CHAPTER 1

INTRODUCTION

Since the advancement of technology, businesses are now able to perform some operations more smoothly and efficiently. Apart from the most discussed aspects such as video or audio editing, there are also other significant features such as photo manipulation, which helps with smooth business operations, especially for industries such as e-commerce, car retail or real estate, etc.

Here, we will have a look at photo manipulation services. It is seen that photo manipulation impacts not only the subject but also the viewer. It can give image dimension as well as a virtual makeover to the photograph, thus aiming to boost the confidence of a person.

This is a process, by which a person in a photograph is transformed or changed to something more creative or appealing, on grounds of the artist's perception, rather than going with the original image. One thing to be sure of is that the clarity and informativeness of the theme were to be taken into consideration.

A manipulated image will have a great impact and is normally considered influential. The image is made to look more potent than it originally is.

At the same time, the photos are used by businesses as a way to boost the marketing strategies or campaigns as well as to conduct intensive training for individuals, whose aim is to create impressive illustrations for personal or career purposes. It would be more effective in reaching out to its target audience.

Photo Manipulation can be a way of getting a second chance to get the image the way you would have wanted to capture it in-camera.

When it comes to photo manipulation and creation, many customers and staffs stick to using the clip art package that came with their word processing application because they think that is all that they can afford. In reality there are many powerful open-source tools available to make your photos look more professional.

Thus, we provide a tool called PIMP to manipulate the photograph easily. PIMP is developed using python with software requirements Windows 7 or above / Linux / Mac os, VS Code, Adobe photoshop and programming language is used python 3.7 and python libraries like Libraries. TKinter , Pillow , json.

PIMP offers an affordable alternative in the place of Adobe Photoshop. We decide to develop PIMP because, it is open-source in the real world.

Purpose to develop PIMP.

Staff and customers were in need of a graphics manipulation program and products like Adobe Photoshop, they were too costly for small manipulation with limited funds. Though picture Image Manipulation using python (PIMP) provided a basic range of features, similar to Photoshop, and looked enough to implement in the application. PIMP can also use this application for free. Customers/staffs can use PIMP to do light photo editing. The images they edit tend to be uploaded to photo sites, or sent to local photo labs for printing. The common uses of this application include cropping, resizing and adjusting levels and saving in different formats. We also use the filter feature when adjusting or building more complex images. Overall, it is easy enough that someone can sit down and do simple image manipulation without much training or study.

With photo manipulation, there is no limit to the effects that can be brought about in a simple photograph. You can make a simple landscape come alive with the colors of the rainbow. You can make any event look and feel more vibrant, fun and appealing with photo editing. These photographs can be fixed even if they are damaged. Photo editing can bring to life any picture with more color and joy!

Helpful in E-Learning

Many of the college needs the digital/virtual learning in their education thus teachers and students go through the perfect images to make the class interactive and if it comes for student to make the project look in attractive. not only to the learning purpose but for also the campus/college cover magazine ,id card etc.
Photo editing has become such a huge part of modern culture that even those who have never used it have often heard of it. when it comes to our college we had a some purpose to to work with the images, for our faculties and students but they used the paid source and some other applications which are not that much satisfiable they used to spend lot of time in editing the images and retouching.
when it comes to the teachers/lectures there was not much interactive connection between their e-learning materials towards the students. when it comes to students they were not able to give the presentation efficiently all above both were using few paid source of applications which was not that useful for using those applications often though they had to pay the expensive fees to work on it.
to overcome all these issues we chose to develop the image manipulation application called PIMP(Picture Image Manipulations) which slightly enact as the gimp software
PIMP is a software application for image editing and photo retouching for use on Windows computers. PIMP offers users the ability to create, enhance, or otherwise edit images, adding effects and to edit images, create new high-quality images, or both. etc.
Since people do judge a book by its cover (it's true), making a decent-looking image is a good thing.
We are going to strip away all the complexity for you to introduce through the essential features that you need to know to quickly and easily process your digital images, improve their look, and prepare them for use in your e-learning materials. If you're a teacher, a trainer, or an instructional designer, then you use a ton of images in your teaching and learning materials. It doesn't matter if you teach face-to-face, blended, or fully online. Images help you explain concepts, engage with your learners, and enhance the usability of your materials. If you're a student, then you too, are using a lot of images to illustrate your points, demonstrate your understanding of concepts, and showcase your work. Anyone who gives a presentation to a class, at a conference, or any business or organization, needs to have a solid foundation in digital image literacy to refine your story and drive your points home in a memorable way. For all of these reasons, you need a basic understanding of how to use PIMP to get this done as quickly and as efficiently as possible.
The first BIG use is to manipulate images!
This "manipulate images" is an umbrella term and there are thousands of things you can do. Here are just a few examples...
Add or remove filters
change the color of an object.
Add effects
Fix an old photo to make it new
Improve Your Product Photos
Rotating the image & cropping the image

when it comes for teachers teaching with objects and photographs creates a direct, sensory connection between learners and their subjects that results in new levels of interest and attention. Teaching with objects also creates students with higher levels of visual literacy faculties/lectures have reported that their use of images in the classroom has led to increased student interactivity and discussion. Teaching with images can also help develop students' visual literacy skills, which contributes to their overall critical thinking skills and lifelong learning when it comes to student Images manipulated by PIMP help you explain concepts,

engage with your learners, and enhance the usability of your materials.as a student, then you too, are using a lot of images to illustrate your points, demonstrate your understanding of concepts, and showcase your work.
This is how PIMP helped in college and also other editing images.

When to use PIMP

However, on the other side, if you do not work professionally and need a pocket-friendly tool, PIMP can just be the right one for you. Although you can't say that working one PIMP doesn't require concerned knowledge, still you can begin with basic image manipulation and editing work. Working on PIMP won't bother you much as it is open-source and free. PIMP can be the right companion to begin your editing journey with.

CHAPTER:2

Implementation (Python)

Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamic semantics, used for general purpose programming. It was created by Guido van Rossum, and first released on February 20, 1991.

While you may know the python as a large snake, the name of the Python programming language comes from an old BBC television comedy sketch series called Monty Python's Flying Circus. One of the amazing features of Python is the fact that it is actually one person's work. Usually, new programming languages are developed and published by large companies employing lots of professionals, and due to copyright rules, it is very hard to name any of the people involved in the project. Python is an exception. Of course, van Rossum did not develop and evolve all the Python components himself. The speed with which Python has spread around the world is a result of the continuous work of thousands (very often anonymous) programmers, testers, users

Why Python?

What makes Python so special? How does it happen that programmers, young and old, experienced and novice, want to use it? How did it happen that large companies adopted Python and implemented their flagship products using it?

There are many reasons – we've listed some of them already, but let's enumerate them again in a more practical manner: it's easy to learn – the time needed to learn Python is shorter than for many other languages; this means that it's possible to start the actual programming faster; it's easy to teach – the teaching workload is smaller than that needed by other languages; this means that the teacher can put more emphasis on general (language-independent) programming techniques, not wasting energy on exotic tricks, strange exceptions and incomprehensible rules;it's easy to use for writing new software – it's often possible to write code faster when using Python; it's easy to understand – it's also often easier to understand someone else's code faster if it is written in Python; it's easy to obtain, install and deploy – Python is free, open and multiplatform; not all languages can boast that.

CHAPTER:3

OBJECTIVES

Upon successful completion of this project, members should be able to:

- Think and plan before editing photos.
- Have a basic working knowledge of digital photo editing.
- Edit good photos into great photos.
- Create visually interesting photos of people, places and things.
- Feel comfortable experimenting with their editing skills to improve photos.
- Identify opportunities for creatively editing photos.
- Use digital photography editing to communicate with people/the world.
- Tell photo stories in an interesting and engaging way.
- Challenge themselves with a variety of photo editing techniques.
- Identify editing techniques that have been applied to photos.

There are a lot of reasons to edit your photos. For most people, the biggest reason is to fix a mistake. Let's say you took a great photo but there was some dirt on your lens. Now you have a dark spot on your photo. Also, your photo may seem kind of dark. You can easily lighten a photo by changing the "brightness" level. Now your photo is flawless! Another reason you might edit your photo is to produce artistic effects. For example, maybe your image seems a little "busy". This might mean using what's called a "blur" effect to create a point of focus that draws the eye in. For instance, if you have a photo of a bunch of fall leaves on the grass, but they're all in focus and it seems distracting, you can use the blur feature to blur out some of the leaves, which is essentially changing the "depth-of-field" of your photo. You are creating a shorter depth-of-field and focusing on just one area of the image, which creates something with an entirely different feel. With your photo editing software, you can reveal the true potential of your photos. Using the skills you'll learn in this project, you'll be able to manipulate your pictures any way you like, bringing colors to life, making focus that much sharper, and even bringing photos back from the brink of the trash folder.

CHAPTER:4

SOFTWARE REQUIREMENTS SPECIFICATION

4.1 HARDWARE REQUIREMENTS:

- Processor – Intel or AMD with 64 bit support
- Memory – Minimum 4 GB of RAM
- Disk Space – 100MB
- Graphics Memory – 512MB or above

4.2 SOFTWARE REQUIREMENTS:

- Windows 7 or above
- Visual studio code
- Adobe Photoshop

4.3 PROGRAMMING LANGUAGE:

- Python 3.10.4

4.4 LIBRARIES:

- **TKinter** : It is the de facto way in python to create the graphical user interface and is included in all python dictionaries
- **Pillow** : Python pillow library is used to image class within it to show the image. The image modules that belong to the pillow package have a few inbuilt functions such as load images or create new images, etc. Opening, Rotating and Displaying an Image Opening an image is a basic operation of the image processing.
- **Json** : JSON is a syntax for storing and exchanging data. JSON is text, written with JavaScript object notation. Python has a built-in package called json, which can be used to work with JSON data. If you have a JSON string, you can parse it by using the json. loads () method. The result will be a Python dictionary.
- **Ttk bootstrap :** python -m pip install ttkbootstrap Simple Usage Instead of using long, complicated ttk style classes, you can use simple keywords with the "bootstyle" parameter. import ttkbootstrap as ttk from ttkbootstrap. **constants import * root = ttk**.

The Tools and Techniques of PIMP

There are many problems that can be fixed or elements that can be enhanced with photo editing software. Most editing programs employ the features and tools that we'll encounter in this project listed below. Some examples of photo editing programs are Adobe Photoshop, iPhoto, Corel Paint shop, Photo scape, Google Picasa and GIMP, to name a few. Don't worry if you're using a different program or your software's features have different names than what I've listed below. It's all basically the same concept. I'll be exploring some of these tools in greater detail in the coming units, so consider this an introduction to all the handy tools you can use to edit your photos. The best way to find the specific tool in your program is to search for the term (for example, "cropping") in the program's "Help" section.

Rotate: When you take a vertical photo, sometimes your computer mistakes it for a horizontal photo when you load it into your editing program. For example, you may have taken a photo of the CN Tower, but when you view it on your computer, it looks like it fell over. No problem! With the handy "rotate" tool, which often looks like a rounded arrow, you can turn the photo at 90-degree increments until you get it perfect.

Cropping: Cropping allows you to create a new image by isolating a portion of your existing photo and cutting out unwanted areas. The cropping tool usually looks like a box . Click this tool and enter the values to crop photo and then click on "preview".

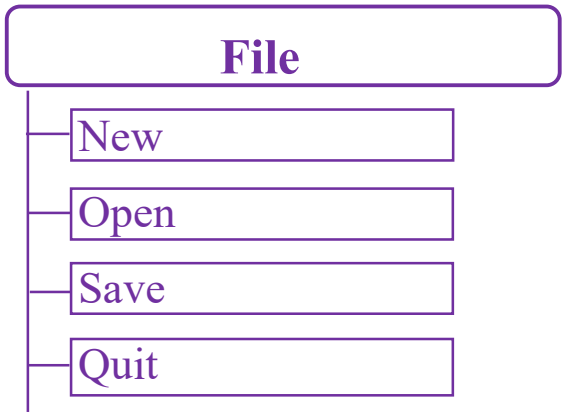
Contrast/Brightening: Contrast and brightening tools work hand in hand with the tools listed above to help you repair your exposure. You can use the brightness tool to brighten or darken your overall photo. The contrast tool's addition of colour depth can help define the tones in your photo, making it appear more evenly exposed. In addition, if your photo has very little colour depth (and this is especially noticeable if you decide to make your photo black and white), the contrast tool will also provide more definition and interest. Contrast tools can also be used to soften colour depth, which you might want to use if you've taken a portrait that makes your subject look a little rough around the edges. These tools look like slider bars along a line that can be moved left to right to adjust the overall contrast and brightness of your photo.

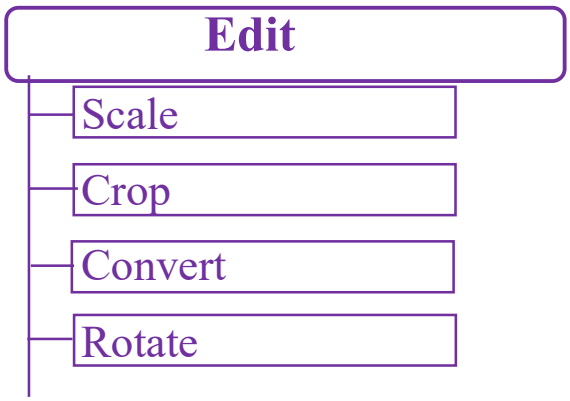
Saturation: tools help to enhance or subdue the amount of colour in your picture. This tool will be present under "filters".

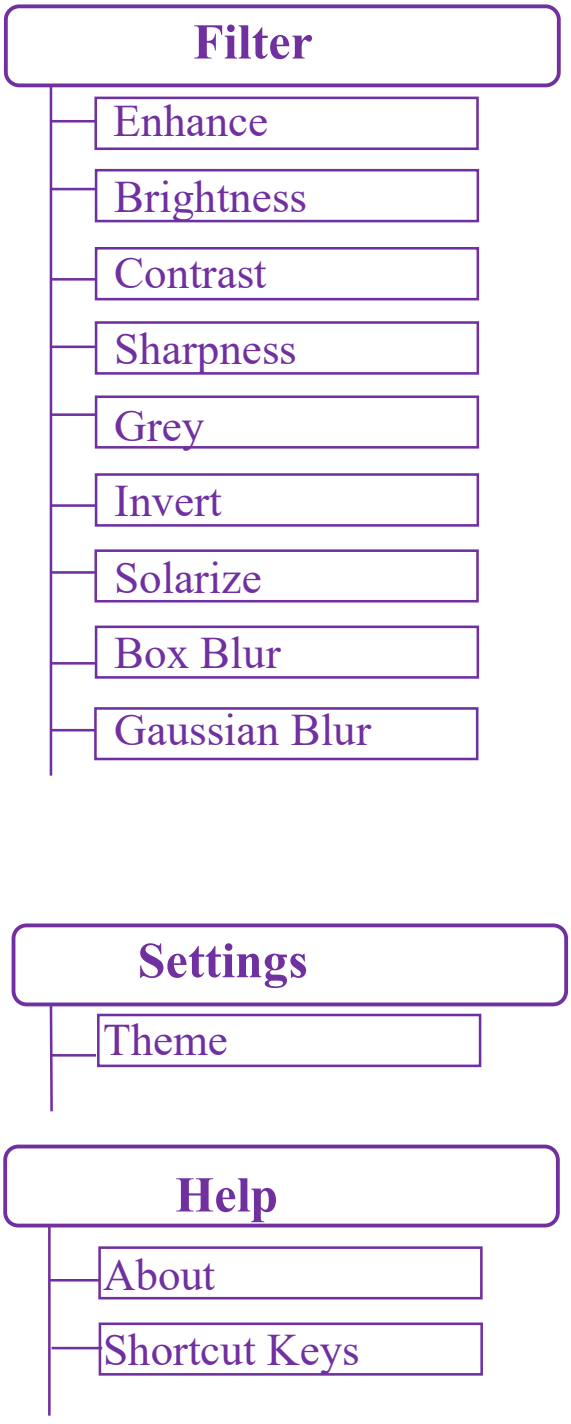
Sharpening: Was your focus off just a little bit? Is your subject a tiny bit fuzzy around the edges? You can use the "sharpen" tool to make the edges more crisp or defined. Sharpen tools can usually be found in your "filters" menu. Like brightness and contrast, this tool looks like a slider bar that can be moved left and right. Sharpening is one of the tools that you should use last; consider it the "finishing touch". It should also be used sparingly, it's one feature where the term "less is more" is especially important. Try it out and you'll see what I mean. Too much sharpening can make your photo look unnatural and "pixelated".

CHAPTER:5

LIST OF TOOLS ARE SHOWN AS DIAGRAM THAT USED IN PIMP

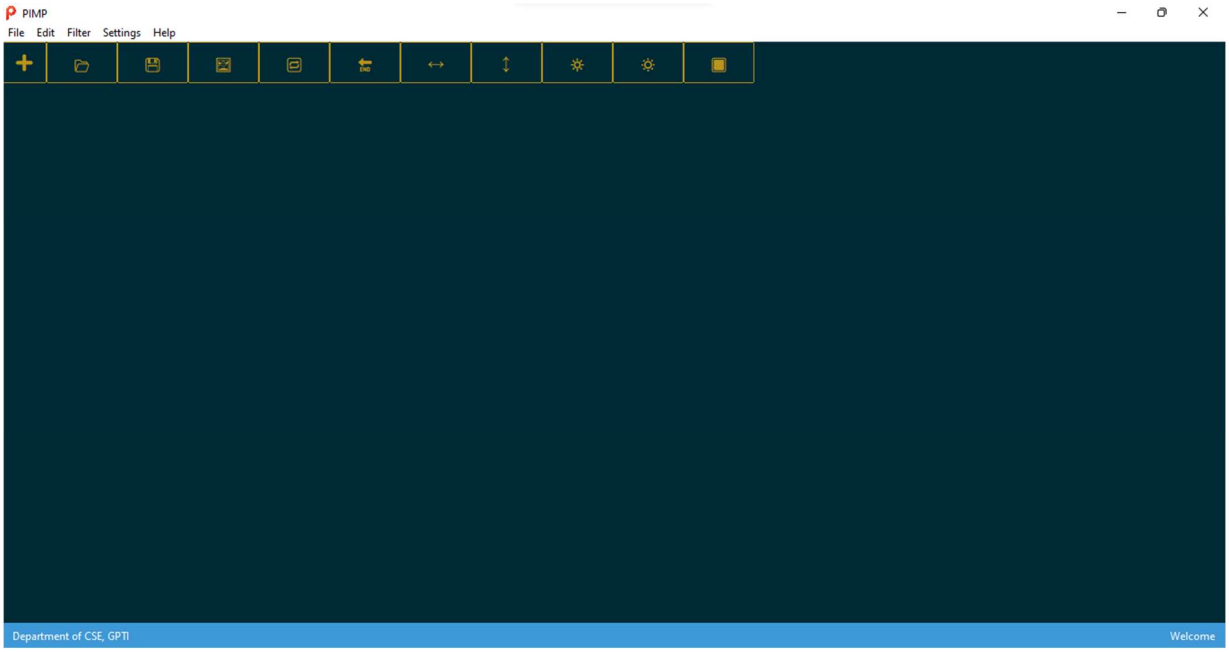





Filter
Enhance
Brightness
Contrast
Sharpness
Grey
Invert
Solarize
Box Blur
Gaussian Blur
Settings
Theme
Help
About
Shortcut Keys

OUTLINE:

Final Outline of our project



CHAPTER:6

SOURCE CODE:

Models folder
Model.py

```
import json
import settings as st
import datetime

def update_config_file(data, file_path = st.CONFIG_FILE):
    with open(file_path) as config_file:
        config = json.load(config_file)
        config.update(data)

    with open(file_path,'w') as config_file:
        json.dump(config, config_file, indent=4)


def update_last_login_date_time():
    now = datetime.datetime.now()
    data = {"LAST_LOGIN" : str(now)}
    update_config_file(data)


def update_last_image():
    pass
```

Views folder
Menu.py

```
import tkinter as tk
from tkinter import messagebox
from ttkbootstrap.themes import standard



def create_menu(root, command_list, theme_var):
    # create menu
    menu = tk.Menu(root)

    # ### File Menu ####
    file = tk.Menu(menu, tearoff=0)

    file.add_command(label='New',underline=0,
command=command_list['create_new_image'],accelerator="Ctrl+N")
    file.add_command(label='Open', command=command_list['open_image'],underline=0,
accelerator="Ctrl+O")
```

```
    file.add_command(label='Save', command=command_list['save_image'],underline=0,
accelerator="Ctrl+S")
    file.add_command(label='Quit', command=root.destroy,underline=0,
accelerator="Alt+F4")
    menu.add_cascade(label='File', menu=file,underline=0)

    # ### Edit Menu ####
    edit = tk.Menu(menu, tearoff=0)

    edit.add_command(label='Scale',
command=command_list['scale_image_frame'],underline=0)
    edit.add_command(label='Crop',
command=command_list['crop_image_frame'],underline=0)
    edit.add_command(label='Convert',underline=1)
    edit.add_command(label='Rotate', command=command_list['rotate_frame'],underline=0)
    menu.add_cascade(label='Edit', menu=edit,underline=0)

    # ### Filter Menu ####
    filter = tk.Menu(menu, tearoff=0)

    filter.add_command(label='Enhance Color',
command=command_list['color_enhance_frame'],underline=0)
    filter.add_command(label='Brightness',
command=command_list['brightness_enhance_frame'],underline=0)
    filter.add_command(label='Contrast',
command=command_list['contrast_enhance_frame'],underline=0)
    filter.add_command(label='Sharpness',
command=command_list['sharpness_enhance_frame'],underline=0)
    filter.add_command(label='Grey',
command=command_list['gray_image_frame'],underline=0)
    filter.add_command(label='Invert',command=command_list['invert_image_frame'],underli
ne=0)
    filter.add_command(label='Solarize',
command=command_list['solarize_frame'],underline=1)
    filter.add_command(label='Box Blur',
command=command_list['box_blur_frame'],underline=2)
    filter.add_command(label='Gaussian Blur',
command=command_list['gauss_blur_frame'],underline=1)
    menu.add_cascade(label='Filter', menu=filter,underline=1)

    # ### Settings Menu ####
    settings_menu = tk.Menu(menu, tearoff=0)

    theme_list = standard.STANDARD_THEMES
    menu.add_cascade(label='Settings', menu=settings_menu,underline=0)

    theme_menu = tk.Menu(settings_menu, tearoff=0)
    settings_menu.add_cascade(menu=theme_menu, label='Theme',underline=0)

    for theme in theme_list:
        theme_menu.add_radiobutton(
```

```
            label=theme,
            value=theme,
            variable=theme_var
        )

    #### Help Menu ####
    help_menu = tk.Menu(menu, tearoff=0)
    def show_about():
        data = '''
            1. 175CS19010 - Chaithra L
            2. 175CS19014 – Harish V
            3. 175CS19025 - Mokshaprada P
            4. 175CS19031 – Rudresh S
             5. 175cs19035 – Sudarshan S

            Guide:Mrs Reshma M

            HOD:Dr.Parameshwarappa.S

         GOVERNMENT POLYTECHNIC IMMADIHALLI
        '''
        messagebox.showinfo("About us !", data)

    help_menu.add_command(label='About', command=show_about,underline=0)
    help_menu.add_command(label='Shortcut Keys',
command=command_list['shortcut_key_page'],underline=0)
    menu.add_cascade(label='Help', menu=help_menu,underline=0)

    return menu
print("menu")
```

Views.py

```
from msilib import text
import pathlib
import trace
from PIL import Image, ImageTk, ImageOps, ImageEnhance, ImageFilter
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog
from tkinter import messagebox
from tkinter.constants import ANCHOR, CENTER, UNDERLINE
import ttkbootstrap as bttk
from ttkbootstrap.icons import Emoji
from ttkbootstrap.tooltip import ToolTip
from ttkbootstrap.scrolled import ScrolledFrame
from ttkbootstrap.dialogs.colorchooser import ColorChooserDialog
import settings as st
from .menu import create_menu
from models.model import update_config_file, update_last_login_date_time
```

```
update_last_login_date_time()

root = bttk.Window(
        title = st.MAIN_WIN_TITLE,
        iconphoto=st.LOGO,
        themename=st.BASE_THEME
        )

# root.wm_attributes("transparentcolor", 'grey')
theme_var = tk.StringVar(root, st.BASE_THEME)
status_var = tk.StringVar(root,"Welcome")
new_image_color_var = tk.StringVar(root, "#0000ff")
height_entry_var = tk.IntVar(root, "100")
width_entry_var = tk.IntVar(root, "100")
rotate_angle_var = tk.IntVar(root, "0")
color_enh_var = tk.DoubleVar(root, "1.0",)
contrast_enh_var = tk.DoubleVar(root, "1")
brightness_enh_var = tk.DoubleVar(root, "1")
sharpness_enh_var = tk.DoubleVar(root, "1")
solarize_enh_var = tk.IntVar(root, "100")
box_blur_enh_var = tk.IntVar(root, "0")
gaussian_blur_enh_var = tk.IntVar(root, "0")
top_left_x_var = tk.IntVar(root, "0")
top_left_y_var = tk.IntVar(root, "0")
bottom_right_x_var = tk.IntVar(root, "0")
bottom_right_y_var = tk.IntVar(root, "0")
scale_img_var = tk.DoubleVar(root, "1")

style = bttk.Style()

bttk.Style().configure('TButton', font="-size 14")

def set_theme(*args):
    theme = theme_var.get()
    theme_update = {'THEME': theme}
    update_config_file(theme_update)
    style.theme_use(theme)
    bttk.Style().configure('TButton', font="-size 14")


theme_var.trace('w', set_theme)


def display_image_1(image_path):
    img_frame_height = image_frame.winfo_height()
    img_frame_width = image_frame.winfo_width()

    with Image.open(image_path) as img:
        img_width = img.width
        img_height= img.height
```

```
        ratio = min((img_frame_width/img_width), (img_frame_height/img_height))
        if ratio < 1:
            img_scale = ImageOps.scale(img, ratio)
            img_scale.save(st.PREVIEW_IMAGE)
        else:
            img.save(st.PREVIEW_IMAGE)

    with Image.open(st.PREVIEW_IMAGE) as img:
        image_label.image = ImageTk.PhotoImage(img)
        image_label.config(image=image_label.image)


def display_image(image_path):
    img_frame_height = image_frame.winfo_height()
    img_frame_width = image_frame.winfo_width()

    with Image.open(image_path) as img:
        img_width = img.width
        img_height= img.height

        ratio = min((img_frame_width/img_width), (img_frame_height/img_height))
        if ratio < 1:
            img_scale = ImageOps.scale(img, ratio)
            image_label.image = ImageTk.PhotoImage(img_scale)
        else:
            image_label.image = ImageTk.PhotoImage(img)

        image_label.config(image=image_label.image)
        status_var.set(f"Image Width = {img_width} and height = {img_height}")

def shortcut_key_page():

    sk_page = tk.Toplevel(root, padx=35, pady=5)
    sk_page.title("Shortcut Keys")
    sk_page.geometry("500x300")
    sk_page.resizable(False, False)
    sk_page.grab_set()

    sk_page.columnconfigure(0, weight=1)
    sk_page.columnconfigure(1, weight=1)
    sk_page_label = bttk.Label(sk_page, text="Shortcut Keys", anchor=CENTER, font='bold
25')
    sk_page_label.grid(row=0, column=0, columnspan=2, sticky='nsew', padx=10,
pady=(50,10))

    bttk.Label(sk_page, text="Command", anchor=CENTER).grid(row=1, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Key Binding", anchor=CENTER).grid(row=1, column=1,
sticky='nsew')
    bttk.Separator(sk_page).grid(row=2, column=0, columnspan=2, sticky='nsew')
```

```
    bttk.Label(sk_page, text="Ctrl + O", anchor=CENTER).grid(row=3, column=0,
sticky='nsew', pady=(20,0))
    bttk.Label(sk_page, text="Open Image", anchor=CENTER).grid(row=3, column=1,
sticky='nsew', pady=(20,0))

    bttk.Label(sk_page, text="Ctrl + S", anchor=CENTER).grid(row=4, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Save Image", anchor=CENTER).grid(row=4, column=1,
sticky='nsew')

    bttk.Label(sk_page, text="Ctrl + I", anchor=CENTER).grid(row=5, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Insert Image", anchor=CENTER).grid(row=5, column=1,
sticky='nsew')

    bttk.Label(sk_page, text="Ctrl + r", anchor=CENTER).grid(row=6, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Rotate Right 90 deg", anchor=CENTER).grid(row=6,
column=1, sticky='nsew')

    bttk.Label(sk_page, text="Ctrl + Shift + R", anchor=CENTER).grid(row=7, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Rotate Left 90 deg", anchor=CENTER).grid(row=7, column=1,
sticky='nsew')

    bttk.Label(sk_page, text="Ctrl + h", anchor=CENTER).grid(row=8, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Flip Horizontal", anchor=CENTER).grid(row=8, column=1,
sticky='nsew')

    bttk.Label(sk_page, text="Ctrl + Shift + H", anchor=CENTER).grid(row=9, column=0,
sticky='nsew')
    bttk.Label(sk_page, text="Flip Vertical", anchor=CENTER).grid(row=9, column=1,
sticky='nsew')


def open_image(*args):
    filename = filedialog.askopenfilename(
        initialdir=st.INITIAL_DIR,
        title = "Open Image File",
        filetype=(
            ("Image files", ("*.jpg","*.jpeg","*.png")),
            ("JPEG files", "*.jpeg"),
            ("PNG files", "*.png"),
            ("JPG files", "*.jpg"),
            )
        )

    if filename:
        file_path = filename.rsplit('/',1)[0]
        st.INITIAL_DIR = file_path
```

```
        st.INITIAL_IMAGE = filename
        data = {
                "INITIAL_DIR":file_path,
                "INITIAL_IMAGE":filename,
                }
        update_config_file(data)

        with Image.open(filename) as img:
           img.save(st.LAST_IMAGE)

        display_image(st.LAST_IMAGE)
        status_var.set(f"Open file {filename}")


def choose_color():
    cd = ColorChooserDialog()
    cd.show()
    color = cd.result
    if color:
        new_image_color_var.set(color.hex)


def generate_image():
    try:
        height = int(height_entry_var.get())
        width = int(width_entry_var.get())
        color = new_image_color_var.get()
    except:
        messagebox.showerror("Error in Data", "Width, Height should be in numbers greater
than 10")
        return None
    if height < 10 or width<10:
        messagebox.showerror("Error in Data", "Width, Height should be in numbers greater
than 10")
        return None
    img1 = Image.new('RGBA',(width,height),color)

    img1.save(st.LAST_IMAGE)
    display_image(st.LAST_IMAGE)
    status_var.set(f"New Image Created with height= {height}, width={width} and color=
{color}")


def create_new_image(*args):
    global image_option_frame
    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
```

```
    image_option_frame.columnconfigure(1, weight=2)

    width_label = bttk.Label(image_option_frame,text="Width", bootstyle="inverse-info",
padding=5)
    width_label.grid(row=0, column=0, sticky='nsew')
    height_label = bttk.Label(image_option_frame,text="Height", bootstyle="inverse-info",
padding=5)
    height_label.grid(row=1, column=0, sticky='nsew')
    Color_label = bttk.Label(image_option_frame,text="Choose Color", bootstyle="inverse-
info", padding=5)
    Color_label.grid(row=2, column=0, sticky='nsew')

    width_entry = bttk.Entry(image_option_frame, textvariable=width_entry_var,
bootstyle='info')
    width_entry.grid(row=0, column=1, sticky='nsew')
    height_entry = bttk.Entry(image_option_frame, textvariable=height_entry_var,
bootstyle='info')
    height_entry.grid(row=1, column=1, sticky='nsew')

    color_btn = bttk.Button(image_option_frame, text="color", command=choose_color)
    color_btn.grid(row=2, column=1, sticky='nsew', padx=10, pady=10)
    generate_btn = bttk.Button(image_option_frame, text="Create",
command=generate_image)
    generate_btn.grid(row=3, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    status_var.set(f"Chosen New Image option")
    display_image(st.LAST_IMAGE)


def save_image():
    filename = filedialog.asksaveasfilename(
        defaultextension='.png',
        initialdir=st.INITIAL_DIR,
        filetype=(
            ("Image files", ("*.jpg","*.jpeg","*.png")),
            ("JPEG files", "*.jpeg"),
            ("PNG files", "*.png"),
            ("JPG files", "*.jpg"),
            ),
        title = "Save Image",
        )
    if filename:
        ext = filename.rsplit('.')[-1]
        if ext in ('jpg', 'jpeg', 'png'):
            with Image.open(st.LAST_IMAGE) as img:
                try:
                    img.save(filename)
                except:
                    img.convert('RGB').save(filename,"JPEG")
            status_var.set(f"Image Saved in {filename}")
```

```
        else:
            status_var.set("Error in Saving")
            messagebox.showerror("File Save", "Unable to Save file with the given file name.")



def rotate_image(*args):
    file = pathlib.Path(st.LAST_IMAGE)
    if file.exists():
        angle = rotate_angle_var.get()
        with Image.open(st.LAST_IMAGE) as img:
            preview_img = img.rotate(angle, expand=True)
            preview_img.save(st.PREVIEW_IMAGE)

        display_image(st.PREVIEW_IMAGE)
    else:
        messagebox.showerror("Broken Image", "Please open an image (Ctrl+O) or create a new
one (Ctrl+N)")



def save_preview():
    file = pathlib.Path(st.PREVIEW_IMAGE)
    if file.exists():
        with Image.open(file) as img:
            img.save(st.LAST_IMAGE)
        file.unlink()
    display_image(st.LAST_IMAGE)


def cancel_preview():
    file = pathlib.Path(st.PREVIEW_IMAGE)
    if file.exists():
        file.unlink()
    display_image(st.LAST_IMAGE)


def color_enhance_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_filter = ImageEnhance.Color(img)
        new_img = img_filter.enhance(color_enh_var.get())
        new_img.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def brightness_enhance_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_filter = ImageEnhance.Brightness(img)
        new_img = img_filter.enhance(brightness_enh_var.get())
        new_img.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)
```

```
def contrast_enhance_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_filter = ImageEnhance.Contrast(img)
        new_img = img_filter.enhance(contrast_enh_var.get())
        new_img.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def sharpness_enhance_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_filter = ImageEnhance.Sharpness(img)
        new_img = img_filter.enhance(sharpness_enh_var.get())
        new_img.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def solarize_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_solarize = ImageOps.solarize(img, threshold=solarize_enh_var.get())
        img_solarize.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def scale_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_scale = ImageOps.scale(img, scale_img_var.get())
        img_scale.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def gray_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_scale = ImageOps.grayscale(img)
        img_scale.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def invert_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_scale = ImageOps.invert(img)
        img_scale.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def box_blur_image():
    with Image.open(st.LAST_IMAGE) as img:
        img_blur = img.filter(filter=ImageFilter.BoxBlur(box_blur_enh_var.get()))
        img_blur.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def gauss_blur_image():
```

```
    with Image.open(st.LAST_IMAGE) as img:
        img_blur = img.filter(filter=ImageFilter.GaussianBlur(gaussian_blur_enh_var.get()))
        img_blur.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def crop_image():
    with Image.open(st.LAST_IMAGE) as img:
        x1 = top_left_x_var.get()
        y1 = top_left_y_var.get()
        x2 = bottom_right_x_var.get()
        y2 = bottom_right_y_var.get()
        img_crop = img.crop((x1,y1,x2,y2))
        img_crop.save(st.PREVIEW_IMAGE)
    display_image(st.PREVIEW_IMAGE)


def rotate_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    width_label = bttk.Label(image_option_frame,text="ROTATE",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    width_label.grid(row=0, column=0,columnspan=2, sticky='nsew')
    height_label = bttk.Label(image_option_frame,text="Rotate", bootstyle="inverse-info",
padding=5)
    height_label.grid(row=1, column=0, sticky='nsew')

    height_entry = bttk.Entry(image_option_frame, textvariable=rotate_angle_var,
bootstyle='info')
    height_entry.grid(row=1, column=1, sticky='nsew')

    preview_btn = bttk.Button(image_option_frame, text="Preview", command=rotate_image)
    preview_btn.grid(row=3, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=4, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=4, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Rotate Image Options")
```

```
def color_enhance_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Enhance Color",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Color", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=color_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=color_enh_var,from_ = 0.5, to =
2.0)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=color_enhance_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Color Enhance Options")


def brightness_enhance_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
```

```
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Brightness",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Brightness", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=brightness_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=brightness_enh_var, from_ = 0.5,
to = 2.0)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=brightness_enhance_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Brightness Options")


def contrast_enhance_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Contrast",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Contrast", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=contrast_enh_var,
bootstyle="inverse-info", padding=5)
```

```
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=contrast_enh_var, from_ = 0.5, to
= 2.0)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=contrast_enhance_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Contrast Options")


def sharpness_enhance_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Sharpness",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Sharpness", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=sharpness_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=sharpness_enh_var, from_ = 0.5, to
= 2.0)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=sharpness_enhance_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
```

```
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Sharpness Options")


def solarize_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Solarize Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Solarize ", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=solarize_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=solarize_enh_var, from_ = 1, to =
100)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=solarize_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Sharpness Options")


def box_blur_frame():
    global image_option_frame
```

```
    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Blur Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Blur ", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=box_blur_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=box_blur_enh_var, from_ = 0, to =
10)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=box_blur_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Box Blur Options")


def gauss_blur_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Gauss Blur Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
```

```
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Blur ", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')
    color_val = bttk.Label(image_option_frame,textvariable=gaussian_blur_enh_var,
bootstyle="inverse-info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=gaussian_blur_enh_var, from_ = 0,
to = 10)
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview",
command=gauss_blur_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Gaussian Blur Options")


def scale_image_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Scale Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')

    color_label = bttk.Label(image_option_frame,text="Scale ", bootstyle="inverse-info",
padding=5)
    color_label.grid(row=1, column=0, sticky='nsew')

 color_val = bttk.Label(image_option_frame,textvariable=scale_img_var, bootstyle="inverse-
info", padding=5)
    color_val.grid(row=1, column=1, sticky='nsew')
    color_scale = bttk.Scale(image_option_frame, variable=scale_img_var, from_ = 0.5, to =
2)
```

```
    color_scale.grid(row=2, column=0,columnspan=2, sticky='nsew', padx=10)

    preview_btn = bttk.Button(image_option_frame, text="Preview", command=scale_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Scale Image")


def gray_image_frame():
    global image_option_frame

    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Gray Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')


    preview_btn = bttk.Button(image_option_frame, text="Preview", command=gray_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Gray Image")



def invert_image_frame():
    global image_option_frame

    image_option_frame.grid_forget()
```

```
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Invert Image",
anchor=tk.CENTER,bootstyle="inverse-info", padding=10)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')


    preview_btn = bttk.Button(image_option_frame, text="Preview", command=invert_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)
    status_var.set(f"Gray Image")


def crop_image_frame(*args):
    with Image.open(st.LAST_IMAGE) as img:
        x,y = img.size
        bottom_right_x_var.set(x)
        bottom_right_y_var.set(y)

    global image_option_frame
    image_option_frame.grid_forget()
    image_option_frame.destroy()
    image_option_frame = bttk.Frame(image_frame, style='info', width=150)
    image_option_frame.grid(row=0, column=1, sticky='nsew')

    image_option_frame.columnconfigure(0, weight=1)
    image_option_frame.columnconfigure(1, weight=2)

    title_label = bttk.Label(image_option_frame,text="Crop Image",anchor=tk.CENTER,
bootstyle="inverse-info", padding=5)
    title_label.grid(row=0, column=0,columnspan=2, sticky='nsew')



  top_left_x_label = bttk.Label(image_option_frame,text="Top Left X", bootstyle="inverse-
info", padding=5)
    top_left_x_label.grid(row=1, column=0, sticky='nsew')
```

```
top_left_y_label = bttk.Label(image_option_frame,text="Top Left Y", bootstyle="inverse-
info", padding=5)
    top_left_y_label.grid(row=2, column=0, sticky='nsew')


    top_left_x = bttk.Entry(image_option_frame, textvariable=top_left_x_var, bootstyle='info')
    top_left_x.grid(row=1, column=1, sticky='nsew')
    top_left_y = bttk.Entry(image_option_frame, textvariable=top_left_y_var, bootstyle='info')
    top_left_y.grid(row=2, column=1, sticky='nsew')


    bottom_right_x_label = bttk.Label(image_option_frame,text="Bottom Right X",
bootstyle="inverse-info", padding=5)
    bottom_right_x_label.grid(row=3, column=0, sticky='nsew')
    bottom_right_y_label = bttk.Label(image_option_frame,text="Bottom Right Y",
bootstyle="inverse-info", padding=5)
    bottom_right_y_label.grid(row=4, column=0, sticky='nsew')


    bottom_right_x = bttk.Entry(image_option_frame, textvariable=bottom_right_x_var,
bootstyle='info')
    bottom_right_x.grid(row=3, column=1, sticky='nsew')
    bottom_right_y = bttk.Entry(image_option_frame, textvariable=bottom_right_y_var,
bootstyle='info')
    bottom_right_y.grid(row=4, column=1, sticky='nsew')

    preview_btn = bttk.Button(image_option_frame, text="Preview", command=crop_image)
    preview_btn.grid(row=10, column=0, columnspan=2, sticky='nsew', padx=20, pady=15)

    cancel_btn = bttk.Button(image_option_frame, text=Emoji.get("CROSS MARK"),
command=cancel_preview, bootstyle='danger')
    cancel_btn.grid(row=11, column=0, sticky='nsew', padx=20, pady=15)

    save_btn = bttk.Button(image_option_frame, text=Emoji.get('FLOPPY DISK'),
command=save_preview, bootstyle='success')
    save_btn.grid(row=11, column=1, sticky='nsew', padx=20, pady=15)
    display_image(st.LAST_IMAGE)



 status_var.set(f"Crop Image")
    display_image(st.LAST_IMAGE)

def rotate_90_right_image(*args):

    rotate_angle_var.set(90)
    rotate_image()

   save_preview()

def rotate_90_left_image(*args):
```

```
    rotate_angle_var.set(-90)
    rotate_image()
    save_preview()


def flip_image(direction):
    file = pathlib.Path(st.PREVIEW_IMAGE)
    if file.exists():
        with Image.open(file) as img:
            flip_img = img.transpose(direction)
        file.unlink()
    else:
        file = pathlib.Path(st.LAST_IMAGE)
        with Image.open(file) as img:
            flip_img = img.transpose(direction)

    flip_img.save(st.LAST_IMAGE)
    display_image(st.LAST_IMAGE)

def flip_horizontal_image(*args):
    flip_image(Image.FLIP_LEFT_RIGHT)

def flip_vertical_image(*args):
    flip_image(Image.FLIP_TOP_BOTTOM)

root.geometry(f'{int(root.winfo_screenwidth()-100)}x{int(root.winfo_screenheight()-100)}')



##### Main Frame #####


##### Buttons Frame #####
root.columnconfigure(0, weight=1)
top_button_frame = bttk.Frame(root)
top_button_frame.grid(row=0, column=0, sticky='nsew')

new_btn = bttk.Button(top_button_frame,
                      text=Emoji.get('HEAVY PLUS SIGN'),
                      style='primary-outline',
                      padding=10,
                      command=create_new_image
                      )
new_btn.pack(side=tk.LEFT)
ToolTip(new_btn, text="New Image")

open_btn = bttk.Button(top_button_frame,
                      text=Emoji.get('open file folder'),
                      style='primary-outline',
                      padding=10,
                      command=open_image,
                      width=5,
```

```
                )
open_btn.pack(side=tk.LEFT)
ToolTip(open_btn, text="Open Image")


save_btn = bttk.Button(top_button_frame,
                text=Emoji.get('FLOPPY DISK'),
                style='primary-outline',
                padding=10,
                command=save_image,
                width=5,
                )
save_btn.pack(side=tk.LEFT)
ToolTip(save_btn, text="Save Image")


scale_btn = bttk.Button(top_button_frame,
                text=Emoji.get('FRAME WITH PICTURE'),
                style='primary-outline',
                padding=10,
                command=scale_image_frame,
                width=5,
                )
scale_btn.pack(side=tk.LEFT)
ToolTip(scale_btn, text="Scale Image")


rotate_right_btn = bttk.Button(top_button_frame,
                text=Emoji.get('CLOCKWISE RIGHTWARDS AND LEFTWARDS OPEN
CIRCLE ARROWS'),
                style='primary-outline',
                padding=10,
                command=rotate_90_right_image,
                width=5,
                )
rotate_right_btn.pack(side=tk.LEFT)
ToolTip(rotate_right_btn, text="Rotate 90° Clockwise")


rotate_left_btn = bttk.Button(top_button_frame,
                text=Emoji.get('ANTICLOCKWISE DOWNWARDS AND UPWARDS OPEN
CIRCLE ARROWS'),
                style='primary-outline',

                 padding=10,
                command=rotate_90_left_image,
                width=5,
                )
rotate_left_btn.pack(side=tk.LEFT)
ToolTip(rotate_left_btn, text="Rotate 90° AntiClockWise")
```

```
flip_horiz_btn = bttk.Button(top_button_frame,
                text=Emoji.get('LEFT RIGHT ARROW'),
                style='primary-outline',
                padding=10,
                command=flip_horizontal_image,
                width=5,
                )
flip_horiz_btn.pack(side=tk.LEFT)
ToolTip(flip_horiz_btn, text="Flip Horizontal")


flip_vertical_btn = bttk.Button(top_button_frame,
                text=Emoji.get('UP DOWN ARROW'),
                style='primary-outline',
                padding=10,
                command=flip_vertical_image,
                width=5,
                )
flip_vertical_btn.pack(side=tk.LEFT)
ToolTip(flip_vertical_btn, text="Flip Vertical")


brightness_btn = bttk.Button(top_button_frame,
                text=Emoji.get('HIGH BRIGHTNESS SYMBOL'),
                style='primary-outline',
                padding=10,
                command=brightness_enhance_frame,
                width=5,
                )
brightness_btn.pack(side=tk.LEFT)
ToolTip(brightness_btn, text="Brightness")

contrast_btn = bttk.Button(top_button_frame,
                text=Emoji.get('LOW BRIGHTNESS SYMBOL'),
                style='primary-outline',
                padding=10,
                command=contrast_enhance_frame,
                width=5,
                )
contrast_btn.pack(side=tk.LEFT)
ToolTip(contrast_btn, text="Contrast")


crop_btn = bttk.Button(top_button_frame,
                text=Emoji.get('WHITE SQUARE BUTTON'),
                style='primary-outline',
                padding=10,
                command=crop_image_frame,
                width=5,
                )
crop_btn.pack(side=tk.LEFT)
```

```
ToolTip(crop_btn, text="Crop")



##### Image Frame #####
root.rowconfigure(1, weight=1)
root.columnconfigure(0, weight=1)

# image_frame = ScrolledFrame(root, bootstyle="danger", padding=10)
image_frame = bttk.Frame(root, padding=5)
image_frame.grid(row=1, column=0, sticky='nsew')
image_frame.update()

img_frame_height = image_frame.winfo_height()
img_frame_width = image_frame.winfo_width()

image_frame.rowconfigure(0, weight=1)
image_frame.columnconfigure(0, weight=1)
# image_frame.columnconfigure(1, weight=1)

try:
    img_temp = ImageTk.PhotoImage(Image.open(st.LAST_IMAGE))
    image_label = bttk.Label(image_frame, image=img_temp)
except:
    image_label = bttk.Label(image_frame)

image_label.grid(row=0, column=0, sticky='nsew')

popup_menu=tk.Menu(image_label,tearoff=0)
popup_menu.add_command(label="Open Image", command=open_image,
accelerator="Ctrl+o")
popup_menu.add_command(label="Save Image", command=save_image,
accelerator="Ctrl+S")
popup_menu.add_command(label="Rotate Right", command=rotate_90_right_image,
accelerator="Ctrl+R")
popup_menu.add_command(label="Rotate Left", command=rotate_90_right_image,
accelerator="Ctrl+Shift+R")
popup_menu.add_command(label="Flip Horizontal", command=flip_horizontal_image,
accelerator="Ctrl+h")
popup_menu.add_command(label="Flip Vertical", command=flip_vertical_image,
accelerator="Ctrl+Shift+H")

def image_right_click_menu(event):
    # print(event)
    try:
        popup_menu.tk_popup(event.x_root, event.y_root,0)
    except:
        popup_menu.grab_release()
```

```
image_label.bind("<Button-3>", image_right_click_menu)

root.bind_all("<Control-n>", create_new_image)
root.bind_all("<Control-o>", open_image)
root.bind_all("<Control-r>", rotate_90_right_image)
root.bind_all("<Control-R>", rotate_90_left_image)
root.bind_all("<Control-h>", flip_horizontal_image)
root.bind_all("<Control-H>", flip_vertical_image)

command_list = {
        'shortcut_key_page':shortcut_key_page,
        'open_image':open_image,
        'save_image':save_image,
        'rotate_frame':rotate_frame,
        'create_new_image':create_new_image,
        'color_enhance_frame':color_enhance_frame,
        'brightness_enhance_frame':brightness_enhance_frame,
        'contrast_enhance_frame':contrast_enhance_frame,
        'sharpness_enhance_frame':sharpness_enhance_frame,
        'solarize_frame':solarize_frame,
        'box_blur_frame':box_blur_frame,
        'gauss_blur_frame':gauss_blur_frame,
        'crop_image_frame':crop_image_frame,
        'scale_image_frame':scale_image_frame,
        'gray_image_frame':gray_image_frame,
        'invert_image_frame':invert_image_frame,
}

menu = create_menu(root,command_list, theme_var)
root.config(menu=menu)

print("view")
```

CHAPTER: 7

TESTING

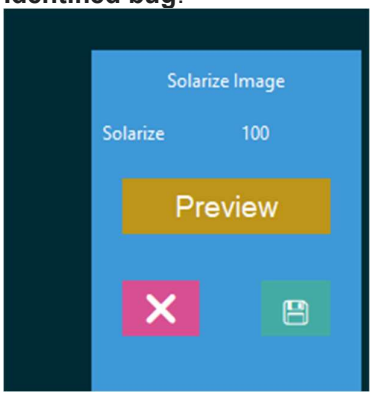
Software testing is a critical element of the ultimate review of specification design and coding .Testing software leads to uncovering of errors in the software functional and performance requirements are met .Testing also provides a good indication of software reliability and software quality as a whole.

Manual testing

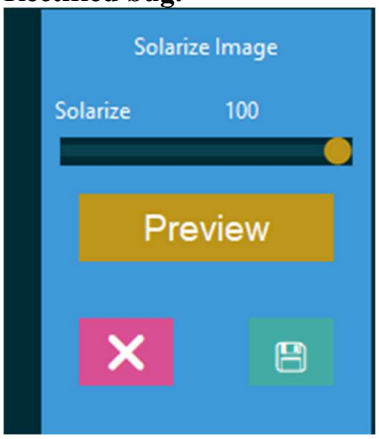
Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective. It requires a tester to play the role of an end user where by they use most of the application's features to ensure correct behaviour . through manual testing we rectified a bug and we corrected it.

The bug found in our project is when we click on solarize the option frame slide bar doesn't open/response we identified the bug and then we rectified it.

Identified bug.



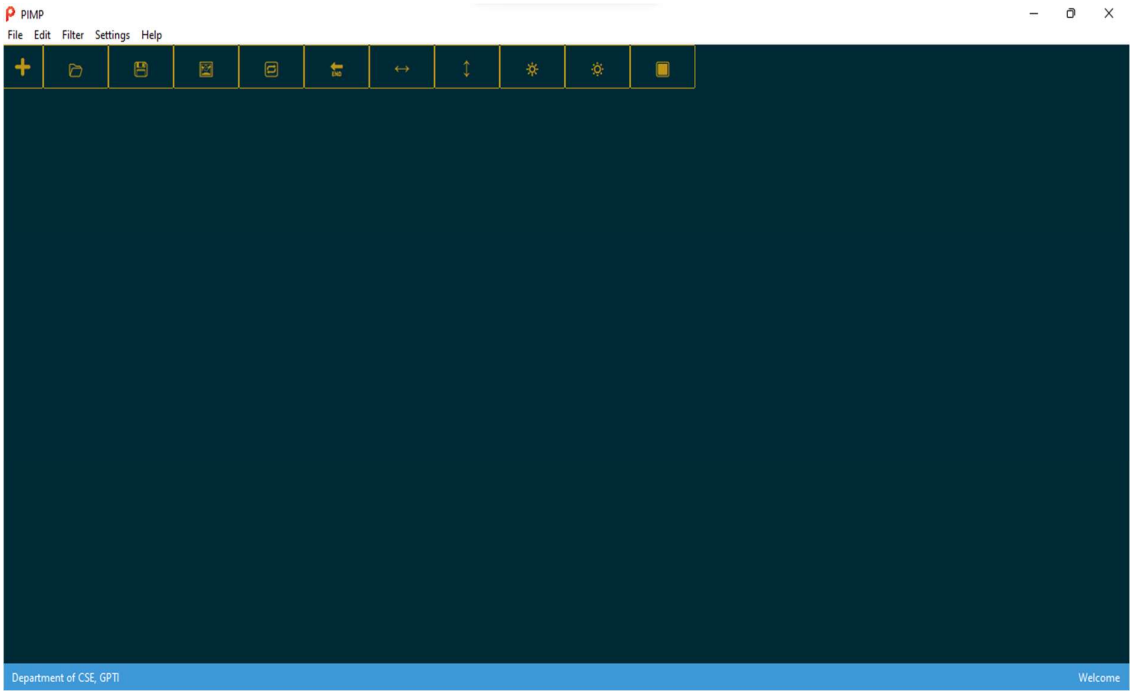
Rectified bug.



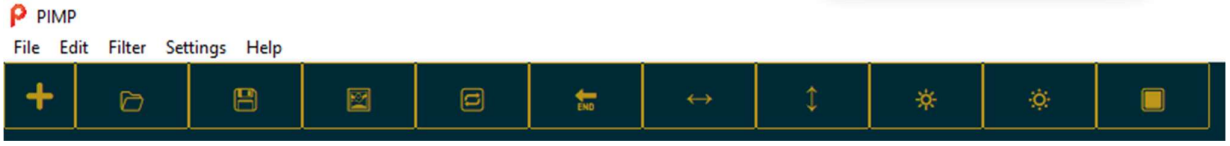
CHAPTER :8

OUTPUTS:

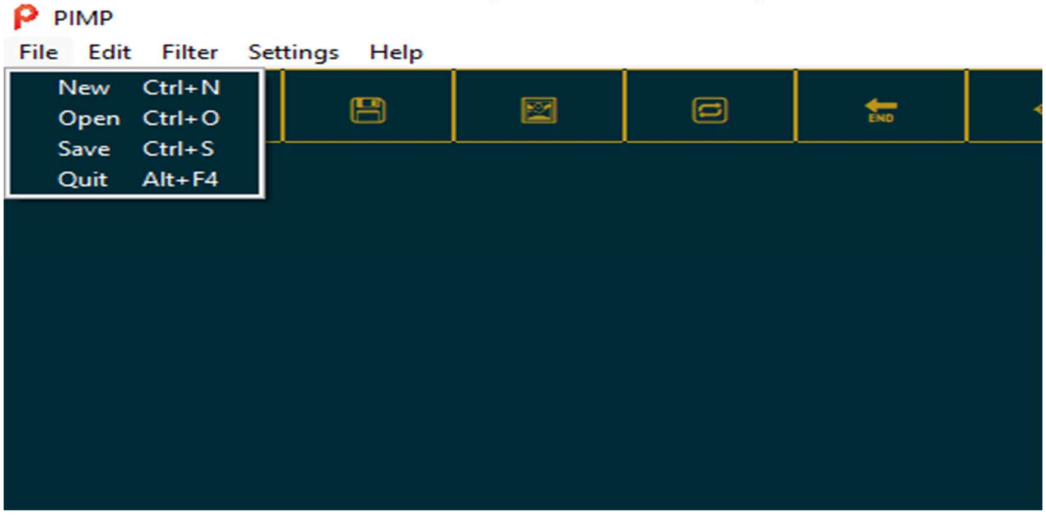
- Over all outline



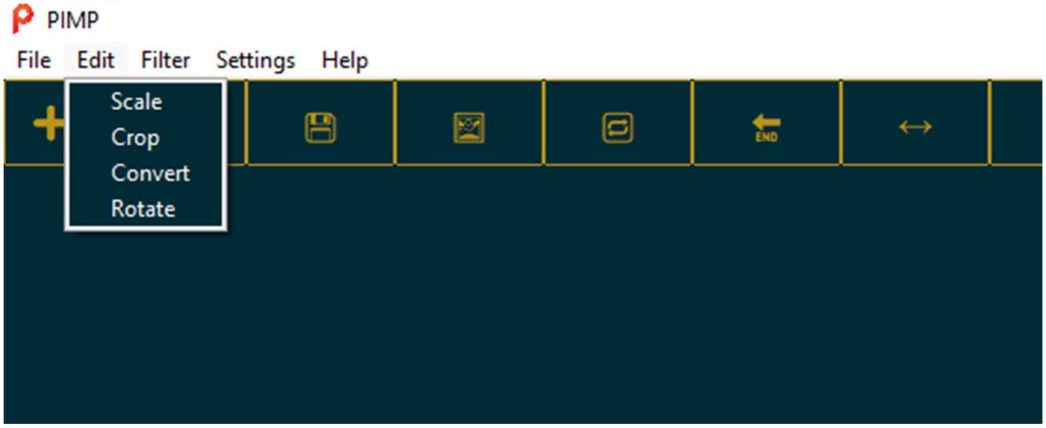
- Title bar, menu bar and icon bar



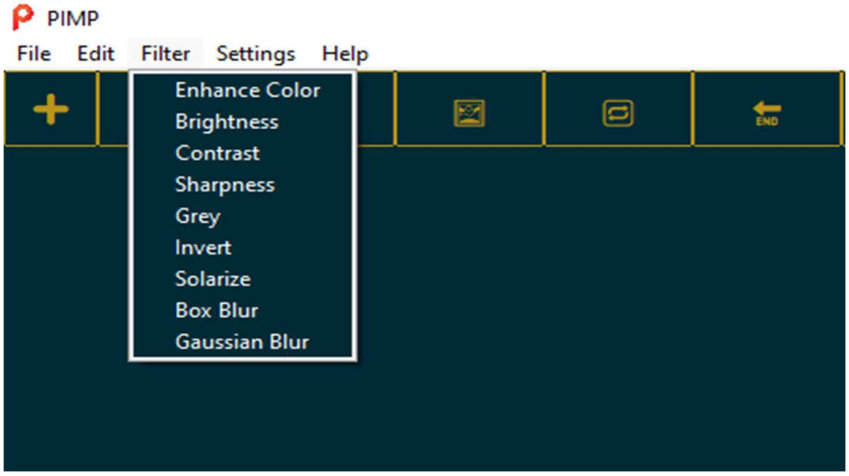
- Under the file menu there are New, Open ,Save and Quit options u can use shortcut keys.



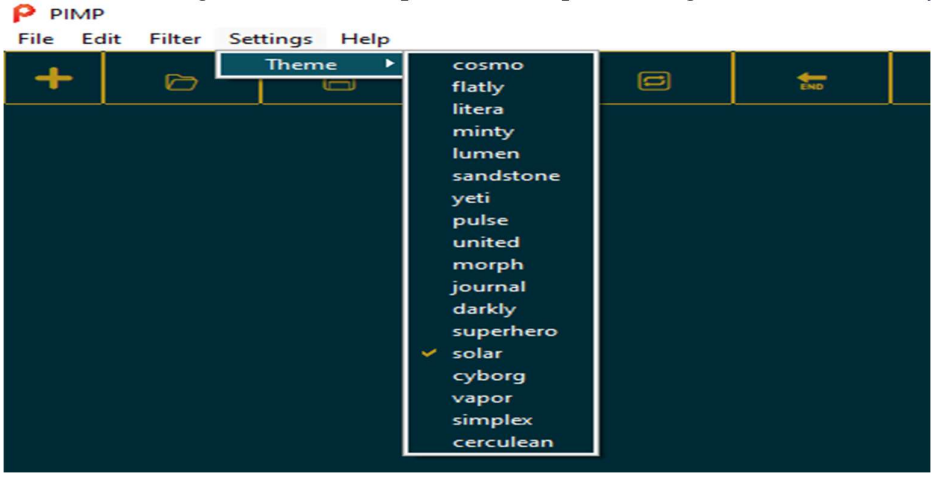
- Under the Edit menu there are Scale, Crop, Convert, Rotate options use this option on your selected photos.



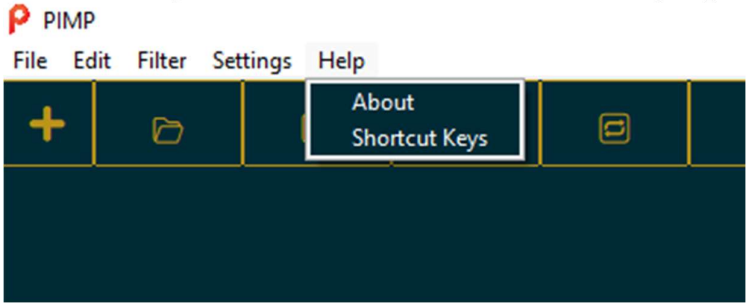
- Under the Filter menu there are Enhance color, Brightness, Contrast, Sharpness, Grey, Invert, Solarize, Box Blur, Gaussian Blur use the filters to fullfil your needs and enjoy.



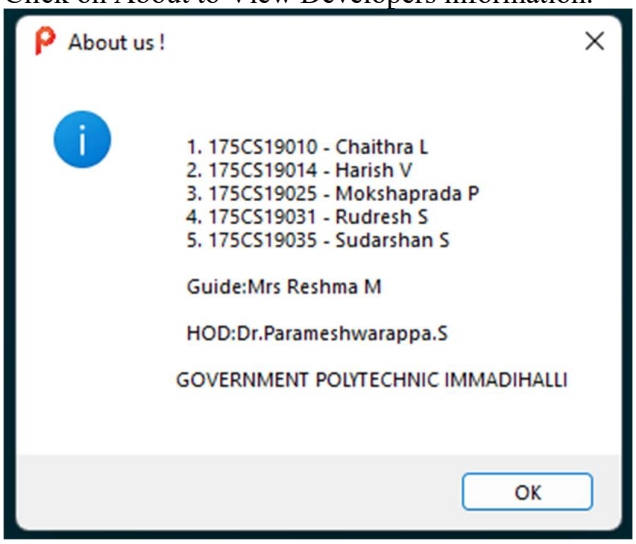
- Under the settings menu theme option will helps to change the theme which you want.



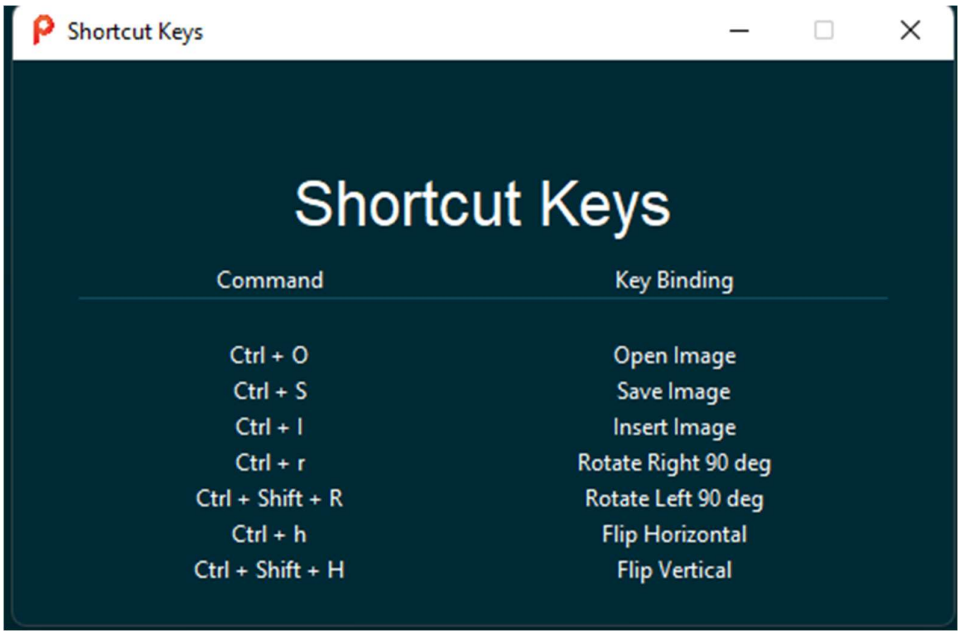
Under the Help menu there are About and Shortcut keys options



Click on About to View Developers information.

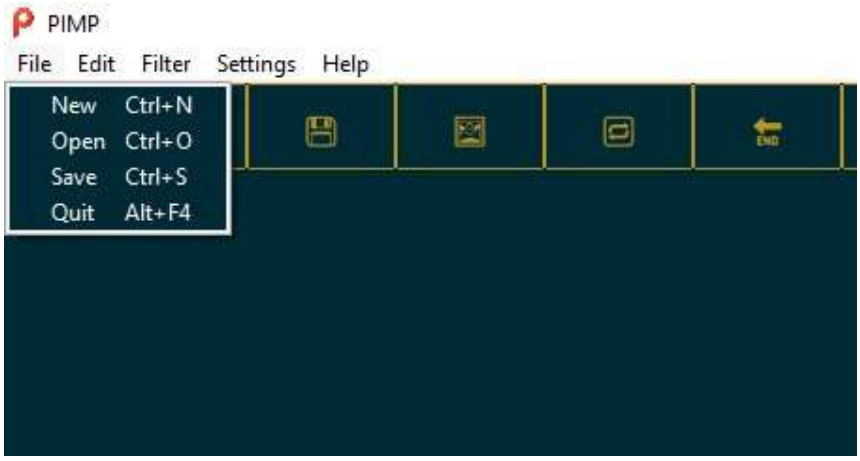


Click on Shortcut keys to know the shortcut keys in the application.

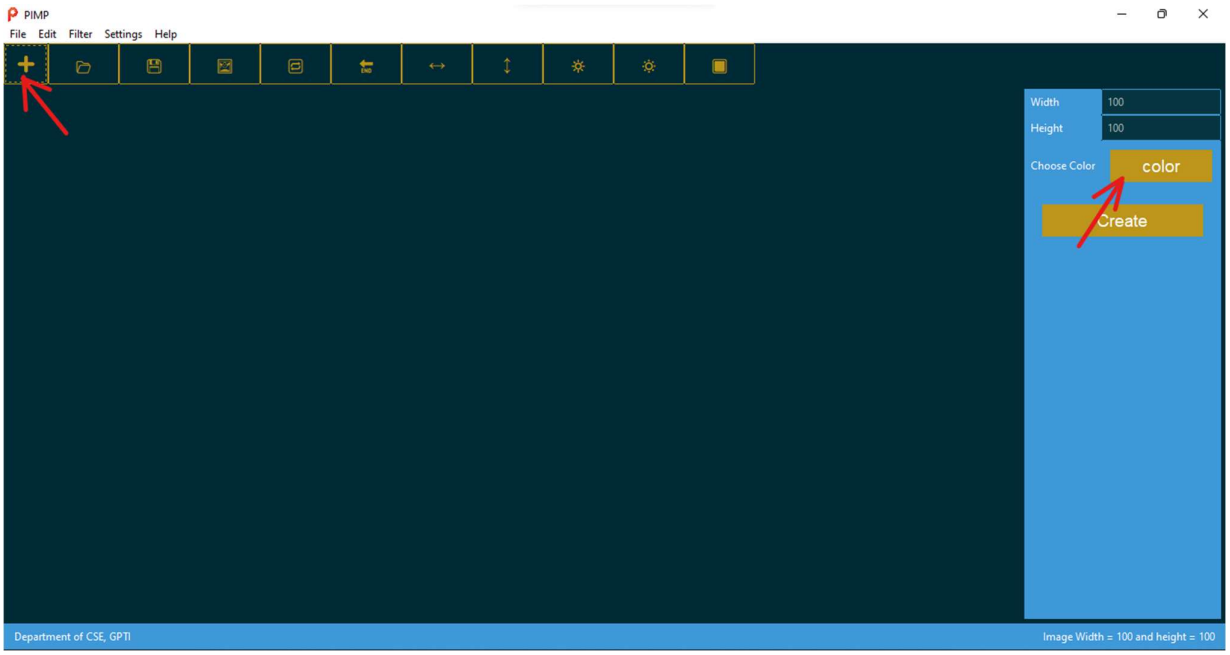


To Create new Image:

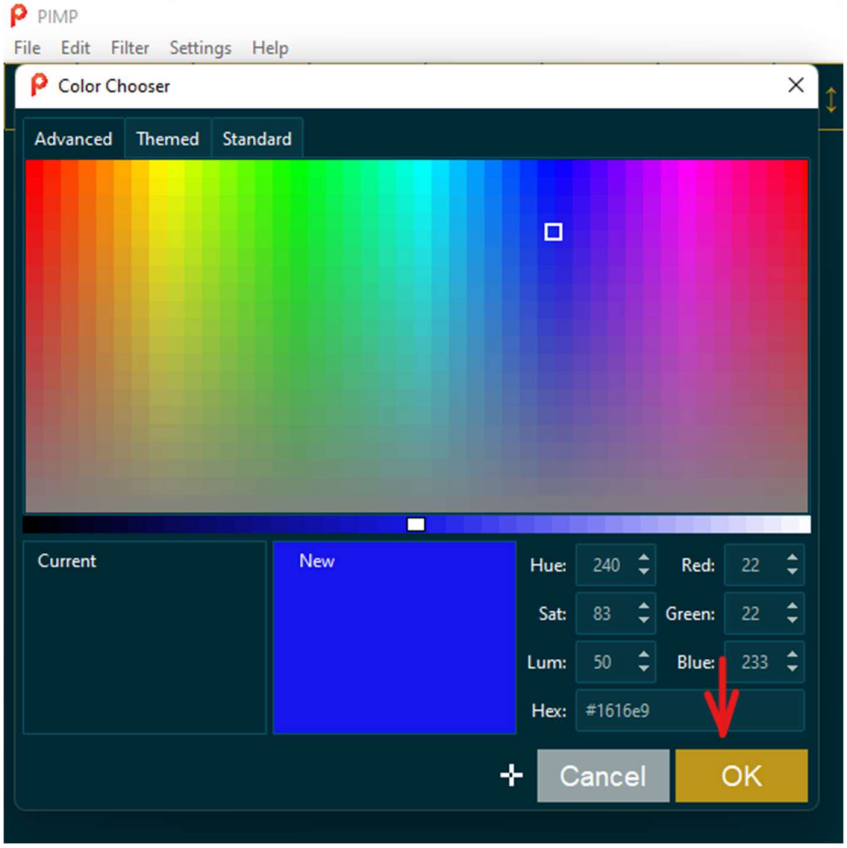
1.Click on file button on the menu bar and then click on new or Ctrl + N



2. Left click on New Image icon on the image frame and option frame will appear on right hand side then click on choose color.

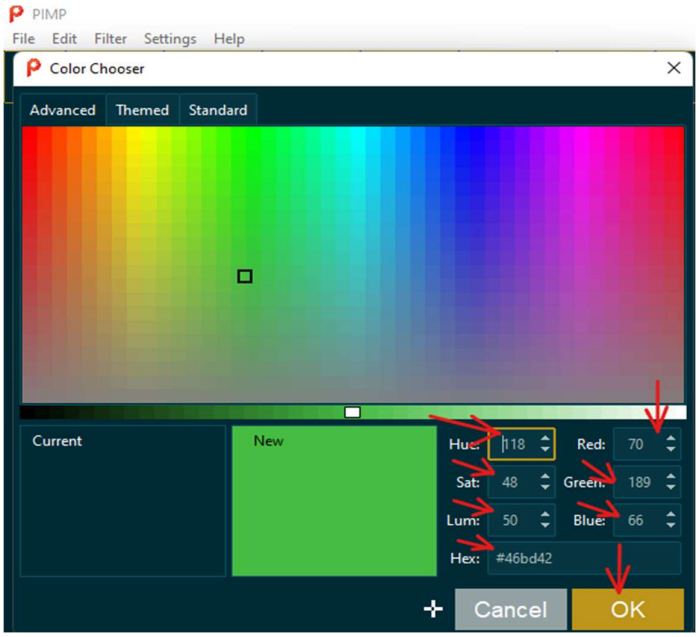


3. Now the color chooser window will appear you have to select color which you want to create, After selecting the color Left click on OK button.

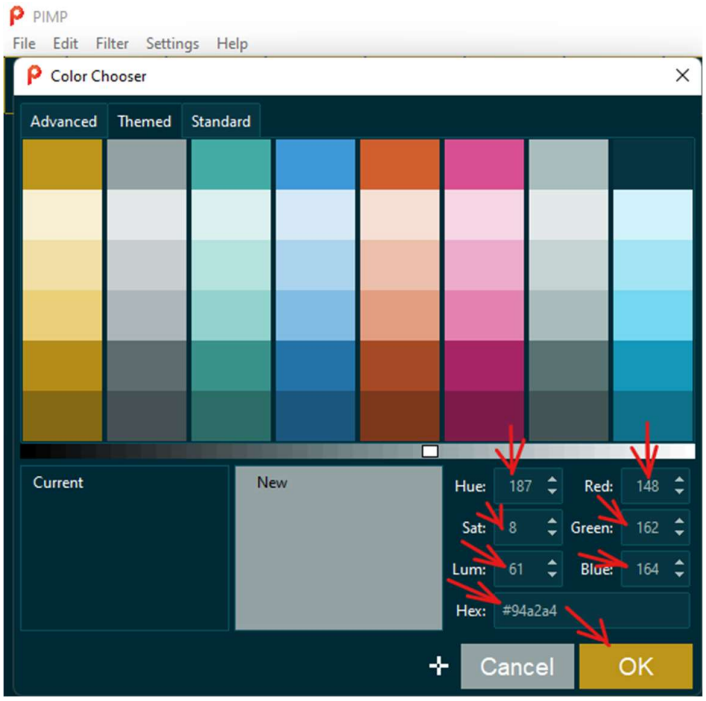


OR

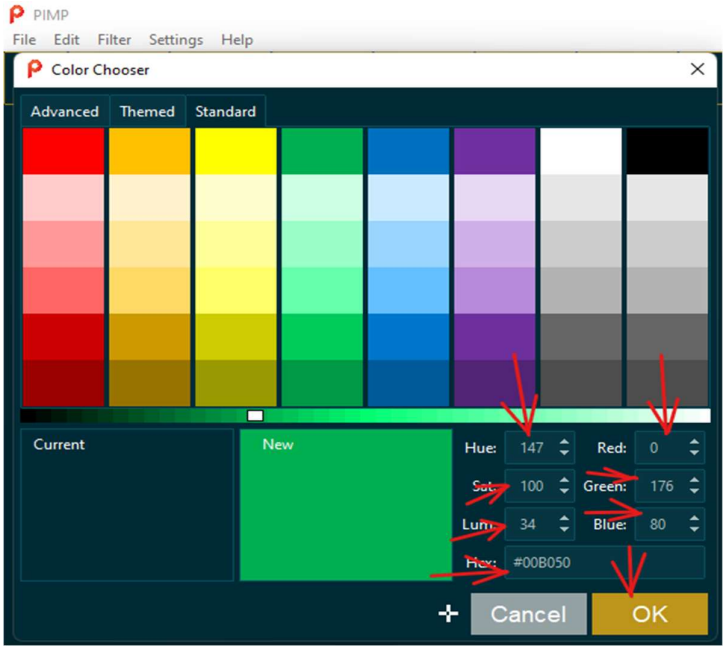
You can enter the values of the color in selected fields which you want and Then click on OK.



Left click on the theme, there you can select the color you want and You can enter the values of color and then click OK.

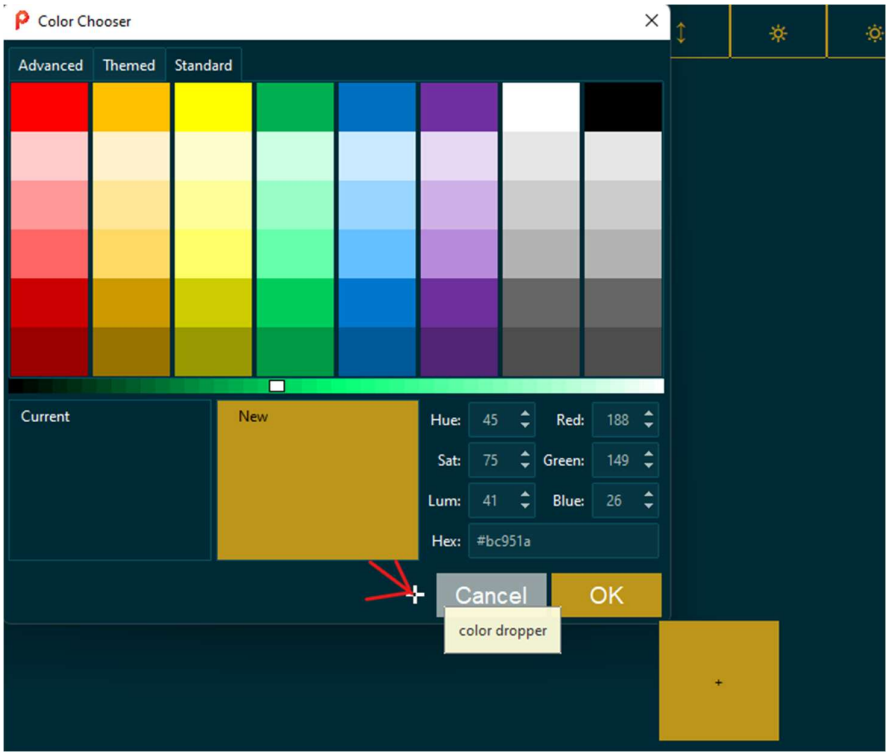


Left click on the Standard, there you can select the color you want and You can enter the values of color and then click OK.

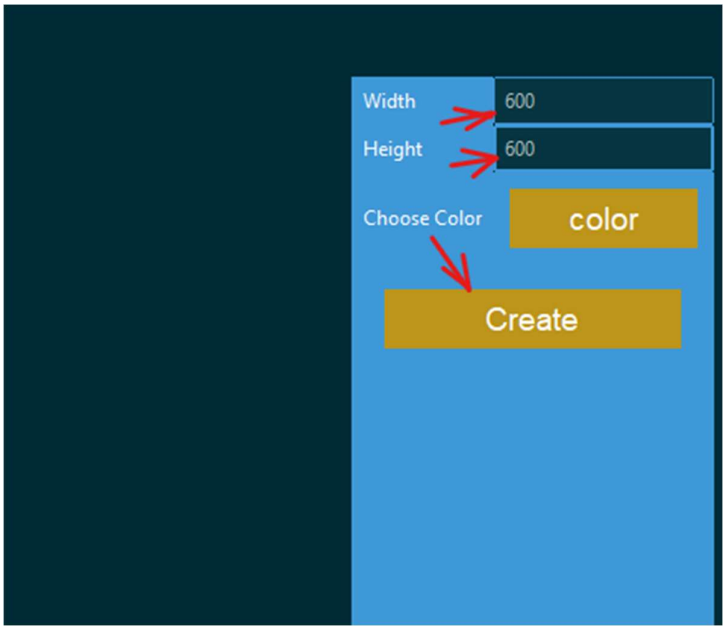


OR

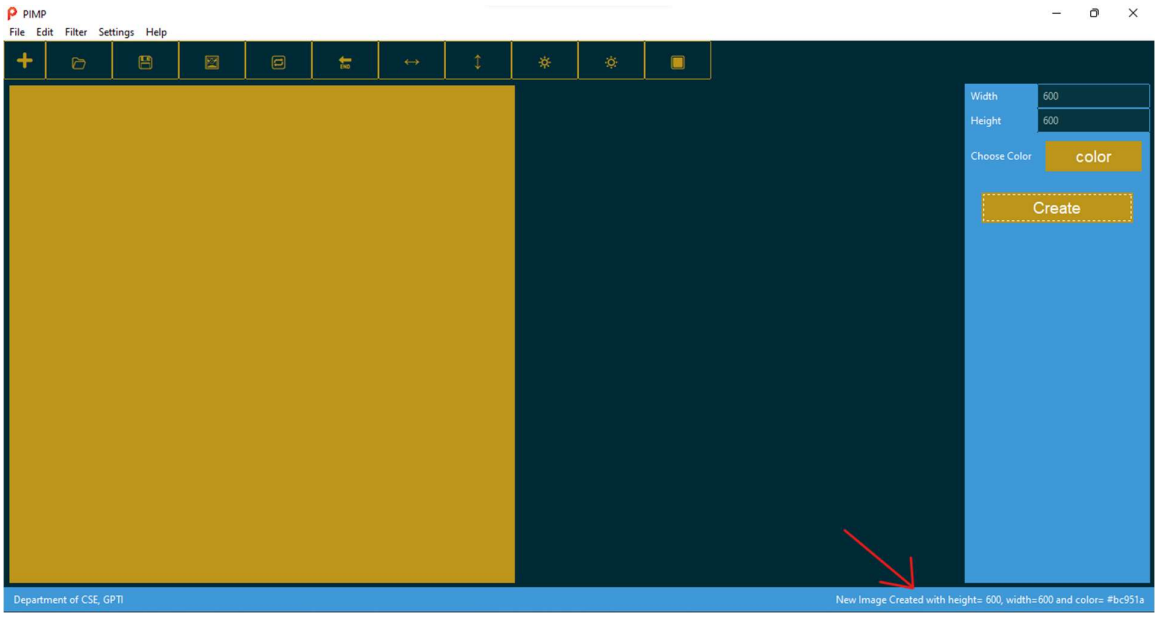
You can copy the copy on the screen using color dropper Left click on + symbol on color chooser window move the cursor on the screen which color you want stop there and Left click the color will automatically selected and then click on OK .



After selecting the color on option frame enter the width and height then click on Create.

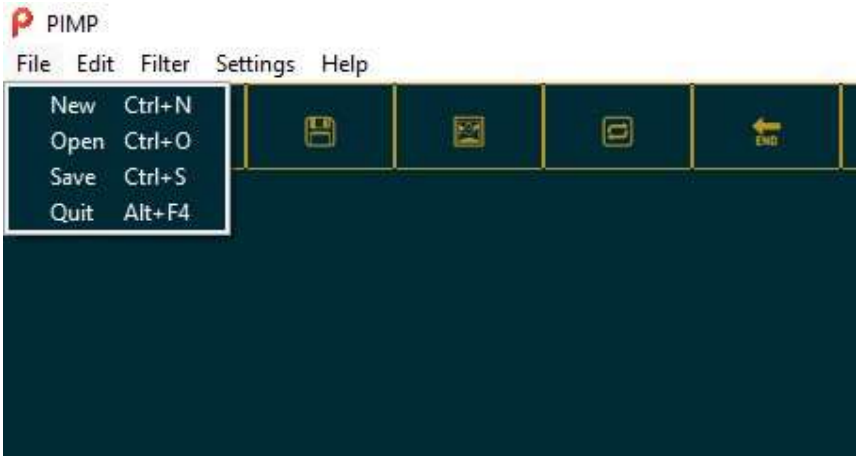


Now the new image will be created the height, width, color code will display on the status bar.



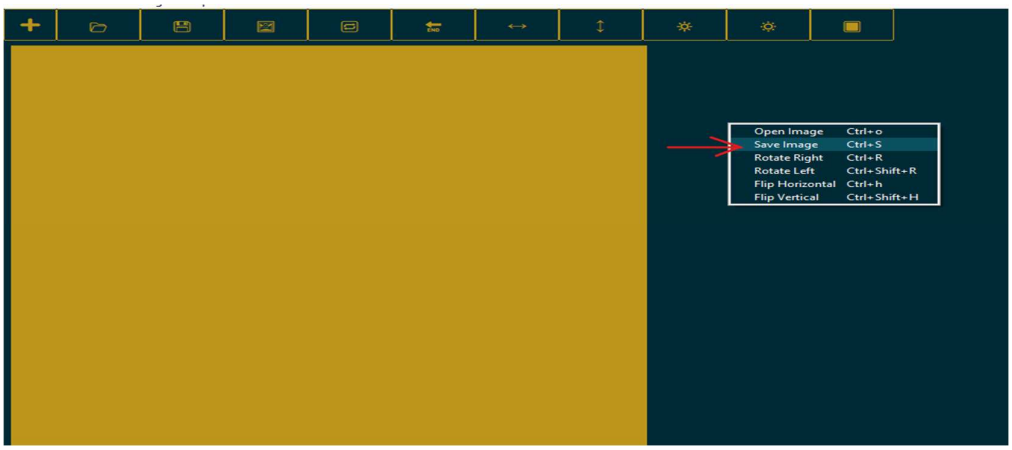
TO SAVE AN IMAGE:

1.Click on file button on the menu bar and then click on save or Ctrl + S



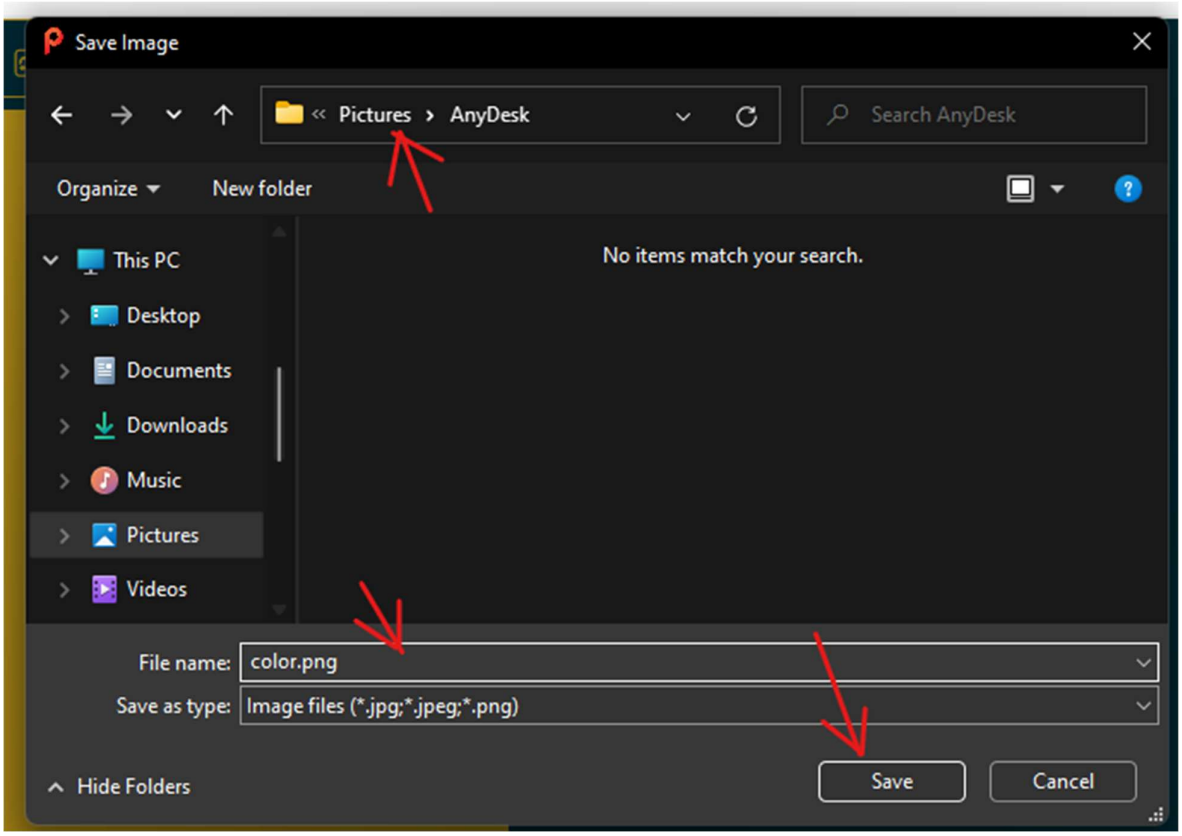
OR

Right click on the document window then click on save.



OR

Left click on Save Image icon on the image frame. Now the save Image window will appear there select the location where you want to save the Image and Give name to the image and select the type of the Image to save. Click on Save.



Created Image will be saved in the selected location.

CHAPTER 9

LIMITATIONS OF THE PROJECT

- This application only supports image type jpeg, jpg, png
- For cropping the image there is no Selection tool , enter values to crop the image
- There are Limited Filters to apply.
- There is no options to add text on the Image.
- There is no options to add border to the image.

CHAPTER:10

FUTURE ENHANCEMENT

- Selection bar can be added
- More Filters can be added
- More Image type can be added(RAW,ICO,TIFF..etc)
- Background Eraser can be added.
- Adding text on to the Image.
- Adding Borders to the Image.
- Adding Layers to the Image.

CHAPTER:11

REFERENCES

https://ttkbootstrap.readthedocs.io/en/latest/
https://github.com/skilldisk/Webinar-Automate-python-19.06.2020/tree/master/Image
https://docs.python.org/3/library/tkinter.html
https://pillow.readthedocs.io/en/stable/
https://docs.python.org/3/library/json.html#module-json
https://docs.python.org/3/library/pathlib.html#module-pathlib
https://www.gimp.org/about/
https://www.adobe.com/in/products/