A screenshot of a computer

Description automatically generated

Sheet1 (PV)  
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Sheet2 (Timesheet)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Sheet3 (Work Life Balance)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Sheet4

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Sheet5

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

ThisWorkbook

Option Explicit

Dim DataUpdateOK As Boolean

**Private Sub Workbook\_Open**() 'this fires before Workbook\_Activate()

Dim shp As Object

Dim bTS As Boolean, bWL As Boolean, bInfo As Boolean

Dim OpenMsg As String

Dim dlgFileSaveAs As Dialog

Dim txtFileName As String

Dim wb As Workbook

Dim applicationCanClose As Boolean

TemplateFile = False

Dim i As Integer

Call logDataEntry("Version: 2021 Jun24 (" & Format(Range("TSVersion"), "00.00") & ")") ' added assetID to logging

Call logDataEntry("AssetID:" & Environ("COMPUTERNAME"))

'chg180412 - test if button error - version 11Apr18 13:09

On Error GoTo err\_trap

bTS = Abs(Sheets("PV").TimesheetButton.Height - 28.5) > 5 Or Abs(Sheets("PV").TimesheetButton.Width - 166.5) > 5

bWL = Abs(Sheets("PV").WorkLifeButton.Height - 28.5) > 5 Or Abs(Sheets("PV").WorkLifeButton.Width - 166.5) > 5

bInfo = Abs(Sheets("PV").InformationButton.Height - 28.5) > 5 Or Abs(Sheets("PV").InformationButton.Width - 166.5) > 5

On Error GoTo 0

If bTS Or bWL Or bInfo Then

err\_trap:

If True Then 'unlock timesheet

Application.ScreenUpdating = False

Dim currentSheet As String

currentSheet = ActiveSheet.Name

Sheets("Timesheet").Unprotect Password:=Pword

Sheets("Timesheet").Activate

'Columns("V:Z").EntireColumn.Hidden = False

Range(Columns(EntryDateCol), Columns(RWECol)).EntireColumn.Hidden = False

ActiveWindow.DisplayHeadings = True

Sheets("Work Life Balance").Unprotect Password:=Pword

Sheets("Work Life Balance").Activate

ActiveWindow.DisplayHeadings = True

Sheets("Admin").Unprotect Password:=Pword

Sheets("Admin").Activate

ActiveWindow.DisplayHeadings = True

Sheets("Activity").Unprotect Password:=Pword

Sheets("Activity").Activate

ActiveWindow.DisplayHeadings = True

Sheets("PV").Unprotect Password:=Pword

Sheets("PV").Select

Sheets("PV").Cells(1, 1).Select

ActiveWindow.DisplayHeadings = True

ActiveWindow.DisplayWorkbookTabs = True

Application.DisplayFormulaBar = True

Application.ExecuteExcel4Macro "SHOW.TOOLBAR(""Ribbon"",True)"

Sheets(currentSheet).Activate

Application.ScreenUpdating = True

End If

For i = Sheets("PV").Shapes.Count To 4 Step -1

Sheets("PV").Shapes(i).Delete

Next i

For i = 1 To 1000

DoEvents

Next i

'TimesheetButton

ActiveSheet.OLEObjects.Add(ClassType:="Forms.CommandButton.1", Link:=False \_

, DisplayAsIcon:=False, Left:=120, Top:=191.25, Width:=166.5, Height:=28.5 \_

).Name = "TimesheetButton"

Sheets("PV").Shapes("TimesheetButton").OLEFormat.Object.Object.Caption = "Go to my Timesheet"

Sheets("PV").Shapes("TimesheetButton").OLEFormat.Object.Enabled = False

'WorkLifeButton

ActiveSheet.OLEObjects.Add(ClassType:="Forms.CommandButton.1", Link:=False \_

, DisplayAsIcon:=False, Left:=120, Top:=224.25, Width:=166.5, Height:=28.5 \_

).Name = "WorkLifeButton"

Sheets("PV").Shapes("WorkLifeButton").OLEFormat.Object.Object.Caption = "Work / Life Balance"

Sheets("PV").Shapes("WorkLifeButton").OLEFormat.Object.Enabled = False

'InformationButton

ActiveSheet.OLEObjects.Add(ClassType:="Forms.CommandButton.1", Link:=False \_

, DisplayAsIcon:=False, Left:=120, Top:=258.75, Width:=166.5, Height:=28.5 \_

).Name = "InformationButton"

ActiveSheet.Shapes("InformationButton").OLEFormat.Object.Object.Caption = "Timesheet Manager"

ActiveSheet.Shapes("InformationButton").OLEFormat.Object.Enabled = False

For i = 1 To 1000

DoEvents

Next i

'Call LockTimesheet

MsgBox "An error occurred. You must close all excel workbooks and reopen the timesheet.", vbCritical, "Interface Error"

Call logDataEntry("Recovered from Error Chg180212: Layout rebuilt.")

Application.EnableEvents = False

Application.DisplayAlerts = False

ActiveWorkbook.Save

ThisWorkbook.Saved = True

ThisWorkbook.Application.Quit 'stops vba running, the following lines never have a chance to run

ActiveWorkbook.Close savechanges:=False

Set Application.ThisWorkbook = Nothing

Application.Quit

Else

Call logDataEntry("Opening... " & ActiveWorkbook.FullName)

'TimesheetButton

Set shp = Sheets("PV").TimesheetButton

With shp

.Enabled = False

With .Font

.Name = "Calibri"

.Size = 14

.Bold = True

End With

End With

'WorkLifeButton

Set shp = Sheets("PV").WorkLifeButton

With shp

.Enabled = False

With .Font

.Name = "Calibri"

.Size = 14

.Bold = True

End With

End With

'InformationButton

Set shp = Sheets("PV").InformationButton

With shp

.Enabled = False

With .Font

.Name = "Calibri"

.Size = 14

.Bold = True

End With

End With

End If

Sheets("PV").WorkLifeButton.Enabled = False

Sheets("PV").TimesheetButton.Enabled = False

Sheets("PV").InformationButton.Enabled = False

LockTimesheet

Sheets("PV").Select

If InStr(LCase(ThisWorkbook.Name), "template") > 0 Then

Fm\_Template.Show 'attempt to rename template

End If

If InStr(LCase(ThisWorkbook.Name), "template") > 0 Then

TemplateFile = True

Call logDataEntry("Failed to save timesheet as per instructions. Current location is: " & ActiveWorkbook.FullName)

MsgBox "Exiting. Please try again."

Application.EnableEvents = False

Application.DisplayAlerts = False

ThisWorkbook.Saved = True

ThisWorkbook.Application.Quit

End

ElseIf InStr(1, LCase(ActiveWorkbook.Path), LCase("D:\")) = 0 Then 'chg180521 - prevent users saving on home drive

Call logDataEntry("Workbook residing in unapproved folder " & ActiveWorkbook.FullName)

MsgBox "Please save your Timesheet under 'O:\PVGROUPS\Timesheets\2020-21\WorkCentres'. ", vbCritical

Else

Sheets("PV").InformationButton.Enabled = True

DoEvents

Fm\_Welcome.Show

End If

End Sub

**Private Sub Workbook\_Activate**()

Application.OnKey "{F3}", ""

Application.OnKey "+{F3}", "" 'Shift F3

Application.OnKey "^{F3}", "" 'Ctl F3

Application.OnKey "{F5}", "" 'F5

Application.OnKey "+{F5}", "" 'Shift F5

Dim oCtrl As Office.CommandBarControl

'Disable all Cut menus

For Each oCtrl In Application.CommandBars.FindControls(ID:=21)

oCtrl.Enabled = False

Next oCtrl

'Disable all Copy menus

For Each oCtrl In Application.CommandBars.FindControls(ID:=19)

oCtrl.Enabled = False

Next oCtrl

'Disable CellDragAndDrop

Application.CellDragAndDrop = False

'Hide Ribbon

ActiveWindow.DisplayWorkbookTabs = False

Application.DisplayFormulaBar = False

Application.ExecuteExcel4Macro "SHOW.TOOLBAR(""Ribbon"",False)"

End Sub

**Private Sub Workbook\_Deactivate**()

Dim oCtrl As Office.CommandBarControl

'Enable all Cut menus

For Each oCtrl In Application.CommandBars.FindControls(ID:=21)

oCtrl.Enabled = True

Next oCtrl

'Enable all Copy menus

For Each oCtrl In Application.CommandBars.FindControls(ID:=19)

oCtrl.Enabled = True

Next oCtrl

Application.CellDragAndDrop = True

'Show Tabs, Formula bar and Ribbon

ActiveWindow.DisplayWorkbookTabs = True

Application.DisplayFormulaBar = True

Application.ExecuteExcel4Macro "SHOW.TOOLBAR(""Ribbon"",true)"

Dim N As Long

For N = 1 To 100000000: Next

End Sub

**Private Sub Workbook\_BeforeClose**(Cancel As Boolean)

Dim User\_ID As String

If Range("LastName") = "" Then

ThisWorkbook.Saved = True 'timesheet not set up

Else

'chg180413 - lost data bug: copy a snapshot of the workbook to ErrorLogs folder

On Error Resume Next 'dont fail on this code

Dim fso As Scripting.FileSystemObject

Set fso = New Scripting.FileSystemObject 'under Options > Tools > References > Microsoft Scripting Runtime

If Range("PersonID") = "" Then User\_ID = Environ("USERNAME") Else User\_ID = Format(Range("PersonID"), "000000")

Call fso.CopyFile(ActiveWorkbook.FullName, "O:\PVgroups\Timesheets\Issues list\Error Logs\" & User\_ID & "\_bak.xlsm", True)

If Err.Number = 0 Then

Call logDataEntry("Backup successful")

Else

Call logDataEntry("Failed to save to " & "O:\PVgroups\Timesheets\Issues list\Error Logs\" & User\_ID & "\_bak.xlsm" & " (" & Err.Number & " - " & Err.Description & ")")

End If

On Error GoTo 0

End If

End Sub

**Private Sub Workbook\_SheetSelectionChange**(ByVal Sh As Object, ByVal Target As Range)

With Application

.CellDragAndDrop = False

.CutCopyMode = False 'Clear clipboard

End With

End Sub

'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Private Sub Workbook\_BeforeSave**(ByVal SaveAsUI As Boolean, Cancel As Boolean)

'Application.StatusBar = "Saving..."

If SaveAsUI Then

'MsgBox "Workbook\_BeforeSave 'SaveAsUi' DISABLED. Enable prior to release": GoTo Cont

MsgBox " Cannot use 'Save-As' option. Please click 'Save' to save your timesheet", vbExclamation, MsgCaption

Cancel = True

Exit Sub

End If

Cont:

If Range("LastName") <> "" Then

UpdateDatabaseRecords

If Not DataUpdateOK Then

MsgBox "Thank you, your timesheet has been saved." & vbCr & vbCr & \_

"NOTE: Your information cannot be sent to Oracle at this time.", vbInformation, MsgCaption

Call logDataEntry("UPDATE FAILURE > Your information connot be sent to Oracle at this time. " & ActiveWorkbook.FullName & ", AssetID:" & Environ("computerName"))

Else

MsgBox "Thank you, your timesheet has been saved.", vbInformation, MsgCaption

Call logDataEntry("KILL")

End If

End If

Sheets("PV").Select

Application.GoTo Reference:=Range("a1"), Scroll:=True

End Sub

**Private Sub UpdateDatabaseRecords**()

'Need to refresh Program list here as Programs are remaped during save

'Remapping is necessary so that any changes to the Program Table, are applied when saving Program data

If SetupRun Then DataUpdateOK = True: Exit Sub 'The timesheet has just been setup, no need to update

Dim Rec As Integer

Dim Hours As Single

Dim catagory As Integer

Dim Cmd, Status, OraDate As String

Dim SavedOK As Boolean

Dim WorkDate As Date

Dim recordsAffected As Long

Dim errMessage As String

DataUpdateOK = True

If Range("LastName") = "" Then

Call logDataEntry("UpdateDatabaseRecords > Timesheet not set up (LastName=null) > Failed.")

DataUpdateOK = True: Exit Sub 'Not setup

End If

If LCase(Range("TSVersion")) = "test" Then DataUpdateOK = True: Exit Sub

If Not ProgramListUpToDate Then UpdateProgramList (0)

If Not ProgramListUpToDate Then

Call logDataEntry("UpdateDatabaseRecords > ProgramList Not UpToDate > Failed.")

DataUpdateOK = False

Exit Sub 'Must not be able to connect. Error message will be displayed on return as DataUpdateOK = False

End If

Call OpenPVTSConnection(tmode)

If Not ConOpen Then

Call logDataEntry("UpdateDatabaseRecords > Cannot connect to database > Failed.")

DataUpdateOK = False

Exit Sub 'Error message will be displayed on return as DataUpdateOK = False

End If

On Error GoTo OraFail

'------ Update Labour records -----

'Test each timesheet row, If it is marked "entered" or "updated", save the data.

For Rec = Range("StartDate").row To Range("EndDate").row

OraDate = Format(Sheets("Timesheet").Cells(Rec, DateCol), "dd-mmm-yyyy")

Status = Sheets("Timesheet").Cells(Rec, StatusCol)

If LCase(Status) = "saved" Or Status = "" Or WorkDate > Now() Then GoTo Next\_Rec 'no need to save

'Status must be "entered" or "updated"

If LCase(Status) = "updated" Then

Cmd = "UPDATE PV\_TS\_LABOUR\_T "

Cmd = Cmd & "SET STATUS = 'closed' "

Cmd = Cmd & "WHERE Person\_ID = " & Range("PersonID") & " "

Cmd = Cmd & "AND WORK\_DATE = To\_Date('" & OraDate & "','DD-MON-YYYY')"

conn.Execute Cmd, recordsAffected

' Call logDataEntry(" SQL(" & Rec & ") > " & Cmd & " (" & recordsAffected & " records affected)")

Call logDataEntry(" SQL(" & Rec & ") > " & Cmd & " (" & IIf(recordsAffected < 1, "Error: Should have more than ", "") & recordsAffected & " records affected)")

Cmd = "DELETE from PV\_TS\_TIMESHEET\_T "

Cmd = Cmd & "where WORK\_DATE = To\_Date('" & OraDate & "','DD-MON-YYYY') "

Cmd = Cmd & "AND PERSON\_ID = " & Range("PersonID") & " "

conn.Execute Cmd, recordsAffected

' Call logDataEntry(" SQL(" & Rec & ") > " & Cmd & " (" & recordsAffected & " records affected)")

Call logDataEntry(" SQL(" & Rec & ") > " & Cmd & " (" & IIf(recordsAffected < 1, "Error: Should have more than ", "") & recordsAffected & " records affected)")

End If

'Save the Timesheet entry to the timesheet table

Cmd = "INSERT INTO PV\_TS\_TIMESHEET\_T "

Cmd = Cmd & "(PERSON\_ID, "

Cmd = Cmd & " WORK\_DATE, "

Cmd = Cmd & " TIME\_START, "

Cmd = Cmd & " TIME\_FINISH, "

Cmd = Cmd & " TIME\_LUNCH, "

Cmd = Cmd & " TIME\_EXTRA\_BREAK, "

Cmd = Cmd & " TIME\_TOTAL, "

Cmd = Cmd & " TIME\_ACCRUED, "

Cmd = Cmd & " TIME\_TIL, "

Cmd = Cmd & " TIME\_LEAVE, "

Cmd = Cmd & " TIME\_OVERTIME, "

Cmd = Cmd & " TIME\_COMM\_SVS, "

Cmd = Cmd & " T\_COMMENT, "

Cmd = Cmd & " LOCATION\_ID, "

Cmd = Cmd & " FUND\_SRC, "

Cmd = Cmd & " ACTIVITY, "

Cmd = Cmd & " NOTES, "

Cmd = Cmd & " ENTRY\_DATE, "

Cmd = Cmd & " ON\_DUTY, "

Cmd = Cmd & " DUTY\_CATAGORY, "

Cmd = Cmd & " STATUS, "

Cmd = Cmd & " RWE\_DAY) "

Cmd = Cmd & "VALUES "

Cmd = Cmd & " (" & Range("PersonID") & ", " 'Number PERSON\_ID

Cmd = Cmd & " '" & OraDate & "', " ' Date WORK\_DATE

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, StartCol) & "', " 'Text TIME\_START

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, FinishCol) & "', " 'Text TIME\_FINISH

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, BreakCol) & "', " 'Text TIME\_LUNCH

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, ExtraCol) & "', " 'Text

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, TotalCol) & "', " 'Text TIME\_TOTAL

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, ATCol) & "', " 'Text

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, TILCol) & "', " 'Text TIME\_TIL

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, LeaveCol) & "', " 'Text

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, OTCol) & "', " 'Text TIME\_OVERTIME

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, CSCol) & "', " 'Text

'Cmd = Cmd & " '" & Replace(Sheets("Timesheet").Cells(Rec, CommentsCol), "'", "''") & "', " 'Text T\_COMMENT

Cmd = Cmd & " '" & Left(Replace(Sheets("Timesheet").Cells(Rec, CommentsCol), "'", "''"), 255) & "', " 'Text T\_COMMENT 'restricted length of comment May2020 as per chg200506

Cmd = Cmd & " " & Val(Sheets("Timesheet").Cells(Rec, LocationIDCol)) & ", " 'Number

Cmd = Cmd & " '" & Sheets("Timesheet").Cells(Rec, FundCol) & "', " 'Text

Cmd = Cmd & " '" & Replace(Sheets("Timesheet").Cells(Rec, ActivityCol), "'", "''") & "', " 'Text

'Cmd = Cmd & " '" & Replace(Sheets("Timesheet").Cells(Rec, NotesCol), "'", "''") & "', " 'Text

Cmd = Cmd & " '" & Replace(Sheets("Timesheet").Cells(Rec, NotesCol), ")'", "''") & " (Saved " & Format(Now(), "DD-MMM-YY HH:MM:SS") & ")', " 'Text

Cmd = Cmd & " '" & Format(Sheets("Timesheet").Cells(Rec, EntryDateCol), "dd-mmm-yyyy") & "', " 'Date

Cmd = Cmd & " " & CInt(Sheets("Timesheet").Cells(Rec, DutyCol)) & ", " 'Number

Cmd = Cmd & " " & CInt(Sheets("Timesheet").Cells(Rec, CatagoryCol)) & ", " 'Number

Cmd = Cmd & " 'saved', " 'Text

Cmd = Cmd & " " & CInt(Sheets("Timesheet").Cells(Rec, RWECol)) & ") " 'Number

conn.Execute Cmd, recordsAffected

Call logDataEntry(" SQL(" & Rec & ") > " & Cmd & " (" & recordsAffected & " records affected)")

If recordsAffected = 0 Then GoTo OraFail

'Status is either "entered" or "updated"

SavedOK = True 'save the next timesheet unless code fails

catagory = Val(Sheets("Timesheet").Cells(Rec, CatagoryCol)) 'Either 1,2,3,4,5,6 or 7

Select Case catagory

Case 1: 'PV Work day

Call logDataEntry("UpdateDatabaseRecords > Case 1: 'PV Work day")

'There are always hours recorded for PV labour

'Get the hours then call SaveLabourRecords

Hours = TextToTime(Sheets("Timesheet").Cells(Rec, TotalCol)) 'Not affected by Variance display

If Hours > 0 Then SavedOK = SaveLabourRecords(Rec, Hours)

Case 2: 'ER

Call logDataEntry("UpdateDatabaseRecords > Case 2: 'ER")

'No hours are recorded on the timesheet for ER labour

'PV pays for normal work hours ONLY

'So if normal work hours = 0 on a weekday, then DEPI pay, consequently there is no labour to save

'If it is a weekend day, if an RDO was earned, PV is paying normal hours for the day

If Sheets("Timesheet").Cells(Rec, RWECol) = 1 Then

Hours = GetWeekendHours() 'ALWAYS returns a decimal value

Else

Hours = Range("PaidHours").Cells(GetPeriodDayNumber(Sheets("Timesheet").Cells(Rec, DateCol)))

End If

If Hours > 0 Then SavedOK = SaveLabourRecords(Rec, Hours)

Case 3, 4: 'Leave HOURS MUST ALWAYS be > 0 . {No Hours, No Leave)

Call logDataEntry("UpdateDatabaseRecords > Case 3, 4: 'Leave HOURS")

Hours = TextToTime(Sheets("Timesheet").Cells(Rec, LeaveCol))

Hours = Abs(Hours)

If Hours > 0.25 Then SavedOK = SaveLabourRecords(Rec, Hours)

'Case 4: 'LEAVE - Hours must ALWAYS be > 0 . {No Hours, No Leave)

' Hours = TextToTime(Sheets("Timesheet").Cells(Rec, LeaveCol))

' Hours = Abs(Hours)

' If Hours > 0 Then SavedOK = SaveLabourRecords(Rec, Hours)

Case 5: 'TIL

Call logDataEntry("UpdateDatabaseRecords > Case 5: 'TIL")

'No labour to record.

Case 6: 'RDO

Call logDataEntry("UpdateDatabaseRecords > Case 6: 'RDO")

'No labour to record.

Case 7: 'PERSONAL TIME - DAY OFF

Call logDataEntry("UpdateDatabaseRecords > Case 7: 'PERSONAL TIME - DAY OFF")

'No labour to record.

End Select

If Not SavedOK Then

Call logDataEntry("UpdateDatabaseRecords > SavedOK=False > (Row=" & Rec & ") - Error" & Err.Number & ": " & Err.Description & " > SQL: " & Cmd & " (" & recordsAffected & " records affected)")

Else

Sheets("Timesheet").Cells(Rec, StatusCol) = "saved"

End If

If False Then

OraFail:

errMessage = Err.Number & ": " & Err.Description

DataUpdateOK = False

Call logDataEntry("UpdateDatabaseRecords > OraFail > (Row=" & Rec & ") - Error" & errMessage & " > SQL: " & Cmd & " (" & recordsAffected & " records affected)")

MsgBox errMessage

errMessage = ""

Err.Clear

End If

Next\_Rec:

Next Rec

On Error GoTo UpdateError

If True Then 'False for offline timesheets

' ----- Now update Work Life data -----

'Test each work/life balance row and update if marked "updated"

Dim p As Integer

For p = 1 To 12

Cmd = "UPDATE PV\_TS\_WORKLIFE\_T "

Cmd = Cmd & "SET "

Cmd = Cmd & "LOCATION\_ID= " & Range("WorkCentre") & ", " 'This is the current "Home" location '\*\*\*NEW\*\*\*

Cmd = Cmd & "DAYS\_PV\_DUTY=" & CInt(Range("WorkLife").Cells(p, 2)) & ", "

Cmd = Cmd & "DAYS\_ER\_DUTY=" & CInt(Range("WorkLife").Cells(p, 3)) & ", "

Cmd = Cmd & "DAYS\_OFF\_DUTY=" & CInt(Range("WorkLife").Cells(p, 4)) & ", "

Cmd = Cmd & "DAYS\_ANNUAL\_LEAVE=" & CInt(Range("WorkLife").Cells(p, 5)) & ", "

Cmd = Cmd & "DAYS\_OTHER\_LEAVE=" & CInt(Range("WorkLife").Cells(p, 6)) & ", "

Cmd = Cmd & "DAYS\_WE\_DUTY=" & CInt(Range("WorkLife").Cells(p, 7)) & ", "

Cmd = Cmd & "DAYS\_ROSTERED\_OFF=" & CInt(Range("WorkLife").Cells(p, 8)) & ", "

Cmd = Cmd & "DAYS\_RDO\_BALANCE=" & CInt(Range("WorkLife").Cells(p, 9)) & ", "

Cmd = Cmd & "HOURS\_SPONSORED=" & Round(Range("WorkLife").Cells(p, 10), 2) & ", "

Cmd = Cmd & "HOURS\_OVERTIME=" & Round(Range("WorkLife").Cells(p, 11), 2) & ", "

Cmd = Cmd & "HOURS\_ACCRUED=" & Round(Range("WorkLife").Cells(p, 12), 2) & ", "

Cmd = Cmd & "HOURS\_IN\_LIEU=" & Round(Range("WorkLife").Cells(p, 13), 2) & ", "

Cmd = Cmd & "HOURS\_AT\_BALANCE=" & Round(Range("WorkLife").Cells(p, 14), 2) & ", "

Cmd = Cmd & "PV\_ROLE= '" & Replace(Range("Position"), "'", "''") & "', "

Cmd = Cmd & "ER\_ROLE= '" & Replace(Range("FireRole"), "'", "''") & "' "

Cmd = Cmd & "WHERE PERSON\_ID = '" & Range("PersonID") & "' " 'MOD MOD MOD

Cmd = Cmd & "AND PERIOD = " & p & " "

Cmd = Cmd & "AND FINANCIAL\_YEAR = '" & OraFinYear & "'"

conn.Execute Cmd, recordsAffected

Call logDataEntry(" SQL > " & Cmd & " (" & recordsAffected & " records affected)")

Range("WorkLife").Cells(p, 15) = "saved"

Next p

'Now save employee dynamic information (last updated, rdo balance, file location, password etc)

Cmd = "UPDATE PV\_TS\_USER\_T "

Cmd = Cmd & "SET "

Cmd = Cmd & "LAST\_UPDATE='" & Format(Date, "dd-mmm-yyyy") & "' , "

Cmd = Cmd & "AT\_BALANCE=" & Range("ATBalance") & ", "

Cmd = Cmd & "AT\_OPEN=" & Range("ATCarried") & ", "

Cmd = Cmd & "RDO\_BALANCE=" & Range("RDOBalance") & ", "

Cmd = Cmd & "RDO\_OPEN=" & Range("RDOCarried") & ", "

Cmd = Cmd & "START\_DATE='" & Format(Range("StartDate"), "dd-mmm-yyyy") & "' , "

Cmd = Cmd & "FILE\_LOCATION='" & Replace(ThisWorkbook.Path & "\" & ThisWorkbook.Name, "'", "''") & "', "

Cmd = Cmd & "PASSWORD='" & Range("Pkey") & "' "

Cmd = Cmd & "WHERE PERSON\_ID = '" & Range("PersonID") & "'" 'MOD MOD MOD

conn.Execute Cmd, recordsAffected

Call logDataEntry(" SQL > " & Cmd & " (" & recordsAffected & " records affected)")

End If

conn.Close

Set conn = Nothing

'DataUpdateOK = True 'modified 7Jun jm - DataUpdateOK defaults to true and is changed when an error occurs

Exit Sub

UpdateError:

errMessage = Err.Number & ": " & Err.Description

DataUpdateOK = False

Call logDataEntry("UpdateDatabaseRecords > UpdateError > Error" & errMessage & " > SQL: " & Cmd & " (" & recordsAffected & " records affected)")

MsgBox errMessage

If conn.State <> 0 Then conn.Close

Set conn = Nothing

End Sub

**Function SaveLabourRecords**(Rec As Integer, Hours As Single) As Boolean

Dim row, col, prog As Integer

Dim ActivityRowNumber As Integer

Dim ProgNum As String

Dim Cmd, FundSourceNumber As String

Dim sh1 As Worksheet

Dim sh2 As Worksheet

Dim TotalTime, Percent As Single

Dim recordsAffected As Long

Set sh1 = Sheets("Activity")

Set sh2 = Sheets("Timesheet")

ActivityRowNumber = getActivityRowNumber(sh2.Cells(Rec, ActivityCol)) 'Get the Activity list row number

If ActivityRowNumber = 0 Then Exit Function 'Just incase the activity is unknown (fail-safe)

row = Range("ActivityList").row + ActivityRowNumber - 1 'this is the sheet row number of the activity

col = Range("ActivityList").Column 'this is the sheet column of the activity list

FundSourceNumber = sh2.Cells(Rec, FundCol) 'string

For prog = 1 To 6

ProgNum = sh1.Cells(row, col + (prog \* 2) - 1)

If ProgNum = "" Then GoTo Next\_Prog ' less than 6 programs in the activity

ProgNum = Get\_Program\_Map\_To(ProgNum)

Percent = sh1.Cells(row, col + (prog \* 2))

TotalTime = Hours \* Percent

If Round(TotalTime, 2) = 0 Then GoTo Next\_Prog 'Just incase (fail-safe)

'Save the data

Cmd = "INSERT INTO PV\_TS\_LABOUR\_T "

Cmd = Cmd & "(WORK\_DATE, "

Cmd = Cmd & " ENTRY\_DATE, "

Cmd = Cmd & " STATUS, "

Cmd = Cmd & " Person\_ID, "

Cmd = Cmd & " PROGRAM\_ID, "

Cmd = Cmd & " FUND\_SOURCE\_NUMBER, "

Cmd = Cmd & " HOURS, "

Cmd = Cmd & " LOCATION\_ID) "

Cmd = Cmd & "VALUES "

Cmd = Cmd & " ('" & Format(sh2.Cells(Rec, DateCol), "dd-mmm-yyyy") & "', "

Cmd = Cmd & " '" & Format(sh2.Cells(Rec, EntryDateCol), "dd-mmm-yyyy") & "', "

Cmd = Cmd & " 'open', "

Cmd = Cmd & " '" & Range("PersonID") & "', "

Cmd = Cmd & " '" & ProgNum & "', "

Cmd = Cmd & " '" & FundSourceNumber & "', "

Cmd = Cmd & " " & TotalTime & ", "

Cmd = Cmd & " " & Val(Cells(Rec, LocationIDCol)) & ")"

conn.Execute Cmd, recordsAffected

Call logDataEntry(" SQL > " & Cmd & " (" & recordsAffected & " records affected)")

If recordsAffected = 0 Then GoTo OraFail:

Next\_Prog:

Next prog

SaveLabourRecords = True

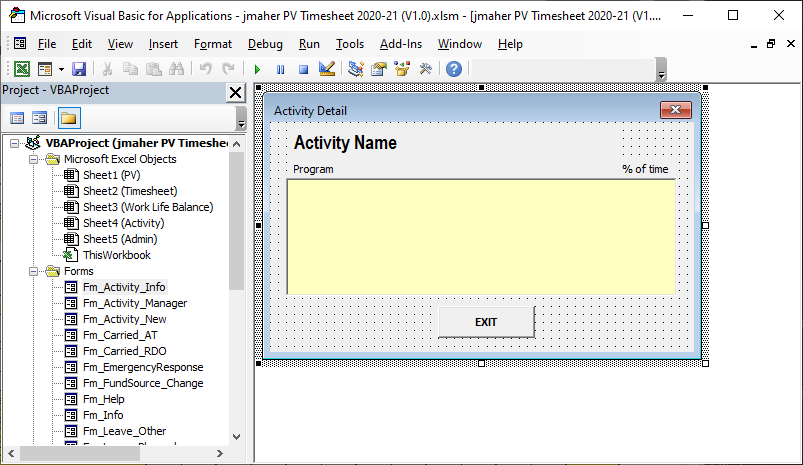
Exit Function

OraFail:

Call logDataEntry("SaveLabourRecords > OraFail > Error" & Err.Number & ": " & Err.Description)

Call logDataEntry(" Failed SQL > " & Cmd & " (" & recordsAffected & " records affected)")

End Function

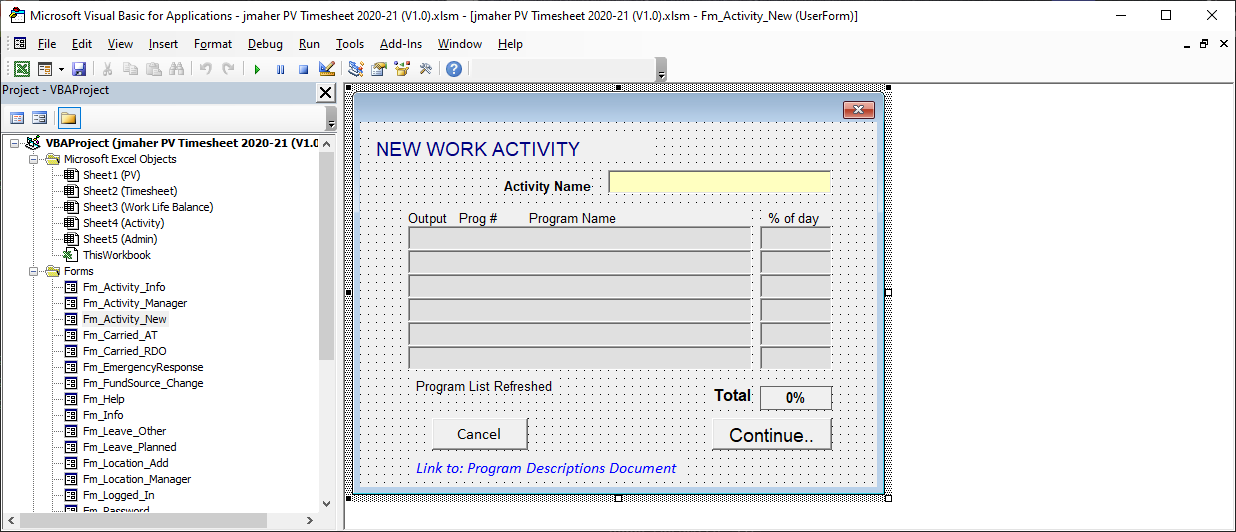


A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a key

Description automatically generated

A screenshot of a computer

Description automatically generated

RecordHours

A screenshot of a computer

Description automatically generated

Option Explicit

Dim ThisDay As Date

Dim CurRow As Integer 'Currently selected Row

Dim PaidHours As Single 'Paid Hours on selected day

Dim RecHrsActivating As Boolean

Dim Calculating As Boolean

Dim FormReady As Boolean

Dim VarianceVisible As Boolean

Dim variance As Single

Dim CommentRequired As Boolean

Const VarianceOption = "Custom Variance"

**Private Sub UserForm\_Activate**()

Dim DayNum As Integer

Dim N As Integer

With Fm\_Record\_Hours

.StartUpPosition = 0

.Left = Application.Left + (0.5 \* Application.Width) - (0.5 \* .Width)

.Top = Application.Top + (0.5 \* Application.Height) - (0.5 \* .Height)

End With

DoEvents

'Tab order will set focus to AssistantButton

If Not ActiveSheet.Name = "Timesheet" Then Me.Hide: MsgBox "Illegal Form Activate event!", vbExclamation, MsgCaption: Exit Sub

Application.ScreenUpdating = False

RecHrsActivating = True

CurRow = ActiveCell.row

ThisDay = Cells(CurRow, DateCol)

DayNum = GetPeriodDayNumber(ThisDay)

HideVariance

DayLabel = Format(ThisDay, "dddd")

DateLabel = Format(ThisDay, "dd / MMM / yyyy")

StartTimeBox = ""

FinishTimeBox = ""

BreakLabel.Caption = "0:00"

ExtraBreakBox = ""

TotalHoursLabel.Caption = "0:00"

LocationBox = ""

ActivityBox = ""

VarianceTypeBox = ""

PaidHours = Range("PaidHours").Cells(DayNum)

If PaidHours = 0 \_

And Weekday(ThisDay, 2) > 5 \_

And Range("Takes\_RDOs") = "yes" Then

PaidHours = GetWeekendHours()

End If

'Is this a public Holiday? NOTE: There can be no Variance on a Public Holiday. All hours are paid

If PubHol Then 'PubHol is a global boolean set by Select Activity Form

ShowPublicHoliday

HolidayLabel.Caption = "Public Holiday : " & GetPublicHoliday(ThisDay)

If PenaltyBox.ListCount = 0 Then

PenaltyBox.AddItem ("Extra Pay Option") 'default

PenaltyBox.AddItem ("Accrued Time Option")

End If

PenaltyBox = PenaltyBox.List(0) 'default

Else

HidePublicHoliday

End If

If Range("VarianceFormat") = "decimal" Then

Lb\_NormalHours\_2 = Format(PaidHours, "0.00") 'Can be 0

Else

Lb\_NormalHours\_2 = Format(PaidHours / 24, "h:mm") 'Can be 0

End If

N = 2

LocationBox.Clear

While Range("MyLocationList").Cells(N, 1) <> ""

LocationBox.AddItem GetLocationName(Range("MyLocationList").Cells(N, 1))

N = N + 1

Wend

If LocationBox.ListCount = 0 Then 'should never happen

LocationBox.AddItem GetLocationName(Range("WorkCentre"))

End If

'Load default location (if it is in LocationBox List)

If Range("DefaultLocation") <> "" Then

LocationBox = Range("DefaultLocation")

If Not LocationBox.MatchFound Then LocationBox = ""

End If

'Load Activities

ActivityBox.Clear

Select Case Caller

Case 1: 'Fire & Emergency call

ActivityBox.AddItem (Fm\_EmergencyResponse.DutyBox)

ActivityBox = Fm\_EmergencyResponse.DutyBox

Lb\_MoreActivities.Visible = False 'Hidden for an F&E caller

Lb\_NewActivity.Visible = False

Case Else: 'PV Work day

N = 2

While Range("ActivityList").Cells(N, 1) <> ""

If Range("ActivityList").Cells(N, 14) = "open" Then ActivityBox.AddItem (Range("ActivityList").Cells(N, 1))

N = N + 1

Wend

Lb\_MoreActivities.Visible = True

Lb\_NewActivity.Visible = True

End Select

'Load Default start time

If Range("NormalStart") <> "" Then

StartTimeBox = Range("NormalStart")

If StartTimeBox.MatchFound Then

CalculateFinishTime

CalculateWorkedHours

Else

StartTimeBox = ""

End If

End If

FundLabel = GetFundSourceName(Range("PVFundSource"))

CheckForm

FindFocus

RecHrsActivating = False

Application.ScreenUpdating = True

End Sub

**Private Sub FundButton\_Click**()

Fm\_FundSource\_Change.Show

If Fm\_FundSource\_Change.FundBox.Text <> "" Then

FundLabel = Fm\_FundSource\_Change.FundBox.Text

Range("PVFundSource") = GetFundSourceNumber(FundLabel)

End If

FindFocus

End Sub

**Private Sub FundLabel\_Click**()

Call FundButton\_Click

End Sub

**Private Sub Label12\_Click**() 'Activity Label

Dim N As Integer

If Caller = 1 Then Exit Sub

N = 2

ActivityBox.Clear

While Range("ActivityList").Cells(N, 1) <> ""

If Range("ActivityList").Cells(N, 14) = "open" Then ActivityBox.AddItem (Range("ActivityList").Cells(N, 1))

N = N + 1

Wend

ActivityBox.DropDown

End Sub

**Private Sub Label13\_**Click() 'Location Label

Dim N As Integer

N = 2

LocationBox.Clear

While Range("MyLocationList").Cells(N, 1) <> ""

LocationBox.AddItem (GetLocationName(Range("MyLocationList").Cells(N, 1))) '\*\*\*NEW\*\*\* LocationListMod

N = N + 1

Wend

LocationBox.DropDown

End Sub

**Private Sub AssistantButton\_**Click()

Fm\_Timesheet\_Assistant.Show

Call UserForm\_Activate

End Sub

**Private Sub StartTimeBox\_Click**()

If RecHrsActivating Then Exit Sub

RecHrsActivating = True

CalculateFinishTime

CalculateWorkedHours

CheckForm

FindFocus

RecHrsActivating = False

End Sub

**Private Sub StartTimeBox\_AfterUpdate**()

'This event is caused when the control looses focus, so it should NOT set focus or call a routine which will set focus

If RecHrsActivating Then Exit Sub

If StartTimeBox = "" Then CheckForm: Exit Sub

If Not StartTimeBox.MatchFound Then

StartTimeBox = Format(TextToTime(StartTimeBox) / 24, "h:mm") 'this format will not allow time > 23:59

End If

CalculateFinishTime

CalculateWorkedHours

CheckForm

End Sub

**Private Sub FinishTimeBox\_Click**()

If RecHrsActivating Then Exit Sub

RecHrsActivating = True

CalculateWorkedHours

CheckForm

FindFocus

RecHrsActivating = False

End Sub

**Private Sub FinishTimeBox\_AfterUpdate**()

If RecHrsActivating Then Exit Sub

If FinishTimeBox = "" Then Exit Sub

If Not FinishTimeBox.MatchFound Then

FinishTimeBox = Format(TextToTime(FinishTimeBox) / 24, "h:mm")

End If

CalculateWorkedHours

CheckForm

FindFocus

End Sub

**Private Sub ExtraBreakBox\_**Click()

If RecHrsActivating Then Exit Sub

RecHrsActivating = True

CalculateWorkedHours

CheckForm

FindFocus

RecHrsActivating = False

End Sub

**Private Sub ExtraBreakBox\_AfterUpdate**()

If RecHrsActivating Then Exit Sub

If Not ExtraBreakBox.MatchFound Then ExtraBreakBox = Format(TextToTime(ExtraBreakBox) / 24, "h:mm")

CalculateWorkedHours

CheckForm

FindFocus

End Sub

**Private Sub ActivityBox\_Change**()

If RecHrsActivating Then Exit Sub

CheckForm

FindFocus

End Sub

**Private Sub LocationBox\_Change**()

If RecHrsActivating Then Exit Sub

CheckForm

FindFocus

End Sub

**Private Sub PenaltyBox\_Click**()

If RecHrsActivating Then Exit Sub

CalculateWorkedHours

CheckForm

FindFocus

End Sub

**Private Sub VarianceTypeBox\_**Click()

If RecHrsActivating Then Exit Sub

If Calculating Then Exit Sub

CheckForm

FindFocus

End Sub

**Private Sub CommentBox\_Change**()

If RecHrsActivating Then Exit Sub

If Len(CommentBox.Text) > 255 Then

CommentBox.Text = Left(CommentBox.Text, 255) 'chg200506 for new year template

End If

CheckForm

End Sub

**Private Sub PenaltyBox\_change**()

If Len(PenaltyBox.Text) > 255 Then

PenaltyBox.Text = Left(PenaltyBox.Text, 255) 'chg200506 for new year template

End If

End Sub

**Private Sub ContinueButton\_Click**()

Dim LocID As Long

If VarianceTypeBox = VarianceOption Then

Fm\_Variance.Show

If Not Fm\_Variance.ContinueButtonClicked Then Exit Sub '(user cancelled)

End If

Application.ScreenUpdating = False

ClearTimesheetRow (CurRow) 'Remove any existing data

Select Case VarianceTypeBox

Case "Time in Lieu"

Cells(CurRow, TILCol) = TimeToText(variance, 1)

Case "Accrued Time"

Cells(CurRow, ATCol) = TimeToText(variance, 1)

Case "Community Service"

Cells(CurRow, CSCol) = TimeToText(variance, 1)

Case "Overtime (subject to approval)"

Cells(CurRow, OTCol) = TimeToText(variance, 1)

Case "Leave (subject to approval)"

Cells(CurRow, LeaveCol) = TimeToText(variance, 1)

Case VarianceOption: 'VarianceOption is a constant declared in this module

If variance < 0 Then

If Fm\_Variance.TextBox1 <> "" Then Cells(CurRow, TILCol) = "-" & Fm\_Variance.TextBox1

If Fm\_Variance.TextBox2 <> "" Then Cells(CurRow, LeaveCol) = "-" & Fm\_Variance.TextBox2

Else

If Fm\_Variance.TextBox1 <> "" Then Cells(CurRow, ATCol) = Fm\_Variance.TextBox1 'Always Accrued Time in this box

If Fm\_Variance.TextBox2 <> "" Then Cells(CurRow, OTCol) = Fm\_Variance.TextBox2 'Always Overtime

End If

End Select

Me.Hide

'record times on timesheet as text

Cells(CurRow, StartCol) = StartTimeBox

Cells(CurRow, FinishCol) = FinishTimeBox

Cells(CurRow, BreakCol) = BreakLabel

Cells(CurRow, ExtraCol) = ExtraBreakBox

Cells(CurRow, TotalCol) = TotalHoursLabel

Cells(CurRow, LocationCol) = LocationBox

Cells(CurRow, LocationCol).Font.Color = RGB(0, 0, 0) 'black

Cells(CurRow, ActivityCol) = ActivityBox 'this is always a listed activity

Cells(CurRow, ActivityCol).Font.Color = RGB(0, 0, 0) 'black

Cells(CurRow, CommentsCol).Font.Color = RGB(0, 0, 0) 'black

'Check the LocationID number. If it is 0 (work from home etc) default to home workcentre

LocID = GetLocationID(LocationBox) 'returns long

If LocID < 100 Then LocID = Range("WorkCentre")

Cells(CurRow, LocationIDCol) = LocID 'long

Cells(CurRow, CommentsCol) = CommentBox

Cells(CurRow, EntryDateCol) = Date

If Left(ActivityBox, 8) = "Rest Day" Then Cells(CurRow, DutyCol) = 0 Else Cells(CurRow, DutyCol) = 1 'for conditional format

Cells(CurRow, CatagoryCol) = 1 ' Activity Number (PV Duty)

'Public Holiday - enter Comments and variance type

If PubHol Then

If variance > 0 Then 'the user chose extra time option

Cells(CurRow, ATCol) = Format(variance / 24, "h:mm")

End If

Cells(CurRow, CommentsCol) = PenaltyBox

End If

If Cells(CurRow, StatusCol) = "saved" Or Cells(CurRow, StatusCol) = "updated" Then

Cells(CurRow, StatusCol) = "updated"

Else

Cells(CurRow, StatusCol) = "entered"

End If

Cells(CurRow, FundCol) = Range("PVFundSource")

'Add an RDO for the weekend day worked

If Range("Takes\_RDOs") = "yes" And Weekday(ThisDay, 2) > 5 Then

Cells(CurRow, RWECol) = 1

Else

Cells(CurRow, RWECol).ClearContents

End If

UpdateWorkLifeBalance (GetPeriod(Cells(CurRow, DateCol))) 'Timesheet summary data for the period

'Call CalculateTimesheetSummary(GetStartDate(), GetEndDate(ActiveCell.row))

Call CalculateTimesheetSummary(0)

Application.ScreenUpdating = True

Cells(CurRow, DateCol).Select

End Sub

**Private Sub Lb\_MoreActivities\_Click**()

Dim N As Integer

If Caller = 1 Then Exit Sub 'Called from F&E, only one activity available

ActivityBox.Clear

N = 2

While Range("ActivityList").Cells(N, 1) <> ""

If Range("ActivityList").Cells(N, 14) = "closed" Or Range("ActivityList").Cells(N, 14) = "open" Then ActivityBox.AddItem (Range("ActivityList").Cells(N, 1))

N = N + 1

Wend

ActivityBox.DropDown

End Sub

**Private Sub Lb\_MoreLocations\_Click**()

Dim N As Integer

LocationBox.Clear

N = 2

While Range("PVLocationList").Cells(N, 1) <> ""

LocationBox.AddItem (Range("PVLocationList").Cells(N, 2))

N = N + 1

Wend

LocationBox.DropDown

End Sub

**Private Sub Lb\_NewActivity\_Click**()

Fm\_Activity\_New.Show

If Fm\_Activity\_New.ActivityNameBox <> "" Then

ActivityBox.AddItem (Fm\_Activity\_New.ActivityNameBox)

ActivityBox = Fm\_Activity\_New.ActivityNameBox

CheckForm

End If

End Sub

**Private Sub CalculateWorkedHours**()

'This will populate the break, total hours and variance boxs

Dim TempVar As String

Dim ST As Single ' Start Time

Dim FT As Single ' Finish Time

Dim BR As Single ' Break

Dim EB As Single ' Extra Break

Dim WorkedHours As Single ' Total Hours Worked

If StartTimeBox = "" Or FinishTimeBox = "" Then Exit Sub

Calculating = True 'Disable click events (Extra break, Variance)

'-----------------------------------------------------------------------------

ST = TextToTime(StartTimeBox) ' convert Start Time string to single

FT = TextToTime(FinishTimeBox) 'Finish Time

If ExtraBreakBox <> "" Then

EB = TextToTime(ExtraBreakBox)

If Not ExtraBreakBox.MatchFound Then ExtraBreakBox = Format(EB / 24, "h:mm")

Else

EB = 0

End If

If FT < ST Then FT = FT + 24 'Finished on the following day

WorkedHours = FT - ST

If WorkedHours > 5 Then

BR = 0.5

BreakLabel = "0:30"

Else

BR = 0

BreakLabel = "0:00"

End If

WorkedHours = FT - ST - BR - EB 'this is always positive

TotalHoursLabel = Format(WorkedHours / 24, "h:mm")

'-----------------------------------------------------------------------------

'CALCULATE VARIANCE

If PubHol Then

If PenaltyBox <> PenaltyBox.List(0) Then 'Accrued Time Option Selected

variance = WorkedHours

If variance > 7.6 Then variance = 7.6 'Change requested by Tony Duras

Else 'Extra Pay Option Selected

variance = 0

End If

Else 'NOT PUBLIC HOLIDAY

variance = Round(WorkedHours - PaidHours, 2)

If Abs(variance) > 0.01 Then 'There is a variance (either + or -)

ShowVariance

VarianceTypeBox.Clear

VarHoursLabel = Format(variance / 24, "h:mm") 'Show the Variance time

'Short day ---------------------------------

If variance < 0 Then

VarHoursLabel = "- " & VarHoursLabel

If Range("ATBalance") >= Abs(variance) Then

'Sufficient TIL so show TIL Option

VarianceTypeBox.AddItem ("Time in Lieu")

End If

VarianceTypeBox.AddItem ("Leave (subject to approval)")

If Range("ATBalance") > 0 Then

VarianceTypeBox.AddItem (VarianceOption)

End If

VarianceTypeBox = VarianceTypeBox.List(0)

Else 'Long day ---------------------------------

If LCase(Range("ATAgreement")) = "yes" Then 'And Range("ATBalance") < Range("ATLimit") Then

VarianceTypeBox.AddItem ("Accrued Time")

End If

VarianceTypeBox.AddItem ("Overtime (subject to approval)")

VarianceTypeBox.AddItem ("Community Service")

End If

If Not VarianceTypeBox.MatchFound Then VarianceTypeBox = VarianceTypeBox.List(0)

Else 'NO VARIANCE

variance = 0

VarianceTypeBox = ""

HideVariance

End If

End If

Calculating = False

End Sub

**Private Sub CheckForm**()

FormReady = True

If StartTimeBox = "" Then FormReady = False: StartTimeBox.BackColor = Red Else StartTimeBox.BackColor = Buff

If FinishTimeBox = "" Then FormReady = False: FinishTimeBox.BackColor = Red Else FinishTimeBox.BackColor = Buff

If variance <> 0 Then

If Not PubHol Then

If Not VarianceTypeBox.MatchFound Then FormReady = False: VarianceTypeBox.BackColor = Red Else VarianceTypeBox.BackColor = Buff

End If

IsCommentRequired

If CommentRequired And Trim(CommentBox) = "" Then FormReady = False: CommentBox.BackColor = Red Else CommentBox.BackColor = Buff

End If

If Not ActivityBox.MatchFound Then FormReady = False: ActivityBox.BackColor = Red Else ActivityBox.BackColor = Buff

If Not LocationBox.MatchFound Then FormReady = False: LocationBox.BackColor = Red Else LocationBox.BackColor = Buff

If FormReady Then

ContinueButton.Enabled = True

Else

ContinueButton.Enabled = False

End If

End Sub

**Private Sub CalculateFinishTime**()

If LCase(Range("AutoCalculateHours")) = "yes" And PaidHours > 0 Then

If PaidHours > 5 Then 'Max 5 hours work without unpaid 1/2 hour break

FinishTimeBox = Format((TextToTime(StartTimeBox) + PaidHours + 0.5) / 24, "h:mm") 'this format will not allow time > 23:59

Else

FinishTimeBox = Format((TextToTime(StartTimeBox) + PaidHours) / 24, "h:mm") 'this format will not allow time > 23:59

End If

End If

End Sub

**Private Sub FindFocus**()

If FormReady Then ContinueButton.SetFocus: Exit Sub

If StartTimeBox = "" Then StartTimeBox.SetFocus: Exit Sub

If FinishTimeBox = "" Then FinishTimeBox.SetFocus: Exit Sub

If LocationBox = "" Then LocationBox.SetFocus: Exit Sub

If ActivityBox = "" Then ActivityBox.SetFocus: Exit Sub

If VarianceVisible Then CommentBox.SetFocus: Exit Sub

AssistantButton.SetFocus

End Sub

**Private Sub ShowVariance**()

Me.Height = 184

VarianceVisible = True

VarianceTypeBox.Top = Me.Height - 79

VarHoursLabel.Top = VarianceTypeBox.Top

VarianceLabel.Top = Me.Height - 81

CommentBox.Top = VarianceTypeBox.Top

CommentLabel.Top = VarianceLabel.Top

CancelButton.Top = Me.Height - 54

ContinueButton.Top = CancelButton.Top

AssistantButton.Top = CancelButton.Top

FundButton.Top = CancelButton.Top

FundLabel.Top = CancelButton.Top + 12

End Sub

**Private Sub HideVariance**()

Me.Height = 159

VarianceVisible = False

VarianceTypeBox.Top = Me.Height + 30

VarianceLabel.Top = Me.Height + 30

VarHoursLabel.Top = Me.Height + 30

CommentBox = ""

CommentBox.Top = Me.Height + 30

CommentLabel.Top = Me.Height + 30

CancelButton.Top = Me.Height - 56

ContinueButton.Top = CancelButton.Top

AssistantButton.Top = CancelButton.Top

FundButton.Top = CancelButton.Top

FundLabel.Top = CancelButton.Top + 12

End Sub

**Private Sub ShowPublicHoliday**()

PenaltyBox.Visible = True

HolidayLabel.Visible = True

Lb\_NormalHours\_1.Visible = False

Lb\_NormalHours\_2.Visible = False

End Sub

**Private Sub HidePublicHoliday**()

PenaltyBox.Visible = False

HolidayLabel.Visible = False

Lb\_NormalHours\_1.Visible = True

Lb\_NormalHours\_2.Visible = True

End Sub

**Private Sub CancelButton\_Click**()

Me.Hide

Fm\_Select\_Activity.Show

End Sub

**Private Sub IsCommentRequired**()

Dim CR As Boolean

CR = False

If Not PubHol Then

CommentRequired = True: CR = True

If VarianceTypeBox = "Accrued Time" Or VarianceTypeBox = "Time in Lieu" Then

If Abs(variance) <= Range("ATMax") Then CommentRequired = False: CR = False

End If

Else

CommentRequired = False

Exit Sub

End If

If CR Then

CommentLabel.ForeColor = DarkRed

Else

CommentLabel.ForeColor = Black

End If

End Sub

A screenshot of a computer

Description automatically generated

Option Explicit

'Normal Width, Height & Top of Images/Labels

Const NormalWidth = 96

Const NormalHeight = 66

Const ImageTop = 42

Const LabelTop = 114

Const LeftMargin = 36

Const Gap = 2

Dim Option1 As String

Dim Option2 As String

Dim Option3 As String

Dim Option4 As String

Dim Option5 As String

Dim Option6 As String

Dim CurRow As Integer 'Selected Row

Dim ThisDay As Date

Dim DayNum As Integer

Dim PaidHours As Single

Dim ImageSelected As Boolean 'to stop flicker as mouse moves overan form

Dim SelectedImage As Integer 'to stop flicker as mouse moves overan image

**Private Sub ImageSelect**(ImageNumber As Integer, Img As Object, Txt As String)

If ImageNumber = SelectedImage Then Exit Sub

DrawImages

Img.Left = Img.Left - 18

Img.Width = Img.Width + 36

Img.Height = Img.Height + 36

Img.Top = Img.Top - 9

Img.ZOrder (0) 'top

Label1.Caption = Txt

'Label1.Left = Img.Left - 60

ImageSelected = True

SelectedImage = ImageNumber

End Sub

**Private Sub UserForm\_Activate()** 'Always called when sheet("Timesheet") is Active

Image3.Visible = False

Image4.Visible = False

Image5.Visible = False

Image6.Visible = False

'Dim lngStatus As Long

'Dim typWhere As POINTAPI

'lngStatus = GetCursorPos(typWhere)

'typWhere.x & Chr(13) & "y: " & typWhere.y, vbInformation, "Pixels"

With Fm\_Select\_Activity

.StartUpPosition = 0

.Left = Application.Left + (0.5 \* Application.Width) - 300

'.Left = Application.Left + (0.5 \* Application.Width) - (0.5 \* .Width)

.Top = Application.Top + (0.5 \* Application.Height) - (0.5 \* .Height)

'.Top = ActiveCell.Top \* 0.75

.Width = 170

End With

DoEvents

CurRow = ActiveCell.row

If Not IsDate(Cells(CurRow, DateCol)) Then Me.Hide: MsgBox "Illegal form call", vbExclamation, MsgCaption: Exit Sub

ThisDay = Cells(CurRow, DateCol)

Option1 = "PV Workday"

Option2 = "Emergency Readiness & Response"

Option3 = "Planned Leave"

Option4 = "Other Leave"

Option5 = "Time in Lieu"

Option6 = "Rostered Day Off"

Application.ScreenUpdating = False

On Error GoTo OptionsError

Me.Caption = Format(ThisDay, "dddd dd/mmm/yyyy")

'--------------------------------------------------

'Public Holiday (can be Sat/Sun) All hours are paid, so no AT/TIL/RDO/Lve/AL

If IsPublicHoliday(ThisDay) Then

PubHol = True

Me.Caption = Me.Caption & " - " & GetPublicHoliday(ThisDay)

Image6.Visible = True

If Weekday(ThisDay, 2) > 5 Then

Option6 = "Weekend : Day off"

Else

Option6 = "Public Holiday : Day off"

End If

GoTo ShowForm '------- DONE

Else

PubHol = False

End If

'--------------------------------------------------

'Weekend

If Weekday(ThisDay, 2) > 5 Then

Image6.Visible = True

Option6 = "Weekend Day off"

If Range("RosteredDays") <> 0 And Range("Takes\_RDOs") = "yes" Then

Image4.Visible = True 'Incidental leave

End If

GoTo ShowForm '------- DONE

End If

'--------------------------------------------------

'Week Day

PaidHours = Range("PaidHours").Cells(GetPeriodDayNumber(ThisDay))

If PaidHours = 0 Then 'NOT A NORMAL WORKDAY SO, No Leave, No RDO, No TIL

Image6.Visible = True

Option6 = "Personal Time : Day off"

Else 'NORMAL WORKDAY

Image3.Visible = True 'Planned Leave

Image4.Visible = True 'Incidental leave

If Range("RDOBalance") > 0 Then

Image6.Visible = True

Option6 = "Rostered Day Off"

End If

'Accrued Time off (TIL)

If Range("ATBalance") >= PaidHours Then

Image5.Visible = True

End If

End If

ShowForm:

SelectedImage = 0 'to stop flicker s mouse moves overan image

PresentForm

Application.ScreenUpdating = True

Exit Sub

OptionsError:

Me.Hide: MsgBox "An error occured while preparing the form", vbExclamation, MsgCaption

End Sub

**Private Sub UserForm\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

If ImageSelected Then DrawImages

ImageSelected = False

SelectedImage = 0

Label1.Caption = "What did you do today? "

End Sub

**Private Sub Image1\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(1, Me.Image1, Option1)

End Sub

Private Sub Image1\_Click()

Me.Hide

Caller = 0

Fm\_Record\_Hours.Show

End Sub

**Private Sub Image2\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(2, Me.Image2, Option2)

End Sub

Private Sub Image2\_Click()

Me.Hide

Caller = 1

Fm\_EmergencyResponse.Show

End Sub

**Private Sub Image3\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(3, Me.Image3, Option3)

End Sub

Private Sub Image3\_Click()

Me.Hide

Fm\_Leave\_Planned.Show

End Sub

**Private Sub Image4\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(4, Me.Image4, Option4)

End Sub

Private Sub Image4\_Click()

Me.Hide

Fm\_Leave\_Other.DayCountBox.Value = "1"

Fm\_Leave\_Other.Show

End Sub

**Private Sub Image5\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(5, Me.Image5, Option5)

End Sub

**Private Sub Image6\_MouseMove**(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)

Call ImageSelect(6, Me.Image6, Option6)

End Sub

**Private Sub Image5\_Click()**

Application.ScreenUpdating = False

ClearTimesheetRow (CurRow)

Cells(CurRow, TILCol) = "-" & Format((PaidHours / 24), "h:mm")

Cells(CurRow, LocationIDCol) = 0 'won't get recorded in Oracle

Cells(CurRow, ActivityCol) = "Time in lieu"

Cells(CurRow, ActivityCol).Font.Color = RGB(51, 51, 255)

Cells(CurRow, LocationCol) = "Day off"

Cells(CurRow, LocationCol).Font.Color = RGB(51, 51, 255)

Cells(CurRow, EntryDateCol) = Date 'entry date

Cells(CurRow, DutyCol) = 0

Cells(CurRow, CatagoryCol) = 5 'time in lieu

If Cells(CurRow, StatusCol) = "saved" Or Cells(CurRow, StatusCol) = "updated" Then

Cells(CurRow, StatusCol) = "updated"

Else

Cells(CurRow, StatusCol) = "entered"

End If

Me.Hide

Cells(CurRow, FundCol) = ""

UpdateWorkLifeBalance (GetPeriod(ThisDay))

'Call CalculateTimesheetSummary(GetStartDate(), GetEndDate(ActiveCell.row))

Call CalculateTimesheetSummary(0)

Application.ScreenUpdating = True

End Sub

**Private Sub Image6\_Click**()

Application.ScreenUpdating = False

ClearTimesheetRow (CurRow)

Cells(CurRow, LocationIDCol) = 0 'won't get recorded in Oracle

Cells(CurRow, ActivityCol) = Label1.Caption

Cells(CurRow, ActivityCol).Font.Color = RGB(51, 51, 255)

Cells(CurRow, LocationCol) = "Day off"

Cells(CurRow, LocationCol).Font.Color = RGB(51, 51, 255)

If Label1.Caption = "Rostered Day Off" Then

Cells(CurRow, CatagoryCol) = 6

Else

Cells(CurRow, CatagoryCol) = 7

End If

Cells(CurRow, EntryDateCol) = Date 'entry date

Cells(CurRow, DutyCol) = 0

If Cells(CurRow, StatusCol) = "saved" Or Cells(CurRow, StatusCol) = "updated" Then

Cells(CurRow, StatusCol) = "updated"

Else

Cells(CurRow, StatusCol) = "entered"

End If

Cells(CurRow, FundCol) = ""

UpdateWorkLifeBalance (GetPeriod(ThisDay))

'Call CalculateTimesheetSummary(GetStartDate(), GetEndDate(ActiveCell.row))

Call CalculateTimesheetSummary(0)

Application.ScreenUpdating = True

Me.Hide

End Sub

'-----------------------------------------------------

**Sub DrawImages**()

Dim Count As Integer

'Always visible

Image1.Left = LeftMargin

Image1.Width = NormalWidth

Image1.Height = NormalHeight

Image1.Top = ImageTop

Image1.ZOrder (1)

'Always visible

Image2.Left = Image1.Left + NormalWidth + Gap

Image2.Width = NormalWidth

Image2.Height = NormalHeight

Image2.Top = ImageTop

Image2.ZOrder (1)

Count = 2

If Image3.Visible Then

Image3.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image3.Width = NormalWidth

Image3.Height = NormalHeight

Image3.Top = ImageTop

Image3.ZOrder (1)

Count = Count + 1

End If

If Image4.Visible Then

Image4.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image4.Width = NormalWidth

Image4.Height = NormalHeight

Image4.Top = ImageTop

Image4.ZOrder (1)

Count = Count + 1

End If

If Image5.Visible Then

Image5.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image5.Width = NormalWidth

Image5.Height = NormalHeight

Image5.Top = ImageTop

Image5.ZOrder (1)

Count = Count + 1

End If

If Image6.Visible Then

Image6.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image6.Width = NormalWidth

Image6.Height = NormalHeight

Image6.Top = ImageTop

Image6.ZOrder (1)

Count = Count + 1

End If

End Sub

**Sub PresentForm**()

Dim Count As Integer

Dim W As Integer

Dim C As Long

'Always visible

Image1.Left = LeftMargin

Image1.Width = NormalWidth

Image1.Height = NormalHeight

Image1.Top = ImageTop

Image1.ZOrder (1)

'Always visible

Image2.Left = Image1.Left + NormalWidth + Gap

Image2.Width = NormalWidth

Image2.Height = NormalHeight

Image2.Top = ImageTop

Image2.ZOrder (1)

Count = 2

If Image3.Visible Then

Image3.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image3.Width = NormalWidth

Image3.Height = NormalHeight

Image3.Top = ImageTop

Image3.ZOrder (1)

Count = Count + 1

End If

If Image4.Visible Then

Image4.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image4.Width = NormalWidth

Image4.Height = NormalHeight

Image4.Top = ImageTop

Image4.ZOrder (1)

Count = Count + 1

End If

If Image5.Visible Then

Image5.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image5.Width = NormalWidth

Image5.Height = NormalHeight

Image5.Top = ImageTop

Image5.ZOrder (1)

Count = Count + 1

End If

If Image6.Visible Then

Image6.Left = LeftMargin + ((NormalWidth + Gap) \* Count)

Image6.Width = NormalWidth

Image6.Height = NormalHeight

Image6.Top = ImageTop

Image6.ZOrder (1)

Count = Count + 1

End If

For W = 175 To 75 + (Count \* 100) Step 10

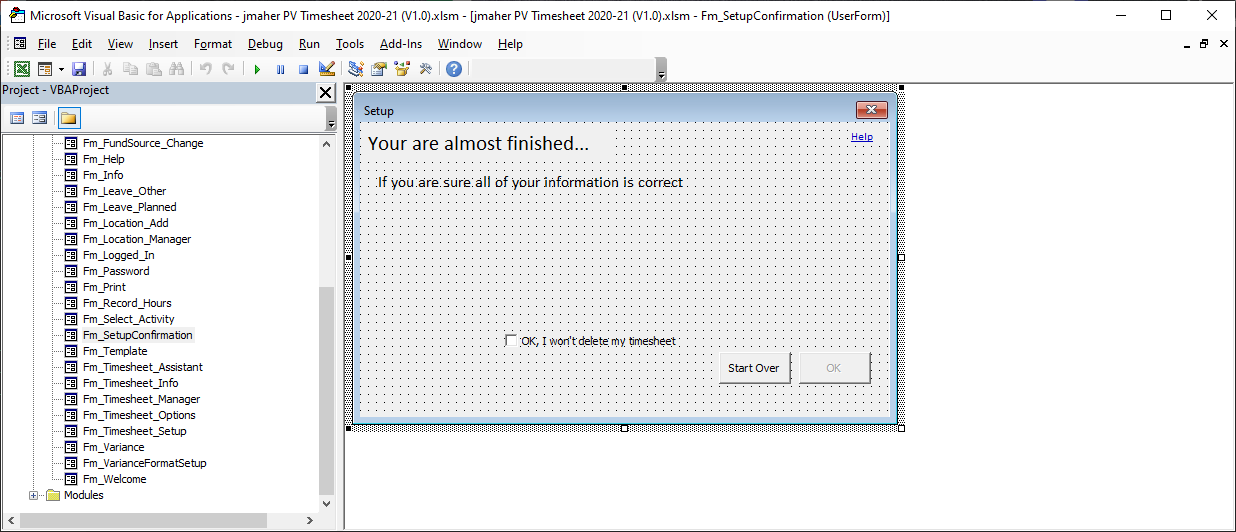
Me.Width = W

For C = 1 To 750000: Next

Next

Label1.Left = (Me.Width / 2) - (Label1.Width / 2)

End Sub



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Functions\_Routines

Option Explicit

**Function GetFolder() As String**

Dim fldr As FileDialog

Dim sItem As String

Dim strPath As String

Set fldr = Application.FileDialog(msoFileDialogFolderPicker)

With fldr

.Title = "Select where your printed timesheet will be stored"

.AllowMultiSelect = False

.InitialFileName = strPath

If .Show <> -1 Then GoTo NextCode

sItem = .SelectedItems(1)

End With

NextCode:

GetFolder = sItem

Set fldr = Nothing

End Function

**Sub ToggleFundSrcCol()**

If Columns(FundCol).Hidden Then

Columns(FundCol).Hidden = False

Else

Columns(FundCol).Hidden = True

End If

End Sub

**Function GetWeekendHours() As Single**

'returns weekend hours if weekend hours are anywhere on the PaidHours Table

'or returns the highest hours value for a weekday

Dim T1 As Single

T1 = 0

If Val(Range("PaidHours").Cells(1)) > 0 Then T1 = Val(Range("PaidHours").Cells(1))

If Val(Range("PaidHours").Cells(7)) > T1 Then T1 = Val(Range("PaidHours").Cells(7))

If Val(Range("PaidHours").Cells(8)) > T1 Then T1 = Val(Range("PaidHours").Cells(8))

If Val(Range("PaidHours").Cells(14)) > T1 Then T1 = Val(Range("PaidHours").Cells(14))

If T1 > 0 Then GetWeekendHours = T1 Else GetWeekendHours = 7.6

End Function

**Public Sub CalculateTimesheetSummary(i As Integer)**

'Update timesheet summary at bottom of timesheet

'NOTE i is used only to hide the routine

Dim SD, ED As Date

Dim ATCarried As Single 'Accrued time carried

Dim TIL As Single

Dim AT As Single 'Accrued Time

Dim N, RDOCarried, WeekendDaysWorked, RDOs As Integer

Dim RecDate As Date

Dim sh1 As Worksheet

Dim TimeStr As String

'Get the start date of the displayed timesheet

N = Range("StartDate").row

While Rows(N).Hidden

N = N + 1

Wend

SD = Cells(N, DateCol) 'Start Date

'Get the end date of the displayed timesheet

While Not Rows(N).Hidden And N < Range("EndDate").row + 1

N = N + 1

Wend

ED = Cells(N - 1, DateCol) 'End Date

'-------------------------------------------------

Set sh1 = Sheets("Timesheet")

WeekendDaysWorked = 0: RDOs = 0: AT = 0: TIL = 0

ATCarried = Val(Range("ATCarried"))

RDOCarried = Val(Range("RDOCarried"))

' Calculate opening balance for AT, RDOs as of PeriodStart Date------------------

N = Range("StartDate").row

While sh1.Cells(N, DateCol) < SD

If sh1.Cells(N, ATCol) <> "" Then ATCarried = ATCarried + TextToTime(sh1.Cells(N, ATCol))

If sh1.Cells(N, TILCol) <> "" Then TIL = TIL + TextToTime(sh1.Cells(N, TILCol))

'Get Weekend days worked and RDO totals

If Weekday(sh1.Cells(N, DateCol), 2) > 5 Then

If sh1.Cells(N, RWECol) = 1 Then WeekendDaysWorked = WeekendDaysWorked + 1 'Count weekend days worked (for RDO purposes)

Else

If Val(sh1.Cells(N, CatagoryCol)) = 6 Then RDOCarried = RDOCarried - 1 'Count RDOs

End If

N = N + 1

Wend

ATCarried = ATCarried + TIL 'This is the AT OPENING for the displayed period

Range("Timesheet\_Summary").Cells(2, 7) = TimeToText(ATCarried, 0) 'Always H:MM

TIL = 0

RDOCarried = RDOCarried + WeekendDaysWorked

Range("Timesheet\_Summary").Cells(8, 7) = RDOCarried 'Opening RDO Balance

WeekendDaysWorked = 0

'----------------- Now, Calculate totals for displayed period -----------

While sh1.Cells(N, DateCol) <= ED And N < Range("EndDate").row + 1

If sh1.Cells(N, ATCol) <> "" Then AT = AT + TextToTime(sh1.Cells(N, ATCol))

If sh1.Cells(N, TILCol) <> "" Then TIL = TIL + TextToTime(sh1.Cells(N, TILCol))

If Weekday(sh1.Cells(N, DateCol), 2) > 5 Then

If sh1.Cells(N, RWECol) = 1 Then WeekendDaysWorked = WeekendDaysWorked + 1 'Count weekend days worked (for RDO purposes)

Else

If Val(sh1.Cells(N, CatagoryCol)) = 6 Then RDOs = RDOs + 1 'Count RDOs

End If

N = N + 1

Wend

Range("Timesheet\_Summary").Cells(3, 7) = TimeToText(AT, 0)

Range("Timesheet\_Summary").Cells(4, 7) = TimeToText(TIL, 0)

Range("Timesheet\_Summary").Cells(5, 7) = TimeToText(ATCarried + AT + TIL, 0)

Range("Timesheet\_Summary").Cells(8, 7) = RDOCarried

Range("Timesheet\_Summary").Cells(9, 7) = WeekendDaysWorked

Range("Timesheet\_Summary").Cells(10, 7) = RDOs

Range("Timesheet\_Summary").Cells(11, 7) = RDOCarried + WeekendDaysWorked - RDOs

End Sub

**Function ActivityExists(ActivityName As String) As Boolean**

'Check for duplicate name

Dim N% 'Integer

N = 2

Do While Range("ActivityList").Cells(N, 1) <> ""

If LCase(ActivityName) = LCase(Range("ActivityList").Cells(N, 1)) Then ActivityExists = True: Exit Do

N = N + 1

Loop

End Function

**Function PadString(S As String, N As Integer) As String**

While Len(S) < N

S = S & " "

Wend

PadString = S

End Function

**Function GetPublicHoliday(D As Date) As String**

Dim N As Integer

For N = 1 To Range("PublicHolidays").Cells.Count

If Range("PublicHolidays").Cells(N) = D Then

GetPublicHoliday = Range("PublicHolidays").Cells(N, 0)

Exit For

End If

Next

End Function

**Function IsPublicHoliday(D As Date) As Boolean**

Dim N As Integer

IsPublicHoliday = False

For N = 1 To Range("PublicHolidays").Cells.Count

If Range("PublicHolidays").Cells(N) = D Then

IsPublicHoliday = True

Exit For

End If

Next

End Function

**Function GetPeriodDayNumber(D As Date) As Integer**

'Get the day number in a pay period for a given date

Dim N As Integer

N = 1

While Range("PeriodTable").Cells(N, 3) < D

N = N + 1

Wend

GetPeriodDayNumber = DateDiff("d", Range("PeriodTable").Cells(N), D) + 1

End Function

**Function GetPassword(C As String) As String**

Fm\_Password.Label1.Caption = C

Fm\_Password.Show

GetPassword = Fm\_Password.PasswordBox

End Function

**Function GetPeriod(D As Date) As Integer**

'Get the pay period number for a given date

Dim p As Integer

p = Month(D)

If p > 6 Then

GetPeriod = p - 6

Else

GetPeriod = p + 6

End If

End Function

**Function GetFundSourceNumber(S As String) As String**

Dim N As Integer

N = 2

GetFundSourceNumber = "000" 'Default return value

Do While Range("FundSourceList").Cells(N, 1) <> ""

If Range("FundSourceList").Cells(N, 2) = S Then

GetFundSourceNumber = Range("FundSourceList").Cells(N, 1)

Exit Do

End If

N = N + 1

Loop

End Function

**Function TextToTime(T As String) As Single**

'Convert a time string e.g. "4:25" to single precision integer

If T = vbNullString Then TextToTime = 0: Exit Function

Dim IsNeg As Boolean

Dim colon As Integer

Dim M, Hours, mins As Single

If Left(T, 1) = "-" Then

IsNeg = True

T = Right(T, Len(T) - 1) 'remove negative sign

Else

IsNeg = False

End If

If InStr(T, ":") > 0 Then

'Is there a colon?

colon = InStr(T, ":") 'get location of the colon in the string

If colon = 0 Then colon = Len(T) + 1: T = T & ":00" 'eg user typed '12' instead of 12:00

Hours = Val(Left(T, colon - 1))

If Hours > 23 Then Hours = 0

mins = Val(Right(T, Len(T) - colon))

If mins > 59 Then mins = 0

M = Hours \* 60

M = M + mins

If IsNeg Then M = M \* -1

TextToTime = M / 60

Else

Hours = Round(CSng(T), 2)

If IsNeg Then Hours = Hours \* -1

'mins = Round(Hours - Fix(T), 2)

TextToTime = Hours

End If

End Function

**Function TimeToText(T As Single, F As Integer) As String** 'F=Format Flag, 0=HMS, 1=Dec/HMS depending on user option [Range("VarianceFormat")]

If F = 1 And Range("VarianceFormat") = "decimal" Then

TimeToText = Format(T, "0.00")

Else

'if the number is negative, return "-" prefix

Dim H, M, R As String

Dim S As Single

S = T

If S < 0 Then R = "-": S = S \* -1 Else R = ""

H = Format(CStr(Fix(S)), "0")

S = Round(S - Fix(S), 2)

M = Format(S \* 60, "00")

TimeToText = R & H & ":" & M

End If

'JJJJ

End Function

**Function GetFundSourceName(S As String) As String**

Dim N As Integer

N = 2

Do While Range("FundSourceList").Cells(N, 1) <> ""

If S = Range("FundSourceList").Cells(N, 1) Then

GetFundSourceName = Range("FundSourceList").Cells(N, 2)

Exit Do

End If

N = N + 1

Loop

End Function

**Function ActivityIsUsed(ActivityName As String) As Boolean**

Dim row As Integer

row = Sheets("Timesheet").Range("StartDate").row

Do While IsDate(Sheets("Timesheet").Cells(row, DateCol))

If Sheets("Timesheet").Cells(row, ActivityCol) = ActivityName Then ActivityIsUsed = True: Exit Do

row = row + 1

Loop

End Function

**Function GetProgramName(S As String) As String**

Dim N As Integer

N = 2

Do While Range("ProgramList").Cells(N, 2) <> ""

If S = Range("ProgramList").Cells(N, 2) Then

S = Range("ProgramList").Cells(N, 3)

Exit Do

End If

N = N + 1

Loop

N = 2

Do While Range("ProgramList").Cells(N, 2) <> ""

If S = Range("ProgramList").Cells(N, 2) Then

GetProgramName = PadString(Range("ProgramList").Cells(N, 1), 5) & " " & Range("ProgramList").Cells(N, 2) & " " & Range("ProgramList").Cells(N, 4)

Exit Do

End If

N = N + 1

Loop

End Function

**Function getActivityRowNumber(ActivityName As String) As Integer**

Dim N As Integer

getActivityRowNumber = 0

N = 2

Do While Range("ActivityList").Cells(N, 1) <> ""

If Range("ActivityList").Cells(N, 1) = ActivityName Then

getActivityRowNumber = N

Exit Do

End If

N = N + 1

Loop

End Function

**Function GetLocationID(L As String) As Long**

Dim N As Integer

GetLocationID = 0

N = 2

Do While Range("PVLocationList").Cells(N, 1) <> ""

If L = Range("PVLocationList").Cells(N, 2) Then

GetLocationID = Range("PVLocationList").Cells(N)

Exit Do

End If

N = N + 1

Loop

End Function

**Function GetLocationName(L As Long) As String**

Dim N As Integer

GetLocationName = ""

N = 2

Do While Range("PVLocationList").Cells(N, 1) <> ""

If L = Range("PVLocationList").Cells(N, 1) Then

GetLocationName = Range("PVLocationList").Cells(N, 2)

Exit Do

End If

N = N + 1

Loop

End Function

**Private Function GetData(Path, File, Sheet, Address)**

Dim Data$

Data = "'" & Path & "[" & File & "]" & Sheet & "'!" & \_

Range(Address).Range("A1").Address(, , xlR1C1)

GetData = ExecuteExcel4Macro(Data)

End Function

**Function Get\_Program\_Map\_To(S As String) As String**

'Returns remapped Program number.

'If the Program number cannor be found then the value passed is returned

Get\_Program\_Map\_To = S

Dim N As Integer

N = 2

Do While Range("ProgramList").Cells(N, 2) <> ""

If Range("ProgramList").Cells(N, 2) = S Then

Get\_Program\_Map\_To = Range("ProgramList").Cells(N, 3)

Exit Do

End If

N = N + 1

Loop

End Function

**Public Sub UpdateWorkLifeBalance(period As Integer**) 'Update work-life balance info for the period

Dim N, Trow, Pd As Integer

Dim ActivityNumber, WeekendsWorked As Integer

Dim Var, VarV, ATBal As Single

Dim sh1 As Worksheet

Set sh1 = Sheets("Timesheet")

'\*\*\*\*\* Work Life Table Column References \*\*\*\*\*

Const ND = 2 'Normal duty

Const ER = 3 'Emergency Response

Const OD = 4 'Off duty column

Const AL = 5 'Annual Leave

Const OL = 6 'Other Leave

Const WE = 7 'Weekeend days worked

Const RDO = 8 'rostered days off

Const RDOB = 9 ' rostered days balance

Const CS = 10 'Community Service

Const OT = 11 'Overtime

Const AT = 12 'Accrued Time

Const TIL = 13 'Time in lieu

Const ATB = 14 'At balance

Const STAT = 15 ' status column ( saved, updated)

Application.ScreenUpdating = False

'clear the row for the period on the work-life balance sheet

For N = 2 To 14

Range("WorkTable").Cells(period, N).ClearContents

Next

Range("ATOpen") = TimeToText(Range("ATCarried"), 0) '0 = return as HH:MM

Range("RDOOpen") = Range("RDOCarried")

Range("WorkTable").Cells(period, STAT) = "updated" 'flag that row has been updated

'-------------------------------------------------------------

For Trow = Range("StartDate").row To Range("EndDate").row 'All timesheet rows

Pd = GetPeriod(sh1.Cells(Trow, DateCol)) 'Period

If Pd = period Then

'Process any Variance

'Time in lieu

If sh1.Cells(Trow, TILCol) <> vbNullString Then

VarV = (TextToTime(sh1.Cells(Trow, TILCol)) / 24) \* -1

Range("WorkTable").Cells(period, TIL) = Range("WorkTable").Cells(period, TIL) + VarV

End If

'Accrued Time

If sh1.Cells(Trow, ATCol) <> vbNullString Then

VarV = TextToTime(sh1.Cells(Trow, ATCol)) / 24

Range("WorkTable").Cells(period, AT) = Range("WorkTable").Cells(period, AT) + VarV

End If

'Overtime

If sh1.Cells(Trow, OTCol) <> vbNullString Then

VarV = TextToTime(sh1.Cells(Trow, OTCol)) / 24

Range("WorkTable").Cells(period, OT) = Range("WorkTable").Cells(period, OT) + VarV

End If

'Community Service

If sh1.Cells(Trow, CSCol) <> vbNullString Then

VarV = TextToTime(sh1.Cells(Trow, CSCol)) / 24

Range("WorkTable").Cells(period, CS) = Range("WorkTable").Cells(period, CS) + (VarV)

End If

'Rostered Weekend days

If sh1.Cells(Trow, RWECol) = 1 Then

Range("WorkTable").Cells(period, WE) = Val(Range("WorkTable").Cells(period, WE)) + 1 'weekend day worked

End If

'Type of Duty

ActivityNumber = sh1.Cells(Trow, CatagoryCol)

Select Case ActivityNumber

Case 1: 'PV Duty

Range("WorkTable").Cells(period, ND) = Val(Range("WorkTable").Cells(period, ND)) + 1 'Total days PV duty

Case 2: 'Emergency Response

Range("WorkTable").Cells(period, ER) = Val(Range("WorkTable").Cells(period, ER)) + 1

Case 3: 'Planned Leave

Range("WorkTable").Cells(period, AL) = Range("WorkTable").Cells(period, AL) + 1

Case 4: 'incidental Leave

Range("WorkTable").Cells(period, OL) = Val(Range("WorkTable").Cells(period, OL)) + 1

Case 5: 'Personal Time (Time in lieu)

Case 6: 'RDO

Range("WorkTable").Cells(period, RDO) = Val(Range("WorkTable").Cells(period, RDO)) + 1 'RDO

Range("WorkTable").Cells(period, OD) = Val(Range("WorkTable").Cells(period, OD)) + 1 'Off Duty

Case Else

Range("WorkTable").Cells(period, OD) = Val(Range("WorkTable").Cells(period, OD)) + 1 'IF not PV work or ER work then must be off duty

End Select

End If

Next

Range("WorkTable").Cells(1, ATB) = Range("AtOpen") + Range("WorkTable").Cells(1, AT) - Range("WorkTable").Cells(1, TIL)

'Changed RDO (below) to show RDOs for all staff (some staff change Rostered Duty during year)

Range("WorkTable").Cells(1, RDOB) = Val(Range("RDOOpen")) + Val(Range("WorkTable").Cells(1, WE)) - Val(Range("WorkTable").Cells(1, RDO))

WeekendsWorked = Val(Range("WorkTable").Cells(1, WE))

'Update WorkLife page

For N = 2 To 12

Range("WorkTable").Cells(N, ATB) = Range("WorkTable").Cells(N - 1, ATB) + Range("WorkTable").Cells(N, AT) - Range("WorkTable").Cells(N, TIL)

Range("WorkTable").Cells(N, RDOB) = Val(Range("WorkTable").Cells(N - 1, RDOB)) + Val(Range("WorkTable").Cells(N, WE)) - Val(Range("WorkTable").Cells(N, RDO))

WeekendsWorked = WeekendsWorked + Val(Range("WorkTable").Cells(N, WE))

Next N

'Update Admin page

Range("RDOBalance") = Val(Range("WorkTable").Cells(12, RDOB))

Range("ATBalance") = Range("WorkTable").Cells(12, ATB) \* 24

Range("WeekendsWorked") = WeekendsWorked

Application.ScreenUpdating = True

End Sub

**Public Sub LoadPersonInformation(uID As Long) 'V7.8b**

Dim Cmd, RWE, RWER As String

Dim TotalHours As Single

Dim N As Integer

InfoLoadedOK = False

Call OpenPVTSConnection(tmode)

If Not ConOpen Then Exit Sub

On Error GoTo dberr

Set RS = New Adodb.Recordset

Cmd = "select \* from tmsheet.pv\_employee\_hours\_v "

Cmd = Cmd & "WHERE PERSON\_ID = " & uID & ""

RS.Open Cmd, conn, adOpenStatic, adLockReadOnly

If RS.RecordCount <> 1 Then

'This is an error situation. Alert user

RS.Close: conn.Close

Set conn = Nothing: Set RS = Nothing

MsgBox "Critical: You must contact the Service Desk. " & vbCr & "PERSON\_ID = " & uID & " NOT found in VIEW pv\_employee\_hours. " & vbCr & "No employee record not found. Request an 'offline template'. ", vbExclamation, MsgCaption

Call logDataEntry("LoadPersonInformation > Sub Failed: Unable to identify user (pv\_employee\_hours). SQL > " & Cmd & " (" & ActiveWorkbook.FullName & ")")

Exit Sub

End If

If IsNull(RS.Fields("W1\_Day1\_Hours")) Then Range("PaidHours").Cells(1) = 0 Else Range("PaidHours").Cells(1) = Round(Val(RS.Fields("W1\_Day1\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day2\_Hours")) Then Range("PaidHours").Cells(2) = 0 Else Range("PaidHours").Cells(2) = Round(Val(RS.Fields("W1\_Day2\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day3\_Hours")) Then Range("PaidHours").Cells(3) = 0 Else Range("PaidHours").Cells(3) = Round(Val(RS.Fields("W1\_Day3\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day4\_Hours")) Then Range("PaidHours").Cells(4) = 0 Else Range("PaidHours").Cells(4) = Round(Val(RS.Fields("W1\_Day4\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day5\_Hours")) Then Range("PaidHours").Cells(5) = 0 Else Range("PaidHours").Cells(5) = Round(Val(RS.Fields("W1\_Day5\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day6\_Hours")) Then Range("PaidHours").Cells(6) = 0 Else Range("PaidHours").Cells(6) = Round(Val(RS.Fields("W1\_Day6\_Hours")), 2)

If IsNull(RS.Fields("W1\_Day7\_Hours")) Then Range("PaidHours").Cells(7) = 0 Else Range("PaidHours").Cells(7) = Round(Val(RS.Fields("W1\_Day7\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day1\_Hours")) Then Range("PaidHours").Cells(8) = 0 Else Range("PaidHours").Cells(8) = Round(Val(RS.Fields("W2\_Day1\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day2\_Hours")) Then Range("PaidHours").Cells(9) = 0 Else Range("PaidHours").Cells(9) = Round(Val(RS.Fields("W2\_Day2\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day3\_Hours")) Then Range("PaidHours").Cells(10) = 0 Else Range("PaidHours").Cells(10) = Round(Val(RS.Fields("W2\_Day3\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day4\_Hours")) Then Range("PaidHours").Cells(11) = 0 Else Range("PaidHours").Cells(11) = Round(Val(RS.Fields("W2\_Day4\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day5\_Hours")) Then Range("PaidHours").Cells(12) = 0 Else Range("PaidHours").Cells(12) = Round(Val(RS.Fields("W2\_Day5\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day6\_Hours")) Then Range("PaidHours").Cells(13) = 0 Else Range("PaidHours").Cells(13) = Round(Val(RS.Fields("W2\_Day6\_Hours")), 2)

If IsNull(RS.Fields("W2\_Day7\_Hours")) Then Range("PaidHours").Cells(14) = 0 Else Range("PaidHours").Cells(14) = Round(Val(RS.Fields("W2\_Day7\_Hours")), 2)

TotalHours = Range("PaidHours").Cells(1) + Range("PaidHours").Cells(2) + Range("PaidHours").Cells(3) \_

+ Range("PaidHours").Cells(4) + Range("PaidHours").Cells(5) + Range("PaidHours").Cells(6) \_

+ Range("PaidHours").Cells(7) + Range("PaidHours").Cells(8) + Range("PaidHours").Cells(9) \_

+ Range("PaidHours").Cells(10) + Range("PaidHours").Cells(11) + Range("PaidHours").Cells(12) \_

+ Range("PaidHours").Cells(13) + Range("PaidHours").Cells(14)

Range("TotalHours") = Round(TotalHours, 2)

RWE = RS.Fields("ROSTER\_WEEKEND")

If Len(RWE) > 4 Then 'Rostered

RWE = Right(RWE, Len(RWE) - 4)

If Len(RWE) > 2 Then RWE = Left(RWE, 2)

N = Val(RWE)

If N = 0 Then N = 1 'if N=1 then Rostered employee working 'As Worked' weekends

Else

N = 0 'Non-Rostered employee

End If

Range("RosteredDays") = N

RS.Close

Call logDataEntry("LoadPersonInformation > Rostered days OK: " & Range("RosteredDays") & " (" & ActiveWorkbook.FullName & ")")

'-------- ---------------

Cmd = "select Last\_Name, First\_Name, Job\_Title, Office\_ID, OFFICE "

Cmd = Cmd & "FROM PV\_EMPLOYEE\_DETAILS\_MV "

Cmd = Cmd & "WHERE Person\_ID = " & uID & ""

RS.Open Cmd, conn, adOpenStatic, adLockReadOnly

If RS.RecordCount <> 1 Then

'This is an error situation. Alert user

RS.Close

conn.Close

Set conn = Nothing: Set RS = Nothing

MsgBox "Critical: You must contact the Service Desk. " & vbCr & "PERSON\_ID = " & uID & " NOT found in VIEW pv\_employee\_details\_mv. " & vbCr & "No employee record not found. Request an 'offline template'. ", vbExclamation, MsgCaption

Call logDataEntry("LoadPersonInformation > Sub Failed: Cannot identify user (PV\_EMPLOYEE\_DETAILS\_MV). SQL > " & Cmd & " (" & ActiveWorkbook.FullName & ")")

Exit Sub

End If

Range("FirstName") = RS.Fields("First\_Name")

Range("LastName") = RS.Fields("Last\_Name")

Range("Position") = RS.Fields("Job\_Title")

Range("WorkCentre") = RS.Fields("Office\_ID")

Range("Office") = RS.Fields("OFFICE")

Range("PersonID") = uID

RS.Close

conn.Close

Set conn = Nothing: Set RS = Nothing

InfoLoadedOK = True

Call logDataEntry("LoadPersonInformation > Done. InfoLoadedOK = True (" & ActiveWorkbook.FullName & ")")

Exit Sub

dberr:

If conn.State <> 0 Then conn.Close

Set conn = Nothing: Set RS = Nothing

MsgBox "Timesheet Error " & Err.Description, vbInformation, MsgCaption

Call logDataEntry("LoadPersonInformation > Sub Failed: " & MsgCaption & " - " & Err.Description & " (" & ActiveWorkbook.FullName & ")")

End Sub

**Public Function ImportActivities(FileLocation As String) As Boolean**

'---------------------------------------------------------------

'1st Activity Row = 8

'Activity Columns = 2 to 15

'Import data saved on Sheets("PV")

'STOP import when 1st column = "Approved Leave"

'---------------------------------------------------------------

Dim NotImported As String

Dim ImportComplete As Boolean

Dim ImportPath, FileName, FilePath, Address As String

Dim R, N, row, Column As Integer

Dim ImportData As Variant

Dim ActivityName As String

Const SheetName = "Activity"

Dim sh1 As Worksheet

Set sh1 = Sheets("Activity")

NotImported = vbNullString

On Error GoTo ImpErr

If Dir(FileLocation, vbDirectory) = vbNullString Then FileLocation = vbNullString

If FileLocation = vbNullString Then

Dim getFile As Variant

MsgBox "IMPORT ACTIVITIES" & vbCr & vbCr & \_

"When you click 'OK' a file location window will open." & vbCr & \_

"Navigate to the folder where the Timesheet you want" & vbCr & \_

"to import Activities from is located, select the Timesheet" & vbCr & \_

"file then click 'Open'", vbInformation, MsgCaption

getFile = Application.GetOpenFilename("Image Files (\*.xlsm),\*.xlsm", , "Please select the Timesheet file to open", , "False")

If getFile = "" Or getFile = False Then 'Exit if user cancelled

MsgBox "Activity import cancelled", vbExclamation, MsgCaption

Call logDataEntry("ImportActivities > User failed to select a file from the dialog. ")

ImportActivities = True

Exit Function

End If

FileLocation = CStr(getFile)

End If

N = InStrRev(FileLocation, "\"): If N = 0 Then Exit Function 'ERROR

ImportPath = Left(FileLocation, N)

FileName = Right(FileLocation, Len(FileLocation) - N)

Column = 2

row = 8 ' Row of first Activity in Timesheet.

Address = Cells(row - 1, Column).Address

ActivityName = GetData(ImportPath, FileName, SheetName, Address)

If Left(ActivityName, 3) <> "BOF" Then

MsgBox "Cannot locate Activity List in source file. Cannot continue.", vbExclamation, MsgCaption

Call logDataEntry("ImportActivities > Failed to locate Activity List in source file. Cannot continue. [" & ImportPath & "\" & FileName & "]" & SheetName & "!" & Address & "")

Exit Function

End If

Application.ScreenUpdating = False

GetActivity:

Column = 2

Address = Cells(row, Column).Address

ActivityName = GetData(ImportPath, FileName, SheetName, Address)

If ActivityName = "Approved Leave" Then GoTo IComplete 'All custom activiries have been imported

If ActivityExists(ActivityName) Then

NotImported = NotImported & vbCr & ActivityName

row = row + 1

GoTo GetActivity

End If

'insert a new activity row

sh1.Range("B" & row & ":O" & row).Insert Shift:=xlDown

sh1.Cells(row, 2) = ActivityName

For Column = 3 To 14 'Activity columns

Address = Cells(row, Column).Address

ImportData = GetData(ImportPath, FileName, SheetName, Address)

If Not (ImportData = 0 Or ImportData = vbNullString) Then

If Column Mod 2 <> 0 Then

sh1.Cells(row, Column) = Get\_Program\_Map\_To(CStr(ImportData))

Else

sh1.Cells(row, Column) = ImportData

End If

End If

Next Column

sh1.Cells(row, 15) = "open"

row = row + 1

If Not ImportComplete Then GoTo GetActivity

IComplete:

Application.ScreenUpdating = True

ImportActivities = True

Call logDataEntry("ImportActivities > Done. ImportActivities = True. Source " & ImportPath & "\" & FileName & "")

If NotImported <> vbNullString Then

Call logDataEntry("ImportActivities > The following activities were not imported as they already exist:" & NotImported & " (" & ActiveWorkbook.FullName & ")")

MsgBox "The following activities were not imported as they already exist:" & vbCr & NotImported, vbInformation, MsgCaption

End If

Exit Function

ImpErr:

ImportActivities = False

Application.ScreenUpdating = True

Call logDataEntry("ImportActivities > Sub Failed (" & FileLocation & ") " & ImportPath & "\" & FileName & ", " & ActiveWorkbook.FullName)

End Function

**Sub UpdateProgramList(i As Integer)** ' i used to hide routine ONLY

Dim Cmd As String

Dim X, C, R As Integer

If LCase(Range("TSVersion")) = "test" Then

ProgramListUpToDate = True

Exit Sub

End If

OpenPVTSConnection (tmode)

If Not ConOpen Then Exit Sub

On Error GoTo oraerror

Application.ScreenUpdating = False

'Get the program list from the server

Cmd = "SELECT Program\_Output, Program\_ID, Map\_To,Program\_Name from TMSHEET.PV\_TS\_PROGRAM\_T"

Cmd = Cmd & " ORDER BY Program\_Output, Program\_ID"

Set RS = New Adodb.Recordset

RS.Open Cmd, conn, adOpenStatic, adLockReadOnly

If RS.RecordCount > 0 Then

R = Range("ProgramList").row + 1

C = Range("ProgramList").Column

While Sheets("Admin").Cells(R, C) <> ""

Sheets("Admin").Cells(R, C).ClearContents

Sheets("Admin").Cells(R, C + 1).ClearContents

Sheets("Admin").Cells(R, C + 2).ClearContents

Sheets("Admin").Cells(R, C + 3).ClearContents

R = R + 1

Wend

Sheets("Admin").Cells(Range("ProgramList").row + 1, C).CopyFromRecordset RS

ProgramListUpToDate = True

Else

MsgBox "Program List not available "

End If

RS.Close: Set RS = Nothing

conn.Close: Set conn = Nothing

Application.ScreenUpdating = True

Exit Sub

oraerror:

If conn.State <> 0 Then conn.Close

Set conn = Nothing: Set RS = Nothing

Application.ScreenUpdating = True

MsgBox "Program list update error", vbExclamation, MsgCaption

End Sub

**Sub UpdatePublicHolidayList(i As Integer**) ' i used to hide routine ONLY

Dim Cmd As String

Dim N, X, Holcol, HolRow, DayRow As Integer

'if ispublicholiday

HolidayListUpToDate = False

OpenPVTSConnection (tmode)

If Not ConOpen Then Exit Sub

Set RS = New Adodb.Recordset

Application.ScreenUpdating = False

On Error GoTo oraerror

'Get the public Holiday list from the server

Cmd = "SELECT HolidayName, HolidayDate "

Cmd = Cmd & "from PV\_TS\_PublicHolidays\_T "

Cmd = Cmd & "WHERE HolidayDate >= to\_date('01-Jul-2020','DD-MON-YYYY') and HolidayDate <= to\_date('30-Jun-2021','DD-MON-YYYY') "

Cmd = Cmd & "ORDER BY HolidayDate "

RS.Open Cmd, conn, adOpenStatic, adLockReadOnly

N = RS.RecordCount

If N > 0 Then

Holcol = Sheets("Admin").Range("PublicHolidays").Column

HolRow = Sheets("Admin").Range("PublicHolidays").row

While Sheets("Admin").Cells(HolRow, Holcol) <> "" ' R = Range("PublicHolidays").row To Range("PublicHolidays").row + Range("PublicHolidays").Rows.Count - 2

Sheets("Admin").Cells(HolRow, Holcol - 1).ClearContents

Sheets("Admin").Cells(HolRow, Holcol).ClearContents

HolRow = HolRow + 1

Wend

Sheets("Admin").Cells(Range("PublicHolidays").row, Holcol - 1).CopyFromRecordset RS

RS.Close: Set RS = Nothing

conn.Close: Set conn = Nothing

DayRow = Sheets("Timesheet").Range("StartDate").row

While Sheets("Timesheet").Cells(DayRow, DateCol) <> ""

If IsDate(Sheets("Timesheet").Cells(DayRow, DateCol)) Then

If IsPublicHoliday(Sheets("Timesheet").Cells(DayRow, DateCol)) Then

If Sheets("Timesheet").Cells(DayRow, NotesCol) = "" Then

Sheets("Timesheet").Cells(DayRow, NotesCol) = GetPublicHoliday(Sheets("Timesheet").Cells(DayRow, DateCol))

End If

End If

End If

DayRow = DayRow + 1

Wend

HolidayListUpToDate = True

Else

Call logDataEntry("UpdatePublicHolidayList() > " & "Public Holiday list update failed: No data found in [PV\_TS\_PublicHolidays\_T]")

MsgBox "NOTE: Public Holiday List not updated. "

End If

Application.ScreenUpdating = True

Exit Sub

oraerror:

Call logDataEntry("UpdatePublicHolidayList() > " & "Public Holiday list update failed: " & Err.Description)

If conn.State <> 0 Then conn.Close

Set conn = Nothing: Set RS = Nothing

Application.ScreenUpdating = True

MsgBox "Public Holiday list update error" & vbCr & Err.Description, vbExclamation, MsgCaption

End Sub

**Sub LoadLocationList**()

Dim Cmd As String

Dim C, R, N, W As Integer

Dim defaultID As Long

Dim Match As Boolean

If LCase(Range("TSVersion")) = "test" Then Exit Sub

Call OpenPVTSConnection(tmode)

If Not ConOpen Then Exit Sub

On Error GoTo oraerror

defaultID = GetLocationID(Range("DefaultLocation")) 'chg170926

Application.ScreenUpdating = False

'Get the location list from the server

Cmd = "SELECT DISTINCT LOCATION\_ID, LOCATION\_CODE FROM TMSHEET.PV\_LOCATION\_MV"

Cmd = Cmd & " ORDER BY LOCATION\_CODE"

Set RS = New Adodb.Recordset

RS.Open Cmd, conn, adOpenStatic, adLockReadOnly

If RS.RecordCount > 0 Then

R = Range("PVLocationList").row + 1

C = Range("PVLocationList").Column

While Sheets("Admin").Cells(R, C) <> ""

Sheets("Admin").Cells(R, C).ClearContents

Sheets("Admin").Cells(R, C + 1).ClearContents

R = R + 1

Wend

Sheets("Admin").Cells(Range("PVLocationList").row + 1, C).CopyFromRecordset RS

N = 2

While Range("PVLocationList").Cells(N, 1) <> "": N = N + 1: Wend

Range("PVLocationList").Cells(N, 1) = 1: Range("PVLocationList").Cells(N, 2) = "Work from Home"

N = N + 1

Range("PVLocationList").Cells(N, 1) = 2: Range("PVLocationList").Cells(N, 2) = "Other Agency Office"

N = N + 1

Range("PVLocationList").Cells(N, 1) = 3: Range("PVLocationList").Cells(N, 2) = "F&E Deployment"

N = N + 1

Range("PVLocationList").Cells(N, 1) = 4: Range("PVLocationList").Cells(N, 2) = "Remote Location"

'Check if "Default Location" in Timesheet Assistant is up to date (or deleted)

Range("DefaultLocation") = GetLocationName(defaultID) 'chg170926

'Check "My Locations" for and changes / deletes

N = 2

While Range("MyLocationList").Cells(N, 1) <> ""

W = 2

Do While Range("PVLocationList").Cells(W, 1) <> ""

If Range("PVLocationList").Cells(W, 1) = Range("MyLocationList").Cells(N, 1) Then

Match = True

Exit Do

End If

W = W + 1

Loop

If Not Match Then

Sheets("Admin").Range("MyLocationList").Cells(N, 1).Delete Shift:=xlUp

End If

N = N + 1

Wend

LocationListUpToDate = True

Else

Call logDataEntry("LoadLocationList > Fail > Cannot find Location list on server")

MsgBox "Cannot find Location list on server", vbExclamation, MsgCaption

End If

RS.Close

conn.Close

Set conn = Nothing: Set RS = Nothing

Application.ScreenUpdating = True

Exit Sub

oraerror:

If conn.State <> 0 Then conn.Close

Set conn = Nothing: Set RS = Nothing

Application.ScreenUpdating = True

Call logDataEntry("LoadLocationList > Location list update failed > ")

MsgBox "Location list update error", vbExclamation, MsgCaption

End Sub

'Constants

Global Const tmode = "prod" ' change to "dev" for test. "prod" for release

Global Const Pword = ""

Global Const MsgCaption = "Parks Victoria Timesheet"

Global Const OraFinYear = "2020-21"

Global Const FinYear = "2020-2021"

Global Const Grey = &H8000000F

Global Const Red = &HC0E0FF

Global Const Green = &HC0FFC0

Global Const White = &H80000005

Global Const Buff = &HC0FFFF

Global Const DarkRed = &HC0&

Global Const Black = &H80000006

Public ActNum As Integer 'Activity Number - used by Fm\_Activity\_Info and Fm\_Acrivity\_New

Public CancelTimeEntry As Boolean

Public PubHol As Boolean

Public Caller As Integer

Public LocationListUpToDate As Boolean

Public FundSourceListUpToDate As Boolean

Public InfoLoadedOK As Boolean

Public ConOpen As Boolean

Public savedConnString As String ' Oracle connectionString (remote)

Public EditActivityName As String 'Name of Activity being edited

Public ProgramListUpToDate As Boolean

Public HolidayListUpToDate As Boolean

Public SetupRun As Boolean 'True when the setup routine runs. Stops the "UpdatedatabaseRecords" routine running.

Public TemplateFile As Boolean