Bash Script Project Documentation

Project Title: System Monitoring Script

Introduction:

This bash script monitors system resource usage (CPU, Memory, Disk) and sends email alerts when usage exceeds predefined thresholds. The script gathers real-time system metrics, logs them in a CSV file, creates graphs with Gnuplot, and sends email alerts for resource usage issues.

The purpose of this script is to help system administrators track the health of their systems, ensuring optimal resource usage and alerting them to any potential performance issues. The script helps monitor key system resources, generates graphical reports, and sends alerts, making system management and troubleshooting easier.

Author:

- **Rhosebrian Jamisola**, **Clarence Japinan**, **January Iverson Libutan**, **Jorge Magno, Arvin Colonia**

Version:

- **1.0**

Project Overview:

The script collects real-time CPU, Memory, and Disk usage statistics from a Linux-based system, evaluates these metrics against user-defined thresholds, logs the usage data for historical tracking, generates graphical representations of resource trends, and sends email alerts when thresholds are exceeded to its limit.

Key functionalities include:

- **Monitoring System Resources:** Continuously monitor CPU, Memory, and Disk usage.

- **Alerting on Resource Overages:** If resource usage exceeds thresholds limit, the script generates alert notifications.
- **Logging and Reporting:** Logs the collected data to a CSV file for analysis and generates graphical reports to visualize.
- **Automated Execution via Cron Job:** The script can be scheduled to run at specified intervals using cron jobs to ensure continuous monitoring.
- **Email Notifications:** Alerts are sent to specified email addresses using the `msmtp` utility.

Features:

1. **System Resource Monitoring:**
   - Tracks CPU, Memory, and Disk usage in real-time.
   - Collects statistics every time the script runs.
2. **Email Alerts:**
   - Sends automated email notifications when resource usage crosses predefined thresholds.
   - Uses `msmtp` for sending email notifications.
3. **Graphical Reporting:**
   - Generates visual reports using Gnuplot. ○ Provides a graphical representation of CPU, Memory, and Disk usage trends over time.
4. **Data Logging:**
   - Logs resource usage to a CSV file for historical analysis.
   - Maintains a detailed record of system resource usage.
5. **Automated Scheduling:**
   - Can be scheduled to run at regular intervals using cron jobs. ○ Supports continuous system monitoring and alerting.

Usage:

The script can be executed from the command line using the following command:

```
chmod +x system_monitor.sh
./system_monitor.sh
```

1. **Threshold Variables (within the script):**

- CPU_THRESHOLD: Percentage of CPU usage that triggers an alert. Default: 50%.
- MEM_THRESHOLD: Percentage of memory usage that triggers an alert. Default: 50%.
- DISK_THRESHOLD: Percentage of disk usage that triggers an alert. Default: 50%.

To modify thresholds, edit these variables in the script:

`nano system_monitor.csv`.

2. **Scheduling (option):** Users can schedule the script with cron for periodic execution (e.g., every 5 minutes):

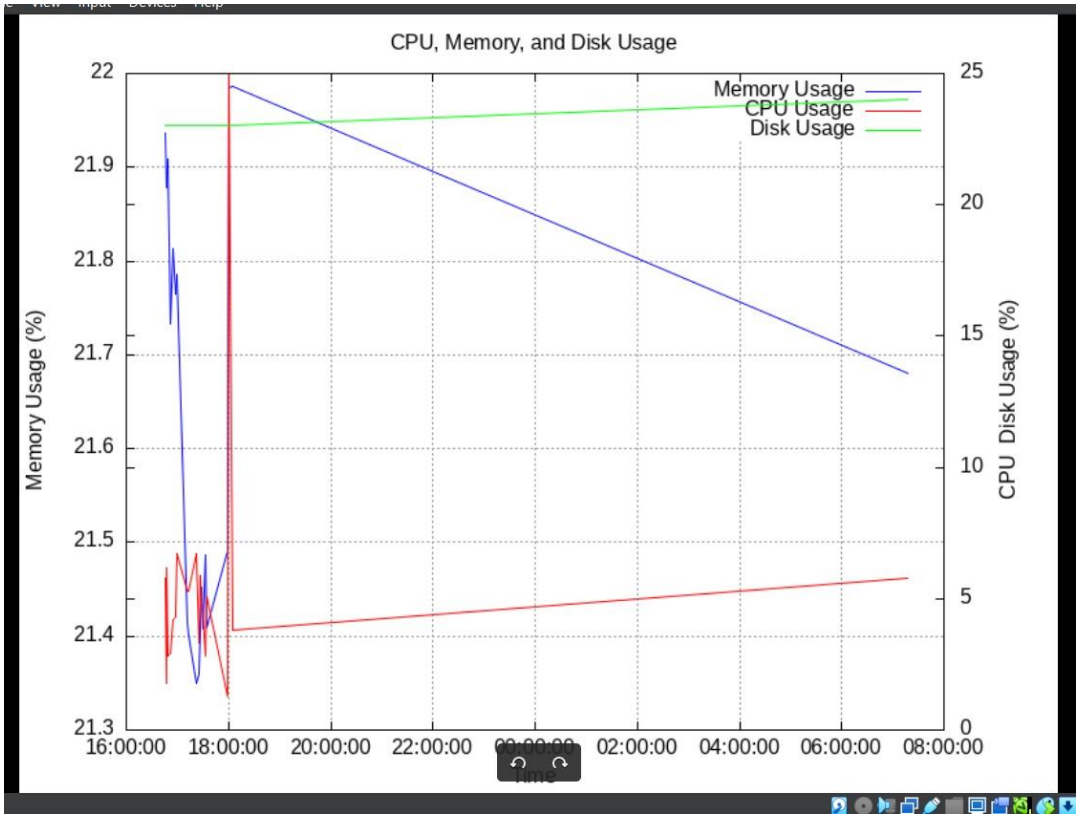
Output:

1. **CSV Log File**:
   The script logs system resource usage into `~/FinalProject/system_monitor.csv`.

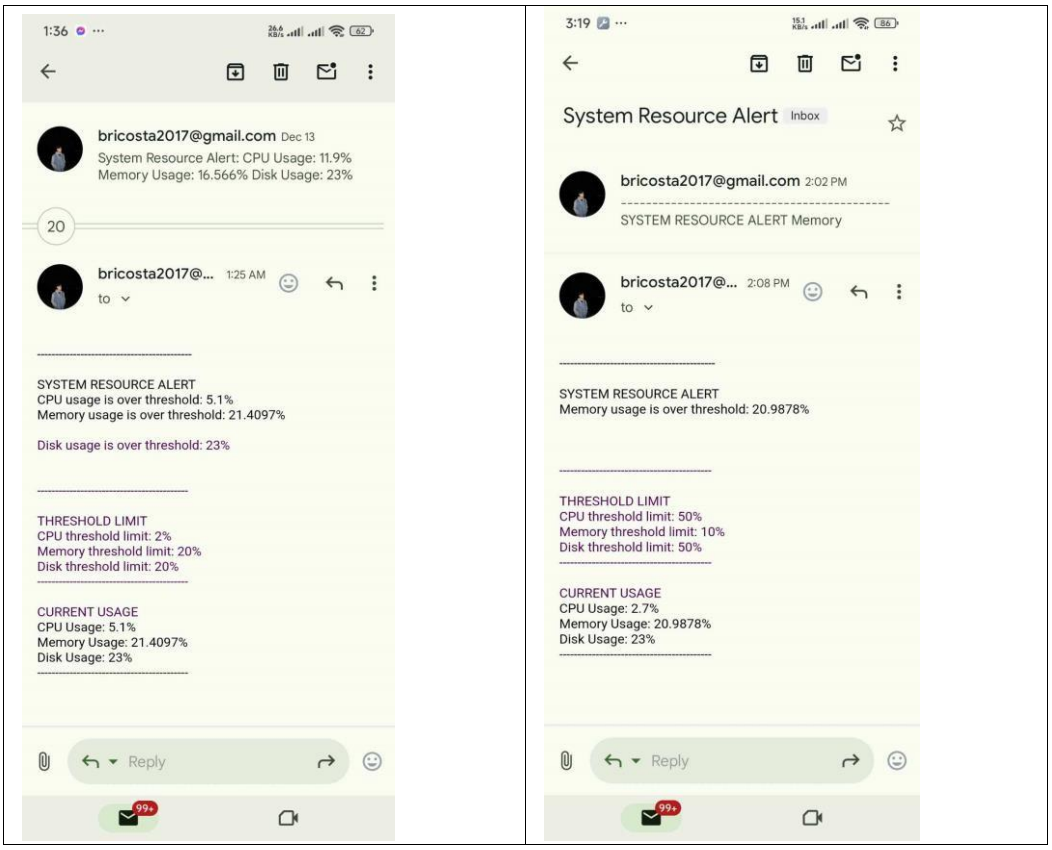
```
Jamisola@Jamisola:~/FinalProject$ cat ~/FinalProject/system_monitor.csv
2024-12-13 16:46:10,5.8,21.937,23
2024-12-13 16:46:51,1.8,21.8776,23
2024-12-13 16:47:20,6.2,21.8837,23
2024-12-13 16:48:21,2.8,21.9092,23
2024-12-13 16:52:26,2.9,21.7331,23
2024-12-13 16:55:10,4.2,21.8129,23
2024-12-13 16:58:22,4.3,21.765,23
2024-12-13 16:59:12,6.7,21.7861,23
2024-12-13 17:11:34,5.3,21.4151,23
2024-12-13 17:12:58,5.3,21.4008,23
2024-12-13 17:22:41,6.7,21.3504,23
2024-12-13 17:26:00,3.3,21.3591,23
2024-12-13 17:26:34,5.9,21.4094,23
2024-12-13 17:28:25,4.8,21.4525,23
2024-12-13 17:30:32,4.5,21.4073,23
2024-12-13 17:32:31,2.8,21.4868,23
2024-12-13 17:33:45,5.1,21.4097,23
2024-12-13 17:58:23,1.3,21.4906,23
2024-12-13 18:00:50,25,21.9846,23
2024-12-13 18:05:08,3.8,21.9862,23
2024-12-14 06:16:38,2.7,20.9878,23
2024-12-14 06:19:48,3.5,21.7354,24
```

The format of the log file includes Timestamp,CPU Usage (%),Memory Usage (%),Disk Usage (%)

2. **Graph File**:
   A graphical representation of resource usage is generated as `~/FinalProject/resource_ usage.png` to open use `xdg-open resource_usage.png`



3. **Email Alerts**:
   If any resource usage exceeds the specified thresholds, the script will send an alert email to the configured recipient (`ALERT_EMAIL`).



Requirements:

1. **Operating System**: A Linux-based system (tested on Ubuntu 24.04.1 LTS).
2. **Package installed**:
   - **Gnuplot**: Required for generating graphical reports.

```
~/FinalProject$ sudo apt install -y gnuplot
```

```
Jamisola@Jamisola:~/FinalProject$ gnuplot --version
gnuplot 6.0 patchlevel 0
```

   - **msmtp**: Required for sending email notifications.

```
~/FinalProject$ sudo apt install -y msmtp
```

```
Jamisola@Jamisola:~/FinalProject$ msmtp --version
msmtp version 1.8.24
```

- **bc**: Required for numerical calculations.

```
~/FinalProject$ sudo apt install -y bc
```

```
Jamisola@Jamisola:~/FinalProject$ bc --version
bc 1.07.1
```

3. **MSMTP Configuration**:
   - The `msmtp` tool is used to send email alerts. You need to configure `~/.msmtprc` file with your SMTP credentials.

```
Jamisola@Jamisola:~/FinalProject$ nano ~/.msmtprc
```

   - Configuration for Gmail:

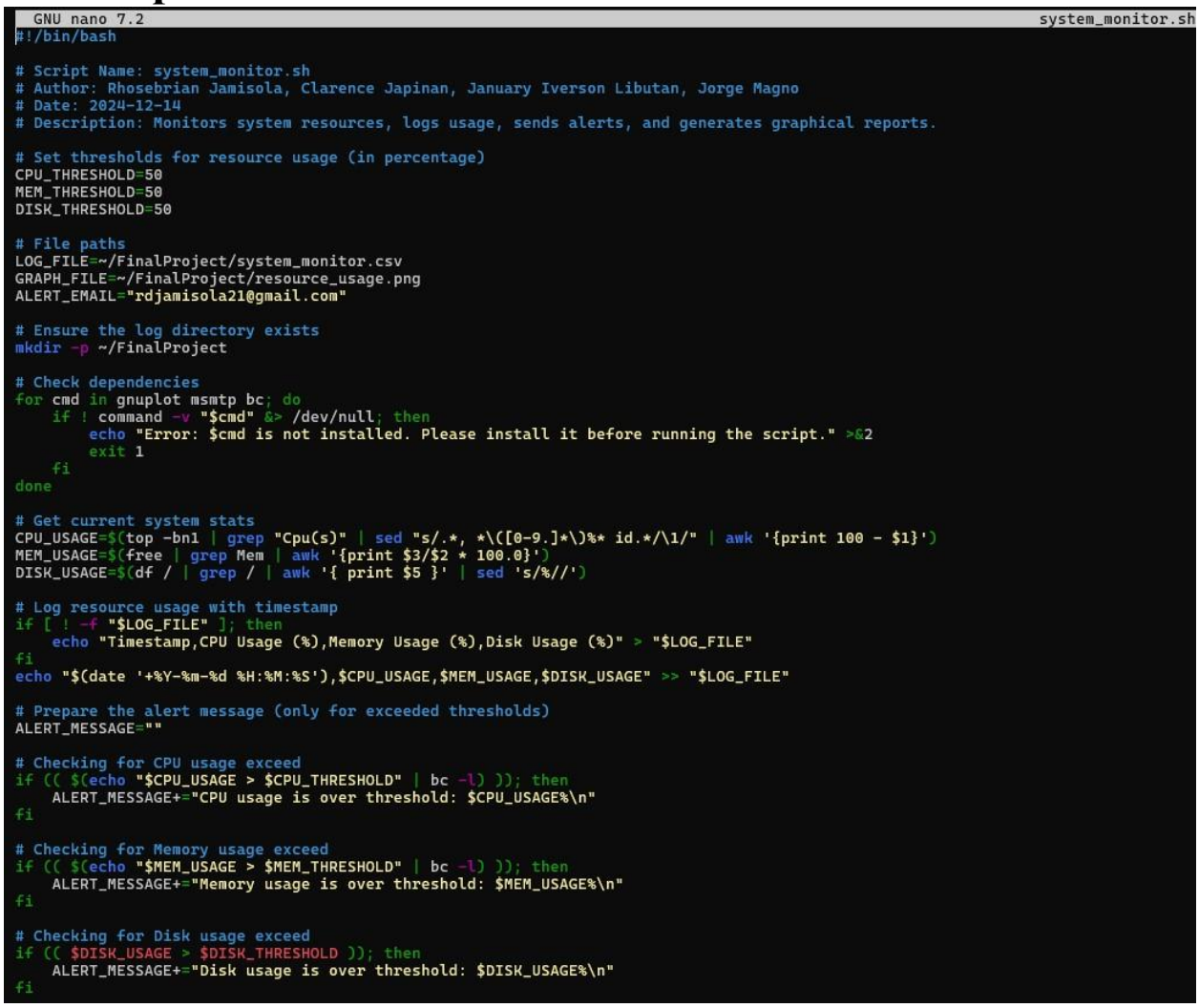
```
  GNU nano 7.2
defaults
auth           on
tls            on
tls_trust_file /etc/ssl/certs/ca-certificates.crt
logfile        ~/.msmtp.log

account gmail
host smtp.gmail.com
port 587
user bricosta2017@gmail.com
password qdfh qgzf xabj btqf
from bricosta2017@gmail.com

account default : gmail
```

   - Make sure the `msmtprc` file is configured properly and the permissions are set to secure it: `chmod 600 ~/.msmtprc`

```
Jamisola@Jamisola:~/FinalProject$ chmod 600 ~/.msmtprc
```

Script Explanation:

Full script



```
  GNU nano 7.2                                                                system_monitor.sh
#!/bin/bash

# Script Name: system_monitor.sh
# Author: Rhosebrian Jamisola, Clarence Japinan, January Iverson Libutan, Jorge Magno
# Date: 2024-12-14
# Description: Monitors system resources, logs usage, sends alerts, and generates graphical reports.

# Set thresholds for resource usage (in percentage)
CPU_THRESHOLD=50
MEM_THRESHOLD=50
DISK_THRESHOLD=50

# File paths
LOG_FILE=~/FinalProject/system_monitor.csv
GRAPH_FILE=~/FinalProject/resource_usage.png
ALERT_EMAIL="rdjamisola21@gmail.com"

# Ensure the log directory exists
mkdir -p ~/FinalProject

# Check dependencies
for cmd in gnuplot msmtp bc; do
    if ! command -v "$cmd" &> /dev/null; then
        echo "Error: $cmd is not installed. Please install it before running the script." >&2
        exit 1
    fi
done

# Get current system stats
CPU_USAGE=$(top -bn1 | grep "Cpu(s)" | sed "s/.*, *\([0-9.]*\)%* id.*/\1/" | awk '{print 100 - $1}')
MEM_USAGE=$(free | grep Mem | awk '{print $3/$2 * 100.0}')
DISK_USAGE=$(df / | grep / | awk '{ print $5 }' | sed 's/%//')

# Log resource usage with timestamp
if [ ! -f "$LOG_FILE" ]; then
    echo "Timestamp,CPU Usage (%),Memory Usage (%),Disk Usage (%)" > "$LOG_FILE"
fi
echo "$(date '+%Y-%m-%d %H:%M:%S'),$CPU_USAGE,$MEM_USAGE,$DISK_USAGE" >> "$LOG_FILE"

# Prepare the alert message (only for exceeded thresholds)
ALERT_MESSAGE=""

# Checking for CPU usage exceed
if (( $(echo "$CPU_USAGE > $CPU_THRESHOLD" | bc -l) )); then
    ALERT_MESSAGE+="CPU usage is over threshold: $CPU_USAGE%\n"
fi

# Checking for Memory usage exceed
if (( $(echo "$MEM_USAGE > $MEM_THRESHOLD" | bc -l) )); then
    ALERT_MESSAGE+="Memory usage is over threshold: $MEM_USAGE%\n"
fi

# Checking for Disk usage exceed
if (( $DISK_USAGE > $DISK_THRESHOLD )); then
    ALERT_MESSAGE+="Disk usage is over threshold: $DISK_USAGE%\n"
fi
```

```
# If there are any alerts, send the email
if [[ -n "$ALERT_MESSAGE" ]]; then
    EMAIL_BODY="----------------------------------------\n\n"
    EMAIL_BODY+="SYSTEM RESOURCE ALERT\n"
    EMAIL_BODY+="$ALERT_MESSAGE\n\n"
    EMAIL_BODY+="----------------------------------------\n\n"
    EMAIL_BODY+="THRESHOLD LIMIT\n"
    EMAIL_BODY+="CPU threshold limit: $CPU_THRESHOLD%\n"
    EMAIL_BODY+="Memory threshold limit: $MEM_THRESHOLD%\n"
    EMAIL_BODY+="Disk threshold limit: $DISK_THRESHOLD%\n"
    EMAIL_BODY+="----------------------------------------\n\n"
    EMAIL_BODY+="CURRENT USAGE\n"
    EMAIL_BODY+="CPU Usage: $CPU_USAGE%\n"
    EMAIL_BODY+="Memory Usage: $MEM_USAGE%\n"
    EMAIL_BODY+="Disk Usage: $DISK_USAGE%\n"
    EMAIL_BODY+="----------------------------------------\n"

    echo -e "Subject: System Resource Alert\n\n$EMAIL_BODY" | msmtp "$ALERT_EMAIL"
fi

# Generate the graph using gnuplot
gnuplot <<- EOF
    # Set the terminal to PNG
    set terminal png size 800,600

    # Set the output file for the graph
    set output "$GRAPH_FILE"

    # Set graph titles and labels
    set title "CPU, Memory, and Disk Usage"
    set xlabel "Time"
    set xdata time
    set timefmt "%Y-%m-%d %H:%M:%S"
    set format x "%H:%M:%S"
    set ylabel "Memory Usage (%)"
    set y2label "CPU & Disk Usage (%)"
    set grid

    # Use two Y axes for Memory, CPU, and Disk usage
    set y2tics
    set ytics nomirror
    set y2tics mirror

    # Ensure CSV is read properly and specify the correct separator
    set datafile separator ","

    # Plot the data from the CSV file
    plot "$LOG_FILE" using 1:3 with lines title "Memory Usage" axis x1y1 linecolor rgb "blue", \
         "$LOG_FILE" using 1:2 with lines title "CPU Usage" axis x1y2 linecolor rgb "red", \
         "$LOG_FILE" using 1:4 with lines title "Disk Usage" axis x1y2 linecolor rgb "green"
EOF

# Final output message
echo "System monitoring completed. Log: $LOG_FILE | Graph: $GRAPH_FILE"

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo
```

Explanation:

1. System Resource Collection:

```
# Set thresholds for resource usage (in percentage)
CPU_THRESHOLD=50
MEM_THRESHOLD=50
DISK_THRESHOLD=50
```

- **Explanation**:
    - Sets resource usage thresholds. Alerts are triggered when these limits are exceeded.

---

2. File Paths:

```
# File paths
LOG_FILE=~/FinalProject/system_monitor.csv
GRAPH_FILE=~/FinalProject/resource_usage.png
ALERT_EMAIL="rdjamisola21@gmail.com"
```

- **Explanation**:
  - Defines where the script logs metrics and saves the graph.

---

3. Dependencies Check:

```
# Check dependencies
for cmd in gnuplot msmtp bc; do
    if ! command -v "$cmd" &> /dev/null; then
        echo "Error: $cmd is not installed. Please install it before running the script." >&2
        exit 1
    fi
done
```

- **Explanation**:
  This block checks if the required dependencies (`gnuplot`, `msmtp`, `bc`) are installed on the system.
  - If any of these tools are missing, the script prints an error message and exits.

---

4. System Resource Collection:

```
# Get current system stats
CPU_USAGE=$(top -bn1 | grep "Cpu(s)" | sed "s/.*, *\([0-9.]*\)%* id.*/\1/" | awk '{print 100 - $1}')
MEM_USAGE=$(free | grep Mem | awk '{print $3/$2 * 100.0}')
DISK_USAGE=$(df / | grep / | awk '{ print $5 }' | sed 's/%//')
```

- **Explanation**:
  - **CPU_USAGE**: Extracts CPU usage percentage using the `top` command.
  - **MEM_USAGE**: Calculates memory usage percentage using the `free` command.
    - **DISK_USAGE**: Retrieves disk usage percentage using `df`.

5. Data Logging:

```
# Log resource usage with timestamp
if [ ! -f "$LOG_FILE" ]; then
    echo "Timestamp,CPU Usage (%),Memory Usage (%),Disk Usage (%)" > "$LOG_FILE"
fi
echo "$(date '+%Y-%m-%d %H:%M:%S'),$CPU_USAGE,$MEM_USAGE,$DISK_USAGE" >> "$LOG_FILE"
```

- **Explanation**:
    - **LOG_FILE**: Specifies where the log file will be stored (`~/FinalProject/system_monitor.csv`). ○ If the log file doesn't exist, it creates the header (`Timestamp, CPU Usage (%), Memory Usage (%), Disk Usage (%)`). ○ Appends current timestamp and resource usage data to the file.

6. Generate Alert:

```
# Prepare the alert message (only for exceeded thresholds)
ALERT_MESSAGE=""

# Checking for CPU usage exceed
if (( $(echo "$CPU_USAGE > $CPU_THRESHOLD" | bc -l) )); then
    ALERT_MESSAGE+="CPU usage is over threshold: $CPU_USAGE%\n"
fi

# Checking for Memory usage exceed
if (( $(echo "$MEM_USAGE > $MEM_THRESHOLD" | bc -l) )); then
    ALERT_MESSAGE+="Memory usage is over threshold: $MEM_USAGE%\n"
fi

# Checking for Disk usage exceed
if (( $DISK_USAGE > $DISK_THRESHOLD )); then
    ALERT_MESSAGE+="Disk usage is over threshold: $DISK_USAGE%\n"
fi
```

Explanation:

- **Thresholds**: `CPU_THRESHOLD`, `MEM_THRESHOLD`, and `DISK_THRESHOLD` are userdefined and represent the resource usage limits.
- **Alert Logic**: If any of the resource usage values exceed these thresholds, the script builds a message to notify the user.

7. Email Alert:

```
# If there are any alerts, send the email
if [[ -n "$ALERT_MESSAGE" ]]; then
    EMAIL_BODY="----------------------------------------\n\n"
    EMAIL_BODY+="SYSTEM RESOURCE ALERT\n"
    EMAIL_BODY+="$ALERT_MESSAGE\n\n"
    EMAIL_BODY+="----------------------------------------\n\n"
    EMAIL_BODY+="THRESHOLD LIMIT\n"
    EMAIL_BODY+="CPU threshold limit: $CPU_THRESHOLD%\n"
    EMAIL_BODY+="Memory threshold limit: $MEM_THRESHOLD%\n"
    EMAIL_BODY+="Disk threshold limit: $DISK_THRESHOLD%\n"
    EMAIL_BODY+="----------------------------------------\n\n"
    EMAIL_BODY+="CURRENT USAGE\n"
    EMAIL_BODY+="CPU Usage: $CPU_USAGE%\n"
    EMAIL_BODY+="Memory Usage: $MEM_USAGE%\n"
    EMAIL_BODY+="Disk Usage: $DISK_USAGE%\n"
    EMAIL_BODY+="----------------------------------------\n"

    echo -e "Subject: System Resource Alert\n\n$EMAIL_BODY" | msmtp "$ALERT_EMAIL"
fi
```

Explanation:

- **ALERT_EMAIL**: The email address where notifications will be sent.
- The email contains system resource usage details and alerts, if applicable.

---

8. Genarate Graph:

```
# Generate the graph using gnuplot
gnuplot <<- EOF
    # Set the terminal to PNG
    set terminal png size 800,600

    # Set the output file for the graph
    set output "$GRAPH_FILE"

    # Set graph titles and labels
    set title "CPU, Memory, and Disk Usage"
    set xlabel "Time"
    set xdata time
    set timefmt "%Y-%m-%d %H:%M:%S"
    set format x "%H:%M:%S"
    set ylabel "Memory Usage (%)"
    set y2label "CPU & Disk Usage (%)"
    set grid

    # Use two Y axes for Memory, CPU, and Disk usage
    set y2tics
    set ytics nomirror
    set y2tics mirror

    # Ensure CSV is read properly and specify the correct separator
    set datafile separator ","

    # Plot the data from the CSV file
    plot "$LOG_FILE" using 1:3 with lines title "Memory Usage" axis x1y1 linecolor rgb "blue", \
         "$LOG_FILE" using 1:2 with lines title "CPU Usage" axis x1y2 linecolor rgb "red", \
         "$LOG_FILE" using 1:4 with lines title "Disk Usage" axis x1y2 linecolor rgb "green"
EOF

# Final output message
echo "System monitoring completed. Log: $LOG_FILE | Graph: $GRAPH_FILE"
```

- **Explanation**:
    - Generates a PNG graph showing CPU, Memory, and Disk usage over time.

Troubleshooting:

1. Missing Dependencies:

- If the script fails due to missing dependencies, ensure `gnuplot`, `msmtp`, and `bc` are installed.
- Use the following commands to install required packages:
- `sudo apt update`
- `sudo apt install -y gnuplot msmtp bc`

2. Email Alerts Not Sending:

- Ensure the `msmtp` configuration is properly set up.
- Check the permissions of `~/.msmtprc` file to ensure it's not readable by others:
- `chmod 600 ~/.msmtprc`

3. Graph Not Generating:

- Ensure Gnuplot is installed and the required directories have write permissions.
- Verify the script runs in the intended environment with sufficient permissions to generate output files.
