Design Document for Image Processing System

Desktop Application

Version 1.0 approved

Prepared by

Group 2

Vũ Trọng Dũng

Bùi Minh Sơn

Nguyễn Việt Hoàng

Trần Quang Huy

Hồ Vũ Tuấn Minh

Lecturer

Mr. Trần Bình Dương

Process Impact

February 6 , 2025

2. Revision History

Name	Date	Reason For Changes	Version
Vũ Trọng Dũng	26/02/2025	4.5 4.6 4.7 6.5	1.0
Bùi Minh Sơn	27/02/2025	4.1 5.1.1 5.1.3 6.4	1.0
Nguyễn Việt Hoàng	25/02/2025	6.1 6.3	1.0
Trần Quang Huy	25/02/2025	5.1.2	1.0
Hồ Vũ Tuấn Minh	27/02/2025	4.2 4.3 6.2	1.0

3. Table of Contents, Table of Figures , Table of Tables

4. Requirements Modeling

4.1. Business workflows

This Image Processing System has three main workflows: **Image Format Conversion**, **Image Splitting** and **Image Merging**

Image Format Conversion

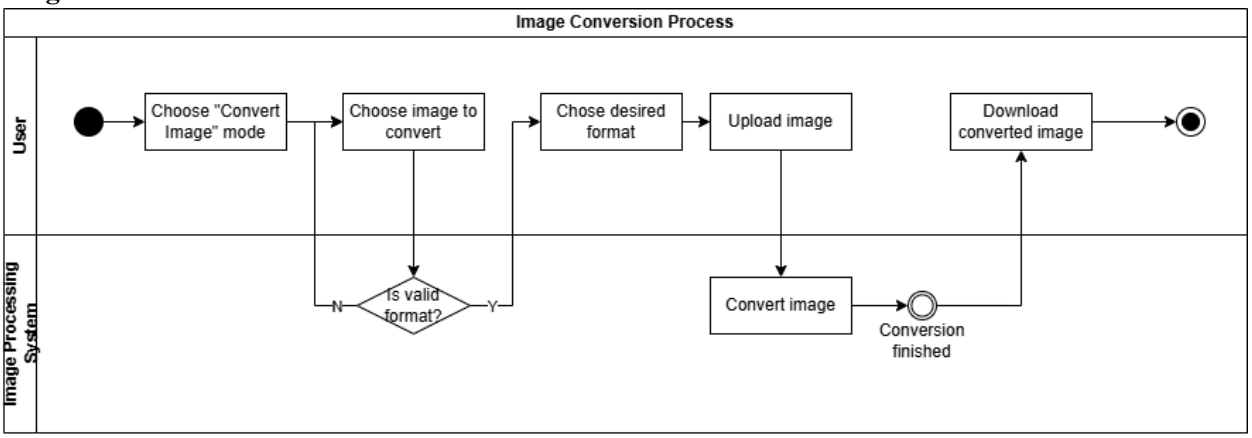

Figure 4.1.1. Business workflow of the Image format conversion process

Step	Step Name	Description
Step 1	Choose “Convert Image” mode	User chooses “Convert Image” mode to convert image format.
Step 2	Choose image to convert	User chooses an image to convert from their computer’s file system. If the image’s format is invalid, return to step 2, else, proceed.
Step 3	Choose desired format	User chooses the format which they want to convert their image to. The desired format must not be the same as the chosen image’s format.
Step 4	Upload image	User uploads the image to the server.
Step 5	Convert image	The system converts the image format to the desired format.
Step 6	Download converted image	The user downloads the converted image to their computer.

Table 4.1.1: Description of the Image format conversion workflow

Image Splitting

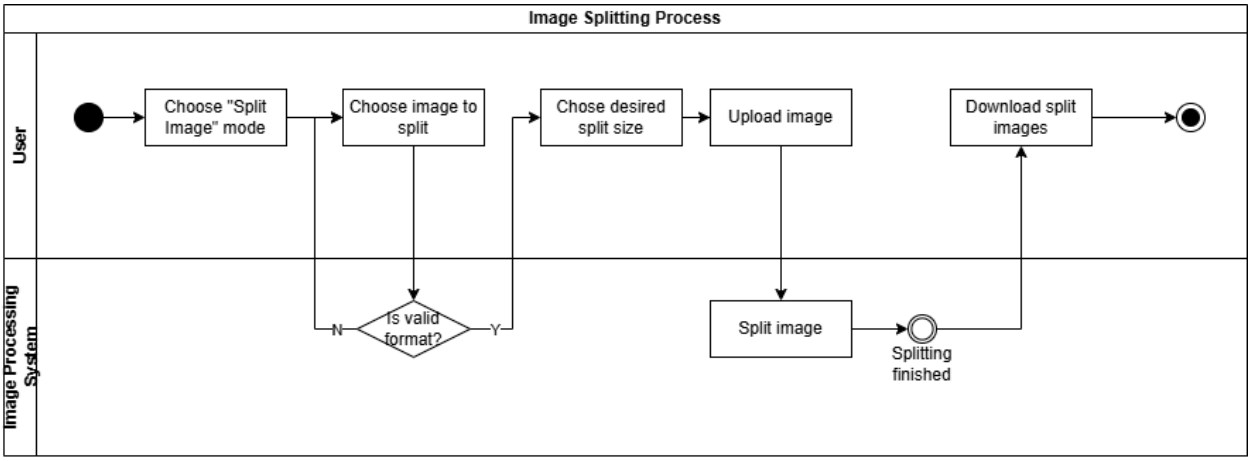

Figure 4.1.2. Business workflow of the Image splitting process

Step	Step Name	Description
Step 1	Choose “Split Image” mode	User chooses “Split Image” mode to split an image.
Step 2	Choose image to split	User chooses an image to convert from their computer’s file system. If the image’s format is invalid, return to step 2, else, proceed.
Step 3	Choose desired split size	User chooses the split size they want. The split size is the size of the square tiles which are split from the original image.
Step 4	Upload image	User uploads the image to the server.
Step 5	Split image	The system split the image into square tiles with the desired split size.
Step 6	Download split images	The user downloads the divided image (image tiles) to their computer.

Table 4.1.2: Description of the Image splitting workflow

Image Merging

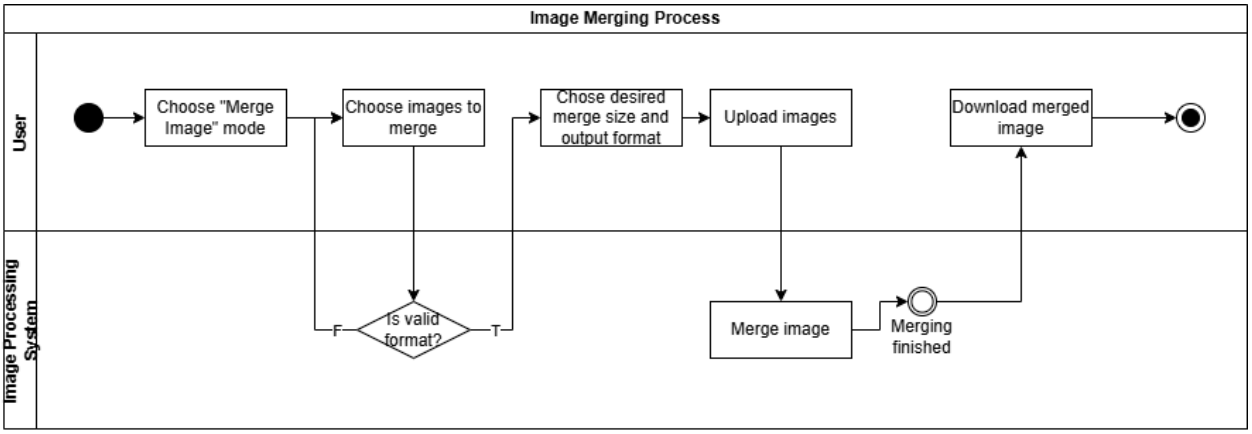

Figure 4.1.3. Business workflow of the Image merging process

Step	Step Name	Description
Step 1	Choose “Merge Image” mode	User chooses “Merge Image” mode to merge images into a larger one.
Step 2	Choose images to merge	User chooses images to merge from their computer’s file system. If the format of any image is invalid, return to step 2, else, proceed.
Step 3	Choose desired merge size and output format	User chooses the desired size (width and height) and the format of the merged image.
Step 4	Upload image	User uploads the images to the server.
Step 5	Merge images	The system merges the images into a large image with the desired size.
Step 6	Download merged image	The user downloads the merged image to their computer.

Table 4.1.3: Description of the Image merging workflow

4.2. Use-case diagrams

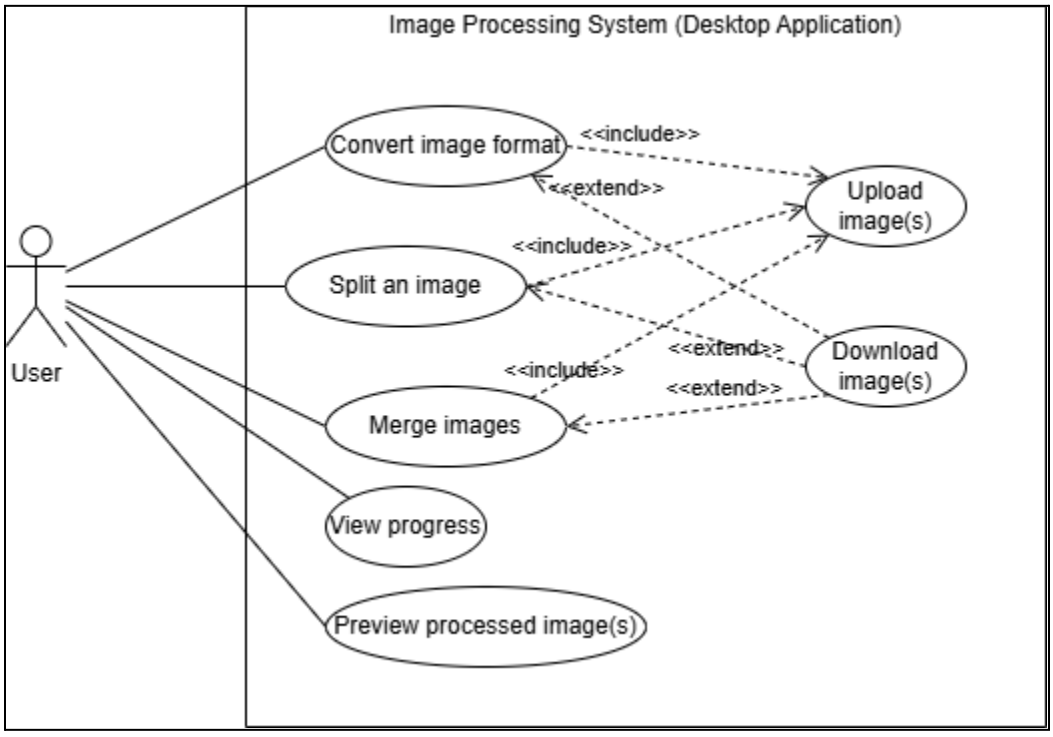

Figure 4.2.1: Use case diagram

4.3. Use-case description for each use-cases

4.3.1. Use Case List

Primary Actor	Secondary actor	Use Case name	Description
User	None	UC-1: Convert image to another format	
User	None	UC-2: Split an image into smaller square tiles	
User	None	UC-3: Merge small images into a larger image	
User	None	UC-4: View progress	
User	None	UC-5: Upload image(s)	
User	None	UC-6: Download image(s)	
User	None	UC-7: Preview processed image(s) before downloading	

Table 4.3.1: Use Case List

4.3.2. Use Case Description

UC-1: Convert image to another format

UC ID and Name:	UC-1: Convert image to another format		
Created By:	Bui Minh Son	**Date Created:**	2/6/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	The user wants to convert their images to another format.		
Description:	As a user, I want to convert my images to another format.		
Preconditions:	None		
Post-conditions:	None		
Normal Flow:	1: The user visits the screen “Convert Image”. 2: The user chooses the desired format, then chooses the image for converting. 3: The user then clicks “Convert”, and waits until the conversion is done. 4: The user can choose more images, and the conversion for those images are processed concurrently. 5: When conversion is done, the user can preview the converted image. 6: The user clicks “Save” to save the image to their machine, or “Discard” to discard.		
Alternative Flows:	None		
Exceptions:	1: Invalid image format. 1.1: After choosing the image, a message will be displayed on the screen, notifying that the image format is not accepted. 1.2: The user then chooses another image.		
Priority:	High		
Frequency of Use:	Frequent		
Business Rules:	BR-1		
Other Information:	None		
Assumptions:	None		

UC-2: Split an image into smaller square tiles

UC ID and Name:	UC-2: Split an image into smaller square tiles		
Created By:	Bui Minh Son	**Date Created:**	2/6/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	The user wants to split images into smaller square tiles.		
Description:	As a user, I want to split my images into smaller square tiles.		
Preconditions:	None		
Post-conditions:	None		
Normal Flow:	1: The user visits the screen “Split Image”. 2: The user chooses the desired tile size and output format, then chooses the image for splitting. 3: The user then clicks “Split”, and waits until the splitting is done. 4: The user can choose more images, and the splitting for those images are processed concurrently. 5: When splitting is done, the user can preview the tiles. 6: The user clicks “Save” to save the images to their machine, or “Discard” to discard.		
Alternative Flows:	None		
Exceptions:	1: Invalid image format. 1.1: After choosing the image, a message will be displayed on the screen, notifying that the image format is not accepted. 1.2: The user then chooses another image.		
Priority:	High		
Frequency of Use:	Frequent		
Business Rules:	BR-1		
Other Information:	None		
Assumptions:	None		

UC-3: Merge small images into a larger image

UC ID and Name:	UC-3: Merge small images into a larger image		
Created By:	Bui Minh Son	**Date Created:**	2/6/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	The user wants to merge small images into a larger image.		
Description:	As a user, I want to merge my images into a larger image.		
Preconditions:	None		
Post-conditions:	None		
Normal Flow:	1: The user visits the screen “Merge Image”. 2: The user chooses the desired image size and output format, then chooses the images for merging. 3: The user then clicks “Merge”, and waits until the merging is done. 4: When the merging is done, the user can preview the merged image. 5: The user clicks “Save” to save the image to their machine, or “Discard” to discard.		
Alternative Flows:	None		
Exceptions:	1: Invalid image format. 1.1: After choosing the image, a message will be displayed on the screen, notifying that the image format is not accepted. 1.2: The user then chooses another image.		
Priority:	High		
Frequency of Use:	Not frequent		
Business Rules:	BR-1		
Other Information:	None		
Assumptions:	None		

UC-4: View progress

UC ID and Name:	UC-4: View progress		
Created By:	Bui Minh Son	**Date Created:**	2/6/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	This function triggers when the server is processing image(s) for the user.		
Description:	As a user, I want to view the progress of image processing.		
Preconditions:	PRE-1: The system is processing the user's uploaded image.		
Post-conditions:	POST-1: Upon finishing processing, the progress bar disappears.		
Normal Flow:	1: The system updates the progress after finishing each step in the process.		
Alternative Flows:	None		
Exceptions:	None		
Priority:	Normal		
Frequency of Use:	Frequent		
Business Rules:			
Other Information:	None		
Assumptions:	None		

UC-5: Upload image(s)

UC ID and Name:	UC-5: Upload image(s)		
Created By:	Bui Minh Son	**Date Created:**	2/8/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	This function triggers when the user clicks the “Upload image” button.		
Description:	The user uploads image(s) to the server for processing.		
Preconditions:	PRE-1: The user must log into the system.		
Post-conditions:	POST-1: All user-uploaded images are on the server’s temporary folder, ready for processing.		
Normal Flow:	1: User chooses the processing mode (convert, merge, split). 2: User chooses the desired output format. 3: User clicks on the “Upload image” button, and chooses the image to be processed. 4: After that, click “Upload”. 5: Image is uploaded and is stored in a temporary folder on the server, and a task created and is queued for processing. 6: A progress bar for the uploaded image appeared on the screen. Meanwhile, the user can still choose more images to upload.		
Alternative Flows:	None		
Exceptions:	1: Invalid image format. 1.1: After step 4, a message will be displayed on the screen, notifying that the image format is not accepted. 1.2: The user then chooses another image. 2: Error during uploading image. 1.1: After step 4, a message will be displayed on the screen, notifying the user to try again later.		
Priority:	Normal		
Frequency of Use:	Frequent		
Business Rules:	BR-1		
Other Information:	None		
Assumptions:	None		

UC-6: Download image(s)

UC ID and Name:	UC-6: Download image(s)		
Created By:	Bui Minh Son	**Date Created:**	2/8/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	This function triggers every time when a user's uploaded image is processed and the user clicks on the "Download" button.		
Description:	The user downloads the processed image(s) to their local machine.		
Preconditions:	PRE-1: The user must log into the system. PRE-2: The system has finished processing the user's uploaded image.		
Post-conditions:	POST-1: The processed image is downloaded to the user's local machine.		
Normal Flow:	1: The image is finished processing, the progress bar disappears, and the "Download" button appears. 2: The user clicks on the "Download" button. 3: The image is downloaded to the user's local machine.		
Alternative Flows:	None		
Exceptions:	1: Error during downloading image 1.1: After step 2 of the normal flow, or the step 1.3 of the alternative flow 1, a message appears on the screen to notify the user to try again later		
Priority:	Normal		
Frequency of Use:	Frequent		
Business Rules:	None		
Other Information:	None		
Assumptions:	None		

UC-7: Preview processed image(s)

UC ID and Name:	UC-7: Preview processed image(s) before downloading		
Created By:	Bui Minh Son	**Date Created:**	2/6/2025
Primary Actor:	User	**Secondary Actors:**	None
Trigger:	This function is triggered when image processing is completed		
Description:	As a user, I want to preview the processed image, so I can decide whether or not to download it.		
Preconditions:	PRE-1: The system has finished processing the user's uploaded image.		
Post-conditions:	None		
Normal Flow:	1: After processing finished, a button “Preview” appears. 2: The user clicks on “Preview”. 3: The processed image will appear on the screen for the user to preview.		
Alternative Flows:	None		
Exceptions:	None		
Priority:	Normal		
Frequency of Use:	Not frequent		
Business Rules:	None		
Other Information:	None		
Assumptions:	None		

4.4. Activity Diagram

None

4.5. Non-functional Requirements

4.5.1. Usability

- **User-Friendly Interface**: The desktop application should have a simple, intuitive interface with clear labels, tooltips, and a well-organized layout.
- **Offline Usability**: The system must be fully functional without requiring internet connectivity, as it is intended for offline usage.

- **Support for Drag and Drop**: Allow users to drag and drop images directly into the application for easier uploads.

4.5.2. Maintainability & Scalability

- **Modular Codebase**: The application should follow a modular architecture to facilitate easy bug fixes, updates, and future feature enhancements.

- **Scalability for Features**: Allow the addition of advanced image processing functionalities, such as image filters, watermarking, or batch processing, in future versions.

4.5.3. Data Storage & File Handling

- **Local Storage Management**: Allow users to choose where processed images are stored on their machine.

- **Support for Multiple Image Formats**: The desktop version must support 4 image formats, including JPEG, PNG, BMP, WEBM, and allow seamless format conversions.

4.6. Business Rules Table

ID	Rule Definition	Type of Rule	Static or Dynamic
BR-1	Accepted image formats include .jpg, .bmp, .png and .webm	Validation Rule	Static

Table 4.6.1: Business Rules Table

4.7. Concurrent Tasks Description

There are 4 tasks that require concurrent processing, which are:
- Converting image format
- Splitting an image
- Merging images
- Updating progress to the user

5. Analysis Modeling

5.1. Static Modeling Diagrams and Explanation

5.1.1. Contextual External Class Diagram.

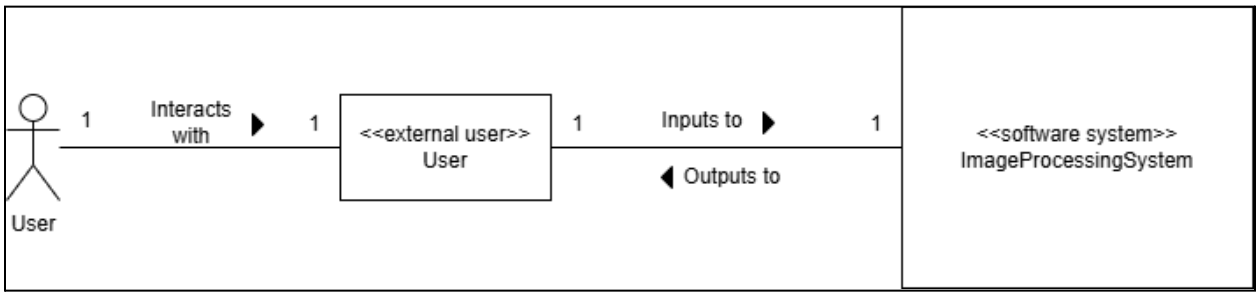

Figure 5.1.1.1. Contextual external class diagram

The **Image Processing System** interacts with two external classes that facilitate user access:

- **User:** Human users interacting with the system through their devices.
- **User (external user):** The <<external user>> class represents the standard I/O devices such as screen, keyboard or mouse, through which the human Users interact with the system. They enable data input, image uploads, and system responses display, and is the primary access point for human users.

5.1.2. Business Processing Classes Hierarchical Structure

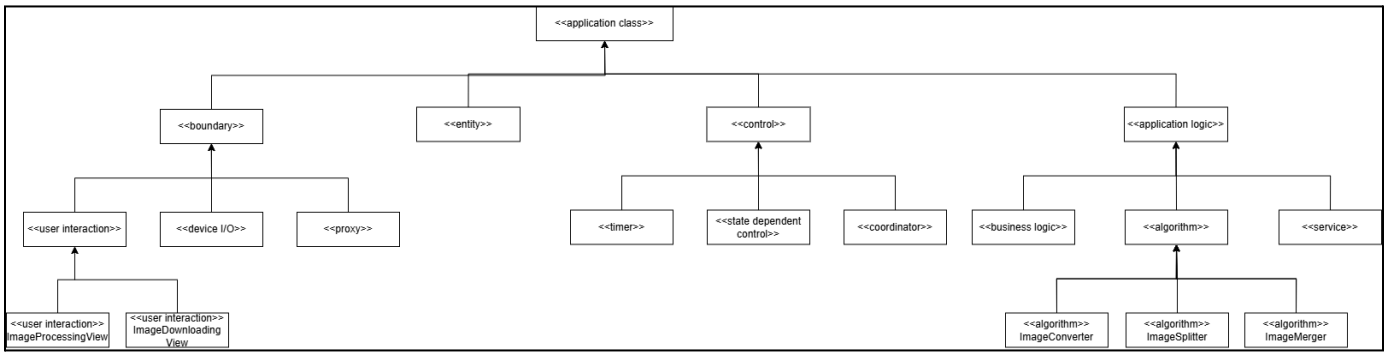

Figure 5.1.2.1. Business Processing Classes Hierarchical Structure

The diagram classifies application classes by stereotypes, organizing them into hierarchical layers. At the top level, the system has the **<<application class>>** stereotype, which represents all application-related classes. Below it, classes are categorized into more specific stereotypes based on their roles in the system:

<<user interaction>>

This stereotype represents UI-related classes that users interact with. This stereotype is in the **<<boundary>>** super-stereotype. Classes under this stereotype are:

- ***Image Processing View***: The view where user uploads images for processing and views progress
- ***ImageDownloadingView:*** The view where user can preview and download processed images

<<application logic>>

This stereotype represents the core functional components responsible for executing the primary business processes of the Image Processing System. This stereotype contains:

- **<<algorithm>>:** Represents classes or components that implement computational logic, data transformation, or mathematical operations for processing data. Classes under this stereotype are:
    - ***Image Converter***: Handles format conversion of uploaded images.
    - ***Image Splitter***: Splits images into smaller parts.
    - ***Image Merger***: Merges multiple images into a single output file.

5.1.3. Classes which perform Concurrent Tasks responsibility

There are 5 classes that perform concurrent tasks:
- Image Converter
- Image Splitter
- ImageMerger
- ImageProcessingView
- ImageDownloadingView

5.2. Dynamic Modeling Diagrams and Explanation

5.2.1. Communication Diagrams

None

5.2.2. Integrated Communication Diagrams

None

5.2.3. Concurrent Communication Diagrams

Convert image format

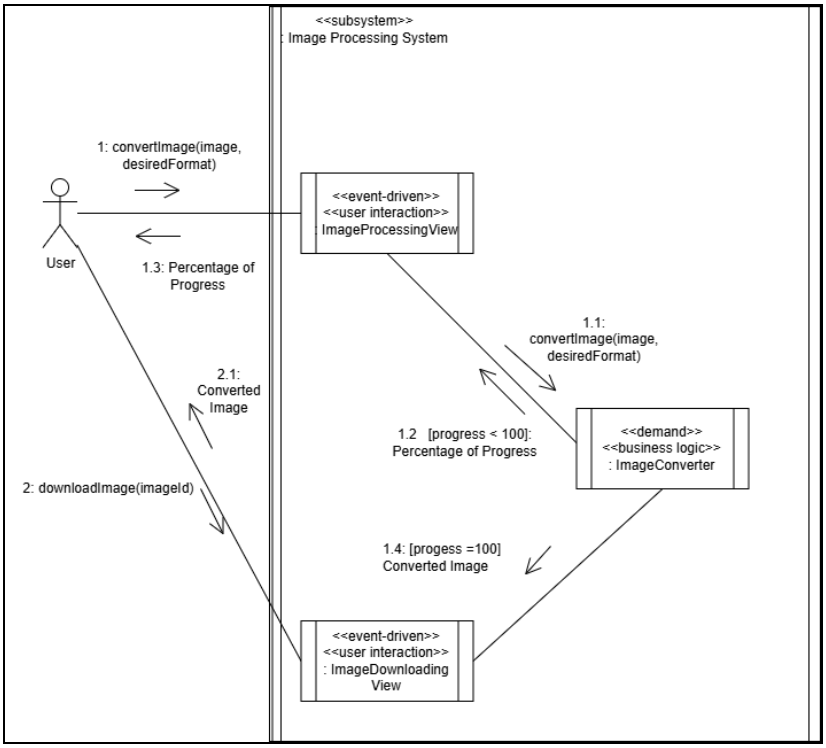

Figure 5.2.3.1: Communication diagram for use-case "Convert image format"

Description:
1: The user uploads an image and initiates an image conversion request with the desired format.
1.1: The *ImageProcessingView* forwards the request to the *ImageConverter*.
1.2 and 1.3: While converting, the *ImageConverter* updates the progress to the user via the *ImageProcessingView*.
1.4: When the conversion is finished, the *ImageConverter* sends the converted image to the *ImageDownloadingView* where the user can preview or download it.
2: The user requests to download the image.
2.1: The system returns the converted image to the user to save on their local machine.

Split an image

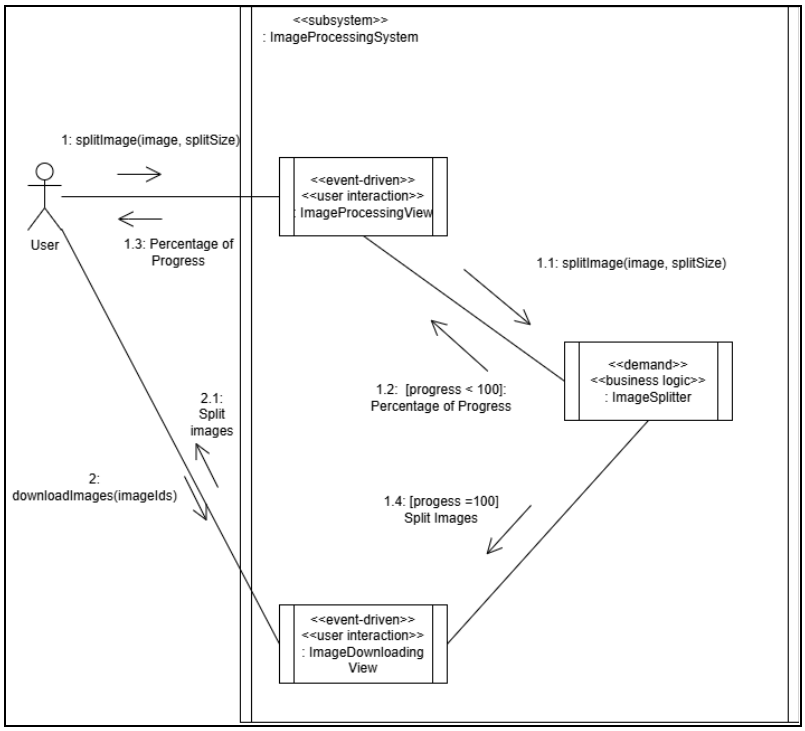

Figure 5.2.3.2: Communication diagram for use-case "Split an image"

Description:
1: The user uploads an image and initiates an image splitting request with the desired split size.
1.1: The *ImageProcessingView* forwards the request to the *ImageSplitter*.
1.2 and 1.3: While converting, the *ImageSplitter* updates the progress to the user via the *ImageProcessingView*.
1.4: When the splitting is finished, the *ImageSplitter* sends the split image to the *ImageDownloadingView* where the user can preview or download them.
2: The user requests to download the images.
2.1: The system returns the split images to the user to save on their local machine.

Merge images

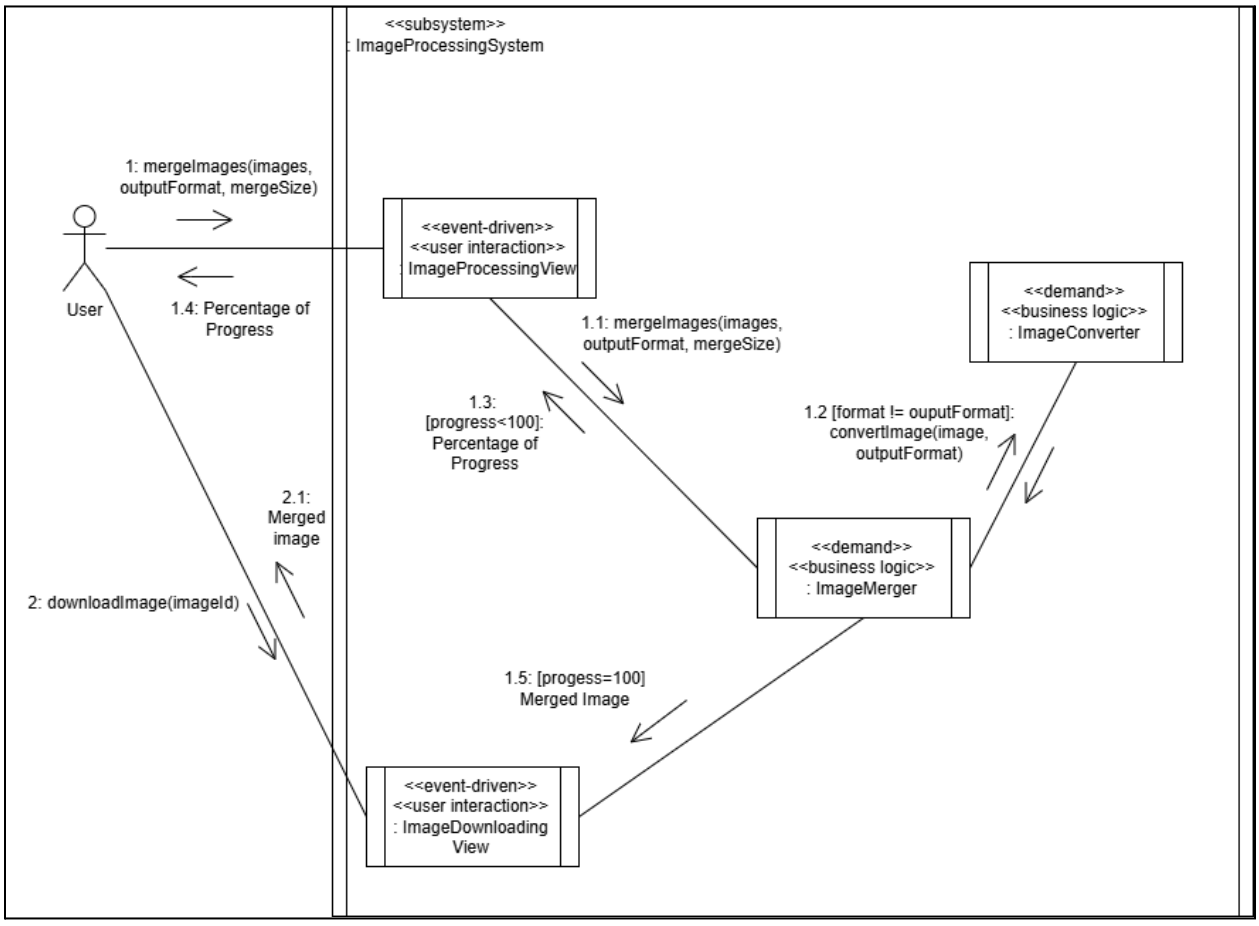

Figure 5.2.3.3: Communication diagram for use-case "Merge images"

Description:
1: The user uploads multiple images and initiates an image merging request with the desired format and merge size.
1.1: The *ImageProcessingView* forwards the request to the *ImageMerger*.
1.2: If the images' format differs from the output format, they are first converted using the *ImageConverter*, then the image is returned to the *ImageMerger*.
1.3 and 1.4: While converting, the *ImageMerger* updates the progress to the user via the *ImageProcessingView*.
1.5: When the merging is finished, the *ImageMerger* sends the merged image to the *ImageDownloadingView* where the user can preview or download it.
2: The user requests to download the image.
2.1: The system returns the merged image to the user to save on their local machine.

6. Design Modeling

6.1. List of Architecture Patterns which be selected

Layered Architecture Pattern

- The system is organized into 2 layers, each with a distinct responsibility, ensuring modularity, separation of concerns, and maintainability between layers.
- The layers of the system will be described in the figure *6.2.1* below.

6.2. Architecture in three views

6.2.1. Architecture in Static View

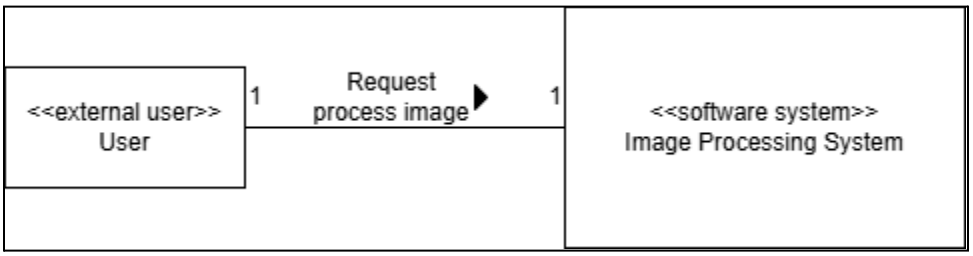

Figure 6.2.1.1: Architecture in Static View

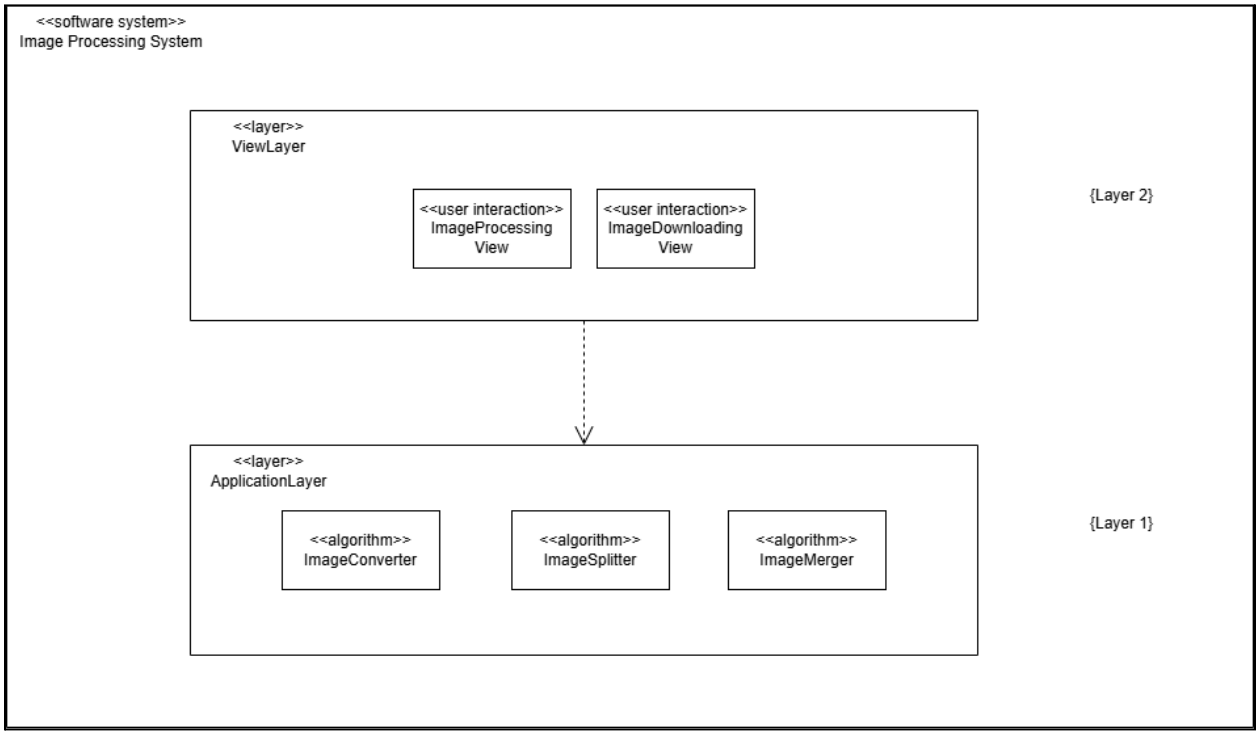

Figure 6.2.1.2: Internal layered architecture of the software system

The Image Processing System is structured into two layers.

Layer 1: Application Layer

- Implements core functionalities of the system.
- Includes algorithm components:
    - **Image Converter**: Converts images into different formats.
    - **Image Splitter**: Splits images into multiple parts.
    - **Image Merger**: Merges multiple images into one.

Layer 2: View Layer

- Provides user interfaces for system interaction.
- Contains various views:
    - **Image Processing View**: Allows users to process images.
    - **Image Downloading View**: Enables users to download processed images.

6.2.2. Architecture in Dynamic View

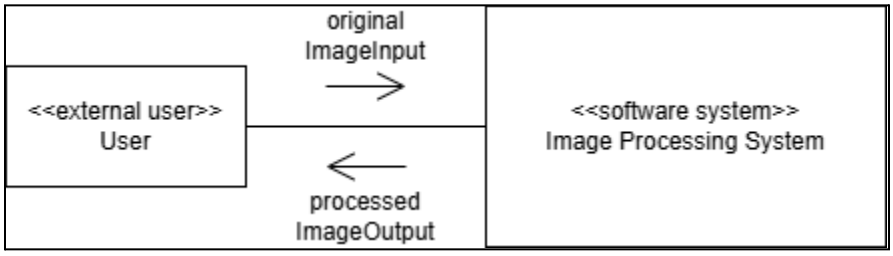

Figure 6.2.2.1: Architecture in dynamic view

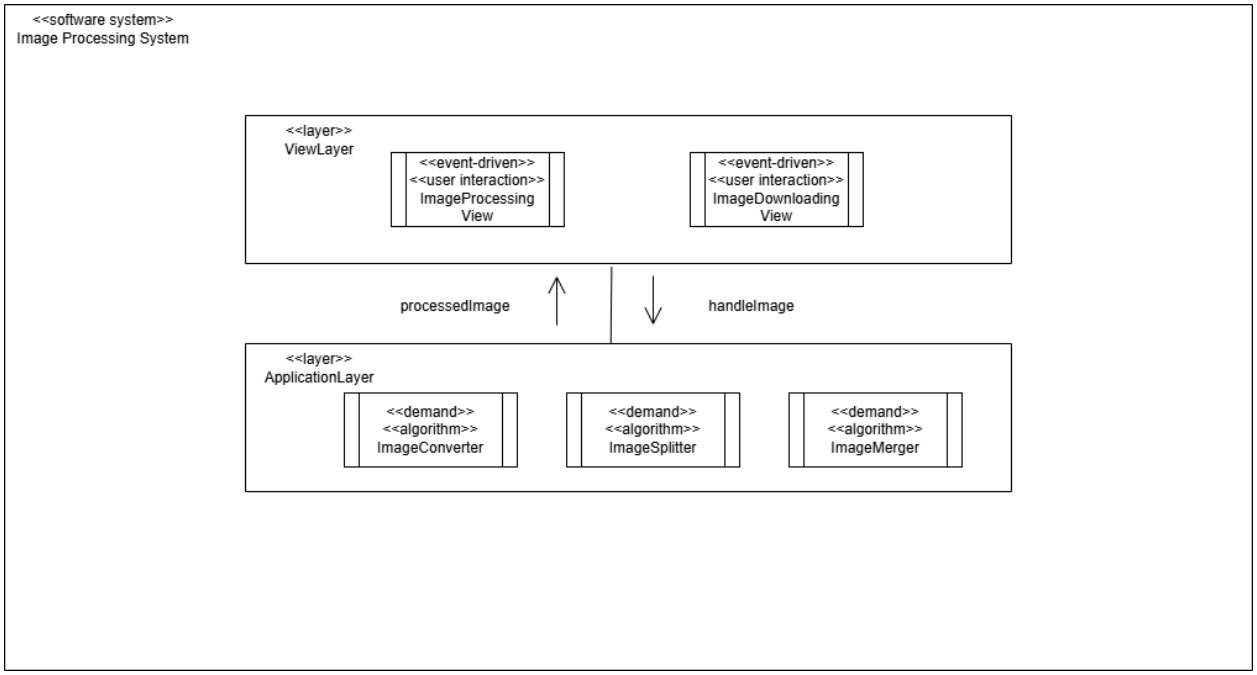

Figure 6.2.2.2: Dynamic view of the internal layered architecture

This layered architecture diagram illustrates the message flow between components in the Image Processing System. Below is an explanation of how data flows:

- User uploads or selects images via the ImageProcessing View.
- The system sends the image to the appropriate algorithm within the Application Layer.
- One or more algorithms process the image based on the user's demand.
- The processed image is sent back to the View Layer.
- Users can then download the final output through the ImageDownloading View.

6.3. Detailed Specifications for each component and class

None

6.4. Concurrent Tasks/Classes Specifications

6.4.1. ImageConverter

Stereotype: **<<algorithm>>**
Purpose: Handle converting image format. Implements interface **IImageConverter**.

Methods

Method name	Access modifier	Parameters	Return type	Description
convertImage	public	(image: Image, desiredFormat: String)	Image	Convert user-uploaded image to the format desired by the user

6.4.2. ImageSplitter

Stereotype: **<<algorithm>>**
Purpose: Handle splitting an image into smaller tiles. Implements interface **IImageSplitter**.

Methods

Method name	Access modifier	Parameters	Return type	Description
splitImage	public	(image: Image, splitSize: int)	Image	Split an user-uploaded image into image tiles with size desired by the user

6.4.3. ImageMerger

Stereotype: **<<algorithm>>**
Purpose: Handle merging multiple images into a larger image. Implements interface **IImageMerger**.

Attributes

Attribute name	Access modifier	Type	Description
imageConverter	private	IImageConverter	Handle converting images to desired output format

Constructor

Parameter	Type	Description

imageConverter	any implementation of interface **IImageConverter**	Inject an implementation of interface **IImageConverter** into the class

Methods

Method name	Access modifier	Parameters	Return type	Description
mergeImages	public	(images: List<Image>, outputFormat: String mergeSize: double)	Image	Merge user-uploaded images into a larger image with size and format desired by the user.

6.4.4. ImageProcessingView

- Stereotype: **<<user interaction>>**
- Purpose: The view where user can upload images for processing (converting, splitting, merging) and get updates about the progress

Interactions

Interaction	Parameters	Description
convertImage	(image: Image, desiredFormat: String)	Send user requests to convert image format
splitImage	(image: Image, splitSize: int)	Send user requests to split an image
mergeImages	(images: List<Image>, outputFormat: String, mergeSize: double)	Send user requests to merge images
viewPercentage OfProgress	*N/A*	View the progress made to process an image

6.4.5. ImageDownloadingView

- Stereotype: **<<user interaction>>**
- Purpose: The view where user can preview and download processed images

Interactions

Interaction	Parameters	Description
previewImage	(imageId: int)	Preview an image before downloading

downloadImage	(imageId: int)	Save an image to the user's local device

6.5. List of quality attributes and evaluation results

This section evaluates how the desktop version of the Image Processing Application meets key non-functional requirements (NFRs). Since the desktop version differs from the web version by being designed for single-user offline use, the evaluation focuses on relevant offline scenarios.

6.5.1. Usability

Availability ensures the system is stable, reliable, and consistently available to the user.

Evaluation:

- **Offline Operation**: Unlike web applications, the desktop version is fully functional offline, enhancing availability in environments without internet access.
- **UI**: The user interface is designed to be minimalistic, intuitive and easy to use for users, with clear labels, tooltips and a well-organized layout.

6.5.2. Maintainability

Maintainability ensures that the system can be efficiently maintained, including bug fixing, applying updates, and extending functionality.

Evaluation:
The desktop application is developed with a modular monolithic architecture, where image processing functions (e.g., conversion, splitting, merging) are separated into distinct modules.

- Each module (e.g., "Image Conversion" or "Image Splitting") has clearly defined functions and minimal dependencies, improving maintainability.
- Code documentation, version control (e.g., Git), and logging mechanisms enhance error tracing and debugging.

6.5.3. Modifiability/Scalability

Modifiability ensures that the application can easily adapt to future changes, such as adding new image processing features.

Evaluation:
The application follows a layered structure, separating the user interface (UI), core business logic, and data handling. This structure makes it easier to update or modify any layer independently:

- UI Modifiability: The UI can be modified to improve the user experience (e.g., adding drag-and-drop image uploads).

- Logic Modifiability: Developers can add new image processing functions (e.g., image watermarking) by creating additional modules.

6.5.4 Data Storage & File Handling

- **Local Storage Management**: Along with the operating system, users can choose where processed images are stored on their machine.

- **Support for Multiple Image Formats**: The version supports users to change among these 4 image formats, including JPEG, PNG, BMP, WEBM, and allow seamless format conversions.