

**SOURCE CODE**

**Belajar Python**

Mudah belajar python dengan menggunakan source code.

**Step By Step**

Langkah solusi jitu belajar python dengan langkah-langkah yang benar!.

MODUL PRAKTIKUM

PBO

**MODUL AJAR**

**Nama Kelompok  
1. R.Satria Maulana A.W.P.P. (170511019)  
2. Imanda (210511089)  
3. Moh. Rifki Rahmadan (210511020)**

# **DAFTAR ISI**

[**DAFTAR ISI** ii](#_Toc157375133)

[**A.** **KATA PENGANTAR** v](#_Toc157375134)

[**B.** **PEMBAHASAN** vi](#_Toc157375135)

[**1.** **Python** vi](#_Toc157375136)

[**a.** **Pustaka atau Modul (Library or Module):** vi](#_Toc157375137)

[**b.** **Paket (Package):** vi](#_Toc157375138)

[**c.** **Pip (Python Package Installer):** vi](#_Toc157375139)

[**2.** **Pemrograman** **Berorientasi** **Objek** vii](#_Toc157375140)

[**a.** **Encapsulation (Enkapsulasi):** vii](#_Toc157375141)

[**b.** **Inheritance (Warisan):** vii](#_Toc157375142)

[**c.** **Polymorphism (Polimorfisme):** viii](#_Toc157375143)

[**C.** **INSTALASI** ix](#_Toc157375144)

[**1.** **Instalasi Pyton** ix](#_Toc157375145)

[**a.** **Unduh Python dari Situs Resmi** ix](#_Toc157375146)

[**b.** **Pilih Installer yang Tepat** ix](#_Toc157375147)

[**c.** **Unduh Installer Python** ix](#_Toc157375148)

[**d.** **Jalankan Installer Python** ix](#_Toc157375149)

[**e.** **Proses Instalasi** x](#_Toc157375150)

[**f.** **Verifikasi Instalasi** x](#_Toc157375151)

[**D.** **PERTEMUAN 1** 1](#_Toc157375152)

[**1.** **Materi Pengenalan Python (Variabel, Tipe Data, Operator)** 1](#_Toc157375153)

[**a.** **Rumus Luas Permukaan Bangun Ruang Dan Volume.** 1](#_Toc157375154)

[**b.** **Pembahasan** 2](#_Toc157375155)

[**c.** **Latihan** 4](#_Toc157375156)

[**E.** **PERTEMUAN 2** 8](#_Toc157375157)

[**1.** **Materi IF, Loop, Array, Tkinker** 8](#_Toc157375158)

[**a.** **Luas Permukaan Bangun Ruang Dan Volume Berbasis Tkinker.** 8](#_Toc157375159)

[**b.** **Pembahasan** 8](#_Toc157375160)

[**c.** **Latihan** 14](#_Toc157375161)

[**F.** **PERTEMUAN 3** 15](#_Toc157375162)

[**1.** **Materi Function** 15](#_Toc157375163)

[**a.** **Membuat Aplikasi Perhitungan Menggunakan KivyMD.** 15](#_Toc157375164)

[**b.** **Pembahasan** 16](#_Toc157375165)

[**c.** **Latihan** 22](#_Toc157375166)

[**G.** **PERTEMUAN 4** 24](#_Toc157375167)

[**1.** **Module** 24](#_Toc157375168)

[**a.** **Import Modul Tertentu & Import Massive.** 24](#_Toc157375169)

[**b.** **Pembahasan** 25](#_Toc157375170)

[**d.** **Latihan** 30](#_Toc157375171)

[**H.** **PERTEMUAN 5** 31](#_Toc157375172)

[**1.** **Quiz** 31](#_Toc157375173)

[**a.** **Quizziz.** 31](#_Toc157375174)

[**b.** **Latihan** 32](#_Toc157375175)

[**I.** **PERTEMUAN 6** 33](#_Toc157375176)

[**1.** **Module** 33](#_Toc157375177)

[**a.** **Materi Aplikasi Bantu (Tools)** 33](#_Toc157375178)

[**b.** **Pembahasan** 33](#_Toc157375179)

[**J.** **PERTEMUAN 7** 50](#_Toc157375180)

[**1.** **Materi Konverter Suhu Berbasis Tkinter** 50](#_Toc157375181)

[**a.** **Luas Permukaan Bangun Ruang Dan Volume Berbasis Tkinker.** 50](#_Toc157375182)

[**b.** **Pembahasan** 50](#_Toc157375183)

[**c.** **Latihan** 51](#_Toc157375184)

[**K.** **PERTEMUAN 8** 52](#_Toc157375185)

[**Ujian Tengah Semester** 52](#_Toc157375186)

[**L.** **PERTEMUAN 9** 53](#_Toc157375187)

[**1.** **Materi Class Dan Object** 53](#_Toc157375188)

[**a.** **Belajar Membuat Aplikasi Berbasis Database.** 53](#_Toc157375189)

[**b.** **Pembahasan** 53](#_Toc157375190)

[**d.** **Latihan** 56](#_Toc157375191)

[**M.** **PERTEMUAN 10** 57](#_Toc157375192)

[**1.** **Pengenalan Python Database Menggunakan XAMPP** 57](#_Toc157375193)

[**a.** **Apache HTTP Server** 57](#_Toc157375194)

[**b.** **MySQlL** 57](#_Toc157375195)

[**c.** **PHP** 57](#_Toc157375196)

[**d.** **Perl** 58](#_Toc157375197)

[**e.** **phpMyAdmin** 58](#_Toc157375198)

[**2.** **Langkah-langkah instalisasi XAMPP :** 59](#_Toc157375199)

[**N.** **PERTEMUAN 11** 61](#_Toc157375200)

[**1.** **MySQL Membuat Database Menggunakan CMD** 61](#_Toc157375201)

[**a.** **Python Database.** 61](#_Toc157375202)

[**b.** **Pembahasan** 61](#_Toc157375203)

[**4)** **Latihan** 66](#_Toc157375204)

[**O.** **PERTEMUAN 12** 67](#_Toc157375205)

[**1.** **Python Database** 67](#_Toc157375206)

[**a.** **Python Application Database Using Remote Mysql Server On freedb.tech** 67](#_Toc157375207)

[**b.** **Pembahasan** 67](#_Toc157375208)

[**P.** **PERTEMUAN 13** 73](#_Toc157375209)

[**1.** **Membuat Aplikasi Login Multi-User Python-Mysql** 73](#_Toc157375210)

[**2.** **Langkah Instalasi** 73](#_Toc157375211)

[**3.** Hasil 74](#_Toc157375212)

[**Q.** **PERTEMUAN 14** 76](#_Toc157375213)

[**Membuat Buku Modul Praktikum 14 Pertemuan** 76](#_Toc157375214)

# **KATA PENGANTAR**

Dengan tulus ikhlas dan penuh syukur, marilah kita sambut kehadiran Allah Subhanahu wa Ta'ala, Sang Maha Pencipta, Pemelihara, dan Pemberi Rahmat. Keagungan-Nya meliputi segala aspek kehidupan, membawa cahaya petunjuk dan kasih sayang yang tiada tara. Dalam setiap hembusan angin, gemercik air, dan dedaunan yang berbisik, kita merasakan kehadiran-Nya yang begitu dekat dan menghiasi setiap sudut dunia.

Allah, Sang Pencipta alam semesta ini, hadir dalam setiap detak jantung, dalam setiap keindahan yang terpancar dari ciptaan-Nya. Kehadiran-Nya adalah sumber kebijaksanaan, inspirasi, dan kekuatan bagi setiap jiwa yang mencari-Nya dengan tulus. Dalam keseharian kita, Allah hadir sebagai pilar yang kokoh, memberikan arahan dan bimbingan melalui petunjuk-Nya yang agung.

Semoga kehadiran Allah swt. senantiasa menjadi sumber kebahagiaan, ketenangan, dan keberkahan bagi kita semua. Di setiap langkah perjalanan hidup ini, marilah kita merenung dan mensyukuri nikmat-Nya yang tiada terhingga. Dengan kehadiran-Nya, kita berharap dapat menapaki jalan hidup yang penuh berkah dan kemuliaan. Subhanallah, segala puji bagi Allah yang Maha Agung dan Maha Penyayang.

# **PEMBAHASAN**

## **Python**

Python adalah bahasa pemrograman tingkat tinggi yang dikenal dengan sintaksis yang bersih, mudah dibaca, dan ekspresif. Beberapa pendapat para ahli dan sumber terpercaya tentang Python melibatkan penghargaan terhadap kemudahan penggunaan, fleksibilitas, dan kekuatan bahasa ini.

### **Pustaka atau Modul (Library or Module):**

Dalam bahasa pemrograman Python, pustaka atau modul merujuk pada kumpulan fungsi, kelas, dan variabel yang dapat digunakan untuk menyelesaikan tugas tertentu. Pustaka ini dapat mencakup berbagai jenis fungsionalitas, seperti pemrosesan data, komunikasi jaringan, atau pengelolaan tautan database.

### **Paket (Package):**

Sebuah paket adalah cara untuk mengorganisir modul-modul terkait ke dalam struktur direktori yang terstruktur. Dengan mengelompokkan modul-modul terkait ke dalam paket, kita dapat menciptakan hierarki yang lebih teratur dan mempermudah pengelolaan kode.

### **Pip (Python Package Installer):**

Pip adalah alat yang digunakan dalam ekosistem Python untuk menginstal dan mengelola paket atau modul. Dengan menggunakan pip, Anda dapat dengan mudah mengunduh dan menginstal pustaka-pustaka tambahan dari Python Package Index (PyPI), repositori paket Python yang luas.

#### **Contoh penggunaan pip:**

pip install nama-paket

Listing 1 Code Pip Install

Dalam perintah di atas, "nama-paket" akan digantikan dengan nama pustaka atau modul yang ingin Anda instal. Pip kemudian akan mengunduh paket tersebut dari PyPI dan menginstalnya di lingkungan Python Anda.

Jadi, secara ringkas, pip membantu Anda mengelola pustaka atau modul Python dengan cara yang efisien, memungkinkan integrasi mudah dengan kode yang Anda kembangkan.

## **Pemrograman** **Berorientasi** **Objek**

Dalam pemrograman berorientasi objek, setiap objek apapun akan memiliki data kit (sifat, berupa variabel maupun konstanta) dan metode (perilaku atau kemampuan melakukan sesuatu, berupa fungsi). Jadi, objek dapat didefinisikan sebagai suatu entitas yang memiliki data dan metode.

Semua bahasa pemrograman berorientasi objek menyediakan mekanisme untuk dapat membantu mengimplementasikan model yang berorientasi objek. Prinsip tersebut adalah encapulation, inheritence, dan polymorphism.

### **Encapsulation (Enkapsulasi):**

Enkapsulasi melibatkan pembungkusan data dan metode yang beroperasi pada data ke dalam satu unit tunggal, yaitu objek. Dengan kata lain, enkapsulasi memungkinkan kita menyembunyikan rincian implementasi internal objek dan hanya mengekspos fungsionalitas yang diperlukan. Ini membantu dalam menciptakan antarmuka yang bersih dan meningkatkan keamanan serta pemeliharaan kode.

### **Inheritance (Warisan):**

Warisan memungkinkan suatu objek untuk mengakses sifat dan metode dari objek lainnya. Objek yang mewarisi disebut anak kelas (subclass), dan objek yang memberikan warisan disebut kelas induk atau superclass. Dengan menggunakan warisan, kita dapat membangun hierarki kelas yang memfasilitasi penggunaan kembali kode dan mengorganisasi struktur program secara hierarkis.

### **Polymorphism (Polimorfisme):**

Polimorfisme mengacu pada kemampuan suatu metode atau fungsi untuk bekerja dengan cara yang berbeda tergantung pada konteks pemanggilan. Dalam konteks pemrograman berorientasi objek, polimorfisme memungkinkan suatu metode memiliki nama yang sama tetapi perilaku yang berbeda tergantung pada kelas objek yang memanggilnya. Hal ini dapat meningkatkan fleksibilitas dan modularitas kode.

Pemrograman berorientasi objek bertujuan untuk menciptakan struktur program yang lebih intuitif, mudah dimengerti, dan mudah dipelihara. Dengan mengikuti prinsip-prinsip ini, pengembang dapat membuat sistem yang lebih modular, bersih, dan dapat diubah sesuai kebutuhan.

# **INSTALASI**

## **Instalasi Pyton**

Berikut adalah panduan langkah demi langkah untuk mengunduh dan menginstal Python pada sistem operasi Windows:

### **Unduh Python dari Situs Resmi**

Buka browser web dan kunjungi situs resmi Python: Python Official Website. Pada halaman utama, arahkan kursor ke menu "Downloads" dan klik "Python 3.x.x" (versi terbaru saat ini) di bagian tengah halaman.

### **Pilih Installer yang Tepat**

Setelah mengklik versi Python yang diinginkan, Anda akan diarahkan ke halaman unduhan. Gulir ke bawah hingga menemukan dua opsi installer: "Windows Installer" dan "Windows x86-64 executable installer". Pilih installer sesuai dengan arsitektur sistem Anda (32-bit atau 64-bit).

Catatan: *Jika Anda tidak yakin arsitektur sistem Anda, Anda dapat memeriksanya dengan klik kanan pada "This PC" atau "Computer" di desktop atau file explorer, lalu pilih "Properties".*

### **Unduh Installer Python**

Klik pada installer yang telah dipilih. Ini akan memulai proses pengunduhan installer Python.

### **Jalankan Installer Python**

Setelah selesai mengunduh, buka file installer Python yang baru diunduh. Ini biasanya berada di direktori "Downloads". Pastikan opsi "Add Python 3.x to PATH" dicentang. Ini akan memudahkan akses ke Python dari command prompt. Klik "Install Now" untuk memulai proses instalasi.

### **Proses Instalasi**

Installer akan mulai menginstal Python di sistem Anda. Proses ini mungkin memakan beberapa waktu. Pastikan opsi "Disable path length limit" dicentang (opsional, tetapi disarankan). Setelah selesai, centang opsi "Disable path length limit" dan klik "Close".

### **Verifikasi Instalasi**

Buka command prompt atau PowerShell dan ketik :

python --version atau python -V.

Listing 2 Verifikasi Instalasi Python

Ini seharusnya dapat menampilkan versi Python yang baru diinstal.Selanjutnya, ketik:

python --version atau python -V.

Listing 3 Check Python Version

Untuk memastikan bahwa pip (Python Package Installer) juga diinstal. Selamat, Anda telah berhasil mengunduh dan menginstal Python pada sistem Windows Anda. Sekarang Anda dapat mulai menggunakan Python untuk pengembangan perangkat lunak.

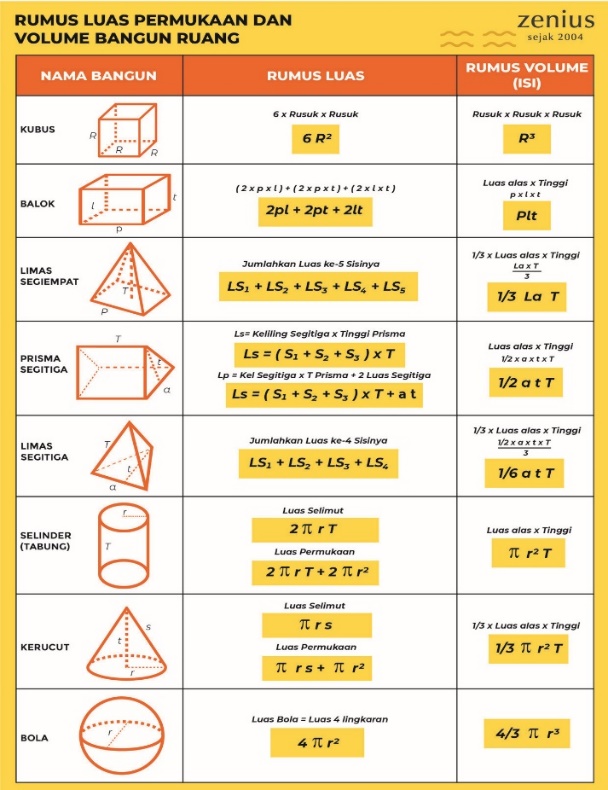
# **PERTEMUAN 1**

## **Materi Pengenalan Python (Variabel, Tipe Data, Operator)**

### **Rumus Luas Permukaan Bangun Ruang Dan Volume.**

Setelah pembahasan pengertian python dan langkah instalasi didalam pembahasan sebelumnya kita akan mencoba sebuah latihan, latihan ini bertujuan untuk membuat kode perhitungan luas bangun ruang dan volume dengan rumus matematika dasar dengan menggunakan kode python secara sederhana, untuk untuk dapat mempelajarinya kita harus mengerti terlebih dahulu konsep dasar dari bahasa pemograman python dan pastikan anda sudah mengikuti langkah-langkah instalasi python didalam pembahasan sebelumnya.

Untuk menghitung luas dan volume bangun ruang kita harus mengetahui rumus dari bangun ruang yang akan kita hitung menggunakan pemograman python, gambar rumus dapat dilihat sebagai berikut :



Gambar 2 Menghitung Luas Permukan Dan Volume Tabung

### **Pembahasan**

Berikut adalah contoh bagaimana kita dapat membuat aplikasi berbasis python untuk menghitung luas dan volume bangun ruang :

#### **Menghitung Luas Permukaan Dan Volume Tabung**

Dibawah ini adalah source code python lengkap untuk membuat aplikasi menghitung luas permukaan dan volume tabung, kodenya dapat dilihat sebagai berikut :

import math

print("Aplikasi menghitung luas permukaan dan volume tabung")

# Atur nilai variabel

jari\_jari = 5.0

tinggi = 10.0

# Rumus

luas\_permukaan = 2 \* math.pi \* jari\_jari \* (jari\_jari + tinggi)

volume = math.pi \* jari\_jari\*\*2 \* tinggi

# Output

print("Jari-Jari Tabung =", jari\_jari)

print("Tinggi Tabung =", tinggi)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 4 Source Code Menghitung Luas Permukan Dan Volume Tabung

Penjelasan dapat anda simak sebagai berikut:

##### **Module**

kode dibawah ini adalah bagian module untuk mengimport modul math bawaan dari python, module ini digunakan untuk menghitung rumus perhitungan termasuk pada luas permukan dan volume tabung.

import math

Listing 5 Source Code Import Modul Math

##### **Print (“Judul”)**

kode dibawah ini adalah bagian untuk mengatur judul karena kita menaruhnya dibagian atas maka kode ini akan otomatis di print (ditampilkan) pada saat kode pertama kali dimuat untuk menjadi judul.

print("Aplikasi menghitung luas permukaan dan volume tabung")

Listing 6 Source Code print (“Judul”)

##### **Atur Nilai Variabel**

kode dibawah ini adalah bagian untuk mengatur nilai variabel pada python kode ini digunakan untuk memberikan value atau nilai yang ingin dihitung hasilnya dari aplikasi ini.

# Atur nilai variabel

jari\_jari = 5.0

tinggi = 10.0

Listing 7 Source Code Nilai Variabel

##### **Rumus**

kode dibawah ini adalah bagian untuk mengatur dan memasukan rumus perhitungan pada aplikasi python untuk menghitung luas permukan dan volume tabung.

# Rumus

luas\_permukaan = 2 \* math.pi \* jari\_jari \* (jari\_jari + tinggi)

volume = math.pi \* jari\_jari\*\*2 \* tinggi

Listing 8 Source Code Nilai Variabel

##### **Output**

kode dibawah ini adalah bagian untuk mengatur output yang telah diambil dari nilai variable dan rumus perhitungan pada aplikasi python untuk menghitung luas permukan dan volume tabung dan kemudian menampilkanya dalam bentuk teks.

# Output

print("Jari-Jari Tabung =", jari\_jari)

print("Tinggi Tabung =", tinggi)

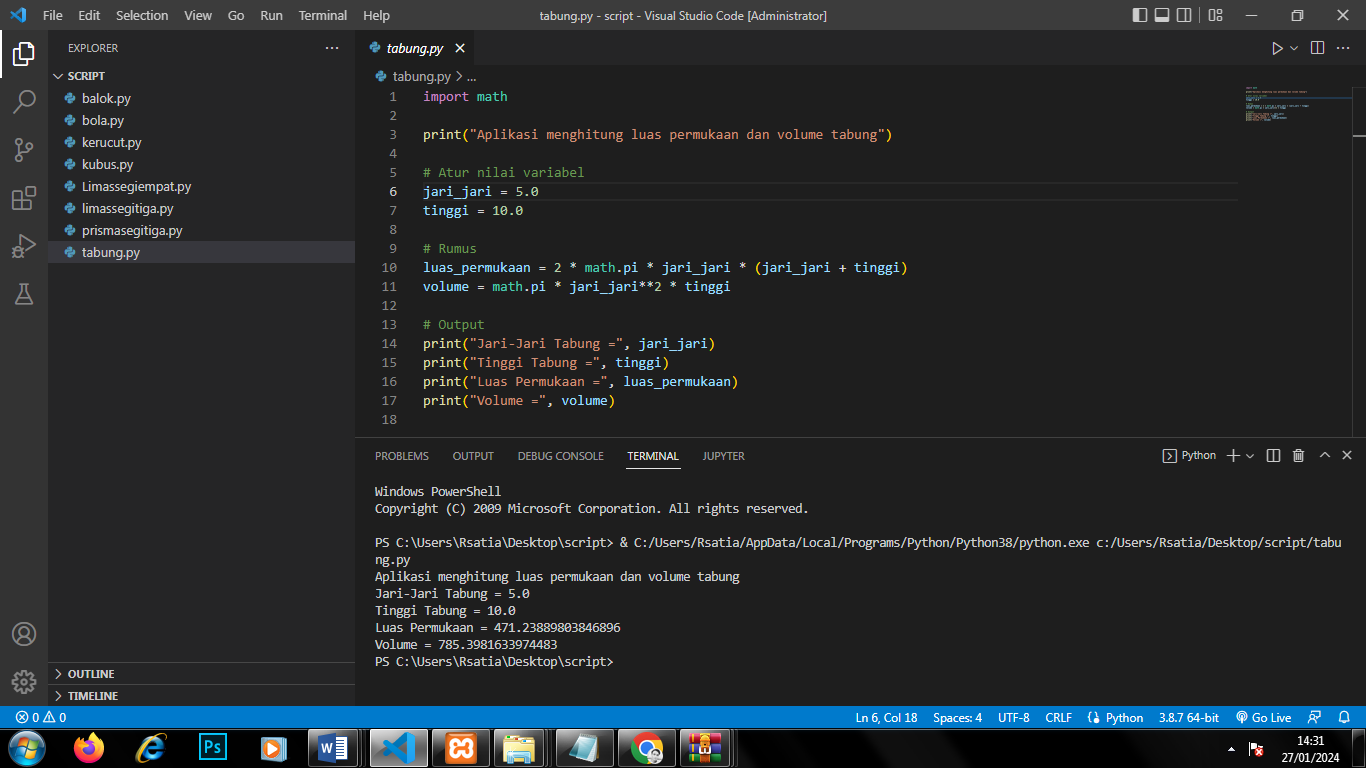
print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 9 Source Code Nilai Variabel

##### **Hasil**

Berikut adalah gambar kode dari menghitung luas permukan dan volume tabung yang berhasil dijalankan menggunakan source code sebelumnya.



Gambar 2 Menghitung Luas Permukan Dan Volume Tabung

### **Latihan**

Setelah kita belajar bagaimana struktur dan fungsi didalam pertemuan ini mari kita coba untuk menggunakan listing code yang lain, berikut adalah beberapa source code listing yang bisa anda coba dirumah untuk latihan :

#### **Menghitung Luas Permukaan Dan Volume Balok**

print("Aplikasi menghitung luas permukaan dan volume balok")

# Atur nilai variabel

panjang = 20

lebar = 4

tinggi = 6

# Rumus

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

volume = panjang \* lebar \* tinggi

# Output

print("Panjang =", panjang)

print("Lebar =", lebar)

print("Tinggi =", tinggi)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 10 Source Code Balok

#### **Menghitung Luas Permukaan Dan Volume Bola**

import math

print("Aplikasi menghitung luas permukaan dan volume bola")

# Atur nilai variabel

jari\_jari = 3.0

# Rumus

luas\_permukaan = 4 \* math.pi \* jari\_jari\*\*2

volume = (4/3) \* math.pi \* jari\_jari\*\*3

# Output

print("Jari-Jari Bola =", jari\_jari)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 11 Source Code Bola

#### **Menghitung Luas Permukaan Dan Volume Kubus**

print("Aplikasi menghitung luas permukaan dan volume kubus")

# Atur nilai variabel

sisi = 6

# Rumus

luas\_permukaan = 6 \* sisi\*\*2

volume = sisi\*\*3

# Output

print("Panjang Sisi =", sisi)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 12 Source Code Kubus

#### **Menghitung Luas Permukaan Dan Volume Kerucut**

import math

print("Aplikasi menghitung luas permukaan dan volume kerucut")

# Atur nilai variabel

jari\_jari = 4.0

tinggi = 7.0

# Hitung panjang sisi kerucut

sisi = math.sqrt(jari\_jari\*\*2 + tinggi\*\*2)

# Rumus

luas\_permukaan = math.pi \* jari\_jari \* (jari\_jari + sisi)

volume = (1/3) \* math.pi \* jari\_jari\*\*2 \* tinggi

# Output

print("Jari-Jari Kerucut =", jari\_jari)

print("Tinggi Kerucut =", tinggi)

print("Panjang Sisi Kerucut (s) =", sisi)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 13 Source Code Kerucut

#### **Menghitung Luas Permukaan Dan Volume Limas Segi Empat**

print("Aplikasi menghitung luas permukaan dan volume limas segi empat")

# Atur nilai variabel

luas\_alas = 16

panjang\_sisi = 4

tinggi = 8

# Rumus

luas\_permukaan = luas\_alas + 4 \* (0.5 \* panjang\_sisi \* tinggi)

volume = (1/3) \* luas\_alas \* tinggi

# Output

print("Luas Alas =", luas\_alas)

print("Panjang Sisi Dasar =", panjang\_sisi)

print("Tinggi Limas =", tinggi)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 14 Source Code Limas Segi Empat

#### **Menghitung Luas Permukaan Dan Volume Prisma Segitiga**

print("Aplikasi menghitung luas permukaan dan volume prisma segitiga")

# Atur nilai variabel

luas\_segitiga\_alas = 24

keliling\_segitiga\_alas = 18

tinggi\_prisma = 10

# Rumus

luas\_permukaan = (2 \* luas\_segitiga\_alas) + (keliling\_segitiga\_alas \* tinggi\_prisma)

volume = luas\_segitiga\_alas \* tinggi\_prisma

# Output

print("Luas Segitiga Alas =", luas\_segitiga\_alas)

print("Keliling Segitiga Alas =", keliling\_segitiga\_alas)

print("Tinggi Prisma =", tinggi\_prisma)

print("Luas Permukaan =", luas\_permukaan)

print("Volume =", volume)

Listing 15 Source Code Prisma Segitiga

# **PERTEMUAN 2**

## **Materi IF, Loop, Array, Tkinker**

### **Luas Permukaan Bangun Ruang Dan Volume Berbasis Tkinker.**

Latihan ini bertujuan untuk membuat kode perhitungan luas bangun ruang dan volume dengan rumus matematika dasar dengan menggunakan kode python menggunakan jendela modul tkinker menggunakan logika IF, Loop, array, dengan meneruskan materi dari pertemuan sebelumnya.

Untuk membuat aplikasi menghitung luas dan volume bangun ruang menggunakan modul tkinker kita harus menginstal modul tkinker terlebih dahulu dengan membuka cmd atau powershell :

* *Untuk membuka Command Prompt, tekan tombol Windows + R, ketik cmd, dan tekan Enter.*
* *Untuk membuka PowerShell, tekan tombol Windows + R, ketik powershell, dan tekan Enter.*

Kemudian anda dapat menggunakan kode ini untuk menginstal tkinter, kodenya dapat dilihat sebagai berikut :

python -m tkinter

Listing 16 Instalasi Tkinter

### **Pembahasan**

#### **Pengantar IF Statements:**

Dalam pemrograman, pernyataan IF adalah konstruksi dasar yang memungkinkan eksekusi blok kode tertentu berdasarkan kondisi tertentu. Pernyataan kondisional ini mengevaluasi suatu ekspresi dan, jika benar, menjalankan blok kode yang terkait. Pernyataan IF sangat penting untuk mengendalikan alur program, memungkinkan pengembang membuat aplikasi yang fleksibel dan dinamis.

#### **Peran Loop dalam Pemrograman:**

Loop memainkan peran penting dalam pemrograman dengan memungkinkan eksekusi berulang dari serangkaian pernyataan. Loop FOR, misalnya, umumnya digunakan untuk mengulang melalui suatu urutan (seperti array atau rentang angka) dan melakukan operasi tertentu selama setiap iterasi. Di sisi lain, loop WHILE melanjutkan eksekusi selama kondisi tertentu tetap benar. Konstruksi perulangan ini berkontribusi pada efisiensi dan fleksibilitas program, memungkinkan otomatisasi tugas-tugas yang repetitif.

#### **Array: Struktur Data Yang Kuat:**

Array adalah struktur data penting yang menyimpan koleksi elemen, memungkinkan manajemen data yang efisien. Dalam bahasa pemrograman, array dapat berupa berbagai jenis, termasuk array numerik, array string, atau array objek kustom. Kemampuan untuk mengorganisir dan mengakses data melalui array menyederhanakan kode, meningkatkan keterbacaan, dan memfasilitasi manipulasi dataset besar. Memahami cara mendeklarasikan, menginisialisasi, dan memanipulasi array sangat penting untuk pemrograman yang efektif.  
Berikut adalah contoh bagaimana kita dapat membuat aplikasi berbasis python untuk menghitung luas dan volume bangun ruang menggunakan window dan logika if, loop array tkinter :

#### **Luas Permukaan Dan Volume Tabung & Balok Berbasis Tkinter**

Dibawah ini adalah source code python lengkap untuk membuat aplikasi menghitung luas permukaan dan volume tabung menggunakan window thinker dan logika if, loop array, dapat dilihat sebagai berikut :

import math

import tkinter as tk

from tkinter import Label, Entry, Button

class Shape:

def \_\_init\_\_(self, jenis, window, index):

self.jenis = jenis

self.window = window

self.index = index

self.labels = [] # to store labels associated with each input field

self.create\_widgets()

def create\_widgets(self):

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Panjang:"))

self.labels[-1].pack()

self.entry\_dim1 = Entry(self.window)

self.entry\_dim1.pack()

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Tinggi:"))

self.labels[-1].pack()

self.entry\_dim2 = Entry(self.window)

self.entry\_dim2.pack()

if self.jenis == 'balok':

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Lebar:"))

self.labels[-1].pack()

self.entry\_dim3 = Entry(self.window)

self.entry\_dim3.pack()

def destroy\_widgets(self):

for label in self.labels:

label.pack\_forget()

self.entry\_dim1.pack\_forget()

if hasattr(self, 'entry\_dim2'):

self.entry\_dim2.pack\_forget()

if hasattr(self, 'entry\_dim3'):

self.entry\_dim3.pack\_forget()

def calculate(self):

if self.jenis == 'tabung':

r = float(self.entry\_dim1.get())

h = float(self.entry\_dim2.get())

luas\_permukaan = 2 \* math.pi \* r \* (r + h)

volume = math.pi \* r\*\*2 \* h

elif self.jenis == 'balok':

panjang = float(self.entry\_dim1.get())

tinggi = float(self.entry\_dim2.get())

lebar = float(self.entry\_dim3.get())

luas\_permukaan = 2 \* (panjang \* tinggi + panjang \* lebar + tinggi \* lebar)

volume = panjang \* tinggi \* lebar

else:

return # Unknown shape

hasil\_text = f"{self.jenis.capitalize()} {self.index} - Luas Permukaan: {luas\_permukaan:.2f}, Volume: {volume:.2f}\n"

return hasil\_text

class App:

def \_\_init\_\_(self, master):

self.master = master

master.title("Aplikasi Tabung dan Balok")

master.geometry("400x400")

self.shapes = []

# Create buttons

Button(master, text="Tambah Tabung", command=lambda: self.add\_shape('tabung')).pack()

Button(master, text="Tambah Balok", command=lambda: self.add\_shape('balok')).pack()

Button(master, text="Hapus", command=self.remove\_shape).pack()

Button(master, text="Hitung", command=self.calculate).pack()

# Label for displaying results

self.label\_hasil = Label(master, text="")

self.label\_hasil.pack()

def add\_shape(self, jenis):

# Hapus objek balok tanpa field jika ada

self.shapes = [shape for shape in self.shapes if not (shape.jenis == 'balok' and not hasattr(shape, 'entry\_dim2'))]

new\_shape = Shape(jenis, self.master, len(self.shapes) + 1)

self.shapes.append(new\_shape)

def remove\_shape(self):

if self.shapes:

shape\_to\_remove = self.shapes.pop()

shape\_to\_remove.destroy\_widgets()

def calculate(self):

hasil\_text = ""

for index, shape in enumerate(self.shapes, start=1):

result = shape.calculate()

if result:

hasil\_text += result

self.label\_hasil.config(text=hasil\_text)

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

root = tk.Tk()

app = App(root)

root.mainloop()

elif self.jenis == 'balok':

panjang = float(self.entry\_dim1.get())

tinggi = float(self.entry\_dim2.get())

lebar = float(self.entry\_dim3.get())

luas\_permukaan = 2 \* (panjang \* tinggi + panjang \* lebar + tinggi \* lebar)

volume = panjang \* tinggi \* lebar

else:

return # Unknown shape

hasil\_text = f"{self.jenis.capitalize()} {self.index} - Luas Permukaan: {luas\_permukaan:.2f}, Volume: {volume:.2f}\n"

return hasil\_text

class App:

def \_\_init\_\_(self, master):

self.master = master

master.title("Aplikasi Tabung dan Balok")

master.geometry("400x400")

self.shapes = []

# Create buttons

Button(master, text="Tambah Tabung", command=lambda: self.add\_shape('tabung')).pack()

Button(master, text="Tambah Balok", command=lambda: self.add\_shape('balok')).pack()

Button(master, text="Hapus", command=self.remove\_shape).pack()

Button(master, text="Hitung", command=self.calculate).pack()

# Label for displaying results

self.label\_hasil = Label(master, text="")

self.label\_hasil.pack()

def add\_shape(self, jenis):

# Hapus objek balok tanpa field jika ada

self.shapes = [shape for shape in self.shapes if not (shape.jenis == 'balok' and not hasattr(shape, 'entry\_dim2'))]

new\_shape = Shape(jenis, self.master, len(self.shapes) + 1)

self.shapes.append(new\_shape)

def remove\_shape(self):

if self.shapes:

shape\_to\_remove = self.shapes.pop()

shape\_to\_remove.destroy\_widgets()

def calculate(self):

hasil\_text = ""

for index, shape in enumerate(self.shapes, start=1):

result = shape.calculate()

if result:

hasil\_text += result

self.label\_hasil.config(text=hasil\_text)

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

root = tk.Tk()

app = App(root)

root.mainloop()

def calculate(self):

hasil\_text = ""

for index, shape in enumerate(self.shapes, start=1):

result = shape.calculate()

if result:

hasil\_text += result

self.label\_hasil.config(text=hasil\_text)

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

root = tk.Tk()

app = App(root)

root.mainloop()

Listing 17 Source Code Menghitung Luas Permukan Dan Volume Tabung

Penjelasan dapat anda simak sebagai berikut :

##### **Import Module**

kode dibawah ini adalah bagian module untuk mengimport modul math bawaan dari python untuk menghitung rumus perhitungan termasuk pada luas permukan dan volume tabung, kemudian import tkhinter module ini digunakan untuk

membuat antarmuka grafis sederhana (GUI) menggunakan modul tkinter. Program ini menciptakan jendela dengan kotak input, tombol "Hitung", dan label untuk menampilkan hasil perhitungan akar kuadrat dari angka yang dimasukkan oleh pengguna.

import math

import tkinter as tk

from tkinter import Label, Entry, Button

Listing 18 Source Code Import Module

##### **Class Shape**

Class ini digunakan untuk mewakili objek-objek geometris seperti tabung dan balok dalam aplikasi tkinter. Setiap objek Shape memiliki properti seperti jenis (tabung atau balok), jendela tempat objek tersebut ditampilkan, dan indeks untuk identifikasi. Kelas ini memiliki metode untuk membuat elemen-elemen antarmuka pengguna (widgets) seperti label dan entry, menghitung luas permukaan dan volume berdasarkan input pengguna, serta menghapus elemen-elemen tersebut dari tampilan jika diperlukan. Dengan adanya kelas Shape, aplikasi dapat membuat, mengelola, dan menghitung properti geometris dari objek-objek yang berbeda secara dinamis, berikut adalah potongan kodenya :

class Shape:

def \_\_init\_\_(self, jenis, window, index):

self.jenis = jenis

self.window = window

self.index = index

self.labels = [] # to store labels associated with each input field

self.create\_widgets()

def create\_widgets(self):

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Panjang:"))

self.labels[-1].pack()

self.entry\_dim1 = Entry(self.window)

self.entry\_dim1.pack()

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Tinggi:"))

self.labels[-1].pack()

self.entry\_dim2 = Entry(self.window)

self.entry\_dim2.pack()

if self.jenis == 'balok':

self.labels.append(Label(self.window, text=f"{self.jenis.capitalize()} {self.index} - Lebar:"))

self.labels[-1].pack()

self.entry\_dim3 = Entry(self.window)

self.entry\_dim3.pack()

def destroy\_widgets(self):

for label in self.labels:

label.pack\_forget()

self.entry\_dim1.pack\_forget()

if hasattr(self, 'entry\_dim2'):

self.entry\_dim2.pack\_forget()

if hasattr(self, 'entry\_dim3'):

self.entry\_dim3.pack\_forget()

def calculate(self):

if self.jenis == 'tabung':

r = float(self.entry\_dim1.get())

h = float(self.entry\_dim2.get())

luas\_permukaan = 2 \* math.pi \* r \* (r + h)

volume = math.pi \* r\*\*2 \* h

elif self.jenis == 'balok':

panjang = float(self.entry\_dim1.get())

tinggi = float(self.entry\_dim2.get())

lebar = float(self.entry\_dim3.get())

luas\_permukaan = 2 \* (panjang \* tinggi + panjang \* lebar + tinggi \* lebar)

volume = panjang \* tinggi \* lebar

else:

return # Unknown shape

hasil\_text = f"{self.jenis.capitalize()} {self.index} - Luas Permukaan: {luas\_permukaan:.2f}, Volume: {volume:.2f}\n"

return hasil\_text

Listing 19 Source Code Class Shape

##### **Class App**

Class App ini berperan untuk bertanggung jawab sebagai pengelola antarmuka pengguna dan kontrol utama aplikasi. Fungsinya meliputi inisialisasi atribut utama, penambahan dan penghapusan objek bentuk, serta tampilan hasil perhitungan di antarmuka aplikasi, berikut adalah potongan kodenya :

class App:

def \_\_init\_\_(self, master):

self.master = master

master.title("Aplikasi Tabung dan Balok")

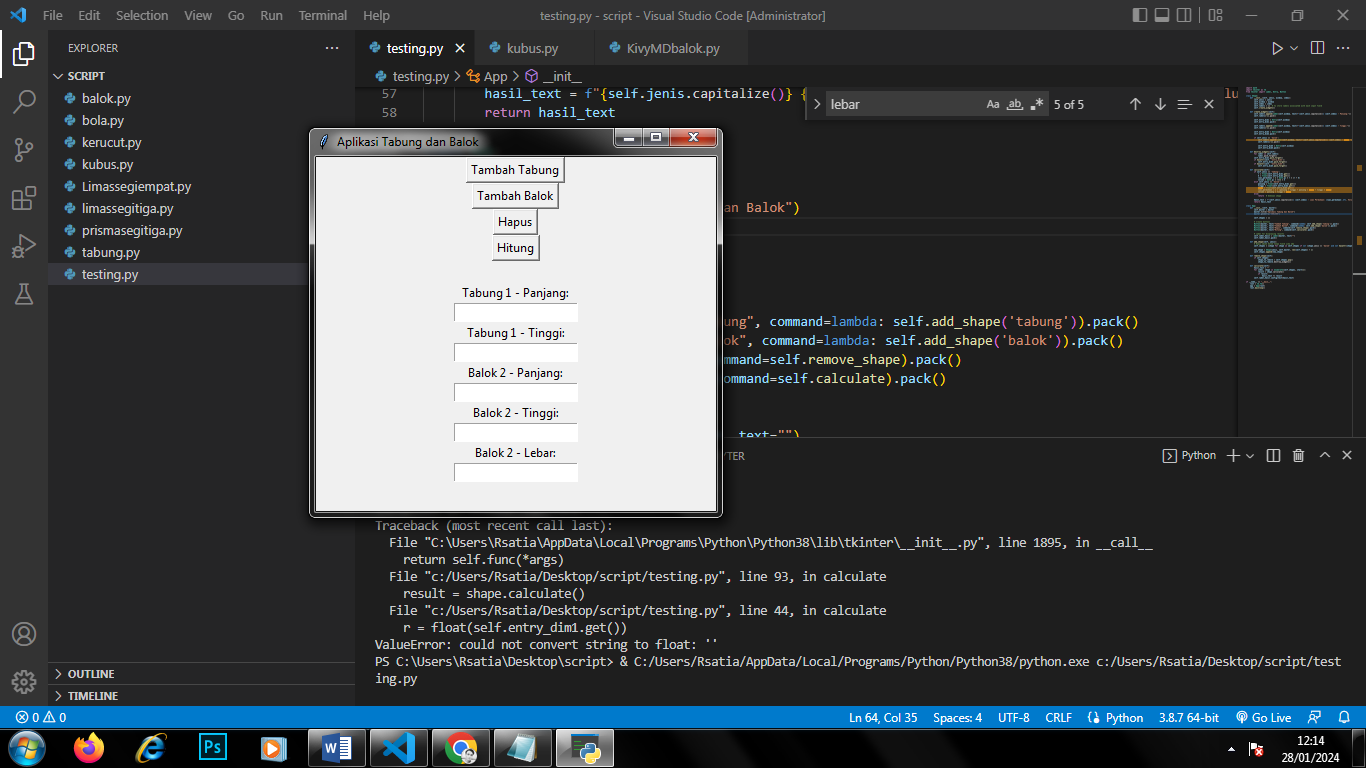
master.geometry("400x400")

self.shapes = []

Listing 20 Source Code Class App

##### **Hasil**

Berikut adalah gambar kode dari menghitung luas permukan dan volume tabung yang berhasil dijalankan menggunakan source code sebelumnya.



Gambar 2 Luas Permukan Dan Volume Tabung & Balok Menggunakan Window Tkinter

### **Latihan**

Setelah kita belajar bagaimana struktur dan fungsi didalam pertemuan ini mari kita coba untuk menggunakan listing code yang lain, gunakanlah kode dari pertemuan sebelumnya untuk latihan!.

# **PERTEMUAN 3**

## **Materi Function**

Function adalah blok kode terorganisir yang dapat dieksekusi secara terpisah untuk melakukan tugas tertentu. Fungsi dapat menerima input (argumen), melakukan operasi, dan mengembalikan output. Keunggulan penggunaan fungsi termasuk pengorganisasian kode, pengurangan pengulangan kode yang sering, dan kemampuan untuk membuat kode lebih mudah dimengerti dan dikelola. Fungsi didefinisikan menggunakan kata kunci def diikuti oleh nama fungsi, parameter, dan blok kode yang akan dieksekusi.

### **Membuat Aplikasi Perhitungan Menggunakan KivyMD.**

Pada Pertemuan kali ini kita akan belajar membuat aplikasi perhitungan menggunakan KivyMD.

Untuk membuat aplikasi perhitungan luas dan volume bangun ruang menggunakan modul KivyMD kita harus menginstal modul KivyMD terlebih dahulu dengan membuka cmd atau powershell :

* *Untuk membuka Command Prompt, tekan tombol Windows + R, ketik cmd, dan tekan Enter.*
* *Untuk membuka PowerShell, tekan tombol Windows + R, ketik powershell, dan tekan Enter.*

Kemudian anda dapat menggunakan kode ini untuk menginstal tkinter, kodenya dapat dilihat sebagai berikut :

pip install kivymd

Listing 21 Instalasi KivyMD

### **Pembahasan**

Berikut adalah contoh untuk membuat aplikasi berbasis python dan KivyMD untuk menghitung luas dan volume bangun ruang :

#### **Aplikasi Perhitungan Balok Menggunakan KivyMD**

from kivymd.app import MDApp

from kivymd.uix.screen import MDScreen

from kivymd.uix.gridlayout import MDGridLayout

from kivymd.uix.button import MDRaisedButton

from kivymd.uix.textfield import MDTextField

from kivymd.uix.label import MDLabel

class CalculatorApp(MDApp):

def build(self):

self.title = "Aplikasi Menghitung Luas Dan Volume Balok"

self.theme\_cls.primary\_palette = "Blue"

layout = MDScreen()

# Title Labels

title\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=300, height=100, pos\_hint={'center\_x': 0.5, 'center\_y': 0.9})

title\_label = MDLabel(

text="Hitung Luas Dan Volume Balok",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

title\_box.add\_widget(title\_label)

layout.add\_widget(title\_box)

# Input Fields

input\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=300, height=250, pos\_hint={'center\_x': 0.5, 'center\_y': 0.6})

self.panjang\_text = MDTextField(

hint\_text="Panjang (cm)",

helper\_text="Enter the length",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.lebar\_text = MDTextField(

hint\_text="Lebar (cm)",

helper\_text="Enter the width",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.tinggi\_text = MDTextField(

hint\_text="Tinggi (cm)",

helper\_text="Enter the height",

helper\_text\_mode="on\_focus",

input\_type="number"

calculate\_button = MDRaisedButton(text="Calculate")

calculate\_button.bind(on\_release=self.calculate)

reset\_button = MDRaisedButton(text="Reset")

reset\_button.bind(on\_release=self.reset)

button\_box.add\_widget(calculate\_button)

button\_box.add\_widget(reset\_button)

layout.add\_widget(button\_box)

# Result Labels

result\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=500, height=200, pos\_hint={'center\_x': 0.5, 'center\_y': 0.38})

self.luas\_permukaan\_label = MDLabel(

text="Luas Permukaan:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

self.volume\_label = MDLabel(

text="Volume:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

# Output Labels

self.panjang\_label = MDLabel()

self.lebar\_label = MDLabel()

self.tinggi\_label = MDLabel()

self.output\_label = MDLabel()

result\_box.add\_widget(self.panjang\_label)

result\_box.add\_widget(self.lebar\_label)

result\_box.add\_widget(self.tinggi\_label)

result\_box.add\_widget(self.luas\_permukaan\_label)

result\_box.add\_widget(self.volume\_label)

result\_box.add\_widget(self.output\_label)

layout.add\_widget(result\_box)

return layout

def calculate(self, instance):

try:

panjang = float(self.panjang\_text.text)

lebar = float(self.lebar\_text.text)

tinggi = float(self.tinggi\_text.text)

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

volume = panjang \* lebar \* tinggi

self.luas\_permukaan\_label.text = f"Luas Permukaan: {luas\_permukaan:.2f} cm^2"

self.volume\_label.text = f"Volume: {volume:.2f} cm^3"

self.output\_label.text = ""

except ValueError:

self.luas\_permukaan\_label.text = "Error: Invalid input"

self.volume\_label.text = "Volume:"

def reset(self, instance):

self.panjang\_text.text = ""

self.lebar\_text.text = ""

self.tinggi\_text.text = ""

self.luas\_permukaan\_label.text = "Luas Permukaan:"

self.volume\_label.text = "Volume:"

self.output\_label.text = ""

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

app = CalculatorApp()

app.run()

Dibawah ini adalah source code python lengkap untuk membuat aplikasi menghitung luas permukaan dan volume tabung, kodenya dapat dilihat sebagai berikut :

)

input\_box.add\_widget(self.panjang\_text)

input\_box.add\_widget(self.lebar\_text)

input\_box.add\_widget(self.tinggi\_text)

layout.add\_widget(input\_box)

# Buttons

button\_box = MDGridLayout(cols=2, spacing=10, size\_hint=(None, None), width=300, height=50, pos\_hint={'center\_x': 0.5, 'center\_y': 0.44})

calculate\_button = MDRaisedButton(text="Calculate")

calculate\_button.bind(on\_release=self.calculate)

reset\_button = MDRaisedButton(text="Reset")

reset\_button.bind(on\_release=self.reset)

button\_box.add\_widget(calculate\_button)

button\_box.add\_widget(reset\_button)

layout.add\_widget(button\_box)

# Result Labels

result\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=500, height=200, pos\_hint={'center\_x': 0.5, 'center\_y': 0.38})

self.luas\_permukaan\_label = MDLabel(

text="Luas Permukaan:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

self.volume\_label = MDLabel(

text="Volume:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

# Output Labels

self.panjang\_label = MDLabel()

self.lebar\_label = MDLabel()

self.tinggi\_label = MDLabel()

self.output\_label = MDLabel()

result\_box.add\_widget(self.panjang\_label)

result\_box.add\_widget(self.lebar\_label)

result\_box.add\_widget(self.tinggi\_label)

result\_box.add\_widget(self.luas\_permukaan\_label)

result\_box.add\_widget(self.volume\_label)

result\_box.add\_widget(self.output\_label)

layout.add\_widget(result\_box)

return layout

def calculate(self, instance):

try:

panjang = float(self.panjang\_text.text)

lebar = float(self.lebar\_text.text)

tinggi = float(self.tinggi\_text.text)

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

volume = panjang \* lebar \* tinggi

self.luas\_permukaan\_label.text = f"Luas Permukaan: {luas\_permukaan:.2f} cm^2"

self.volume\_label.text = f"Volume: {volume:.2f} cm^3"

self.output\_label.text = ""

except ValueError:

self.luas\_permukaan\_label.text = "Error: Invalid input"

self.volume\_label.text = "Volume:"

def reset(self, instance):

self.panjang\_text.text = ""

self.lebar\_text.text = ""

self.tinggi\_text.text = ""

self.luas\_permukaan\_label.text = "Luas Permukaan:"

self.volume\_label.text = "Volume:"

self.output\_label.text = ""

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

app = CalculatorApp()

app.run()

def calculate(self, instance):

try:

panjang = float(self.panjang\_text.text)

lebar = float(self.lebar\_text.text)

tinggi = float(self.tinggi\_text.text)

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

volume = panjang \* lebar \* tinggi

self.luas\_permukaan\_label.text = f"Luas Permukaan: {luas\_permukaan:.2f} cm^2"

self.volume\_label.text = f"Volume: {volume:.2f} cm^3"

self.output\_label.text = ""

except ValueError:

self.luas\_permukaan\_label.text = "Error: Invalid input"

self.volume\_label.text = "Volume:"

def reset(self, instance):

self.panjang\_text.text = ""

self.lebar\_text.text = ""

self.tinggi\_text.text = ""

self.luas\_permukaan\_label.text = "Luas Permukaan:"

self.volume\_label.text = "Volume:"

self.output\_label.text = ""

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

app = CalculatorApp()

app.run()

Listing 21 Instalasi Source Code Aplikasi Perhitungan Balok KivyMD

Penjelasan dapat anda simak sebagai berikut:

##### **Module**

kode dibawah ini adalah bagian module untuk mengimport modul dari KivyMD menggunakan pemograman python, module ini digunakan untuk memanggil fungsi-fungsi yang disediakan oleh kivymd, Dengan menggunakan kelas-kelas ini, kita dapat membuat antarmuka pengguna dengan tombol, input teks, label, dan tata letak grid. Contohnya bisa seperti membuat layar dengan beberapa widget seperti tombol, input teks, dan label, dan mengatur tata letaknya menggunakan MDGridLayout. Saat tombol ditekan, kita dapat menangani logika atau peristiwa tertentu dalam metode yang sesuai dengan kelas aplikasi yang dibuat dengan mewarisi MDApp.

from kivymd.app import MDApp

from kivymd.uix.screen import MDScreen

from kivymd.uix.gridlayout import MDGridLayout

from kivymd.uix.button import MDRaisedButton

from kivymd.uix.textfield import MDTextField

from kivymd.uix.label import MDLabel

Listing 22 Source Code Import Modul KivyMD

##### **Class** **CalculatorApp**

Class CalculatorApp ini bertanggung jawab sebagai pengelola antarmuka pengguna dan kontrol utama aplikasi. Fungsinya meliputi inisialisasi atribut utama, penambahan dan penghapusan objek bentuk, serta tampilan hasil perhitungan di antarmuka aplikasi.

class CalculatorApp(MDApp):

def build(self):

self.title = "Aplikasi Menghitung Luas Dan Volume Balok"

self.theme\_cls.primary\_palette = "Blue"

layout = MDScreen()

# Title Labels

title\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=300, height=100, pos\_hint={'center\_x': 0.5, 'center\_y': 0.9})

title\_label = MDLabel(

text="Hitung Luas Dan Volume Balok",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

title\_box.add\_widget(title\_label)

layout.add\_widget(title\_box)

# Input Fields

input\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=300, height=250, pos\_hint={'center\_x': 0.5, 'center\_y': 0.6})

self.panjang\_text = MDTextField(

hint\_text="Panjang (cm)",

helper\_text="Enter the length",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.lebar\_text = MDTextField(

hint\_text="Lebar (cm)",

helper\_text="Enter the width",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.tinggi\_text = MDTextField(

hint\_text="Tinggi (cm)",

helper\_text="Enter the height",

helper\_text\_mode="on\_focus",

input\_type="number"

)

input\_box.add\_widget(self.panjang\_text)

input\_box.add\_widget(self.lebar\_text)

input\_box.add\_widget(self.tinggi\_text)

layout.add\_widget(input\_box)

# Buttons

button\_box = MDGridLayout(cols=2, spacing=10, size\_hint=(None, None), width=300, height=50, pos\_hint={'center\_x': 0.5, 'center\_y': 0.44})

calculate\_button = MDRaisedButton(text="Calculate")

calculate\_button.bind(on\_release=self.calculate)

reset\_button = MDRaisedButton(text="Reset")

reset\_button.bind(on\_release=self.reset)

button\_box.add\_widget(calculate\_button)

button\_box.add\_widget(reset\_button)

layout.add\_widget(button\_box)

self.panjang\_text = MDTextField(

hint\_text="Panjang (cm)",

helper\_text="Enter the length",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.lebar\_text = MDTextField(

hint\_text="Lebar (cm)",

helper\_text="Enter the width",

helper\_text\_mode="on\_focus",

input\_type="number"

)

self.tinggi\_text = MDTextField(

hint\_text="Tinggi (cm)",

helper\_text="Enter the height",

helper\_text\_mode="on\_focus",

input\_type="number"

)

input\_box.add\_widget(self.panjang\_text)

input\_box.add\_widget(self.lebar\_text)

input\_box.add\_widget(self.tinggi\_text)

layout.add\_widget(input\_box)

# Buttons

button\_box = MDGridLayout(cols=2, spacing=10, size\_hint=(None, None), width=300, height=50, pos\_hint={'center\_x': 0.5, 'center\_y': 0.44})

calculate\_button = MDRaisedButton(text="Calculate")

calculate\_button.bind(on\_release=self.calculate)

reset\_button = MDRaisedButton(text="Reset")

reset\_button.bind(on\_release=self.reset)

button\_box.add\_widget(calculate\_button)

button\_box.add\_widget(reset\_button)

layout.add\_widget(button\_box)

# Result Labels

result\_box = MDGridLayout(cols=1, size\_hint=(None, None), width=500, height=200, pos\_hint={'center\_x': 0.5, 'center\_y': 0.38})

self.luas\_permukaan\_label = MDLabel(

text="Luas Permukaan:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

self.volume\_label = MDLabel(

text="Volume:",

halign="center",

theme\_text\_color="Primary",

font\_style="H6"

)

# Output Labels

self.panjang\_label = MDLabel()

self.lebar\_label = MDLabel()

self.tinggi\_label = MDLabel()

self.output\_label = MDLabel()

result\_box.add\_widget(self.panjang\_label)

result\_box.add\_widget(self.lebar\_label)

result\_box.add\_widget(self.tinggi\_label)

result\_box.add\_widget(self.luas\_permukaan\_label)

result\_box.add\_widget(self.volume\_label)

result\_box.add\_widget(self.output\_label)

layout.add\_widget(result\_box)

return layout

def calculate(self, instance):

try:

panjang = float(self.panjang\_text.text)

lebar = float(self.lebar\_text.text)

tinggi = float(self.tinggi\_text.text)

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

volume = panjang \* lebar \* tinggi

self.luas\_permukaan\_label.text = f"Luas Permukaan: {luas\_permukaan:.2f} cm^2"

self.volume\_label.text = f"Volume: {volume:.2f} cm^3"

self.output\_label.text = ""

except ValueError:

self.luas\_permukaan\_label.text = "Error: Invalid input"

self.volume\_label.text = "Volume:"

def reset(self, instance):

self.panjang\_text.text = ""

self.lebar\_text.text = ""

self.tinggi\_text.text = ""

self.luas\_permukaan\_label.text = "Luas Permukaan:"

self.volume\_label.text = "Volume:"

self.output\_label.text = ""

Listing 23 Source Code Class CalculatorApp

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

blok if \_\_name\_\_ == "\_\_main\_\_": digunakan untuk mengeksekusi kode tertentu hanya jika skrip dijalankan secara langsung, bukan diimpor sebagai modul ke dalam skrip lain. Dalam konteks kode yang diberikan, ini membuat aplikasi GUI Tkinter dijalankan saat skrip dieksekusi secara langsung.

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

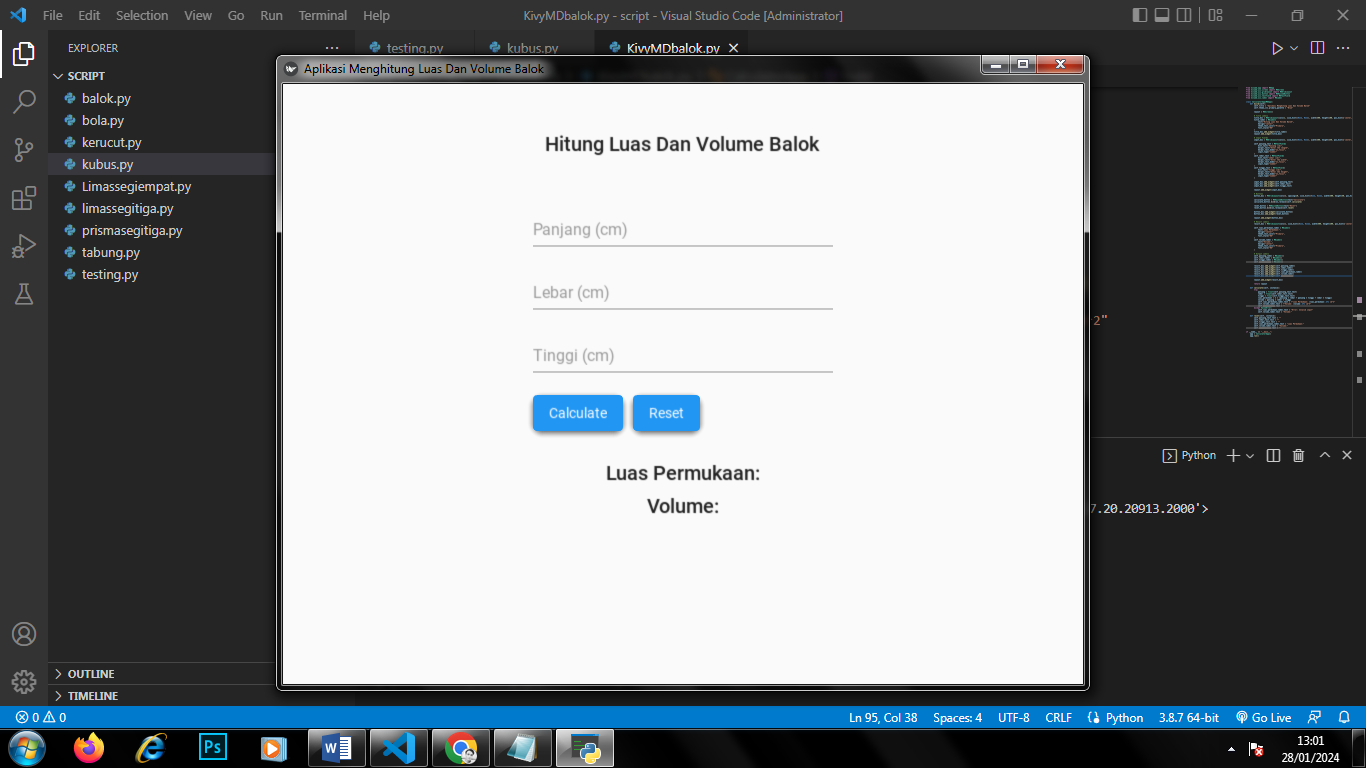
app = CalculatorApp()

app.run()

Listing 24 Source Code if \_\_name\_\_ == "\_\_main\_\_":

##### **Hasil**

Berikut adalah gambar dari aplikasi menghitung luas permukan dan volume yang berhasil dijalankan menggunakan source code sebelumnya.



Gambar 3 Aplikasi Perhitungan Balok Menggunakan KivyMD

### **Latihan**

Setelah kita belajar bagaimana struktur dan fungsi didalam pertemuan ini mari kita coba untuk mengerjakan ini dengan membentuk kelompok berisi paling byk 3 org Pembagian Tugas:

* **Kelas E dan D3:**

**Mempelajari tentang Flet-Python**

Target : Membuat Aplikasi Perhitungan python menggunakan Flet-Python

* **Kelas C dan D:**

**Mempelajari tentang PyQT, Qt Designer**

Target: Memuat Aplikasi Perhitungan menggunakan pyqt

* **Kelas A :**

**Mempelajari tentang Flask-Python**

Target : Membuat aplikasi Perhitungan menggunakan Flask

* **Kelas B:**

**Mempelajari tentang KivyMD-Python**

Target : Membuat aplikasi Perhitungan menggunakan KivyMD

# **PERTEMUAN 4**

## **Module**

Modul dalam Python adalah file yang berisi definisi dan pernyataan Python. Untuk memanggil Python lain dari satu skrip, gunakan konsep modul dengan pernyataan import. Contoh, modul.py dapat diimpor dan fungsi-fungsinya dapat digunakan di modul2.py. Modul mempermudah pengelolaan dan struktur proyek Python.

### **Import Modul Tertentu & Import Massive.**

Dalam Python, untuk mengimpor fungsi atau objek tertentu dari suatu modul, kita dapat menggunakan :

**from module\_name import function\_name**

Listing 25 Source Code Import Modul Tertentu Contoh 1

Atau kita juga dapat mencoba dengan contoh kode dibawah ini :

**from module\_name import \*.**

Listing 26 Source Code Import Modul Tertentu contoh 2

Sebaliknya, dengan import module\_name, kita mengimpor keseluruhan modul,

**import module\_name**

Listing 27 Source Code Import Modul Massive

Dan untuk menggunakan fungsi atau objek di dalamnya, kita harus menyertakan nama modul. Pilihan antara import modul tertentu atau import keseluruhan modul tergantung pada kebutuhan struktur kode dan untuk mencegah konflik nama.

### **Pembahasan**

Berikut adalah contoh bagaimana kita dapat mengimport modul tertentu maupun modul massive :

#### **Import Fungsi Tertentu**

Dibawah ini adalah source code untuk import fungsi tertentu menggunakan rumus balok yang akan kita beri nama sebagai *main.py* :

import tkinter as tk

from rumus\_balok import hitung\_luas\_permukaan, hitung\_volume

def hitung\_dan\_tampilkan():

panjang = float(panjang\_entry.get())

lebar = float(lebar\_entry.get())

tinggi = float(tinggi\_entry.get())

luas\_permukaan = hitung\_luas\_permukaan(panjang, lebar, tinggi)

volume = hitung\_volume(panjang, lebar, tinggi)

hasil\_label.config(text=f"Luas Permukaan: {luas\_permukaan}\nVolume: {volume}")

app = tk.Tk()

app.title("Kalkulator Balok")

panjang\_label = tk.Label(app, text="Panjang:")

panjang\_label.pack()

panjang\_entry = tk.Entry(app)

panjang\_entry.pack()

lebar\_label = tk.Label(app, text="Lebar:")

lebar\_label.pack()

lebar\_entry = tk.Entry(app)

lebar\_entry.pack()

tinggi\_label = tk.Label(app, text="Tinggi:")

tinggi\_label.pack()

tinggi\_entry = tk.Entry(app)

tinggi\_entry.pack()

hitung\_button = tk.Button(app, text="Hitung", command=hitung\_dan\_tampilkan)

hitung\_button.pack()

hasil\_label = tk.Label(app, text="")

hasil\_label.pack()

app.mainloop()

Listing 28 Source Code Import Fungsi Tertentu Main.py

Kemudian kita buat file baru yang kita namakan sebagai rumus.py, dibawah ini adalah source code untuk import fungsi tertentu yang akan kita beri nama sebagai *rumus\_balok.py* :

# rumus\_balok.py

def hitung\_luas\_permukaan(panjang, lebar, tinggi):

luas\_permukaan = 2 \* (panjang \* lebar + panjang \* tinggi + lebar \* tinggi)

return luas\_permukaan

def hitung\_volume(panjang, lebar, tinggi):

volume = panjang \* lebar \* tinggi

return volume

Listing 29 Source Code Import Fungsi Tertentu rumus\_balok.py

Penjelasan dapat anda simak sebagai berikut:

##### **Main.py**

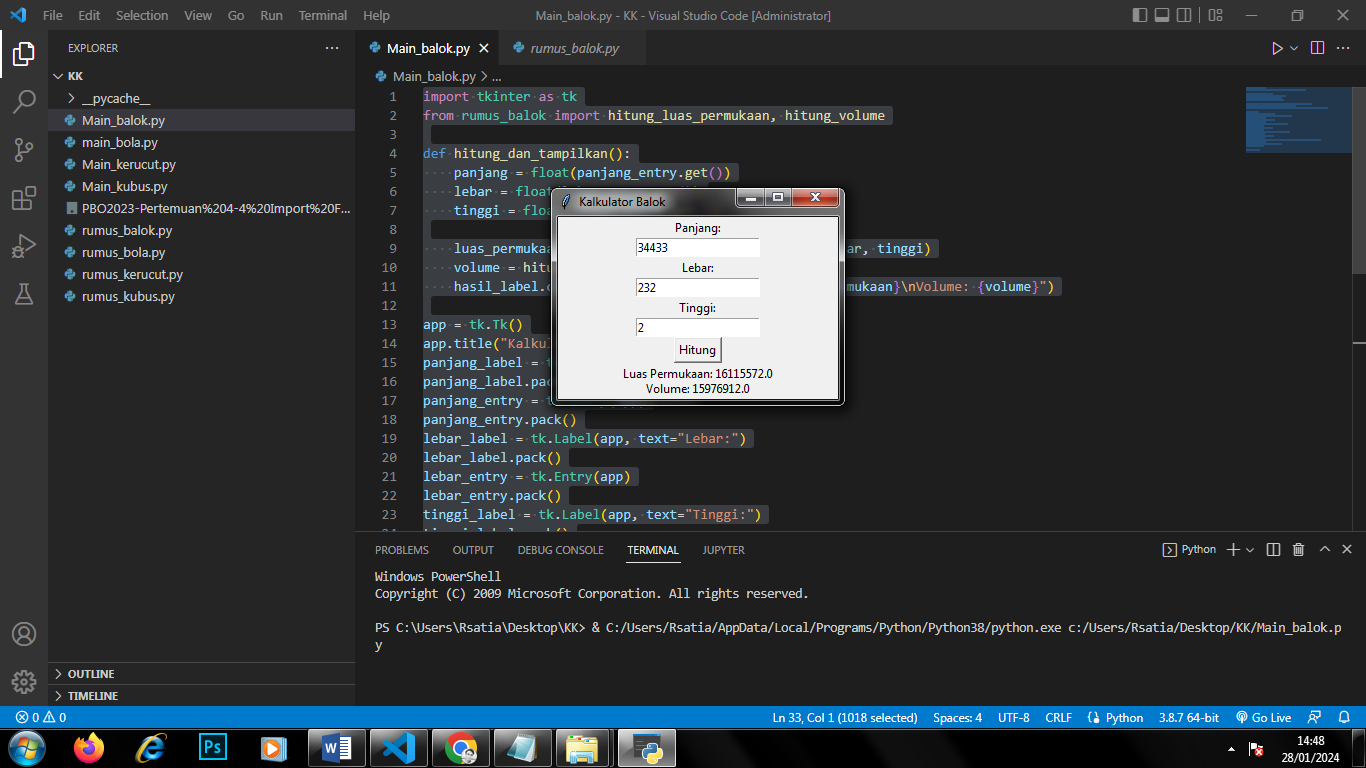
Main.py mengimpor modul tkinter sebagai tk, dan juga mengimpor fungsi hitung\_luas\_permukaan dan hitung\_volume dari modul rumus\_balok. Dan menggunakan fungsi-fungsi tersebut untuk menghitung luas permukaan dan volume berdasarkan input pengguna dari antarmuka grafis.

##### **Rumus\_Balok.py**

Mendefinisikan fungsi hitung\_luas\_permukaan dan hitung\_volume dan Fungsi-fungsi ini menyediakan rumus untuk menghitung luas permukaan dan volume balok berdasarkan panjang, lebar, dan tinggi.

##### **Hasil**

Berikut adalah gambar kode dari import fungsi tertentu yang berhasil dijalankan menggunakan source code sebelumnya.



Gambar 4 Import Fungsi Tertentu Balok Main.py

#### **Import Fungsi Massive**

Dibawah ini adalah source code untuk import fungsi tertentu yang akan kita beri nama sebagai *main.tabung.py* :

import tkinter as tk

from rumus\_tabung import \*

def hitung\_dan\_tampilkan():

jari\_jari = float(jari\_jari\_entry.get())

tinggi = float(tinggi\_entry.get())

luas\_permukaan = hitung\_luas\_permukaan\_tabung(jari\_jari, tinggi)

volume = hitung\_volume\_tabung(jari\_jari, tinggi)

hasil\_label.config(text=f"Luas Permukaan: {luas\_permukaan}\nVolume: {volume}")

app = tk.Tk()

app.title("Kalkulator Tabung")

jari\_jari\_label = tk.Label(app, text="Jari-jari:")

jari\_jari\_label.pack()

jari\_jari\_entry = tk.Entry(app)

jari\_jari\_entry.pack()

tinggi\_label = tk.Label(app, text="Tinggi:")

tinggi\_label.pack()

tinggi\_entry = tk.Entry(app)

tinggi\_entry.pack()

hitung\_button = tk.Button(app, text="Hitung", command=hitung\_dan\_tampilkan)

hitung\_button.pack()

hasil\_label = tk.Label(app, text="")

hasil\_label.pack()

app.mainloop()

Listing 30 Source Code Import Fungsi Massive Main\_Tabung.py

Kemudian kita buat file baru yang kita namakan sebagai rumus\_tabung.py, dibawah ini adalah source code untuk import fungsi massive yang akan kita beri nama sebagai *rumus\_tabung.py* :

import math

def hitung\_luas\_permukaan\_tabung(jari\_jari, tinggi):

luas\_permukaan = 2 \* math.pi \* jari\_jari \* (jari\_jari + tinggi)

return luas\_permukaan

def hitung\_volume\_tabung(jari\_jari, tinggi):

volume = math.pi \* jari\_jari\*\*2 \* tinggi

return volume

Listing 31 Source Code Import Fungsi Massive rumus\_tabung.py

Penjelasan dapat anda simak sebagai berikut:

##### **Main\_Tabung.py**

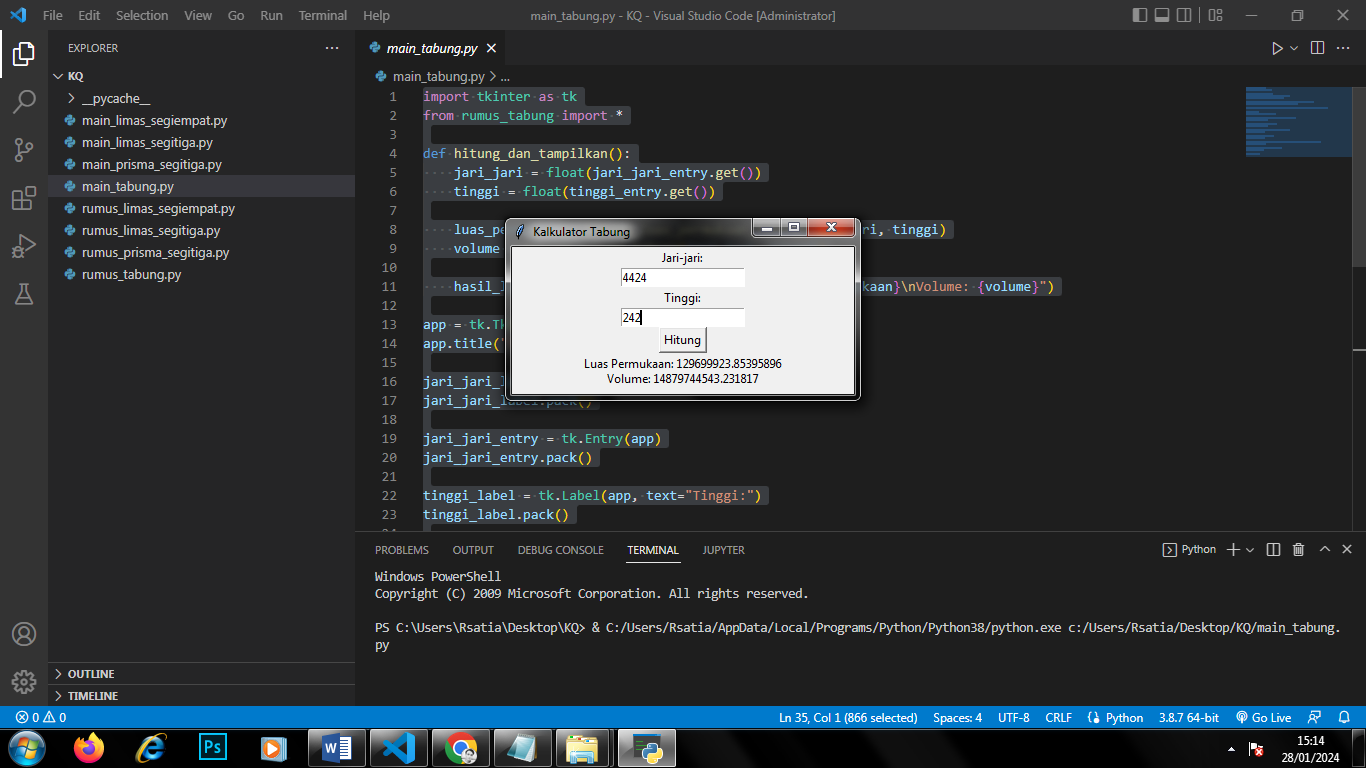
Main\_tabung.py mengimpor modul tkinter sebagai tk dan mengimpor semua fungsi dari modul rumus\_tabung menggunakan from rumus\_tabung import \* lalu menggunakan fungsi-fungsi tersebut untuk menghitung luas permukaan dan volume tabung berdasarkan input pengguna dari antarmuka grafis.

##### **Rumus\_Tabung.py**

Didalam rumus yang telah didefinisikan sebelumnya didalam rumus\_tabung.py Mendefinisikan fungsi diantaranya adalah sebagai berikut, hitung\_luas\_permukaan\_tabung dan juga fungsi hitung\_volume\_tabung. Fungsi-fungsi ini menyediakan rumus untuk menghitung luas permukaan dan volume tabung berdasarkan jari-jari dan tinggi.

##### **Hasil**

Berikut adalah hasil dari menghitung luas menggunakan import massive yang berhasil dijalankan menggunakan source code sebelumnya.



Gambar 5 Import Fungsi Tertentu Tabung Main\_Tabung.py

### **Latihan**

Setelah kita belajar bagaimana struktur dan fungsi didalam pertemuan ini mari kita coba untuk menggunakan listing code yang lain, buatlah beberapa kode import Fungsi Tertentu dan Import Massive seperti contoh yang telah diberikan sebelumnya! Minimal 3:

# **PERTEMUAN 5**

## **Quiz**

### **Quizziz.**

Untuk pertemuan ini akan diadakan sebuah quiz anda dapat mendaftar melalui link yang akan disediakan silahkan daftar dulu untuk mengakses quizziz :

https://quizizz.com/

Setelah registrasi silahkan join kelas :

**Kelas A**

https://quizizz.com/join?class=G755541

**Kelas B**

https://quizizz.com/join?class=X799908

**Kelas C**

https://quizizz.com/join?class=A867700

**Kelas D**

https://quizizz.com/join?class=H686010

**Kelas E**

https://quizizz.com/join?class=K883073

**Kelas D3**

https://quizizz.com/join?class=F268076

**Kelas K**

https://quizizz.com/join?class=T922296

**Kelas L**

https://quizizz.com/join?class=X041348

### **Latihan**

Silahkan daftar dan setelah itu kerjakan quiz yang ada pada link yang sudah diberikan waktu pengerjaan 60 menit pastikan anda membaca dengan teliti soal yang diberikan, selamat mengerjakan!.

# **PERTEMUAN 6**

## **Module**

## **Materi Aplikasi Bantu (Tools)**

Latihan ini bertujuan untuk membuat kode pemutar music, pemutar video dan image to text sederhana menggunakan python.

Untuk membuat didalam pertemuan ini kita harus menginstal modul beberapa modul terlebih dahulu dengan membuka cmd atau powershell :

* *Untuk membuka Command Prompt, tekan tombol Windows + R, ketik cmd, dan tekan Enter.*
* *Untuk membuka PowerShell, tekan tombol Windows + R, ketik powershell, dan tekan Enter.*

Kemudian anda dapat menggunakan kode ini untuk menginstal modul, kodenya dapat dilihat sebagai berikut :

pip install tk

pip install pygame

pip install Pillow

pip install eyed3  
pip install opencv-python-headless pillow pydub

pip install tk pyttsx3

Listing 32 Instalasi Modul

## **Pembahasan**

#### **MP3:**

Dalam pertemuan kali ini kita akan membuat aplikasi python untuk membuat pemutar music menggunakan python untuk dapat melakukanya kita sudah menyiapkan kode lengkap yang bisa anda gunakan sebagai referensi berikut adalah kodenya.

import tkinter as tk

from tkinter import ttk, filedialog, messagebox

import pygame

import threading

from PIL import Image, ImageTk, ImageEnhance

import time

import io

import eyed3

class MP3Player:

def \_\_init\_\_(self, master):

self.master = master

self.master.title("Media Player")

self.master.geometry("300x400") # Initial window size

self.master.minsize(300, 300) # Minimum window size

self.master.configure(bg="#f0f0f0")

self.current\_file = None

self.playing = False

# Set icon

icon\_image = Image.open("folder.jpg") # Change to the path of your icon file

icon\_image = icon\_image.resize((30, 30), Image.ANTIALIAS)

self.icon = ImageTk.PhotoImage(icon\_image)

self.master.iconphoto(False, self.icon)

# Style for buttons

style = ttk.Style()

style.configure("TButton", padding=5, relief="flat", background="#4CAF50", foreground="white")

style.map("TButton", background=[("active", "#45a049")])

# Frame for displaying cover image or video

self.cover\_frame = tk.Frame(self.master, bg="#f0f0f0")

self.cover\_frame.pack(expand=True, fill="both", padx=10, pady=10)

# Label for marquee (scrolling text)

self.marquee\_label = tk.Label(self.cover\_frame, text="", bg="#f0f0f0", font=("Helvetica", 12))

self.marquee\_label.pack(expand=True, fill="x", padx=10, pady=5)

self.marquee\_label.place\_forget()

# Label for displaying cover image or video

self.cover\_label = tk.Label(self.cover\_frame, bg="#f0f0f0", cursor="hand2")

self.cover\_label.pack(expand=True, fill="both")

self.cover\_label.bind("<Enter>", self.hover\_enter)

self.cover\_label.bind("<Leave>", self.hover\_leave)

self.title\_var = tk.StringVar()

self.title\_label = tk.Label(self.cover\_frame, textvariable=self.title\_var, bg="#f0f0f0", font=("Helvetica", 12), wraplength=300)

self.title\_label.pack(expand=True, fill="x", padx=10, pady=5)

self.title\_label.place\_forget()

self.position\_scale = ttk.Progressbar(self.master, orient=tk.HORIZONTAL, length=300, mode='determinate')

self.position\_scale.pack(expand=True, fill="x", padx=10, pady=5)

self.duration\_label = tk.Label(self.master, text="Duration: 0:00", bg="#f0f0f0")

self.duration\_label.pack(expand=True, fill="x", padx=10, pady=5)

# Frame for grouping buttons below the progress bar

button\_frame = tk.Frame(self.master, bg="#f0f0f0")

button\_frame.pack(expand=True, fill="x", padx=10, pady=10)

# Tombol Play/Pause

self.play\_pause\_button = ttk.Button(button\_frame, text="▶ Play", command=self.toggle\_play\_pause, style="TButton")

self.play\_pause\_button.grid(row=0, column=0, padx=5, pady=5, sticky="ew")

self.stop\_button = ttk.Button(button\_frame, text="⏹ Stop", command=self.stop, style="TButton")

self.stop\_button.grid(row=0, column=1, padx=5, pady=5, sticky="ew")

self.choose\_file\_button = ttk.Button(button\_frame, text="Choose File", command=self.choose\_file, style="TButton")

self.choose\_file\_button.grid(row=0, column=2, padx=5, pady=5, sticky="ew")

self.update\_position\_thread = threading.Thread(target=self.update\_position)

self.update\_position\_thread.daemon = True

self.update\_position\_thread.start()

# Bind the window resize event to adjust the layout

self.master.bind("<Configure>", self.on\_window\_configure)

def on\_window\_configure(self, event):

# Update the position of the marquee and title labels when the window is resized

if self.playing:

self.hover\_enter(None)

def choose\_file(self):

file\_path = filedialog.askopenfilename(filetypes=[("Media files", "\*.mp3;\*.mp4")])

if file\_path:

self.current\_file = file\_path

pygame.mixer.init()

if file\_path.lower().endswith('.mp3'):

# MP3 File

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

duration = pygame.mixer.Sound(self.current\_file).get\_length()

self.position\_scale.config(maximum=duration)

self.duration\_label.config(text=f"Duration: {self.format\_time(duration)}")

# Display cover from MP3 file

cover\_info = self.get\_mp3\_cover(self.current\_file)

self.display\_cover(cover\_info)

elif file\_path.lower().endswith('.mp4'):

# MP4 File

messagebox.showinfo("Info", "MP4 video will be displayed above the progress bar.")

def toggle\_play\_pause(self):

if self.playing:

self.pause()

else:

self.play()

def play(self):

if self.current\_file:

pygame.mixer.music.play()

self.playing = True

self.play\_pause\_button.configure(text="⏸ Pause")

def pause(self):

pygame.mixer.music.pause()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

def stop(self):

pygame.mixer.music.stop()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

self.position\_scale.stop()

def update\_position(self):

while True:

if self.playing:

position = pygame.mixer.music.get\_pos() / 1000.0

self.position\_scale['value'] = position

time.sleep(0.1) # Update position every 0.1 seconds

def format\_time(self, seconds):

minutes, seconds = divmod(int(seconds), 60)

return f"{minutes}:{seconds:02}"

def get\_mp3\_cover(self, mp3\_file):

try:

audio = eyed3.load(mp3\_file)

if audio.tag and audio.tag.images:

title = audio.tag.title if audio.tag.title else "Unknown Title"

artist = audio.tag.artist if audio.tag.artist else "Unknown Artist"

return title, artist, Image.open(io.BytesIO(audio.tag.images[0].image\_data)).convert("RGBA")

else:

return None

except Exception as e:

print(f"Error retrieving MP3 cover: {e}")

return None

def display\_cover(self, cover\_info):

if cover\_info:

title, artist, cover\_image = cover\_info

# Set a maximum size for the displayed cover

max\_width, max\_height = 300, 300

cover\_image.thumbnail((max\_width, max\_height))

self.cover\_image\_original = ImageTk.PhotoImage(cover\_image)

self.cover\_image\_hover = ImageTk.PhotoImage(self.darken\_image(cover\_image, 0.5))

self.cover\_label.configure(image=self.cover\_image\_original, width=self.cover\_image\_original.width(), height=self.cover\_image\_original.height())

self.cover\_label.image = self.cover\_image\_original

self.title\_var.set(f"{title} - {artist}") # Menampilkan judul dan artis

else:

# If no cover image, clear the label

self.cover\_label.configure(image=None, width=0, height=0)

self.title\_var.set("")

def darken\_image(self, img, factor):

enhancer = ImageEnhance.Brightness(img)

darkened\_img = enhancer.enhance(factor)

return darkened\_img

def hover\_enter(self, event):

# Darken the cover image, show marquee text, and set the title when the cursor enters the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_hover)

self.marquee\_label.configure(text="Your Scrolling Text Here", width=self.cover\_image\_original.width())

self.marquee\_label.place(relx=0.5, rely=0.5, anchor="center", y=-50) # Adjust the y-coordinate as needed

self.title\_label.place(relx=0.5, rely=0.5, anchor="center", y=50) # Adjust the y-coordinate as needed

def hover\_leave(self, event):

# Restore the original cover image, hide the marquee text, and hide the title when the cursor leaves the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_original)

self.marquee\_label.place\_forget()

self.title\_label.place\_forget()

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

root = tk.Tk()

mp3\_player = MP3Player(root)

root.mainloop()

self.cover\_label.bind("<Enter>", self.hover\_enter)

self.cover\_label.bind("<Leave>", self.hover\_leave)

self.title\_var = tk.StringVar()

self.title\_label = tk.Label(self.cover\_frame, textvariable=self.title\_var, bg="#f0f0f0", font=("Helvetica", 12), wraplength=300)

self.title\_label.pack(expand=True, fill="x", padx=10, pady=5)

self.title\_label.place\_forget()

self.position\_scale = ttk.Progressbar(self.master, orient=tk.HORIZONTAL, length=300, mode='determinate')

self.position\_scale.pack(expand=True, fill="x", padx=10, pady=5)

self.duration\_label = tk.Label(self.master, text="Duration: 0:00", bg="#f0f0f0")

self.duration\_label.pack(expand=True, fill="x", padx=10, pady=5)

# Frame for grouping buttons below the progress bar

button\_frame = tk.Frame(self.master, bg="#f0f0f0")

button\_frame.pack(expand=True, fill="x", padx=10, pady=10)

# Tombol Play/Pause

self.play\_pause\_button = ttk.Button(button\_frame, text="▶ Play", command=self.toggle\_play\_pause, style="TButton")

self.play\_pause\_button.grid(row=0, column=0, padx=5, pady=5, sticky="ew")

self.stop\_button = ttk.Button(button\_frame, text="⏹ Stop", command=self.stop, style="TButton")

self.stop\_button.grid(row=0, column=1, padx=5, pady=5, sticky="ew")

self.choose\_file\_button = ttk.Button(button\_frame, text="Choose File", command=self.choose\_file, style="TButton")

self.choose\_file\_button.grid(row=0, column=2, padx=5, pady=5, sticky="ew")

self.update\_position\_thread = threading.Thread(target=self.update\_position)

self.update\_position\_thread.daemon = True

self.update\_position\_thread.start()

# Bind the window resize event to adjust the layout

self.master.bind("<Configure>", self.on\_window\_configure)

def on\_window\_configure(self, event):

# Update the position of the marquee and title labels when the window is resized

if self.playing:

self.hover\_enter(None)

def choose\_file(self):

file\_path = filedialog.askopenfilename(filetypes=[("Media files", "\*.mp3;\*.mp4")])

if file\_path:

self.current\_file = file\_path

pygame.mixer.init()

if file\_path.lower().endswith('.mp3'):

# MP3 File

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

duration = pygame.mixer.Sound(self.current\_file).get\_length()

self.position\_scale.config(maximum=duration)

self.duration\_label.config(text=f"Duration: {self.format\_time(duration)}")

# Display cover from MP3 file

cover\_info = self.get\_mp3\_cover(self.current\_file)

self.display\_cover(cover\_info)

elif file\_path.lower().endswith('.mp4'):

# MP4 File

messagebox.showinfo("Info", "MP4 video will be displayed above the progress bar.")

def toggle\_play\_pause(self):

if self.playing:

self.pause()

else:

self.play()

def play(self):

if self.current\_file:

pygame.mixer.music.play()

self.playing = True

self.play\_pause\_button.configure(text="⏸ Pause")

def pause(self):

pygame.mixer.music.pause()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

def stop(self):

pygame.mixer.music.stop()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

self.position\_scale.stop()

def update\_position(self):

while True:

if self.playing:

position = pygame.mixer.music.get\_pos() / 1000.0

self.position\_scale['value'] = position

time.sleep(0.1) # Update position every 0.1 seconds

def format\_time(self, seconds):

minutes, seconds = divmod(int(seconds), 60)

return f"{minutes}:{seconds:02}"

def get\_mp3\_cover(self, mp3\_file):

try:

audio = eyed3.load(mp3\_file)

if audio.tag and audio.tag.images:

title = audio.tag.title if audio.tag.title else "Unknown Title"

artist = audio.tag.artist if audio.tag.artist else "Unknown Artist"

return title, artist, Image.open(io.BytesIO(audio.tag.images[0].image\_data)).convert("RGBA")

else:

return None

except Exception as e:

print(f"Error retrieving MP3 cover: {e}")

return None

def display\_cover(self, cover\_info):

if cover\_info:

title, artist, cover\_image = cover\_info

# Set a maximum size for the displayed cover

max\_width, max\_height = 300, 300

cover\_image.thumbnail((max\_width, max\_height))

self.cover\_image\_original = ImageTk.PhotoImage(cover\_image)

self.cover\_image\_hover = ImageTk.PhotoImage(self.darken\_image(cover\_image, 0.5))

self.cover\_label.configure(image=self.cover\_image\_original, width=self.cover\_image\_original.width(), height=self.cover\_image\_original.height())

self.cover\_label.image = self.cover\_image\_original

self.title\_var.set(f"{title} - {artist}") # Menampilkan judul dan artis

else:

# If no cover image, clear the label

self.cover\_label.configure(image=None, width=0, height=0)

self.title\_var.set("")

def darken\_image(self, img, factor):

enhancer = ImageEnhance.Brightness(img)

darkened\_img = enhancer.enhance(factor)

return darkened\_img

def hover\_enter(self, event):

# Darken the cover image, show marquee text, and set the title when the cursor enters the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_hover)

self.marquee\_label.configure(text="Your Scrolling Text Here", width=self.cover\_image\_original.width())

self.marquee\_label.place(relx=0.5, rely=0.5, anchor="center", y=-50) # Adjust the y-coordinate as needed

self.title\_label.place(relx=0.5, rely=0.5, anchor="center", y=50) # Adjust the y-coordinate as needed

def hover\_leave(self, event):

# Restore the original cover image, hide the marquee text, and hide the title when the cursor leaves the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_original)

self.marquee\_label.place\_forget()

self.title\_label.place\_forget()

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

root = tk.Tk()

mp3\_player = MP3Player(root)

root.mainloop()

def on\_window\_configure(self, event):

# Update the position of the marquee and title labels when the window is resized

if self.playing:

self.hover\_enter(None)

def choose\_file(self):

file\_path = filedialog.askopenfilename(filetypes=[("Media files", "\*.mp3;\*.mp4")])

if file\_path:

self.current\_file = file\_path

pygame.mixer.init()

if file\_path.lower().endswith('.mp3'):

# MP3 File

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

duration = pygame.mixer.Sound(self.current\_file).get\_length()

self.position\_scale.config(maximum=duration)

self.duration\_label.config(text=f"Duration: {self.format\_time(duration)}")

# Display cover from MP3 file

cover\_info = self.get\_mp3\_cover(self.current\_file)

self.display\_cover(cover\_info)

elif file\_path.lower().endswith('.mp4'):

# MP4 File

messagebox.showinfo("Info", "MP4 video will be displayed above the progress bar.")

def toggle\_play\_pause(self):

if self.playing:

self.pause()

else:

self.play()

def play(self):

if self.current\_file:

pygame.mixer.music.play()

self.playing = True

self.play\_pause\_button.configure(text="⏸ Pause")

def pause(self):

pygame.mixer.music.pause()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

def stop(self):

pygame.mixer.music.stop()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

self.position\_scale.stop()

def update\_position(self):

while True:

if self.playing:

position = pygame.mixer.music.get\_pos() / 1000.0

self.position\_scale['value'] = position

time.sleep(0.1) # Update position every 0.1 seconds

def format\_time(self, seconds):

minutes, seconds = divmod(int(seconds), 60)

return f"{minutes}:{seconds:02}"

def get\_mp3\_cover(self, mp3\_file):

try:

audio = eyed3.load(mp3\_file)

if audio.tag and audio.tag.images:

title = audio.tag.title if audio.tag.title else "Unknown Title"

artist = audio.tag.artist if audio.tag.artist else "Unknown Artist"

return title, artist, Image.open(io.BytesIO(audio.tag.images[0].image\_data)).convert("RGBA")

else:

return None

except Exception as e:

print(f"Error retrieving MP3 cover: {e}")

return None

def display\_cover(self, cover\_info):

if cover\_info:

title, artist, cover\_image = cover\_info

# Set a maximum size for the displayed cover

max\_width, max\_height = 300, 300

cover\_image.thumbnail((max\_width, max\_height))

self.cover\_image\_original = ImageTk.PhotoImage(cover\_image)

self.cover\_image\_hover = ImageTk.PhotoImage(self.darken\_image(cover\_image, 0.5))

self.cover\_label.configure(image=self.cover\_image\_original, width=self.cover\_image\_original.width(), height=self.cover\_image\_original.height())

self.cover\_label.image = self.cover\_image\_original

self.title\_var.set(f"{title} - {artist}") # Menampilkan judul dan artis

else:

# If no cover image, clear the label

self.cover\_label.configure(image=None, width=0, height=0)

self.title\_var.set("")

def darken\_image(self, img, factor):

enhancer = ImageEnhance.Brightness(img)

darkened\_img = enhancer.enhance(factor)

return darkened\_img

def hover\_enter(self, event):

# Darken the cover image, show marquee text, and set the title when the cursor enters the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_hover)

self.marquee\_label.configure(text="Your Scrolling Text Here", width=self.cover\_image\_original.width())

self.marquee\_label.place(relx=0.5, rely=0.5, anchor="center", y=-50) # Adjust the y-coordinate as needed

self.title\_label.place(relx=0.5, rely=0.5, anchor="center", y=50) # Adjust the y-coordinate as needed

def hover\_leave(self, event):

# Restore the original cover image, hide the marquee text, and hide the title when the cursor leaves the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_original)

self.marquee\_label.place\_forget()

self.title\_label.place\_forget()

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

root = tk.Tk()

mp3\_player = MP3Player(root)

root.mainloop()

def stop(self):

pygame.mixer.music.stop()

self.playing = False

self.play\_pause\_button.configure(text="▶ Play")

self.position\_scale.stop()

def update\_position(self):

while True:

if self.playing:

position = pygame.mixer.music.get\_pos() / 1000.0

self.position\_scale['value'] = position

time.sleep(0.1) # Update position every 0.1 seconds

def format\_time(self, seconds):

minutes, seconds = divmod(int(seconds), 60)

return f"{minutes}:{seconds:02}"

def get\_mp3\_cover(self, mp3\_file):

try:

audio = eyed3.load(mp3\_file)

if audio.tag and audio.tag.images:

title = audio.tag.title if audio.tag.title else "Unknown Title"

artist = audio.tag.artist if audio.tag.artist else "Unknown Artist"

return title, artist, Image.open(io.BytesIO(audio.tag.images[0].image\_data)).convert("RGBA")

else:

return None

except Exception as e:

print(f"Error retrieving MP3 cover: {e}")

return None

def display\_cover(self, cover\_info):

if cover\_info:

title, artist, cover\_image = cover\_info

# Set a maximum size for the displayed cover

max\_width, max\_height = 300, 300

cover\_image.thumbnail((max\_width, max\_height))

self.cover\_image\_original = ImageTk.PhotoImage(cover\_image)

self.cover\_image\_hover = ImageTk.PhotoImage(self.darken\_image(cover\_image, 0.5))

self.cover\_label.configure(image=self.cover\_image\_original, width=self.cover\_image\_original.width(), height=self.cover\_image\_original.height())

self.cover\_label.image = self.cover\_image\_original

self.title\_var.set(f"{title} - {artist}") # Menampilkan judul dan artis

else:

# If no cover image, clear the label

self.cover\_label.configure(image=None, width=0, height=0)

self.title\_var.set("")

def darken\_image(self, img, factor):

enhancer = ImageEnhance.Brightness(img)

darkened\_img = enhancer.enhance(factor)

return darkened\_img

def hover\_enter(self, event):

# Darken the cover image, show marquee text, and set the title when the cursor enters the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_hover)

self.marquee\_label.configure(text="Your Scrolling Text Here", width=self.cover\_image\_original.width())

self.marquee\_label.place(relx=0.5, rely=0.5, anchor="center", y=-50) # Adjust the y-coordinate as needed

self.title\_label.place(relx=0.5, rely=0.5, anchor="center", y=50) # Adjust the y-coordinate as needed

def hover\_leave(self, event):

# Restore the original cover image, hide the marquee text, and hide the title when the cursor leaves the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_original)

self.marquee\_label.place\_forget()

self.title\_label.place\_forget()

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

root = tk.Tk()

mp3\_player = MP3Player(root)

root.mainloop()

self.cover\_label.image = self.cover\_image\_original

self.title\_var.set(f"{title} - {artist}") # Menampilkan judul dan artis

else:

# If no cover image, clear the label

self.cover\_label.configure(image=None, width=0, height=0)

self.title\_var.set("")

def darken\_image(self, img, factor):

enhancer = ImageEnhance.Brightness(img)

darkened\_img = enhancer.enhance(factor)

return darkened\_img

def hover\_enter(self, event):

# Darken the cover image, show marquee text, and set the title when the cursor enters the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_hover)

self.marquee\_label.configure(text="Your Scrolling Text Here", width=self.cover\_image\_original.width())

self.marquee\_label.place(relx=0.5, rely=0.5, anchor="center", y=-50) # Adjust the y-coordinate as needed

self.title\_label.place(relx=0.5, rely=0.5, anchor="center", y=50) # Adjust the y-coordinate as needed

def hover\_leave(self, event):

# Restore the original cover image, hide the marquee text, and hide the title when the cursor leaves the cover label with a smooth transition

self.cover\_label.configure(image=self.cover\_image\_original)

self.marquee\_label.place\_forget()

self.title\_label.place\_forget()

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

root = tk.Tk()

mp3\_player = MP3Player(root)

root.mainloop()

Listing 33 Source Code MP3

##### **Pembahasan MP3**

###### **Inisialisasi dan Konfigurasi GUI**

Pada bagian ini, kelas MP3Player diinisialisasi dengan konstruktor \_\_init\_\_ membuat jendela tkinter dengan judul "Media Player" dan mengatur ukuran awalnya, menyiapkan beberapa atribut, seperti current\_file untuk menyimpan path file yang diputar dan playing untuk menandakan apakah audio sedang diputar atau tidak, mengatur ikon jendela dengan menggunakan gambar yang diukur ulang dari file "folder.jpg",dan mengatur gaya tombol dengan ttk.Style().

class MP3Player:

def \_\_init\_\_(self, master):

# Inisialisasi atribut dan konfigurasi GUI

Listing 34 Source Code Konfigurasi GUI

###### **Desain Antarmuka Pengguna (GUI)**

Bagian ini mencakup pembuatan elemen-elemen GUI seperti frame untuk menampilkan gambar cover atau video, tombol Play/Pause, Stop, dan Choose File.

# Frame untuk menampilkan gambar cover atau video

self.cover\_frame = tk.Frame(self.master, bg="#f0f0f0")

# ... (lanjutan)

# Tombol Play/Pause, Stop, dan Choose File

self.play\_pause\_button = ttk.Button(button\_frame, text="▶ Play", command=self.toggle\_play\_pause, style="TButton")

self.stop\_button = ttk.Button(button\_frame, text="⏹ Stop", command=self.stop, style="TButton")

self.choose\_file\_button = ttk.Button(button\_frame, text="Choose File", command=self.choose\_file, style="TButton")

# ... (lanjutan)

Listing 35 Source Code Desain Antarmuka Pengguna

###### **Metode untuk Menampilkan Informasi**

def format\_time(self, seconds):

# Metode untuk memformat waktu dalam menit dan detik

# ...

def get\_mp3\_cover(self, mp3\_file):

# Metode untuk mendapatkan informasi cover dari file MP3

# ...

def display\_cover(self, cover\_info):

# Metode untuk menampilkan cover album di GUI

# ...

def darken\_image(self, img, factor):

# Metode untuk membuat gambar menjadi lebih gelap

# ...

def hover\_enter(self, event):

# Metode untuk menangani event ketika cursor masuk ke area cover album

# ...

def hover\_leave(self, event):

# Metode untuk menangani event ketika cursor meninggalkan area cover album

# ...

Listing 36 Source Code Metode Menampilkan Informasi

#### **MP4:**

Dalam pertemuan kali ini kita akan membuat aplikasi python untuk membuat pemutar video menggunakan python untuk dapat melakukanya kita sudah menyiapkan kode lengkap yang bisa anda gunakan sebagai referensi berikut adalah kodenya.

import cv2

import tkinter as tk

from tkinter import filedialog

from PIL import Image, ImageTk

from pydub import AudioSegment

from pydub.playback import play

class VideoPlayer:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Responsive Video Player")

self.root.geometry("800x600")

# Tambahkan binding untuk perubahan ukuran window

self.root.bind("<Configure>", self.resize\_video)

self.create\_widgets()

def create\_widgets(self):

# Frame untuk menempatkan tombol-tombol

control\_frame = tk.Frame(self.root)

control\_frame.pack(side=tk.BOTTOM, pady=10)

# Tombol untuk memilih file video

select\_button = tk.Button(control\_frame, text="Pilih Video", command=self.select\_video)

select\_button.pack(side=tk.LEFT, padx=10)

# Tombol untuk memulai pemutaran video

play\_button = tk.Button(control\_frame, text="Mainkan", command=self.play\_video)

play\_button.pack(side=tk.LEFT, padx=10)

# Tombol untuk menghentikan pemutaran video

stop\_button = tk.Button(control\_frame, text="Stop", command=self.stop\_video)

stop\_button.pack(side=tk.LEFT, padx=10)

# Label untuk menampilkan status

self.status\_label = tk.Label(control\_frame, text="")

self.status\_label.pack(side=tk.LEFT, padx=10)

# Inisialisasi variabel video

self.video\_path = None

self.video\_capture = None

def select\_video(self):

# Memilih file video

file\_path =

self.video\_path = file\_path

self.status\_label.config(text=f"File Video: {file\_path}")

def play\_video(self):

if self.video\_path:

# Menghentikan video yang sedang diputar sebelumnya

self.stop\_video()

# Membuka file video

self.video\_capture = cv2.VideoCapture(self.video\_path)

# Mendapatkan informasi video

self.fps = int(self.video\_capture.get(cv2.CAP\_PROP\_FPS))

# Membuat frame untuk menampilkan video

self.video\_frame = tk.Label(self.root)

self.video\_frame.pack(expand=True, fill="both")

# Fungsi untuk membaca dan menampilkan setiap frame video

def update\_frame():

ret, frame = self.video\_capture.read()

if ret:

frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

frame = self.resize\_frame(frame)

img = self.convert\_frame\_to\_image(frame)

self.video\_frame.configure(image=img)

self.video\_frame.img = img

self.root.after(1000 // self.fps, update\_frame)

else:

self.stop\_video()

# Memulai pemutaran video

update\_frame()

# Memainkan audio

audio\_path = self.video\_path

audio = AudioSegment.from\_file(audio\_path)

play(audio)

def resize\_frame(self, frame):

# Mengubah ukuran frame video sesuai dengan ukuran window

window\_width = self.root.winfo\_width()

window\_height = self.root.winfo\_height()

# Menyesuaikan ukuran frame dengan ukuran window

if window\_width > 1 and window\_height > 1:

frame = cv2.resize(frame, (window\_width, window\_height))

return frame

def convert\_frame\_to\_image(self, frame):

# Konversi frame OpenCV menjadi format yang dapat ditampilkan di Tkinter

img = Image.fromarray(frame)

img = ImageTk.PhotoImage(image=img)

return img

def stop\_video(self):

# Menghentikan pemutaran video

if self.video\_capture is not None:

self.video\_capture.release()

self.video\_frame.pack\_forget()

def resize\_video(self, event):

# Mengubah ukuran frame video sesuai dengan ukuran window

if hasattr(self, 'video\_capture') and self.video\_capture is not None:

ret, frame = self.video\_capture.read()

if ret:

frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

frame = self.resize\_frame(frame)

img = self.convert\_frame\_to\_image(frame)

self.video\_frame.configure(image=img)

self.video\_frame.img = img

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

root = tk.Tk()

app = VideoPlayer(root)

root.mainloop()

self.video\_path = file\_path

self.status\_label.config(text=f"File Video: {file\_path}")

def play\_video(self):

if self.video\_path:

# Menghentikan video yang sedang diputar sebelumnya

self.stop\_video()

# Membuka file video

self.video\_capture = cv2.VideoCapture(self.video\_path)

# Mendapatkan informasi video

self.fps = int(self.video\_capture.get(cv2.CAP\_PROP\_FPS))

# Membuat frame untuk menampilkan video

self.video\_frame = tk.Label(self.root)

self.video\_frame.pack(expand=True, fill="both")

# Fungsi untuk membaca dan menampilkan setiap frame video

def update\_frame():

ret, frame = self.video\_capture.read()

if ret:

frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

frame = self.resize\_frame(frame)

img = self.convert\_frame\_to\_image(frame)

self.video\_frame.configure(image=img)

self.video\_frame.img = img

self.root.after(1000 // self.fps, update\_frame)

else:

self.stop\_video()

# Memulai pemutaran video

update\_frame()

# Memainkan audio

audio\_path = self.video\_path

audio = AudioSegment.from\_file(audio\_path)

play(audio)

def resize\_frame(self, frame):

# Mengubah ukuran frame video sesuai dengan ukuran window

window\_width = self.root.winfo\_width()

window\_height = self.root.winfo\_height()

# Menyesuaikan ukuran frame dengan ukuran window

if window\_width > 1 and window\_height > 1:

frame = cv2.resize(frame, (window\_width, window\_height))

return frame

def convert\_frame\_to\_image(self, frame):

# Konversi frame OpenCV menjadi format yang dapat ditampilkan di Tkinter

img = Image.fromarray(frame)

img = ImageTk.PhotoImage(image=img)

return img

def stop\_video(self):

# Menghentikan pemutaran video

if self.video\_capture is not None:

self.video\_capture.release()

self.video\_frame.pack\_forget()

def resize\_video(self, event):

# Mengubah ukuran frame video sesuai dengan ukuran window

if hasattr(self, 'video\_capture') and self.video\_capture is not None:

ret, frame = self.video\_capture.read()

if ret:

frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

frame = self.resize\_frame(frame)

img = self.convert\_frame\_to\_image(frame)

self.video\_frame.configure(image=img)

self.video\_frame.img = img

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

root = tk.Tk()

app = VideoPlayer(root)

root.mainloop()

# Menyesuaikan ukuran frame dengan ukuran window

if window\_width > 1 and window\_height > 1:

frame = cv2.resize(frame, (window\_width, window\_height))

return frame

def convert\_frame\_to\_image(self, frame):

# Konversi frame OpenCV menjadi format yang dapat ditampilkan di Tkinter

img = Image.fromarray(frame)

img = ImageTk.PhotoImage(image=img)

return img

def stop\_video(self):

# Menghentikan pemutaran video

if self.video\_capture is not None:

self.video\_capture.release()

self.video\_frame.pack\_forget()

def resize\_video(self, event):

# Mengubah ukuran frame video sesuai dengan ukuran window

if hasattr(self, 'video\_capture') and self.video\_capture is not None:

ret, frame = self.video\_capture.read()

if ret:

frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

frame = self.resize\_frame(frame)

img = self.convert\_frame\_to\_image(frame)

self.video\_frame.configure(image=img)

self.video\_frame.img = img

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

root = tk.Tk()

app = VideoPlayer(root)

root.mainloop()

Listing 37 Source Code MP4

##### **Pembahasan MP4**

###### **Inisialisasi dan Konfigurasi GUI**

Inisialisasi objek VideoPlayer dengan konstruktor \_\_init\_\_

, membuat jendela tkinter dengan judul "Responsive Video Player" dan ukuran awal 800x600, menambahkan binding untuk event perubahan ukuran window dengan metode resize\_video, membuat elemen-elemen GUI dalam metode create\_widgets.

class VideoPlayer:

def \_\_init\_\_(self, root):

# Inisialisasi atribut dan konfigurasi GUI

# ...

Listing 38 Source Code Inisialisasi Dan Konfigurasi GUI

###### **Desain Antarmuka Pengguna (GUI)**

Metode select\_video memungkinkan pengguna memilih file video, metode play\_video memulai pemutaran video dan audio dari file video yang dipilih, metode stop\_video menghentikan pemutaran video dan membersihkan elemen GUI.

def select\_video(self):

# Metode untuk memilih file video

# ...

def play\_video(self):

# Metode untuk memulai pemutaran video

# ...

def stop\_video(self):

# Metode untuk menghentikan pemutaran video

# ...

Listing 39 Source Code Desain Antarmuka GUI

###### **Metode Untuk Memilih Dan Menampilkan Video**

Metode select\_video memungkinkan pengguna memilih file video, metode play\_video memulai pemutaran video dan audio dari file video yang dipilih, metode stop\_video menghentikan pemutaran video dan membersihkan elemen GUI.

def select\_video(self):

# Metode untuk memilih file video

# ...

def play\_video(self):

# Metode untuk memulai pemutaran video

# ...

def stop\_video(self):

# Metode untuk menghentikan pemutaran video

# ...

Listing 40 Source Code Memilih Dan Menampilkan Video

###### **Metode untuk Memproses dan Menampilkan Video**

Metode resize\_frame mengubah ukuran frame video sesuai dengan ukuran window, convert\_frame\_to\_image mengonversi frame video dari format OpenCV ke format yang dapat ditampilkan di Tkinter, metode update\_frame membaca dan menampilkan setiap frame video secara terus menerus dengan memanfaatkan rekursi dan after dari Tkinter.

def resize\_frame(self, frame):

# Metode untuk mengubah ukuran frame video

# ...

def convert\_frame\_to\_image(self, frame):

# Metode untuk konversi frame video menjadi format yang dapat ditampilkan di Tkinter

# ...

def update\_frame(self):

# Metode untuk membaca dan menampilkan setiap frame video secara terus menerus

# ...

Listing 41 Source Code Memproses Dan Menampilkan Video

###### **Menjalankan Program MP4**

Bagian ini memastikan bahwa program dijalankan sebagai program utama, Membuat objek Tk dari tkinter dan objek Video Player, Menjalankan loop utama dengan root.mainloop()..

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

root = tk.Tk()

app = VideoPlayer(root)

root.mainloop()

Listing 42 Source Code Menjalankan Program MP4

#### **Image To Text:**

Dalam pertemuan kali ini kita akan membuat aplikasi python untuk membuat sebuah aplikasi converter image to text menggunakan python untuk dapat melakukanya kita sudah menyiapkan kode lengkap yang bisa anda gunakan sebagai referensi berikut adalah kodenya :

import tkinter as tk

from tkinter import ttk

import pyttsx3

class TextToSpeechApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Text to Speech")

# Label

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

self.label.grid(row=0, column=0, padx=10, pady=10)

# Entry untuk memasukkan teks

self.text\_entry = ttk.Entry(root, width=40)

self.text\_entry.grid(row=0, column=1, padx=10, pady=10)

# Tombol untuk memutar suara

self.button = ttk.Button(root, text="Putar Suara", command=self.text\_to\_speech)

self.button.grid(row=1, column=0, columnspan=2, pady=10)

def text\_to\_speech(self):

# Mengambil teks dari entry

text\_to\_convert = self.text\_entry.get()

# Inisialisasi mesin text-to-speech

engine = pyttsx3.init()

engine.setProperty('rate', 150) # Kecepatan bicara, bisa diubah sesuai kebutuhan

engine.say(text\_to\_convert)

engine.runAndWait()

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

root = tk.Tk()

app = TextToSpeechApp(root)

root.mainloop()

Listing 43 Source Code Image To Text

##### **Pembahasan Image to Text**

###### **Inisialisasi dan Konfigurasi GUI**

Inisialisasi objek TextToSpeechApp dengan konstruktor \_\_init\_\_ dan membuat jendela tkinter dengan judul "Text to Speech".

class TextToSpeechApp:

def \_\_init\_\_(self, root):

# Inisialisasi atribut dan konfigurasi GUI

# ...

Listing 44 Source Code Inisialisasi Dan Konfigurasi GUI

###### **Membuat Element (GUI)**

Ini digunakan untuk membuat label untuk memasukkan teks, membuat entry untuk memasukkan teks, Membuat tombol untuk memulai konversi teks ke suara.

def select\_video(self):

# Metode untuk memilih file video

# ...

def play\_video(self):

# Metode untuk memulai pemutaran video

# ...

def stop\_video(self):

# Metode untuk menghentikan pemutaran video

# ...

Listing 45 Source Code Membuat Element GUI

###### **Metode untuk Konversi Teks ke Suara**

Metode text\_to\_speech dipanggil saat tombol "Putar Suara" ditekan, mengambil teks dari entry menggunakan self.text\_entry.get(), menginisialisasi mesin text-to-speech (pyttsx3.init()), menyetel kecepatan bicara menggunakan engine.setProperty('rate', 150), Menyebutkan teks dan menjalankan mesin untuk memutar suara dengan engine.say(text\_to\_convert) dan engine.runAndWait().

def text\_to\_speech(self):

# Mengambil teks dari entry

text\_to\_convert = self.text\_entry.get()

# Inisialisasi mesin text-to-speech

engine = pyttsx3.init()

engine.setProperty('rate', 150) # Kecepatan bicara, bisa diubah sesuai kebutuhan

engine.say(text\_to\_convert)

engine.runAndWait()

Listing 46 Source Code Konversi Teks Ke Suara

###### **Menjalankan Program**

Memastikan bahwa program dijalankan sebagai program utama Membuat objek Tk dari tkinter dan objek TextToSpeechApp Menjalankan loop utama dengan root.mainloop().

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

root = tk.Tk()

app = TextToSpeechApp(root)

root.mainloop()

Listing 47 Source Code Menjalankan Program

##### **Latihan**

Silahkan mencoba kode yang sudah disediakan dan di praktekan kirim hasil screenshoot photo dari kode yang berhasil dijalankan dan harap untuk membaca dengan teliti soal yang diberikan, selamat mengerjakan!.

# **PERTEMUAN 7**

## **Materi Konverter Suhu Berbasis Tkinter**

### **Luas Permukaan Bangun Ruang Dan Volume Berbasis Tkinker.**

Materi ini bertujuan untuk memberikan pemahaman tentang penggunaan library `tkinter` dalam membuat aplikasi konversi suhu sederhana. Pada materi ini, kita akan belajar bagaimana membuat antarmuka pengguna menggunakan `tkinter` dan mengimplementasikan logika konversi suhu.

### **Pembahasan**

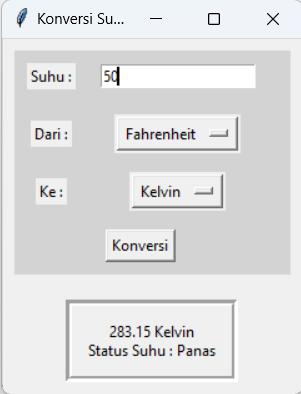
#### **Source Code Converter Suhu**

****

. Gambar 7 Source Code Converter Suhu

#### **Hasil Output**

Loop memainkan peran penting dalam pemrograman dengan memungkinkan eksekusi berulang dari serangkaian pernyataan. Loop FOR, misalnya, umumnya digunakan untuk mengulang melalui suatu urutan (seperti array atau rentang angka) dan melakukan operasi tertentu selama setiap iterasi. Di sisi lain, loop WHILE melanjutkan eksekusi selama kondisi tertentu tetap benar. Konstruksi perulangan ini berkontribusi pada efisiensi dan fleksibilitas program, memungkinkan otomatisasi tugas-tugas yang repetitif.

****

Gambar 8 Hasil Output

### **Latihan**

Buatlah sebuah aplikasi Conversi Suhu berbasis thinker!.

# **PERTEMUAN 8**

## **Ujian Tengah Semester**

Untuk didalam pertemuan kali ini tidak ada karena sudah masuk kedalam fase ujian tengah semester, persiapkan segala sesuatu yang telah kamu pelajari didalam pertemuan sebelumnya, jangan lupa belajar dan berdoa.

# **PERTEMUAN 9**

## **Materi Class Dan Object**

### **Belajar Membuat Aplikasi Berbasis Database.**

Materi ini bertujuan untuk memberikan pemahaman tentang pengembangan aplikasi menggunakan Python dengan memanfaatkan modul `tkinter` untuk antarmuka pengguna dan `mysql.connector` untuk interaksi dengan database MySQL. Tujuan utama adalah membimbing dalam membuat aplikasi manajemen data perawat dengan fungsi dasar seperti menambah, mengedit, menghapus, dan menampilkan data. Selain itu, materi ini juga bertujuan untuk memahamkan konsep penggunaan kelas, properti, dan metode dalam pengembangan aplikasi.

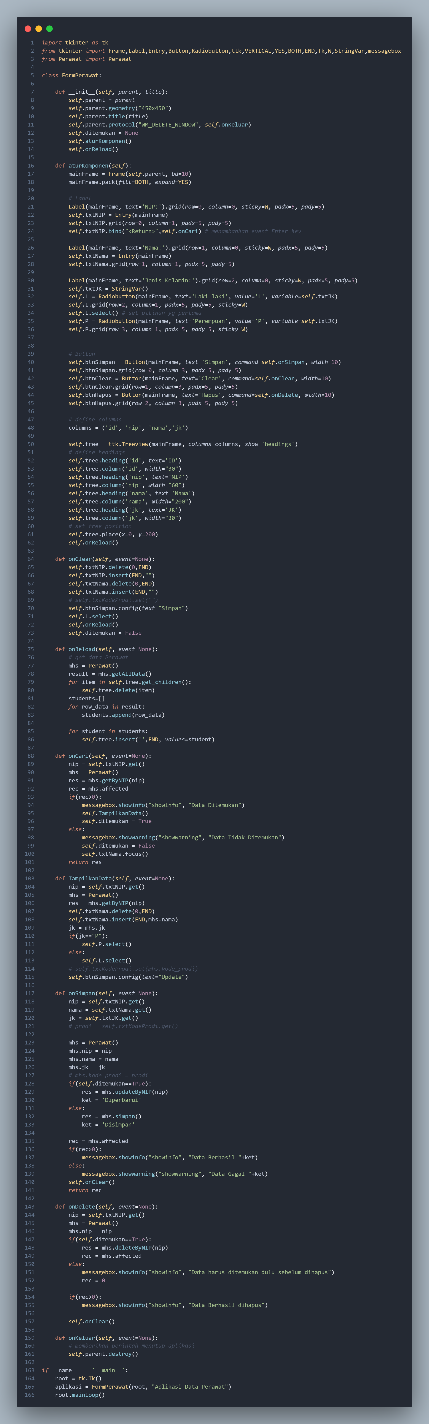
### **Pembahasan**

#### **Source Code DB.py**



. Gambar 9 Source Code DB.py

#### **FrmPerawat.py**



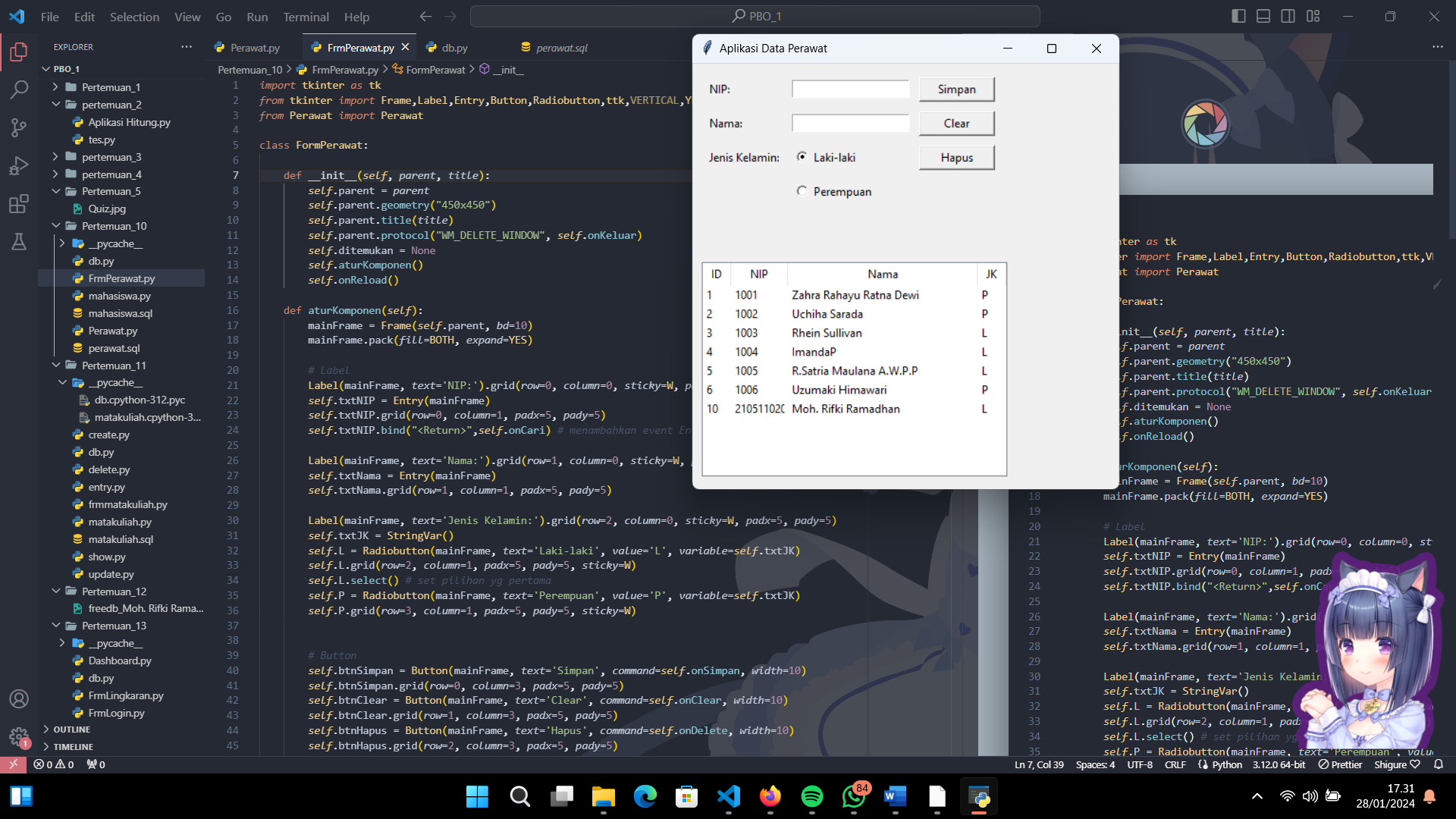
Gambar 10 FrmPerawat.py

#### **Perawat.py**

.

Gambar 11 Perawat.py

#### **Hasil Output**

****

Gambar 12 Hasil Output

### **Latihan**

Buatlah sebuah aplikasi Berbasis Database!.

# **PERTEMUAN 10**

## **Pengenalan Python Database Menggunakan XAMPP**

XAMPP adalah paket perangkat lunak sumber terbuka yang dirancang untuk menyediakan lingkungan server web lokal yang mudah diinstal dan digunakan. Nama XAMPP adalah singkatan dari "X" (menunjukkan sistem operasi apa pun), "Apache" (server web), "MySQL" (sistem manajemen basis data), "PHP" (bahasa pemrograman server-side), dan "Perl" (bahasa pemrograman serbaguna).

Berikut adalah komponen utama dalam paket XAMPP:

### **Apache HTTP Server**

XAMPP mencakup server web Apache, yang merupakan salah satu server web paling populer di dunia. Apache digunakan untuk menyajikan halaman web dan mendukung bahasa pemrograman server-side seperti PHP.

### **MySQlL**

MySQL adalah sistem manajemen basis data (Database Management System - DBMS) yang disertakan dalam XAMPP. Ini memungkinkan Anda membuat, mengelola, dan mengakses basis data yang diperlukan untuk aplikasi web.

### **PHP**

PHP adalah bahasa pemrograman server-side yang kuat dan umum digunakan untuk pengembangan web dinamis. XAMPP menyertakan PHP, yang dapat digunakan bersama dengan Apache untuk membuat situs web yang dinamis.

### **Perl**

Perl adalah bahasa pemrograman serbaguna yang juga disertakan dalam XAMPP. Meskipun PHP lebih umum digunakan untuk pengembangan web, Perl masih digunakan dalam beberapa konteks, terutama untuk pemrograman skrip dan pengolahan teks.

### **phpMyAdmin**

XAMPP menyertakan phpMyAdmin, sebuah antarmuka web yang memungkinkan pengguna untuk dengan mudah mengelola basis data MySQL melalui browser. Ini memberikan alat untuk membuat, mengedit, dan menghapus tabel, serta menjalankan kueri SQL.

Kelebihan utama XAMPP adalah kemudahan instalasinya. XAMPP memungkinkan pengembang web untuk membuat lingkungan server web lokal mereka sendiri tanpa harus menginstal setiap komponen secara terpisah. Ini sangat bermanfaat untuk pengembangan dan pengujian aplikasi web sebelum diluncurkan ke server web yang sebenarnya.

XAMPP tersedia untuk berbagai sistem operasi, termasuk Windows, macOS, dan Linux, membuatnya dapat digunakan di berbagai platform. Meskipun XAMPP sering digunakan untuk tujuan pengembangan dan pengujian, sebaiknya diingat bahwa lingkungan pengembangan lokal dapat memiliki perbedaan dengan server web produksi, dan pengujian yang lebih menyeluruh mungkin diperlukan sebelum meluncurkan aplikasi ke server yang sebenarnya.

## **Langkah-langkah instalisasi XAMPP :**

* **Unduh XAMPP:**

Kunjungi situs resmi XAMPP di https://www.apachefriends.org/index.html dan unduh versi terbaru yang sesuai dengan sistem operasi Anda (Windows, macOS, atau Linux).

* **Mulai Proses Instalasi:**

Setelah selesai mengunduh, buka file installer XAMPP yang telah diunduh dan mulai proses instalasi.

* **Pilih Komponen yang Akan Diinstal:**

Pada layar instalasi, pilih komponen yang ingin Anda instal. Komponen utama termasuk Apache, MySQL, PHP, dan phpMyAdmin. Pastikan untuk memilih komponen yang sesuai dengan kebutuhan pengembangan web Anda.

* **Pilih Folder Instalasi:**

Tentukan folder di mana Anda ingin menginstal XAMPP. Secara default, XAMPP akan diinstal di direktori C:\xampp pada sistem Windows.

* **Selesaikan Instalasi:**

Klik tombol "Next" atau "Install" untuk memulai proses instalasi. Tunggu hingga instalasi selesai.

* **Jalankan XAMPP Control Panel:**

Setelah instalasi selesai, jalankan XAMPP Control Panel. Anda dapat menemukan pintasan ke Control Panel di desktop atau melalui Start Menu.

* **Start Apache dan MySQL:**

Pada XAMPP Control Panel, klik tombol "Start" di sebelah modul Apache dan MySQL. Ini akan memulai server web Apache dan server database MySQL.

* **Uji Instalasi:**

Buka peramban web Anda dan kunjungi http://localhost. Jika instalasi berhasil, Anda akan melihat halaman selamat datang XAMPP.

* **Uji MySQL dengan phpMyAdmin:**

Buka phpMyAdmin di http://localhost/phpmyadmin. Masukkan nama pengguna dan kata sandi (secara default, nama pengguna adalah "root" dan kata sandi kosong).

# **PERTEMUAN 11**

## **MySQL Membuat Database Menggunakan CMD**

### **Python Database.**

Python Database dengan menggunakan XAMPP dapat merujuk pada penggunaan Python sebagai Bahasa pemograman untuk terhubung dan berinteraksi dengan database yang di hosting di server XAMPP. XAMPP sendiri adalah paket perangkat lunak yang menyediakan lingkungan pengembangan web yang mencakup Apache, MySQL, PHP, dan perl.

Dalam konteks ini, Python dapat digunakan sebagai alat untuk mengakses dan memanipulasi data di database MYSQL yang dijalankan di server XAMPP. Library atau modul Python yang umum digunakan untuk interaksi dengan database MySQL adalah “mysql-connector-python” atau “pymsql”. Dengan menggunakan modul-modul ini. Kita dapat membuat koneksi ke database MySQL di server XAMPP. Mengeksekusi query, dan mengelola data.

Ini memungkinkan pengembangan Python untuk mengintegrasikan fungsionalitas pemograman Python dengan database MySQL yang dikelola oleh sever XAMPP .

### **Pembahasan**

#### **Langkah-Langkah Membuat Database:**

* Jalankan XAMPP Control Panel
* File xampp/xampp-control.exe
* Klik tombol start pada : Apache dan Mysql
* Buka aplikasi phpMyAdmin
* <https://localhost/phpmyadmin>
* Membuat sebuah database baru dan beri nama : dbkampus
* Import table mahasiswa lewat file : mahasiswa.sql di menu import phpMyAdmin
* Install library mysql untuk python :

Lakukan instalisasi di terminal vscode dengan :

pip install mysql-connector-python

Listing 48 Source Code Instalasi Mysql

* Copy dan paste contoh database connection python :

<https://codefile.io/f/jZJ4xNBaOg>

#### **Koneksi Database:**

Untuk membuat koneksi database anda dapat mengcopy seluruh source code dibaah ini, berikut adalah source code nya :

import mysql.connector as mc

class DBConnection:

def \_\_init\_\_(self):

self.host = "localhost"

self.port = 3306

self.name = "dbkampus"

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 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:

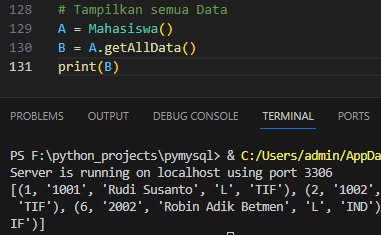
return "Server is offline."

Listing 49 Source db.py

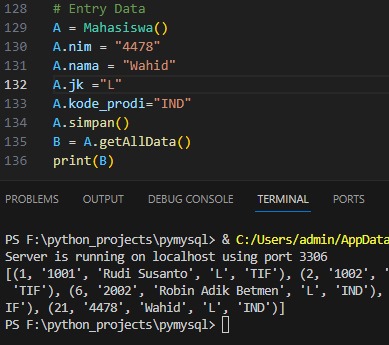
##### **Hasil**

Berikut adalah gambar hasil pengujian yang dilakukan.

###### **Pengujian Tampilkan Semua Data**

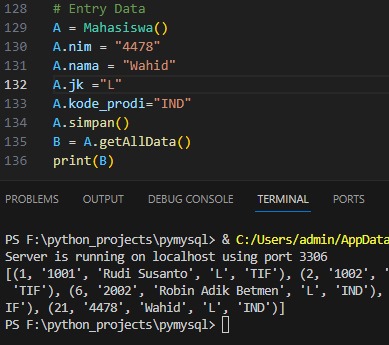
****Gambar 13 Tampilkan Semua Data

###### **Menguji Class Entry Mahasiswa**

****

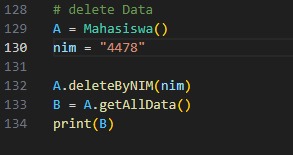
Gambar 14 Class Entry Mahasiswa

###### **Menguji Class Update Mahasiswa**

****

Gambar 15 Class Update Mahasiswa

###### **Menguji Class Delete Mahasiswa**

****

Gambar 16 Class DeleteMahasiswa

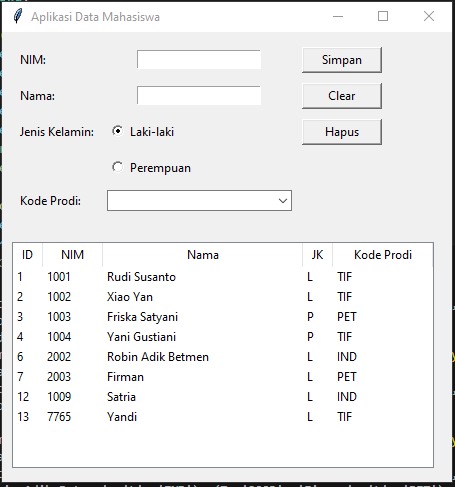
#### **Membuat Aplikasi CRUD Mahasiswa Berbasis Windows**

Buka link : <https://codefile.io/f/6JO1cxm2Tg>

Dan berinama Source Codenya denganFrmMahasiswa.py.

##### **Hasil**

Berikut adalah gambar Crud Mahasiswa Berbasis Windows.

****

Gambar 17 Crud Mahasiswa

### **Latihan**

Setelah kita belajar bagaimana struktur dan fungsi didalam pertemuan ini silahkan coba kodenya sebagai latihan!.

# **PERTEMUAN 12**

## **Python Database**

## **Python Application Database Using Remote Mysql Server On freedb.tech**

Freedb,tect merupakan situs hosting gratis untuk mysql yang bisa diakses atau dikoneksi secara remote. Database hosting gratis ini biasanya digunakan untuk testing atau devlopment antara aplikasi dengan sistem database yang terhubung secara online. Hosting gratis ini biasanya diincar oleh pelajar atau mahasiswa yang dipergunakan untuk edukasi atau pendidikan seperti tugas PR, tugas kuliah, skripsi, dsb.

Sebenarnya untuk melakukan koneksi database MySQL secara online ini tidak ada bedanya dengan koneksi database MySQL yang terletak server lokal seperti XAMPP, LAMP, dan sebagainya. Bedanya cuma bisa dilakukan remote sehingga satu database MySQL dapat diakses oleh beberapa perangkat atau hosting yang berbeda.

Sayangnya, koneksi Database MySQL secara remote dan online ini bisa membahayakan data informasi yang tersimpan di dalam sistem tersebut jika tidak menutupi celah lubang yang menyebabkan kebobolan data yang dilakukan oleh Cyber IT. Jadi, MySQL hosting gratis ini digunakan untuk belajar dalam melakukan koneksi database MySQL ke dalam aplikasi secara remote.

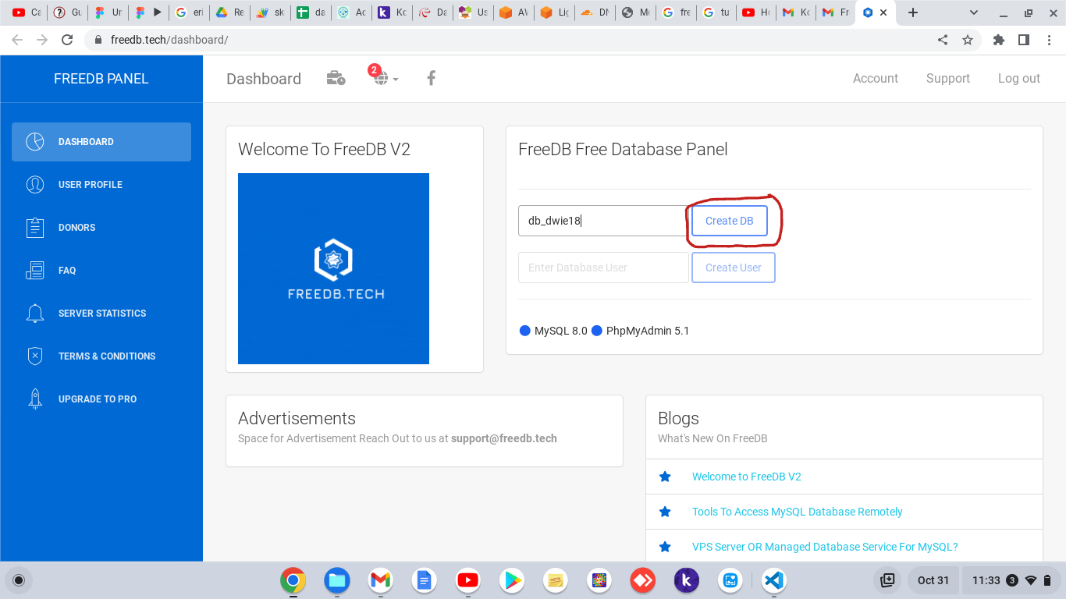
## **Pembahasan**

#### **Instalasi**

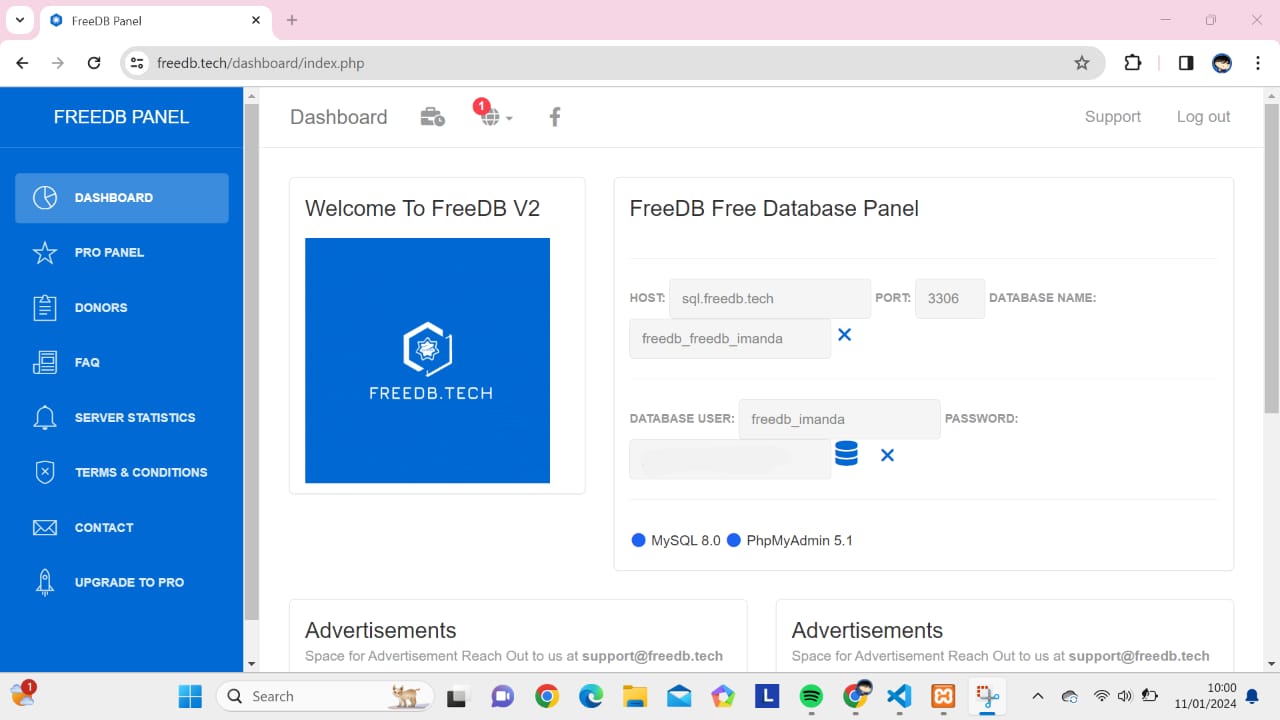
**Langkah-Langkah Registasi Akun :**

* Kunjungi situs https://freedb.tech untuk melakukan daftar akun.
* sikan formulir yang berupa email, username, dan password dengan tepat dan benar, klik Register.
* Isikan formulir untuk survei akun dengan sesuai jawaban kamu, klik Submit.
* Kunjungi situs Mail yang didaftarkan akun FreeDB.Tech tadi untuk melakukan verifikasi akun.
* uka isi kotak masuk gmail yang pengirim dari FreeDB.Tech.
* Klik link “Verify Your Account”.
* Buatlah database baru dengan sesuai kebutuhan kamu, klik Create DB.

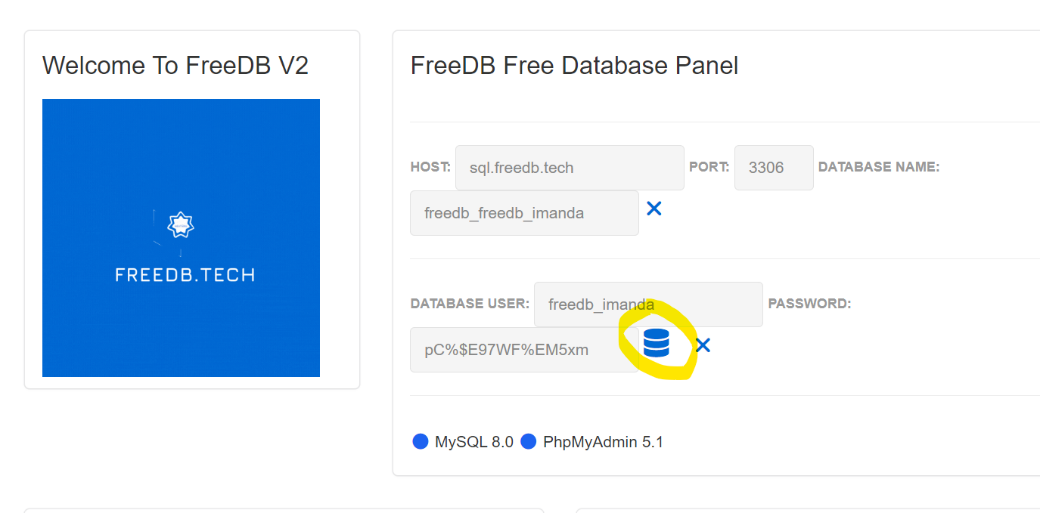
#### **Hasil**



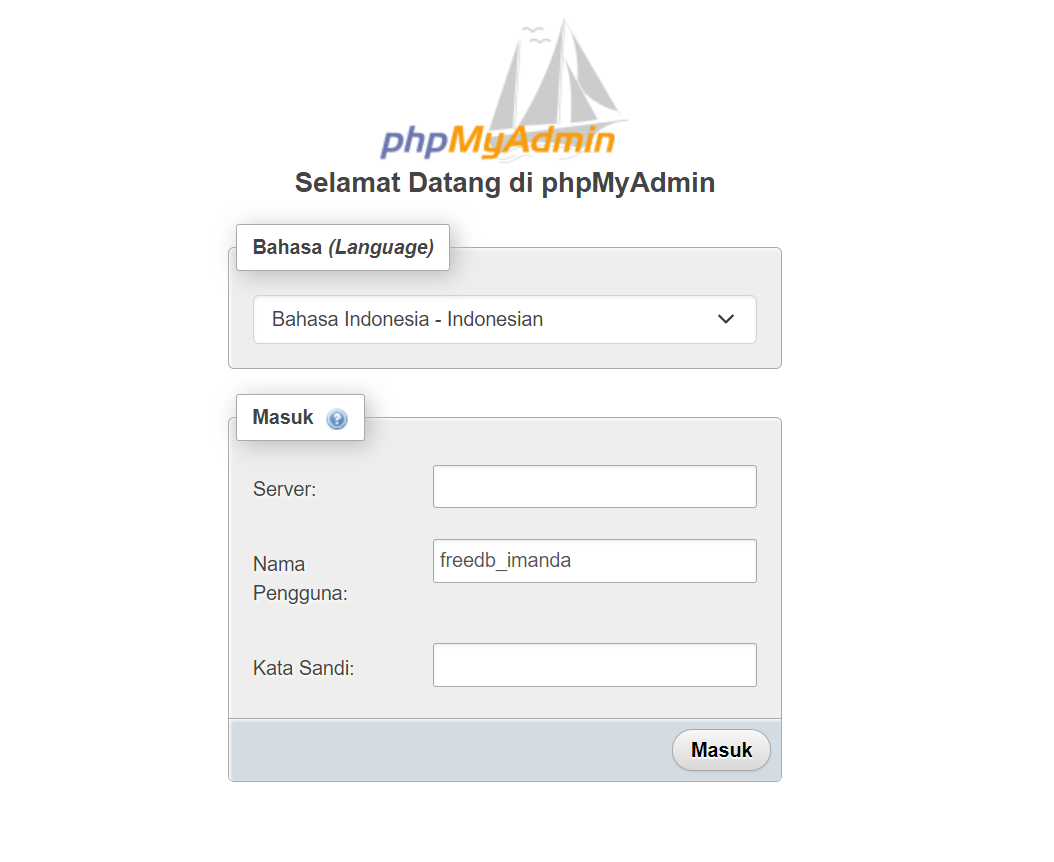
Gambar 18 Langkah Membuat Database



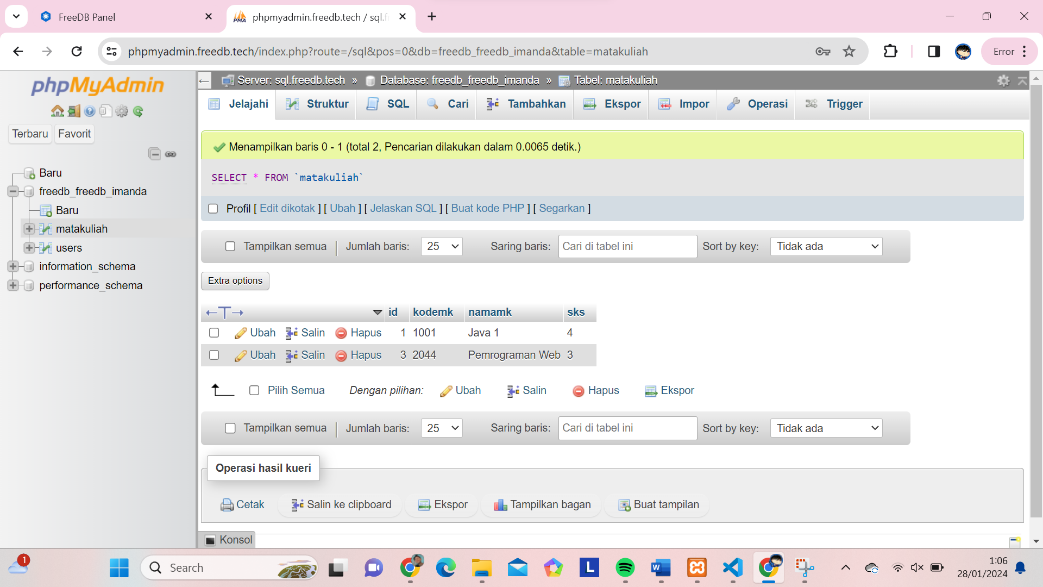
Gambar 19 Dashboard



Gambar 20 Create Database Online



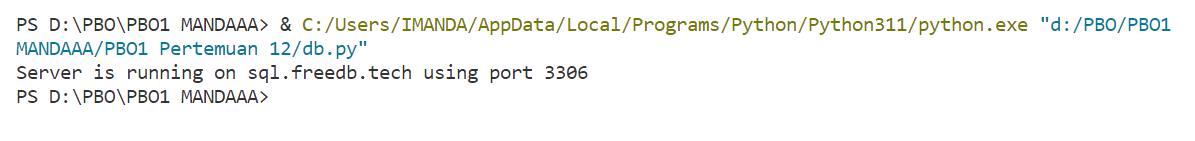
Gambar 21 Login Cpanel



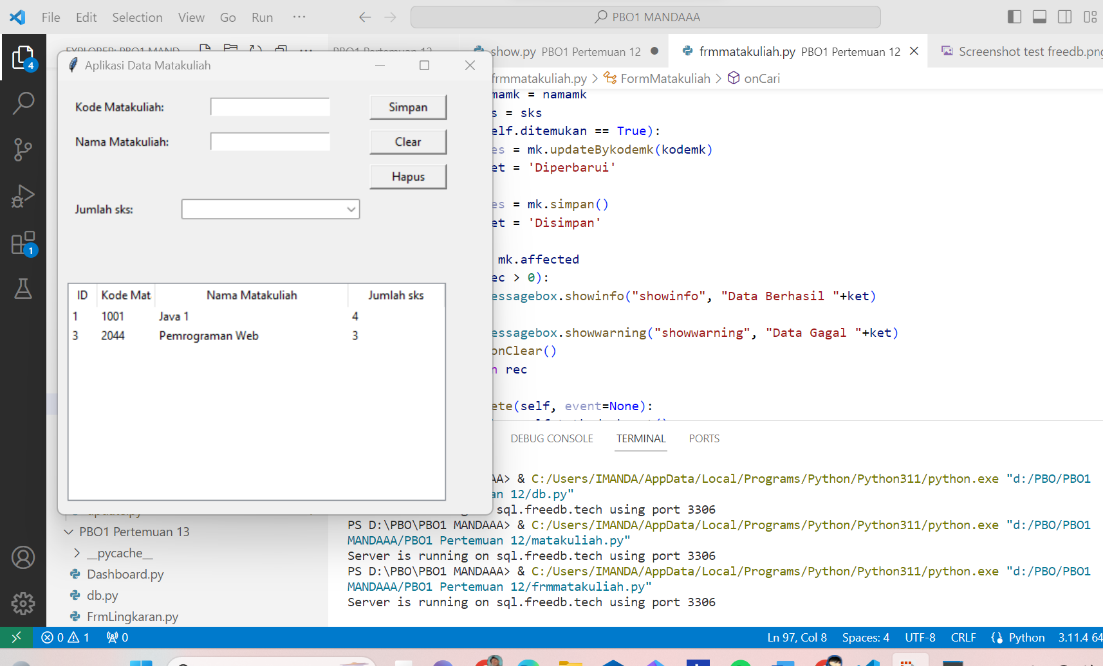
Gambar 22 Database Cpanel



Gambar 23 Koneksi Database



Gambar 24 Hasil Testing



Gambar 25 Crud Mahasiswa Menggunakan Koneksi Database

##### **Latihan**

Silahkan buat database milikmu sendiri!.

# **PERTEMUAN 13**

## **Membuat Aplikasi Login Multi-User Python-Mysql**

Aplikasi login multi-user memainkan peran yang sangat penting untuk memberikan akses yang aman dan terkelola ke sistem atau layanan. Dalam konteks ini, menciptakan aplikasi login multi-user dengan menggunakan Python sebagai Bahasa pemograman dan database phpMyAdmin sebagai penyimpanan data adalah Langkah yang baik.

Aplikasi ini menggabungkan kekuatan Python sebagai bahasa pemrograman yang fleksibel dan mudah dimengerti dengan kemudahan administrasi basis data menggunakan PHPMyAdmin. Dengan menyimpan informasi pengguna di dalam database yang dikelola oleh PHPMyAdmin, kita dapat dengan efisien mengelola akun pengguna, mengamankan kata sandi, dan memberikan hak akses yang sesuai.

Dalam pembahasan selanjutnya, kita akan menjelajahi langkah-langkah praktis untuk merancang dan mengimplementasikan aplikasi login multi-user ini. Mulai dari desain basis data hingga penggunaan Python untuk terhubung dan berinteraksi dengan database PHPMyAdmin, kita akan memastikan bahwa aplikasi ini tidak hanya efektif dalam manajemen pengguna tetapi juga memberikan tingkat keamanan yang optimal.

Dengan penerapan konsep ini, diharapkan aplikasi login multi-user Python dengan database PHPMyAdmin dapat memberikan pengalaman pengguna yang aman dan nyaman dalam mengakses sistem atau aplikasi yang dikembangkan.

## **Langkah Instalasi**

Freedb,tect merupakan situs hosting gratis untuk mysql yang bisa diakses

Langkah-Langkah

1. Yang pertama login pada https://freedb.tech

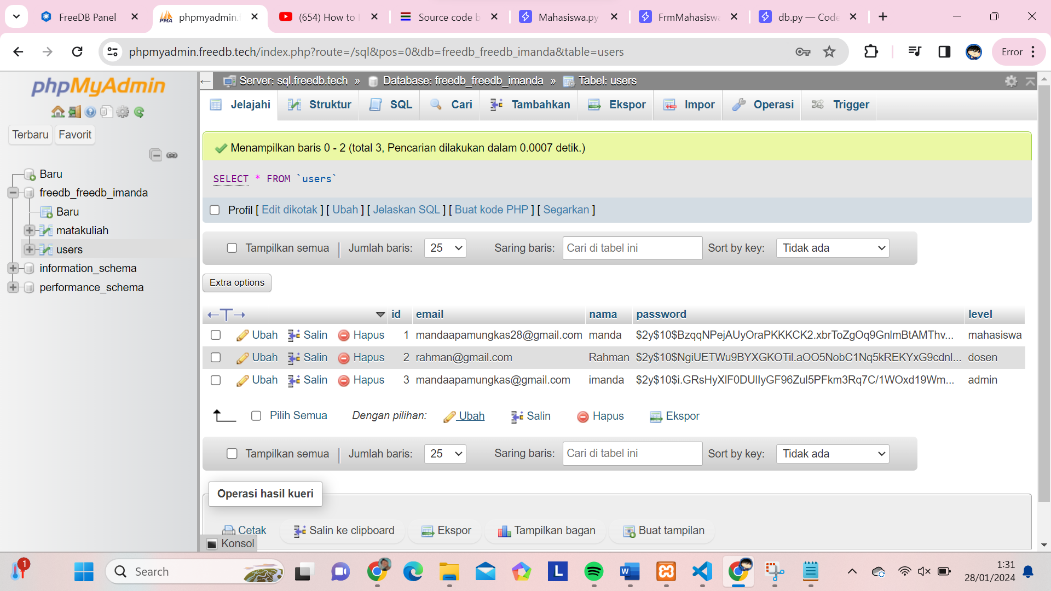
2. Lalu masuk ke : https://phpmyadmin.freedb.tech/

3. Masukkan Server, Nama Pengguna, dan Kata Sandi

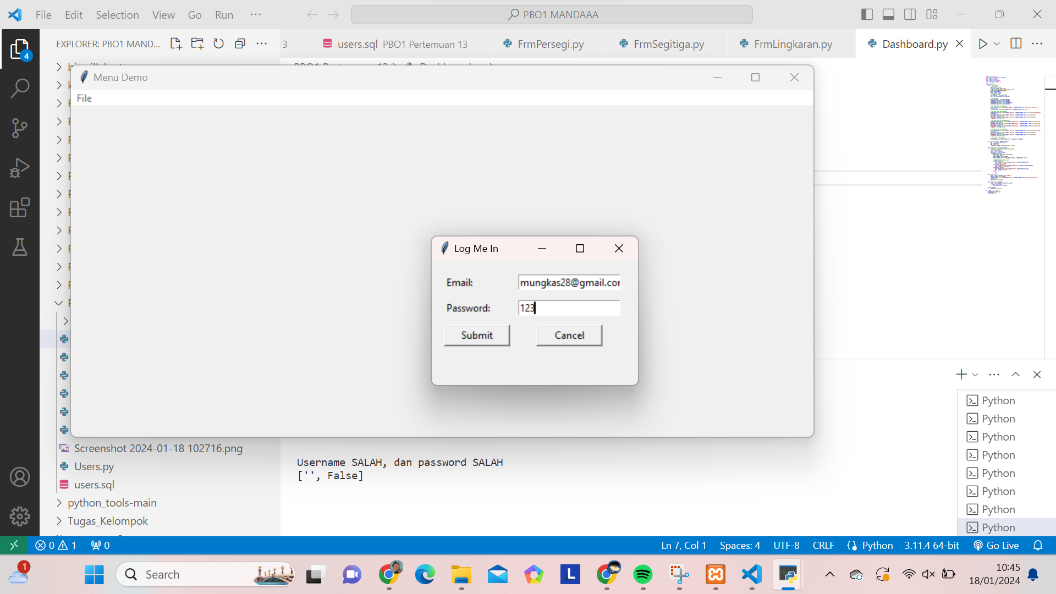
4. Masuk

5. Import file : users.sql

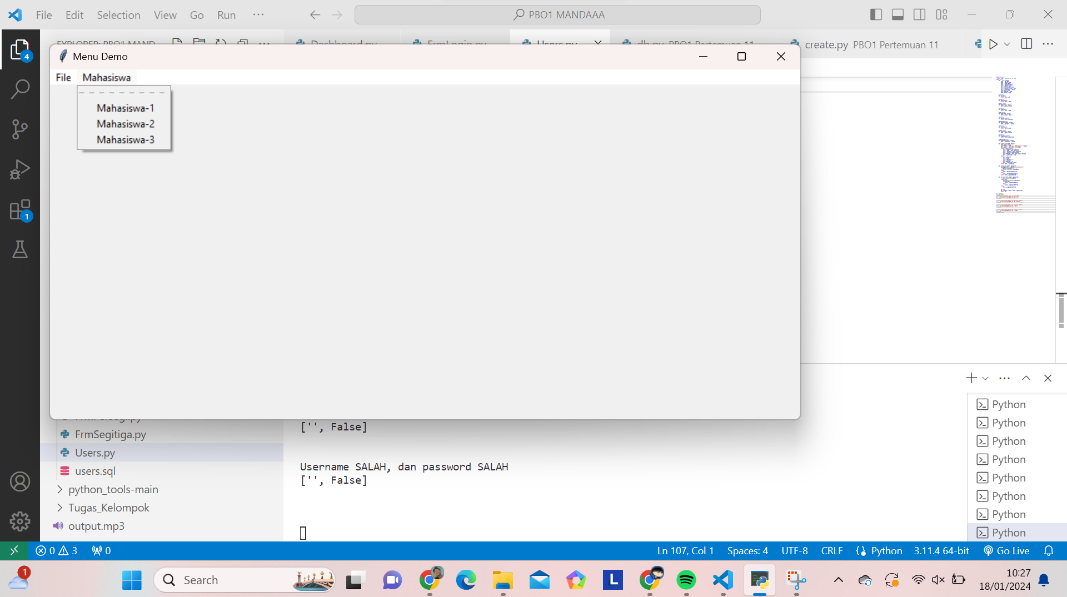
## Hasil



Gambar 26 Langkah Membuat Database

****

Gambar 27 Aplikasi Login

****

Gambar 28 Dashboard Aplikasi

# **PERTEMUAN 14**

## **Membuat Buku Modul Praktikum 14 Pertemuan**

Untuk pertemuan ini adalah Pembuatan Modul Praktikum dengan 14 pertemuan dikerjakan dengan berkelompok maksimal 3 orang dan dikumpulkan paling lambat 28/01/2024.