Jurnal Informatika dan Komputer (JIK)

Vol. 15 No. 2 (2024) | ISSN Media Cetak: 2089 – 4384

IMPLEMENTASI PEMBELAJARAN PEMROGRAMAN DATABASE MENGGUNAKAN PYTHON

Muhammad Romzi[1], Budi Kurniawan[2]
[12]Program Studi Manajemen Informatika, Universitas Mahakarya Asia
[12]Jalan A. Yani. No. 267 A. Baturaja 32113 INDONESIA
Telp: 0735-326169; fax: 0735-326169;
e-mail: ujromzi@gmail.com[1] , budi.skom@gmail.com[2]

Abstract- ***Programming, especially database programming, is one of the essential skills in software development, enabling efficient data management and storage. This study aims to implement database programming learning using the Python programming language, with a focus on using Tkinter, the standard GUI toolkit in Python, and the sqlite3 module as a relational database management system. The learning process covers the creation of windows and components or widgets that can be added to the window, as well as examples of creating programs that are connected to database management, such as storing data, searching for data, finding, and displaying data from the database. Through this implementation, it is hoped that readers will understand the importance of data management in programming and master the basic techniques in designing and building database-driven applications using Python. Additionally, this work provides a strong foundation for beginner developers who are interested in deepening their knowledge of database programming and data-driven applications.***

Keywords**:** ***Database Programming, Python, SQLite Desktop Application, Tkinter***

Intisari- **Pemrograman, khususnya pemrogram database merupakan salah satu keterampilan penting dalam pengembangan perangkat lunak, yang memungkinkan pengelolaan dan penyimpanan data secara efisien. Penelitian ini bertujuan untuk mengimplementasikan pembelajaran pemrograman database menggunakan bahasa pemrograman Python, dengan fokus pada penggunaan Tkinter yang merupakan toolkit GUI standar di Python, dan modul sqlite3 sebagai sistem manajemen database relasional. Pembelajaran ini membahas tentang pembuatan window dan komponen-komponen atau *widget* yang dapat ditambahkan pada window, serta contoh pembuatan program yang terhubung dengan pengelolaan database seperti menyimpan data, pencarian data, menemukan dan menampilkan data yang ada pada database Dengan implementasi pembelajaran ini, diharapkan para pembaca dapat memahami pentingnya pengelolaan data dalam pemrograman, serta menguasai teknik dasar dalam merancang dan membangun aplikasi yang berbasis database menggunakan Python. Selain itu, tulisan ini memberikan dasar-dasar yang kuat bagi para pengembang pemula yang tertarik untuk memperdalam pengetahuan mereka tentang pemrograman database dan aplikasi berbasis data.**

Kata Kunci: Pemrograman Database, Python, SQLite, Aplikasi Desktop, Tkinter

I. PENDAHULUAN

Pemrograman merupakan keterampilan yang semakin penting di era digital ini, di mana kemampuan untuk memahami dan mengembangkan perangkat lunak sangat dibutuhkan dalam berbagai bidang. Belajar pemrograman tidak selalu mudah bagi pemula,

akan tetapi bukan juga sesuatu yang sulit, sehingga menjadikan enggan untuk belajar pemrograman.

Pembelajaran pemrograman dapat dimulai dari memahami konsep-konsep dasar, seperti logika pemrograman, penggunaan variabel, dan struktur data. Konsep dasar pemrograman seperti, runtunan, percabangan, dan perulangan dapat dipelajari secara ringkas menggunakan terminal atau konsol teks. Setelah memahami konsep dasar pemrograman melalui terminal atau konsol teks, pembelajaran dapat dilanjutkan dengan mempelajari penggunaan *Graphical User Interface* (GUI) untuk memvisualisasikan output dari kode yang ditulis dan memahami bagaimana proses pemrograman bekerja dalam konteks aplikasi nyata.

Python, sebagai salah satu bahasa pemrograman populer, sering dijadikan pilihan karena sintaksisnya yang sederhana dan dukungan yang luas terhadap berbagai pustaka (*libraries*) dan alat bantu pengembangan. Python juga mendukung pembelajaran yang berorientasi pada pengembangan aplikasi yang mudah dan terstruktur serta memiliki berbagai pustaka dan framework yang mendukung pengelolaan database, seperti SQLite, MySQL, dan PostgreSQL.

Penelitian ini bertujuan untuk mengimplementasikan dan mempelajari penerapan pemrograman database menggunakan Python, dengan fokus pada pemahaman cara membuat form aplikasi, dan menghubungkan dengan dengan database SQLite.

II. KAJIAN TEORI

2.1 Pemrograman Database

Pemrograman database adalah proses pembuatan, pengelolaan, dan pemeliharaan database dengan menggunakan bahasa pemrograman. Proses tersebut merupakan interaksi antara perangkat lunak dan sistem database untuk menyimpan, mengorganisasi, dan mengambil data secara efisien. Konsep penting yang perlu dipahami adalah dalam pemrograman database adalah:

a. **Database Management System (DBMS):** Sistem yang mengelola data dan memungkinkan pengguna atau aplikasi untuk mengakses serta memodifikasi data. Contoh DBMS antara lain MySQL, PostgreSQL, SQLite, dan MongoDB.
b. **Struktur Data Relasional**: Pada pemrograman database, data disusun dalam bentuk tabel-tabel yang saling terhubung.
c. **SQL (Structured Query Language)**: SQL adalah bahasa yang digunakan untuk mengakses dan memanipulasi data dalam database.
d. **Keamanan dan Optimasi**: Pemrograman database mencakup pertimbangan tentang keamanan data, pengelolaan transaksi, dan optimasi performa untuk memastikan data dikelola dengan aman dan aplikasi berjalan dengan efisien.

Pemrosesan data yang efisien sangat bergantung pada struktur database [1].

2.2 Python

Python diciptakan oleh Guido van Rossum di Belanda pada tahun 1990 dan namanya diambil dari acara televisi kesukaan Guido Monty Python's Flying Circus. Van Rossum mengembangkan Python sebagai hobi, kemudian Python menjadi bahasa pemrograman yang dipakai secara luas dalam industri dan pendidikan [2]. Bahasa pemrograman Python dapat digunakan untuk bekerja dengan database menggunakan pustaka seperti sqlite3, SQLAlchemy, dan MySQLdb. Python memiliki koleksi kepustakaan yang banyak, tersedia modul modulyang 'siap pakai' untuk berbagai keperluan; 2. Memiliki struktur bahasa yang jelas, sederhana, dan mudah dipelajari. 3.Berorientasi objek; 4. Memiliki sistempengelolaan memori otomatis; 5. Bersifat modular. [3]

2.3 TkInter

Phyton memiliki 12 modul untuk membuat aplikasi Graphical User Interface (GUI) salah satunya adalah Tkinter [3]. **Tkinter** adalah pustaka Python yang digunakan untuk

membangun antarmuka pengguna grafis (GUI). Tkinter adalah toolkit GUI standar di Python, pengguna tidak perlu menginstal pustaka tambahan karena sudah termasuk dalam distribusi Python. Tkinter digunakan untuk membuat aplikasi desktop yang memiliki elemen-elemen grafis seperti jendela, tombol, kotak teks, label, dan lainnya.

Tkinter menyediakan berbagai komponen atau widget yang dapat digunakan untuk membuat aplikasi GUI. Berikut adalah beberapa widget utama dalam Tkinter:

a. **Window (Jendela)**
   Jendela utama aplikasi yang berfungsi sebagai wadah bagi widget lainnya.
b. **Label**
   Digunakan untuk menampilkan teks atau gambar pada jendela.
c. **Button**
   Tombol yang dapat diklik untuk menjalankan perintah tertentu.
d. **Entry**
   Kotak teks satu baris yang digunakan untuk memasukkan data.
e. **Text**
   Kotak teks multiline yang digunakan untuk memasukkan atau menampilkan teks yang lebih panjang.
f. **Checkbutton**
   Tombol centang yang memungkinkan pengguna memilih opsi.
g. **Radiobutton**
   Tombol radio untuk memilih salah satu opsi dari beberapa pilihan
h. **Listbox**
   Menampilkan daftar item yang dapat dipilih.
i. **Canvas**
   Widget untuk menggambar bentuk-bentuk grafis seperti garis, lingkaran, dan gambar.
j. **Scrollbar**
   Digunakan untuk menambahkan bar gulir ke widget seperti Text atau Listbox.

Tkinter memiliki tiga layout manager untuk menata posisi dan ukuran widget dalam jendela aplikasi. Layout manager menentukan bagaimana widget ditempatkan di dalam jendela.

a. **pack** ()
   pack() adalah layout manager yang menyusun widget secara vertikal atau horizontal secara otomatis.
b. **grid ()**
   grid () digunakan untuk menyusun widget dalam bentuk grid berbasis baris dan kolom.
c. **place ()**
   place () untuk menentukan posisi widget menggunakan koordinat spesifik.

Tkinter mendukung *event handling*, yaitu menangani aksi yang dilakukan pengguna, seperti klik tombol, penekanan tombol keyboard, dan gerakan mouse[4].

2.4 SQLite

SQLite adalah sistem manajemen database relasional (RDBMS) yang ringan dan berbasis serverless. SQLite sangat praktis untuk aplikasi yang membutuhkan penyimpanan data namun tidak memerlukan fitur-fitur kompleks. Beberapa poin yang ada pada SQLite antara lain:

a. Sederhana dan Mudah Digunakan
   SQLite tidak membutuhkan konfigurasi atau pengaturan jaringan karena bekerja dalam mode *embedded*, database SQLite adalah satu file tunggal yang langsung dapat digunakan. SQLite tidak perlu mengelola proses server secara terpisah.
b. Berbasis File Tunggal
   Seluruh database (termasuk tabel, data, indeks, dan skema) disimpan dalam satu file dengan ekstensi .sqlite atau .db.
c. Kompatibilitas dengan Bahasa Pemrograman
   SQLite dapat digunakan dengan berbagai bahasa pemrograman, seperti Python, JavaScript, C++, dan lainnya. Dalam Python, modul sqlite3 yang memungkinkan pengguna membuat dan mengelola database SQLite dengan mudah.
d. Mendukung SQL Standar
   SQLite mendukung perintah SQL dasar seperti select, insert, update, dan delete, serta

pembuatan *index*, *transaction*, dan *join*, yang diperlukan untuk pengelolaan data.

e. Kapasitas Data
SQLite memungkinkan digunakan untuk aplikasi yang membutuhkan penyimpanan data menengah.

f. Keamanan dan Stabilitas
SQLite telah digunakan secara luas dalam berbagai perangkat dan aplikasi, seperti browser web, aplikasi ponsel, dan lainnya.

2.5 Visual Studio Code

Visual Studio Code adalah editor kode sumber yang ringan namun kuat yang berjalan didesktop dan tersedia untuk Windows, macOS,dan Linux. Muncul dengan dukungan built-in untuk JavaScript, TypeScript dan Node.js dan memiliki ekosistem ekstensi yang kaya untuk bahasa lain (seperti C ++, C #, Java, Python, PHP, Go) dan runtime (seperti .NET dan Unity) [5].

III. METODE PENELITIAN

Penelitian ini menggunakan metode pengembangan atau Research and Development (R&D), yang bertujuan untuk merancang dan mengembangkan materi belajar pemrograman Python. Materi pembelajaran yang dibahas adalah: Penggunaan komponen atau widget pada TkInter dan pembuatan program menggunakan database dengan operasi penyimpanan data, pencarian data, dan menampilkan data.

IV. HASIL DAN PEMBAHASAN

Hasil dan pembahasan pembelajaran pemrograman database, dimulai dari dasar-dasar pembuatan jendela serta komponen-komponennya, kemudian dilanjutkan dengan pembuatan progam.

1) **Membuat Window**

```
#program1
import tkinter as tk
window = tk.Tk()
window.mainloop()
```

Kode program menghasilkan tampilan window seperti gambar 1.

Gambar 1. Tampilan Window Sederhana

Pembahasan Kode Program

- **import tkinter as tk**, adalah perintah untuk mengimpor pustaka Tkinter dengan nama alias tk.
- window = **tk.Tk()**, adalah fungsi untuk membuat jendela
- window.**mainloop()**, adalah perulangan utama yang membuat jendela tetap aktif sampai ditutup oleh pengguna.

Beberapa pengaturan lainnya terkait window adalah:

- window.**title()** perintah untuk mengatur judul dari jendela.
- window.**geometry()** perintah untuk mengatur ukuran jendela dalam format "lebar x tinggi".

2) **Menambahkan Label dan Buttom**

```
#program2
import tkinter as tk
window = tk.Tk()
window.title("Jendela Tkinter")
window.geometry("400x300")

label=tk.Label(window,text="Selama
  t Belajar!", font=("Arial", 14))
label.pack(pady=20)
close_button = tk.Button(window,
  text="Tutup",command=window.quit)
close_button.pack(pady=10)
window.mainloop()
```

Kode program menghasilkan tampilan window seperti gambar 2.

Gambar 2. Label dan Button pada Window

Pembahasan Kode Program

- **tk.Label()** digunakan untuk menambahkan label teks yang ditempatkan di dalam jendela menggunakan **pack()** untuk tata letaknya, sedangkan perintah ***pady*** akan memberi jarak di atas dan bawah label
- **tk.Button()** digunakan untuk menambahkan button, perintah **window.quit** perintah untuk menutup aplikasi saat diklik.

3) **Menambahkan Entry**

Kotak Entry dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
entry=tk.Entry(window)
entry.pack(pady=20)
```

Kode program menghasilkan tampilan window seperti gambar 3.

Gambar 3. Entry pada Window

Pembahasan Kode Program

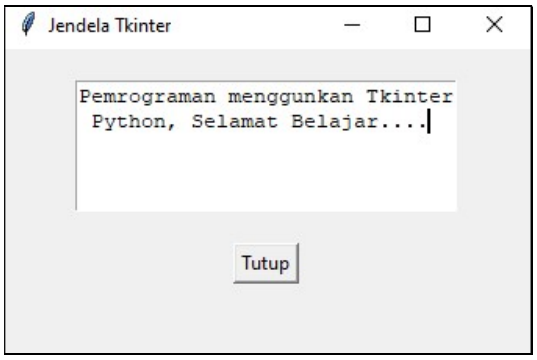
- **tk.Entry()** digunakan untuk menambahkan kotak teks satu baris untuk memasukkan data yang ditempatkan di dalam jendela menggunakan **pack()** untuk tata letaknya, sedangkan perintah ***pady*** akan memberi jarak di atas dan bawah entry.

4) **Text**

Kotak Text dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
text_box=tk.Text   window,height=5,
width=30)
text_box.pack(pady=20)
```

Kode program menghasilkan tampilan window seperti gambar 4.



Gambar 4. Teks pada Window

Pembahasan Kode Program

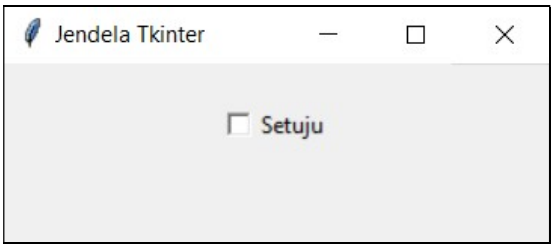
- **tk.Text()** digunakan untuk menambahkan kotak teks multiline yang digunakan untuk memasukkan atau menampilkan teks yang lebih panjang di dalam jendela. **height**=5, **width**=30 menyatakan tinggi dan lebar kotak teks, serta menggunakan **pack()** untuk tata letaknya, sedangkan perintah ***pady*** akan memberi jarak di atas dan bawah entry.

5) **Checkbutton**

Checkbutton dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
check_var=tk.IntVar()
check_button=tk.Checkbutton
(window,text="Setuju",variable=che
ck_var)
check_button.pack()
```

Kode program menghasilkan tampilan window seperti gambar 5.



Gambar 5. Checkbutton pada Window

Pembahasan Kode Program

- **check_var = tk.IntVar()**, check_var adalah variabel yang akan menyimpan status Checkbutton (dicentang atau tidak). **tk.IntVar()** adalah tipe variabel khusus di Tkinter yang menyimpan nilai integer. Jika Checkbutton dicentang, check_var akan bernilai 1. Jika tidak dicentang, check_var akan bernilai 0.
- **check_button = tk.Checkbutton(window, text="Setuju", variable=check_var)**, perintah untuk membuat sebuah **Checkbutton** yang ditampilkan di dalam window. **text="Setuju"** adalah teks yang akan ditampilkan di sebelah Checkbutton. **variable=check_var** mengaitkan status Checkbutton dengan variabel check_var, sehingga nilai check_var akan berubah sesuai dengan status Checkbutton.
- **check_button.pack()** digunakan untuk menampilkan Checkbutton di jendela Tkinter (window) dengan tata letak otomatis.

6) **Radiobutton**

Radiobutton dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
radio_var = tk.IntVar()
radio_button1=tk.Radiobutton
(window,text="Pilihan1",
variable=radio_var, value=1)
radio_button2=tk.Radiobutton
(window,text="Pilihan2",
variable=radio_var, value=2)
radio_button1.pack()
radio_button2.pack()
```

Kode program menghasilkan tampilan window seperti gambar 6.

Gambar 6. Radiobutton pada Window

Pembahasan Kode Program

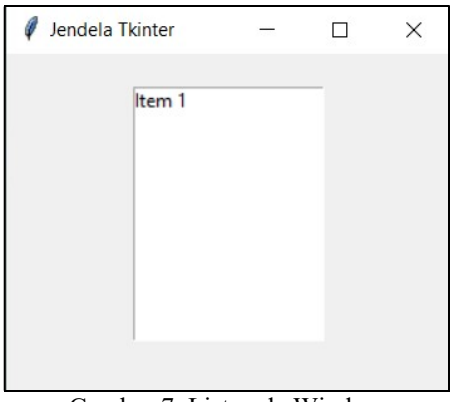
- **radio_var** adalah variabel dari tipe **IntVar()** yang akan digunakan untuk menyimpan nilai dari tombol radio yang dipilih. Jika tombol radio pertama dipilih, nilainya akan menjadi 1, dan jika tombol kedua dipilih, nilainya akan menjadi 2.
- **window** adalah jendela utama tempat tombol radio akan ditampilkan.
- **text="Pilihan1"** adalah teks yang akan ditampilkan di sebelah tombol radio.
- **variable=radio_var** untuk mengaitkan tombol radio ini dengan variabel radio_var.
- **value=1**, jika tombol radio ini dipilih, radio_var akan bernilai 1, **value=2**. Jika tombol ini dipilih, radio_var akan bernilai 2.
- **pack()** digunakan untuk menempatkan tombol-tombol radio pada dalam window.

7) **Listbox**

Listbox dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
listbox = tk.Listbox(window)
listbox.pack()
listbox.insert(tk.END, "Item 1")
```

Kode program menghasilkan tampilan window seperti gambar 7.



Gambar 7. List pada Window

Pembahasan Kode Program

- **tk.Listbox(window)**: untuk membuat komponen *listbox* pada window.
- **listbox**: variabel untuk menyimpan referensi ke objek *listbox* yang dibuat,
- **listbox.pack()**: metode pack() digunakan untuk menempatkan komponen *listbox* di dalam window.
- **listbox.insert (tk.END,"Item1")** :untuk menyisipkan item baru dengan teks "Item 1" ke dalam *listbox*.
- **tk.END**: parameter menunjukkan posisi terakhir di dalam *listbox*.

8) **Canvas**

Listbox dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
canvas=tk.Canvas(window,
width=400, height=300)
canvas.pack()
canvas.create line(0,0,400,300)
```

Kode program menghasilkan tampilan window seperti gambar 8.

Gambar 8. Canvas pada Window

Pembahasan Kode Program

- **tk.Canvas** (window, width=400, height=300): untuk objek canvas pada window, dengan lebar 400 piksel dan tinggi 300 piksel.
- **canvas**: variabel untuk menyimpan referensi ke objek *canvas*.
- **canvas.pack()**: metode pack() digunakan untuk menempatkan canvas di dalam window.
- **canvas.create_line(0, 0, 400, 300)**: menggambar garis pada canvas dari titik awal (0, 0) ke titik akhir (400, 300). (0, 0): titik awal garis pada koordinat kiri atas canvas. (400, 300): titik akhir garis pada koordinat kanan bawah canvas.

9) **Scrollbar**

Listbox dapat ditambahkan pada window, dengan mengetikkan kode berikut:

```
scrollbar=tk.Scrollbar(window)
scrollbar.pack(side=tk.RIGHT,
fill=tk.Y)
```

Kode program menghasilkan tampilan window seperti gambar 9.

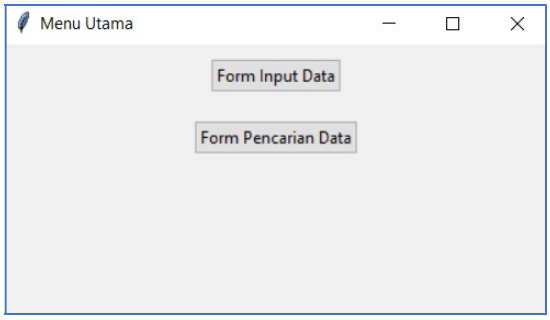
Gambar 9. Scrollbar pada Window

Pembahasan Kode Program

- **tk.Scrollbar(window)**: membuat objek *scrollbar* di dalam window.
- **scrollbar**: variabel untuk menyimpan referensi ke objek *scrollbar*.
- **scrollbar.pack(...)**: metode pack() digunakan untuk menempatkan *scrollbar* di dalam window.
- **side=tk.RIGHT**: menentukan agar *scrollbar* diletakkan di sisi kanan jendela.
- **fill=tk.Y**: Mengatur *scrollbar* untuk memenuhi ruang vertikal (sumbu Y)

dari jendela, sehingga panjangnya sesuai dengan tinggi jendela.

Setelah mempelajari pembuatan window dan penambahan komponen-komponen pada window. Selanjutnya akan dibahas tentang pembuatan program python menggunakan database sqlite. Penjelasan kode program, ditulis sebagai komentar pada bagian atas kode program, untuk memudahkan pembeljaran. Berikut contoh tampilan program yang akan dibuat.



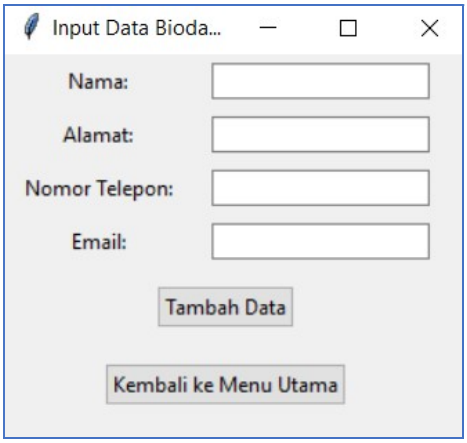
Gambar 10. Jendela Utama Program

Tampilan pada gambar 10, dihasilkan dari kode program berikut:

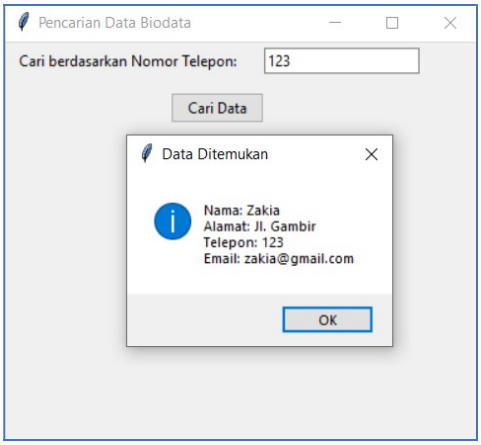
```
import tkinter as tk
from tkinter import ttk, messagebox
import sqlite3

# Fungsi untuk membuat koneksi ke
database dan membuat tabel
def create_database():
    conn =
sqlite3.connect('biodata.db')
# Membuat database biodata.db
    c = conn.cursor()
    c.execute('''CREATE TABLE IF NOT
EXISTS biodata (
                    id INTEGER
PRIMARY KEY,
                    nama TEXT,
                    alamat TEXT,
                    telepon TEXT,
                    email TEXT)''')
    conn.commit()
    conn.close()
# Fungsi untuk menambahkan biodata ke
database
def tambah_data():
    nama = entry_nama.get()
    alamat = entry_alamat.get()
    telepon = entry_telepon.get()
    email = entry_email.get()
    if nama == "" or alamat == "" or
telepon == "" or email == "":
        messagebox.showwarning("Input
Salah", "Semua field harus diisi!")
        return
    conn =
sqlite3.connect('biodata.db')
    c = conn.cursor()
    c.execute("INSERT INTO biodata
(nama, alamat, telepon, email) VALUES
(?, ?, ?, ?)", (nama, alamat, telepon,
email))
    conn.commit()
    conn.close()
    messagebox.showinfo("Sukses",
"Data berhasil ditambahkan!")
    reset_form()
# Fungsi untuk mencari biodata
berdasarkan nomor telepon
def cari_data():
    telepon = entry_cari_telepon.get()

    if telepon == "":
        messagebox.showwarning("Input
Salah", "Nomor telepon harus diisi!")
        return
    conn =
sqlite3.connect('biodata.db')
    c = conn.cursor()
    c.execute("SELECT * FROM biodata
WHERE telepon = ?", (telepon,))
    data = c.fetchone()
    conn.close()
    if data:
        # Menampilkan hasil pencarian
        messagebox.showinfo("Data
Ditemukan", f"Nama: {data[1]}\nAlamat:
{data[2]}\nTelepon: {data[3]}\nEmail:
{data[4]}")
    else:
        messagebox.showinfo("Data
Tidak Ditemukan", "Tidak ada data
dengan nomor telepon tersebut.")
# Fungsi untuk mereset form input
def reset_form():
    entry_nama.delete(0, tk.END)
    entry_alamat.delete(0, tk.END)
    entry_telepon.delete(0, tk.END)
    entry_email.delete(0, tk.END)

# Fungsi untuk membuka form pencarian
def buka_form_pencarian():
    form_menu.withdraw()  #
Menyembunyikan form menu utama
    form_pencarian.deiconify()  #
Menampilkan form pencarian
# Fungsi untuk membuka form input
data
def buka_form_input():
    form_menu.withdraw()  #
Menyembunyikan form menu utama
    form_input_data.deiconify()  #
Menampilkan form input data
```

```
# Membuat jendela form menu utama
form_menu = tk.Tk()
form_menu.title("Menu Utama")
form_menu.geometry("400x200")
# Menambahkan tombol untuk membuka
form input data
button_buka_input =
ttk.Button(form_menu, text="Form
Input Data", command=buka_form_input)
button_buka_input.pack(pady=10)
# Menambahkan tombol untuk membuka
form pencarian data
button_buka_pencarian =
ttk.Button(form_menu, text="Form
Pencarian Data",
command=buka_form_pencarian)
button_buka_pencarian.pack(pady=10)
# Membuat jendela form input data
form_input_data =
tk.Toplevel(form_menu)
form_input_data.title("Input Data
Biodata")
form_input_data.geometry("500x400")
form_input_data.withdraw()  #
Menyembunyikan form input data pada
awalnya
# Membuat form input untuk memasukkan
biodata
label_nama =
ttk.Label(form_input_data,
text="Nama:")
label_nama.grid(row=0, column=0,
padx=10, pady=5)
entry_nama =
ttk.Entry(form_input_data)
entry_nama.grid(row=0, column=1,
padx=10, pady=5)
label_alamat =
ttk.Label(form_input_data,
text="Alamat:")
label_alamat.grid(row=1, column=0,
padx=10, pady=5)
entry_alamat =
ttk.Entry(form_input_data)
entry_alamat.grid(row=1, column=1,
padx=10, pady=5)
label_telepon =
ttk.Label(form_input_data,
text="Nomor Telepon:")
label_telepon.grid(row=2, column=0,
padx=10, pady=5)
entry_telepon =
ttk.Entry(form_input_data)
entry_telepon.grid(row=2, column=1,
padx=10, pady=5)
label_email =
ttk.Label(form_input_data,
text="Email:")
label_email.grid(row=3, column=0,
padx=10, pady=5)
entry_email =
ttk.Entry(form_input_data)
entry_email.grid(row=3, column=1,
padx=10, pady=5)
# Tombol untuk menambah data
button_tambah =
ttk.Button(form_input_data,
text="Tambah Data",
command=tambah_data)
button_tambah.grid(row=4, column=0,
columnspan=2, pady=10)
# Tombol untuk kembali ke menu utama
button_buka_menu =
ttk.Button(form_input_data,
text="Kembali ke Menu Utama",
command=form_input_data.withdraw)
button_buka_menu.grid(row=5, column=0,
columnspan=2, pady=10)
# Membuat jendela form pencarian
form_pencarian =
tk.Toplevel(form_menu)
form_pencarian.title("Pencarian Data
Biodata")
form_pencarian.geometry("500x400")
form_pencarian.withdraw()  #
Menyembunyikan form pencarian pada
awalnya
# Membuat form pencarian berdasarkan
nomor telepon
label_cari_telepon =
ttk.Label(form_pencarian, text="Cari
berdasarkan Nomor Telepon:")
label_cari_telepon.grid(row=0,
column=0, padx=10, pady=5)
entry_cari_telepon =
ttk.Entry(form_pencarian)
entry_cari_telepon.grid(row=0,
column=1, padx=10, pady=5)
# Tombol untuk mencari data
button_cari =
ttk.Button(form_pencarian, text="Cari
Data", command=cari_data)
button_cari.grid(row=1, column=0,
columnspan=2, pady=10)

# Tombol untuk kembali ke menu utama
button_buka_menu_pencarian =
ttk.Button(form_pencarian,
text="Kembali ke Menu Utama",
command=form_pencarian.withdraw)
button_buka_menu_pencarian.grid(row=2,
column=0, columnspan=2, pady=10)
# Menjalankan fungsi untuk membuat
database jika belum ada
create_database()
# Menjalankan form menu utama sebagai
jendela pertama
form_menu.mainloop()
```

Pada menu utama terdapat dua pilihan yaitu form input data dan form pencarian data, seperti pada gambar 11 dan gambar 12.

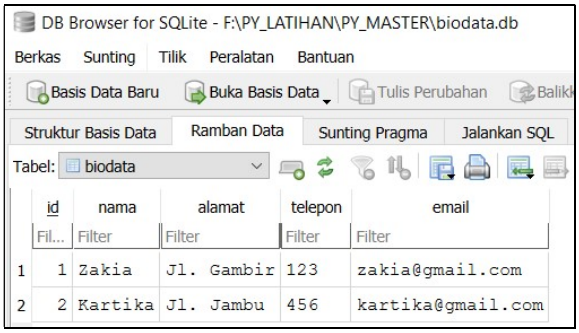


Gambar 11. Window Tambah Data



Gambar 12. Window Pencarian Data

Hasil input data, dapat dilihat juga melalui browser sqllte, seperti pada gambar 13.



Gambar 13. Browser SQLite

VI. KESIMPULAN

Pembelajaran pemrograman berbasis GUI dan database pada Python menambah pemahaman akan kemampuan Python dalam membangun aplikasi interaktif dan menyimpan data secara efisien. Dengan memanfaatkan pustaka seperti tkinter untuk antarmuka grafis dan sqlite3 untuk pengelolaan database, memungkinkan membuat aplikasi yang ramah pengguna dan mendukung kebutuhan penyimpanan data.

DAFTAR PUSTAKA

[1] Sabbrina, A. (2023). Pengenalan Konsep Dasar Dan Penggunaan Database Manajemen Sistem (DBMS). Jurnal Sains dan Teknologi (JSIT), 3(3), 224-232.

[2] K. S. &. C. Maufrais, Introduction to Programming using Python, Boston, 2010.

[3] Enterprise, Jubille, Otodidak Pemrograman Python, Jakarta: Elexmedia Komputindo, 2017

[4] Nova, S., Khotimah, N., & Wahyuningrum, M. Y. A. (2024). Pemanfaatan Chatbot Menggunakan Natural Language Processing Untuk Pembelajaran Dasar-Dasar Gui Tkinter Pada Bahasa Pemrograman Python. *Jurnal Ilmiah Teknik*, *3*(1), 58-65.

[5] Microsoft,https://code.visualstudio.com, Microsoft, 2020. [Online].