LAPORAN PRAKTIKUM KONSEP PEMROGRAMAN

GROUP PROJECT

"Presensi Otomatis Dengan Face Recognition

Menggunakan OpenCV Python"

Disusun oleh:

Nama (NIM)	:	1. Indy Samha Amalina	(22/498253/SV/21199)
		2. Bagas Bima Himawan	(22/497865/SV/21169)
		3. Muhamad Nur Fauzan	(22/497856/SV/21167)
		4. Faiz Gymnastiar Haryogya	(22/503030/SV/21466)
Hari, Tanggal	:	Senin, 5 Desember 2022	
Dosen Pengampu	:	Yuris Mulya Saputra, S.T., M.Sc., Ph.D	
Asisten Praktikum	:	Sakhiya Abida	
		Sigit Bayu Cahyanto	

PROGRAM STUDI SARJANA TERAPAN

TEKNOLOGI REKAYASA INTERNET

DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

2022

LAPORAN PRAKTIKUM KONSEP PEMROGRAMAN

GROUP PROJECT

"Presensi Otomatis Dengan Face Recognition Menggunakan OpenCV Python"

A. Latar Belakang

Dibuatnya *smart attendance* atau kehadiran otomatis berbasis *face recognition* sebagai upaya dalam mempermudah kegiatan presensi pelajar, pekerja, dan orang-orang yang memerlukan pendataan presensi dalam jumlah banyak. Diharapkan dengan adanya program kehadiran otomatis berbasis *scan* wajah dapat memaksimalkan produktivitas dan meminimalisir terjadinya kesalahan dalam perekapan presensi. Program ini pun diharapkan dapat menjadi solusi dalam meminimalisir kecurangan seperti "titip absen" yang dilakukan oleh oknum-oknum yang tidak bertanggung jawab.

B. Tujuan

Melihat adanya praktik kecurangan seperti "titip absen" yang banyak terjadi di masyarakat bahkan instansi pendidikan sampai pemerintahan, kelompok kami mengusulkan ide pembuatan *smart attendance* berbasis *scan* wajah. Kelompok kami membuat program *group project* presensi otomatis dengan *face recognition* menggunakan *packet* OpenCV pada Python. Tak lupa bahwa kami juga membuat *flowchart* beserta algoritma sebagai *blueprint* dari program yang dibuat.

C. Dasar Teori

a. Pengertian *smart attendance*

Smart attendance atau kehadiran otomatis merupakan salah satu contoh penerapan teknologi dalam kehidupan. Objek program kehadiran otomatis ini dibuat menyesuaikan dengan kebutuhan dari penggunanya. Kehadiran otomatis dapat menggunakan kartu, sidik jari, bahkan wajah. Kelompok kami menggunakan objek wajah sebagai prasarana dalam membuat program *smart attendance* ini.

b. Pengertian *software* Anaconda-Navigator

Anaconda Navigator adalah *graphical user interface* (GUI) desktop yang memungkinkan *user* meluncurkan aplikasi dan mengelola paket conda, *environments*, dan *channel* tanpa menggunakan perintah *command line interface* (CLI). Navigator dapat mencari paket di Anaconda.org atau di repositori Anaconda lokal (Anaconda.org, n.d.).

Conda program CLI adalah manajer paket dan manajer *environments*. Ini membantu ilmuwan data memastikan bahwa setiap versi dari setiap paket memiliki semua keterikatan yang diperlukan dan berfungsi dengan benar (Anaconda.org, n.d.).

Navigator adalah *graphical interface* yang memungkinkan *user* bekerja dengan paket dan *environments* tanpa perlu mengetikkan perintah conda di terminal. *User* dapat menggunakannya untuk menemukan paket yang diinginkan, menginstalnya di *environments*, menjalankan paket, dan memperbaruinya, semuanya ada di dalam Navigator (Anaconda.org, n.d.).

c. Pengertian *face recognition*

Face Recognition adalah teknologi dalam Python dalam mengenali atau mendeteksi wajah seseorang. Dengan teknologi ini, sistem akan melakukan analisa-analisa citra dari foto-foto presensi yang tela tersimpan, apakah benar-benar ada ‘wajah manusia’ yang ada foto-foto yang dimaksud (Kurniawan, 2014). Jika ada, program yang kami buat ini akan menyajikan data-data ke dalam rekapitulasi presensi mahasiswa dalam bentuk data pada *spreadsheet*. Dalam program ini, kami menggunakan modul *face_recognition* yang digunakan untuk menyocokkan wajah pengguna dengan yang ada di *database*.

d. Pengertian *packet cv2* (OpenCV)

Menurut Samarth Brahmbhatt, OpenCV atau *Open-source Computer Vision* adalah pustaka pemrograman yang digunakan untuk pemrosesan citra (*image processing*) secara real-time. Pustaka ini pertama kali dikembangkan di Intel Corporation tahun 2000 yang lalu sampai sekarang. Pustaka yang ditulis dalam bahasa pemrograman C++ ini telah tersedia dalam hampir semua bahasa pemrograman besar, seperti Python, Java, dan MATLAB (Kurniawan, 2014).

OpenCV menggunakan algoritma *Haar* untuk melakukan analisis citra. Algoritma *Haar* sendiri merupakan salah satu dari sekian banyak algoritma matematis untuk menghitung sebuah bentuk dari suatu bagian citra yang diproses. Algoritma *Haar* sendiri berkembang dari teknologi *face recognition* pertama yang disebut *Haar wavelets* yang hanya mendeteksi lengkungan-lengkungan badan dan muka, menjadi *Haar-like features* yang mampu mendeteksi bagian-bagian dari citra yang menyerupai bagian dari muka, misalnya bibir, mata, hidung, dll. Data-data *Haar-like features* sendiri saat ini terus dikembangkan oleh sejumlah peneliti dan dapat diunduh dan dipakai secara bebas (Kurniawan, 2014).

e. Pengertian *packet numpy*

Numpy adalah sebuah *library* pada Python yang berguna untuk melakukan perhitungan *scientific*. Di dalamnya terdapat paket-paket untuk melakukan operasi terhadap objek array yang berupa matriks dengan N-dimensi, aljabar linear, dan banyak lagi (Reynaldo, 2019). Kelebihan dari *Numpy array* adalah dapat memudahkan operasi komputasi pada data, dapat mengakomodasi untuk mengakses data secara acak, dan penyimpanannya dianggap sangat efisien (Uswatun, 2021).

Seperti hal yang sempat disinggung sebelumnya, *numpy* dapat digunakan untuk melakukan operasi pada matriks. Operasi matriks itu antara lain penjumlahan, pengurangan, perkalian, transpose, *invers*, determinan, dan lain-lain (Saputra, 2022).

f. Pengertian modul *os*

Modul *os* di Python menyediakan fungsi yang dapat berinteraksi dengan sistem operasi. *Os* berada di bawah modul utilitas standar Python. Modul ini menyediakan cara portabel untuk menggunakan fungsionalitas yang bergantung pada sistem operasi. Modul *os* ini berisikan banyak fungsi yang dapat berinteraksi dengan sistem *file* (geeksforgeeks.org, OS Module in Python with Examples, 2022).

g. Pengertian modul *datetime*

Di dalam Python, *date* dan *time* bukanlah tipe datanya sendiri, tetapi modul bernama *datetime* dapat diimpor untuk bekerja dengan tanggal dan

juga waktu. Modul Python *datetime* hadir di dalam Python, jadi *user* tidak perlu menginstalnya secara eksternal (geeksforgeeks.org, Python datetime module , 2021).

Modul Python *datetime* menyediakan kelas untuk bekerja dengan tanggal dan waktu. Kelas-kelas ini menyediakan sejumlah fungsi untuk menangani tanggal, waktu, dan interval waktu. Tanggal dan waktu adalah objek dalam Python, jadi saat *user* memanipulasinya, *user* sebenarnya memanipulasi objek dan bukan string atau *timestamps*. Modul *datetime* dikategorikan ke dalam enam kelas utama kelas utama (geeksforgeeks.org, Python datetime module , 2021):

- *Date* – Tanggal dengan menggunakan kalender Gregorian saat ini yang berlaku, dan akan selalu berlaku. Atributnya adalah tahun (*year*), bulan (*month*), dan hari (*day*).
- *Time* – Waktu ideal dengan asumsi bahwa setiap hari memiliki tepat 24*60*60 detik. Atributnya adalah jam (*hour*), menit (*minute*), detik (*second*), mikrodetik (*microsecond*), dan *tzinfo*.
- *Datetime* – Ini adalah kombinasi tanggal dan waktu bersama dengan atribut tahun (*year*), bulan (*month*), hari (*day*), jam (*hour*), menit (*minute*), detik (*second*), mikrodetik (*microsecond*), dan *tzinfo*.
- *Timedelta* – Durasi yang menyatakan perbedaan antara dua tanggal, waktu, atau *instance datetime* ke resolusi mikrodetik.
- *Tzinfo* – Ini menyediakan objek informasi zona waktu.
- *Timezone* – Kelas yang mengimplementasikan kelas dasar abstrak *tzinfo* sebagai *offset* tetap dari UTC (fitur baru dalam versi 3.2).

h. Pengertian modul *shutil*

Modul *shutil* Python menyediakan fasilitas untuk melakukan operasi *file* tingkat tinggi. Itu dapat beroperasi dengan objek *file* dan memberi *user* kemampuan untuk menyalin dan menghapus *file*. Ini menangani semantik tingkat rendah seperti membuat dan menutup objek file setelah melakukan semua operasi. Modul ini menyediakan *method* copy() untuk menyalin data, menyalin *file* dengan *method* copyfile(), *method* copytree() untuk menyalin seluruh *directory tree* yang di-root pada sumber ke direktori

tujuan, *method* rmtree() yang dapat digunakan untuk menghapus *directory tree*, dan masih banyak lainnya (javatpoint.com, n.d.).

i. Pengertian *packet tkinter*

Tkinter merupakan *library* yang dapat digunakan untuk membuat aplikasi *interface* (GUI) Python yang berbentuk OOP dari TCL/TK. *Tkinter* menyediakan cara cepat dan mudah yang dengan berorientasikan objek yang kuat dalam membuat aplikasi python berbasiskan GUI. *Tkinter* biasanya secara *default* di-*bundle* dengan Python, atau paket ini sudah tersedia langsung di dalam Python tanpa harus *user* instal terlebih dahulu. TCL (*Tool Command Language*) adalah sebuah bahasa pemrograman dan TK adalah *library* yang digunakan oleh TCL untuk membuat aplikasi GUI (Tineges, 2021).

Paket *tkinter* juga memiliki kelebihan dan kekurangan yang bisa *user* gunakan sebagai bahan pertimbangan sebelum menggunakan *library* ini. Adapun kelebihan dan kekurangan TKINTER sebagai berikut (Tineges, 2021):

Kelebihan:

- *Open source*.
- Karena termasuk dalam Python *library standard*, maka *user* tidak perlu melakukan apapun dan hanya tinggal menggunakannya karena sudah termasuk dalam paket instalasi Python.
- Tidak terlalu rumit dan mudah dipelajari

Kekurangan:

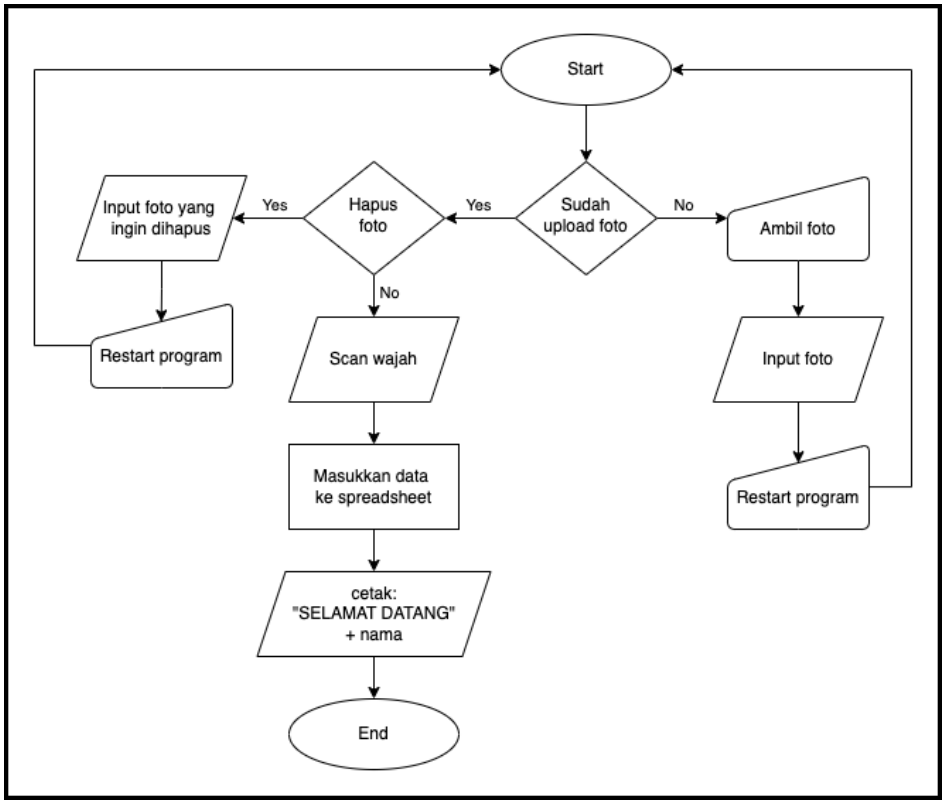
- Tidak ada *widget canggih*, *tkinter* tidak memiliki *widget* yang *advanced* untuk digunakan seperti *datepicker*.
- *Tkinter* juga tidak memiliki (atau belum) IDE GUI seperti QT Designer pada PyQT.

D. Alat dan Bahan

a. Komputer atau laptop;
b. *Software* Visual Studio Code; dan
c. *Software* Anaconda-Navigator;

E. Langkah Praktikum dan Pembahasan

a. Membuat *flowchart* program



b. Algoritma program

a. *Start*.

b. Pengondisian pertama dengan *statement* "apakah *user* sudah meng-*upload* foto?"

a. Jika belum (*no*), *user* mengambil foto secara manual. Kemudian *user* menginputkan foto tersebut. Terakhir, *user* harus me-*restart* program presensi tersebut secara manual. Kemudian program akan kembali ke *start*.

b. Jika sudah (*yes*), maka program akan dilanjutkan ke pengondisian kedua.

c. Pengondisian kedua ini memiliki *statement* "apakah *user* ingin menghapus foto?"

a. Jika ya (*yes*), *user* menginputkan foto mana yang ingin dihapus secara manual. Kemudian *user* harus me-*restart* program presensi tersebut secara manual. Terakhir program akan kembali ke *start*.

b. Jika tidak (*no*), maka program akan dilanjutkan ke tahap selanjutnya

d. Program akan men-*scan* wajah *user*.

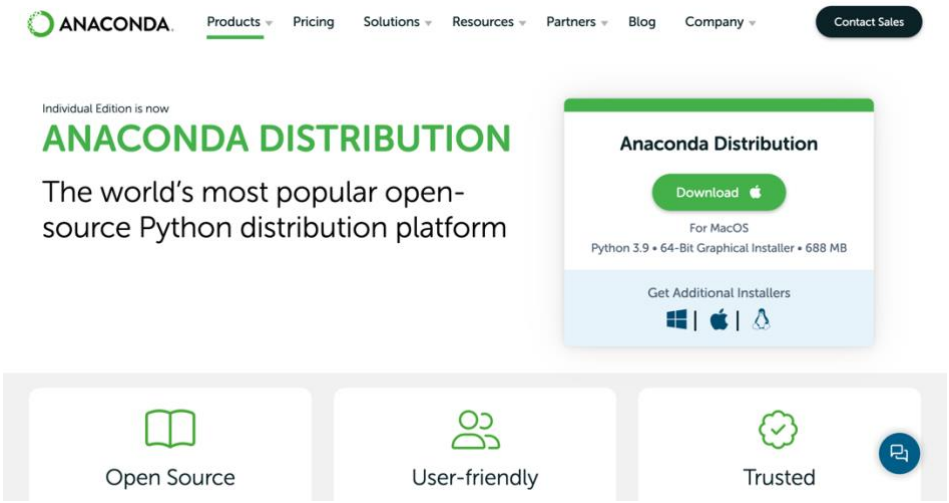
e. Program memasukkan data presensi ke dalam *spreadsheet*.

f. Cetak "SELAMAT DATANG" dan nama *user* tersebut.

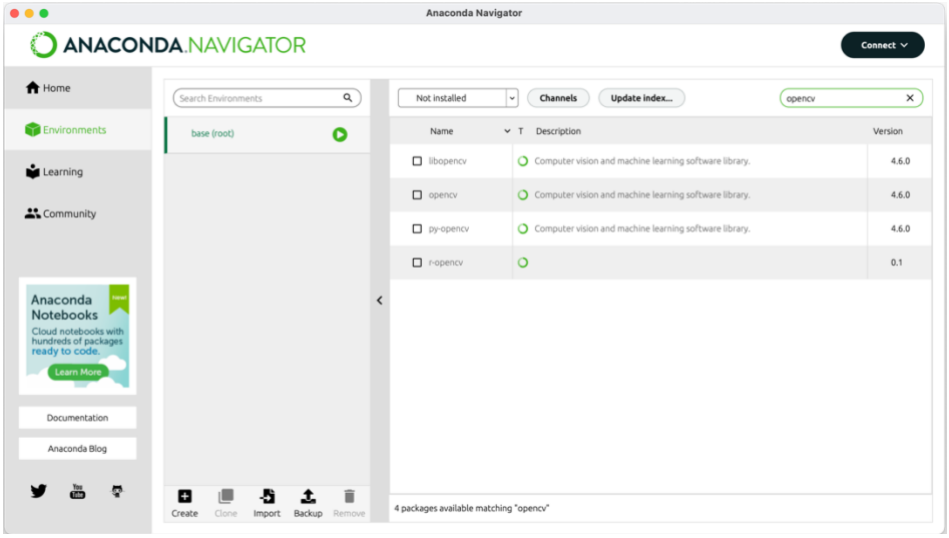
g. *End*.

c. Menginstal *software* Anaconda-Navigator

Kita mengunduh aplikasi Anaconda Navigator pada *website* resminya yaitu https://www.anaconda.com/products/distribution. Aplikasi ini dapat diinstal dalam berbagai OS, seperti Windows, MacOS, dan Linux.



Aplikasi ini kita gunakan untuk menginstal modul dan paket yang akan kita gunakan nantinya. Setelah *software* ini terinstal, kita melakukan penginstalan modul dan paket *numpy*, face_*recognition* dan *cv2*. Modul dan paket lainnya seperti *os, tkinker, datetime,* dan *shutil* merupakan bawaan pada Python sehingga kita tidak perlu menginstalnya.



d. Pembuatan program utama Absen Wajah.py

```
Absen Wajah.py > ...
1   import tkinter as tk
2   from fungsi.fungsi_kehadiran import *
3   from fungsi.fungsi_gui import *
4
5   layar = tk.Tk()
6   layar.title("Universitas Gadjah Mada")
7   layar.geometry("640x360")
8
9   label1 = tk.Label(layar, text = "Program Presensi Mahasiswa TRI", font=("Hershey_Complex", 20, "bold"),
    pady=20)
10
11  gambar = tk.PhotoImage(file = "logo.png")
12  label2 = tk.Label(layar, image=gambar)
13
14  button1 = tk.Button(layar, text = "Input Foto", font = "Hershey_Complex", padx = 33, pady = 4, command =
    foto)
15  button2 = tk.Button(layar, text = "Mulai Presensi", font = "Hershey_Complex",padx = 20, pady = 5, command
    = SystemAttandance)
16  button3 = tk.Button(layar, text = "Hapus Foto", font = "Hershey_Complex",padx = 30, pady = 4, command =
    hapus)
17
18  label1.pack()
19  label2.pack(pady=30)
20  button1.pack()
21  button2.pack()
22  button3.pack()
23  layar.mainloop()
24
```

Program ini merupakan program utama yang nantinya dijalankan. Program ini berisikan GUI yang dibuat dengan *packet tkinter*.

- import tkinter as tk

  Sintaks ini digunakan dalam melakukan impor paket *tkinter* ke dalam program supaya paket tersebut dapat digunakan. Impor ini dilakukan dengan menggunakan *keyword* ***import*** dan nama dari paket yang diimporkan. Kemudian kita membuat alias pada paket tersebut menjadi **"tk"** dengan menggunakan *keyword* ***as*** untuk menyingkatnya jika kita butuh memanggilnya.

- from fungsi.fungsi_kehadiran import *

  Sintaks ini difungsikan untuk memanggil modul yang dibuat sebelumnya bernama **fungsi_kehadiran.py**. Modul ini berada di dalam paket (folder) **fungsi**, jadi saat memanggilnya kita perlu memanggil nama paket tersebut dan dihubungkan dengan nama modulnya menggunakan titik (.). Arti dari sintaks ini adalah dari modul **fungsi_kehadiran.py** di dalam paket **fungsi**, impor semua data di dalamnya. Simbol bintang (*) melambangkan semua data yang dipanggil.

- from fungsi.fungsi_gui import *

Sama seperti sintaks sebelumnya, sintaks ini juga digunakan untuk memanggil modul yang dibuat sebelumnya bernama **fungsi_gui.py**. Modul ini berada di dalam paket (folder) **fungsi**, jadi saat memanggilnya kita perlu memanggil nama paket tersebut dan dihubungkan dengan nama modulnya menggunakan titik (.). Arti dari sintaks ini adalah dari modul **fungsi_gui.py** di dalam paket **fungsi**, impor semua data di dalamnya. Simbol bintang (*) melambangkan semua data yang dipanggil.

- layar = tk.Tk()

  Kita mendefinikan sebuah variabel bernama **"layar"** yang nilainya berupa pemanggilan *method* **Tk()** yang dimiliki oleh paket *tkinter*. Variabel ini bisa dibilang sebagai kanvas atau dasar pada jendela GUI yang dibuat nantinya.

- layar.title("Universitas Gadjah Mada")

  Variabel **"layar"** dihubungkan dengan *method* **tittle()** dengan argumen **"Universitas Gadjah Mada"**. Maksud dari sintaks ini adalah kita membuat judul pada jendela GUI **"layar"** yang dibuat sebelumnya dengan bantuan *method* **tittle()**. Argumen di dalam *method* tersebut merupakan judul yang akan dipakai pada jendela GUI nantinya.

- layar.geometry("640x360")

  Method **geometry()** yang dihubungkan dengan GUI **"layar"** memiliki fungsi untuk membuat ukuran jendela GUI sesuai yang diinginkan. Sintaks ini memiliki maksud untuk menampilkan GUI *tkinter* **"layar"** dengan ukuran yang diambil dari argumen pada *method* **geometry()** sebesar **"640x360"**, artinya jendela nantinya akan berukuran 640 x 360 *pixel*.

- label1 = tk.Label(layar, text = "Program Presensi Mahasiswa TRI", font=("Hershey_Complex", 20, "bold"), pady=20)

  Sintaks ini merupakan pendeklarasian variabel **"label1"**. Variabel ini menggunakan *method* **Label()** milik *tkinter* dengan fungsi untuk mencetak label pada jendela GUI yang dibuat. *Method* ini

memanggil jendela **"layar"** dan beberapa parameter antara lain ***text*** yang digunakan untuk menampilkan *text* berupa *string* di label pada GUI, ***font*** digunakan untuk menentukan *font*, *size*, dan jenis dari tulisan yang ingin ditampilkan, dan terakhir parameter ***pady*** untuk memberikan *padding* secara vertikal pada objek.

- gambar = tk.PhotoImage(file = "logo.png")

  Sintaks ini adalah pendeklarasian variabel **"gambar"** dengan menggunakan *method* **PhotoImage()** yang dimiliki oleh *tkinter*. *Method* ini digunakan untuk menyimpan gambar dengan parameter yang menunjukan nama dari gambar tersebut.

- label2 = tk.Label(layar, image=gambar)

  Sintaks ini merupakan pendeklarasian variabel **"label2"**. Variabel ini menggunakan *method* **Label()** milik *tkinter* dengan fungsi untuk mencetak label pada jendela GUI yang dibuat. *Method* ini memanggil jendela **"layar"** dan parameter ***image*** untuk menampilkan gambar pada variabel yang dideklarasikan sebelumnya.

- button1 = tk.Button(layar, text = "Input Foto", font = "Hershey_Complex", padx = 33, pady = 4, command = foto)

  Sama seperti **Label()**, sintaks ini menggunakan *method* milik *tkinter*. Kita mendeklarasikan variabel **"button1"** untuk menampilkan objek *button* atau tombol pada jendela GUI. *Method* ini memanggil jendela **"layar"** dan beberapa parameter antara lain ***text*** yang digunakan untuk menampilkan *text* berupa *string* di tombol pada GUI, ***font*** digunakan untuk menentukan *font*, *size*, dan jenis dari tulisan yang ingin ditampilkan, parameter ***padx*** dan ***pady*** yang masing-masing memberikan *padding* secara horizontal dan vertikal pada objek, dan terakhir parameter ***command*** untuk menghubungkan tombol ini dengan fungsi **"foto"** yang dipanggil.

- button2 = tk.Button(layar, text = "Mulai Presensi", font = "Hershey_Complex",padx = 20, pady = 5, command = SystemAttandance)

Sama seperti sintaks sebelumnya, sintaks ini menggunakan *method* milik *tkinter*. Kita mendeklarasikan variabel **"button2"** untuk menampilkan objek *button* atau tombol pada jendela GUI. *Method* ini memanggil jendela **"layar"** dan beberapa parameter antara lain ***text*** yang digunakan untuk menampilkan *text* berupa *string* di tombol pada GUI, ***font*** digunakan untuk menentukan *font*, *size*, dan jenis dari tulisan yang ingin ditampilkan, parameter ***padx*** dan ***pady*** yang masing-masing memberikan *padding* secara horizontal dan vertikal pada objek, dan terakhir parameter ***command*** untuk menghubungkan tombol ini dengan fungsi **"SystemAttandance"** yang dipanggil.

- button3 = tk.Button(layar, text = "Hapus Foto", font = "Hershey_Complex",padx = 30, pady = 4, command = hapus)

  Sama seperti sintaks sebelumnya, sintaks ini menggunakan *method* milik *tkinter*. Kita mendeklarasikan variabel **"button3"** untuk menampilkan objek *button* atau tombol pada jendela GUI. *Method* ini memanggil jendela **"layar"** dan beberapa parameter antara lain ***text*** yang digunakan untuk menampilkan *text* berupa *string* di tombol pada GUI, ***font*** digunakan untuk menentukan *font*, *size*, dan jenis dari tulisan yang ingin ditampilkan, parameter ***padx*** dan ***pady*** yang masing-masing memberikan *padding* secara horizontal dan vertikal pada objek, dan terakhir parameter ***command*** untuk menghubungkan tombol ini dengan fungsi **"hapus"** yang dipanggil.

- label1.pack()

  Kita menghubungkan objek **"label1"** yang dibuat sebelumnya dengan *method* **pack()**. *Method* ini digunakan untuk membantu membuat *layout* dengan mengatur tata letak *widget* pada jendela GUI yang dibuat.

- label2.pack(pady=30)

  Kita menghubungkan objek **"label2"** yang dibuat sebelumnya dengan *method* **pack()**. *Method* ini digunakan untuk membantu membuat *layout* dengan mengatur tata letak *widget* pada

jendela GUI yang dibuat. Kita memberikan parameter ***pady*** dengan nilai 30, artinya kita membuat *padding* secara vertikal sebesar 30 *pixel*.

- button1.pack()

  Kita menghubungkan objek **"button1"** yang dibuat sebelumnya dengan *method* **pack()**. *Method* ini digunakan untuk membantu membuat *layout* dengan mengatur tata letak *widget* pada jendela GUI yang dibuat.

- button2.pack()

  Kita menghubungkan objek **"button2"** yang dibuat sebelumnya dengan *method* **pack()**. *Method* ini digunakan untuk membantu membuat *layout* dengan mengatur tata letak *widget* pada jendela GUI yang dibuat.

- button3.pack()

  Kita menghubungkan objek **"button3"** yang dibuat sebelumnya dengan *method* **pack()**. *Method* ini digunakan untuk membantu membuat *layout* dengan mengatur tata letak *widget* pada jendela GUI yang dibuat.

- layar.mainloop()

  Kita menghubungkan jendela GUI **"layar"** yang dibuat sebelumnya dengan *method* **mainloop()**. *Method* ini digunakan untuk me-*loop* tak terbatas dalam menjalankan aplikasi. Jadi GUI *tkinter* akan selalu ditampilkan sampai *user* sendiri yang menutup GUI tersebut.

e. Pembuatan program modul fungsi_kehadiran.py

```
fungsi > fungsi_kehadiran.py > resize
import cv2
import os
import numpy as np
import face_recognition as face_rec
from datetime import datetime

def resize(img, size) :
    width = int(img.shape[1] * size)
    height = int(img.shape[0] * size)
    dimension = (width, height)
    return cv2.resize(img, dimension, interpolation= cv2.INTER_AREA)

def findEncoding(images) :
    imgEncodings = []
    for img in images :
        img = resize(img, 0.50)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encodeimg = face_rec.face_encodings(img)[0]
        imgEncodings.append(encodeimg)
    return imgEncodings

def MarkAttendence(name):
    with open('attendence.csv', 'r+') as f:
        myDatalist =  f.readlines()
        nameList = []
        for line in myDatalist :
            entry = line.split(',')
            nameList.append(entry[0])

        if name not in nameList:
            now = datetime.now()
            timestr = now.strftime('%H:%M')
            f.writelines(f'\n{name}, {timestr}')

path = 'student'
studentImg = []
studentName = []
myList = os.listdir(path)
for cl in myList :
    curimg = cv2.imread(f'{path}/{cl}')
    studentImg.append(curimg)
    studentName.append(os.path.splitext(cl)[0])

def SystemAttandance() :
    EncodeList = findEncoding(studentImg)
    vid = cv2.VideoCapture(0)
    while True :
        success, frame = vid.read()
        Smaller_frames = cv2.resize(frame, (0,0), None, 0.25, 0.25)

        facesInFrame = face_rec.face_locations(Smaller_frames)
        encodeFacesInFrame = face_rec.face_encodings(Smaller_frames, facesInFrame)

        for encodeFace, faceloc in zip(encodeFacesInFrame, facesInFrame) :
            matches = face_rec.compare_faces(EncodeList, encodeFace)
            facedis = face_rec.face_distance(EncodeList, encodeFace)
            print(facedis)
            matchIndex = np.argmin(facedis)

            if matches[matchIndex] :
                name = studentName[matchIndex].upper()
                y1, x2, y2, x1 = faceloc
                y1, x2, y2, x1 = y1*4, x2*4, y2*4, x1*4
                cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 3)
                cv2.rectangle(frame, (x1, y2-25), (x2, y2), (0, 255, 0), cv2.FILLED)
                cv2.putText(frame, name, (x1+6, y2-6), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)
                MarkAttendence(name)
                cv2.putText(frame, "SELAMAT DATANG " + name, (300, 100), cv2.FONT_HERSHEY_COMPLEX, 1,
                (255, 255, 255), 2)

        cv2.imshow('Kehadiran Mahasiswa TRI',frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cv2.destroyAllWindows()
```

Sintaks ini merupakan sebuah modul **fungsi_kehadiran.py** yang disimpan di dalam paket (folder) **fungsi** yang dibuat. Modul ini berisikan fungsi-fungsi yang digunakan pada program kehadiran nantinya.

- Bagian pertama

```
import cv2
import os
import numpy as np
import face_recognition as face_rec
from datetime import datetime
```

Sintaks ini berisikan *keyword* ***from*** dan ***import*** untuk mengimpor modul dan paket yang digunakan pada program modul ini. Kita mengimpor beberapa modul dan paket seperti *cv2, os, numpy, face_recognition,* dan *datetime*. Paket *cv2* digunakan untuk melakukan perekaman pada wajah pengguna, modul *face_recognition* dipakai sebagai sarana dalam menyocokkan antara wajah pengguna dan yang ada di *database*, modul *os* digunakan untuk melakukan interaksi dengan sistem operasi yang digunakan secara langsung, paket *numpy* digunakan untuk melakukan perhitungan aritmatika, dan penggunaan kelas *datetime* pada modul *datetime* yang digunakan untuk mengambil waktu yang terjadi di perangkat yang digunakan.

- Bagian kedua

```
def resize(img, size) :
    width = int(img.shape[1] * size)
    height = int(img.shape[0] * size)
    dimension = (width, height)
    return cv2.resize(img, dimension, interpolation= cv2.INTER_AREA)
```

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **resize()** yang memiliki argumen **"img"** dan **"size"**. Fungsi ini digunakan untuk merubah ukuran dari gambar yang diinputkan *user*. Kita mendefinisikan variabel baru yaitu **"width"** yang nilainya merupakan *integer* yang diambil dari lebar dimensi gambar, **"height"** nilainya adalah *integer* yang diambil dari tinggi dimensi gambar, dan variabel **"dimension"** yang nilainya adalah *tuple* dengan *value* dari variabel **"width"** dan **"height"** yang mendefinisikan sebelumnya. Terakhir kita mengembalikkan nilai dari fungsi **resize()** yang dimiliki *cv2*.

- Bagian ketiga

```
def findEncoding(images) :
    imgEncodings = []
    for img in images :
        img = resize(img, 0.50)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encodeimg = face_rec.face_encodings(img)[0]
        imgEncodings.append(encodeimg)
    return imgEncodings
```

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **findEncoding()** yang memiliki argumen **"images"**. Fungsi ini nantinya digunakan untuk mencari wajah dari mencocokkan antara foto yang ada di *database* dengan wajah dari *user*. Pada fungsi ini kita membuat sebuah *array* kosong bernama **"imgEncodings"**. Kemudian kita lakukan perulangan pada setiap gambar yang disimpan di dalam *database*. Kita membuat gambar pada *database* tersebut menjadi berukuran tertentu dengan fungsi **resize()** dan berwarna dengan fungsi **cvtColor()** milik *cv2*. Hal ini perlu dilakukan karena saat program dijalankan, *cv2* akan membuat foto yang dimiliki menjadi hitam putih sehingga sulit untuk menyocokkannya dengan wajah *user*. Kemudian kita mendeklarasikan variabel **"encodeimg"**, nilai dari variabel ini kemudian akan dimasukkan ke dalam *array* **"imgEncodings"** dan program akan mengembalikkan nilai *array* tersebut.

- Bagian keempat

```
22  def MarkAttendence(name):
23      with open('attendence.csv', 'r+') as f:
24          myDatalist =  f.readlines()
25          nameList = []
26          for line in myDatalist :
27              entry = line.split(',')
28              nameList.append(entry[0])
29
30          if name not in nameList:
31              now = datetime.now()
32              timestr = now.strftime('%H:%M')
33              f.writelines(f'\n{name}, {timestr}')
```

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **MarkAttandance()** yang memiliki argumen **"name"**. Fungsi ini nantinya digunakan untuk mengambil nama dari foto tersebut dan menaruhnya di dalam *file spreadsheet.* Oleh karena itu nama pada setiap foto harus sama dengan nama dari orangnya. Fungsi ini akan memeriksa jika nama dari orang tersebut belum ada di dalam *file spreadsheet*, maka program akan mengambil waktu dengan fungsi **now()** pada modul *datetime* dan mencetak ulang nama dari penggunanya diselingi waktu *scan* berlangsung.

- Bagian kelima

```
path = 'student'
studentImg = []
studentName = []
myList = os.listdir(path)
for cl in myList :
    curimg = cv2.imread(f'{path}/{cl}')
    studentImg.append(curimg)
    studentName.append(os.path.splitext(cl)[0])
```

Kita mendeklarasikan variabel bernama **"path"** yang bernilai *string* **'student'**, maksunya adalah kita menunjukkan folder *database* **"student"** yang dimiliki. Kemudian kita mendeklarasikan dua buah *array* kosong **"studentImg"** dan **"studentName"**. Kita juga mendeklarasikan variabel **"mylist"** yang nilainya adalah *path* atau lokasi dari *database* **"student"** menggunakan *method* **listdir** yang dimiliki modul *os*. Pada setiap gambar yang ada di *database* folder tersebut, kita membaca lokasinya dan menyimpannya di dalam *array* **"studentImg"**. Serta kita juga membaca nama dari setiap gambar dan menyimpannya di dalam *array* **"studentName"**.

- Bagian keenam

```
def SystemAttandance() :
    EncodeList = findEncoding(studentImg)
    vid = cv2.VideoCapture(0)
    while True :
        success, frame = vid.read()
        Smaller_frames = cv2.resize(frame, (0,0), None, 0.25, 0.25)

        facesInFrame = face_rec.face_locations(Smaller_frames)
        encodeFacesInFrame = face_rec.face_encodings(Smaller_frames, facesInFrame)

        for encodeFace, faceloc in zip(encodeFacesInFrame, facesInFrame) :
            matches = face_rec.compare_faces(EncodeList, encodeFace)
            facedis = face_rec.face_distance(EncodeList, encodeFace)
            print(facedis)
            matchIndex = np.argmin(facedis)

            if matches[matchIndex] :
                name = studentName[matchIndex].upper()
                y1, x2, y2, x1 = faceloc
                y1, x2, y2, x1 = y1*4, x2*4, y2*4, x1*4
                cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 3)
                cv2.rectangle(frame, (x1, y2-25), (x2, y2), (0, 255, 0), cv2.FILLED)
                cv2.putText(frame, name, (x1+6, y2-6), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)
                MarkAttendence(name)
                cv2.putText(frame, "SELAMAT DATANG " + name, (300, 100), cv2.FONT_HERSHEY_COMPLEX, 1,
                (255, 255, 255), 2)

        cv2.imshow('Kehadiran Mahasiswa TRI',frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cv2.destroyAllWindows()
```

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **SystemAttandance()**. Fungsi ini menggunakan fungsi dan sintaks yang sebelumnya dideklarasikan dan fungsi ini juga adalah fungsi utama dari program ini. Fungsi ini menggunakan kamera *device* sebagai alat untuk melakukan *scan*. Program akan otomatis

mencari wajah yang tampil pada kamera dan melakuakan *scan* . Kemudian program menyocokkannya dengan foto yang ada di *database*. Jika mengalami kecocokan, maka *program* akan menyimpan nama dari orang pada foto ke dalam *spreadsheet* dan mencetak nama orang tersebut ke layar sebagai tanda bahwa orang tersebut berhasil melakukan presensi. Fungsi ini akan berakhir jika *user* menekan **"q"** pada *keyboard*, maksudnya jendela tempat *scan* ini berlangsung akan ter-*destroy* atau berhenti ditampilkan.

f. Pembuatan program modul fungsi_gui.py

Sintaks ini merupakan sebuah modul **fungsi_gui.py** yang disimpan di dalam paket (folder) **fungsi** yang dibuat. Modul ini berisikan fungsi-fungsi yang digunakan pada GUI nantinya.

- Bagian pertama

Sintaks ini berisikan *keyword* ***from*** dan ***import*** untuk mengimpor modul dan paket yang digunakan pada program modul ini. Kita mengimpor beberapa modul dan paket seperti *tkinter*, *shutil*, dan *os*. Kita menggunakan paket *tkinter* khususnya pada *method* **filedialog()** dan **askopenfilename()** untuk menampilkan jendela yang nantinya dapat membuka sebuah *file*, modul *shutil* digunakan untuk menyalin dan menghapus *file*, dan modul *os* digunakan untuk melakukan interaksi dengan sistem operasi yang digunakan secara langsung.

- Bagain kedua

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **foto()**. Fungsi ini nantinya digunakan untuk *user* dapat menginputkan foto dan secara otomatis program akan menyimpan foto tersebut di dalam *database* (folder) **"student"**. Kita mendeklarasikan sebuah variabel **"foto"** yang nilainya adalah gambar yang *user* inputkan dengan *method* **filedialog()** dan **askopenfilename()** milik *tkinter*. Kemudian dengan menggunakan *method* **move()** milik modul *shutil* yang berargumen nilai variabel **"foto"** dan *string* **"student"**, memiliki maksud bahwa gambar yang telah diinputkan akan otomatis dipindahkan ke dalam folder **"student"**. Arti **"student"** di sini adalah *path* atau jalur untuk memindahkan *file*-nya.

- Bagian ketiga

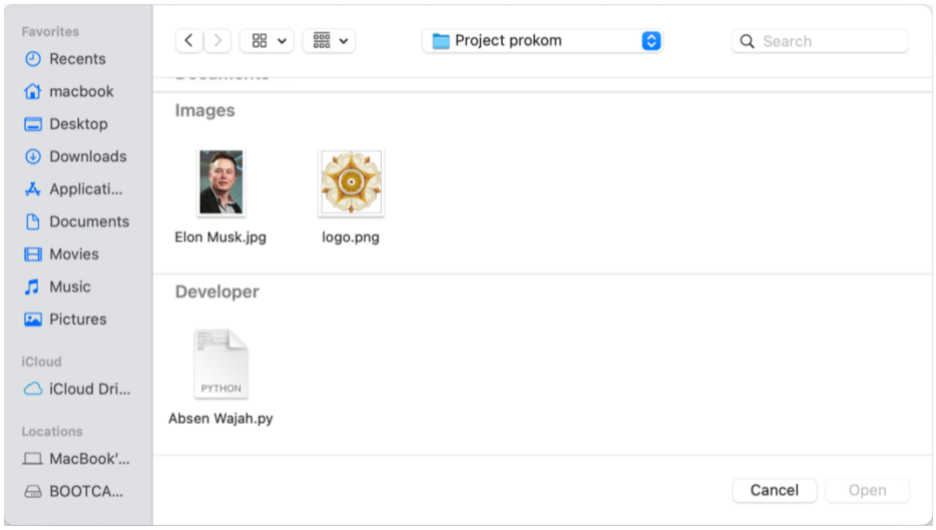
```
10  def hapus() :
11      foto = tk.filedialog.askopenfilename()
12      os.remove(foto)
```

Bagian ini merupakan pendeklarasian fungsi baru dengan *keyword* ***def*** bernama **hapus()**. Fungsi ini nantinya digunakan untuk *user* dapat menginputkan foto yang ada di dalam *database* (folder) **"student"** dan secara otomatis program akan menghapus foto tersebut. Kita mendeklarasikan sebuah variabel **"foto"** yang nilainya adalah gambar yang *user* inputkan dengan *method* **filedialog()** dan **askopenfilename()** milik *tkinter*. Kemudian dengan menggunakan *method* **remove()** milik modul *os* dengan argumen variabel **"foto"**, maksudnya adalah gambar yang telah diinputkan akan otomatis dihapus oleh komputer.

g. *Output*

Ini adalah *output* saat program dieksekusi. Dengan menggunakan paket *tkinter*, kita dapat menampilkan GUI ini. Tulisan "Universitas Gadjah Mada" merupakan *method* **tittle()** yang digunakan pada jendela **"layar"** *tkinter*. "Program Presensi Mahasiswa TRI" dan logo UGM yang ditampilkan ini menggunakan *method* **Label()** untuk menampilkan objek label pada jendela *tkinter*. Tiga buah tombol di jendela tersebut ditampilkan dengan menggunakan *method* **Button()** milik *tkinter*.

a. Ketika tombol "Input Foto" ditekan

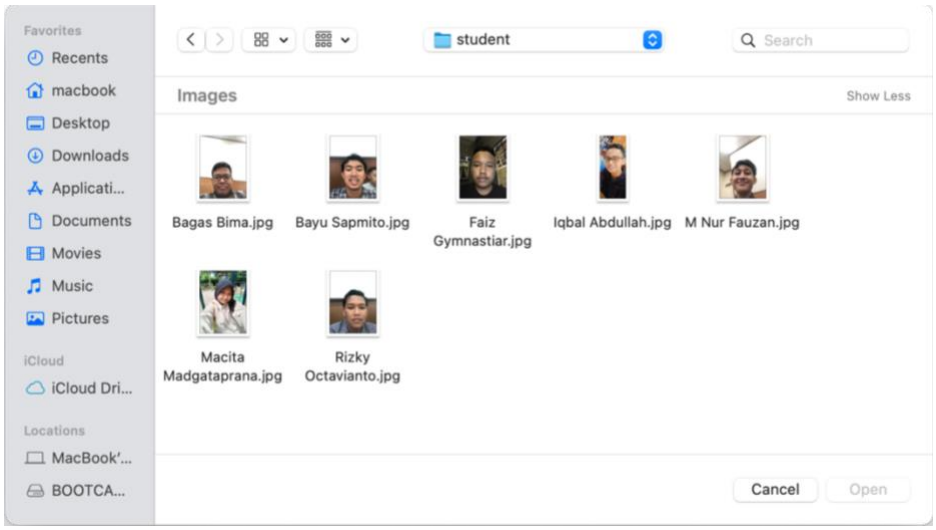


Saat ditekan, fungsi **foto()** akan menampilkan jendela yang nantinya dapat berinteraksi dengan *file-file* pada *device* yang digunakan. *User* dapat menginputkan foto mereka dan secara otomatis program akan menyimpannya di dalam *database* (folder) **"student"**.

b. Ketika tombol "Mulai Presensi" ditekan

Saat ditekan, fungsi **SystemAttandance()** akan berjalan dan program kehadiran otomatis berbasis wajah dapat dilakukan.

c. Ketika tombol "Hapus Foto" ditekan



Saat ditekan, fungsi **hapus()** akan menampilkan jendela yang nantinya dapat berinteraksi dengan *file-file* pada *device* yang digunakan. *User* dapat menginputkan foto mereka yang ada di dalam *database* (folder) **"student"** dan secara otomatis program akan menghapus foto tersebut secara permanen.

F. Kesimpulan

Hadirnya *smart attendance* berbasis wajah ini harapannya dapat mempermudah pekerjaan menjadi lebih produktif lagi dengan kemudahan

penggunaannya. Meskipun dinilai lebih akurat, nyatanya program kehadiran otomatis berbasis *scan* wajah ini tetap memiliki beberapa kekurangan seperti hasil *scan* yang kurang tepat akibat intensitas cahaya di lokasi, kualitas dari kamera, penggunaan aksesori wajah seperti kacamata, dan lain-lain. Selain itu jarak antara wajah dengan kamera serta kemiringan wajah juga berpengaruh dengan hasil yang diberikan nantinya. Kehadiran otomatis berbasis wajah ini dapat bermanfaat di berbagai sektor mulai dari instansi pendidikan, pemerintahan, karyawan, dan lain-lain. *Smart attendance* dengan *face recognition* secara *real time* ini layak untuk dikembangkan secara lanjut dan menggabungkannya dengan beberapa metode lain guna memperoleh tingkat akurasi yang lebih baik.

DAFTAR PUSTAKA

Anaconda.org. (n.d.). *Anaconda Documentation*. Retrieved from anaconda.com: https://docs.anaconda.com/navigator/index.html

Kurniawan, L. M. (2014). MetodeFace Recognition untuk Identifikasi Personil Berdasar Citra Wajah bagi Kebutuhan Presensi Online Universitas Negeri Semarang. *Scientific Journal of Informatics*, 210-220.

Reynaldo, R. R. (2019). Implementasi Metode Viola Jones Dan Convolutional Neural Network Untuk Pengenalan Ekspresi Wajah. *Thesis, Universitas Komputer Indonesia.* , 34.

Uswatun, L. (2021, Maret 9). *Python : Kenali NumPy Array dalam Python* . Retrieved from dqlab.id: https://www.dqlab.id/kenali-numpy-array-dalam-python

Saputra, Y. M. (2022). Modul Petemuan 11: Operasi Matriks pada Python . 1-6.

geeksforgeeks.org. (2022, Juni 16). *OS Module in Python with Examples*. Retrieved from geeksforgeeks.org: https://www.geeksforgeeks.org/os-module-python-examples/

geeksforgeeks.org. (2021, November 17). *Python datetime module* . Retrieved from geeksforgeeks.org: https://www.geeksforgeeks.org/python-datetime-module/

javatpoint.com. (n.d.). *Shutil Module in Python* . Retrieved from javatpoint.com: https://www.javatpoint.com/shutil-module-in-python

Tineges, R. (2021, Desember 8). *Library Python TKINTER untuk Membuat Aplikasi dengan Bahasa Pemrograman Berbasis GUI* . Retrieved from dqlab.id: https://dqlab.id/library-python-tkinter-untuk-membuat-aplikasi-dengan-bahasa-pemrograman-berbasis-gui