# Exercise 1

oc login -u admin -p redhat https://master.lab.example.com

oc get route -n default

oc adm policy add-role-to-user system:registry developer

oc adm policy add-role-to-user system:image-builder developer

oc login -u developer -p redhat https://master.lab.example.com

export TOKEN=$(oc whoami -t)

docker login -u developer -p $TOKEN docker-registry-default.apps.lab.example.com

# Exercise 2

oc login -u developer -p redhat https://master.lab.example.com

oc new-project apollo

oc new-app --name lunar --build-env npm\_config\_registry=http://services.lab.example.com:8081/nexus/content/groups/nodejs http://services.lab.example.com/nodejs-helloworld

oc logs -f bc/lunar

git clone http://services.lab.example.com/nodejs-helloworld

python -m json.tool nodejs-helloworld/package.json

vi ~/nodejs-helloworld/package.json

fix the error

cd nodejs-helloworld

git commit -a -m 'Fixed JSON syntax'

git push

oc start-build --follow bc/lunar

oc get pod

oc expose svc/lunar --hostname lunar.apps.lab.example.com

curl <http://lunar.apps.lab.example.com>

# Exercise 3

Oc login -u developer – p redhat <https://master.lab.example.com>

Oc new-project jedi

**oc create configmap obionekenobi --from-literal APP\_MSG="Use the Force,Luke"**

**ssh root@master**

**oc login -u developer -p redhat https://master.lab.example.com**

**mkdir jedimessages**

**oc get configmap/obionekenobi -o json > ~/jedimessages/message.json**

**exit**

Exercise 4

**oc login -u developer -p redhat** [**https://master.lab.example.com**](https://master.lab.example.com)

**oc new-project Colorado**

oc policy add-role-to-group system:image-puller system:serviceaccounts:Texas

git clone [**http://services.lab.example.com/php-info**](http://services.lab.example.com/php-info)

**cd php-info**

**docker build -t php-info.**

**docker tag php-info** registry.lab.example.com:5000/php-info

**docker push** registry.lab.example.com:5000/php-info

**oc import-image php-info --confirm --from** registry.lab.example.com:5000/php-info **--insecure**

**oc get is**

**oc new-project texas**

**oc new-app --name dallas -i colorado/php-info**

**oc get pods**

**oc expose svc dallas --hostname dallas.apps.lab.example.com**

**tips**

**to see docker images – docker images**

**to delete docker images – docker rmi registry.lab.example.com/php-info**

**to see images – oc get is**

**to delete images – oc delete is php-info**

# Exercise 5

**lab apache-s2i setup**

**sudo yum install source-to-image**

**pwd to make sure you are in /home/student directory**

**s2i create httpd myhttp**

**tree -a myhttp**

**cp ~/DO288/labs/apache-s2i/Dockerfile ~/myhttp**

**cp -Rv ~/DO288/labs/apache-s2i/.s2i ~/myhttp/**

**rm -f ~/myhttp/.s2i/bin/save-artifacts**

**cp ~/DO288/labs/apache-s2i/training.repo ~/myhttp/training.repo**

**cd myhttp**

**docker build -t myhttp .**

**docker images**

**docker tag myhttp registry.lab.example.com:5000/myhttp**

**docker push registry.lab.example.com:5000/myhttp**

**oc login -u developer -p redhat https://master.lab.example.com**

**oc new-project apache-s2i**

**oc import-image myhttp --from registry.lab.example.com:5000/myhttp --confirm –insecure**

**oc get is**

**oc new-app --name myhelloworld myhttp~http://services.lab.example.com/html-helloworld**

**oc logs -f bc/hello**

**oc expose svc hello --hostname hello.apps.lab.example.com**

**curl** [**http://hello.apps.lab.example.com**](http://hello.apps.lab.example.com)

# Exercise 6

Git clone <http://services.lab.example.com/s2i-scripts>

Cd s2i-scripts

Vi .s2i/bin/assemble – to edit the assemble the file

Put the following commands in the file

cp -Rf /tmp/src/\*.html ./

DATE=`date "+%Y-%m-%d'"`

Replace text with “we have a lift off”

Git add .

**Git commit -m -a ‘assemble updated’**

**Git push**

**Oc whomai**

**Oc new-project gemini**

**oc new-app --name rocket httpd~http://services.lab.example.com/s2i-scripts**

**oc expose svc rocket --port 8080 --hostname rocket.apps.lab.example.com**

**curl http://rocket.apps.lab.example.com**

**curl http://rocket.apps.lab.example.com/info.html**

# Exercise 7

wget <http://content.example.com/courses/do288/ocp3.6/materials/labs/review-template.tgz>

wget zxvf review-template.tgz

cp ~/exercise7/review-template/todo-template.yaml ./todo-template.yaml

rm -rf review\*

vi todo-template.yaml

changes to make in the file

1)add missing parameters at the end of the file -

1. **Change name and displayname**
2. **At the end of the file add parameters for PASSWORD, CLEAN\_DATABASE, GIT\_URL**
3. **Look thru the file and replace with ${VAR\_NAME} for git url, hostname, npm proxy, password in 2 places**

**Oc new-project resistance-common**

**oc create -f ~/todo-template.yaml**

**oc new-project yoda**

oc new-app --name skywalker resistance-common/resistance-todo \

-p HOSTNAME=skywalker.apps.lab.example.com \

-p NPM\_PROXY=http://services.lab.example.com:8081/nexus/content/groups/nodejs \

-p PASSWORD=r2d2 \

-p CLEAN\_DATABASE=”true”

Oc logs -f bc/skywalker

Oc get pods

Once completed, you should see 2 running pods

To test the app from web - [**http://skywalker.apps.lab.example.com/todo/index.html**](http://skywalker.apps.lab.example.com/todo/index.html)

**To test app from oc – curl http://skywalker.apps.lab.example.com/todo/api/items-count**

# Exercise 8

Mkdir mars

Cd mars

Pwd

Git clone <https://services.lab.example.com/todo-frontend>

Cd todo-front

Vi Dockerfile

Optimize by making following changes

* Combine LABEL commands with \
* Combine run YUM commands with && \
* Combine run CHOWN and CHMOD commands with && \
* In the CHOWN replace NGINX:NGINX with 1001:0
* In the chmod add X permissions

**docker build -t mars-frontend .**

**docker images**

**docker tag mars-frontend registry.lab.example.com:5000/mars-frontend**

**docker push registry.lab.example.com:5000/mars-frontend**

**oc login -u developer -p redhat https://master.lab.example.com**

**oc new-project mars-common**

**oc import-image rover --from registry.lab.example.com:5000/mars-frontend --confirm –insecure**

**oc get is**

**oc new-project mars-frontend**

**oc new-app --name rover-frontend -i mars-common/rover**

**oc logs -f bc/rover-frontend**

**oc expose svc rover-frontend --hostname rover-frontend.apps.lab.example.com**

**curl** [**http://rover-frontend.apps.lab.example.com**](http://rover-frontend.apps.lab.example.com)

note: in the exam name of external registry is; registry.domainXX.example.com – instead of registry.lab.example.com

web console will master.domainXX.example.com

# Exercise 9

Git clone http://services.lab.example.com/build-app

Cd build-app

Vi hello.py (type print(‘hello world’) in the file and save

Git add .

Git commit -m “python file added”

Git push

Oc login -u developer -p redhat https:/master.lab.example.com

Oc new-project Saturn

Oc new-app –-name titan --build-env npm\_config\_registry=[**http://services.lab.example.com:8081/nexus/content/groups/nodejs**](http://services.lab.example.com:8081/nexus/content/groups/nodejs)

**http://services.lab.example.com/build-app**

oc set build-hook bc/titan –-post-commit –-command -- bash -c “python hello.py”

**oc describe bc/titan | grep Post**

**oc start-build bc/titan -F**

# Exercise 10

Oc new-project Jupiter

Oc new-app –-name europa --build-env npm\_config\_registry=[**http://services.lab.example.com:8081/nexus/content/groups/nodejs**](http://services.lab.example.com:8081/nexus/content/groups/nodejs)

[**http://services.lab.example.com/probes**](http://services.lab.example.com/probes)

oc logs -f bc/Europa

oc get pods

<https://master.lab.example.com> to get to console to set the probes in the deployment config, once changes done, a new deployment should get triggered, you can also see it from monitoring

# Exercise 11

Oc new-project pluto-common

**oc import-image dwarf --from registry.lab.example.com:5000/hello-world-nginx --confirm –insecure**

**oc get is**

**oc new-project pluto**

**oc new-app –-name charon -i pluto-common/dwarf**

oc get pods

oc expose svc/charon –hostname charon.apps.lab.example.com

curl <http://charon.apps.lab.example.com>

# Exercise 12

oc new-project Uranus

Oc new-app –-name oberon --build-env npm\_config\_registry=[**http://services.lab.example.com:8081/nexus/content/groups/nodejs**](http://services.lab.example.com:8081/nexus/content/groups/nodejs)

**http://services.lab.example.com/app-config**

**oc create configmap myappconf --from-literal APP\_MSG="Test Message"**

**oc create secret generic myappfilesec --from-file /home/student/myapp.sec**

**oc get secret/myappfilesec -o json**

**oc set env dc/oberon --from configmap/myappconf**

**oc set volume dc/oberon --add -t secret -m /opt/app-root/secure --name myappsec-vol --secret-name myappfilesec**

**oc rsh myapp-3-*wzdbh* env | grep APP\_MSG**

**oc expose svc oberon –hostname oberon.apps.lab.example.com**

**curl** [**http://oberon.apps.lab.example.com**](http://oberon.apps.lab.example.com)

# Exercise 13

1. oc login -u developer
2. oc new-project indigo
3. oc new-app --name purple --build-env npm\_config\_registry=http://services.lab.example.com:8081/nexus/content/groups/nodejs http://services.lab.example.com/app-config
4. oc logs -f bc/purple
5. oc get pods
6. oc expose svc purple --hostname purple.apps.lab.example.com
7. curl <http://purple.apps.lab.example.com>
8. ssh root@master.lab.example.com
9. oc login -u developer
10. oc project indigo
11. oc create configmap greeterconf --from-literal APP\_MSG=”Hello from the developers”
12. mkdir /root/workdir
13. oc export configmap greeterconf -o json > /root/workdir/greeterconf.json
14. oc set env dc/purple –-from configmap/greeterconf
15. curl <http://purple.apps.lab.example.com>
16. exit