

VBA SOURCE CODE BOOK

PURCHASING MANAGER



Excel For Freelancers

This Automated Purchase Order Finds The Cheapest Vendors



DOWNLOAD APPLICATION



VIEW TRAINING

by: Randy Austin

ABOUT THE AUTHOR

A 4-time Microsoft MVP & lifetime Excel enthusiast, Randy Austin founded Excel For Freelancers in 2017. Excel For Freelancers quickly became the most prominent resource Excel for developers to learn how to turn their passion for Excel into profits by building & selling their own excel-based applications for passive & recurring income.

With over 471,000 YouTube subscribers, 35,448,742 video views, 430+ comprehensive training videos, and a thriving 65,000 member Facebook community, Excel For Freelancers has positioned itself as the #1 Excel developers resource in the world.

Get free content, training, and downloads just by clicking any of the free resources below:



[WEBSITE](#)



[YOUTUBE](#)



[FACEBOOK](#)



[TWITTER](#)



[DISCORD](#)



[INSTAGRAM](#)



[TELEGRAM](#)



[RUMBLE](#)



Microsoft®
Most Valuable
Professional



OUR COURSES & PRODUCTS



This comprehensive program will take you through a 12-phase process that will turn your enthusiasm for Excel into passive income.

[Click here to learn more](#)



16 hour masterclass that will teach you the tips, tricks and techniques on how to create a dynamic single-click dashboard, and a ton more

[Click here to learn more](#)

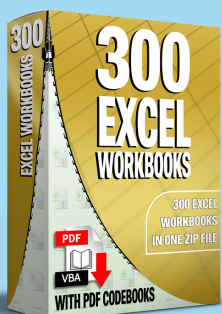
-



This incredible 13-hour freelancer masterclass will teach anyone how to be a successful freelancer with my proven 9-Phase 'Financial Freedom Roadmap' and includes 30+ downloads and exercises.

[Click here to learn more](#)

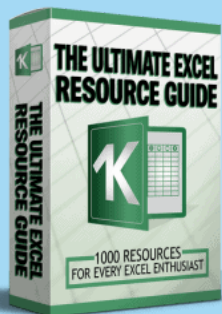
-



Incredible Package of 300 of my BEST Applications with PDF VBA Codebooks packed into a SINGLE ZIP File which also includes the "300 Workbook Library".

[Click here to learn more](#)

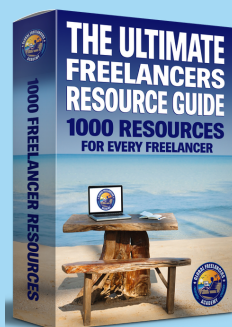
OUR COURSES & PRODUCTS



With 1000 live links, continuously updating content, sort-able and filterable items, you will always have exactly what you need, when you need it.

[Click here to learn more](#)

-



Freelancing essentials from freelance tools to freelance templates, in this Ultimate Freelancer's Resource Guide for freelancers at any stage in their career. With 1,000 Live Links, and a single click-to-update application, you will always have the most current and up-to-date information at your fingertips.

[Click here to learn more](#)

-



Revolutionize the way you work with Excel and take productivity to the next level with the Excel AI Toolpack - the FIRST AI tool designed for ANY Windows Desktop version of Excel. This incredible add-in combines FIVE powerful AI tools, transforming your Excel into an intelligent powerhouse!

[Click here to learn more](#)

Table of Contents

Projects.....	2
VBAProject.....	2
Documents.....	2
Admin.....	2
(Declarations).....	2
Worksheet_Change [Sub].....	2
POItemsDB.....	3
(Declarations).....	3
ProdPricingDB.....	4
(Declarations).....	4
Products.....	5
(Declarations).....	5
Worksheet_Change [Sub].....	5
Worksheet_SelectionChange [Sub].....	5
ProductsDB.....	6
(Declarations).....	6
PurchOrder.....	7
(Declarations).....	7
Worksheet_Change [Sub].....	7
PurchOrderDB.....	8
(Declarations).....	8
ThisWorkbook.....	9
(Declarations).....	9
Modules.....	10
POVendPricing_Macros.....	10
(Declarations).....	10
AddVendorPricing [Sub].....	10
DisplayTop5Vendors [Sub].....	10
ProdPicture_Macros.....	11
(Declarations).....	11
AddProductPicture [Sub].....	11
BrowseForProdPicFolder [Sub].....	11
ClearProductPicture [Sub].....	11
GetPictures [Sub].....	12
GetProductPicture [Sub].....	12
Product_ShowPicture [Sub].....	13
ProdPriceTier_Macros.....	14
(Declarations).....	14
ProdPriceTiers_ListVendors [Sub].....	14
ProdPriceTiers_LoadTiers [Sub].....	14
ProdPriceTiers_SaveTiers [Sub].....	14
Product_Macros.....	16
(Declarations).....	16
Product_AddNew [Sub].....	16
Product_ClearFilter [Sub].....	16
Product_Delete [Sub].....	16
Product_ListProducts [Sub].....	16
Product_Load [Sub].....	16
Product_SavedMsg [Sub].....	17
Product_SaveUpdate [Sub].....	17
Products_GetUniqueCategories [Sub].....	17
Products_GetUniqueSubCategories [Sub].....	18
PurchOrder_Macros.....	19
(Declarations).....	19
PO_AddNew [Sub].....	19
PO_Delete [Sub].....	19
PO_Load [Sub].....	19
PO_Print [Sub].....	20
PO_SaveUpdate [Sub].....	20

```
1 Option Explicit
2
3 Private Sub Worksheet_Change(ByVal Target As Range)
4     'On Custom Field Type Change, Update Format
5     If Not Intersect(Target, Range("E8:E11" )) Is Nothing Then
6         Dim FieldType As String
7         FieldType = Target.Value
8         Select Case FieldType
9             Case Is = "Text"
10            Range("F" & Target.Row).Value = "@"
11            Case Is = "Amount"
12            Range("F" & Target.Row).Value = "$#,##0.00"
13            Case Is = "Percentage"
14            Range("F" & Target.Row).Value = "%"
15            Case Is = "Date"
16            Range("F" & Target.Row).Value = "m/d/yyyy"
17            Case Else
18            Range("F" & Target.Row).Value = "@"
19            End Select
20        End If
21
22        'On Custom Field Type Change, Update Format
23        If Not Intersect(Target, Range("F8:F11" )) Is Nothing Then
24            Dim RowNumb As Long
25            RowNumb = Target.Row
26            Select Case RowNumb
27                Case Is = 8
28                    Products.Range("J11" ).NumberFormat = Target.Value
29                Case Is = 9
30                    Products.Range("M11" ).NumberFormat = Target.Value
31                Case Is = 10
32                    Products.Range("J13" ).NumberFormat = Target.Value
33                Case Is = 11
34                    Products.Range("M13" ).NumberFormat = Target.Value
35            End Select
36        End If
37    End Sub
38
39
40
41
```

1 Option Explicit

2

1	Option Explicit
2	


```
1 Option Explicit
2
3 Private Sub Worksheet_Change(ByVal Target As Range)
4     If Target.CountLarge > 2 Then Exit Sub
5     If Not Intersect(Target, Range("G3" )) Is Nothing Then Product_ListProducts 'List
        Unique Products on Search
6     If Not Intersect(Target, Range("J5" )) Is Nothing And Range("J5" ).Value <> Empty
        Then Products_GetUniqueSubCategories 'Get Unique Sub-Categories
7
8     'On Tier Vendor Change
9     If Not Intersect(Target, Range("M17" )) Is Nothing Then ProdPriceTiers_LoadTiers
10
11     'On Tier Qty Change, but not on Tier load
12     If Not Intersect(Target, Range("N19:N33" )) Is Nothing And Range("B15" ).Value =
        False Then
13         If IsNumeric(Target.Value) = True Then
14             Range("M" & Target.Row + 1).Value = Target.Value + 1 'Increment Next Qty
15         End If
16     End If
17
18 End Sub
19
20 Private Sub Worksheet_SelectionChange(ByVal Target As Range)
21     If Target.CountLarge > 1 Then Exit Sub
22     If Not Intersect(Target, Range("D6:G9999" )) Is Nothing And Range("D" & Target.Row)
        .Value <> Empty Then
23         Range("B2" ).Value = Range("C" & Target.Row).Value
24         Product_Load 'Run Macro to load product
25     End If
26
27     'On Vendors With Price
28     If Not Intersect(Target, Range("I19:J34" )) Is Nothing And Range("I" & Target.Row).
        Value <> Empty Then
29         'Range("B14").Value = Target.Row 'Set Selected Row
30         Range("M17" ).Value = Range("I" & Target.Row).Value 'Set Vendor Name
31     End If
32
33
34
35 End Sub
```

1	Option Explicit
2	

```

1 Option Explicit
2
3 Private Sub Worksheet_Change(ByVal Target As Range)
4     If Target.CountLarge > 1 Then Exit Sub
5     'On Item Change but not on PO Load
6     If Not Intersect(Target, Range("G7:G37" )) Is Nothing And Range("B5" ).Value = False
7         Then
8             Dim ProdRow As Long
9             If Target.Value <> Empty Then 'Selected Items
10                On Error Resume Next
11                ProdRow = ProductsDB.Range("Prod_Name" ).Find(Target.Value, , xlValues,
12                xlWhole).Row
13                On Error GoTo 0
14                If ProdRow = 0 Then Exit Sub
15                Application.ScreenUpdating = False
16                Range("F" & Target.Row).Value = ProductsDB.Range("A" & ProdRow).Value
17                'Product ID
18                Range("H" & Target.Row).Value = ProductsDB.Range("C" & ProdRow).Value 'Prod
19                SKU/UPC
20                Range("I" & Target.Row).Value = ProductsDB.Range("G" & ProdRow).Value
21                'Default Purch Qty
22                Range("J" & Target.Row).Value = ProductsDB.Range("H" & ProdRow).Value 'UOM
23                Range("K" & Target.Row).Value = ProductsDB.Range("I" & ProdRow).Value 'Base
24                Price
25                Range("AA7:AT7" ).Copy
26                Range("AA" & Target.Row & ":AT" & Target.Row).PasteSpecial xlPasteFormulas
27                Application.CutCopyMode = False
28                Application.ScreenUpdating = True
29                Range("G" & Target.Row + 1).Select 'Select Item Below
30            Else
31                Range("F" & Target.Row & ":K" & Target.Row).ClearContents 'Clear Item
32                Details (not DB Row)
33            End If
34        End If
35    End If
36
37    'On PO # Search
38    If Not Intersect(Target, Range("E2" )) Is Nothing And Range("E2" ).Value <> Empty
39        Then
40            Range("B2" ).Value = Range("E2" ).Value 'Set PO ID
41            PO_Load 'Run Macro to Load PO
42        End If
43    End Sub

```

1
2

Option Explicit

1 Option Explicit

2

```

1 Option Explicit
2
3
4 Sub AddVendorPricing()
5     Dim VendNumb As Long, LastProdRow As Long, VendCol As Long
6     With PurchOrder
7         VendNumb = Replace(Application.Caller, "AddVend" , "" )
8         If VendNumb = 0 Then Exit Sub
9         VendCol = 14 + ((VendNumb - 1) * 2)
10        LastProdRow = .Range("G35" ).End(xlUp).Row
11        If LastProdRow = 6 Then Exit Sub 'Exit on No Product
12        .Range("K7:K" & LastProdRow).Value = Range(.Cells(7, VendCol), .Cells(
13            LastProdRow, VendCol)).Value 'Copy Over vendor Prices
14        .Range("H2" ).Value = .Cells(4, VendCol).Value 'Set Vendor Name
15    End With
16 End Sub
17
18 Sub DisplayTop5Vendors()
19     Dim LastRow As Long, LastCol As Long, LastVendRow, VendRow As Long, VendCol As Long
20     Dim PriceRng As Range
21     With PurchOrder
22         .Range("N3:W4,N8:W37,BA2:CI50" ).ClearContents 'Clear Previous Data
23         LastRow = .Range("G37" ).End(xlUp).Row
24         If LastRow = 6 Then
25             MsgBox "Please make sure to add at least one product before displaying top 5
26                 vendors"
27             Exit Sub
28         End If
29         Application.ScreenUpdating = False
30         LastVendRow = VendorDB.Range("A9999" ).End(xlUp).Row
31         .Range("BA2:BB" & LastVendRow - 2).Value = VendorDB.Range("A4:B" & LastVendRow)
32         .Value 'Bring Over Vendors & Vendor ID
33         LastCol = LastVendRow + 23
34         Set PriceRng = .Range(.Cells(5, 27), .Cells(LastRow, LastCol))
35         PriceRng.Copy
36         .Range("BC2" ).PasteSpecial xlPasteValues, , , Transpose:=True
37         With .Sort
38             .SortFields.Clear
39             .SortFields.Add Key:=PurchOrder.Range("BC2" ), SortOn:=xlSortOnValues, Order:=
40                 xlDescending, DataOption:=xlSortNormal 'Sort By Most Matching Products
41             .SortFields.Add Key:=PurchOrder.Range("BD2" ), SortOn:=xlSortOnValues, Order:=
42                 xlAscending, DataOption:=xlSortNormal 'Sort By Lowest Cost
43             .SetRange Range(PurchOrder.Cells(2, 53), PurchOrder.Cells(LastVendRow + 2,
44                 LastRow + 50)) 'Set Range
45             .Apply 'Apply Sort
46         End With
47
48         'Add Top 5 Vendors
49         For VendRow = 2 To 6
50             VendCol = 14 + ((VendRow - 2) * 2)
51             .Cells(3, VendCol).Value = .Range("BA" & VendRow).Value 'Vendor ID
52             .Cells(4, VendCol).Value = .Range("BB" & VendRow).Value 'Vendor Name
53         Next VendRow
54         .Range("N1:W1" ).Copy 'Copy Formulas
55         .Range("N7:W" & LastRow).PasteSpecial xlPasteFormulas
56         Application.CutCopyMode = False
57         .Range("H2" ).Select
58         Application.ScreenUpdating = True
59     End With
60 End Sub

```

```

1 Option Explicit
2 Dim PicPath As String, PicFolder As String
3 #If VBA7 And Win64 Then
4 Private Declare PtrSafe Function URLDownloadToFile Lib "urlmon" _
5 Alias "URLDownloadToFileA" (ByVal pCaller As Long, _
6 ByVal szURL As String, ByVal szFileName As String, _
7 ByVal dwReserved As LongPtr, ByVal lpfnCB As Long) As Long
8 #Else
9 Private Declare Function URLDownloadToFile Lib "urlmon" _
10 Alias "URLDownloadToFileA" ( _
11 ByVal pCaller As Long, _
12 ByVal szURL As String, _
13 ByVal szFileName As String, _
14 ByVal dwReserved As Long, _
15 ByVal lpfnCB As Long _
16 ) As Long
17 #End If
18
19 Sub AddProductPicture()
20 Dim ProdPic As FileDialog
21 Set ProdPic = Application.FileDialog(msoFileDialogFilePicker)
22 PicFolder = [ProductPicFolder]
23 If PicFolder = "" Or Dir(PicFolder, vbDirectory) = "" Then
24 BrowseForProdPicFolder
25 PicFolder = [ProductPicFolder]
26 If PicFolder = "" Or Dir(PicFolder, vbDirectory) = "" Then Exit Sub
27 End If
28 With ProdPic
29 .Title = "Select Item Picture"
30 .Filters.Add "Select Item Picture", "*.jpg, *.png, *.jpeg, *.gif", 1
31 If .Show <> -1 Then GoTo NoSelection
32 PicPath = PicFolder & "\" & Dir(.SelectedItems(1))
33 If .SelectedItems(1) = PicPath Then GoTo SkipCopy
34 If Dir(PicPath, vbDirectory) <> "" Then Kill (PicPath)
35 FileCopy .SelectedItems(1), PicPath 'Copy to new location
36 SkipCopy:
37 End With
38 Products.Range("B9").Value = Dir(PicPath) 'Set Picture Name
39 Product_ShowPicture
40 Product_SaveUpdate
41 NoSelection:
42 End Sub
43
44
45 Sub BrowseForProdPicFolder()
46 Dim ProdPicFolder As FileDialog
47 Set ProdPicFolder = Application.FileDialog(msoFileDialogFolderPicker)
48 With ProdPicFolder
49 .Title = "Browse For Product Picture Folder"
50 .AllowMultiSelect = False
51 If .Show <> -1 Then GoTo NoSelection
52 Admin.Range("D3").Value = .SelectedItems(1)
53 End With
54 NoSelection:
55 End Sub
56
57
58 Sub ClearProductPicture()
59 Products.Range("B9").ClearContents
60 On Error Resume Next
61 Products.Shapes("ProdPic").Delete

```

1

```

1
62 On Error GoTo 0
63 End Sub
64
65
66 Sub GetPictures()
67 Dim ProdRow As Long
68 With ProdDB
69 For ProdRow = 645 To 694
70 .Activate
71 .Range("B" & ProdRow).Select
72 GetProductPicture
73 Next ProdRow
74 End With
75 End Sub
76
77 Sub GetProductPicture()
78 Dim FileName As String, PicPath As String, PicURL As String, ProdName As String
79 Dim Prompt As String, SecretKey As String, Model As String, Temp As String, httpBody
As String, Response As String
80 Dim ProdRow As Long
81 Dim oXMLHTTP As Object
82 With ProdDB
83 If .Range("H2" ).Value = "" Then
84 MsgBox "Please make sure to select a product in order to generate a picture"
85 Exit Sub
86 End If
87 ProdRow = .Range("H2" ).Value 'Product Row
88 ProdName = .Range("B" & ProdRow).Value 'Product Name
89 If ProdName = "" Then
90 MsgBox "Please make sure to select a product in order to generate a picture"
91 Exit Sub
92 End If
93
94 If [ProductPicFolder] = "" Then
95 MsgBox "Please make sure to set a picture folder in the Admin screen"
96 BrowseForProdPicFolder
97 If [ProductPicFolder] = "" Then Exit Sub
98 End If
99 If Dir([ProductPicFolder], vbDirectory) = "" Then
100 MsgBox "Please make sure to set a picture folder in the Admin screen"
101 BrowseForProdPicFolder
102 If [ProductPicFolder] = "" Or Dir([ProductPicFolder], vbDirectory) = "" Then
103 Exit Sub
104 End If
105
106 SecretKey = Admin.Range("K8" ).Value 'API Key
107 Model = Admin.Range("K9" ).Value 'Model
108 Temp = Admin.Range("K10" ).Value 'Temperature
109 If Model = "" Then Model = "text-davinci-003" 'Set Default Model
110 If Temp = "" Then Temp = "0.2" 'Set Default Temp
111 Prompt = ProdName & " with white background"
112 'Set Prompt to meal name
113 Set oXMLHTTP = CreateObject("MSXML2.ServerXMLHTTP" )
114 oXMLHTTP.Open "POST" , "https://api.openai.com/v1/images/generations" , False
115 'Set Open Posts For Text
116 httpBody = "{" "model" ": " "dall-e-3" ", " "prompt" ": " " " & Prompt & " "}"
117 oXMLHTTP.setRequestHeader "Content-Type" , "application/json" 'Set Header Type
118 oXMLHTTP.setRequestHeader "Authorization" , "Bearer " & SecretKey 'Set Header
Authorization
119 Debug.Print httpBody
120 On Error GoTo TimeOut

```

1 2


```

1 2
119 DoEvents
120 oXMLHTTP.send httpBody
121 On Error GoTo 0
122 Response = oXMLHTTP.responseText 'Extract Response
123 If InStr(Response, "You didn't provide an API key" ) > 0 Then
124     MsgBox "No API Key was added. You can find your API key at " & vbCrLf &
125         "https://platform.openai.com/account/api-keys."
126     Exit Sub
127 End If
128 If InStr(Response, "Incorrect API key provided" ) > 0 Then
129     MsgBox "Incorrect API key provided. You can find your API key at " & vbCrLf &
130         "https://platform.openai.com/account/api-keys."
131     Exit Sub
132 End If
133 If InStr(Response, "maximum context length" ) > 0 Then
134     MsgBox "You have reached the maximum content length. Please clear the history
135         above to move forward"
136     Exit Sub
137 End If
138 Debug.Print Response
139 PicURL = Mid(Response, InStr(Response, "url" ) + 7, InStr(Response, "}" ) - InStr
140     (Response, "url" ) - 13)
141 FileName = ProdName & ".jpg" 'Set Unique File Name
142 PicPath = [ProductPicFolder] & "\" & FileName 'Full File Path
143 If Dir(PicPath, vbDirectory) <> "" Then Kill (PicPath) 'Delete any existing
144     picture under the same name
145 URLDownloadToFile 0, PicURL, PicPath, 0, 0
146 .Range("F" & ProdRow).Value = FileName
147
148 Exit Sub
149
150 Sub Product_ShowPicture()
151     PicFolder = [ProductPicFolder] 'Product Picture Folder (if any)
152     With Products
153         On Error Resume Next
154         Products.Shapes("ProdPic" ).Delete 'Delete any picture if it exists
155         On Error GoTo 0
156         PicPath = PicFolder & "\" & .Range("B9" ).Value 'Picture path
157         On Error Resume Next
158         If Dir(PicPath, vbDirectory) = "" Then Exit Sub
159         On Error GoTo 0
160         .Pictures.Insert(PicPath).Name = "ProdPic"
161         With .Shapes("ProdPic" )
162             .LockAspectRatio = msoCTrue
163             If .Width > .Height Then .Width = 85 Else .Height = 70
164             .Left = Products.Range("O3" ).Left + (Products.Range("O:P" ).Width - .Width) /
165                 2 'Center Picture Horizontally
166             .Top = Products.Range("O3" ).Top + (Products.Range("3:7" ).Height - .Height) /
167                 2 'Center Picture Vertically
168         End With
169     End With
170 End Sub

```

```

1 Option Explicit
2 Dim LastRow As Long, LastResultRow As Long
3
4 Sub ProdPriceTiers_ListVendors()
5     Products.Range("H19:J34").ClearContents 'Clear Previous Vendors
6     With ProdPricingDB
7         .Range("O3:Q999").ClearContents 'Clear Previous Results
8         LastRow = .Range("A99999").End(xlUp).Row
9         If LastRow < 4 Then Exit Sub
10        'Get Unique Vendors & Add Emails
11        .Range("A3:D" & LastRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.Range(
12            "K2:K3"), CopyToRange:=.Range("O2:P2"), Unique:=True
13        LastResultRow = .Range("O99999").End(xlUp).Row
14        If LastResultRow < 3 Then Exit Sub
15        .Range("Q3:Q" & LastResultRow).Formula = .Range("Q1").Formula 'Bring Down
16        Vendor Email Formula
17        Products.Range("H19:J" & LastResultRow + 16).Value = .Range("O3:Q" &
18            LastResultRow).Value 'Bring over vendors
19    End With
20 End Sub
21
22 Sub ProdPriceTiers_LoadTiers()
23     Products.Range("M19:Q34").ClearContents 'Clear Previous Prices
24     With ProdPricingDB
25         LastRow = .Range("A99999").End(xlUp).Row
26         If LastRow < 4 Then Exit Sub
27         'Get Unique Vendors & Add Emails
28         .Range("A3:J" & LastRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.Range(
29            "K2:L3"), CopyToRange:=.Range("S2:X2"), Unique:=True
30         LastResultRow = .Range("S99999").End(xlUp).Row
31         If LastResultRow < 3 Then Exit Sub
32         If LastResultRow < 4 Then GoTo SkipSort
33         'Sort By Tier
34         With .Sort
35             .SortFields.Clear
36             .SortFields.Add Key:=ProdPricingDB.Range("S3"), SortOn:=xlSortOnValues, Order
37                 :=xlAscending, DataOption:=xlSortNormal 'Sort
38             .SetRange ProdPricingDB.Range("S3:X" & LastResultRow) 'Set Range
39             .Apply 'Apply Sort
40         End With
41         SkipSort:
42         Products.Range("M19:Q" & LastResultRow + 16).Value = .Range("T3:X" &
43             LastResultRow).Value 'Bring over Pricing Tier Details
44     End With
45 End Sub
46
47 Sub ProdPriceTiers_SaveTiers()
48     Dim LastTierRow As Long, TierRow As Long, TierDBRow As Long
49     With Products
50         If .Range("B3").Value = Empty Then
51             MsgBox "Please save the product before adding any pricing tiers"
52             Exit Sub
53         End If
54         If .Range("B13").Value = Empty Then
55             MsgBox "Please select a Vendor before saving any pricing tiers"
56             .Range("M17").Select
57             Exit Sub
58         End If
59         LastTierRow = .Range("N36").End(xlUp).Row
60         If LastTierRow < 19 Then Exit Sub 'Exit on no tier data
61         For TierRow = 19 To LastTierRow

```

1 2 3

```
56 | 1 2 3 | If .Range("Q" & TierRow).Value = Empty Then 'New Tier DB Row
57 | | TierDBRow = ProdPricingDB.Range("A99999" ).End(xlUp).Row + 1 'First Avail
58 | | Row
58 | | ProdPricingDB.Range("A" & TierDBRow).Value = .Range("B2" ).Value 'Set
59 | | Product ID
59 | | ProdPricingDB.Range("C" & TierDBRow).Value = .Range("B13" ).Value 'Set
60 | | Vendor ID
60 | | ProdPricingDB.Range("I" & TierDBRow).Value = "=Row()" 'Set DB Row
61 | | .Range("Q" & TierRow).Value = TierDBRow 'Set DB Row
62 | | Else 'existing Tier DB Row
63 | | TierDBRow = .Range("Q" & TierRow).Value
64 | | End If
65 | | ProdPricingDB.Range("B" & TierDBRow).Value = .Range("J3" ).Value 'Product
66 | | Name
66 | | ProdPricingDB.Range("D" & TierDBRow).Value = .Range("M17" ).Value 'Vendor
67 | | Name
67 | | ProdPricingDB.Range("E" & TierDBRow).Value = TierRow - 18 'Set Tier #
68 | | ProdPricingDB.Range("F" & TierDBRow).Value = .Range("M" & TierRow).Value
69 | | 'From #
69 | | ProdPricingDB.Range("G" & TierDBRow).Value = .Range("N" & TierRow).Value
70 | | 'To #
70 | | ProdPricingDB.Range("H" & TierDBRow).Value = .Range("P" & TierRow).Value
71 | | 'Price
71 | | Next TierRow
72 | | End With
73 | | ProdPriceTiers_ListVendors 'Reload Product Price Tiers
74 | End Sub
```

```

1 Option Explicit
2 Dim LastRow As Long, LastResultRow As Long, ProdRow As Long, ProdCol As Long, SelRow As Long
3 Dim ProdFolder As String, PicPath As String
4 Sub Product_AddNew()
5     Products.Range("B2,B14,J3,M3,J5,M5,J7,M7,B9,J9,M9,J11,M11,J13,M13,H19:J34,M19:Q34" )
6     .ClearContents
7     On Error Resume Next
8     Products.Shapes("ProdPic" ).Delete 'Delete any picture if it exists
9     On Error GoTo 0
10    Products.Range("J3" ).Select
11 End Sub
12 Sub Product_ClearFilter()
13    Products.Range("E3,G3" ).ClearContents
14 End Sub
15 Sub Product_Delete()
16    With Products
17        If MsgBox("Are you sure you want to delete this Product?" , vbYesNo, "Delete
18        Product" ) = vbNo Then Exit Sub
19        If .Range("B3" ).Value = Empty Then GoTo NotSaved
20        ProdRow = .Range("B3" ).Value 'Products Row
21        ProductsDB.Range(ProdRow & ":" & ProdRow).EntireRow.Delete 'Delete DB Row
22        NotSaved:
23        Products_GetUniqueCategories 'Run Macro To Update Unique Categories
24        Products_GetUniqueSubCategories 'Run Macro To Update Unique Sub-categories
25        Product_AddNew 'Run macro to create new Products
26        Product_ListProducts 'Update Products list
27    End With
28 End Sub
29
30
31 Sub Product_ListProducts()
32    Products.Range("C6:G9999" ).ClearContents 'Clear Existing Products
33    With ProductsDB
34        LastRow = .Range("A9999" ).End(xlUp).Row
35        If LastRow < 4 Then Exit Sub
36        .Range("A3:F" & LastRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.Range(
37        "Z2:Z3" ), CopyToRange:=.Range("AB2:AF2" ), Unique:=True
38        LastResultRow = .Range("AB9999" ).End(xlUp).Row
39        If LastResultRow < 3 Then Exit Sub
40        If LastResultRow < 4 Then GoTo NoSort 'No sort on only 1 Product
41        'Sort Alphabetally based on product name
42        With .Sort
43            .SortFields.Clear
44            .SortFields.Add Key:=ProductsDB.Range("AC3" ), SortOn:=xlSortOnValues, Order:=
45            xlAscending, DataOption:=xlSortNormal 'Sort
46            .SetRange ProductsDB.Range("AB3:AF" & LastResultRow) 'Set Range
47            .Apply 'Apply Sort
48        End With
49        NoSort:
50        Products.Range("C6:G" & LastResultRow + 3).Value = .Range("AB3:AF" &
51        LastResultRow).Value 'Bring over data
52    End With
53 End Sub
54 Sub Product_Load()
55    With Products
56        .Range("B14,J3,M3,J5,M5,J7,M7,B9,J9,M9,J11,M11,J13,M13,H19:J34,M19:Q34" ).
57        ClearContents
58    End With
59 End Sub

```

```

1 2
55 If .Range("B3" ).Value = Empty Then
56     MsgBox "Please select a correct Product to load"
57     Exit Sub
58 End If
59 ProdRow = .Range("B3" ).Value 'Products Row
60 For ProdCol = 2 To 14
61     .Range(ProductsDB.Cells(1, ProdCol).Value).Value = ProductsDB.Cells(ProdRow,
        ProdCol).Value 'Bring over data
62 Next ProdCol
63 End With
64 Product_ShowPicture
65 ProdPriceTiers_ListVendors 'Refresh Product Financial Items
66 End Sub
67
68 Sub Product_SavedMsg()
69     With Products.Shapes("ProductSavedMsg" )
70         Dim i As Long, Delay As Double, StartTime As Double
71         .Visible = msoCTrue
72         For i = 1 To 75
73             .Fill.Transparency = i / 75
74             Delay = 0.005
75             StartTime = Timer
76             Do
77                 DoEvents
78                 Loop While Timer - StartTime < Delay
79             Next i
80             .Visible = msoFalse
81         End With
82     End Sub
83
84
85
86 Sub Product_SaveUpdate()
87     With Products
88         If .Range("j3" ).Value = Empty Then 'Empty Fields
89             MsgBox "Please make sure to enter a Product Name before saving"
90             Exit Sub
91         End If
92         If .Range("B3" ).Value = Empty Then 'New Product
93             ProdRow = ProductsDB.Range("A99999" ).End(xlUp).Row + 1 'First Avail
            Row
94             .Range("B2" ).Value = .Range("B4" ).Value 'New Product ID
95             ProductsDB.Range("A" & ProdRow).Value = .Range("B2" ).Value 'Set Products ID
96         Else 'Existing Products
97             ProdRow = .Range("B3" ).Value 'Existing Products Row
98         End If
99         For ProdCol = 2 To 14
100             ProductsDB.Cells(ProdRow, ProdCol).Value = .Range(ProductsDB.Cells(1, ProdCol)
            .Value).Value 'Save/Update Data
101         Next ProdCol
102         Product_ListProducts 'Update Products list
103         Products_GetUniqueCategories 'Run Macro To List Unique Categories
104         Products_GetUniqueSubCategories 'Run Macro To List Unique Sub-categories
105         Product_SavedMsg
106     End With
107 End Sub
108
109 Sub Products_GetUniqueCategories()
110     With ProductsDB
111         LastRow = .Range("A99999" ).End(xlUp).Row

```

```
1 2
112 If LastRow < 4 Then Exit Sub
113 On Error Resume Next
114 .Names("Criteria" ).Delete
115 On Error GoTo 0
116 .Range("D3:D" & LastRow).AdvancedFilter xlFilterCopy, , CopyToRange:=.Range(
    "T2" ), Unique:=True
117 End With
118 End Sub
119
120 Sub Products_GetUniqueSubCategories()
121 With ProductsDB
122 LastRow = .Range("A99999" ).End(xlUp).Row
123 If LastRow < 4 Then Exit Sub
124 .Range("D3:E" & LastRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.Range(
    "V2:V3" ), CopyToRange:=.Range("X2" ), Unique:=True
125 End With
126 End Sub
```

```

1 Option Explicit
2 Dim PORow As Long, POCol As Long, POItemRow As Long, LastItemRow As Long, ItemRow As
Long
3 Dim LastRow As Long, ResultRow As Long, LastPOItemRow As Long, LastItemResultRow As
Long, ItemDBRow As Long
4
5
6 Sub PO_AddNew()
7     With PurchOrder
8         .Range("E2:E5,H2:I2,F7:K37,M7:M37,N3:W4,N7:W37" ).ClearContents 'Clear PO Fields
9
10        .Range("AA8:AT37" ).ClearContents 'Clear Unused Formulas
11        .Range("B2" ).Value = .Range("B4" ).Value 'Next PO #
12        .Range("E3" ).Value = Date 'PO Time
13        .Range("E4" ).Value = Admin.Range("C19" ).Value 'Set Initial PO Status
14    End With
15 End Sub
16
17 Sub PO_Delete()
18     With PurchOrder
19         If MsgBox("Are you sure you want to delete this Purchase Order?" , vbYesNo,
20             "Delete Purchase Order" ) = vbNo Then Exit Sub
21         If .Range("B3" ).Value = Empty Then GoTo NotSaved
22         PORow = .Range("B3" ).Value 'Set Purchase Order Row
23         PurchOrderDB.Range(PORow & ":" & PORow).EntireRow.Delete
24
25         'Delete PO Items
26         With POItemsDB
27             LastRow = .Range("A99999" ).End(xlUp).Row
28             If LastRow < 4 Then Exit Sub
29             .Range("A3:J" & LastRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.Range(
30                 "O2:O3" ), CopyToRange:=POItemsDB.Range("T2:AA2" ), Unique:=False
31             LastItemResultRow = POItemsDB.Range("T99999" ).End(xlUp).Row 'Last Results
32             Row
33             If LastItemResultRow < 3 Then GoTo NoItems
34
35             For ResultRow = 3 To LastItemResultRow
36                 ItemDBRow = .Range("Z" & ResultRow).Value 'Set Purchase Order Database
37                 Row
38                 .Range("A" & ItemDBRow & ":J" & ItemDBRow).ClearContents 'Clear Fields
39                 (deleting creates issues with results
40             Next ResultRow
41             'Resort DB to remove spaces
42             With .Sort
43                 .SortFields.Clear
44                 .SortFields.Add Key:=POItemsDB.Range("A4" ), SortOn:=xlSortOnValues, Order:=
45                 =xlAscending, DataOption:=xlSortNormal 'Sort
46                 .SetRange POItemsDB.Range("A4:J" & LastRow) 'Set Range
47                 .Apply 'Apply Sort
48             End With
49         End With
50         NoItems:
51         NotSaved:
52         PO_AddNew 'Add New Purchase Order
53     End With
54 End Sub
55
56 Sub PO_Load()
57     With PurchOrder
58         If .Range("B3" ).Value = Empty Then
59
60         End If
61     End With
62 End Sub

```

1 2 3

```

1 2 3
53     MsgBox "Please enter a correct PO #"
54     Exit Sub
55     End If
56     Application.ScreenUpdating = False
57     PORow = .Range("B3" ).Value 'PO Row
58     .Range("E2:E5,H2:I2,F7:K37,M7:M37,N3:W4,N7:W37" ).ClearContents 'Clear PO Fields

59     .Range("B5" ).Value = True 'Set PO Load To True
60     For POCOL = 2 To 5
61         .Range(PurchOrderDB.Cells(1, POCOL).Value).Value = PurchOrderDB.Cells(PORow,
62             POCOL).Value 'Load PO Details
63     Next POCOL
64     End With
65     With POItemsDB
66         'Load In PO Items
67         LastPOItemRow = .Range("A99999" ).End(xlUp).Row 'Last Item Row
68         If LastPOItemRow < 3 Then GoTo NoItems
69         .Range("A3:J" & LastPOItemRow).AdvancedFilter xlFilterCopy, CriteriaRange:=.
70             Range("O2:O3" ), CopyToRange:=POItemsDB.Range("T2:AA2" ), Unique:=False
71         LastItemResultRow = POItemsDB.Range("T99999" ).End(xlUp).Row 'Last Results Row
72         If LastItemResultRow < 3 Then GoTo NoItems
73         For ResultRow = 3 To LastItemResultRow
74             POItemRow = POItemsDB.Range("AA" & ResultRow).Value 'Purchase Order Item Row
75             PurchOrder.Range("F" & POItemRow & ":K" & POItemRow).Value = .Range("T" &
76                 ResultRow & ":Y" & ResultRow).Value 'Item Name, Desc, Qty, Price
77             PurchOrder.Range("M" & POItemRow).Value = .Range("Z" & ResultRow).Value
78             'Item Database Row
79         Next ResultRow
80         'Update Pricing Formulas
81         If LastItemResultRow = 3 Then GoTo SingleItem 'Don't copy formula on single item
82         PurchOrder.Range("AA7:AT7" ).Copy
83         PurchOrder.Range("AA8:AT" & LastItemResultRow + 4).PasteSpecial xlPasteFormulas
84         Application.CutCopyMode = False
85         SingleItem:
86         NoItems:
87         PurchOrder.Range("B5" ).Value = False 'Set PO Load To False
88         PurchOrder.Range("E2" ).Select
89         Application.ScreenUpdating = True
90     End With
91     End Sub
92
93 Sub PO_Print()
94     PurchOrder.PrintOut , , , True, , , False
95 End Sub
96
97 Sub PO_SaveUpdate()
98     With PurchOrder
99         'Determine if New or Existing PO
100        If .Range("B3" ).Value = "" Then 'New PO
101            PORow = PurchOrderDB.Range("A99999" ).End(xlUp).Row + 1 'First Avail. Row
102            .Range("B2" ).Value = .Range("B4" ).Value 'Next PO #
103            PurchOrderDB.Range("A" & PORow).Value = .Range("B2" ).Value 'PO #
104        Else ' Existing PO
105            PORow = .Range("B3" ).Value 'Purchase Order Row
106

```



```
107     End If
108     For POCol = 2 To 6
109         PurchOrderDB.Cells(PORow, POCol).Value = .Range(PurchOrderDB.Cells(1, POCol).
            Value).Value 'Save PO Details
110     Next POCol
111     'Add/Update PO Item
112     LastItemRow = .Range("G37").End(xlUp).Row 'Last Item Row
113     If LastItemRow < 7 Then GoTo NoItems
114     For ItemRow = 7 To LastItemRow
115         If .Range("M" & ItemRow).Value <> Empty Then 'Existing Row
116             POItemRow = .Range("M" & ItemRow).Value 'Existing PO Item Row
117         Else 'New Row
118             POItemRow = POItemsDB.Range("A9999").End(xlUp).Row + 1 'First Avail Item
            Row
119             POItemsDB.Range("A" & POItemRow).Value = .Range("B2").Value 'PO Number
120             POItemsDB.Range("I" & POItemRow).Value = ItemRow 'PO Row
121             POItemsDB.Range("J" & POItemRow).Value = "=Row()" 'Database Row
122             .Range("M" & ItemRow).Value = POItemRow
123         End If
124         POItemsDB.Range("B" & POItemRow & ":H" & POItemRow).Value = .Range("F" &
            ItemRow & ":L" & ItemRow).Value 'Item, SKU, Qty, UOM, Price, Total
125     Next ItemRow
126     NoItems:
127 End With
128 End Sub
```

- , 11
- A**
 - Activate, 12
 - Add, 10, 11, 14, 16, 19
 - AddProductPicture, 11
 - AddVendorPricing, 10
 - Admin, 11, 12, 19
 - AdvancedFilter, 14, 16, 18-20
 - AllowMultiSelect, 11
 - Application, 7, 10, 11, 20
 - Apply, 10, 14, 16, 19
- B**
 - BrowseForProdPicFolder, 11, 12
- C**
 - Caller, 10
 - Cells, 10, 17, 20, 21
 - Clear, 10, 14, 16, 19
 - ClearContents, 7, 10, 11, 14, 16, 19, 20
 - ClearProductPicture, 11
 - Copy, 7, 10, 20
 - CopyToRange, 14, 16, 18-20
 - CountLarge, 5, 7
 - CreateObject, 12
 - CriteriaRange, 14, 16, 18-20
 - CutCopyMode, 7, 10, 20
- D**
 - DataOption, 10, 14, 16, 19
 - Debug, 12, 13
 - Delay, 17
 - Delete, 11, 13, 16, 18, 19
 - Dir, 11-13
 - DisplayTop5Vendors, 10
 - DoEvents, 13, 17
 - dwReserved, 11
- E**
 - Empty, 5, 7, 14-17, 19, 21
 - EntireRow, 16, 19
 - Explicit, 2-11, 14, 16, 19
- F**
 - FieldType, 2
 - FileCopy, 11
 - FileDialog, 11
 - FileName, 12, 13
 - Fill, 17
 - Filters, 11
 - Find, 7
 - Formula, 14
- G**
 - GetPictures, 12
 - GetProductPicture, 12
- H**
 - Height, 13
 - httpBody, 12, 13
- I**
 - i, 17
 - Insert, 13
 - InStr, 13
 - Intersect, 2, 5, 7
 - IsNumeric, 5
- ItemDBRow, 19
- ItemRow, 19, 21
- K**
 - Key, 10, 14, 16, 19
 - Kill, 11, 13
- L**
 - LastCol, 10
 - LastItemResultRow, 19, 20
 - LastItemRow, 19, 21
 - LastPOItemRow, 19, 20
 - LastProdRow, 10
 - LastResultRow, 14, 16
 - LastRow, 10, 14, 16-19
 - LastTierRow, 14
 - LastVendRow, 10
 - Left, 13
 - LockAspectRatio, 13
 - LongPtr, 11
 - lpfnCB, 11
- M**
 - Mid, 13
 - Model, 12
 - MsgBox, 10, 12-14, 16, 17, 19, 20
 - msoCTrue, 13, 17
 - msoFalse, 17
 - msoFileDialogFilePicker, 11
 - msoFileDialogFolderPicker, 11
- N**
 - Name, 13
 - Names, 18
 - NoItems, 19-21
 - NoSelection, 11
 - NoSort, 16
 - NotSaved, 16, 19
 - NumberFormat, 2
- O**
 - Open, 12
 - Order, 10, 14, 16, 19
 - oXMLHTTP, 12, 13
- P**
 - PasteSpecial, 7, 10, 20
 - pCaller, 11
 - PicFolder, 11, 13
 - PicPath, 11-13, 16
 - Pictures, 13
 - PicURL, 12, 13
 - PO_AddNew, 19
 - PO_Delete, 19
 - PO_Load, 7, 19
 - PO_Print, 20
 - PO_SaveUpdate, 20
 - POCol, 19-21
 - POItemRow, 19-21
 - POItemsDB, 19-21
 - PORow, 19-21
 - PriceRng, 10
 - Print, 12, 13
 - PrintOut, 20
 - ProdCol, 16, 17
 - ProdDB, 12
 - ProdFolder, 16
 - ProdName, 12, 13
 - ProdPic, 11
 - ProdPicFolder, 11
 - ProdPriceTiers_ListVendors, 14, 15, 17
 - ProdPriceTiers_LoadTiers, 5, 14
 - ProdPriceTiers_SaveTiers, 14
 - ProdPricingDB, 14, 15
 - ProdRow, 7, 12, 13, 16, 17
 - Product_AddNew, 16
 - Product_ClearFilter, 16
 - Product_Delete, 16
 - Product_ListProducts, 5, 16, 17
 - Product_Load, 5, 16
 - Product_SavedMsg, 17
 - Product_SaveUpdate, 11, 17
 - Product_ShowPicture, 11, 13, 17
 - ProductPicFolder, 11-13
 - Products, 2, 11, 13, 14, 16, 17
 - Products_GetUniqueCategories, 16, 17
 - Products_GetUniqueSubCategories, 5, 16-18
 - ProductsDB, 7, 16-18
 - Prompt, 12
 - PtrSafe, 11
 - PurchOrder, 10, 19, 20
 - PurchOrderDB, 19-21
- R**
 - Range, 2, 5, 7, 10-21
 - Replace, 10
 - Response, 12, 13
 - responseText, 13
 - ResultRow, 19, 20
 - Row, 2, 5, 7, 10, 14-21
 - RowNumb, 2
- S**
 - ScreenUpdating, 7, 10, 20
 - SecretKey, 12
 - SelectedItems, 11
 - SelRow, 16
 - send, 13
 - SetRange, 10, 14, 16, 19
 - setRequestHeader, 12
 - Shapes, 11, 13, 16, 17
 - Show, 11
 - SingleItem, 20
 - SkipCopy, 11
 - SkipSort, 14
 - Sort, 10, 14, 16, 19
 - SortFields, 10, 14, 16, 19
 - SortOn, 10, 14, 16, 19
 - StartTime, 17
 - szFileName, 11
 - szURL, 11
- T**
 - Target, 2, 5, 7
 - Temp, 12
 - TierDBRow, 14, 15
 - TierRow, 14, 15
 - Timeout, 12, 13
 - Timer, 17
 - Title, 11
 - Top, 13
 - Transparency, 17
 - Transpose, 10
- U**
 - Unique, 14, 16, 18-20
 - URLDownloadToFile, 11, 13
- V**
 - Value, 2, 5, 7, 10-17, 19-21

VBA7, 11
vbCrLf, 13
vbDirectory, 11-13
vbNo, 16, 19
vbYesNo, 16, 19
VendCol, 10
VendNumb, 10
VendorDB, 10
VendRow, 10
Visible, 17

W

Width, 13
Win64, 11
Worksheet_Change, 2, 5, 7
Worksheet_SelectionChange, 5

X

xlAscending, 10, 14, 16, 19
xlDescending, 10
xlFilterCopy, 14, 16, 18-20
xlPasteFormulas, 7, 10, 20
xlPasteValues, 10
xlSortNormal, 10, 14, 16, 19
xlSortOnValues, 10, 14, 16, 19
xlUp, 10, 14-21
xlValues, 7
xlWhole, 7

Thank You!

This source code was created and made available to help you gain a better understanding of how VBA is used to create amazing Excel-based applications.

Thank you so much for your continued shares, likes and support. It really helps.



Excel For Freelancers