**Tarea 4: Script en PowerShell para validar el estado de recursos en la máquina Windows**

Para validar el estado de los recursos de una maquina en Windows, se desarrolló el siguiente código de PowerShell:

# Obtener la fecha y hora actual

$currentDate = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

# Obtener el uso de CPU

$cpuLoad = Get-WmiObject win32\_processor | Measure-Object -Property LoadPercentage -Average | Select -ExpandProperty Average

$cpuUsage = [math]::Round($cpuLoad, 2)

# Obtener el uso de RAM

$totalMemory = (Get-WmiObject -Class Win32\_ComputerSystem).TotalPhysicalMemory

$freeMemory = (Get-WmiObject -Class Win32\_OperatingSystem).FreePhysicalMemory

$usedMemory = $totalMemory - ($freeMemory \* 1024)

$usedMemoryGB = [math]::Round($usedMemory / 1GB, 2)

$totalMemoryGB = [math]::Round($totalMemory / 1GB, 2)

# Obtener el espacio en disco

$drives = Get-PSDrive -PSProvider FileSystem | Where-Object { $\_.Used -gt 0 }

$diskInfo = foreach ($drive in $drives) {

    [PSCustomObject]@{

        Name = $drive.Name

        UsedSpaceGB = [math]::Round($drive.Used / 1GB, 2)

        FreeSpaceGB = [math]::Round($drive.Free / 1GB, 2)

        TotalSpaceGB = [math]::Round($drive.Used / 1GB + $drive.Free / 1GB, 2)

    }

}

# HTML

$htmlContent = @"

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Reporte del Estado de Recursos del Sistema</title>

    <style>

        body { font-family: Arial, sans-serif; }

        h1 { color: #333; }

        table { width: 100%; border-collapse: collapse; margin-top: 20px; }

        th, td { padding: 10px; border: 1px solid #ccc; text-align: left; }

        th { background-color: #f4f4f4; }

    </style>

</head>

<body>

    <h1>Reporte del Estado de Recursos del Sistema</h1>

    <p>Fecha y Hora: $currentDate</p>

    <h2>Uso de CPU</h2>

    <p>Uso de CPU: $cpuUsage%</p>

    <h2>Uso de RAM</h2>

    <p>Memoria Usada: $usedMemoryGB GB</p>

    <p>Memoria Total: $totalMemoryGB GB</p>

    <h2>Espacio en Disco</h2>

    <table>

        <tr>

            <th>Unidad</th>

            <th>Espacio Usado (GB)</th>

            <th>Espacio Libre (GB)</th>

            <th>Espacio Total (GB)</th>

        </tr>

"@

foreach ($disk in $diskInfo) {

    $htmlContent += @"

        <tr>

            <td>$($disk.Name)</td>

            <td>$($disk.UsedSpaceGB)</td>

            <td>$($disk.FreeSpaceGB)</td>

            <td>$($disk.TotalSpaceGB)</td>

        </tr>

"@

}

$htmlContent += @"

    </table>

</body>

</html>

"@

# Guardar el resultado en un archivo HTML

$outputPath = "recursos\_sistema\_windows.html"

$htmlContent | Out-File -FilePath $outputPath -Encoding utf8

Write-Output "Reporte generado: $outputPath"

Para ejecutar el script, primero se tiene que guardar con la extensión .ps1, como se muestra en la imagen de ejemplo



Una vez almacenado, se procede a ejecutarlo con el comando . \ + el nombre del script, que en este caso sería el siguiente:  


Una vez ejecutado, el debería de obtener un resultado en un archivo HTML como se muestra en la siguiente imagen



Procedemos a verificar los datos obtenidos dando doble clic al archivo .html y nos debería de mostrar los datos de la siguiente manera:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Con esta validación, podemos concluir que el script está funcionando correctamente y que nos está mostrando la información que realmente necesitamos.