Basic Operations on SharePoint Using PnP PowerShell Scripts

Nakkeeran Natarajan

Table of Contents

1 SharePoint PnP PowerShell Overview 1

2 Prerequisites 2

3 Tool Used 3

4 SharePoint Site Operations 6

4.1 How to retrieve site collection properties 6

4.2 How to create a sub site 7

4.3 How to retrieve current site properties 8

4.4 How to retrieve sub site properties from root site context 9

4.5 How to update a site or sub site 10

4.6 How to delete a sub site 11

4.7 How to retrieve features (site scoped) from a site collection 12

4.8 How to retrieve feature (site scoped) by feature id from a site collection 13

4.9 How to enable a feature (site scoped) on a site collection 15

4.10 How to disable a feature (site scoped) on a site collection 16

4.11 How to retrieve features (web scoped) from a site or sub sites 17

4.12 How to retrieve feature (web scoped) by feature id from a site or sub site 19

4.13 How to enable a feature (web scoped) on a site or a sub site 20

4.14 How to disable a feature (web scoped) on a site or a sub site 22

5 SharePoint List Operations using PnP PowerShell 24

5.1 How to create a list 24

5.2 How to retrieve all lists from a site 25

5.3 How to retrieve a list using list name 26

5.4 How to update a list 28

5.5 How to delete a list 29

6 SharePoint List Views Tasks using PnP PowerShell 31

6.1 How to create a list view 31

6.2 How to retrieve list views 32

6.3 How to retrieve a list view using view name 34

6.4 How to delete a list view by name 36

7 SharePoint List Item Operations using PnP PowerShell 38

7.1 How to create list item 38

7.2 How to retrieve the list items 40

7.3 How to retrieve the list items using query filter 41

7.4 How to update a list item by ID 42
7.5 How to update a list item by query 44
7.6 How to delete a list item by ID 46
7.7 How to delete a list item by query 47
8 SharePoint Content Type tasks using PnP PowerShell 49
8.1 How to retrieve all content types from a site 49
8.2 How to retrieve site content type by content type name 50
8.3 How to retrieve site content type by content type id 52
8.4 How to retrieve site content types from a list 53
8.5 How to create a site content type 54
8.6 How to create a site content type using base content type 56
8.7 How to delete a content type 57
9 SharePoint Field Tasks using PnP PowerShell 59
9.1 How to create a field on SharePoint site 59
9.2 How to add a field to site content type 60
9.3 How to retrieve all fields from a site 62
9.4 How to retrieve a field using field name 63
9.5 How to remove a field from Content Type 64
9.6 How to delete a field from a site 65
9.7 How to create a field on a list 66
9.8 How to retrieve all fields from a list 68
9.9 How to retrieve a field from a list using field name 70
9.10 How to delete a field from a list 72
10 SharePoint Folder tasks using PnP PowerShell 73
10.1 How to create a folder on a library 73
10.2 How to retrieve folders available on a site 75
10.3 How to retrieve folders from library or folders 76
10.4 How to retrieve a folder from library or folder using folder name 78
10.5 How to delete a folder from a library 79
11 SharePoint Basic File tasks using PnP PowerShell 81
11.1 How to add a file 81
11.2 How to retrieves from library/folder 83
11.3 How to retrieve a file using file name 84
11.4 How to download a file 86

11.5 How to check-out a file 87
11.6 How to check-in a file 88
11.7 How to delete a file 90
12 SharePoint Page Tasks using PnP PowerShell 91
12.1 How to create a page using known page template name 91
12.2 How to create a wiki page using string variable 92
12.3 How to create a wiki page using file content 94
12.4 How to retrieve pages from pages library 96
12.5 How to retrieve wiki page content 98
12.6 How to update a wiki page content using string variable 99
12.7 How to update a wiki page content using file content 101
12.8 How to delete a page 103
12.9 How to delete a wiki page 104
13 SharePoint Group and User Operations using PnP PowerShell 105
13.1 How to create a SharePoint group 105
13.2 How to retrieve all SharePoint groups 107
13.3 How to retrieve a SharePoint group using group title 108
13.4 How to update a SharePoint group 110
13.5 How to delete a SharePoint group 111
13.6 How to add user to a SharePoint User List 112
13.7 How to add user to a SharePoint Group 114
14 Taxonomy Tasks using PnP PowerShell 116
14.1 Import term stores using Text file 116
14.2 Import term stores using XML file 118
14.3 Import term stores using string arrays 120
14.4 Export term set to text file 122
14.5 Export term set to XML 125
15 Office 365 Site Collection tasks using PnP PowerShell 128
15.1 How to create a site collection on tenant site 128
15.2 How to retrieve all site collections from tenant site 130
15.3 How to retrieve a site collection from tenant site using URL 131
15.4 How to update a site collection on tenant site 133
15.5 How to delete a site collection from tenant site 135
16 Summary 137

17 References 137

About Author

Nakkeeran Natarajan is a software developer having nearly 5 years of experience in IT industry. He has been working on SharePoint for more than 4 years. Having several Microsoft certificates like MCTS, MS, and MCPS, he has authored several papers, articles, and blogs. Neakkeeran also is a C-sharp Corner MVP.

Nakkeeran Natarajan

(C# Corner MVP)

1 SharePoint PnP PowerShell Overview

The SharePoint On-premise or SharePoint Online sites can be managed remotely using PnP PowerShell Cmdlets. The operations are actually supported by PnP packages. The PnP PowerShell package basically has Client-Side Object Model (CSOM) core components internally available for the operations. The PnP PowerShell packages can be installed on user machines by downloading the installers from the official website.

The PowerShell scripts are used majorly for admin activities on SharePoint. Most of the one-time activities are carried out using PowerShell. The book is designed for developers/admins who can easily learn and implement the functionalities of PnP PowerShell Script.

PowerShell is object oriented programming language. PowerShell majorly helps in automating tasks. What is PnP? **Patterns and Practices** where Microsoft and external members contribute to the implementation practices for SharePoint. PnP libraries are created to serve the needs of developers for easy learning and implementation of logic through client-side programming. There are various PnP programming methodologies. In this book, I have explained about implementing the PnP programming with PowerShell.

Why we need PnP PowerShell? PnP PowerShell internally implements Client Side Object model for its operations. This, in turn, makes the operations adaptable. The same set of operations can be executed on any SharePoint environment. Using PnP PowerShell, the single line of code is used to access any object on SharePoint, where in with traditional client side or server side object model, multiple lines of code are required to access the objects. The code complexity reduces through this implementation. Also, these operations can be executed from any Windows machines that has Windows PowerShell 3.0.

The PnP Core team has introduced various versions of installers for different versions, like SharePoint 2013 & 2016 On-premises and Office 365 environments.

The PnP Cmdlets are supported on both, SharePoint On-premises and SharePoint Online. The lists of Cmdlets are available on Github. You can refer to the https://github.com/OfficeDev/PnP-PowerShell/blob/master/Documentation/readme.md documentation.

SharePoint Online is part of Office 365 which has/provides powerful features without managing the infrastructure. While SharePoint On-premises gives us the luxury of accessing the infrastructure and SharePoint features, there are comparatively very limited ways of accessing the SharePoint Online data. The SharePoint Online approach should have client-side methods to access the content. So, we can use PnP PowerShell to access the SharePoint

Online objects independent of infrastructure, since client-side packages are internally used on PnP libraries. The same can be adapted/compatible for On-premises environments.

Some of the operations that can be performed on SharePoint, are

- Managing site collections and sub sites
- Managing content types, columns, and lists
- Managing users and groups
- Working with folders and files
- Managing pages
- Working with term stores and taxonomy.

The operations related to each of the above items are explained in detail, in the following sections.

2 Prerequisites

Here, you will see the prerequisites required for executing any of the PnP PowerShell operations.

The following prerequisite needs to be installed on the machine.

- Windows PowerShell 3.0
- Various versions of installers are available on the PnP PowerShell Cmdlets site. The installers are available for SharePoint 2013, SharePoint 2016, and SharePoint Online environments, individually. All three installers can be installed on the machine.

The latest version of the installer can be downloaded from the site (https://github.com/OfficeDev/PnP-PowerShell/releases).

In this book, you will see the operations executed by Windows PowerShell.

The examples explained are compatible for any environment, like SharePoint 2013, SharePoint 2016, and Office 365. I have executed the samples using SharePoint Online site.

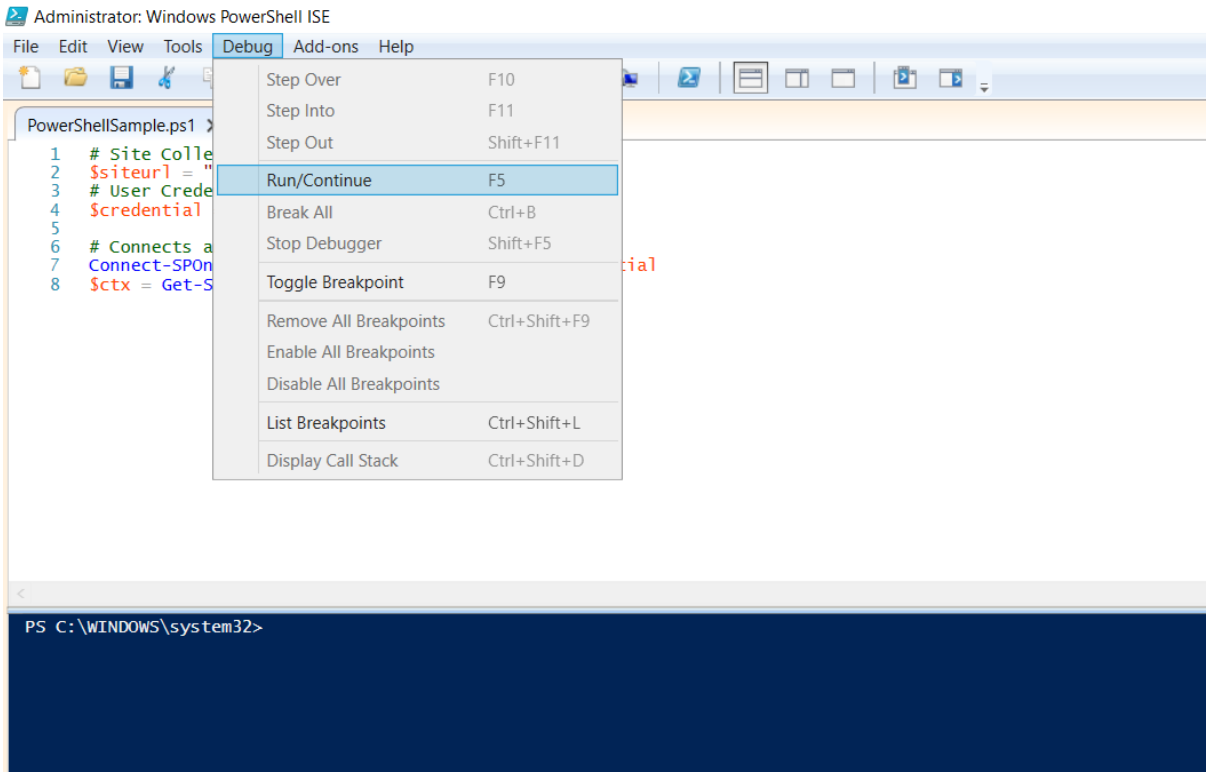
I am not focusing on creating tenant sites for SharePoint Online or creating web applications / site collections for On-premises.

3 Tool Used

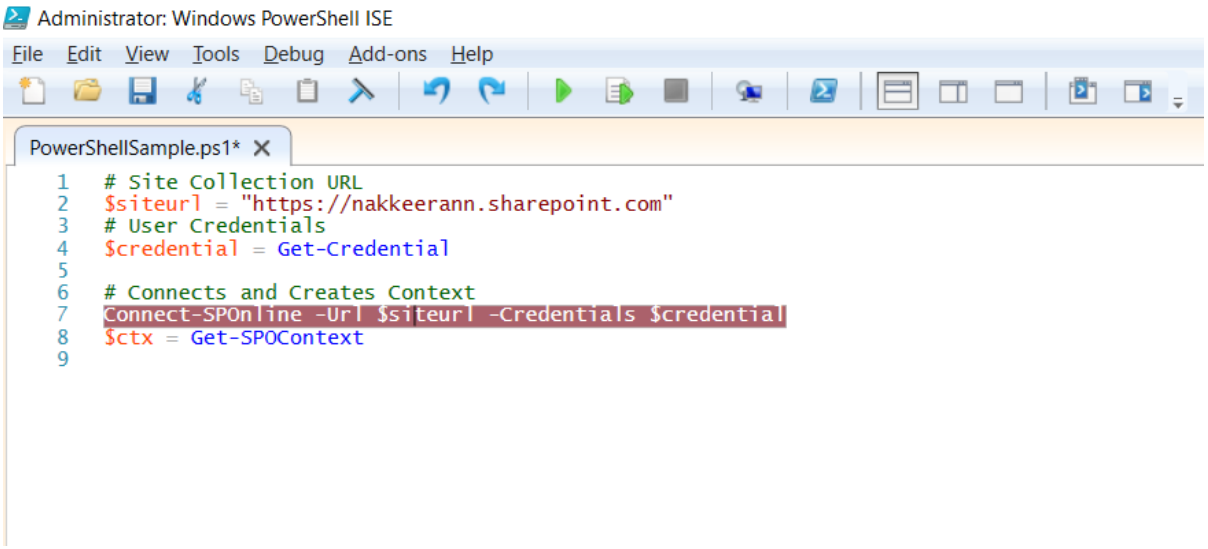
There are multiple ways to create/run the scripts. In this book, I have explained about creating/running the scripts using PowerShell ISE. The PowerShell ISE is available on all the Windows machines which have Windows PowerShell 3.0. The steps explained in the below sections are tested/taken from the development environment. If it's production, you have to create the script files using the below samples and run the same using batch commands.

Running PowerShell: The below operations are executed using PowerShell. The following steps are used to create and run the code.

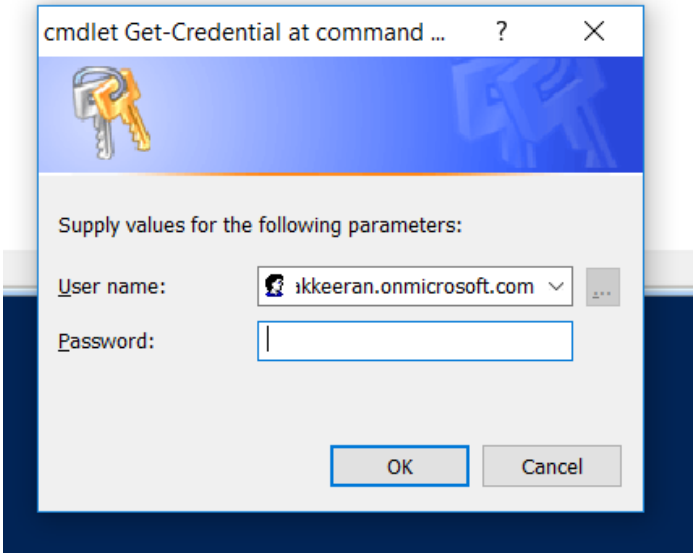
1. Open PowerShell ISE from your machine. (Start -> search for PowerShell ISE -> Right Click and Run as Administrator)
2. Save the file from options or by pressing Ctrl + S.
3. For executing the code, go to Debug option → Run/Continue. Or simply press F5.



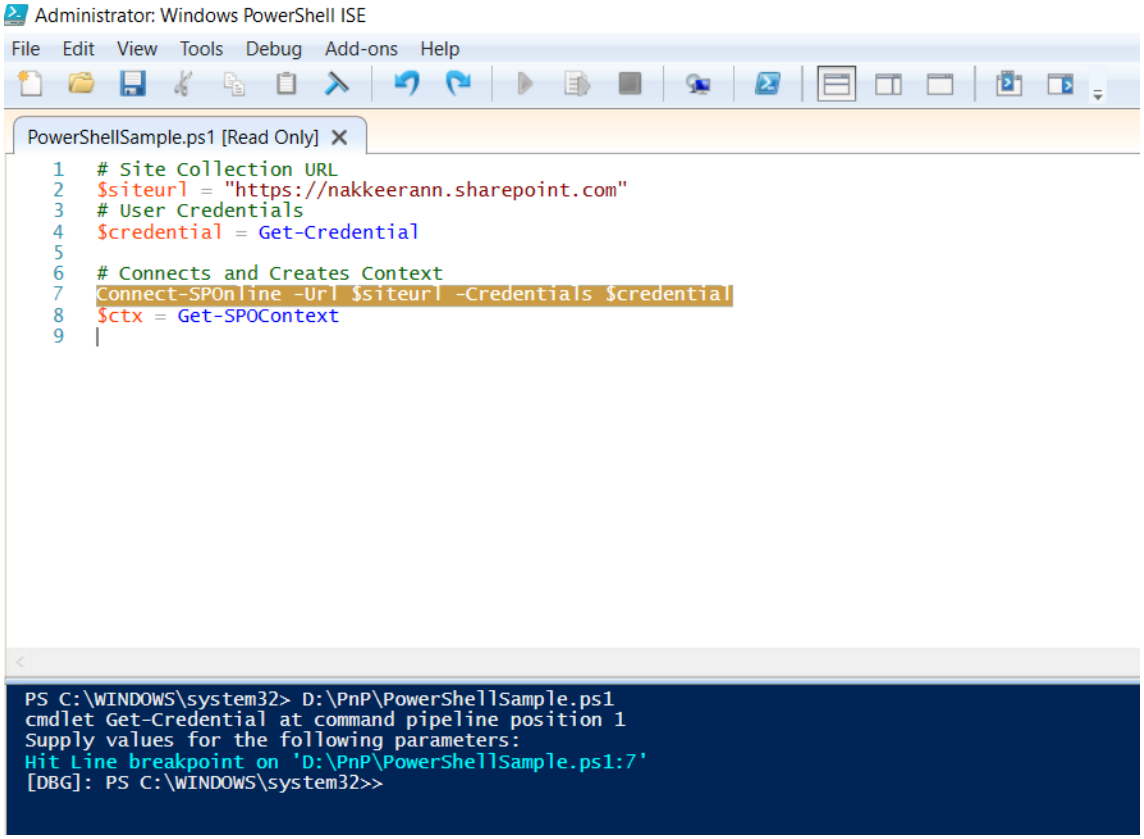
4. For debugging, keep necessary break points on the code. For example, go to line number 7, then go to Debug option → Toggle Break Point. Or just press F9 from the required line.



5. Once executed, user will be prompted for credentials.



6. You can see necessary information/response on the Debugger console.



4 SharePoint Site Operations

In this section, you will learn how to perform create, retrieve, update, and delete sites using PnP PowerShell.

The following cmdlets are used for the operations/examples explained in this section.

- Get-SPOSite
- New-SPOWeb
- Get-SPOWeb
- Set-SPOWeb
- Remove-SPOWeb
- Get-SPOFeature
- Enable-SPOFeature
- Disable-SPOFeature

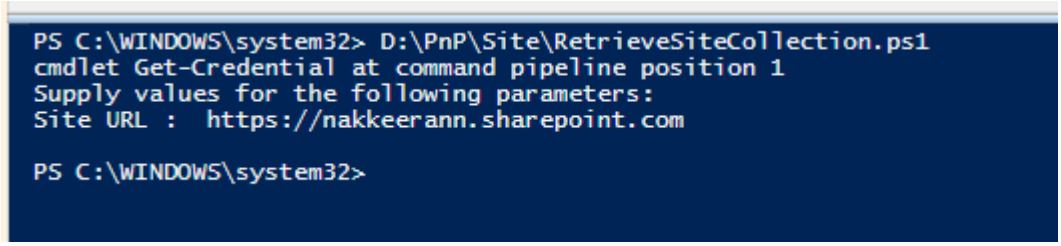
4.1 How to retrieve site collection properties

In this example, you will learn how to retrieve the site collection information using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script, and save the file (D:\PnP\Site\RetrieveSiteCollection.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential
$ctx = Get-SPOContext

# Function
function RetrieveSiteCollection(){
   # Retrieve Site object
   $site = Get-SPOSite
   Write-Host "Site URL : " $site.Url
}

# Calls the function
RetrieveSiteCollection
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.
- The results will be displayed on console, as shown below.



4.2 How to create a sub site

In this example, you will learn how to create a sub site from site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\CreateSubSite.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create sub site on the SharePoint site
function CreateSubSite(){

   # Input Parameters
   $subSiteUrl = "PnPSite"
   $siteTemplate = "STS#0"
   $siteTitle = "PnP Site"
   $localeValue = 1033

   # Create Sub Site
   New-SPOWeb -Url $subSiteUrl -Template $siteTemplate -Title $siteTitle -Locale $locale-
Value
   Write-Host "Sub Site Created"
}

# Calls the function
CreateSubSite # Creates Subsite
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The sub site will be created. You can check the site by pasting the URL (https://nakkeerann.sharepoint.com/PnPSite) on the browser.

4.3 How to retrieve current site properties

In this example, you will learn how to retrieve current site properties using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\CurrentSiteProperties.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve site properties
function RetrieveSite(){
   # Retrieve Web Object
   $web = Get-SPOWeb

   Write-Host "Displays output on the console"
   Write-Host "Site Title: " $web.Title
   Write-Host "Site URL  : " $web.Url
}

# Calls the function
RetrieveSite # Function to retrieve site properties
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Site\CurrentSiteProperties.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Site Title:  Self Team Site
Site URL   :  https://nakkeerann.sharepoint.com
```

The sub site URL can be passed to retrieve sub site properties directly using context.

4.4 How to retrieve sub site properties from root site context

In this example, you will learn how to retrieve sub site properties from root site using PnP PowerShell script.

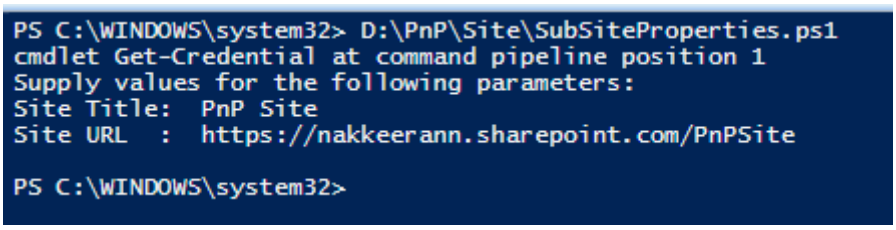
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\SubSiteProperties.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve sub site propertiea
function RetrieveSubSite(){
   # Input Parameters
   $subSiteUrl = "/PnPSite" # Site Relative URL

   # Retrieve web object using sub site relative URL
   $web = Get-SPOWeb -Identity $subSiteUrl

   # Displays output on the console
   Write-Host "Site Title: " $web.Title
   Write-Host "Site URL  : " $web.Url
}

# Calls the Function
RetrieveSubSite # Retrieve Sub site properties
```

- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.



4.5 How to update a site or sub site

In this example, you will learn how to update the site properties using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\UpdateSite.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to update site
function UpdateSite(){
   # Retrieves Web Object
   $web = Get-SPOWeb
   Write-Host "Title before Update: " $web.Title

   $newSiteName = "Nakkeeran's Site"
   # Updates the site title
   Set-SPOWeb -Web $web -Title $newSiteName

   # Retrieves the web
   $web = Get-SPOWeb
   Write-Host "Title After Update: " $web.Title
}

# Calls the Function
UpdateSite # Updates site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Site\UpdateSite.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title before Update:  Self Team Site
Title After Update:  Nakkeeran's Site

PS C:\WINDOWS\system32>
```

To update a sub site, respective sub site's URL should be referenced. The changes can be viewed from the Title, description and logo settings page (https://nakkeerann.sharepoint.com/_layouts/15/prjsetng.aspx).

4.6 How to delete a sub site

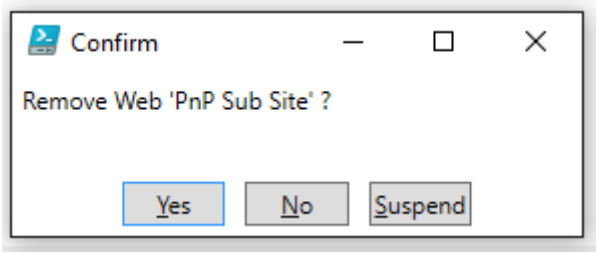
In this example, you will learn how to delete site (web scoped) using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\DeleteSite.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Delete Site Function
function DeleteSite(){
   # Input Parameter
   $siteUrl = "PnPSite" # Site Relative URL of sub site

   # Removes site
   Remove-SPOWeb -Url $siteUrl
}

# Calls the function
DeleteSite # Deletes Site defined
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The confirmation prompt will pop up as show below.



On clicking “yes”, sub site will be deleted.

4.7 How to retrieve features (site scoped) from a site collection

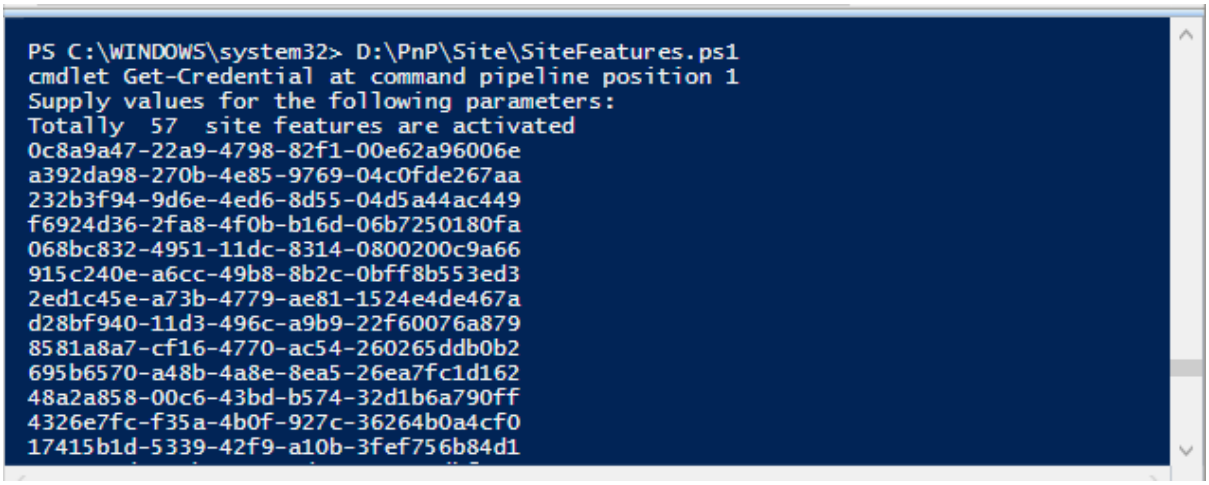
In this example, you will learn how to retrieve the site features of site collection using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\SiteFeatures.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential
$ctx = Get-SPOContext

# Retrieve site features Function
function RetrieveSiteFeatures(){
   # Retrieves Feature by scope
   $siteFeatures = Get-SPOFeature -Scope Site
   Write-Host "Totally " $siteFeatures.Count " site features are activated"

   # Lists all site scoped features
   foreach($siteFeature in $siteFeatures){
      Write-Host $siteFeature.DefinitionId
   }
}

# Calls the function
RetrieveSiteFeatures
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result will be displayed on the console of PowerShell ISE.



4.8 How to retrieve feature (site scoped) by feature id from a site collection

In this example, you will learn how to retrieve particular site collection feature using PnP PowerShell script.

The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required site scoped feature id can be copied from the list. In the following sample, you will learn how to retrieve Document Set feature by ID. The ID of the document set feature is 3bae86a2-776d-499d-9db8-fa4cdc7884f8.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\RetrieveSiteFeature.ps1).

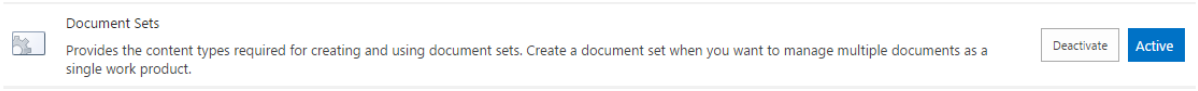
```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Retrieve site feature Function
function RetrieveSiteFeatureById(){
   $featureId = "3bae86a2-776d-499d-9db8-fa4cdc7884f8"

   # Retrieves Document Set Feature by ID
   $siteFeature = Get-SPOFeature -Scope Site -Identity $featureId
   if($siteFeature.DefinitionId -ne $null){
      Write-Host "Feature is active"
   }
   else{
      Write-Host "Couldn't find info about feature id. Feature is not active"
   }
}

# Calls the function
RetrieveSiteFeatureById # Gets site feature details by Id
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result will be displayed on the console of PowerShell ISE.

```
PS C:\WINDOWS\system32> D:\PnP\Site\RetrieveSiteFeature.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Feature is active
```

The feature can be identified on the site collection features page. The site collection features page URL of my site is https://nakkeerann.sharepoint.com/_layouts/15/ManageFeatures.aspx?Scope=Site. The below image shows the active document set feature.



4.9 How to enable a feature (site scoped) on a site collection

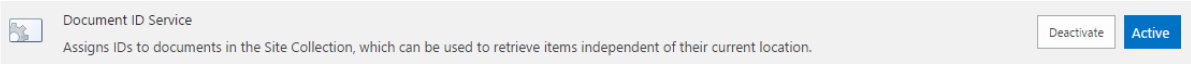
In this example, you will learn how to activate or enable the site collection feature which is not active using PnP PowerShell script. In this example, we will see how to activate the OOB feature which is not active on the site. You will learn how to activate Document ID service feature. The ID of the document ID service feature is b50e3104-6812-424f-a011-cc90e6327318.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\EnableSiteFeature.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Enable site feature Function
function EnableSiteFeature(){
   # Input parameters
   $featureId = "b50e3104-6812-424f-a011-cc90e6327318"

   # Activate Document ID Service feature by ID
   Enable-SPOFeature -Scope Site -Identity $featureId -Force

   # Retrieves and Check if site feature is activated
   $siteFeature = Get-SPOFeature -Scope Site -Identity $featureId
   if($siteFeature.DefinitionId -ne $null){
      Write-Host "Feature is activated"
   }
   else{
      Write-Host "Feature is not activated"
   }
}

# Calls the function
EnableSiteFeature # Enables site feature
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The Document ID service feature is activated. The feature can be identified on the site collection features page. The site collection features page URL of the example site is https://nakkeerann.sharepoint.com/_layouts/15/ManageFeatures.aspx?Scope=Site. The below image shows the document ID service feature which is activated.



The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required site scoped feature id can be copied from the list.

4.10 How to disable a feature (site scoped) on a site collection

In this example, you will learn how to deactivate or disable the feature (site scoped) which is active on the site collection using PnP PowerShell script.

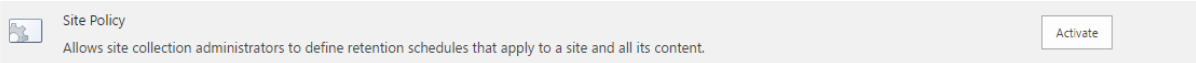
In this example, we will see how to deactivate the OOB feature which is active on the site. The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required site scoped feature id can be copied from the list. You will learn how to deactivate site policy feaure. The ID of the site policy feature is 2fcd5f8a-26b7-4a6a-9755-918566dba90a.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\DisableSiteFeature.ps1).

```
# Site Collection URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Deactivate feature Function
function DeActivateFeature(){

   # Input parameter
   $featureId = "2fcd5f8a-26b7-4a6a-9755-918566dba90a"

   # Deactivates the site feature by id
   Disable-SPOFeature -Identity $featureId -Scope Site
}

# Calls the Function
DeActivateFeature # Deactivates active site feature
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site policy feature is deactivated. The feature can be identified on the site collection features page. The site collection features page URL of the example site is https://nakkeerann.sharepoint.com/_layouts/15/ManageFeatures.aspx?Scope=Site. The below image shows the document ID service feature which is activated.



4.11 How to retrieve features (web scoped) from a site or sub sites

In this example, you will learn how to retrieve the web scoped features from SharePoint site or sub sites using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\ RetrieveWebFeatures.ps1).

```
# Site or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Retrieve Web Scoped Features
function RetrieveWebFeatures(){
   # Retrieves the features
   $webFeatures = Get-SPOFeature -Scope Web
   Write-Host "Totally " $webFeatures.Count " web features are active"

   # Lists all the web scoped features
   foreach($webFeature in $webFeatures){
      Write-Host $webFeature.DefinitionId
   }
}

# Calls the Function
RetrieveWebFeatures # Retrieve web features
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result will be displayed on the console of PowerShell ISE.

```
PS C:\WINDOWS\system32> D:\PnP\Site\RetrieveWebFeatures.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Totally  45  web features are active
00bfea71-d8fe-4fec-8dad-01c19a6e4053
00bfea71-c796-4402-9f2f-0eb9a6e71b18
065c78be-5231-477e-a972-14177cc5b3c7
00bfea71-5932-4f9c-ad71-1557e5751100
0806d127-06e6-447a-980e-2e90b03101b8
00bfea71-4ea5-48d4-a4ad-305cf7030140
2510d73f-7109-4ccc-8a1c-314894deeb3a
d95c97f3-e528-4da2-ae9f-32b3535fbb59
192efa95-e50c-475e-87ab-361cede5dd7f
00bfea71-f600-43f6-a895-40c0de7b0117
a7a2793e-67cd-4dc1-9fd0-43f61581207a
ea23650b-0340-4708-b465-441a41c37af7
0935173a-0ba1-4f08-ae07-447d6d3f7d92
00bfea71-eb8a-40b1-80c7-506be7590102
```

4.12 How to retrieve feature (web scoped) by feature id from a site or sub site

In this example, you will learn how to retrieve particular web scoped feature from a site using PnP PowerShell script.

The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required web scoped feature id can be copied from the list. In the following sample, you will learn how to retrieve Following Content feature by ID. The ID of the Following Content feature is a7a2793e-67cd-4dc1-9fd0-43f61581207a.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\RetrieveWebFeature.ps1).

```
# Site or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com/PnPSite"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Retrieve Web Scoped Feature by ID
function RetrieveWebFeatureById(){

   # Input Parameters
   $featureId = "a7a2793e-67cd-4dc1-9fd0-43f61581207a"

   # Retrieves the feature
   $webFeature = Get-SPOFeature -Identity $featureId -Scope Web

   # Checks if feature is active
   if($webFeature.DefinitionId -ne $null){
      Write-Host "Feature is active"
   }
   else{
      Write-Host "Couldn't find info about feature id. Feature is not active"
   }
}
# Calls the Function
RetrieveWebFeatureById # Retrieve web feature by ID
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result will be displayed on the console of PowerShell ISE.

```
PS C:\WINDOWS\system32> D:\PnP\Site\RetrieveWebFeature.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Feature is active
```

The feature can be identified on the site features page. The features (web scoped) page URL of my site is https://nakkeerann.sharepoint.com/_layouts/15/ManageFeatures.aspx?Scope=Web. The below image shows the active document set feature.

4.13 How to enable a feature (web scoped) on a site or a sub site

In this example, you will learn how to activate or enable the web scoped site feature which is not active using PnP PowerShell script.

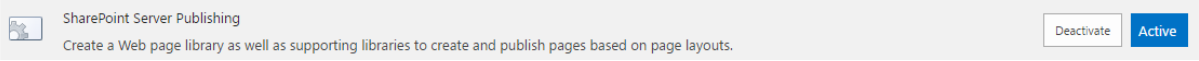
In this example, we will see how to activate the OOB web feature which is not active on the site. The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required web scoped feature id can be copied from the list. You will learn how to activate SharePoint Server Publishing feature on the sub site. The ID of the SharePoint Server Publishing feature is 94c94ca6-b32f-4da9-a9e3-1f3d343d7ecb.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\EnableWebFeature.ps1).

```
# Site Collection URL or site URL or sub site URL
$siteurl = "https://nakkeerann.sharepoint.com/PnPSite" # Sub site URL
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Enable web feature Function
function EnableWebFeature(){

   # Input parameter
   $featureId = "94c94ca6-b32f-4da9-a9e3-1f3d343d7ecb"

   # Activate SharePoint Server Publishing Web feature by ID
   Enable-SPOFeature -Scope Web -Identity $featureId -Force

   # Retrieves and Check if web feature is activated
   $webFeature = Get-SPOFeature -Scope Web -Identity $featureId
   if($webFeature.DefinitionId -ne $null){
      Write-Host "Feature is activated"
   }
   else{
      Write-Host "Feature is not activated"
   }
}

# Calls the function
EnableWebFeature # Enables feature on sub site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint server publishing feature is activated. The feature can be identified on the site features page. The site features page URL of the example site is https://nakkeerann.sharepoint.com/PnPSite/_layouts/15/ManageFeatures.aspx. The below snapshot depicts the feature.



4.14 How to disable a feature (web scoped) on a site or a sub site

In this example, you will learn how to deactivate or disable the web scoped site feature which is not active using PnP PowerShell script.

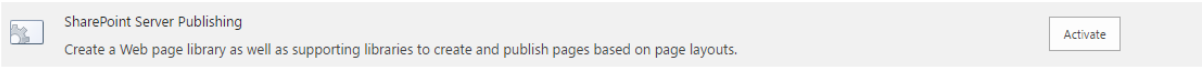
In this example, we will see how to deactivate the OOB web feature which is active on the site. The OOB features list along with feature ID of SharePoint 2013 is available on MSDN blog site. The required web scoped feature id can be copied from the list. You will learn how to deactivate the SharePoint Server Publishing feature on the sub site. The ID of the SharePoint Server Publishing feature is 94c94ca6-b32f-4da9-a9e3-1f3d343d7ecb.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Site\DisableWebFeature.ps1).

```
# Site Collection URL or site URL or sub site URL
$siteurl = "https://nakkeerann.sharepoint.com/PnPSite" # Sub site URL
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Disable web feature Function
function DisableWebFeature(){

   # Input parameters
   $featureId = "94c94ca6-b32f-4da9-a9e3-1f3d343d7ecb"

   # Deactivates SharePoint Server Publishing Web feature by ID
   Disable-SPOFeature -Scope Web -Identity $featureId -Force

   # Retrieves and Check if web feature is activated
   $webFeature = Get-SPOFeature -Scope Web -Identity $featureId
   if($webFeature.DefinitionId -eq $null){
      Write-Host "Feature is deactivated"
   }
   else{
      Write-Host "Feature is still active"
   }
}

# Calls the function
DisableWebFeature # Disables web feature using feature ID
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint server publishing feature is deactivated. The feature can be identified on the site features page. The site features page URL of the example site is https://nakkeerann.sharepoint.com/PnPSite/_layouts/15/ManageFeatures.aspx. The below snapshot depicts the feature.



5 SharePoint List Operations using PnP PowerShell

In this section, you will learn how to work with lists using PnP PowerShell. Some of the basic operations like, create, retrieve, update and delete list operations are explained.

The following commands are used in the examples explained in this section.

- New-SPOList
- Get-SPOList
- Set-SPOList
- Remove-SPOList

5.1 How to create a list

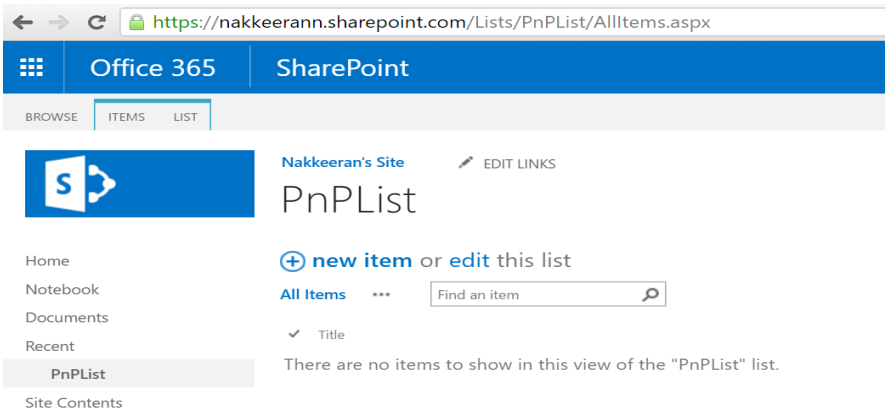
In this example, you will learn how to create a list on SharePoint site using PnP PowerShell script. SharePoint Custom list will be created.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\List\CreateList.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential
$ctx = Get-SPOContext

# Create List Function
function CreateList(){
   # Creates new list
   New-SPOList -Title "PnPList" -Template GenericList -Url "Lists/PnPList"
}

# Calls the function
CreateList # Creates list
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The following snapshot shows the list created on the site.



5.2 How to retrieve all lists from a site

In this example, you will learn how to retrieve all lists from current SharePoint site using PnP PowerShell script.

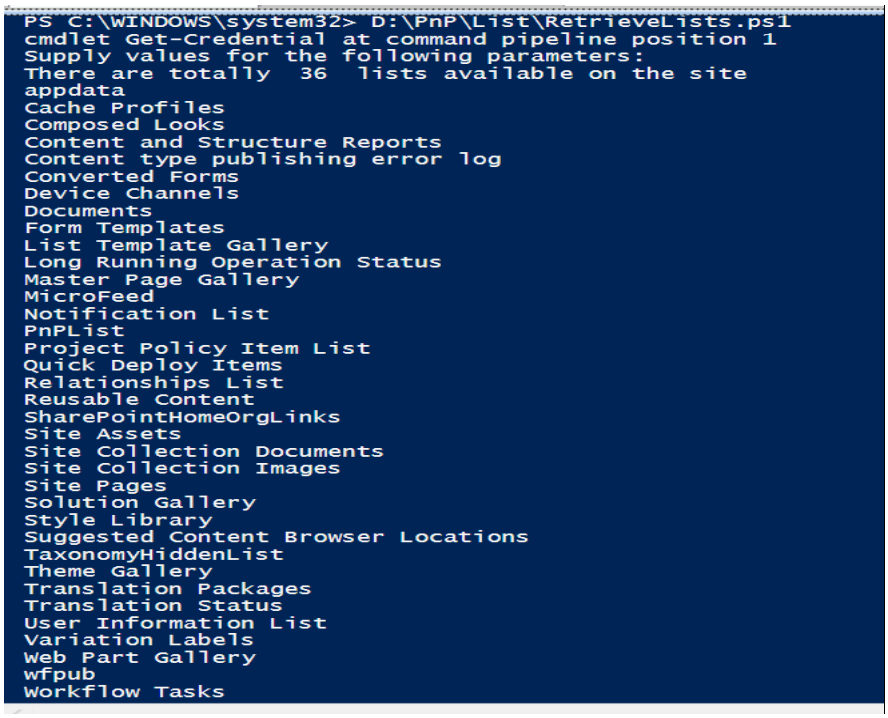
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\List\RetrieveLists.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function
function RetrieveLists(){
   # Gets all available lists from current site
   $lists = Get-SPOList

   Write-Host "There are totally " $lists.count " lists available on the site"
   foreach($list in $lists){
      # Properties for each list
      Write-Host $list.Title
   }
}

# Calls the Function
RetrieveLists #Retrieves all lists from SP site
```

- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.



The result lists down all the SharePoint lists available on the site. Similarly other properties can also be retrieved for the lists by accessing those properties. The following properties can also be retrieved.

- Description
- BaseTemplate
- Created
- Id
- ImageUrl
- ItemCount
- OnQuickLaunch

5.3 How to retrieve a list using list name

In this example, you will learn how to retrieve a particular list from current SharePoint site using PnP PowerShell script. Here, we will see how we can retrieve the list object using list name.

- Open Windows PowerShell ISE as administrator.

- Open a new file, paste the following script and save the file (D:\PnP\List\RetrieveList.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function
function RetrieveList(){

   # Input parameters
   $listName = "PnPList"

   # Gets all available lists from current site
   $list = Get-SPOList $listName

   # Displays on the console
   Write-Host "Title : " $list.Title
   Write-Host "Description : " $list.Description
   Write-Host "BaseTemplate ID : " $list.BaseTemplate
   Write-Host "Created Date : " $list.Created
   Write-Host "List ID : " $list.Id
   Write-Host "ImageUrl : " $list.ImageUrl
   Write-Host "ItemCount : " $list.ItemCount
   Write-Host "OnQuickLaunch : " $list.OnQuickLaunch
}

# Calls the Function
RetrieveList #Retrieves a list from SP site using list name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\List\RetrieveList.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title :  PnPList
Description :
BaseTemplate ID :  100
Created Date :  7/26/2016 5:58:03 PM
List ID :  96c63635-ff6e-438d-82d8-12b17e6be5cb
ImageUrl :  /_layouts/15/images/itgen.png?rev=43
ItemCount :  0
OnQuickLaunch :  False

PS C:\WINDOWS\system32>
```

5.4 How to update a list

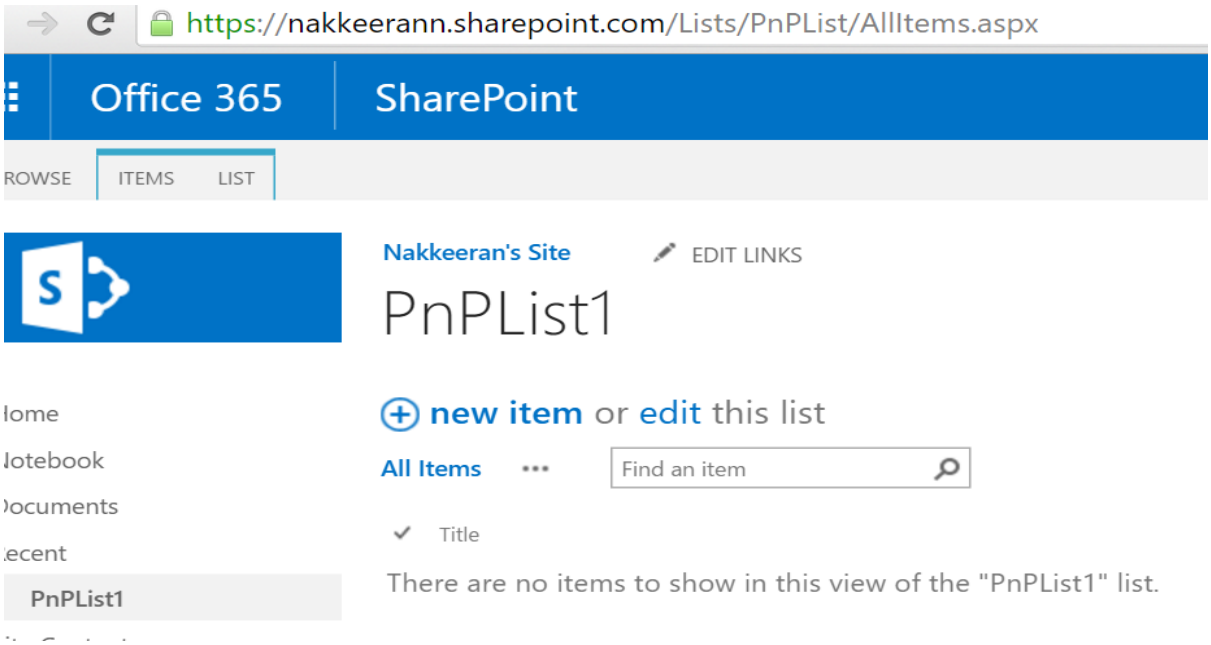
In this example, you will learn how to update a list on the current SharePoint site using PnP PowerShell script. Here, title and permissions are updated for the existing list.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\List\RetrieveList.ps1).

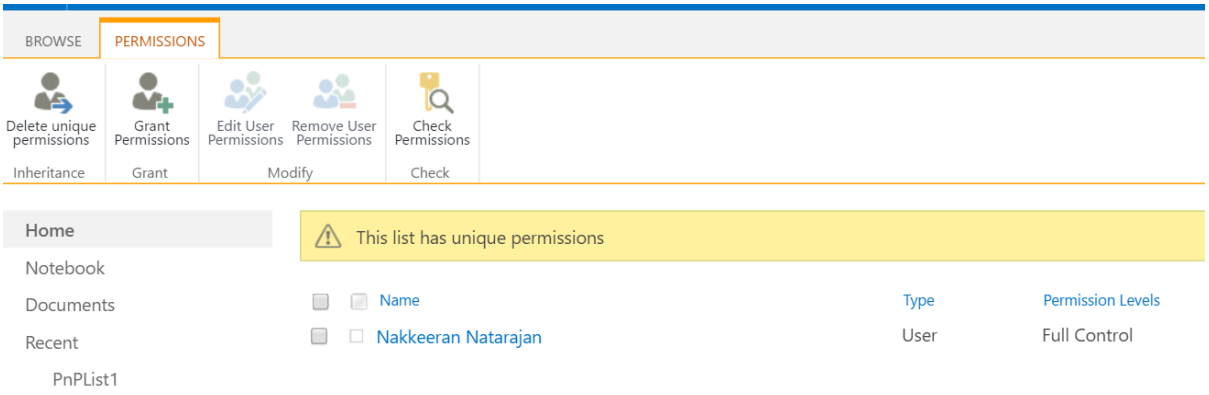
```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update List properties
function UpdateList(){

   # Input parameters
   $oldListName = "PnPList"
   $newListName = "PnPList1"

   # Update the list with new title and sets unique permissions
   Set-SPOList -Identity $oldListName -Title $newListName -BreakRoleInheritance
}

# Calls the Function
UpdateList # Updates list properties
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The updated list snapshot is shown below.



The updated list permissions can be seen below,



5.5 How to delete a list

In this example, you will learn how to delete a list from the current SharePoint site using PnP PowerShell script. To delete a list, list title should be passed as identity value.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\List\RetrieveList.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Delete List
function DeleteList(){

   # Input parameter
   $listName = "PnPList1"

   # Removes list from the site
   Remove-SPOList -Identity $listName -Force
}

# Calls the Function
DeleteList # Deletes List with list name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The list will be deleted from the site.

6 SharePoint List Views Tasks using PnP PowerShell

In this section, you will learn how to work with views using PnP PowerShell. Some of the basic operations like, create, retrieve, update and delete view operations are explained.

The following commands are used in the examples explained in this section.

- Add-SPOView
- Get-SPOView
- Set-SPOView
- Remove-SPOView

6.1 How to create a list view

In this example, you will learn how to create a list view for a list on SharePoint site using PnP PowerShell script. SharePoint Custom list views will be created.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\View\CreateListView.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Create List View Function
function CreateListView(){

   # Input Parameters
   $listName = "PnPList"
   $viewName = "PnPView"
   $fields = @("ID", "Title") # fields to be included in the view

   # Creates new list view
   Add-SPOView -List $listName -Title $viewName -ViewType None -Fields $fields
}

# Calls the function
CreateListView # Creates list view
```

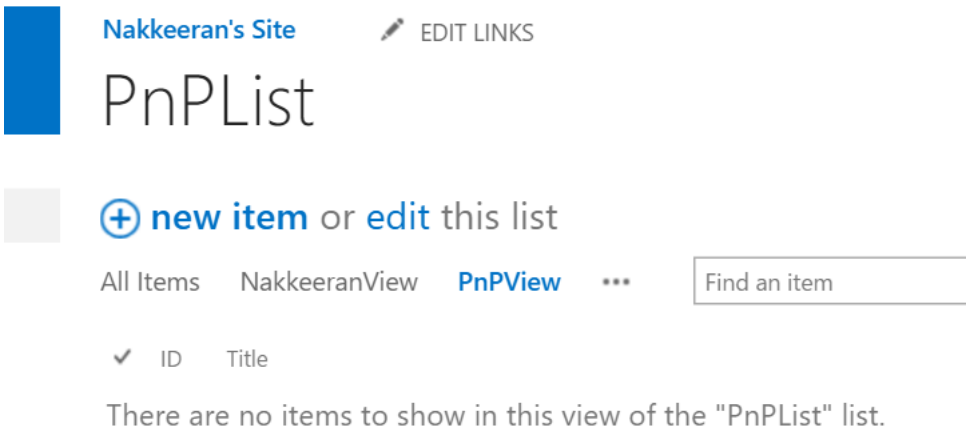
- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The following snapshot shows the executed operation.

```
PS C:\WINDOWS\system32> D:\PnP\View\CreateListView.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Title                          Id                                   DefaultView  PersonalView
-----                          --                                   -----------  ------------
PnPView                        a0e299a2-c800-4983-86f9-f9655e33f008 False        False


PS C:\WINDOWS\system32>
```

The following snapshot shows the list view created for a SharePoint list on the site.



6.2 How to retrieve list views

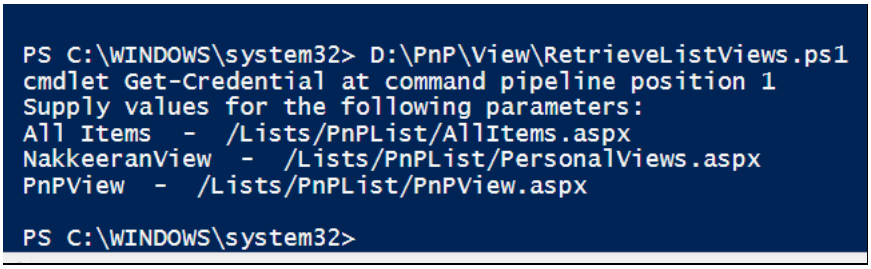
In this example, you will learn how to retrieve all views available for a list from current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\View\RetrieveListViews.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve list views
function RetrieveListViews(){

   # Input parameter
   $listName = "PnPList"

   # Gets all available views for a list from current site
   $views = Get-SPOView -List $listName

   # Displays the response on the console
   foreach($view in $views){
      Write-Host $view.Title " - " $view.ServerRelativeUrl
   }
}

# Calls the Function
RetrieveListViews #Retrieves all views for a list from SP site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.



The result lists down all the list views available for a SharePoint list on the site. Similarly other properties can also be retrieved for the lists by accessing those properties. The following properties can also be retrieved.

- Title
- DefaultView: Boolean shows whether the view is default view or not
- Id: View Id
- JsLink: Existing JS links
- ListViewXml: View XML

- Paged: Is view paged?
- PersonalView: Personal view
- RowLimit: View row limit
- ServerRelativeUrl: View URL
- ViewFields: Fields of the view
- Base view ID
- Content Type ID
- Schema XML

6.3 How to retrieve a list view using view name

In this example, you will learn how to retrieve a particular list view from a SharePoint list on current SharePoint site using PnP PowerShell script. Here, we will see how we can retrieve the list object using list name.

- Open Windows PowerShell ISE as administrator.

- Open a new file, paste the following script and save the file (D:\PnP\View\RetrieveListView.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve list views by title(name)
function RetrieveListViewByTitle(){

   # Input Parameters
   $listName = "PnPList" # List Name
   $viewName = "PnPView" # View Name

   # Gets a list view for a list from current site
   $view = Get-SPOView -List $listName -Identity $viewName

   # Displays results on the console
   Write-Host $view.Title
   Write-Host $view.DefaultView
   Write-Host $view.BaseViewId
   Write-Host $view.Id
   Write-Host $view.Hidden
   Write-Host $view.HtmlSchemaXml
   Write-Host $view.JSLink
   Write-Host $view.ListViewXml
   Write-Host $view.MobileDefaultView
   Write-Host $view.OrderedView
   Write-Host $view.Paged
   Write-Host $view.PersonalView
   Write-Host $view.RowLimit
   Write-Host $view.ServerRelativeUrl
   Write-Host $view.ViewFields
   Write-Host $view.ViewQuery
}

# Calls the Function
RetrieveListViewByTitle #Retrieves a view by title for a list from SP site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\View\RetrieveListView.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Write-Host PnPView
False
1
a0e299a2-c800-4983-86f9-f9655e33f008
False
<View Name="{A0E299A2-C800-4983-86F9-F9655E33F008}" Type="HTML" DisplayName="PnPView" Url="/Lists/PnPList/PnPView.aspx" Level
="1" BaseViewID="1" ContentTypeID="0x" ImageUrl="/_layouts/15/images/generic.png?rev=43"><ViewFields><FieldRef Name="ID" /><F
ieldRef Name="Title" /></ViewFields><Query /><RowLimit Paged="FALSE">30</RowLimit><XslLink Default="TRUE">main.xsl</XslLink><
JSLink>clienttemplates.js</JSLink><Toolbar Type="Standard" /><ParameterBindings><ParameterBinding Name="NoAnnouncements" Loca
tion="Resource(wss,noXinviewofY_LIST)" /><ParameterBinding Name="NoAnnouncementsHowTo" Location="Resource(wss,noXinviewofY_DE
FAULT)" /></ParameterBindings></View>
clienttemplates.js
<View Name="{A0E299A2-C800-4983-86F9-F9655E33F008}" Type="HTML" DisplayName="PnPView" Url="/Lists/PnPList/PnPView.aspx" Level
="1" BaseViewID="1" ContentTypeID="0x" ImageUrl="/_layouts/15/images/generic.png?rev=43" ><Query /><ViewFields><FieldRef Name
="ID" /><FieldRef Name="Title" /></ViewFields><RowLimit Paged="FALSE">30</RowLimit><JSLink>clienttemplates.js</JSLink><XslLin
k Default="TRUE">main.xsl</XslLink><Toolbar Type="Standard"/></View>
False
False
False
False
30
/Lists/PnPList/PnPView.aspx
ID Title
```

6.4 How to delete a list view by name

In this example, you will learn how to delete a list view for a particular list from the current SharePoint site using PnP PowerShell script. To delete a list view, list view title should be passed as identity value.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\View\DeleteListView.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to delete list view
function DeleteView(){

   # Input Parameters
   $listName = "PnPList" # List Name
   $viewName = "PnPView" # View Name

   # Deletes a list view for a list from current site
   Remove-SPOView -List $listName -Identity $viewName -Force
}

# Calls the Function
DeleteView #Deletes a view by title for a list from SP site
```

- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The list view will be deleted for a list from the site.

7 SharePoint List Item Operations using PnP PowerShell

In this section, you will learn how to work with list items using PnP PowerShell. Some of the basic operations like, create, retrieve, update and delete list item operations are explained.

The following commands are used in the examples explained in this section.

- Add-SPOListItem
- Get-SPOListItem
- Set-SPOListItem
- Remove-SPOListItem

7.1 How to create list item

In this example, you will learn how to create item on a list on SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\CreateListItem.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Add List Item Function
function AddListItem(){

   # Input Parameter
   $listName = "PnPList"

   # Creates item with empty fields
   Add-SPOListItem -List $listName
}

# Calls the function
AddListItem # Adds List item
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

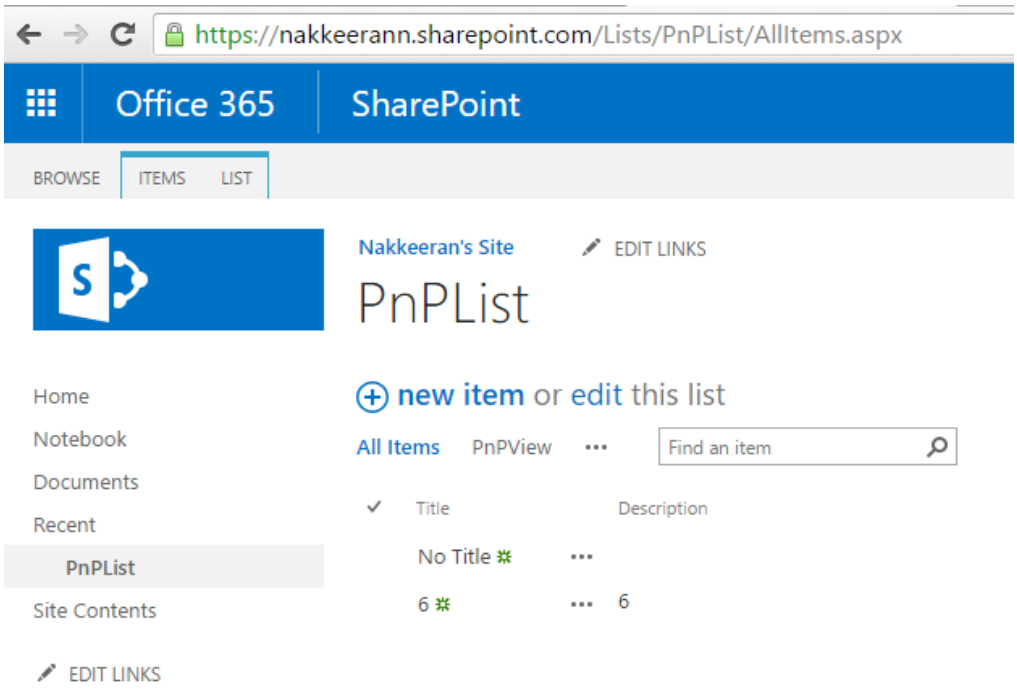
The following snapshot shows the created item on the PowerShell.

```
PS C:\WINDOWS\system32> D:\PnP\ListItem\CreateListItem.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Id    Title                                              Unique Id
--    -----                                              ---------
1                                                        071c600a-b908-4c31-bdcb-54774085f8cf


PS C:\WINDOWS\system32>
```

The following code snippet shows adding an item using item values.

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Add List Item Function
function AddListItemWithValues(){

   # Input Parameters
   $listName = "PnPList"
   $itemValues = @{"Title"="6";"Description"="6"} # Item Values

   # Creates item with values
   Add-SPOListItem -List $listName -Values $itemValues
}

# Calls the function
AddListItemWithValues # Adds List item
```

The following snapshot shows the list along with items created.



7.2 How to retrieve the list items

In this example, you will learn how to retrieve all item values from a list on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\RetrieveListItems.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get all Items
function RetrieveListItems(){

   # Input Parameters
   $listName = "PnPList"
   $fields = "Id","Title"

   # Retrieves items
   $listItems = Get-SPOListItem -List $listName -Fields $fields

   # Write output on the console
   foreach($listItem in $listItems){
      Write-Host "Id : " $listItem["ID"] "Title : " $listItem["Title"]
   }
}
# Calls the Function
RetrieveListItems # Get Items
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\ListItem\RetrieveListItems.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Id :  1 Title :
Id :  2 Title :  6

PS C:\WINDOWS\system32>
```

Similarly other field values can also be retrieved.

7.3 How to retrieve the list items using query filter

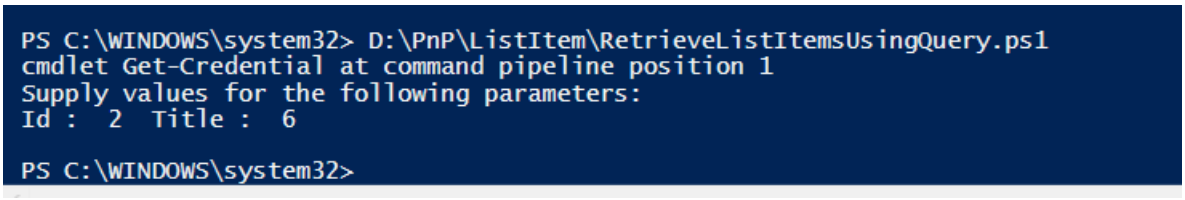
In this example, you will learn how to retrieve required items from a list by building a query on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script, and save the file (D:\PnP\ListItem\ RetrieveListItemsUsingQuery.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get required Items
function RetrieveListItemByQuery(){

   # Input Parameters
   $query = "<View><Query><Where><Eq><FieldRef Name='Title'/><Value
Type='Text'>6</Value></Eq></Where></Query></View>" # Item Query
   $listName = "PnPList" # List Name

   # Retrieves items
   $listItem = Get-SPOListItem -List $listName -Query $query

   # Displays results on console
   Write-Host "Id : " $listItem["ID"] " Title : " $listItem["Title"]
}
# Calls the Function
RetrieveListItemByQuery # Retrieve items by query
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.



7.4 How to update a list item by ID

In this example, you will learn how to update an item present on a list, by ID, on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\ UpdateListItem.ps1).

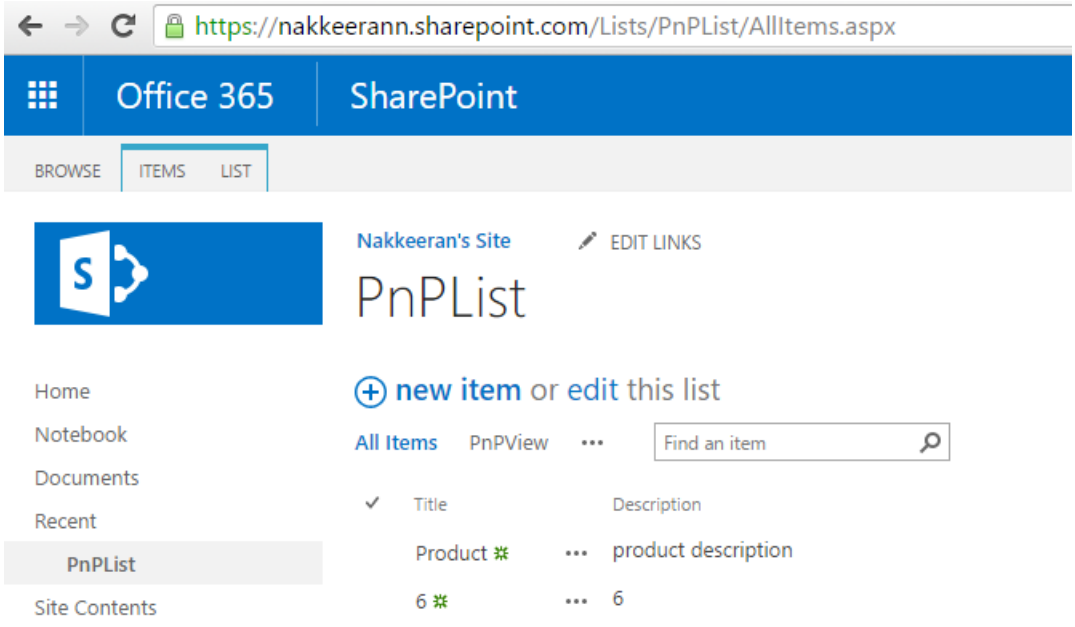
```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update Items
function UpdateListItem(){

   # Input Parameters
   # Item ID
   $itemId = 1
   # Field values Array
   $fieldValues = @{"Title"="Product";"Description"="product description"}
   # List Name
   $listName = "PnPList"

   # Update item using id
   Set-SPOListItem -List $listName -Identity $itemId -Values $fieldValues
}
# Calls the Function
UpdateListItem # Update item by identity
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown.

```
PS C:\WINDOWS\system32> D:\PnP\ListItem\UpdateListItem.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Id    Title                                          Unique Id
--    -----                                          ---------
1     Product                                        071c600a-b908-4c31-bdcb-54774085f8cf


PS C:\WINDOWS\system32>
```

The following snapshot shows the updated list item.



7.5 How to update a list item by query

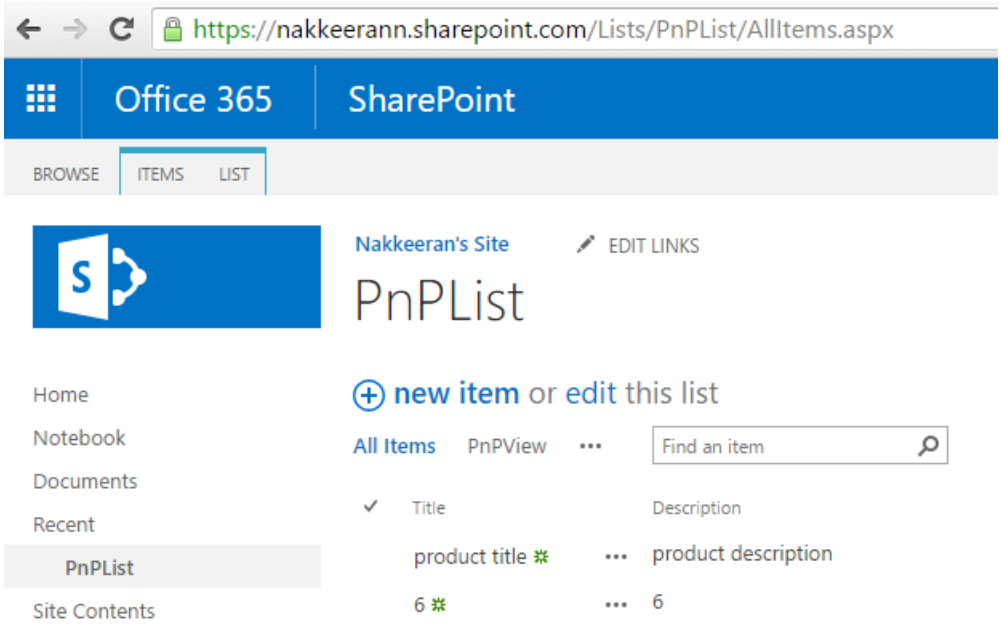
In this example, you will learn how to update item present on a list by using query on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\ UpdateListItemUsingQuery.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update Items using Query
function UpdateListItemByQuery(){

   ## Input Parameters
   # List Name
   $listName = "PnPList"
   # Query to filter the list using item ID
   $query = "<View><Query><Where><Eq><FieldRef Name='Title'/><Value
Type='Text'>product</Value></Eq></Where></Query></View>"

   # Get respective item using query
   $item = Get-SPOListItem -List $listName -Query $query
   if($item -ne $null){
      # Field values Array
      $fieldValues = @{"Title"="product title";"Description"="product description"}

      # Update item by item object retrieved above
      Set-SPOListItem -List $listName -Identity $item -Values $fieldValues
   }
}

# Calls the Function
UpdateListItemByQuery # Update item by query
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The below snapshot shows the list with updated item.



7.6 How to delete a list item by ID

In this example, you will learn how to delete an item present on a list by using ID, on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\ RemoveListItem.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update Items using Query
function DeleteListItem(){

   ## Input Parameters
   # List Name
   $listName = "PnPList"
   # Item Id
   $itemId = 1

   # Removes list item using id
   Remove-SPOListItem -List $listName -Identity $itemId -Force
}

# Calls the Function
DeleteListItem # Delete item by identity
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The item is deleted from the list.

7.7 How to delete a list item by query

In this example, you will learn how to delete an item present on a list by using query, on current SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ListItem\ RemoveListItemUsingQuery.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update Items using Query
function DeleteListItemByQuery(){

   ## Input Parameters
   # List Name
   $listName = "PnPList"
   # Query to Retrieve list Item
   $query = "<View><Query><Where><Eq><FieldRef Name='Title'/><Value
Type='Text'>6</Value></Eq></Where></Query></View>"

   # Retrieve item
   $item = Get-SPOListItem -List $listName -Query $query
   if($item -ne $null){
      # Removes the item from list by referring the item retrieved above
      Remove-SPOListItem -List $listName -Identity $item -Force
   }
}

# Calls the Function
DeleteListItemByQuery # Delete item by query
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The item is deleted from the list with the help of query.

8 SharePoint Content Type tasks using PnP PowerShell

In this section, you will learn how to complete basic content type operations using PnP PowerShell. Some of the basic operations, like create, retrieve, and delete content type operations are explained.

The following commands, used in the examples, are explained in this section.

- Get-SPOContentType
- Add-SPOContentType
- RemoveSPOContentType

8.1 How to retrieve all content types from a site

In this example, you will learn how to retrieve the content types from SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ContentType\ RetrieveContentTypes.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get all Items
function RetrieveContentTypes(){

   # Retrieves Content Types
   $contentTypes = Get-SPOContentType

   Write-Host "Totally " $contentTypes.Count " content types on the site"
   # Write output on the console
   foreach($contentType in $contentTypes){
      Write-Host $contentType.Name
   }
}

# Calls the Function
RetrieveContentTypes # Retrieves all the content types from site level
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below. It shows the cropped image.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\RetrieveContentTypes.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Totally  108  content types on the site
System
Common Indicator Columns
Fixed Value based Status Indicator
SharePoint List based Status Indicator
Excel based Status Indicator
SQL Server Analysis Services based Status Indicator
Item
Circulation
New Word
Category
Site Membership
Community Member
WorkflowServiceDefinition
Reusable HTML
Health Analyzer Rule Definition
Published Link
Resource
Reusable Text
Official Notice
Phone Call Memo
Project Policy
Page Output Cache
Device Channel
Holiday
What's New Notification
WorkflowServiceSubscription
```

Similarly, other field values can also be retrieved. Some of the properties that can be retrieved are -

- ID
- Description
- Group
- Schema XML
- Scope

8.2 How to retrieve site content type by content type name

In this section, you will learn how to retrieve the required site content type from SharePoint site using PnP PowerShell script. The Item Site content type is retrieved in the following example.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script, and save the file (D:\PnP\ContentType\ RetrieveContentType.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get all Items
function RetrieveContentTypeByName(){

   # Set Content Type Name
   $contentTypeName = "Item"

   # Retrieves Content Type
   $contentType = Get-SPOContentType -Identity $contentTypeName

   # Displays the results on console
   Write-Host "ID          :" $contentType.ID
   Write-Host "Description :" $contentType.Description
   Write-Host "Group       :" $contentType.Group
   Write-Host "Scope       :" $contentType.Scope
   Write-Host "SchemaXML   :" $contentType.SchemaXML
}

# Calls the Function
RetrieveContentTypeByName # Retrieves a content type from site level
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\RetrieveContentType.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
ID          : 0x01
Description : Create a new list item.
Group       : List Content Types
Scope       : /
SchemaXML   : <ContentType ID="0x01" Name="Item" Group="List Content Types" Description="Create a new list item." Version="0" FeatureId="{695b6570-a48b-
4a8e-8ea5-26ea7fc1d162}"><Folder TargetName="_cts/Item" /><Fields><Field ID="{c042a256-787d-4a6f-8a8a-cf6ab767f12d}" Name="ContentType" SourceID="http:/
/schemas.microsoft.com/sharepoint/v3" StaticName="ContentType" Group="_Hidden" Type="Computed" DisplayName="Content Type" Sealed="TRUE" Sortable="FALSE"
 RenderXMLUsingPattern="TRUE" PITarget="MicrosoftWindowsSharePointServices" PIAttribute="ContentTypeID" Customization=""><FieldRefs><FieldRef ID="{03e45
e84-1992-4d42-9116-26f756012634}" Name="ContentTypeId" /></FieldRefs><DisplayPattern><MapToContentType><Column Name="ContentTypeId" /></MapToContentType
></DisplayPattern></Field><Field ID="{fa564e0f-0c70-4ab9-b863-0177e6ddd247}" Name="Title" SourceID="http://schemas.microsoft.com/sharepoint/v3" StaticNa
me="Title" Group="_Hidden" Type="Text" DisplayName="Title" Required="TRUE" FromBaseType="TRUE" Customization="" ShowInNewForm="TRUE" ShowInEditForm="TRU
E"></Field></Fields><XmlDocuments><XmlDocument NamespaceURI="http://schemas.microsoft.com/sharepoint/v3/contenttype/forms"><FormTemplates xmlns="http://
schemas.microsoft.com/sharepoint/v3/contenttype/forms"><Display>ListForm</Display><Edit>ListForm</Edit><New>ListForm</New></FormTemplates></XmlDocument>
</XmlDocuments></ContentType>

PS C:\WINDOWS\system32>
```

Similarly, other field values can also be retrieved.

8.3 How to retrieve site content type by content type id

In this section, you will learn how to retrieve the required site content type with the help of content type ID, from SharePoint site using PnP PowerShell script. The Item content type is retrieved in the following example. The ID of item content type is 0x01. The above example gets content type id from content type name and here it is vice versa. Get-SPOContentType command is used for retrieving the information along with ID parameter.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script, and save the file (D:\PnP\ContentType\ RetrieveContentTypeByID.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get Content Type by ID
function RetrieveContentTypeByID(){
   # Set Content Type Name
   $contentTypeId = "0x01"
   # Retrieves Content Type
   $contentType = Get-SPOContentType -Identity $contentTypeId

   # Displays the results on console
   Write-Host "Name       :" $contentType.Name
   Write-Host "Description :" $contentType.Description
   Write-Host "Group      :" $contentType.Group
   Write-Host "Scope      :" $contentType.Scope
   Write-Host "SchemaXML   :" $contentType.SchemaXML
}

# Calls the Function
RetrieveContentTypeByID # Retrieves a content type from site level
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\RetrieveContentTypeByID.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Name        : Item
Description : Create a new list item.
Group       : List Content Types
Scope       : /
SchemaXML   : <ContentType ID="0x01" Name="Item" Group="List Content Types" Description="Create a new list item." Version="0" FeatureId="{695b6570-a48b-
4a8e-8ea5-26ea7fc1d162}"><Folder TargetName="_cts/Item" /><Fields><Field ID="{c042a256-787d-4a6f-8a8a-cf6ab767f12d}" Name="ContentType" SourceID="http:/
/schemas.microsoft.com/sharepoint/v3" StaticName="ContentType" Group="_Hidden" Type="Computed" DisplayName="Content Type" Sealed="TRUE" Sortable="FALSE"
 RenderXMLUsingPattern="TRUE" PITarget="MicrosoftWindowsSharePointServices" PIAttribute="ContentTypeID" Customization=""><FieldRefs><FieldRef ID="{03e45
e84-1992-4d42-9116-26f756012634}" Name="ContentTypeId" /></FieldRefs><DisplayPattern><MapToContentType><Column Name="ContentTypeId" /></MapToContentType
></DisplayPattern></Field><Field ID="{fa564e0f-0c70-4ab9-b863-0177e6ddd247}" Name="Title" SourceID="http://schemas.microsoft.com/sharepoint/v3" StaticNa
me="Title" Group="_Hidden" Type="Text" DisplayName="Title" Required="TRUE" FromBaseType="TRUE" Customization="" ShowInNewForm="TRUE" ShowInEditForm="TRU
E"></Field></Fields><XmlDocuments><XmlDocument NamespaceURI="http://schemas.microsoft.com/sharepoint/v3/contenttype/forms"><FormTemplates xmlns="http://
schemas.microsoft.com/sharepoint/v3/contenttype/forms"><Display>ListForm</Display><Edit>ListForm</Edit><New>ListForm</New></FormTemplates></XmlDocument>
</XmlDocuments></ContentType>

PS C:\WINDOWS\system32>
```

Similarly, other content type values can also be retrieved.

8.4 How to retrieve site content types from a list

In this section, you will learn how to retrieve the content types available for a list from SharePoint site using PnP PowerShell script. Get-SPOContentType command is used for retrieving the information along with list parameter.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ContentType\ RetrieveCTsFromList.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get Content Types from List
function RetrieveCTsFromList(){
   # Set List Name
   $listName = "PnPList"
   # Retrieves Content Types from list
   $contentTypes = Get-SPOContentType -List $listName

   # Displays the results on console
   Write-Host "Totally " $contentTypes.Count " content types available on this list"
   foreach($contentType in $contentTypes){
      Write-Host $contentType.Name
   }
}

# Calls the Function
RetrieveCTsFromList # Retrieves site content types from list
```

- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\RetrieveCTsFromList.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Totally  2  content types available on this list
Item
Folder
```

The above snapshot shows the default content types of a list. Remember, we have not added or removed any content types to/from the list.

8.5 How to create a site content type

In this section, you will learn how to create a new site content type on SharePoint site using PnP PowerShell script. The default base content used for creating new content type is item. The site content types can be created on site collections or sub sites. The site URL decides the content type scope. The scope can be either root site collection or sub sites.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ContentType\ CreateSiteContentType.ps1).

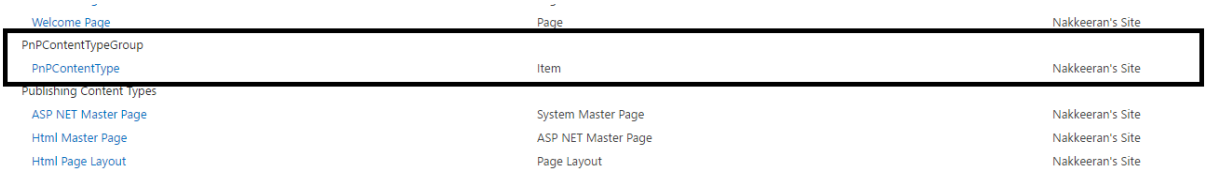
```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function To Create New Content Type
function CreateContentType(){
   # Set Content Type Name
   $contentTypeName = "PnPContentType"
   # Set Description
   $contentTypeDescription = "PnP Content Type"
   # Set Group Name
   $contentTypeGroup = "PnPContentTypeGroup"

   # Creates New Content Type Using above inputs
   Add-SPOContentType -Name $contentTypeName -Description $contentTypeDescription -
Group $contentTypeGroup
}
# Calls the Function
CreateContentType # Create New Content Type with name, description
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\CreateSiteContentType.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Name                           Id                                 Group                          Description
----                           --                                 -----                          -----------
PnPContentType                 0x01004BA6B38572A7A74A99915...     PnPContentTypeGroup            PnP Content Type
```

The following snapshot shows the content type created on the site. The content types can be found on site content type's page (https://nakkeerann.sharepoint.com/_layouts/15/mngctype.aspx).



8.6 How to create a site content type using base content type

In this section, you will learn how to create a new site content type based on the base content type on SharePoint site using PnP PowerShell script. The default base content type used for creating new content type is item. In this example, let us take consider page content type as base content type. Basically new content type will inherit the base content type. The site content types can be created on site collections or sub sites. The site URL decides the content type scope. The scope can be either root site collection or sub sites. Add-SPOContentType is used to create a content type along with other parameters.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ContentType\ CreateSiteContentTypeUsingCT.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function To Create New Content Type Using Parent Content
function CreateContentTypeUsingCT(){
   # Set Base Content Type Name
   $baseCTName = 'Page'
   # Retrieve Base Content Type (Page)
   $baseCT = Get-SPOContentType -Identity $baseCTName

   # Set Content Type Name
   $contentTypeName = "PnPPageContentType"
   # Set Description
   $contentTypeDescription = "PnP Page Content Type"
   # Set Group Name
   $contentTypeGroup = "PnPContentTypeGroup"

   # Creates New Content Type Using above inputs and base content type retrieved
   Add-SPOContentType -Name $contentTypeName -Description $contentTypeDescription -
Group $contentTypeGroup -ParentContentType $baseCT
}
# Calls the Function
CreateContentTypeUsingCT # Create New Content Type with name, description using parent
content type
```

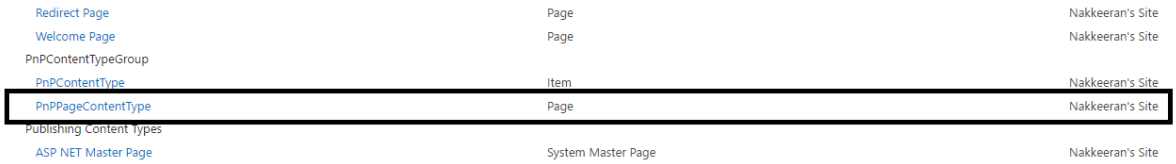
- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\ContentType\CreateSiteContentTypeUsingCT.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Name                           Id                             Group                    Description
----                           --                             -----                    -----------
PnPPageContentType             0x010100C568DB52D9D0A14D9B2... PnPContentTypeGroup      PnP Page Content Type


PS C:\WINDOWS\system32>
```

The following snapshot shows the content type created on the site. The content types can be found on site content type's page (https://nakkeerann.sharepoint.com/_layouts/15/mngctype.aspx).



8.7 How to delete a content type

In this section, you will learn how to delete a content type form the SharePoint site using PnP PowerShell script. The site URL decides the content type scope. The scope can be either root site collection or sub sites. Remove-SPOContentType command is used to delete a content type by identity parameter.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\ContentType\ RemoveSiteContentType.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function To Create New Content Type Using Parent Content
function RemoveContentType(){
   # Input - Set Content Type Name
   $contentTypeName = "PnPContentType"

   # Removes content type from the site using name
   Remove-SPOContentType -Identity $contentTypeName –Force
}

# Calls the Function
RemoveContentType # Delete the content type from site using name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The content type will be deleted from the site.

9 SharePoint Field Tasks using PnP PowerShell

In this topic, you will learn how to perform basic field operations like add, retrieve or delete to/from sites/content types using PnP PowerShell script. Fields are otherwise called as columns.

The Cmdlets used for the below operations are,

- Get-SPOField
- Add-SPOField
- Add-SPOFieldToContentType
- Remove-SPOFieldFromContentType

9.1 How to create a field on SharePoint site

In this section, you will learn how to create a site column (field) on SharePoint site using PnP PowerShell script. Add-SPOField is used to create a site column along with display Name, internal name, group name and type parameters.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\CreateField.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create a Field (site column)
function CreateField(){

   # Input Parameters
   $fieldName = "PnPSiteColumn"
   $fieldGroup = "PnPGroup"

   # Creates Field
   Add-SPOField -DisplayName $fieldName -InternalName $fieldName -Group $fieldGroup -
Type Text
}

# Calls the Function
CreateField # Creates a field (site column) on the current SharePoint site
```

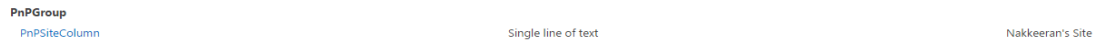
- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Field\CreateField.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Title         InternalName  Id
-----         ------------  --
PnPSiteColumn PnPSiteColumn f3034a32-dc12-427a-b7b7-921a93b666e3
```

The created field can be viewed on the site columns page (https://nakkeerann.sharepoint.com/_layouts/15/mngfield.aspx). The below snapshot shows the field created on site column page.



9.2 How to add a field to site content type

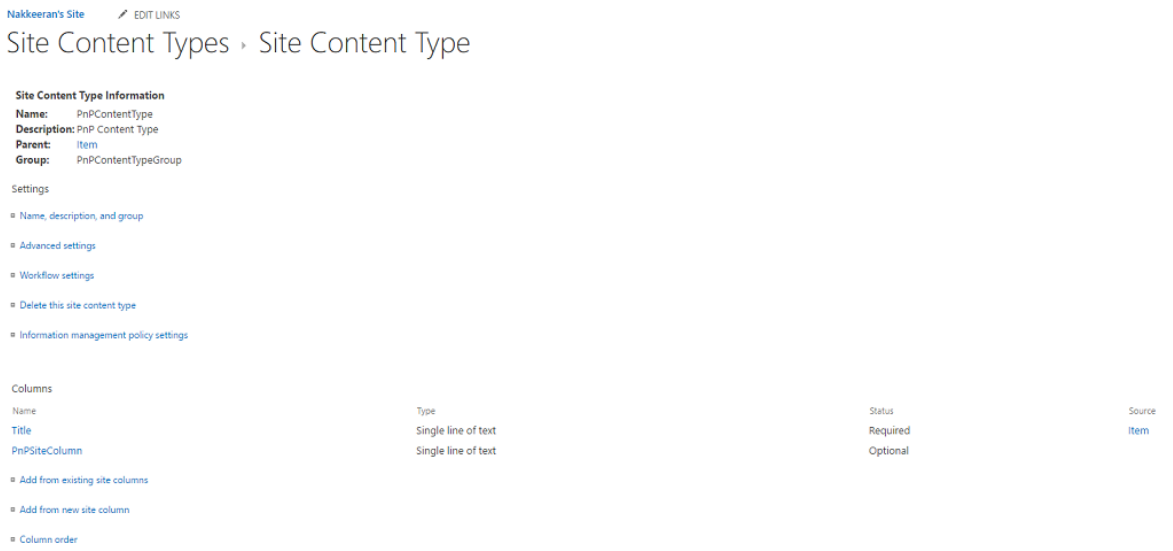
In this section, you will learn how to create a site column (field) on SharePoint site using PnP PowerShell script. Add-SPOFieldToContentType command is used to add fields to site content type. The existing site column is added to existing site content type in the following example.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\AddFieldToCT.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to add field (site column) to content type
function AddFieldToCT(){

   # Input Parameters
   $fieldName = "PnPSiteColumn"
   $contentTypeName = "PnPContentType"

   # Adds existing field to content type
   Add-SPOFieldToContentType -Field $fieldName -ContentType $contentTypeName
}

# Calls the Function
AddFieldToCT # Adds an existing field (site column) to existing content type on the current SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The result can be viewed by opening the "PnPContentType" content type page. The URL for content type's page is https://nakkeerann.sharepoint.com/_layouts/15/mngctype.aspx. The below snapshot shows the field added to the content type.



9.3 How to retrieve all fields from a site

In this section, you will learn how to retrieve all the fields (columns) from SharePoint site using PnP PowerShell script.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RetrieveFields.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get all Field and their titles
function RetrieveFields(){
   # Retrieves Fields or Site Columns
   $fields = Get-SPOField

   Write-Host "Totally " $fields.Count " fields available on the site"
   # Write output on the console
   foreach($field in $fields){
      Write-Host $field.Title
   }
}

# Calls the Function
RetrieveFields # Retrieves all the fields from site level
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below. It shows the cropped image.

```
PS C:\WINDOWS\system32> D:\PnP\Field\RetrieveFields.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Totally  716  fields available on the site
Lower values are better
Comments Lookup
Other Address Country/Region
Instance Id
Categories
Enterprise Keywords
Alias
Manager's Name
Total File Stream Size
Number of Ratings
Children's Names
Date Modified
Company
Event Cancelled
UID
Page Image
Home Address State Or Province
```

Some of the properties that can be retrieved are

- ID
- Title
- Description
- Group
- Default Value
- Schema XML
- Scope

9.4 How to retrieve a field using field name

In this section, you will learn how to retrieve the required field (columns) using field internal name from SharePoint site by PnP PowerShell script. Get-SPOField cmdlet is used along with identity parameter for retrieving field.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RetrieveFieldByName.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get a Field
function RetrieveFieldByName(){

   # Input Parameter
   $fieldName = "PnPSiteColumn"

   # Retrieves Field or Site Column
   $field = Get-SPOField -Identity $fieldName

   # Write output on the console
   Write-Host "Field ID : " $field.Id
   Write-Host "Field Title : " $field.Title
   Write-Host "Description : " $field.Description
   Write-Host "Default Value : " $field.DefaultValue
   Write-Host "Group : " $field.Group
   Write-Host "Schema XML : " $field.SchemaXml
   Write-Host "Scope : " $field.Scope
}

# Calls the Function
RetrieveFieldByName # Retrieves required field by using field name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Field\RetrieveFieldByName.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Field ID :  a2e73a56-e5b4-443d-8fbc-e378e4b3e8dd
Field Title :  PnPSiteColumn
Description :
Default Value :
Group :  PnPGroup
Schema XML :  <Field Type="Text" Name="PnPSiteColumn" DisplayName="PnPSiteColumn" ID="{a2e73a56-e5b4-443d-8fbc-e378e4b3e8dd}" Group="PnPGroup" Required=
"FALSE" SourceID="{2223cc30-649f-4ff1-a787-7f36c777f791}" StaticName="PnPSiteColumn" Version="1" />
Scope :  /
```

9.5 How to remove a field from Content Type

In this section, you will learn how to remove a site column (field) from site content type on SharePoint site using PnP PowerShell script. Remove-SPOFieldFromContentType command is used to remove fields from site content type. The field name and content type names are the parameters for completing the operation.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RemoveFieldFromCT.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to remove field (site column) from content type
function RemoveColumnFromCT(){

   # Input Parameters
   $columnName = "PnPSiteColumn"
   $contentTypeName = "PnPContentType"

   # Removes the field from content type
   Remove-SPOFieldFromContentType -Field $columnName -ContentType $contentType-
Name
}

# Calls the Function
RemoveColumnFromCT # Removes field from the SharePoint site content type
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site column “PnPSiteColumn” will be removed from the site content type “PnPContentType”.

The result can be viewed by opening the “PnPContentType” content type page. The URL for content type’s page is https://nakkeerann.sharepoint.com/_layouts/15/mngctype.aspx.

9.6 How to delete a field from a site

In this section, you will learn how to delete a site column (field) from SharePoint site using PnP PowerShell script. Remove-SPOField command is used to remove fields from the site. The field name is used to complete the operation

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RemoveField.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to remove field (site column) from site
function RemoveColumn(){

   # Input parameter for removing column
   $columnName = "PnPSiteColumn"

   # Removes the field from site
   Remove-SPOField -Identity $columnName -Force
}

# Calls the Function
RemoveColumn # Removes field from the SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The field (column) will be deleted from the site. The URL to access site columns is https://nakkeerann.sharepoint.com/_layouts/15/mngfield.aspx.

9.7 How to create a field on a list

In this section, you will learn how to create a list column (field) on SharePoint list using PnP PowerShell script. Add-SPOField is used to create a column along with list name, display Name, internal name, group name and type parameters.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\CreateListField.ps1).

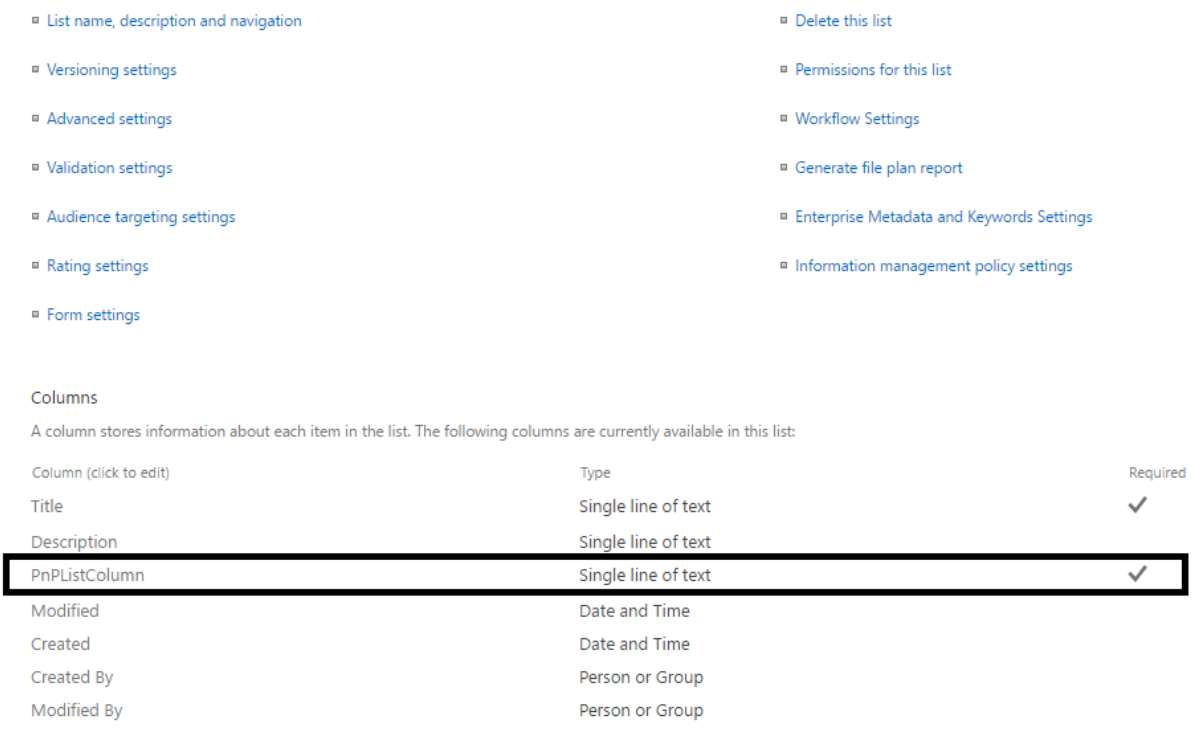
```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create a list Field (list column)
function CreateListField(){

   # Input Parameters
   $listColumnName = "PnPListColumn"
   $columnGroup = "PnPGroup"
   $listName = "PnPList"

   # Creates List Field
   Add-SPOField -DisplayName $listColumnName -InternalName $listColumnName -Group
$columnGroup -Type Text -List $listName -AddToDefaultView -Required
}

# Calls the Function
CreateListField # Creates a list field (list column) on "PnPList" list of Share-Point site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Field\CreateListField.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:

Title          InternalName  Id
-----          ------------  --
PnPListColumn PnPListColumn a0daa0c4-dab7-444d-be49-d4544f97bf66
```

The below snapshot shows the field created on SharePoint list.



9.8 How to retrieve all fields from a list

In this section, you will learn how to retrieve all the fields (columns) from a SharePoint list using PnP PowerShell script. Get-SPOField command is used to retrieve all the fields. The list parameters filters the fields only from a list.

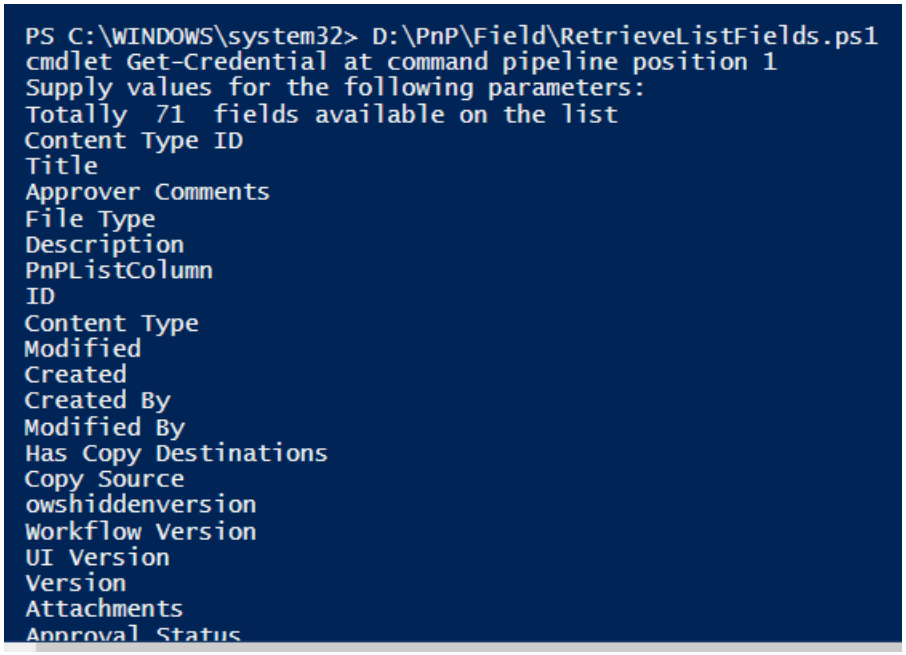
The below code shows retrieving the fields and displays the field (column) names.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RetrieveListFields.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get all Field Items with properties from a list
function RetrieveListFields(){
   # Retrieves Fields or Columns
   $fields = Get-SPOField -List "PnPList"

   Write-Host "Totally " $fields.Count " fields available on the list"
   # Write output on the console
   foreach($field in $fields){
      Write-Host $field.Title
   }
}

# Calls the Function
RetrieveListFields # Retrieves all the fields from a particular list
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below. It shows the cropped image. The image shows the field titles.



Some of the properties that can be retrieved are

- ID
- Title
- Description
- Group
- Default Value
- Schema XML
- Scope

9.9 How to retrieve a field from a list using field name

In this section, you will learn how to retrieve the required field (columns) using field internal name from SharePoint list by PnP PowerShell script. Get-SPOField cmdlet is used to retrieve fields from site. List and identity parameters are used to filter out to get the respective field from list.

The following example shows retrieving PnPListColumn Column from PnPList, which was created above.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RetrieveListFieldByName.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Get a field with properties from a list
function RetrieveListFields(){

   # Input Parameters
   $listName = "PnPList"
   $listColumName = "PnPListColumn"

   # Retrieves Fields or Columns
   $column = Get-SPOField -List $listName -Identity $listColumName

   # Write output on the console
   Write-Host "Column Title  :" $column.Title
   Write-Host "Description   :" $column.Description
   Write-Host "Group Name    :" $column.Group
   Write-Host "Internal Name :" $column.InternalName
   Write-Host "Static Name   :" $column.StaticName
   Write-Host "Scope         :" $column.Scope
   Write-Host "Type          :" $column.TypeDisplayName
   Write-Host "Schema XML    :" $column.SchemaXml
   Write-Host "Is Required?  :" $column.Required
   Write-Host "Is read only? :" $column.ReadOnlyField
   Write-Host "Unique?       :" $column.EnforceUniqueValues
}

# Calls the Function
RetrieveListFields # Retrieves a field from a particular list with properties
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Field\RetrieveListFieldByName.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Column Title  : PnPListColumn
Description   :
Group Name    : PnPGroup
Internal Name : PnPListColumn
Static Name   : PnPListColumn
Scope         : /Lists/PnPList
Type          : Single line of text
Schema XML    : <Field Type="Text" Name="PnPListColumn" DisplayName="PnPListColumn" ID="{a0daa0c4-dab7-444d-be49-d4544f97bf66}" Group="PnPGroup" Require
d="TRUE" SourceID="{5e9a2ec1-3954-4481-9354-62cd1f253305}" StaticName="PnPListColumn" ColName="nvarchar4" RowOrdinal="0" Version="2" />
Is Required?  : True
Is read only? : False
Unique?       : False

PS C:\WINDOWS\system32>
```

9.10 How to delete a field from a list

In this section, you will learn how to delete a list column (field) from SharePoint list using PnP PowerShell script. Remove-SPOField command is used to remove fields from the site. The field name is used to complete the operation.

The following example shows removing “PnPListColumn” field from “PnPList” list.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Field\RemoveListField.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to remove field (site column) from list
function RemoveListColumn(){

   # Input Parameters
   $listName = "PnPList"
   $listColumName = "PnPListColumn"

   # Removes the field from list
   Remove-SPOField -List $listName -Identity $listColumName -Force
}

# Calls the Function
RemoveListColumn # Removes field from the SharePoint list
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The field (column) will be deleted from the SharePoint list.

10 SharePoint Folder tasks using PnP PowerShell

In this topic, you will learn how to perform basic folder operations like add, retrieve or delete to/from libraries using PnP PowerShell scripts.

The Cmdlets used for the below operations are,

- Add-SPOFolder
- Get-SPOFolder
- Remove-SPOFolder

10.1 How to create a folder on a library

In this section, you will learn how to create folders on SharePoint library using PnP PowerShell script. Add-SPOFolder cmdlet is used to create a folder on document library. Name and folder path are passed as parameters for creating folder.

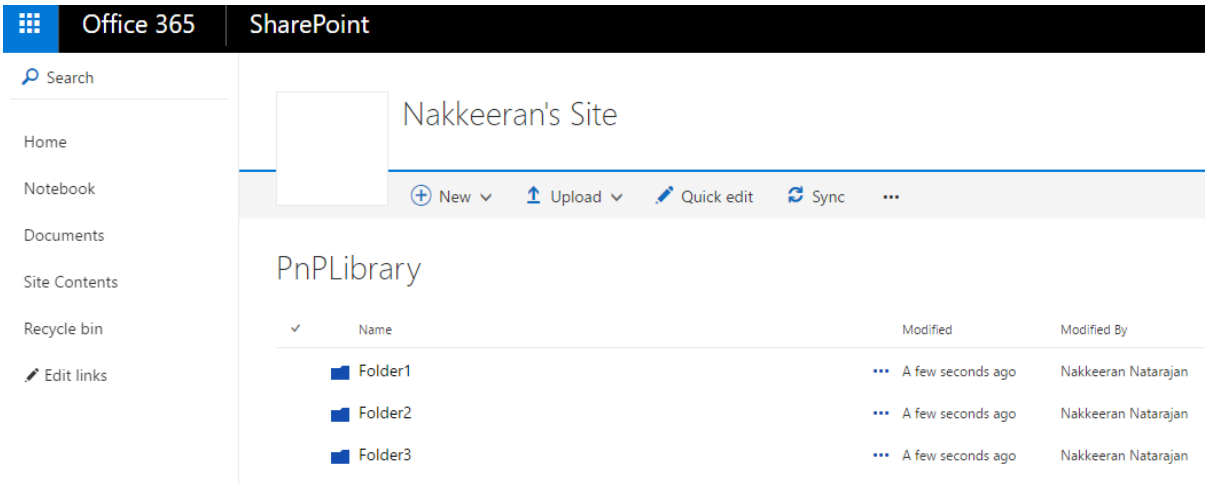
The following example creates a folders “Folder1”, etc under “PnPLibrary” library.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Folder\CreateFolder.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Create folder on document library
function CreateFolders(){

   # Input Parameters
   $folderName = "Folder1"
   $libraryName = "/PnPLibrary"

   # Creates "Folder1" under "PnPLibrary" library
   Add-SPOFolder -Name $folderName -Folder $libraryName
   # Similarly other folders are created
   Add-SPOFolder -Name "Folder2" -Folder "/PnPLibrary"
   Add-SPOFolder -Name "Folder3" -Folder "/PnPLibrary"
   Write-Host "Folders Created"
}

# Calls the Function
CreateFolders # Creates folders on SP document library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The folder is created on a document library. The below snapshot shows the created folders.



10.2 How to retrieve folders available on a site

In this section, you will learn how to retrieve all the folders from SharePoint site using PnP PowerShell script. Get-SPOFolder item cmdlet is used to retrieve the folders along with folder relative URL. To get the site folders, "/" is used as value.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Folder\RetrieveFolders.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve folders from site
function RetrieveFolders(){

   $folderPath = "/"

   # Retrieves folders
   $folders = Get-SPOFolderItem -FolderSiteRelativeUrl $folderPath
   # Write output on the console
   foreach($folder in $folders){
      Write-Host $folder.Name " - " $folder.ServerRelativeUrl

   }
}

# Calls the Function
RetrieveFolders # Retrieve folders from site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Folder\RetrieveFolders.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
_catalogs  -  /_catalogs
_cts  -  /_cts
_private  -  /_private
_vti_pvt  -  /_vti_pvt
Cache Profiles  -  /Cache Profiles
DeviceChannels  -  /DeviceChannels
FormServerTemplates  -  /FormServerTemplates
images  -  /images
IWConvertedForms  -  /IWConvertedForms
Lists  -  /Lists
Long Running Operation Status  -  /Long Running Operation Status
m  -  /m
Notification Pages  -  /Notification Pages
PnPLibrary  -  /PnPLibrary
ProjectPolicyItemList  -  /ProjectPolicyItemList
PublishedLinks  -  /PublishedLinks
Quick Deploy Items  -  /Quick Deploy Items
Relationships List  -  /Relationships List
Reports List  -  /Reports List
ReusableContent  -  /ReusableContent
Shared Documents  -  /Shared Documents
SiteAssets  -  /SiteAssets
SiteCollectionDocuments  -  /SiteCollectionDocuments
SiteCollectionImages  -  /SiteCollectionImages
SitePages  -  /SitePages
Style Library  -  /Style Library
Translation Packages  -  /Translation Packages
Translation Status  -  /Translation Status
Variation Labels  -  /Variation Labels
WorkflowTasks  -  /WorkflowTasks
default.aspx  -  /default.aspx
GettingStarted.aspx  -  /GettingStarted.aspx
newsfeed.aspx  -  /newsfeed.aspx

PS C:\WINDOWS\system32>
```

Similarly, other properties can be accessed.

10.3 How to retrieve folders from library or folders

In this section, you will learn how to retrieve all the folders from SharePoint library using PnP PowerShell script. Get-SPOFolder item cmdlet is used to retrieve the folders along with folder relative URL. Item type is used as parameter to filter required items.

The following example shows retrieving only folder items from a library or folder.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Folder\RetrieveLibraryFolders.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve folders from library or folders
function RetrieveLibraryFolders(){

   # Input
   $folderPath = "PnPLibrary"

   # Retrieves folders from library or folder path specified
   $folders = Get-SPOFolderItem -FolderSiteRelativeUrl $folderPath -ItemType Folder
   # Write output on the console
   foreach($folder in $folders){
      Write-Host "URL : " $folder.ServerRelativeUrl
      Write-Host "Name: " $folder.Name
   }
}

# Calls the Function
RetrieveLibraryFolders # Retrieves folders from library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Folder\RetrieveLibraryFolders.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
URL :  /PnPLibrary/Folder1
Name:  Folder1
URL :  /PnPLibrary/Folder2
Name:  Folder2
URL :  /PnPLibrary/Folder3
Name:  Folder3
URL :  /PnPLibrary/Forms
Name:  Forms
```

10.4 How to retrieve a folder from library or folder using folder name

In this section, you will learn how to retrieve required folder from SharePoint library or folder using PnP PowerShell script. Get-SPOFolder item cmdlet is used to retrieve the folders along with folder relative URL and folder name. Item type is used as parameter to filter required items.

The following example shows retrieving required sub folder item from a folder. Same logic is applicable for getting folder from library.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Folder\RetrieveFolder.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve folders from library or folders
function RetrieveLibraryFolderByName(){

   # Input Parameters
   $folderPath = "/PnPLibrary/Folder1"
   $subfolderName = "Folder11"

   # Retrieves folders from library or folder path specified
   $folder = Get-SPOFolderItem -FolderSiteRelativeUrl $folderPath -ItemName $subfolder-
Name
   # Write output on the console
   Write-Host "URL : " $folder.ServerRelativeUrl
   Write-Host "Name: " $folder.Name

}

# Calls the Function
RetrieveLibraryFolderByName # Retrieves folders from library
```

- Run the script from Debug menu or by pressing F5.

- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\Folder\RetrieveFolder.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
URL :  /PnPLibrary/Folder1/Folder11
Name:  Folder11

PS C:\WINDOWS\system32>
```

10.5 How to delete a folder from a library

In this section, you will learn how to delete a folder from SharePoint library or folder using PnP PowerShell script. Remove-SPOFolder item cmdlet is used to delete folders from library using the folder relative URL and folder name parameters.

The following example deletes a sub folder from the folder using folder path and sub folder name specified. Same logic is applicable for deleting folder from library.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Folder\RemoveFolder.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to delete a folder from library or folders
function RemoveFolder(){

   # Input Parameters
   $folderPath = "/PnPLibrary/Folder1"
   $subfolderName = "Folder11"

   # Removes folder from library or folder path specified
   Remove-SPOFolder -Folder $folderPath -Name $subfolderName -Force

   Write-Host "Folder deleted"

}

# Calls the Function
RemoveFolder # Deletes a folder by name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The sub folder will be deleted from the folder. Similarly, folder from the library can be deleted.

11 SharePoint Basic File tasks using PnP PowerShell

In this topic, you will learn how to perform basic file operations like add, find, retrieve, download or delete to/from libraries using PnP PowerShell scripts.

The Cmdlets used for the below operations are,

- Add-SPOFile
- Find-SPOFile
- Get-SPOFile
- Remove-SPOFile
- Set-SPOFileCheckedOut
- Set-SPOFileCheckedIn

11.1 How to add a file

In this section, you will learn how to add a file to SharePoint library using PnP PowerShell script. Add-SPOFile cmdlet is used to add a file from local system to SharePoint document library. Local file path and server file path are mandatory parameters for adding a file.

The following example adds a file to SharePoint library folder.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\AddFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to add a file to SP library or folder
function AddFile(){
   $localFilePath = "D:\PnP\Nakkeeran.docx" # Source
   $serverFolder = "/PnPLibrary" # Target
   # Adds file to SP library
   $file = Add-SPOFile -Path $localFilePath -Folder $serverFolder
   # Write output on the console
   Write-Host "Title : " $file.Name
   Write-Host "URL   : " $file.ServerRelativeUrl
}

# Calls the Function
AddFile # Adds the file to library
```

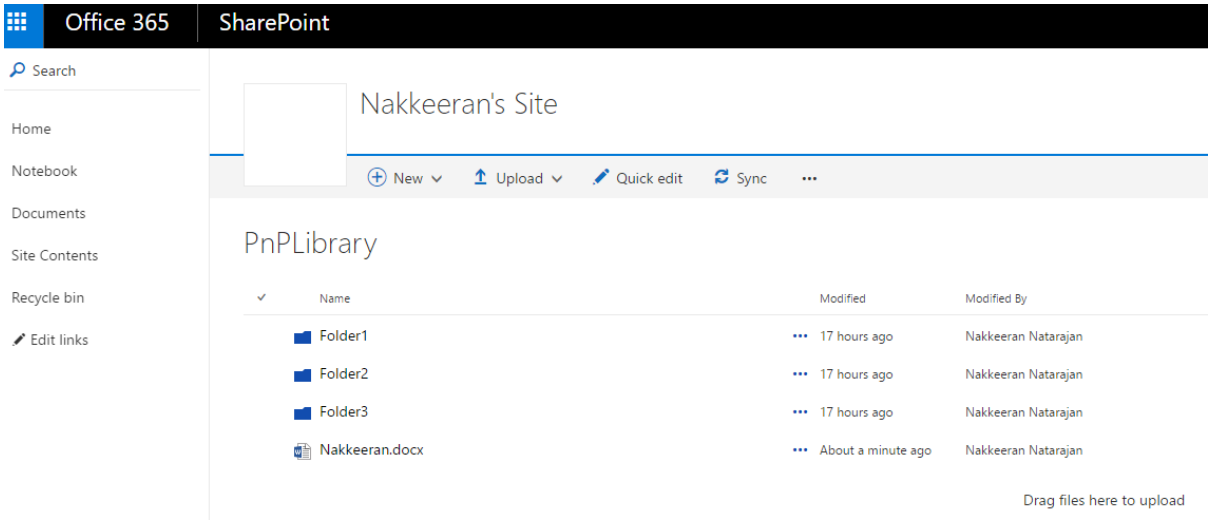
- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The file is successfully added to SharePoint library. Similarly, the files can be added to folders on Library. The respective folder URL should be given. For example, the folder parameter should be "/PnPLibrary/Folder1".

The below snapshot shows the console response.

```
PS C:\WINDOWS\system32> D:\PnP\File\AddFile.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title :  Nakkeeran.docx
URL   :  /PnPLibrary/Nakkeeran.docx

PS C:\WINDOWS\system32>
```

The below snapshot shows the library with added file.



11.2 How to retrieves from library/folder

In this section, you will learn how to retrieve all the files from SharePoint library using PnP PowerShell script. Get-SPOFolder item cmdlet is used to retrieve the files along with folder relative URL. Item type is used as parameter to filter required items. The item type should 'File'.

The following example shows retrieving only file items from a library or folder.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\RetrieveFiles.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve files from library
function RetrieveFiles(){

   # Input parameters
   $folderPath = "/PnPLibrary" # Relative Path to retrieve files

   # Retrieves files
   $files = Get-SPOFolderItem -FolderSiteRelativeUrl $folderPath -ItemType File
   # Write output on the console
   foreach($file in $files){
      Write-Host $file.Name " - " $file.ServerRelativeUrl

   }
}

# Calls the Function
RetrieveFiles # Retrieve files from SP library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\File\RetrieveFiles.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Nakkeeran.docx  -  /PnPLibrary/Nakkeeran.docx
```

11.3 How to retrieve a file using file name

In this section, you will learn how to retrieve only required file from SharePoint library using PnP PowerShell script. Get-SPOFolder item cmdlet is used to retrieve the files along with folder relative URL. Item type is used as parameter to filter required items. The item type should ‘File’. The item name is passed to get the required file.

The following example shows retrieving only required file item from a library or folder.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\RetrieveFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve required file from library
function RetrieveFile(){

   # Input parameters
   $folderPath = "/PnPLibrary" # Relative Path to retrieve files
   $fileName = "Nakkeeran.docx" # File Name

   # Retrieves file
   $file = Get-SPOFolderItem -FolderSiteRelativeUrl $folderPath -ItemType File -ItemName
$fileName

   # Write output on the console
   Write-Host "Name : " $file.Name
   Write-Host "Path : " $file.ServerRelativeUrl

}

# Calls the Function
RetrieveFile # Retrieve file from SP library by file name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The results will be displayed on console, as shown below.

```
PS C:\WINDOWS\system32> D:\PnP\File\RetrieveFile.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Name :  Nakkeeran.docx
Path :  /PnPLibrary/Nakkeeran.docx
```

11.4 How to download a file

In this section, you will learn how to download a file from SharePoint library using PnP PowerShell script. Get-SPOFile cmdlet is used to download a file from local system to SharePoint document library. Server file path, local file path and local file name are required parameters for downloading a file.

The following example downloads a file from SharePoint library.

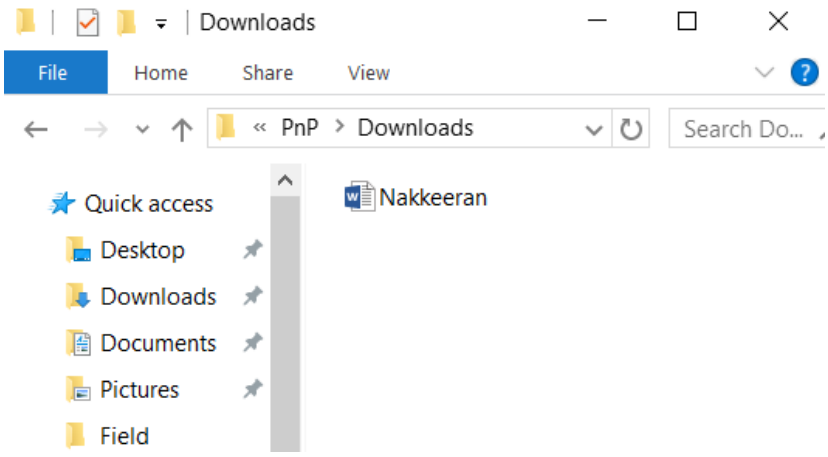
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\DownloadFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to download a file
function DownloadFile(){

   # Input parameters
   $filePath = "/PnPLibrary/Nakkeeran.docx" # Path of SP file
   $localPath = "D:\PnP\Downloads" # Local path to save file
   $localFileName = "Nakkeeran.docx" # Local system file name

   # Download a file from the path specified
   Get-SPOFile -ServerRelativeUrl $filePath -Path $localPath -FileName $localFileName
   Write-Host "File Saved"
}

# Calls the Function
DownloadFile # Download a file
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The file is successfully downloaded to local from SharePoint library. Similarly, the files can be downloaded from folders/subfolders on Library. The respective folder URL should be given. For example, the folder parameter should be “/PnPLibrary/Folder1” to download a file from subfolder.

The below snapshot shows the file downloaded from SP library.



11.5 How to check-out a file

In this section, you will learn how to checkout a file from SharePoint library using PnP PowerShell script. Set-SPOFileCheckedOut cmdlet is used to checkout a file SharePoint document library. Server relative URL of file is required parameter for checking out a file.

The following example checks out a file from SharePoint library.

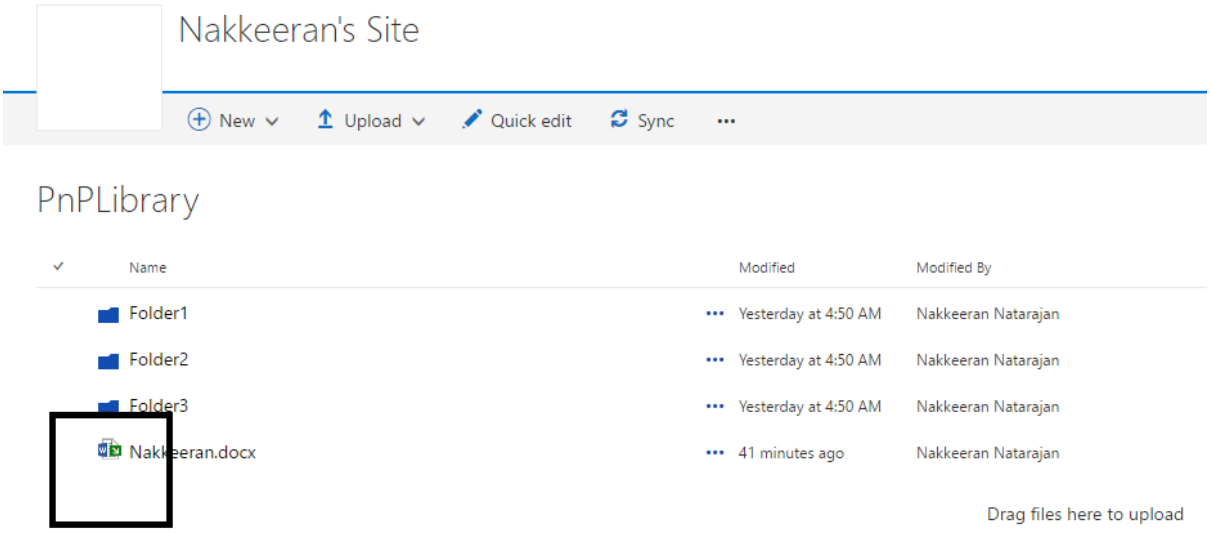
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\CheckOutFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to checkout a file on SP library or folder
function CheckoutFile(){

   # Input Parameter
   $filePath = "/PnPLibrary/Nakkeeran.docx"

   # Checks out file on SP library
   Set-SPOFileCheckedOut -Url $filePath

   Write-Host "File is checked out for editing"
}

# Calls the Function
CheckoutFile # Check out file on the library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The file is successfully checked out for editing on SharePoint library. Similarly, the files can be checked out from folders/subfolders on Library. The respective folder URL should be given. For example, the folder parameter should be "/PnPLibrary/Folder1" to checkout a file from subfolder.

The below snapshot shows the file checked out from SP library.



11.6 How to check-in a file

In this section, you will learn how to checkin a file on SharePoint library using PnP PowerShell script. Set-SPOFileCheckedIn cmdlet is used to check in a file on SharePoint document library. Server relative URL of file is required parameter for check in operation.

The following example checks in a file on SharePoint library.

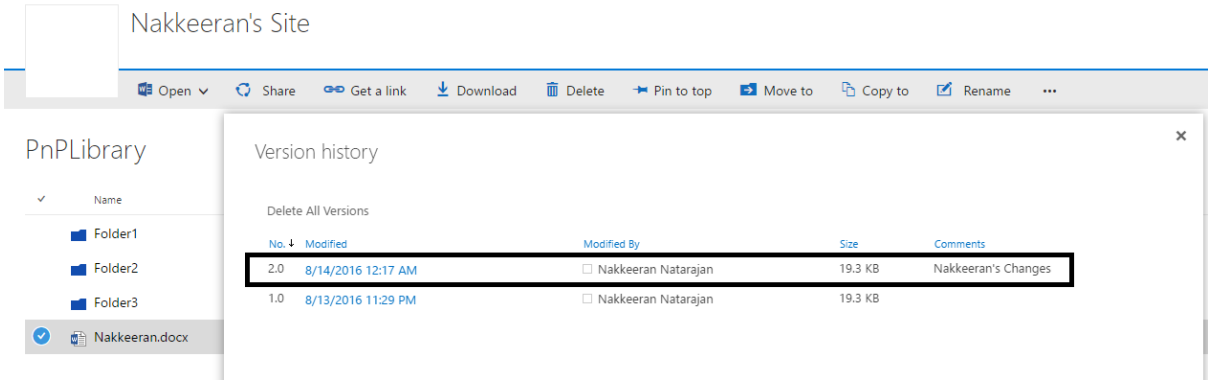
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\File\CheckInFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to checkin a file on SP library or folder
function CheckInFile(){

   # Input Parameter
   $filePath = "/PnPLibrary/Nakkeeran.docx" # SP file relative path
   $checkInComment = "Nakkeeran's Changes" # Comment for publishing file

   # Checks in file on SP library
   Set-SPOFileCheckedIn -Url $filePath -CheckinType MajorCheckIn -Comment $checkInCom-
ment

   Write-Host "File is checked in"
}

# Calls the Function
CheckInFile # Check in file on the library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The file is successfully checked in on SharePoint library. Similarly, the files can be checked in to folders/subfolders on Library. The respective folder URL should be given. For example, the folder parameter should be "/PnPLibrary/Folder1" to check in a file from subfolder.

The below snapshot shows the file checked in details on SP library. The version history of file shows the recent check in information.



11.7 How to delete a file

In this section, you will learn how to delete a file from SharePoint library using PnP PowerShell script. Remove-SPOFile cmdlet is used to delete a file from SharePoint document library. Server relative file URL is required for deleting a file.

The following example deletes a file from SharePoint library.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file.

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to delete a file
function RemoveFile(){
   # Input parameter
   $filePath = "/PnPLibrary/Nakkeeran.docx" # Path of SP file

   # Deltes a file from the path specified
   Remove-SPOFile -ServerRelativeUrl $filePath -Force
   Write-Host "File deleted"

}

# Calls the Function
RemoveFile # Deletes a file
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The file is successfully deleted from SharePoint library. Similarly, the files can be deleted from folders/subfolders on Library. The respective file URL should be given. For example, the URL parameter should be “/PnPLibrary/Folder1/filename” to delete a file from subfolder.

12 SharePoint Page Tasks using PnP PowerShell

In this topic, you will learn how to perform basic page operations like add, update, retrieve and delete to/from SharePoint site using PnP PowerShell scripts.

The Cmdlets used for the below operations are,

- Add-SPOPublishingPage
- Add-SPOWikiPage
- Get-SPOListItem
- Get-SPOWikiPageContent
- Set-SPOWikiPageContent
- Remove-SPOListItem
- Remove-SPOWikiPage

12.1 How to create a page using known page template name

In this section, you will learn how to create a page on SharePoint site using PnP PowerShell script. Add-SPOPublishingPage cmdlet is used to create a page on the site. Page name and page template name are required for creating a page.

The following example creates a page on SharePoint pages library.

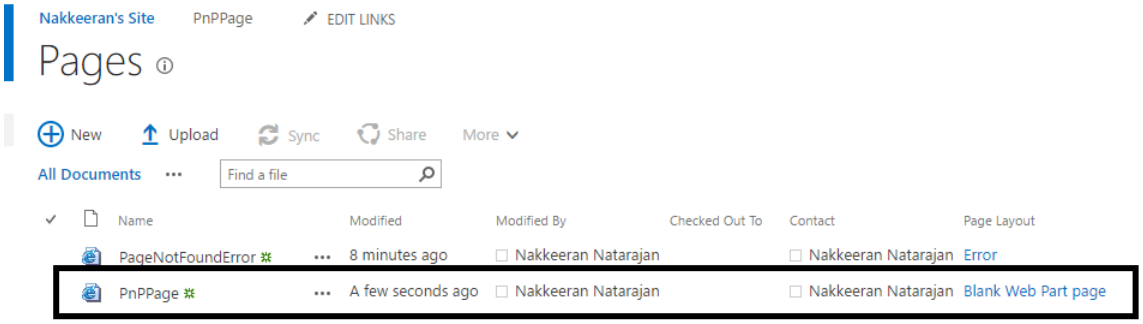
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\CreatePage.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create a SharePoint page
function CreatePage(){

   # Input Parameters
   $pageName = "PnPPage" # Page Name
   $pageTemplate = "BlankWebPartPage" # Page layout for a page

   # Create page on pages library
   Add-SPOPublishingPage -PageName $pageName -PageTemplateName $pageTemplate -
Title $pageName -Publish

   Write-Host "Page Created"
}

# Calls the Function
CreatePage # Creates a page on pages library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page is successfully created on SharePoint pages library.

The below snapshot shows the pages library with created page.



12.2 How to create a wiki page using string variable

In this section, you will learn how to create a wiki page on SharePoint site using PnP PowerShell script. Add-SPOWikiPage cmdlet is used to create a wiki page on the site. Page name and page content are passed as parameters for creating page.

Note: The wiki pages library should be created before executing the below operation.

The following example creates a wiki page on the site using page content.

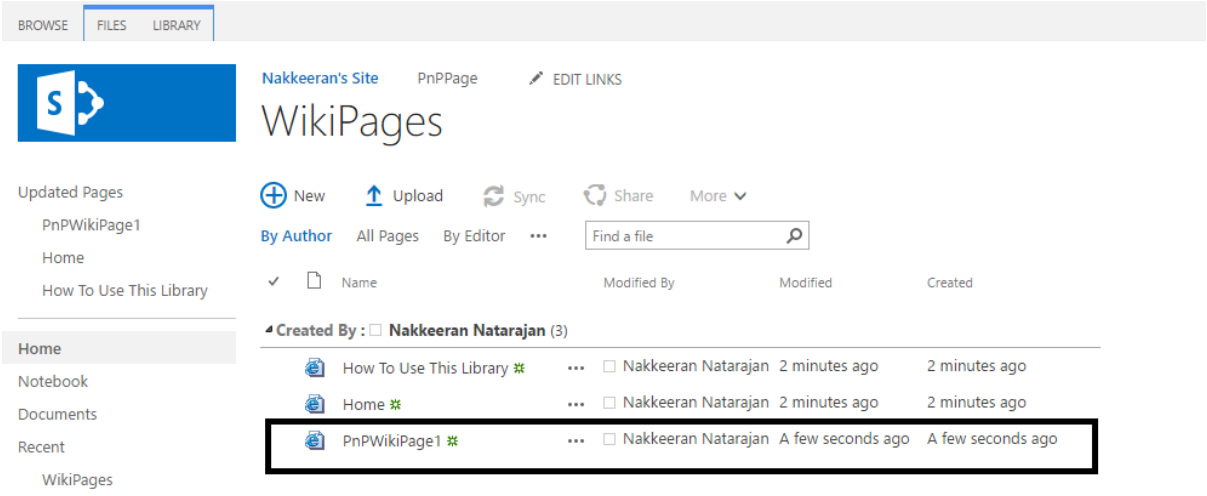
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\CreateWikiPage.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create a Wiki page
function CreateWikiPage(){

   # Input Parameters
   $pagePath = "/WikiPages/PnPWikiPage1.aspx" # Page URL with Name
   $pageContent = "The wiki page is created by Nakkeeran. Its created using the PnP Pow-
erShell Cmdlet." # Page Content

   # Create wiki page
   Add-SPOWikiPage -ServerRelativePageUrl $pagePath -Content $pageContent

   Write-Host "Wiki Page Created"
}

# Calls the Function
CreateWikiPage # Creates a wiki page on site
```

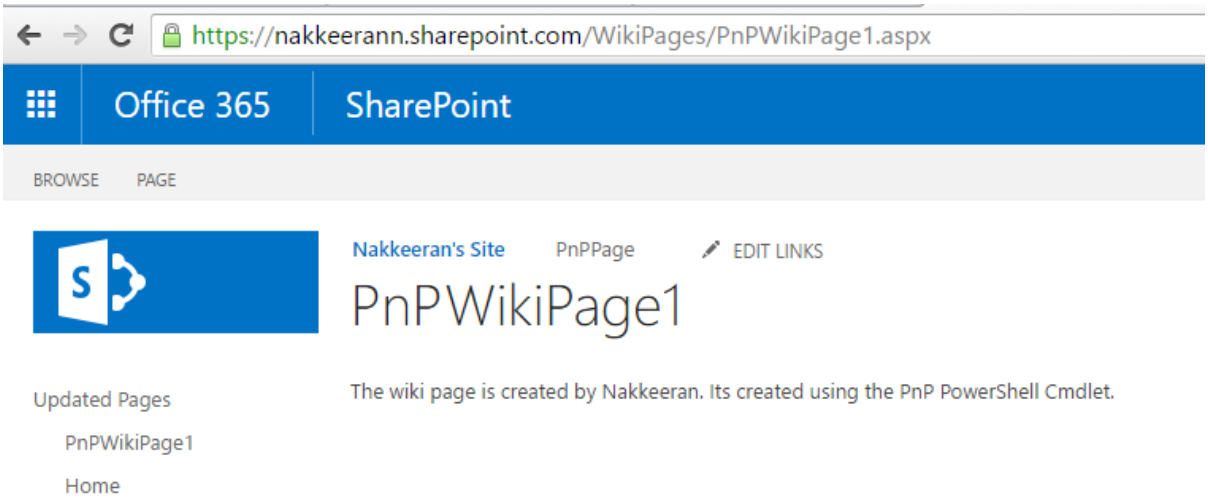
- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page is successfully created on SharePoint site wiki page's library.

The below snapshot shows the wiki pages library with created page.



The below snapshot shows the page with page content.



12.3 How to create a wiki page using file content

In this section, you will learn how to create a wiki page on SharePoint site with page content saved locally on file system using PnP PowerShell script. Add-SPOWikiPage cmdlet is used to create a wiki page on the site. Page name and page content are passed as parameters for creating page.

Note: The wiki pages library should be created before executing the below operation.

The following example creates a wiki page on the site using page content.

- Open text editor and save the following wiki page content (D:\PnP\Page\WikiPageContent.ps1).

```
<div>
<h2>PnP Wiki Page</h2>
<br/>
<div>The page is created by <b>PnP PowerShell command</b></div>
<br/>
<div><i>The content is updated from local File</i> - By Nakkeeran</div>

</div>
```

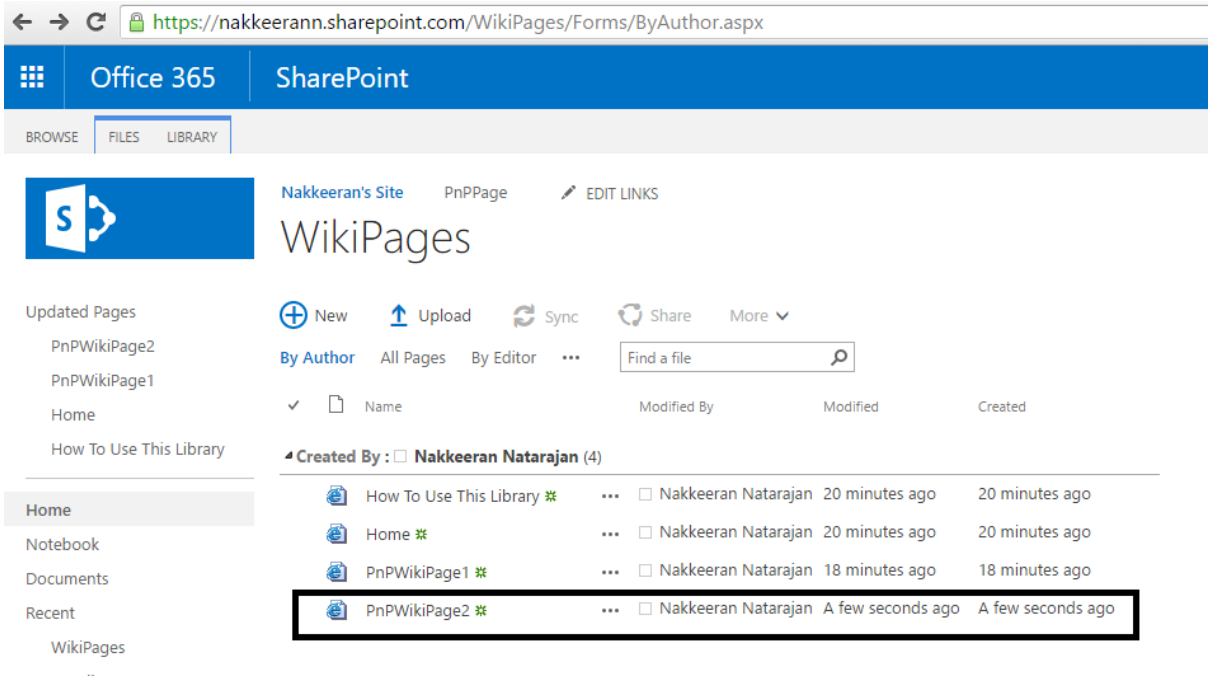
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\CreateWikiPageUsingFileContent.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to create a Wiki page using local file
function CreateWikiPageUsingFile(){

   # Input Parameters
   $pagePath = "/WikiPages/PnPWikiPage2.aspx" # Page URL with Name
   $pageContent = [IO.File]::ReadAllText("D:\PnP\Page\WikiPageContent.txt")  # Page Content

   # Create wiki page
   Add-SPOWikiPage -ServerRelativePageUrl $pagePath -Content $pageContent

   Write-Host "Wiki Page Created"
}

# Calls the Function
CreateWikiPageUsingFile # Creates a wiki page on site
```

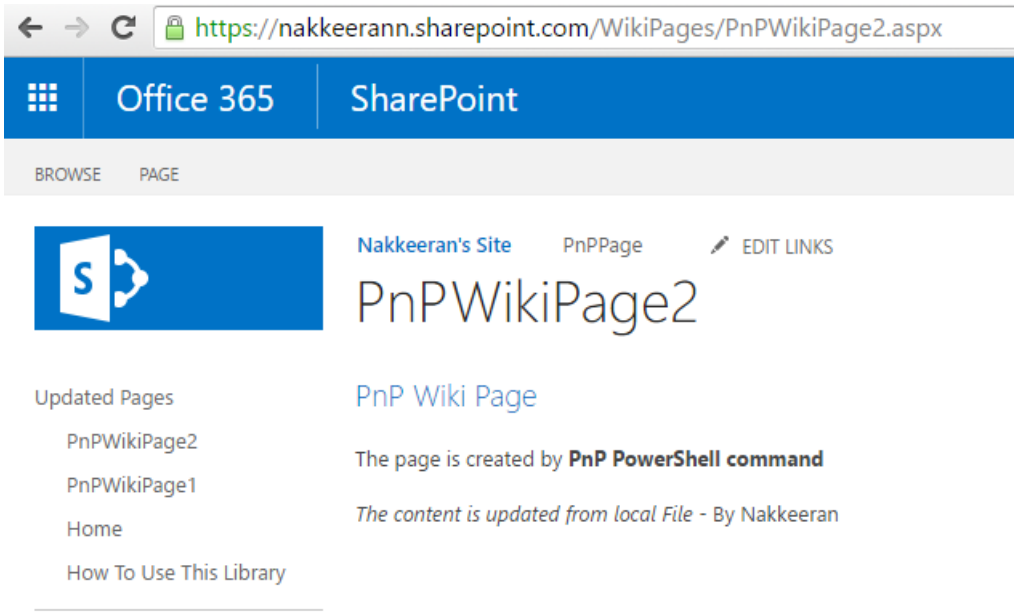
- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page is successfully created on SharePoint site wiki page's library.

The below snapshot shows the wiki pages library with created page.



The below snapshot shows the page with page content.



12.4 How to retrieve pages from pages library

In this section, you will learn how to retrieve pages from Pages Library using PnP PowerShell script. List operation cmdlet Get-SPOListItem is used for retrieving the pages. Page library name is passed as parameter for getting pages.

Note: The publishing pages library should be available before executing the below operation.

The following example retrieves the pages from pages library.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\RetrievePages.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve pages from Pages Library
function RetrievePages(){

   # Input Parameters
   $pagesLibrary = "Pages" # Page Library Name

   # Retrieves pages using get list command
   $pages = Get-SPOListItem -List "Pages"

   # Displays output on console
   foreach($page in $pages){
      Write-Host "Title   : " $page["Title"]
      Write-Host "Page URL: " $page["FileRef"]
      Write-Host "--------------------------------"
   }

}

# Calls the Function
RetrievePages # Retrieves pages
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The pages are retrieved successfully from pages library.

The below PowerShell console shows the pages retrieved.

```
PS C:\WINDOWS\system32> D:\PnP\Page\RetrievePages.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title    :  Page not found
Page URL:  /Pages/PageNotFoundError.aspx
--------------------------------
Title    :  PnPPage
Page URL:  /Pages/PnPPage.aspx
--------------------------------

PS C:\WINDOWS\system32>
```

Similarly the required pages can be retrieved using the CAML query. Other properties of a page can also be retrieved.

12.5 How to retrieve wiki page content

In this section, you will learn how to retrieve page content of wiki page using PnP PowerShell script. Get-SPOWikiPageContent cmdlet is used for retrieving the wiki page content. The wiki page URL is passed as parameter for retrieving.

The following example retrieves the page content of wiki page.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\RetrieveWikiPageContent.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve wiki page content
function RetrieveWikiPageContent(){

   # Input Parameters
   $wikiPageUrl = "/WikiPages/PnPWikiPage2.aspx" # Page URL

   # Retrieves wiki page content
   $wikiPageContent = Get-SPOWikiPageContent -ServerRelativePageUrl $wikiPageUrl

   # Display content on console
   Write-Host $wikiPageContent

}

# Calls the Function
RetrieveWikiPageContent # Retrieves wiki page content
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page content is retrieved successfully from a wiki page.

The below PowerShell console shows the page content retrieved.

```
PS C:\WINDOWS\system32> D:\PnP\Page\RetrieveWikiPageContent.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
<div class="ExternalClass939DFF88076540A5A49AEC83AEDD7EB3"><div>
<h2>PnP Wiki Page</h2>
<br>
<div>The page is created by <b>PnP PowerShell command</b></div>
<br>
<div><i>The content is updated from local File</i> - By Nakkeeran</div>

</div>  </div>

PS C:\WINDOWS\system32>
```

12.6 How to update a wiki page content using string variable

In this section, you will learn how to update page content of wiki page using PnP PowerShell script. Set-SPOWikiPageContent cmdlet is used for updating the wiki page content. The wiki page URL and content are passed as parameters for updating.

The following example retrieves the page content of wiki page.

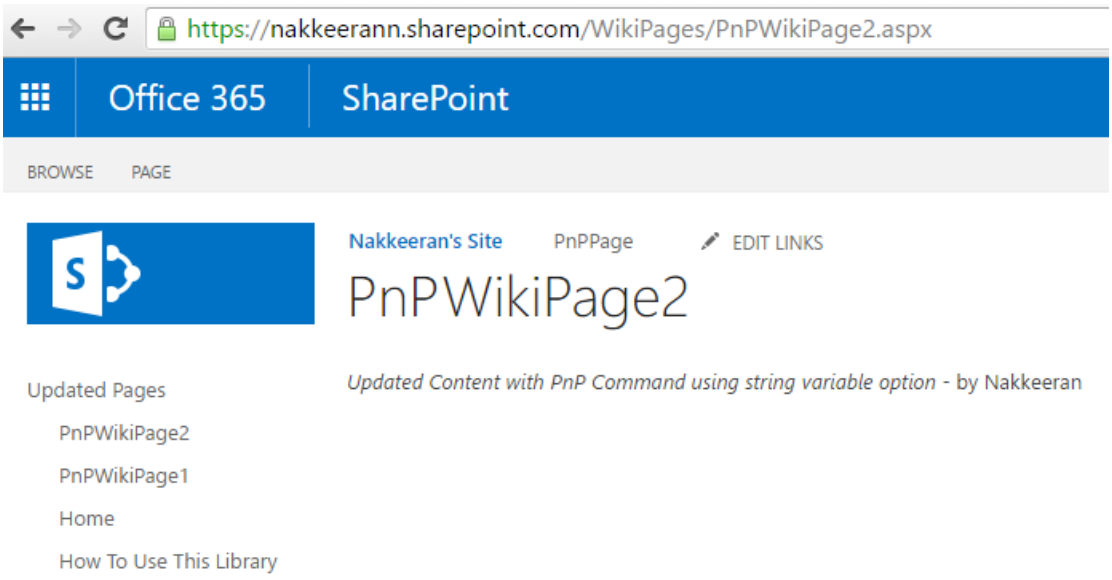
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\UpdateWikiPageContent.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update wiki page content using string
function UpdateWikiPageContent(){

   # Input Parameters
   $wikiPageUrl = "/WikiPages/PnPWikiPage2.aspx" # Page URL
   $updateContent = "<i>Updated Content with PnP Command using string variable option</i> - by Nakkeeran" # Page Content

   # Updates wiki page content using above string content
   Set-SPOWikiPageContent -ServerRelativePageUrl $wikiPageUrl -Content $updateContent

   # Display content on console
   Write-Host "Updated wiki page content"

}

# Calls the Function
UpdateWikiPageContent # Update wiki page content using String content
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page content is updated successfully to a wiki page.

The below snapshot shows the page updated.



12.7 How to update a wiki page content using file content

In this section, you will learn how to update page content of wiki page with file content using PnP PowerShell script. Set-SPOWikiPageContent cmdlet is used for updating the wiki page content. The wiki page URL and file content path are passed as parameters for updating.

The following example updates the page content of wiki page using text content saved on local system.

- Open text editor and save the following wiki page content (D:\PnP\Page\WikiPageContent.ps1).

```
<div>
<h2>PnP Wiki Page</h2>
<br/>
<div>The page is created for showing <b>PnP PowerShell command</b> operation</div>
<br/>
<div><i>The content is updated from local File</i> - By Nakkeeran</div>


</div>
```

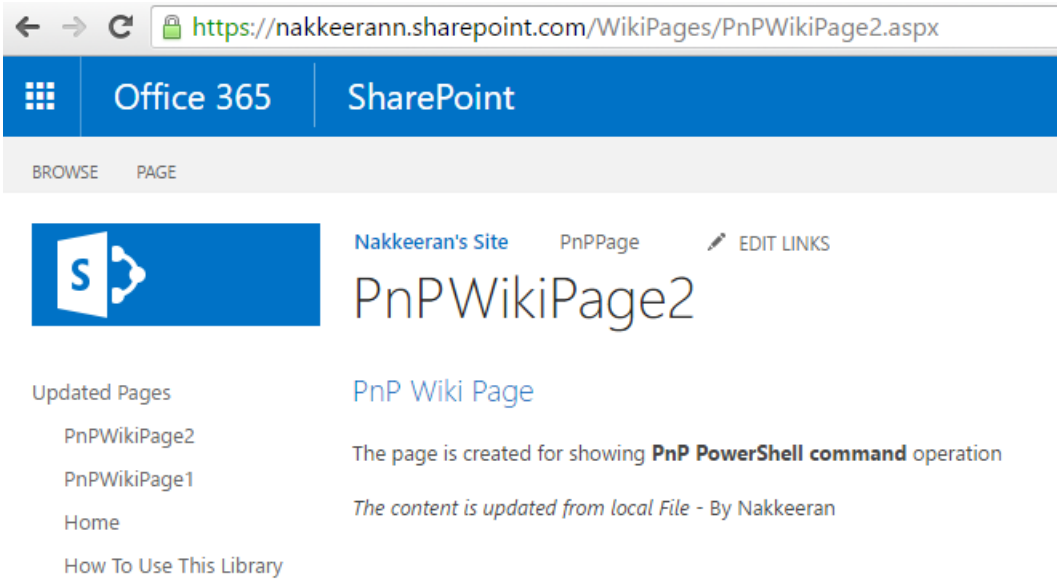
- Open Windows PowerShell ISE as administrator.
    - Open a new file, paste the following script and save the file (D:\PnP\Page\UpdateWikiPageContentUsingFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Update wiki page content using file
function UpdateWikiPageContentUsingFile(){

   # Input Parameters
   $wikiPageUrl = "/WikiPages/PnPWikiPage2.aspx" # Page URL
   $localFilePath = "D:\PnP\Page\WikiPageContent.txt" # File Page Content

   # Updates wiki page content using above file content
   Set-SPOWikiPageContent -ServerRelativePageUrl $wikiPageUrl -Path "D:\PnP\Page\Wik-
iPageContent.txt"

   # Display content on console
   Write-Host "Updated wiki page content"

}

# Calls the Function
UpdateWikiPageContentUsingFile # Update wiki page content using file content
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page content is updated successfully to a wiki page.

The below snapshot shows the page updated.



12.8 How to delete a page

In this section, you will learn how to delete a page on SharePoint site using PnP PowerShell script. The list operation Remove-SPOListItem cmdlet is used to delete a page on the site.

The page object is required for deletion. So first page object should be retrieved using page librart and page name. Then the page object is deleted.

The following example shows deleting a publishing page on the site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\RemovePage.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to delete a SharePoint page
function RemovePage(){

   # Input Parameters
   $pageName = "PnPPage" # Page Name
   $pageLibrary = "Pages" # Pages Library Name
   $pageQuery = "<View><Query><Where><Contains><FieldRef Name='Title'/><Value
Type='Text'>" + $pageName + "</Value></Contains></Where></Query></View>"

   # Retrieves page object using get list operation
   $page = Get-SPOListItem -List $pageLibrary -Query $pageQuery

   # Deletes a page
   Remove-SPOListItem -List $pageLibrary -Identity $page -Force

   Write-Host "Page Deleted"
}

# Calls the Function
RemovePage # Removes a page from pages library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The page is successfully deleted from SharePoint site.

12.9 How to delete a wiki page

In this section, you will learn how to delete a wiki page from SharePoint site using PnP PowerShell script. Remove-SPOWikiPage cmdlet is used to delete a wiki page from the site. The page relative URL is passed as parameter for deletion.

The following example shows deleting a wiki page from the site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Page\RemoveWikiPage.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to delete a SharePoint page
function RemoveWikiPage(){

   # Input Parameters
   $pagePath = "/WikiPages/PnPWikiPage1.aspx" # Page URL with Name

   # Deletes a wiki page from wiki page library
   Remove-SPOWikiPage -ServerRelativePageUrl $pagePath

   Write-Host "Page Deleted"
}

# Calls the Function
RemoveWikiPage # Removes a wiki page from wiki pages library
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The wiki page is successfully deleted from SharePoint site.

13 SharePoint Group and User Operations using PnP PowerShell

In this topic, you will learn how to perform basic SharePoint group operations like create, retrieve, update and delete groups to/from SharePoint site using PnP PowerShell scripts. Also we will see how to add user to site and add user to group using the script.

The page URL to access the groups is https://nakkeerann.sharepoint.com/_layouts/15/groups.aspx. The groups can be viewed/modified based on the permissions.

The Cmdlets used for the below operations are,

- New-SPOGroup
- Get-SPOGroup
- Set-SPOGroup
- Remove-SPOGroup
- New-SPOUser

13.1 How to create a SharePoint group

In this section, you will learn how to create a SharePoint group on SharePoint site using PnP PowerShell script. New-SPOGroup cmdlet is used to create a group on the site. Group name and group description are required for creating a group.

The following example creates a page on SharePoint pages library.

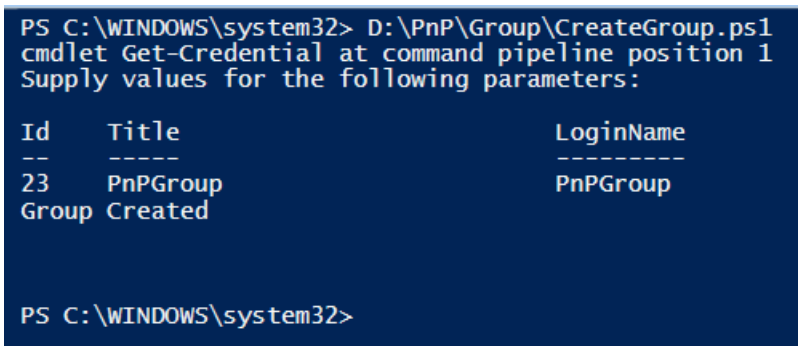
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\CreateGroup.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to Create SharePoint Group on SharePoint page
function CreateGroup(){

   # Input Parameters
   $groupName = "PnPGroup" # Group Name
   $groupDescription = "SharePoint group created from PnP PowerShell" # Group Description

   # Create SharePoint group on SharePoint site
   New-SPOGroup -Title $groupName -Description $groupDescription -AllowRequestToJoinLeave -AutoAcceptRequestToJoinLeave

   Write-Host "Group Created"
}

# Calls the Function
CreateGroup # Creates a SharePoint group on SharePoint site
```

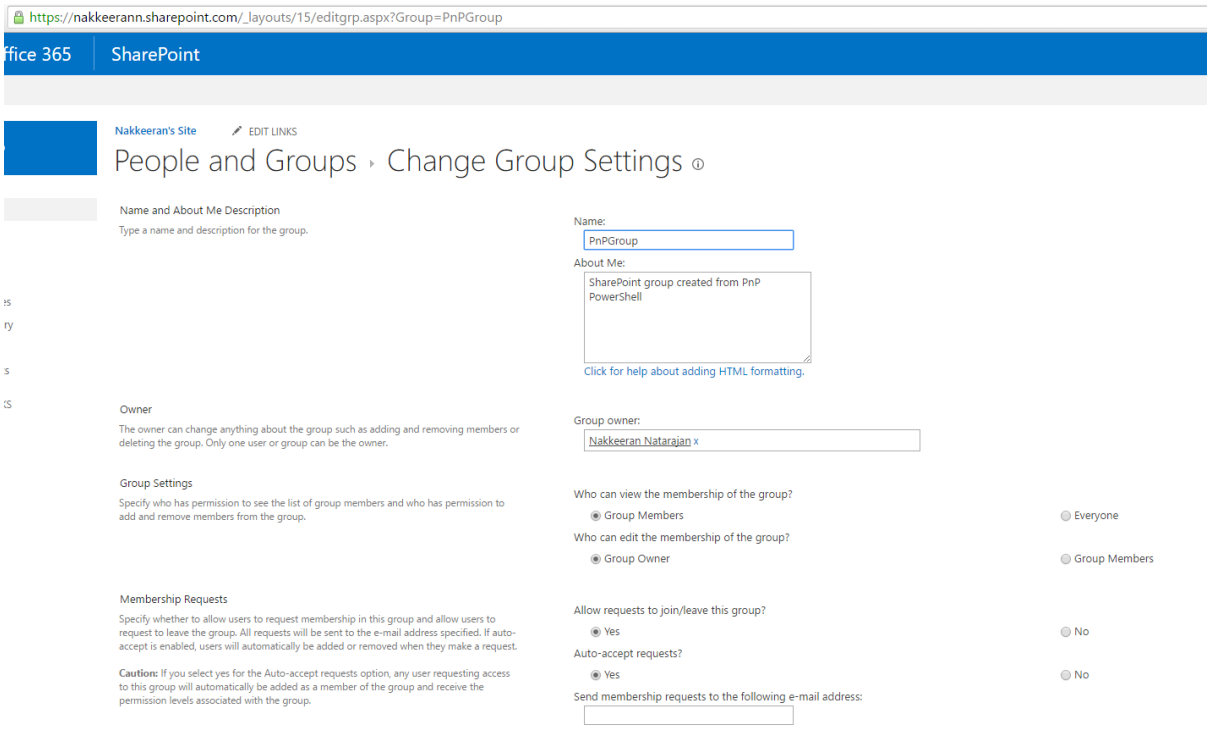
- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint group is successfully created on SharePoint site.

The below snapshot shows the group created on console.



The below snapshot shows the SharePoint group created on SharePoint portal.



13.2 How to retrieve all SharePoint groups

In this section, you will learn how to retrieve all the SharePoint groups from SharePoint site using PnP PowerShell script. Get-SPOGroup cmdlet is used to retrieve all SharePoint groups from the site.

The following example retrieves SharePoint groups from site.

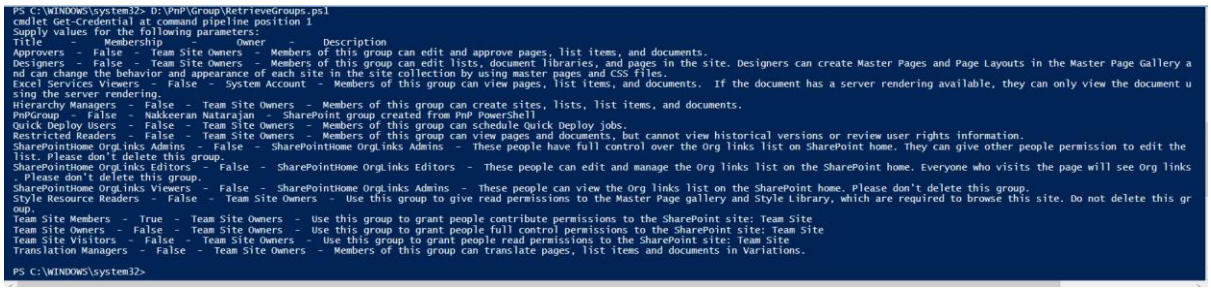
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\RetrieveGroups.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve SharePoint Groups from the site
function RetrieveGroups(){

   # Retrieves SharePoint groups from SharePoint site
   $groups = Get-SPOGroup

   # Displays output on console
   Write-Host "Title   -   Membership   -   Owner   -   Description"
   foreach($group in $groups){
      Write-Host $group.Title " - " $group.AllowMembersEditMembership  " - " $group.Own-
erTitle " - " $group.Description
   }
}

# Calls the Function
RetrieveGroups # Retrieves SharePoint groups from SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint groups are retrieved successfully.

The below snapshot shows the groups retrieved on console.



13.3 How to retrieve a SharePoint group using group title

In this section, you will learn how to retrieve only required SharePoint group by using group title with PnP PowerShell script. Get-SPOGroup cmdlet is used to retrieve all SharePoint groups from the site using group title as parameter.

The following example retrieves required SharePoint group from site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\RetrieveGroup.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to retrieve a SharePoint Group using group name
function RetrieveGroupByTitle(){

   # Input Parameter
   $groupName = "PnPGroup"

   # Retrieves SharePoint groups from SharePoint site using group name
   $group = Get-SPOGroup -Identity $groupName

   # Displays output on console
   Write-Host "Title          : " $group.Title
   Write-Host "Description      : " $group.Description
   Write-Host "Can Members edit? : " $group.AllowMembersEditMembership
   Write-Host "Owner          : " $group.OwnerTitle
}

# Calls the Function
RetrieveGroupByTitle # Retrieves required SharePoint group from SharePoint site using
name
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint group is retrieved successfully using group name.

The below snapshot shows the group retrieved on console.

```
PS C:\WINDOWS\system32> D:\PnP\Group\RetrieveGroup.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title             :  PnPGroup
Description       :  SharePoint group created from PnP PowerShell
Can Members edit? :  False
Owner             :  Nakkeeran Natarajan

PS C:\WINDOWS\system32>
```

13.4 How to update a SharePoint group

In this section, you will learn how to update the SharePoint group by using group title with PnP PowerShell script. Set-SPOGroup cmdlet is used to update the group on the site using group title with necessary changes.

The following example updates SharePoint group on site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\UpdateGroup.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to update a SharePoint Group using group name
function UpdateGroup(){

   # Input Parameter
   $groupName = "PnPGroup"
   $newTitle = "PnPSharePointGroup"
   $groupDescription = "SharePoint Group for PnP"
   $groupPermission = "Edit" # Contribute Access
   $groupOwner = "Nakkeeran" # Owner

   # Updates SharePoint groups from SharePoint site using group name
   Set-SPOGroup -Identity $groupName -Title $newTitle -Description $groupDescription -Ad-
dRole $groupPermission

   Write-Host "Group Updated"
}

# Calls the Function
UpdateGroup # Updates SharePoint group on SharePoint site using name
```

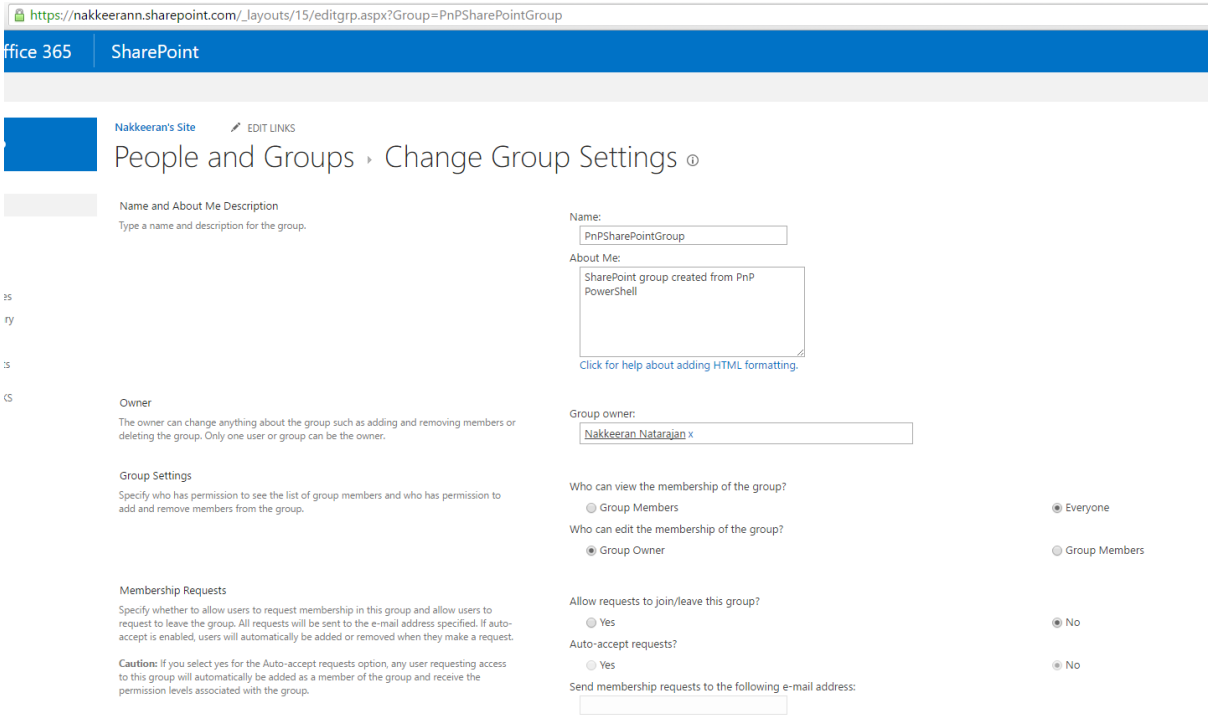
- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

To update only the permissions of a group, there is a separate command used.

Set-SPOGroupPermissions -Identity "PnPGroup" -AddRole "Edit"

The SharePoint group is updated successfully using group name.

The below snapshot shows the updated group.



13.5 How to delete a SharePoint group

In this section, you will learn how to delete the SharePoint group by using group title with PnP PowerShell script. Remove-SPOGroup cmdlet is used to delete the group on the site.

The following example deletes SharePoint group on site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\RemoveGroup.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to remove a SharePoint Group using title
function RemoveGroup(){

   # Input Parameter
   $groupName = "PnPSharePointGroup"

   # Removes SharePoint groups from SharePoint
   Remove-SPOGroup -Identity $groupName -Force

   Write-Host "Group Deleted"
}

# Calls the Function
RemoveGroup # Removes SharePoint group from SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The SharePoint group is deleted successfully from site.

13.6 How to add user to a SharePoint User List

In this section, you will learn how to add the SharePoint user to site user information list by using PnP PowerShell script. New-SPOUser cmdlet is used for add operation.

The following example adds user to SharePoint.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\CreateUser.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to add SharePoint User on to SharePoint site
function AddUser(){

   # Input Parameters
   $userLogin = "nirmal@nakkeerann.onmicrosoft.com" # Login ID

   $web = Get-SPOWeb
   # Add SharePoint user on SharePoint site
   New-SPOUser -LoginName $userLogin -Web $web

   Write-Host "UserAdded"
}

# Calls the Function
AddUser # Adds user to SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

User is added to the SharePoint site user list successfully.

The following snapshot shows the added user.

```
PS C:\WINDOWS\system32> D:\PnP\Group\CreateUser.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:


Email                  :
Groups                 : {}
IsShareByEmailGuestUser :
IsSiteAdmin            : False
UserId                 : Microsoft.SharePoint.Client.UserIdInfo
Id                     : 25
IsHiddenInUI           : False
LoginName              : i:0#.f|membership|nirmal@nakkeerann.onmicrosoft.com
Title                  : Nirmal S
PrincipalType          : User
Context                : Microsoft.SharePoint.Client.ClientContext
Tag                    :
Path                   : Microsoft.SharePoint.Client.ObjectPathIdentity
ObjectVersion          :
ServerObjectIsNull     : False
TypedObject            : Microsoft.SharePoint.Client.User

UserAdded
```

13.7 How to add user to a SharePoint Group

In this section, you will learn how to add the SharePoint user to SharePoint site group using PnP PowerShell script. Add-SPOUserToGroup cmdlet is used for add operation.

The following example adds user to SharePoint.

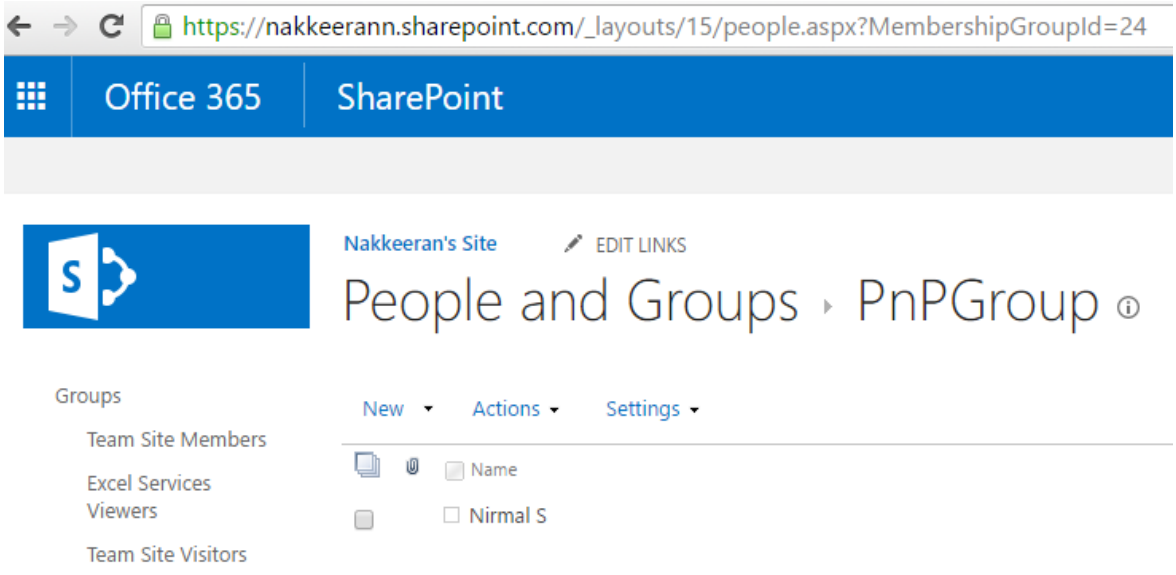
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Group\AddUserToGroup.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential
# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to add SharePoint User on to SharePoint group
function AddUserToGroup(){

   # Input Parameters
   $userLogin = "nirmal@nakkeerann.onmicrosoft.com" # Login ID
   $groupName = "PnPGroup"

   # Add SharePoint user to SharePoint group
   Add-SPOUserToGroup -LoginName $userLogin -Identity $groupName

}

# Calls the Function
AddUserToGroup # Adds user to SharePoint group
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

User is added to the SharePoint site user list successfully.

The following snapshot shows the user added to group.



14 Taxonomy Tasks using PnP PowerShell

In this topic, you will learn how to perform taxonomy operations using PnP PowerShell scripts.

The following cmdlets are used for the operations below.

The taxonomy term store of a site can be accessed on term stores page. For example, the term store URL is https://nakkeerann.sharepoint.com/_layouts/15/termstoremanager.aspx.

Please make sure, you are term store administrator before executing any of the operations. It can be changed on the tenant site. The URL will be https://nakkeerann-admin.sharepoint.com/_layouts/15/termstoremanager.aspx.

14.1 Import term stores using Text file

In this section, you will learn how to import a term store data stored on a text file in to SharePoint taxonomy term store using PnP PowerShell.

The following example explains in detail of importing a text file term store in to Taxonomy term store on SharePoint site.

- First save the term set. Open a text editor and paste the following content. Save the file as text file (For example, "D:\PnP\Taxonomy\Terms.txt").

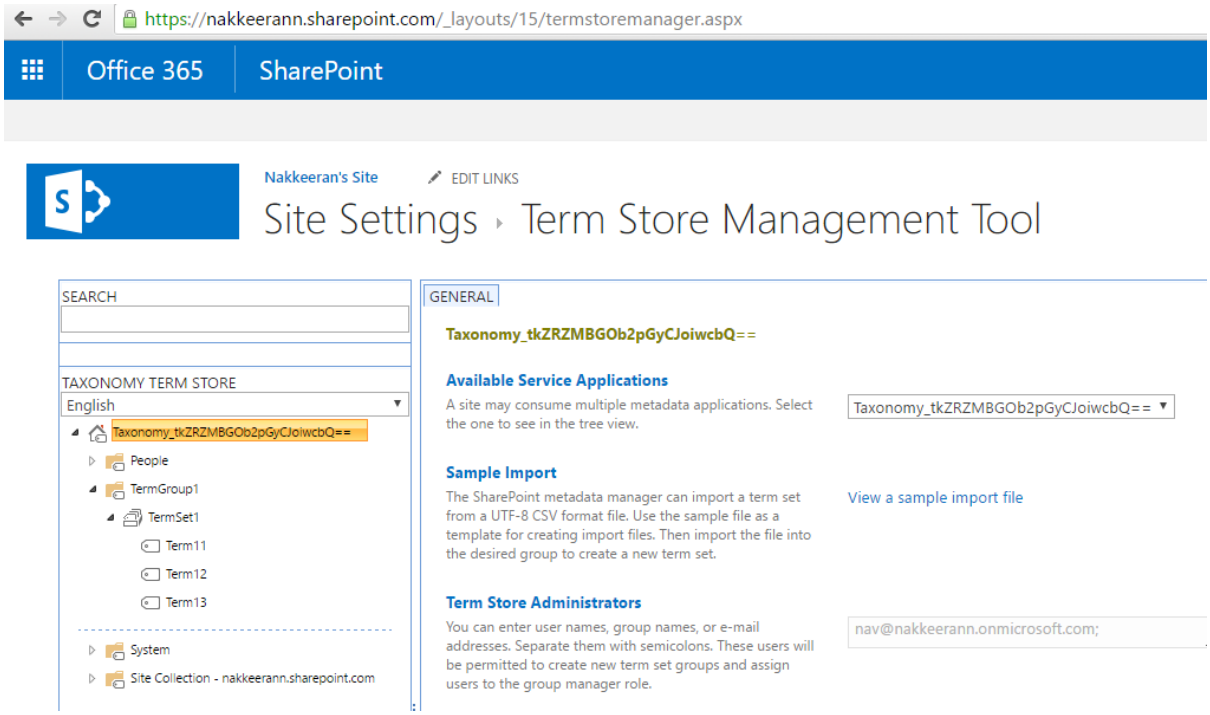
```
TermGroup1|TermSet1|Term11

TermGroup1|TermSet1|Term12

TermGroup1|TermSet1|Term13
```

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Taxonomy\ImportTestFile.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to import term set data in to SP Taxonomy
function ImportTerms(){

   # Input Parameter
   $termsPath = "D:\PnP\Taxonomy\Terms.txt"

   # Imports taxonomy data into SharePoint
   Import-SPOTaxonomy -Path $termsPath

   Write-Host "Terms Imported"
}

# Calls the Function
ImportTerms # Imports terms in to SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The tem group is created along with term sets. The following snapshot shows the term group (TermGroup1) added in the example.



14.2 Import term stores using XML file

In this section, you will learn how to import a term store data stored on a XML file in to SharePoint taxonomy term store using PnP PowerShell.

The following example explains in detail of importing a XML file term store in to Taxonomy term store on SharePoint site.

- First save the term set. Open a text editor and paste the following content. Save the file as XML file (For example, "D:\PnP\Taxonomy\Terms.xml").

```
<pnp:TermGroups
xmlns:pnp="http://schemas.dev.office.com/PnP/2015/12/ProvisioningSchema">
 <pnp:TermGroup Name="TermGroup1" Description="">
  <pnp:TermSets>
   <pnp:TermSet Name="TermSet3" Description="">
    <pnp:Terms>
     <pnp:Term Name="Term31" />
     <pnp:Term Name="Term32" />
     <pnp:Term Name="Term33" />
    </pnp:Terms>
   </pnp:TermSet>
   <pnp:TermSet Name="TermSet4" Description="">
    <pnp:Terms>
     <pnp:Term Name="Term41" />
     <pnp:Term Name="Term42" />
    </pnp:Terms>
   </pnp:TermSet>
  </pnp:TermSets>
 </pnp:TermGroup>
</pnp:TermGroups>
```

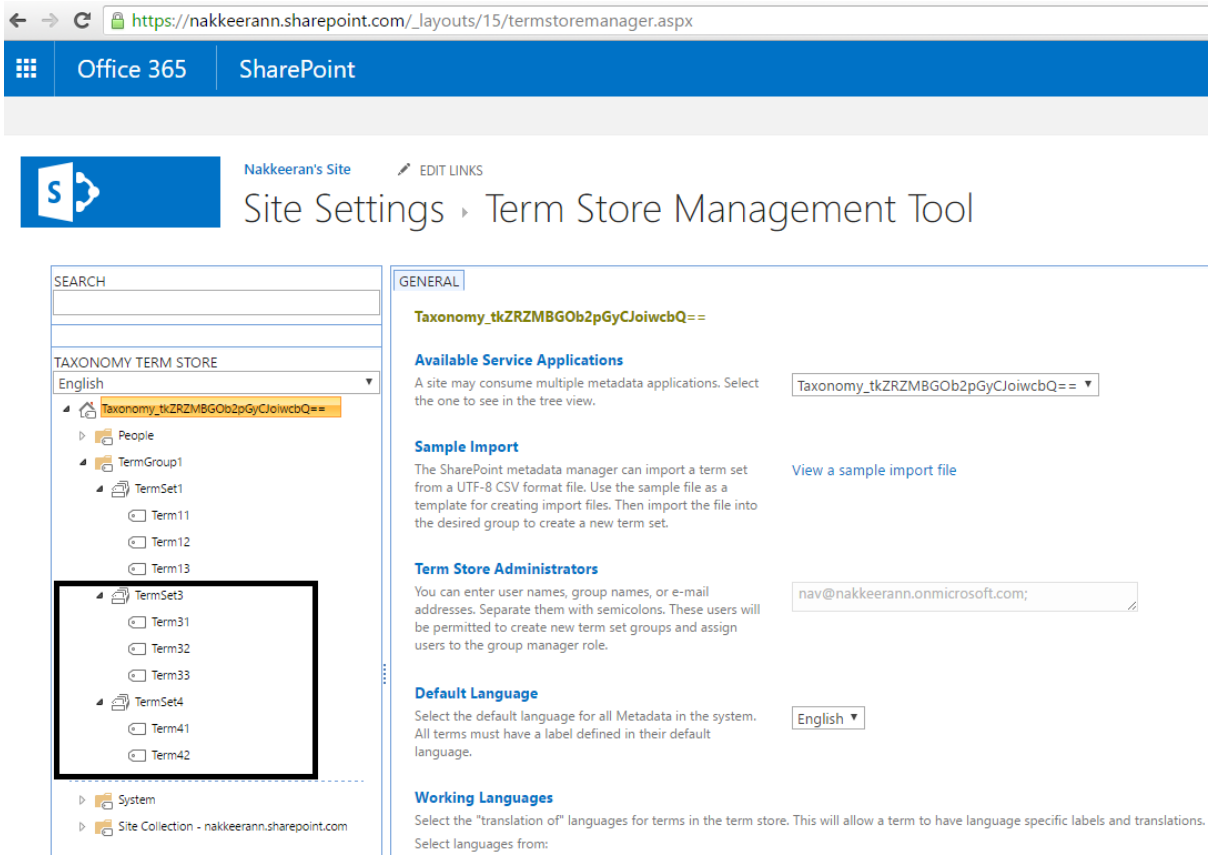
- Open Windows PowerShell ISE as administrator.

- Open a new file, paste the following script and save the file (D:\PnP\Taxonomy\ImportXML.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to import term set data in to SP Taxonomy from XML
function ImportTermsFromXML(){

   # Input Parameter
   $termsPath = "D:\PnP\Taxonomy\Terms.xml" # XML File path

   # Imports XML taxonomy data into SharePoint
   Import-SPOTermGroupFromXml -Path $termsPath

   Write-Host "Terms Imported"
}

# Calls the Function
ImportTermsFromXML # Imports terms in to SharePoint site from XML file
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The term sets are appended to the existing term group, since term group was already present. If term group doesn't exists, then new term group will be created and term sets are added. The following snapshot shows the term sets added to term group (TermGroup1) in the example.



14.3 Import term stores using string arrays

In this section, you will learn how to import a term store data stored as array variables in to SharePoint taxonomy term store using PnP PowerShell.

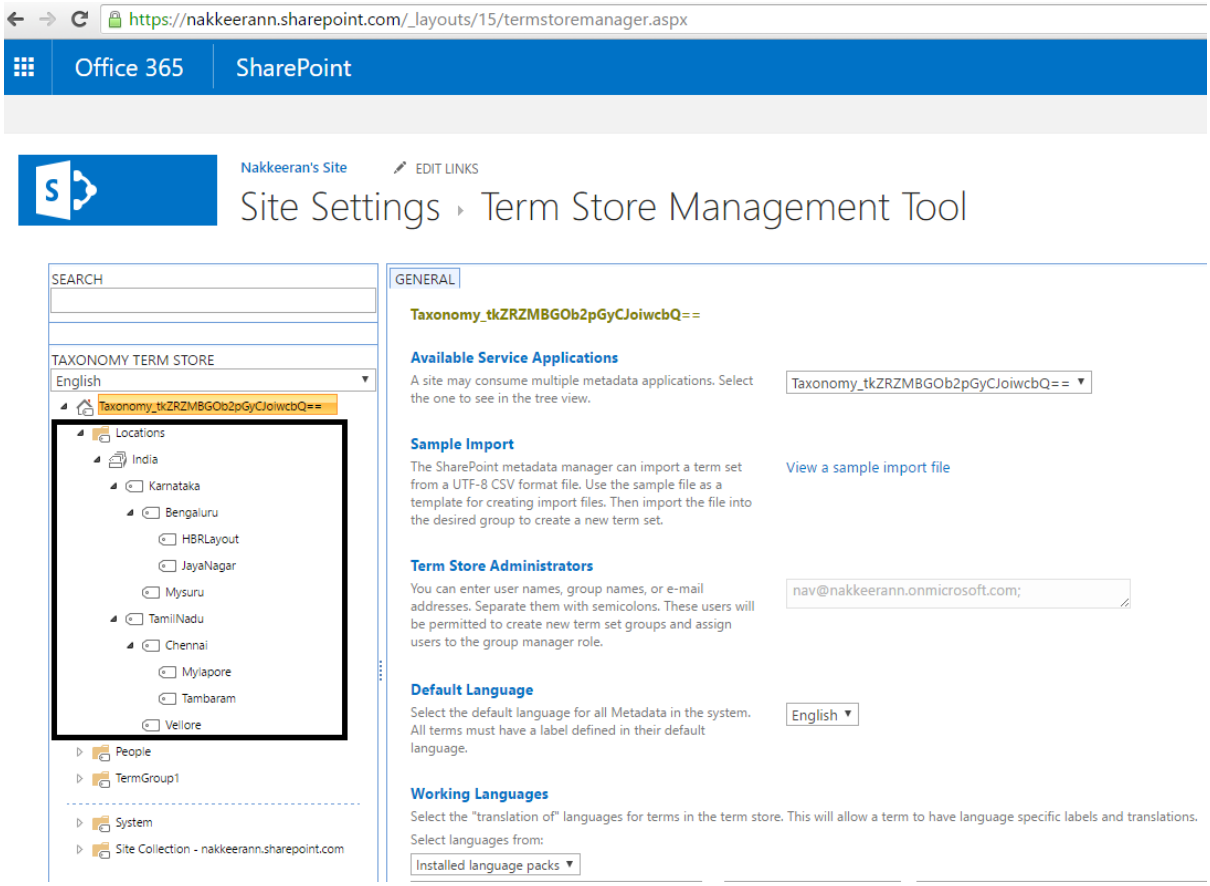
The following example explains in detail of importing a variable array value term stores in to Taxonomy term store on SharePoint site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Taxonomy\ImportArray.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to import term set array data in to SP Taxonomy
function ImportArrayTerms(){

   # Input Parameters
   # Term sets are arrays
   $termSet1 = 'Locations|India|Karnataka|Bengaluru|JayaNagar','Locations|India|Karnataka|Bengaluru|HBRLayout','Locations|India|Karnataka|Mysuru'
   $termSet2 = 'Locations|India|TamilNadu|Chennai|Tambaram','Locations|India|TamilNadu|Chennai|Mylapore','Locations|India|TamilNadu|Vellore'

   # Imports above taxonomy data array into SharePoint
   Import-SPOTaxonomy -Terms $termSet1
   Import-SPOTaxonomy -Terms $termSet2

   Write-Host "Terms Imported"
}

# Calls the Function
ImportArrayTerms # Imports terms array in to SharePoint site
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

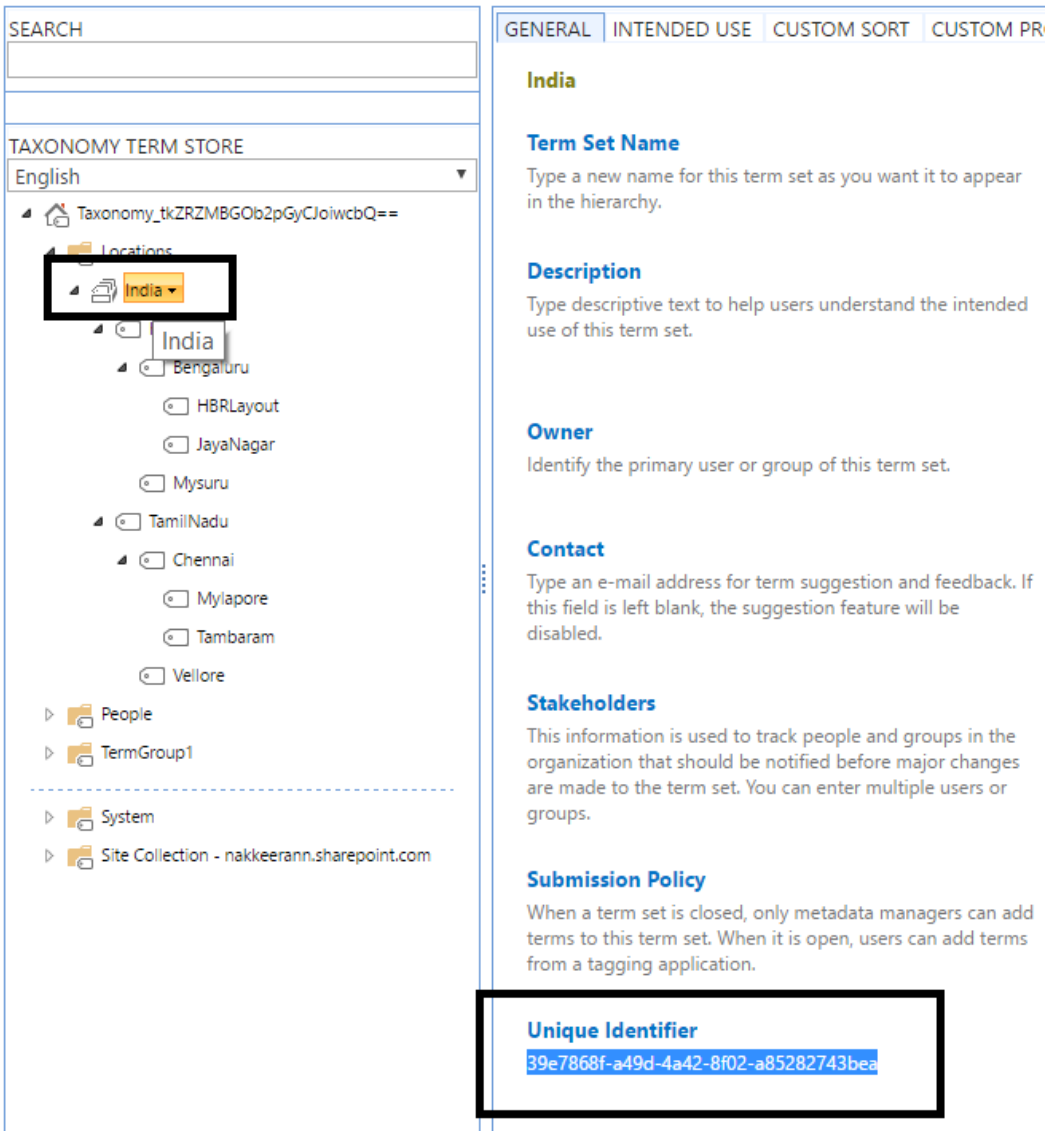
The term group is created along with term sets and terms on taxonomy term store. The following snapshot shows the term sets added to term group (Location) in the example.



14.4 Export term set to text file

In this section, you will learn how to export a term set collection from term store to a text file using PnP PowerShell script.

The following example explains in detail of exporting term set data from SharePoint term to local file path. The file path and term set ID are provided for exporting the terms. The term set id can be identified from the term store page. The following snapshot shows the term set with term id.



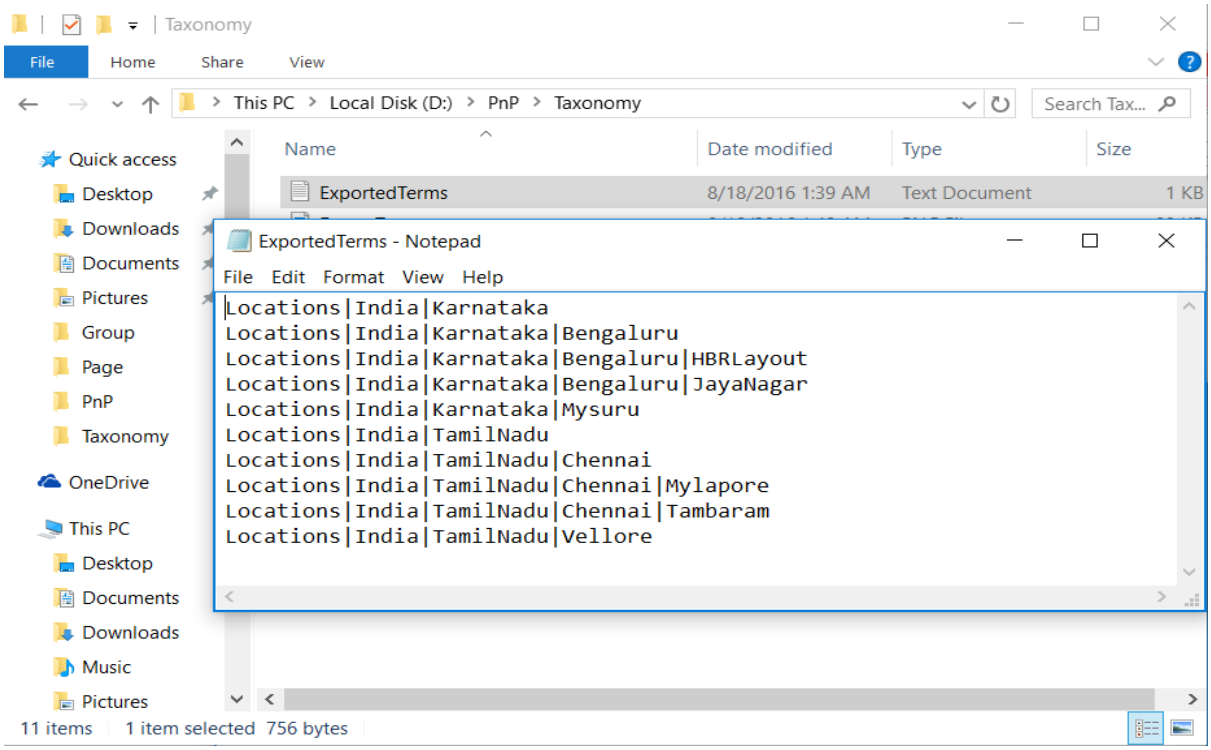
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Taxonomy\ExportText.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to export term set from Term Store to Local file
function ExportTerms(){

   # Input Parameters
   $filePath = "D:\PnP\Taxonomy\ExportedTerms.txt"

   # Exports taxonomy data from SP term store to local file system
   Export-SPOTaxonomy -Path $filePath -TermSetId "39e7868f-a49d-4a42-8f02-
a85282743bea"

   Write-Host "Terms Exported"
}

# Calls the Function
ExportTerms # Export terms to file
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The term set is successfully exported from term store to local text file.

The following snapshot shows the exported text file along with terms.



14.5 Export term set to XML

In this section, you will learn how to export a term set collection from term store to a XML file using PnP PowerShell script.

The following example explains in detail of exporting term set data from SharePoint term to local XML file. The file path and term set ID are provided for exporting the terms. The term set name can be taken from the term store page. In this example, we will take “Locations” term group created in the above examples.

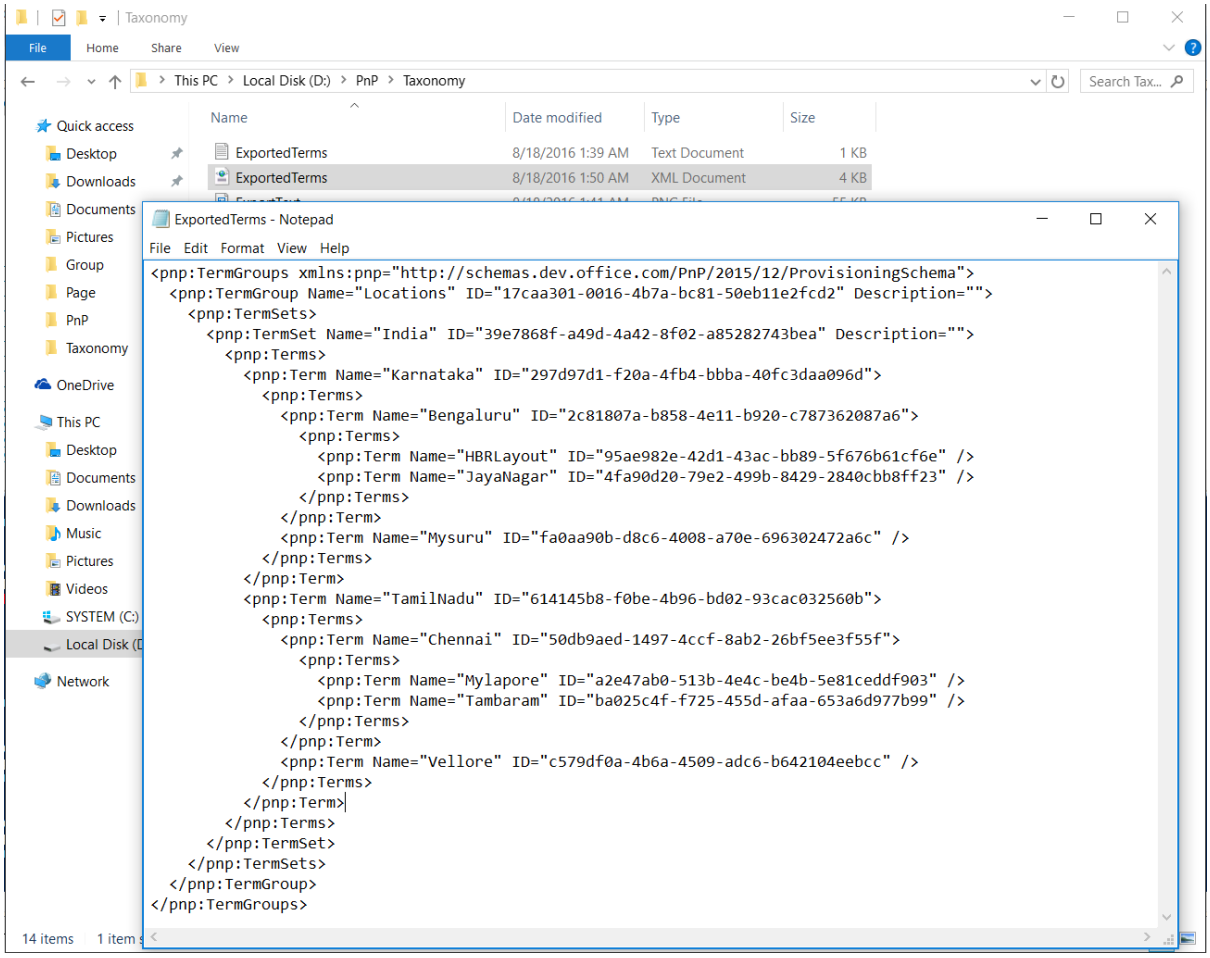
- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Taxonomy\ExportXML.ps1).

```
# Site Collection URL or Sub Site URL
$siteurl = "https://nakkeerann.sharepoint.com"
# User Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $siteurl -Credentials $credential

# Function to export term set from Term Store to Local XML file
function ExportTermsAsXML(){

   # Input Parameters
   $filePath = "D:\PnP\Taxonomy\ExportedTerms.xml"
   $termSetName = "Locations"

   # Exports taxonomy data from SP term store to local file system (XML)
   Export-SPOTermGroupToXml -Identity $termSetName -Out $filePath

   Write-Host "Terms Exported"
}

# Calls the Function
ExportTermsAsXML # Export terms to XML file
```

- Run the script from Debug menu or by pressing F5.
- Enter the username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The term set is successfully exported from term store to local XML file.

The following snapshot shows the exported XML file along with terms.



15 Office 365 Site Collection tasks using PnP PowerShell

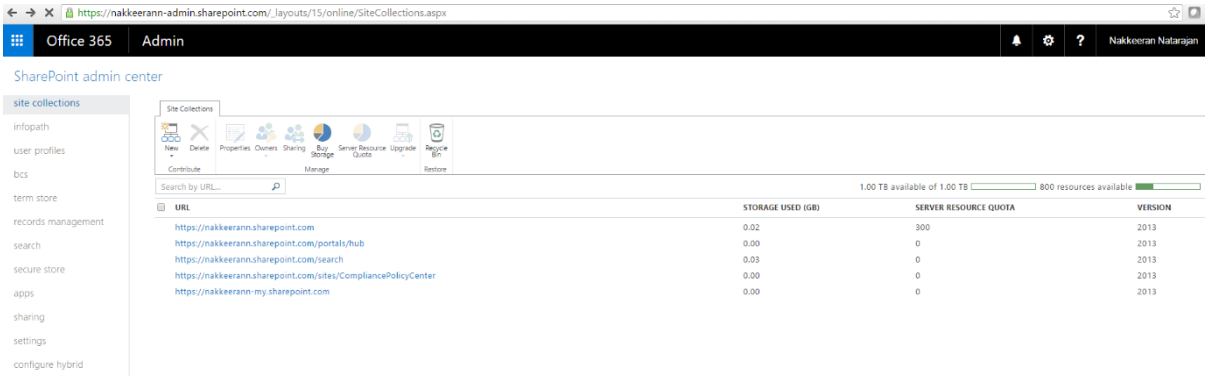
In this topic, you will learn how to perform operations on Office 365 tenant site using PnP PowerShell scripts. These operations can be used only on Office 365 sites.

The following cmdlets are majorly used for the operations in this topic.

- New-SPOTenantSite
- Get-SPOTenantSite
- Set-SPOTenantSite
- Remove-SPOTenantSite

The Office 365 tenant site will look like, https://nakkeerann-admin.sharepoint.com.

The site collections can be accessed from https://nakkeerann-admin.sharepoint.com/_layouts/15/online/SiteCollections.aspx. The following snapshot shows the site collections page.



15.1 How to create a site collection on tenant site

In this section, you will learn how to create site collection on Office 365 tenant site. New-SPOTenantSite command is used to create a site collection on the tenant site. The other properties like title, site collection URL, template id, resource & storage quotas, and owner information are passed as parameters along with the command.

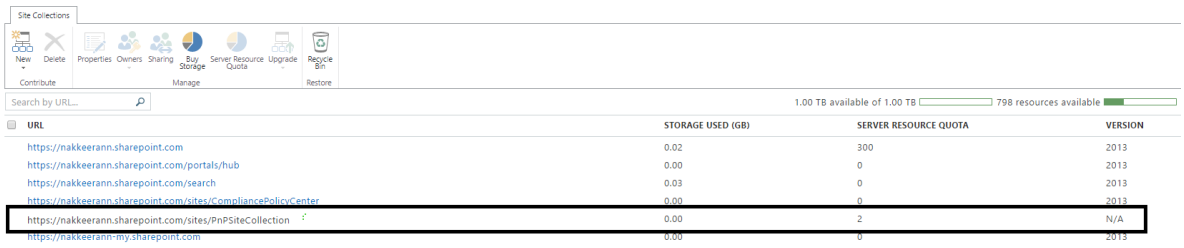
The following example steps helps creating a site collection.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Tenant\CreateSiteCollection.ps1).

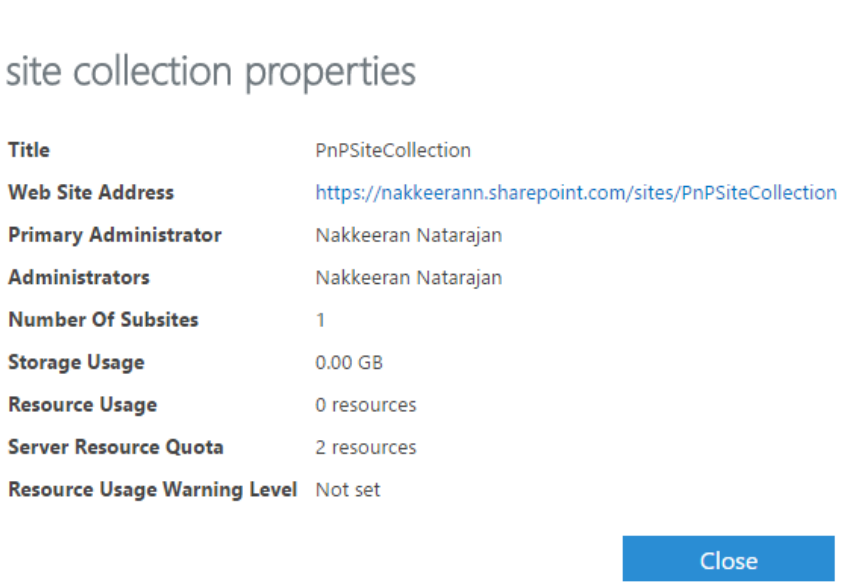
```
# Tenant Site Collection URL
$tenantSiteURL = "https://nakkeerann-admin.sharepoint.com"

# Tenant admin user Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $tenantSiteURL -Credentials $credential

# Function to create site collection on Office 365 tenant site
function CreateSiteCollection(){

   # Input Parameters
   $siteName = "PnPSiteCollection"
   $siteURL = "https://nakkeerann.sharepoint.com/sites/PnPSiteCollection"
   $siteDescription = "Site Collection Created By PnP PowerShell"
   $siteTemplate = "STS#0"
   $resources = 2
   $storage = 30
   $siteOwner = "nav@nakkeerann.onmicrosoft.com"
   $timeZone = 13

   # Creates site collection on o365 site with above inputs
   New-SPOTenantSite -Title $siteName -Url $siteURL -Description $siteDescription -Tem-
plate $siteTemplate -ResourceQuota $resources -StorageQuota $storage -Owner $site-
Owner -TimeZone $timeZone

   Write-Host "Site Collection Created"
}

# Calls the Function
CreateSiteCollection # Creates site collection on Office 365 tenant site
```

- Run the script from Debug menu or by pressing F5.
- Enter the tenant admin username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site collection is created successfully. The below snapshot shows the site collection which is getting provisioned. This process will take few mins.



The below snapshot shows the site collection information.



15.2 How to retrieve all site collections from tenant site

In this section, you will learn how to retrieve site collections available on tenant site using PnP PowerShell commands. Get-SPOTenantSite cmdlet is used for retrieving site collections from the tenant site.

The following example steps helps retrieving all site collections from tenant site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Tenant\RetrieveSiteCollections.ps1).

```
# Tenant Site Collection URL
$tenantSiteURL = "https://nakkeerann-admin.sharepoint.com"

# Tenant admin user Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $tenantSiteURL -Credentials $credential

# Function to retrieve site collections from Office 365 tenant site
function RetrieveSiteCollections(){

   # Retrieves site collections on o365 site
   $sites = Get-SPOTenantSite -Detailed -IncludeOneDriveSites -Force

   # Displays the site collections from tenant on the console
   Write-Host "There are " $sites.count " site collections present"
   foreach($site in $sites){
      Write-Host $site.Url " - " $site.Template
   }
}

# Calls the Function
RetrieveSiteCollections # Retrieves site collection from Office 365 tenant site
```

- Run the script from Debug menu or by pressing F5.
- Enter the tenant admin username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site collections are retrieved from the tenant site successfully. The below snapshot shows the site collections retrieved on the console.

```
PS C:\WINDOWS\system32> D:\PnP\Tenant\RetrieveSiteCollections.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
There are  7  site collections present
https://nakkeerann.sharepoint.com/  -  STS#0
https://nakkeerann.sharepoint.com/portals/hub  -  POINTPUBLISHINGHUB#0
https://nakkeerann.sharepoint.com/search  -  SRCHCEN#0
https://nakkeerann.sharepoint.com/sites/CompliancePolicyCenter  -  POLICYCTR#0
https://nakkeerann.sharepoint.com/sites/PnPSiteCollection  -  STS#0
https://nakkeerann-my.sharepoint.com/  -  SPSMSITEHOST#0
https://nakkeerann-my.sharepoint.com/personal/nav_nakkeerann_onmicrosoft_com  -  SPSPERS#10

PS C:\WINDOWS\system32>
```

15.3 How to retrieve a site collection from tenant site using URL

In this section, you will learn how to retrieve required site collection available on tenant site by using site collection URL with PnP PowerShell commands. Get-SPOTenantSite cmdlet is used for retrieving site collections from the tenant site. The site collection URL is passed as parameter to get the details.

The following example steps helps retrieving required site collection properties from tenant site.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Tenant\RetrieveSiteCollection.ps1).

```
# Tenant Site Collection URL
$tenantSiteURL = "https://nakkeerann-admin.sharepoint.com"

# Tenant admin user Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $tenantSiteURL -Credentials $credential

# Function to retrieve required site collection from Office 365 tenant site by site name
function RetrieveSiteCollection(){

   # Retrieves required site collection on o365 site using site name
   $site = Get-SPOTenantSite -Url "https://nakkeerann.sharepoint.com/sites/PnPSiteCollec-
tion" -Detailed

   # Displays site collection's properties
   Write-Host "Title      : " $site.Title
   Write-Host "URL        : " $site.Url
   Write-Host "Template   : " $site.Template
   Write-Host "Status     : " $site.Status
   Write-Host "Storage (MB): " $site.StorageMaximumLevel
   Write-Host "Used (MB)  : " $site.StorageUsage
   Write-Host "RESOURCES  : " $site.UserCodeMaximumLevel
   Write-Host "Owner      : " $site.Owner
   Write-Host "Sharing    : " $site.SharingCapability
   Write-Host "subsites   : " $site.WebsCount
}

# Calls the Function
RetrieveSiteCollection # Retrieves required site collection from Office 365 tenant site by site
name
```

- Run the script from Debug menu or by pressing F5.
- Enter the tenant admin username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site collection is retrieved from the tenant site successfully. The below snapshot shows the site collection properties retrieved on the console.

```
PS C:\WINDOWS\system32> D:\PnP\Tenant\RetrieveSiteCollection.ps1
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Title          :  PnPSiteCollection
URL            :  https://nakkeerann.sharepoint.com/sites/PnPSiteCollection
Template       :  STS#0
Status         :  Active
Storage (MB):  1048576
Used (MB)      :  2
RESOURCES      :  2
Owner          :  nav@nakkeerann.onmicrosoft.com
Sharing        :  Disabled
subsites       :  1

PS C:\WINDOWS\system32>
```

15.4 How to update a site collection on tenant site

In this section, you will learn how to update a site collection available on Office 365 tenant by using site collection URL with PnP PowerShell commands. Set-SPOTenantSite cmdlet is used for updating site collection properties. The site collection URL and other optional properties are passed as parameters to update the details.

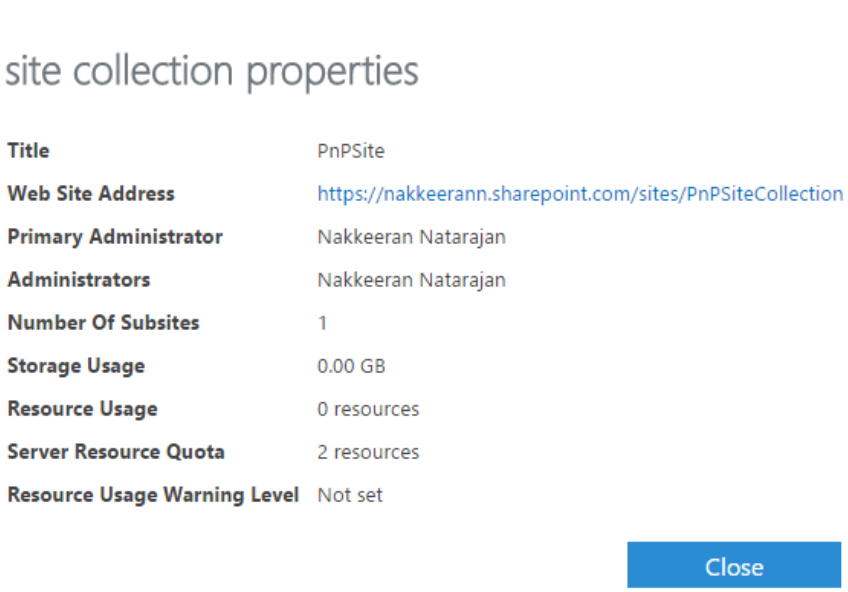
The following example steps helps updating required site collection properties.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script, and save the file (D:\PnP\Tenant\UpdateSiteCollection.ps1).

```
# Tenant Site Collection URL
$tenantSiteURL = "https://nakkeerann-admin.sharepoint.com"

# Tenant admin user Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $tenantSiteURL -Credentials $credential

# Function to update site collection properties on Office 365
function UpdateSiteCollection(){

   # Input parameters
   $siteURL = "https://nakkeerann.sharepoint.com/sites/PnPSiteCollection"
   $newTitle = "PnPSite"

   # Updates site collection
   Set-SPOTenantSite -Url $siteURL -Title $newTitle -Sharing ExistingExternalUserSharin-
gOnly

   Write-Host "Site Collection Properties Updated"
}

# Calls the Function
UpdateSiteCollection # Updates site collection properties on Office 365
```

- Run the script from Debug menu or by pressing F5.
- Enter the tenant admin username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site collection is updated successfully. The below snapshot shows the site collection properties updated. The site collection properties can be viewed from the site collections page of Office 365 tenant site.



15.5 How to delete a site collection from tenant site

In this section, you will learn how to delete a site collection from SharePoint Office 365 tenant by using site collection URL with PnP PowerShell commands. Remove-SPOTenantSite cmdlet is used for deleting site collection, using site collection URL as parameter.

The following example steps helps deleting the o365 site collection.

- Open Windows PowerShell ISE as administrator.
- Open a new file, paste the following script and save the file (D:\PnP\Tenant\RemoveSiteCollection.ps1).

```
# Tenant Site Collection URL
$tenantSiteURL = "https://nakkeerann-admin.sharepoint.com"

# Tenant admin user Credentials
$credential = Get-Credential

# Connects and Creates Context
Connect-SPOnline -Url $tenantSiteURL -Credentials $credential

# Function to remove site collection from SharePoint Office 365
function RemoveSiteCollection(){

   # Input parameters
   $siteURL = "https://nakkeerann.sharepoint.com/sites/PnPSiteCollection"
   $newTitle = "PnPSite"

   # Removes site collection
   Remove-SPOTenantSite -Url $siteURL -SkipRecycleBin -Force

   Write-Host "Site Collection Deleted"
}

# Calls the Function
RemoveSiteCollection # Removes site collection from SharePoint Office 365
```

- Run the script from Debug menu or by pressing F5.
- Enter the tenant admin username (nav@nakkeerann.sharepoint.com) and password (***) of SharePoint site in the credentials popup.

The site collection is deleted successfully from SharePoint O365 tenant site.

16 Summary

Thus, we have learnt programming on PnP PowerShell Script. The book covered all the basic operations required for accessing various objects on SharePoint instances. The advanced or complex code can be developed and executed from the above base samples. The operations explained in the above sections are compatible for SharePoint 2013, SharePoint 2016, and SharePoint online (Office 365) versions.

17 References

- https://github.com/OfficeDev/PnP-PowerShell