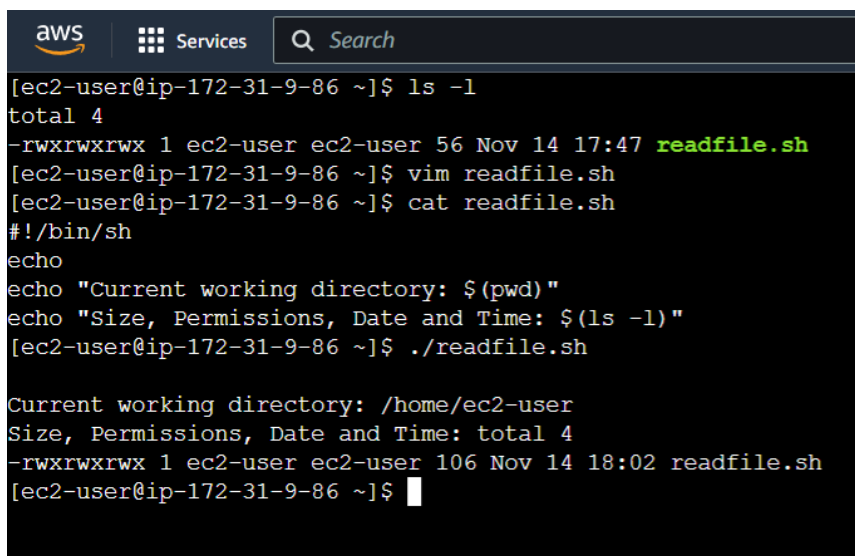


Assignment Sheet

Assignment 1: - How to upload HTML web pages on Apache2 web server in EC-2 Instance?
Please justify with step-by-step answers.

1. Launch EC2 instance.
2. Take SSH access of your instance and update machine using command
 - `sudo yum update -y` and
 - `sudo yum upgrade -y`
3. Install Apache web server.
 - `sudo yum install -y httpd`
4. Start webserver with command.
 - `sudo systemctl start httpd`
5. To configure web server such a way that it will start automatically with each system boot.
 - `sudo systemctl enable httpd`
6. Now open apache test page using public IP address of instance.
7. Upload required HTML web pages on path `/var/www/html/`.

Assignment 2: Create `readfile.sh` in which you can read the information of PWD like size, permission, date time etc.



```
aws Services Search
[ec2-user@ip-172-31-9-86 ~]$ ls -l
total 4
-rwxrwxrwx 1 ec2-user ec2-user 56 Nov 14 17:47 readfile.sh
[ec2-user@ip-172-31-9-86 ~]$ vim readfile.sh
[ec2-user@ip-172-31-9-86 ~]$ cat readfile.sh
#!/bin/sh
echo
echo "Current working directory: $(pwd)"
echo "Size, Permissions, Date and Time: $(ls -l)"
[ec2-user@ip-172-31-9-86 ~]$ ./readfile.sh

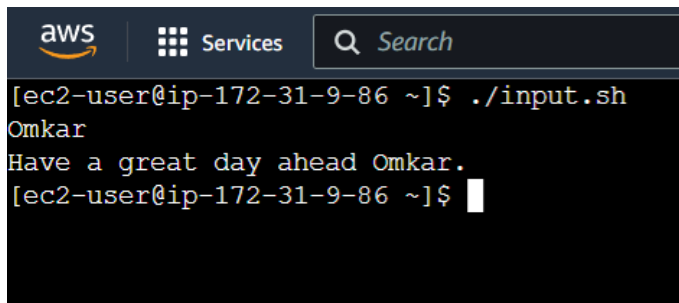
Current working directory: /home/ec2-user
Size, Permissions, Date and Time: total 4
-rwxrwxrwx 1 ec2-user ec2-user 106 Nov 14 18:02 readfile.sh
[ec2-user@ip-172-31-9-86 ~]$
```

Assignment 3: Take an input of name from user and print Have a great day ahead {name}

```
#!/bin/bash
```

```
read Name
```

```
echo "Have a great day ahead $Name."
```

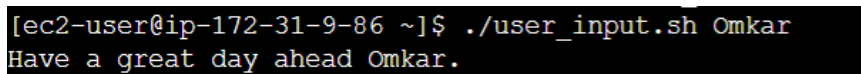
A screenshot of an AWS terminal window. The top bar shows the AWS logo, a 'Services' menu, and a search bar. The terminal prompt is '[ec2-user@ip-172-31-9-86 ~]\$'. The user has run './input.sh', which has printed 'Omkar' and 'Have a great day ahead Omkar.'. The prompt is now '[ec2-user@ip-172-31-9-86 ~]\$' with a cursor.

```
[ec2-user@ip-172-31-9-86 ~]$ ./input.sh
Omkar
Have a great day ahead Omkar.
[ec2-user@ip-172-31-9-86 ~]$
```

OR

```
#!/bin/bash
```

```
echo "Have a great day ahead $1."
```

A screenshot of an AWS terminal window showing the execution of a script. The prompt is '[ec2-user@ip-172-31-9-86 ~]\$'. The user has run './user_input.sh Omkar', which has printed 'Have a great day ahead Omkar.'. The prompt is now '[ec2-user@ip-172-31-9-86 ~]\$' with a cursor.

```
[ec2-user@ip-172-31-9-86 ~]$ ./user_input.sh Omkar
Have a great day ahead Omkar.
[ec2-user@ip-172-31-9-86 ~]$
```

Assignment 4 : Let's take a scenario of fintech app program in which we want to have three separate outputs for 3 different situations:

->The balance is less than zero

->The balance is zero

->The balance is above zero

For instance, in the following program, use the if, elif, else statements to display different outputs in different scenarios:

Use "if" condition to check if the balance is less than zero. If this condition evaluates to true, display the message using the echo command: "Balance is less than zero, Please add more funds else you will be charged penalty".

If the above condition does not match, then use "elif" condition to check if the balance is equal to zero. If it evaluates to true, display the message: Balance is zero, please add funds

If none of the above condition matches, use the "else" condition to display the: Your balance is above zero.

```
#!/bin/bash
read balance
if [ $balance -lt 0 ]
then
echo "Balance is less than zero, Please add more funds else you will be charged penalty."
elif [ $balance -eq 0 ]
then
echo "Balance is zero, please add funds."
else
echo "Your balance is above zero."
fi
```

Assignment 5 : Debug and define briefly about the following program :-

```
#!/bin/bash
# Print a message about disk usage.
space_free=$( df -h | awk '{ print $5 }' | sort -n | tail -n 1 | sed 's/%//' )
case $space_free in
[1-5]*)
echo Plenty of disk space available
;;
[6-7]*)
echo There could be a problem in the near future
;;
8*)
echo Maybe we should look at clearing out old files
;;
9*)
echo We could have a serious problem on our hands soon
;;
*)
echo Something is not quite right here
;;
esac
```

- Case statement is used when there are multiple options to choose from.
- Case statement begins with 'case' keyword and ends with 'esac' keyword.
- Pattern case is ends with parenthesis ')' and body ends with ;;.
- Default pattern case is noted by '*' symbol.
- When one patterns is matched, case statement stops matching other cases.
- df -h -> command shows disk space in human readable format.
- awk '{ print \$5 }' -> command splits the line and prints \$n variable.
- | -> pipe operator takes output of one command and gives it to next command as input.
- sort -n -> sort numbers in ascending order.
- tail -n -> command prints last 'n' numbers of inputs.
- sed 's/%//' -> SED stands for Stream Editor, it can perform multiple operations on file like search, replace, insert, delete.
- s specifies substitution operation, % is replaced with blank.