Google Photos Navigation POC - Proof of Concept

Project Purpose

Build a **simple test application** to validate that we can reliably automate navigation in Google Photos by clicking on the right side of the photo to advance to the next photo.

Why This POC?

Before investing time in building a full tagging assistant application, we need to verify the core assumption: **Can we programmatically click the right side of the photo area in Google Photos to navigate between photos?**

This POC will answer that question in 1-2 hours instead of discovering problems after days of development.

POC Scope (Minimal Viable Test)

What It Does:

1. Opens Chrome/Edge browser to photos.google.com
2. Pauses for user to manually:
   - Sign into Google Photos
   - Navigate to any photo
   - Click the (i) button to open info panel (optional, just for visual reference)
3. Displays simple UI window with "NEXT" button
4. When user clicks "NEXT":
   - App locates the photo display area in Google Photos
   - Clicks on the right side of the photo (to advance)
   - Waits 2-3 seconds for photo to load
   - Ready for next click
5. Counter shows how many photos navigated

What It Does NOT Do:

- ❌No description reading or writing
- ❌No name management
- ❌No data persistence
- ❌No progress tracking
- ❌No error recovery
- ❌No configuration or settings

This is purely a navigation test.

Technical Requirements

Platform

- Python 3.8+
- **GUI Framework Options** (cross-platform Windows 11 & macOS):
  - **PyQt5/PyQt6** - Modern, polished look (recommended)
  - **Kivy** - Modern, good cross-platform support
  - **Tkinter** - Built into Python, simpler but basic look
  - **Flask** - Web-based local app option (browser-based UI)
- **Browser Automation**: Selenium or Playwright
- **Target OS**: Windows 11 and macOS

Recommended: PyQt or Kivy for modern appearance and best cross-platform compatibility.

Two-Window System

Window 1: Test UI (User Control)

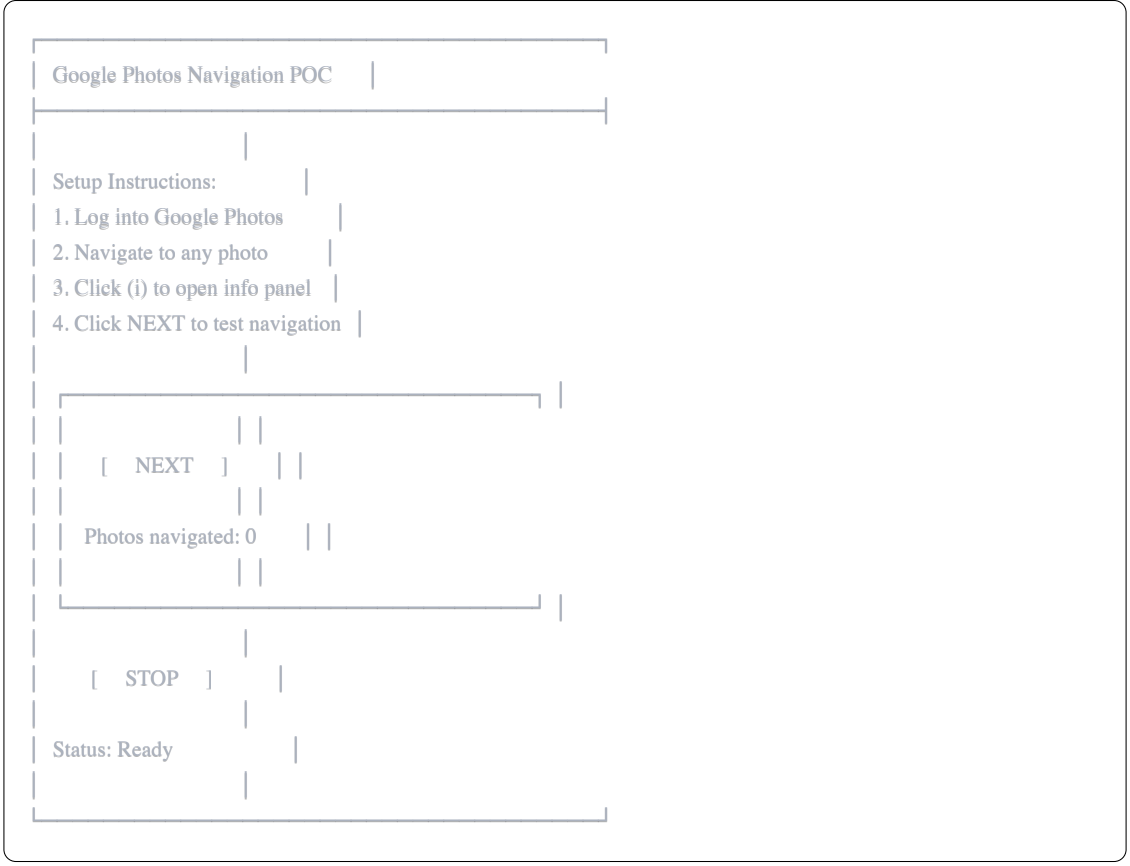
- Simple window with one button
- Shows click counter
- User clicks "NEXT" to test navigation

Window 2: Browser (Google Photos)

- Automated Chrome/Edge window
- User logs in manually
- User navigates to any photo manually
- App will click the > button when user clicks "NEXT"

User Interface

Test UI Window (Simple)



How To Use (Test Process)

Step 1: Start the App

- Run the Python script
- Browser window opens to photos.google.com
- Test UI window appears

Step 2: Manual Setup

- In browser window: Log into Google Photos
- Navigate to any photo (any date, any album)
- Optionally click (i) to open info panel
- Return to Test UI window

Step 3: Test Navigation

- Click "NEXT" button in UI
- Observe: Does photo advance in browser?
- Click "NEXT" again
- Repeat 20-30 times

Step 4: Validation

Success criteria:

- ✅> button is found and clicked each time
- ✅Photo advances to next photo reliably
- ✅Works consistently across 20-30+ clicks
- ✅No errors or failures
- ✅Info panel stays open (if opened)

Failure indicators:

- ❌Can't find > button
- ❌Clicks wrong element
- ❌Photos don't advance
- ❌Errors after several clicks
- ❌Unreliable behavior

Step 5: Stop

- Click "STOP" button to close browser and exit

Implementation Details

Browser Automation

- Launch Chrome/Edge in visible mode (headful)
- Navigate to photos.google.com
- Wait for user to complete manual setup
- Locate photo display area/container
- Click on right side of photo area (approximately 75% from left edge)
- Simple 2-3 second wait after click

Error Handling (Minimal)

- If photo area not found: Display error in status
- If click fails: Display error in status
- User must manually troubleshoot (POC only)

No Data Storage

- Counter resets each run
- No persistence needed
- No configuration files

Success Criteria

POC Passes If:

1. Can navigate through 30+ photos without failure
2. Photo area is found reliably every time
3. Clicking right side advances to next photo consistently
4. Navigation feels smooth and consistent
5. No manual intervention needed during test
6. Works on both Windows and macOS (if testing both)

POC Fails If:

- Can't locate photo area consistently
- Clicks don't register or advance photos
- Navigation is unreliable
- Requires constant manual fixes
- Google Photos structure prevents automation

Next Steps Based On Results

If POC Succeeds:

✅Core assumption validated
✅Proceed with full application development
✅Use same navigation logic in final app
✅Confidence that project is viable

If POC Fails:

❌Investigate why navigation failed
❌Try alternative selectors or approaches
❌May need to reconsider project approach
❌Saves time before building complete app

Deliverables

What I Need:

1. **Python script** (poc_navigation_test.py)
   - Simple, well-commented code
   - ~100-200 lines maximum
   - Using PyQt, Kivy, Tkinter, or Flask (developer's choice - PyQt/Kivy preferred)
2. **Setup instructions**
   - How to install Python dependencies
   - How to install ChromeDriver/WebDriver
   - GUI framework installation (if needed)
   - Step-by-step for Windows 11 and macOS
3. **Quick start guide**
   - How to run the script
   - What to expect
   - How to interpret results

Optional: If offering multiple framework options, provide a starter template recommendation based on ease of setup and modern appearance.

What I'll Provide:

- Feedback on whether navigation works reliably
- Any error messages or issues encountered
- Decision on whether to proceed with full app
- Preference on GUI framework for full application (based on POC experience)

Timeline

- **POC Development**: 1-2 hours
- **Testing**: 15-30 minutes
- **Decision**: Immediate (proceed or pivot)

Technical Notes

Browser Element to Click

Easier approach: Click anywhere on the right side of the photo area to advance:

- Right half of photo = next photo
- Left half of photo = previous photo
- No need to find specific button element
- Much simpler and more reliable
- Just identify photo container and click right side

Simple Wait Strategy

After clicking >:

```python
time.sleep(2.5)  # Simple fixed wait
```

No need for complex waits in POC.

UI Button Handler

```python
def on_next_click():
    # Find and click > button in browser
    # Increment counter
    # Update status
```

Questions This POC Answers

1. ✅Can we find the photo display area programmatically?
2. ✅Can we click on the right side to advance reliably?
3. ✅Does Google Photos respond to automated clicks on the photo area?
4. ✅Is navigation consistent across multiple photos?
5. ✅Are there any rate limits or bot detection for navigation?

After POC: We'll know if the full project is viable or needs a different approach.

Framework Selection Guidance

GUI Framework Comparison

PyQt5/PyQt6 ✅Recommended

- Modern, professional appearance
- Excellent cross-platform support
- Well-documented
- Industry standard for desktop apps
- Slightly more complex setup

Kivy ✅Good Alternative

- Modern, mobile-inspired UI
- Great cross-platform support
- Good for touch interfaces
- Different look and feel

Tkinter ⚠️Simple Option

- Built into Python (no extra install)
- Basic appearance
- Easiest to set up
- Limited styling options

Flask (Web-based) 🌐Alternative Approach

- Browser-based UI (HTML/CSS/JS)
- Familiar web technologies
- Easy to make look modern
- Runs locally, accessed via browser

Developer's Choice: Use PyQt or Kivy for best results. If simplicity is priority, Tkinter works. If you prefer web technologies, Flask is viable.

Ready to build this POC first before proceeding to the full application.