NAME:- Talari Vyshnavi

USER ID:- 34725

BATCH ID:- 25VID2550- B5

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

**ASSIGNMENT**

TOPIC :-

* PowerShell cmdlets
* How to run Objects, Arrays, Variables in PowerShell
* Modularization
* Understand basic platform value, concepts and usage
* **PowerShell cmdlets:-**

PowerShell cmdlets ( pronounced command-lets ) are small, single-function commands built into PowerShell.

They are the basic building blocks you use to perform tasks – such as retrieving data, configuring setting, or automating processes.

Key characteristics:

* Verd-Noun Naming
* Consistent Syntax
* Object-Based
* Pipeline Support
  + **How to run Objects, Arrays, Variables in PowerShell:-**

Variables in PowerShell –

* Variables start with a $ symbol.
* You can store text, numbers, or objects in them.

**Example:-** $Name = “Vyshnavi”

Arrays in PowerShell **–**

* + - Arrays store multiple value in one variable.
    - Arrays start with @( ) or can be created by assigning multiple values.

**Example** :- $Fruits = @( “Apple”, “Banan”, “Cherry” )

Objects in PowerShell –

* Objects are more structured – they have properties and methods.
* Many cmdlets return objects, not just text.

**Example:**- $Proc = Get-Process | Select- object -First1

Example to run Objects, Arrays and variables in PowerShell:-

Get-Process

$name = Read-Host ”please enter name”

Write-Host "Hello, $name!"A screenshot of a computer

AI-generated content may be incorrect.

* **Modularization :-**

Modularization means breaking your code into smaller, reusable, Well-organized parts instead of writing everything in one big script.

Importance:-

Reusability – Write once use many times.

Readability – Easier for you to understand.

Maintainability – Fix/update one module without touching everything.

Collaboration – Different people can work on different modules.

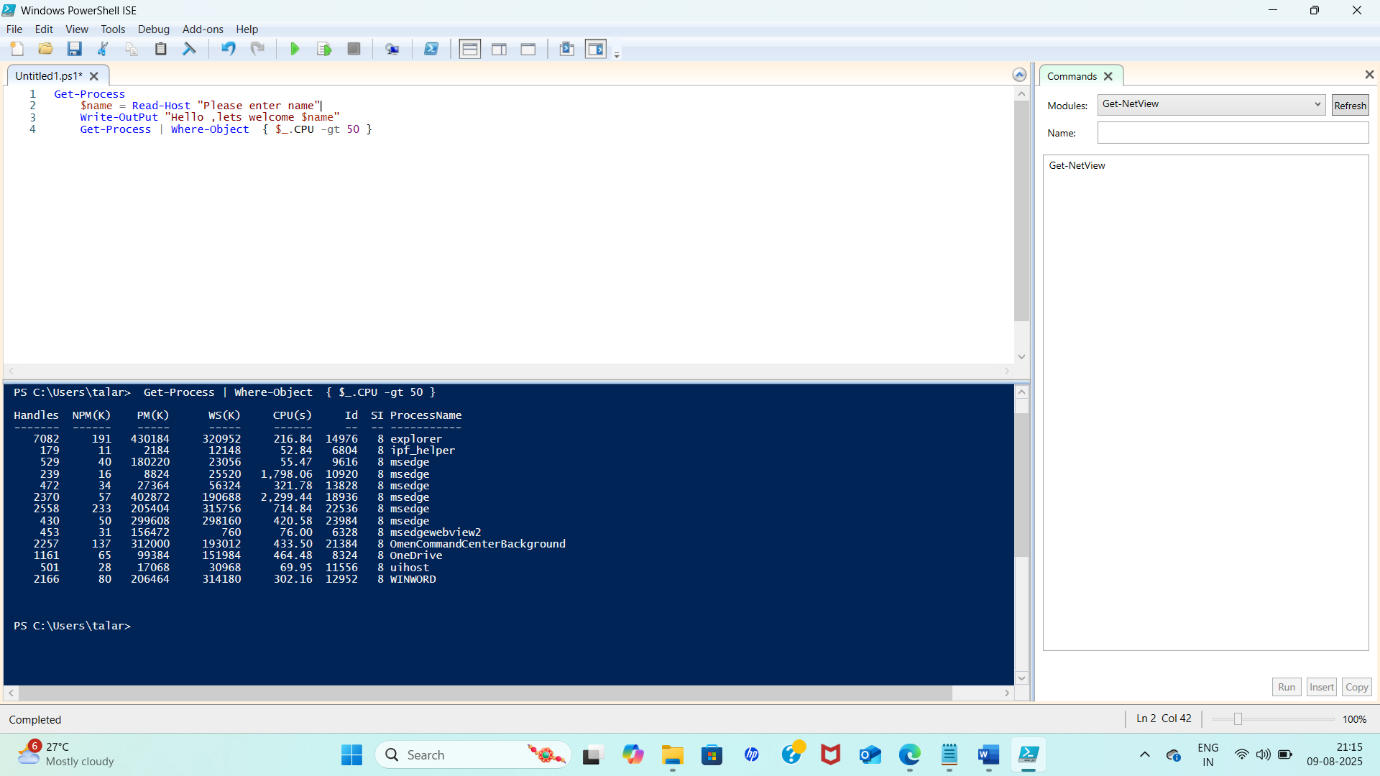
**Way to do Modularization in PowerShell**

Functions :- Put repeated logic in a function so you can call it anytime.

Script :- Save code in a .ps1 file and reuse it by running ( .\MyScript .ps1 ).

Modules :- Store related functions/cmdlets in a .psm1 file. Import with – Import-Module MyModule .psm1 .

Dot Scouring :- Lets you run another script in the current session so its variables/functions are available ( . .\HelperFunctions.ps1 ).

Example:- Get-Process | Where-Object { $\_.CPU -gt 50 }

* **Understand basic platform value, concepts and usage :-**

Usually means getting familiar with the core principles, the key ideas, and how to actually work with the platform effectively.

* Platform values:-

These are the principles and benefits the platform is built around. For most IT platforms, the core values are:

* + Reliability – It should work consistently without failures.
  + Security – Keeps your data safe from unauthorized access.
  + Scalability – Can handle more users or workloads as needed.
  + Flexibility – Supports different tasks and use cases.
  + Efficiency – Saves time and resources compared to doing things manually.
* Platform Concepts :-

These are the foundational ideas you need to understand before using platform . for example:

* Users & roles
* Permissions
* Resources
* Automation
* Integration
* Platform Usage :-

This is the practical side – how you actually use the platform day-to-day.

* Logging in & Navigating – Knowing how to sign in, use menus, and find features.
* Running Commands or Actions -For example, in PowerShell, using Get-Command to discover cmdlets.
* Managing Resources – Creating, modifying, or deleting items in the platform.
* Following Best Practices – Using security, naming conventions, and backups.