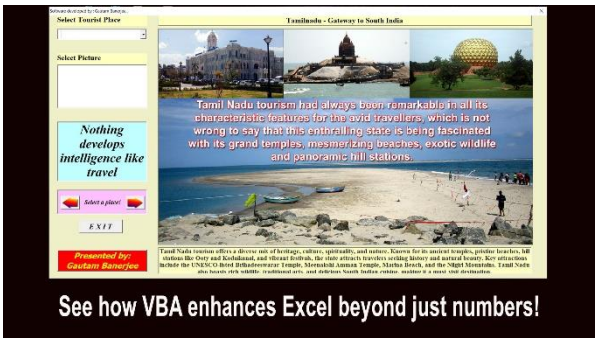

Excel VBA Picture Viewer | Interactive Image Viewer for Tourism Guide

"Learn how to create a fully functional Picture Viewer in Excel using VBA! 🚀 This interactive tourism guide lets users select a tourist place and view related images with just a click. Perfect for showcasing destinations, products, or personal galleries. Watch till the end to see how VBA enhances Excel beyond just numbers! Don't forget to like, share, and subscribe for more amazing Excel automation tutorials! 📊

How to Set Up the Picture Viewer:

1. Create a folder (e.g., “D:\MyClick”) to store the Excel workbook.
2. Create a subfolder (e.g., "D:\MyClick\MyPictures") to store the pictures.
3. Ensure the picture resolution is 1280x720 pixels for a smooth viewing experience.
4. The Excel workbook must be saved as a 'Macro-Enabled Workbook' (.xlsm) file format; otherwise, the macros will not function.

5. Rename "Sheet1" to “TouristPlace”.
Fill the range starting from **cell A2** with the following place names:
(You can assign different places as per your preference.)

- Chennai
- Rameswaram
- Kodaikanal
- Kanyakumari
- Pondicherry
- City_Of_Joy

6. Create new sheets using the exact names listed above:
(Ensure the sheet names match the entries in the "TouristPlace" sheet.)

- Chennai
- Rameswaram
- Kodaikanal
- Kanyakumari
- Pondicherry
- City_Of_Joy

7. Enter picture names in each sheet as follows from Column A2:
(Each picture name should follow a consistent format based on the sheet name.)

- **Chennai:** Chennai01, Chennai02, Chennai03, …
- **Rameswaram:** Rameswaram01, Rameswaram02, Rameswaram03, …
- **Kodaikanal:** Kodaikanal01, Kodaikanal02, Kodaikanal03, …

- *(Repeat this pattern for all sheets.)*

Ensure that the picture names follow the specified format and are stored in the folder: "D:\MyClick\MyPictures" (I am using only .jpg format)

Let's design the **UserForm for your Picture Viewer** with these elements:

UserForm Layout

- **Few Labels**
- **ComboBox** (`cmbPlaces`): To select a tourist place.
- **ListBox** (`lstPictures`): To display available pictures.
- **Image Control** (`imgPreview`): To show the selected picture.
- **Buttons**:
  - **Next / Previous** (`btnNext`, `btnPrev`) – Navigate through images.
  - **Exit** (`btnExit`) – Close the form.
  - **Two extra buttons** *(PicView2 and CmdAdvance)* **to open two other forms**

Steps to Create the UserForm:

1. **Open the VBA Editor** (**ALT + F11**)
2. **Insert a New UserForm** (**Insert > UserForm**)
3. **Rename the UserForm to "MainForm" from the Properties window**
4. **Design the Form:**
   - Add **ComboBox (`cmbPlaces`)** for tourist places
   - Add **ListBox (`lstPictures`)** for images
   - Add **Image Control (`imgPreview`)**
   - Add **Buttons (`btnNext`, `btnPrev`, `btnExit`, *`PicView2 and CmdAdvance`*)**
   - **Add Few Labels**
   - **Don’t change the above objects name**

To set the properties of the **UserForm** named `"MainForm"` in Excel VBA, follow these steps

1. Open the VBA Editor and Select MainForm

1. Press **`ALT + F11`** to open the **VBA Editor**.
2. In the **Project Explorer** (left panel), expand **VBAProject (Your Workbook Name)**.
3. Double-click **`MainForm`** to open it.
4. Press **`F4`** to open the **Properties Window** (if it's not already open).

2. Set Important Properties in the Properties Window

Property Name	Value (Example)	Description
Name	`MainForm`	(Already set) Name of the UserForm
Caption	`"Picture Viewer - Tourist Guide"`	Title displayed on the form
BackColor	`Select Color palette`	Set background color
BorderStyle	`1 - fmBorderStyleSingle`	Sets a fixed border
StartUpPosition	`2 - CenterScreen`	Centers the form when opened
Height	`600`	Set form height (adjust as needed)
Width	`1100`	Set form width (adjust as needed)
Font	`Arial, 12`	Sets default font for text
AutoSize	`False`	Prevents automatic resizing
Enabled	`True`	Ensures the form is active

How to Insert an Image Control in MainForm and Set Properties

1. Insert an Image Control in MainForm

1. **Open the VBA Editor:** Press **`ALT + F11`**.
2. **Open MainForm:** In the **Project Explorer**, double-click on **MainForm**.
3. **Open the Toolbox:** If the **Toolbox** is not visible, press **`CTRL + T`**.
4. **Add Image Control:**
   - Click on the **"Image"** control (📷 icon) in the Toolbox.
   - Click and drag on the form to place the **Image Control**.

2. Set Properties for Image Control

1. Select the **Image Control** (e.g., `Image1`).
2. Open the **Properties Window** (`F4`).
3. Modify these properties:

Property Name	Value (Example)	Description
Name	`MainImageFrame`	Set a meaningful name
Picture	`(None)`	Leave empty (will load dynamically)
BackStyle	`0 - fmBackStyleTransparent`	Makes background transparent
BorderStyle	`1 - fmBorderStyleSingle`	Adds a single-line border
SizeMode	`1 - fmSizeModeStretch`	Ensures image fits properly
Width	`825`	Adjust as needed
Height	`475`	Adjust as needed

How to Insert a ComboBox in MainForm and Its Functionality

1□. Insert a ComboBox in MainForm

1. **Open the VBA Editor** (`ALT + F11`).
2. **Open MainForm**: In **Project Explorer**, double-click **MainForm**.
3. **Open the Toolbox** (`CTRL + T` if not visible).
4. **Add ComboBox**:
   - Click the **"ComboBox"** control (dropdown icon).
   - Click and drag on the form to place it.
5. **Rename the ComboBox**:
   - Select it and press **F4** (Properties Window).
   - Change **Name** to `"ComboPlace"` (or any relevant name).

2□. Set Important Properties for ComboBox

Property Name	Value (Example)	Description
Name	`ComboPlace`	Set a meaningful name
Style	`0 - fmStyleDropDownCombo`	Allows manual entry + dropdown
BorderStyle	`1 - fmBorderStyleSingle`	Adds a border
ColumnCount	`1`	Number of columns (usually 1)
TextAlign	`1 - fmTextAlignLeft`	Aligns text to the left
AutoTab	`False`	Prevents auto-jump to the next control

3□. What is the Function of a ComboBox?

- A **ComboBox** allows users to **select a value from a list** or **type a custom value**.
- In the **Picture Viewer**, we can use it to **select tourist places** (folder names).
- When a user selects a place, the ListBox can show images from that folder.

How to Insert a ListBox in MainForm and How It Works

A **ListBox** in VBA displays a list of items (e.g., image file names) that users can select. In your **Picture Viewer**, it will show the available pictures from the selected folder.

1□. Steps to Insert a ListBox in MainForm

1. **Open the VBA Editor** (`ALT + F11`).
2. **Open MainForm**: In **Project Explorer**, double-click **MainForm**.
3. **Open the Toolbox** (`CTRL + T` if not visible).
4. **Add a ListBox**:
   - Click the **"ListBox"** control (▇ icon).

- Click and drag on the form to place it.

5. **Rename the ListBox**:
   - Select it and press **F4** (Properties Window).
   - Change **Name** to "***ListOfPictures***" (or any relevant name).

2□. Set Important Properties for ListBox

Property Name	Value (Example)	Description
Name	`ListOfPictures`	Set a meaningful name
MultiSelect	`0 - fmMultiSelectSingle`	Allows selecting only one item
ColumnCount	`1`	Number of columns (usually 1)
TextAlign	`1 - fmTextAlignLeft`	Aligns text to the left
BorderStyle	`1 - fmBorderStyleSingle`	Adds a border
IntegralHeight	`True`	Ensures proper item display
Enabled	`True`	Ensures users can interact with it

3□. How a ListBox Works in the Picture Viewer

- It displays **all image file names** from the selected tourist place folder.
- When a user **clicks an image name**, it will be displayed in the **Image Control (`MainImageFrame`)**.

How to Insert a Command Button in MainForm, Set Properties, and Make It Work

A **Command Button** is used to trigger an action when clicked. In your **Picture Viewer**, buttons can be used to:

✓ Load pictures

✓ Navigate to next/previous image

✓ Exit the form

1. Steps to Insert a Command Button in MainForm

1. **Open the VBA Editor** (`ALT + F11`).
2. **Open MainForm**: In **Project Explorer**, double-click **MainForm**.
3. **Open the Toolbox** (`CTRL + T` if not visible).
4. **Add a Command Button**:
   - Click the **"CommandButton"** control
   - Click and drag on the form to place it.
5. **Rename the Command Button**:
   - Select it and press **F4** (Properties Window).
   - Change the **Name** (e.g., **`"btnNext"`**, **`"btnPrev"`**, **`"btnExit"`**).

- Change the **Caption** (text on the button)

2. Set Important Properties for Command Button

Property Name	Value (Example)	Description
Name	btnPrev	Button to view Previous Pictures
Caption	"Previous"	Text displayed on the button
Font	Arial, 12, Bold	Adjust font size and style
BackColor	&H00C0C0C0&	Light gray button

3. How a Command Button Works

- When **clicked**, it runs a VBA **macro (Sub procedure)**.
- Each button performs a **specific task**, such as **loading images, navigating, or exiting** the form.

What is the Use of a Frame in VBA and How to Insert It?

A **Frame** in VBA is used to **group related controls together**, improving form organization and appearance. In your **Picture Viewer**, you can use Frames to:

✅ **Group controls** (e.g., ComboBox, ListBox, and Buttons for better alignment).

✅ **Improve form design** by separating sections (e.g., Image Display vs. Controls).

✅ **Enhance user experience** by making navigation clearer.

1. Steps to Insert a Frame in MainForm

1. **Open the VBA Editor** (`ALT + F11`).
2. **Open MainForm**: In **Project Explorer**, double-click **MainForm**.
3. **Open the Toolbox** (`CTRL + T` if not visible).
4. **Add a Frame**:
   - Click the **"Frame"** control (box icon).
   - Click and drag to place it on the form.
5. **Rename the Frame**:
   - Select it and press **F4** (Properties Window).
   - Change **Name** to `"fraControls"` (or any relevant name).
   - Change **Caption** (e.g., `"Controls"` or `"Image Selection"`).

2. Set Important Properties for a Frame

Property Name	Value (Example)	Description
Name	**fraControls**	Identifies the frame in VBA
Caption	**"Controls"**	Title shown on the frame
BorderStyle	**1 - fmBorderStyleSingle**	Adds a border
BackColor	**&H00E0E0E0&**	Light gray background
SpecialEffect	**2 - fmSpecialEffectSunken**	Gives a sunken appearance

3. How to Use Frames in the Picture Viewer

- You can **place other controls inside a Frame**, such as:
  - **ComboBox (`cmbPlaces`)** – To select a tourist place
  - **ListBox (`lstPictures`)** – To list available pictures
  - **Buttons (`btnLoad`, `btnNext`, `btnPrev`, `btnExit`)** – For navigation

☞ **Controls inside a Frame move together** when repositioning the Frame!

4. Example: Organizing the Controls Inside a Frame

1. **Insert a Frame (`fraControls`)** on the form.
2. **Drag & Drop** the ComboBox, ListBox, and Buttons into the Frame.
3. The Frame will now act as a container for these controls.

How to Insert a Label in MainForm and Its Function

A **Label** in VBA is used to display text, such as titles, instructions, or image details. In your **Picture Viewer**, Labels can show:

✓ The selected tourist place

✓ The currently displayed image name

✓ Instructions for users

1. Steps to Insert a Label in MainForm

1. **Open the VBA Editor** (`ALT + F11`).
2. **Open MainForm**: In **Project Explorer**, double-click **MainForm**.
3. **Open the Toolbox** (`CTRL + T` if not visible).
4. **Add a Label**:
   - Click the **"Label"** control ( icon).
   - Click and drag on the form to place it.
5. **Rename the Label**:
   - Select it and press **F4** (Properties Window).
   - Change the **Name** (e.g., `"LblCaption"`, `"LblDes"`, `"lblStatus"`).
   - Change the **Caption** (default text).

2. Set Important Properties for a Label

Property Name	Value (Example)	Description
Name	**LblCaption**	Used in VBA code to reference the label
Caption	**"Picture Viewer"**	Text displayed on the form
Font	**Arial, 14, Bold**	Sets the font style and size
BackStyle	**0 - fmBackStyleTransparent**	Makes background transparent

Property Name	Value (Example)	Description
TextAlign	**2 - fmTextAlignCenter**	Centers text in the label
BorderStyle	**0 - None**	Removes border

3. How a Label Works in the Picture Viewer

- Displays **static text** like the title or instructions.
- Updates **dynamically** (e.g., showing the current image name).

How to Create a Desktop Shortcut for Your Excel File and Change the Icon

You can create a shortcut for your Excel file (`D:\MyClick\YourFile.xlsx`) on the **desktop** manually or using **VBA code**.

How to Create Shortcut on Desktop?

1. **Go to Your Excel File**
   - Open **File Explorer** and navigate to:
     📂 `D:\MyClick\YourFile.xlsm`
2. **Create a Shortcut**
   - **Right-click** the Excel file → **Send to** → **Desktop (create shortcut)**
   - A shortcut will appear on your desktop.
3. **Change the Shortcut Icon**
   - **Right-click the shortcut** on the desktop → **Properties**
   - Click **Change Icon**
   - Select an icon (`.ico`) file or choose from the default Windows icons.
   - Click **OK** and **Apply**.

How to Display a UserForm and Hide the Excel Workbook

When you open your **Picture Viewer**, you may want to:
✅ Show only the **UserForm** (`MainForm`)
✅ Hide the **Excel workbook** (`Application Window`)
✅ Restore the workbook when closing the form

1. VBA Code to Show UserForm and Hide Excel

Add this code to the **Workbook_Open** event to automatically open the form and hide Excel:

◆ In "ThisWorkbook" Module (Auto-Run on Open)

```
Private Sub Workbook_Open()
    Application.Visible = False   ' Hide Excel
    MainForm.Show                 ' Show UserForm
End Sub
```

✅ **Hides Excel completely** and displays `MainForm`.

2. VBA Code to Restore Excel When Closing the Form

Modify the **UserForm_Close** event to show Excel again when exiting:

◆ In MainForm (UserForm Code)

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
    Application.Visible = True    ' Show Excel when closing the form
End Sub
```

✅ Reopens Excel when you close **`MainForm`.**

Explanation of VBA Code: Close Button in MainForm

The following VBA code is used to **close the UserForm (`MainForm`)** and **restore the Excel application** when a button (`CloseMainForm`) is clicked.

```
Private Sub CloseMainForm_Click()
   Unload Me  ' Close the form
   Application.Visible = True  ' Show Excel again
End Sub
```

How This Code Works

1. **`Private Sub CloseMainForm_Click()`**

- This is a **Click event** for a button named `CloseMainForm`.
- Runs when the user clicks the **Close button** on the form.

2. **`Unload Me`**

- `Me` refers to the **current form** (`MainForm`).
- `Unload Me` **closes and removes the form from memory**.
- Any variables or processes related to the form will be cleared.

3. **`Application.Visible = True`**

- Makes **Excel visible again** after closing the form.
- This is important if Excel was hidden when the form was opened.

When to Use This Code?

✅ **If Excel was hidden** when the form opened (`Application.Visible = False`).
✅ **If you want to close the form and return to the workbook** instead of exiting Excel.
✅ **When creating a standalone-like UserForm experience** (like your Picture Viewer).

Explanation of `UserForm_QueryClose` Event in VBA

The following code executes when the user tries to close the UserForm.

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
    ' Show Excel when form is closed
    Application.Visible = True ' Show Excel before closing
    ThisWorkbook.Saved = True  ' Prevent save prompt
End Sub
```

How This Code Works

1. **UserForm_QueryClose** Event

- This **event triggers when the form is about to close**.
- It allows you to **control what happens before the form actually closes**.

2. **Application.Visible = True**

- If Excel was **hidden when the form opened**, this line **makes Excel visible again**.
- Without this, Excel would remain hidden after closing the form.

3. **ThisWorkbook.Saved = True**

- **Prevents the "Do you want to save changes?" prompt** when closing Excel.
- Marks the workbook as **already saved**, even if there are unsaved changes.

When Is This Code Useful?

✅ If your **UserForm hides Excel (`Application.Visible = False`)** on startup.
✅ If you **don't want Excel to ask "Save Changes?"** every time you close the form.

What Happens When the User Closes the Form?

✓ Excel **becomes visible again** if it was hidden.
✓ The workbook is **marked as saved**, so there's **no save prompt** when exiting Excel.

Explanation of `UserForm_Initialize` Event in VBA

The `UserForm_Initialize` event is triggered **automatically** when the form is opened. It sets up the form by:
✅ Hiding Excel
✅ Loading a list of places into a combo box
✅ Displaying a default image
✅ Formatting labels and background
✅ Adding tooltips

1️⃣. Hiding the Excel Application

```
Application.Visible = False
```

🔹 When the form loads, Excel is **hidden** to create a standalone-like experience.

2️⃣. Setting Up the Worksheet Reference

```
Set ws = ThisWorkbook.Sheets("TouristPlace")
```

🔹 Sets `ws` as a reference to the **"TouristPlace"** worksheet where place names are stored.

3️⃣. Finding the Last Row in Column A (Avoiding Blank Spaces)

```
lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row
```

🔹 This finds the **last filled row** in **Column A** (avoiding empty spaces).
🔹 Ensures only existing data is used.

4️⃣. Populating the Combo Box (`ComboPlace`) with Place Names

```
ComboPlace.Clear
For Each cell In ws.Range("A2:A" & lastRow)
    If Trim(cell.Value) <> "" Then ' Avoid adding blank cells
        ComboPlace.AddItem cell.Value
    End If
Next cell
```

🔹 Clears any previous items in the combo box.
🔹 Loops through **Column A** (from row **2 to lastRow**) and **adds only non-empty values** to `ComboPlace`.

5□. Stting Up the Default Image

```
MainImageFrame.PictureSizeMode = fmPictureSizeModeZoom
picFolder = "D:\MyClick\MyPictures\"
defaultImgPath = picFolder & "HomeScreen.jpg"

If Dir(defaultImgPath) <> "" Then
    MainImageFrame.Picture = LoadPicture(defaultImgPath)
Else
    MainImageFrame.Picture = LoadPicture("") ' Clear image if not found
End If
```

- **Sets the picture mode** to `Zoom` so images fit properly.
- **Defines the folder path** where pictures are stored.
- **Loads a default image** (`HomeScreen.jpg`) if it exists.
- If the image **is not found**, it clears the picture frame.

6□. Formatting Labels (Text, Size, Alignment, Color)

```
LblCaption.Caption = "Tamilnadu - Gateway to South India"
LblDes.Caption = "Tamil Nadu tourism offers a diverse mix of heritage,
culture, spirituality, and nature..."
LblCaption.TextAlign = fmTextAlignCenter
LblDes.TextAlign = fmTextAlignCenter
LblCaption.Font.Size = 16
LblCaption.Font.Bold = True
LblDes.Font.Size = 14
LblDes.ForeColor = RGB(50, 50, 50) ' Dark gray for better readability
Me.BackColor = RGB(240, 240, 200) ' Light yellow background for a warm feel
```

- Adds a **heading (`LblCaption`) and description (`LblDes`)** about Tamil Nadu tourism.
- **Centers the text** and adjusts the **font size & color** for better readability.
- Sets a **light yellow background** (`RGB(240, 240, 200)`) for a warm feel.

7□. Adding a Presentation Credit

```
Label3.Caption = "Presented by: Gautam Banerjee"
Label3.TextAlign = fmTextAlignCenter
Label3.Font.Bold = True
```

- Displays "`Presented by: Gautam Banerjee`".
- Makes the text **bold** and **centered**.

8□. Setting the Picture Mode to Stretch

```
MainImageFrame.PictureSizeMode = fmPictureSizeModeStretch
```

- Ensures that images **fill the frame** while maintaining proportions.

9□. Initializing the Picture Count Label

```
PicCount.Caption = "Select a place!"
```

🔹 The `PicCount` label tells the user to **select a place** from the combo box.

10. Adding Tooltips for Navigation Buttons

```
AddToolTip BtnPrev, "Click here to View previous image"
AddToolTip BtnNext, "Click here to View next image"
```

🔹 Adds **tooltips** to the **Previous** and **Next** buttons to improve usability.
🔹 When the user hovers over these buttons, they will see a small **popup hint**.

💡 Summary

✓ **Hides Excel** when the form opens.
✓ **Loads place names** into a combo box from `TouristPlace` sheet.
✓ **Displays a default image** (`HomeScreen.jpg`).
✓ **Formats labels** with text, size, color, and alignment.
✓ **Adds tooltips** for better user experience.

Explanation of `ComboPlace_Change` Event in VBA

The `ComboPlace_Change` event runs **when the user selects a place** from the `ComboPlace` drop-down list.

📌 What This Code Does?

✅ Updates the **caption** (`LblCaption`) to welcome the user to the selected place.
✅ Fetches and **displays the description** (`LblDes`) of the selected place.
✅ Clears and **populates the ListBox (`ListOfPictures`)** with available images.
✅ **Automatically loads the first image** from the list.
✅ **Counts and displays** the number of available pictures.

1□. Setting Up Variables & Worksheet References

```
Dim ws As Worksheet, wsSpot As Worksheet
Dim selectedSpot As String
Dim lastRow As Long, lastRow1 As Long
Dim spotCell As Range, picCell As Range
Dim imgPath As String, picFolder As String

' Define the worksheet where tourist spots are listed
Set ws = ThisWorkbook.Sheets("TouristPlace")
```

- `ws` refers to the **"TouristPlace"** sheet (where place names and descriptions are stored).
- `wsSpot` is an optional sheet where **images of the selected place** are listed.
- `selectedSpot` stores the **place name selected** from `ComboPlace`.
- `picFolder` will store the **folder path where images are saved**.

2. Getting the Selected Place Name

```
selectedSpot = ComboPlace.Value
```

- Retrieves the **place name** selected in `ComboPlace`.

3. Updating the Caption Label (`LblCaption`)

```
LblCaption.Caption = "Welcome to " & selectedSpot
```

- Updates the label with `"Welcome to [selected place]"`.

4. Finding the Description from the "TouristPlace" Sheet

```
lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row
For Each spotCell In ws.Range("A2:A" & lastRow)
    If spotCell.Value = selectedSpot Then
        LblDes.Caption = spotCell.Offset(0, 1).Value ' Get description from
Column B
        Exit For
    End If
Next spotCell
```

- Finds the **last row** in **Column A** (where place names are listed).
- Loops through **each place name** and checks if it matches `selectedSpot`.
- If a match is found, it **gets the description** from **Column B** and displays it in `LblDes`.

5. Defining the Image Path

```
picFolder = "D:\MyClick\MyPictures\"
imgPath = picFolder & "Caption" & selectedSpot
```

- **Defines the folder path** where images are stored.
- Constructs an **image file name** based on `"Caption" + Selected Spot`.
- The extension (`.jpg`) is not added yet because multiple formats might exist.

6. Clearing the ListBox (`ListOfPictures`)

```
ListOfPictures.Clear
```

- **Removes previous images** from `ListOfPictures` before adding new ones.

7. Checking If a Sheet for the Selected Spot Exists

```
On Error Resume Next
Set wsSpot = ThisWorkbook.Sheets(selectedSpot)
On Error GoTo 0
```

- Tries to set `wsSpot` to a **sheet named after the selected place**.
- **Avoids an error** if the sheet **does not exist**.

8. Populating the ListBox with Picture Names

```
If Not wsSpot Is Nothing Then
    lastRow1 = wsSpot.Cells(wsSpot.Rows.Count, 1).End(xlUp).Row
    For Each picCell In wsSpot.Range("A2:A" & lastRow1)
        If picCell.Value <> "" Then
            ListOfPictures.AddItem picCell.Value ' Add Picture Name from
Column A
        End If
    Next picCell
End If
```

- If the sheet **exists**, it gets the **last row** in **Column A**.
- Loops through each row in **Column A** and adds the **image names** to `ListOfPictures`.

9. Auto-Selecting & Loading the First Picture

```
If ListOfPictures.ListCount > 0 Then
    ListOfPictures.ListIndex = 0 ' Select first item
    Call ListOfPictures_Click ' Load the first image automatically
End If
```

- If at least **one picture is found**, it **automatically selects** the first one.
- Calls `ListOfPictures_Click` to **display the image immediately**.

10. Updating the Picture Count

```
PicCount.Caption = "Pictures: " & ListOfPictures.ListCount
```

🔻 Displays the **number of pictures** found.

💡 Summary

✓ **Updates Labels** (`LblCaption` & `LblDes`) when a place is selected.
✓ **Finds and displays the description** from the "TouristPlace" sheet.
✓ **Looks for a sheet matching the selected place** (if available).
✓ **Loads image names** into `ListOfPictures`.
✓ **Auto-selects and displays** the first image.
✓ **Shows the total picture count** in `PicCount`.

Explanation of `ListOfPictures_Click` Event in VBA

The `ListOfPictures_Click` event **triggers when a user clicks on a picture name** in the ListBox (`ListOfPictures`).

📌 What This Code Does?

✅ Gets the **selected place** and **selected picture name** from `ComboPlace` and `ListOfPictures`.
✅ **Checks if a worksheet exists** for the selected place.
✅ **Finds the selected picture** in Column A of the sheet.
✅ **Retrieves the picture caption and description** from Columns B & C.
✅ **Loads the image** (tries `.jpg` first, then `.png`).
✅ **Updates the picture count** (e.g., "Picture 1 of 10").

1️⃣. Declaring Variables

```
Dim wsSpot As Worksheet
Dim selectedSpot As String, selectedPic As String
Dim picFolder As String, imgPath As String
Dim lastRow As Long
Dim picCell As Range
```

🔻 `wsSpot` refers to the worksheet where **images for the selected place** are stored.
🔻 `selectedSpot` stores the **place name selected** in `ComboPlace`.
🔻 `selectedPic` stores the **picture name selected** in `ListOfPictures`.
🔻 `picFolder` stores the **path where images are saved**.
🔻 `imgPath` is used to construct the **full file path for the image**.

2️⃣. Getting the Selected Place & Picture Name

```
selectedSpot = ComboPlace.Value
selectedPic = ListOfPictures.Value
```

- Retrieves the **selected place name** from `ComboPlace`.
- Retrieves the **selected picture name** from `ListOfPictures`.

3□. Setting the Image Folder Path

```
picFolder = "D:\MyClick\MyPictures\"
```

- Defines the folder where images are stored.

4□. Checking If a Sheet for the Selected Spot Exists

```
On Error Resume Next
Set ws = ThisWorkbook.Sheets(selectedSpot)
On Error GoTo 0
```

- Tries to set `ws` to the **worksheet matching the selected place**.
- **Prevents an error** if the sheet does not exist.

5□. Finding the Selected Picture in the Sheet

```
If Not ws Is Nothing Then
    lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row

    For Each picCell In ws.Range("A2:A" & lastRow)
        If picCell.Value = selectedPic Then
```

- If the sheet **exists**, find the **last row** in **Column A**.
- Loops through **each picture name** in Column A.
- **If a match is found, continue to update labels and load the image**.

6□. Updating the Caption & Description

```
LblCaption.Caption = picCell.Offset(0, 1).Value
LblDes.Caption = picCell.Offset(0, 2).Value
```

- Retrieves the **caption** from **Column B** and updates `LblCaption`.
- Retrieves the **description** from **Column C** and updates `LblDes`.

7□. Loading the Image

```
imgPath = picFolder & selectedPic & ".jpg"
If Dir(imgPath) <> "" Then
    MainImageFrame.Picture = LoadPicture(imgPath)
Else
```

```
    imgPath = picFolder & selectedPic & ".png"
    If Dir(imgPath) <> "" Then
        MainImageFrame.Picture = LoadPicture(imgPath)
    Else
        MainImageFrame.Picture = LoadPicture("") ' Clear image if not found
    End If
End If
```

- Constructs the **full file path** for `.jpg`.
- If `.jpg` exists, it **loads the image**.
- If `.jpg` **does not exist**, it tries `.png`.
- If neither `.jpg` nor `.png` exist, it **clears the image**.

8. Updating the Picture Count

```
If ListOfPictures.ListCount > 0 Then
    PicCount.Caption = "Picture " & (ListOfPictures.ListIndex + 1) & " of " &
ListOfPictures.ListCount
Else
    PicCount.Caption = "No pictures available"
End If
```

- If pictures are available, it **displays the current picture number**.
- If **no pictures are available**, it updates `PicCount` to **"No pictures available"**.

9. Cleaning Up

```
Set ws = Nothing
```

- Frees up memory by removing the worksheet reference.

Summary

✓ **Displays the selected image's caption and description** from the worksheet.
✓ **Loads the corresponding image** (checks `.jpg` first, then `.png`).
✓ **Updates the picture count** (e.g., "Picture 2 of 5").
✓ **Clears the image if no matching file is found**.

📌 Explanation of `BtnNext_Click` Event in VBA

The `BtnNext_Click` event is triggered **when the "Next" button is clicked**. It **moves to the next image** in the ListBox (`ListOfPictures`). If the last image is reached, it **loops back to the first image**.

1. Error Handling Setup

```
On Error GoTo ErrorHandler
```

- If an error occurs, execution jumps to the `ErrorHandler` section.
- This **prevents crashes** and displays a message if an issue occurs.

2. Exit If No Pictures Are Available

```
If ListOfPictures.ListCount = 0 Then Exit Sub
```

- **If the ListBox is empty**, the procedure **exits immediately** to avoid errors.

---

3. Fade Out the Current Image

```
MainImageFrame.Visible = False
DoEvents
```

- Hides the image (`MainImageFrame`) before switching pictures.
- `DoEvents` allows Excel to **process pending screen updates**, ensuring a smooth transition effect.

4. Move to the Next Picture

```
If ListOfPictures.ListIndex < ListOfPictures.ListCount - 1 Then
    ListOfPictures.ListIndex = ListOfPictures.ListIndex + 1
Else
    ListOfPictures.ListIndex = 0
End If
```

- If **not at the last image**, move to the **next** one.
- If **at the last image**, **loop back** to the first image (circular navigation).

5. Update the Display

```
Call ListOfPictures_Click
```

- Calls the `ListOfPictures_Click` event, which:

✓ Loads the **new image** from the folder.
✓ Updates the **caption and description**.
✓ Updates the **image count display**.

6. Fade In the New Image

```
MainImageFrame.Visible = True
```

🔻 **Re-displays the image** after the transition.

7□. Error Handling

```
ErrorHandler:
    MainImageFrame.Visible = True ' Ensure image stays visible
    MsgBox "Error moving to next image: " & Err.Description, vbExclamation
```

🔻 If an error occurs:

✅ Ensures the image **remains visible**.

✅ Displays a **message box** with the error description.

💡 Summary

✓ Moves to the **next picture** in the ListBox.

✓ **Loops back** to the first image when reaching the last one.

✓ **Fades out** and **fades in** images for a smoother effect.

✓ Uses **error handling** to avoid crashes.

Explanation of `BtnPrev_Click` Subroutine

This VBA procedure allows users to navigate **backward** through the image list in your Picture Viewer.

```
Private Sub BtnPrev_Click()
    On Error GoTo ErrorHandler
```

- **Error Handling:** If an error occurs, the code jumps to the `ErrorHandler` section.

```
' Exit if no pictures available
If ListOfPictures.ListCount = 0 Then Exit Sub
```

- **Prevents errors** by checking if there are any images in the ListBox.
- If the list is empty, the procedure stops immediately.

```
    ' Fade out current image
    MainImageFrame.Visible = False
    DoEvents
```

- **Hides the current image** before switching to the previous one.
- `DoEvents` allows VBA to process other tasks while executing the loop, preventing lag.

```
    ' Move to previous or loop to end
```

```
    If ListOfPictures.ListIndex > 0 Then
        ListOfPictures.ListIndex = ListOfPictures.ListIndex - 1
    Else
        ListOfPictures.ListIndex = ListOfPictures.ListCount - 1
    End If
```

- If **not at the first item**, move **one step back** (`-1`).
- If at the **first item**, loop back to the **last item** (`ListCount - 1`).
- This **circular navigation** ensures that the slideshow keeps running smoothly.

```
    ' Update display
    Call ListOfPictures_Click
```

- Calls `ListOfPictures_Click` to **load the new image**.
- This ensures the description and caption are updated along with the image.

```
    ' Fade in new image
    MainImageFrame.Visible = True
```

- **Shows the newly selected image** after it has been updated.

```
    Exit Sub

ErrorHandler:
    MasterFormImage.Visible = True ' Ensure image stays visible
    MsgBox "Error moving to previous image: " & Err.Description,
vbExclamation
```

- **If an error occurs**, the code:
  ✓ Ensures the image remains visible.
  ✓ Displays an error message with details.

📌 Key Features of This Code

✓ **Ensures smooth navigation** backward through images.
✓ **Loops back to the last image** if the first image is reached.
✓ **Includes error handling** to prevent crashes.
✓ **Hides and shows images** for a smooth transition.

Explanation of `AddToolTip` Function

This **VBA subroutine** dynamically **adds a tooltip** to a specified control in a **UserForm**. It provides **hover text** (a small popup message when you place the mouse over a control).

1. **Requires Microsoft Forms 2.0 Object Library**

- This ensures access to UI elements like `MSForms.Control`, `Label`, etc.

2. **Creates a New Label Control (Tooltip)**

```
Set tip = Me.Controls.Add("Forms.Label.1", "tip" & ctrl.Name, True)
```

   - This dynamically **creates a new Label** named `"tip" & ctrl.Name` (e.g., `tipBtnNext` for `BtnNext`).
   - The third argument `True` makes it part of the form.

3. **Configures Tooltip Appearance**

```
.Visible = False  ' Hidden by default
.AutoSize = True  ' Adjusts size based on text
.BackStyle = fmBackStyleOpaque  ' Solid background
.BackColor = RGB(255, 255, 200) ' Light yellow background
.BorderStyle = fmBorderStyleSingle ' Adds border
.BorderColor = RGB(150, 150, 150) ' Gray border
.Caption = tipText ' Sets tooltip text
.WordWrap = True ' Allows multi-line text
.Width = 200 ' Sets max width
```

   - This ensures **aesthetic tooltips** that are clearly readable.

4. **Assigns Tooltip Text to Control**

```
ctrl.ControlTipText = tipText
```

   - This is **Excel's built-in tooltip feature**, so when the user hovers over the control, it shows `tipText`.

📌 Example Usage

If you want to add a tooltip to `BtnNext`, you call:

```
AddToolTip BtnNext, "Click to view the next image."
```

👉 **Result:** When the user hovers over `BtnNext`, they see **"Click to view the next image."**

🌟 Why This is Useful?

✅ **Improves User Experience** → Users understand what each button does.
✅ **No Need for Manual Tooltips** → Auto-assigns tooltips to controls.
✅ **Customizable Look** → You can modify colors, width, etc.

📌 Summary of the VBA Code for Image Viewer in Excel

This code creates an **Excel-based image viewer** with functionalities such as navigating images, tooltips, keyboard shortcuts, and UI enhancements.

1□. Workbook & UserForm Events

◆ Workbook_Open()

- Hides the Excel application when the workbook opens.
- Displays the `MainForm` (UserForm) automatically.

◆ **UserForm_QueryClose()**

- When the form is closed, **Excel is made visible** again.
- Prevents the **"Do you want to save changes?"** prompt.

◆ **UserForm_Initialize()**

- Hides Excel and **initializes the UI** when the form loads.
- Loads **places from the "TouristPlace" sheet** into a dropdown (`ComboPlace`).
- Displays a **default image** (`HomeScreen.jpg`).
- Sets **label formatting and background color**.
- Adds **tooltips** for navigation buttons.

2□. Image Navigation (Next/Previous)

◆ **BtnNext_Click()**

- Moves to the **next image** in the list.
- If it's the **last image**, it loops back to the **first image**.
- Uses `DoEvents` for **smooth image transition**.

◆ **BtnPrev_Click()**

- Moves to the **previous image** in the list.
- If it's the **first image**, it loops back to the **last image**.

3□. Image Display & List Handling

◆ **ListOfPictures_Click()**

- Loads the selected **image** from `D:\MyClick\MyPictures\`.
- Fetches **caption and description** from a sheet named after the selected place.
- Tries to load **both `.jpg` and `.png` formats**.

- Updates **image count label**.

◆ **ComboPlace_Change()**

- Populates `ListOfPictures` with images from the **selected place's sheet**.
- Updates **image description** from `TouristPlace` sheet.
- Automatically **selects and displays** the first image.

4□. Form Closing & Visibility

◆ **CloseMainForm_Click() / MasterFormExit_Click()**

- Closes the form and **restores Excel visibility**.

5□. Keyboard Shortcuts

◆ **UserForm_KeyDown()**

- **Arrow Keys (←, ↑, →, ↓)** → Navigate images.
- **Spacebar (␣)** → Start slideshow.
- **Escape (`Esc`)** → Close the form.

6□. Tooltips for UI Enhancements

◆ **AddToolTip()**

- Dynamically **creates a tooltip label** for any control.
- Uses **light yellow background** and **border styling**.

🌟 Key Features & Benefits

✅ **Excel stays hidden** for a cleaner UI experience.
✅ **Images are loaded dynamically** based on the selected place.
✅ **Keyboard shortcuts** improve usability.
✅ **Looping navigation** prevents reaching a dead end.
✅ **Tooltips enhance user guidance**.

BEFORE COPYING AND PASTING THE CODE, MAKE SURE TO COMPLETE ALL THE STEPS MENTIONED EARLIER!

Full Code of Main Form: Copy and Paste

Steps to Paste Code into Another Form:

1. Press **`ALT + F11`** to open the **Visual Basic Editor**.
2. Press **`CTRL + R`** to open **Project Explorer** (if not visible).
3. Find the form where you want to paste the code.
4. Double-click the form to open its **code window**.
5. Press **`CTRL + V`** to paste the copied code.

```
Private Sub Workbook_Open()
   Application.Visible = False
   MainForm.Show
End Sub

Private Sub CloseMainForm_Click()
  Unload Me ' Close the form
  Application.Visible = True
End Sub

Private Sub MasterFormExit_Click()
 Unload Me ' Close the form
 Application.Visible = True
End Sub

Private Sub CmdAdvance_Click()
Unload Me
SliderFormWithMenu.Show
End Sub

Private Sub PicView2_Click()
Unload Me
SliderForm.Show
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
   ' Show Excel when form is closed
   Application.Visible = True ' Show Excel before closing
   ThisWorkbook.Saved = True  ' Prevent save prompt
End Sub

Private Sub BtnNext_Click()
On Error GoTo ErrorHandler

   ' Exit if no pictures available
   If ListOfPictures.ListCount = 0 Then Exit Sub

   ' Fade out current image
   MainImageFrame.Visible = False
```

```
    DoEvents

    ' Move to next or loop to beginning
    If ListOfPictures.ListIndex < ListOfPictures.ListCount - 1 Then
        ListOfPictures.ListIndex = ListOfPictures.ListIndex + 1
    Else
        ListOfPictures.ListIndex = 0
    End If

    ' Update display
    Call ListOfPictures_Click

    ' Fade in new image
    MainImageFrame.Visible = True

    Exit Sub

ErrorHandler:
    MainImageFrame.Visible = True ' Ensure image stays visible
    MsgBox "Error moving to next image: " & Err.Description, vbExclamation
End Sub

Private Sub BtnPrev_Click()
  On Error GoTo ErrorHandler

    ' Exit if no pictures available
    If ListOfPictures.ListCount = 0 Then Exit Sub

    ' Fade out current image
    MainImageFrame.Visible = False
    DoEvents

    ' Move to previous or loop to end
    If ListOfPictures.ListIndex > 0 Then
        ListOfPictures.ListIndex = ListOfPictures.ListIndex - 1
    Else
        ListOfPictures.ListIndex = ListOfPictures.ListCount - 1
    End If

    ' Update display
    Call ListOfPictures_Click

    ' Fade in new image
    MainImageFrame.Visible = True
```

```
    Exit Sub

ErrorHandler:
    MasterFormImage.Visible = True ' Ensure image stays visible
    MsgBox "Error moving to previous image: " & Err.Description, vbExclamation
End Sub

Private Sub ComboPlace_Change()
Dim ws As Worksheet, wsSpot As Worksheet
    Dim selectedSpot As String
    Dim lastRow As Long, lastRow1 As Long
    Dim spotCell As Range, picCell As Range
    Dim imgPath As String, picFolder As String

    ' Define the worksheet where tourist spots are listed
    Set ws = ThisWorkbook.Sheets("TouristPlace")

    ' Get the selected spot name
    selectedSpot = ComboPlace.Value

    ' Update the "SpotName" label
    LblCaption.Caption = "Welcome to " & selectedSpot

    ' Find the selected spot in Column A of ComboMenu and get its description from Column B
    lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row
    For Each spotCell In ws.Range("A2:A" & lastRow)
        If spotCell.Value = selectedSpot Then
            LblDes.Caption = spotCell.Offset(0, 1).Value ' Get description from Column B
            Exit For
        End If
    Next spotCell

    ' Define the folder path where images are stored
    picFolder = "D:\MyClick\MyPictures\"


    ' Set the image path based on naming convention (Caption + Selected Spot)
    imgPath = picFolder & "Caption" & selectedSpot ' No need to add .jpg here as we'll try both extensions


    ' Clear the ListBox to remove the previous pictures
    ListOfPictures.Clear
```

```
    ' Check if a sheet exists with the selected spot name
    On Error Resume Next
    Set wsSpot = ThisWorkbook.Sheets(selectedSpot)
    On Error GoTo 0

    ' If sheet exists, populate the ListBox with Picture Names from Column A
    If Not wsSpot Is Nothing Then
        lastRow1 = wsSpot.Cells(wsSpot.Rows.Count, 1).End(xlUp).Row
        For Each picCell In wsSpot.Range("A2:A" & lastRow1)
            If picCell.Value <> "" Then
                ListOfPictures.AddItem picCell.Value ' Add Picture Name from Column A
            End If
        Next picCell
    End If

    ' Auto-select first picture in the list if available
If ListOfPictures.ListCount > 0 Then
    ListOfPictures.ListIndex = 0 ' Select first item
    Call ListOfPictures_Click ' Load the first image automatically
End If

' Update the picture count label
    PicCount.Caption = "Pictures: " & ListOfPictures.ListCount
End Sub

Private Sub ListOfPictures_Click()
 Dim wsSpot As Worksheet
    Dim selectedSpot As String, selectedPic As String
    Dim picFolder As String, imgPath As String
    Dim lastRow As Long
    Dim picCell As Range

    ' Get selected spot and picture name
    selectedSpot = ComboPlace.Value
    selectedPic = ListOfPictures.Value

    ' Define image folder path
    picFolder = "D:\MyClick\MyPictures\"

    ' Check if the corresponding sheet exists
    On Error Resume Next
    Set ws = ThisWorkbook.Sheets(selectedSpot)
    On Error GoTo 0
```

```
    If Not ws Is Nothing Then
        lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row

        ' Find the selected picture in Column A
        For Each picCell In ws.Range("A2:A" & lastRow)
            If picCell.Value = selectedPic Then
                ' Get Caption from Column B ("PictureCaption")
                LblCaption.Caption = picCell.Offset(0, 1).Value

                ' Get Description from Column C ("PictureDescription")
                LblDes.Caption = picCell.Offset(0, 2).Value

                ' Load Image (Picture Name is stored in Column A)
                imgPath = picFolder & selectedPic ' Do not add .jpg or .png here

               ' Check for .jpg first
               imgPath = picFolder & selectedPic & ".jpg"
               If Dir(imgPath) <> "" Then
                  MainImageFrame.Picture = LoadPicture(imgPath)
               Else
                  ' If .jpg not found, check for .bmp
                  imgPath = picFolder & selectedPic & ".png"
                     If Dir(imgPath) <> "" Then
                        MainImageFrame.Picture = LoadPicture(imgPath)
                     Else
                      ' Clear image if neither .jpg nor .bmp is found
                       MainImageFrame.Picture = LoadPicture("")
                     End If
            End If
            End If
        Next picCell
    End If

    ' Update the picture count and current selection
    If ListOfPictures.ListCount > 0 Then
        PicCount.Caption = "Picture " & (ListOfPictures.ListIndex + 1) & " of " & ListOfPictures.ListCount
    Else
        PicCount.Caption = "No pictures available"
    End If

    Set ws = Nothing
End Sub

Private Sub UserForm_Initialize()
```

```
    Dim ws As Worksheet
    Dim lastRow As Long
    Dim rng As Range
    Dim cell As Range
    Dim defaultImgPath As String
    Dim picFolder As String

    ' Hide Excel application when form loads
    Application.Visible = False

    ' Set worksheet reference
    Set ws = ThisWorkbook.Sheets("TouristPlace")

    ' Find last row in column A (avoiding blank spaces)
    lastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row

    ' Ensure only non-empty values are added
    ComboPlace.Clear
    For Each cell In ws.Range("A2:A" & lastRow)
        If Trim(cell.Value) <> "" Then ' Avoid adding blank cells
            ComboPlace.AddItem cell.Value
        End If
    Next cell

    ' Set Picture Size Mode to Zoom (so the image fits well)
    MainImageFrame.PictureSizeMode = fmPictureSizeModeZoom

    ' Define the folder path where images are stored
    picFolder = "D:\MyClick\MyPictures\"

    ' Define the default image path
    defaultImgPath = picFolder & "HomeScreen.jpg"

    ' Load the default image if it exists
    If Dir(defaultImgPath) <> "" Then
        MainImageFrame.Picture = LoadPicture(defaultImgPath)
    Else
        MainImageFrame.Picture = LoadPicture("") ' Clear image if not found
    End If

    ' Center align text for SpotName and SpotDescription labels
    LblCaption.Caption = "Tamilnadu - Gateway to South India"
    LblDes.Caption = "Tamil Nadu tourism offers a diverse mix of heritage, culture, spirituality, and nature.
Known for its ancient temples, pristine beaches, hill stations like Ooty and Kodaikanal, and vibrant
```

festivals, the state attracts travelers seeking history and natural beauty. Key attractions include the UNESCO-listed Brihadeeswarar Temple, Meenakshi Amman Temple, Marina Beach, and the Nilgiri Mountains. Tamil Nadu also boasts rich wildlife, traditional arts, and delicious South Indian cuisine, making it a must-visit destination."

```
    LblCaption.TextAlign = fmTextAlignCenter
    LblDes.TextAlign = fmTextAlignCenter
    LblCaption.Font.Size = 16
    LblCaption.Font.Bold = True
    LblDes.Font.Size = 14
    LblDes.ForeColor = RGB(50, 50, 50) ' Dark gray for better readability
    Me.BackColor = RGB(240, 240, 200) ' Light yellow background for a warm feel
    Label3.Font.Bold = True
    'Label3.ForeColor = RGB(255, 69, 0) ' Orange for impact

     ' Set Picture Size Mode to Stretch
    MainImageFrame.PictureSizeMode = fmPictureSizeModeStretch

    Label3.Caption = "Presented by: Gautam Banerjee"
    Label3.TextAlign = fmTextAlignCenter

    ' Initialize the picture count label
    PicCount.Caption = "Select a place!"

    ' Add tooltips
    AddToolTip BtnPrev, "Click to View previous image"
    AddToolTip BtnNext, "Click to View next image"

End Sub

Private Sub AddToolTip(ctrl As MSForms.Control, tipText As String)
    ' Requires reference to "Microsoft Forms 2.0 Object Library"
    Dim tip As MSForms.Control
    Set tip = Me.Controls.Add("Forms.Label.1", "tip" & ctrl.Name, True)

    With tip
        .Visible = False
        .AutoSize = True
        .BackStyle = fmBackStyleOpaque
        .BackColor = RGB(255, 255, 200)
        .BorderStyle = fmBorderStyleSingle
        .BorderColor = RGB(150, 150, 150)
        .Caption = tipText
        .WordWrap = True
```

```
        .Width = 200
    End With

    ctrl.ControlTipText = tipText
End Sub
```

Second Picture Form : SliderForm

Steps to Create the Another UserForm:

1. **Open the VBA Editor** (**ALT + F11**)
2. **Insert a New UserForm** (**Insert > UserForm**)
3. **Rename the UserForm to " SliderForm" from the Properties window**
4. **Design the Form:**
    - Add **Image Control (`ImageSlider`)**
    - Add **Buttons (`cmdPrev, cmdNext, cmdExit`)**
    - **Add Labels or other object (Optional)**
    - **Don't change the above objects name**

Copy the Full Code:

```
Option Explicit

' Declare module-level variables
Private ImageFiles() As String
Private CurrentImageIndex As Long
Private ImageCount As Long

' Constants
Private Const IMAGE_FOLDER As String = "D:\MyClick\MyPictures\"

' Initialize the form
Private Sub UserForm_Initialize()
    ' Load all image files from the folder
    LoadImageFiles

    ' Set initial state
    CurrentImageIndex = -1
    UpdateButtons

    ' Set image control properties
    With ImageSlider
        .PictureSizeMode = fmPictureSizeModeStretch
        .BorderStyle = fmBorderStyleNone
    End With
```

```
    ' Show first image if available
    If ImageCount > 0 Then
        CurrentImageIndex = 0
        ShowCurrentImage
    End If
End Sub

' Load all JPG images from the folder
Private Sub LoadImageFiles()
    Dim FileName As String
    Dim i As Long

    ' Count JPG files
    FileName = Dir(IMAGE_FOLDER & "*.jpg")
    ImageCount = 0

    While FileName <> ""
        ImageCount = ImageCount + 1
        FileName = Dir()
    Wend

    ' Redim array to hold file names
    If ImageCount > 0 Then
        ReDim ImageFiles(0 To ImageCount - 1)

        ' Store file names
        FileName = Dir(IMAGE_FOLDER & "*.jpg")
        i = 0

        While FileName <> ""
            ImageFiles(i) = FileName
            i = i + 1
            FileName = Dir()
        Wend
    Else
        MsgBox "No JPG images found in the specified folder.", vbExclamation
        cmdPrev.Enabled = False
        cmdNext.Enabled = False
    End If
End Sub

' Show current image
Private Sub ShowCurrentImage()
```

```
    On Error Resume Next ' In case image can't be loaded
    ImageSlider.Picture = LoadPicture(IMAGE_FOLDER & ImageFiles(CurrentImageIndex))
    On Error GoTo 0
    LblPictureInfo.Caption = "File: D:\MyClick\MyPictures\" & ImageFiles(CurrentImageIndex) & vbCrLf
    UpdateButtons
End Sub

' Update button states based on current position
Private Sub UpdateButtons()
    cmdPrev.Enabled = (CurrentImageIndex > 0)
    cmdNext.Enabled = (CurrentImageIndex < ImageCount - 1)
End Sub

' Show previous image
Private Sub cmdPrev_Click()
    If CurrentImageIndex > 0 Then
        CurrentImageIndex = CurrentImageIndex - 1
        ShowCurrentImage
    End If
End Sub

' Show next image
Private Sub cmdNext_Click()
    If CurrentImageIndex < ImageCount - 1 Then
        CurrentImageIndex = CurrentImageIndex + 1
        ShowCurrentImage
    End If
End Sub

' Exit the form
Private Sub cmdExit_Click()
    Unload Me
    MainForm.Show
End Sub
```

You can change the picture folder name, but you must update the code accordingly:

```
Private Const IMAGE_FOLDER As String = "Your Folder Name"
```

3rd. Picture Form : SliderForm

Steps to Create the Another UserForm:

1. **Open the VBA Editor** (**ALT + F11**)
2. **Insert a New UserForm** (**Insert > UserForm**)

3. **Rename the UserForm to "*SliderFormWithMenu*" from the Properties window**
4. **Design the Form:**
    - Add **Image Control (`ImageSlider`)**
    - Add **Buttons (`cmdPrev, cmdNext, cmdExit`)**
    - **Add Labels or other object (Optional)**
    - **Don't change the above objects name**

Copy the Full Code:

```
Option Explicit

' Declare module-level variables
Private ImageFiles() As String
Private CurrentImageIndex As Long
Private ImageCount As Long
Private SelectedFolder As String ' Store user-selected folder

' Initialize the form
Private Sub UserForm_Initialize()
    ' Ask user to select a folder
    If Not SelectImageFolder Then
        MsgBox "No folder selected. Exiting...", vbExclamation
        Unload Me
        Exit Sub
    End If

    ' Load images from selected folder
    LoadImageFiles

    ' Set initial state
    CurrentImageIndex = -1
    UpdateButtons

    ' Set image control properties
    With ImageSlider
        .PictureSizeMode = fmPictureSizeModeZoom
        .BorderStyle = fmBorderStyleNone
    End With

    ' Show first image if available
    If ImageCount > 0 Then
        CurrentImageIndex = 0
        ShowCurrentImage
    End If
End Sub
```

```
Private Function SelectImageFolder() As Boolean
    Dim FileDialog As Object
    Set FileDialog = Application.FileDialog(3) ' msoFileDialogFilePicker

    With FileDialog
        .Title = "Select an Image File"
        .Filters.Clear
        .Filters.Add "JPG Images", "*.jpg"
        .Filters.Add "PNG Images", "*.png"
        .AllowMultiSelect = False

        If .Show = -1 Then
            ' Extract the folder from the selected file path
            SelectedFolder = Left(.SelectedItems(1), InStrRev(.SelectedItems(1), "\"))
            SelectImageFolder = True
        Else
            SelectImageFolder = False
        End If
    End With

    Set FileDialog = Nothing
End Function
' Load all JPG images from the folder
Private Sub LoadImageFiles()
    Dim FileName As String
    Dim i As Long

    ' Ensure folder is selected
    If SelectedFolder = "" Then Exit Sub

    ' Count JPG files
    FileName = Dir(SelectedFolder & "*.jpg")
    ImageCount = 0

    While FileName <> ""
        ImageCount = ImageCount + 1
        FileName = Dir()
    Wend

    ' Redim array to hold file names
    If ImageCount > 0 Then
        ReDim ImageFiles(0 To ImageCount - 1)
```

```
        ' Store file names
        FileName = Dir(SelectedFolder & "*.jpg")
        i = 0

        While FileName <> ""
            ImageFiles(i) = FileName
            i = i + 1
            FileName = Dir()
        Wend
    Else
        MsgBox "No JPG images found in the selected folder.", vbExclamation
        cmdPrev.Enabled = False
        cmdNext.Enabled = False
    End If
End Sub

' Show current image
Private Sub ShowCurrentImage()
    On Error Resume Next ' In case image can't be loaded
    ImageSlider.Picture = LoadPicture(SelectedFolder & ImageFiles(CurrentImageIndex))
    On Error GoTo 0

    ' Display file information in LblPictureInfo
    LblPictureInfo.Caption = "File: " & ImageFiles(CurrentImageIndex) & vbCrLf & _
                             "Location: " & SelectedFolder

    UpdateButtons
End Sub

' Update button states
Private Sub UpdateButtons()
    cmdPrev.Enabled = (CurrentImageIndex > 0)
    cmdNext.Enabled = (CurrentImageIndex < ImageCount - 1)
End Sub

' Show previous image
Private Sub cmdPrev_Click()
    If CurrentImageIndex > 0 Then
        CurrentImageIndex = CurrentImageIndex - 1
        ShowCurrentImage
    End If
End Sub

' Show next image
```

```
Private Sub cmdNext_Click()
    If CurrentImageIndex < ImageCount - 1 Then
        CurrentImageIndex = CurrentImageIndex + 1
        ShowCurrentImage
    End If
End Sub

' Exit the form
Private Sub cmdExit_Click()
    Unload Me
    MainForm.Show
End Sub
```

If you face any difficulties running the program, you can ask in the comment box.

Thanks

Gautam Banerjee

E-mail: gincom1@yahoo.com

For Donation: Mobile No. 97483 27614