**Automated Monitoring and Maintenance for an E-commerce Website**

**Problem Statement:**

An e-commerce company runs its website on a Linux-based web server. The company faces challenges in maintaining the server's health and ensuring the website is always up and running.

Manual monitoring and maintenance tasks are time-consuming and prone to human error. The company needs an automated solution to monitor system performance, perform regular updates, and back up critical data to ensure continuous availability and reliability of the website

**Step by step Implementation:**

**1) Virtualized Environment Setup:**

1.creating the new virtual machine : We can use the oracle

VM VirtualBox

VM Download link:

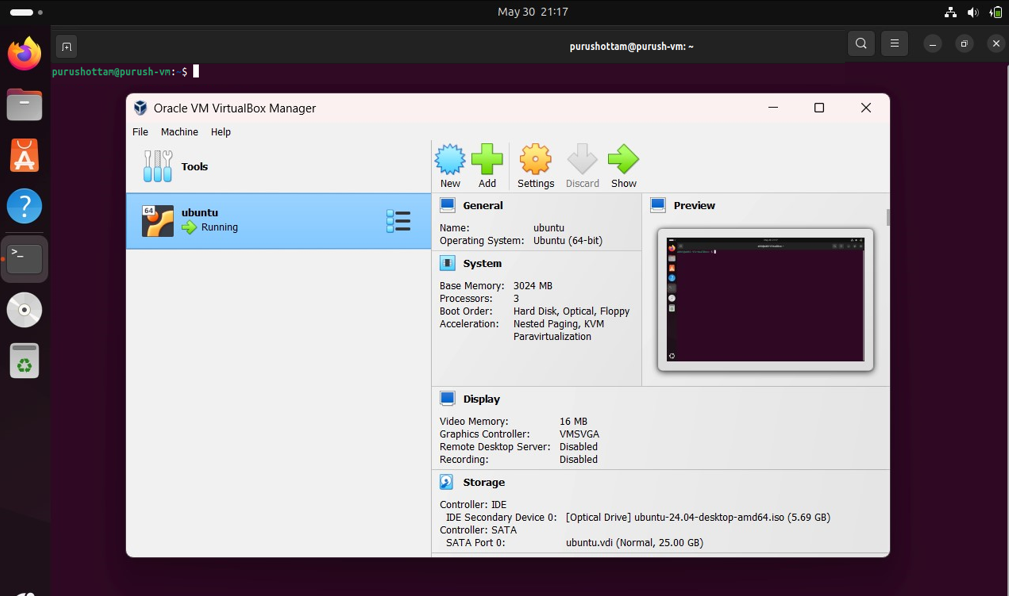
<https://www.virtualbox.org/wiki/Downloads>

2.ISO image download link:

[https://ubuntu.com/download/desktop/thank-you?versio n=24.04&architecture=amd64&lts=true](https://ubuntu.com/download/desktop/thank-you?version=24.04&architecture=amd64&lts=true)

3. VM Configuration as:

* + Type: Linux
  + Version: Ubuntu 64-bit
  + Memory: 3024 MB ● Virtual Hard Disk: 30 GB



3. Setup the Ubuntu Server:

* Update and upgrade the package manager

Sudo apt update && sudo apt upgrade -y

* Install apache server:

sudo apt install apache2 -y

* Start apache server:

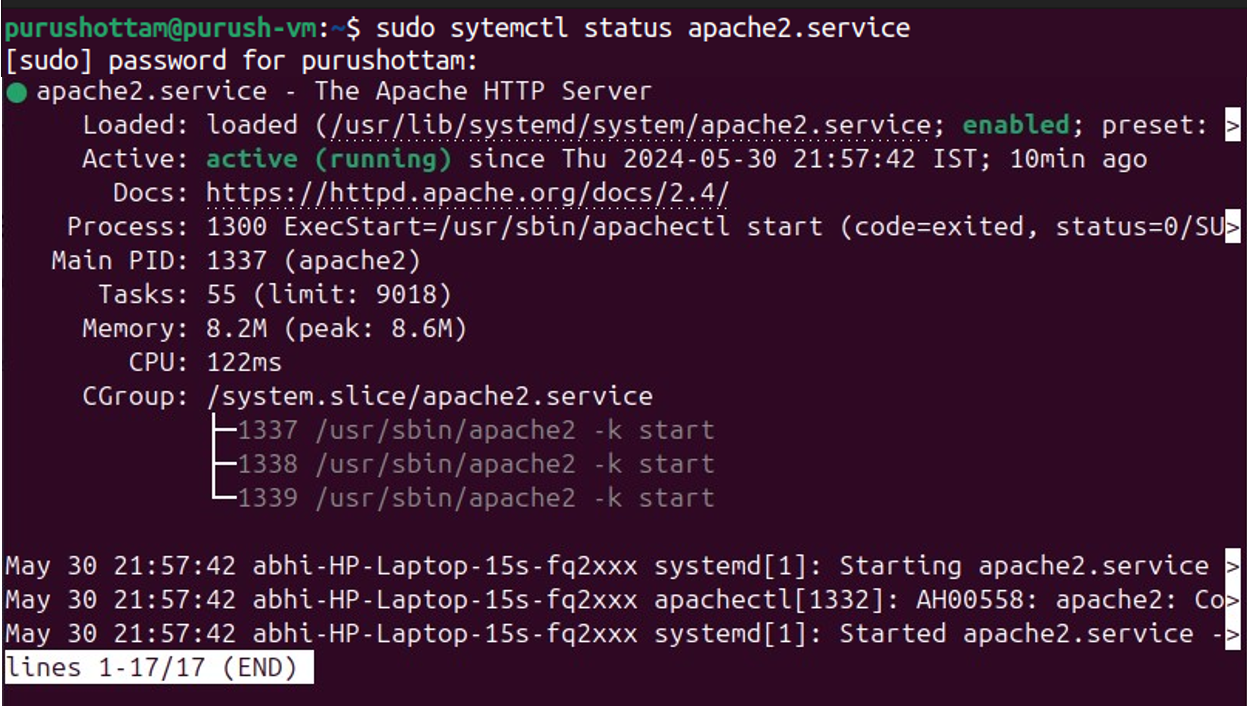
sudo systemctl start apache2

* Enabled apache server:

# sudo systemctl enable apache2

● Check the status of the apache server:

**Sudo systemctl status apache2**.service



1. **Monitoring System Performance:**

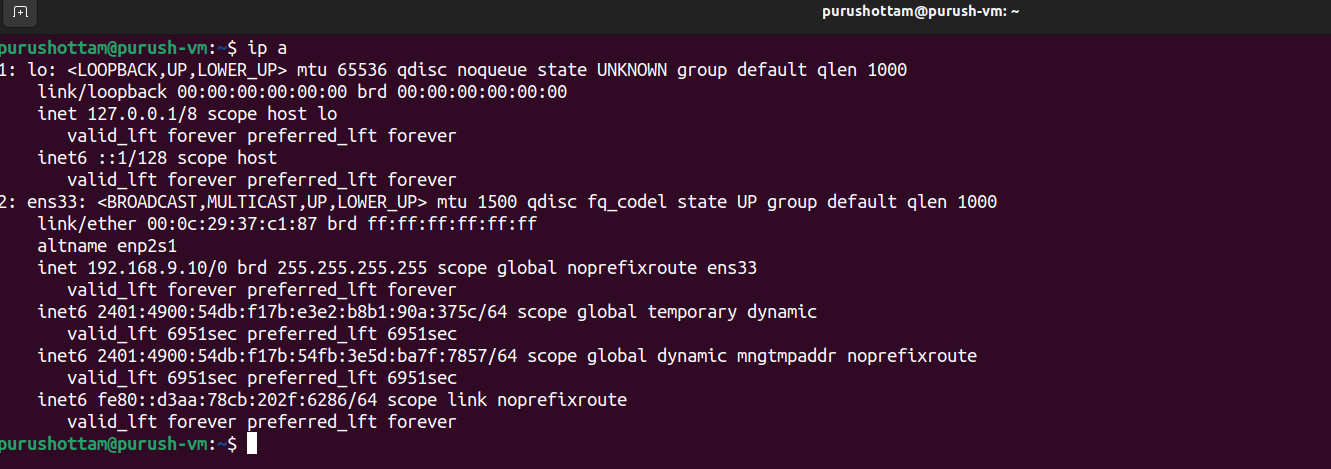
**1.** The top command in Linux is a powerful tool for monitoring system activity in real-time

* + - **Process Information**: top lists all running processes along with their process ID (PID), user, CPU usage, memory usage, and more.
    - **CPU Usage**: top displays a summary of CPU usage at the top, showing the total CPU usage, usage breakdown by user processes, system processes, and idle CPU time.
    - **Memory Usage**: Along with CPU usage, top also provides information about memory usage, including total memory, used memory, free memory, and memory usage by individual processes
  1. Check disk usage: It use to check the memory uses of the system

**df -h**

* 1. Monitor network activity

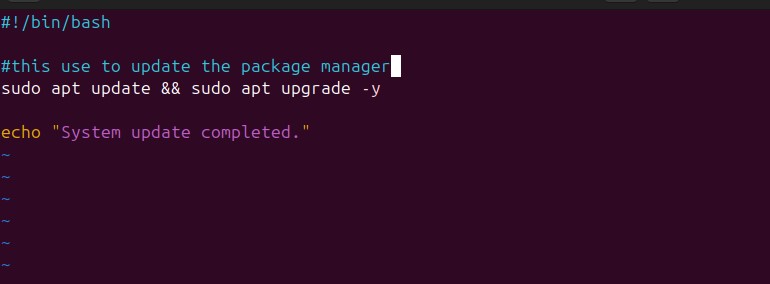
|  |  |
| --- | --- |
| **ip a** | **—---> Get the network info** |



**netstat -tuln —---> Displays information about active network connections and listening ports.**

1. **Automate System Updates:**

1. Create an Update Script:



In this script we use to update the package manager

# Sudo apt update && sudo apt upgrade -y

1. Make the script executable: Using this command we can give the execute permission to the all user to execute the script

**chmod** +**x update\_system**.sh

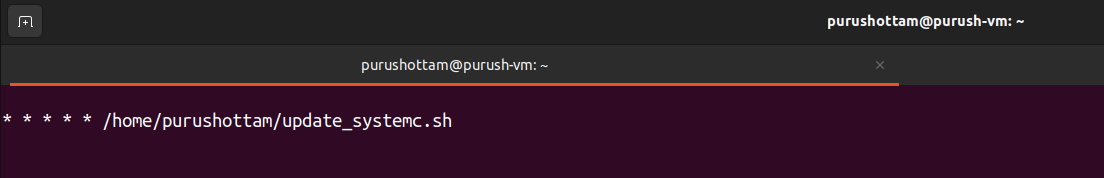
1. Schedule the Script: Create the cron job to run this script on every 2AM

**Crontab -e**

# 0 2 \* \* \* /home/purushottam/update\_system.sh

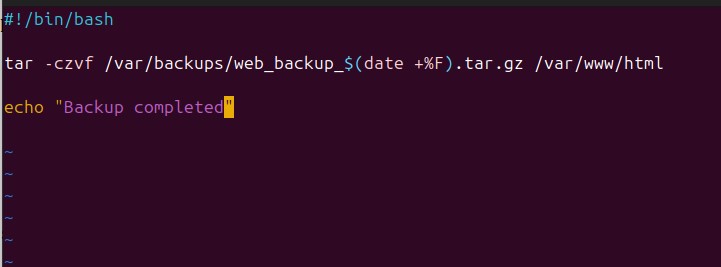
0: This field specifies the minute when the command will run. In this case, 0 means the command will run at the start of the hour (2:00 AM).

2: This field specifies the hour when the command will run. In this case, 2 means the command will run at 2:00 AM.

\*: The asterisks in the next three fields represent wildcard characters, meaning "every" or "any". So \* \* \* specifies "every day of the month", "every month", and "every day of the week".

**4) Automate Data Backup:**

1. Create a Backup Script:



In this script we can backup the /var/www/html directory content , it contains our website file.

tar -czvf /var/backups/web\_backup\_$(date +%F).tar.gz /var/www/html

**tar**: This is the command-line utility used for archiving files. It can create, view, and manipulate tar archives.

**-czvf**: These are options passed to tar with specific meanings:

* -c: Create a new archive.
* -z: Compress the archive using gzip.
* -v: Verbose mode, which displays progress and filenames as the archive is created.
* -f: Specifies the filename of the archive.

**/var/backups/web\_backup\_$(date +%F).tar.gz**: This is

the path and filename of the resulting archive. It's constructed dynamically using the date command:

* $(date +%F): This part of the command substitutes the output of the date command, which generates the current date in the format YYYY-MM-DD (%F).

1. Make the script executable: using chmod we can provide the execution permission.

**chmod** +**x backup**.sh

1. Schedule the Script: schedule the backup script weekly on

Sundays at 3 AM

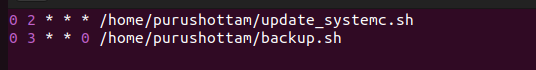
**crontab -e**

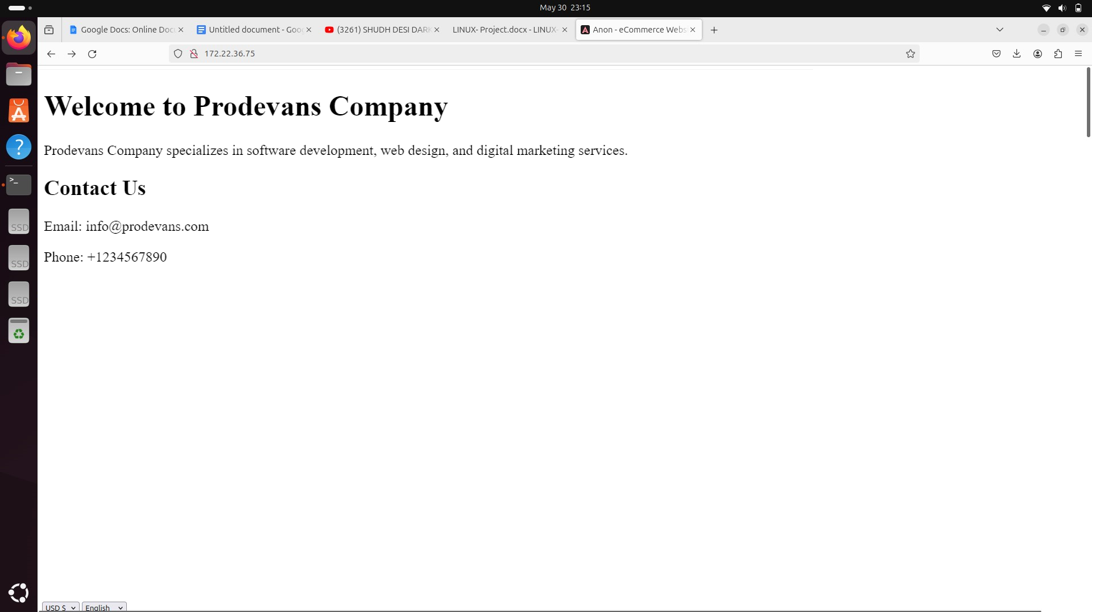
# 0 3 \* \* 0 /home/purushottam/backup.sh

0: This field specifies the minute when the command will run.

Here, 0 means the command will run at the beginning of the hour. 3: This field specifies the hour when the command will run. In this case, 3 means the command will run at 3:00 AM.

\*: The asterisks in the next three fields represent wildcard characters, meaning "every" or "any". So \* \* \* specifies "every day of the month", "every month", and "every day of the week". 0: This field specifies the day of the week when the command will run. In cron notation, Sunday is represented by 0



Website that host on the apache server: