ALL SCENARIOS ARE BASED ON REAL TIME AUTOMATION WORK

❄❄❄❄❄❄❄❄❄❄❄❄ ShineInCareer Note ❄❄❄❄❄❄❄❄❄❄❄❄

Thank you for joining my Tenthlivesession ☺ , looking forward to meet you in eleventhlivesession (29-01-2022) Coming Saturday, Topic is yet to be decided☺ ☺

All Sessions are taken on every Saturday between 4-6 PM unless there is a festival on that day ☺

❄❄❄❄❄

TOPIC 1 – SHELL SCRIPTING COMMANDS

Prerequisites

- Create a file using vim test.sh
- Add the below dummy data

******

hi
hello
This is praveen here
elevenlivesession
Welcome
Manager

****

- Use Centos flavours of linux/Mac and change the ‘ to ' in below commands if you get any error
- NOTE : Few Real time Automations are just for your understanding whatever I have showed in demo please try those in below url https://www.onlinegdb.com/online_bash_shell

1) sed -n '1,3'p test.sh ------> prints 1 to 3 lines
2) awk '{print}' test.sh -------> prints the whole file
3) awk ’/Manager/{print}’ test.sh prints lines with "manager" word
4) awk -F'[.]' '/^version/{print $1"."$2"."$3"."$4+1}’ test.sh
version=20.1.2.3
5) sed -i ‘s/tenth/eleven/g’ test.sh
6) myVar=`echo $hello | sed 's/=/ /g' | awk '{print $2}'`
7) date1=`echo $date| sed 's/-//g’`
8) ls -ltr | head -10

TOPIC 2

AUTOMATION 1 - IF ELSE BLOCK WITH JFROG REST API

CICD=true
WORKSPACE=/apps/opt/users/
JOB_BASE_NAME=Test_demo

```
if [[ $CICD == true ]]
Then
 echo "CI/CD pipe line check"
file="${WORKSPACE}/html/basic_report.html"
REPORTNAME=${JOB_BASE_NAME}_${BUILD_NUMBER}. Test_demo_10
 echo "CICD Check starting"
if [ -f "$file" ]; then
        echo "testReport file found sending to artifactory".    -T--→ Target
         curl -H X-JFrog-Art-Api:Token -T $file
https://oneartifactorycloud/artifactory/CICD/Reports/$REPORTNAME.html
        curl -H X-JFrog-Art-Api:Token -T $file
https://oneartifactorycloud/artifactory/CICD/Reports/$REPORTNAME.html
   else
        echo "testReport not found"
   fi
fi
```

TOPIC 3

AUTOMATION 2 – FOR LOOP

```
vim  forloop.sh
for line in $(cat test.sh)
do
echo $line
done
sh forloop.sh
```

AUTOMATION 2.1 – COMPLEX FOR LOOP WITH JIRA REST API

```
for ip in $(cat ip.txt)    10.118,10.119
do
        curl -u GITLAB:Token -X PUT --data
'{"update":{"labels":[{"add":""""$TEAMNAME-$version""""}]}}' --header "Content-Type:
application/json" https://jira.com/rest/api/2/issue/2341          curl -u GITLAB:Token -X PUT
--data '{"update":{"comment":[{"add":{"body":""""$version""""}}]}}' --header "Content-Type:
application/json" https://jira. com/rest/api/2/issue/$line
echo "Name read from file - $line"
done
```

TOPIC 4

AUTOMATION 3 – ARRAY DECLARATION

Vim arrayloop.sh
Add the below data

```
array=(helloservice hiservice nameservice managerservice teamservice)
for line in "${array[@]}"
do
echo $line
done
```

Testfilename - sh arrayloop.sh

TOPIC 5

AUTOMATION 4 – ARCHIVE THE DATA WITH FIND/MTIME/TAR/NAME COMMAND

```
cd /apps/logs/backup.txt
 find . -type f -mtime +7-exec mv '{}' /apps/logs/Log_Backup \;
cd /apps/logs/Log_Backup
find /apps/logs/Log_Backup -type f -name '*log*' > include-file
tar -cvf $(hostname)_$(date +%Y%m%d%H%M%S).tar.gz -T include-file
```

TOPIC 6

AUTOMATION 5 – PLAYING WITH DATES

```
ONE=14
dataset_date=`date`
Fourteendaysold=`date -d "$dataset_date - $ONE days" +%F`
echo $Fourteendaysold
TWO=7
dataset_date=`date`
sevendaysold=`date -d "$dataset_date - $TWO days" +%F`
echo $sevendaysold
date1=`echo $sevendaysold | sed 's/-//g’`
date2=`echo $Fourteendaysold | sed 's/-//g'`
echo $date1 with $date2
```

TOPIC 7

AUTOMATION 6 – WHILE LOOP

```
#!/bin/sh
a=0
while [ $a -lt 10 ]
do
  echo $a
  a=`expr $a + 1`
done
```

TOPIC 8

AUTOMATION 7 – SENDEMAIL TO TEAMS

```
/usr/sbin/sendmail -i -t <<
Subject: $1 server process status
```

From: from mail id
To: to mail id
Hi Team,
Please check for $1 service in TEST server which has $3
process running with below list of KIT IDs
$2
Regards,
ShineInCareer
MESSAGE_END

TOPIC 9

AUTOMATION 8 – SPECIAL VARIABLES IN LINUX

```
Vim hello.sh
sh hello.sh linkedin place good
echo $#
echo $?
echo $0
echo  $1 service in TEST server which has $3  $2
echo "With *:"
for arg in "$*"; do echo "<$arg>"; done
echo
echo "With @:"
for arg in "$@"; do echo "<$arg>"; done

With *:
<linkedin place good>

With @:
<linkedin>
<place>
<good>
```

TOPIC 10

AUTOMATION 9 – CREATE AN HTML FILE

```
cat > html <<'EOF'
<html>
<body>
<p><strong>MODULE_VERSION - {{module_version}}</strong></p>
<table border="1" cellspacing="0" cellpadding="2.5" valign="top"
```

width="100%" align="justify"
style="width: 100%; max-width: 1200px; background-color: #ffffff">
<tr>
<th>Key</th>
</tr>
EOF
Paste test.sh test2.txt tesxt100.txt | while read key dog zebra; do
cat >> jira.html <<EOF
<tr>
<td>$ key </td>
</tr>
EOF
done

TOPIC 11
AUTOMATION 10 – SONAR/GITLAB REST API/ JQ COMMANDS

Prerequisite – Install JQ
For LINUX - yum install jq
For MAC – brew install jq

curl -u "GITLAB:$gitlab"
"https://sonar/api/qualitygates/project_status?projectKey=${sonarProjectKey}&branch=${PIPELINE_GIT_BRANCH}" >> status.json

curl --request POST --header "PRIVATE-TOKEN: $TOKEN"
"https://gitlab/api/v4/projects/$PROJECT_ID/protected_branches?name=$MS_TARGET_BRANCH&push_access_level=40&merge_access_level=40" >> protectedbranches.txt

Add the below json data to file vim json.sh

```
{
   "employee": {
      "name":      "sonoo",
      "ID":     56000,
      "Status":    true
   }
}
```

Run commands
cat status.json| jq '.employee'
cat status.json| jq '.employee.name'