**Email:** meghanabadgujar07@gmail.com

**Name:** Meghana Samadhan Badgujar  
**Batch Name:** WiproNGA\_DWS\_B5\_25VID2550

**User ID:** 34737

**Batch ID:** 25VID2550

**Date:** 8th August 2025

**Assignment Topic:**

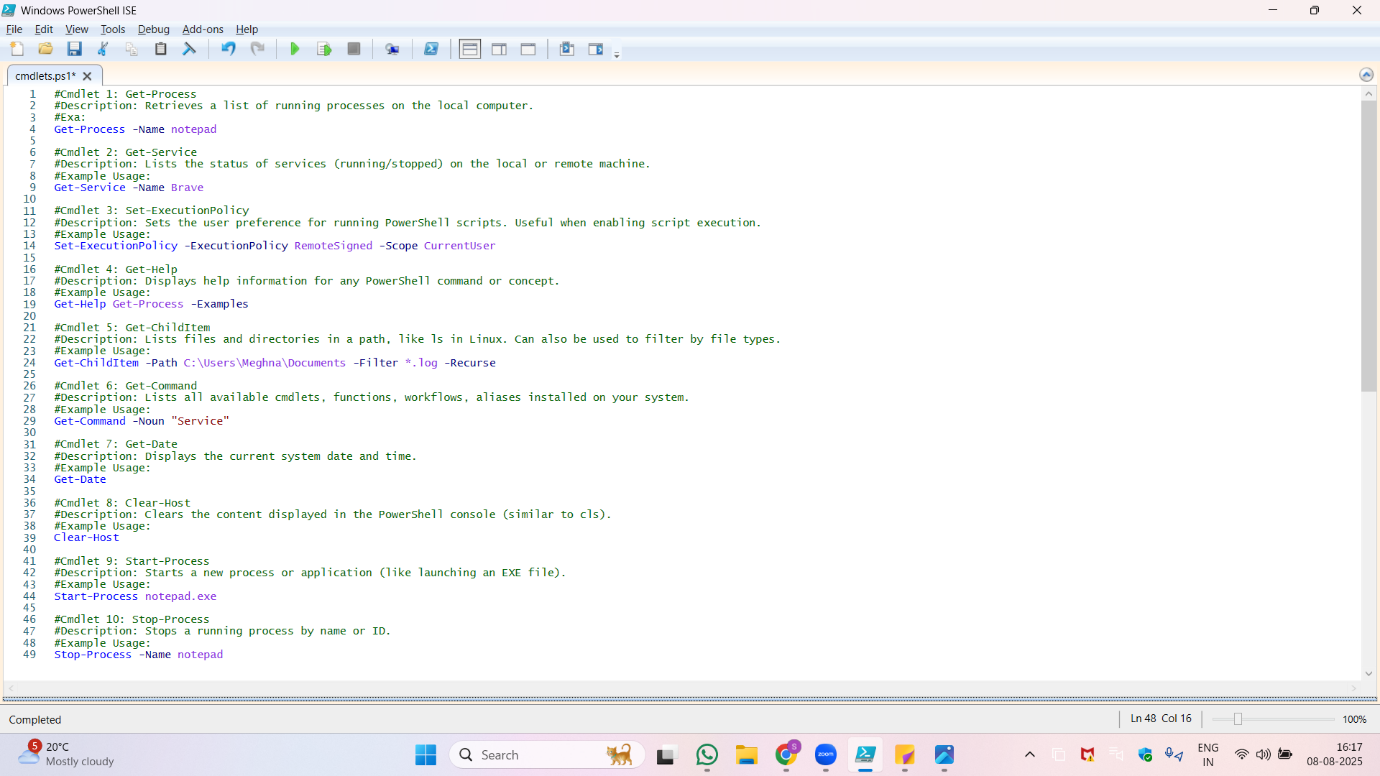
* **PowerShell Fundamentals**

**1. Introduction to Cmdlets:**

**Cmdlets** (pronounced ***command-lets***) are the building blocks of PowerShell scripting. They are small, task-specific commands that follow the Verb-Noun naming pattern (e.g., Get-Process, Set-Date) and are used to automate system administration tasks.

* **Key Characteristics:**
* **Structured Naming**: Verb-Noun format (e.g., Get-Service)
* **Pipeline-Compatible**: Accept input and pass output through the pipeline
* **Object-Oriented**: Work with .NET objects instead of plain text
* **Built-in & Custom**: You can use predefined cmdlets or write your own
* **Non-Executable**: Cmdlets are not standalone .exe files—they are part of PowerShell
* **Real-World Impact:**

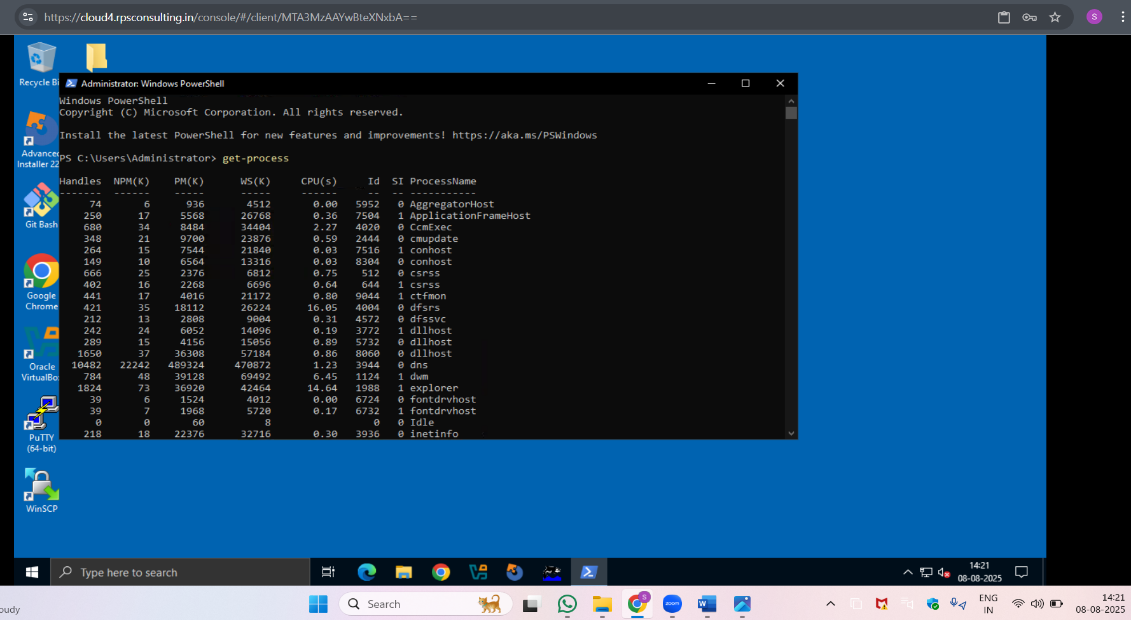
PowerShell cmdlets streamline server management, user provisioning, cloud automation, and routine tasks across industries (e.g., inventory checks, VM management).

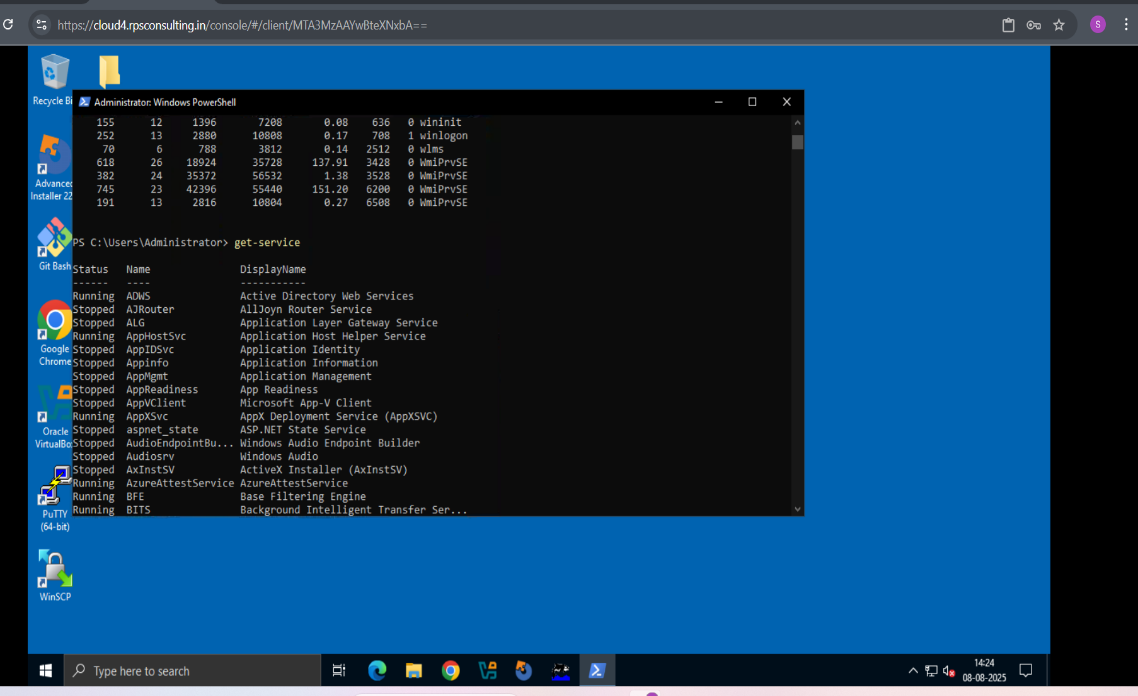


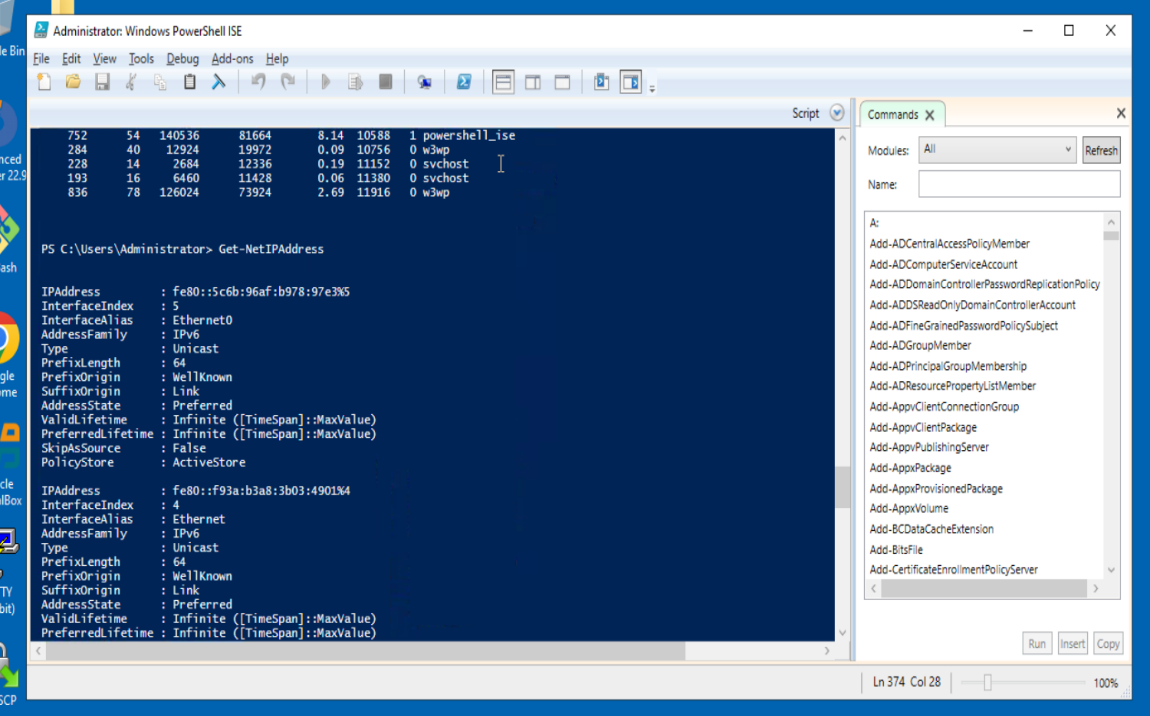
**2. Key Cmdlets to Know**

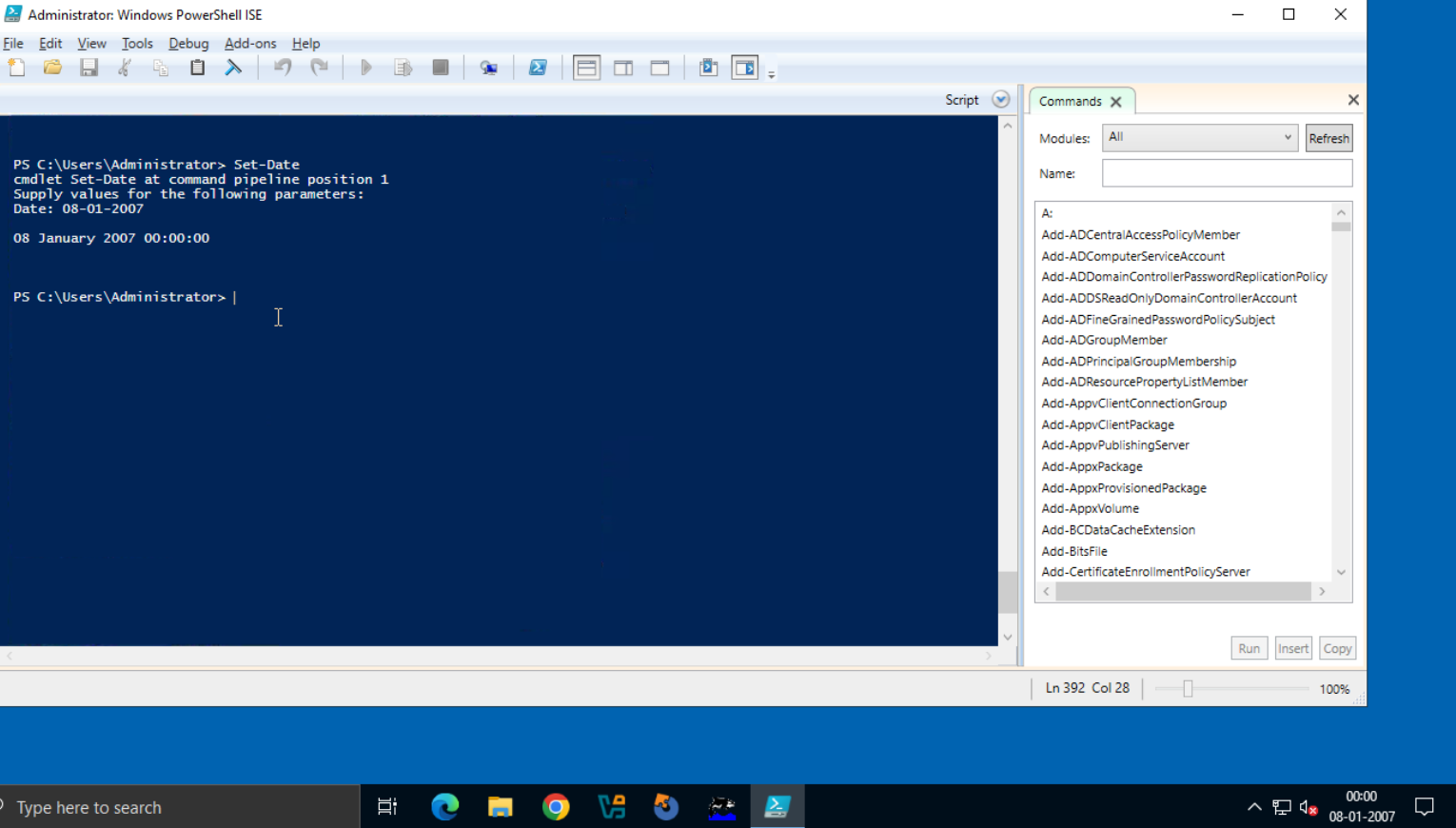
Here are some commonly used cmdlets:

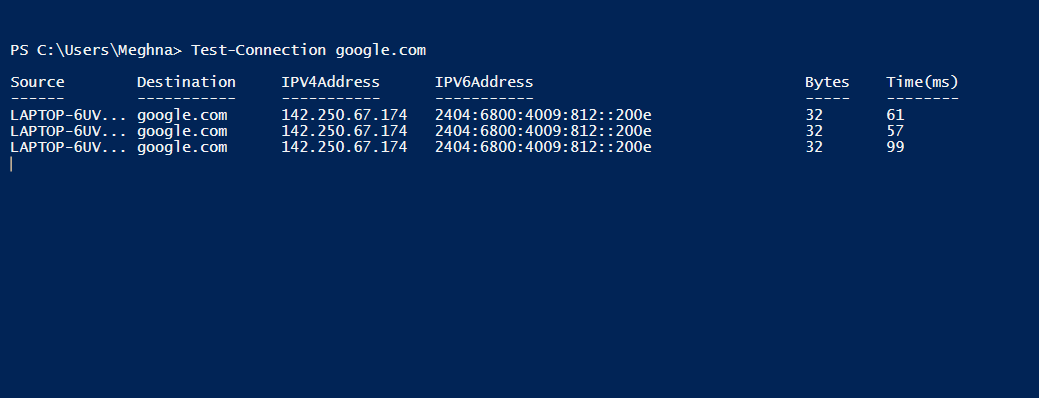
|  |  |  |
| --- | --- | --- |
| **Cmdlet** | **Description** | **Example Usage** |
| Get-Process | Lists running processes | Get-Process |
| Get-Service | Retrieves service statuses | Get-Service -Name wuauserv |
| Stop-Service | Stops a service | Stop-Service -Name spooler |
| Get-ChildItem | Lists files/folders (like dir) | Get-ChildItem C:\Temp |
| Get-EventLog | Displays recent event logs | Get-EventLog -LogName System |
| Set-ExecutionPolicy | Sets script execution policy | Set-ExecutionPolicy RemoteSigned |











**3. The PowerShell Pipeline**

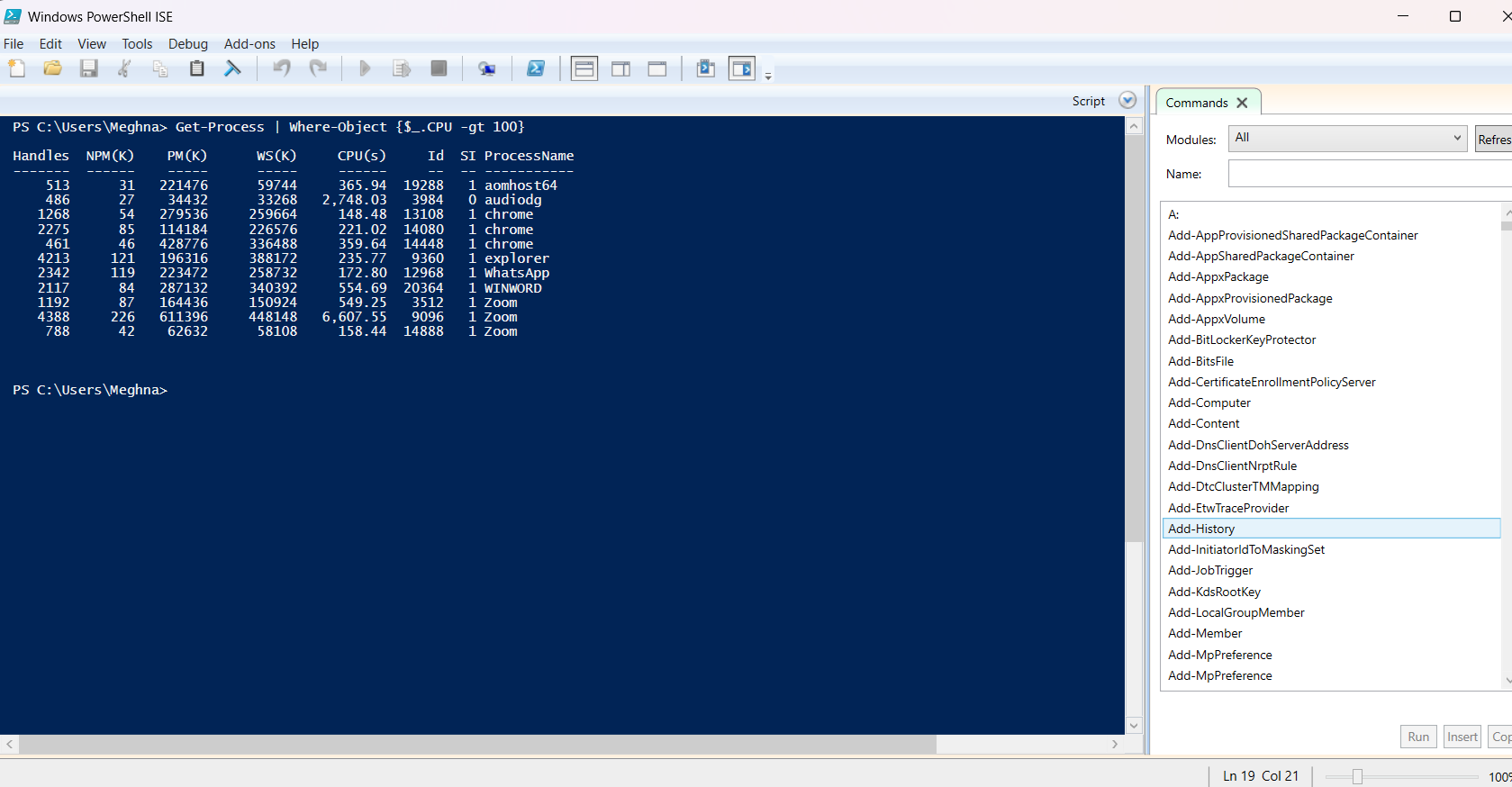
The **pipeline (|)** in PowerShell allows **chaining multiple cmdlets** together. The output of one cmdlet becomes the input for the next.

**Example:**

Get-Process | Where-Object {$\_.CPU -gt 100}

This command:

* Retrieves all running processes
* Filters those using **more than 100 units of CPU**



* **Pipeline Advantage:**
* Saves time and reduces manual effort
* Enables complex operations with simple syntax
* Makes scripts modular and readable

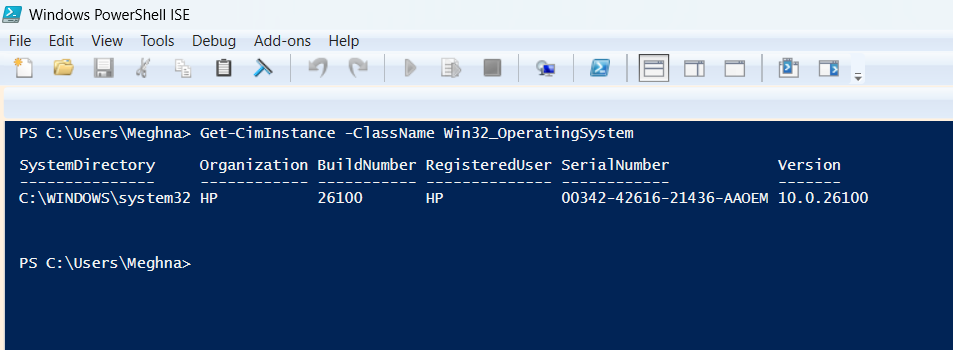
**4. WMI & PowerShell Integration**

**WMI (Windows Management Instrumentation)** provides low-level access to system information. PowerShell interacts with WMI to fetch and manage system data like hardware, OS details, services, and more.

* **Cmdlets to Use:**
* Get-WmiObject (older method)
* Get-CimInstance (modern, preferred method)
* **Example:**

Get-CimInstance -ClassName Win32\_OperatingSystem

- This returns details about the OS like version, install date, memory, etc.

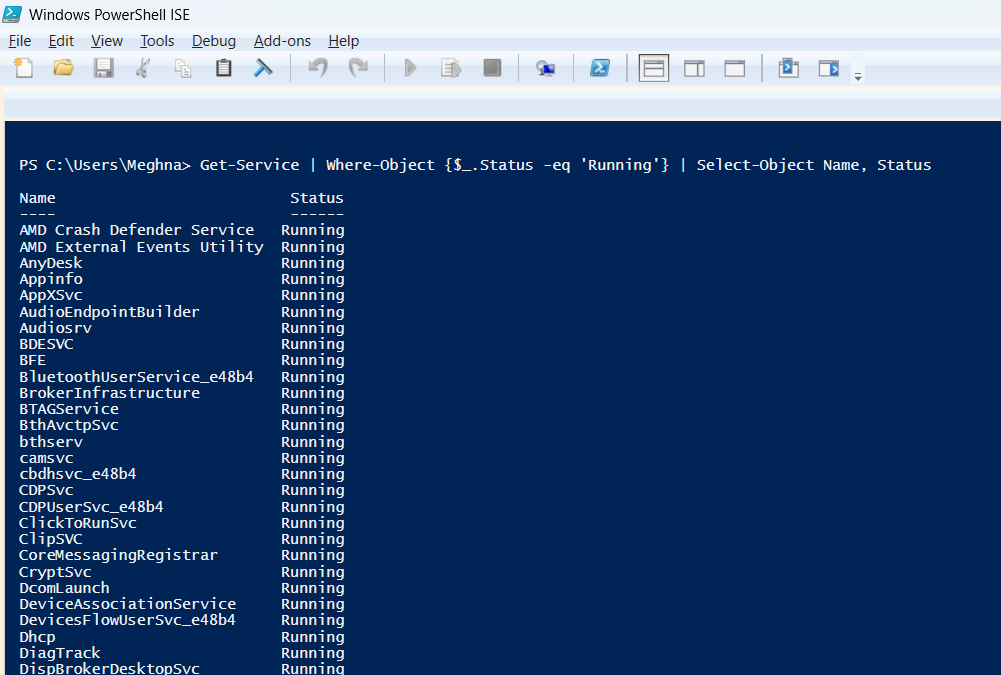


**5. Pipeline Filtering & Operators**

PowerShell offers filtering and logical operators to refine results in scripts.

* **Common Filters:**
* Where-Object – Filter items based on conditions
* Select-Object – Select specific properties
* **Logical Operators:**
* -eq (equals), -ne (not equal)
* -gt, -lt (greater than, less than)
* -like, -match (pattern matching)
* -and, -or, -not (logical combinations)
* **Example:**

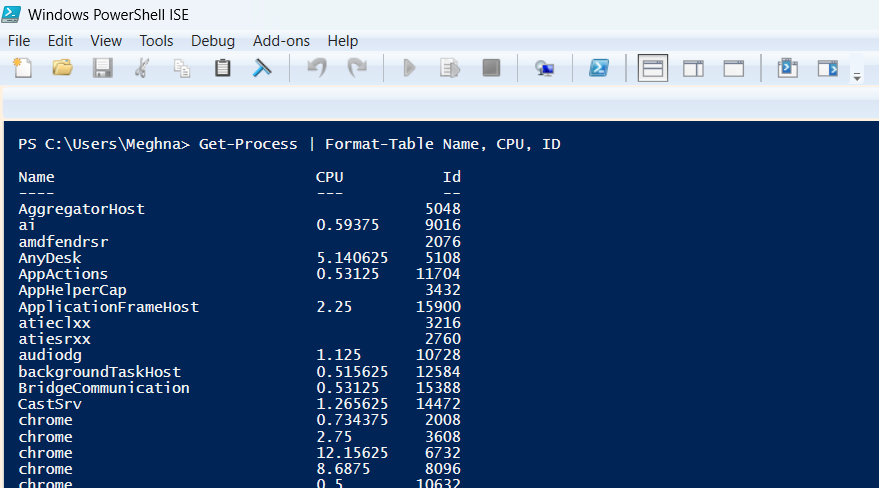
Get-Service | Where-Object {$\_.Status -eq 'Running'} | Select-Object Name, Status



**6. Input, Output & Formatting**

* **Input:**
* **User Input**: via Read-Host
* **File Input**: via Get-Content
* **Output:**
* PowerShell outputs objects by default
* You can **redirect output** to:
  + Files (Out-File)
  + CSVs (Export-Csv)
  + JSON (ConvertTo-Json)
* **Formatting:**
* Format-Table – tabular view
* Format-List – detailed list view
* **Example:**

Get-Process | Format-Table Name, CPU, ID



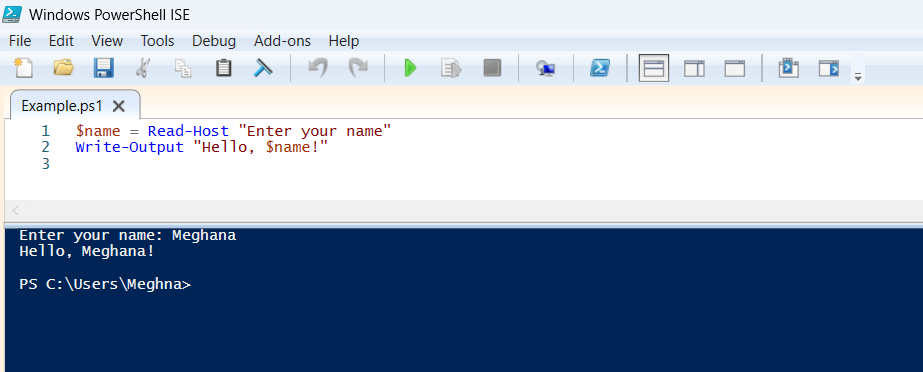
**7. Scripting Overview**

PowerShell scripts are saved with the .ps1 extension. A script can include variables, conditions, loops, and functions.

* **Sample Script:**

$name = Read-Host "Enter your name"

Write-Output "Hello, $name!"



* **Scripting Benefits:**
* Automate daily tasks
* Schedule scripts using Task Scheduler
* Reuse code across machines and teams