Linux Assigment-1

Task-1

File and Directory Operations

1. Create a directory structure as follows:

```
-> mkdir project
   Cd project
   Mkdir scripts logs data
```

2. Create three text files: data1.txt, data2.txt, and data3.txt in the data/ folder. Populate them with random text using the echo or seq command.

```
-> cd data
   Seq 1 10 > data1.txt
   Seq 11 20 > data2.txt
   Seq 21 30 > data3.txt
```

3. Combine all three files into one file named combined.txt using a command.

```
-> cat data1.txt data2.txt data3.txt >> combined.txt
```

4. Move all .txt files from data/ to logs/ that are larger than 100 bytes using find and mv.

```
-> find project/data/ -type f -name "*.txt" -size +100c -exec mv {} project/logs/ \;
OR
find project/data/ -type f -name "*.txt" -size +100c | xargs -I {} mv {} project/logs/
```

Task-2

Create a user named deploybot with:

· /bin/bash as the shell

· No login privileges (hint: /usr/sbin/nologin)

· A home directory under /opt/deploybot

· Add deploybot to a new group named devops.

· Set file permissions so that only users in the devops group can read and write to /opt/devops-data

```
->  sudo groupadd devops

->  useradd –d /opt/deploybot -s /bin/bash -G devops deploybot

->  usermod –s /sbin/nologin deploybot

-> touch /opt/devops-data

-> chown :devops /opt/devops-data

-> chmod 660 /opt/devops-data
```

Task-3

User & Permission Management

Create two new users: devuser and testuser.

· Set passwords for each user (use passwd, no need to share them).

· Create a shared directory /shared and:

· Give read/write permission to devuser only.

· Ensure testuser cannot access it. Use chmod, chown, and chgrp where needed

```
-> useradd devuser

-> passwd devuser

-> useradd testuser

-> passwd testuser

-> mkdir shared

-> chown devuser shared

-> chmod 600 shared
```

Task-4 Process and System Monitoring

Use ps, top, or htop to list the top 5 memory-consuming processes. Save the output to top_memory.txt.

Write a one-liner command that shows:

· Current disk usage (df -h)

· And appends it to disk_status.log with the current timestamp.

```
-> ps aux | sort -k 4 -nr | head -n 6 > top_memory.txt
```

or

```
-> ps aux --sort=-%mem | head -n 6 >top_memory.txt
```

```
-> date +"%Y-%m-%d %H:%M:%S" >> disk_status.log && df -h >> disk_status.log
```

Task-5

Shell Scripting & Automation

Write a script service_checker.sh that:

· Takes a list of services (e.g., nginx, docker) as arguments

· Checks if each service is active

· Logs status in /var/log/service_status.log with a timestamp Set up a cron job to run this script every 15 minutes.

-> service_checker.sh

```
#!/bin/sh

log_file="/var/log/service_status.log"
timestamp=$(date "+%Y-%m-%d %H:%M:%S")
```

```
#!/bin/bash

log_file="/var/log/service_status.log"
```

```
timestamp=$(date +"%Y-%m-%d %H:%M:%S")



for service in "$@";
do
    if systemctl is-active "$service";
    then
        echo "$timestamp - $service is Active" >> "$log_file"
    else

        echo "$timestamp - $service is Inactive" >> "$log_file"
    fi
done
```

-> Crontab –e

Add this line

```
*/15 * * * * /service_checker.sh ngnix docker ssh
```