**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

**DAY 6 TASK:**

**Write a Shell Script to Monitor Logs**Create a script that monitors server logs for errors and alerts you.

**NAME:** NITHYASRI S.K

**DEPARTMENT:** ADS

**INTRODUCTION:**

Log files play a critical role in IT systems, as they record activities and events generated by applications, servers, and network devices. Monitoring these logs helps identify issues such as errors, warnings, and suspicious activities that may require immediate attention. Automating the monitoring process ensures efficiency and reduces the risk of missing critical information.

This PoC demonstrates the creation of a PowerShell script to monitor logs in real-time. The script will detect specific keywords (like "error") in a log file and alert the user when such events occur.

**OVERVIEW:**

This project involves writing and running a PowerShell script that continuously scans a log file for specific keywords.

The script:

1. Reads the log file in real time.

2. Matches new entries in the log against predefined keywords (e.g., "error").

3. Triggers an alert when a match is found. This solution is lightweight and practical for system administrators and IT professionals to monitor logs on Windows systems.

**OBJECTIVE**:

The objective of this project is to:

1. Automate the process of monitoring log files for critical events.

2. Learn how to create and execute PowerShell scripts on a Windows system.

3. Demonstrate real-time detection of keywords like "error" in log files.

4. Enhance troubleshooting efficiency by providing immediate alerts for critical events.

**IMPORTANCE:**

1. **Proactive Issue Detection** By monitoring logs in real time, this project helps detect errors and issues as they occur, reducing downtime and improving system reliability.

2. **Learning Automation Tools** This project introduces PowerShell scripting, a powerful automation tool, to beginners. It provides hands-on experience with practical applications.

3. **Cost-Effective Solution** Using PowerShell eliminates the need for expensive third-party monitoring tools while still achieving effective log monitoring.

4. **Time Efficiency** Automation saves significant manual effort in scanning logs, allowing IT professionals to focus on resolving issues. 5. **Scalability** The script can be adapted to monitor multiple log files or handle complex use cases, making it a foundational step toward advanced automation.

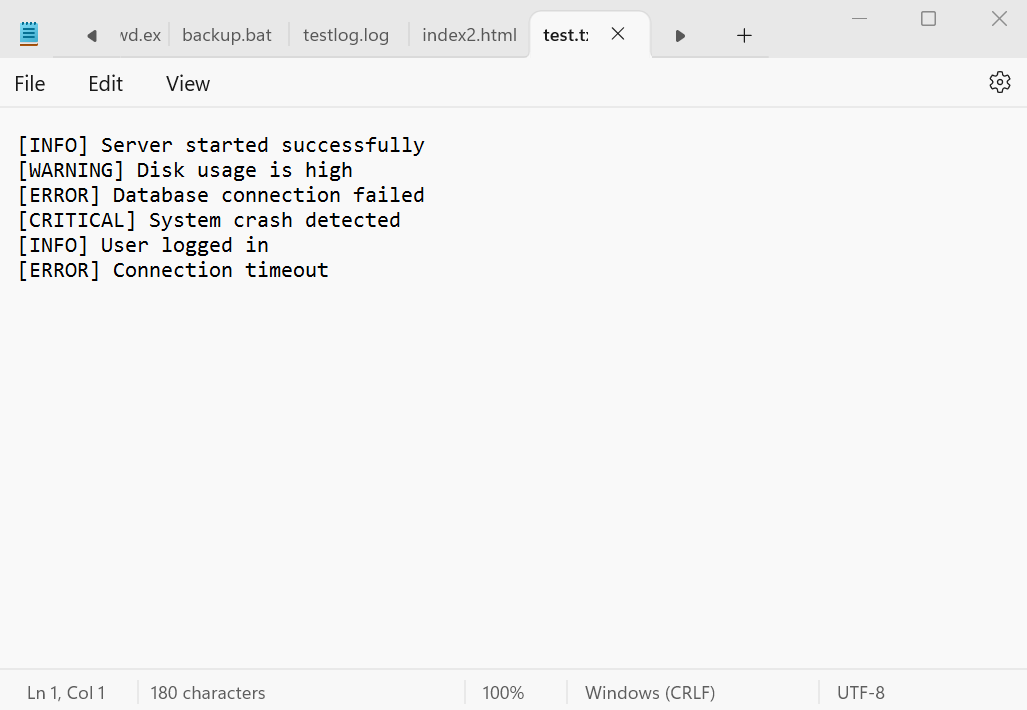
**STEP BY STEP OVERVIEW:**

Step 1: Create a log called testlog.log in your desktop with some commands like:

[INFO] Server started successfully

[WARNING] Disk usage is high

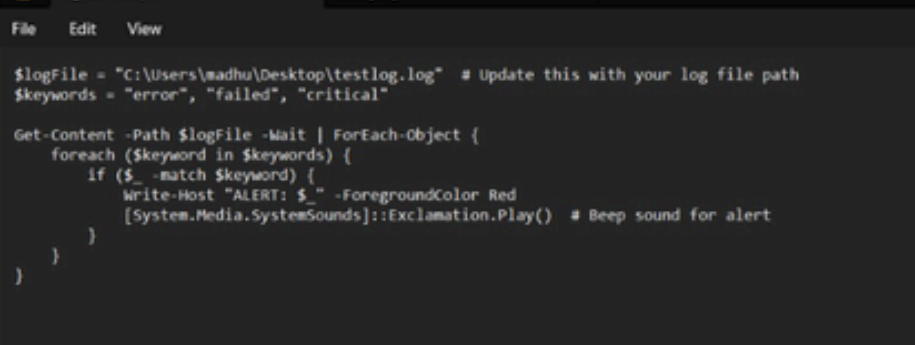
[ERROR] Database connection failed



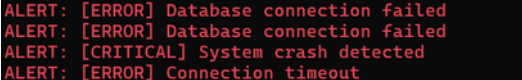
Step 2 Open PowerShell Press Win + R, type powershell, and hit Enter to open PowerShell.



Step 3 Open Notepad and copy the following script:



Step 4: Run the PowerShell Script. Open PowerShell and navigate to your Desktop using the command:

****

**OUTCOMES :**

The outcome of this task is a fully automated log monitoring system that continuously checks log les for specic 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 eciency 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 les. It enhances system reliability by facilitating faster issue resolution and provides user-friendly notications with both visual and sound alerts. Overall, this task results in a more ecient and proactive log monitoring proce