S1IF 08 MM$ - KELOMPOK 3

Anggota :

Hilmy Ahmad Haidar (20102170)

Susilo Firmansyah (20102183)

Usman Pamungkas (20102198)

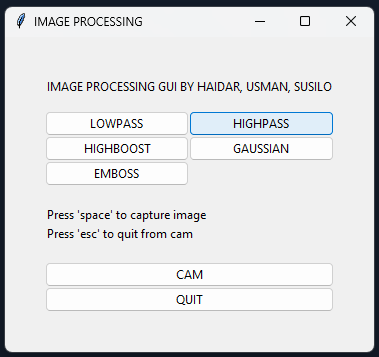
Menu Utama

Adalah tampilan awal dari GUI yang akan dijalankan

**Source Code** reading.py

|  |
| --- |
| *from* tkinter *import* \*  *from* tkinter *import* ttk  *import* os  *import* cv2  *import* numpy *as* np  *from* matplotlib *import* pyplot *as* plt  *def* lowpass():    os.system('python UTS/lowpass.py')  *def* highpass():    os.system('python UTS/highpass.py')    *def* highboost():    os.system('python UTS/highboost.py')  *def* emboss():    os.system('python UTS/emboss.py')  *def* gaussian():    os.system('python UTS/gaussian.py')  *def* cam():    os.system('python UTS/cam.py')  root = Tk()  frm = ttk.Frame(root, padding=40)  root.title("IMAGE PROCESSING")  root.maxsize(600, 400)  frm.grid()  ttk.Label(frm, text="IMAGE PROCESSING GUI BY HAIDAR, USMAN, SUSILO\n").grid(columnspan=2, row=1)  ttk.Button(frm, text="LOWPASS", command=lowpass).grid(column=0, row=2, sticky='ew')  ttk.Button(frm, text="HIGHPASS", command=highpass).grid(column=1, row=2, sticky='ew')  ttk.Button(frm, text="HIGHBOOST", command=highboost).grid(column=0, row=4, sticky='ew')  ttk.Button(frm, text="GAUSSIAN", command=gaussian).grid(column=1, row=4, sticky='ew')  ttk.Button(frm, text="EMBOSS", command=emboss).grid(column=0, row=6, sticky='ew')  ttk.Label(frm, text="").grid(columnspan=3, row=7, sticky='ew')  ttk.Label(frm, text="Press 'space' to capture image").grid(columnspan=3, row=8, sticky='ew')  ttk.Label(frm, text="Press 'esc' to quit from cam").grid(columnspan=3, row=9, sticky='ew')  ttk.Label(frm, text="").grid(columnspan=3, row=10, sticky='ew')  ttk.Button(frm, text="CAM", command=cam).grid(columnspan=3, row=11, sticky='ew')  ttk.Button(frm, text="QUIT", command=root.destroy).grid(columnspan=3, row=12, sticky='ew')  root.mainloop() |

**Output :**



**Modul Camera**

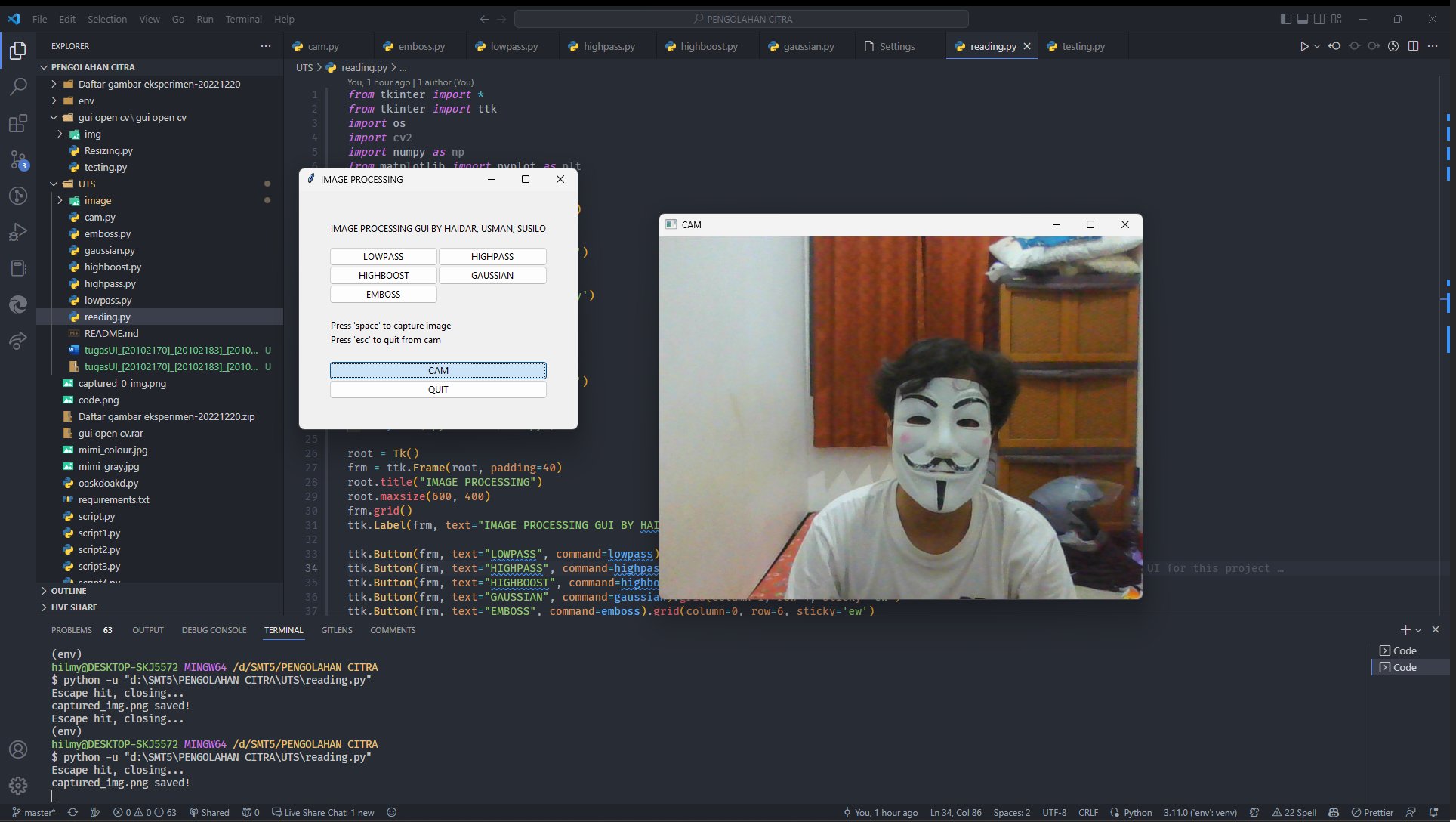
1. Camera

Camera digunakan untuk mengambil gambar yang akan di olah pada modul – modul image processing

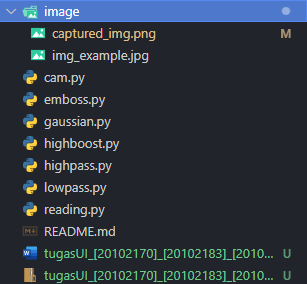
**Source Code** cam.py

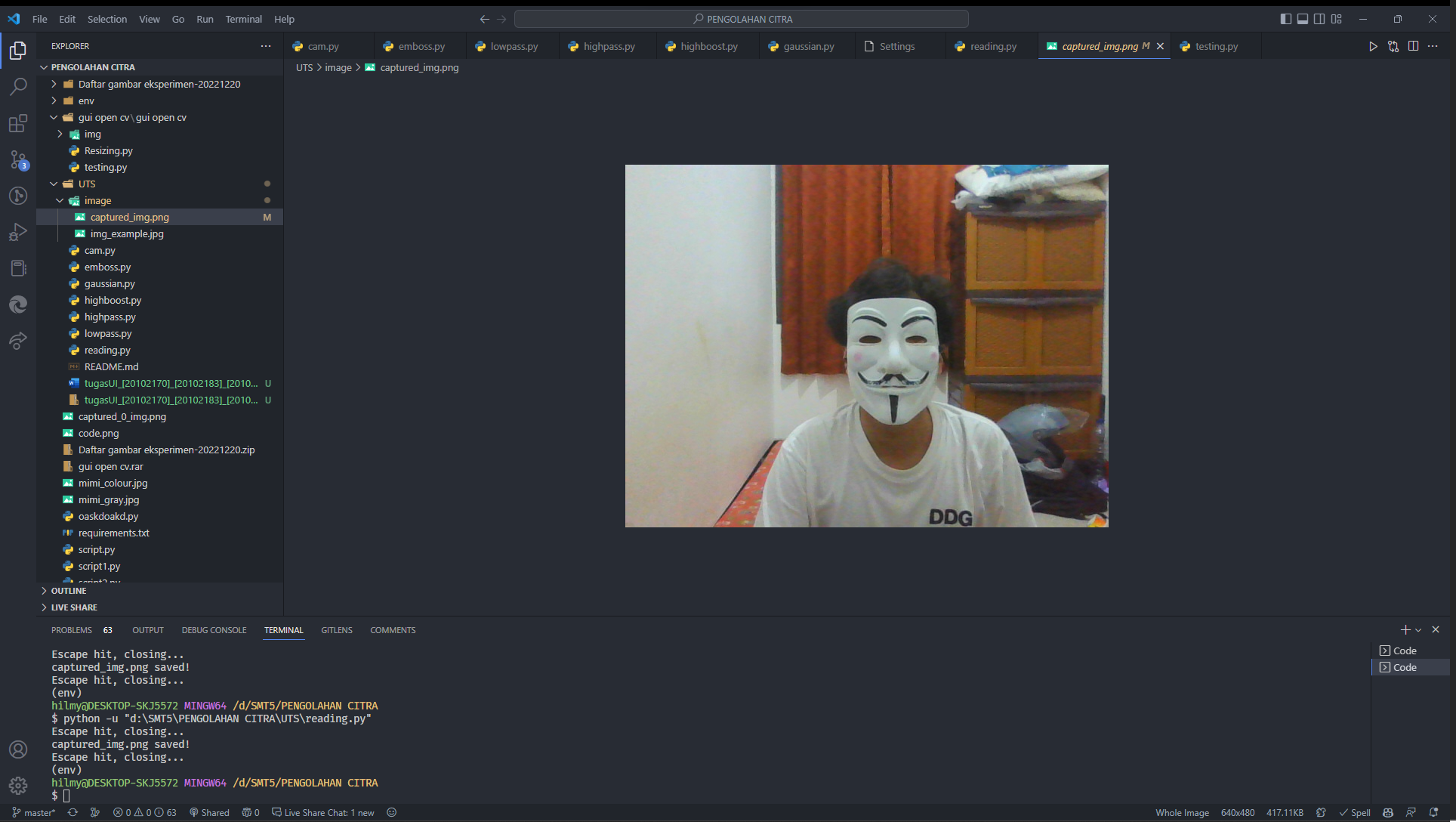
|  |
| --- |
| *import* cv2  *# initialize the camera*  *# If you have multiple camera connected with*  *# current device, assign a value in cam\_port*  *# variable according to that*  cam\_port = 0  cam = cv2.VideoCapture(cam\_port)  cv2.namedWindow("CAM")  img\_counter = 0  *while* True:  *# reading the input using the camera*      result, image = cam.read()  *if* not result:          print("failed to grab image")  *break*  *# show the image*      cv2.imshow("CAM", image)      k = cv2.waitKey(1)  *if* k%256 == 27:  *# If*  *# ESC pressed*  *# Close the CAM window*          print("Escape hit, closing...")  *break*  *elif* k%256 == 32:  *# Else If*  *# SPACE pressed*  *# Capture image to local file*          img\_path = "UTS/image/"          img\_name = "captured\_img.png".format(img\_counter)  *# save the image*          cv2.imwrite(img\_path + img\_name, image)          print("{} saved!".format(img\_name))  *# img\_counter += 1*  cam.release()  cv2.destroyAllWindows() |

**Output**



Untuk image capture otomatis tersimpan didalam file image di direktori project





**Modul Image Processing**

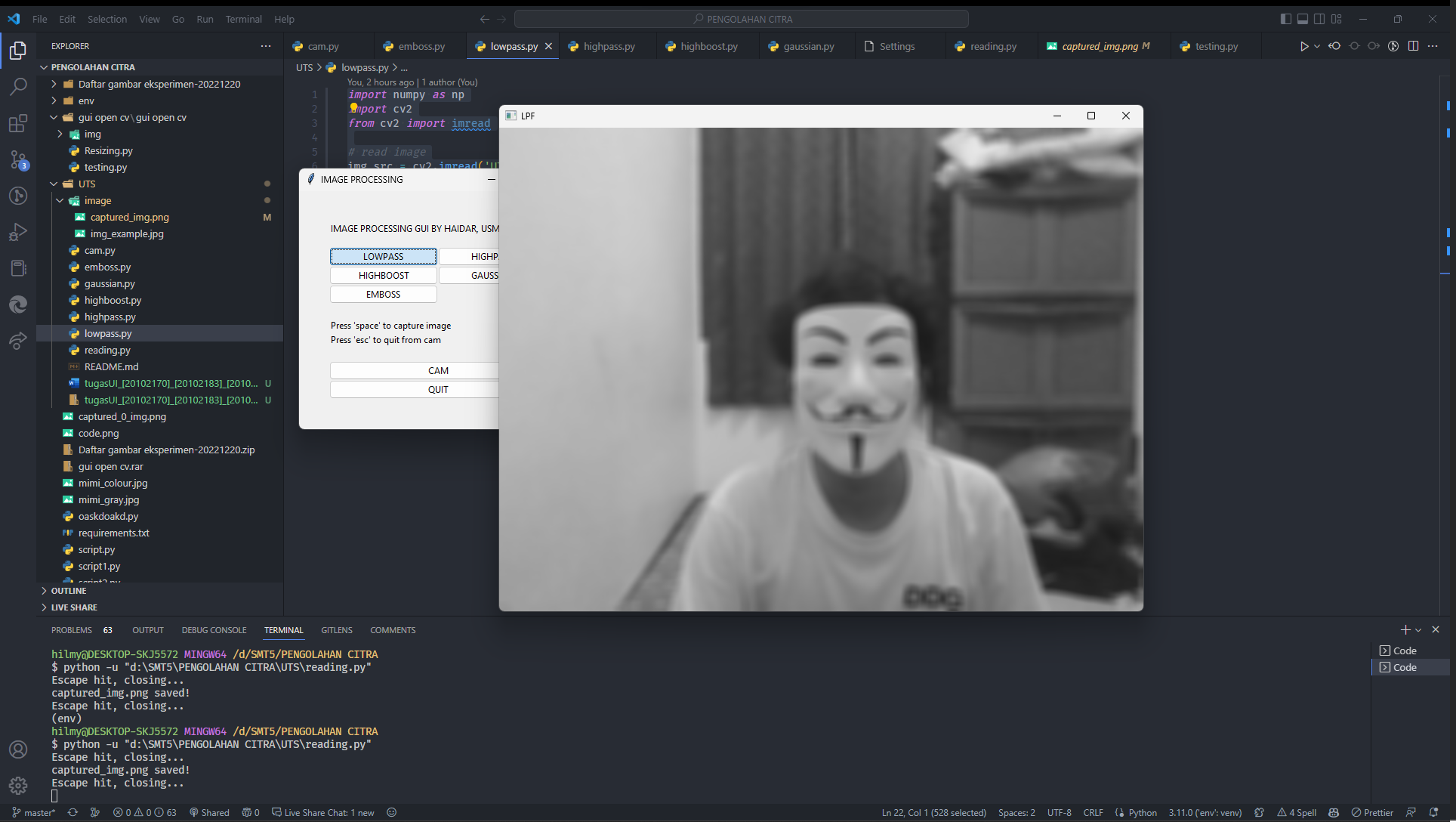
1. Lowpass

Modul lowpass memproses gambar yang telah di capture menggunakan modul camera untuk diprosess

**Source Code** lowpass.py

|  |
| --- |
| *import* numpy *as* np  *import* cv2  *from* cv2 *import* imread  *# read image*  img\_src = cv2.imread('UTS\image\captured\_img.png', 0)  scale\_percent = 20  width = int(img\_src.shape[1] \* scale\_percent / 15)  height = int(img\_src.shape[0] \* scale\_percent / 15)  img\_size = (width, height)  img = cv2.resize(img\_src, img\_size, interpolation=cv2.INTER\_AREA)  kernel = np.ones((13,13), np.float32)/169  flt\_img = cv2.filter2D(src=img, ddepth=-1, kernel=kernel)  cv2.imshow('LPF', flt\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

**Output**

****

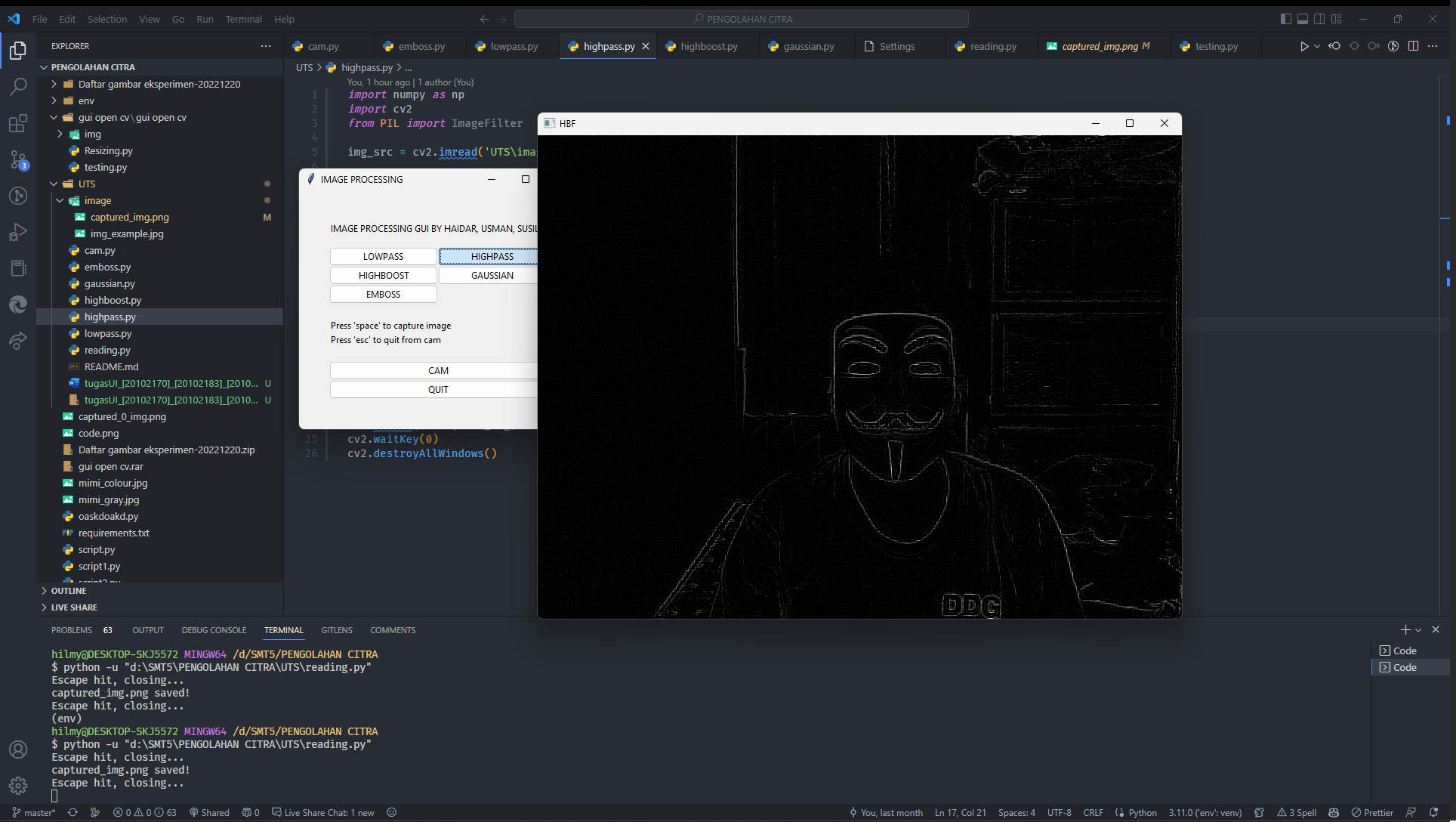
1. Highpass

Modul highpass memproses gambar yang telah di capture menggunakan modul camera untuk diprosess

**Source Code** highpass.py

|  |
| --- |
| *import* numpy *as* np  *import* cv2  *from* PIL *import* ImageFilter  img\_src = cv2.imread('UTS\image\captured\_img.png', 0)  scale\_percent = 20  width = int(img\_src.shape[1] \* scale\_percent / 15)  height = int(img\_src.shape[0] \* scale\_percent / 15)  img\_size = (width, height)  img = cv2.resize(img\_src, img\_size, interpolation=cv2.INTER\_AREA)  kernel = np.array(      [          [-1, -1, -1],          [-1, 8, -1],          [-1, -1, -1],      ]  )  flt\_img = cv2.filter2D(src=img, ddepth=-1, kernel=kernel)  cv2.imshow('HBF', flt\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

**Output**

****

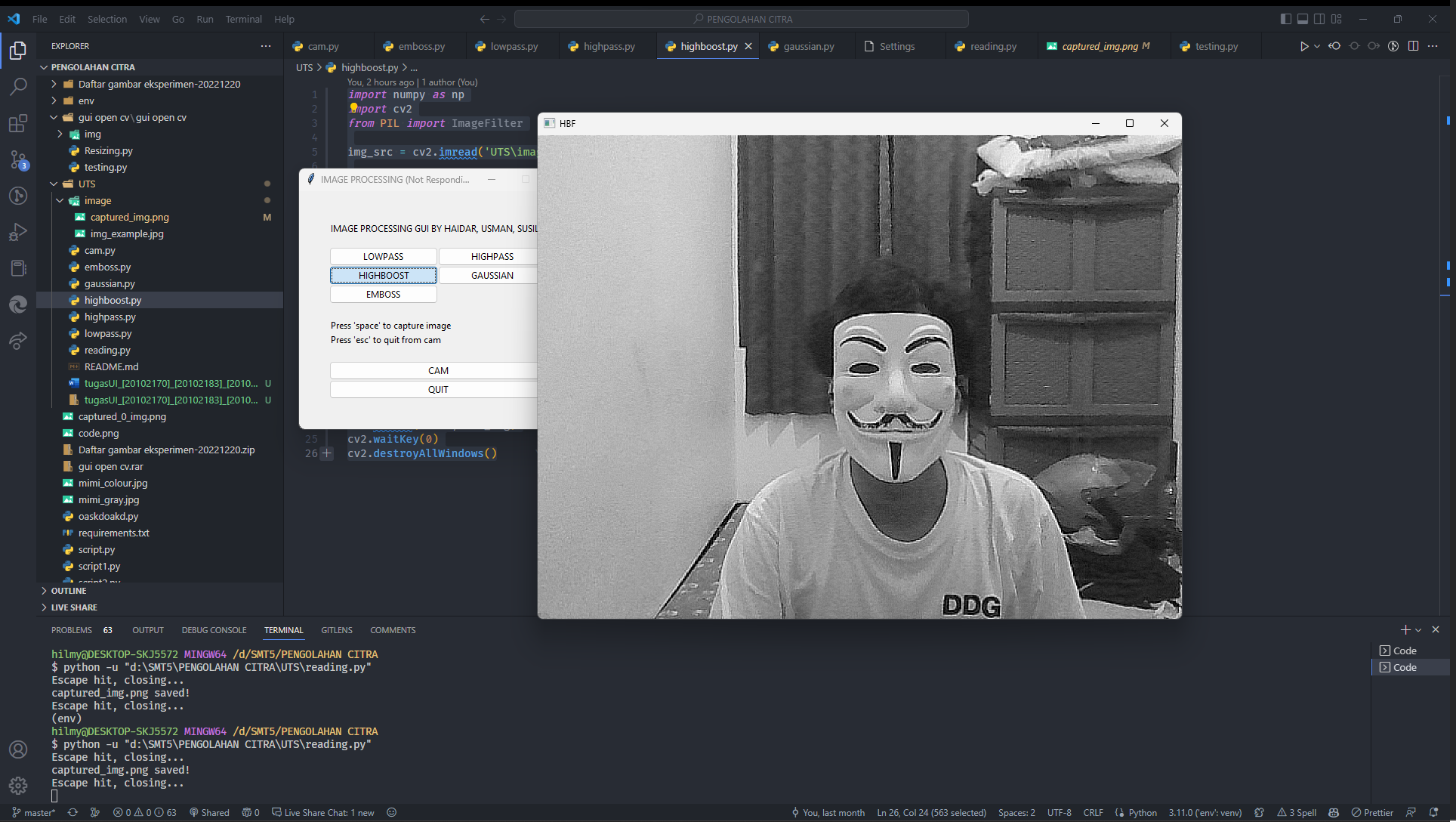
1. Highboost

Modul highboost memproses gambar yang telah di capture menggunakan modul camera untuk diprosess

**Source Code** highboost.py

|  |
| --- |
| *import* numpy *as* np  *import* cv2  *from* PIL *import* ImageFilter  img\_src = cv2.imread('UTS\image\captured\_img.png', 0)  scale\_percent = 20  width = int(img\_src.shape[1] \* scale\_percent / 15)  height = int(img\_src.shape[0] \* scale\_percent / 15)  img\_size = (width, height)  img = cv2.resize(img\_src, img\_size, interpolation=cv2.INTER\_AREA)  kernel = np.array(    [      [-1, -1, -1],      [-1, 9, -1],      [-1, -1, -1],    ]  )  flt\_img = cv2.filter2D(src=img, ddepth=-1, kernel=kernel)  cv2.imshow('HBF', flt\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

**Output**



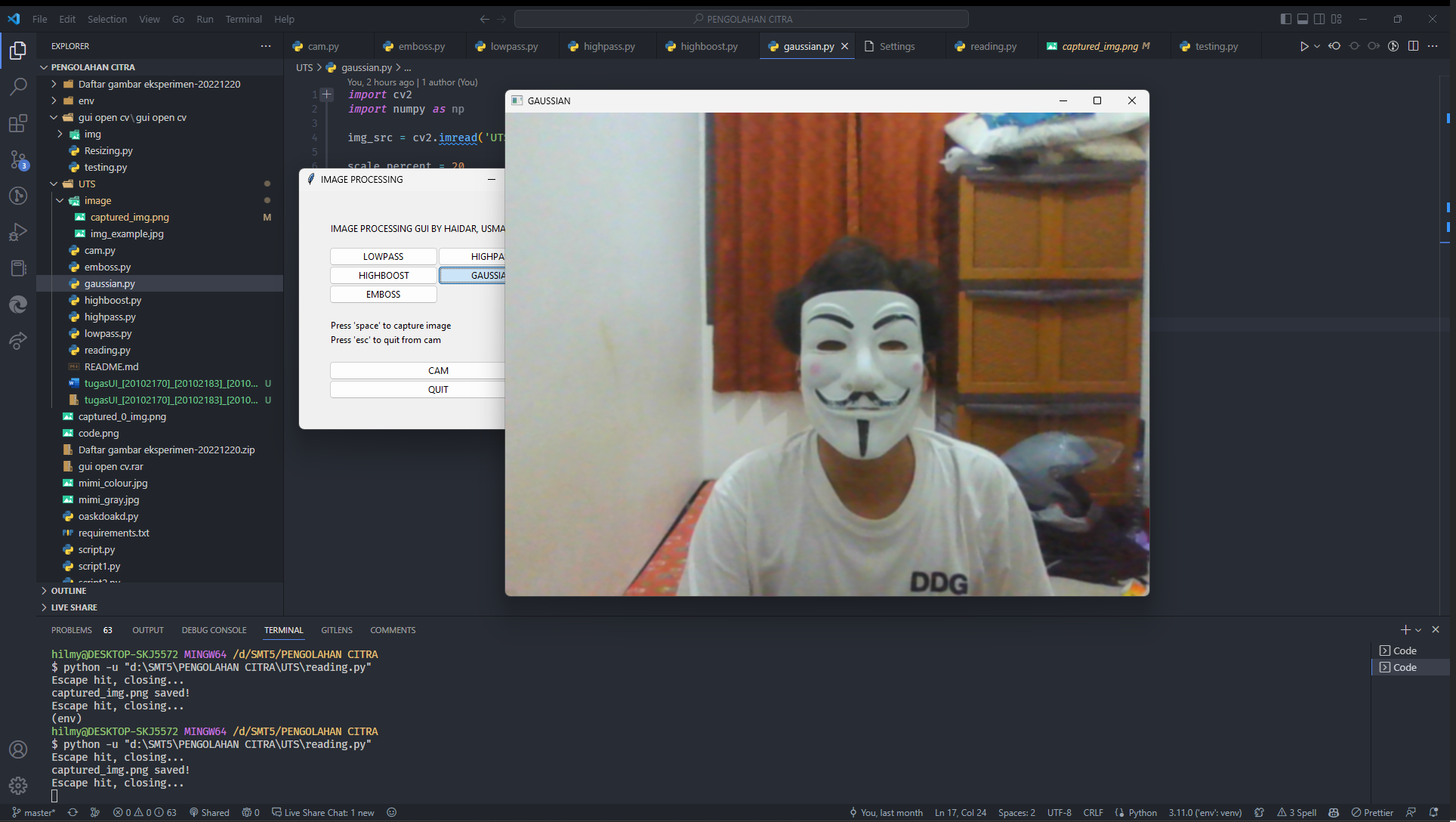
1. Gaussian

Modul gaussian memproses gambar yang telah di capture menggunakan modul camera untuk diprosess

**Source Code** gaussian.py

|  |
| --- |
| *import* cv2  *import* numpy *as* np  img\_src = cv2.imread('UTS\image\captured\_img.png')  scale\_percent = 20  width = int(img\_src.shape[1] \* scale\_percent / 15)  height = int(img\_src.shape[0] \* scale\_percent / 15)  img\_size = (width, height)  img = cv2.resize(img\_src, img\_size, interpolation=cv2.INTER\_AREA)  flt\_img = cv2.GaussianBlur(src=img, ksize=(5,5), sigmaX=0, sigmaY=0)  cv2.imshow('GAUSSIAN', flt\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

**Output**



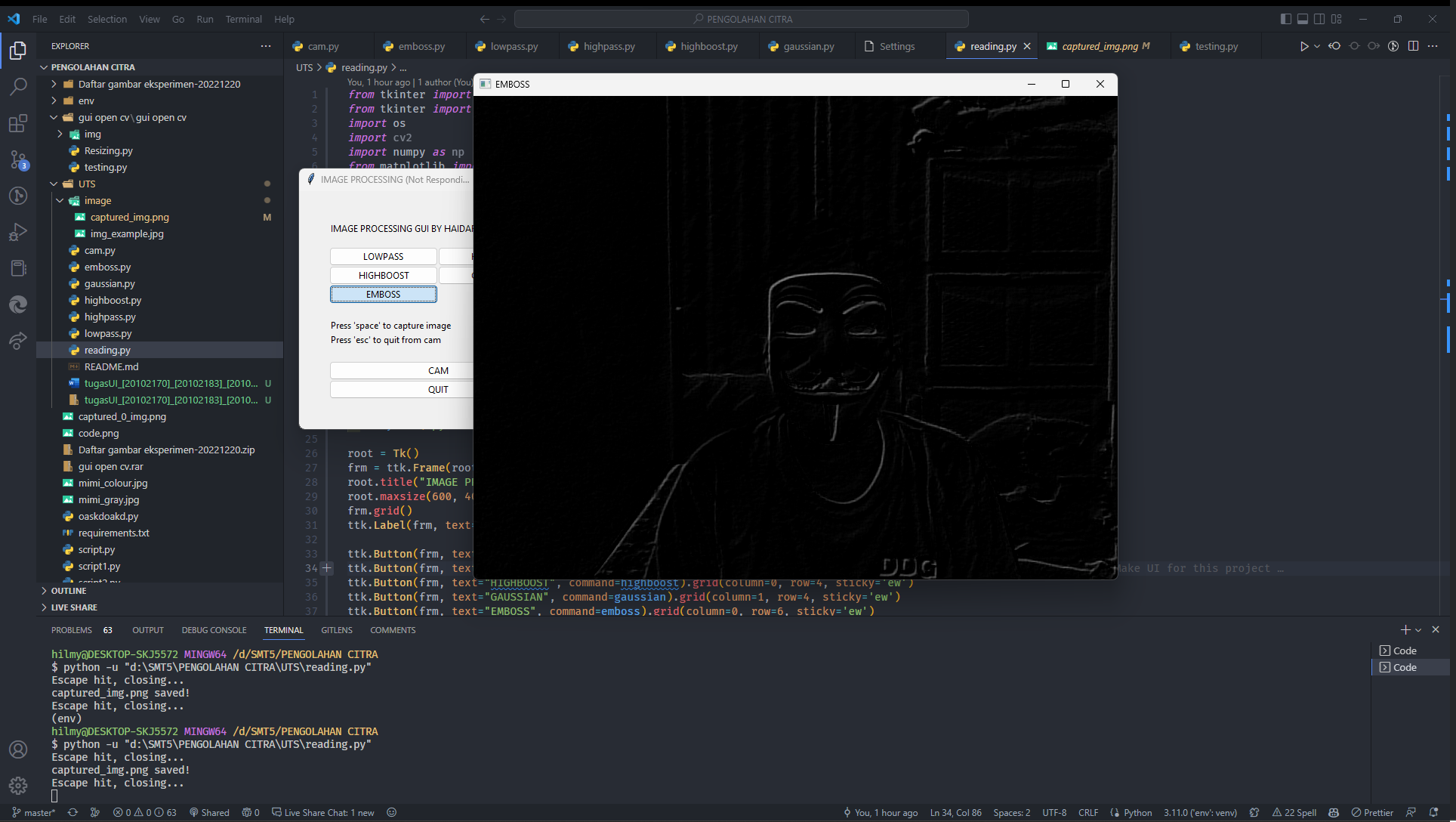
1. Emboss

Modul emboss memproses gambar yang telah di capture menggunakan modul camera untuk diprosess

**Source Code** emboss.py

|  |
| --- |
| *import* numpy *as* np  *import* cv2  *from* PIL *import* ImageFilter  img\_src = cv2.imread('UTS\image\captured\_img.png', 0)  scale\_percent = 20  width = int(img\_src.shape[1] \* scale\_percent / 15)  height = int(img\_src.shape[0] \* scale\_percent / 15)  img\_size = (width, height)  img = cv2.resize(img\_src, img\_size, interpolation=cv2.INTER\_AREA)  kernel = np.array(    [      [-1, 0, 0],      [0, 0, 0],      [0, 0, 1]    ]  )  flt\_img = cv2.filter2D(src=img, ddepth=-1, kernel=kernel)  cv2.imshow('EMBOSS', flt\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

**Output**



Untuk source code dapat dilihat di [link](https://github.com/hilmyha/opencv-image-filtering) github repository