Tugas Besar IF1210

Laporan Tugas Besar IF1210

Mata Kuliah Algoritma dan Pemrograman 1
(IF1210)

Kelas: 02

Kelompok: K02-H

Yusuf Faishal Listyardi	13524014
Jason Edward Salim	13524034
Eliana Natalie Widjojo	13524116
Nathaniel Christian	13524122
Safira Berlianti	13524128

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.

Anggota kelompok:

No.	**Nama**	**NIM**
1	Yusuf Faishal Listyardi	13524014
2	Jason Edward Salim	13524034
3	Eliana Natalie Widjojo	13524116
4	Nathaniel Christian	13524122
5	Safira Berlianti	13524128

Daftar Isi

Halaman Pernyataan....2
Daftar Isi....3
Daftar Tabel....6
Daftar Gambar....7
Deskripsi Persoalan....8
Rencana Implementasi....9
Pembagian Kerja Anggota Kelompok....11
Pembagian Kerja Laporan Anggota Kelompok....13
Checklist Hasil Rancangan....14
Desain Command....15
Desain Kamus Data....17
- auth.h....17
- checkup.h....18
- command.h....18
- diagnosis.h....18
- dokter.h....20
- hospital.h....20
- manager.h....21
- matrix.h....22
- ngobatin.h....24
- obat.h....26
- stack.h....26
- utils.h....27
- user.h....27

Desain Dekomposisi Algoritmik dan Fungsional....32
- F01 – Login....32
- F02 – Register Pasien....32
- F03 – Logout....33
- F04 – Lupa Password....33
- F05 – Menu & Help....34
- F06 – Denah Rumah Sakit....34
- F07 – Lihat User....35
- F08 – Cari User....35
- F09 – Lihat Antrian....36
- F10 – Tambah & Assign Dokter....36
- F11 – Diagnosis....36
- F12 – Ngobatin....37
- F13 – Pulang Dok....37

F14 – Daftar Check-up....38
F15 – Antrian Saya!....38
F16 – Minum Obat....38
F17 – Minum Penawar....39
F18 – Exit....39
Spesifikasi....40
F01 – Login....40
F02 – Register Pasien....41
F03 – Logout....42
F04 – Lupa Password....42
F05 – Menu & Help....43
F06 – Denah Rumah Sakit....44
F07 – Lihat User....45
F08 – Cari User....46
F09 – Lihat Antrian....47
F10 – Tambah & Assign Dokter....48
F11 – Diagnosis....49
F12 – Ngobatin....50
F13 – Aku Boleh Pulang Ga, Dok?....52
F14 – Daftar Checkup....54
F15 – Antrian Saya....56
F16 – Minum Obat....57
F17 – Minum Penawar....58
F18 – Exit....58
Pengujian Program....60
F01 – LOGIN....60
F02 – REGISTER....60
F03 – LOGOUT....60
F04 – LUPA_PASSWORD....60
F05 – HELP....61
F06 – LIHAT_DENAH / LIHAT_RUANGAN....61
F07 – LIHAT_USER / LIHAT_PASIEN / LIHAT_DOKTER....62
F08 – CARI_USER / CARI_PASIEN / CARI_DOKTER....63
F09 – LIHAT_ANTRIAN....64
F10 – TAMBAH_DOKTER....65
F11 – DIAGNOSIS....65
F12 – NGOBATIN....65
F13 – PULANGDOK....66
F14 – DAFTAR_CHECKUP....66
F15 – ANTRIAN....66
F16 – MINUM_OBAT....67

F17 – MINUM_PENAWAR..... 67
F18 – EXIT.....67
Lampiran..... 68

Daftar Tabel

Tabel 1 Rencana Implementasi 10
Tabel 2 Pembagian Kerja Anggota Kelompok 12
Tabel 3 Pembagian Kerja Laporan Anggota Kelompok 13
Tabel 4 Checklist Hasil Rancangan 14
Tabel 5 Desain Command 16
Tabel 6 Pengujian Program - F01 51
Tabel 7 Pengujian Program - F02 51
Tabel 8 Pengujian Program - F03 51
Tabel 9 Pengujian Program - F04 52
Tabel 10 Pengujian Program - F05 52
Tabel 11 Pengujian Program - F06 (1) 52
Tabel 12 Pengujian Program - F06 (2) 53
Tabel 13 Pengujian Program - F07 (1) 53
Tabel 14 Pengujian Program - F07 (2) 54
Tabel 15 Pengujian Program - F07 (3) 54
Tabel 16 Pengujian Program - F08 (1) 54
Tabel 17 Pengujian Program - F08 (2) 55
Tabel 18 Pengujian Program - F08 (3) 55
Tabel 19 Pengujian Program - F09 56
Tabel 20 Pengujian Program - F10 56
Tabel 21 Pengujian Program - F11 56
Tabel 22 Pengujian Program - F12 56
Tabel 23 Pengujian Program - F13 57
Tabel 24 Pengujian Program - F14 57
Tabel 25 Pengujian Program - F15 57
Tabel 26 Pengujian Program - F16 58
Tabel 27 Pengujian Program - F17 58
Tabel 28 Pengujian Program - F18 58

Daftar Gambar

Gambar 1 Dekomposisi Algoritmik dan Fungsional - F01 32
Gambar 2 Dekomposisi Algoritmik dan Fungsional - F02 32
Gambar 3 Dekomposisi Algoritmik dan Fungsional - F03 33
Gambar 4 Dekomposisi Algoritmik dan Fungsional - F04 33
Gambar 5 Dekomposisi Algoritmik dan Fungsional - F05 33
Gambar 6 Dekomposisi Algoritmik dan Fungsional - F06 34
Gambar 7 Dekomposisi Algoritmik dan Fungsional - F07 34
Gambar 8 Dekomposisi Algoritmik dan Fungsional - F08 35
Gambar 9 Dekomposisi Algoritmik dan Fungsional - F09 35
Gambar 10 Dekomposisi Algoritmik dan Fungsional - F10 36
Gambar 11 Dekomposisi Algoritmik dan Fungsional - F11 36
Gambar 12 Dekomposisi Algoritmik dan Fungsional - F12 36
Gambar 13 Dekomposisi Algoritmik dan Fungsional - F13 37
Gambar 14 Dekomposisi Algoritmik dan Fungsional - F14 37
Gambar 15 Dekomposisi Algoritmik dan Fungsional - F15 37
Gambar 16 Dekomposisi Algoritmik dan Fungsional - F16 38
Gambar 17 Dekomposisi Algoritmik dan Fungsional - F17 38
Gambar 18 Dekomposisi Algoritmik dan Fungsional - F18 38

Deskripsi Persoalan

Tugas besar ini meminta kita untuk merancang program sistem rumah sakit. Sistem rumah sakit harus memiliki fitur otentikasi yang dapat mengelola pengguna dengan *role* yang berbeda-beda seperti pasien, dokter, dan manager, manajemen data pengguna serta penyimpanannya pada file eksternal, pengelolaan ruangan dan antrean, diagnosis dan pengobatan yang interaktif, dan disusun dengan pendekatan *modular programming* dan penggunaan ADT.

Rencana Implementasi

Implementasi ADT	Fitur	Deskripsi Implementasi	Alasan Implementasi
ADT Sederhana	D03 – Load	Menyimpan data pengguna dalam ADT Sederhana User.	Mempermudah akses data pengguna.
ADT List	F07 – Lihat User, F08 – Cari User, F11 – Diagnosis	Menyimpan daftar data pengguna / penyakit ADT Sederhana User / Penyakit dalam ADT List.	Mempermudah akses daftar data pengguna untuk fitur seperti Lihat User dan Cari User serta daftar data penyakit.
ADT Linked List		Digunakan pada ADT Queue	Untuk membuat ADT Queue dinamis
ADT Matrix	F06 / D01 – Denah Rumah Sakit	Untuk menyimpan info pada setiap ruangan yang ada pada rumah sakit, info berupa (nama ruangan, id dokter,id pasien, dan lain lain)	Untuk mempermudah print denah sesuai spesifikasi dan mempemudah akses
ADT Set	F02 – Register Pasien, F10 – Tambah Dokter	Menyimpan daftar data pengguna dengan role spesifik dalam ADT Set	Untuk mempermudah pencarian *credential* yang sama dengan input.
ADT Map	F12 – Ngobatin	Menyimpan daftar penyakit beserta obatnya	Untuk mempermudah pencarian obat berdasarkan penyakit.
ADT Stack	F13 – Pulang Dok, F16 – Minum Obat, F17 – Minum Penawar	Pada F16, setelah pasien memilih obat untuk diminum, obat dipindahkan dari inventory ke stack perut. Pada F17, obat yang terakhir diminum (berada pada top stack) akan dikeluarkan dari stack	Obat yang diminum menggunakan struktur LIFO (Last in First Out) dan F17 mengharuskan obat yang terakhir diminum keluar.

		perut dan dikembalikan ke inventory. Pada F13. stack perut pasien akan dibandingkan dengan urutan obat yang seharusnya dikonsumsi.	
ADT Queue	F09 – Lihat Antrian, F14 – Daftar Check-Up, F15 – Antrian Saya!	Melihat daftar antrian pasien di setiap ruangan, mendaftar check-up ke dokter, melihat status antrian pribadi pasien	Mempermudah aplikasi fitur
File External	D03 – Load, D04 – Save	Membaca file user.csv dan config.txt pada suatu folder.	Untuk membaca, mengubah, dan menyimpan file eksternal.
Fungsi & Prosedur	Semua Fitur	Menyimpan algoritma yang membantu pengerjaan.	Mempermudah pengerjaan dengan pendekatan modular.
Array Search, Sort, Filter	F07 – Lihat User, F08 – Cari User	Mencari, mengurut, dan menyaring suatu array	Pada fitur Lihat User, kita memfilter daftar pengguna agar tidak ada manager di dalamnya dan juga terdapat fitur *sort* secara *ascending*/*descending* untuk menampilkan daftar user pada CLI. Pada fitur Cari User, dilakukan metode array search berdasarkan ID, Nama, atau pun penyakit (khusus pasien).

Tabel 1 Rencana Implementasi

Pembagian Kerja Anggota Kelompok

Fitur	Implementasi	NIM Desainer	NIM Coder	NIM Tester
F01 – Login	Prosedur Login, ADT List	13524034 13524122	13524034 13524122	13524034 13524122
F02 – Register Pasien	Prosedur RegisterUser, ADT Set	13524034 13524122	13524034 13524122	13524034 13524122
F03 – Logout	Prosedur Logout	13524034 13524122	13524034 13524122	13524034 13524122
F04 – Lupa Password	Prosedur ResetPassword	13524122	13524122	13524122
F05 – Menu & Help	Prosedur Help	13524034	13524034	13524034
F06 – Denah Rumah Sakit	Prosedur LihatDenah, ADT Matriks	13524014	13524014	13524014
F07 – Lihat User	Prosedur LihatUser, LihatPasien, LihatDokter	13524122	13524122	13524122 13524034
F08 – Cari User	Prosedur CariUser, CariPasien, CariDokter, ADT List	13524122	13524122	13524122 13524034
F09 – Lihat Antrian	ADT Queue + Linked List	13524034	13524034	13524034
F10 – Tambah Dokter	Prosedur TambahDokter, ADT Set	13524034 13524122	13524034 13524122	13524034 13524122
F11 – Diagnosis	Prosedur Diagnosis, ADT Map	13524034	13524034	13524034
F12 – Ngobatin	ADT Map	13524034	13524034	13524034
F13 – Pulang Dok 🥺 ?	Prosedur PulangDok, ADT Stack	13524128	13524128	13524128 13524034
F14 – Daftar Check-Up	Prosedur DaftarCheckup, ADT Map, ADT List	13524116 13524122	13524116 13524122	13524116 13524122

F15 – Antrian Saya!	Prosedur LihatAntrianSaya, ADT Queue	13524116 13524122	13524116 13524034 13524122	13524116 13524034
F16 – Minum Obat	Prosedur MinumObat, ADT Stack	13524128	13524128	13524128 13524034
F17 – Minum Penawar	Prosedur MinumPenawar, ADT Stack	13524128	13524128	13524128 13524034
F18 – Exit	Switch case EXIT pada main.c	13524034	13524034	13524034
Load	Prosedur LoadUsers, LoadConfig, LoadObat, LoadObatMap, LoadPenyakit, ADT List, Matrix, Map, dan File Eksternal	13524034 13524014	13524034 13524014	13524034 13524014
Save	Prosedur SaveUsers dan SaveConfig, File Eksternal	13524034 13524014	13524034 13524014	13524034 13524014

Tabel 2 Pembagian Kerja Anggota Kelompok

Pembagian Kerja Laporan Anggota Kelompok

Bagian Laporan	Nama	NIM
Desain Kamus Data	Safira Berlianti	13524128
Desain Dekomposisi Algoritmik dan Fungsional	Yusuf Faishal Listyardi	13524014
Spesifikasi	Nathaniel Christian Eliana Natalie Widjojo	13524122 13524116
Pengujian Program	Jason Edward Salim	13524034

Tabel 3 Pembagian Kerja Laporan Anggota Kelompok

Checklist Hasil Rancangan

Fitur	Desain	Implementasi	Testing
F01 – Login	V	V	V
F02 – Register Pasien	V	V	V
F03 – Logout	V	V	V
F04 – Lupa Password	V	V	V
F05 – Menu & 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 Antrian	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 – Antrian Saya!	V	V	V
F16 – Minum Obat	V	V	V
F17 – Minum Penawar	V	V	V
F18 – Exit	V	V	V
D01 – D04	V	V	V
B01	V	V	V

Tabel 4 Checklist Hasil Rancangan

Keterangan: V: sudah selesai dikerjakan, X: dikerjakan, tapi belum selesai, -: tidak dikerjakan sama sekali.

Desain *Command*

Nama *Command*	Masukan	Keluaran
F01 - LOGIN	Username, password	Masuk ke session user sesuai credential yang tepat, gagal jika credential tidak cocok.
F02 - REGISTER	Username, password	User baru yang terdaftar pada daftar user.
F03 - LOGOUT	-	Keluar dari session user spesifik.
F04 - LUPA_PASSWORD	Username, kode unik RLE	Password untuk username direset jika kode unik benar.
F05 - HELP	-	Daftar command yang dapat digunakan untuk role spesifik.
F06 - LIHAT_DENAH	-	Denah rumah sakit yang sesuai dengan config.txt
F07 - LIHAT_USER / LIHAT_PASIEN / LIHAT_DOKTER	Referensi pengurutan (ID/Nama) dan (Ascending/Descending) dalam bentuk integer	Daftar user / pasien / dokter sesuai referensi pengurutan yang diinput.
F08 - CARI_USER / CARI_PASIEN / CARI_DOKTER	Input berdasarkan (ID/Nama) atau Penyakit (Khusus Pasien)	Data user (jika ada) berdasarkan input.
F09 - LIHAT_SEMUA_ANTRIAN	-	Tampilan denah diikuti dengan detail ruang yang tidak kosong beserta antrian
F10 - TAMBAH_DOKTER	Input username yang belum ada dan password	User dengan role dokter terdaftarkan.
F10 - ASSIGN_DOKTER	Username dokter dan ruangan	Assign dokter ke ruangan
F11 - DIAGNOSIS	-	Penyakit yang diderita pasien
F12 - NGOBATIN	-	Daftar urutan obat berdasarkan penyakit pasien

F13 - PULANGDOK	-	Validasi pasien boleh pulang atau tidak. Jika pasien sudah boleh pulang, kembali ke menu utama dan pasien meninggalkan ruangan.
F14 - DAFTAR_CHECKUP	Data medis dasar pasien dan pilihan dokter dari daftar yang tersedia	Penempatan dalam antrian dokter
F15 - ANTRIAN	-	Status antrian user yang berisikan dokter, ruangan, dan posisi antrian.
F16 - MINUM_OBAT	Nomor obat yang akan diminum dari daftar obat yang tersedia	Obat dipindahkan dari inventory ke stack perut
F17 - MINUM_PENAWAR	-	Obat pada top stack perut dikembalikan ke inventory jika stack tidah kosong.
F18 - EXIT	Input folder untuk penyimpanan	File tersimpan pada folder

Tabel 5 Desain Command

Desain Kamus Data

auth.h

```
constant MAX_USERS : integer = 100
constant MAX_SET_SIZE : integer = 200
constant MAX_UNIQUE_CODE : integer = 50

type Set : <
      username : array [0..MAX_SET_SIZE-1] of string,
      count : integer
>

function CreateNewSet() → Set
{ Membuat set baru, count = 0 }

procedure InsertSet (input/output set : Set, input value : string)
{ Insert elemen ke Set }

function IsInSet (set : Set, value : string) → boolean
{ Cek apakah value sudah ada di Set }

procedure Login (input userList : UserList, input/output session : Session)
{ Login user }

procedure RegisterUser (userList : UserList, Session *session)
{ Daftar user, khusus pasien }

procedure TambahDokter  (input userList : UserList, input session: Session)
{ Menambahkan user, digunakan manager }

procedure Logout (input/output session : Session)
{ Logout user }

procedure ResetPassword (input/output userList : UserList, input session:
Session)
{ Reset password }
```

checkup.h

```
procedure InputDataMedis(output suhu : real, output sistolik : integer, output diastolik : integer, output detak : integer, output saturasi : real, output gula : integer, output berat : real, output tinggi : integer, output kolesterol : integer, output trombosit : integer);
{ Fungsi validasi input data medis }

procedure DaftarCheckup(input userList : UserList, input/output session : Session, input/output denahRumahSakit : Matrix);
{ Fungsi untuk memasukkan data medis pasien serta masuk ke queue dokter }

procedure LihatAntrianSaya(input userList : UserList, input session : Session, input denahRumahSakit : Matrix);
{ Pasien yang sudah melakukan pendaftaran dapat melihat status antriannya }
```

command.h

```
constant COMMAND_CAPACITY : integer = 17

type Command : <
      name : string,
      key : integer
>

{ Definisi ADT List Statik untuk List Command }
type CommandList : <
      command : array [0..COMMAND_CAPACITY-1] of Command
>

{ KONSTRUKTOR }
procedure CreateCommandList(output commandList : CommandList, input COMMAND_READY[] : string)
{ Membuat List Statik yang berisikan command yang dapat digunakan }
```

diagnosis.h

```
constant PENYAKIT_CAPACITY : integer = 500

{ Definisi ADT Sederhana Penyakit }
type Penyakit : <
```

```
      id : integer,
      namaPenyakit : string,
      suhuTubuhMin : real,
      suhuTubuhMax : real,
      tekananDarahSistolikMin : integer,
      tekananDarahSistolikMax : integer,
      tekananDarahDiastolikMin : integer,
      tekananDarahDiastolikMax : integer,
      detakJantungMin : integer,
      detakJantungMax : integer,
      saturasiOksigenMin : real,
      saturasiOksigenMax : real,
      kadarGulaDarahMin : integer,
      kadarGulaDarahMax : integer,
      beratBadanMin : real,
      beratBadanMax : real,
      tinggiBadanMin : integer,
      tinggiBadanMax : integer,
      kadarKolesterolMin : integer,
      kadarKolesterolMax : integer,
      trombositMin : integer,
      trombositMax : integer
>

type PenyakitList : <
      penyakit : array [0..PENYAKIT_CAPACITY-1] of Penyakit,
      nEff : integer
>

procedure LoadPenyakit(output penyakitList : PenyakitList, input inputFolder
: string)
{ Membaca file eksternal penyakit.csv dan menuliskannya ke dalam List
penyakitList }

procedure Diagnosis(input user : User, input penyakitList : PenyakitList,
output namaPenyakit : string)
{ Mendiagnosis apakah user terjangkit suatu penyakit atau tidak }

function GetPenyakitID(penyakitList : PenyakitList, namaPenyakit : string) →
integer
{ Mengembalikan ID Penyakit berdasarkan nama penyakit }
```

```
procedure SearchRuangan(input doctorId : integer, input denahHospital :
Matrix, output indeksRuangan : integer)
{ Mengembalikan index ruangan yang berisi pasien yang ditangani dokter yang
sedang login }

procedure SavePenyakit(input folderAsal : string, input folderTujuan :
string)
{ Menyimpan penyakit pasien ke file config.txt }
```

dokter.h

```
type Node : < data : integer, next : Node >
type Queue : < head : Node, tail : Node, length : integer >
{ Node primitives )

function createNode (value : integer) → Node
{ Membuat node baru dengan data = value, next = NULL }

procedure createQueue (output q : Queue)
{ Menginisialisasi queue kosong: head = NULL, tail = NULL, length = 0 }

procedure enqueue (input/output q : Queue, input newNode : Node)
{ Menambahkan newNode ke akhir queue }

function dequeue (input/output q : Queue) → integer
{ Menghapus node dari depan queue dan mengembalikan datanya }

function isEmptyQueue (input q : Queue) → boolean
{ Mengecek apakah queue kosong }

procedure freeQueue(input/output q : Queue)
{ Menghapus semua node dalam queue untuk menghindari memory leak }
```

hospital.h

```
procedure LoadConfig (output denahHospital : Matrix, input inputFolder :
string, output userlist : UserList)
{ Membaca file config.txt dan mengisi data denahHospital serta userlist }
```

```
procedure UbahInput (input input : string, output row : integer, output col : integer)
{Menentukan baris dan kolom yang sesuai dengan input ruangan }

procedure LihatDenah (input denahHospital : Matrix)
{ Menampilkan denah rumah sakit }

procedure LihatRuangan (input denahHospital : Matrix, input input : string, input userList : UserList)
{ Menampilkan detail ruangan tertentu }

procedure SaveConfig (input denahHospital : Matrix, input inputFolder : string, input userList : UserList)
{ Menyimpan konfigurasi denahHospital dan userList ke file config.txt }
```

manager.h

```
procedure LihatUser (input userList : UserList, input session : Session)
{ Hanya digunakan oleh manager, untuk melihat user, sort berdasarkan id atau nama }

procedure LihatPasien (input userList : UserList, input session : Session)
{ Hanya digunakan oleh manager, untuk melihat pasien, sort berdasarkan id atau nama }

procedure LihatDokter (input userList : UserList, input session : Session)
{ Hanya digunakan oleh manager, untuk melihat dokter, sort berdasarkan id atau nama }

procedure SelectionSort (input/output userList : UserList, input n, basedOn, order : integer)
{ Sorting dengan selection sort, berdasarkan id atau nama }

procedure PrintPilihan (input pil1, pil2 : integer)
{ Function bantuan untuk menampilkan pilihan }

procedure PrintList (input userList : UserList, input basedOn, order : integer)
{ Print userlist }

procedure CariUser (input userList : UserList, input session : Session)
```

```
{ Cari user dengan binary search }

procedure CariPasien (input userList : UserList, input session : Session)
{ Cari pasien dengan binary search }

procedure CariDokter (input userList : UserList, input session : Session)
{ Cari dokter dengan binary search }

function BinarySearchUser (userList : UserList, id : integer, index :
integer) → boolean
{ searching user dengan binary search → untuk search IDE }

function SequenceSearchUser (userList : UserList, username : string, index :
integer) → boolean
{ searching user dengan sequence search → untuk search username }

function GetUserAt (userList : UserList, idx : integer) → User
{ Get user from UserList by indeks }

function GetUserById (userList : UserList, userId : integer) → User
{ Get user from UserList by its id }

procedure SetUserAt (input/output userList : UserList, input idx : integer,
input user : User)
{ Set user in UserList by indeks }

function AppendUser (userList : UserList, user : User) → boolean
{ Add user to UserList (if not full) }

procedure AssignDokter(input userList: UserList, input/output
denahRumahSakit: Matrix)
{ Assign dokter ke ruangan tertentu }
```

matrix.h

```
type Ruangan : <
      jumlahPasienDalamRuangan : integer,
      jumlahPasienDiAntrian : integer,
      dokter : integer,
      kapasitasRuangan : integer,
      kapasitasAntrian : integer,
```

```
      namaRuangan : string,
      antrianPasien : Queue
>

type Matrix :
      < data : matrix [0..MAX_ROWS-1, 0..MAX_COLS-1] of Ruangan,
      rows : integer,
      cols : integer >

procedure CreateMatrix (input rows : integer, input cols : integer, output M
: Matrix)
{ Membuat matrix baru dengan ukuran rows x cols }

function isRowValid (rows : integer, M : Matrix) → boolean
{ Mengembalikan true jika baris valid pada Matrix M, dan sebaliknya }

function isColsValid (cols : integer, M : Matrix) → boolean
{ Mengembalikan true jika kolom valid pada Matrix M, dan sebaliknya }

function GetRows (M : Matrix) → integer
{ Mengembalikan jumlah baris matrix M }

function GetCols (M : Matrix) → integer
{ Mengembalikan jumlah kolom matrix M }

procedure FindDokter (input M : Matrix, output row : integer, output col :
integer, output namaRuangan : string, input dokterId : integer)
{ Mencari lokasi dokter dalam matrix dan mengembalikan posisi serta nama
ruangannya }

function GetRuangan (M : Matrix, row : integer, col : integer) → Ruangan
{ Mengembalikan nilai elemen pada baris row dan kolom col }

function SetElement (M : Matrix, row : integer, col : integer, value :
Ruangan) → boolean
{ Mengubah nilai elemen pada baris row dan kolom col, return true jika
berhasil }

procedure InisialisasiNamaRuangan (input/output M : Matrix)
{ Inisialisasi semua elemen matrix M dengan <Huruf><Angka> }
```

```
procedure printAntrianRuangan (input ruangan : Ruangan, input userList :
UserList)
{ Mencetak daftar pasien dalam antrian ruangan }
```

ngobatin.h

```
constant OBAT_CAPACITY : integer = 500
constant VAL_UNDEF : string = "-"
constant STR_LENGTH : integer = 500

{ Definisi ADT Map Obat-Penyakit beserta ADT List Obat }

{ Elemen Map berdasarkan penyakitId }
type ObatEntry : <
      penyakitId : integer,
      obatId : array[0..99] of integer,
      urutan : integer
>

{ Map }
type ObatMap : <
      buffer : array[0..499] of ObatEntry,
      length : integer,
      banyakObat : integer
>

{ List }
type ObatList : <
      obats : array[0..OBAT_CAPACITY-1] of Obat,
      length : integer
>

{ Obat )
type Obat : <
      id : integer,
      nama : string
>

procedure CreateObatMap(output obatMap : ObatMap)
{ Membuat ObatMap kosong }
```

```
procedure CreateObatList(output obatList : ObatList)
{ Membuat ObatList kosong }

function ObatMapLength(obatMap : ObatMap) → integer
{ Mereturn panjang dari obatMap }

function ObatListLength(obatList : ObatList) → integer
{ Mereturn panjang dari obatList }

function GetObatEntry(obatMap : ObatMap, i : integer) → ObatEntry
{ Mereturn Entry obat pada indeks ke-i }

function GetObatById(obatList : ObatList, idx : integer) → Obat
{ Mereturn obat sesuai dengan id yang diberikan }

function GetObatByName(obatList : ObatList, namaPenyakit : string) → Obat
{ Mereturn obat seusai dengan nama yang diberikan }

function SearchObatIndex(obatMap : ObatMap, penyakit : string) → integer
{ Mencari indeks dari obat berdasarkan nama penyakit }

procedure PrintObat(input obatMap : ObatMap, input penyakitId : integer,
input obatList : ObatList, input namaPenyakit : string, output
arrayUrutanObat[][500] : string, input/output pasien : User)
{ Mencetak obat-obatan yang harus dikonsumsi berdasarkan penyakit }

procedure LoadObat(output obatList : ObatList, input inputFolder : string)
{ Membaca file eksternal obat.csv dan menuliskannya ke dalam List obattList }

procedure LoadObatMap(output obatMap : ObatMap, input inputFolder : string)
{ Membaca file eksternal obat_penyakit.csv dan menuliskannya ke dalam Map
obattMap }

procedure SaveObat(input folderAsal : string, input folderTujuan : string)
{ Menyimpan urutan obat dan stack perut pasien ke file config.txt }

procedure MinumObat(input obatList : ObatList, input/output user : User,
input arrayNamaObat[][500] : string)
```

obat.h

```
procedure MinumObat(input/output user : User)
{ Prosedur untuk minum obat }

procedure MinumPenawar(input/output user : User)
{ Prosedur untuk minum penawar }

function IsUrut(urutanObat : ObatList, perut : Stack) → boolean
{ Mengecek apakah obat yang diminum sesuai dengan urutan }

procedure PulangDok(input/output user : User, input urutanObat, obatList :
ObatList)
{ Prodedur untuk F13 }
```

stack.h

```
constant MAX_OBAT_LENGTH : integer = 50
constant MAX_STACK_SIZE : integer = 100
constant IDX_UNDEF : integer = -1

procedure CreateEmptyStack (output s : Stack)
{ Membuat stack kosong }

function IsStackEmpty (input s : Stack) → boolean
{ Mengecek apakah stack kosong }

function IsStackFull (input s : Stack) → boolean
{ Mengecek apakah stack penuh }

procedure PrintStackObat (input s : Stack)
{ Mencetak isi stack }

procedure Push (input/output s : Stack, input val : Obat)
{ Menambahkan elemen ke top stack }

procedure Pop (input/output s : Stack, output val : Obat)
{ Mengambil elemen dari top stack }
```

utils.h

```
procedure ToLower (output target : string, input str : string)
{ Membuat copy dari suatu string yang di lowercase }

procedure ToLowerCase (input/output str : string)
{ Make lowercase string }

procedure ToUpperCase (input/output str : string)
{ Make uppercase string }

function RealToStr (x : real) → string
{ Mengembalikan real dalam bentuk string }

function IntToStr (x : integer) → string
{ Mengembalikan integer dalam bentuk string }

procedure Help (input session : Session)
{ Memberikan tampilan Help berdasarkan session (sudah login atau belum) }

procedure RunLenEncode (input str : string, output encoded : string)
{ Untuk fitur reset password }
```

user.h

```
constant MAX_USERNAME_LENGTH : integer = 50
constant MAX_PASSWORD_LENGTH : integer = 50

{ Definisi ADT User }
type User : <
      id : integer,
      username : string,  { case-insensitive }
      password : string,
      role : string,      { "manager"/"dokter"/"pasien" }
      riwayatPenyakit : string,
      diagnosa : integer,
      ngobatin : integer,
```

```
      obat : array[0..99] of integer,        { inventory obat }
      urutanNgobat : array[0..99] of integer,{ urutan konsumsi obat }
      jumlahNgobat : integer,
      jumlahObat : integer,
      perut : Stack,             { obat yang sudah dimakan }
      jumlahObatMasukPerut : integer,

      { Data kesehatan, -1 jika tidak ada data }
      suhuTubuh : real,
      tekananDarahSistolik : integer,
      tekananDarahDiastolik : integer,
      detakJantung : integer,
      saturasiOksigen : real,
      kadarGulaDarah : integer,
      beratBadan : real,
      tinggiBadan : integer,
      kadarKolesterol : integer,
      trombosit : integer
>

type UserList : <
      users : array[0..99] of User,
      count : integer,
      pasienDenganObat : integer,
      pasienKondisiPerut : integer
>

type Session : <
      loggedIn : integer,    { 1 = sudah login, 0 = belum }
      currentUser : User
>

{ KONSTRUKTOR }
procedure CreateUser(
    output user : User,
    input id : integer,
    input username, password, role, riwayatPenyakit : string,
    input suhuTubuh : real,
    input tekananDarahSistolik, tekananDarahDiastolik : integer,
    input detakJantung : integer,
```

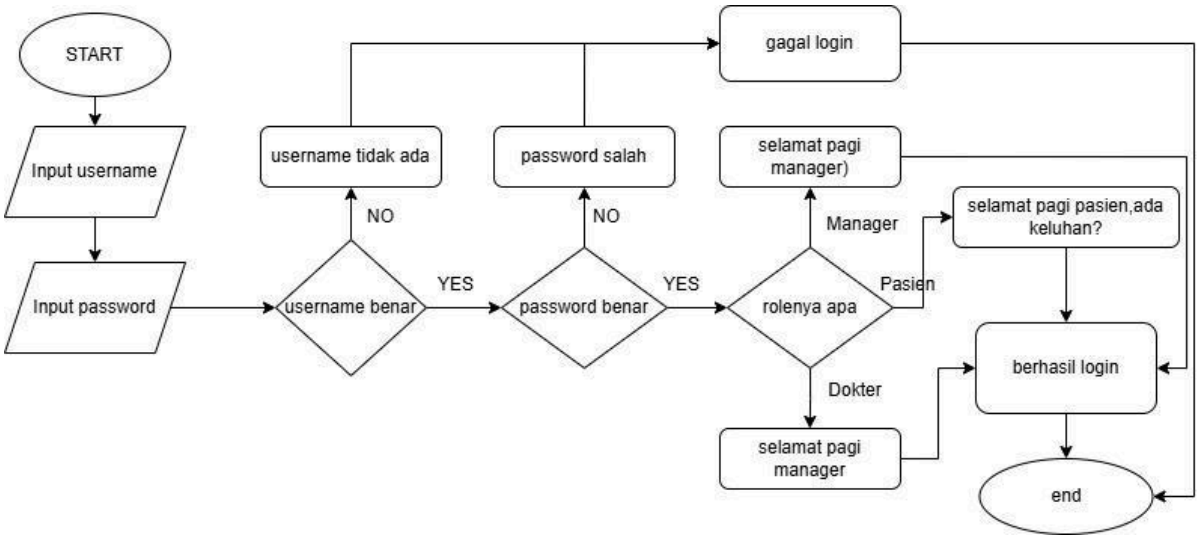
```
    input saturasiOksigen : real,
    input kadarGulaDarah : integer,
    input beratBadan : real,
    input tinggiBadan, kadarKolesterol, trombosit : integer
)
{ Membentuk user berdasarkan komponen-komponen yang dimasukkan }

{ SELEKTOR / GETTER }
function GetID (user : User) → integer
{ Mendapatkan komponen ID dari user }

function GetUsername (user : User) → string
{ Mendapatkan komponen Username dari user }

function GetPassword (user : User) → string
{ Mendapatkan komponen Password dari user }

function GetRole (user : User) → string
{ Mendapatkan komponen Role dari user }

function GetRiwayatPenyakit (user : User) → string
{ Mendapatkan komponen RiwayatPenyakit dari user }

function GetSuhuTubuh (user : User) → real
{ Mendapatkan komponen SuhuTubuh dari user }

function GetTekananDarahSistolik (user : User) → integer
{ Mendapatkan komponen TekananDarahSistolik dari user }

function GetTekananDarahDiastolik (user : User) → integer
{ Mendapatkan komponen TekananDarahDiastolik dari user }

function GetDetakJantung (user : User) → integer
{ Mendapatkan komponen DetakJantung dari user }

function GetSaturasiOksigen (user : User) → real
{ Mendapatkan komponen SaturasiOksigen dari user }

function GetKadarGulaDarah (user : User) → real
{ Mendapatkan komponen KadarGulaDarah dari user }
```

```
function GetBeratBadan (user : User) → real
{ Mendapatkan komponen BeratBadan dari user }

function GetTinggiBadan (user : User) → integer
{ Mendapatkan komponen TinggiBadan dari user }

function GetKadarKolesterol (user : User) → integer
{ Mendapatkan komponen KadarKolesterol dari user }

function GetKadarKolesterolLDL (user : User) → integer
{ Mendapatkan komponen KadarKolesterolLDL dari user }

function GetTrombosit (user : User) → integer
{ Mendapatkan komponen Trombosit dari user }

{ PENGUBAH / SETTER }
procedure SetID (input/output user : User, input val : integer)
{ Mengubah nilai komponen ID dari user menjadi val }

procedure SetUsername (input/output user : User, input val : string)
{ Mengubah nilai komponen Username dari user menjadi val }

procedure SetPassword (input/output user : User, input val : string)
{ Mengubah nilai komponen Password dari user menjadi val }

procedure SetRole (input/output user : User, input val : string)
{ Mengubah nilai komponen Role dari user menjadi val }

procedure SetRiwayatPenyakit (input/output user : User, input val : string)
{ Mengubah nilai komponen RiwayatPenyakit dari user menjadi val }

procedure SetSuhuTubuh (input/output user : User, input val : real)
{ Mengubah nilai komponen SuhuTubuh dari user menjadi val }

procedure SetTekananDarahSistolik (input/output user : User, input val :
integer)
{ Mengubah nilai komponen TekananDarahSistolik dari user menjadi val }

procedure SetTekananDarahDiastolik (input/output user : User, input val :
integer)
```

```
{ Mengubah nilai komponen TekananDarahDiastolik dari user menjadi val }

procedure SetDetakJantung (input/output user : User, input val : integer)
{ Mengubah nilai komponen DetakJantung dari user menjadi val }

procedure SetSaturasiOksigen (input/output user : User, input val : real)
{ Mengubah nilai komponen SaturasiOksigen dari user menjadi val }

procedure SetKadarGulaDarah (input/output user : User, input val : integer)
{ Mengubah nilai komponen KadarGulaDarah dari user menjadi val }

procedure SetBeratBadan (input/output user : User, input val : real)
{ Mengubah nilai komponen BeratBadan dari user menjadi val }

procedure SetTinggiBadan (input/output user : User, input val : integer)
{ Mengubah nilai komponen TinggiBadan dari user menjadi val }

procedure SetKadarKolesterol (input/output user : User, input val : integer)
{ Mengubah nilai komponen KadarKolesterol dari user menjadi val }

procedure SetKadarKolesterolLDL (input/output user : User, input val :
integer)
{ Mengubah nilai komponen KadarKolesterolLDL dari user menjadi val }

procedure SetTrombosit (input/output user : User, input val : integer)
{ Mengubah nilai komponen Trombosit dari user menjadi val }

procedure AddUser (input/output userList : UserList, input newUser : User)
{ Menambahkan user ke dalam array userList }

{ KELOMPOK OPERASI BACA / TULIS FILE EKSTERNAL }
procedure LoadUsers (input/output userList : UserList, input inputFolder :
string)
{ Membaca file eksternal dan memasukkan data user yang terdaftar ke dalam
userList }

procedure SaveUsers (input/output userList : UserList, input inputFolder :
string)
{ Menyimpan array userList ke dalam file eksternal user.csv }
```

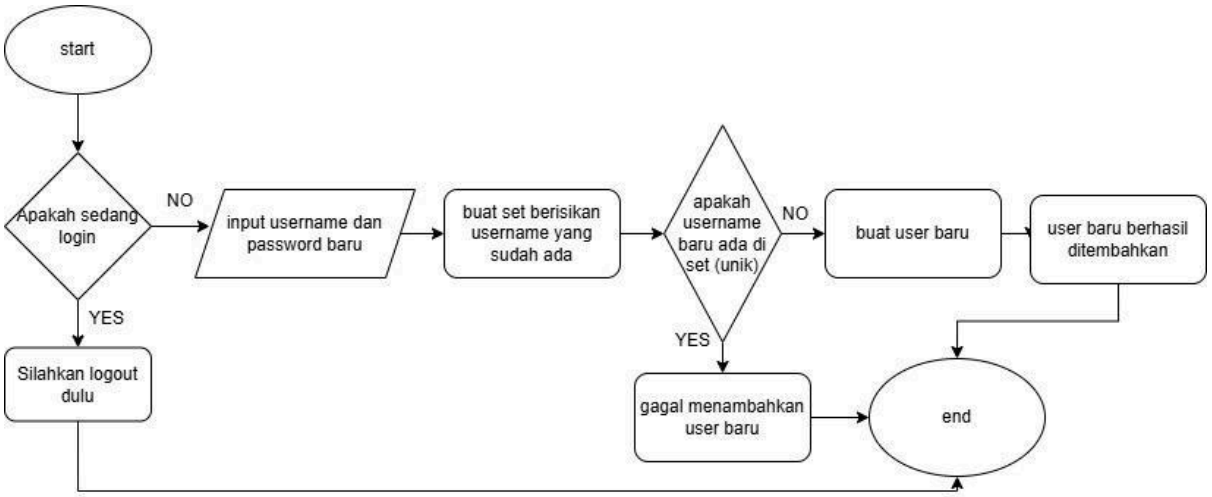
Desain Dekomposisi Algoritmik dan Fungsional

F01 – Login



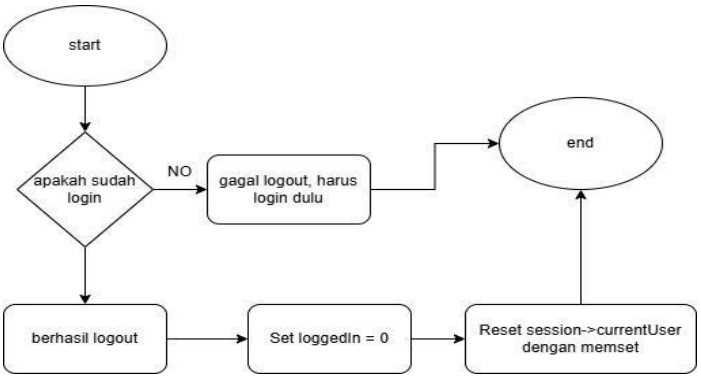
Gambar 1 Dekomposisi Algoritmik dan Fungsional - F01

F02 – Register Pasien



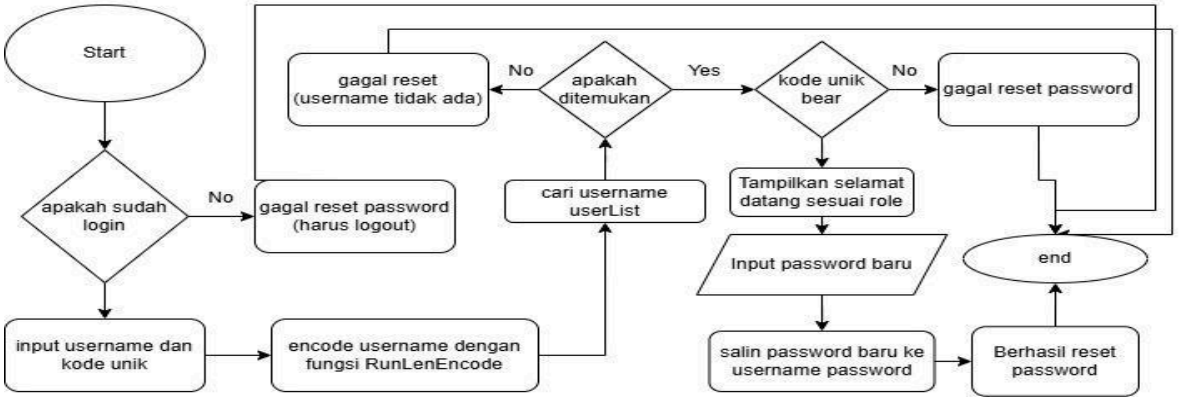
Gambar 2 Dekomposisi Algoritmik dan Fungsional - F02

F03 – Logout



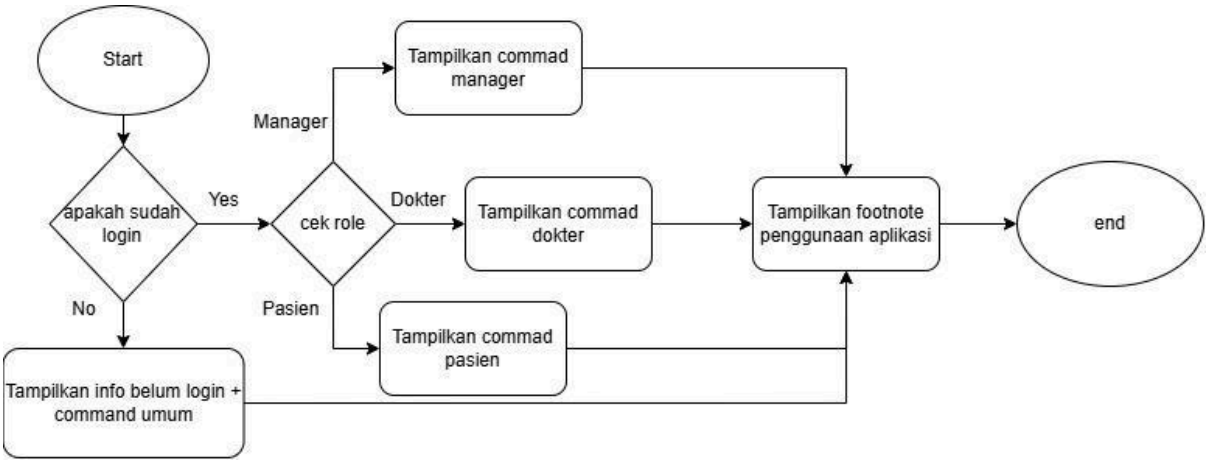
Gambar 3 Dekomposisi Algoritmik dan Fungsional - F03

F04 – Lupa Password



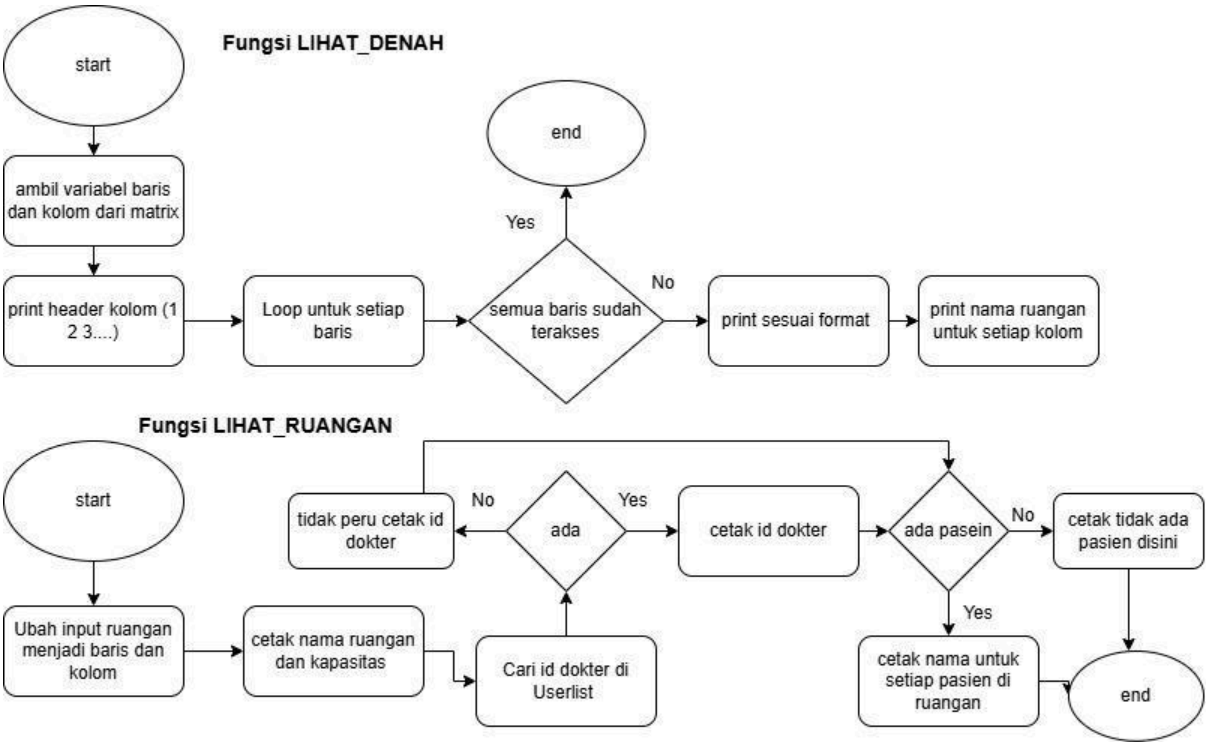
Gambar 4 Dekomposisi Algoritmik dan Fungsional - F04

F05 – Menu & Help



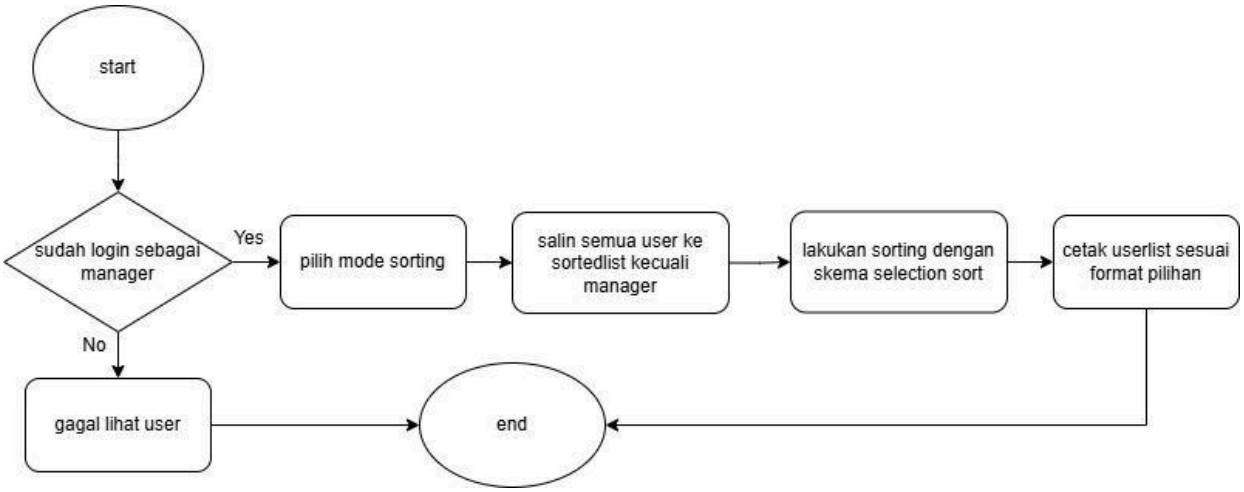
Gambar 5 Dekomposisi Algoritmik dan Fungsional - F05

F06 – Denah Rumah Sakit



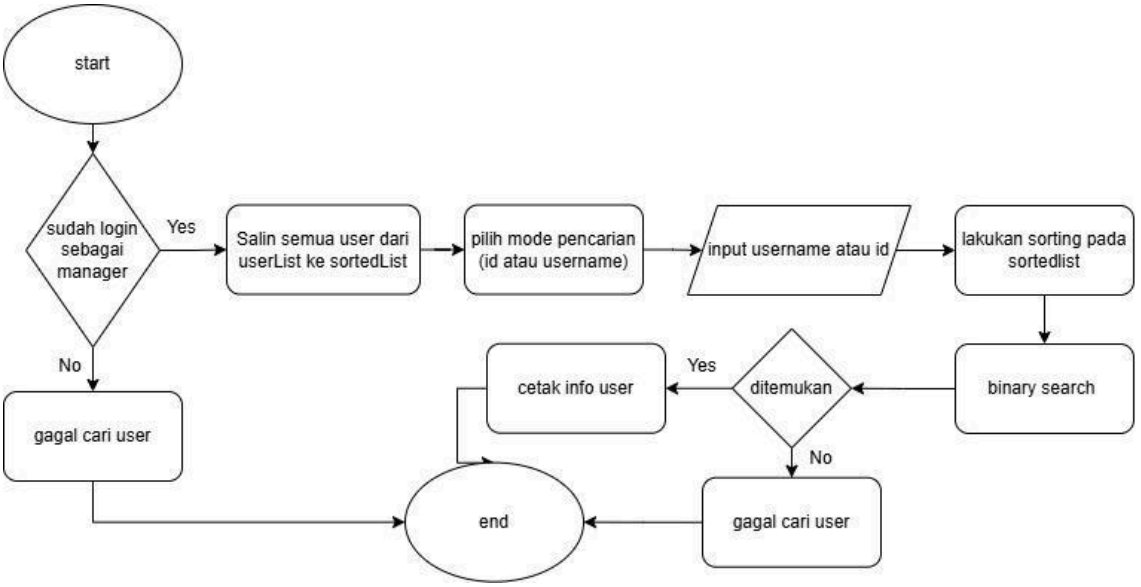
Gambar 6 Dekomposisi Algoritmik dan Fungsional - F06

F07 – Lihat User



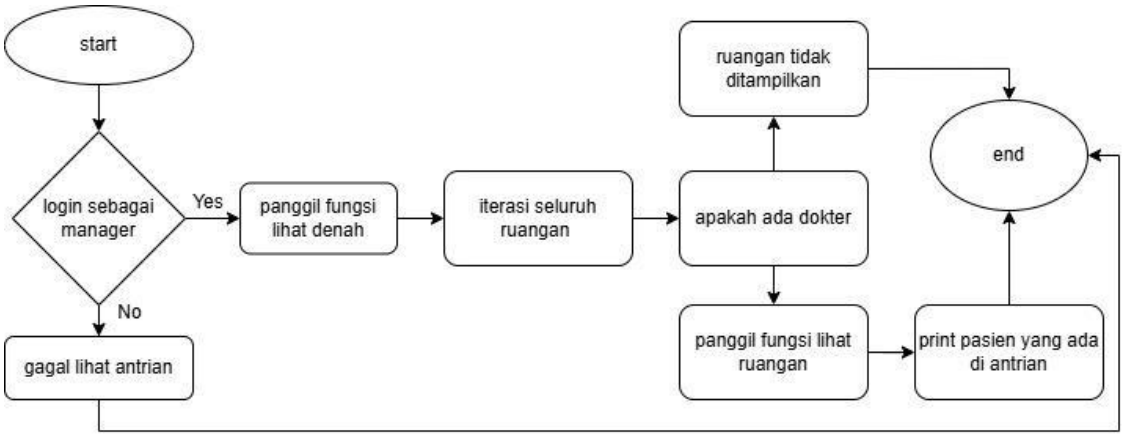
Gambar 7 Dekomposisi Algoritmik dan Fungsional - F07

F08 – Cari User



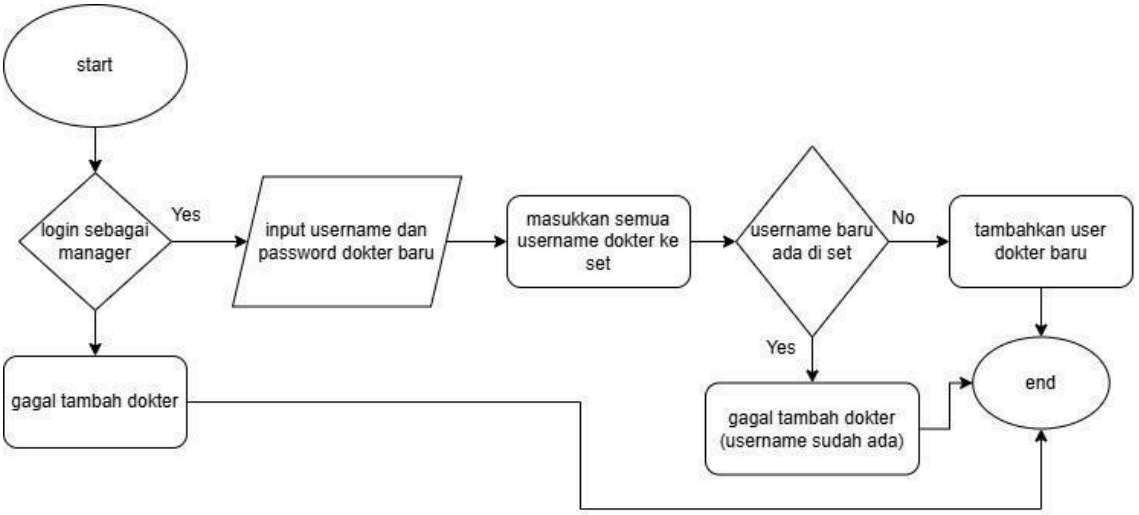
Gambar 8 Dekomposisi Algoritmik dan Fungsional - F08

F09 – Lihat Antrian



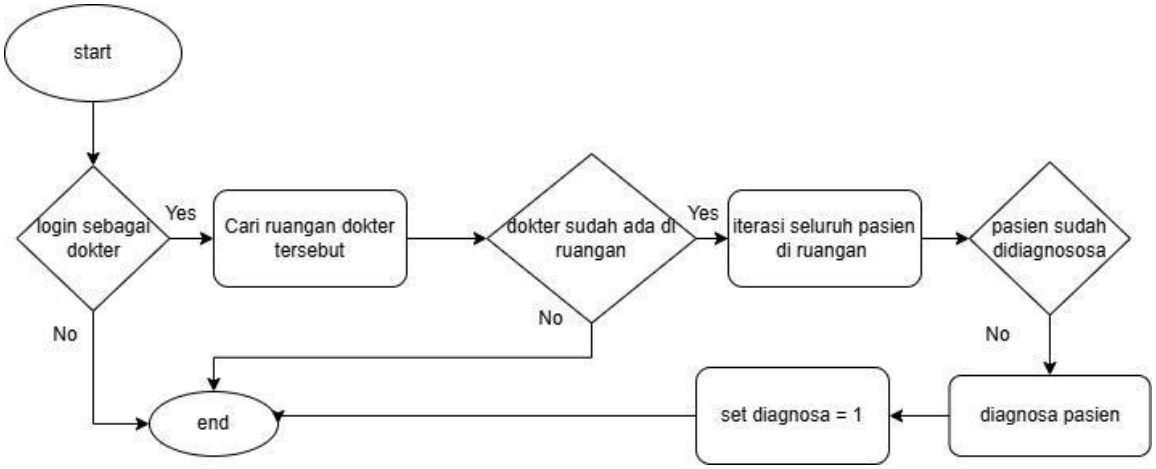
Gambar 9 Dekomposisi Algoritmik dan Fungsional - F09

F10 – Tambah & Assign Dokter



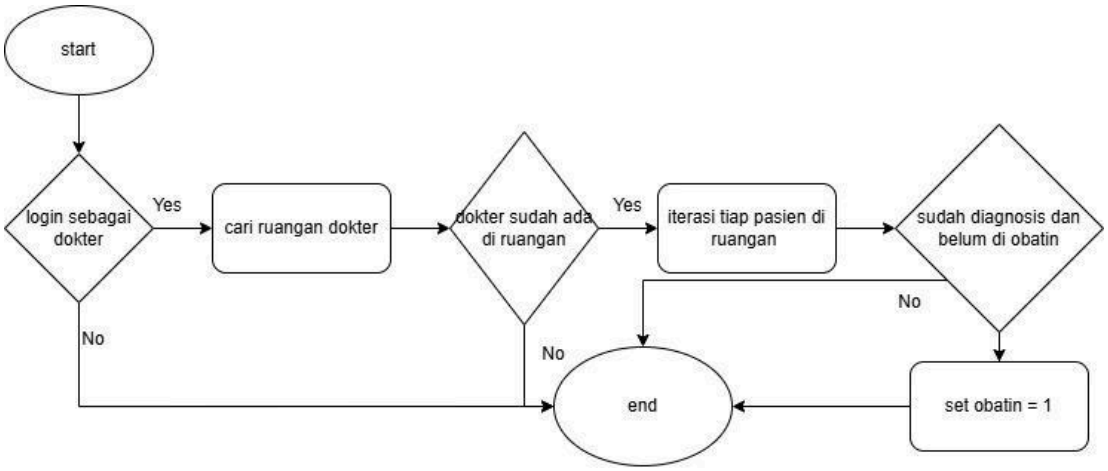
Gambar 10 Dekomposisi Algoritmik dan Fungsional - F10

F11 – Diagnosis



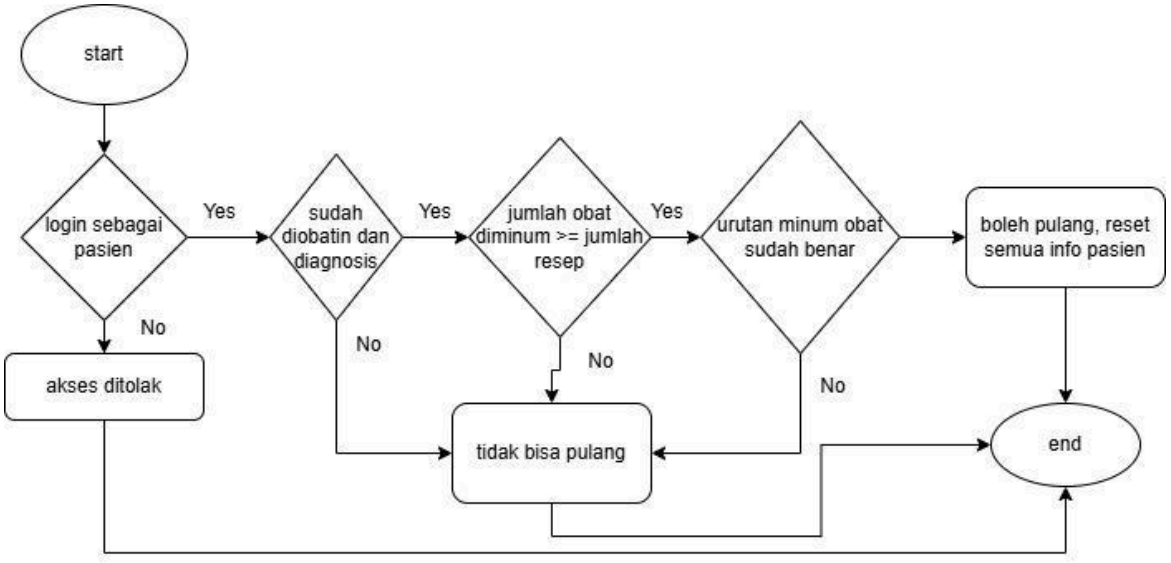
Gambar 11 Dekomposisi Algoritmik dan Fungsional - F11

F12 – Ngobatin



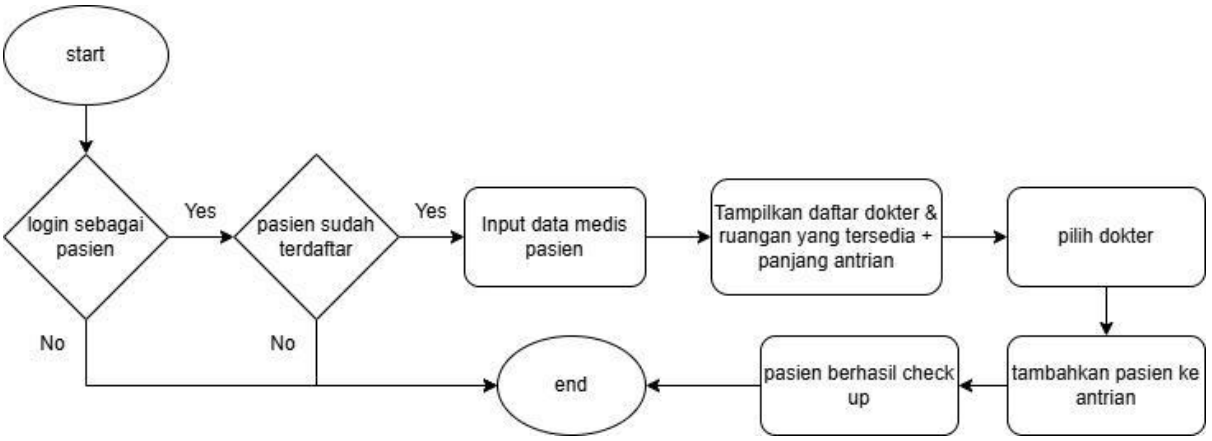
Gambar 12 Dekomposisi Algoritmik dan Fungsional - F12

F13 – Pulang Dok



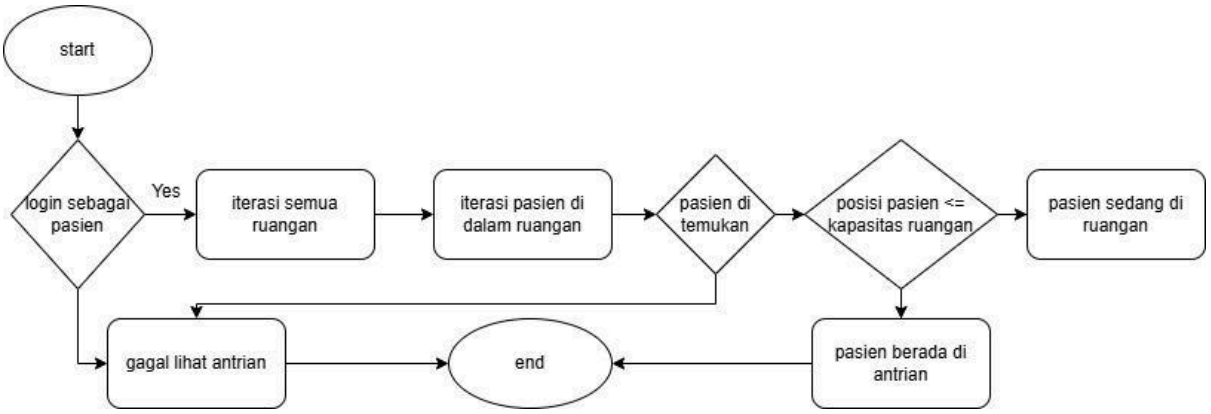
Gambar 13 Dekomposisi Algoritmik dan Fungsional - F13

F14 – Daftar Check-up



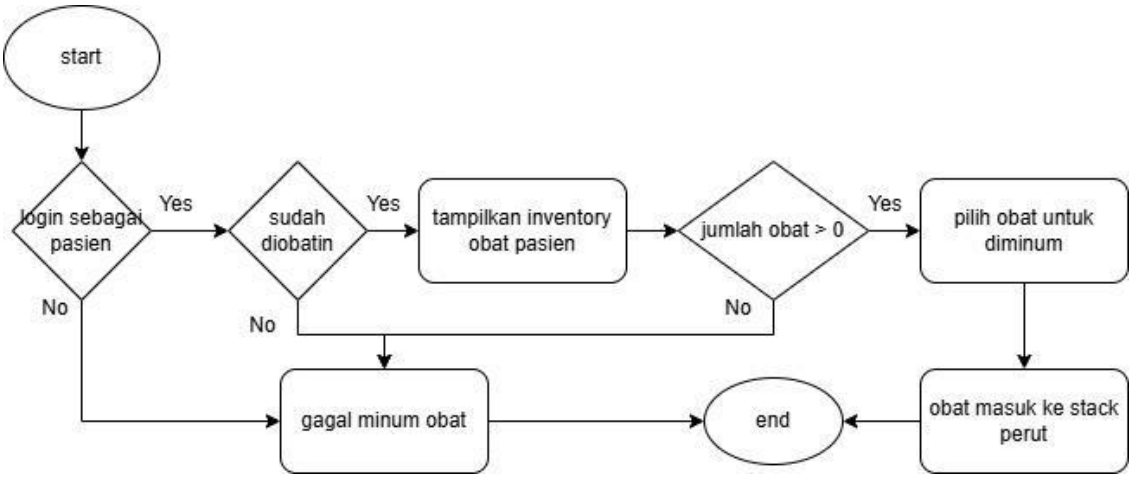
Gambar 14 Dekomposisi Algoritmik dan Fungsional - F14

F15 – Antrian Saya!



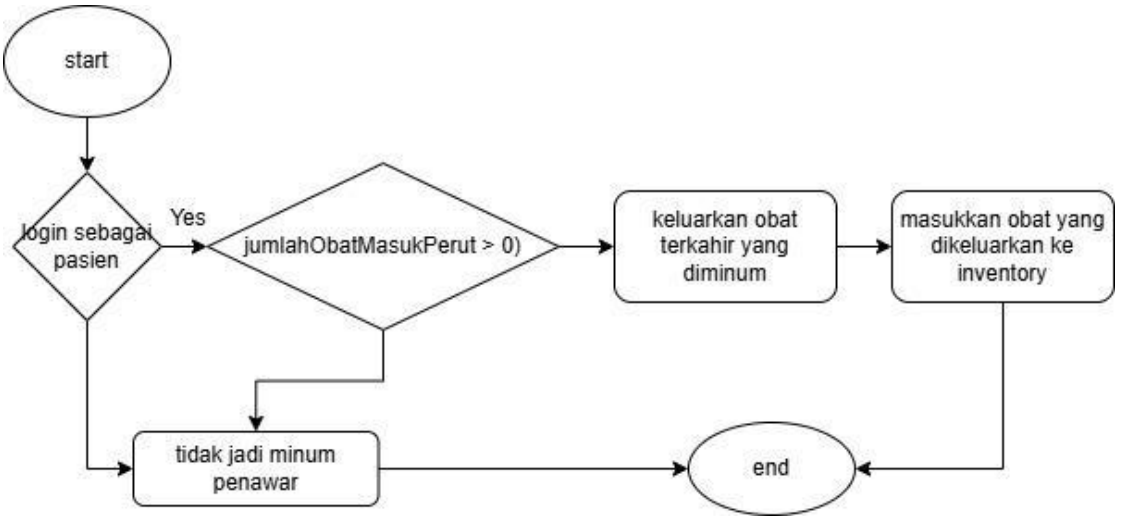
Gambar 15 Dekomposisi Algoritmik dan Fungsional - F15

F16 – Minum Obat



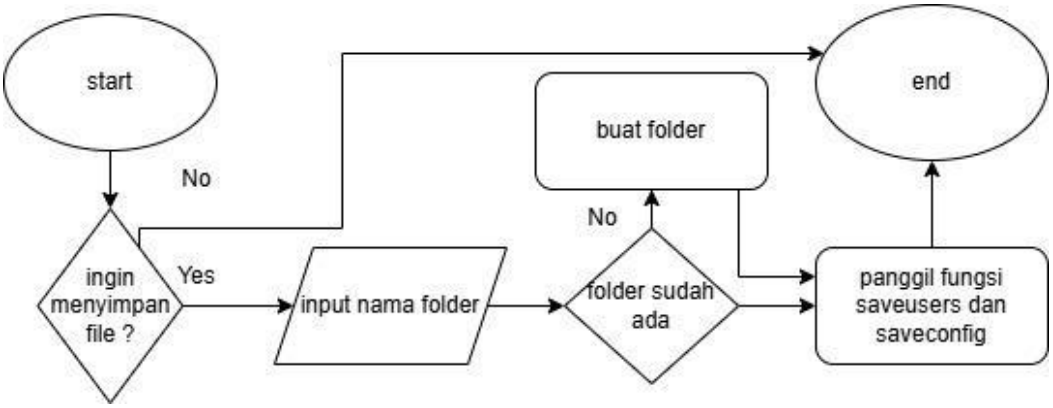
Gambar 16 Dekomposisi Algoritmik dan Fungsional - F16

F17 – Minum Penawar



Gambar 17 Dekomposisi Algoritmik dan Fungsional - F17

F18 – Exit



Gambar 18 Dekomposisi Algoritmik dan Fungsional - F18

Spesifikasi

F01 – Login

```
procedure login (input/output userList: UserList, input/output session:
Session)
{ Spesifikasi: Memasukkan user ke dalam 'session', jika user sudah di dalam
'session', berarti user sudah login.
I.S : Pengguna berada di halaman login, belum terautentikasi.
F.S : Pengguna berhasil login dan masuk ke menu utama, atau gagal login dan
kembali ke halaman utama. }

KAMUS LOKAL
      usernameInput, passwordInput: string
      i: integer

ALGORITMA
      if (session.loggedIn = true) then
            output("Anda sudah login sebagai ",
session.currentUser.username)
            return

      output("Username: ")
      input(usernameInput)
      output("Password: ")
      input(passwordInput)

      i traversal (0..userlist→count-1)
            if (usernameInput = userList.users[i].username) then
                  if (passwordInput = userList.users[i].password) then
                        session.loggedIn ← 1
                        session.currentUser ← userList.users[i]
                        depend on (session.currentUser.role)
                              (session.currentUser.role = "manager") :
                                    output("Selamat pagi Manager ")
                                    output(session.currentUser.username,
"!")
                              (session.currentUser,role = "dokter") :
                                    output("Selamat pagi Dokter ")
                                    output(session.currentUser.username, "!"
)
                              (session.currentUser.role ≠ "manager" and
                              session.currentUser,role ≠ "dokter") :
                                    output("Selamat pagi ")
                                    output(session.currentUser.username)
                                    output("! Ada kelihan apa?")
                        return
                  else {passwordInput ≠ userList.users[i].password}
                        output("Password salah untuk pengguna yang bernama
")
                        output(usernameInput, " !")
                        return
```

```
      output("Tidak ada Manager, Dokter, atau pun Pasien yang bernama ")
      output(usernameInput, "!")
```

F02 – Register Pasien

```
procedure RegisterUser (input/output userList: UserList, input/output
session: Session)
{ Spesifikasi: Mendaftarkan pasien baru ke dalam program.
I.S : Pengguna berada di halaman registrasi, data user belum terdaftar.
F.S : Data user baru berhasil ditambahkan ke sistem, atau registrasi gagal
karena data tidak valid/username sudah ada. }

KAMUS LOKAL
      username, password, lowered, usernamelowered: string
      i: integer
      usernameSet: Set
      newUser: User

ALGORITMA
      if (session.loggedIn = true) then
            output("Logout dulu sebelum register!")
            return

      output("Username baru: ")
      input(username)
      output("Password baru: ")
      input(password)
      usernameSet ← CreateNewSet()
      i traversal [0..userlist→count-1]
            lowered ← userList.users[i].username
            ToLowerCase(lowered)
            InsertSet(usernameSet, lowered)
      usernameLowered ← username
      ToLowerCase(usernameLowered)
      if (IsInSet(usernameSet, usernameLowered = true) then
            output("Registrasi gagal! User dengan nama ")
            output(username, "sudah terdaftar.")
            return

      CreateUser(newUser, userList.count+1, username, password, "pasien",
"-", -1, -1, -1,
                  -1, -1, -1, -1, -1, -1, -1, -1)
      AddUser(userList, newUser)
      output("Pasien ", newUser.username, " berhasil ditambahkan!")
```

F03 – Logout

```
procedure Logout (input/output session: Session)
{ Spesifikasi: Mengeluarkan user dari 'session', sekarang 'session' bertanda
tidak login
I.S : Pengguna dalam keadaan sudah login
F.S : Pengguna keluar dari sesi dan kembali ke halaman login }

KAMUS LOKAL

ALGORITMA
      if (session.loggedIn = 0) then
            output("Logout gagal! Anda belum login, silahkan login terlebih
dahulu
                  sebelum melakukan logout.")
      else { session.loggedIn ≠ 0 }
            output("Sampai jumpa, ", session.currentUser.username,"!")
            session.loggedIn ← 0
            memset(session.currentUser, 0, sizeof(session.currentUser))
```

F04 – Lupa Password

```
procedure ResetPassword (input/output userList: UserList, input/output
session: Session)
{ Spesifikasi: Fitur untuk melakukan perubahan password sebuah akun user.
I.S : Pengguna berada di halaman lupa password, ingin mereset password
F.S : Password pengguna berhasil direset, atau gagal jika data tidak valid }

KAMUS LOKAL
      username, kode, usn_encoded, newPass: string
      i, ada: integer

ALGORITMA
      if (session.loggedIn = true) then
            output("Logout dulu sebelum reset password")
            return
      output("Username: ")
      input(username)
      output("Kode Unik: ")
      input(kode)
      RunLenEncode(username, usn_enconded)
      ada ← 0
      i traversal [0..userlist.count-1]
            if (username = userList.users[i].username) then
                  ada ← 1
                  if (kode ≠ usn_encoded) then
                        output("Kode unik salah!")
                        return
                  depends on (userList.users[i].role) :
                        (userList.users[i].role = "manager") :
                                                output("Selamat pagi
Manager ")
```

```
                        (userList.users[i].role = "pasien") :
                                                output("Selamat pagi ")
                        (userList.users[i].role = "dokter"):
                                                output("Selamat pagi Dokter
")
                    output(userList.users[i].username, ", silahkan update
password Anda")
                    output("Password baru: ")
                    input(newPass)
                    userList.users[i].password ← newPass
                    output("Password berhasil direset!")
                    return
_
      if (ada = 0) then
            output("Username tidak terdaftar!")
```

F05 – Menu & Help

```
procedure Help(input session: Session)
{ Spesifikasi: Menampilkan daftar menu/fungsi yang dapat digunakan oleh
pengguna sesuai dengan rolenya.
I.S : Pengguna berada di menu utama.
F.S : Pengguna dapat melihat rincian profile dan daftar menu/fungsi yang
tersedia beserta penjelasannya. }

KAMUS LOKAL
currentRole: integer
i: integer
idx: integer

ALGORITMA
if session.currentUser.role = "manager" then
    output("Halo Manager ...")
else if session.currentUser.role = "dokter" then
    output("Halo Dokter ...")
else if session.currentUser.role = "pasien" then
    output("Selamat datang ...")

if session.currentUser.role = "manager" then
    currentRole ← 3
else if session.currentUser.role = "dokter" then
    currentRole ← 2
else if session.currentUser.role = "pasien" then
    currentRole ← 1

idx ← 1
for i in [0...16] do
    if CommandAccess[i] = currentRole or CommandAccess[i] = 5 or
(CommandAccess[i] = 4 and currentRole ≠ 0) then
        output(idx, arrayCommand[i])
        idx ← idx + 1
```

```
output("-> SAVE: Menyimpan perubahan")
output("-> EXIT: Keluar dari dunia Nimons")
output("Footnote: ...")
```

F06 – Denah Rumah Sakit

```
procedure LihatDenah (input denahHospital: Matrix, input input: character,
input/output userList: UserList)
{ Spesifikasi: Menampilkan denah rumah sakit.
I.S : Pengguna berada di menu denah rumah sakit, data denah sudah dimuat
F.S : Denah rumah sakit ditampilkan di layar }

KAMUS LOKAL
      lebar, panjang, i, j: integer

ALGORITMA
      lebar ← denahHospital.cols
      panjang ← denahHospital.rows
      output(" ")
      j traversal [0..lebar-1]
            output("     ",  j+1)
      i traversal [0..panjang-1]
            output("   +")
            j traversal [0..lebar-1]
                  output("-----+")
            output(" ", 'A'+i, " ")
            j traversal [0..lebar-1]
                  output("| ", denahHospital.data[i][j].namaRuangan)
            output("|")
            output("   +")
            j traversal [0..lebar-1]
                  output("-----+")

procedure LihatRuangan (input denahHospital: Matrix, input input: character,
input/output userList: UserList)
{ Spesifikasi: Menampilkan ruangan rumah sakit.
I.S : Pengguna berada di menu denah rumah sakit, data denah sudah dimuat
F.S : Informasi ruangan rumah sakit ditampilkan di layar }

KAMUS LOKAL
      Dokter, pasien: string
      row, col, i, j: integer
      r: Ruangan

ALGORITMA
      UbahInput(input, row, col)
      r ←GetRuangan(denahHospital, row, col)
      output("--- Detail Ruangan ", r.namaRuangan,  " ---")
      output("Kapasitas  : ", r.kapasitas)
      dokter ← "Tidak Ada"
      i traversal [0..userList.count-1]
            if (userList.users[i].id = r.dokter) then
                  if (userList.users[i].role = "dokter") then
```

```
                        dokter ← userList.users[i].username
                  else { userList.users[i].role ≠ "dokter" }
                        output("ID ", userList.users[i].id, " bukanlah id
dokter")
                  break
      if (dokter = "Tidak Ada") then
            output("Dokter     : -")
      else { dokter ≠ "Tidak Ada")
            output("Dokter     : ", dokter)

      if (r.jumlahPasien = 0 or dokter = "Tidak Ada") then
            output("  Tidak ada pasien di dalam ruangan saat ini.")
      else { jumlahPasienn lebih dari 0 atau dokter ada }
            output("Pasien di dalam ruangan : ")
            i traversal [0..r.jumlahPasien-1]
                  j traversal [0..userList.count-1]
                        if (userList.users[j].id = r.pasien[i] and
userList.users[j].role =
                              "pasien") then
                                if (userList.users[j].role = "pasien") then
                                      pasien ← userList.users[j].username
                                else { user bukan pasien }
                                      output("ID ", userList.users[j].id)
                                      output(" bukanlah id pasien")
                                break
                  output("  ", i+1, ". ", pasien)
      output("-------------------------------")
```

F07 – Lihat User

```
procedure LihatUser (input userList: UserList, input session: Session)
{ Spesifikasi: Menampilkan seluruh user yang ada di database dengan
urutan/filter tertentu.
I.S : Manager berada di menu lihat user, data user sudah ada di sistem.
F.S : Daftar user berhasil ditampilkan sesuai filter/urutan yang dipilih. }

KAMUS LOKAL
      pil1, pil2, i: integer
      sortedList: UserList
      user: User

ALGORITMA
      if (session.loggedIn = false or GetRole(session.currentUser ≠
"manager")) then
            output("Akses ditolak. Fitur ini hanya dapat diakses oleh
manager.")
            return
      else { sudah log in dan berupa manager }
            PrintPilihan(pil1, pil2)
            sortedList.count ← 0
```

```
            i traversal [0..userList.count-1]
                  user ← GetUserAt(userList, i)
                  if (GetRole(user) ≠ "manager") then
                        AppendUser (sortedList, user)
            SelectionSort(sortedList, sortedList.count, pil1, pil2)
            PrintList(sortedList, pil1, pil2)
```

F08 – Cari User

```
procedure CariUser (input userList: UserList, input session: Session)
{ Spesifikasi: Melakukan searching dan menampilkan user berdasarkan beberapa
pilihan.
I.S : Manager berada di menu cari user, data user sudah ada di sistem.
F.S : Data user yang dicari berhasil ditemukan dan ditampilkan, atau pesan
tidak ditemukan. }

KAMUS LOKAL
      pil, i, idInput, index: integer
      nama: string
      sortedList: UserList
      user: User

ALGORITMA
      if (session.loggedIn = false or GetRole(session.currentUser ≠
"manager")) then
            output("Akses ditolak. Fitur ini hanya dapat diakses oleh
manager.")
            return
      sortedList.count ← 0
      i traversal [0..userList.count-1]
            AppendUser(sortedList, GetUserAt(userList, i))
      CariPilihan(pilihan)
      depends on (pilihan)
      (pilihan = 1):
            output("> Masukkan nomor ID user: ")
            input(idInput)
            SelectionSort(sortedList, sortedList.count, 1, 1)
            if (BinarySearchUser(sortedList, idInput, index) = true) then
                  user ← GetUserAt(sortedList, index)
                  output("Menampilkan user dengan nomor ID ", idInput, ":")
                  output("ID | Nama     | Role   | Penyakit")
                  output("-----------------------------------")
                  output(GetID(user), " | ", GetUsername(user), " | ",
GetRole(user), " |
                        ", GetRiwayatPenyakit(user))
            else { BinarySearchUser(sortedList, idInput, index) ≠ true }
                  output("Tidak ditemukan user dengan ID", idInput)

      (pilihan = 2):
            output("> Masukkan nama user: ")
            input(nama)
```

```
            SelectionSort(sortedList, sortedList.count, 2, 1)
            if (SequenceSearchUser(sortedList, nama, index) = true) then
                  user ← GetUserAt(sortedList, index)
                  output("Menampilkan pengguna dengan nama ", nama, ":")
                  output("ID | Nama     | Role   | Penyakit")
                  output("-----------------------------------")
                  output(GetID(user), " | ", GetUsername(user), " | ",
GetRole(user), " |
                        ", GetRiwayatPenyakit(user))
            else { BinarySearchUser(sortedList, idInput, index) ≠ true }
                  output("Tidak ditemukan user dengan nama", nama)
      (pilihan ≠ 1 and pilihan ≠ 2):
            output("Pilihan tidak valid")
```

F09 – Lihat Antrian

```
procedure LihatAntrian(input inputCommand: string, input session: Session,
input denahRumahSakit: Denah, input userList: UserList)
{ Spesifikasi: Mencetak antrian seluruh ruangan yang terdapat dokter.
I.S : Manager berada di menu LIHAT_SEMUA_ANTRIAN, data rumah sakit ada di
sistem.
F.S : Data ruangan yang memiliki dokter dicetak ke layar. }

KAMUS LOKAL
      ruangan: string
      i, j: integer

ALGORITMA
      if (session.loggedIn = 1) then
            i ← 0
            j ← 0
                  while (inputCommand[i] ≠ NULL dan inputCommand[i] ≠ ' ')
do
                        i ← i + 1
            if inputCommand[i] = ' ' then
                        i ← i + 1
                  while (inputCommand[i] ≠ NULL dan j < panjang maksimum
ruangan - 1) do
                        ruangan[j] ← inputCommand[i]
                        i ← i + 1
                        j ← j + 1
            ruangan[j] ← NULL_TERMINATOR
            LihatRuangan(denahRumahSakit, ruangan, userList)
      else
            output("Anda harus login terlebih dahulu!")
```

F10 – Tambah & Assign Dokter

```
procedure AssignDokter (input userList: UserList, input session: Session)
{ I.S : Manager berada di menu TAMBAH_DOKTER atau ASSIGN_DOKTER
```

```
F.S : Manager menambahkan user dengan role dokter atau meng-assign dokter ke
ruangan rumah sakit. }

KAMUS LOKAL
      i, j, indexDokter, row, col, dokterSudahDiAssign,
ruanganSudahDitempati: integer
      username, ruangan, ruanganDokter, namaDokterDiRuangan: string
      dokter, user: User
      r: Ruangan

ALGORITMA
      if (session.loggedIn = false or GetRole(session.currentUser ≠
"manager")) then
            output("Akses ditolak. Fitur ini hanya dapat diakses oleh
manager.")
            return
      output("Username: ")
      input(username)

      if (SequenceSearchUser(userList, username, indexDokter = false) then
            output("User dengan nama ", username, " tidak ditemukan.")
            return
      dokter ← GetUserAt(userList, indexDokter)
      if (GetRole(dokter) ≠ "dokter") then
            output("User dengan nama", username, "bukan dokter.")
            return
      output("Ruangan: ")
      input(ruangan)
      UbahInput(ruangan, row, col)
      if (isColsValid(col, denahRumahSakit) = 0 and isRowValid(row,
denahRumahSakit)
           = 0) then
            output("Ruangan ", ruangan, " tidak ditermukan.")
            return
      r ← GetRuangan(denahRumahSakit, row, col)
      dokterSudahDiAssign ← 0
      ruanganDokter ← " "
      i traversal [0..denahRumahSakit.rows-1]
            j traversal[0..denahRumahSakit.cols-1]
                  if (denahRumahSakit.data[i][j].dokter = GetID(dokter))
then
                        dokterSudahDiAssign = 1
                        ruanganDokter ←
denahRumahSakit.data[i][j].namaRuangan
                        break
            if (dokterSudahDiAssign = 1) then
                  break
      ruanganSudahDitempati ← (r.dokter ≠ -1)
      namaDokterDiRuangan ← " "
      if (ruanganSudahDitempati = true) then
            i traversal [0..userList.count-1]
                  user ← GetUserAt(userList, i)
                  if (GetID(user = r.dokter) then
```

```
                        namaDokterDiRuangan ← GetUsername(user)
                        break
      depends on (ruanganSudahDitempati and dokterSudahDiAssign)
            (ruanganSudahDitempati = false and dokterSudahDiAssign = false):
                  r.dokter ← GetID(dokter)
                  output("Dokter ", GetUsername(dokter), " berhasil diassign
ke ruangan
                        ", r.namaRuangan, "!")
            (ruanganSudahDitempati = false and dokterSudahDiAssign = true):
                  output("Dokter ", GetUsername(dokter), " sudah diassign ke
ruangan
                        ", ruanganDokter, "!")
            (ruanganSudahDitempati = true and dokterSudahDiAssign = false):
                  output("Dokter ", GetUsername(dokter), " sudah menempati
ruangan
                        ", r.namaRuangan, "!")
                  output("Silahkan cari ruangan lain untuk dokter ",
                         GetUsername(dokter), ".")
            (ruanganSudahDitempati = true and dokterSudahDiAssign = true):
                  output("Dokter ", GetUsername(dokter), " sudah menempati
ruangan
                        ", ruanganDokter, "!")
                  output("Ruangan ", r.namaRuangan, " juga sudah ditempati
dokter ",
                         GetUsername(dokter), "!")
```

F11 – Diagnosis

```
procedure Diagnosis(input userList: UserList, input penyakitList:
PenyakitList, input session: Session)
{ Spesifikasi: Dokter melakukan diagnosis terhadap pasien pada ruangan yang
dipegangnya.
I.S : Dokter berada di menu DIAGNOSIS, data pasien pada ruangan dokter
terdefinisi.
F.S : Dokter mendiagnosis pasien terkena penyakit atau tidak dan mencetak
hasil ke layar. }

KAMUS LOKAL
      indeksRuangan: array[2] of integer
      currentRuangan: Ruangan
      current: Node
      i: integer
      found: boolean
      namaPenyakit: string

ALGORITMA
      SearchRuangan(session.currentUser.id, denahRumahSakit, indeksRuangan)
      if (indeksRuangan[0] = -1 and indeksRuangan[1] = -1) then
            output("Anda tidak ter-assign pada ruangan manapun.")
      else
            currentRuangan ← GetRuangan(denahRumahSakit, indeksRuangan[0],
                             indeksRuangan[1])
```

```
            current ← currentRuangan.antrianPasien.head
            found ← false
            while (current ≠ NULL and found = false) do
                        i traversal [0...userList.count-1]
                        if (userList.users[i].id = current.data and
userList.users[i].role =
                        "pasien") then
                                    if (userList.users[i].diagnosa = 0) then

Diagnosis(userList.users[i], penyakitList,
                                                namaPenyakit)

                                if (namaPenyakit = "") then
                                             output(user.username,
"tidak terdiagnosa
                                                penyakit apapun")
                                                else
                                                output(user.username, "terdiagnosa
penyakit",
                                                namaPenyakit)
                                    userList.users[i].diagnosa ← 1
                                          found ← true
                        else
                                          output("Pasien", user.username, "sudah
didiagnosis")
                                          found ← true
            if (found = false) then
                  output("Tidak ada pasien dalam antrian yang belum
didiagnosis.")
```

F12 – Ngobatin

```
procedure Ngobatin(input/output userList: UserList, input penyakitList:
PenyakitList, input session: Session, input obatList: ObatList, input
obatMap: ObatMap)
{ Spesifikasi: Dokter memberikan pengobatan kepada pasien yang telah
didiagnosis.
I.S : Pasien telah didiagnosis oleh dokter.
F.S : Pasien menerima pengobatan berupa resep dan urutan konsumsi obat. }

KAMUS LOKAL
      indeksRuangan: array[2] of integer
      currentRuangan: Ruangan
      current: Node
      found: boolean
      count, penyakitId: integer
      namaPenyakit: string

ALGORITMA
      if (session.loggedIn = false or GetRole(session.currentUser) ≠
"pasien") then
                  output("Error: Akses ditolak. Login sebagai pasien
terlebih dahulu!")
```

```
                    return

      pasienId ← GetID(session.currentUser)
      found ← false

      i traversal [0..denahRumahSakit.rows - 1]
                  j  traversal [0..denahRumahSakit.cols - 1]
                  ruangan ← denahRumahSakit.data[i][j]
                  curr ← ruangan.antrianPasien.head
                  posisi ← 1

                  while (curr ≠ null) do
                              if (curr.data = pasienId) then
                                          dokter ← GetUserById(userList,
ruangan.dokter)

                                          if (posisi -
ruangan.kapasitasRuangan ≤ 0 or
                                                ruangan.antrianPasien.length -
                                    ruangan.kapasitasRuangan ≤ 0) then
                                          output("Anda sedang berada di
dalam ruangan
                                    dokter!")
                                    else
                                    output("Status Antrian Anda:")
                                    output("Dokter  : Dr. ",
dokter.username)
                                    output("Ruangan : ",
ruangan.namaRuangan)
                                    output("Posisi  : ", posisi -
ruangan.kapasitasRuangan,
                                                      " dari ",
ruangan.antrianPasien.length -
                                    ruangan.kapasitasRuangan)

                                    found ← true
                                    return
                  curr ← curr.next
                        posisi ← posisi + 1

      if (found = false) then
                  output("Anda belum terdaftar dalam antrian check-up!")
                  output("Silakan daftar terlebih dahulu.")
```

F13 – Aku Boleh Pulang Ga, Dok?

```
procedure PulangDok(input/output session: Session, input/output
denahRumahSakit: Matrix)
{ Spesifikasi: Pasien dapat pulang jika telah sembuh dan konsumsi obat sesuai
urutan.
```

```
I.S : Pasien telah menerima diagnosis dan pengobatan dari dokter.
F.S : Pasien dinyatakan sembuh dan dikeluarkan dari antrian, atau diberi
instruksi lanjutan jika belum. }

KAMUS LOKAL
      i, j, idx: integer
      benar: boolean
      curr: Node
      idKeluar: integer

ALGORITMA
      if (not session.loggedIn or GetRole(session.currentUser) ≠ "pasien")
then
            output("Akses ditolak. Fitur ini hanya dapat diakses oleh
pasien.")
            return

      if (session.currentUser.ngobatin = 0 or session.currentUser.diagnosa =
0) then
            output("Kamu belum menerima diagnosis atau pengobatan dari
dokter, jangan
                  buru-buru pulang!")
            return

      if (session.currentUser.jumlahObatMasukPerut ≥
session.currentUser.jumlahNgobat)
      then
            benar ← true
            for idx ← 0 to session.currentUser.jumlahNgobat - 1 do
                        if (session.currentUser.perut.obat[idx + 1].id ≠
                        session.currentUser.urutanNgobat[idx]) then
                        benar ← false
                              break

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

            i traversal [0..denahRumahSakit.rows - 1]
            j traversal [0..denahRumahSakit.cols - 1]
                        curr ← denahRumahSakit.data[i][j].antrianPasien.head
                        while (curr ≠ null) do
                                    if (curr.data = session.currentUser.id)
then
                                    idKeluar ←

Dequeue(denahRumahSakit.data[i][j].antrianPasien)
                                    if

(denahRumahSakit.data[i][j].jumlahPasienDalamRuangan
                              > 0) then
```

```
denahRumahSakit.data[i][j].jumlahPasienDalamRuangan ←
denahRumahSakit.data[i][j].jumlahPasienDalamRuangan - 1
                                if
(denahRumahSakit.data[i][j].jumlahPasienDiAntrian > 0)
                            then

denahRumahSakit.data[i][j].jumlahPasienDiAntrian ←

denahRumahSakit.data[i][j].jumlahPasienDiAntrian - 1

                                output("Pasien dengan ID ", idKeluar, " telah
keluar dari
                                antrian.")
                                session.currentUser.ngobatin ← 0
                                session.currentUser.diagnosa ← 0
                                session.currentUser.suhuTubuh ← -1
                                session.currentUser.tekananDarahSistolik ← -1

session.currentUser.tekananDarahDiastolik ← -1
                                session.currentUser.detakJantung ← -1
                                      session.currentUser.saturasiOksigen ← -1
                                session.currentUser.kadarGulaDarah ← -1
                                session.currentUser.beratBadan ← -1
                                      session.currentUser.tinggiBadan ← -1
                                session.currentUser.kadarKolesterol ← -1
                                session.currentUser.trombosit ← -1
                                break
                                curr ← curr.next
            else
                            output("Maaf, tapi kamu masih belum bisa pulang!")
                            output("Urutan peminuman obat yang diharapkan:")
                            idx traversal [0..session.currentUser.jumlahNgobat -
1]
                            output(session.currentUser.urutanNgobat[idx], if idx
<
                            session.currentUser.jumlahNgobat - 1 then " -> "
else "")

                            output("Urutan peminum obat Anda:")
                            idx traversal
[0..session.currentUser.jumlahObatMasukPerut - 1]
                            output(session.currentUser.perut.obat[idx + 1].id,
if idx <
                            session.currentUser.jumlahObatMasukPerut - 1 then "
-> " else
                            "")

                            output("Silahkan kunjungi dokter untuk meminta
penawar yang
                    sesuai!")
           else
```

```
                output("Jumlah obat yang kamu minum masih kurang dari yang
                diresepkan oleh dokter!")
```

F14 – Daftar Checkup

```
procedure DaftarCheckup(input/output userList: UserList, input/output
session: Session, input/output denahRumahSakit: Matrix)
{ Spesifikasi: Pasien mendaftar checkup, mengisi data medis, dan memilih
dokter.
I.S : Pasien belum terdaftar untuk checkup.
F.S : Pasien terdaftar dalam antrian checkup. }

KAMUS LOKAL
      i, j, row, col: integer
      suhu, saturasi, berart: float
      sistolik, diastolik, detak, gula, tinggi, kolesterol, trombosit:
integer
      curr: Node
      ruangan: Ruangan
      sudahTerdaftar: boolean
      dokterCount, pilihanDokter, selectedDokterId: integer
      dokterList: array of integer
      namaRuangan: string

ALGORITMA
      if (not session.loggedIn or GetRole(session.currentUser) ≠ "pasien")
then
            output("Akses ditolak. Login sebagai pasien terlebih dahulu.")
            return

      sudahTerdaftar ← false

      i traversal [0..denahRumahSakit.rows - 1]
            j traversal [0..denahRumahSakit.cols - 1]
                  ruangan ← denahRumahSakit.data[i][j]
                  curr ← ruangan.antrianPasien.head
                  while (curr ≠ null) do
                  if (curr.data = session.currentUser.id) then
                              sudahTerdaftar ← true
                              break
                  curr ← curr.next
                  if (sudahTerdaftar) then
                  break
            if (sudahTerdaftar) then
                  break

      if (sudahTerdaftar) then
            output("Anda sudah terdaftar dalam antrian check-up!")
            return
```

```
      InputDataMedis(suhu, sistolik, diastolik, detak, saturasi, gula,
berat, tinggi, kolesterol,
      trombosit)

      session.currentUser.suhuTubuh ← suhu
      session.currentUser.tekananDarahSistolik ← sistolik
      session.currentUser.tekananDarahDiastolik ← diastolik
      session.currentUser.detakJantung ← detak
      session.currentUser.saturasiOksigen ← saturasi
      session.currentUser.kadarGulaDarah ← gula
      session.currentUser.beratBadan ← berat
      session.currentUser.tinggiBadan ← tinggi
      session.currentUser.kadarKolesterol ← kolesterol
      session.currentUser.trombosit ← trombosit

      dokterCount ← 0
      output("Daftar Dokter yang Tersedia:")

      i traversal [0..userList.count - 1]
            if (userList.users[i].role = "dokter") then
                  FindDokter(denahRumahSakit, row, col, namaRuangan,
                  userList.users[i].id)
                        ruangan ← GetRuangan(denahRumahSakit, row, col)
                        if (ruangan ≠ null) then
                        antrian ← ruangan.antrianPasien.length
                  else
                        antrian ← 0

                        output(dokterCount + 1, ". Dr. ",
userList.users[i].username, " - ", namaRuangan, " (Antrian: ", antrian, ")")

                        dokterList[dokterCount] ← i
                        dokterCount ← dokterCount + 1

      output("Pilih dokter (1-", dokterCount, "): ")
      input(pilihanDokter)

      if (pilihanDokter < 1 or pilihanDokter > dokterCount) then
            output("Pilihan tidak valid!")
            return

      pilihanDokter ← pilihanDokter - 1
      selectedDokterId ← userList.users[dokterList[pilihanDokter]].id

      FindDokter(denahRumahSakit, row, col, namaRuangan, selectedDokterId)
      ruangan ← GetRuangan(denahRumahSakit, row, col)

      if (ruangan = null) then
            output("Error: Dokter tidak memiliki ruangan!")
            return

      curr ← ruangan.antrianPasien.head
```

```
      while (curr ≠ null) do
            if (curr.data = session.currentUser.id) then
                  output("Anda sudah terdaftar dalam antrian ini!")
                  return
             curr ← curr.next

      enqueue(ruangan.antrianPasien, createNode(session.currentUser.id))
      output("Pendaftaran berhasil! Anda berada di posisi antrian ke-",
ruangan.antrianPasien.length, " di ruangan ", ruangan.namaRuangan)
```

F15 – Antrian Saya

```
procedure LihatAntrianSaya(input userList: UserList, input session: Session,
input denahRumahSakit: Matrix)
{ I.S : Pasien telah terdaftar untuk checkup.
   F.S : Pasien melihat posisi dalam antrian. }
KAMUS LOKAL
      i, j, posisi, pasienId: integer
      found: boolean
      ruangan: Ruangan
      curr: Node
      dokter: User
ALGORITMA
      if (session.loggedIn = false or GetRole(session.currentUser) ≠
"pasien") then
            output("Error: Akses ditolak. Login sebagai pasien terlebih
dahulu!")
            return
      pasienId ← GetID(session.currentUser)
      found ← false
      i traversal [0..denahRumahSakit.rows - 1]
            j traversal [0..denahRumahSakit.cols - 1]
                  ruangan ← denahRumahSakit.data[i][j]
                  curr ← ruangan.antrianPasien.head
                  posisi ← 1

                  while (curr ≠ null) do
                  if (curr.data = pasienId) then
                        dokter ← GetUserById(userList, ruangan.dokter)
                        if (posisi - ruangan.kapasitasRuangan ≤ 0 or
                        ruangan.antrianPasien.length -
                        ruangan.kapasitasRuangan ≤ 0) then
                              output("Anda sedang berada di dalam
ruangan
                                    dokter!")
                        else
                              output("Status Antrian Anda:")
                              output("Dokter  : Dr. ",
dokter.username)
                              output("Ruangan : ",
ruangan.namaRuangan)
```

```
                                    output("Posisi  : ", posisi -
                                          ruangan.kapasitasRuangan," dari ",
                                          ruangan.antrianPasien.length -
                                          ruangan.kapasitasRuangan)
                              found ← true
                              return
                        curr ← curr.next
                        posisi ← posisi + 1
      if (found = false) then
            output("Anda belum terdaftar dalam antrian check-up!")
            output("Silakan daftar terlebih dahulu.")
```

F16 – Minum Obat

```
procedure Minum Obat (input userList: UserList, input session: Session,
input denahRumahSakit: Matrix)
{ I.S : Pasien memiliki obat dalam inventory.
  F.S : Pasien mengonsumsi obat dari inventory. }
KAMUS LOKAL
      i, N: integer
      obat: Obat

ALGORITMA
      if (user.jumlahObat = 0) then
            output("Anda tidak memiliki obat di inventory")
      else
            do
                  output(">>> Pilih obat untuk diminum: ")
                  input(N)
                  if (N > user.jumlahObat or N ≤ 0) then
                        output("Pilihan nomor tidak tersedia!")
            while (N > user.jumlahObat or N ≤ 0)
            output("Obat berhasil diminum!")
            obat ← GetObatById(obatList, user.obat[N - 1])
            Push(user.perut, obat)
            user.jumlahObatMasukPerut ← user.jumlahObatMasukPerut + 1
            i traversal [N - 1..user.jumlahObat - 2]
                  user.obat[i] ← user.obat[i + 1]
                  user.jumlahObat ← user.jumlahObat - 1
```

F17 – Minum Penawar

```
procedure MinumPenawar(input/output session: Session)
{ Coding bagian ini bukan merupakan function/prosedur
I.S : Pasien telah mengonsumsi obat.
F.S : Pasien mengembalikan obat ke inventory. }

KAMUS LOKAL
      muntahin: Obat
```

```
ALGORITMA
      if (session.currentUser.role = "pasien") then
            if (session.currentUser.jumlahObatMasukPerut > 0) then
                  Pop(session.currentUser.perut, muntahin)

session.currentUser.obat[session.currentUser.jumlahObat] ←
                  muntahin.id
                  session.currentUser.jumlahObat ←
session.currentUser.jumlahObat +
                  1
                  output("Obat berhasil dimuntahkan dan dikembalikan
ke inventory
                  (ew), lain kali jangan makan obat sembarangan ya dek~")
                  session.currentUser.jumlahObatMasukPerut ←
                  session.currentUser.jumlahObatMasukPerut - 1
            else
                  output("Perut kosong!! Belum ada obat yang
dimakan.")
      else
            output("Akses ditolak. Fitur ini hanya dapat diakses oleh
pasien.")
```

F18 – Exit

```
procedure Exit(input userList: UserList, input folder: string, input
denahRumahSakit: Denah)
Spesifikasi: Melakukan pemberhentian program.
I.S : Pengguna berada di aplikasi.
F.S : Keluar dari aplikasi, atau aplikasi berhenti dijalankan. }

KAMUS LOKAL
      simpan, command, inputFolder: string

ALGORITMA
      do
            output("Apakah Anda mau melakukan penyimpanan file yang sudah
diubah?
                  (y/n) ")
            input(simpan)
            ToLowerCase(simpan)
      while (simpan ≠ "y" and simpan ≠ "n")

      if (simpan = "y") then
            output("Masukkan nama folder: ")
            input(inputFolder)
            command ← "[ -d " + inputFolder + " ] || mkdir " + inputFolder
            system(command)
            SaveUsers(userList, inputFolder)
            SaveObat(folder, inputFolder)
            SavePenyakit(folder, inputFolder)
```

```
            SaveConfig(denahRumahSakit, inputFolder, userList)
```

Pengujian Program

F01 – LOGIN

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

Tabel 6 Pengujian Program - F01

F02 – REGISTER

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

Tabel 7 Pengujian Program - F02

F03 – LOGOUT

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

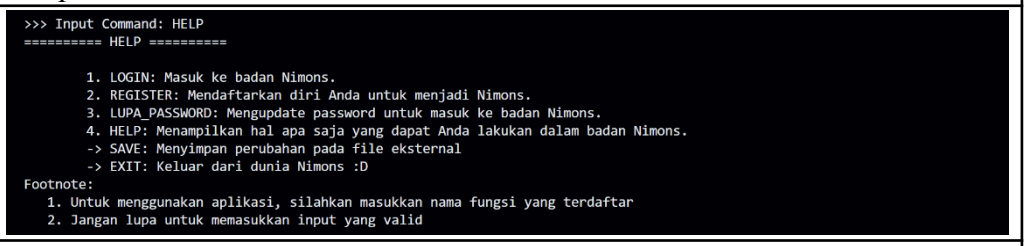
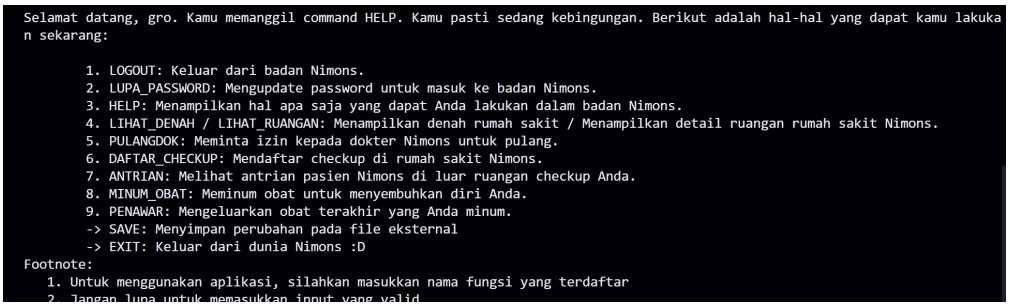
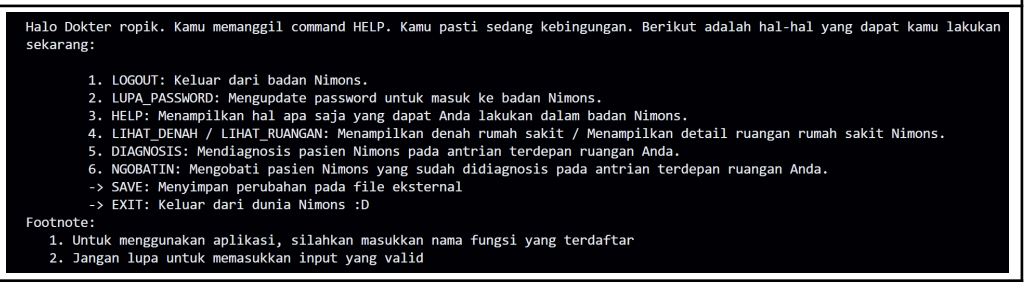
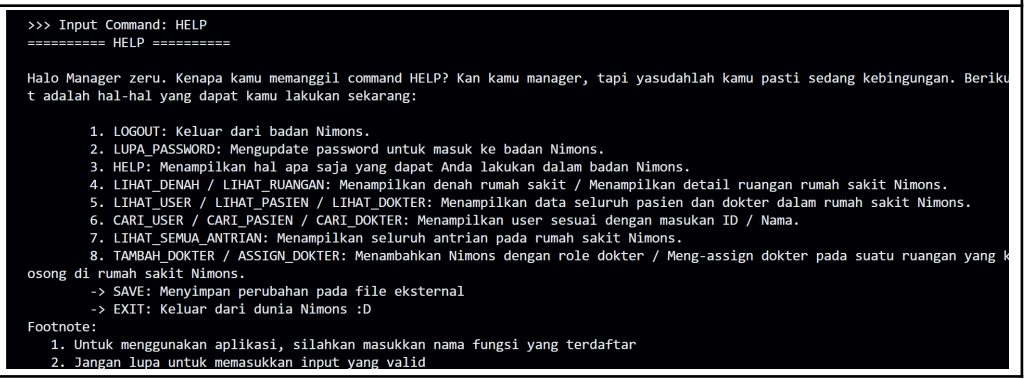
Tabel 8 Pengujian Program - F03

F04 – LUPA_PASSWORD

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

Tabel 9 Pengujian Program - F04

F05 – HELP

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 10 Pengujian Program - F05

F06 – LIHAT_DENAH / LIHAT_RUANGAN

1. Lihat Denah

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

Tabel 11 Pengujian Program - F06 (1)

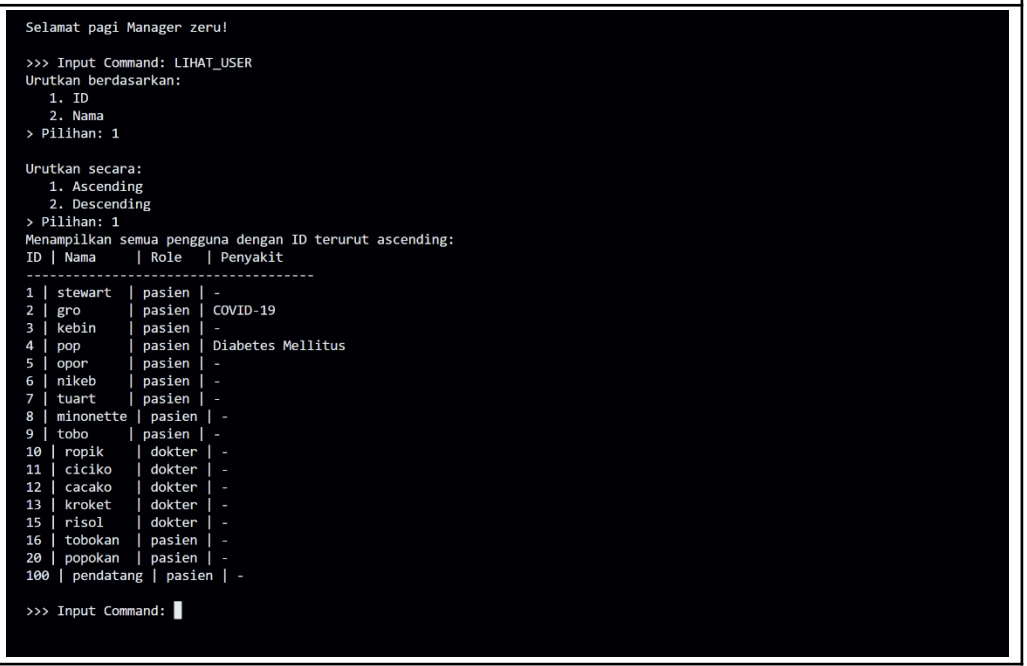
2. Lihat Ruangan

Case	Tampilan
Belum Login	
Pasien Dokter Manager	

Tabel 12 Pengujian Program - F06 (2)

F07 – LIHAT_USER / LIHAT_PASIEN / LIHAT_DOKTER

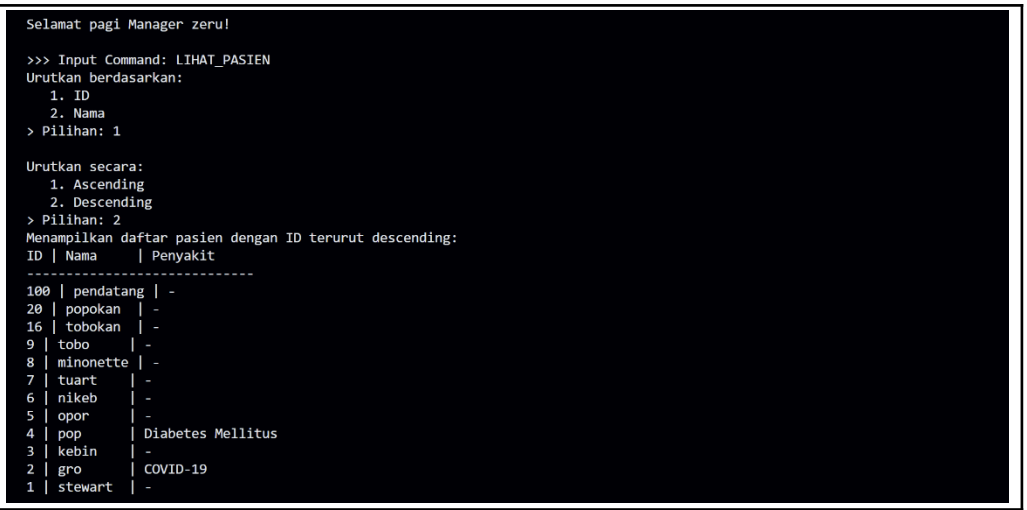
1. Lihat User

Case	Tampilan
Belum Login	
Pasien Dokter	
Manager	

Tabel 13 Pengujian Program - F07 (1)

2. Lihat Pasien

Case	Tampilan
Belum Login	
Pasien Dokter	

Manager	

Tabel 14 Pengujian Program - F07 (2)

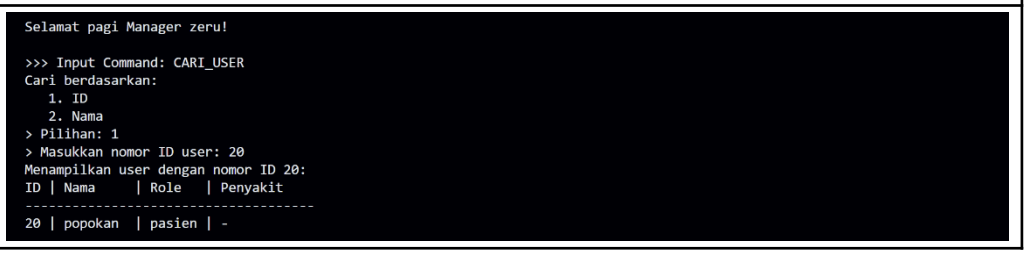
3. Lihat Dokter

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 15 Pengujian Program - F07 (3)

F08 – CARI_USER / CARI_PASIEN / CARI_DOKTER

1. Cari User

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 16 Pengujian Program - F08 (1)

2. Cari Pasien

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 17 Pengujian Program - F08 (2)

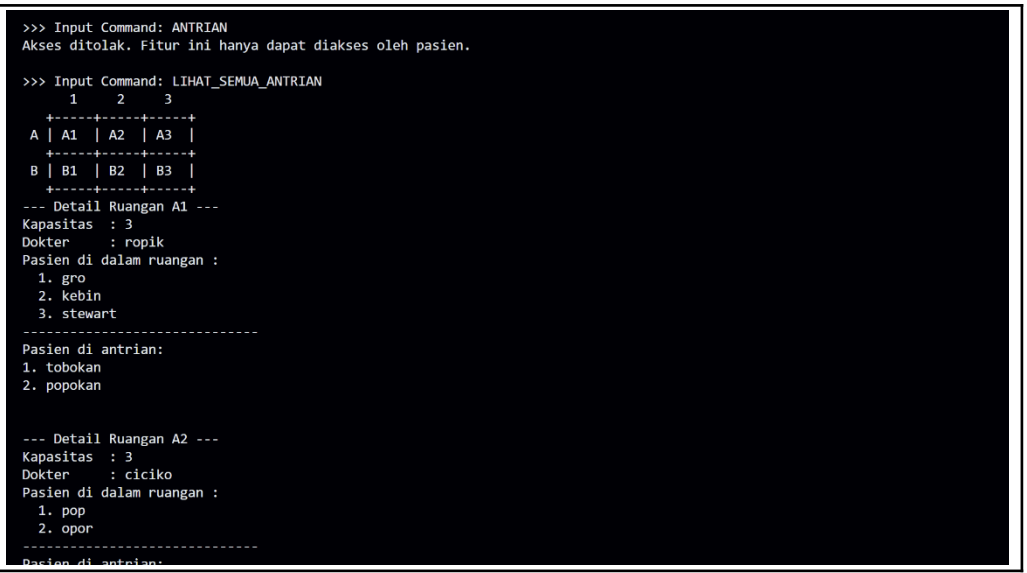
3. Cari Dokter

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 18 Pengujian Program - F08 (3)

F09 – LIHAT_ANTRIAN

Case	Tampilan
Belum Login	
Pasien	
Dokter	

Manager	

Tabel 19 Pengujian Program - F09

F10 – TAMBAH_DOKTER

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 20 Pengujian Program - F10

F11 – DIAGNOSIS

Case	Tampilan
Belum Login	
Pasien	
Manager	
Dokter	

Tabel 21 Pengujian Program - F11

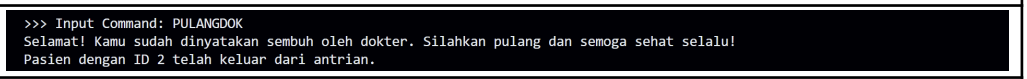
F12 – NGOBATIN

Case	Tampilan
Belum Login	
Pasien	
Manager	

Dokter	

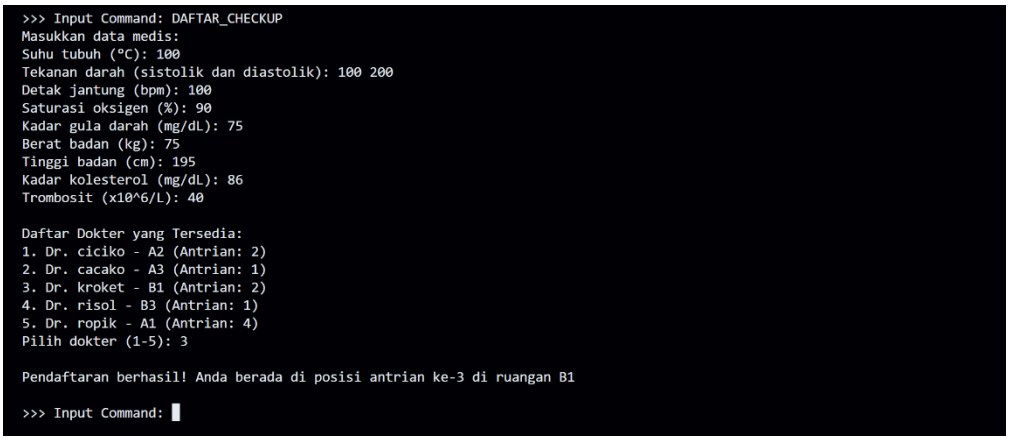
Tabel 22 Pengujian Program - F12

F13 – PULANGDOK

Case	Tampilan
Belum Login	
Dokter	
Manager	
Pasien	

Tabel 23 Pengujian Program - F13

F14 – DAFTAR_CHECKUP

Case	Tampilan
Belum Login	
Dokter	
Manager	
Pasien	

Tabel 24 Pengujian Program - F14

F15 – ANTRIAN

Case	Tampilan
Belum Login	
Dokter	
Manager	
Pasien	

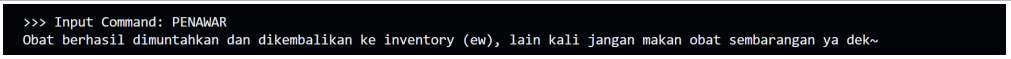
Tabel 25 Pengujian Program - F15

F16 – MINUM_OBAT

Case	Tampilan
Belum Login	
Dokter	
Manager	
Pasien	

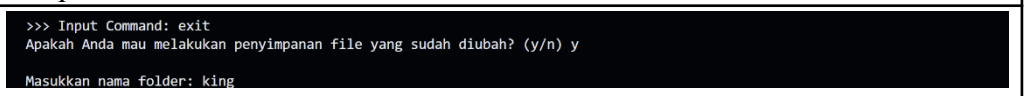
Tabel 26 Pengujian Program - F16

F17 – MINUM_PENAWAR

Case	Tampilan
Belum Login	
Dokter	
Manager	
Pasien	

Tabel 27 Pengujian Program - F17

F18 – EXIT

Case	Tampilan
Belum Login	
Pasien	
Dokter	
Manager	

Tabel 28 Pengujian Program - F18

Lampiran

- IF1210_FormAsistensiTB_1_K02-H
- IF1210_FormAsistensiTB_2_K02-H