IMAGE PROCESSING USING WEBASSEMBLY

A PROJECT REPORT

Submitted by

Mayank Singh Tomar - 201B153

Himanshu Tiwari - 201B373

Chirag Jain - 201B087

Name of Supervisor: **Mr. Navaljeet Singh Arora**

Submitted in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

at

JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY,
GUNA, MADHYA PRADESH (INDIA) - 473226

December 2022

DECLARATION

We hereby declare that the work entitled "**IMAGE PROCESSING USING WEBASSEMBLY**", submitted for the B. Tech. (CSE) degree is our original work and the project has not formed the basis for the award of any other degree, diploma, fellowship or any other similar titles.

Mayank Singh Tomar(201B153)

Himanshu Tiwari(201B373)

Chirag Jain(201B087)

Jaypee University of Engineering and Technology,

Guna, Madhya Pradesh (India) - 473226

Date:

CERTIFICATE

This is to certify that the project titled "**IMAGE PROCESSING USING WEBASSEMBLY**" is the bonafide work carried out by **Mayank Singh Tomar, Himanshu Tiwari** and **Chirag Jain**, a student of B Tech (CSE) of Jaypee University of Engineering and Technology, Guna (M.P) during the academic year 2022-23, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (Computer Science and Engineering) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar tile.

Signature of the Guide

Jaypee University of Engineering and Technology,

Guna, Madhya Pradesh (India) - 473226

Date:

ABSTRACT

This is a web-based Image Processing web app that runs on the principle of *WebAsssembly.* Its main functionalities include applying proper transformation on the image. The user is given with option of selecting the way to add images via camera, or computer or check functionalities on default images.

WebAssembly or *Wasm* defines as a portable binary-code format and a corresponding text format for executable programs as well as software interfaces for facilitating interactions between such programs and their host environment; i.e. *Wasm* enables high-performance application on web pages. Therefore, it provides apps with a high optimistic rate and better control over the content, helping the app in reaching the best performance. Due to the easy compatible nature of webassembly our app works on various cross platforms enabling users for the best possible experience. In contrast, providing various tools at high speed while maintaining user data security. We designed our app in such a manner that it is an easy-to-read project making it highly modifiable. The current version of the app only includes basic functionalities like crop, rotate scale and blur effect, etc., and has the accessibility to add other features in the future like cartoonify, color change, etc. making our app highly customizable. This app is made keeping various conditions in mind and is only possible with the help and support of various open-source organizations, W3Schools, and other article-providing platforms.

ACKNOWLEDGEMENT

We would like to express our gratitude and appreciation to all those who gave us the opportunity to complete this project. Special thanks are due to our supervisor **Mr. Navaljeet Singh Arora** whose help, stimulating suggestions, and encouragement helped us in the time of the development process and in writing this report. We would also like to thank our parents and friends who helped us a lot in finalizing this project within the limited period.

Thanking you

Mayank Singh Tomar(201B153)

Himanshu Tiwari (201B373)

Chirag Jain(201B087)

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Problem Definition

Image transformation using traditional techs like photoshop or any other open source software used to phase different image latency issues like large transaction time, lowering image quality, data loss, etc. The project is based on introducing new techs to relinquish the issues faced by the older techs. The web provides a way to reduce or eliminate the issues like time delay, data loss, and image quality. We require many image processing applications in our daily life from work related to our daily life necessities making these applications our basic necessities. Thus, our application based on webassembly will provide a better and more effective to process images in a more accurate and hygienic way while maintaining user data privacy policies. Furthermore, Our application will be able to overcome the limitation of different devices as it can provide almost the same specifications on whether the device is mobile or PC helping users to not face any device incompatibility issues and making compromises in working and daily life. Thus, helping in reaching our goal of providing an every-device comfortable app that can serve its purpose exclusively and make it as much as user and developer friendly.

1.2 Project Overview

The project "**Image Processing Using WebAssembly**" mainly focuses on-

1. Reducing time delay while processing images.
2. Reducing image quality loss on the image while processing images.
3. Making the least possible user interference while making it user-friendly.
4. Proving users with a secure environment while maintaining a user privacy policy.
5. While making it device friendly and easily interactable on various devices equally.

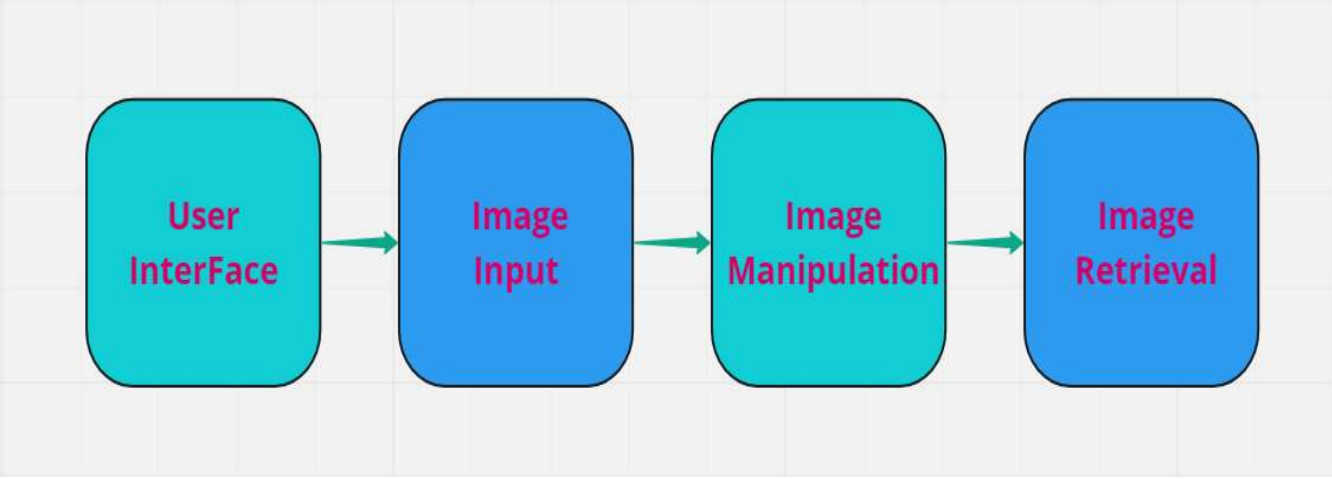


Figure 1. Block Diagram

1.3 Hardware Specifications

CPU (3.0 GHz or faster)
Processor (64-bit Dual Core processor Intel Core-3)
Memory (4GB++(DDR4) RAM And 64GB++ ROM)
Unix Or **Windows** enabled
SSD or External device storage
Camera for capturing Images

1.4 Software Specifications

1.4.1 Backend

Operating System (Linux-Ubuntu 18.04 LTS or 20.4 LTS)
Windows Visual Studio 2022 (16.11 LTS or 17.4 LTS)
RustUp (1.41 or 1.74)
Wasm-Pack (0.10.3)

1.4.2 Frontend

Browsers
Chrome (80.0.3987.116 to 108.0.5359.72)
Safari (version 12 to version 15)
Explorer (explorer 10 to 11)
Firefox (80.0 to 107.0)

CHAPTER 2
LITERATURE SURVEY

2.1 Existing Systems

There are many existing systems like adobe Photoshop, photo-editors using compiling languages like python, javascript, C#, c++, java, etc. However, there are issues that include high build size, and more device compatibility issues, which provide one thing than miss another one.
These systems provide the least possible user interference making it less user-friendly and hard to use while also risking user's data loss in the process as all are backend based.

2.2 Proposed System

Our Image manipulation app works on the backend less process contradicting all the above-mentioned issues such as time latency rate, heavy build file, heavy packages, device compatibility, etc. The process is straightforward and simple consisting of steps like user input and checking backends build for the sake of functionality and later applying the functionality to the image on hand and retrieving the image via a single click. We neglected multiple processes that other systems include such as the database route process, converting into cookies and any other applications on images other than the asked functionality. Because of this our app provides a much faster application while maintaining user privacy and data loss issues.

2.3 Feasibility Study

Financial Feasibility: The cost of image manipulation is almost non-existent and also doesn't require any heavy or fancy hardware tools making it charge-free. The software it uses is almost all commonly used software which is open source above making it both a user and developer-friendly product.

Economic Feasibility: As per the other alternatives available in the market as well as the existing software in the market the economic feasibility of our project exceeds or equally stands making it economically viable. The growth project may differ from other products as it is mostly an open-source project for a better user experience.

Technical Feasibility: Almost all the software used in the process of compiling as well as building are open source-based products making them freely available to us. Techs used here are all new techs in comparison to techs and langs used in other products making the user able to contribute to their growth while benefiting at the same time, making the project extremely technically feasible.

Operational Feasibility: Operational feasibility is the measurement of how the problem described in the problem definition is solved by the proposed system. It can only be measured via the user feedback process as the user uses the product and advice any further changes or improvements. We have made our app in such a way that user-suggested changes can be applied interactively. User-friendliness is also one of the major coefficients for calculating operational feasibility and our application stands affirmatively in user experience as the user gets to see a ware minimum of tech stuff on the front end making it easy to operate by any form of user.

2.3.1 Comparison

On Comparing our tech with other techs present in the market we get shocking results, showing the ways we can further improve our project and showing our standing in the market on basis of technology. The comparison is between existing systems present in the market to our product on the stand of the same function applied by all functions while keeping in mind certain criteria on which we carry out our measurement.

Existing Sys ➡️ Functions ⬇️	Python Imaging Lib (Using Py)	ImageMagick (Using C)	Our App (Using WASM+Rust)
On The Basis of average Time Taken			
Crop	40 mili-secs	22 mili-secs	8 mili-secs

Flip	20 mili-secs	12 mili-secs	7 mili-secs
Scale	25 mili-secs	14 mili-secs	8 mili-secs
Rotate	23 mili-secs	15 mili-secs	6 mili-secs

NOTE: These measures are on the average basis including determining conditions like system checked on.

Table 1. Table of comparison of existing systems

2.3.2 Comparison Results

As the table shows us that time latency or time taken in the process is extremely reduced as compared to any other processes; while we can also see that image quality loss is less in the process as compared to other process telling us that our process deals least data loss. On the other hand, we are also not using any form of database or anything that deals damage to users privacy making it more user-friendly compared to its peers and it can be feasible to use on various devices making its user experience much better than others. These results tell us that our problem definition is successfully triggered and we were able to achieve better results than expected.

CHAPTER 3
SYSTEM ANALYSIS & DESIGN

3.1 Requirements Specification

3.1.1 Frontend

JavaScript

JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB, and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

ReactJS

The React. js framework is an open-source JavaScript framework and library developed by Facebook. It's used for building interactive user interfaces and web applications quickly and efficiently with significantly less code than you would with vanilla JavaScript. In React, you develop your applications by creating reusable components that you can think of as independent Lego blocks. These components are individual pieces of a final interface, which, when assembled, form the application’s entire user interface.

CSS

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML, or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media. CSS is among the core languages of the open web and is standardized across Web browsers according to W3C specifications. You might have heard about CSS1, CSS2.1, or even CSS3. There will never be a CSS3 or a CSS4; rather, everything is now CSS without a version number.

WebPack

Webpack is a free and open-source module bundler for JavaScript.It is made primarily for JavaScript, but it can transform front-end assets such as HTML, CSS, and images if the corresponding loaders are included. Webpack takes modules with dependencies and generates static assets representing those modules.Webpack is a static module bundler used for JavaScript applications. Since webpack understands only JavaScript and JSON files, It transforms front-end assets such as HTML, CSS, and images into valid modules if the corresponding loaders are included. While Processing your application webpack internally builds a dependency graph that maps every module your project needs and produces one or more output bundles. Webpack takes the dependencies and generates a dependency graph allowing web developers to use a modular approach for their web application development purposes. It can be used from the command line or can be configured using a configuration file which is named webpack.config.js. This file defines rules, plugins, etc., for a project.

Redux

Redux is a predictable state container designed to help you write JavaScript apps that behave consistently across client, server, and native environments, and are easy to test. While it's mostly used as a state management tool with React, you can use it with any other JavaScript framework or library. It's lightweight at 2KB (including dependencies), so you don't have to worry about it making your application's asset size bigger. Redux is simply a store to store the state of the variables in your app. Redux creates a process and procedures to interact with the store so that components will not just update or read the store randomly. Similar to the bank. It does not mean because you have money in the bank that you can go anytime, open the vault, and make money. You have to go through certain steps to withdraw money. In short, Redux is a way to manage the "state" or we can say it is a cache or storage that can be accessed by all components in a structured way. It has to be accessed through a "Reducer" and "Actions". The reducer is nothing but a pure function that takes currentState and Action and returns a new state. A valid Reducer can return to the current state. Redux Toolkit is our officially recommended approach for writing Redux logic. It wraps around the Redux core and contains packages and functions that we think are essential for building a Redux app. Redux Toolkit builds in our suggested best practices, simplifies most Redux tasks, prevents common mistakes, and makes it easier to write Redux applications.

3.1.2 Backend

WebAssembly

WebAssembly, developed by the W3C, is in the words of its creators a “compilation target.” Developers don’t write WebAssembly directly; they write in the language of their choice, which is then compiled into WebAssembly bytecode. The bytecode is then run on the client—typically in a web browser—where it’s translated into native machine code and executed at high speed. WebAssembly code is meant to be faster to load, parse, and execute than JavaScript. When WebAssembly is used by a web browser, there is still the overhead of downloading the Wasm module and setting it up. For larger Wasm projects, those modules can run to several megabytes, so those delays can be significant. But all else being equal, WebAssembly runs faster. The main goal of WebAssembly is to enable high-performance applications on web pages, "but it does not make any Web-specific assumptions or provide Web-specific features, so it can be employed in other environments as well." It is an open standard and aims to support any language on any operating system, and in practice, all of the most popular languages already have at least some level of support. While WebAssembly was initially designed to enable near-native code execution speed in the web browser, it has been considered valuable outside of such, in more generalized contexts. Since WebAssembly's runtime environments (RE) are low-level virtual stack machines that can be embedded into host applications, some of them have found a way to standalone runtime environments like Wasmtime and Wasmer. The support includes mobile web browsers for iOS and Android. As of October 2022, 96% of installed browsers support WebAssembly (version 1.0).

Rust

Rust is a multi-paradigm, general-purpose programming language. Rust emphasizes performance, type safety, and concurrency. Rust enforces memory safety—that is, that all references point to valid memory—without requiring the use of a garbage collector or reference counting present in other memory-safe languages. To simultaneously enforce memory safety and prevent concurrent data races, Rust's "borrow checker" tracks the object lifetime of all references in a program during compilation. Rust is popular for systems programming but also offers high-level features including some functional programming constructs. Rust is blazingly fast and memory-efficient: with no runtime or garbage collector, it can power performance-critical services, run on embedded devices, and easily integrate with other languages. Rust’s rich type system and ownership model guarantee memory safety and thread safety — enabling you to eliminate many classes of bugs at compile-time. Rust is more significantly influenced by functional programming languages, including Standard ML, OCaml, Haskell, and Erlang. Rust is strongly typed and statically typed: all types of variables must be known during compilation, and assigning a value of a different type to a variable will result in a compilation error. The default integer type is i32, and the default floating point type is f64. If the type of a literal number is not explicitly provided, either it is inferred from the context or the default type is used. Unlike other languages, Rust does not use null pointers to indicate a lack of data, as doing so can lead to accidental dereferencing.

Wasm-Pack

This tool seeks to be a one-stop shop for building and working with rust-generated WebAssembly that you would like to interop with JavaScript, in the browser, or with Node.js. wasm-pack helps you build rust-generated WebAssembly packages that you could publish to the npm registry, or otherwise use alongside any javascript packages in workflows that you already use, such as webpack.

Wasm-Bindgen

wasm-Bingen only generates bindings and glue for the JavaScript imports you actually use and the Rust functionality that you export. For example, importing and using the document.querySelector method doesn't cause Node.prototype.appendChild or window. alert to be included in the bindings as well. The goal of wasm-Bingen is to provide a bridge between the types of JS and Rust. It allows JS to call a Rust API with a string, or a Rust function to catch a JS exception. a Rust library and CLI tool that facilitates high-level interactions between wasm modules and JavaScript. The wasm-Bingen tool and crate are only one part of the Rust and WebAssembly ecosystem. The wasm-Bingen tool is sort of half polyfill for features like the host bindings proposal and half features for empowering high-level interactions between JS and wasm-compiled code (currently mostly from Rust).

3.1.3 IDE

Visual Studio 2022

Visual Studio 2022 is the best Visual Studio ever. Our first 64-bit IDE makes it easier to work with even bigger projects and more complex workloads. Visual Studio 2022 will help you go from idea to code faster than ever. Developer productivity and quality-of-life improvements are at the heart of Visual Studio 2022, and we're excited for you to try it out. Simply put, Visual Studio 2022 will let you bring your ideas to life.

Visual Studio Code

Visual Studio Code is a streamlined code editor with support for development
operations like debugging, task running, and version control. It aims to provide just
the tools a developer needs for a quick code-build-debug cycle and leaves more
complex workflows to fuller featured IDEs, such as Visual Studio IDE.

3.2 FlowCharts

3.2.1 Sequence Diagram

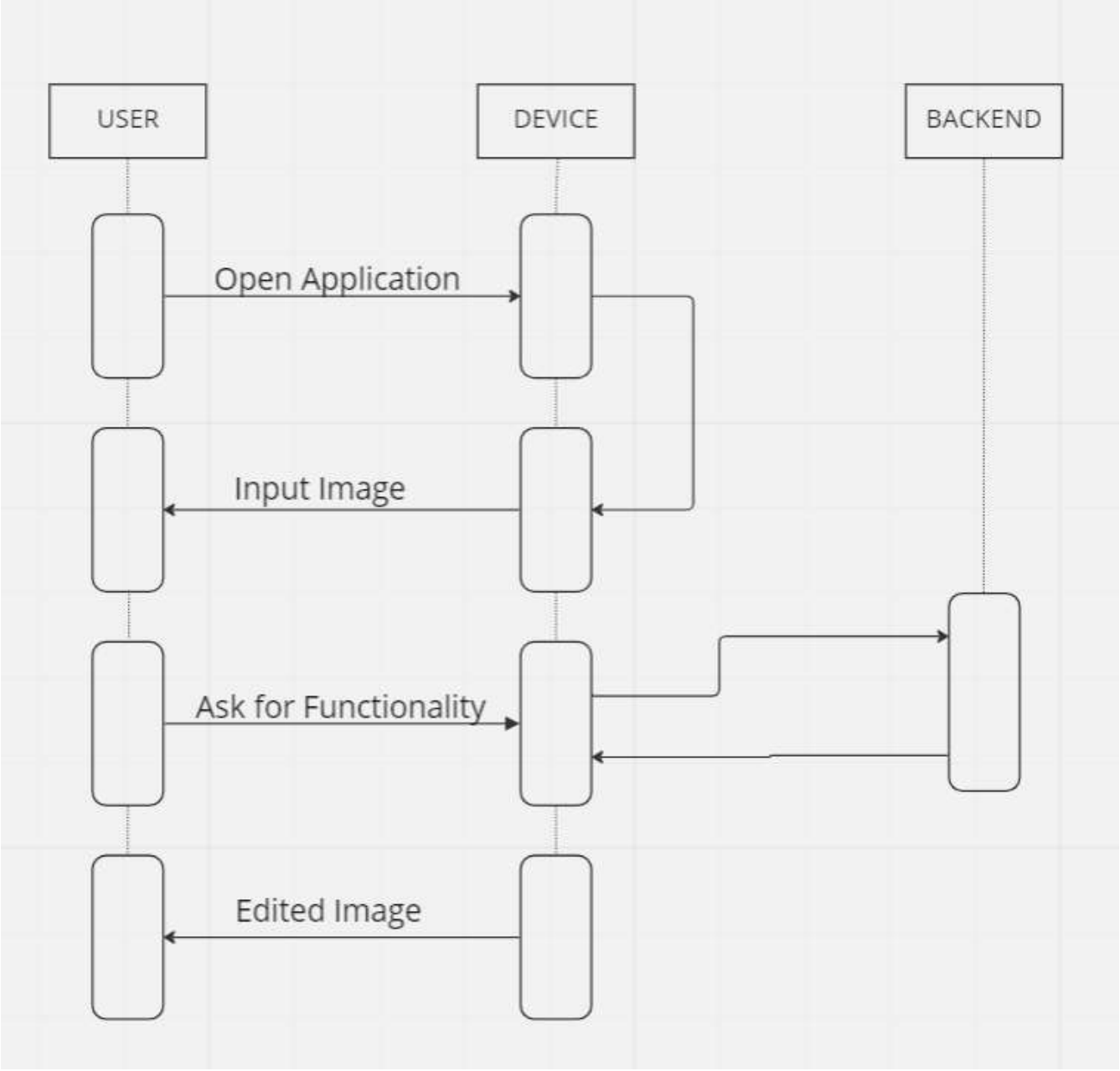


Figure 2. Sequence Diagram

Explanation

The first step includes the User interacting with the user interface provided by the developer i.e. web-app UI. The first thing displayed on the UI includes a default image and option chart from which the user can select which functionality is viable for the image. The User can also change the default image from the *Open* switch in the navbar which allows the user to set a new image. Once the user is finished with the image input, its job is to select any of the available functionality; which later will be applied to the image. When the user selects the image, the frontend section work is now complete and now the functionality is searched in the backend, and in due time will be applied of the pre-requisite image. Now, the user gets the edited image on the device screen, with the viable option of saving and restoring the image. Once the image is saved it will be automatically saved to the user's local storage, and now the user can work with the new image.

3.2.2 UseCase Diagram

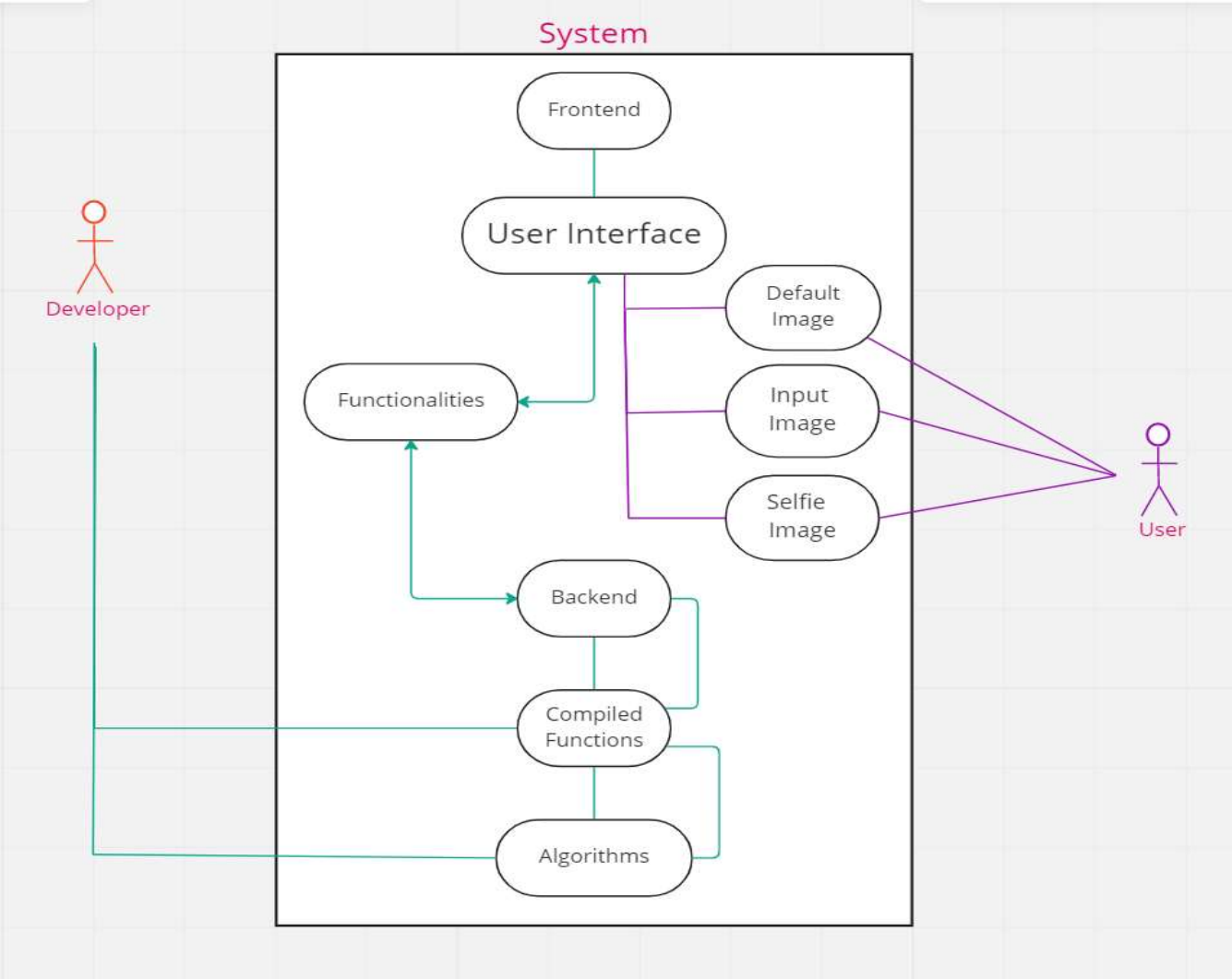


Figure 3. UseCase Diagram

Explanation

Upon opening the system user is represented with an option to select an image; while the user has the option to select any of the functionality but it is limited to the functionalities provided by the developer. Whereas, developers have the option to add any other features in the form of algorithms.

3.2.3 Conceptual Diagram

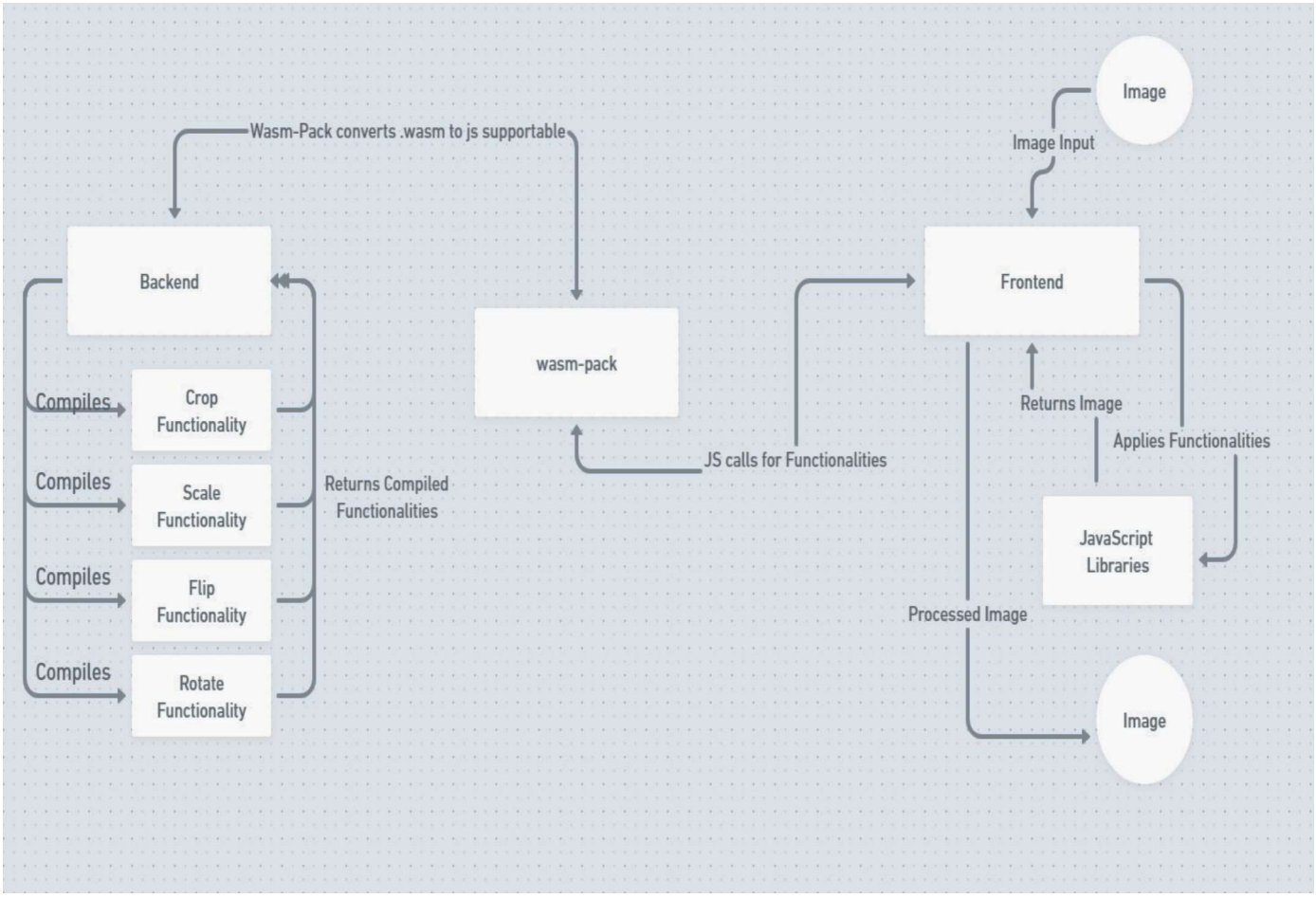


Figure 4. Conceptual Diagram

Explanation

Upon interaction with the user inputs the image in the frontend, and later selects a functionality that calls *wasm-pack* pack compiled functions from the backend; the backend looks for asked function and then returns the functionality asked via the same route. The image displayed will now show applied changes and now the user keeps or restore the image. The saved will be automatically downloaded.

3.3 Design & User Interface

3.3.1 Frontend Design

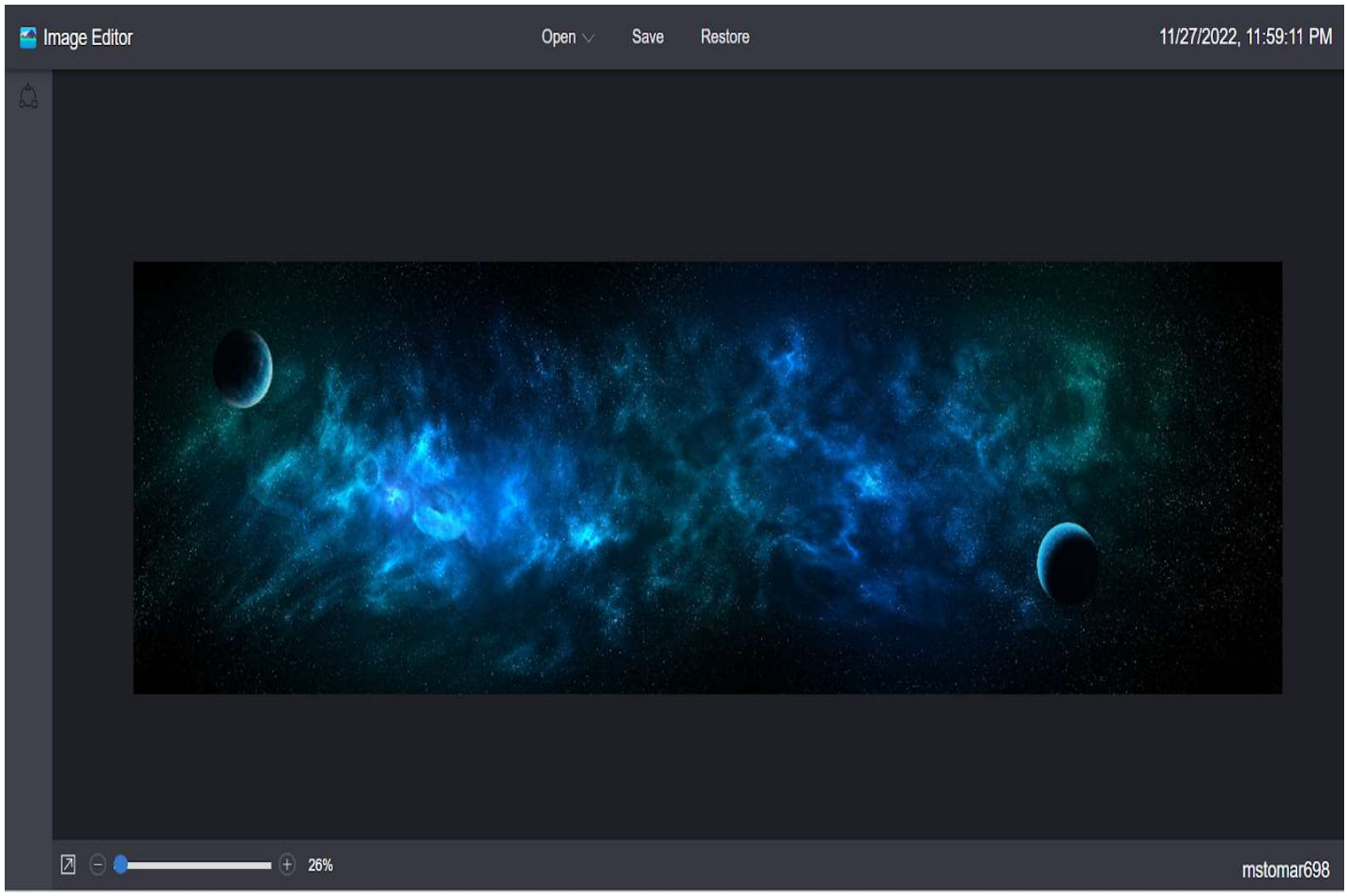


Figure 5. Frontend Design

Components and Functionalities

Image

Its part of the design where the image to be edited or image edited is presented on the screen. The image can be defaulted or taken from the camera.

Open
It is a dropdown where you get to select multiple default images and also have the option to select to go with either camera or select a file from the computer.

Save
It is a button to save the edited image. It will download the image automatically on clicking it. The image will not be restorable on saved.

Restore
It is a button to restore the edited image. It is a button that restores the edited image to its original size. It will only be effective until the image is not saved.

Zoom-Scroll-Bar
It is a scroll bar to magnify the image as per the requirement. The default zoom ratio is set as per the image pixel ratio.

Crop
It is a functionality in which we can take a fixed part of the image out of the original image; later the original image will be discarded and cropped part will be saved.

Scale
It is a functionality in which we can change the original ratio of the image to a new one is saved.

Rotate
It is a functionality in which we can change the default height and width to the new width and height of the image. The ratio remains the same as the previous one but the direction changes.

Camera
It is an option that can be selected from the dropdown in the open section. This functionality allows users to capture images from the webcam directly to edit them later. The image captured can be retaken. On this note, if the webcam is missing user will receive with an error page.

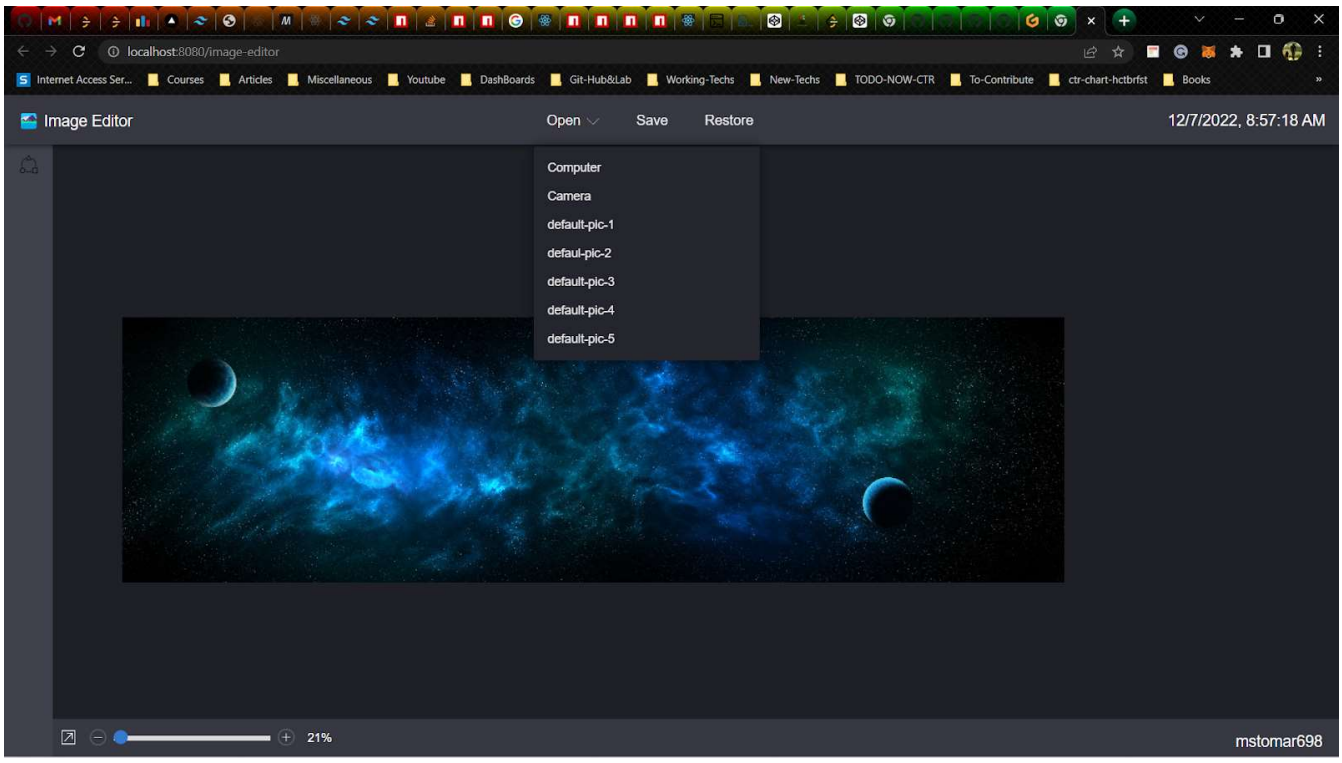
Computer
It is an option that can be selected from the dropdown in open-section. This functionality allows the user to select an image from its local machine storage. Users can select only one image at a time.

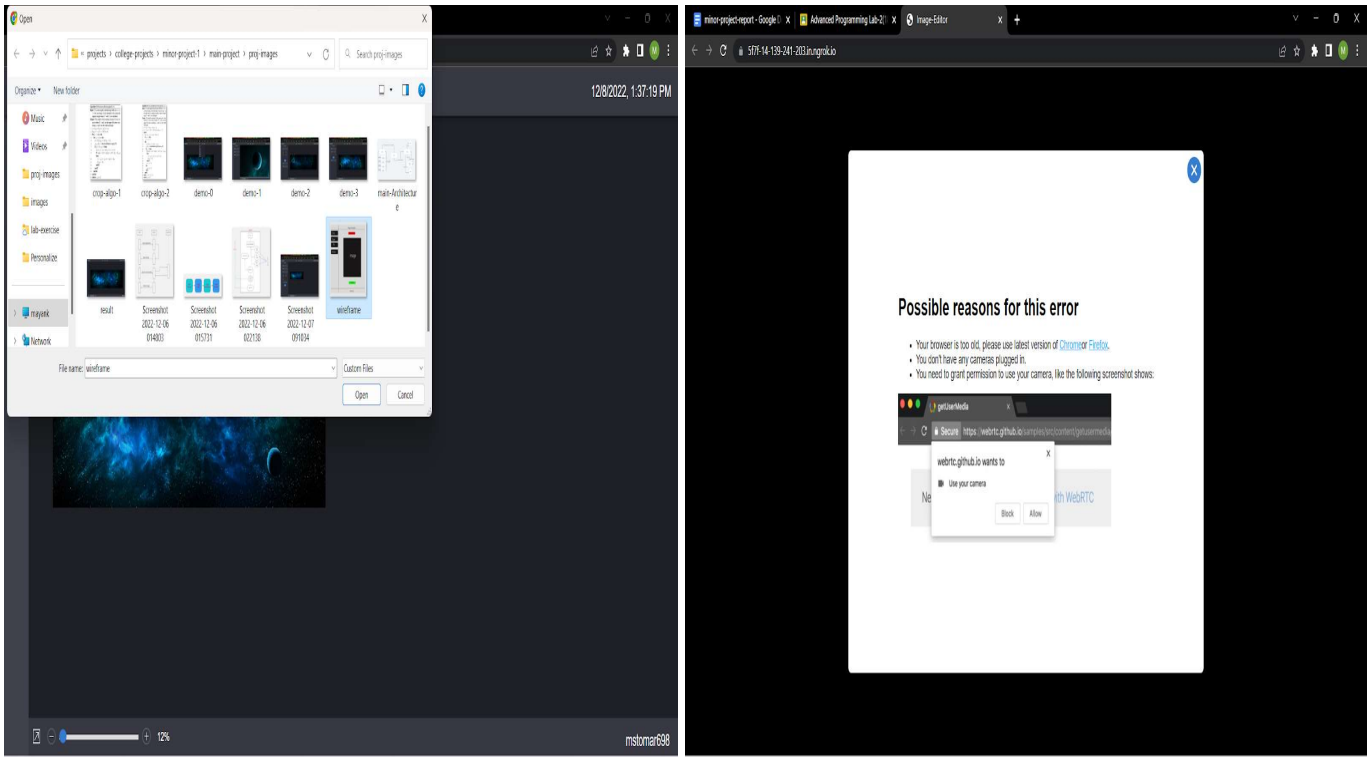
Default-Pic-1
It is an option that can be selected from the dropdown in the open section. This is a list of default images provided by the developer for practicing functionality.

CHAPTER 4
RESULTS/OUTPUTS

With the help of this project, users will be able to reduce there image processing time to a minimum and also provide the user with a lot of other benefits which previously existing systems.
The results are presented in the form of screenshots below to give a better understanding of the project

ScreenShots




Possible reasons for this error
Your browser is too old, please use latest version of Chrome or Firefox.
You don't have any cameras plugged in.
You need to grant permission to use your camera, like the following screenshot shows:
webrtc.github.io wants to
Use your camera
Block
Allow

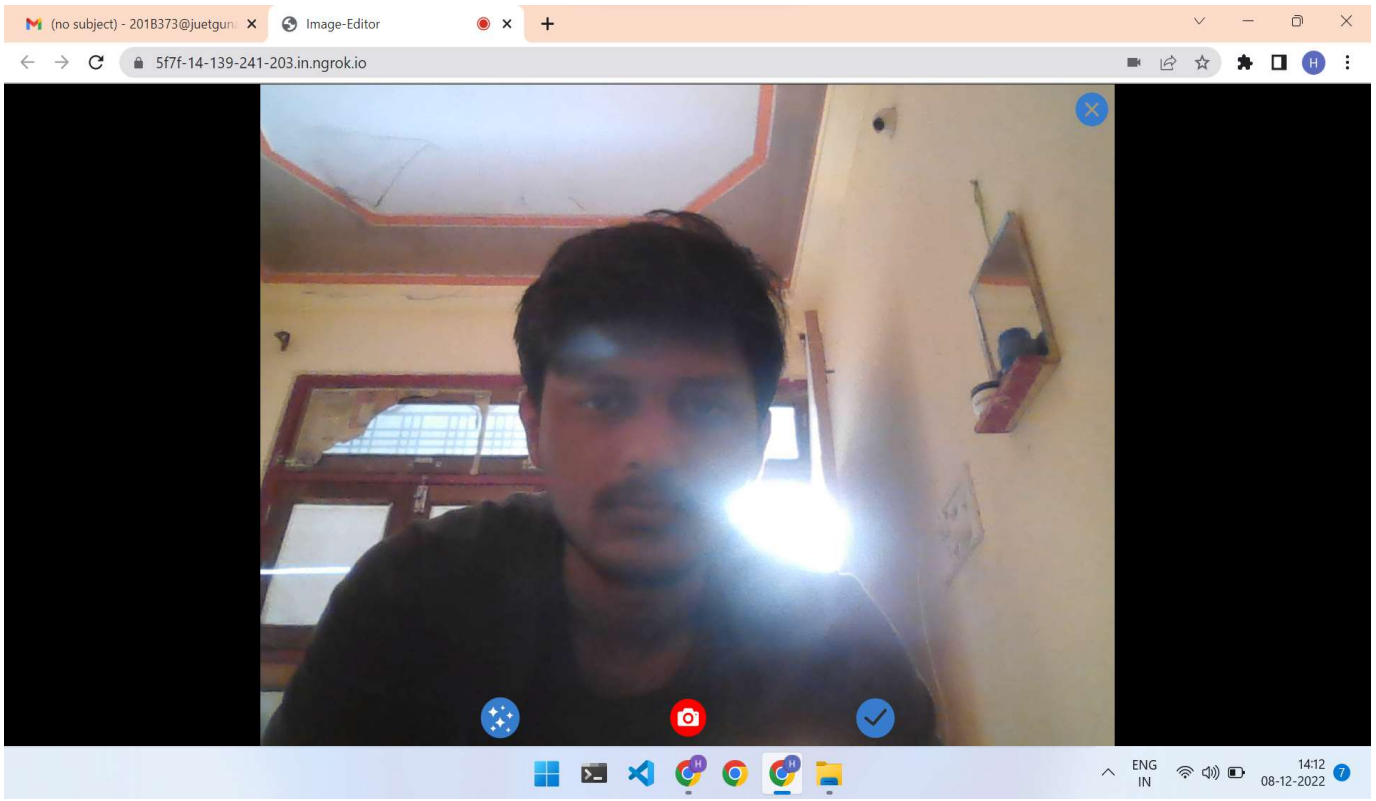
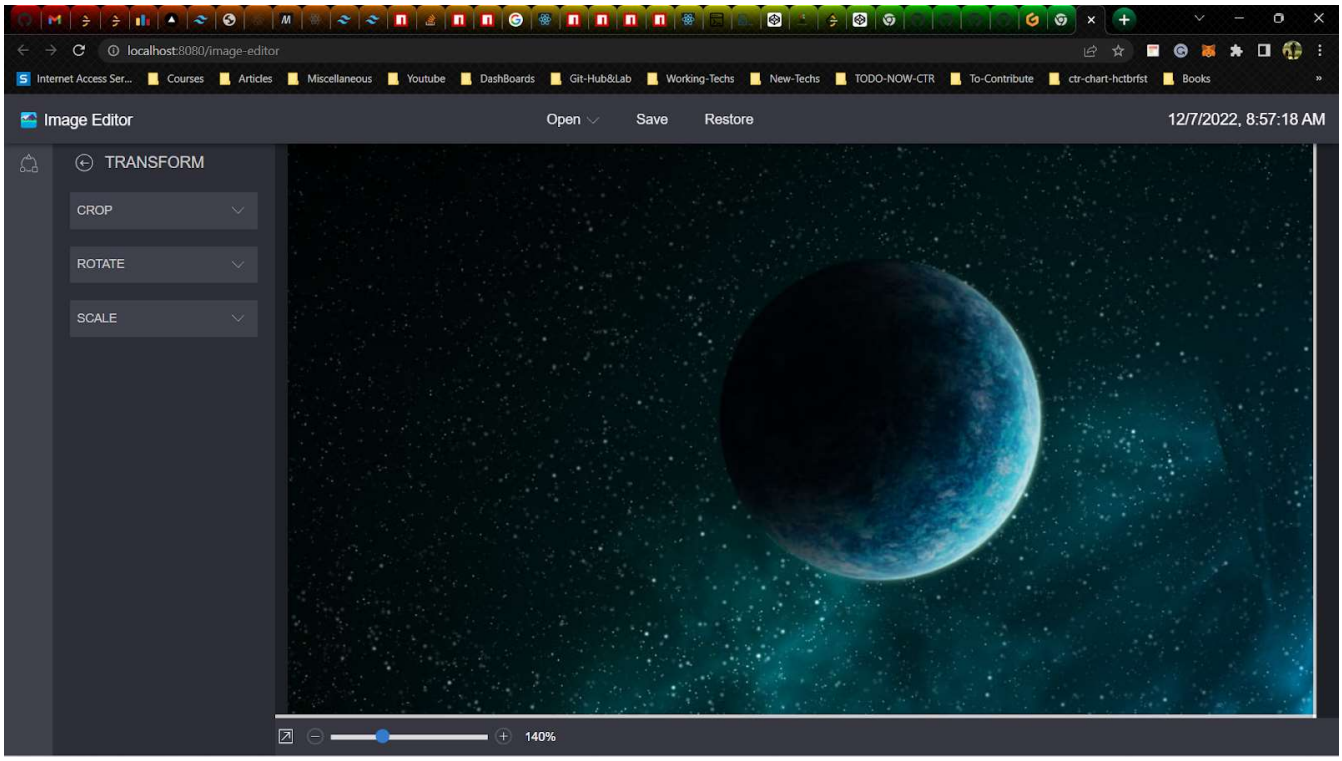
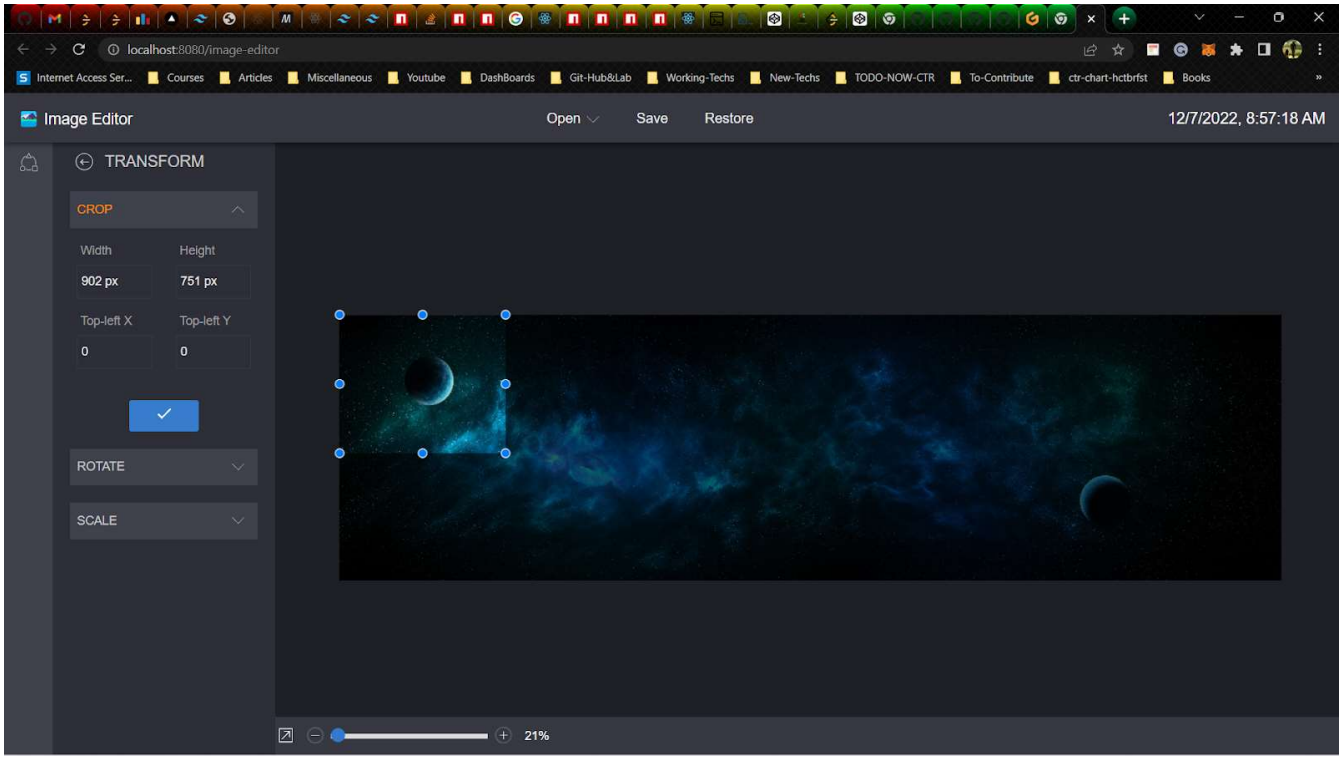


Image-Editor
5f7f-14-139-241-203.in.ngrok.io
ENG
IN
14:12
08-12-2022


localhost:8080/image-editor
Image Editor
Open
Save
Restore
12/7/2022, 8:57:18 AM
TRANSFORM
CROP
ROTATE
SCALE
140%


localhost:8080/image-editor
Image Editor
Open
Save
Restore
12/7/2022, 8:57:18 AM
TRANSFORM
CROP
Width
Height
902 px
751 px
Top-left X
Top-left Y
0
0
ROTATE
SCALE
21%

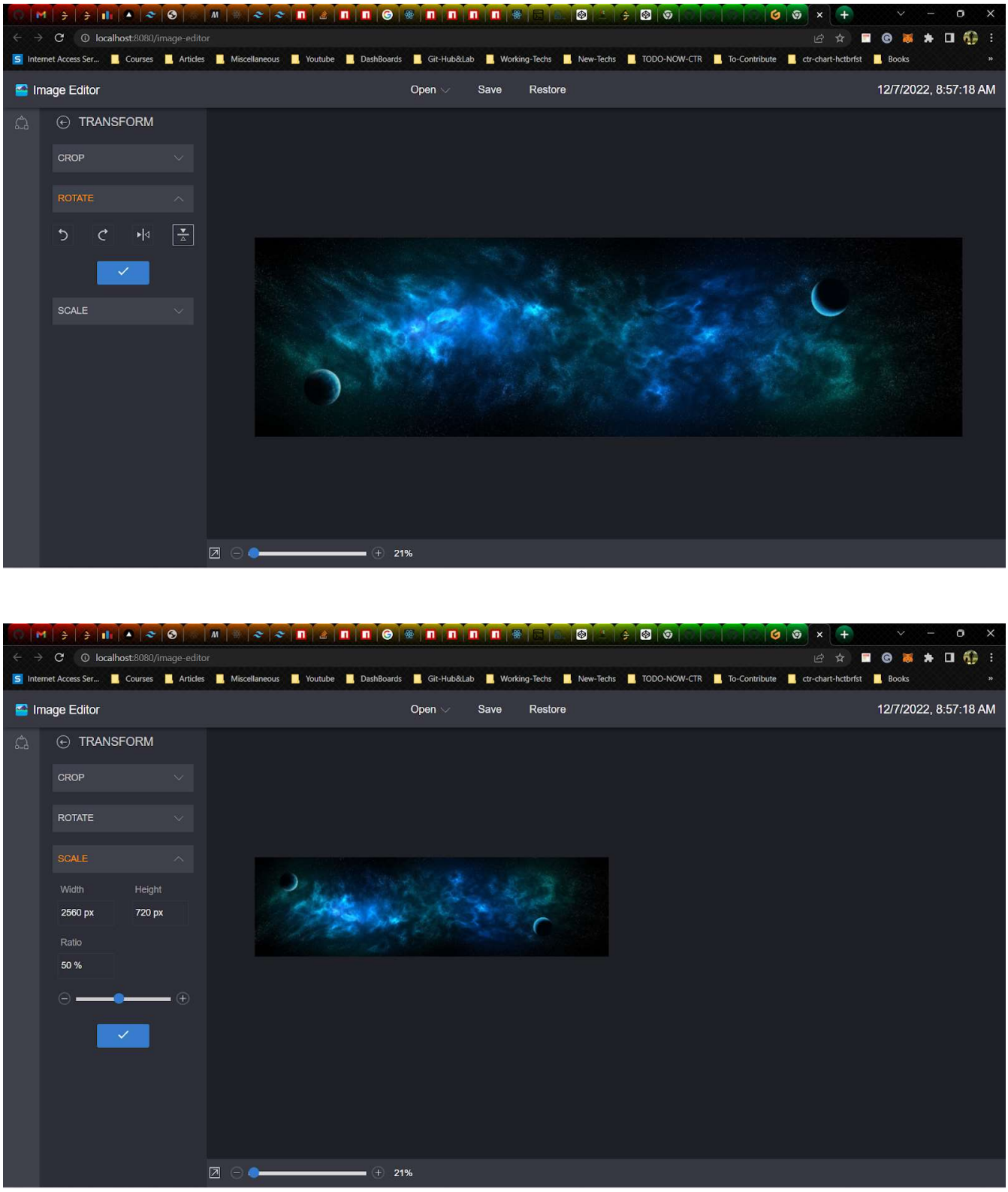


Figure 6. ScreenShots Of Results

CHAPTER 5
CONCLUSIONS/RECOMMENDATIONS

The aim of this project was to develop a web app that can conduct image manipulation on pre-defined or given images while comparing it to pre-existing systems and improve further.

5.1 Conclusion

Finally, we conclude that in the given time period while working on this project we learned many new technologies, and concepts and have also learned about working in a team.
Our project image processing using web assembly is based on webassembly and many other platforms specified earlier.
This insulates the application from technical implementation and enhancement to support future technologies in a transparent manner without having a major impact on the application. This also enables the portability of applications to other operating systems and environments.
This Project followed the following SDL, which involved the steps of
Requirement Analysis
Design
Coding
Testing
Implementation
Maintenance
Deployment

Thus, we were able to understand in greater detail the various software engineering processes and were able to successfully apply them in our project making it more feasible and easy to use.

The app successfully provides the following functionalities to:

For User:

Crop
Scale
Rotate
Camera

For Developer:

Easy Integration in projects
Easy to modify
Easy to update functionalities
Easy to add other and new features

5.2 Scope For Future Enhancement

The application can be further enhanced as an image manipulating web app, that just not only can perform basic level functionalities but also perform more typical applications such as cartoonify, color change, etc. We can also make it more user-friendly by making it totally available locally so that it would not require the internet anymore.

CHAPTER 6
REFERENCES

J. Lach, "BPI: Batch processing images," in *Proc. of the 2008 1st Int. Conf. on Information Technology, IT 2008, 19-21 May 2008, Gdansk, Poland* [Online]. Available: IEEE Xplore, http://www.ieee.org. [Accessed: 10 Sept. 2017].

A. Rakotomamonjy and V. Guigue, "BCI competition III: Dataset II - Ensemble of SVMs for BCIP300 Speller, " IEEE Transactions on Biomedical Engineering, vol. 55, no. 3, pp. 1147-54, March 2008. [Online]. Available: IEEE Xplore, http://www.ieee.org. [Accessed: July 11, 2019].

Web App Inconsideration:
(https://docs.rs/photon-rs/latest/photon_rs/)

Reference for Code:
(https://github.com/silvia-odwyer/photon/blob/master/crate/src/transform.rs)

Newsletter For project Base:
(https://hackernoon.com/essential-guide-to-image-processing-with-webassembly-q11u33hq)

Newsletter For Rust:
(https://this-week-in-rust.org/blog/archives/index.html)

Javascript Help:
(https://www.w3schools.com/js/)

Rust-Wasm Book:
(https://rustwasm.github.io/docs/book/)

Algorithm:
(www.hackerearth.com/practice/notes/image-cropping-and-scaling-algorithm-using-linear-algebra/)

Cropping Algorithms:
(https://learnopencv.com/cropping-an-image-using-opencv/)

Newsletter Reference:
(https://neptune.ai/blog/image-processing-python)

Books on image-processing:
(http://poseidon.csd.auth.gr/LAB_PUBLICATIONS/Books/dip_material/chapter_2/chap2en.pdf)

Image-Processing Algorithms:
(http://kiwi.bridgeport.edu/cpeg585/Algorithms_for_Image_Processing_and_Computer_Vision.pdf)

matrix-notes:
(http://elib.mi.sanu.ac.rs/files/journals/ncd/12/ncd12017.pdf)

Research Paper:
(https://openaccess.thecvf.com/content_cvpr_2016/papers/Chen_Automatic_Image_Cropping_CVPR_2016_paper.pdf)