USER REBOOT NOTIFICATION

EXCEL VBA CODE

Marissa Brand, 2024

Sub RebootNotif ()

' ORIGINAL CODE BY MARISSA J BRAND

' The purpose of this VBA script is to take information our company receives regarding user workstation status, find which users require a reboot on their workstation, and alert them that an update/installment is needed via Outlook email.

```
' Declare Variables
    Dim wsCombined As Worksheet
    Dim combinedWs As Worksheet
    Dim lastRow As Long
    Dim existingWsName As String
    Dim firstDataRow As Long
    Dim rng As Range
    Dim tbl As ListObject
    Dim i As Long
    Dim WkStnName As String
    Dim matchedUser As String
    Dim savePath As String
    Dim ws As Worksheet
    Dim attachmentAdded As Boolean
    Dim existingWorkbook As Workbook
    Dim OutlookApp As Object
    Dim OutlookMailItem As Object
    Dim recipientEmail As String

' Name the existing worksheet where data will be combined as “Combined Data”
    existingWsName = "Combined Data"

' Check if the existing worksheet exists in the workbook
    On Error Resume Next
    Set combinedWs = ThisWorkbook.Worksheets(existingWsName)
    On Error GoTo 0
' If worksheet doesn't exist, create it
    If combinedWs Is Nothing Then
        Set combinedWs = ThisWorkbook.Worksheets.Add
        combinedWs.Name = existingWsName
    End If

' Start pasting data from cell A1
    lastRow = 1

' This next section of code runs through the workbook and combines the worksheets with the same format to be in “Combined Data”, and copies it all over neatly into one sheet
    For Each wsCombined In ThisWorkbook.Worksheets
        If wsCombined.Name <> existingWsName Then ' Skip the existing combined worksheet itself
' Copy data from current worksheet to the combined worksheet
            wsCombined.UsedRange.Copy
            combinedWs.Cells(lastRow, 1).PasteSpecial Paste:=xlPasteValues
            Application.CutCopyMode = False
' Move to the next available row in the combined worksheet
```

```
            lastRow = lastRow + wsCombined.UsedRange.Rows.Count

' Delete the current worksheet
            Application.DisplayAlerts = False 'Hide the delete confirmation dialog
            wsCombined.Delete
            Application.DisplayAlerts = True
        End If
    Next wsCombined

' Autofit the columns in the combined worksheet
    combinedWs.Columns.AutoFit

' Find the first row with data in the combined worksheet
    On Error Resume Next
    firstDataRow = combinedWs.Cells.Find(What:="*", _
                        SearchOrder:=xlByRows, _
                        SearchDirection:=xlNext).Row
    On Error GoTo 0

' Delete any blank rows above the first line of data
    If firstDataRow > 1 Then
        combinedWs.Rows("1:" & firstDataRow - 1).Delete
    End If

' Delete duplicate rows based on all columns in the combined worksheet, but keep the first occurrence of duplicates
    If combinedWs.Cells(1, 1).CurrentRegion.Rows.Count > 1 Then
        Dim lastCol As Long
        lastCol = combinedWs.Cells(1, combinedWs.Columns.Count).End(xlToLeft).Column
        combinedWs.Range("A2", combinedWs.Cells(combinedWs.Rows.Count, lastCol).End(xlUp)).RemoveDuplicates
Columns:=Array(1, 2, 3, 4, 5, 6), Header:=xlYes
    End If

' Delete rows where "Installed" is found in column [THE LETTER OF THE COLUMN THE REBOOT STATUS OF
WORKSTATIONS IS CURRENTLY DISPLAYED + 1 COLUMN TO THE RIGHT].
' The column it is currently displayed in will not be the same when the code finishes running due to the next section.
    lastRow = combinedWs.Cells(combinedWs.Rows.Count, "E").End(xlUp).Row
    For i = lastRow To 2 Step -1 ' Start from the last row and move up
        If combinedWs.Cells(i, "E").Value = "Installed" Or combinedWs.Cells(i, "A").Value = "Computer Name" Then
            combinedWs.Rows(i).Delete
        End If
    Next i

' Insert a column to the right of Column A with the header "Last Known User"
    combinedWs.Columns("B:B").Insert Shift:=xlToRight
    combinedWs.Cells(1, 2).Value = "Last Known User"
' Autofit the "Last Known User" column
    combinedWs.Columns("B").AutoFit

' Define the range of the table
    Set rng = combinedWs.Range("A1").CurrentRegion

' Create a table (ListObject) from the range
    Set tbl = combinedWs.ListObjects.Add(xlSrcRange, rng, , xlYes)
    tbl.Name = "Table1"
```

```
    tbl.TableStyle = "TableStyleLight13"
    tbl.AutoFilter.ShowAllData

' Find the last row with data in Column A
    lastRow = combinedWs.Cells(combinedWs.Rows.Count, "A").End(xlUp).Row

' Loop through each cell in Column A, start from 2 to skip header row
    For i = 2 To lastRow
' Get the value in Column A (workstation name)
        WkStnName = combinedWs.Cells(i, 1).Value

' Match workstation name to user - add as many workstations and last known users as you require using Select Case
        Select Case WkStnName
 Case "[WORKSTATION NAME]"
                matchedUser = "[LAST KNOWN WORKSTATION USER'S MAILBOX/USERNAME]"
 Case "[WORKSTATION NAME]"
                matchedUser = "[LAST KNOWN WORKSTATION USER'S MAILBOX/USERNAME]"
 Case "[WORKSTATION NAME]"
                matchedUser = "[LAST KNOWN WORKSTATION USER'S MAILBOX/USERNAME]"
 Case "[WORKSTATION NAME]"
                matchedUser = "[LAST KNOWN WORKSTATION USER'S MAILBOX/USERNAME]"
 Case "[WORKSTATION NAME]"
                matchedUser = "[LAST KNOWN WORKSTATION USER'S MAILBOX/USERNAME]"
            Case Else
                matchedUser = "Undetermined"
' If a user cannot be found while the code is running, the text string will read as "Undetermined" in the table
        End Select

' Inserts the matched workstation user into Column B
        combinedWs.Cells(i, 2).Value = matchedUser
    Next i

' Delete rows where "Undetermined" is found in column B.
' IMPORTANT TO NOTE: workstations that are not found to have a known user will not receive an email as they will be deleted from the table when the code runs to send an email based on the list of usernames generated.
    lastRow = combinedWs.Cells(combinedWs.Rows.Count, "B").End(xlUp).Row
    For i = lastRow To 2 Step -1 ' Start from the last row and move upwards
        If combinedWs.Cells(i, "B").Value = "Installed" Or combinedWs.Cells(i, "B").Value = "Undetermined" Then
            combinedWs.Rows(i).Delete
        End If
    Next i

' Set the active worksheet
    Set ws = ActiveSheet

' Specify the save path for the PDF
    savePath = "[INSERT SAVE PATH]"

' Set worksheet orientation to landscape
    ws.PageSetup.Orientation = xlLandscape

' Set worksheet to fit to 1 page wide and 1 page tall
    ws.PageSetup.FitToPagesWide = 1
    ws.PageSetup.FitToPagesTall = 1
```

```
' Export the active sheet as PDF
  ws.ExportAsFixedFormat Type:=xlTypePDF, Filename:=savePath, Quality:=xlQualityStandard

' Set the active worksheet
  Set ws = ActiveSheet

' Save path for the PDF
  savePath = "[INSERT SAVE PATH]"

' Set worksheet orientation to landscape
  ws.PageSetup.Orientation = xlLandscape
' Set worksheet to fit to 1 page wide and 1 page tall
  ws.PageSetup.FitToPagesWide = 1
  ws.PageSetup.FitToPagesTall = 1

' Export the active sheet as PDF
  ws.ExportAsFixedFormat Type:=xlTypePDF, Filename:=savePath, Quality:=xlQualityStandard

' Notify user upon successful export
  MsgBox "PDF file saved to [SAVED LOCATION]", vbInformation

' Send email to each recipient in Column B (BCC), find last row with data in Column B
  lastRow = ws.Cells(ws.Rows.Count, "B").End(xlUp).Row

' Create Outlook application object
  Set OutlookApp = CreateObject("Outlook.Application")

' Create new email
  Set OutlookMailItem = OutlookApp.CreateItem(0)

' Construct BCC recipients
  For i = 2 To lastRow ' Assuming data starts from row 2 and row 1 is header
    recipientEmail = ws.Cells(i, "B").Value & "@[EMAIL DOMAIN e.g. GMAIL].com"

' Add each recipient to BCC field
    If Len(OutlookMailItem.BCC) > 0 Then
      OutlookMailItem.BCC = OutlookMailItem.BCC & "; " & recipientEmail
    Else
      OutlookMailItem.BCC = recipientEmail
    End If
  Next i

' Compose the email
  With OutlookMailItem
        .Subject = "Workstation Reboot Needed"
        .Body = "If you have received this email, it is necessary that you reboot your computer as soon as possible. Thank
        you!"

' Display the email for review before sending
    .Display
    '.Send   ' without this line of code, the email would send automatically
  End With
```

```
' Clean up after the code has performed its primary task
    Set OutlookMailItem = Nothing
    Set OutlookApp = Nothing

' Notify user that emails have been sent
    MsgBox "Process Complete.", vbInformation
```

End Sub