LAPORAN STUDI KASUS
APLIKASI RESERVASI HOTEL

DISUSUN OLEH :

NAMA : BIRGITA EGI AZH FARRYAH

NPM : 5230411255

KELAS : VIII

MATA KULIAH : PEMROGRAMAN BERORIENTASI OBJEK PRAKTIK

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS TEKNOLOGI YOGYAKARTA

2024

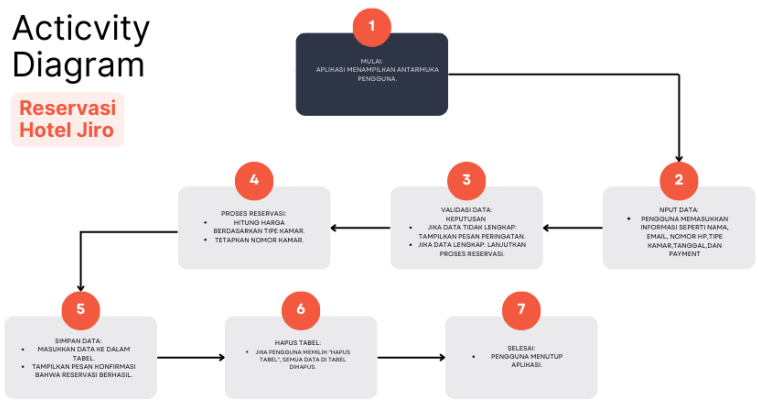
PENDAHULUAN

Aplikasi *Reservasi Hotel Jiro* adalah sebuah program yang dibuat untuk mempermudah proses pemesanan kamar di sebuah hotel. Aplikasi ini dibangun menggunakan bahasa pemrograman Python dengan bantuan pustaka **Tkinter** yang digunakan untuk membangun antarmuka pengguna grafis (GUI). Fitur utama dari aplikasi ini termasuk pemesanan kamar, pemilihan tipe kamar, penentuan harga, serta pengelolaan data pemesanan seperti nama pelanggan, email, dan metode pembayaran.

STUDI KASUS

Studi kasus ini mengembangkan aplikasi *Reservasi Hotel Jiro* untuk mempermudah proses pemesanan kamar hotel baik untuk pelanggan maupun pengelola hotel. Aplikasi memungkinkan pelanggan untuk memasukkan data seperti nama, email, nomor telepon, tipe kamar yang diinginkan, serta tanggal check-in dan check-out. Sistem ini otomatis memberikan nomor kamar dan mencatat detail reservasi ke dalam tabel yang dapat diakses oleh staf hotel. Aplikasi ini juga menyediakan berbagai metode pembayaran, seperti uang tunai, kartu kredit, dan transfer bank, serta memastikan data yang dimasukkan valid sebelum reservasi diproses. Dengan demikian, aplikasi ini meningkatkan efisiensi dan mengurangi kesalahan dalam pemesanan kamar hotel.

A. ACTIVITY DIAGRAM



B. PENJELASAN KODE PROGRAM

1. Import Library

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

Penjelasan:

Pada bagian awal program yaitu mengimpor pustaka – pustaka yang diperlukan:

- **Tkinter** (tk): Digunakan untuk membangun elemen-elemen GUI seperti tombol, label, dan form input.
- **ttk**: Digunakan untuk elemen GUI tambahan seperti **ComboBox** (untuk dropdown menu) dan **Treeview** (untuk tabel).
- **messagebox**: Digunakan untuk menampilkan pesan pop-up, seperti pemberitahuan atau peringatan kepada pengguna.

2. Mendefinisikan Kelas

```
class ReservasiHotel:
    def __init__(self, root):
        self.root = root
        self.root.title("Aplikasi Reservasi Hotel Jiro")

        self.root.geometry("900x700")
```

Penjelasan:

Bagian ini mendefinisikan kelas **ReservasiHotel** yang berfungsi sebagai aplikasi utama. Kelas ini membutuhkan objek **Tk** yang akan digunakan untuk membuat jendela utama aplikasi.

- root adalah jendela utama dari aplikasi.
- self.root.title memberi judul pada jendela aplikasi.
- self.root.geometry mengatur ukuran jendela aplikasi menjadi **900x700 piksel**.

3. Menentukan Data Harga dan Nomor Kamar

```
self.harga_kamar = {
            "single": 1000000,
            "double": 1700000,
            "suite": 3000000,
            "deluxe": 4000000
        }
```

```
        self.nomor_kamar_counter = {
            "single": 101,
            "double": 201,
            "suite": 301,
            "deluxe": 401
    }
```

Penjelasan:

Pada bagian ini, kita membuat dua dictionary untuk menyimpan data harga kamar dan nomor kamar yang akan digunakan:

- self.harga_kamar: Menyimpan harga untuk masing-masing tipe kamar.
- self.nomor_kamar_counter: Menyimpan nomor kamar pertama untuk setiap tipe kamar (misalnya, tipe kamar *single* dimulai dari nomor 101, *double* dari 201, dan seterusnya). Nomor kamar ini akan otomatis bertambah setiap kali ada pemesanan.

4. Membuat Widget untuk Antarmuka Pengguna

```
def buat_widget(self):
        # Frame untuk judul
        frame_judul = tk.Frame(self.root, bg="#A9A9A9")  # Abu-abu gelap
        frame_judul.pack(pady=10)
        label_judul = tk.Label(
            frame_judul, text="Aplikasi Reservasi Hotel Jiro",
           font=("Arial", 18, "bold"), fg="#FFFFFF", bg="#A9A9A9"  # Putih text on dark
gray
```

Penjelasan:

Pada bagian ini adalah membuat frame dan label untuk menampilkan judul aplikasi.

1. Frame Judul (frame_judul):
    - Dibuat menggunakan **tk.Frame**, yang berfungsi sebagai wadah atau kontainer.
    - **Warna latar belakang (background)** frame diatur menjadi abu-abu gelap dengan kode warna #A9A9A9.
    - **pack(pady=10)** menambahkan jarak vertikal (padding) sebesar 10 piksel di sekitar frame, membuatnya tidak terlalu rapat dengan elemen lainnya.
2. Label Judul (label_judul):
    - Label ini menampilkan teks "Aplikasi Reservasi Hotel Jiro" yang merupakan judul aplikasi.
    - Font yang digunakan adalah **Arial**, dengan ukuran **18** dan gaya **bold** (tebal), membuat teks terlihat lebih menonjol.
    - Warna teksnya diatur menjadi putih (**fg="#FFFFFF"**), dengan latar belakang yang sama seperti frame, yaitu abu-abu gelap (**bg="#A9A9A9"**).

- Label ini ditempatkan di dalam frame menggunakan metode **pack()**, yang secara otomatis akan memposisikannya di tengah frame.

5. **Membuat Frame Sebagai Tempat Meletakkan Elemen**

```
frame_input = tk.Frame(self.root, bd=2, relief="solid", bg="#D3D3D3")  # Abu-abu terang
    frame_input.pack(pady=10, padx=20, fill="x")
```

Penjelasan:

Kode ini membuat **frame (bingkai)** sebagai tempat elemen input pada antarmuka aplikasi. Frame diberi:

- **Batas tebal (bd=2)** dengan **garis solid (relief="solid")**,
- **Warna latar abu-abu terang (bg="#D3D3D3")** untuk tampilan bersih,
- **Jarak luar**: 10 piksel vertikal (pady) dan 20 piksel horizontal (padx),
- **Lebar penuh (fill="x")** agar frame memenuhi lebar jendela.

Tujuannya: Mengelompokkan elemen input agar lebih rapi dan terorganisasi.

6. **Menambahkan Label Dan Kotak Input Nama**

```
tk.Label(frame_input, text="Nama Lengkap:", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=0, column=0, padx=10, pady=5, sticky="w")
            self.nama_entry = tk.Entry(frame_input, width=30, bg="#FFFFFF", fg="#000000")
    self.nama_entry.grid(row=0, column=1, padx=10, pady=5)
```

Penjelasan:

Kode ini menambahkan **label** dan **kotak input** untuk mengisi nama lengkap:

1. **Label "Nama Lengkap"** ditempatkan di **baris 0, kolom 0** pada frame frame_input dengan:
   - Teks rata kiri (anchor="w")
   - Warna latar abu-abu terang (bg="#D3D3D3")
   - Teks hitam (fg="#000000")
   - Jarak dalam (padx=10, pady=5)
   - Menempel ke kiri (sticky="w")
2. **Kotak input (Entry)** untuk memasukkan nama ditempatkan di **baris 0, kolom 1** dengan:
   - Lebar 30 karakter (width=30)

- Latar putih dan teks hitam
- Jarak dalam serupa.

Tujuannya: Memberikan tempat untuk pengguna memasukkan nama mereka.

7. **Menambahkan Label Dan Kotak Input Data Pelanggan Email**

```
# Email pelanggan
        tk.Label(frame_input, text="Email:", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=1, column=0, padx=10, pady=5, sticky="w")
        self.email_entry = tk.Entry(frame_input, width=30, bg="#FFFFFF", fg="#000000")
    self.email_entry.grid(row=1, column=1, padx=10, pady=5)
```

Penjelasan:

1. Label untuk Email Pelanggan

- **tk.Label**: Membuat label berisi teks "Email:" untuk menunjukkan kepada pengguna bahwa kotak input di sebelahnya adalah untuk mengisi email.
- **Pengaturan properti**:
  - **text="Email:"**: Menampilkan teks "Email".
  - **anchor="w"**: Menempatkan teks rata ke kiri.
  - **bg="#D3D3D3"**: Latar belakang label berwarna abu-abu terang agar serasi dengan frame input.
  - **fg="#000000"**: Teks label berwarna hitam agar mudah dibaca.
- **grid(row=1, column=0, padx=10, pady=5, sticky="w")**: Menempatkan label di baris ke-1, kolom ke-0 pada grid dengan margin horizontal (**padx**) dan vertikal (**pady**) sebesar 10 dan 5 piksel. Properti **sticky="w"** memastikan label diratakan ke kiri.

2. Kotak Input untuk Email

- **tk.Entry**: Membuat kotak input untuk pengguna mengetikkan email mereka.
- **Pengaturan properti**:
  - **width=30**: Memberikan lebar kolom input yang cukup untuk menampung alamat email.
  - **bg="#FFFFFF"**: Latar belakang kolom input berwarna putih.
  - **fg="#000000"**: Teks yang diketikkan pengguna akan berwarna hitam.
- **grid(row=1, column=1, padx=10, pady=5)**: Menempatkan kotak input di baris ke-1, kolom ke-1 pada grid dengan margin horizontal dan vertikal yang sama seperti label.

```
# Nomor HP pelanggan
        tk.Label(frame_input, text="Nomor HP:", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=2, column=0, padx=10, pady=5, sticky="w")
```

```
        self.phone_entry = tk.Entry(frame_input, width=30, bg="#FFFFFF", fg="#000000")
        self.phone_entry.grid(row=2, column=1, padx=10, pady=5)
```

Penjelasan:

3. Label untuk Nomor HP

- **tk.Label**: Membuat label berisi teks "Nomor HP:" untuk menunjukkan bahwa kotak input di sebelahnya digunakan untuk mengisi nomor HP pelanggan.
- **Pengaturan properti**:
    - Sama seperti label email, hanya teksnya yang berbeda: **text="Nomor HP:"**.
- **grid(row=2, column=0, padx=10, pady=5, sticky="w")**: Menempatkan label di baris ke-2, kolom ke-0 pada grid.

4. Kotak Input untuk Nomor HP

- **tk.Entry**: Membuat kotak input untuk memasukkan nomor HP pelanggan.
- **Pengaturan properti**:
    - Sama seperti kotak input email, hanya lokasi di grid yang berbeda.
- **grid(row=2, column=1, padx=10, pady=5)**: Menempatkan kotak input di baris ke-2, kolom ke-1 pada grid.

8. Membuat Bagian Untuk Memilih Tipe Kamar

```
# Pilihan tipe kamar
        tk.Label(frame_input, text="Pilih Tipe Kamar:", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=3, column=0, padx=10, pady=5, sticky="w")
        self.room_var = tk.StringVar(value="")
        room_options = list(self.harga_kamar.keys())
        self.room_menu = ttk.Combobox(frame_input, values=room_options, textvariable=self.room_var, state="readonly", width=27)
        self.room_menu.grid(row=3, column=1, padx=10, pady=5)
```

Penjelasan:

1. **Label "Pilih Tipe Kamar":**
    - Menampilkan teks **"Pilih Tipe Kamar:"** untuk memberi tahu pengguna bahwa bagian ini digunakan untuk memilih jenis kamar.
    - Label ini ditempatkan di baris ketiga, kolom pertama, dengan teks hitam dan latar abu-abu terang.
    - Penempatan teks diratakan ke kiri.

2. **Variabel self.room_var:**
    - self.room_var adalah variabel yang digunakan untuk menyimpan pilihan tipe kamar yang dipilih pengguna.
    - Pada awalnya, variabel ini diberi nilai kosong ("").
3. **Daftar opsi tipe kamar (room_options):**
    - Membuat daftar pilihan tipe kamar (misalnya, "single", "double", "suite", "deluxe") berdasarkan kunci dalam dictionary self.harga_kamar.
    - Daftar ini akan digunakan sebagai opsi pada menu pilihan kamar.
4. **Menu pilihan tipe kamar (self.room_menu):**
    - Membuat menu dropdown (combo box) yang memungkinkan pengguna memilih salah satu tipe kamar dari daftar room_options.
    - textvariable=self.room_var menghubungkan pilihan pengguna dengan variabel self.room_var, sehingga pilihan yang dipilih pengguna dapat diakses dari variabel ini.
    - Opsi menu bersifat **readonly**, sehingga pengguna hanya bisa memilih dari daftar yang tersedia tanpa mengetik manual.
    - Lebar menu diatur menjadi 27 karakter.
5. **Penempatan menu:**

- Menu dropdown ditempatkan di baris ketiga, kolom kedua, di sebelah kanan label "Pilih Tipe Kamar".
- Jarak antar elemen diatur dengan padding horizontal dan vertikal.

9. Membuat Bagian Tanggal Check In dan Check Out

```
# Tanggal check-in
            tk.Label(frame_input, text="Check-in (YYYY-MM-DD):", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=4, column=0, padx=10, pady=5, sticky="w")
            self.check_in_entry = tk.Entry(frame_input, width=30, bg="#FFFFFF", fg="#000000")
        self.check_in_entry.grid(row=4, column=1, padx=10, pady=5)

        # Tanggal check-out
            tk.Label(frame_input, text="Check-out (YYYY-MM-DD):", anchor="w", bg="#D3D3D3", fg="#000000").grid(row=5, column=0, padx=10, pady=5, sticky="w")
            self.check_out_entry = tk.Entry(frame_input, width=30, bg="#FFFFFF", fg="#000000")
        self.check_out_entry.grid(row=5, column=1, padx=10, pady=5)
```

Penjelasam:

1. Tanggal Check-in:

- **Label "Check-in (YYYY-MM-DD)":**
  Menampilkan teks **"Check-in (YYYY-MM-DD):"** yang memberi petunjuk kepada pengguna bahwa mereka harus memasukkan tanggal check-in dalam format **Tahun-Bulan-Tanggal** (YYYY-MM-DD).
  Label ini diletakkan pada baris keempat, kolom pertama, dengan teks berwarna hitam dan latar belakang abu-abu terang, serta rata kiri.

- **Input untuk Tanggal Check-in (self.check_in_entry):**
  Membuat **kolom input** (entry) tempat pengguna bisa mengetikkan tanggal check-in mereka.
  Kolom ini memiliki lebar 30 karakter, latar belakang putih, dan teks berwarna hitam.

- **Penempatan:**
  Kolom input diletakkan di baris keempat, kolom kedua, tepat di sebelah kanan label "Check-in".
  Ada padding horizontal dan vertikal yang memberi jarak antar elemen.

2. Tanggal Check-out:

   - **Label "Check-out (YYYY-MM-DD)":**
     Menampilkan teks **"Check-out (YYYY-MM-DD):"** untuk memberi petunjuk pada pengguna agar mengisi tanggal check-out dalam format yang sama.
     Label ini diletakkan pada baris kelima, kolom pertama, dengan teks hitam dan latar abu-abu terang, serta rata kiri.

   - **Input untuk Tanggal Check-out (self.check_out_entry):**
     Membuat **kolom input** (entry) tempat pengguna dapat mengetikkan tanggal check-out mereka.
     Kolom ini juga memiliki lebar 30 karakter, latar belakang putih, dan teks berwarna hitam.

   - **Penempatan:**
     Kolom input diletakkan pada baris kelima, kolom kedua, tepat di sebelah kanan label "Check-out", dengan jarak yang sama seperti sebelumnya.

10. Pilihan Metode Pembayaran

```
# Pilihan metode pembayaran
        tk.Label(frame_input, text="Metode Pembayaran:", anchor="w", bg="#D3D3D3",
fg="#000000").grid(row=6, column=0, padx=10, pady=5, sticky="w")
        self.payment_var = tk.StringVar(value="")
        payment_options = ["Cash", "Credit Card", "E-Wallet", "Bank Transfer"]
            self.payment_menu = ttk.Combobox(frame_input, values=payment_options,
textvariable=self.payment_var, state="readonly", width=27)
        self.payment_menu.grid(row=6, column=1, padx=10, pady=5)
```

Penjelasan:

1. Label "Metode Pembayaran":

- Label:
  Label ini menampilkan teks "Metode Pembayaran:" yang memberi petunjuk kepada pengguna untuk memilih metode pembayaran yang mereka inginkan. Label ini diletakkan pada baris ke-6, kolom pertama. Warna teksnya hitam dan latar belakangnya abu-abu terang.

2. Dropdown Menu untuk Metode Pembayaran (self.payment_menu):

- Variabel payment_var:
  Variabel ini digunakan untuk menyimpan nilai yang dipilih oleh pengguna. Nilai ini akan dikaitkan dengan dropdown menu.
- Daftar Pilihan Pembayaran (payment_options):
  Ini adalah daftar pilihan metode pembayaran yang bisa dipilih pengguna, yaitu "Cash" (Tunai), "Credit Card" (Kartu Kredit), "E-Wallet" (Dompet Digital), dan "Bank Transfer" (Transfer Bank).
- Combobox (ttk.Combobox):
  Ini adalah dropdown menu yang memungkinkan pengguna memilih salah satu dari pilihan yang tersedia (Cash, Credit Card, E-Wallet, Bank Transfer). Dropdown menu ini memiliki lebar 27 karakter dan hanya memungkinkan pemilihan bukan pengetikan (karena state="readonly").

3. Penempatan:

- Dropdown menu ini diletakkan pada baris ke-6, kolom ke-2, tepat di sebelah kanan label "Metode Pembayaran:".
- Ada padding horizontal dan vertikal untuk memberi jarak antar elemen.

11. Tombol Tambah Reservasi

```
# Tombol tambah pesanan
        tk.Button(self.root, text="Tambah Reservasi", command=self.tambah_reservasi, bg="#A9A9A9", fg="#000000", font=("Arial", 12)).pack(pady=10)
```

Penjelasan:

1. Tombol "Tambah Reservasi":

- Widget Tombol (tk.Button)
  Kode ini membuat sebuah tombol dengan teks "Tambah Reservasi" di dalam aplikasi. Tombol ini akan digunakan untuk menambah data reservasi yang diinput oleh pengguna.

2. Perintah yang Dijalankan (command=self.tambah_reservasi):

- Ketika tombol ini ditekan, perintah atau fungsi self.tambah_reservasi akan dijalankan. Fungsi ini akan memproses data yang dimasukkan pengguna (seperti nama, email, tipe kamar, dll.) dan menambahkannya ke dalam tabel daftar reservasi.

3. Pengaturan Tampilan Tombol:

- Warna Latar Belakang (bg="#A9A9A9"), Tombol ini memiliki warna latar belakang abu-abu gelap.
- Warna Teks (fg="#000000"), Warna teks pada tombol adalah hitam.
- Font(font=("Arial",12))
  Teks pada tombol menggunakan font Arial dengan ukuran 12.

4. Penempatan Tombol (pack(pady=10)):

- Tombol ini ditempatkan secara otomatis di bawah elemen lainnya menggunakan metode pack().
- pady=10 memberikan jarak vertikal sebesar 10 piksel antara tombol dan elemen lain di sekitarnya

12. Frame Tabel Data Reservasi

```
# Frame untuk tabel daftar reservasi
    frame_tabel = tk.Frame(self.root, bg="#D3D3D3")  # Abu-abu terang untuk tabel
    frame_tabel.pack(padx=20, pady=10, fill="both", expand=True)
```

1. Membuat Frame untuk Tabel:

- Widget Frame (tk.Frame)
  Kode ini membuat sebuah frame (wadah) di dalam aplikasi yang digunakan untuk menampilkan tabel daftar reservasi. Frame ini memiliki latar belakang abu-abu terang (bg="#D3D3D3"), yang membuatnya lebih terlihat kontras dengan elemen lainnya.

2. Penempatan Frame dalam Aplikasi (pack()):

- padx=20, pady=10
  Frame ini diberi jarak (padding) 20 piksel di kiri dan kanan (padx=20) serta 10 piksel di atas dan bawah (pady=10). Ini memastikan frame tidak menempel langsung dengan batas layar atau elemen lainnya.
- fill="both"
  Opsi fill="both" membuat frame ini mengisi ruang yang tersedia secara horizontal dan vertikal, mengikuti ukuran jendela aplikasi.
- expand=True
  Dengan expand=True, frame ini akan "mengembang" dan menyesuaikan ukurannya agar tetap mengisi ruang kosong yang tersedia dalam window. Ini penting agar frame selalu memiliki ukuran yang cukup untuk menampilkan tabel dengan baik.

13. Scrollbar Untuk Tabel

```
# Scrollbar untuk tabel
    scrollbar_y = ttk.Scrollbar(frame_tabel, orient="vertical")
    scrollbar_x = ttk.Scrollbar(frame_tabel, orient="horizontal")
```

Kode ini membuat dua scrollbar (bilah gulir) untuk tabel daftar reservasi agar pengguna dapat menggulir tabel jika data dalam tabel lebih banyak dari yang bisa ditampilkan dalam satu layar.

Penjelasan per bagian:

1. scrollbar_y = ttk.Scrollbar(frame_tabel, orient="vertical")
    - Membuat scrollbar vertikal (orient="vertical") yang akan digunakan untuk menggulir tabel secara vertikal (ke atas atau ke bawah).
    - frame_tabel adalah tempat di mana scrollbar ini akan diletakkan, yaitu di dalam frame tempat tabel berada.
2. scrollbar_x = ttk.Scrollbar(frame_tabel, orient="horizontal")
    - Membuat scrollbar horizontal (orient="horizontal") yang memungkinkan pengguna menggulir tabel secara horizontal (ke kiri atau ke kanan) jika tabel memiliki banyak kolom.
    - Sama seperti scrollbar vertikal, scrollbar ini juga akan diletakkan di dalam frame_tabel.

Tujuan:

- Scrollbar vertikal digunakan untuk menggulir baris tabel yang panjang.
- Scrollbar horizontal digunakan untuk menggulir kolom tabel yang lebih lebar dari lebar layar atau tampilan jendela aplikasi.

Kedua scrollbar ini membantu memastikan bahwa tabel tetap dapat diakses dengan mudah meskipun datanya lebih besar dari ruang tampilan yang tersedia.

14. Tabel Daftar Reservasi

```
# Tabel daftar reservasi
    self.tabel_pesanan = ttk.Treeview(
        frame_tabel,
            columns=("Nama", "Email", "Phone", "Tipe Kamar", "Nomor Kamar", "Harga",
"Check-in", "Check-out", "Payment"),
        show="headings",
        yscrollcommand=scrollbar_y.set,
        xscrollcommand=scrollbar_x.set
    )
    scrollbar_y.config(command=self.tabel_pesanan.yview)
    scrollbar_x.config(command=self.tabel_pesanan.xview)
    scrollbar_y.pack(side="right", fill="y")
    scrollbar_x.pack(side="bottom", fill="x")
```

```
self.tabel_pesanan.pack(fill="both", expand=True)

for col in self.tabel_pesanan["columns"]:
    self.tabel_pesanan.heading(col, text=col)
    self.tabel_pesanan.column(col, anchor="center", width=120)
```

Kode ini digunakan untuk membuat dan menampilkan tabel daftar reservasi dengan beberapa kolom yang berisi informasi tentang pesanan hotel, serta menambahkan scrollbar agar tabel bisa digulir baik secara vertikal maupun horizontal jika data lebih banyak dari ruang yang tersedia.

Membuat Tabel (ttk.Treeview)

ttk.Treeview adalah widget yang digunakan untuk menampilkan data dalam bentuk tabel di Tkinter.
frame_tabel adalah tempat (frame) di mana tabel ini akan diletakkan.
columns menentukan kolom-kolom yang ada dalam tabel, misalnya "Nama", "Email", "Phone", dll.
show="headings" menampilkan nama kolom (header) di bagian atas tabel tanpa menampilkan kolom nomor internal.
yscrollcommand=scrollbar_y.set dan **xscrollcommand=scrollbar_x.set** menghubungkan scrollbar vertikal dan horizontal dengan tabel, sehingga saat pengguna menggulir scrollbar, tampilan tabel juga ikut bergerak.

Mengonfigurasi Scrollbar

scrollbar_y.config(command=self.tabel_pesanan.yview) mengatur agar scrollbar vertikal dapat mengendalikan tampilan vertikal dari tabel.

scrollbar_x.config(command=self.tabel_pesanan.xview) mengatur agar scrollbar horizontal dapat mengendalikan tampilan horizontal dari tabel.

Menampilkan Scrollbar dan Tabel

scrollbar_y.pack(side="right", fill="y") menempatkan scrollbar vertikal di sebelah kanan tabel dan mengisi ruang secara vertikal.

scrollbar_x.pack(side="bottom", fill="x") menempatkan scrollbar horizontal di bagian bawah tabel dan mengisi ruang secara horizontal.

self.tabel_pesanan.pack(fill="both", expand=True) menempatkan tabel di dalam frame dan memastikan tabel mengisi seluruh ruang yang tersedia.

Menambahkan Judul Kolom dan Menyusun Tampilan Kolom

for col in self.tabel_pesanan["columns"]: melakukan iterasi melalui setiap kolom dalam tabel.

self.tabel_pesanan.heading(col, text=col) menambahkan nama kolom sebagai judul di bagian atas tabel.

self.tabel_pesanan.column(col, anchor="center", width=120) mengatur agar setiap kolom memiliki teks yang terpusat (centered) dan lebar kolom sebesar 120 piksel.

15. Tombol Clear Table

```
# Tombol clear table
        tk.Button(self.root, text="Hapus Tabel", command=self.hapus_tabel, bg="#A9A9A9",
fg="#000000", font=("Arial", 12)).pack(pady=10)
```

Kode ini membuat sebuah tombol dengan label **"Hapus Tabel"** yang berfungsi untuk menghapus semua data yang ada di dalam tabel reservasi. Ketika tombol ini ditekan, perintah untuk menghapus tabel (self.hapus_tabel) akan dijalankan. Tombol diberi desain dengan warna latar belakang abu-abu (#A9A9A9) dan teks berwarna hitam (#000000), serta menggunakan font Arial dengan ukuran 12. Tombol ini ditempatkan di bawah dengan jarak (padding) 10 piksel.

16. Mengambil Data Yang Dimasukkan Oleh Pengguna Ke Dalam Form Dan Menyimpannya

```
def tambah_reservasi(self):
        nama = self.nama_entry.get().strip()
        email = self.email_entry.get().strip()
        phone = self.phone_entry.get().strip()
        room_type = self.room_var.get().strip()
        check_in = self.check_in_entry.get().strip()
        check_out = self.check_out_entry.get().strip()
        payment_method = self.payment_var.get().strip()
```

Kode ini berfungsi untuk mengambil data yang dimasukkan oleh pengguna ke dalam formulir (form) reservasi dan menyimpannya dalam variabel.

- **nama = self.nama_entry.get().strip()**: Mengambil teks dari kolom nama (entry) dan menghapus spasi kosong di awal dan akhir.
- **email = self.email_entry.get().strip()**: Mengambil teks dari kolom email dan menghapus spasi kosong di awal dan akhir.
- **phone = self.phone_entry.get().strip()**: Mengambil teks dari kolom nomor telepon dan menghapus spasi kosong di awal dan akhir.
- **room_type = self.room_var.get().strip()**: Mengambil tipe kamar yang dipilih dari dropdown (combobox) dan menghapus spasi kosong.
- **check_in = self.check_in_entry.get().strip()**: Mengambil tanggal check-in yang dimasukkan dan menghapus spasi kosong.
- **check_out = self.check_out_entry.get().strip()**: Mengambil tanggal check-out yang dimasukkan dan menghapus spasi kosong.

- **payment_method = self.payment_var.get().strip()**: Mengambil metode pembayaran yang dipilih dan menghapus spasi kosong.

17. Memeriksa Data Yang Belum Diisi

```
if not all([nama, email, phone, room_type, check_in, check_out, payment_method]):
        messagebox.showwarning("Peringatan", "Harap isi semua data!")
        return
```

Kode ini memeriksa apakah ada salah satu data yang belum diisi oleh pengguna. Berikut penjelasan langkah demi langkah:

- **if not all([nama, email, phone, room_type, check_in, check_out, payment_method]):**: Kode ini mengecek apakah ada nilai yang kosong (atau tidak diisi) dari daftar data yang diperoleh sebelumnya (nama, email, phone, room_type, check_in, check_out, payment_method). Fungsi all() akan mengembalikan True jika semua elemen dalam daftar memiliki nilai (tidak kosong), dan False jika ada yang kosong.
- **messagebox.showwarning("Peringatan", "Harap isi semua data!")**: Jika ada data yang kosong, maka akan muncul pesan peringatan (warning) yang memberi tahu pengguna untuk mengisi semua kolom formulir.
- **return**: Setelah menunjukkan pesan peringatan, return digunakan untuk menghentikan eksekusi fungsi lebih lanjut, sehingga proses reservasi tidak dilanjutkan sampai semua data diisi.

18. Penetapan Harga Dan Pemberian No Kamar unik

```
harga = self.harga_kamar.get(room_type, 0)
      room_number = self.nomor_kamar_counter[room_type]
      self.nomor_kamar_counter[room_type] += 1


            self.tabel_pesanan.insert("", "end", values=(nama, email, phone, room_type,
room_number, f"Rp {harga:,}", check_in, check_out, payment_method))
                messagebox.showinfo("Sukses", f"Reservasi atas nama {nama} berhasil
ditambahkan!\nNomor Kamar: {room_number}")
```

harga = self.harga_kamar.get(room_type, 0): Kode ini mencari harga kamar berdasarkan jenis kamar yang dipilih oleh pengguna (misalnya "single", "double", "suite", dll). self.harga_kamar.get(room_type, 0) akan mengembalikan harga yang sesuai jika tipe kamar ditemukan, atau 0 jika tipe kamar tidak ditemukan.

room_number = self.nomor_kamar_counter[room_type]: Kode ini mengambil nomor kamar yang tersedia untuk tipe kamar yang dipilih. Setiap tipe kamar memiliki nomor kamar awal yang disimpan dalam self.nomor_kamar_counter (misalnya, kamar tipe "single" dimulai dari nomor 101).

self.nomor_kamar_counter[room_type] += 1: Setelah mendapatkan nomor kamar yang akan digunakan, kode ini memperbarui nomor kamar untuk tipe kamar tersebut (misalnya, jika tipe kamar "single" baru saja digunakan untuk kamar 101, maka

self.nomor_kamar_counter["single"] akan bertambah menjadi 102). Ini memastikan nomor kamar berikutnya akan unik.

self.tabel_pesanan.insert("", "end", values=(...)): Kode ini menambahkan data reservasi yang baru ke dalam tabel reservasi. self.tabel_pesanan.insert() digunakan untuk memasukkan informasi pelanggan (nama, email, tipe kamar, nomor kamar, harga, tanggal check-in, tanggal check-out, dan metode pembayaran) ke dalam tabel.

messagebox.showinfo("Sukses", f"Reservasi atas nama {nama} berhasil ditambahkan!\nNomor Kamar: {room_number}"): Setelah data berhasil ditambahkan ke tabel, pesan informasi (info) akan muncul di layar yang memberitahukan pengguna bahwa reservasi mereka berhasil ditambahkan, dan juga menunjukkan nomor kamar yang diberikan.

19. Membersihkan Semua Kolom Input Pada Form

```
# Clear the form fields after adding a reservation
        self.clear_form()

    def hapus_tabel(self):
        for item in self.tabel_pesanan.get_children():
            self.tabel_pesanan.delete(item)

    def clear_form(self):
        self.nama_entry.delete(0, tk.END)
        self.email_entry.delete(0, tk.END)
        self.phone_entry.delete(0, tk.END)
        self.room_menu.set("")
        self.check_in_entry.delete(0, tk.END)
        self.check_out_entry.delete(0, tk.END)
        self.payment_menu.set("")
```

self.clear_form():

- Fungsi ini dipanggil setelah menambah reservasi untuk membersihkan semua field input pada form (misalnya nama, email, nomor telepon, pilihan kamar, tanggal check-in, dll) agar form siap untuk input berikutnya.

self.nama_entry.delete(0, tk.END):

- Kode ini menghapus teks yang ada dalam kolom input untuk "Nama Lengkap". Fungsi delete(0, tk.END) menghapus semua teks dari awal hingga akhir kolom input.

self.email_entry.delete(0, tk.END):

- Sama seperti sebelumnya, ini menghapus teks dari kolom input untuk "Email".

self.phone_entry.delete(0, tk.END):

- Ini menghapus teks dari kolom input untuk "Nomor HP".

self.room_menu.set(""):

- Ini mengatur nilai yang dipilih di dropdown (combo box) untuk "Pilih Tipe Kamar" menjadi kosong, sehingga pilihan tipe kamar akan ter-reset.

self.check_in_entry.delete(0, tk.END):

- Ini menghapus teks dari kolom input untuk "Tanggal Check-in".

self.check_out_entry.delete(0, tk.END):

- Ini menghapus teks dari kolom input untuk "Tanggal Check-out".

self.payment_menu.set(""):

- Ini mengatur pilihan dropdown untuk "Metode Pembayaran" menjadi kosong, menghapus pilihan sebelumnya.

20. Fungsi Utama Yang Menjalankan Aplikasi

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

if __name__ == "__main__"::

- Baris ini memastikan bahwa kode di bawahnya hanya akan dijalankan jika skrip ini dijalankan secara langsung, bukan diimpor sebagai modul oleh skrip lain. Ini adalah cara umum untuk memulai program Python.

root = tk.Tk():

- Kode ini membuat **jendela utama** aplikasi dengan menggunakan modul tkinter. tk.Tk() adalah objek dasar untuk membuat antarmuka pengguna grafis (GUI) di tkinter, yang akan menjadi jendela aplikasi utama.

app = ReservasiHotel(root):

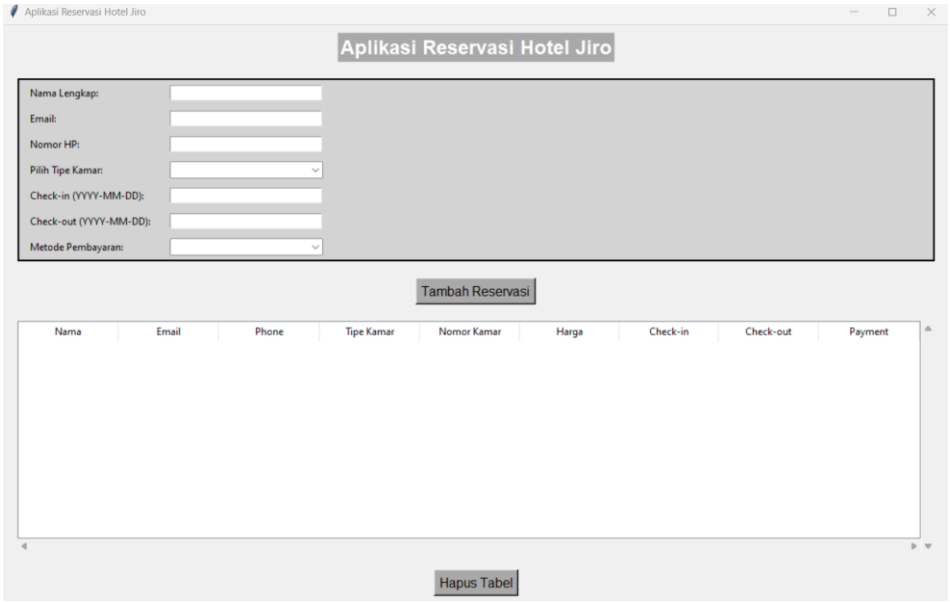
- Di sini, objek ReservasiHotel (yang sudah didefinisikan sebelumnya) dibuat dengan jendela utama root sebagai argumen. Ini memulai aplikasi dan menampilkan antarmuka pengguna yang sudah dibuat sebelumnya dalam kelas ReservasiHotel.

root.mainloop():

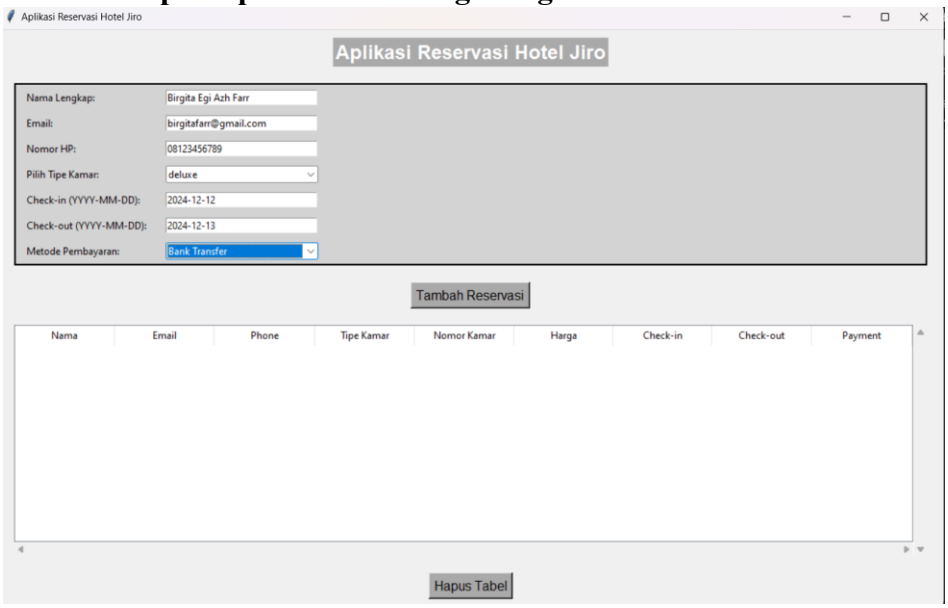
- Kode ini menjalankan **loop utama** aplikasi GUI. mainloop() adalah metode dari objek Tk() yang membuat aplikasi tetap berjalan dan menunggu interaksi dari pengguna (seperti klik tombol atau pengisian form). Program akan terus aktif hingga jendela ditutup.

C. HASIL

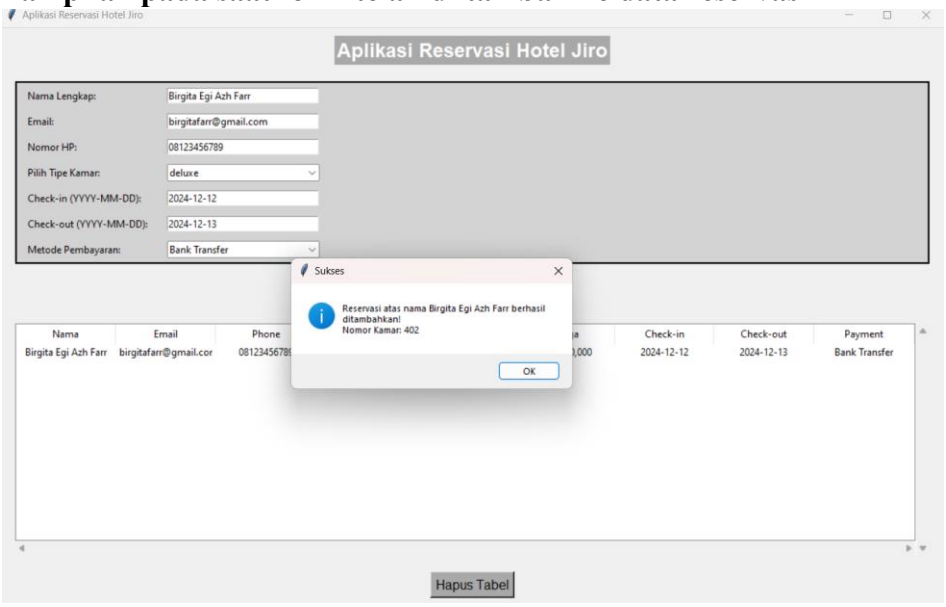
1. **Tampilan saat setelah code dijalankan**



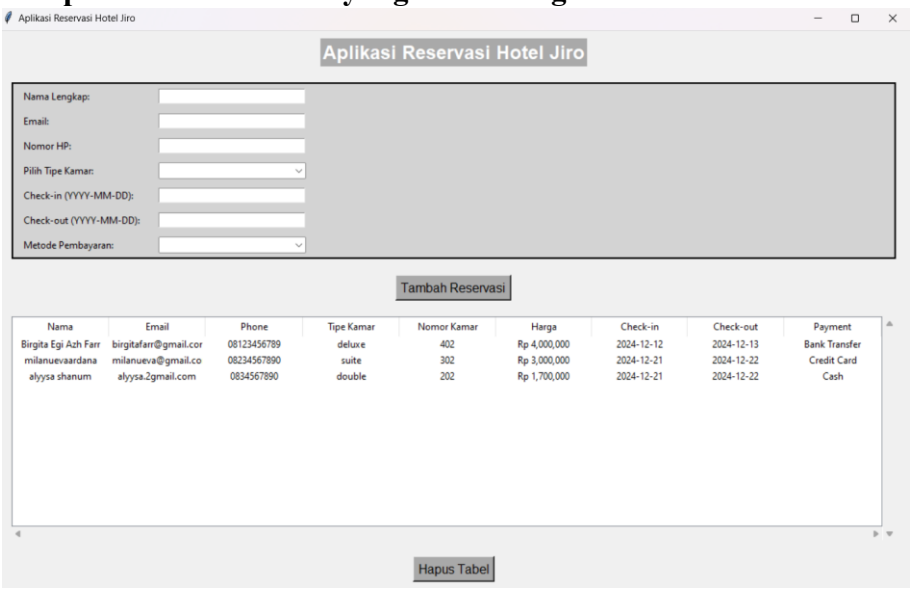
2. **Tampilan pada saat sedang mengisi form**



3. **Tampilan pada saat form telah di tambah ke data reservasi**



4. **Tampilan data customer yang telah mengisi form**



D. KESIMPULAN

Aplikasi **Reservasi Hotel Jiro** dibuat untuk mempermudah proses pemesanan kamar hotel dengan menyediakan fitur lengkap seperti input data pelanggan, validasi informasi, penghitungan harga otomatis berdasarkan tipe kamar, serta pengelolaan data reservasi melalui tabel. Aplikasi ini memanfaatkan Python dan pustaka Tkinter untuk membangun antarmuka pengguna yang mudah dipahami, sehingga dapat membantu meningkatkan efisiensi operasional hotel sekaligus meminimalkan kesalahan dalam pengelolaan reservasi.

E. PENUTUP

Dengan fitur-fitur yang tersedia, aplikasi ini tidak hanya bermanfaat bagi pelanggan yang ingin melakukan reservasi dengan mudah, tetapi juga membantu pihak pengelola hotel dalam mencatat dan mengelola data reservasi secara rapi. Ke depan, aplikasi ini dapat dikembangkan lebih lanjut, seperti menambahkan fitur laporan harian, integrasi pembayaran digital, atau koneksi dengan database untuk menyimpan data secara permanen. Hal ini akan membuat aplikasi semakin relevan dan dapat digunakan pada berbagai skala operasional hotel.