PHOTO EDITOR

Live project report file

Submitted in partial fulfilment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

SUBMITTED BY:		SUBMITTED TO:
Kunaal Kaushal Singh	: 11022210021	Ms. Jyotishna
Shivankur Sharma	: 11022210033	
Jefin Majo	: 11022210036	
Arpit	: 11022210058	

DEPARTMENT OF COMPUTER SCIENCE

SRM UNIVERSITY DELHI-NCR, SONEPAT HARYANA-131029

APPROVAL SHEET

This Project work entitled **Photo Editor** by **Kunaal Kaushal Singh (11022210021) , Shivankur Sharma (11022210033) , Jefin Majo (11022210036)** and **Arpit (11022210058)** is approved for the degree of **Bachelor of Technology.**

Examiners

Supervisor(s)

Project Coordinator

Head of the Department

Prof.(Dr.) Puneet Goswami

Date:

Place: Sonipat

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the project entitled "**Photo Editor**" in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology and submitted in the Department of Computer Science & Engineering of SRM University, Delhi-NCR, Sonipat, Haryana, (India) is an authentic record of our own work carried out under the supervision of **Ms. Jyotishna** as live project in second semester during the academic year 2023-24. The matter presented in this project has not been submitted for the award of any other degree of this or any other Institute / University.

CERTIFICATE

This is to certify that the project entitled “Photo Editor” by Kunaal Kaushal Singh (11022210021) , Shivankur Sharma (11022210033) , Jefin Majo (11022210036) and Arpit (11022210058) to the Department of Computer Science & Engineering of SRM University Delhi-NCR, Sonipat, Haryana, India in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology under the Faculty of Engineering and Technology is an authentic record of the work carried out by her/him under my supervision. This work fulfils the requirement for which it has been submitted.

This project has not been submitted to any other University or Institution for any other degree and is submitted as live project in third semester during the academic year 2023-24.

Kunaal Kaushal Singh – 11022210021

Shivankur Sharma – 11022210033

Jefin Majo – 11022210036

Arpit – 11022210058

ACKNOWLEDGEMENT

This project was a quite a learning experience for us at every step. After the completion of this project, we have gained so much experience about the training and boost up my confidence. The successful completion of this project marks the beginning of an ever - going learning experience of converting ideas and concepts into real life, practical system. We feel the experience gained during the project will lead us to gain the bright prospect in the future.

I would like to give thanks to **Ms. Jyotishna** for giving us the opportunity to work on his project.

ABSTRACT

In an era dominated by visual communication, the need for user-friendly and accessible photo editing tools has never been more pronounced. Our project, a simple yet powerful Photo Editor developed in Python, seeks to address this demand by offering a versatile solution for users of all levels. With a focus on ease of use and essential functionalities, this project aims to empower individuals to enhance their images effortlessly.

This Photo Editor is designed to provide a seamless experience through a well-crafted graphical user interface built using the Tkinter library. Leveraging the capabilities of the PILLOW library, the editor facilitates fundamental image adjustments, filters, and the addition of text overlays. Whether users are looking to fine-tune personal photographs or add creative elements to their visuals, this project offers a user-friendly environment for achieving desired results.

As we delve into the intricacies of image processing and graphical interface design, this project not only serves as a practical tool but also as an exploration into the fusion of technology and creativity. Join us on this journey as we present a comprehensive yet accessible Photo Editor tailored for the modern digital landscape.

PROBLEM DOMAIN

Practical Uses for Our Photo Editor

Personal Touch-ups:

Scenario: Enhance your personal photos with easy adjustments like brightness, color tweaks, and simple filters.

Casual Social Media Posts:

Scenario: Create eye-catching posts for platforms like Instagram or Facebook with added filters, overlays, and text.

Basic Educational Material:

Scenario: Spice up presentations or projects by adding annotations, graphics, and visuals without the fuss.

Easy Graphic Tweaks:

Scenario: If you're new to design, our editor offers a hassle-free way to experiment with basic graphic elements.

Playful Artistic Touch:

Scenario: Unleash your creativity by experimenting with artistic filters and effects for a playful touch.

Our Photo Editor is designed with simplicity in mind, making it accessible for various everyday needs.

TECHNOLOGIES USED

The Photo Editor code utilizes several technologies to create a simple yet effective image editing tool. Here's a description of the key technologies used:

Tkinter:

Role: Tkinter is Python's standard GUI (Graphical User Interface) toolkit, and it is used to build the graphical interface for the Photo Editor.

Description: Tkinter provides the basic building blocks for creating windows, frames, buttons, and other GUI elements. It facilitates the organization and layout of the user interface components.

PIL (Pillow):

Role: PIL (Python Imaging Library) or its fork Pillow is used for image processing tasks in the Photo Editor.

Description: PIL is a powerful library for opening, manipulating, and saving various image file formats. In this project, Pillow is employed to handle image resizing, applying filters, and converting images to a format compatible with Tkinter.

filedialog, simpledialog, and colorchooser (Tkinter submodules):

Role: These modules are part of the Tkinter library and are used for interacting with the user, specifically for opening file dialogs, input dialogs, and color selection dialogs.

Description: filedialog is used for opening a file dialog to select an image. simpledialog helps in getting input from the user, such as text for adding to the canvas. colorchooser enables the selection of colors for text in the editor.

ttk (Tkinter Themed Widgets):

Role: The ttk module is used for creating themed widgets in Tkinter, and it is employed here for the ComboBox.

Description: ttk.Combobox is utilized to create a drop-down menu for selecting different image filters. Themed widgets provide a more modern and consistent look to the GUI.

Canvas Widget (Tkinter):

Role: The Canvas widget is a crucial component for displaying images and other graphical elements.

Description: The canvas serves as the main drawing area where images are loaded, text is added, and filters are applied. It provides a versatile space for visual content.

KEY FEATURES

Image Loading and Display:

The code incorporates functionality to load images from the user's file system. Upon selection, the image is opened, resized for optimal display, and showcased on the canvas. This allows users to easily bring their desired images into the editor.

Text Insertion:

Users can enhance their images by adding text directly onto the canvas. The editor prompts users to input text through a dialog, and the entered text is dynamically placed at the center of the canvas, providing a convenient way to include textual elements in images.

Color Selection for Text:

The editor empowers users with the ability to customize the appearance of added text. Through a color selection dialog, users can choose their preferred text color, adding a personalized touch to the textual elements within the image.

Canvas Clearing:

The "Clear" button offers users a quick means to reset the canvas, removing all content. This feature is valuable for users who want to start anew or make iterative adjustments to their creations without clutter.

Image Filtering:

The code integrates image filtering capabilities, enabling users to apply various visual effects to the loaded image. Filters such as "Black and

White," "Blur," "Emboss," "Sharpen," and "Smooth" are implemented, allowing users to experiment with different artistic styles.

Combobox for Filter Selection:

The use of a Combobox provides an organized and user-friendly way to select different image filters. Users can easily choose from the available filters, and the selected filter is promptly applied to the loaded image, offering a seamless editing experience.

Modularity and Structured Code:

The code is well-structured and modular, with each function serving a specific purpose. This approach enhances code readability, making it easier to maintain and expand upon. The modular design promotes a clear separation of concerns, contributing to overall code organization.

Graphical User Interface (GUI):

Tkinter, as the GUI toolkit, contributes to the creation of an intuitive interface. Buttons and widgets are strategically placed within a left frame, creating an aesthetically pleasing layout. The canvas, serving as the primary drawing area, ensures a focused and visually coherent user experience.

Event Handling:

Events such as button clicks and filter selections are effectively captured through event binding. This allows the editor to respond dynamically to user actions, ensuring a responsive and interactive editing environment.

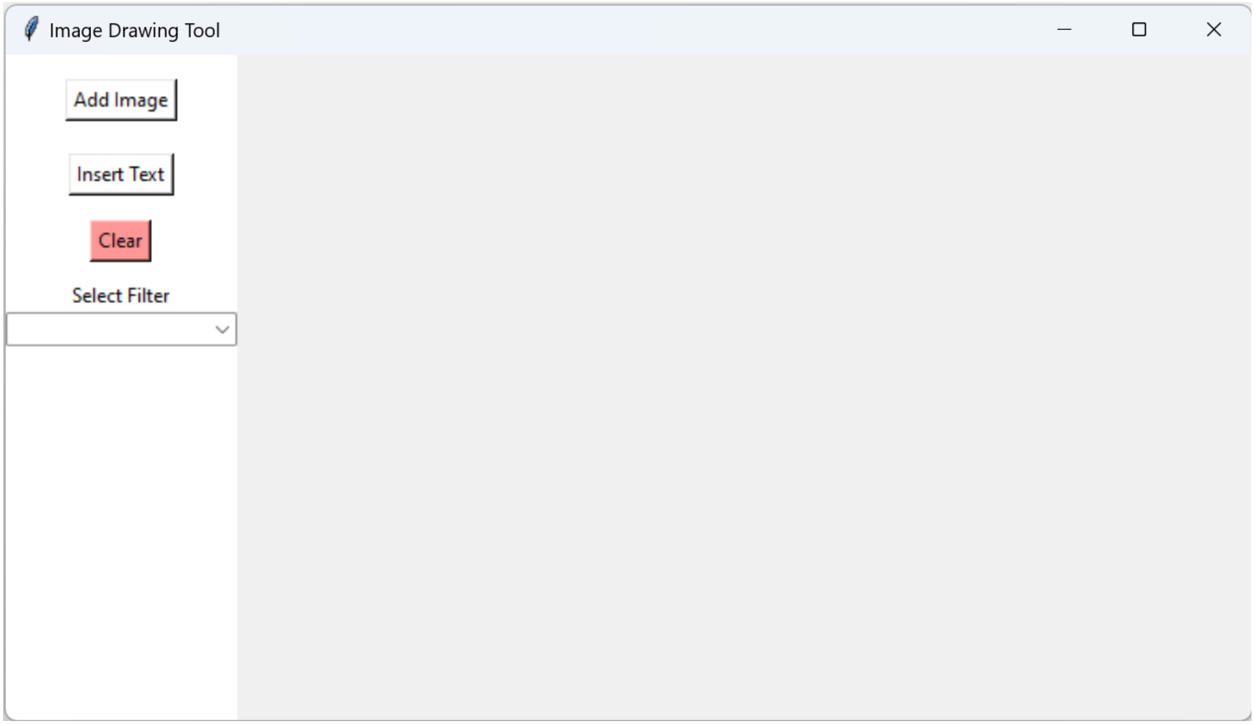
User-Friendly Interaction:

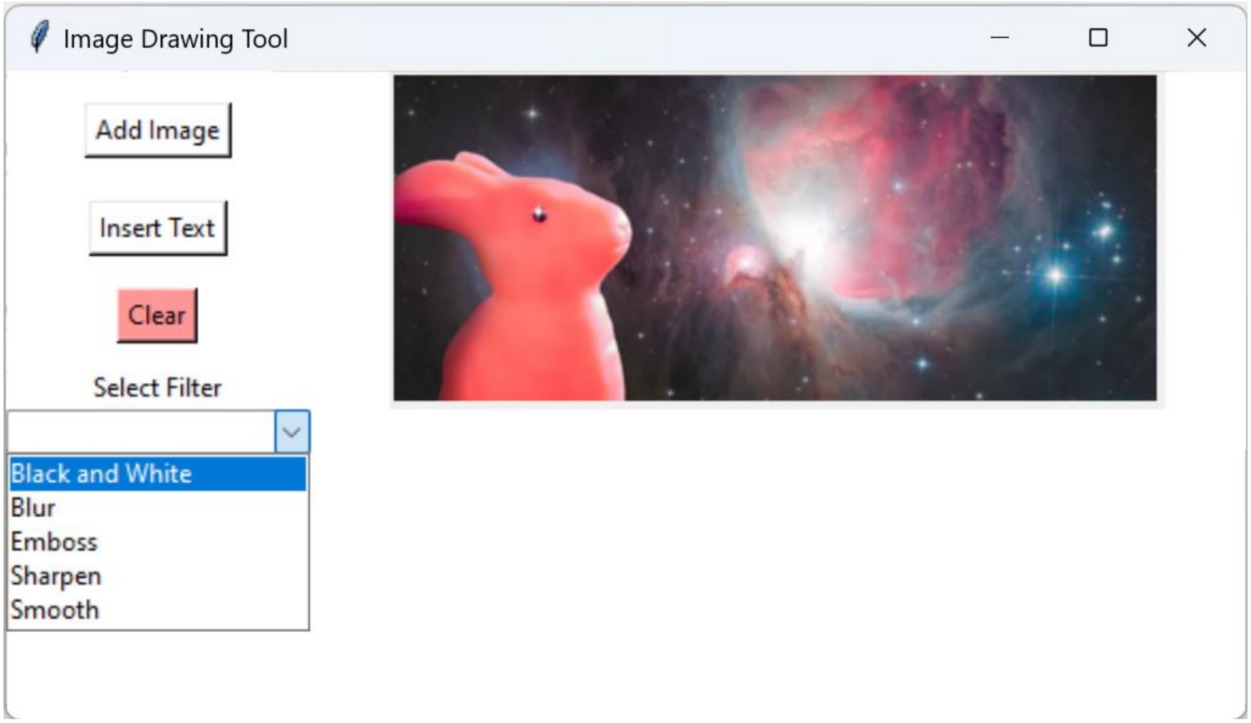
The overall design of the editor prioritizes user-friendliness. Simple buttons for common actions and dialogs for user input contribute to an intuitive interface, ensuring that users can easily navigate and utilize the various features of the Photo Editor. This focus on user experience makes the editor accessible to a broad audience.

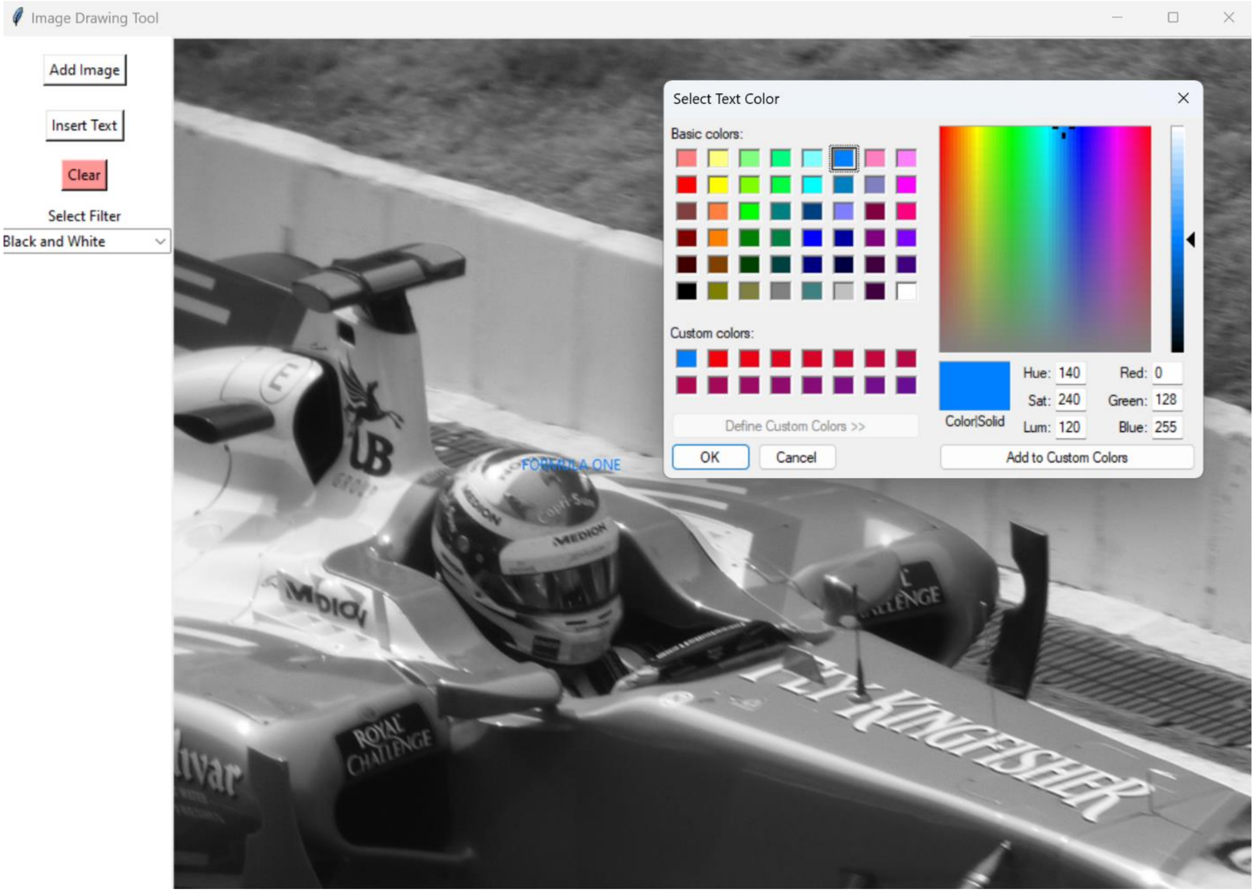
SOURCE CODE

```
import tkinter as tk
from tkinter import filedialog, simpledialog
from tkinter import colorchooser
from PIL import Image, ImageOps, ImageTk, ImageFilter
from tkinter import ttk

root = tk.Tk()
root.geometry("1000x600")
root.title("Image Drawing Tool")
root.config(bg="white")
file_path = ""
def add_image():
    global file_path
    file_path = filedialog.askopenfilename(
        initialdir="D:/codefirst.io/Tkinter Image Editor/Pictures")
    image = Image.open(file_path)
    width, height = int(image.width / 2), int(image.height / 2)
    image = image.resize((width, height))
    canvas.config(width=image.width, height=image.height)
    image = ImageTk.PhotoImage(image)
    canvas.image = image
    canvas.create_image(0, 0, image=image, anchor="nw")


def change_color():
    color = colorchooser.askcolor(title="Select Text Color")[1]
    insert_text(color)


def insert_text(color):
    text = simpledialog.askstring("Input", "Enter text:")
    if text:
        canvas.create_text(canvas.winfo_width() / 2, canvas.winfo_height() / 2,
text=text, fill=color)


def clear_canvas():
    canvas.delete("all")
    canvas.create_image(0, 0, image=canvas.image, anchor="nw")
def apply_filter(filter):
    image = Image.open(file_path)
    width, height = int(image.width / 2), int(image.height / 2)
    image = image.resize((width, height))
    if filter == "Black and White":
        image = ImageOps.grayscale(image)
    elif filter == "Blur":
        image = image.filter(ImageFilter.BLUR)
    elif filter == "Sharpen":
        image = image.filter(ImageFilter.SHARPEN)
    elif filter == "Smooth":
        image = image.filter(ImageFilter.SMOOTH)
    elif filter == "Emboss":
        image = image.filter(ImageFilter.EMBOSS)
    image = ImageTk.PhotoImage(image)
    canvas.image = image
    canvas.create_image(0, 0, image=image, anchor="nw")


left_frame = tk.Frame(root, width=200, height=600, bg="white")
left_frame.pack(side="left", fill="y")

canvas = tk.Canvas(root, width=750, height=600)
canvas.pack()

image_button = tk.Button(left_frame, text="Add Image",
                         command=add_image, bg="white")
image_button.pack(pady=15)

color_button = tk.Button(
    left_frame, text="Insert Text", command=change_color, bg="white")
color_button.pack(pady=5)

clear_button = tk.Button(left_frame, text="Clear",
                         command=clear_canvas, bg="#FF9797")
clear_button.pack(pady=10)

filter_label = tk.Label(left_frame, text="Select Filter", bg="white")
filter_label.pack()
filter_combobox = ttk.Combobox(left_frame, values=["Black and White", "Blur",
                                             "Emboss", "Sharpen", "Smooth"])
filter_combobox.pack()

filter_combobox.bind("<<ComboboxSelected>>",
                     lambda event: apply_filter(filter_combobox.get()))

root.mainloop()
```

REVIEW IMAGES







Select Filter: Black and White

Hue: 140
Sat: 240
Lum: 120
Red: 0
Green: 128
Blue: 255

CONCLUSION

In conclusion, the Photo Editor project presents a simple yet effective solution for users seeking a user-friendly and accessible tool for basic image editing. Leveraging the Tkinter library for GUI development and the Pillow library for image processing, the editor allows users to seamlessly load images, add text, apply filters, and clear the canvas with ease.

The project's modular and structured code design fosters readability and maintainability, while its intuitive graphical user interface ensures a straightforward editing experience. Key features such as text insertion with customizable colors, image filtering, and canvas clearing contribute to a versatile and engaging platform.

As the project evolves, future enhancements could include advanced features such as undo/redo functionality, image rotation, layer support, and additional image filters. These enhancements would further empower users and broaden the editor's capabilities.

In essence, the Photo Editor project is a stepping stone towards a more feature-rich image editing tool, balancing simplicity with potential for future expansion. Whether used for personal image enhancements, creative projects, or educational purposes, the Photo Editor provides a foundation for users to explore and express their creativity in a visually dynamic way.