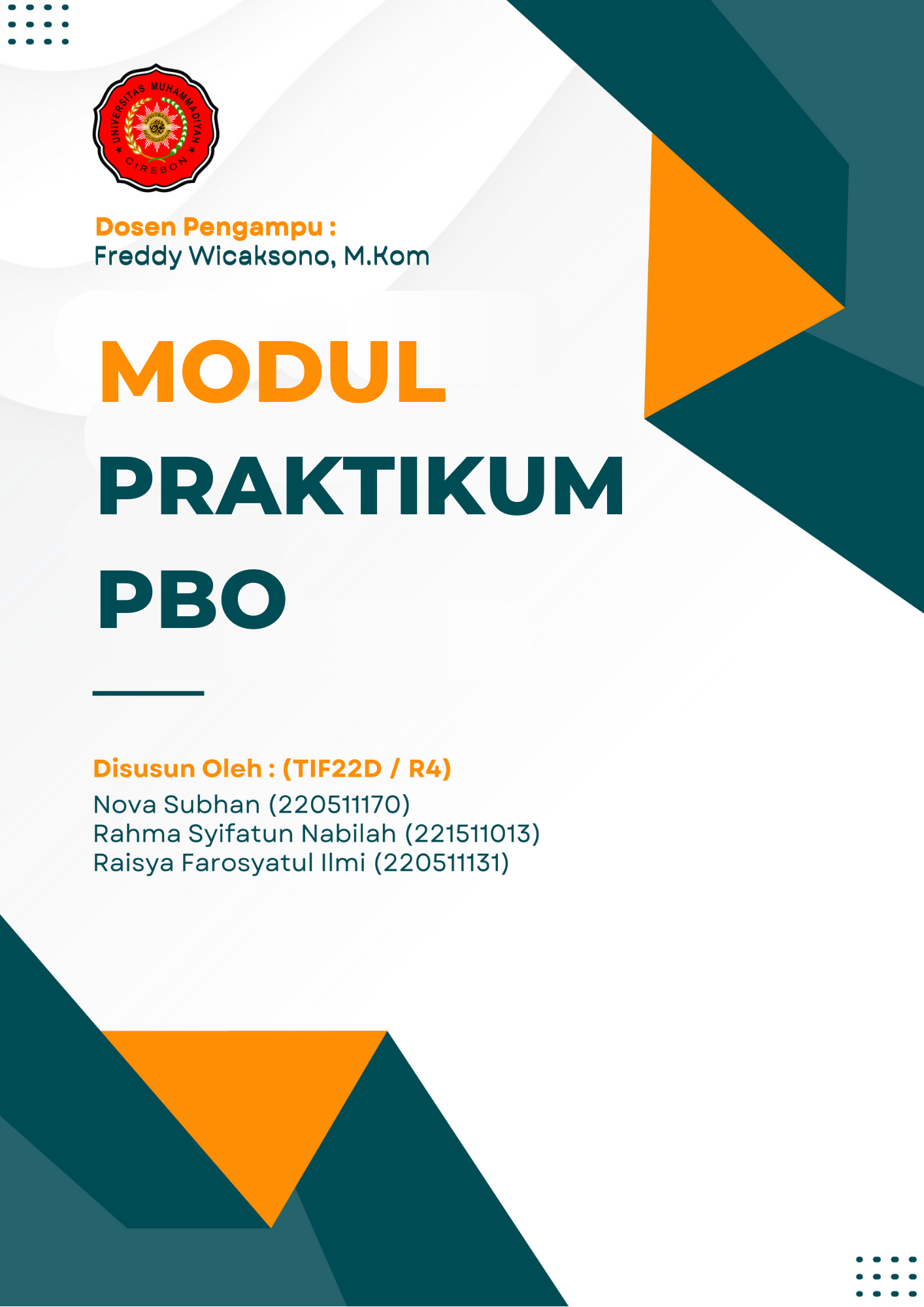
**,**



**KATA PENGANTAR**

Segala puji bagi Allah SWT yang telah memberikan kami kemudahan sehingga kami dapat menyelesaikan Modul ini dengan tepat waktu. Tanpa pertolongan-Nya tentunya kami tidak akan sanggup untuk menyelesaikan Modul ini dengan baik. Shalawat serta salam semoga terlimpah curahkan kepada baginda tercinta kita yaitu Nabi Muhammad SAW,yang kita nanti- natikan syafa'atnya di akhirat nanti.

**Modul ini membahas materi pertemuan bahasa Python dalam mata kuliah Praktikum PBO**. **Kami memandang PBO sebagai pilar utama dalam merancang aplikasi yang efektif secara fungsional dan mudah dipahami serta dikembangkan**

**Penulis menyadari bahwa Modul ini masih jauh dari kata sempurna dan masih banyak terdapat kesalahan serta kekurangan dalam penulisannya. Untuk itu, penulis berharap teman-teman semua dapat memaklumi dan memberikan saran dan kritik yang membangun**.Semoga makalah ini memberikan banyak manfaat bagi kita semua tentang pentingnya mempelajari bahasa pemrograman pyhton dalam perkembangan teknologi informasi.

Cirebon, 25 Januari 2023

Penulis

**DAFTAR ISI**

**Hal**

KATA PENGANTAR ii

DAFTAR ISI iii

BAB I PENDAHULUAN 1

1.1 Latar Belakang 1

1.2 Rumusan Masalah 1

1.3 Tujuan 1

BAB II PEMBAHASAN 2

2.1 Pertemuan 1 2

2.2 Pertemuan 2 4

2.3 Pertemuan 3 4

2.4 Pertemuan 4 7

2.5 Pertemuan 6 10

2.6 Pertemuan 7 13

2.7 Pertemuan 9 20

2.8 Pertemuan 10 25

2.9 Pertemuan 11 28

2.10 Pertemuan 12 35

2.11 Pertemuan 13 43

BAB III PENUTUP 50

3.1 Kesimpulan 50

**BAB I**

**PENDAHULUAN**

1.1Latar Belakang

Bahasa pemrograman Python diciptakan oleh Guido van Rossum, seorang programmer berkebangsaan Belanda, pada tahun 1989. Awalnya, Python dikembangkan di Centrum Wiskunde & Informatica (CWI), sebuah lembaga penelitian.Guido van Rossum ingin menciptakan bahasa pemrograman yang serupa dengan ABC, tetapi lebih kuat dan dapat digunakan untuk berbagai keperluan. Python pun dikembangkan dengan fokus pada kemudahan penggunaan, keterbacaan kode, dan produktivitas.

Python secara resmi dirilis kepada publik pada tahun 1991. Sejak saat itu, Python telah menjadi salah satu bahasa pemrograman yang paling populer di dunia.Python adalah bahasa pemrograman yang serbaguna dan dapat digunakan untuk berbagai keperluan. Oleh karena itu, Python merupakan bahasa pemrograman yang penting untuk dipelajari oleh siapa saja yang ingin berkarir di bidang teknologi informasi.

1.2 Rumusan Masalah

* Bagaimana cara menggunakan kontrol aliran dalam bahasa pemrograman Python?
* Apa saja tipe data yang tersedia dalam bahasa pemrograman Python?
* Bagaimana cara menggunakan operator dalam bahasa pemrograman Python?

1.3 Tujuan  
Tujuan mempelajari bahasa pemrograman python adalah untuk memberikan pemahaman dasar tentang bahasa pemrograman Python kepada siapapum yang mempelajarinya. Diharapkan dapat memahami konsep-konsep dasar Python, seperti sintaks, tipe data, variabel, operator, kontrol flow, dan fungsi.

**BAB II**

**PEMBAHASAN**

**2.1 PERTEMUAN 1**

Pertemuan 1 adalah membahas tentang rumus untuk menghitung volume dan luas permukaan beberapa bangun ruang yang umum, yaitu:

**1. Balok:**

Volume (V) = panjang × lebar × tinggi

Luas Permukaan (LP) = 2 × (panjang × lebar + panjang × tinggi + lebar × tinggi)

**2. Bola:**

Volume (V) = (4/3) × π × r³ (r adalah jari-jari bola)

Luas Permukaan (LP) = 4 × π × r²

**3. Kerucut:**

Volume (V) = (1/3) × π × r² × tinggi (r adalah jari-jari kerucut)

Luas Permukaan (LP) = π × r × (r + s) (s adalah panjang garis pelukis)

**4. Kubus:**

Volume (V) = sisi³ (sisi adalah panjang sisi kubus)

Luas Permukaan (LP) = 6 × sisi²

**5. Limas Segiempat:**

Volume (V) = (1/3) × luas alas × tinggi

Luas Permukaan (LP) = luas alas + 4 × (1/2) × keliling alas × tinggi

**6. Limas Segitiga:**

Volume (V) = (1/3) × luas alas × tinggi

Luas Permukaan (LP) = luas alas + 3 × (1/2) × keliling alas × tinggi

**7. Persegi Panjang:**

Luas (L) = panjang × lebar

Keliling (K) = 2 × (panjang + lebar)

**8. Prisma Segitiga:**

Volume (V) = (1/2) × luas alas × tinggi prisma

Luas Permukaan (LP) = luas alas + 3 × (1/2) × keliling alas × tinggi prisma

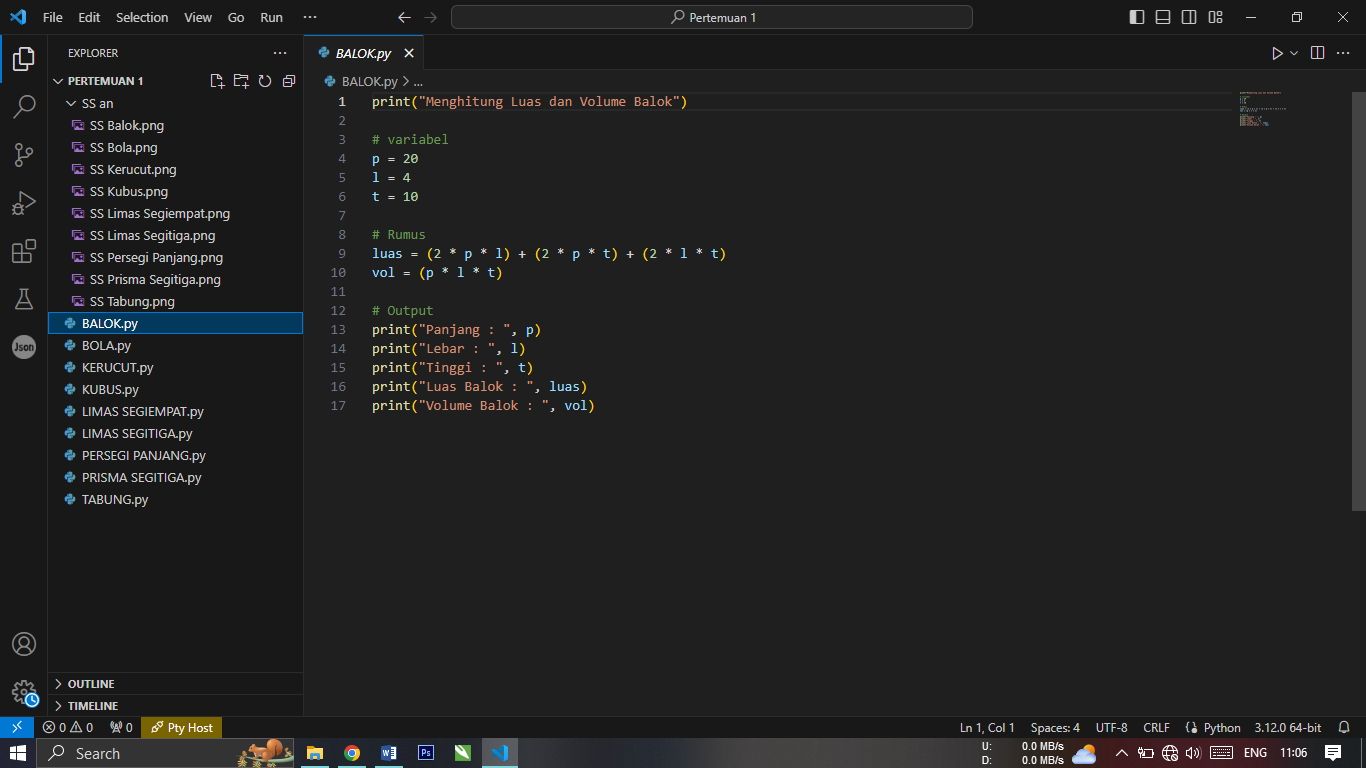
**9. Tabung:**

Volume (V) = π × r² × tinggi

Luas Permukaan (LP) = 2 × π × r × (r + tinggi)

Catatan: π (pi) adalah konstanta matematika yang kurang lebih setara dengan 3.14159.

**Dan Berikut Contoh Source Codenya :**



**2.2 Pertemuan 2**

IF adalah sebuah konsep dalam pemrograman yang digunakan untuk pengambilan keputusan atau kontrol aliran program. Konsep ini memungkinkan sebuah program untuk mengevaluasi apakah kondisi benar atau salah.

Definisi Looping pada Python Secara umum, looping atau perulangan pada Python adalah instruksi kode program yang dieksekusi berulang kali.

**2.3 Pertemuan 3**

Pertemuan 3 adalah lanjutan dari pertemuan 1 dengan tambahan pengaplikasian menggunakan Windows yaitu Tkinter. Dengan program Tkinter kita bisa membuat program tersebut menjadi tampilan Windows.

`Tkinter` adalah modul dalam bahasa pemrograman Python yang menyediakan alat untuk membuat antarmuka grafis pengguna (GUI). Nama "tkinter" sendiri adalah singkatan dari "Tk Interface." Tkinter menggunakan toolkit Tk, yang adalah toolkit GUI yang awalnya dikembangkan untuk bahasa pemrograman Tcl/Tk. Tkinter telah menjadi bagian integral dari distribusi standar Python, sehingga Anda tidak perlu menginstal paket tambahan untuk menggunakannya.Dengan menggunakan tkinter, Anda dapat membuat berbagai elemen GUI, seperti jendela, tombol, kotak teks, daftar, dan banyak lagi. Tkinter menyediakan cara yang relatif mudah untuk membuat aplikasi dengan antarmuka grafis sederhana dan menengah.

Berikut adalah contoh sederhana penggunaan tkinter untuk membuat jendela kosong:

import tkinter as tk

# Membuat objek jendela

jendela = tk.Tk()

# Menambahkan judul jendela

jendela.title("Contoh Tkinter")

# Menampilkan jendela

jendela.mainloop()

**Berikut adalah program balok menggunakan Tkinter :**

import tkinter as tk

from tkinter import Frame,Label,Entry,Button,END, W

def hitung\_luas():

    p = float(txtpanjang.get())

    l = float(txtlebar.get())

    t = float(txttinggi.get())

    L = round(2 \* p \* l) + (2 \* p \* t) + (2 \* l \* t)

    txtLuas.delete(0,END)

    txtLuas.insert(END,L)

def hitung\_volume():

    p = float(txtpanjang.get())

    l = float(txtlebar.get())

    t = float(txttinggi.get())

    v = round(p \* l \* t)

    txtVolume.delete(0,END)

    txtVolume.insert(END,v)

def hitung():

    hitung\_luas()

    hitung\_volume()

# Buat Objek Tk Inter

app = tk.Tk()

# Tambahkan Judul

app.title("Kalkulator Luas Dan Volume Balok by NohanX7")

# Windows

frame = Frame(app)

frame.pack(padx=20, pady=20)

# Label Nama

nama= Label(frame, text="Nova Subhan / 220511170 / R4")

nama.grid(row=0, column=0, sticky=W, padx=5, pady=5)

# Label Panjang

panjang= Label(frame, text="Panjang :")

panjang.grid(row=1, column=0, sticky=W, padx=5, pady=5)

# Label Lebar

lebar= Label(frame, text="Lebar :")

lebar.grid(row=2, column=0, sticky=W, padx=5, pady=5)

# Label Tinggi

tinggi= Label(frame, text="Tinggi :")

tinggi.grid(row=3, column=0, sticky=W, padx=5, pady=5)

# Textbox Panjang

txtpanjang = Entry(frame)

txtpanjang.grid(row=1, column=1)

# Textbox Lebar

txtlebar = Entry(frame)

txtlebar.grid(row=2, column=1)

# Textbox Tinggi

txttinggi = Entry(frame)

txttinggi.grid(row=3, column=1)

# Button

hitung\_button = Button(frame, text="Hitung", command=hitung)

hitung\_button.grid(row=4, column=1, sticky=W, padx=5, pady=5)

# Output Label Luas

luas = Label(frame, text="Luas : ")

luas.grid(row=5, column=0, sticky=W, padx=5, pady=5)

# Output Label Volume

volume = Label(frame, text="Volume : ")

volume.grid(row=6, column=0, sticky=W, padx=5, pady=5)

# Output Textbox Luas

txtLuas = Entry(frame)

txtLuas.grid(row=5, column=1, sticky=W, padx=5, pady=5)

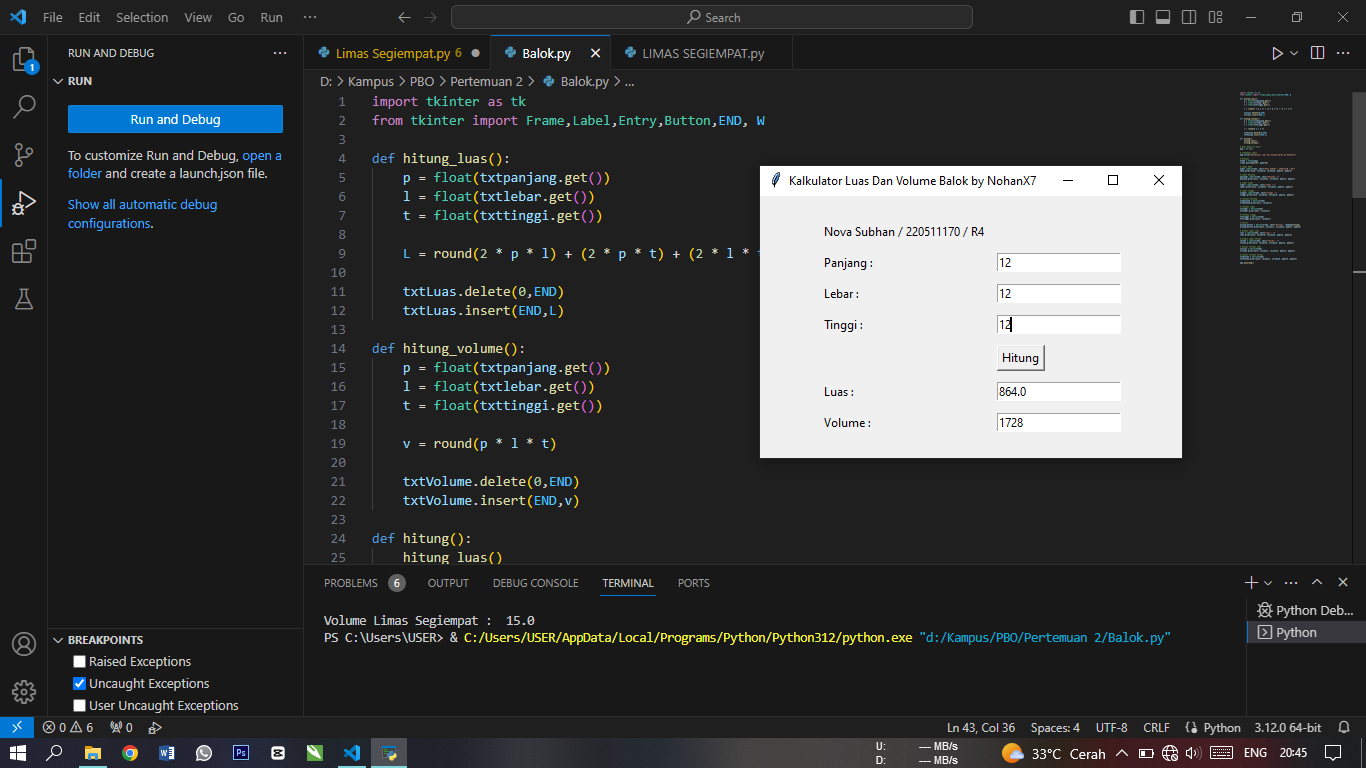
# Output Textbox Volume

txtVolume = Entry(frame)

txtVolume.grid(row=6, column=1, sticky=W, padx=5, pady=5)

app.mainloop()

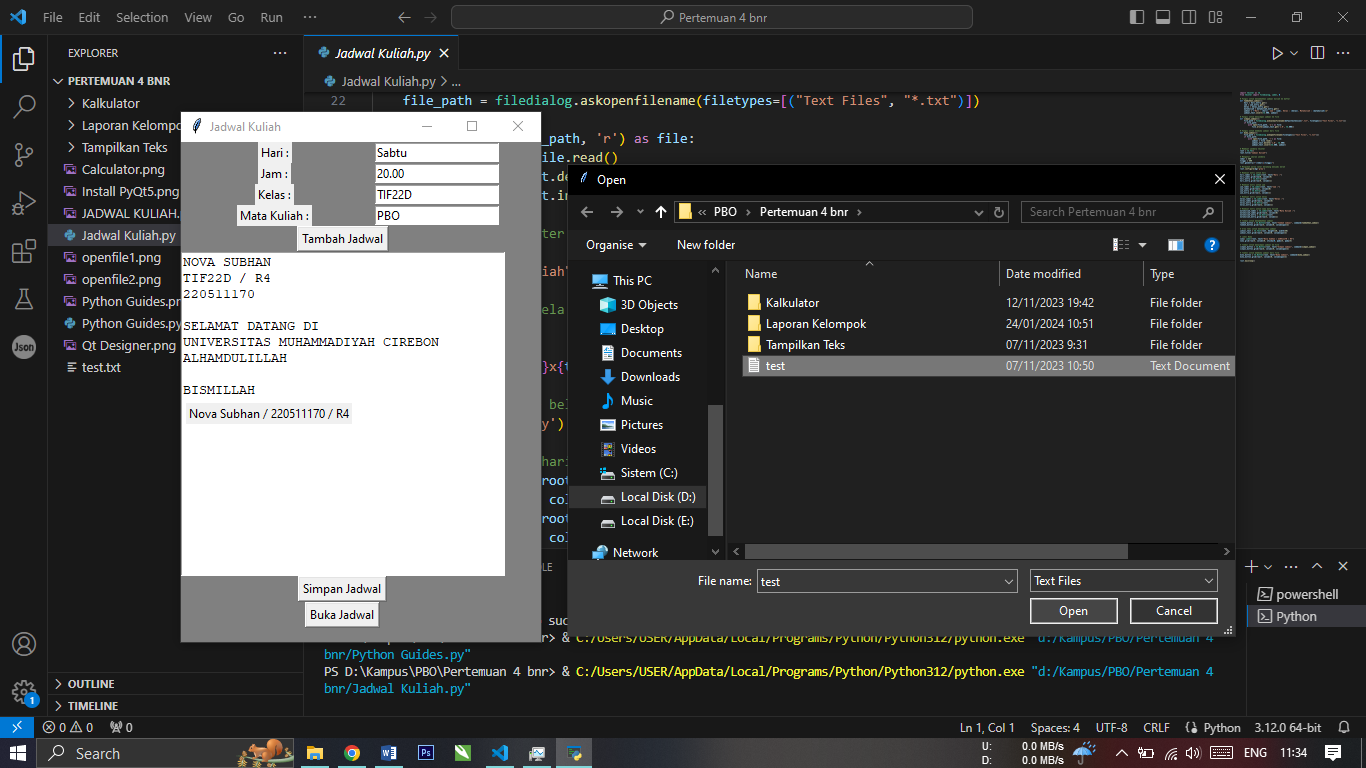
**Berikut adalah hasil output program balok menggunakan Tkinter :**



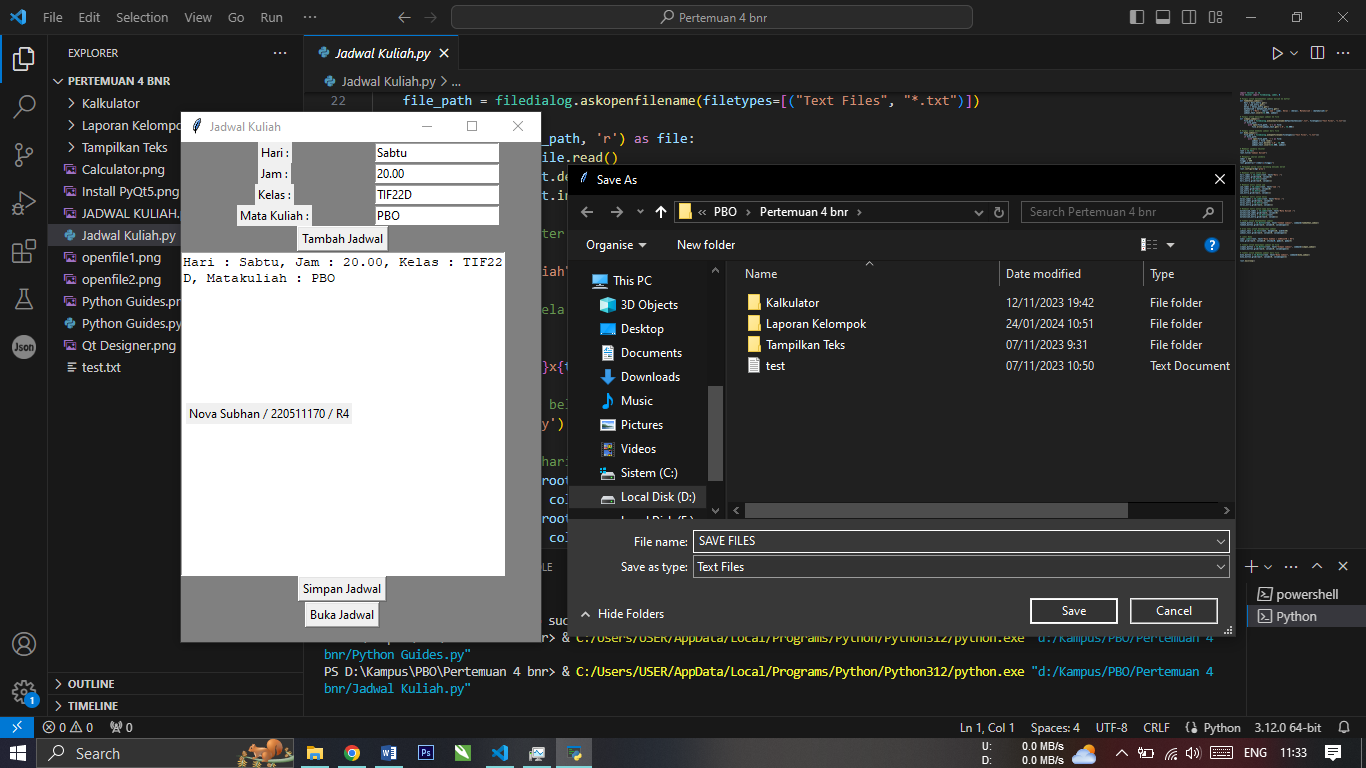
**2.4 Pertemuan 4**

Pertemuan 4 adalah membahas tentang Python Guides. Pertemuan ini menjelaskan tentang cara Open & Save Files dalam bentuk format .txt.

**Berikut adalah hasil output untuk Open Files :**



**Berikut adalah hasil output untuk Save Files :**



**Berikut adalah Program untuk Python Guides :**

import tkinter as tk

from tkinter import filedialog, Label, W

# Fungsi untuk menambahkan jadwal kuliah ke daftar

def tambahkan\_jadwal():

    hari = hari\_entry.get()

    jam = jam\_entry.get()

    kelas = kelas\_entry.get()

    matakuliah = matakuliah\_entry.get()

    jadwal = f"Hari : {hari}, Jam : {jam}, Kelas : {kelas}, Matakuliah : {matakuliah}\n"

    jadwal\_text.insert(tk.END, jadwal)

# Fungsi untuk menyimpan jadwal ke file

def simpan\_jadwal():

    file\_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text Files", "\*.txt")])

    if file\_path:

        with open(file\_path, 'w') as file:

            file.write(jadwal\_text.get('1.0', tk.END))

# Fungsi untuk membuka jadwal dari file

def buka\_jadwal():

    file\_path = filedialog.askopenfilename(filetypes=[("Text Files", "\*.txt")])

    if file\_path:

        with open(file\_path, 'r') as file:

            jadwal = file.read()

            jadwal\_text.delete('1.0', tk.END)

            jadwal\_text.insert(tk.END, jadwal)

# Membuat jendela tkinter

root = tk.Tk()

root.title("Jadwal Kuliah")

# Mengatur ukuran jendela

lebar = 360

tinggi = 500

root.geometry(f"{lebar}x{tinggi}")

# Mengubah warna latar belakang menjadi merah

root.configure(bg='grey')

# Membuat entri untuk hari

hari\_label = tk.Label(root, text="Hari :")

hari\_label.grid(row=0, column=0)

hari\_entry = tk.Entry(root)

hari\_entry.grid(row=0, column=1)

# Membuat entri untuk jam

jam\_label = tk.Label(root, text="Jam :")

jam\_label.grid(row=1, column=0)

jam\_entry = tk.Entry(root)

jam\_entry.grid(row=1, column=1)

# Membuat entri untuk kelas

kelas\_label = tk.Label(root, text="Kelas :")

kelas\_label.grid(row=2, column=0)

kelas\_entry = tk.Entry(root)

kelas\_entry.grid(row=2, column=1)

# Membuat entri untuk nama mata kuliah

matakuliah\_label = tk.Label(root, text="Mata Kuliah :")

matakuliah\_label.grid(row=3, column=0)

matakuliah\_entry = tk.Entry(root)

matakuliah\_entry.grid(row=3, column=1)

# Tombol untuk menambahkan jadwal

tambah\_button = tk.Button(root, text="Tambah Jadwal", command=tambahkan\_jadwal)

tambah\_button.grid(row=4, column=0, columnspan=2)

# Area teks untuk menampilkan jadwal

jadwal\_text = tk.Text(root, height=20, width=40)

jadwal\_text.grid(row=5, column=0, columnspan=2)

# Label Nama

nama= Label(root, text="Nova Subhan / 220511170 / R4")

nama.grid(row=5, column=0, sticky=W, padx=5, pady=5)

# Tombol untuk menyimpan jadwal ke file

simpan\_button = tk.Button(root, text="Simpan Jadwal", command=simpan\_jadwal)

simpan\_button.grid(row=6, column=0, columnspan=2)

# Tombol untuk membuka jadwal dari file

buka\_button = tk.Button(root, text="Buka Jadwal", command=buka\_jadwal)

buka\_button.grid(row=7, column=0, columnspan=2)

root.mainloop()

**Pertemuan 5 Kuis**

**2.5 Pertemuan 6**

Pertemuan 6 adalah membahas tentang membuat menu dashboard dan translate"Dashboard" dalam konteks pemrograman Python bisa merujuk pada antarmuka pengguna grafis (GUI) atau tampilan visual yang menyajikan informasi secara ringkas dan mudah dipahami. Terdapat beberapa pustaka dan alat di Python yang memungkinkan pembuatan dashboard dengan mudah.

**Berikut adalah Program dari Dashboard :**

import tkinter as tk

from tkinter import Menu

from FrmPersegi import \*

from FrmSegitiga import \*

from FrmLingkaran import \*

from FrmBalok import \*

from FrmKerucut import \*

from FrmKubus import \*

from FrmLimasSegiempat import \*

from FrmLimasSegitiga import \*

from FrmPrismaSegitiga import \*

from FrmTabung import \*

from FrmBola import \*

# root window

root = tk.Tk()

root.title('Menu Demo by Nova Subhan / 220511170 / R4/D')

#root.attributes('-fullscreen', True)

root.geometry("900x400")

# create a menubar

menubar = Menu(root)

root.config(menu=menubar)

# create a menu

file\_menu = Menu(menubar)

app\_menu = Menu(menubar)

data\_menu = Menu(menubar)

# add a menu item to the menu

file\_menu.add\_command(

    label='File Open', command=root.destroy

)

file\_menu.add\_command(

    label='Exit', command=root.destroy

)

app\_menu.add\_command(

    label='App Persegi', command= lambda: new\_window("Luas Persegi", FrmPersegi)

)

app\_menu.add\_command(

    label='App Segitiga', command= lambda: new\_window("Luas Segitiga", FrmSegitiga)

)

app\_menu.add\_command(

    label='App Lingkaran', command= lambda: new\_window("Luas Lingkaran", FrmLingkaran)

)

app\_menu.add\_command(

    label='App Balok', command= lambda: new\_window("Luas Balok", FrmBalok)

)

app\_menu.add\_command(

    label='App Kerucut', command= lambda: new\_window("Luas Kerucut", FrmKerucut)

)

app\_menu.add\_command(

    label='App Kubus', command= lambda: new\_window("Luas Kubus", FrmKubus)

)

app\_menu.add\_command(

    label='App Limas Segiempat', command= lambda: new\_window("Luas Limas Segiempat", FrmLimasSegiempat)

)

app\_menu.add\_command(

    label='App Limas Segitiga', command= lambda: new\_window("Luas Limas Segitiga", FrmLimasSegitiga)

)

app\_menu.add\_command(

    label='App Prisma Segitiga', command= lambda: new\_window("Luas Prisma Segitiga", FrmPrismaSegitiga)

)

app\_menu.add\_command(

    label='App Tabung', command= lambda: new\_window("Luas Tabung", FrmTabung)

)

app\_menu.add\_command(

    label='App Bola', command= lambda: new\_window("Luas Bola", FrmBola)

)

def new\_window( number, \_class):

    new = tk.Toplevel()

    new.transient()

    new.grab\_set()

    \_class(new, number)

# add the File menu to the menubar

menubar.add\_cascade(

    label="File", menu=file\_menu

)

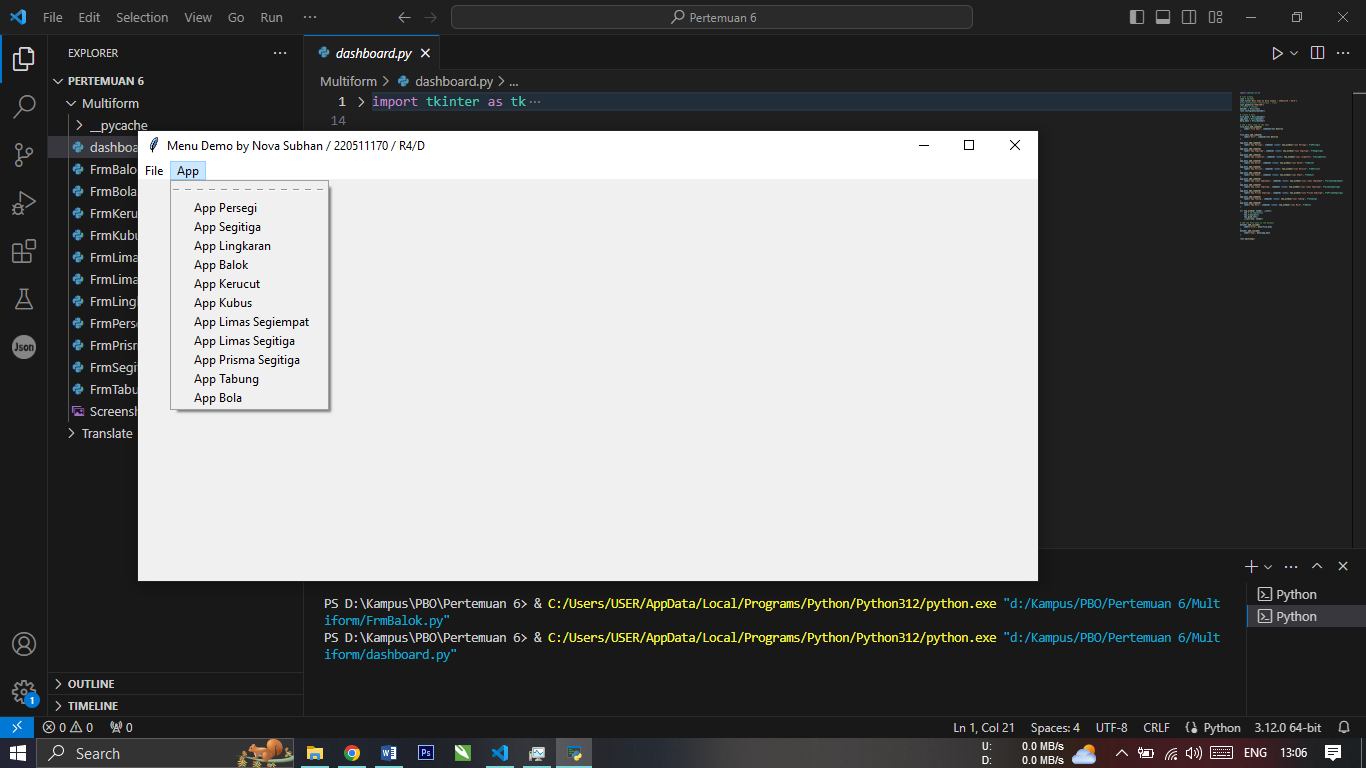
menubar.add\_cascade(

    label="App", menu=app\_menu

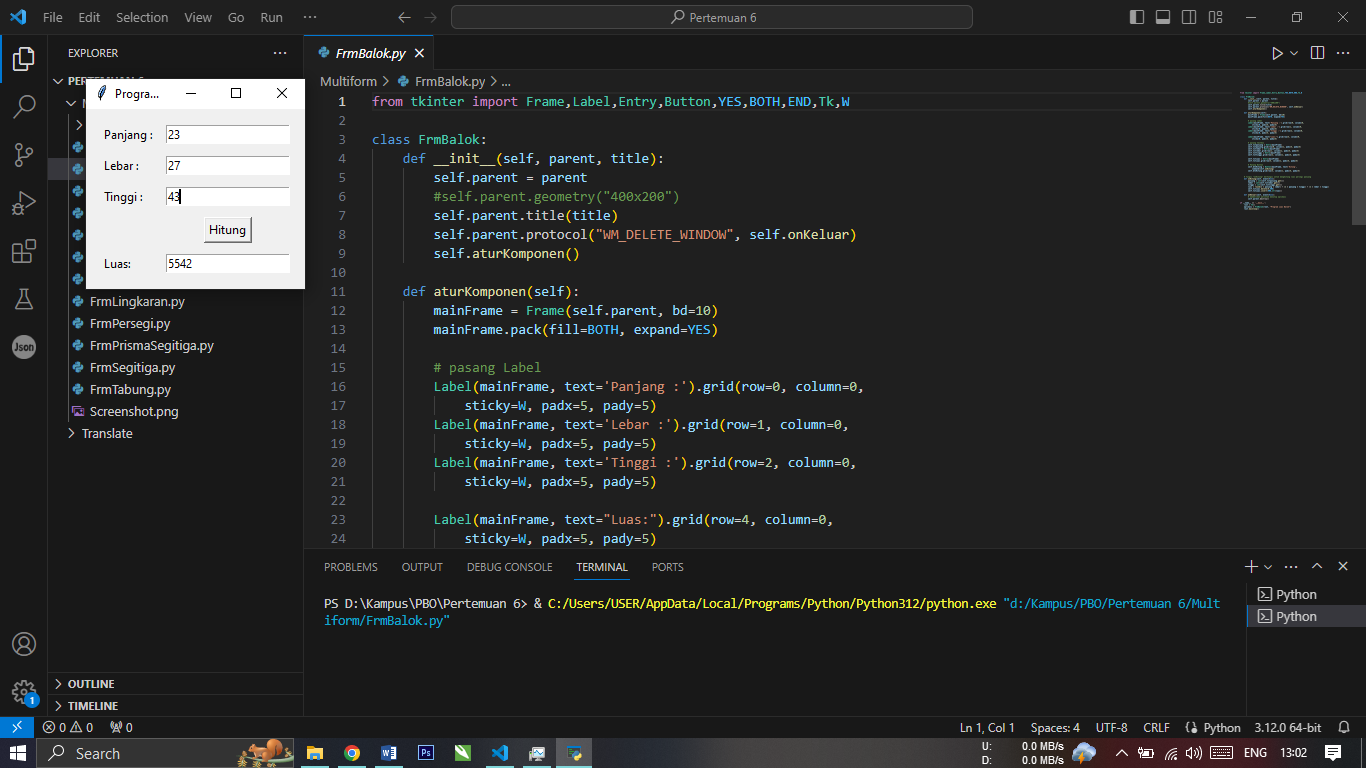
)

root.mainloop()

**Berikut adalah hasil output untuk Dashboard :**



**Berikut adalah salah satu contoh dari program FrmBalok :**



**2.6 Pertemuan 7**

Pertemuan 7 adalah membahas tentang 4 Program yaitu :

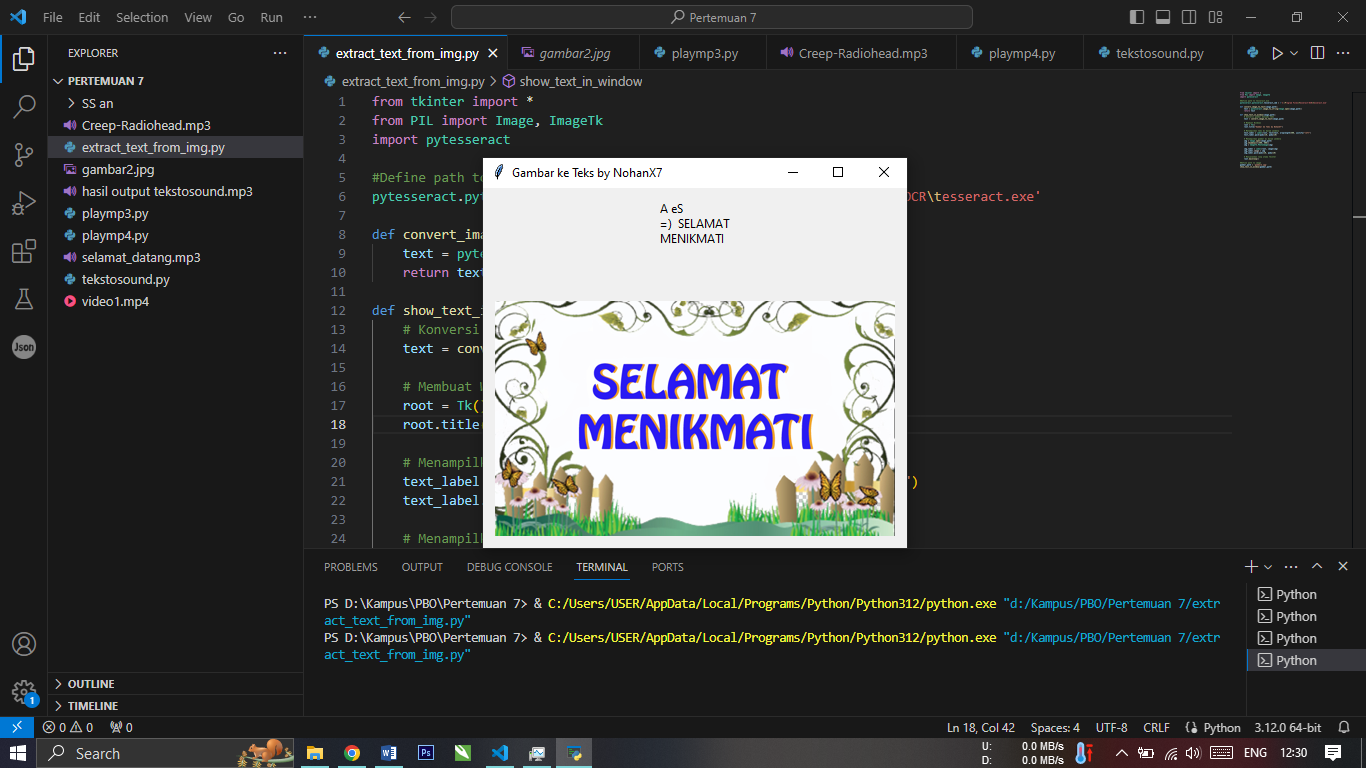
* Extract\_Text\_From\_Img
* Play Mp3
* Play Mp4
* Teks To Sound
* **Extract\_Text\_From\_Img**

Extract\_Text\_From\_Img adalah program untuk mengekstrak teks yang ada didalam gambar menjadi teks tulisan.

**Berikut adalah teks yang ada di gambar :**



**Berikut adalah Hasil Output Teks From Image :**



**Dan ini adalah hasil Programnya :**

from tkinter import \*

from PIL import Image, ImageTk

import pytesseract

#Define path to tessaract.exe

pytesseract.pytesseract.tesseract\_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

def convert\_image\_to\_text(image\_path):

    text = pytesseract.image\_to\_string(Image.open(image\_path))

    return text

def show\_text\_in\_window(image\_path):

    # Konversi gambar menjadi teks

    text = convert\_image\_to\_text(image\_path)

    # Membuat Windows

    root = Tk()

    root.title("Gambar ke Teks by NohanX7")

    # Menampilkan teks di dalam jendela

    text\_label = Label(root, text=text, wraplength=400, justify="left")

    text\_label.pack(padx=10, pady=10)

    # Menampilkan gambar di dalam jendela

    img = Image.open(image\_path)

    img.thumbnail((400, 400))

    img = ImageTk.PhotoImage(img)

    img\_label = Label(root, image=img)

    img\_label.image = img

    img\_label.pack(padx=10, pady=10)

    # Menjalankan loop utama Tkinter

    root.mainloop()

#Define path to image

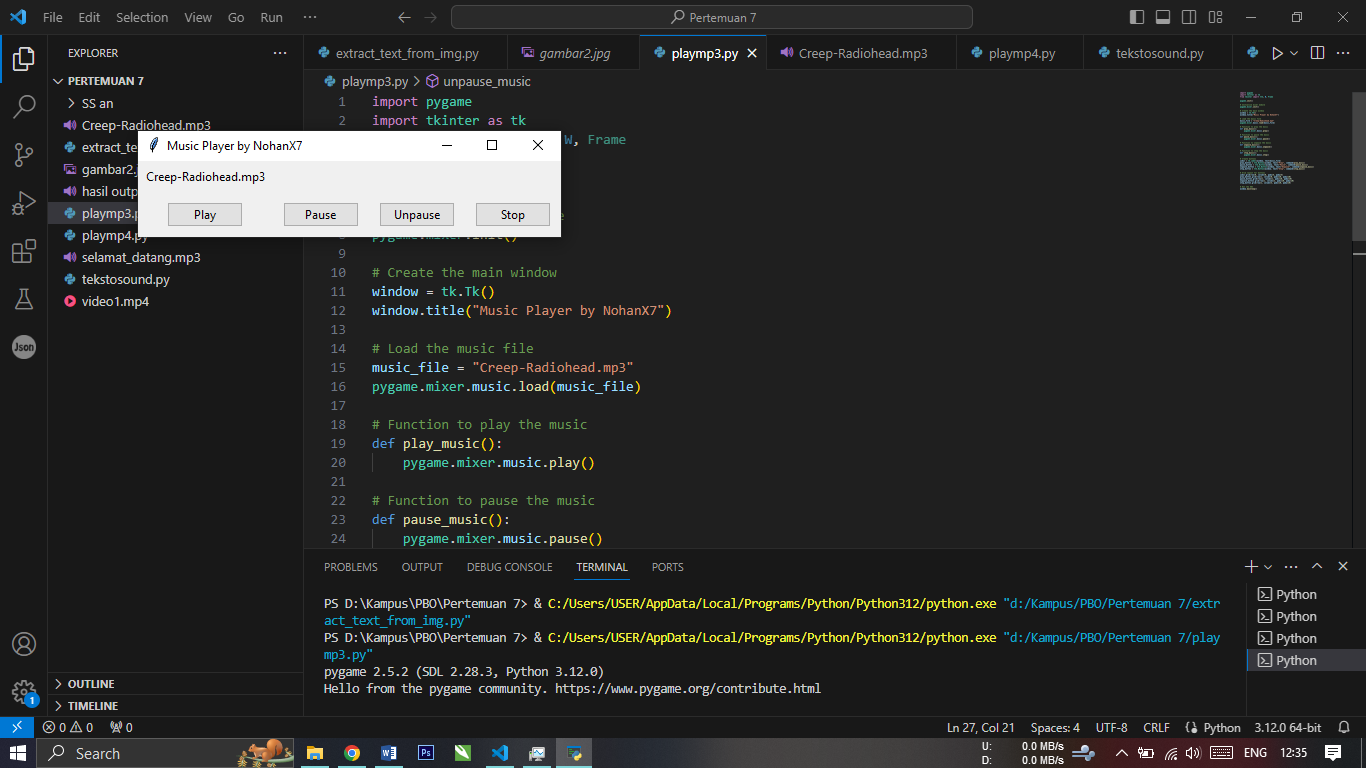
gambar\_path = 'gambar2.jpg'

show\_text\_in\_window(gambar\_path)

* **Program untuk memutar Mp3**

Untuk memutar file mp3 dipython,anda dapat menggunakan pustaka pygame.

**Berikut adalah Hasil Output Play Mp3:**



**Berikut adalah tampilan programnya:**

import pygame

import tkinter as tk

from tkinter import ttk, W, Frame

pygame.init()

# Initialize mixer module

pygame.mixer.init()

# Create the main window

window = tk.Tk()

window.title("Music Player by NohanX7")

# Load the music file

music\_file = "Creep-Radiohead.mp3"

pygame.mixer.music.load(music\_file)

# Function to play the music

def play\_music():

    pygame.mixer.music.play()

# Function to pause the music

def pause\_music():

    pygame.mixer.music.pause()

# Function to unpause the music

def unpause\_music():

    pygame.mixer.music.unpause()

# Function to stop the music

def stop\_music():

    pygame.mixer.music.stop()

# Create buttons

judul = tk.Label(window, text=music\_file)

play\_button = ttk.Button(window, text="Play", command=play\_music)

pause\_button = ttk.Button(window, text="Pause", command=pause\_music)

unpause\_button = ttk.Button(window, text="Unpause", command=unpause\_music)

stop\_button = ttk.Button(window, text="Stop", command=stop\_music)

# Grid layout for buttons

judul.grid(row=0, column=0, padx=5, pady=5)

play\_button.grid(row=1, column=0, padx=10, pady=10)

pause\_button.grid(row=1, column=1, padx=10, pady=10)

unpause\_button.grid(row=1, column=2, padx=10, pady=10)

stop\_button.grid(row=1, column=3, padx=10, pady=10)

# Run the GUI

window.mainloop()

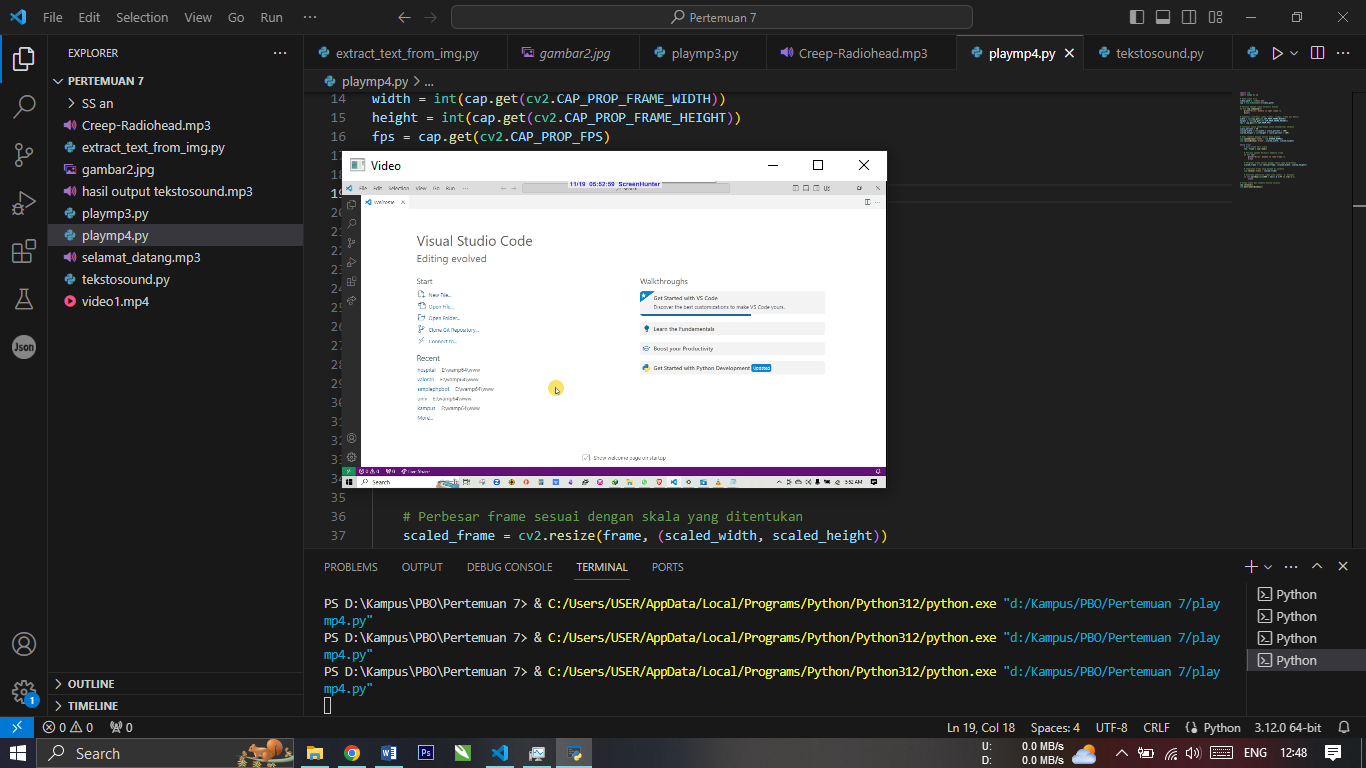
* **Program untuk memainkan Mp4**

Untuk memainkan file video MP4 di Python, Anda dapat menggunakan pustaka seperti OpenCV atau Pygame.

Pastikan untuk mengganti `'nama\_file\_video.mp4'` dengan nama sebenarnya dari file video MP4 yang ingin Anda putar. Jalankan skrip tersebut, dan jendela pemutaran video akan muncul, memainkan video hingga Anda menekan tombol 'q' untuk keluar.

Perlu diingat bahwa untuk menjalankan skrip ini, Anda memerlukan instalasi OpenCV, dan Anda juga memerlukan code yang tepat untuk format video yang Anda gunakan. Jika Anda mengalami kesulitan, pastikan code video yang diperlukan terpasang di sistem Anda.

**Berikut adalah Hasil Output Play Mp4:**



**Berikut adalah tampilan programnya:**

import cv2

import numpy as np

# Buka video file

video\_path = 'video1.mp4'

cap = cv2.VideoCapture(video\_path)

# Periksa apakah video berhasil dibuka

if not cap.isOpened():

    print("Error: Unable to open video.")

    exit()

# Dapatkan informasi video (lebar, tinggi, frame per detik)

width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))

height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))

fps = cap.get(cv2.CAP\_PROP\_FPS)

# Tentukan skala pengurangan untuk menampilkan jendela

scale\_percent = 40

scaled\_width = int(width \* scale\_percent / 100)

scaled\_height = int(height \* scale\_percent / 100)

# Buat jendela dengan ukuran yang diubah

cv2.namedWindow('Video', cv2.WINDOW\_NORMAL)

cv2.resizeWindow('Video', scaled\_width, scaled\_height)

while True:

    # Baca frame dari video

    ret, frame = cap.read()

    # Periksa apakah berhasil membaca frame

    if not ret:

        print("Error: Unable to read frame.")

        break

    # Perbesar frame sesuai dengan skala yang ditentukan

    scaled\_frame = cv2.resize(frame, (scaled\_width, scaled\_height))

    # Tampilkan frame yang diubah di jendela

    cv2.imshow('Video', scaled\_frame)

    # Hentikan pemutaran video jika tombol 'q' ditekan

    if cv2.waitKey(int(1000 / fps)) & 0xFF == ord('q'):

        break

# Tutup video dan jendela ketika selesai

cap.release()

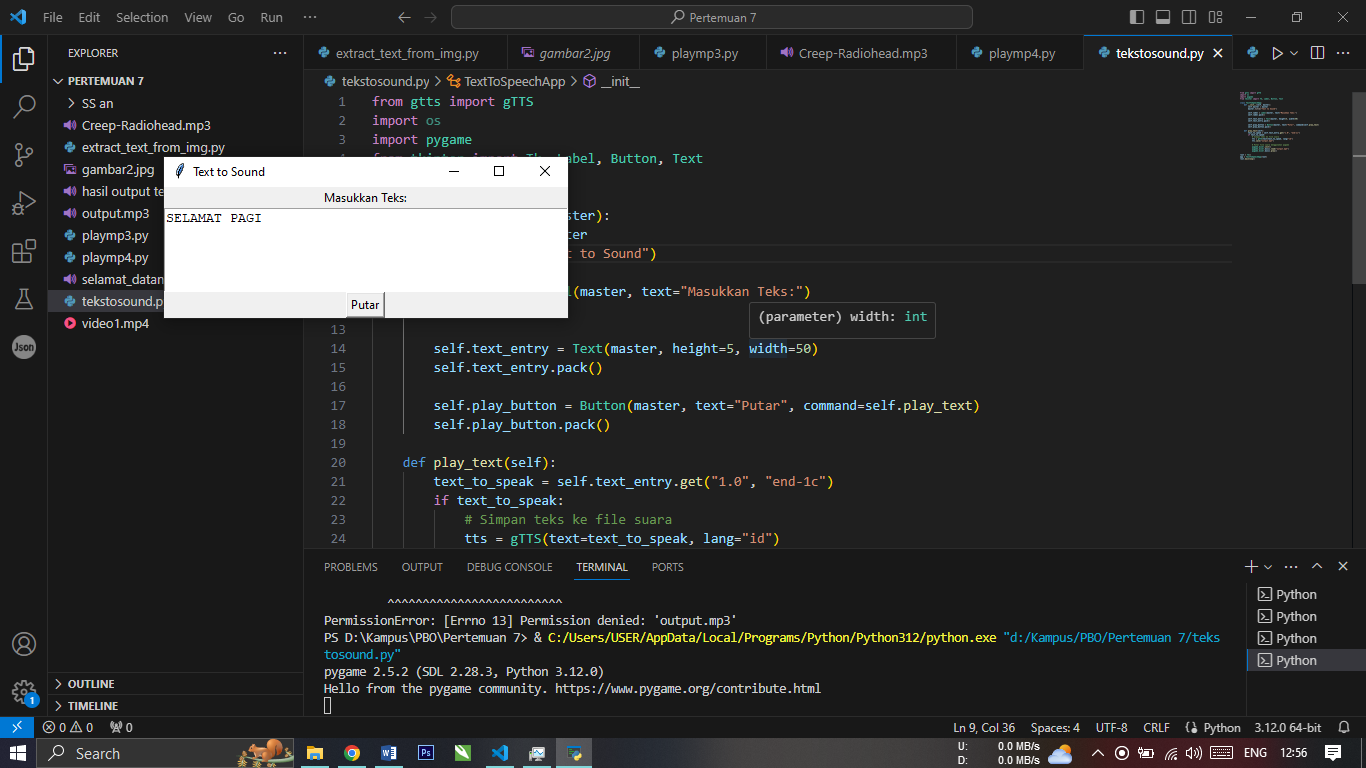
cv2.destroyAllWindows()

* **Program untuk mengekstrak teks to sound**

Untuk membuat program yang mengekstrak teks dan mengonversinya ke suara (text-to-speech) di Python, Anda dapat menggunakan pustaka Text-to-Speech seperti `gTTS` (Google Text-to-Speech).

Anda dapat menyesuaikan teks yang ingin diubah menjadi suara sesuai kebutuhan Anda. Untuk gTTS, perhatikan bahwa suara akan disimpan sebagai file MP3 ('output.mp3' dalam contoh tersebut) dan diputar menggunakan perintah sistem. Untuk pyttsx3, suara akan langsung diucapkan.

Pilih salah satu metode sesuai dengan kebutuhan dan preferensi Anda. Pastikan Anda memiliki koneksi internet saat menggunakan gTTS karena gTTS memerlukan koneksi ke layanan Google untuk menghasilkan suara.



**Berikut adalah tampilan programnya:**

from gtts import gTTS

import os

import pygame

from tkinter import Tk, Label, Button, Text

class TextToSpeechApp:

    def \_\_init\_\_(self, master):

        self.master = master

        master.title("Text to Sound")

        self.label = Label(master, text="Masukkan Teks:")

        self.label.pack()

        self.text\_entry = Text(master, height=5, width=50)

        self.text\_entry.pack()

        self.play\_button = Button(master, text="Putar", command=self.play\_text)

        self.play\_button.pack()

    def play\_text(self):

        text\_to\_speak = self.text\_entry.get("1.0", "end-1c")

        if text\_to\_speak:

            # Simpan teks ke file suara

            tts = gTTS(text=text\_to\_speak, lang="id")

            tts.save("output.mp3")

            # Putar file suara menggunakan pygame

            pygame.mixer.init()

            pygame.mixer.music.load("output.mp3")

            pygame.mixer.music.play()

root = Tk()

app = TextToSpeechApp(root)

root.mainloop()

**Pertemuan 8 UTS**

**2.7 Pertemuan 9**

**Suhu opp,Terstruktur & tidak terstruktur**

Object-Oriented Programming (OOP) adalah paradigma pemrograman yang menggunakan objek sebagai unit dasar untuk membangun program. Dalam konteks OOP, ada dua pendekatan yang berbeda:

1. OOP Modular,mengacu pada pendekatan yang menggunakan kombinasi paradigma pemrograman modular dan OOP. Pemrograman modular adalah pendekatan di mana program dibagi menjadi modul atau unit yang lebih kecil, yang masing-masing memiliki tugas tertentu dan dapat berinteraksi satu sama lain.
2. OOP Murni,atau sering juga disebut sebagai "Object-Oriented Programming Pure", adalah pendekatan yang sepenuhnya mengikuti prinsip-prinsip OOP tanpa campur tangan paradigma pemrograman lainnya seperti modular, prosedural, atau fungsional.

Dalam prakteknya, banyak bahasa pemrograman dan proyek-proyek menggunakan pendekatan campuran antara paradigma modular dan OOP, sementara yang lain lebih memilih untuk menggunakan OOP secara murni untuk menjaga konsistensi dan kejelasan dalam struktur dan desain programnya.

**Program Celcius OOP Modular**

# File-1 (celcius.py)

class Celcius:

  def \_\_init\_\_(self, suhu):

    self.suhu = suhu

  def get\_celcius(self):

    val = self.suhu

    return val

  def get\_fahrenheit(self):

    val = (9/5 \* self.suhu) + 32

    return val

  def get\_reamur(self):

    val = (4/5 \* self.suhu)

    return val

  def get\_kelvin(self):

    val = self.suhu + 273

    return val

# File-2 (Frmcelcius.py)

from tkinter import Frame,Label,Entry,Button,YES,BOTH,END,Tk,W

from celcius import \*

class FrmCelcius:

    def \_\_init\_\_(self, parent, title):

        self.parent = parent

        #self.parent.geometry("400x400")

        self.parent.title(title)

        self.parent.protocol("WM\_DELETE\_WINDOW", self.onKeluar)

        self.aturKomponen()

    def aturKomponen(self):

        mainFrame = Frame(self.parent, bd=10)

        mainFrame.pack(fill=BOTH, expand=YES)

        Label(mainFrame, text='Celcius:').grid(row=0, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Fahrenheit:").grid(row=2, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Reamur:").grid(row=3, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Kelvin:").grid(row=4, column=0,

            sticky=W, padx=5, pady=5)

        self.txtCelcius = Entry(mainFrame)

        self.txtCelcius.grid(row=0, column=1, padx=5, pady=5)

        self.txtFahrenheit = Entry(mainFrame)

        self.txtFahrenheit.grid(row=2, column=1, padx=5, pady=5)

        self.txtReamur = Entry(mainFrame)

        self.txtReamur.grid(row=3, column=1, padx=5, pady=5)

        self.txtKelvin = Entry(mainFrame)

        self.txtKelvin.grid(row=4, column=1, padx=5, pady=5)

        self.btnHitung = Button(mainFrame, text='Hitung',

            command=self.onHitung)

        self.btnHitung.grid(row=1, column=1, padx=5, pady=5)

    def onHitung(self):

        C = Celcius(int(self.txtCelcius.get()))

        # Suhu dalam Fahrenheit

        F = C.get\_fahrenheit()

        self.txtFahrenheit.delete(0,END)

        self.txtFahrenheit.insert(END,str(F))

        # Suhu dalam Fahrenheit

        R = C.get\_reamur()

        self.txtReamur.delete(0,END)

        self.txtReamur.insert(END,str(R))

        # Suhu dalam Fahrenheit

        K = C.get\_kelvin()

        self.txtKelvin.delete(0,END)

        self.txtKelvin.insert(END,str(K))

    def onKeluar(self, event=None):

        # memberikan perintah menutup aplikasi

        self.parent.destroy()

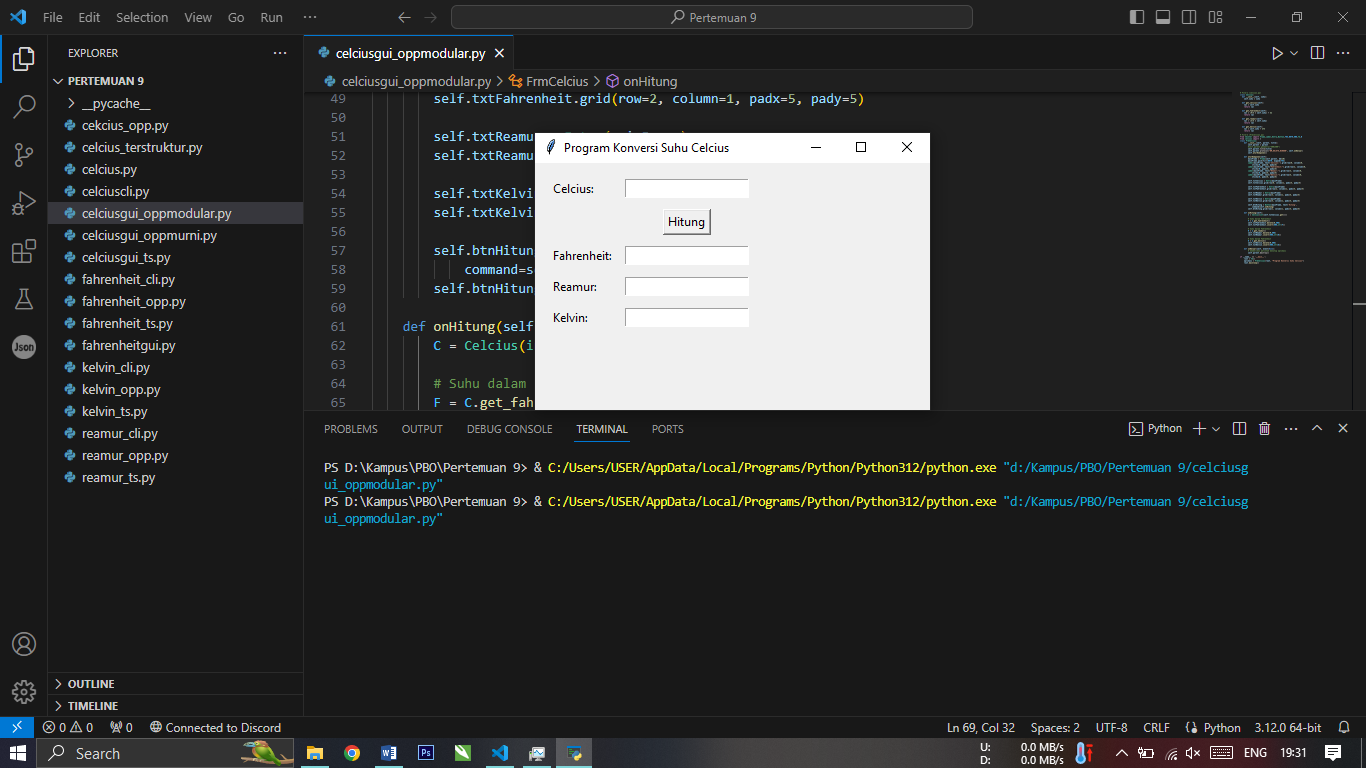
if \_\_name\_\_ == '\_\_main\_\_':

    root = Tk()

    aplikasi = FrmCelcius(root, "Program Konversi Suhu Celcius")

    root.mainloop()

**Hasil Output Celcius OPP Modular**



**Program Celcius OOP Murni**

from tkinter import Frame,Label,Entry,Button,YES,BOTH,END,Tk,W

class FrmCelcius:

    def \_\_init\_\_(self, parent, title):

        self.parent = parent

        #self.parent.geometry("400x400")

        self.parent.title(title)

        self.parent.protocol("WM\_DELETE\_WINDOW", self.onKeluar)

        self.aturKomponen()

    def aturKomponen(self):

        mainFrame = Frame(self.parent, bd=10)

        mainFrame.pack(fill=BOTH, expand=YES)

        Label(mainFrame, text='Celcius:').grid(row=0, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Fahrenheit:").grid(row=2, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Reamur:").grid(row=3, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Kelvin:").grid(row=4, column=0,

            sticky=W, padx=5, pady=5)

        self.txtCelcius = Entry(mainFrame)

        self.txtCelcius.grid(row=0, column=1, padx=5, pady=5)

        self.txtFahrenheit = Entry(mainFrame)

        self.txtFahrenheit.grid(row=2, column=1, padx=5, pady=5)

        self.txtReamur = Entry(mainFrame)

        self.txtReamur.grid(row=3, column=1, padx=5, pady=5)

        self.txtKelvin = Entry(mainFrame)

        self.txtKelvin.grid(row=4, column=1, padx=5, pady=5)

        self.btnHitung = Button(mainFrame, text='Hitung',

            command=self.onHitung)

        self.btnHitung.grid(row=1, column=1, padx=5, pady=5)

    def get\_fahrenheit(self, suhu):

        val = (9/5 \* suhu) + 32

        return val

    def get\_reamur(self, suhu):

        val = (4/5 \* suhu)

        return val

    def get\_kelvin(self, suhu):

        val = suhu + 273

        return val

    def onHitung(self):

        # Suhu dalam Fahrenheit

        suhu = self.txtCelcius.get()

        F = self.get\_fahrenheit(float(suhu))

        self.txtFahrenheit.delete(0,END)

        self.txtFahrenheit.insert(END,str(F))

        # Suhu dalam Reamur

        R = self.get\_reamur(float(suhu))

        self.txtReamur.delete(0,END)

        self.txtReamur.insert(END,str(R))

        # Suhu dalam Kelvin

        K = self.get\_kelvin(float(suhu))

        self.txtKelvin.delete(0,END)

        self.txtKelvin.insert(END,str(K))

    def onKeluar(self, event=None):

        # memberikan perintah menutup aplikasi

        self.parent.destroy()

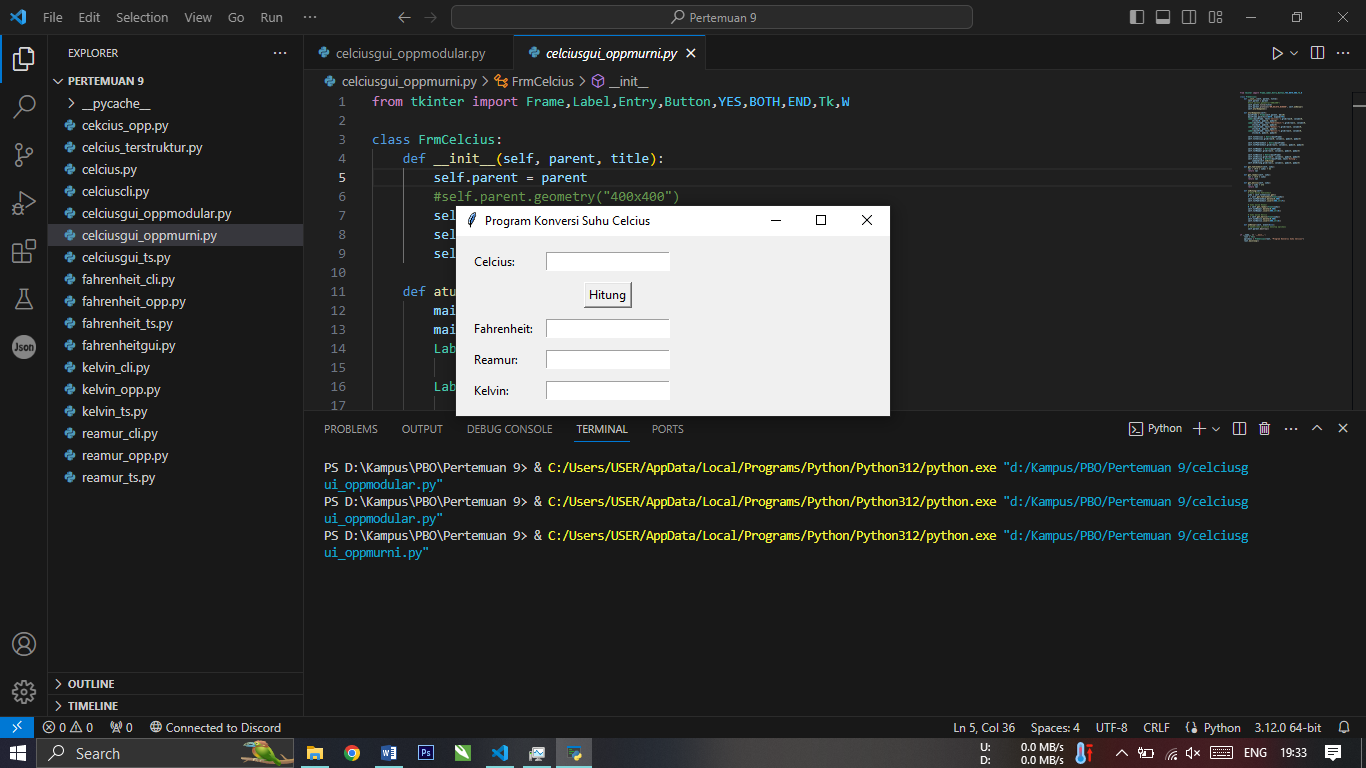
if \_\_name\_\_ == '\_\_main\_\_':

    root = Tk()

    aplikasi = FrmCelcius(root, "Program Konversi Suhu Celcius")

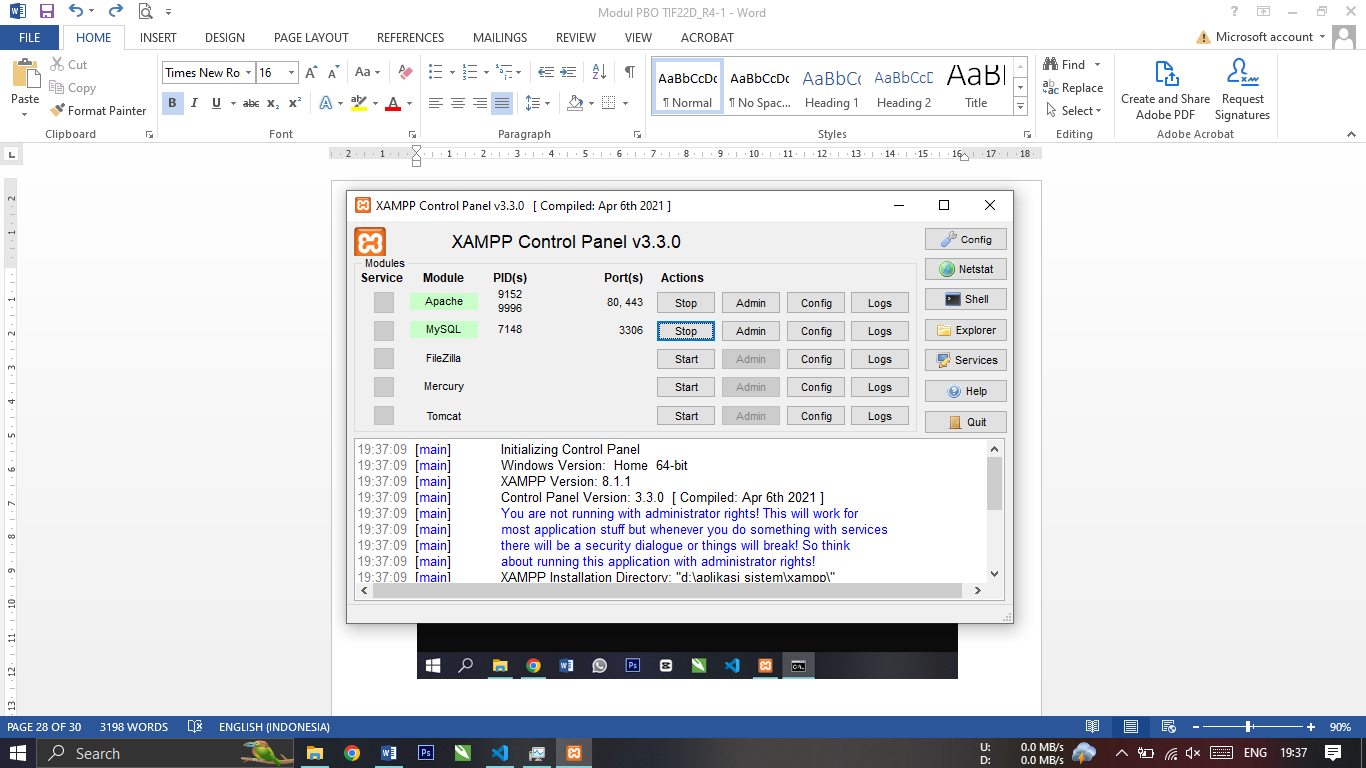
    root.mainloop()

**Hasil Output OOP Murni**

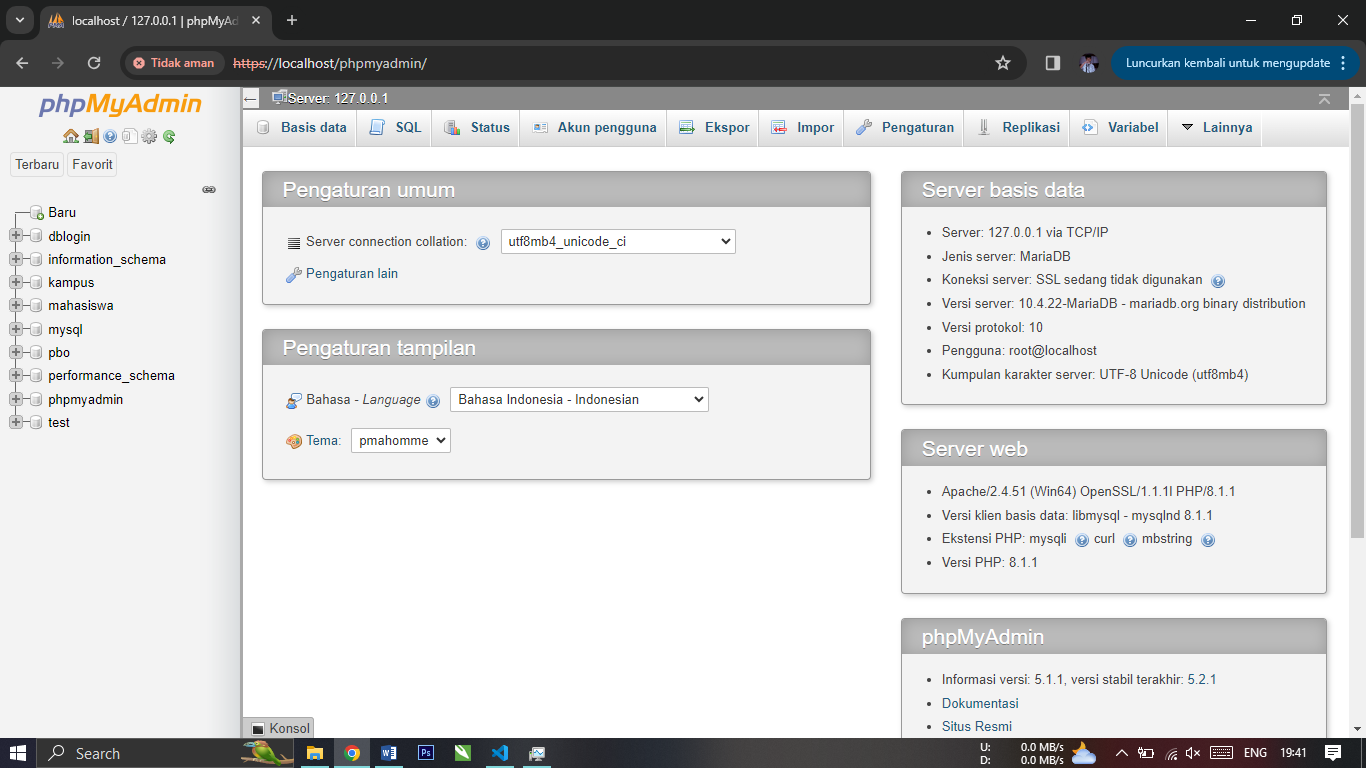


**2.8 Pertemuan 10**

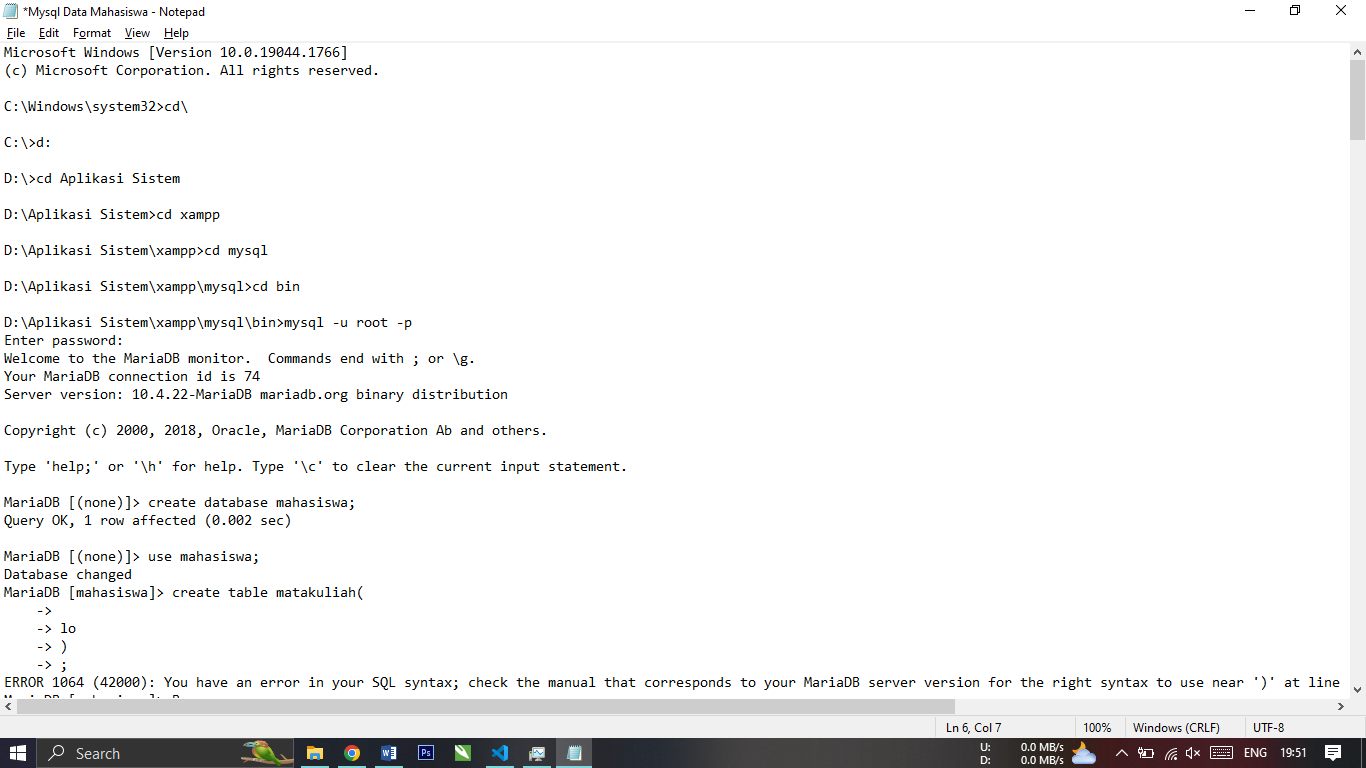
Sebelum memasukkan programnya,harus menginstal apk xampp terlebih dahulu,agar program bisa dijalankan.



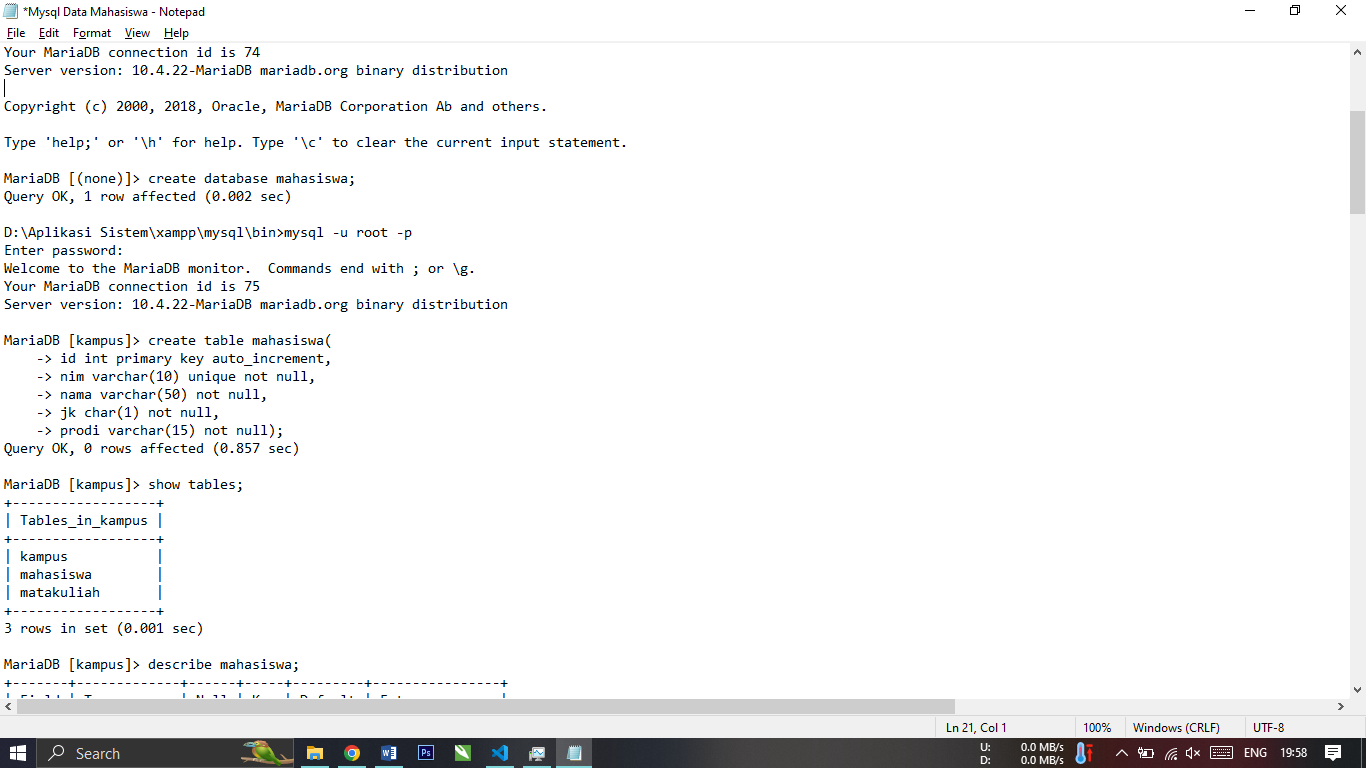
**Jika Sudah, Start dan Pastikan Apache dan Mysql sudah warna Hijau**



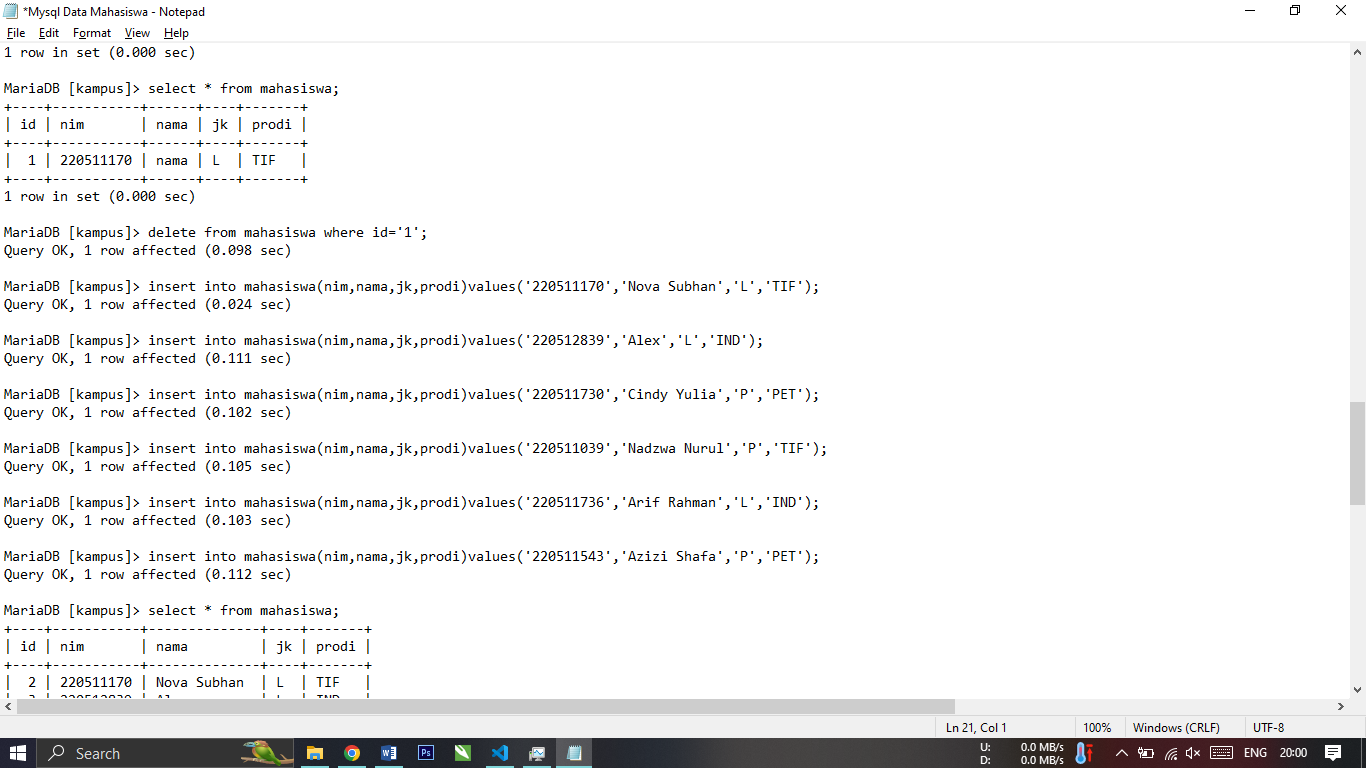
**Lalu Pergi ke** [**https://localhost/phpmyadmin**](https://localhost/phpmyadmin) **dan buka CMD**



**Pastikan masuk ke directory file bin Xampp nya dengan masukan perintah diatas**

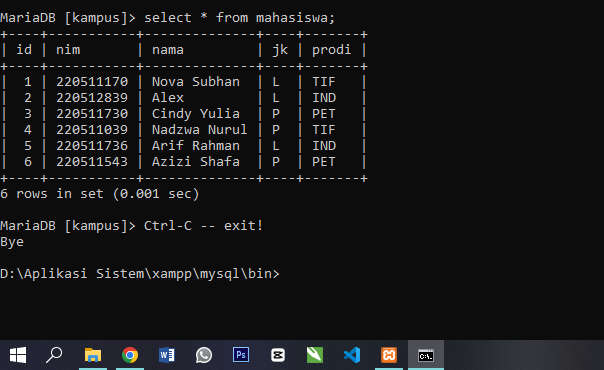


**Buat Database Mahasiswa dan Buat Table Mahasiswa nya**

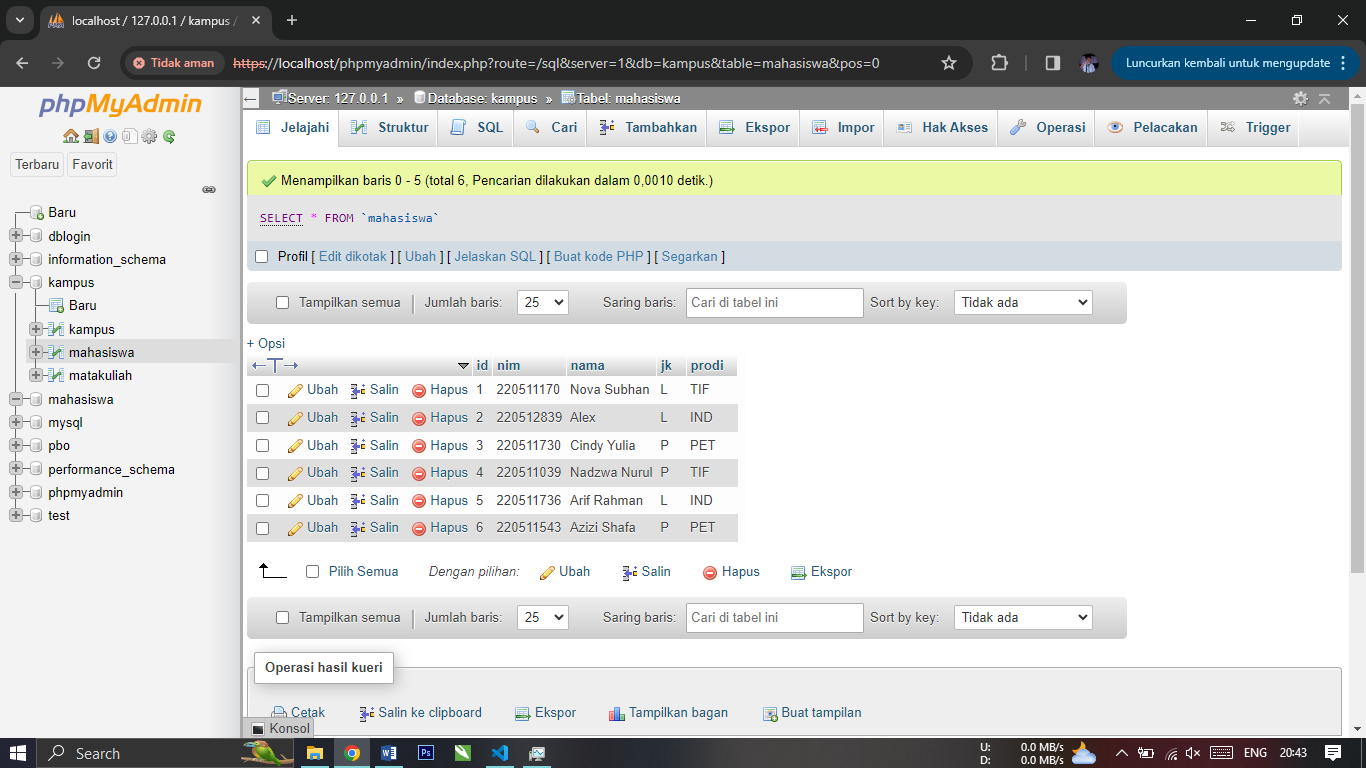


**Masukkan Nama-Nama Mahasiswa yang Diinginkan**

**Dan Dibawah Hasil Outputnya**

****

**Dibawah adalah tampilan di phpmyadmin nya dan bisa juga edit melalui ini**



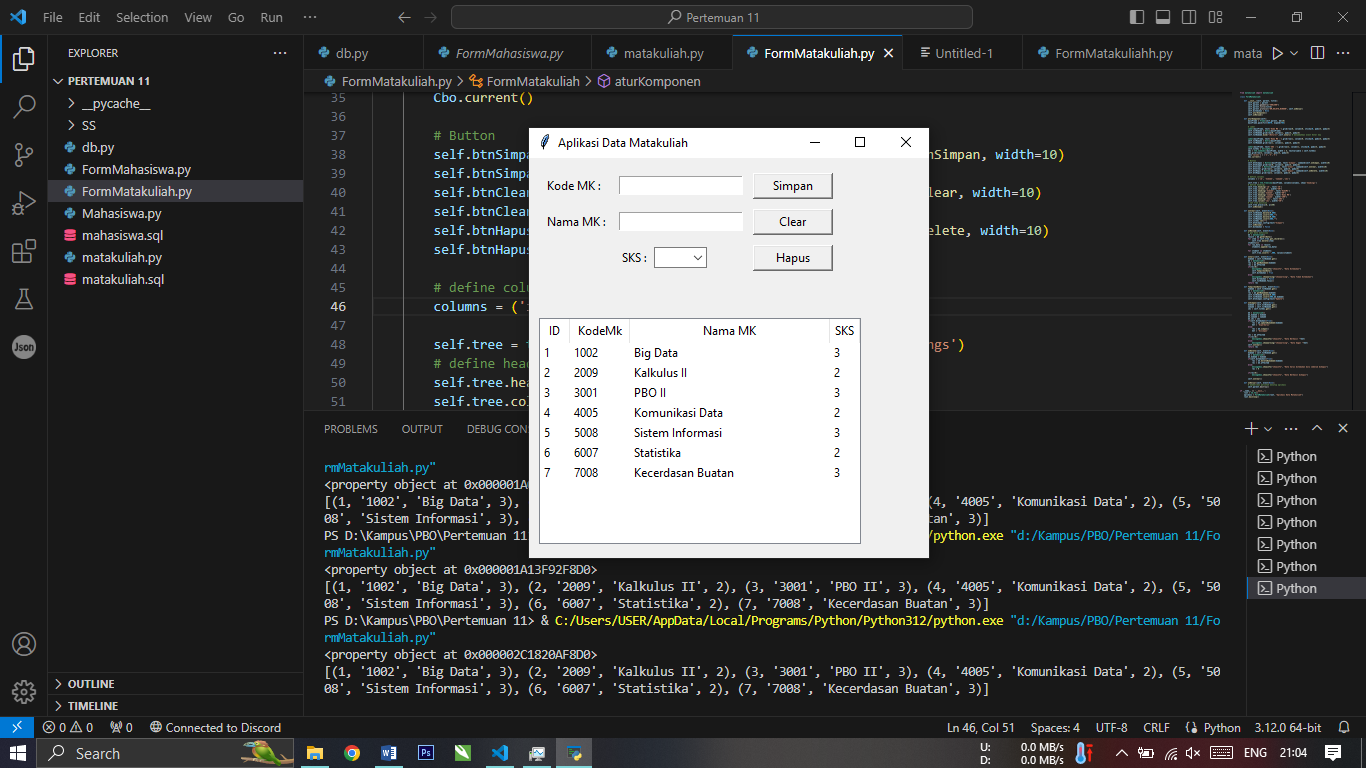
**2.9 Pertemuan 11**

Membahas tentang mengkoneksikan dari VSCode ke Xampp

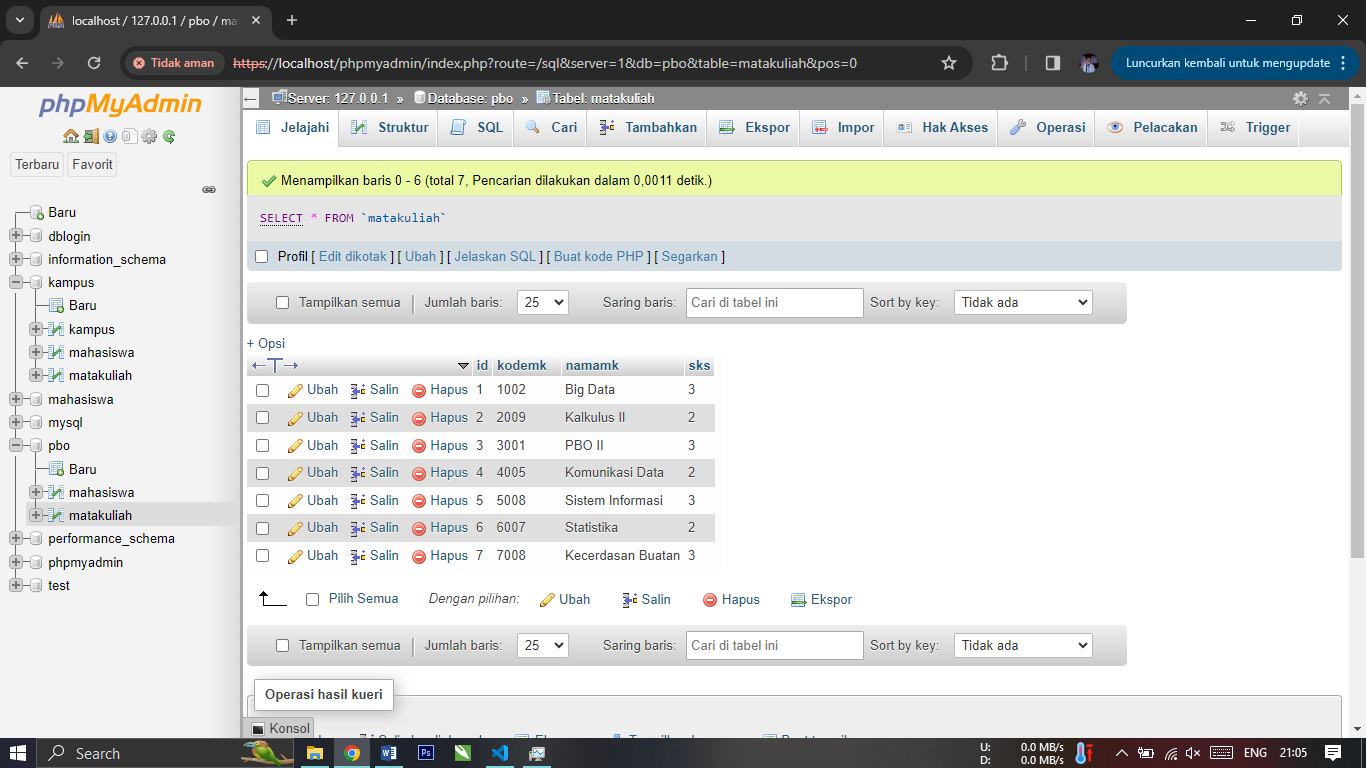
Yang Pertama

* **Program DB ini fungsinya untuk mengkoneksikan**
* import mysql.connector as mc
* class DBConnection:
* def \_\_init\_\_(self):
* self.host = "localhost"
* self.port = 3306
* self.name = "pbo"
* self.user = "root"
* self.password = ""
* self.conn = None
* self.cursor = None
* self.result = None
* self.connected = False
* self.affected = 0
* self.connect()
* @property
* def connection\_status(self):
* return self.connected
* def connect(self):
* try:
* self.conn = mc.connect(host = self.host,
* port = self.port,
* database = self.name,
* user = self.user,
* password = self.password)
* self.connected = True
* self.cursor=self.conn.cursor()
* except mc.Error as e:
* self.connected = False
* return self.conn
* def disconnect(self):
* if(self.connected==True):
* self.conn.close
* else:
* self.conn = None
* def findOne(self, sql):
* self.connect()
* self.cursor.execute(sql)
* self.result = self.cursor.fetchone()
* return self.result
* def findAll(self, sql):
* self.connect()
* self.result = self.cursor.execute(sql)
* self.result = self.cursor.fetchall()
* return self.result
* def insert(self, sql):
* self.connect()
* self.cursor.execute(sql)
* self.conn.commit()
* self.affected = self.cursor.rowcount
* return self.affected
* def update(self, sql, val):
* self.connect()
* self.cursor.execute(sql, val)
* self.conn.commit()
* self.affected = self.cursor.rowcount
* return self.affected
* def delete(self, sql):
* self.connect()
* self.cursor.execute(sql)
* self.conn.commit()
* self.affected = self.cursor.rowcount
* return self.affected
* def show(self, sql):
* self.connect()
* self.cursor.execute(sql)
* self.result = self.cursor.fetchone()
* return self.result
* @property
* def info(self):
* if(self.connected==True):
* return "Server is running on " + self.host + ' using port ' + str(self.port)
* else:
* return "Server is offline."
* A = DBConnection
* B = A.info
* print(B)
* **Program Matakuliah ini yang nantinya bisa disambungkan ke phpmyadmin**
* import tkinter as tk
* from tkinter import Frame,Label,Entry,Button,Radiobutton,ttk,VERTICAL,YES,BOTH,END,Tk,W,StringVar,messagebox
* from matakuliah import matakuliah
* class FormMatakuliah:
* def \_\_init\_\_(self, parent, title):
* self.parent = parent
* self.parent.geometry("400x400")
* self.parent.title(title)
* self.parent.protocol("WM\_DELETE\_WINDOW", self.onKeluar)
* self.ditemukan = None
* self.aturKomponen()
* self.onReload()
* def aturKomponen(self):
* mainFrame = Frame(self.parent, bd=10)
* mainFrame.pack(fill=BOTH, expand=YES)
* # Label
* Label(mainFrame, text='Kode MK :').grid(row=0, column=0, sticky=W, padx=5, pady=5)
* self.txtKodemk = Entry(mainFrame)
* self.txtKodemk.grid(row=0, column=1, padx=5, pady=5)
* self.txtKodemk.bind("<Return>",self.onCari) # menambahkan event Enter key
* Label(mainFrame, text='Nama MK :').grid(row=1, column=0, sticky=W, padx=5, pady=5)
* self.txtNamamk = Entry(mainFrame)
* self.txtNamamk.grid(row=1, column=1, padx=5, pady=5)
* Label(mainFrame, text='SKS :').grid(row=2, column=1, sticky=W, padx=5, pady=5)
* self.txtSks = StringVar()
* Cbo = ttk.Combobox(mainFrame, width = 5, textvariable = self.txtSks)
* Cbo.grid(row=2, column=1, padx=5, pady=5)
* Cbo['values'] = ('2','3','4')
* Cbo.current()
* # Button
* self.btnSimpan = Button(mainFrame, text='Simpan', command=self.onSimpan, width=10)
* self.btnSimpan.grid(row=0, column=3, padx=5, pady=5)
* self.btnClear = Button(mainFrame, text='Clear', command=self.onClear, width=10)
* self.btnClear.grid(row=1, column=3, padx=5, pady=5)
* self.btnHapus = Button(mainFrame, text='Hapus', command=self.onDelete, width=10)
* self.btnHapus.grid(row=2, column=3, padx=5, pady=5)
* # define columns
* columns = ('id', 'kodemk', 'namamk','sks')
* self.tree = ttk.Treeview(mainFrame, columns=columns, show='headings')
* # define headings
* self.tree.heading('id', text='ID')
* self.tree.column('id', width="30")
* self.tree.heading('kodemk', text='KodeMk')
* self.tree.column('kodemk', width="60")
* self.tree.heading('namamk', text='Nama MK')
* self.tree.column('namamk', width="200")
* self.tree.heading('sks', text='SKS')
* self.tree.column('sks', width="30")
* # set tree position
* self.tree.place(x=0, y=150)
* self.onReload()
* def onClear(self, event=None):
* self.txtKodemk.delete(0,END)
* self.txtKodemk.insert(END,"")
* self.txtNamamk.delete(0,END)
* self.txtNamamk.insert(END,"")
* self.txtSks.set("")
* self.btnSimpan.config(text="Simpan")
* self.onReload()
* self.ditemukan = False
* def onReload(self, event=None):
* # get data mahasiswa
* mk = matakuliah()
* result = mk.getAllData()
* for item in self.tree.get\_children():
* self.tree.delete(item)
* students=[]
* for row\_data in result:
* students.append(row\_data)
* for student in students:
* self.tree.insert('',END, values=student)
* def onCari(self, event=None):
* kodemk = self.txtKodemk.get()
* mk = matakuliah()
* res = mk.getBykodemk(kodemk)
* rec = mk.affected
* if(rec>0):
* messagebox.showinfo("showinfo", "Data Ditemukan")
* self.TampilkanData()
* self.ditemukan = True
* else:
* messagebox.showwarning("showwarning", "Data Tidak Ditemukan")
* self.ditemukan = False
* self.txtKodemk.focus()
* return res
* def TampilkanData(self, event=None):
* kodemk = self.txtKodemk.get()
* mk = matakuliah()
* res = mk.getBykodemk(kodemk)
* self.txtKodemk.delete(0,END)
* self.txtKodemk.insert(END,mk.kodemk)
* self.btnSimpan.config(text="Update")
* def onSimpan(self, event=None):
* kodemk = self.txtKodemk.get()
* namamk = self.txtNamamk.get()
* sks = self.txtSks.get()
* mk = matakuliah()
* mk.kodemk = kodemk
* mk.namamk = namamk
* mk.sks = sks
* if(self.ditemukan==True):
* res = mk.updateBykodemk(kodemk)
* ket = 'Diperbarui'
* else:
* res = mk.simpan()
* ket = 'Disimpan'
* rec = mk.affected
* if(rec>0):
* messagebox.showinfo("showinfo", "Data Berhasil "+ket)
* else:
* messagebox.showwarning("showwarning", "Data Gagal "+ket)
* self.onClear()
* return rec
* def onDelete(self, event=None):
* kodemk = self.txtKodemk.get()
* mk = matakuliah()
* mk.kodemk = kodemk
* if(self.ditemukan==True):
* res = mk.deleteBykodemk(kodemk)
* rec = mk.affected
* else:
* messagebox.showinfo("showinfo", "Data harus ditemukan dulu sebelum dihapus")
* rec = 0
* if(rec>0):
* messagebox.showinfo("showinfo", "Data Berhasil dihapus")
* self.onClear()
* def onKeluar(self, event=None):
* # memberikan perintah menutup aplikasi
* self.parent.destroy()
* if \_\_name\_\_ == '\_\_main\_\_':
* root = tk.Tk()
* aplikasi = FormMatakuliah(root, "Aplikasi Data Matakuliah")
* root.mainloop()

**Hasil Output Matakuliah**



**Tampilan di Phpmyadmin**

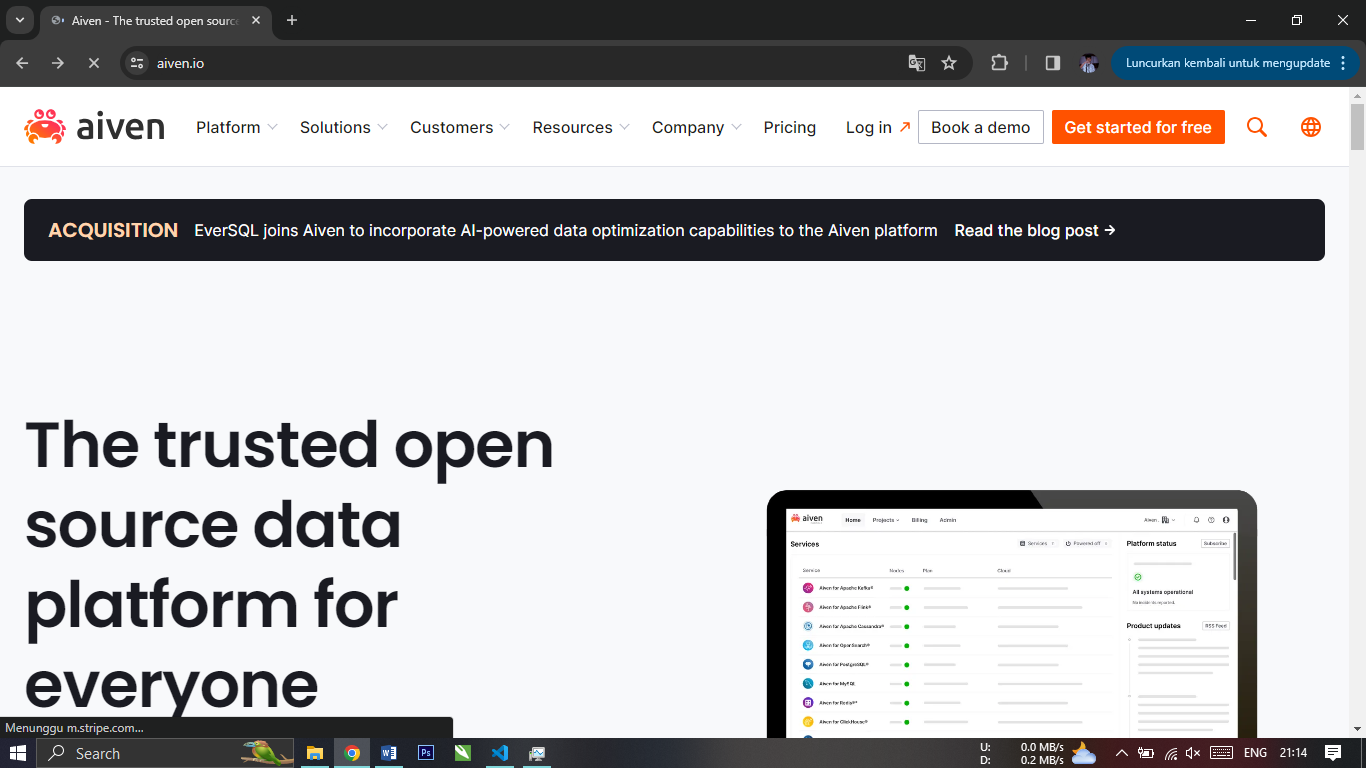


**2.10 Pertemuan 12**

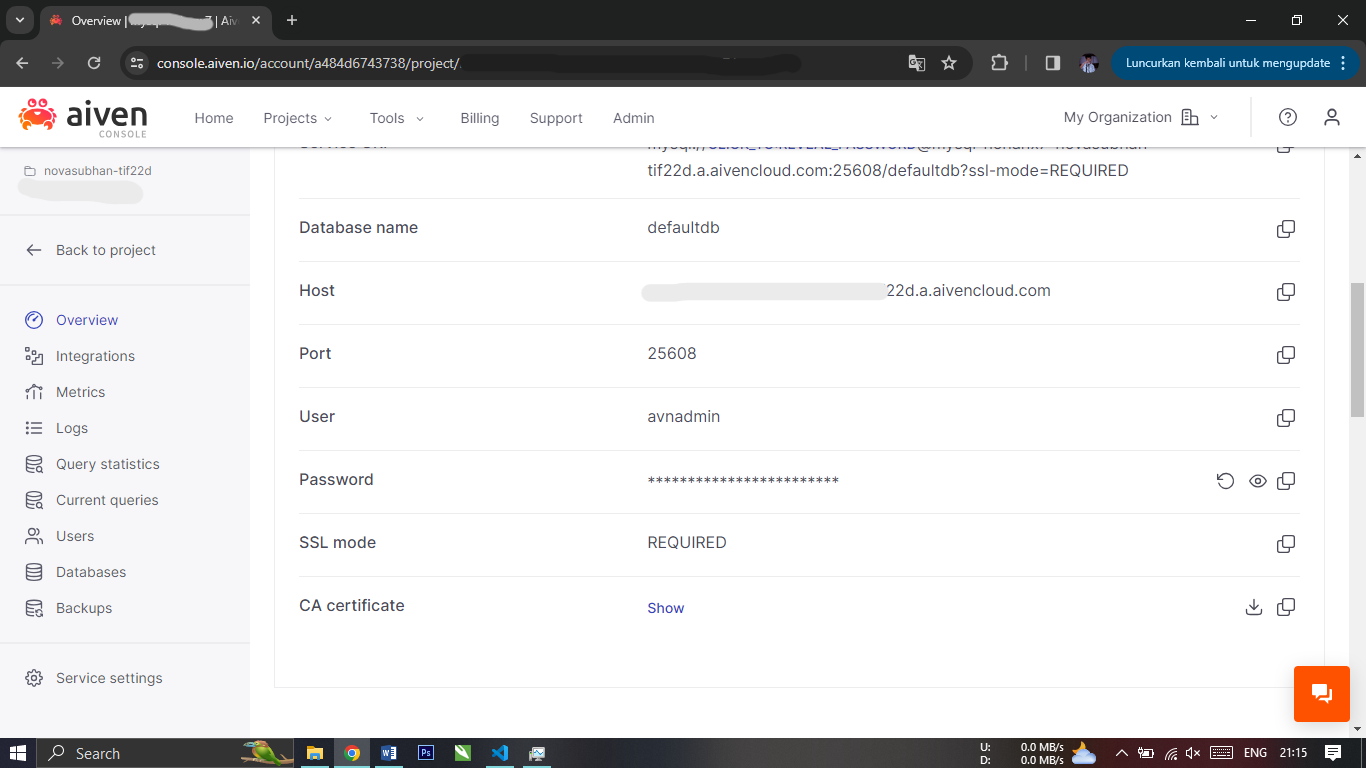
Dipertemuan ini membahas tentang menyambungkan VScode ke layanan basis data sharing online disini kelompok kami menggunakan Aiven.io.

Aiven adalah layanan basis data daring yang menyediakan informasi metadata tentang audio CD.Untuk menggunakan Aiven dalam pemrograman Python, Anda dapat menggunakan pustaka Python yang mendukung protokol Aiven, seperti `aiven.py`. Ingatlah bahwa untuk menggunakan Aiven, Anda perlu memiliki akses internet yang aktif.

**Kalian Bisa Klik saja bagian Get Started For Free untuk membuat database online kalian**

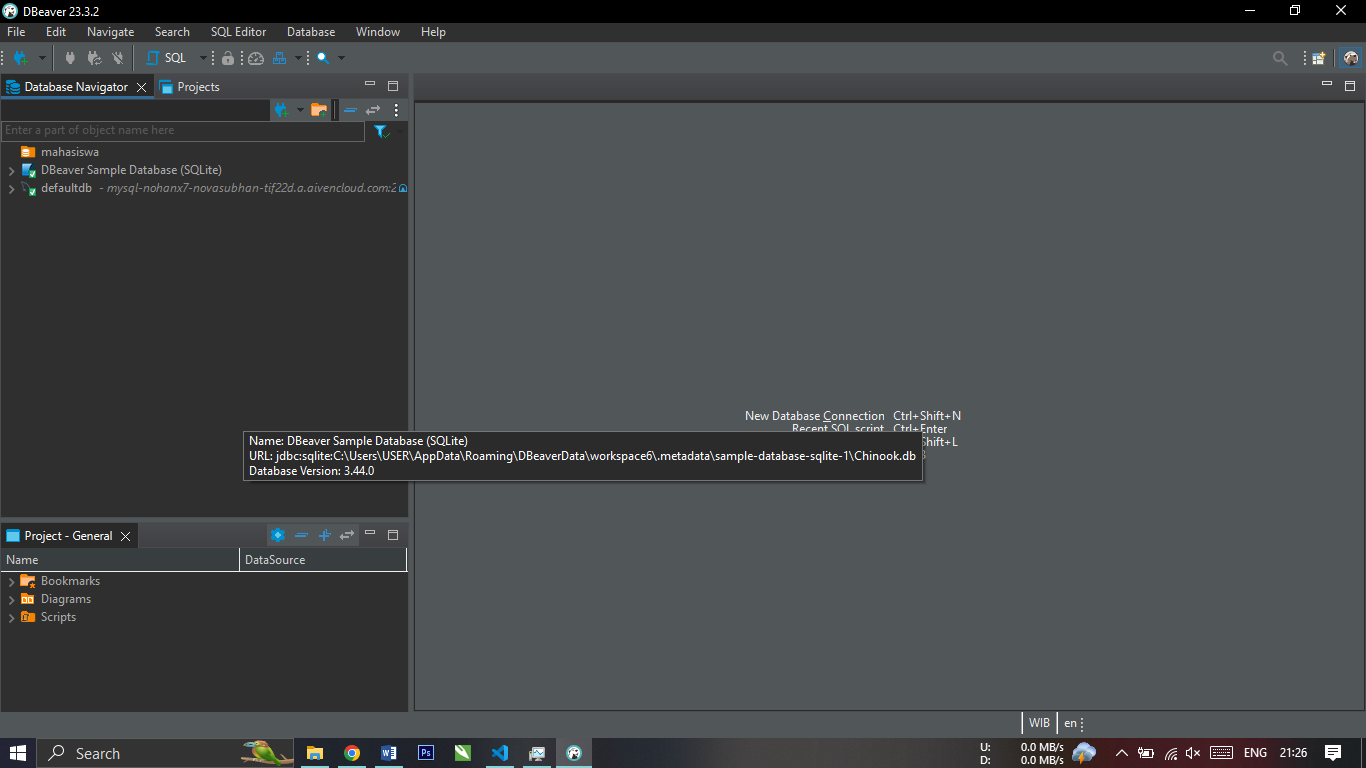


**Lalu disini kalian akan mendapatkan Database Name Host Port dll**



**Lalu Unduh CA Certificate / ca.pem dan tempatkan di program kalian**

**Lalu install aplikasi DBeaver dibawah ini**



**Program DB akan selalu ada karena fungsinya untuk mengkoneksikan**

**Dan samakan dengan apa yang ada di Aiven.io**

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(self):

        self.host = "PRIVATE.a.aivencloud.com"

        self.port = 25608

        self.name = "defaultdb"

        self.user = "avnadmin"

        self.password = "…PRIVATE…"

        self.ssl='ca.pem'

        self.pure=True

        self.conn = None

        self.cursor = None

        self.result = None

        self.connected = False

        self.affected = 0

        self.connect()

    @property

    def connection\_status(self):

        return self.connected

    def connect(self):

        try:

            self.conn = mc.connect(host = self.host,

                                    port = self.port,

                                    database = self.name,

                                    user = self.user,

                                    ssl\_ca=self.ssl,

                                    use\_pure=self.pure,

                                    password = self.password)

            self.connected = True

            self.cursor=self.conn.cursor()

        except mc.Error as e:

            self.connected = False

        return self.conn

    def disconnect(self):

        if(self.connected==True):

            self.conn.close

        else:

            self.conn = None

    def findOne(self, sql):

        self.connect()

        self.cursor.execute(sql)

        self.result = self.cursor.fetchone()

        return self.result

    def findAll(self, sql):

        self.connect()

        self.result = self.cursor.execute(sql)

        self.result = self.cursor.fetchall()

        return self.result

    def insert(self, sql):

        self.connect()

        self.cursor.execute(sql)

        self.conn.commit()

        self.affected = self.cursor.rowcount

        return self.affected

    def update(self, sql, val):

        self.connect()

        self.cursor.execute(sql, val)

        self.conn.commit()

        self.affected = self.cursor.rowcount

        return self.affected

    def delete(self, sql):

        self.connect()

        self.cursor.execute(sql)

        self.conn.commit()

        self.affected = self.cursor.rowcount

        return self.affected

    def show(self, sql):

        self.connect()

        self.cursor.execute(sql)

        self.result = self.cursor.fetchone()

        return self.result

    @property

    def info(self):

        if(self.connected==True):

            return "Server is running on " + self.host + ' using port ' + str(self.port)

        else:

            return "Server is offline."

A = DBConnection

B = A.info

print(B)

**Dan Berikut hasil program matakuliahnya**

import tkinter as tk

from tkinter import Frame,Label,Entry,Button,Radiobutton,ttk,VERTICAL,YES,BOTH,END,Tk,W,StringVar,messagebox

from matakuliah import matakuliah

class FormMatakuliah:

    def \_\_init\_\_(self, parent, title):

        self.parent = parent

        self.parent.geometry("400x400")

        self.parent.title(title)

        self.parent.protocol("WM\_DELETE\_WINDOW", self.onKeluar)

        self.ditemukan = None

        self.aturKomponen()

        self.onReload()

    def aturKomponen(self):

        mainFrame = Frame(self.parent, bd=10)

        mainFrame.pack(fill=BOTH, expand=YES)

        # Label

        Label(mainFrame, text='Kode MK :').grid(row=0, column=0, sticky=W, padx=5, pady=5)

        self.txtKodemk = Entry(mainFrame)

        self.txtKodemk.grid(row=0, column=1, padx=5, pady=5)

        self.txtKodemk.bind("<Return>",self.onCari) # menambahkan event Enter key

        Label(mainFrame, text='Nama MK :').grid(row=1, column=0, sticky=W, padx=5, pady=5)

        self.txtNamamk = Entry(mainFrame)

        self.txtNamamk.grid(row=1, column=1, padx=5, pady=5)

        Label(mainFrame, text='SKS :').grid(row=2, column=1, sticky=W, padx=5, pady=5)

        self.txtSks = StringVar()

        Cbo = ttk.Combobox(mainFrame, width = 5, textvariable = self.txtSks)

        Cbo.grid(row=2, column=1, padx=5, pady=5)

        Cbo['values'] = ('2','3','4')

        Cbo.current()

        # Button

        self.btnSimpan = Button(mainFrame, text='Simpan', command=self.onSimpan, width=10)

        self.btnSimpan.grid(row=0, column=3, padx=5, pady=5)

        self.btnClear = Button(mainFrame, text='Clear', command=self.onClear, width=10)

        self.btnClear.grid(row=1, column=3, padx=5, pady=5)

        self.btnHapus = Button(mainFrame, text='Hapus', command=self.onDelete, width=10)

        self.btnHapus.grid(row=2, column=3, padx=5, pady=5)

        # define columns

        columns = ('ID', 'Kode MK', 'Nama MK','SKS')

        self.tree = ttk.Treeview(mainFrame, columns=columns, show='headings')

        # define headings

        self.tree.heading('ID', text='ID')

        self.tree.column('ID', width="30")

        self.tree.heading('Kode MK', text='KodeMk')

        self.tree.column('Kode MK', width="60")

        self.tree.heading('Nama MK', text='Nama MK')

        self.tree.column('Nama MK', width="200")

        self.tree.heading('SKS', text='SKS')

        self.tree.column('SKS', width="30")

        # set tree position

        self.tree.place(x=0, y=150)

        self.onReload()

    def onClear(self, event=None):

        self.txtKodemk.delete(0,END)

        self.txtKodemk.insert(END,"")

        self.txtNamamk.delete(0,END)

        self.txtNamamk.insert(END,"")

        self.txtSks.set("")

        self.btnSimpan.config(text="Simpan")

        self.onReload()

        self.ditemukan = False

    def onReload(self, event=None):

        # get data mahasiswa

        mk = matakuliah()

        result = mk.getAllData()

        for item in self.tree.get\_children():

            self.tree.delete(item)

        students=[]

        for row\_data in result:

            students.append(row\_data)

        for student in students:

            self.tree.insert('',END, values=student)

    def onCari(self, event=None):

        kodemk = self.txtKodemk.get()

        mk = matakuliah()

        res = mk.getBykodemk(kodemk)

        rec = mk.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Ditemukan")

            self.TampilkanData()

            self.ditemukan = True

        else:

            messagebox.showwarning("showwarning", "Data Tidak Ditemukan")

            self.ditemukan = False

            self.txtKodemk.focus()

        return res

    def TampilkanData(self, event=None):

        kodemk = self.txtKodemk.get()

        mk = matakuliah()

        res = mk.getBykodemk(kodemk)

        self.txtKodemk.delete(0,END)

        self.txtKodemk.insert(END,mk.kodemk)

        self.btnSimpan.config(text="Update")

    def onSimpan(self, event=None):

        kodemk = self.txtKodemk.get()

        namamk = self.txtNamamk.get()

        sks = self.txtSks.get()

        mk = matakuliah()

        mk.kodemk = kodemk

        mk.namamk = namamk

        mk.sks = sks

        if(self.ditemukan==True):

            res = mk.updateBykodemk(kodemk)

            ket = 'Diperbarui'

        else:

            res = mk.simpan()

            ket = 'Disimpan'

        rec = mk.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil "+ket)

        else:

            messagebox.showwarning("showwarning", "Data Gagal "+ket)

        self.onClear()

        return rec

    def onDelete(self, event=None):

        kodemk = self.txtKodemk.get()

        mk = matakuliah()

        mk.kodemk = kodemk

        if(self.ditemukan==True):

            res = mk.deleteBykodemk(kodemk)

            rec = mk.affected

        else:

            messagebox.showinfo("showinfo", "Data harus ditemukan dulu sebelum dihapus")

            rec = 0

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil dihapus")

        self.onClear()

    def onKeluar(self, event=None):

        # memberikan perintah menutup aplikasi

        self.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    root = tk.Tk()

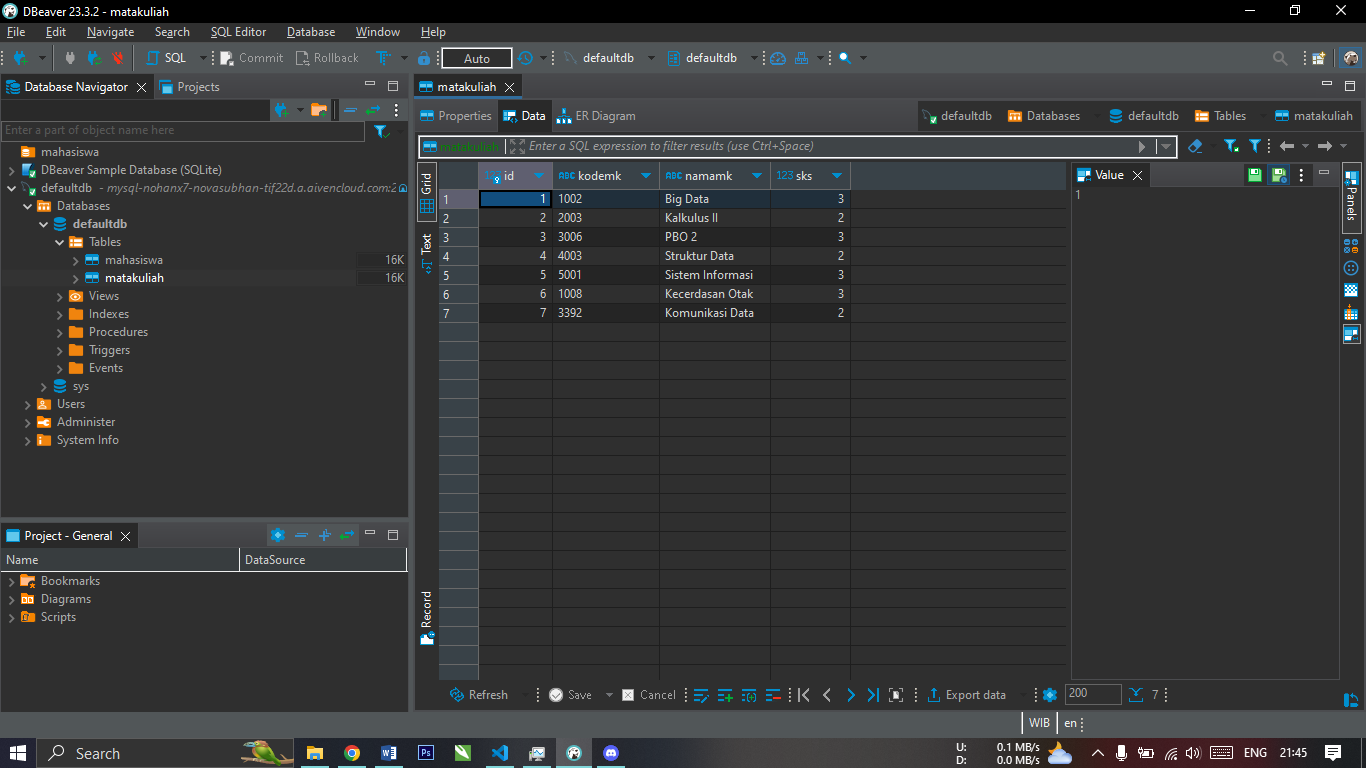
    aplikasi = FormMatakuliah(root, "Aplikasi Data Matakuliah")

    root.mainloop()

**Hasil Output Program Matakuliah**



**Dan ini hasil Output Database di Aiven**



**2.11 Pertemuan 13**

Login user db Untuk membuat sistem login pengguna dalam pemrograman Python yang menggunakan database, Anda biasanya akan menggunakan modul atau pustaka untuk berinteraksi dengan database (misalnya, SQLite, MySQL, PostgreSQL, dll.).

Pastikan Anda mengganti bagian `add\_user` dengan cara yang lebih aman seperti menambahkan fungsi registrasi yang mencakup validasi input pengguna dan perlindungan terhadap serangan SQL injection. Selain itu, pastikan juga untuk menyimpan kredensial database dengan aman dan tidak menyimpan sandi pengguna dalam bentuk teks biasa di database.

Dengan membuat akun untuk login dan passwordnya anda bisa login melalui dashboard dan setelah itu anda akan mendapatkan akses lebih untuk itu.

**Berikut untuk program Dashboard nya**

import tkinter as tk

from tkinter import Menu, messagebox

from FrmLogin import \*

from FrmPersegi import \*

from FrmSegitiga import \*

from FrmLingkaran import \*

class Dashboard:

    def \_\_init\_\_(self):

        # root window

        self.root = tk.Tk()

        self.root.title('Menu Demo')

        # self.root.attributes('-fullscreen', True)

        self.root.geometry("900x400")

        self.\_\_data = None

        self.\_\_level = None

        # create a menubar

        self.menubar = Menu(self.root)

        self.root.config(menu=self.menubar)

        # create menus

        self.file\_menu = Menu(self.menubar)

        self.guest\_menu = Menu(self.menubar)

        self.admin\_menu = Menu(self.menubar)

        self.mahasiswa\_menu = Menu(self.menubar)

        self.dosen\_menu = Menu(self.menubar)

        # add menu items to File menu

        self.file\_menu.add\_command(label='Login', command=lambda: self.new\_window("Log Me In", FormLogin))

        self.file\_menu.add\_command(label='Exit', command=self.root.destroy)

        # add menu items to menu Admin

        self.admin\_menu.add\_command(label='Admin-Persegi', command=lambda: self.new\_window("Luas Persegi", FrmPersegi))

        self.admin\_menu.add\_command(label='Admin-Segitiga', command=lambda: self.new\_window("Luas Segitiga", FrmSegitiga))

        self.admin\_menu.add\_command(label='Admin-Lingkaran', command=lambda: self.new\_window("Luas Lingkaran", FrmLingkaran))

        # add menu items to menu Mahasiswa

        self.mahasiswa\_menu.add\_command(label='Mahasiswa-Persegi', command=lambda: self.new\_window("Luas Persegi", FrmPersegi))

        self.mahasiswa\_menu.add\_command(label='Mahasiswa-Segitiga', command=lambda: self.new\_window("Luas Segitiga", FrmSegitiga))

        self.mahasiswa\_menu.add\_command(label='Mahasiswa-Lingkaran', command=lambda: self.new\_window("Luas Lingkaran", FrmLingkaran))

        # add menu items to menu Dosen

        self.dosen\_menu.add\_command(label='Dosen-Persegi', command=lambda: self.new\_window("Luas Persegi", FrmPersegi))

        self.dosen\_menu.add\_command(label='Dosen-Segitiga', command=lambda: self.new\_window("Luas Segitiga", FrmSegitiga))

        self.dosen\_menu.add\_command(label='Dosen-Lingkaran', command=lambda: self.new\_window("Luas Lingkaran", FrmLingkaran))

        # add menus to the menubar

        self.menubar.add\_cascade(label="File", menu=self.file\_menu)

    def new\_window(self, number, \_class):

        new = tk.Toplevel(self.root)

        new.transient()

        new.grab\_set()

        \_class(new, number, self.update\_main\_window)

    def update\_main\_window(self, data):

        # Method to receive data from child windows

        self.\_\_data = data

        level = self.\_\_data[0]

        loginvalid = self.\_\_data[1]

        if(loginvalid==True):

            index = self.file\_menu.index('Login')

            # hapus menu login

            self.file\_menu.delete(index)

            self.file\_menu.add\_command(label='Logout', command=self.Logout)

            # tambahkan menu sesuai level

            if(level=='admin'):

                self.menubar.add\_cascade(label="Admin", menu=self.admin\_menu)

                self.\_\_level = 'Admin'

            elif(level=='mahasiswa'):

                self.menubar.add\_cascade(label="Mahasiswa", menu=self.mahasiswa\_menu)

                self.\_\_level = 'Mahasiswa'

            elif(level=='dosen'):

                self.menubar.add\_cascade(label="Dosen", menu=self.dosen\_menu)

                self.\_\_level = 'Dosen'

            else:

                pass

    def Logout(self):

        index = self.file\_menu.index('Logout')

        self.file\_menu.delete(index)

        self.file\_menu.add\_command(label='Login', command=lambda: self.new\_window("Log Me In", FormLogin))

        self.remove\_all\_menus()

    def remove\_all\_menus(self):

        index = self.menubar.index(self.\_\_level)

        if index is not None:

            self.menubar.delete(index)

    def run(self):

        self.root.mainloop()

if \_\_name\_\_ == "\_\_main\_\_":

    menu\_app = Dashboard()

    menu\_app.run()

**Dan Tidak lupa untuk selalu masukkan program DB nya**

**Berikut adalah program dari users.py**

import bcrypt

from db import DBConnection as mydb

class Users:

    def \_\_init\_\_(self):

        self.\_\_id= None

        self.\_\_email= None

        self.\_\_nama= None

        self.\_\_password= None

        self.\_\_level= None

        self.\_\_uservalid = None

        self.\_\_passwordvalid = None

        self.\_\_loginvalid = None#

        self.conn = None

        self.affected = None

        self.result = None

    @property

    def id(self):

        return self.\_\_id

    @property

    def email(self):

        return self.\_\_email

    @email.setter

    def email(self, value):

        self.\_\_email = value

    @property

    def nama(self):

        return self.\_\_nama

    @nama.setter

    def nama(self, value):

        self.\_\_nama = value

    @property

    def password(self):

        return self.\_\_password

    @password.setter

    def password(self, value):

        self.\_\_password = value

    @property

    def level(self):

        return self.\_\_level

    @level.setter

    def level(self, value):

        self.\_\_level = value

    @property

    def loginvalid(self):

        return self.\_\_loginvalid

    @loginvalid.setter

    def loginvalid(self, value):

        self.\_\_loginvalid = value

    def cekUsername(self, email):

        self.conn = mydb()

        sql="SELECT \* FROM users WHERE email='" + email + "'"

        self.result = self.conn.findOne(sql)

        if(self.result!=None):

            self.\_\_email = self.result[1]

            self.\_\_nama = self.result[2]

            self.\_\_password = self.result[3]

            self.\_\_level = self.result[4]

            self.affected = self.conn.cursor.rowcount

            self.\_\_uservalid = True

        else:

            self.\_\_email = ''

            self.\_\_nama = ''

            self.\_\_password = ''

            self.\_\_level = ''

            self.affected = 0

            self.\_\_uservalid = False

        return self.\_\_uservalid

    def cekPassword(self, password):

        hashedpass=self.\_\_password.encode('utf-8')

        c = password.encode('utf-8')

        d = bcrypt.checkpw(c, hashedpass)

        if(d):

            self.\_\_passwordvalid=True

        else:

            self.\_\_passwordvalid=False

        return self.\_\_passwordvalid

    def Validasi(self, email, password):

        a = self.cekUsername(email)

        if(a==True):

            b = self.cekPassword(password)

            if(b==True):

                self.\_\_loginvalid=True

            else:

                self.\_\_loginvalid=False

        else:

            self.\_\_loginvalid=False

        val = []

        val = [self.\_\_level, self.\_\_loginvalid]

        return val

'''A = Users()

print("\n\n")

print("Username BENAR, dan password BENAR")

B = A.Validasi('nurjati@umc.ac.id','123')

print(B)

print("\n\nUsername BENAR, dan password SALAH")

C = A.Validasi('nurjati@umc.ac.id','1234')

print(C)

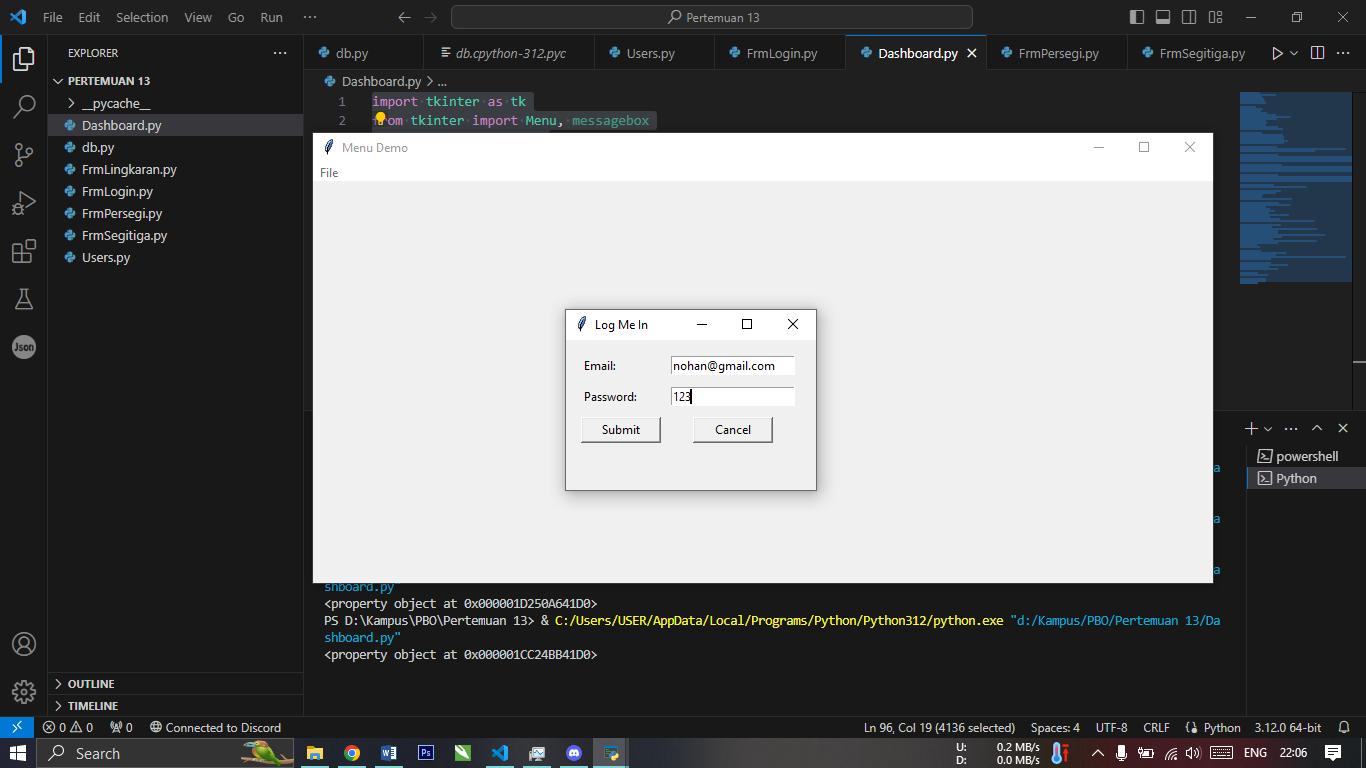
print("\n\nUsername SALAH, dan password BENAR")

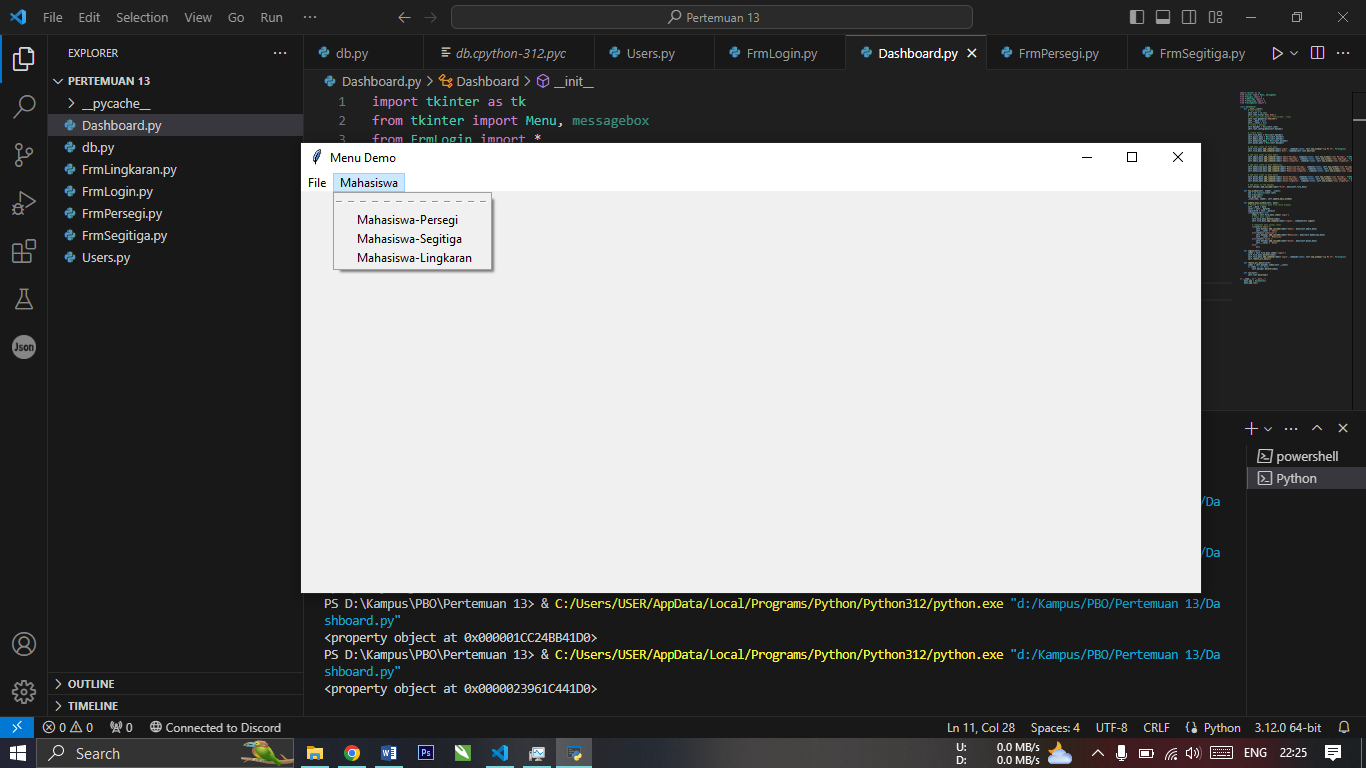
D = A.Validasi('baim@umc.ac.id','123')

print(D)

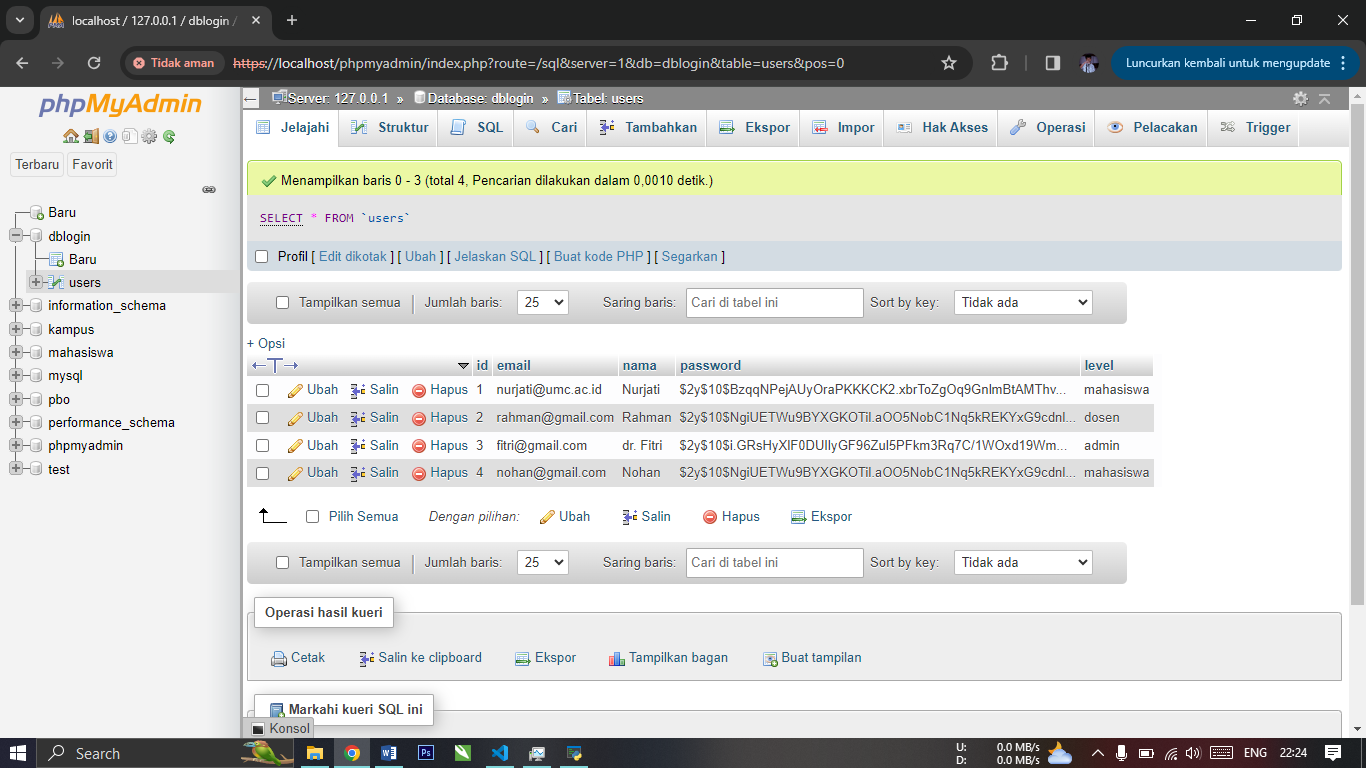
print("\n\n")'''

**Berikut Hasil Output untuk Log-in Dashboard**





**Akun phpmyadmin**



**BAB III**

**PENUTUP**

* 1. **KESIMPULAN**

Python adalah bahasa pemrograman yang cocok untuk pemula maupun profesional. Python merupakan bahasa pemrograman yang mudah dipelajari, sehingga cocok untuk pemula yang ingin belajar pemrograman. Python juga merupakan bahasa pemrograman yang serbaguna, Python memiliki beberapa keunggulan, yaitu: Sintaks yang sederhana dan mudah dipahami,Fleksibel dan dapat digunakan untuk berbagai keperluan,Memiliki komunitas yang aktif dan dukungan yang luas.