LAPORAN PRAKTIKUM

PEMROGRAMAN BERBASIS OBJEK

" MANAJEMEN RENTAL PS"

Disusun Oleh:

Nama: Zidan Amikul Afham

NPM: 5230411330

Sarjana Informatika

Universitas Teknologi Yogyakarta

1. Pendahuluan

Laporan ini menjelaskan implementasi sebuah sistem rental PlayStation menggunakan bahasa pemrograman Python dan library Tkinter. Sistem ini memiliki fitur seperti input data pelanggan, pemilihan jenis PlayStation, kelas, durasi, dan room, serta menampilkan riwayat pesanan.

2. Deskripsi Program

Program manajemen rental PS ini dibangun dengan menggunakan bahasa pemrograman Python, yang memiliki banyak fitur yang mendukung pemrograman berbasis objek. Program ini dirancang dengan beberapa fitur utama:

1. Input Data Pelanggan: Pengguna dapat memasukkan informasi dasar pelanggan, seperti nama, kelas PS yang disewa, jenis PS, serta durasi sewa yang diinginkan.

2. Perhitungan Harga Otomatis: Berdasarkan data yang dimasukkan, program secara otomatis menghitung harga sewa berdasarkan jenis PS dan durasi yang dipilih.

3. Tabel Riwayat Pesanan: Program menampilkan tabel riwayat pesanan, yang mencakup nama pelanggan, kelas PS, jenis PS, durasi, dan total harga.

4. Fitur Pembersihan Tabel dan Formulir Input: Program menyediakan tombol untuk membersihkan data input dan tabel riwayat pesanan, memberikan kemudahan bagi pengguna dalam melakukan pemesanan baru.

3. Kode Program

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

class PSRentalAppWithRoom:
    def __init__(self, root):
        self.root = root
        self.root.title("Sistem Rental PS")
        self.root.geometry("400x600")
        self.root.configure(bg="#17153B")

        self.orders = []
        self.price_data = {
            "PS 3": {"Reguler": 5000, "VIP": 0, "VVIP": 0},
            "PS 4": {"Reguler": 10000, "VIP": 15000, "VVIP": 0},
            "PS 5": {"Reguler": 20000, "VIP": 25000, "VVIP": 30000},
        }

        self.create_widgets()
```

```
    def create_widgets(self):
        tk.Label(
            self.root,
            text="YKC GAME SPACE",
            font=("Arial", 30, "bold"),
            bg="#17153B",
            fg="#C8ACD6"
        ).pack(pady=10)

        frame = tk.Frame(self.root, bg="#17153B")
        frame.pack(pady=10)

        tk.Label(frame, text="Nama:", bg="#17153B", fg="#C8ACD6",
font=("Arial", 10)).grid(row=0, column=0, sticky="w")
        self.name_entry = tk.Entry(frame, width=25)
        self.name_entry.grid(row=0, column=1, pady=5)

        tk.Label(frame, text="Kelas:", bg="#17153B", fg="#C8ACD6",
font=("Arial", 10)).grid(row=1, column=0, sticky="w")
        self.class_var = tk.StringVar(value="Reguler")
        class_combobox = ttk.Combobox(
            frame, textvariable=self.class_var, values=["Reguler", "VIP",
"VVIP"], width=22
        )
        class_combobox.grid(row=1, column=1, pady=5)
        class_combobox.bind("<<ComboboxSelected>>", self.update_options)

        tk.Label(frame, text="PS:", bg="#17153B", fg="#C8ACD6",
font=("Arial", 10)).grid(row=2, column=0, sticky="w")
        self.ps_var = tk.StringVar(value="PS 3")
        self.ps_combobox = ttk.Combobox(frame, textvariable=self.ps_var,
values=["PS 3", "PS 4"], width=22)
        self.ps_combobox.grid(row=2, column=1, pady=5)

        tk.Label(frame, text="Room:", bg="#17153B", fg="#C8ACD6",
font=("Arial", 10)).grid(row=3, column=0, sticky="w")
        self.room_var = tk.StringVar(value="Room 1")
        self.room_combobox = ttk.Combobox(frame,
textvariable=self.room_var, values=["Room 1"], width=22)
        self.room_combobox.grid(row=3, column=1, pady=5)

        tk.Label(frame, text="Durasi (jam):", bg="#17153B", fg="#C8ACD6",
font=("Arial", 10)).grid(row=4, column=0, sticky="w")
        self.duration_entry = tk.Entry(frame, width=25)
```

```
        self.duration_entry.grid(row=4, column=1, pady=5)

        tk.Button(
            frame, text="Tambah Pesanan", command=self.add_order,
bg="#433D8B", fg="white", font=("Arial", 10)
        ).grid(row=5, column=1, pady=10, sticky="e")

        tk.Label(self.root, text="Riwayat Pesanan", bg="#17153B",
fg="#C8ACD6", font=("Arial", 12, "bold")).pack(pady=5)
        self.order_table = ttk.Treeview(
            self.root, columns=("Nama", "Kelas", "PS", "Room", "Durasi",
"Harga"), show="headings", height=10
        )
        self.order_table.heading("Nama", text="Nama")
        self.order_table.heading("Kelas", text="Kelas")
        self.order_table.heading("PS", text="PS")
        self.order_table.heading("Room", text="Room")
        self.order_table.heading("Durasi", text="Durasi")
        self.order_table.heading("Harga", text="Total Harga")
        self.order_table.pack(fill=tk.BOTH, expand=True, padx=10, pady=5)

        tk.Button(
            self.root, text="Clear Tabel", command=self.clear_table,
bg="#dc3545", fg="white", font=("Arial", 10)
        ).pack(pady=5)

    def update_options(self, event=None):
        """Update pilihan PS dan Room berdasarkan kelas yang dipilih."""
        kelas = self.class_var.get()

        if kelas == "Reguler":
            self.ps_combobox["values"] = ["PS 3", "PS 4"]
            self.room_combobox["values"] = [f"Room {i}" for i in range(1,
9)]
        elif kelas == "VIP":
            self.ps_combobox["values"] = ["PS 4", "PS 5"]
            self.room_combobox["values"] = [f"Room {i}" for i in range(1,
7)]
        elif kelas == "VVIP":
            self.ps_combobox["values"] = ["PS 5"]
            self.room_combobox["values"] = [f"Room {i}" for i in range(1,
5)]

        self.ps_var.set(self.ps_combobox["values"][0])
        self.room_var.set(self.room_combobox["values"][0])
```

```
    def calculate_price(self, ps, kelas, duration):
        return self.price_data[ps][kelas] * int(duration)

    def add_order(self):
        name = self.name_entry.get()
        kelas = self.class_var.get()
        ps = self.ps_var.get()
        room = self.room_var.get()
        duration = self.duration_entry.get()

        if not name or not duration.isdigit():
            messagebox.showerror("Error", "Pastikan semua data diisi
dengan benar.")
            return

        if kelas not in self.price_data[ps]:
            messagebox.showerror("Error", f"{ps} tidak tersedia untuk
kelas {kelas}.")
            return

        total_price = self.calculate_price(ps, kelas, duration)
        order = (name, kelas, ps, room, f"{duration} jam", f"Rp
{total_price:,}")
        self.order_table.insert("", "end", values=order)
        self.orders.append(order)
        self.clear_inputs()

    def clear_inputs(self):
        self.name_entry.delete(0, tk.END)
        self.duration_entry.delete(0, tk.END)
        self.class_var.set("Reguler")
        self.update_options()

    def clear_table(self):
        for row in self.order_table.get_children():
            self.order_table.delete(row)
        self.orders.clear()
        messagebox.showinfo("Clear", "Tabel berhasil dibersihkan.")


if __name__ == "__main__":
    root = tk.Tk()
    app = PSRentalAppWithRoom(root)
    root.mainloop()
```

4. Penjelasan Atribute dan Fungsi

1. __init__

Fungsi ini adalah konstruktor yang akan dijalankan saat aplikasi dijalankan. Di sini, semua atribut utama seperti root (jendela utama), daftar pesanan, dan data harga ditentukan. Fungsi ini juga memanggil fungsi `create_widgets` untuk membangun elemen GUI.

2. create_widgets

Fungsi ini bertugas membuat semua elemen antarmuka pengguna (GUI), seperti label, input field, dan tombol. Elemen-elemen ini diatur menggunakan container (frame) untuk mempermudah penempatan.

3. update_options

Fungsi ini digunakan untuk memperbarui opsi PlayStation dan room berdasarkan kelas yang dipilih oleh pengguna.

4. calculate_price

Fungsi ini menghitung total harga berdasarkan jenis PlayStation, kelas, dan durasi sewa.

5. add_order

Fungsi ini menangani proses penambahan pesanan. Data seperti nama pelanggan, kelas, PlayStation, room, dan durasi divalidasi terlebih dahulu sebelum dihitung total harganya dan ditambahkan ke tabel riwayat pesanan.

6. clear_inputs

Fungsi ini digunakan untuk menghapus semua input yang telah dimasukkan pengguna setelah pesanan berhasil ditambahkan.

7. clear_table

Fungsi ini membersihkan seluruh data dari tabel riwayat pesanan dan daftar pesanan. Notifikasi juga diberikan kepada pengguna setelah proses selesai.

5. Penutup

Kode ini menunjukkan implementasi aplikasi rental PS sederhana dengan paradigma OOP dan menggunakan Tkinter sebagai GUI. Penggunaan struktur kelas mempermudah pengelolaan elemen-elemen aplikasi dan meningkatkan keterbacaan kode.