GenX FX Hosting & Billing Summary

Hosting Details

Hosting Plan	Stellar Plan
Main Domain	genxfx.org
Start Date	19 Sep 2025
Server	server354.web-hostname-hosting.com
Server IP	192.64.117.27
Nameserver 1	dns1.namecheaphosting.com
Nameserver 2	dns2.namecheaphosting.com

Billing Address

MOUYLENG KEA
A6..V9 Chamkasomrong
Phnompenh, Battambang, 02354, KH
Tel: +855.0963376851
Email: keamouyleng369@gmail.com

Deployment Summary

Root Path	D:/GenX_FX
Scripts	start-genx.bat, deploy-genx.bat
Deployment Mode	Manual or CI/CD-triggered
Status	Fully Configured & Running

Pro Tips & Maintenance

- SSL Automation: Use Let's Encrypt (cPanel/Certbot) for auto-renewal
- Monitoring: UptimeRobot, Prometheus + Grafana, Logtail/Papertrail
- Secrets Rotation: Quarterly via GitHub Actions or PowerShell
- Backup: Automate DB + Config backups to S3/Azure Blob w/ versioning
- CI/CD: Hook deploy-genx.bat to GitHub Actions for push-to-main deploy
- Maintenance: Run scripts as Admin, log deployments, update README regularly

□ Appendix: Get-WindowsInfo.ps1 Script

```
# Save this as Get-WindowsInfo.ps1 and run in PowerShell (Run as Administrator recommended)
# Output files
$timestamp = (Get-Date).ToString("yyyyMMdd_HHmmss")
$outFolder = "$env:USERPROFILE\Desktop\WindowsInfo_$timestamp"
New-Item -Path $outFolder -ItemType Directory -Force | Out-Null
$jsonFile = Join-Path $outFolder "windows_info.json"
$textFile = Join-Path $outFolder "windows_info.txt"

function SafeGet($scriptblock) {
   try { & $scriptblock } catch { "ERROR: $($_.Exception.Message)" }
}

# Collect info
$info = [ordered]@{
   CollectedAt = Get-Date
   ComputerName = $env:COMPUTERNAME
   UserName     = $env:USERNAME
   OS           = SafeGet { Get-ComputerInfo -Property "OsName","OsVersion","OsArchitecture","Window
   HostInfo     = SafeGet { Get-CimInstance -ClassName Win32_ComputerSystem | Select-Object Manuf
   BIOS         = SafeGet { Get-CimInstance -ClassName Win32_BIOS | Select-Object SerialNumber,SMBI
   CPU          = SafeGet { Get-CimInstance -ClassName Win32_Processor | Select-Object Name,Manufac
   GPU          = SafeGet { Get-CimInstance -Namespace root\cimv2 -ClassName Win32_VideoController
   Memory       = SafeGet { Get-CimInstance -ClassName Win32_PhysicalMemory | Select-Object Manuf
   Disks        = SafeGet { Get-CimInstance -ClassName Win32_DiskDrive | Select-Object Model,Interface
   Volumes      = SafeGet { Get-Volume | Select-Object DriveLetter,FileSystemLabel,FileSystem,SizeRem
   Network      = SafeGet { Get-NetIPAddress -AddressFamily IPv4 -ErrorAction SilentlyContinue | Where
   NetAdapters  = SafeGet { Get-NetAdapter | Select-Object Name,Status,MacAddress,LinkSpeed }
   InstalledHotfixes = SafeGet { Get-HotFix | Select-Object Description,HotFixID,InstalledOn -ErrorActio
   InstalledApps = SafeGet { Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Unin
   Services     = SafeGet { Get-Service | Where-Object { $_.Status -eq 'Running' } | Select-Object Name
   Processes    = SafeGet { Get-Process | Sort-Object -Property CPU -Descending | Select-Object -First 2
   Uptime       = SafeGet { (Get-CimInstance -ClassName Win32_OperatingSystem).LastBootUpTime }
   EventSummary = SafeGet { Get-EventLog -LogName System -Newest 100 | Group-Object EntryType
}

if ($info.OS -is [object]) {
   $info.OS = $info.OS | Select-Object OsName,OsVersion,OsArchitecture,WindowsProductName,Windo
}

$info | ConvertTo-Json -Depth 5 | Out-File -FilePath $jsonFile -Encoding utf8
$infoText = $info.GetEnumerator() | ForEach-Object { "$($_.Name):n$($_.Value | Out-String)n" } -join "
$infoText | Out-File -FilePath $textFile -Encoding utf8

Write-Host "Report saved to: $outFolder" -ForegroundColor Green
Write-Host "Computer: $($info.ComputerName)  User: $($info.UserName)"
Write-Host "OS:"; $info.OS | Format-List
Write-Host "Top running processes (by CPU):"
```

```
$info.Processes | Format-Table -AutoSize
```