Paint Application

A PROJECT REPORT
for
Mini Project-I (K24MCA18P)
Session (2024-25)

Submitted by

Verma Alokkumar Amardayal

202410116100239

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. Divya Singhal
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
(DECEMBER- 2024)

CERTIFICATE

Certified that **Verma Alokkumar Amardayal 2426MCA1472** has/ have carried out the project work having "**Paint Application"** (**Mini Project-I, K24MCA18P**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Ms. Divya Singhal
Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Dr. Arun Kr. Tripathi
Dean
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Paint Application

Verma Alokkumar Amardayal

ABSTRACT

The Paint Application is a versatile canvas-based platform designed to allow users to create drawings using simple hand motions. By tracking hand movements and capturing finger motion, the application enables intuitive drawing where the fingertips act as markers. This innovative approach provides a user-friendly interface, making it accessible to people of all ages. The Paint Application leverages **Tkinter** technology, which is widely regarded as a foundational element for creating **GUI**-based applications and is increasingly relevant in Augmented Reality projects.

The core features of the **Paint Application** include basic drawing tools that allow users to paint effortlessly on a blank canvas. Additionally, it supports functionalities like erasing, resizing, drawing geometric shapes, clearing the canvas, and selecting colors from a customizable palette. Users can also import images, apply edits, and save their work in various formats for further use.

Developed entirely in **Python**, the application integrates both basic and advanced programming techniques, making it a powerful tool for drawing and image manipulation. It utilizes color tracking and detection processes to enhance usability, where color markers produce a mask on the original canvas. This feature-rich application serves as an ideal project for learning Python's capabilities and exploring creative solutions in graphical programming.

Keywords: Tkinter, Python, Paint Application, GUI

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Ms. Divya Singhal** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Verma Alokkumar Amardayal

TABLE OF CONTENTS

Page Number

Chapter 1

Introduction

Drawing or Sketching using hand is everyone's wish. Some or the other time we imagine writing in air using our hand. So, here came the project from this concept where we create a canvas and pick the colors required using our hand and draw the required design or write anything you wish.

Paint is an **endearingly straightforward** program that offers very little in the way of basic and advanced features. It is **really easy to use**, even for newbies. The uncomplicated user interface features tools down the left-hand side. Our Paint Application is **completely free.** Just download and run the .exe file and Paint Application will open automatically.

The user draw straight horizontal, vertical, or diagonal lines with the straight line tool. The user may also **draw Ellipse, Oval, Square and Rectangle Shapes** in Paint Application. Drawing Application in Python Tkinter GUI, where we can simply draw something on the canvas using a **pencil** and erase it with an **eraser**, as well as the ability to change the **thickness of a pencil and eraser**. We may also modify the **canvas's background color, We Import Image and edit them** and **save** as any specific file or Image format on our local computer.

1.1 Key Features of Paint Application

The Paint Application is built to meet specific usability and performance criteria, including:

- **Ease of Use:** Simple and intuitive interface for all users.
- **Customizability:** Allows selection of colors, shapes, and brush sizes.
- **Efficiency:** Lightweight software requiring minimal system resources.
- **Versatility:** Capable of drawing, take shapes, choose color, change background, editing images, applying filters, and saving files.

1.2 Underlying Technologies

1.2.1 Front-End

❖ Tkinter

Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library. Tkinter has several strengths. It's **cross-platform**, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run.

- Although Tkinter is considered the de facto Python GUI framework, it's not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you're looking for.
- However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern sheen is unnecessary, and the top priority is to quickly build something that's functional and cross-platform.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

➢ The front-end is developed using Tkinter, a standard GUI framework in Python. Tkinter provides:

- Native cross-platform support
- Lightweight GUI rendering
- Basic widgets such as buttons, labels, and canvases

1.2.2 Back-End

❖ **Python**

Python has been among the most in-demand programming languages, and its popularity is expected to continue skyrocketing in 2022. That is especially true when it comes to choosing technology for back-end development.

Python was among the top five most widely used programming languages around the world, only yielding the palm to JavaScript, HTML/CSS, and SQL. Popularity is a reasonably good reference point when it comes to choosing the best technology for app development and, in particular, for designing the app's back-end. And it rarely comes out of thin air – there is the positive experience of thousands or even millions of people who remained satisfied upon using the language.

Python is recognized as one of the best programming languages for startups as it allows developing applications that are clear and simple. At the same time, it is flexible and easy to scale, which means it always leaves room for maneuver, such as quickly getting from a prototype, an MVP, or just a small project to a full-fledged, complex app.

Python is a robust back-end programming language used and trusted by many renowned companies, ensuring efficiency, security, scalability, and most of the qualitative parameters applicable to modern apps.

➢ The **back-end relies on Python's PIL (Pillow) library** for image manipulation, including:

- Applying filters (e.g., blur, sharpen, emboss)
- Loading and resizing images
- Saving canvas drawings as image files

1.3 Goals

The Paint Application aims to strike a balance between functionality and simplicity. The primary objectives include:

1. **Accessibility**: Provide a tool that can be easily used by people of all technical skill levels, including beginners.
2. **Functionality**: Offer essential drawing features alongside advanced options like image editing and shape tools to meet a variety of user needs.

3. **Efficiency**: Design a lightweight application that operates smoothly on common hardware without excessive resource consumption.
4. **Customizability**: Enable users to personalize their drawing experience, including choosing colors, shapes, and canvas settings.

In summary, the goal is to create a versatile drawing tool that combines ease of use with powerful features, making it suitable for casual users and developers exploring GUI-based applications.

1.4 Target Audience

This application caters to:

- Hobbyists looking for basic digital drawing tools.
- Students and educators for diagram creation.
- Developers exploring Tkinter GUI capabilities.

Chapter 2

Feasibility Study

Digital drawing and painting applications have been widely used for both creative expression and educational purposes. Early software such as MS Paint laid the groundwork for user-friendly digital art platforms, offering basic tools for drawing, coloring, and editing. These tools inspired the development of modern, more versatile applications, including those built using Python and its graphical libraries.

Python's Tkinter library, known for its simplicity and platform independence, has become a popular choice for creating GUI-based drawing tools. Tkinter allows developers to create interactive applications with features such as customizable shapes, dynamic resizing, and intuitive interfaces. Additionally, Python's PIL (Pillow) library enhances the functionality of such tools by enabling advanced image processing capabilities, such as color manipulation, filter application, and image resizing.

The Paint Application draws upon these technologies to offer a lightweight yet feature-rich platform. Unlike traditional drawing tools, this project introduces innovative features for drawing, making the experience more interactive and immersive. The integration of both basic and advanced Python programming demonstrates the language's versatility in creating practical applications.

While many existing solutions cater to professional use, this application is designed for beginners and hobbyists, focusing on simplicity, accessibility, and functionality. It bridges the gap between creativity and technology, providing a robust platform for digital expression.

2.1 Feasibility Study in many factors

The feasibility study assesses whether the Paint Application is practical and achievable within the constraints of technology, resources, and user expectations. It evaluates operational, technical, and economic feasibility.

2.1.1 Operational Feasibility

The Paint Application meets user requirements for an accessible, feature-rich, and lightweight drawing tool. It integrates user-friendly features like shape tools, color customization, and image editing, ensuring ease of use. The operational feasibility ensures that the application can be implemented without requiring significant changes to the existing hardware or software infrastructure.

2.1.2 Technical Feasibility

Technical feasibility examines whether the technology stack supports the project's requirements. The Paint Application leverages Python, Tkinter, and PIL libraries, all of which are widely supported and well-documented. These technologies enable smooth implementation of GUI features, image manipulation, and platform compatibility. This ensures that the project can be developed using available resources and expertise.

2.1.3 Economic Feasibility

Economic feasibility ensures that the project is cost-effective. The Paint Application uses Python, an open-source language, and free libraries like Tkinter and Pillow, significantly reducing development costs. Additionally, the lightweight nature of the application ensures low operational costs, making it viable for both individual developers and small teams.

2.2 Literature Review

The development of the Paint Application is informed by existing literature on GUI design, image processing, and user experience. Key findings include:

2.2.1 Importance of Usability in GUI Design

Studies highlight that a simple and intuitive interface significantly enhances user satisfaction. By leveraging Tkinter, the application provides a clean, organized layout that aligns with these principles.

2.2.2 Advancements in Image Processing with Python

The PIL (Pillow) library provides robust tools for image manipulation, including resizing, filtering, and color adjustments. Research on Python's role in image processing emphasizes its effectiveness for lightweight applications like Paint.

2.2.3 Benefits of Open-Source Development

Open-source tools like Python ensure cost-efficiency and community support. Literature underscores the value of open-source projects for rapid prototyping and scalability, both of which are relevant to the Paint Application.

2.3 Feasibility Findings

The Paint Application is deemed highly feasible based on:

1. **Operational Feasibility**: Aligns with user needs and system capabilities.
2. **Technical Feasibility**: Leverages reliable and widely-supported technologies.
3. **Economic Feasibility**: Minimizes development and operational costs through open-source tools.

Chapter 3

Project Objective

The primary objective of this project is to develop a user-friendly and versatile Paint Application that combines basic and advanced drawing tools with modern features like image editing and fingertip-based interaction. The application aims to provide an intuitive platform for users of all skill levels, from beginners to hobbyists, to express their creativity digitally.

Key Objectives:

1. **Interactive Drawing Experience**:
    - Enhance user engagement by incorporating intuitive freehand drawing tools like pencils, shapes, and lines.
    - Introduce fingertip or gesture tracking technology for a modern, innovative approach to drawing.
2. **Functional GUI-Based Drawing Application**:
    - Deliver a robust platform for creating and editing digital artworks.
    - Simplify controls for drawing geometric shapes, freehand sketches, and managing color selections.
3. **Scalability**: Design the application to accommodate future enhancements and additional features with minimal refactoring.
4. **Reliability**: Provide a stable and bug-free application through rigorous testing and debugging.
5. **Security**: Protect user data and maintain the integrity of files saved and shared through the application.
6. **Comprehensive Toolset**:
    - Integrate tools such as erasers, shape outlines, resizing, and brush thickness adjustments.
    - Provide dynamic customization options for both novice and advanced users.

7. **Image Import and Editing**:
    - Allow seamless image imports for editing and enhancement directly on the canvas.
    - Enable advanced filters, cropping, color correction, and blending options.
8. **Platform Independence**:
    - Ensure compatibility and stability across major operating systems, including Windows, macOS, and Linux.
    - Maintain lightweight performance for devices with limited resources.
9. **Saving and Sharing Capabilities**:
    - Offer options to save creations in various formats like PNG, JPEG, and BMP.
    - Implement sharing options for exporting to cloud storage or social platforms.
10. **Cost-Effectiveness**: Utilize open-source tools and libraries to minimize development costs while delivering high-quality functionality.
11. **Educational Integration**:
    - Incorporate features suitable for educational use, such as adding labels, annotations, and exporting diagrams for teaching purposes.
12. **Collaborative Drawing Support**:
    - Explore real-time collaborative tools, allowing multiple users to work on the same canvas remotely.
13. **Enhanced Performance Optimization**:

- Reduce latency in drawing actions for real-time responsiveness.
- Implement low-power modes for devices with constrained battery life.

Chapter 4

Hardware and Software Requirements

Hardware is the physical components that a computer system requires to function. It encompasses everything with a circuit board that operates within a PC or laptop; including the motherboard, graphics card, CPU (Central Processing Unit), ventilation fans, webcam, power supply, and so on.

Software refers to the set of programs, data, and instructions that enable a computer to perform specific tasks and functions. It includes operating systems, application software, and utility programs that work together to manage hardware resources and provide an interface for user interaction.

To run the application software of the system in the computer, the minimum **hardware configuration required** is as below:

- 1.7 GHz Pentium processor or other compatible
- Intel chipset motherboard
- 4GB RAM or More
- Color Monitor or LCD
- Keyboard
- Mouse

Software Requirements:

- **Operating Systems:** Windows, Linux, macOS
- **Development Tools:** Python 3.9+, Visual Studio Code IDE
- **Libraries:**
    - Tkinter: For GUI development
    - PIL (Pillow): For image handling and editing

Chapter 5

Project Flow

5.1. Requirement Analysis:

Objective: Define the user needs and gather all the essential functionalities needed for the application.

- **Identify Core Features:**
  - **Drawing Tools:** Users should be able to draw on a canvas using various tools such as a pencil, brush, and eraser.
  - **Shapes:** Include features for drawing basic shapes circle, rectangle, line.
  - **Saving and Loading Files:** Allow users to save their work in common formats (e.g., JPEG, PNG, BMP) and load previously saved files.
  - **Image Manipulation:** Include basic image manipulation tools.
  - **Color Selection:** Offer a color palette or a color picker tool for users to customize their drawing experience.
- **Gather User Feedback:**
  - Survey or interview potential users to understand their specific needs, preferences, and pain points regarding drawing applications.
  - Collect input on features they deem essential, such as specific drawing tools or file formats.
- **Prioritize Features:**
  - Based on user input, prioritize the features that are most important to the user base, ensuring the minimum viable product (MVP) addresses key requirements.

5.2. Design Phase:

Objective: Plan the application's user interface (UI) and ensure it provides an intuitive and engaging experience.

- **User Interface Layout:**
  - **Canvas Area:** The primary drawing area should be the focus, with sufficient space for users to draw and manipulate their images.
  - **Toolbars:** Place a toolbar for drawing tools, color selections, editing functions and shapes, which should be easy to access and change during use.
- **Design User Workflows:**
  - **Draw a Shape:** The user selects a tool, chooses a color, and draws on the canvas.
  - **Edit Image:** Users can use selection tools to modify the image.
  - **Save File:** The user can save the drawing in various formats, select file paths, and set preferences for file quality or compression.
  - **Navigation:** Users should be able to use mouse gestures.
- **Wireframes and Prototypes:**
  - Create wireframes (basic layout sketches) to visualize the structure of the application.
  - Develop interactive prototypes to test the user flow and gather feedback before implementation.

5.3. Development Phase:

Objective: Implement the application's core features, integrating the functionalities defined in the design phase.

- **Core Functionalities:**
  - **Drawing Tools Implementation:** Code the functionality for basic tools such as freehand drawing, shape creation (rectangles, circles, lines), and an eraser tool.
  - **Color and Brush Selection:** Develop a color picker and brush size selector. Allow users to choose colors from a palette or custom input (e.g., HEX/RGB).

- **File Operations:** Implement file handling for opening, saving, and exporting images. Ensure users can load images and save their creations in multiple formats.
- **Image Manipulation Tools:** Develop image editing features Implement filters for basic image adjustments like brightness, contrast, or grayscale.

- **Platform Compatibility:**
  - Ensure the application is compatible with multiple platforms (e.g., Windows, macOS, Linux) by using cross-platform frameworks such as Tkinter, wxPython, or Qt for GUI development.

5.4. Testing Phase:

Objective: Ensure the application functions as expected and meets user needs.

- **Functional Testing:**
  - Test all drawing tools, shapes, and image manipulation features to ensure they work correctly.
  - Test saving and loading files to verify that files are stored properly and can be opened without corruption.
  - Ensure the undo and redo functionalities are working as expected, handling multiple user actions in sequence.
- **Usability Testing:**
  - Conduct user testing with potential users to gauge the application's ease of use. Ensure that the interface is intuitive and that users can easily navigate and perform tasks.
  - Observe users performing tasks like drawing, saving files, and applying filters to detect any usability issues.
- **Bug Fixes and Performance Testing:**
  - Address any bugs or glitches identified during testing.
  - Conduct performance testing, particularly if the application deals with large images or complex operations, to ensure the app runs smoothly without lag or crashes.

- **Cross-Platform Testing:**
    - Test the application on different operating systems to ensure consistent functionality and appearance.

5.5. Deployment:

Objective: Package and distribute the application for users to download and use.

- **Packaging:**
    - Compile the application into a standalone executable file (e.g., .exe for Windows, .app for macOS). Use packaging tools like PyInstaller or cx_Freeze for Python applications.
    - Ensure the packaged version includes all dependencies and assets necessary for the application to run smoothly.
- **Cross-Platform Compatibility:**
    - For macOS and Linux, ensure the application runs smoothly by testing it on different system versions and resolving any platform-specific issues.
    - If possible, offer the application as a web-based version (e.g., using HTML5 Canvas for basic drawing), allowing users to access it from a browser.
- **Distribution:**
    - Make the application available for download on a website or through platforms like GitHub or other software distribution platforms.
    - Consider submitting the application to app stores (e.g., Microsoft Store, Mac App Store) for wider reach.
- **Post-Deployment Support:**
    - Provide documentation, FAQs, and a support system to assist users with any post-deployment issues.
    - Monitor feedback for bug reports or feature requests and plan for future updates.

5.6. Use Case Diagram

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system. The use cases are represented by either circles or ellipses. It sepecify the expected behaviour and not the exact method of making it happen.

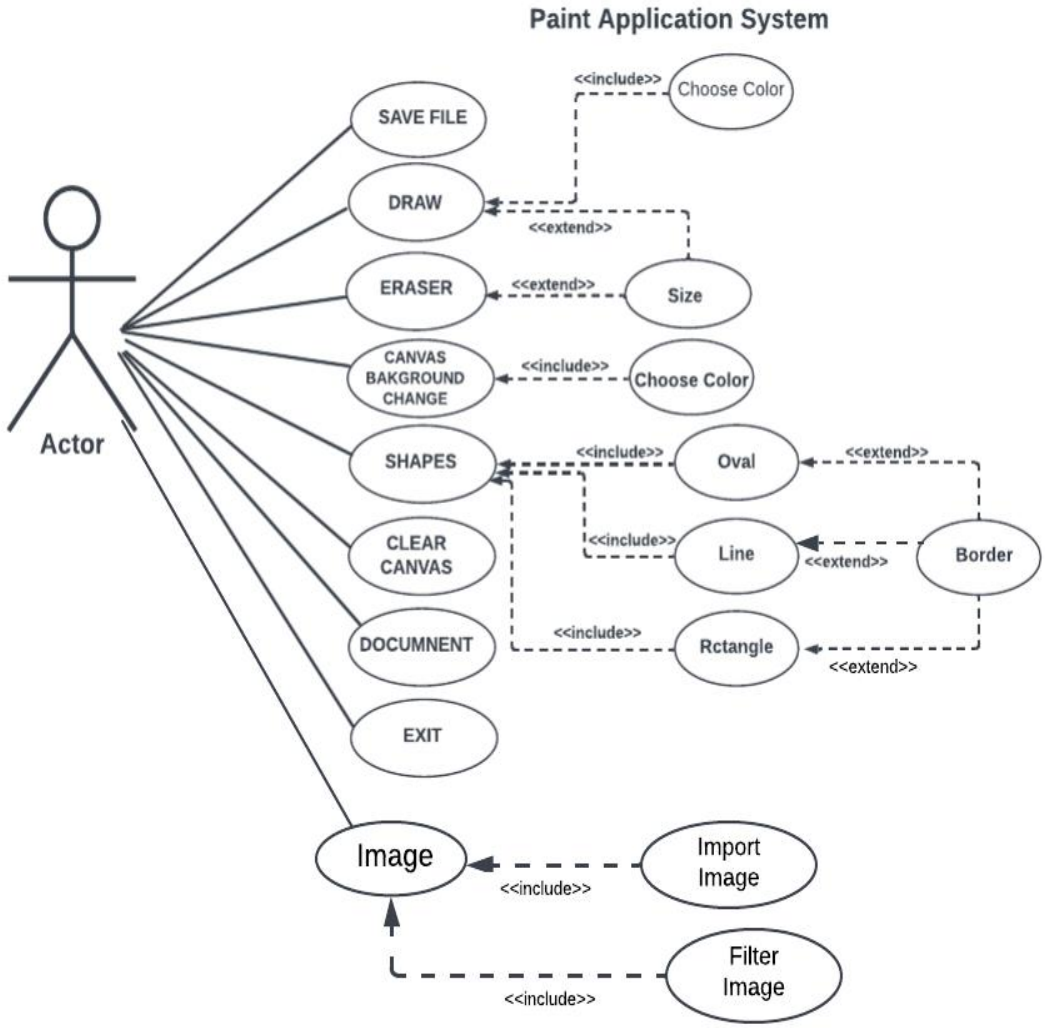


Fig.5.1 Use Case diagram of Paint Application

The main purpose of a use case diagram is to portray the dynamic aspect of a system. It accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons, use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system.

Following are the purposes of a use case diagram given below:

1. It gathers the system's needs.
2. It depicts the external view of the system.
3. It recognizes the internal as well as external factors that influence the system.
4. It represents the interaction between the actors.

❖ Product Functions

- **Shapes:** It provides tools for line, circle, and rectangle.
- **Color Picker :** Users can choose the color for the pen and eraser.
- **Drawing:** Those shapes and lines selected can be drawn on a canvas.
- **Eraser:** The user can erase parts of the drawing.
- **Document Button:** A button to open, Documentation or user manual for reference.
- **Scale Adjustment:** A scale feature that lets users adjust brush size or tool dimensions.
- **Change Canvas Background:** Allows users to set a custom background color for the canvas.
- **Clear:** Clear or removing all drawn shapes and resetting it to the default background color.
- **Image:** A container where the button and combobox are placed.It organizes the GUI elements related to image loading and filtering.
- **Save Image:** It allows the saving of the drawing as an image.

Chapter 6

Project Outcome

The Paint Application achieves several practical outcomes, making it a versatile and user-friendly tool for digital drawing and editing. The application offers a dynamic canvas where users can draw freehand, create geometric shapes, and adjust the thickness of tools such as pencils and erasers. It provides an intuitive interface, ensuring accessibility for beginners while maintaining functionality for advanced users.

The application enables users to import images, apply filters, and perform basic edits, enhancing its utility beyond a simple drawing tool. Customizable features like shape outlines, color palettes, and background adjustments make it adaptable to various creative needs.

Users can save their creations in common file formats, ensuring ease of access and compatibility with other platforms. The lightweight nature of the software ensures smooth operation on devices with minimal hardware requirements.

Overall, the Paint Application combines simplicity and efficiency, offering a robust platform for creative expression and practical image manipulation.

- **Drawing and Editing Tools:** Users can draw freehand, use predefined shapes, and erase as needed.
- **Image Manipulation:** Import, edit, and save images in various formats.
- **Customization Options:** Choose colors, resize shapes, and apply filters to enhance images.
- **Ease of Use:** A simple interface that supports both novice and experienced users.

6.1 Drawing Tools

Objective: Provide essential tools for creating and editing freehand drawings and shapes.

- **Pencil Tool:**
    - **Functionality:** The pencil tool allows users to draw freehand on the canvas. The tool should replicate a real pencil, offering smooth, continuous lines, and should be versatile enough for detailed drawing.
    - **Customization:**
        - **Pen Size:** Allow users to adjust the thickness of the pencil strokes, ranging from thin to thick lines. This can be done via a slider or a dropdown selection in the toolbar.
    - **Color Selection:** Integrate the color palette for users to pick a color for their pencil tool. Allow them to select custom colors using an RGB or HEX input field.
- **Eraser Tool:**
    - **Functionality:** The eraser removes parts of the drawing on the canvas, whether it's freehand strokes or shapes. This tool should work seamlessly to erase any unwanted strokes or elements.
    - **Customization:**
        - **Size Adjustment:** Users should be able to adjust the size of the eraser tool, just like the pencil tool. A slider or input box can let them fine-tune the eraser's diameter.

6.2 Shapes and Colors

Objective: Provide predefined shapes for structured designs and a flexible color selection system.

- **Predefined Shapes:**
    - **Rectangle:** Users can draw rectangles by clicking and dragging on the canvas. This tool can be used for geometric designs or as the basis for other shapes.
    - **Circle:** The circle tool lets users draw perfect circles (or ellipses) on the canvas by defining a start and end point or by dragging from a fixed point.

    - **Line:** Users can draw straight lines by selecting two points on the canvas. This tool is ideal for geometric patterns, guides, or diagrams.
- **Color Selection:**
    - **Predefined Palette:** Offer a set of common colors (e.g., basic primary colors, black, white, gray). These can be displayed as a grid or palette for quick access.
    - **Custom Color Options:** Provide a color picker tool that allows users to choose colors using RGB values, HSL, or HEX codes. Additionally, allow users to save custom colors to a "favorites" section for easy future access.

6.3 Image Features

Objective: Allow users to enhance their creations by importing external images, applying filters, and exporting their work in various formats.

- **Import:**
    - **Functionality:** Users can load images from their local device onto the canvas. The import tool should allow them to choose from common formats (e.g., PNG, JPEG, BMP) and display a preview before insertion.
- **Edit:**
    - **Filters:** Offer a suite of image manipulation filters that users can apply to imported images, such as:
        - **Blur:** Softens the image to create a smooth, diffused effect. Useful for backgrounds or soft-focus effects.
        - **Sharpen:** Enhances image details and edges, increasing contrast between adjacent pixels.
        - **Emboss:** Applies a 3D effect to the image, giving it a raised or indented appearance.
        - **Grayscale:** Converts the image to black and white, which can be useful for certain design effects.

- **Save:**
    - **Export Formats:** Allow users to export their drawings and modified images in various formats such as:
        - **Raster formats:** PNG, JPEG, BMP (for pixel-based images).
        - **PDF:** Allow users to save their work as a PDF file, especially useful for multi-page projects or if the user needs to share the image for printing.
    - **Save Options:** Enable users to define quality and resolution settings when exporting the file, particularly for images intended for high-resolution printing.

6.4 Advanced Options

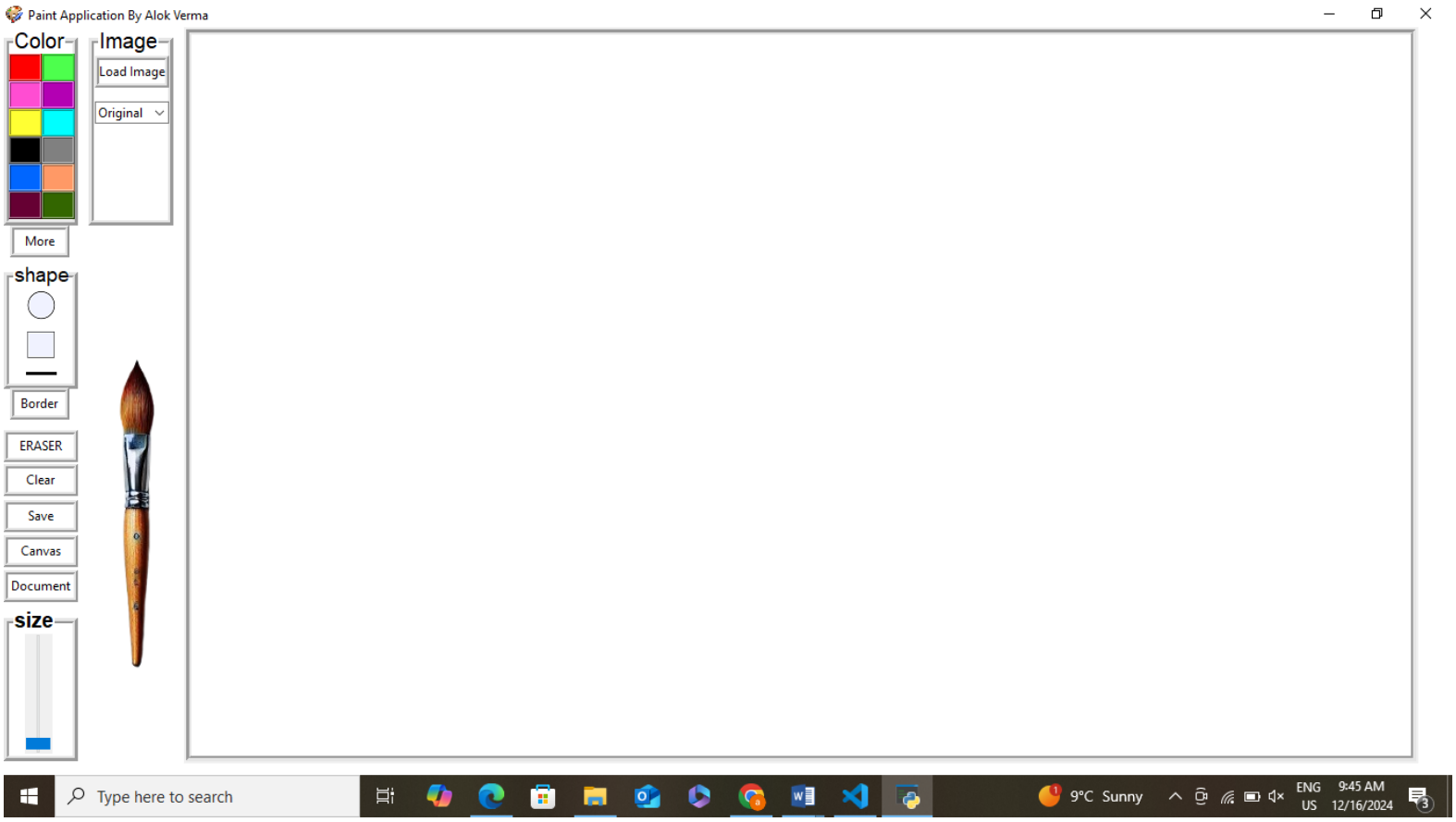
Objective: Offer enhanced customization and convenience to make the drawing process smoother and more flexible.

- **Adjustable Pen and Eraser Sizes:**
    - Provide a range of sizes for both the pen (pencil tool) and eraser, allowing users to choose the right size for their needs. This can be implemented as a slider or as preset sizes (small, medium, large).
    - **Effect on Precision:** Larger pen sizes would be ideal for bold strokes or filling large areas, while smaller sizes help with fine detail work. Similarly, large erasers are great for quick cleanup, and small ones are perfect for precise removals.
- **Canvas Clearing:**
    - **Clear the Canvas:** Include a "clear canvas" button that allows users to erase all content from the drawing area quickly. This feature should prompt for confirmation to avoid accidental deletion of work.
    - **Reset to Default:** Optionally, provide a reset feature that clears the
- **Background Color Customization:**
    - Allow users to change the background color of the canvas to suit their preferences or to create a specific mood for the drawing. This feature can support plain colors.

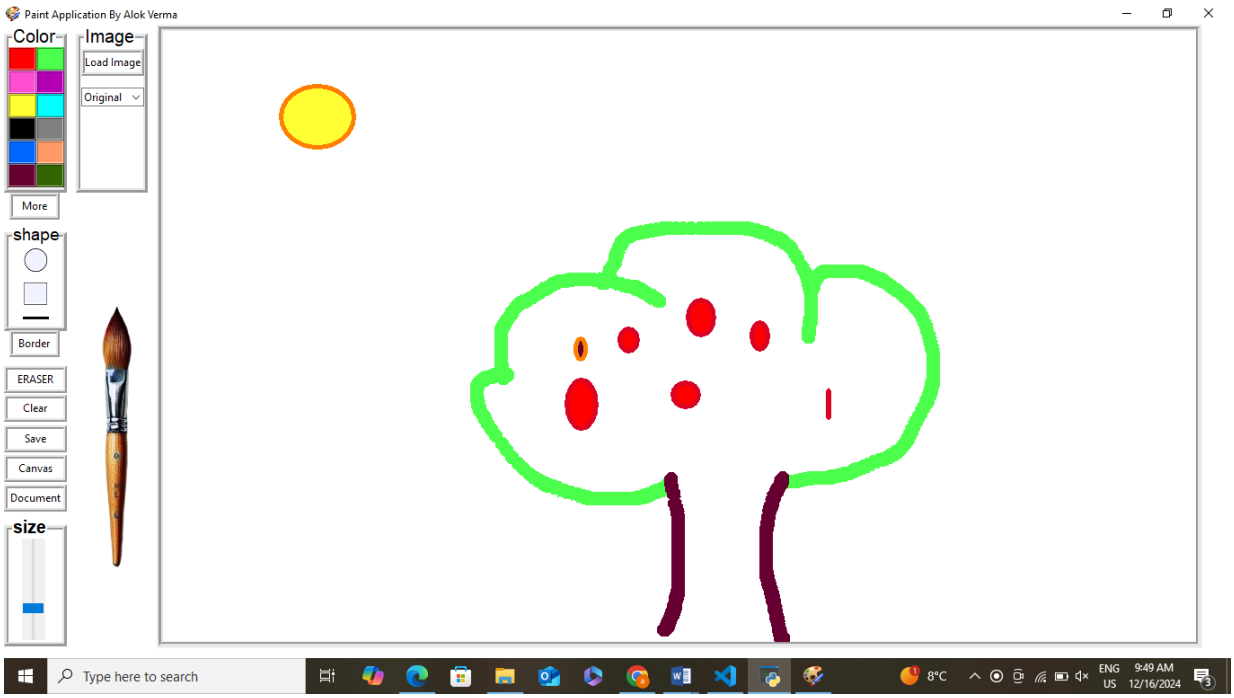
By expanding on these tools and options, the application will provide a comprehensive and flexible drawing experience, suitable for both casual users and more advanced designers.

6.5 ScreenShots

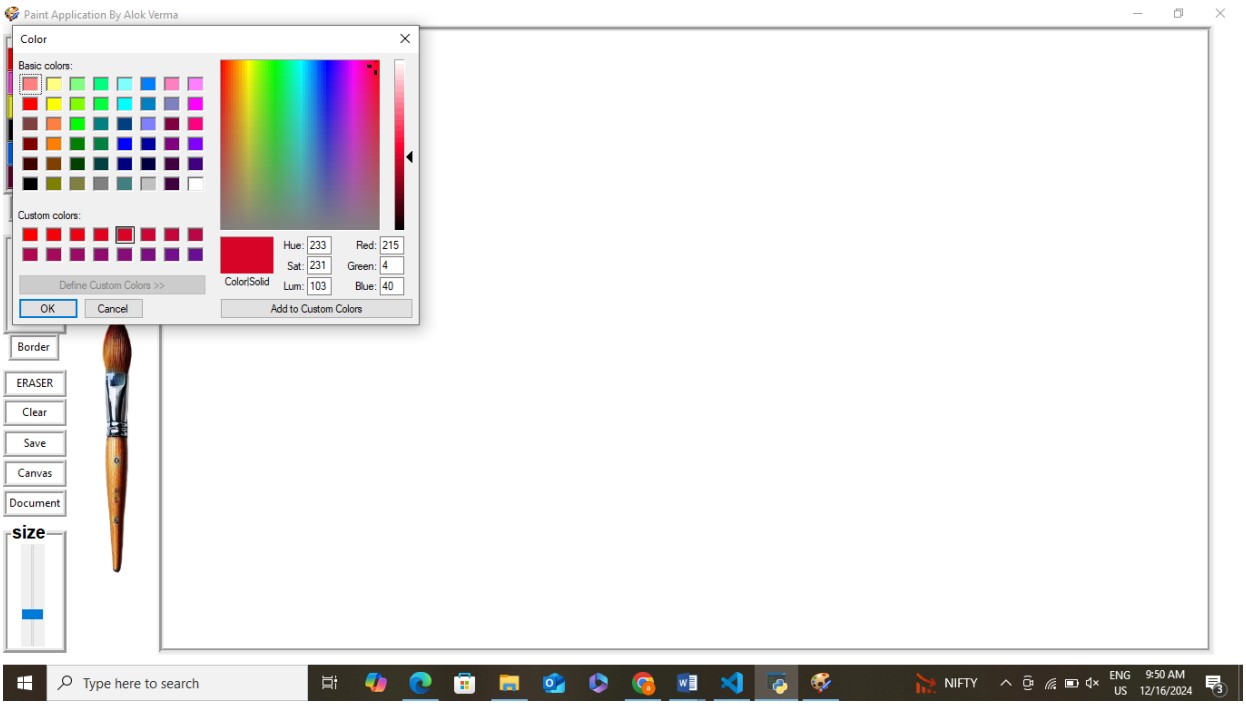
- **Main Window**



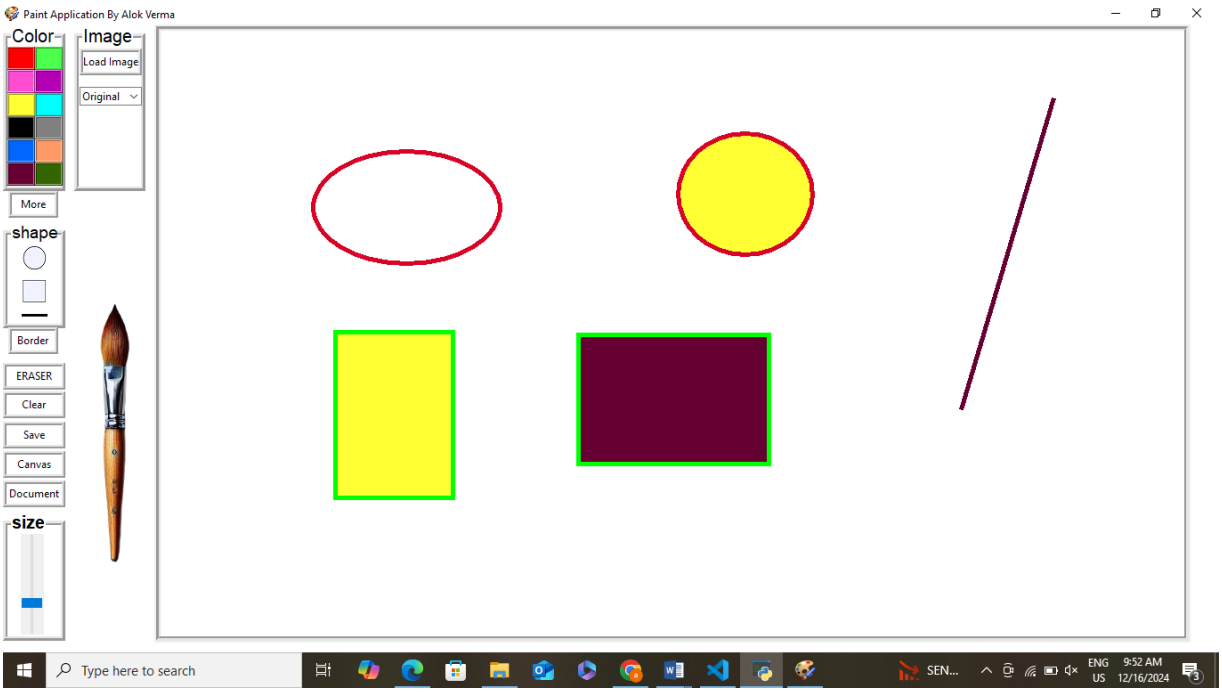
- **Pencil**



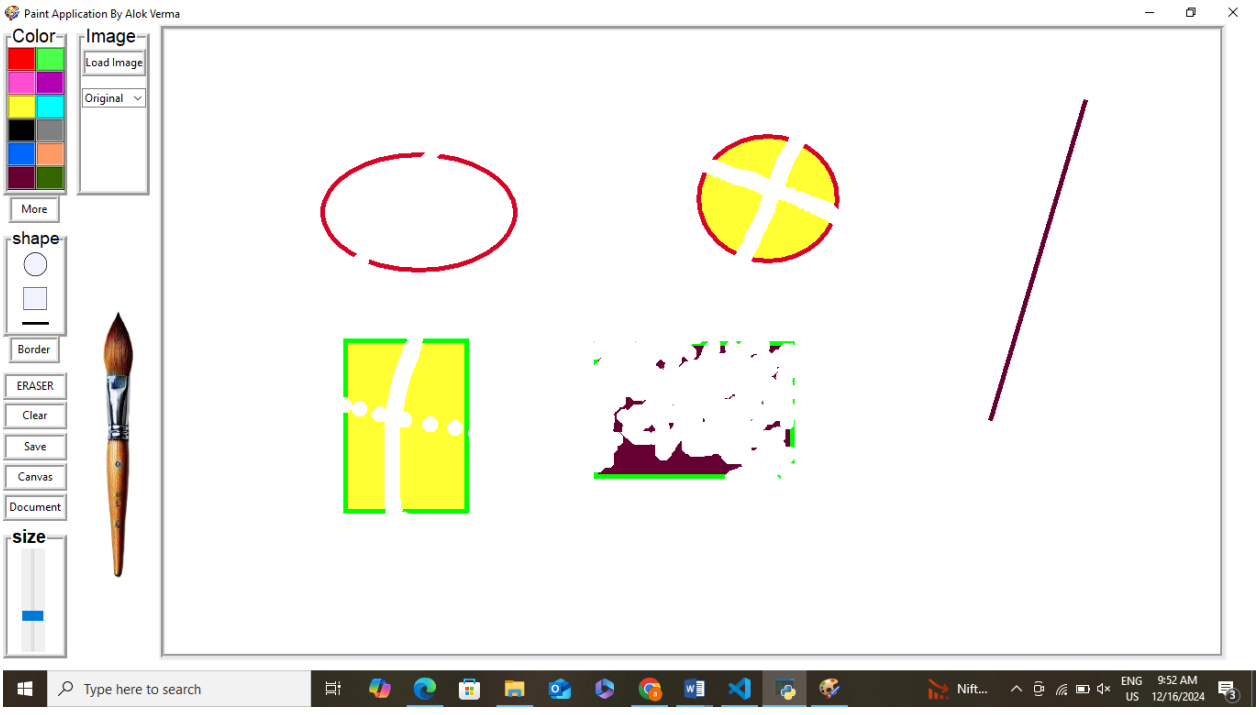
- **Color Selection**



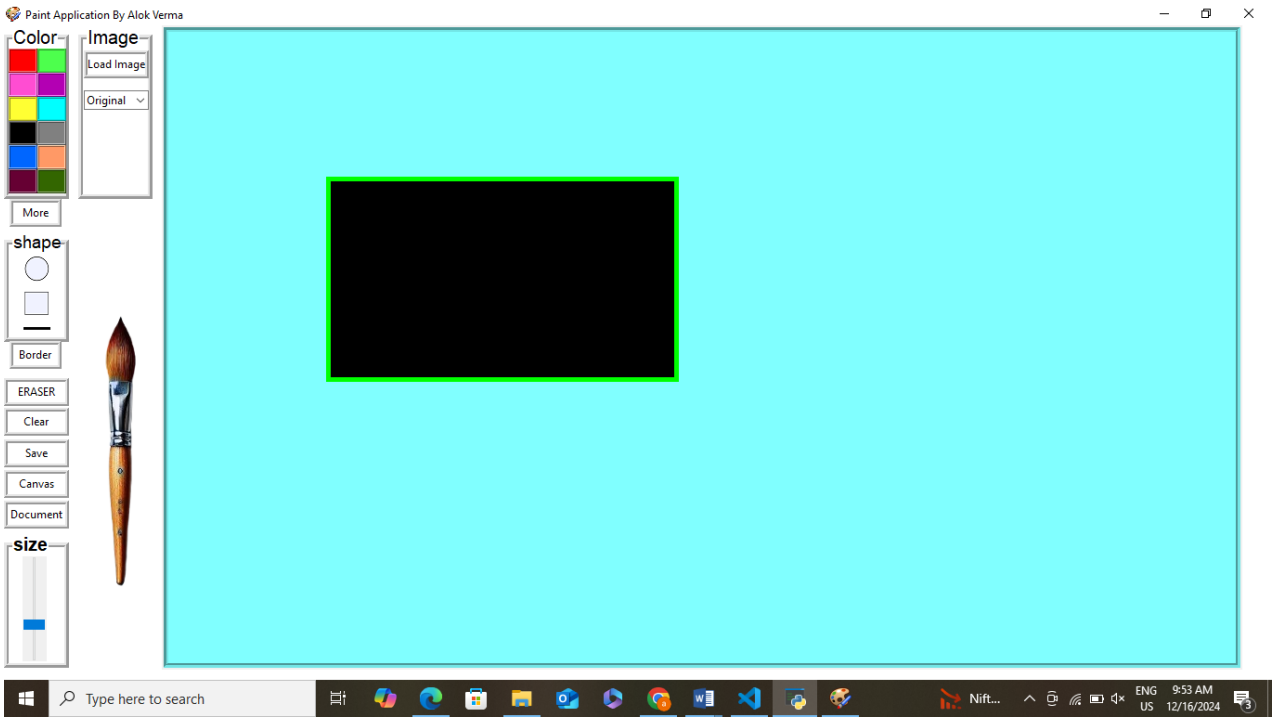
- **Shapes**



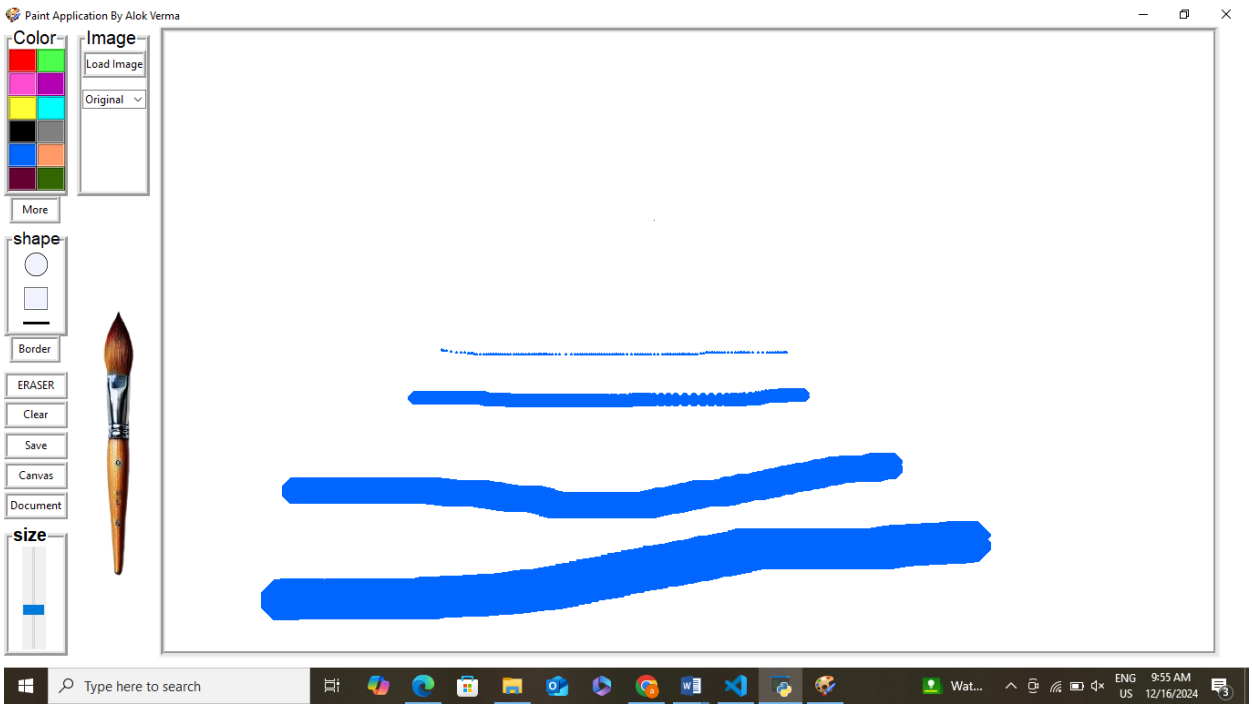
- **Eraser**



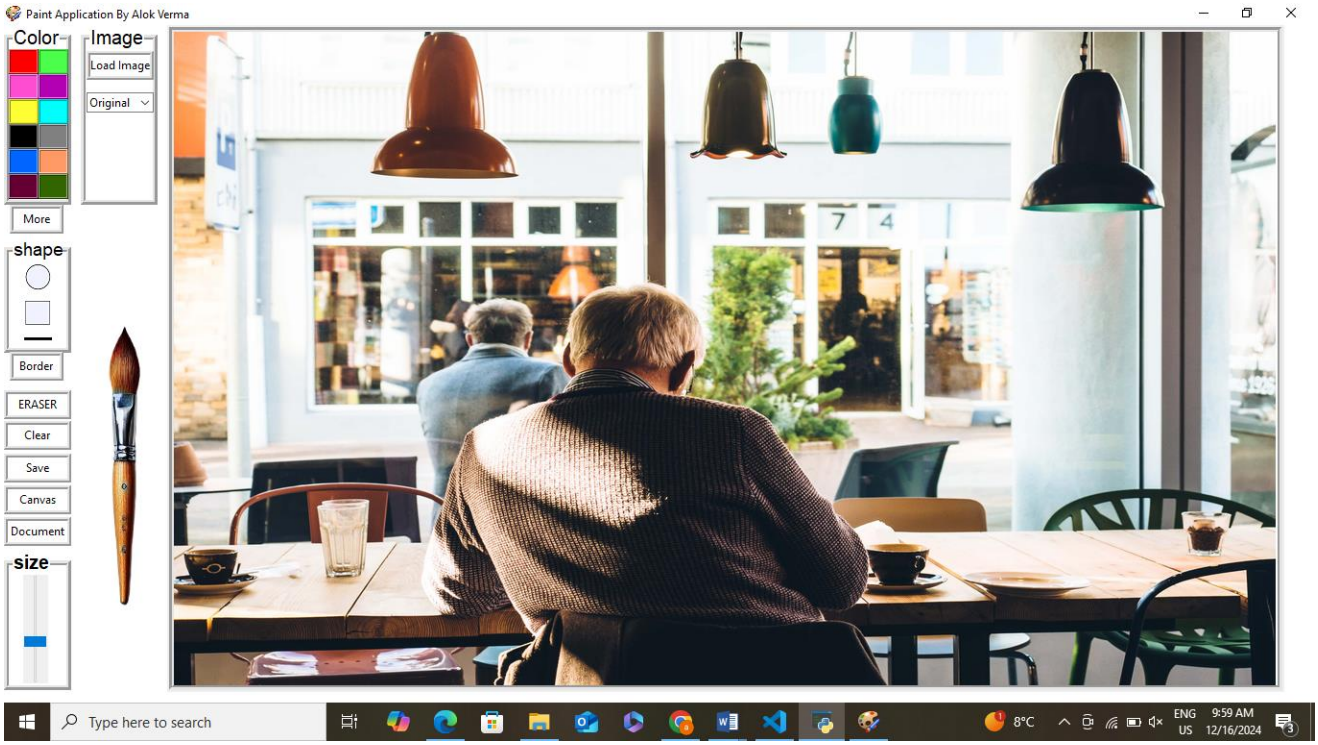
- **Canvas Bacground**



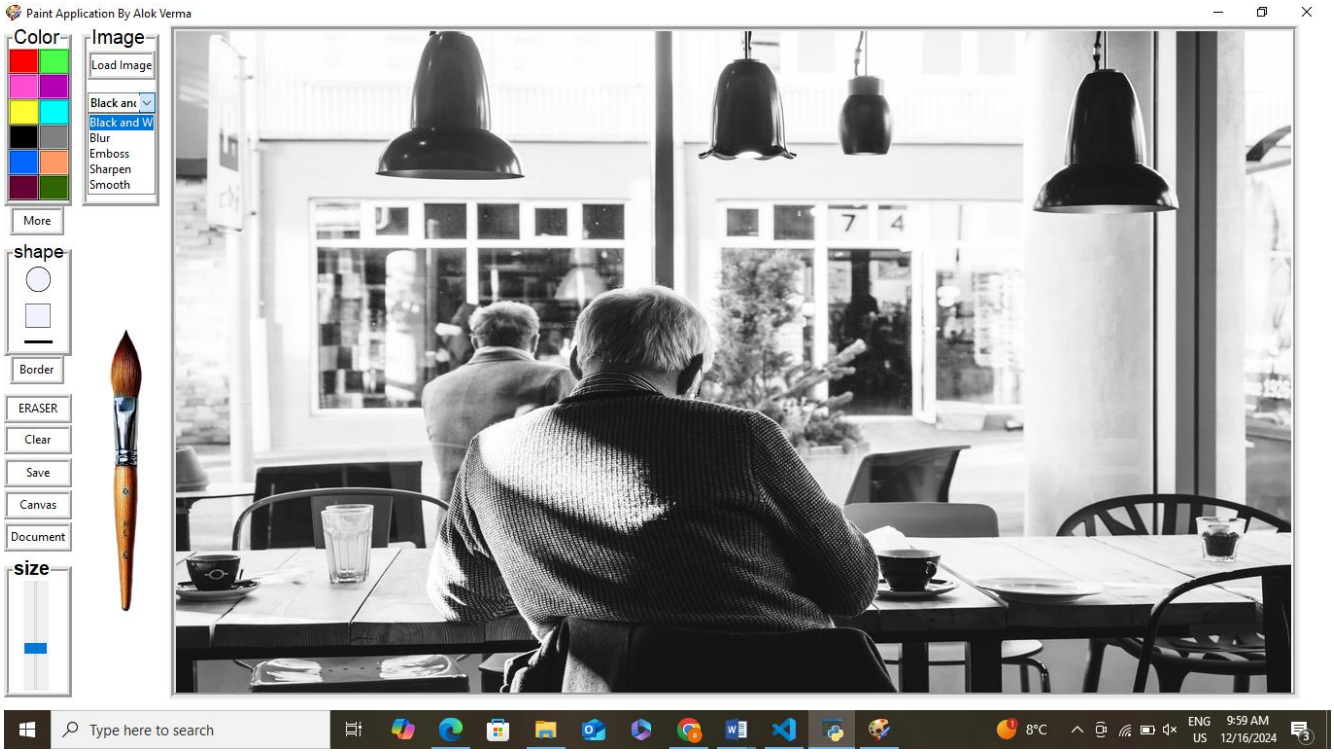
- **Scale for Thikness resize**



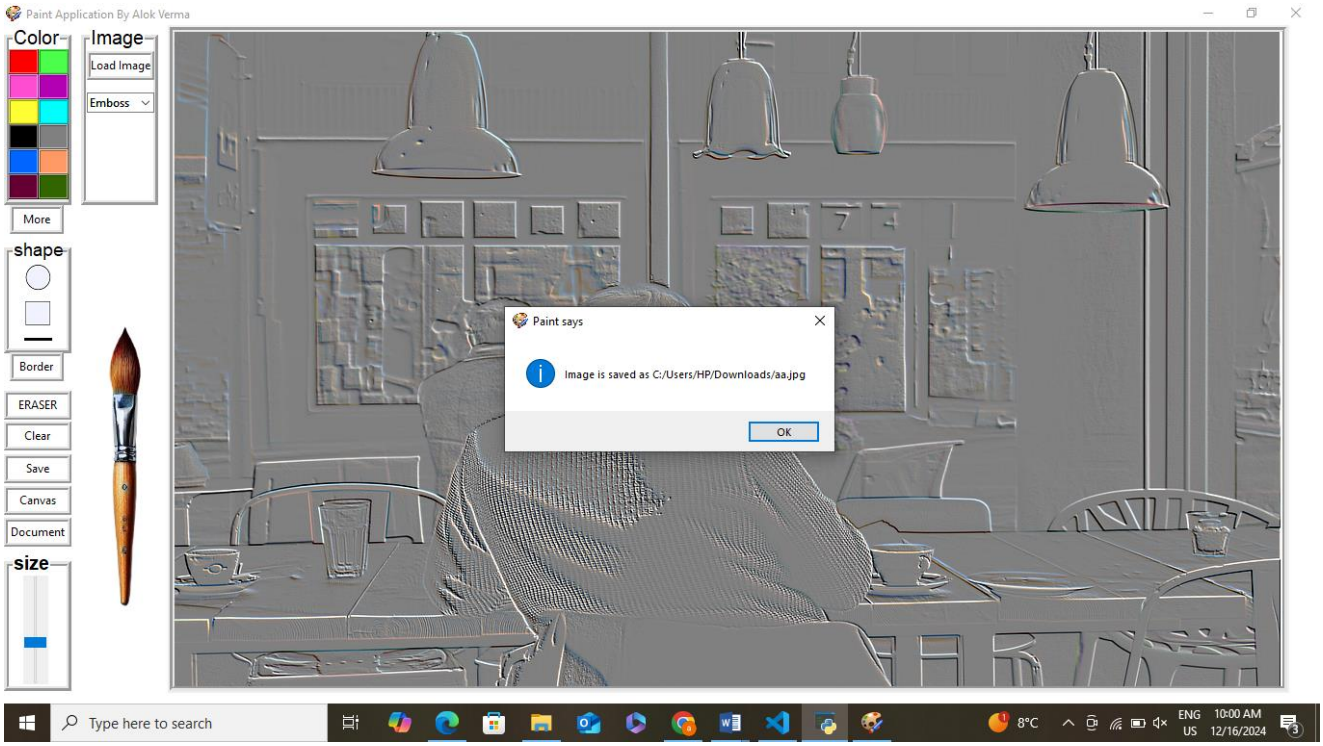
- **Image Import**

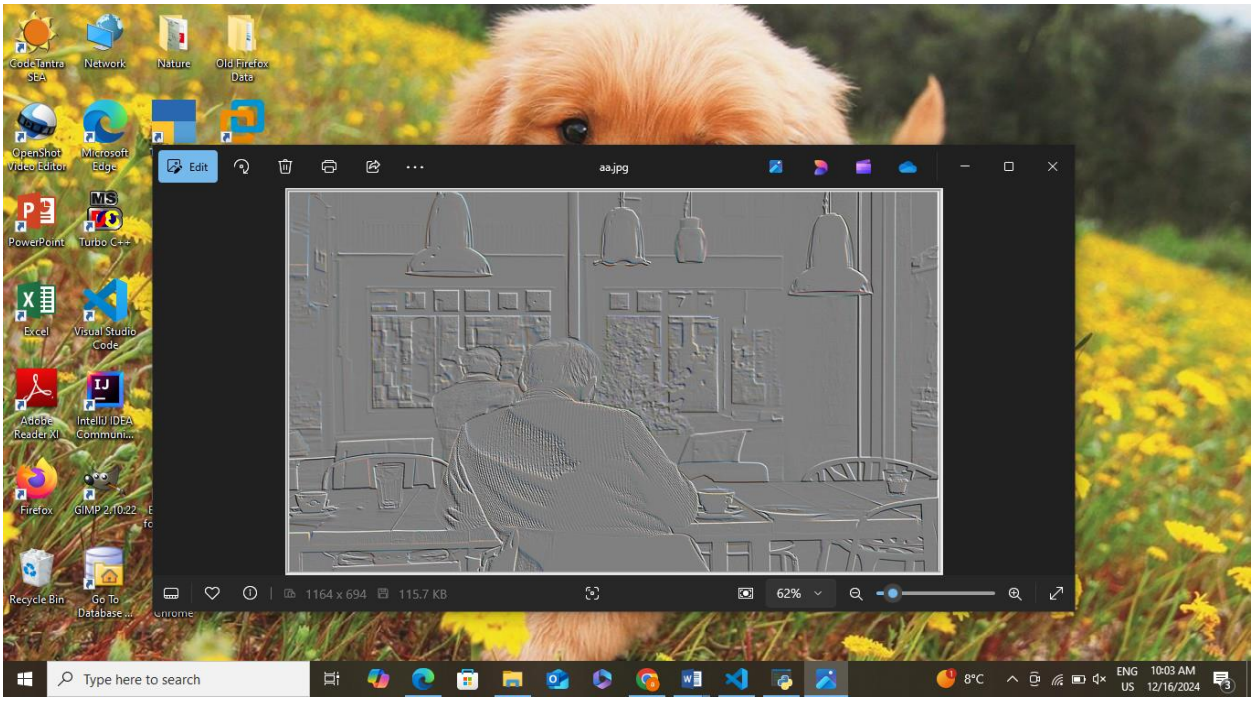


- **Image Filteration**

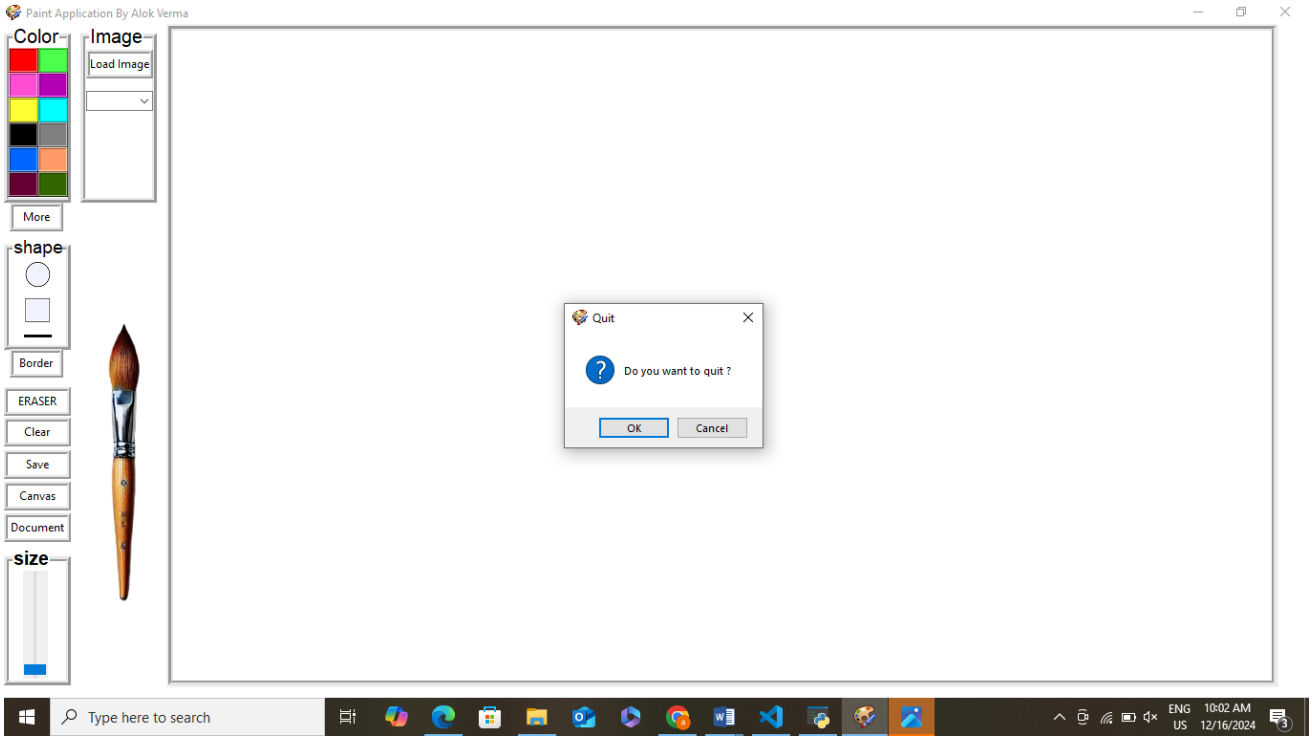


- **Save**


Paint Application By Alok Verma
Color
Image
Load Image
Emboss
More
shape
Border
ERASER
Clear
Save
Canvas
Document
size
Paint says
Image is saved as C:/Users/HP/Downloads/aa.jpg
OK
Type here to search
8°C
ENG
US
10:00 AM
12/16/2024


Edit
aa.jpg
1164 x 694
115.7 KB
62%
Type here to search
8°C
ENG
US
10:03 AM
12/16/2024

- **Quit Popup**



6.6 Scope of the System

6.6.*1 User Benefits*

- It is User-friendly.
- It is Easy to use.
- It is open source.
- It consist lots of color options.
- Less time consuming.
- It has some predefined Shapes that is very useful for User.

- User can save file permanently.
- Easily Upgradable.
- Simple and intuitive controls for all age groups.
- Runs seamlessly on Windows, macOS, and Linux.

6.6.2 Limitations

- Shapes are not resizable post-drawing.
- No undo/redo functionality.
- Currently unavailable for mobile and tablet devices.

6.7 Future Enhancements

Some of the features are lack in our system. We will try to add all those features on future. Our main Focus on some of the features in future are:-

Key areas of improvement include:

1. Undo/Redo functionality.
2. Additional pre-defined shapes.
3. Improved image editing features.
4. Support for mobile platforms.
5. Adding cut , copy and paste method.
6. Adding different types of effect options.
7. Adding Text on Canvas.

Chapter 7

REFERENCES

1. Python Documentation: https://www.python.org
2. Tkinter Documentation: https://wiki.python.org/moin/TkInter
3. Pillow (PIL) Documentation: https://pillow.readthedocs.io
4. W3Schools Python GUI Tutorial: https://www.w3schools.com
5. TutorialsPoint Python GUI Programming: https://www.tutorialspoint.com/python/python_gui_programming.htm