Muhamad Agus S 212310004

Bagas Aji 212310009

Bagas Banu 212310026

Tugas Pengolahan citra (Color Image Processing)

Berikut adalah langkah-langkah untuk membuat aplikasi sederhana pengolahan Color image Processing menggunakan Python di Visual Studio Code (VSCode). Aplikasi ini akan memanfaatkan pustaka OpenCV untuk melakukan operasi mengubah warna dasar.

Langkah 1: Persiapan Lingkungan

1. Install Python (jika belum): Unduh dan instal Python dari [python.org](https://www.python.org/downloads/).

2. Buka VSCode: Pastikan Python telah terdeteksi di VSCode. Kalau belum, pilih interpreter Python yang benar.

3. Install Ekstensi Python: Buka Extensions di sidebar VSCode dan cari "Python", lalu instal ekstensi tersebut.

Langkah 2: Buat Folder Proyek

1. Buat folder baru untuk proyek ini (misalnya Test.py).

2. Di dalam VSCode, buka folder proyek tersebut dengan *File > Open Folder*.

Langkah 3:

1. Buka *Terminal* di VSCode (dengan menekan Ctrl + Shift + P, lalu ketik Terminal: Create New Integrated Terminal).
2. Instal pip install opencv-python pillow
3. Input Code
4. Jalankan python3 namafile.py

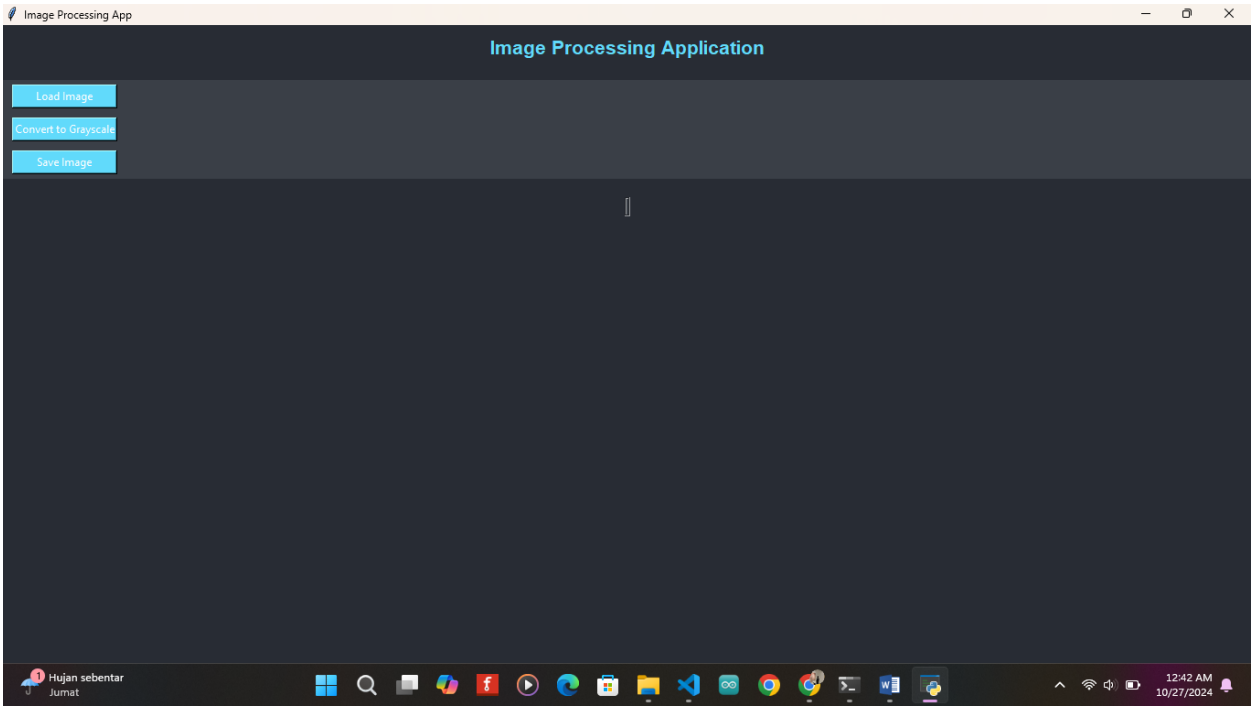
CODE

```
import tkinter as tk
from tkinter import filedialog, messagebox
from tkinter import Frame, Button, Label
import cv2
from PIL import Image, ImageTk
import numpy as np

class ImageProcessingApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Image Processing App")
        self.root.configure(bg="#282c34")
```

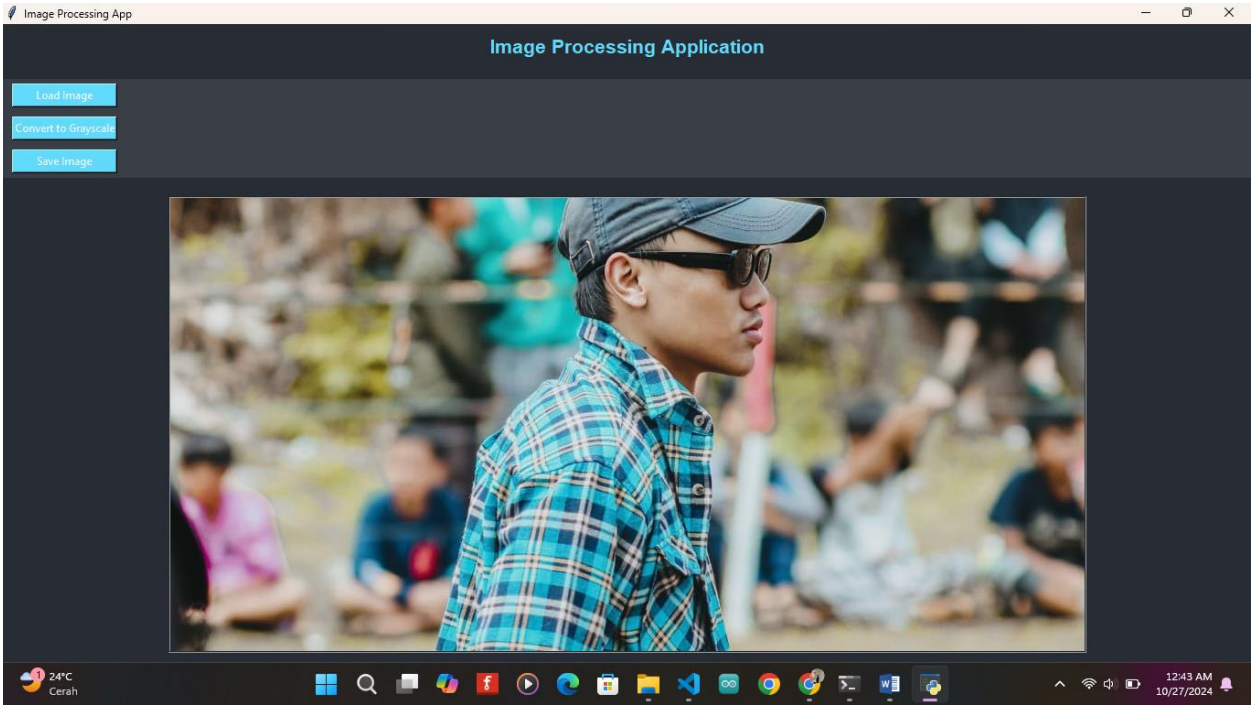
```
        self.image = None  # Gambar asli
        self.processed_image = None  # Gambar yang telah diproses
        self.tk_image = None

        # Judul aplikasi
        self.title_label = Label(root, text="Image Processing Application",
font=("Arial", 16, "bold"), fg="#61dafb", bg="#282c34")
        self.title_label.pack(pady=10)

        # Frame untuk tombol dengan background berbeda
        self.button_frame = Frame(root, bg="#3a3f47")
        self.button_frame.pack(pady=10, fill="x")

        # Tombol untuk memuat gambar
        self.load_button = Button(self.button_frame, text="Load Image",
command=self.load_image, bg="#61dafb", fg="white", width=15)
        self.load_button.grid(row=0, column=0, padx=10, pady=5)

        # Tombol untuk konversi grayscale
        self.gray_button = Button(self.button_frame, text="Convert to
Grayscale", command=self.convert_to_grayscale, bg="#61dafb", fg="white",
width=15)
        self.gray_button.grid(row=1, column=0, padx=10, pady=5)

        # Tombol untuk menyimpan gambar
        self.save_button = Button(self.button_frame, text="Save Image",
command=self.save_image, bg="#61dafb", fg="white", width=15)
        self.save_button.grid(row=2, column=0, padx=10, pady=5)

        # Label untuk menampilkan gambar
        self.image_label = Label(root, bg="#3a3f47", borderwidth=2,
relief="groove")
        self.image_label.pack(pady=10)

    def load_image(self):
        file_path = filedialog.askopenfilename(filetypes=[("Image files",
"*.jpg *.jpeg *.png")])
        if not file_path:
            return
        self.image = cv2.imread(file_path)
        self.processed_image = None  # Reset gambar yang telah diproses
        self.display_image(self.image)

    def display_image(self, img):
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
        pil_image = Image.fromarray(img_rgb)
        self.tk_image = ImageTk.PhotoImage(pil_image)

        # Update label gambar
        self.image_label.config(image=self.tk_image)
        self.image_label.image = self.tk_image  # Keep reference

    def convert_to_grayscale(self):
        if self.image is not None:
            self.processed_image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
            self.display_image(cv2.cvtColor(self.processed_image,
cv2.COLOR_GRAY2RGB))
        else:
            messagebox.showerror("Error", "Please load an image first.")

    def save_image(self):
        if self.processed_image is not None:
            file_path = filedialog.asksaveasfilename(defaultextension=".jpg",
                                                     filetypes=[("JPEG
files", "*.jpg"),
                                                                ("PNG files",
"*.png")])
            if file_path:
                # Simpan gambar yang telah diproses
                cv2.imwrite(file_path, self.processed_image)
                messagebox.showinfo("Success", "Image saved successfully!")
        else:
            messagebox.showerror("Error", "No processed image to save. Please
convert an image first.")

if __name__ == "__main__":
    root = tk.Tk()
    app = ImageProcessingApp(root)
    root.mainloop()
```

Output



Sebelum di convert



Setelah di ubah