TUGAS PEMROGRAMAN BERORIENTASI OBJEK PRAKTIK

Disusun oleh:

YOGI PRANOTO

5230411284

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS & TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

YOGYAKARTA

2024

1. Latar Belakang Studi Kasus

Pada dunia kerja maupun pendidikan, tugas atau pekerjaan yang harus diselesaikan sering kali menjadi hal yang menumpuk dan membingungkan jika tidak dicatat dengan baik. Banyak orang mengalami kesulitan dalam mengelola tugas-tugas mereka, sehingga terjadi penundaan atau bahkan lupa dengan beberapa tugas yang harus dikerjakan. Untuk itu, aplikasi "To-Do List" atau daftar tugas diperlukan untuk memudahkan dalam mencatat, mengelola, dan memantau tugas-tugas yang harus diselesaikan.

Aplikasi ini dirancang untuk membantu penggunanya dalam membuat, menandai, menghapus, dan menampilkan daftar tugas yang harus diselesaikan. Dengan tampilan yang sederhana dan mudah digunakan, pengguna dapat dengan cepat menambah tugas, mengatur tenggat waktu, prioritas, dan mata kuliah terkait dengan tugas tersebut.

2. Penjelasan Program

Aplikasi To-Do List ini dibangun menggunakan Python dengan library Tkinter untuk antarmuka grafis (GUI) dan ttk untuk elemen-elemen seperti combobox dan treeview. Selain itu, aplikasi ini menggunakan re untuk validasi input tanggal dengan format yang tepat (dd-mm-yyyy).

- **Fitur Utama Aplikasi:**

1. **Menambahkan Tugas:** Pengguna dapat menambahkan tugas baru dengan memasukkan nama tugas, tenggat waktu, prioritas, dan mata kuliah. Setelah tugas ditambahkan, aplikasi akan menampilkan tugas tersebut di daftar tugas yang tersedia.
2. **Validasi Input:**
    - Validasi format tanggal untuk memastikan tenggat waktu yang dimasukkan sesuai dengan format yang benar (dd-mm-yyyy).
    - Validasi prioritas untuk memastikan tugas diberi prioritas yang valid (Tinggi, Sedang, atau Rendah).
3. **Menandai Tugas yang Selesai:** Aplikasi ini dilengkapi dengan tombol "Tandai Selesai" yang memungkinkan pengguna untuk menandai tugas yang telah diselesaikan. Setelah tugas ditandai, statusnya akan diperbarui menjadi "Selesai" pada kolom status di daftar tugas.
4. **Hapus Tugas:** Pengguna dapat memilih tugas yang ingin dihapus. Ada juga tombol untuk menghapus semua tugas yang ada dalam daftar.

- **Fitur Tampilan:**
    - **Input Nama Tugas:** Pengguna dapat memasukkan nama tugas yang harus diselesaikan.
    - **Input Tenggat Waktu:** Pengguna dapat mengisi tenggat waktu untuk setiap tugas dengan format dd-mm-yyyy.
    - **Input Prioritas:** Pengguna dapat memilih prioritas tugas (Tinggi, Sedang, atau Rendah).
    - **Input Mata Kuliah:** Pengguna dapat menambahkan nama mata kuliah terkait dengan tugas tersebut.

- **Daftar Tugas:** Semua tugas yang ditambahkan akan ditampilkan dalam sebuah tabel, yang menunjukkan nama tugas, tenggat waktu, prioritas, mata kuliah, dan status (apakah selesai atau belum selesai).
- **Tombol Hapus:** Pengguna dapat menghapus tugas tertentu atau menghapus semua tugas.
- **Tombol Tandai Selesai:** Pengguna dapat menandai tugas yang sudah selesai, yang akan mengubah status tugas menjadi "Selesai" di daftar.

➢ **Flow Aplikasi:**

- Pengguna membuka aplikasi dan memasukkan tugas baru melalui form input.
- Tugas yang dimasukkan akan muncul di daftar tugas dengan status "Belum Selesai".
- Pengguna dapat menandai tugas sebagai "Selesai" dengan memilih tugas dan menekan tombol "Tandai Selesai".
- Pengguna dapat menghapus tugas tertentu atau semua tugas yang ada dalam daftar.

3. CODE :

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


class ToDoApp:
    def __init__(self, root):
        self.root = root
        self.root.title("To-Do List")
        self.root.geometry("600x450")


        self.widget_create()


    def widget_create(self):


        title_label = tk.Label(self.root, text="To-Do List - Catat Tugas
Anda", font=("Arial", 18))
        title_label.pack(pady=10)


        input_frame = tk.Frame(self.root)
```

```
        input_frame.pack(pady=10)


        tk.Label(input_frame, text="Nama Tugas: ").grid(row=0, column=0,
padx=5, pady=5)
        self.task_name_entry = tk.Entry(input_frame, width=30)
        self.task_name_entry.grid(row=0, column=1, padx=5, pady=5)


        tk.Label(input_frame, text="Tenggat Waktu (dd-mm-yyyy):
").grid(row=1, column=0, padx=5, pady=5)
        self.task_deadline_entry = tk.Entry(input_frame, width=30)
        self.task_deadline_entry.grid(row=1, column=1, padx=5, pady=5)


        tk.Label(input_frame, text="Prioritas: ").grid(row=2, column=0,
padx=5, pady=5)
        self.task_priority_combobox = ttk.Combobox(input_frame,
values=["Tinggi", "Sedang", "Rendah"], width=27, state="readonly")
        self.task_priority_combobox.grid(row=2, column=1, padx=5, pady=5)


        tk.Label(input_frame, text="Mata Kuliah: ").grid(row=3, column=0,
padx=5, pady=5)
        self.task_subject_entry = tk.Entry(input_frame, width=30)
        self.task_subject_entry.grid(row=3, column=1, padx=5, pady=5)


        add_button = tk.Button(self.root, text="Tambah Tugas",
command=self.add_task, bg="lightblue")
        add_button.pack(pady=10)


        self.task_treeview = ttk.Treeview(self.root, columns=("Nama Tugas",
"Tenggat Waktu", "Prioritas", "Mata Kuliah", "Status"), show="headings")
        self.task_treeview.heading("Nama Tugas", text="Nama Tugas")
        self.task_treeview.heading("Tenggat Waktu", text="Tenggat Waktu")
        self.task_treeview.heading("Prioritas", text="Prioritas")
        self.task_treeview.heading("Mata Kuliah", text="Mata Kuliah")
        self.task_treeview.heading("Status", text="Status")
        self.task_treeview.column("Nama Tugas", width=200)
        self.task_treeview.column("Tenggat Waktu", width=150)
        self.task_treeview.column("Prioritas", width=100)
```

```
        self.task_treeview.column("Mata Kuliah", width=100)
        self.task_treeview.column("Status", width=100)
        self.task_treeview.pack(pady=10)

        delete_frame = tk.Frame(self.root)
        delete_frame.pack(pady=5)

        delete_one_button = tk.Button(delete_frame, text="Hapus Tugas
Terpilih", command=self.delete_task, bg="salmon")
        delete_one_button.grid(row=0, column=0, padx=5)

        delete_all_button = tk.Button(delete_frame, text="Hapus Semua
Tugas", command=self.delete_all_tasks, bg="red", fg="white")
        delete_all_button.grid(row=0, column=1, padx=5)

        mark_done_button = tk.Button(delete_frame, text="Tandai Selesai",
command=self.mark_task_done, bg="lightgreen")
        mark_done_button.grid(row=0, column=2, padx=5)

    def add_task(self):
        """Menambahkan tugas ke dalam daftar"""
        task_name = self.task_name_entry.get()
        task_deadline = self.task_deadline_entry.get()  # Tenggat waktu
dari input Entry
        task_priority = self.task_priority_combobox.get()
        task_subject = self.task_subject_entry.get()

        if task_name and task_priority and task_deadline and task_subject:
            if task_priority not in ["Tinggi", "Sedang", "Rendah"]:
                messagebox.showerror("Error", "Pilih prioritas dari opsi
yang tersedia!")
                return

            if not self.is_valid_date(task_deadline):
                messagebox.showerror("Error", "Format tenggat waktu harus
dd-mm-yyyy!")
                return
```

```
            self.task_treeview.insert("", "end", values=(task_name,
task_deadline, task_priority, task_subject, "Belum Selesai"))
            self.task_name_entry.delete(0, tk.END)
            self.task_deadline_entry.delete(0, tk.END)  # Reset tenggat
waktu
            self.task_priority_combobox.set("")  # Reset prioritas
            self.task_subject_entry.delete(0, tk.END)  # Reset mata kuliah
        else:
            messagebox.showerror("Error", "Semua field harus diisi!")

    def delete_task(self):
        selected_item = self.task_treeview.selection()
        if selected_item:
            for item in selected_item:
                self.task_treeview.delete(item)
        else:
            messagebox.showwarning("Peringatan", "Pilih tugas yang ingin
dihapus!")

    def delete_all_tasks(self):
        if messagebox.askyesno("Konfirmasi", "Anda yakin ingin menghapus
semua tugas?"):
            for item in self.task_treeview.get_children():
                self.task_treeview.delete(item)

    def is_valid_date(self, date_str):
        date_pattern = r"^\d{2}-\d{2}-\d{4}$"  # Format dd-mm-yyyy
        return re.match(date_pattern, date_str) is not None

    def mark_task_done(self):
        selected_item = self.task_treeview.selection()
        if selected_item:
            for item in selected_item:
                self.task_treeview.item(item, values=(
                    self.task_treeview.item(item, "values")[0],
```

```
                    self.task_treeview.item(item, "values")[1],
                    self.task_treeview.item(item, "values")[2],
                    self.task_treeview.item(item, "values")[3],
                    "Selesai"  # Menandai status tugas
                ))
        else:
            messagebox.showwarning("Peringatan", "Pilih tugas yang ingin ditandai sebagai selesai!")

if __name__ == '__main__':
    root = tk.Tk()
    app = ToDoApp(root)
    root.mainloop()
```

4. Activity Diagram