PROJECT NAME –

Shell Scripting Automation

Project Report

Submitted By:

Name: **Karandeep**
UID: **23BCA10034**
Class: **23BCA2A**
Subject: **Linux Administration Lab**
Subject Code: **23CAP-305**

Submitted To:

Name: **Mr. Rajat Patial**
Designation: **Assistant Professor**

Project Title: Shell Scripting Automation for System Monitoring

Aim

To develop a **Bash shell script** that automatically displays system information such as uptime, CPU, memory, disk usage, and logged-in users — providing an automated, lightweight way to monitor system performance.

Objective

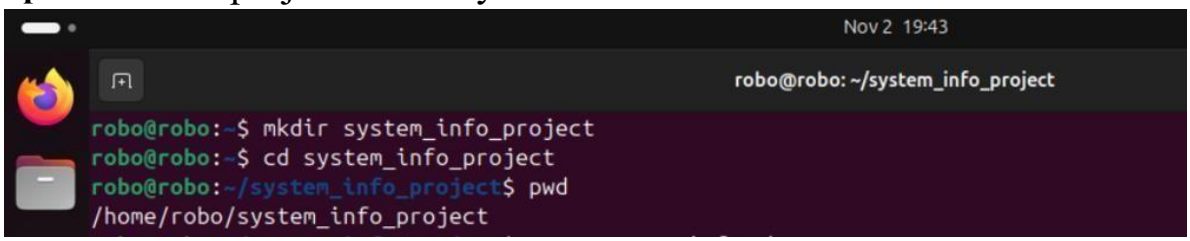
- To understand how to use Linux shell scripting for system automation.
- To create a simple monitoring script that provides important system metrics.
- To automate routine administrative tasks like resource checking.
- To demonstrate the use of core Linux commands for automation purposes.

Commands Used and Their Purpose

Command	**Purpose / Description**
#!/bin/bash	Specifies the script interpreter (Bash).
echo	Prints formatted text or messages.
uptime -p	Shows how long the system has been running.
lscpu	Displays information about the CPU architecture.
grep "Model name"	Extracts the CPU model name from lscpu output.
who	Lists all users currently logged into the system.
date	Displays the current system date and time.
chmod +x	Grants execute permission to the script file.
./system_info.sh	Runs the script from the current directory.

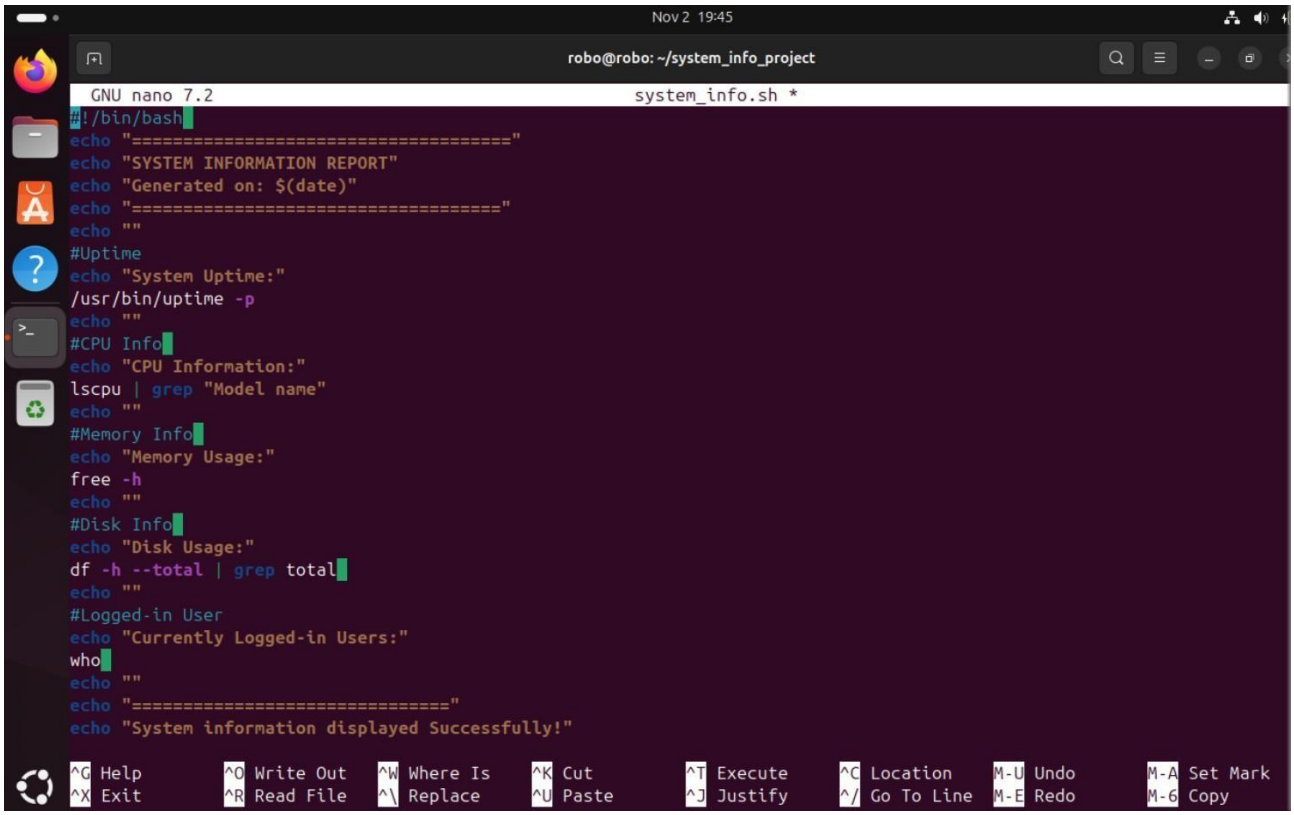
Working and Implementation Steps

Step 1: Create a project directory



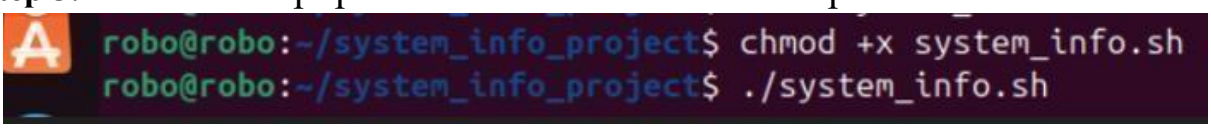
Step 2: Create and open the script file

Step 3: Write the following code:

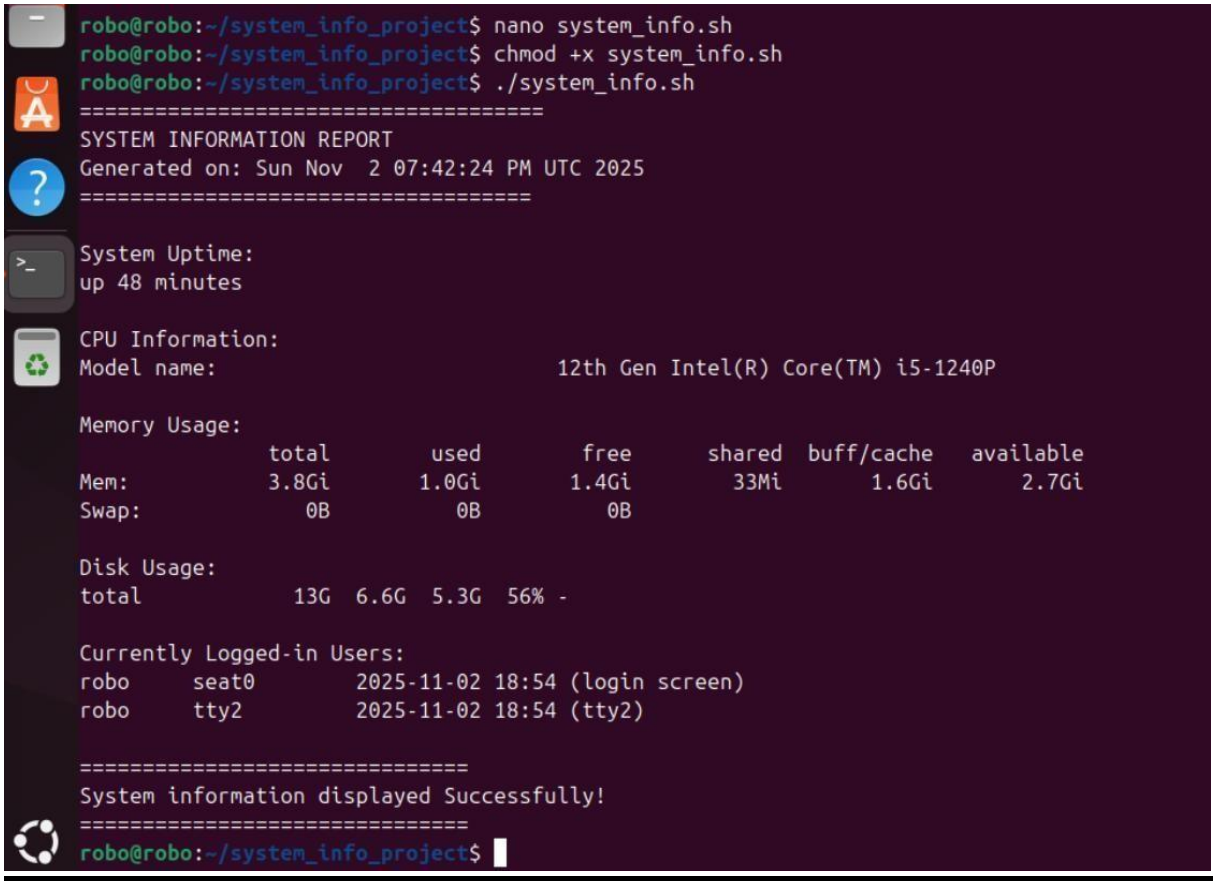


Step 4: Save and exit Nano (Ctrl + O, Enter, Ctrl + X).

Step 5: Give the script permission to run. Run the script



Output



Advantages

Advantage	Description
Automation	Reduces repetitive manual checks by automating system info retrieval.
Simplicity	Easy to write and execute — perfect for beginners.
Extensible	Can be easily expanded with more system monitoring features.

No Dependencies	Uses only built-in Linux commands.
Instant Output	Displays essential system details in seconds.

Disadvantages

Disadvantage	Description
Limited Information	Does not include OS or kernel version details.
Text-Only Output	No graphical or visual interface.
Permission Restrictions	Some system data may require elevated privileges.
No Historical Tracking	The script only shows current status unless redirected to a log file.

Use Cases

- System Administrators: To quickly view system uptime, memory, and CPU status.
- Developers: To check system resource availability before deploying or testing.
- Support Teams: For quick diagnosis of performance-related issues.
- Server Maintenance: To monitor Linux server health automatically.

Future Scope

- Redirect output to a log file for daily records.
- Email or notify admin if resource usage exceeds set limits.
- Integrate with cron jobs for scheduled automation.
- Extend functionality for remote servers using SSH.
- Create graphical dashboards using web interfaces or Python integration.

Conclusion

This project successfully demonstrates the use of Bash shell scripting for automating system information retrieval.
By running simple Linux commands in a structured script, system administrators can monitor critical metrics efficiently.

It provides a strong foundation for advanced system automation and DevOps workflows.

Github Link: [GitHub - Karan49557/PBL_linux]