

## **Placement Empowerment Program**

### ***Cloud Computing and DevOps Centre***

#### **Write a Shell Script to Monitor Logs**

Create a script that monitors server logs for errors and alerts you.

Name: Bhavyaa.V

Department: IT

# Introduction and Overview

This Shell script for log monitoring on Windows is designed to help track and manage system logs for specific events, such as errors. It utilizes built-in log monitoring tools (like PowerShell or command line utilities) to parse through log files, searching for keywords like "error." Once an error is detected, the script triggers notifications, such as sending an alert via email or simply outputting an error message to the console, to inform administrators or users of potential issues. This process helps ensure that system anomalies or failures are promptly addressed.

## Objective

The goal of this project is to:

1. **Automate Log Monitoring:** Continuously monitor system logs for specific keywords (like "error") to identify potential issues in real-time.
2. **Alert Notifications:** Trigger notifications (via email or console alerts) whenever critical log entries are detected, ensuring quick response to errors.
3. **Enhance System Reliability:** Provide proactive monitoring and timely alerts to help maintain system health and reduce downtime or failures.

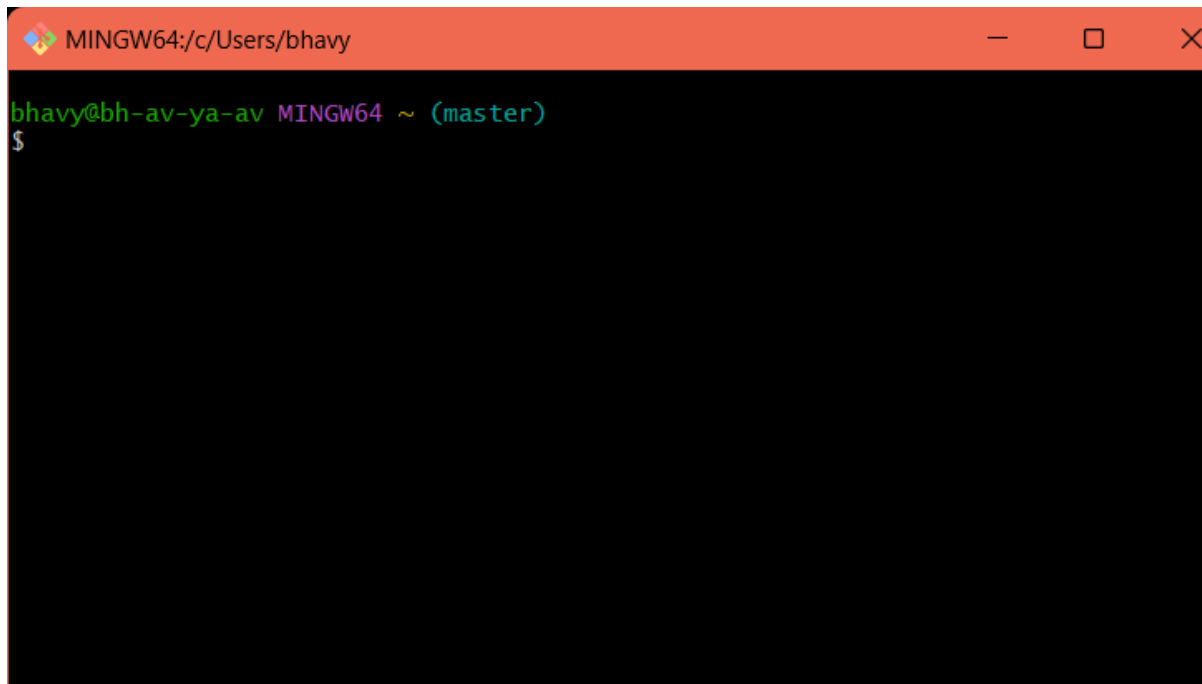
## Importance

1. **Early Detection of Issues:** By monitoring logs for errors, potential system issues or failures can be identified early, allowing for quicker resolution before they escalate into more severe problems.
2. **Proactive System Management:** Automated log monitoring ensures that administrators stay informed about system health without needing to manually check logs, making it easier to manage and maintain system reliability.
3. **Improved Efficiency:** Automated alerts save time and effort by eliminating the need for constant log checking, allowing system administrators to focus on resolving issues rather than searching for them.

# Step-by-Step Overview

## Step1: Open your Shell Environment

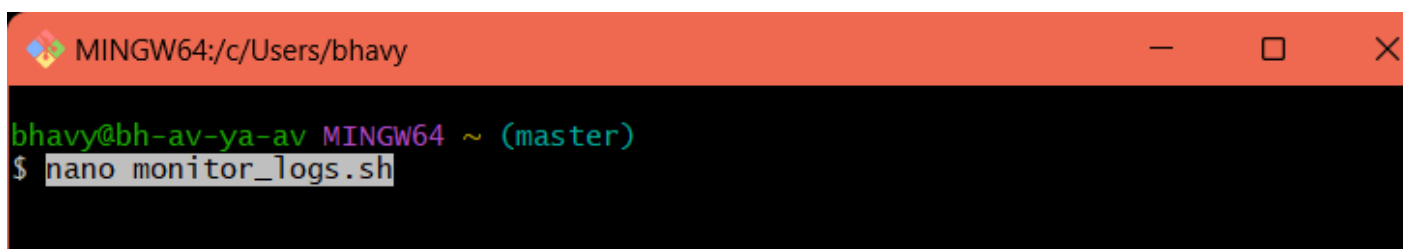
**Git Bash** or **Cygwin** can be used for running Unix-like shell commands on Windows.

A screenshot of a terminal window titled "MINGW64:/c/Users/bhavy". The terminal has a black background with green text. The prompt shows the user "bhavy" at host "bh-av-ya-av" in a "MINGW64" environment, currently in the "~ (master)" directory. The prompt is "\$".

```
MINGW64:/c/Users/bhavy
bhavy@bh-av-ya-av MINGW64 ~ (master)
$
```

## Step 2: Create Your Shell Script

Open the shell terminal and create a new script using your favorite text editor.

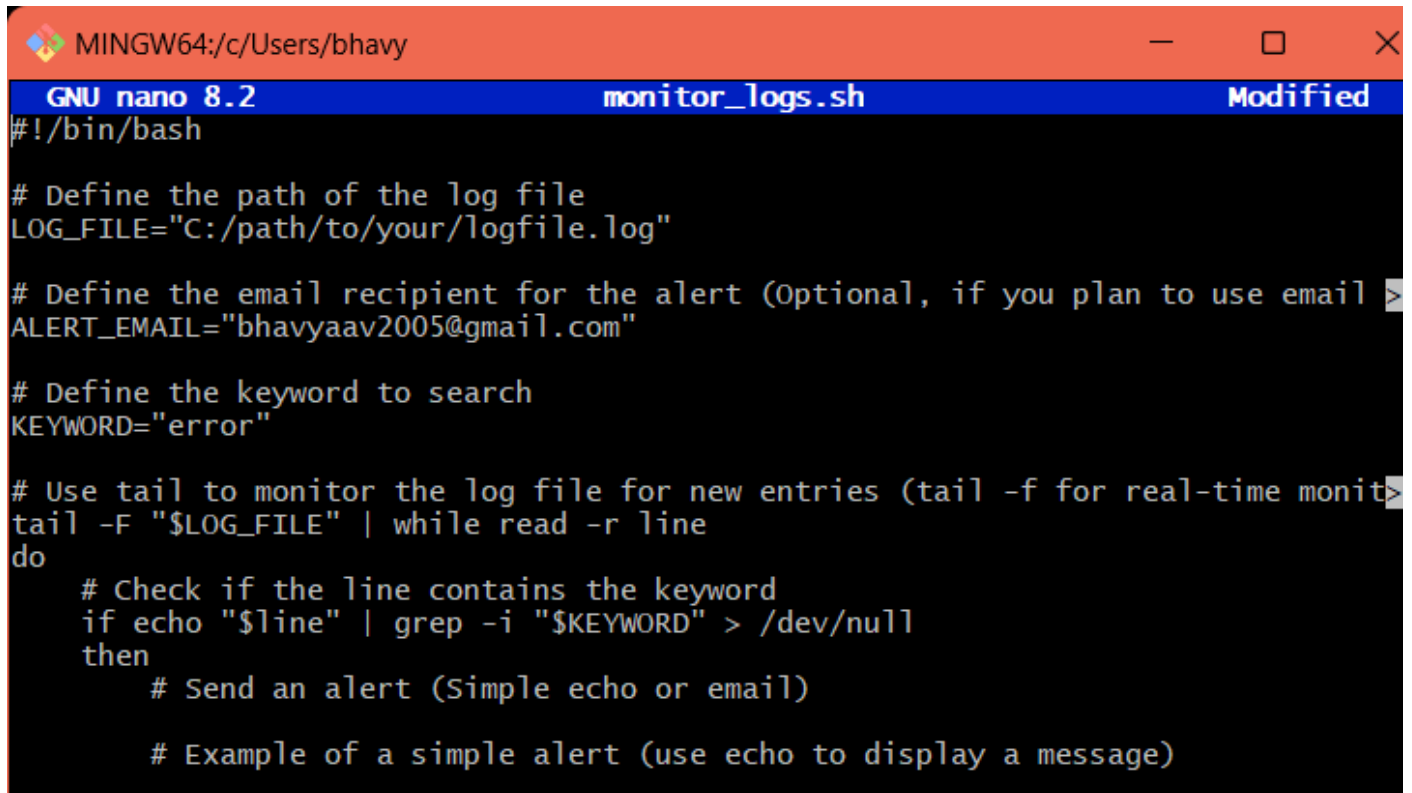
A screenshot of a terminal window titled "MINGW64:/c/Users/bhavy". The terminal has a black background with green text. The prompt shows the user "bhavy" at host "bh-av-ya-av" in a "MINGW64" environment, currently in the "~ (master)" directory. The prompt is "\$". The command "nano monitor\_logs.sh" is entered and highlighted in grey.

```
MINGW64:/c/Users/bhavy
bhavy@bh-av-ya-av MINGW64 ~ (master)
$ nano monitor_logs.sh
```

This will open a file where you can write your script.

## Step 3: Write the Script

Here's a basic script to monitor logs and send alerts if an "error" is found in the logs:



```
#!/bin/bash

# Define the path of the log file
LOG_FILE="C:/path/to/your/logfile.log"

# Define the email recipient for the alert (Optional, if you plan to use email)
ALERT_EMAIL="bhavyaav2005@gmail.com"

# Define the keyword to search
KEYWORD="error"

# Use tail to monitor the log file for new entries (tail -f for real-time monitoring)
tail -F "$LOG_FILE" | while read -r line
do
    # Check if the line contains the keyword
    if echo "$line" | grep -i "$KEYWORD" > /dev/null
    then
        # Send an alert (Simple echo or email)

        # Example of a simple alert (use echo to display a message)
```

## Step 4: Make the Script Executable

Once the script is written, you need to make it executable.

```
chmod +x monitor_logs.sh
```

## Step 5: Run the Script

Execute the script to start monitoring logs.

```
./monitor_logs.sh
```

# Expected Outcome

By completing this POC, you will:

1. **Timely Issue Identification:** The script will promptly detect errors or anomalies in log files, reducing the time taken to notice system problems.
2. **Automated Notifications:** Administrators or users will receive immediate alerts (via email or console), ensuring that critical issues are addressed as soon as they occur.
3. **Enhanced System Stability:** With continuous log monitoring and early alerts, system downtime and failures will be minimized, leading to improved overall stability.
4. **Efficient Resource Management:** The automated nature of the script will free up system administrators' time, allowing them to focus on more complex tasks while ensuring logs are consistently monitored.