LAPORAN PROYEK AKHIR
DASAR DASAR PEMROGRAMAN
SISTEM ADMINISTRASI DAN PEMBAYARAN KLINIK

Disusun Oleh:

KELOMPOK 2

Aura Putri Anindita Syarif	2509116094
Nabila Viviana Asri	2509116098
Farah Hikmatul Maula	2509116099

Asisten Laboratorium:

TAUFIK RAMADHANI	**DWI PEBRIYANTO P**
2409116001	**2409116012**

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS TEKNIK

UNIVERSITAS MULAWARMAN

2024

KATA PENGANTAR

Puji syukur atas ke hadirat Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, laporan project akhir dengan judul *"*Sistem Administrasi dan Pembayaran Klinik*"* dapat diselesaikan dengan baik dan tepat waktu.

Laporan ini disusun sebagai bentuk penerapan pengetahuan dalam bidang pemrograman dan sistem informasi, khususnya dalam perancangan sistem yang dapat membantu proses administrasi serta pembayaran pada sebuah klinik agar lebih efektif dan efisien. Dalam penyusunan laporan ini, penulis berterimakasih atas bantuan, dukungan, serta bimbingan dari berbagai pihak.

Dalam menyusun makalah ini, penulis sadar bahwa masih banyak terdapat kesalahan dan kekurangan baik dari segi isi maupun penyajian. Penulis berharap saran yang dapat membangun dari berbagai pihak untuk perbaikan di masa yang akan datang.

Akhir kata, semoga laporan ini dapat memberikan manfaat serta menjadi referensi bagi pembaca dalam memahami penerapan sistem administrasi dan pembayaran berbasis teknologi pada layanan klinik.

Oktober, 2025

Penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Deskripsi Masalah

Perkembangan teknologi informasi saat ini memberikan pengaruh yang besar terhadap berbagai sektor, termasuk sektor kesehatan. Salah satu penerapan teknologi yang penting adalah dalam pengelola sistem administrasi dan pembayaran klinik yang menjadi study kasus pada laporan ini.

Proses administrasi yang masih dilakukan secara manual sering menimbulkan berbagai kendala seperti keterlambatan pelayanan, kesalahan pencatatan data pasien, serta kesulitan dalam pengelolaan laporan keuangan. Jika dilakukan secara manual, proses ini rentan terhadap kesalahan. Untuk mengatasi permasalahan tersebut, dibutuhkan suatu sistem yang mampu membantu pengelolaan data administrasi dan pembayaran yang terintegrasi dengan komputer.

Oleh karena itu, perlu dikembangkan sebuah program *Sistem Administrasi dan Pembayaran Klinik* yang dapat mengelola seluruh data secara terintegrasi, menggunakan penyimpanan data dengan format CSV, serta menyediakan fitur autentikasi pengguna, pemesanan layanan, pengelolaan data medis, dan pencatatan transaksi yang terintegrasi.

1.2 Rumusan Masalah

Berdasarkan deskripsi masalah di atas, maka rumusan masalah yang ada ialah:

1. Bagaimana bisa merancang sistem yang dapat mempermudah proses pendaftaran pasien dan login pengguna?
2. Bagaimana sistem dapat mengelola data layanan medis seperti penambahan, pengeditan, dan penghapusan data layanan?
3. Bagaimana sistem dapat mencatat dan menampilkan riwayat pemeriksaan atau pesanan layanan oleh pasien?
4. Bagaimana sistem dapat membantu admin dalam memperbarui status pemeriksaan pasien secara cepat?
5. Bagaimana sistem dapat menyimpan serta menampilkan data secara otomatis dalam format tabel yang rapi dan mudah dibaca?

1.3 Batasan Masalah

Dalam mengembangkan program lebih fokus di satu tujuan, maka batasan masalah yang ada sebagai berikut:

1. Sistem menggunakan dua peran untuk melakukan login yaitu admin dan pasien.
2. Layanan yang diberikan di klinik yaitu : Kosultasi umum, pemeriksaan gigi, tes laboratorium, vaksinasi dan medical chekup.
3. Admin bisa menambahkan data untuk layanan terbaru.
4. Dapat melihat riwayat medis yang dilakukan sebelumnya.
5. Pasien dapat melakukan reservasi untuk layanan kesehatan.

1.4 Tujuan

Tujuan pembuatan sistem administrasi dan pembayaran klinik adalah:

1. Membuat sistem yang mampu membantu proses administrasi pasien dengan cepat dan terorganisir.
2. Meningkatkan efisiensi dalam proses pelayanan klinik.
3. Untuk mempermudah proses registrasi, login, dan pengelolaan pengguna.
4. Untuk memungkinkan pasien melakukan pemesanan layanan dan mencatat riwayat pemeriksaan.
5. Untuk membantu admin dalam memperbarui status pemeriksaan pasien dan melihat seluruh data transaksi.

1.5 Manfaat

Ada banyak manfaat yang diperoleh dari pembuatan sistem ini bagi pihak klinik, pegawai, dan pasien diantaranya ialah:

1. Mempercepat dan memudahkan proses pelayanan, pencatatan data serta mengurangi resiko kesalahan.
2. Mempermudah pengelolaan data pasien dan catatan keuangan.
3. Memberikan kemudahan dalam pendaftaran dan pemesanan layanan tanpa proses manual.
4. Pelayanan yang lebih cepat dan teratur dalam proses administrasi dan pembayaran bagi pasien.

BAB II
PERANCANGAN

2.1 Analisis Program

Program ini dibuat untuk memudahkan pelanggan/pasien dalam memesan layanan klinik, membuat pelayanan menjadi cepat, mempermudah transaksi pembayaran layanan menggunakan e-money. Dalam program ini kami menggunakan beberapa tools untuk menunjang program kami diantaranya yaitu:

1. Conditional Statement (if, elif, dan else)

```
455         if choice == "1":
456             display_layanans(layanans)
457         elif choice == "2":
458             add_service(layanans)
459         elif choice == "3":
460             edit_service(layanans)
461         elif choice == "4":
462             delete_service(layanans)
463         elif choice == "5":
464             view_medical_records(username, riwayat_medis, True)
465         elif choice == "6":
466             update_record_status(riwayat_medis)
467         elif choice == "7":
468             print("Logging out...")
469             break
470         else:
471             print("Pilihan tidak valid!")
```

Gambar 2. 1 Pengginaan IF, ELIF, ELSE

If, elif, dan else digunakan untuk mengambil sebuah keputusan dengan beberapa pilihan atau percabangan. If digunakan untuk memilih kondisi true (benar) sedangkan else digunakan jika pernyaataan false (salah).

2. Perulangan (Looping)

```
113             for record in riwayat_medis:
114                 writer.writerow([
115                     record["date"],
116                     record["username"],
117                     record["patient_name"],
118                     record["service_id"],
119                     record["service_name"],
120                     record["price"],
121                     record["symptoms"],
122                     record["status"],
123                     record.get("balance_before", 0),
124                     record.get("balance_after", 0)
125                 ])
126
```

Gambar 2. 2 Penggunaan Looping

```
130     while True:
131         username = input("Masukkan username (3-20 karakter, alfanumerik): ")
132         if not username.isalnum() or not (3 <= len(username) <= 20):
133             print("Username harus alfanumerik dan 3-20 karakter!")
134             continue
135         if username in users:
136             print("Username sudah terdaftar!")
137             continue
138         break
```

Gambar 2. 3 Penggunaan Looping

Perulangan digunakan untuk memberi perintah dan dilakukan secara secara berulang-ulang kepada computer. Terdapat dua jenis perulangan yaitu for dan while. For digunakan Ketika jumlah perulangan diketahui, sedangkan while digunakan ketika melakukan perulangan dengan kondisi tertentu.

3. Function

```
126
127 def register(users):
128     global layanans, riwayat_medis
129     print("\n=== Register Pasien Baru ===")
130     while True:
131         username = input("Masukkan username (3-20 karakter, alfanumerik): ")
132         if not username.isalnum() or not (3 <= len(username) <= 20):
133             print("Username harus alfanumerik dan 3-20 karakter!")
134             continue
135         if username in users:
136             print("Username sudah terdaftar!")
137             continue
138         break
139
```

Gambar 2. 4 Penggunaan Def

Function adalah blok berisi kode yang berfungsi untuk menjalankan suatu tugas tertentu dan dapat dipanggil ketika fungsi tersebut ingin digunakan. Perintah yang digunakan adalah def.

4. Pretty Table

```
197     table = PrettyTable()
198     table.field_names = ["ID", "Nama Layanan", "Biaya", "Status"]
199     for sid, layanan in layanans.items():
200         status = "Tersedia" if layanan["availability"] else "Tidak Tersedia"
201         table.add_row([sid, layanan["name"], f"Rp {layanan['price']:,}", status])
202     print("\n=== Daftar Layanan Klinik ===")
203     print(table)
```

Gambar 2. 5 Penggunaan Pretty Table

Pretty table adalah library yang digunakan untuk membuat dan menampilkan data dalam format tabel ASCII (American Standard Code for Information Interchange) yang menarik secara visual dan mudah dibaca di hasil output program.

5. Pwinput

```
140     password = pwinput.pwinput(prompt="Masukkan password: ")
141     full_name = input("Nama Lengkap: ")
142     while True:
143         age = input("Usia (angka): ")
144         if not age.isdigit():
145             print("Usia harus angka!")
146             continue
147         age = int(age)
148         break
149     address = input("Alamat: ")
150     while True:
151         phone = input("No. Telepon (maks 15 digit): ")
152         if not phone.isdigit() or len(phone) > 15:
153             print("No. Telepon harus angka dan maksimal 15 digit!")
154             continue
155         break
```

Gambar 2. 6 Penggunaan Pwinput

Pwinput adalah library python yang digunakan untuk menyembunyikan karakter kata sandi yang diketik (menampilkan tanda bintang/asterisk) sehingga tidak terlihat oleh orang lain saat program berjalan.

6. CSV

```
1   import csv
2   import os
```

Gambar 2. 7 Penggunaan CSV

CSV (Comma Separated Value) adalah file teks sederhana yang menyimpan data dalam format table yang nilai-nilai dipisahkan dengan koma dan baris-baris data dipisahkan oleh baris baru. Pada program ini menggunakan csv.DictReader dan csv.writer.

7. Library os

```
1   import csv
2   import os
3   from prettytable import PrettyTable
4   import pwinput
5   from datetime import datetime
```

Gambar 2. 8 Penggunaan Library OS

Library os merupakan modul bawaan (built-in) yang memungkinkan program pada python berinteraksi langsung dengan system operasi computer.

8. Create

```
157     while True:
158         try:
159             initial_balance = int(input("Saldo awal E-Money (Rp): "))
160             if initial_balance < 0:
161                 print("Saldo awal tidak boleh negatif!")
162                 continue
163             break
164         except ValueError:
165             print("Masukkan angka yang valid!")
166 
167     users[username] = {
168         "password": password,
169         "role": "patient",
170         "full_name": full_name,
171         "age": age,
172         "address": address,
173         "phone": phone,
174         "e_money": initial_balance
175     }
176     save_data(users, layanans, riwayat_medis)
177     print("Registrasi berhasil!")
178     return username
```

Gambar 2. 9 Penggunaan Create

Create digunakan untuk pembuatan database dan menambahkan data baru ke dalam penyimpanan data. Salah satu contoh dalam penggunaan create yaitu pembuatan akun user baru di sebuah website/aplikasi.

9. Read

```python
def display_layanans(layanans):
    if not layanans:
        print("Tidak ada layanan tersedia.")
        return

    table = PrettyTable()
    table.field_names = ["ID", "Nama Layanan", "Biaya", "Status"]
    for sid, layanan in layanans.items():
        status = "Tersedia" if layanan["availability"] else "Tidak Tersedia"
        table.add_row([sid, layanan["name"], f"Rp {layanan['price']:,}", status])
    print("\n=== Daftar Layanan Klinik ===")
    print(table)
```

Gambar 2. 10 Penggunaan Read

Read digunakan untuk menampilkan data atau record yang tersimpan dalam database. Salah satu contoh penerapan read yaitu melihat profile yang sudah dibuat.

10. Update

```python

    if new_status in ["Selesai", "Batal", "Menunggu"]:
        record["status"] = new_status
        save_data(users, layanans, riwayat_medis)
        print("Status berhasil diperbarui!")
    else:
        print("Status tidak valid!")
```

Gambar 2. 11 Penggunaan Update

Sesuai namanya, update digunakan untuk mengupdate atau memperbarui data yang dsudah ada dalam database. Salah satu contoh penerapan update adalah mengubah isi konten sebuah program.

11. Delete

```python
def delete_service(layanans):
    global users, riwayat_medis
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang akan dihapus: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return

    confirm = input(f"Anda yakin ingin menghapus layanan {layanans[sid]['name']}? (y/n): ")
    if confirm.lower() == 'y':
        del layanans[sid]
        save_data(users, layanans, riwayat_medis)
        print("Layanan berhasil dihapus!")
    else:
        print("Penghapusan dibatalkan.")
```

Gambar 2. 12 Penggunaan Delete

Delete digunakan untuk menghapus database maupun data atau record yang ada di dalam database. Salah satu contoh penerapannya yaitu menghapus akun yang sudah dibuat sebelumnya.

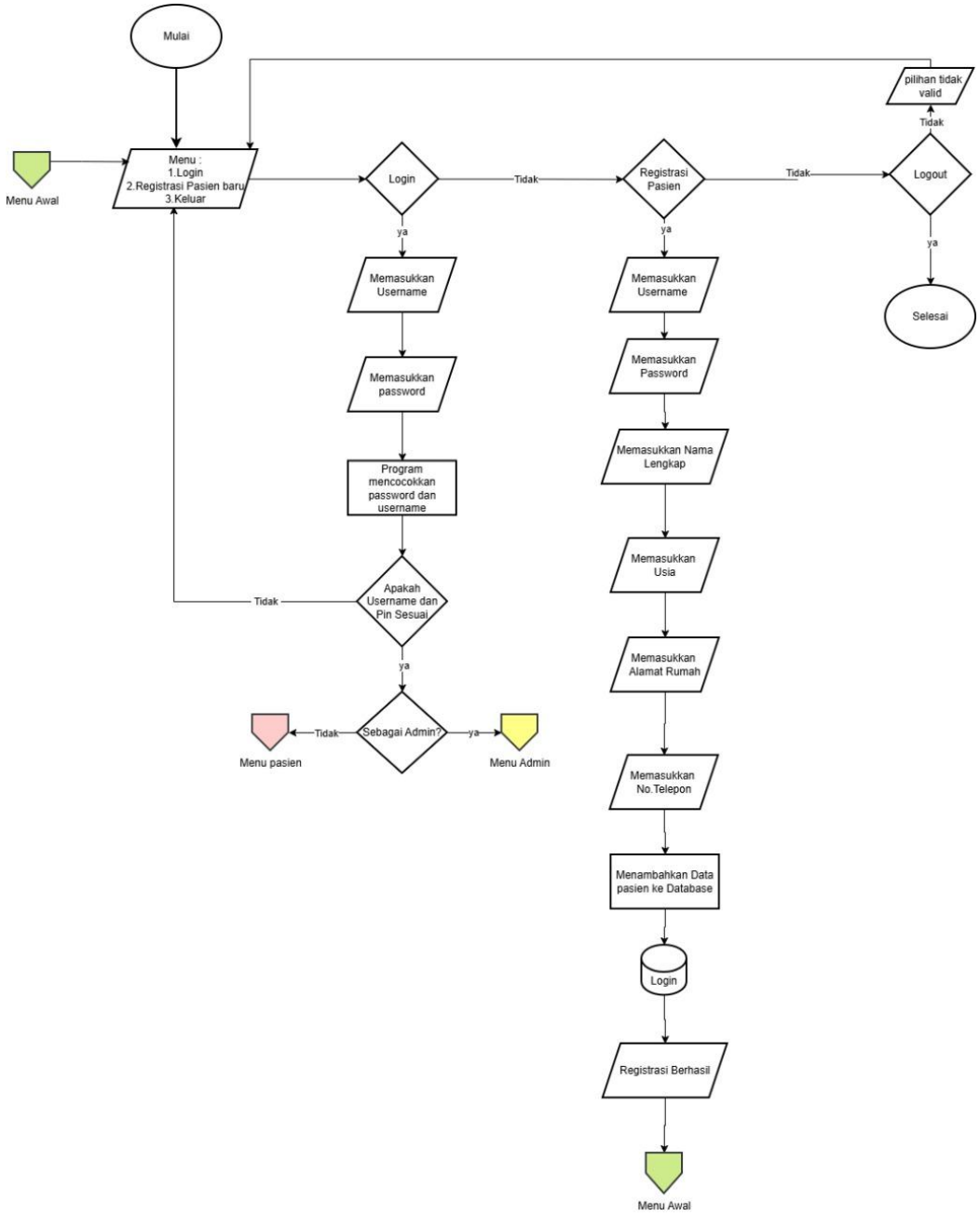
12. Error Handling

```
364  def top_up_balance(username, users):
365      global layanans, riwayat_medis
366      print("\n=== Top Up E-Money ===")
367      print(f"Saldo E-money Anda saat ini: Rp {users[username]['e_money']:,}")
368      while True:
369          try:
370              amount = int(input("Masukkan jumlah top up (Rp): "))
371              if amount <= 0:
372                  print("Jumlah harus positif!")
373                  continue
374              if amount > 5000000:
375                  print("Top up maksimal 5 juta!")
376                  continue
377              break
378          except ValueError:
379              print("Masukkan angka yang valid!")
```

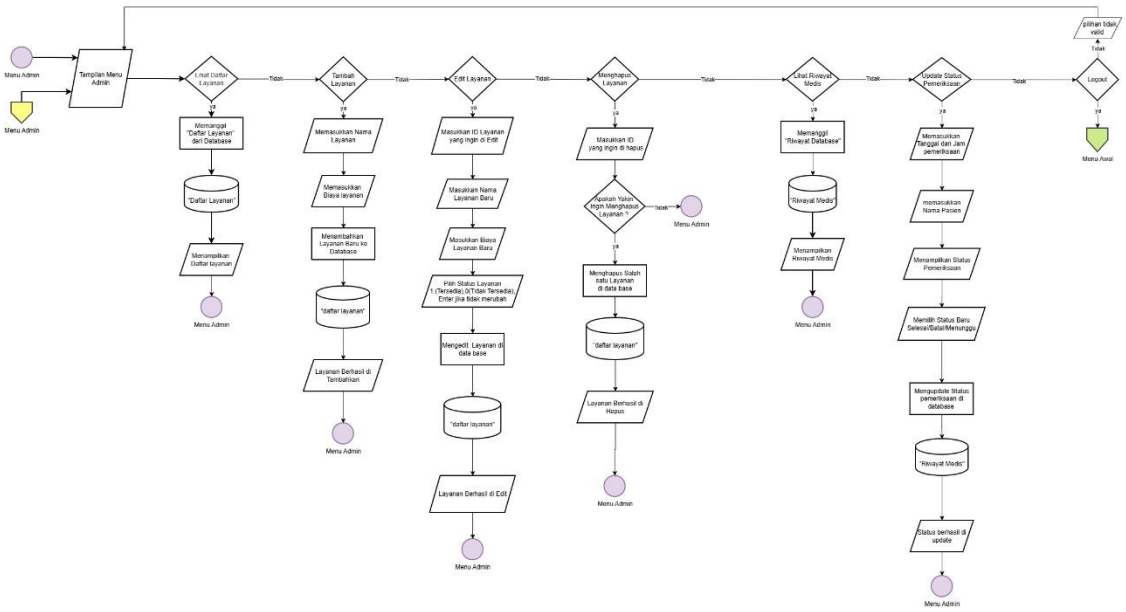
Gambar 2. 13 Penggunaan Error Handling

Error Handling digunakan untuk suatu proses merancang dan mengimplementasikan strategi untuk menangani kesalahan atau kondisi tak terduga yang mungkin terjadi selama eksekusi program.

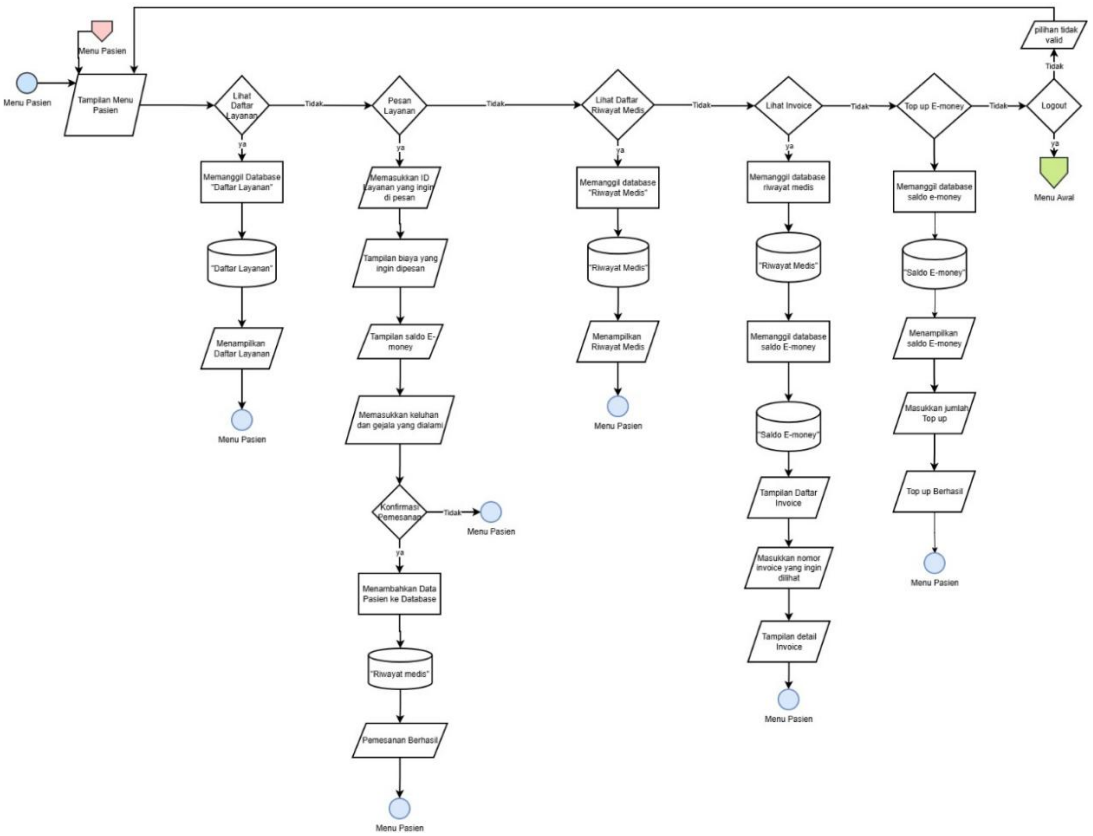
2.2 Flowchart



Gambar 2. 14 Menu Awal



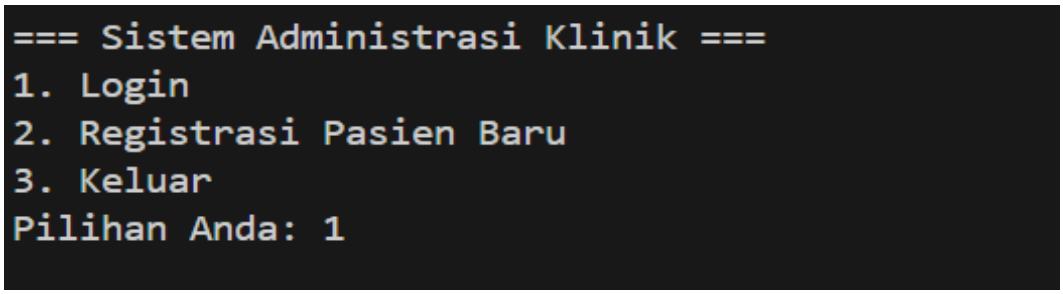
Gambar 2. 15 Menu Admin



Gambar 2. 16 Menu Pasien

BAB III
HASIL DAN PEMBAHASAN

3.1 Implementasi Program



Gambar 3. 1 Output Program

Program ini menampilkan menu utama dari sistem administrasi klinik yang terdiri atas tiga pilihan, yaitu login, registrasi pasien baru, dan keluar (logout). Pada menu login, pengguna dapat masuk menggunakan akun yang sudah terdaftar. Terdapat dua peran pengguna dalam sistem ini, yaitu admin dan pasien (pelanggan). Jika pengguna berperan sebagai pasien namun belum memiliki akun, maka dapat memilih menu kedua, yaitu registrasi pasien baru. Pada proses registrasi, pengguna akan diminta untuk mengisi data seperti username, password, nama lengkap, usia, alamat, dan nomor telepon agar akun dapat dibuat dengan sukses. Sementara itu, admin dapat langsung masuk menggunakan akun yang telah tersedia tanpa perlu melakukan registrasi.

Gambar 3. 2 Output Program

Ketika pengguna login sebagai pasien, akan muncul menu utama yang berisi enam pilihan, yaitu daftar layanan, pesan layanan, riwayat medis, lihat invoice, top

up e-money, dan logout. Pada menu daftar layanan, sistem akan menampilkan berbagai jenis layanan yang tersedia di klinik.

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 1

=== Daftar Layanan Klinik ===
+----+------------------+-----------+----------+
| ID |   Nama Layanan   |   Biaya   |  Status  |
+----+------------------+-----------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
+----+------------------+-----------+----------+
```

Gambar 3. 3 Output Program

daftar layanan klinik pada tampilan tersebut berfungsi untuk menampilkan informasi mengenai berbagai jenis layanan medis yang tersedia di klinik. Tabel ini berisi empat kolom utama, yaitu ID, Nama Layanan, Biaya, dan Status. Kolom ID digunakan sebagai penanda atau nomor urut setiap layanan, sedangkan Nama Layanan menjelaskan jenis pelayanan medis yang dapat dipilih oleh pasien, seperti Konsultasi Umum, Pemeriksaan Gigi, Tes Laboratorium, Vaksinasi, dan Medical Check-up. Kolom Biaya menunjukkan tarif atau harga yang harus dibayar untuk masing-masing layanan, sementara Status menampilkan ketersediaan layanan tersebut, yang dalam tampilan ini seluruhnya berstatus “Tersedia”.

```
Pilihan Anda: 2

=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
+----+------------------+------------+----------+

Masukkan ID layanan yang diinginkan: 5

Biaya layanan: Rp 600,000
Saldo E-money Anda: Rp 1,050,000
Keluhan/Gejala: cek kesehatan
Konfirmasi pemesanan? (y/n): y
Transaksi berhasil!
```

Gambar 3. 4 Output Program

Menu pesan layanan memungkinkan pasien untuk memesan layanan terlebih dahulu tanpa harus menunggu antrean. Di dalam menu ini, akan ditampilkan daftar layanan beserta ID-nya. Pasien dapat memilih layanan yang diinginkan dengan memasukkan ID tersebut. Setelah itu, sistem akan menampilkan biaya layanan, saldo e-money pasien, serta meminta pasien untuk mengisi keluhan atau gejala dan melakukan konfirmasi pemesanan. Setelah proses selesai, sistem otomatis menampilkan invoice pembayaran.

```
Pilihan Anda: 3

=== Riwayat Medis ===
+---------------------+--------+------------------+--------------+------------+---------+
|      Tanggal        | Pasien |     Layanan      |    Keluhan   |   Biaya    | Status  |
+---------------------+--------+------------------+--------------+------------+---------+
| 2025-10-26 17:03:58 | Farah  | Medical Check-up | cek kesehatan | Rp 600,000 | Selesai |
+---------------------+--------+------------------+--------------+------------+---------+
```

Gambar 3. 5 Output Program

Selanjutnya, pada menu riwayat medis, pasien dapat melihat daftar layanan yang telah dipesan dan diselesaikan. Menu lihat invoice menampilkan detail transaksi seperti tanggal, jenis layanan, dan biaya yang telah dilakukan. Melalui menu top up e-money, pasien dapat menambah saldo dengan memasukkan jumlah

uang yang ingin diisikan, dan sistem akan mengonfirmasi keberhasilan top up tersebut. Terakhir, menu logout digunakan untuk keluar dari program.

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 3

=== Riwayat Medis ===
+---------------------+--------+-----------------+--------------+------------+---------+
|       Tanggal       | Pasien |     Layanan     |    Keluhan   |   Biaya    |  Status |
+---------------------+--------+-----------------+--------------+------------+---------+
| 2025-10-26 17:03:58 | Farah  | Medical Check-up | cek kesehatan | Rp 600,000 | Selesai |
+---------------------+--------+-----------------+--------------+------------+---------+
```

Gambar 3. 6 Output Program

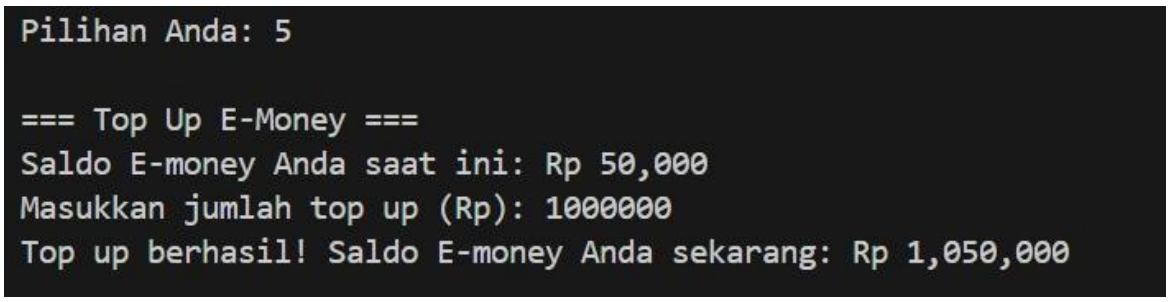
fitur Riwayat Medis ini berfungsi untuk mencatat dan menampilkan rekam medis pasien secara otomatis setelah pelayanan dilakukan. Dengan adanya fitur ini, pasien dan pihak klinik dapat dengan mudah menelusuri layanan yang telah digunakan serta memantau perkembangan kondisi kesehatan pasien dari waktu ke waktu.

```
6. Logout
Pilihan Anda: 4

=== Daftar Invoice ===

1. Tanggal: 2025-10-26 17:03:58 - Layanan: Medical Check-up - Biaya: Rp 600,000

Masukkan nomor invoice yang ingin dilihat (atau Enter untuk kembali): 1

==================================================
               INVOICE PEMBAYARAN
==================================================
Tanggal: 2025-10-26 17:03:58
Pasien: Farah
Layanan: Medical Check-up
Biaya: Rp 600,000
Saldo E-money sebelum: Rp 1,050,000
Dibayar: Rp 600,000
Saldo E-money setelah: Rp 450,000
==================================================
Terima kasih telah menggunakan layanan kami!
==================================================
```

Gambar 3. 7 Output Program

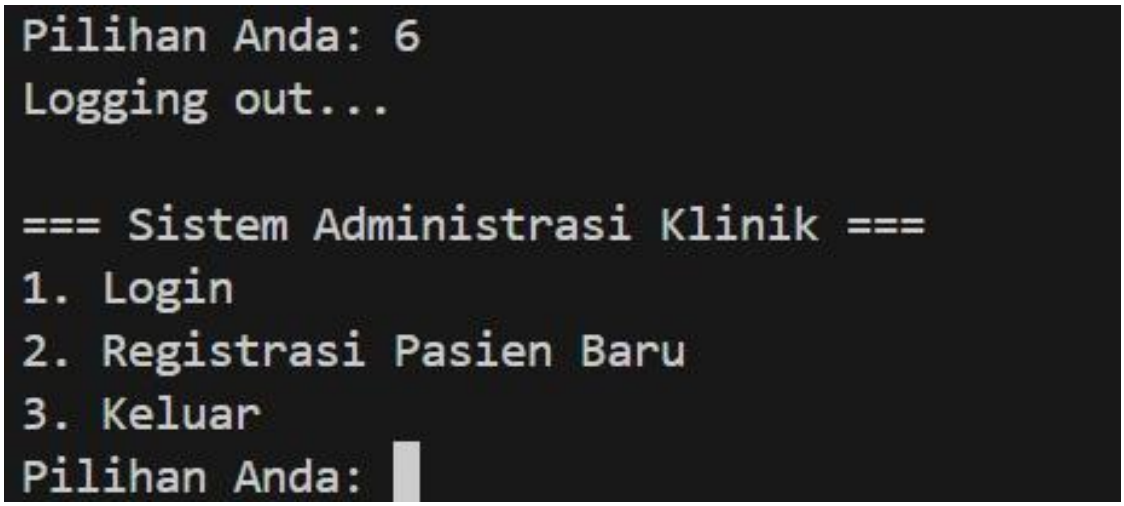
program menampilkan daftar invoice yang tersedia, dan dalam kasus ini hanya ada satu invoice dengan layanan *Medical Check-up* seharga Rp 600.000. Pengguna kemudian mengetik angka daftar invoice untuk melihat detail invoice tersebut. Program lalu menampilkan rincian lengkap pembayaran, termasuk tanggal transaksi, nama pasien Farah, jenis layanan, biaya, serta saldo e-money sebelum dan

sesudah pembayaran. Setelah itu, sistem menampilkan pesan penutup yang berisi ucapan terima kasih karena telah menggunakan layanan tersebut



Gambar 3. 8 Output Program

program menampilkan saldo e-money pengguna saat ini sebesar misal Rp 50.000, lalu meminta pengguna untuk memasukkan jumlah uang yang ingin ditambahkan. Pengguna mengetik jumlah top up sebesar Rp 1.000.000. Setelah proses top up dilakukan, program menampilkan pesan bahwa top up berhasil, dan saldo e-money pengguna kini bertambah menjadi Rp 1.050.000. Dengan demikian, program berjalan dengan urutan menampilkan saldo awal, menerima input jumlah top up, memproses penambahan saldo, lalu menampilkan saldo akhir setelah top up berhasil.



Gambar 3. 9 output Program

sistem menampilkan pesan “Logging out...” yang menandakan bahwa proses keluar dari akun sedang dilakukan. Setelah logout berhasil, program secara otomatis menampilkan kembali menu utama sistem administrasi klinik.

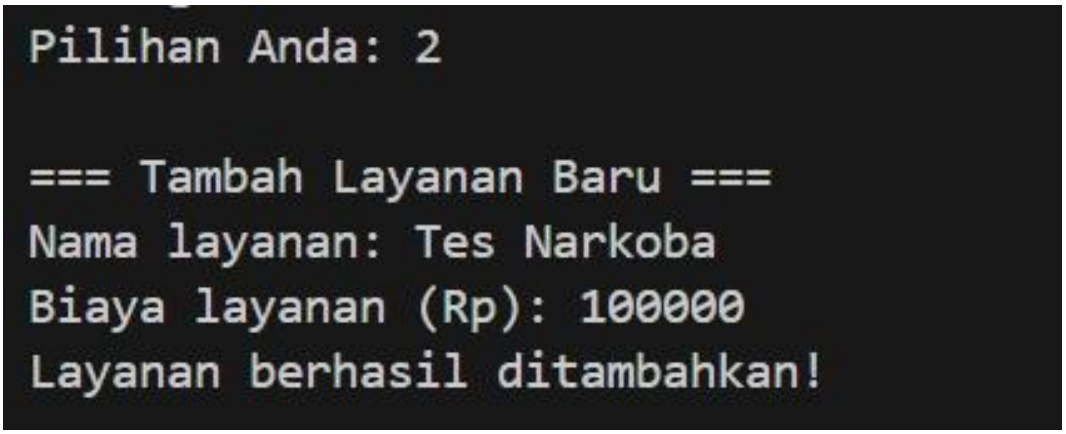
Gambar 3. 10 output Program

Ketika login sebagai admin, sistem akan menampilkan menu utama yang terdiri dari beberapa pilihan, yaitu melihat daftar layanan, menambah layanan, mengedit layanan, menghapus layanan, melihat seluruh riwayat medis pasien, memperbarui status pemeriksaan, dan logout.

Gambar 3. 11 Output Program

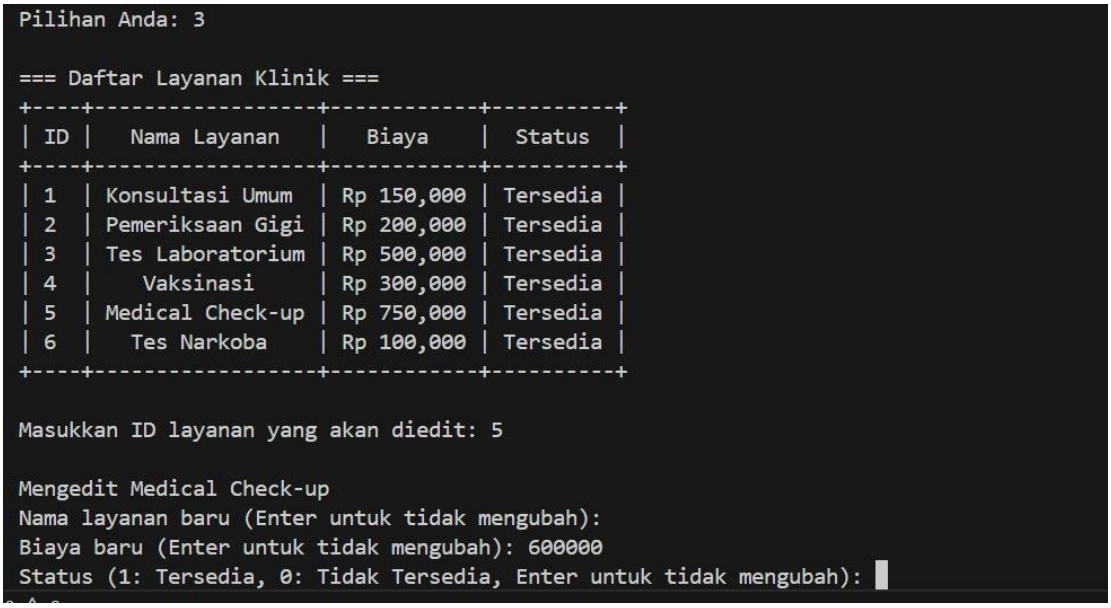
Pada menu daftar layanan, admin dapat melihat berbagai jenis layanan yang tersedia di klinik. Menu tambah layanan digunakan untuk menambahkan layanan baru dengan cara memasukkan nama layanan dan biaya layanan yang akan disediakan. Selanjutnya, menu edit layanan berfungsi untuk memperbarui data layanan yang sudah ada. Admin cukup memasukkan ID layanan yang ingin diubah, kemudian

dapat memperbarui nama, biaya, atau status ketersediaan layanan (tersedia/tidak tersedia). Jika ada data yang tidak ingin diubah, admin cukup menekan *Enter* untuk melewatinya.



Gambar 3. 12 Output Program

alur program ini menunjukkan proses sederhana namun terstruktur di mana admin dapat menambahkan layanan medis baru ke dalam sistem klinik dengan menginput nama dan biaya layanan melalui antarmuka berbasis teks.



Gambar 3. 13 Output Program

program menampilkan daftar seluruh layanan klinik dalam bentuk tabel yang berisi informasi seperti ID layanan, nama layanan, biaya, dan status ketersediaan.

```
Pilihan Anda: 4

=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
| 6  |   Tes Narkoba    | Rp 100,000 | Tersedia |
+----+------------------+------------+----------+

Masukkan ID layanan yang akan dihapus: 6
Anda yakin ingin menghapus layanan Tes Narkoba? (y/n): y
Layanan berhasil dihapus!
```

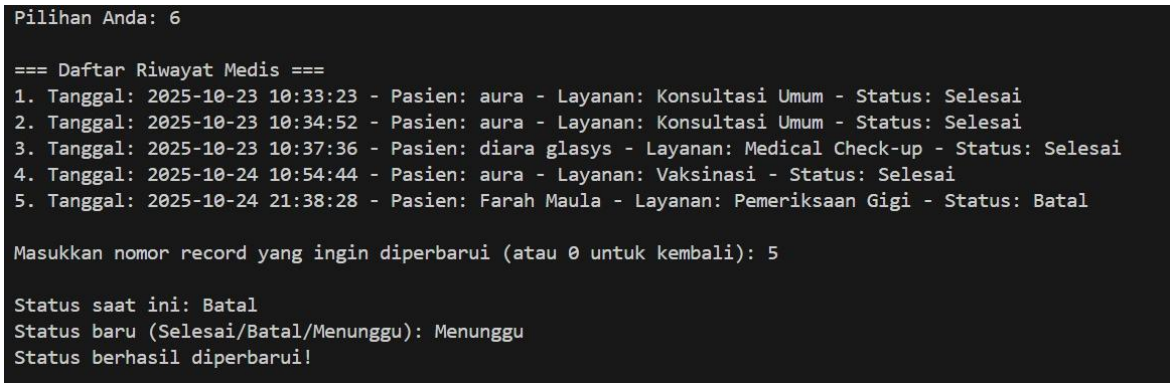
Gambar 3. 14 Output Program

Menu hapus layanan memungkinkan admin menghapus layanan tertentu dengan memasukkan ID layanan yang ingin dihapus.

```
Pilihan Anda: 5

=== Riwayat Medis ===
+---------------------+-------------+-----------------+-----------+------------+---------+
|       Tanggal       |    Pasien   |     Layanan     | Keluhan   |   Biaya    | Status  |
+---------------------+-------------+-----------------+-----------+------------+---------+
| 2025-10-23 10:33:23 |     aura    | Konsultasi Umum |   demam   | Rp 150,000 | Selesai |
| 2025-10-23 10:34:52 |     aura    | Konsultasi Umum |    Mati   | Rp 150,000 | Selesai |
| 2025-10-23 10:37:36 | diara glasys| Medical Check-up|   darah   | Rp 750,000 | Selesai |
| 2025-10-24 10:54:44 |     aura    |    Vaksinasi    |   covid   | Rp 300,000 | Selesai |
| 2025-10-24 21:38:28 | Farah Maula | Pemeriksaan Gigi| sakit gigi| Rp 200,000 |  Batal  |
+---------------------+-------------+-----------------+-----------+------------+---------+
```

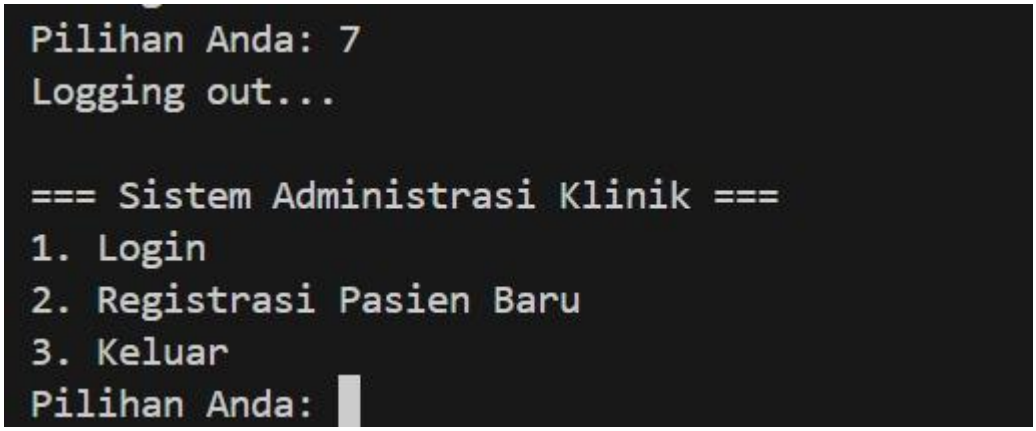
Gambar 3. 15 Output Program

Melalui menu riwayat medis, admin dapat melihat seluruh catatan pemeriksaan pasien yang telah tercatat di sistem.



Gambar 3. 16 Output Program

sistem menampilkan daftar riwayat medis pasien yang berisi informasi seperti tanggal pemeriksaan, nama pasien, jenis layanan, dan status pemeriksaan (misalnya: *Selesai*, *Batal*, atau *Menunggu*). Admin kemudian diminta untuk memasukkan nomor record yang ingin diperbarui. Admin memilih record nomor lima, yang sebelumnya memiliki status *Batal*. Program kemudian menampilkan status saat ini dan meminta admin untuk memasukkan status baru. Admin mengganti status dari *Batal* menjadi *Menunggu*. Setelah perubahan dikonfirmasi, sistem menampilkan pesan “Status berhasil diperbarui!” sebagai tanda bahwa data riwayat medis pasien telah diperbarui dengan sukses.



Gambar 3. 17 Output Program

menu logout digunakan untuk keluar dari menu admin lalu Kembali ke menu awal

3.2 Alur Program

```python
import csv
import os
from prettytable import PrettyTable
import pwinput
from datetime import datetime
```

Gambar 3. 18 Alur Program

Pada alur program pertama, kami mengimport csv, os, pretty table, pwinput, dan date time. Import csv digunakan untuk memanggil sebuah data yang telah disimpan disuatu file. Import os berfungsi memuat modul operating system (OS) Python, yang menyediakan berbagai fungsi untuk berinteraksi dengan sistem operasi tempat Python berjalan, seperti mengelola berkas dan direktori. Import pretty table digunakan untuk membuat table agar menjadi lebih rapi. Import pwinput digunakan untuk menyamarkan password yang akan diinput. Import date time digunakan untuk tanggal dan waktu.

layanan.csv

```
id,name,price,availability
1,Konsultasi Umum,150000,True
2,Pemeriksaan Gigi,200000,True
3,Tes Laboratorium,500000,True
4,Vaksinasi,300000,True
5,Medical Check-up,600000,True
```

Gambar 3. 19 File CSV

riwayat_medis.csv

```
date,username,patient_name,service_id,service_name,price,symptoms,status,balance_before,balance_after
2025-10-23 10:33:23,aura,aura,1,Konsultasi Umum,150000,demam,Selesai,1000000,850000
2025-10-23 10:34:52,aura,aura,1,Konsultasi Umum,150000,Mati,Selesai,850000,700000
2025-10-23 10:37:36,Diara,diara glasys,5,Medical Check-up,750000,darah,Selesai,50000000,49250000
2025-10-24 10:54:44,aura,aura,4,Vaksinasi,300000,covid,Selesai,1500000,1200000
2025-10-24 21:38:28,farah,Farah Maula,2,Pemeriksaan Gigi,200000,sakit gigi,Menunggu,250000,50000
2025-10-26 17:03:58,Farah,Farah,5,Medical Check-up,600000,cek kesehatan,Selesai,1050000,450000
```

Gambar 3. 20 File CSV

users.csv

```
username,password,role,full_name,age,address,phone,e_money
admin,admin123,admin,Administrator,,,,0
aura,aura123,patient,aura,18,suwandi,0812345,1500000
Diara,diara123,patient,diara glasys,18,samarinda,082252238904,49250000
Farah,farah12345,patient,Farah,17,damanhuri,081256291051,50000
```

Gambar 3. 21 File CSV

Pada program ini digunakan untuk memanggil dan mengelola file csv sebagai tempat penyimpanan data utama dalam program.

```python
def initialize_data():
    if not os.path.exists(USERS_FILE):
        with open(USERS_FILE, "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(["username", "password", "role", "full_name", "age", "address", "phone", "e_money"])
            writer.writerow(["admin", "admin123", "admin", "Administrator", "", "", "", "0"])

    if not os.path.exists(SERVICES_FILE):
        with open(SERVICES_FILE, "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(["id", "name", "price", "availability"])
            initial_services = [
                ["1", "Konsultasi Umum", "150000", "True"],
                ["2", "Pemeriksaan Gigi", "200000", "True"],
                ["3", "Tes Laboratorium", "500000", "True"],
                ["4", "Vaksinasi", "300000", "True"],
                ["5", "Medical Check-up", "750000", "True"]
            ]
            writer.writerows(initial_services)

    if not os.path.exists(RIWAYAT_MEDIS_FILE):
        with open(RIWAYAT_MEDIS_FILE, "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(["date", "username", "patient_name", "service_id", "service_name", "price", "symptoms", "status",
                             "balance_before", "balance_after"])
```

Gambar 3. 22 Alur Program

```python
def load_data():
    users = {}
    layanans = {}
    riwayat_medis = []

    with open(USERS_FILE, "r", newline="") as f:
        reader = csv.DictReader(f)
        for row in reader:
            users[row["username"]] = {
                "password": row["password"],
                "role": row["role"],
                "full_name": row["full_name"],
                "age": row["age"],
                "address": row["address"],
                "phone": row["phone"],
                "e_money": int(row.get("e_money", 0))
            }

    with open(SERVICES_FILE, "r", newline="") as f:
        reader = csv.DictReader(f)
        for row in reader:
            layanans[row["id"]] = {
                "name": row["name"],
                "price": int(row["price"]),
                "availability": row["availability"].lower() == "true"
            }
```

Gambar 3. 23 Alur Program

```
63
64  with open(RIWAYAT_MEDIS_FILE, "r", newline="") as f:
65      reader = csv.DictReader(f)
66      riwayat_medis = [
67          {
68              "date": row["date"],
69              "username": row["username"],
70              "patient_name": row["patient_name"],
71              "service_id": row["service_id"],
72              "service_name": row["service_name"],
73              "price": int(row["price"]),
74              "symptoms": row["symptoms"],
75              "status": row["status"],
76              "balance_before": int(row.get("balance_before", 0)),
77              "balance_after": int(row.get("balance_after", 0))
78          }
79          for row in reader
80      ]
81
82  return users, layanans, riwayat_medis
```

Gambar 3. 24 Alur Program

Pada function initialize_data dan load_data digunakan untuk proses menyimpan dan memanggil data user agar singkron dengan data yang ada di csv.

```
518 if __name__ == "__main__":
519     initialize_data()
520     users, layanans, riwayat_medis = load_data()
521
522     while True:
523         print("\n=== Sistem Administrasi Klinik ===")
524         print("1. Login")
525         print("2. Registrasi Pasien Baru")
526         print("3. Keluar")
527
528         choice = input("Pilihan Anda: ")
529
530         if choice == "1":
531             username = login(users)
532             if username:
533                 if users[username]["role"] == "admin":
534                     admin_menu(username)
535                 else:
536                     patient_menu(username)
537         elif choice == "2":
538             register(users)
539         elif choice == "3":
540             print("Terima kasih telah menggunakan sistem kami!")
541             break
542         else:
543             print("Pilihan tidak valid!")
```

Gambar 3. 25 Alur Program

Pada program ini bisa login dengan 2 role yaitu sebagai admin dan user (pasien). Apabila sebelumnya sebagai user belum pernah membuat akun bisa melakukan registrasi di pilihan kedua.

```python
def register(users):
    global layanans, riwayat_medis
    print("\n=== Register Pasien Baru ===")
    while True:
        username = input("Masukkan username (3-20 karakter, alfanumerik): ")
        if not username.isalnum() or not (3 <= len(username) <= 20):
            print("Username harus alfanumerik dan 3-20 karakter!")
            continue
        if username in users:
            print("Username sudah terdaftar!")
            continue
        break

    password = pwinput.pwinput(prompt="Masukkan password: ")
    full_name = input("Nama Lengkap: ")
    while True:
        age = input("Usia (angka): ")
        if not age.isdigit():
            print("Usia harus angka!")
            continue
        age = int(age)
        break
    address = input("Alamat: ")
    while True:
        phone = input("No. Telepon (maks 15 digit): ")
        if not phone.isdigit() or len(phone) > 15:
            print("No. Telepon harus angka dan maksimal 15 digit!")
            continue
        break
```

Gambar 3. 26 Alur Program Regis

```python

    while True:
        try:
            initial_balance = int(input("Saldo awal E-Money (Rp): "))
            if initial_balance < 0:
                print("Saldo awal tidak boleh negatif!")
                continue
            break
        except ValueError:
            print("Masukkan angka yang valid!")

    users[username] = {
        "password": password,
        "role": "patient",
        "full_name": full_name,
        "age": age,
        "address": address,
        "phone": phone,
        "e_money": initial_balance
    }
    save_data(users, layanans, riwayat_medis)
    print("Registrasi berhasil!")
    return username
```

Gambar 3. 27 Alur Program Regis

```
=== Sistem Administrasi Klinik ===
1. Login
2. Registrasi Pasien Baru
3. Keluar
Pilihan Anda: 2

=== Register Pasien Baru ===
Masukkan username (3-20 karakter, alfanumerik): Farah
Masukkan password: **********
Nama Lengkap: Farah
Usia (angka): 17
Alamat: damanhuri
No. Telepon (maks 15 digit): 081256291051
Saldo awal E-Money (Rp): 50000
Registrasi berhasil!
```

Gambar 3. 28 Output Program

Saat melakukan registrasi harus mengisi username, password, nama lengkap, usia,alamat, no telepon, saldo awal e-money.

```python
def login(users):
    print("\n=== Login ===")
    username = input("Masukkan username: ")
    password = pwinput.pwinput(prompt="Masukkan password: ")

    if username in users and users[username]["password"] == password:
        print(f"Selamat datang, {users[username].get('full_name', username)}!")
        return username
    else:
        print("Username atau password salah!")
        return None
```

Gambar 3. 29 Alur Program Login

```
=== Sistem Administrasi Klinik ===
1. Login
2. Registrasi Pasien Baru
3. Keluar
Pilihan Anda: 1

=== Login ===
Masukkan username: Farah
Masukkan password: **********
Selamat datang, Farah!

=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda:
```

Gambar 3. 30 Output Program Login

Setelah melakukan registrasi bisa langsung login dengan menggunakan username dan password yang telah dibuat.

```python
def patient_menu(username):
    global layanans, riwayat_medis, users
    try:
        while True:
            print("\n=== Menu Pasien ===")
            print("1. Lihat Daftar Layanan")
            print("2. Pesan Layanan")
            print("3. Lihat Riwayat Medis")
            print("4. Lihat Invoice")
            print("5. Top Up E-Money")
            print("6. Logout")

            choice = input("Pilihan Anda: ")

            if choice == "1":
                display_layanans(layanans)
            elif choice == "2":
                book_service(username, layanans, riwayat_medis)
            elif choice == "3":
                view_medical_records(username, riwayat_medis)
            elif choice == "4":
                view_invoices(username, riwayat_medis)
            elif choice == "5":
                top_up_balance(username, users)
            elif choice == "6":
                print("Logging out...")
                break
```

Gambar 3. 31 Alur Program Menu Pasien

```
508                 else:
509                     print("Pilihan tidak valid!")
510         except KeyboardInterrupt:
511             print("\n\nProgram dihentikan oleh pengguna (Ctrl+C).")
512             exit(0)
513         except Exception as e:
514             print(f"\nTerjadi kesalahan: {e}")
515             print("Program akan keluar.")
516             exit(1)
```

Gambar 3. 32 Alur Program Menu Pasien

```
193 def display_layanans(layanans):
194     if not layanans:
195         print("Tidak ada layanan tersedia.")
196         return
197 
198     table = PrettyTable()
199     table.field_names = ["ID", "Nama Layanan", "Biaya", "Status"]
200     for sid, layanan in layanans.items():
201         status = "Tersedia" if layanan["availability"] else "Tidak Tersedia"
202         table.add_row([sid, layanan["name"], f"Rp {layanan['price']:,}", status])
203     print("\n=== Daftar Layanan Klinik ===")
204     print(table)
```

Gambar 3. 33 Alur Progran Daftar Layanan

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 1

=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
+----+------------------+------------+----------+
```

Gambar 3. 34 Output Program

Pada menu 1 memperlihatkan daftar layanan yang tersedia.

```python
def book_service(username, layanans, riwayat_medis):
    global users
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang diinginkan: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return

    service = layanans[sid]
    if not service["availability"]:
        print("Layanan tidak tersedia saat ini!")
        return

    print(f"\nBiaya layanan: Rp {service['price']:,}")
    print(f"Saldo E-money Anda: Rp {users[username]['e_money']:,}")
    symptoms = input("Keluhan/Gejala: ")
    confirm = input("Konfirmasi pemesanan? (y/n): ")

    if confirm.lower() == 'y':
        if users[username]["e_money"] >= service["price"]:
            users[username]["e_money"] -= service["price"]
            status = "Selesai"
            print("Transaksi berhasil!")
        else:
            print("Saldo E-money tidak cukup!")
            return
```

Gambar 3. 35 Alur Program Pesan Layanan

```python

        balance_before = users[username]["e_money"] + service["price"]
        balance_after = users[username]["e_money"]

        record = {
            "date": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
            "username": username,
            "patient_name": users[username]["full_name"],
            "service_id": sid,
            "service_name": service["name"],
            "price": service["price"],
            "symptoms": symptoms,
            "status": status,
            "balance_before": balance_before,
            "balance_after": balance_after
        }
        riwayat_medis.append(record)
        save_data(users, layanans, riwayat_medis)

```

Gambar 3. 36 Alur Program Pesan Layanan

```
312
313         # Display invoice
314         print("\n" + "="*50)
315         print("                INVOICE PEMBAYARAN")
316         print("="*50)
317         print(f"Tanggal: {record['date']}")
318         print(f"Pasien: {record['patient_name']}")
319         print(f"Layanan: {record['service_name']}")
320         print(f"Biaya: Rp {record['price']:,}")
321         print(f"Saldo E-money sebelum: Rp {users[username]['e_money'] + service['price']:,}")
322         print(f"Dibayar: Rp {service['price']:,}")
323         print(f"Saldo E-money setelah: Rp {users[username]['e_money']:,}")
324         print("="*50)
325         print("Terima kasih telah menggunakan layanan kami!")
326         print("="*50)
```

Gambar 3. 37 Alur Program Pesan Layanan

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 2

=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
+----+------------------+------------+----------+

Masukkan ID layanan yang diinginkan: 5

Biaya layanan: Rp 600,000
Saldo E-money Anda: Rp 1,050,000
Keluhan/Gejala: cek kesehatan
Konfirmasi pemesanan? (y/n): y
Transaksi berhasil!
```

Gambar 3. 38 Output Program Pesan Layanan

Menu kedua pasien dapat membuat pesanan layanan dengan memilih ID nya lalu memasukkan.

```python
def view_medical_records(username, riwayat_medis, is_admin=False):
    table = PrettyTable()
    table.field_names = ["Tanggal", "Pasien", "Layanan", "Keluhan", "Biaya", "Status"]

    filtered_records = riwayat_medis if is_admin else [r for r in riwayat_medis if r["username"] == username]

    if not filtered_records:
        print("Tidak ada riwayat medis.")
        return

    for r in filtered_records:
        table.add_row([
            r["date"],
            r["patient_name"],
            r["service_name"],
            r["symptoms"],
            f"Rp {r['price']:,}",
            r["status"]
        ])

    print("\n=== Riwayat Medis ===")
    print(table)
```

Gambar 3. 39 Alur Program Lihat Riwayat Medis

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 3

=== Riwayat Medis ===
+---------------------+--------+------------------+---------------+------------+---------+
|       Tanggal       | Pasien |     Layanan      |    Keluhan    |   Biaya    | Status  |
+---------------------+--------+------------------+---------------+------------+---------+
| 2025-10-26 17:03:58 | Farah  | Medical Check-up | cek kesehatan | Rp 600,000 | Selesai |
+---------------------+--------+------------------+---------------+------------+---------+
```

Gambar 3. 40 Output Program Riwayat Medis

Menu ketiga pasien dapat melihat riwayat medisnya sendiri.

```python
def view_invoices(username, riwayat_medis):
    filtered_records = [r for r in riwayat_medis if r["username"] == username and r["status"] == "Selesai"]

    if not filtered_records:
        print("Tidak ada invoice yang tersedia.")
        return

    print("\n=== Daftar Invoice ===")
    for i, r in enumerate(filtered_records, 1):
        print(f"\n{i}. Tanggal: {r['date']} - Layanan: {r['service_name']} - Biaya: Rp {r['price']:,}")

    choice = input("\nMasukkan nomor invoice yang ingin dilihat (atau Enter untuk kembali): ")
    if choice == "":
        return
    try:
        idx = int(choice) - 1
        if 0 <= idx < len(filtered_records):
            record = filtered_records[idx]
            print("\n" + "="*50)
            print("               INVOICE PEMBAYARAN")
            print("="*50)
            print(f"Tanggal: {record['date']}")
            print(f"Pasien: {record['patient_name']}")
            print(f"Layanan: {record['service_name']}")
            print(f"Biaya: Rp {record['price']:,}")
            print(f"Saldo E-money sebelum: Rp {record['balance_before']:,}")
            print(f"Dibayar: Rp {record['price']:,}")
            print(f"Saldo E-money setelah: Rp {record['balance_after']:,}")
            print("="*50)
```

Gambar 3. 41 Alur Program Melihat Invoive

```
359             else:
360                 print("Nomor invoice tidak valid!")
361         except ValueError:
362             print("Masukkan angka yang valid!")
```

Gambar 3. 42 Alur Program Melihat Invoice

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 4

=== Daftar Invoice ===

1. Tanggal: 2025-10-26 17:03:58 - Layanan: Medical Check-up - Biaya: Rp 600,000

Masukkan nomor invoice yang ingin dilihat (atau Enter untuk kembali): 1

==================================================
               INVOICE PEMBAYARAN
==================================================
Tanggal: 2025-10-26 17:03:58
Pasien: Farah
Layanan: Medical Check-up
Biaya: Rp 600,000
Saldo E-money sebelum: Rp 1,050,000
Dibayar: Rp 600,000
Saldo E-money setelah: Rp 450,000
==================================================
Terima kasih telah menggunakan layanan kami!
==================================================
```

Gambar 3. 43 Output Program Invoice

Menu keempat melihat invoice pembayaran yang sudah dilakukan.

```
364 def top_up_balance(username, users):
365     global layanans, riwayat_medis
366     print("\n=== Top Up E-Money ===")
367     print(f"Saldo E-money Anda saat ini: Rp {users[username]['e_money']:,}")
368     while True:
369         try:
370             amount = int(input("Masukkan jumlah top up (Rp): "))
371             if amount <= 0:
372                 print("Jumlah harus positif!")
373                 continue
374             if amount > 5000000:
375                 print("Top up maksimal 5 juta!")
376                 continue
377             break
378         except ValueError:
379             print("Masukkan angka yang valid!")
```

Gambar 3. 44 Alur Program Top Up

```
=== Menu Pasien ===
1. Lihat Daftar Layanan
2. Pesan Layanan
3. Lihat Riwayat Medis
4. Lihat Invoice
5. Top Up E-Money
6. Logout
Pilihan Anda: 5

=== Top Up E-Money ===
Saldo E-money Anda saat ini: Rp 50,000
Masukkan jumlah top up (Rp): 1000000
Top up berhasil! Saldo E-money Anda sekarang: Rp 1,050,000
```

Gambar 3. 45 Output Program Top Up

Menu kelima pasien dapat melakukan top up e-money.

```
441  def admin_menu(username):
442      global layanans, riwayat_medis, users
443      try:
444          while True:
445              print("\n=== Menu Admin ===")
446              print("1. Lihat Daftar Layanan")
447              print("2. Tambah Layanan")
448              print("3. Edit Layanan")
449              print("4. Hapus Layanan")
450              print("5. Lihat Semua Riwayat Medis")
451              print("6. Update Status Pemeriksaan")
452              print("7. Logout")
453
454              choice = input("Pilihan Anda: ")
455
456              if choice == "1":
457                  display_layanans(layanans)
458              elif choice == "2":
459                  add_service(layanans)
460              elif choice == "3":
461                  edit_service(layanans)
462              elif choice == "4":
463                  delete_service(layanans)
464              elif choice == "5":
465                  view_medical_records(username, riwayat_medis, True)
466              elif choice == "6":
467                  update_record_status(riwayat_medis)
468              elif choice == "7":
```

Gambar 3. 46 Alur Program Admin

```
469                  print("Logging out...")
470                  break
471              else:
472                  print("Pilihan tidak valid!")
473      except KeyboardInterrupt:
474          print("\n\nProgram dihentikan oleh pengguna (Ctrl+C).")
475          exit(0)
476      except Exception as e:
477          print(f"\nTerjadi kesalahan: {e}")
478          print("Program akan keluar.")
479          exit(1)
```

Gambar 3. 47 Alur Program Admin

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda:
```

Gambar 3. 48 Output Program Admin

Ini adalah menu admin.

Pada daftar layanan sama seperti daftar layanan yang ada di menu pasien.

```
206  def add_service(layanans):
207      global users, riwayat_medis
208      print("\n=== Tambah Layanan Baru ===")
209      name = input("Nama layanan: ")
210      while True:
211          try:
212              price = int(input("Biaya layanan (Rp): "))
213              break
214          except ValueError:
215              print("Masukkan angka yang valid!")
216
217      sid = str(max(map(int, layanans.keys())) + 1) if layanans else "1"
218      layanans[sid] = {
219          "name": name,
220          "price": price,
221          "availability": True
222      }
223      save_data(users, layanans, riwayat_medis)
224      print("Layanan berhasil ditambahkan!")
```

Gambar 3. 49 Alur Program Tambah Layanan

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda: 2

=== Tambah Layanan Baru ===
Nama layanan: Tes Narkoba
Biaya layanan (Rp): 100000
Layanan berhasil ditambahkan!
```

Gambar 3. 50 Output Program Tambah Layanan

Admin dapat menambahkan layanan pada menu ke 2.

```
226  def edit_service(layanans):
227      global users, riwayat_medis
228      display_layanans(layanans)
229      sid = input("\nMasukkan ID layanan yang akan diedit: ")
230      if sid not in layanans:
231          print("Layanan tidak ditemukan!")
232          return
233
234      print(f"\nMengedit {layanans[sid]['name']}")
235      name = input("Nama layanan baru (Enter untuk tidak mengubah): ")
236      price_str = input("Biaya baru (Enter untuk tidak mengubah): ")
237      avail_str = input("Status (1: Tersedia, 0: Tidak Tersedia, Enter untuk tidak mengubah): ")
238
239      if name:
240          layanans[sid]["name"] = name
241      if price_str:
242          try:
243              layanans[sid]["price"] = int(price_str)
244          except ValueError:
245              print("Biaya tidak valid! Menggunakan biaya lama.")
246      if avail_str in ['0', '1']:
247          layanans[sid]["availability"] = bool(int(avail_str))
248
249      save_data(users, layanans, riwayat_medis)
250      print("Layanan berhasil diperbarui!")
```

Gambar 3. 51 Alur Program Edit Layanan

```
=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 750,000 | Tersedia |
| 6  |   Tes Narkoba    | Rp 100,000 | Tersedia |
+----+------------------+------------+----------+

Masukkan ID layanan yang akan diedit: 5

Mengedit Medical Check-up
Nama layanan baru (Enter untuk tidak mengubah):
Biaya baru (Enter untuk tidak mengubah): 600000
Status (1: Tersedia, 0: Tidak Tersedia, Enter untuk tidak mengubah):
Layanan berhasil diperbarui!
```

Gambar 3. 52 Output Program Edit Layanan

Menu ketiga admin dapat mengedit daftar layanan yang sudah ada.

```python
def delete_service(layanans):
    global users, riwayat_medis
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang akan dihapus: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return

    confirm = input(f"Anda yakin ingin menghapus layanan {layanans[sid]['name']}? (y/n): ")
    if confirm.lower() == 'y':
        del layanans[sid]
        save_data(users, layanans, riwayat_medis)
        print("Layanan berhasil dihapus!")
    else:
        print("Penghapusan dibatalkan.")
```

Gambar 3. 53 Alur Program Hapus Layanan

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda: 4

=== Daftar Layanan Klinik ===
+----+------------------+------------+----------+
| ID |   Nama Layanan   |   Biaya    |  Status  |
+----+------------------+------------+----------+
| 1  | Konsultasi Umum  | Rp 150,000 | Tersedia |
| 2  | Pemeriksaan Gigi | Rp 200,000 | Tersedia |
| 3  | Tes Laboratorium | Rp 500,000 | Tersedia |
| 4  |    Vaksinasi     | Rp 300,000 | Tersedia |
| 5  | Medical Check-up | Rp 600,000 | Tersedia |
| 6  |   Tes Narkoba    | Rp 100,000 | Tersedia |
+----+------------------+------------+----------+

Masukkan ID layanan yang akan dihapus: 6
Anda yakin ingin menghapus layanan Tes Narkoba? (y/n): y
Layanan berhasil dihapus!
```

Gambar 3. 54 Output Program Hapus Layanan

Pada menu keempat admin dapat menghapus layanan yang sudah ada.

```python
def view_medical_records(username, riwayat_medis, is_admin=False):
    table = PrettyTable()
    table.field_names = ["Tanggal", "Pasien", "Layanan", "Keluhan", "Biaya", "Status"]

    filtered_records = riwayat_medis if is_admin else [r for r in riwayat_medis if r["username"] == username]

    if not filtered_records:
        print("Tidak ada riwayat medis.")
        return

    for r in filtered_records:
        table.add_row([
            r["date"],
            r["patient_name"],
            r["service_name"],
            r["symptoms"],
            f"Rp {r['price']:,}",
            r["status"]
        ])

    print("\n=== Riwayat Medis ===")
    print(table)
```

Gambar 3. 55 Alur Program Lihat Riwayat Medis

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda: 5

=== Riwayat Medis ===
+---------------------+--------------+------------------+------------+------------+---------+
|       Tanggal       |    Pasien    |     Layanan      |  Keluhan   |   Biaya    | Status  |
+---------------------+--------------+------------------+------------+------------+---------+
| 2025-10-23 10:33:23 |     aura     | Konsultasi Umum  |   demam    | Rp 150,000 | Selesai |
| 2025-10-23 10:34:52 |     aura     | Konsultasi Umum  |    Mati    | Rp 150,000 | Selesai |
| 2025-10-23 10:37:36 | diara glasys | Medical Check-up |   darah    | Rp 750,000 | Selesai |
| 2025-10-24 10:54:44 |     aura     |    Vaksinasi     |   covid    | Rp 300,000 | Selesai |
| 2025-10-24 21:38:28 | Farah Maula  | Pemeriksaan Gigi | sakit gigi | Rp 200,000 |  Batal  |
+---------------------+--------------+------------------+------------+------------+---------+
```

Gambar 3. 56 Output Lihat Riwayat Medis

Pada menu kelima ini admin dapat melihat semua riwayat medis pasien.

```python
def update_record_status(riwayat_medis):
    global users, layanans
    if not riwayat_medis:
        print("Tidak ada riwayat medis untuk diperbarui.")
        return

    print("\n=== Daftar Riwayat Medis ===")
    for i, r in enumerate(riwayat_medis, 1):
        print(f"{i}. Tanggal: {r['date']} - Pasien: {r['patient_name']} - Layanan: {r['service_name']} - Status: {r['statu

    while True:
        try:
            choice = int(input("\nMasukkan nomor record yang ingin diperbarui (atau 0 untuk kembali): "))
            if choice == 0:
                return
            if 1 <= choice <= len(riwayat_medis):
                record = riwayat_medis[choice - 1]
                break
            else:
                print("Nomor record tidak valid! Masukkan angka antara 1 dan", len(riwayat_medis))
        except ValueError:
            print("Input harus berupa angka! Silakan coba lagi.")

    print("\nStatus saat ini:", record["status"])
    new_status = input("Status baru (Selesai/Batal/Menunggu): ")
```

Gambar 3. 57 Alur Program Update Status Pemeriksaan

```python

    if new_status in ["Selesai", "Batal", "Menunggu"]:
        record["status"] = new_status
        save_data(users, layanans, riwayat_medis)
        print("Status berhasil diperbarui!")
    else:
        print("Status tidak valid!")
```

Gambar 3. 58 Alur Program Update Status Pemeriksaan

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda: 6

=== Daftar Riwayat Medis ===
1. Tanggal: 2025-10-23 10:33:23 - Pasien: aura - Layanan: Konsultasi Umum - Status: Selesai
2. Tanggal: 2025-10-23 10:34:52 - Pasien: aura - Layanan: Konsultasi Umum - Status: Selesai
3. Tanggal: 2025-10-23 10:37:36 - Pasien: diara glasys - Layanan: Medical Check-up - Status: Selesai
4. Tanggal: 2025-10-24 10:54:44 - Pasien: aura - Layanan: Vaksinasi - Status: Selesai
5. Tanggal: 2025-10-24 21:38:28 - Pasien: Farah Maula - Layanan: Pemeriksaan Gigi - Status: Batal

Masukkan nomor record yang ingin diperbarui (atau 0 untuk kembali): 5

Status saat ini: Batal
Status baru (Selesai/Batal/Menunggu): Menunggu
Status berhasil diperbarui!
```

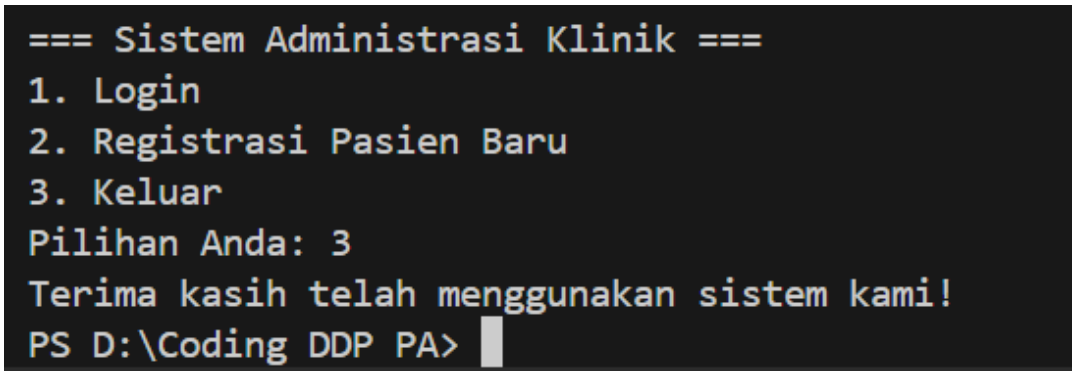
Gambar 3. 59 Output Program Update Status Pemeriksaan

Pada menu keenam ini admin dapat mengubah status riwayat medis pasien seperti selesai/batal/menunggu.

```
=== Menu Admin ===
1. Lihat Daftar Layanan
2. Tambah Layanan
3. Edit Layanan
4. Hapus Layanan
5. Lihat Semua Riwayat Medis
6. Update Status Pemeriksaan
7. Logout
Pilihan Anda: 7
Logging out...

=== Sistem Administrasi Klinik ===
1. Login
2. Registrasi Pasien Baru
3. Keluar
Pilihan Anda:
```

Gambar 3. 60 Output Program Logout

Di menu ketujuh ini admin dapat keluar dari system admin.



Gambar 3. 61 Output Keluar dari Sistem

Di menu utama ini bisa keluar dari system dengan memilih menu 3.

3.3 Source Code

Tabel 3.1 Source Code

```
import csv
import os
from prettytable import PrettyTable
import pwinput
from datetime import datetime

USERS_FILE = "users.csv"
SERVICES_FILE = "layanan.csv"
RIWAYAT_MEDIS_FILE = "riwayat_medis.csv"

def initialize_data():
    if not os.path.exists(USERS_FILE):
        with open(USERS_FILE, "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(["username", "password", "role", "full_name", "age", "address", "phone", "e_money"])
            writer.writerow(["admin", "admin123", "admin", "Administrator", "", "", "", "0"])

    if not os.path.exists(SERVICES_FILE):
        with open(SERVICES_FILE, "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(["id", "name", "price", "availability"])
            initial_services = [
                ["1", "Konsultasi Umum", "150000", "True"],
                ["2", "Pemeriksaan Gigi", "200000", "True"],
                ["3", "Tes Laboratorium", "500000", "True"],
                ["4", "Vaksinasi", "300000", "True"],
                ["5", "Medical Check-up", "750000", "True"]
            ]
            writer.writerows(initial_services)

    if not os.path.exists(RIWAYAT_MEDIS_FILE):
        with open(RIWAYAT_MEDIS_FILE, "w", newline="") as f:
            writer = csv.writer(f)
              writer.writerow(["date", "username", "patient_name", "service_id", "service_name", "price", "symptoms",
"status", "balance_before", "balance_after"])

def load_data():
    users = {}
    layanans = {}
    riwayat_medis = []

    with open(USERS_FILE, "r", newline="") as f:
        reader = csv.DictReader(f)
        for row in reader:
            users[row["username"]] = {
                "password": row["password"],
                "role": row["role"],
                "full_name": row["full_name"],
                "age": row["age"],
```

```
            "address": row["address"],
            "phone": row["phone"],
            "e_money": int(row.get("e_money", 0))
        }

    with open(SERVICES_FILE, "r", newline="") as f:
        reader = csv.DictReader(f)
        for row in reader:
            layanans[row["id"]] = {
                "name": row["name"],
                "price": int(row["price"]),
                "availability": row["availability"].lower() == "true"
            }

    with open(RIWAYAT_MEDIS_FILE, "r", newline="") as f:
        reader = csv.DictReader(f)
        riwayat_medis = [
            {
                "date": row["date"],
                "username": row["username"],
                "patient_name": row["patient_name"],
                "service_id": row["service_id"],
                "service_name": row["service_name"],
                "price": int(row["price"]),
                "symptoms": row["symptoms"],
                "status": row["status"],
                "balance_before": int(row.get("balance_before", 0)),
                "balance_after": int(row.get("balance_after", 0))
            }
            for row in reader
        ]

    return users, layanans, riwayat_medis

def save_data(users, layanans, riwayat_medis):
    with open(USERS_FILE, "w", newline="") as f:
        writer = csv.writer(f)
        writer.writerow(["username", "password", "role", "full_name", "age", "address", "phone", "e_money"])
        for username, data in users.items():
            writer.writerow([
                username,
                data["password"],
                data["role"],
                data["full_name"],
                data["age"],
                data["address"],
                data["phone"],
                data["e_money"]
            ])
```

```
    with open(SERVICES_FILE, "w", newline="") as f:
        writer = csv.writer(f)
        writer.writerow(["id", "name", "price", "availability"])
        for layanan_id, data in layanans.items():
            writer.writerow([
                layanan_id,
                data["name"],
                data["price"],
                str(data["availability"])
            ])

    with open(RIWAYAT_MEDIS_FILE, "w", newline="") as f:
        writer = csv.writer(f)
            writer.writerow(["date",  "username",  "patient_name",  "service_id",  "service_name",  "price",  "symptoms",
"status", "balance_before", "balance_after"])
        for record in riwayat_medis:
            writer.writerow([
                record["date"],
                record["username"],
                record["patient_name"],
                record["service_id"],
                record["service_name"],
                record["price"],
                record["symptoms"],
                record["status"],
                record.get("balance_before", 0),
                record.get("balance_after", 0)
            ])

def register(users):
    global layanans, riwayat_medis
    print("\n=== Register Pasien Baru ===")
    username = input("Masukkan username: ")
    if username in users:
        print("Username sudah terdaftar!")
        return None

    password = pwinput.pwinput(prompt="Masukkan password: ")
    full_name = input("Nama Lengkap: ")
    age = input("Usia: ")
    address = input("Alamat: ")
    phone = input("No. Telepon: ")

    users[username] = {
        "password": password,
        "role": "patient",
        "full_name": full_name,
        "age": age,
        "address": address,
        "phone": phone,
```

```
        "e_money": 0
    }
    save_data(users, layanans, riwayat_medis)
    print("Registrasi berhasil!")
    return username

def login(users):
    print("\n=== Login ===")
    username = input("Masukkan username: ")
    password = pwinput.pwinput(prompt="Masukkan password: ")

    if username in users and users[username]["password"] == password:
        print(f"Selamat datang, {users[username].get('full_name', username)}!")
        return username
    else:
        print("Username atau password salah!")
        return None

def display_layanans(layanans):
    if not layanans:
        print("Tidak ada layanan tersedia.")
        return

    table = PrettyTable()
    table.field_names = ["ID", "Nama Layanan", "Biaya", "Status"]
    for sid, layanan in layanans.items():
        status = "Tersedia" if layanan["availability"] else "Tidak Tersedia"
        table.add_row([sid, layanan["name"], f"Rp {layanan['price']:,}", status])
    print("\n=== Daftar Layanan Klinik ===")
    print(table)

def add_service(layanans):
    global users, riwayat_medis
    print("\n=== Tambah Layanan Baru ===")
    name = input("Nama layanan: ")
    while True:
        try:
            price = int(input("Biaya layanan (Rp): "))
            break
        except ValueError:
            print("Masukkan angka yang valid!")

    sid = str(max(map(int, layanans.keys())) + 1) if layanans else "1"
    layanans[sid] = {
        "name": name,
        "price": price,
        "availability": True
    }
    save_data(users, layanans, riwayat_medis)
    print("Layanan berhasil ditambahkan!")
```

```
def edit_service(layanans):
    global users, riwayat_medis
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang akan diedit: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return

    print(f"\nMengedit {layanans[sid]['name']}")
    name = input("Nama layanan baru (Enter untuk tidak mengubah): ")
    price_str = input("Biaya baru (Enter untuk tidak mengubah): ")
    avail_str = input("Status (1: Tersedia, 0: Tidak Tersedia, Enter untuk tidak mengubah): ")

    if name:
        layanans[sid]["name"] = name
    if price_str:
        try:
            layanans[sid]["price"] = int(price_str)
        except ValueError:
            print("Biaya tidak valid! Menggunakan biaya lama.")
    if avail_str in ['0', '1']:
        layanans[sid]["availability"] = bool(int(avail_str))

    save_data(users, layanans, riwayat_medis)
    print("Layanan berhasil diperbarui!")

def delete_service(layanans):
    global users, riwayat_medis
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang akan dihapus: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return

    confirm = input(f"Anda yakin ingin menghapus layanan {layanans[sid]['name']}? (y/n): ")
    if confirm.lower() == 'y':
        del layanans[sid]
        save_data(users, layanans, riwayat_medis)
        print("Layanan berhasil dihapus!")
    else:
        print("Penghapusan dibatalkan.")

def book_service(username, layanans, riwayat_medis):
    global users
    display_layanans(layanans)
    sid = input("\nMasukkan ID layanan yang diinginkan: ")
    if sid not in layanans:
        print("Layanan tidak ditemukan!")
        return
```

```
service = layanans[sid]
if not service["availability"]:
    print("Layanan tidak tersedia saat ini!")
    return

print(f"\nBiaya layanan: Rp {service['price']:,}")
print(f"Saldo E-money Anda: Rp {users[username]['e_money']:,}")
symptoms = input("Keluhan/Gejala: ")
confirm = input("Konfirmasi pemesanan? (y/n): ")

if confirm.lower() == 'y':
    if users[username]["e_money"] >= service["price"]:
        users[username]["e_money"] -= service["price"]
        status = "Selesai"
        print("Transaksi berhasil!")
    else:
        print("Saldo E-money tidak cukup!")
        return

    balance_before = users[username]["e_money"] + service["price"]
    balance_after = users[username]["e_money"]

    record = {
        "date": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
        "username": username,
        "patient_name": users[username]["full_name"],
        "service_id": sid,
        "service_name": service["name"],
        "price": service["price"],
        "symptoms": symptoms,
        "status": status,
        "balance_before": balance_before,
        "balance_after": balance_after
    }
    riwayat_medis.append(record)
    save_data(users, layanans, riwayat_medis)

    # Display invoice
    print("\n" + "="*50)
    print("              INVOICE PEMBAYARAN")
    print("="*50)
    print(f"Tanggal: {record['date']}")
    print(f"Pasien: {record['patient_name']}")
    print(f"Layanan: {record['service_name']}")
    print(f"Biaya: Rp {record['price']:,}")
    print(f"Saldo E-money sebelum: Rp {users[username]['e_money'] + service['price']:,}")
    print(f"Dibayar: Rp {service['price']:,}")
    print(f"Saldo E-money setelah: Rp {users[username]['e_money']:,}")
    print("="*50)
```

```
        print("Terima kasih telah menggunakan layanan kami!")
        print("="*50)

def view_invoices(username, riwayat_medis):
    filtered_records = [r for r in riwayat_medis if r["username"] == username and r["status"] == "Selesai"]

    if not filtered_records:
        print("Tidak ada invoice yang tersedia.")
        return

    print("\n=== Daftar Invoice ===")
    for i, r in enumerate(filtered_records, 1):
        print(f"\n{i}. Tanggal: {r['date']} - Layanan: {r['service_name']} - Biaya: Rp {r['price']:,}")

    choice = input("\nMasukkan nomor invoice yang ingin dilihat (atau Enter untuk kembali): ")
    if choice.isdigit():
        idx = int(choice) - 1
        if 0 <= idx < len(filtered_records):
            record = filtered_records[idx]
            print("\n" + "="*50)
            print("              INVOICE PEMBAYARAN")
            print("="*50)
            print(f"Tanggal: {record['date']}")
            print(f"Pasien: {record['patient_name']}")
            print(f"Layanan: {record['service_name']}")
            print(f"Biaya: Rp {record['price']:,}")
            print(f"Saldo E-money sebelum: Rp {record['balance_before']:,}")
            print(f"Dibayar: Rp {record['price']:,}")
            print(f"Saldo E-money setelah: Rp {record['balance_after']:,}")
            print("="*50)
            print("Terima kasih telah menggunakan layanan kami!")
            print("="*50)

def top_up_balance(username, users):
    global layanans, riwayat_medis
    print("\n=== Top Up E-Money ===")
    while True:
        try:
            amount = int(input("Masukkan jumlah top up (Rp): "))
            if amount <= 0:
                print("Jumlah harus positif!")
                continue
            break
        except ValueError:
            print("Masukkan angka yang valid!")

    users[username]["e_money"] += amount
    save_data(users, layanans, riwayat_medis)
    print(f"Top up berhasil! Saldo E-money Anda sekarang: Rp {users[username]['e_money']:,}")
```

```
def view_medical_records(username, riwayat_medis, is_admin=False):
    table = PrettyTable()
    table.field_names = ["Tanggal", "Pasien", "Layanan", "Keluhan", "Biaya", "Status"]

    filtered_records = riwayat_medis if is_admin else [r for r in riwayat_medis if r["username"] == username]

    if not filtered_records:
        print("Tidak ada riwayat medis.")
        return

    for r in filtered_records:
        table.add_row([
            r["date"],
            r["patient_name"],
            r["service_name"],
            r["symptoms"],
            f"Rp {r['price']:,}",
            r["status"]
        ])

    print("\n=== Riwayat Medis ===")
    print(table)

def update_record_status(riwayat_medis):
    global users, layanans
    view_medical_records(None, riwayat_medis, True)
    print("\nUpdate Status Pemeriksaan")
    date = input("Masukkan tanggal pemeriksaan (YYYY-MM-DD): ")
    patient = input("Masukkan nama pasien: ").lower()

    found_records = [r for r in riwayat_medis if r["date"].startswith(date + " ") and r["patient_name"].lower() == patient]
    if not found_records:
        print("Record tidak ditemukan!")
        return

    record = found_records[0]
    print("\nStatus saat ini:", record["status"])
    new_status = input("Status baru (Selesai/Batal/Menunggu): ")

    if new_status in ["Selesai", "Batal", "Menunggu"]:
        record["status"] = new_status
        save_data(users, layanans, riwayat_medis)
        print("Status berhasil diperbarui!")
    else:
        print("Status tidak valid!")

def admin_menu(username):
    global layanans, riwayat_medis, users
    while True:
        print("\n=== Menu Admin ===")
```

```
        print("1. Lihat Daftar Layanan")
        print("2. Tambah Layanan")
        print("3. Edit Layanan")
        print("4. Hapus Layanan")
        print("5. Lihat Semua Riwayat Medis")
        print("6. Update Status Pemeriksaan")
        print("7. Logout")

        choice = input("Pilihan Anda: ")

        if choice == "1":
            display_layanans(layanans)
        elif choice == "2":
            add_service(layanans)
        elif choice == "3":
            edit_service(layanans)
        elif choice == "4":
            delete_service(layanans)
        elif choice == "5":
            view_medical_records(username, riwayat_medis, True)
        elif choice == "6":
            update_record_status(riwayat_medis)
        elif choice == "7":
            print("Logging out...")
            break
        else:
            print("Pilihan tidak valid!")

def patient_menu(username):
    global layanans, riwayat_medis, users
    while True:
        print("\n=== Menu Pasien ===")
        print("1. Lihat Daftar Layanan")
        print("2. Pesan Layanan")
        print("3. Lihat Riwayat Medis")
        print("4. Lihat Invoice")
        print("5. Top Up E-Money")
        print("6. Logout")

        choice = input("Pilihan Anda: ")

        if choice == "1":
            display_layanans(layanans)
        elif choice == "2":
            book_service(username, layanans, riwayat_medis)
        elif choice == "3":
            view_medical_records(username, riwayat_medis)
        elif choice == "4":
            view_invoices(username, riwayat_medis)
        elif choice == "5":
```

```
            top_up_balance(username, users)
        elif choice == "6":
            print("Logging out...")
            break
        else:
            print("Pilihan tidak valid!")

if __name__ == "__main__":
    initialize_data()
    users, layanans, riwayat_medis = load_data()

    while True:
        print("\n=== Sistem Administrasi Klinik ===")
        print("1. Login")
        print("2. Registrasi Pasien Baru")
        print("3. Keluar")

        choice = input("Pilihan Anda: ")

        if choice == "1":
            username = login(users)
            if username:
                if users[username]["role"] == "admin":
                    admin_menu(username)
                else:
                    patient_menu(username)
        elif choice == "2":
            register(users)
        elif choice == "3":
            print("Terima kasih telah menggunakan sistem kami!")
            break
        else:
            print("Pilihan tidak valid!")
```

BAB IV
PENUTUP

4.1 Kesimpulan

Dari penjelasan diatas, disimpulakan bahwa program system administrasi dan pembayaran klinik merupakan solusi yang dirancang untuk meningkatkan efisien, efektifitas, keteraturan dalam proses administrasi klinik dan mempermudah proses pelayanan atau pengelolaan data secara digital. Bagi pasien sistem ini memudahkan proses registrasi, pemesanan layanan tanpa harus antre, pengecekan riwayat medis dan lainnya sehingga memberikan kenyamanan dan waktu yang efisien. Sementara bagi admin, sistem ini membantu dalam mengelola layanan klinik sehingga kegiatan operasional klinik menjadi lebih terorganisir dengan baik. Dengan adanya sistem ini klinik dapat memberikan pelayanan yang lebih professional dan terintegrasi antara pasien, admin, dan data administrasi yang tersimpan dengan baik.

4.2 Saran

Agar sistem administrasi dan pembayaran klinik ini dapat berkembang menjadi lebih baik, beberapa saran yang bisa dipertimbangkan diantaranya:

1. Menambahkan fitur notifikasi otomatis kepada pasien untuk mengingatkan terkait jadwal pemeriksaan yang sudah dipesan.
2. Mengembangkan pilihan pembayaran digital seperti qris, dana, ovo, gopay, dan virtual account bank.

Meningkatkan keamanan data dengan menggunakan system autentikasi ganda agar data pasien dan transaksi lebih terjamin.

DAFTAR PUSTAKA

https://youtube.com/shorts/po9mi-gGk1Y?si=5JXghuEC9-Fde01W

LAMPIRAN

Lampiran 1 : Tabel Kontribusi

Nama	Kontribusi	Bagian
Aura Putri Anindita Syarif (2509116094)	Konsep, Coding	1. Referensi konsep program 2. Coding
Nabila Viviana Asri (2509116098)	Flowchart, Laporan	1. Flowchart program 2. Laporan
Farah Hikmatul Maula (2509116099)	Laporan	1. Konsep dan penyusunan Laporan 2. *Finishing* Laporan