End-to-End Guide for SharePoint Token Creation and Site Management

Step 1: Create an App Registration in Azure AD (for Token Generation)

Sign in to Azure Portal:

Go to https://portal.azure.com

Navigate to "Azure Active Directory" > "App registrations" > "New registration"

Register a new application:

Give your app a name (e.g., "SharePoint Management App")

Select account types (usually "Accounts in this organizational directory only")

Set redirect URI (for web apps: https://yourdomain.com, for daemon apps: no redirect needed)

Click "Register"

Configure API permissions:

Go to "API permissions" > "Add a permission"

Select "SharePoint" > "Application permissions"

Add permissions like:

Sites.FullControl.All

User.ReadWrite.All

Click "Grant admin consent"

Create a client secret:

Go to "Certificates & secrets" > "New client secret"

Set expiration and click "Add"

Copy the secret value (you won't see it again)

Note your credentials:

Application (client) ID

Directory (tenant) ID

Client secret value

Step 2: Generate an Access Token

You can generate a token using PowerShell, C#, or Postman. Here's a PowerShell example:

powershell

$tenantId = "your-tenant-id"

$clientId = "your-client-id"

$clientSecret = "your-client-secret"

$tokenUrl = "https://login.microsoftonline.com/$tenantId/oauth2/v2.0/token"

$body = @{

client\_id = $clientId

scope = "https://graph.microsoft.com/.default"

client\_secret = $clientSecret

grant\_type = "client\_credentials"

}

$response = Invoke-RestMethod -Uri $tokenUrl -Method Post -Body $body

$accessToken = $response.access\_token

Write-Host "Access Token: $accessToken"

Step 3: Create a SharePoint Site

Using the token, you can create a SharePoint site via Microsoft Graph API:

powershell

$headers = @{

"Authorization" = "Bearer $accessToken"

"Content-Type" = "application/json"

}

$siteUrl = "https://graph.microsoft.com/v1.0/sites/root/lists"

# For a Team site

$body = @{

"displayName" = "My New Site"

"description" = "Site created via API"

"template" = "team site"

} | ConvertTo-Json

$response = Invoke-RestMethod -Uri $siteUrl -Method Post -Headers $headers -Body $body

$response | ConvertTo-Json -Depth 5

Step 4: Add Users to the Site

powershell

$siteId = "your-site-id" # From the previous response

$userId = "user@domain.com" # Or user object ID

$url = "https://graph.microsoft.com/v1.0/sites/$siteId/lists('permissions')/items"

$body = @{

"fields" = @{

"Title" = "Member"

"Permission" = "Edit"

"User" = $userId

}

} | ConvertTo-Json

Invoke-RestMethod -Uri $url -Method Post -Headers $headers -Body $body

Step 5: Remove Users from the Site

powershell

$permissionId = "permission-id-to-remove" # You'd need to get this first

$url = "https://graph.microsoft.com/v1.0/sites/$siteId/lists('permissions')/items/$permissionId"

Invoke-RestMethod -Uri $url -Method Delete -Headers $headers

Alternative: Using PnP PowerShell

For simpler SharePoint management:

Install PnP PowerShell:

powershell

Install-Module -Name PnP.PowerShell

Connect to SharePoint:

powershell

Connect-PnPOnline -Url https://yourtenant.sharepoint.com -ClientId $clientId -ClientSecret $clientSecret

Create a site:

powershell

New-PnPSite -Type TeamSite -Title "My Site" -Alias "mysite" -Description "My new site"

Add a user:

powershell

Add-PnPUserToGroup -LoginName "user@domain.com" -Identity "Members" -WebUrl "https://yourtenant.sharepoint.com/sites/mysite"

Remove a user:

powershell

Remove-PnPUserFromGroup -LoginName "user@domain.com" -Identity "Members" -WebUrl "https://yourtenant.sharepoint.com/sites/mysite"

Important Notes:

Always secure your client secrets

Follow the principle of least privilege when assigning permissions

Consider using certificate-based authentication for production

Audit all API activities

Handle token expiration (typically 60-90 minutes) by implementing token refresh logic