Capstone Project

Bash Scripting Suite for System Maintenance

OBJECTIVE:
Write a suite of Bash scripts to automate system maintenance tasks such as backup, system updates, and log monitoring

CODE

i. main.sh

```
#!/bin/bash
# --- Configuration ---
# Define the paths to your other scripts.
# If they are in the same directory, this is all you need.
BACKUP_SCRIPT="./backup.sh"
MAINTENANCE_SCRIPT="./maintenance.sh"
LOG_MONITOR_SCRIPT="./log_monitor.sh"
# --- Main Menu Loop ---
while true; do
    clear # Clear the screen for a clean menu
    echo "===================================="
    echo "  System Maintenance Suite Menu"
    echo "===================================="
    echo "1. Run System Backup"
    echo "2. Run System Maintenance (Update & Clean)"
    echo "3. Monitor System Logs (Check for 'Failed')"
    echo "4. Exit"
    echo "------------------------------------"
    echo -n "Enter your choice [1-4]: "

    read choice

    # --- Handle User Choice ---
    case $choice in
        1)
            echo "Running backup script..."
            # Run the backup script
            $BACKUP_SCRIPT
            ;;
        2)
            echo "Running maintenance script (requires sudo)..."
            # Run the maintenance script with sudo
            sudo $MAINTENANCE_SCRIPT
            ;;
        3)
            echo "Running log monitor (requires sudo)..."
            # Run the log monitor script with sudo
            sudo $LOG_MONITOR_SCRIPT
            ;;
        4)
            echo "Exiting suite. Goodbye!"
            break # Exit the while loop
            ;;
        *)
            echo "Invalid choice. Please enter a number between 1 and 4."
            ;;
    esac

    echo ""
echo "Press Enter to return to the menu..."
    read # Pause and wait for user to press Enter
done
```

ii. backup.sh

```
#!/bin/bash
set -e # Exit immediately if a command fails

# --- Configuration ---
SOURCE_DIR="/home/kali/Documents/WIPRO_PROJECT/Capstone_Project/Bash_Scri
pting"
BACKUP_DIR="/home/kali/Documents/Backup/my project"
FILENAME="backup_$(date +'%Y-%m-%d_%H-%M-%S').tar.gz"

# --- Error Handling ---
if [ ! -d "$SOURCE_DIR" ]; then
    echo "ERROR: Source directory $SOURCE_DIR does not exist."
    echo "Backup failed."
    exit 1 # Exit with a failure code
fi

# --- Script Body ---
mkdir -p "$BACKUP_DIR"
echo "Starting backup of $SOURCE_DIR..."
tar -czvf "$BACKUP_DIR/$FILENAME" "$SOURCE_DIR"
echo "Backup complete! File created: $BACKUP_DIR/$FILENAME"
```

iii. maintenance.sh

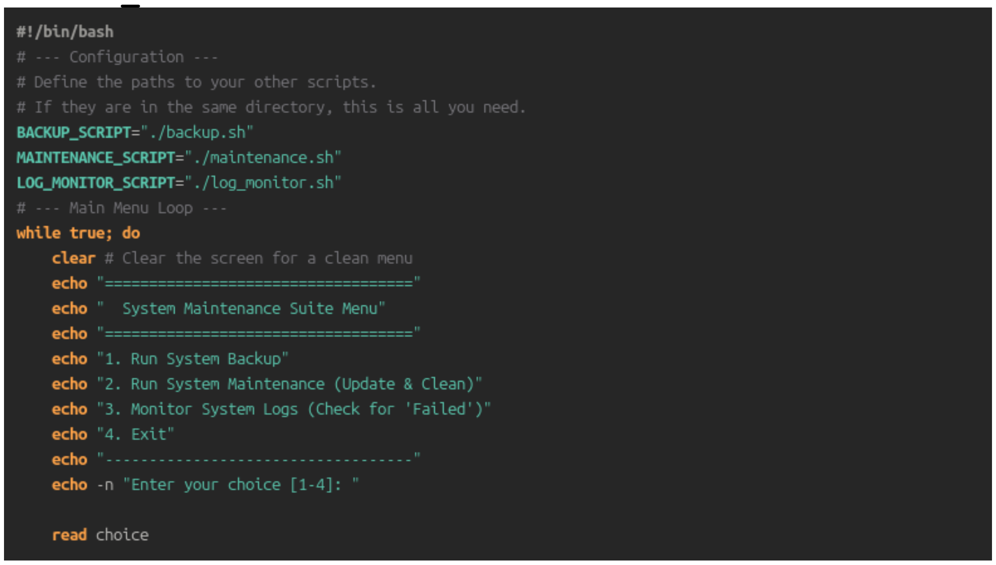
```
#!/bin/bash
set -e     # Exit immediately if a command exits with a non-zero status.
echo "Starting system maintenance..."
# --- 1. Update Package Lists ---
# Fetches the list of available updates from the repositories.
echo "Updating package lists..."
sudo apt update
# --- 2. Perform Upgrades ---
# Installs the newest versions of all packages currently installed.
# The '-y' flag automatically answers 'yes' to all prompts.
echo "Installing system updates..."
sudo apt upgrade -y
# --- 3. Autoremove Old Packages ---
# Removes packages (like old kernels) that were automatically installed
# to satisfy dependencies but are no longer needed.
echo "Removing old and unused packages..."
sudo apt autoremove -y
# --- 4. Clean Package Cache ---
# Clears out the local cache of retrieved package files.
# This frees up disk space, as the .deb files are no longer needed
# After installation.
echo "Cleaning up package cache..."
sudo apt clean
echo "System maintenance complete!"
```

iv. log_monitor.sh

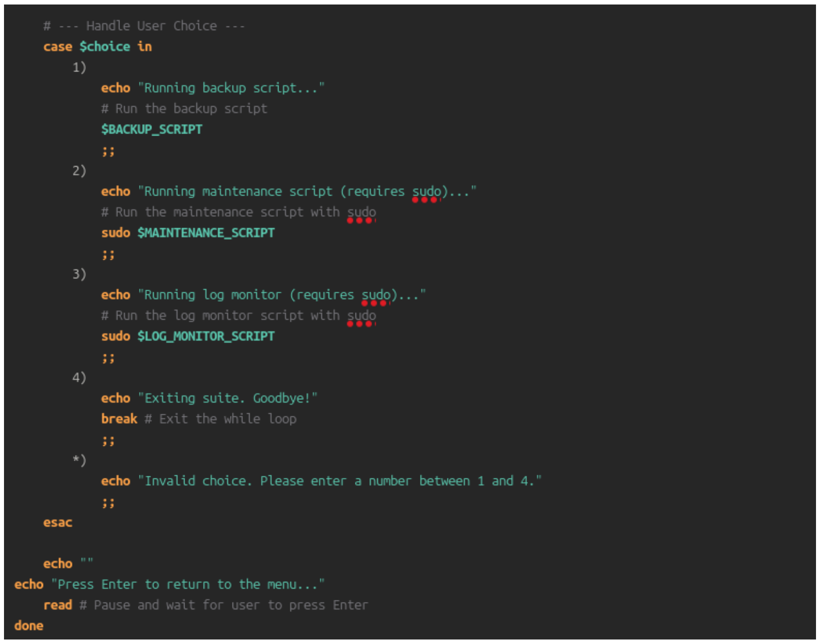
```
#!/bin/bash
set -e   # Exit immediately if a command fails
# --- Configuration ---
LOG_FILE="/var/log/auth.log"
KEYWORD="Failed"
# --- Error Handling ---
if [ ! -r "$LOG_FILE" ]; then
    echo "ERROR: Log file $LOG_FILE does not exist or is not readable."
    echo "Log monitor failed."
    exit 1 # Exit with a failure code
fi
# --- Script Body ---
echo "Starting log scan for '$KEYWORD' in $LOG_FILE..."
MATCHES=$(grep -i -n "$KEYWORD" "$LOG_FILE" || true)

if [ -n "$MATCHES" ]; then
    echo "----------------------------------------"
    echo "ALERT: Potential issues found!"
    echo "----------------------------------------"
    echo "$MATCHES"
else
    echo "Scan complete. No issues found."
fi
```

SCREENSHOTS

i. main_1.sh

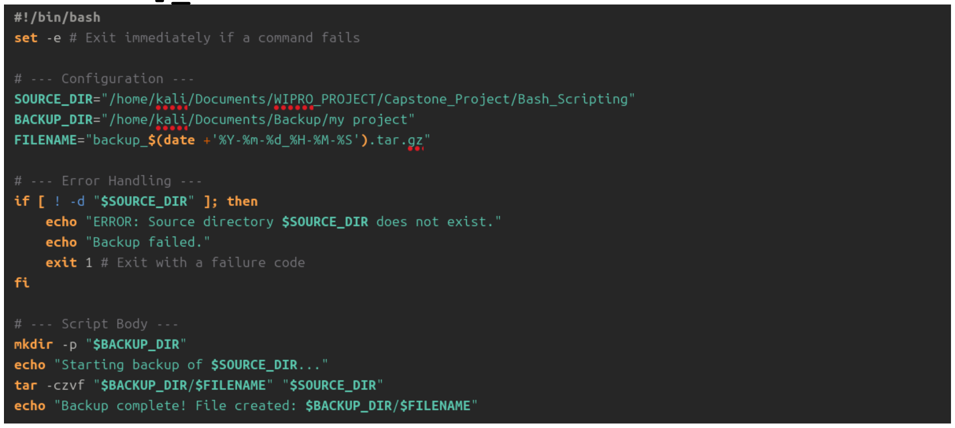


```
#!/bin/bash
# --- Configuration ---
# Define the paths to your other scripts.
# If they are in the same directory, this is all you need.
BACKUP_SCRIPT="./backup.sh"
MAINTENANCE_SCRIPT="./maintenance.sh"
LOG_MONITOR_SCRIPT="./log_monitor.sh"
# --- Main Menu Loop ---
while true; do
    clear # Clear the screen for a clean menu
    echo "================================="
    echo "  System Maintenance Suite Menu"
    echo "================================="
    echo "1. Run System Backup"
    echo "2. Run System Maintenance (Update & Clean)"
    echo "3. Monitor System Logs (Check for 'Failed')"
    echo "4. Exit"
    echo "---------------------------------"
    echo -n "Enter your choice [1-4]: "

    read choice
```



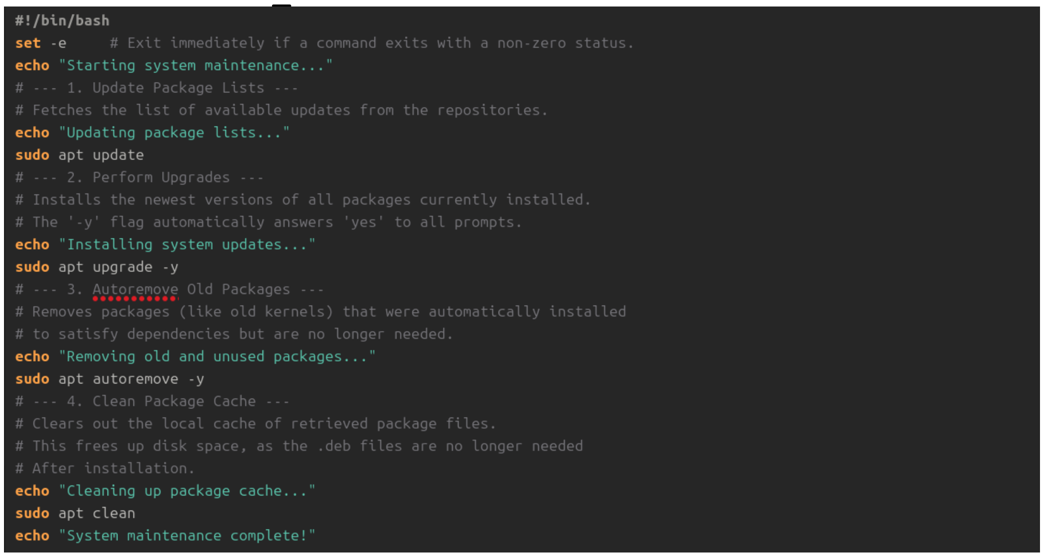
```bash
    # --- Handle User Choice ---
    case $choice in
        1)
            echo "Running backup script..."
            # Run the backup script
            $BACKUP_SCRIPT
            ;;
        2)
            echo "Running maintenance script (requires sudo)..."
            # Run the maintenance script with sudo
            sudo $MAINTENANCE_SCRIPT
            ;;
        3)
            echo "Running log monitor (requires sudo)..."
            # Run the log monitor script with sudo
            sudo $LOG_MONITOR_SCRIPT
            ;;
        4)
            echo "Exiting suite. Goodbye!"
            break # Exit the while loop
            ;;
        *)
            echo "Invalid choice. Please enter a number between 1 and 4."
            ;;
    esac

    echo ""
echo "Press Enter to return to the menu..."
    read # Pause and wait for user to press Enter
done
```

ii. backup_1.sh



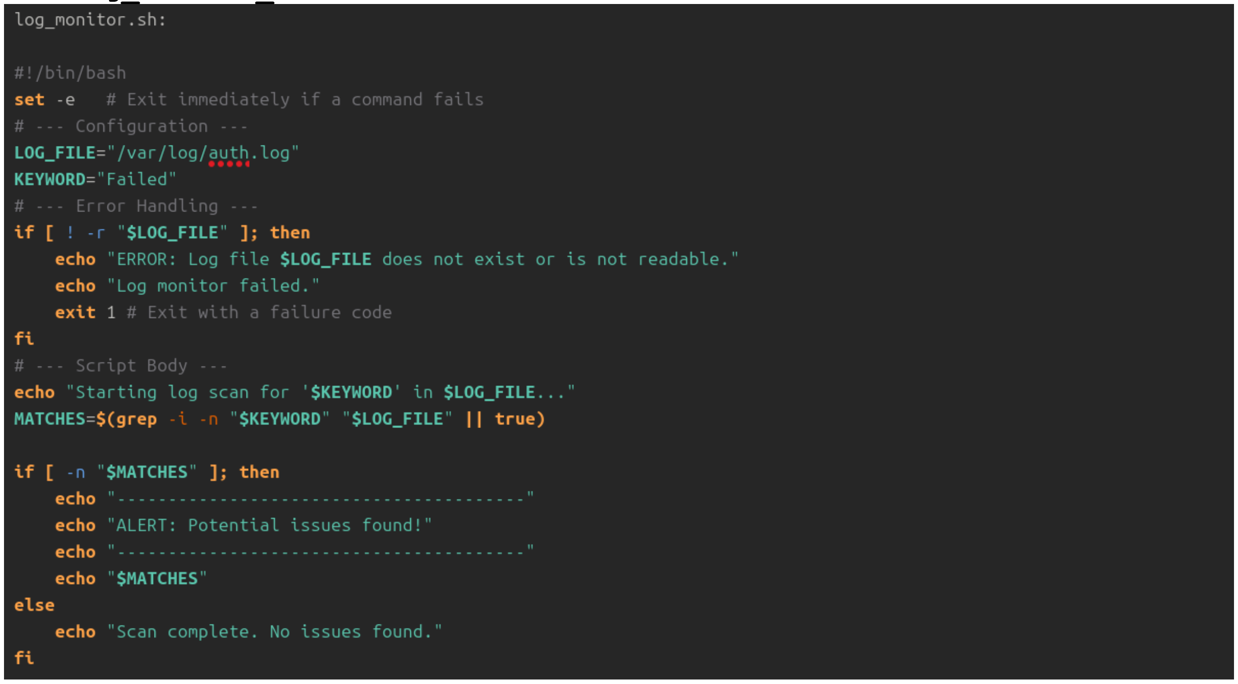
```bash
#!/bin/bash
set -e # Exit immediately if a command fails

# --- Configuration ---
SOURCE_DIR="/home/kali/Documents/WIPRO_PROJECT/Capstone_Project/Bash_Scripting"
BACKUP_DIR="/home/kali/Documents/Backup/my project"
FILENAME="backup_$(date +'%Y-%m-%d_%H-%M-%S').tar.gz"

# --- Error Handling ---
if [ ! -d "$SOURCE_DIR" ]; then
    echo "ERROR: Source directory $SOURCE_DIR does not exist."
    echo "Backup failed."
    exit 1 # Exit with a failure code
fi

# --- Script Body ---
mkdir -p "$BACKUP_DIR"
echo "Starting backup of $SOURCE_DIR..."
tar -czvf "$BACKUP_DIR/$FILENAME" "$SOURCE_DIR"
echo "Backup complete! File created: $BACKUP_DIR/$FILENAME"
```

iii. maintenance_1.sh



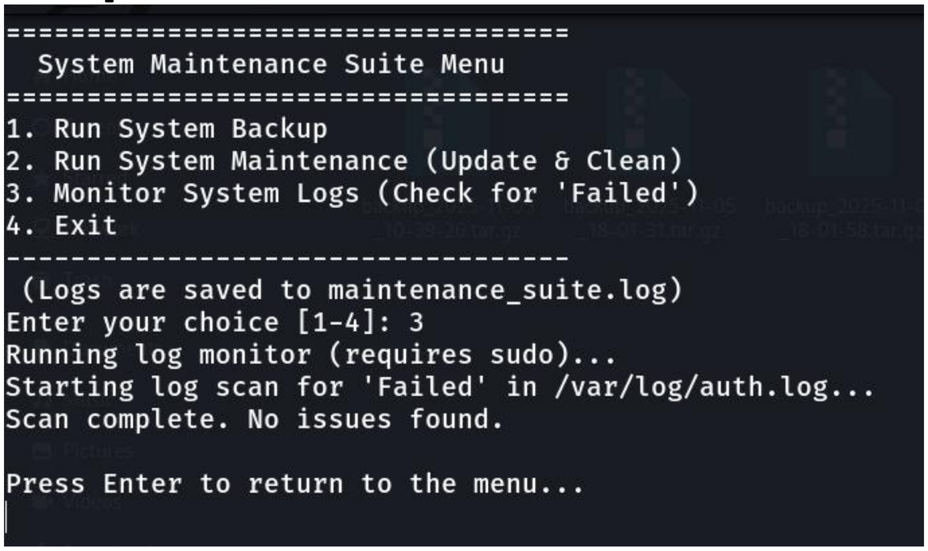
```bash
#!/bin/bash
set -e     # Exit immediately if a command exits with a non-zero status.
echo "Starting system maintenance..."
# --- 1. Update Package Lists ---
# Fetches the list of available updates from the repositories.
echo "Updating package lists..."
sudo apt update
# --- 2. Perform Upgrades ---
# Installs the newest versions of all packages currently installed.
# The '-y' flag automatically answers 'yes' to all prompts.
echo "Installing system updates..."
sudo apt upgrade -y
# --- 3. Autoremove Old Packages ---
# Removes packages (like old kernels) that were automatically installed
# to satisfy dependencies but are no longer needed.
echo "Removing old and unused packages..."
sudo apt autoremove -y
# --- 4. Clean Package Cache ---
# Clears out the local cache of retrieved package files.
# This frees up disk space, as the .deb files are no longer needed
# After installation.
echo "Cleaning up package cache..."
sudo apt clean
echo "System maintenance complete!"
```

iv. log_monitor_1.sh



```
log_monitor.sh:

#!/bin/bash
set -e   # Exit immediately if a command fails
# --- Configuration ---
LOG_FILE="/var/log/auth.log"
KEYWORD="Failed"
# --- Error Handling ---
if [ ! -r "$LOG_FILE" ]; then
    echo "ERROR: Log file $LOG_FILE does not exist or is not readable."
    echo "Log monitor failed."
    exit 1 # Exit with a failure code
fi
# --- Script Body ---
echo "Starting log scan for '$KEYWORD' in $LOG_FILE..."
MATCHES=$(grep -i -n "$KEYWORD" "$LOG_FILE" || true)

if [ -n "$MATCHES" ]; then
    echo "----------------------------------------"
    echo "ALERT: Potential issues found!"
    echo "----------------------------------------"
    echo "$MATCHES"
else
    echo "Scan complete. No issues found."
fi
```

v. output



```
===================================
  System Maintenance Suite Menu
===================================
1. Run System Backup
2. Run System Maintenance (Update & Clean)
3. Monitor System Logs (Check for 'Failed')
4. Exit
-----------------------------------
 (Logs are saved to maintenance_suite.log)
Enter your choice [1-4]: 3
Running log monitor (requires sudo)...
Starting log scan for 'Failed' in /var/log/auth.log...
Scan complete. No issues found.

Press Enter to return to the menu...
```