<Company Name>

ShutterSort
Software Architecture Document

Version 0.1

Revision History

Date	Version	Description	Author
11/10/2023	0.1	Describes the software architecture for ShutterSort.	Riley Meyerkorth

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

The primary purpose of this Software Architecture Document is to provide a detailed representation of the architecture of the ShutterSort application. It aims to capture and communicate the significant architectural decisions made during the system's development. This document will:

- Detail the high-level structure of ShutterSort, illustrating the system's software architecture through different views.
- Serve as a guide for the development team, ensuring a shared understanding of the architectural vision and direction.
- Act as a reference point for future maintenance and scalability of the system.

The intended audience for this document includes:

- **Development Team**: To understand the high-level architecture and how different components interact.
- **Project Managers**: To grasp the architectural strategies and align them with project timelines and resources.
- **Quality Assurance Team:** To use architectural insights for testing and validation purposes.
- **Future Contributors:** To provide a clear understanding of the system for future development enhancement efforts.

1.2 Scope

The scope of this Software Architecture Document encompasses the ShutterSort application in its entirety. It covers the structural design, major components, and the interactions between these components that collectively define the system's architecture.

1.3 Definitions, Acronyms, and Abbreviations

- **ShutterSort:** The mentioned software for automatically sorting photos.
- **User Interface (UI):** The space where interactions between the human user and the machine occur.
- **Machine Learning (ML):** A field of computer science that uses statistical techniques to give computer systems the ability to "learn" from data.

This list is not exhaustive and will be expanded as needed throughout the development of ShutterSort.

1.4 References

- GitHub Repository
  - Title: ShutterSort
  - Date: September 24th, 2023
  - Publishing Organization: CRRANkS
  - Source: GitHub
- PMP
  - Title: Project Management Plan
  - Date: September 24th, 2023
  - Publishing Organization: CRRANkS
  - Source: GitHub Repository, Canvas group page
- SRS
  - Title: Software Requirements Specifications
  - Date: October 13th, 2023
  - Publishing Organization: CRRANkS
  - Source: GitHub Repository, Canvas group page

1.5 Overview

This Software Architecture Document (SAD) for the ShutterSort application is organized into distinct sections, each aimed at providing a comprehensive understanding of the software's architecture. The document is structured as follows:

1. Introduction
2. Architectural Representation
3. Architectural Goals and Constraints
4. Use-Case View
5. Logical View
6. Interface Description
7. Size and Performance
8. Quality

This structured approach ensures that the SAD provides a thorough and detailed portrayal of the ShutterSort architecture, catering to the needs of various stakeholders including developers, project managers, and technical analysts.

2. Architectural Representation

The architectural representation of ShutterSort is designed to provide a clear and comprehensive understanding of the software system's structure and behavior. This architecture is delineated through multiple views, each focusing on specific aspects of the system.

The views that are necessary for ShutterSort are as follows:

- Logical View
  - Packages/Modules
  - Classes and Objects
  - Services
  - Interfaces
- Process View
  - Process Flow
  - Interactions
  - Threads
  - Libraries
  - Classes

3. Architectural Goals and Constraints

This section outlines the key architectural goals and constraints that shape the design and implementation of ShutterSort. These elements are critical in guiding the architectural decisions and ensuring that the system aligns with its intended purpose and operational context.

3.1 Goals

- **Scalability**: The architecture should support scalability, both in terms of handling an increasing number of users and managing large volumes of data efficiently.
- **Security/Privacy**: Ensuring the security and privacy of user data is paramount, particularly given the sensitive nature of personal photographs.
- **Usability**: The user interface should be intuitive and accessible, catering to both professional photographers and general users with varying levels of technical expertise.

3.2 Constraints

- **Team Schedule**: Architectural decisions may be influenced by the project's timeline and the structure of the development team.
- **Development Tools**: The choice of development tools and technologies may be influenced by factors like team expertise, cost, and compatibility with existing systems.

4. Use-Case View

N/A

4.1 Use-Case Realizations

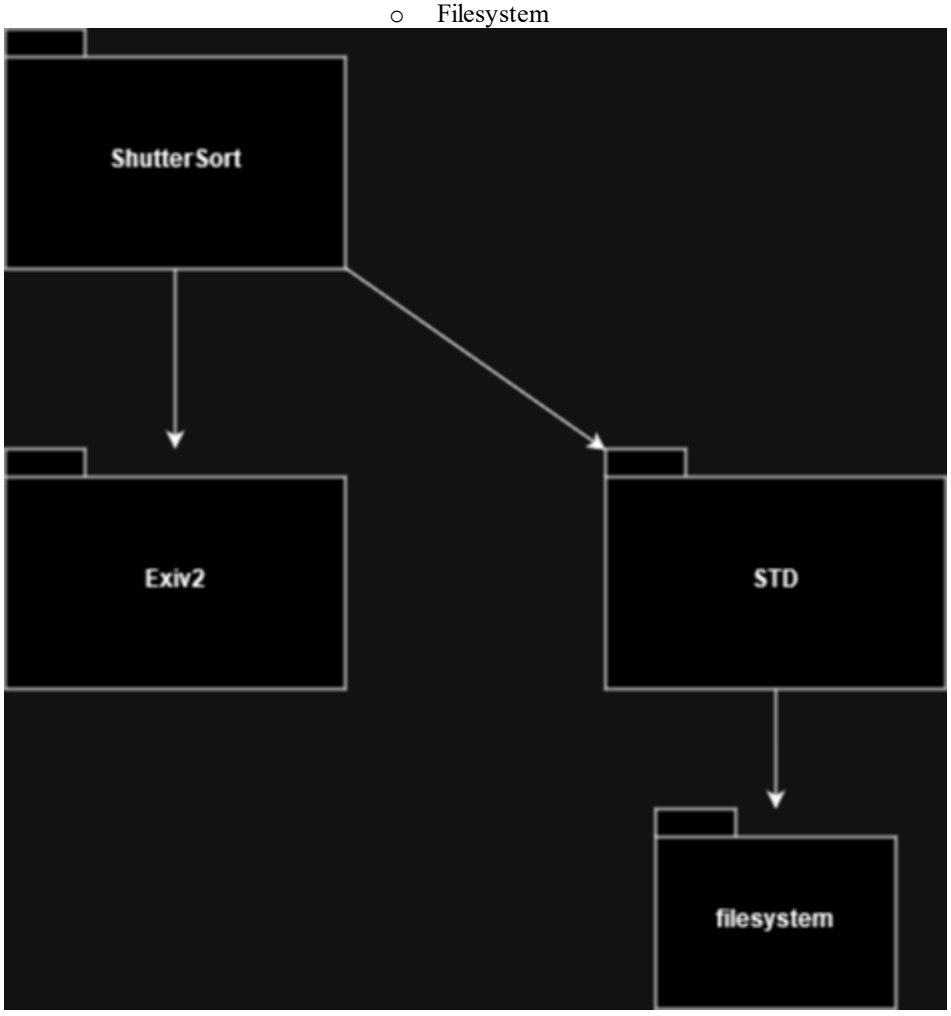
N/A

5. Logical View

5.1 Overview

There are 3 prominent “packages” used for the ShutterSort project. They are:

- ShutterSort (our files/library)
- Exiv2
- The Standard Library
    - Filesystem



5.2 Architecturally Significant Design Modules or Packages

5.2.1 ShutterSort

ShutterSort is the main application package. It contains all of the custom code and classes that we have written, including the main process of the application.

5.2.1.1 Anchor

The Anchor class stores data and methods regarding an individual anchor point.

5.2.1.2 LocalImage

The LocalImage class stores information on a specific image file and it's metadata using a Metadata object.

5.2.1.3 Sorter

The Sorter class contains methods for sorting a set of images based on a stack of SortingMethod objects.

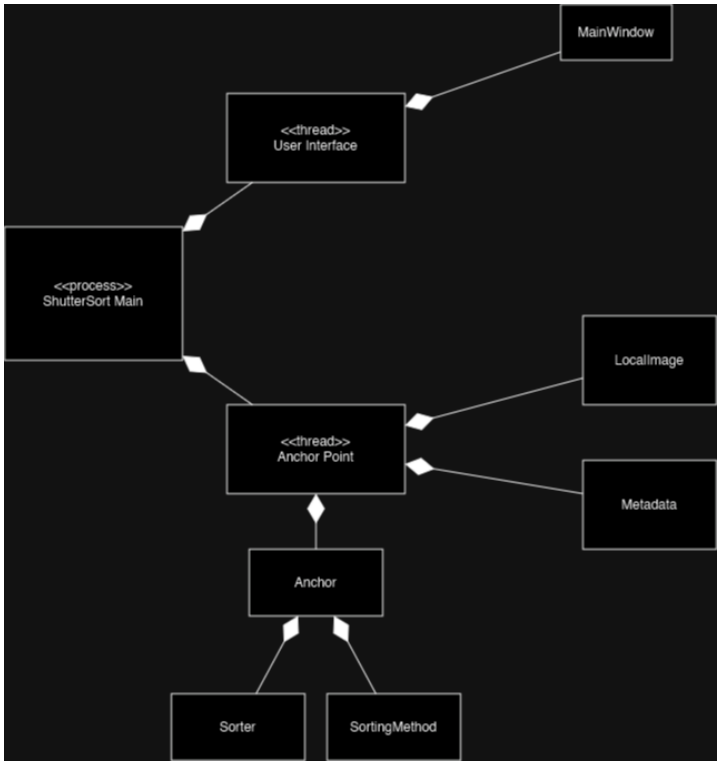
5.2.1.4 SortingMethod

The SortingMethod class contains data regarding a particular method, including the data range, the sorting function, the sorting tag, the name of the method, and the status.

5.2.1.5 Metadata

The Metadata class contains a hash table with metadata tags and values for a given image. It utilizes Exiv2 to read the metadata.

6. Process View



7. Interface Description

The 2 major interfaces for ShutterSort are the GUI and the anchor points. The valid inputs and their outputs are as follows:

Input	Output
Image file(s)	Software sorts/moves the images based on the set sorting parameters
Non-image file(s)	Software moved images to a miscellaneous folder
Folder/directory	Software recursively searches and sorts each sub-folder

The main screen format would be window only, with fullscreen support being added in later versions.

8. Size and Performance

N/A

9. Quality

This section of the Software Architecture Document addresses how the architecture of ShutterSort contributes to the various non-functional quality attributes of the system. These attributes are crucial in determining the system's overall effectiveness, user satisfaction, and long-term viability.

9.1 Extensibility

The architecture of ShutterSort is designed to be extensible, allowing for easy addition of new features and integration of new technologies as they emerge.
This is achieved through modular design and the use of well-defined interfaces, which allow for individual components of the software to be updated or extended without impacting the entire system.

9.2 Reliability

Reliability is a key consideration, ensuring that ShutterSort performs consistently under expected conditions.
The architecture supports reliability through redundant systems, robust error handling, and thorough testing of individual components. The use of proven technologies and frameworks also contributes to the system's stability.

9.3 Maintainability

The system is designed for ease of maintenance to ensure its longevity and adaptability to changing requirements.
This includes clear documentation, adherence to coding standards, and a design that isolates individual components, making them easier to understand, test, and modify.

9.4 Security and Privacy

Given the sensitive nature of personal photographs, security and privacy are paramount.
The architecture incorporates strong encryption, secure data storage, and privacy-by-design principles.
Regular security audits and compliance with data protection regulations are also integral to the architectural strategy.