**3. ⁠Terminate users based on update in CSV file to AD:-**

1. we need to add AccountExpirationDate to the CSV file.

2.Then we need to add the powershell script for the existing and below is the code .

**CODE:-**

# Import Active Directory module for running AD cmdlets

Import-Module ActiveDirectory

# Store the data from the updated CSV in the $ADUsers variable

$ADUsers = Import-Csv "Downloads\NewUsersFinal.csv" -Delimiter ";"

# Define UPN (User Principal Name)

$UPN = "megthink.com"

# Loop through each row containing user details in the CSV file

foreach ($User in $ADUsers) {

try {

# Handle the AccountExpirationDate: If it's empty, set it to $null

$AccountExpirationDate = if ($User.AccountExpirationDate -and $User.AccountExpirationDate.Trim()) {

# Parse the expiration date from the CSV (ensure it's a valid DateTime)

try {

[datetime]::Parse($User.AccountExpirationDate)

}

catch {

Write-Host "Invalid expiration date for user $($User.Username), skipping." -ForegroundColor Red

$null

}

} else {

$null

}

# Define the parameters using a hashtable for user creation

$UserParams = @{

SamAccountName = $User.Username

UserPrincipalName = "$($User.Username)@$UPN"

Name = "$($User.FirstName) $($User.LastName)"

GivenName = $User.FirstName

Surname = $User.LastName

Enabled = $True

DisplayName = "$($User.FirstName) $($User.LastName)"

Path = $User.OU

State = $User.State

EmailAddress = $User.Email

Department = $User.Department

AccountPassword = (ConvertTo-SecureString $User.Password -AsPlainText -Force)

ChangePasswordAtLogon = $True

}

# Check if the user already exists in AD

$ExistingUser = Get-ADUser -Filter "SamAccountName -eq '$($User.Username)'"

if ($ExistingUser) {

# User exists, handle expiration date update or removal logic

if ($AccountExpirationDate) {

# If expiration date is provided, check if it has passed

$CurrentDate = Get-Date

if ($AccountExpirationDate -lt $CurrentDate) {

# If the expiration date has passed, disable the account

disable-ADAccount -Identity $ExistingUser

Write-Host "Account for $($User.Username) has expired and is now disabled." -ForegroundColor Red

# Optionally, if you want to delete the user account, uncomment the next line:

Remove-ADUser -Identity $ExistingUser

Write-Host "User $($User.Username) has been deleted due to expiration." -ForegroundColor Red

}

else {

# If the expiration date is in the future, set the expiration date

Set-ADUser -Identity $User.Username -AccountExpirationDate $AccountExpirationDate

Write-Host "Account expiration date set for user $($User.Username)." -ForegroundColor Green

}

}

else {

# No expiration date is provided, keep the account active

Write-Host "No expiration date provided for user $($User.Username), account remains active." -ForegroundColor Green

}

}

else {

# User does not exist, proceed to create the new user account

New-ADUser @UserParams

Write-Host "The user $($User.Username) is created." -ForegroundColor Green

}

}

catch {

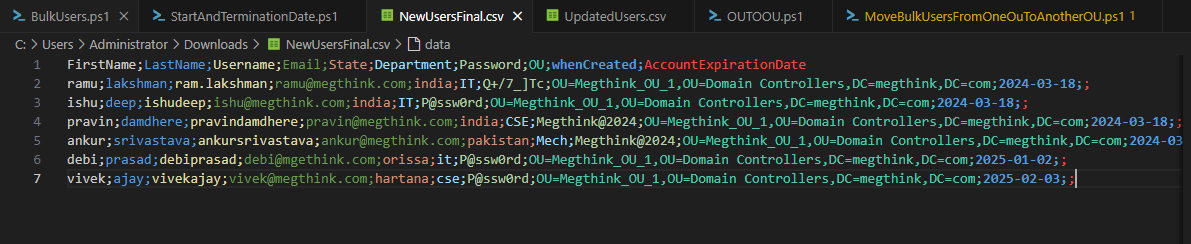
# Handle any errors that occur during account creation or updating

Write-Host "Failed to create or update user $($User.Username) - $\_" -ForegroundColor Red

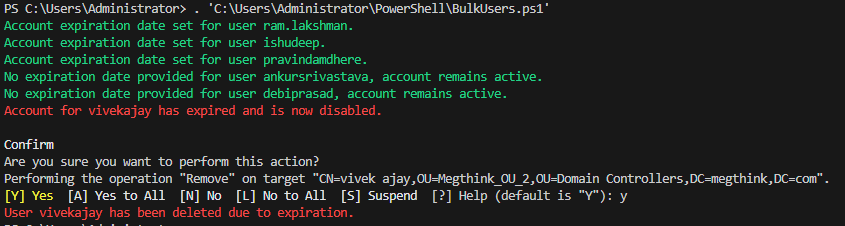
}

}

CSV FILE:-



Output ➖



**Uml:-**

