```powershell
##############################################
### EMI Automation Script
### ECI.EMI.Automation.Prod.ps1
##############################################


### Get Parameters form Invoke-ServerRequest.ps1
###-------------------------------------------------
Param(
    [Parameter(Mandatory = $True,Position=0)][string]$RequestID,
    [Parameter(Mandatory = $True,Position=1)][string]$HostName,
    [Parameter(Mandatory = $True,Position=2)][string]$Env,
    [Parameter(Mandatory = $True,Position=3)][string]$AdministrativeUserName,
    [Parameter(Mandatory = $True,Position=4)][string]$AdministrativePassword,
    [Parameter(Mandatory = $False,Position=5)][string]$ConfigurationMode
)

#####################################
### Function: Set-TranscriptPath
#####################################
function Start-ECI.Transcript
{
    Param(
    [Parameter(Mandatory = $False)][string]$TranscriptPath,
    [Parameter(Mandatory = $False)][string]$TranscriptName,
    [Parameter(Mandatory = $False)][string]$HostName
    )

    function Generate-RandomAlphaNumeric
    {
        Param([Parameter(Mandatory = $False)][int]$Length)

        if(!$Length){[int]$Length = 15}

        ##ASCII
        #48 -> 57 :: 0 -> 9
        #65 -> 90 :: A -> Z
        #97 -> 122 :: a -> z

        for ($i = 1; $i -lt $Length; $i++)
        {
            $a = Get-Random -Minimum 1 -Maximum 4
            switch ($a)
            {
                1 {$b = Get-Random -Minimum 48 -Maximum 58}
                2 {$b = Get-Random -Minimum 65 -Maximum 91}
                3 {$b = Get-Random -Minimum 97 -Maximum 123}
            }
            [string]$c += [char]$b
        }

        Return $c
    }

    ### Stop Transcript if its already running
    try {Stop-transcript -ErrorAction SilentlyContinue} catch {}

    $TimeStamp  = Get-Date -format "yyyyMMddhhmss"
    $Rnd = (Generate-RandomAlphaNumeric)

    ### Set Default Path
    if(!$TranscriptPath){$global:TranscriptPath = "C:\Scripts\Transcripts"}

    ### Make sure path ends in "\"
    $LastChar = $TranscriptPath.substring($TranscriptPath.length-1)
    if ($LastChar -ne "\"){$TranscriptPath = $TranscriptPath + "\"}

    ### Create Transcript File Name
    if($TranscriptName)
```

```powershell
    {
        $global:TranscriptFile = $TranscriptPath + $HostName + "_PowerShell_transcript"
        + "." + $TranscriptName + "." + $Rnd + "." + $TimeStamp + ".txt"
    }
    else
    {
        $global:TranscriptFile = $TranscriptPath + "PowerShell_transcript" + "." + $Rnd
        + "." + $TimeStamp + ".txt"
    }
    ### Start Transcript Log
    Start-Transcript -Path $TranscriptFile -NoClobber
}

#~~~~~~~~~~~~~~~~~ Temporary Kludges ~~~~~~~~~~~~~~~~~~~~~~~
function cLoBbEr-HostName
{
    Param([Parameter(Mandatory = $True)][string]$HostName)

    $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' * 50)`r`n
    "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 50) -ForegroundColor Magenta

    $RNG = Get-Random -Minimum 1000 -Maximum 9999

    Write-Host "Original Request HostName  : " $HostName -ForegroundColor Magenta

    $HostName = $HostName + "-" + $RNG

    Write-Host "New RNG Host Name          : " $HostName -ForegroundColor Magenta

    $global:HostName = $HostName
    Return $HostName
}
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


### Set ECI Automation Environment Parameters
### ------------------------------------------------
function Set-ECI.RequestParameters
{
    param(
    [Parameter(Mandatory = $True)] [string]$RequestID,
    [Parameter(Mandatory = $True)] [string]$Env,
    [Parameter(Mandatory = $True)] [string]$AdministrativeUserName,
    [Parameter(Mandatory = $True)] [string]$AdministrativePassword
    )

    $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' * 75)`r`n
    "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75) -ForegroundColor Gray

    switch ($Env)
    {
        "Dev"          { $global:Environment = "Development" }
        "Stage"        { $global:Environment = "Staging"     }
        "Prod"         { $global:Environment = "Production"  }
        "Development"  { $global:Environment = "Development" }
        "Staging"      { $global:Environment = "Staging"     }
        "Production"   { $global:Environment = "Production"  }
    }

    $RunasUser = Whoami

    #Write-Host "Setting Global Variable ENV: " $Env -ForegroundColor Cyan

    [int]$global:RequestID         = $RequestID
    $global:Env                    = $Env
    $global:Environment            = $Environment
    $global:AdministrativeUserName = $AdministrativeUserName
    $global:AdministrativePassword = $AdministrativePassword

```

```powershell
    $Parameters = [ordered]@{
        RequestID              = $RequestID
        Env                    = $Env
        Environment            = $Environment
        AdministrativeUserName = $AdministrativeUserName
        AdministrativePassword = $AdministrativePassword
        RunasUser              = $RunasUser
    }

    Write-Host `r`n('=' * 75)`r`n  "Running PowerShell Worker Process as User: "
    $RunasUser `r`n('=' * 75)`r`n  -ForegroundColor DarkGray

    Return $Parameters
}

### Import ECI Modules
### ------------------------------------------------
function Import-ECI.Root.ModuleLoader
{
    Param([Parameter(Mandatory = $True)][string]$Env)

    $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' * 75)`r`n
    "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75) -ForegroundColor Cyan

    ######################################
    ### Bootstrap Module Loader
    ######################################

    ### Connect to the Repository & Import the ECI.ModuleLoader
    ### ------------------------------------------------------------------------
    $AcctKey         = ConvertTo-SecureString -String
    "VSRMGJZNI4vn0nf47J4bqVd5peNiYQ/8+ozlgzbuA1FUnn9hAoGRM9Ib4HrkxOyRJkd4PHE8j36+pfnCUw3o
    8Q==" -AsPlainText -Force
    $Credentials     = $Null
    $Credentials     = New-Object System.Management.Automation.PSCredential
    -ArgumentList "Azure\eciscripts", $AcctKey
    $RootPath        = "\\eciscripts.file.core.windows.net\clientimplementation"

    New-PSDrive -Name X -PSProvider FileSystem -Root $RootPath -Credential $Credentials
    -Scope Global

    .
    "\\eciscripts.file.core.windows.net\clientimplementation\Root\$Env\ECI.Root.ModuleLoa
    der.ps1" -Env $Env
}



### ================================================
### Invoke ECI.EMI.Automation.VM
### ================================================
function Invoke-ECI.EMI.Automation.VM
{
    Param(
            [Parameter(Mandatory = $True)][int]$ServerID,
            [Parameter(Mandatory = $True)][int]$RequestID,
            [Parameter(Mandatory = $True)][string]$HostName,
            [Parameter(Mandatory = $True)][string]$ConfigurationMode,
            [Parameter(Mandatory = $True)][string]$Env,
            [Parameter(Mandatory = $True)][string]$Environment
         )

    $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('=' * 100)`r`n
    "EXECUTING FUNCTION: " $FunctionName `r`n('=' * 100) -ForegroundColor Cyan

    ####################
    ### Provision VM
    ####################

```

```
195         $VMParameters = @{
196                 ServerID          = $ServerID
197                 RequestID         = $RequestID
198                 Environment       = $Environment
199                 ConfigurationMode = $ConfigurationMode
200         }
201
202         ### ECI.ConfigServer.Invoke-ProvisionVM.ps1
203         ###-------------------------------------------------
204         $File = "ECI.EMI.Automation.VM"
205         $FilePath =  "\\eciscripts.file.core.windows.net\clientimplementation\" +
            $Environment + "\ECI.Modules." + $Env + "\" + $File + "." + $Env + "\" + $File  +
            "." + $Env + ".ps1"
206         #Try
207         #{
208             . ($FilePath) @VMParameters
209         #}
210         #Catch
211         #{
212         #    Write-ECI.ErrorStack
213         #}
214     }
215
216     ### ================================================
217     ### Invoke ECI.EMI.Automation.OS
218     ### ================================================
219     function Invoke-ECI.EMI.Automation.OS
220     {
221         Param(
222                 [Parameter(Mandatory = $True)][int]$ServerID,
223                 [Parameter(Mandatory = $True)][string]$HostName,
224                 [Parameter(Mandatory = $True)][string]$ConfigurationMode,
225                 [Parameter(Mandatory = $True)][string]$Env,
226                 [Parameter(Mandatory = $True)][string]$Environment
227              )
228
229         $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('=' * 100)`r`n
            "EXECUTING FUNCTION: " $FunctionName `r`n('=' * 100) -ForegroundColor Cyan
230
231         ####################c
232         ### Configure OS
233         ####################
234
235         $OSParameters = @{
236                 ServerID                = $ServerID
237                 HostName                = $HostName
238                 ConfigurationMode       = $ConfigurationMode
239                 ServerRole              = $ServerRole
240                 IPv4Address             = $IPv4Address
241                 SubnetMask              = $SubnetMask
242                 DefaultGateway          = $DefaultGateway
243                 PrimaryDNS              = $PrimaryDNS
244                 SecondaryDNS            = $SecondaryDNS
245                 ClientDomain            = $ClientDomain
246                 AdministrativeUserName  = $AdministrativeUserName
247                 AdministrativePassword  = $AdministrativePassword
248         }
249
250         ### ECI.EMI.Automation.OSConfiguration.Invoke.ps1
251         ###-------------------------------------------------
252         $File = "ECI.EMI.Configure.OS"
253         $FilePath =  "\\eciscripts.file.core.windows.net\clientimplementation\" +
            $Environment + "\ECI.Modules." + $Env + "\" + $File + "." + $Env + "\" + $File +
            ".Invoke"  + "." + $Env + ".ps1"
254         Try
255         {
256             . ($FilePath) @OSParameters
257         }
258         Catch
```

```
259         {
260             Write-ECI.ErrorStack
261         }
262 
263     }
264 
265     ### ================================================
266     ### Invoke ECI.EMI.Automation.Role
267     ### ================================================
268     function InvokeECI.EMI.Automation.Role
269     {
270         $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('=' * 100)`r`n
            "EXECUTING FUNCTION: " $FunctionName `r`n('=' * 100) -ForegroundColor Cyan
271 
272         ####################
273         ### Configure Roles
274         ####################
275 
276         $RoleParameters = @{
277                 ServerID     = $ServerID
278                 HostName     = $HostName
279                 ServerRole   = $ServerRole
280                 BuildVersion = $BuildVersion
281         }
282 
283         ### ECI.ConfigServer.Invoke-ConfigureRoles.ps1
284         ###-----------------------------------------------
285         $File = "ECI.EMI.Automation.Role"
286         $FilePath =  "\\eciscripts.file.core.windows.net\clientimplementation\" +
            $Environment + "\ECI.Modules." + $Env + "\" + $File + "." + $Env + "\" + $File  +
            "." + $Env + ".ps1"
287 
288         Try
289         {
290             . ($FilePath) @RoleParameters
291         }
292         Catch
293         {
294 
295             Write-ECI.ErrorStack
296         }
297     }
298 
299     ### ================================================
300     ### Invoke ECI.EMI.Automation.QA
301     ### ================================================
302     function Invoke-ECI.EMI.Automation.QA
303     {
304         $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('=' * 100)`r`n
            "EXECUTING FUNCTION: " $FunctionName `r`n('=' * 100) -ForegroundColor Cyan
305 
306         ####################
307         ### Provision VM
308         ####################
309 
310         $QAParameters = @{
311                 ServerID          = $ServerID
312                 Environment       = $Environment
313                 ConfigurationMode = $ConfigurationMode
314         }
315 
316         ### ECI.ConfigServer.Invoke-ProvisionVM.ps1
317         ###-----------------------------------------------
318         $File = "ECI.EMI.Automation.QA"
319         $FilePath =  "\\eciscripts.file.core.windows.net\clientimplementation\" +
            $Environment + "\ECI.Modules." + $Env + "\" + $File + "." + $Env + "\" + $File  +
            "." + $Env + ".ps1"
320         Try
321         {
```

```
322             . ($FilePath) @QAParameters
323         }
324         Catch
325         {
326             Write-ECI.ErrorStack
327         }
328     }
329
330     ############
331     ### Main
332     ############
333     &{ ### Execute the Script
334
335         BEGIN
336         {
337             ######################################################################
338             ### START AUTOMATION PROCESS
339             ######################################################################
340
341             ### Set Variables
342             ###-----------------------------------
343             $global:AutomationStartTime   = (Get-Date)
344             $global:ProgressPreference    = "SilentlyContinue"
345             $global:ErrorActionPreference = "Stop"
346             $global:VerifyErrorCount      = 0
347             $global:Abort                 = $False
348             #$global:ECIError              = $False
349             Start-ECI.Transcript -TranscriptPath "C:\Scripts\_VMAutomationLogs\$HostName\"
                -TranscriptName "ECI.EMI.Automation.$Env.ps1" -HostName $HostName
350             Set-ECI.RequestParameters -RequestID $RequestID -Env $Env
                -AdministrativeUserName $AdministrativeUserName -AdministrativePassword
                $AdministrativePassword
351
352     ### PreCheck-ECI.EMI.Automation  ### need to develop this function!!!!!!!!!!!!!!!!
353
354             ### Default Configuration Mode
355             ###-----------------------------------
356             if(!$ConfigurationMode)
357             {
358                 ### Set Default Configuration Mode
359                 #$global:ConfigurationMode     = "Configure"   ### Configure/Report
360                 $global:ConfigurationMode     = "Report"      ### Configure/Report
361             }
362
363             elseif($ConfigurationMode)
364             {
365                 $global:ConfigurationMode = $ConfigurationMode
366             }
367             Write-Host "Configuration Mode: " $ConfigurationMode -ForegroundColor Yellow
368
369             ### Import Modules
370             ###-----------------------------------
371             Import-ECI.Root.ModuleLoader -Env $Env  # <----
                MOVE-TO-INVOKE-SERVERREQUEST.PS1 ?????
372             Set-ECI.PS.BufferSize
373             Import-ECI.EMI.Automation.VMWareModules -Env $Env -Environment $Environment
                -ModuleName "VMWare*" -ModuleVersion "10.0.0"
374
375             ### Get Parameters from SQL
376             ###-----------------------------------
377             $global:DevOps_ConnectionString  =
                "Server=automate1.database.windows.net;Initial Catalog=DevOps;User
                ID=devops;Password=JKFLKA8899*(*(32faiuynv;” # <-- Need to Encrypt Password
                !!!!!!
378             Get-ECI.EMI.Automation.SystemConfig -Env $Env -DevOps_ConnectionString
                $DevOps_ConnectionString
379             Get-ECI.EMI.Automation.BuildVersion
380             Get-ECI.EMI.Automation.ServerRequest
381             Get-ECI.EMI.Automation.ServerRole
```

```
382             Get-ECI.EMI.Automation.VMWareTemplate
383             Get-ECI.EMI.Automation.OSCustomizationSpec
384             Get-ECI.EMI.Automation.VMParameters
385             Get-ECI.EMI.Automation.OSParameters
386             #cLoBbEr-HostName -HostName $HostName
387             Create-ECI.EMI.Automation.VMName -GPID $GPID -HostName $HostName
388
389             ### ECI Error Log
390             $global:ECIErrorLogFile = $AutomationLogPath + "\" + $HostName + "\" +
                "ECIErrors_" + $HostName + ".txt" #!!!!!!!!!!!!!!!!!! WRITE TO SQL
391             Write-Host "ECIErrorLogFile:" $ECIErrorLogFile -ForegroundColor Magenta
392         }
393
394         PROCESS
395         {
396             ######################################################################
397             ### EXECUTE AUTOMATION PROCESS
398             ######################################################################
399
400             Create-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -HostName
                $HostName -ServerStatus "Server Request Recieved"
401
402             ### Get/Set Server Record
403             ###----------------------------
404             Check-ECI.EMI.Automation.ServerRecord -GPID $GPID -CWID $CWID -HostName $HostName
405
406             if($ServerExists -eq $False)
407             {
408
                    ###
                                        ### <---------- ConfigurationMode --- MAIN SWITCH !!!!!
409                 #$global:ConfigurationMode = "Configure"
410                 #Write-Host "Automation Configuration Mode: " $ConfigurationMode
                    -ForegroundColor DarkYellow
411
412                 $ServerParams = @{
413                     RequestID     = $RequestID
414                     GPID          = $GPID
415                     CWID          = $CWID
416                     HostName      = $HostName
417                     ServerRole    = $ServerRole
418                     BuildVersion  = $BuildVersion
419                 }
420                 Create-ECI.EMI.Automation.ServerRecord @ServerParams
421                 Get-ECI.EMI.Automation.ServerID -RequestID $RequestID
422             }
423             elseif($ServerExists -eq $True)
424             {
425                 $global:ServerID = $ServerID
426                 #$global:ConfigurationMode =
                    "Report"                                                        ###
                    <---------- ConfigurationMode --- MAIN SWITCH !!!!!
427                 Write-Host "Automation Configuration Mode: " $ConfigurationMode `r`n('-' *
                    50)`r`n -ForegroundColor DarkYellow
428             }
429
430             Create-ECI.EMI.Automation.CurrentStateRecord -ServerID
                (Get-ECI.EMI.Automation.ServerID -RequestID $RequestID)
431
432             ###----------------------------
433             ### Provision VM
434             ###----------------------------
435
436             $StatusParams = @{
437                 RequestID         = $RequestID
438                 ServerID          = $ServerID
439                 HostName          = $HostName
440                 VerifyErrorCount  = 0
441                 Abort             = $False
```

```
442             ServerStatus      = "VM Provisioning-Started"
443             Environment       = $Environment
444         }
445         #Update-ECI.EMI.Automation.ServerStatus @StatusParams
446         Update-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -ServerID
            $ServerID -HostName $HostName -VerifyErrorCount "0" -Abort $False -ServerStatus
            "VM Provisioning-Started"
447
448         $VMParams = @{
449             ServerID          = $ServerID
450             HostName          = $HostName
451             ConfigurationMode = $ConfigurationMode
452             Env               = $Env
453             Environment       = $Environment
454         }
455         #Invoke-ECI.EMI.Automation.VM @VMParams
456         Invoke-ECI.EMI.Automation.VM -ServerID $ServerID -RequestID $RequestID
            -HostName $HostName -ConfigurationMode $ConfigurationMode -Env $Env
            -Environment $Environment
457
458         $ServerParams = @{
459             ServerID          = $ServerID
460             VMName            = $VMName
461             ServerUUID        = $ServerUUID
462             vCenterUUID       = $vCenterUUID
463             VMID              = $VMID
464         }
465         #Update-ECI.EMI.Automation.ServerRecord @ServerParams
466         Update-ECI.EMI.Automation.ServerRecord -ServerID $ServerID -VMName $VMName
            -ServerUUID $ServerUUID -vCenterUUID $vCenterUUID -VMID $VMID
467
468         $StatusParams = @{
469             RequestID         = $RequestID
470             ServerID          = $ServerID
471             HostName          = $HostName
472             VerifyErrorCount  = $VerifyErrorCount
473             Abort             = $False
474             ElapsedTime       = $VMElapsedTime
475             ServerStatus      = "VM Provisioning-Completed"
476         }
477         #Update-ECI.EMI.Automation.ServerStatus @StatusParams
478         Update-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -ServerID
            $ServerID -HostName $HostName -VerifyErrorCount $VerifyErrorCount -Abort $False
            -ElapsedTime $VMElapsedTime -ServerStatus "VM Provisioning-Completed"
479
480         ###-----------------------------
481         ### Configure OS
482         ###-----------------------------
483         $StatusParams = @{
484             RequestID         = $RequestID
485             ServerID          = $ServerID
486             HostName          = $HostName
487             VerifyErrorCount  = 0
488             Abort             = $False
489             ServerStatus      = "OS Configuration-Started"
490         }
491         #Update-ECI.EMI.Automation.ServerStatus @StatusParams
492         Update-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -ServerID
            $ServerID -HostName $HostName -VerifyErrorCount 0 -Abort $False -ServerStatus
            "OS Configuration-Started"
493
494         $ConfigureOSParams = @{
495             ServerID          = $ServerID
496             HostName          = $HostName
497             ConfigurationMode = $ConfigurationMode
498             Env               = $Env
499             Environment       = $Environment
500         }
501         #Invoke-ECI.EMI.Automation.OS @ConfigureOSParams
```

```
502           Invoke-ECI.EMI.Automation.OS -ServerID $ServerID -HostName $HostName
              -ConfigurationMode $ConfigurationMode -Env $Env -Environment $Environment
503
504           $StatusParams = @{
505               RequestID         = $RequestID
506               ServerID          = $ServerID
507               HostName          = $HostName
508               VerifyErrorCount  = $VerifyErrorCount
509               Abort             = $False
510               ElapsedTime       = $OSElapsedTime
511               ServerStatus      = "OS Configuration-Completed"
512           }
513           #Update-ECI.EMI.Automation.ServerStatus @StatusParams
514           Update-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -ServerID
              $ServerID -HostName $HostName -VerifyErrorCount $VerifyErrorCount -Abort $False
              -ElapsedTime $OSElapsedTime -ServerStatus "OS Configuration-Completed"
515
516           ###-----------------------------
517           ### Configure Role
518           ###-----------------------------
519           #Update-ServerStatus -RequestID $RequestID -ServerID $ServerID -HostName
              $HostName -Verify $Verify -Abort $Abort -ServerStatus "Configuring Role"
520           #Invoke-ECI.EMI.Automation.Role
521
522           ###-----------------------------
523           ### QA Server
524           ###-----------------------------
525           #Update-ServerStatus -RequestID $RequestID -ServerID $ServerID -HostName
              $HostName -Verify $Verify -Abort $Abort -ServerStatus "Configuring Role"
526           Invoke-ECI.EMI.Automation.QA
527
528       }
529
530       END
531       {
532           ######################################################################
533           ### END AUTOMATION PROCESS
534           ######################################################################
535
536           ### Remove ECI Modules from Guest
537           ###-----------------------------------------
538           Delete-ECI.EMI.Automation.ECIModulesonVMGuest
539
540           $script:AutomationStopTime = Get-Date
541           $global:AutomationTime = ($AutomationStopTime-$AutomationStartTime)
542
543           ### Build Complete: Update Server Final Status
544           ###-----------------------------------------
545           $StatusParams = @{
546               RequestID         = $RequestID
547               ServerID          = $ServerID
548               HostName          = $HostName
549               VerifyErrorCount  = $VerifyErrorCount
550               Abort             = $False
551               ElapsedTime       = $AutomationTime
552               ServerStatus      = "Build Complete"
553           }
554           #Update-ECI.EMI.Automation.ServerStatus @StatusParams
555           Update-ECI.EMI.Automation.ServerStatus -RequestID $RequestID -ServerID
              $ServerID -HostName $HostName -VerifyErrorCount $VerifyErrorCount -Abort $False
              -ElapsedTime $AutomationTime -ServerStatus "Build Complete"
556
557           ### Write Status
558           ###-----------------------------------------
559           Write-Host `r`n('=' * 100)`r`n `r`n('*' * 100)`r`n`r`n('-' * 100)`r`n(' ' *
              25)  "ALL PROVISIONING & CONFIGURATION IS COMPLETE!" `r`n('-' *
              100)`r`n`r`n('*' * 100)`r`n `r`n('=' * 100)`r`n`r`n`r`n`r`n -ForegroundColor Cyan
560
561           ### Get Server Current State from SQL
```

```
562           ###-----------------------------------------
563           Write-Host `n`n('-' * 100)`n (' ' * 25)"GETTING SERVER SUMMARY FROM DATABASE"
              `n('-' * 100)`n -ForegroundColor Gray
564           Get-ECI.EMI.Automation.ServerRequest-SQL      -RequestID $RequestID
565           Get-ECI.EMI.Automation.ServerRecord-SQL       -ServerID $ServerID
566           Get-ECI.EMI.Automation.ServerCurrentState-SQL -ServerID $ServerID
567           Get-ECI.EMI.Automation.ServerDesiredState-SQL -ServerID $ServerID
568           Get-ECI.EMI.Automation.ServerConfigLog-SQL    -ServerID $ServerID
569
570           ##########################################################
571           ### Write Server Build Tag & Email Notification
572           ##########################################################
573           Send-ECI.ServerStatus -ServerID $ServerID -Abort $Abort -VerifyErrorCount
              $VerifyErrorCount
574
575           ### Write Script End
576           ###-----------------------------------------
577           Write-Host `r`n`r`n('=' * 75)`n "ECI.EMI.Automation: Total Execution Time:`t"
              $AutomationTime `r`n('=' * 75)`r`n -ForegroundColor Gray
578           Write-Host "================= END ALL AUTOMATION SCRIPTS ================= "
              -ForegroundColor Gray
579           Stop-Transcript
580
581           $ECIDebuggingMode = $False
582           if($ECIDebuggingMode -eq $True)
583           {
584               Write-ECI.ErrorStack -Detailed -NoExit | Out-File -FilePath $TranscriptFile
                  -Append ### REMEMBER! Errors are expected from try/catch tests!
585           }
586
587           Write-Host "END ALL AUTOMATION SCRIPTS:" `r`n (Get-Date)
588           ### END ALL AUTOMATION SCRIPTS
589       }
590   }
591
592   ### uNDer cOnStRucTioN ####################
593   ### eNd cOnStRucTioN ####################
594
595
```