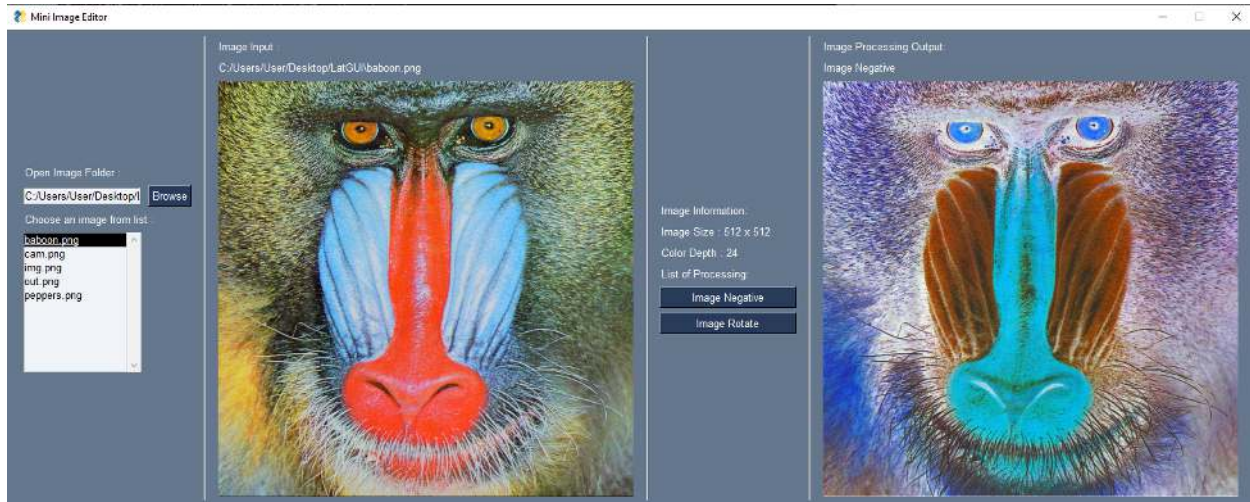


MODUL PRAKTIKUM

Pengolahan Citra Digital

Mini Image Editor dengan PySimpleGUI



Jurusan/Program Studi : Pendidikan Teknik Informatika
Mata Kuliah : Pengolahan Citra Digital
Kode : TIK19504
Semester : 5 / Ganjil
SKS : 3 SKS
Prasyarat : Struktur Data
Dosen Pengampu : Made Windu Antara Kesiman, S.T., M.Sc., Ph.D
Dr. I Made Gede Sunarya, S.Kom., M.Cs
I Made Dendi Maysanjaya, S.Pd., M.Eng



Universitas Pendidikan Ganesha

September 2021

Langkah Persiapan :

1. Install Python
2. Install package PySimpleGUI
3. Install package Pillow

Saran : Bisa menggunakan *open source package management system and environment management system Conda* : <https://docs.conda.io/en/latest/>

Langkah Pengembangan Program :

Langkah 1 : Mendesain layout interface

Buatlah file `img_viewer.py`, dengan code seperti berikut ini :

```
import PySimpleGUI as sg
import os.path

# Kolom Area No 1: Area open folder and select image
file_list_column = [
    [
        sg.Text("Open Image Folder :"),
    ],
    [
        sg.In(size=(20, 1), enable_events=True, key="ImgFolder"),
        sg.FolderBrowse(),
    ],
    [
        sg.Text("Choose an image from list :"),
    ],
    [
        sg.Listbox(
            values=[], enable_events=True, size=(18, 10), key="ImgList"
        )
    ],
]

# Kolom Area No 2: Area viewer image input
image_viewer_column = [
    [sg.Text("Image Input :")],
    [sg.Text(size=(40, 1), key="FilepathImgInput")],
    [sg.Image(key="ImgInputViewer")],
]
```

```

]

# Kolom Area No 3: Area Image info dan Tombol list of processing
list_processing = [
    [
        sg.Text("Image Information:"),
    ],
    [
        sg.Text(size=(20, 1), key="ImgSize"),
    ],
    [
        sg.Text(size=(20, 1), key="ImgColorDepth"),
    ],
    [
        sg.Text("List of Processing:"),
    ],
    [
        sg.Button("Image Negative", size=(20, 1), key="ImgNegative"),
    ],
    [
        sg.Button("Image Rotate", size=(20, 1), key="ImgRotate"),
    ],
]

# Kolom Area No 4: Area viewer image output
image_viewer_column2 = [
    [sg.Text("Image Processing Output: ")],
    [sg.Text(size=(40, 1), key="ImgProcessingType")],
    [sg.Image(key="ImgOutputViewer")],
]

# Gabung Full layout
layout = [
    [
        sg.Column(file_list_column),
        sg.VSeparator(),
        sg.Column(image_viewer_column),
        sg.VSeparator(),
        sg.Column(list_processing),
        sg.VSeparator(),
        sg.Column(image_viewer_column2),
    ],
]

window = sg.Window("Mini Image Editor", layout)

# Run the Event Loop

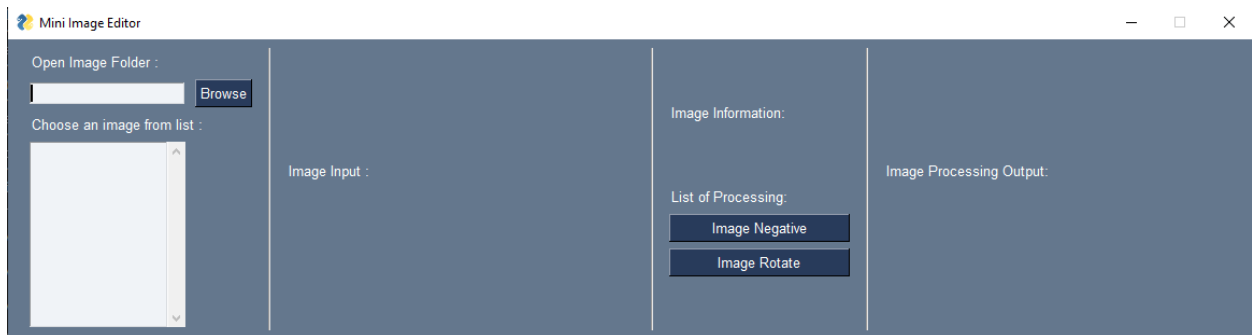
```

```
while True:
    event, values = window.read()

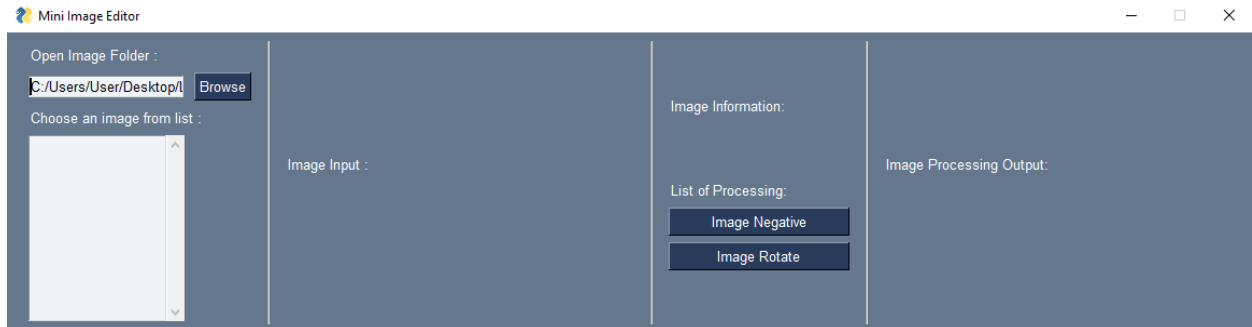
    if event == "Exit" or event == sg.WIN_CLOSED:
        break

window.close()
```

Hasil eksekusi :



Jika tombol Browse diklik, maka kita akan dapat memilih Folder, dan path folder tersebut akan muncul di text box image folder :



Namun, list file yang ada di dalam folder tersebut belum ditampilkan.

Langkah 2 : Menampilkan list file citra pada folder yang dipilih

Lengkapi bagian code Event Loop dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
# Run the Event Loop
while True:
    event, values = window.read()

    if event == "Exit" or event == sg.WIN_CLOSED:
        break

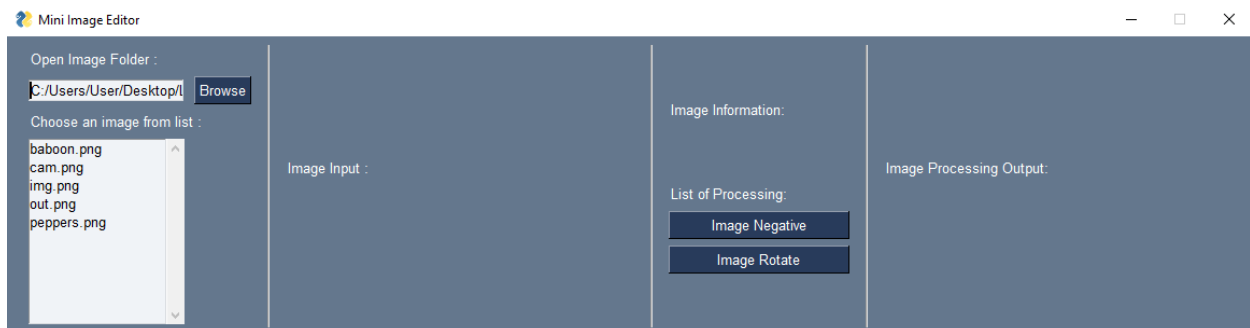
    # Folder name was filled in, make a list of files in the folder
    if event == "ImgFolder":
        folder = values["ImgFolder"]

        try:
            # Get list of files in folder
            file_list = os.listdir(folder)
        except:
            file_list = []

        fnames = [
            f
            for f in file_list
            if os.path.isfile(os.path.join(folder, f))
            and f.lower().endswith((".png", ".gif"))
        ]

        window["ImgList"].update(fnames)
```

Hasil eksekusi :



Sekarang, setelah memilih folder, list file citra yang ada di dalam folder tersebut akan ditampilkan di text box image list. Namun, ketika salah satu file citra tersebut dipilih (diklik), belum terjadi aksi tertentu.

Langkah 3 : Menampilkan citra input dan informasi ukuran dan color depth dari citra tersebut

Lengkapi bagian import package dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
import PySimpleGUI as sg
import os.path
from PIL import Image, ImageOps
```

Kemudian, lengkapi bagian code Event Loop dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
# Run the Event Loop
while True:
    event, values = window.read()

    if event == "Exit" or event == sg.WIN_CLOSED:
        break

    # Folder name was filled in, make a list of files in the folder
    if event == "ImgFolder":
        folder = values["ImgFolder"]

        try:
            # Get list of files in folder
            file_list = os.listdir(folder)
        except:
            file_list = []

        fnames = [
            f
            for f in file_list
            if os.path.isfile(os.path.join(folder, f))
            and f.lower().endswith((".png", ".gif"))
        ]

        window["ImgList"].update(fnames)

    elif event == "ImgList": # A file was chosen from the listbox

        try:
            filename = os.path.join(
                values["ImgFolder"], values["ImgList"][0]
            )
```

```

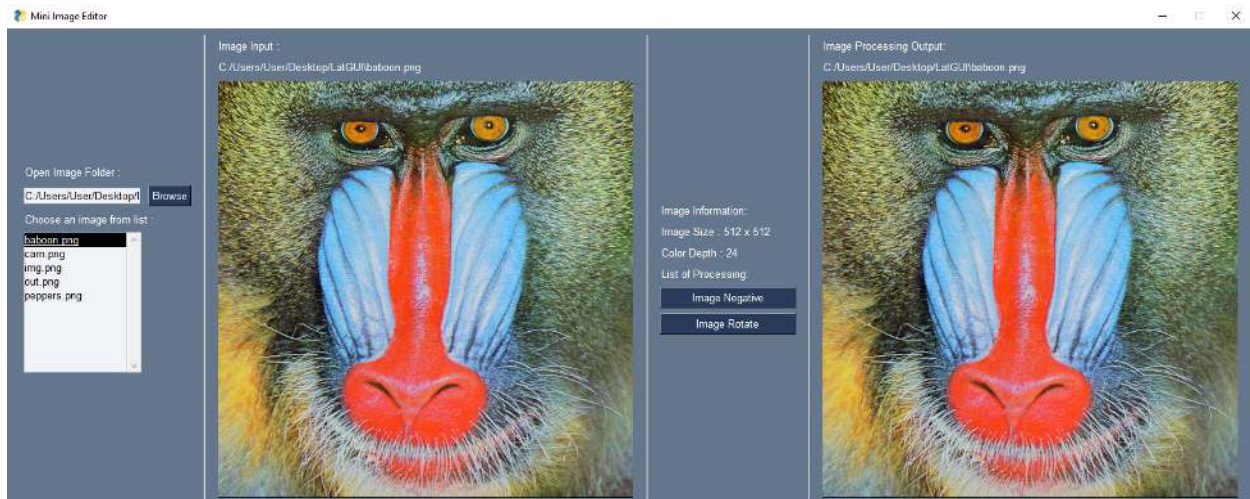
window["FilepathImgInput"].update(filename)
window["ImgInputViewer"].update(filename=filename)
window["ImgProcessingType"].update(filename)
window["ImgOutputViewer"].update(filename=filename)
img_input = Image.open(filename)
#img_input.show()

#Size
img_width, img_height = img_input.size
window["ImgSize"].update("Image Size : "+str(img_width)+" x "+str(img_height))

#Color depth
mode_to_coldepth = {"1": 1, "L": 8, "P": 8, "RGB": 24, "RGBA": 32, "CMYK": 32, "YCbCr": 24, "LAB":
24, "HSV": 24, "I": 32, "F": 32}
coldepth = mode_to_coldepth[img_input.mode]
window["ImgColorDepth"].update("Color Depth : "+str(coldepth))
except:
pass

```

Hasil eksekusi :



Ketika sebuah file citra dipilih, maka citra tersebut akan ditampilkan baik pada area Image Input maupun pada area Image Processing Output. Disamping itu, di area Image Information akan ditampilkan informasi ukuran dan color depth dari citra tersebut.

Langkah 4 : Mengimplementasikan algoritma Image Negative

Buatlah file baru bernama `processing_list.py` dengan sebuah fungsi `ImageNegative` sebagai berikut :

```
from PIL import Image, ImageOps

def ImgNegative(img_input,coldepth):
    #solusi 1
    #img_output=ImageOps.invert(img_input)

    #solusi 2
    if coldepth!=24:
        img_input = img_input.convert('RGB')

    img_output = Image.new('RGB',(img_input.size[0],img_input.size[1]))
    pixels = img_output.load()
    for i in range(img_output.size[0]):
        for j in range(img_output.size[1]):
            r, g, b = img_input.getpixel((i, j))
            pixels[i,j] = (255-r, 255-g, 255-b)

    if coldepth==1:
        img_output = img_output.convert("1")
    elif coldepth==8:
        img_output = img_output.convert("L")
    else:
        img_output = img_output.convert("RGB")

    return img_output
```

Lengkapi bagian import package dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
import PySimpleGUI as sg
import os.path
from PIL import Image, ImageOps
from processing_list import *
```

Kemudian, lengkapi bagian code Event Loop dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
#nama image file temporary setiap kali processing output
filename_out = "out.png"
```



```

# Run the Event Loop
while True:
    event, values = window.read()

    if event == "Exit" or event == sg.WIN_CLOSED:
        break

    # Folder name was filled in, make a list of files in the folder
    if event == "ImgFolder":
        folder = values["ImgFolder"]

        try:
            # Get list of files in folder
            file_list = os.listdir(folder)
        except:
            file_list = []

        fnames = [
            f
            for f in file_list
            if os.path.isfile(os.path.join(folder, f))
            and f.lower().endswith((".png", ".gif"))
        ]

        window["ImgList"].update(fnames)

    elif event == "ImgList": # A file was chosen from the listbox

        try:
            filename = os.path.join(
                values["ImgFolder"], values["ImgList"][0]
            )
            window["FilepathImgInput"].update(filename)
            window["ImgInputViewer"].update(filename=filename)
            window["ImgProcessingType"].update(filename)
            window["ImgOutputViewer"].update(filename=filename)
            img_input = Image.open(filename)
            #img_input.show()

            #Size
            img_width, img_height = img_input.size
            window["ImgSize"].update("Image Size : "+str(img_width)+" x "+str(img_height))

            #Color depth
            mode_to_coldepth = {"1": 1, "L": 8, "P": 8, "RGB": 24, "RGBA": 32, "CMYK": 32, "YCbCr": 24, "LAB":
24, "HSV": 24, "I": 32, "F": 32}
            coldepth = mode_to_coldepth[img_input.mode]

```

```

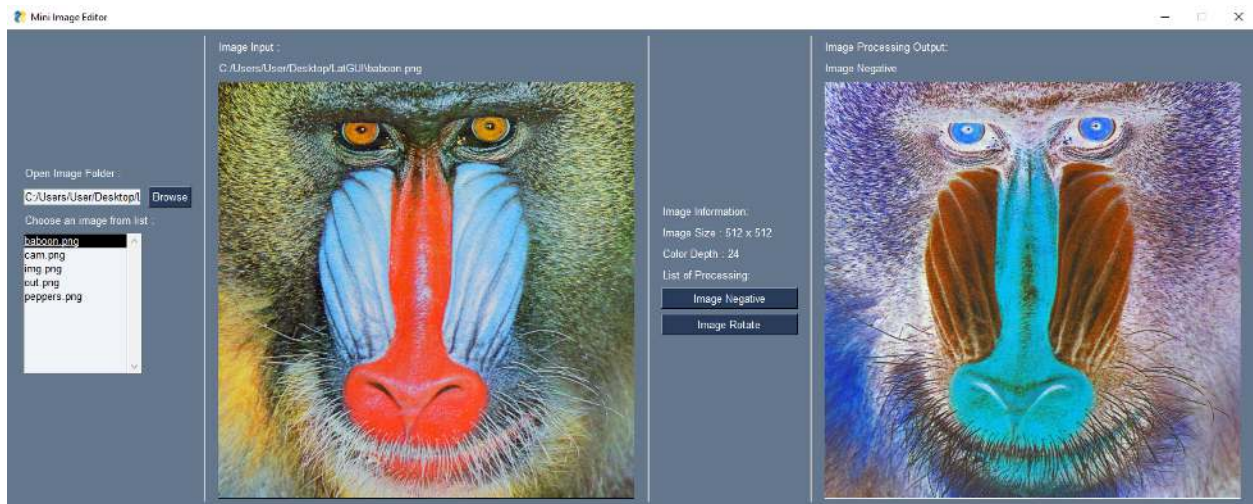
window["ImgColorDepth"].update("Color Depth : "+str(coldepth))
except:
    pass

elif event == "ImgNegative":

    try:
        window["ImgProcessingType"].update("Image Negative")
        img_output=ImgNegative(img_input,coldepth)
        img_output.save(filename_out)
        window["ImgOutputViewer"].update(filename=filename_out)
    except:
        pass

```

Hasil eksekusi :



Ketika tombol Image Negative diklik, maka area Image Processing Output akan menampilkan citra negative dari citra input.

Langkah 5 : Mengimplementasikan algoritma Image Rotate

Lengkapi file `processing_list.py` dengan sebuah fungsi baru yaitu `ImageRotate` sebagai berikut (perhatikan code berwarna biru) :

```
from PIL import Image, ImageOps

def ImgNegative(img_input,coldepth):
    #solusi 1
    #img_output=ImageOps.invert(img_input)

    #solusi 2
    if coldepth!=24:
        img_input = img_input.convert('RGB')

    img_output = Image.new('RGB',(img_input.size[0],img_input.size[1]))
    pixels = img_output.load()
    for i in range(img_output.size[0]):
        for j in range(img_output.size[1]):
            r, g, b = img_input.getpixel((i, j))
            pixels[i,j] = (255-r, 255-g, 255-b)

    if coldepth==1:
        img_output = img_output.convert("1")
    elif coldepth==8:
        img_output = img_output.convert("L")
    else:
        img_output = img_output.convert("RGB")

    return img_output

def ImgRotate(img_input,coldepth,deg,direction):
    #solusi 1
    #img_output=img_input.rotate(deg)

    #solusi 2
    if coldepth!=24:
        img_input = img_input.convert('RGB')

    img_output = Image.new('RGB',(img_input.size[1],img_input.size[0]))
    pixels = img_output.load()
    for i in range(img_output.size[0]):
        for j in range(img_output.size[1]):
            if direction=="C":
                r, g, b = img_input.getpixel((j,img_output.size[0]-i-1))
            else:
                r, g, b = img_input.getpixel((img_input.size[1]-j-1,i))
```

```
pixels[i,j] = (r, g, b)

if coldepth==1:
    img_output = img_output.convert("1")
elif coldepth==8:
    img_output = img_output.convert("L")
else:
    img_output = img_output.convert("RGB")

return img_output
```

Kemudian, lengkapi bagian code Event Loop dari code sebelumnya pada file `img_viewer.py` menjadi sebagai berikut (perhatikan code berwarna biru) :

```
# Run the Event Loop
while True:
    event, values = window.read()

    if event == "Exit" or event == sg.WIN_CLOSED:
        break

    # Folder name was filled in, make a list of files in the folder
    if event == "ImgFolder":
        folder = values["ImgFolder"]

        try:
            # Get list of files in folder
            file_list = os.listdir(folder)
        except:
            file_list = []

        fnames = [
            f
            for f in file_list
            if os.path.isfile(os.path.join(folder, f))
            and f.lower().endswith((".png", ".gif"))
        ]

        window["ImgList"].update(fnames)

    elif event == "ImgList": # A file was chosen from the listbox

        try:
            filename = os.path.join(
                values["ImgFolder"], values["ImgList"][0]
```

```

)
window["FilepathImgInput"].update(filename)
window["ImgInputViewer"].update(filename=filename)
window["ImgProcessingType"].update(filename)
window["ImgOutputViewer"].update(filename=filename)
img_input = Image.open(filename)
#img_input.show()

#Size
img_width, img_height = img_input.size
window["ImgSize"].update("Image Size : "+str(img_width)+" x "+str(img_height))

#Color depth
mode_to_coldepth = {"1": 1, "L": 8, "P": 8, "RGB": 24, "RGBA": 32, "CMYK": 32, "YCbCr": 24, "LAB":
24, "HSV": 24, "I": 32, "F": 32}
coldepth = mode_to_coldepth[img_input.mode]
window["ImgColorDepth"].update("Color Depth : "+str(coldepth))
except:
    pass

elif event == "ImgNegative":

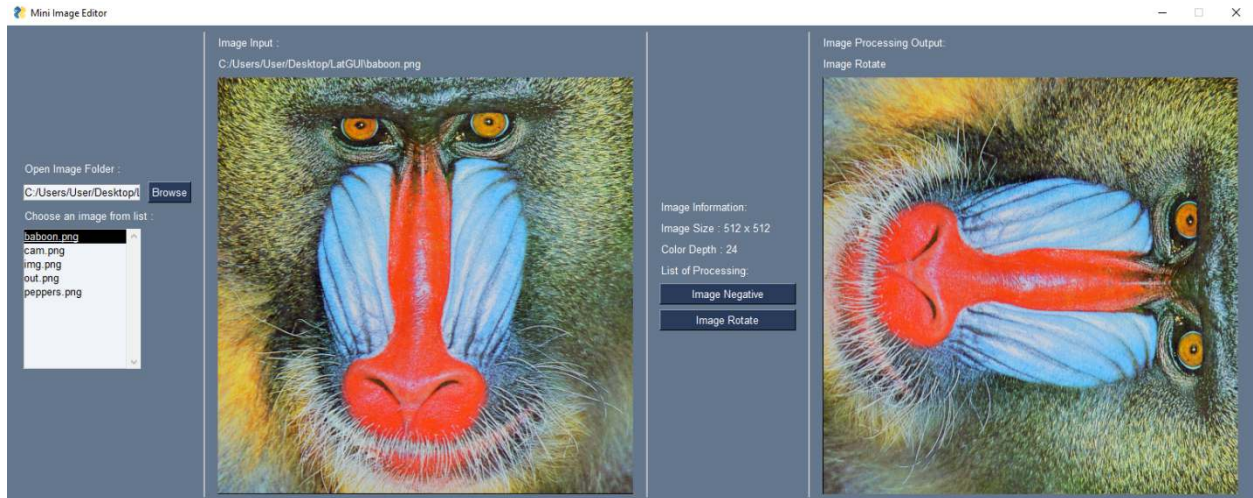
    try:
        window["ImgProcessingType"].update("Image Negative")
        img_output=ImgNegative(img_input,coldepth)
        img_output.save(filename_out)
        window["ImgOutputViewer"].update(filename=filename_out)
    except:
        pass

elif event == "ImgRotate":

    try:
        window["ImgProcessingType"].update("Image Rotate")
        img_output=ImgRotate(img_input,coldepth,90,"C")
        img_output.save(filename_out)
        window["ImgOutputViewer"].update(filename=filename_out)
    except:
        pass

```

Hasil eksekusi :



Ketika tombol Image Rotate diklik, maka area Image Processing Output akan menampilkan citra rotasi 90 derajat arah jarum jam dari citra input.

Complete Code - Modularitas Program

I. MODUL VIEWER

Modul Viewer - File : img_viewer.py - Part 1

```
import PySimpleGUI as sg
import os.path
from PIL import Image, ImageOps
from processing_list import *
```

Modul Viewer - File : img_viewer.py - Part 2

```
# Kolom Area No 1: Area open folder and select image
file_list_column = [
    [
        sg.Text("Open Image Folder :"),
    ],
    [
        sg.In(size=(20, 1), enable_events=True, key="ImgFolder"),
        sg.FolderBrowse(),
    ],
    [
        sg.Text("Choose an image from list :"),
    ],
    [
        sg.Listbox(
            values=[], enable_events=True, size=(18, 10), key="ImgList"
        )
    ],
]
```

Modul Viewer - File : img_viewer.py - Part 3

```
# Kolom Area No 2: Area viewer image input
image_viewer_column = [
    [sg.Text("Image Input :")],
    [sg.Text(size=(40, 1), key="FilepathImgInput")],
    [sg.Image(key="ImgInputViewer")],
]
```

Modul Viewer - File : img_viewer.py - Part 4

```
# Kolom Area No 3: Area Image info dan Tombol list of processing
list_processing = [
    [
        sg.Text("Image Information:"),
    ],
    [
        sg.Text(size=(20, 1), key="ImgSize"),
    ],
    [
        sg.Text(size=(20, 1), key="ImgColorDepth"),
    ],
    [
        sg.Text("List of Processing:"),
    ],
    [
        sg.Button("Image Negative", size=(20, 1), key="ImgNegative"),
    ],
    [
        sg.Button("Image Rotate", size=(20, 1), key="ImgRotate"),
    ],
]
```

Modul Viewer - File : img_viewer.py - Part 5

```
# Kolom Area No 4: Area viewer image output
image_viewer_column2 = [
    [sg.Text("Image Processing Output:"),
    [sg.Text(size=(40, 1), key="ImgProcessingType")],
    [sg.Image(key="ImgOutputViewer")],
]
```

Modul Viewer - File : img_viewer.py - Part 6

```
# Gabung Full layout
layout = [
    [
        sg.Column(file_list_column),
        sg.VSeparator(),
        sg.Column(image_viewer_column),
    ],
]
```



```
sg.VSeperator(),
sg.Column(list_processing),
sg.VSeperator(),
sg.Column(image_viewer_column2),
]
]
```

Modul Viewer - File : img_viewer.py - Part 7

```
window = sg.Window("Mini Image Editor", layout)
#nama image file temporary setiap kali processing output
filename_out = "out.png"

# Run the Event Loop
while True:
    event, values = window.read()

    if event == "Exit" or event == sg.WIN_CLOSED:
        break

    # Folder name was filled in, make a list of files in the folder
    if event == "ImgFolder":
        folder = values["ImgFolder"]

        try:
            # Get list of files in folder
            file_list = os.listdir(folder)
        except:
            file_list = []

        fnames = [
            f
            for f in file_list
            if os.path.isfile(os.path.join(folder, f))
            and f.lower().endswith((".png", ".gif"))
        ]

        window["ImgList"].update(fnames)

    elif event == "ImgList": # A file was chosen from the listbox

        try:
            filename = os.path.join(
                values["ImgFolder"], values["ImgList"][0]
            )
```

```

window["FilepathImgInput"].update(filename)
window["ImgInputViewer"].update(filename=filename)
window["ImgProcessingType"].update(filename)
window["ImgOutputViewer"].update(filename=filename)
img_input = Image.open(filename)
#img_input.show()

#Size
img_width, img_height = img_input.size
window["ImgSize"].update("Image Size : "+str(img_width)+" x "+str(img_height))

#Color depth
mode_to_coldepth = {"1": 1, "L": 8, "P": 8, "RGB": 24, "RGBA": 32, "CMYK": 32, "YCbCr": 24, "LAB":
24, "HSV": 24, "I": 32, "F": 32}
coldepth = mode_to_coldepth[img_input.mode]
window["ImgColorDepth"].update("Color Depth : "+str(coldepth))
except:
    pass

elif event == "ImgNegative":

    try:
        window["ImgProcessingType"].update("Image Negative")
        img_output=ImgNegative(img_input,coldepth)
        img_output.save(filename_out)
        window["ImgOutputViewer"].update(filename=filename_out)
    except:
        pass

elif event == "ImgRotate":

    try:
        window["ImgProcessingType"].update("Image Rotate")
        img_output=ImgRotate(img_input,coldepth,90,"C")
        img_output.save(filename_out)
        window["ImgOutputViewer"].update(filename=filename_out)
    except:
        pass

window.close()

```

II. MODUL PROCESSING

Modul Processing - File : processing_list.py - Part 1

```

from PIL import Image, ImageOps

```

Modul Processing - File : processing_list.py - Part 2

```
def ImgNegative(img_input,coldepth):
    #solusi 1
    #img_output=ImageOps.invert(img_input)

    #solusi 2
    if coldepth!=24:
        img_input = img_input.convert('RGB')

    img_output = Image.new('RGB',(img_input.size[0],img_input.size[1]))
    pixels = img_output.load()
    for i in range(img_output.size[0]):
        for j in range(img_output.size[1]):
            r, g, b = img_input.getpixel((i, j))
            pixels[i,j] = (255-r, 255-g, 255-b)

    if coldepth==1:
        img_output = img_output.convert("1")
    elif coldepth==8:
        img_output = img_output.convert("L")
    else:
        img_output = img_output.convert("RGB")

    return img_output
```

Modul Processing - File : processing_list.py - Part 3

```
def ImgRotate(img_input,coldepth,deg,direction):
    #solusi 1
    #img_output=img_input.rotate(deg)

    #solusi 2
    if coldepth!=24:
        img_input = img_input.convert('RGB')

    img_output = Image.new('RGB',(img_input.size[1],img_input.size[0]))
    pixels = img_output.load()
    for i in range(img_output.size[0]):
        for j in range(img_output.size[1]):
            if direction=="C":
                r, g, b = img_input.getpixel((j,img_output.size[0]-i-1))
```

```
else:
    r, g, b = img_input.getpixel((img_input.size[1]-j-1,i))
    pixels[i,j] = (r, g, b)

if coldepth==1:
    img_output = img_output.convert("1")
elif coldepth==8:
    img_output = img_output.convert("L")
else:
    img_output = img_output.convert("RGB")

return img_output
```