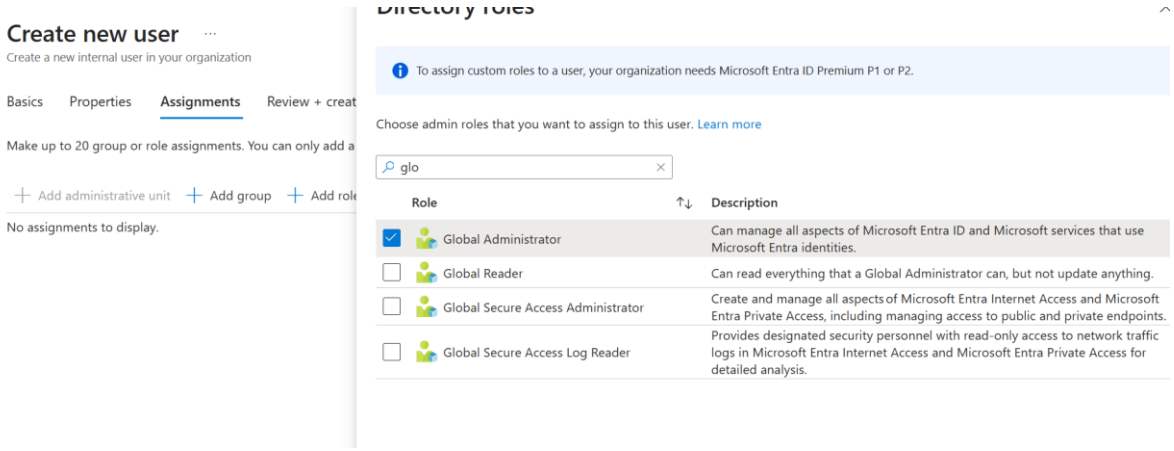
Lab 1 (Creating and assign users and groups)

Part I. Create emergency “break-glass” or emergency account

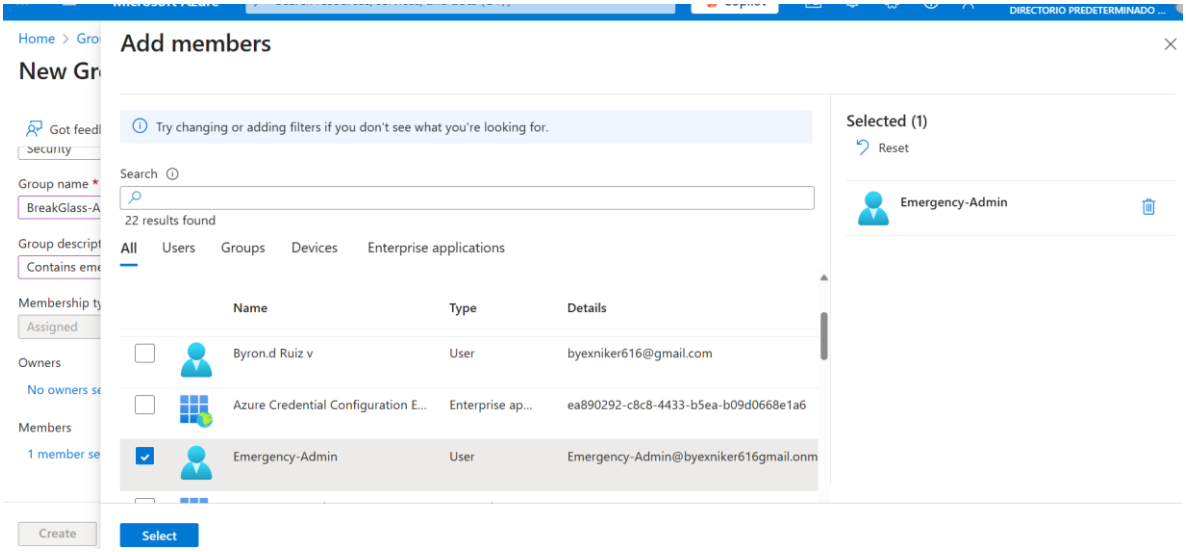
- Entra admin center → **Identity > Users > New user**.
- Name: Emergency-Admin
- Strong 24+ char password; store securely (dEXx9vRIp6M459txmFSpWLo5)

Since this is a backup account for admin, this must have Global admin permissions later we will protect it with PIM



Part II.

Create a group to add the new “break-glass account”

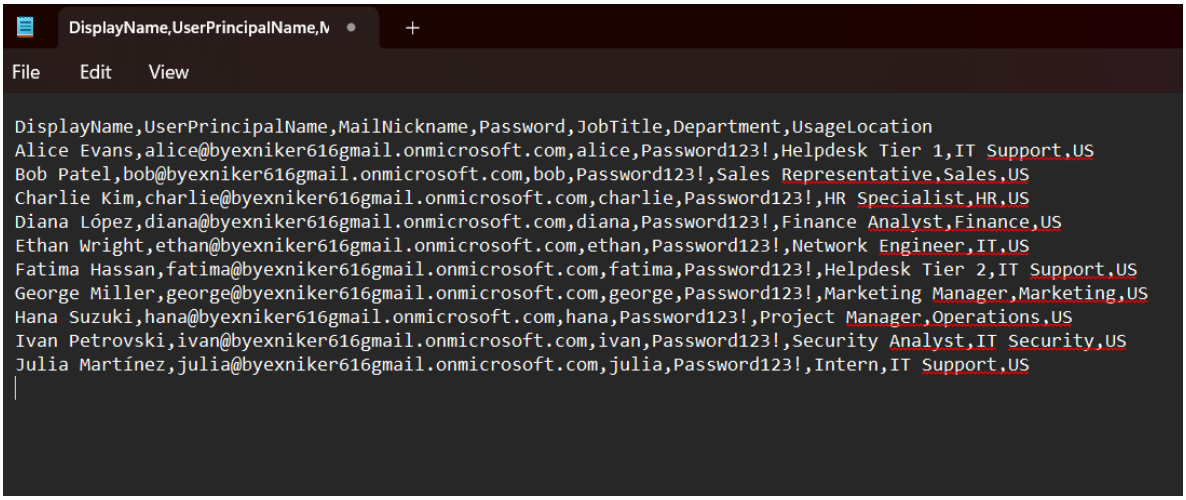


This group must be excluded from MFA look down to serve as backup in case of failure

Part III.

Once we have created the first user and group, we can continue with the creation of more users. We let's use a "CSV file "to make faster the integration of several users at same time.

First, we'll need a "CSV" file with all user and parameters for each one, in this case we'll use notepad for creating this file.



Once we have the file stored in our designated path, we'll use power shell commands to execute our recent create csv file,

```
PS C:\Users\user> $users = import-csv -path 'C:\Users\user\Documents\Azure Lab\UsersTest.csv'
PS C:\Users\user>
```

Then verify the correct creation with the following command to see all users exist in the tenant.

```
PS C:\Users\user> $users | Format-Table

DisplayName    UserPrincipalName                       MailNickname Password     JobTitle            Department  Usa
                                                                                                               gel
                                                                                                               oca
                                                                                                               tio
                                                                                                               n
-----------    -----------------                       ------------ --------     --------            ----------  ---
Alice Evans    alice@byexniker616gmail.onmicrosoft.com   alice        Password123! Helpdesk Tier 1     IT Support  US
Bob Patel      bob@byexniker616gmail.onmicrosoft.com     bob          Password123! Sales Representative Sales      US
Charlie Kim    charlie@byexniker616gmail.onmicrosoft.com charlie      Password123! HR Specialist       HR          US
Diana López    diana@byexniker616gmail.onmicrosoft.com   diana        Password123! Finance Analyst     Finance     US
Ethan Wright   ethan@byexniker616gmail.onmicrosoft.com   ethan        Password123! Network Engineer    IT          US
Fatima Hassan  fatima@byexniker616gmail.onmicrosoft.com  fatima       Password123! Helpdesk Tier 2     IT Support  US
George Miller  george@byexniker616gmail.onmicrosoft.com  george       Password123! Marketing Manager   Marketing   US
Hana Suzuki    hana@byexniker616gmail.onmicrosoft.com    hana         Password123! Project Manager     Operations  US
Ivan Petrovski ivan@byexniker616gmail.onmicrosoft.com    ivan         Password123! Security Analyst    IT Security US
Julia Martínez julia@byexniker616gmail.onmicrosoft.com   julia        Password123! Intern              IT Support  US
```

Part IV.

Create groups to assign users for each role.

```
PS C:\Users\user> # Grupo Helpdesk
PS C:\Users\user> New-MgGroup `
>>   -DisplayName "Lab-Helpdesk" `
>>   -MailEnabled:$false `
>>   -MailNickname "lab-helpdesk" `
>>   -SecurityEnabled:$true

DisplayName  Id                                   MailNickname Description GroupTypes
-----------  --                                   ------------ ----------- ----------
Lab-Helpdesk 79af3ab8-875d-435f-b969-89ee7ab61e3f lab-helpdesk             {}
```

Then add user for each role group

```
PS C:\Users\user> New-MgGroupMember -GroupId $group.Id -DirectoryObjectId $user.Id
PS C:\Users\user> $group = Get-MgGroup -Filter "DisplayName eq 'Lab-Helpdesk'"
PS C:\Users\user> $user  = Get-MgUser -UserId "alice@byexniker616gmail.onmicrosoft.com"
PS C:\Users\user>
PS C:\Users\user> New-MgGroupMember -GroupId $group.Id -DirectoryObjectId $user.Id
PS C:\Users\user>
```

Part V.

Now we'll assign each user to respective group, let's use job tittle as a criteria for this example

```
PS C:\Users\user> Get-MgUser -all | select DisplayName, Department, JobTitle

DisplayName     Department JobTitle
-----------     ---------- --------
Alice                      it
Bob Patel                  Sales Representative
Byron.d Ruiz v
Charlie Kim                HR Specialist
Diana Lopez                Finance Analyst
Emergency-Admin            Break-Glass Account
Ethan Wright               Network Engineer
Fatima Hassan              Helpdesk Tier 2
George Miller              Marketing Manager
Hana Suzuki                Project Manager
Ivan Petrovski             Security Analyst
Julia MartÃ-nez            Intern
Test User 01


PS C:\Users\user> Get-MgGroup -all

DisplayName  Id                                   MailNickname Description                              GroupTypes
-----------  --                                   ------------ -----------                              ----------
Lab-HR       554f5dff-eb25-4236-b013-ad17713b0280 lab-hr                                             {}
Break-glass  5d13f5a1-dc59-4102-b8db-20dd858e0f7a 355792dd-9   group exclusive for glass-break porpose {}
Lab-Helpdesk 79af3ab8-875d-435f-b969-89ee7ab61e3f lab-helpdesk                                         {}
Lab-Sales    970b27d8-74b1-4ea4-a274-57397cfb558c lab-sales                                            {}
```

Let's review their attributes with the commands above.

Then now we create a mapping that's fits with the group.

And then we use power shell script to add to group

```
PS C:\Users\user> # ----------------------------
PS C:\Users\user> # JobTitle to GroupId mapping
PS C:\Users\user> # ----------------------------
PS C:\Users\user> $groupMap = @{
>>     "sales specialyst"      = "970b27d8-74b1-4ea4-a274-57397cfb558c" # Lab-Sales
>>     "sales representative"  = "970b27d8-74b1-4ea4-a274-57397cfb558c" # Lab-Sales
>>     "sales manager"         = "970b27d8-74b1-4ea4-a274-57397cfb558c" # Lab-Sales
>>     "sales intern"          = "970b27d8-74b1-4ea4-a274-57397cfb558c" # Lab-Sales
>>
>>     "hr specialist"         = "554f5dff-eb25-4236-b013-ad17713b0280" # Lab-HR
>>     "hr manager"            = "554f5dff-eb25-4236-b013-ad17713b0280" # Lab-HR
>>     "hr analyst"            = "554f5dff-eb25-4236-b013-ad17713b0280" # Lab-HR
>>
>>     "helpdesk analyst"      = "79af3ab8-875d-435f-b969-89ee7ab61e3f" # Lab-Helpdesk
>>     "helpdesk engineer"     = "79af3ab8-875d-435f-b969-89ee7ab61e3f" # Lab-Helpdesk
>>     "helpdesk tier 2"       = "79af3ab8-875d-435f-b969-89ee7ab61e3f" # Lab-Helpdesk
>>     "intern helpdesk"       = "79af3ab8-875d-435f-b969-89ee7ab61e3f" # Lab-Helpdesk
>>
>>     "break-glass account"   = "5d13f5a1-dc59-4102-b8db-20dd858e0f7a" # Break-glass
>> }
```

And now very if the users were assigning correctly

For this task we can use the following script

```
“# Get all groups

$groups = Get-MgGroup -All

foreach ($group in $groups) {

  Write-Host "Group: $($group.DisplayName) ($($group.Id))" -ForegroundColor Green

  # Get members of the group

  $members = Get-MgGroupMember -GroupId $group.Id

  if ($members) {

    foreach ($member in $members) {

      # Get additional details for each member

      $memberDetails = Get-MgDirectoryObjectById -Ids $member.Id

      Write-Host "  - $($memberDetails.AdditionalProperties['displayName'])" -ForegroundColor Yellow

    }

  } else {

    Write-Host "  No members" -ForegroundColor Gray

  }

  Write-Host ""

}”
```

```
PS C:\Users\user> foreach ($group in $groups) {
>>     Write-Host "Group: $($group.DisplayName) ($($group.Id))" -ForegroundColor Green
>>
>>     # Get members of the group
>>     $members = Get-MgGroupMember -GroupId $group.Id
>>
>>     if ($members) {
>>         foreach ($member in $members) {
>>             # Get additional details for each member
>>             $memberDetails = Get-MgDirectoryObjectById -Ids $member.Id
>>             Write-Host "  - $($memberDetails.AdditionalProperties['displayName'])" -ForegroundColor Yellow
>>         }
>>     } else {
>>         Write-Host "  No members" -ForegroundColor Gray
>>     }
>>     Write-Host ""
>> }
Group: Lab-HR (554f5dff-eb25-4236-b013-ad17713b0280)
  - Alice
  - Charlie Kim
  - Ethan Wright
  - Diana Lopez
  - Julia MartÃ-nez
  - George Miller
  - Hana Suzuki
  - Ivan Petrovski

Group: Break-glass  (5d13f5a1-dc59-4102-b8db-20dd858e0f7a)
  - Emergency-Admin

Group: Lab-Helpdesk (79af3ab8-875d-435f-b969-89ee7ab61e3f)
  - Alice
  - Ethan Wright
  - Diana Lopez
  - Fatima Hassan
  - Julia MartÃ-nez

Group: Lab-Sales (970b27d8-74b1-4ea4-a274-57397cfb558c)
  - Alice
  - Test User 01
  - Bob Patel
  - George Miller
```