USER & LICENSE INVENTORY ALLOCATION

EXCEL VBA CODE & INFORMATION

Marissa Brand, 2024

EXCEL WORKBOOK FILE FUNCTION & IMPORTANCE

This workbook should be saved as "**macro-enabled**", which means the excel file is capable of supporting the code pre-entered into Visual Basic, within the Excel application.

Visual Basic Application (VBA) is a programming language used within Microsoft applications, in this case, Excel.

VBA contains subroutines of code, or Macros, that perform different actions within the workbook, such as data organization and analysis. These subroutines are located within the Project Explorer window of the "Developer" tab in an Excel workbook.

The function of this file is to outline the process of assigning technological licenses to employees and users, and to automate it to the greatest extent possible. Using these macros within the workbook will modify user and license allocation, while simultaneously updating inventory figures within the configured datatable.

INTERACTING WITH THE USERS & LICENSES ("U&L") EXCEL FILE

The file must be saved as an "Excel Macro-Enabled Workbook", or the code will not run.

1. After saving the workbook as an "Excel Macro-Enabled Workbook". If it is not already, the "Developer" tab will need to be added to the ribbon displayed at the top of the Application. To do this, click; "File" > "Options" > "Customize Ribbon". Within the "Main Tabs" pop-up window, "Developer" will appear as an option next to a checkbox. Select the box and hit "OK".

2. The name of each worksheet is equivalent to its function; adding users, replacing users, and removing users all from the datatable on the "Users & Licenses" worksheet.

3. The "User & Licenses" worksheet within the workbook contains all information. The table in the worksheet displays the names of the users that have been entered, as well as an "x", or multiple, across the row, to indicate which license(s) that user is tied to. At the far right of the table, the number of licenses per user shows. Below the table, the total number of each license, how many are in-use, and the availability on-hand can be found. This worksheet should always be protected to ensure no unnecessary changes are made to the datatable.

The "Users & Licenses" Worksheet contains the following information:

- *Names of Licenses*
- *Names of Users (as they are added)*
- *What Licenses are tied to each User*
- *Number of Licenses per User*
- *Total number of Active Users*
- *Number of Active Users per License*
- *Total number of Licenses*
- *Number of Licenses per type*
- *Total number of licenses available*
- *Number of Licenses available per type*

4. In the "Users & Licenses" worksheet, in cells **B4:G4**, enter the number of licenses on-site, *whether in-use or simply on-hand*, per each type of license.

5. In order for the workbook to maintain its authenticity, it is necessary to protect each worksheet individually. This allows for changes to be made only to pre-selected cells within each sheet. It locks the value of cells that either don't need to be changed, or will automatically change regardless. Protecting each worksheet ensures that the information the workbook contains is true and accurate.

6. In order to ensure integrity, click the "Review" tab at the top of the ribbon. If the worksheet is protected, "Unprotect Worksheet" will show below the icon, and no more action is required. If the icon reads "Protect Sheet", it is necessary to lock it. Up to this point, the "Users & Licenses" worksheet is the only sheet that should be unprotected. Select the icon, and lock the worksheet using the password "[INSERT PASSWORD TO UNLOCK WORKSHEET]". If this *exact*

password is not used to lock the sheet, the code will not run correctly. Do not change any other settings when setting a password.

7. Cell **A4** on the "ADD USER", "REPLACE USER", and "REMOVE USER" worksheets, and cell **A6** on the "REPLACE USER" worksheet are the only cells whose value can and need to be changed in order to run the VBA code. These cells are highlighted in yellow.
   The "Users & Licenses" worksheet will adjust automatically when the code is run using the macro assigned to each "SUBMIT" button within the worksheets.

Worksheet	Function	Cell(s) to Modify	Action
ADD USER	Add a new user	Cell A4	Enter User Name, Select Licenses, Press "SUBMIT"
REPLACE USER	Replace an existing user with a new one	Cell A4 (Old User), Cell A6 (New User)	Enter Old and New User Names, Press "SUBMIT"
REMOVE USER	Remove an existing user	Cell A4	Enter User Name, Press "SUBMIT"

8. What is typed in the yellow cells will not change the table in the "Users & Licenses" worksheet until the "SUBMIT" button of the active worksheet is pressed. The "SUBMIT" button will prompt the macro to run, and if the operation is successful, a message box will pop up.

9. Close the message box, and the changes should now be reflected in the table.
   Save the changes made to the workbook before closing the Excel application.

NOTE: The only part of the workbook as a whole that must be adjusted on a user-to-user basis is the number of total licenses per type of license (Step 4). This must be done before protecting and saving the workbook as macro-enabled (Step 5). This section of the workbook can be found in the "Users & Licenses" worksheet in cells B4:G4.

VBA CODE FOR “U&L” WORKBOOK

Marissa Brand, 2024

User Addition

Sub UserAddition()

' ORIGINAL CODE BY MARISSA J BRAND

' The function of this macro is to take the cell value entered in cell A4 of the “ADD USER” worksheet. Cell A4 works as the place of manual entry for the new user. The user will be added to the table *exactly* as their name is typed. After typing the user’s full name, select the license(s). If “TRUE” is displayed in the C column of the respective row, that license has been successfully selected. If no text or “FALSE” is displayed, that license is not selected. Press the “SUBMIT” button once the user’s name is entered completely and all applicable licenses have been selected.

```
' Unprotects the worksheets involved in the macro.
ThisWorkbook.Sheets("ADD USER").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"

' Declares Variables
    Dim ws As Worksheet
    Dim wsData As Worksheet
    Dim tbl As ListObject
    Dim tableName As String
    Dim newRow As ListRow
    Dim valueFromCell As Variant
    Dim insertionRow As Long
    Dim i As Long
    Dim j As Long
    Dim valueToCheck As Variant
    Dim previousRowHeight As Double

' Sets the worksheet where this macro is executed (adjust worksheet name as needed) and defines the table that will be the target location. If not completed, a message box will appear with the following message: “"'UandL' Table not found on the worksheet 'Users & Licenses'." and the macro will stop running.
    Set ws = ThisWorkbook.Sheets("Users & Licenses")
    tableName = "UandL"
On Error Resume Next
Set tbl = ws.ListObjects(tableName)
On Error GoTo 0
    If tbl Is Nothing Then
        MsgBox "'UandL' Table not found on the worksheet 'Users & Licenses'."
        Exit Sub
    End If

' Loops through the table rows starting from the bottom to safely delete blank rows.
    For i = tbl.ListRows.Count To 1 Step -1
        If WorksheetFunction.CountA(tbl.ListRows(i).Range) = 0 Then
            tbl.ListRows(i).Delete
        End If
    Next i
```

```
' This code sets the worksheet where the data to be added is located, gets the value from cell A4 in "ADD USER" sheet, finds the insertion row, and then adds the new row into the “UandL” table, as well as the value from cell A4.
  Set wsData = ThisWorkbook.Sheets("ADD USER")
  valueFromCell = wsData.Range("A4").Value
  insertionRow = ws.Range("Info").Row - 1
  Set newRow = tbl.ListRows.Add(insertionRow)
  newRow.Range(1, 1).Value = valueFromCell

' Set "x" in columns B to G of the new row based on conditions in the "ADD USER" sheet. The “x” that will display in the table following the execution of the code will represent the license(s) that the user is tied to. The finer details of this portion of code provide the font color, size, and name for the “x” value. If there is no applicable license for the user being entered into the table, no “x’s” will appear in the new row.
  For j = 2 To 7
  ' Check if B6 to B11 in "ADD USER" sheet contains "TRUE" for this column
    valueToCheck = wsData.Cells(6 + j - 2, 2).Value  ' Offset by j to correspond to rows 6 to 11
    If UCase(valueToCheck) = "TRUE" Then
      newRow.Range(1, j).Value = "x"
      With newRow.Range(1, j).Font
        .Size = 10
        .Color = RGB(255, 255, 255)  ' White color
        .Name = "Calibri"
      End With
    Else
      newRow.Range(1, j).ClearContents
    End If
  Next j

' This portion of code sets the formatting conditions of the newly inserted row.
  If tbl.ListRows.Count > 1 Then
    previousRowHeight = tbl.ListRows(tbl.ListRows.Count - 1).Range.RowHeight
    newRow.Range.RowHeight = previousRowHeight
  End If

' Set font colors for “Info” range and cell H2
  ws.Range("Info").Font.Color = RGB(255, 255, 255)
  ws.Range("H2").Font.Color = RGB(19, 47, 73)

' Message Box that will pop up if the operation is successful.
  MsgBox "User '" & valueFromCell & "' has been added."

' Protects the worksheets involved in the running of the macro
ThisWorkbook.Sheets("ADD USER").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
```

End Sub

User Replacement

Sub UserReplacement()

' ORIGINAL CODE BY MARISSA J BRAND

' The function of this macro is to automate the process of passing on the *exact* licenses of an inactive/old user in the table of the "Users & Licenses" worksheet to a new user. This code neither adds or removes a row to the table. This code takes the value entered in A4 of the "REPLACE USER" worksheet, matches it with its equal value in Column A of the "Users & Licenses" worksheet, and replaces that value with the value in cell A6 of the "REPLACE USER" worksheet.

ONLY RUN THIS MACRO IF THE NAME OF THE USER TIED TO THE LICENSES IS THE ONLY THING CHANGING. USE "ADD USER" AND "REMOVE USER" FOR SEPARATE USER MODIFICATIONS.

```
' Unprotects the worksheets involved in the macro.
ThisWorkbook.Sheets("REPLACE USER").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"

'Declare Variables
    Dim wsSource As Worksheet
    Dim wsTarget As Worksheet
    Dim sourceValue As Variant
    Dim replaceValue As Variant
    Dim targetRange As Range
    Dim cell As Range

' This code sets the source and target workbooks, or starting place and destination
    Set wsSource = ThisWorkbook.Worksheets("Replace User") ' Adjust sheet name as needed
    Set wsTarget = ThisWorkbook.Worksheets("Users & Licenses") ' Adjust sheet name as needed

' This code retrieves the values from the "REPLACE USER" worksheet that will be found and replaced within the table on the primary worksheet.
    sourceValue = wsSource.Range("A4").Value
    replaceValue = wsSource.Range("A6").Value

' This line sets, or defines, the cell range that will be searched through to find the value equal to that of cell A4. In this case, that range is every row in columns A through H of the "Users & Licenses" worksheet.
    Set targetRange = wsTarget.Range("A:H") ' Adjust the range as needed

' The last part of this macro searches through each cell in the defined range, checks if that value is equal to A4 in the "REPLACE USER" worksheet, and will continue looping through each cell until it finds the equal match. Once it does, it replaces the value in that cell with the value of cell A6 in the "REPLACE USER" worksheet. No other information or value in the row should be affected when this macro is executed.
    For Each cell In targetRange
        If cell.Value = sourceValue Then
            cell.Value = replaceValue
        End If
    Next cell
```

```
' This message Box will pop up if the operation is successful.
MsgBox "User '" & replaceValue & "' has been replaced with User '" & sourceValue & "'. License allocation remains unchanged."

' Protects the worksheets involved in the running of the macro
ThisWorkbook.Sheets("REPLACE USER").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
```

End Sub

User Removal

```
Sub UserRemoval()
' ORIGINAL CODE BY MARISSA J BRAND

' The function of this macro is to take the value of cell A4 in the “REMOVE USER” worksheet, and match it to its equivalent value in Column A of the “UandL” Table. If there is no match found, the macro will stop running. If there is a match found, the code will operate to delete the entire row corresponding to that value. All the formulas within the “Users & Licenses” worksheet operate to reflect any changes made based on macros in this workbook.

' Unprotects the worksheets involved in the macro.
ThisWorkbook.Sheets("REMOVE USER").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Unprotect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"

' Declare Variables
    Dim ws As Worksheet
    Dim wsData As Worksheet
    Dim tbl As ListObject
    Dim tableName As String
    Dim valueFromCell As Variant
    Dim foundRow As ListRow
    Dim searchValue As Variant
    Dim i As Long

' This line sets or defines the worksheet and table that the User will be removed from. In this case, that worksheet is the “Users & Licenses” worksheet and the “UandL” Table within that worksheet.
    Set ws = ThisWorkbook.Worksheets("Users & Licenses")
    tableName = "UandL"

' If the table is not found, this portion of code will yield the Message Box: “'UandL' Table not found on the active sheet." and the code will stop running.
    On Error Resume Next
    Set tbl = ws.ListObjects(tableName)
    On Error GoTo 0
    If tbl Is Nothing Then
        MsgBox "'UandL' Table not found on the active sheet."
        Exit Sub
    End If

' This code deletes any rows from the table that are completely blank.
    For i = tbl.ListRows.Count To 1 Step -1
        If Application.WorksheetFunction.CountA(tbl.ListRows(i).Range) = 0 Then
            tbl.ListRows(i).Delete
        End If
    Next i

' This code sets the worksheet “REMOVE USER” as the location that the value in cell A4 will be pulled from.
    Set wsData = ThisWorkbook.Sheets("REMOVE USER")
    searchValue = wsData.Range("A4").Value
```

```
' This final string of code loops through the rows of the “UandL” Table to find and match the value in cell A4 of the “REMOVE USER” worksheet with the respective row. If that value is found, the entire row is deleted, and the inputted formulas will automatically account for the available licenses following the deletion of the user.
Set foundRow = Nothing
   For Each foundRow In tbl.ListRows
      If foundRow.Range(1, 1).Value = searchValue Then
         foundRow.Delete
         Exit For
      End If
   Next foundRow
   If foundRow Is Nothing Then
      MsgBox "No matching value found in the table."
   Else
      MsgBox "User '" & searchValue & "' has been removed."
   End If

' Protects the worksheets involved in the running of the macro
ThisWorkbook.Sheets("REMOVE  USER").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
ThisWorkbook.Sheets("Users & Licenses").Protect Password:="[INSERT PASSWORD TO UNLOCK WORKSHEET]"
```

End Sub

IMPORTANT: “Emergency” Information

VBA CODE PASSWORD: [SET PASSWORD]

This password accesses the VBA code within the workbook.

To set a password to the VBA code for security and integrity purposes, open Visual Basic within the workbook. Next, select “Tools”, then “VBAProjectProperties”, navigate to the “Protection” tab, click the checkbox for “Lock project for viewing”, and complete the project by setting a password.

This password should **only** be shared with those that require it.

It should **only** be used to access the code with the intent to modify passwords for the goal of ensuring the efficiency and functionality of the code.