

Online Interactive Flag Builder

Szymon Eryk Swendrowski

Department of Computer Science and Information Systems, Birkbeck College, University of

London

Submitted: 05/05/2023

Supervisor: Dr David Weston

Student ID number: 13821605

Abstract

This project explores the creation of an interactive online application that lets the user make flag templates. It will accomplish this by letting the user use common elements of best practice flag design such as horizontal and vertical bars, a border, a cross – hereby referred to as **shapes**, in a drag and drop setting. The user will be able to access the application through a desktop website and will be able to drag and drop selected shapes onto a flag shaped **template**. The website will then let the user click a button that will feed the template to an AI art tool that will output an upscaled, realistic version of the flag to the user. The website will include features like letting the user – upload their own images to use as shapes (if they are in the transparent PNG format), pick colours for shapes from a palette, adjust the scale and position of the shapes on a 2D axis, remove previously dropped in shapes from the template, set the dimensions of their flag as well as its basic shape and, save their template to come back to later.

Contents

Glossary	4
Abbreviations	6
Figure	7
Tables	8
<u>1.0 Introduction</u>	<u>9</u>
1.1 Vexillographic Principles	9
1.2 AI Image Models	11
1.21 Stable Diffusion	12
1.22 Midjourney	13
<u>2.0 Project Aims</u>	<u>16</u>
<u>3.0 Project Objectives</u>	<u>18</u>
3.1 Web Application Architecture	18
3.2 The Front-end	19
3.21 Drag and Drop Application	19
3.22 GUI Design	20
3.3 The Back End	23
3.31 Web Server	23
3.32 Image Storage	23
3.33 Image Upscaling	25
3.4 Software	26
<u>4.0 Methodology</u>	<u>27</u>
4.1 Research and Design	27
4.2 Implementation	27
4.3 Evaluation of Aims	28
4.4 Work Process Flow	28
4.5 Work Plan	29
<u>5.0 Potential Problems</u>	<u>31</u>
<u>6.0 Further Research and Development</u>	<u>32</u>
<u>7.0 Appendix</u>	<u>33</u>
<u>8.0 References</u>	<u>34</u>

Glossary

Vexillology

The study of flags including their types, forms, history, symbolism, and functions. (Institute, 2023)

Vexillography

The art of flag design itself. (Institute, 2023)

Shapes

Basic patterns that flag often share and ones that are recommended for use by the Joint Commission on Vexillographic Principles of the FI and NAVA. (Association, 2014)

Template

The canvas on which the user will drag and drop shapes onto to create their flag's structure.

Observe

The side of the flag that you see when the flagpole is on the left – in western tradition this is seen as the front of the flag. (Association, 2014)

Reverse

Opposite side of the flag to the observe – seen as the back of the flag. (Association, 2014)

Charge/Devices

What appears on the flag – a simple geometric shape, a symbol, a seal etc. (Association, 2014)

Text-to-Image

A machine learning model which takes a natural language description as an input and outputs an image matching that description. (Petrovavlov, 2022)

Single Page Website

A website that contains only one HTML page, instead of being split up into different pages.

(Costa, 2018)

Fork

A program which has been developed and based off the source code of a different program.

(Alexandria, 2023)

Scope Creep

Changes and continuous or uncontrolled growth in a project's scope, after the project begins. (Lewis, 2002)

Abbreviations

FI	Flag Institute
NAVA	North American Vexillological Association
API	Application Programming Interface
VoIP	Voice over Internet Protocol
UI	User interface
SOLID	Single Responsibility Open-Closed Liskov Substitution Interface Independence Dependency Inversion
TDD	Test Driven Development
REST	Representational State Transfer
API	Application Programming Interface
GUI	Graphical User Interface
PNG	Portable Network Graphics
BLOBs	Binary Large Objects
NTFS	New Technology File System

Figures

Figure 1: The basic structure of a hoisted flag	9
Figure 2: LDM deionising process	13
Figure 3: The Midjourney UI	14
Figure 4: Midjourney request	15
Figure 5: Midjourney image prompt	15
Figure 6: Implementation architecture	18
Figure 7: Graphical user interface design principles	22
Figure 8: The degradation in read performance	25

Tables

Table 1: Examples of basic flag design patterns	11
Table 2: Proposed work plan for the project	29
Table 3: Potential problems for the project	31

Introduction

This introduction will cover some of the background topics serving as the basis for the project, specifically the principles of flag design and the currently popular AI art tools on the market.

Vexillographic Principles

Vexillographies basic tenants include minimalism and distinctiveness. A flag should be simple enough that it can be easily drawn by a child but also striking enough to be drawn from memory and not confused with other flags. The three aspects of a flag, the **colour**, **structure**, and **devices** should follow these tenants. (Association, 2014)

Using fewer colours results in a more simple and bold design. Contrast is important so using light on dark or dark on light is preferred. Over the course of human history, the colours used for flags have been restricted by technology, however due to the advent of modern printing techniques there are more colours available than ever before. Additionally, it is important to consider the edge of a flag, this region should stand out to avoid a flag blending in with its surroundings, colour and contrast can make this easier to ensure.

A lot of vexillographic theory involving structure is to do with how the flag flies, and even though this doesn't directly impact our application, it is not unreasonable to expect

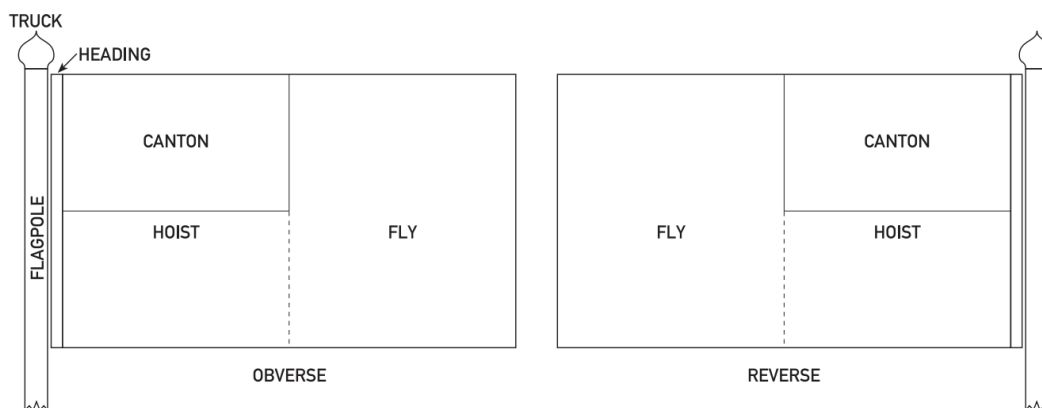


Figure 1: The basic structure of a hoisted flag (Association, 2014)

that a user might want to print out their finished flag and hoist it up. Thus, these principles are still important.








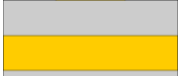
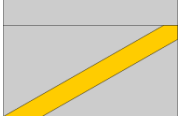



When a flag flies or is at rest the **hoist** – the half of the flag nearest to the flagpole, is the most visible part as compared to the **fly** – the half of the flag furthest from the flagpole. The most prominent part of the flag is the **canton** – the centre, and alongside the hoist, should be where most devices appear since any that are put in the fly would be obscured when the flag is at rest. Another important structural consideration is that most flags are landscape and thus, any flag that deviates from that, requires balancing. Finally for structure, it is important to have a uniform design on the observe and reverse since that undermines both basic tenants, it's complex and costly, and undermines recognition.

The final aspect, devices should follow 4 principles:

1. Preferably **only one**, if more than one use different background colours to make them distinct.
2. They should be **simple graphical representations** of things rather than realistic depictions (this excludes coat of arms as they are too complex).
3. **No writing.**
4. Devices that have a direction should be **moving towards the flagpole** as that is the leading edge.

All the 3 discussed aspects have common patterns across previously designed flags, below is a table of examples of these patterns, all of these, as well as others, will be used as shapes for the application.

Table 1: Examples of basic flag design patterns (*images by M. W. Toews*)

Name	Images
Border/Bordure	
Canton	
Quadrisection	
Greek Cross (Couped Cross)	
Symmetric Cross	
Nordic Cross	
Pale	
Fess	
Bend	
Chevron	
Pall	
Saltire	

AI Image Models

The current landscape of AI art tools is dominated by two AI text-to-image models, **Stable Diffusion** and **Midjourney**.

Stable Diffusion

Stable diffusion is an open-source deep learning text-to-image model that was released in August 2022 and made widely available to the public. This fact, along with the ability to run on most machines locally instead of on the cloud is quite different to most AI art tools of today.

It works by letting users input a text prompt of the image that they wish to generate. Further applications also allow users to input an image and a parameter that determines the degree to which the generated image should resemble the input image.

The generated image is created using a type of diffusion model called a **latent diffusion model** (Group, 2022), hereby referred to as LDM, using information from a training dataset. This training dataset contains pairs of images and captions scrapped from the web, specifically 5 billion images, each filtered by resolution, presence of a watermark and their subjective visual quality – assigned as a predicted “attractiveness” score. (Baio, 2022) The specifics of how stable diffusion works is beyond the scope of this project, but the basics of how the LDM works are discussed below.

The **diffusion** aspect refers to the generation of images by iteratively deionising (i.e., reducing) random noise on the image until a certain number of steps have been reached. This number of steps can be configured as a parameter. The process can be visually seen in the figure on the next page. This process starts out by picking an image from the training dataset that best matches the prompt, then generates some random noise and iteratively reducing this noise until the end condition is reached.

Latent refers to the fact that the model infers the meaning behind the image through the diffusion process by using a mathematical structure called a latent space. This

information speeds up the process as this information is used to generate the final image.

(Robin Rombach, 2022)

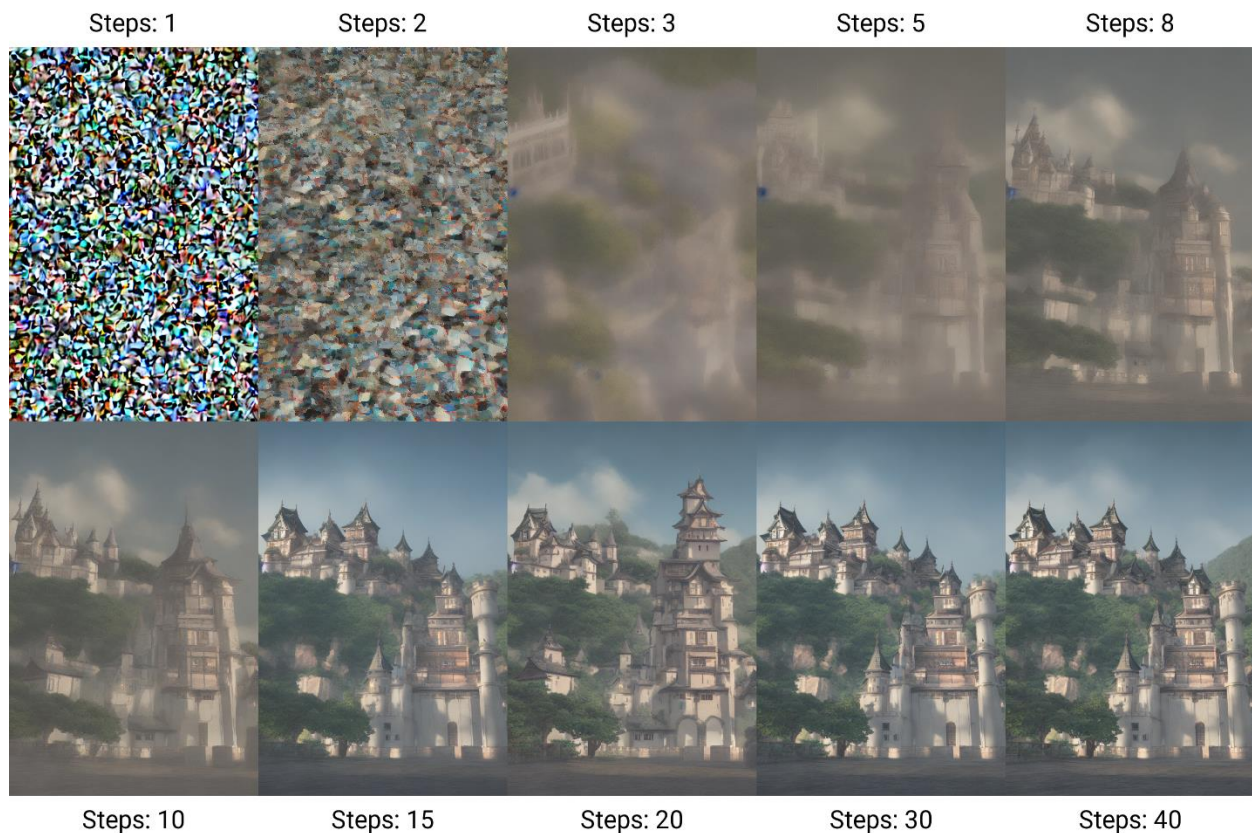


Figure 2: LDM deionising process (*image by Wikipedia user BenliSquare*)

Midjourney

Midjourney is a text-to-image AI program that, according to the creators themselves (Mostaque, 2022), uses underlying technology that is based on Stable Diffusion. The software is currently in open beta and is accessible only through a VoIP and instant messaging social platform called Discord. Specifically, users submit requests by messaging a bot, either on the official Midjourney server or through a third-party server. The requests are of the structure “/imagine prompt”, where the prompt field contains the description of the image the users wish to create. (Midjourney, Inc., 2022)

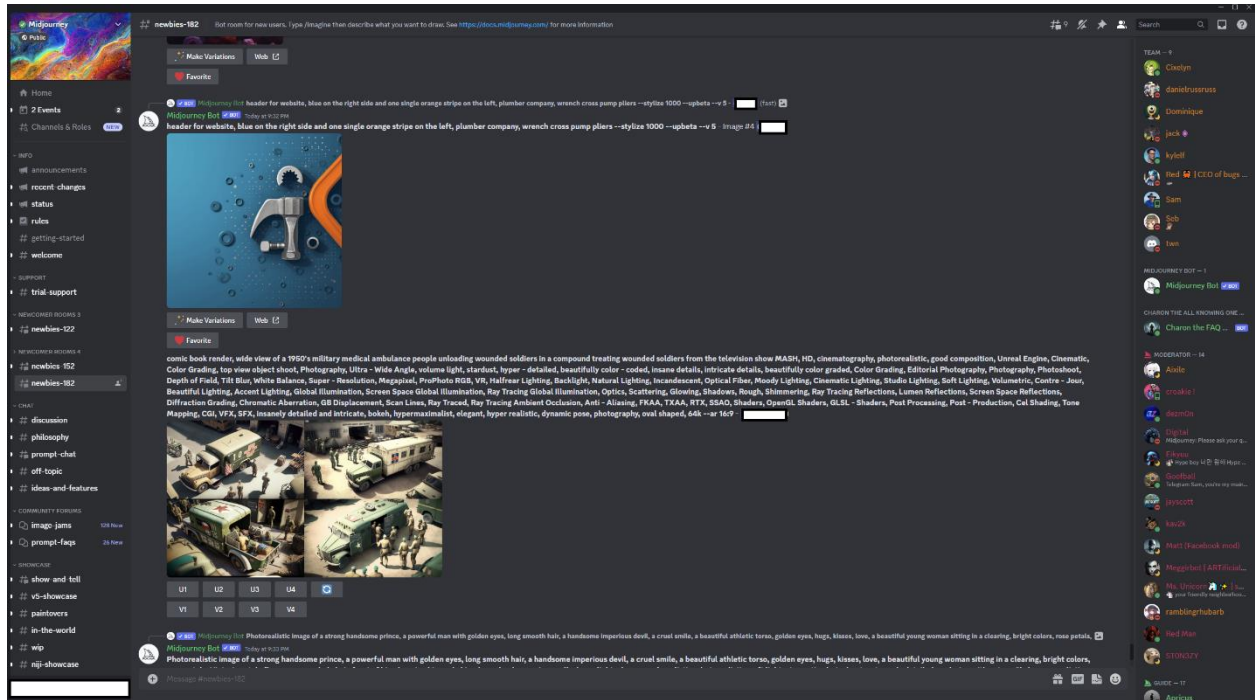


Figure 3: The Midjourney UI

The figure above shows the Midjourney UI as presented through discord. On the left there are channels for users to generate images as well as discuss topics. In the centre there are generated images by other users. Finally, on the bottom there is the message window in which the user can submit requests.

Looking at a request closer, figure 4 shows a request at the top, followed by the output on the bottom. The output is always 4 image variations. The buttons on the very bottom are options for the user. The U options upscale the chosen image (1 top-left, 2 top-right, 3 bottom-left and 4 bottom-right), and the V options make a variation of the chosen image. The refresh icon generates a new set of 4 images based off the same prompt.

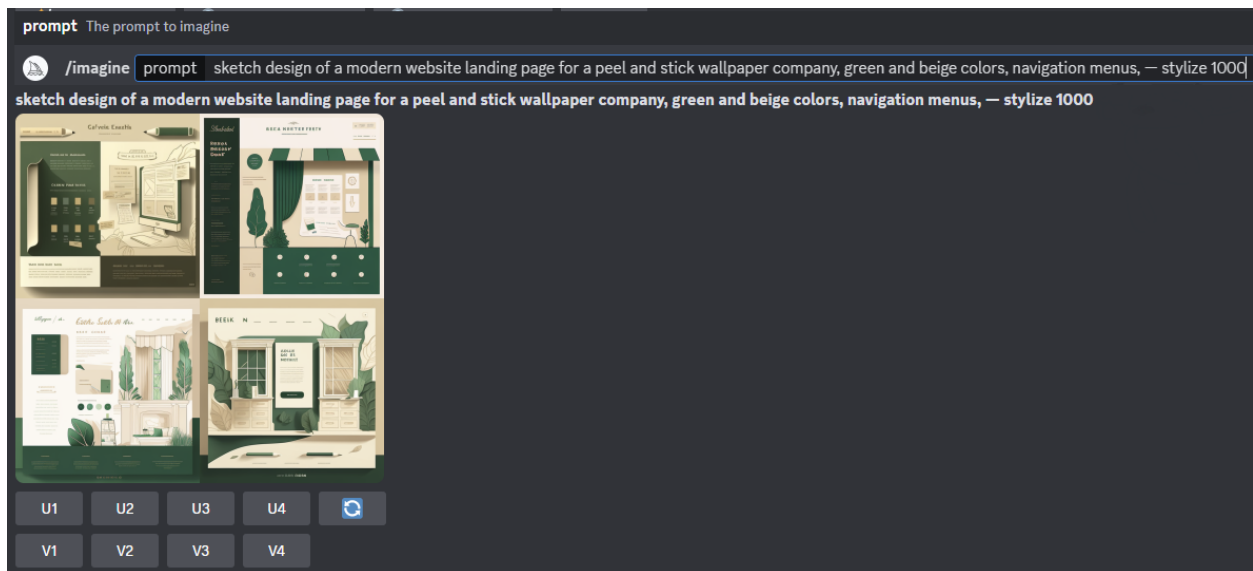


Figure 4: Midjourney request

Midjourney also supports using images as prompts, provided you still submit a text prompt with it, this is shown in the figure below.

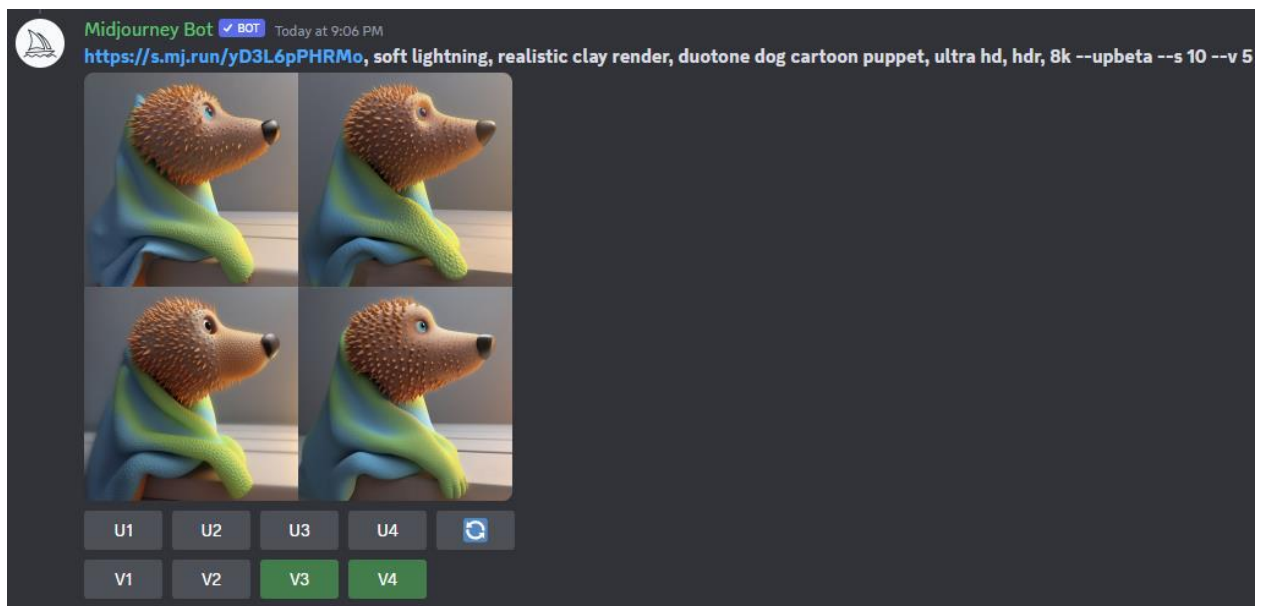


Figure 5: Midjourney image prompt

Project Aims

The primary aim of the project is to build an interactive online experience that lets users create flag templates with a simple drag and drop graphical interface, with the shapes, colours, and options available to users being dictated by vexillographic principles. Users should be able to save their template to come back to later. The interface should be intuitive, look visually pleasing and feel satisfying to use. The aim of the application is to then output a scaled-up version of the flag to allow a user to get an idea for how their template would look like if taken further as a design to make a real flag.

As said by the very creators of some of these AI art tools (Holz, 2022), their main purpose is not to make artists obsolete, but rather give them a tool they can use for idea visualization. As such this project seems to be a valid use case of such tools. Furthermore, commercially this project can show that the relevance of these tools is greater than thought as they can be applied to systems not directly involved in the art market but rather in different fields.

The topic was selected due to not only an interest in flag design and machine learning AI art tools but also as an opportunity to demonstrate and develop further some of the skills learned during the taught modules of the MSc. Specifically,

- Developing a front-end
- Implementing a back-end that communicates with the front-end.
- Creating and managing a system to store the images for easy access.
- Practice using the Java language in a web-based setting by implementing a REST API.
- Practice developing a project using an Agile work methodology.

Additionally, this project will also help with practicing implementing industry standard practice such as the SOLID principles and standard design patterns in code. Considering the before mentioned relevance of AI art tools and machine learning in general, this project also provides an excellent opportunity to develop highly marketable skills.

Project Objectives

To meet the above laid out project aims, the web application must be not only functional but also interactive and responsive. Its primary functions will include:

1. A drag and drop application on the front end that will enable the user to create the flag templates.
2. An implementation on the back end that saves the templates to a storage system.
3. A storage system that contains the templates to allow users to come back to their templates later.
4. A process such that when the user submits their template for upscaling, the server sends the request and outputs the result to the user.

To meet these functional requirements, the objectives below should be considered.

Web Application Architecture

The primary functions can be implemented in the following architecture:

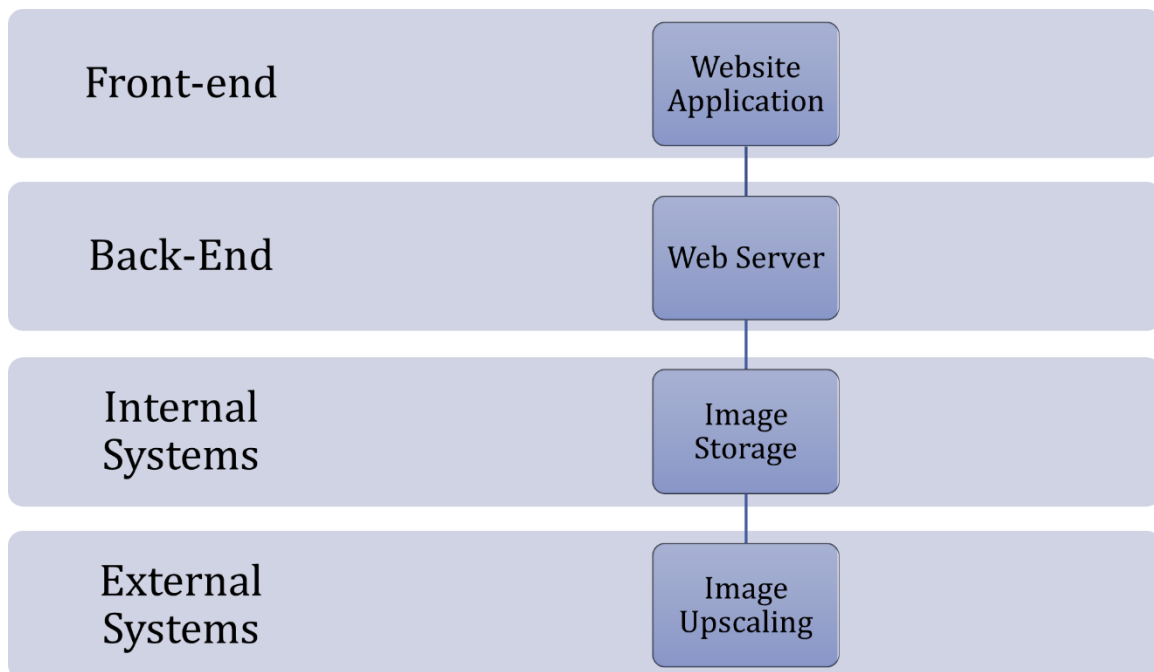


Figure 6: Implementation architecture

The advantage of such an approach is that each of the constituent parts can be developed side by side to make sure they work together seamlessly. The obvious disadvantage is the complexity of such a system creates various points of potential failure, such as the templates not being retrieved correctly by the back end to the front-end or the request sent to the API not working. These can be minimized using TDD to spot functionality issues and correct them during the development phase (Cloudkasten, 2016). An alternative would be to develop everything on one system, such as using Java to develop both the application itself, a GUI for the user to interact with it and the functionality of a REST API to upscale the templates. Though much less complex comparatively, making it easier to implement, one could argue that it does not entirely meet the project aims. It would be far from interactive and responsive while also missing out on allowing demonstration and development of the skills and technologies learned on the MSc.

The Front-end

Drag and Drop Application

The interface by which the user will create the flag templates will have all the tools needed such as the functionality to drag and drop shapes onto the template canvas, resize them after they're on the canvas as well as changing their colour. Additionally, the user will be able to erase shapes already on the canvas, as well as interlay shapes on top of each other, for example a symbol on top of vertical bars. The position and size of all these shapes will be kept track of using global events.

Some shapes will be free flowing and can be placed anywhere on the canvas, like crests, crosses etc. However, the more geometric shapes like canton and quadrisection will snap into their respective sections. This will be accomplished through zones on the

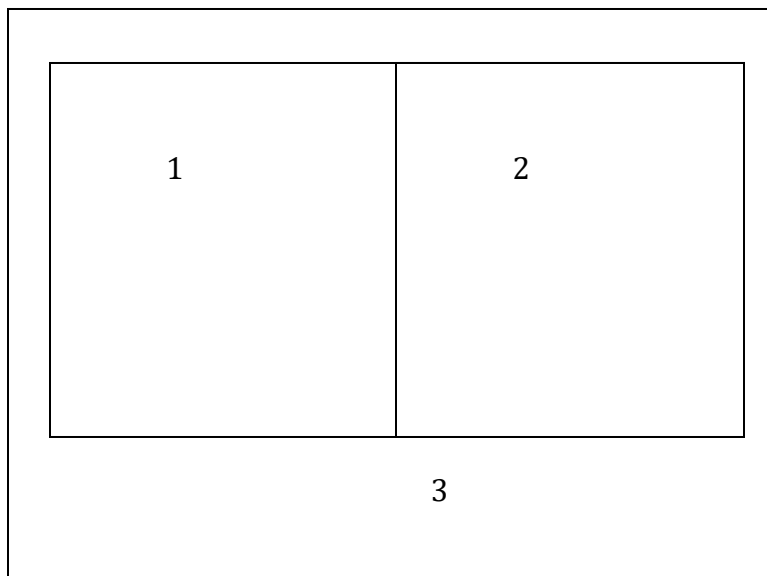
template that will change size and position depending on the aspect ratio and resolution chosen by the user. The user will be able to upload custom transparent PNG to use as a shape (a free flowing one specifically).

All of this will be implemented on the front-end as this will allow the application to be fast and responsive – which is key in meeting the project aims.

GUI Design

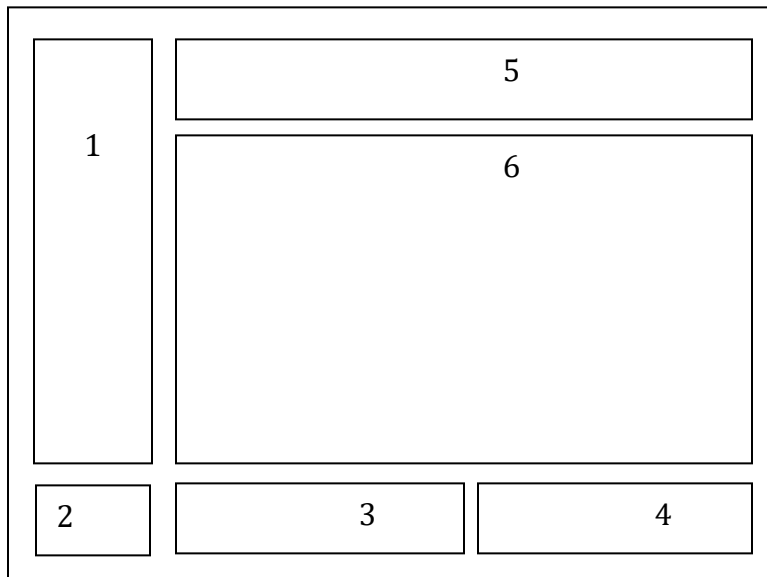
The following set of diagrams represent the GUI design that will allow the user to interact with the application. It is important to note that even though these are labelled as pages, the design will be a single page website to meet the aim of making the website easy to use.

Start Page:



1. Information on how to use and interact with the application.
2. Drop town menu of different aspect ratio and resolutions for templates.
3. The main application in the background greyed-out.

Main Page:

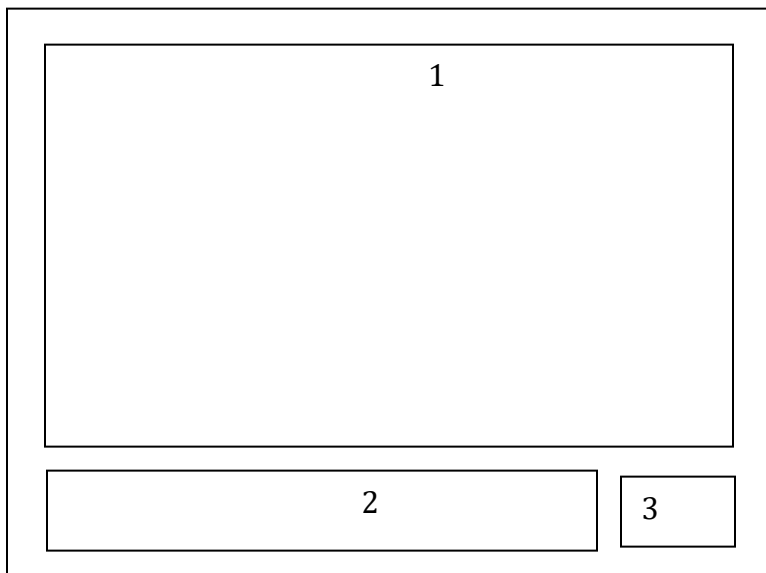


1. List of shapes to choose from.
2. Button to submit transparent PNG to use as shape.
3. Tool buttons such as an eraser tool, erase all tool etc.
4. Buttons to save the

template to come back to later, start a new template by taking the user back to the start page and to submit their current template for upscaling.

5. Colour palette to set colour of selected shape or shape to be selected.
6. Window displaying the template itself.

Page After Submission for Upscaling:



1. Window displaying generated image.
2. Options to generate image again as well as help information and the option to go back to template.
3. Button to save generated image.

The key points to consider for any potential GUI design are the following: (Afshan Ejaz, 2019)

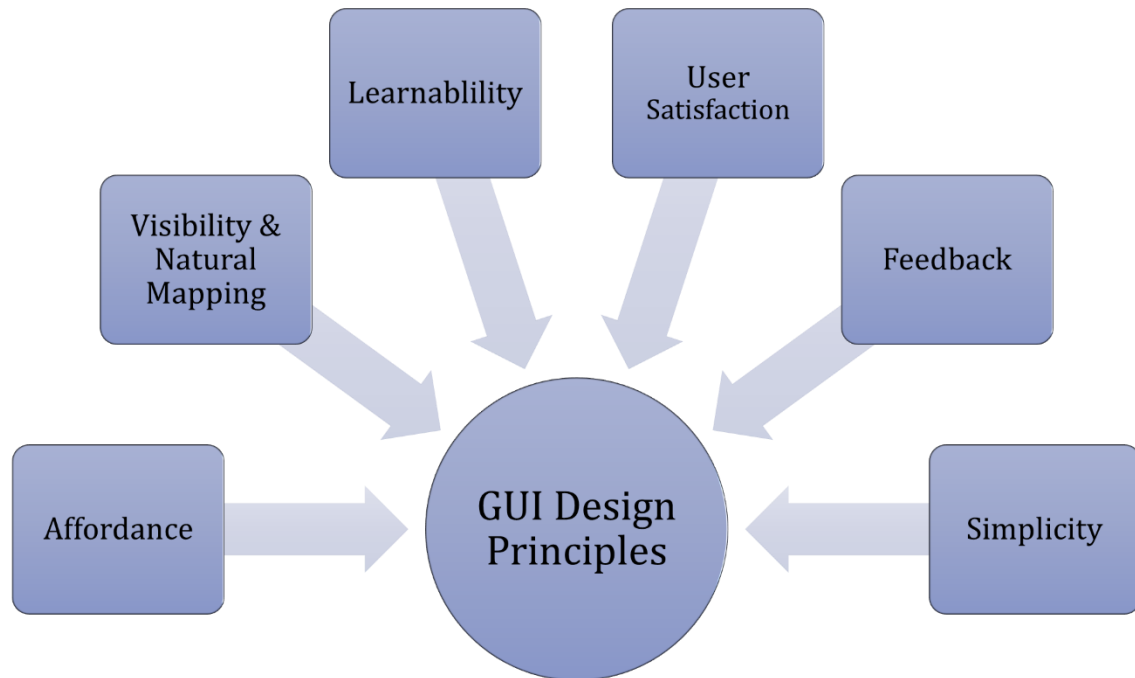


Figure 7: Graphical user interface design principles

Each of these can be explained as follows:

- **Affordance:** The user should be able to tell what something does just by looking at it.
- **Visibility & Natural Mapping:** The interface should follow the common standards and patterns which help the user interact with itself by letting them intuitively understand it.
- **Learnability:** The user should be able to use the system easily.
- **User Satisfaction:** The user experience should be positive.
- **Feedback:** The interface should communicate what is happening to the user.
- **Simplicity:** The interface should be simple as to be efficient

An example of some of the following would be ensuring that when resizing shapes on the template, feedback is provided to the user about how much they're resizing the shape as well as, providing an indicator showing a double arrow when hovering over the corner of the shape to confirm to the user the function of this tool.

Conforming to these principles will ensure that the project aim of providing an interactive and responsive application to the user is met.

The Back End

Web Server

Initially the website will be hosted on a local server to facilitate testing and make the implementation easier to build. As the development process progresses, it will be hosted on a web server to allow access from non-local machines. It will manage the upload and download of images from the client-side application to the server-side image storage as well as interactions between the front-end and the back end.

Image Storage

On the issue of storing the images involved, like the templates and the final images, there are two possible choices, storing them in a database or in a file system. The consideration as to which one to choose is based on two factors: the image sizes and the volume of overwrites.

In terms of image sizes, due to the limits applied by the application to image resolutions and image colours – specifically their **bit depth** (the number of bits needed to store colours) (Future Learn, n.d.), it's not hard to imagine that the templates themselves won't have significant file sizes, less than 1 MB but certainly more than 256 KB. This is especially true when you also consider image compression. Additionally, most Stable

Diffusion APIs also limit their generated images resolution, as well as compressing them.
(Robin Rombach, 2022)

The templates will be overwritten as a user might work on their template multiple times and comeback to it progressively. However, it is also true that once the template is used to generate an image, even if used to generate an image again, it will generate a different image. Thus, the number of overwrites wouldn't be too large but is significant.

Using findings from a paper by Microsoft Research (Russell Sears, 2006), one arrives at the following conclusion. Reading image files is more important than writing them since the write can be done in the background by the server while a read affects how quickly a user can load their saved template. Looking at the findings of the paper, at the size of objects (such as the images of the templates) between 512KB and 1MB the file system outperforms the database after two and four overwrites as seen in the following figure.

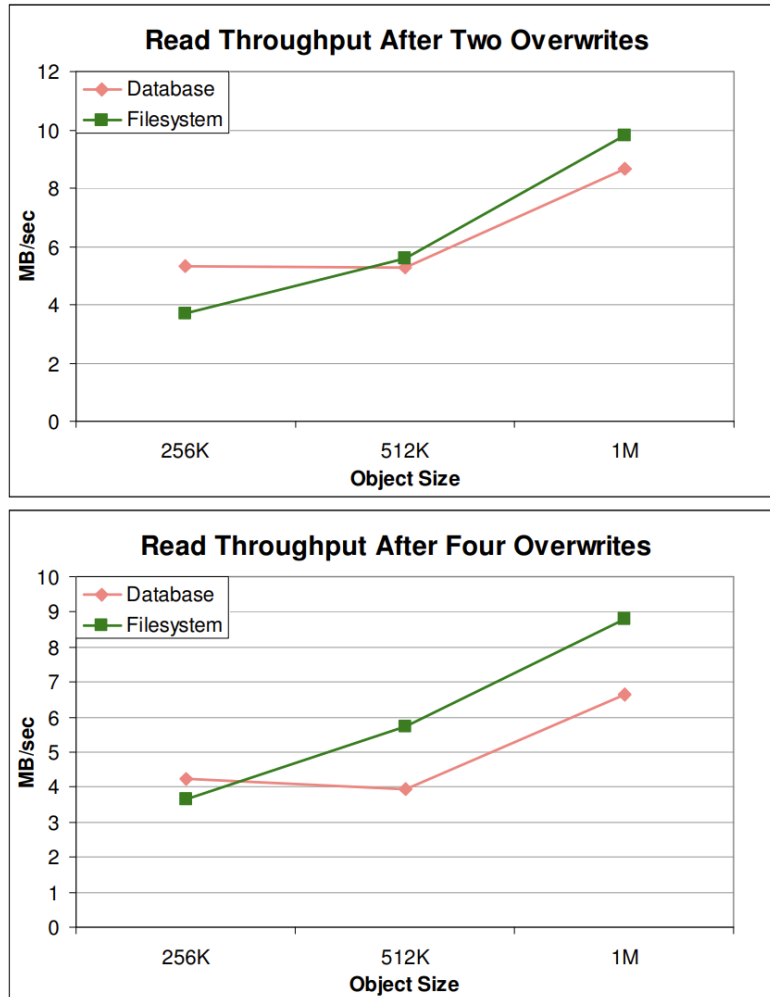


Figure 8: The degradation in read performance for 256K, 512K, and 1MB BLOBS (*Russell Sears, 2006*)

Thus, the images for both the templates and the generated images will be in a NFTS.

Image upscaling

Stable diffusion is the more suitable choice for the generation of images as it is open-source and thus, has many widely available APIs. This suits this application better as it will allow requests to be submitted automatically by the back end, instead of having to manually submit them through Midjourney's discord interface. It is also important to consider the point of prompt optimisation – testing will need to be done to determine the prompt that generates the best images.

Software

In terms of software, the choices considered need to strike the right balance of functionality and ease of use. An additional point to consider is how they relate to the project aims, especially the aim of demonstrating skills learned on the MSc. Taking all of this into account below are some of the software choices for this implementation as well as the reasons for their choice.

For the front-end website HTTP, JavaScript and CSS will be used and for the back-end server implementation, PHP will be employed with XAMPP used for local testing. These are the languages and tools taught on the Internet and Web Technologies module in the MSc, and developmental experience with them has already been attained during the coursework for this module. Specifically, these languages have cross-functionality that makes them easy to integrate into a working application together.

For the drag and drop application itself, JavaScript libraries and frameworks can be useful, as they are well suited for developing such applications. The question of popularity is an important one since popular frameworks are much more marketable as skills since more developers use them. React is by far the most popular front-end framework (See Appendix A for details). Furthermore, it has a lot of open-source utilities that make it easy to create complex drag and drop functionality. (React-DnD)

The REST API functionality will be implemented using the Java language as this was how it was introduced and taught on the Software Design and Programming module in the MSc and so is best aligned with the project aims.

Methodology

Research and Design

This exploratory phase will involve deeper research into:

- Looking at how other drag and drop applications implement their functionality and how their implementations can influence this one.
- Browsing and analysing different websites employment of GUI design to ensure the GUI design chosen is suitable.
- Researching different implementations of NFTS in a web-based setting and which one is the most appropriate to use here.
- Comparing different Stable Diffusion APIs to pick the one best suited for the implementation.

In terms of design, a more detailed development timeline will be drawn up.

Implementation

The implementation will involve sprint like cycles of developing functionality for each of the components of the system iteratively, starting with a basic prototype in the first sprint to the finished implementation in the last sprint. The advantage of such a work structure is that it lends itself well to TDD and allows the system parts to be developed together simultaneously to ensure correct functionality.

Phase 1 – Exploratory phase

The first phase will involve getting all the necessary tools for the cycle ready – such as software, resources, and planning. Aims of the cycle will be finalised and planning required such as designing class diagrams will be performed.

Phase 2 – Front-end client development

The second phase will involve development of the drag and drop application as well as the website it is hosted on.

Phase 3 – Back-end server development

The third phase is all about developing the server component of the architecture, to make sure it can communicate with both the front-end, the image storage solution and the back-end API.

Phase 4 – Rest API development

For this phase, the REST API that sends the templates to the image upscaling implementation through Stable Diffusion will be developed.

Phase 5 - Checklist

This final phase will involve checking that the implementation of the current iteration meets all the aims outlined in the first phase and will also involve some final testing before moving on to the next cycle.

Evaluation of Aims

The functionality of each iteration of the application will be evaluated against the project aims. If time permits this will be done iteratively as the project goes on after the completion of each prototype developed. Evaluation will include presentation, ease of use and functionality relating to the project aims. Furthermore, it will be ensured that sufficient documentation is present to make development more streamlined.

Work Process Flow

To ensure a good quality project, a log will be kept that tracks the progress of the project overall. Additionally, regular meetings with the project supervisor will be held to

maintain a steady development speed. Both have the added benefit of allowing the opportunity for continuous self-evaluation as mentioned in the previous section. A timetable will be drawn up of a work plan that is more detailed and outlines targets, prioritising the most important application features to ensure the project meets the aims outlined in this proposal.

In terms of the actual software development, as mentioned before a TDD approach will be used to ensure functionality issues are spotted early and debugged through testing. Additionally, decoupling of code and reducing dependencies will be used which result in lower maintenance and modifications costs (Stevens, Myers, & Constantine, 2000). As previously discussed, SOLID principles as well as design patterns (Erich Gamma, 1994) will be employed to make the code easily extendable, and easily modifiable. To further ensure this, it is also advantageous to use dependency injection and reflection where possible.

Work Plan

The table below shows the outlined work plan split up into weeks. The write up will occur simultaneously while the project is being developed. The benefit of such a parallel workflow is the opportunity to evaluate and reformat the project as recommended by the project supervisor.

Table 2: Proposed work plan for developing the project

Time Period	Plan
8th May – 14th May	Research and Design – complete the research and tasks outlined in the section.
15th May – 21st May	Start of Implementation – first cycle , design a basic version of the front-end.
22nd May – 28th May	Finish designing front-end and start developing a functional back end with the basic architecture for the REST API.
29th May – 4th June	By this stage there should be an initial prototype ready with basic functionality

	and a bare bones implementation architecture.
5th June – 11th June	Start of second cycle – refine drag and drop functionality.
12 June – 18th June	Continue refining the drag and drop and make sure it is integrated with the back end.
19th June – 25th June	By this stage the application should have most of the basic features on the front-end.
26th June – 2nd July	Start of third cycle – finish work on the front-end from the previous cycle and then focus on refining the back end.
3rd July – 9th July	Integrate most of the REST API functionality into the architecture.
10th July – 16th July	By this stage the back end should be mostly complete and functional.
17th July – 23rd July	Start of fourth cycle – refine front-end and back end functionality.
24th July – 31st July	Finish integrating the REST API with the application and make sure it all works together.
1st August – 6th August	By this stage the REST API functionality should be fully completed, and the website should be accessible from non-local machines.
7th August – 13th August	Start of the fifth final cycle – this phase is mostly for polishing and refining all the components.
14th August – 20th August	Make sure all the systems are fully working together and that the application is ready to use.
21st August – 27th August	By this stage the application should be fully integrated and be at a presentable level.
28th August – 3rd September	Debug and perform final tests.
4th September – 10th September	Evaluate project aims fully.
11th September – 17th September	Format report in desired format and submit when ready.
18th September (Deadline)	

Potential Problems

The table below outlines some of the potential problems that can occur during the development of the project as well as some of the preventative measures that will be undertaken to reduce them.

Table 3: Potential problems for the project

Problem	Preventative Measure
Disorganized development	<ul style="list-style-type: none">• Use the agile methodology outlined to ensure focus is maintained on basic functionality.
Time management issues	<ul style="list-style-type: none">• Follow iterative workflow to make sure at each stage there is a working application.
Scope creep	<ul style="list-style-type: none">• Stick to the refined work plan finalize in the research and design phase.• If good new features are thought of, document instead of implement.
Unexpected complexity	<ul style="list-style-type: none">• Prioritize basic features over the more complicated ones, especially in initial stages of development.• If features cannot be implemented, document why this is and how they could be implemented with more time and resources.
Unsuitable architecture	<ul style="list-style-type: none">• First stage of methodology outlines that class diagrams will be drawn to ensure architecture is suitable.• Prioritize basic functionality over complex architecture.
Cost of API	<ul style="list-style-type: none">• Use free API alternative.• Use Basic Tier to lower costs.• Enquire about educational license (making sure to use Birkbeck email address and CC project supervisor).
Cost of server hosting	<ul style="list-style-type: none">• Use free alternatives like 000WebHost.• Take advantage of free trials just to demonstrate the website works.• If the above don't work, only implement on local machine

Further Research and Development

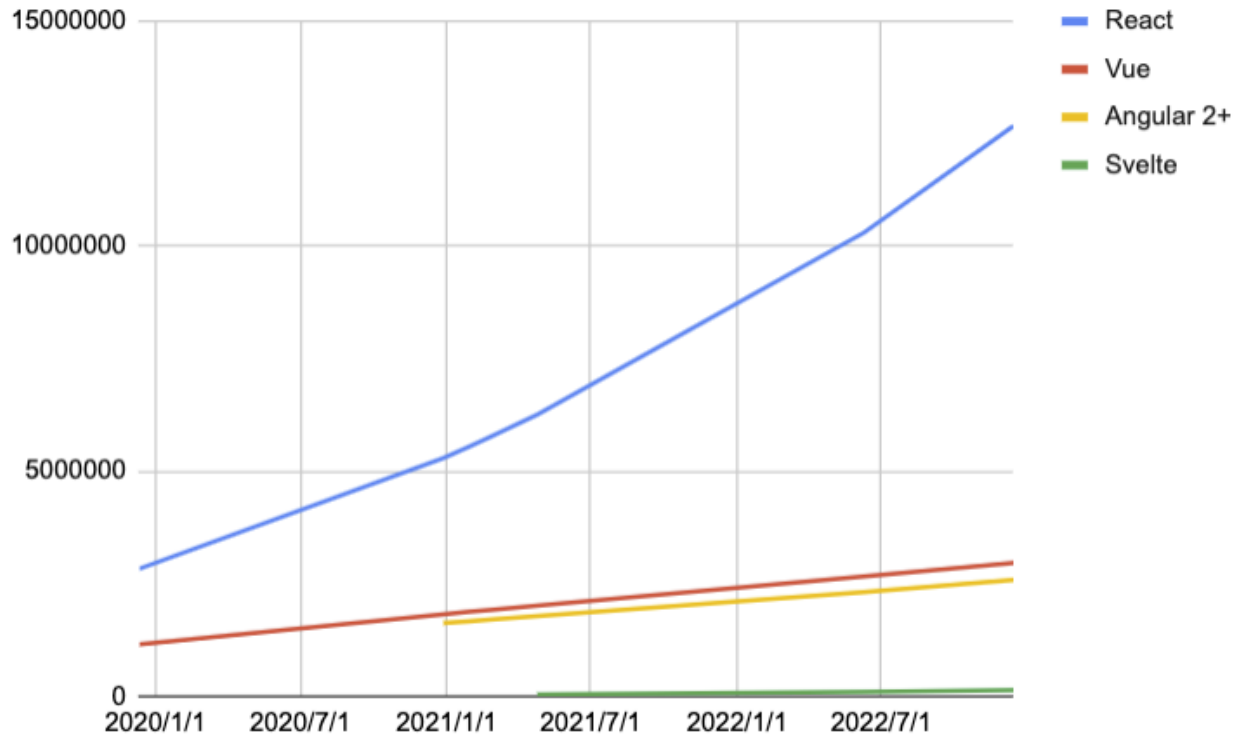
An obvious idea to take the implementation further is to allow users to be more involved with the image upscaling aspect, perhaps through being allowed to select what prompts will be fed into the process. Additionally, one could also implement a system that allows users to save multiple templates to come back to later, however considering that the project is already complex enough, only one template will be allowed to be worked on at a time by a single user in this implementation.

Even though this implementation deals with allowing users to design flag templates this approach could be applied to other designs as well. For example, a potential interested party might be logo designers, they could take the principles of logo design and use them to create a similar templating tool that could output scaled up logos to show them how their prototypes would look like if turned into reality. It's not hard to imagine such a tool would be economically advantageous as it would streamline the design process for corporations and let them allocate the saved time elsewhere.

Stable diffusion itself isn't specifically designed to be used to generate images of flags and so a model could be developed that is specifically trained on images of flags so that it produces better results. This general idea of optimising stable diffusion for different implementations has been done before by creating forks. For example, Thomas Moore created a fork of stable diffusion to generate tiled images – images that can be used in seamless repeating patterns (Moore, 2022) (See Appendix B for examples).

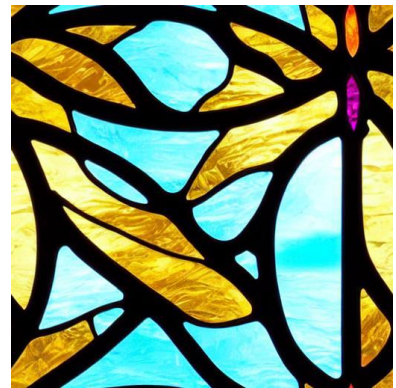
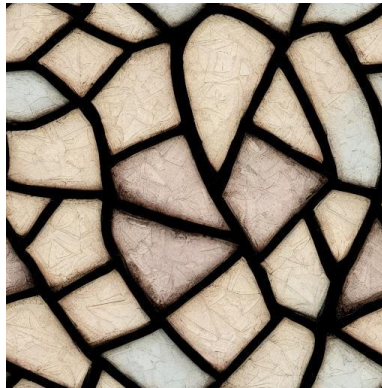
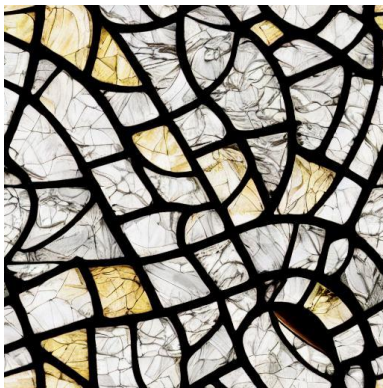
Appendix

Appendix A - Number of Github repositories dependent on different JavaScript frameworks. (tkrotoff, 2019)



Appendix B - Examples of images generated by Thomas Moore's Material Stable

Diffusion with the prompt: *Stained glass window, base color, albedo, 4k*



References

- Afshan Ejaz, D. S. (2019). Graphic user interface design principles for designing Augmented Reality applications. *International Journal of Advanced Computer Science and Applications*, 209-216.
- Alexandria. (2023). *Fork (Software) Definition*. Retrieved from CoinMarketCap: <https://coinmarketcap.com/alexandria/glossary/fork-software>
- Association, F. I. (2014). *Guiding Principles of Flag Design*. Retrieved from https://www.flaginstitute.org/pdfs/Flag_Design_Commission_Report.pdf
- Baio, A. (2022, August 30). Exploring 12 Million of the 2.3 Billion Images Used to Train Stable Diffusion's Image Generator. *Waxy*.
- Cloudkasten. (2016). *Test driven development (TDD) as an approach for better software*. Retrieved from Cloudkasten Corporation Web site: <https://cloudkasten.net/test-driven-development/>
- Costa, R. (2018, December 23). *Single page vs multi-page websites: Design battle!* Retrieved from Justinmind: <https://www.justinmind.com/blog/single-page-vs-multi-page-websites-design-battle>
- Erich Gamma, J. V. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Future Learn. (n.d.). *File size calculation*. Retrieved from Future Learn: <https://www.futurelearn.com/info/courses/representing-data-with-images-and-sound/0/steps/53142>
- Group, C. -M. (2022, September 17). *Stable Diffusion Repository on GitHub*. Retrieved from GitHub: <https://github.com/Stability-AI/stablediffusion>

- Holz, D. (2022, August 1). David Holz, founder of AI art generator Midjourney, on the future of imaging. (T. Claburn, Interviewer)
- Institute, T. F. (2023). *About Vexillology*. Retrieved from Flag Institute:
<https://www.flaginstitute.org/wp/about/about-vexillology/>
- Lewis, J. P. (2002). *Fundamentals of project management*. New York; Toronto: AMACOM.
- Midjourney, Inc. (2022). *Quick Start*. Retrieved from Midjourney Docs:
<https://docs.midjourney.com/docs>
- Moore, T. (2022, August 22). *Github*. Retrieved from Material Stable Diffusion Repository:
https://github.com/TomMoore515/material_stable_diffusion
- Mostaque, E. (2022, August). *Twitter*. Retrieved from Tweet by Emad Mostaque:
<https://twitter.com/EMostaque/status/1561917541743841280?s=20>
- Petropavlov, K. (2022, December 28). *Text-to-image AI models and how we replaced designers with machines*. Retrieved from LinkedIn:
<https://www.linkedin.com/pulse/text-to-image-ai-models-how-we-replaced-designers-kirill/>
- React-DnD. (n.d.). *React DnD Docs*. Retrieved from GitHub: <https://react-dnd.github.io/react-dnd/about>
- Robin Rombach, A. B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models (A.K.A. LDM & Stable Diffusion). *Interdisciplinary Center for Scientific Computing*.
- Russell Sears, C. v. (2006). *To BLOB or Not To BLOB: Large Object Storage in a Database or a Filesystem?* Redmond, Washington: Microsoft Research, University of California at Berkeley.

Stevens, W. P., Myers, G. J., & Constantine, L. L. (2000). Structured design. *IBM Systems Journal*.

tkrotoff. (2019). *Front-end frameworks popularity (React, Vue, Angular and Svelte)*. GitHub Gist.