



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: Madhumitha H

Department: ADS

Introduction:

This task involves creating a script that monitors log files for errors and alerts the user when specific keywords, like "error" or "critical," are detected. Using PowerShell in Windows, the script continuously checks the log file for updates. When an error is found, it displays an alert and plays a sound for notification. This method provides an efficient way to monitor system logs in real-time without requiring advanced tools. It is useful for identifying and responding to issues as soon as they appear in the logs.

Overview

Here's a concise overview of what you did in **6 steps**:

1. **Create the Script:** You wrote a PowerShell script that monitors a log file for error keywords like "error" and "critical."
2. **Set the Log File Path:** You specified the path of the log file (either a test log or a system log file).
3. **Monitor the Log File:** The script uses Get-Content to continuously monitor the log file in real-time for any updates.
4. **Search for Errors:** It searches each line for predefined error keywords and triggers an alert when a match is found.
5. **Display Alerts:** Upon detecting an error, the script prints a red-colored alert message and plays a sound to notify you.
6. **Test the Script:** You tested the script by adding error messages to a sample log file, confirming that it correctly detected errors and provided the alerts.

This simple process allows you to monitor logs for issues and stay informed about system events efficiently.

Objectives:

1. **Improve System Monitoring Efficiency:** Automate the process of log monitoring, reducing the need for manual checks and enabling proactive error management.
2. **Customizable Keyword Search:** Allow customization of the script to monitor different types of errors, based on specific keywords, making the tool adaptable to various environments.

3. **Enhance Troubleshooting:** By identifying errors as soon as they occur, the script helps in faster troubleshooting and minimizing potential system downtime.
4. **Integration with Notification Systems:** Enable the script to integrate with external notification systems (e.g., email or SMS) to notify system administrators or users about critical issues in real-time.
5. **Scalable to Multiple Log Files:** Adapt the script to handle and monitor multiple log files simultaneously, making it scalable for larger systems or multiple servers.
6. **Resource Optimization:** Ensure that the script runs efficiently without consuming excessive system resources, even when monitoring large log files in real-time.
7. **Simplified User Alerts:** Provide clear, readable output with alerts that are easy to interpret, ensuring users can quickly understand the severity of an issue.

Step-by-Step Overview

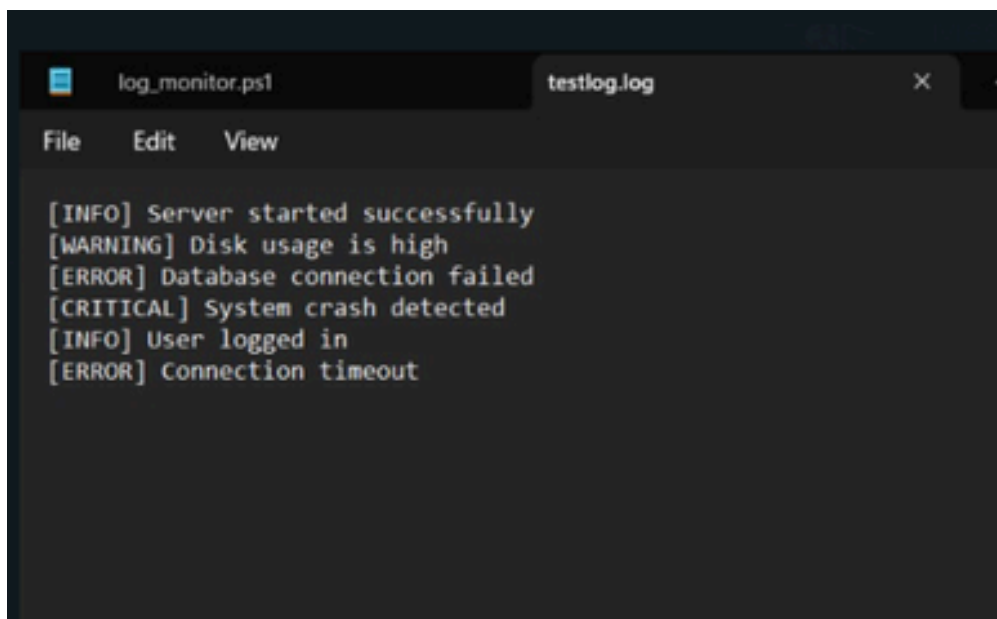
Step 1:

Create a log file called testlog.log in your desktop with some commands like:

[INFO] Server started successfully

[WARNING] Disk usage is high

[ERROR] Database connection failed

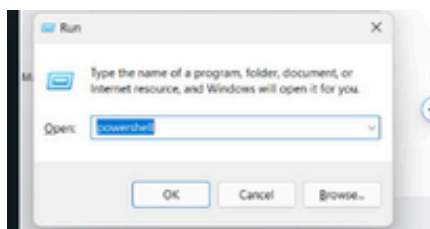


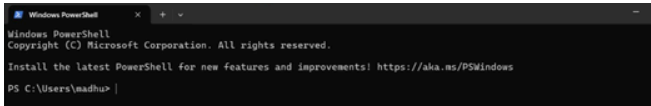
```
[INFO] Server started successfully
[WARNING] Disk usage is high
[ERROR] Database connection failed
[CRITICAL] System crash detected
[INFO] User logged in
[ERROR] Connection timeout
```

Step 2

Open PowerShell

- Press Win + R, type powershell, and hit **Enter** to open PowerShell.





Step 3

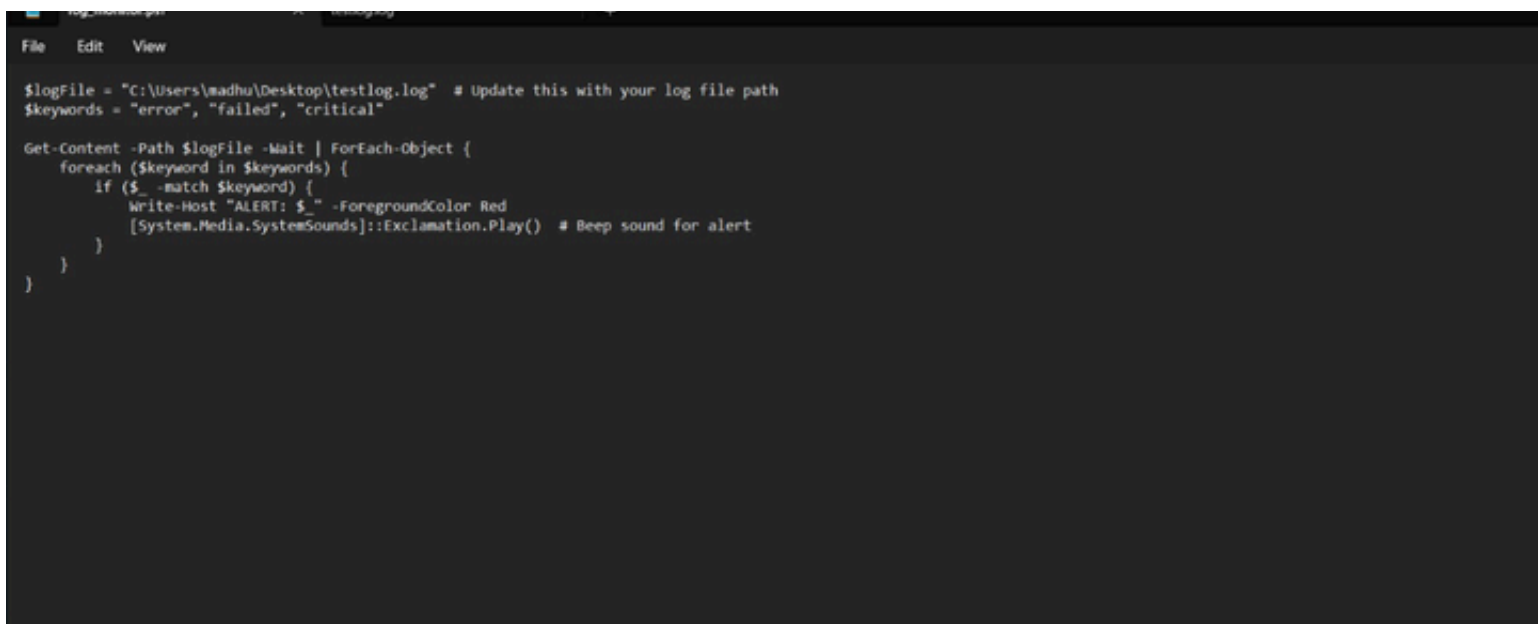
Create a PowerShell Script

- Open **Notepad** and copy the following script:

`$logFile = "C:\Users\YourUsername\Desktop\testlog.log" # Path to your log file`

`$keywords = "error", "failed", "critical"`

```
Get-Content -Path $logFile -Wait | ForEach-Object {  
    foreach ($keyword in $keywords) {  
        if ($_ -match $keyword) {  
            Write-Host "ALERT: $_" -ForegroundColor Red  
            [System.Media.SystemSounds]::Exclamation.Play() # Beep sound for alert  
        }  
    }  
}
```



put this code and add the location of the testlog.log file in this script to run it .and save this file in desktop aslog_monito.ps1.

Step 4:

Run the PowerShell Script

- Open **PowerShell** and navigate to your **Desktop** using the command:

cd desktop

or cd C:\Users\YourUsername\Desktop

```
PS C:\Users\madhu> cd desktop
```

Step 5

Run the script by typing:

.\log_monitor.ps1

```
PS C:\Users\madhu\desktop> .\log_monitor.ps1
ALERT: [ERROR] Database connection failed
ALERT: [ERROR] Database connection failed
ALERT: [CRITICAL] System crash detected
ALERT: [ERROR] Connection timeout
```

Thus the file successfully runs and detects the errors and alerts you.

Outcomes :

The outcome of this task is a fully automated log monitoring system that continuously checks log files for specific errors or critical events in real-time. When an issue is detected, the script provides immediate alerts, making it easier to identify and address problems as soon as they occur. This approach improves troubleshooting efficiency by ensuring that critical errors are noticed without delay. The script is customizable, allowing for tailored keyword searches and can be adapted to monitor various log files. It enhances system reliability by facilitating faster issue resolution and provides user-friendly notifications with both visual and sound alerts. Overall, this task results in a more efficient and proactive log monitoring process.