Network labs
Next level network for great careers

# PowerShell

You are the PowerShell Administrator at **Contoso IT Services**, and you've recently been assigned to automate tasks for a new project supporting a hybrid Active Directory and Windows infrastructure. Your automation tasks must support audit compliance, system maintenance, and AD user provisioning based on user input.

You are expected to use **PowerShell scripting**, **conditional logic**, **WMI**, **event logs**, and **error handling techniques** to accomplish the following tasks.

**Q1:** Write a PowerShell script that performs the following:

1. Prompts the administrator for user inputs:
    - First Name, Last Name, Department, Title, Password
2. Automatically builds the following:
    - Display Name = <FirstName> <LastName>
    - SamAccountName = <FirstInitial><LastName>
3. Creates an **Organizational Unit** (OU) named: Contoso-Staff-Users (if it doesn't exist).
4. Creates a new AD user in that OU with:
    - Password never expires
    - Must change password at next logon
    - Enabled by default

5.Export the details of the created user into a .CSV file with the current date in the filename.

**Q2:** You're tasked with monitoring Windows Services and System Processes on a remote server as part of routine health checks.

Network labs
Next level network for great careers

**Instructions:**

Create a script that does the following using **if, elseif, and else** conditions:

1. Ask the user for the target **computer name**.
2. Check if the computer is reachable using Test-Connection.
3. If reachable:
    - List all **stopped services**.
    - List all **running processes with CPU usage above 10%**.

4.Log the output in a file named HealthCheck_<Date>.log in C:\Logs.

**Q3:** As part of system performance analysis, you need to log boot times and store output systematically.

1. Start a PowerShell Transcript session using the following customized filename format: <YourName>-BootCheck-<TimeStamp>.log

Use current time (Get-Date) to generate the timestamp dynamically and sanitize it for file naming.

2. Query the system using WMI to get the last boot time:

$BootTimeRaw = (Get-CimInstance -ClassName Win32_OperatingSystem).LastBootUpTime

3. Convert $BootTimeRaw to a proper DateTime object and calculate the **uptime** in hours.
4. Print:
    - Last boot time
    - Current system time
    - Total uptime in hours