# Index

| Date | Topic | Practise Date | Live | Practise status | Doubts | Status | IMP |
|---|---|---|---|---|---|---|---|
| April -2025 | | | | | | | |
| Linux | | | | | | | |
| 27/April/2025 | Demo _session | 27/April/2025 | Attened | Completed | | | |
| | | | | | | | |
| 28/April/2025 | Sesssion -1 | 28/April/2025 | Attened | Completed | | | |
| 29/April/2025 | Sesssion -2 | 29/April/2025 | Attened | Completed | | | |
| 30/April/2025 | Sesssion -3 | 30/April/2025 | Attened | Completed | | | |
| May -2025 | | | | | | | |
| 1/May/2025 | Sesssion -4 | 1/May/2025 | Attened | Completed | | | |
| 2/May/2025 | Sesssion -5 | 2/May/2025 | Attened | Completed | | | |
| | | | | | | | |
| 5/May/2025 | Sesssion -6 | 5/May/2025 | Watched Later | Completed | | | |
| 6/May/2025 | Sesssion -7 | 6/May/2025 | Attened | Completed | | | |
| 7/May/2025 | Sesssion -8 | 7/May/2025 | Attened | Completed | | | |
| 8/May/2025 | Sesssion -9 | 8/May/2025 | Attened | Completed | | | Trouble shottingpart - 56:00 Min |
| 9/May/2025 | Sesssion -10 | 9/May/2025 | Attened | Completed | | | |
| | | | | | | | |
| 12/May/2025 | Sesssion -11 | 12/May/2025 | Attened | Completed | | | |
| 13/May/2025 | Sesssion -12 | 13/May/2025 | Attened | Completed | | | |
| 14/May/2025 | Sesssion -13 | 14/May/2025 | Attened | Completed | | | |
| Shell | | | | | | | |
| 15/May/2025 | Sesssion -14 | 15/May/2025 | Attened | Completed | | | |
| 16/May/2025 | Sesssion -15 | 16/May/2025 | Attened | Completed | | | |
| | | | | | | | |
| 19/May/2025 | Sesssion -16 | 19/May/2025 | Attened | Completed | | | |
| 20/May/2025 | Sesssion -17 | 20/May/2025 | Attened | Completed | | | |
| 21/May/2025 | Sesssion -18 | 21/May/2025 | Attened | Completed | | | |
| 22/May/2025 | Sesssion -19 | 22/May/2025 | Attened | Completed | | | |
| 23/May/2025 | Sesssion -20 | 23/May/2025 | Attened | Completed | | | |
| | | | | | | | |
| 26/May/2025 | Sesssion -21 | 26/May/2025 | Attened | Completed | | | |
| Ansible | | | | | | | |
| 27/May/2025 | Sesssion -22 | 27/May/2025 | Attened | Completed | | | |
| 28/May/2025 | Sesssion -23 | 28/May/2025 | Attened | Completed | | | |
| 29/May/2025 | Sesssion -24 | 29/May/2025 | Attened | Completed | | | |
| 30/May/2025 | Sesssion -25 | 30/May/2025 | Attened | Completed | No Notes in this session | | |
| | | | | | | | |
| 2/June/2025 | Sesssion -26 | 2/June/2025 | Attened | Completed | | | |
| 3/June/2025 | Sesssion -27 | 3/June/2025 | Attened | Completed | | | |
| 4/June/2025 | Sesssion -28 | 4/June/2025 | Attened | Completed | | | |
| 5/June/2025 | Sesssion -29 | 5/June/2025 | Attened | Completed | | | |
| Terraform | | | | | | | |
| 6/June/2025 | Sesssion -30 | 8/June/2025 | Attened | Completed | | | |
| | | | | | | | |
| 9/June/2025 | Sesssion -31 | 10/June/2025 | Attened | Completed | | | |
| 10/June/2025 | Sesssion -32 | 12/June/2025 | Attened | Completed | | | |
| 11/June/2025 | Sesssion -33 | 14/June/2025 | Attened | Completed | | | |
| 12/June/2025 | Sesssion -34 | 14/June/2025 | Attened | Completed | | | |
| No session | | | | | | | |
| | | | | | | | |

# Index

| Date | Topic | Practise Date | Live | Practise status | Doubts | Status | IMP |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

| | |
|---|---|
| Linux Basic Commands | https://www.linkedin.com/posts/mallikarjuna-bidurakanthu-870371221_basiclinuxcommands-activity-7324656080234434560-j2pO?utm_source=share&utm_medium=member_desktop&rcm=ACoAACkHIQEB8ikVTo2hQ574ivZeyb42tQp0nGg |
| sudo dnf clean all | |
| | sudo set-prompt frontend |
| | check logs /var/log  > less messages<br><br>Shift + G to check the bottom lines |
| | nginx logs - /var/log/nginx/access.log<br>backend component logs - /var/log/messages |
| etc | Extra Configurations |
| | we have our own AMI   --> In companies also we have our OWN AMI<br>devops-practice, ec2-user ,  DevOps321<br>not keybased, this is password based |
| | AMI  - pre configured OS with some packages |
| Shift + insert | Shortcut to paste the clipboard to Vim file |
| ls -l /tmp | ls -l /folder_name |
| How to delete all the swap files at a time | vim -r /etc/momgod.conf -- by this cmd you can edit with required input and exit<br>after for deleting the swap file<br>rm /etc/mongod.conf.swp |
| sudo su | It will login to root user but the directcry will be ec2-user |
| sudo su - | It will login to root user and as well the directory also will change to root user |
| Node Js | package.json -> node_modules |
| Java | pom.xml -> /target/shipping.jar |
| | /usr/share/nginx/html/*                - Website connect is present in this path which we see as applications |
| | tail -f access.log                -> -f is following the logs |
| | telnet mysql.daws84s.cloud 3306      - Is used to connect particular IP/ port or not     - Exit from telnet Ctrl+ ]  and then quit |
| | /etc/systemd/system/backend.service                                  - systemctl daemon-reload |
| | /etc/nginx/default.d/expense.con |
| | -> Front end logs      cd /var/log/nginx -> tail -f access.log   - For running logs |
| | -> Backend logs -> Catalogue logs cd -  /var/log/   > less messages |
| | netstat -lntp             -> netstat -lntp    -  l List ,  n - Port Number , t - Tcp , p for process |
| | nslookup IP    - To check the record is updated or not |
| | vim /etc/systemd/system/catalogue.service                                    -   file_name.service is mandet extenstion   and if you made any changes in this directory this systemctl daemon-reload is the mandet command |
| | For any application mostly the backend port is 8080 |
| | Backend applications: port no - 8080 will be opened<br>Frontend applications: port no - 80 will be opened<br><br>Mysql port: 3306 |
| | To check catalogue is connectinf to mongodb or not we need to user the command in catalogue terminal - telenet mongodb url port no i.e telenet mongodb.daws84s.cloud 27017 |
| | G - To go to the bottom of a long file in Vim, use:<br><br><br>Bonus Vim Navigation Tips:<br>gg → Go to the top of the file.<br><br>:n → Go to line number n, e.g., :100 to jump to line 100.<br><br>Ctrl + f → Scroll forward one screen.<br><br>Ctrl + b → Scroll backward one screen. |
| | https://www.geeksforgeeks.org/maven-build-lifecycle/ |
| CTRL + Mouse Drag and Drop | It will help in deuplicate the file in VS code |
| | sudo dnf remove nginx mysql -y |
| | git remote set-url origin https://sriharidevops2155:ghp_YDmApFEXkazAVI7UGIXxbqnAgMX17D30wjjm@github.com/sriharidevops2155/robo-shop-full-document |
| tee | The tee command reads from standard input and writes to both:<br><br>Standard output (your terminal), and<br><br>One or more files. |
| &>> | Redirect both stdout and stderr and Append both to file |
| curl http://localhost:8080/health | [ ec2-user@ip-172-31-80-33 ~/shell-roboshop ]$ curl http://localhost:8080/health<br>{"app":"OK","mongo":true} |
| mkdir -p /app | |
| sudo visudo | Session 21 at 15:20 |
| $TO_TEAM | Special variable will work only in double quotes - At  Session 21 -> 45:41 |
| du -sh * | |
| ssh ec2-user@IP_ADDRESS | To login to other system |
| | ssh ec2-user@172.31.89.180 'echo "Hello Ansible" > /tmp/ansible.txt' |
| vvv is for checking logs | ansible-playbook -vvv -i inventory.ini 14-conditions.yml |
| Hide logs when playbook is running | ansible-playbook playbook.yml -q                  -----------------> q = quiet |
| 2. Use no_log: true in Specific Tasks | - name: Hide sensitive command output<br>  command: some_command<br>  no_log: true |
| List and Map Differences | [ - List , { - Map |
| { - Map | Key = Value |
| | {<br>      Name = "HelloWorld"<br>      Purpose = "Variables-demo"<br>   } |
| terraform fmt | |
| | $ terraform console<br>> max(12,13,14)<br>14 |
| Terraform | Yellow colour is change , Green + is creation/add , ~ is change ,  - remove |

Hey everyone! I hope the following two Q&As will be helpful to some of you.

How DNS and Global Infrastructure Work Together in Large-Scale Applications?

Here's a high-level overview - Big companies like Facebook do not dynamically change the IP addresses of their public-facing production servers frequently. Instead, they use technologies like global load balancers, Anycast routing, and Content Delivery Networks (CDNs). These systems may route users to different IP addresses, but that's not because the actual backend server IPs are changing dynamically — it's due to traffic optimization(optimize traffic flow), improve geographic routing, and load balancing(balance server load).

As a result, the IP address that a user sees may change every few minutes, depending on factors like location and network conditions. These IP addresses are cached based on the domain's DNS TTL (Time To Live). Once the TTL expires, the DNS resolver performs a new lookup to fetch the most updated IP address for continued routing.

Why use a reverse proxy?

Here's a high-level overview - Let's take Facebook or Instagram as examples. When we use these platforms, our devices connect to an alias IP address - not the actual backend application server. You might wonder why that is. The reason is that these applications use technologies like global load balancers, Anycast routing, and Content Delivery Networks (CDNs) to route users to different IP addresses based on factors such as location and network conditions and connects to edge server(part of a Content Delivery Network, that cache and distribute content closer to end users to improve speed and reduce load on the actual server)

Depending on your location, your regular feed on Facebook or Instagram may show local advertisements, trending topics, or popular personalities. Even the data servers serving this content may be located at the nearest possible point to you, ensuring reduced latency and better performance as content (like images, videos, and feeds) may be served from a edge(nearest located) server.

From a security perspective, reverse proxies also play a crucial role — they hide the backend infrastructure and help absorb or filter out malicious traffic before it reaches the core servers.

| Interview Questions | | | |
|---|---|---|---|
| | Linux and Shell | | |
| Linux Basic Commands | https://www.linkedin.com/posts/mallikarjuna-bidurakanthu-870371221_basiclinuxcommands-activity-7324656080234434560-j2pO?utm_source=share&utm_medium=member_desktop&rcm=ACoAACkHIQEB8ikVTo2hQ574ivZeyb42tQp0nGg | | |
| | find /tmp/* -mtime +7 -exec rm {} \;   - https://stackoverflow.com/questions/31389483/find-and-delete-file-or-folder-older-than-x-days | | |
| | Interview Question: How will you get the inode number<br><br>Answers: Using command:  stat  <file_name> | | |
| | cat /proc/cpuinfo | | |
| | iostat | | |
| FInd the logs older than 14 days and delete them | find . -name "*.log" -mtime +14 | | |
| Recent Challange you faced in your scripting | FILES=$(find $SOURCE_DIR -name "*.log" -mtime +$DAYS) \| zip -@ "$ZIP_FILE"    - > Session 20 1:44 min | | |
| Shell Assignments | https://github.com/bajibpathan/jds_shell_assigment/tree/mainn | | |
| Siva Sir Interview Preparation Q& A | https://learn.joindevops.com/learn/Interview-Preparation | | |
| How can you call other scripts from current script | We can call by 2 ways<br><br>1. Through source    -> Through source both scritps have same process instance id and both scripts can access variable and functions (Variables will execute automatic and functions will run only when we cal)<br><br>2. Through ./            --> ./ approach diff process and it cannot access variables | | |
| How can you genarally exit the shell script if you get error | We can have 2 options, You can set -e with trap function or else we can write one validate function that will print and check the exit status and exit  if any error | | |
| | set -e<br><br>failure(){<br>echo "Failed at : $1 $2"<br>}<br><br>trap 'failure "${LINENO}" "${BASH_COMMAND}"' ERR | | |
| | Purpose of Hidden Files:<br>Hidden files often contain configuration settings, user preferences, and other important system-level information. They may be used by applications or the operating system itself.<br>Examples:<br>.bashrc: Configuration file for the Bash shell.<br>.ssh: Directory containing SSH keys.<br>.config: Directory containing application-specific configuration files.<br>.gitignore: Git configuration file for ignoring certain files or patterns during version control. | | |
| | Interview Question: How do you check running log<br>Answer:   tail -f <log-file>  -      -f is follow     -> Tail is the last lines and -f is the latest last lines updating while taking logs     - https://join-devops.slack.com/archives/C08PV10QXGV/p1746160328981459?thread_ts=1746157244.503419&cid=C08PV10QXGV<br><br>how to check running logs in linux<br><br>Use tail -f /var/log/ or less /var/log/<br><br>OR<br><br>Go to your log file parent directory using<br>cd log directory path<br>Then do<br>tail -f logfilename | | |
| | Interview Question: What do you do if you want to replace a word in a file<br><br>Answer: Using substitue i.e  s/Acutal_word/Replace_word/word to change | | |
| | Interview Question: Which file you change to provide the root access<br><br>Answer: /etc/sudoers --> you can do changes to provide root access<br>or add user to wheel group without password<br><br>We can add then into wheel group or we can edit sudoers file to provide root access | | |
| Where you use shell scripting in your project | Usually we write in docker files , Jenkins pipe lines like wise and alo for monitering scripts and configuration management also we are using | | |
| What is configuration | In a plain server we are adding all the dependencies and propertie files to make the server host the application | | |
| | Ansible in background uses shell and linux | | |
| What is crontab | Crontab is the way of scheduling the scripts or commands in linux -> We can schedule particular date and time and day and month and it will store the logs also | | |
| What are the ansiblemodules you used | dnf , service , replace , copy , shell, command module etc | | |
| - name: diable default niginx<br>  command: dnf module disable nginx -y<br><br>- name:<br>  command: dnf module enable nginx:1.24 -y | - name: diable default niginx<br>  command: dnf module disable nginx -y ; dnf module enable nginx:1.24 -y | | |
| If you want to execute specific task from a playbook then what else you will do | We can make use of tags, We are using in our project  so we have VM deployments using ansible and we are using ansbile tags for deployment related task | | |
| Have you used Vault in your ansible ? | Yes we used ansible vault we are storing our mysql user name and password in our vault and we will give the decription password in the command line | | |
| How do you create vault file | ansible-vault file_name | | |
| At session- 33 -> 11 June 2025 13:40 min | In Interviews: Before we are using dynamodb locking, But recently I migrated it to S3 native locking | | |

**Session - 1**

**https://github.com/daws-84s**

**Environments** **DEV, QA, SIT, UAT, PERF, PRE-PROD, PROD**

**SIT- System Integration Testing**

**PERF - Performance**

**Linux --> Linus torvalds**

**windows**
**-----------**
**not open source**
**costly**
**not secure must install anti virus**
**slow**
**too many graphics**
**frequent restarts**
**tough to update/upgrade**

**Linux --> 9MB**
**Secure**
**high speed --> mostly text**
**no need of restarts**
**install or update packages is easy**
**free, community support**
**low resources**

**DevOps is the process of building, deploying and testing the code written by developer on the same day instead of doing after complete development..**
**we can acheive this using continous integration, continous deployment, continous delivery and continous testing.**
**We can do faster releases with less defect using DevOps... basically automation mindset to get the best results**

**DEV, QA, SIT, UAT, PERF, PRE-PROD, PROD**

**min 2, max any number**

Linux --> Linus torvalds
=================
UNIX --> Hardware and Software(Unix OS) together
CPU, RAM, Hard Disk, OS(Software)
Laptop --> IBM BIOS

Linux --> from the sratch using C language based on unix principles. He invented git to store this

windows
-----------
not open source
costly
not secure must install anti virus
slow
too many graphics
frequent restarts
tough to update/upgrade

Linux --> 9MB
Secure
high speed --> mostly text
no need of restarts
install or update packages is easy
free, community support
low resources

## Session - 2

what is computer?
=================
IP enabled device

laptop == mobile == server == chip == tv ==> computer
server ==> only to host applications

RAM, OS, HD, Processor --> IP enabled device

Client Server architecture

Networking between computers

1. Network
2. Facebook application

serve --> lawyer serves to us, we are his client

facebook is in Linux server, if problem comes we need to login to server..

https://www.joindevops.com

Protocol: https
Port: 443
IP: DNS IP
Username and password --> authentication

SSH --> Secure shell
Port: 22
IP, Username and password/private-key

Firewall --> checks inbound and outbound traffic

inbound --> incoming traffic --> port number 22, 103.149.59.114
outbound --> outgoing traffic

authentication
=================
1. what you know --> username password
2. what you have --> keys, tokens, authentication, **rsa token**
3. what you are --> fingerprint, retina, palm, face

Server == **box** == node - > **In companies server is generally noted as box by everyone**

lock = public
key = private

key pairs --> public key and private key

1. create key pair
2. create firewall
3. import public key to aws account
4. create server and attach public key
5. attach firewall to server

Git Bash --> Mini Linux in windows
SSH client

browser --> http/https client
Linux Server --> SSH Server

git bash == putty == super putty == mobaxterm == mac terminal = windows cmd

ssh clients

pwd --> present working directory

User Directory
C:\Users\siva --> windows format
**/c/Users/siva --> linux format**

Windows accept space in file name -> Linux won't accept

windows --> not case sensitive Siva == siva == Siva
**Linux --> case sensitive --> Siva != siva no space in linux**

/c/devops/daws-84s == C:\devops\daws-84s

ssh-keygen -f <file-name>

ssh-keygen -f daws-84s

**ssh-rsa long-random-code Sri Hari. Bandi@DESKTOP-NSG82HF**

**Region --> us-east-1 - N.Virginia -> AWS is first created in US at N>virgina and hence using any service in this locations is cheaper in terms of cost**

0.0.0.0/0

98.81.70.98, 22, SSH, ec2-user, private-key

ssh -i <private-key> ec2-user@98.81.70.98

**OS --> Redhat, Ubuntu, centos, fedore, suse, oracle linux, amazon linux, debian, rocky linux, kali, solaris**

Linux is os or not --> Linux is kernel

**OS --> to interact with hardware**

**Kernal --> heart and brain of OS**

**Kernel + UI == OS**

Redhat --> kernel + user interface ==> Redhat OS
Debian --> Linux kernel + UI ==> Debian

**99.9% same all Linux distros/flavours**

Redhat --> Open source --> code is free
Redhat enterprise RHEL --> support

**Current RHEL is OS version is 9**

**RHEL = AWS Linux = Centos = Fedora = Rocky Linux = AlmaLinux**

**Linux is a kernal, RHEL is the distribution or Linux flavour**

**Search for Linux Distributions in Linux in Google -> It will show all dirstibutions**

command <options> <inputs>

**uname -> Will show which kernal we are using -> FYI Amazon Linux bacially uses Linux Kernal**

uname -a .

**uname --help**

**will display all options avilable in uname command**

**/home/ec2-user --> linux home directory**
**/c/users/siva --> windows home directory**

cd --> change directory

cd .. --> one step back

ls --> list subdirectories

```
drwx------. 3 ec2-user ec2-user  95 May  1 05:52 .
drwxr-xr-x. 3 root     root      22 May  1 05:46 ..
-rw-------. 1 ec2-user ec2-user  28 May  1 05:53 .bash_history
```

d is directory
- Files

## Session - 3

ssh

/c/devops/daws-84s/daws-84s --> full path **- > Absolute Path** Starts from scratch - i.e from /c drive

ssh -i /daws-84s ec2-user@Ip -> Relative path

**$ --> normal user**
**# --> root/admin/super user**
**/root --> root user home directory**

command <options> <inputs>

/ --> root directory

ls -l --> long listing format in alphabetical order
ls -lr --> long listing format in reverse alphabetical order
ls -lt --> latest files on top
ls -ltr --> latest at bottom
ls -la --> all files including hidden files and folders
ls -lth -> h is human readable

touch <file-name> --> creates empty file

cat > <file-name> --> type text, enter and ctrl+d
tac <file_name>  also will display the file output

previous content will be replaced
cat >> <file-name> --> appends text to previous content

> --> usually called as redirection

**mkdir <name> --> creates directory**
**mkdir -p <dirname/dirname/dirname>**
**rmdir --> remove empty directory**
**rm -f --> forcefully removes file**
**rm -rf --> recursively forcefully delete the files and folders inside to**o

CRUD --> create read update delete

cp <source> <destination> --> copy files/folders
mv <source> <destination> --> cut and paste

**wget <URL> --> downloads the file**
**curl <URL> --> shows on the screen**

cat <file-name> | grep <word-to-search>

**cat <file-name> | grep -i  <word-to-search> it will also show case sensitive**

grep <word-to-search> <file>

grep --help

https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1

```
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f9
session-02.txt
```

**f is fragment here**

```
echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1,9
https:/session-02.txt
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1-9
https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1-4
https://raw.githubusercontent.com/daws-84s
```

awk command
------------
echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $1F}'

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $NF}'    -> NF gives here last word

log files --> tail -f <log-file>

**find <where to search> -name <file-name to search in where to search directory>**

```
[ec2-user@ip-172-31-16-199 etc]$ awk -F ":" '{print $1F ,"|" ,$3F}' passwd
```

vim --> visually improved

```
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $1f}'
https:
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $nf}'
https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $Nf}'
https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt
[ec2-user@ip-172-31-16-199 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $NF}'
session-02.txt
[ec2-user@ip-172-31-16-199 ~]$
```

cat passwd | head -> Gives you the top 10 lines in output
cat passwd | taik-> Gives you the bottom 10 lines in output

cat passwd | head -n6     -> top 6 lines
cat passwd | tail -n6     -> top 6 lines

find . -name "*name*" what is the text that have name it will give in output

**Interview Question: How do you check running log**
**Answer:   tail -f <log-file>  -      -f is follow     -> Tail is the last lines and -f is the latest last lines updating while taking logs     - https://join-devops.slack.com/archives/C08PV10QXGV/p1746160328981459?thread_ts=1746157244.503419&cid=C08PV10QXGV**

**how to check running logs in linux**

**Use tail -f /var/log/ or less /var/log/**

**OR**

**Go to your log file parent directory using**
**cd log directory path**
**Then do**
**tail -f logfilename**

```
[root@ip-172-31-16-199 ~]# find / -name config
/etc/selinux/config
/etc/crypto-policies/config
/sys/kernel/config
/sys/devices/pci0000:00/0000:00:01.0/config
/sys/devices/pci0000:00/0000:00:00.0/config
/sys/devices/pci0000:00/0000:00:01.3/config
/sys/devices/pci0000:00/0000:00:03.0/config
/sys/devices/pci0000:00/0000:00:01.1/config
/sys/devices/pci0000:00/0000:00:02.0/config
/usr/lib/modules/6.1.134-150.224.amzn2023.x86_64/config
/usr/lib/python3.9/site-packages/cloudinit/config
/usr/lib/python3.9/site-packages/awscli/botocore/data/config
[root@ip-172-31-16-199 ~]#
```

**Expected Interview Questions from Session-03**

**1. What is the difference between cut and awk commands in linux**

**echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1**

**command explanation**

**echo url means, it will print the url**
**|  uses the output of previous command**
**cut >> to cut the printed url**
**-d "/" >> delimiter(sperating the url)  and url is getting seperated, here the seperator is /**
**-f1 >> this indicates the first fragemnt of the url**

**echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $1F}'**

**command explanation**

**echo url means, it will print the url**
**| uses the output of previous command**
**-F "/" >> Fragment and the url is getting seperated, here the seperator is /**
**"/" '{print $1F}'  >> indicates to print the first fragment of the url**

**The main difference is in cut command if we want to print the last part of the url, it is little difficult(we have to count the total no of fragments manually and use in the command). But this is resolved in awk command using '{print $NF}'**

**========================================================**

**2. Print the list of users in your linux server.**

**cp /etc/passwd mylinuxusers   >>> copying the list of users in the current linux server to a floder named "mylinuxusers"**
**Now use the ask command**
**awk -F ":" '{print $1F}'**
**We may not give the same answer to the interviewer , but we explain the above example how to use awk command to print the list of users.**

## Session - 4

**vim editor - Visually Improved**

CAT >
CAT >>

vim <file-name>

3 MODEs
Esc, Colon, insert MODEs

command MODE
==============
:q -> quit the file
:wq --> write and quit
:wq! --> force write and quit -> if this file is opened by 2 members at a time -> Now we are force writing and quiting.
:q! --> exit without saving
:/<word-search> --> search for the word from top **or direct / also we can use**
:?<word-search> --> search for the word from bottom - if you type n in then it will show from starting

:/ is from top and ? is from bottom.

**:noh --> no highlight**

**:set nu --> To set the numbers in rows wise.**

**:set nonu --> To unset the numbers.**

**d**

:27 d --> delete 27th line

:%d --> total content will be deleted

**Interview Question: What do you do if you want to replace a word in a file**

**Answer: Using substitue i.e s/Acutal_word/Replace_word/word to change**

:3s/sbin/SBIN --> in 3rd line by default only the first occurence of sbin will be replaced with SBIN

:3s/sbin/SBIN/g --> all occurence in that line

:%s/sbin/SBIN/g --> all occurences in the file

**Here**
**s for substitute**
**g for global**

If this is the case, use ":recover" or "vim -r devopssamplepractise"

Esc Mode
===============
Esc + u --> undo
yy --> copy the line where you are - Yank - 10yy it will copy the 10 lines
p --> paste
dd --> cut the line - single d Enter will cut 2 lines
gg --> takes to top of the file
shift+g --> takes us to bottom of the file or **CAPITAL - G**

**Clipboard to Vim Paste shortcut -**

Linux Administration
=====================
User management

Only root user/Administator can add the users to machine

CRUD

useradd <user-name> --> create user

id <user-name> --> displays the user info

```
[root@ip-172-31-16-199 ~]# id suresh
uid=1001(suresh) gid=1001(suresh) groups=1001(suresh)
```

gid is primary group id

groups - is secondary group id

group --> list of similar users
devops team have 20 members
create devops group, add team members to the group

**in linux when you create user, by default group also will be created with same name..**
**primary group and secondary group**

**1 primary group, atleast one secondary group**

cat /etc/passwd
cat /etc/group

groupadd devops --> devops group will be created

usermod -g devops suresh
usermod -aG testers suresh **-aG- - a- Append G - Secondary Group**

**[root@ip-172-31-16-199 ~]# gpasswd -d suresh devops**
**Removing user suresh from group devops**

groupdel suresh

useradd suresh

passwd suresh

ssh suresh@ip

vim /etc/ssh/sshd_config

Change password auth to yes

sshd -t - to test

systemctl restart sshd

passwd <user-name>

```
Sri Hari. Bandi@DESKTOP-NSG82HF MINGW64 /
$ ssh suresh@34.224.86.3
suresh@34.224.86.3's password:
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/         https://aws.amazon.com/linux/amazon-linux-2023
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[suresh@ip-172-31-16-199 ~]$
```

**IAM Team**

## Session - 5

**Permissions**
**==============**
**R -> 4**
**W -> 2**
**X -> 1**

```
-                    rw-            r--            r--
file/             user/   group      others
                   owner
directory    u       g       o
```

ec2-user ec2-user
user               group

chmod ugo+w devops.txt

**chmod ugo+rwx devops.txt**

chmod 700 devops.txt

**only owner/root user can change the permissions**

chown <user>:<group> devops.txt

chmod suresh devops.txt -> This will change the owner ship to suresh

chmod suresh(Owner)

file ownership can only be modified by root user
/etc/ssh/sshd_config

how can you give key based access to linux user?

ssh-keygen -f sivakumar

**65,536 ports 0-65,535**

1. create user
2. sivakumar can send his public key to ad min user
3. /home/sivakumar admin creates .ssh in /home/sivakumar folder
4. sivakumar is the only owner to this folder... 700
5. create a file called authorized_keys with max access 600
6. admin keeps sivakumar public key here.
7. now sivakumar should be able to login

port is like flat number

systemctl status sshd

**25:15 minute** ssh -i <key> ec2-user@IP - **In a server there are lot of services running ex http service , mail service , hhtps , ssh and my sql like this it can run n number of services**

**And here there will a particular port for each service. As here we gave ssh so that it will go for port no 22**

**For each service a particular port is assigned like a flat in Appartment.**

**Interview Question: Which file you change to provide the root access**

**Answer: /etc/sudoers --> you can do changes to provide root access**
**or add user to wheel group without password**

**We can add then into wheel group or we can edit sudoers file to provide root access**

**vim /etc/sudoers**

**If you add any user to wheel group that user will get the root access - 38:30**

package management
====================
package manager connects to internet windows website. downloads them and install them

apt-get
yum/dnf

dnf install <package-name>

dnf git remove -y

/etc/yum.repos.d

cat authorized_keys.

service management
===================
systemctl start <service-name>
systemctl stop <service-name>
systemctl status <service-name>

**sudo systemctl status sshd**

systemctl restart <service-name>

systemctl enable <service-name> -> If laptop is restated the service will run automatically ex- bluetooth for this to start autmatically if the system is restarted or stop and on then it will run by default

systemctl disable <service-name>

install nginx and start it

**protocol, port, ip, username and password**

**http 80 <IP>**

dnf list installed

**http - > Port Number is 80**

**https -> Port Number is 443**

dnf list avilable

```
[ec2-user@ip-172-31-16-199 ~]$ sudo su
[root@ip-172-31-16-199 ec2-user]# cd /tc
bash: cd: /tc: No such file or directory
[root@ip-172-31-16-199 ec2-user]# cd /etc
[root@ip-172-31-16-199 etc]# find yum
find: ‘yum’: No such file or directory
[root@ip-172-31-16-199 etc]# find -name "*yum*"
./yum.repos.d
[root@ip-172-31-16-199 etc]# cd ./yum.repos.d/
[root@ip-172-31-16-199 yum.repos.d]# ls
amazonlinux.repo  kernel-livepatch.repo
[root@ip-172-31-16-199 yum.repos.d]#
```

**To create a user via Keys**

**usedadd user_name**
**make sure home directory will be created using a user_name**

**cd to that user_name**

**mkdir .ssh**
**cd .ssh**
**touch authorized_keys**

**In a new terminal we can use ssh-keygen -f user_name**

**From here paste the pub key to vim authorized_keys**

**now come to pwd as .ssh**

**then chmod 600 -R .ssh**

**then chown username:group .ssh**

**And to create a user without key then**

**useradd username**

**passwd username**

**vim /etc/ssh/sshd_config change password auth to yes**

**Session - 6**

IPv4process management
====================
TM --> TL --> TM --> Senior --> Junior --> Frehser

TM --> 5 tasks for 5 TL

ls -l --> it creates a process and assign a process ID, executes the command in kernel --> get the o/p --> display on screen

ps - > Process

ps -ef - every process - Full-format listing

PID - Process instance ID

```
UID         PID    PPID  C STIME TTY          TIME CMD
root          1       0  0 May02 ?        00:00:15 /usr/lib/systemd/systemd --switched-root --system --deserialize=32
root          2       0  0 May02 ?        00:00:00 [kthreadd]
root          3       2  0 May02 ?        00:00:00 [rcu_gp]
root          4       2  0 May02 ?        00:00:00 [rcu_par_gp]
root          5       2  0 May02 ?        00:00:00 [slub_flushwq]
root          6       2  0 May02 ?        00:00:00 [netns]
root          8       2  0 May02 ?        00:00:00 [kworker/0:0H-events_highpri]
root         10       2  0 May02 ?        00:00:00 [mm_percpu_wq]
```

PID and PPID

ps -ef | grep <process-name>

sleep 10 - It will run in foreground and it will the users to interact with CLI for some time

sleep 10 & -> It will run in background

(&) - ampersand.

How do you take the process to run in background

Using & ampersand

Sleep 100 &

foreground and background process
& to take process into background

If a press is stuck or not responding we can use - kill PID - 21:00

kill PID --> request to terminate
kill -9 PID -> order to stop -> It will stop forecfully

systemctl status nginx
ps -ef | grep nginx

we have application running --> PID

10 users --> 0.1% CPU 0.1% RAM
2000 users --> 10% CPU, 20% RAM
10 users

top -p PID --> check the resources used by particular process

Network management
===========================
how do you check open ports in linux

netstat -lntp

systemctl status <service-name>

ps -ef | grep service-name

netstat -lntp - l List , n - Port Number , t - Tcp , p for process

**IPv4 - 192.0.2.1**

**IPv6 - 2001:0db8::1**

**By running this 4 commands we can check the process or application is running proper or not:**
**systemctl status <service-name>**
**ps -ef | grep service-name**
**netstat -lntp**
**top -p PID**

top -p PID

htop -p PID

htop software we have to install manually

dnf install htop -y

## 3 tier architecture:

===========================
desktop vs web application

**desktop: drawbacks-**
resources usage more
hang
installation
upgrade
storage
compatability issues
no data security/backup
can't access everywhere

architecture is important

road side breakfast shop
==========================
single person --> less than 10

taking order, cooking the item, serving, payment collection

hotel
====================
cook --> prepares the item
owner --> counter --> issuing tokens

let's say 20 members

100 members

restaurant
====================
captain --> welcomes the customer
waiter --> takes the order
chef --> prepares the order
waiter --> garnish with cabbage, onion, lemon, keera, etc.

directly to chef
=================
1. security
2. queue management --> reduces taste of item

```
select * from users where username = 'siva@joindevops.com' and password = 'siva123'
```

**visual studio, putty, terraform, aws cli v2**

Java - Maven Build tool
.NET - New Get
Python - PIP
Node JS - NPM

**Session - 7**

we have our own AMI   --> **In companies also we have our OWN AMI**
devops-practice, ec2-user ,  DevOps321
not keybased, this is password based

AMI --> Amazon machine image

image --> we and our surroundings

AMI --> base OS --> redhat 9 + nginx + applications + packages --> take it as an image

**AMI  - pre configured OS with some packages**

public ip vs private ip

192.168.1.102 --> private
103.149.59.114 --> public

**Internal Communication should be done using private IP**

**Outside of Network or Extranal Communication should be done using Public IP**

**ISP - Internet Service Provider**

Linux is physical server... nginx is virtual server running in linux server

nginx is

**nginx is the server which serves web technologies  ex: Tomcat etc , LiteSpeed Web**

yum install nginx -y

dnf install nginx -y

**dnf is the advanced version of yum, there are modules in dnf**

apt-get install nginx -y

**/etc/nginx/nginx.conf --> configurations. you must restart the service when there are some config changes**

**root /usr/share/nginx/html - Host File in Webserver / Frontend**

http://44.201.38.173:80

**index.html --> default html**

godaddy --> domain registar, almost 1000 every year
hostinger --> cheapest domains for 1st year, for every renewal it is costly

joindevops --> 62819 37079
word = meaning

key --> value

computers only understand 0 and 1.. usually these binaries will be converted to decimal

facebook --> 143.234.53.98  --> **Ip will be converted as domain name by Domain Name System i.e facebook**

Root Servers are Managed by 13 Members  in word wide by non Profit Organizations https://www.iana.org/domains/root/servers - these team have full authoritry to manage DNS

Domain Name system

DNS resolver is the component in ISP(**ISP - Internet Service Provider**) that is responsible to provide IP address.

DNS resolver contacts to root servers

Example like How Autor is for Oxford Dictonay like the same or DNS Resolver Root servers are the authorized servers

**TLD - Top Level Domain**

facebook.com --> TLD .com
daws84s.site --> TLD is .site

.in, .io, .org, .net, .edu, .gov, .co, .ai, .shop, .pizza, .space, .online

.siva can be registered as TLD. godaddy, hostinger, etc are domain sellers.

Forst we have to register our TLD with root servers and later our TLD will be saled by  -> godaddy, hostinger, etc are domain sellers

TLD - **Root servers**
Domain Sellers - **godaddy , hostinger**

What root servers will do is it will scan the request ex if it is .com then it will route to that .com TLD infomation   - 1:15 hr

Name Servers DNS/Name Servers

**Types of Records:**

A --> IP address
CNAM --> points to another domain
MX --> mail records
TXT --> domain verifications
NS --> who is managing this domain now

Who is Domain Registry here - Godaddy or Hostigner

**Domain registar responsibility to update the details to TLD**

Name Server - NS --> who is managing this domain now

Nameservers - Port like Airtel sim to jio

TTL -> Time To Live  -> Example like if we continously browse the facebook.com and if TTL is 1 hr then my browsing cache will be there for 1 hr and after 1 hr if try the request will come as new

https://toolbox.googleapps.com/apps/dig/

## Session - 8

**google.com --> browser cache --> OS --> OS Cache --> ISP DNS resolver --> root servers** (Root Servers wil not have the IP info and they are just maintaining the TLD information) **will give TLD info--> TLD info --> .com TLD Will have name servers --> name servers (**Who are managing that domain**) --> A record**

name servers --> who are managing that domain now
domain registar --> a mediator to sell the domain, they have default name servers and we are changing the authority to AWS Acc

create hosted zone in AWS, get the NS records and update in domain provider
TTL --> time to live is DNS cache before issuing next DNS request

**A --> IpV4**
CNAME --> points to another domain
MX --> mail records
TXT --> domain verification
NS --> Nameservers
**AAAA -> Ipv6**

750 hours --> t2.micro or t3.micro
31 days

1 instance --> 1month
2 instance --> 15 days
11 instances --> 3 days

mobaxterm

**sudo set-prompt frontend**

**MySQL/Oracle/Postgress/MSSQL --> RDBMS --> tables and columns, primary keys and foreign keys**

**MongoDB --> NoSQL --> documents - Is used for storing for heavy data / big data - Also when compare to MY SQL it is tough to maintain MongoDB**

**And Example for some MongoDB applications are - Flipkart , Amazon etc**

**MongoDB Latest v8 and we are using v7**

**MongoDB is in json format**

```
{
	"name": "sivakumar",
	"email": "info@joindevops.com"
}
```

127.0.0.1 == localhost

**system/service user --> non human users which will not have login, and these are intented to run applications**

we will create one folder and download our code there

dnf install nginx
dnf install mongodb-org

dnf install catalogue --> can't

/app --> this directory to have catalogue code

I can keep /app as home directory to systemuser

nginx:x:991:990:Nginx web server:/var/lib/nginx:/sbin/nologin

**/etc/shadow --> encrypted password here**

roboshop user --> home dir /app and nologin

useradd --system --home /app --shell /sbin/nologin --comment "roboshop system user" roboshop

Java/NodeJS/Python

**Node JS - v20**

java script --> .js
java --> .java
python --> .py
go --> .go
php --> .php
c language --> .c

**every programming have dependencies or libraries - Example house construction - 52.18 Min**

**build tool -> compile(syntax check), download the dependencies, ready to run**
**build file**

**Every programming language will have build tool and build file i.e for node js below is - 54: 30 Sec**

**Build tool - nodejs --> npm (node package manager)**
**build file --> package.json - > it contains required libraries and scripts**

**npm install --> it search for package.json in the current directory - 55: 54 Sec**

After running npn install it will download all the dependencies from package.json and a new directory will be created with **node_modules** in the same path

you install programming language
create one directory
create one user
downlod the code
install dependencies

**we need to create systemctl file --> starting, stopping, restarting, enable/disable are easy for this application**

**/etc/systemd/system --> create .service file, catalogue.service file**

**systemctl daemon-reload - We are telling systemd to reload so it will detect new service.**

**nslookup IP**

**Session - 9**

MySQL
=======
table and column database --> RDBMS

Linux Server --> MySQL server

House --> Linux Server
Room --> DB server
Rac --> Schema --> tables
1DB server may have 1 or many
transactions, users,

**After running npn install it will download all the dependencies from package.json and a new directory will be created with node_modules in the same path**

**Mostly for:**

**Backend applications: port no - 8080 will be opened**
**Frontend applications: port no - 80 will be opened**

**Mysql port: 3306**

telnet mysql.daws84s.site 3306 --> checks whether mysql is allowing connections to it on port 3306

/etc/nginx/nginx.conf

prime --> server, vpn is client for prime
client --> mobile

we are hiding client identity --> forward proxy
actual client --> forward proxy --> server

1. content filtering
2. traffic monitoring
3. anonymous client
4. geo restrictions bypass
5. caching

Reverse proxy
----------------
Nginx can be used as load balancer, reverse proxy server, web server

1. client is not aware of server identity
2. security
3. load balancing
4. SSL termination

Frontend --> Backend

telnet backend.daws84s.site 8080
login to backend and check systemctl status backend
**check logs /var/log > less messages**

**Shift + G to check the bottom lines**

http://daws84s.site/api/transaction

HTTP Status codes
-------------------
1XX --> information
2XX --> success
3XX --> redirection

errors
---------
4XX --> client side error
5XX --> server side error

CRUD
=======
INSERT
SELECT
UPDATE
DELETE

reports --> daily, weekly, monthly, quarterly
bank-records
user
notifications
ai/ml

saas

configuring server
-------------------------
Linux server creation
programming language installation includes build tool
create one dedicated directory for our application /app
create one system user for our application roboshop
download the code into our directory
install dependencies
create systemctl service
if required load data
start the service

telnet mysql.daws84s.site 3306 --> connected
user password problem
query problem

**Session - 10**

frontend
daws84s.site/api/catalogue -->

http://daws84s.site/api/catalogue/categories

http://daws84s.site/api/catalogue/products/Artificial%20Intelligence

```
[
    {
        "_id": "681d63fc59c47e0558c59f36",
        "sku": "Ewooid",
        "name": "Ewooid",
        "description": "Fully sentient assistant",
        "price": 200,
        "instock": 0,
        "categories": [
            "Artificial Intelligence"
        ]
    },
    {
        "_id": "681d63fc59c47e0558c59f3e",
        "sku": "STAN-1",
        "name": "Stan",
        "description": "Observability guru",
        "price": 67,
        "instock": 1000,
        "categories": [
            "Robot",
            "Artificial Intelligence"
        ]
    },
    {
        "_id": "681d63fc59c47e0558c59f35",
        "sku": "Watson",
        "name": "Watson",
        "description": "Probably the smartest AI on the planet",
        "price": 2001,
        "instock": 2,
        "categories": [
            "Artificial Intelligence"
        ]
    }
]
```

/var/log/nginx/access.log

/var/log/messages

HD(ROM) --> RAM --> Apps

App --> DB

**Redis (REmote DIctionary Server) -> It is known for ultra-fast performance due to its in-memory(stores in RAM) architecture**

Send a connection req to DB
Connection establish
Run a query against DB
DB should fetch the data from HD
Send response to app
Close the connection

App --> Cache
App --> Cache --> DB --> Cache

## Session - 11

mongodb
catalogue
frontend

redis
user
cart

mysql
shipping

rabbitmq
payment
dispatch

2XX --> Success
4XX --> client side errors
5XX --> server side errors
1XX --> informational
3XX --> redirectional

HTTP Methods --> GET, POST, PUT, DELETE, OPTIONS

Nouns and Verbs

CRUD --> Create Read Update Delete

User --> Noun, Create User --> Verbs

http://daws84s.site/user and method GET --> getUser  --> readuser
http://daws84s.site/user and method POST --> CreateUser
http://daws84s.site/user and method PUT --> updateUser
http://daws84s.site/user and method DELETE --> deleteUser

http://daws84s.site/order and method POST

http://daws84s.site/orders and method GET

http://daws84s.site/orders/O654S5HG and method GET

http://daws84s.site/api/catalogue/categories --> http://catalogue.daws84s.site/categories
http://daws84s.site/api/catalogue/products/Artificial%20Intelligence

http://localhost:8080/products/Artificial%20Intelligence

http://daws84s.site/api/catalogue/product/Ewooid

http://localhost:8080/product/Ewooid

backend apps always do CRUD operations on DB

nginx logs - /var/log/nginx/access.log
backend component logs - /var/log/messages

**Java --> Maven --> pom.xml**
**groupId, artifactId, version**

**group id** - com.hdfc
**artifact id** - banking.savings.smsbanking
**version:** 1.0.0

Java --> source code is with .java extension
java code compile --> bytecode --> run bytecode/compiled code

**There is no need of compling node JS and python in specifc and we run them direclty as compile will be done in during run time**

**Java code first we have to compile and we will get bite code**

**java code compile --> bytecode --> run bytecode/compiled code**

nodejs --> npm

maven --> java

**.java --> java source file extension and after compiling we will get .class as it we get .class**
**.class --> java bytecode extension**

**Out of Java , nodejs and Python which is speed** ? > **Java as compilation is already done as system likes to execute**. > **But in nodejs and Python while run time compilation will take**

**when we install nodejs we got build tool npm**

**here for java if you install maven java will install by default and vice versa**

**mvn clean package --> clean previous installs and do the fresh packaging**
**.jar --> packaging in java(bytecode)      -> There is no need of packaging in node js and python bcz there is no compilation and it will run directy but in Java we should must create jar file**

**node_modules -> Dependiences are downloaded here**
**target - Java application is binded here**

mvn compile --> compile the java source code and create bytecode in .class format in target folder
mvn package --> pack the java class files into single file called shipping-<version>.jar

maven life cycle --> clean compile test and package
package == compile+test+package

room --> DB server
racks --> schema
tables

python
file extension --> .py
build tool --> pip
build file --> requirements.txt

install PL
create one directory

**Session - 12**

MQ Database
================
**Synchronous** and **Asynchornous**

sync vs async(fire and forget)

async  is like (fire and forget) i.e we sent a msg and we don't need to bouther of it as it will be delivered when other side system is up on automatic

**Synchronous: Request responde module - i.e Request expects immediate response**
http://daws84s.site
request sent --> waits for 1 min --> if no response then error
http expects immidiate response --> sync

**ASynchronous communication -  MQ - Message Queue - When other side system is down it will wait until system is up and no data lost**
===================
1. point to point -> message will be delivered to single system
2. topic and subscribe --> message will be delivered to all subscribers

when I upload video, all my subscribers get notifications

Example for Asyc communication is: order is accepted ans sent request to Third party delivery.

**MQ - Message Queue:**

ActiveMQ
RabbitMQ
IBMMQ
Kafka

robotshop and we have a vendor to delivere our orders

we get 1000's of messages from our deliverey partner

**Maven Build Workflow:**
Below is a typical Maven build workflow for the default lifecycle.

**Validate Phase:** Check the project configuration and validate if all information required is present.
**Compile Phase**: Compile the source code of the project.
**Test Phase:** Execute unit tests using a testing framework.
**Package Phase:** Bundle the compiled code into a JAR/WAR file.
**Verify Phase:** Perform checks on the packaged code.
**Install Phase:** Install the package to the local repository for use as a dependency in other projects locally.
**Deploy Phase:** Deploy the package to a remote repository for sharing with other developers.

**Session - 13**

payment --> python

extension --> .py
pip
requirements.txt

**softlink/symlink hardlink and inode**

**shortcut ex chome in desktop and notepad** (is also a file pointing real file) --> **points to other real/original file**
if I delete real file, shortcut will break

**inode --> this is the metadata of the file**

**file permissions**
**ownership**
**size**
**original memory address location of the file**

**ls -i -> inode -> list files with i node**

**stat <file_name>**

**Interview Question: How will you get the inode number**

**Answers: Using command: stat <file_name>**

**lrwxrwxrwx 1 user user 15 May 13 10:00 shortcut.txt -> /home/user/file.txt**

**lrwxrwxrwx - starting l is link file**

**Sym link will be created with: ln -s <original-file> <symlink-file> - AT 33:19 -> example in front end ln -s nginx.conf /root/nginx.conf**

**Hard link: ln <original-file> <symlink-file> --> AT**

**symlink is like shortcut to original file. hardlink is like a copy of file. it is like backup**

**symlink and original file will have a diff inode. symlinks are used to point diff versions**

**find / -name "*.conf"**

**find / -inum <i_node_num>**

/etc/nginx/nginx.conf --> /etc/nginx-1.conf
/etc/nginx/nginx.conf --> /etc/nginx-2.conf

dnf-4 install nginx -y

/usr/bin/dnf --> dnf-4

**Symlink will have diff inodes**

**Hardlinks will have same inode**

**ls -li -> lrwxrwxrwx 1 user user 15 May 13 10:00 shortcut.txt -> /home/user/file.txt**

lrwxrwxrwx **1 -> Number of link files**

**47:30 Sec**

**which dnf**

**if original file is deleted symlink will break. if original file deleted hard link will work**

**symlink is used to point diff versions, hardlink is used as backup to original file**

**symlink can be created to folders, we can't create hard link to folders**

**symlink will work with diff types of disks, hardlink can't be created for diff disks**

**memory troubleshooting**
-------------------------
memory - RAM
storage - HD

free

free -m -> In mega bites

free -h -> Human readable format

**Swap memory is from HD as virtual RAM**
HD --> RAM --> Programs

using top command we can see the running memory usuage

**ps -ef it will list all the process running**

**auxlory**
ps aux -> along with process it will display cpu and ram usuage

**To see the top 5 memory consuming process:**
ps aux --sort=%mem | head -n 5

ps aux --sort=%mem | tail

ps aux --sort=-%mem | tail

ps aux --sort=-%cpu| tail

**disk usage -> df -hT**

du -sh <folder-path> - s- summarize h - Human readable
du -sh * --> all files and folders size will be displayed

some time in var/log -> from log days some many logs are logged and is uses more memory and some time it will go for GB's and what we do is we wil try trouble shoot and do archive the logs

**/app --> /opt/app** - opt Optional -> Usually third party apps is processed in /opt/app directory

du -ah / | sort -rh | head -n 20

du -ah /  Show sizes of all files and folders from the root /, in human-readable form
sort -rh  Sort the output in reverse (-r) order by human-readable size (-h)
head -n 20  Show the top 20 largest entries from the sorted list

**Linux Directory structure -** https://github.com/daws-84s/concepts/blob/main/linux-directory-structure.MD

**/usr/share/nginx/html --> nginx root directory**
/usr/bin
/usr/sbin
/usr/local/bin/

/proc --> processes started by kernel

cd / - is root directory

cd /root -> root user home folder

**$ is where ever we want to refer a variable we should use dollar before it**

## Session - 14

1. time
2. human errors

100 logs carry from farm to home

all commands keep it in file, run it as a file(Coding)

if I run a command --> I have common sense, I can understand whether it is success or not...

variables
data types
functions
conditions
loops
error handling

90% design + 10% coding

Coding
1. programming --> developers. data strcuture, alogorithms, design patterns
less resources(RAM, CPU, HD), less time
2. scripting --> DevOps. simple scripting

shell vs python
\==============
**UI --> Shell/CMD --> Kernel**
UI --> Python --> Shell/CMD --> Kernel

**Shell is native in Linux. If you want any automation inside linux server, shell is the perfect choice.**
**python is useful when you are interacting with other/external systems.** **--> Python is used when you are interacting the linux with github and get some server**

Keep code in version control/source code management(SCM)

1. security
2. tough to share the code - When multiple team members are working
3. no versions

Centralised vs Distributed
\===================
google drive --> 10 members
open google docs in browser, do the change

donwload the doc in local laptop, open in word do the changes then upload to drive

SVN vs Git(GitHub, Gitlab, Bitbucket, etc.)

IDE(Integrated development env) --> vscode, eclipse, intellij, sublime, etc.

1. create repo in github
2. clone/download the repo
3. add to staging/temp area.
4. commit
5. push changes

```
git add . ; git commit -m "message"; git push origin main
```
. means all the changes

<file_name>.sh
sh <script-name>

**#!/bin/bash** --> shebang, it should be the first line in shell script, it is the interpeter to check the syntax and execute the commands..

sh == bash

x=0, y=1, z=3
solve the expression and finally substitute variables

label --> item
key=value

variable is a container to hold the value, you can use it wherever you require. and this centralised, if you change at once place it will reflect wherever you use. DRY (Don't repeat yourself)

args/arguments

sh 04-variables.sh value1 value2 -> value1 and value2 are arguments
$1=value1
$2=value2

```
sh variables.sh "Revanth Reddy" KTR
Revanth Reddy: Hey KTR, how are you ?
KTR: Hello Revanth Reddy, I am fine. How are you doing
Revanth Reddy: I am fine too. What's up?
Revanth Reddy: Nothing, Just going to Mars now, will you come?
Revanth Reddy: Sorry, You can carry on! I will come once you are come back
```

```
NUMBER1=100
NUMBER2=200

SUM=$(($NUMBER1+$NUMBER2))

echo "sum of $NUMBER1 and $NUMBER2 = $SUM"

TIMESTAMP=$(date)

echo "Scrpit executed at: $TIMESTAMP"
```

Here NUMBER1, NUMBER2 TIMESTAMP are variables which is holding some value in it

**1:17:11 IMP**

Data Types
\================
int i=0
float
double
decimal
string
char
boolean
array arraylist set map

**1:23:00 IMP**

**MOVIES=("Court" "HIT3" "Pushpa2" "Thandel")**

~~echo "first movie name is $MOVIES[0]"~~

**echo "first movie name is ${MOVIES[0]}"**

$NAME == ${NAME}

**Session - 15**

**Special variables in shell**
============================
we no need to define the variable, shell will provide to us
$1

**if you want all variables passed to script: $@**
**number of variables passed: $#**
**script name: $0**
**present working directory: $PWD**
**home directory of the user who is running the script: $HOME**
**which user is running the script: $USER**
**PID of the current script: $$**
**sleep 10 &     -> & Ampresent will make your script run in background**
**PID of the last command running in background: $!**
**pervious/last command exit status $?**

**How do you check the exit status in shell script:**
**By running $?**

| Exit Code | Meaning |
|---|---|
| 0 | Success (command executed without error) |
| 1 | General error |
| 2 | Misuse of shell builtins (e.g. missing keyword, bad syntax) |
| 126 | Command found but not executable |
| 127 | Command not found |
| 128 | Invalid argument to exit |
| 130 | Script terminated by Ctrl+C (SIGINT) |
| 137 | Script killed (e.g., kill -9, SIGKILL) |
| 139 | Segmentation fault (often core dumped**)** |
| 255 | Exit status out of range or custom fatal error |

```
sh special-vars devops aws azure
All variables passed to the script: devops aws azure
Number of variables: 3
```

Conditions
---------------

```
if(expression){
	this block runs if expression is true
}
else{
	this block runs if expression is false
}

if(raining){
	take umbrella
}
else{
	no need to take
}
```

raining false, ! false.. false*false = true

```
if(!raining){

}
else{

}
```

write a program if given number is less than 10 or not

```
if(number < 10){
	print number is less than 10
}
else{
	print number is not less than 10
}
```

-gt --> greater than
-lt --> less than
-eq --> equal
-ne --> not equal

install my**sql**
**---------------**
**1. user is running with root access or not**            **(i**d , sudo id, If you want only id info then id -u  && root user id is always 0)
2. if root user proceed to install
3. else show th**e error that** p**lease run with root acce**ss

exit status
---------------
0-127

success --> 0
other than 0 --> failure

**dnf list installed <package_name>**

Functions
---------------
a machine is functioning --> doing something

coffee maker
inputs: coffee beans, milk, water
output: coffee

function takes input and do some work, you can call whenever you require

```
function coffeeMaker(milk, coffee beans, water){
	boilwater
	crushbeans
	mix milk, water and powder
	return coffee
}

coffeeMaker(milk, coffee beans, water)
```

repeated code you can keep it in function and call whenever you require

mysql nginx python3

you can provide args to function just like you do to the script

```
sh install-command.sh mysql python3 nginx
```

**Session - 16**

Colors in shell
---------------------

```
\e[<code>m
```

31 -> red 32 -> green 33 -> yellow

```
echo -e "Hellow World" ------------------------> -e means enabling the colours

echo -e "\e[31m Hellow World"        -----------------------------------> \e escaping the special characters
Hellow World


echo -e "\e[31m Colour are implemented \e[0m"        ---------->  \e[0m You should speciy till whil line
```

redirections
---------------------

```
< -- denotes input
> --> denotes output
```

**ls -l > output.log** --------------> It stores only sucess output

**ls -l 1> output.log --> 1 denotes success output If you dont give anything also it will take sucess by default i,e ls -l > output.log**
lfadflafd 2> output.log --> 2 denotes failure output

**ls -l &> output.log -----------------> & Store what ever i.e sucess/failure logs**

14-logs.sh.log
14-logs.log

14-logs.sh

mkdir -p /var/log/shellpractise-logs    ---------------> -p is for if alredy folder is created ignore the command and if folder is not exists then create a folder

I want basic log on the screen too then use **tee**

EXAMPLE: echo "Script started executing at: $(date)" | tee -a &>>$LOG_FILE                        --------------------> -a for append

loops
-------------------

```
for(int i=0; i<=100; i++){
       print i;
}
```

```
1. create 11 instances
2. rename them
3. update r53 record
       if( instance is frontend )
              update public ip
       else
              update private ip
```

frontend --> API --> backend

authentication and authorization
===================================

authentication --> prove yourself
authorization --> prove yourself, check whether you have access or not

access key and secret key

aws configure

```
 aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]:
```

aws s3 ls --> just check working or not, no o/p also fine except error

```
1. create instances
       AMI ID, SG ID, SUBNET ID
```

AWS CLI to Create ec2 and get IP - https://www.google.com/search?q=AWS+CLI+to+Create+ec2+and+get+IP&sca_esv=b97c0088c4eec1a1&rlz=1C1GCEU_enIN1161IN1161&sxsrf=AHTn8zquWxqYb6I8h_VZCONhgjk57z2lZA%3A1747829493701&ei=9cltaPnHKsWXnesPgb_AyAE&ved=0ahUKEwi5luagxLSNAxXFS2cHHYEfEBkQ4dUDCBA&uact=5&oq=AWS+CLI+to+Create+ec2+and+get+IP&gs_lp=Egxnd3Mtd2l6LXNlcnAiIEFXUyBDTEkgdG8gQ3JlYXRlIGVjMiBhbmQgZ2V0IElQSNYhUIINWMAccAR4AZABAZgB0wKgAbsIqgEHMC4yLjIuMbgBA8gBAPgBAZgCBqACogPCAgoQABiwAxjWBBhHwgIIECEYoAEYwwTCAgoQIRigARjDBBgKwgIEECEYCpgDAIgGAZAGCJIHBTQuMS4xoAfFCrIHBTAuMS4xuAeJA8IHBTAuMS41yAcY&sclient=gws-wiz-serp

```
aws ec2 run-instances --image-id ami-09c813fb71547fc4f --instance-type t2.micro --security-group-ids sg-01bc7ebe005fb1cb2 --tag-specifications
"ResourceType=instance,Tags=[{Key=Name, Value=test}]" --query "Instances[0].PrivateIpAddress" --output text

aws ec2 describe-instances --instance-ids i-0abcdef123456789 --query "Reservations[0].Instances[0].PrivateIpAddress" --output text
```

**IAM stands for Identity and Access Management.**

**Session - 17**

1. Process idea
2. Coding syntax

https://repost.aws/knowledge-center/simple-resource-record-route53-cli

sed editor --> stream editor / streamline editor

adding the lines
deleting the lines
replace the words

sed -i/-e 'expression' filename

adding the line

sed -e '1 a Hello Wolrd' users --> Adding the line **1** is first line **a** is apped/after and in the place of a if we give **i** it will insert

sed -i '2 d' users -> Delete 2nd line

sed -e 's/sbin/LOGOUT/' users It will sibstitue sbin to logout and if you add /g at last s/sbin/LOGOUT/g it will substitue in all entries

sed -e '/Overflow/ d' users it will delete the line where Overflow text is there

**-e --> temp change onto the screen -> It will help in come cases like without making changes in original file we can do executions on run time**
**-i --> it makes the permenant change in file directly without showing the output in screen**

idempotency
==============
irrespective of the number of time you run something it should not change the result

API Development

POST --> not idempotency
check user already exist or not, if not create otherwise show the error
first time entry will be created

GET --> idempotent

PUT(UPDATE) --> idempotent, but for better user experience, same details are already updated

DELETE --> idempotent, but for better user experience, user already deleted

## Session - 18

frontend --> user

sed -i "s/127.0.0.1/0.0.0.0/g"  -i "/protected-mode/ c protected-mode no" /etc/redis/redis.conf      -> if we want to change multiple texts in a single file and c is from remove complete line and add text provided

**Session - 19**

DRY --> don't repeat yourself

write a seperate file, create functions with all common code, call when required...

How can you call other scripts from current script

script-1.sh
script-2.sh

how you call script-2 from script-1

**./script-2.sh**
**=========================**
**can't access scritp-1 variables**
**PID are different**

**source ./script-2.sh**
**=========================**
**same PID for both scripts**
**SCRIPT-2 can acess SCRIPT-1 variables**
**SCRIPT-1 can access SCRIPT-2 variables**

**set -e -> This will stop execution as soon as it script hit with error**

**trap**

```
for item in $@
do

done
```

item in $@ --> condition

```
while [ condition ];
do
   # statements
   # commands
done
```

In projects generally 14 days old logs are preffered to delete or archieve and in Banking projects 14 days older logs also not deleted

```
while IFS=
do

done < delete-oldlogs.sh
```
--> This input we are giving from file

```
while IFS=
do

done <<< delete-oldlogs.sh
```
--> For <<< 3 backward buttons is like we are pushing the input from script output

## Session - 20

```
find <source_directory> -name "*.log" -mtime +14
```

source_directory
find the files
delete the files

14 days old log files can be zipped and placed into destination directory...storage team usually monitors the destination directory if they find any files there they move to another server i.e files server

source_directory
destination_directory

need to check source_directory directory exist or not
destination directory also should be checked

find the files
zip the files
move to destination directory
then remove the files in source_directory

```
sh backup.sh <source-directory>  <destination_directory> <days>
```

```
DAYS=${3:-14}  # :- if DAYS are provided that will be considered, otherwise default 14 days
```

app-logs-$TIMESTAMP.zip

crontab --> scheduling the scripts in linux --> usually non business hours at 3 or 4AM

```
* * * * * sudo sh /home/ec2-user/shell-practice/20-backup.sh
/home/ec2-user/source-dir /home/ec2-user/dest-dir
/home/ec2-user/source-dir /home/ec2-user/dest-dir
/home/ec2-user/source-dir /home/ec2-user/dest-dir
```

corntab -l

corntab -e        > will work like vim for corntab

corntab

**/var/log/corntab   -> 1:49:37 min**

```
sudo tail -f /var/log/corn
```

```
send email/gmail from linux
===========================
we have few mail comands
```

sudo cp backup.sh /usr/local/bin/backup

corntab -e

/app-logs --> application continously writing log files

12ki new cart.log will be created
cart-23-05-2025.log

df -hT --> every 30min --> send email if usage is more than 70%

mlfhlchsfxeiqrax

```
{
echo "To: info@joindevops.com"
echo "Subject: Testing Gmail"
echo "Content-Type: text/html"
echo ""
echo "Testing Gmail Body"
} | msmtp "info@joindevops.com"
```

```
while IFS=
do

done < delete-oldlogs.sh                    --> This input we are giving from file

while IFS=
do

done <<< delete-oldlogs.sh                    --> For <<< 3 backward buttons is like we are pushing the input from script output

while IFS= read -r line
do

then <<< $FILES_TO_DEL
```

## Session - 21

which ls

```
/home/ec2-user/.local/bin
:/home/ec2-user/bin:
/usr/local/bin
:/usr/bin
:/usr/local/sbin
:/usr/sbin
```

Crontab
========
/usr/bin:/bin

recently I was given a script to create backup of the files, I successfully created the script and shceduled in crontab, but command not found in crontab. When I executed it is working fine manually. Later I understood PATH is bare minimum in crontab so I changed my script as command into /usr/bin directory. It started working fine.

**In projects generally the thresold value is b/w genarally 70-80%**

**sudo visudo.**

disc utilization command

```
defaults
auth          on
tls           on
tls_trust_file /etc/ssl/certs/ca-bundle.crt
logfile       /var/log/msmtp.log

account       gmail
host          smtp.gmail.com
port          587
from          your_from_email@gmail.com
user          your_from_email@gmail.com
password      app-password

account default : gmail
```

```
{
echo "To: <your-to-email>"
echo "Subject: Test Email"
echo "Content-Type: text/html"
echo ""
echo "This is test email from Shell scripting session"
} | msmtp "<your-to-email>"
```

/var/log is consuming more than 80%

Hi DevOps Team,

There is High Disk Usage alert in the system. Please check

```
High Disk Usage on /run: 2
High Disk Usage on /: 30
High Disk Usage on /var: 22
High Disk Usage on /var/tmp: 3
High Disk Usage on /home: 5
High Disk Usage on /var/log: 4
High Disk Usage on /boot: 53
High Disk Usage on /boot/efi: 6
High Disk Usage on /var/log/audit: 2
```

Regards,
Monitoring Team

https://www.onlinewebtoolkit.com/text-to-html

Configuration Management
========================
Provision server

system packages install
programming runtime install   -  nodejs ,  java , python
application folders, users,etc
download the code
install dependencies
create service files
start the server

deployment
=======================
stop the service
1. remove the old version
2. download the new version, install dependencies
3. start the server

disadvantages  [1:09 min]
================
not idempotent
poor error handling
if we have more servers, very difficult to run --> not scalable
scripts works only for specific distro --> homogenous
syntax is little tough
password security is poor

Configuration Management Tools --> ansible, chef, rundeck, puppet, etc...

**Session - 22**

Coding
	variables
	data types
	conditions
	loops
	functions
	error handling

how do you call shell script
./script.sh

source ./script.sh

backup script
disk utilisation
cpu/memory utilisation
crontab

no error handling
if more servers, difficult to handle...not scalable
homogenous
syntax is little tough to understand
password security is not available
not idempotent

configuration management
=======================
it is provisioning the server for hosting the application...
system packages
programming langugage
folder, user creation, etc
download the code
install dependencies
create systemctl services
start the service/server

push vs pull
======================

US -> India

### pull/poll model

**Fedx shop**

**HYD**
**1. go to fedx daily**
**2. check whether you got letter or not**

pull/poll model
====================
1. time waste
2. petrol waste
3. resource waste
4. increasing traffic on roads

push model
====================
HOME
1. whenever letter receive in local fedx shop
2. delivery boy delivers the letter to us

SYSTEM-1 --> SYSTEM-2

chef --> pull based configuration management server

1. ssh connection
2. create the file with content
3. closed the connection

Remote login ssh@IP_ADDRESS

Ansible takes remote login using SSH protocol to the nodes
It will do the task
come out of the server

Ansible installation command - sudo dnf install ansible -y

Linux Command == Ansible Module/Collection - Adhoc commands

<command> <options> <inputs>

ls
ls -l
ls -lrt

cp <source-dir> <dest-dir>

ansible all -i IP, -e ansible_user=ec2-user -e ansible_password=DevOps321 -m ping

Shell Script == Ansible Playbook

YAML --> Yet another markup langugage
.yml or .yaml

Bank
==========
white paper to deposit

acc no, date, branch, name, money, denomination, sign

forms/templates

Date
acc_no
branch
name
amount
denomination

DTO --> XML, JSON

XML --> Extensive Markup language

<Username>sivakumarredy</Username>

```
<User>
	<Username>sivakumarredy</Username>
	<Password>siva123</Password>
</User>
```

```
{
	"username": "sivakumarredy",
	"password": "siva123"
}
```

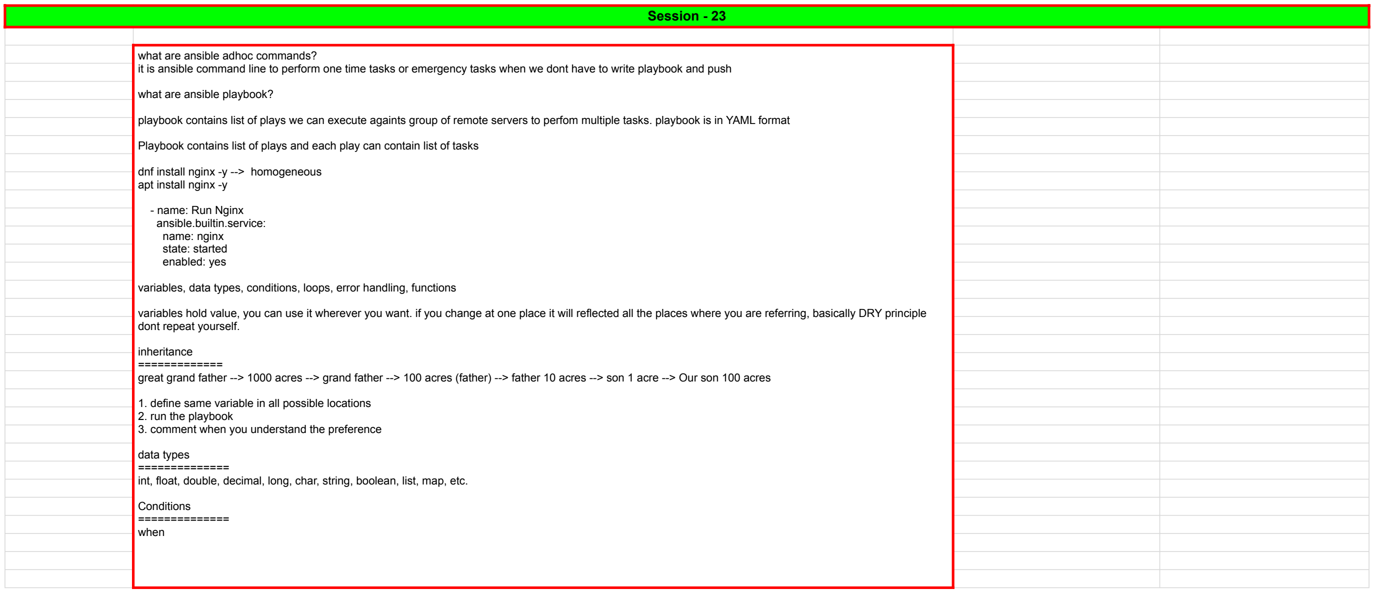
What object can do, what object knows. I am trainer

exp, what tools teaching, timings --> properties
functions --> take sessions, prepare interview questions, take resume sessions

```
<Student> --> object
	<Firstname>Sivakumar Reddy</Firstname> --> field
	<Lastname>M</Lastname>
	<DOB>2024-01-01</DOB>
	<Addresses> -->object
```

```
        <PermAddress> -->object
            <DoorNo>143</DoorNo>
            <Street>Gandi Nagar</Street>
        </PermAddress>
        <CurrentAddress>
            <DoorNo>143</DoorNo>
            <Street>Gandi Nagar</Street>
        </CurrentAddress>
    </Addresses>
</Student>
```

```
{
    "firstname": "Sivakumar",
    "lastname": "M",
    "dob": "2024-01-01",
    "addresses": [
        {
            "type": "perm",
            "doorno": "143",
            "street": "gandi nagar"
        },
        {
            "type": "perm",
            "doorno": "143",
            "street": "gandi nagar"
        }
    ]
}
```



**Session - 23**

what are ansible adhoc commands?
it is ansible command line to perform one time tasks or emergency tasks when we dont have to write playbook and push

what are ansible playbook?

playbook contains list of plays we can execute againts group of remote servers to perfom multiple tasks. playbook is in YAML format

Playbook contains list of plays and each play can contain list of tasks

dnf install nginx -y -->  homogeneous
apt install nginx -y

```
  - name: Run Nginx
    ansible.builtin.service:
      name: nginx
      state: started
      enabled: yes
```

variables, data types, conditions, loops, error handling, functions

variables hold value, you can use it wherever you want. if you change at one place it will reflected all the places where you are referring, basically DRY principle
dont repeat yourself.

inheritance
=============
great grand father --> 1000 acres --> grand father --> 100 acres (father) --> father 10 acres --> son 1 acre --> Our son 100 acres

1. define same variable in all possible locations
2. run the playbook
3. comment when you understand the preference

data types
==============
int, float, double, decimal, long, char, string, boolean, list, map, etc.

Conditions
==============
when

**Session - 24**

facts == variables

packaga
1. gather facts about target servers
2. if os_family is redht it will run dnf state name  -y
3. if os_family is debain it will run apt state name -y

```
loops
============
for i in {1..100}
do
	echo $i
done

loop:
-
-
```

```
functions == filters
================
command-1 | command-2
```

Ramesh, Suresh, Raheem, Joseph --> String

, --> delimiting

Ramesh
Suresh
Raheem
Joseph

255.255.255.255

300.567.980.789

if there are no modules available
1. develop your module
2. use shell or command module

shell vs command

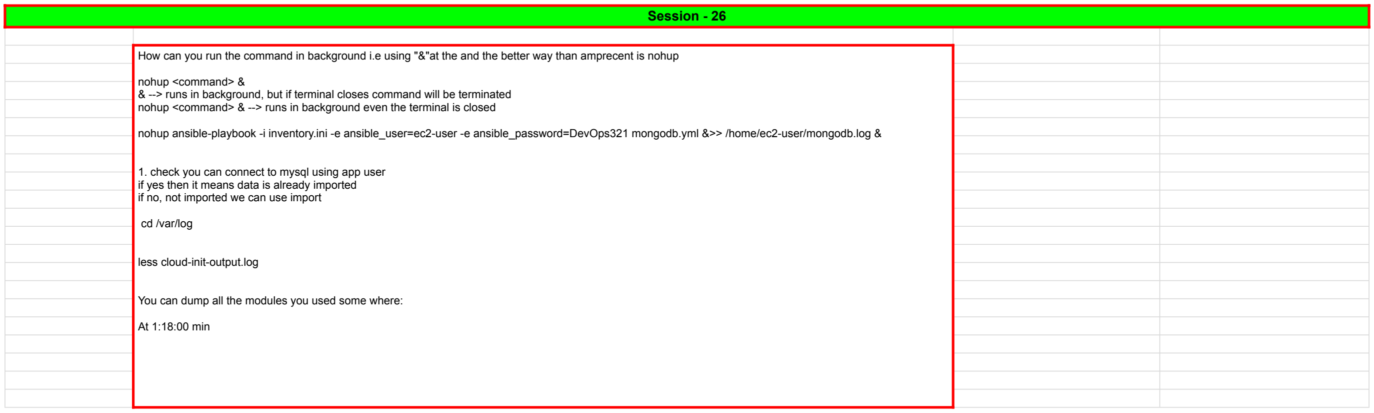
```
you are in your house
----------------------
1. I will enter into your home and call you
	problems: security issue, you are seeing everything inside house
	shell module --> if you are using shell module, you are logging inside the server and running the tasks. redirections, variables, pipes, etc will work here
2. I will stay outside and call you
	problem: you don't get any access to things inside
	command module --> executing command from outside. you will get linux full environment
	redirections, piping and env variables will not work here
```

VARIABLE=$(command)

register

Every time the first preference should be given to command module only and if the command module is not working and it needs some special variables or redirections or pipeing then we will go for shell module.

And usually command module is more secure i.e for simple task we can use command module and for complex tasks including  redirections or pipeing or some special variablewe can go for shell module



**Session - 26**

How can you run the command in background i.e using "&"at the and the better way than amprecent is nohup

```
nohup <command> &
& --> runs in background, but if terminal closes command will be terminated
nohup <command> & --> runs in background even the terminal is closed

nohup ansible-playbook -i inventory.ini -e ansible_user=ec2-user -e ansible_password=DevOps321 mongodb.yml &>> /home/ec2-user/mongodb.log &
```

1. check you can connect to mysql using app user
if yes then it means data is already imported
if no, not imported we can use import

```
 cd /var/log

less cloud-init-output.log
```

You can dump all the modules you used some where:

At 1:18:00 min

**Session - 27**

Ansible Roles
===============
DRY
Variables
Functions

create system user
install programming language
download code
create directory
unzip code
install dependencies
create systemctl service
start the service

Roles is a proper directory structure to write ansible playbooks, we can reuse roles. it will have tasks, vars, templates, files, handlers, etc.

```
{"changed": false, "msg": "Could not find or access 'mongo.repo'\nSearched in:\n\t/home/ec2-user/ansible-roboshop-roles/roles/mongodb/files/mongo.
repo\n\t/home/ec2-user/ansible-roboshop-roles/roles/mongodb/mongo.repo\n\t/home/ec2-user/ansible-roboshop-roles/roles/mongodb/tasks/files/mongo.
repo\n\t/home/ec2-user/ansible-roboshop-roles/roles/mongodb/tasks/mongo.repo\n\t/home/ec2-user/ansible-roboshop-roles/files/mongo.repo\n\t/home/ec2-
user/ansible-roboshop-roles/mongo.repo on the Ansible Controller.\nIf you are using a module and expect the file to exist on the remote, see the remote_src
option"}
```

template --> we can keep some placeholders, you can put the values at run time

file --> content inside a specific file

ansible templates
=================
follows jinja2 formatting, we can keep some placeholders, actual values will be provided through variables at runtime.

tasks
main.yaml --> playbook related tasks are her
files
<file-name> --> you can keep all the files required here
templates
<template-file> --> we can keep all the templates with placeholders here. usually we follow jinja2 templating, variables values can be supplied
vars
main.yaml --> all variables required for roles can be kept here.
handlers --> handlers are notifiers in ansible. when there is a change in something if you want to notify other task we can use handlers. for example change in nginx configure can notify restart nginx task in handlers
main.yaml

DB Admin
==========
1. App team requests DB admin team to create schema
2. DB team will create a seperate shipping user and handover that to app team.
3. shipping user will have only access to shipping schema

**Session - 28**

Ansible Tags
=============
deployment or release new version

at 18:50 IMP

**include_role:**

```
- name: Deployment for catalogue
  tags:
    - deployment
  include_role:
    name: common
    tasks_from: deployment
```

**include_role: When condition will be applieed for only the app-setup task and not to full common roles included tasks ->> and also tags will not work in include roles -> At 47:31 IMP**

import_role: When condition will be applied for all the tasks in inthe entire role i.e all the tasks in common role -------------> **At 43:15 IMP -> At 47:51 IMP i.e Import role throughs syntax errors as soon as ansible-playbook command is fired without executing any task.**

**49:46 IMP**

**Use import_role when:**
**You know exactly which role you want to load before playbook execution.**
**You don't need to loop or conditionally load roles.**
**Use include_role when:**
**You need to conditionally include a role using when.**
**You want to loop over roles or pass different parameters at runtime.**

announce downtime 6hrs
--------------------
stop catalogue service
remove old code -> I can remove /app directory, create again and download new version code
download new code
restart catalogue

We can include other roles using include_role or import_role

include_role --> it will include tasks in run time, it will validate also in run time, it will not pre process.

Tags and when conditions apply only to the include_role statement itself, not the tasks within the role.

import_role --> ansible validate import_role before playbook execution
Tags and when conditions apply to the imported role and its tasks.

**ansible error handling**
**======================**
**Error handling means what should we do when error comes, what should we do when error not comes.**

**if we are able to handle the error, we can keep ignore_errors true and then execute another task in case of failure**

```
id roboshop
        if created already, we are skipping
        otherwise we are creating
```

imagine, there is no user module available

a task executes id roboshop -> task fails if user not available, usually script exit because of this error

another task create user, if rc != 0

vault
============
you can create a file and encrypt it..

Ansible Vault
===================
This is a feature of ansible which allows us to protect the playbooks
via a password. Playbooks created using vault can be viewed,edited or
executed only if we know the password

1 To create a vault playbook
```
  ansible-vault create playbook_name.yml
```

2 To view the content of a vault playbook
```
  ansible-vault view playbook_name.yml
```

3 To edit the content of a vault playbook
```
  ansible-vault edit playbook_name.yml
```

4 To convert an ordinary playbook into a vault playbook
```
  ansible-vault encrypt playbook_name.yml
```

5 To convert a vault playbook into an ordinary playbook
```
  ansible-vault decrypt playbook_name.yml
```

6 To reset the password of a vault playbook
```
  ansible-vault rekey playbook_name.yml
```

```
ansible-playbook -i inventory.ini mysql.yml -e "component=mysql" --ask-vault-password
```

It will ask for the password in run time: --ask-vault-password

SSM parameter store

Linux --> Server, Ansible --> Platform
HashiCorp Vault --> Install, manage, update, upgrade

Linux/ansible integrate with vault softwares

We were using ansible vault, but there are challenges to maintain the passwords and vaut files, so I recently proposed SSM parameter store, I did POC and Removed vault and implemented SSM parameter store successfully...

## Session - 29

Ansible dynamic inventory
========================
inventory --> static -> hosts are fixed

inventory --> Dynamic -> hosts are not fixed i.e example IPL match hotstar and also like flipkart and amazon on big billionares day

AutoScaling --> Dynamic Inventory -> when traffic increases, infra also increases. when traffic decreases infra also decreases
100 servers

ansible queries the ec2 instances based on some criteria...

boto3 and botocore

us-east-1 region
we need to filter the instances
    running instances
    Name --> Frontend

```
 ansible-inventory -i inventory.aws_ec2.yml --graph
```

*.aws_ec2.yml/yaml

forks --> How many number of instances ansible can connected

ansible.cfg

1. ANSIBLE_CONFIG - 22:26 IMP
2. ansible.cfg in pwd
3. home directory
4. etc

Beter option is in which ever the current repo or current working directory we are it is recommnded to have ansible.conf in that particular directory

What is Ansible Dynamic inventory:- Ansible can query the cloud provider to fetch hosts at run time, this will be useful in the autoscaling kind of dynamic environments.

**export** keyword to setup the environmental variable - **22:26 IMP**

**unset**

```
- name: install nginx
  hosts: dynamic_frontend
  serial: 3
  become: yes
```

Ansible cannot under the state i.e it cannot handle the infrasturucture perfectly which is it cannnot under stand the state perfectly

Why we don't use ansible for infra creation:

Ansible is very poor in **state management and this state managment is perfect in Terraorm**

Ansible can create aws infra also...

Everything is module/collection. Ansible may require libraries. to connect AWS we need boto3 and botocore. aws configure authentication too..

1. create instances
2. create r53 records

ec2_output.results

"{{item.instances[0].private_ip_address}}"

**ssh-keygen -t rsa -f <file_name>**

**Session - 30**

Ansible architecture
Configuration management
Variables --> prefernce
Conditions
Loops
Functions or Filters
Data types
Developed playbooks
Roles
Tags, Dynamic inventory, vault, using SSM parameter store, handlers, include vs import, etc...
Deployments through Ansible

roboshop --> create user --> nodejs

frontend --> HTMl, CSS, JS --> name, email, password

a function in nodejs

```
addUser(name, email, password){
    isUserAlreadyExist
    if not available createUser
    sendResponseToFrontend
}
```

RoboShop Createuser through Ansible

custom modules/collections
===============
library/          # roles can also include custom modules
write python code

```
myroboshop.createUser
    name:
    email:
    password:
```

Ansible executes modules/collections on remote server

plugins: lookup('amazon.aws_ssm','name', region: '') --> ansible server
plugins adds extra functionality to ansible server. like lookup, inventory, filter, etc...

```
myroboshop.addCart
    product_name:
    product_id:
```

you want to check the price before adding to cart

lookup('roboshop.checkproduct', product_id: '1234')

filter: roboshop-price | roboshopINRprice

Collections or Modules / Plugin: Collectionswill be executed on the remote host, Plugins add extra functionality to ansible server like lookup , inventory , filters etc

ansible is in very poor in state management, for example if someone edits infra manually in console ansible can't detect that it may create duplicate resources if you run again...

ansible is perfect doing configurations with in the server...

manually created infra
=====================
time
errors
if something goes wrong, tracking is very difficult
restoring the infra is time taking..
version control

Advantages of Infra as a code
=================
version control --> versioning of our code, review, history of infra, collobaration
CRUD --> operations on infra is easy creating, updating, reading and deleting
consistent infra -> IaaC can setup same infra across all environments, basically working in DEV not working in PROD
inventory management --> we can check IaaC to understand what are the services we are using
cost --> switch off the infra in non business hours, switch on again in business hours... we can schedule to stop the servers at 8PM start again at 8AM
dependency management --> IaaC can understand the dependencies between resources..
Modules --> we can extend terraform code using modules

terraform init -->

HCL --> Hashicorp configuration language

```
{
    key = value
}
```

everything is called as resource

```
resource "type-of-resource" "name-you-provide" {

    ami_id =
    name =
    sg_id =
    instance_type =
    subnet_id =
}
```

**Session - 31**

you can issue/fire terraform command only where you have .tf files

command-1 | command-2

register: <variable>

if you are creating a resource, it gives us o/p using this we can create other resources.

variables datatypes conditions loops functions error handling

**Variables** ----------------> Variables are based on dry principal i.e based on Don't Repeat Yourself, You can create some variable and assign a value to that and you can use it where ever you wantt, It is a central place if you change the value it can refelefted every where you are reffereing to

```
============
variables file
terraform.tfvars --> same filename for terraform to consider
command line
env variables TF_VAR_<variable-name>               -> export TF_VAR_sg_name=env-allow-all
```

```
terraform paln -var "sg_name=cmd-allow-all"
```

```
unset TF_VAR_sg_name=env-allow-all
```

1. command line
2. tfvars
3. env variables
4. default values
5. user prompt -> 39:00 min

```
Conditions
==================
if(expression){
      these statements if expression is true
}
else{
      these statements if expression is false
}
```

```
expression ? "this value if true" : "this will run if false"
```

Generally in Development environment we run small instaces and if it is a prod we will run with big instances.

```
loops
===================
1. count based loops
2. for loops
3. dynamic block
```

**you can write count of resources you require under the resource definition**

```
interpolation   1:03 IMP            ->   name    = "${var.instances[count.index]}.${var.domain_name}"    -> If you want to concat string along with variable
=============
```

1. if you have list, go for count
2. if you have map or set, go for for each

**Session - 32**

"${}-hello"

```
dynamic blocks
================
dynamic ingress {
        for_each = var.ingress_ports
        content {
                from_port        = ingress.value["from_port"]
        to_port          = ingress.value["to_port"]
        protocol         = "-1"
        cidr_blocks      = var.cidr_blocks
        ipv6_cidr_blocks = ["::/0"]
        }
}
```

**Block: AT 04:45 sec IMP**

```
  ingress {
        from_port        = var.from_port
        to_port          = var.to_port
        protocol         = "-1"
        cidr_blocks      = var.cidr_blocks
        ipv6_cidr_blocks = ["::/0"]
    }
```

```
functions
===============
length
toset
```

labels == tags

```
100 bags --> white
tag --> details
name = siva
flight = IN307
dest = dubai
```

```
common_tags
==============
Project = roboshop
Terraform = true
Version="1.0"
```

```
variable tags
==============
Component = cart
Name = cart
Version = "1.1"
```

```
merging merge(common_tags, variable tags)
=========
Project = roboshop
Terraform = true
Component = cart
Name = cart
Version = 1.1
```

```
data sources   -> data sources are quering the information from providers
=================
variables --> inputs
outputs --> print the info after creating resources
```

it can query the info from provider

What is variable: Key = Value pair but if the value is dynamic -> There we can use locals

```
locals
=================
```

locals can have expressions, you can assign a name to it and use it wherever you require

locals are like variables holding values against keys, but you can refer variables inside locals, expressions, functions..

calling variable in the same variable file in some other exepression like interpolation will not work

i.e we can refer variables in variables

**variables can be overridden, locals can't be overridden**

At 53:27 IMP

Which ever can be overwritten we can put in variable and which ever can't be over written can be put in locals

```
state
==================
```

IaaC --> declarative way of creating infra, whatever you declare IaaC tool should create provided you follow right syntax

.tf files --> desired/declared infra --> expectation
what exists in AWS --> actual infra --> reality

state files --> terraform use this file to track what it created in provider

```
terraform plan
==================
reads .tf files --> understand what user wants
read state file --> empty
query the provider --> already infra exists or not
```

it starts create

created infra

terraform plan
reads .tf files, state file --> matched

I deleted instance in console manually
reads .tf files, state file --> matched
checks provider to verify desired infra vs actual infra

```
when you change tf code
======================
.tf files --> understands what user wants
state file --> not matched
```

actual infra --> user don't want r53 records.

AT 1:17 hr IMP  (Apply terrafrom apply --auto-approve then delete/ comment someting from .tf files and then fire terraform plan then terrform will destory the resource the that is commented)

terraform uses state file to track what it is created in the provider, every time we run terraform commands terraform check whether desired infra is matching actual infra through state file...

keeping state file in local will not work in collobarative environment. terraform does not understand what were the resources created by others, so it may create duplicate resources or else errors

AT 1:27 Hr IMP

**What is terraform state file: Terraform uses state file to track what is created in the provider, Every time when you run terraform commands i.e terraform check weather the desired infra is matching the actual infra or not through state file.**

**So keeping the state file in local is not good in collabortive environment it may give you duplicate entries , resources or erros , So we should maintain the state file in remote environment like s3 buckets with locking.**

**At any time we should not edit state file as it is allowed to control only by terraform.**

## Session - 33

In Interviews: Before we are using dynamodb locking, But recently I migrated it to S3 native locking

Provisioning / Server Provisining mean in general we are creating a server and install/configure all our required packages inside server

Terraform provisioners are built-in features that allow you to execute scripts or commands on a local or remote machine as part of resource creation or destruction

Provisioners
==============
when you create the server using terraform, we can take some actions using provisioners
1. local-exec
2. remote-exec

provisioners on_failure = continue

where I run terraform command, that is local to terraform
remote means server I created using terraform

creation time provisioners

destory time provisioners

multiple infra using terraform
==============================
DEV, QA, SIT, UAT, PRE-PROD, PERF, PROD

daws84s.site

**AWS EC2 Instances:**
roboshop-dev-mongodb
roboshop-dev-redis
roboshop-dev-mysql
roboshop-dev-rabbitmq

roboshop-uat-mongodb

roboshop-prod-mongodb

**Records:**
mongodb-dev.daws84s.site
mongodb-prod.daws84s.site

dev and prod

pros
======
no need to duplicate the code
consistency

cons
======
should be very cautious changes done in DEV may go to PROD also by mistake

```
terraform init --help

terraform init -backend-config=dev/backend.tf

terraform plan --help

terraform paln -var-file=dev/dev.tfvars

terraform apply -var-file=dev/dev.tfvars

terraform init --reconfigure -backend-config=prod/backend.tf

terraform apply -var-file=prod/prod.tfvars
```

**ma**

```
terraform init -backend-config=dev/backend.tf

terraform paln -var-file=dev/dev.tfvars

terraform apply -var-file=dev/dev.tfvars -auto-approve

terraform destroy -var-file=dev/dev.tfvars -auto-approve

==================
```

All backend bucket is pointing to dev and hence again reconfiguring it to prod

```
terraform init --reconfigure -backend-config=prod/backend.tf

terraform apply -var-file=prod/prod.tfvars


=======================
```

Workspaces:
------------------------

```
terrafrom workspace


terraform workspace
Usage: terraform [global options] workspace

  new, list, show, select and delete Terraform workspaces.

Subcommands:
    delete    Delete a workspace
    list      List Workspaces
    new       Create a new workspace
    select    Select a workspace
    show      Show the name of the current workspace


--------------

terraform workspace new dev


terraform workspace list


terraform workspace select dev

terraform.workspace = dev/prod    -> At 16:18 sec

lookup(map, key)  --> dev

lookup(map, qa, "t3.micro")


terrafrom workspace select prod
```