LAPORAN PRAKTIKUM

POSTTEST (5)

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

M.Fahrianor (2409106089)

Kelas (B2’24)

PROGRAM STUDI INFORMATIKA

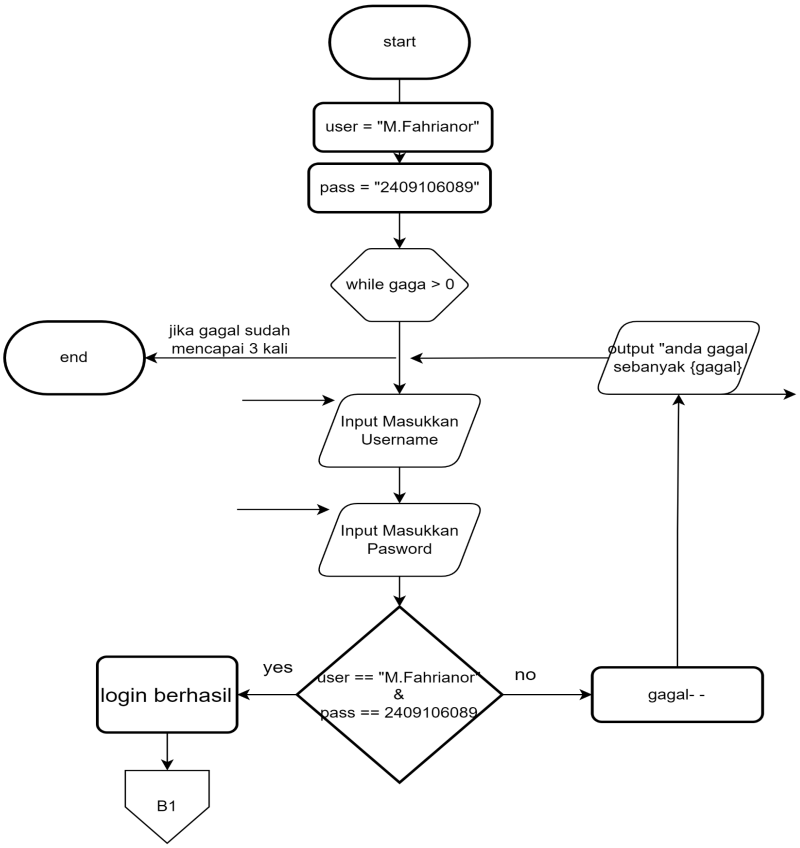
UNIVERSITAS MULAWARMAN

SAMARINDA

2025

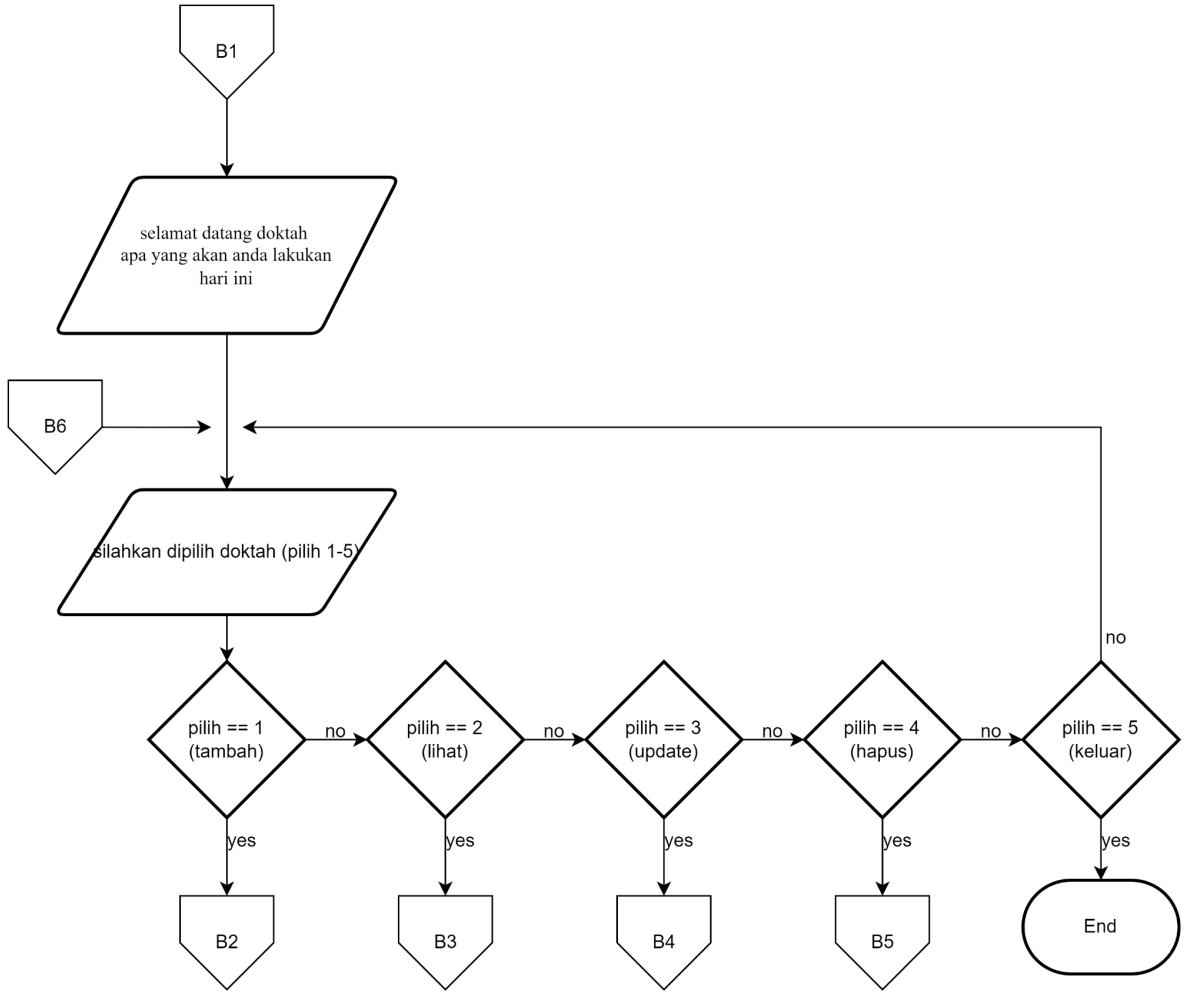
1. Flowchart

A. Login



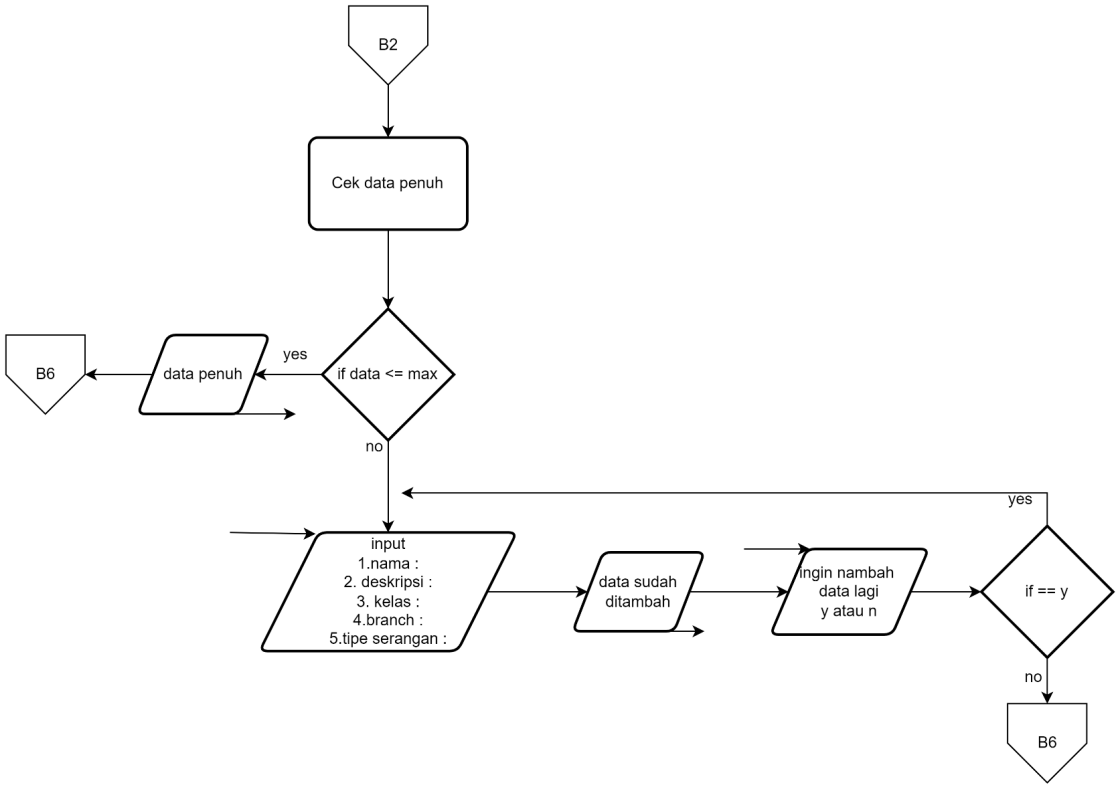
Gambar 1.1 Login

B. Menu Utama



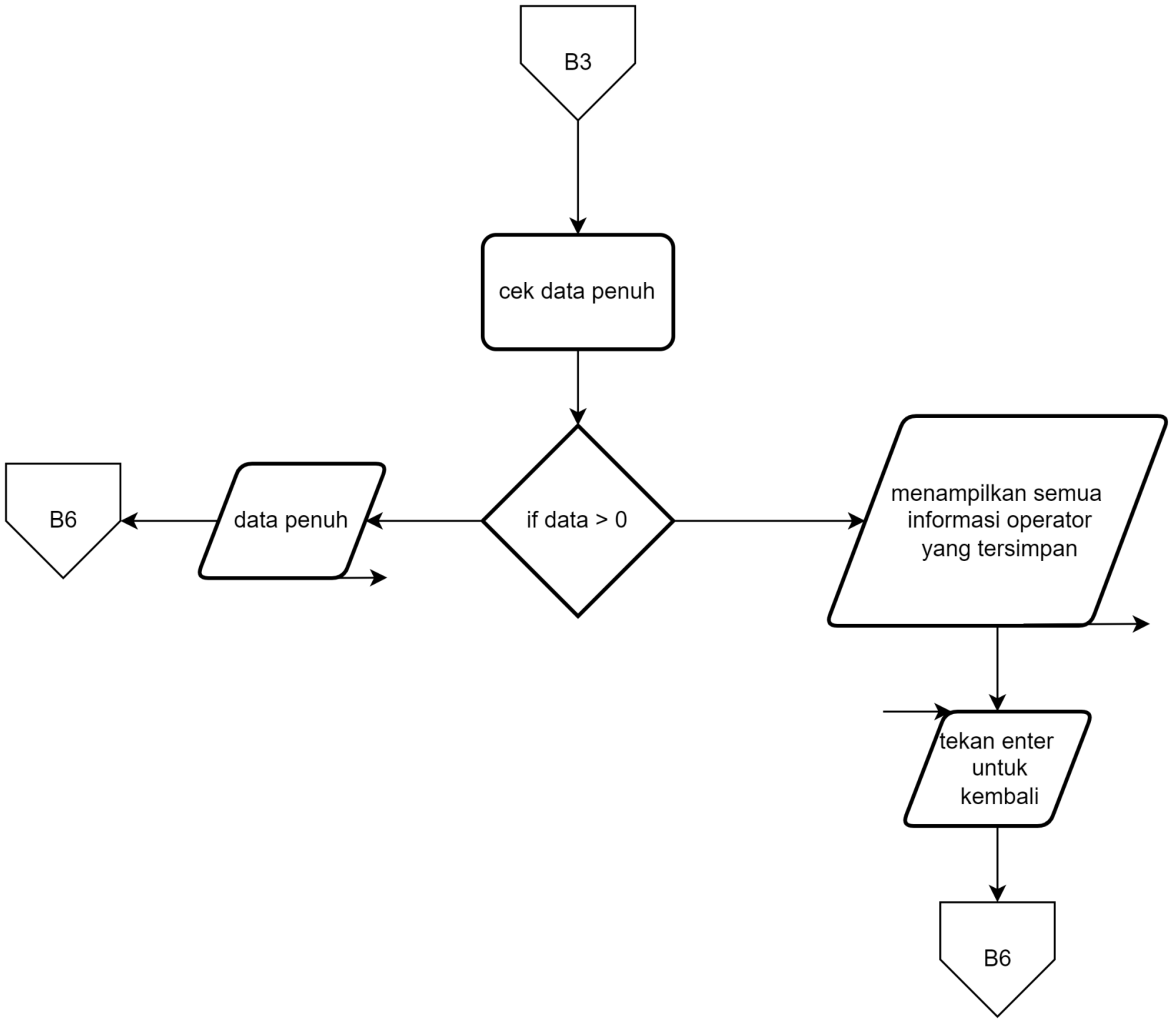
Gambar 1.2 Menu utama

C. Tambah data



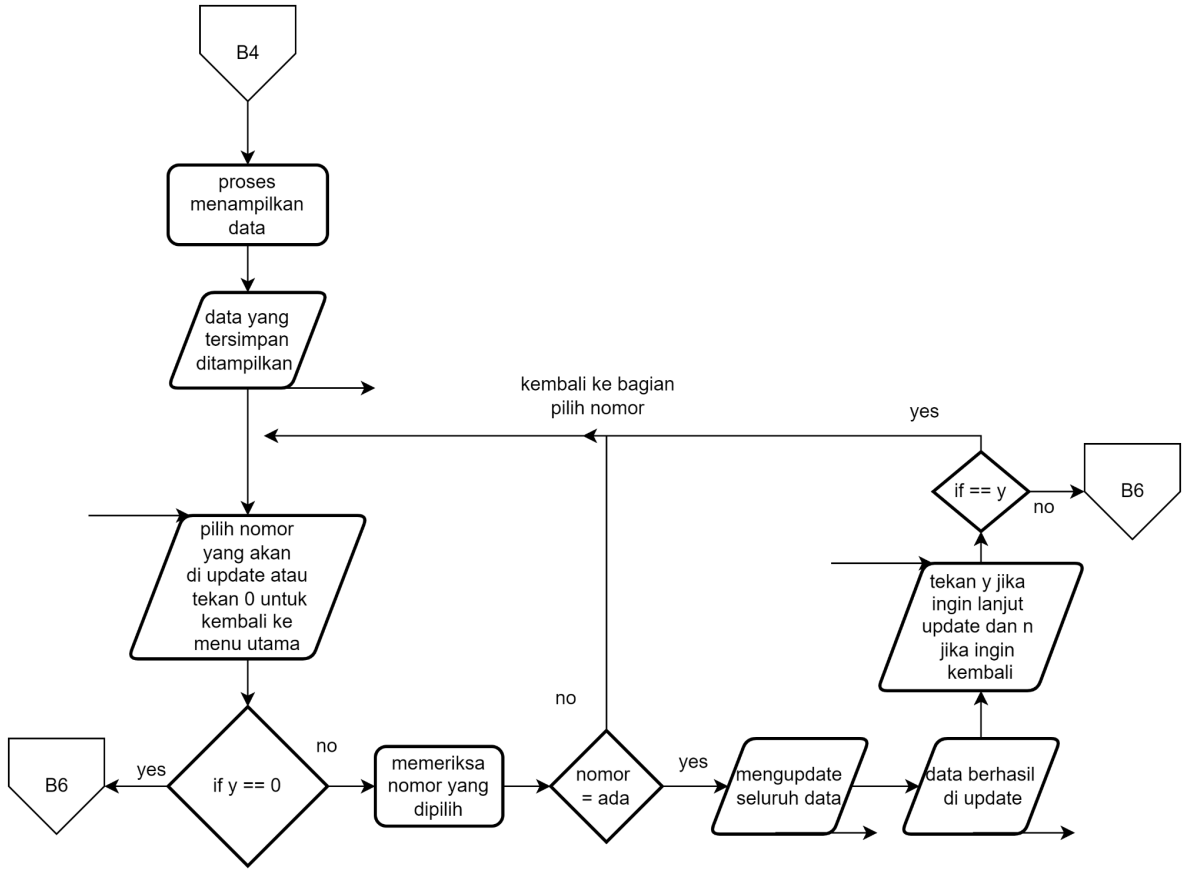
Gambar 1.3 Tambah data

D. Lihat data



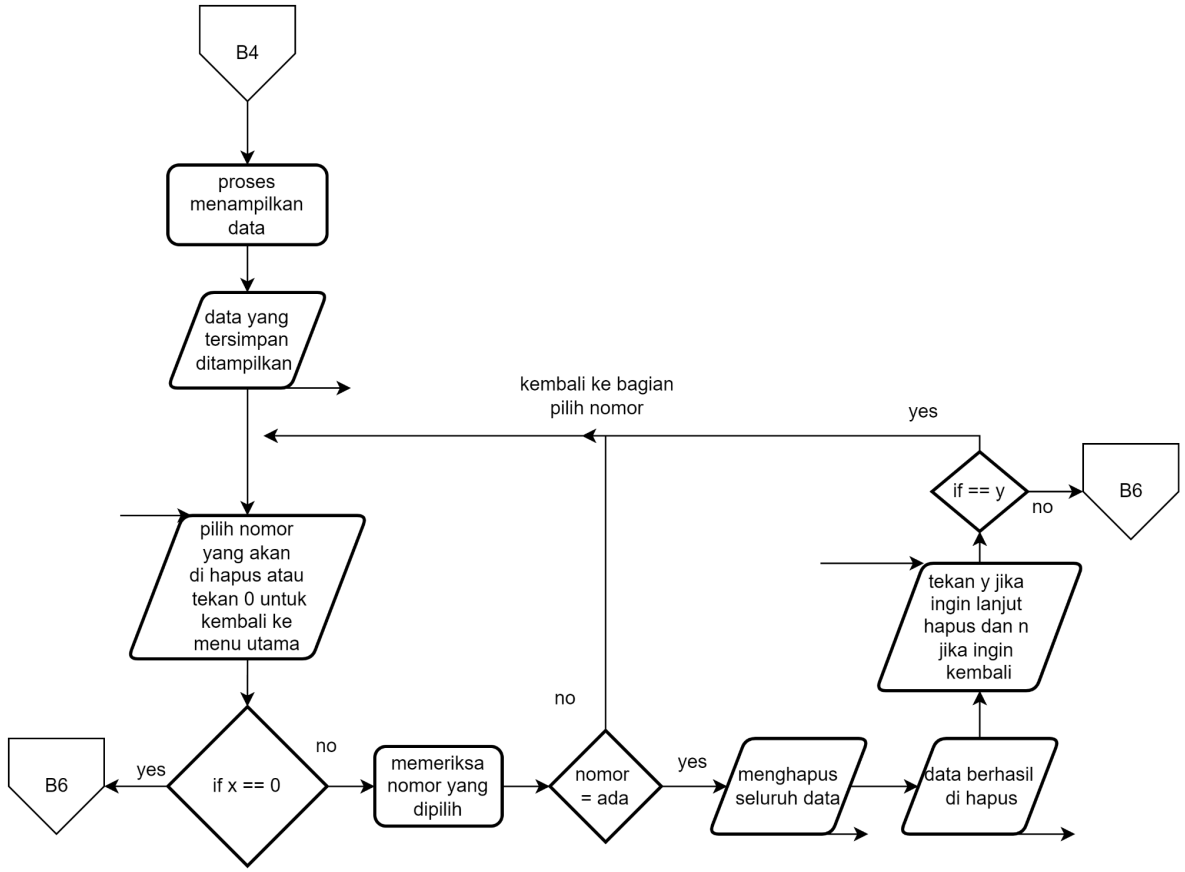
Gambar 1.4 Lihat data

E. Update data



Gambar 1.5 Update data

F. Hapus data



Gambar 1.6 Hapus data

2. Analisis Program

2.1 Deskripsi Singkat Program

Program ini merupakan program CRUD (create, read, update, delete) yang memiliki judul "Sistem informasi karakter game Arknights", sistem ini dibuat dengan menggunakan fungsi, parameter, prosedur, dan juga menggunakan prinsip dari pointer yang dapat terlihat pada program terdapat fungsi dengan parameter address-of dan dereference yang dimana pada program ini akan terdapat 5 sub program yag dibuat menggunakan fungsi. Didalam program tepatnya sebelum masuk ke menu utama akan dilakukan login yang dimana pengguna akan diminta memasukkan username dan password,dan ketika pengguna salah sebanyak 3 kali

maka program akan otomatis berhenti, tetapi jika berhasil pengguna akan masuk ke dalam menu utama, yang di dalam menu utama ada beberapa pilihan yaitu, tambah data operator, lihat data operator, update data operator, hapus data operator, dan keluar. pada program ini pengguna tidak akan bisa keluar kecuali pengguna memilih opsi keluar.

3. Source Code

A. Struct dan Data awal

Ini adalah program array of struct

Source Code:

```
#define max 20
int pos = 3;

// struct
struct Operator
{
    string karakter;
    string deskripsi;
    string kelas;
    string branch;
    string tipe;
};
Operator op[max];
void dataawal(){
    op[0].karakter = "Eyjafjalla";
    op[0].deskripsi = "Dengan nama asli Adele Nauman, dan seorang putri dari
mendiang Katia dan Magna Nauman yaitu pasangan vulcanologist yang terkenal dari
Leithanien";
    op[0].kelas = "Caster";
    op[0].branch =  "Core caster";
    op[0].tipe = "Arts";
```

Gambar 3.1 Struct dan Data awal

B. Fitur Login

Ini adalah fitur login untuk pengguna

Source Code:

```
bool login(string userbenar, string passbenar) {
    string username;
    string password;
    int gagal = 3;

    while (gagal > 0) {
        cout << "Masukkan username anda doktah: ";
        cin >> username;
        cout << "Masukkan password anda doktah: ";
        cin >> password;
        if (username == userbenar && password == passbenar) {
            return true;
        }else {
            gagal--;
            cout << "" << endl;
            cout << "Kesempatan kamu untuk masuk hanya tersisa " << gagal << ".
Jika kamu gagal login maka kamu bukanlah doktah" << endl;
        }
        if (gagal == 0) {
            system("cls");
            cout << "==================================" << endl;
            cout << "Kamu gagal. Kamu diblokir dari PRTS" << endl;
            cout << "==================================" << endl;
            return 0;
        }
    }
    return false;
}
```

Gambar 3.2 Fitur Login

C. Menu Utama

Ini merupakan tampilan menu utama jika pengguna berhasil login

Source Code:

```
// menu utama
    do
    {
        system("cls");
        cout << "================================================" << endl;
        cout << "Selamat datang doktah apa yang akan anda lakukan hari ini" <<
endl;
        cout << "================================================" << endl;
        cout << "" << endl;
        cout << "=========================" << endl;
        cout << "1. Tambah data operator\n";
        cout << "2. Lihat data operator\n";
        cout << "3. Update data operator\n";
        cout << "4. Hapus data operator\n";
        cout << "5. Keluar" << endl;
        cout << "=========================" << endl;
        cout << "" << endl;
        cout << "Siahkan dipilih doktah: ";
        cin >> pilih;
        cin.ignore();
```

Gambar 3.3 Menu utama

D. Tambah Data Operator

Pada gambar ini merupakan tampilan tambah data informasi yang menggunakan prosedur dan parameter di dalamnya dan terdapat parameter dereference pada bagian (Operator *op, int *pos), *pos >= max, dan op[*pos], yang dimana memiliki fungsi untuk mengakses dan memanipulasi data asli secara langsung dari memori.(Operator *op, int *pos) menunjukkan bahwa fungsi menerima alamat dari array Operator dan variabel int pos. Dengan menggunakan *pos >= max, program bisa membandingkan nilai posisi data saat ini dengan batas maksimum secara langsung, tanpa membuat salinan variabel. Sedangkan op[*pos] digunakan untuk menunjuk ke elemen ke berapa dalam array

Source Code:

```
void tambahdata(Operator *op, int *pos){
    string loop;
    system("cls");
            do
            {
                if (*pos >= max)
                {
                    system("cls");
                    cout << "========================" << endl;
                    cout << "Data sudah max doktah\n";
                    cout << "========================" << endl;
                    cout << "" << endl;
                    cout << "Tekan enter untuk kembali: ";
                    cin.get();
                    break;
                }else if (*pos < max){
                    system("cls");
                    cout << "====================================" << endl;
                    cout << "Masukkan nama operator: ";
                    getline(cin, op[*pos].karakter);
                    cout << "Masukkan deskripsi operator: ";
                    getline(cin, op[*pos].deskripsi);
                    cout << "Masukkan class operator: ";
                    getline(cin, op[*pos].kelas);
                    cout << "Masukkan branch operator: ";
                    getline(cin, op[*pos].branch);
                    cout << "Masukkan tipe serangan: ";
                    getline(cin, op[*pos].tipe);
                    cout << "====================================" << endl;
                    (*pos)++;
                    cout << "" << endl;
                    cout << "Apakah anda ingin menambah operator lagi doktah?
(y/n): ";
                    cin >> loop;
                    cin.ignore();
                }
```

Gambar 3.4 Tambah Data

E. Lihat Data Operator

Pada gambar ini merupakan tampilan dan proses melihat data, sama seperti tambah data lihat data juga menggunakan prosedur dan parameter di dalamnya dan juga menggunakan parameter dereference yang dimana (Operator *op, int *pos) menunjukkan bahwa fungsi menerima alamat dari array Operator dan variabel int pos dan *pos > 0 yang digunakan untuk mengecek apakah ada data yang sudah tersimpan sebelumnya atau tidak.

Source Code:

```
void lihatope(Operator *op, int *pos){
    string loop;
    system("cls");
            if (*pos > 0) {
                cout << "Data tersimpan:\n";
                for (int a = 0; a < *pos; a++) {
                    cout << a + 1 << ". Nama     : " << op[a].karakter << endl;
                    cout << "   Deskripsi    : " << op[a].deskripsi << endl;
                    cout << "   Kelas        : " << op[a].kelas << endl;
                    cout << "   Branch       : " << op[a].branch << endl;
                    cout << "   Tipe Serangan: " << op[a].tipe << endl;
                    cout << "---------------------------------" << endl;
                }
            } else {
                cout << "Data kosong\n";
            }
            cout << "Tekan enter untuk kembali: ";
            cin.get();
}
```

Gambar 3.5 Lihat Data Operator

F. Update Data Operator

Pada gambar ini merupakan fitur update data yang dimana saat memilih fitur ini di dalam menu maka yang pertama akan dilakukan sistem adalah menampilkan data yang tersimpan dan kemudian pengguna dapat memilih data nomor berapa yang akan di-update. Fitur ini menggunakan prosedur dan parameter dengan dereference, yaitu pada bagian (Operator *op, int *pos), *pos > 0, dan op[a]. Bagian (Operator *op, int *pos) menunjukkan bahwa fungsi menerima alamat dari array Operator dan jumlah data yang tersimpan. Penggunaan *pos > 0

berfungsi untuk mengecek apakah ada data yang dapat di-update secara langsung dari memori. Sedangkan op[a] digunakan untuk menampilkan nama-nama operator yang tersimpan berdasarkan jumlah data yang telah dimasukkan.

Source Code:

```
void updateope(Operator *op, int *pos){
    string loop;
    if (*pos > 0)
            {
                int y;
                do
                {
                    system("cls");
                    cout << "================================" << endl;
                    cout << "Daftar Operator yang Tersimpan:\n";
                    cout << "================================" << endl;
                    for (int a = 0; a < *pos; a++) {
                        cout << a + 1 << ". " << op[a].karakter << endl;
                    }
                    cout << "======================================" << endl;
                    cout << "Tekan 0 jika ingin kembali ke menu utama" << endl;
                    cout << "======================================" << endl;
                    cout << "" << endl;
                    cout << "\nMasukkan nomor operator yang ingin diupdate: ";
                    cin >> y;
                    cin.ignore();
                    if (y == 0 )
                    {
                        cout << "Kita kembali ke menu utama doktah\n";
                        break;
                    } else if(y > 0 && y <= *pos)
```

Gambar 3.6 Update Data Operator

G. Hapus Data Operator

pada gambar ini merupakan fitur hapus data yang dimana saat memilih fitur ini di dalam menu maka yang pertama akan dilakukan sistem adalah menampilkan data yang tersimpan dan kemudian pengguna dapat memilih data nomor berapa yang akan dihapus hal ini sama seperti penjelasan update data yang menjadi pembeda adalah pada update program akan

mengganti data lama pada array menjadi data baru sedangkan pada hapus program akan menghapus data dari array yang tersimpan

Source Code:

```
void hapusope(Operator *op, int *pos){
    string loop;
    if (*pos > 0)
            {
                int x;

                do
                {
                    system("cls");
                    cout << "================================" << endl;
                    cout << "Daftar Operator yang Tersimpan:\n";
                    cout << "================================" << endl;
                    for (int a = 0; a < *pos; a++) {
                        cout << a + 1 << ". " << op[a].karakter << endl;
                    }

                    cout << "====================================" << endl;
                    cout << "Tekan 0 jika ingin kembali ke menu utama" << endl;
                    cout << "====================================" << endl;
                    cout << "" << endl;
                    cout << "\nMasukkan nomor operator yang ingin dihapus: ";
                    cin >> x;
                    cin.ignore();

                    if (x == 0)
                    {
                        cout << "Kita kembali ke menu utama doktah\n";
                        break;
                    }else if (x > 0 && x <= *pos)
                    {
                        for (int i = x; i < *pos; i++)
                        {
                            op[i-1].karakter = op[i].karakter;
                            op[i-1].deskripsi = op[i].deskripsi;
                            op[i-1].kelas = op[i].kelas;
                            op[i-1].branch = op[i].branch;
                            op[i-1].tipe = op[i].tipe;
                        }
                        (*pos)--;
```

Gambar 3.7 Hapus Data Operator

H. Log Out

Gambar ini adalah fitur log out

Source Code:

```
void logout(){
    system("cls");
            cout << "======================" << endl;
            cout << "Selamat tinggal doktah"<< endl;
            cout << "======================" << endl;
}
```

Gambar 3.8 Log Out

J. Penulisan pada menu utama

Gambar ini adalah tampilan dari program yang digunakan untuk memanggil seluruh prosedur yang ada di luar int main() dan dalam program ini menggunakan parameter address-of

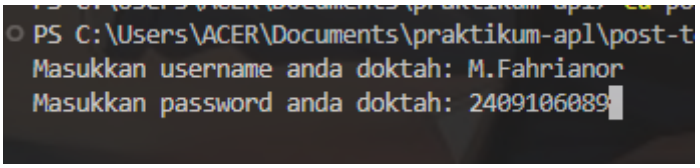
Source Code:

```
if (pilih == 1)
        {
            tambahdata(op, &pos);
        }else if (pilih == 2)
        {
            lihatope(op, &pos);
        }else if (pilih == 3)
        {
            updateope(op, &pos);
        }else if (pilih == 4)
        {
            hapusope(op, &pos);
        }else if (pilih == 5)
        {
            logout();
            break;
        }
```

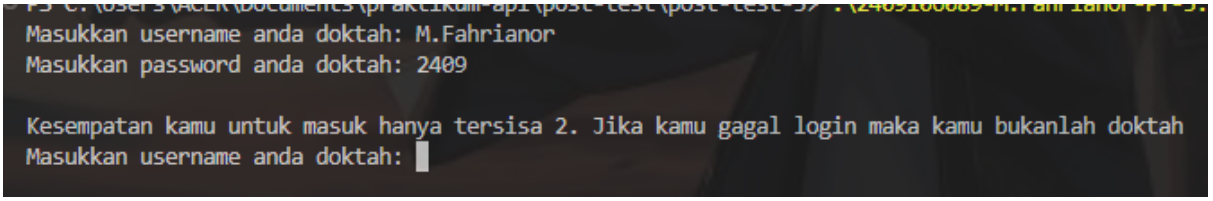
Gambar 3.9 Penulisan pada menu utama

4. Hasil Output

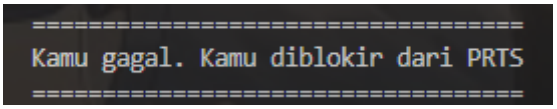
4.1 Hasil Output



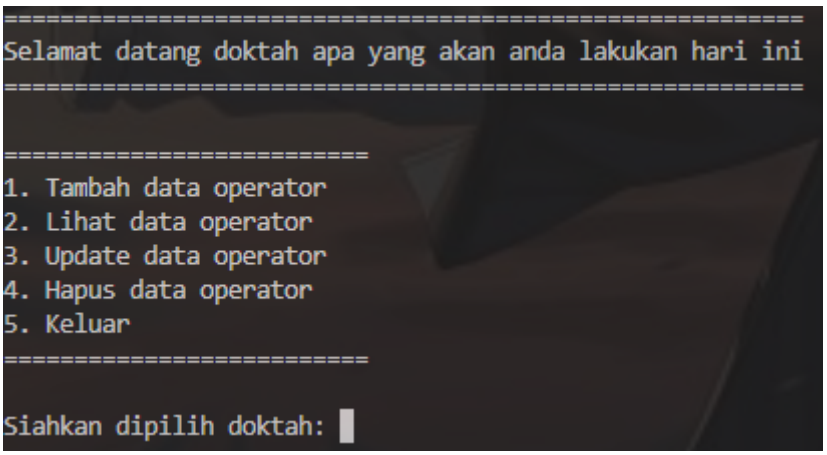
Gambar 4.1 Login



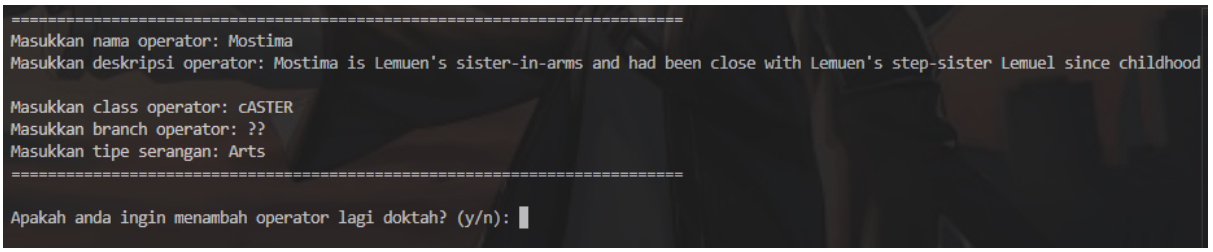
Gambar 4.2 Gagal kurang dari 3 kali



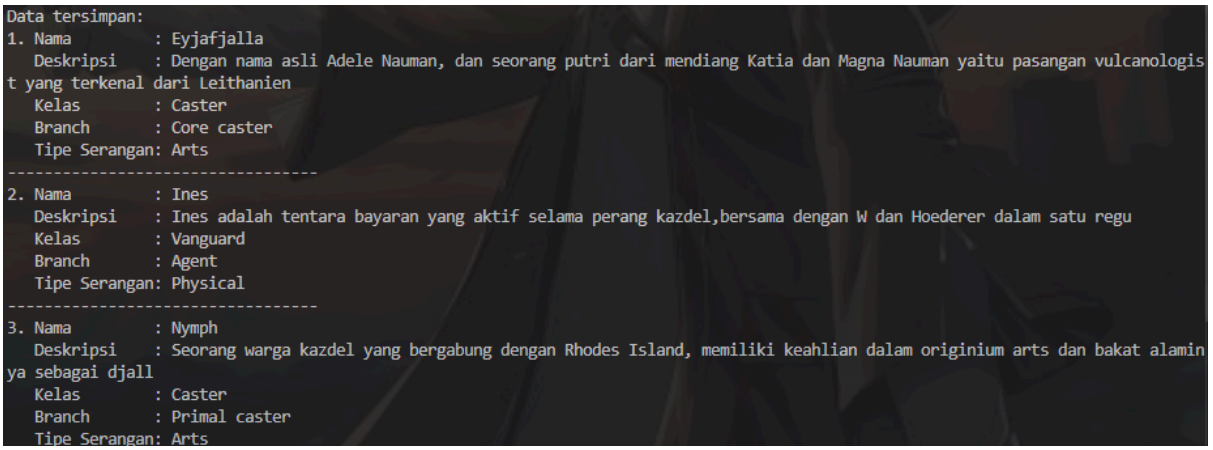
Gambar 4.3 Gagal sudah mencapai 3 kali



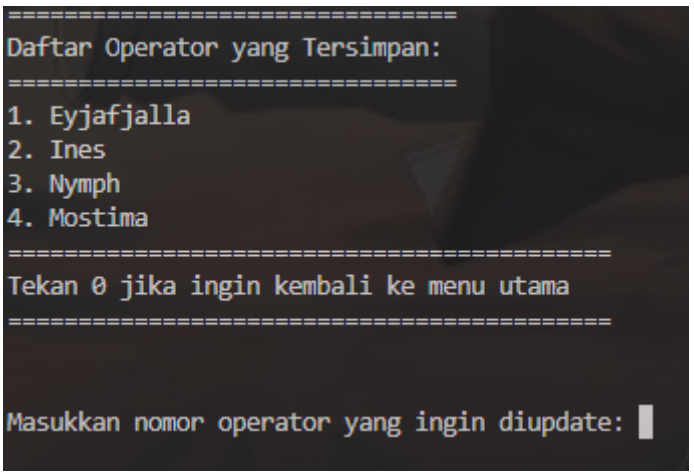
Gambar 4.4 Login berhasil dan menu utama

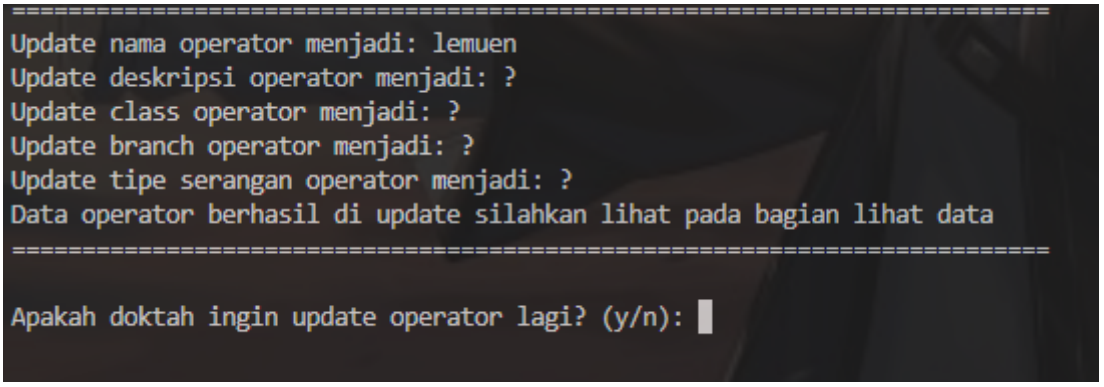


Gambar 4.5 Tambah data

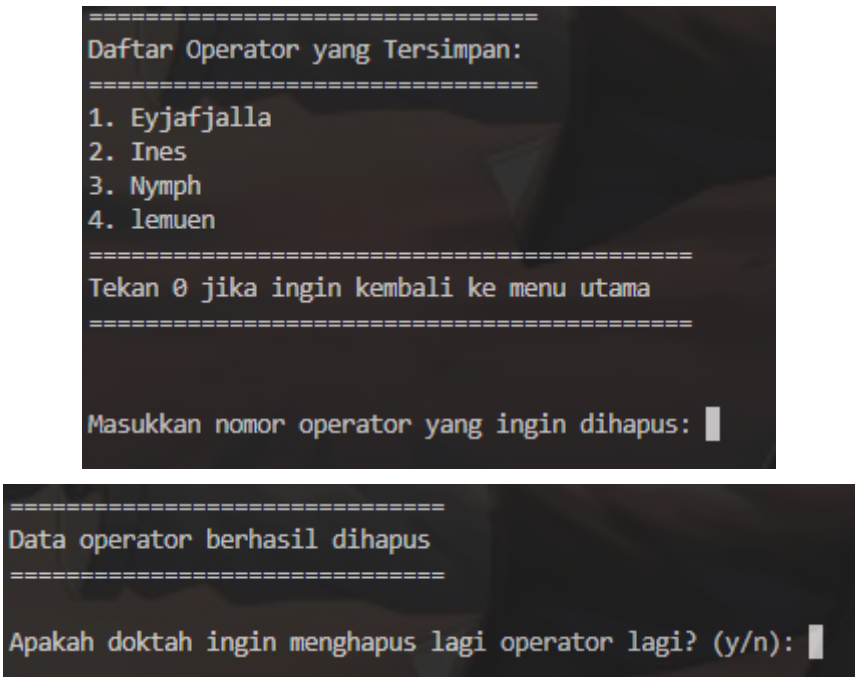


Gambar 4.6 Tampilan lihat data

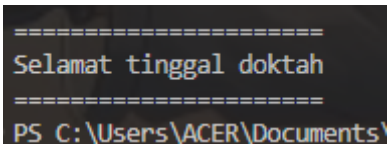




Gambar 4.6 Menu update dan hasil update



Gambar 4.7 Menu hapus dan hasil hapus

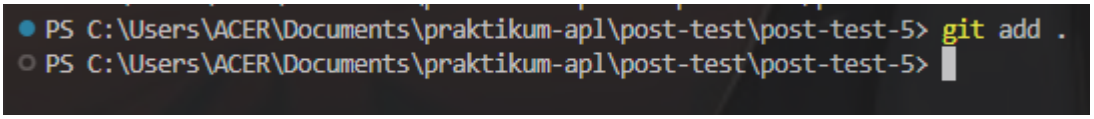


Gambar 4.8 Log out

5. Langkah-langkah Git

A. Git add

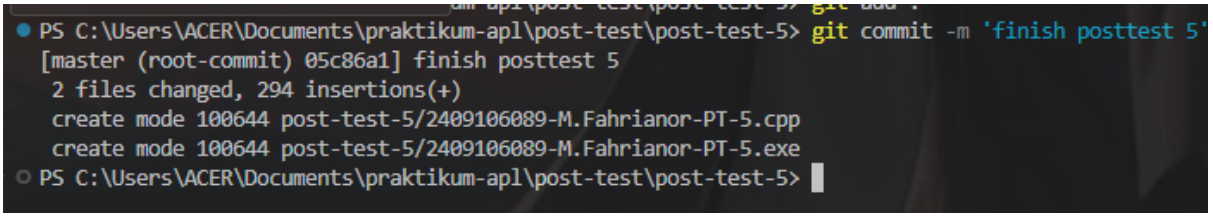
Perintah git add digunakan untuk menambahkan file apa saja sebelum dilakukannya commit
.



Gambar 5.2 Git add

B. Git commit

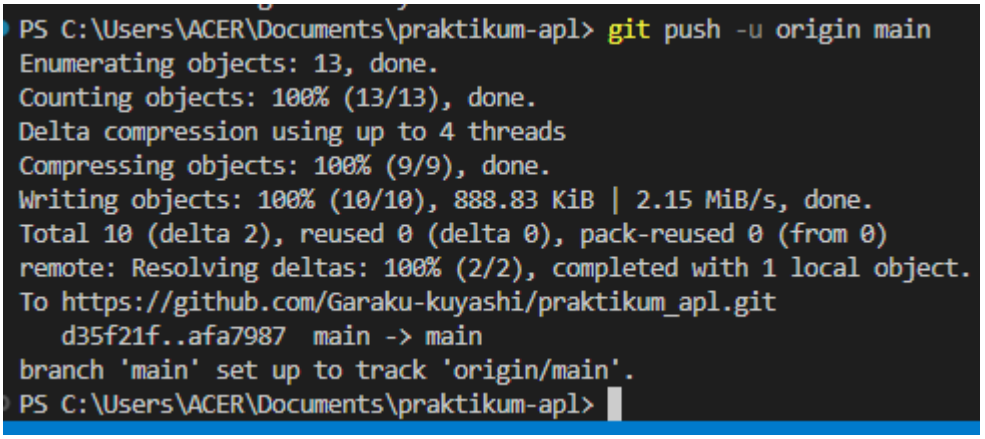
Perintah git commit digunakan untuk menyimpan perubahan yang telah di tambahkan ke dalam repository git



Gambar 5.3 Git commit

C. Git push

perintah git push digunakan untuk mengunggah perubahan dari repository lokal ke github.



Gambar 5.5 Git push