香港城市大學
City University of Hong Kong

MS3111 Quantitative Business Analysis with Visual Basic for Applications

FINAL REPORT

Automated Monthly, Genre-wise and Branch-wise Reports of a Cinema Company (with Add/Delete Record Feature)

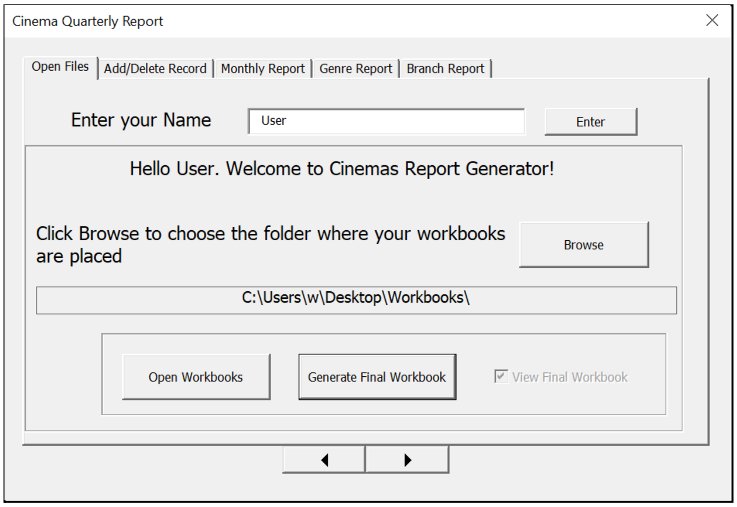
Group 27

Student Name	**SID**
Nazym NUSSIPKALIYEVA	55859108
Abhimanyu BHATI	55991421

Table of Contents

1. Introduction

Our project aims to provide monthly, genre-wise and branch-wise reports of a Cinema for a given quarter. For the purpose of this project, we have extracted data from four workbooks containing relevant data sets: *Movies.xlsx, Box Office.xlsx, Production Companies.xlsx, and Cinema Info.xlsx*. Using these data sets, we derive hypothetical business reports using Excel workbooks and automate the reporting process using VBA. A UserForm 'UserFormCinemaApp' is created to showcase the running application.



Cinema Quarterly Report

Enter your Name: User

Hello User. Welcome to Cinemas Report Generator!

Click Browse to choose the folder where your workbooks are placed

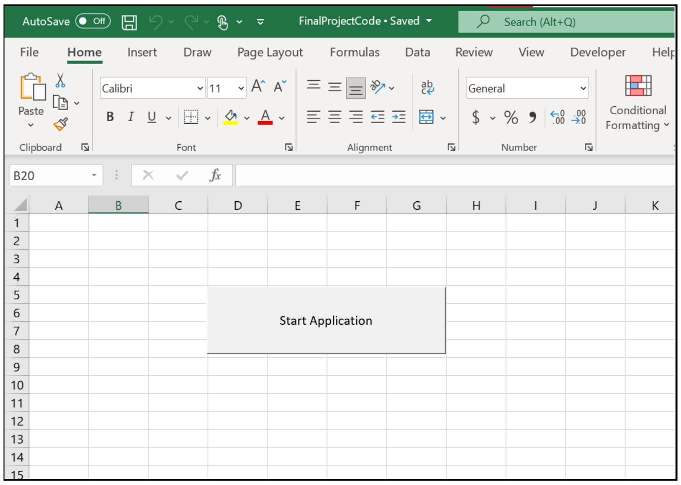
C:\Users\w\Desktop\Workbooks\

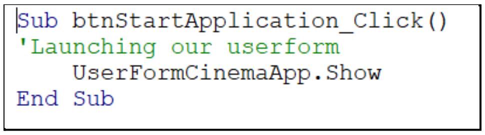
The first page of the UserForm allows the user to enter their name and also browse for the "Workbooks" folder from the computer. This folder must contain the four data sets mentioned above. Then, the user opens these workbooks and generates the final workbook by clicking on the corresponding CommandButton Controls. The final workbook is a derived data set using relevant columns and data from the four workbooks mentioned. After the workbooks have been opened and the final workbook has been generated, the other Pages of the MultiPage Control will be enabled, where the user can add/delete records, and generate the revenue report charts and statistics by month, genre, and branch. Users can select specific charts and statistics or generate them all by clicking on the corresponding OptionButton and CheckBox Controls. These reports are added to the Report Workbook which contains at most 3 sheets, one for each type of report (monthly, genre, branch). The user can add/delete records at runtime and can refresh these reports to see changes in the statistics, charts etc. due to the record manipulation.

The next sections will elaborate more on the details of the coding and data manipulation.

2. UserForm Controls

The application can be started by clicking the Start Application button on Sheet1 of the FinalProjectCode.xlsm file.

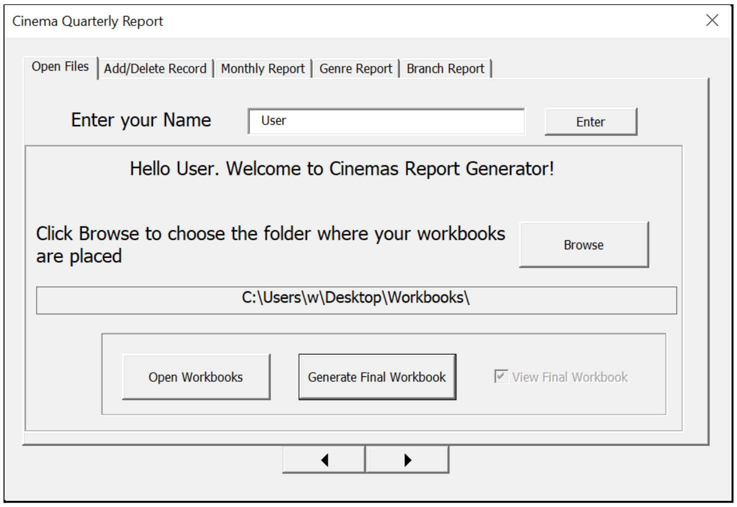




```
Sub btnStartApplication_Click()
'Launching our userform
    UserFormCinemaApp.Show
End Sub
```

2.1 UserForm Controls

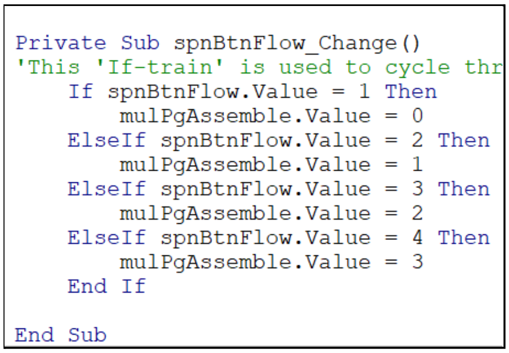
The Userform "UserFormCinemaApp" with title "Cinema Quarterly Report" (screenshot is shown below) contains the following Controls: MultiPage Control 'mulPgAssemble' with five Pages and a SpinButton 'spnBtnFlow', where each Page of the MultiPage Control consists of its own set of Controls. The SpinButton is used to navigate through the different pages of the MultiPag Control.



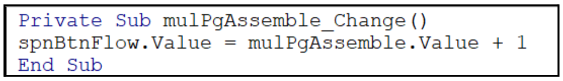
Hello User. Welcome to Cinemas Report Generator!

Click Browse to choose the folder where your workbooks are placed

C:\Users\w\Desktop\Workbooks\



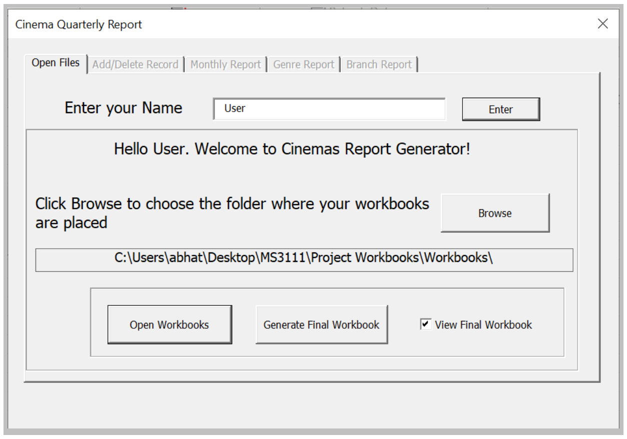
```
Private Sub spnBtnFlow_Change()
'This 'If-train' is used to cycle thr
    If spnBtnFlow.Value = 1 Then
        mulPgAssemble.Value = 0
    ElseIf spnBtnFlow.Value = 2 Then
        mulPgAssemble.Value = 1
    ElseIf spnBtnFlow.Value = 3 Then
        mulPgAssemble.Value = 2
    ElseIf spnBtnFlow.Value = 4 Then
        mulPgAssemble.Value = 3
    End If

End Sub
```



```
Private Sub mulPgAssemble_Change()
spnBtnFlow.Value = mulPgAssemble.Value + 1
End Sub
```

The screenshot above-right shows the change event of the SpinButton Control "spnBtnFlow". When the user clicks the SpinButton the Change() event of the control is invoked in which we change the MultiPage Control's value to the next or the previous one depending on the case. In case the user manually clicks one of the MultiPage Control tabs, then we handle this event by updating the SpinButton value so that these two controls remain consistent with each other and both could help the user navigate through the pages.

2.2 Page “mulPgOpenFiles” Controls



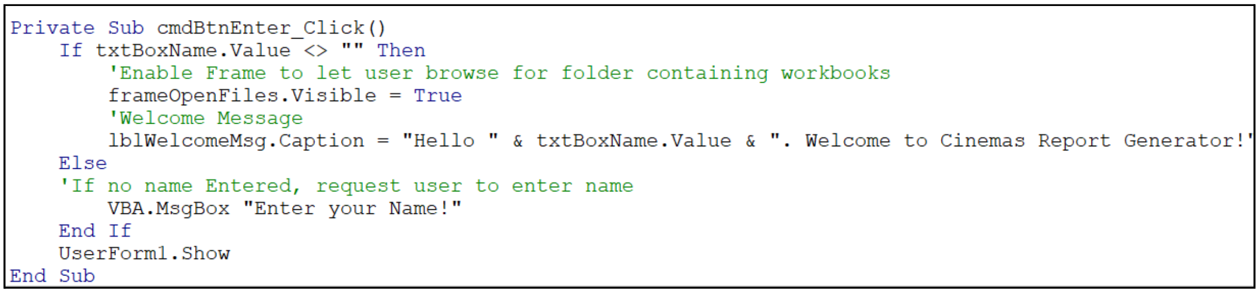
User

Hello User. Welcome to Cinemas Report Generator!

Click Browse to choose the folder where your workbooks are placed

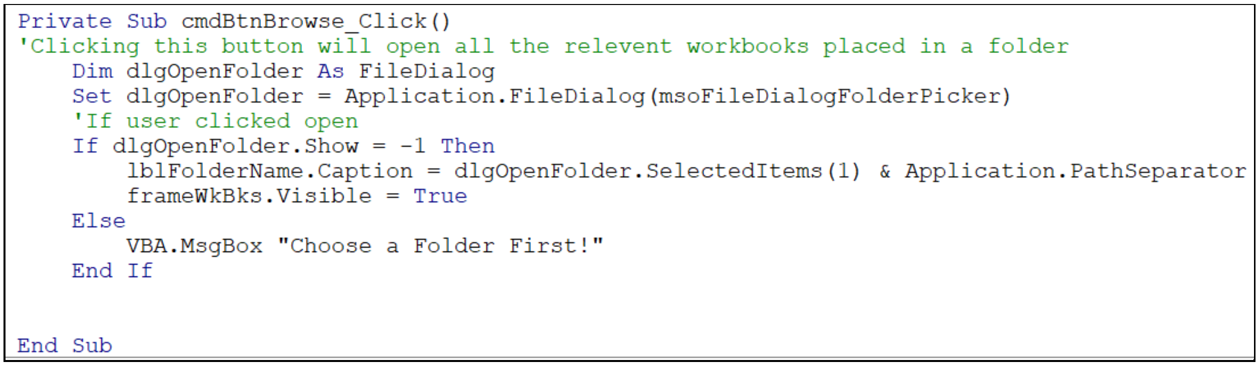
C:\Users\abhat\Desktop\MS3111\Project Workbooks\Workbooks\

Page “mulPgOpenFiles” consists of the Frame Control “frameOpenFiles”, the Label Control “lblEnterName”, TextBox Control “txtBoxName”, and CommandButton Control “cmdBtnEnter”.
Frame “frameOpenFiles” consists of Label Controls “lblWelcomeMsg”, “lblChooseFolder”, “lblFolderName”, CommandButton Control “cmdBtnBrowse” and another Frame Control “frameWkBks”.
Frame “frameWkBks” consists of CommandButton Controls “cmdBtnOpenWkbks”, “cmdBtnGenWkBk” and CheckBox Control “chkBoxViewWkBk”.



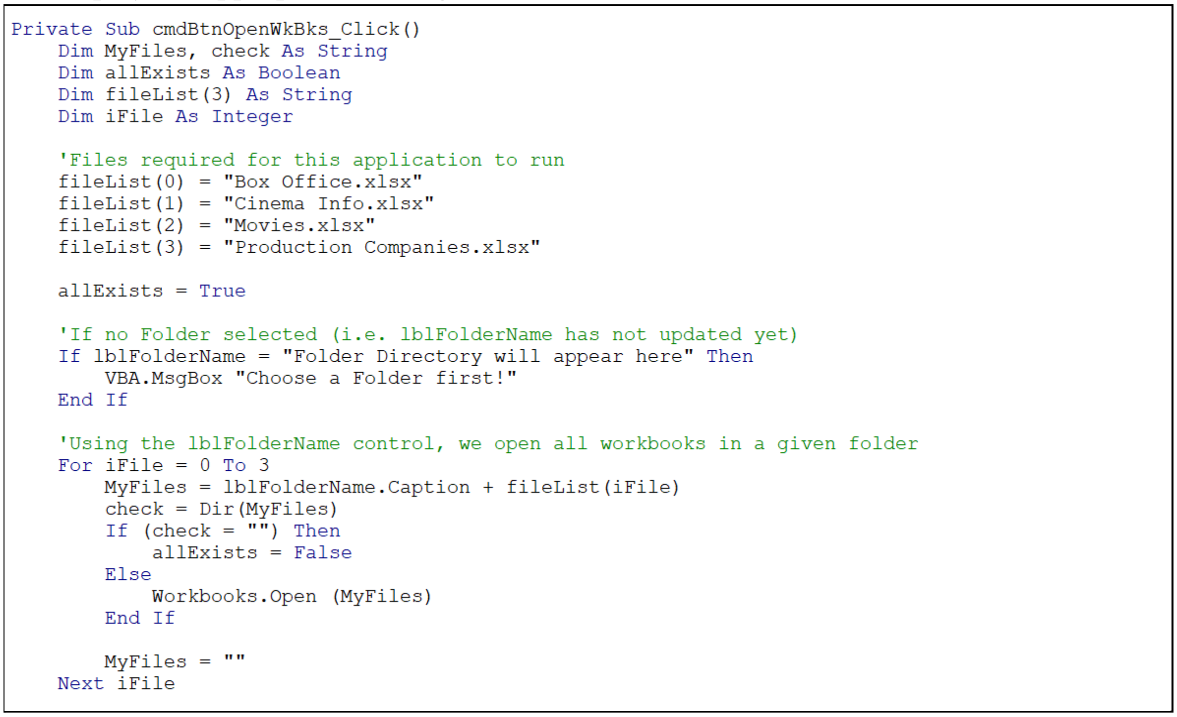
```
Private Sub cmdBtnEnter_Click()
    If txtBoxName.Value <> "" Then
        'Enable Frame to let user browse for folder containing workbooks
        frameOpenFiles.Visible = True
        'Welcome Message
        lblWelcomeMsg.Caption = "Hello " & txtBoxName.Value & ". Welcome to Cinemas Report Generator!'
    Else
    'If no name Entered, request user to enter name
        VBA.MsgBox "Enter your Name!"
    End If
    UserForm1.Show
End Sub
```

The above screenshot contains the code for the click event of the CommandButton “cmdBtnEnter”. After the user enters the name in the TextBox Control “txtBoxName”, it welcomes the user by changing the Caption property of the Label Control “lblWelcomeMsg” and enables the Frame Control “frameOpenFiles” by changing its Visible property to True. If the user does not enter the name, VBA.MsgBox function throws a dialogue box that prompts the user to enter something in the “txtBoxName” field.



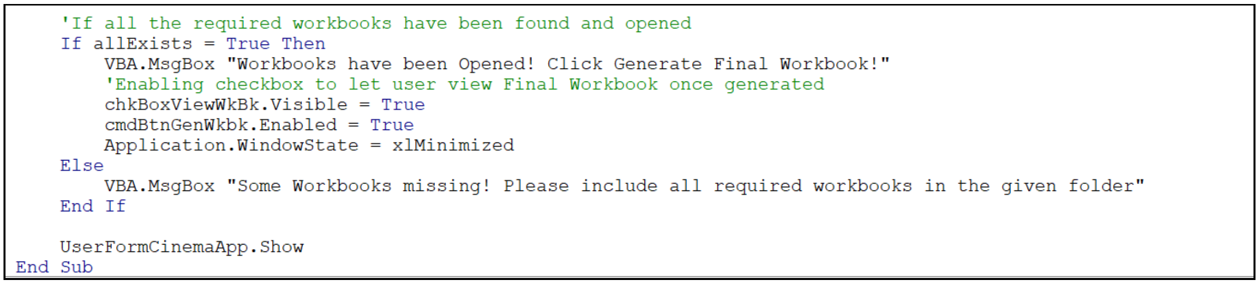
```
Private Sub cmdBtnBrowse_Click()
'Clicking this button will open all the relevent workbooks placed in a folder
    Dim dlgOpenFolder As FileDialog
    Set dlgOpenFolder = Application.FileDialog(msoFileDialogFolderPicker)
    'If user clicked open
    If dlgOpenFolder.Show = -1 Then
        lblFolderName.Caption = dlgOpenFolder.SelectedItems(1) & Application.PathSeparator
        frameWkBks.Visible = True
    Else
        VBA.MsgBox "Choose a Folder First!"
    End If


End Sub
```

The above screenshot contains the code for the click event of the CommandButton “cmdBtnBrowse”. It allows the user to choose the folder with relevant workbooks. If the user chooses and opens the folder, the application displays the path of the folder location in the Caption property of the Label Control “lblFolderName” and enables the Frame Control “frameWkBks” by changing the Visible property to True. Otherwise, the VBA.MsgBox function will display the appropriate message.



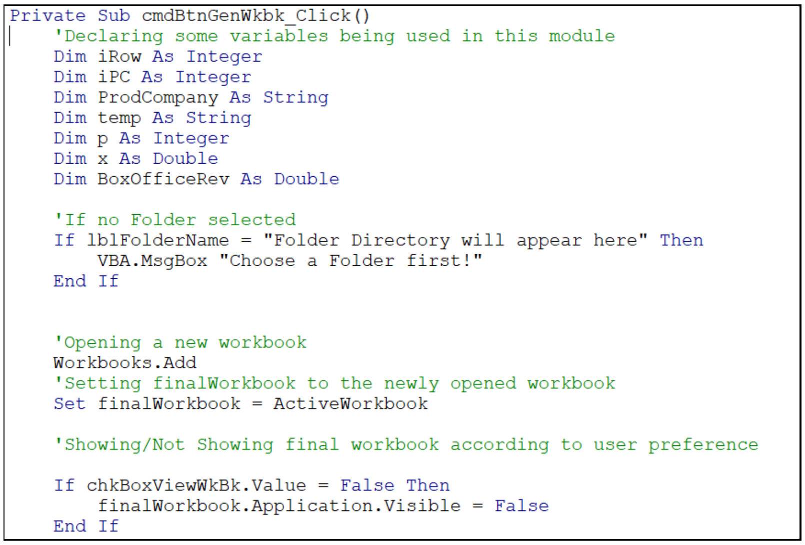
```
Private Sub cmdBtnOpenWkBks_Click()
    Dim MyFiles, check As String
    Dim allExists As Boolean
    Dim fileList(3) As String
    Dim iFile As Integer

    'Files required for this application to run
    fileList(0) = "Box Office.xlsx"
    fileList(1) = "Cinema Info.xlsx"
    fileList(2) = "Movies.xlsx"
    fileList(3) = "Production Companies.xlsx"

    allExists = True

    'If no Folder selected (i.e. lblFolderName has not updated yet)
    If lblFolderName = "Folder Directory will appear here" Then
        VBA.MsgBox "Choose a Folder first!"
    End If

    'Using the lblFolderName control, we open all workbooks in a given folder
    For iFile = 0 To 3
        MyFiles = lblFolderName.Caption + fileList(iFile)
        check = Dir(MyFiles)
        If (check = "") Then
            allExists = False
        Else
            Workbooks.Open (MyFiles)
        End If

        MyFiles = ""
    Next iFile
```

The above screenshot contains part of the code for the click event of the CommandButton “cmdBtnOpenWkBks”. We store the names of the datasets required for the proper functioning of this application in an array of string data types, called ‘fileList’. We then check if a folder is selected and if not, a dialog box is thrown to inform the user. Next, we check if all the required datasets are present in the chosen folder using a for loop. If all datasets are present, the boolean variable ‘allExists’ exits the for loop with the value ‘True.’



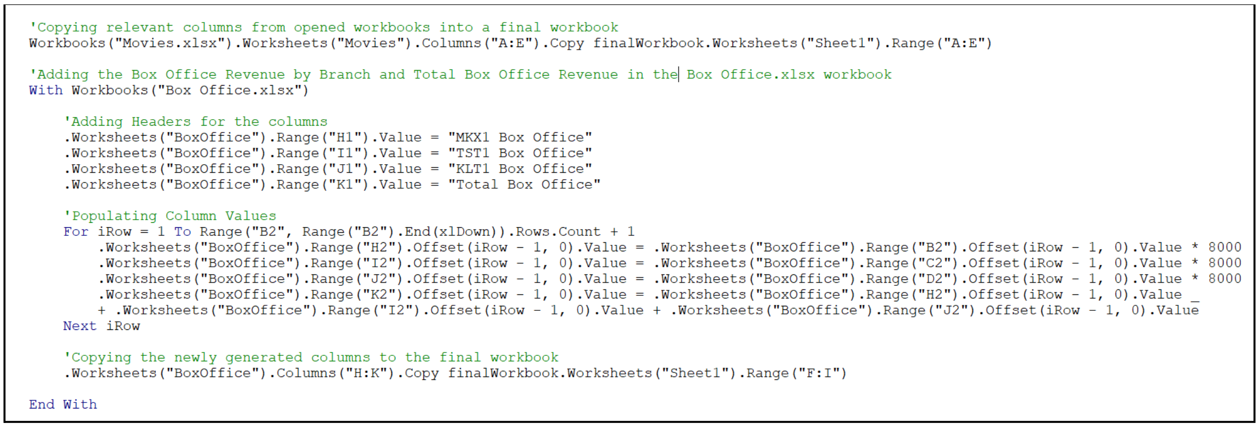
```
    'If all the required workbooks have been found and opened
    If allExists = True Then
        VBA.MsgBox "Workbooks have been Opened! Click Generate Final Workbook!"
        'Enabling checkbox to let user view Final Workbook once generated
        chkBoxViewWkBk.Visible = True
        cmdBtnGenWkbk.Enabled = True
        Application.WindowState = xlMinimized
    Else
        VBA.MsgBox "Some Workbooks missing! Please include all required workbooks in the given folder"
    End If

    UserFormCinemaApp.Show
End Sub
```

Now, if all the required datasets are present in the folder, we open the mentioned files programmatically. If not, we inform the user about the missing datasets. Once they are successfully opened, we enable “cmdBtnGenWkBk” so that the user can click the CommandButton Control to generate the Final Workbook (derived dataset) and can also check the CheckBox Control “chkBoxViewWkBk” before clicking the CommandButton to view the Final Workbook after generation.



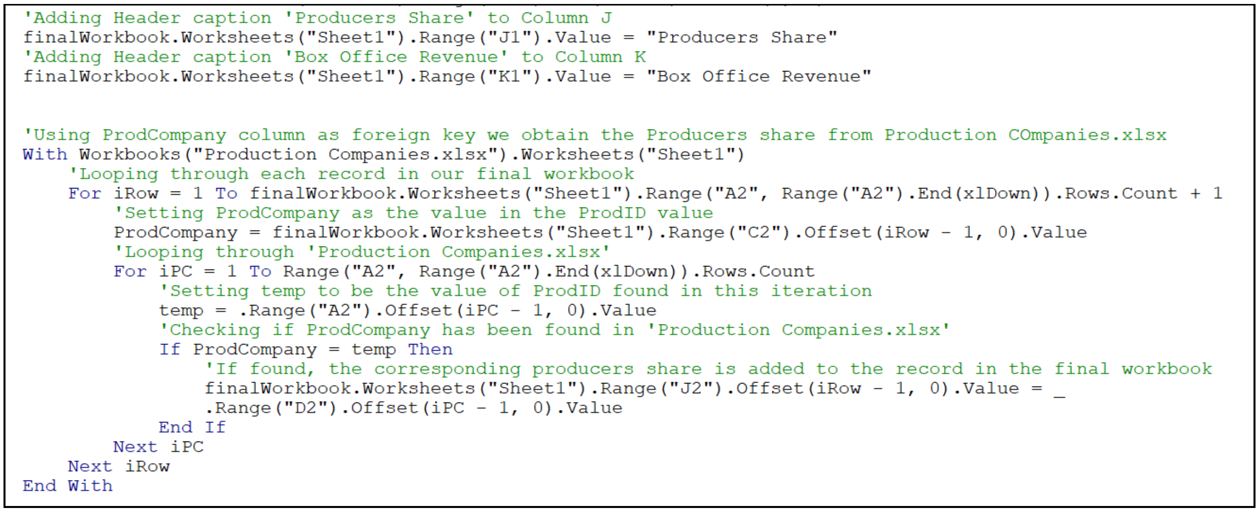
```
Private Sub cmdBtnGenWkbk_Click()
    'Declaring some variables being used in this module
    Dim iRow As Integer
    Dim iPC As Integer
    Dim ProdCompany As String
    Dim temp As String
    Dim p As Integer
    Dim x As Double
    Dim BoxOfficeRev As Double

    'If no Folder selected
    If lblFolderName = "Folder Directory will appear here" Then
        VBA.MsgBox "Choose a Folder first!"
    End If


    'Opening a new workbook
    Workbooks.Add
    'Setting finalWorkbook to the newly opened workbook
    Set finalWorkbook = ActiveWorkbook

    'Showing/Not Showing final workbook according to user preference

    If chkBoxViewWkBk.Value = False Then
        finalWorkbook.Application.Visible = False
    End If
```

The above and below screenshots contain the code for the click event of the CommandButton “cmdBtnGenWkBk”. Used variables were declared at a procedure level with appropriate data types. If no folder was selected by the user in the previous step, we inform the user. Next, it opens a new Workbook and sets it as the active Workbook.



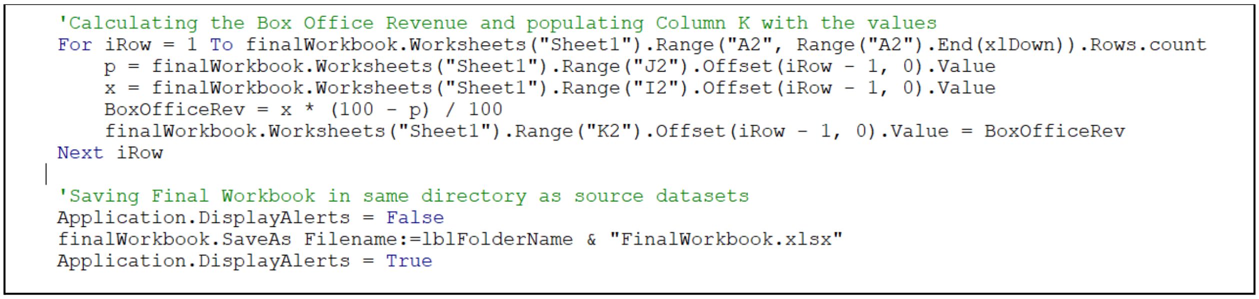
```
 'Copying relevant columns from opened workbooks into a final workbook
 Workbooks("Movies.xlsx").Worksheets("Movies").Columns("A:E").Copy finalWorkbook.Worksheets("Sheet1").Range("A:E")

 'Adding the Box Office Revenue by Branch and Total Box Office Revenue in the Box Office.xlsx workbook
 With Workbooks("Box Office.xlsx")

     'Adding Headers for the columns
     .Worksheets("BoxOffice").Range("H1").Value = "MKX1 Box Office"
     .Worksheets("BoxOffice").Range("I1").Value = "TST1 Box Office"
     .Worksheets("BoxOffice").Range("J1").Value = "KLT1 Box Office"
     .Worksheets("BoxOffice").Range("K1").Value = "Total Box Office"

     'Populating Column Values
     For iRow = 1 To Range("B2", Range("B2").End(xlDown)).Rows.Count + 1
         .Worksheets("BoxOffice").Range("H2").Offset(iRow - 1, 0).Value = .Worksheets("BoxOffice").Range("B2").Offset(iRow - 1, 0).Value * 8000
         .Worksheets("BoxOffice").Range("I2").Offset(iRow - 1, 0).Value = .Worksheets("BoxOffice").Range("C2").Offset(iRow - 1, 0).Value * 8000
         .Worksheets("BoxOffice").Range("J2").Offset(iRow - 1, 0).Value = .Worksheets("BoxOffice").Range("D2").Offset(iRow - 1, 0).Value * 8000
         .Worksheets("BoxOffice").Range("K2").Offset(iRow - 1, 0).Value = .Worksheets("BoxOffice").Range("H2").Offset(iRow - 1, 0).Value _
         + .Worksheets("BoxOffice").Range("I2").Offset(iRow - 1, 0).Value + .Worksheets("BoxOffice").Range("J2").Offset(iRow - 1, 0).Value
     Next iRow

     'Copying the newly generated columns to the final workbook
     .Worksheets("BoxOffice").Columns("H:K").Copy finalWorkbook.Worksheets("Sheet1").Range("F:I")

 End With
```

Next we add the headers and copy the relevant column values from *Movies.xlsx* to the newly generated final workbook. We also compute new variables "MKX1 Box Office", "TST1 Box Office", "KLT1 Box Office" and "Total Box Office." The calculation formula is straightforward. We multiply the shows of the respective cinema branch with 8000 (this value used for simplicity) to generate the total box office collection for a given movie in a given branch. "MKX1" is the CinemaID of the Mong Kok branch of the cinema. These CinemaIDs are available in *Cinema Info.xlsx.* We add the three variable's values and store the corresponding value in the "Total Box Office" Column. These new column variables are computed and populated in *Box Office.xlsx* first and then are copied over to our final workbook.



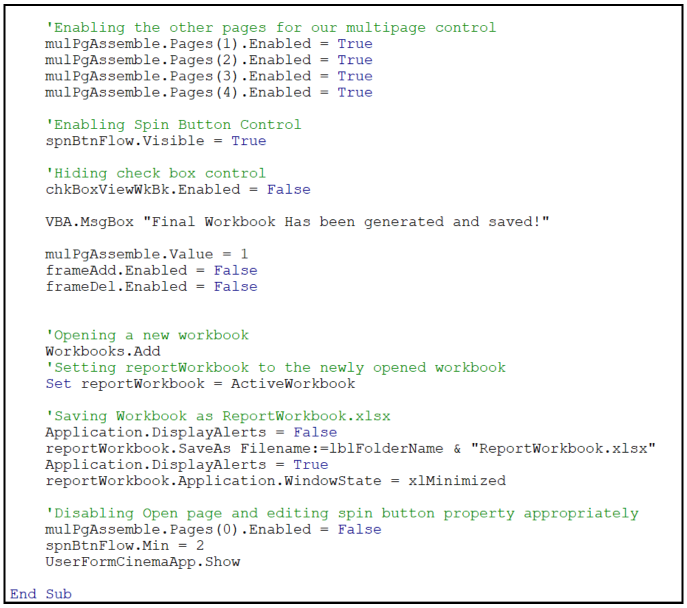
```
'Adding Header caption 'Producers Share' to Column J
finalWorkbook.Worksheets("Sheet1").Range("J1").Value = "Producers Share"
'Adding Header caption 'Box Office Revenue' to Column K
finalWorkbook.Worksheets("Sheet1").Range("K1").Value = "Box Office Revenue"


'Using ProdCompany column as foreign key we obtain the Producers share from Production COmpanies.xlsx
With Workbooks("Production Companies.xlsx").Worksheets("Sheet1")
    'Looping through each record in our final workbook
    For iRow = 1 To finalWorkbook.Worksheets("Sheet1").Range("A2", Range("A2").End(xlDown)).Rows.Count + 1
        'Setting ProdCompany as the value in the ProdID value
        ProdCompany = finalWorkbook.Worksheets("Sheet1").Range("C2").Offset(iRow - 1, 0).Value
        'Looping through 'Production Companies.xlsx'
        For iPC = 1 To Range("A2", Range("A2").End(xlDown)).Rows.Count
            'Setting temp to be the value of ProdID found in this iteration
            temp = .Range("A2").Offset(iPC - 1, 0).Value
            'Checking if ProdCompany has been found in 'Production Companies.xlsx'
            If ProdCompany = temp Then
                'If found, the corresponding producers share is added to the record in the final workbook
                finalWorkbook.Worksheets("Sheet1").Range("J2").Offset(iRow - 1, 0).Value = _
                .Range("D2").Offset(iPC - 1, 0).Value
            End If
        Next iPC
    Next iRow
End With
```

We use the ProdCompany column as the foreign key to obtain the Producers share values from *Production Companies.xlsx*. The corresponding Producers share is then added to the record in the final Workbook.



```
'Calculating the Box Office Revenue and populating Column K with the values
For iRow = 1 To finalWorkbook.Worksheets("Sheet1").Range("A2", Range("A2").End(xlDown)).Rows.count
    p = finalWorkbook.Worksheets("Sheet1").Range("J2").Offset(iRow - 1, 0).Value
    x = finalWorkbook.Worksheets("Sheet1").Range("I2").Offset(iRow - 1, 0).Value
    BoxOfficeRev = x * (100 - p) / 100
    finalWorkbook.Worksheets("Sheet1").Range("K2").Offset(iRow - 1, 0).Value = BoxOfficeRev
Next iRow

'Saving Final Workbook in same directory as source datasets
Application.DisplayAlerts = False
finalWorkbook.SaveAs Filename:=lblFolderName & "FinalWorkbook.xlsx"
Application.DisplayAlerts = True
```

The above screenshot shows how Box Office revenue is calculated. Box Office Revenue is another newly computed variable which is essentially how much the Cinema earns from a movie after giving away the share of the producers from the Total Box Office Collection. The corresponding column in the Final Workbook is populated. After generating the Final Workbook, the file is saved in the same directory as the selected folder as *FinalWorkbook.xlsx*.

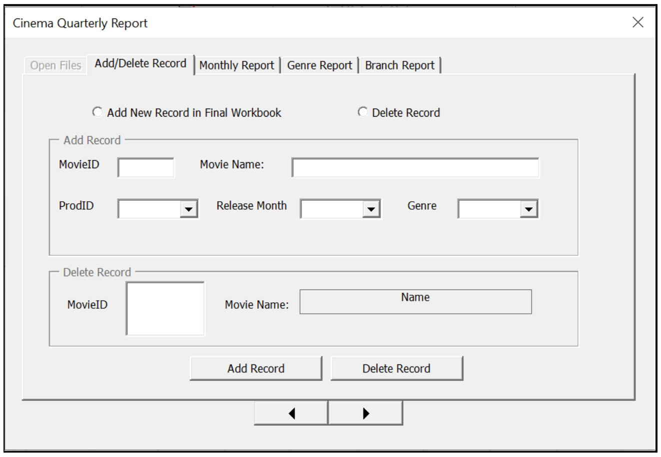


```
    'Enabling the other pages for our multipage control
    mulPgAssemble.Pages(1).Enabled = True
    mulPgAssemble.Pages(2).Enabled = True
    mulPgAssemble.Pages(3).Enabled = True
    mulPgAssemble.Pages(4).Enabled = True

    'Enabling Spin Button Control
    spnBtnFlow.Visible = True

    'Hiding check box control
    chkBoxViewWkBk.Enabled = False

    VBA.MsgBox "Final Workbook Has been generated and saved!"

    mulPgAssemble.Value = 1
    frameAdd.Enabled = False
    frameDel.Enabled = False


    'Opening a new workbook
    Workbooks.Add
    'Setting reportWorkbook to the newly opened workbook
    Set reportWorkbook = ActiveWorkbook

    'Saving Workbook as ReportWorkbook.xlsx
    Application.DisplayAlerts = False
    reportWorkbook.SaveAs Filename:=lblFolderName & "ReportWorkbook.xlsx"
    Application.DisplayAlerts = True
    reportWorkbook.Application.WindowState = xlMinimized

    'Disabling Open page and editing spin button property appropriately
    mulPgAssemble.Pages(0).Enabled = False
    spnBtnFlow.Min = 2
    UserFormCinemaApp.Show

End Sub
```

Other remaining Pages of the MultiPage are enabled along with the SpinButton Control, while the CheckBox Control is hidden. When the Final Workbook is generated, VBA Message Box displays the following message: "Final Workbook has been generated and saved!" The user is then directed to the next Page, where the User adds new records and deletes entered records.
To prepare this page we set frameAdd and frameDel's values to false. More will be elaborated on this in further sections

Along with generating the Final Workbook, the Application opens and saves a new Workbook for the Reports called *ReportWorkbook.xlsx*. We then disable the Open Files tab of our MultiPage Control as all the workbooks are now prepared and ready for reporting. We also change the Min value of our spin button control due to the same reason.

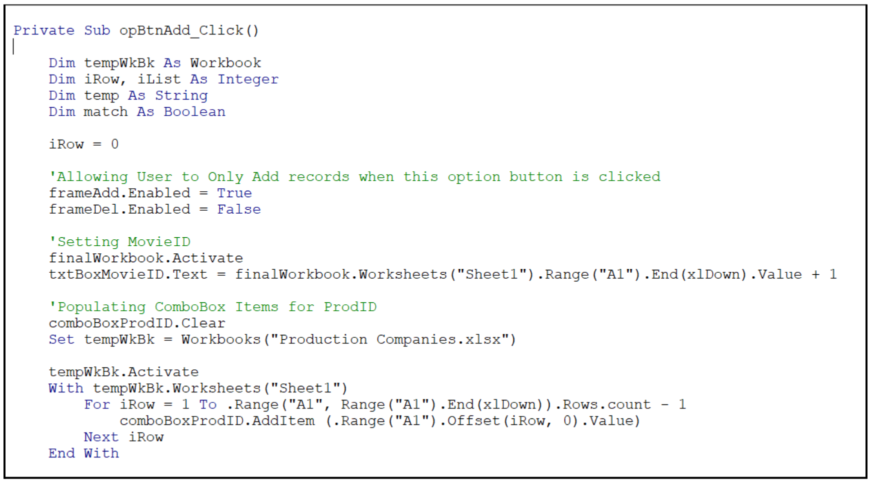
2.3 Page "mulPgAddDel" Controls



Page “mulPgAddDel” consists of the two Frame Controls “frameAdd”, “frameDel”, two OptionButton Controls “opBtnAdd”, “opBtnDel”, and two CommandButton Controls “cmdBtnAddRec”, “cmdButtonDel”

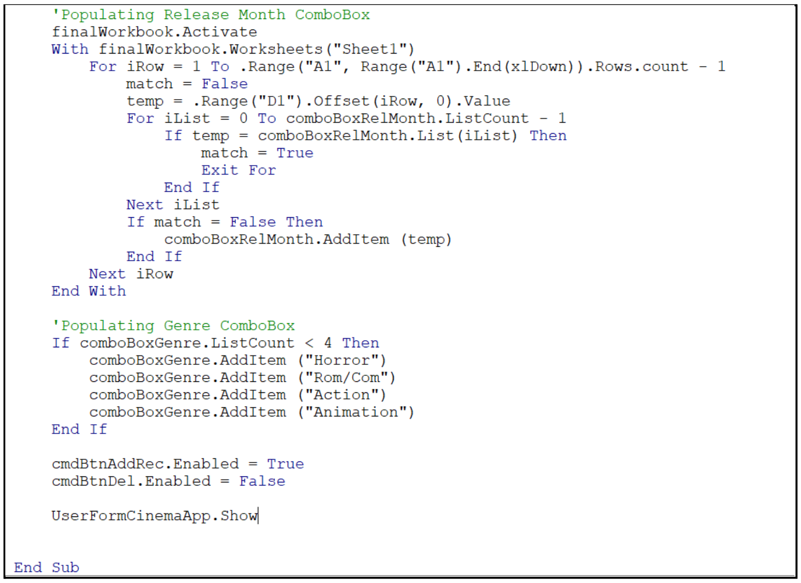
Frame Control “frameAdd” consists of five Label Controls “lblMovieID”, “lblMovieName”, “lblProdID”, “lblRelMonth”, and “lblGenre”, two TextBox Controls “txtBoxMovieID”, “txtBoxMovieName”, three ComboBox Controls “comboBoxProdID”, “comboBoxRelMonth”, “comboBoxGenre”

Frame Control “frameDel” consists of three Label Controls “lblMovieIDDel”, “lblMovieNameDel”, “lblDelMovieName”, and the ListBox Control “listBoxDel”
The screenshots below show the code for the Click() event of the Option Button Controls



```
Private Sub opBtnAdd_Click()

    Dim tempWkBk As Workbook
    Dim iRow, iList As Integer
    Dim temp As String
    Dim match As Boolean

    iRow = 0

    'Allowing User to Only Add records when this option button is clicked
    frameAdd.Enabled = True
    frameDel.Enabled = False

    'Setting MovieID
    finalWorkbook.Activate
    txtBoxMovieID.Text = finalWorkbook.Worksheets("Sheet1").Range("A1").End(xlDown).Value + 1

    'Populating ComboBox Items for ProdID
    comboBoxProdID.Clear
    Set tempWkBk = Workbooks("Production Companies.xlsx")

    tempWkBk.Activate
    With tempWkBk.Worksheets("Sheet1")
        For iRow = 1 To .Range("A1", Range("A1").End(xlDown)).Rows.count - 1
            comboBoxProdID.AddItem (.Range("A1").Offset(iRow, 0).Value)
        Next iRow
    End With
```

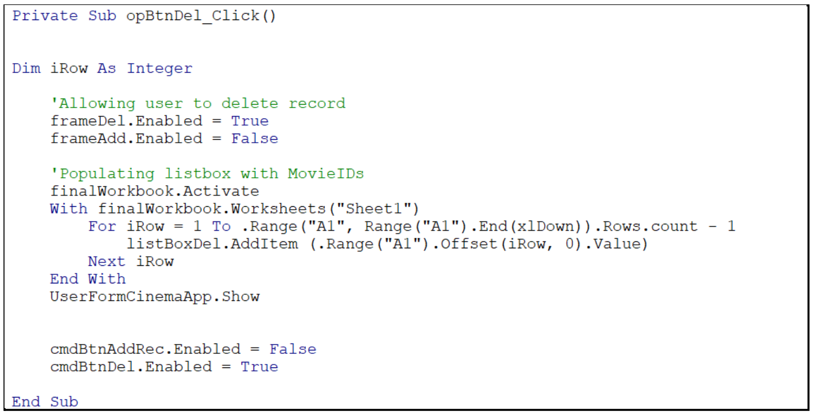
\



```
    'Populating Release Month ComboBox
    finalWorkbook.Activate
    With finalWorkbook.Worksheets("Sheet1")
        For iRow = 1 To .Range("A1", Range("A1").End(xlDown)).Rows.count - 1
            match = False
            temp = .Range("D1").Offset(iRow, 0).Value
            For iList = 0 To comboBoxRelMonth.ListCount - 1
                If temp = comboBoxRelMonth.List(iList) Then
                    match = True
                    Exit For
                End If
            Next iList
            If match = False Then
                comboBoxRelMonth.AddItem (temp)
            End If
        Next iRow
    End With

    'Populating Genre ComboBox
    If comboBoxGenre.ListCount < 4 Then
        comboBoxGenre.AddItem ("Horror")
        comboBoxGenre.AddItem ("Rom/Com")
        comboBoxGenre.AddItem ("Action")
        comboBoxGenre.AddItem ("Animation")
    End If

    cmdBtnAddRec.Enabled = True
    cmdBtnDel.Enabled = False

    UserFormCinemaApp.Show


End Sub
```

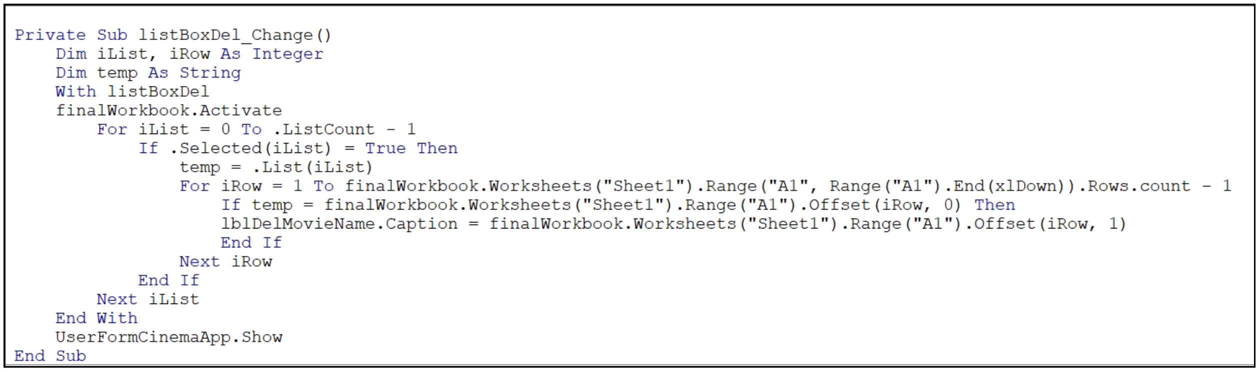
The above screenshots show the code we entered for setting the field value of “txtBoxMovieID” and populating the Combo Box Controls. If this option button is selected, then we disable “cmdBtnDel” and enable “cmdBtnAddRec”

The below screenshot shows the Click() event code for the “opBtnDel” button. We populate the listbox control and toggle the enabling of Command Buttons from what is seen in the previous screenshot



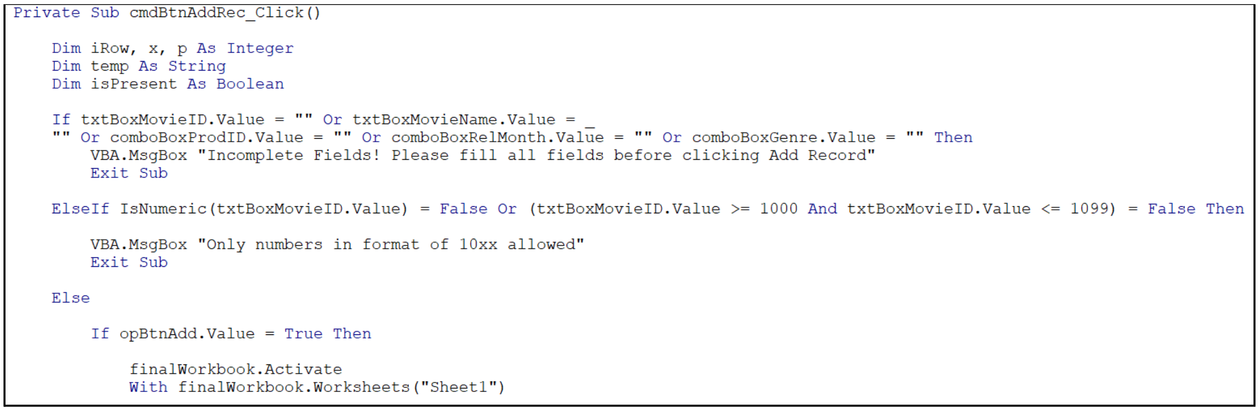
```
Private Sub opBtnDel_Click()


Dim iRow As Integer

    'Allowing user to delete record
    frameDel.Enabled = True
    frameAdd.Enabled = False

    'Populating listbox with MovieIDs
    finalWorkbook.Activate
    With finalWorkbook.Worksheets("Sheet1")
        For iRow = 1 To .Range("A1", Range("A1").End(xlDown)).Rows.count - 1
            listBoxDel.AddItem (.Range("A1").Offset(iRow, 0).Value)
        Next iRow
    End With
    UserFormCinemaApp.Show


    cmdBtnAddRec.Enabled = False
    cmdBtnDel.Enabled = True

End Sub
```

The code below shows the Change() event of the “listBoxDel” control. Here, upon selection of listbox item by user, we display the corresponding Movie Name of the selected Movie ID by changing the caption property of the “lblDelMovieName” Control.



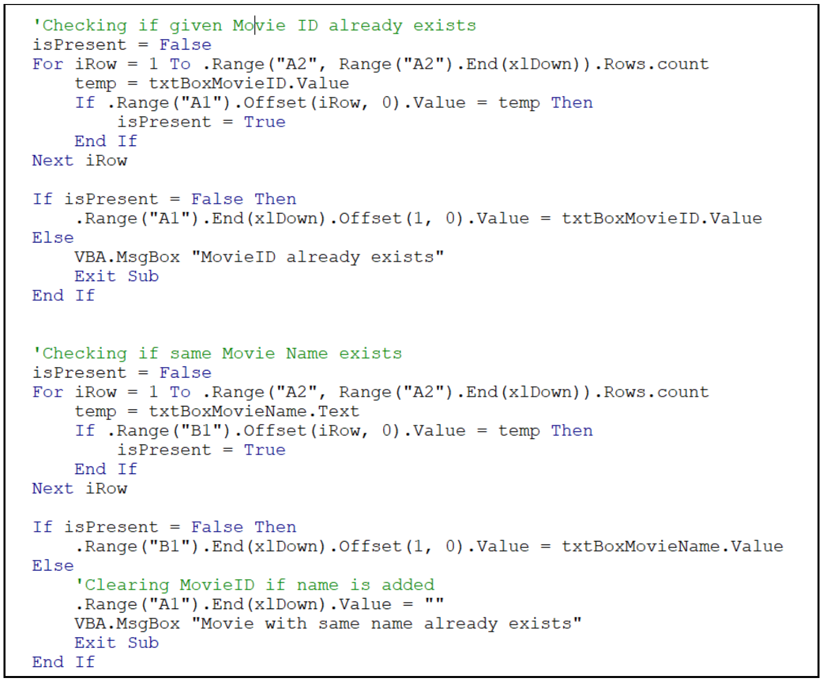
```
Private Sub listBoxDel_Change()
    Dim iList, iRow As Integer
    Dim temp As String
    With listBoxDel
    finalWorkbook.Activate
        For iList = 0 To .ListCount - 1
            If .Selected(iList) = True Then
                temp = .List(iList)
                For iRow = 1 To finalWorkbook.Worksheets("Sheet1").Range("A1", Range("A1").End(xlDown)).Rows.count - 1
                    If temp = finalWorkbook.Worksheets("Sheet1").Range("A1").Offset(iRow, 0) Then
                    lblDelMovieName.Caption = finalWorkbook.Worksheets("Sheet1").Range("A1").Offset(iRow, 1)
                    End If
                Next iRow
            End If
        Next iList
    End With
    UserFormCinemaApp.Show
End Sub
```

The below screenshots contain the code for the click event of the CommandButton “cmdBtnAddRec”.



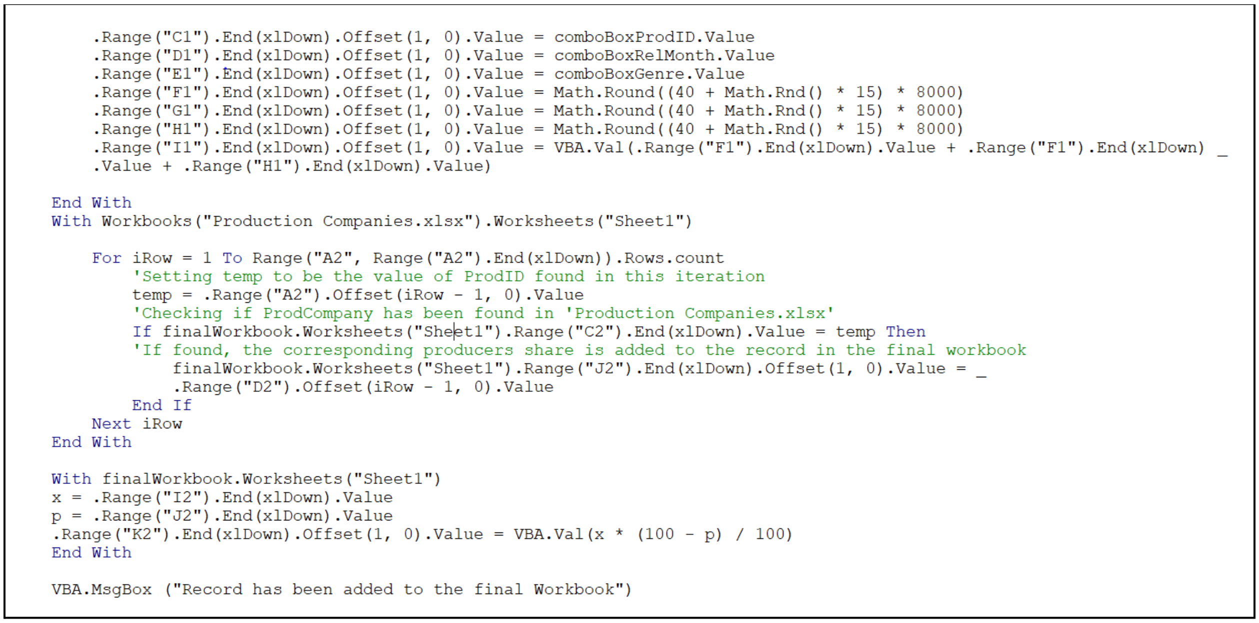
```
Private Sub cmdBtnAddRec_Click()

    Dim iRow, x, p As Integer
    Dim temp As String
    Dim isPresent As Boolean

    If txtBoxMovieID.Value = "" Or txtBoxMovieName.Value = _
    "" Or comboBoxProdID.Value = "" Or comboBoxRelMonth.Value = "" Or comboBoxGenre.Value = "" Then
        VBA.MsgBox "Incomplete Fields! Please fill all fields before clicking Add Record"
        Exit Sub

    ElseIf IsNumeric(txtBoxMovieID.Value) = False Or (txtBoxMovieID.Value >= 1000 And txtBoxMovieID.Value <= 1099) = False Then

        VBA.MsgBox "Only numbers in format of 10xx allowed"
        Exit Sub

    Else

        If opBtnAdd.Value = True Then

            finalWorkbook.Activate
            With finalWorkbook.Worksheets("Sheet1")
```

The “If-Else” statement was used to check if the user has filled all required fields for adding the new record. If yes, the Application reads the entered values and adds them to the last row of the final workbook. To keep the MovieID’s consistent with the record, the allowed format for entering the Movie ID is “10xx” where xx are the numbers from 00 to 99. Otherwise, the VBA Message Box informs the User that only numbers in format 10xx are allowed. It also gives this message if the user tries to enter a non numeric value into the field.



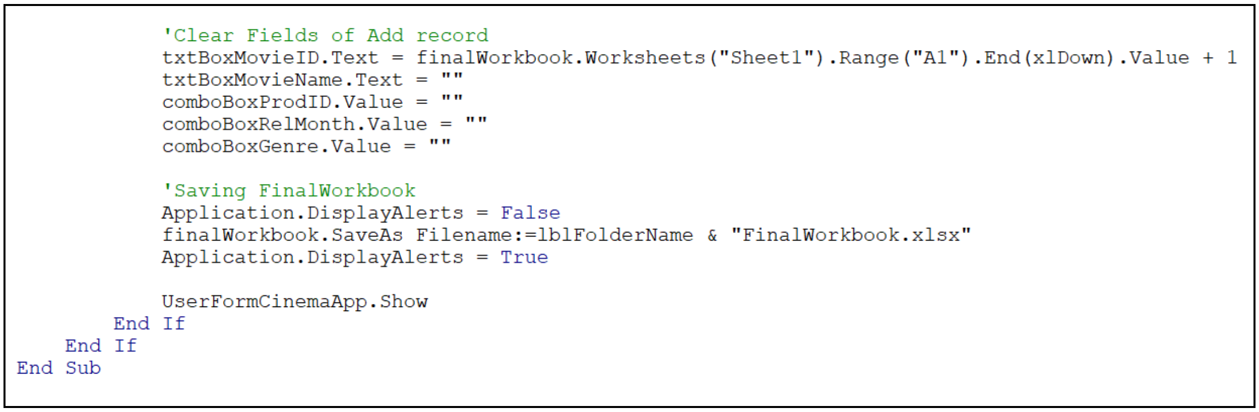
```
'Checking if given Movie ID already exists
isPresent = False
For iRow = 1 To .Range("A2", Range("A2").End(xlDown)).Rows.count
    temp = txtBoxMovieID.Value
    If .Range("A1").Offset(iRow, 0).Value = temp Then
        isPresent = True
    End If
Next iRow

If isPresent = False Then
    .Range("A1").End(xlDown).Offset(1, 0).Value = txtBoxMovieID.Value
Else
    VBA.MsgBox "MovieID already exists"
    Exit Sub
End If


'Checking if same Movie Name exists
isPresent = False
For iRow = 1 To .Range("A2", Range("A2").End(xlDown)).Rows.count
    temp = txtBoxMovieName.Text
    If .Range("B1").Offset(iRow, 0).Value = temp Then
        isPresent = True
    End If
Next iRow

If isPresent = False Then
    .Range("B1").End(xlDown).Offset(1, 0).Value = txtBoxMovieName.Value
Else
    'Clearing MovieID if name is added
    .Range("A1").End(xlDown).Value = ""
    VBA.MsgBox "Movie with same name already exists"
    Exit Sub
End If
```

Next, it checks if the entered Movie ID or movie Name entered in the field already exists using the For Loop and If statements. If they already exist, VBA Message Box informs the user and doesn't add the record



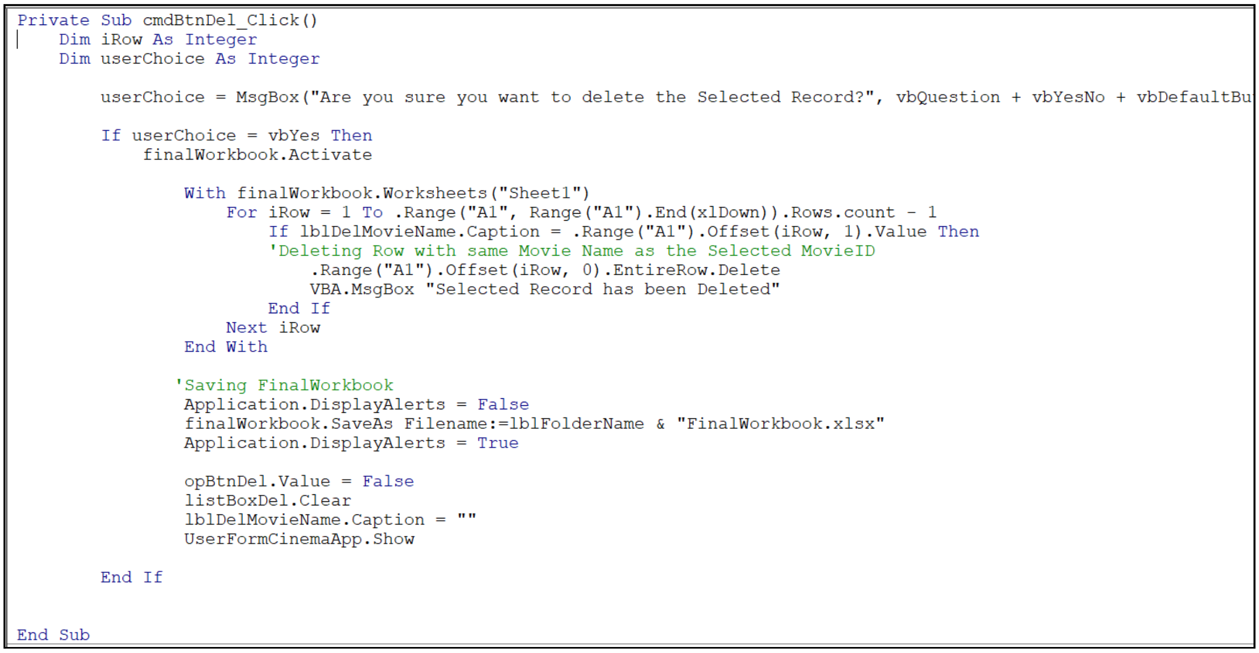
```
        .Range("C1").End(xlDown).Offset(1, 0).Value = comboBoxProdID.Value
        .Range("D1").End(xlDown).Offset(1, 0).Value = comboBoxRelMonth.Value
        .Range("E1").End(xlDown).Offset(1, 0).Value = comboBoxGenre.Value
        .Range("F1").End(xlDown).Offset(1, 0).Value = Math.Round((40 + Math.Rnd() * 15) * 8000)
        .Range("G1").End(xlDown).Offset(1, 0).Value = Math.Round((40 + Math.Rnd() * 15) * 8000)
        .Range("H1").End(xlDown).Offset(1, 0).Value = Math.Round((40 + Math.Rnd() * 15) * 8000)
        .Range("I1").End(xlDown).Offset(1, 0).Value = VBA.Val(.Range("F1").End(xlDown).Value + .Range("F1").End(xlDown) _
        .Value + .Range("H1").End(xlDown).Value)

    End With
    With Workbooks("Production Companies.xlsx").Worksheets("Sheet1")

        For iRow = 1 To Range("A2", Range("A2").End(xlDown)).Rows.count
            'Setting temp to be the value of ProdID found in this iteration
            temp = .Range("A2").Offset(iRow - 1, 0).Value
            'Checking if ProdCompany has been found in 'Production Companies.xlsx'
            If finalWorkbook.Worksheets("Sheet1").Range("C2").End(xlDown).Value = temp Then
            'If found, the corresponding producers share is added to the record in the final workbook
                finalWorkbook.Worksheets("Sheet1").Range("J2").End(xlDown).Offset(1, 0).Value = _
                .Range("D2").Offset(iRow - 1, 0).Value
            End If
        Next iRow
    End With

    With finalWorkbook.Worksheets("Sheet1")
    x = .Range("I2").End(xlDown).Value
    p = .Range("J2").End(xlDown).Value
    .Range("K2").End(xlDown).Offset(1, 0).Value = VBA.Val(x * (100 - p) / 100)
    End With

    VBA.MsgBox ("Record has been added to the final Workbook")
```

When the records are added, the Application clears the fields and values of the Add Record Frame Controls.



```
            'Clear Fields of Add record
            txtBoxMovieID.Text = finalWorkbook.Worksheets("Sheet1").Range("A1").End(xlDown).Value + 1
            txtBoxMovieName.Text = ""
            comboBoxProdID.Value = ""
            comboBoxRelMonth.Value = ""
            comboBoxGenre.Value = ""

            'Saving FinalWorkbook
            Application.DisplayAlerts = False
            finalWorkbook.SaveAs Filename:=lblFolderName & "FinalWorkbook.xlsx"
            Application.DisplayAlerts = True

            UserFormCinemaApp.Show
        End If
    End If
End Sub
```

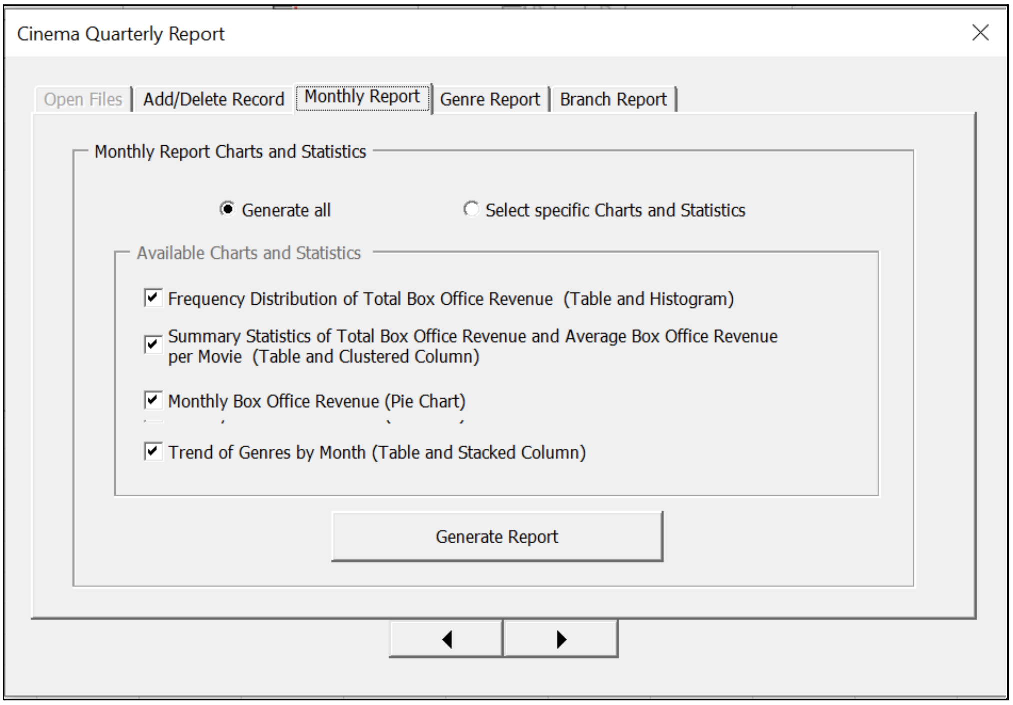
The below screenshot contains the code for the click event of the CommandButton “cmdButtonDel”.



```
Private Sub cmdBtnDel_Click()
    Dim iRow As Integer
    Dim userChoice As Integer

        userChoice = MsgBox("Are you sure you want to delete the Selected Record?", vbQuestion + vbYesNo + vbDefaultBu

        If userChoice = vbYes Then
            finalWorkbook.Activate

                With finalWorkbook.Worksheets("Sheet1")
                    For iRow = 1 To .Range("A1", Range("A1").End(xlDown)).Rows.count - 1
                        If lblDelMovieName.Caption = .Range("A1").Offset(iRow, 1).Value Then
                        'Deleting Row with same Movie Name as the Selected MovieID
                            .Range("A1").Offset(iRow, 0).EntireRow.Delete
                            VBA.MsgBox "Selected Record has been Deleted"
                        End If
                    Next iRow
                End With

               'Saving FinalWorkbook
                Application.DisplayAlerts = False
                finalWorkbook.SaveAs Filename:=lblFolderName & "FinalWorkbook.xlsx"
                Application.DisplayAlerts = True

                opBtnDel.Value = False
                listBoxDel.Clear
                lblDelMovieName.Caption = ""
                UserFormCinemaApp.Show

        End If

End Sub
```

The application confirms if the user wants to delete the record using a VBA MsgBox. If the user clicks Yes in the corresponding dialog box, then We find the record and delete it using the “EntireRow.Delete” method. We then inform the user about the deletion using a VBA Msg Box again. We then save the file and clear the ListBox and movie name label.

2.4 Page “mulPgMonthlyReport” Controls



● Generate all

○ Select specific Charts and Statistics

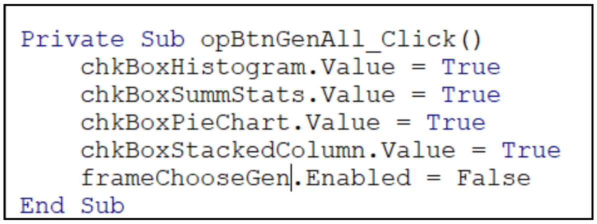
☑ Frequency Distribution of Total Box Office Revenue (Table and Histogram)

☑ Summary Statistics of Total Box Office Revenue and Average Box Office Revenue per Movie (Table and Clustered Column)

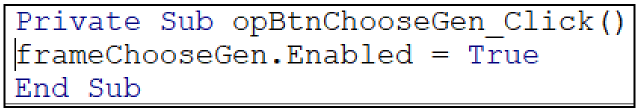
☑ Monthly Box Office Revenue (Pie Chart)

☑ Trend of Genres by Month (Table and Stacked Column)

Page “mulPgMonthlyReport” consists of the Frame Control “frameMonthlyReport” which contains two OptionButton Controls “opBtnGenAll”, “opBtnChooseGen”, one CommandButton Control “cmdBtnMonthlyReport” and another Frame Control “frameChooseGen” which contains four CheckBox Controls “chkBoxHistogram”, “chkBoxSummStats”, “chkBoxPieChart”, “chkBoxStackedColumn.”

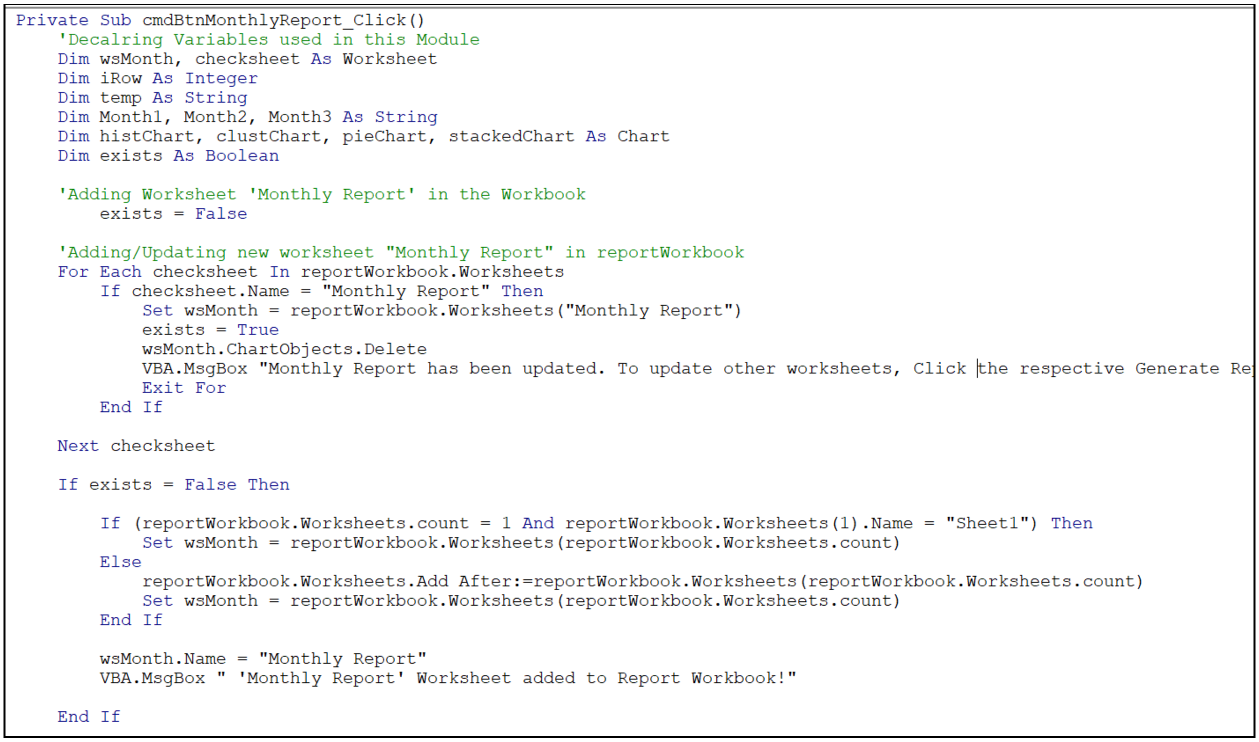


```
Private Sub opBtnGenAll_Click()
    chkBoxHistogram.Value = True
    chkBoxSummStats.Value = True
    chkBoxPieChart.Value = True
    chkBoxStackedColumn.Value = True
    frameChooseGen.Enabled = False
End Sub
```



```
Private Sub opBtnChooseGen_Click()
frameChooseGen.Enabled = True
End Sub
```

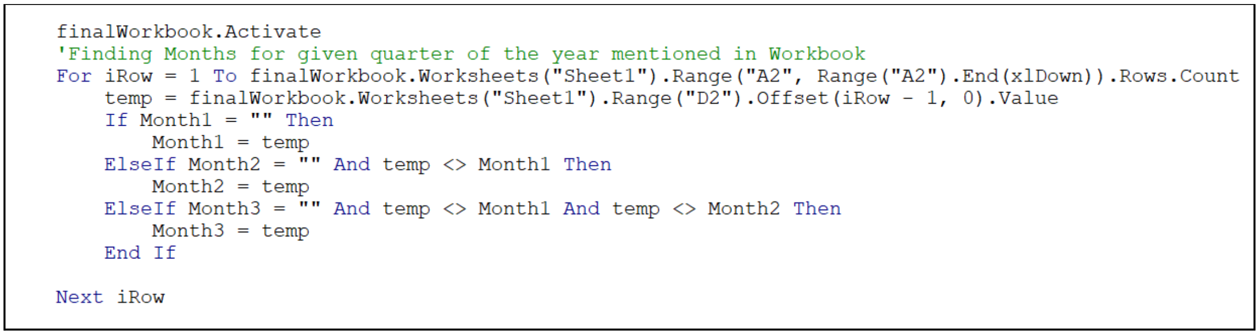
The above screenshots contain the code for the click event of the OptionButton Controls. When optBtnGenAll is clicked, it changes the Value Property of all CheckBox Controls to True and disables the Frame Control “frameChooseGen”. When the opBtnChooseGen is clicked, the Frame Control is enabled so that the user could check appropriate CheckBox Controls for the needed charts and statistics as per their preference.



```
Private Sub cmdBtnMonthlyReport_Click()
    'Decalring Variables used in this Module
    Dim wsMonth, checksheet As Worksheet
    Dim iRow As Integer
    Dim temp As String
    Dim Month1, Month2, Month3 As String
    Dim histChart, clustChart, pieChart, stackedChart As Chart
    Dim exists As Boolean

    'Adding Worksheet 'Monthly Report' in the Workbook
        exists = False

    'Adding/Updating new worksheet "Monthly Report" in reportWorkbook
    For Each checksheet In reportWorkbook.Worksheets
        If checksheet.Name = "Monthly Report" Then
            Set wsMonth = reportWorkbook.Worksheets("Monthly Report")
            exists = True
            wsMonth.ChartObjects.Delete
            VBA.MsgBox "Monthly Report has been updated. To update other worksheets, Click the respective Generate Re
            Exit For
        End If

    Next checksheet

    If exists = False Then

        If (reportWorkbook.Worksheets.count = 1 And reportWorkbook.Worksheets(1).Name = "Sheet1") Then
            Set wsMonth = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
        Else
            reportWorkbook.Worksheets.Add After:=reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
            Set wsMonth = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
        End If

        wsMonth.Name = "Monthly Report"
        VBA.MsgBox " 'Monthly Report' Worksheet added to Report Workbook!"

    End If
```

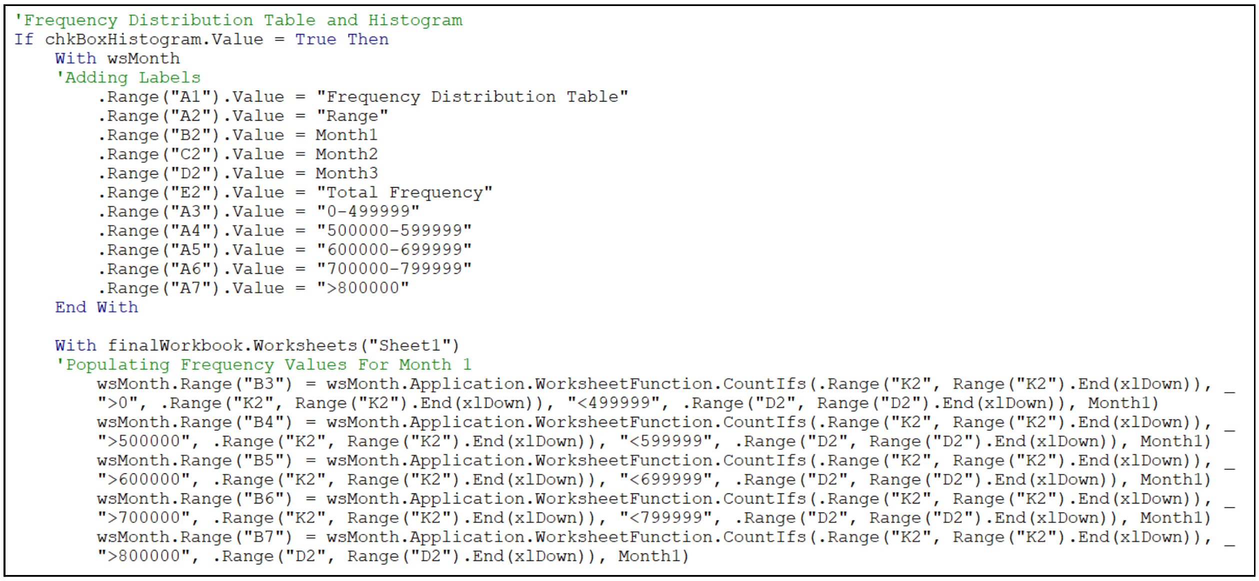
The above and below screenshots contain the code for the click event of the CommandButton Control “cmdBtnMonthlyReport”. Used variables were declared at a procedure level with appropriate data types. The application is designed in a way which allows the user to generate any of the 3 report sheets in any order of preference and even update them after adding/deleting records. Hence, we use a for loop to check if the worksheet has already been opened. The boolean variable ‘exists’ exits the loop with the value ‘True’ if it is the case. We use variable ‘wsMonth’ to refer to this sheet and set it to the same. We also delete previously made charts as they will need to be updated.

If ‘exists’ exits the loop with a value of ‘False’, then the worksheet has to be created. The nested if condition under the ‘If exists=False then’ avoids different Report sheets to overwrite each other as it renames “Sheet1” which would be the default worksheet upon creation of the workbook and if no such sheet exists, it adds a sheet after the already existing worksheets. We then rename ‘wsMonth’ to “Monthly Report” to complete the process.

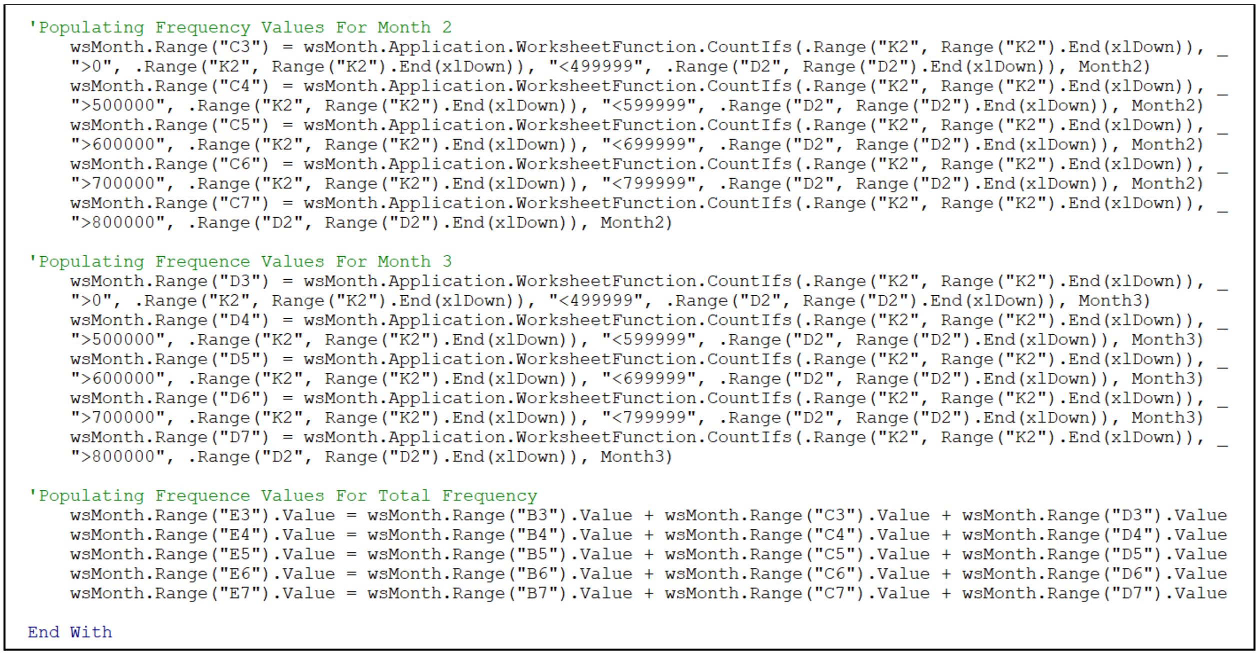


```
    finalWorkbook.Activate
    'Finding Months for given quarter of the year mentioned in Workbook
    For iRow = 1 To finalWorkbook.Worksheets("Sheet1").Range("A2", Range("A2").End(xlDown)).Rows.Count
        temp = finalWorkbook.Worksheets("Sheet1").Range("D2").Offset(iRow - 1, 0).Value
        If Month1 = "" Then
            Month1 = temp
        ElseIf Month2 = "" And temp <> Month1 Then
            Month2 = temp
        ElseIf Month3 = "" And temp <> Month1 And temp <> Month2 Then
            Month3 = temp
        End If

    Next iRow
```

The above section of code stores the three months of the given quarter in the variables Month1, Month2, Month3. For this particular set of workbooks, the values are found to be Oct, Nov, Dec as these workbooks are for the fourth quarter of the year

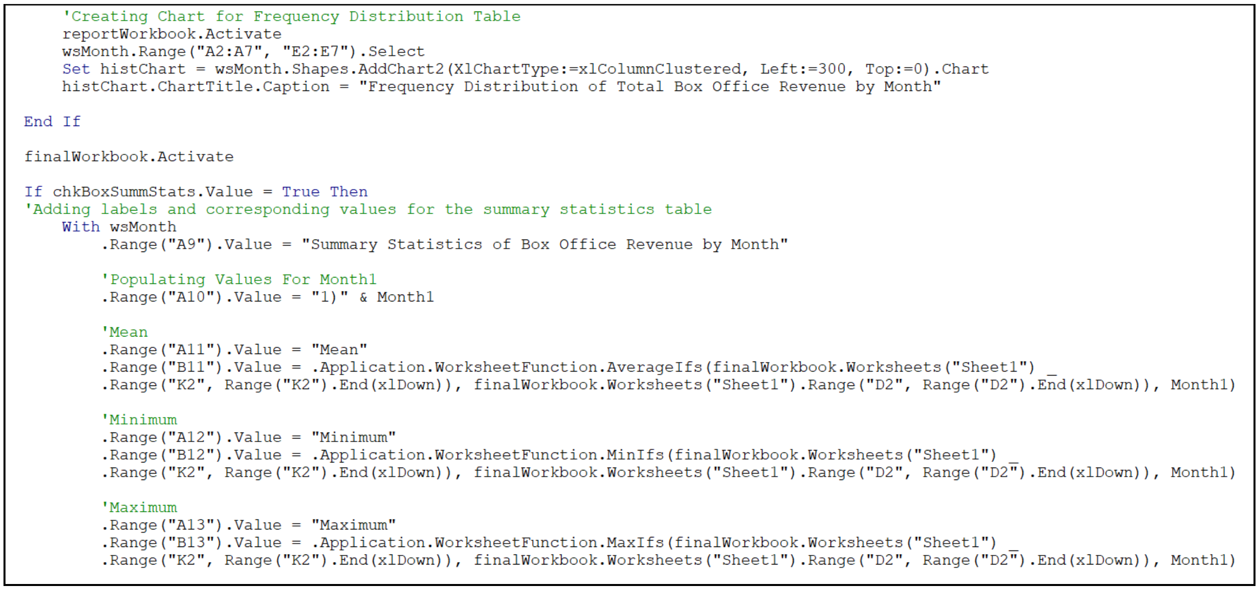


```
'Frequency Distribution Table and Histogram
If chkBoxHistogram.Value = True Then
    With wsMonth
    'Adding Labels
        .Range("A1").Value = "Frequency Distribution Table"
        .Range("A2").Value = "Range"
        .Range("B2").Value = Month1
        .Range("C2").Value = Month2
        .Range("D2").Value = Month3
        .Range("E2").Value = "Total Frequency"
        .Range("A3").Value = "0-499999"
        .Range("A4").Value = "500000-599999"
        .Range("A5").Value = "600000-699999"
        .Range("A6").Value = "700000-799999"
        .Range("A7").Value = ">800000"
    End With

    With finalWorkbook.Worksheets("Sheet1")
    'Populating Frequency Values For Month 1
        wsMonth.Range("B3") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
        ">0", .Range("K2", Range("K2").End(xlDown)), "<499999", .Range("D2", Range("D2").End(xlDown)), Month1)
        wsMonth.Range("B4") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
        ">500000", .Range("K2", Range("K2").End(xlDown)), "<599999", .Range("D2", Range("D2").End(xlDown)), Month1)
        wsMonth.Range("B5") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
        ">600000", .Range("K2", Range("K2").End(xlDown)), "<699999", .Range("D2", Range("D2").End(xlDown)), Month1)
        wsMonth.Range("B6") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
        ">700000", .Range("K2", Range("K2").End(xlDown)), "<799999", .Range("D2", Range("D2").End(xlDown)), Month1)
        wsMonth.Range("B7") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
        ">800000", .Range("D2", Range("D2").End(xlDown)), Month1)
```

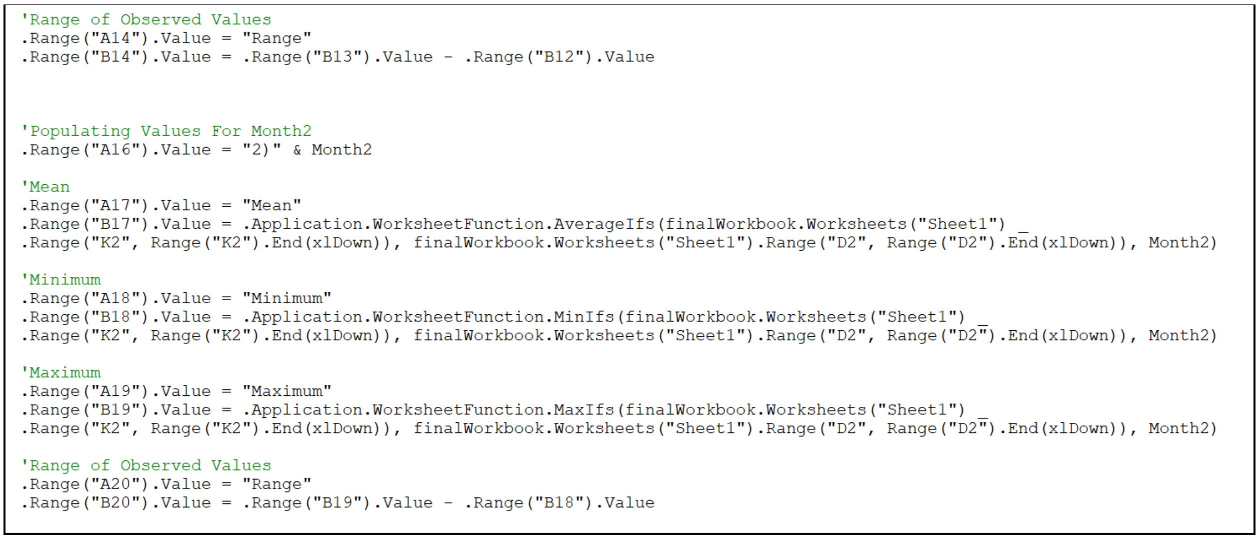


```
 'Populating Frequency Values For Month 2
    wsMonth.Range("C3") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">0", .Range("K2", Range("K2").End(xlDown)), "<499999", .Range("D2", Range("D2").End(xlDown)), Month2)
    wsMonth.Range("C4") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">500000", .Range("K2", Range("K2").End(xlDown)), "<599999", .Range("D2", Range("D2").End(xlDown)), Month2)
    wsMonth.Range("C5") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">600000", .Range("K2", Range("K2").End(xlDown)), "<699999", .Range("D2", Range("D2").End(xlDown)), Month2)
    wsMonth.Range("C6") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">700000", .Range("K2", Range("K2").End(xlDown)), "<799999", .Range("D2", Range("D2").End(xlDown)), Month2)
    wsMonth.Range("C7") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">800000", .Range("D2", Range("D2").End(xlDown)), Month2)

 'Populating Frequence Values For Month 3
    wsMonth.Range("D3") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">0", .Range("K2", Range("K2").End(xlDown)), "<499999", .Range("D2", Range("D2").End(xlDown)), Month3)
    wsMonth.Range("D4") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">500000", .Range("K2", Range("K2").End(xlDown)), "<599999", .Range("D2", Range("D2").End(xlDown)), Month3)
    wsMonth.Range("D5") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">600000", .Range("K2", Range("K2").End(xlDown)), "<699999", .Range("D2", Range("D2").End(xlDown)), Month3)
    wsMonth.Range("D6") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">700000", .Range("K2", Range("K2").End(xlDown)), "<799999", .Range("D2", Range("D2").End(xlDown)), Month3)
    wsMonth.Range("D7") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
    ">800000", .Range("D2", Range("D2").End(xlDown)), Month3)

 'Populating Frequence Values For Total Frequency
    wsMonth.Range("E3").Value = wsMonth.Range("B3").Value + wsMonth.Range("C3").Value + wsMonth.Range("D3").Value
    wsMonth.Range("E4").Value = wsMonth.Range("B4").Value + wsMonth.Range("C4").Value + wsMonth.Range("D4").Value
    wsMonth.Range("E5").Value = wsMonth.Range("B5").Value + wsMonth.Range("C5").Value + wsMonth.Range("D5").Value
    wsMonth.Range("E6").Value = wsMonth.Range("B6").Value + wsMonth.Range("C6").Value + wsMonth.Range("D6").Value
    wsMonth.Range("E7").Value = wsMonth.Range("B7").Value + wsMonth.Range("C7").Value + wsMonth.Range("D7").Value

 End With
```

The above screenshots show the code for the frequency distribution table and the histogram for three months. We make use of excel functions like CountIfs which allow us to enter multiple ranges and conditions which are to be checked for those ranges in pairs as parameters using "Application.WorksheetFunction.CountIfs(range1,condition,range2,…)." We populate these frequency values by month and also in total adding various levels to our frequency table for more insight.

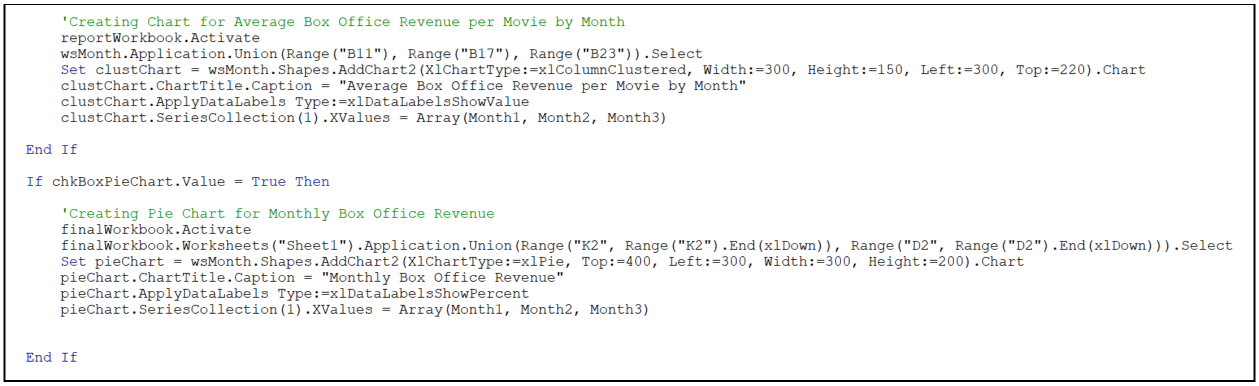


```
    'Creating Chart for Frequency Distribution Table
    reportWorkbook.Activate
    wsMonth.Range("A2:A7", "E2:E7").Select
    Set histChart = wsMonth.Shapes.AddChart2(XlChartType:=xlColumnClustered, Left:=300, Top:=0).Chart
    histChart.ChartTitle.Caption = "Frequency Distribution of Total Box Office Revenue by Month"

End If

finalWorkbook.Activate

If chkBoxSummStats.Value = True Then
'Adding labels and corresponding values for the summary statistics table
    With wsMonth
        .Range("A9").Value = "Summary Statistics of Box Office Revenue by Month"

        'Populating Values For Month1
        .Range("A10").Value = "1)" & Month1

        'Mean
        .Range("A11").Value = "Mean"
        .Range("B11").Value = .Application.WorksheetFunction.AverageIfs(finalWorkbook.Worksheets("Sheet1") _
        .Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month1)

        'Minimum
        .Range("A12").Value = "Minimum"
        .Range("B12").Value = .Application.WorksheetFunction.MinIfs(finalWorkbook.Worksheets("Sheet1") _
        .Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month1)

        'Maximum
        .Range("A13").Value = "Maximum"
        .Range("B13").Value = .Application.WorksheetFunction.MaxIfs(finalWorkbook.Worksheets("Sheet1") _
        .Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month1)
```



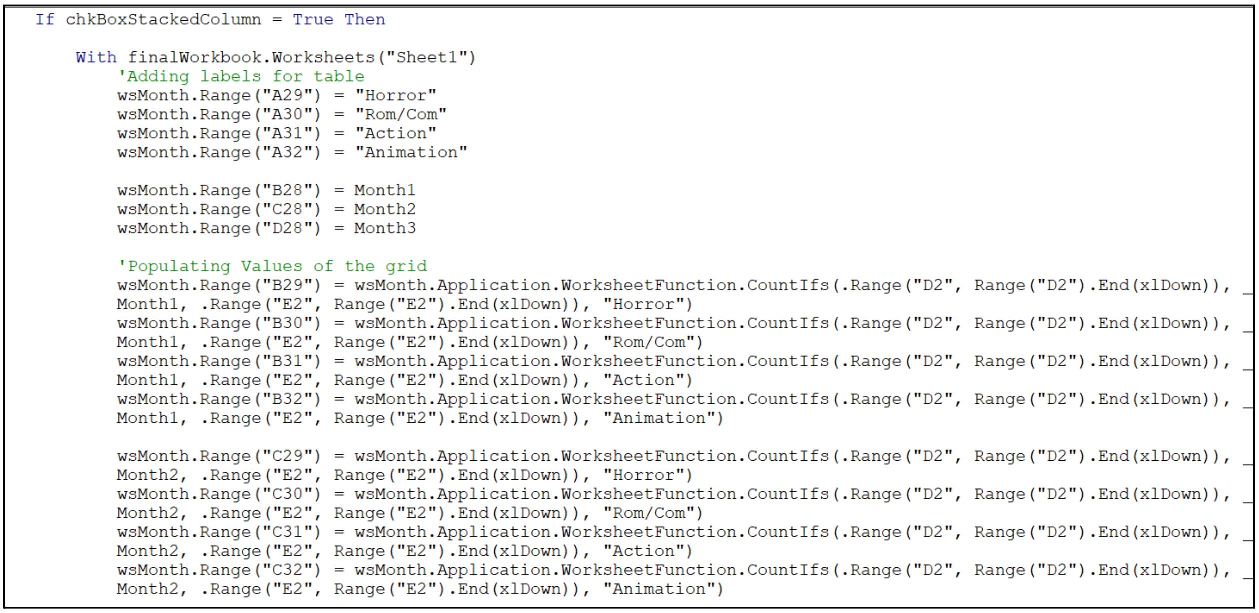
```
'Range of Observed Values
.Range("A14").Value = "Range"
.Range("B14").Value = .Range("B13").Value - .Range("B12").Value



'Populating Values For Month2
.Range("A16").Value = "2)" & Month2

'Mean
.Range("A17").Value = "Mean"
.Range("B17").Value = .Application.WorksheetFunction.AverageIfs(finalWorkbook.Worksheets("Sheet1") _
.Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month2)

'Minimum
.Range("A18").Value = "Minimum"
.Range("B18").Value = .Application.WorksheetFunction.MinIfs(finalWorkbook.Worksheets("Sheet1") _
.Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month2)

'Maximum
.Range("A19").Value = "Maximum"
.Range("B19").Value = .Application.WorksheetFunction.MaxIfs(finalWorkbook.Worksheets("Sheet1") _
.Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("D2", Range("D2").End(xlDown)), Month2)

'Range of Observed Values
.Range("A20").Value = "Range"
.Range("B20").Value = .Range("B19").Value - .Range("B18").Value
```

The above screenshots show the code for the generation of the clustered column chart to represent the frequency table statistics and also the code for the summary statistics table for the three months. Here we make use of AverageIfs, MinIfs, MaxIfs to find the Mean, Minimum and Maximum from a range of cells. These are also part of excel WorksheetFunctions and work similarly to CountIfs. The code for population of Month3 values are not included here, however are present in the final code of our project.

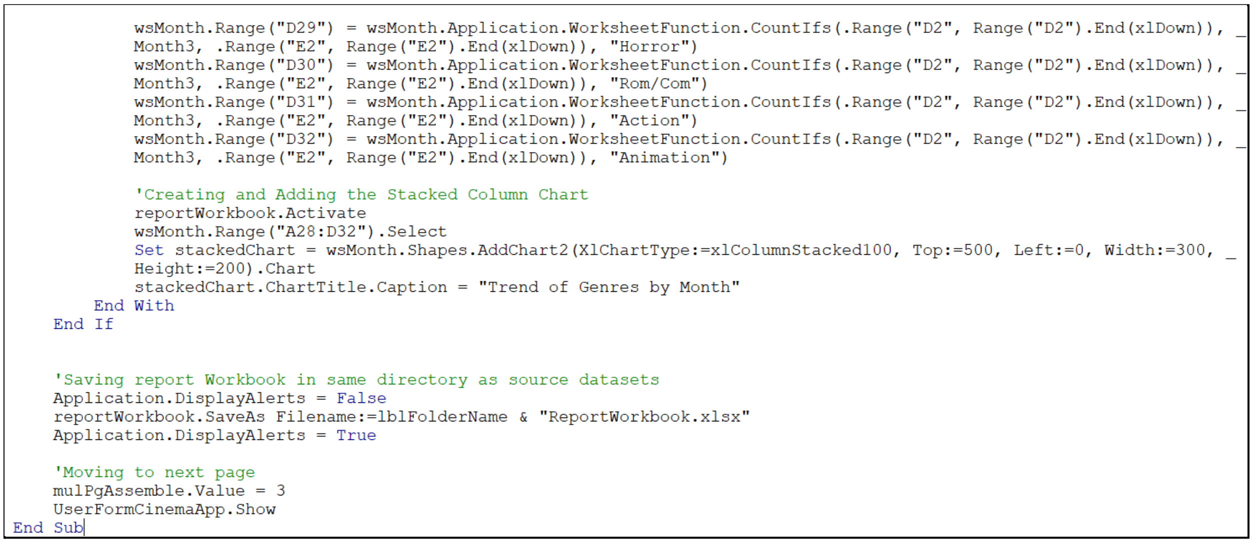


```
    'Creating Chart for Average Box Office Revenue per Movie by Month
    reportWorkbook.Activate
    wsMonth.Application.Union(Range("B11"), Range("B17"), Range("B23")).Select
    Set clustChart = wsMonth.Shapes.AddChart2(XlChartType:=xlColumnClustered, Width:=300, Height:=150, Left:=300, Top:=220).Chart
    clustChart.ChartTitle.Caption = "Average Box Office Revenue per Movie by Month"
    clustChart.ApplyDataLabels Type:=xlDataLabelsShowValue
    clustChart.SeriesCollection(1).XValues = Array(Month1, Month2, Month3)

End If

If chkBoxPieChart.Value = True Then

    'Creating Pie Chart for Monthly Box Office Revenue
    finalWorkbook.Activate
    finalWorkbook.Worksheets("Sheet1").Application.Union(Range("K2", Range("K2").End(xlDown)), Range("D2", Range("D2").End(xlDown))).Select
    Set pieChart = wsMonth.Shapes.AddChart2(XlChartType:=xlPie, Top:=400, Left:=300, Width:=300, Height:=200).Chart
    pieChart.ChartTitle.Caption = "Monthly Box Office Revenue"
    pieChart.ApplyDataLabels Type:=xlDataLabelsShowPercent
    pieChart.SeriesCollection(1).XValues = Array(Month1, Month2, Month3)


End If
```

The above screenshot shows the code for the clustered column chart for the Average Box Office Revenue in each of the three months and also the pie chart for the Box Office Revenue for the three months.



```
If chkBoxStackedColumn = True Then

    With finalWorkbook.Worksheets("Sheet1")
        'Adding labels for table
        wsMonth.Range("A29") = "Horror"
        wsMonth.Range("A30") = "Rom/Com"
        wsMonth.Range("A31") = "Action"
        wsMonth.Range("A32") = "Animation"

        wsMonth.Range("B28") = Month1
        wsMonth.Range("C28") = Month2
        wsMonth.Range("D28") = Month3

        'Populating Values of the grid
        wsMonth.Range("B29") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month1, .Range("E2", Range("E2").End(xlDown)), "Horror")
        wsMonth.Range("B30") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month1, .Range("E2", Range("E2").End(xlDown)), "Rom/Com")
        wsMonth.Range("B31") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month1, .Range("E2", Range("E2").End(xlDown)), "Action")
        wsMonth.Range("B32") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month1, .Range("E2", Range("E2").End(xlDown)), "Animation")

        wsMonth.Range("C29") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month2, .Range("E2", Range("E2").End(xlDown)), "Horror")
        wsMonth.Range("C30") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month2, .Range("E2", Range("E2").End(xlDown)), "Rom/Com")
        wsMonth.Range("C31") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month2, .Range("E2", Range("E2").End(xlDown)), "Action")
        wsMonth.Range("C32") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
        Month2, .Range("E2", Range("E2").End(xlDown)), "Animation")
```

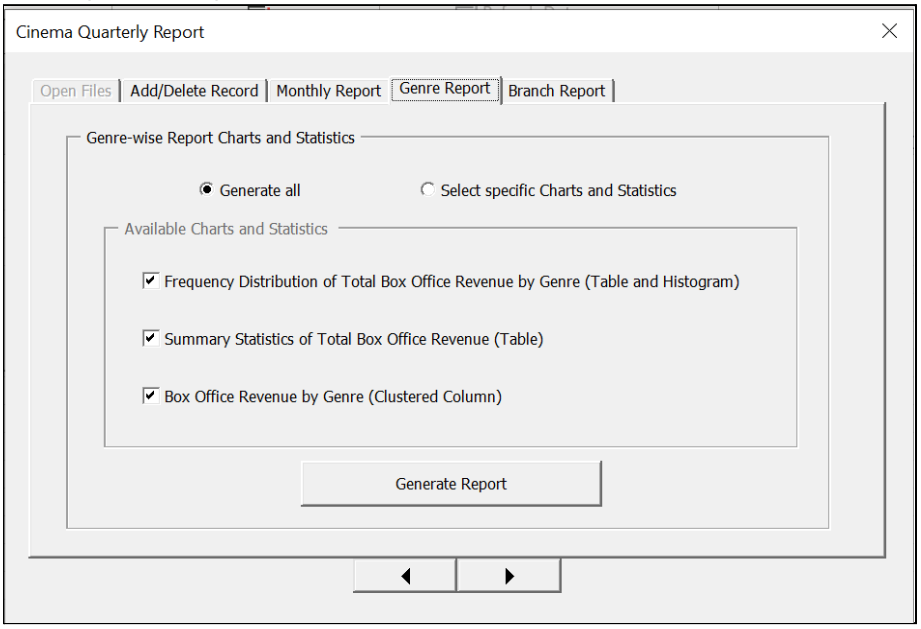


```
            wsMonth.Range("D29") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
            Month3, .Range("E2", Range("E2").End(xlDown)), "Horror")
            wsMonth.Range("D30") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
            Month3, .Range("E2", Range("E2").End(xlDown)), "Rom/Com")
            wsMonth.Range("D31") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
            Month3, .Range("E2", Range("E2").End(xlDown)), "Action")
            wsMonth.Range("D32") = wsMonth.Application.WorksheetFunction.CountIfs(.Range("D2", Range("D2").End(xlDown)), _
            Month3, .Range("E2", Range("E2").End(xlDown)), "Animation")

            'Creating and Adding the Stacked Column Chart
            reportWorkbook.Activate
            wsMonth.Range("A28:D32").Select
            Set stackedChart = wsMonth.Shapes.AddChart2(XlChartType:=xlColumnStacked100, Top:=500, Left:=0, Width:=300, _
            Height:=200).Chart
            stackedChart.ChartTitle.Caption = "Trend of Genres by Month"
        End With
    End If


    'Saving report Workbook in same directory as source datasets
    Application.DisplayAlerts = False
    reportWorkbook.SaveAs Filename:=lblFolderName & "ReportWorkbook.xlsx"
    Application.DisplayAlerts = True

    'Moving to next page
    mulPgAssemble.Value = 3
    UserFormCinemaApp.Show
End Sub
```

The above screenshots show the code for the population and stacked column chart creation of the trend of genres by months. We assume that the genre types are constant throughout the year. Hence we coded them manually.

Once we are done with all the calculations and chart generations we save our workbook and direct the user to the next page in the MultiPage control

2.5 Page “mulPgGenreReport” Controls



Genre-wise Report Charts and Statistics

● Generate all ○ Select specific Charts and Statistics

Available Charts and Statistics

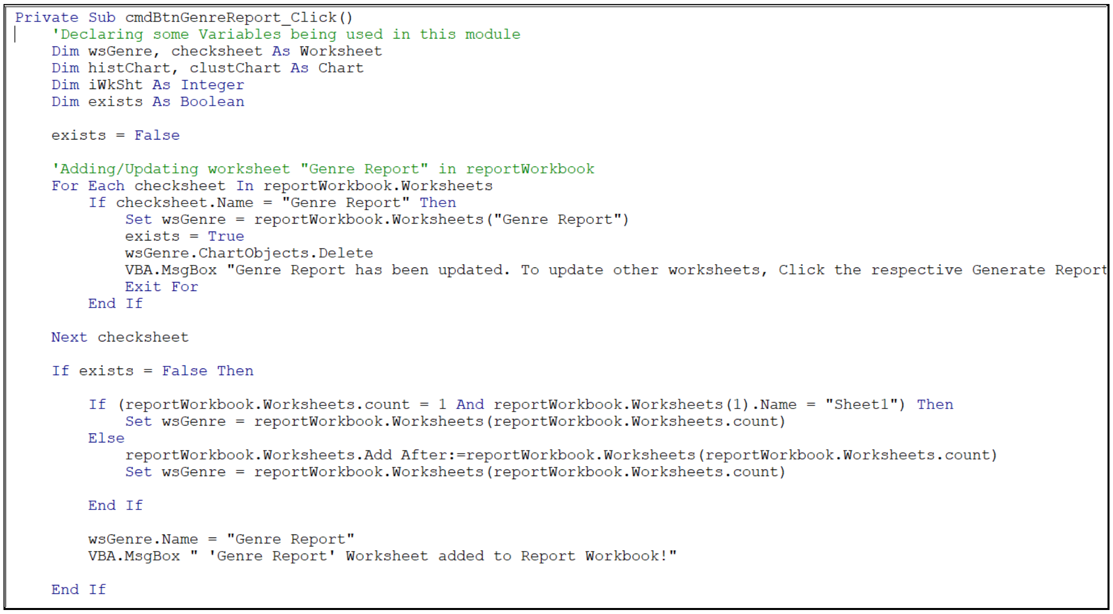
☑ Frequency Distribution of Total Box Office Revenue by Genre (Table and Histogram)

☑ Summary Statistics of Total Box Office Revenue (Table)

☑ Box Office Revenue by Genre (Clustered Column)

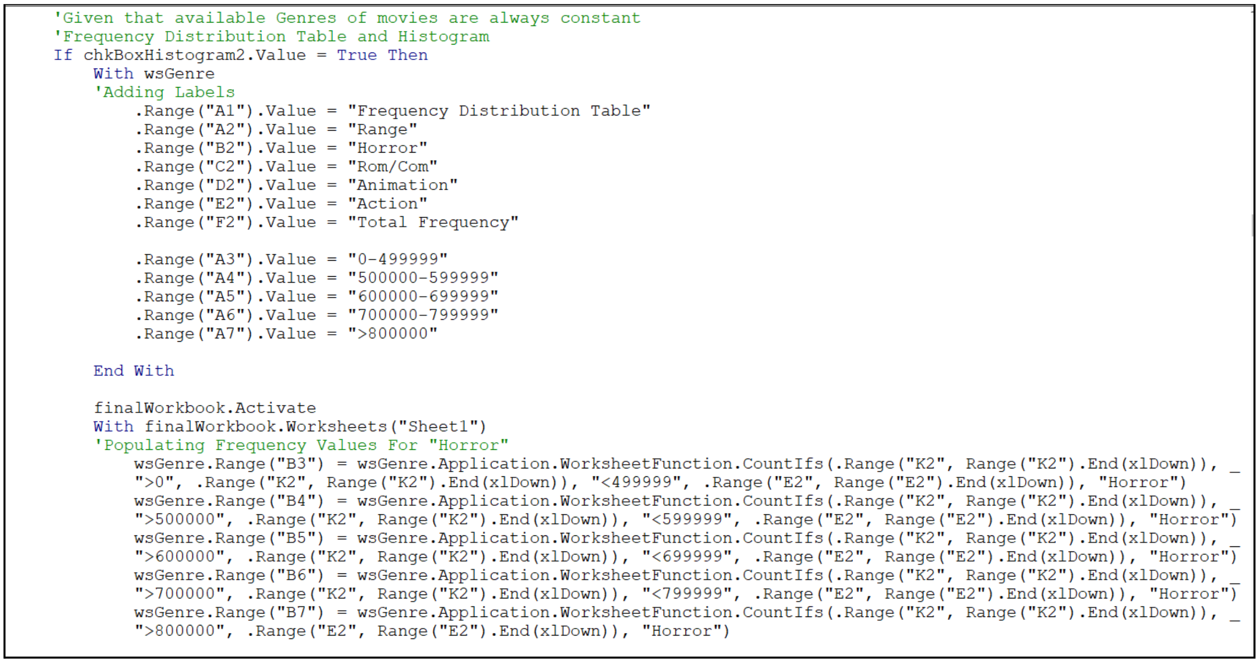
Page “mulPgGenreReport” consists of the Frame Controls “frameGenreReport” which contains two OptionButton Controls “opBtnGenAll2”, “opBtnChooseGen2”, a CommandButton Control “cmdBtnGenreReport” and another frame control “frameChooseGen2” which contains three CheckBox Controls “chkBoxHistogram2”, “chkBoxSummStats2”, “chkBoxClusterColumn”.

Option Button Controls were coded in a similar way to the previous Page. And the below screenshots contain the code for the CommandButton Control “cmdBtnGenreReport”



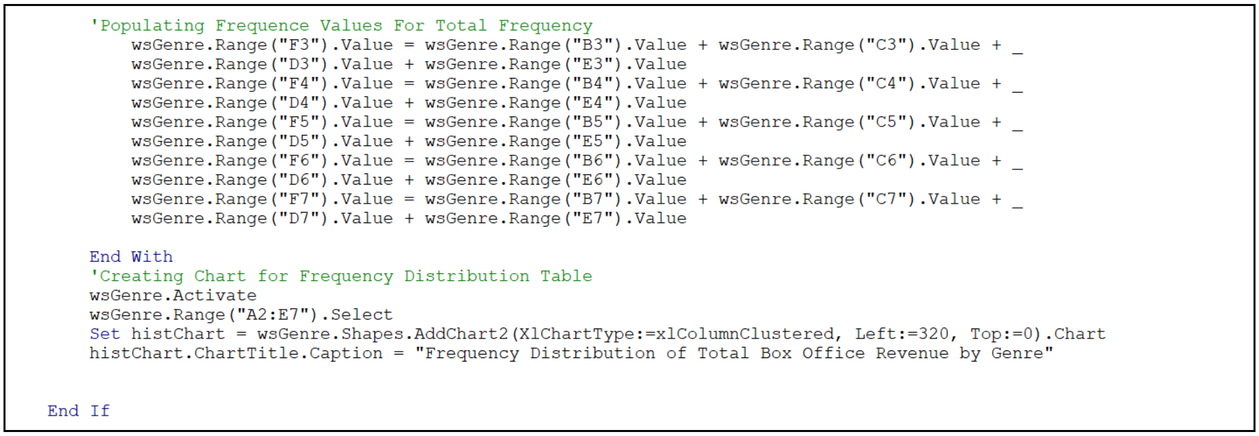
```
Private Sub cmdBtnGenreReport_Click()
    'Declaring some Variables being used in this module
    Dim wsGenre, checksheet As Worksheet
    Dim histChart, clustChart As Chart
    Dim iWkSht As Integer
    Dim exists As Boolean

    exists = False

    'Adding/Updating worksheet "Genre Report" in reportWorkbook
    For Each checksheet In reportWorkbook.Worksheets
        If checksheet.Name = "Genre Report" Then
            Set wsGenre = reportWorkbook.Worksheets("Genre Report")
            exists = True
            wsGenre.ChartObjects.Delete
            VBA.MsgBox "Genre Report has been updated. To update other worksheets, Click the respective Generate Report
            Exit For
        End If

    Next checksheet

    If exists = False Then

        If (reportWorkbook.Worksheets.count = 1 And reportWorkbook.Worksheets(1).Name = "Sheet1") Then
            Set wsGenre = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
        Else
            reportWorkbook.Worksheets.Add After:=reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
            Set wsGenre = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)

        End If

        wsGenre.Name = "Genre Report"
        VBA.MsgBox " 'Genre Report' Worksheet added to Report Workbook!"

    End If
```

Used variables were declared at a procedure level with appropriate data types. It adds the new Worksheet “Genre Report” to the previously opened reportWorkbook in a similar manner and technique described for the Monthly Report in the “mulPgMonthlyReport” section of this report.



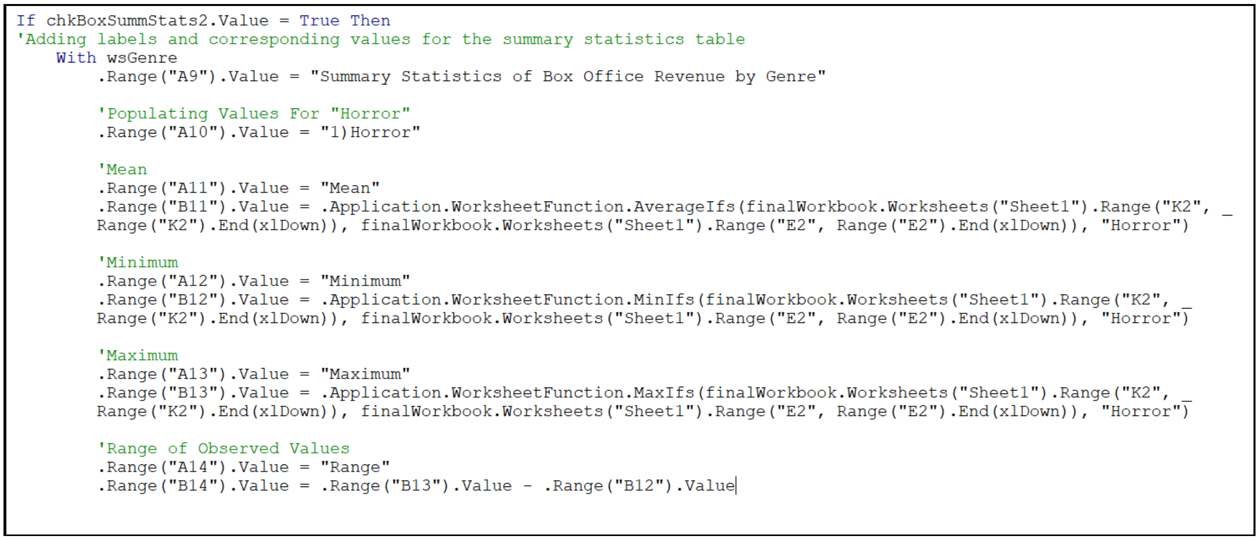
```
    'Given that available Genres of movies are always constant
    'Frequency Distribution Table and Histogram
    If chkBoxHistogram2.Value = True Then
        With wsGenre
        'Adding Labels
            .Range("A1").Value = "Frequency Distribution Table"
            .Range("A2").Value = "Range"
            .Range("B2").Value = "Horror"
            .Range("C2").Value = "Rom/Com"
            .Range("D2").Value = "Animation"
            .Range("E2").Value = "Action"
            .Range("F2").Value = "Total Frequency"

            .Range("A3").Value = "0-499999"
            .Range("A4").Value = "500000-599999"
            .Range("A5").Value = "600000-699999"
            .Range("A6").Value = "700000-799999"
            .Range("A7").Value = ">800000"

        End With

        finalWorkbook.Activate
        With finalWorkbook.Worksheets("Sheet1")
        'Populating Frequency Values For "Horror"
            wsGenre.Range("B3") = wsGenre.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
            ">0", .Range("K2", Range("K2").End(xlDown)), "<499999", .Range("E2", Range("E2").End(xlDown)), "Horror")
            wsGenre.Range("B4") = wsGenre.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
            ">500000", .Range("K2", Range("K2").End(xlDown)), "<599999", .Range("E2", Range("E2").End(xlDown)), "Horror")
            wsGenre.Range("B5") = wsGenre.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
            ">600000", .Range("K2", Range("K2").End(xlDown)), "<699999", .Range("E2", Range("E2").End(xlDown)), "Horror")
            wsGenre.Range("B6") = wsGenre.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
            ">700000", .Range("K2", Range("K2").End(xlDown)), "<799999", .Range("E2", Range("E2").End(xlDown)), "Horror")
            wsGenre.Range("B7") = wsGenre.Application.WorksheetFunction.CountIfs(.Range("K2", Range("K2").End(xlDown)), _
            ">800000", .Range("E2", Range("E2").End(xlDown)), "Horror")
```

The If statement is used to see if the user wants to report a Frequency Distribution of Total Box Office Revenue by Genre (Table and Histogram) and clicks the corresponding CheckBox Control. If yes, the Program computes the frequency values for every genre and total frequency at the end. This is similarly seen in cmdBtnMonthlyReport_Click() and cmdButtonBranchReport_Click() subs of the code. We make use of CountIfs again to calculate the frequency for different levels of Genre and total as well.



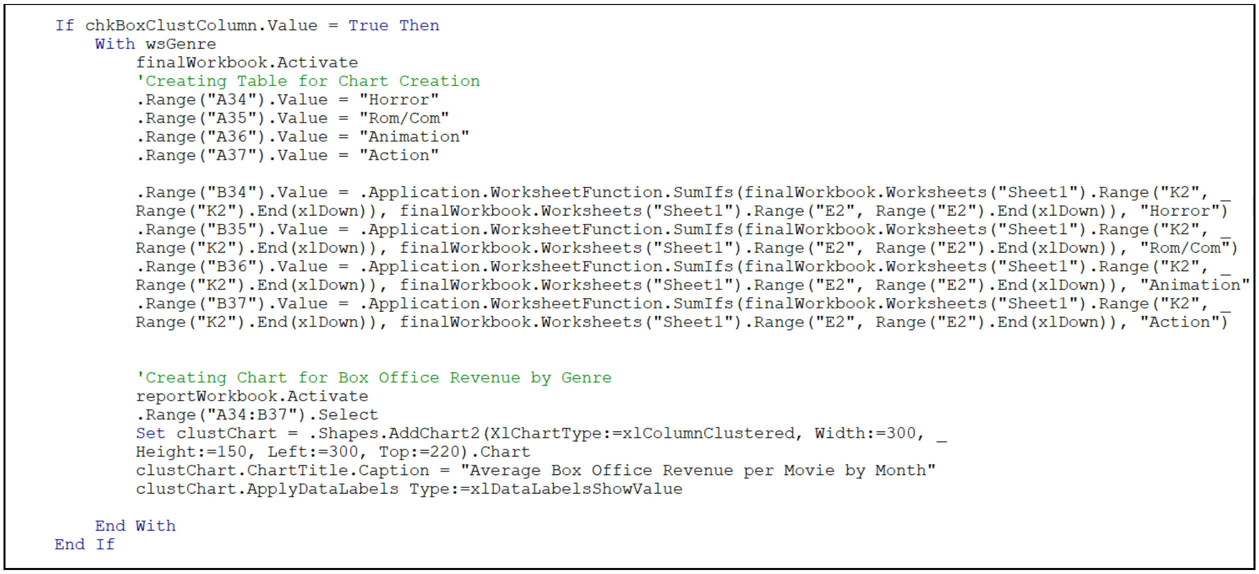
```
        'Populating Frequence Values For Total Frequency
            wsGenre.Range("F3").Value = wsGenre.Range("B3").Value + wsGenre.Range("C3").Value + _
            wsGenre.Range("D3").Value + wsGenre.Range("E3").Value
            wsGenre.Range("F4").Value = wsGenre.Range("B4").Value + wsGenre.Range("C4").Value + _
            wsGenre.Range("D4").Value + wsGenre.Range("E4").Value
            wsGenre.Range("F5").Value = wsGenre.Range("B5").Value + wsGenre.Range("C5").Value + _
            wsGenre.Range("D5").Value + wsGenre.Range("E5").Value
            wsGenre.Range("F6").Value = wsGenre.Range("B6").Value + wsGenre.Range("C6").Value + _
            wsGenre.Range("D6").Value + wsGenre.Range("E6").Value
            wsGenre.Range("F7").Value = wsGenre.Range("B7").Value + wsGenre.Range("C7").Value + _
            wsGenre.Range("D7").Value + wsGenre.Range("E7").Value

        End With
        'Creating Chart for Frequency Distribution Table
        wsGenre.Activate
        wsGenre.Range("A2:E7").Select
        Set histChart = wsGenre.Shapes.AddChart2(XlChartType:=xlColumnClustered, Left:=320, Top:=0).Chart
        histChart.ChartTitle.Caption = "Frequency Distribution of Total Box Office Revenue by Genre"


    End If
```

After computing the values, it creates a chart for the frequency distribution of Total Box Office Revenue by Genre.

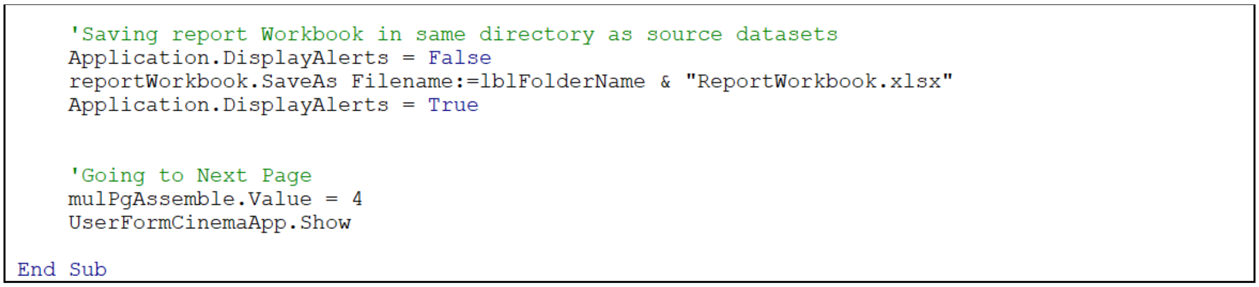


```
If chkBoxSummStats2.Value = True Then
'Adding labels and corresponding values for the summary statistics table
    With wsGenre
        .Range("A9").Value = "Summary Statistics of Box Office Revenue by Genre"

        'Populating Values For "Horror"
        .Range("A10").Value = "1)Horror"

        'Mean
        .Range("A11").Value = "Mean"
        .Range("B11").Value = .Application.WorksheetFunction.AverageIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
        Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Horror")

        'Minimum
        .Range("A12").Value = "Minimum"
        .Range("B12").Value = .Application.WorksheetFunction.MinIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
        Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Horror")

        'Maximum
        .Range("A13").Value = "Maximum"
        .Range("B13").Value = .Application.WorksheetFunction.MaxIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
        Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Horror")

        'Range of Observed Values
        .Range("A14").Value = "Range"
        .Range("B14").Value = .Range("B13").Value - .Range("B12").Value
```

Next, it creates a summary statistics table for Total Box Office Revenue by genre. It populates values of mean, maximum, minimum and value of observed values for every genre in the same worksheet of “Genre Report.” The code for the “Horror” Genre can be seen above



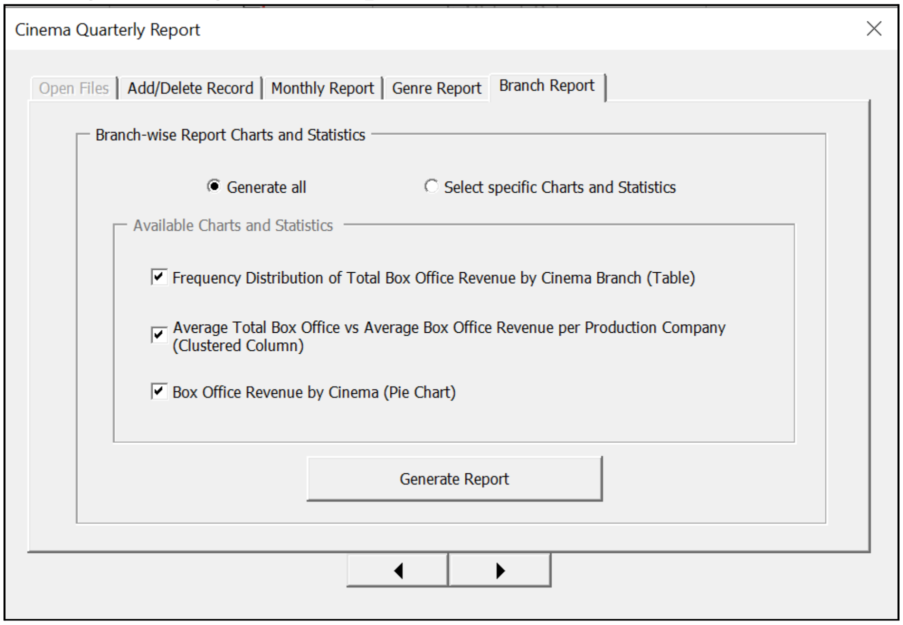
```
    If chkBoxClustColumn.Value = True Then
        With wsGenre
            finalWorkbook.Activate
            'Creating Table for Chart Creation
            .Range("A34").Value = "Horror"
            .Range("A35").Value = "Rom/Com"
            .Range("A36").Value = "Animation"
            .Range("A37").Value = "Action"

            .Range("B34").Value = .Application.WorksheetFunction.SumIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
            Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Horror")
            .Range("B35").Value = .Application.WorksheetFunction.SumIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
            Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Rom/Com")
            .Range("B36").Value = .Application.WorksheetFunction.SumIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
            Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Animation"
            .Range("B37").Value = .Application.WorksheetFunction.SumIfs(finalWorkbook.Worksheets("Sheet1").Range("K2", _
            Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1").Range("E2", Range("E2").End(xlDown)), "Action")


            'Creating Chart for Box Office Revenue by Genre
            reportWorkbook.Activate
            .Range("A34:B37").Select
            Set clustChart = .Shapes.AddChart2(XlChartType:=xlColumnClustered, Width:=300, _
            Height:=150, Left:=300, Top:=220).Chart
            clustChart.ChartTitle.Caption = "Average Box Office Revenue per Movie by Month"
            clustChart.ApplyDataLabels Type:=xlDataLabelsShowValue

        End With
    End If
```



```
    'Saving report Workbook in same directory as source datasets
    Application.DisplayAlerts = False
    reportWorkbook.SaveAs Filename:=lblFolderName & "ReportWorkbook.xlsx"
    Application.DisplayAlerts = True


    'Going to Next Page
    mulPgAssemble.Value = 4
    UserFormCinemaApp.Show

End Sub
```

It also created a clustered column for the Box Office Revenue by genre. After the report for this Page has been generated, we save the workbook and direct the user to the next Page for the Branch Report automatically.

2.6 Page "mulPgBranchReport" Controls



● Generate all ○ Select specific Charts and Statistics

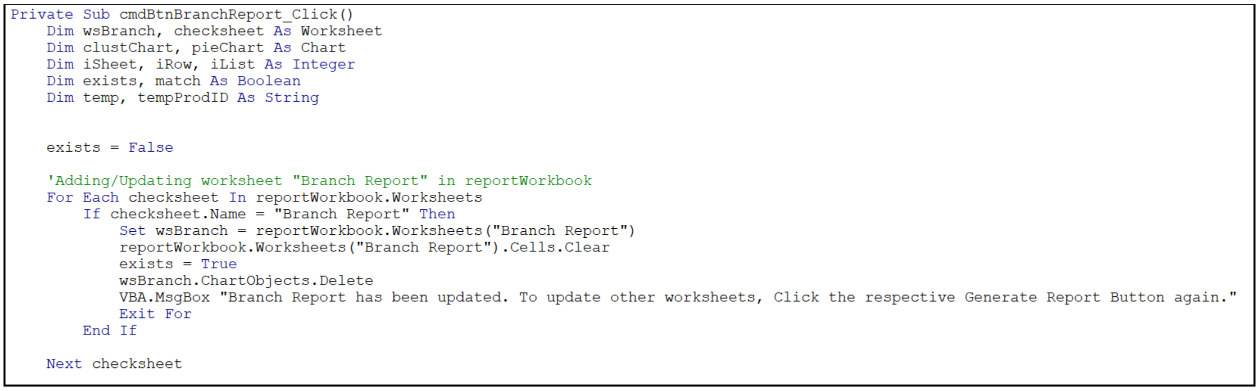
☑ Frequency Distribution of Total Box Office Revenue by Cinema Branch (Table)

☑ Average Total Box Office vs Average Box Office Revenue per Production Company (Clustered Column)

☑ Box Office Revenue by Cinema (Pie Chart)

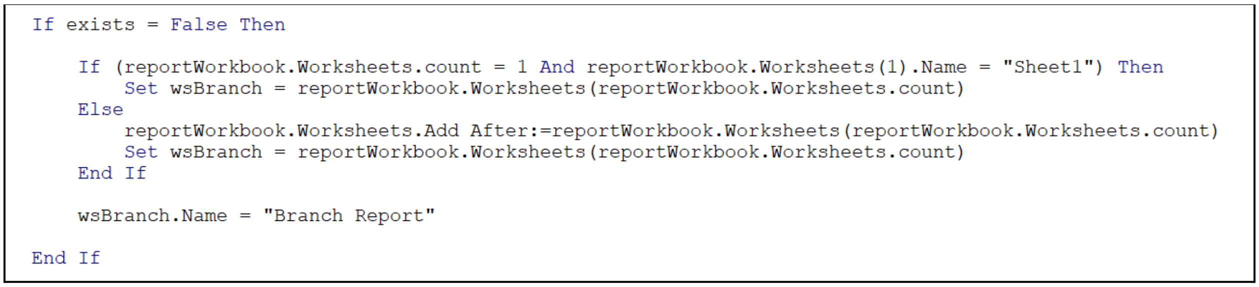
Page “mulPgBranchReport” consists of the Frame Control “frameBranchReport” which includes two OptionButton Controls “opBtnGenAll3”, “opBtnChooseGen3 a CommandButton Control “cmdBtnBranchReport and another Frame Control “frameChooseGen3” which includes three CheckBox Controls “chkBoxFreqDist”, “chkBoxClustColumn2”, “chkBoxPieChart2”.

Option Button Controls were coded the same way as for the previous Pages. And the below screenshots contain the code for the CommandButton Control “cmdBtnBranchReport”



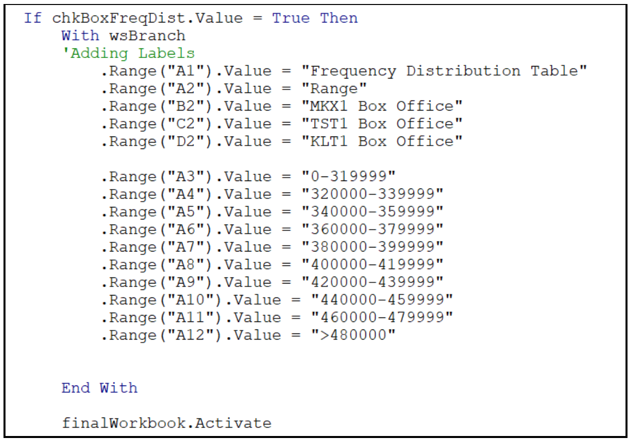
```
Private Sub cmdBtnBranchReport_Click()
    Dim wsBranch, checksheet As Worksheet
    Dim clustChart, pieChart As Chart
    Dim iSheet, iRow, iList As Integer
    Dim exists, match As Boolean
    Dim temp, tempProdID As String


    exists = False

    'Adding/Updating worksheet "Branch Report" in reportWorkbook
    For Each checksheet In reportWorkbook.Worksheets
        If checksheet.Name = "Branch Report" Then
            Set wsBranch = reportWorkbook.Worksheets("Branch Report")
            reportWorkbook.Worksheets("Branch Report").Cells.Clear
            exists = True
            wsBranch.ChartObjects.Delete
            VBA.MsgBox "Branch Report has been updated. To update other worksheets, Click the respective Generate Report Button again."
            Exit For
        End If

    Next checksheet
```

Used variables were declared at a procedure level with appropriate data types. If there is a worksheet named “Branch Report”, it updates the current “Branch Report” Worksheet with the new information.

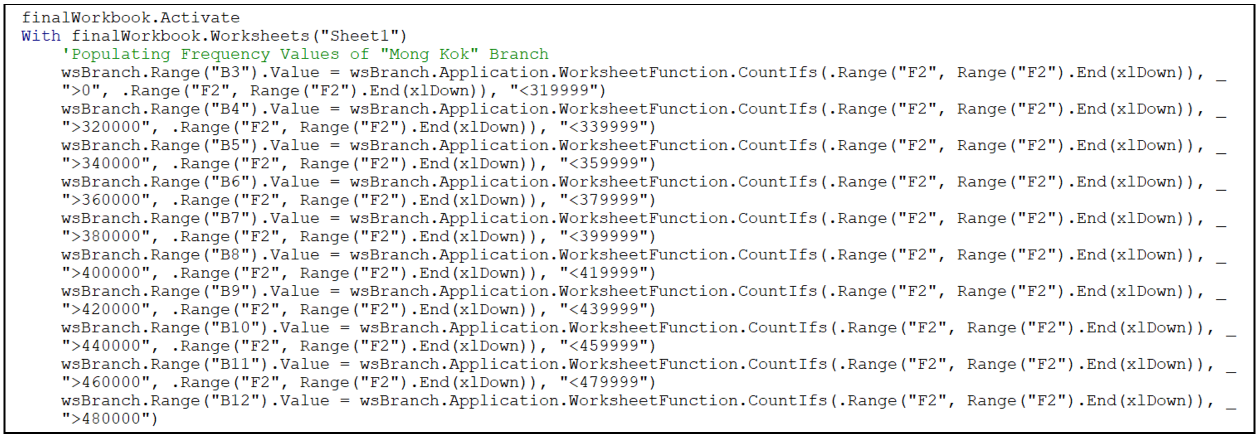


```
If exists = False Then

    If (reportWorkbook.Worksheets.count = 1 And reportWorkbook.Worksheets(1).Name = "Sheet1") Then
        Set wsBranch = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
    Else
        reportWorkbook.Worksheets.Add After:=reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
        Set wsBranch = reportWorkbook.Worksheets(reportWorkbook.Worksheets.count)
    End If

    wsBranch.Name = "Branch Report"

End If
```

Otherwise, it adds the new Worksheet “Branch Report” to the previously opened Report Workbook in a similar manner and technique described for the Monthly Report in the “mulPgMonthlyReport” section of this report.

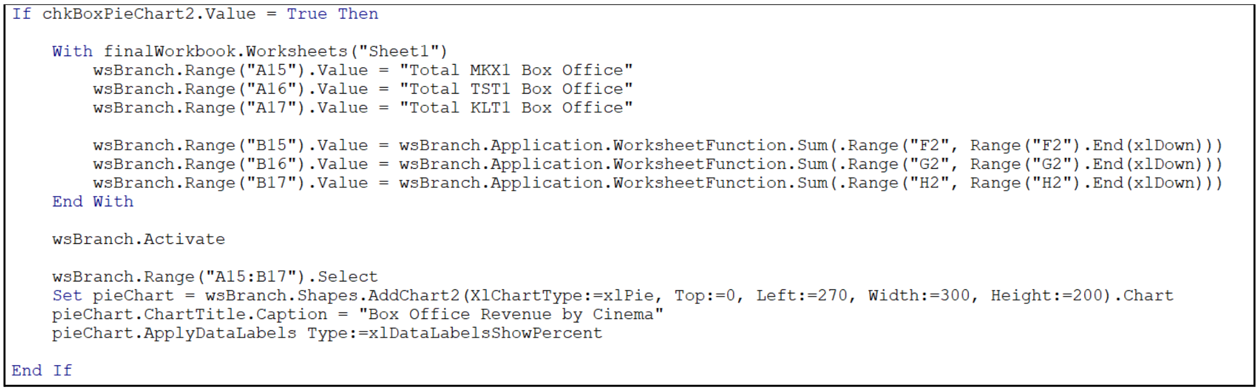


```
If chkBoxFreqDist.Value = True Then
    With wsBranch
    'Adding Labels
        .Range("A1").Value = "Frequency Distribution Table"
        .Range("A2").Value = "Range"
        .Range("B2").Value = "MKX1 Box Office"
        .Range("C2").Value = "TST1 Box Office"
        .Range("D2").Value = "KLT1 Box Office"

        .Range("A3").Value = "0-319999"
        .Range("A4").Value = "320000-339999"
        .Range("A5").Value = "340000-359999"
        .Range("A6").Value = "360000-379999"
        .Range("A7").Value = "380000-399999"
        .Range("A8").Value = "400000-419999"
        .Range("A9").Value = "420000-439999"
        .Range("A10").Value = "440000-459999"
        .Range("A11").Value = "460000-479999"
        .Range("A12").Value = ">480000"


    End With

    finalWorkbook.Activate
```



```
finalWorkbook.Activate
With finalWorkbook.Worksheets("Sheet1")
    'Populating Frequency Values of "Mong Kok" Branch
    wsBranch.Range("B3").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">0", .Range("F2", Range("F2").End(xlDown)), "<319999")
    wsBranch.Range("B4").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">320000", .Range("F2", Range("F2").End(xlDown)), "<339999")
    wsBranch.Range("B5").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">340000", .Range("F2", Range("F2").End(xlDown)), "<359999")
    wsBranch.Range("B6").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">360000", .Range("F2", Range("F2").End(xlDown)), "<379999")
    wsBranch.Range("B7").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">380000", .Range("F2", Range("F2").End(xlDown)), "<399999")
    wsBranch.Range("B8").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">400000", .Range("F2", Range("F2").End(xlDown)), "<419999")
    wsBranch.Range("B9").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">420000", .Range("F2", Range("F2").End(xlDown)), "<439999")
    wsBranch.Range("B10").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">440000", .Range("F2", Range("F2").End(xlDown)), "<459999")
    wsBranch.Range("B11").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">460000", .Range("F2", Range("F2").End(xlDown)), "<479999")
    wsBranch.Range("B12").Value = wsBranch.Application.WorksheetFunction.CountIfs(.Range("F2", Range("F2").End(xlDown)), _
    ">480000")
```

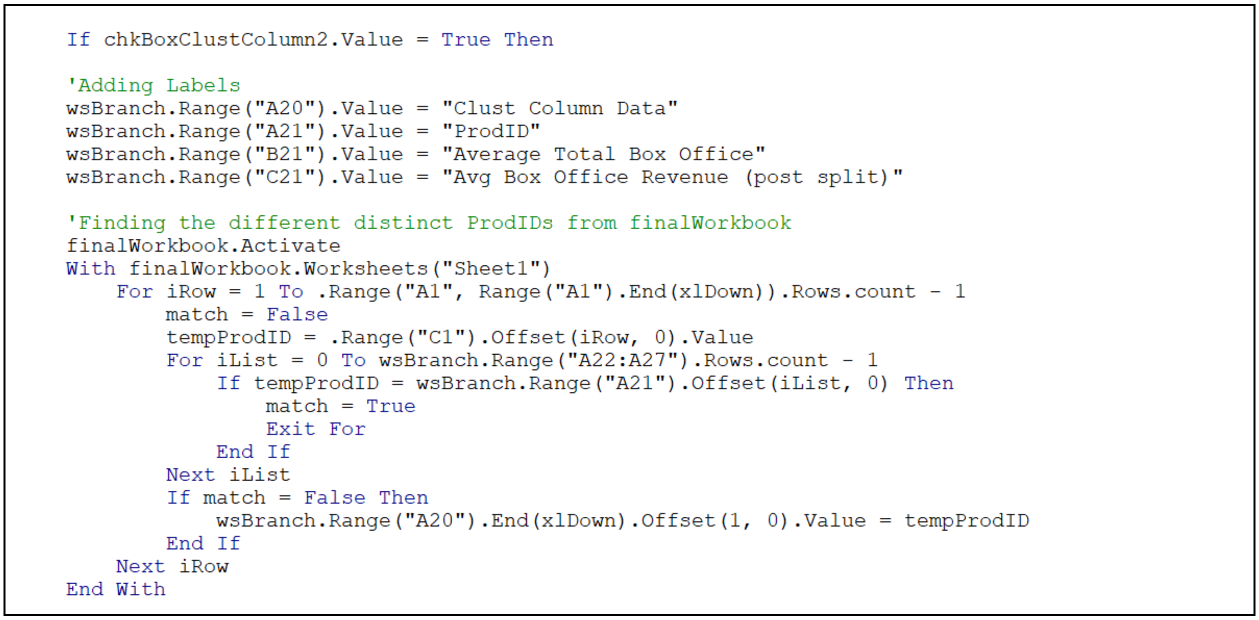
The If statement is used to see if the user wants to report a Frequency Distribution of Total Box Office Revenue by Branch (Table) and clicks the corresponding CheckBox Control. If yes, the Program computes the frequency values for every Branch.



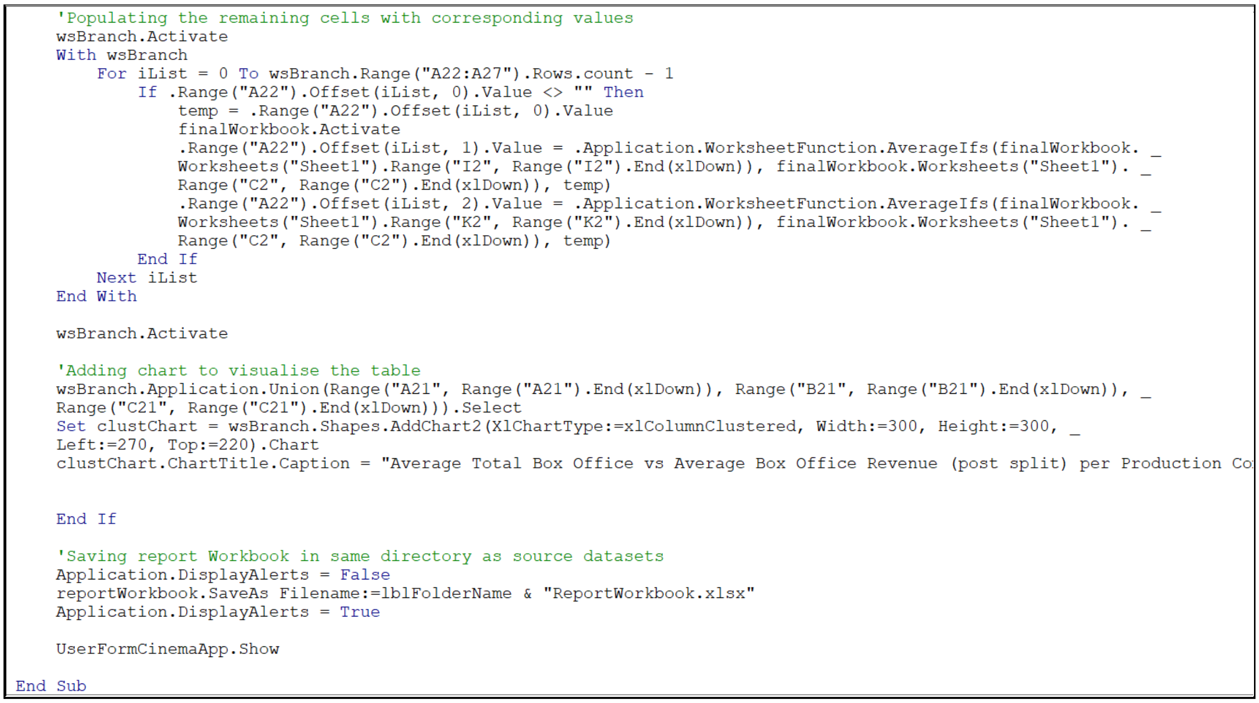
```
If chkBoxPieChart2.Value = True Then

    With finalWorkbook.Worksheets("Sheet1")
        wsBranch.Range("A15").Value = "Total MKX1 Box Office"
        wsBranch.Range("A16").Value = "Total TST1 Box Office"
        wsBranch.Range("A17").Value = "Total KLT1 Box Office"

        wsBranch.Range("B15").Value = wsBranch.Application.WorksheetFunction.Sum(.Range("F2", Range("F2").End(xlDown)))
        wsBranch.Range("B16").Value = wsBranch.Application.WorksheetFunction.Sum(.Range("G2", Range("G2").End(xlDown)))
        wsBranch.Range("B17").Value = wsBranch.Application.WorksheetFunction.Sum(.Range("H2", Range("H2").End(xlDown)))
    End With

    wsBranch.Activate

    wsBranch.Range("A15:B17").Select
    Set pieChart = wsBranch.Shapes.AddChart2(XlChartType:=xlPie, Top:=0, Left:=270, Width:=300, Height:=200).Chart
    pieChart.ChartTitle.Caption = "Box Office Revenue by Cinema"
    pieChart.ApplyDataLabels Type:=xlDataLabelsShowPercent

End If
```

The above screenshot shows how we have created a pie chart for Box Office Revenue by Cinema

Lastly, the screenshots below show how we generate the Average Total Box Office and Average Box Office Revenue per Production Company chart in the same worksheet of “Branch Report” i.e. revenue pre vs revenue post removal of production companies’ share. There are in total six production companies in *Production Companies.xlsx*, however the companies with ProdID “SNY1” and “VBP1” have no movies releasing this quarter. That being said, one can add a record using our “mulPgAddDel” page. Hence we have programmed the chart generation in a way that allows the inclusion of such records if required. Upon updating, one can notice the changes in the charts in the report. We then save the workbook



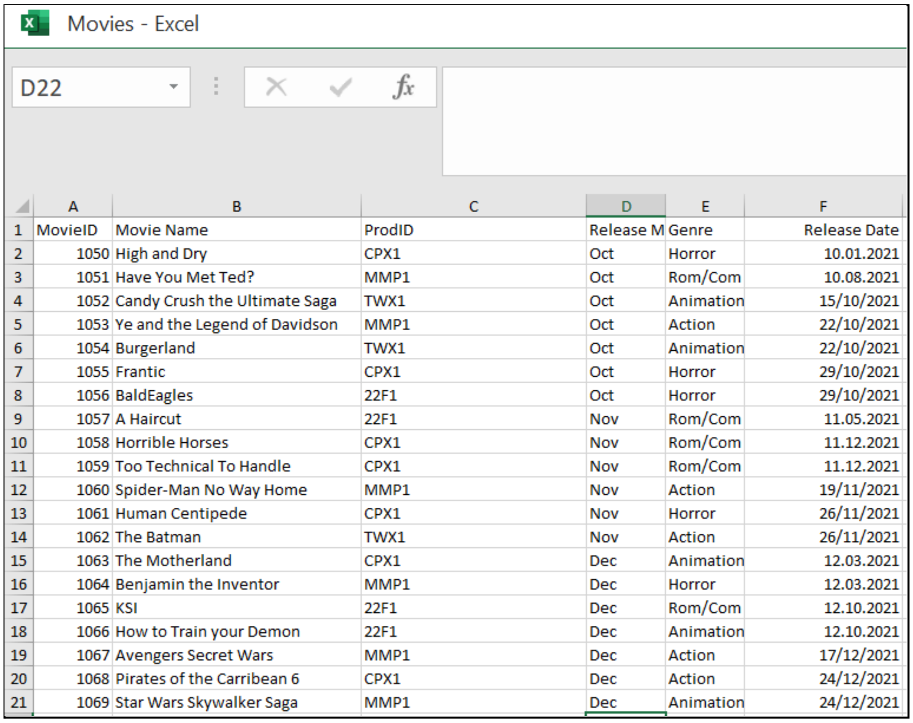
```
    If chkBoxClustColumn2.Value = True Then

    'Adding Labels
    wsBranch.Range("A20").Value = "Clust Column Data"
    wsBranch.Range("A21").Value = "ProdID"
    wsBranch.Range("B21").Value = "Average Total Box Office"
    wsBranch.Range("C21").Value = "Avg Box Office Revenue (post split)"

    'Finding the different distinct ProdIDs from finalWorkbook
    finalWorkbook.Activate
    With finalWorkbook.Worksheets("Sheet1")
        For iRow = 1 To .Range("A1", Range("A1").End(xlDown)).Rows.count - 1
            match = False
            tempProdID = .Range("C1").Offset(iRow, 0).Value
            For iList = 0 To wsBranch.Range("A22:A27").Rows.count - 1
                If tempProdID = wsBranch.Range("A21").Offset(iList, 0) Then
                    match = True
                    Exit For
                End If
            Next iList
            If match = False Then
                wsBranch.Range("A20").End(xlDown).Offset(1, 0).Value = tempProdID
            End If
        Next iRow
    End With
```



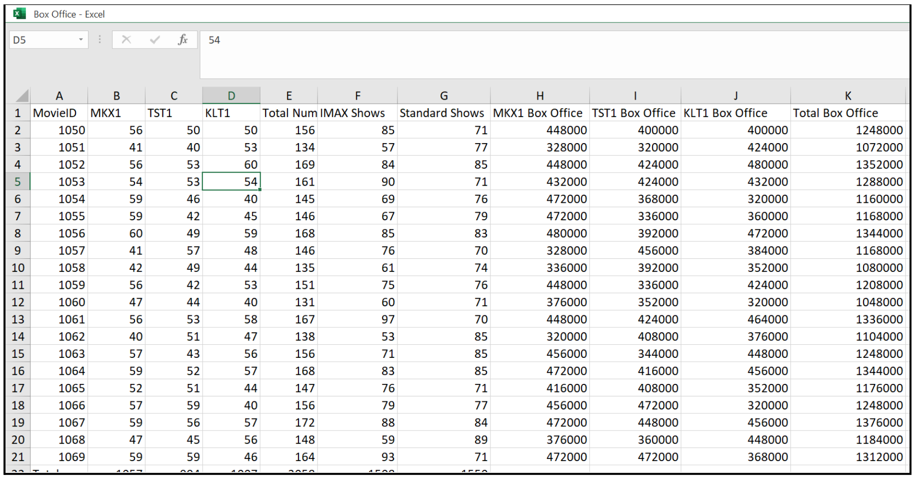
```
    'Populating the remaining cells with corresponding values
    wsBranch.Activate
    With wsBranch
        For iList = 0 To wsBranch.Range("A22:A27").Rows.count - 1
            If .Range("A22").Offset(iList, 0).Value <> "" Then
                temp = .Range("A22").Offset(iList, 0).Value
                finalWorkbook.Activate
                .Range("A22").Offset(iList, 1).Value = .Application.WorksheetFunction.AverageIfs(finalWorkbook. _
                Worksheets("Sheet1").Range("I2", Range("I2").End(xlDown)), finalWorkbook.Worksheets("Sheet1"). _
                Range("C2", Range("C2").End(xlDown)), temp)
                .Range("A22").Offset(iList, 2).Value = .Application.WorksheetFunction.AverageIfs(finalWorkbook. _
                Worksheets("Sheet1").Range("K2", Range("K2").End(xlDown)), finalWorkbook.Worksheets("Sheet1"). _
                Range("C2", Range("C2").End(xlDown)), temp)
            End If
        Next iList
    End With

    wsBranch.Activate

    'Adding chart to visualise the table
    wsBranch.Application.Union(Range("A21", Range("A21").End(xlDown)), Range("B21", Range("B21").End(xlDown)), _
    Range("C21", Range("C21").End(xlDown))).Select
    Set clustChart = wsBranch.Shapes.AddChart2(XlChartType:=xlColumnClustered, Width:=300, Height:=300, _
    Left:=270, Top:=220).Chart
    clustChart.ChartTitle.Caption = "Average Total Box Office vs Average Box Office Revenue (post split) per Production Co


    End If

    'Saving report Workbook in same directory as source datasets
    Application.DisplayAlerts = False
    reportWorkbook.SaveAs Filename:=lblFolderName & "ReportWorkbook.xlsx"
    Application.DisplayAlerts = True

    UserFormCinemaApp.Show

End Sub
```

3. Workbooks and Worksheets Objects

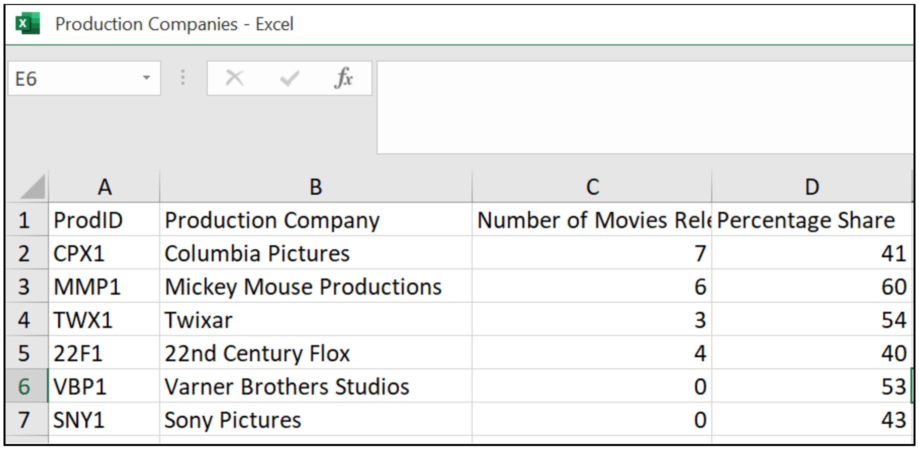
Final Workbook and the Reports are generated using the following four Workbooks, uploaded by the User: *Movies.xlsx, Box Office.xlsx, Production Companies.xlsx, and Cinema Info.xlsx*. The screenshots are shown below.



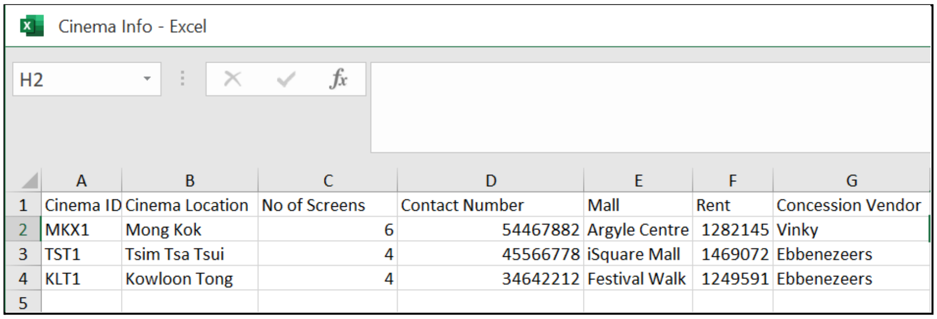
	A	B	C	D	E	F
1	MovieID	Movie Name	ProdID	Release M	Genre	Release Date
2	1050	High and Dry	CPX1	Oct	Horror	10.01.2021
3	1051	Have You Met Ted?	MMP1	Oct	Rom/Com	10.08.2021
4	1052	Candy Crush the Ultimate Saga	TWX1	Oct	Animation	15/10/2021
5	1053	Ye and the Legend of Davidson	MMP1	Oct	Action	22/10/2021
6	1054	Burgerland	TWX1	Oct	Animation	22/10/2021
7	1055	Frantic	CPX1	Oct	Horror	29/10/2021
8	1056	BaldEagles	22F1	Oct	Horror	29/10/2021
9	1057	A Haircut	22F1	Nov	Rom/Com	11.05.2021
10	1058	Horrible Horses	CPX1	Nov	Rom/Com	11.12.2021
11	1059	Too Technical To Handle	CPX1	Nov	Rom/Com	11.12.2021
12	1060	Spider-Man No Way Home	MMP1	Nov	Action	19/11/2021
13	1061	Human Centipede	CPX1	Nov	Horror	26/11/2021
14	1062	The Batman	TWX1	Nov	Action	26/11/2021
15	1063	The Motherland	CPX1	Dec	Animation	12.03.2021
16	1064	Benjamin the Inventor	MMP1	Dec	Horror	12.03.2021
17	1065	KSI	22F1	Dec	Rom/Com	12.10.2021
18	1066	How to Train your Demon	22F1	Dec	Animation	12.10.2021
19	1067	Avengers Secret Wars	MMP1	Dec	Action	17/12/2021
20	1068	Pirates of the Carribean 6	CPX1	Dec	Action	24/12/2021
21	1069	Star Wars Skywalker Saga	MMP1	Dec	Animation	24/12/2021



	A	B	C	D	E	F	G	H	I	J	K
1	MovieID	MKX1	TST1	KLT1	Total Num	IMAX Shows	Standard Shows	MKX1 Box Office	TST1 Box Office	KLT1 Box Office	Total Box Office
2	1050	56	50	50	156	85	71	448000	400000	400000	1248000
3	1051	41	40	53	134	57	77	328000	320000	424000	1072000
4	1052	56	53	60	169	84	85	448000	424000	480000	1352000
5	1053	54	53	54	161	90	71	432000	424000	432000	1288000
6	1054	59	46	40	145	69	76	472000	368000	320000	1160000
7	1055	59	42	45	146	67	79	472000	336000	360000	1168000
8	1056	60	49	59	168	85	83	480000	392000	472000	1344000
9	1057	41	57	48	146	76	70	328000	456000	384000	1168000
10	1058	42	49	44	135	61	74	336000	392000	352000	1080000
11	1059	56	42	53	151	75	76	448000	336000	424000	1208000
12	1060	47	44	40	131	60	71	376000	352000	320000	1048000
13	1061	56	53	58	167	97	70	448000	424000	464000	1336000
14	1062	40	51	47	138	53	85	320000	408000	376000	1104000
15	1063	57	43	56	156	71	85	456000	344000	448000	1248000
16	1064	59	52	57	168	83	85	472000	416000	456000	1344000
17	1065	52	51	44	147	76	71	416000	408000	352000	1176000
18	1066	57	59	40	156	79	77	456000	472000	320000	1248000
19	1067	59	56	57	172	88	84	472000	448000	456000	1376000
20	1068	47	45	56	148	59	89	376000	360000	448000	1184000
21	1069	59	59	46	164	93	71	472000	472000	368000	1312000

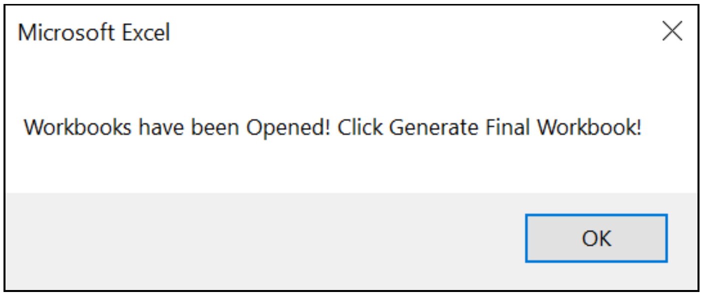


	A	B	C	D
1	ProdID	Production Company	Number of Movies Rel	Percentage Share
2	CPX1	Columbia Pictures	7	41
3	MMP1	Mickey Mouse Productions	6	60
4	TWX1	Twixar	3	54
5	22F1	22nd Century Flox	4	40
6	VBP1	Varner Brothers Studios	0	53
7	SNY1	Sony Pictures	0	43



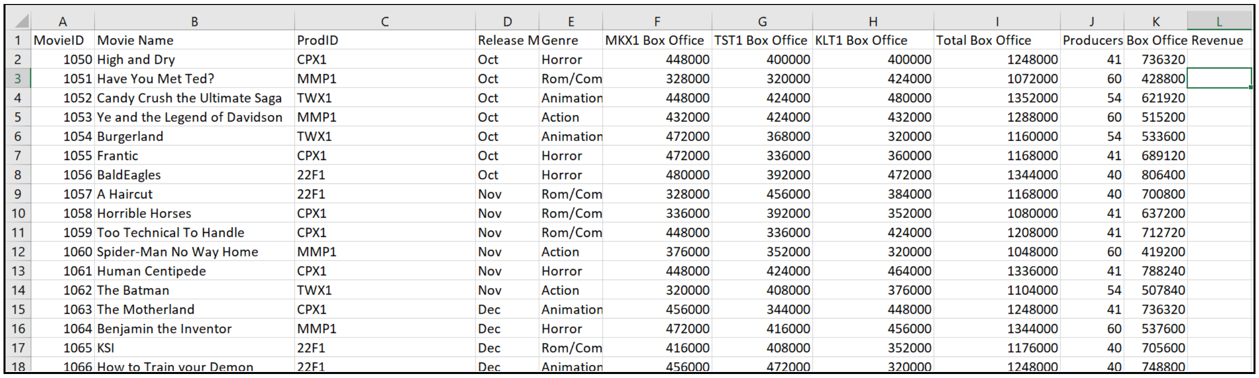
	A	B	C	D	E	F	G
1	Cinema ID	Cinema Location	No of Screens	Contact Number	Mall	Rent	Concession Vendor
2	MKX1	Mong Kok	6	54467882	Argyle Centre	1282145	Vinky
3	TST1	Tsim Tsa Tsui	4	45566778	iSquare Mall	1469072	Ebbenezeers
4	KLT1	Kowloon Tong	4	34642212	Festival Walk	1249591	Ebbenezeers
5							

When all four Workbooks are uploaded and the button "Open Workbooks" is clicked, the Workbooks are Opened programmatically one by one and the following VBA Message Dialogue appears:



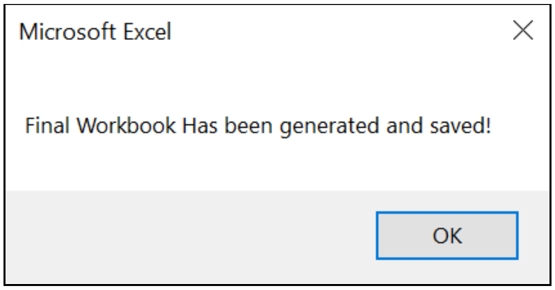
Workbooks have been Opened! Click Generate Final Workbook!

Final Workbook consists of the all movies information with respective column variables: MovieID, Movie Name, Production Company ID, Release Month, Revenue generated by three branches-Mongkok, TST, and Kowloon Tong (computed variables), Total Revenue(computed variable), Producer's Share Percentage and the 'Box Office Revenue' (computed variable) which is the revenue generated after cutting the Producer's share from the Total Box Office Revenue. Latter was calculated by multiplying the Total Box Office Revenue by (100-Producer's Share)/100.



	A	B	C	D	E	F	G	H	I	J	K	L
1	MovieID	Movie Name	ProdID	Release M	Genre	MKX1 Box Office	TST1 Box Office	KLT1 Box Office	Total Box Office	Producers	Box Office Revenue	
2	1050	High and Dry	CPX1	Oct	Horror	448000	400000	400000	1248000	41	736320	
3	1051	Have You Met Ted?	MMP1	Oct	Rom/Com	328000	320000	424000	1072000	60	428800	
4	1052	Candy Crush the Ultimate Saga	TWX1	Oct	Animation	448000	424000	480000	1352000	54	621920	
5	1053	Ye and the Legend of Davidson	MMP1	Oct	Action	432000	424000	432000	1288000	60	515200	
6	1054	Burgerland	TWX1	Oct	Animation	472000	368000	320000	1160000	54	533600	
7	1055	Frantic	CPX1	Oct	Horror	472000	336000	360000	1168000	41	689120	
8	1056	BaldEagles	22F1	Oct	Horror	480000	392000	472000	1344000	40	806400	
9	1057	A Haircut	22F1	Nov	Rom/Com	328000	456000	384000	1168000	40	700800	
10	1058	Horrible Horses	CPX1	Nov	Rom/Com	336000	392000	352000	1080000	41	637200	
11	1059	Too Technical To Handle	CPX1	Nov	Rom/Com	448000	336000	424000	1208000	41	712720	
12	1060	Spider-Man No Way Home	MMP1	Nov	Action	376000	352000	320000	1048000	60	419200	
13	1061	Human Centipede	CPX1	Nov	Horror	448000	424000	464000	1336000	41	788240	
14	1062	The Batman	TWX1	Nov	Action	320000	408000	376000	1104000	54	507840	
15	1063	The Motherland	CPX1	Dec	Animation	456000	344000	448000	1248000	41	736320	
16	1064	Benjamin the Inventor	MMP1	Dec	Horror	472000	416000	456000	1344000	60	537600	
17	1065	KSI	22F1	Dec	Rom/Com	416000	408000	352000	1176000	40	705600	
18	1066	How to Train your Demon	22F1	Dec	Animation	456000	472000	320000	1248000	40	748800	

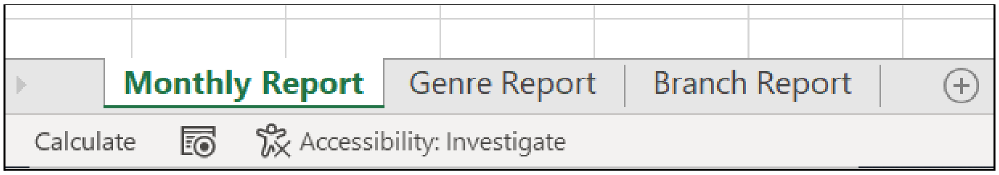
When the Final Workbook was generated, the following VBA Message Box appears:



Microsoft Excel

Final Workbook Has been generated and saved!

Programmatically, the Report Workbook is opened before the User clicks the respective Report Generating CommandButtons. Then, the respective worksheets are added to the Report Workbook when monthly, genre-wise, and branch-wise reports are requested. Report Workbooks are discussed in the next section of creating Tables and Charts.

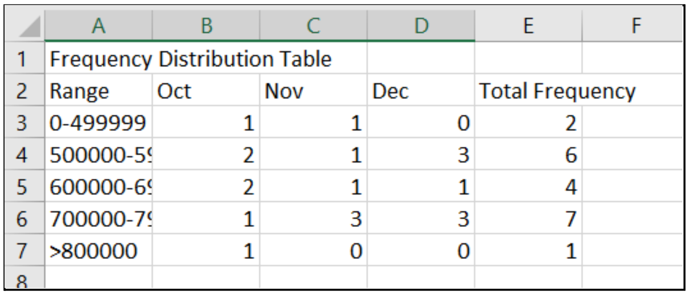


4. Creating Tables and Charts

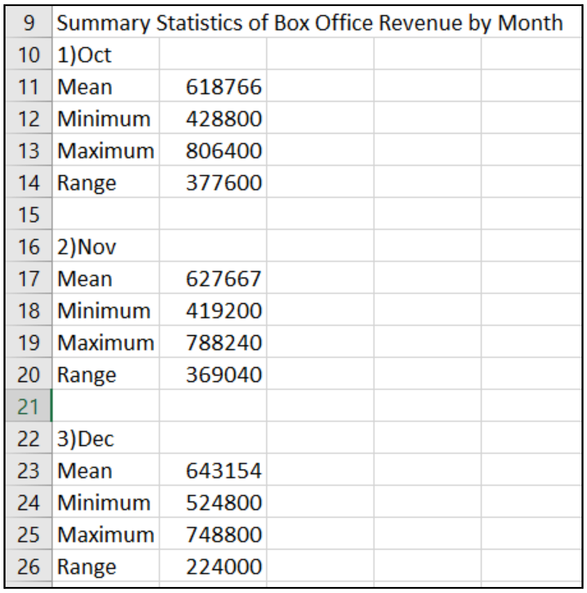
All tables and charts are for the records which are added to *FinalWorkbook.xlsx* by default.

4.1 Monthly Report

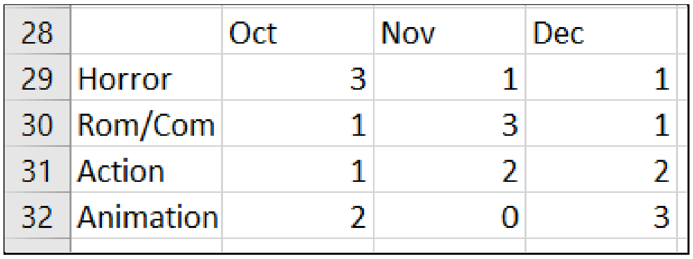
Following Tables and Charts are computed if the user requests all kinds of information listed in the OptionButton Controls on the Page "mulPgMonthlyReport" of the Userform.



	A	B	C	D	E	F
1	Frequency Distribution Table					
2	Range	Oct	Nov	Dec	Total Frequency	
3	0-499999	1	1	0	2	
4	500000-5	2	1	3	6	
5	600000-6	2	1	1	4	
6	700000-7	1	3	3	7	
7	>800000	1	0	0	1	
8						



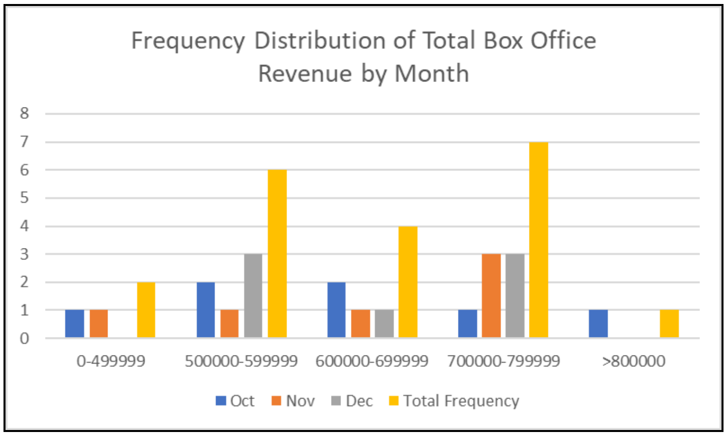
9	Summary Statistics of Box Office Revenue by Month	
10	1)Oct	
11	Mean	618766
12	Minimum	428800
13	Maximum	806400
14	Range	377600
15		
16	2)Nov	
17	Mean	627667
18	Minimum	419200
19	Maximum	788240
20	Range	369040
21		
22	3)Dec	
23	Mean	643154
24	Minimum	524800
25	Maximum	748800
26	Range	224000



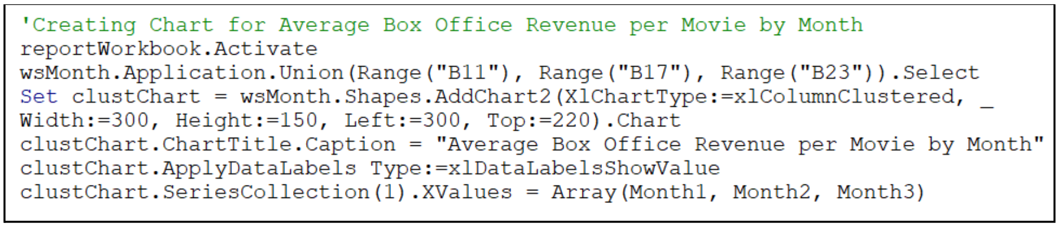
28		Oct	Nov	Dec
29	Horror	3	1	1
30	Rom/Com	1	3	1
31	Action	1	2	2
32	Animation	2	0	3

The following code creates a chart for Frequency Distribution of Total Box Office Revenue by Month using entries in the Frequency Distribution Table ("A2:A7", "E2:E7").

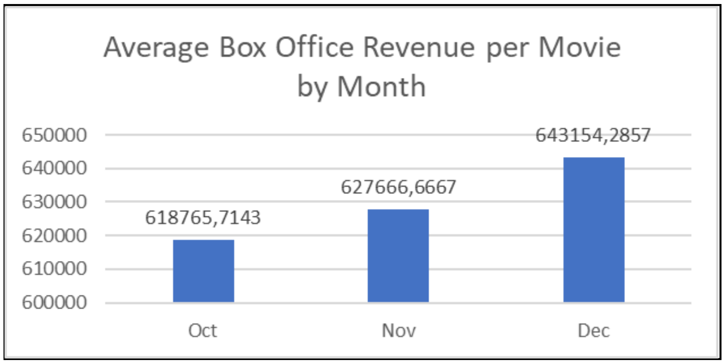
```
'Creating Chart for Frequency Distribution Table
wsMonth.Activate
wsMonth.Range("A2:A7", "E2:E7").Select
Set histChart = wsMonth.Shapes.AddChart2 _
(XlChartType:=xlColumnClustered, Left:=300, Top:=0).Chart
histChart.ChartTitle.Caption = _
"Frequency Distribution of Total Box Office Revenue by Month"
```



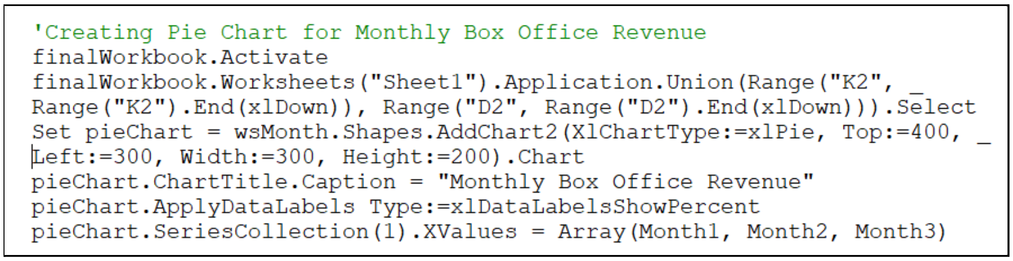
The code below creates a Clustered Chart for the “Average Box Office Revenue per Movie by Month” using entries in the mean value of the table “Summary Statistics for Box Office Revenue by Month (“B11”, “B17”, “B23”).



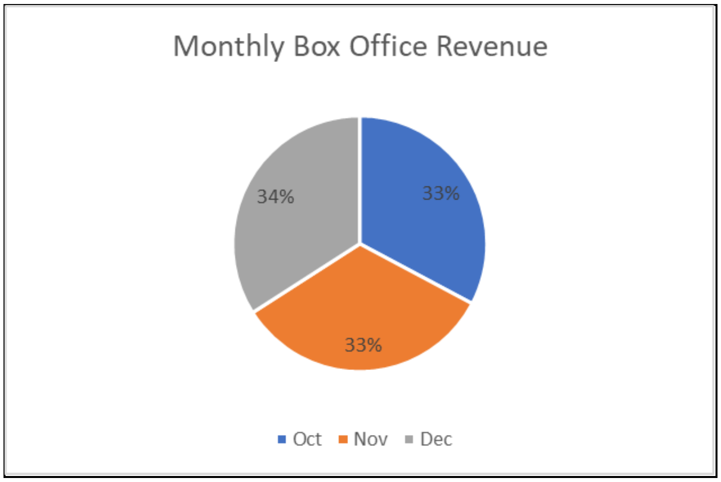
```
'Creating Chart for Average Box Office Revenue per Movie by Month
reportWorkbook.Activate
wsMonth.Application.Union(Range("B11"), Range("B17"), Range("B23")).Select
Set clustChart = wsMonth.Shapes.AddChart2(XlChartType:=xlColumnClustered, _
Width:=300, Height:=150, Left:=300, Top:=220).Chart
clustChart.ChartTitle.Caption = "Average Box Office Revenue per Movie by Month"
clustChart.ApplyDataLabels Type:=xlDataLabelsShowValue
clustChart.SeriesCollection(1).XValues = Array(Month1, Month2, Month3)
```



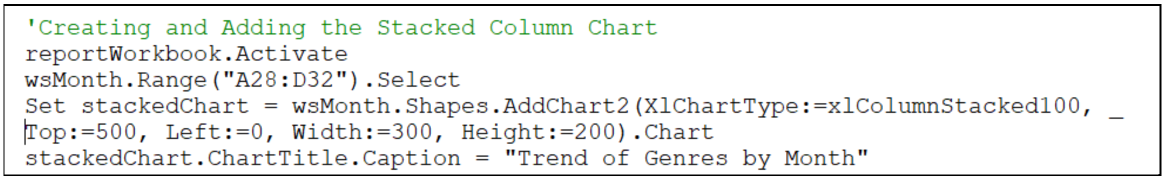
The code below creates a Pie Chart for the “Monthly Box Office Revenue” using column values of “K” and “D” of the Final Workbook, where “D” accounts for the Month Name Column and “K” for the Total Box Office Revenue generated after subtracting the producer company’s share.



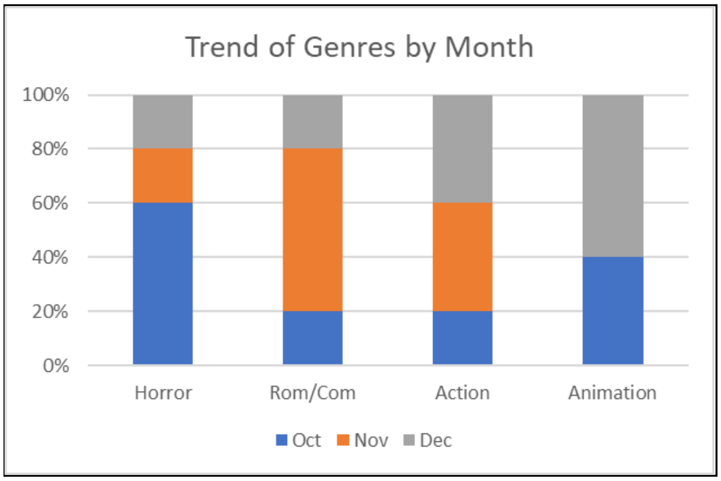
```
'Creating Pie Chart for Monthly Box Office Revenue
finalWorkbook.Activate
finalWorkbook.Worksheets("Sheet1").Application.Union(Range("K2", _
Range("K2").End(xlDown)), Range("D2", Range("D2").End(xlDown))).Select
Set pieChart = wsMonth.Shapes.AddChart2(XlChartType:=xlPie, Top:=400, _
Left:=300, Width:=300, Height:=200).Chart
pieChart.ChartTitle.Caption = "Monthly Box Office Revenue"
pieChart.ApplyDataLabels Type:=xlDataLabelsShowPercent
pieChart.SeriesCollection(1).XValues = Array(Month1, Month2, Month3)
```



The code below creates a Stacked Column Chart for the “Trend of Genres my Month” using values in the range (“A2:A7”, “E2:E7”).

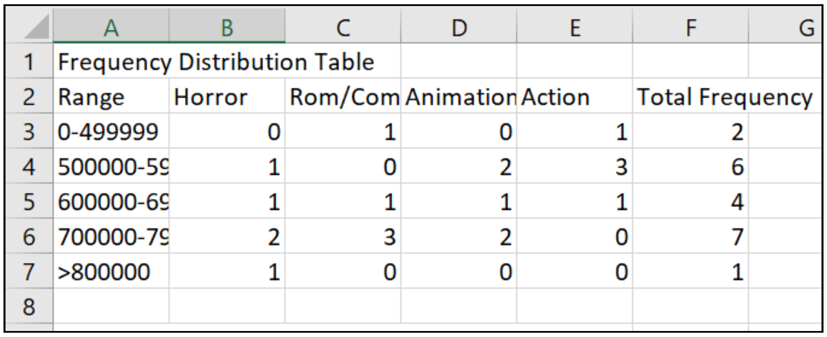


```
'Creating and Adding the Stacked Column Chart
reportWorkbook.Activate
wsMonth.Range("A28:D32").Select
Set stackedChart = wsMonth.Shapes.AddChart2(XlChartType:=xlColumnStacked100, _
Top:=500, Left:=0, Width:=300, Height:=200).Chart
stackedChart.ChartTitle.Caption = "Trend of Genres by Month"
```

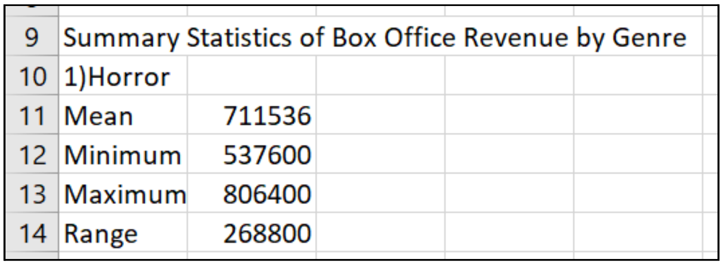


4.2 Genre-wise Report

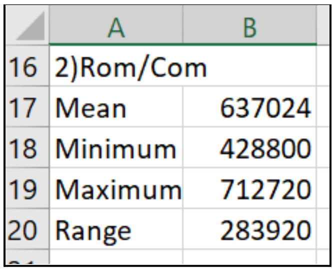
Following Tables and Charts are computed if the user requests all kinds of information listed in the OptionButton Controls on the Page “mulPgGenreReport” of the Userform.



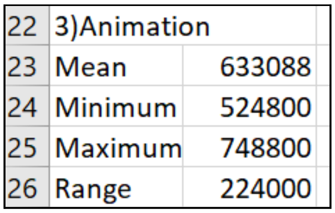
	A	B	C	D	E	F	G
1	Frequency Distribution Table						
2	Range	Horror	Rom/Com	Animation	Action	Total Frequency	
3	0-499999	0	1	0	1	2	
4	500000-59	1	0	2	3	6	
5	600000-69	1	1	1	1	4	
6	700000-79	2	3	2	0	7	
7	>800000	1	0	0	0	1	
8							



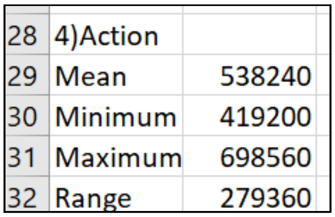
9	Summary Statistics of Box Office Revenue by Genre	
10	1)Horror	
11	Mean	711536
12	Minimum	537600
13	Maximum	806400
14	Range	268800



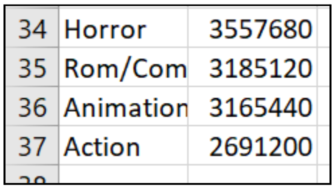
	A	B
16	2)Rom/Com	
17	Mean	637024
18	Minimum	428800
19	Maximum	712720
20	Range	283920



22	3)Animation	
23	Mean	633088
24	Minimum	524800
25	Maximum	748800
26	Range	224000



28	4)Action	
29	Mean	538240
30	Minimum	419200
31	Maximum	698560
32	Range	279360

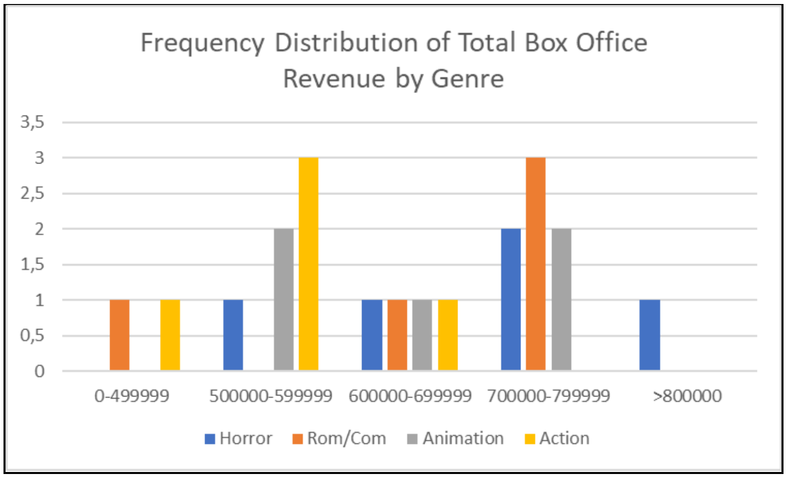


34	Horror	3557680
35	Rom/Com	3185120
36	Animation	3165440
37	Action	2691200

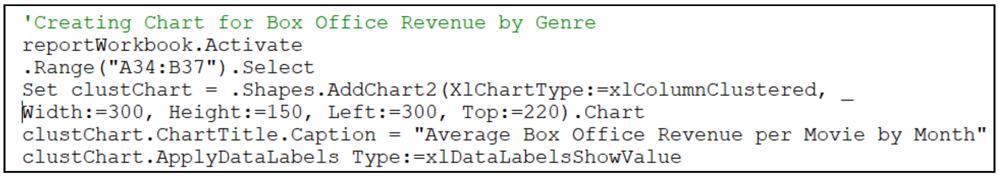
Table for chart^

The following code creates a chart for Frequency Distribution of Total Box Office Revenue by Genre using entries in the Frequency Distribution Table (“A2:E7”).

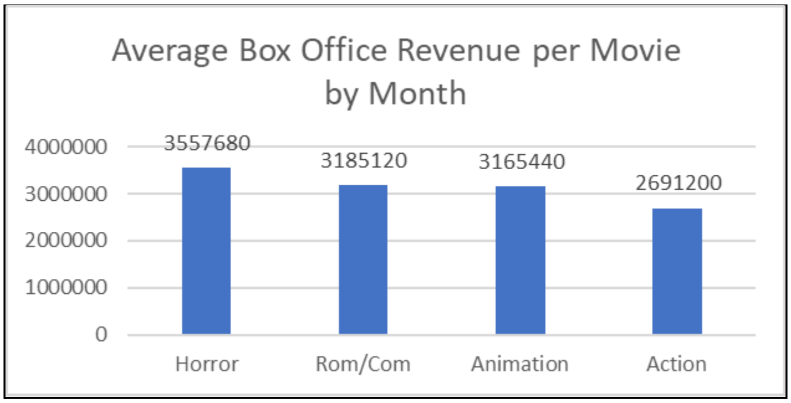
```
'Creating Chart for Frequency Distribution Table
wsGenre.Activate
wsGenre.Range("A2:E7").Select
Set histChart = wsGenre.Shapes.AddChart2(XlChartType:=xlColumnClustered, Left:=320, Top:=0).Chart
histChart.ChartTitle.Caption = "Frequency Distribution of Total Box Office Revenue by Genre"
```



The code below creates a Bar Chart for the “Average Box Office Revenue per Movie by Genre” using entries in the range “A34:B37”

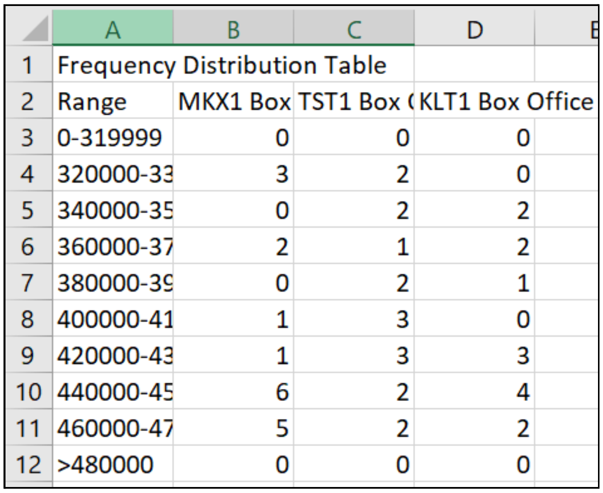


```
'Creating Chart for Box Office Revenue by Genre
reportWorkbook.Activate
.Range("A34:B37").Select
Set clustChart = .Shapes.AddChart2(XlChartType:=xlColumnClustered, _
Width:=300, Height:=150, Left:=300, Top:=220).Chart
clustChart.ChartTitle.Caption = "Average Box Office Revenue per Movie by Month"
clustChart.ApplyDataLabels Type:=xlDataLabelsShowValue
```

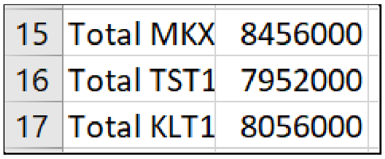


4.3 Branch-wise Report

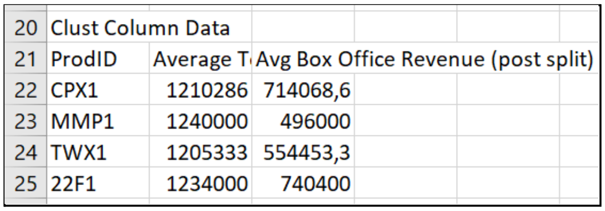
Following Tables and Charts are computed if the user requests all kinds of information listed in the OptionButton Controls on the Page “mulPgBranchReport” of the Userform.



	A	B	C	D	E
1	Frequency Distribution Table				
2	Range	MKX1 Box	TST1 Box (	KLT1 Box Office	
3	0-319999	0	0	0	
4	320000-33	3	2	0	
5	340000-35	0	2	2	
6	360000-37	2	1	2	
7	380000-39	0	2	1	
8	400000-41	1	3	0	
9	420000-43	1	3	3	
10	440000-45	6	2	4	
11	460000-47	5	2	2	
12	>480000	0	0	0	

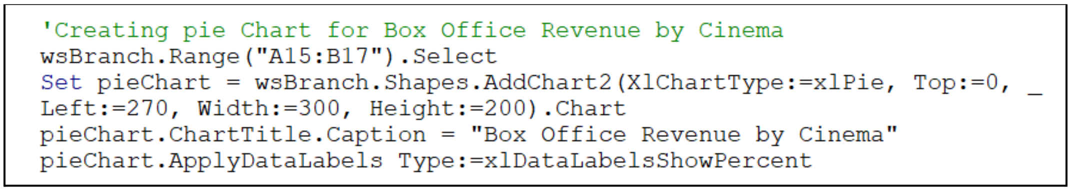


15	Total MKX	8456000
16	Total TST1	7952000
17	Total KLT1	8056000

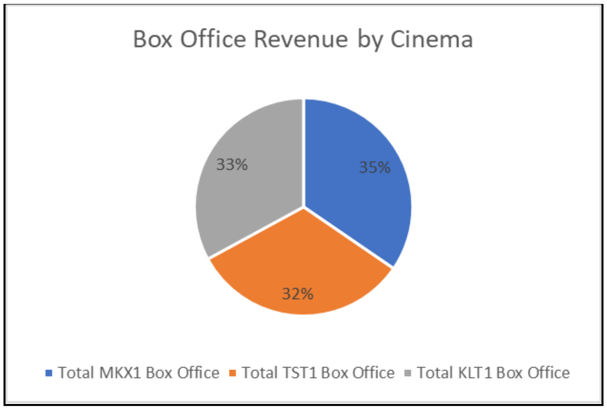


20	Clust Column Data		
21	ProdID	Average T	Avg Box Office Revenue (post split)
22	CPX1	1210286	714068,6
23	MMP1	1240000	496000
24	TWX1	1205333	554453,3
25	22F1	1234000	740400

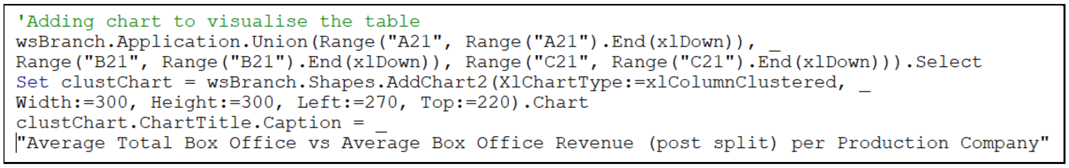
The following code creates a Pie chart for Box Office Revenue by Cinema Location or Branch using entries in the Range “A15:B17”.



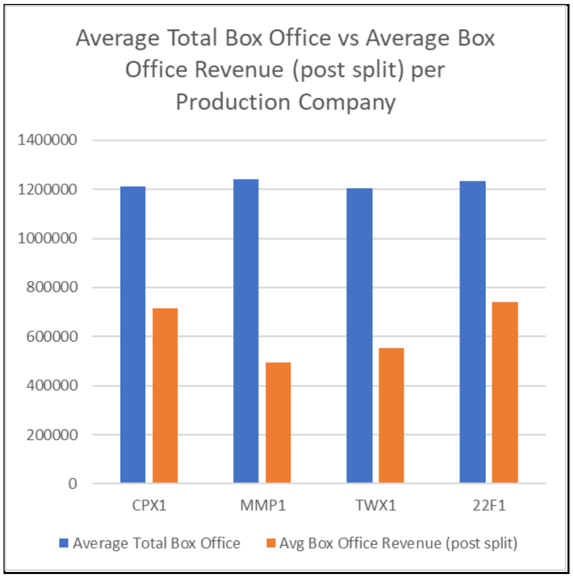
```
'Creating pie Chart for Box Office Revenue by Cinema
wsBranch.Range("A15:B17").Select
Set pieChart = wsBranch.Shapes.AddChart2(XlChartType:=xlPie, Top:=0, _
Left:=270, Width:=300, Height:=200).Chart
pieChart.ChartTitle.Caption = "Box Office Revenue by Cinema"
pieChart.ApplyDataLabels Type:=xlDataLabelsShowPercent
```



The code below creates a Bar Chart for the “Average Total Box Office and Average Box Office Revenue(post-split) per Production Company” using entries in the range “A21:C25”



```
'Adding chart to visualise the table
wsBranch.Application.Union(Range("A21", Range("A21").End(xlDown)), _
Range("B21", Range("B21").End(xlDown)), Range("C21", Range("C21").End(xlDown))).Select
Set clustChart = wsBranch.Shapes.AddChart2(XlChartType:=xlColumnClustered, _
Width:=300, Height:=300, Left:=270, Top:=220).Chart
clustChart.ChartTitle.Caption = _
"Average Total Box Office vs Average Box Office Revenue (post split) per Production Company"
```



4. Conclusion

This Application for the Box Office Cinema carries out all core functions needed to compute three kinds of reports with relevant tables and charts. It includes most of the commonly used UserForm Controls like Labels, TextBoxes, ListBox, OptionButton, ComboBox etc. Generating reports using this application is user-friendly because the filters can be applied by the user and the saved Report Workbook contains only the required data. Moreover, it allows users to manipulate records at runtime and generate updated charts and statistics.

This application makes use of data extraction and data merging techniques and also computes for new variables. All the reporting is done using data from our derived dataset (*FinalWorkbook.xlsx*) and does not have any connection to the source datasets. Most of the data is programmed and accessed dynamically. Therefore, this automation of our reporting task can be used to report on data periodically. The period assumed is every quarter of the year (i.e. every 3 months).

The Program displays the frequency distribution of Total Box Office Revenue by Months of the quarter, Genre of the movies, and by Branch of the Cinema. It also generated Tables and Clustered Columns for the Summary Statistics of Total Box Office Revenue and Average Box Office Revenue per Movie, along with a Pie Chart for the Revenue and Stacked Columns for the Trend of Genres my Months. These reports will help the cinema make more informed decisions regarding their business in terms of which type of movie works in which month, observing trends of genres, which production company gives the cinema the most returns, which branch of their cinema needs most attention etc.