LAPORAN PERANCANGAN APLIKASI MY WORKOUT (PANTAU AKTIVITAS & KALORI OLAHRAGA)

Matakuliah:

[Pemrograman Berorientasi Objek]

Oleh:

Angely Intan Marcella

24091397047

2024B

PROGRAM STUDI D4 MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

2025

BAB I

PENDAHULUAN

1.1 Latar Belakang

Olahraga adalah kegiatan yang sangat penting untuk menjaga kesehatan dan kebugaran tubuh. Namun, banyak orang yang berolahraga tanpa membuat catatan yang teratur, seperti jenis olahraga, durasi latiha, serta jumlah kalori yang telah terbakar. hal ini mengakibatkan pengguna mengalami kesulitan untuk memantau konsistensi olahraga serta menilai hasil dari aktivitas yang telah dilakukan.

Dengan kemajuan teknologi, aplikasi My woekout berbasis python menggunakan GUI Tkinter. Aplikasi ini dirancang untuk mencatat data olahraga, menghtiung estimasi kalori, menampilkan statistik. Serta meneriapkan prinsip Pemrograman Berorientasi Objek (OOP). Pengembangan aplikasi ini juga diharapkan dapat membantu pengguna dalam memantau aktivitas olahraga sekaligus berfungsi sebagai penerapan konsep OOP dan struktur data dalam sebuah proyek nyata.

1.2 Rumusan Masalah

1. Bagaimana merancang aplikasi pengelolaan data olahraga brbasis GUI Tkinter?
2. Bagaimana cara menghitung kalori yang terbakar dari berbagai jenis olahraga?
3. Bagaimana cara menampilkan statistik dan analisis data workout secara viusal?
4. Bagaimana menerapkan konep OOP pada aplikasi My Workout?

1.3 Tujuan

Tujuan dari pengembangan aplikasi ini yaitu:

1. Membuat aplikasi pengelolaan data olahraga berbasis GUI.
2. Menerapkan prinsip-prinsip OOP (Encapsualation, Inheritance, dan Polymorpishm).

1.4 Manfaat

Bagi Pengguna:

- Memudahkan pencatatan aktivitas olahraga harian.
- Memantau progress dan target kalori.
- Menganalisis pola olahraga melalui visualisasi data.

Bagi Pengembang:

- Meningkatkan pemahaman konsep OOP.
- Belajar pengembangan aplikasi GUI dengan python.

BAB II

PERANCANGAN SISTEM

2.1 Deskripsi Singkat

My Workout adalah aplikasi berbasis GUI Tkinter yang digunakan untuk mencatat dan mengelola data aktivitas olahraga. Aplikasi ini memungkinkan pengguna memmasukkan jenis olahraga, durasi olahraga, dan intensitas latihan, kemudian menghitung estimasi kalori serta menampilkan statistik dan grafik. Aplikasi ini dikembangkan menggunakan konsep Pemrograman Berorientasi Objek (OOP) dan struktur data untuk memudahkan pengelolaan serta analisis data olahraga.

2.2 Alur Aplikasi

Alur kerja pada aplikasi My Workout ini dirancang untuk memudahkan pengguna dalam melakukan pencatatan dan pemantauan aktivitas olahraga. Berikut adalah penjelasan alur aplikasi:

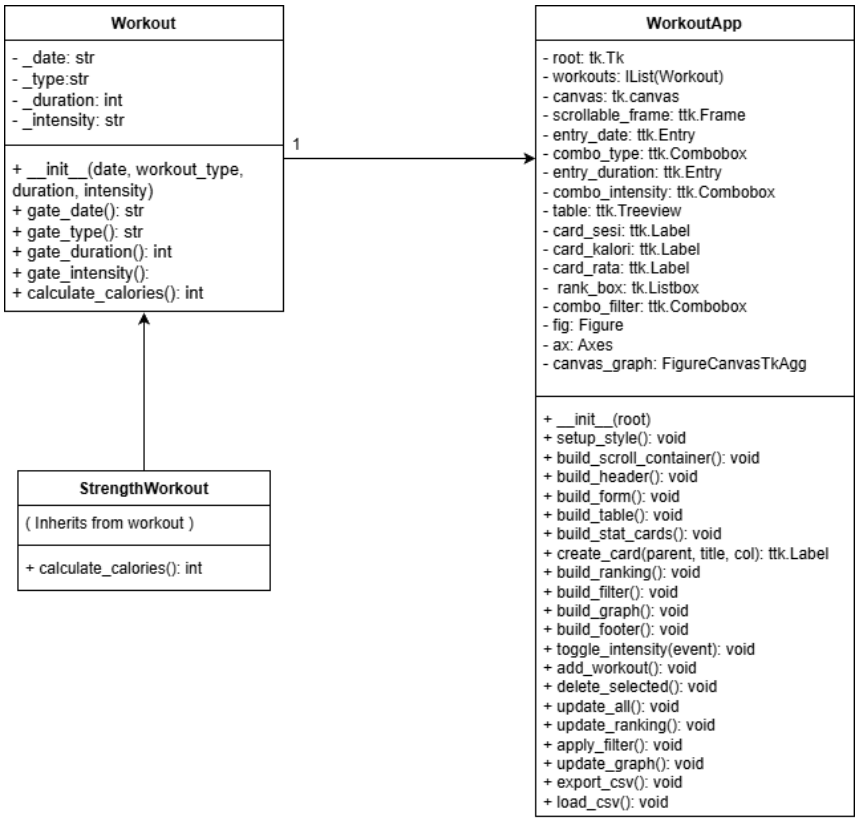
1. **Aplikasi dijalankan**
   Sistem akan menampilkan halaman utama (form input, tabel, statistik, dan grafik batang)
2. **Pengguna menginputkan data workout**
   Pengguna mengisi tanggal, jenis olahraga, durasi, dan intensitas (jika diperlukan).
3. **Sistem melakukan validasi input**
   Jika durasi tidak valid maka akan menampilkan pesan eror, lalu kembali ke form untuk mengisi ulang. Sedangkan jika valid akan melanjutkan proses.
4. **Sistem menentukan jenis objek**
   Jika jenis angkat beban akan membuat objek StrengthWorkout, jika bukan membuat objek workout.
5. **Data disimpan dan ditampilkan**
   Workout ditambahkan ke list dan tabel riwayat.
6. **Sistem memperbarui tampilan**
   Statistik (total sesi, kalori, rata-rata), ranking dan grafik otomatis akan diperbarui.
7. **Pengguna dapat melakukan aksi lanjutan**
   Filter data, hapus data, atau export ke CSV.

2.3 Diagram Kelas (class diagram)

Pada aplikasi ini memiliki beberapa kelas utama yaitu sebagai berikut:

- **Workout**: kelas induk yang menyimpan data dasar workout.
- **StrengthWorkout**: kelas turunan dari Workout untuk olahraga dengan intensitas.

- **WorkoutApp**: kelas utama yang mengelola GUI dan interaksi pengguna.



Relasi antara kelas workout dan workoutApp adalah one-to-many karena satu aplikasi dapat menyimpan banyak data workout. Dan relasi inheritance dari antara class StrengthWorkout dan Workout karna strengthworkout mewarisi dari workout.

2.3.1 Deskripsi Class

1. Class workout

Pada class ini adalah class dasar untuk merepresentasikan data workout. Class ini menyimpan data tanggal, jenis latihan, durasi, dan intensitas. Menyediakan method getter serta perhitungan kalori sebagai bentuk **encapsulation.**

- **Atribut**
  - _date: Tanggal Workout
  - _type: Jenis olahraga
  - _duration: Durasi dalam menit
  - _intensity: Intensitas workout

- **Method**
  - Get_date(), get_type(), get_duration(), get_intensity(): Getter methods
  - Calculate_calories(): menghitung kalori yang terbakar berdasarkan jenis dan durasi olahraga yang diinput.

2. Class StrengthWorkout

Class ini mmerupakan kelas turunan dari class workout. Class ini mewarisi seluruh atribut dan method dari workout serta melakukan override pada method calculate_calories() untuk perhitungan kalori yang berbeda sesuai dengan latihan kekuatan (inheritance & polymorpishm).

- **Inheritance**:
  - Mewarisi semua atribut dan method dari class workout
  - Tidak mendefinisikan ulang sonstructor dan atribut
  - Hanya meng-override method yang berbeda implementasinya

- **Method Override**:
  - **Calculate_calories()**: meghitung kalori berdasarkan intensitas (ringan/sedang/berat).

3. Class WorkoutApp

Pada class ini berfungsi sebagai pengendali utama aplikasi GUI Tkinter. Class ini mengelola antarmuka pengguna, menyimpan kumpulan objek workout dalam struktur data list, mengatur input, tabel data, statistik, grafik, dan proses tambah, hapus, filter, dan ekspor data csv.

- **Atribut**
  - Root: tk.Tk untuk window aplikasi Tkinter
  - Workouts: penyimpanan data latihan
  - Canvas: untuk area grafik
  - Table: tabel riwayat
  - Combo_type, entry_duration, combo_filter: komponen input
  - Card_sesi, card_kalori, card_rata: untuk statistik

- **Method berdasrkan kategori**
  1. Initialization & setup
     - **__init__(root)** : constructor, inisialisasi semua komponen aplikasi
     - **Setup_style()**: setup tema, warna, dan style untuk UI
     - **Build_scroll_container()**: membuat container yang dapat di-scroll.
  2. Method UI
     - **Build_header()** : membuat header dengan judul aplikasi
     - **Build_form() :** membuat form input workout
     - **Build_table()**: membuat tabel untuk riwayat workout
     - **Build_stat_cards():** membuat kartu statistik (total sesi, kalori, rata-rata)

- **Create_card(parent, title, col):** helper method untuk membuat satu kartu statistik
- **Build_ranking()**: emmbuat listbox ranking boros kalori
- **Build_filter()**: membuat dropdown dan tombol filter
- **Build_graph()**: membuat grafik batang (bar chart) menggunakan matplotlib
- **Build_footer():** membuat tombol-tombol footer (export, hapus)

3. Method logic & event Handling
   - **Toggle_intensity(event):** mengaktifkan/ menonaktifkan pilihan intensitas sesuai jenis olahraga tertentu.
   - **Add_workout():** menambah data workout baru ke dalam list dan tabel.
   - **Delete_selected()**: menghapus workout yang dipilih dari tabel
   - **Apply_filter()**: menerapkan filter periode (semua/mingguan/bulanan)
4. Update data dan tampilan
   - **Update_all:** memperbarui semua tampilan (cards, ranking, graph)
   - **Update_ranking()**: update listbox ranking berdasarkan kalori
   - Update_graph(): updategrafik bar chart kalori per jenis olahraga
5. Operasi File
   - **Save_data_to_file()**: auto save data ke CSV (tanpa notifikasi)
   - **Export_csv_manual()**: export manual ke CSV (dengan notifikasi sukses)
   - **Load_csv()**: load data dari CSV saat aplikasi dibuka
   - **On_closing()**: auto save data saat aplikasi ditutup

BAB III

IMPLEMENTASI SISTEM

3.1 Implementasi konsep OOP

- **Encapsulation**

```python
# Encapsulation
class Workout:
    def __init__(self, date, workout_type, duration, intensity="-"):
        self._date = date
        self._type = workout_type
        self._duration = duration
        self._intensity = intensity

    def get_date(self): return self._date
    def get_type(self): return self._type
    def get_duration(self): return self._duration
    def get_intensity(self): return self._intensity
```

Encapsulation diterapkan pada kelas workout dengan penggunaan atribut yangbersifat protected, seperti _date, _type, _duration, dan _intensity. Akses data dilakukan melalui metode getter untuk menjaga keamanan data.

- **Inheritance**

```python
# Inheritance
class StrengthWorkout(Workout):
    def calculate_calories(self):
        faktor = {"Ringan": 3, "Sedang": 5, "Berat": 7}
        return self.get_duration() * faktor.get(self.get_intensity(), 0)
```

Inheritance diterapkan melalui kelas StrengthWorkout yang merupakan turunan kelas Workout. Kelas ini mewarisi seluruh stribut dan method dari class induk, kemudian menambahkan atau memodifikasi perilaku tertentu. Dengan pewarisan ini, aplikasi dapat membedakan perhitungan kalori untuk jenis olahraga tertentu tanpa harus menulis ulang seluruh kode.

- **Abstraction**

```python
kalori = sum(w.calculate_calories() for w in self.workouts)
```

```python
self.table.insert("", tk.END, values=(
    w.get_date(), w.get_type(), w.get_duration(),
    w.get_intensity(), w.calculate_calories()
))
```

Konsep abstraction pada aplikasi Workout Tracker diterapkan dengan menyembunyikan detail proses perhitungan dan pengolahan data di dalam metode kelas. Aplikasi hanya berinteraksi melalui method seperti calculate_calories() dan getter tanpa mengetahui detail implementasinya. Dengan demikian, pengguna kelas dapat menggunakan fungsi utama tanpa harus memahami logika internal yang kompleks.

- **Polymorpishm**

```
#polymorphism
def calculate_calories(self):
    faktor = {
        "Lari": 6, "Jalan Santai": 3, "Bersepeda": 5, "Renang": 7,
        "Yoga": 4, "Skipping": 10, "HIIT": 12
    }
    return self._duration * faktor.get(self._type, 5)
```

```
class StrengthWorkout(Workout):
    def calculate_calories(self):
        faktor = {"Ringan": 3, "Sedang": 5, "Berat": 7}
        return self.get_duration() * faktor.get(self.get_intensity(), 0)
```

Polymorphism diterapkan melalui penggunaan method calculate_calories() yang emmiliki implementasi berbeda pada class Workout dan StrengthWorkout. Meskipun method yang dipanggil memiliki nama yang sama, hasil perhitungan kalori dapat berbeda tergantung pada jenis objek yang digunakan. Hal ini membuat kode lebih fleksibel dan mudah dikembangkan untuk menambahkan jenis workout baru.

3.2 implementasi Struktur Data

- **List**

```
class WorkoutApp:
    def __init__(self, root):
        self.root = root
        self.workouts = []
```

Pada self.workouts berfungsi sebagai penyimpanan utama yang menampung objek workout dan strengthworkout. Setiap pengguna menambahkan data workout, objek workout baru akan dimasukkan ke dalam list ini dengan menggunakan operasi append(). Dengan penggunaan list ini, pengelolaan data workout menjadi lebih efeisen dan rapi, dan ini sesuai dengan kebutuhan aplikasi berbasis GUI.

- **Tree**

```
def build_table(self):
    frame = ttk.LabelFrame(self.scrollable_frame, text="Riwayat Workout")
    frame.grid(row=1, column=1, padx=20, pady=10, sticky="nw")

    columns = ("Tanggal", "Jenis", "Durasi", "Intensitas", "Kalori")
    self.table = ttk.Treeview(frame, columns=columns, show="headings", height=10)
```

Pada aplikasi ini mengimplementasikan tree yang menggunakan komponen Trreeview dari pustaka ttk, di mana setiap data direpresentasikan sebagai sebuah node. Implementasi ini berfungsi untuk menampilkan data workout dalam bentuk tabel. Sehingga mempermudah pengguna dalam melihat, memilih, mengelola data workout secara terstruktur.

3.3 Scrollable Layout

```
def build_scroll_container(self):
    self.canvas = tk.Canvas(self.root, bg="#F4F6F8")
    scrollbar = ttk.Scrollbar(self.root, orient="vertical", command=self.canvas.yview)

    self.scrollable_frame = ttk.Frame(self.canvas)
    self.scrollable_frame.bind(
        "<Configure>",
        lambda e: self.canvas.configure(scrollregion=self.canvas.bbox("all"))
    )

    self.canvas.create_window((0, 0), window=self.scrollable_frame, anchor="nw")
    self.canvas.configure(yscrollcommand=scrollbar.set)
```

Scrollable container pada aplikasi ini diimplementasikan menggunakan kombinasi widget canvas, Frame, dan scrollbar. Canvas berfungsi sebagai wadah utama mendukung scroll, karena widget frame pada tkinter tidak menydeiakan fitur scroll secara langsung. Di dalam canvas tersebut ditempatkan sebuah scrollable_frame yang menjadi container bagi seluruh komponen antarmuka aplikasi. Event <Configure> digunakan untuk mendeteksi perubahan ukuran konten, sehingga area scroll (scrollregion) pada canvas dapat diperbarui secara otomatis berdasarkan batas keseluruhan widget. Metode create_window() digunakan untuk menanamkan frame ke dalam canvas dengan posisi awal di pojok kiri atas agar tata letak tetap konsisten. Selanjutnya, scrollbar dihubungkan dengan canvas melalui properti yscrollcommand, sehingga pengguna dapat menggulir tampilan secara vertikal untuk mengakses seluruh bagian aplikasi yang tidak muat dalam satu layar.

3.4 Form Input Data

```
self.entry_date = ttk.Entry(frame)
self.entry_date.grid(row=0, column=1)
self.entry_date.insert(0, datetime.now().strftime("%d-%m-%Y"))
```

```
self.combo_type.grid(row=1, column=1)
self.combo_type.set("Lari")
self.combo_type.bind("<<ComboboxSelected>>", self.toggle_intensity)

self.entry_duration = ttk.Entry(frame)
self.entry_duration.grid(row=2, column=1)

self.combo_intensity = ttk.Combobox(
    frame, values=["Ringan", "Sedang", "Berat"], state="disabled"
)
```

Pada kode ini implementasi form input data. Form input digunakan sebagai media interaksi utama antara pengguna dan ssitem. Pengguna dapat memasukkan tanggal workout, memilih jenis olahraga, mengisi durasi latihan, serta menentukan intentias jika diperlukan. Penggunaan combobox membantu membatasi input agar sesuai dengan data yang telah ditentukan

3.5 validasi Input

```
except ValueError:
    messagebox.showerror("Error", "Durasi harus angka")
```

Validasi input diterapkan untuk memastikan bahwa data yang dimasukkan sesuai dengan format yang diharapkan. Pada bagian ini, durasi workout divalidasi agar berupa angka. Jika pengguna memasukkan data yang tidak valid, sistem akan menampilkan pesan kesalahan sehingga dapat mencegah kesalahan perhitungan dan kerusakan data.

3.6 Filter Data (mingguan dan bulanan)

```
if periode == "Mingguan":
    batas_waktu = now - datetime.timedelta(days=7)
elif periode == "Bulanan":
    batas_waktu = datetime.datetime(now.year, now.month, 1)
```

Fitur filter memungkinkan pengguna melihat data workout berdasarkan rentang waktu tertentu, yaitu mingguan dan bulanan. Sistem akan membandingkan tanggal workout dengan batas waktu yang ditentukan, sehingga hanya data yang sesuai periode yang ditampilkan di tabel.

3.6 Perhitungan Statistik

```
def update_all(self):
    total = len(self.workouts)
    kalori = sum(w.calculate_calories() for w in self.workouts)
    rata = kalori // total if total else 0
```

Bagian ini digunakan untuk menghitung statistik dasar aplikasi, yaitu total sesi workout, total kalori yang terbakar, dan rata-rata kalori per sesi. Hasil perhitungan ditampilkan secara real-time pada kartu statistik sehingga pengguna dapat langsung melihat ringkasan aktivitasnya.

3.7 Ranking Olahraga

```
for i, (j, k) in enumerate(sorted(data.items(), key=lambda x: x[1], reverse=True), 1):
    self.rank_box.insert(tk.END, f"{i}. {j} - {k} kalori")
```

Ranking olahraga dibuat untuk menunjukkan jenis olahraga yang paling banyak membakar kalori. Data diurutkan secara menurun berdasarkan total kalori, kemudian ditampilkan dalam bentuk daftar berurutan. Fitur ini membantu pengguna memahami efektivitas tiap jenis olahraga

3.8 Grafik Batang

```
data:
 self.ax.bar(data.keys(), data.values(), color="#3B84F8")
 self.ax.set_title("Total Kalori per Olahraga", fontsize=10)
 self.ax.tick_params(axis='x', rotation=30, labelsize=8)
 self.ax.tick_params(axis='y', labelsize=8)
 self.fig.tight_layout()
```

Pada kode tersebut untuk pembuatan grafik batang. Grafik batang digunakan untuk menampilkan total kalori per jenis olahraga. Grafik akan diperbarui secara dinamis menggunakan matplotlib. Warna dan label dengan menggunakan warna #3B84F8.

3.9 Penyimpanan Data (CSV)

```
def save_data_to_file(self):
    with open("workout_data.csv", "w", newline="") as f:
        writer = csv.writer(f)
        writer.writerow(["Tanggal", "Jenis", "Durasi", "Intensitas", "Kalori"])
        for w in self.workouts:
            writer.writerow([
                w.get_date(), w.get_type(),
                w.get_duration(), w.get_intensity(),
                w.calculate_calories()
            ])

def load_csv(self):
    if not os.path.exists("workout_data.csv"):
        return
    with open("workout_data.csv", "r", newline="") as f:
        reader = csv.reader(f)
        try:
```

Data workout disimpan ke dalam file CSV agar tetap tersedia meskipun aplikasi ditutup. Format CSV dipilih karena sederhana, ringan, dan mudah dibaca kembali oleh sistem

3.10 Auto save saat aplikasi ditutup

```
self.load_csv()
self.root.protocol("WM_DELETE_WINDOW", self.on_closing)
```

```
def on_closing(self):
    try:
        self.save_data_to_file()
    except Exception as e:
        messagebox.showerror("Error Save", f"Gagal menyimpan data: {e}")
    finally:
        self.root.destroy()
```

Fitur auto-save memastikan seluruh data disimpan secara otomatis ketika pengguna menutup aplikasi. Mekanisme ini mencegah kehilangan data akibat kelalaian pengguna dalam menyimpan secara manual.

3.11 Fitur dan Cara Penggunaan Aplikasi

3.11.1 Fitur Aplikasi

1. **Fitur Tambah Workout**

   Fitur ini digunakan untuk memasukkan data aktivitas olahraga yang dilakukan pengguna. Data yang dapat diinput meliputi tanggal latihan, jenis olahraga, durasi, intensitas, dan catatan tambahan. Sistem akan menghitung jumlah kalori secara otomatis berdasarkan jenis olahraga dan durasi yang diinput.

2. **Fitur Riwayat Workout**

   Fitur ini menampilkan seluruh data workout yang telah disimpan dalam bentuk tabel. Pengguna dapat melihat detail latihan, melakukan pencarian data, menghapus data tertentu, serta mengekspor data latihan ke dalam file CSV. Tabel riwayat dilengkapi dengan scrollbar untuk memudahkan navigasi saat data banyak.

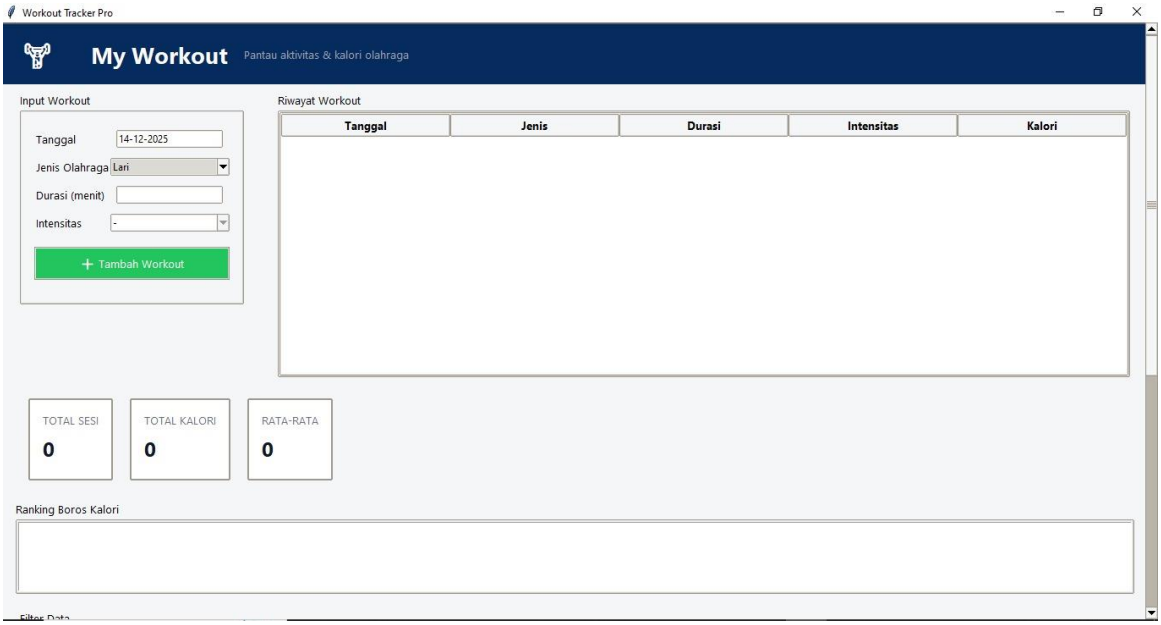
3. **Fitur Statistik Workout**

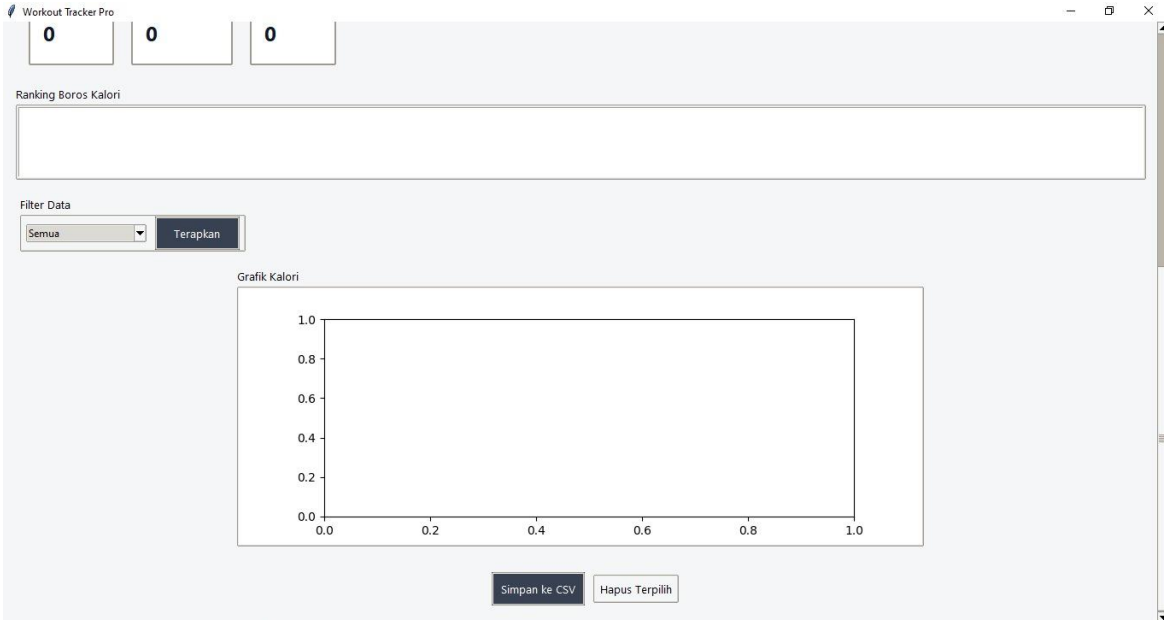
   Fitur statistik menyajikan ringkasan data olahraga pengguna, meliputi total jumlah latihan, total durasi latihan, total kalori terbakar, dan rata-rata durasi latihan. Selain itu, ditampilkan grafik batang yang menggambarkan jumlah kalori terbakar dalam 30 hari terakhir.

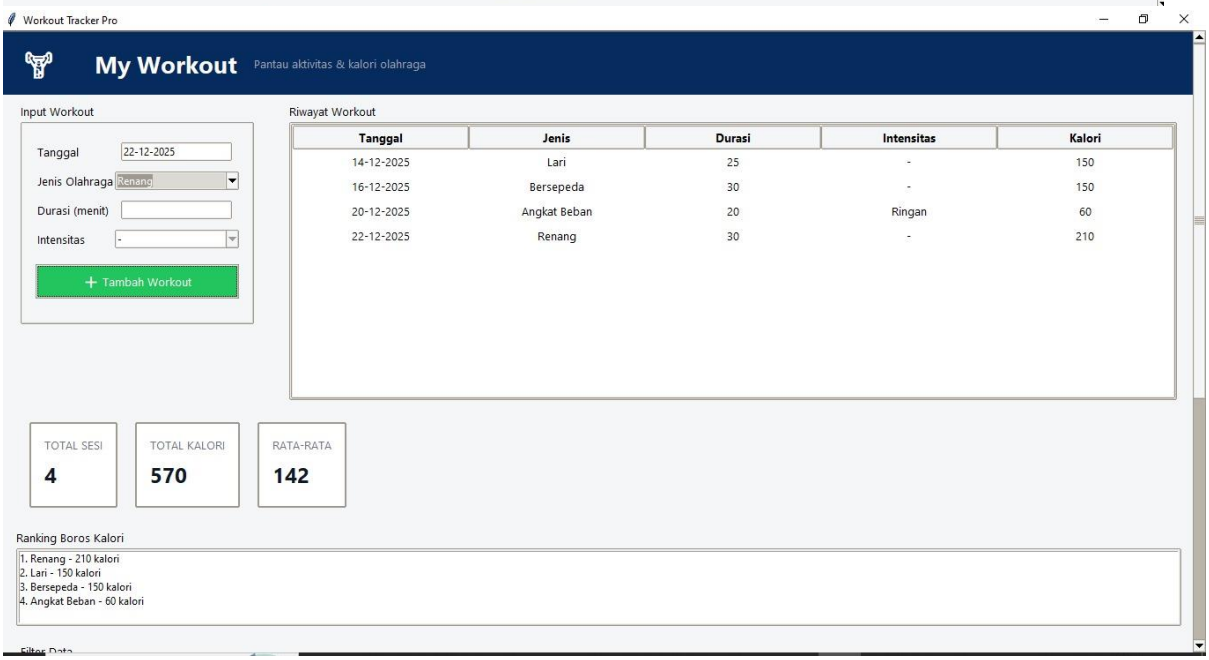
3.11.2 Cara Penggunaan Aplikasi

- Jalankan aplikasi My Workout.
- Kemudian isi data tanggal, jenis olahraga, durasi, intensitas, pada halaman form input.
- Pastikan durasi diisi dengan benar agar sistem dapat menghitung kalori secara otomatis.
- kemudian klik tombol tambah workout. Lalu data akan tersimpan.
- Data terlihat di tabel riwayat workout. Setelah penginputan
- Untuk melihat statistik dan ekspor data CSV cukup scroll ke bawah, akan tampak tombol ekspor csv dan grafik perkembangan aktivitas olahraga.

3.12 Screnshot Aplikasi




Workout Tracker Pro
0
0
0
Ranking Boros Kalori
Filter Data
Semua
Terapkan
Grafik Kalori
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Simpan ke CSV
Hapus Terpilih


Workout Tracker Pro
My Workout
Pantau aktivitas & kalori olahraga
Input Workout
Tanggal
22-12-2025
Jenis Olahraga
Renang
Durasi (menit)
Intensitas
-
+ Tambah Workout
Riwayat Workout
Tanggal
Jenis
Durasi
Intensitas
Kalori
14-12-2025
Lari
25
-
150
16-12-2025
Bersepeda
30
-
150
20-12-2025
Angkat Beban
20
Ringan
60
22-12-2025
Renang
30
-
210
TOTAL SESI
4
TOTAL KALORI
570
RATA-RATA
142
Ranking Boros Kalori
1. Renang - 210 kalori
2. Lari - 150 kalori
3. Bersepeda - 150 kalori
4. Angkat Beban - 60 kalori

BAB IV

PENUTUP

4.1 Kesimpulan

Berdasarkan hasil perancangan dan implementasi yang telah dilakukan, dapat disimpulkan bahwa aplikasi *My Workout* berhasil dikembangkan sebagai aplikasi pengelolaan data olahraga berbasis GUI menggunakan Tkinter. Aplikasi ini mampu mencatat aktivitas olahraga pengguna, menghitung estimasi kalori terbakar, serta menyajikan informasi dalam bentuk tabel, statistik, dan grafik batang. Penerapan konsep Pemrograman Berorientasi Objek (OOP) seperti encapsulation, inheritance, dan polymorphism membantu dalam penyusunan kode agar lebih terstruktur, mudah dipahami, dan mudah dikembangkan. Selain itu, penggunaan struktur data list memungkinkan pengelolaan data workout dilakukan secara efisien. Secara

keseluruhan, aplikasi ini telah memenuhi tujuan yang ditetapkan dan dapat digunakan sebagai alat bantu pencatatan aktivitas olahraga secara sederhana dan efektif.

4.2 Tantangan Selama Pengembangan

Selama proses pengembangan aplikasi, terdapat beberapa tantangan yang dihadapi. Tantangan utama adalah pengaturan tampilan antarmuka pengguna yang cukup kompleks karena aplikasi memiliki banyak komponen dalam satu halaman, seperti form input, tabel data, statistik, dan grafik. Hal ini diatasi dengan menerapkan konsep scrollable frame dan pemisahan komponen ke dalam beberapa container agar tampilan tetap rapi dan mudah digunakan. Tantangan lainnya adalah pengelolaan data workout agar dapat diperbarui secara otomatis ketika data ditambah atau dihapus. Solusi yang digunakan adalah dengan membuat metode khusus untuk memperbarui seluruh tampilan, seperti tabel, statistik, dan grafik, setiap kali terjadi perubahan data. Selain itu, integrasi grafik Matplotlib ke dalam Tkinter juga menjadi tantangan tersendiri, yang diselesaikan dengan menggunakan canvas khusus agar grafik dapat ditampilkan dengan baik di dalam aplikasi.

4.3 Saran Pengembangan Selanjutnya

Untuk pengembangan selanjutnya, aplikasi *My Workout* masih dapat ditingkatkan dengan menambahkan beberapa fitur tambahan. Penggunaan database seperti SQLite dapat diterapkan untuk menyimpan data secara lebih permanen dan terstruktur dibandingkan file CSV. Selain itu, aplikasi dapat dikembangkan dengan sistem akun pengguna sehingga data latihan dapat dikelola secara individual. Pengembangan visualisasi data juga dapat ditingkatkan dengan menambahkan grafik yang lebih interaktif serta fitur analisis mingguan atau bulanan. Dengan pengembangan lebih lanjut, aplikasi ini diharapkan dapat memberikan manfaat yang lebih luas bagi pengguna dalam memantau dan meningkatkan aktivitas olahraga secara berkelanjutan.