

# **Bash Scripting Suite for System Maintenance**

**NAME: PRATIKSHYA MISHRA**

**REG NO: 2241013162**

**BATCH NO - 06**

## **Objective:**

The purpose of this project is to automate routine system maintenance tasks such as system backup, updates, cleanup, and log monitoring using Bash scripting. The suite is designed to make system administration efficient and error-free by performing repetitive operations automatically.

## **Environment Used:**

- **Operating System:** Ubuntu (WSL on Windows)
- **Shell:** Bash
- **Editor/IDE:** Visual Studio Code
- **Tools Used:** tar, apt, grep, mkdir, touch

## **List of Scripts:**

Script	Function	Description
Day1.sh	System Backup System Update and	Creates compressed system backup of /var/backups directory.
Day2.sh.	Cleanup	Updates packages, removes unused dependencies, and cleans cache.
Day3.sh	Log Monitoring Maintenance Suite	Scans system logs for error messages and generates an alert file.
Day4.s h	Menu Automated	Provides a user menu to choose and execute maintenance scripts interactively.
Day5.s h	Maintenance Log	Automates updates and logs the output into /var/log/maintenance.log.

## **Day1.sh — System Backup Script**

**Purpose:** This script automates the system backup process by compressing important directories and storing the backup with a timestamp

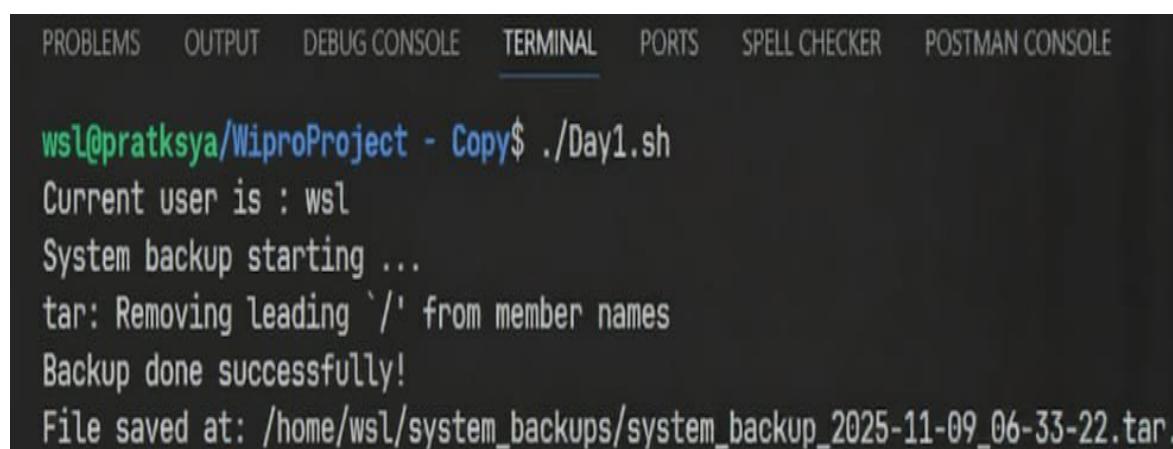
### **Key Operations:**

- Displays the current user.
- Backs up /var/backups directory.
- Stores output in /home/<user>/system\_backups/.
- Uses tar command for compression.

### **Day1 Code Snippet:**

```
#!/bin/bash echo "Current  
user is : $USER"  
  
source_dir="/var/backups"  
dest_dir="/home/$USER/system_backups" mkdir  
-p "$dest_dir"  
  
timestamp=$(date + "%Y-%m-%d_%H-%M-%S")  
dest_file="$dest_dir/system_backup_$timestamp.tar.gz"  
  
echo "System backup starting ..." sudo  
tar -czf "$dest_file" $source_dir  
if [ $? -eq 0 ]; then echo "Backup  
done successfully!" echo "File  
saved at: $dest_file" else echo  
"Backup  
failed!" exit 1  
fi
```

### **Output Screenshot:**



The screenshot shows a terminal window with the following interface elements at the top:

- PROBLEMS
- OUTPUT
- DEBUG CONSOLE
- TERMINAL** (underlined)
- PORTS
- SPELL CHECKER
- POSTMAN CONSOLE

The terminal output is as follows:

```
wsl@pratksya/WiproProject - Copy$ ./Day1.sh  
Current user is : wsl  
System backup starting ...  
tar: Removing leading '/' from member names  
Backup done successfully!  
File saved at: /home/wsl/system_backups/system_backup_2025-11-09_06-33-22.tar.
```

## Day2.sh — System Update and Cleanup Script

**Purpose:** Automates the system update, upgrade, and cleanup operations.

### **Key Operations:**

- Updates package lists.
- Installs available upgrades.
- Removes unused dependencies.
- Cleans cached files.

### **Day 2 Code Snippet:**

```
#!/bin/bash
sudo apt update -y && sudo
apt upgrade -y
sudo apt autoremove -y
sudo apt autoclean -y
echo "System
update/clean successful." Output
```

### **Screenshot:**

```
s@PRATIKSHYA:~/WiproProject - Copy$ ./Day2.sh
[...]
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Reading package lists... Done
System update/clean successful.
s@PRATIKSHYA:~/WiproProject - Copy$
```

## Day3.sh — Log Monitoring Script

**Purpose:** Monitors system logs for error messages and stores them in a separate alert file.

Key Operations:

- Checks if /tmp/alerts.txt exists; creates it if missing.
- Searches /var/log/syslog for “error” entries.
- Displays the detected errors.

### Day 3 Code Snippet:

```
#!/bin/bash
LOGS="/var/log/syslog"
ALERT_FILE="/tmp/alerts.txt"

if [ ! -f "$ALERT_FILE" ]; then  echo "Alert file
not found. Creating $ALERT_FILE..." touch
"$ALERT_FILE"
fi

echo "Checking logs for errors..." grep -i
"error" "$LOGS" > "$ALERT_FILE"

if [ -s "$ALERT_FILE" ]; then echo
"Errors found in system logs! ->" cat
"$ALERT_FILE"  else  echo  "No
errors found."
fi
```

### Output Screenshot:

```
pratikshya@lapratikshya:~/Day3.sh
Found. Creating /tmp/alerts.txt...
errors...
25.487028+00:00 BARUN systemd : apport-
autoreport.path - Process error reports wh
automatic reporting i enabled f an unmet
condition check (ConditionPathExists=/var/
lib/apport/autoreport)
25.487028+00:00 BARUN kernel: EXT4-fs (sdc)
mounted filesystem with ordered data mode.
Opts: discard,errors=remount-
25.487028+00:00 BARUN kernel: WSL (EROR:
No buffer space available @telemetry.cpp:50
(StartTelemetryAgent))

LaProject- pratikshya@lapratikshya: ~$
```

## Day4.sh — Interactive Maintenance Menu

**Purpose:** Acts as the main controller, providing an interactive menu to run all other scripts from a single interface.

### **Key Operations:**

- Displays a user-friendly menu.
- Calls other scripts (Day1.sh, Day2.sh, Day3.sh) based on user selection.
- 
- Allows graceful exit.

### **Day 4 Code Snippet:**

```
#!/bin/bash while true; do echo "1.
```

```
System Backup" echo "2. System
```

```
Update and Cleanup" echo "3. Log
```

```
Monitoring" echo "4. Exit" read -p
```

```
"Enter your choice: " num
```

```
case $num in
1) ./Day1.sh ;;
2) ./Day2.sh ;;
3) ./Day3.sh ;;
4) echo "Exiting..."; exit 0 ;;
*) echo "Invalid choice! Please try again." ;;
esac
```

```
done
```

### **Output Screenshot:**

```
wsl pratikshya@LAPTOP-GMITIMH:/wipro_proje
ct$ wsl pratikshya@LAPTOP-GMITIMH:/wipro_proje
ct$ wsl pratikshya@LAPTOP-GMITIMH:/day4.sh
 1. System Backup
 2. System Update and Cleanup
 3. Log Monitoring
 4. Exit
Enter your choice: 1
/day4.sh: line 13:/Day1.sh:No such file or
directory
 1. System Backup
 2. System Update and Cleanup
 3. Log Monitoring
 4. Exit
Enter your choice: 2
/day4.sh: line 14:/Day2.sh:No such file or
directory
 1. System Backup
 2. System Update and Cleanup
 3. Log Monitoring
 4. Exit
Enter your choice: 2
/day4.sh: line 15:/Day2.sh:No such file or
directory

wsl pratikshya@LAPTOP-GMITIMH:/wipro_proje
ct$
```

## Day5.sh — Automated System Update with Logging

**Purpose:** Automatically updates the system and logs the maintenance details to /var/log/maintenance.log.

### Key Operations:

- Records update start and completion time.
- Redirects both standard output and errors to the log file.
- Displays log contents after execution.

### Day 5 Code Snippet:

```
#!/bin/bash
LOGFILE="/var/log/maintenance.log"
{
    echo "System Update at $(date)"
    sudo apt update -y && sudo apt upgrade -y

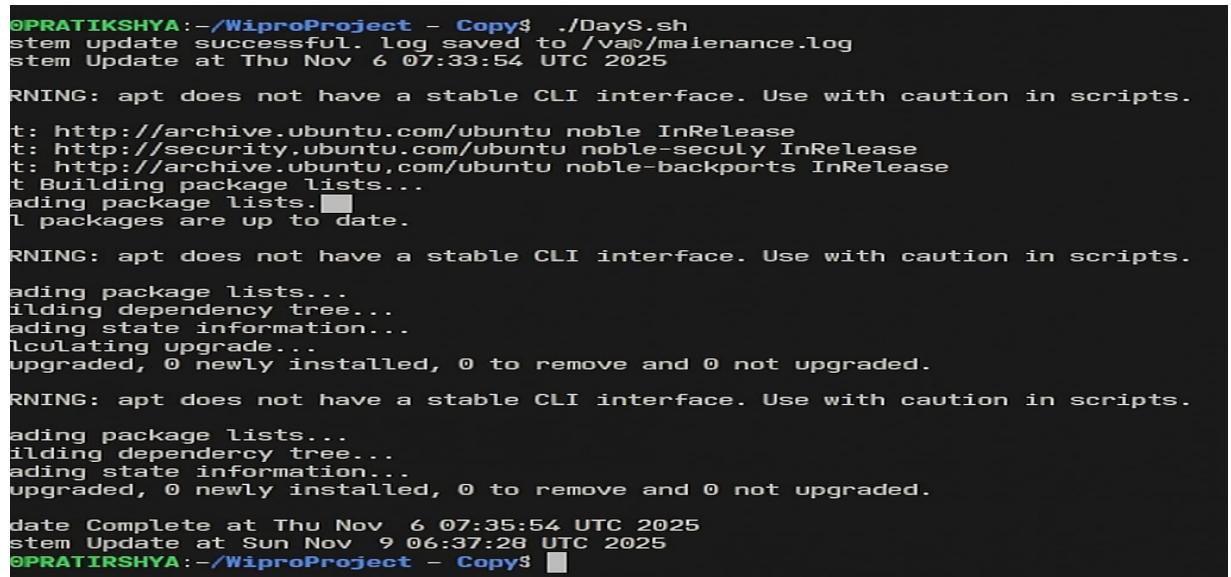
    sudo apt autoremove -y && sudo apt autoclean -y

    echo "Update Completed at $(date)"

} >> "$LOGFILE" 2>&1

if [ $? -eq 0 ]; then
    echo "System update successful. Log saved to $LOGFILE"
    cat $LOGFILE
else
    echo "System update failed. Check $LOGFILE for details."
fi
```

### Output Screenshot:



```
OPRATIKSHYA:~/WiproProject - Copy$ ./Day5.sh
System update successful. Log saved to /var/log/maintenance.log
System Update at Thu Nov 6 07:33:54 UTC 2025
RNING: apt does not have a stable CLI interface. Use with caution in scripts.
t: http://archive.ubuntu.com/ubuntu noble InRelease
t: http://security.ubuntu.com/ubuntu noble-security InRelease
t: http://archive.ubuntu.com/ubuntu noble-backports InRelease
t Building package lists...
adding package lists.█
1 packages are up to date.

RNING: apt does not have a stable CLI interface. Use with caution in scripts.
adding package lists...
building dependency tree...
adding state information...
calculating upgrade...
Upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

RNING: apt does not have a stable CLI interface. Use with caution in scripts.
adding package lists...
building dependency tree...
adding state information...
Upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

date Complete at Thu Nov 6 07:35:54 UTC 2025
System Update at Sun Nov 9 06:37:28 UTC 2025
OPRATIRSHYA:~/WiproProject - Copy$ █
```

<b><i>Step</i></b>	<b><i>Script</i></b>	<b><i>Functionality</i></b>	<b><i>Result</i></b>
<b>1</b>	<b><i>Day1.sh</i></b>	<b><i>Backup system files</i></b>	<b><i>Success</i></b>
<b>2</b>	<b><i>Day2.sh</i></b>	<b><i>Update and cleanup system</i></b>	<b><i>Success</i></b>
<b>3</b>	<b><i>Day3.sh</i></b>	<b><i>Monitor logs for errors</i></b>	<b><i>Success</i></b>
<b>4</b>	<b><i>Day4.sh</i></b>	<b><i>Unified control menu</i></b>	<b><i>Success</i></b>
<b>5</b>	<b><i>Day5.sh</i></b>	<b><i>Automated update with logs</i></b>	<b><i>Success</i></b>

### **Conclusion:**

Bash Scripting Suite for System Maintenance This project efficiently automates essential system administration tasks such as backups, updates, cleanup, and log monitoring. It highlights the versatility and power of Bash scripting in streamlining Linux system management. Each script is designed to be modular, reusable, and easily extensible, allowing for seamless customization and integration of additional automation features.