LAPORAN TUGAS PBO

“APLIKASI PENITIPAN KUCING”

Disusun Oleh:

Muhammad Ilham Yunanto

5230411277

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS & TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2024

1. **Latar Belakang**

Dalam kehidupan modern, banyak pemilik hewan peliharaan, khususnya kucing, menghadapi kesulitan saat mereka perlu bepergian atau sibuk bekerja. Hal ini menciptakan kebutuhan akan jasa penitipan kucing. Namun, sering kali pengelolaan data pelanggan, informasi kucing, serta periode penitipan dilakukan secara manual, yang dapat menyebabkan ketidakteraturan. Oleh karena itu, aplikasi penitipan kucing berbasis *desktop* ini dirancang untuk membantu pengelola jasa penitipan dalam menyimpan data pelanggan dengan efisien.

2. **Pembahasan Code**

   a. **Import**

```
import tkinter as tk
from tkinter import ttk, messagebox
from tkcalendar import DateEntry  # Pemilih tanggal
from datetime import datetime
```

tkinter: Pustaka bawaan Python untuk membuat antarmuka grafis (GUI).

- **tk**: Digunakan untuk elemen dasar GUI seperti *window*, *label*, *button*, dll.
- **ttk**: Modul *Themed Tkinter* untuk membuat elemen GUI dengan tampilan lebih modern, seperti *combobox* dan *treeview*.
- messagebox: Digunakan untuk menampilkan kotak dialog seperti notifikasi, pesan kesalahan, atau konfirmasi.

tkcalendar.DateEntry: Komponen tambahan untuk memilih tanggal dengan tampilan kalender.

datetime.datetime: Digunakan untuk memproses data tanggal, seperti menghitung selisih hari antara dua tanggal atau mendapatkan tanggal saat ini.

   b. **Class CatCareApp**

```
class CatCareApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Aplikasi Penitipan Kucing")
        self.root.geometry("850x650")
        self.root.configure(bg="cornflowerblue")  # Warna latar
belakang
        self.daily_rate = 20000  # Tarif harian Rp 20.000
        self.create_widgets()
```

- **__init__**: Fungsi khusus Python yang secara otomatis dijalankan saat sebuah objek dari kelas ini dibuat.

- **root:** Parameter yang merepresentasikan jendela utama (main window) aplikasi, biasanya berupa objek tk.Tk dari pustaka tkinter.
- **self.root**: Menyimpan referensi ke jendela utama sehingga dapat digunakan di seluruh kelas.

- **title("Aplikasi Penitipan Kucing")**: Memberikan judul pada jendela utama aplikasi, yang akan muncul di bilah judul (*title bar*).

- **geometry("850x650")**: Mengatur ukuran awal jendela utama menjadi lebar **850 piksel** dan tinggi **650 piksel**.

- **configure(bg="cornflowerblue")**: Mengubah warna latar belakang (*background color*) jendela menjadi biru muda (cornflowerblue).

- **self.daily_rate**: Variabel untuk menyimpan tarif harian penitipan kucing, yaitu **Rp 20.000** per hari.

c. def create_widgets(self):

```
def create_widgets(self):
        # Title
        title_label = tk.Label(
            self.root, text="Aplikasi Penitipan Kucing", font=("Arial",
24, "bold"), bg="cornflowerblue", fg="lavender"
        )
        title_label.pack(pady=20)

        # Frame Input
        input_frame = tk.Frame(self.root, bg="lightsteelblue", bd=2,
relief="groove")
        input_frame.pack(pady=10, padx=20, fill="x")

        # Nama Pemilik
        owner_label = tk.Label(input_frame, text="Nama Pemilik:",
font=("Arial", 12), bg="lightsteelblue")
        owner_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")
        self.owner_entry = tk.Entry(input_frame, width=30, font=("Arial",
12))
        self.owner_entry.grid(row=0, column=1, padx=10, pady=10)

        # Nama Kucing
        cat_name_label = tk.Label(input_frame, text="Nama Kucing:",
font=("Arial", 12), bg="lightsteelblue")
```

```
        cat_name_label.grid(row=1, column=0, padx=10, pady=10, sticky="w")
        self.cat_name_entry = tk.Entry(input_frame, width=30,
font=("Arial", 12))
        self.cat_name_entry.grid(row=1, column=1, padx=10, pady=10)

        # Jenis Kucing
        breed_label = tk.Label(input_frame, text="Jenis Kucing:",
font=("Arial", 12), bg="lightsteelblue")
        breed_label.grid(row=2, column=0, padx=10, pady=10, sticky="w")
        self.breed_var = tk.StringVar(value="")
        breed_options = ["Persia", "Anggora", "Maine Coon", "Bengal",
"Spinx", "Lainnya"]
        self.breed_combobox = ttk.Combobox(input_frame,
values=breed_options, textvariable=self.breed_var, state="readonly",
font=("Arial", 12), width=28)
        self.breed_combobox.grid(row=2, column=1, padx=10, pady=10)

        # Tanggal Penitipan
        start_date_label = tk.Label(input_frame, text="Tanggal
Dititipkan:", font=("Arial", 12), bg="lightsteelblue")
        start_date_label.grid(row=3, column=0, padx=10, pady=10,
sticky="w")
        self.start_date_entry = DateEntry(input_frame, width=12,
font=("Arial", 12), background="blue", foreground="white",
date_pattern="dd-mm-yyyy")
        self.start_date_entry.grid(row=3, column=1, padx=10, pady=10,
sticky="w")

        # Tanggal Pengambilan
        end_date_label = tk.Label(input_frame, text="Tanggal Diambil:",
font=("Arial", 12), bg="lightsteelblue")
        end_date_label.grid(row=4, column=0, padx=10, pady=10, sticky="w")
        self.end_date_entry = DateEntry(input_frame, width=12,
font=("Arial", 12), background="blue", foreground="white",
date_pattern="dd-mm-yyyy")
        self.end_date_entry.grid(row=4, column=1, padx=10, pady=10,
sticky="w")

        # Tombol Tambah Data
        add_button = tk.Button(
            self.root, text="Tambah Data", command=self.add_data,
font=("Arial", 12, "bold"), bg="limegreen", fg="white"
        )
        add_button.pack(pady=10)
```

```
        # Tabel Data Penitipan
        table_frame = tk.Frame(self.root)
        table_frame.pack(pady=10, padx=20, fill="both", expand=True,)

        style = ttk.Style()
        style.configure("Treeview", font=("Arial", 12), rowheight=25)
        style.configure("Treeview.Heading", font=("Arial", 12, "bold"),
background="lightsteelblue", foreground="black")

        self.cat_table = ttk.Treeview(
            table_frame,
            columns=("Pemilik", "Nama Kucing", "Jenis", "Tanggal Mulai",
"Tanggal Akhir", "Total Biaya"),
            show="headings",
        )
        self.cat_table.heading("Pemilik", text="Nama Pemilik")
        self.cat_table.heading("Nama Kucing", text="Nama Kucing")
        self.cat_table.heading("Jenis", text="Jenis Kucing")
        self.cat_table.heading("Tanggal Mulai", text="Tanggal Dititipkan")
        self.cat_table.heading("Tanggal Akhir", text="Tanggal Diambil")
        self.cat_table.heading("Total Biaya", text="Total Biaya (Rp)")

        self.cat_table.column("Pemilik", width=150, anchor="center")
        self.cat_table.column("Nama Kucing", width=150, anchor="center")
        self.cat_table.column("Jenis", width=100, anchor="center")
        self.cat_table.column("Tanggal Mulai", width=100, anchor="center")
        self.cat_table.column("Tanggal Akhir", width=100, anchor="center")
        self.cat_table.column("Total Biaya", width=120, anchor="center")
        self.cat_table.pack(fill="both", expand=True)

        # Tombol Hapus
        button_frame = tk.Frame(self.root, bg="cornflowerblue")
        button_frame.pack(pady=10)

        clear_button = tk.Button(
            button_frame, text="Hapus Semua Data",
command=self.clear_data, font=("Arial", 12, "bold"), bg="red", fg="white"
        )
        clear_button.grid(row=0, column=0, padx=10)
```

Create Widget di dalamnya digunakan untuk mengatur tata letak dari aplikasi yang telah kita buat termasuk mengatur warna bacground, warna font, jenis font dll.

d. **Def add_data**

```
def add_data(self):
        owner = self.owner_entry.get()#mengambil data
        cat_name = self.cat_name_entry.get()
        breed = self.breed_var.get()
        start_date = self.start_date_entry.get_date()
        end_date = self.end_date_entry.get_date()

        if owner and cat_name and breed:
            try:
                # Hitung selisih hari
                duration = (end_date - start_date).days
                if duration < 0:
                    raise ValueError("Tanggal Diambil harus setelah
Tanggal Dititipkan.")

                # Hitung total biaya
                total_cost = duration * self.daily_rate

                # Tambahkan data ke tabel
                self.cat_table.insert("", "end", values=(owner, cat_name,
breed, start_date.strftime("%d-%m-%Y"), end_date.strftime("%d-%m-%Y"),
f"{total_cost:,}"))
                self.clear_inputs()
            except Exception as e:
                messagebox.showerror("Error", str(e))
        else:
            messagebox.showerror("Error", "Pastikan semua data diisi
dengan benar!")
```

- Data input diambil dari komponen GUI (entry box atau combobox) menggunakan metode **.get().**
- **get_date()** digunakan untuk mendapatkan data dari elemen **DateEntry** sebagai objek datetime

- **(end_date - start_date).days**: Menghitung selisih antara tanggal diambil dan tanggal dititipkan dalam satuan hari.
- Jika durasi bernilai negatif (tanggal diambil lebih awal dari tanggal dititipkan), program akan **mengangkat error** (raise ValueError).
- Menghitung total biaya penitipan dengan mengalikan durasi penitipan (`duration`) dengan tarif harian **(`self.daily_rate`).**
- **Self.cat_table.insert():** digunakan untuk menambah baris baru ke tabel

- **strftime("%d-%m-%Y")**: Memformat tanggal ke dalam format DD-MM-YYYY.
- **f"{total_cost:,}"**: Memformat angka total biaya dengan tanda koma sebagai pemisah ribuan.

e. Clear

```
def clear_inputs(self):
        self.owner_entry.delete(0, tk.END)
        self.cat_name_entry.delete(0, tk.END)
        self.breed_var.set("")
        self.start_date_entry.set_date(datetime.now())
        self.end_date_entry.set_date(datetime.now())

    def clear_data(self):
        confirm = messagebox.askquestion("Konfirmasi", "Apakah Anda yakin
ingin menghapus semua data?")
        if confirm == "yes":
            for item in self.cat_table.get_children():
                self.cat_table.delete(item)
```

Digunakan untuk menghapus data

3. Hasil

4. **Implementasi**

   Aplikasi ini dapat diimplementasikan di jasa penitipan kucing dengan langkah berikut:

   1. Penggunaan Input Data: Pemilik jasa dapat mengisi informasi pelanggan, seperti nama pemilik, nama kucing, jenis kucing, dan tanggal penitipan.
   2. Penyimpanan Data Otomatis: Semua data pelanggan langsung tersimpan di tabel yang terintegrasi.
   3. Perhitungan Biaya: Total biaya penitipan dihitung secara otomatis berdasarkan durasi penitipan dan tarif harian.
   4. Manajemen Data: Data yang tidak relevan atau usang dapat dihapus dengan mudah menggunakan tombol "Hapus Semua Data".

5. **Kesimpulan**

Aplikasi ini memberikan solusi yang efisien untuk pengelolaan data penitipan kucing. Dengan antarmuka yang sederhana dan fungsi-fungsi utama seperti pencatatan data, perhitungan otomatis, dan manajemen tabel, aplikasi ini dapat menjadi alat yang andal untuk membantu bisnis jasa penitipan hewan peliharaan.