Nome do computador: CEOSOFT-059
Tipo: Desktop
Gerado por: alex

Relatório gerado em: 02/10/2025 - 17:58

INFORMAÇÕES DO SISTEMA

Versão do sistema operacional: Microsoft Windows 11 Pro (Version 10.0.26100) - 64 bits
Status do sistema operacional: Ativado

Versão do Office: Microsoft Office Professional Plus 2019 - pt-br 16.0.19127.20264
Status do Office: Ativado

SQL Server 1: Instância: Default | Versão: 16.0.1000.6 | Status: Stopped

Antivírus 1: Kaspersky | Status: Ativado
Antivírus 2: Windows Defender | Status: Desconhecido

Rede: Domínio: ceosoftwaread.net

INFORMAÇÕES DE HARDWARE

Memória RAM total: 7.87 GB
Pente de Memória 1: 8 GB | DDR4 | 2400 MHz | DIMM
Fabricante: 8945

Processador 1: Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz
Cores: 4 físicos | 8 lógicos
Cache: L2: 1 MB | L3: 8 MB
Fabricante: GenuineIntel

Disco 1: ST1000DM010-2EP102 | Tamanho: 931.51 GB
Tipo: HDD | Interface: IDE
Disco 2: KINGSTON SA400S37960G | Tamanho: 894.25 GB
Tipo: HDD | Interface: IDE
Unidade C: (Sem rótulo) | Total: 893.51 GB | Usado: 558.26 GB | Livre: 335.25 GB | Sistema: NTFS
Unidade D: (Sem rótulo) | Total: 465.37 GB | Usado: 318.79 GB | Livre: 146.58 GB | Sistema: NTFS
Unidade E: (Partição 2) | Total: 435.29 GB | Usado: 135.91 GB | Livre: 299.38 GB | Sistema: NTFS
Unidade F: (Swapping) | Total: 29.30 GB | Usado: 1.76 GB | Livre: 27.54 GB | Sistema: NTFS

Monitor 1: DEL | Modelo: DELL E1910 | Serial: X234R03V0SYL
Monitor 2: GSM | Modelo: LG FULL HD | Serial: 16843009

Teclado conectado: SIM
Mouse conectado: SIM

Placa mãe: ASUSTeK COMPUTER INC. | Modelo: H110M-C/BR | Serial: 180414466200454

Placa de Rede 1: Realtek PCIe GBE Family Controller | Fabricante: Realtek | Velocidade: 100 Mbps | MAC: 2C:FD:A1:C8:77:61

Placa de Vídeo 1: BB Capture Driver | Fabricante: Blueberry Consultants | Memória: N/A | Tipo: Onboard (Integrada)
Placa de Vídeo 2: NVIDIA GeForce GT 220 | Fabricante: NVIDIA | Memória: 1 GB | Tipo: Offboard (Dedicada)
Placa de Vídeo 3: Intel(R) HD Graphics 630 | Fabricante: Intel Corporation | Memória: 1 GB | Tipo: Onboard (Integrada)

Impressora 1: OneNote (Desktop) | Serial/ID: NÃO OBTIDO | Fabricante: NÃO OBTIDO | Modelo: Send to Microsoft OneNote 16 Driver
Impressora 2: NPI1F7468 (HP LaserJet Professional M1212nf MFP) | Serial/ID: NÃO OBTIDO | Fabricante: NÃO OBTIDO | Modelo: HP LaserJet Mono PCLmS Class Driver
Impressora 3: Microsoft Print to PDF | Serial/ID: NÃO OBTIDO | Fabricante: NÃO OBTIDO | Modelo: Microsoft Print To PDF
Impressora 4: HP LaserJet 1020 | Serial/ID: NÃO OBTIDO | Fabricante: NÃO OBTIDO | Modelo: HP LaserJet 1020

ADMINISTRADORES

Administrador 1: CEOSOFT-059\Administrador
Administrador 2: CEOSOFT-059\CEOSOFTWARE
Administrador 3: CEOSOFTWAREAD\Admins. do domínio
Administrador 4: CEOSOFTWAREAD\alex
Administrador 5: CEOSOFTWAREAD\Angelo.Gabriel
Administrador 6: CEOSOFTWAREAD\geovani
Administrador 7: CEOSOFTWAREAD\Marco.Aurelio

SOFTWARES INSTALADOS

1. Android Studio | Versão: 2025.1 | Editor: Google LLC
2. Arquivo do WinRAR | Versão: N/A | Editor: N/A
3. Arquivos de Suporte à Instalação do Microsoft SQL Server 2008 | Versão: 10.3.5500.0 | Editor: Microsoft Corporation
4. Atualizações da NVIDIA 10.4.0 | Versão: 10.4.0 | Editor: NVIDIA Corporation
5. BraZip Central | Versão: 25.0.3 | Editor: BraZip Tecnologia
6. CSServiceCenter | Versão: 1.0.0 | Editor: CEOSoftware Sistemas de Informática
7. Data Dynamics ActiveReports Professional 2 | Versão: 2.3.0.1261 | Editor: Data Dynamics, Ltd.
8. dbForge SQL Complete, v6.15.5 | Versão: 6.15.5 | Editor: Devart
9. DiagnosticsHub_CollectionService | Versão: 17.14.36412 | Editor: Microsoft Corporation
10. Eclipse Temurin JDK with Hotspot 21.0.8+9 (x64) | Versão: 21.0.8.9 | Editor: Eclipse Adoptium
11. Eclipse Temurin JDK with Hotspot 8u462-b08 (x64) | Versão: 8.0.462.8 | Editor: Eclipse Adoptium
12. ERP CEOSoftware 8_2 | Versão: 8.2 | Editor: CEOSoftware Sistemas de Informática
13. Ferramentas de Build do Visual Studio 2019 | Versão: 16.11.51 | Editor: Microsoft Corporation
14. FileZilla 3.69.3 | Versão: 3.69.3 | Editor: Tim Kosse
15. Flashback Express 7 | Versão: 7.7.0.445 | Editor: Blueberry Software
16. GatewayComponents | Versão: 16.22.5 | Editor: Microsoft Corporation
17. Git | Versão: 2.51.0.2 | Editor: The Git Development Community
18. Google Chrome | Versão: 140.0.7339.208 | Editor: Google LLC
19. Gravador VSS da Microsoft para SQL Server 2022 | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
20. HP LaserJet 1020 Series | Versão: N/A | Editor: N/A
21. Instalação do Microsoft SQL Server 2014 (em inglês) | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
22. Kaspersky | Versão: 21.22.7.466 | Editor: Kaspersky
23. Microsoft .NET Framework 4 Multi-Targeting Pack | Versão: 4.0.30319 | Editor: Microsoft Corporation
24. Microsoft .NET Host - 8.0.20 (x64) | Versão: 64.80.39230 | Editor: Microsoft Corporation
25. Microsoft .NET Host FX Resolver - 8.0.20 (x64) | Versão: 64.80.39230 | Editor: Microsoft Corporation
26. Microsoft .NET Runtime - 8.0.20 (x64) | Versão: 64.80.39230 | Editor: Microsoft Corporation
27. Microsoft Application Error Reporting | Versão: 12.0.6012.5000 | Editor: Microsoft Corporation
28. Microsoft Edge | Versão: 140.0.3485.94 | Editor: Microsoft Corporation
29. Microsoft Edge WebView2 Runtime | Versão: 140.0.3485.94 | Editor: Microsoft Corporation
30. Microsoft Fabric Integration Runtime | Versão: 5.55.9295.2 | Editor: Microsoft Corporation
31. Microsoft Help Viewer 1.1 | Versão: 1.1.40219 | Editor: Microsoft Corporation
32. Microsoft Help Viewer 1.1 Language Pack - PTB | Versão: 1.1.40219 | Editor: Microsoft Corporation
33. Microsoft ODBC Driver 11 for SQL Server | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
34. Microsoft ODBC Driver 17 for SQL Server | Versão: 17.10.6.1 | Editor: Microsoft Corporation
35. Microsoft Office 2003 Web Components | Versão: 11.0.8173.0 | Editor: Microsoft Corporation
36. Microsoft Office Professional Plus 2019 - pt-br | Versão: 16.0.19127.20264 | Editor: Microsoft Corporation
37. Microsoft OLE DB Driver for SQL Server | Versão: 18.7.4.0 | Editor: Microsoft Corporation
38. Microsoft Report Viewer 2014 Runtime | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
39. Microsoft SQL Server 2005 (64-bit) | Versão: N/A | Editor: Microsoft Corporation
40. Microsoft SQL Server 2005 Analysis Services (64-bit) (SQL2005) | Versão: 9.00.1399.06 | Editor: Microsoft Corporation
41. Microsoft SQL Server 2005 Backward compatibility | Versão: 8.05.1054 | Editor: Microsoft Corporation
42. Microsoft SQL Server 2005 Tools (64-bit) | Versão: 9.00.1399.06 | Editor: Microsoft Corporation
43. Microsoft SQL Server 2008 R2 Management Objects | Versão: 10.51.2500.0 | Editor: Microsoft Corporation
44. Microsoft SQL Server 2012 Native Client | Versão: 11.0.2100.60 | Editor: Microsoft Corporation
45. Microsoft SQL Server 2014 (64 bits) | Versão: N/A | Editor: N/A
46. Microsoft SQL Server 2014 Transact-SQL ScriptDom | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
47. Microsoft SQL Server 2022 (64-bit) | Versão: N/A | Editor: N/A

48. Microsoft SQL Server 2022 RsFx Driver | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
49. Microsoft SQL Server 2022 Setup (English) | Versão: 16.0.4212.1 | Editor: Microsoft Corporation
50. Microsoft SQL Server Native Client | Versão: 9.00.1399.06 | Editor: Microsoft Corporation
51. Microsoft SQL Server Setup Support Files (English) | Versão: 9.00.1399.06 | Editor: Microsoft Corporation
52. Microsoft SQL Server System CLR Types | Versão: 10.51.2500.0 | Editor: Microsoft Corporation
53. Microsoft System CLR Types para SQL Server 2014 | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
54. Microsoft Teams Meeting Add-in for Microsoft Office | Versão: 1.24.31301 | Editor: Microsoft
55. Microsoft Visual C++ 2010 x86 Runtime - 10.0.40219 | Versão: 10.0.40219 | Editor: Microsoft Corporation
56. Microsoft Visual C++ 2013 x64 Additional Runtime - 12.0.40664 | Versão: 12.0.40664 | Editor: Microsoft Corporation
57. Microsoft Visual C++ 2013 x64 Minimum Runtime - 12.0.40664 | Versão: 12.0.40664 | Editor: Microsoft Corporation
58. Microsoft Visual C++ 2019 X64 Debug Runtime - 14.29.30157 | Versão: 14.29.30157 | Editor: Microsoft Corporation
59. Microsoft Visual C++ 2019 X86 Debug Runtime - 14.29.30157 | Versão: 14.29.30157 | Editor: Microsoft Corporation
60. Microsoft Visual C++ 2022 X64 Additional Runtime - 14.44.35211 | Versão: 14.44.35211 | Editor: Microsoft Corporation
61. Microsoft Visual C++ 2022 X64 Minimum Runtime - 14.44.35211 | Versão: 14.44.35211 | Editor: Microsoft Corporation
62. Microsoft Visual C++ 2022 X86 Additional Runtime - 14.44.35211 | Versão: 14.44.35211 | Editor: Microsoft Corporation
63. Microsoft Visual C++ 2022 X86 Minimum Runtime - 14.44.35211 | Versão: 14.44.35211 | Editor: Microsoft Corporation
64. Microsoft Visual Studio 2010 Shell (Isolated) - PTB | Versão: 10.0.40219 | Editor: Microsoft Corporation
65. Microsoft Visual Studio Installer | Versão: 3.14.2086.54749 | Editor: Microsoft Corporation
66. Microsoft Visual Studio Setup Configuration | Versão: 3.12.2140.44225 | Editor: Microsoft Corporation
67. Microsoft Visual Studio Setup WMI Provider | Versão: 3.12.2140.44225 | Editor: Microsoft Corporation
68. Navegador para SQL Server 2022 | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
69. Node.js | Versão: 22.19.0 | Editor: Node.js Foundation
70. Npgsql 3.2.6 | Versão: 3.2.6-3 | Editor: EnterpriseDB
71. NVIDIA Driver de áudio HD 1.3.30.1 | Versão: 1.3.30.1 | Editor: NVIDIA Corporation
72. NVIDIA Driver de gráficos 342.01 | Versão: 342.01 | Editor: NVIDIA Corporation
73. NVIDIA Driver do 3D Vision 342.01 | Versão: 342.01 | Editor: NVIDIA Corporation
74. NVIDIA Install Application | Versão: 2.1002.197.1706 | Editor: NVIDIA Corporation
75. NVIDIA Stereoscopic 3D Driver | Versão: 7.17.12.6514 | Editor: NVIDIA Corporation
76. Office 16 Click-to-Run Extensibility Component | Versão: 16.0.19127.20154 | Editor: Microsoft Corporation
77. Office 16 Click-to-Run Extensibility Component 64-bit Registration | Versão: 16.0.19127.20154 | Editor: Microsoft Corporation
78. Office 16 Click-to-Run Licensing Component | Versão: 16.0.19127.20264 | Editor: Microsoft Corporation
79. Office 16 Click-to-Run Localization Component | Versão: 16.0.19127.20154 | Editor: Microsoft Corporation
80. On-premises data gateway | Versão: 3000.278.5 | Editor: Microsoft Corporation
81. OpenVPN 2.4.6-I602 | Versão: 2.4.6-I602 | Editor: OpenVPN Technologies, Inc.
82. Oracle VirtualBox 7.2.2 | Versão: 7.2.2 | Editor: Oracle and/or its affiliates
83. Painel de controle da NVIDIA 342.01 | Versão: 342.01 | Editor: NVIDIA Corporation
84. pgAgent_PG17 4.2.3 | Versão: 4.2.3-1 | Editor: EnterpriseDB
85. PgBouncer 1.24.1 | Versão: 1.24.1-1 | Editor: EnterpriseDB
86. pgJDBC 42.7.2 | Versão: 42.7.2-1 | Editor: EnterpriseDB
87. Políticas do Microsoft SQL Server 2014 | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
88. PostgreSQL 17 | Versão: 17.6-1 | Editor: PostgreSQL Global Development Group
89. psqlODBC 13.02.0000 | Versão: 13.02.0000-1 | Editor: EnterpriseDB
90. Python 3.13.7 Add to Path (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
91. Python 3.13.7 Core Interpreter (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
92. Python 3.13.7 Development Libraries (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
93. Python 3.13.7 Documentation (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
94. Python 3.13.7 Executables (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
95. Python 3.13.7 pip Bootstrap (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
96. Python 3.13.7 Standard Library (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
97. Python 3.13.7 Tcl/Tk Support (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
98. Python 3.13.7 Test Suite (64-bit) | Versão: 3.13.7150.0 | Editor: Python Software Foundation
99. Python Launcher | Versão: 3.13.7150.0 | Editor: Python Software Foundation
100. Realtek High Definition Audio Driver | Versão: 6.0.1.7841 | Editor: Realtek Semiconductor Corp.
101. Serv. de Compilador Transact-SQL do Microsoft SQL Server 2014 | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
102. Smart Connect | Versão: 8.0.0.114.001 | Editor: (C) Motorola
103. Sped Fiscal ICMS IPI 5.0.3 | Versão: 5.0.3 | Editor: Receita Federal do Brasil

104. SQL Server 2014 Client Tools | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
105. SQL Server 2014 Common Files | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
106. SQL Server 2014 Management Studio | Versão: 12.0.2000.8 | Editor: Microsoft Corporation
107. SQL Server 2022 Batch Parser | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
108. SQL Server 2022 Common Files | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
109. SQL Server 2022 Connection Info | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
110. SQL Server 2022 Database Engine Services | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
111. SQL Server 2022 Database Engine Shared | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
112. SQL Server 2022 DMF | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
113. SQL Server 2022 Shared Management Objects | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
114. SQL Server 2022 Shared Management Objects Extensions | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
115. SQL Server 2022 SQL Diagnostics | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
116. SQL Server 2022 XEvent | Versão: 16.0.1000.6 | Editor: Microsoft Corporation
117. SQL Server Management Studio 21 | Versão: 21.5.14 | Editor: Microsoft Corporation
118. TAP-Windows 9.21.2 | Versão: 9.21.2 | Editor: N/A
119. TortoiseSVN 1.14.9.29743 (64 bit) | Versão: 1.14.29743 | Editor: TortoiseSVN
120. Visual Studio 2010 Prerequisites - English | Versão: 10.0.30319 | Editor: Microsoft Corporation
121. vs_communitymsires | Versão: 17.14.36015 | Editor: Microsoft Corporation
122. vs_communitysharedmsi | Versão: 17.14.36025 | Editor: Microsoft Corporation
123. vs_communityx64msi | Versão: 17.14.36025 | Editor: Microsoft Corporation
124. vs_CoreEditorFonts | Versão: 17.7.40001 | Editor: Microsoft Corporation
125. vs_filehandler_amd64 | Versão: 17.14.36024 | Editor: Microsoft Corporation
126. vs_filehandler_x86 | Versão: 17.14.36024 | Editor: Microsoft Corporation
127. vs_FileTracker_Singleton | Versão: 17.14.36015 | Editor: Microsoft Corporation
128. vs_githubprotocolhandlermsi | Versão: 17.14.36015 | Editor: Microsoft Corporation
129. vs_minshellinteropsharedmsi | Versão: 17.14.36015 | Editor: Microsoft Corporation
130. vs_minshellinteropx64msi | Versão: 17.14.36015 | Editor: Microsoft Corporation
131. vs_minshellmsires | Versão: 17.14.36015 | Editor: Microsoft Corporation
132. vs_minshellsharedmsi | Versão: 17.14.36024 | Editor: Microsoft Corporation
133. vs_minshellx64msi | Versão: 17.14.36301 | Editor: Microsoft Corporation
134. vs_ssmsprotocolselectormsi | Versão: 17.14.36015 | Editor: Microsoft Corporation
135. vs_ssmsprotocolselectormsires | Versão: 17.14.36015 | Editor: Microsoft Corporation
136. vs_vswebprotocolselectormsi | Versão: 17.14.36323 | Editor: Microsoft Corporation
137. vs_vswebprotocolselectormsires | Versão: 17.14.36323 | Editor: Microsoft Corporation
138. Windows Subsystem for Linux | Versão: 2.5.9.0 | Editor: Microsoft Corporation
139. WinRAR 7.13 (64-bit) | Versão: 7.13.0 | Editor: win.rar GmbH

INFORMAÇÕES DE REDE

Adaptador 1: Realtek PCIe GBE Family Controller

IP: 192.168.0.159 | Gateway: 192.168.0.1 | DNS: 192.168.0.3, 192.168.0.4 | MAC: 2C:FD:A1:C8:77:61

SEGURANÇA DO SISTEMA

Firewall:

Perfil: Domain | Status: Desativado
Perfil: Private | Status: Desativado
Perfil: Public | Status: Desativado

Windows Update:

3 atualizações pendentes

LOG DE DEBUG (CSINFO)

--- CSInfo debug entry ---
TIMESTAMP_UTC: 2025-10-02T20:57:59.415790Z
COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $adapters = Get-WmiObject Win32_NetworkAdapterConfiguration | Where-Object { $_.IPEnabled -eq $true }\n $out = @()\n foreach ($a in $adapters) {\n $ip = $a.IPAddress[0]\n $gw = $a.DefaultIPGateway[0]\n $dns = $a.DNSServerSearchOrder -join '', ''\n $mac = $a.MACAddress\n $desc = $a.Description\n $out += [PSCustomObject]@{\n IP = $ip; Gateway = $gw; DNS = $dns; MAC = $mac; Descricao = $desc\n }\n }\n $out | ConvertTo-Json -Compress\n ']
COMPUTER: None
TIMEOUT: 20
DURATION_SECONDS: 9.225192785263062
RETURN_CODE: 0
OUTPUT:
{"IP":"192.168.0.159","Gateway":"192.168.0.1","DNS":"192.168.0.3,
192.168.0.4","MAC":"2C:FD:A1:C8:77:61","Descricao":"Realtek PCIe GBE Family Controller"}
--- CSInfo debug entry ---
TIMESTAMP_UTC: 2025-10-02T20:58:08.858396Z
COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n try {\n $profiles = Get-NetFirewallProfile\n $out = @()\n foreach ($p in $profiles) {\n $out += [PSCustomObject]@{\n Perfil = $p.Name; Ativado = $p.Enabled\n }\n }\n $out | ConvertTo-Json -Compress\n } catch { "NÃO OBTIDO" }\n ']
COMPUTER: None
TIMEOUT: 20
DURATION_SECONDS: 9.343075275421143
RETURN_CODE: 0
OUTPUT:
[{"Perfil":"Domain","Ativado":0},{"Perfil":"Private","Ativado":0},{"Perfil":"Public","Ativado":0}]
--- CSInfo debug entry ---
TIMESTAMP_UTC: 2025-10-02T20:58:28.327733Z
COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n try {\n $session = New-Object -ComObject Microsoft.Update.Session\n $searcher = $session.CreateUpdateSearcher()\n $result = $searcher.Search("IsInstalled=0")\n $count = $result.Updates.Count\n if ($count -eq 0) { "Nenhuma atualização pendente" }\n else { "$count atualizações pendentes" }\n } catch { "NÃO OBTIDO" }\n ']
COMPUTER: None
TIMEOUT: 60
DURATION_SECONDS: 19.459089756011963
RETURN_CODE: 0
OUTPUT:
3 atualizações pendentes
--- CSInfo debug entry ---
TIMESTAMP_UTC: 2025-10-02T20:58:29.024704Z
COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n Get-Process | Select-Object -First 10 Name,Id,CPU | ConvertTo-Json -Compress\n ']
COMPUTER: None
TIMEOUT: 20
DURATION_SECONDS: 0.6962659358978271
RETURN_CODE: 0
OUTPUT:
[{"Name":"AggregatorHost","Id":8992,"CPU":null},{"Name":"AppActions","Id":3036,"CPU":2.40625},{"Name":"ApplicationFrameHost","Id":15324,"CPU":3.671875},{"Name":"audiodg","Id":14844,"CPU":29},{"Name":"avp","Id":4456,"CPU":null},{"Name":"avpui","Id":11728,"CPU":202.890625},{"Name":"backgroundTaskHost","Id":6192,"CPU":0.25},{"Name":"backgroundTaskHost","Id":16692,"CPU":0.453125},{"Name":"backgroundTaskHost","Id":16788,"CPU":2.328125},{"Name":"backgroundTaskHost","Id":17136,"CPU":0.375}]
--- CSInfo debug entry ---
TIMESTAMP_UTC: 2025-10-02T20:58:30.197793Z
COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $names = @("wuauserv", "WinDefend", "BITS", "Spooler", "LanmanServer", "LanmanWorkstation")\n $out = @()\n foreach ($n in $names) {\n $s = Get-Service -Name $n -ErrorAction SilentlyContinue\n if ($s) {\n $out += [PSCustomObject]@{Nome=$s.Name; Status=$s.Status}\n }\n }\n $out | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 1.1725218296051025

RETURN_CODE: 0

OUTPUT:

[{"Nome":"wuauserv","Status":4},{"Nome":"WinDefend","Status":1},{"Nome":"BITS","Status":1},{"Nome":"Spooler","Status":4},{"Nome":"LanmanServer","Status":4},{"Nome":"LanmanWorkstation","Status":4}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:30.715281Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n try {\n $namespace = "root\\\\SecurityCenter2"\n $firewallProducts = Get-WmiObject -Namespace $namespace -Class FirewallProduct -ErrorAction SilentlyContinue\n $out = @()\n foreach ($fw in $firewallProducts) {\n $out += $fw.displayName\n }\n if ($out.Count -gt 0) {\n $out -join ", "\n } else {\n "Windows Firewall (padrão)"\n }\n } catch {\n "NÃO OBTIDO"\n }\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.5165574550628662

RETURN_CODE: 0

OUTPUT:

Windows Firewall (padrão)

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:31.417936Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; Get-CimInstance Win32_Battery | ConvertTo-Json -Compress']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.6319191455841064

RETURN_CODE: 0

OUTPUT:

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:31.829509Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; (Get-CimInstance Win32_SystemEnclosure | Select-Object -ExpandProperty ChassisTypes | ConvertTo-Json -Compress)']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.4110293388366699

RETURN_CODE: 0

OUTPUT:

3

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:32.271099Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; (Get-CimInstance Win32_OperatingSystem | Select-Object -Property Caption,Version,OSArchitecture | ConvertTo-Json -Compress)']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.44107556343078613

RETURN_CODE: 0

OUTPUT:

{"Caption":"Microsoft Windows 11 Pro","Version":"10.0.26100","OSArchitecture":"64 bits"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:33.485470Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n try {\n # Usar slmgr diretamente\n $slmgrResult = & cscript //nologo C:\\Windows\\System32\\slmgr.vbs /xpr\n $output = $slmgrResult -join "`n"\n \n # Verificar diferentes padrões de texto para ativação\n if ($output -match "ativada permanentemente|permanently activated|permanently|permanente") {\n "Ativado"\n } elseif ($output -match "grace period|período de carência|carência") {\n "Período de carência"\n } elseif ($output -match

"notification|notificação") {\n "Período de notificação"\n } elseif ($output -match "not activated|não ativado|não está ativado") {\n "Não ativado"\n } else {\n # Se não conseguir interpretar, retornar a saída original limpa\n ($output -replace "`r", "" -replace "`n", " ").Trim()\n }\n } catch {\n try {\n # Método alternativo usando Get-WmiObject\n $licenses = Get-WmiObject -Class SoftwareLicensingProduct | Where-Object {\n $_.Name -like "*Windows*" -and $_.PartialProductKey -ne $null\n } | Select-Object -First 1\n \n if ($licenses) {\n switch ($licenses.LicenseStatus) {\n 1 { "Ativado" }\n 0 { "Não licenciado" }\n 2 { "Período de carência" }\n 3 { "OOT (Out of Tolerance)" }\n 4 { "OOB (Out of Box)" }\n 5 { "Notificação" }\n 6 { "Carência estendida" }\n default { "Status desconhecido: $($licenses.LicenseStatus)" }\n }\n } else {\n "Nenhuma licença encontrada"\n }\n } catch {\n "Erro na verificação"\n }\n }\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 1.2135586738586426

RETURN_CODE: 0

OUTPUT:

Ativado

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:47.333912Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n try {\n # Verificar Office através do registro\n $officeApps = @("Word", "Excel", "PowerPoint", "Outlook")\n $activated = $false\n \n foreach ($app in $officeApps) {\n try {\n $comObject = New-Object -ComObject "$app.Application"\n if ($comObject) {\n $activated = $true\n $comObject.Quit()\n break\n }\n } catch {}\n }\n \n if ($activated) {\n "Ativado"\n } else {\n # Método alternativo: verificar chaves de ativação no registro\n $regPaths = @(\n "HKLM:\\SOFTWARE\\Microsoft\\Office*\\Registration",\n "HKLM:\\SOFTWARE\\WOW6432Node\\Microsoft\\Office*\\Registration"\n)\n \n $foundActivation = $false\n foreach ($path in $regPaths) {\n try {\n $items = Get-ChildItem $path -ErrorAction SilentlyContinue\n if ($items) {\n $foundActivation = $true\n break\n }\n } catch {}\n }\n \n if ($foundActivation) {\n "Possivelmente ativado"\n } else {\n "Não detectado"\n }\n }\n } catch {\n "NÃO OBTIDO"\n }\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 13.829737424850464

RETURN_CODE: 0

OUTPUT:

Ativado

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:49.302271Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $sqlInstances = @()\n \n # Buscar serviços do SQL Server\n try {\n $services = Get-WmiObject -Class Win32_Service -Filter "Name LIKE \'%SQL%\'" -ErrorAction SilentlyContinue\n $sqlServices = $services | Where-Object { $_.Name -match "MSSQL\\$" -or $_.Name -eq "MSSQLSERVER" }\n \n foreach ($service in $sqlServices) {\n $instanceName = if ($service.Name -eq "MSSQLSERVER") { "Default" } else { $service.Name -replace "MSSQL\\$", "" }\n $status = $service.State\n \n # Tentar obter versão do registro\n $version = ""\n try {\n if ($instanceName -eq "Default") {\n $regPath = "HKLM:\\SOFTWARE\\Microsoft\\Microsoft SQL Server\\MSSQL*\\MSSQLServer\\CurrentVersion"\n } else {\n $regPath = "HKLM:\\SOFTWARE\\Microsoft\\Microsoft SQL Server\\MSSQL*\\MSSQLServer\\CurrentVersion"\n }\n \n $versionKeys = Get-ChildItem "HKLM:\\SOFTWARE\\Microsoft\\Microsoft SQL Server\\" -ErrorAction SilentlyContinue | \n Where-Object { $_.Name -match "MSSQL\\d+\\." }\n \n foreach ($key in $versionKeys) {\n $currentVersionPath = Join-Path $key.PSPath "MSSQLServer\\CurrentVersion"\n if (Test-Path $currentVersionPath) {\n $versionReg = Get-ItemProperty $currentVersionPath -ErrorAction SilentlyContinue\n if ($versionReg.CurrentVersion) {\n $version = $versionReg.CurrentVersion\n break\n }\n }\n }\n } catch {}\n \n $sqlInstances += [PSCustomObject]@{\n Instance = $instanceName\n Status = $status\n Version = $version\n }\n }\n } catch {}\n \n # Se não encontrou serviços, tentar pelo registro\n if ($sqlInstances.Count -eq 0) {\n try {\n $regKeys = Get-ChildItem "HKLM:\\SOFTWARE\\Microsoft\\Microsoft SQL Server\\" -ErrorAction SilentlyContinue | \n Where-Object { $_.Name -match "MSSQL\\d+\\." }\n \n foreach ($key in $regKeys) {\n $setupPath = Join-Path $key.PSPath "Setup"\n if (Test-Path $setupPath) {\n $setup = Get-ItemProperty $setupPath -ErrorAction SilentlyContinue\n if ($setup.SqlProgramDir) {\n $sqlInstances += [PSCustomObject]@{\n Instance = $setup.Edition\n Status = "Unknown"\n Version = $setup.Version\n }\n }\n }\n }\n } catch {}\n }\n \n $sqlInstances | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 1.967275857925415

RETURN_CODE: 0

OUTPUT:

{"Instance":"Default","Status":"Stopped","Version":"16.0.1000.6"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:49.876309Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $antivirusList = @()\n \n # Método 1: Windows Security Center (funciona no Windows 10/11)\n try {\n $namespace = "root\\SecurityCenter2"\n $antivirusProducts = Get-WmiObject -Namespace $namespace -Class AntiVirusProduct -ErrorAction SilentlyContinue\n \n foreach ($av in $antivirusProducts) {\n $name = $av.displayName\n $state = $av.productState\n \n # Decodificar o estado do produto\n $hex = [Convert]::ToString($state, 16).PadLeft(6, "0")\n $s2 = $hex.Substring(2,2) \n \n $enabled = "Desconhecido"\n \n # Verificar se está ativado (bit 4 e 5)\n if ([Convert]::ToInt32($s2, 16) -band 0x10) {\n $enabled = "Ativado"\n } elseif ([Convert]::ToInt32($s2, 16) -band 0x00) {\n $enabled = "Desativado"\n }\n \n $antivirusList += [PSCustomObject]@{\n Name = $name\n Enabled = $enabled\n }\n }\n } catch {}\n \n # Método 2: Verificar pelo registro (programas instalados)\n if ($antivirusList.Count -eq 0) {\n try {\n $uninstallKeys = @(\n "HKLM:\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall*",\n "HKLM:\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Uninstall*"\n)\n \n $antivirusNames = @(\n "Avast", "AVG", "Kaspersky", "Norton", "McAfee", "Bitdefender", \n "ESET", "Trend Micro", "F-Secure", "Panda", "Sophos", "Malwarebytes",\n "Windows Defender", "Microsoft Defender", "Symantec", "Avira",\n "Quick Heal", "Comodo", "360 Total Security", "Baidu Antivirus"\n)\n \n foreach ($key in $uninstallKeys) {\n $programs = Get-ItemProperty $key -ErrorAction SilentlyContinue\n foreach ($program in $programs) {\n $displayName = $program.DisplayName\n if ($displayName) {\n foreach ($avName in $antivirusNames) {\n if ($displayName -like "*$avName*") {\n $antivirusList += [PSCustomObject]@{\n Name = $displayName\n Enabled = "Instalado"\n }\n break\n }\n }\n }\n }\n } catch {}\n }\n \n # Método 3: Verificar Windows Defender especificamente\n if ($antivirusList.Count -eq 0) {\n try {\n $defenderStatus = Get-MpComputerStatus -ErrorAction SilentlyContinue\n if ($defenderStatus) {\n $enabled = if ($defenderStatus.RealTimeProtectionEnabled) { "Ativado" } else { "Desativado" }\n \n $antivirusList += [PSCustomObject]@{\n Name = "Windows Defender"\n Enabled = $enabled\n }\n }\n } catch {}\n }\n \n # Se nada foi encontrado, verificar se há pelo menos o Windows Defender básico\n if ($antivirusList.Count -eq 0) {\n try {\n $defenderService = Get-Service -Name "WinDefend" -ErrorAction SilentlyContinue\n if ($defenderService) {\n $status = if ($defenderService.Status -eq "Running") { "Em execução" } else { $defenderService.Status }\n $antivirusList += [PSCustomObject]@{\n Name = "Windows Defender (Serviço)"\n Enabled = $status\n }\n }\n } catch {}\n }\n \n # Remover duplicatas baseado no nome\n $uniqueList = @()\n $seenNames = @()\n foreach ($av in $antivirusList) {\n $cleanName = $av.Name -replace \'\\s*\\d+.*$\', \'\' -replace \'\\s*\\(.*\\)\', \'\' -replace \'\\s+\', \' \'\n $cleanName = $cleanName.Trim()\n \n if ($seenNames -notcontains $cleanName) {\n $seenNames += $cleanName\n $uniqueList += [PSCustomObject]@{\n Name = $cleanName\n Enabled = $av.Enabled\n }\n }\n }\n \n $uniqueList | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.5734639167785645

RETURN_CODE: 0

OUTPUT:

[{"Name":"Kaspersky","Enabled":"Ativado"},{"Name":"Windows Defender","Enabled":"Desconhecido"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:50.635860Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $computer = Get-CimInstance -ClassName Win32_ComputerSystem\n if ($computer.PartOfDomain) {\n "Domínio: $($computer.Domain)"\n } else {\n "Workgroup: $($computer.Workgroup)"\n }\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.7586431503295898

RETURN_CODE: 0

OUTPUT:

Domínio: ceosoftwaread.net

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:51.096179Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; (Get-CimInstance Win32_ComputerSystem | Select-Object -Property TotalPhysicalMemory | ConvertTo-Json -Compress)']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.4596824645996094

RETURN_CODE: 0

OUTPUT:

{"TotalPhysicalMemory":8451526656}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:51.641646Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $memoryModules = @()\n \n try {\n $modules = Get-CimInstance Win32_PhysicalMemory -ErrorAction SilentlyContinue\n \n foreach ($module in $modules) {\n $manufacturer = $module.Manufacturer\n $partNumber =

$module.PartNumber\n $serialNumber = $module.SerialNumber\n $capacity = $module.Capacity\n $speed = $module.Speed\n $memoryType = $module.MemoryType\n $formFactor = $module.FormFactor\n $deviceLocator = $module.DeviceLocator\n $bankLabel = $module.BankLabel\n \n # Converter capacidade para GB\n $capacityGB = if ($capacity) { [math]::Round($capacity / 1GB, 2).ToString() + " GB" } else { "N/A" }\n \n # Converter velocidade\n $speedMHz = if ($speed) { $speed.ToString() + " MHz" } else { "N/A" }\n \n # Converter tipo de memória\n $memTypeText = switch ($memoryType) {\n 20 { "DDR" }\n 21 { "DDR2" }\n 22 { "DDR2 FB-DIMM" }\n 24 { "DDR3" }\n 26 { "DDR4" }\n 34 { "DDR5" }\n default { \n # Tentar deduzir pelo speed (heurística) se $memoryType for 0, null ou desconhecido\n if ($speed -and $speed -ge 2133) { "DDR4" }\n elseif ($speed -and $speed -ge 1066) { "DDR3" } \n elseif ($speed -and $speed -ge 533) { "DDR2" }\n elseif ($speed -and $speed -ge 200) { "DDR" }\n else { "Desconhecido" }\n }\n }\n \n # Converter fator de forma\n $formFactorText = switch ($formFactor) {\n 8 { "DIMM" }\n 12 { "SO-DIMM" }\n 13 { "Micro-DIMM" }\n default { "Form $formFactor" }\n }\n \n $memoryModules += [PSCustomObject]@{\n Manufacturer = if ($manufacturer) { $manufacturer.Trim() } else { "N/A" }\n PartNumber = if ($partNumber) { $partNumber.Trim() } else { "N/A" }\n SerialNumber = if ($serialNumber) { $serialNumber.Trim() } else { "N/A" }\n Capacity = $capacityGB\n Speed = $speedMHz\n MemoryType = $memTypeText\n FormFactor = $formFactorText\n Location = if ($deviceLocator) { $deviceLocator } else { $bankLabel }\n }\n }\n } catch {}\n \n $memoryModules | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.543830394744873

RETURN_CODE: 0

OUTPUT:

{"Manufacturer":"8945","PartNumber":"MD401GNSE-CB3M2 00","SerialNumber":"26171602","Capacity":"8 GB","Speed":"2400 MHz","MemoryType":"DDR4","FormFactor":"DIMM","Location":"DIMM_A1"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:53.861224Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $processorInfo = @()\n \n try {\n $processors = Get-CimInstance Win32_Processor -ErrorAction SilentlyContinue\n \n foreach ($cpu in $processors) {\n $name = $cpu.Name\n $manufacturer = $cpu.Manufacturer\n $architecture = $cpu.Architecture\n $cores = $cpu.NumberOfCores\n $logicalProcessors = $cpu.NumberOfLogicalProcessors\n $maxClockSpeed = $cpu.MaxClockSpeed\n $currentClockSpeed = $cpu.CurrentClockSpeed\n $l2CacheSize = $cpu.L2CacheSize\n $l3CacheSize = $cpu.L3CacheSize\n $socket = $cpu.SocketDesignation\n \n # Converter arquitetura para texto\n $archText = switch ($architecture) {\n 0 { "x86" }\n 1 { "MIPS" }\n 2 { "Alpha" }\n 3 { "PowerPC" }\n 6 { "Intel Itanium" }\n 9 { "x64" }\n default { "Desconhecida ($architecture)" }\n }\n \n # Converter velocidades de MHz para GHz\n $maxSpeed = if ($maxClockSpeed) { [math]::Round($maxClockSpeed / 1000, 2).ToString() + " GHz" } else { "N/A" }\n $currentSpeed = if ($currentClockSpeed) { [math]::Round($currentClockSpeed / 1000, 2).ToString() + " GHz" } else { "N/A" }\n \n # Converter cache para KB/MB\n $l2Cache = if ($l2CacheSize) { \n if ($l2CacheSize -ge 1024) { [math]::Round($l2CacheSize / 1024, 2).ToString() + " MB" }\n else { $l2CacheSize.ToString() + " KB" }\n } else { "N/A" }\n \n $l3Cache = if ($l3CacheSize) { \n if ($l3CacheSize -ge 1024) { [math]::Round($l3CacheSize / 1024, 2).ToString() + " MB" }\n else { $l3CacheSize.ToString() + " KB" }\n } else { "N/A" }\n \n $processorInfo += [PSCustomObject]@{\n Name = $name\n Manufacturer = $manufacturer\n Architecture = $archText\n Cores = $cores\n LogicalProcessors = $logicalProcessors\n MaxSpeed = $maxSpeed\n CurrentSpeed = $currentSpeed\n L2Cache = $l2Cache\n L3Cache = $l3Cache\n Socket = $socket\n }\n }\n } catch {}\n \n $processorInfo | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 2.2190401554107666

RETURN_CODE: 0

OUTPUT:

{"Name":"Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz","Manufacturer":"GenuineIntel","Architecture":"x64","Cores":4,"LogicalProcessors":8,"MaxSpeed":"3,6 GHz","CurrentSpeed":"3,6 GHz","L2Cache":"1 MB","L3Cache":"8 MB","Socket":"LGA1151"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:55.695674Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $disks = @()\n try {\n $physicalDisks = Get-CimInstance Win32_DiskDrive -ErrorAction SilentlyContinue\n foreach ($disk in $physicalDisks) {\n $model = $disk.Model\n $size = [math]::Round($disk.Size / 1GB, 2)\n $mediaType = $disk.MediaType\n $interface = $disk.InterfaceType\n \n # Tentar determinar se é SSD ou HDD\n $diskType = "HDD"\n if ($mediaType -like "*SSD*" -or $model -like "*SSD*" -or $model -like "*Solid State*") {\n $diskType = "SSD"\n } elseif ($mediaType -like "*Fixed*" -or $interface -eq "SCSI") {\n # Usar SMART para detectar SSD (método alternativo)\n try {\n $smartData = Get-WmiObject -Namespace root\\wmi -Class MSStorageDriver_FailurePredictData -ErrorAction SilentlyContinue | Where-Object {$_.InstanceName -like "*$($disk.PNPDeviceID)*"}\n if ($smartData) {\n $diskType = "SSD"\n }\n } catch {}\n }\n \n # Obter informações de espaço das partições associadas ao disco físico\n $totalUsed = 0\n $totalFree = 0\n $partitions = ""\n try {\n # Método mais direto: obter partições pelo índice do disco\n $associatedPartitions = Get-CimInstance -Query "ASSOCIATORS OF {Win32_DiskDrive.DeviceID=\'$($disk.DeviceID)\'} WHERE AssocClass=Win32_DiskDriveToDiskPartition" -ErrorAction SilentlyContinue\n \n foreach ($partition in $associatedPartitions) {\n $logicalDisks = Get-CimInstance -Query "ASSOCIATORS OF {Win32_DiskPartition.DeviceID=\'$($partition.DeviceID)\'} WHERE AssocClass=Win32_LogicalDiskToDiskPartition" -ErrorAction SilentlyContinue\n \n foreach ($logicalDisk in $logicalDisks) {\n if ($logicalDisk.Size) {\n $driveSize =

[math]::Round($logicalDisk.Size / 1GB, 2)\n $driveFree = [math]::Round($logicalDisk.FreeSpace / 1GB, 2)\n $driveUsed = $driveSize - $driveFree\n $totalUsed += $driveUsed\n $totalFree += $driveFree\n if ($partitions) { $partitions += ", " }\n $partitions += "$($logicalDisk.DeviceID) ($driveFree GB livre de $driveSize GB)"\n }\n }\n \n # Se o método acima não funcionou, tentar método alternativo\n if ($totalFree -eq 0 -and $totalUsed -eq 0) {\n $diskPartitions = Get-CimInstance Win32_DiskPartition -ErrorAction SilentlyContinue | Where-Object { $_.DiskIndex -eq $disk.Index }\n foreach ($partition in $diskPartitions) {\n $logicalDisks = Get-CimInstance Win32_LogicalDiskToDiskPartition -ErrorAction SilentlyContinue | Where-Object { $_.Antecedent -like '*$($partition.DeviceID)*' }\n foreach ($logicalDiskRel in $logicalDisks) {\n $deviceId = ($logicalDiskRel.Dependent -split \'"\')[1]\n $drive = Get-CimInstance Win32_LogicalDisk -ErrorAction SilentlyContinue | Where-Object { $_.DeviceID -eq $deviceId }\n if ($drive -and $drive.Size) {\n $driveSize = [math]::Round($drive.Size / 1GB, 2)\n $driveFree = [math]::Round($drive.FreeSpace / 1GB, 2)\n $driveUsed = $driveSize - $driveFree\n $totalUsed += $driveUsed\n $totalFree += $driveFree\n if ($partitions) { $partitions += ", " }\n $partitions += "$($drive.DeviceID) ($driveFree GB livre de $driveSize GB)"\n }\n }\n }\n } catch {}\n \n $espacoLivre = if ($totalFree -gt 0) { "$totalFree GB" } else { "N/A" }\n $espacoUsado = if ($totalUsed -gt 0) { "$totalUsed GB" } else { "N/A" }\n $particoesInfo = if ($partitions) { $partitions } else { "N/A" }\n \n $disks += [PSCustomObject]@{\n Modelo = $model\n Tamanho = "$size GB"\n EspacoUsado = $espacoUsado\n EspacoLivre = $espacoLivre\n Particoes = $particoesInfo\n Tipo = $diskType\n Interface = $interface\n }\n }\n } catch {}\n $disks | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 1.8338797092437744

RETURN_CODE: 0

OUTPUT:

[{"Modelo":"ST1000DM010-2EP102","Tamanho":"931.51 GB","EspacoUsado":"N/A","EspacoLivre":"N/A","Particoes":"N/A","Tipo":"HDD","Interface":"IDE"},{"Modelo":"KINGSTON SA400S37960G","Tamanho":"894.25 GB","EspacoUsado":"N/A","EspacoLivre":"N/A","Particoes":"N/A","Tipo":"HDD","Interface":"IDE"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:56.159567Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $drives = @()\n \n try {\n $logicalDrives = Get-CimInstance Win32_LogicalDisk -Filter "DriveType=3" -ErrorAction SilentlyContinue\n foreach ($drive in $logicalDrives) {\n $deviceId = $drive.DeviceID\n $size = if ($drive.Size) { [math]::Round($drive.Size / 1GB, 2) } else { 0 }\n $freeSpace = if ($drive.FreeSpace) { [math]::Round($drive.FreeSpace / 1GB, 2) } else { 0 }\n $usedSpace = $size - $freeSpace\n $fileSystem = $drive.FileSystem\n $volumeName = $drive.VolumeName\n \n $drives += [PSCustomObject]@{\n Drive = $deviceId\n Size = $size\n Used = $usedSpace\n Free = $freeSpace\n FileSystem = $fileSystem\n Label = if ($volumeName) { $volumeName } else { "Sem rótulo" }\n }\n }\n \n # Ordenar por letra da unidade\n $sortedDrives = $drives | Sort-Object Drive\n \n } catch {\n $sortedDrives = @()\n }\n \n $sortedDrives | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.4631989002227783

RETURN_CODE: 0

OUTPUT:

[{"Drive":"C:","Size":893.51,"Used":558.26,"Free":335.25,"FileSystem":"NTFS","Label":"Sem rótulo"},{"Drive":"D:","Size":465.37,"Used":318.78999999999996,"Free":146.58,"FileSystem":"NTFS","Label":"Sem rótulo"},{"Drive":"E:","Size":435.29,"Used":135.91000000000003,"Free":299.38,"FileSystem":"NTFS","Label":"Partição 2"},{"Drive":"F:","Size":29.3,"Used":1.7600000000000016,"Free":27.54,"FileSystem":"NTFS","Label":"Swapping"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:56.588254Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', "[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $s = @()\n try {\n $mons = Get-WmiObject -Namespace root\\wmi -Class WmiMonitorID -ErrorAction SilentlyContinue\n foreach ($m in $mons) {\n $arr = $m.SerialNumberID\n $manuf = ($m.ManufacturerName | ForEach-Object {[char]$_}) -join ''\n $model = ($m.UserFriendlyName | ForEach-Object {[char]$_}) -join ''\n $serial = ($arr | ForEach-Object {[char]$_}) -join ''\n $s += [PSCustomObject]@{Fabricante=$manuf; Modelo=$model; Serial=$serial}\n }\n } catch {}\n $s | ConvertTo-Json -Compress\n "]

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.42789554595947266

RETURN_CODE: 0

OUTPUT:

[{"Fabricante":"DEL\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000","Modelo":"DELL E1910\u0000\u0000\u0000","Serial":"X234R03V0SYL\u0000\u0000\u0000\u0000"},{"Fabricante":"GSM\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000","Modelo":"LG FULL HD\u0000\u0000\u0000","Serial":"16843009\u0000\u0000\u0000\u0000\u0000\u0000\u0000\u0000"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:57.016892Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $hasKeyboard = $false\n $hasMouse = $false\n \n try {\n $keyboards = Get-WmiObject -Class Win32_Keyboard -ErrorAction SilentlyContinue\n if ($keyboards) { $hasKeyboard = $true }\n \n $mice = Get-WmiObject -Class Win32_PointingDevice -ErrorAction SilentlyContinue\n if ($mice) { $hasMouse = $true }\n } catch {}\n \n [PSCustomObject]@{\n HasKeyboard = $hasKeyboard\n HasMouse = $hasMouse\n } | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.4279167652130127

RETURN_CODE: 0

OUTPUT:

{"HasKeyboard":true,"HasMouse":true}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:57.411771Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; (Get-CimInstance Win32_BaseBoard | Select-Object Manufacturer,Product,SerialNumber | ConvertTo-Json -Compress)']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.3940725326538086

RETURN_CODE: 0

OUTPUT:

{"Manufacturer":"ASUSTeK COMPUTER INC.","Product":"H110M-C/BR","SerialNumber":"180414466200454"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:58.040756Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $networkAdapters = @()\n \n try {\n # Obter adaptadores de rede ativos\n $adapters = Get-CimInstance Win32_NetworkAdapter -Filter "NetConnectionStatus=2" -ErrorAction SilentlyContinue\n \n foreach ($adapter in $adapters) {\n $name = $adapter.Name\n $manufacturer = $adapter.Manufacturer\n $speed = "N/A"\n $macAddress = $adapter.MACAddress\n \n # Tentar obter velocidade\n if ($adapter.Speed) {\n $speedMbps = [math]::Round($adapter.Speed / 1000000, 0)\n $speed = "$speedMbps Mbps"\n }\n \n # Verificar se é adaptador físico (não virtual)\n if ($adapter.PhysicalAdapter -eq $true -or $name -notlike "*Virtual*" -and $name -notlike "*Loopback*" -and $macAddress) {\n $networkAdapters += [PSCustomObject]@{\n Name = $name\n Manufacturer = $manufacturer\n Speed = $speed\n MACAddress = $macAddress\n }\n }\n }\n \n # Se não encontrou pelo método acima, tentar método alternativo\n if ($networkAdapters.Count -eq 0) {\n $netAdapters = Get-NetAdapter -Physical -ErrorAction SilentlyContinue | Where-Object { $_.Status -eq "Up" }\n foreach ($net in $netAdapters) {\n $speed = if ($net.LinkSpeed) { \n $speedValue = $net.LinkSpeed\n if ($speedValue -ge 1000000000) {\n [math]::Round($speedValue / 1000000000, 1).ToString() + " Gbps"\n } else {\n [math]::Round($speedValue / 1000000, 0).ToString() + " Mbps"\n }\n } else { "N/A" }\n \n $networkAdapters += [PSCustomObject]@{\n Name = $net.Name\n Manufacturer = $net.DriverProvider\n Speed = $speed\n MACAddress = $net.MacAddress\n }\n }\n } catch {}\n \n $networkAdapters | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.6285207271575928

RETURN_CODE: 0

OUTPUT:

{"Name":"Realtek PCIe GBE Family Controller","Manufacturer":"Realtek","Speed":"100 Mbps","MACAddress":"2C:FD:A1:C8:77:61"}

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:58.534264Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $videoCards = @()\n \n try {\n $cards = Get-CimInstance Win32_VideoController -ErrorAction SilentlyContinue\n \n foreach ($card in $cards) {\n $name = $card.Name\n $manufacturer = $card.AdapterCompatibility\n $memory = "N/A"\n $driver = $card.DriverVersion\n $type = "Desconhecido"\n \n # Determinar memória de vídeo\n if ($card.AdapterRAM -and $card.AdapterRAM -gt 0) {\n $memoryGB = [math]::Round($card.AdapterRAM / 1GB, 2)\n $memory = "$memoryGB GB"\n }\n \n # Tentar determinar se é onboard ou offboard\n if ($name -like "*Intel*" -and ($name -like "*HD Graphics*" -or $name -like "*UHD Graphics*" -or $name -like "*Iris*")) {\n $type = "Onboard (Integrada)"\n } elseif ($name -like "*AMD*" -and $name -like "*Radeon*" -and ($name -like "*Vega*" -or $name -like "*APU*")) {\n $type = "Onboard (Integrada)"\n } elseif ($name -like "*NVIDIA*" -or $name -like "*AMD Radeon RX*" -or $name -like "*GeForce*" -or $name -like "*Quadro*") {\n $type = "Offboard (Dedicada)"\n } elseif ($card.PNPDeviceID -like "*PCI\\VEN_*") {\n $type = "Offboard (Dedicada)"\n } else {\n $type = "Onboard (Integrada)"\n }\n \n $videoCards += [PSCustomObject]@{\n Name = $name\n Manufacturer = $manufacturer\n Memory = $memory\n Driver = $driver\n Type = $type\n }\n }\n } catch {}\n \n $videoCards | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.4929046630859375

RETURN_CODE: 0

OUTPUT:

[{"Name":"BB Capture Driver","Manufacturer":"Blueberry Consultants","Memory":"N/A","Driver":"3.40.0.0","Type":"Onboard (Integrada)"},{"Name":"NVIDIA GeForce GT 220 ","Manufacturer":"NVIDIA","Memory":"1 GB","Driver":"21.21.13.4201","Type":"Offboard (Dedicada)"},{"Name":"Intel(R) HD Graphics 630","Manufacturer":"Intel Corporation","Memory":"1 GB","Driver":"26.20.100.7528","Type":"Onboard (Integrada)"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:58:59.475252Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $o = @()\n try {\n $pr = Get-CimInstance Win32_Printer -ErrorAction SilentlyContinue\n foreach ($p in $pr) {\n $name = $p.Name\n $pnp = $p.PNPDeviceID\n $serial = ""\n $manuf = $p.Manufacturer\n $model = $p.DriverName\n if ($pnp) {\n try {\n $prop = Get-PnpDeviceProperty -InstanceId $pnp -KeyName \'DEVPKEY_Device_SerialNumber\' -ErrorAction SilentlyContinue\n if ($prop) { $serial = $prop.Data }\n } catch {}\n }\n if (-not $serial) { $serial = $pnp }\n $o += [PSCustomObject]@{Name=$name; Serial=$serial; Fabricante=$manuf; Modelo=$model}\n }\n } catch {}\n $o | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.9404575824737549

RETURN_CODE: 0

OUTPUT:

[{"Name":"OneNote (Desktop)","Serial":null,"Fabricante":null,"Modelo":"Send to Microsoft OneNote 16 Driver"},{"Name":"NPI1F7468 (HP LaserJet Professional M1212nf MFP)","Serial":null,"Fabricante":null,"Modelo":"HP LaserJet Mono PCLmS Class Driver"},{"Name":"Microsoft Print to PDF","Serial":null,"Fabricante":null,"Modelo":"Microsoft Print To PDF"},{"Name":"HP LaserJet 1020","Serial":null,"Fabricante":null,"Modelo":"HP LaserJet 1020"}]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:59:00.102970Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $adminUsers = @()\n \n try {\n # Obter membros do grupo de administradores\n $adminGroup = Get-LocalGroup -Name "Administradores" -ErrorAction SilentlyContinue\n if (-not $adminGroup) {\n $adminGroup = Get-LocalGroup -Name "Administrators" -ErrorAction SilentlyContinue\n }\n \n if ($adminGroup) {\n $members = Get-LocalGroupMember -Group $adminGroup -ErrorAction SilentlyContinue\n foreach ($member in $members) {\n $name = $member.Name\n $objectClass = $member.ObjectClass\n $principalSource = $member.PrincipalSource\n \n # Remover o nome do computador/domínio do nome do usuário\n if ($name -like "**") {\n $name = ($name -split "\\\\")[-1]\n }\n \n $adminUsers += $name\n }\n }\n \n # Ordenar por nome em ordem crescente e remover duplicatas\n $sortedUsers = $adminUsers | Sort-Object | Get-Unique\n \n } catch {\n # Método alternativo usando WMI se o método acima falhar\n try {\n $group = Get-WmiObject -Class Win32_Group -Filter "Name=\'Administrators\' OR Name=\'Administradores\'" -ErrorAction SilentlyContinue | Select-Object -First 1\n if ($group) {\n $members = Get-WmiObject -Class Win32_GroupUser -ErrorAction SilentlyContinue | Where-Object { $_.GroupComponent -like "*$($group.Name)*" }\n foreach ($member in $members) {\n $userPath = $member.PartComponent\n if ($userPath -match \'Name="([^"]+)"\') {\n $userName = $matches[1]\n $sortedUsers += $userName\n }\n }\n $sortedUsers = $sortedUsers | Sort-Object | Get-Unique\n }\n } catch {\n $sortedUsers = @()\n }\n }\n \n $sortedUsers | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.6272013187408447

RETURN_CODE: 0

OUTPUT:

["CEOSOFT-059\\Administrador","CEOSOFT-059\\CEOSOFTWARE","CEOSOFTWAREAD\\Admins. do domínio","CEOSOFTWAREAD\\alex","CEOSOFTWAREAD\\Angelo.Gabriel","CEOSOFTWAREAD\\geovani","CEOSOFTWAREAD\\Marco.Aurelio"]

--- CSInfo debug entry ---

TIMESTAMP_UTC: 2025-10-02T20:59:00.952173Z

COMMAND: ['powershell', '-NoProfile', '-NonInteractive', '-ExecutionPolicy', 'Bypass', '-Command', '[Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \n $softwareList = @()\n \n try {\n $uninstallKeys = @(\n "HKLM:\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall*",\n "HKLM:\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Uninstall*"\n)\n \n foreach ($key in $uninstallKeys) {\n $programs = Get-ItemProperty $key -ErrorAction SilentlyContinue\n foreach ($program in $programs) {\n $displayName = $program.DisplayName\n $version = $program.DisplayVersion\n $publisher = $program.Publisher\n \n if ($displayName -and $displayName.Trim() -ne "") {\n # Filtrar alguns itens desnecessários\n if ($displayName -notlike "*Update*" -and \n $displayName -notlike "*Hotfix*" -and\n $displayName -notlike "*Security Update*" -and\n $displayName -notlike "KB*" -and\n $displayName -ne

"Microsoft Visual C++ 2019 X64 Additional Runtime" -and\n $displayName -notlike "*Redistributable*") {\n \n $softwareList += [PSCustomObject]@{\n Name = $displayName\n Version = if ($version) { $version } else { "N/A" }\n Publisher = if ($publisher) { $publisher } else { "N/A" }\n }\n }\n }\n }\n }\n \n # Remover duplicatas e ordenar por nome\n $uniqueSoftware = $softwareList | Group-Object Name | ForEach-Object { $_.Group | Select-Object -First 1 }\n $sortedSoftware = $uniqueSoftware | Sort-Object Name\n \n } catch {\n $sortedSoftware = @()\n }\n \n $sortedSoftware | ConvertTo-Json -Compress\n ']

COMPUTER: None

TIMEOUT: 20

DURATION_SECONDS: 0.8482410907745361

RETURN_CODE: 0

OUTPUT:

[{"Name":"Android Studio","Version":"2025.1","Publisher":"Google LLC"},{"Name":"Arquivo do WinRAR","Version":"N/A","Publisher":"N/A"},{"Name":"Arquivos de Suporte à Instalação do Microsoft SQL Server 2008 ","Version":"10.3.5500.0","Publisher":"Microsoft Corporation"},{"Name":"Atualizações da NVIDIA 10.4.0","Version":"10.4.0","Publisher":"NVIDIA Corporation"},{"Name":"BraZip Central","Version":"25.0.3","Publisher":"BraZip Tecnologia"},{"Name":"CSServiceCenter","Version":"1.0.0","Publisher":"CEOSoftware Sistemas de Informática"},{"Name":"Data Dynamics ActiveReports Professional 2","Version":"2.3.0.1261","Publisher":"Data Dynamics, Ltd."},{"Name":"dbForge SQL Complete, v6.15.5","Version":"6.15.5","Publisher":"Devart"},{"Name":"DiagnosticsHub_CollectionService","Version":"17.14.36412","Publisher":"Microsoft Corporation"},{"Name":"Eclipse Temurin JDK with Hotspot 21.0.8+9 (x64)","Version":"21.0.8.9","Publisher":"Eclipse Adoptium"},{"Name":"Eclipse Temurin JDK with Hotspot 8u462-b08 (x64)","Version":"8.0.462.8","Publisher":"Eclipse Adoptium"},{"Name":"ERP CEOSoftware 8_2","Version":"8.2","Publisher":"CEOSoftware Sistemas de Informática"},{"Name":"Ferramentas de Build do Visual Studio 2019","Version":"16.11.51","Publisher":"Microsoft Corporation"},{"Name":"FileZilla 3.69.3","Version":"3.69.3","Publisher":"Tim Kosse"},{"Name":"Flashback Express 7","Version":"7.7.0.445","Publisher":"Blueberry Software"},{"Name":"GatewayComponents","Version":"16.22.5","Publisher":"Microsoft Corporation"},{"Name":"Git","Version":"2.51.0.2","Publisher":"The Git Development Community"},{"Name":"Google Chrome","Version":"140.0.7339.208","Publisher":"Google LLC"},{"Name":"Gravador VSS da Microsoft para SQL Server 2022","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"HP LaserJet 1020 Series","Version":"N/A","Publisher":"N/A"},{"Name":"Instalação do Microsoft SQL Server 2014 (em inglês)","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Kaspersky","Version":"21.22.7.466","Publisher":"Kaspersky"},{"Name":"Microsoft .NET Framework 4 Multi-Targeting Pack","Version":"4.0.30319","Publisher":"Microsoft Corporation"},{"Name":"Microsoft .NET Host - 8.0.20 (x64)","Version":"64.80.39230","Publisher":"Microsoft Corporation"},{"Name":"Microsoft .NET Host FX Resolver - 8.0.20 (x64)","Version":"64.80.39230","Publisher":"Microsoft Corporation"},{"Name":"Microsoft .NET Runtime - 8.0.20 (x64)","Version":"64.80.39230","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Application Error Reporting","Version":"12.0.6012.5000","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Edge","Version":"140.0.3485.94","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Edge WebView2 Runtime","Version":"140.0.3485.94","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Fabric Integration Runtime","Version":"5.55.9295.2","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Help Viewer 1.1","Version":"1.1.40219","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Help Viewer 1.1 Language Pack - PTB","Version":"1.1.40219","Publisher":"Microsoft Corporation"},{"Name":"Microsoft ODBC Driver 11 for SQL Server","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Microsoft ODBC Driver 17 for SQL Server","Version":"17.10.6.1","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Office 2003 Web Components","Version":"11.0.8173.0","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Office Professional Plus 2019 - pt-br","Version":"16.0.19127.20264","Publisher":"Microsoft Corporation"},{"Name":"Microsoft OLE DB Driver for SQL Server","Version":"18.7.4.0","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Report Viewer 2014 Runtime","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2005 (64-bit)","Version":"N/A","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2005 Analysis Services (64-bit) (SQL2005)","Version":"9.00.1399.06","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2005 Backward compatibility","Version":"8.05.1054","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2005 Tools (64-bit)","Version":"9.00.1399.06","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2008 R2 Management Objects","Version":"10.51.2500.0","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2012 Native Client ","Version":"11.0.2100.60","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2014 (64 bits)","Version":"N/A","Publisher":"N/A"},{"Name":"Microsoft SQL Server 2014 Transact-SQL ScriptDom ","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2022 (64-bit)","Version":"N/A","Publisher":"N/A"},{"Name":"Microsoft SQL Server 2022 RsFx Driver","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server 2022 Setup (English)","Version":"16.0.4212.1","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server Native Client","Version":"9.00.1399.06","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server Setup Support Files (English)","Version":"9.00.1399.06","Publisher":"Microsoft Corporation"},{"Name":"Microsoft SQL Server System CLR Types","Version":"10.51.2500.0","Publisher":"Microsoft Corporation"},{"Name":"Microsoft System CLR Types para SQL Server 2014","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Teams Meeting Add-in for Microsoft Office","Version":"1.24.31301","Publisher":"Microsoft"},{"Name":"Microsoft Visual C++ 2010 x86 Runtime - 10.0.40219","Version":"10.0.40219","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2013 x64 Additional Runtime - 12.0.40664","Version":"12.0.40664","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2013 x64 Minimum Runtime - 12.0.40664","Version":"12.0.40664","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2019

X64 Debug Runtime - 14.29.30157","Version":"14.29.30157","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2019 X86 Debug Runtime - 14.29.30157","Version":"14.29.30157","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2022 X64 Additional Runtime - 14.44.35211","Version":"14.44.35211","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2022 X64 Minimum Runtime - 14.44.35211","Version":"14.44.35211","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2022 X86 Additional Runtime - 14.44.35211","Version":"14.44.35211","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual C++ 2022 X86 Minimum Runtime - 14.44.35211","Version":"14.44.35211","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual Studio 2010 Shell (Isolated) - PTB","Version":"10.0.40219","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual Studio Installer","Version":"3.14.2086.54749","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual Studio Setup Configuration","Version":"3.12.2140.44225","Publisher":"Microsoft Corporation"},{"Name":"Microsoft Visual Studio Setup WMI Provider","Version":"3.12.2140.44225","Publisher":"Microsoft Corporation"},{"Name":"Navegador para SQL Server 2022","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"Node.js","Version":"22.19.0","Publisher":"Node.js Foundation"},{"Name":"Npgsql 3.2.6","Version":"3.2.6-3","Publisher":"EnterpriseDB"},{"Name":"NVIDIA Driver de áudio HD 1.3.30.1","Version":"1.3.30.1","Publisher":"NVIDIA Corporation"},{"Name":"NVIDIA Driver de gráficos 342.01","Version":"342.01","Publisher":"NVIDIA Corporation"},{"Name":"NVIDIA Driver do 3D Vision 342.01","Version":"342.01","Publisher":"NVIDIA Corporation"},{"Name":"NVIDIA Install Application","Version":"2.1002.197.1706","Publisher":"NVIDIA Corporation"},{"Name":"NVIDIA Stereoscopic 3D Driver","Version":"7.17.12.6514","Publisher":"NVIDIA Corporation"},{"Name":"Office 16 Click-to-Run Extensibility Component","Version":"16.0.19127.20154","Publisher":"Microsoft Corporation"},{"Name":"Office 16 Click-to-Run Extensibility Component 64-bit Registration","Version":"16.0.19127.20154","Publisher":"Microsoft Corporation"},{"Name":"Office 16 Click-to-Run Licensing Component","Version":"16.0.19127.20264","Publisher":"Microsoft Corporation"},{"Name":"Office 16 Click-to-Run Localization Component","Version":"16.0.19127.20154","Publisher":"Microsoft Corporation"},{"Name":"On-premises data gateway","Version":"3000.278.5","Publisher":"Microsoft Corporation"},{"Name":"OpenVPN 2.4.6-I602 ","Version":"2.4.6-I602","Publisher":"OpenVPN Technologies, Inc."},{"Name":"Oracle VirtualBox 7.2.2","Version":"7.2.2","Publisher":"Oracle and/or its affiliates"},{"Name":"Painel de controle da NVIDIA 342.01","Version":"342.01","Publisher":"NVIDIA Corporation"},{"Name":"pgAgent_PG17 4.2.3","Version":"4.2.3-1","Publisher":"EnterpriseDB"},{"Name":"PgBouncer 1.24.1","Version":"1.24.1-1","Publisher":"EnterpriseDB"},{"Name":"pgJDBC 42.7.2","Version":"42.7.2-1","Publisher":"EnterpriseDB"},{"Name":"Políticas do Microsoft SQL Server 2014 ","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"PostgreSQL 17 ","Version":"17.6-1","Publisher":"PostgreSQL Global Development Group"},{"Name":"psqlODBC 13.02.0000","Version":"13.02.0000-1","Publisher":"EnterpriseDB"},{"Name":"Python 3.13.7 Add to Path (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Core Interpreter (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Development Libraries (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Documentation (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Executables (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 pip Bootstrap (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Standard Library (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Tcl/Tk Support (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python 3.13.7 Test Suite (64-bit)","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Python Launcher","Version":"3.13.7150.0","Publisher":"Python Software Foundation"},{"Name":"Realtek High Definition Audio Driver","Version":"6.0.1.7841","Publisher":"Realtek Semiconductor Corp."},{"Name":"Serv. de Compilador Transact-SQL do Microsoft SQL Server 2014 ","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"Smart Connect","Version":"8.0.0.114.001","Publisher":"(C) Motorola"},{"Name":"Sped Fiscal ICMS IPI 5.0.3","Version":"5.0.3","Publisher":"Receita Federal do Brasil"},{"Name":"SQL Server 2014 Client Tools","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2014 Common Files","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2014 Management Studio","Version":"12.0.2000.8","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Batch Parser","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Common Files","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Connection Info","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Database Engine Services","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Database Engine Shared","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 DMF","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Shared Management Objects","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 Shared Management Objects Extensions","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 SQL Diagnostics","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server 2022 XEvent","Version":"16.0.1000.6","Publisher":"Microsoft Corporation"},{"Name":"SQL Server Management Studio 21","Version":"21.5.14","Publisher":"Microsoft Corporation"},{"Name":"TAP-Windows 9.21.2","Version":"9.21.2","Publisher":"N/A"},{"Name":"TortoiseSVN 1.14.9.29743 (64 bit)","Version":"1.14.29743","Publisher":"TortoiseSVN"},{"Name":"Visual Studio 2010 Prerequisites - English","Version":"10.0.30319","Publisher":"Microsoft Corporation"},{"Name":"vs_communitymsires","Version":"17.14.36015","Publisher":"Microsoft Corporation"},{"Name":"vs_communitysharedmsi","Version":"17.14.36025","Publisher":"Microsoft Corporation"},{"Name":"vs_communityx64msi","Version":"17.14.36025","Publisher":"Microsoft

Corporation"},{"Name":"vs_CoreEditorFonts","Version":"17.7.40001","Publisher":"Microsoft
Corporation"},{"Name":"vs_filehandler_amd64","Version":"17.14.36024","Publisher":"Microsoft
Corporation"},{"Name":"vs_filehandler_x86","Version":"17.14.36024","Publisher":"Microsoft
Corporation"},{"Name":"vs_FileTracker_Singleton","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_githubprotocolhandlermsi","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_minshellinteropsharedmsi","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_minshellinteropx64msi","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_minshellmsires","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_minshellsharedmsi","Version":"17.14.36024","Publisher":"Microsoft
Corporation"},{"Name":"vs_minshellx64msi","Version":"17.14.36301","Publisher":"Microsoft
Corporation"},{"Name":"vs_ssmsprotocolselectormsi","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_ssmsprotocolselectormsires","Version":"17.14.36015","Publisher":"Microsoft
Corporation"},{"Name":"vs_vswebprotocolselectormsi","Version":"17.14.36323","Publisher":"Microsoft
Corporation"},{"Name":"vs_vswebprotocolselectormsires","Version":"17.14.36323","Publisher":"Microsoft
Corporation"},{"Name":"Windows Subsystem for Linux","Version":"2.5.9.0","Publisher":"Microsoft Corporation"},{"Name":"WinRAR
7.13 (64-bit)","Version":"7.13.0","Publisher":"win.rar GmbH"}]