

**LAPORAN TUGAS BESAR  
ALGORITMA DAN PEMROGRAMAN 1 (IF1210)  
PROGRAM “MANAJEMEN RUMAH SAKIT”**



Disusun oleh:

Kelompok K1-I

Ahmad Zaky Robbani (13524045)

Kurt Mikhael Purba (13524065)

Angelina Andra Alanna (13524079)

Fauzan Mohamad Abdul Ghani (13524113)

Amanda Aurellia Salsabilla (13524131)

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
BANDUNG  
2025**

Kelompok kami yang terdiri dari:

1. Ahmad Zaky Robbani - 13524045
2. Kurt Mikhael Purba - 13524065
3. Angelina Andra Alanna - 13524079
4. Fauzan Mohamad Abdul Ghani - 13524113
5. Amanda Aurellia Salsabilla - 13524131

Dengan ini menyatakan bahwa,

*“Saya menyatakan bahwa saya mengerjakan tugas besar ini dengan sejujur-jujurnya, tanpa menggunakan cara yang tidak dibenarkan. Apabila di kemudian hari diketahui saya mengerjakan tugas besar ini dengan cara yang tidak jujur, saya bersedia mendapatkan konsekuensinya, yaitu mendapatkan nilai E pada mata kuliah IF1210 Algoritma dan Pemrograman 1 Semester 2 2024/2025.”*

Laporan ini disusun sebagai bentuk dokumentasi pengembangan Tugas Besar 1 mata kuliah Algoritma dan Pemrograman 1 IF1210 dengan tema pembuatan program manajemen rumah sakit. Program ini dirancang untuk membantu proses manajemen rumah sakit, mulai dari manajemen pasien, dokter, hingga administrasi. Fitur-fitur yang dikembangkan mencakup fungsi-fungsi manajemen dasar seperti pendaftaran, pemeriksaan, transaksi, laporan, serta pengaturan data.

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>3</b>
<b>DAFTAR TABEL.....</b>	<b>8</b>
<b>DAFTAR GAMBAR.....</b>	<b>8</b>
<b>BAB I</b>	
<b>PENDAHULUAN.....</b>	<b>9</b>
1.1 Deskripsi Persoalan.....	10
<b>BAB II</b>	
<b>RANCANGAN SISTEM.....</b>	<b>11</b>
2.1 Rencana Implementasi Fitur Wajib.....	11
Tabel 2.1 Rencana Implementasi Fitur Wajib.....	11
2.2 Pembagian Kerja Anggota.....	15
Tabel 2.2 Pembagian Kerja Anggota.....	15
2.3 Checklist Hasil Rancangan, Implementasi, dan Testing.....	17
Tabel 2.3 Checklist Hasil Rancangan, Implementasi, dan Testing.....	17
2.4 Desain Command.....	18
Tabel 2.4 Desain Command.....	18
2.5 Desain Kamus Data.....	31
Tabel 2.5 Desain Kamus Data.....	31
2.6 Desain Dekomposisi Algoritmik dan Fungsional.....	35
Gambar 2.6.1 Flowchart Login.....	36
Gambar 2.6.2 Flowchart Register.....	37
Gambar 2.6.3 Flowchart Logout.....	38
Gambar 2.6.4 Flowchart Lupa Password.....	39
Gambar 2.6.5 Flowchart Menu & Help.....	40
Gambar 2.6.6 Flowchart Denah Rumah Sakit.....	41
Gambar 2.6.7 Flowchart Lihat User.....	44
Gambar 2.6.8 Flowchart Cari User.....	47
Gambar 2.6.9 Flowchart Lihat Antrian.....	48
Gambar 2.6.10 Flowchart Tambah Dokter.....	49
Gambar 2.6.11 Flowchart Diagnosis.....	50
Gambar 2.6.12 Flowchart Ngobatin.....	51
Gambar 2.6.13 Flowchart Aku boleh pulang ga, dok 😊 ?.....	52
Gambar 2.6.14 Flowchart Daftar Check-Up.....	53
Gambar 2.6.15 Flowchart Antrian Saya!.....	54
Gambar 2.6.16 Flowchart Minum Obat.....	55
Gambar 2.6.17 Flowchart Minum Penawar.....	56
Gambar 2.6.18 Flowchart Exit.....	57
Gambar 2.6.19 Flowchart SkipAntrian.....	58

Gambar 2.6.20 Flowchart Cancel Antrian.....	59
Gambar 2.6.21 Flowchart Lihat Dompet (Bananarich).....	60
Gambar 2.6.22 Flowchart Lihat Keuangan (Banarich).....	61
Gambar 2.6.23 Flowchart Gacha Gaming (Banarich).....	62
Gambar 2.6.24 Flowchart Unassign Dokter (Denah Dinamis).....	63
Gambar 2.6.25 Flowchart Tambah Baris (Denah Dinamis).....	64
Gambar 2.6.26 Flowchart Tambah Kolom (Denah Dinamis).....	64
Gambar 2.6.27 Flowchart Kurangi Baris (Denah Dinamis).....	65
Gambar 2.6.28 Flowchart Kurangi Kolom (Denah Dinamis).....	65
<b>2.7 Spesifikasi dan Notasi Algoritmik.....</b>	<b>65</b>
<b>2.7.1 Kamus Global.....</b>	<b>65</b>
<b>2.7.2 ADT Map.....</b>	<b>67</b>
<b>2.7.3 ADT Matrix.....</b>	<b>68</b>
<b>2.7.4 ADT Obat.....</b>	<b>69</b>
<b>2.7.5 ADT Penyakit.....</b>	<b>70</b>
<b>2.7.6 ADT Queue.....</b>	<b>70</b>
<b>2.7.7 ADT Stack.....</b>	<b>71</b>
<b>2.7.8 ADT User.....</b>	<b>71</b>
<b>2.7.9 ADT Set.....</b>	<b>73</b>
<b>2.7.10 File Eksternal.....</b>	<b>74</b>
<b>2.7.11 Fitur.....</b>	<b>74</b>

**BAB III**  
**PENGUJIAN DAN HASIL.....** **137**

<b>3.1 Dokumentasi Input dan Output Valid.....</b>	<b>137</b>
<b>3.1.1 Login.....</b>	<b>137</b>
Gambar 3.1.1 Antarmuka Login Pengguna.....	137
<b>3.1.2 Register.....</b>	<b>137</b>
Gambar 3.1.2 Antarmuka Register Pasien.....	137
<b>3.1.3 Logout.....</b>	<b>137</b>
Gambar 3.1.3 Antarmuka Logout Pengguna.....	137
<b>3.1.4 Lupa Password.....</b>	<b>138</b>
Gambar 3.1.4 Antarmuka Lupa Password.....	138
<b>3.1.5 Menu dan Help.....</b>	<b>138</b>
Gambar 3.1.5.1 Antarmuka Menu dan Help.....	138
Gambar 3.1.5.2.....	139
Antarmuka Menu dan Help Pasien.....	139
Gambar 3.1.5.3 Antarmuka Menu dan Help Dokter.....	139
Gambar 3.1.5.4 Antarmuka Menu dan Help Manager.....	140
<b>3.1.6 Lihat Denah.....</b>	<b>140</b>
Gambar 3.1.6 Antarmuka Lihat Denah.....	140
<b>3.1.7 Lihat User.....</b>	<b>141</b>

Gambar 3.1.7 Antarmuka Lihat User.....	141
3.1.8 Cari User.....	141
Gambar 3.1.8 Antarmuka Cari User.....	141
3.1.9 Lihat Antrean.....	142
Gambar 3.1.9 Antarmuka Lihat Antrean.....	143
3.1.10 Tambah Dokter.....	143
Gambar 3.1.10 Antarmuka Lihat Antrean untuk ruangan tidak berpasien, ruangan kosong tidak ditampilkan.....	143
3.1.11 Diagnosis.....	143
Gambar 3.1.11 Antarmuka Diagnosis.....	143
3.1.12 Ngobatin.....	143
Gambar 3.1.12 Antarmuka Ngobatin.....	143
3.1.13 Aku boleh pulang ga, dok?.....	144
Gambar 3.1.13.....	145
Antramuka Aku boleh pulang ga, dok?.....	145
3.1.14 Daftar Check-Up.....	145
Gambar 3.1.14 Antarmuka Daftar Check-up.....	146
3.1.15 Antrean Saya.....	146
Gambar 3.1.15 Antarmuka Antrean Saya.....	146
3.1.16 Minum Obat.....	146
Gambar 3.1.16 Antarmuka Minum Obat\.....	146
3.1.17 Minum Penawar.....	146
Gambar 3.1.17 Antarmuka Minum Penawar.....	147
3.1.18 Exit.....	147
Gambar 3.1.18 Antarmuka Exit.....	147
Gambar 3.1.19.1 Antarmuka Skip Antrean.....	147
Gambar 3.1.19.2 Antarmuka Cancel Antrean.....	147
Gambar 3.1.20 Antarmuka Denah Dinamis.....	148
Gambar 3.1.21 Antarmuka Aura.....	149
3.1.21 Bonus BananarichGambar 3.1.22.1 Antarmuka Gacha.....	149
Gambar 3.1.22.2 Antarmuka Lihat Dompet.....	149
Gambar 3.1.22.3 Antarmuka Lihat Dompet.....	149
3.1.22 Bonus Dead or Alive.....	150
Gambar 3.1.22.3 Antarmuka User Mati.....	150
3.2 Dokumentasi Input dan Output Tidak Valid.....	150
Berikut merupakan beberapa contoh kasus input dan output tidak valid di berbagai fitur.....	150
Gambar 3.2.1 Tampilan antarmuka Daftar Check-up yang menampilkan pesan kesalahan dan mengulang input hingga data yang dimasukkan valid.....	150
Gambar 3.2.2 Tampilan antramuka Login yang menampilkan pesan kesalahan dan mengulang input hingga username/password sesuai.....	151
Gambar 3.2.3 Tampilan antarmuka saat Pasien Meninggal (disebabkan kesalahan dalam urutan meminum obat).....	151

Gambar 3.2.4 Sistem Menolak Login dari Akun Pasien Meninggal.....	151
Gambar 3.2.5 Tampilan antarmuka saat Mendiagnosis Pasien yang telah Terdiagnosa.....	152
Gambar 3.2.6 Tampilan Antarmuka saat Ngobatin Pasien yang telah Mendapatkan Obat.....	152
Gambar 3.2.7 Tampilan Antarmuka saat Skip Antrean.....	152
Gambar 3.2.8 Tampilan Antarmuka saat Cancel Antrean.....	152
Gambar 3.2.8 Tampilan Antarmuka saat Minum Penawar.....	153
3.3 Penjelasan Hasil Uji.....	153
<b>LAMPIRAN.....</b>	<b>154</b>
1. Hasil Pindai Form Asistensi.....	154

## DAFTAR TABEL

Tabel 2.1 Rencana Implementasi Fitur Wajib.....	11
Tabel 2.2 Pembagian Kerja Anggota.....	15
Tabel 2.3 Checklist Hasil Rancangan, Implemenetasi, dan Testing.....	17
Tabel 2.4 Desain Command.....	18
Tabel 2.5 Desain Kamus Data.....	31

## DAFTAR GAMBAR

Gambar 2.6.1 Flowchart Login.....	35
Gambar 2.6.2 Flowchart Register.....	36
Gambar 2.6.3 Flowchart Logout.....	37
Gambar 2.6.4 Flowchart Lupa Password.....	38
Gambar 2.6.5 Flowchart Menu & Help.....	39
Gambar 2.6.6 Flowchart Denah Rumah Sakit.....	40
Gambar 2.6.7 Flowchart Lihat User.....	43
Gambar 2.6.8 Flowchart Cari User.....	46
Gambar 2.6.9 Flowchart Lihat Antrian.....	47
Gambar 2.6.10 Flowchart Tambah Dokter.....	48
Gambar 2.6.11 Flowchart Diagnosis.....	49
Gambar 2.6.12 Flowchart Ngobatin.....	50
Gambar 2.6.13 Flowchart Aku boleh pulang ga, dok 😊 ?.....	51
Gambar 2.6.14 Flowchart Daftar Check-Up.....	52
Gambar 2.6.15 Flowchart Antrian Saya!.....	53
Gambar 2.6.16 Flowchart Minum Obat.....	54
Gambar 2.6.17 Flowchart Minum Penawar.....	55
Gambar 2.6.18 Flowchart Exit.....	56
Gambar 2.6.19 Flowchart SkipAntrian.....	57
Gambar 2.6.20 Flowchart Cancel Antrian.....	58
Gambar 2.6.21 Flowchart Lihat Dompet (Bananrich).....	59
Gambar 2.6.22 Flowchart Lihat Keuangan (Bananrich).....	60
Gambar 2.6.23 Flowchart Gacha Gaming (Bananrich).....	61
Gambar 2.6.24 Flowchart Unassign Dokter (Denah Dinamis).....	62
Gambar 2.6.25 Flowchart Tambah Baris (Denah Dinamis).....	63
Gambar 2.6.26 Flowchart Tambah Kolom (Denah Dinamis).....	63
Gambar 2.6.27 Flowchart Kurangi Baris (Denah Dinamis).....	64
Gambar 2.6.28 Flowchart Kurangi Kolom (Denah Dinamis).....	64
Gambar 3.1.1 Antarmuka Login Pengguna.....	136

Gambar 3.1.2 Antarmuka Register Pasien.....	136
Gambar 3.1.3 Antarmuka Logout Pengguna.....	136
Gambar 3.1.4 Antarmuka Lupa Password.....	137
Gambar 3.1.5.1 Antarmuka Menu dan Help.....	137
Gambar 3.1.5.2 Antarmuka Menu dan Help Pasien.....	138
Gambar 3.1.5.3 Antarmuka Menu dan Help Dokter.....	138
Gambar 3.1.5.4 Antarmuka Menu dan Help Manager.....	139
Gambar 3.1.6 Antarmuka Lihat Denah.....	139
Gambar 3.1.7 Antarmuka Lihat User.....	140
Gambar 3.1.8 Antarmuka Cari User.....	140
Gambar 3.1.9 Antarmuka Lihat Antrian.....	142
Gambar 3.1.10 Antarmuka Lihat Antrian untuk ruangan tidak berpasien, ruangan kosong tidak ditampilkan.....	142
Gambar 3.1.11 Antarmuka Diagnosis.....	142
Gambar 3.1.12 Antarmuka Ngobatin.....	142
Gambar 3.1.13 Antriamuka Aku boleh pulang ga, dok?.....	144
Gambar 3.1.14 Antarmuka Daftar Check-up.....	145
Gambar 3.1.15 Antarmuka Antrian Saya.....	145
Gambar 3.1.16 Antarmuka Minum Obat\.....	145
Gambar 3.1.17 Antarmuka Minum Penawar.....	146
Gambar 3.1.18 Antarmuka Exit.....	146
Gambar 3.1.19.1 Antarmuka Skip Antrian.....	146
Gambar 3.1.19.2 Antarmuka Cancel Antrian.....	146
Gambar 3.1.20 Antarmuka Denah Dinamis.....	147
Gambar 3.1.21 Antarmuka Aura.....	148
Gambar 3.1.22.2 Antarmuka Lihat Dompet.....	148
Gambar 3.1.22.3 Antarmuka Lihat Dompet.....	148
Gambar 3.1.22.3 Antarmuka User Mati.....	149
Gambar 3.2.1 Tampilan antarmuka Daftar Check-up yang menampilkan pesan kesalahan dan mengulang input hingga data yang dimasukkan valid.....	149
Gambar 3.2.2 Tampilan antriamuka Login yang menampilkan pesan kesalahan dan mengulang input hingga username/password sesuai.....	150
Gambar 3.2.3 Tampilan antarmuka saat Pasien Meninggal (disebabkan kesalahan dalam urutan meminum obat).....	150
Gambar 3.2.4 Sistem Menolak Login dari Akun Pasien Meninggal.....	150
Gambar 3.2.5 Tampilan antarmuka saat Mendiagnosis Pasien yang telah Terdiagnosa.....	151
Gambar 3.2.6 Tampilan Antarmuka saat Ngobatin Pasien yang telah Mendapatkan Obat.....	151
Gambar 3.2.7 Tampilan Antarmuka saat Skip Antrian.....	151
Gambar 3.2.8 Tampilan Antarmuka saat Cancel Antrian.....	151
Gambar 3.2.8 Tampilan Antarmuka saat Minum Penawar.....	151

# BAB I

## PENDAHULUAN

### 1.1 Deskripsi Persoalan

Tugas besar ini mengharuskan kita untuk membuat sistem manajemen rumah sakit Nimon dengan 18 fitur utama (F01–F18) yang mencakup seluruh alur kerja rumah sakit, mulai dari login sampai exit. Berikut penjelasan masing-masing fungsionalitas utama.

Pada bagian Login dan Manajemen Akun, sistem harus mampu membedakan akses berdasarkan peran pengguna, yaitu Manager (yang sudah terdaftar sejak awal), Dokter (ditambahkan oleh Manager), dan Pasien (mendaftar sendiri). Fitur login (F01) harus menangani kesalahan input dengan memberikan pesan yang jelas, seperti "Username atau password salah!" tanpa menyebabkan program crash. Untuk pendaftaran pasien (F02), username harus unik dan diverifikasi menggunakan Set. Selain itu, sistem menyediakan opsi logout (F03) agar pengguna dapat berganti akun, serta fitur lupa password (F04) yang menggunakan kode unik berbasis Run-Length Encoding dari username, seperti "Je2fr2ey" untuk username "Jeffrey".

Pada bagian Fitur Dasar, mencakup menu bantuan (F05) yang menampilkan daftar perintah sesuai peran pengguna, denah rumah sakit (F06) dalam bentuk matriks untuk IF/EL/EB, serta fitur pencarian dan pengurutan user (F07–F08) menggunakan binary search untuk efisiensi. Fitur antrian (F09) menampilkan daftar pasien yang mengantri per dokter.

Pada kategori Fitur Dokter dan Pasien, Manager dapat menambahkan dokter baru (F10) dan menempatkannya di ruangan tertentu. Dokter dapat mendiagnosis penyakit pasien (F11) menggunakan data dari penyakit.csv dan memberikan resep obat (F12) berdasarkan urutan yang tepat dari obat\_penyakit.csv. Pasien dapat mengecek status penyembuhan (F13) dengan memverifikasi konsumsi obat yang benar, mendaftar check-up (F14) menggunakan Queue, serta melihat posisi antrian (F15). Selain itu, pasien dapat mengonsumsi obat (F16) yang disimpan dalam Stack dan menggunakan penawar (F17) jika terjadi kesalahan urutan minum obat. Terakhir, program menyediakan opsi exit (F18) dengan konfirmasi penyimpanan data.

## BAB II

### RANCANGAN SISTEM

#### 2.1 Rencana Implementasi Fitur Wajib

*Tabel 2.1 Rencana Implementasi Fitur Wajib*

Fitur	Implementasi ADT	Deskripsi Implementasi	Alasan Implementasi
F01 - Login	ADT Sederhana	Memverifikasi username dan password user yang disimpan di list.	List memudahkan akses dan pencarian user.
F02 - Register	ADT List Dinamis dan ADT Set	Menambahkan user baru ke List dan Set untuk memastikan data unik.	ADT list dinamis dan set dikombinasikan untuk menyimpan data “tanpa batas” dalam 2 array yang terurut berdasarkan ID dan berdasarkan username.
F03 - Logout	ADT Sederhana	Menghapus status login user.	Cukup hanya dengan menggunakan ADT Sederhana.
F04 - Lupa Password	ADT List	Mencari user di List lalu mereset password.	List mendukung pencarian data user.
F05 - Menu dan Help	ADT Sederhana	Menampilkan menu utama melalui command help.	Cukup hanya dengan menggunakan ADT Sederhana.
F06 - Denah Rumah Sakit	ADT Sederhana, Matriks, Queue	Rumah sakit direpresentasikan sebagai matriks ruangan. Setiap ruangan memiliki komponen User, Queue of User, dan kapasitas.	Setiap ruangan memiliki perilaku yang sama sehingga diabstraksi menjadi struct Ruangan. Untuk rumah sakit, diabstraksi menjadi matriks Ruangan.

			Queue of User menyimulasikan perilaku antrean dengan baik.
F07 - Lihat User	ADT List	Menampilkan semua user dari List.	List mendukung pencarian (linear/binary search).
F08 - Cari User	ADT List	Mencari user tertentu dalam List.	List mendukung pencarian (linear/binary search).
F09 - Lihat Antrean	ADT Matrix dan ADT Queue	Menampilkan antrean user di tiap Matriks Ruangan.	Matrix digunakan untuk menampilkan ruangan. Queue digunakan untuk meniru antrean.
F10 - Tambah Dokter	ADT List, ADT Set, dan ADT Matrix.	Menambahkan dokter baru ke List, Set, dan update Matrix.	List digunakan untuk menyimpan data, Set digunakan untuk memastikan username yang dimasukkan adalah unik, dan Matrix digunakan untuk update ruangan.
F11 - Diagnosis	ADT List	Menyimpan hasil diagnosis ke List.	List digunakan untuk menyimpan hasil diagnosis setiap user.
F12 - Ngobatin	ADT Map	Memetakan user ke status pengobatan di Map.	Map memudahkan akses status berdasarkan user.
F13 - Aku boleh pulang ga, dok 😊?	ADT List dan ADT Stack	Menyimpan urutan minum obat di Stack dan melakukan skema validasi apakah semua obat sudah diminum dengan urutan yang	Stack digunakan untuk menyimpan urutan minum obat.

		benar.	
F14 - Daftar Check-Up	ADT Map, ADT Queue, ADT Linked List	Menyimpan daftar check-up user di Map, Queue, dan Linked List.	Map digunakan untuk menyimpan data user, Queue digunakan untuk menentukan posisi antrean, dan penggunaan Linked List mempermudah penambahan secara dinamis.
F15 - Antrean Saya!	ADT Map dan ADT Queue	Menampilkan posisi antrean user dari Map dan Queue.	Map digunakan untuk menemukan data user dan Queue digunakan untuk menentukan posisi antrean.
F16 - Minum Obat	ADT List dan ADT Stack	Menyimpan riwayat urutan minum obat ke Stack.	Stack digunakan untuk menyimpan riwayat urutan minum obat.
F17 - Minum Penawar	ADT List dan ADT Stack	Menyimpan riwayat minum penawar ke Stack.	Stack digunakan untuk mengakses riwayat urutan obat paling terbaru yang sudah diminum oleh user untuk dikeluarkan.
F18 - Exit	File Eksternal	Menyimpan semua data ke file eksternal sebelum keluar.	File Eksternal digunakan untuk menyimpan perubahan data secara permanen.
B02 - Denah Dinamis	ADT Matrix	Denah ruangan bisa ditambah baris atau kolomnya, baris dan kolomnya juga bisa dikurangi jika tidak ada pasien, dokter bisa di-unassign untuk dicabut dari	Matriks statik cukup untuk memberikan kebebasan akan layout rumah sakit dengan lahan terbatas.

		ruangan jika tidak sedang melayani pasien.	
B03 - Aura!	ADT Sederhana dan ADT List	Setiap dokter memiliki aura yang dapat bertambah ketika berhasil memulangkan pasien dan berkurang ketika pasiennya mati.	ADT sederhana untuk mengabstraksi user, sedangkan ADT List untuk menyimpan data selama program bekerja.
B04 - Banarich!	ADT Sederhana dan ADT List	Pasien dan dokter memiliki dompetnya masing-masing, sedangkan manajer mengelola keuangan rumah sakit. Dompet pasien bertambah jika bermain gacha dan berkurang jika daftar checkup. Dompet dokter bertambah jika pasien mendaftar checkup dengan dirinya. Harga checkup ditentukan oleh aura.	ADT sederhana untuk mengabstraksi user, sedangkan ADT List untuk menyimpan data yang di-update berkala selama program berlangsung.
B05 - Dead or Alive ?!	ADT User dan ADT List	Ketika pasien salah minum obat dan meminum penawar, nyawa pasien akan berkurang satu (defaultnya 3). Ketika nyawa pasien bernilai 0 maka pasien akan dinyatakan meninggal dan akun pasien akan dihapus dari database.	ADT-User berguna untuk menyimpan data nyawa pasien (Terletak di dalam tipe data user). ADT-User juga berguna ketika kita ingin mengupdate isi datatbase dengan data yang baru ,khususnya ketika ada pasien yang meninggal maka data tentang pasien

			tersebut akan dihapus dari database.
B06 - Mainin Antrean	ADT Queue dan ADT Matrix	Pasien yang sudah ada dalam antrean memiliki pilihan untuk mengcancel antrean atau mengeskip antrean. Ketika pasien skip antrean, maka pasien akan diletakan di paling depan antrean ruangan. Ketika pasien cancel antrean maka pasien akan dihapus dari queue antrean ruangan tersebut.	ADT-Matriks berguna untuk mengupdate antrian di setiap ruangan dokter setiap ada pasien yang skip ataupun cancel antrean. ADT-Queue berguna untuk melakukan manipulasi pada antrean user yang ada.

## 2.2 Pembagian Kerja Anggota

Tabel 2.2 Pembagian Kerja Anggota

Fitur	Implementasi	NIM Desainer	NIM Coder	NIM Tester
F01 - Login	13524079 13524065	13524079 13524065	13524079	13524079
F02 - Register	13524079	13524079 13524065	13524079	13524079
F03 - Logout	13524113	13524113 13524065	13524113	13524113
F04 - Lupa Password	13524065	13524065	13524065	13524065
F05 - Menu dan Help	13524131	13524131 13524065	13524131	13524131
F06 - Denah Rumah Sakit	13524045	13524045	13524045	13524045
F07 - Lihat	13524131	13524131	13524131	13524131

User		13524045	13524045	
F08 - Cari User	13524113	13524113 13524045	13524113	13524113
F09 - Lihat Antrean	13524045	13524045	13524045	13524045
F10 - Tambah Dokter	13524045	13524045	13524045	13524045
F11 - Diagnosis	13524065	13524065	13524065	13524065
F12 - Ngobatin	13524065	13524065	13524065	13524065 13524045
F13 - Aku boleh pulang ga, dok 😊?	13524131	13524131 13524065	13524131	13524131 13524065
F14 - Daftar Check-Up	13524045	13524045 13524065	13524045	13524045
F15 - Antrean Saya!	13524079	13524079	13524079	13524079
F16 - Minum Obat	13524113	13524113 13524065	13524113	13524113 13524065
F17 - Minum Penawar	13524079	13524079	13524079	13524079 13524065
F18 - Exit	13524065	13524065	13524065	13524065
B02 - Denah Dinamis	13524045	13524045	13524045	13524045
B03 - Aura	13524045	13524045	13524045	13524045
B04 - Banarich	13524045	13524045 13524065	13524045	13524045
B05 - Dead or Alive	13524065	13524065	13524065	13524065
B06 - Mainan Antrean	13524065	13524065	13524065	13524065

### 2.3 Checklist Hasil Rancangan, Implementasi, dan Testing

Tabel 2.3 Checklist Hasil Rancangan, Implementasi, dan Testing

Fitur	Desain	Implementasi	Testing
F01 - Login	V	V	V
F02 - Register	V	V	V
F03 - Logout	V	V	V
F04 - Lupa Password	V	V	V
F05 - Menu dan Help	V	V	V
F06 - Denah Rumah Sakit	V	V	V
F07 - Lihat User	V	V	V
F08 - Cari User	V	V	V
F09 - Lihat Antrean	V	V	V
F10 - Tambah Dokter	V	V	V
F11 - Diagnosis	V	V	V
F12 - Ngobatin	V	V	V
F13 - Aku boleh pulang ga, dok 😊?	V	V	V
F14 - Daftar Check-Up	V	V	V
F15 - Antrean Saya!	V	V	V
F16 - Minum Obat	V	V	V
F17 - Minum Penawar	V	V	V
F18 - Exit	V	V	V
B02 - Denah	V	V	V

dinamis			
B03 - Aura	V	V	V
B04 - Bananrich	V	V	V
B05 - Dead or Alive	V	V	V
B06 - Mainan Antrean	V	V	V

## 2.4 Desain Command

Tabel 2.4 Desain Command

Fitur	Command
F01 - Login	# Kasus 1: LOGIN sebagai Manager <code>&gt;&gt;&gt; login</code>  Username: <b>userI</b> Password: <b>pass999</b>
	# Kasus 2: LOGIN sebagai Dokter <code>&gt;&gt;&gt; login</code>  Username: <b>userI</b> Password: <b>pass999</b>
	# Kasus 3: LOGIN sebagai Pasien <code>&gt;&gt;&gt; login</code>  Username: <b>userB</b> Password: <b>pass22</b>
	# Kasus 4: Username yang dimasukkan tidak terdaftar <code>&gt;&gt;&gt; login</code> Username: <b>userZ</b> User dengan nama userZ tidak ditemukan! Tidak ada Manager, Dokter, atau Pasien yang bernama userZ! Username:
	# Kasus 5: Password salah <code>&gt;&gt;&gt; login</code> Username: <b>userB</b> Password: <b>pass33</b> Username atau password salah untuk pengguna yang bernama userB! Password:
F02 - Register	# Kasus 1: REGISTER sebagai Pasien

	<pre>&gt;&gt;&gt; register Username: andre Password: rahasia123  Pasien andre berhasil ditambahkan dengan ID 3!</pre>
	<pre># Kasus 2: Sudah LOGIN sebagai user lain &gt;&gt;&gt; register Kamu sudah LOGIN sebagai dokter userJ! Silakan LOGOUT sebelum REGISTER sebagai pengguna lain.</pre>
	<pre># Kasus 3: Username tidak valid &gt;&gt;&gt; register Username: user123 Registrasi gagal! Username hanya boleh berisi huruf (tanpa angka atau simbol). Username:</pre>
	<pre># Kasus 4: Username sudah digunakan &gt;&gt;&gt; register Username: andre Registrasi gagal! Username sudah terdaftar. Username:</pre>
F03 - Logout	<pre># Kasus 1: Dalam keadaan LOGIN &gt;&gt;&gt; logout Berhasil LOGOUT! Silakan LOGIN kembali untuk mengakses fitur rumah sakit.</pre>
	<pre># Kasus 2: Dalam keadaan belum LOGIN &gt;&gt;&gt; logout Kamu belum LOGIN sebagai role apapun.</pre>
F04 - Lupa Password	<pre># Kasus 1: Username terdaftar &gt;&gt;&gt; lupa_password Username: userB Kode Unik: userB Halo pasien userB, silakan daftarkan ulang password anda! Password Baru: pass222 Password berhasil diubah! Silakan LOGIN dengan password baru.</pre>
	<pre># Kasus 2: Username tidak terdaftar &gt;&gt;&gt; lupa_password Username: userZ User dengan nama userZ tidak ditemukan! Username tidak terdaftar!</pre>
	<pre># Kasus 3: Kode unik salah &gt;&gt;&gt; lupa_password Username: userA Kode Unik: u2serA</pre>

	Kode unik salah!
F05 - Menu dan Help	<pre># Kasus 1: Dalam keadaan belum LOGIN &gt;&gt;&gt; help Kamu belum LOGIN sebagai role apapun. Silahkan LOGIN terlebih dahulu.  FITUR YANG TERSEDIA 1. LOGOUT : Keluar dari akun yang sedang digunakan 2. REGISTER : Membuat akun baru sebagai Pasien 3. LUPA_PASSWORD : Reset password jika lupa 4. HELP : Menampilkan menu bantuan ini 5. EXIT : Keluar dari program  FOOTNOTE 1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar 2. Jangan lupa untuk memasukkan input yang valid 3. Gunakan command yang sesuai dengan role Anda</pre>
	<pre># Kasus 2: LOGIN sebagai Manager &gt;&gt;&gt; help Halo Manager userI. Kenapa kamu memanggil command HELP? Kamu manager, tapi yasudahlah kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:  FITUR MANAGER 1. LOGOUT : Keluar dari akun yang sedang digunakan 2. LIHAT_DENAH : Melihat denah rumah sakit 3. LIHAT_USER : Melihat data seluruh pengguna 4. CARI_USER : Mencari data pengguna 5. LIHAT_ANTREAN : Melihat rincian seluruh ruangan tidak kosong 6. TAMBAH_DOKTER : Mendaftarkan Dokter baru ke sistem 7. ASSIGN_DOKTER : Menempatkan Dokter ke ruangan 8. HELP : Menampilkan menu bantuan ini 9. EXIT : Keluar dari program  FOOTNOTE 1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar 2. Jangan lupa untuk memasukkan input yang valid 3. Gunakan command yang sesuai dengan role Anda</pre>
	<pre># Kasus 3: LOGIN sebagai Dokter &gt;&gt;&gt; help Halo, Dokter userJ!. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:</pre>

	<p style="text-align: center;">FITUR DOKTER</p> <table> <tbody> <tr><td>1. LOGOUT</td><td>: Keluar dari akun yang sedang digunakan</td></tr> <tr><td>2. LIHAT_DENAH</td><td>: Melihat denah rumah sakit</td></tr> <tr><td>3. DIAGNOSIS</td><td>: Melakukan diagnosis penyakit Pasien</td></tr> <tr><td>4. NGOBATIN</td><td>: Memberi obat kepada Pasien</td></tr> <tr><td>5. HELP</td><td>: Menampilkan menu bantuan ini</td></tr> <tr><td>6. EXIT</td><td>: Keluar dari program</td></tr> </tbody> </table> <p style="text-align: center;">FOOTNOTE</p> <ol style="list-style-type: none"> <li>1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar</li> <li>2. Jangan lupa untuk memasukkan input yang valid</li> <li>3. Gunakan command yang sesuai dengan role Anda</li> </ol>	1. LOGOUT	: Keluar dari akun yang sedang digunakan	2. LIHAT_DENAH	: Melihat denah rumah sakit	3. DIAGNOSIS	: Melakukan diagnosis penyakit Pasien	4. NGOBATIN	: Memberi obat kepada Pasien	5. HELP	: Menampilkan menu bantuan ini	6. EXIT	: Keluar dari program						
1. LOGOUT	: Keluar dari akun yang sedang digunakan																		
2. LIHAT_DENAH	: Melihat denah rumah sakit																		
3. DIAGNOSIS	: Melakukan diagnosis penyakit Pasien																		
4. NGOBATIN	: Memberi obat kepada Pasien																		
5. HELP	: Menampilkan menu bantuan ini																		
6. EXIT	: Keluar dari program																		
# Kasus 4: LOGIN sebagai Pasien <b>&gt;&gt;&gt; help</b> Selamat datang, userB. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:	<p style="text-align: center;">FITUR PASIEN</p> <table> <tbody> <tr><td>1. LOGOUT</td><td>: Keluar dari akun yang sedang digunakan</td></tr> <tr><td>2. LIHAT_DENAH</td><td>: Melihat denah rumah sakit</td></tr> <tr><td>3. BOLEH_PULANG</td><td>: Konsultasi dengan Dokter terkait kepulangan</td></tr> <tr><td>4. DAFTAR_CHECKUP</td><td>: Mendaftarkan diri untuk pemeriksaan Dokter</td></tr> <tr><td>5. STATUS_ANTREAN</td><td>: Melihat status antrean</td></tr> <tr><td>6. MINUM_OBAT</td><td>: Meminum obat yang diberikan Dokter</td></tr> <tr><td>7. MINUM_PENAWAR</td><td>: Mengeluarkan obat terakhir yang diminum</td></tr> <tr><td>8. HELP</td><td>: Menampilkan menu bantuan ini</td></tr> <tr><td>9. EXIT</td><td>: Keluar dari program</td></tr> </tbody> </table> <p style="text-align: center;">FOOTNOTE</p> <ol style="list-style-type: none"> <li>1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar</li> <li>2. Jangan lupa untuk memasukkan input yang valid</li> <li>3. Gunakan command yang sesuai dengan role Anda</li> </ol>	1. LOGOUT	: Keluar dari akun yang sedang digunakan	2. LIHAT_DENAH	: Melihat denah rumah sakit	3. BOLEH_PULANG	: Konsultasi dengan Dokter terkait kepulangan	4. DAFTAR_CHECKUP	: Mendaftarkan diri untuk pemeriksaan Dokter	5. STATUS_ANTREAN	: Melihat status antrean	6. MINUM_OBAT	: Meminum obat yang diberikan Dokter	7. MINUM_PENAWAR	: Mengeluarkan obat terakhir yang diminum	8. HELP	: Menampilkan menu bantuan ini	9. EXIT	: Keluar dari program
1. LOGOUT	: Keluar dari akun yang sedang digunakan																		
2. LIHAT_DENAH	: Melihat denah rumah sakit																		
3. BOLEH_PULANG	: Konsultasi dengan Dokter terkait kepulangan																		
4. DAFTAR_CHECKUP	: Mendaftarkan diri untuk pemeriksaan Dokter																		
5. STATUS_ANTREAN	: Melihat status antrean																		
6. MINUM_OBAT	: Meminum obat yang diberikan Dokter																		
7. MINUM_PENAWAR	: Mengeluarkan obat terakhir yang diminum																		
8. HELP	: Menampilkan menu bantuan ini																		
9. EXIT	: Keluar dari program																		
F06 - Denah Rumah Sakit	# Kasus 1: Dalam keadaan belum LOGIN <b>&gt;&gt;&gt; lihat_denah</b> LOGIN untuk melihat denah!  # Kasus 2: Lihat denah rumah sakit <b>&gt;&gt;&gt; lihat_denah</b> +---+---+---+   A1   A2   A3   +---+---+---+																		

	<pre>  B1   B2   B3   +---+---+---+</pre>
	<pre># Kasus 3: Ruangan terdapat Dokter, Pasien dan antrean Pasien &gt;&gt;&gt; lihat_ruangan Masukkan kode ruangan: A1 --- Informasi Ruangan A1 --- Kapasitas      : 3 Dokter        : userJ Pasien di dalam ruangan: 1. userB 2. userC 3. userA Antrean pasien: 1. userP 2. userT</pre>
	<pre># Kasus 4: Ruangan terdapat Dokter dan Pasien tetapi tidak ada antrean &gt;&gt;&gt; lihat_ruangan Masukkan kode ruangan: A2 --- Informasi Ruangan A2 --- Kapasitas      : 3 Dokter        : userK Pasien di dalam ruangan: 1. userD 2. userE Antrean pasien: Tidak ada pasien dalam antrean saat ini.</pre>
	<pre># Kasus 5: Ruangan kosong &gt;&gt;&gt; lihat_ruangan Masukkan kode ruangan: B2 --- Informasi Ruangan B2 --- Kapasitas      : 3 Dokter        : - Pasien di dalam ruangan: Tidak ada pasien di dalam ruangan saat ini.</pre>
	<pre># Kasus 6: Kode ruangan tidak valid &gt;&gt;&gt; lihat_ruangan Masukkan kode ruangan: A4 Ruangan dengan kode A4 tidak ditemukan!</pre>
F07 - Lihat User	<pre># Kasus 1: Manajer melihat semua pengguna &gt;&gt;&gt; lihat_user Urutkan berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 1  Urutan sort?</pre>

	<pre> 1. ASC (A-Z) 2. DESC (Z-A) &gt;&gt;&gt; Pilihan : 1  Menampilkan seluruh pengguna dalam sistem ID   Nama        Role       Penyakit ----- 1   userA       Pasien     Influenza 2   userB       Dokter     - 3   userC       Manajer    Diabetes </pre>
	<pre> # Kasus 2: Manajer hanya melihat Dokter &gt;&gt;&gt; lihat_dokter Urutkan berdasarkan? 1. ID 2. Nama 3. Aura &gt;&gt;&gt; Pilihan: 3  Urutan sort? 1. ASC (A-Z) 2. DESC (Z-A) &gt;&gt;&gt; Pilihan: 1  ID   Nama        Aura ---+-----+--- 19   userS       0 14   userN       0 11   userK       1 13   userM       2 12   userL       2 15   userO       3 10   userJ       3 </pre>
	<pre> # Kasus 4: Tidak ada data pengguna &gt;&gt;&gt; lihat_user Tidak ada pengguna yang terdaftar dalam sistem! </pre>
	<pre> # Kasus 5: Bukan Manajer yang mengakses &gt;&gt;&gt; lihat_user Hanya manajer yang dapat mengakses fitur ini! </pre>
F08 - Cari User	<pre> # Kasus 1: Mencari data user berdasarkan ID &gt;&gt;&gt; cari_user Cari berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 1 &gt;&gt;&gt; masukkan nomor ID : 1 Menampilkan pengguna dengan ID 1 ID   Nama        Role       Penyakit -----+ </pre>

	<pre> 1   userA   pasien   </pre>
	<pre> # Kasus 2: ID tidak ditemukan &gt;&gt;&gt; cari_user Cari berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 1 &gt;&gt;&gt; masukkan nomor ID : 50 User dengan id 50 tidak ditemukan! Tidak ditemukan pengguna dengan ID 50. </pre>
	<pre> # Kasus 3: Mencari data user berdasarkan nama &gt;&gt;&gt; cari_user Cari berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 2 &gt;&gt;&gt; Masukkan nama : userB Menampilkan pengguna dengan nama userB ID   Nama        Role        Penyakit ----- 2   userB       pasien     </pre>
	<pre> # Kasus 4: Nama tidak ditemukan &gt;&gt;&gt; cari_user Cari berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 2 &gt;&gt;&gt; Masukkan nama : kebin User dengan nama kebin tidak ditemukan! Tidak ditemukan pengguna dengan nama kebin. </pre>
	<pre> # Kasus 5: Mencari data Pasien &gt;&gt;&gt; cari_pasien Cari pasien berdasarkan? 1. ID 2. Nama 3. Penyakit &gt;&gt;&gt; Pilihan : 3 Masukkan nama penyakit : influenza Tidak ditemukan pasien dengan penyakit influenza </pre>
	<pre> # Kasus 6: Mencari data Dokter &gt;&gt;&gt; cari_dokter Cari dokter berdasarkan? 1. ID 2. Nama &gt;&gt;&gt; Pilihan : 2 &gt;&gt;&gt; Masukkan nama : userJ Menampilkan dokter dengan nama userJ ID   Nama </pre>

	<pre>----- 10   userJ</pre>																		
F09 - Lihat Antrean	<pre># Kasus 1: Role bukan manager &gt;&gt;&gt; lihat_antrean Anda tidak memiliki akses untuk fitur lihat antrean!</pre> <hr/> <pre># Kasus 2: Menampilkan denah dan detail antrean &gt;&gt;&gt; LIHAT ANTREAN</pre> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td style="text-align: center;">1</td><td style="text-align: center;">2</td><td style="text-align: center;">3</td></tr> <tr><td style="text-align: center;">A</td><td style="text-align: center;">A1</td><td style="text-align: center;">A2</td><td style="text-align: center;">A3</td><td style="text-align: center;"> </td></tr> <tr><td style="text-align: center;">B</td><td style="text-align: center;">B1</td><td style="text-align: center;">B2</td><td style="text-align: center;">B3</td><td style="text-align: center;"> </td></tr> <tr><td style="text-align: center;"> </td><td style="text-align: center;"> </td><td style="text-align: center;"> </td><td style="text-align: center;"> </td><td style="text-align: center;">+-----+</td></tr> </table> <pre>--- Informasi Ruangan A1 --- Kapasitas : 3 Dokter : userJ Pasien di dalam ruangan: 1. userB 2. userC 3. userA Antrean pasien: 1. userP 2. userT</pre> <hr/> <pre># Kasus antrean kosong --- Informasi Ruangan A2 --- Kapasitas : 3 Dokter : userK Pasien di dalam ruangan: 1. userD 2. userE Antrean pasien: Tidak ada pasien dalam antrean saat ini.</pre> <hr/> <pre>--- Informasi Ruangan A3 --- Kapasitas : 3 Dokter : userL Pasien di dalam ruangan: 1. userF Antrean pasien: Tidak ada pasien dalam antrean saat ini.</pre> <hr/>	1	2	3	A	A1	A2	A3		B	B1	B2	B3						+-----+
1	2	3																	
A	A1	A2	A3																
B	B1	B2	B3																
				+-----+															

	<pre> --- Informasi Ruangan B1 --- Kapasitas      : 3 Dokter        : userM Pasien di dalam ruangan: 1. userH 2. userG Antrean pasien: Tidak ada pasien dalam antrean saat ini.  ----- # Ruangan kosong (tanpa dokter) tidak ditampilkan # Kasus tidak ada pasien --- Informasi Ruangan B3 --- Kapasitas      : 3 Dokter        : userO Pasien di dalam ruangan: Tidak ada pasien di dalam ruangan saat ini.  -----</pre>
F10 - Tambah Dokter	<pre> # Kasus 1: Dokter sudah terdaftar &gt;&gt;&gt; tambah_dokter Username: userJ User userJ sudah terdaftar. Daftar dengan username lain atau gunakan command LUPA_PASSWORD jika kamu lupa password akunmu. Username:</pre>
	<pre> # Kasus 2: Dokter belum terdaftar &gt;&gt;&gt; tambah_dokter Username: userZ Password: Pass00 Dokter userZ berhasil ditambahkan dengan ID 21!</pre>
	<pre> # Kasus 3: Role bukan Manager &gt;&gt;&gt; tambah_dokter Anda tidak memiliki akses untuk fitur tambah dokter!</pre>
F11 - Diagnosis	<pre> # Kasus 3: Terdapat Pasien yang harus diperiksa &gt;&gt;&gt; diagnosis Pasien userB terdiagnosis menderita penyakit COVID-19.</pre>
	<pre> # Kasus 3: Pasien sudah diperiksa dan belum sembuh &gt;&gt;&gt; diagnosis Pasien userB sudah terdiagnosis sebelumnya!</pre>
	<pre> # Kasus 3: Role bukan Dokter &gt;&gt;&gt; diagnosis Hanya dokter yang dapat melakukan diagnosis!</pre>

F12 - Ngobatin	<p># Kasus 1: Role bukan dokter  <b>&gt;&gt;&gt; ngobatin</b>  Hanya dokter yang dapat memberikan pengobatan!</p> <p># Kasus 2: Tidak ada pasien yang sedang didiagnosis  <b>&gt;&gt;&gt; ngobatin</b>  Tidak ada pasien yang sedang menunggu pengobatan.</p> <p># Kasus 3: Pasien sudah sembuh  <b>&gt;&gt;&gt; ngobatin</b>  Pasien userB sudah sembuh dan tidak memerlukan pengobatan.</p> <p># Kasus 4: Pasien berhasil diobati  <b>&gt;&gt;&gt; ngobatin</b>  Pasien userB berhasil diobati. Semoga lekas sembuh!</p>
F13 - Aku boleh pulang ga, dok 😊?	<p># Kasus 1: Role bukan pasien  <b>&gt;&gt;&gt; pulang</b>  Hanya pasien yang dapat meminta izin untuk pulang!</p> <p># Kasus 2: Pasien belum diperiksa/didiagnosis  <b>&gt;&gt;&gt; pulang</b>  Kamu belum diperiksa! Tidak boleh pulang dulu, ya.</p> <p># Kasus 3: Pasien sudah diperiksa tapi belum sembuh  <b>&gt;&gt;&gt; pulang</b>  Kamu belum sembuh. Harap menjalani pengobatan terlebih dahulu.</p> <p># Kasus 4: Pasien sudah sembuh  <b>&gt;&gt;&gt; pulang</b>  Kamu sudah sembuh! Silakan pulang dan jaga kesehatan, ya!</p>
F14 - Daftar Check-Up	<p># Kasus 1: Role bukan pasien  <b>&gt;&gt;&gt; daftar_checkup</b>  Hanya pasien yang dapat mendaftar check-up!</p> <p># Kasus 2: Pasien sudah terdaftar di antrean mana pun  <b>&gt;&gt;&gt; daftar_checkup</b>  Kamu sudah terdaftar di antrean check-up. Mohon tunggu giliranmu, ya!</p> <p># Kasus 3: Berhasil mendaftar check-up  <b>&gt;&gt;&gt; daftar_checkup</b>  Silakan masukkan data kesehatan anda  # Input salah  Suhu tubuh (celcius): 0  Suhu tubuh harus berupa angka positif!  Suhu tubuh (celcius): 10</p>

Tekanan darah (sistol/diastol, misal 120 80): 120 80  
Detak jantung (bpm): 100  
Saturasi oksigen (%): 100  
Kadar gula darah (mg/dL): 100  
Berat badan (kg): 100  
Tinggi badan (cm): 100  
Kadar kolesterol (mg/dL): 100  
Trombosit (ribu/uL): 1001

# Kasus 3.1: dokter tersedia

Berikut adalah daftar dokter yang tersedia:

1. Dr. userJ - Ruangan A1

Aura : 3  
Pasien di ruangan : 3  
Pasien di antrean : 2  
Biaya checkup : 5

2. Dr. userK - Ruangan A2

Aura : 1  
Pasien di ruangan : 2  
Pasien di antrean : -  
Biaya checkup : 2

3. Dr. userL - Ruangan A3

Aura : 2  
Pasien di ruangan : 1  
Pasien di antrean : -  
Biaya checkup : 4

4. Dr. userM - Ruangan B1

Aura : 2  
Pasien di ruangan : 2  
Pasien di antrean : -  
Biaya checkup : 4

5. Dr. userO - Ruangan B3

Aura : 3  
Pasien di ruangan : -  
Pasien di antrean : -  
Biaya checkup : 5

Pilih dokter yang tersedia (1-5, 0 untuk batal): 4

Pendaftaran berhasil!

Anda terdaftar antrean checkup dengan dr. userM di ruangan B1.

Anda bisa langsung masuk ke ruangan

# Kasus 3.2: dokter tidak tersedia

Maaf, tidak ada dokter yang tersedia saat ini.

Coba sesaat lagi.

	<pre># Kasus 1: Pasien masih dalam antrean (belum masuk ruangan) <b>&gt;&gt;&gt; antrean_saya</b> Status antrian Anda: Dokter: Dr. userJ Ruang: A2 Posisi antrian: 4 dari 6</pre>
F15 - Antrean Saya!	<pre># Kasus 2: Pasien sudah masuk ke ruangan <b>&gt;&gt;&gt; antrean_saya</b> Anda sedang berada di dalam ruangan dokter!</pre>
	<pre># Kasus 3: Pasien belum mendaftar antrean check-up <b>&gt;&gt;&gt; antrean_saya</b> Anda belum terdaftar dalam antrian check-up! Silakan daftar terlebih dahulu dengan command DAFTAR_CHECKUP.</pre>
	<pre># Kasus 1: obat masih ada di inventory <b>&gt;&gt;&gt; minum_obat</b> ===== DAFTAR OBAT ===== 1. Ibuprofen 2. Jus Daun Ganja 3. Senyuman Sang Tercinta 4. Lumpur Lapindo 5. Air Mata Phoenix <b>&gt;&gt;&gt;Pilih obat untuk diminum : 1</b> GULUKGULUKGULUK... Ibuprofen berhasil diminum!!!</pre>
F16 - Minum Obat	<pre># Kasus 2: obat sudah habis/belum diberikan oleh dokter <b>&gt;&gt;&gt; minum_obat</b> obat abis bang</pre>
	<pre># Kasus 1: Perut kosong <b>&gt;&gt;&gt; minum_penawar</b> Perut kosong!! Belum ada obat yang dimakan.</pre>
	<pre># Kasus 2: Berhasil memuntahkan obat dan masuk kembali ke inventory <b>&gt;&gt;&gt; minum_penawar</b> Uwekkk!!! Antasida keluar dan kembali ke inventory</pre>
F17 - Minum Penawar	<pre># Kasus 3: Inventory penuh <b>&gt;&gt;&gt; minum_penawar</b> Inventory penuh! Amoxicillin tidak bisa dimasukkan kembali.</pre>
	<pre># Kasus 1: Masih ada user aktif <b>&gt;&gt;&gt; exit</b> Rumah sakit hanya dapat dihancurkan jika tidak ada pengguna yang mengakses! Silakan LOGOUT untuk EXIT</pre>

F18 - Exit

	<pre># Kasus 2: Berhasil keluar dan tidak menyimpan &gt;&gt;&gt; exit Mau save perubahan? (Y/N) N Terima kasih telah menggunakan aplikasi Rumah Sakit Nimons! Sampai jumpa lagi!</pre>
	<pre># Kasus 3: Berhasil keluar dan menyimpan &gt;&gt;&gt; exit Mau save perubahan? (Y/N) Y ✓ Data berhasil disimpan! Terima kasih telah menggunakan aplikasi Rumah Sakit Nimons! Sampai jumpa lagi!</pre>
B02 - Denah Dinamis	<pre># Kasus 1: Menambah baris &gt;&gt;&gt; tambah_baris Anda tidak terdaftar dalam antrean!</pre>
	<pre># Kasus 2: Menambah kolom &gt;&gt;&gt; tambah_kolom Anda tidak terdaftar dalam antrean!</pre>
	<pre># Kasus 3: Menghapus baris &gt;&gt;&gt; kurang_baris Anda tidak terdaftar dalam antrean!</pre>
	<pre># Kasus 4: Menghapus kolom &gt;&gt;&gt; kurang_kolom Anda tidak terdaftar dalam antrean!</pre>
B04 - Banarich	<pre># Kasus 1: Melakukan gacha &gt;&gt;&gt; gacha_gaming Kchak.. Kchak.. Thunk! Kamu mendapatkan 25 banarich!</pre>
	<pre># Kasus 2: Melihat isi dompet &gt;&gt;&gt; lihat_dompet Dompetmu saat ini berisi 70 banarich.</pre>
	<pre># Kasus 3: Melihat kondisi keuangan rumah sakit &gt;&gt;&gt; lihat_keuangan Rumah sakit saat ini memiliki 10 banarich. Tetaplah bekerja keras untuk memperkaya pemilik rumah sakit!</pre>
B05 - Dead or Alive??	<pre>&gt;&gt;&gt; minum_penawar Dokter sedang memberikan obat penawar dan kamu sedang minum. Nyawa kamu sudah 0, kamu meninggal.</pre>
B06 - Mainan Antrian	<pre># Kasus 1: Pasien tidak ditemukan di antrean manapun &gt;&gt;&gt; skip_antrian Anda tidak terdaftar dalam antrean!</pre>

	<pre># Kasus 2: Pasien sudah di dalam ruangan &gt;&gt;&gt; skip_antrian Anda sudah berada di dalam ruangan, tidak bisa skip antrean!</pre>
	<pre># Kasus 3: Pasien sudah di antrean terdepan &gt;&gt;&gt; skip_antrian Anda sudah berada di antrean terdepan, tidak perlu skip!</pre>
	<pre># Kasus 4: Berhasil skip antrean &gt;&gt;&gt; skip_antrian ✓ Anda berhasil dipindahkan ke antrean terdepan! Posisi baru Anda: Antrean ke-1 (setelan yang sedang dalam ruangan)</pre>
	<pre># Kasus 1: Belum masuk ruangan &gt;&gt;&gt; cancel_antrean ✓ Anda berhasil membatalkan antrean dokter userJ ruangan A1</pre>
	<pre># Kasus 2: Sudah masuk ruangan &gt;&gt;&gt; cancel_antrean Anda sudah berada di dalam ruangan, tidak bisa membatalkan antrean!</pre>

## 2.5 Desain Kamus Data

Tabel 2.5 Desain Kamus Data

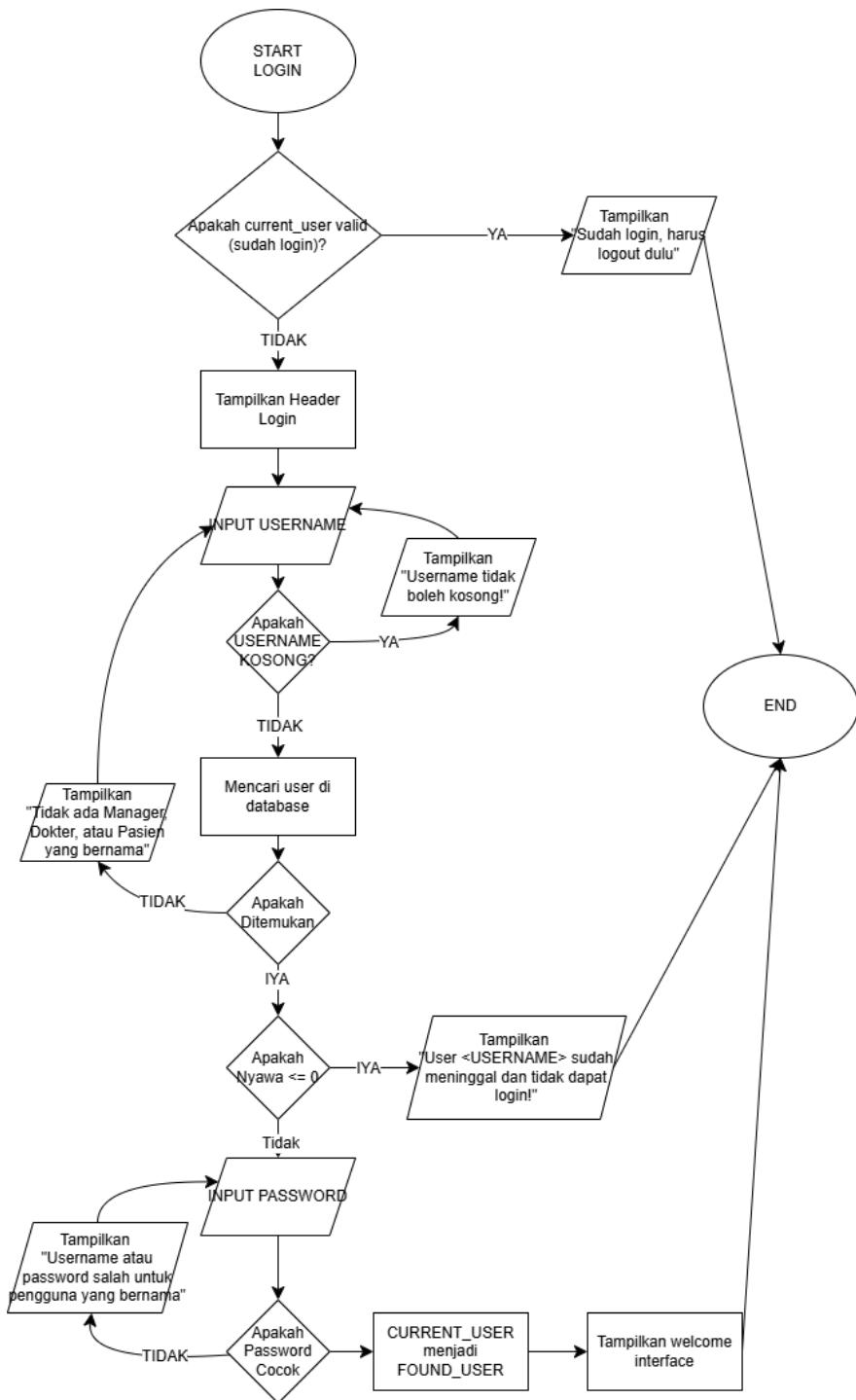
Fitur	Kamus Data
F01 - Login	username : <u>string</u> password : <u>string</u> found_user : User i : <u>integer</u> validUsername : <u>boolean</u>
F02 - Register	username, password : <u>string</u> temp_username : <u>string</u> valid : <u>boolean</u> i, m, l, r : <u>integer</u> cmp : <u>integer</u> set_name_lower : <u>string</u> pos : <u>integer</u> new_user : User
F03 - Logout	-
F04 - Lupa Password	{validasiKodeUnik} i, k : <u>integer</u> len : <u>integer</u> temp : <u>string</u>

	<pre> count          : <u>integer</u> hurufUsername : <u>char</u> hurufCount    : <u>string</u> hasil         : <u>integer</u>  { lupaPassword } username, kodeUnik, passwordBaru      : <u>string</u> found_index, i                         : <u>integer</u> user_by_id                            : User </pre>
F05 - Menu dan Help	<pre> role_lower : <u>string</u> i           : <u>integer</u> </pre>
F06 - Denah Rumah Sakit	<pre> { lihatDenah} i, j : <u>integer</u>  { lihatRuangan } ruang_code : <u>string</u> row, col   : <u>integer</u> </pre>
F07 - Lihat User	<pre> { lihatUser } kriteria_urutan, arah_urutan: <u>integer</u> jumlah_pengguna            : <u>integer</u> indeks,       indeks_luar,     indeks_dalam: <u>integer</u> perlu_tukar                 : <u>boolean</u> hasil_perbandingan          : <u>integer</u> daftar_pengguna_terurut    : <u>array of</u> User pengguna_sekarang           : User nama_penyakit               : <u>string</u>  { lihatDokter } kriteria_urutan, arah_urutan: <u>integer</u> jumlah_dokter                : <u>integer</u> i, j                          : <u>integer</u> perlu_tukar                  : <u>boolean</u> hasil_perbandingan          : <u>integer</u> daftar_dokter                : <u>array of</u> User temp, dokter_sekarang        : User  { lihatPasien } kriteria_urutan, arah_urutan: <u>integer</u> jumlah_pasien, i, j          : <u>integer</u> perlu_tukar                  : <u>boolean</u> hasil_perbandingan          : <u>integer</u> daftar_pasien                : <u>array of</u> User temp, pasien_sekarang        : User </pre>

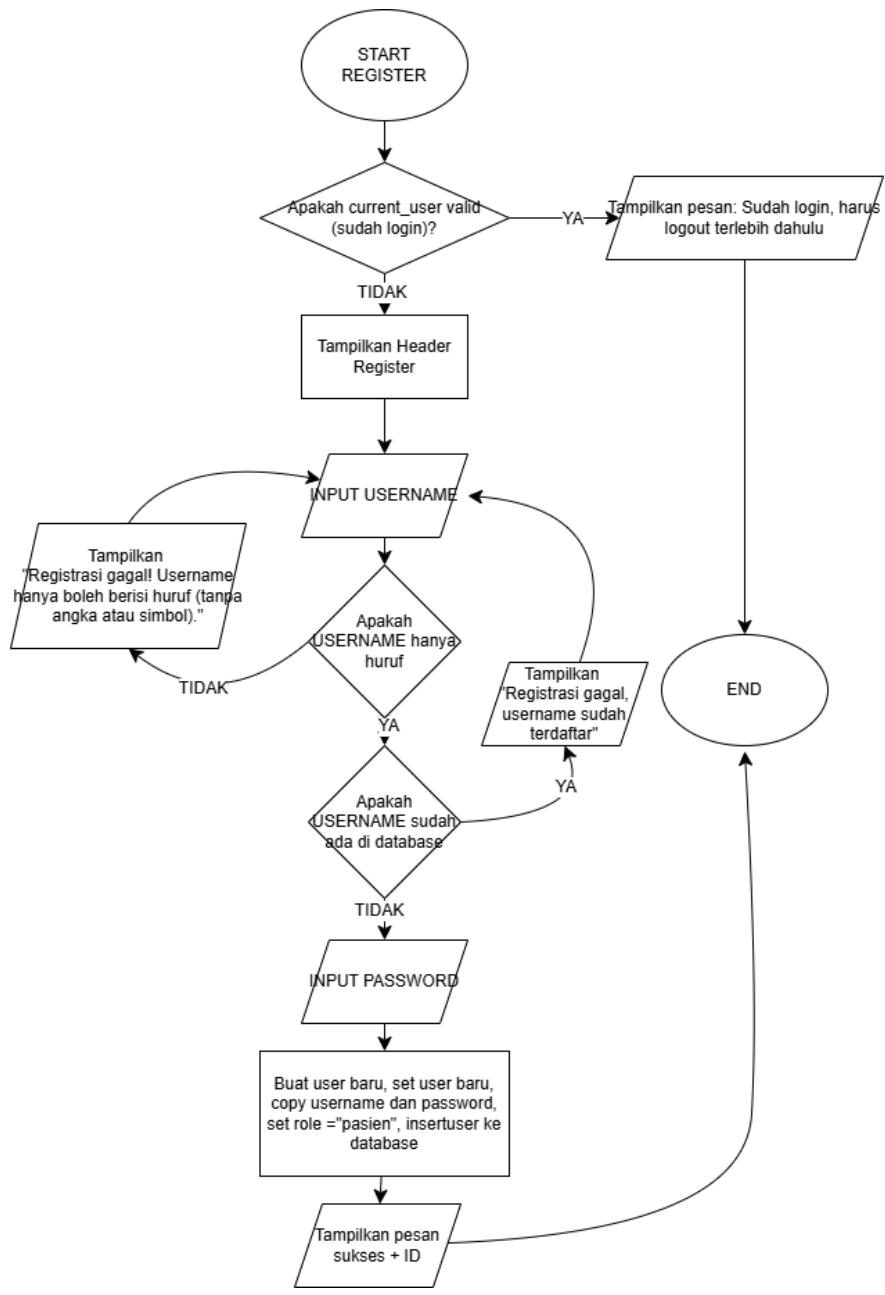
	<pre> nama_penyakit      : <u>string</u> </pre>
F08 - Cari User	<pre> { cariUser } sortBy, sortOrder    : <u>integer</u> dokterList          : <u>array of</u> User n, i, j              : <u>integer</u> shouldSwap          : <u>boolean</u> temp                 : User  { cariPasien } pilihan, pilihan2, sort   : <u>integer</u> id, jumlahpasien        : <u>integer</u> nama, nama_penyakit     : <u>string</u> pasiendicari           : ListUser temp                   : User i, j, comp              : <u>integer</u>  { cariDokter } pilihan      : <u>integer</u> id           : <u>integer</u> nama         : <u>string</u> p             : User </pre>
F09 - Lihat Antrean	<i>i, j : <u>integer</u></i>
F10 - Tambah Dokter	<pre> nama      : <u>string</u> password  : <u>string</u> i         : <u>integer</u> new_user  : User valid     : <u>boolean</u> </pre>
F11 - Diagnosis	<pre> pasien_diagnosis    : Patient penyakit_terpilih   : <u>string</u> i                   : <u>integer</u> diagnosis_berhasil : <u>boolean</u> </pre>
F12 - Ngobatin	<pre> i, j, k, p, m       : <u>integer</u> idxI, idxJ          : <u>integer</u> pasien               : User obat                : ListObat found               : <u>boolean</u> hadMedications      : <u>boolean</u> userDuluTidakAdaObat : <u>boolean</u> jumlah_obat         : <u>integer</u> userByID, userByName : pointer to user </pre>

F13 - Aku boleh pulang ga, dok 😊?	resep : ListObat found : boolean n, i, idxSalah : integer urutanBenar : boolean
F14 - Daftar Check-Up	pointer : *User suhu_tubuh : real tekanan_darah_sistolik : integer tekanan_darah_diastolik : integer detak_jantung : integer saturasi_oksigen : real kadar_gula_darah : integer tinggi_badan : integer kadar_kolesterol : integer trombosit : integer berat_badan : real drccount, pilihan : integer count, row, col : integer current : Ruangan foundPointer : boolean
F15 - Antrean Saya!	i, j, pos, total : integer found : boolean ruangan : Ruangan q : Address
F16 - Minum Obat	pilihan : integer obat_pilihan : Obat
F17 - Minum Penawar	resep : ListObat ditemukan : boolean urutanBenar : boolean keluar : boolean n, i, j : integer terakhir : Obat username_temp : string
F18 - Exit	c, opt : char
main	current_user : User folderName : string command : string c : integer selesai : boolean

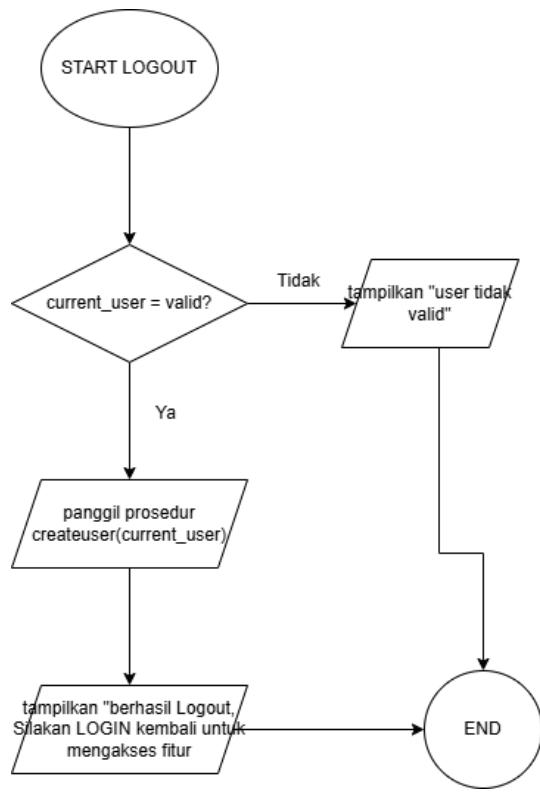
## 2.6 Desain Dekomposisi Algoritmik dan Fungsional



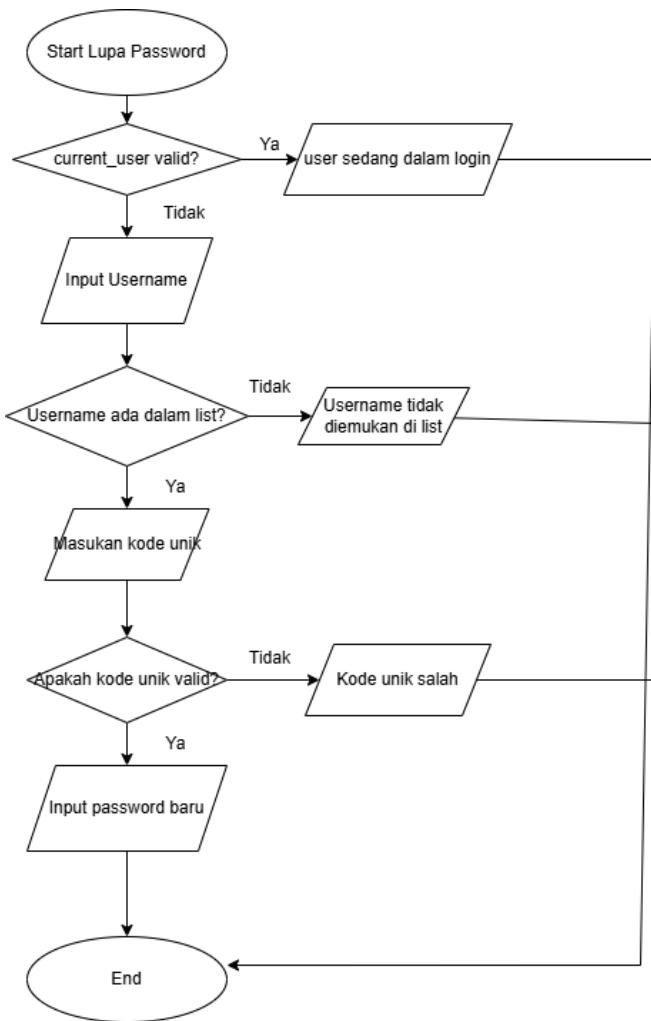
Gambar 2.6.1 Flowchart Login



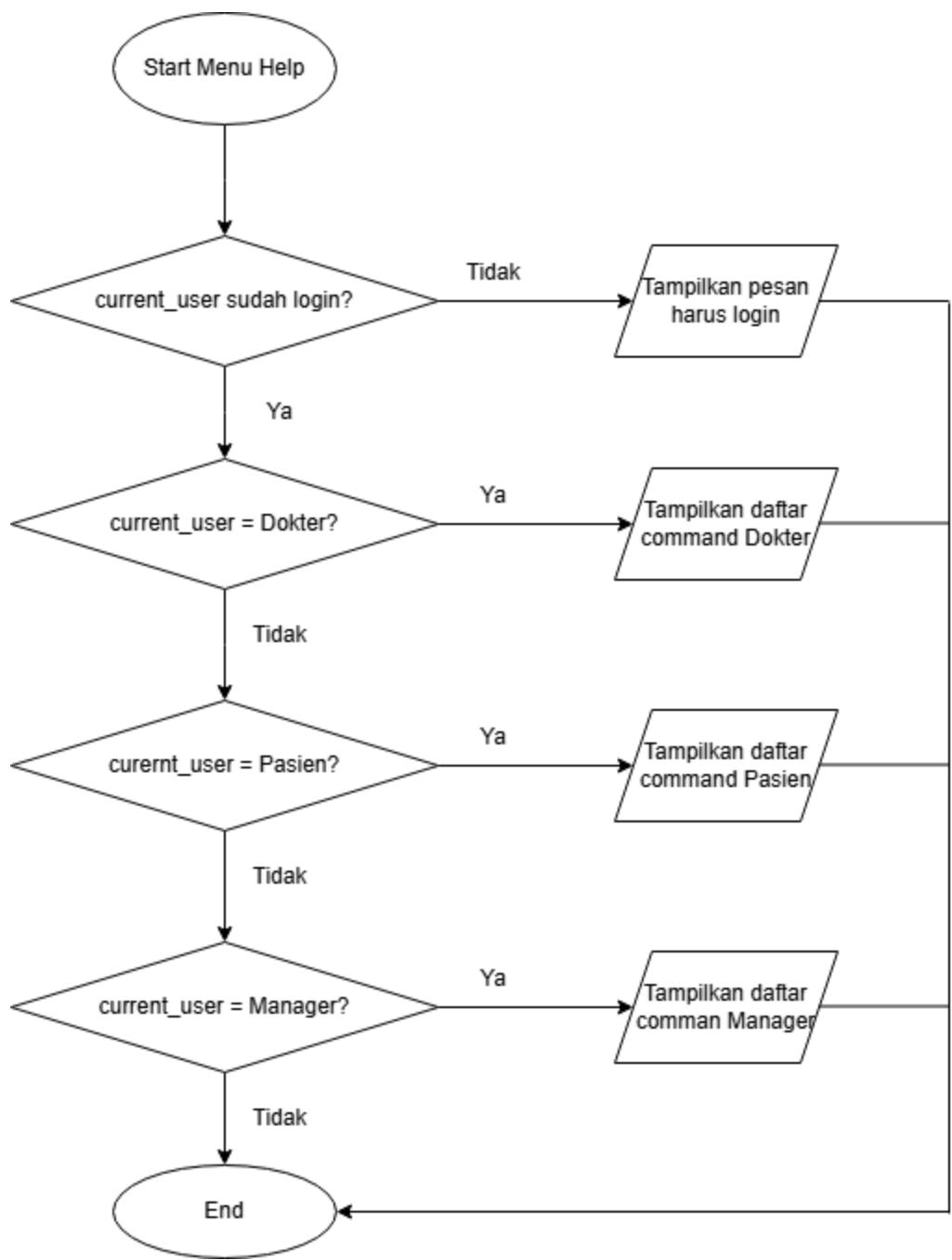
Gambar 2.6.2 Flowchart Register



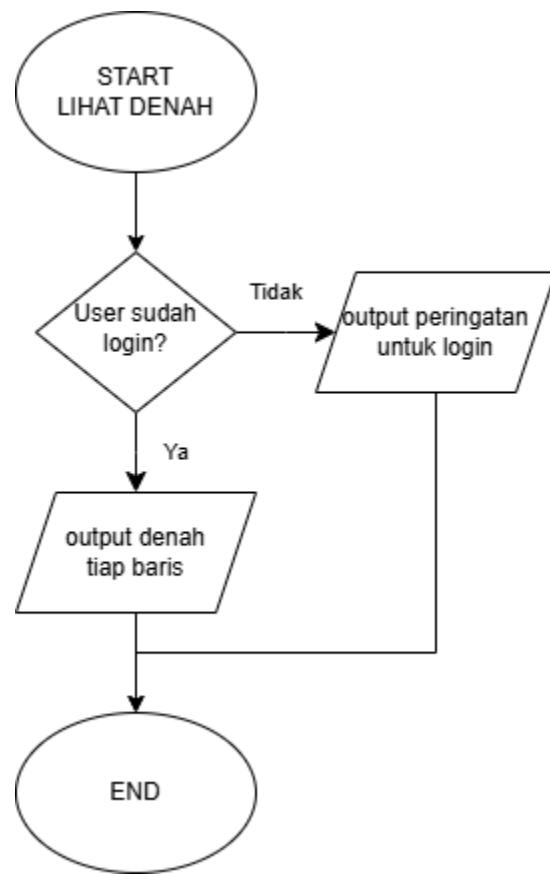
Gambar 2.6.3 Flowchart Logout



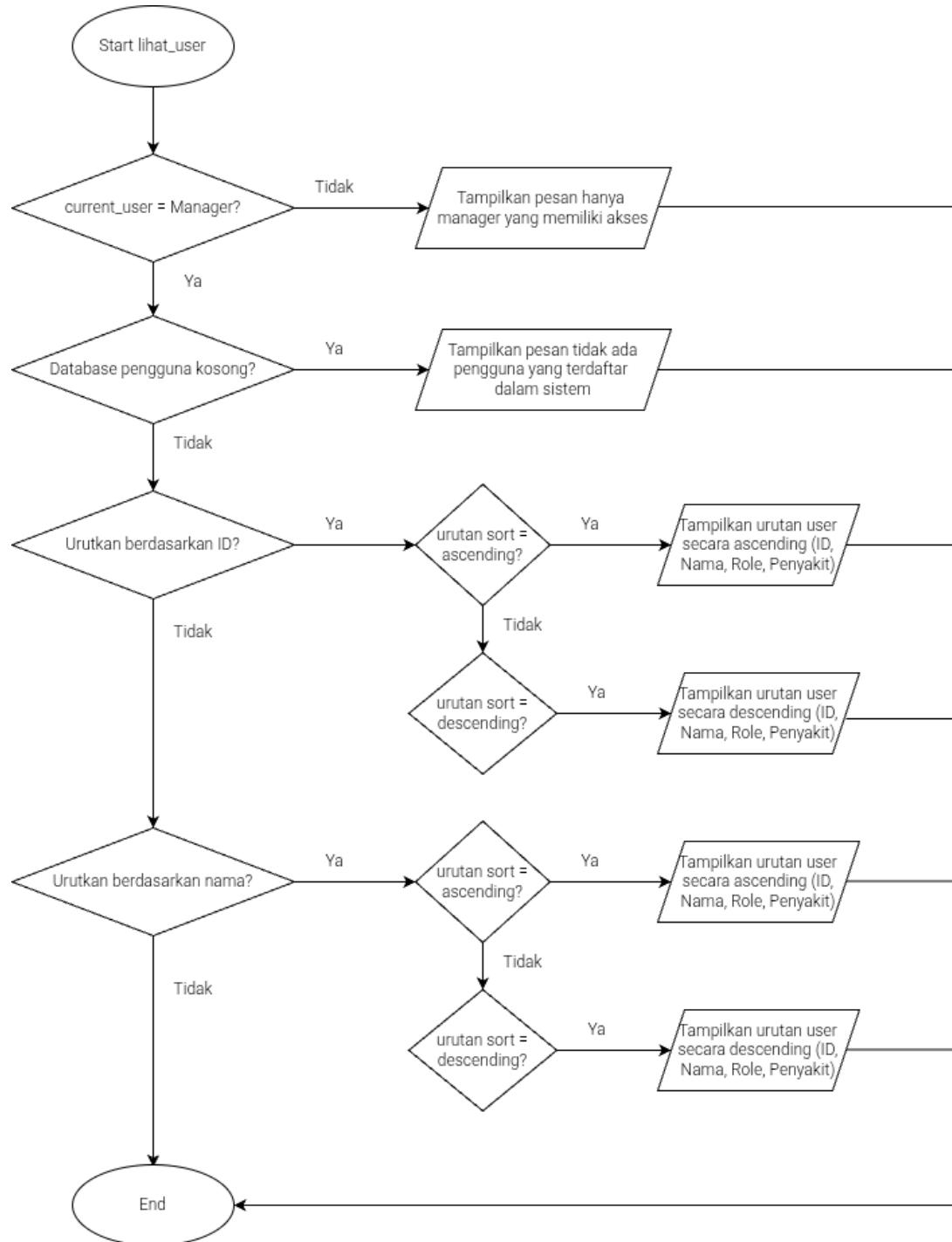
Gambar 2.6.4 Flowchart Lupa Password

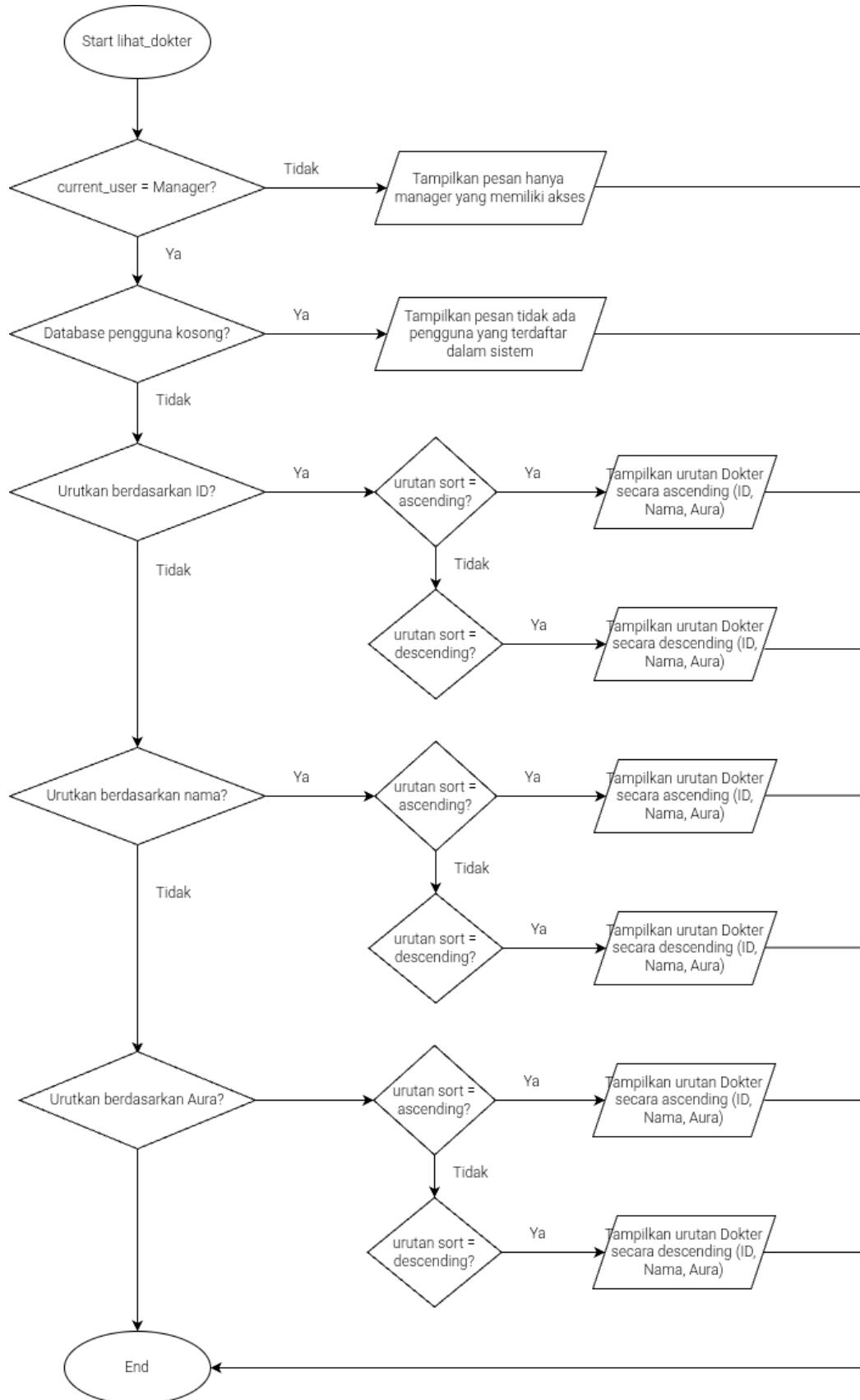


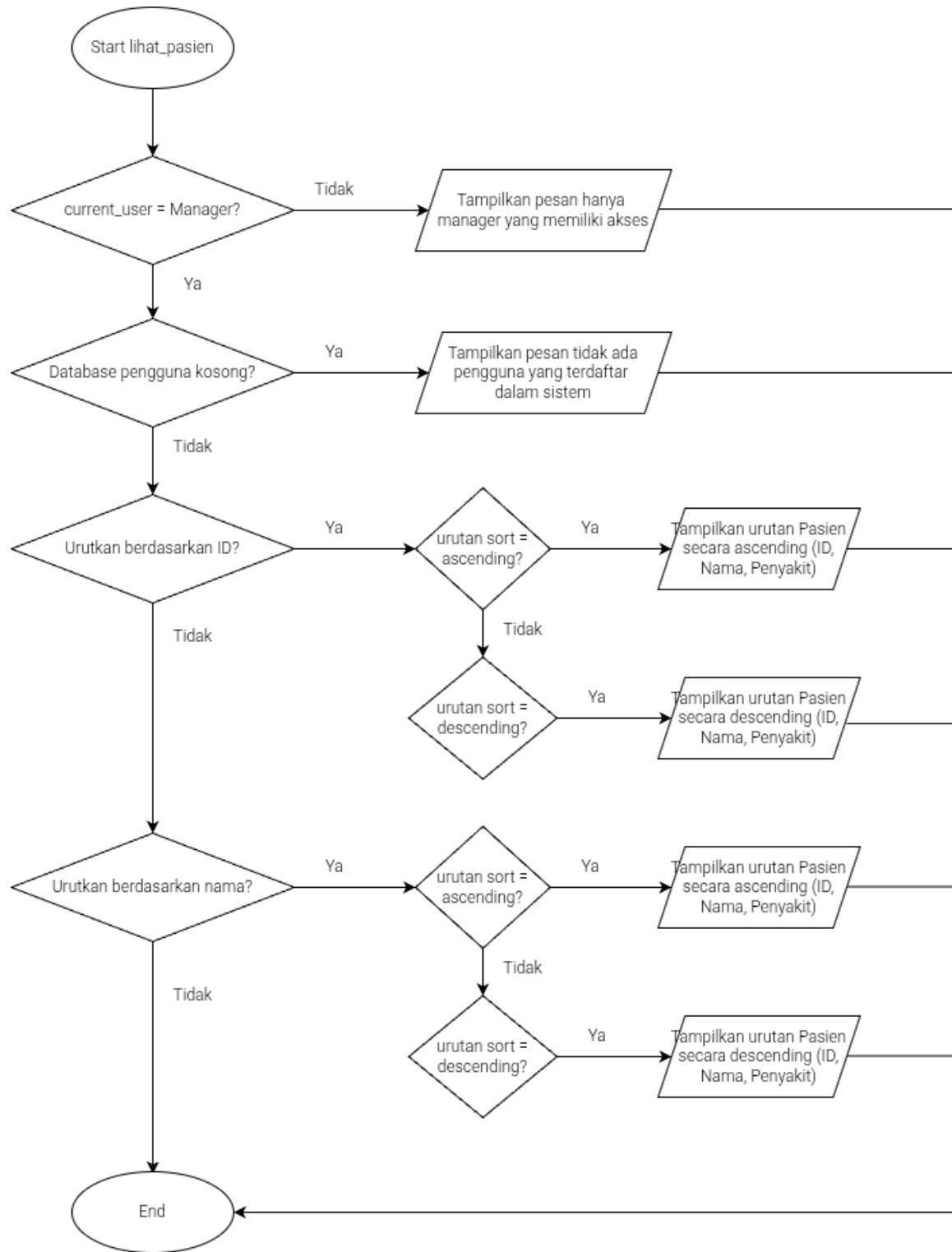
Gambar 2.6.5 Flowchart Menu & Help



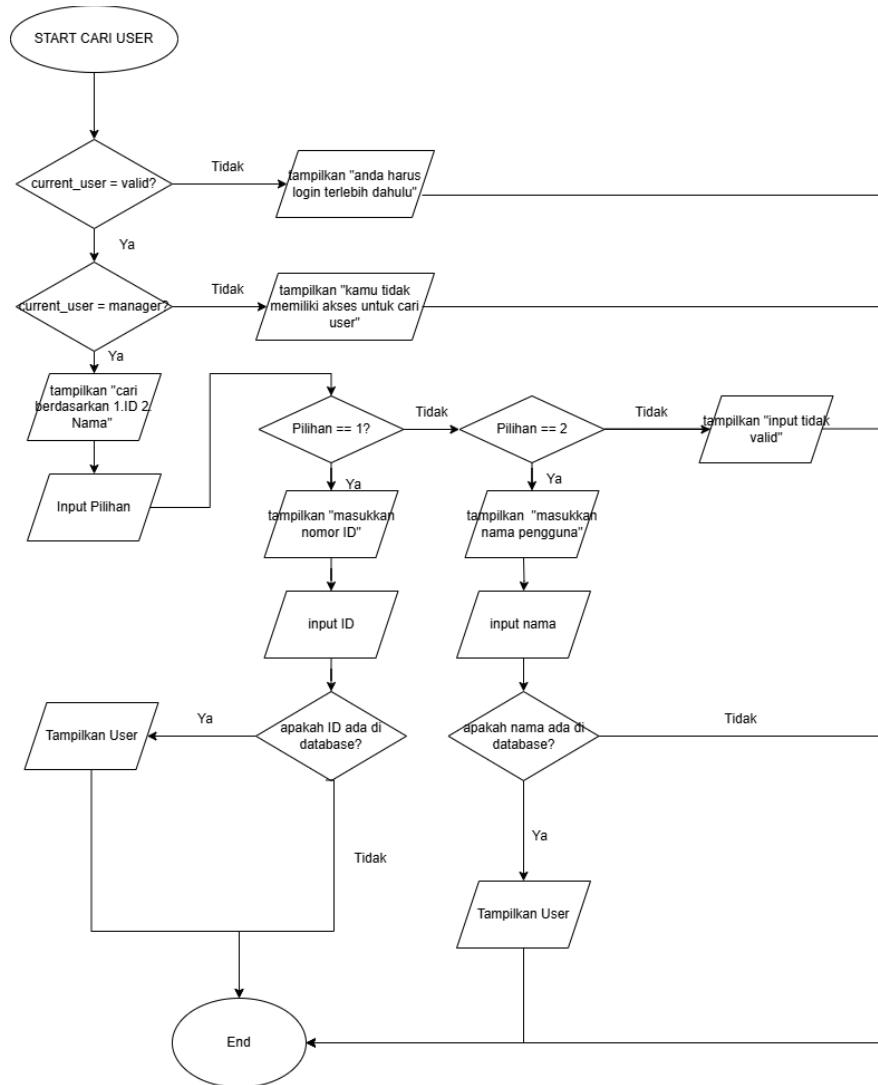
Gambar 2.6.6 Flowchart Denah Rumah Sakit

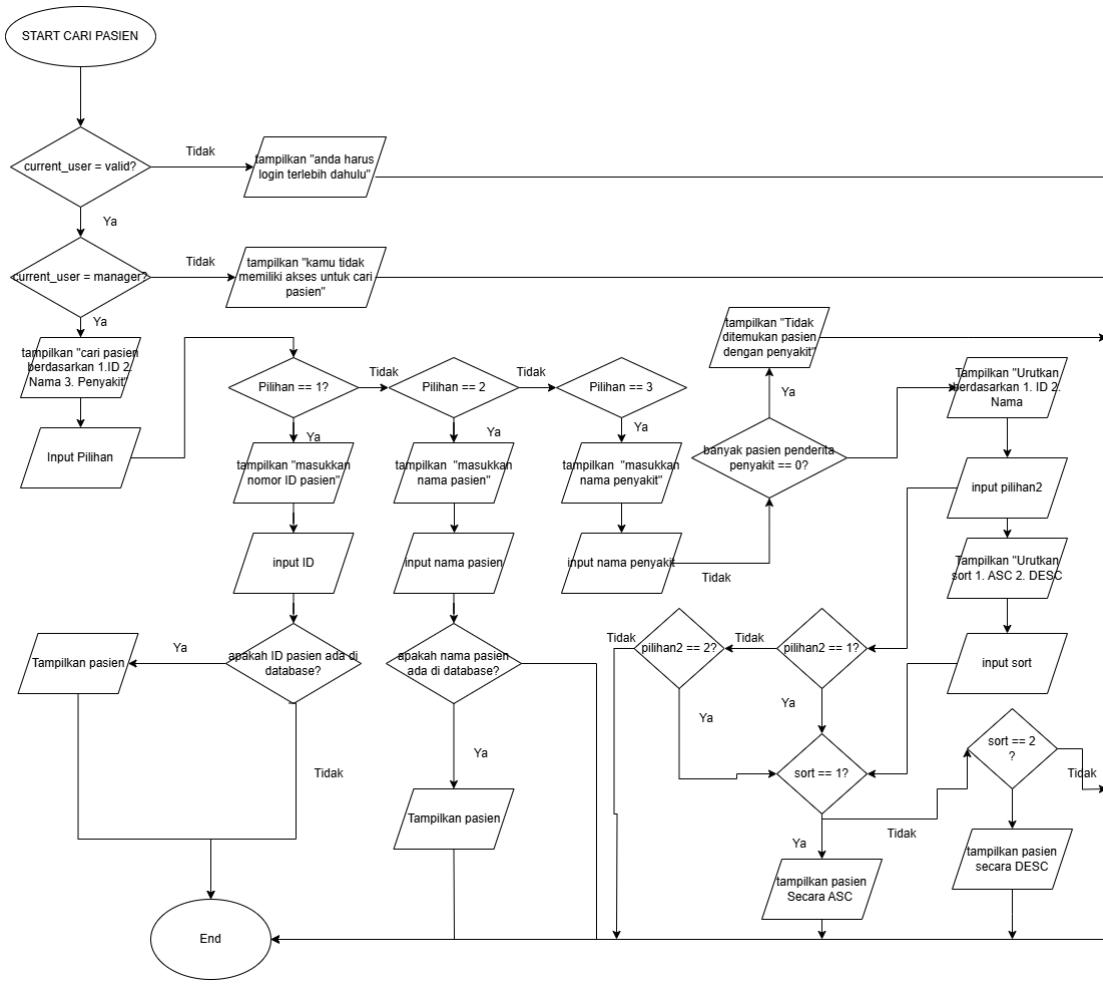






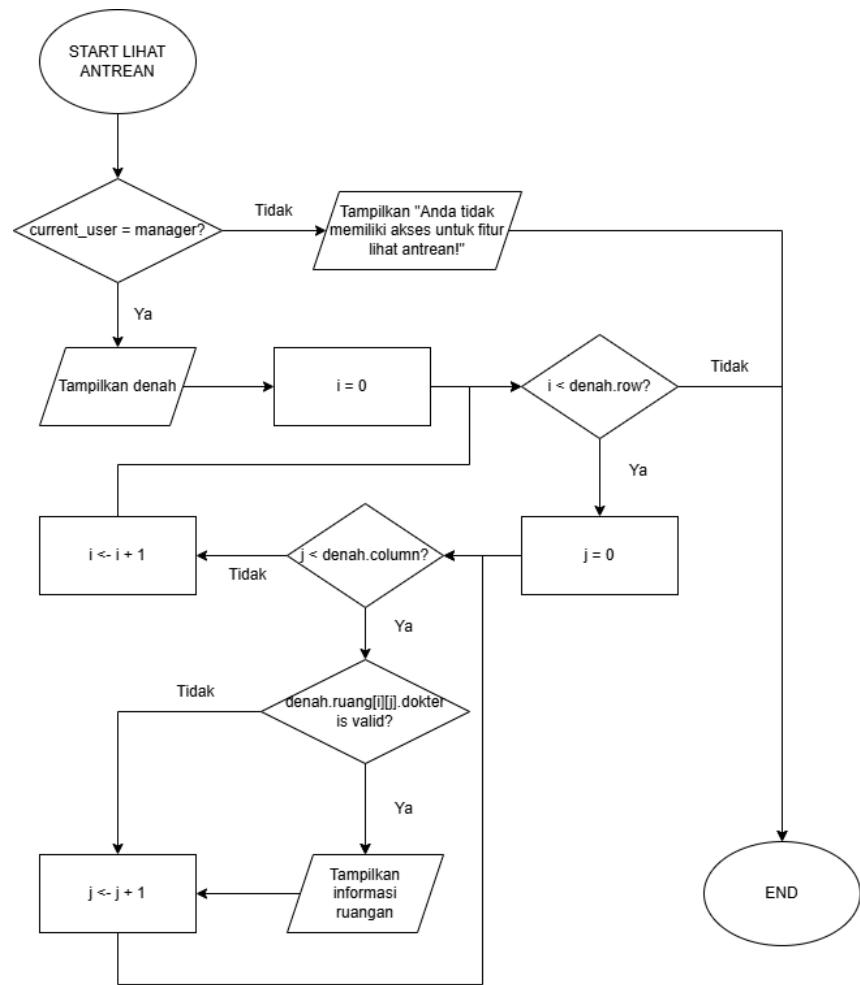
Gambar 2.6.7 Flowchart Lihat User



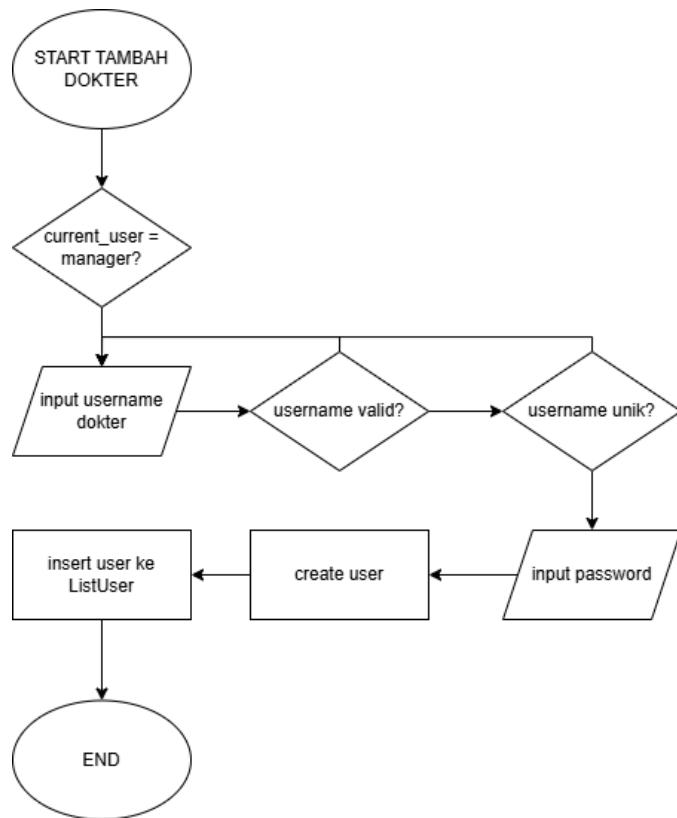




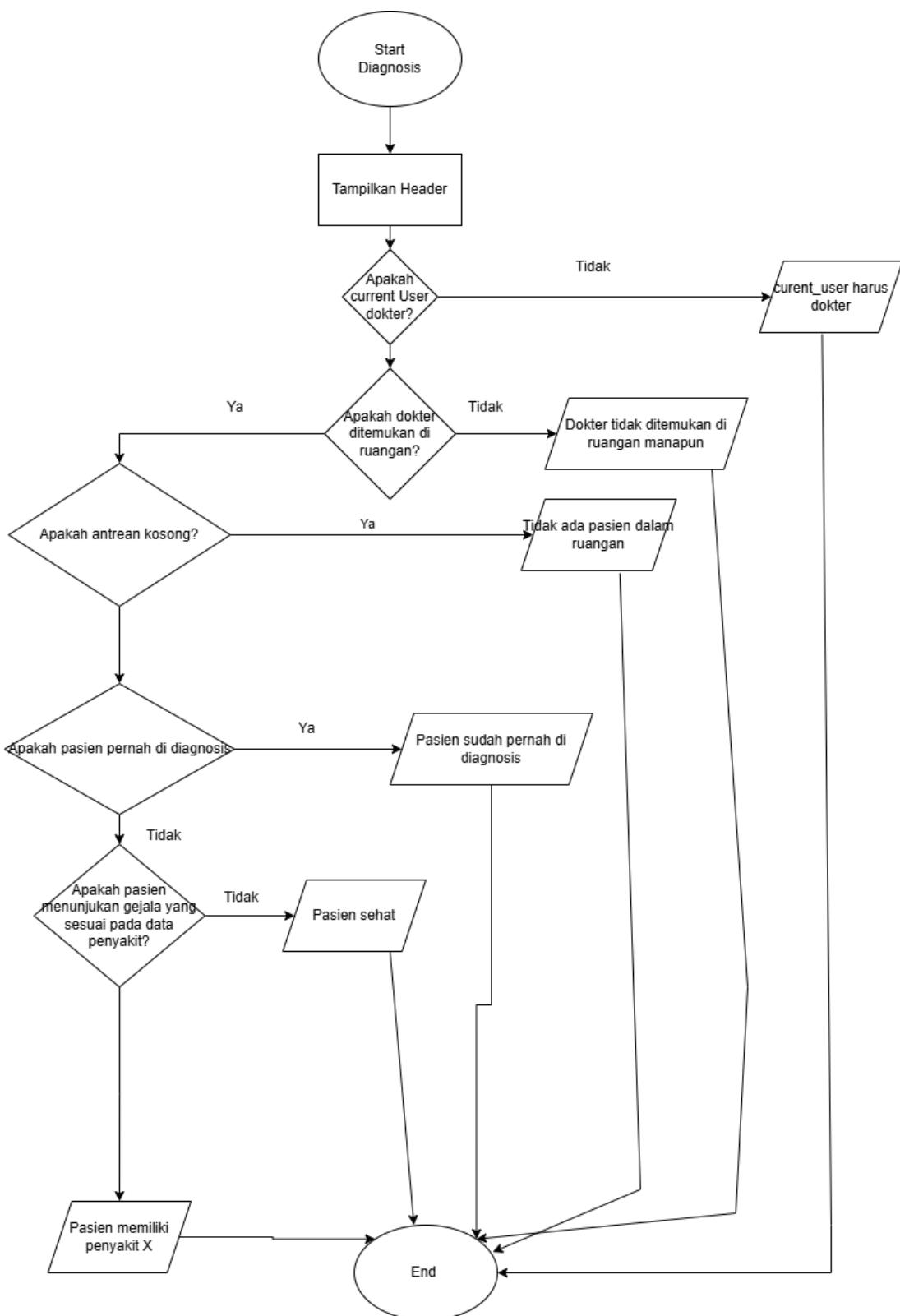
Gambar 2.6.8 Flowchart Cari User



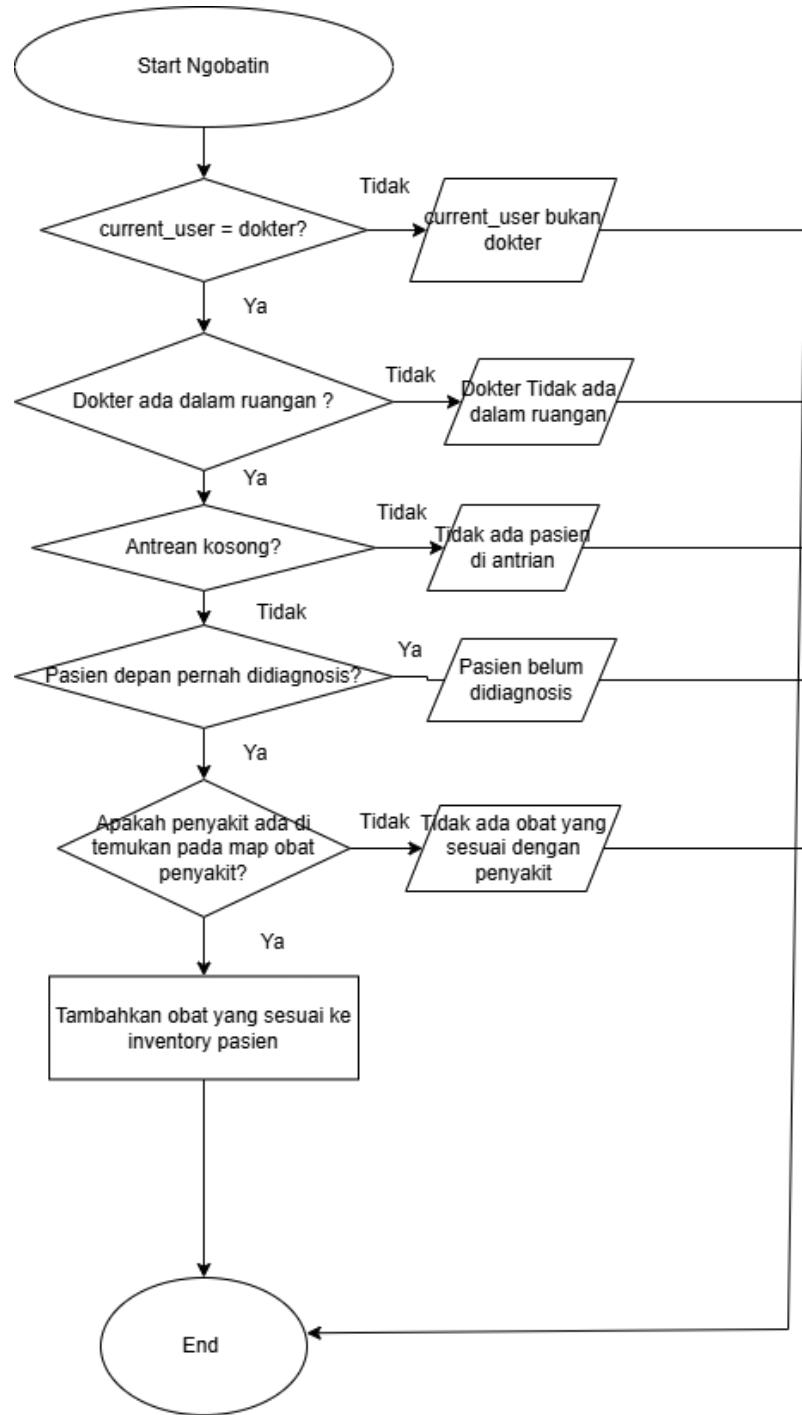
Gambar 2.6.9 Flowchart Lihat Antrian



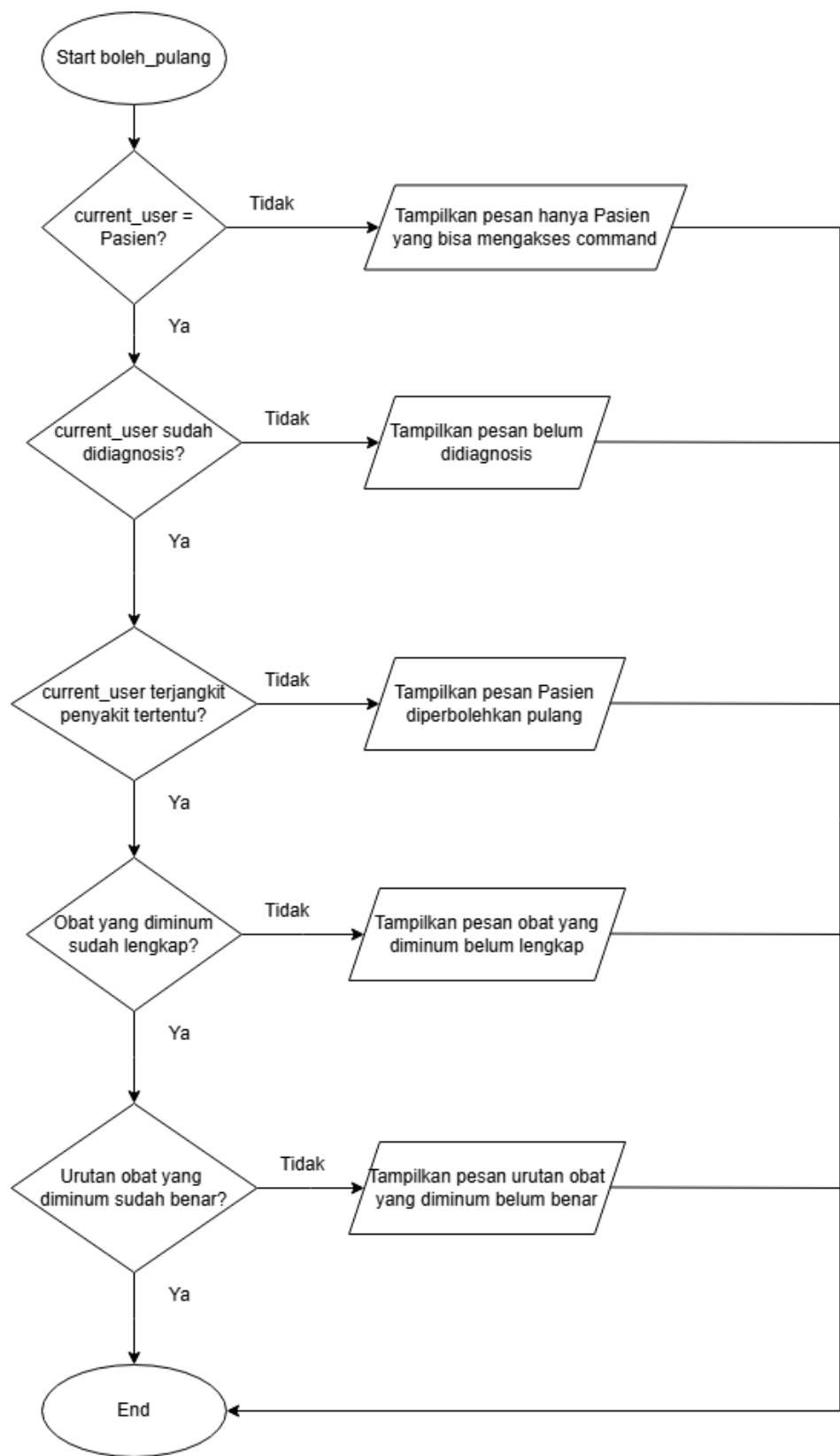
Gambar 2.6.10 Flowchart Tambah Dokter



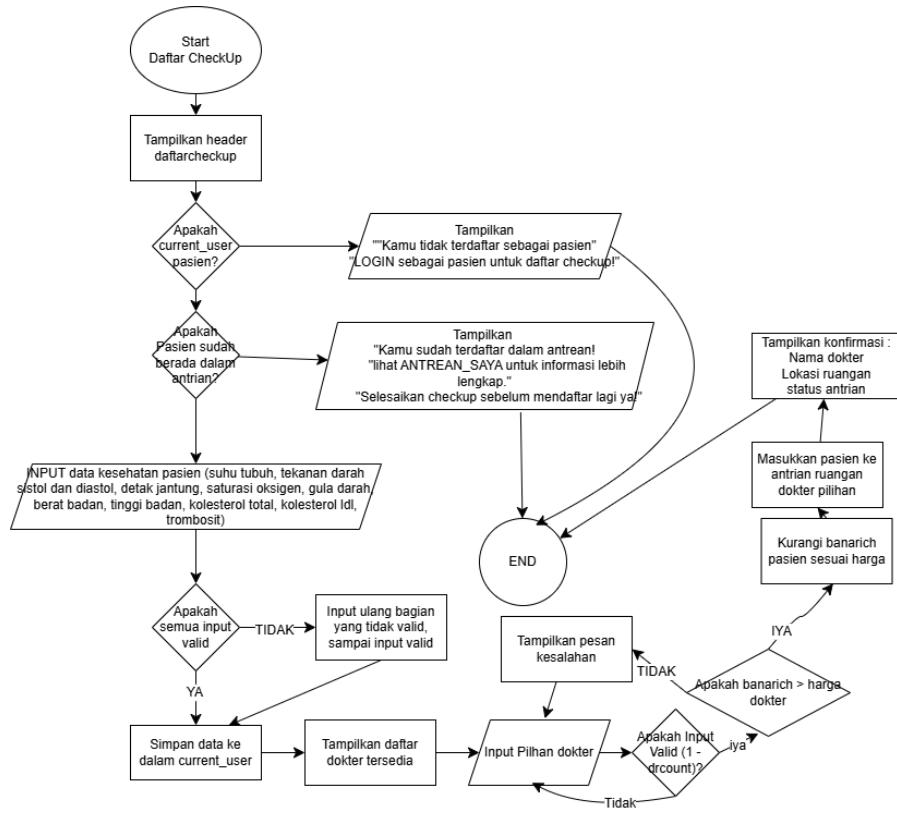
*Gambar 2.6.11 Flowchart Diagnosis*



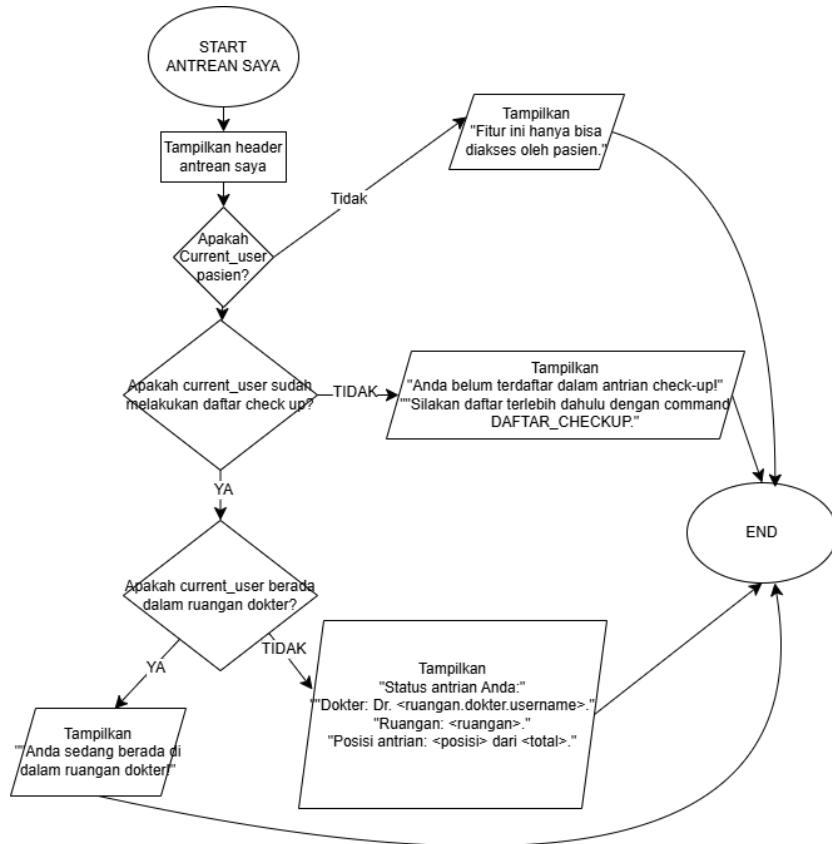
Gambar 2.6.12 Flowchart Ngobatin



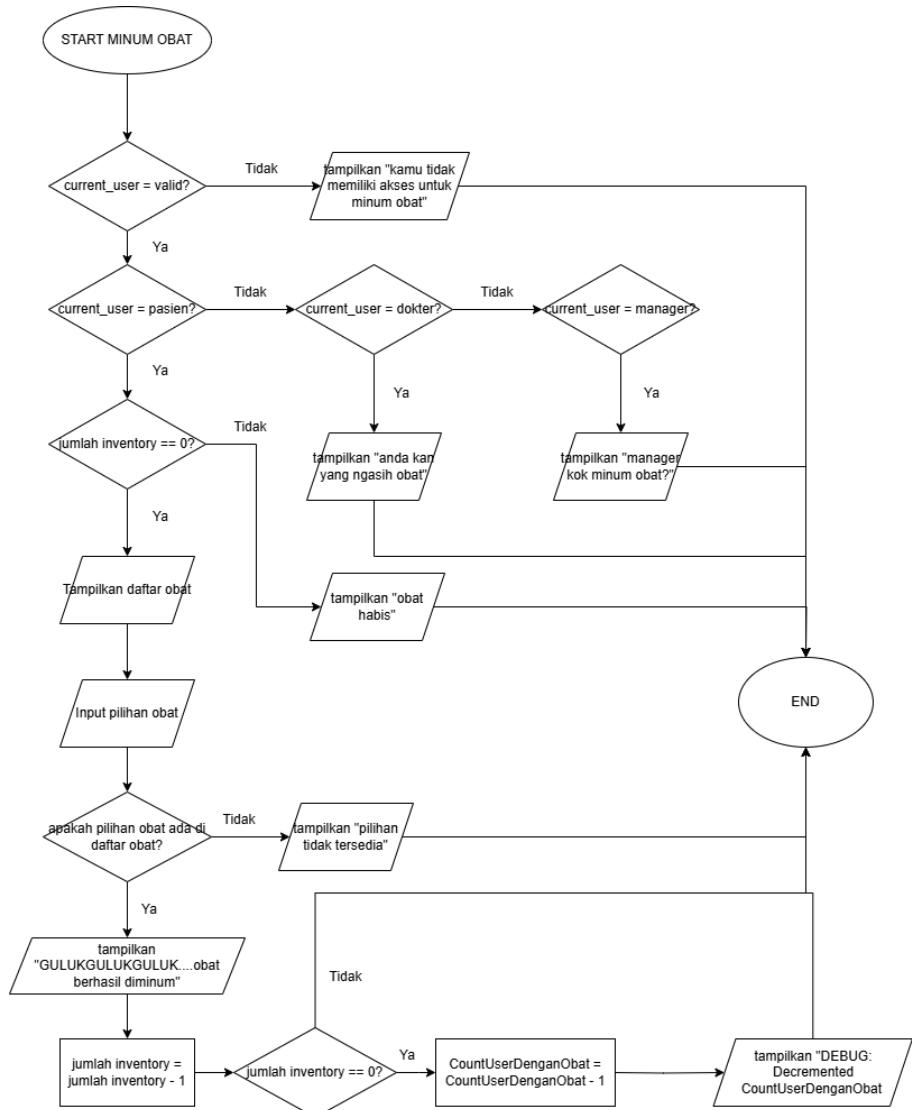
Gambar 2.6.13 Flowchart Aku boleh pulang ga, dok 😊 ?



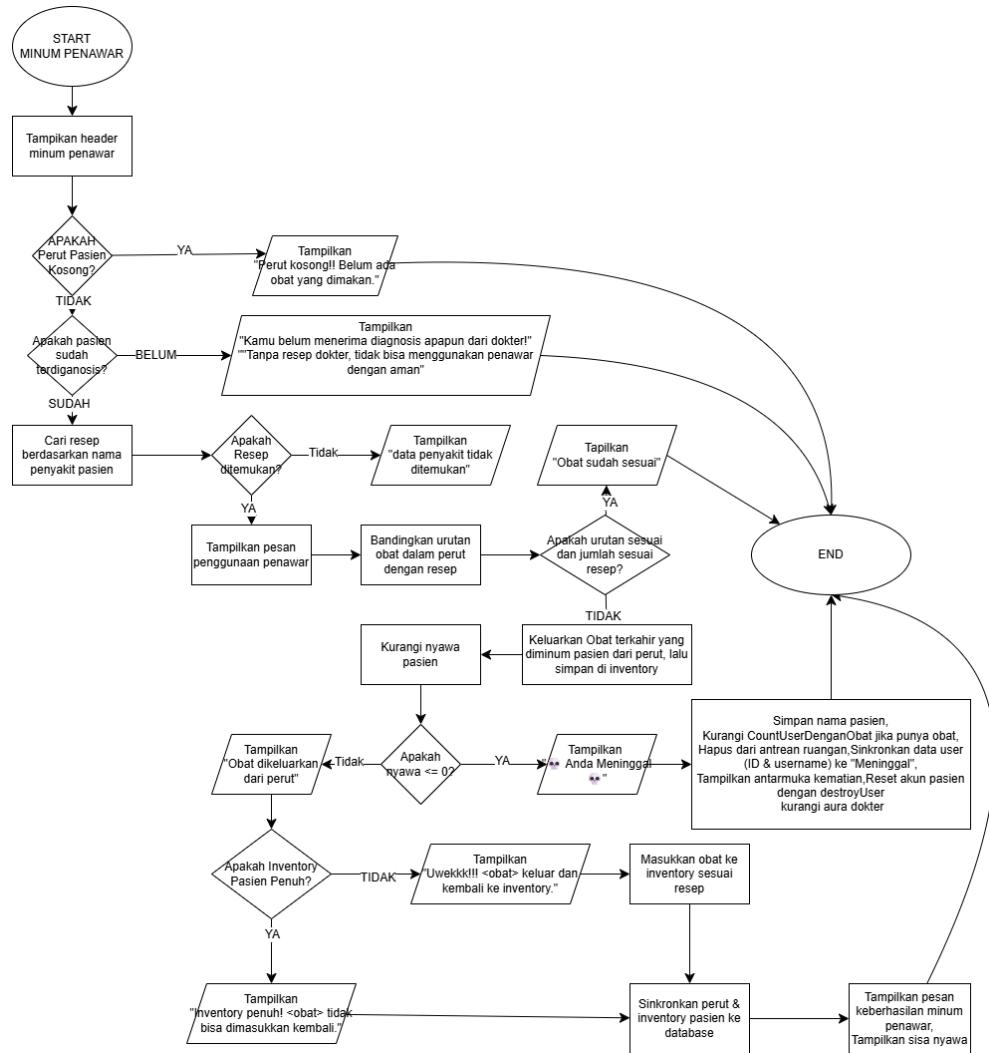
Gambar 2.6.14 Flowchart Daftar Check-Up



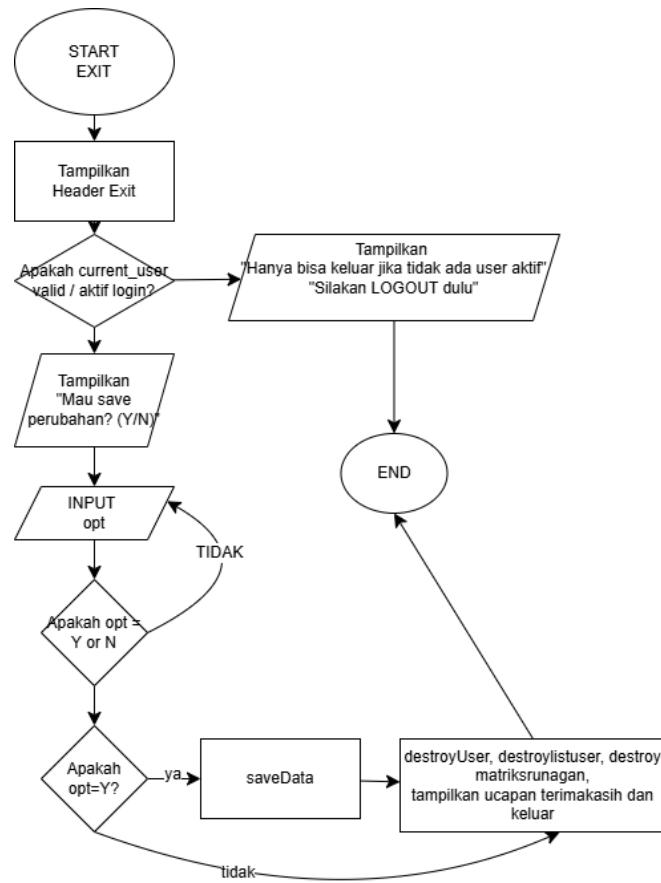
Gambar 2.6.15  
Flowchart Antrian Saya!



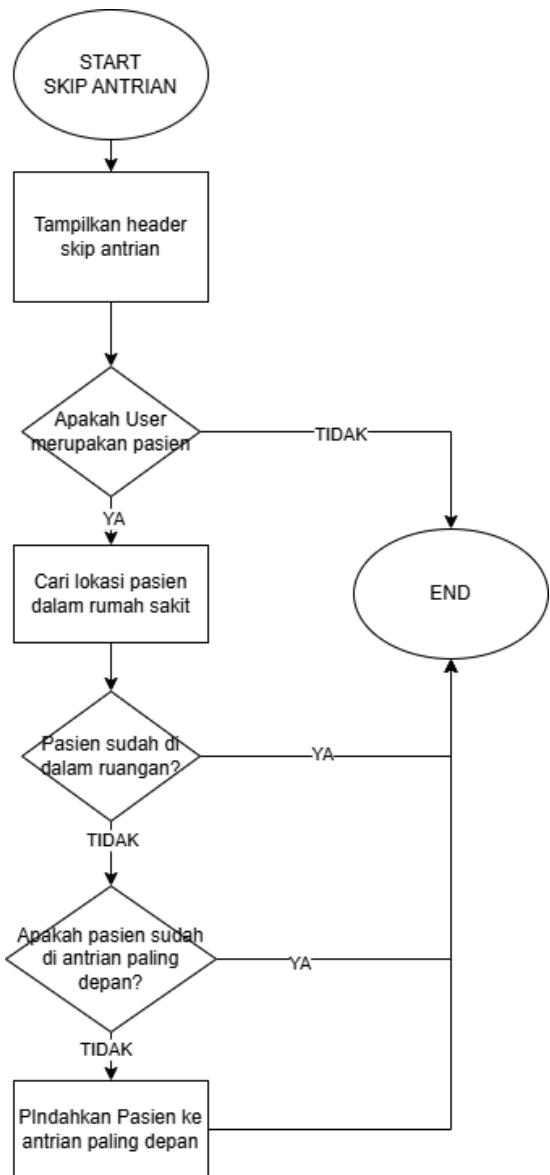
Gambar 2.6.16 Flowchart Minum Obat



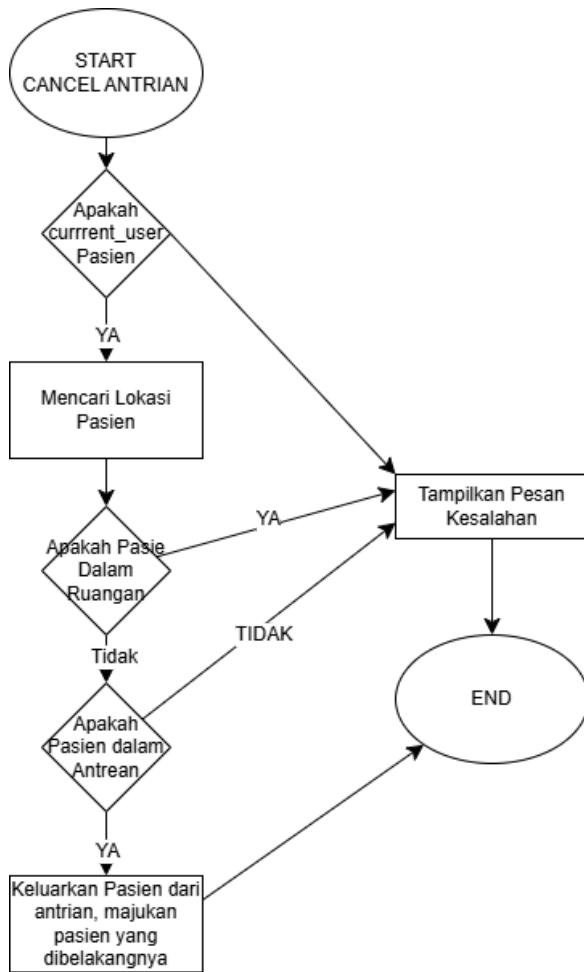
Gambar 2.6.17 Flowchart Minum Penawar



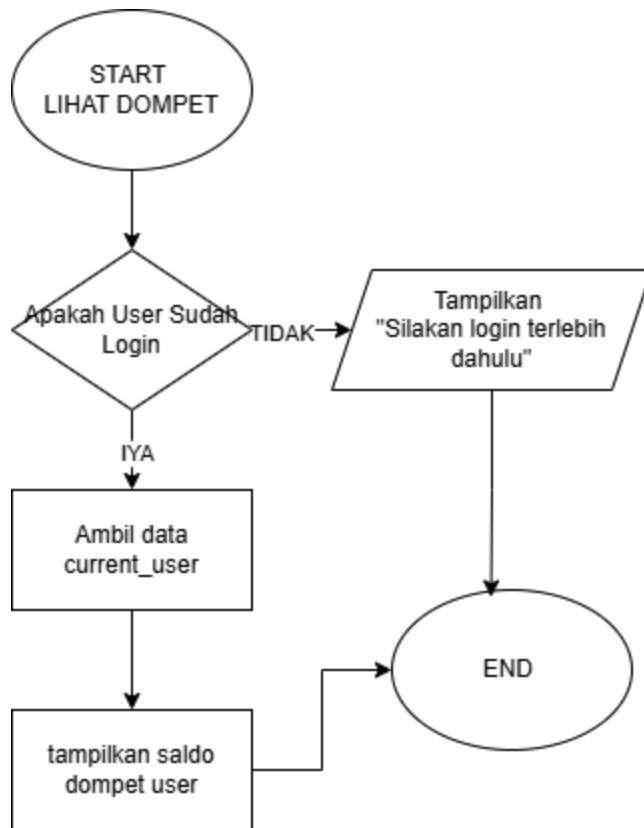
Gambar 2.6.18 Flowchart Exit



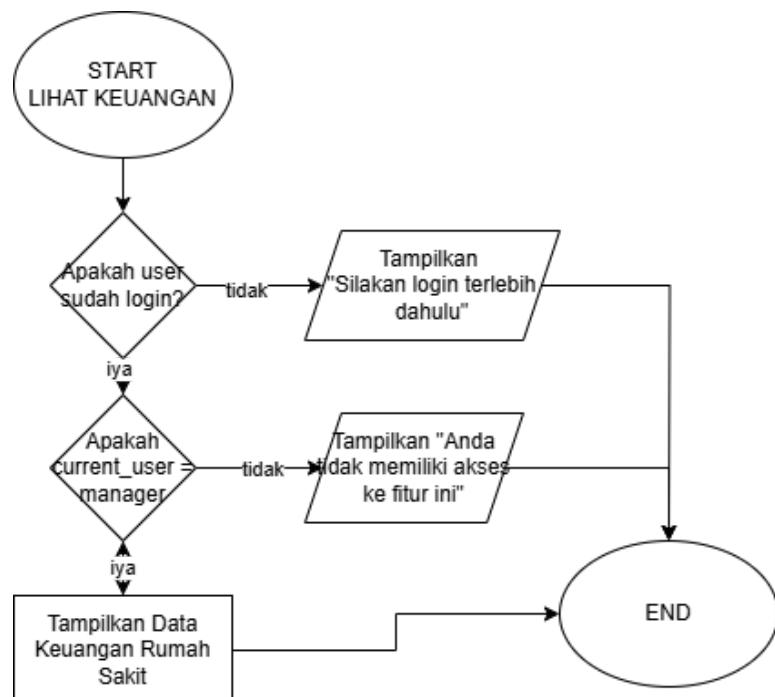
Gambar 2.6.19 Flowchart SkipAntrian



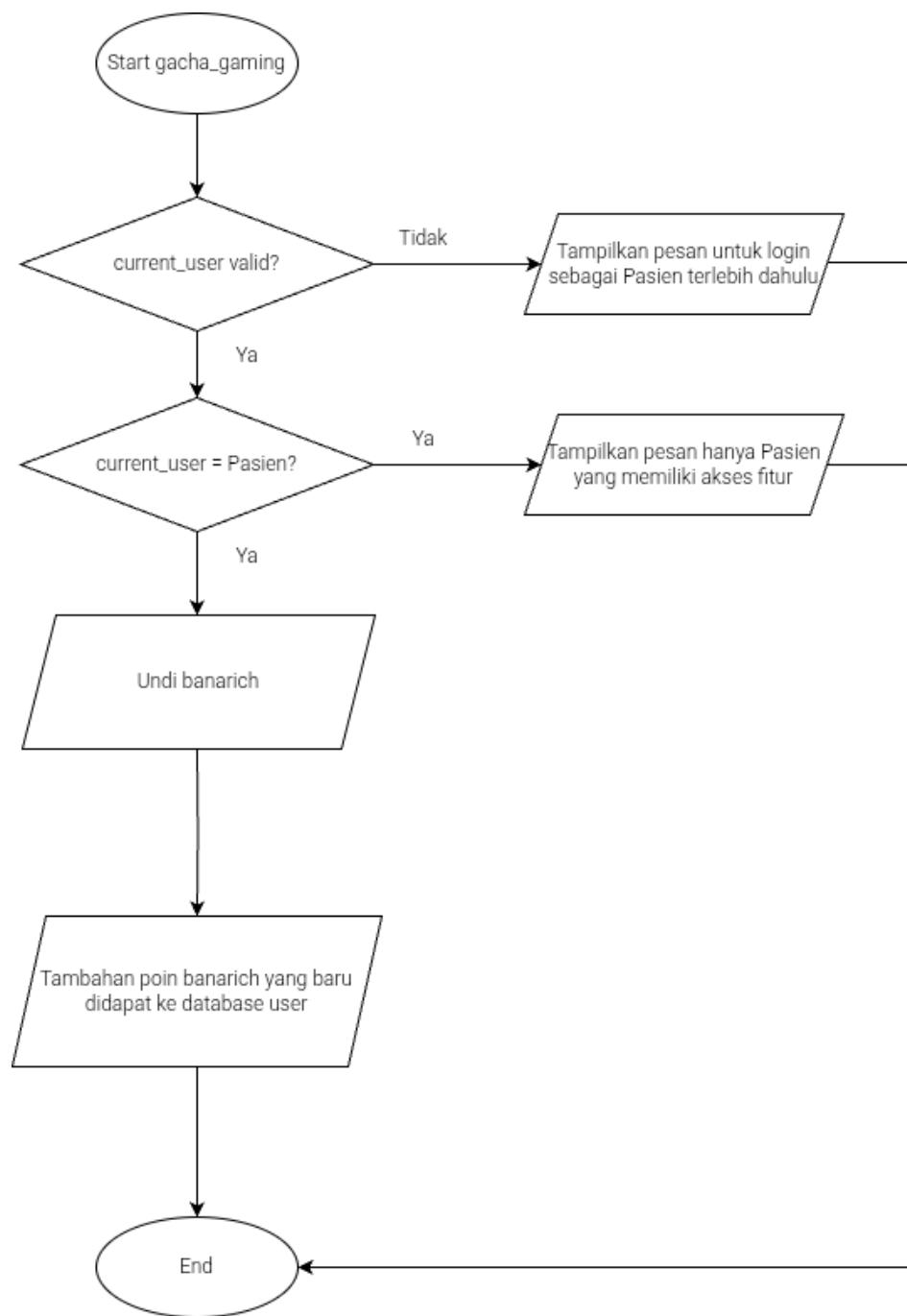
Gambar 2.6.20 Flowchart Cancel Antrian



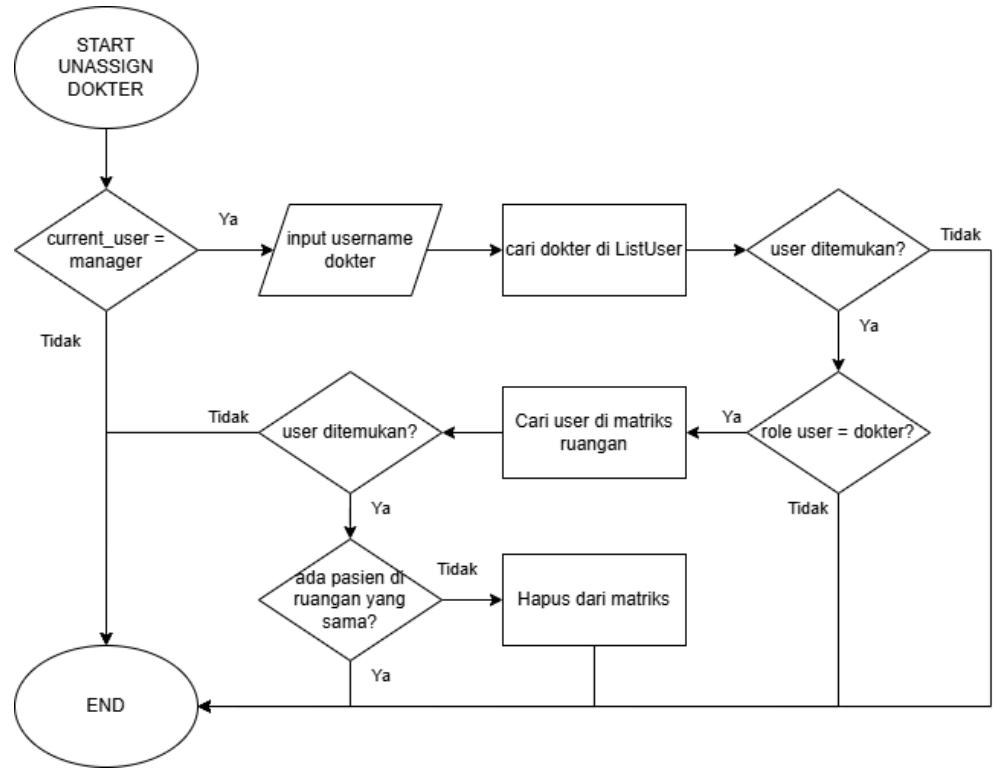
Gambar 2.6.21 Flowchart Lihat Dompet (Bananarich)



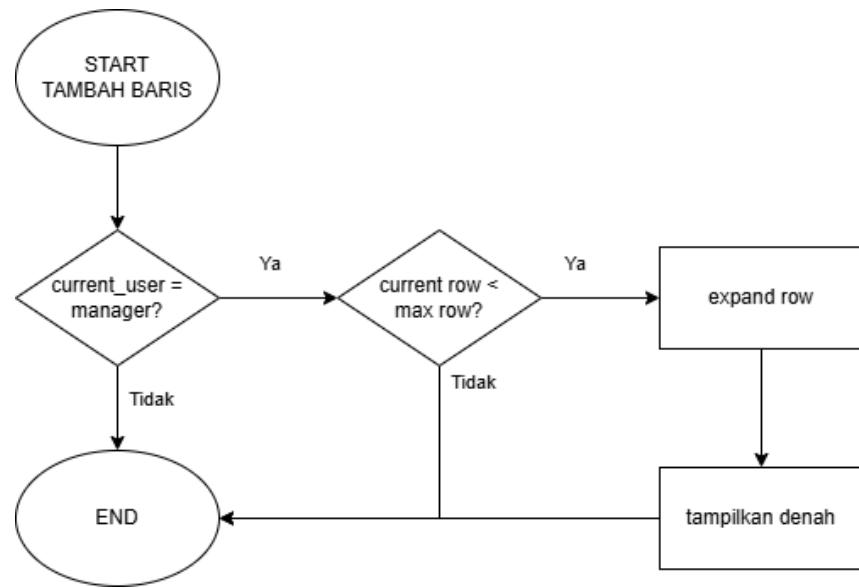
Gambar 2.6.22 Flowchart Lihat Keuangan (Banarich)



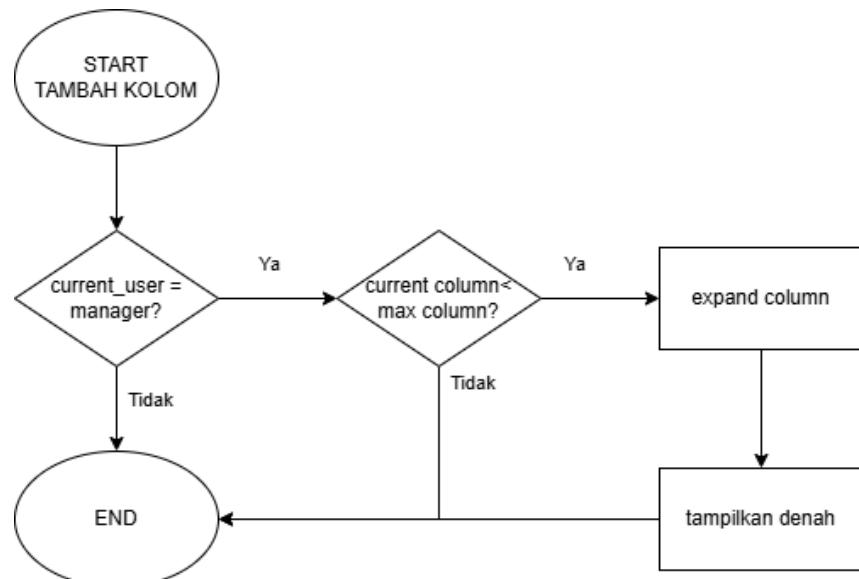
Gambar 2.6.23 Flowchart Gacha Gaming (Banarich)



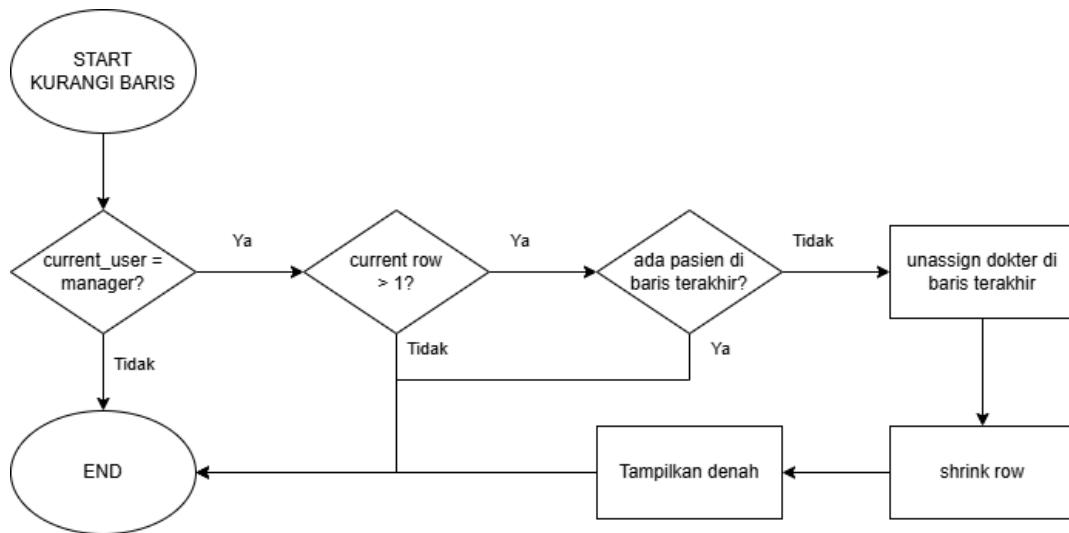
Gambar 2.6.24 Flowchart Unassign Dokter (Denah Dinamis)



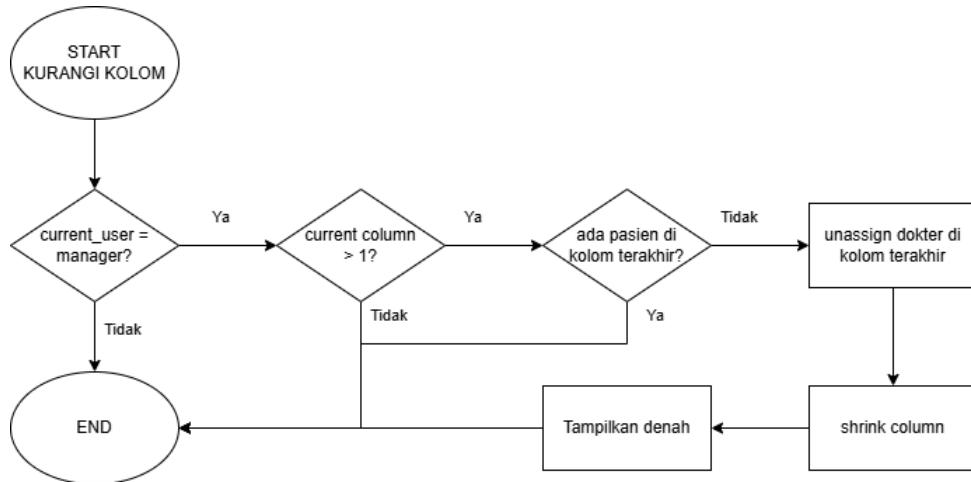
Gambar 2.6.25 Flowchart Tambah Baris (Denah Dinamis)



Gambar 2.6.26 Flowchart Tambah Kolom (Denah Dinamis)



Gambar 2.6.27 Flowchart Kurangi Baris (Denah Dinamis)



Gambar 2.6.28 Flowchart Kurangi Kolom (Denah Dinamis)

## 2.7 Spesifikasi dan Notasi Algoritmik

### 2.7.1 Kamus Global

1. Obat

```
type Obat : < obat_id : integer,  
              nama_obat : string >
```

2. StackObat

```
type StackObat : < data : array [0..STACK_MAX_CAPACITY-1] of Obat,  
              top : integer >
```

3. ListObat

```
type ListObat : < data : array [0..MAX_OBAT-1] of Obat,  
              jumlah : integer >
```

4. Penyakit

```

type Penyakit :
< id : integer,
  nama : string,
  suhu_min, suhu_max : real,
  tekanan_sistolik_min : integer,
  tekanan_sistolik_max : integer,
  tekanan_diastolik_min : integer,
  tekanan_diastolik_max : integer,
  detak_jantung_min : integer,
  detak_jantung_max : integer,
  saturasi_oksigen_min : integer,
  saturasi_oksigen_max : integer,
  kadar_gula_min : integer,
  kadar_gula_max : integer,
  berat_badan_min : real,
  berat_badan_max : real,
  tinggi_badan_max : integer,
  tinggi_badan_min : integer,
  kadar_kolesterol_min : integer,
  kadar_kolesterol_max : integer,
  kadar_kolesterol_ldl_min : integer,
  kadar_kolesterol_ldl_max : integer,
  trombosit_min : integer,
  trombosit_max : integer>

```

#### 5. ListPenyakit

```

type ListPenyakit : < data : array [0..MAX_PENYAKIT-1] of Penyakit,
                      jumlah : integer>

```

#### 6. User

```

type User : < id : integer,
             username, password, role : string,
             suhu_tubuh : real,
             tekanan_darah_sistolik : integer,
             tekanan_darah_diastolik : integer,
             detak_jantung : integer,
             saturasi_oksigen : integer,
             kadar_gula_darah : integer,
             berat_badan : real,
             tinggi_badan : integer,
             kadar_kolesterol : integer,
             kadar_kolesterol_ldl : integer

```

```

trombosit          : integer,
inventory         : ListObat,
perut              : StackObat,
riwayat_penyakit  : Penyakit >

```

7. SetDin

```

type SetDin : < buffer      : array of User,
                capacity    : integer,
                jumlah     : integer >

```

8. ListDin

```

type ListDin : < buffer      : array of User,
                capacity    : integer,
                jumlah     : integer >

```

9. ListUser

```

type ListUser : < id_list    : ListDin,
                  uname_set   : SetDin,
                  jumlah     : integer >

```

10. UsernameSet

```

type UsernameSet : < user_name   : array [0..MAX_USER-1] of string,
                     length       : integer >

```

11. Queue

```

type Queue : < pasien : User,
               next   : Address >

```

12. MapEntry

```

type MapEntry : < key      : string,
                 value    : ListObat >

```

13. MapObatPenyakit

```

type MapObatPenyakit : < data    : array [0..MAX_PENYAKIT-1] of MapEntry,
                           length  : integer >

```

14. MatriksRuangan

```

type MatriksRuangan : < row      : integer,
                         column    : integer,
                         ruang     : array [1..MAX_ROWS, 1..MAX_COLS]
                           of Ruangan >

```

15. Ruangan

```

type Ruangan : < kapasitas  : integer,
                  dokter     : user,
                  antrean    : queue >

```

### 2.7.2 ADT Map

1. procedure createMap(input/output m\_o\_p: MapObatPenyakit)

- { I.S. m\_o\_p belum terinisialisasi }  
 { F.S. m\_o\_p diinisialisasi dengan panjang 0, seluruh key kosong, dan semua list obat kosong }
2. procedure destroyMap(input/output m\_o\_p: MapObatPenyakit)
 

{ I.S. m\_o\_p berisi data penyakit dan list obat }  
 { F.S. Semua list obat dikosongkan, dan panjang map di-reset ke 0 }
  3. procedure printMapObatPenyakit(input map: MapObatPenyakit, input length: integer)
 

{ I.S. map berisi data penyakit dan daftar obat }  
 { F.S. Data map ditampilkan ke layar dalam format penyakit dan daftar obatnya }

### 2.7.3 ADT Matrix

1. procedure createRuang(input/output ruang: Ruangan)
 

{ I.S. ruang belum terinisialisasi }  
 { F.S. ruang diinisialisasi dengan kapasitas 0, dokter kosong, antrean kosong }
2. procedure destroyRuang(input/output ruang: Ruangan)
 

{ I.S. ruang terisi dokter dan/atau antrean }  
 { F.S. kapasitas di-reset, antrean dihancurkan, dokter di-reset }
3. function isRuangEmpty(ruang: Ruangan) → boolean

{ Menerima ruangan dan mengembalikan true jika tidak ada dokter di dalam ruangan }
4. procedure printInfoRuang(input denah: MatriksRuang, input baris, kolom: integer)
 

{ I.S. denah terdefinisi, indeks baris dan kolom valid }  
 { F.S. Menampilkan info kapasitas, dokter, pasien aktif, dan antrean dari ruangan terpilih }
5. procedure createMatriksRuang(input/output denah: MatriksRuang)
 

{ I.S. denah belum terisi }  
 { F.S. Semua elemen ruangan di denah diinisialisasi, row dan column diset 0 }
6. procedure expandRow(input/output denah: MatriksRuang)
 

{ I.S. denah memiliki baris sebanyak n }  
 { F.S. jumlah baris bertambah menjadi n+1 }
7. procedure expandCol(input/output denah: MatriksRuang)
 

{ I.S. denah memiliki kolom sebanyak n }  
 { F.S. jumlah kolom bertambah menjadi n+1 }
8. procedure shrinkRow(input/output denah: MatriksRuang)
 

{ I.S. denah memiliki baris lebih dari 0 }  
 { F.S. jumlah baris dikurangi 1 }
9. procedure shrinkCol(input/output denah: MatriksRuang)
 

{ I.S. denah memiliki kolom lebih dari 0 }  
 { F.S. jumlah kolom dikurangi 1 }

10. procedure setDokterID(input/output denah: MatriksRuang, input index, id\_dokter: integer, input database: ListUser)
   
 { I.S. database berisi daftar user, index dan id valid }
   
 { F.S. dokter dari database dengan ID tertentu ditetapkan ke ruangan pada index }
11. procedure addPasienToRuang(input/output denah: MatriksRuang, input index, id\_pasien: integer, input database : ListUser)
   
 { I.S. database berisi user pasien, index dan id valid }
   
 { F.S. pasien ditambahkan ke antrean ruangan pada index }
12. function isUserInMatriks(user: User, denah: MatriksRuang) → boolean
  
 { Menerima user dan mengembalikan true jika user ditemukan di dalam ruangan sebagai dokter atau pasien }
13. procedure destroyMatriksRuang(input/output denah: MatriksRuang)
   
 { I.S. denah terisi ruangan }
   
 { F.S. semua ruangan dihancurkan dan row, column diset 0 }

#### 2.7.4 ADT Obat

1. procedure createObat(input/output new\_obat: Obat)
   
 { I.S. new\_obat belum diinisialisasi }
   
 { F.S. new\_obat.obat\_id = 0 dan new\_obat.nama\_obat = "" }
2. \ufunction isObatValid(obat: Obat) → boolean
  
 { Mengembalikan true jika obat obat id dan nama obat valid, bila tidak maka akan mengembalikan false }
3. procedure createListObat(input/output list\_obat: ListObat)
   
 { I.S. list\_obat belum diinisialisasi }
   
 { F.S. all elements of list\_obat initialized empty, list\_obat.jumlah = 0 }
4. function isEmptyObat(list\_obat: ListObat) → boolean
  
 { Mengembalikan true jika list\_obat tidak memiliki elemen obat (kosong), dan false jika terdapat minimal satu obat di dalam list\_obat }
5. function isFullObat(list\_obat: ListObat) → boolean
  
 { Mengembalikan true bisa list obat sudah full, dan false bila belum }
6. procedure insertLastObat(input/output list\_obat: ListObat, input obat: Obat)
   
 { I.S. list\_obat mungkin belum penuh }
   
 { F.S. add obat to ke ujung list obat bila belum penuh }
7. procedure deleteLastObat(input/output list\_obat: ListObat, output deleted\_obat: Obat)
   
 { I.S. list\_obat mungkin tidak kosong }
   
 {F.S. Menghapus elemen terakhir dari list\_obat dan menyimpannya di deleted\_obat}
8. function lengthListObat(list\_obat: ListObat) → integer
  
 { Mengembalikan panjang isi list\_obat }
9. procedure printListObat(input list\_obat: ListObat)

```
{ I.S. menerima list_obat }
{ F.S. print all obat in list_obat }
```

#### 2.7.5 ADT Penyakit

1. procedure createPenyakit(input/output p: Penyakit)  
    { I.S. p mungkin belum terinisialisasi }  
    { F.S. semua atribut p diinisialisasi ke nilai default (0 atau "") }
2. function isPenyakitValid(p: Penyakit) → boolean  
    { Mengembalikan nilai TRUE jika p.id != 0 dan p.nama tidak kosong, else FALSE }
3. procedure createListPenyakit(input/output lp: ListPenyakit)  
    { I.S. lp mungkin belum terinisialisasi }  
    { F.S. semua elemen dalam lp diinisialisasi ke default dan lp.jumlah = 0 }
4. function lengthPenyakit(database\_penyakit: ListPenyakit) → integer  
    { Mengembalikan nilai jumlah elemen valid dalam database\_penyakit }
5. function isEmpty(lp: ListPenyakit) → boolean  
    { Mengembalikan nilai TRUE jika lp kosong, else FALSE }
6. function isFull(lp: ListPenyakit) → boolean  
    { Mengembalikan nilai TRUE jika lp penuh, else FALSE }
7. procedure insertLast(input/output lp: ListPenyakit, input p: Penyakit)  
    { I.S. lp mungkin belum penuh }  
    { F.S. p ditambahkan ke akhir lp jika ada ruang, else cetak pesan error }
8. procedure deleteLast(input/output lp: ListPenyakit, output p: Penyakit)  
    { I.S. lp mungkin tidak kosong }  
    { F.S. elemen terakhir dihapus dari lp dan disimpan ke p, else cetak pesan error }
9. function getPenyakit(lp: ListPenyakit, index: integer) → Penyakit  
    { Mengembalikan nilai penyakit pada index jika valid, else penyakit kosong dan cetak pesan error }
10. function getPenyakitByName(daftar\_penyakit: ListPenyakit, nama: string) → Penyakit  
        { Mengembalikan nilai penyakit dengan nama sesuai (case-insensitive), else penyakit kosong }

#### 2.7.6 ADT Queue

1. function createQueue() → Queue\*  
    { Mengembalikan pointer ke Queue kosong yang baru dibuat }
2. procedure destroyQueue(input/output antrean: Queue)  
    { I.S. antrean teralokasi dengan sejumlah elemen }  
    { F.S. semua node dalam antrean terdeallocate }
3. function isQueueEmpty(antrean: Queue) → boolean  
    { Mengembalikan TRUE jika antrean kosong (pasien tidak valid dan next NULL), else FALSE }

4. function queueLength(antrean: Queue) → integer  
     { Mengembalikan jumlah elemen dalam antrean }
5. procedure enqueue(input/output antrean: Queue, input pasien: User)  
     { I.S. antrean sudah terinisialisasi }  
     { F.S. pasien ditambahkan ke belakang antrean }
6. function dequeue(input/output antrean: Queue) → User  
     { Mengembalikan User di depan antrean dan menghapus node depan }
7. function findInQueue(antrean: Queue, pasien: User) → integer  
     { Mengembalikan indeks posisi pasien dalam antrean, -1 jika tidak ditemukan }
8. function getQueueHead(antrean: Queue\*) → User  
     { Mengembalikan User di depan antrean tanpa menghapus }

#### 2.7.7 ADT Stack

1. procedure createStackObat(input/output s\_o: StackObat)  
     { I.S. s\_o mungkin belum terinisialisasi }  
     { F.S. stack s\_o sudah dibuat dan kosong, s\_o.top = -1 }
2. procedure destroyStackObat(input/output s\_o: StackObat)  
     { I.S. s\_o terdefinisi dan berisi elemen stack }  
     { F.S. s\_o dikosongkan dan semua data elemen direset ke default, s\_o.top = -1 }
3. function isStackEmpty(input s\_o: StackObat) → boolean  
     { Mengembalikan true jika stack kosong, false jika tidak }
4. function isStackFull(input s\_o: StackObat) → boolean  
     { Mengembalikan true jika stack penuh, false jika tidak }
5. function stackLength(input s\_o: StackObat) → integer  
     { Mengembalikan jumlah elemen dalam stack }
6. function getTopStack(input s\_o: StackObat) → Obat  
     { Mengembalikan elemen teratas stack tanpa menghapusnya, jika kosong mengembalikan Obat kosong (default) }
7. procedure pushStack(input/output s\_o: StackObat, input o: Obat)  
     { I.S. stack s\_o tidak penuh }  
     { F.S. elemen o ditambahkan ke atas stack, s\_o.top bertambah 1 }
8. function popStack(input/output s\_o: StackObat) → Obat  
     { Mengembalikan elemen teratas stack dan menghapusnya, jika kosong mengembalikan Obat kosong (default) }

#### 2.7.8 ADT User

1. procedure createUser(input/output account: User)  
     { I.S. account belum terinisialisasi }  
     { F.S. account terisi nilai default }
2. procedure destroyUser(input/output account: User)  
     { I.S. account berisi data User }  
     { F.S. account ter-reset ke nilai default }

3. function isUserValid(account: User) → boolean  
     { Mengembalikan TRUE jika username tidak kosong, FALSE jika kosong }
4. function isSameUser(input account1: User, input account2: User) → boolean  
     { Mengembalikan TRUE jika id dan username kedua user sama }
5. procedure createListDin(input/output list: ListDin, input capacity: integer)  
     { I.S. list belum terinisialisasi }  
     { F.S. list teralokasi dengan kapasitas tertentu dan setiap elemen user diinisialisasi dengan createUser }
6. procedure createSetDin(input/output set: SetDin, input capacity: integer)  
     { I.S. set belum teralokasi }  
     { F.S. set teralokasi dengan kapasitas tertentu dan setiap elemen user diinisialisasi dengan createUser }
7. procedure createListUser(input/output accountList: ListUser, input capacity: integer)  
     { I.S. accountList belum terinisialisasi }  
     { F.S. accountList berisi ListDin dan SetDin dengan kapasitas tertentu, jumlah user = 0 }
8. procedure destroyListDin(input/output list: ListDin)  
     { I.S. list berisi data dan sudah dialokasikan }  
     { F.S. list dikosongkan dan memori buffer dibebaskan }
9. procedure destroySetDin(input/output set: SetDin)  
     { I.S. set berisi data dan sudah dialokasikan }  
     { F.S. set dikosongkan dan memori buffer dibebaskan }
10. procedure destroyListUser(input/output accountList: ListUser)  
     { I.S. accountList berisi data user }  
     { F.S. accountList dikosongkan, memori list dan set dibebaskan }
11. procedure geserListDin(input/output list: ListDin, input start\_idx: integer, input jarak: integer)  
     { I.S. list berisi data user }  
     { F.S. elemen list digeser sejauh jarak mulai dari indeks start\_idx ke kanan (jarak > 0) atau ke kiri (jarak < 0) }
12. procedure geserSetDin(input/output set: SetDin, input start\_idx: integer, input jarak: integer)  
     { I.S. set berisi data user }  
     { F.S. elemen set digeser sejauh jarak mulai dari indeks start\_idx ke kanan (jarak > 0) atau ke kiri (jarak < 0) }
13. procedure expandShrinkListUser(input/output accountList: ListUser, input modifier: float)  
     { I.S. accountList berisi data dan kapasitas tertentu }

- { F.S. kapasitas accountList diperbesar atau diperkecil sesuai modifier dan data disalin ke buffer baru }
14. procedure insertUser(input/output accountList: ListUser, input account: User)
 

{ I.S. accountList berisi data user }  
 { F.S. account ditambahkan ke accountList di posisi yang tepat sesuai ID dan username, kapasitas diperbesar jika perlu }
  15. procedure deleteUser(input/output accountList: ListUser, input account: User)
 

{ I.S. accountList berisi user yang ingin dihapus }  
 { F.S. user dihapus dari accountList, kapasitas diperkecil jika perlu }
  16. function listLength(input accountList: ListUser) → integer

{ Mengembalikan jumlah user dalam accountList }
  17. function getUserByID(input accountList: ListUser, input id\_user: integer) → User pointer

{ Mengembalikan pointer ke user dengan ID id\_user, atau NULL jika tidak ditemukan }
  18. function getUserByName(input accountList: ListUser, input nama: string) → User pointer

{ Mengembalikan pointer ke user dengan username nama, atau NULL jika tidak ditemukan }
  19. function isUsernameInSet(input accountList: ListUser, input nama: string) → boolean

{ Mengembalikan TRUE jika username nama ada dalam set, FALSE jika tidak }
  20. procedure tambahObatKeInventory(input/output user: User, input obat: Obat)
 

{ I.S. user memiliki inventory obat, mungkin kosong }  
 { F.S. obat ditambahkan ke inventory user, dan variabel global CountUserDenganObat bertambah jika inventory sebelumnya kosong dan bukan saat load config }

### 2.7.9 ADT Set

1. procedure setUsernameSet(input/output set\_nama: UsernameSet)
 

{ I.S. set\_nama belum terisi }  
 { F.S. set\_nama kosong dengan semua elemen string "" dan length = 0 }
2. procedure destroyUsernameSet(input/output set\_nama: UsernameSet)
 

{ I.S. set\_nama berisi data username }  
 { F.S. set\_nama dikosongkan dan length di-reset ke 0 }
3. function isUsernameInSet(set\_nama: UsernameSet, nama: string) → boolean

{ Mengembalikan TRUE jika nama terdapat di set\_nama, else FALSE }
4. procedure addUsernameToSet(input/output set\_nama: UsernameSet, input nama: string)
5. procedure copyListToSet(input database: ListUser, input/output daftar\_nama: UsernameSet)

### 2.7.10 File Eksternal

1. function folderExists(folderName: string) → integer  
    { Mengembalikan 1 jika config.txt ada di folder, 0 jika tidak }
2. procedure TambahObatKePasien(input/output database: ListUser, input pasienId: integer, input obatId: integer, input semuaObat: ListObat)  
    { I.S. database, semuaObat terdefinisi; f.s. inventory pasien bertambah 1 obat jika cocok dan belum penuh }
3. function parse\_field(file: FILE pointer, buffer: string, max\_len: integer) → integer  
    { Mengembalikan EOF jika akhir file, atau karakter pemisah field }
4. function bacaPenyakitCSV(filename: string, penyakitArr: array of Penyakit, maxPenyakit: integer) → integer  
    { Membaca file CSV penyakit dan mengisi penyakitArr, mengembalikan jumlah penyakit yang berhasil dibaca }
5. procedure bacaUserCSV(input/output folderName: string, input/output database: ListUser, input daftar\_penyakit: ListPenyakit)  
    { Membaca file user.csv dari folderName dan mengisi ListUser database }
6. procedure BacaPenyakitCSV(input folderName: string, output lp: array of recordPenyakit, output CountPenyakit: integer)
7. procedure bacaConfig(input folderName: string, input/output denah: MatriksRuang, input/output database: ListUser, input semuaObat: ListObat)  
    { Membaca file config.txt }
8. procedure saveConfig(input/output folderName: string, input denah: MatriksRuang, input database: ListUser)  
    { Menyimpan konfigurasi ke file config.txt }

### 2.7.11 Fitur

1. F01 - Login  
procedure Login(input database: ListUser, input/output current\_user: User)  
    { I.S. Data user telah dimuat ke dalam database }  
    { F.S. Jika login berhasil, current\_user berisi data user yang login. jika gagal, current\_user tidak diubah }

#### KAMUS LOKAL

username	:	<u>string</u>
password	:	<u>string</u>
found_user	:	User
i	:	<u>integer</u>
validUsername	:	<u>boolean</u>

#### ALGORITMA

```
if (isValidUser(current_user)) then
```

```

output("Kamu sudah LOGIN sebagai " +
current_user.role + " " + current_user.username +
"!")
output("Silakan LOGOUT sebelum LOGIN sebagai pengguna
lain.")
→

loginHeaderInterface()

{ Input Username }
validUsername ← false
repeat
    output("Username: ")
    input(username)
    hapus newline dari username

    if (strlen(username) = 0) then
        output("Username tidak boleh kosong!")
    else
        found_user ← NULL
        i ← 1
    while (i ≤ database.jumlah and found_user = NULL) do
        if (lowercase(username) =
lowercase(database.data[i].username)) then
            found_user ← database.data[i]
            i ← i + 1

            if (found_user = NULL) then
output("Tidak ada Manager, Dokter, atau Pasien yang
bernama " + username + "!")
            else
                validUsername ← true
    until (validUsername)

{ Cek apakah user sudah meninggal }
if (found_user.nyawa ≤ 0) then
output("User " + found_user.username + " sudah
meninggal dan tidak dapat login!")
    output("💀 Rest in Peace 💀")
→

{ Input Password }
berhasilLogin ← false
while (not berhasilLogin) do
    output("Password: ")

```

```

input(password)
hapus newline dari password

if           (lowercase(password) = 
lowercase(found_user.password)) then
    current_user ← found_user
    welcomeLoginInterface(current_user)
    berhasilLogin ← true
else
output("Username atau password salah untuk pengguna
yang bernama " + username + "!")

```

## 2. F02 - Register Pasien

```

procedure registerPasien(input/output username_set:
UsernameSet, input/output database: ListUser)
{ I.S. username_set dan database mungkin telah berisi data user sebelumnya }
{ F.S. Jika username valid dan unik, maka user baru ditambahkan ke database
dan username_set }

```

### KAMUS LOKAL

username, password	: <u>string</u>
temp_username	: <u>string</u>
valid	: <u>boolean</u>
i, m, l, r	: <u>integer</u>
cmp	: <u>integer</u>
set_name_lower	: <u>string</u>
pos	: <u>integer</u>
new_user	: <u>User</u>

### ALGORITMA

```

{ Validasi: sudah login }
if (isValid(current_user)) then
    output("Kamu sudah LOGIN sebagai " +
current_user.role + " " + current_user.username
+ "!")
    output("Silakan LOGOUT sebelum REGISTER sebagai
pengguna lain.")
    →

registerHeaderInterface()

repeat
    valid ← true

    output("Username: ")

```

```

input(username)
hapusNewline(username)

{ Validasi 1: hanya huruf }
i ← 0
while (i < strlen(username)) do
    if (not((username[i] ≥ 'a' AND username[i]
    ≤ 'z') OR (username[i] ≥ 'A' AND
    username[i] ≤ 'Z'))) then
        output("Registrasi gagal! Username
        hanya boleh berisi huruf (tanpa angka
        atau simbol).")
        valid ← false
        break
    i ← i + 1

    { Validasi 2: tidak boleh ada username duplikat
}

if (valid) then
    if (isUsernameInSet(database, username))
    then
        output("Registrasi gagal! Username
        sudah terdaftar.")
        valid ← false
until (valid)

output("Password: ")
input(password)
hapusNewline(password)

{ Buat user baru }
createUser(new_user)

if (database.jumlah > 0) then
    new_user.id ← database.data[database.jumlah -
    1].id + 1
else
    new_user.id ← 1

new_user.username ← username
new_user.password ← password
new_user.role ← "pasien"

{ Tambahkan ke database }
insertUser(database, new_user)

```

```
output("Pasien " + username + " berhasil ditambahkan  
dengan ID " + toString(new_user.id) + "!"")
```

### 3. F03 - Logout

```
procedure logout(input/output current_user : User)  
{ I.S. current_user sudah terdefinisi }  
{ F.S. current_user di-reset menjadi user baru }
```

## KAMUS LOKAL

-

## ALGORITMA

```
logoutHeaderInterface()  
  
if (not isUserValid(current_user)) then  
    output("Kamu belum LOGIN sebagai role apapun.")  
    →  
  
    createUser(current_user)  
    output("Berhasil LOGOUT!")  
    output("Silakan LOGIN kembali untuk mengakses fitur  
rumah sakit.")
```

### 4. F04 - Lupa Password

```
function validasiKodeUnik(username: string, kodeUnik:  
string) → integer  
{ Mengembalikan 1 jika kode unik sesuai hasil encoding dari username, 0 jika  
tidak }
```

## KAMUS LOKAL

i, k	:	<u>integer</u>
len	:	<u>integer</u>
temp	:	<u>string</u>
count	:	<u>integer</u>
hurufUsername	:	<u>char</u>
hurufCount	:	<u>string</u>
hasil	:	<u>integer</u>

## ALGORITMA

```
len ← strlen(username)  
temp ← alokasiStringKosong(len + 1)  
k ← 0  
i ← 0  
while (i < len) do  
    count ← 1
```

```

        hurufUsername ← username[i]

        { Hitung jumlah kemunculan karakter berurutan
          yang sama }
        while ((i + 1 < len) and (username[i + 1] =
        hurufUsername)) do
            count ← count + 1
            i ← i + 1

        { Jika berulang, masukkan angka count ke temp }
        if (count > 1) then
            hurufCount ← konversiIntegerKeString(count)
            l ← 0
            while (l < strlen(hurufCount)) do
                temp[k] ← hurufCount[l]
                k ← k + 1
                l ← l + 1

            { Masukkan karakter ke temp }
            temp[k] ← hurufUsername
            k ← k + 1
            i ← i + 1

            temp[k] ← karakterNull { Penutup string '\0' }

        if (strcmp(temp, kodeUnik) = 0) then
            hasil ← 1
        else
            hasil ← 0

        dealokasi(temp)
        → hasil

procedure gantiPasswordDiList(input/output user: User,
input passwordBaru: string)
{ I.S. user dan passwordBaru sudah terdefinisi }
{ F.S. Password milik user diubah menjadi passwordBaru }

```

## KAMUS LOKAL

---

ALGORITMA

```

strncpy(user.password, passwordBaru,
sizeof(user.password) - 1)
user.password[sizeof(user.password) - 1] ← '\0'

procedure lupaPassword(input/output database: ListUser,
input current_user: User)
{ I.S. Data user tersimpan dalam database }
{ F.S. Password user diubah jika username dan kode unik valid }

```

## KAMUS LOKAL

```

username, kodeUnik, passwordBaru      : string
found_index, i                      : integer
user_by_id                          : User

```

## ALGORITMA

```

if (isValid(current_user)) then
    output("Kamu sudah LOGIN sebagai " +
    current_user.role + " " + current_user.username
    + "!")
    output("Silakan LOGOUT sebelum menggunakan
    fitur LUPA_PASSWORD.")
    →

lupaPasswordHeaderInterface()

output("Username: ")
input(username)
hapus newline dari username

found_index ← -1
i ← 0
while (i < database.jumlah and found_index = -1) do
    if (username = database.data[i].username) then
        found_index ← i
        break
    i ← i + 1

    if (found_index = -1) then
        output("Username tidak terdaftar!")
    →

output("Kode Unik: ")
input(kodeUnik)
hapus newline dari kodeUnik

```

```

if (not (validasiKodeUnik(username, kodeUnik))) then
    output("Kode unik salah!")
    →

    output("Halo " + database.data[found_index].role + "
" + database.data[found_index].username + ", silakan
daftarkan ulang password anda!")
    output("Password Baru: ")
    input(passwordBaru)

gantiPasswordDiList(database.data[found_index],
passwordBaru)

user_by_id           ←      getUserByID(database,
database.data[found_index].id)
if (user_by_id ≠ NULL) then
    gantiPasswordDiList(user_by_id, passwordBaru)

output("Password berhasil diubah! Silakan login
dengan password baru.")

```

##### 5. F05 - Menu dan Help

```

procedure help(input current_user: User)
{ I.S. User sedang menggunakan sistem }
{ F.S. Ditampilkan daftar prosedur yang tersedia untuk current_user }

```

#### KAMUS LOKAL

```

role_lower : string
i           : integer

```

#### ALGORITMA

```

output("===== HELP =====")
if not(isUserValid(current_user)) then
    output("Kamu belum login sebagai role apapun.
Silahkan login terlebih dahulu.")
    output("1. LOGIN: Masuk ke dalam akun yang
sudah terdaftar")
    output("2. REGISTER: Membuat akun baru")
else
    role_lower ← current_user.role
    i traversal [0..strlen(role_lower)-1]
        if (role_lower[i] ≥ 'A' and role_lower[i]
≤ 'Z') then

```

```

        role_lower[i] ← role_lower[i] + ('a'
        - 'A')

    if (contains(role_lower, "dokter")) then
        output("Halo,           Dokter           "
        current_user.username + "!")
        output("Berikut adalah hal-hal yang dapat
        kamu lakukan:")
        output("1. LOGOUT")
        output("2. LUPA_PASSWORD")
        output("3. LIHAT_DENAH")
        output("4. DIAGNOSIS")
        output("5. NGOBATIN")
        output("6. EXIT")
    else if (contains(role_lower, "pasien")) then
        output("Selamat datang,           "
        current_user.username + "!")
        output("Berikut adalah hal-hal yang dapat
        kamu lakukan:")
        output("1. LOGOUT")
        output("2. LUPA_PASSWORD")
        output("3. LIHAT_DENAH")
        output("4. BOLEH_PULANG")
        output("5. DAFTAR_CHECKUP")
        output("6. STATUS_ANTREAN")
        output("7. MINUM_OBAT")
        output("8. MINUM_PENAWAR")
        output("9. EXIT")
    else if (contains(role_lower, "manager")) then
        output("Halo       Manager           "
        current_user.username + "!")
        output("Berikut adalah hal-hal yang dapat
        kamu lakukan:")
        output("1. LOGOUT")
        output("2. LUPA_PASSWORD")
        output("3. LIHAT_DENAH")
        output("4. LIHAT_USER")
        output("5. CARI_USER")
        output("6. LIHAT_ANTREAN")
        output("7. TAMBAH_DOKTER")
        output("8. EXIT")
    else
        output("Data      tidak      ditemukan      dalam
        sistem.")

```

```

output("Footnote:")
output("1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar")
output("2. Jangan lupa untuk memasukkan input yang valid")
output("3. Gunakan command yang sesuai dengan role Anda")
output("=====")

```

#### 6. F06 - Denah Rumah Sakit

```

procedure lihatDenah(input current_user: User, input denah: MatriksRuang)
{ I.S. User sudah login dan sistem menyimpan denah ruangan }
{ F.S. Ditampilkan denah ruangan jika user valid }

```

#### KAMUS LOKAL

i, j : integer

#### ALGORITMA

```

lihatDenahHeaderInterface()
if not(isValidUser(current_user)) then
    output("Login untuk melihat denah!")
    →

j traversal [0..denah.column - 1]
    output("----", tanpa newline)
    if (j = denah.column - 1) then
        output("+")

i traversal [0..denah.row - 1]
    j traversal [0..denah.column - 1]
        output("| " + karakter(i+65) + (j+1) + "
", tanpa newline)
        if (j = denah.column - 1) then
            output("|")

    j traversal [0..denah.column - 1]
        output("----", tanpa newline)
        if (j = denah.column - 1) then
            output("+")

procedure lihatRuang(input current_user: User, input denah: MatriksRuang)
{ I.S. User sedang dalam sistem dan mengakses ruangan }
{ F.S. Informasi ruangan ditampilkan jika kode valid }

```

## KAMUS LOKAL

```
ruang_code : string
row, col   : integer
```

## ALGORITMA

```
lihatRuanganHeaderInterface()

if      (current_user.role      ≠      "manager"      and
current_user.role ≠ "dokter" and current_user.role ≠
"pasien") then
    output("Anda tidak memiliki izin untuk
mengakses ruangan!")
    →

output("Masukkan kode ruangan: ")
input(ruang_code)

if (strlen(ruang_code) < 2) then
    output("Format kode ruangan tidak valid!")
    →

row ← (ruang_code[0]) - ('A')
col ← (ruang_code[1]) - ('0') - 1

if (row < 0 and row ≥ denah.row and col < 0 or col ≥
denah.column) then
    output("Ruang dengan kode " + ruang_code + "
tidak ditemukan!")
    →

printInfoRuang(denah, row, col)
```

### 7. F07 - Lihat User

```
procedure lihatUser(input current_user: User, input
database_pengguna: ListUser)
{ Menampilkan daftar semua user dalam sistem yang diurutkan berdasarkan
pilihan manager }
```

## KAMUS LOKAL

```
kriteria_urutan, arah_urutan      : integer
jumlah_pengguna                      : integer
indeks, indeks_luar, indeks_dalam    : integer
perlu_tukar                            : boolean
hasil_perbandingan                   : integer
```

```

daftar_pengguna_terurut      : array of User
pengguna_sekarang            : User
nama_penyakit                : string

```

## ALGORITMA

```

lihatUserHeaderInterface()

if (current_user.role ≠ "manager") then
    output("Hanya manager yang dapat mengakses
    fitur ini!")
    →

if (database_pengguna.id_list.jumlah = 0) then
    output("Tidak ada pengguna yang terdaftar dalam
    sistem!")
    →

output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")
input(kriteria_urutan)

output("\\nUrutan sort?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")
input(arah_urutan)

jumlah_pengguna ← database_pengguna.id_list.jumlah
allocate daftar_pengguna_terurut with size
jumlah_pengguna

indeks transversal [0..jumlah_pengguna - 1]
    daftar_pengguna_terurut[indeks]           ←
    database_pengguna.id_list.buffer[indeks]

i transversal [0..jumlah_pengguna - 2]
    j transversal [0..jumlah_pengguna - 2 - i]
        if (kriteria_urutan = 1) then
            if (arah_urutan = 1) then
                perlu_tukar           ←
                (daftar_pengguna_terurut[j].id
                > daftar_pengguna_terurut[j + 1].id)

```

```

        else
            perlu_tukar           ←
                (daftar_pengguna_terurut[j].id
                < daftar_pengguna_terurut[j + 1].id)

        else
            hasil_perbandingan   ←
                compare_case_insensitive(daftar_pengguna_terurut[j].username,
                                            daftar_pengguna_terurut[j + 1].username)
                if (arah_urutan = 1) then
                    perlu_tukar           ←
                        (hasil_perbandingan > 0)
                else
                    perlu_tukar           ←
                        (hasil_perbandingan < 0)

            if (perlu_tukar) then
                temp ← daftar_pengguna_terurut[j]
                daftar_pengguna_terurut[j]           ←
                daftar_pengguna_terurut[j + 1]
                daftar_pengguna_terurut[j + 1] ← temp

output("ID | Nama          | Role      | Penyakit")
output("-----")
)

indeks transversal [0..jumlah_pengguna - 1]
pengguna_sekarang           ←
daftar_pengguna_terurut[indeks]
if (pengguna_sekarang.riwayat_penyakit.nama = "-")           or
length(pengguna_sekarang.riwayat_penyakit.nama) = 0) then
    nama_penyakit ← "-"
else
    nama_penyakit           ←
    pengguna_sekarang.riwayat_penyakit.nama

output(pengguna_sekarang.id + " | " +
pengguna_sekarang.username + " | " +
pengguna_sekarang.role + " | " + nama_penyakit)

deallocate daftar_pengguna_terurut

```

```

procedure lihatDokter(input current_user: User, input
database : ListUser)
{ Menampilkan daftar dokter yang terdaftar, dengan pilihan pengurutan oleh
Manager }

```

### KAMUS LOKAL

kriteria_urutan, arah_urutan	: <u>integer</u>
jumlah_dokter	: <u>integer</u>
i, j	: <u>integer</u>
perlu_tukar	: <u>boolean</u>
hasil_perbandingan	: <u>integer</u>
daftar_dokter	: <u>array of</u> User
temp, dokter_sekarang	: User

### ALGORITMA

```

lihatUserHeaderInterface()

if (current_user.role ≠ "manager") then
    output("Hanya manager yang dapat mengakses
fitur ini!")
    →

if (database.id_list.jumlah = 0) then
    output("Tidak ada user yang terdaftar!")
    →

output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output("3. Aura")
output(">>> Pilihan: ")
input(kriteria_urutan)

output("\\nUrutan sort?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")
input(arah_urutan)

jumlah_dokter ← 0
allocate daftar_dokter with size
database.id_list.jumlah

```

```

i traversal [0..database.id_list.jumlah-1]
    if (lower_case(database.id_list.buffer[i].role)
    = "dokter") then
        daftar_dokter[jumlah_dokter] ←
        database.id_list.buffer[i]
        jumlah_dokter ← jumlah_dokter + 1

    if (jumlah_dokter = 0) then
        output("Tidak ada dokter yang terdaftar.")
        deallocate daftar_dokter
    →

i traversal [0..jumlah_dokter-2]
    j traversal [0..jumlah_dokter-2-i]
        if (kriteria_urutan = 1) then
            if (arah_urutan = 1) then
                perlu_tukar ←
                (daftar_dokter[j].id >
                 daftar_dokter[j+1].id)
            else
                perlu_tukar ←
                (daftar_dokter[j].id <
                 daftar_dokter[j+1].id)
        else if (kriteria_urutan=2) then
            hasil_perbandingan ←
            compare_case_insensitive(daftar_dokte
r[j].username,
                           daftar_dokter[j+1].username)
            if (arah_urutan = 1) then
                perlu_tukar ←
                (hasil_perbandingan > 0)
            else
                perlu_tukar ←
                (hasil_perbandingan < 0)
        else if (kriteria_urutan=3) then
            if (arah_urutan = 1) then
                perlu_tukar ←
                (daftar_dokter[j].aura >
                 daftar_dokter[j+1].aura)
            else
                perlu_tukar ←
                (daftar_dokter[j].aura >
                 daftar_dokter[j+1].aura)

        if (perlu_tukar) then

```

```

        temp ← daftar_dokter[j]
        daftar_dokter[j] ← daftar_dokter[j+1]
        daftar_dokter[j+1] ← temp

output("ID | Nama | Aura")
output("-----")
i traversal [0..jumlah_dokter-1]
    output(daftar_dokter[i].id + " | " +
    daftar_dokter[i].username + " | " +
    daftar_dokter[i].aura)

deallocate daftar_dokter

procedure lihatPasien(input current_user: User, input
database: ListUser)
{ Menampilkan daftar pasien yang terdaftar beserta penyakit, dengan pilihan
pengurutan oleh manager }

```

#### KAMUS LOKAL

kriteria_urutan, arah_urutan	:	<u>integer</u>
jumlah_pasien, i, j	:	<u>integer</u>
perlu_tukar	:	<u>boolean</u>
hasil_perbandingan	:	<u>integer</u>
daftar_pasien	:	<u>array of</u> User
temp, pasien_sekarang	:	User
nama_penyakit	:	<u>string</u>

#### ALGORITMA

```

lihatUserHeaderInterface()

if (current_user.role ≠ "manager") then
    output("Hanya manager yang dapat mengakses
fitur ini!")
    →

if (database.id_list.jumlah = 0) then
    output("Tidak ada user yang terdaftar!")
    →

output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")
 kriteria_urutan

```

```

output("\\nUrutan sort?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")
(arah_urutan)

jumlah_pasien ← 0
allocate daftar_pasien with size
database.id_list.jumlah

i traversal [0..database.id_list.jumlah-1]
    if (lower_case(database.id_list.buffer[i].role)
    = "pasien") then
        daftar_pasien[jumlah_pasien] ←
        database.id_list.buffer[i]
        jumlah_pasien ← jumlah_pasien + 1

if (jumlah_pasien = 0) then
    output("Tidak ada pasien yang terdaftar.")
    deallocate daftar_pasien
→

i traversal [0..jumlah_pasien-2]
    j traversal [0..jumlah_pasien-2-i]
        if (kriteria_urutan = 1) then
            if (arah_urutan = 1) then
                perlu_tukar
                (daftar_pasien[j].id >
                 daftar_pasien[j+1].id)
            else
                perlu_tukar
                (daftar_pasien[j].id <
                 daftar_pasien[j+1].id)
        else
            hasil_perbandingan ←
            compare_case_insensitive(daftar_pasien[j].username,
                                       daftar_pasien[j+1].username)
            if (arah_urutan = 1) then
                perlu_tukar ←
                (hasil_perbandingan > 0)
            else
                perlu_tukar ←
                (hasil_perbandingan < 0)

```

```

        if (perlu_tukar) then
            temp ← daftar_pasien[j]
            daftar_pasien[j] ← daftar_pasien[j+1]
            daftar_pasien[j+1] ← temp

output("\\nID | Nama           | Penyakit")
output("-----")
i traversal [0..jumlah_pasien-1]
    pasien_sekarang ← daftar_pasien[i]
    if (pasien_sekarang.riwayat_penyakit.nama = "-"
    or
    length(pasien_sekarang.riwayat_penyakit.nama) =
    0) then
        nama_penyakit ← "-"
    else
        nama_penyakit
        pasien_sekarang.riwayat_penyakit.nama ←

output(pasien_sekarang.id      +      "      |      "
pasien_sekarang.username     +      "      |      "
nama_penyakit)

deallocate daftar_pasien

```

#### 8. F08 - Cari User

```

procedure cariUser(input current_user: User, input
database: ListUser)
{ I.S. User telah login dan mengakses fitur pencarian user }
{ F.S. Menyajikan daftar user dengan role dokter, disusun sesuai preferensi }

```

#### KAMUS LOKAL

sortBy, sortOrder	: <u>integer</u>
dokterList	: <u>array of</u> User
n, i, j	: <u>integer</u>
shouldSwap	: <u>boolean</u>
temp	: User

#### ALGORITMA

```

cariUserHeaderInterface()

if (not isUserValid(current_user)) then
    output("Kamu harus login terlebih dahulu")
    →

```

```

if (current_user.role ≠ "manager") then
    output("Kamu tidak memiliki akses untuk cari
    user")
    →

if (database.id_list.jumlah = 0) then
    output("Tidak ada user yang terdaftar!")
    →

output("Urutkan berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan: ")
input(sortBy)

output("Urutan sort?")
output("1. ASC (A-Z)")
output("2. DESC (Z-A)")
output(">>> Pilihan: ")
input(sortOrder)

n ← 0
allocate dokterList with size database.id_list.jumlah
elemen

i traversal [0..database.id_list.jumlah-1]
    if (database.id_list.buffer[i].role = "dokter")
    then
        dokterList[n] ← database.id_list.buffer[i]
        n ← n + 1

if (n = 0) then
    output("Tidak ada dokter yang terdaftar.")
    deallocation dokterList
    →

i traversal [0..n-2]
    j traversal [0..n-2]
        shouldSwap ← false
        if (sortBy = 1) then
            if (sortOrder = 1) then
                shouldSwap ← (dokterList[j].id
                > dokterList[j + 1].id)
            else

```

```

        shouldSwap ← (dokterList[j].id
                      < dokterList[j + 1].id)

else
    if (sortOrder = 1) then
        shouldSwap ←
            (dokterList[j].username >
             dokterList[j + 1].username)
    else
        shouldSwap ←
            (dokterList[j].username <
             dokterList[j + 1].username)

if (shouldSwap) then
    temp ← dokterList[j]
    dokterList[j] ← dokterList[j + 1]
    dokterList[j + 1] ← temp

output("ID | Nama")
i traversal [0..n-1]
    output(dokterList[i].id + " | " +
              dokterList[i].username)

deallocation dokterList

procedure cariPasien(input current_user: User, input
database: ListUser)
{ I.S. User berada dalam sistem dan ingin mencari informasi pasien }
{ F.S. Menampilkan hasil pencarian pasien berdasarkan kriteria pencarian }

```

#### KAMUS LOKAL

pilihan, pilihan2, sort	: <u>integer</u>
id, jumlahpasien	: <u>integer</u>
nama, nama_penyakit	: <u>string</u>
pasiendicari	: ListUser
temp	: User
i, j, comp	: <u>integer</u>

#### ALGORITMA

```

cariPasienHeaderInterface(current_user)
if (not isValid(current_user)) then
    output("Kamu harus login terlebih dahulu")
    →
if (current_user.role ≠ "manager") then

```

```

output("Kamu tidak memiliki akses untuk cari
pasien")
→
output("Cari pasien berdasarkan?")
output("1. ID")
output("2. Nama")
output("3. Penyakit")
output(">>> Pilihan : ")
if (pilihan = 1) then
    output(">>> masukkan nomor ID pasien: ")
    if (p = null or p.role ≠ "pasien") then
        output("Tidak ditemukan pasien dengan ID "
        + id)
        →
        output("Menampilkan pasien dengan ID " + id)
        output("ID | Nama | Penyakit")
        output(p.id + " | " + p.username + " | " +
        p.riwayat_penyakit.nama)
        →
if (pilihan = 2) then
    output(">>> Masukkan nama : ")
    if (p = null or p.role ≠ "pasien") then
        output("Tidak ditemukan pasien dengan nama
        " + nama)
        →
        output("Menampilkan pasien dengan nama " +
nama)
        output("ID | Nama | Penyakit")
        output(p.id + " | " + p.username + " | " +
p.riwayat_penyakit.nama)
        →
if (pilihan = 3) then
    output("Masukkan nama penyakit : ")
    allocate      pasiendicari      with      size
    database.id_list.jumlah
    i transversal [0..database.id_list.jumlah - 1]

```

```

    if      (database.id_list.buffer[i].role      =
    "pasien"                                and
    database.id_list.buffer[i].riwayat_penyaki
    t.nama = nama_penyakit) then
        pasiendicari.id_list.buffer[jumlahpas
        ien] ← database.id_list.buffer[i]
        jumlahpasien ← jumlahpasien + 1
        pasiendicari.id_list.jumlah           ←
        jumlahpasien

    if (jumlahpasien = 0) then
        output("Tidak ditemukan pasien dengan
        penyakit " + nama_penyakit)
        dealokasiListUser(pasiendicari)
        →
        output("Urutkan berdasarkan?")
        output("1. ID")
        output("2. Nama")
        output(">>> Pilihan : ")
        input(pilihan2)
        output("Urutkan sort ?")
        output("1. ASC (A-Z)")
        output("2. DESC (Z-A)")
        output(">>> Pilihan : ")
        input(sort)
        i transversal [0..jumlahpasien - 2]
            j transversal [0..jumlahpasien - i - 2]
                if (pilihan2 = 1) then
                    comp                         ←
                    pasiendicari.id_list.buffer[j].
                    id                            -
                    pasiendicari.id_list.buffer[j+1
                    ].id
                else
                    comp                         ←
                    compareIgnoreCase(pasiendicari.
                    id_list.buffer[j].username,
                    pasiendicari.id_list.buffer[j+1
                    ].username)
                if ((sort = 1 or comp > 0) or (sort =
                2 or comp < 0)) then
                    temp                         ←
                    pasiendicari.id_list.buffer[j]
                    pasiendicari.id_list.buffer[j]
                    ← pasiendicari.id_list.buffer[j
                    + 1]

```

```

                pasiendicari.id_list.buffer[j +
                1] ← temp
output("Menampilkan pasien dengan penyakit " +
nama_penyakit)
output("ID | Nama | Penyakit")
i transversal [0..jumlahpasien - 1]
    p ← pasiendicari.id_list.buffer[i]
    output(p.id + " | " + p.username + " | " +
p.riwayat_penyakit.nama)
deallocate pasiendicari
→

procedure cariDokter(input current_user: User, input
database: ListUser)
{ I.S. User berada dalam sistem dan ingin mencari informasi dokter }
{ F.S. Menampilkan hasil pencarian dokter berdasarkan ID atau Nama }


```

## KAMUS LOKAL

pilihan	:	<u>integer</u>
id	:	<u>integer</u>
nama	:	<u>string</u>
p	:	User

## ALGORITMA

```

cariDokterHeaderInterface(current_user)
if (not isValid(current_user)) then
    output("Kamu harus login terlebih dahulu")
    →
if (current_user.role ≠ "manager") then
    output("Kamu tidak memiliki akses untuk cari
dokter")
    →
output("Cari dokter berdasarkan?")
output("1. ID")
output("2. Nama")
output(">>> Pilihan : ")
input(pilihan)

if (pilihan = 1) then
    output(">>> masukkan nomor ID dokter : ")
    input(id)
    p ← getUserByID(database, id)
    if (p = null or p.role ≠ "dokter") then


```

```

        output("Tidak ditemukan dokter dengan ID "
        + id)
        →
        output("Menampilkan dokter dengan ID " + id)
        output("ID | Nama")
        output(p.id + " | " + p.username)
        →

if (pilihan = 2) then
    output(">>> Masukkan nama : ")
    input(nama)
    p ← getUserByName(database, nama)
    if (p = null or p.role ≠ "dokter") then
        output("Tidak ditemukan dokter dengan nama
        " + nama)
        →
        output("Menampilkan dokter dengan nama " +
        nama)
        output("ID | Nama")
        output(p.id + " | " + p.username)
        →

```

#### 9. F09 - Lihat Antrian

```

procedure lihatAntrean(input current_user: User, input
denah: MatriksRuangan)
{ Menampilkan denah rumah sakit dan detail ruangan yang tidak kosong, hanya
untuk manager }

```

#### KAMUS LOKAL

i, j : integer

#### ALGORITMA

```

lihatAntreanHeaderInterface()
if (lower_case(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk fitur
    lihat antrean!")
    →
    output(">>> LIHAT ANTREAN")
    output("*** DENAH RUMAH SAKIT ***")
    lihatDenah(current_user, denah)
    output("*** DETAIL RUANGAN YANG TIDAK KOSONG ***")

i traversal [0..denah.row-1]
    j traversal [0..denah.column-1]

```

```

    if (isValidUser(denah.ruang[i][j].dokter))
    then
        printInfoRuang(denah, i, j)

```

#### 10. F10 - Tambah Dokter

```

procedure tambahDokter(input current_user: User,
input/output database: ListUser)
{ I.S. Seorang manajer ingin menambahkan dokter ke dalam sistem }
{ F.S. Dokter baru berhasil ditambahkan ke dalam database jika input valid }

```

#### KAMUS LOKAL

```

nama      : string
password  : string
i         : integer
new_user  : User
valid     : boolean

```

#### ALGORITMA

```

tambahDokterHeaderInterface()

if (lower(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk fitur
    tambah dokter!")
    →

do
    valid ← TRUE
    output("Username: ")
    input(nama)
    i ← 0
    while (i < strlen(nama) and valid) do
        if ((nama[i] < 'A' or nama[i] > 'Z') and
            (nama[i] < 'a' or nama[i] > 'z')) then
            output("Registrasi gagal! Username
            hanya boleh berisi huruf (tanpa angka
            atau simbol).")
            valid ← FALSE
        else
            i ← i + 1

        if (isUsernameInSet(database, nama)) then
            output("User " + nama + " sudah
            terdaftar.")
            output("Daftar dengan username lain")
            output("atau")

```

```

        output("gunakan command LUPA_PASSWORD jika
        kamu lupa password akunmu.")
        valid ← FALSE
    while (not(valid))

        output("Password: ")
        input(password)

        createUser(new_user)

        if (database.jumlah > 0) then
            new_user.id ←
            database.id_list.buffer[database.jumlah - 1].id
            +
            1
        else
            new_user.id ← 1

        new_user.username ← nama
        new_user.password ← password
        new_user.role ← "dokter"

        insertUser(database, new_user)
        output("Dokter " + nama + " berhasil ditambahkan
        dengan ID " + new_user.id + "!")

```

### 11. F11 - Diagnosis

```

procedure diagnosis(input current_user: User, input
ensiklopedia: ListPenyakit, input antrean : Queue)
{ I.S. User dokter sudah login dan antrian pasien tersedia }
{ F.S. Pasien pada antrian paling depan didiagnosis dan riwayat penyakit
diperbarui jika sesuai }

```

### KAMUS LOKAL

<u>pasien_diagnosis</u>	:	Patient
<u>penyakit_terpilih</u>	:	<u>string</u>
i	:	<u>integer</u>
<u>diagnosis_berhasil</u>	:	<u>boolean</u>

### ALGORITMA

```

if (current_user.role ≠ "dokter") then
    output("Hanya dokter yang dapat melakukan
    diagnosis!")
    →

```

```

if
(isQueueEmpty(denah_ruang.ruang[idxI][idxJ].antrean))
then
    output("Tidak ada pasien dalam antrean.")
    →

pasien_diagnosis ← antrean.peek()
output("Mulai diagnosis untuk pasien: " +
pasien_diagnosis.nama)
diagnosis_berhasil ← false

i traversal [0..ensiklopedia.length-1]
    output("Apakah pasien menunjukkan gejala " +
ensiklopedia[i].nama_penyakit + "? (ya/tidak)")
    input(penyakit_terpilih)
    if (penyakit_terpilih = "ya") then
        pasien_diagnosis.riwayat_penyakit.tambah(e
nsiklopedia[i].nama_penyakit)
        diagnosis_berhasil ← true
        break

if (diagnosis_berhasil) then
    output("Diagnosis selesai: pasien didiagnosis
menderita " + ensiklopedia[i].nama_penyakit)
else
    output("Diagnosis tidak menemukan penyakit yang
sesuai.")

```

## 12. F12 - Ngobatin

```

procedure ngobatin(input current_user: user, input/output
pasien: user, input obat_penyakit: MapObatPenyakit)
{ I.S. current_user sudah login, pasien sudah terpilih, data obat_penyakit
tersedia }
{ F.S. Jika current_user dokter dan pasien sudah diagnosa, obat ditambahkan ke
inventory pasien }

```

## KAMUS LOKAL

i, j, k, p, m	: <u>integer</u>
idxI, idxJ	: <u>integer</u>
pasien	: User
obat	: ListObat
found	: <u>boolean</u>
hadMedications	: <u>boolean</u>
userDuluTidakAdaObat	: <u>boolean</u>
jumlah_obat	: <u>integer</u>

```
userByID, userByName : pointer to user
```

## ALGORITMA

```
idxI, idxJ ← -1
output(">>> NGOBATIN")

if (current_user.role ≠ "dokter") then
    output("Hanya dokter yang dapat memberikan obat!")
    →

{ Cari lokasi ruangan dokter }
i ← 0
while (i < denah_ruang.row and idxI = -1) do
    j ← 0
    while (j < denah_ruang.column) do
        if (denah_ruang.ruang[i][j].dokter = current_user) then
            idxI ← i
            idxJ ← j
            j ← denah_ruang.column { untuk keluar dari inner loop }
        else
            j ← j + 1
    i ← i + 1

if (idxI = -1 or idxJ = -1) then
    output("Dokter tidak ditemukan di ruangan manapun!")
    →

if (isQueueEmpty(ruang[idxI][idxJ].antrean)) then
    output("Tidak ada pasien dalam antrean!")
    →

pasien ← HEAD(ruang[idxI][idxJ].antrean)

if (pasien.riwayat_penyakit.nama = "") then
    output("Pasien " + pasien.username + " belum terdiagnosis!")
    →

if (pasien.isGetObat = true) then
    output("Pasien telah mendapat obat")
    →
```

```

found ← false

{ Cari obat untuk penyakit pasien }
p traversal [0..obat_penyakit.length-1]
    if           (pasien.riwayat_penyakit.nama      =
    obat_penyakit.data[p].key) then
        found ← true
        output("Pasien " + pasien.username + "
terdiagnosis menderita penyakit " +
pasien.riwayat_penyakit.nama)
        output("Obat yang harus diberikan:")

        obat ← obat_penyakit.data[p].value
        hadMedications ← (pasien.inventory.jumlah
> 0)
        userDuluTidakAdaObat ← not hadMedications
        pasien.inventory.jumlah ← 0

        if (hadMedications = TRUE) then
            CountUserDenganObat ←
            CountUserDenganObat - 1

        jumlah_obat ← min(obat.jumlah, MAX_OBAT)
        m traversal [0..jumlah_obat-1]
            output("          (ID:          "      +
            obat.data[m].obat_id      +      ")      "      +
            obat.data[m].nama_obat)
            pasien.inventory.data[m] ←
            obat.data[m]

        pasien.inventory.jumlah ← jumlah_obat

        if (userDuluTidakAdaObat and jumlah_obat >
0) then
            CountUserDenganObat ←
            CountUserDenganObat + 1

{ Sinkronisasi ke database (dua buffer) }
userByID      ←      getUserByID(database,
pasien.id)
userByName     ←      getUserByName(database,
pasien.username)

if (userByID ≠ null) then

```

```

        userByID.inventory ← pasien.inventory

        if (userByName ≠ null) then
            userByName.inventory ←
            pasien.inventory

            output("Semua obat berhasil diberikan ke
            inventory pasien " + pasien.username)
            pasien.isGetObat ← TRUE
            break

        if (not found) then
            output("Tidak ada obat yang tersedia untuk
            penyakit " + pasien.riwayat_penyakit.nama + "
            dalam database!")

```

### 13. F13 - Aku boleh pulang ga, dok?

```

procedure bolehPulang(input/output current_user: User,
input map_obat_penyakit: MapObatPenyakit, input/output
database_pengguna: ListUser, input/output rumah_sakit:
MatriksRuang)
{ I.S. current_user sudah login sebagai pasien dan data rumah sakit, obat,
penyakit, serta database pengguna terisi }
{ F.S. Jika pasien sudah memenuhi syarat, pasien dihapus dari antrean dan
statusnya di-reset }

```

### KAMUS LOKAL

resep	: ListObat
found	: <u>boolean</u>
n, i, idxSalah	: <u>integer</u>
urutanBenar	: <u>boolean</u>

### ALGORITMA

```

bolehPulangHeaderInterface()

if (lowercase(current_user.role) ≠ "pasien") then
    output("Hanya pasien yang dapat mengakses fitur
    ini!")
    →

if (current_user.riwayat_penyakit.nama = "-" or
strlen(current_user.riwayat_penyakit.nama) = 0)
then
    output("Kamu belum menerima diagnosis apapun
    dari dokter, jangan buru-buru pulang!")

```

```

→

found ← false
i traversal [0..map_obat_penyakit.length-1] and not
(found) do
    if (map_obat_penyakit.data[i].key = current_user.riwayat_penyakit.nama) then
        resep ← map_obat_penyakit.data[i].value
        found ← true

if (not (found)) then
    output("Data penyakit " + current_user.riwayat_penyakit.nama + " tidak ditemukan dalam database obat.")
→

output("Dokter sedang memeriksa keadaanmu...")

n ← current_user.perut.top + 1
urutanBenar ← true
idxSalah ← -1

for ((i ← 0 to n - 1) and (i < resep.jumlah)) do
    if (current_user.perut.data[i].nama_obat ≠ resep.data[i].nama_obat) then
        urutanBenar ← false
        idxSalah ← i
        break

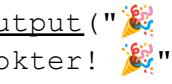
if (not urutanBenar or n > resep.jumlah) then
    output("Maaf, tapi kamu masih belum bisa pulang!")
    output("Urutan peminuman obat yang diharapkan:")
    i traversal [0..resep.jumlah-1]
        output(resep.data[i].nama_obat + (if i ≠ resep.jumlah - 1 then " → " else "\n"))
output("Urutan obat yang kamu minum:")
    i traversal [0..n-1]
        if (i = idxSalah and idxSalah ≠ -1) then
            output("<<<Nama obat ke-" + i + " dalam warna merah>>>")
        else
            output(current_user.perut.data[i].nama_obat)

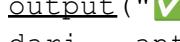
```

```

        if (i ≠ n - 1) then
            output(" -> ")
output("Silahkan kunjungi dokter untuk meminta
penawar yang sesuai!")
output("Gunakan perintah: MINUM_PENAWAR")
→

if (n ≠ resep.jumlah) then
    output("Masih ada obat yang belum kamu
habiskan, minum semuanya dulu yukk!")
→

output(" Selamat! Kamu sudah dinyatakan sembuh oleh
dokter! ")
output("Silahkan pulang dan semoga sehat selalu!")

keluar ← FALSE
i ← 0
while (i < rumah_sakit.row and not keluar) do
    j ← 0
    while (j < rumah_sakit.column and not keluar)
do
    if isRuangEmpty(rumah_sakit.ruang[i][j]) then
        if isQueueEmpty(rumah_sakit.ruang[i][j].antrean) and
            isSameUser(rumah_sakit.ruang[i][j].antrean.pasien, current_user) then
                dequeue(rumah_sakit.ruang[i][j].antrean)
                output(" Kamu telah dihapus
dari antrean ruangan " +
(char)('A' + i) + (j + 1))
                keluar ← TRUE
            j ← j + 1
        i ← i + 1

strcpy(current_user.riwayat_penyakit.nama, "-")
current_user.riwayat_penyakit.id ← -1
createListObat(current_user.inventory)
createStackObat(current_user.perut)

```

```

user_id           ←      getUserByID(database_pengguna,
current_user.id)
if (user_id ≠ NULL) then
    strcpy(user_id.riwayat_penyakit.nama, "-")
    user_id.riwayat_penyakit.id ← -1
    createListObat(user_id.inventory)
    createStackObat(user_id.perut)

user_name         ←      getUserByName(database_pengguna,
current_user.username)
if (user_name ≠ NULL) then
    strcpy(user_name.riwayat_penyakit.nama, "-")
    user_name.riwayat_penyakit.id ← -1
    createListObat(user_name.inventory)
    createStackObat(user_name.perut)

```

#### 14. F14 - Daftar Check-Up

```

procedure daftarCheckUp(input/output current_user: User,
input/output database: ListUser, input/output
rumah_sakit: MatriksRuangan)
{ I.S. current_user sudah login, database dan rumah_sakit sudah tersedia }
{ F.S. Jika current_user role pasien, data kesehatan disimpan dan pasien
terdaftar ke antrean dokter }

```

#### KAMUS LOKAL

pointer	: *User
suhu_tubuh	: real
tekanan_darah_sistolik,	: integer
tekanan_darah_diastolik	: integer
detak_jantung	: integer
saturasi_oksigen	: real
kadar_gula_darah, tinggi_badan	: integer
kadar_kolesterol, trombosit	: integer
berat_badan	: real
drcount, pilihan, count, row, col	: integer
current	: Ruangan
foundPointer	: boolean
baris, kolom	: integer

#### ALGORITMA

```

output(">>> DAFTAR CHECK UP")

{ Validasi role pasien }
if (lowercase(current_user.role) ≠ "pasien") then
    output("Kamu tidak terdaftar sebagai pasien!")

```

```

output("LOGIN sebagai pasien untuk daftar
checkup!")
→

if (isUserInMatriks(current_user, rumah_sakit)) then
    output("Kamu sudah terdaftar dalam antrean!")
    output("Lihat ANTREAN untuk informasi lebih
lengkap.")
    output("Selesaikan checkup sebelum mendaftar
lagi ya!")
→

pointer ← NULL
foundPointer ← false
i traversal [0..database.jumlah-1]
    if (isSameUser(current_user, database.data[i])
    and not(foundPointer)) then
        pointer ← alamat database.data[i]
        foundPointer ← true

{ Input data kesehatan }
output("Silakan masukkan data kesehatan anda")

do
    output("Suhu tubuh (celcius): ")
    input(suhu_tubuh)
    if (suhu_tubuh <= 0) then
        output("Suhu tubuh harus berupa angka
positif!")
while (suhu_tubuh <= 0)
pointer.suhu_tubuh ← suhu_tubuh
current_user.suhu_tubuh ← suhu_tubuh

do
    output("Tekanan darah (sistol/diastol, misal
120 80): ")
    input(tekanan_darah_sistolik,
tekanan_darah_diastolik)
    if (tekanan_darah_sistolik      <=      0      or
tekanan_darah_diastolik <= 0) then
        output("Tekanan darah harus berupa angka
positif!")
while      (tekanan_darah_sistolik      <=      0      or
tekanan_darah_diastolik <= 0)

```

```

pointer.tekanan_darah_sistolik ←
tekanan_darah_sistolik
pointer.tekanan_darah_diastolik ←
tekanan_darah_diastolik
current_user.tekanan_darah_sistolik ←
tekanan_darah_sistolik
current_user.tekanan_darah_diastolik ←
tekanan_darah_diastolik

do
    output("Detak jantung (bpm): ")
    input(detak_jantung)
    if (detak_jantung <= 0) then
        output("Detak jantung harus berupa angka
positif!")
while (detak_jantung <= 0)
pointer.detak_jantung ← detak_jantung
current_user.detak_jantung ← detak_jantung

do
    output("Saturasi oksigen (%): ")
    input(saturasi_oksigen)
    if (saturasi_oksigen <= 0 or saturasi_oksigen >
100) then
        output("Saturasi oksigen harus berupa
angka positif ≤ 100!")
while (saturasi_oksigen <= 0 or saturasi_oksigen >
100)
pointer.saturasi_oksigen ← saturasi_oksigen
current_user.saturasi_oksigen ← saturasi_oksigen

do
    output("Kadar gula darah (mg/dL): ")
    input(kadar_gula_darah)
    if (kadar_gula_darah <= 0) then
        output("Kadar gula darah harus berupa
angka positif!")
while (kadar_gula_darah <= 0)
pointer.kadar_gula_darah ← kadar_gula_darah
current_user.kadar_gula_darah ← kadar_gula_darah

do
    output("Berat badan (kg): ")
    input(berat_badan)
    if (berat_badan <= 0) then

```

```

        output("Berat badan harus berupa angka
        positif!")
while (berat_badan <= 0)
pointer.berat_badan ← berat_badan
current_user.berat_badan ← berat_badan

do
    output("Tinggi badan (cm) : ")
    input(tinggi_badan)
    if (tinggi_badan <= 0) then
        output("Tinggi badan harus berupa angka
        positif!")
while (tinggi_badan <= 0)
pointer.tinggi_badan ← tinggi_badan
current_user.tinggi_badan ← tinggi_badan

do
    output("Kadar kolesterol (mg/dL) : ")
    input(kadar_kolesterol)
    if (kadar_kolesterol <= 0) then
        output("Kadar kolesterol harus berupa
        angka positif!")
while (kadar_kolesterol <= 0)
pointer.kadar_kolesterol ← kadar_kolesterol
current_user.kadar_kolesterol ← kadar_kolesterol

do
    output("Trombosit (ribu/uL) : ")
    input(trombosit)
    if (trombosit <= 0) then
        output("Trombosit harus berupa angka
        positif!")
while (trombosit <= 0)
pointer.trombosit ← trombosit
current_user.trombosit ← trombosit

{ Tampilkan daftar dokter }
output("Berikut adalah daftar dokter yang tersedia:")
drcount ← 0
allocate price with size rumah_sakit.row *
rumah_sakit.column
allocate rooms with size rumah_sakit.row *
rumah_sakit.column
row traversal [0..rumah_sakit.row-1]
    col traversal [0..rumah_sakit.col-1]

```

```

        current ← rumah_sakit.ruang[row][col]
        if      (not    isRuanganEmpty(current)    and
queueLength(current.antrean) < max_queue)
then
            drccount ← drccount + 1
            rooms[drccount - 1] := address of
            rumah_sakit.ruang[row][col]
            output(drccount,      ".      Dr.      ",
            current.dokter.username, " - Ruangan
            ", char('A' + row), col + 1)
            output("                  Aura:      ",
            current.dokter aura)

            if      (isQueueEmpty(current.antrean))
then
            output("      Pasien di ruangan:
            -")
            output("      Pasien di antrean:
            -")
            else if (queueLength(current.antrean)
            > current.kapasitas) then
            output("      Pasien di ruangan:
            ", current.kapasitas)
            output("      Pasien di antrean:
            ", queueLength(current.antrean)
            - current.kapasitas)
            else
            output("      Pasien di ruangan:
            ",
            queueLength(current.antrean))
            output("      Pasien di antrean:
            -")

            if (current.dokter.aura < 0) then
                price[drccount - 1] ← 0
            else
                price[drccount - 1] ← (3 * 
                current.dokter.aura / 2) + 1
                output("      Biaya checkup:      ",
                price[drccount - 1])

if (drccount > 0) then
    while (forever) do
        output("Pilih dokter yang tersedia (1-",
        drccount, ", 0 untuk batal): ")

```

```

input(pilihan)
if (pilihan >= 0 and pilihan <= drcount)
then
    if (pilihan = 0) then
        output("Pendaftaran           checkup
Anda dibatalkan.")

resetDataKesehatan(current_user)
    resetDataKesehatan(pointerID)
    resetDataKesehatan(pointerName)
    →
    else if (current_user.banarich <
price[pilihan - 1]) then
        output("Maaf,     banarich     kamu
tidak cukup.")
    else
        break
else
    output("Pilihan tidak valid")
    clear input buffer

pointerID.banarich ← pointerID.banarich -
price[pilihan - 1]
pointerName.banarich ← pointerName.banarich -
price[pilihan - 1]
current_user.banarich ← current_user.banarich -
price[pilihan - 1]
jatah_RS ← price[pilihan - 1] / 5

if (jatah_RS = 0) then
    jatah_RS ← 1
keuangan ← keuangan + jatah_RS
jatah_dokter ← price[pilihan - 1] - jatah_RS
dokter_in_list ← getUserByID(database,
rooms[pilihan - 1].dokter.id)
dokter_in_set ← getUserByName(database,
rooms[pilihan - 1].dokter.username)
dokter_in_list.banarich ←
dokter_in_list.banarich + jatah_dokter
dokter_in_set.banarich ← dokter_in_set.banarich +
jatah_dokter
offset ← (address of rooms[pilihan - 1] -
address of rumah_sakit) / size of Ruangan
baris ← offset div MAX_COLS
kolom := offset mod MAX_COLS

```

```

        enqueue(rooms[pilihan - 1].antrean,
current_user)
        output("Pendaftaran berhasil!")
        output("Anda terdaftar antrean checkup dengan
dr. ", rooms[pilihan - 1].dokter.username, " di
ruangan ", char(65 + baris), kolom + 1)
        if (queueLength(rooms[pilihan - 1].antrean) ≤
rooms[pilihan - 1].kapasitas) then
            output("Anda bisa langsung masuk ke
ruangan")
        else
            output("Nomor antrean anda: ",
queueLength(rooms[pilihan - 1].antrean) -
rooms[pilihan - 1].kapasitas)

        free(price)
        free(rooms)
    else
        output("Maaf, tidak ada dokter yang tersedia
saat ini.")

```

### 15. F15 - Antrean Saya

```

procedure antreanSaya(input current_user: User, input
rumah_sakit: MatriksRuangan)
{I.S. Pengguna terdapat dalam sistem dan ingin mengetahui status antrean
check-up }
{F.S. Menampilkan status antrean atau informasi jika belum terdaftar }

```

### KAMUS LOKAL

```

i, j, pos, total      : integer
found                 : boolean
ruangan               : Ruangan
q                      : Address

```

### ALGORITMA

```

antreanSayaHeaderInterface()

if (current_user.role ≠ "pasien") then
    output("Fitur ini hanya bisa diakses oleh
pasien.")
    →

found ← FALSE
i ← 0

```

```

while (i < rumah_sakit.row and not found) do
    j ← 0
    while (j < rumah_sakit.column and not found) do
        ruangan ← rumah_sakit.ruang[i][j]

        if (not (isRuanganEmpty(ruangan))) then
            q ← ruangan.antrean
            pos ← 1
            total ← queueLength(ruangan.antrean)

            while (q ≠ null) do
                if (isSameUser(q.pasien,
                    current_user)) then
                    if (pos ≤
                        ruangan.kapasitas) then
                        output("Anda sedang
                            berada di dalam
                            ruangan dokter!")
                    else
                        output("Status
                            antrian Anda:")
                        output("Dokter: Dr. "
                            +
                            ruangan.dokter.username)
                        output("Ruang: " +
                            chr('A' + i) + (j +
                            1))
                        output("Posisi
                            antrian: " + pos + "
                            dari " + total)
                found ← TRUE
                break
                q ← q.next
                pos ← pos + 1
            j ← j + 1
            i ← i + 1

            if (not(found)) then
                output("Anda belum terdaftar dalam antrian
                    check-up!")
                output("Silakan daftar terlebih dahulu dengan
                    command DAFTAR_CHECKUP.")

```

## 16. F16 - Minum Obat

```

procedure minumObat(input/output current_user: User,
input/output database: ListUser)
{ I.S. User sedang login dan memiliki obat di inventaris }
{ F.S. Salah satu obat berhasil diminum dan dimasukkan ke perut }

```

## KAMUS LOKAL

```

pilihan : integer
obat_pilihan : Obat

```

## ALGORITMA

```

minumObatHeaderInterface()

if (not isUserValid(current_user)) then
    output("Kamu tidak memiliki akses untuk minum
    obat!")
    →

if (lower(current_user.role) = "manager") then
    output("Manajer kok minum obat? Kalo stres
    bilang...")
    →

if (lower(current_user.role) = "dokter") then
    output("Anda kan yang ngasih obat")
    →

if (current_user.inventory.jumlah = 0) then
    output("Obat abis bang")
    →

output("===== DAFTAR OBAT =====")
i traversal [0..current_user.inventory.jumlah-1]
    output(i+1 + ". " +
    current_user.inventory.data[i].nama_obat)

output(">>> Pilih obat untuk diminum : ")
input(pilihan)

if (pilihan < 1 or pilihan >
current_user.inventory.jumlah) then
    output("Pilihan tidak tersedia!")
    →

```

```

obat_pilihan ← current_user.inventory.data[pilihan - 1]
pushStack(current_user.perut, obat_pilihan)
output("GULUKGULUKGULUK... " + obat_pilihan.nama_obat
+ " berhasil diminum!!!")

    i                                traversal
[pilihan-1..current_user.inventory.jumlah-2]
    current_user.inventory.data[i]           ←
    current_user.inventory.data[i+1]          ←
    current_user.inventory.jumlah           ←
    current_user.inventory.jumlah - 1

if (current_user.inventory.jumlah = 0) then
    CountUserDenganObat ← CountUserDenganObat - 1

userByID ← getUserByID(database, current_user.id)
userByName           ←           getUserByName(database,
current_user.username)

if (userByID ≠ null) then
    userByID.inventory ← current_user.inventory
    userByID.perut ← current_user.perut

if (userByName ≠ null) then
    userByName.inventory ← current_user.inventory
    userByName.perut ← current_user.perut

```

### 17. F17 - Minum Penawar

```

procedure minumPenawar(input/output pasien: User; input
                      map_obat penyakit: MapObatPenyakit; input/output
                      database_pengguna: ListUser; input/output rumah_sakit:
                      MatriksRuang)

```

{ I.S. Stack perut milik pasien mungkin berisi obat, dan pasien sudah memiliki diagnosa penyakit }

{ F.S. Jika urutan obat salah, satu obat terakhir dikeluarkan dari perut, nyawa pasien berkurang, dan pasien bisa meninggal }

### KAMUS LOKAL

resep	: ListObat
ditemukan	: <u>boolean</u>
urutanBenar	: <u>boolean</u>
keluar	: <u>boolean</u>
n, i, j	: <u>integer</u>
terakhir	: Obat

```

username_temp    : string

ALGORITMA
minumPenawarHeaderInterface()

if (isEmpty(pasien.perut)) then
output("Perut kosong!! Belum ada obat yang
dimakan.")
→

if ((pasien.riwayat_penyakit.nama = "-") or
(length(pasien.riwayat_penyakit.nama) = 0)) then
output("Kamu belum menerima diagnosis apapun
dari dokter!")
output("Tanpa resep dokter, tidak bisa
menggunakan penawar dengan aman.")
→

ditemukan ← false
i ← 0
while (i < map_obat_penyakit.length) and (not
ditemukan) do
if (map_obat_penyakit.data[i].key =
pasien.riwayat_penyakit.nama) then
resep ← map_obat_penyakit.data[i].value
ditemukan ← true
i ← i + 1

if (not ditemukan) then
output("Data penyakit " +
pasien.riwayat_penyakit.nama + " tidak
ditemukan dalam database obat.")
→

output("Dokter akan membantu kamu menggunakan penawar
untuk mengeluarkan obat... ! ! ! ")
n ← pasien.perut.top + 1
urutanBenar ← true
i ← 0

while ((i < n) and (i < resep.jumlah)) and
urutanBenar do
if (pasien.perut.data[i].nama_obat ≠
resep.data[i].nama_obat) then

```

```

        urutanBenar ← false
        i ← i + 1

if (not urutanBenar) or (n > resep.jumlah) then
    if (not isStackEmpty(pasien.perut)) then
        terakhir ← popStack(pasien.perut)
        pasien.nyawa ← pasien.nyawa - 1

    if (pasien.nyawa ≤ 0) then
        output("💀 Anda Meninggal 💀")
        username_temp ← pasien.username

if pasien.inventory.jumlah > 0 then
    CountUserDenganObat ←
    CountUserDenganObat - 1

        keluar ← false
        i ← 0
while (i < rumah_sakit.row) and (not
        keluar) do
            j ← 0
while (j < rumah_sakit.column)
        and (not keluar) do
            if
                (!isRuanganEmpty(rumah_sak
                    it.ruang[i][j])) then
                    if
                        (!isQueueEmpty(rumah_
                            sakit.ruang[i][j].ant
                            rean)) and
                        (isSameUser(rumah_sak
                            it.ruang[i][j].antrea
                            n.pasien, pasien))
                    then
                        dequeue(rumah_s
                            akit.ruang[i][j]
                            ].antrean)

            dokter ←
            rumah_sakit.rua
            ng[i][j].dokter

            pointer_id ←
            getUserByID(dat

```

```
abase_pengguna,  
dokter.id)
```

```
pointer_name ←  
getUserByName(d  
atabase_penggun  
a,  
dokter.username
```

```
pointer_id.aura  
←  
pointer_id.aura  
- 1
```

```
pointer_name.au  
ra ←  
pointer_name.au  
ra - 1
```

```
dokter.aura ←  
dokter.aura - 1
```

```
output("dr. " +  
dokter.username  
+ " mengalami  
penurunan aura  
karena  
malpraktik")
```

```
output("📝  
Pasien dihapus  
dari antrian  
ruangan " +  
char('A' + i) +  
(j + 1) + ".")  
keluar ← true  
j ← j + 1  
i ← i + 1
```

```
user_id ←  
getUserByID(database_pengguna,  
pasien.id)  
user_name ←  
getUserByName(database_pengguna,  
pasien.username)
```

```

        if (user_id ≠ null) then
            user_id.nyawa ← 0
            user_id.riwayat_penyakit.nama ←
                "Meninggal"

        if (user_name ≠ null) then
            user_name.nyawa ← 0
            user_name.riwayat_penyakit.nama
                ← "Meninggal"

matiInterface(username_temp)
destroyUser(pasien)
→

output("⚠️ Penawar bekerja! Mengeluarkan
        satu obat teratas dari perutmu...")
output("Uwekkk!!! " + terakhir.nama_obat +
        " telah dikeluarkan dari perutmu.")

if (not isFullObat(pasien.inventory)) then
    if pasien.inventory.jumlah = 0 then
        CountUserDenganObat ←
        CountUserDenganObat + 1
        pasien.isGetObat ← true

        insertObatSesuaiResep(pasien.inventory,
            terakhir, resep)
        output("➡️ Obat " +
            terakhir.nama_obat + " dikembalikan
            ke inventory sesuai urutan resep.")
    else
        output("⚠️ Inventory penuh! Obat " +
            terakhir.nama_obat + " tidak bisa
            dikembalikan.")

user_id ← getUserByID(database_pengguna,
    pasien.id)
user_name ← getUserByName(database_pengguna,
    pasien.username)

if (user_id ≠ null) then
    user_id.perut ← pasien.perut
    user_id.inventory ← pasien.inventory

```

```

        if (user_name ≠ null) then
            user_name.perut ← pasien.perut
            user_name.inventory ← pasien.inventory

            output("＼n💡 Penawar berhasil digunakan! Coba
                    minum obat sesuai urutan resep.")
            output("Sisa nyawa kamu: " + pasien.nyawa)
            else
                output("✅ Obat yang kamu minum sudah sesuai
                        resep dokter!")
            output("Tidak perlu menggunakan penawar. Kamu
                    bisa mencoba BOLEH_PULANG untuk pulang.")

```

#### 18. F18 - Exit

```

function exitRumahSakit(var current_user: User, var
denah: MatriksRuangan, var database: ListUser,
daftar_penyakit: ListPenyakit, daftar_obat: ListObat,
map: MapObatPenyakit) → boolean
{ Menghentikan aplikasi rumah sakit dan membersihkan data, hanya jika tidak
ada user yang sedang login }

```

#### KAMUS LOKAL

c, opt : char

#### ALGORITMA

```

exitHeaderInterface()

if (isValid(current_user)) then
    output("Rumah sakit hanya dapat dihancurkan
            jika tidak ada pengguna yang mengakses!")
    output("Silakan LOGOUT untuk EXIT")
    → false

fflush(stdin)
repeat
    output("Mau save perubahan? (Y/N)")
    input(opt)
    while (input buffer tidak kosong) do
        buang karakter dari input buffer

        opt ← uppercase(opt)
until (opt = 'Y' or opt = 'N')

if (opt = 'Y') then

```

```

        saveData(&rumah_sakit,
                  &database_user,
                  database_penyakit,
                  map_obat_penyakit)

destroyUser(current_user)
destroyListUser(&database_user)
destroyMatriksRuang(&rumah_sakit)

output("Terima kasih telah menggunakan aplikasi Rumah
Sakit Nimons!")
output("Sampai jumpa lagi!")

→ true

```

#### 19. main

```

function main(input: argv :string) → integer
{ Fungsi ini menjalankan proses inisialisasi sistem, memuat data dari file
eksternal, menerima input perintah dari pengguna, serta mengelola eksekusi
perintah sesuai hak akses dan status pengguna. Prosedur akan terus berjalan
hingga pengguna memberikan perintah "EXIT" yang valid }

```

#### KAMUS LOKAL

current_user	:	User
folderName	:	<u>string</u>
command	:	<u>string</u>
c	:	<u>integer</u>
selesai	:	<u>boolean</u>

#### ALGORITMA

```

selesai ← false
SelamatDatang()

if (stren(argv) < 2) then
    output("Tidak ada nama folder yang diberikan!")
    output("Usage : ./main <<nama_folder>>")
    → ExitFailure

folderName ← argv[1]

if not(folderExists(folderName)) then
    output("Folder \" + folderName + "\" tidak
ditemukan.")
    →

output("Loading...")

```

```

createUser(current_user)
loadData(folderName,                                     database_user,
database_penyakit, database_obat, map_obat_penyakit,
rumah_sakit, max_queue)

while not(selesai) do
    command ← ""
    output(">>> ")
    input(command)
    while ((c = getchar()) != '\n' and c != EOF) do
        { Empty loop body }

    if (command = "EXIT") then
        if           (exitRumahSakit(current_user,
                                         rumah_sakit,                 database_user,
                                         database_penyakit,            database_obat,
                                         map_obat_penyakit)) then
            break
    else if (command = "LOGIN") then
        login(current_user, database_user)
    else if (command = "REGISTER") then
        registerPasien(current_user,
                        database_user)
    else if (command = "LOGOUT") then
        logout(current_user)
    else if (command = "LUPA_PASSWORD") then
        lupaPassword(current_user, database_user)
    else if (command = "HELP") then
        help(current_user)
    else if (command = "LIHAT_DENAH") then
        lihatDenah(current_user, rumah_sakit)
    else if (command = "LIHAT_RUANGAN") then
        lihatRuangan(current_user, rumah_sakit)
    else if (command = "LIHAT_ANTREAN") then
        lihatAntrean(current_user, rumah_sakit)
    else if (command = "TAMBAH_DOKTER") then
        tambahDokter(current_user, database_user)
    else if (command = "ASSIGN_DOKTER") then
        assignDokter(current_user,   database_user,
                      rumah_sakit)
    else if (command = "DAFTAR_CHECKUP") then
        daftarCheckUp(current_user, database_user,
                       rumah_sakit)
    else if (command = "DIAGNOSIS") then

```

```

        diagnosis(current_user, database_penyakit,
        rumah_sakit, database_user)
    else if (command = "NGOBATIN") then
        ngobatin(current_user, rumah_sakit,
        map_obat_penyakit, database_user)
    else if (command = "ANTREAN_SAYA") then
        antreanSaya(current_user, rumah_sakit)
    else if (command = "MINUM_OBAT") then
        minumObat(current_user)
    else if (command = "MINUM_PENAWAR") then
        minumPenawar(current_user)
    else
        output("Command invalid!")
        output("Masukkan \"HELP\" jika anda tidak
        tahu harus apa:)")

```

## 20. Skip Antrian

```

procedure skipAntrean(input/output current_user: User,
input/output rumah_sakit: MatriksRuangan)
{ I.S. Pasien sudah login dan ingin mempercepat antreannya ke antrean terdepan
}
{ F.S. Jika valid, posisi antrean pasien dipindah ke antrean terdepan (setelah
pasien dalam ruangan) }

```

### KAMUS LOKAL

i, j	: <u>integer</u>
idxI, idxJ	: <u>integer</u> <- -1
idxPasien	: <u>integer</u>
found	: <u>boolean</u>
current	: Ruangan
q	: Address
userDalam	: <u>array</u> [0..MAX_USER] <u>of</u> User
antreanUsers	: <u>array</u> [0..MAX_USER] <u>of</u> User
sisaUsers	: <u>array</u> [0..MAX_USER] <u>of</u> User
tempCount	: <u>integer</u>
antreanCount	: <u>integer</u>
sisaCount	: <u>integer</u>
prioritas	: User

### ALGORITMA

```

skipAntreanHeaderInterface()

idxI ← -1
idxJ ← -1
found ← FALSE

```

```

if (current_user.role ≠ "pasien") then
    output("Hanya pasien yang dapat mengakses fitur
ini!")
    →

i ← 0
while (i < rumah_sakit.row and not found) do
    j ← 0
    while (j < rumah_sakit.column and not found) do
        current ← rumah_sakit.ruang[i][j]
        if      (not(isRuanganEmpty(current))      and
        not(isQueueEmpty(current.antrean))) then
            q ← current.antrean
            while (q ≠ NULL) do
                if          (isSameUser(q.pasien,
                current_user)) then
                    idxI ← i
                    idxJ ← j
                    found ← TRUE
                    break
                q ← q.next
            j ← j + 1
            i ← i + 1

if (not found or idxI = -1 or idxJ = -1) then
    output("Anda tidak terdaftar dalam antrean!")
    →

idxPasien                                ←
findInQueue(rumah_sakit.ruang[idxI][idxJ].antrean,
current_user)
if (idxPasien = -1) then
    output("Anda tidak terdaftar dalam antrean!")
    →

if                               (idxPasien           <
rumah_sakit.ruang[idxI][idxJ].kapasitas) then
    output("Anda sudah berada di dalam ruangan,
tidak bisa skip antrean!")
    →

if                               (idxPasien           =
rumah_sakit.ruang[idxI][idxJ].kapasitas) then

```

```

output("Anda sudah berada di antrean terdepan,
tidak perlu skip!")
→

tempCount ← 0
k traversal
[0..rumah_sakit.ruang[idxI][idxJ].kapasitas-1]
if (not
isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean)) then
    userDalam[tempCount] ←
    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
    tempCount ← tempCount + 1

antreanCount ← 0
while (not
isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean))
and not
isSameUser(rumah_sakit.ruang[idxI][idxJ].antrean.pasi
en, current_user) do
    antreanUsers[antreanCount] ←
    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
    antreanCount ← antreanCount + 1

if (not
isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean)
and
isSameUser(rumah_sakit.ruang[idxI][idxJ].antrean.pasi
en, current_user)) then
    prioritas ←
    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)

sisaCount ← 0
while (not
isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean))
do
    sisaUsers[sisaCount] ←
    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
    sisaCount ← sisaCount + 1

k traversal [0..tempCount-1]
enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
userDalam[k])

```

```

        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        prioritas)

    k traversal [0..antreanCount-1]
        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        antreanUsers[k])

    k traversal [0..sisaCount-1]
        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        sisaUsers[k])

        output("✓ Anda berhasil dipindahkan ke antrean
        terdepan!")
        output("Posisi baru Anda: Antrean ke-1 (setelah
        pasien yang sedang dalam ruangan)")

```

## 21. Cancel Antrean

```

prosedur cancelAntrean (input/output: current_user: User,
rumah_sakit: MatriksRuangan)
{ I.S. current_user sedang login sebagai pasien dan mungkin berada dalam
antrean }
{ F.S. Jika current_user berada dalam antrean, maka ia dibatalkan dari antrean;
antrean direkonstruksi }

```

## KAMUS LOKAL

idxI, idxJ, idxPasien	:	integer
i, j	:	<u>integer</u>
found	:	<u>boolean</u>
q	:	Address
userDalam, antreanUsers	:	<u>array of</u> User
sisaUsers	:	<u>array of</u> User
tempCount, antreanCount	:	<u>integer</u>
sisaCount	:	<u>integer</u>

## ALGORITMA

```

cancelAntreanHeaderInterface()

if (current_user.role ≠ "pasien") then
    output("Hanya pasien yang dapat mengakses fitur
    ini!")
    →

    idxI ← -1
    idxJ ← -1
    found ← false

```

```

{ Cari ruangan tempat user berada dalam antrean }
i ← 0
while (i < rumah_sakit.row and found = false) do
    j ← 0
    while (j < rumah_sakit.column and found = false) do
        if (not isRuangEmpty(rumah_sakit.ruang[i][j]) and not isQueueEmpty(rumah_sakit.ruang[i][j].antrean)) then
            q ← rumah_sakit.ruang[i][j].antrean
            while (q ≠ NULL) do
                if (isSameUser(q.pasien, current_user)) then
                    idxI ← i
                    idxJ ← j
                    found ← true
                    q ← q.next
                j ← j + 1
            i ← i + 1

            if (not found or idxI = -1 or idxJ = -1) then
                output("Anda tidak terdaftar dalam antrean!")
            →
                idxPasien ←
                findInQueue(rumah_sakit.ruang[idxI][idxJ].antrean,
                current_user)

            if (idxPasien < rumah_sakit.ruang[idxI][idxJ].kapasitas) then
                output("Anda sudah berada di dalam ruangan,
                tidak bisa membatalkan antrean!")
            →
                if (idxPasien = rumah_sakit.ruang[idxI][idxJ].kapasitas) then
                    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
                    output("✓ Anda berhasil membatalkan antrean
                    dokter " +
                    rumah_sakit.ruang[idxI][idxJ].dokter.username +
                    " ruangan " + chr('A' + idxI) + intToStr(idxJ +
                    1))

```

```

→

{ Simpan pasien dalam ruangan }
tempCount ← 0
i traversal
[0..rumah_sakit.ruang[idxI][idxJ].kapasitas-1]
    if (not
        isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean)) then
            userDalam[tempCount] ←
            dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
            tempCount ← tempCount + 1

{ Simpan antrean sebelum user }
antreanCount ← 0
while (not
    isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean)
    and not
        isSameUser(rumah_sakit.ruang[idxI][idxJ].antrean.pasi
en, current_user)) do
        antreanUsers[antreanCount] ←
        dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
        antreanCount ← antreanCount + 1

{ Hapus current_user }
if (not
    isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean)
    and
        isSameUser(rumah_sakit.ruang[idxI][idxJ].antrean.pasi
en, current_user)) then
            dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)

{ Simpan sisa antrean }
sisaCount ← 0
while (not
    isQueueEmpty(rumah_sakit.ruang[idxI][idxJ].antrean))
do
    sisaUsers[sisaCount] ←
    dequeue(rumah_sakit.ruang[idxI][idxJ].antrean)
    sisaCount ← sisaCount + 1

{ Rekonstruksi antrean }
i traversal [0..tempCount-1]

```

```

        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        userDalam[i])
    i traversal [0..antreanCount-1]
        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        antreanUsers[i])
    i traversal [0..sisaCount-1]
        enqueue(rumah_sakit.ruang[idxI][idxJ].antrean,
        sisaUsers[i])

    output("✓ Anda berhasil membatalkan antrean dokter "
+ rumah_sakit.ruang[idxI][idxJ].dokter.username + "
ruangan " + chr('A' + idxI) + intToStr(idxJ + 1))

```

## 22. B02 - Denah Dinamis

```

procedure unassign_dokter(input current_user : User,
input database : ListUser, input/output denah :
MatriksRuangan)
{ I.S. User sudah login }
{ F.S. Dokter yang dipilih akan di-unassign dari ruangan jika memenuhi syarat }

```

## KAMUS LOKAL

nama_dokter	: string
found_user	: pointer to User
row, col	: integer
row_ruang, col_ruang	: integer
ruang_dokter	: pointer to Ruangan

## ALGORITMA

```

row_ruang, col_ruang ← -1
ruang_dokter ← null

if (lowercase(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk fitur
unassign dokter!")
    →

output("Username: ")
input(nama_dokter)
found_user ← getUserByName(database, nama_dokter)

if (found_user = null) then
    output("Username tidak terdaftar!")
    →

if (lowercase(found_user^.role) ≠ "dokter") then

```

```

output("User ", found_user^.username, " bukan
seorang dokter!")
→

row traversal [0..denah.row-1]
    col traversal [0..denah.col-1]
        if
            (isSameUser(denah.ruang[row][col].dokter,
            found_user^)) then
                output("Dokter
                found_user^.username, " di-assign di
                ruangan ", chr(row + 'A'), col + 1)
                ruang_dokter ← address of
                denah.ruang[row][col]
                row_ruang ← row
                col_ruang ← col

        if (ruang_dokter = null) then
            output("Dokter ", found_user^.username, " belum
            di-assign ke ruang mana pun!")
        else if (not isQueueEmpty(ruang_dokter^.antrean))
    then
        if (queueLength(ruang_dokter^.antrean) >
        ruang_dokter^.kapasitas) then
            output("Ada pasien di ruangan dan antrean
            untuk dokter ini!")
        else
            output("Ada pasien di ruangan dokter
ini!")
        output("Dokter yang sedang melayani pasien
tidak bisa di-unassign!")
    else
        destroyUser(ruang_dokter^.dokter)
        output("Berhasil unassign dokter
        found_user^.username, ", ruangan ",",
        chr(row_ruang + 'A'), col_ruang + 1, "
        kosong.")

procedure tambah_baris(input current_user : User,
input/output denah : MatriksRuangan)
{ I.S. User sudah login }
{ F.S. Jika user adalah manager dan ada sisa lahan, baris ruangan ditambah }


```

## KAMUS LOKAL

```
jumlah_ruangan_baru : integer
```

## ALGORITMA

```
if (lowercase(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk
    mengubah layout rumah sakit!")
    →

if (denah.row < MAX_ROWS) then
    expandRow(denah)
    lihatDenah(current_user, denah)
    output("Baris ruangan ", chr('A' + denah.row -
    1), " ditambahkan ke rumah sakit!")
    jumlah_ruangan_baru ← denah.column
    output("Jumlah ruangan bertambah sebanyak ", 
    jumlah_ruangan_baru, "!")
    output("Jumlah baris saat ini: ", denah.row)

if (denah.row = MAX_ROWS) then
    output("Tidak ada lahan tersisa untuk
    menambah baris.")
else
    output("Masih tersisa lahan untuk menambah
    ", MAX_ROWS - denah.row, " baris.")

else
    output("Gagal menambah baris: lahan sudah
    habis!")
    output("Jumlah baris saat ini: ", denah.row)

procedure tambah_kolom(input current_user : User,
input/output denah : MatriksRuangan)
{ I.S. User sudah login }
{ F.S. Jika user adalah manager dan ada sisa lahan, kolom ruangan ditambah }
```

## KAMUS LOKAL

```
jumlah_ruangan_baru : integer
```

## ALGORITMA

```
if (lowercase(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk
    mengubah layout rumah sakit!")
    →

if (denah.column < MAX_COLS) then
```

```

expandCol(denah)
lihatDenah(current_user, denah)
output("Kolom ruangan ", denah.column, " ditambahkan ke rumah sakit!")
jumlah_ruangan_baru ← denah.row
output("Jumlah ruangan bertambah sebanyak ", jumlah_ruangan_baru, "!")
output("Jumlah kolom saat ini: ", denah.column)

if (denah.column = MAX_COLS) then
    output("Tidak ada lahan tersisa untuk menambah kolom.")
else
    output("Masih tersisa lahan untuk menambah ", MAX_COLS - denah.column, " kolom.")
else
    output("Gagal menambah kolom: lahan sudah habis!")
output("Jumlah kolom saat ini: ", denah.column)

procedure kurangi_baris(input current_user : User,
input/output denah : MatriksRuang)
{ I.S. User sudah login }
{ F.S. Jika baris terakhir kosong, baris dihapus dan dokter di-unassign }

```

## KAMUS LOKAL

```

j : integer
deletable : boolean

```

## ALGORITMA

```

if (lowercase(current_user.role) ≠ "manager") then
    output("Anda tidak memiliki akses untuk mengubah layout rumah sakit!")
    →

if (denah.row > 1) then
    deletable ← true

    j traversal [0..denah.column-1]
        if (not isQueueEmpty(denah.ruang[denah.row-1][j].antrean)) then
            output("WARNING: dr. ", denah.ruang[denah.row-1][j].dokter.us

```

```

        ername, " sedang melayani pasien di
ruangan ", chr(denah.row + 64), j +
1)
deletable ← false

if (deletable) then
    shrinkRow(denah)
    lihatDenah(current_user, denah)

    j traversal [0..denah.column-1]
    if
        (isUserValid(denah.ruang[denah.row-1]
[j].dokter)) then
            output("dr.           ",
denah.ruang[denah.row-1][j].dok
ter.username, " di-unassign
dari ruangan ", chr(denah.row +
64), j + 1)

            destroyUser(denah.ruang[denah.r
ow-1][j].dokter)

            output("Jumlah baris saat ini: ",
denah.row)
        else
            output("Gagal mengurangi baris: dokter di
baris terakhir sedang melayani pasien!")
    else
        output("Gagal mengurangi baris: rumah sakit
butuh ruangan untuk beroperasi!")
        output("Saat ini tersisa 1 baris ruangan untuk
pasien!")

procedure kurangi_kolom(input current_user : User,
input/output denah : MatriksRuang)
{ I.S. User sudah login }
{ F.S. Jika kolom terakhir kosong, kolom dihapus dan dokter di-unassign }


```

## KAMUS LOKAL

```

i          : integer
deletable : boolean

```

## ALGORITMA

```

if (lowercase(current_user.role) ≠ "manager") then

```

```

output("Anda tidak memiliki akses untuk
mengubah layout rumah sakit!")
→

if (denah.column > 1) then
    deletable ← true

    i traversal [0..denah.row-1]
        if (not
            isQueueEmpty(denah.ruang[i][denah.column-1
                ].antrean)) then
                output("WARNING: dr. ", denah.ruang[i][denah.column-1].dokter
                    .username, " sedang melayani pasien
                    di ruangan ", chr(i + 65),
                    denah.column)
                deletable ← false

        if (deletable) then
            shrinkCol(denah)
            lihatDenah(current_user, denah)

            i traversal [0..denah.row-1]
                if
                    (isUserValid(denah.ruang[i][denah.col
                        umn-1].dokter)) then
                        output("dr. ", denah.ruang[i][denah.column-1].
                            dokter.username, " di-unassign
                            dari ruangan ", chr(i + 65),
                            denah.column)

                        destroyUser(denah.ruang[i][dena
                            h.column-1].dokter)

                output("Jumlah kolom saat ini: ", denah.column)
            else
                output("Gagal mengurangi kolom: dokter di
                    kolom terakhir sedang melayani pasien!")
            else
                output("Gagal mengurangi kolom: rumah sakit
                    butuh ruangan untuk beroperasi!")
                output("Saat ini tersisa 1 kolom ruangan untuk
                    pasien!")

```

### 23. B04 - Banarich

```
procedure lihat_dompet(input current_user : User)
{ I.S. User telah login atau belum }
{ F.S. Menampilkan isi dompet user jika valid dan bukan manager }
```

### KAMUS LOKAL

-

### ALGORITMA

```
if (not isUserValid(current_user)) then
    output("Login untuk mengakses dompet!")
    →

if (lowercase(current_user.role) = "manager") then
    output("Manager tidak memiliki dompet pribadi!")
    output("Gunakan LIHAT_FINANSIAL untuk melihat kondisi keuangan rumah sakit!")
    →

procedure lihat_finansial(input current_user : User)
{ I.S. User telah login sebagai manager }
{ F.S. Menampilkan jumlah keuangan rumah sakit }
```

### KAMUS LOKAL

-

### ALGORITMA

```
if (lowercase(current_user.role) ≠ "manager") then
    output("Hanya manager yang berhak melihat kondisi keuangan rumah sakit!")
    →

output("Rumah sakit saat ini memiliki ", keuangan, " banarich.")
output("Tetaplah bekerja keras untuk memperkaya pemilik rumah sakit!")

procedure gacha_gaming(input/output current_user : pointer to User, input/output database_user : pointer to ListUser, input/output prev_rng : unsigned)
{ I.S. current_user sudah login }
```

{ F.S. Jika user adalah pasien, maka akan mendapatkan sejumlah banarich secara acak }

## KAMUS LOKAL

```
gain           : unsigned
pointer_id     : pointer to User
pointer_username : pointer to User
```

## ALGORITMA

```
if (not isUserValid(current_user)) then
    output("Login sebagai pasien untuk menggunakan
    fitur gacha!")
    →

if (lowercase(current_user.role) ≠ "pasien") then
    output("Hanya pasien yang dapat menggunakan
    fitur gacha!")
    →

output("Kchak.. Kchak.. Thunk!")
gain ← (3 * prev_rng + 1) mod 29
prev_rng ← gain
output("Kamu mendapatkan ", gain, " banarich!")

current_user.banarich ← current_user.banarich + gain

pointer_id           ←      getUserByID(database_user,
current_user.id)
pointer_id.banarich ← current_user.banarich

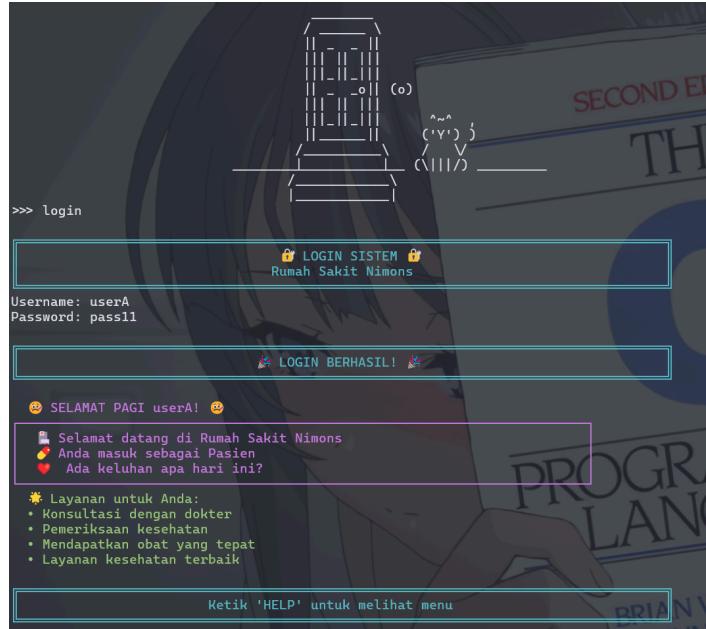
pointer_username     ←      getUserByName(database_user,
current_user.username)
pointer_username.banarich ← current_user.banarich
```

## BAB III

# PENGUJIAN DAN HASIL

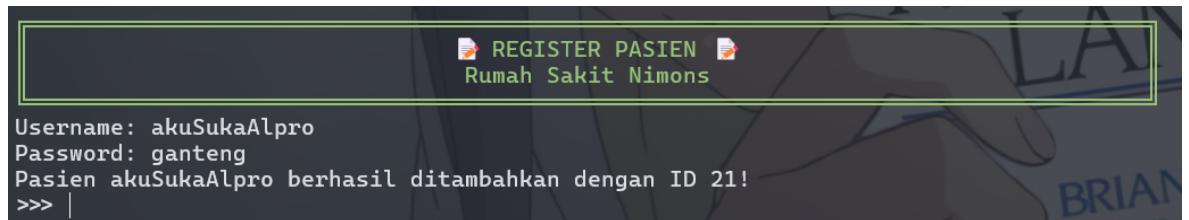
### 3.1 Dokumentasi Input dan Output Valid

#### 3.1.1 Login



Gambar 3.1.1 Antarmuka Login Pengguna

#### 3.1.2 Register



Gambar 3.1.2 Antarmuka Register Pasien

#### 3.1.3 Logout



Gambar 3.1.3 Antarmuka Logout Pengguna

### 3.1.4 Lupa Password

```
>>> lupa_password

    🔒 LUPA PASSWORD 🔒
    Rumah Sakit Nimons

Username: userA
Kode Unik: userA
Halo pasien userA, silakan daftarkan ulang password anda!
Password Baru: akuCintaAlpro
Password berhasil diubah! Silakan login dengan password baru.

>>> login

    🔒 LOGIN SISTEM 🔒
    Rumah Sakit Nimons

Username: userA
Password: akuCintaAlpro

    🎉 LOGIN BERHASIL! 🎉

    😊 SELAMAT PAGI userA! 😊

    🌟 Selamat datang di Rumah Sakit Nimons
    🍯 Anda masuk sebagai Pasien
    ❤️ Ada keluhan apa hari ini?

    ★ Layanan untuk Anda:
    • Konsultasi dengan dokter
    • Pemeriksaan kesehatan
    • Mendapatkan obat yang tepat
    • Layanan kesehatan terbaik

    Ketik 'HELP' untuk melihat menu

>>> |
```

Gambar 3.1.4 Antarmuka Lupa Password

### 3.1.5 Menu dan Help

```
>>> help

    🎨 HELP MENU 🎨
    Rumah Sakit Nimons

⚠️ Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.

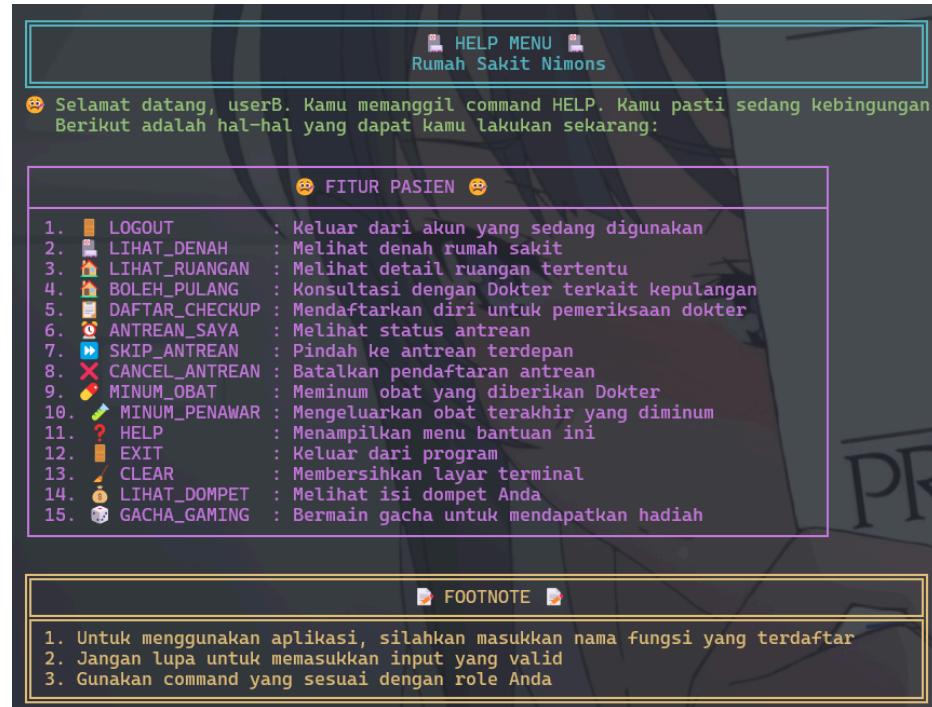
    🔒 FITUR YANG TERSEDIA 🔒

1. 🎖️ LOGIN : Masuk ke dalam akun yang sudah terdaftar
2. 📝 REGISTER : Membuat akun baru sebagai pasien
3. 🔒 LUPA_PASSWORD: Reset password jika lupa
4. ? HELP : Menampilkan menu bantuan ini
5. 🚪 EXIT : Keluar dari program

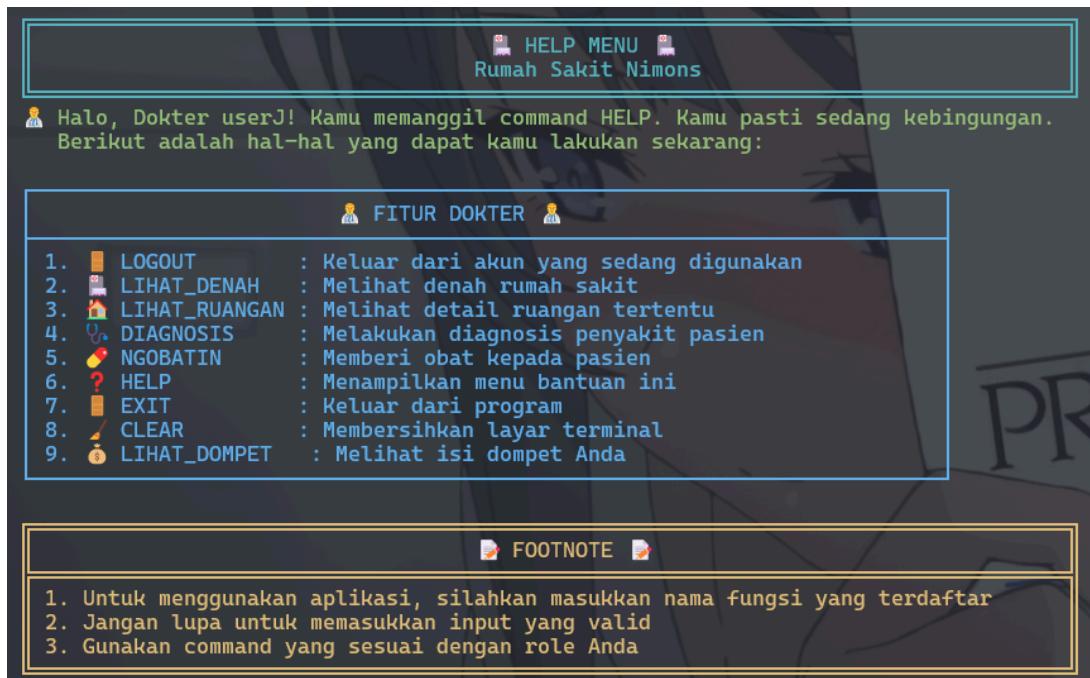
    📄 FOOTNOTE 📄

1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid
3. Gunakan command yang sesuai dengan role Anda
```

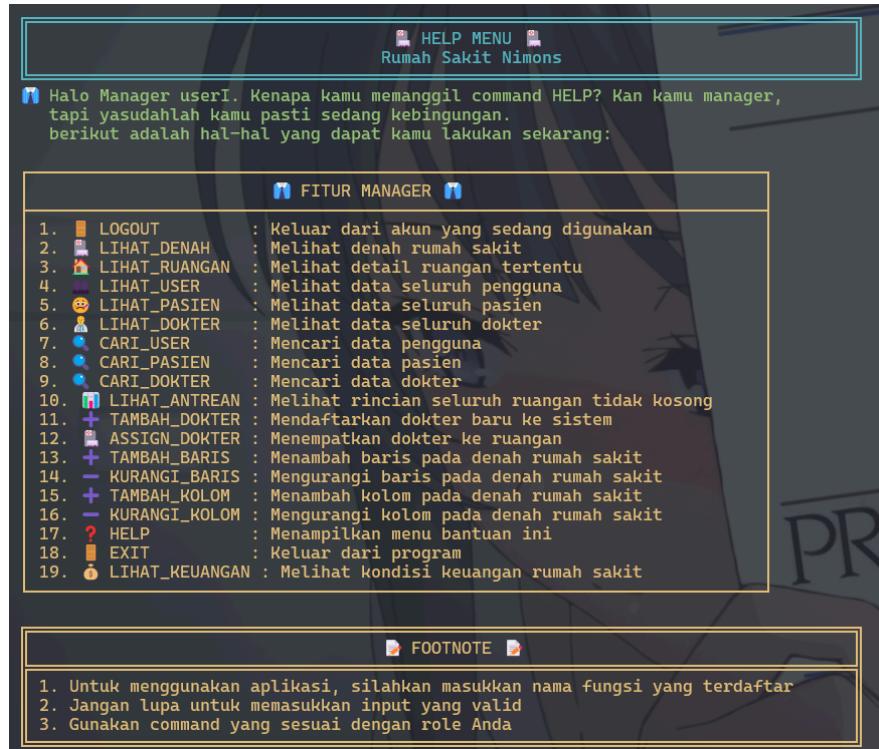
Gambar 3.1.5.1 Antarmuka Menu dan Help



Gambar 3.1.5.2 Antarmuka Menu dan Help Pasien

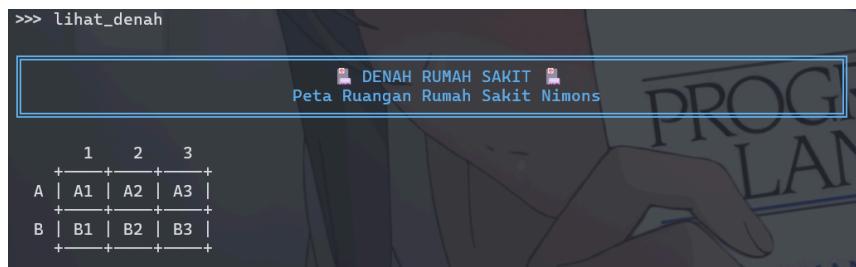


Gambar 3.1.5.3 Antarmuka Menu dan Help Dokter



Gambar 3.1.5.4 Antarmuka Menu dan Help Manager

### 3.1.6 Lihat Denah



Gambar 3.1.6 Antarmuka Lihat Denah

### 3.1.7 Lihat User

```
>>> lihat_user

          ♡ DAFTAR PENGGUNA ♡
Data Semua Pengguna Sistem Nimon

Urutkan berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2

Urutan sort?
1. ASC (A-Z)
2. DESC (Z-A)
>>> Pilihan: 2

ID | Nama      | Role     | Penyakit
--|---|---|---|
20 | userT    | pasien   | -
19 | userS    | dokter   | -
18 | userR    | pasien   | -
17 | userQ    | pasien   | -
16 | userP    | pasien   | -
15 | userO    | dokter   | -
14 | userN    | dokter   | -
13 | userM    | dokter   | -
12 | userL    | dokter   | -
11 | userK    | dokter   | -
10 | userJ    | dokter   | -
9  | userI    | manager  | -
8  | userH    | pasien   | -
7  | userG    | pasien   | -
6  | userF    | pasien   | -
5  | userE    | pasien   | -
4  | userD    | pasien   | -
3  | userC    | pasien   | -
2  | userB    | pasien   | -
1  | userA    | pasien   | -
```

Gambar 3.1.7 Antarmuka Lihat User

### 3.1.8 Cari User

```
>>> cari_user

          🔎 CARI PENGGUNA 🔎
Pencarian Data Pengguna Sistem

Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 1

>>> Masukkan nomor ID user: 1

Menampilkan pengguna dengan nomor ID 1 ...
ID | Nama      | Role     | Penyakit
--|---|---|---|
1  | userA    | pasien   | -
>>> cari_user

          🔎 CARI PENGGUNA 🔎
Pencarian Data Pengguna Sistem

Cari berdasarkan?
1. ID
2. Nama
>>> Pilihan: 2

>>> Masukkan nama user: Userb

Menampilkan pengguna dengan nama Userb ...
ID | Nama      | Role     | Penyakit
--|---|---|---|
2  | userB    | pasien   | -
>>>
```

Gambar 3.1.8 Antarmuka Cari User

### 3.1.9 Lihat Antrean

```
    LIHAT ANTREAN
Status Antrean Seluruh Ruangan Aktif

>>> LIHAT ANTREAN

      1   2   3
A | A1 | A2 | A3 |
+---+---+---+
B | B1 | B2 | B3 |
+---+---+---+


--- Informasi Ruangan A1 ---
Kapasitas      : 3
Dokter        : userJ
Pasien di dalam ruangan:
1. userB
2. userC
3. userA
Antrean pasien:
1. userP
2. userT
3. userQ

-----
--- Informasi Ruangan A2 ---
Kapasitas      : 3
Dokter        : userK
Pasien di dalam ruangan:
1. userD
2. userE
Antrean pasien:
Tidak ada pasien dalam antrean saat ini.

-----
--- Informasi Ruangan A3 ---
Kapasitas      : 3
Dokter        : userL
Pasien di dalam ruangan:
1. userF
Antrean pasien:
Tidak ada pasien dalam antrean saat ini.

-----
--- Informasi Ruangan B1 ---
Kapasitas      : 3
Dokter        : userM
Pasien di dalam ruangan:
1. userH
2. userG
Antrean pasien:
Tidak ada pasien dalam antrean saat ini.

-----
--- Informasi Ruangan B3 ---
Kapasitas      : 3
Dokter        : userO
Pasien di dalam ruangan:
Tidak ada pasien di dalam ruangan saat ini.
```

Gambar 3.1.9 Antarmuka Lihat Antrean

### 3.1.10 Tambah Dokter

```
+ TAMBAH DOKTER +  
Pendaftaran Dokter Baru ke Sistem  
  
Username: D0KT3R_4L4Y  
Registrasi gagal! Username hanya boleh berisi huruf (tanpa angka atau simbol).  
Username: UserI  
User UserI sudah terdaftar.  
Daftar dengan username lain  
atau  
gunakan command LUPA_PASSWORD jika kamu lupa password akunmu.  
Username: dokterbaru  
Password: purapuradokterpadahalpembunuhberantai  
Dokter dokterbaru berhasil ditambahkan dengan ID 21!  
>>> |
```

Gambar 3.1.10 Antarmuka Lihat Antrian untuk ruangan tidak berpasien, ruangan kosong tidak ditampilkan

### 3.1.11 Diagnosis

```
>>> diagnosis  
  
+-----+  
| _DIAGNOSIS PASIEN |  
| Pemeriksaan dan Diagnosis Medis |  
+-----+  
  
>>> DIAGNOSIS  
Pasien userB terdiagnosis menderita penyakit COVID-19.  
>>> ngobatin
```

Gambar 3.1.11 Antarmuka Diagnosis

### 3.1.12 Ngobatin

```
+-----+  
| PEMBERIAN OBAT |  
| Resep dan Distribusi Obat Pasien |  
+-----+  
  
>>> NGOBATIN  
Pasien userB terdiagnosis menderita penyakit COVID-19.  
Obat yang harus diberikan:  
  (ID: 3) Ibuprofen  
  (ID: 17) Jus Daun Ganja  
  (ID: 37) Senyuman Sang Tercinta  
  (ID: 59) Lumpur Lapindo  
  (ID: 79) Air Mata Phoenix  
Semua obat berhasil diberikan ke inventory pasien userB.
```

Gambar 3.1.12 Antarmuka Ngobatin

### 3.1.13 Aku boleh pulang ga, dok?



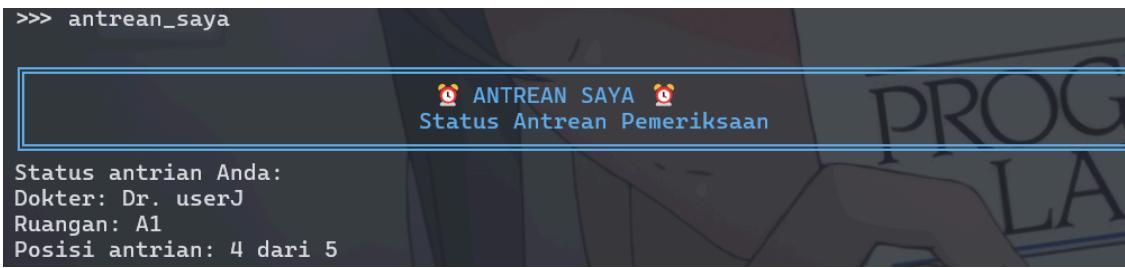
Gambar 3.1.13 Antarmuka Aku boleh pulang ga, dok?

### 3.1.14 Daftar Check-Up

DAFTAR CHECKUP	
Pendaftaran Pemeriksaan Kesehatan	
Silakan masukkan data kesehatan anda	
Suhu tubuh (celcius): 38	
Tekanan darah (sistol/diastol, misal 120 80): 120 80	
Detak jantung (bpm): 80	
Saturasi oksigen (%): 98	
Kadar gula darah (mg/dL): 40	
Berat badan (kg): 60	
Tinggi badan (cm): 170	
Kadar kolesterol (mg/dL): 20	
Trombosit (ribu/uL): 210	
Berikut adalah daftar dokter yang tersedia:	
1.	Dr. userJ - Ruangan A1
Aura	: 3
Pasien di ruangan	: 3
Pasien di antrean	: 2
Biaya checkup	: 5
2.	Dr. userK - Ruangan A2
Aura	: 1
Pasien di ruangan	: 2
Pasien di antrean	: -
Biaya checkup	: 2
3.	Dr. userL - Ruangan A3
Aura	: 2
Pasien di ruangan	: 1
Pasien di antrean	: -
Biaya checkup	: 4
4.	Dr. userM - Ruangan B1
Aura	: 2
Pasien di ruangan	: 2
Pasien di antrean	: -
Biaya checkup	: 4
5.	Dr. userO - Ruangan B3
Aura	: 3
Pasien di ruangan	: -
Pasien di antrean	: -
Biaya checkup	: 5
Pilih dokter yang tersedia (1-5, 0 untuk batal): 5	
Pendaftaran berhasil!	
Anda terdaftar antrean checkup dengan dr. userO di ruangan B3.	
Anda bisa langsung masuk ke ruangan	
>>>	

Gambar 3.1.14 Antarmuka Daftar Check-up

### 3.1.15 Antrean Saya



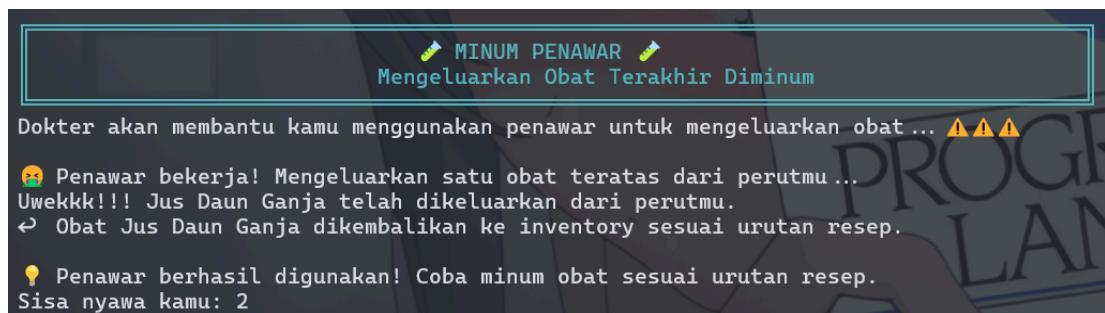
Gambar 3.1.15 Antarmuka Antrean Saya

### 3.1.16 Minum Obat



Gambar 3.1.16 Antarmuka Minum Obat

### 3.1.17 Minum Penawar



Gambar 3.1.17 Antarmuka Minum Penawar

### 3.1.18 Exit

```
>>> exit

[ EXIT SISTEM ]
Terima kasih telah menggunakan
Sistem Rumah Sakit Nimons!

Mau save perubahan? (Y/N)
y
Masukkan nama folder untuk menyimpan data: save
Folder "save" berhasil dibuat.
Berhasil menyimpan konfigurasi ke config.txt
Berhasil menyimpan data user ke user.csv
Berhasil menyimpan data penyakit ke penyakit.csv
Berhasil menyimpan data obat ke obat.csv
Berhasil menyimpan data obat penyakit ke obat_penyakit.csv
Semua data berhasil disimpan ke folder "save".
Terima kasih telah menggunakan aplikasi Rumah Sakit Nimons!
Sampai jumpa lagi!
```

Gambar 3.1.18 Antarmuka Exit

### 3.1.19 Bonus Mainan Antrean

```
>>> skip_antrean

[ ► SKIP ANTREAN ► ]
Pindah ke Antrean Terdepan

[ ✓ Anda berhasil dipindahkan ke antrean terdepan di ruangan dokter userJ ruangan A1
Posisi baru Anda: Antrean ke-1 (setelah pasien yang sedang dalam ruangan)
>>> |
```

Gambar 3.1.19.1 Antarmuka Skip Antrean

```
>>> cancel_antrean

[ X CANCEL ANTREAN X ]
Batalkan Pendaftaran Antrean

[ ✓ Anda berhasil membatalkan antrean dokter userJ ruangan A1
```

Gambar 3.1.19.2 Antarmuka Cancel Antrean

### 3.1.20 Bonus Denah Dinamis

```
>>> tambah_baris
```

DENAH RUMAH SAKIT  
Peta Ruangan Rumah Sakit Nimons

	1	2	3	
A	A1	A2	A3	
B	B1	B2	B3	
C	C1	C2	C3	

Baris ruangan C ditambahkan ke rumah sakit!

Jumlah ruangan bertambah sebanyak 3!

Jumlah baris saat ini: 3

Masih tersisa lahan untuk menambah 23 baris.

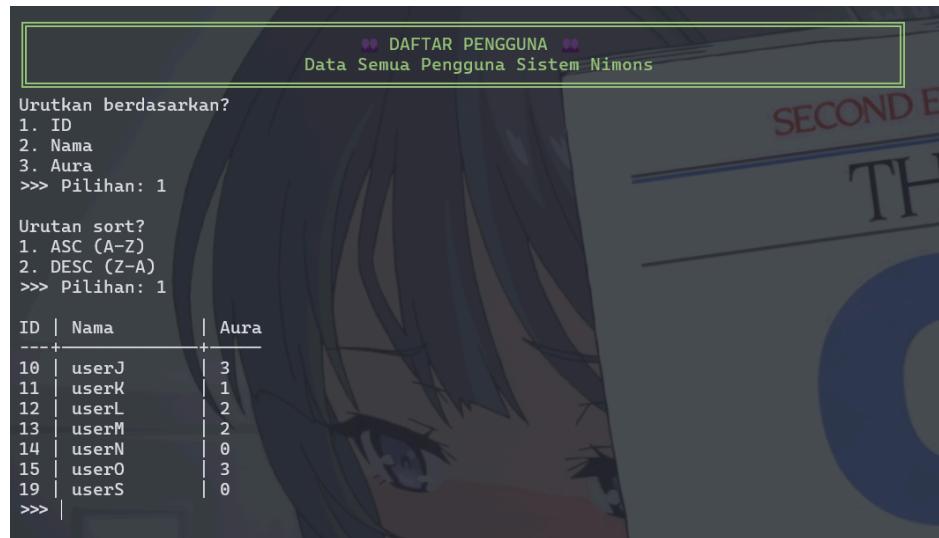
```
>>> tambah_kolom
```

DENAH RUMAH SAKIT  
Peta Ruangan Rumah Sakit Nimons

	1	2	3	4	
A	A1	A2	A3	A4	
B	B1	B2	B3	B4	
C	C1	C2	C3	C4	

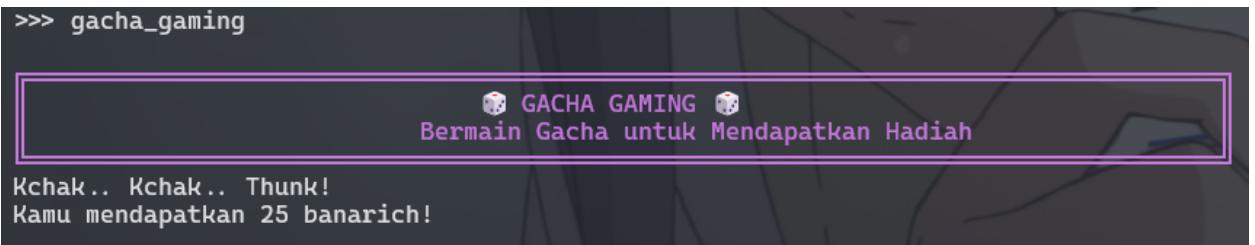
Gambar 3.1.20 Antarmuka Denah Dinamis

### 3.11.21 Bonus Aura



Gambar 3.1.21 Antarmuka Aura

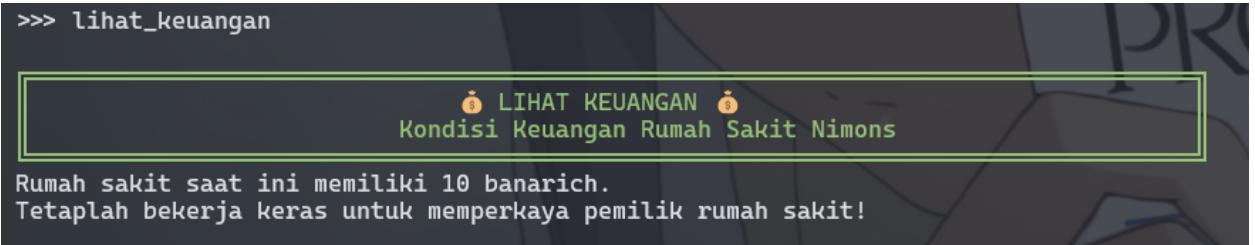
### 3.1.21 Bonus Bananarich



Gambar 3.1.22.1 Antarmuka Gacha Gaming

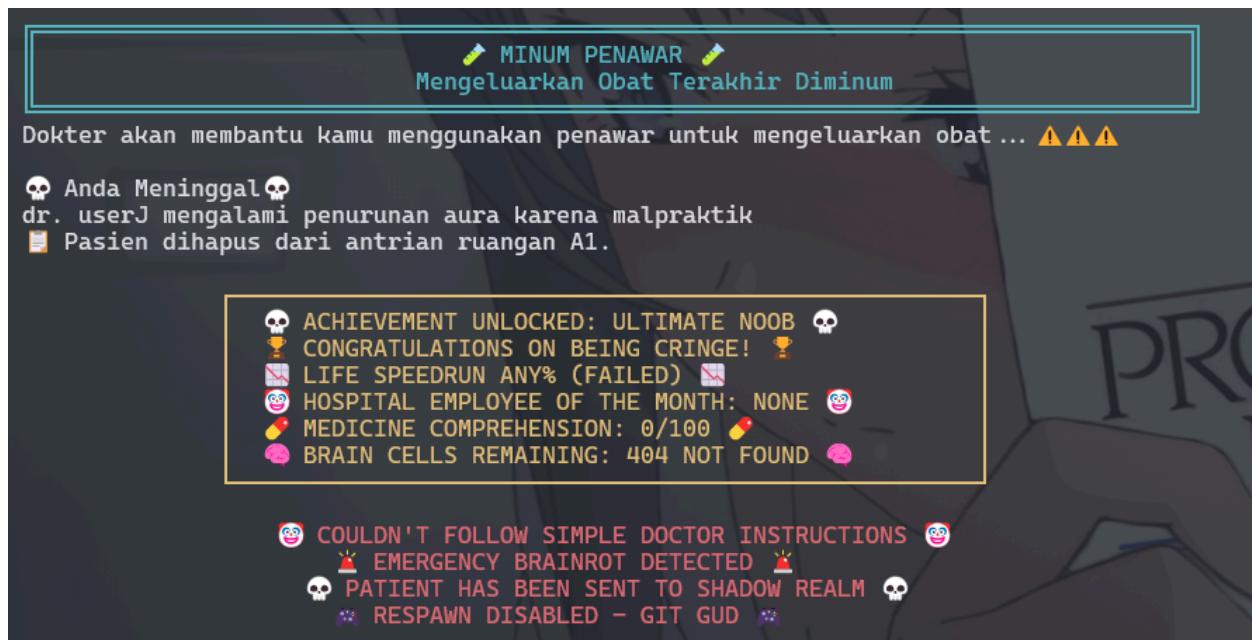


Gambar 3.1.22.2 Antarmuka Lihat Dompet



Gambar 3.1.22.3 Antarmuka Lihat Dompet

### 3.1.22 Bonus Dead or Alive



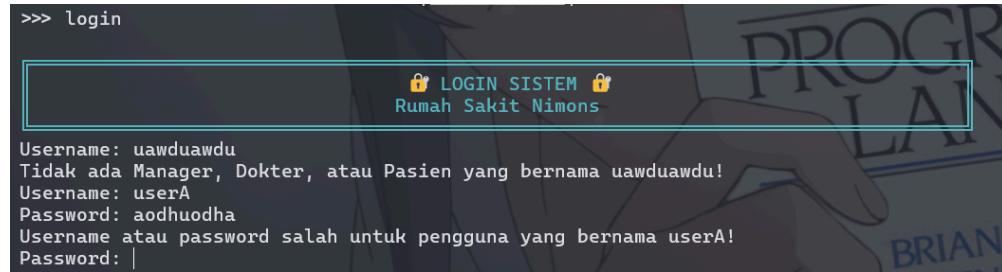
Gambar 3.1.22.3 Antarmuka User Mati

## 3.2 Dokumentasi Input dan Output Tidak Valid

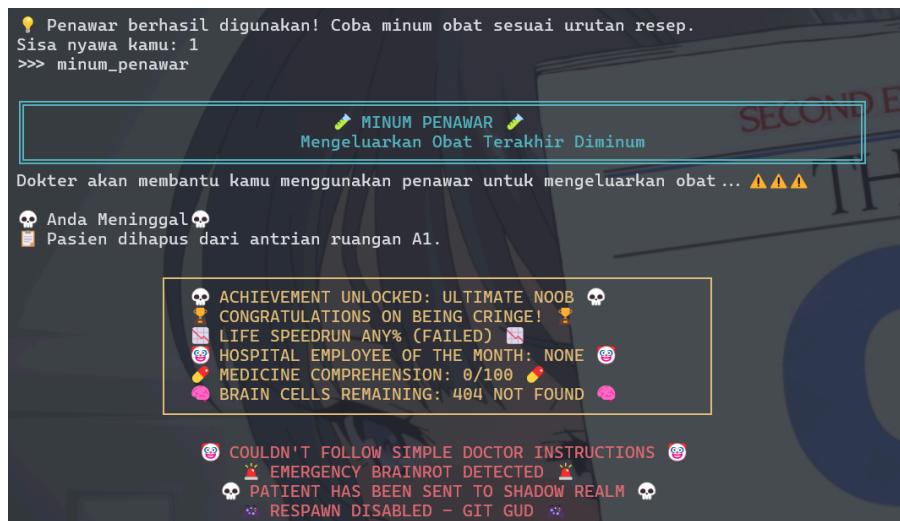
Berikut merupakan beberapa contoh kasus input dan output tidak valid di berbagai fitur.



Gambar 3.2.1 Tampilan antarmuka Daftar Check-up yang menampilkan pesan kesalahan dan mengulang input hingga data yang dimasukkan valid.



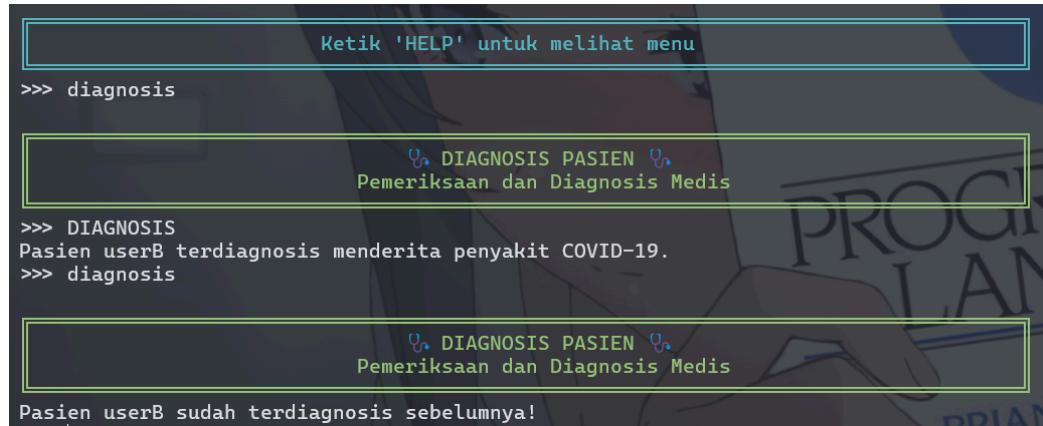
Gambar 3.2.2 Tampilan antarmuka Login yang menampilkan pesan kesalahan dan mengulang input hingga username/password sesuai



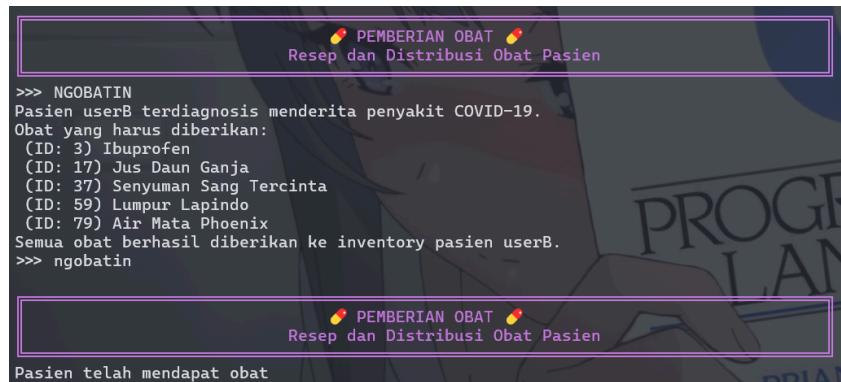
Gambar 3.2.3 Tampilan antarmuka saat Pasien Meninggal (disebabkan kesalahan dalam urutan meminum obat)



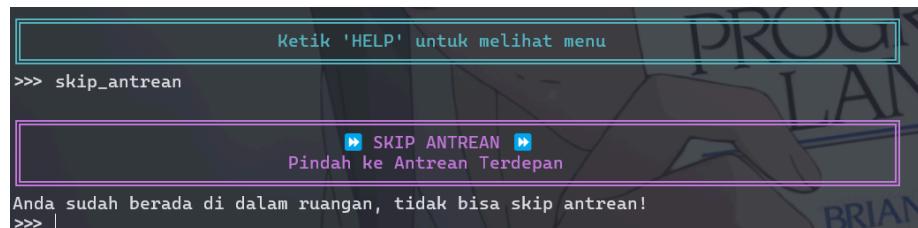
Gambar 3.2.4 Sistem Menolak Login dari Akun Pasien Meninggal



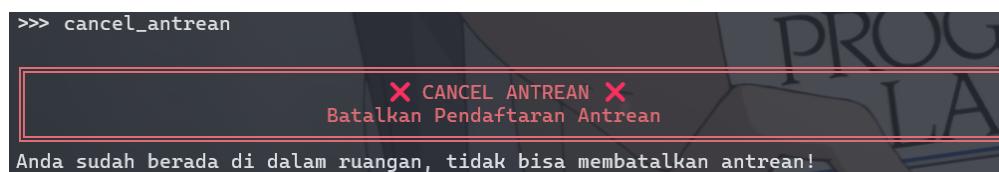
Gambar 3.2.5 Tampilan antarmuka saat Mendiagnosis Pasien yang telah Terdiagnosa



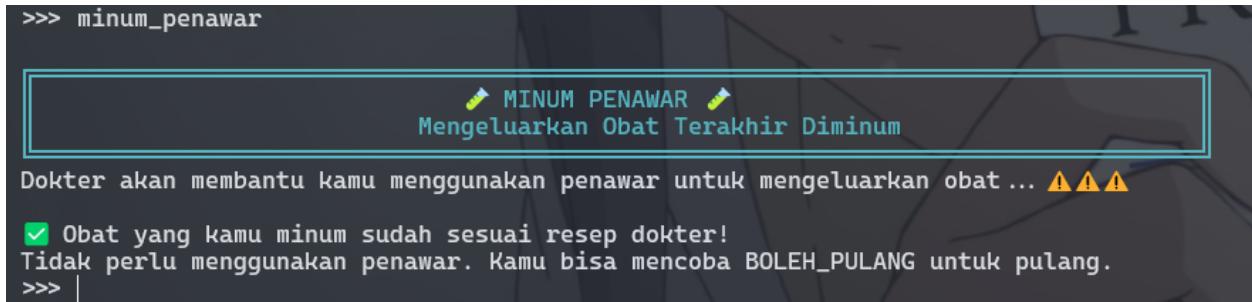
Gambar 3.2.6 Tampilan Antarmuka saat Ngobatin Pasien yang telah Mendapatkan Obat



Gambar 3.2.7 Tampilan Antarmuka saat Skip Antrean



Gambar 3.2.8 Tampilan Antarmuka saat Cancel Antrean



Gambar 3.2.8 Tampilan Antarmuka saat Minum Penawar

### 3.3 Penjelasan Hasil Uji

Kami telah melakukan testing dari program rumah sakit yang telah kami buat. Program yang kami buat sudah cukup baik dalam mengantisipasi kasus-kasus inputan yang tidak valid. Jika kita menginput username dan password, maka program akan memberikan notifikasi bahwa input username dan password kita salah. Pada fitur diagnosis, jika seorang user telah didiagnosis maka diagnosis selanjutnya tidak akan jalan. Kemudian, dalam fitur ngobatin, jika pasien sudah pernah mendapat obat, maka pasien itu tidak perlu mendapatkan obat lagi. Pada fitur status checkup, ketika inputan user bukan angka, maka program akan menyarankan kita untuk melakukan input berupa angka dengan range yang disarankan. Pada kasus *Dead or Alive*, jika seorang user meninggal, maka ketika kita mencoba login dengan akun user itu, maka program akan menolak proses loginnya.

# LAMPIRAN

## 1. Hasil Pindai Form Asistensi

Form MoM Asistensi Tugas Besar IF1210/Algoritma dan Pemrograman 1 Sesi. 2 2024/2025	
Nomor Asisten : 1 No. Kelompok/Kelas : LK01 Tanggal asistensi : 30 April 2025	
Anggota kelompok : NIM / Nama yang Hadir 1 13524131 / Amanah Andilla Sabahill 2 13524065 / Kurt Mihael Purba 3 13524045 / Ahmad Zaky Robbani 4 13524113 / Fauzan Mohamad Abdul Ghani 5 13524079 / Angelina Andra Alanna 6 -	
Asisten pembimbing : NIM / Nama 13522164/Valentino Chrysle Triad	
Catatan Asistensi:	
<p>Rangkuman Diskusi</p> <ul style="list-style-type: none"><li>Senja harus dipakai di F00 (khusus IF).</li><li>Kalau dokternya tambah dengannya tetapi dokter nganggur (ada bonus bisa nambah ruang).</li><li>unassigned doctor masih dalam BONUS.</li><li>Setiap hari ada 1000 pasien, tidak ada ditarif check up lagi.</li><li>Gak boleh kurangin rx, bolah nambah.</li><li>Satu fatur boleh beberapa rx, minimal 1 di F00.</li><li>Jumlah penawaranya unlimited.</li><li>Sampa dia sehat baru ququevna berkurang. Kalau engga anggap di stay di ruang itu terus.</li><li>Help menu sebaiknya berubah sesuai kondisi pasien (kayak udah check up atau belum, kalau udah maka menu check up tidak tersedia)</li><li>Pas Logon Konsul, muncul menu konsul (Perlu akses menu dan help dulu)</li><li>Nama Pasien, Tujuan dan Obat belum sesuai kreatifitas masing-masing.</li><li>Laporan Pengacara (PENTING!)</li><li>Load, Register, Logout.</li></ul>	
Tindak Lanjut	
A. Mulai Pengembangan, dan pembagian tugas berdasarkan milestone	
Dokumentasi	

Form MoM Asistensi Tugas Besar  
IF1210/Algoritma dan Pemrograman 1  
Sesi. 2 2024/2025



Form MoM Asisten Tugas Besar  
IF1210/Algoritma dan Pemrograman 1  
Sem. 2 2024/2025

Nomor Asisten	: 1							
No. Kelompok/Kelas	: I/K01							
Tanggal asistensi	: 09 Mei 2025							
Anggota kelompok	<table border="1"> <tr><td>NIM / Nama (Hanya yang Hadir)</td></tr> <tr><td>1 13524131 / Amanda Aurelia Salsabilla</td></tr> <tr><td>2 13524065 / Kurt Michael Purba</td></tr> <tr><td>3 13524045 / Ahmad Zaky Robbani</td></tr> <tr><td>4 13524113 / Fauzan Mohamad Abdul Ghani</td></tr> <tr><td>5 13524079 / Angelina Andra Alanna</td></tr> <tr><td>6 -</td></tr> </table>	NIM / Nama (Hanya yang Hadir)	1 13524131 / Amanda Aurelia Salsabilla	2 13524065 / Kurt Michael Purba	3 13524045 / Ahmad Zaky Robbani	4 13524113 / Fauzan Mohamad Abdul Ghani	5 13524079 / Angelina Andra Alanna	6 -
NIM / Nama (Hanya yang Hadir)								
1 13524131 / Amanda Aurelia Salsabilla								
2 13524065 / Kurt Michael Purba								
3 13524045 / Ahmad Zaky Robbani								
4 13524113 / Fauzan Mohamad Abdul Ghani								
5 13524079 / Angelina Andra Alanna								
6 -								
Asisten pembimbing	<table border="1"> <tr><td>NIM / Nama</td></tr> <tr><td>13522164/Valentino Chrylic Triad</td></tr> </table>	NIM / Nama	13522164/Valentino Chrylic Triad					
NIM / Nama								
13522164/Valentino Chrylic Triad								
Catatan Asistensi:								
Rangkuman Diskusi	<ul style="list-style-type: none"> <li>Kedua ayah bebas, gak butuh manager doang, jumlah manager bebas.</li> <li>Quese bisa cuangan unlimited, kapasitas dalam cuangan tentu sendiri, diagnosis hanya bisa satu-satu walaupun dalam cuangan banyak orang. Didiat dari ucitaannya, pokoknya tetep yang paling pertama yang dilakukan.</li> <li>Makefile gak harus sekedar, tapi nanti harus bisa decompile pakai makefile.</li> <li>Isi config boleh gaes, dengan syarat diberikan penjelasan struktur config.</li> </ul>							
Tindak Lanjut	<p>Melanjutkan Pengajaran, dengan pemahaman yang lebih baik.</p>							
Dokumentasi								



Form MoM Asisten Tugas Besar  
IF1210/Algoritma dan Pemrograman 1  
Sem. 2 2024/2025

Nomor Asisten	: 02							
No. Kelompok/Kelas	: K-I/K01							
Tanggal asistensi	: 20/05/2025							
Anggota kelompok	<table border="1"> <tr><td>NIM / Nama (Hanya yang Hadir)</td></tr> <tr><td>1 13524131 / Amanda Aurelia Salsabilla</td></tr> <tr><td>2 13524065 / Kurt Michael Purba</td></tr> <tr><td>3 13524045 / Ahmad Zaky Robbani</td></tr> <tr><td>4 13524113 / Fauzan Mohamad Abdul Ghani</td></tr> <tr><td>5 13524079 / Angelina Andra Alanna</td></tr> <tr><td>6 -</td></tr> </table>	NIM / Nama (Hanya yang Hadir)	1 13524131 / Amanda Aurelia Salsabilla	2 13524065 / Kurt Michael Purba	3 13524045 / Ahmad Zaky Robbani	4 13524113 / Fauzan Mohamad Abdul Ghani	5 13524079 / Angelina Andra Alanna	6 -
NIM / Nama (Hanya yang Hadir)								
1 13524131 / Amanda Aurelia Salsabilla								
2 13524065 / Kurt Michael Purba								
3 13524045 / Ahmad Zaky Robbani								
4 13524113 / Fauzan Mohamad Abdul Ghani								
5 13524079 / Angelina Andra Alanna								
6 -								
Asisten pembimbing	<table border="1"> <tr><td>NIM / Nama</td></tr> <tr><td>13522164/Valentino Chrylic Triad</td></tr> </table>	NIM / Nama	13522164/Valentino Chrylic Triad					
NIM / Nama								
13522164/Valentino Chrylic Triad								
Catatan Asistensi:								
Rangkuman Diskusi	<ol style="list-style-type: none"> <li>1 minggu sebelum deadline semuanya harus sudah terintegrasi.</li> <li>ID termasuk bertambah, walau sudah ada pasien yang sembuh. Id dia tidak akan di replace dengan orang baru.</li> <li>Lebih enak pakai stack dan queue untuk diagnosis, dengan key value nya itu obatnya</li> <li>Daftar Check up tidak harus menggunakan map</li> <li>Ketika sudah ada include a di include b, maka di main hanya perlu include b dengan syarat sudah menggunakan ifndef</li> <li>Kasih tau dulu kalau ada yang salah, walaupun masih ada obat yang belum diminum</li> </ol>							
Tindak Lanjut	<ol style="list-style-type: none"> <li>1. Melanjutkan <del>Progress</del> dengan pemahaman yang lebih lengkap</li> </ol>							
Dokumentasi								

