**Table of Contents**

[Introduction: 2](#_Toc51854003)

[Initial Browser Settings: 2](#_Toc51854004)

[Initial Set Up: 4](#_Toc51854005)

[Initial Login: 5](#_Toc51854006)

[Create Dummy Files: 6](#_Toc51854007)

[Running Environments: 7](#_Toc51854008)

[Invoking/Calling another PS Script: 7](#_Toc51854009)

[Create Folder: 9](#_Toc51854010)

[Create Multiple Folders: 10](#_Toc51854011)

[Create Multiple Folders on Multiple Libraries: 12](#_Toc51854012)

[Create Sub Folder: 15](#_Toc51854022)

[Rename Folder: 16](#_Toc51854023)

[Rename Library: 17](#_Toc51854024)

[Rename File: 18](#_Toc51854025)

[Rename File with Extension: 19](#_Toc51854026)

[Rename Multiple Files: 20](#_Toc51854027)

[Download a File by a Specific Path: 21](#_Toc51854028)

[Download Multiple Files by a Specific Folder: 22](#_Toc51854029)

[Get Folder Permissions Report: 23](#_Toc51854030)

[Assign Folder Permissions: 26](#_Toc51854031)

[Assign Folder Permissions via Restrictions: 27](#_Toc51854032)

[Un Assign Folder Permissions via Restrictions: 28](#_Toc51854033)

[Assign Folder Permissions via SP Group: 28](#_Toc51854034)

[Delete a Large Folder: 29](#_Toc51854035)

[Delete a File by it’s Path: 30](#_Toc51854036)

[Assign Multi Folder Permissions via CSV: 31](#_Toc51854037)

[Simple Upload: 33](#_Toc51854038)

[Simple Upload Files into a Folder: 35](#_Toc51854039)

[Simple Download: 36](#_Toc51854040)

[External User Authentication: 38](#_Toc51854041)

[Login Included: 39](#_Toc51854042)

[Using CSV: 39](#_Toc51854043)

[Using Arguments: 39](#_Toc51854044)

[Total Scripts: 39](#_Toc51854045)

[Summary: 39](#_Toc51854046)

# Introduction:

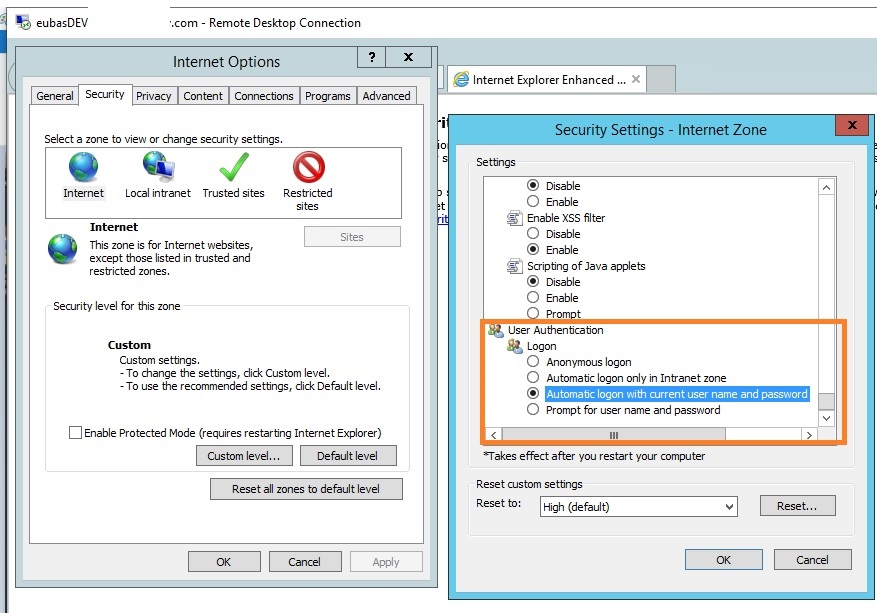
Through this User Guide we are trying to cover most of the major areas of Folder collaboration with related Files management of the given SPO Libraries.

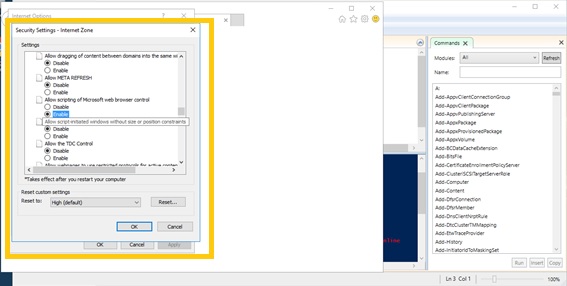
We are using Patterns and Practices module PnP PowerShell libraries to have a smart experience and generating the related outputs for all kinds of business needs.

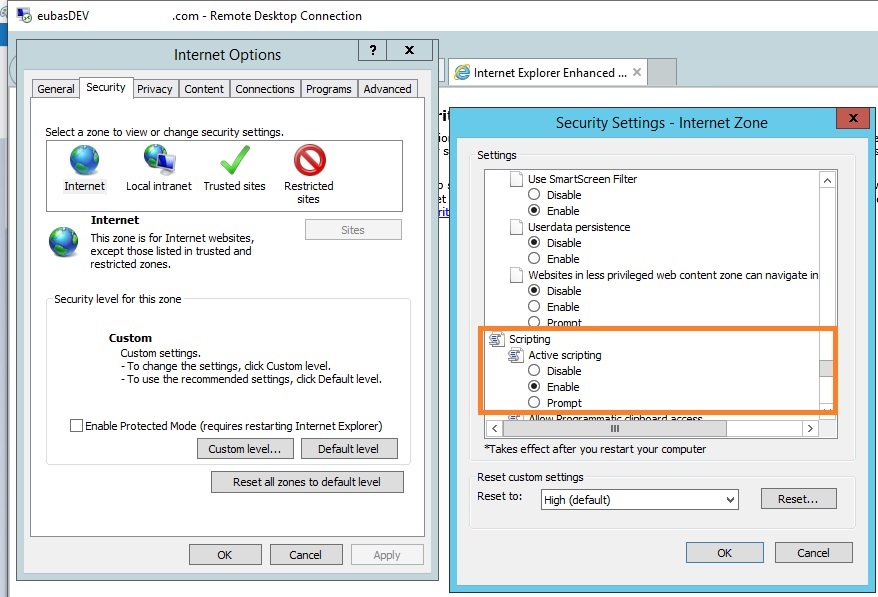
For more reference please refer here: <https://docs.microsoft.com/en-us/powershell/module/sharepoint-pnp/?view=sharepoint-ps>

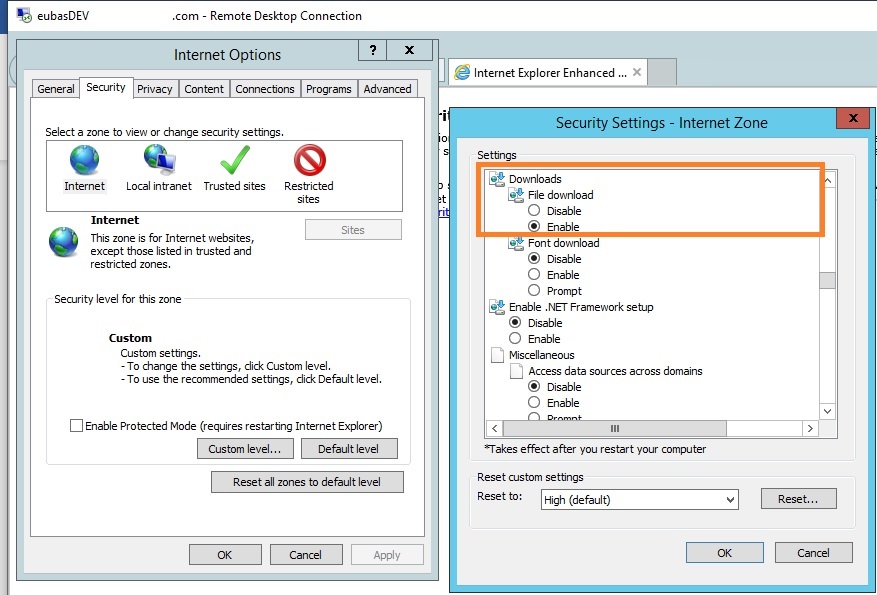
# Initial Browser Settings:

Please open the Primary/Parent Browser in your machine >> Click the Top Right corner Cog >> Go To Internet Options >> Security Tab >> Custom Level >> Enable the below options as highlighted >> Click Ok >> Restart the Machine for all Settings to get reflected.









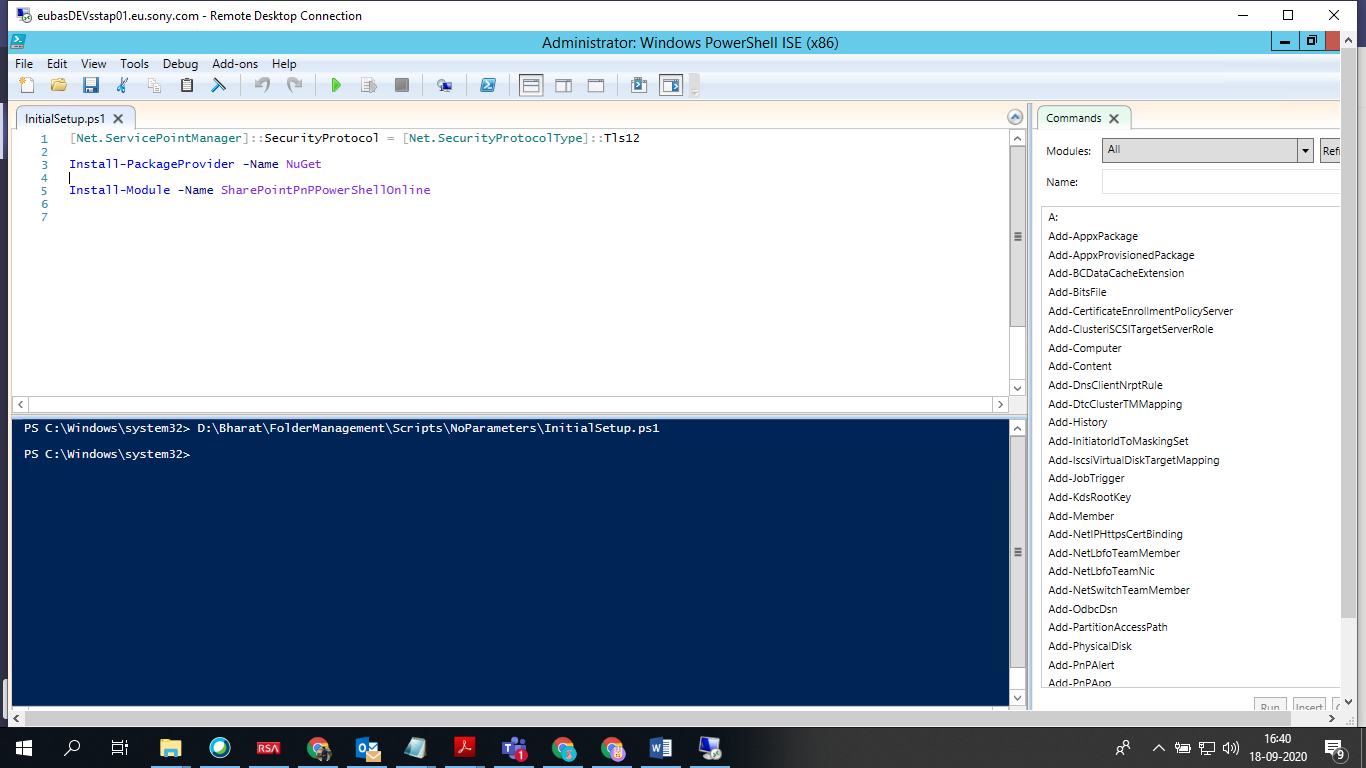
# Initial Set Up:

Please run the below set of PS code lines before working with all our PnP scripts.

***[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12***

***Install-PackageProvider -Name NuGet***

***Install-Module -Name SharePointPnPPowerShellOnline***

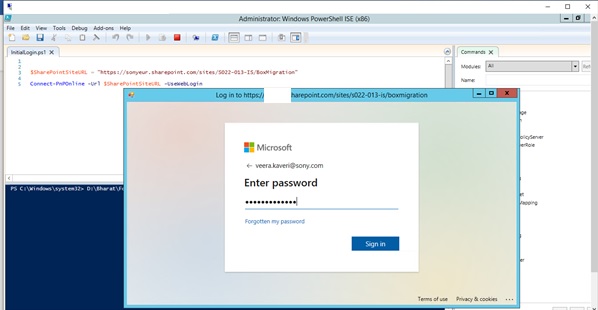


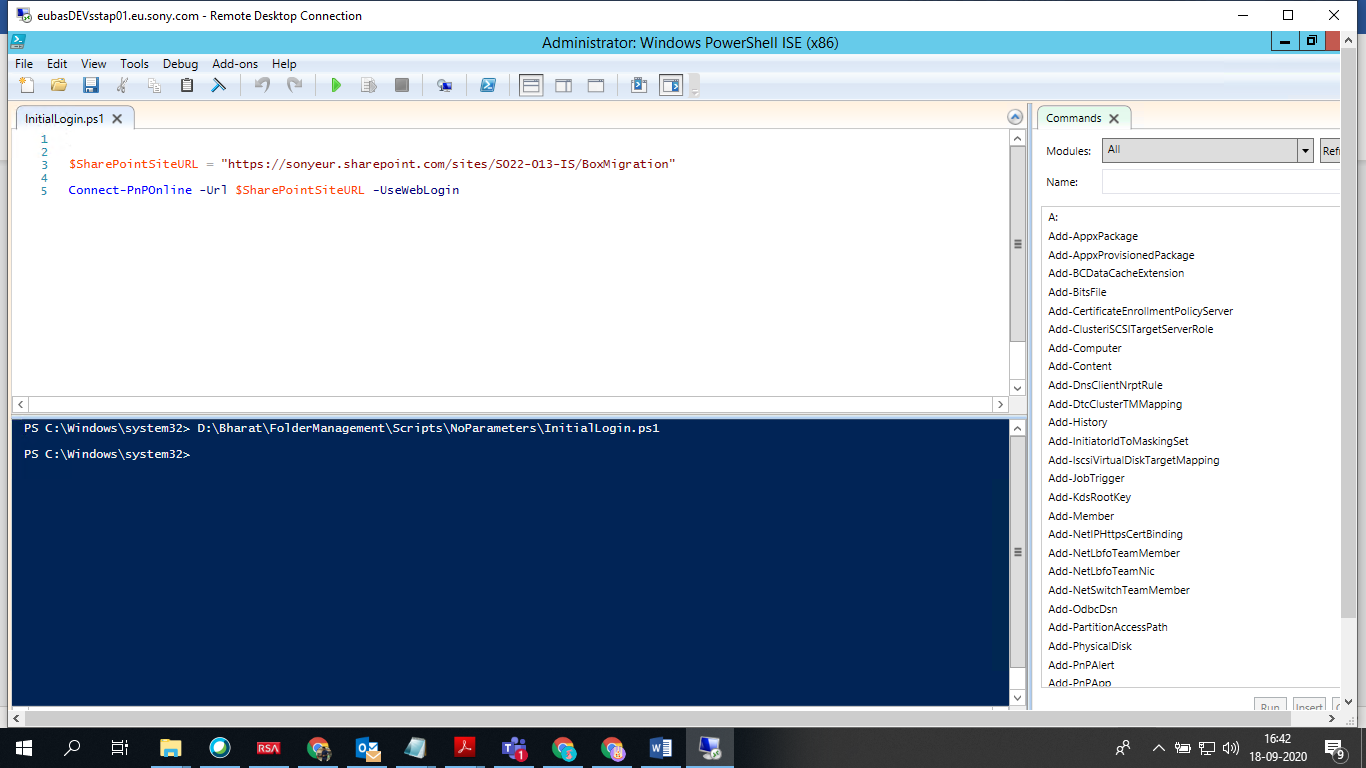
# Initial Login:

Please run the below set of PS code lines by replacing the highlighted details before working with all our PnP scripts that covers the Login process and stores the credentials and re-uses it whenever you run next set of PnP PowerShell scripts.

***$SharePointSiteURL = "<Site URL>"***

***Connect-PnPOnline -Url $SharePointSiteURL -UseWebLogin***

**



# Create Dummy Files:

Please run the below set of CMD script lines for creating all kinds of dummy files with required extensions by making and changing it to the directory.

***md <FilePath>***

***cd <FilePath>***

***Fsutil file createnew dummyfile1.pdf 2000000***

***Fsutil file createnew dummyfile2.doc 2000000***

***Fsutil file createnew dummyfile3.ppt 2000000***

***Fsutil file createnew dummyfile4.dat 2000000***

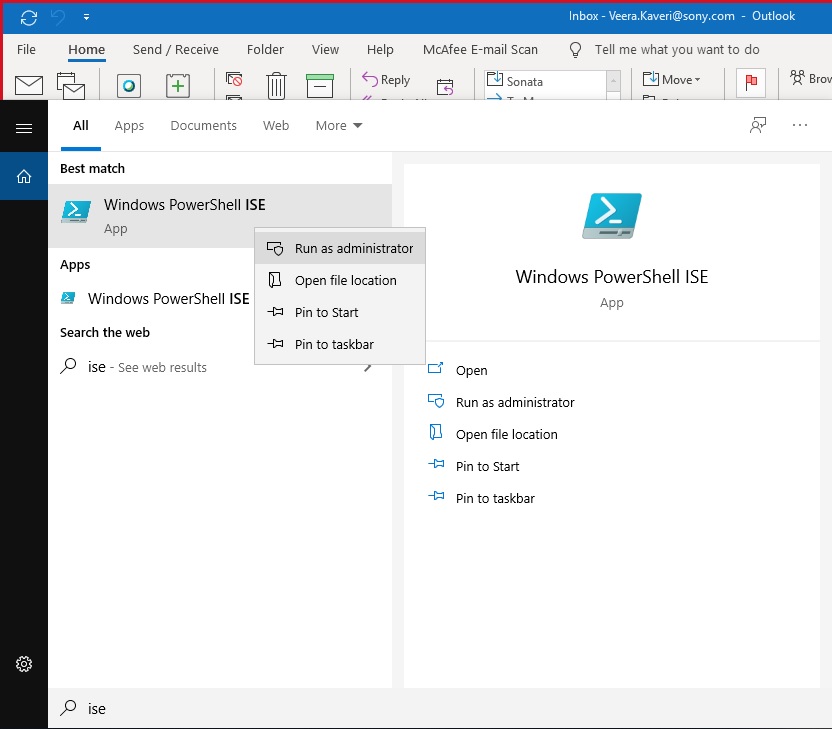
**Note: Only run in the CMD prompt**

Above CMD lines create dummy files with above titles and mentioned extensions each with 20 MB sizes.

We can use the above concept for all kinds of testing, validation, uploading, downloading of dummy files via PnP PS scripting on a given SPO Library’s Folders or Sub-Folders etc.

# Running Environments:

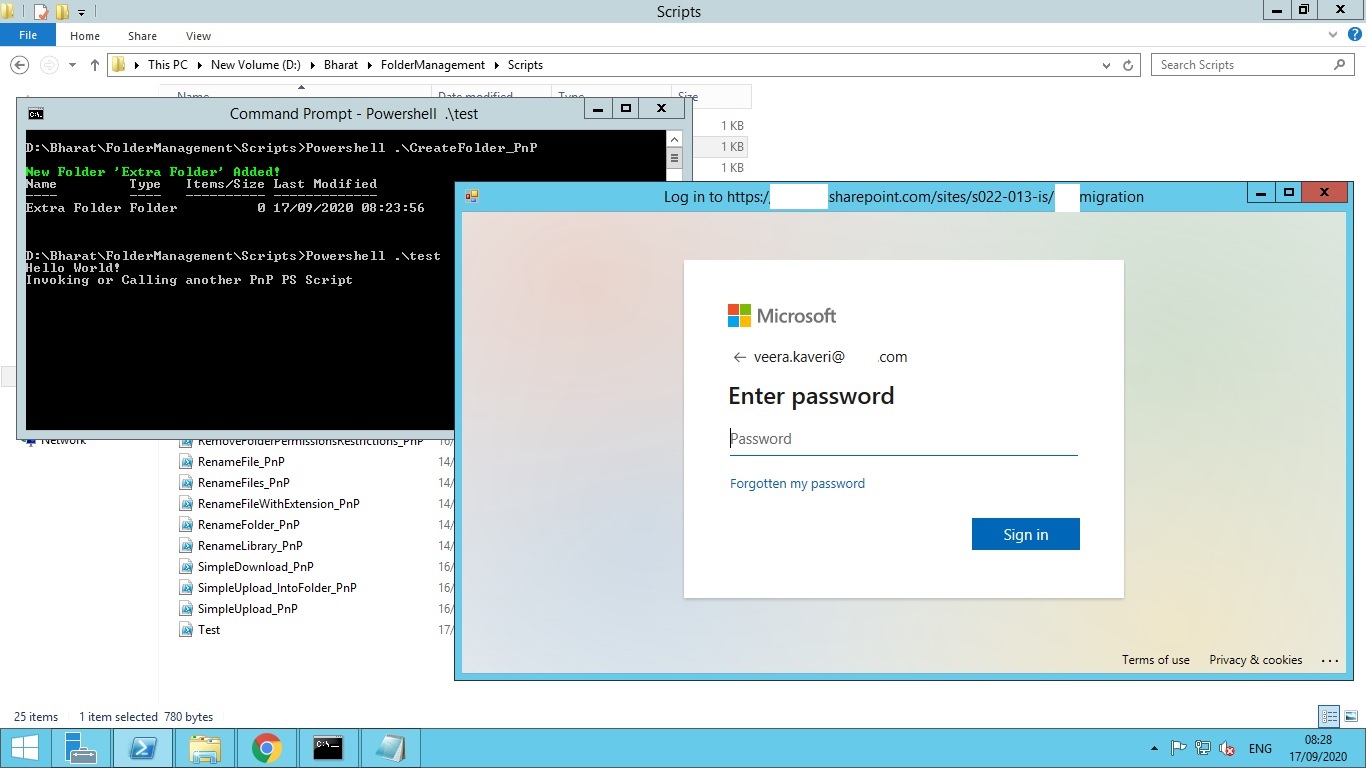
Suggested Environment to run all the below PnP PowerShell scripts are **using Windows PowerShell ISE** only **Right Click as Admin** as shown below:

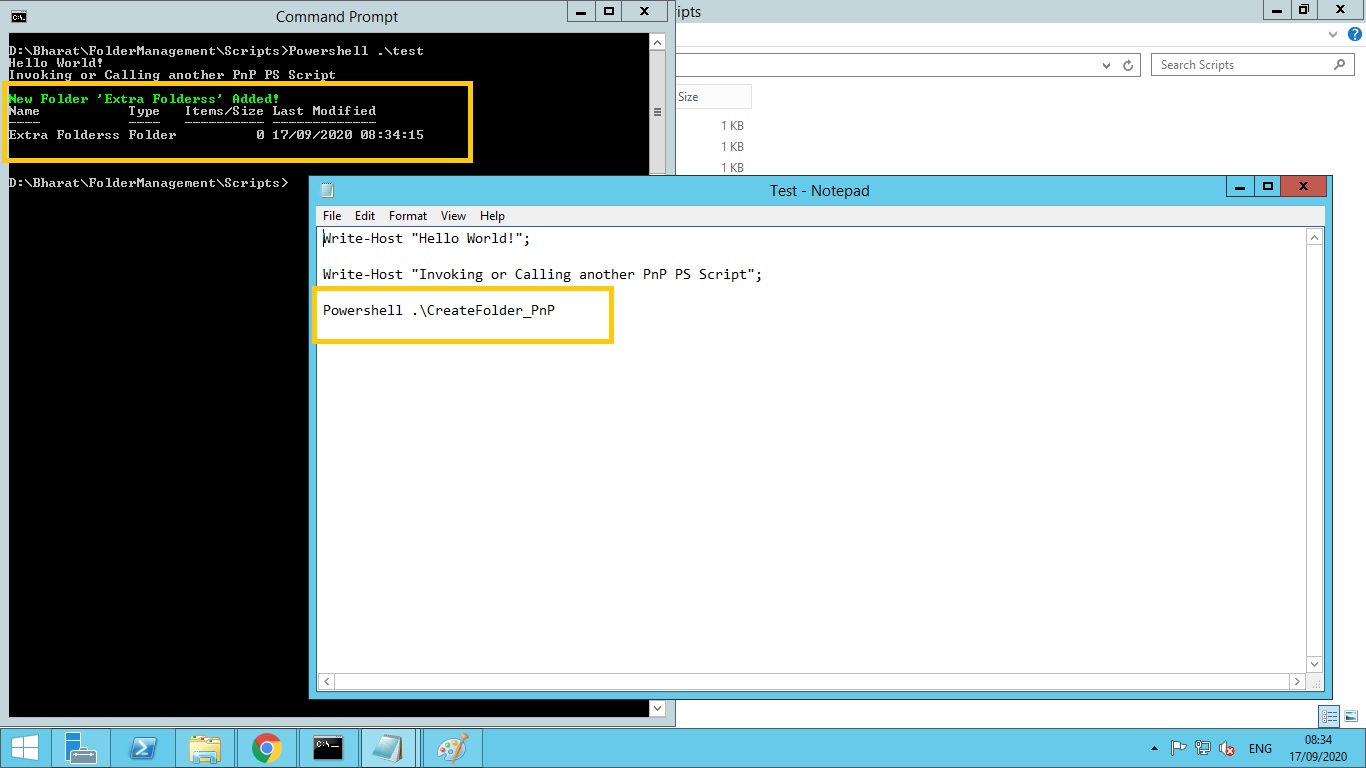


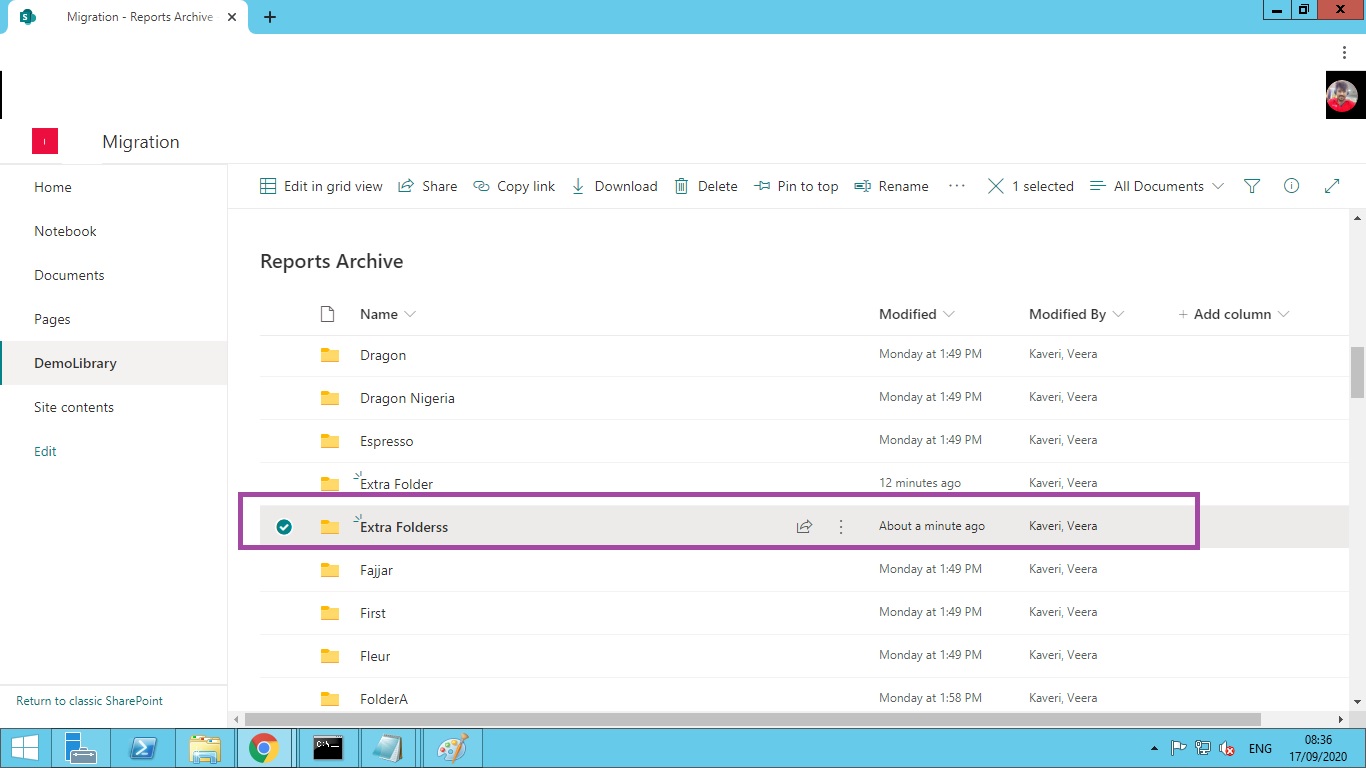
# **Invoking/Calling another PS Script:**

We can also use Command Line Prompt Tool to run below PnP scripts as an alternative as shown in the below illustrated example:

***Write-Host "Hello World!";  
  
Write-Host "Invoking or Calling another PnP PS Script";  
  
Powershell .\CreateFolder\_PnP***







**Note**

All the below PnP Scripts support **MFA [multi factor authentication]** which will be very helpful for the end user to enter his password at the run time only for once and rest of the times will login automatically with the same login details if he has permissions on the site collections which are referred in the PnP PowerShell scripts.

Below Scripts can be used across a Modern as well as Classic SPO Libraries.

# Create Folder:

Here is a simple way to create a folder on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell ISE .

#Config Variables

$SharePointSiteURL = "#########################"

$FolderName= "Extra Folder"

$RelativeURL= "/Reports Archive"

Try {

#Connect to PnP Online

Connect-PnPOnline -Url $SharePointSiteURL -UseWebLogin

#sharepoint online create folder powershell

Add-PnPFolder -Name $FolderName -Folder $RelativeURL -ErrorAction Stop

Write-host -f Green "New Folder '$FolderName' Added!"

}

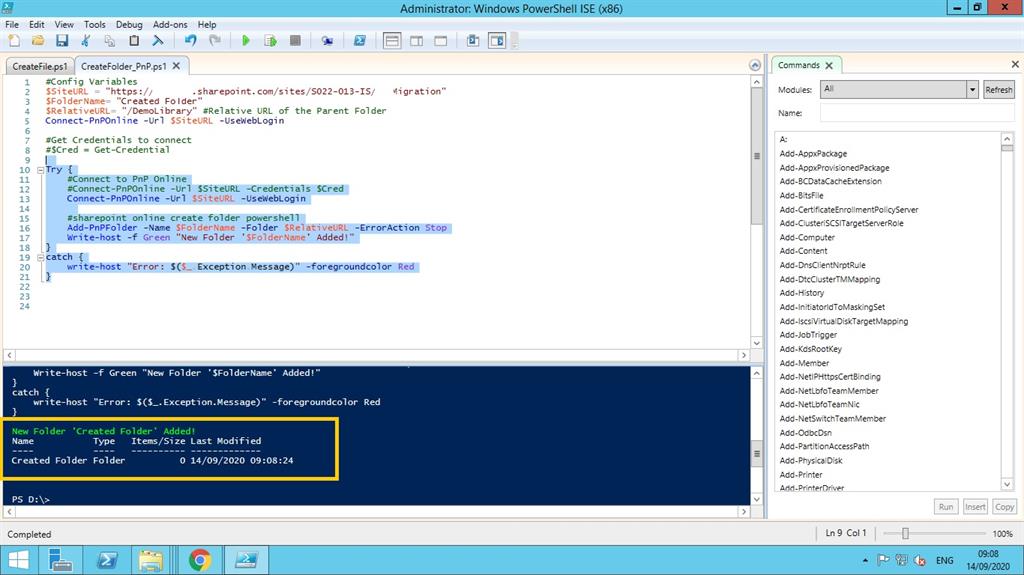
catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

}

Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script,



# Create Multiple Folders:

Here is a simple way to create multiple folders on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "#########################"

$CSVFilePath = "D:\Bharat\Folders\_SameLib.csv"

Try {

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get the CSV file

$CSVFile = Import-Csv $CSVFilePath

#Read CSV file and create document document library

ForEach($Row in $CSVFile)

{

#Create Folder if it doesn't exist

Resolve-PnPFolder -SiteRelativePath $Row.FolderSiteRelativeURL | Out-Null

Write-host "Ensured Folder:"$Row.FolderSiteRelativeURL -f Green

}

}

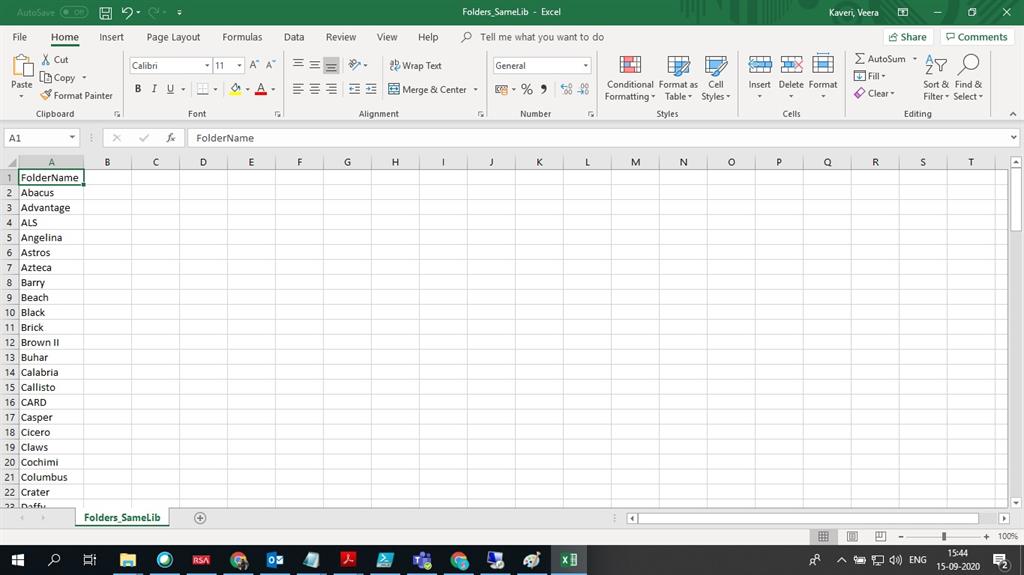
catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

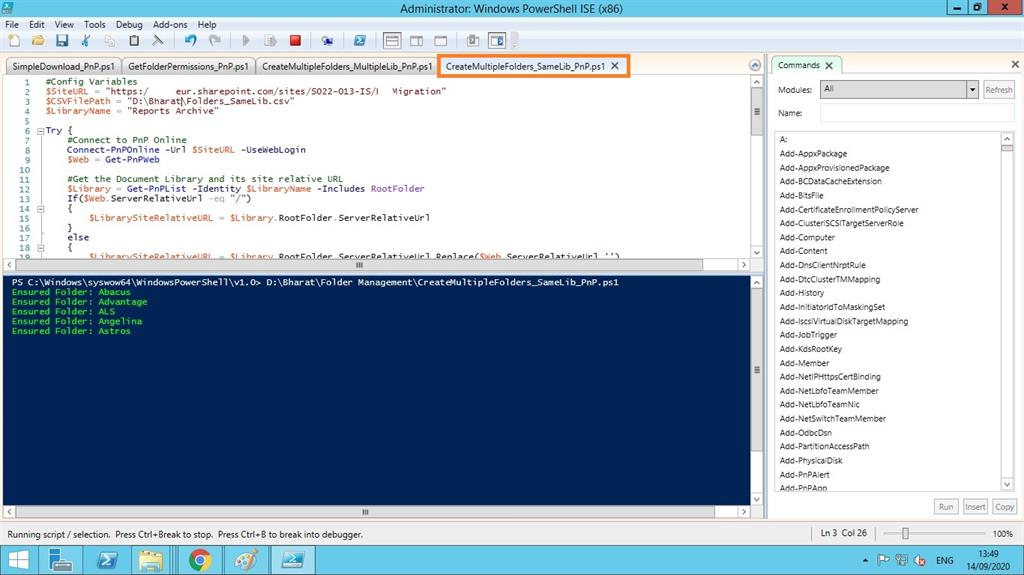
}

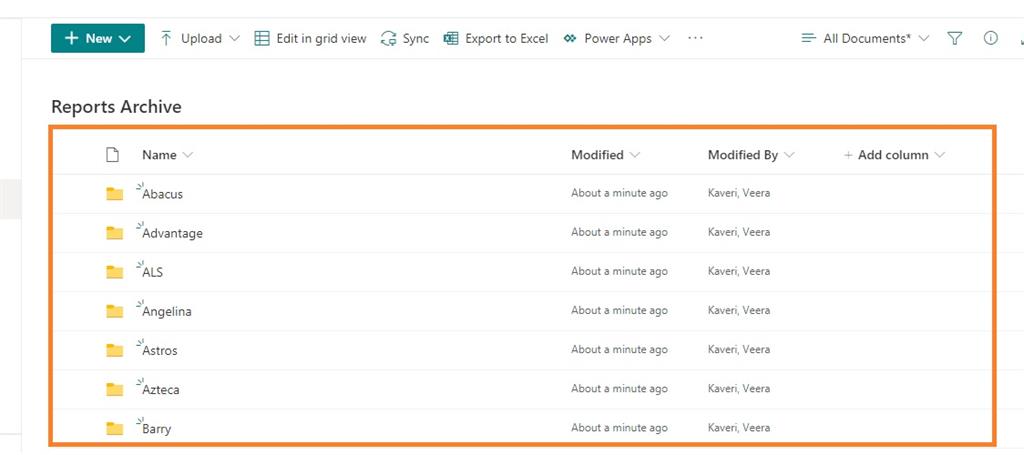
Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

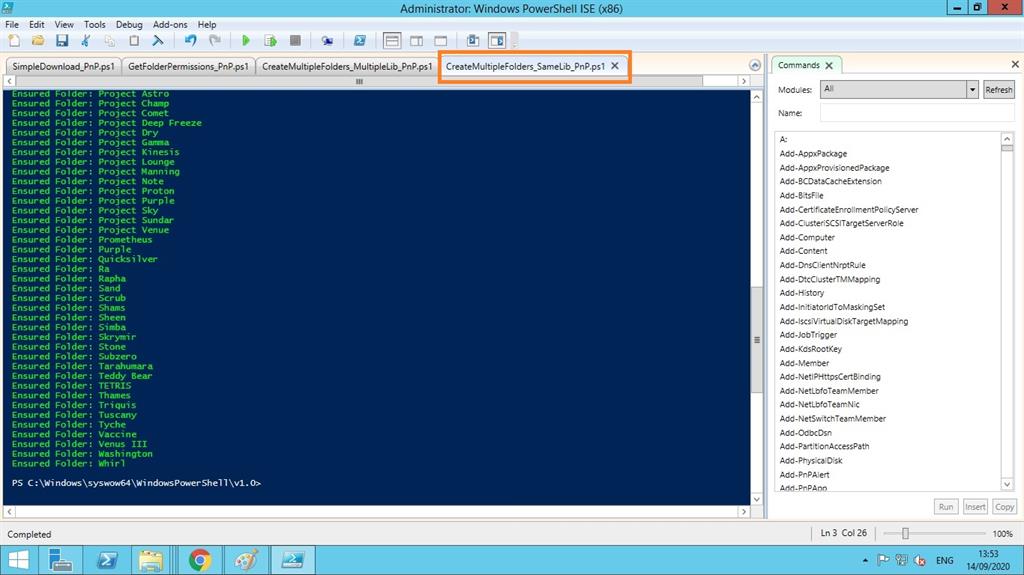
Above csv file mentioned in the script should look like this with column, values, sheet name same as file name saved etc. as shown below:



Here is the snapshot after a successful running of the script,







# Create Multiple Folders on Multiple Libraries:

# Here is a simple way to create multiple folders on a given multiple SPO libraries using PnP PowerShell.

# Here's the PnP PowerShell code to execute on a windows PowerShell.

# 

#Config Variables

$SiteURL = "#########################”

$CSVFilePath = "D:\Bharat\Folders\_MultipleLibs.csv"

Try {

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get the CSV file

$CSVFile = Import-Csv $CSVFilePath

#Read CSV file and create document document library

ForEach($Row in $CSVFile)

{

#Create Folder if it doesn't exist

Resolve-PnPFolder -SiteRelativePath $Row.FolderSiteRelativeURL | Out-Null

Write-host "Ensured Folder:"$Row.FolderSiteRelativeURL -f Green

}

}

catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

}

# 

# Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

# 

# Above csv file should look like this with column, values, sheet name same as file name saved etc. as shown below:

# https://www.c-sharpcorner.com/UploadFile/BlogImages/09152020051705AM/csv2.jpg

# 

# Here is the snapshot after a successful running of the script:

# https://www.c-sharpcorner.com/UploadFile/BlogImages/09152020051705AM/CreateMultipleFolders_MultiLib_PnP_Snap.jpg

# https://www.c-sharpcorner.com/UploadFile/BlogImages/09152020051705AM/CreateMultipleFolders_MultiLib_PnP_Snap2.jpg

# https://www.c-sharpcorner.com/UploadFile/BlogImages/09152020051705AM/CreateMultipleFolders_MultiLib_PnP_Snap3.jpg

Above concept can be used across a Modern as well as Classic SPO Libraries.

# Create Sub Folder:

Here is a simple way to create a sub-folder on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "##############################"

$FolderName= "Created Sub Folder"

$RelativeURL= "/DemoLibrary/Created Folder"

Connect-PnPOnline -Url $SiteURL -UseWebLogin

Try {

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#sharepoint online create folder powershell

Add-PnPFolder -Name $FolderName -Folder $RelativeURL -ErrorAction Stop

Write-host -f Green "New Sub Folder '$FolderName' Added!"

}

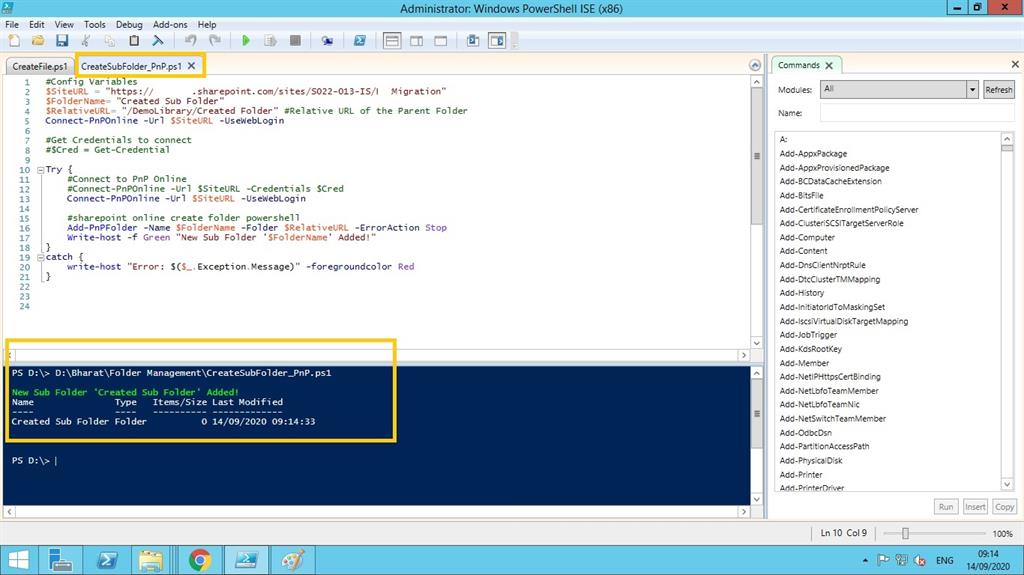
catch {

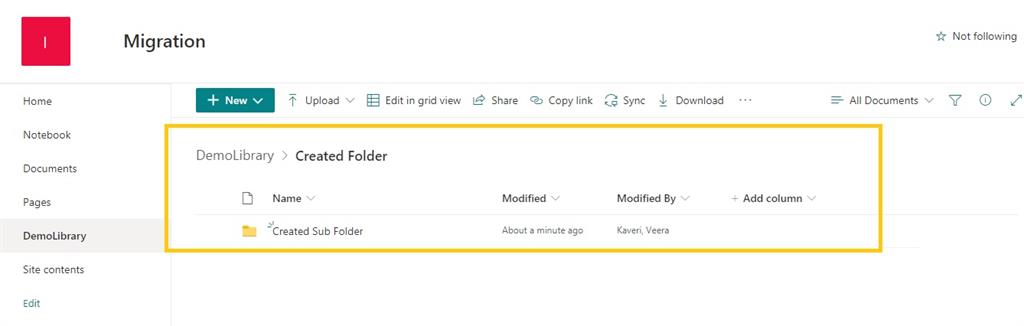
write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

}

Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

Here is the snapshot(s) after a successful running of the script:





# Rename Folder:

Here is a simple way to Rename a Folder on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "################"

$FolderURL= "Reports Archive/TestABC"

#Site Relative URL

$FolderNewName ="Reports Archive"

#Connect to PnP Online

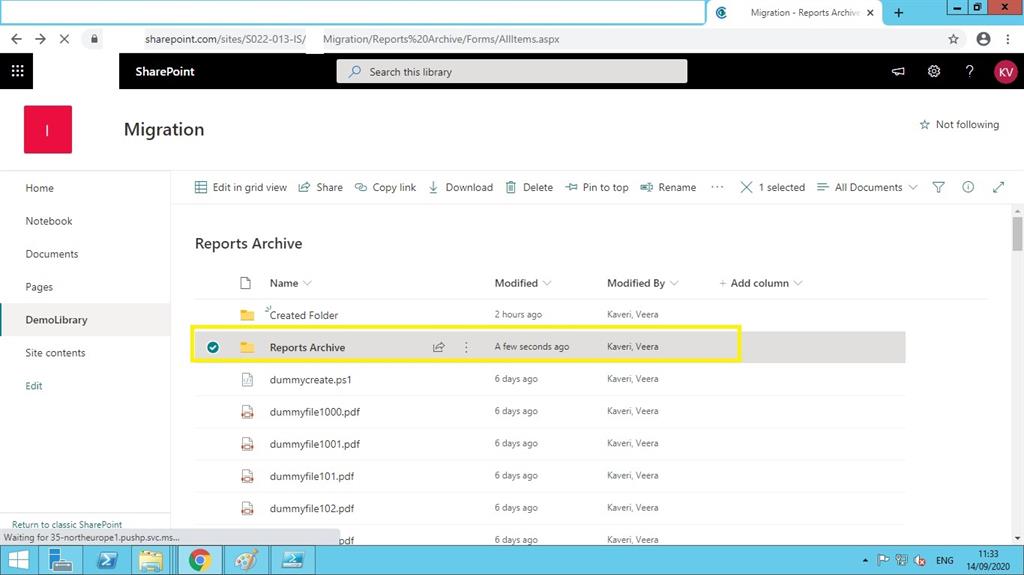
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Rename the Folder

Rename-PnPFolder -Folder $FolderURL -TargetFolderName $FolderNewName

Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



Above concept can be used across a Modern as well as Classic SPO Libraries.

# Rename Library:

Here is a simple way to Rename a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "##################"

$LibURL= "Reports Archive/"

#Site Relative URL

$FolderNewName ="Reports Updated"

#Connect to PnP Online

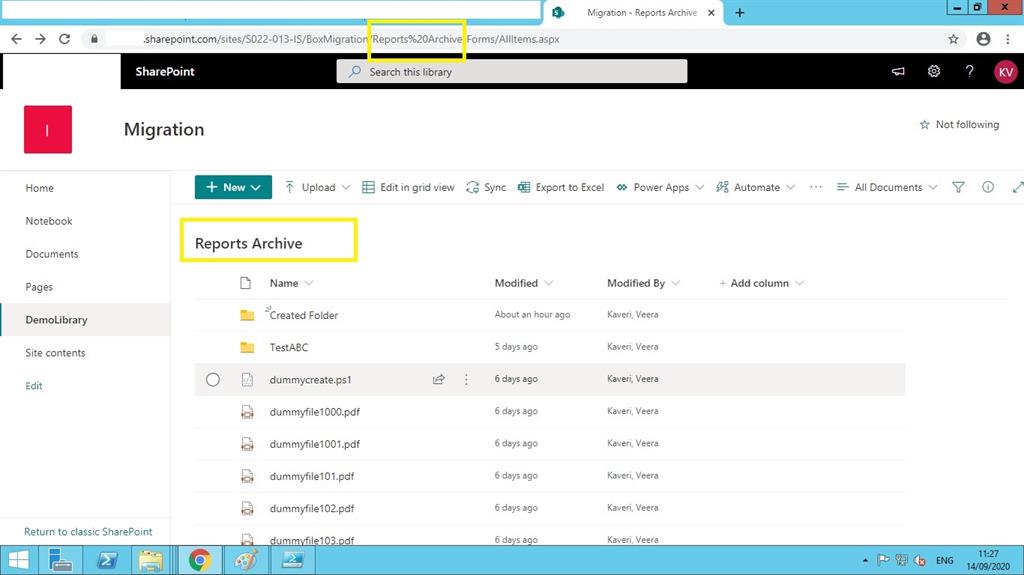
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Rename the Folder

Rename-PnPFolder -Folder $LibURL -TargetFolderName $FolderNewName

Please configure above highlighted points to have your own folder created via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



Above concept can be used across a Modern as well as Classic SPO Libraries.

# Rename File:

Here is simple way to rename file in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "#######################"

$FileURL= "Reports Archive/Created Folder/test2.txt"

$NewFileName ="UpdatedNAMEfile.txt"

#Connect to PnP Online

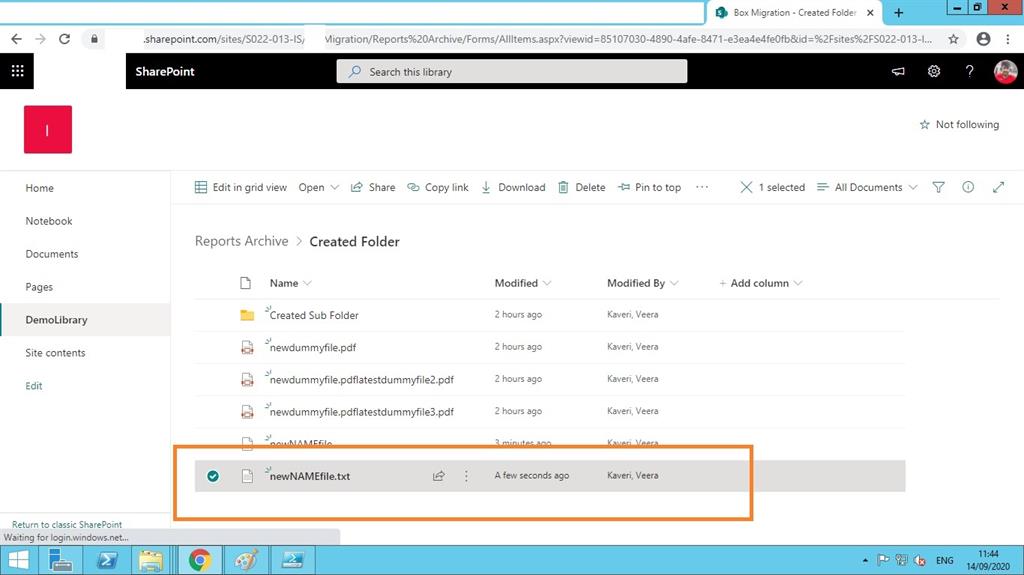
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Rename the File

Rename-PnPFile -SiteRelativeUrl $FileURL -TargetFileName $NewFileName -Force

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



Above concept can be used across a Modern as well as Classic SPO Libraries.

# Rename File with Extension:

Here is simple way to Rename file along with Extension in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "###############"

$FileURL= "Reports Archive/Created Folder/newNAMEfile.txt"

#Always include file extension

$NewFileName ="Topdf.pdf"

#Connect to PnP Online

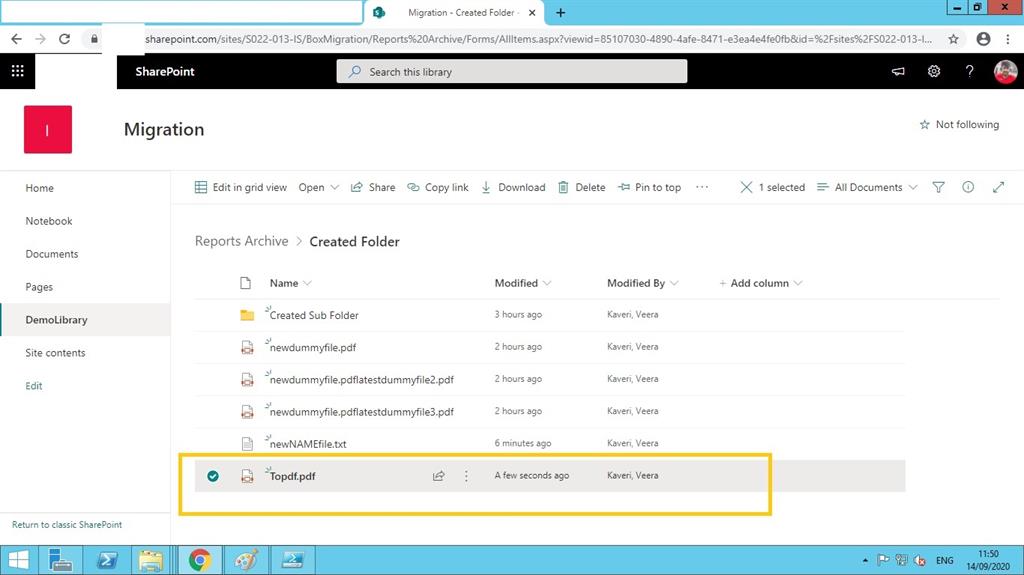
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Rename the File

Rename-PnPFile -SiteRelativeUrl $FileURL -TargetFileName $NewFileName -Force

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



Above concept can be used across a Modern as well as Classic SPO Libraries.

# Rename Multiple Files:

Here is a simple way to rename multiple files in a given SPO library using PnP powershell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "######################”

$LibraryName = "Reports Archive"

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get All Files from the document library

$Files = Get-PnPListItem -List $LibraryName -PageSize 1000 | Where {$\_["FileLeafRef"] -like "\*dummy\*"}

#Loop through each File

Write-host -f Green "Number of Files Found:"$Files.Count

ForEach($File in $Files)

{

Write-Host ("Renaming File '{0}' at {1}" -f $File["FileLeafRef"], $File["FileRef"])

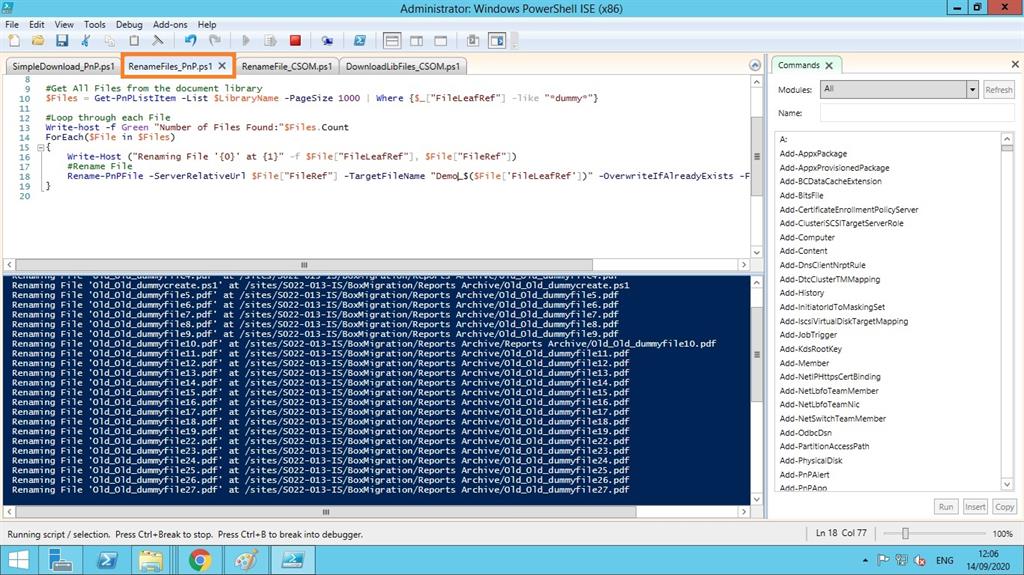
#Rename File

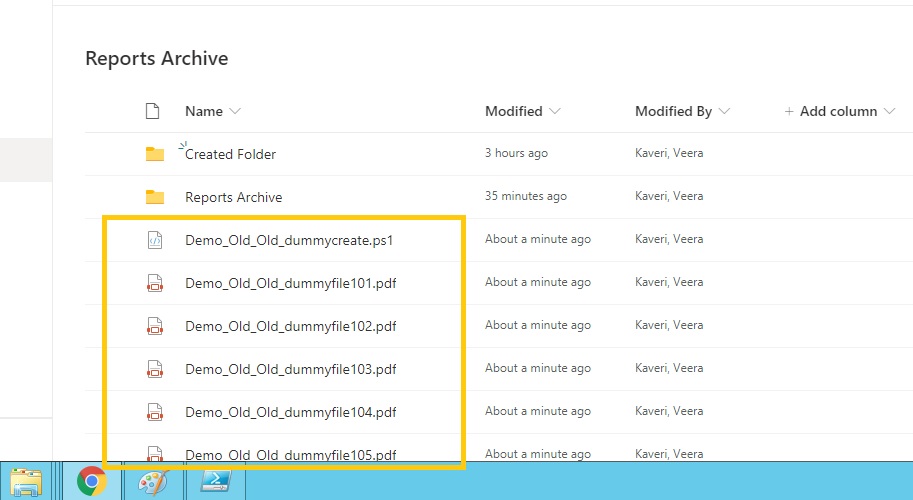
Rename-PnPFile -ServerRelativeUrl $File["FileRef"] -TargetFileName "SpecialDemo\_$($File['FileLeafRef'])" -OverwriteIfAlreadyExists -Force

}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:





Above concept can be used across a Modern as well as Classic SPO Libraries.

# Download a File by a Specific Path:

Here is a simple way to download a file by path specification in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "##########################”

$ListName="Reports Archive"

$FileRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder/Topdf.pdf"

$DownloadPath ="D:\Bharat"

Try {

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL -Useweblogin

#powershell download file from sharepoint online

Get-PnPFile -Url $FileRelativeURL -Path $DownloadPath -AsFile

}

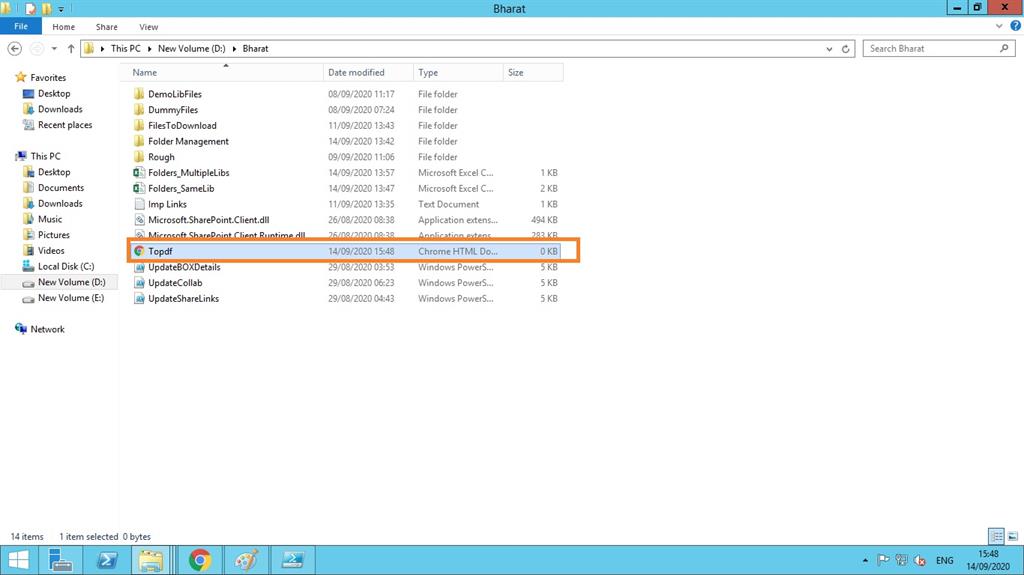
catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



# Download Multiple Files by a Specific Folder:

Here is a simple way to download all files by a specified path folder in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "##########################”

$ListName="Reports Archive"

$FileRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder/"

$DownloadPath ="D:\Bharat"

Try {

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL -Useweblogin

#powershell download file from sharepoint online

Get-PnPFile -Url $FileRelativeURL -Path $DownloadPath -AsFile

}

catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Above concept can be used across a Modern as well as Classic SPO Libraries.

# Get Folder Permissions Report:

Here is a simple way to get the total Permissions report for a given Folder on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

Function Get-PnPPermissions([Microsoft.SharePoint.Client.SecurableObject]$Object)

{

Try {

#Get permissions assigned to the Folder

Get-PnPProperty -ClientObject $Object -Property HasUniqueRoleAssignments, RoleAssignments

#Check if Object has unique permissions

$HasUniquePermissions = $Object.HasUniqueRoleAssignments

#Loop through each permission assigned and extract details

$PermissionCollection = @()

Foreach($RoleAssignment in $Object.RoleAssignments)

{

#Get the Permission Levels assigned and Member

Get-PnPProperty -ClientObject $RoleAssignment -Property RoleDefinitionBindings, Member

#Get the Principal Type: User, SP Group, AD Group

$PermissionType = $RoleAssignment.Member.PrincipalType

$PermissionLevels = $RoleAssignment.RoleDefinitionBindings | Select -ExpandProperty Name

#Remove Limited Access

$PermissionLevels = ($PermissionLevels | Where { $\_ –ne "Limited Access"}) -join ","

If($PermissionLevels.Length -eq 0) {Continue}

#Get SharePoint group members

If($PermissionType -eq "SharePointGroup")

{

#Get Group Members

$GroupMembers = Get-PnPGroupMembers -Identity $RoleAssignment.Member.LoginName

#Leave Empty Groups

If($GroupMembers.count -eq 0){Continue}

ForEach($User in $GroupMembers)

{

#Add the Data to Object

$Permissions = New-Object PSObject

$Permissions | Add-Member NoteProperty User($User.Title)

$Permissions | Add-Member NoteProperty Type($PermissionType)

$Permissions | Add-Member NoteProperty Permissions($PermissionLevels)

$Permissions | Add-Member NoteProperty GrantedThrough("SharePoint Group: $($RoleAssignment.Member.LoginName)")

$PermissionCollection += $Permissions

}

}

Else

{

#Add the Data to Object

$Permissions = New-Object PSObject

$Permissions | Add-Member NoteProperty User($RoleAssignment.Member.Title)

$Permissions | Add-Member NoteProperty Type($PermissionType)

$Permissions | Add-Member NoteProperty Permissions($PermissionLevels)

$Permissions | Add-Member NoteProperty GrantedThrough("Direct Permissions")

$PermissionCollection += $Permissions

}

}

#Export Permissions to CSV File

$PermissionCollection | Export-CSV $ReportFile -NoTypeInformation

Write-host -f Green "`n\*\*\* Folder Permission Report Generated Successfully!\*\*\*"

}

Catch {

write-host -f Red "Error Generating Folder Permission Report!" $\_.Exception.Message

}

}

#region \*\*\*Parameters\*\*\*

$SiteURL="#########################"

$ReportFile="D:\Bharat\FolderPermissionRpt.csv"

$FolderRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Project Champ"

#endregion

#Connect to the Site collection

Connect-PnPOnline -URL $SiteURL -UseWebLogin

#Get the Folder from URL

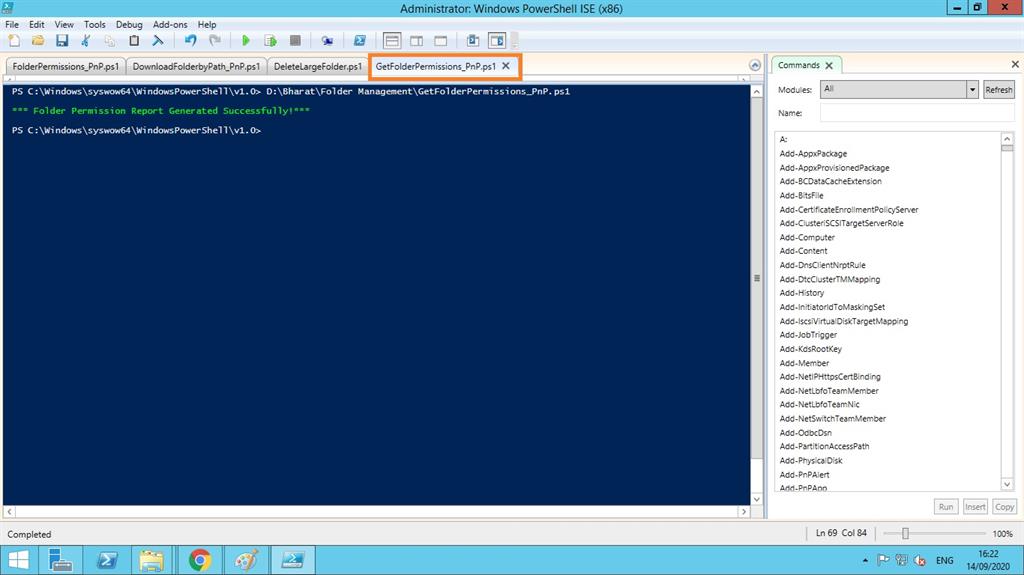
$Folder = Get-PnPFolder -Url $FolderRelativeURL

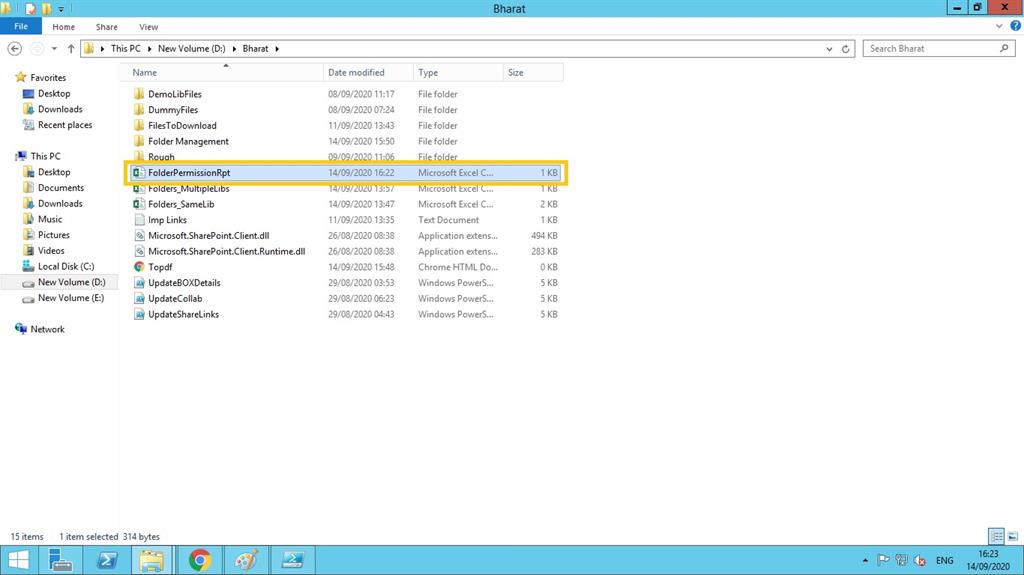
#Call the function to generate permission report

Get-PnPPermissions $Folder.ListItemAllFields

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here are the snapshots after a successful running of the script:







# Assign Folder Permissions:

Here is a simple way to assign a Folder permissions in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a windows PowerShell.

#Config Variables

$SiteURL = "###################"

$ListName="Reports Archive"

$FolderServerRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder"

$UserAccount = "suresh.gade@sony.com"

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -Useweblogin

#Get the Folder from URL

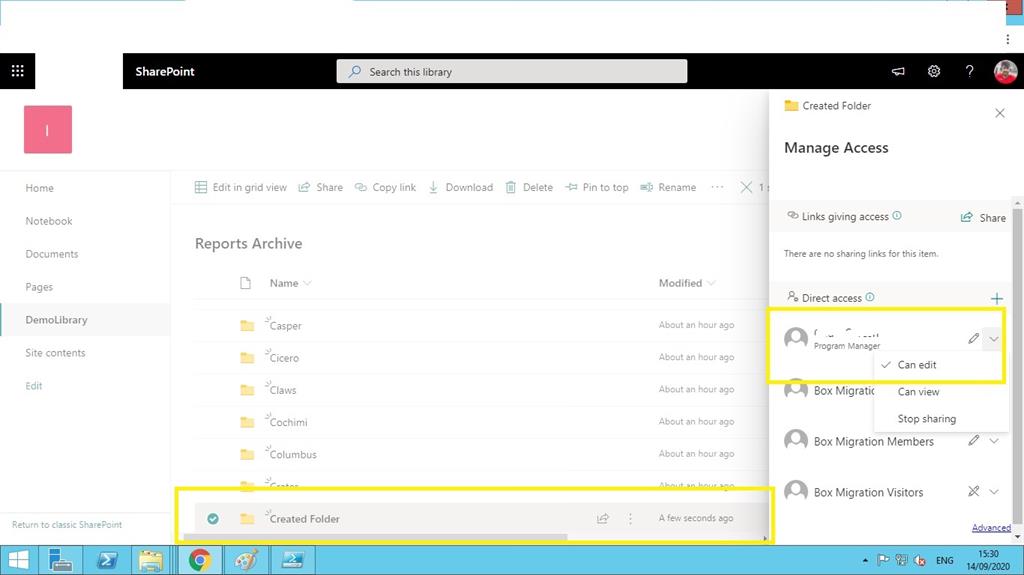
$Folder = Get-PnPFolder -Url $FolderServerRelativeURL

#Set Permission to Folder

Set-PnPListItemPermission -List $ListName -Identity $Folder.ListItemAllFields -User $UserAccount -AddRole 'Contribute'

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



# Assign Folder Permissions via Restrictions:

Here is a simple way to assign a Folder permissions via restrictions in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "###########################"

$ListName="Reports Archive"

$FolderServerRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder"

#Connect to PnP Online

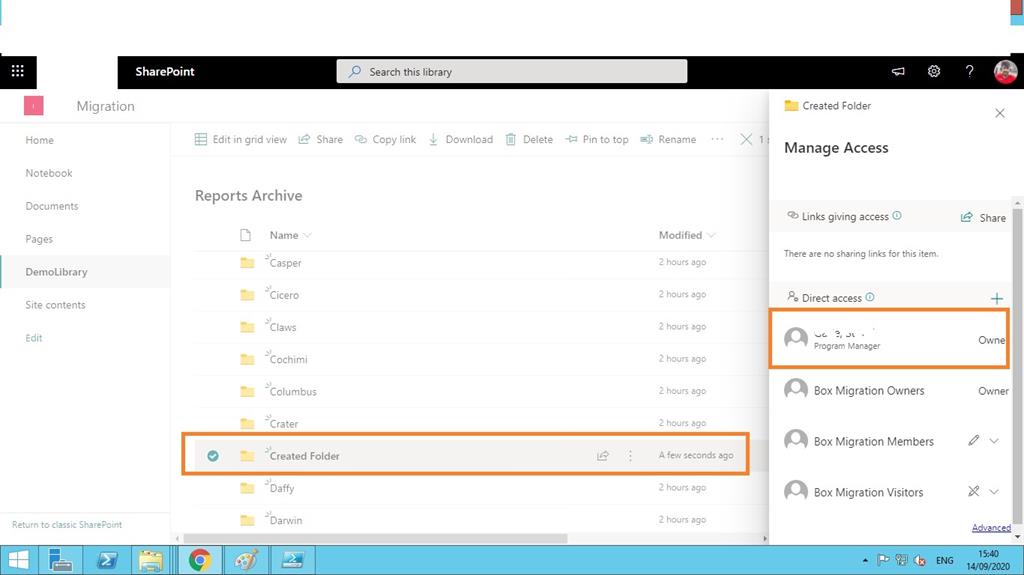
Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Set folder permissions - Add User

Set-PnPfolderPermission -List $ListName -identity $FolderServerRelativeURL -User "suresh.gade@sony.com" -AddRole "Full Control"

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Here is the snapshot after a successful running of the script:



# Un Assign Folder Permissions via Restrictions:

Here is a simple way to unassign/remove Folder permissions via restrictions in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "######################"

$ListName="Reports Archive"

$FolderServerRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder"

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#To remove user, use:- -RemoveRole

Set-PnPfolderPermission -List $ListName -identity $FolderServerRelativeURL -User "suresh.gade@sony.com" -RemoveRole "Full Control"

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

You can check manually by going to that respective folder if the permissions have been removed or not.

# Assign Folder Permissions via SP Group:

Here is a simple way to assign a Folder permissions via a SP group in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "##################”

$ListName="Reports Archive"

$FolderServerRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder"

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Grant folder permissions to SharePoint Group which should exist!

Set-PnPfolderPermission -List $ListName -identity $FolderServerRelativeURL -AddRole "Read" -Group "Box Migration Members"

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

**Prerequisites:**

Above mentioned SP group is expected to exist before either created by default or to be created separately in order to configure in the above script.

Above mentioned Folder is also expected to exist before or need to be created

separately in order to configure in the above script.

# Delete a Large Folder:

Here is a simple way to delete a Folder/Large Folder in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "################"

$ListName="Reports Archive"

$FolderServerRelativeURL = "/sites/S022-013-IS/BoxMigration/Reports Archive/Created Folder"

Try {

#Connect to PnP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get All Items from Folder in Batch

$ListItems = Get-PnPListItem -List $ListName -FolderServerRelativeUrl $FolderServerRelativeURL -PageSize 2000 | Sort-Object ID -Descending

#Powershell to delete all files from folder

ForEach ($Item in $ListItems)

{

Remove-PnPListItem -List $ListName -Identity $Item.Id -Recycle -Force

Write-host "Removed File:"$Item.FieldValues.FileRef

}

}

Catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

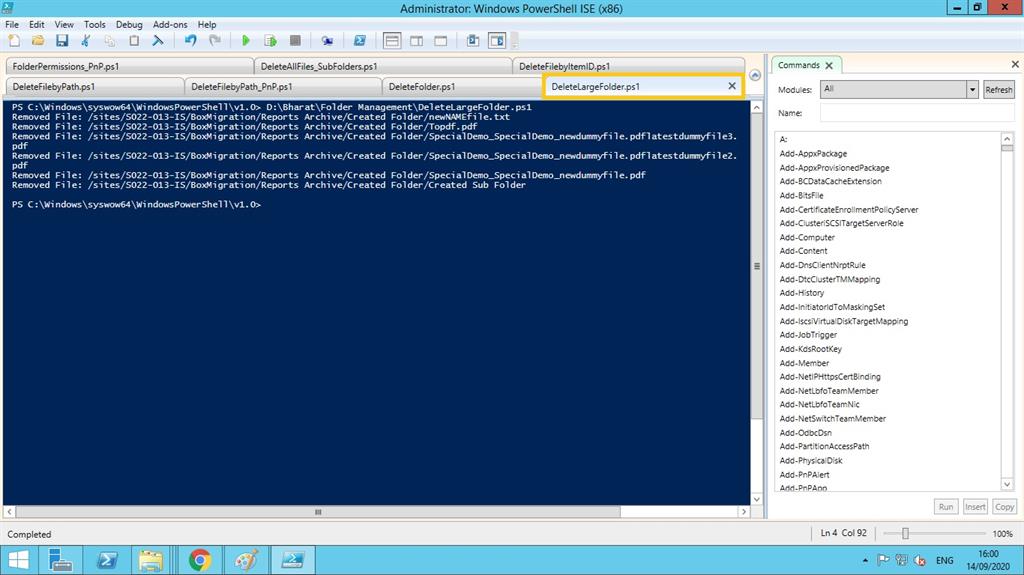
}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

**Prerequisites:**

Above mentioned Folder is also expected to exist before or need to be created separately in order to configure in the above script. Above value 2000 is mentioned for removing/deleting a large library & set it to 1000 for a normal library.

Here is the output after running the above script using your Windows PowerShell ISE tool.



# Delete a File by it’s Path:

Here is a simple way to delete a File by it's path on a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Config Variables

$SiteURL = "################"

$FileRelativeURL ="/sites/S022-013-IS/BoxMigration/Reports Archive/SpecialDemo\_SpecialDemo\_Demo\_dummyfile704.pdf"

Try {

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Try to Get File

$File = Get-PnPFile -Url $FileRelativeURL -ErrorAction SilentlyContinue

If($File)

{

#Delete the File

Remove-PnPFile -ServerRelativeUrl $FileRelativeURL -Force

Write-Host -f Green "File $FileRelativeURL deleted successfully!"

}

Else

{

Write-Host -f Yellow "Could not Find File at $FileRelativeURL"

}

}

catch {

write-host "Error: $($\_.Exception.Message)" -foregroundcolor Red

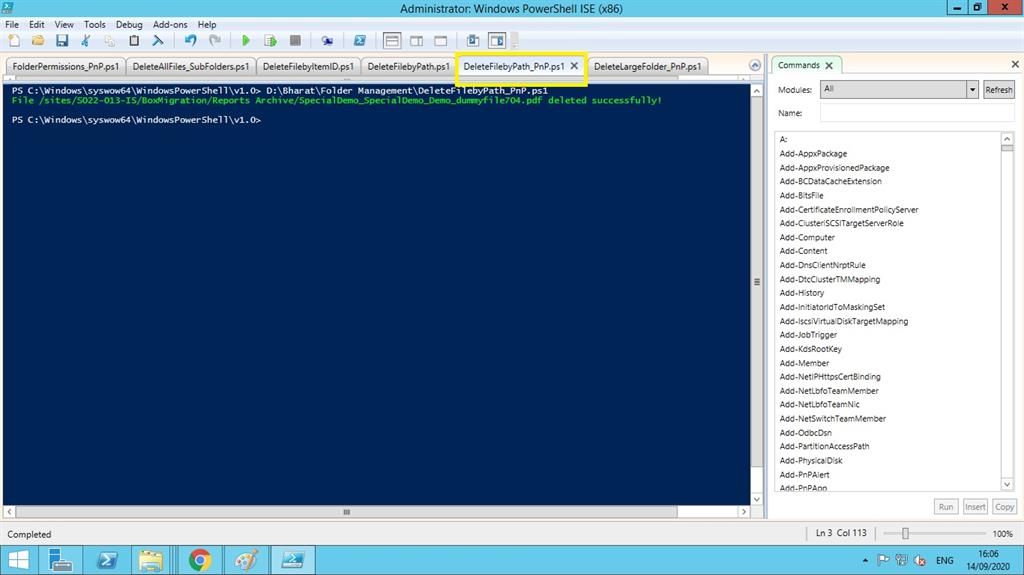
}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

**Prerequisites:**

Above mentioned Folder/File/File Path is also expected to exist before or need to be created separately in order to configure in the above script.

Output: Here is the output after running the above script using your Windows PowerShell ISE tool.



# Assign Multi Folder Permissions via CSV:

Here is a simple way to assign multiple Folder permissions via a CSV file in a given SPO library using PnP PowerShell.

Here's the PnP PowerShell code to execute on a Windows PowerShell.

#Variables

$CSVPath ="D:\Bharat\users.csv"

#Get data from CSV

$CSVData = Import-Csv $CSVPath

#Iterate through each row in CSV

ForEach($Row in $CSVData)

{

Try {

#Connect to SharePoint Online Site

Write-host "Connecting to Site: "$Row.SiteURL

Connect-PnPOnline -Url $Row.SiteURL -UseWebLogin

#Get the group

$Group = Get-PnPGroup -Identity $Row.GroupName

#Add Each user to the Group

Add-PnPUserToGroup -LoginName $Row.UserAccount -Identity $Group

Write-host -f Green "`tAdded User $($Row.UserAccount) to $($Group.Title)"

DisConnect-PnPOnline

}

Catch {

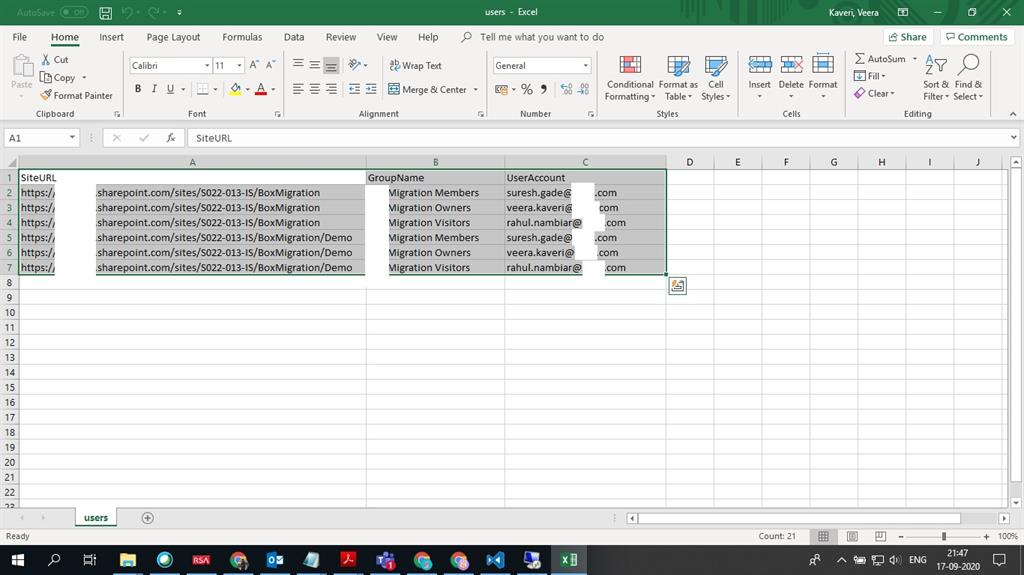
write-host -f Red "Error Adding User to Group:" $\_.Exception.Message

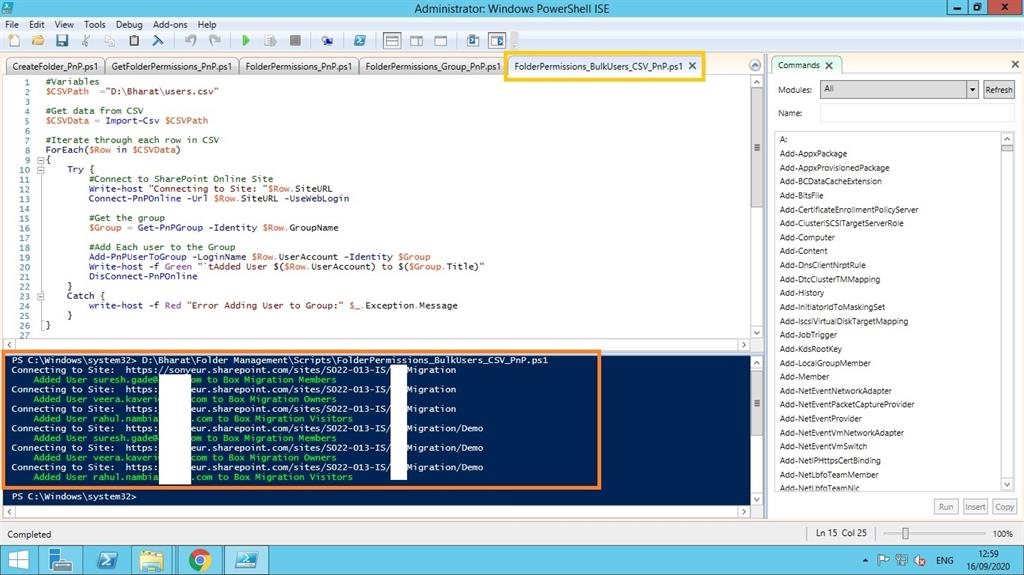
}

}

Please configure above highlighted points to have above requirement implemented via PnP PowerShell programmatically.

Users csv file should look like this:



Here is the snapshot after a successful running of the script:

# Simple Upload:

Let’s try to upload all the files present from your local desktop path to a SharePoint online library.

Here is uploading the PnP script to run. You need to do the configurations accordingly.

$SiteURL = "##############################"

$FilesPath = "C:\Users\7000023594\Dummy Files2"

$SiteRelativePath = "/Reports Archive"

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get All Files from a Local Folder

$Files = Get-ChildItem -Path $FilesPath -Force -Recurse

#Upload All files from the directory

ForEach ($File in $Files)

{

Write-host "Uploading $($File.Directory)\$($File.Name)"

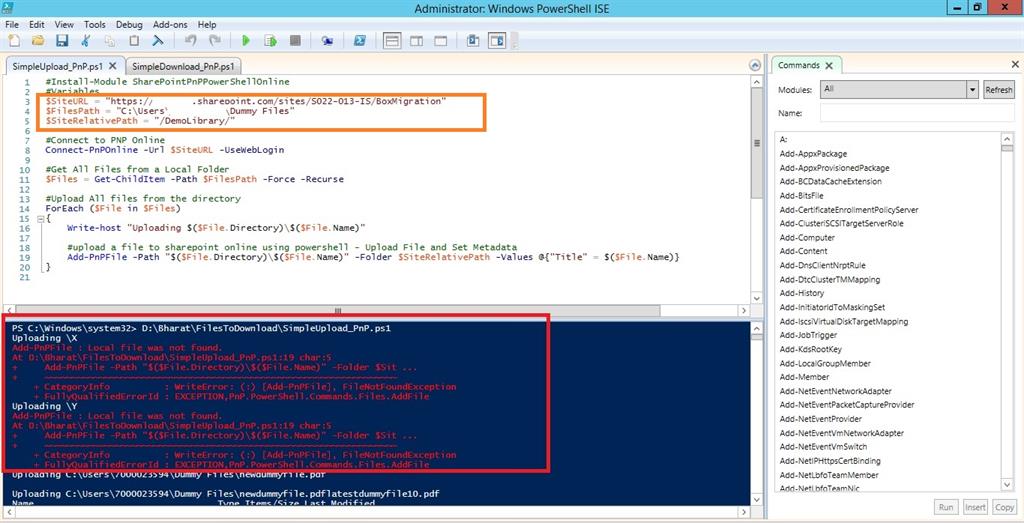
#upload a file to sharepoint online using powershell - Upload File and Set Metadata

Add-PnPFile -Path "$($File.Directory)\$($File.Name)" -Folder $SiteRelativePath -Values @{"Title" = $($File.Name)}

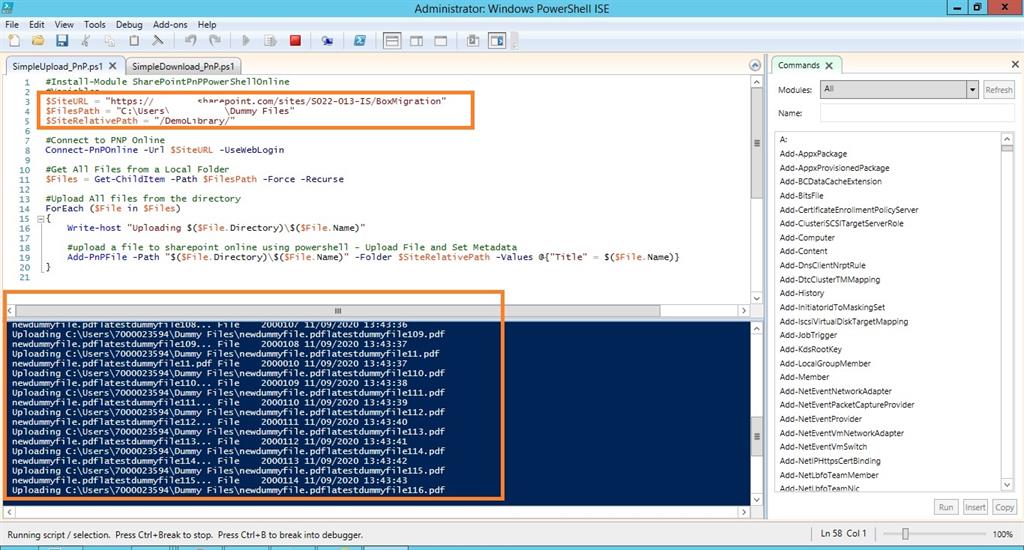
}

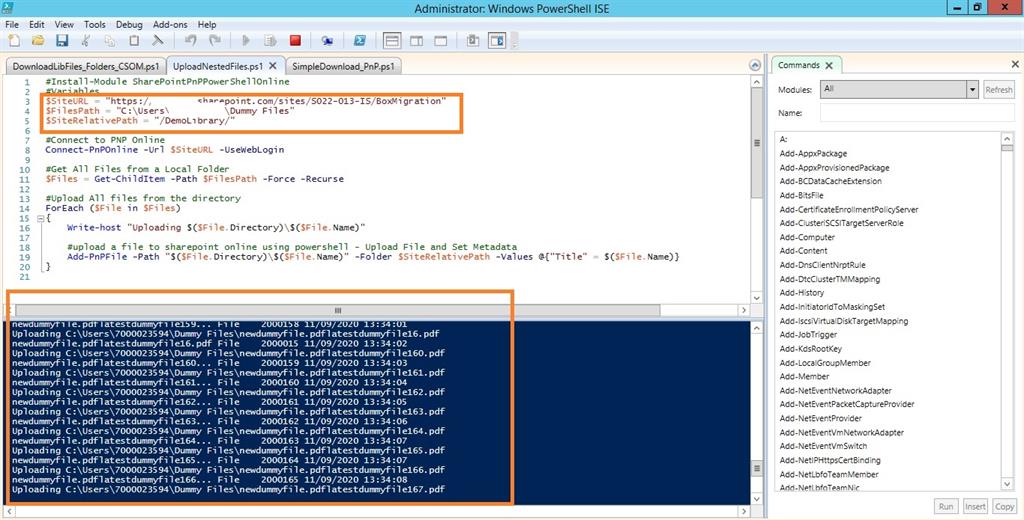
Note

It can only upload files but not folders from the given folder path, as explained in the above script.



Here is the output, as shown below.





***Upload Speed for 1k Dummy Files: 1.86 GB/19 Mins = 0.09 GB/Min***

I would also suggest go for the direct drag and drop modern upload option on any Document Library, which completes faster + includes Folders/Subfolders too!!

Note

The above speeds can vary w.r.t to the LAN/Wi-Fi Connection too!!

We've come to know the above PnP script for our business requirements. It can be programmed to be automated by calling it into another program or via Azure Web Jobs/Azure Functions etc.

# Simple Upload Files into a Folder:

Here is a simple way to upload all the files present in your local desktop path into a SharePoint online Library’s Folder.

It’s the same process as above but with a small change.

$SiteURL = "##############################"

$FilesPath = "C:\Users\7000023594\Dummy Files2"

$SiteRelativePath = "/Reports Archive/Barry"

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL -UseWebLogin

#Get All Files from a Local Folder

$Files = Get-ChildItem -Path $FilesPath -Force -Recurse

#Upload All files from the directory

ForEach ($File in $Files)

{

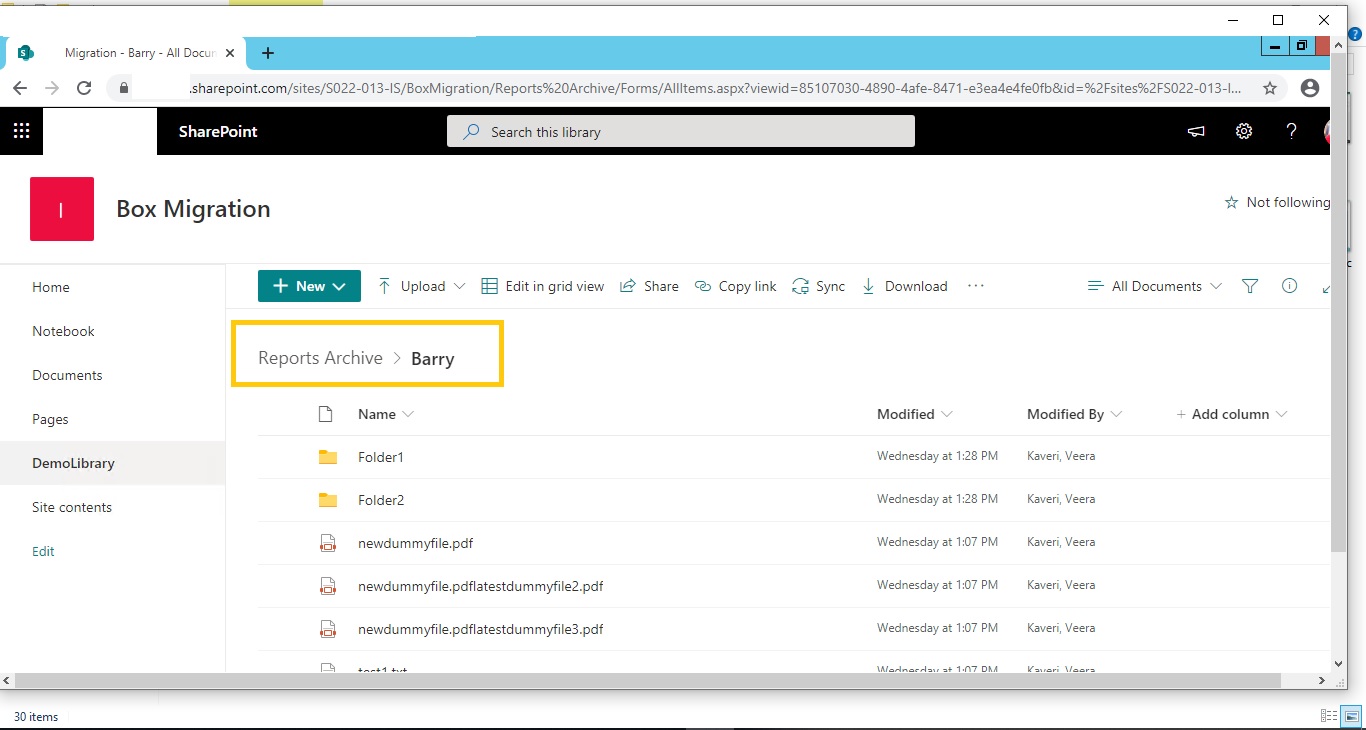
Write-host "Uploading $($File.Directory)\$($File.Name)"

#upload a file to sharepoint online using powershell - Upload File and Set Metadata

Add-PnPFile -Path "$($File.Directory)\$($File.Name)" -Folder $SiteRelativePath -Values @{"Title" = $($File.Name)}

}

Here is the Output reflected after running above script:



# Simple Download:

Here is a simple way to download all the files present into your local desktop path from a given SharePoint online library/folder.

Here is the downloading PnP script to run. You need to do the configurations accordingly.

$SharePointSiteURL = "######################"

# Change this SharePoint Site URL

$SharedDriveFolderPath = "D:\Bharat\DemoLibFiles4"

# Change this Network Folder path

$SharePointFolderPath = "/Reports Archive/Barry"

#$SiteRelativePath = "/Reports Archive/Barry"

#Connecting to SharePoint site

Connect-PnPOnline -Url $SharePointSiteURL -UseWebLogin

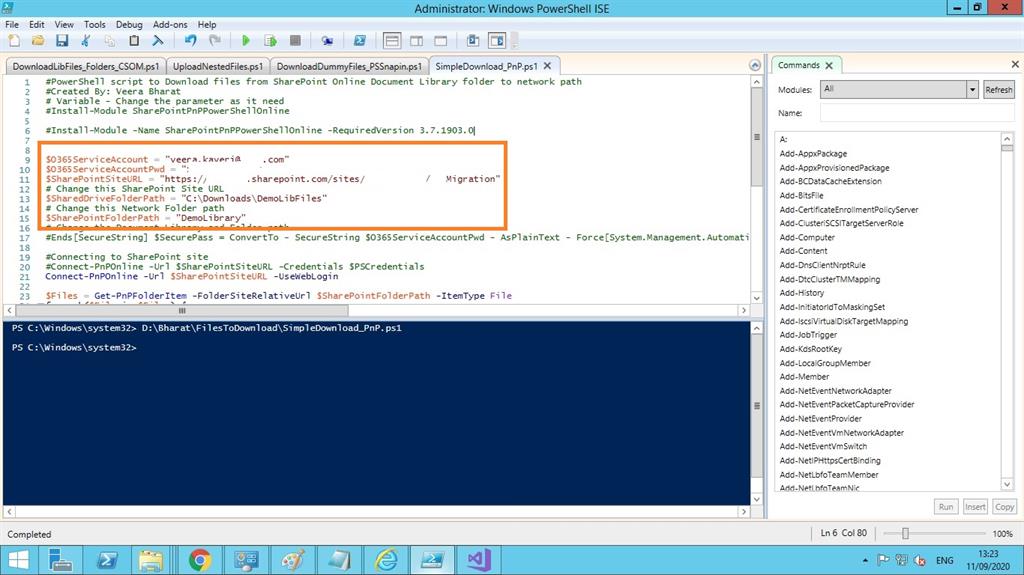
$Files = Get-PnPFolderItem -FolderSiteRelativeUrl $SharePointFolderPath -ItemType File

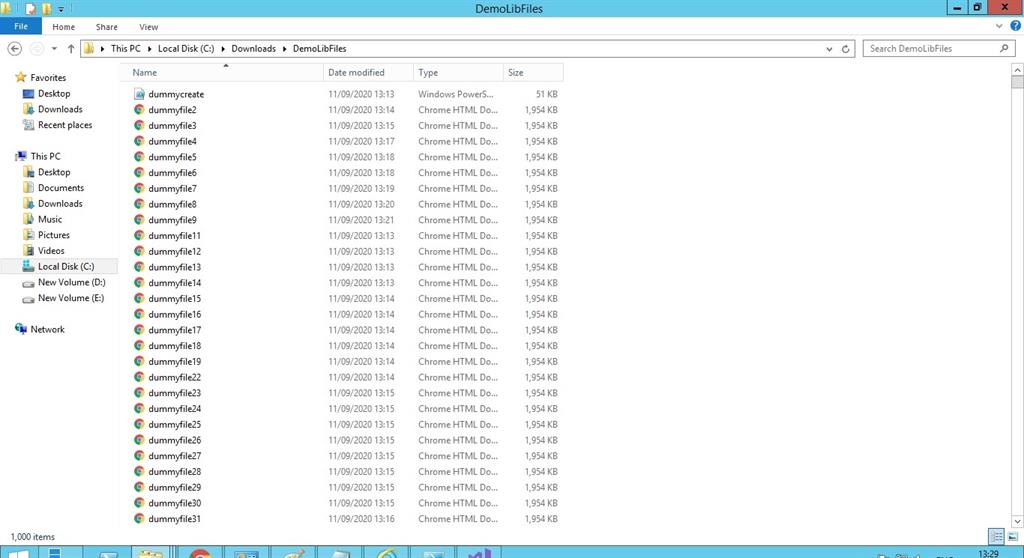
foreach($File in $Files) {

Get-PnPFile -Url $File.ServerRelativeUrl -Path $SharedDriveFolderPath -FileName $File.Name -AsFile

}

Here is the output, as shown below.





***Download Speed for 1k Dummy Files: 1.86 GB/9 Mins = 0.18 GB/Min***

Note

The above speeds can vary with the LAN/Wi-Fi Connection too!!

The above PnP script is useful for our business requirement and can be programmed to be automated by calling it into other programs, or via Azure web jobs/Azure functions, etc.

# External User Authentication:

Here is a simple way to run your PnP Files Uploading script to a Library by hard coding the External User Credentials provided it has been already shared external to that specific SP Site collection.

$FilesPath = "D:\Bharat\DemoLibFiles4"

$SiteRelativePath = "/DemoLibrary/Astros"

$Username = "##################"

$Password = ‘##################’

$SiteCollection = "https://sonyeur.sharepoint.com/sites/S022-013-IS/BoxMigration"

[SecureString]$SecurePass = ConvertTo-SecureString $Password -AsPlainText -Force

[System.Management.Automation.PSCredential]$PSCredentials = New-Object System.Management.Automation.PSCredential($Username, $SecurePass)

Connect-PnPOnline -Url $SiteCollection -Credentials $PSCredentials

#Get All Files from a Local Folder

$Files = Get-ChildItem -Path $FilesPath -Force -Recurse

#Upload All files from the directory

ForEach ($File in $Files)

{

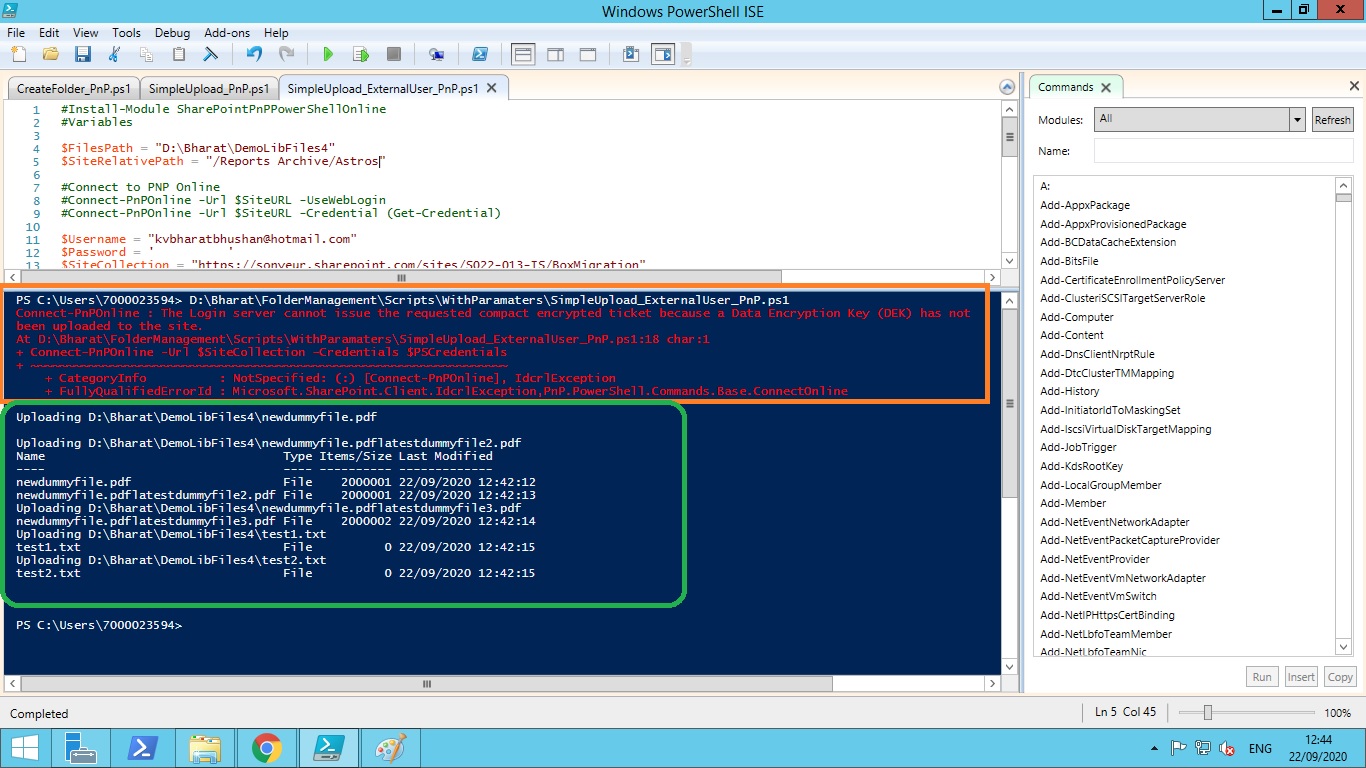
Write-host "Uploading $($File.Directory)\$($File.Name)"

#upload a file to sharepoint online using powershell - Upload File and Set Metadata

Add-PnPFile -Path "$($File.Directory)\$($File.Name)" -Folder $SiteRelativePath -Values @{"Title" = $($File.Name)}

}

Here is the Output after running the above script:



**Important Note:** Above External User Account Login details hard coding will help us to seamlessly run the scripts for External Users provided they are shared access permissions to that specific SP Site Collection. We can also leverage the same concept for Internal User Account that avoids entering the password during the scripts running.

Use only application account details like some **Registered** **Service Accounts.**

# Login Included:

There is a special folder named With **LoginIncluded** inside Scripts folder which has two sample scripts that can be easily run without manual entry of login details as explained in the above section.

# Using CSV:

There is a special folder named **With CSV** inside Scripts folder which has all scripts that can easily run by passing the inputs for bulk items via a CSV file.

This approach can be used to run them on a CMD tool for bulk number of items.

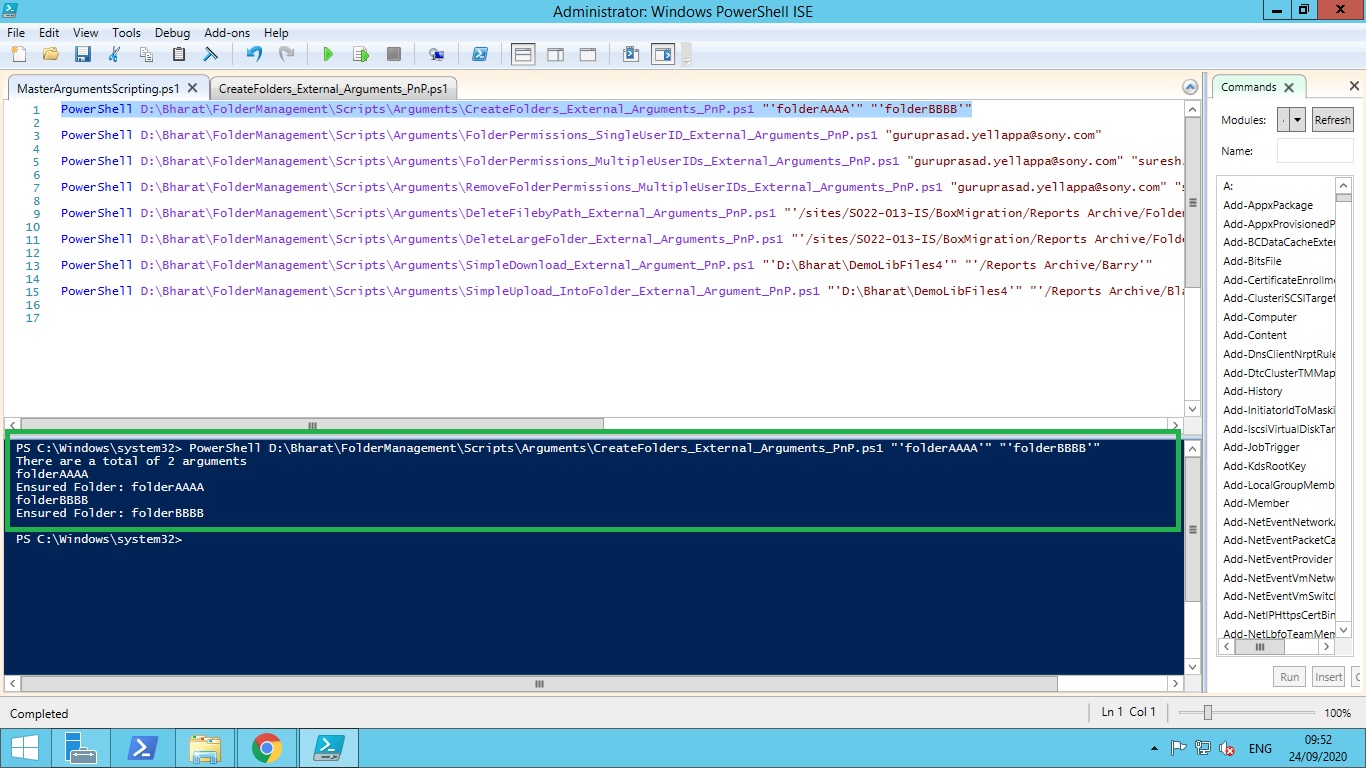
# Using Arguments:

There is a special folder named **Arguments** inside Scripts folder which has all scripts that can be easily run by passing the Arguments externally without configuring the scripts in detail.

This approach can be used to run them on a CMD tool for bulk number of items.

Please refer **MasterArgumentsScripting.ps1** file for a quick template.

Example with related screenshot



# Total Scripts:

Here are the Zipped files of all the above scripts along with CSV files & screenshots for your quick reference and usage.



# Summary:

All the above concepts can be completely utilised for an end Programmer to call/invoke the above PnP scripts accordingly as per their need and also automate them as scheduled jobs in their solutions by proper configuring and development with their respective Programming languages and it’s associated environment.