Software Design Notes

1. Revision History

Rev.	Revision Notes	ECO	Date
A	Initial Release	N/A	21NOV2022
B	Extended existing sections to improve focus on the purpose and scope of each test. Added COMPONENTS UNDER TEST section to collect the information needed to generate a failure-mode traceability report to show what the test plan does and does not cover.	1012	01DEC2022
C	Need to add REVERT TO DATABASE button to the form and EDIT DATABASE button to the spreadsheet. This will allow test plans to be edited from Excel and signed using the form. Need to add an UPDATE FORMS button in the spreadsheet. Need to add a test plan version tracking ability. So that test plans can be changed and automatic regression testing can be identified. Need to make a TESTbase Generator Form. create templates for each type of test, with standard nomenclature, so that the test can be written by adding signal points, reference designators, and component and circuit parameter values.	IDEAS	

2. Introduction

2.1. Purpose

Present software design notes for the VBA enabling a simple EE Test Documentation system.

2.2. Scope

This document applies to the VBA subroutines listed in Table 1 – Button Click Subroutines.

Table 1 – Button Click Subroutines

Subroutine	Description
saveButton_Click()	• Saves Word document • Creates PDF document • Creates/updates log entry
saveAndCloseButton_Click()	• Saves and closes Word document • Creates PDF document • Creates/updates log entry
closeWithoutSavingButton_Click()	• Closes Word document without saving • Creates PDF document • Does not update log
deleteButton_Click()	• Deletes Word document • Creates PDF document • Deletes log entry
refreshData_Click()	• Updates data for analysis

2.3. File Locations

All Project Files

- C:\Users\nolan\Documents\BexarVBA\994 TESTbase

3. Software Specification Requirements

SSR1 – Usability
No text (title)

SSR1.1
Able to be run from a computer with Microsoft Office.

SSR1.2
Only requires Word, Excel and a PDF viewer.

SSR1.3
Able to be setup on a USB drive (i.e. thumb drive).

SSR1.4
Does not require password to use the document system.

SSR1.5
Able to functionally access through the Autodesk EAGLE control panel project navigator.

SSR1.6
Able to backup to a GitHub repository.

SSR2 – Functionality

No text (title)

SSR2.1

Saves files in folders described in Table 2 – Folder Structure.

Table 2 – Folder Structure

Folder	Contents
…\TESTbase	• Word template for creating new catalog entries • Word template for creating new inventory entries • Excel file with data analysis
…\TESTbase\Tests	• Word documents
…\TESTbase\Tests\PDFs	• PDF documents
~~…\TESTbase\Test Reports~~	~~• Word documents~~
~~…\TESTbase\Test Reports\PDFs~~	~~• PDF documents~~
…\TESTbase\data	• Excel file with data • This folder is hidden

SSR2.2

Functions properly when VBA macros are enabled and the VBA project model is trusted.

SSR2.3

Protect the VBA source code with the password in Table 3 – Password.

Table 3 – Password

Password	9hvmq6KchrZ+V&&j

SSR2.4
Excel Files (title)

SSR2.4.1
Data workbook file placed in the hidden **…\TESTbase\data** folder.

SSR2.4.2
Analysis workbook file placed in the **…\TESTbase** folder.

SSR2.4.3
Display one (1) button that runs a VBA macro function listed in Table 4 – Excel Button Macros

Table 4 – Excel Button Macros

Button	Macro
Refresh	refreshData_Click()

SSR2.4.4
Protect the data workbook file's data sheet with the password in Table 5 – Password.

Table 5 – Password

Password	WT$?@L&x=F5qYfC#

SSR2.5
PDF Documents (title)

SSR2.5.1
Create a PDF whenever a button is clicked.

SSR2.5.2
Stamp with username, computer ID, datatime and PDF document filepath.

SSR2.5.3
Title as [entry name] & "_" & [PARTNUMBER_YYYYMMDDHHMMSS] & ".pdf".

SSR2.5.4
Save catalog PDFs in the **…\TESTbase\Tests\PDFs** folder.

SSR2.5.5
Save report PDFs in the **…\TESTbase\Test Reports\PDFs** folder.

SSR2.5.6
Enable/disable the buttons to show in the PDF which button caused the PDF to be generated.

SSR2.6
Word Template (title)

SSR2.6.1
Save in the **…\TESTbase** folder.

SSR2.6.2
Display four (4) buttons that run VBA macro functions listed in Table 6 – Word Button Macros.

Table 6 – Word Button Macros

Button	Macro
Save	saveButton_Click()
Save and Close	saveAndCloseButton_Click()
Close Without Saving	closeWithoutSavingButton_Click()
Delete	deleteButton_Click()

SSR2.6.3
Automatically save new documents when “Save” or “Save and Close” buttons are clicked according to Table 7 - Document Save Information.

Table 7 - Document Save Information

Document Type	File Details
Test	• Filename: [TEST number] & “.doc” • File location: …\TESTbase\Tests
Report	• Filename: [report date in format YYYYMMDDHHMMSS] & “.doc” • File location: …\TESTbase\Test Reports

SSR2.6.4
Save form contents in excel data file as an entry (row) when “Save” or “Save and Close” buttons are clicked for the first time.

SSR2.6.5
Update excel data file entry (row) with form contents when “Save” or “Save and Close” buttons are clicked after the first click.

SSR2.6.6
Delete excel data file entry (row) and Word document when “Delete” button is clicked.

4. Source Code

This source code is placed in the ThisDocument section of the Word templates.

4.1. Table 8 – VBA Source Code in Engineering Change Order Form (Word) Template

```
'Nolan Manteufel
'2020 - 2022
'VBA SOURCE CODE PASSWORD: 9hvmq6KchrZ+V&&j

'Used to protect data
Private Const MY_PASSWORD = "WT$?@L&x=F5qYfC#"
Private Const DOC_FOLDER = "\Tests\"
Private Const LOGBOOK = "\data\recorddata.xlsx"

'Used to set text
Private Const NEW_ENTRY_KEY As String = " "
Private Const STAMP_PLACEHOLDER_TEXT As String = " "
Private Const INVALID_REQUIRED_TEXT = "Click or tap here to enter text."

'Used to access form Content Controls by Tags
Private Const datetimestamp = "datetimestamp"
Private Const TEST_NAME = "TEST_NAME"
Private Const TEST_NO = "TEST_NO"
Private Const UUT_PN = "UUT_PN"
Private Const LAST_TEST_NO = "LAST_TEST_NO"
Private Const TEST_TYPE = "TEST_TYPE"
Private Const DESCRIPTION = "DESCRIPTION"
Private Const APPROVED_BY_Tag = "APPROVED_BY"
Private Const POWER = "POWER"
Private Const STIMULI = "STIMULI"
Private Const MEASUREMENT = "MEASUREMENT"
Private Const IMAGE1CAPTION = "IMAGE1CAPTION"
Private Const IMAGE2CAPTION = "IMAGE2CAPTION"
Private Const SIGNALPIN = "SIGNALPIN"
Private Const RETURNPIN = "RETURNPIN"
Private Const INSTRUMENT = "INSTRUMENT"
Private Const INSTRUMENTACCURACY = "INSTRUMENTACCURACY"
Private Const PARAMETER = "PARAMETER"
Private Const LL = "LL"
Private Const TV = "TV"
Private Const UL = "UL"
Private Const UNITS = "UNITS"
Private Const ITNEXTSTEP = "ITNEXTSTEP"
Private Const ITADJUST = "ITADJUST"
Private Const ITREPLACE = "ITREPLACE"
Private Const OTHNEXTSTEP = "OTHNEXTSTEP"
Private Const OTHADJUST = "OTHADJUST"
Private Const OTHREPLACE = "OTHREPLACE"
Private Const OTLNEXTSTEP = "OTLNEXTSTEP"
Private Const OTLADJUST = "OTLADJUST"
Private Const OTLREPLACE = "OTLREPLACE"
Private Const OTHUNABLENEXTSTEP = "OTHUNABLENEXTSTEP"
Private Const OTHUNABLEREPLACE = "OTHUNABLEREPLACE"
Private Const OTLUNABLENEXTSTEP = "OTLUNABLENEXTSTEP"
Private Const OTLUNABLEREPLACE = "OTLUNABLEREPLACE"

'Used to access record-stamp Content Controls by Tags
Private Const ccStampComputerID = "ccStampComputerID"
Private Const ccStampUserID = "ccStampUserID"
Private Const ccStampDatetime = "ccStampDatetime"
Private Const ccStampPDF = "ccStampPDF"

'Data key constants
Private Const dataKeyTag = "dataKey" 'for Content Control Tag

'Used to move data
Private Const DATA_ARRAY_LENGTH As Long = 40
Private Const DATAKEY_HEADER = "dataKey"

Dim myDoc As Document
Dim CCs As ContentControls
Dim cc As ContentControl

'Stamp variables
Dim macroUser As String
Dim macroComputer As String
Dim dateTime As String
Dim PDFFILEPATHNAMEEXT As String

'CC variables
Dim testName As String
Dim testNumber As String
Dim uutPN As String
Dim lastTest As String
Dim dataKeyString As String
Dim newEntry As Boolean

Dim rowIndexFinder As Long
Dim rowIndex As Long

Dim dataKeyIndexFinder As Long
Dim dataKeyIndex As Long

Private Sub getClickDetails()

'User
macroUser = Environ$("username")
macroUser = LCase(macroUser)

'Computer
macroComputer = Environ$("computername")
macroComputer = LCase(macroComputer)

'Datetime
Dim timeOnly As String
Dim dateOnly As String
```

```
timeOnly = Format(Time, "hhmmss")
dateOnly = Format(Date, "yyyymmdd")
dateTime = dateOnly & timeOnly

'TEST Name
testName = ActiveDocument.SelectContentControlsByTag(TEST_NAME)(1).Range.Text

'TEST Number
testNumber = ActiveDocument.SelectContentControlsByTag(TEST_NO)(1).Range.Text

'UUT PN
uutPN = ActiveDocument.SelectContentControlsByTag(UUT_PN)(1).Range.Text

'DWG Rev (new)
lastTest = ActiveDocument.SelectContentControlsByTag(LAST_TEST_NO)(1).Range.Text

'Data key
Set cc = ActiveDocument.SelectContentControlsByTag(dataKeyTag)(1)
dataKeyString = cc.Range.Text

'New entry?
If (dataKeyString = NEW_ENTRY_KEY) Then
   newEntry = True
   With cc
   .LockContents = False
   .Range.Text = dateTime
   .LockContents = True
   End With
Else
   newEntry = False
End If
End Sub

Private Sub saveDoc()

'Doc filename
Dim myDocFullPath As String
myDocFullPath = myDoc.AttachedTemplate.Path

'Auto name/save document if it is new
If (myDoc.Path = "") Then
   'Lock the TEST Number content control
   ActiveDocument.SelectContentControlsByTag(TEST_NO)(1).LockContents = True
   'Create path and name for new Word document
   myDocFullPath = myDocFullPath & DOC_FOLDER & testNumber & ".docx"
   'Save new document
   myDoc.SaveAs2 filename:=myDocFullPath
Else
   myDoc.Save
End If
End Sub

Private Sub saveData()
```

```
'Set settings
Application.StatusBar = "Saving data..."

'Variables
Dim dataArray(DATA_ARRAY_LENGTH) As String

Dim rowIndexFinder As Long
Dim rowIndex As Long

'Populate dataArray
Dim ccIndex As Long
ccIndex = 1
For Each cc In CCs
   dataArray(ccIndex) = cc.Range.Text
   ccIndex = ccIndex + 1
Next cc

'Update status bar
Application.StatusBar = "Updating log ..."

Dim myIssueLogFullPath As String
myIssueLogFullPath = myDoc.AttachedTemplate.Path & LOGBOOK

Dim excelApp As Excel.Application
Dim dataWB As Excel.Workbook
Dim dataSheet As Excel.Worksheet
Dim logSheet As Excel.Worksheet

'Launch excel application
Set excelApp = New Excel.Application

'Open data workbook
Set dataWB = excelApp.Workbooks.Open(filename:=myIssueLogFullPath)
Set dataSheet = dataWB.Worksheets("data")

'Unprotect the log
dataSheet.Unprotect (MY_PASSWORD)

'Get dataKey column index
For dataKeyIndexFinder = 1 To dataSheet.UsedRange.Columns.Count
   If (DATAKEY_HEADER = dataSheet.Cells(1, dataKeyIndexFinder).Value) Then
   dataKeyIndex = dataKeyIndexFinder
   dataKeyIndexFinder = dataSheet.UsedRange.Columns.Count
   End If
Next

'Get entry row index
If (newEntry) Then
   rowIndex = dataSheet.UsedRange.Rows.Count + 1
   dataArray(0) = rowIndex
Else
   For rowIndexFinder = 2 To dataSheet.UsedRange.Rows.Count
```

```
    Application.StatusBar = "Searching through records ... " & rowIndexFinder & " of " & dataSheet.UsedRange.Rows.Count
    If (dataKeyString = dataSheet.Cells(rowIndexFinder, dataKeyIndex).Value) Then
      rowIndex = rowIndexFinder
      rowIndexFinder = dataSheet.UsedRange.Rows.Count
    End If
  Next
End If

'Log the entry
dataSheet.Cells(rowIndex, 1).Value = rowIndex
Dim colIndex As Long
For colIndex = 1 To DATA_ARRAY_LENGTH
  Application.StatusBar = "Updating record ... field " & colIndex & " of " & DATA_ARRAY_LENGTH
  dataSheet.Cells(rowIndex, colIndex + 1).Value = dataArray(colIndex)
Next

''Protect the log
dataSheet.Protect (MY_PASSWORD)

'Close log
Word.Application.StatusBar = "Saving ... " & myIssueLogFullPath
dataWB.Close SaveChanges:=True

'Release object references
Set logSheet = Nothing
Set dataSheet = Nothing
Set dataWB = Nothing
Set excelApp = Nothing

'Clear status bar
Application.StatusBar = ""
End Sub

Private Sub deleteData()

'Set settings
Application.StatusBar = "Deleting data..."

'temporaryDoc template
Dim temporaryDocTemplate As String
temporaryDocTemplate = myDoc.AttachedTemplate.FullName

Dim temporaryDoc As Document
Set temporaryDoc = Documents.Add(temporaryDocTemplate)

Dim deleteDoc As String
deleteDoc = myDoc.FullName

Dim rowIndexFinder As Long
Dim rowIndex As Long

Dim myIssueLogFullPath As String
myIssueLogFullPath = myDoc.AttachedTemplate.Path & LOGBOOK
```

```
Dim excelApp As Excel.Application
Dim dataWB As Excel.Workbook
Dim dataSheet As Excel.Worksheet
Dim logSheet As Excel.Worksheet

'Launch excel application
Set excelApp = New Excel.Application

'Open data workbook
Set dataWB = excelApp.Workbooks.Open(filename:=myIssueLogFullPath)
Set dataSheet = dataWB.Worksheets("data")

'Unprotect the log
dataSheet.Unprotect (MY_PASSWORD)

'Get dataKey column index
For dataKeyIndexFinder = 1 To dataSheet.UsedRange.Columns.Count
   If (DATAKEY_HEADER = dataSheet.Cells(1, dataKeyIndexFinder).Value) Then
   dataKeyIndex = dataKeyIndexFinder
   dataKeyIndexFinder = dataSheet.UsedRange.Columns.Count
   End If
Next

'Get entry rowIndex
For rowIndexFinder = 2 To dataSheet.UsedRange.Rows.Count
   Application.StatusBar = "Searching through log ... row " & rowIndexFinder & " of " & dataSheet.UsedRange.Rows.Count
   If (dataKeyString = dataSheet.Cells(rowIndexFinder, dataKeyIndex).Value) Then
      rowIndex = rowIndexFinder
      rowIndexFinder = dataSheet.UsedRange.Rows.Count
   End If
Next

'Delete the row
dataSheet.Rows(rowIndex).Delete

'Renumber the rowIndex column
For rowIndex = 2 To dataSheet.UsedRange.Rows.Count
Application.StatusBar = "Renumbering log ... row " & rowIndex & " of " & dataSheet.UsedRange.Rows.Count
dataSheet.Cells(rowIndex, 1).Value = rowIndex
Next

''Protect the log
dataSheet.Protect (MY_PASSWORD)

'Close log
Application.StatusBar = "Saving ... " & myIssueLogFullPath
dataWB.Close SaveChanges:=True

'Release object references
Set logSheet = Nothing
Set dataSheet = Nothing
Set dataWB = Nothing
Set excelApp = Nothing

'Delete word document
myDoc.Close SaveChanges:=False
Kill deleteDoc

'Close temporary document
If (Word.Application.Documents.Count = 1) Then
   Word.Application.Quit SaveChanges:=wdDoNotSaveChanges
Else
   Application.StatusBar = ""
   temporaryDoc.Close SaveChanges:=False
End If
End Sub

'Sub validReqText()
'MsgBox validRequiredText
'End Sub

Private Function validRequiredText() As Boolean

Dim requiredText(2) As String
requiredText(1) = myDoc.SelectContentControlsByTag(TEST_NO)(1).Range.Text
requiredText(2) = myDoc.SelectContentControlsByTag(UUT_PN)(1).Range.Text

'If (requiredText(1) = INVALID_REQUIRED_TEXT Or requiredText(2) = INVALID_REQUIRED_TEXT) Then
If (requiredText(1) = INVALID_REQUIRED_TEXT) Then
   validRequiredText = False
Else
   validRequiredText = True
End If
End Function

Private Sub unlockStamps()

'Unlock stamp CCs
ActiveDocument.SelectContentControlsByTag(ccStampComputerID)(1).LockContents = False
ActiveDocument.SelectContentControlsByTag(ccStampUserID)(1).LockContents = False
ActiveDocument.SelectContentControlsByTag(ccStampDatetime)(1).LockContents = False
ActiveDocument.SelectContentControlsByTag(ccStampPDF)(1).LockContents = False

End Sub

Private Sub setStamps()

unlockStamps

'Set PDF stamp fields
ActiveDocument.SelectContentControlsByTag(ccStampComputerID)(1).Range.Text = macroComputer
ActiveDocument.SelectContentControlsByTag(ccStampUserID)(1).Range.Text = macroUser
ActiveDocument.SelectContentControlsByTag(ccStampDatetime)(1).Range.Text = dateTime
ActiveDocument.SelectContentControlsByTag(ccStampPDF)(1).Range.Text = PDFFILEPATHNAMEEXT
```

```
End Sub

Private Sub clearStamps()

'Clear PDF stamp fields
ActiveDocument.SelectContentControlsByTag(ccStampComputerID)(1).Range.Text = STAMP_PLACEHOLDER_TEXT
ActiveDocument.SelectContentControlsByTag(ccStampUserID)(1).Range.Text = STAMP_PLACEHOLDER_TEXT
ActiveDocument.SelectContentControlsByTag(ccStampDatetime)(1).Range.Text = STAMP_PLACEHOLDER_TEXT
ActiveDocument.SelectContentControlsByTag(ccStampPDF)(1).Range.Text = STAMP_PLACEHOLDER_TEXT

lockStamps

End Sub

Private Sub lockStamps()

'Lock stamp CCs
ActiveDocument.SelectContentControlsByTag(ccStampComputerID)(1).LockContents = True
ActiveDocument.SelectContentControlsByTag(ccStampUserID)(1).LockContents = True
ActiveDocument.SelectContentControlsByTag(ccStampDatetime)(1).LockContents = True
ActiveDocument.SelectContentControlsByTag(ccStampPDF)(1).LockContents = True

End Sub

Private Sub savePDF()

'Set settings
Application.StatusBar = "Saving PDF..."

'PDF filename
'PDFFILEPATHNAMEEXT = myDoc.Path & "\PDFs\" & testNumber & "_" & dateTime & ".pdf"
PDFFILEPATHNAMEEXT = myDoc.Path & "\PDFs\" & testNumber & ".pdf"

'Update document stamps
setStamps

'Set footer datatime stamp
myDoc.SelectContentControlsByTag(datetimestamp)(1).SetPlaceholderText Text:=dateTime

'Save as PDF
myDoc.ExportAsFixedFormat _
    OutputFileName:=PDFFILEPATHNAMEEXT, _
    ExportFormat:=wdExportFormatPDF, _
    OpenAfterExport:=False

'Clear footer datatime stamp
myDoc.SelectContentControlsByTag(datetimestamp)(1).SetPlaceholderText Text:=STAMP_PLACEHOLDER_TEXT

'Update document stamps
clearStamps

Application.StatusBar = ""
End Sub
```

```
Private Sub saveButton_Click()

'Set settings
Application.ScreenUpdating = False

'Set objects
Set myDoc = ActiveDocument
Set CCs = myDoc.ContentControls

'Are required fields valid?
If validRequiredText Then
   'Set button enables
   saveButton.Enabled = True
   saveAndCloseButton.Enabled = False
   closeWithoutSavingButton.Enabled = False
   deleteButton.Enabled = False
   'Run subroutines
   getClickDetails
   saveDoc
   savePDF
   saveData
   'Reset button enables
   saveButton.Enabled = True
   saveAndCloseButton.Enabled = True
   closeWithoutSavingButton.Enabled = True
   deleteButton.Enabled = True
   'Save button changes
   ActiveDocument.Save
   'Message user
   MsgBox ("Congratulations, your document has been saved successfully.")
Else
   'Message user
   MsgBox ("A TEST NUMBER is required.")
End If

'Reset settings
Application.ScreenUpdating = True
End Sub

Private Sub saveAndCloseButton_Click()

'Set settings
Application.ScreenUpdating = False

'Set objects
Set myDoc = ActiveDocument
Set CCs = myDoc.ContentControls

If validRequiredText Then
   'Set button enables
   saveButton.Enabled = False
   saveAndCloseButton.Enabled = True
```

```
    closeWithoutSavingButton.Enabled = False
    deleteButton.Enabled = False
    'Run subroutines
    getClickDetails
    saveDoc
    savePDF
    saveData
    'Reset button enables
    saveButton.Enabled = True
    saveAndCloseButton.Enabled = True
    closeWithoutSavingButton.Enabled = True
    deleteButton.Enabled = True
    'Close document
    If (Application.Documents.Count = 1) Then
        ActiveDocument.Save
        Application.Quit
    Else
        ActiveDocument.Close SaveChanges:=True
    End If
Else
    MsgBox ("A TEST NUMBER is required.")
End If
'Reset settings
Application.ScreenUpdating = True
End Sub

Private Sub closeWithoutSavingButton_Click()

'Set settings
Application.ScreenUpdating = False

'Set objects
Set myDoc = ActiveDocument
Set CCs = myDoc.ContentControls

If validRequiredText Then
    'Set button enables
    saveButton.Enabled = False
    saveAndCloseButton.Enabled = False
    closeWithoutSavingButton.Enabled = True
    deleteButton.Enabled = False
    'Run subroutines
    getClickDetails
    savePDF
    If (Application.Documents.Count = 1) Then
        Application.Quit SaveChanges:=wdDoNotSaveChanges
    Else
        ActiveDocument.Close SaveChanges:=False
    End If
Else
    MsgBox ("A TEST NUMBER is required.")
End If
```

```
'Reset settings
Application.ScreenUpdating = True
End Sub

Private Sub deleteButton_Click()

'Set settings
Application.ScreenUpdating = False

'Set objects
Set myDoc = ActiveDocument
Set CCs = myDoc.ContentControls

If validRequiredText Then
   'Set button enables
   saveButton.Enabled = False
   saveAndCloseButton.Enabled = False
   closeWithoutSavingButton.Enabled = False
   deleteButton.Enabled = True
   'Run subroutines
   getClickDetails
   savePDF
   deleteData
Else
   MsgBox ("A TEST NUMBER is required.")
End If
End Sub
```

4.2. Table 9 – VBA Source Code in Report Form (Word) Template

4.3. Table 10 – VBA Source Code in the Analysis (Excel) Workbook

```
'Nolan Manteufel
'August 2022
'PASSWORD: 9hvmq6KchrZ+V&&j

Dim myWB As Workbook
Dim myWS As Worksheet
Dim dataWB As Workbook
Dim dataWS As Worksheet

Dim dataWorkbookFullPath As String

Dim usedRowCount As Integer
Dim usedColCount As Integer

Private Const DATASHEET_PASSWORD = "WT$?@L&x=F5qYfC#"
Private Const RECORD_DATABOOK = "\data\recorddata.xlsx"
Private Const REPORTS_DATABOOK = "\data\reportdata.xlsx"
Private Const HOMESHEET = "overview"

Private Function copySheets(originSheet As Worksheet, destinationSheet As Worksheet) As Boolean

'Clear desitination sheet
destinationSheet.Cells.Clear

'Find the data
usedRowCount = originSheet.UsedRange.Rows.Count
usedColCount = originSheet.UsedRange.Columns.Count

'Copy data
With originSheet
.Range(.Cells(1, 1), .Cells(usedRowCount, usedColCount)).Copy
End With

'Paste data
destinationSheet.Range("A1").PasteSpecial Paste:=xlPasteValues

End Function

Private Sub refreshData_Click()
'Save application settings
Dim applicationState(2) As Boolean
applicationState(0) = Application.ScreenUpdating
applicationState(1) = Application.EnableEvents
applicationState(2) = Application.DisplayAlerts

'Set application settings
Application.ScreenUpdating = False
Application.EnableEvents = False
Application.DisplayAlerts = False

'Update data
```

```
' CATALOG DATA
'Set data destination sheet
Set myWB = ActiveWorkbook
Set myWS = myWB.Sheets("orders")

'Set data origin sheet
dataWorkbookFullPath = myWB.Path & RECORD_DATABOOK
Set dataWB = Workbooks.Open(Filename:=dataWorkbookFullPath)
Set dataWS = dataWB.Worksheets("data")
dataWS.Unprotect (DATASHEET_PASSWORD)

'Copy data
Dim result As Boolean
result = copySheets(dataWS, myWS)

'Close catalog data workbook and clear objects
dataWB.Close SaveChanges:=False
Set dataWS = Nothing
Set dataWB = Nothing

' INVENTORY DATA
'Set data destination sheet
Set myWS = myWB.Sheets("reports")

'Set data origin sheet
dataWorkbookFullPath = myWB.Path & REPORTS_DATABOOK
Set dataWB = Workbooks.Open(Filename:=dataWorkbookFullPath)
Set dataWS = dataWB.Worksheets("data")
dataWS.Unprotect (DATASHEET_PASSWORD)

'Copy data
result = copySheets(dataWS, myWS)

'Close inventory data workbook and clear objects
dataWB.Close SaveChanges:=False
Set dataWS = Nothing
Set dataWB = Nothing

'Activate original sheet
myWB.Worksheets(HOMESHEET).Activate

'Clear objects
Set myWS = Nothing
Set myWB = Nothing

'Reset application settings
Application.ScreenUpdating = applicationState(0)
Application.EnableEvents = applicationState(1)
Application.DisplayAlerts = applicationState(2)

'Confirmation message
MsgBox "Congratuations, the data updated successfully."
```

```
End Sub
```