***LOGS – Photo Catalog***

**Software Deployment Plan**

***[Latest update: 2025-10-20*** *(keep updated when you make changes)****]***

**1. System Requirements**

**Supported Operating Systems:**

* Linux: Ubuntu 20.04 LTS and openSUSE Leap 15 (or later)
* Windows: Windows 10 and Windows 11

**Programming Language:**

* **Python 3.12**

**Required Hardware:**

* CPU: Dual-core processor (2.0 GHz or higher)
* Memory: Minimum 4 GB RAM
* Storage: 500 MB free space
* Display: Minimum resolution 1280×720

**Software Dependencies:**

* SQLite 3.50.4 (bundled with Python)
* PyQt6 (6.9.1) – GUI framework
* Standard Python libraries: os, sqlite3, datetime, typing
* Package manager: pip

All dependencies are installed automatically during setup.

**2. Deployment Strategy Summary**

The Photo Catalog Application will be deployed once, as a final packaged desktop application for Linux and Windows.

The deployment will include:

* A source distribution (.zip) for transparency and developer access.
* A pre-built executable for end users, generated using PyInstaller.

Tools and Methods:

* Python 3.12 for runtime and scripts
* pip for dependency installation
* PyInstaller for packaging into executable form

**3. Installation Package Contents**

***3.1 Required source or compiled files***

main.py – Application entry point

DatabaseManager.py – Handles SQLite storage

FileScanner.py – Scans image directories

PhotoImporter.py – Imports photos and metadata

SearchEngine.py – Search by title/tags

SimilaritySearch.py – Finds related photos

***3.2 Required third-party components***

PyQt6 (6.9.1)

SQLite (3.50.4)

PyInstaller (6.1)

***3.4 Required graphical assets, configuration and other non-program files***

assets/icons/ – Interface icons

assets/images/ – Default placeholders

config/settings.json – Configuration settings

***3.5 Documentation files to be provided***

README.md – Setup and usage guide

UserManual.pdf – End-user instructions

***3.6 Development files and components that must be excluded***

.git/, .gitignore

\_\_pycache\_\_/ folders

Developer test databases or log files

Temporary or debug scripts

**4. Additional Code Required for Deployment**

*[List any additional scripts or programs that you will need to create in order to do the deployment. These might include shell or batch scripts, SQL scripts, executables... really depends on your strategy.]*

|  |  |
| --- | --- |
| **File** | **Purpose** |
| install\_dependencies.sh | Installs Python dependencies (Linux) |
| install\_dependencies.bat | Installs Python dependencies (Windows) |
| build\_executable.py | Runs PyInstaller to generate final build |
| init\_database.sql | Creates database schema if missing |
| run\_app.sh / run\_app.bat | Simplified startup scripts |

**5. Deployment Tasks**

*[Make a detailed, ordered list of all the tasks your team will need to do in order to package and deploy your project. This can be used as a checklist when you are actually working on the deployment.]*

|  |  |  |
| --- | --- | --- |
| Step | Task | Description |
| 1 | Prepare Environment | Ensure Python 3.12 and pip are installed on target systems. |
| 2 | Obtain Package | Download or unzip the final release folder. |
| 3 | Install Dependencies | Run install script for your OS. |
| 4 | Initialize Database | First launch creates photo\_catalog.db automatically. |
| 5 | Build Executable | Run python build\_executable.py |
| 6 | Verify Build | Confirm executable runs on both Linux and Windows. |
| 7 | Package Distribution Kit | Bundle executable + docs into final ZIP. |
| 8 | Test Deployment | Execute deployment test plan. |
| 9 | Submit Final Package | Upload completed package and documentation to Canvas. |

**6. Deployment Test Plan**

*[How will you determine that your deployment is working as required, on different platforms? This should be a list of test cases.]*

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Objective** | **Input** | **Expected Result** |
| D-01 | Verify environment | python --version | Python 3.12 displayed |
| D-02 | Install dependencies | Run setup script | Packages install correctly |
| D-03 | Launch application | Run executable | GUI opens successfully |
| D-04 | Create database | Launch app (first time) | photo\_catalog.db created |
| D-05 | File scanning | Select image folder | Images listed |
| D-06 | Add photo | Import valid photo | Added and saved in DB |
| D-07 | Duplicate photo | Add existing file | Warning message displayed |
| D-08 | Search photos | Use title or tag | Correct results shown |
| D-09 | Similarity search | Select tagged image | Related photos listed |
| D-10 | Edit metadata | Update tags | Changes saved to DB |
| D-11 | Remove photo | Delete entry | Record deleted |
| D-12 | End-to-end test | Add → Search → Edit → Remove | Full workflow works correctly |
| D-13 | Platform consistency | Test on Linux & Windows | Same behavior observed |