COMP6003 - Assignment 1: User Stories

By Angel Amenze-Osarenren (aa2551), Oghosa Okundaye (Ogo7), Shania Newman (sn471) and Solomon (SR892)

Contents

User stories

1.1: Redo Functionality - ANGEL

As a user I want to be able to redo my actions so that I can go back and forth through my changes and redo actions I have mistakenly undone.

In conversation, the user expressed at times they may slip and mistakenly press the undo button one, or a few too many times. At the moment the remote application does not allow the user to change between current and previous image states. The user expressed that they do not feel they have much autonomy and their decisions feel more critical because of this.

To hand autonomy back to the user, a redo button can be implemented. This would store a user's undone changes to the canvas in either an ArrayList or Stack, using Last-In-First-Out (LIFO) logic, so that the most recently undone canvas is reloaded onto the canvas when the user clicks the redo button. The redo button would also allow users to iterate through all undone canvas until, either the last undone canvas has been redone or the user makes a new generation/change after the undo button has been clicked.

Acceptance criteria:

- There is a redo button available for the user.
- The canvas displays the previously undone canvas.
- The user can continue to press redo until the canvas reverts to their desired initial state.
- The redo button only becomes available to click when a redo action can be performed.
-

Priority: High - This is a Must Have as users may feel like they have no choice if they can not redo.
Estimate: 3 pts - The task is relatively easy to implement. The most challenging element will be choosing the method of implementation that will perform well with the redo button logic.

1.2: Save Functionality - ANGEL

As a user I want to save the images generated in Imogene so that I can download and keep them for future use and ensure I do not lose my current generated image.

One point raised by the user is that they would like to store with an identifiable name and access their image later. The user also wants to be able to save a canvas without having to decide on a filename, for instances where they only need to save the canvas for quick, or immediate use. The user also expressed a need to be able to return to their current Imogene project anytime they want.

To allow users to continue from their current canvas, a save button could be implemented. Upon clicking this would generate a default file name for the user. But, allow the user to create a custom file name using file format tags and a timestamp of when it was saved, so each file is unique. Users should be able to choose the file format they would like their image stored in i.e., PNG, JPEG and BMP and this would be included in the file name by default. A cancel button should also be available, if the user wishes to no longer save their canvas, or make more changes before saving their canvas.

Acceptance criteria:

- The user can press the “Save Image” button.
- The user can choose which format i.e. jpeg, png ect, they would like their generated image to be saved in.
- Users can either choose custom filenames or have them auto generated.
- If the image is successfully saved, the user is informed with a form of visual feedback.

Priority: High - This is a must have, as users may feel there is no point in using the app if they can not save and revisit their canvas, or simply view their canvas later.

Estimate: 6 pts - The task is slightly more challenging to implement compared to the save button. The most challenging part will be implementing file choice and optional filenaming together.

1.3:
~~Image Scaling (Customisable image size)~~ - OGHOSA

As a user,
I want to be able to change the width and height of the generated image
So that I can create images for my desired dimensions that can fit different displays or for exportation.

An issue that is raised by the user is that the application displays images using default, fixed dimensions. Users have no ability to change how large or small the generated image is, which limits the flexibility when the display is on restricted display sizes or by choice from the user. Allowing the user to input their desired image dimensions for e.g., 200 x 300 ect, would make the system more accessible and adaptable to different preferences, display sizes or use cases.

The feature could be implemented through an option button on the panel of the application or a dialogue box that includes text fields or drop down options for width and height. When the user applies the changes, the generated image should automatically update with the specified dimensions.

Acceptance Criteria:

- The user can input a custom number for the width and height of the image through the application interface.
- The system checks the input value is valid - The values are positive and within limits of the overall application window.
- The image display updates dynamically when the new dimensions are set.
- Default image size values are restored on reset.

Priority: High
- Provides noticeable improvement to the usability and accessibility of the imogene application.

Estimate: 3 Story Points
- Moderate complexity to implement function, low risk and effort.

1.4:

~~Image Scaling (Customise upscaling factor~~) - OGHOSA

As a user
I want to be able to change how much I increase the image display
So that I can control how large the image appears on the screen without affecting its resolution.

Another issue raised by the user is that the application currently uses a fixed upscaling factor, therefore, the image displayed is enlarged by a preset value defined in the code. This prevents users from adjusting how zoomed-in or zoomed-out the image appears. This would also limit the range of devices in which the application could be deployed on.

The inclusion of features that would allow users to customise the upscaling factor directly within the application interface was suggested. As by adjusting these factors, users can control how large the image appears on the screen whilst maintaining the images resolution.

This functionality could be implemented through a slider, numeric input field, or dropdown menus on the panel of the application. When the user modifies the scaling value, the displayed image should dynamically update to show the user defined zoom level, providing a more flexible and personalised experience.

Acceptance Criteria:

- The user can set and adjust the upscaling factor through the application interface.
- The image display updates immediately to show the new upscaled image without having to restart Imogene.
- Invalid inputs trigger an error message and do not continue with the change.
- The upscaling change remains consistent during the current session until the user has changed or cleared the image to restart.

Priority: Medium
- Not a critical function just allows for more flexibility and control by the user.

Estimate: 3 Story Points

1.5:
- Low complexity, effort and risk.

Missing API calls (Restoring hidden buttons) - SHANIA

As a user,
I want to have the access to the same features and controls available in desktop mode as website mode ***so that I*** have access to imogenes full functionality regardless of which mode I choose.

The user story directly addresses the issue of missing functionality and inconsistencies between the two modes of the application. In the original problem many features and controls present in local mode were hidden in remote mode and the GA functionality continued to operate locally rather than being executed through the backend. By expressing the goal of equal functionality across both modes, the user story clearly outlines the need to restore the missing elements and implement the necessary backend API calls to ensure full integration. This phrasing also makes it easier for the development team to understand that the end goal is to provide a seamless and unified experience, ensuring users can access all of Imogene's tools and capabilities regardless of whether they use the desktop or website version.

Acceptance Criteria:
1. All previously hidden UI elements in remote mode are restored and visible.
2. Each button connects properly to the backend using API calls.
3. Feature behaviour in remote matches local mode functionality
4. Error handling is implemented for all API call failures.
5. Testing confirms that both modes have the same features and work correctly.

Priority: High
- This directly impacts user accessibility and usability across both modes.

Estimate: 6pts
- Involves identifying all missing API calls and hidden controls
- Requires both the UI restoration and the backend API wiring and testing work.
- Includes testing to ensure full functionality use.

1.6:

Missing API calls (GA generation to the backend) - SHANIA

As a user,
I want the genetic algorithm (GA) image generation to run on the backend server ***so that*** my device doesn't have to do heavy processing.

The user story highlights how moving the GA to the backend means the server handles the crunching while the device just sends parameters and displays progress and results, therefore would get faster responses, steadier performance and lower local resource use. It also opens up helpful improvements behind the scene such as not forcing every user to have a powerful computer. Overall its user focused yet sufficiently precise for engineers and testers.

Acceptance Criteria:

1. GA generation runs fully on the backend server.
2. The user's device only sends input parameters.
3. The backend handles all heavy processing without slowing down the user's device.
4. Proper error messages are shown if the server fails or times out.
5. Testing confirms that results and performance are consistent.

Priority: High

- Important for improving performance, stability, and user experience.

Estimate: 8pts

- This includes moving the GA process to the server and testing to make sure everything runs smoothly. (setting up communication between the app and backend)

1.7:

~~Load Functionality~~ SOLLIE
As a user I want to upload my own images into Imogene so that I can apply filters or effects to them.

After a discussion with users they felt that it would be helpful to add their own personal images to the app so they can add filters to share with their friends. At the moment users can only generate a random colour or generate each pixel a random colour. The users felt as though this is limiting as there is no personal feature to interact with, furthermore, the users did not see much value in generating random colours. When a user selects a file the image should be visible and be in scale with the current canvas.

Acceptance criteria:

- The user pressed the "Upload Image" button.
- Only allow files with the correct format eg: jpeg, png ect, rejected formats will be greyed out and not able to upload them.
- Resize the image to fit the app's dimension without distortion. ● If this is all successful the image will appear on the canvas

Priority: High - As this task must be completed in this sprint as it is a core functionality for the project.
Estimate: 3 pts - As the task is not too complete, the hardest aspect is resizing the image without distortion.

1.8: Undo Functionality - SOLLIE
As a user I want to completely undo all changes, so that I can revert my image back to its original state

After a discussion with users they expressed that they would like a function to revert any changes made. The users stated that an undo feature is important to take a filter off that they do not like or in case of a mis-click meaning their previous image is not lost. As the moment there is no undo function meaning in the event of a mis-click or a filter the user does not like then the user will have to restart back to the original state and re apply all the filters they want. When the undo button is pressed the canvas should display the previous canvas.

1.8:

Acceptance criteria:

- There is an available undo button for the user to press.
- The canvas displays the previous canvas.
- The user can continue to press undo to keep getting the previous images until the canvas is back to blank.
- If there is nothing to undo then do not change anything

Priority: High - This is a Must Have as users may get frustrated if they cannot undo.
Estimate: 3 pts - As the task is simple with a low complexity, however the logic is not certain as there are a few different ways of getting the undo button working.