A screen shot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer screen

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

A black rectangle with white text

Description automatically generated

* Import-Module AzureAD:
  + This line imports the Azure Active Directory (Azure AD) module into the PowerShell session.
  + This module contains several cmdlets administrators and developers use to create, configure, and manage Azure AD resources in their directory.
* Import-Module MSOnline:
  + This line imports the MSOnline module, which provides cmdlets necessary to manage Office 365 and Azure Active Directory from PowerShell.
* $UserCredential = Get-Credential:
  + This command is used to get a credential object (username and password) from the user.
  + This information will be used to authenticate with Azure AD and MS Online.
* Connect-AzureAD -Credential $UserCredential:
  + This line uses the stored credentials from the variable $UserCredential to establish a connection to Azure Active Directory.
  + This connection is necessary for running cmdlets from the AzureAD module.
* Connect-MsolService -Credential $UserCredential:
  + Similarly, this line establishes a connection to Microsoft Online Services, which is needed for running cmdlets from the MSOnline module.

The following section from $userForm = New-Object System.Windows.Forms.Form through $userForm.Controls.Add($categoryDropdown) creates a simple form using Windows Forms, which allows for graphical user interface interaction with the user.

**The form contains three components:**

* Two TextBox objects for input of first name and last name ($firstNameBox, $lastNameBox).
* A ComboBox object ($categoryDropdown) for selecting the category of the user being created.
* It includes options for "AGM", "GM", "Franchise Users", "New Corporate Users", "Corporate SkyZone Users", and "SkyZone GMs".
* $okButton = New-Object System.Windows.Forms.Button...$userForm.Controls.Add($okButton):
  + This section creates an "OK" button for the form.
  + When clicked, it triggers an event that collects data from the form, generates a confirmation form, and runs code to create a new Azure AD user based on the inputted information.
* $okButton.Add\_Click({...}):
  + This section is the click event handler for the "OK" button.
  + It includes code to collect the data entered into the form and create a new user.
  + The entered first name, last name, and user category are stored in $firstName, $lastName, and $category, respectively.
* $confirmationForm = New-Object System.Windows.Forms.Form...$confirmationForm.Controls.Add($confirmationLabel):
  + These lines generate a confirmation form with a label displaying the data entered into the original form.
  + This confirmation step allows the user to verify the entered information before the script proceeds with creating a new Azure AD user.
* $okButton = New-Object System.Windows.Forms.Button...$confirmationForm.Close() and $newUser = New-AzureADUser -DisplayName $displayName -PasswordProfile $passwordProfile -UserPrincipalName $userPrincipalName -AccountEnabled $true -MailNickName $userPrincipalName -UsageLocation US -OtherMails $userPrincipalName -ImmutableId $userPrincipalName:
  + These sections create a new "OK" button for the confirmation form and define what happens when the button is clicked.
  + If the user confirms the details, the script creates a new Azure AD user with the provided details.
* $okButton.Add\_Click({...}):
  + This section is another click event handler for the "OK" button.
  + This code is run if the user clicks "OK" on the confirmation form.
  + It sets user properties based on the user category selected, prompts another confirmation, and finally creates the user in Azure AD.
    - The if statement inside this click event (if ($category -eq "AGM") {...}) is used to assign different properties to the new user depending on their role.
    - It assigns job title, department, office, usage location, and license depending on the selected category.
* New-MsolUser -UserPrincipalName $email -DisplayName $displayName -FirstName $firstName -LastName $lastName -Password (ConvertTo-SecureString -AsPlainText "P@ssw0rd" -Force) -ForceChangePassword $true:
  + This cmdlet creates a new user in Azure Active Directory and sets the initial password to "P@ssw0rd", which the user will be forced to change on their next login.
* Set-MsolUser -UserPrincipalName $email -Title $userProperties.title -Department $userProperties.department -Office $userProperties.office -UsageLocation $userProperties.usagelocation:
  + This cmdlet sets the new user's specified properties (title, department, office, and usage location).
* Set-MsolUserLicense -UserPrincipalName $email -AddLicenses $userProperties.license:
  + This cmdlet assigns the specified license to the new user.
* Write-Host "User created with email: $email":
  + This line prints a message to the console stating that the user was created and providing the user's email.
* $noButton = New-Object System.Windows.Forms.Button...$confirmationForm.Controls.Add($noButton):
  + This section creates a "No" button for the confirmation form.
  + If clicked, it resets the original form and re-displays it.
* function Generate-RandomPassword {...}:
  + This function generates a random password of a specified length using a given character set.
  + It isn't used in the script provided but can be used to generate random passwords for new users.
* [void]$userForm.ShowDialog():
  + Finally, this line displays the form built earlier in the script.
  + This form collects the information from the administrator to create the new Azure AD user.