**Project Title: Automating Daily Security Tasks Using Windows CLI, Bash, and PowerShell.**

**Name: Oluwaseun Akinola**

**5/23/2025**

**Executive Summary:**

**Automating Daily Security Tasks Using Windows CLI, PowerShell, and Bash**

**A brief overview of the project’s objectives and importance.**

The goal of this project was to help me build practical skills in scripting by automating simple but important security tasks on both Windows and Linux systems. As someone preparing for a career in cybersecurity, I wanted to learn how to make routine checks more efficient and reduce the chances of missing something important.

**The security automation tasks are implemented in Windows CLI, PowerShell, and Bash.**

I used three different tools:

Windows CLI (.bat file) to check disk space, list running processes, and make sure Windows Defender is running.

PowerShell (.ps1 file) to give a quick summary of disk usage, show which services are running, and check how long the system has been up.

Bash (.sh file) to capture the last 5 login attempts and see if the sshd service is running on a Linux machine.

1. **Windows CLI Script (CLIScript.bat)**

@echo off

echo Running daily security tasks...

:: Disk space check

echo Checking disk space...

wmic logicaldisk get size,freespace,caption | findstr /R "[A-Z]:" > disk\_space\_log.txt

:: List active processes

echo Listing active processes...

tasklist > processes.txt

:: Check Windows Defender status

echo Checking Windows Defender service status...

sc query WinDefend > service\_status.txt

:: Final success message

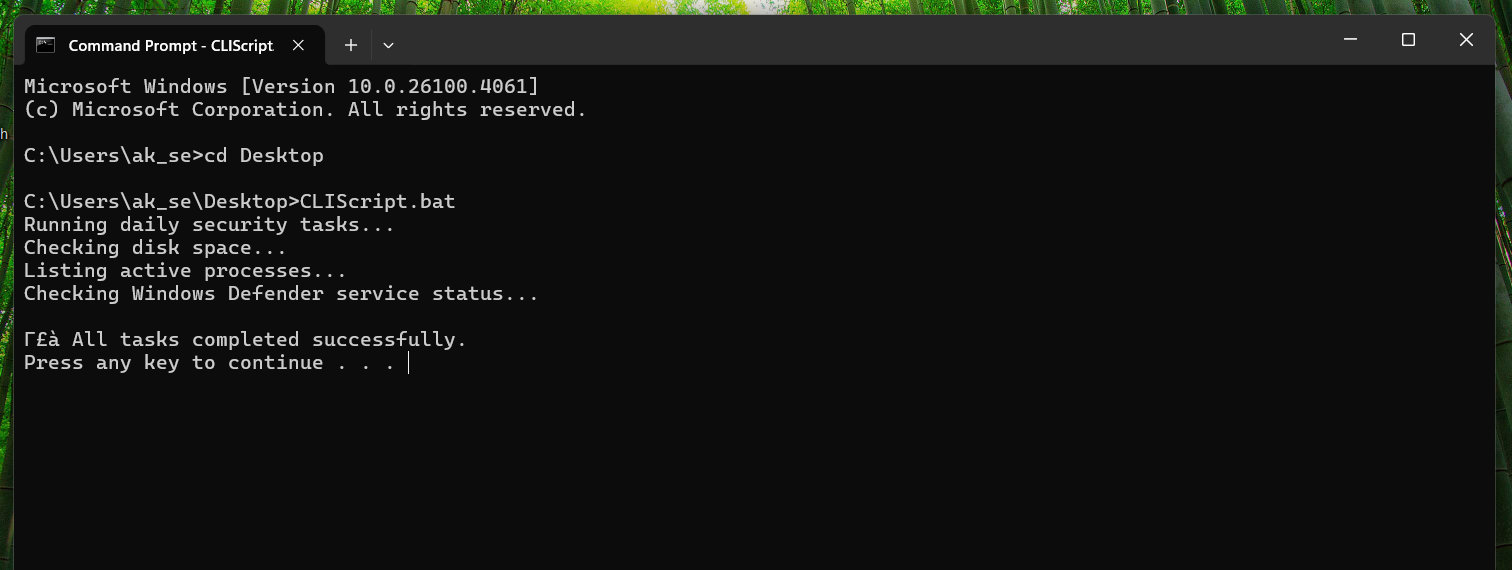
echo.

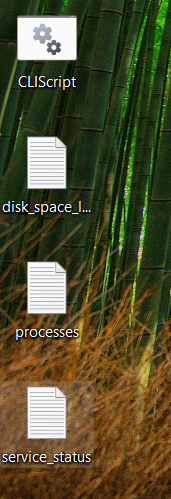
echo ✅ All tasks completed successfully.

:: Keep the window open

pause

**Screenshot showing task is successful**

****

****

1. **PowerShell Script(PSScript.ps1)**

# PSScript.ps1

# Automates daily security tasks: disk check, service status, and system uptime

Write-Output "Running daily security tasks..."

# === Disk Space Summary ===

Write-Output "Checking disk space..."

Get-PSDrive -PSProvider FileSystem | Select-Object Name,

@{Name="Used(GB)";Expression={[math]::round(($\_.Used/1GB),2)}},

@{Name="Free(GB)";Expression={[math]::round(($\_.Free/1GB),2)}},

@{Name="Total(GB)";Expression={[math]::round(($\_.Used + $\_.Free)/1GB,2)}} |

Out-File -FilePath disk\_summary.txt -Encoding UTF8

# === Running Services Detail ===

Write-Output "Retrieving running services..."

Get-Service | Where-Object {$\_.Status -eq 'Running'} |

Select-Object Name, DisplayName, Status, StartType |

Sort-Object Name |

Out-File -FilePath running\_services.txt -Encoding UTF8

# === System Uptime ===

Write-Output "Calculating system uptime..."

$uptime = (Get-Date) - (Get-CimInstance Win32\_OperatingSystem).LastBootUpTime

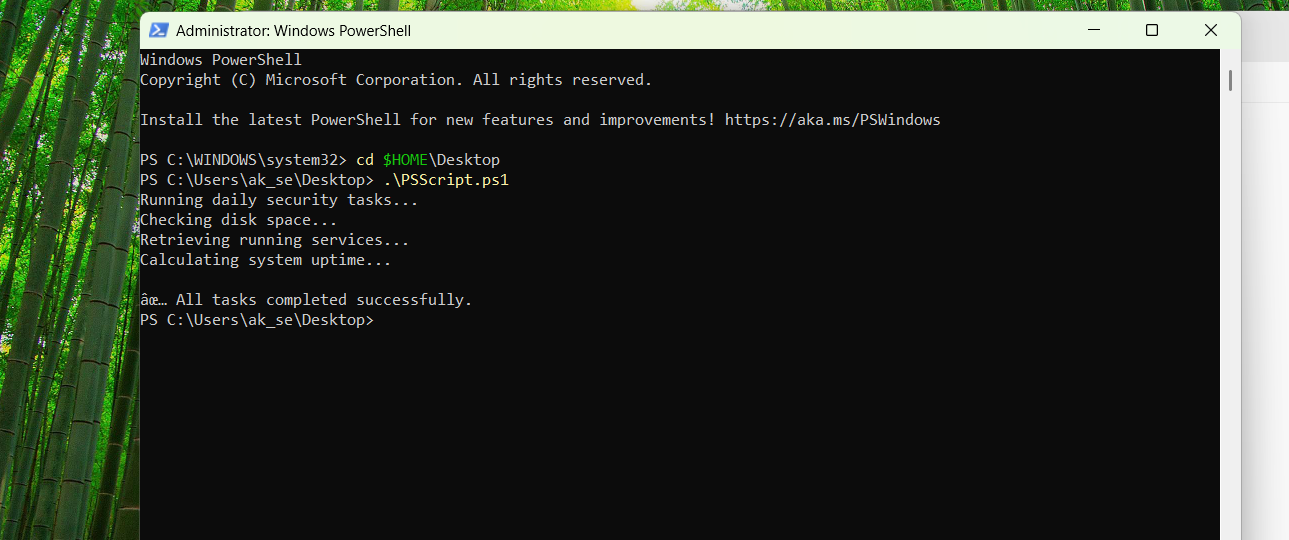
"System Uptime: $($uptime.Days) days, $($uptime.Hours) hours, $($uptime.Minutes) minutes" |

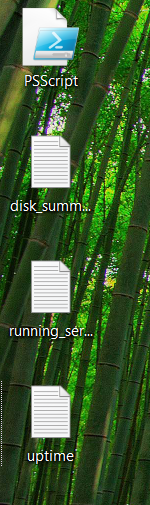
Out-File -FilePath uptime.txt -Encoding UTF8

# === Final Success Message ===

Write-Output "`n✅ All tasks completed successfully."

**Screenshot showing a successful task:**





1. **Bash Script**:

#!/bin/bash

echo "Running daily security tasks..."

# === Last 5 login attempts ===

echo "Saving last 5 login attempts..."

last -n 5 > login\_attempts.log

# === Check if sshd is running ===

echo "Checking sshd status..."

if pgrep sshd > /dev/null

then

echo "sshd is running" > sshd\_status.log

else

echo "sshd is NOT running" > sshd\_status.log

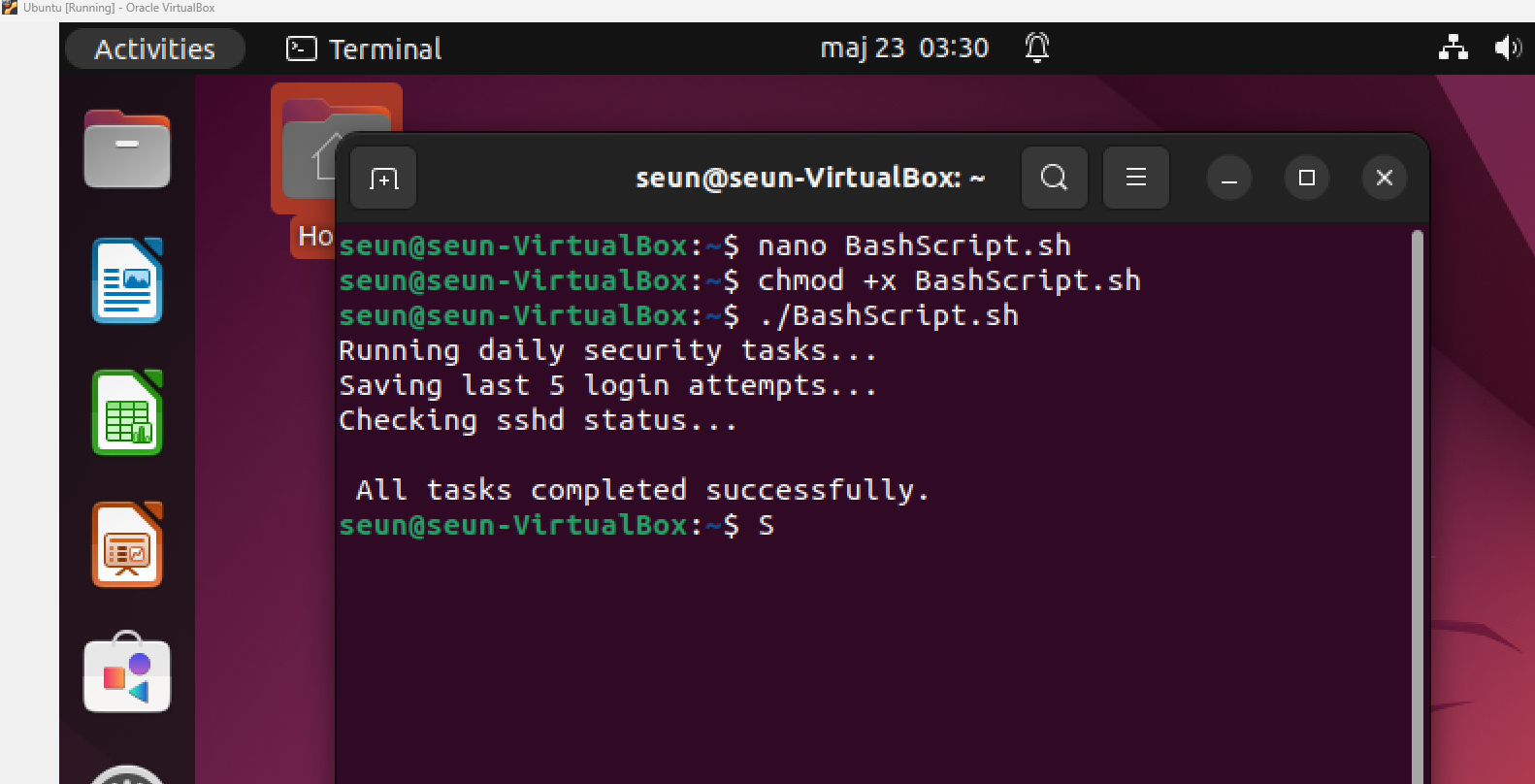
fi

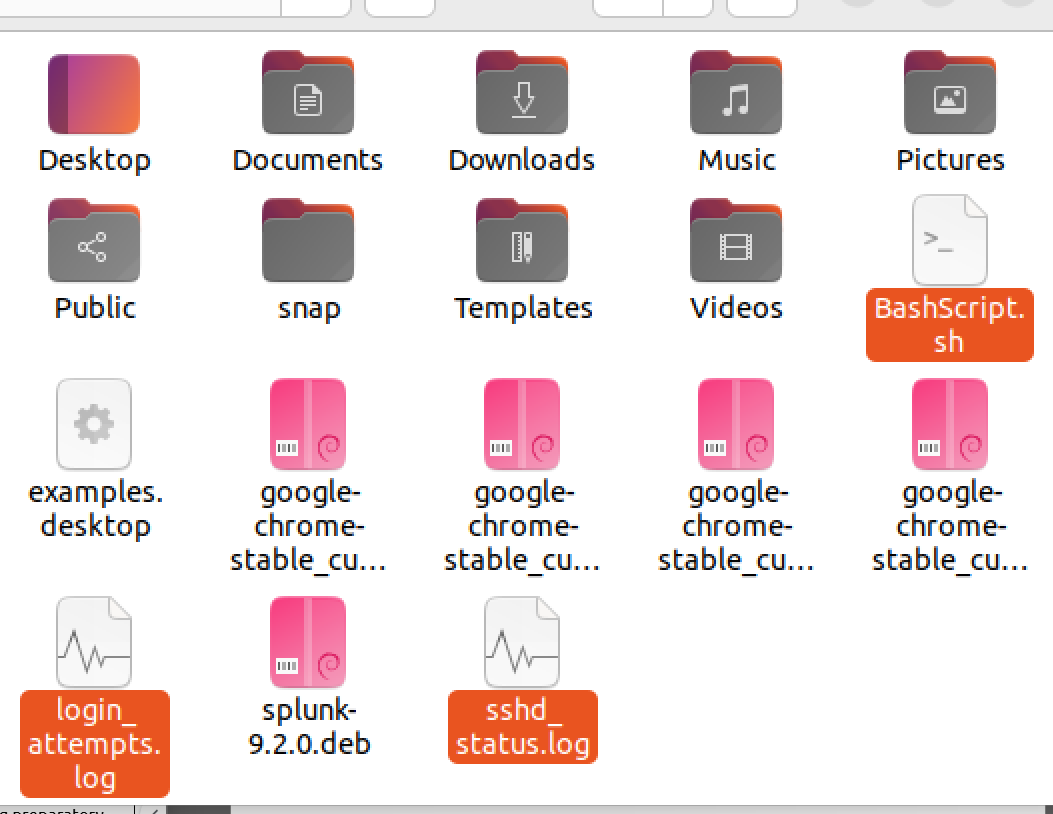
# === Success Message ===

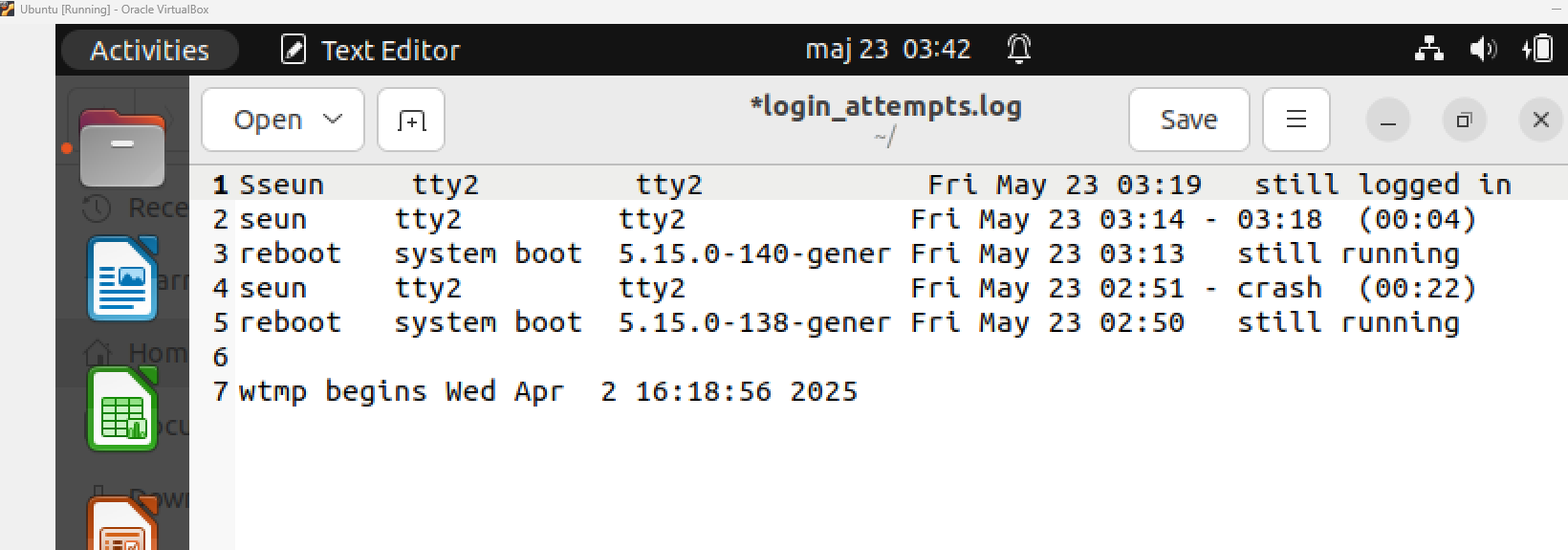
echo

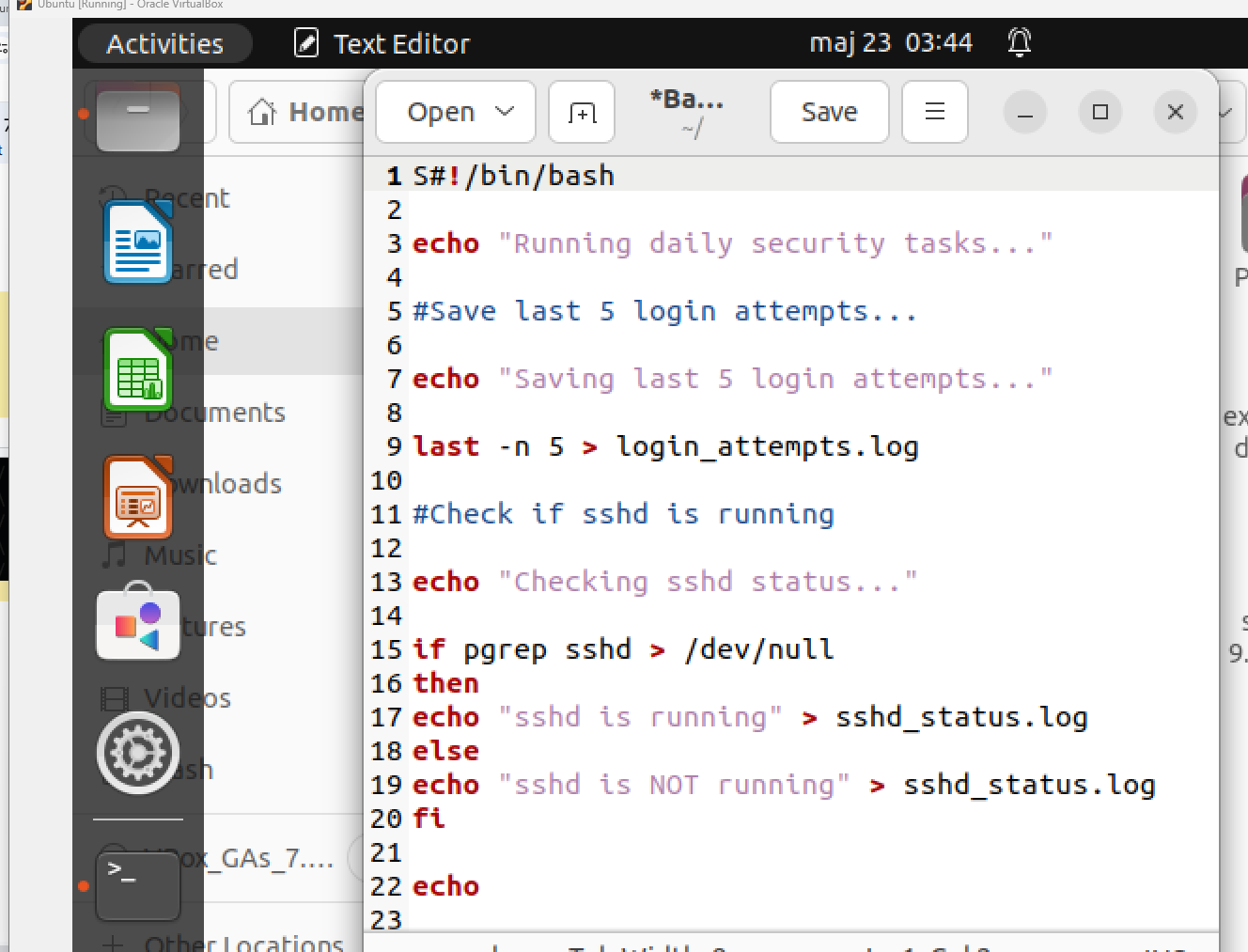
echo "✅ All tasks completed successfully."

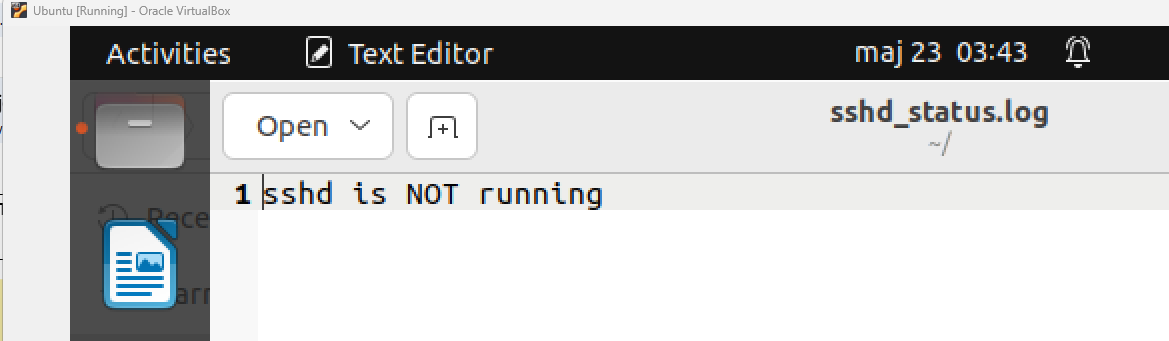
**Screenshot showing a successful task:**











**A summary of key findings, challenges faced, and resolutions.**

One of the issues I ran into was with the Windows CLI script—some characters in the disk space output were unreadable. I fixed that by adjusting how the command handled its output. I also had to make sure my Bash script had the right permissions to run, which I resolved using the chmod command.

**The impact of automation on system security and efficiency**

This project showed me how useful automation is in cybersecurity. It saves time, reduces mistakes, and gives consistent results. I now feel more confident writing basic scripts to support system monitoring and security tasks in real-world environments.