

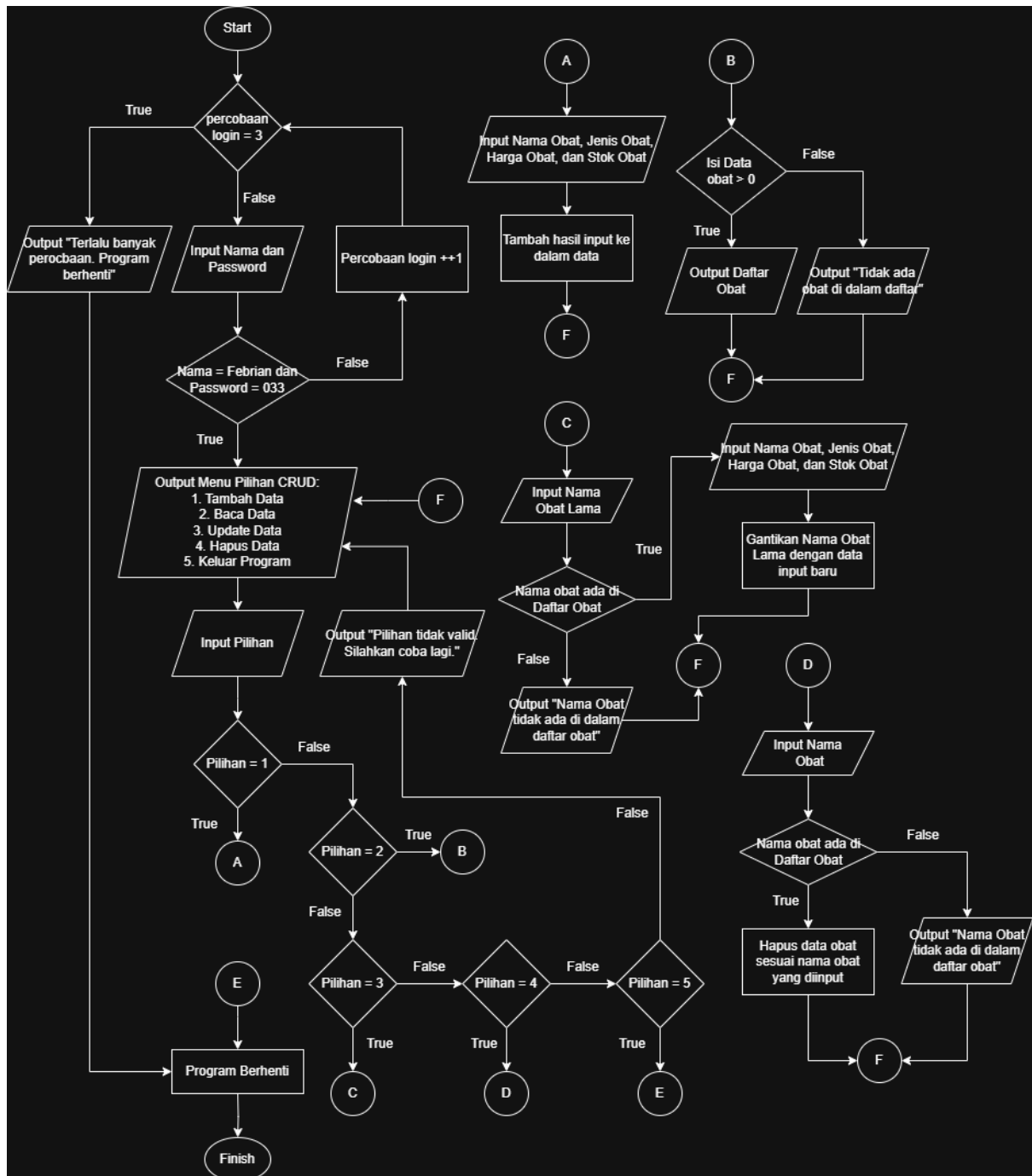
LAPORAN PRAKTIKUM
POSTTEST 2
ALGORITMA PEMROGRAMAN LANJUT



Disusun oleh:
Febrian Pratama Saputra (2409106033)
Kelas (A2 '24)

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

1. Flowchart



Gambar 1.1 Flowchart Algoritma

2. Analisis Program

2.1 Deskripsi Singkat Program

Program berbasis C++ ini berfungsi untuk mengurus data stok obat di sebuah apotek dengan fitur CRUD (Create, Read, Update, Delete). Program ini telah diamankan dengan fitur login untuk mencegah akses tidak sah.

2.2 Penjelasan Alur & Algoritma

Pertama, program meminta user untuk memasukkan Nama dan Password ke dalam terminal. Jika Nama dan Password tidak sama dengan yang ada di dalam data program, maka user diperkenalkan untuk mencoba lagi. Akan tetapi, jika percobaan login telah melebihi 3 kali, maka program akan langsung berhenti. Jika Nama dan Password sama dengan ada yang di dalam data program, maka user akan dialihkan ke menu utama program.

Pada menu utama program, program akan mengeluarkan sebuah menu di mana user bisa memilih di antara opsi di antara 1-5. Opsi pertama mengalihkan user untuk menambahkan sebuah data di dalam sistem yang berupa Nama Obat, Jenis Obat, Harga Obat, dan Stok Obat (Create). Opsi kedua menampilkan daftar obat yang telah di input oleh user jika ada (Read). Opsi ketiga mengalihkan user untuk memasukkan sebuah nama obat yang ada di dalam data untuk mengubah datanya (Update). Opsi keempat mengalihkan user untuk memasukkan sebuah nama obat untuk dihapus datanya dari sistem (Delete). Opsi kelima mengakhiri program tersebut. User akan di kembalikan lagi ke menu utama program setelah melaksanakan operasi CRUD. Jika user ingin keluar dari program, maka ia harus memasukkan pilihan kelima untuk keluar.

3. Source Code

A. Fitur Login

Fitur ini berfungsi untuk mengamankan program dari akses tidak sah. Program akan berhenti jika user salah memasukkan kredensial login 3x.

```
string namaLogin, passwordLogin;
string namaBenar = "Febrian", passwordBenar = "033";
int percobaanLogin = 0;

while (percobaanLogin < 3) {
    cout << "Masukkan Username: "; cin >> namaLogin;
    cout << "Masukkan Password: "; cin >> passwordLogin;

    if (namaLogin == namaBenar && passwordLogin == passwordBenar) {
        cout << "Login berhasil! Selamat datang pengguna program.\n";
        break;
    } else {
        cout << "Username atau Password salah! Silahkan coba lagi.\n";
        percobaanLogin++;
    }
}

if (percobaanLogin == 3) {
    cout << "Maaf! Terlalu banyak percobaan gagal. Program akan segera berhenti.\n";
    return 0;
}
```

B. Fitur Menu

Fitur ini berfungsi sebagai menu utama user untuk melakukan CRUD data. Program akan selalu looping kembali ke fitur ini untuk setiap fitur CRUD.

```
cout << "\nManajemen Stok Obat Apotek";
cout << "\n1. Tambah Obat";
cout << "\n2. Tampilkan Obat";
cout << "\n3. Ubah Obat";
cout << "\n4. Hapus Obat";
cout << "\n5. Keluar";
cout << "\nPilih menu: ";
cin >> pilihan;
```

C. Fitur CRUD

Fitur utama program. Fitur ini memberikan user kemampuan untuk melaksanakan CRUD (Create, Read, Update, Delete) pada sebuah data.

```
if (pilihan == 1) {
    cout << "\nMasukkan Nama Obat: "; cin >> namaObat[jumlah];
    cout << "Masukkan Jenis Obat: "; cin >> jenisObat[jumlah];
    cout << "Masukkan Harga Obat: "; cin >> hargaObat[jumlah];
    cout << "Masukkan Stok Obat: "; cin >> stokObat[jumlah];
    jumlah++;
    cout << "Obat berhasil ditambahkan!\n";
} else if (pilihan == 2) {
    if (jumlah == 0) {
        cout << "Tidak ada obat dalam sistem.\n";
    } else {
        cout << "\nDaftar Obat:\n";
        for (int i = 0; i < jumlah; i++) {
            cout << "Nama: " << namaObat[i] << ", Jenis: " <<
jenisObat[i] << ", Harga: " << hargaObat[i] << ", Stok: " << stokObat[i] <<
"\n";
        }
    }
} else if (pilihan == 3) {
    string ubah;
    cout << "\nMasukkan Nama Obat yang ingin diubah: "; cin >> ubah;
    for (int i = 0; i < jumlah; i++) {
        if (namaObat[i] == ubah) {
            cout << "Masukkan Nama Baru: "; cin >> namaObat[i];
            cout << "Masukkan Jenis Baru: "; cin >> jenisObat[i];
            cout << "Masukkan Harga Baru: "; cin >> hargaObat[i];
            cout << "Masukkan Stok Baru: "; cin >> stokObat[i];
            cout << "Obat berhasil diperbarui!\n";
            break;
        } else {cout << "Nama obat tidak ditemukan!";}
    }
} else if (pilihan == 4) {
    string hapus;
    cout << "\nMasukkan Nama Obat yang ingin dihapus: "; cin >> hapus;
    for (int i = 0; i < jumlah; i++) {
        if (namaObat[i] == hapus) {
            for (int j = i; j < jumlah - 1; j++) {
                namaObat[j] = namaObat[j + 1];
                jenisObat[j] = jenisObat[j + 1];
                hargaObat[j] = hargaObat[j + 1];
                stokObat[j] = stokObat[j + 1];
            }
            jumlah--;
            cout << "Obat berhasil dihapus!\n";
            break;
        }
    }
}
```

```
}  
}  
}
```

4. Uji Coba dan Hasil Output

4.1 Uji Coba

1. Skenario 1: User salah input username atau password 3x
2. Skenario 2: User ingin menambahkan obat yang bernama “Penincillin” yang berjenis “Antibiotik” dengan harga 30.000 yang memiliki stok 40 obat.
3. Skenario 3: User ingin melihat daftar obat yang ada di dalam data.
4. Skenario 4: User ingin mengubah data obat “Penincillin” menjadi sebuah obat yang bernama “Paracetamol” yang berjenis “Analgesik” dengan harga 20.000 yang memiliki stok 50 obat.
5. Skenario 5: User ingin menghapus obat “Paracetamol” dari data.

4.2 Hasil Output

```
Masukkan Username: TangaIbuki  
Masukkan Password: NatusmeIroha  
Username atau Password salah! Silahkan coba lagi.  
Masukkan Username: TakanashiHoshino  
Masukkan Password: SunaookamiShiroko  
Username atau Password salah! Silahkan coba lagi.  
Masukkan Username: SorasakiHina  
Masukkan Password: ShiromiIori  
Username atau Password salah! Silahkan coba lagi.  
Maaf! Terlalu banyak percobaan gagal. Program akan segera berhenti.
```

Gambar 4.0.1 Skenario 1: Gagal Login

```
Masukkan Username: Febrian
Masukkan Password: 033
Login berhasil! Selamat datang pengguna program.

Manajemen Stok Obat Apotek
1. Tambah Obat
2. Tampilkan Obat
3. Ubah Obat
4. Hapus Obat
5. Keluar
Pilih menu: 1

Masukkan Nama Obat: Penincillin
Masukkan Jenis Obat: Antibiotik
Masukkan Harga Obat: 30000
Masukkan Stok Obat: 40
Obat berhasil ditambahkan!
```

Gambar 4.0.2 Skenario 2: Create

```
Manajemen Stok Obat Apotek
1. Tambah Obat
2. Tampilkan Obat
3. Ubah Obat
4. Hapus Obat
5. Keluar
Pilih menu: 2

Daftar Obat:
Nama: Penincillin, Jenis: Antibiotik, Harga: 30000, Stok: 40
```

Gambar 0.3 Skenario 3: Read

```

Manajemen Stok Obat Apotek
1. Tambah Obat
2. Tampilkan Obat
3. Ubah Obat
4. Hapus Obat
5. Keluar
Pilih menu: 3

Masukkan Nama Obat yang ingin diubah: Penincillin
Masukkan Nama Baru: Paracetamol
Masukkan Jenis Baru: Analgesik
Masukkan Harga Baru: 20000
Masukkan Stok Baru: 50
Obat berhasil diperbarui!

Manajemen Stok Obat Apotek
1. Tambah Obat
2. Tampilkan Obat
3. Ubah Obat
4. Hapus Obat
5. Keluar
Pilih menu: 2

Daftar Obat:
Nama: Paracetamol, Jenis: Analgesik, Harga: 20000, Stok: 50

```

Gambar 4.0.4 Skenario 4: Update

```

Manajemen Stok Obat Apotek
1. Tambah Obat
2. Tampilkan Obat
3. Ubah Obat
4. Hapus Obat
5. Keluar
Pilih menu: 4

Masukkan Nama Obat yang ingin dihapus: Paracetamol
Obat berhasil dihapus!

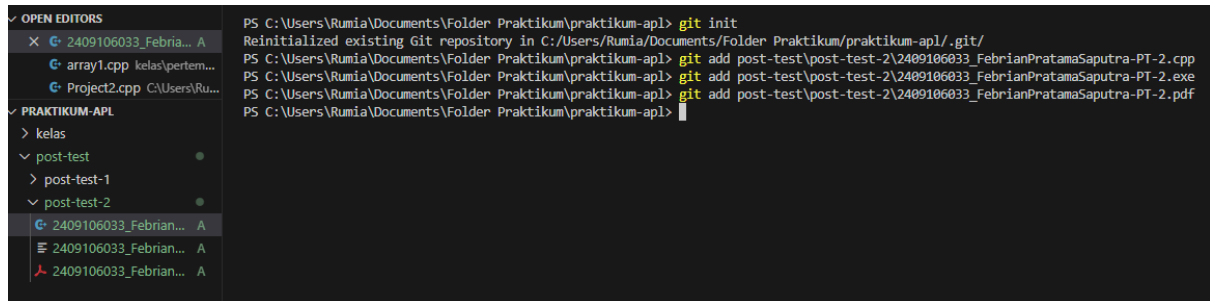
```

Gambar 4.0.5 Skenario 5: Delete

5. Langkah-langkah Git

1. Git add

Berfungsi untuk menambahkan file-file yang ingin dicommit. Kali ini kita akan menambahkan file untuk dicommit satu per satu.

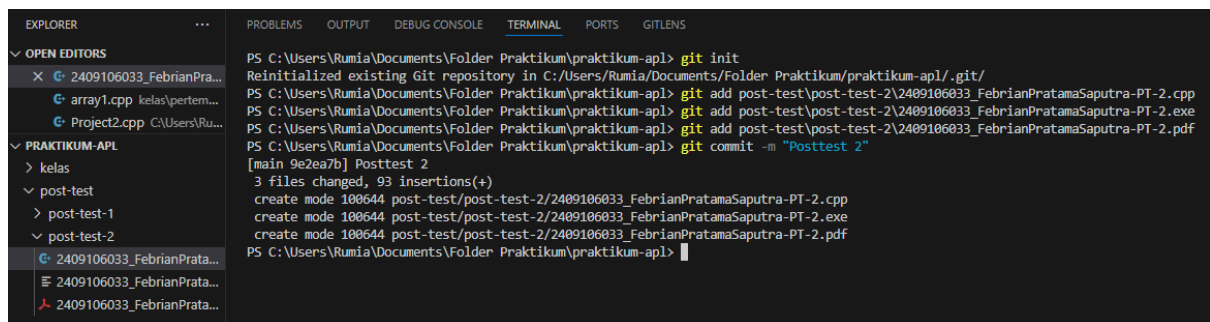


```
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git init
Reinitialized existing Git repository in C:/Users/Rumia/Documents/Folder Praktikum/praktikum-apl/.git/
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.cpp
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.exe
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.pdf
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl>
```

Gambar 5.1 Git Add

2. Git commit

Berfungsi untuk mengcommit file yang ditambahkan untuk dijadikan sebuah checkpoint.

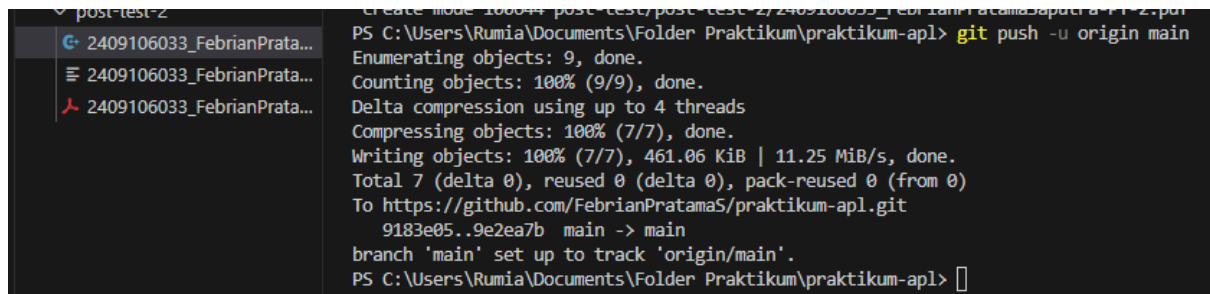


```
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git init
Reinitialized existing Git repository in C:/Users/Rumia/Documents/Folder Praktikum/praktikum-apl/.git/
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.cpp
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.exe
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git add post-test\post-test-2\2409106033_FebrianPratamaSaputra-PT-2.pdf
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git commit -m "Posttest 2"
[main 9e2ea7b] Posttest 2
3 files changed, 93 insertions(+)
create mode 100644 post-test/post-test-2/2409106033_FebrianPratamaSaputra-PT-2.cpp
create mode 100644 post-test/post-test-2/2409106033_FebrianPratamaSaputra-PT-2.exe
create mode 100644 post-test/post-test-2/2409106033_FebrianPratamaSaputra-PT-2.pdf
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl>
```

Gambar 5.2 Git Commit

3. Git push

Berfungsi untuk menyambungkan git lokal ke github yang telah di hubungkan melalaui git remote.



```
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl> git push -u origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 461.06 KiB | 11.25 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/FebrianPratamaS/praktikum-apl.git
9183e05..9e2ea7b main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\Rumia\Documents\Folder Praktikum\praktikum-apl>
```

Gambar 5.3 Git Push