

**LAPORAN TUGAS BESAR**  
**MATA KULIAH IF1210 ALGORITMA DAN**  
**PEMROGRAMAN 1**  
**PROGRAM RUMAH SAKIT NIMON**

Dosen Pengampu : Dr. Riza Satria Perdana, S. T, M.T.

Kelas / Kelompok : 03 / A



Disusun oleh:

Leticia Aldina Trulykinanti	(18223108)
Laras Hati Mahendra	(18223118)
Rezky M. Hafiz Batubara	(18224033)
Ellaine Juvina	(18224071)
Muhammad Afif H.	(18224099)

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

**2025**

## HALAMAN PERNYATAAN

*"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."*

Yang mengeluarkan pernyataan,

Leticia Aldina Trulykinanti [18223108]

Laras Hati Mahendra [18223118]

Rezky Muhammad Hafiz Batubara [18224033]

Ellaine Juvina [18224071]

Muhammad Afif Habiburrahman [18224099 ]

## DAFTAR ISI

LAPORAN TUGAS BESAR.....	1
HALAMAN PERNYATAAN.....	2
DAFTAR ISI.....	3
DAFTAR TABEL.....	4
DAFTAR GAMBAR.....	5
A. DESKRIPSI PERSOALAN.....	7
B. RENCANA IMPLEMENTASI.....	8
C. DAFTAR PEMBAGIAN KERJA KELOMPOK.....	9
D. CHECKLIST HASIL Pengerjaan Tugas Besar.....	13
E. DESAIN PERINTAH.....	14
F. DESAIN KAMUS DATA.....	23
G. DESAIN DEKOMPOSISI ALGORITMIK DAN FUNGSIONAL PROGRAM.....	25
H. SPESIFIKASI FUNGSIONAL PROGRAM.....	35
I. LAMPIRAN SOURCE CODE.....	45
II. LAMPIRAN HASIL PENGUJIAN PROGRAM.....	63
III. LAMPIRAN ASISTENSI.....	68

## **DAFTAR TABEL**

Tabel 1 Daftar ADT.....	8
Tabel 2 Daftar Pembagian Kerja Kelompok.....	9
Tabel 3 Daftar Pembagian Pembuatan Laporan.....	11
Tabel 4 Checklist Hasil Pengerjaan Tugas Besar.....	12

## DAFTAR GAMBAR

<b>Gambar 1 Nimons</b>	<b>7</b>
<b>Gambar 2 Flowchart login</b>	<b>25</b>
<b>Gambar 3 Register</b>	<b>27</b>
<b>Gambar 4 Flowchart Logout</b>	<b>27</b>
<b>Gambar 5 Flowchart Lupa Password</b>	<b>28</b>
<b>Gambar 6 Flowchart Menu &amp; Help</b>	<b>29</b>
<b>Gambar 7 Flowchart Denah Rumah Sakit</b>	<b>30</b>
<b>Gambar 8 Flowchart Lihat User</b>	<b>30</b>
<b>Gambar 9 Flowchart Cari User</b>	<b>30</b>
<b>Gambar 10 Flowchart Lihhat Antrian</b>	<b>31</b>
<b>Gambar 11 Flowchart Tambah Dokter</b>	<b>34</b>
<b>Gambar 12 Flowchart Diagnosis</b>	<b>34</b>
<b>Gambar 13 Flowchart Ngobatin</b>	<b>34</b>
<b>Gambar 14 Flowchart Aku boleh pulang ga, dok 🥺?</b>	<b>34</b>
<b>Gambar 15 Flowchart Daftar Check-Up</b>	<b>34</b>
<b>Gambar 16 Flowchart Antrian Saya!</b>	<b>34</b>
<b>Gambar 17 Flowchart Minum Obat</b>	<b>34</b>
<b>Gambar 18 Flowchart Minum Penawar</b>	<b>34</b>
<b>Gambar 19 Flowchart Exit</b>	<b>35</b>
<b>Gambar 20 Source code struktur data user.h</b>	<b>46</b>
<b>Gambar 21 Source code header ADT list_user.h</b>	<b>47</b>
<b>Gambar 22 Source code implementasi ADT list_user.h</b>	<b>49</b>
<b>Gambar 24 Source code implementasi ADT set.h</b>	<b>51</b>
<b>Gambar 25 Source code F01: login.h</b>	<b>52</b>
<b>Gambar 26 Source code F01: login.c</b>	<b>53</b>
<b>Gambar 27 Source code F02: register.h</b>	<b>54</b>
<b>Gambar 28 Source code F02: register.c</b>	<b>55</b>
<b>Gambar 29 Source code F03: logout.h</b>	<b>56</b>
<b>Gambar 30 Source code F03: logout.c</b>	<b>56</b>
<b>Gambar 31 Source code F04: lupa_password.h</b>	<b>56</b>
<b>Gambar 32 Source code F04: lupa_password.c</b>	<b>57</b>
<b>Gambar 33 Source code F05: help.h</b>	<b>59</b>
<b>Gambar 34 Source code F05: help.c</b>	<b>60</b>
<b>Gambar 27 Source code F06: lihat_denah.h</b>	<b>61</b>
<b>Gambar 27 Source code F06: lihat_denah.c</b>	<b>62</b>
<b>Gambar 28 Source code F10: f10.h</b>	<b>63</b>
<b>Gambar 29 Source code F10: f10.c</b>	<b>63</b>
<b>Gambar 30 Source code F18: exit.h</b>	<b>63</b>

<b>Gambar 31 Source code F18: exit.c</b>	<b>64</b>
<b>Gambar 32 Source code F15</b>	<b>64</b>
<b>Gambar 33 Source code F15</b>	<b>64</b>
<b>Gambar 34 Source code F15</b>	<b>64</b>
<b>Gambar 35 Source code F15</b>	<b>64</b>
<b>Gambar 36 Hasil pengujian F01-Login</b>	<b>65</b>
<b>Gambar 37 Hasil pengujian F02-Register</b>	<b>65</b>
<b>Gambar 38 Hasil pengujian F03-Logout</b>	<b>66</b>
<b>Gambar 39 Hasil pengujian F04-Lupa Password</b>	<b>66</b>
<b>Gambar 40 Hasil pengujian F05-menu dan help</b>	<b>67</b>
<b>Gambar 41 Hasil pengujian F06-</b>	<b>68</b>
<b>Gambar 42 Hasil pengujian F07-</b>	<b>68</b>
<b>Gambar 43 Hasil pengujian F10-</b>	<b>68</b>
<b>Gambar 44 Hasil pengujian F11 -</b>	<b>68</b>
<b>Gambar 45 Hasil pengujian F12 -</b>	<b>68</b>
<b>Gambar 46 Hasil pengujian F15 -</b>	<b>68</b>
<b>Gambar 47 Form asistensi pertama</b>	<b>71</b>
<b>Gambar 48 Form asistensi pertama</b>	<b>71</b>

## A. DESKRIPSI PERSOALAN



**Gambar 1** Nimons

Tugas Besar ini merupakan tugas untuk membuat sebuah program simulasi sistem informasi rumah sakit yang melibatkan sejumlah tokoh, yaitu pasien, dokter, dan manajer rumah sakit sebagai aktor utama dalam sistem (plot story). Program ini bertujuan untuk menangani berbagai aktivitas dalam lingkungan rumah sakit seperti registrasi pasien, login, logout, pengelolaan data pengguna, dan aktivitas medis lainnya.

Terdapat 16 fungsi wajib yang harus diimplementasikan, mencakup operasi dasar seperti registrasi, login, melihat profil, pencatatan medis, serta manajemen pengguna dan data. Selain itu, tersedia pula 5 fungsi bonus yang dapat dikerjakan untuk menambah nilai, seperti fitur pencarian lanjutan, statistik, atau fitur keamanan tambahan seperti enkripsi password. Dalam proses pengembangan program ini, kami menggunakan bahasa pemrograman C dan memanfaatkan file CSV sebagai basis penyimpanan data permanen.

Namun, terdapat beberapa batasan dalam pembuatan program ini. Kami tidak diperkenankan menggunakan library eksternal dalam bentuk apa pun (selain library standar C seperti `stdio.h`, `string.h`, `dll`), tidak diperbolehkan membuat file CSV sementara (temporary), serta diharuskan mematuhi struktur modular sesuai dengan pembagian file header (.h) dan source (.c). Semua fungsi wajib harus diimplementasikan sendiri tanpa meng

## B. RENCANA IMPLEMENTASI

Tabel 1 Daftar ADT

ADT	Fitur Terkait	Deskripsi Implementasi	Alasan Penggunaan
<b>ADT List (Array of Struct)</b>	F01 - Login, F02 - Register Pasien, F07 - Lihat User, F08 - Cari User, F10 - Tambah Dokter	Menyimpan seluruh user (dokter, pasien, manager) dalam array statis yang ukurannya dibatasi.	Akses cepat dengan indeks dan sederhana untuk pencarian berurutan dan sorting.
<b>ADT Set</b>	F02 - Register Pasien, F10 - Tambah Dokter	Menyimpan username dalam struktur set (tanpa duplikat, case-insensitive).	Untuk validasi keunikan username yang sensitif terhadap duplikasi (case-insensitive).
<b>ADT Stack</b>	F16 - Minum Obat, F17 - Minum Penawar, F13 - Pulang Dok	Stack digunakan untuk menyimpan urutan obat yang diminum oleh pasien, karena urutan konsumsi penting.	Stack cocok untuk model LIFO (Last-In First-Out), di mana obat terakhir yang diminum bisa dikeluarkan lebih dulu (F17).
<b>ADT Queue</b>	F06 - Denah Rumah Sakit (ruangan), F09 - Lihat Antrian, F14 - Daftar Check-up	Queue digunakan untuk mengatur antrian pasien yang mendaftar ke dokter secara FIFO.	Mengatur alur pasien secara adil berdasarkan urutan pendaftaran.
<b>ADT Linked List</b>	F14 - Daftar Check-up, F15 - Antrian Saya!	Mewakili antrian per ruangan dokter, terhubung antar node pasien.	Digunakan karena lebih fleksibel dibanding array, terutama jika jumlah pasien dinamis.
<b>ADT Map</b>	F11 - Diagnosis, F12 - Ngobatin	Menyimpan relasi penyakit → daftar obat dan pasien → kondisi checkup.	Efisien untuk mapping penyakit ke obat dan lookup data pasien.



## C. DAFTAR PEMBAGIAN KERJA KELOMPOK

**Tabel 2 Daftar Pembagian Kerja Kelompok**

<b>Fitur</b>	<b>Implementasi</b>	<b>NIM Desainer</b>	<b>NIM Coder</b>	<b>NIM Tester</b>
F00 - Rencana Implementasi	ADT List, ADT Set, ADT Stack, ADT Queue, ADT Linked List, ADT Map	18224099	18224099	1822409.9
F01 - Login	Dibuat prosedur login_system untuk memverifikasi username dan password, serta menetapkan current_user sesuai data yang valid di sistem.	18223118	18223118	18223118 18224099
F02 - Register Pasien	Membuat fungsi pendukung is_username_unique yang berfungsi untuk melakukan pemeriksaan keunikan username dan membuat prosedur register_pasien yang akan menghubungkan proses input data dari pengguna.	18223118	18223118	18223118 18224099
F03 - Logout	Membuat prosedur logout_system yang akan menghubungkan kondisi status login pengguna dengan prosedur keluar dari akun secara sistematis.	18223118	18223118	18223118 18224099
F04 - Lupa Password	Membuat prosedur lupa_password_system yang akan menghubungkan proses verifikasi identitas pengguna dengan proses mereset password dan prosedur generate kode unik	18223118	18223118	18223118 18224099
F05 - Menu & Help	Dibuat prosedur `help_system` untuk menampilkan daftar command sesuai peran pengguna. Jika belum login, hanya command dasar seperti	18223118	18223118	18223118 18224099

	login dan register yang ditampilkan.			
F06 - Denah Rumah Sakit	Dibuat prosedur lihatdenah untuk menampilkan daftar ruangan dalam bentuk tabel, dan lihat_ruangan untuk menampilkan detail isi ruangan tertentu sesuai nomor yang dipilih.	18224033	18224033	18224033 18224099
F07 - Lihat User		18224033	18224033	18224033
F08 - Cari User		18224033	18224033	18224033
F09 - Lihat Antrian				
F10 - Tambah Dokter	Dibuat prosedur tambah_dokter untuk menambahkan akun dokter baru oleh Manager, serta prosedur assign_dokter untuk menetapkan dokter ke ruangan tertentu.	18224033	18224033	18224033 18224099
F11 - Diagnosis		18224071	18224071	18224071
F12 - Ngobatin				
F13 - Aku boleh pulang ga dok?		18223108	18223108	18223108
F14 - Daftar Check-up				
F15 - Antrian Saya				

F16 - Minum Obat		18223108	18223108	18223108
F17 - Minum Penawar		18223108	18223108	18223108
F18 - Exit	Dibuat prosedur exit_system yang menanyakan konfirmasi pengguna untuk keluar dari sistem dan menghentikan program jika disetujui.	18223108	18223108	18223108 18224099

**Tabel 3 Daftar Pembagian Pembuatan Laporan**

No.	Bagian Laporan	NIM
1	Halaman Cover	18223108
2	Daftar Isi	18223108, 18223118
3	Daftar Tabel	18223108, 18223118
4	Daftar Gambar	18223108
5	Deskripsi Persoalan	18223118
6	Daftar Pembagian Tugas	18223108, 18223118, 18224033, 18224071, 18224099
7	Checklist Hasil Rangkaian, Implementasi, dan Uji Coba	18223108, 18223118, 18224033, 18224099
8	Desain Perintah	18223118, 18224033
9	Desain Kamus Data	18223118, 18224099, 18224033
10	Desain Dekomposisi Algoritmik dan Fungsional Program	18224099, 18223118, 18223108, 18224033
11	Spesifikasi	18223108, 18223118, 18224033, 18224071, 18224099
12	Hasil Pengujian Program	18223118, 18224099

13	Catatan dan Lampiran	18223118, 18224099
----	----------------------	--------------------

## D. CHECKLIST HASIL Pengerjaan Tugas Besar

Tabel 4 Checklist Hasil Pengerjaan Tugas Besar

Fitur	Desain	Implementasi	Testing
F01 - Register	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F02 - Login	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F03 - Logout	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F04 - Lupa Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F05 - Menu & Help	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F06 - Denah Rumah Sakit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F07 - Lihat User	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F08 - Cari User 26	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F09 - Lihat Antrian	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F10 - Tambah Dokter	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
F11 - Diagnosis	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F12 - Ngobatin	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F13 - Aku boleh pulang ga, dok 🥺?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F14 - Daftar Check-Up	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F15 - Antrian Saya!	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F16 - Minum Obat	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F17 - Minum Penawar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F18 - Exit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## E. DESAIN PERINTAH

### 1. F01 - Login

#### **PROCEDURE login\_system()**

{ IS: Tidak ada user yang login. User memasukkan username dan password. }  
{ FS: Jika username dan password cocok dengan data sistem, user berhasil login dan diberikan sambutan berdasarkan peran.  
    Jika username tidak terdaftar, sistem menampilkan pesan bahwa user tidak terdaftar sebagai Manager, Dokter, atau Pasien  
    Jika password salah, sistem menampilkan pesan kegagalan login karena kesalahan password dan akan kembali tanpa melanjutkan login. }

# User berhasil login dengan username nimonsslatte dan password yang sesuai  
# **DISCLAIMER:** Tampilan/ interface dibebaskan, berikut hanya contoh saja

# Kasus 1: Login sebagai Manager

# **Masukkan**

>>> **LOGIN**

Username: **nimonsslatte**

Password: **nimonatutgajah23**

# Sistem berhasil menyambut user dengan ucapan selamat pagi menggunakan username nimonsslatte yang berhasil login.  
Selamat pagi Manager nimonsslatte!

# **Keluaran**

# User diberikan deskripsi dengan menggunakan perintah “help” untuk melihat daftar perintah yang tertera.

# User berhasil login dengan username Neroifa dan password yang sesuai

# Kasus 2: Login sebagai Dokter

# **Masukkan**

>>> **LOGIN**

Username: **Neroifa**

Password: **Neroifa123**

# Sistem berhasil menyambut user dengan ucapan selamat pagi menggunakan username Neroifa yang berhasil login.  
Selamat pagi Dokter Neroifa!

# **Keluaran**

# User diberikan deskripsi dengan menggunakan perintah “help” untuk melihat daftar perintah yang tertera.

# User berhasil login dengan username GRO dan password yang sesuai

# Kasus 3: Login sebagai Pasien

# **Masukkan**

>>> **LOGIN**

Username: **GRO**

Password: **NeroifaCantik**

# Sistem berhasil menyambut user dengan ucapan selamat pagi menggunakan username GRO yang berhasil login.  
Selamat pagi GRO! Ada keluhan apa ?

# **Keluaran**

# User diberikan deskripsi dengan menggunakan perintah “help” untuk melihat daftar perintah yang tertera.

# User gagal login dengan username **NONEXISTENT\_USE**

# Kasus 4: Tidak ada username yang terdaftar

# **Masukkan**

>>> **LOGIN**

Masukan username: **NONEXISTENT\_USER**

Masukan password: **passwordrandom**

# **Keluaran**

# User diberikan deskripsi mengenai pesan bahwa log in gagal karena user sudah log in dengan username tersebut sebab tidak ada manager yang bernama **NONEXISTENT\_USE**.

Tidak ada Manager, Dokter, atau pun Pasien yang bernama NONEXISTENT\_USER!

# User gagal login dengan username **nimonsganteng**

# Kasus 5: Kasus password salah

# **Masukkan**

>>> **LOGIN**

Masukan username: **nimonsganteng**

Masukan password: **wrongpassword**

# **Keluaran**

# User diberikan deskripsi mengenai pesan bahwa log in gagal karena user sudah log in dengan username tersebut sebab password salah.

Password salah untuk pengguna yang bernama nimonsganteng!

## 2. F02 - Register Pasien

### **PROCEDURE register\_pasien()**

{ IS: Program dalam keadaan aktif dan tidak ada user yang sedang login. User ingin melakukan pendaftaran sebagai pasien dengan memasukkan username dan password. }

{ FS: Jika username valid dan belum digunakan serta password benar, maka pasien ditambahkan dan sistem menampilkan pesan berhasil.

- Jika ada user yang sedang login, maka registrasi dibatalkan dan ditampilkan peringatan untuk logout terlebih dahulu.
- Jika username mengandung karakter non-alfabet, maka registrasi dibatalkan dan sistem menampilkan pesan kesalahan.
- Jika username sudah terdaftar, maka registrasi dibatalkan dan ditampilkan pesan bahwa nama sudah digunakan.
- Jika input tidak valid atau kapasitas penuh, maka registrasi tidak dilanjutkan dan sistem menampilkan pesan yang sesuai}

# Kasus 1: Pasien bernama Denis belum ada

# User melakukan registrasi dengan username “Denis” yang belum terdaftar sebelumnya.

# **Masukkan**

>>> **REGISTER**

Username: **Denis**

Password: **aditgimananihdiit**

# Sistem berhasil menambahkan pasien baru dengan nama Denis.

Pasien Denis berhasil ditambahkan!

# User kemudian melakukan login menggunakan username dan password yang telah didaftarkan.

# **Masukkan**

```

>>> LOGIN
Username: Denis
Password: aditgimananihdi

# Sistem menyambut user dengan sapaan selamat pagi sesuai peran pasien.
Selamat pagi Denis! Ada keluhan apa ?

# Pasien baru bernama Denis berhasil diregistrasikan dan dapat login dengan username
serta password yang sesuai.
# Sistem menyambut pasien yang berhasil login dan siap menampilkan daftar perintah
dengan perintah "help".

# Kasus 2: Pasien bernama Denis sudah ada.
# Masukkan
>>> REGISTER
Username: Denis
Password: aditgimananihdi

# Sistem menolak registrasi karna nama Denis sudah terdaftar sebagai pasien.
Registrasi gagal! Pasien dengan nama Denis sudah terdaftar.

# Keluaran
# Sistem membatalkan proses registrasi dan menampilkan pesan bahwa nama pasien sudah
digunakan.
# User tidak dapat mendaftarkan akun dengan username yang sama.

```

### 3. F03 - Logout

```

PROCEDURE logout_system()
{ IS: Program dalam keadaan berjalan, dan variabel current_user menunjuk ke user yang sedang login
(jika ada). }
{ FS: Jika ada user yang sedang login, maka user berhasil logout dan current_user diset menjadi NULL.
Jika belum login, maka sistem menampilkan pesan bahwa logout gagal dan meminta user login
terlebih dahulu. }

# DISCLAIMER: Tampilan/ interface dibebaskan, berikut hanya contoh saja
# Kasus 1: sedang dalam keadaan logged in
# User melakukan logout setelah berhasil login sebelumnya.
# Masukkan
>>> LOGOUT
# Keluar dari akun

Sampai jumpa!

# Keluaran
# User berhasil logout dan sistem menyambut dengan pesan "Sampai jumpa!".
# Setelah logout, user tidak lagi terhubung dengan sistem (current_user diset ke
NULL).

# Kasus 2: sedang dalam keadaan belum logged in
# Masukkan
>>> LOGOUT
Logout gagal!
Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout

# Keluaran

```



```
# Sistem menampilkan pesan bahwa user belum login dan meminta untuk login terlebih dahulu sebelum dapat logout.
```

#### 4. F04 - Lupa Password

##### **PROCEDURE lupa\_password()**

```
{ IS: Program dalam keadaan berjalan, dan variabel current_user menunjuk ke user yang sedang login (jika ada). }  
{ FS: - Jika username ditemukan dan kode unik yang dimasukkan cocok, maka user akan diminta untuk memasukkan password baru, dan password user berhasil diperbarui.  
      - Jika username tidak ditemukan, maka sistem menampilkan pesan bahwa username tidak terdaftar.  
      - Jika kode unik yang dimasukkan tidak sesuai dengan kode unik yang dihasilkan sistem, maka sistem membatalkan proses dan menampilkan pesan bahwa kode unik salah.  
      - Jika ada input yang tidak valid, sistem membatalkan proses dan menampilkan pesan kesalahan input. }
```

```
# Kasus 1: Manager, Dokter, atau Pasien bernama Jeffreey ada
```

##### **# Masukkan**

```
>>> LUPA_PASSWORD
```

```
Username: Jeffreey
```

```
Kode Unik: Je2fr2ey
```

```
Halo Dokter Jeffreey, silakan daftarkan ulang password anda!
```

```
Password Baru: JR1234
```

```
>>> LOGIN
```

```
Username: Jeffreey
```

```
Password: JR1234
```

```
Selamat pagi Dokter Jeffreey!
```

##### **# Keluaran**

```
# Sistem berhasil mengenali username, mencocokkan kode unik, dan meminta user untuk memasukkan password baru.
```

```
# Setelah berhasil, user dapat login menggunakan password yang baru.
```

```
# Kasus 2: Manager, Dokter, atau Pasien bernama NONEXISTENT_USER tidak ada
```

##### **# Masukkan**

```
>>> LUPA_PASSWORD
```

```
Username: NONEXISTENT_USER
```

```
Kode Unik: NONEXISTENT_USER
```

```
Username tidak terdaftar!
```

##### **# Keluaran**

```
# Sistem tidak menemukan username dalam data pengguna dan menampilkan pesan bahwa username tidak terdaftar.
```

```
# Kasus 3: Kode unik untuk username nimonsslatte bukan nimonsslatte tetapi adalah nimon2sla2te
```

```
>>> LUPA_PASSWORD
```

```
Username: nimonsslatte
```

```
Kode Unik: nimonsslatte
```

```
Kode unik salah!
```

##### **# Keluaran**

```
# Username ditemukan, namun kode unik tidak cocok sehingga proses ubah password
dibatalkan dengan pesan kesalahan.
```

## 5. F05 - Menu & Help

### PROCEDURE help\_system()

{ IS: Program dalam keadaan berjalan. Pengguna dapat belum login atau sudah login sebagai Manager, Dokter, atau Pasien. }

{ FS: Sistem menampilkan daftar perintah yang bisa digunakan sesuai dengan status login dan peran pengguna (jika sudah login).

- Jika belum login, sistem menampilkan instruksi untuk login atau register. }

# Kasus 1: belum dalam keadaan logged in

>>> **HELP**

===== HELP =====

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

1. LOGIN: Masuk ke dalam akun yang sudah terdaftar
2. REGISTER: Membuat akun baru

Footnote:

1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

# Kasus 2: Sudah login sebagai Dokter

>>> **HELP**

===== HELP =====

Halo Dokter **Neroifa**. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

1. LOGOUT: Keluar dari akun yang sedang digunakan
2. DIAGNOSIS: Melakukan diagnosis penyakit pasien berdasarkan kondisi tubuh pasien
3. # ...dan seterusnya

Footnote:

1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

# Kasus 3: Sudah login sebagai Pasien

>>> **HELP**

===== HELP =====

Selamat datang, **GRO**. Kamu memanggil command HELP. Kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

1. LOGOUT: Keluar dari akun yang sedang digunakan
2. DAFTAR\_CHECKUP: Mendaftarkan diri untuk pemeriksaan dokter
3. # ...dan seterusnya

Footnote:

1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid

```
# Kasus 4: Sudah login sebagai Dokter
>>> HELP

===== HELP =====

Halo Manager nimonslatte. Kenapa kamu memanggil command HELP? Kan kamu manager,
tapi yasudahlah kamu pasti sedang kebingungan. Berikut adalah hal-hal yang dapat
kamu lakukan sekarang:

1. LOGOUT: Keluar dari akun yang sedang digunakan
2. TAMBAH_DOKTER: Mendaftarkan dokter baru ke sistem
3. # ...dan seterusnya

Footnote:
1. Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar
2. Jangan lupa untuk memasukkan input yang valid
```

## 6. F06 - Denah Rumah Sakit

- Lihat Denah

```
PROCEDURE lihat_denah (Input: ListRuangan)
{ IS: Program dalam keadaan berjalan. Pengguna dapat belum login atau sudah login sebagai Manager,
Dokter, atau Pasien. }
{ FS: Sistem menampilkan daftar perintah yang bisa digunakan sesuai dengan status login dan peran
pengguna (jika sudah login).
- Jika belum login, sistem menampilkan instruksi untuk login atau register. }
```

```
>>> LIHAT_DENAH

+-----+-----+-----+-----+-----+-----+-----+
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
+-----+-----+-----+-----+-----+-----+-----+
```

- Lihat Ruangan

```
PROCEDURE lihat_ruangan (Input: ListRuangan, input: Integer)
{ IS: Program dalam keadaan berjalan. Pengguna dapat belum login atau sudah login sebagai Manager,
Dokter, atau Pasien. }
{ FS: Sistem menampilkan daftar perintah yang bisa digunakan sesuai dengan status login dan peran
pengguna (jika sudah login).
- Jika belum login, sistem menampilkan instruksi untuk login atau register. }
```

```
# Kasus 1: ruangan terdapat dokter dan pasien
>>> LIHAT_RUANGAN 1

--- Detail Ruangan 1 ---
Kapasitas : 3
Dokter    : Dr. Strange
Pasien di dalam ruangan:
  1. John Smith
  2. John Lennon
-----

# Kasus 2: ruangan terdapat dokter dan tidak ada pasien
```

```

>>> LIHAT_RUANGAN 2

--- Detail Ruangan 2 ---
Kapasitas : 3
Dokter    : Dr. Oz
Pasien di dalam ruangan:
    Tidak ada pasien di dalam ruangan saat ini.
-----

# Kasus 3: ruangan tidak terdapat dokter.
>>> LIHAT_RUANGAN 3

--- Detail Ruangan 3 ---
Kapasitas : 3
Dokter    : -
Pasien di dalam ruangan:
    Tidak ada pasien di dalam ruangan saat ini.
-----

```

F07 - Lihat User

F08 - Cari User

F09 - Lihat Antrian

10. F10 - Tambah Dokter

- Tambah Dokter

**PROCEDURE tambah\_dokter** (input/output: ListUser, input/output: Set)  
 { IS: Program dalam keadaan berjalan. Pengguna sudah login sebagai manajer. Manajer ingin menambahkan dokter dengan menginput username dan password dari dokter.}  
 { FS: Jika username dokter valid dan belum digunakan, maka dokter ditambahkan dan sistem menampilkan pesan berhasil.

- Jika pengguna masih belum login, maka sistem akan menampilkan pesan yang sesuai dan tidak akan bisa menambahkan dokter.
- Jika pengguna login bukan sebagai manajer, maka sistem akan menampilkan pesan akses ditolak dan tambah dokter akan dibatalkan.
- Jika username dokter mengandung karakter non-alfabet, maka tambah dokter dibatalkan dan sistem menampilkan pesan kesalahan.
- Jika username dokter sudah terdaftar, maka tambah dokter dibatalkan dan ditampilkan pesan bahwa nama sudah digunakan.

Jika input tidak valid atau kapasitas penuh, maka tambah dokter tidak dilanjutkan dan sistem menampilkan pesan yang sesuai}

```

# Kasus 1: Dokter bernama Budi belum ada
>>> TAMBAH_DOKTER
Username: Budi
Password: budi123

Dokter Budi berhasil ditambahkan!
>>> LOGOUT

Sampai jumpa!

>>> LOGIN

```

Username: <b>Budi</b> Password: <b>budi123</b>  Selamat pagi Dokter <b>Budi</b> !
# Kasus 2: Dokter bernama Budi sudah ada >>> <b>TAMBAH_DOKTER</b> Username: <b>Denis</b> Password: <b>denis123</b>  Sudah ada Dokter bernama <b>Denis</b> !
# Kasus 3: Pengguna Belum Login >>> <b>TAMBAH_DOKTER</b>  Tidak ada pengguna yang sedang login. Silahkan login terlebih dahulu.
# Kasus 4: Pengguna Bukan Manajer >>> <b>TAMBAH_DOKTER</b>  Akses ditolak. Anda bukan Manajer.

- **Assign Dokter**

<b>PROCEDURE assign_dokter</b> (input/output: ListUser, input/output: ListRuangan) { IS: Program dalam keadaan berjalan. Pengguna sudah login sebagai manajer. Manajer ingin melakukan assign ruangan ke dokter tertentu yang belum memiliki ruangan.} { FS: Jika username dokter valid, maka dokter akan diassign ke salah satu ruangan dan sistem menampilkan pesan berhasil. - Jika pengguna masih belum login atau login bukan sebagai manajer, maka sistem akan menampilkan pesan yang sesuai dan tidak akan bisa melakukan assign dokter. - Jika username dokter tidak ditemukan karena belum ada sebelumnya, maka assign dokter dibatalkan dan sistem menampilkan pesan kesalahan. - Jika dokter sudah menempati ruangan tertentu, maka assign dokter akan gagal dan sistem akan menampilkan pesan kesalahan. - Jika ruangan tertentu sudah ditempati dokter lain, maka assign dokter akan gagal dan sistem akan menampilkan pesan kesalahan }
# Kasus 1: Dokter bernama Budi belum ada # Kasus 1: Ruangan Kosong dan dokter belum di assign di ruang manapun >>> <b>ASSIGN_DOKTER</b> Username: <b>Budi</b> Ruangan: <b>1</b>  Dokter <b>Budi</b> berhasil diassign ke ruangan <b>1</b> !
# Kasus 2: Ruangan Kosong dan dokter sudah di assign di ruang lain >>> <b>ASSIGN_DOKTER</b> Username: <b>Budi</b> Ruangan: <b>2</b>  Dokter <b>Budi</b> sudah diassign ke ruangan <b>1</b> !
# Kasus 3: Ruangan tidak kosong dan dokter belum di assign di ruang manapun >>> <b>ASSIGN_DOKTER</b> Username: <b>Adi</b> Ruangan: <b>11</b>

Dokter **Budi** sudah menempati ruangan 1!  
Silakan cari ruangan lain untuk dokter **Adi**.

```
# Kasus 4: Ruangan tidak kosong dan dokter sudah di assign di ruang lain
>>> ASSIGN_DOKTER
Username: Ahuy
Ruangan: 1
```

Dokter **Ahuy** sudah menempati ruangan 3!  
Ruangan 1 juga sudah ditempati dokter **Budi**!

F11 - Diagnosis

F12 - Ngobatin

F13 - Aku boleh pulang ga, dok?

F14 - Daftar Check Up

F15 - Antrian Saya

F16 - Minum Obat

F17 - Minum Penawar

18. F18 - Exit

#### **PROCEDURE exit()**

{ IS: Program sedang berjalan. Pengguna diminta untuk memilih apakah ingin menyimpan perubahan sebelum keluar (y/n). }

{ FS: Jika input 'y' atau 'Y', sistem akan menjalankan prosedur penyimpanan dan keluar dari program.  
Jika input 'n' atau 'N', program langsung keluar tanpa menyimpan.

Jika input tidak valid, pengguna akan diminta untuk mengulang hingga memasukkan input yang valid ('y' atau 'n'). }

```
# Kasus 1: User ingin keluar dari program
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n) y
```

```
# Keluar program
```

```
# Kasus 2: User tidak keluar dari program
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n) n
```

```
# Menjalankan fungsi-fungsi lain dan tidak keluar dari program
```

## F. DESAIN KAMUS DATA

### 1. F01 - Login

**KAMUS LOKAL**

Login, found\_user, valid\_password : Boolean  
username, password, role : String  
i : Integer

### 2. F02 - Register Pasien

**KAMUS LOKAL**

is\_valid\_input, is\_alpha\_only, is\_username\_unique: Boolean  
username, password : String

### 3. F03 – Logout

**KAMUS LOKAL**

is\_logged\_in : Boolean

### 4. F04 – Lupa Password

**KAMUS LOKAL**

username, kode\_unik\_input, kode\_unik\_asli, password\_baru : String  
i : integer  
Is\_valid\_input, is\_kode\_sesuai : Boolean

### 5. F05 – Menu & Help

**KAMUS LOKAL**

username : String

### 6. F06 – Denah Rumah Sakit

**KAMUS LOKAL**

r : pointer to Ruangan  
i : integer  
global ruangan : ListRuangan  
global current\_user : pointer to User

### 7. F07 – Lihat User

--

8. F08 – Cari User

9. F09 – Lihat Antrian

10. F10 - Tambah Dokter

**KAMUS LOKAL**  
username, password : String  
nomor\_ruangan, ruangan\_idx : integer  
dokter\_baru : User  
dokter\_found, ruangan\_found : boolean  
global current\_user : pointer to User  
global users : ListUser  
global ruangan : ListRuangan

11. F11 - Diagnosis

12. F12 - Ngobatin

13. F13 - Aku boleh pulang ga, dok 🥺?

14. F14 - Daftar Check-Up

15. F15 - Antrian Saya!

16. F16 - Minum Obat

17. F17 - Minum Penawar

18. F18 - Exit



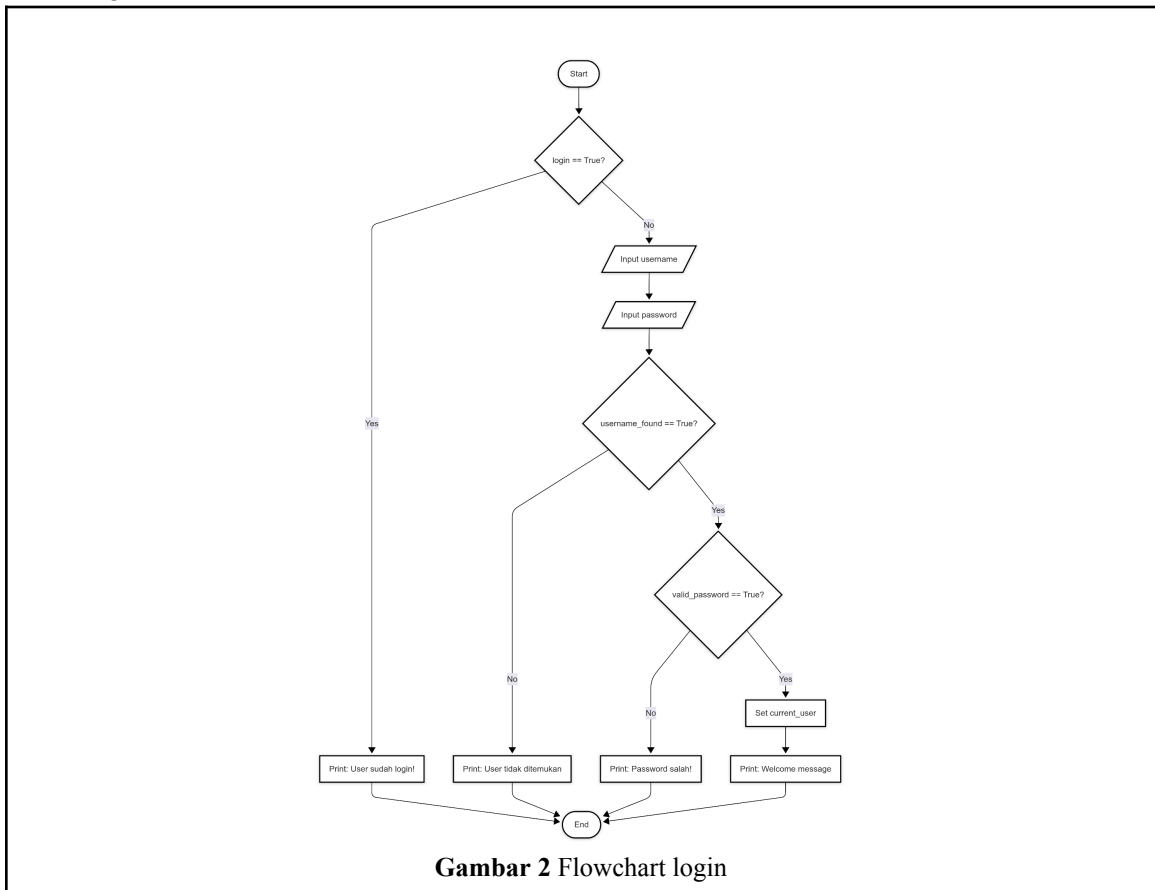
### KAMUS LOKAL

username: String

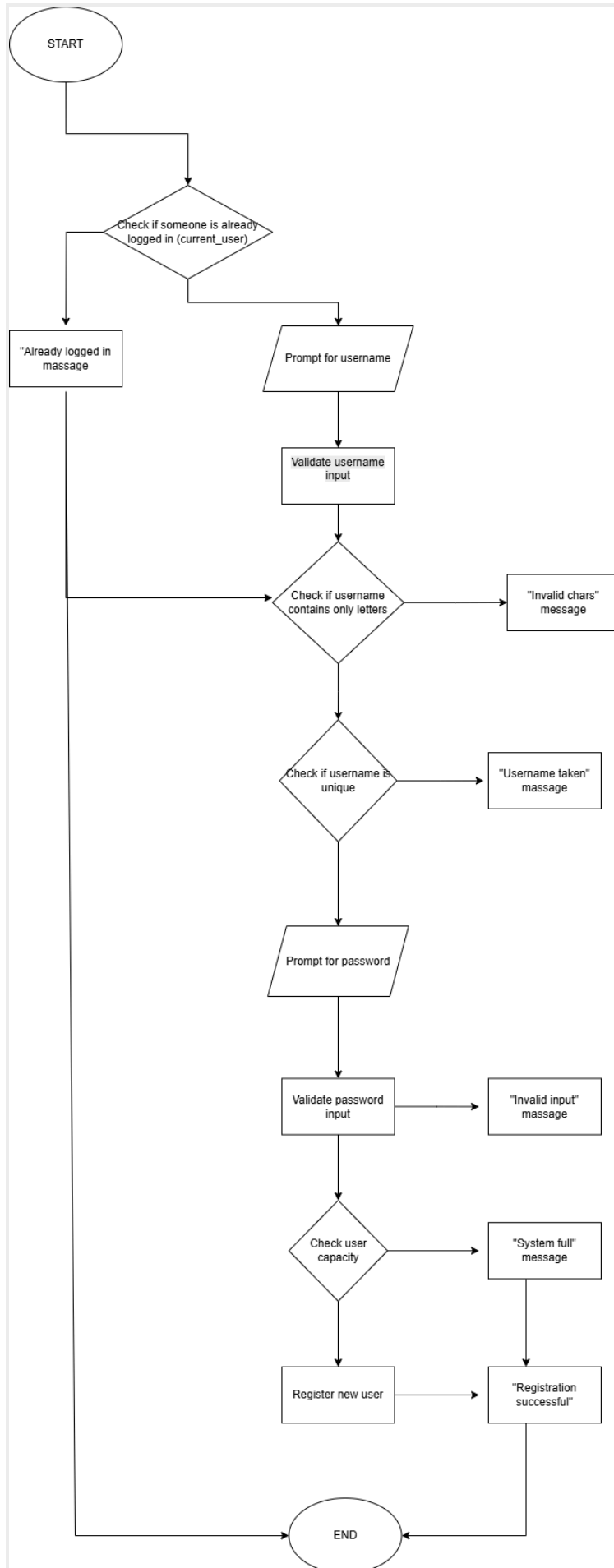
input: Character

## G. DESAIN DEKOMPOSISI ALGORITMIK DAN FUNGSIONAL PROGRAM

### 1. F01 - Login

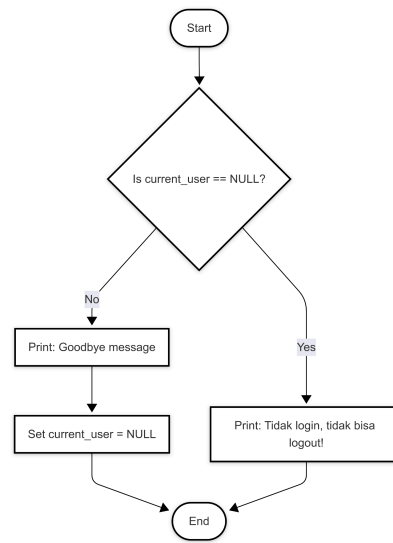


### 2. F02 - Register Pasien



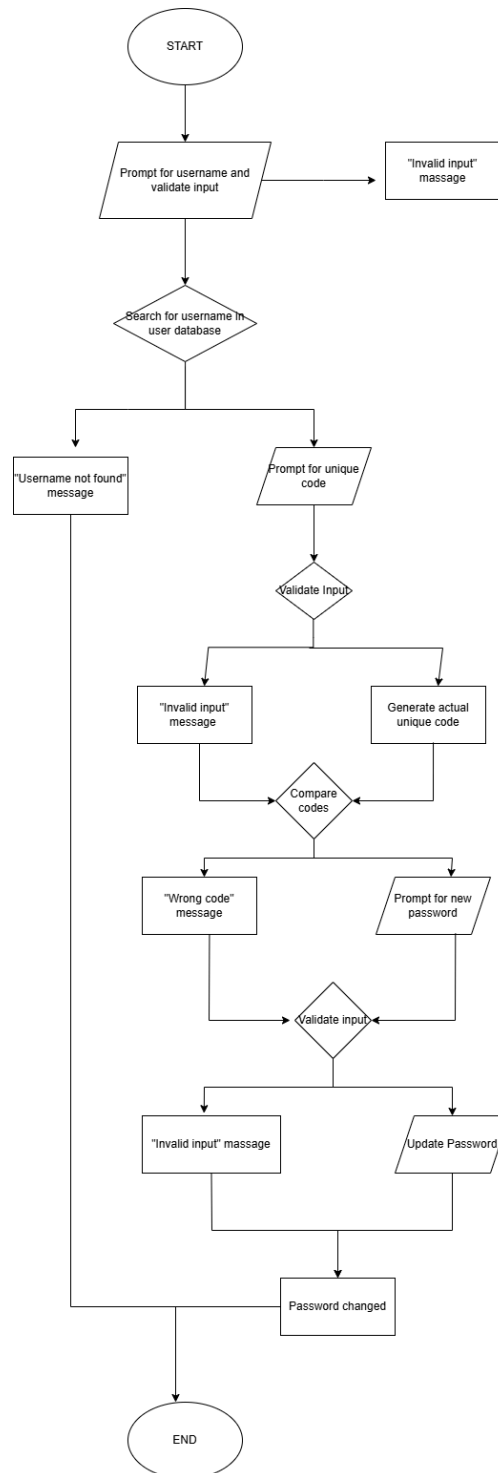
**Gambar 3 Register**

3. F03 - Logout

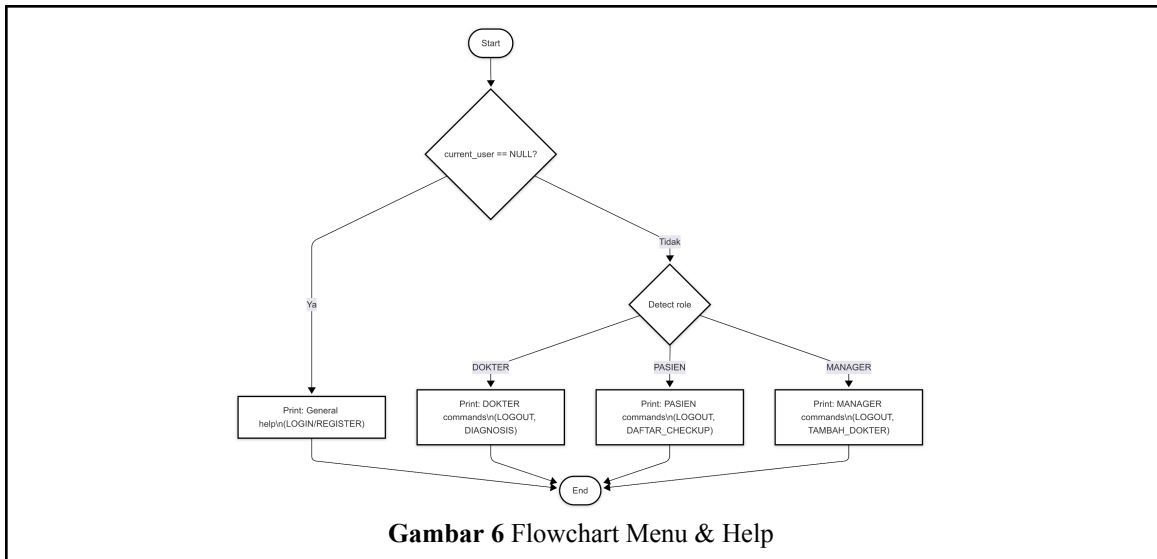


**Gambar 4 Flowchart Logout**

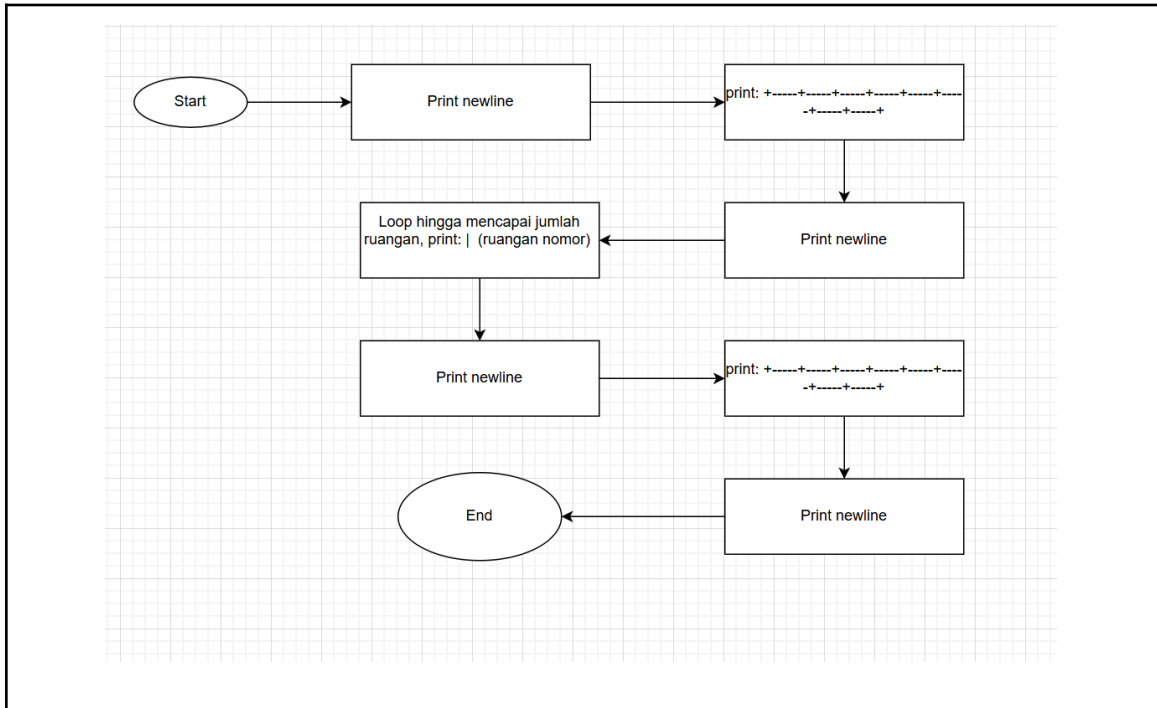
4. F04 - Lupa Password

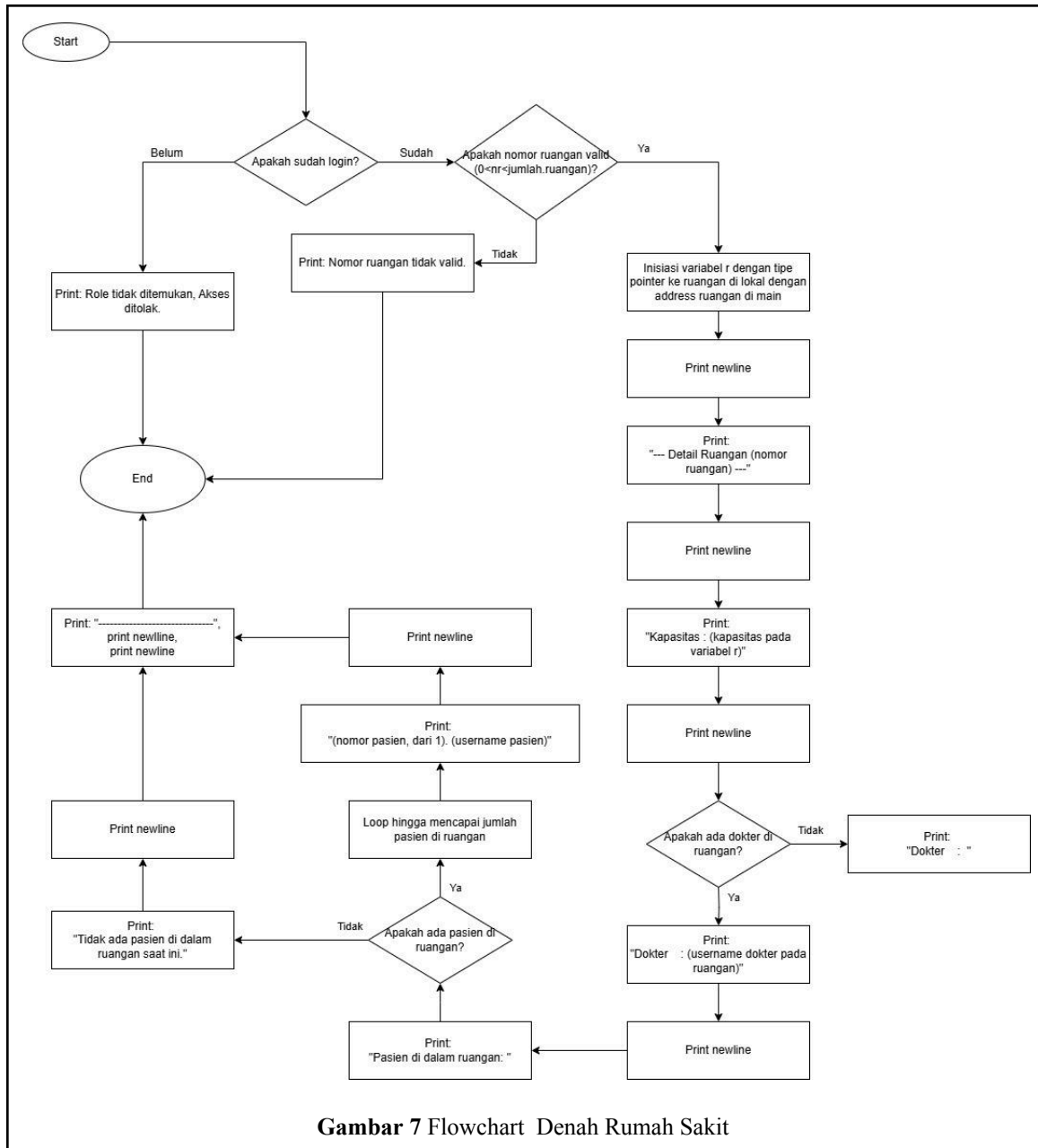


**Gambar 5** Flowchart Lupa Password

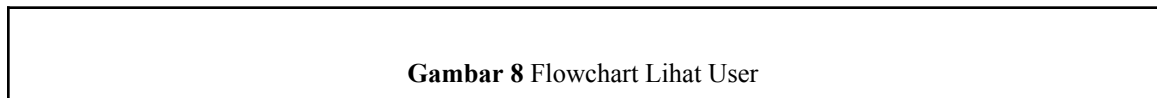


## 6. F06 - Denah Rumah Sakit

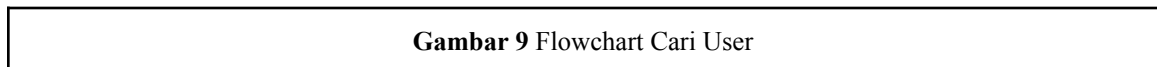




7. F07 - Lihat User



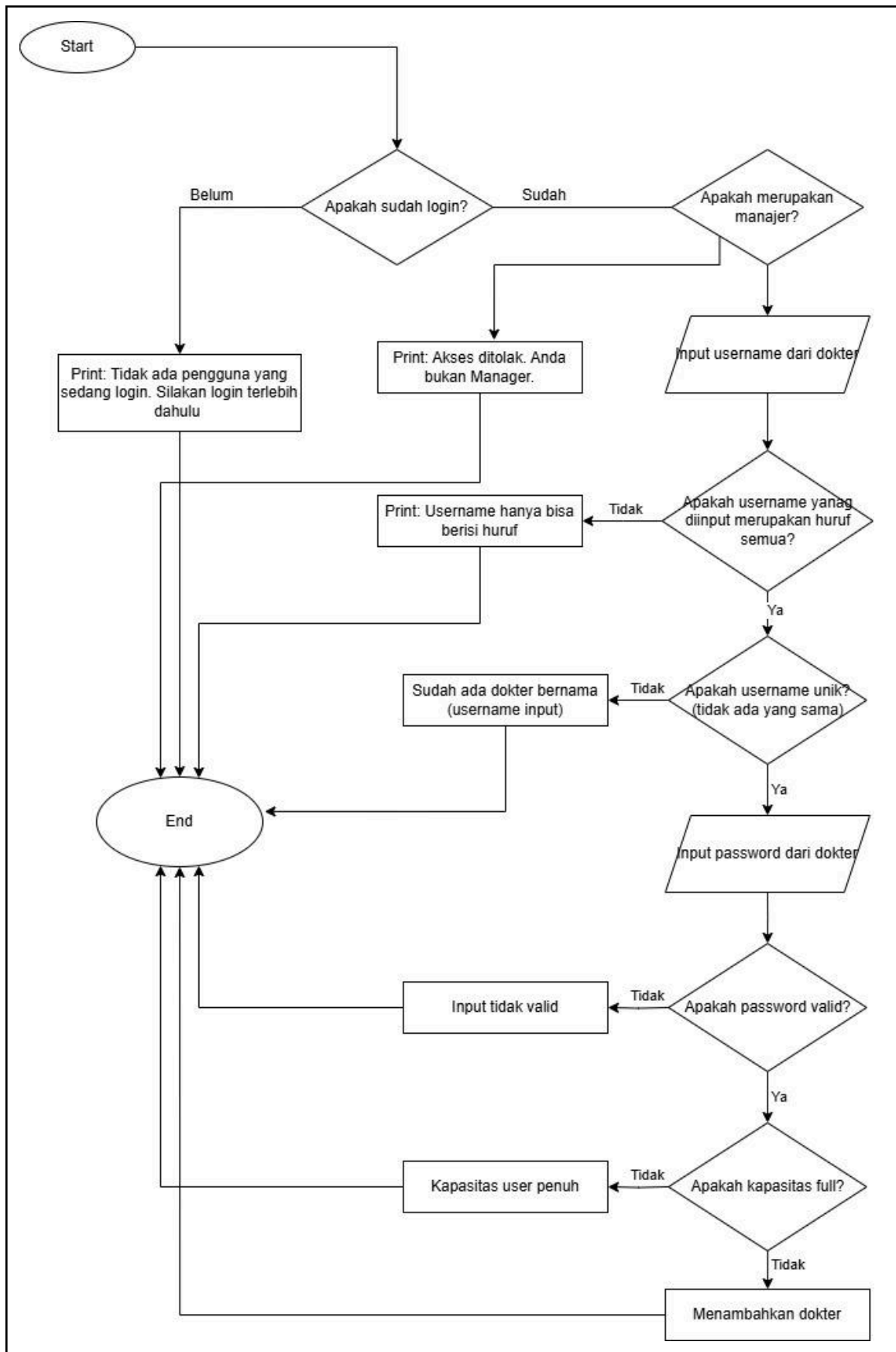
8. F08 - Cari User



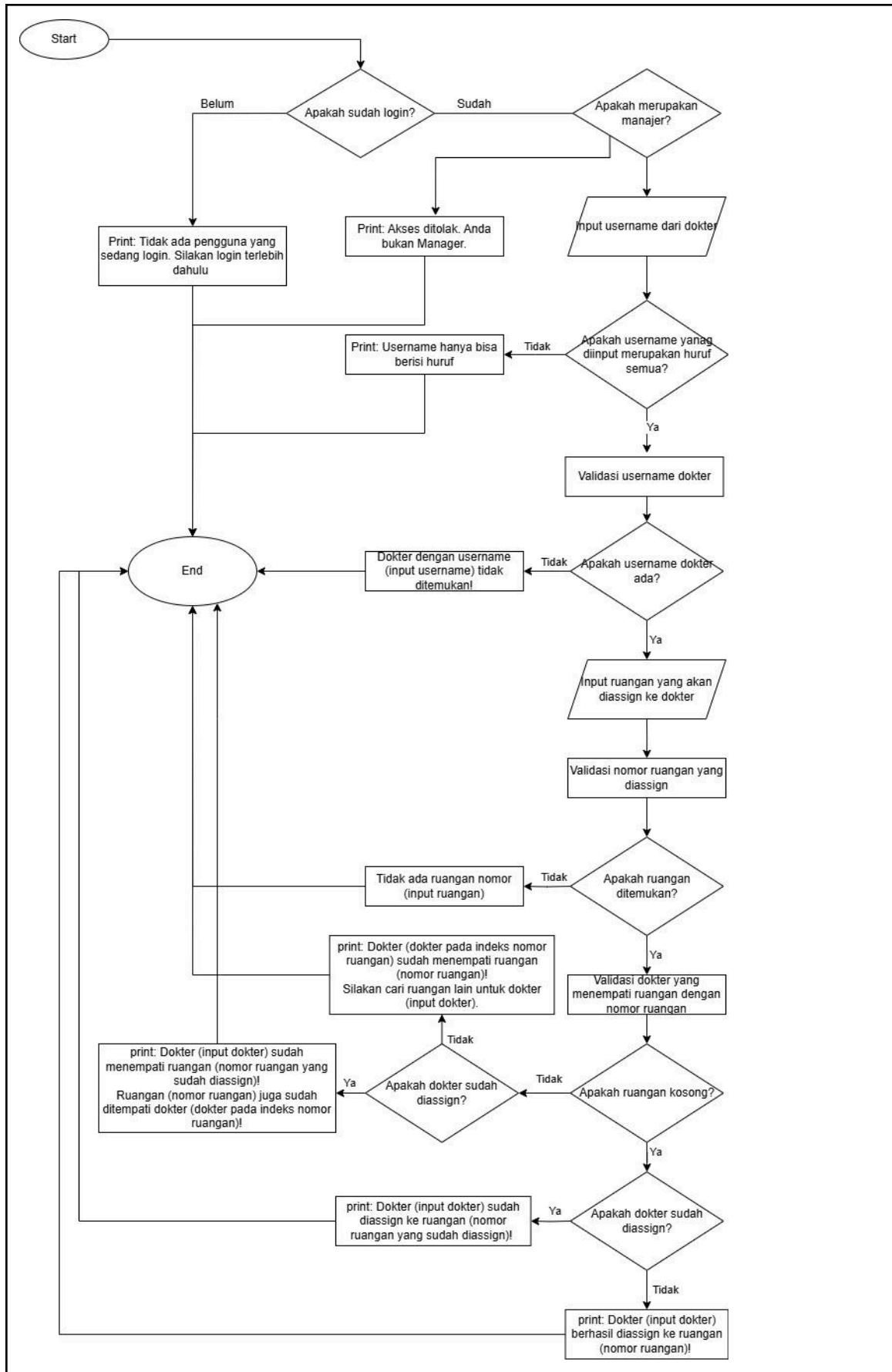
9. F09 - Lihat Antrian

**Gambar 10** Flowchart Lihat Antrian

10. F10 - Tambah Dokter







**Gambar 11** Flowchart Tambah Dokter

11. F11 - Diagnosis

**Gambar 12** Flowchart Diagnosis

12. F12 - Ngobatin

**Gambar 13** Flowchart Ngobatin

13. F13 - Aku boleh pulang ga, dok 🥺?

**Gambar 14** Flowchart Aku boleh pulang ga, dok 🥺?

14. F14 - Daftar Check-Up

**Gambar 15** Flowchart Daftar Check-Up

15. F15 - Antrian Saya!

**Gambar 16** Flowchart Antrian Saya!

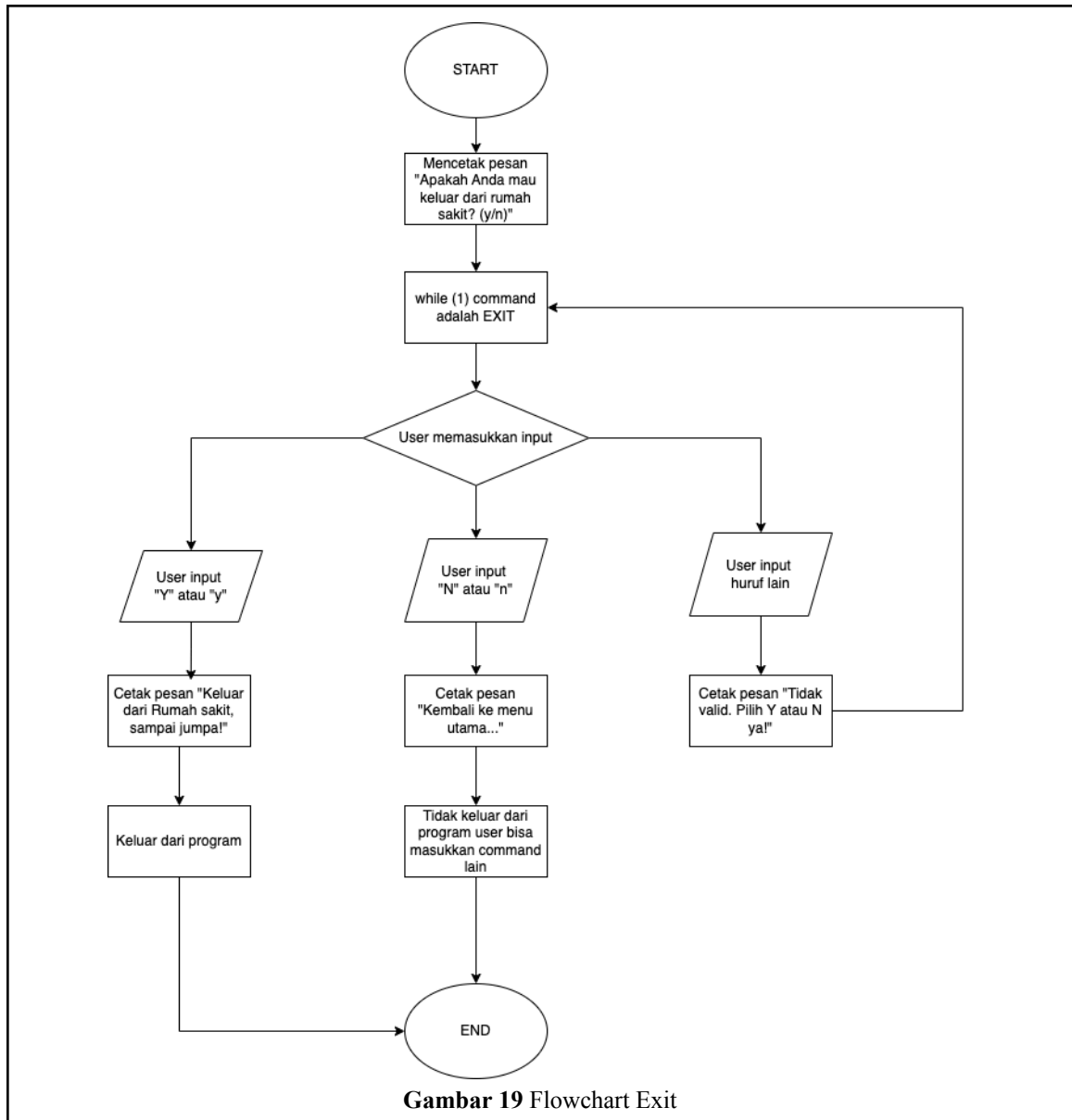
16. F16 - Minum Obat

**Gambar 17** Flowchart Minum Obat

17. F17 - Minum Penawar

**Gambar 18** Flowchart Minum Penawar

18. F18 - Exit



## H. SPESIFIKASI FUNGSIONAL PROGRAM

### 1. F01 - Login

**procedure** login\_system()

{login dengan memasukkan username dan password dan memvalidasinya}

**I.S.** Tidak ada user yang login. User memasukkan username dan password

**F.S.** User berhasil login atau user tidak dapat login

#### KAMUS LOKAL

Login, found\_user, valid\_password : Boolean

```
username, password, role : String  
i : Integer
```

```
procedure login_system ()  
  global login  
  global current_user  
  global username  
  global role  
  global users  
  global user_count  
  if login then  
    output("Login gagal!")  
    output('Anda telah login dengan username', username, 'silahkan  
    lakukan "logout" sebelum melakukan login kembali.')  
  else  
    user <- input("Username : ")  
    password <- input(Password : ")  
  
    user_match <- False  
    password_match <- False  
  
    i <- 0  
    while (i < user_count) do  
      if (users[i].username == username) then  
        user_match <- True  
        if (users[i].password == password) then  
          password_match <- True  
          current_user <- &users[i]  
          break  
      i ← i + 1  
  
    if (user_match) then  
      if (password_match) then  
        login <- True  
        username <- current_user.username  
        role <- current_user.role  
        output("Selamat pagi ", role_to_string(role), " ",  
username, " !")  
        if (role == "pasien") then  
          output("Ada keluhan apa?")  
        else  
          output("")  
      else  
        output("Password salah untuk pengguna yang bernama ",  
username, " !")  
      else  
        output("Tidak ada Manager, Dokter, atau pun Pasien yang  
bernama ", username, " !")
```

## 2. F02 - Register

**procedure** register\_pasien()  
{Melakukan pendaftaran pasien baru ke dalam sistem}  
**I.S.** Program aktif tanpa user yang login; user ingin mendaftar sebagai pasien dengan username dan password.  
**F.S.** Pasien terdaftar jika username valid dan belum dipakai, serta tidak ada user yang login. Jika tidak, registrasi dibatalkan dengan pesan kesalahan.

### KAMUS LOKAL

is\_valid\_input, is\_alpha\_only, is\_unique: Boolean  
username, password : String

```
global current_user
global user_count
global users

if current_user != NULL then
    output("Anda sudah login. Silakan logout terlebih dahulu.")
else
    username <- input("Username:")
    if not is_alpha_string(username) then
        output("Username hanya boleh berisi huruf!")
        return

    if not is_username_unique(username) then
        output("Registrasi gagal! Pasien dengan nama ", username,
sudah terdaftar.")
        return

    password <- input("Password: ")

    if user_count >= MAX_USERS then
        output("Kapasitas user penuh!")
        return

    users[user_count].username <- username
    users[user_count].password <- password
    users[user_count].role <- ROLE_PASIEN
    user_count <- user_count + 1

    output("Pasien ", username, " berhasil ditambahkan!")
```

## 3. F03 - Logout

**procedure** F03()  
 {Prosedur untuk melakukan logout dari sistem dan menghapus informasi user yang sedang login}  
**I.S.** Program dalam keadaan berjalan, dan variabel current\_user menunjuk ke user yang sedang login (jika ada)  
**F.S.** Jika ada user yang login, maka user berhasil logout dan current\_user diset menjadi NULL; jika belum login, sistem menampilkan pesan untuk login terlebih dahulu.

**KAMUS LOKAL**  
 is\_logged\_in : Boolean

**global** current\_user  
**if** current\_user == NULL **then**  
     **output** ("Logout gagal!")  
     **output** ("Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout")  
**else**  
     **output** ("Sampai jumpa ", current\_user.username, "!")  
     current\_user <- Null

#### 4. F04 - Lupa Password

**procedure** lupa\_password()  
 {Prosedur untuk mereset password user yang lupa dengan memverifikasi username dan kode unik}  
**I.S.** Program dalam keadaan berjalan, dan variabel current\_user menunjuk ke user yang sedang login (jika ada)  
**F.S.** Jika username ditemukan dan kode unik sesuai, user diminta memasukkan password baru; jika tidak, sistem menampilkan pesan kesalahan yang sesuai.

**KAMUS LOKAL**  
 username, kode\_unik\_input, kode\_unik\_asli, password\_baru : String  
 i : integer  
 Is\_valid\_input, is\_kode\_sesuai : Boolean

**global** users  
**global** user\_count  
     **output** ("Username: ")  
     **input**(username)  
     user <- NULL  
     for i <- 0 to user\_count - 1 do  
         **if** (users[i].username == username) **then**  
             user <- &users[i]  
             break  
     **if** user == NULL **then**  
         output("Username tidak terdaftar!")

```

        return
    output ("Kode Unik")
    input(kode_unik_input)

    generate_kode_unik(user.username, kode_unik_asli)
    if (kode_unik_input != kode_unik_asli) then
        output("Kode unik salah!")
        return

    output("Halo ", role_to_string(user.role), " ", user.username, ",
    silakan daftarkan ulang password anda!")
    output("Password Baru: ")
    input(password_baru)

    user.password <- password_baru
    output("Password berhasil diubah!")

```

#### 5. F05 - Menu & Help

```

procedure help_system()
{menampilkan daftar perintah yang tersedia sesuai status login dan
peran pengguna}
I.S. Pengguna dapat belum login atau sudah login sebagai Manager,
Dokter, atau Pasien
F.S. Sistem menampilkan perintah yang sesuai dengan status dan peran
pengguna, atau instruksi login/register jika belum login

```

##### KAMUS LOKAL

username : String

```

procedure help_system()
if current_user == NULL then
    output("Kamu belum login sebagai role apapun. Silahkan login
    terlebih dahulu.\n")
    output("LOGIN: Masuk ke dalam akun yang sudah terdaftar")
    output("REGISTER: Membuat akun baru")
else
    output("Halo ", role_to_string(current_user.role), " ",
    current_user.username, ". ")

    if current_user.role == ROLE_DOKTER then
        output("Kamu memanggil command HELP. Kamu pasti sedang
        kebingungan.")
        output("Berikut adalah hal-hal yang dapat kamu lakukan
        sekarang:\n")
        output("LOGOUT: Keluar dari akun yang sedang digunakan")
        output("DIAGNOSIS: Melakukan diagnosis penyakit pasien
        berdasarkan kondisi tubuh pasien")

```

```

    else if current_user.role == ROLE_PASIEN then
        output("Kamu memanggil command HELP. Kamu pasti sedang kebingungan.")
        output("Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n")
        output("LOGOUT: Keluar dari akun yang sedang digunakan")
        output("DAFTAR_CHECKUP: Mendaftarkan diri untuk pemeriksaan dokter")

    else if current_user.role == ROLE_MANAGER then
        output("Kenapa kamu memanggil command HELP? Kan kamu manager, tapi yasudahlah kamu pasti sedang kebingungan.")
        output("Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n")
        output("LOGOUT: Keluar dari akun yang sedang digunakan")
        output("TAMBAH_DOKTER: Mendaftarkan dokter baru ke sistem")
output("\\nFootnote:")
output("Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar")
output("Jangan lupa untuk memasukkan input yang valid")

```

## 6. F06 - Denah Rumah Sakit

- Lihat denah

```

procedure lihat_denah(input: ListRuangan)
{Prosedur untuk menampilkan denah rumah sakit berupa nomor-nomor ruangan}
I.S. Data ruangan sudah terisi dengan jumlah ruangan dan nomor masing-masing.
F.S. Menampilkan daftar nomor ruangan dalam format denah horizontal.

```

### KAMUS LOKAL

```

i : integer
global ruangan : ListRuangan

```

```

procedure lihat_denah(input: ListRuangan)

output newline
output("+---+---+---+---+---+---+---+---+")
for i traversal [0... ruangan.jumlah - 1]
    output("| ", ruangan.ruang[i].nomor, " ", end="")
output("|")
output("+---+---+---+---+---+---+---+---+")

```



- Lihat Ruangan

<p><b><u>procedure</u></b> lihat_ruangan (input: ListRuangan, input: integer)</p> <p>{Prosedur untuk menampilkan detail sebuah ruangan rumah sakit berdasarkan nomor ruangan}</p> <p>I.S. Pengguna sudah login dengan role Manager, Dokter, atau Pasien.</p> <p>F.S. Jika nomor ruangan valid, detail ruangan ditampilkan; jika tidak, tampilkan pesan kesalahan.</p>
<p><b>KAMUS LOKAL</b></p> <p>r : pointer to Ruangan</p> <p>i : integer</p> <p>global ruangan : ListRuangan</p> <p>global current_user : pointer to User</p>
<p><b><u>procedure</u></b> lihat_ruangan (input: ListRuangan, input: integer)</p> <p><u>If</u> (current_user.role != ROLE_MANAGER and current_user.role != ROLE_DOKTER and current_user.role != ROLE_PASIEN) <u>then</u></p> <p>    <u>output</u>("Role tidak ditemukan. Akses ditolak.")</p> <p>    -&gt;</p> <p><u>If</u> (num &lt;= 0 or num &gt; ruangan.jumlah) <u>then</u></p> <p>    <u>output</u>("Nomor ruangan tidak valid.")</p> <p>    -&gt;</p> <p>r &lt;- &amp;ruangan.ruang[num - 1]</p> <p><u>output</u>("--- Detail Ruangan ", r.nomor, " ---")</p> <p><u>output</u>("Kapasitas : ", r.kapasitas)</p> <p><u>output</u>("Dokter : ", r.dokter)</p> <p><u>output</u>("Pasien di dalam ruangan:")</p> <p><u>if</u> (r.pasien.Count == 0) <u>then</u></p> <p>    <u>output</u>("Tidak ada pasien di dalam ruangan saat ini.")</p> <p><u>else then</u></p> <p>    i <u>traversal</u> [0... r.pasien.Count - 1]</p> <p>        <u>output</u>(" ", i + 1, ". ", r.pasien.Elements[i])</p> <p><u>output</u>("-----")</p>

7. F07 - Lihat User
8. F08 - Cari User
9. F09 - Lihat Antrian
10. F10 - Tambah Dokter

- Tambah dokter

**PROCEDURE** tambah\_dokter (input/output: ListUser, input/output: Set)  
{Prosedur untuk menambahkan user baru dengan role dokter, hanya bisa dilakukan oleh Manajer}

I.S. Sistem dalam keadaan berjalan dan pengguna sudah login.

F.S. Jika valid, dokter baru ditambahkan ke list pengguna dan username-nya disimpan di set; jika tidak, tampilkan pesan kesalahan.

#### **KAMUS LOKAL**

username, password : string  
new\_user : User  
global current\_user : pointer to User  
global users : ListUser  
global usernames : Set

**PROCEDURE** tambah\_dokter (input/output: ListUser, input/output: Set)

If (current\_user == NULL) then  
    output("Tidak ada pengguna yang sedang login. Silakan login terlebih dahulu.")  
    ->

if (current\_user.role != ROLE\_MANAGER) then  
    output("Akses ditolak. Anda bukan Manager.")  
    ->

output("Username: ")  
input(username)

if (!is\_alpha\_string(username)) then  
    output("Username hanya boleh berisi huruf!")  
    ->

if (!is\_username\_unique(usernames, username)) then  
    output("Sudah ada dokter bernama ", username, "!")  
    ->

output("Password: ")  
input(password)

if (IsFull(users)) then  
    output("Kapasitas user penuh!")  
    ->

new\_user.username <- username  
new\_user.password <- password  
new\_user.role <- ROLE\_DOKTER

SetEl(users, NbElmt(users), new\_user)

```
SetLength(users, NbElmt(users) + 1)
Insert(usernames, username)

output("Dokter ", username, " berhasil ditambahkan!")
```

- Assign dokter

**PROCEDURE** assign\_dokter(input/output: ListUser, input/output: ListRuangan)  
{Prosedur untuk menetapkan seorang dokter ke ruangan tertentu, dilakukan oleh Manager}  
I.S. Sistem dalam keadaan berjalan dan pengguna sudah login.  
F.S. Jika valid, dokter ditugaskan ke ruangan; jika tidak, tampilkan pesan kesalahan.

**KAMUS LOKAL**

username : string  
nomor\_ruangan, ruangan\_idx : integer  
dokter\_baru : User  
dokter\_found, ruangan\_found : boolean  
global current\_user : pointer to User  
global users : ListUser  
global ruangan : ListRuangan

**PROCEDURE** assign\_dokter(input/output: ListUser, input/output: ListRuangan)

if (current\_user == NULL) then  
    output("Tidak ada pengguna yang sedang login. Silakan login terlebih dahulu.")  
    ->

if (current\_user.role != ROLE\_MANAGER) then  
    output("Akses ditolak. Anda bukan Manager.")  
    ->

output("Username: ")  
input(username)

if (!is\_alpha\_string(username)) then  
    output("Username hanya boleh berisi huruf!")  
    ->

dokter\_found <- false  
i traversal [GetFirstIdx(users)... GetLastIdx(users)]  
    if (users[i].username == username and users[i].role == ROLE\_DOKTER) then  
        dokter\_baru <- users[i]

```

        dokter_found <- true
        break

    if (!dokter_found) then
        output("Dokter dengan username ", username, " tidak ditemukan!")
        ->

    output("Ruangan: ")
    input(nomor_ruangan)

    ruangan_found <- false
    ruangan_idx <- -1
    i traversal [0... ruangan.jumlah - 1]
        if (ruangan.ruang[i].nomor == nomor_ruangan) then
            ruangan_found <- true
            ruangan_idx <- i
            break

    if (!ruangan_found) then
        output("Tidak ada ruangan nomor ", nomor_ruangan, "!")
        return

    i traversal [0... ruangan.jumlah - 1]
        if (ruangan.ruang[i].dokter == username) then
            output("Dokter ", username, " sudah diassign ke ruangan ",
            ruangan.ruang[i].nomor, "!")
            ->

    if (ruangan.ruang[ruangan_idx].dokter == "") then
        ruangan.ruang[ruangan_idx].dokter <- username
        output("Dokter ", username, " berhasil diassign ke ruangan ",
        nomor_ruangan, "!")
        ->

    else if (ruangan.ruang[ruangan_idx].dokter != "-" and ruangan_idx !=
    -1) then
        output("Dokter ", ruangan.ruang[ruangan_idx].dokter, " sudah
        menempati ruangan ", nomor_ruangan, "!")
        output("Silakan cari ruangan lain untuk dokter ", username, ".")
        ->

```

11. F11 - Diagnosis
12. F12 - Ngobatin
13. F13 - Aku boleh pulang ga, dok?
14. F14 - Daftar Check Up
15. F15 - Antrian Saya
16. F16 - Minum Obat

17. F17 - Minum Penawar

18. F18 - Exit

```
procedure exit_system()  
{Prosedur untuk keluar dari sistem rumah sakit}  
I.S. Program dalam keadaan berjalan, current_user mungkin sedang login atau tidak  
F.S. Jika user sedang login, sistem meminta konfirmasi keluar dan menutup program jika dikonfirmasi; jika tidak login, sistem meminta login terlebih dahulu.
```

KAMUS LOKAL  
input : string[10]

```
ALGORITMA  
if current_user == NULL then  
    output ("Tidak ada pengguna yang sedang login. Silakan login terlebih dahulu untuk keluar.")  
else  
    output ("Apakah Anda mau keluar dari rumah sakit? (y/n)")  
    repeat  
        output ("Pilihan Anda: ")  
        input (input)  
  
        if (input == "Y" or input == "y") then  
            output (current_user.username, " keluar dari Rumah Sakit,")  
            output ("Sampai jumpa!")  
            exit(0)  
        else if (input == "N" or input == "n") then  
            output ("Kembali ke menu utama...")  
            return  
        else  
            output ("Tidak valid. Pilih Y atau N ya!")  
    forever  
endif
```

## I. LAMPIRAN SOURCE CODE

**Gambar 20** Source code struktur data user.h

```
#ifndef USER_H
#define USER_H

#include "boolean.h"

#define MAX_USERS 100
#define USERNAME_LEN 50
#define PASSWORD_LEN 50

// Enum untuk role pengguna
typedef enum {
    ROLE_MANAGER,
    ROLE_DOKTER,
    ROLE_PASIEN,
    ROLE_NONE
} UserRole;

// Struktur data user
typedef struct {
    char username[USERNAME_LEN];
    char password[PASSWORD_LEN];
    UserRole role;
} User;

// Variabel global yang digunakan di seluruh program
extern int user_count;           // Jumlah user yang terdaftar
extern User* current_user;      // Pointer ke user yang sedang login

// Deklarasi fungsi utility
// Di user.h
// Deklarasi saja (tanpa implementasi)
const char* role_to_string(UserRole role);

boolean is_alpha_string(const char* str); // Deklarasi saja!
#endif
//user.h
```

Gambar 21 Source code header ADT list\_user.h

```
#ifndef List_User_H
#define List_User_H

#include "Boolean.h"
#include "../..//header/user.h"

#define MAX_CAPACITY 100
#define IDX_UNDEF -999 /* Tidak terdefinisi */

typedef int IdxType;

typedef struct {
    User data[MAX_CAPACITY];
    IdxType length;
} ListUser;

/* Indeks yang digunakan [0..MAX_CAPACITY-1] */

/* ***** KONSTRUKTOR ARRAY ***** */
/* Konstruktor : create tabel kosong */
void MakeEmpty(ListUser *L);
/* I.S. sembarang */
/* F.S. Terbentuk list L kosong dengan kapasitas MAX_CAPACITY */

/* ***** SELEKTOR ***** */
/* *** Banyaknya elemen *** */
int NbElmt(ListUser L);
/* Mengirimkan banyaknya elemen efektif list */
/* Mengirimkan nol jika list kosong */

/* *** Daya tampung container *** */
int MaxNbEl(ListUser L);
/* Mengirimkan maksimum elemen yang dapat ditampung oleh list */

/* *** Selektor INDEKS *** */
IdxType GetFirstIdx(ListUser L);
/* Prekondisi : List L tidak kosong */
/* Mengirimkan indeks elemen pertama */

IdxType GetLastIdx(ListUser L);
/* Prekondisi : List L tidak kosong */
/* Mengirimkan indeks elemen terakhir */

/* *** Menghasilkan sebuah elemen *** */
User GetElmt(ListUser L, IdxType i);
/* Prekondisi : List tidak kosong, i antara FirstIdx(L)..LastIdx(L) */
/* Mengirimkan elemen list yang ke-i */
```

```

/* *** Selektor SET : Mengubah nilai list dan elemen list *** */
void SetTab(ListUser Lin, ListUser *Lout);
/* I.S. Lin terdefinisi, sembarang */
/* F.S. Lout berisi salinan Lin */
/* Assignment Lout = Lin */

void SetEl(ListUser *L, IdxType i, User v);
/* I.S. L terdefinisi, sembarang */
/* F.S. Elemen L yang ke-i bernilai v */
/* Mengeset nilai elemen list yang ke-i sehingga bernilai v */

void SetLength(ListUser *L, IdxType N);
/* I.S. L terdefinisi, sembarang */
/* F.S. Nilai indeks efektif L bernilai N */
/* Mengeset nilai i
typedef struct <unnamed> ListUser
❖ Generate Copilot summary
lai N */

/* ***** Test
boolean IsIdxValid(ListUser L, IdxType i);
/* Prekondisi : i sembarang */
/* Mengirimkan true jika i adalah indeks yang valid utk ukuran list */
/* yaitu antara indeks yang terdefinisi utk container */

boolean IsIdxEff(ListUser L, IdxType i);
/* Prekondisi : i sembarang */
/* Mengirimkan true jika i adalah indeks yang terdefinisi utk list */
/* yaitu antara GetFirstIdx(L)..GetLastIdx(L) */

/* ***** TEST KOSONG/PENUH ***** */
/* *** Test tabel kosong *** */
boolean IsEmpty(ListUser L);
/* Mengirimkan true jika list L kosong, mengirimkan false jika tidak */

/* *** Test tabel penuh *** */
boolean IsFull(ListUser L);
/* Mengirimkan true jika tabel T penuh, mengirimkan false jika tidak */

#endif

```



**Gambar 22** Source code implementasi ADT list\_user.h

```
#include "../header/List_User.h"

void MakeEmpty(ListUser *L) {
    L->length = 0;
}

int NbElmt(ListUser L) {
    return L.length;
}

int MaxNbEl(ListUser L) {
    return MAX_CAPACITY;
}

IdxType GetFirstIdx(ListUser L) {
    return 0;
}

IdxType GetLastIdx(ListUser L) {
    return L.length - 1;
}

User GetElmt(ListUser L, IdxType i) {
    return L.data[i];
}

void SetTab(ListUser Lin, ListUser *Lout) {
    Lout->length = Lin.length;
    for (int i = 0; i < Lin.length; i++) {
        Lout->data[i] = Lin.data[i];
    }
}

void SetEl(ListUser *L, IdxType i, User v) {
    L->data[i] = v;
    if (L->length < i + 1) {
        L->length = i + 1;
    }
}

void SetLength(ListUser *L, IdxType N) {
    L->length = N;
}

boolean IsIdxValid(ListUser L, IdxType i) {
    return i >= GetFirstIdx(L) && i < MAX_CAPACITY;
}
```

```

boolean IsIdxEff(ListUser L, IdxType i) {
    return i >= GetFirstIdx(L) && i <= GetLastIdx(L);
}

boolean IsEmpty(ListUser L) {
    return L.length == 0;
}

boolean IsFull(ListUser L) {
    return L.length == MAX_CAPACITY;
}

```

Gambar 23 Source code header ADT set.h

```

#ifndef SET_H
#define SET_H

#include "boolean.h"

#define Nil 0
#define MaxEl 100
#define USERNAME_LEN 50 // Panjang maksimum username

typedef struct {
    char Elements[MaxEl][USERNAME_LEN]; // Array untuk menyimpan username
    int Count; // Jumlah elemen dalam Set
} Set;

extern Set usernames; // Deklarasi variabel global untuk menyimpan daftar username

/* Definisi Set S kosong : S.Count = Nil */

/* *** Konstruktor/Kreator *** */
void CreateEmpty(Set *S);
/* I.S. Sembarang */
/* F.S. Membuat sebuah Set S kosong berkapasitas MaxEl */
/* Ciri Set kosong : count bernilai Nil */

/* ***** Predikat Untuk test keadaan KOLEKSI ***** */
boolean Set_IsEmpty(Set S);
/* Mengirim true jika Set S kosong */
/* Ciri Set kosong : count bernilai Nil */

boolean Set_IsFull(Set S);
/* Mengirim true jika Set S penuh */
/* Ciri Set penuh : count bernilai MaxEl */

/* ***** Operator Dasar Set ***** */
void Insert(Set *S, const char *Elmt);
/* Menambahkan Elmt sebagai elemen Set S. */
/* I.S. S mungkin kosong, S tidak penuh
   S mungkin sudah beranggotakan Elmt */
/* F.S. Elmt menjadi anggota dari S. Jika Elmt sudah merupakan anggota, operasi tidak dilakukan */

void Delete(Set *S, const char *Elmt);
/* Menghapus Elmt dari Set S. */
/* I.S. S tidak kosong
   Elmt mungkin anggota / bukan anggota dari S */
/* F.S. Elmt bukan anggota dari S */

boolean IsMember(Set S, const char *Elmt);
/* Mengembalikan true jika Elmt adalah anggota dari S */

boolean is_username_unique(const Set *usernames, const char *username);

#endif

```

**Gambar 24** Source code implementasi ADT set.h

```
#include "../header/set.h"
#include <stdio.h>
#include <string.h> // Untuk operasi string

void CreateEmpty(Set *S) {
    S->Count = Nil;
}

boolean Set_IsEmpty(Set S) {
    return S.Count == Nil;
}

boolean Set_IsFull(Set S) {
    return S.Count == MaxEl;
}

void Insert(Set *S, const char *Elmt) {
    if (Set_IsFull(*S)) return; // Tidak bisa menambahkan jika penuh
    if (IsMember(*S, Elmt)) return; // Tidak menambahkan elemen duplikat

    strcpy(S->Elements[S->Count], Elmt); // Salin elemen ke dalam Set
    S->Count++;
}

void Delete(Set *S, const char *Elmt) {
    if (Set_IsEmpty(*S)) return; // Tidak bisa menghapus jika kosong

    int idx = -1;
    for (int i = 0; i < S->Count; i++) {
        if (strcmp(S->Elements[i], Elmt) == 0) { // Bandingkan string
            idx = i;
            break;
        }
    }

    if (idx == -1) return; // Elemen tidak ditemukan

    for (int i = idx; i < S->Count - 1; i++) {
        strcpy(S->Elements[i], S->Elements[i + 1]); // Geser elemen
    }
    S->Count--;
}

boolean IsMember(Set S, const char *Elmt) {
    for (int i = 0; i < S.Count; i++) {
        if (strcmp(S.Elements[i], Elmt) == 0) { // Bandingkan string
            return true;
        }
    }
    return false;
}
```

```

boolean is_username_unique(const Set *usernames, const char *username) {
    for (int i = 0; i < usernames->Count; i++) {
        const char *a = usernames->Elements[i];
        const char *b = username;

        while (*a && *b) {
            char a_char = *a;
            char b_char = *b;

            if (a_char >= 'A' && a_char <= 'Z') a_char += 32;
            if (b_char >= 'A' && b_char <= 'Z') b_char += 32;

            if (a_char != b_char) break;

            a++;
            b++;
        }

        if (*a == '\0' && *b == '\0') {
            return false;
        }
    }
    return true;
}

```

Gambar 25 Source code F01: login.h

```

1  #ifndef LOGIN_H
2  #define LOGIN_H
3
4  #include "user.h"
5  #include "../ADT/header/ListofUser.h"
6  void login_system(ListUser *users);
7
8  #endif
9  // login.h

```

Gambar 26 Source code F01: login.c

```
#include "../header/login.h"
#include <stdio.h>
#include <string.h>

void login_system(ListUser *users) {
    // KAMUS LOKAL
    boolean found_user = false, valid_password = false; // Apakah username ditemukan, Apakah password valid
    char username[USERNAME_LEN]; // Input username
    char password[PASSWORD_LEN]; // Input password
    char role[20]; // Role pengguna
    int i; // Indeks iterasi

    // Cek apakah sudah ada user yang login
    if (current_user != NULL) {
        printf("Anda sudah login sebagai %s %s. Silakan logout terlebih dahulu untuk login dengan akun lain.\n",
            role_to_string(current_user->role), current_user->username);
        return;
    }

    // Input username
    printf("Masukan username: ");
    if (scanf("%49s", username) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    // Input password
    printf("Masukan password: ");
    if (scanf("%49s", password) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    // Iterasi untuk mencari username
    for (i = GetFirstIdx(*users); i <= GetLastIdx(*users); i++) {
        User user = GetElmt(*users, i);

        if (strcmp(user.username, username) == 0) {
            found_user = true; // Username ditemukan

            if (strcmp(user.password, password) == 0) {
                valid_password = true; // Password cocok
                current_user = &users->data[i];

                // Tampilkan pesan selamat datang
                printf("Selamat pagi %s %s!",
                    role_to_string(current_user->role),
                    current_user->username);

                if (current_user->role == ROLE_PASIEN) {
                    printf("Ada keluhan apa ?\n");
                } else {
                    printf("\n");
                }
                return;
            }
            if (!valid_password) { // Password tidak cocok
                printf("Password salah untuk pengguna yang bernama %s!\n", username);
                return;
            }
        }
    }

    if (!found_user) {
        printf("Tidak ada Manager, Dokter, atau pun Pasien yang bernama %s!\n", username);
    }
}
```

**Gambar 27** Source code F02: register.h

```
1  ✓ #ifndef REGISTER_H
2    #define REGISTER_H
3
4  ✓ #include "../ADT/header/list_user.h"
5    #include "../ADT/header/set.h"
6
7    void register_pasien(ListUser *users, Set *usernames);
8
9    #endif
```

Gambar 28 Source code F02: register.c

```
#include "../header/register.h"
#include "../header/user.h"
#include <stdio.h>
#include <string.h>

void register_pasien(ListUser *users, Set *usernames) {
    if (current_user != NULL) {
        printf("Anda sudah login. Silakan logout terlebih dahulu.\n");
        return;
    }

    char username[USERNAME_LEN];
    char password[PASSWORD_LEN];

    printf("Username: ");
    if (scanf("%49s", username) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    if (!is_alpha_string(username)) {
        printf("Username hanya boleh berisi huruf!\n");
        return;
    }

    if (!is_username_unique(usernames, username)) {
        printf("Registrasi gagal! Username %s sudah digunakan!\n", username);
        return;
    }

    printf("Password: ");
    if (scanf("%49s", password) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    if (IsFull(*users)) {
        printf("Kapasitas user penuh!\n");
        return;
    }

    User new_user;
    strcpy(new_user.username, username);
    strcpy(new_user.password, password);
    new_user.role = ROLE_PASIEN;

    SetEl(users, NbElmt(*users), new_user);
    SetLength(users, NbElmt(*users) + 1);
    Insert(usernames, username);

    printf("Pasien %s berhasil ditambahkan!\n", username);
}
```

**Gambar 29** Source code F03: logout.h

```
#ifndef LOGOUT_H
#define LOGOUT_H

#include "user.h"

void logout_system();

#endif
```

**Gambar 30** Source code F03: logout.c

```
#include "../header/logout.h"
#include <stdio.h>

void logout_system() {
    boolean is_logged_in = (current_user != NULL);
    if (!is_logged_in) {
        printf("Logout gagal!\n");
        printf("Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout\n");
        return;
    }
    printf("Sampai jumpa %s!\n", current_user->username);
    current_user = NULL;
}
```

**Gambar 31** Source code F04: lupa\_password.h

```
#ifndef LUPA_PASSWORD_H
#define LUPA_PASSWORD_H

#include "../ADT/header/ListofUser.h"

void lupa_password_system(ListUser *users);
void generate_kode_unik(const char* username, char* kode_unik);

#endif
```



Gambar 32 Source code F04: lupa\_password.c

```
#include "../header/lupa_password.h"
#include <stdio.h>
#include <string.h>

void generate_kode_unik(const char* username, char* kode_unik) {
    int len = strlen(username);
    int count = 1;
    int pos = 0;

    for (int i = 1; i <= len; i++) {
        if (i < len && username[i] == username[i-1]) {
            count++;
        } else {
            if (count > 1) {
                pos += sprintf(kode_unik + pos, "%d", count);
            }
            kode_unik[pos++] = username[i-1];
            count = 1;
        }
    }
    kode_unik[pos] = '\0';
}
```

```

void lupa_password_system(ListUser *users) {
    char username[USERNAME_LEN];
    char kode_unik_input[100];
    char kode_unik_asli[100];
    char password_baru[PASSWORD_LEN];

    printf("Username: ");
    if (scanf("%49s", username) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    User* user = NULL;
    for (int i = GetFirstIdx(*users); i <= GetLastIdx(*users); i++) {
        if (strcmp(GetElmt(*users, i).username, username) == 0) {
            user = &users->data[i];
            break;
        }
    }

    if (user == NULL) {
        printf("Username tidak terdaftar!\n");
        return;
    }

    printf("Kode Unik: ");
    if (scanf("%99s", kode_unik_input) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    generate_kode_unik(user->username, kode_unik_asli);

    if (strcmp(kode_unik_input, kode_unik_asli) != 0) {
        printf("Kode unik salah!\n");
        return;
    }

    printf("Halo %s %s, silakan daftarkan ulang password anda!\n",
           role_to_string(user->role), user->username);
    printf("Password Baru: ");
    if (scanf("%49s", password_baru) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }
    if (strcmp(password_baru, user->password) == 0) {
        printf("Password baru tidak boleh sama dengan password lama!\n");
        return;
    }

    strcpy(user->password, password_baru);
    printf("Password berhasil diubah!\n");
}

```

**Gambar 33** Source code F05: help.h

```
#ifndef HELP_H
#define HELP_H

#include "user.h"

void help_system();

#endif
// help.h
```

Gambar 34 Source code F05: help.c

```
#include "../header/help.h"
#include <stdio.h>

void help_system() {
    printf("\n===== HELP =====\n\n");

    if (current_user == NULL) {
        printf("Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.\n\n");
        printf("LOGIN: Masuk ke dalam akun yang sudah terdaftar\n");
        printf("REGISTER: Membuat akun baru\n");
    } else {
        printf("Halo %s %s. ", role_to_string(current_user->role), current_user->username);

        if (current_user->role == ROLE_DOKTER) {
            printf("Kamu memanggil command HELP. Kamu pasti sedang kebingungan.\n");
            printf("Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n\n");
            printf("LOGOUT: Keluar dari akun yang sedang digunakan\n");
            printf("DIAGNOSIS: Melakukan diagnosis penyakit pasien berdasarkan kondisi tubuh pasien\n");
        } else if (current_user->role == ROLE_PASIEN) {
            printf("Kamu memanggil command HELP. Kamu pasti sedang kebingungan.\n");
            printf("Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n\n");
            printf("LOGOUT: Keluar dari akun yang sedang digunakan\n");
            printf("DAFTAR_CHECKUP: Mendaftarkan diri untuk pemeriksaan dokter\n");
        } else if (current_user->role == ROLE_MANAGER) {
            printf("Kenapa kamu memanggil command HELP? Kan kamu manager, ");
            printf("tapi yasudahlah kamu pasti sedang kebingungan.\n");
            printf("Berikut adalah hal-hal yang dapat kamu lakukan sekarang:\n\n");
            printf("LOGOUT: Keluar dari akun yang sedang digunakan\n");
            printf("TAMBAH_DOKTER: Mendaftarkan dokter baru ke sistem\n");
        }
    }

    printf("\nFootnote: \n");
    printf("Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar\n");
    printf("Jangan lupa untuk memasukkan input yang valid\n");
}
```

**Gambar 27** Source code F06: lihat\_denah.h

```
#ifndef LIHAT_DENAH_H
#define LIHAT_DENAH_H

#include "../ADT/header/list_user.h"
#include "../ADT/header/set.h"
#include "../header/register.h"
#include "../header/ruangan.h"

void lihat_denah (ListRuangan ruangan);

void lihat_ruangan (ListRuangan ruangan, int num);

#endif
```

```
#include <stdio.h>
#include "../header/lihat_denah.h"

void lihat_denah (ListRuangan ruangan) {
    printf("\n");
    printf("+-----+-----+-----+-----+-----+-----+\n");
    for (int i = 0; i < ruangan.jumlah; i++) {
        printf("|   %d   ", ruangan.ruang[i].nomor);
    }
    printf("\n");
    printf("+-----+-----+-----+-----+-----+-----+\n");
}

void lihat_ruangan (ListRuangan ruangan, int num){
    while (current_user->role != ROLE_MANAGER && current_user->role != ROLE_DOKTER
           && current_user->role != ROLE_PASIEN) {
        printf ("Role tidak ditemukan. Akses ditolak.\n");
        return;
    }

    if (num <= 0 || num > ruangan.jumlah) {
        printf("Nomor ruangan tidak valid.\n");
        return;
    }

    Ruangan *r = &ruangan.ruang[num-1];

    printf("\n--- Detail Ruangan %d ---\n", r->nomor);
    printf("Kapasitas : %d\n", r->kapasitas);
    printf("Dokter      : %s\n", r->dokter);
    printf("Pasien di dalam ruangan:\n");

    if (r->pasien.Count == 0) {
        printf("Tidak ada pasien di dalam ruangan saat ini.\n");
    } else {
        for (int i = 0; i < r->pasien.Count; i++) {
            printf("  %d. %s\n", i + 1, r->pasien.Elements[i]);
        }
    }
    printf("-----\n\n");
}
```

**Gambar 28** Source code F10: f10.h

```
header > C f10.h
1  #ifndef F10_H
2  #define F10_H
3
4  #include "../ADT/header/list_user.h"
5  #include "../ADT/header/set.h"
6  #include "../header/register.h"
7  #include "../header/init_data.h"
8  #include "../header/ruangan.h"
9
10 void tambah_dokter (ListUser *users, Set *usernames);
11 void assign_dokter (ListUser *users, ListRuangan *ruangan);
12
13 #endif
```

Gambar 29 Source code F10: f10.c

```
#include <stdio.h>
#include "../header/f10.h"

void tambah_dokter (ListUser *users, Set *usernames){
    if (current_user == NULL) {
        printf("\nTidak ada pengguna yang sedang login. Silakan login terlebih dahulu.\n");
        return;
    }

    if (current_user->role != ROLE_MANAGER){
        printf ("Akses ditolak. Anda bukan Manager.\n");
        return;
    }

    char username[USERNAME_LEN];
    char password[PASSWORD_LEN];

    printf("Username: ");
    if (scanf("%49s", username) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    if (!is_alpha_string(username)) {
        printf("Username hanya boleh berisi huruf!\n");
        return;
    }

    if (!is_username_unique(usernames, username)) {
        printf("Sudah ada dokter bernama %s!\n", username);
        return;
    }

    printf("Password: ");
    if (scanf("%49s", password) != 1) {
        printf("Input tidak valid!\n");
        while (getchar() != '\n');
        return;
    }

    if (IsFull(*users)) {
        printf("Kapasitas user penuh!\n");
        return;
    }

    User new_user;
    strcpy(new_user.username, username);
    strcpy(new_user.password, password);
    new_user.role = ROLE_DOKTER;

    SetEl(users, NbElmt(*users), new_user);
    SetLength(users, NbElmt(*users) + 1);
    Insert(usernames, username); // Tambahkan username ke dalam Set

    printf("Dokter %s berhasil ditambahkan!\n", username);
}
```



**Gambar 30** Source code F18: exit.h

```
#ifndef __EXIT_H__
#define __EXIT_H__

#include <stdio.h>
#include <stdlib.h>
#include "boolean.h"
#include "../ADT/header/List_user.h"

void exit_system(ListUser *users);

#endif
```

**Gambar 31** Source code F18: exit.c

```
#include "../header/exit.h" // Untuk akses exit_system
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void exit_system(ListUser *users) {
    // Cek apakah ada pengguna yang sedang login
    if (current_user == NULL) {
        printf("\nTidak ada pengguna yang sedang login. Silakan login terlebih dahulu untuk keluar.\n");
        return;
    }

    char input[10];

    printf("\nApakah Anda mau keluar dari rumah sakit? (y/n)\n");

    while (1) {
        printf("Pilihan Anda: ");
        scanf("%s", input);

        if (strcmp(input, "Y") == 0 || strcmp(input, "y") == 0) {
            printf("\n%s keluar dari Rumah Sakit,\nSampai jumpa!\n", current_user->username);
            exit(0);
        } else if (strcmp(input, "N") == 0 || strcmp(input, "n") == 0) {
            printf("\nKembali ke menu utama...\n");
            return;
        } else {
            printf("Tidak valid. Pilih Y atau N ya!\n");
        }
    }
}
```

**Gambar 32** Source code F15

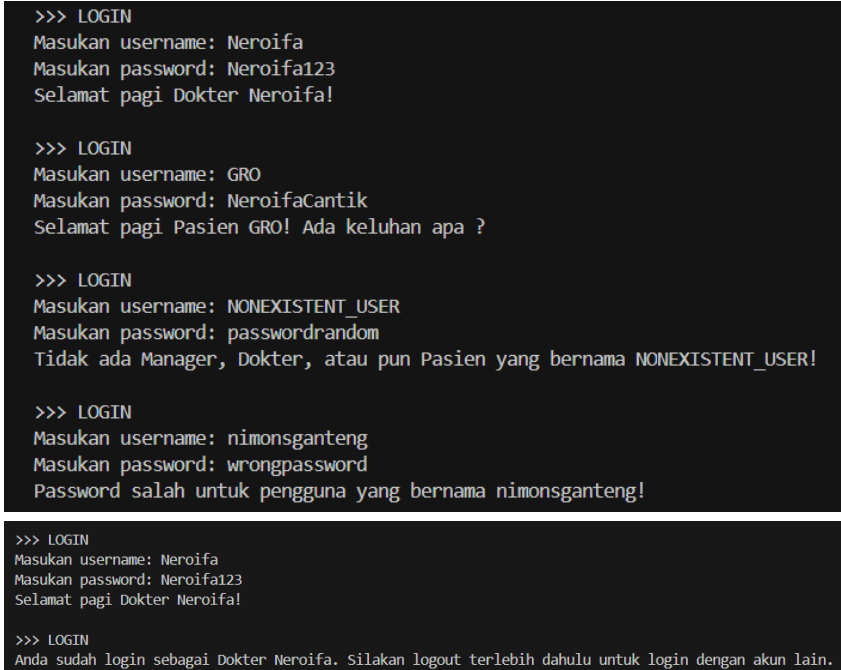
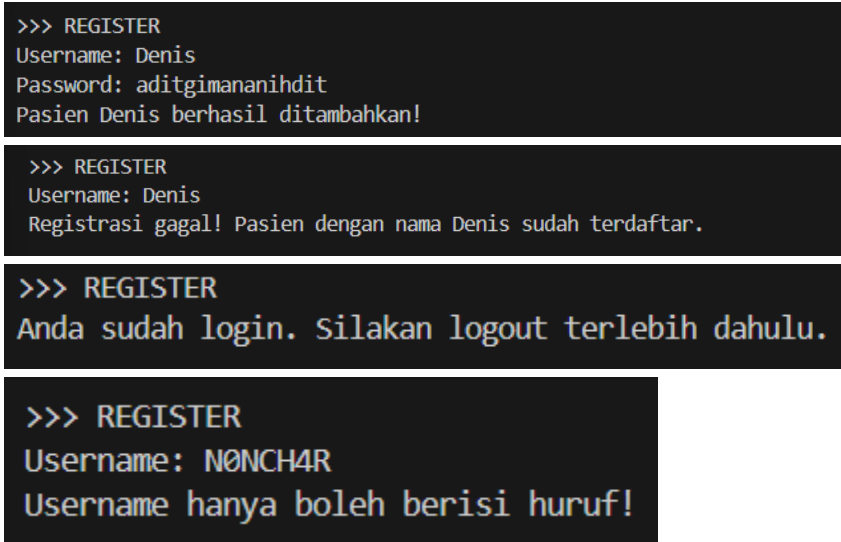
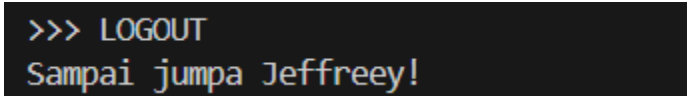
**Gambar 33** Source code F15

**Gambar 34** Source code F15

**Gambar 35** Source code F15

## II. LAMPIRAN HASIL PENGUJIAN PROGRAM

**Tabel 4** Hasil Pengujian Program

Fitur	Hasil Pengujian
F01 - Login	 <p><b>Gambar 36</b> Hasil pengujian F01-Login</p>
F02 - Register	 <p><b>Gambar 37</b> Hasil pengujian F02-Register</p>
F03 - Logout	

	<pre> &gt;&gt;&gt; LOGOUT Logout gagal! Anda belum login, silahkan login terlebih dahulu sebelum melakukan logout </pre> <p style="text-align: center;"><b>Gambar 38</b> Hasil pengujian F03-Logout</p>
F04 - Lupa Password	<pre> &gt;&gt;&gt; LUPA_PASSWORD Username: NONEXISTENT_USER Username tidak terdaftar!  &gt;&gt;&gt; REGISTER Anda sudah login. Silakan logout terlebih dahulu.  &gt;&gt;&gt; LOGOUT Sampai jumpa Jeffreey!  &gt;&gt;&gt; LUPA_PASSWORD Username: NONEXISTENT_USER Username tidak terdaftar!  &gt;&gt;&gt; LUPA_PASSWORD Username: nimonsslatte Kode Unik: nimonsslatte Kode unik salah!  &gt;&gt;&gt; LUPA_PASSWORD Username: Neroifa Kode Unik: Neroifa Halo Dokter Neroifa, silakan daftarkan ulang password anda! Password Baru: Neroifa321 Password berhasil diubah!  &gt;&gt;&gt; LUPA_PASSWORD Username: Neroifa Kode Unik: Neroifa Halo Dokter Neroifa, silakan daftarkan ulang password anda! Password Baru: Neroifa123 Password baru tidak boleh sama dengan password lama! </pre> <p style="text-align: center;"><b>Gambar 39</b> Hasil pengujian F04-Lupa Password</p>
F05 - Menu & Help	<pre> &gt;&gt;&gt; HELP  ===== HELP =====  Kamu belum login sebagai role apapun. Silahkan login terlebih dahulu.  LOGIN: Masuk ke dalam akun yang sudah terdaftar REGISTER: Membuat akun baru  Footnote: Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar Jangan lupa untuk memasukkan input yang valid </pre>

```
>>> HELP
```

```
===== HELP =====
```

Halo Dokter Neroifa. Kamu memanggil command HELP. Kamu pasti sedang kebingungan.

Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan

DIAGNOSIS: Melakukan diagnosis penyakit pasien berdasarkan kondisi tubuh pasien

Footnote:

Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

Jangan lupa untuk memasukkan input yang valid

```
>>> LOGOUT
```

Sampai jumpa Neroifa!

```
>>> LOGIN
```

Masukan username: GRO

Masukan password: NeroifaCantik

Selamat pagi Pasien GRO! Ada keluhan apa ?

```
>>> HELP
```

```
===== HELP =====
```

Halo Pasien GRO. Kamu memanggil command HELP. Kamu pasti sedang kebingungan.

Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan

DAFTAR\_CHECKUP: Mendaftarkan diri untuk pemeriksaan dokter

Footnote:

Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

Jangan lupa untuk memasukkan input yang valid

```
>>> HELP
```

```
===== HELP =====
```

Halo Manager nimonsslatte. Kenapa kamu memanggil command HELP? Kan kamu manager, tapi yasudahlah kamu pasti sedang kebingungan.

Berikut adalah hal-hal yang dapat kamu lakukan sekarang:

LOGOUT: Keluar dari akun yang sedang digunakan

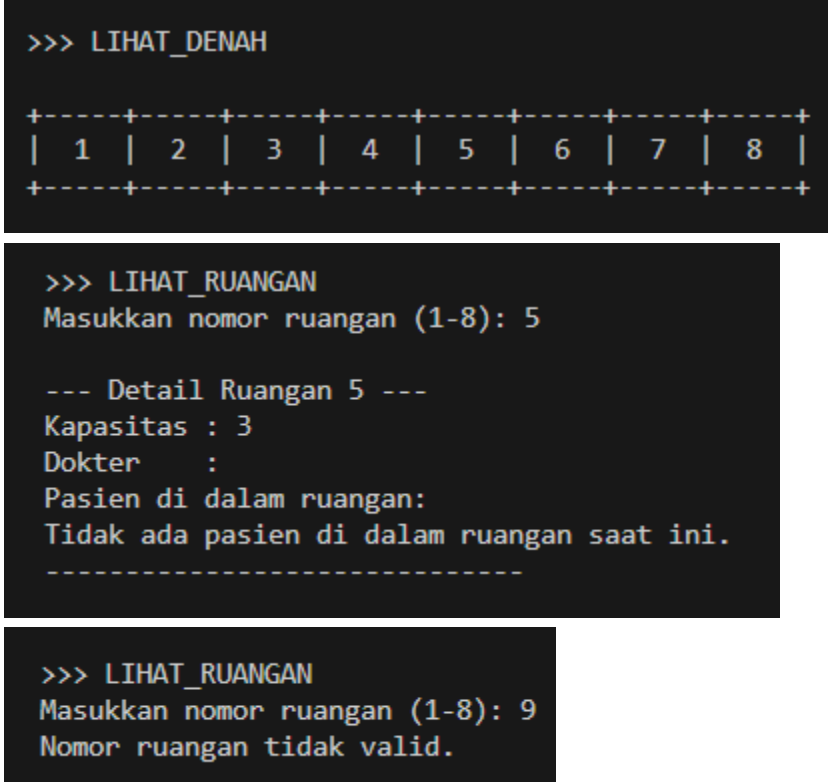
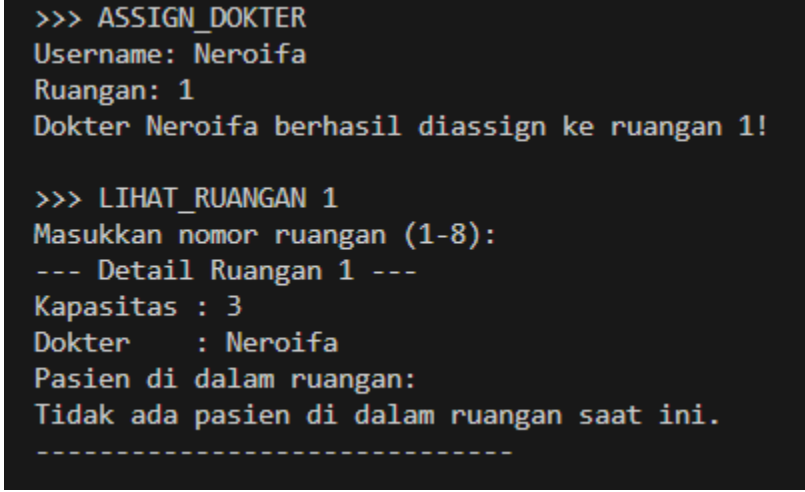
TAMBAH\_DOKTER: Mendaftarkan dokter baru ke sistem

Footnote:

Untuk menggunakan aplikasi, silahkan masukkan nama fungsi yang terdaftar

Jangan lupa untuk memasukkan input yang valid

**Gambar 40** Hasil pengujian F05-menu dan help

<p>F06 - Lihat Denah &amp; Ruang</p>	 <pre> &gt;&gt;&gt; LIHAT_DENAH  +-----+-----+-----+-----+-----+-----+-----+   1   2   3   4   5   6   7   8   +-----+-----+-----+-----+-----+-----+-----+  &gt;&gt;&gt; LIHAT_RUANGAN Masukkan nomor ruangan (1-8): 5  --- Detail Ruang 5 --- Kapasitas : 3 Dokter    : Pasien di dalam ruangan: Tidak ada pasien di dalam ruangan saat ini. -----  &gt;&gt;&gt; LIHAT_RUANGAN Masukkan nomor ruangan (1-8): 9 Nomor ruangan tidak valid. </pre> <p><b>Gambar 41</b> Hasil pengujian F06-</p>
<p>F07 -</p>	<p><b>Gambar 42</b> Hasil pengujian F07-</p>
<p>F08 -</p>	
<p>F09 -</p>	
<p>F10 - Tambah &amp; Assign Dokter</p>	 <pre> &gt;&gt;&gt; ASSIGN_DOKTER Username: Neroifa Ruang: 1 Dokter Neroifa berhasil diassign ke ruangan 1!  &gt;&gt;&gt; LIHAT_RUANGAN 1 Masukkan nomor ruangan (1-8): --- Detail Ruang 1 --- Kapasitas : 3 Dokter    : Neroifa Pasien di dalam ruangan: Tidak ada pasien di dalam ruangan saat ini. ----- </pre>

	  <p><b>Gambar 43</b> Hasil pengujian F10-</p>
F11 -	<b>Gambar 44</b> Hasil pengujian F11 -
F12 -	<b>Gambar 45</b> Hasil pengujian F12 -
F13 -	
F14 -	
F15 -	<b>Gambar 46</b> Hasil pengujian F15 -
F16 -	
F17 -	

F18 - Exit

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n)  
Pilihan Anda: N
```

```
Kembali ke menu utama...
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n)  
Pilihan Anda: n
```

```
Kembali ke menu utama...
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n)  
Pilihan Anda: Y
```

```
User keluar dari Rumah Sakit,  
Sampai jumpa!
```

```
192:tubes alpro 2025 wahyuindragunawan$
```

```
>>> LOGIN
```

```
Masukan username: GRO
```

```
Masukan password: NeroifaCantik
```

```
Selamat pagi Pasien GRO! Ada keluhan apa ?
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n)  
Pilihan Anda: n
```

```
Kembali ke menu utama...
```

```
>>> EXIT
```

```
Apakah Anda mau keluar dari rumah sakit? (y/n)  
Pilihan Anda: y
```

```
GRO keluar dari Rumah Sakit,  
Sampai jumpa!
```

```
192:tubes alpro 2025 wahyuindragunawan$
```



### III. LAMPIRAN ASISTENSI

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

Nomor Asistensi : 1  
No. Kelompok/Kelas : A/K03  
Tanggal asistensi : 2-Mei-2025

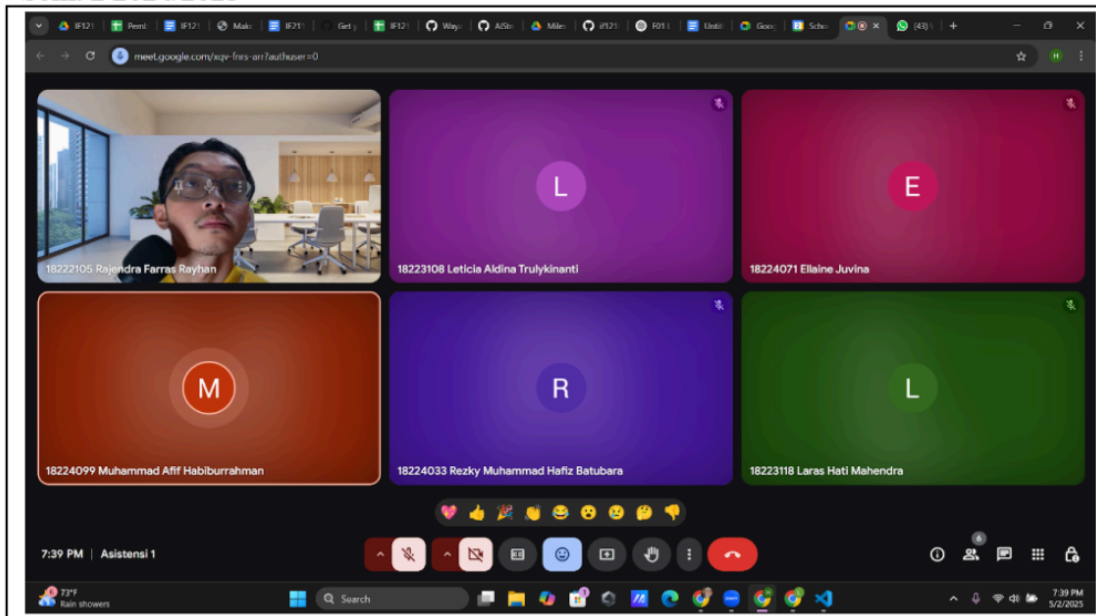
Anggota kelompok	NIM / Nama (Hanya yang Hadir)	
	1	18223108 / Leticia Aldina Trulykinanti
	2	18223118 / Laras Hati Mahendra
	3	18224033 / Rezky Muhammad Hafiz Batubara
	4	18224071 / Ellaine Juvina
	5	18224099 / Muhammad Afif Habibburrahman
Asisten pembimbing	NIM / Nama	
	18222105 / Rajendra Farras Rayhan	

Catatan Asistensi:

<b>Rangkuman Diskusi</b>
<ol style="list-style-type: none"><li>1. Penulisan header harus disesuaikan terkait alur output-inputnya data dan jangan lupa buat makefile agar bisa di run langsung.</li><li>2. Struktur file sebaiknya modular, data tidak disimpan secara hardcoded di satu file saja.</li><li>3. ADT (Abstract Data Type) disarankan disimpan dalam folder terpisah seperti adt/.</li><li>4. Fungsi gabungan diperbolehkan, contoh: antrian bisa pakai map, queue, dan linked list.</li><li>5. Milestone 1: F1-F6, F10, F18 (login, register, logout, password check, cari user, exit point).</li><li>6. Minimal progress 40% harus dicapai sebelum deadline (11 Mei 2025).</li><li>7. Laporan harus pakai ringkasan fungsi yang dijalankan.</li></ol>
<b>Tindak Lanjut</b>
<ol style="list-style-type: none"><li>1. Sesuaikan isi header dengan fungsionalitas modul terkait (misalnya: login.h, register.h, dll).</li><li>2. Pisahkan data seperti user, obat, dll ke dalam file .h masing-masing (user.h, obat.h).</li><li>3. Buat folder adt/ yang berisi file list.h, stack.h, queue.h, dll untuk struktur modular.</li><li>4. Buat dokumentasi pemanggilan antar ADT agar referensi jelas dan terstruktur.</li><li>5. Fokus implementasi fitur-fitur dasar dan buat dokumentasi per fitur yang selesai.</li><li>6. Membuat <u>tracker progress/sheets</u> <a href="#">A-K03_Pembagian Tugas</a></li><li>7. Siapkan dokumen laporan dan mulai isi deskripsi fungsi yang telah dibuat.</li></ol>
<b>Dokumentasi</b>

**Gambar 47** Form asistensi pertama

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



**Gambar 48** Form asistensi pertama