# Slideshow Application

Designed by Peter McGuinness

# Index

## 1. Requirements Analysis (Planning Phase)

**Goal:** Define what the application should do and identify technical constraints.

**Functional Requirements:**

- Load and display images from the user’s system.
- Navigate between images (Next, Previous).
- Play and pause a slideshow.
- Display thumbnails for easy selection.
- Toggle fullscreen mode.
- Allow users to add images dynamically.

**Non-Functional Requirements:**

- Responsive UI with smooth transitions.
- Efficient image handling (preloading, caching).
- Minimal memory usage.
- Cross-platform compatibility (Windows, macOS, Linux).

**Technology Stack:**

- **Programming Language:** Python

- **Libraries:**
    - Tkinter (GUI)
    - PIL (Image handling)
    - OS, filedialog (File selection)

---

## 2. System Design (Architectural Phase)

**Goal:** Define how components interact and structure the code.

**Design Components:**

- **GUI Layout:**
    - Main Frame (Houses all UI elements).
    - Thumbnail Panel (Scrollable, contains mini previews).
    - Image Display Canvas (Shows the main image).
    - Control Panel (Buttons: Play, Pause, Next, Previous, Upload).
- **Data Flow:**
    - Load images → Store paths → Generate thumbnails → Display selected image.
    - Slideshow Timer → Cycles through images automatically.

- **Event Handling:**
    - Button Clicks (Load image, Next, Previous).
    - Keyboard Shortcuts (Fullscreen toggle, Exit).

---

## 3. Implementation (Development Phase)

**Goal:** Write the code based on the design plan.

**Development Tasks:**

✅ Set up Tkinter main window and UI elements.
✅ Implement image loading and thumbnail generation.
✅ Implement navigation (Next, Previous, Thumbnail selection).
✅ Add slideshow functionality (Auto-switching images).
✅ Implement fullscreen mode and keyboard shortcuts.
✅ Optimize memory management (Dispose of unused images).

---

## 4. Testing (Quality Assurance Phase)

**Goal:** Ensure the application is bug-free and works as expected.

**Testing Methods:**

- **Unit Testing:** Test functions like `upload_images()`, `show_image()`.
- **GUI Testing:** Ensure buttons and keyboard shortcuts work correctly.

- **Performance Testing:** Measure memory usage and image loading speed.
- **User Acceptance Testing (UAT):** Ensure UI is intuitive.

**Test Cases:**

- ✅ Upload images and check if they appear in the list.
- ✅ Click “Next” and “Previous” to navigate images.
- ✅ Play slideshow and check if images transition.
- ✅ Test fullscreen mode toggle (F11, Escape).
- ✅ Try uploading unsupported file types (e.g., `.txt`).

---

## 5. Deployment (Release Phase)

**Goal:** Package and distribute the application for use.

**Deployment Tasks:**

- Convert the script into an **executable** (`.exe` for Windows, `.app` for macOS).
- Package dependencies using **PyInstaller** or **cx_Freeze**.
- Create a README with installation instructions.
- Distribute via GitHub, a website, or an app store.

---

## 6. Maintenance (Support Phase)

**Goal:** Provide updates, fix bugs, and improve performance.

**Ongoing Tasks:**

- Fix reported bugs.
- Improve UI/UX (e.g., add transitions, drag-and-drop image loading).
- Optimize performance (e.g., reduce memory consumption).
- Add new features (e.g., background music, captions).

---

## Assets Needed:

📂 **Code Assets:** Python script, UI elements, dependencies.
🖼️ **Image Assets:** Sample images for testing.
📖 **Documentation:** User guide, developer notes.