Pengembangan Sistem Pencatatan Pasien Terintegrasi Berbasis C di Klinik X

Rafi Ananta Alden
Muhammad Zaki Fazansyah
Mahardhika Putra Adipratama
Mohammad Ari Alexander Aziz
Aira Ardistya Akbarsyah
Program Studi Teknik Elektro
Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung (STEI ITB)
Bandung, Indonesia
13222087@mahasiswa.itb.ac.id
13222093@mahasiswa.itb.ac.id
18322018@mahasiswa.itb.ac.id
13222081@mahasiswa.itb.ac.id
13222015@ mahasiswa.itb.ac.id

Abstract: Sistem Pencatatan Pasien Terintegrasi merupakan sistem pencatatan dan pengelolaan data pasien secara digital untuk klinik X, mencakup informasi pribadi, riwayat medis, diagnosis, dan catatan perawatan, yang dibuat menggunakan bahasa C dan tampilan antarmuka GUI. Dengan adanya sistem pencatatan pasien terintegrasi, diharapkan dapat memudahkan klinik X dalam mengelola data pasien.

Keywords: Sistem Pencatatan, Bahasa C, Data

I. PENDAHULUAN

A. Latar Belakang

Pada era digital saat ini, efisiensi dan akurasi dalam pengelolaan data kesehatan sangatlah penting. Klinik-klinik kesehatan, baik kecil maupun besar, menghadapi tantangan dalam mengelola informasi pasien secara efektif. Penggunaan sistem manual atau berbasis kertas sering kali menyebabkan kesalahan, duplikasi data, dan kesulitan dalam pengaksesan informasi yang cepat dan tepat.

Klinik X, sebagai salah satu penyedia layanan kesehatan, mengalami permasalahan terkait. Pengelolaan data pasien yang mencakup informasi pribadi, riwayat medis, diagnosis, dan catatan perawatan sering kali memakan waktu dan rawan menghadapi kesalahan jika tidak didukung oleh sistem yang terkomputerisasi. Oleh karena itu, diperlukan sebuah aplikasi yang dapat membantu mempermudah pengelolaan data pasien secara lebih efisien dan akurat.

B. Tujuan

Tujuan dari dibuatnya aplikasi pencatatan pasien ini adalah:

1. **Meningkatkan Efisiensi Pengelolaan Data**: Mempermudah proses pencatatan, pengubahan, pencarian, dan penghapusan data pasien.
2. **Meminimalisasi *Human Error***: Mengurangi risiko kesalahan dalam pencatatan data yang sering terjadi pada sistem manual.
3. **Meningkatkan Aksesibilitas Informasi**: Memastikan data pasien dapat diakses dengan cepat dan mudah oleh petugas medis yang berwenang.
4. **Menyediakan Analisis Data**: Memfasilitasi analisis data pasien untuk membantu pengambilan keputusan medis dan manajerial.

5. **Menyimpan Data Secara Terpusat**: Mengintegrasikan semua data pasien dalam satu sistem terpusat yang dapat diakses oleh berbagai departemen di klinik.

C. Pentingnya Aplikasi

Pengembangan aplikasi pencatatan pasien berbasis bahasa pemrograman C ini penting bagi Klinik X karena beberapa alasan:

- **Efisiensi Operasional**: Aplikasi ini akan mengurangi beban administrasi dan meningkatkan produktivitas staf klinik dengan mengotomatisasi banyak tugas rutin.
- **Keakuratan Data**: Dengan menggunakan sistem terkomputerisasi, risiko kesalahan data dapat diminimalisasi yang pada akhirnya akan meningkatkan kualitas pelayanan kepada pasien.
- **Keamanan Data**: Sistem yang terkomputerisasi dengan baik dapat menyediakan mekanisme keamanan yang lebih baik untuk melindungi data pasien dari akses yang tidak sah.
- **Kepuasan Pasien**: Dengan pengelolaan data yang lebih baik, klinik dapat memberikan layanan yang lebih cepat dan akurat kepada pasien sehingga dapat meningkatkan kepuasan pasien.
- **Dasar untuk Pengembangan Lebih Lanjut**: Aplikasi ini dapat menjadi dasar bagi pengembangan lebih lanjut menuju sistem yang lebih kompleks dan terintegrasi seperti rekam medis elektronik (EMR) yang lebih komprehensif.

Dengan latar belakang, tujuan, dan pentingnya aplikasi ini, diharapkan proyek pengembangan sistem pencatatan pasien di Klinik X dapat berjalan dengan baik dan memberikan manfaat yang signifikan bagi klinik dan pasien.

II. RUMUSAN MASALAH

Klinik X sedang menghadapi berbagai tantangan dalam pengelolaan data pasien yang berdampak pada efisiensi, akurasi, dan kualitas layanan. Beberapa masalah utama yang perlu diidentifikasi dan dipecahkan melalui pengembangan aplikasi pencatatan pasien meliputi:

A. Pengelolaan Data Pasien

- **Efisiensi Pencatatan**: Sistem manual saat ini menyebabkan proses pencatatan data pasien menjadi lambat dan rentan terhadap kesalahan. Bagaimana cara merancang sistem pencatatan digital yang dapat mempercepat proses ini?
- **Integrasi Informasi**: Informasi pasien tersebar di berbagai sumber, mulai dari data pribadi hingga riwayat medis. Bagaimana cara mengintegrasikan semua data tersebut ke dalam satu sistem yang mudah diakses dan dikelola?
- **Aksesibilitas dan Pencarian**: Bagaimana cara memastikan data pasien dapat diakses dengan cepat oleh petugas medis yang memerlukan, serta menyediakan fungsi pencarian yang efektif untuk menemukan informasi spesifik?

B. Pengelolaan Riwayat Kedatangan dan Tindakan Medis

- **Pencatatan Riwayat**: Riwayat kedatangan pasien, diagnosis, dan tindakan medis perlu dicatat dengan rinci dan akurat. Bagaimana cara merancang sistem yang dapat mencatat semua informasi ini secara sistematis dan mudah diakses kembali?
- **Penyajian Informasi**: Petugas medis memerlukan akses cepat ke riwayat medis pasien untuk pengambilan keputusan klinis. Bagaimana cara menyajikan informasi ini dalam format yang mudah dipahami dan digunakan?

C. Pelaporan Keuangan

- **Otomatisasi Laporan**: Pembuatan laporan pendapatan bulanan dan tahunan secara manual memakan waktu dan rawan kesalahan. Bagaimana cara mengotomatisasi proses ini untuk menghasilkan laporan yang akurat dan tepat waktu?

- **Analisis Keuangan**: Bagaimana cara menyusun laporan yang dapat mencatat pendapatan, untuk setiap bulan dan juga tahun untuk total pendapatan dan serta rata-ratanya?

D. Analisis Data Kesehatan

- **Pengelompokan Data Penyakit**: Klinik perlu mengetahui jenis-jenis penyakit yang paling umum hingga yang paling jarang terjadi di antara pasiennya. Bagaimana cara merancang sistem yang dapat mengelompokkan dan menganalisis data ini setiap bulan dan tahun?
- **Dukungan Pengambilan Keputusan**: Bagaimana cara menyajikan data ini dalam bentuk yang mudah dipahami untuk membantu perencanaan dan pengambilan keputusan di klinik?

E. Informasi untuk Petugas Medis

- **Jadwal Kontrol**: Petugas medis perlu mengetahui jadwal kontrol pasien secara efisien. Bagaimana cara merancang sistem yang dapat memberikan informasi ini secara real-time dan mengingatkan petugas akan pasien yang perlu kembali untuk kontrol?
- **Pembaruan Informasi**: Bagaimana cara memastikan bahwa informasi tentang jadwal kontrol dan kebutuhan medis pasien selalu up-to-date dan dapat diakses dengan mudah oleh petugas medis?

III. STUDI LITERATUR

A. GRAPHICAL USER INTERFACE (GUI)

Antarmuka pengguna grafis, atau GUI (/ˈguːi/ GOO-ee), adalah sebuah bentuk antarmuka pengguna yang memungkinkan pengguna berinteraksi dengan perangkat elektronik melalui ikon grafis dan indikator visual seperti notasi sekunder. Dalam banyak aplikasi, GUI digunakan sebagai pengganti antarmuka pengguna berbasis teks, yang didasarkan pada label perintah yang diketik atau navigasi teks. GUI diperkenalkan sebagai reaksi terhadap kurva pembelajaran yang dianggap curam dari antarmuka baris perintah (CLI), yang memerlukan perintah untuk diketik pada keyboard komputer.[1]

B. BAHASA PEMROGRAMAN C

Bahasa pemrograman C adalah bahasa pemrograman komputer bertujuan umum yang dibuat pada tahun 1972 oleh Dennis Ritchie untuk Sistem Operasi Unix di Bell Telephone Laboratories. Dengan desain, fitur C dengan jelas mencerminkan kemampuan CPU yang ditargetkan.
Meskipun C dibuat untuk memprogram sistem dan jaringan komputer namun bahasa ini juga sering digunakan dalam mengembangkan software aplikasi. C juga banyak dipakai oleh berbagai jenis platform sistem operasi dan arsitektur komputer, bahkan terdapat beberapa compiler yang sangat populer telah tersedia. C secara luar biasa memengaruhi bahasa populer lainnya, terutama C++ yang merupakan extensi dari C.[2]

C. FILE EXTERNAL

Dalam Bahasa C tersedia fitur untuk mengolah sebuah file eksternal. Proses pengolahan meliputi membuka file, menulis file, membaca file, dan menutup file.

1. Membuka File
   Proses membuka suatu file eksternal dapat dilakukan dengan menggunakan fungsi fopen(). Fungsi tersebut akan menerima parameter berupa nama file dan mode yang digunakan. Syntax yang digunakan adalah berikut:

```
FILE *fopen( const char * filename, const char * mode );
```

2. Menulis File Proses
   Menulis file dapat dilakukan menggunakan dua cara,. Cara pertama dengan satu persatu tiap karakter dengan syntax:

```
int fputc( int c, FILE *fp );
```

   Cara kedua secara langsung dalam bentuk string, dengan syntax:

```
int fputs( const char *s, FILE *fp );
```

   Kedua metode tersebut akan menerima dua buah parameter yaitu data yang akan ditulis dalam bentuk karakter atau string dan file yang akan digunakan untuk menulis.

3. Membaca File
   Mirip seperti proses menulis, pada proses membaca file juga terdapat dua buah metode yang dapat digunakan yaitu dengan membaca karakter menggunakan syntax:

```
int fgetc( FILE * fp );
```

   dan membaca satu baris string menggunakan syntax:

```
char *fgets( char *buf, int n, FILE *fp );
```

   Pembacaan file berdasarkan karakter, syntax baris pertama, hanya memerlukan sebuah parameter yaitu file yang akan dibaca. Berbeda dengan metode pembacaan karakter, pembacaan berdasarkan string memerlukan tiga buah parameter yaitu variabel untuk menyimpan string yang dibaca, panjang string yang bisa dibaca, dan nama file yang dibaca.

4. Menutup File
   Proses menutup suatu file dapat dilakukan dengan mudah menggunakan fungsi fclose().

```
int fclose( FILE *fp );
```

   Fungsi tersebut akan memberikan nilai EOF jika terjadi eror saat menutup sebuah file.[3]

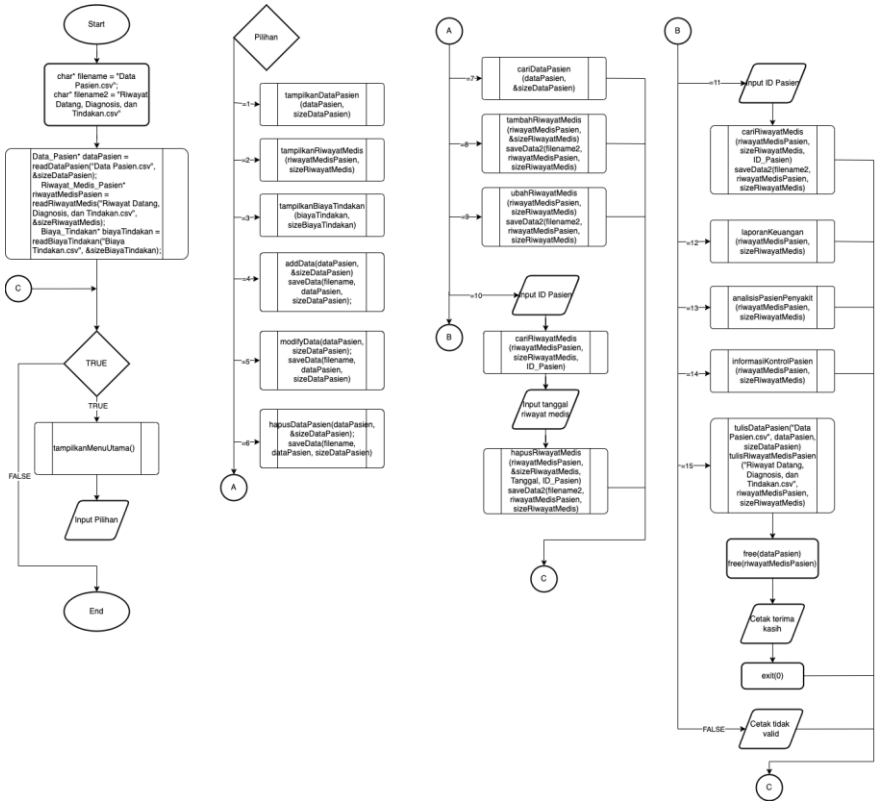
IV. METODOLOGI PEMBUATAN APLIKASI

Bagian ini berisi *flowchart*, *Data Flow Diagram* (DFD), dan pemaparan mengenai *Graphical User Interface* (GUI) yang diintegrasikan ke dalam program yang dibuat.

A. Flowchart

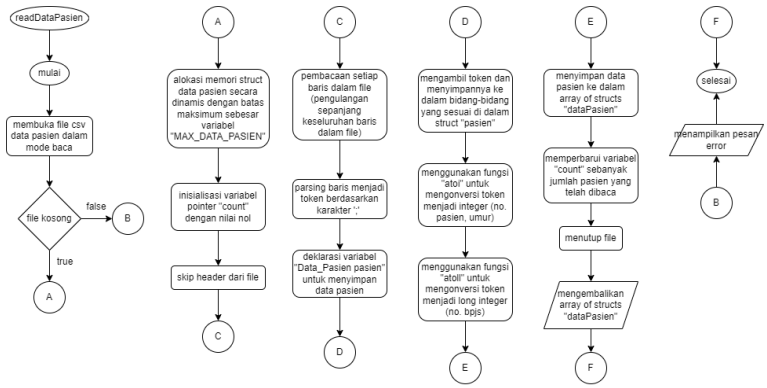
1) Program Utama dan Beberapa Fungsi Penunjang

a) Program Utama



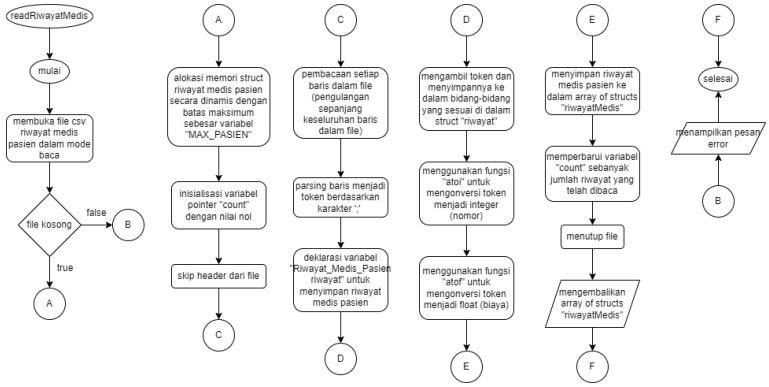
Gambar 4.A.1.a Flowchart Main Program

b) Fitur Baca Data Pasien



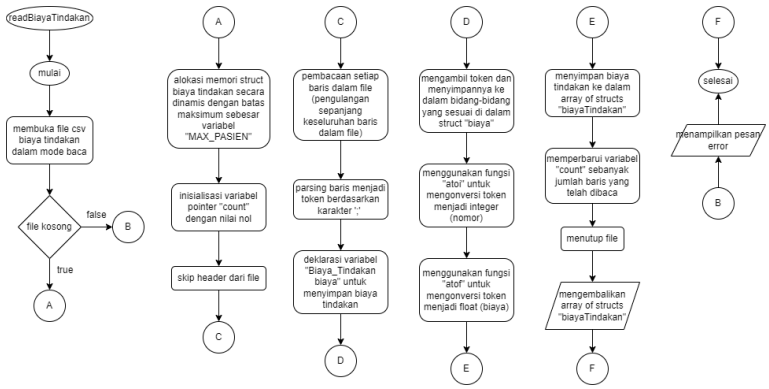
Gambar 4.A.1.b Flowchart Fitur Baca Data Pasien

c) Fitur Baca Riwayat Medis Pasien



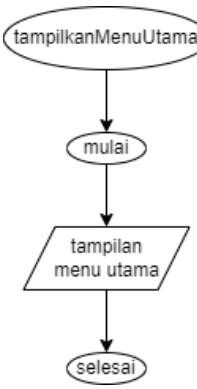
Gambar 4.A.1.c Flowchart Fitur Baca Riwayat Medis Pasien

d) Fitur Baca Biaya Tindakan



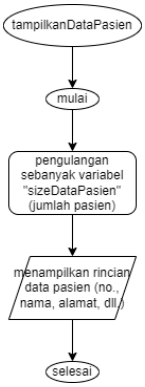
Gambar 4.A.1.d Flowchart Fitur Baca Biaya Tindakan

e) Fitur Tampilkan Menu Utama



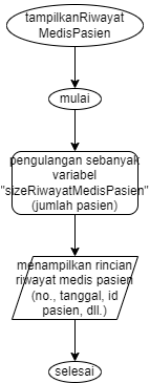
Gambar 4.A.1.e Flowchart Fitur Tampilkan Menu Utama

f) Fitur Tampilkan Data Pasien



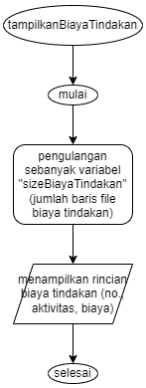
Gambar 4.A.1.f Flowchart Fitur Tampilkan Data Pasien

g) Fitur Tampilkan Riwayat Medis Pasien



Gambar 4.A.1.g Flowchart Fitur Tampilkan Riwayat Medis Pasien

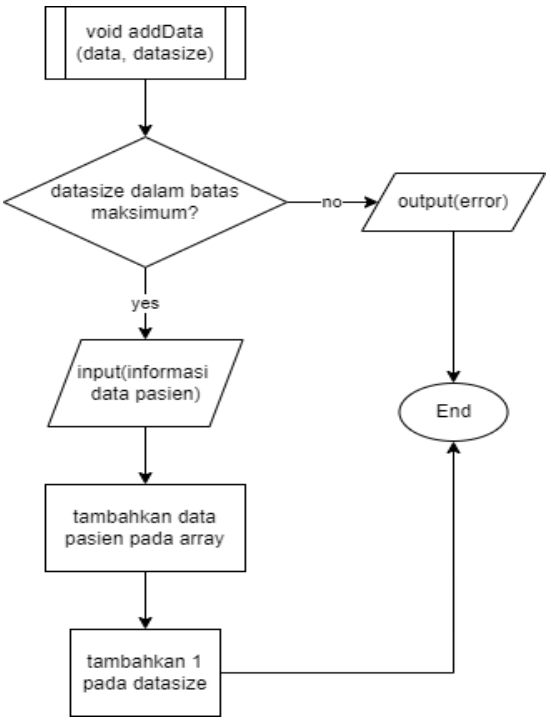
h) Fitur Tampilkan Biaya Tindakan



Gambar 4.A.1.h Flowchart Fitur Tampilkan Biaya Tindakan

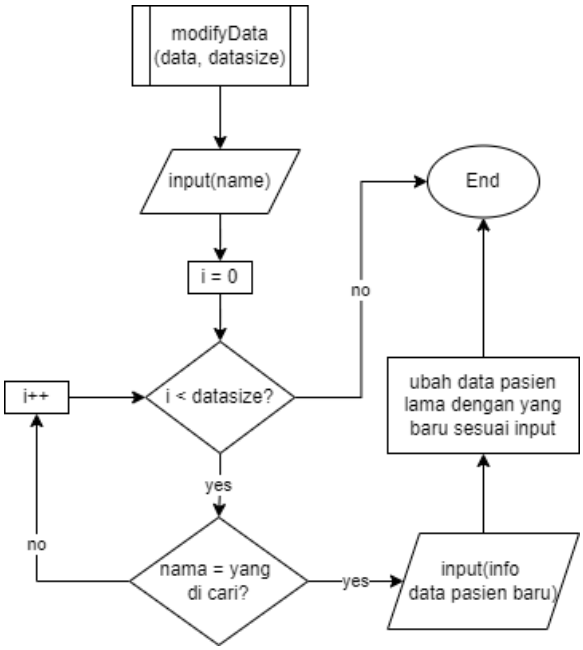
2) Manajemen Data Pasien

a) Fitur Tambah Data Pasien



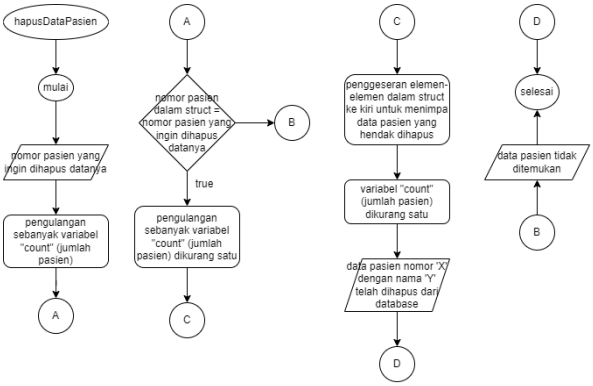
Gambar 4-A-2-a Flowchart Fungsi “addData”

b) Fitur Ubah Data Pasien



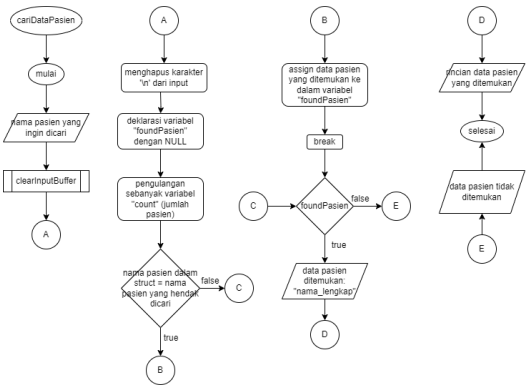
Gambar 4-A-2-b Flowchart Fungsi “modifyData”

c) Fitur Hapus Data Pasien



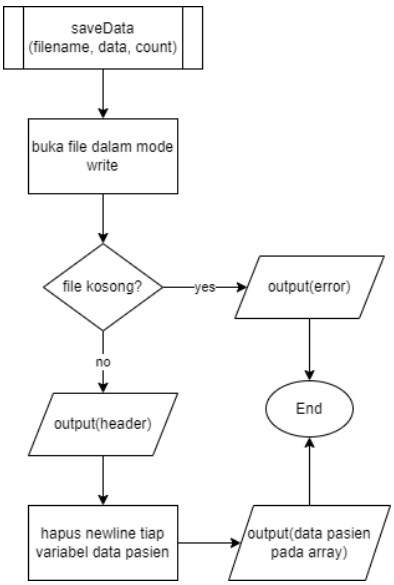
Gambar 4-A-2-c Flowchart Fungsi “hapusDataPasien”

d) Fitur Cari Data Pasien



Gambar 4-A-2-d Flowchart Fungsi “cariDataPasien”

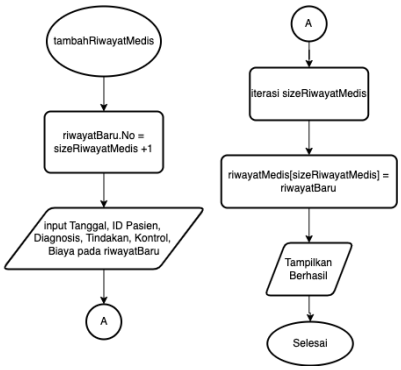
e) Fitur Simpan Data Pasien



Gambar 4-A-2-e Flowchart Fungsi “saveData”

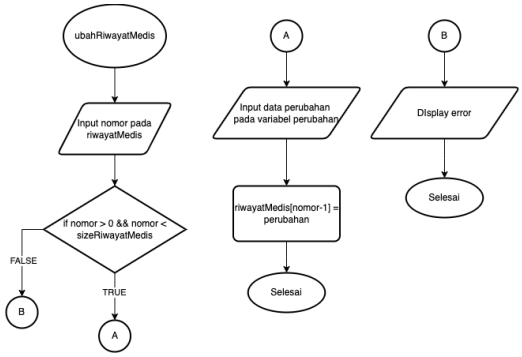
3) *Riwayat Medis dan Tindakan*

a) Fitur Tambah Riwayat Keterangan, Diagnosis, dan Tindakan



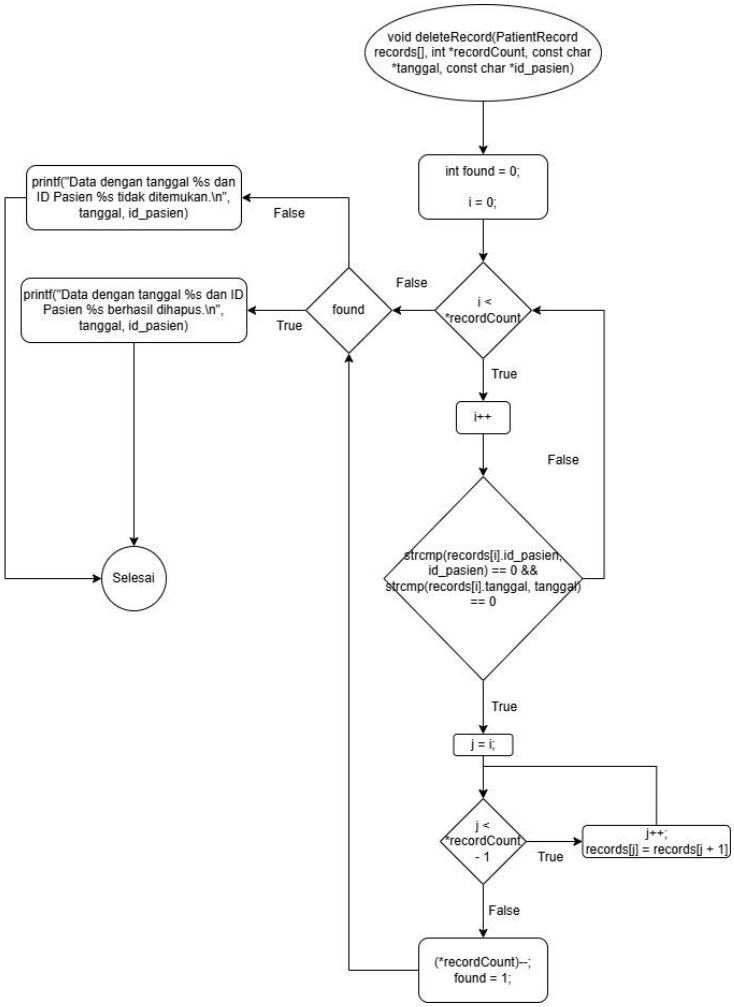
Gambar 4-A-3-a Flowchart Fungsi "tambahRiwayatMedis"

b) Fitur Ubah Riwayat Keterangan, Diagnosis, dan Tindakan



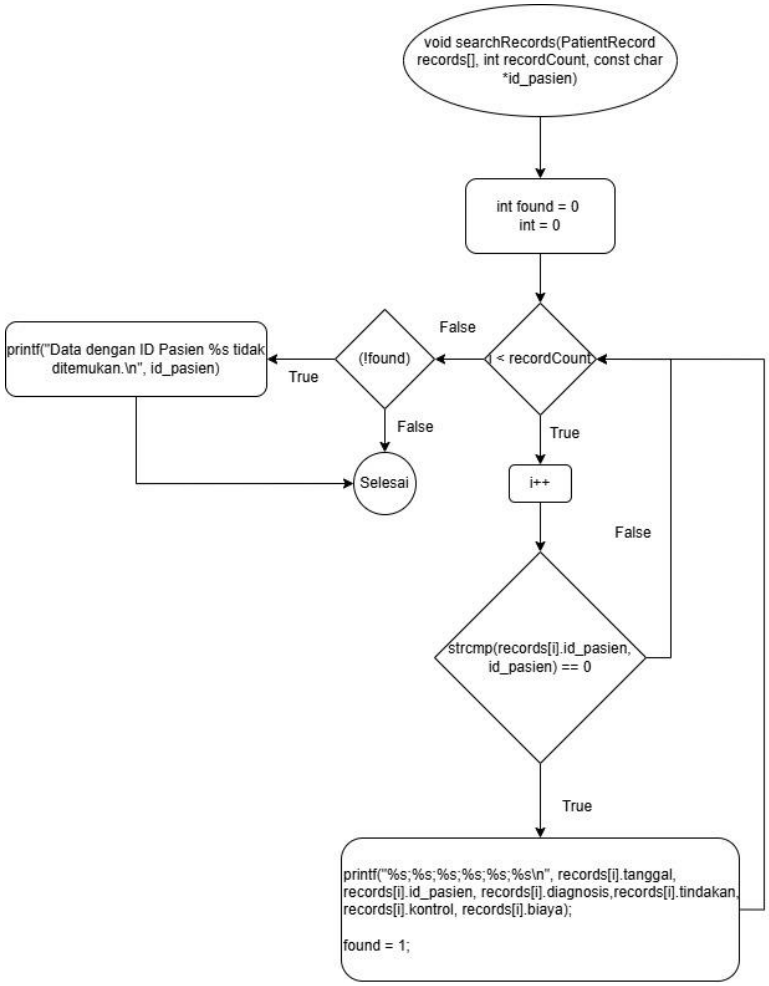
Gambar 4-A-3-b Flowchart Fungsi "ubahRiwayatMedis"

c) Fitur Hapus Riwayat Keterangan, Diagnosis, dan Tindakan



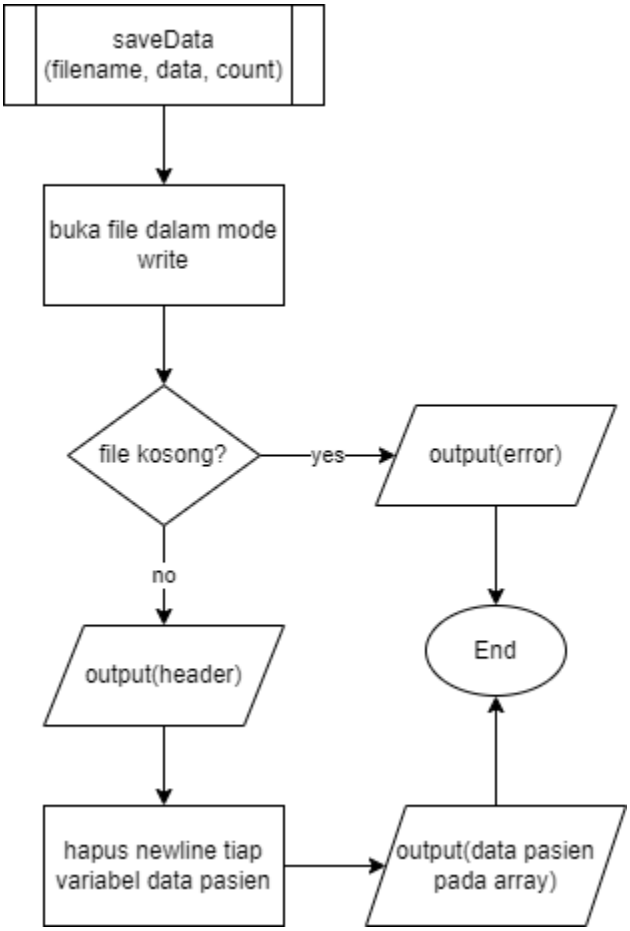
Gambar 4-A-3-c Flowchart Fitur Hapus Riwayat Datang, Diagnosis, dan Tindakan

d) Fitur Cari Riwayat Keterangan, Diagnosis, dan Tindakan



Gambar 4-A-3-d Flowchart Fitur Cari Riwayat Datang, Diagnosis, dan Tindakan

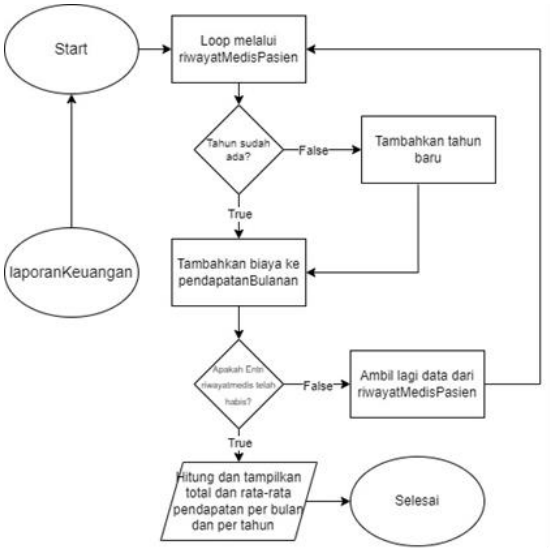
e) Fitur Simpan Riwayat Keterangan, Diagnosis, dan Tindakan



Gambar 4-A-3-e Flowchart Fungsi “saveData”

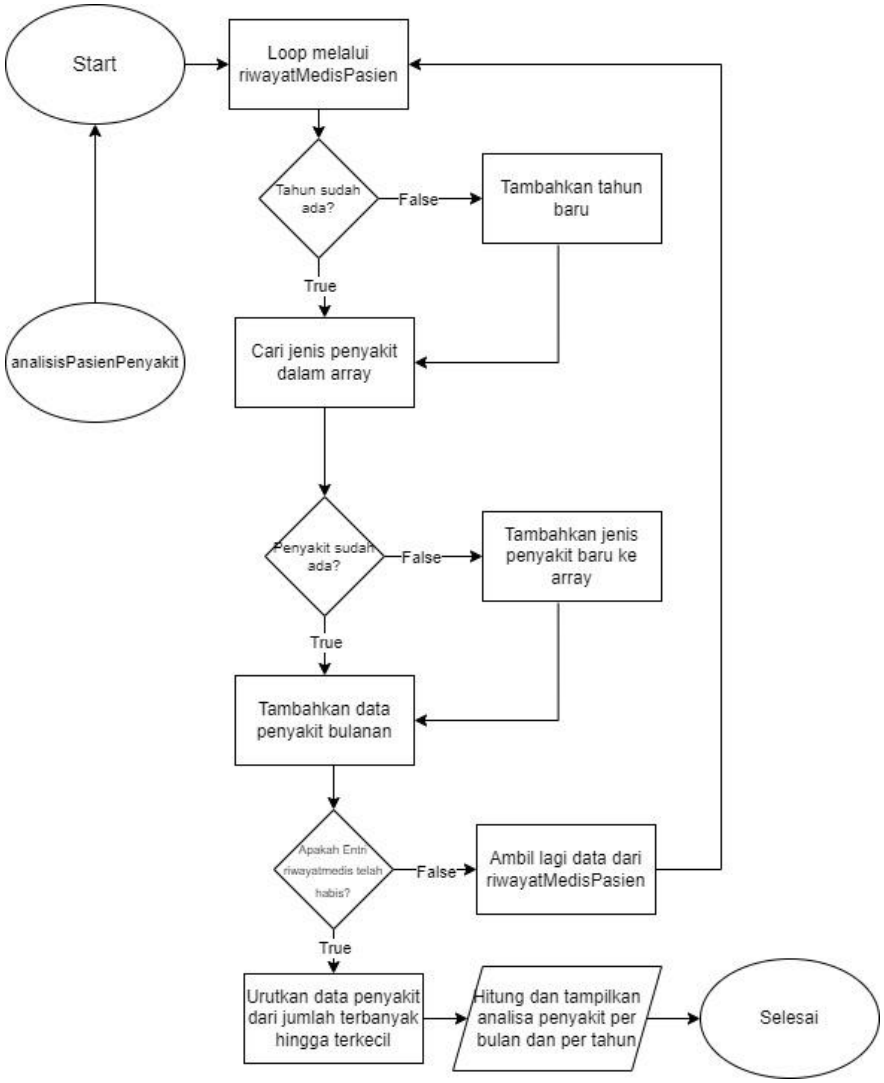
4) Laporan Keuangan, Analisis Penyakit, dan Informasi Kontrol Pasien

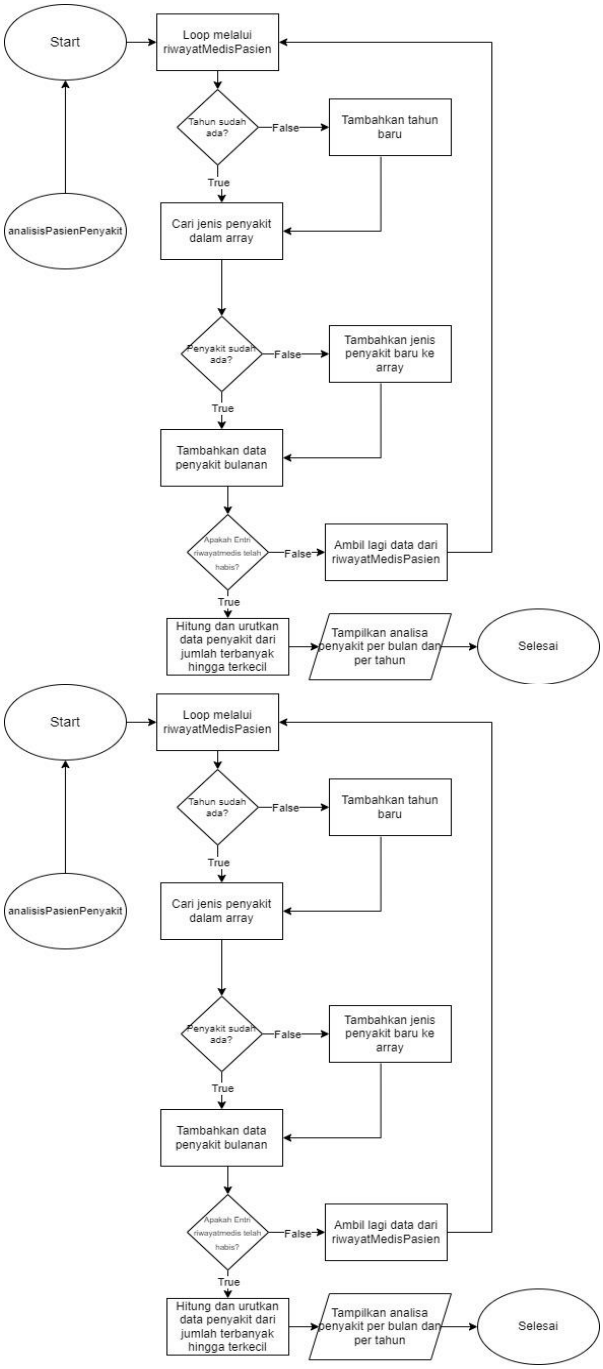
a) Fitur Laporan Keuangan (Bulanan, Tahunan, Rata-rata Tahun)



Gambar 4-A-4-a Flowchart Fitur Laporan Keuangan

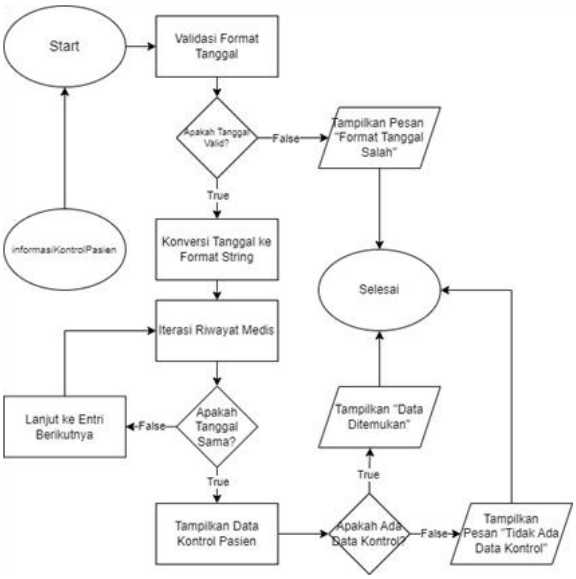
b) Fitur Analisis Penyakit Pasien


Start
Loop melalui riwayatMedisPasien
Tahun sudah ada?
False
Tambahkan tahun baru
True
analisisPasienPenyakit
Cari jenis penyakit dalam array
Penyakit sudah ada?
False
Tambahkan jenis penyakit baru ke array
True
Tambahkan data penyakit bulanan
Apakah Entri riwayatmedis telah habis?
False
Ambil lagi data dari riwayatMedisPasien
True
Urutkan data penyakit dari jumlah terbanyak hingga terkecil
Hitung dan tampilkan analisa penyakit per bulan dan per tahun
Selesai



Gambar 4-A-4-b Flowchart Fitur Analisis Pasien dan Penyakit

c) Fitur Informasi Kontrol Pasien

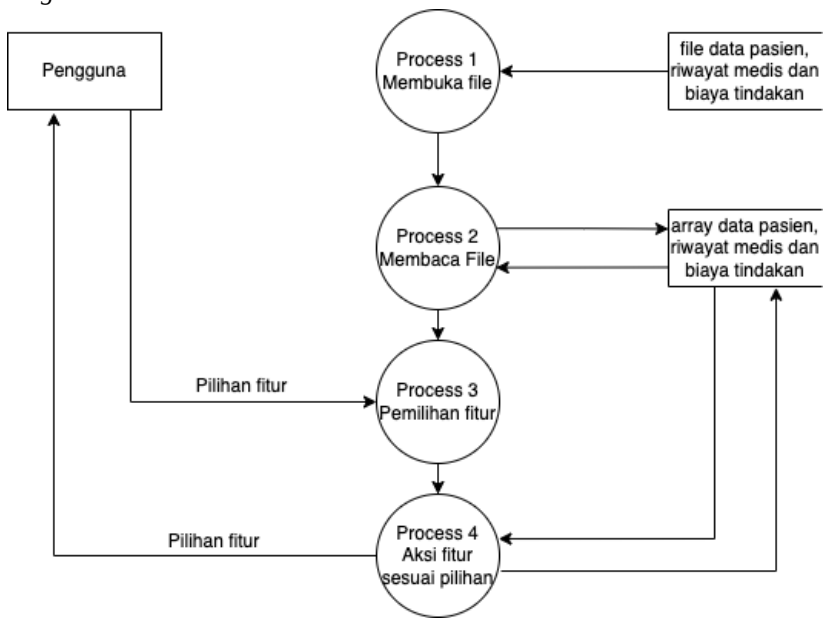


Gambar 4-A-4-c Flowchart Fitur Informasi Kontrol Pasien

B. Data Flow Diagram (DFD)

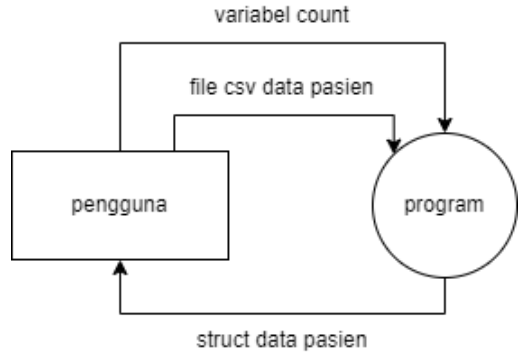
1) Program Utama dan Beberapa Fungsi Penunjang

a) Program Utama

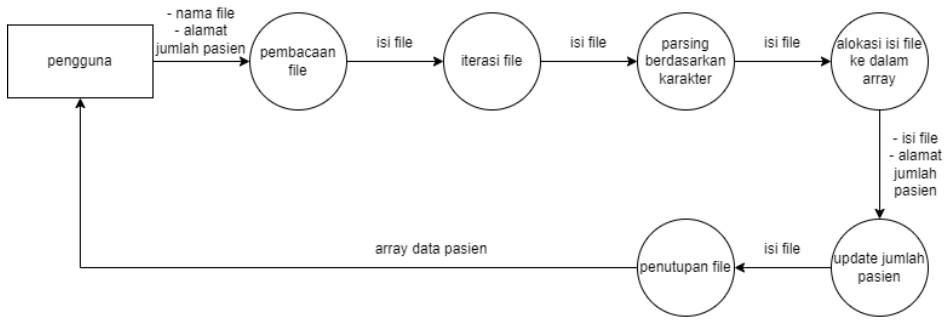


Gambar 4-B-1-a DFD Level 1 Main Program

b) Fitur Baca Data Pasien

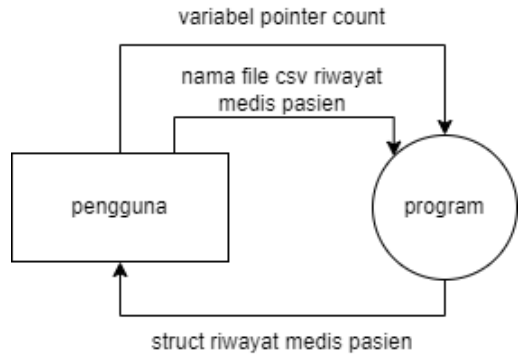


Gambar 4-B-1-b.1 DFD Level 0 Fitur Baca Data Pasien



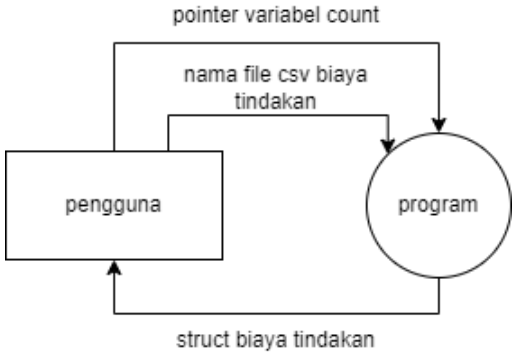
Gambar 4-B-1-b.2 Gambar DFD Level 1 Fitur Baca Data Pasien

c) Fitur Baca Riwayat Medis Pasien



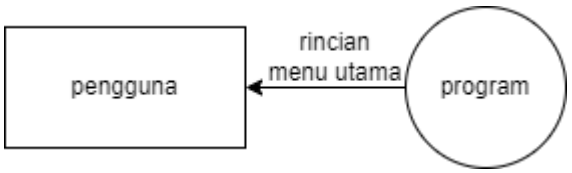
Gambar 4-B-1-c DFD Level 0 Fitur Baca Riwayat Medis Pasien

d) Fitur Baca Biaya Tindakan



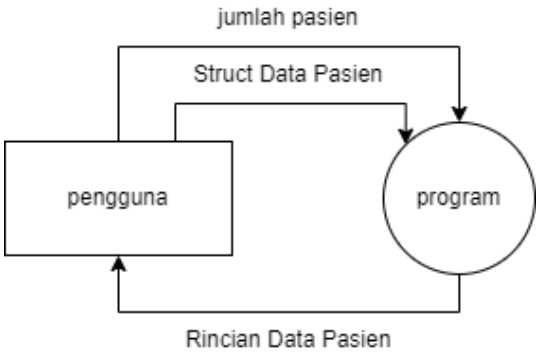
Gambar 4-B-1-d DFD Level 0 Fitur Baca Biaya Tindakan

e) Fitur Tampilkan Menu Utama



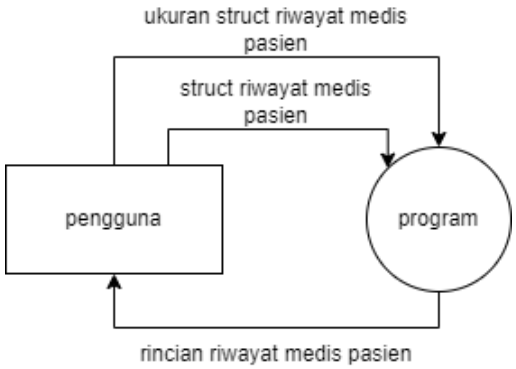
Gambar 4-B-1-e DFD Fitur Tampilkan Menu Utama

f) Fitur Tampilkan Data Pasien



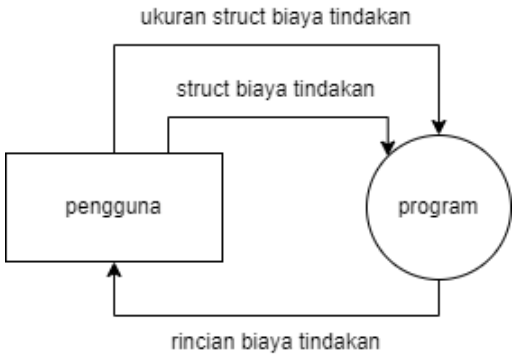
Gambar 4-B-1-f DFD Level 0 Fitur Tampilkan Data Pasien

g) Fitur Tampilkan Riwayat Medis Pasien



Gambar 4-B-1-g DFD Level 0 Fitur Tampilkan Riwayat Medis Pasien

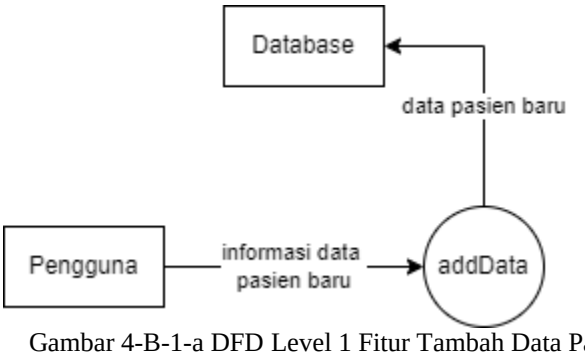
h) Fitur Tampilkan Biaya Tindakan



Gambar 4-B-1-h DFD Level 0 Fitur Tampilkan Biaya Tindakan

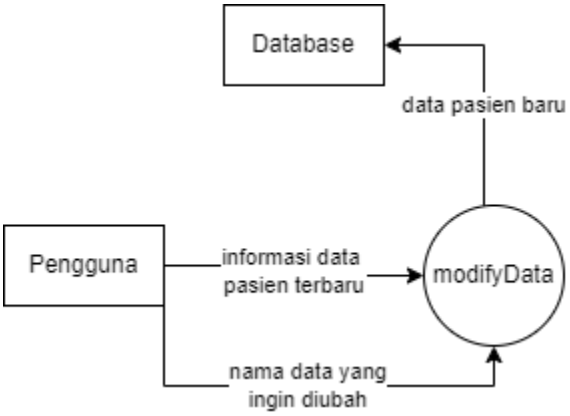
1) Manajemen Data Pasien

a) Fitur Tambah Data Pasien



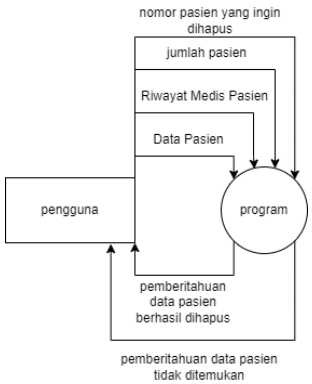
Gambar 4-B-1-a DFD Level 1 Fitur Tambah Data Pasien

b) Fitur Ubah Data Pasien

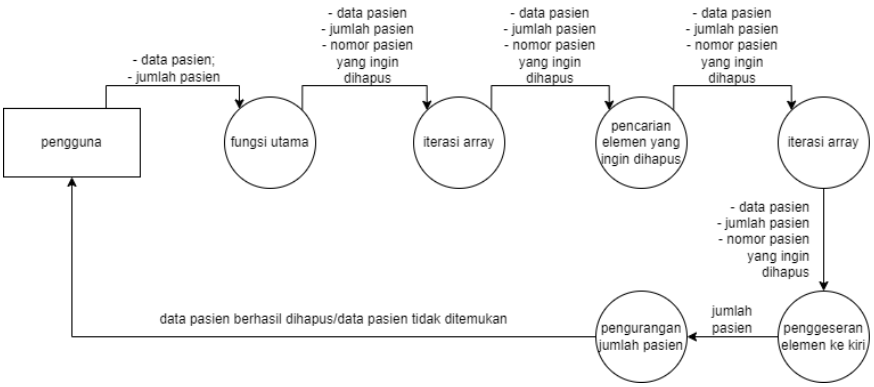


Gambar 4-B-1-b DFD Level 1 Fitur Ubah Data Pasien

c) Fitur Hapus Data Pasien



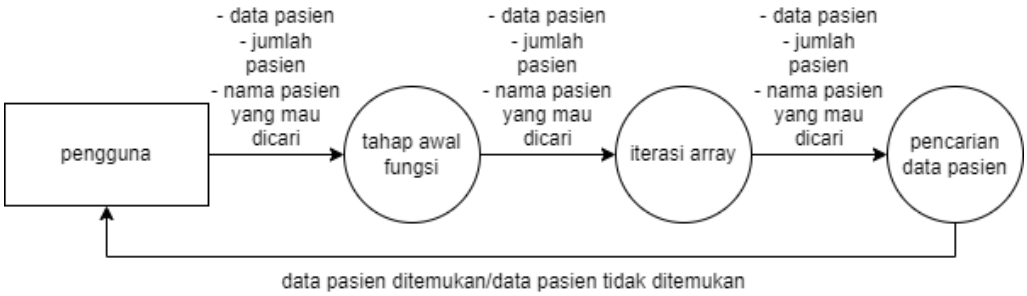
Gambar 4-B-1-c-1 DFD Level 0 Fitur Hapus Data Pasien



Gambar 4-B-1-c-2 DFD Level 1 Fitur Hapus Data Pasien

d) Fitur Cari Data Pasien

Gambar 4-B-1-d-1 DFD Level 0 Fitur Cari Data Pasien



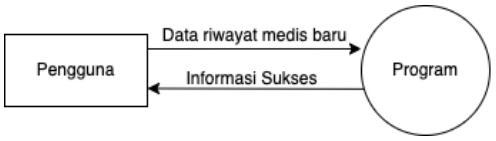
Gambar 4-B-1-d-2 DFD Level 1 Fitur Cari Data Pasien

e) Fitur Simpan Data Pasien

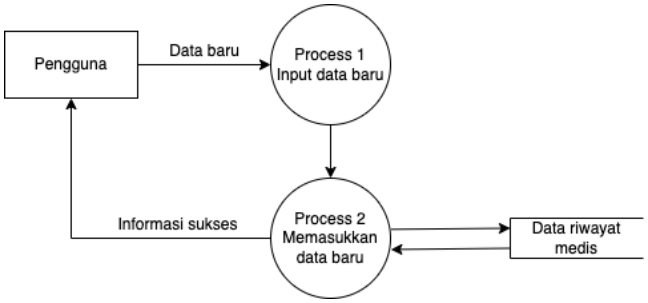
Gambar 4-B-1-e DFD Level 1 Fitur Simpan Data Pasien

2) Riwayat Medis dan Tindakan

a) Fitur Tambah Riwayat Keterangan, Diagnosis, dan Tindakan

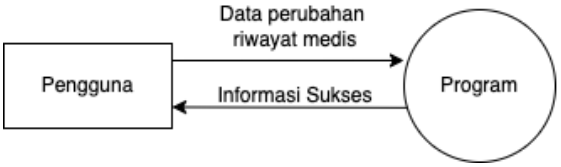


Gambar 4-B-2-a.1 DFD Level 0 Fitur tambah riwayat medis

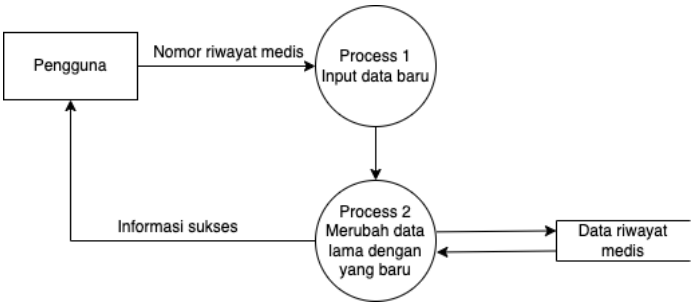


Gambar 4-B-2-a.2 DFD Level 1 Fitur tambah riwayat medis

b) Fitur Ubah Riwayat Keterangan, Diagnosis, dan Tindakan

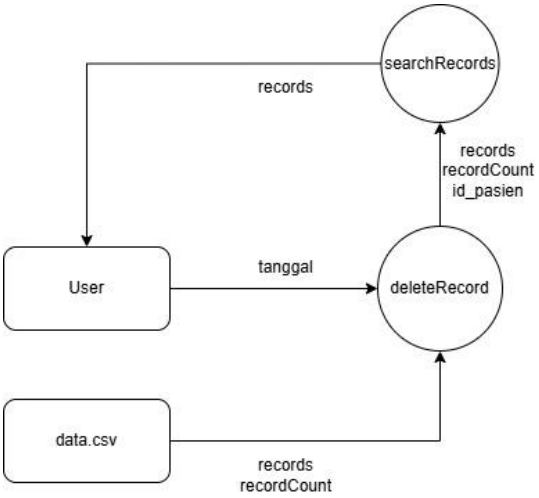


Gambar 4-B-2-b.1 DFD Level 0 Fitur ubah riwayat medis



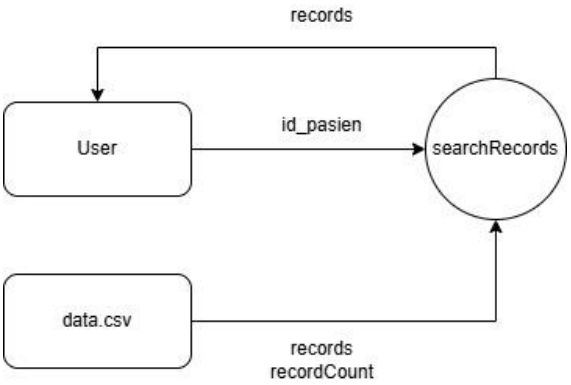
Gambar 4-B-2-b.2 DFD Level 1 Fitur ubah riwayat medis

c) Fitur Hapus Riwayat Keterangan, Diagnosis, dan Tindakan



Gambar 4-B-2-c DFD Level 1 Fitur hapus riwayat medis

d) Fitur Cari Riwayat Keterangan, Diagnosis, dan Tindakan



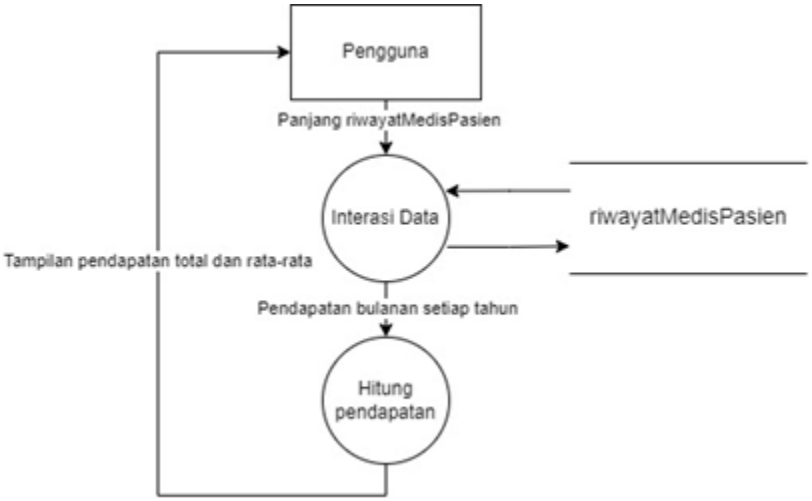
Gambar 4-B-2-d DFD Level 1 Fitur cari riwayat medis

e) Fitur Simpan Riwayat Keterangan, Diagnosis, dan Tindakan

Gambar 4-B-2-e DFD Level 1 Fitur Simpan Riwayat Pasien

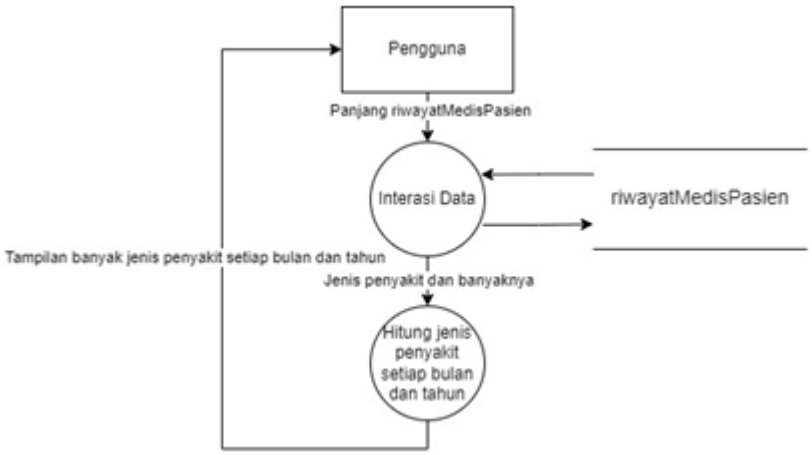
3) Laporan Keuangan, Analisis Penyakit, dan Informasi Kontrol Pasien

a) Fitur Laporan Keuangan (Bulanan, Tahunan, Rata-rata Tahun)



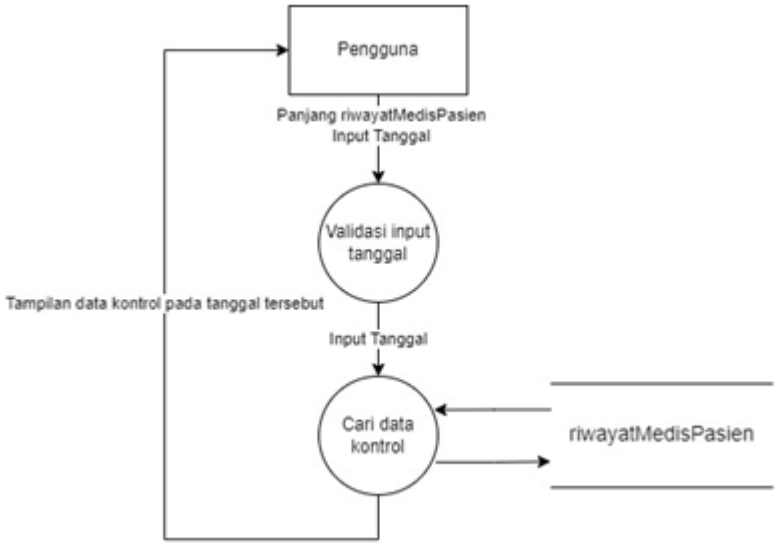
Gambar 4-B-3-a DFD Level 1 Fitur Laporan keuangan

b) Fitur Analisis Penyakit Pasien



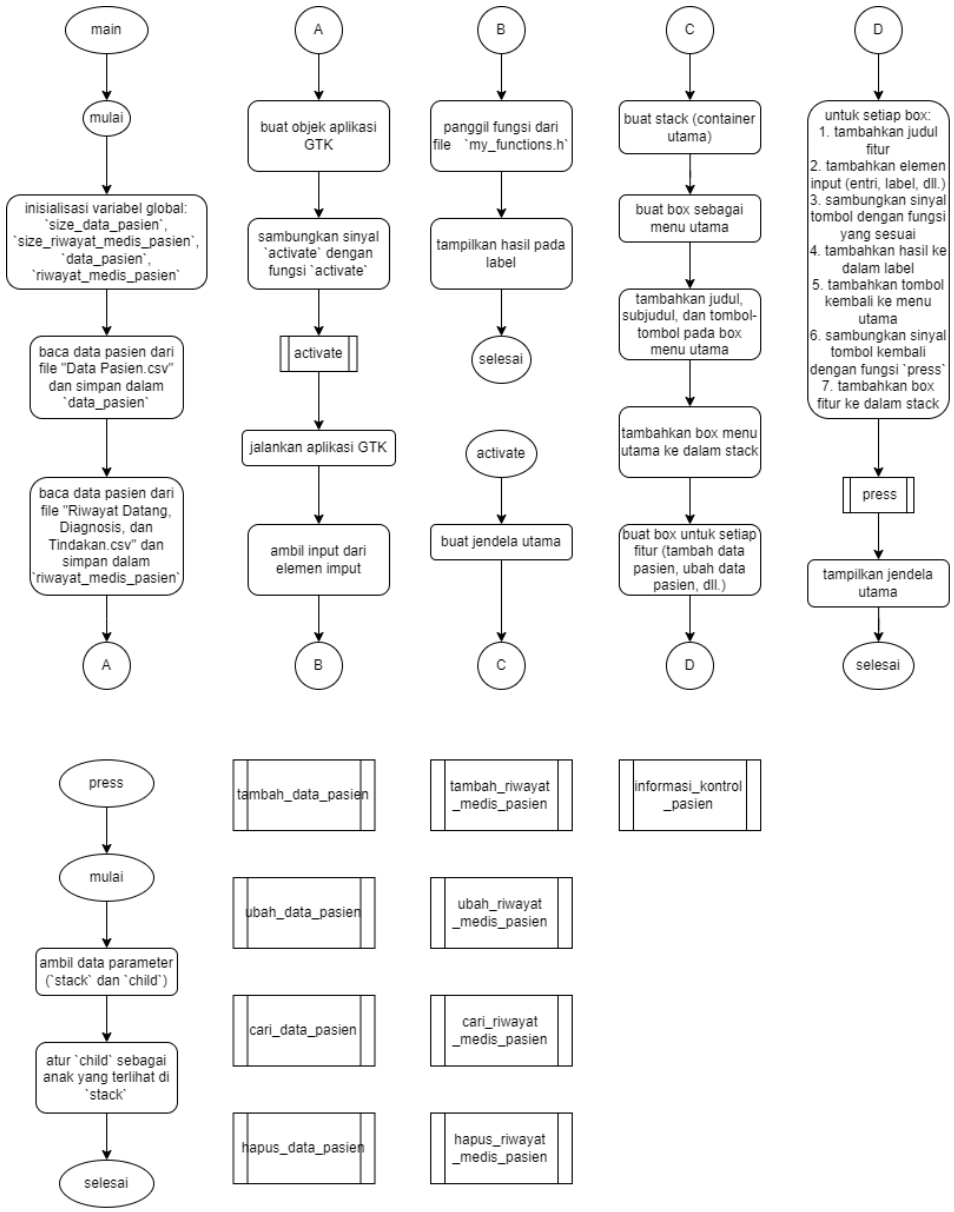
Gambar 4-B-3-b DFD Level 1 Fitur Analisis penyakit pasien

c) Fitur Informasi Kontrol Pasien



Gambar 4-B-3-c DFD Level 1 Fitur kontrol pasien

C. Graphical User Interface (GUI)



Gambar 4-C Flowchart Implementasi GUI

V. IMPLEMENTASI DAN ANALISIS

1) Manajemen Data Pasien

a) Fitur Tambah Data Pasien

Fitur ini diimplementasikan dengan fungsi addData(). Fungsi ini memeriksa apakah jumlah pasien saat ini (*count) kurang dari maksimum yang diizinkan (MAX_PASIEN). Jika ya, maka proses penambahan data dilanjutkan. Nomor pasien (No) diatur ke nilai *count + 1. Fungsi printf digunakan untuk meminta input dari pengguna dan fgets digunakan untuk membaca string input, termasuk nama lengkap, alamat, kota, tempat lahir, tanggal lahir, dan ID pasien.

Fungsi strcspn digunakan untuk menghapus karakter newline yang ditambahkan oleh fgets saat pengguna menekan enter. Untuk membaca umur dan nomor BPJS, digunakan variabel sementara (temp dan temp2) dan sscanf untuk mengonversi string ke tipe data yang sesuai (integer dan long long integer). Setelah semua data dimasukkan, nilai *count dinaikkan untuk menunjukkan bahwa satu data pasien telah berhasil ditambahkan. Jika array sudah penuh, fungsi akan mencetak pesan bahwa array sudah penuh dan tidak akan menambahkan data baru.

Kompleksitas waktu dari fungsi addData adalah O(1), atau konstan, karena tidak ada loop atau operasi rekursif yang jumlah iterasinya tergantung pada ukuran input. Semua operasi di dalam fungsi ini (seperti fgets, strcspn, dan sscanf) dijalankan satu kali per pemanggilan fungsi, tidak peduli berapa banyak data pasien yang sudah ada.
Kompleksitas ruang dari fungsi ini juga O(1), atau konstan. Meskipun fungsi ini menambahkan data ke array, array itu sendiri tidak diperbesar atau dikurangi dalam fungsi; hanya indeks yang diubah. Variabel sementara seperti temp dan temp2 digunakan untuk membaca input dan memiliki ukuran tetap.

b) Fitur Ubah Data Pasien

Fitur ini diimplementasikan dengan fungsi modifyData. Fungsi modifyData digunakan untuk memodifikasi data pasien dalam array berdasarkan nama lengkap. Fungsi meminta pengguna memasukkan nama lengkap pasien yang datanya ingin dimodifikasi. Menggunakan loop for untuk mencari data pasien dengan nama yang cocok dalam array. Jika ditemukan, fungsi meminta pengguna untuk memasukkan data baru. Fungsi fgets dan strcspn digunakan untuk membaca dan membersihkan input dari newline. Setelah data dimodifikasi, loop dihentikan dengan break.

Kompleksitas waktu dari fungsi modifyData adalah O(n), di mana n adalah jumlah data pasien (count). Ini karena ada loop for yang iterasinya maksimal sebanyak jumlah data pasien. Dalam kasus terburuk, jika nama yang dicari adalah nama terakhir dalam array atau tidak ada sama sekali, loop akan berjalan melalui seluruh array.
Kompleksitas ruang dari fungsi ini adalah O(1), atau konstan. Meskipun fungsi ini memodifikasi data dalam array, tidak ada alokasi memori tambahan yang signifikan yang terjadi selain variabel lokal sementara seperti name, temp, dan temp2 yang memiliki ukuran tetap.

c) Fitur Hapus Data Pasien

Fungsi `hapusDataPasien` digunakan untuk menghapus data pasien dari array `dataPasien` dan `riwayatMedisPasien` berdasarkan nomor pasien yang dimasukkan oleh pengguna. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Pertama, fungsi meminta pengguna untuk memasukkan nomor pasien yang ingin dihapus datanya menggunakan `printf` dan `scanf`.
b. Fungsi kemudian mencari data pasien dengan nomor yang sesuai dalam array `dataPasien` menggunakan perulangan `for`. Jika data pasien ditemukan, fungsi melakukan langkah-langkah berikut: a. Menyimpan nama lengkap pasien dalam variabel `nama_pasien` menggunakan `strcpy`. b. Menggeser elemen-elemen setelah posisi data pasien yang dihapus ke kiri menggunakan perulangan `for` bersarang. Pergeseran ini dilakukan untuk mengisi kekosongan yang ditimbulkan setelah penghapusan data pasien. c. Mengurangi nilai `count` (jumlah elemen dalam array) dengan 1 karena satu data pasien telah dihapus. d. Menampilkan pesan konfirmasi bahwa data pasien telah dihapus menggunakan `printf`. e. Keluar dari fungsi menggunakan `return`.

c. Jika data pasien tidak ditemukan setelah perulangan selesai, fungsi akan menampilkan pesan "Data pasien tidak ditemukan" menggunakan `printf`.

Fungsi ini memiliki kompleksitas waktu `O(n)`, di mana `n` adalah jumlah data pasien dalam array. Hal ini disebabkan oleh perulangan `for` yang digunakan untuk mencari data pasien yang ingin dihapus dan pergeseran elemen-elemen setelah penghapusan.

Lebih lanjut, fungsi ini memiliki kompleksitas ruang (*space complexity*) sebesar `O(1)`. Hal ini disebabkan fungsi ini tidak mengalokasikan memori secara dinamis. Hal ini didukung dengan hanya digunakannya variabel-variabel statis seperti `no_pasien`, `i`, dan `j` bertipe integer, serta `nama_pasien` bersifat string. Di sisi lain, dapat dilihat bahwa *array* `dataPasien` dan `riwayatMedisPasien` sudah dialokasikan sebelumnya di luar fungsi.

Secara keseluruhan, fungsi ini cukup efisien dalam menghapus data pasien dari array karena tidak perlu membuat array baru atau mengalokasikan memori tambahan.

d) Fitur Cari Data Pasien

Fungsi `cariDataPasien` digunakan untuk mencari data pasien dalam *array* `dataPasien` berdasarkan nama pasien yang dimasukkan oleh pengguna. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Fungsi meminta pengguna untuk memasukkan nama pasien yang ingin dicari menggunakan `printf` dan `fgets`. Nama pasien yang dimasukkan disimpan dalam variabel `nama_pasien`.
b. Fungsi memanggil fungsi `clearInputBuffer` untuk membersihkan buffer input sebelum membaca input dari pengguna.
c. Karakter newline (`\n`) pada akhir input yang dimasukkan oleh pengguna dihapus menggunakan `strcspn` dan assignment karakter `\0`.
d. Fungsi menginisialisasi pointer `foundPasien` dengan nilai `NULL`.
e. Fungsi melakukan pencarian data pasien dengan nama yang sesuai dalam array `dataPasien` menggunakan perulangan `for`.
f. Dalam perulangan, fungsi membandingkan nama pasien dalam setiap elemen array `dataPasien` dengan `nama_pasien` yang dimasukkan oleh pengguna menggunakan `strcmp`.
g. Jika ditemukan kecocokan nama, pointer `foundPasien` akan diisi dengan alamat memori dari elemen array `dataPasien` yang cocok, dan perulangan akan dihentikan dengan `break`.
h. Setelah perulangan selesai, fungsi akan memeriksa apakah `foundPasien` bernilai `NULL` atau tidak.
i. Jika `foundPasien` tidak `NULL`, fungsi akan menampilkan data pasien yang ditemukan menggunakan `printf`.
j. Jika `foundPasien` bernilai `NULL`, fungsi akan menampilkan pesan "Data pasien tidak ditemukan" menggunakan `printf`.

Kompleksitas waktu dari fungsi `cariDataPasien` adalah `O(n)`, di mana `n` adalah jumlah data pasien dalam array `dataPasien`. Hal ini disebabkan oleh perulangan `for` yang digunakan untuk mencari data pasien dengan nama yang sesuai.

Di sisi lain, kompleksitas ruang dari fungsi ini adalah `O(1)`. Hal ini disebabkan tidak ada memori dinamis yang dialokasikan. Hal ini dibuktikan dengan dipakainya variabel-variabel statis, di antaranya `nama_pasien` (tipe data `char` dengan ukuran sebesar lima puluh) dan `foundPasien` (tipe data *pointer* ke `Data_Pasien`).

Ada pun kinerja dari fungsi ini dipengaruhi oleh beberapa faktor. Faktor yang pertama adalah kompleksitas waktu yang mana waktu pencarian dapat menjadi lambat untuk kasus dengan jumlah data pasien yang besar. Sejumlah operasi *string* juga menjadi faktor karena operasi-operasi ini dapat menjadi lambat jika panjang *string* nama pasien cukup besar.

Secara keseluruhan, kinerja dari fungsi `cariDataPasien` cukup baik untuk kasus dengan jumlah data pasien yang kecil hingga sedang. Namun, untuk kasus dengan jumlah data pasien yang besar, kinerja dapat menjadi lambat karena kompleksitas waktu yang linear.

e) Fitur Simpan Data Pasien

Fitur ini diimplementasikan dengan fungsi saveData. Fungsi saveData digunakan untuk menyimpan data pasien dari array ke file CSV. Fungsi membuka file dengan nama yang diberikan untuk menulis ("w"). Jika file berhasil dibuka, fungsi menulis header CSV. Loop for digunakan untuk menulis data setiap pasien ke file. Fungsi fprintf digunakan untuk menulis data dengan format CSV. Setelah semua data ditulis, file ditutup.

Kompleksitas waktu dari fungsi saveData adalah O(n), di mana n adalah jumlah data pasien (count). Ini karena ada loop for yang menulis setiap data pasien ke dalam file, dan loop ini berjalan sebanyak jumlah data pasien.
Kompleksitas ruang dari fungsi ini juga O(1), atau konstan. Fungsi ini tidak menggunakan memori tambahan yang skalanya bergantung pada ukuran input; hanya menggunakan variabel lokal sementara dan file pointer.

2) Riwayat Medis dan Tindakan

a) Fitur Tambah Riwayat Keterangan, Diagnosis, dan Tindakan

Fungsi `tambahRiwayatMedis` menambahkanst data riwayat medis pada *array* `riwayatMedis`. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Fungsi membuat variabel `riwayatBaru` dengan tipe data struct `Riwayat_Medis_Pasien`.
b. Fungsi menerima input berupa Tanggal, ID Pasien, Diagnosis, Tindakan, dan Kontrol. Nomor riwayat medis baru berupa `sizeRiwayatMedis` yang ditambah 1. Input dilakukan dengan fungsi `fgets`.
c. Karakter newline (`\n`) pada akhir input yang dimasukkan oleh pengguna dihapus menggunakan `strcspn` dan assignment karakter `\0`.
d. Fungsi memanggil fungsi `clearInputBuffer` untuk membersihkan buffer input sebelum membaca input dari pengguna.
e. Dilakukan iterasi pada `sizeRiwayatMedis` untuk menunjukkan data riwayat medis bertambah satu.
f. Lalu, elemen terakhir pada array `riwayatMedis` di-*assign* dengan `riwayatBaru`.
g. Ketika semua langkah telah dijalankan, fungsi akan menampilkan informasi bahwa penambahan riwayat medis telah berhasil.

Setiap baris pada fungsi ini dilakukan tepat satu kali tanpa dilakukannya perulangan. Maka dari itu, kompleksitas waktu dari fungsi ini adalah O(1). Di sisi lain, variabel yang digunakan pada fungsi ini juga merupakan variabel statis, yaitu tidak dilakukan alokasi memori. Kompleksitas ruang pada fungsi ini juga O(1).

Kinerja dari fungsi ini sudah baik karena tidak bergantung dengan variabel apa pun. Fungsi ini akan bekerja dengan waktu yang sama pada ukuran data yang besar maupun kecil.

b) Fitur Ubah Riwayat Keterangan, Diagnosis, dan Tindakan

Fungsi `tambahRiwayatMedis` menambahkan data riwayat medis pada *array* `riwayatMedis`. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Fungsi membuat variabel `perubahan` dengan tipe data struct `Riwayat_Medis_Pasien`.
b. Fungsi menerima input nomor riwayat medis dari pengguna.
c. Jika nomor tidak berada pada data, maka fungsi akan mencetak error, lalu fungsi selesai.
d. Dilakukan assignment pada `perubahan.No` sesuai input nomor pasien
e. Fungsi menerima input berupa Tanggal, ID Pasien, Diagnosis, Tindakan, dan Kontrol. Input dilakukan dengan fungsi `fgets`. Input tersebut disimpan pada variabel `perubahan`.
f. Karakter newline (`\n`) pada akhir input yang dimasukkan oleh pengguna dihapus menggunakan `strcspn` dan assignment karakter `\0`.
g. Fungsi memanggil fungsi `clearInputBuffer` untuk membersihkan buffer input sebelum membaca input dari pengguna.
h. Elemen ke `no-1` pada *array* `riwayatMedis` di-*assign* dengan variabel `perubahan`.
i. Ketika semua langkah telah dijalankan, fungsi akan menampilkan informasi bahwa perubahan riwayat medis telah berhasil.

Kompleksitas waktu pada fungsi ini adalah O(1) karena setiap baris dijalankan tepat satu kali tanpa adanya perulangan. Kompleksitas ruang pada fungsi ini adalah O(1) karena tidak ada alokasi memori pada variabel manapun. Kinerja fungsi ini sudah baik karena waktu ataupun memori yang digunakan tidak bergantung dengan variabel apapun.

c) Fitur Hapus Riwayat Keterangan, Diagnosis, dan Tindakan

Fungsi hapusRiwayatMedis digunakan untuk menghapus sebuah entri dalam array records berdasarkan tanggal (Tanggal) dan ID pasien (ID_Pasien). Fungsi ini melakukan iterasi melalui array untuk mencari entri yang cocok dengan kedua parameter tersebut. Jika ditemukan, entri tersebut dihapus dengan cara menggeser semua elemen setelahnya ke satu posisi sebelumnya, lalu mengurangi jumlah total entri (recordCount). Fungsi ini kemudian memberikan pesan apakah data berhasil dihapus atau tidak ditemukan.

Detail Implementasi:

1. Inisialisasi: Variabel found diatur ke 0 untuk menandakan apakah entri telah ditemukan.
2. Pencarian: Fungsi iterasi melalui array records dari indeks 0 hingga *recordCount.
3. Perbandingan: Setiap elemen diperiksa apakah ID_Pasien dan Tanggal cocok.
4. Penghapusan: Jika cocok, elemen tersebut dihapus dengan menggeser semua elemen setelahnya ke satu posisi sebelumnya.
5. Pengurangan Count: recordCount dikurangi satu.
6. Output: Pesan hasil dihapus atau tidak ditemukan ditampilkan.

Fungsi hapusRiwayatMedis memiliki kompleksitas waktu O(n) karena iterasi pertama melalui array records hingga menemukan entri yang sesuai berdasarkan Tanggal dan ID_Pasien. Setelah ditemukan, fungsi ini menggeser semua elemen setelah entri tersebut ke depan untuk menutupi posisi yang dihapus, yang juga memerlukan waktu

linier. Dengan demikian, waktu eksekusi fungsi ini bertambah secara proporsional dengan jumlah entri dalam array.

d) Fitur Cari Riwayat Keterangan, Diagnosis, dan Tindakan

Fungsi cariRiwayatMedis digunakan untuk mencari dan menampilkan semua entri dalam array records yang cocok dengan ID pasien (ID_Pasien). Fungsi ini melakukan iterasi melalui array dan memeriksa setiap elemen untuk melihat apakah ID pasien sesuai dengan parameter yang diberikan. Jika ditemukan, entri tersebut ditampilkan. Jika tidak ada entri yang cocok ditemukan, fungsi menampilkan pesan bahwa data tidak ditemukan.

Detail Implementasi:

1. Inisialisasi: Variabel found diatur ke 0 untuk menandakan apakah entri telah ditemukan.
2. Pencarian: Fungsi iterasi melalui array records dari indeks 0 hingga recordCount.
3. Perbandingan: Setiap elemen diperiksa apakah ID_Pasien cocok.
4. Output Data: Jika cocok, data ditampilkan.
5. Output: Jika tidak ada data yang cocok ditemukan, pesan data tidak ditemukan ditampilkan.

Fungsi cariRiwayatMedis memiliki kompleksitas waktu O(n) karena melakukan iterasi melalui seluruh array records untuk mencari semua entri yang cocok dengan ID_Pasien yang diberikan. Setiap kali menemukan entri yang cocok, fungsi mencetak detail entri tersebut. Karena pencarian dan pencetakan dilakukan dalam satu iterasi linear, waktu eksekusi fungsi ini juga bertambah secara proporsional dengan jumlah entri dalam array.

3) Laporan Keuangan, Analisis Penyakit, dan Informasi Kontrol Pasien

a) Fitur Laporan Keuangan (Bulanan, Tahunan, Rata-rata Tahun

Fungsi `laporanKeuangan` digunakan untuk menghitung pendapatan bulanan dan tahunan dari riwayat medis pasien yang disimpan dalam array `riwayatMedisPasien`. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Fungsi melakukan iterasi melalui setiap entri dalam array `riwayatMedisPasien`.
b. Tanggal pada setiap entri dipecah menjadi hari, bulan, dan tahun menggunakan fungsi `strtok`.
c. Nama bulan dikonversi menjadi indeks bulan menggunakan serangkaian `if-else` statements.
d. Fungsi memeriksa apakah tahun sudah ada dalam array `pendapatan`:
   - Jika tahun sudah ada, indeks tahun ditemukan dan digunakan.
   - Jika tahun belum ada, tahun baru ditambahkan ke array pendapatan dan diinisialisasi.
e. Biaya pada entri riwayat medis ditambahkan ke total pendapatan bulanan dan tahunan.
f. Jumlah data bulanan juga diperbarui.
g. Setelah semua entri diiterasi, pendapatan bulanan dan tahunan dihitung dan ditampilkan.

Proses pencarian dan pengelompokan data berdasarkan tahun dan bulan dalam riwayat medis pasien memerlukan waktu `O(n)`, di mana `n` adalah jumlah riwayat medis pasien (`sizeRiwayatMedis`). Hal ini karena setiap entri dalam riwayat medis diproses sekali. Sedangkan pencarian tahun dalam array `pendapatan` memerlukan

waktu `O(m)`, di mana `m` adalah jumlah tahun yang berbeda dalam data. Dalam kasus terburuk, pencarian ini dilakukan untuk setiap entri, sehingga kompleksitas waktu keseluruhan menjadi `O(n * m)`. Penggunaan memori dinamis untuk menyimpan data pendapatan tahunan mengakibatkan kompleksitas ruang `O(m)`.

b) Fitur Analisis Penyakit Pasien

Fungsi analisisPasienPenyakit digunakan untuk menganalisis jenis penyakit yang diderita oleh pasien per bulan dan per tahun dari riwayat medis pasien yang disimpan dalam array riwayatMedisPasien. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Struktur `analisapenyakit` menyimpan informasi mengenai tahun, jenis penyakit, jumlah penyakit bulanan, dan jumlah total penyakit.
b. Fungsi mengiterasi setiap entri dalam array `riwayatMedisPasien`.
c. Tanggal pada setiap entri dipecah menjadi hari, bulan, dan tahun menggunakan fungsi `strtok`.
d. Nama bulan dikonversi menjadi indeks bulan menggunakan serangkaian `if-else` statements.
e. Fungsi memeriksa apakah tahun sudah ada dalam array `datapenyakit`:
   - Jika tahun sudah ada, indeks tahun ditemukan dan digunakan.
   - Jika tahun belum ada, tahun baru ditambahkan ke array `datapenyakit` dan diinisialisasi.
f. Fungsi memeriksa apakah jenis penyakit sudah ada dalam array `jenispenyakit`:
   - Jika jenis penyakit sudah ada, indeks jenis penyakit ditemukan dan digunakan.
   - Jika jenis penyakit belum ada, jenis penyakit baru ditambahkan ke array `jenispenyakit` dan diinisialisasi.
g. Jumlah penyakit bulanan diperbarui sesuai dengan jenis penyakit dan bulan yang sesuai.
h. Setelah semua entri diiterasi, fungsi akan menghitung dan mengurutkan jenis penyakit terbanyak sampai terkecil untuk setiap tahun.
i. Terakhir, fungsi menampilkan jumlah penyakit per bulan dan total per tahun untuk setiap jenis penyakit.

Proses iterasi melalui riwayat medis memerlukan waktu `O(n)`, di mana `n` adalah jumlah entri dalam `riwayatMedisPasien`. Pencarian tahun dan jenis penyakit dalam array memerlukan waktu `O(m)` dan `O(p)`, di mana `m` adalah jumlah tahun yang berbeda dan `p` adalah jumlah jenis penyakit yang berbeda. Sedangkan untuk pengurutan jenis penyakit dari yang terbanyak memerlukan kompleksitas waktu `O(p log p)` menggunakan quicksort. Dalam kasus terburuk, kompleksitas waktu keseluruhan menjadi `O(n * (m + p) + p log p)`. Penggunaan memori dinamis untuk menyimpan data penyakit tahunan mengakibatkan kompleksitas ruang `O(m * p)`.

c) Fitur Informasi Kontrol Pasien

Fungsi `informasiKontrolPasien` digunakan untuk mencari dan menampilkan informasi kontrol pasien berdasarkan tanggal yang dimasukkan oleh pengguna. Berikut adalah penjelasan implementasi fungsi tersebut:

a. Fungsi meminta pengguna untuk memasukkan tanggal yang ingin dicari dengan format dd mm yyyy.
b. Input tanggal diparsing menggunakan `sscanf` untuk memisahkan hari, bulan, dan tahun.
c. Fungsi memeriksa apakah format tanggal yang dimasukkan valid.

d. Jika format tidak valid, fungsi akan menampilkan pesan kesalahan dan keluar.
e. Fungsi juga memeriksa apakah bulan yang dimasukkan berada dalam rentang 1 hingga 12. Jika tidak, fungsi akan menampilkan pesan kesalahan dan keluar.
f. Tanggal yang dimasukkan oleh pengguna dikonversi menjadi format string yang sesuai dengan format tanggal dalam `riwayatMedisPasien`.
g. Fungsi menggunakan array `bulanStr` untuk mengonversi nomor bulan menjadi nama bulan dalam bahasa Indonesia.
h. Fungsi mengiterasi melalui setiap entri dalam `riwayatMedisPasien` untuk mencari tanggal yang sesuai dengan `tanggalDicariFormatted`.
i. Jika ditemukan entri yang sesuai, fungsi akan menampilkan informasi kontrol pasien seperti nomor, ID pasien, diagnosis, tindakan, kontrol, dan biaya.
j. Jika ada data kontrol pada tanggal yang dicari, fungsi akan menampilkan informasi tersebut dan menampilkan pesan bahwa tidak ada data yang ditemukan.

Proses iterasi melalui `riwayatMedisPasien` memerlukan waktu `O(n)`, di mana `n` adalah jumlah entri dalam `riwayatMedisPasien dengan` Parsing dan validasi tanggal memerlukan waktu konstan `O(1)`. Penggunaan memori untuk variabel lokal seperti `tanggalDicari`, `tanggalDicariFormatted`, dan `bulanStr` adalah konstan `O(1)`.

VI. KESIMPULAN

Pembuatan aplikasi pencatatan pasien dilakukan untuk memudahkan klinik X dalam perekaman pasien-pasien yang datang. Aplikasi ini dimulai dengan membuka tiga file csv yang sudah berisi data pasien, data riwayat pasien, dan data biaya tindakan. Pembacaan file disimpan dalam tiga array berbeda: array data pasien, array riwayat medis, dan array biaya tindakan. Pengelolaan data seperti penambahan, pencarian, penghapusan, dan pemberian informasi dilakukan pada ketiga array tersebut. Lalu, hasil olahan file kemudian ditulis kembali pada file csv.

Setelah membuat aplikasi pencatatan pasien dan melakukan analisis, didapatkan kesimpulan seperti berikut.

1. Pengelolaan data seperti pencatatan, pengubahan, pencarian, dan penghapusan data pasien tidak perlu lagi dilakukan pada sebuah buku karena aplikasi ini sudah memuat semua fitur yang dibutuhkan
2. Masih terdapat kemungkinan human error pada aplikasi ini karena proses pencatatan data masih dilakukan oleh pengguna. Namun, human error dapat terminimalisasikan karena tampilan dari aplikasi yang lebih mudah dibaca dibandingkan dengan buku.
3. Pengaksesan data dilakukan dengan memilih fitur yang diinginkan, lalu memilih data yang ingin diakses. Hal ini membuat pengaksesan data menjadi jauh lebih cepat.
4. Fitur analisis data meliputi laporan keuangan klinik, laporan jumlah pasien dan penyakitnya, serta informasi mengenai pasien yang perlu kembali kontrol sehingga klinik dapat membuat keputusan yang tepat.
5. Seluruh data akan langsung disimpan pada file csv sehingga dapat diakses oleh siapapun.

VII. REFERENSI

[1]. "Command line vs. GUI". Computer Hope. Retrieved 2024-06-17.
[2]. Ritchie, Dennis M. (March 1993). "The Development of the C Language". ACM SIGPLAN Notices. ACM. 28 (3): 201–208. doi:10.1145/155360.155580.

[3]. Darmakusuma, R., Widyanto, D., Brillianshah, E. J., & Kurniawan, I. T. (2023). EL2208 Praktikum Pemecahan Masalah dengan C. Program Studi Teknik Elektro, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

Author 1 short CV and photograph

Rafi Ananta Alden adalah seorang mahasiswa Teknik Elektro di Institut Teknologi Bandung (ITB) yang memiliki prospek akademik yang kuat dan turut aktif dalam beberapa kegiatan kemahasiswaan dan organisasi. Ia kerap kali berkontribusi dalam kepanitiaan untuk sejumlah acara yang diselenggarakan oleh HME ITB dan Kabinet Gema Karsa KM ITB. Beberapa *skill* yang dimiliki meliputi Microsoft Office, Google Workspace, dan C. Ada pun ia juga memiliki kemampuan berbahasa Inggris yang cukup bagus, dibuktikan dengan skor Duolingo English Test (DET) sebesar 140/160

Author 2 short CV and photograph

Muhammad Zaki Fazansyah adalah mahasiswa sarjana Teknik Biomedis di Institut Teknologi Bandung yang memiliki latar belakang kuat dalam manajemen acara dan keterampilan teknis. Ia aktif dalam berorganisasi di IEEE ITB SB. Kemampuannya dalam bidang teknis mencakup MS Office, GSuite, Python, dan C, skor TOEFL iBT 104 dan sertifikat DELF B1 dalam bahasa Prancis.

Author 3 short CV and photograph

Mahardhika Putra Adipratama, seorang mahasiswa Teknik Elektro di Institut Teknologi Bandung, menunjukkan dedikasinya dalam bidang robotika sebagai anggota aktif di Unit Robotika ITB, divisi pemrograman. Dengan skor IELTS sebesar 7.5, ia juga menorehkan prestasi akademik dengan meraih nilai A* dalam matematika dan matematika lanjutan A level Cambridge. ia juga meraih juara pertama di Pesta Sains Nasional IPB dan juara kedua di phi Science Olympiad. Mahardhika memiliki keahlian teknis dalam pemrograman C, C++, dan Python, serta pengalaman dengan framework ROS dan software simulasi robot Gazebo.

Author 4 short CV and photograph

Mohammad Ari Alexander merupakan mahasiswa sarjana Teknik Elektro di Institut Teknologi Bandung yang saat ini menempuh semester 4. Kemampuan teknikalnya meliputi excel, python, C, C++, VHDL, dan PCB design. Ketertarikannya di bidang lain meliputi design thinking dan pembuatan kebijakan, terlihat dari kontribusinya dalam amandemen RUK KM ITB dan di kementrian riset dan sistematisasi sumber daya manusia KM ITB 24/25.

Author 5 short CV and photograph

Aira Ardistya Akbarsyah, mahasiswa sarjana dari Teknik Elektro STEI-R Institut Teknologi Bandung. Saat ini sedang menjalani semester 4 perkuliahan. Kemampuan teknis yang dimilikinya meliputi Office, Solidworks, bahasa Python, C, dan VHDL.