4r            1/1       Running   0          12m
pod/plantsbywebsphereee6-operator-f985d9796-5p5p8   1/1       Running   0          13m

NAME                            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)
AGE
service/plantsbywebsphereee6    ClusterIP   172.30.215.234   <none>        9080/TCP
12m

NAME                                          DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
deployment.apps/plantsbywebsphereee6          1         1         1            1
12m
deployment.apps/plantsbywebsphereee6-operator   1         1         1            1
13m

NAME                                                    DESIRED   CURRENT   READY
AGE
replicaset.apps/plantsbywebsphereee6-c8b46b768          1         1         1
12m
replicaset.apps/plantsbywebsphereee6-operator-f985d9796   1         1         1
13m

NAME                                          HOST/PORT
PATH      SERVICES                PORT       TERMINATION   WILDCARD
route.route.openshift.io/plantsbywebsphereee6   plantsbywebsphereee6-
lab5.apps.icp4a.pot.com             plantsbywebsphereee6   9080-tcp
None

NAME
READY     REASON
clusteringress.networking.internal.knative.dev/route-17de13e9-fe3a-11e9-9829-
000c29ef9df2   True
[ibmdemo@icp4a appmodern]$
```

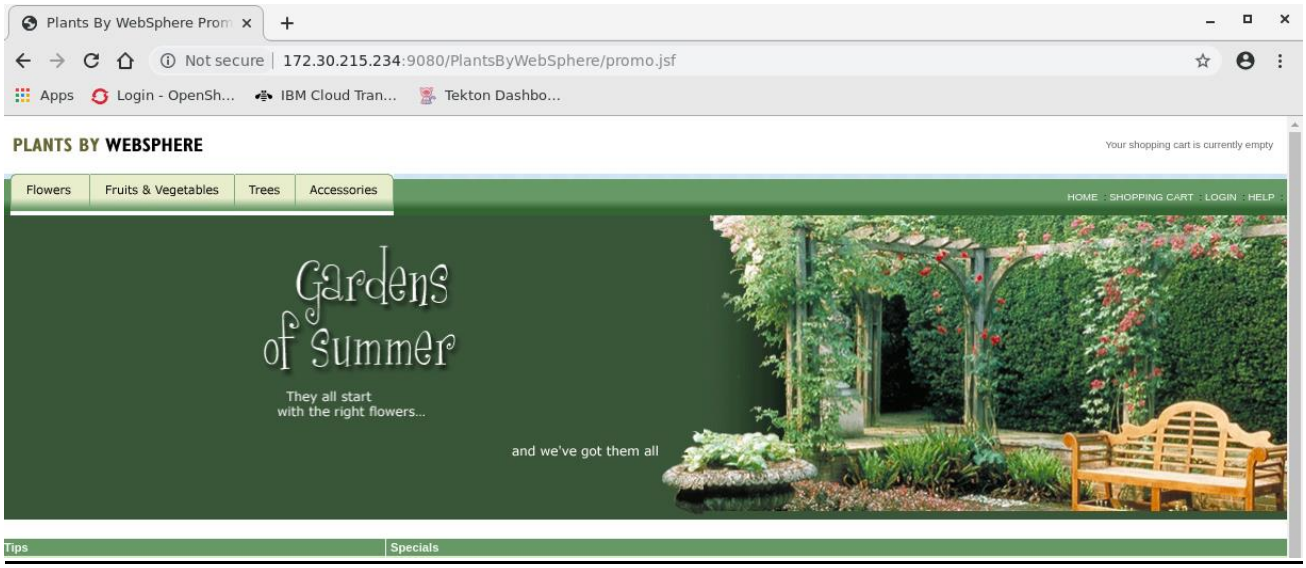
__1. You can access the application using the **ClusterIP**

_a. Open the **Chrome browser** and using the **ClusterIP** for your deployment construct the following URL http://<ClusterIP>:9080/PlantsByWebSphere

In the example above, this is http://172.30.20.9:9080/PlantsByWebSphere

Note: the ClusterIP in your environment will be different

You can use the ClusterIP to access the application because this is a single VM cluster

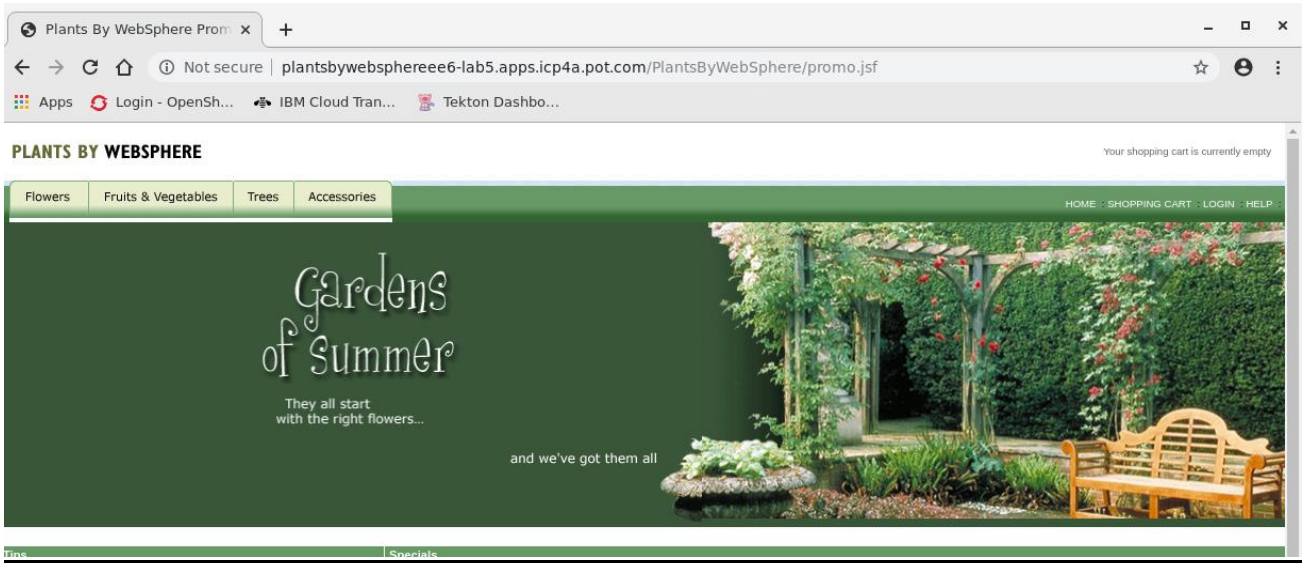


__18. The PlantsByWebSphere application can also be accessed via the **route** created

http://plantsbywebsphereee6-lab5.apps.icp4a.pot.com/PlantsByWebSphere

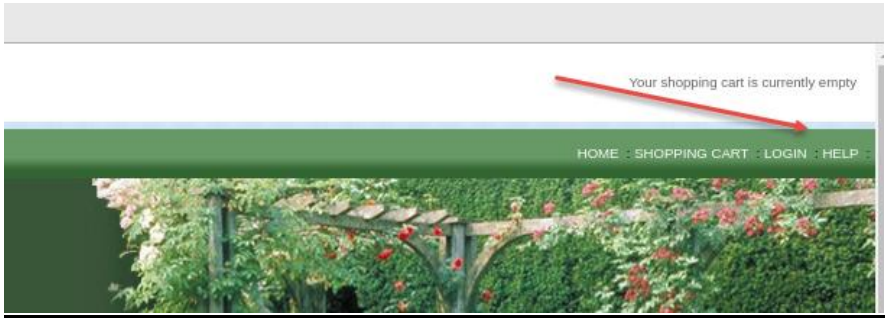
The route provides for external access to the application.

Note: the format of the route is the default, but can be modified

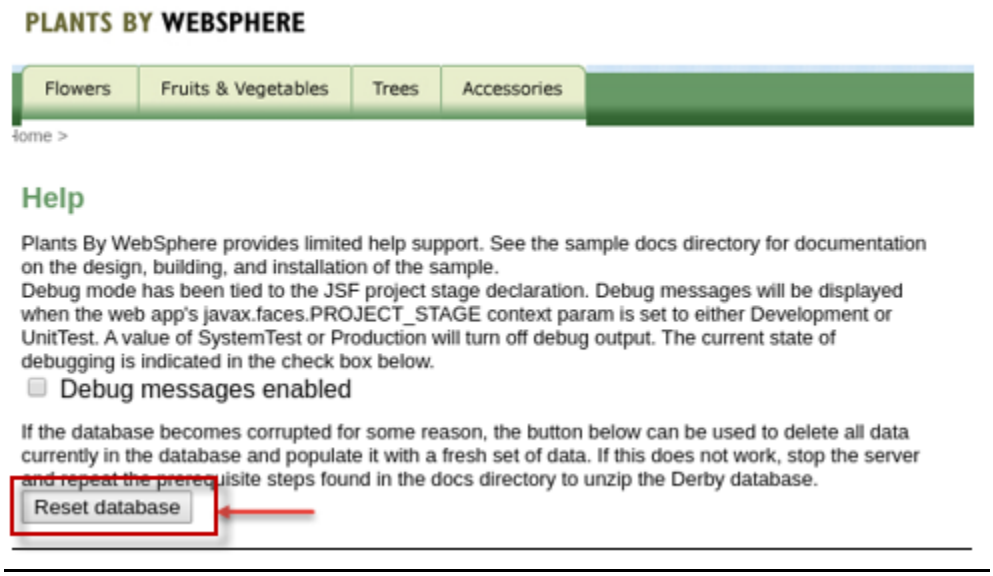


__19. Now populate the DB2 database to verify the application

_a. Click on `Help` in the upper right-hand corner of the Plants By WebSphere page



_b. Click on `Reset database` which will populate the database tables and is required before navigating through the application.



5.4 Conclusion

You have now seen how to run Transformation Advisor in the IBM Cloud Pak for Applications to analyze existing Java applications running on tWAS and assess the effort needed to migrate that application to Liberty.
Additionally, you used the Transformation Advisor generated deployment artifacts to deploy the application in Liberty on RHOCP.

End of Lab 05: Application Modernization using IBM Transformation Advisor

Appendix: SkyTap Tips for labs

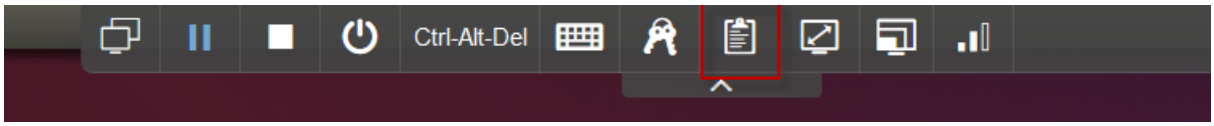
5.5 How to use Copy / Paste between local desktop and Skytap VM

Using copy / Paste capabilities between the lab document (PDF) on your local workstation to the VM is a good approach to more efficiently work through a lab, while reducing the typing errors that often occur when manually entering data.

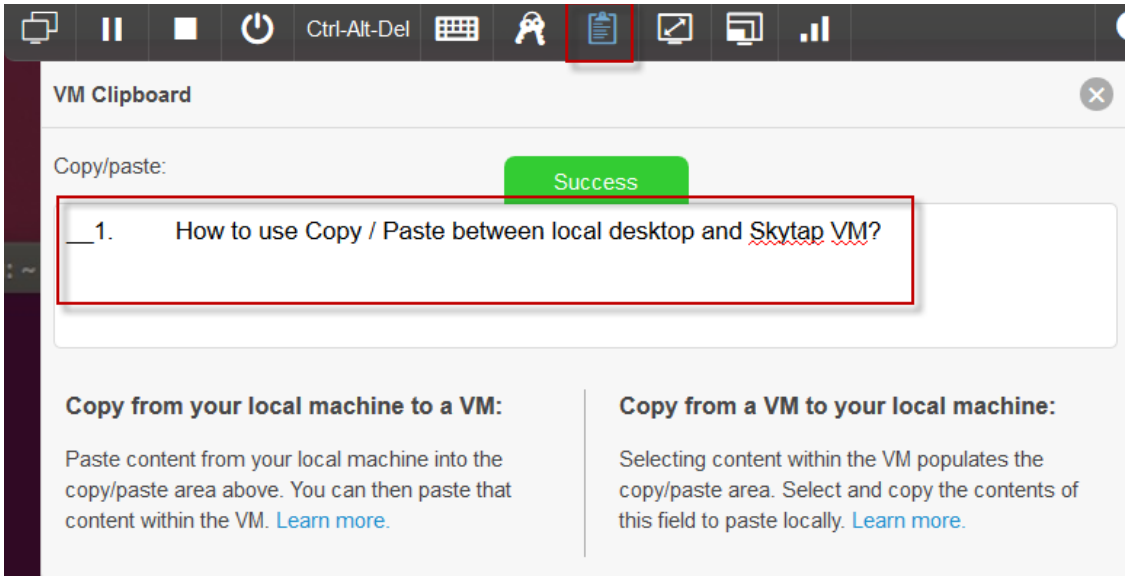
__1. In SkyTap, you will find that any text copied to the clipboard on your local workstation is not available to be pasted into the VM on SkyTap. So how can you easily accomplish this?

__a. First copy the text you intend to paste, from the lab document, to the clipboard on your local workstation, as you always have (CTRL-C)

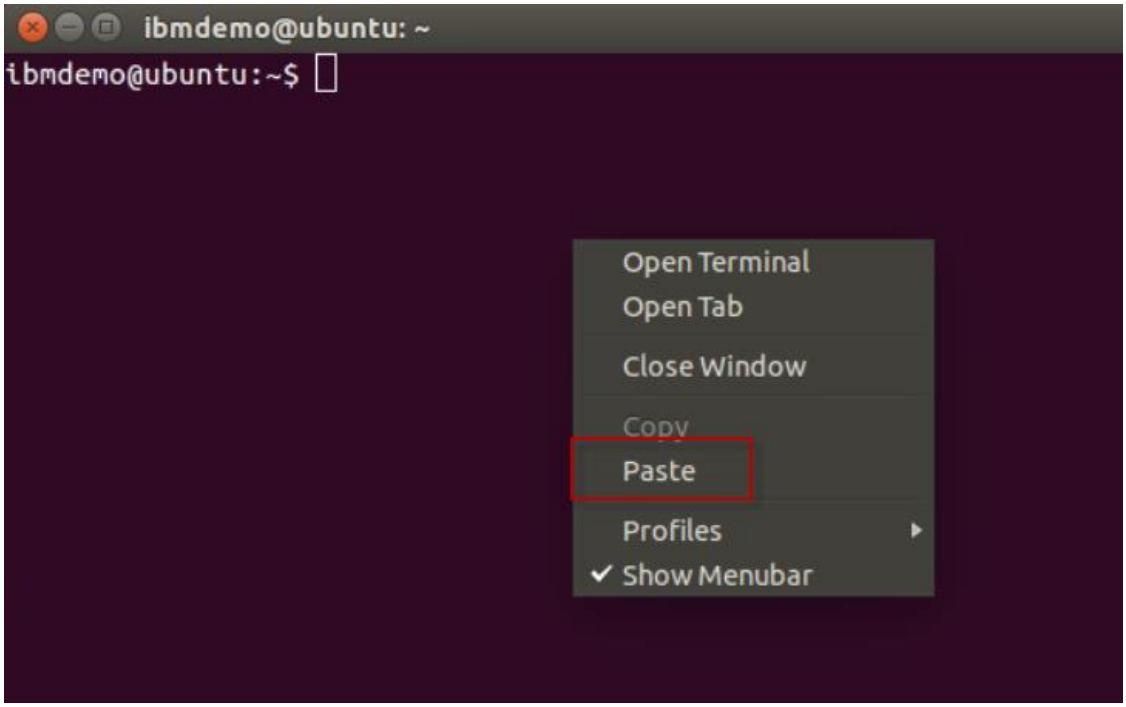
__b. Return to the SkyTap environment and click on the Clipboard at the top of the SkyTap session window.



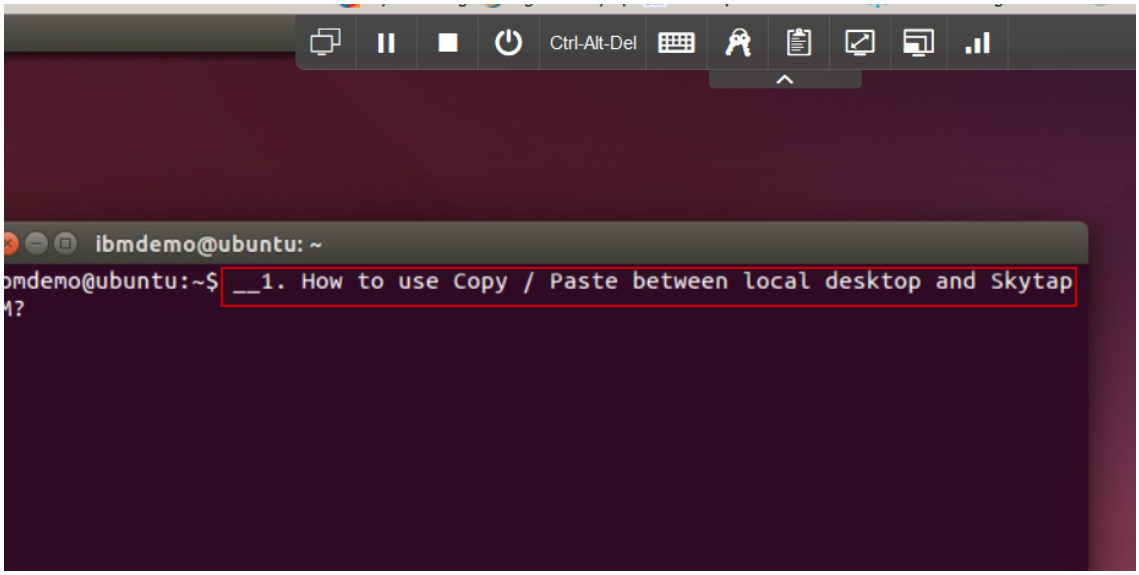
__c. Use **CTRL-V** to paste the content into the Copy/paste VM clipboard. Or use the **paste** menu item that is available in the dialog, when you right mouse click in the clipboard text area.



__d. Once the text is pasted, just navigate away to the VM window where you want to paste the content. Then, use **CTRL-C**, or right mouse click & us the **paste menu item** to paste the content.



__e. The text is pasted into the VM



Note: The very first time you do this, if the text does not paste, you may have to paste the contents into the Skytap clipboard twice. This is a known Skytap issue. It only happens on the 1st attempt to copy / paste into Skytap.