CS/SE/CE 3354 Software Engineering

Deliverable 1

ITSAMI

Image Tagging Software Applicable for Many Images

Team #: 10

Group Names:

Taufeeq Ali, Connor Ford, Fernando Hernandez, Bowen Liu, Tanner Raley, Asher Schubert, San Yun, Joel Zuniga

1) Updated Version of Proposal

Refined Project Proposal

1. Motivation

Digital collage art involves local collection of thousands of images with varying styles. Artists have a lot to choose from between blueprints, text documents, 3D renders, even corrupt program outputs. A system to automatically tag and search through local archives of images already exists in the form of a command-line project called rclip by Yurij Mikhalevich but is based on the aging OpenAI CLIP model. A new system could be developed that utilizes a GUI for viewing images based on search queries and runs on a newer pre-trained image tagging model.

2. Goals:

Our goal is to develop an intuitive and efficient image tagging software that leverages a modern pre-trained image tagging model, offering a graphical user interface (GUI) for seamless navigation and search through large local archives of diverse images. By improving upon existing solutions like rclip, we aim to provide artists and users with a more user-friendly and powerful tool to automatically tag, organize, and retrieve images based on descriptive queries.

3. Delegated Tasks by Members

a. Frontend (GUI)
-Asher Schubert
-San Yun
-Joel Zuniga
b. Backend (Model Interface)
-Taufeeq Ali
-Connor Ford
c. Backend (GUI Interface)
-Fernando Hernandez
-Bowen Liu
-Tanner Raley

2) Repository Link

https://github.com/connor-ford/itsami

3) Delegation of Tasks

- **Taufeeq Ali**: Class Diagram
- **Connor Ford**: Outlining Software Process Model, Functional Requirements, GitHub repository
- **Fernando Hernandez**: Use-Case Diagram
- **Bowen Liu**: Updated Version of Proposal
- **Tanner Raley**: Non-Functional Requirements, Architectural Design
- **Asher Schubert**: Activity Diagram
- **San Yun**: Class Diagram
- **Joel Zuniga**: Functional Requirements, Non-Functional Requirements

4) Software Process Model

We will be using the Waterfall process model, as a linear, sequential approach with distinct, independent phases would suit the static nature of our project. By separating requirements, design, implementation, and testing into distinct phases of the timeline, we can ensure a better final product than we would by using an incremental or iterative process model.

5) Software Requirements

5.a) Functional Requirements

- The user needs to be able to load images and their tags into the application.
- The application needs to be able to display images and their tags.
- The application needs to be able to process images and create tags.
- The application needs to be able to save an image's tags for later use.
- The application must have a search function to find images by their tags.
- The model must be able to be used through a Command Line Interface (CLI) in addition to within the application.

5.b) Non-Functional Requirements

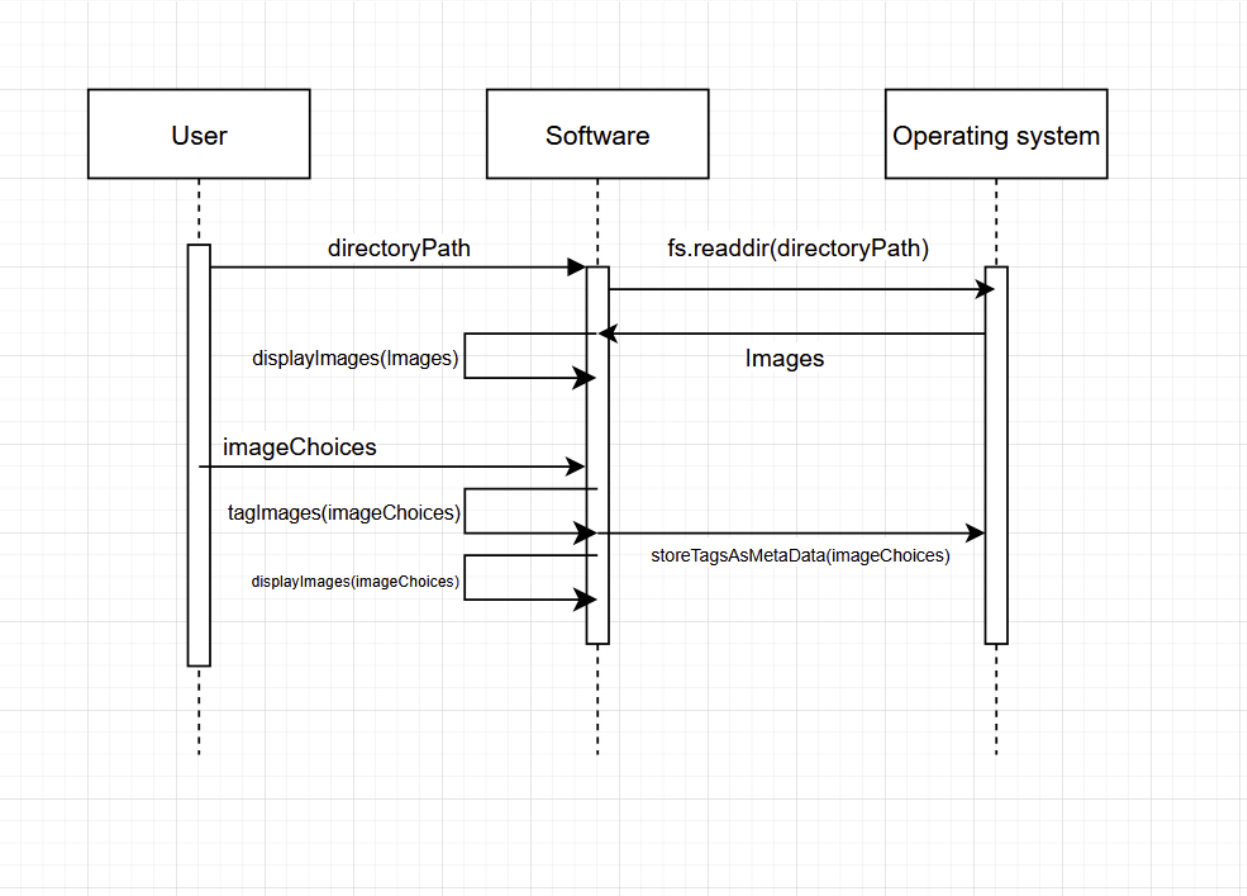
- The application must be able to access and save large amounts of images to the user's system along with their associated tags.

- The application must return an image tagging request within a reasonable amount of time.
- The application needs to be able to support large amounts of tags per image.
- The application must be portable.

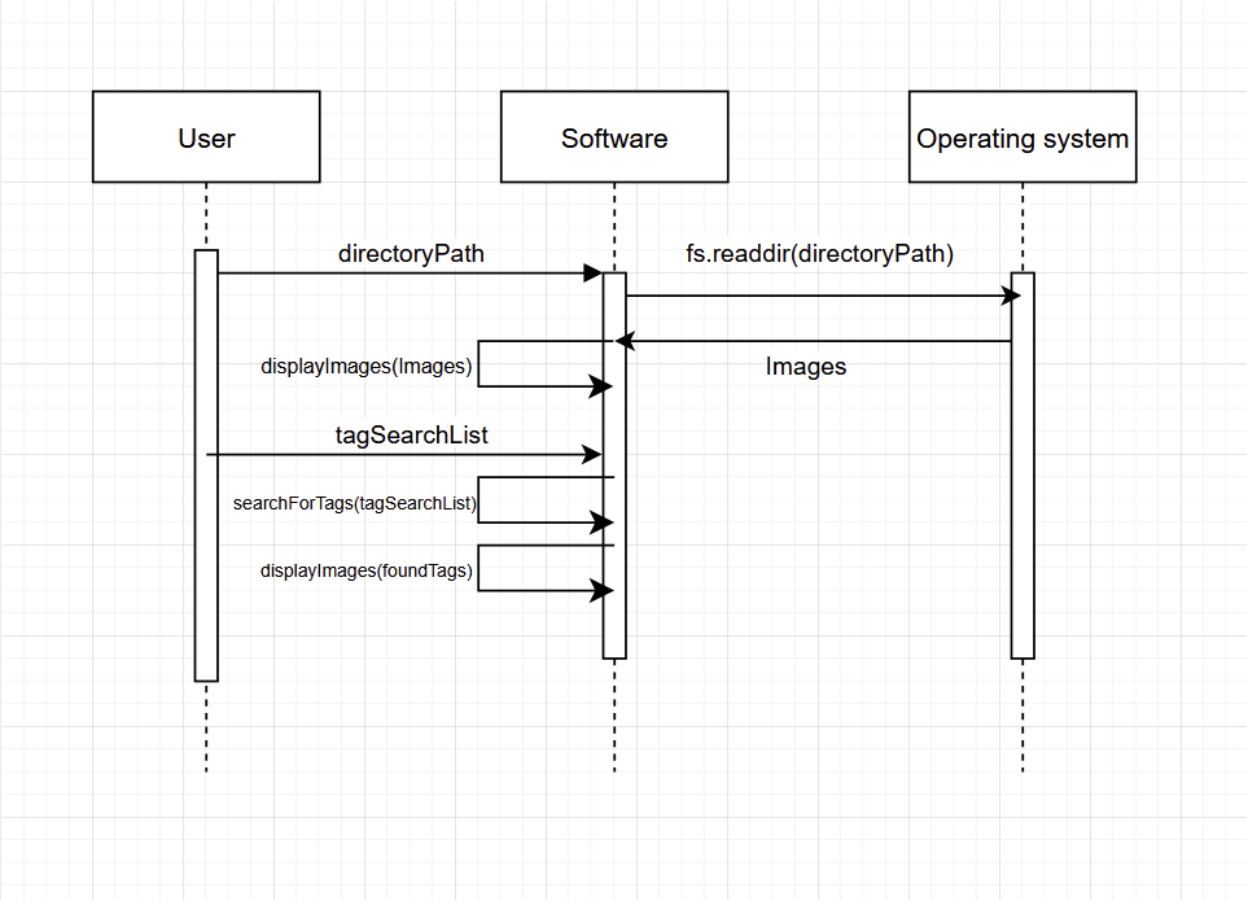
6) Diagrams

Sequence Diagrams

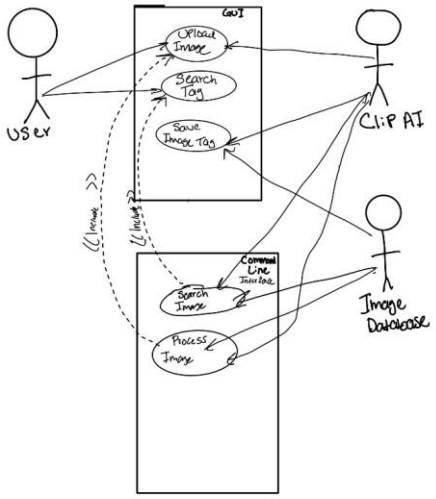
1) Tagging Images



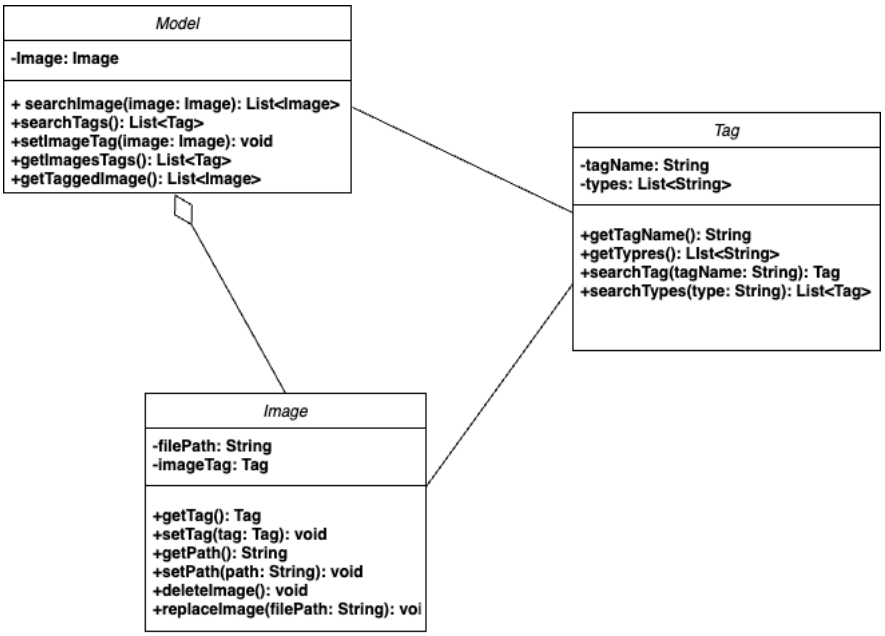
2) Searching Images



Use Case Diagram



Class Diagram



Class	Data	Method
Image	String filePath Tag imageTag	getTag() setTag() getPath()

		setPath() deleteImage() replaceImage(String filePath)
Tag	String tagName List<String> types	getTagName() getTypes() searchTag() searchTypes()
Model	Image Image	searchImage(Image) searchTags() setImageTag(Image) getImageTags() getTaggedImage()

7) Architectural Design

The architectural design pattern that our group has chosen to utilize is the Repository pattern. This pattern was selected because the system requires the application to generate large amounts of data (specifically, image tags) that must be stored for extended periods. In addition, the chosen pattern implements a structured method to handle data access, allowing for ease of adding, modifying, or deleting tags that have been associated with images.