**NAME:-** Talari Vyshnavi

**USER ID:-** 34725

**EMAIL:-** [talari.vyshu@gmail.com](mailto:talari.vyshu@gmail.com)

**BATCH ID:-** 25VID2550

**BATCH NO:-** 5

**ASSIGNMENT**

**TOPICS:**-

* Introducing to Cmdlets
* The PowerShell Pipeline
* Key Cmdlets
* WMI & PowerShell
* Pipeline Filtering & Operators
* Input, Output & Formatting
* Scripting Overview
* **Introducing to Cmdlets** :-

Cmdlets are lightweighted commands used in windows PowerShell to perform specific tasks or operations. They are the building blocks of PowerShell scripting and automation.

* Key Features of Cmdlets:
* Cmdlets can pass data to one another using the pipeline.
* PowerShell comes with hundreds of built-in cmdlets.
* You can create custom cmdlets PowerShell or .NET.
* All cmdlets follow the Verb-Noun naming for readability and consistency.
* How cmdlets works:
* Input: Takes user input via parameters or pipeline.

Example- Get-Service -Name spooler.

* Processing: Executes logic using .NET

Example- Internally uses ProcessRecord ( ).

* Output: Returns .NET objects.

Example- Results can be piped or saved.

* **The PowerShell Pipeline :-**

The PowerShell pipeline (|) allows you to pass the ***output*** of one cmdlet as the ***input*** to another. It is a powerful feature that supports automations and scripting by ***chaining*** commands together.

Sample example- ***Get-process | Where -object { $\_.CPU -gt 100 } | Sort-object CPU -Descending***

A computer screen with a black screen

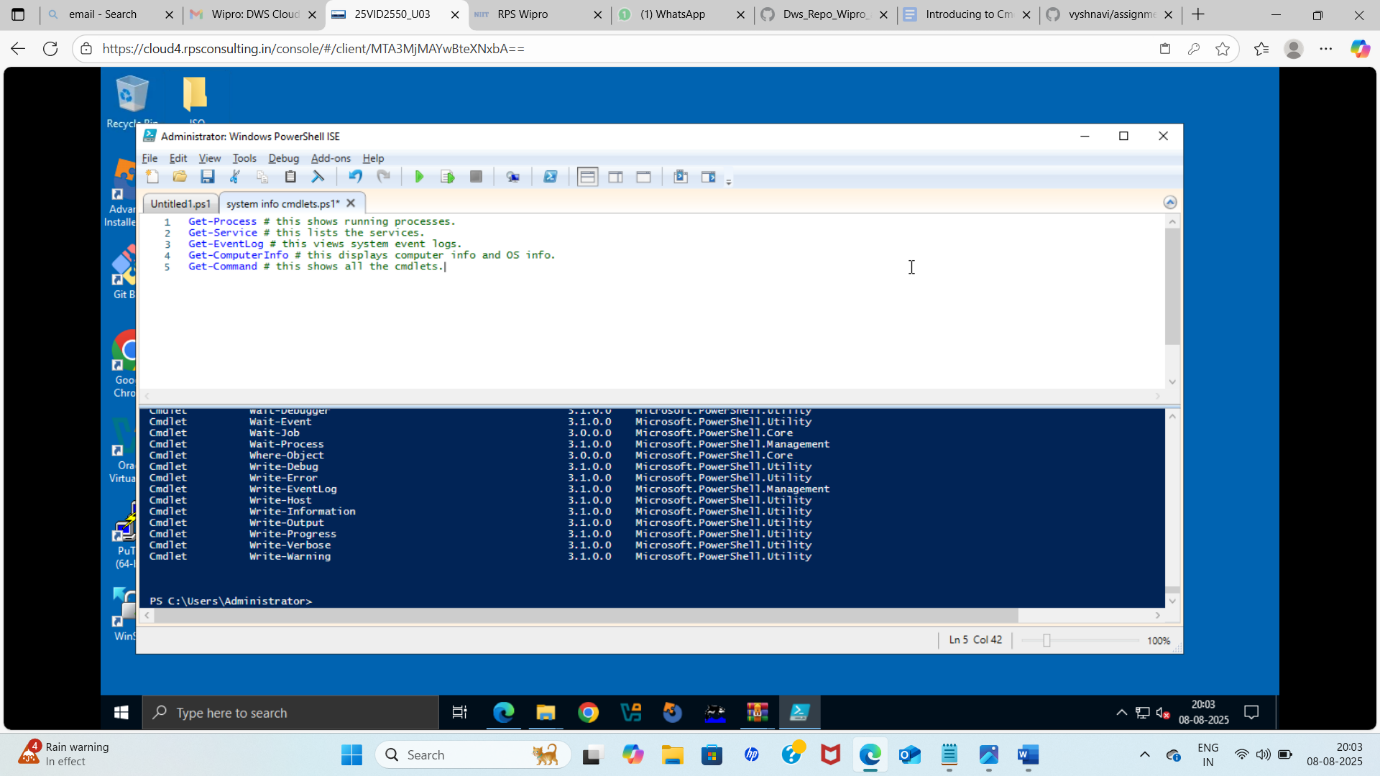
AI-generated content may be incorrect.

Result for the above example.

* **Key Cmdlets :-**

**System information :**

* + - Get-Process :- Shows running processes
    - Get-Service :- Lists service on the system
    - Get-EventLog :- Views system event logs
    - Get-ComputerInfo :-Displays system hardware & Os info
    - Get-Command :- Lists all available cmdlets



* + - Get-Childerns
    - Start-Services
    - Stop-Services
    - Set-Location
    - Set-Content
    - Copy-Item
  + Move-Item
  + Set-Content
  + Get-Help
  + Get-ADUser
  + Select-ADUser
  + Where-Object
  + Set-Object
  + Set-Date
* **WMI** stands for Windows Management Instrumentation, WMI is a Microsoft technology **WMI & PowerShell**

used to access and manage the internal datd of windows-based systems.

Think of it as a bridge to get detailed system info like CPU, memory, disk, OS, drivers, etc…

WMI uses a database of classes like:

Win32\_OperatingSystem

Win32\_LogicalDisk

Win32\_Service

Win32\_Process

**POWERSHELL**

Understanding PowerShell cmdlets is a critical step toward mastering automation and scripting in Windows environments. By diving into hands-on projects, you’ll engage more actively with PowerShell, which can greatly improve your comprehension and retention of cmdlet structures and functions. Practical exercises allow you to learn from real-world scenarios, deepen your understanding, and build your confidence.

Let’s explore some engaging projects that will solidify your understanding of PowerShell cmdlets.

These are commands in ***powershell***

* + **Pipeline Filtering & Operators**

One of the most powerful features of PowerShell is the ability to use Cmdlets in pipelines. This means you can chain multiple Cmdlets together, passing the output of one as the input to the next.

**Example of a Pipeline**

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

* **Input, Output & Formatting**

Input :- Input refers to the data or parameters you pass into a cmdlet, script, or function.

Example-Get-Srevice -Name

Output :- Output is the result produced by a cmdlet, function, or script. It could be:

* A text result
* An object
* A list or table

Formatting :- Formatting control how output is displayed. Powershell automatically formats output as:

* Table
* List
* Wide

* **Scripting Overview :-**

PowerShell scripting is the process of writing and executing a series as powershell commands in a file to automate tasks.

A PowerShell scripting is a text file containing a sequence of cmdlets, functions, logic and comments. It helps in automating repetitive administrative tasks.