**Shell Scripting**

Automation is the process where we try to reduce the manual activities.

Shell scripting is a type of programming that involves writing commands for a Unix or Unix-like operating system shell (such as Bash, Zsh, or Ksh) to automate tasks, process text files, or perform system administration tasks. It allows users to create scripts that can be executed in the command-line interface.

The basic syntax for shell scripting involves writing a series of commands in a text file and saving it with a **".sh"** file extension. The file must be made executable using the **"chmod"** command. The first line of the file should specify the shell that will be used to execute the commands (for example, **"#!/bin/bash"**).

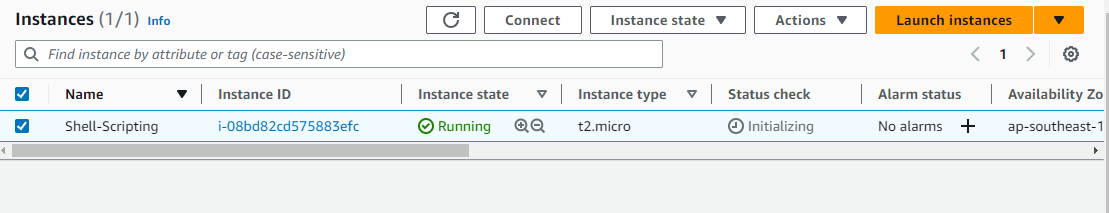
Here's an **example of a basic shell script**: bashCopy code

**#!/bin/bash echo "Hello, world!"**

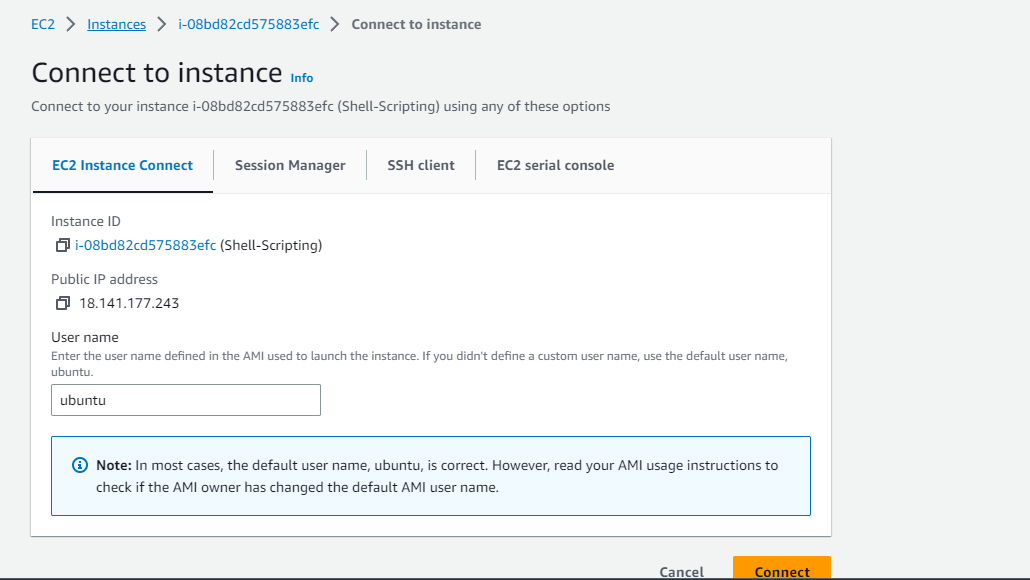
This script will print the message "Hello, world!" when executed. The first line specifies that the Bash shell will be used to execute the commands, and the "echo" command is used to output the message to the console.

Other basic syntax elements of shell scripting include variables (e.g. x=10), conditionals (e.g. if statements), loops (e.g. for and while loops), and input/output redirection (e.g. > and < operators for writing and reading from files).

**Install the Linux machine instance on aws:**



**To connect to Linux machine instance:**

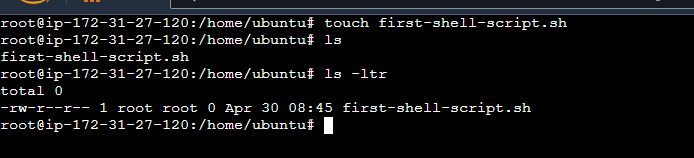


**How to create a file using Shell Scripting:**

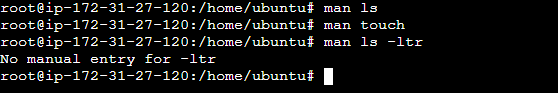
#touch first-shell-script.sh

#ls

#ls –ltr



**Man command in Linux:**



**To write a file in Linux using VI command:**

#vi first-shell-script.sh

Press **i** and go to insert mode

Write the file:

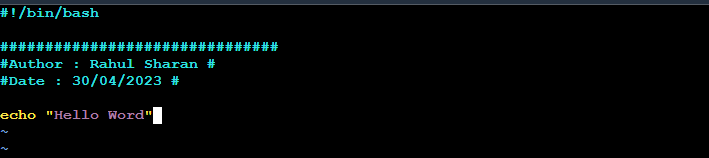
#!/bin/bash, #!/bin/dash, #!/bin/sh, #!/bin/ksh

**echo “Hello World”**

Press **ESC** and go out from the insert mode and save the file

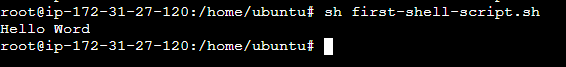
**:wq!**





**To execute the file using the following command:**

# sh first-shell-script.sh

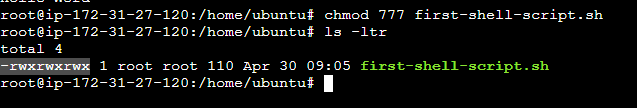


**To grant permission in Linux using CHMOD Command:**

User (owner), group and other user

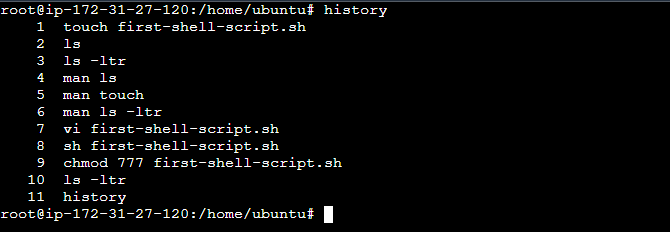
Read (4), write (2), and execute (1) =7 full permission

#chmod 777 first-shell-script.sh



**To check the history of Linux commands:**

#history



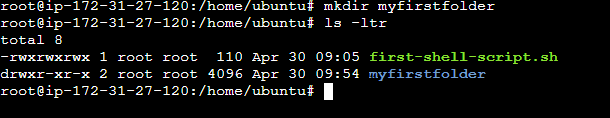
**To check the current or present working directory:**

#pwd



**To create a folder by using the following command:**

#mkdir myfirstfolder



**To change the directory in Linux by using the following command:**

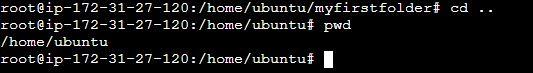
# cd myfirstfolder



**Go back to previous directory by using following command:**

#cd ..

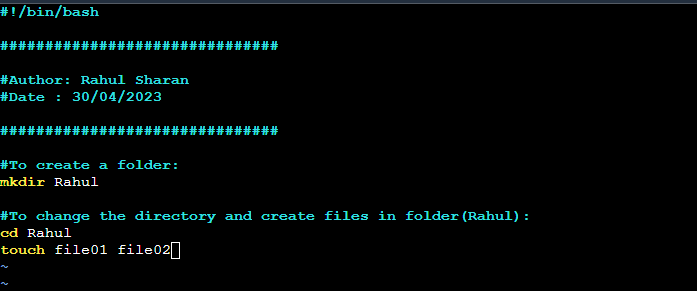
#pwd



**To create folder and files by using the shell scripting:**

#vi folder-files.sh





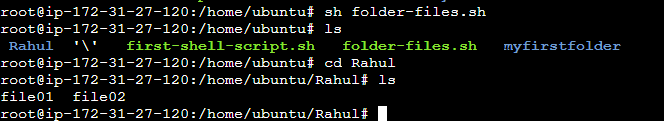
**To grant permission:**

#chmod 777 folder-files.sh



**To execute the file:**

#sh folder-files.sh



**What is the purpose of Shell Scripting in DevOps?**

For example: Rahul works as a DevOps Engineer at XYZ.com. He has noticed that xyz.com manages nearly 1,000 VMs, all of which are Linux-based. He is responsible for monitoring the health of all these VM nodes. In this scenario, he must manually check each VM one by one.

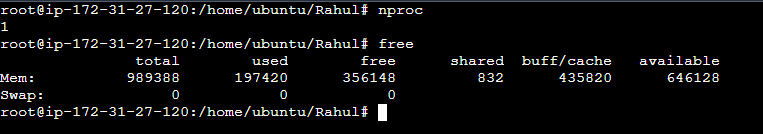
Developers have been encountering issues with node health, such as CPU usage reaching capacity, memory becoming fully utilized, and Linux machines performing sluggishly. Rahul finds himself spending a significant amount of time checking each individual machine, which is a time-consuming process.

To streamline the process of monitoring the health of all VM nodes, it is considered a best practice to create a shell script and store it on GitHub. By executing the shell script on his local machine, Rahul can efficiently identify which VMs are experiencing issues or are down.

**To check the CPU and RAM of a Linux machine:**

#nproc

#free



#top

