INDEX

Name : Nanda kishora C R Class VII Year 2021

Expt. No.	Date	Title of the Experiments	Page No.	Date of Submission	Remarks
1.	12/11/21	Github	1-3	19/11/21	
2.	19/11/21	shell script -1	4-4	17/12/21	
3.	19/11/21	Shell Script -2	5-6	17/12/21	
4.	17/12/21	Tomcat application manager	7-9	24/12/21	
5.	24/12/21	Maven Project	10-11	07/01/22	
6.	7/01/21	Jenkins deployment for Docker	12-13	14/01/22	
7.	14/01/21	Ansible	14-18	21/01/22	

Date 12/11/2021

Experiment No. 1

① Conduct the assessment using Git & github handle local, working and remote directories.

Open your git bash terminal in your working local directory folder.

Create Repository :-

$git init // this initializes the library and assign the work of directory

$ git clone // maps the entire repository & features the significante of working directory

Synchronizing repositories :-

$git add remote origin <link> // extract only the http link from your created git hub repository.

Now for pulling the contents from remote repository perform

$git pull origin main/master // this enables the items of remote repo to local directory created.

Changes to local directory

Now create the text file in your local repository & check the status in your git bash

$git status // update the status of created file in your working directory with red highlight.

Now add the contents to stating area by typing

$git add <createdfile.txt>

After this commit the file by operation of commit

$git commit // this commit total commits corresponding to the created files

Creating the branches:

$git branch new_branch // this creates newer branch by adding the new one

$git new_branch checkout // take the working directory from master to new branch

$git status
$ git add. // this add total files on the local directory

Again checkout to master

$git checkout master

Edit the file in your local repo

It should not list the files of new branch

Check by typing

$git checkout new_branch

$ls

Date.....

Experiment No.....

Push your local directory files to remote repository

$git push origin master/main

All the files need to be popped in your remote github repository.

(2) write the shell script to deploy the application.

```
#!bin/bash
cd../
echo Stoping Batch Module....
appHome = `pwd`
pid = $(ps - Aef|grep `pwd`|grep -v grep | awk ' {print $2}')
kill -9 $pid
[echo = $(ps - A.]
echo Removing nohup. out file.....
if [-f bin/nohup.out]
then
rm -rf bin/nohup.out
fi
cd standalong/log/
timeNow = '(date --date='now' + %Y-%m-%d_%H')'
cd ${appHome}/bin
echo Restarting Application.....
nohup ./standalone.sh - bmanagement = 127.0.0.1 -b0.0.0.0 &
sleep 5
```

(3) Write a shell script to backup system logs on daily basis for School Management System.

```
SCRIPT_NAME = "LOG_BACKUP"
LOG_FILE = 'logBackupDaily.log'
function logger(){
    echo -e "\n[`date +"%F %T"`][${USER}@${HOSTNAME}|
        ${SCRIPT_NAME}]: $1" >> ${LOG_FILE}
}
function logInfo(){
    echo -e "\n[`date +"%F %T"`][INFO]: $1" >> ${LOG_FILE}
}
logger "=====Starting Log Backup Process====="
logInfo "=== Getting Dates in Required Format ==="
DATE = $(date + %d)

DATE_TO_TAR = $(date --date="1 day ago" + %F)
DATE_TO_TAR_2 = $(date --date="1 day ago" + %m-%d-%Y)
DATE_TO_TAR_AL = $(date --date="3 days ago" + %F)
DATE_TO_TAR_ALF = $(date --date="3 days ago" + %Y-%m_%d)
DATE_TO_RM_AL = $(date --date="30 days ago" + %Y_%m_%d)
DATE_TO_RM_SL = $(date --date="30 days ago" + %F)

if [[ $DATE == '01' ]]; then
    cd /opt/stguser/HOUSE_KEEPING/LOGS/; logInfo "`pwd`"
    > logBackupDaily.log
    > clearDiskSpace.log
fi
logInfo "=== NULLIFYING bin nohup.out FILES FOR ALL APPLICATION ==="
```

Date.....

Experiment No.....

```
cd /home/School_Management_System/SMS_LOGS/; logInfo " 'pwd' "
ls -l sms.log.${DATE_TO_TAR}-* |awk '{print $9}' | xargs tar -
    cvzf sms_${DATE_TO_TAR}.tgz | xargs rm -rvf
mv -v sms_${DATE_TO_TAR}.tgz /backup/SMS_LOGS/
```

Date 17/12/21

Experiment No. 4

(4) Enabling Tomcat application manager from client IP through continuous deployment.

Pre-requisites.

EC2 instances with Java v1.8.x

Install Apache Tomcat

1. Download tomcat packages from https://tomcat.apache.org/download-80.cgi onto /opt on EC2 instance.

Note: Make sure you change <version> with the tomcat version which you downloaded.

#tomcat directory creation.

$cd /opt

$ wget https://mirrors.fibergrid.in/apache/tomcat/tomcat-8/v8.5.35/bin/apache-tomcat-8.5.35.tar.gz

$ tar -xvzf /opt/apache-tomcat-<version>.tar.gz

2. give executing permissions to startup.sh & shutdown.sh which are under bin.

3. $chmod +x /opt/apache-tomcat-<version>/bin/startup.sh

$chmod +x /opt/apache-tomcat-<version>/bin/shutdown.sh

4. create link files for tomcat startup.sh and shutdown.sh

5. $ln -s /opt/apache-tomcat-<version>/bin/startup.sh /urs/local/bin tomcatup

Date.....

Experiment No.....

6. $ln -s /opt/apache-tomcat-<version>/bin/shutdown.sh /urs/local/bin/tomcatdown.

Check point:

access tomcat application from browser on port 8080
http://<public_IP>:8080
using unique ports for each application is a best practice in an environment. But tomcat & Jenkins runs on port number 8080. hence lets change tomcat port number to 8090. change port number in conf/server.xml file under tomcat home

$cd /opt/apache-tomcat-<version>/conf
# update port number in the "Connector port" field in server.xml.
# restart tomcat after configuration update
$ tomcatdown
$ tomcatup

Check point:

Access tomcat application from browser on port 8090
http://<Public_IP>:8090

1. now application is accessible on port 8090. but tomcat application doesn't allow to login from browser. changing a default parameter. in context.xml does address this issue.

2. #search for context.xml
$find / -name context.xml

Date.....

Experiment No.....

3. above command give 3 context.xml files. comment() Value ClassName field on files which are under webapps directory. After that restart tomcat services to effect these changes. At the time of writing this lecture below 2 files are updated

4. $/opt/tomcat/webapps/host-manager/META-INF/context.xml
5. $/opt/tomcat/webapps/manager/META-INF/context.xml
6.
7. # Restart tomcat Services
8. $ tomcatdown
   $ tomcatup

9. Update users information in the tomcat-users.xml file goto tomcat home directory & Add below users to conf/tomcat-users.xml file

10. <role rolename = "manager-gui" />
11. <role rolename = "manager-script" />
12. <role rolename = "manager-jmx" />
13. <role rolename = "manager-status" />
14. <user username = "admin" password = "admin" roles = "manger-gui, manager-script, manager-jmx, manager-status" />
15. <user username = "deployer" password = "deployer" roles = "manager script" />
16. Restart services & try to login to tomcat application from the browser. This time it should be successful.

⑤ Continuous boild integration & deployment of client-server method through maven Project.

Prerequisite : Jenkins.war, java 8, mobaxtrem client app, aws, Tomcat server setup by ec2, gi thub

Setting up a Maven Project.

1. Open Jenkins & create new item with maven option select ok.
2. From your github repo copy the https url & paste it in Jenkins job created by selecting git
3. Under build you will be directed with pom.xml & provide clean install packages.
4. Go to build now & console will be success.
5. Now open your AWS & copy the setup servers public IP to mobarterm client
6. Remote hoste: Public IP (AWS) and username : ec2-user
   Advanced settings: select the key generated.
7. After the ec2 instance is enabled at your terminal, switch to root ($ sudo su -)
   $cd /opt
   $ cd Apache folder
   $ ls
   $ cd bin
   $ ./startup.sh
8. Access the browser with IP publicip :8080 → this leads to tomcat server page. Only if you have configured the server.
9. You can change the port number of tomcat server to 8090 by selecting
   $ cd conf
   $ vi server.xml → here change the 8080 number to 8090

Date.....

Experiment No.....

10. Goto to Jenkins & your project configuration. in post steps no war file will be found to enable it goto to step 11
11. Now go back to your jenkins & install deploy to container plugin.
12. Goto to manager Jenkins & mange credentials, now add credentials, credentials can be found at Client. Type the command as.
    $ cat tomcat-users.xml
13. Now in Jenkins credentials add the contents both password & username as deployer & description as tomcat_credentials even id is same as description.
14. Go to step-10 & repeat the procedure to find war content.
15. Now war is found & Click on it & input the following
    **/*.war
16. In add container select tomcat 8. & save the content
17. Now go to client app & type in this.
    $cd webapp (under apache directory)
    $ls
    $ls -ltr
    $Now you will be able to vie the new file.

Date 07/01/2022

Experiment No. 6

⑥ Enabling Docker image on client terminal through Jenkins deployment.

1. Launch an EC2 instance for Docker host

2. Install docker on EC2 instance & start services.
   $ yum install docker
   $ service docker start

3. create a new user for Docker management & add him to Docker (defaut) group.
   $ useradd dockeradmin
   $ passwd dockeradmin
   $ usermod -aG docker dockeradmin

4. write a Docker file under /opt/docker
   $ mkdir /opt/docker

```
### vi Dockerfile
#Pull base image
From tomcat:8-jre8

#Maintainer
MAINTAINER "ISE"

#Copy war file on to container
Copy./webapp.war/usr/local/tomcat/webapps
```

5. Login to Jenkins console & add Docker server to execute commands from Jenkins.
   Manage Jenkins --> Configure system --> Publish over SSH --> add Docker server & credentials.

6. Create Jenkins job

A) Source Code Management
Repository : url from github.
Branches to build : */master

B) Build Root POM : pom.xml
Goals & options : clean install packages

C) send files or execute commands over SSH Name : docker_host
Source files : webapp/target/*.war Remove prefix : webapp/target/ Remote directory : // opt // docker
Exec command[s] :
docker stop ronshaw_demo ;
docker rm -f ronshaw_demo ;
docker image rm -f valaxy_demo ;
cd/opt/docker ;
docker build -t valaxy-demo .

D) Send files or execute commands over SSH
Name : docker_host
Exec command : docker run -d --name valaxy_demo -p 8080:8080 ronshaw_demo

7. Login to Docker host & Check images & Containers.

8. Execute Jenkins job

9. Check images & Containers again on Docker host. This time an image & Container get creates through Jenkins job

10. Access web application from browser which is running on Container <docker_host_Public_IP>:8080

Date 14/01/2022

Experiment No. 7

(7) Enabling Ansible server IP for duplication with client for webapp deployment.

First setup is launching of two ec2 instances one containing of ansible host & control server respectively

Note:- for security group you need to enable only ssh 22 & launch 2 instances.

open client mobaxterm SSH
Copy public IP of host & control server ansible from AWS that have recently launched.

In control server follow this procedure.

```
$ sudo su -
$ yum update
```

In ansible client

```
$ sudo su -
$ yum update
```

In control server

```
$ yum install ansible
$ rpm -Uvh https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
$ yum install ansible
$ ansible --version

$ useradd ansadmin
$ passwd ansadmin
```

Date.....

Experiment No.....

$ Yum install ansible

$ rpm -Uvh https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

$ yum install ansible

$ ansible -version

In ansible client

$ useradd ansadmin

$ passwd ansadmin

In control server

$ visudo

Type in the following instruction atlast of the editor

##Read drop-in files from /etc/sudoers.d (the # here does not mean a comment)

#includedir /etc/sudoers.d

ec2-user ALL=(ALL) NOPASSWD:ALL

ansadmin ALL=(ALL) NOPASSWD:ALL

In ansible client

$ visudo

Type in the following instruction atlast of the editor.

Read drop-in files from /etc/sudoers.d (the # here does not mean a comment)

included /etc/sudoers.d

ec2-user ALL=(ALL) NOPASSWD:ALL

ansadmin ALL=(ALL) NOPASSWD:ALL

Date.....

Experiment No.....

In Control Server.

```
$cd /etc/ssh
$ ls
$ vi sshd_config
```

Type the following by enabling the password authentication has yes

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
# PermitEmptyPassword no
# PasswordAuthentication no
$ service sshd restart
```

In ansible client

```
$visudo
$ cd /etc/ssh
$ ls
$ vi sshd_config
```

Type in the following by enabling the password authentication has yes

```
# To disable tunneled clear text password, change to no here!
PasswordAuthentication yes
# PermitEmptyPassword no
# PasswordAuthentication no
$ service sshd restart
```

Now right click on the ansible control server tab in your client click on duplicate tab. Tab will open a duplicate tab & features the replication of ansible server

Date.....

Experiment No.....

In Duplicate Control server tab (here you need to generate ssh)

$ sudo su anadmin

$ ssh-keygen

Press enter don't type password

$ ls -la

There should be file by name .ssh or else error

Go to ansible client

$ ifconfig

This gives you the private ip of your ansible client & copy this IP

In Duplicate Control server tab.

$ cd .ssh

$ ls

$ ssh-copy-id <privake ip of ansible client>

This is how you deploy several clients into server control area dynamically

To come out of client ip.

$ ssh <client private ip>

$ exit

Connection will be closed

To execute & deploy features of ansible configuration in devops strategy. in duplicate server follow the procedure.

$ sudo vi /etc/ansible / hosts

Delete the content & type the following this way like given below.

[all_hosts]

172.31.24.232

~

~

~

```
$ ansible all -m ping
$ ls
$ cd..
$ ls
$ pwd
$ cat > hello.html
<h1> hello welcome... welcome to devops </h1>
$ ls
```

In ansible client

```
$ cd /home/ansadmin
$ ls
$ pwd
$ ls
```

Go back to your control server

```
$ ansible all -m copy "src=/home/ansadmin/hellow.html
                       dest=/home/ansadmin"
```

In your client

```
$ ls -l
```