LAPORAN PRAKTIKUM

POSTTEST 5

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Ikhsan (2409106118)

Kelas (C2 ‘24)

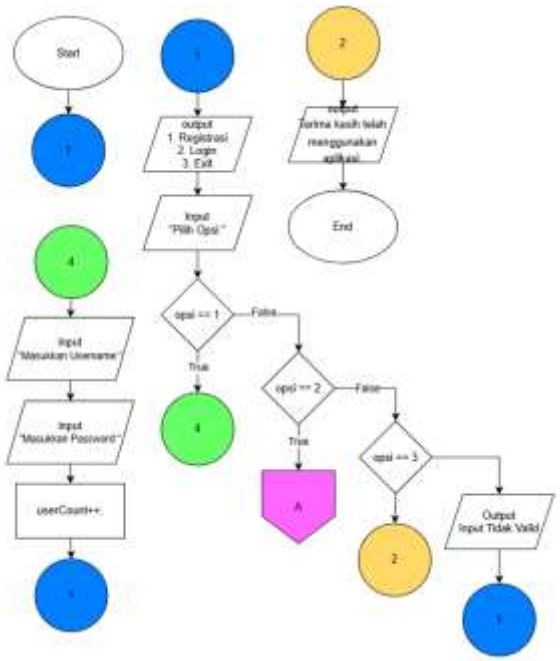
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

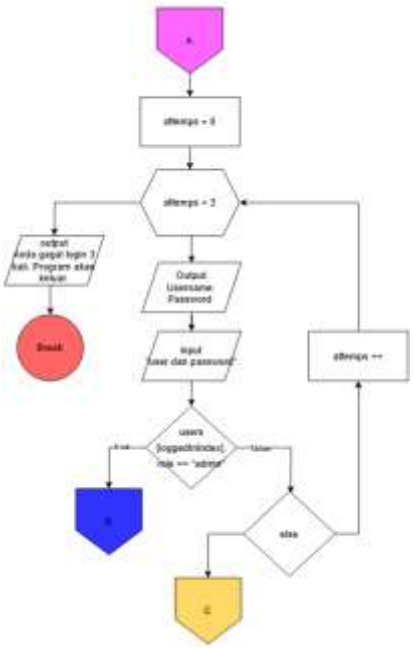
SAMARINDA

2025

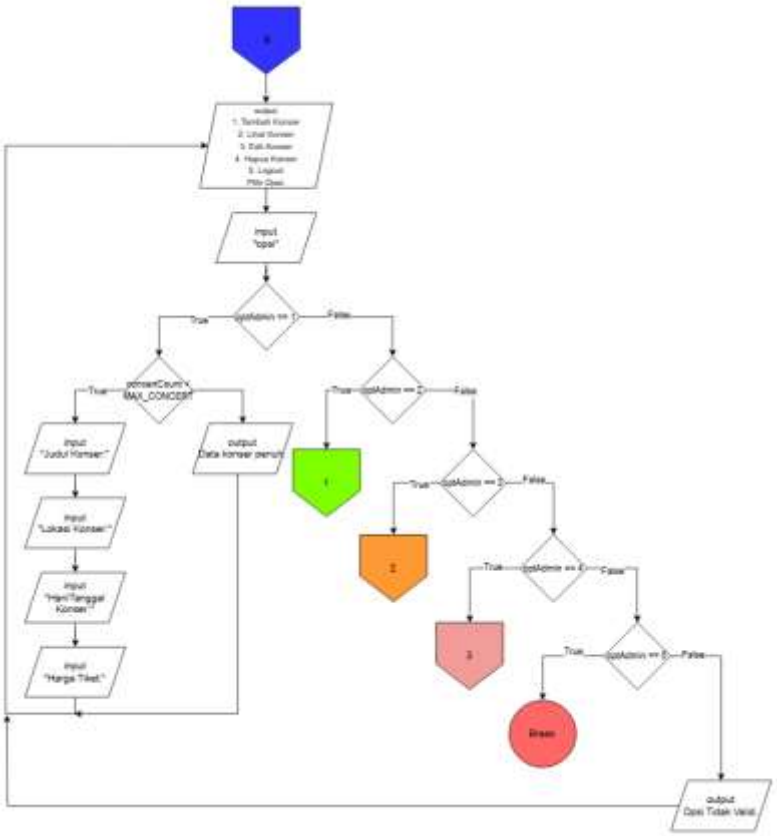
1. Flowchart



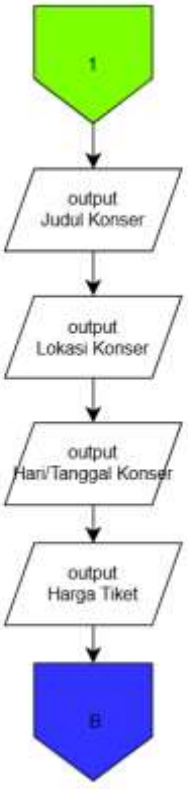
Gambar 1.1 Flowchart



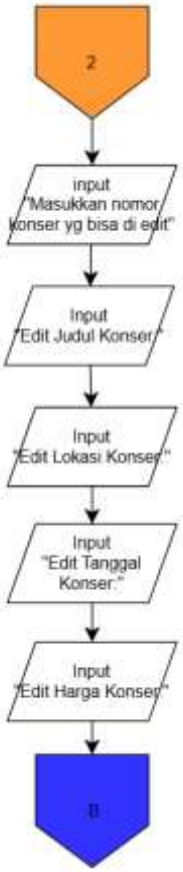
Gambar 1.2 Flowchart



Gambar 1.3 Flowchart

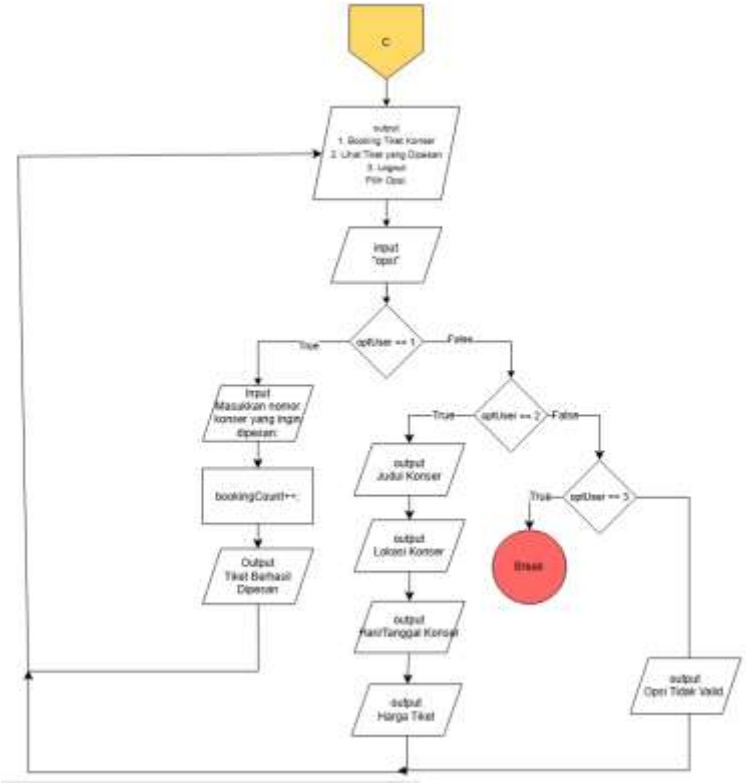


Gambar 1.4 Flowchart



Gambar 1.5 Flowchart

Gambar 1.6 Flowchart



Gambar 1.7 Flowchart

2. Analisis Program

2.1 Tujuan dan Fungsi Program

Program ini dibuat untuk mengelola pemesanan tiket konser secara sederhana dengan fitur:

1. **Registrasi & Login Pengguna**
    - Pengguna dapat mendaftar dan masuk ke sistem.
    - Tersedia dua jenis akun: **admin** (untuk manajemen data konser) dan **user** (untuk pemesanan tiket).
2. **Fitur Admin**
    - Menambahkan konser baru.
    - Menampilkan daftar konser dalam bentuk tabel.
    - Mengedit data konser yang sudah ada.
    - Menghapus konser dari sistem.
3. **Fitur Pengguna**
    - Menampilkan daftar konser dalam bentuk tabel.
    - Memesan tiket konser.
    - Menampilkan daftar tiket yang telah dipesan dalam bentuk tabel.

2.2 Manfaat Utama

- **Mudah Digunakan:** Antarmuka berbasis teks yang sederhana, dengan struktur menu yang jelas dan terarah.
- **Manajemen Tiket Efektif:** Admin memiliki kontrol penuh terhadap data konser, sementara pengguna cukup fokus pada pemesanan dan melihat daftar tiket.
- **Keamanan:** Sistem login membatasi tiga kali percobaan untuk mencegah akses tidak sah.
- **Penyimpanan Data Sementara:** Data konser, pengguna, dan pemesanan disimpan dalam array selama program berjalan, sesuai untuk penggunaan lokal atau simulasi kecil.

2.3 Desain Modularitas dan Alur Program

Untuk menjaga **keterbacaan** dan **pemeliharaan kode**, setiap fitur menu dipisahkan menjadi **subprogram** (fungsi atau prosedur) yang spesifik:

- registrasi(...): Prosedur untuk pendaftaran pengguna baru.
- login(...): Fungsi yang mengembalikan indeks pengguna jika berhasil login, atau –1 jika gagal.
- tambahKonser(...), lihatKonser(...), editKonser(...), hapusKonser(...): Prosedur admin untuk pengelolaan data konser.
- bookingTiket(...), lihatBooking(...): Prosedur pengguna untuk melakukan pemesanan tiket dan melihat daftar tiket yang telah dipesan.

Setiap subprogram menerima **parameter** berupa data yang diperlukan (misalnya array dan variabel jumlah elemen), sehingga tidak bergantung pada variabel global. Prosedur digunakan untuk aksi-aksi tanpa nilai kembali, sedangkan fungsi login digunakan untuk mengembalikan nilai berupa hasil autentikasi pengguna.

2.4 Nilai Tambah: Penerapan Pointer (Address-of & Dereference)

Sebagai nilai tambah dari program, kami menerapkan konsep **pointer** dalam dua fungsi, yaitu:

1. Fungsi dengan Parameter Address-of

```
void tambahSatuPengguna(int* alamatJumlahUser) {
   (*alamatJumlahUser)++;
}
```

Fungsi ini digunakan untuk menambahkan jumlah pengguna secara langsung melalui **alamat memori** variabel jumlahUser. Nilai jumlahUser akan bertambah karena pointer mengakses dan memodifikasinya secara langsung dari alamat aslinya. Fungsi ini dipanggil menggunakan operator &:

```
tambahSatuPengguna(&jumlahUser);
```

2. Fungsi dengan Parameter Dereference

```
void tampilkanJumlahKonser(int* ptrJumlahKonser) {
   cout << "Jumlah konser saat ini: " << *ptrJumlahKonser << "\n";
}
```

Fungsi ini digunakan untuk **menampilkan nilai** dari jumlahKonser melalui **dereferensi pointer**. Fungsi menerima parameter berupa alamat, kemudian menampilkan nilai di dalam alamat tersebut. Pemanggilannya adalah:

```
tampilkanJumlahKonser(&jumlahKonser);
```

Kesimpulan Penerapan Pointer

Dengan menggunakan fungsi berparameter pointer (address-of dan dereference), program menjadi lebih fleksibel dalam **mengakses dan memodifikasi nilai secara langsung dari memori**. Ini menunjukkan bahwa program tidak hanya mengikuti prinsip modular, tetapi juga memahami dan mengimplementasikan manajemen memori tingkat rendah yang umum digunakan dalam pemrograman bahasa C++.

3. Source Code

```cpp
#include <iostream>

using namespace std;

#define MAKS_KONSER 100
#define MAKS_BOOKING 100
#define MAKS_USER 100

struct Konser {
    string judul, lokasi, tanggal, harga;
};

struct Booking {
```

```
    Konser konser;
};

struct Pengguna {
    string nama, sandi, peran;
    int jumlahBooking;
    Booking daftarBooking[MAKS_BOOKING];
};

// Overloading tampilkanPesan
void tampilkanPesan(const string &pesan) {
    cout << pesan << "\n";
}
void tampilkanPesan(const string &pesan, int x) {
    cout << pesan << x << "\n";
}

// Prosedur registrasi
void registrasi(Pengguna pengguna[], int &jumlahUser) {
    tampilkanPesan("\n=== Registrasi Pengguna Baru ===");
    if (jumlahUser < MAKS_USER) {
        cout << "Masukkan nama pengguna: "; cin >>
pengguna[jumlahUser].nama;
        cout << "Masukkan kata sandi: ";     cin >>
pengguna[jumlahUser].sandi;
        pengguna[jumlahUser].peran = "user";
        pengguna[jumlahUser].jumlahBooking = 0;
        jumlahUser++;
        tampilkanPesan("Registrasi berhasil! Silakan login.");
    } else {
        tampilkanPesan("Maksimum pengguna tercapai.");
    }
}

// Fungsi login
int login(Pengguna pengguna[], int jumlahUser) {
    int percobaan = 0;
    while (percobaan < 3) {
        string nama, sandi;
        cout << "Nama pengguna: "; cin >> nama;
        cout << "Kata sandi: ";    cin >> sandi;
        for (int i = 0; i < jumlahUser; i++) {
            if (pengguna[i].nama == nama && pengguna[i].sandi == sandi) {
                tampilkanPesan("Login berhasil!");
                return i;
            }
        }
        percobaan++;
```

```
        tampilkanPesan("Login gagal. Sisa percobaan: ", 3 - percobaan);
    }
    tampilkanPesan("Anda gagal login 3 kali. Program keluar.");
    return -1;
}

// Rekursif menampilkan tabel konser
void tampilkanTabelKonserRekursif(const Konser konser[], int idx, int
jumlah) {
    if (idx == 0) {
        cout << "+----+----------------------+-------------+--------------
--+----------+\n";
        cout << "| No | Judul                | Lokasi      |
Tanggal         | Harga    |\n";
        cout << "+----+----------------------+-------------+--------------
--+----------+\n";
    }
    if (idx >= jumlah) {
        cout << "+----+----------------------+-------------+--------------
--+----------+\n";
        return;
    }
    auto &k = konser[idx];
    printf("| %2d | %-20s | %-12s | %-15s | %8s |\n",
        idx+1,
        k.judul.c_str(),
        k.lokasi.c_str(),
        k.tanggal.c_str(),
        k.harga.c_str()
    );
    tampilkanTabelKonserRekursif(konser, idx+1, jumlah);
}

// Prosedur admin
void tambahKonser(Konser konser[], int &jumlah) {
    if (jumlah >= MAKS_KONSER) {
        tampilkanPesan("Data konser penuh.");
        return;
    }
    tampilkanPesan("\n=== Tambah Konser ===");
    cin.ignore();
    cout << "Judul Konser: ";    getline(cin, konser[jumlah].judul);
    cout << "Lokasi Konser: ";   getline(cin, konser[jumlah].lokasi);
    cout << "Tanggal Konser: ";  getline(cin, konser[jumlah].tanggal);
    cout << "Harga Tiket: ";     getline(cin, konser[jumlah].harga);
    jumlah++;
    tampilkanPesan("Konser berhasil ditambahkan.");
}
```

```
void lihatKonser(const Konser konser[], int jumlah) {
    tampilkanTabelKonserRekursif(konser, 0, jumlah);
}

void editKonser(Konser konser[], int jumlah) {
    if (!jumlah) { tampilkanPesan("Tidak ada konser untuk diedit."); return;
}
    lihatKonser(konser, jumlah);
    cout << "Pilih nomor konser: "; int no; cin >> no; cin.ignore();
    if (no < 1 || no > jumlah) { tampilkanPesan("Nomor tidak valid.");
return; }
    int i = no - 1;
    cout << "Edit Judul (" << konser[i].judul << "): "; getline(cin,
konser[i].judul);
    cout << "Edit Lokasi (" << konser[i].lokasi << "): "; getline(cin,
konser[i].lokasi);
    cout << "Edit Tanggal (" << konser[i].tanggal << "): "; getline(cin,
konser[i].tanggal);
    cout << "Edit Harga (" << konser[i].harga << "): "; getline(cin,
konser[i].harga);
    tampilkanPesan("Konser berhasil diubah.");
}

void hapusKonser(Konser konser[], int &jumlah) {
    if (!jumlah) { tampilkanPesan("Tidak ada konser untuk dihapus.");
return; }
    lihatKonser(konser, jumlah);
    cout << "Pilih nomor konser: "; int no; cin >> no;
    if (no < 1 || no > jumlah) { tampilkanPesan("Nomor tidak valid.");
return; }
    for (int i = no - 1; i < jumlah - 1; i++) konser[i] = konser[i + 1];
    jumlah--;
    tampilkanPesan("Konser berhasil dihapus.");
}

// Prosedur user
void bookingTiket(Pengguna &u, const Konser konser[], int jumlah) {
    lihatKonser(konser, jumlah);
    cout << "Pilih nomor konser: "; int no; cin >> no;
    if (no < 1 || no > jumlah) { tampilkanPesan("Nomor tidak valid.");
return; }
    u.daftarBooking[u.jumlahBooking++].konser = konser[no - 1];
    tampilkanPesan("Tiket berhasil dipesan.");
}

void lihatBooking(const Pengguna &u) {
```

```
    if (!u.jumlahBooking) { tampilkanPesan("Anda belum memesan tiket.");
return; }
    cout << "+----+----------------------+-------------+----------------+-
---------+\n";
    cout << "| No | Judul                | Lokasi       | Tanggal         |
Harga    |\n";
    cout << "+----+----------------------+-------------+----------------+-
---------+\n";
    for (int i = 0; i < u.jumlahBooking; i++) {
        auto &k = u.daftarBooking[i].konser;
        printf("| %2d | %-20s | %-12s | %-15s | %8s |\n",
            i+1,
            k.judul.c_str(),
            k.lokasi.c_str(),
            k.tanggal.c_str(),
            k.harga.c_str()
        );
    }
    cout << "+----+----------------------+-------------+----------------+-
---------+\n";
}

// Fungsi dengan parameter address-of
void tambahSatuPengguna(int* alamatJumlahUser) {
    (*alamatJumlahUser)++;
}

// Fungsi dengan parameter dereference
void tampilkanJumlahKonser(int* ptrJumlahKonser) {
    cout << "Jumlah konser saat ini: " << *ptrJumlahKonser << "\n";
}

int main() {
    Konser konser[MAKS_KONSER] = {
        {"Midnigth Serenade","Samarinda","11-Oktober-2024","200000"},
        {"Arctic Monkey","Balikpapan","12-Oktober-2024","500000"}
    };
    int jumlahKonser = 2;

    Pengguna pengguna[MAKS_USER];
    int jumlahUser = 2;
    pengguna[0] = {"admin","admin","admin",0};
    pengguna[1] = {"Ikhsan","2409106118","user",0};

    while (true) {
        tampilkanPesan("\n=== Pemesanan Tiket Konser ===");
        tampilkanPesan("1. Register");
        tampilkanPesan("2. Login");
```

```
        tampilkanPesan("3. Exit");
        tampilkanPesan("Pilih opsi: ");
        int opsi; cin >> opsi;
        if (opsi == 1) {
            registrasi(pengguna, jumlahUser);
        }
        else if (opsi == 2) {
            int idx = login(pengguna, jumlahUser);
            if (idx < 0) break;
            if (pengguna[idx].peran == "admin") {
                while (true) {
                    tampilkanPesan("\n-- Menu Admin --");
                    tampilkanPesan("1.Tambah");
                    tampilkanPesan("2.Lihat");
                    tampilkanPesan("3.Edit");
                    tampilkanPesan("4.Hapus");
                    tampilkanPesan("5.Logout");
                    tampilkanPesan("Pilih: ");
                    int o; cin >> o;
                    if (o == 1) tambahKonser(konser, jumlahKonser);
                    else if (o == 2) lihatKonser(konser, jumlahKonser);
                    else if (o == 3) editKonser(konser, jumlahKonser);
                    else if (o == 4) hapusKonser(konser, jumlahKonser);
                    else if (o == 5) break;
                }
            } else {
                while (true) {
                    tampilkanPesan("\n-- Menu User --");
                    tampilkanPesan("1.Booking Tiket Konser");
                    tampilkanPesan("2.Lihat Tiket yang Dipesan");
                    tampilkanPesan("3.Logout");
                    tampilkanPesan("Pilih: ");
                    int o; cin >> o;
                    if (o == 1) bookingTiket(pengguna[idx], konser,
jumlahKonser);
                    else if (o == 2) lihatBooking(pengguna[idx]);
                    else if (o == 3) break;
                }
            }
        }
        else break;
    }

    // Panggilan fungsi address-of dan dereference
    tambahSatuPengguna(&jumlahUser);
    tampilkanPesan("\nSimulasi: Jumlah pengguna ditambah satu secara
manual.");
    tampilkanPesan("Total pengguna sekarang: ", jumlahUser);
```

```
    tampilkanPesan("\nMenampilkan jumlah konser dengan pointer:");
    tampilkanJumlahKonser(&jumlahKonser);

    return 0;
}
```

4. Uji Coba dan Hasil Output

```
=== Pemesanan Tiket Konser ===
1. Register
2. Login
3. Exit
Pilih opsi:
3. Exit
Pilih opsi:
1

=== Registrasi Pengguna Baru ===
Masukkan nama pengguna: Jamal
Masukkan kata sandi: 12345678
Registrasi berhasil! Silakan login.

=== Pemesanan Tiket Konser ===
1. Register
2. Login
3. Exit
Pilih opsi:
2
Nama pengguna: Jamal
Kata sandi: 12345678
Login berhasil!

-- Menu User --
1.Booking Tiket Konser
2.Lihat Tiket yang Dipesan
3.Logout
Pilih:
1
+----+----------------------+--------------+-----------------+----------+
| No | Judul                | Lokasi       | Tanggal         | Harga    |
+----+----------------------+--------------+-----------------+----------+
|  1 | Midnigth Serenade    | Samarinda    | 11-Oktober-2024 |   200000 |
|  2 | Arctic Monkey        | Balikpapan   | 12-Oktober-2024 |   500000 |
+----+----------------------+--------------+-----------------+----------+
Pilih nomor konser:
```

Gambar 4.1 Output Regis, Login dan Beli Tiket

```
-- Menu User --
1.Booking Tiket Konser
2.Lihat Tiket yang Dipesan
3.Logout
Pilih:
2
+----+----------------------+-------------+-----------------+----------+
| No | Judul                | Lokasi      | Tanggal         | Harga    |
+----+----------------------+-------------+-----------------+----------+
|  1 | Arctic Monkey        | Balikpapan  | 12-Oktober-2024 |   500000 |
+----+----------------------+-------------+-----------------+----------+

-- Menu User --
1.Booking Tiket Konser
2.Lihat Tiket yang Dipesan
3.Logout
Pilih:
3

=== Pemesanan Tiket Konser ===
1. Register
2. Login
3. Exit
Pilih opsi:
```

Gambar 4.2 Output Lihat Tiket Konser Acc Baru

```
-- Menu Admin --
1.Tambah
2.Lihat
3.Edit
4.Hapus
5.Logout
Pilih:
1

=== Tambah Konser ===
Judul Konser: FUR
Lokasi Konser: Tanah Grogot
Tanggal Konser: 17-Agustus-2026
Harga Tiket: 150000
Konser berhasil ditambahkan.
```

Gambar 4.3 Output Admin Tambah Konser

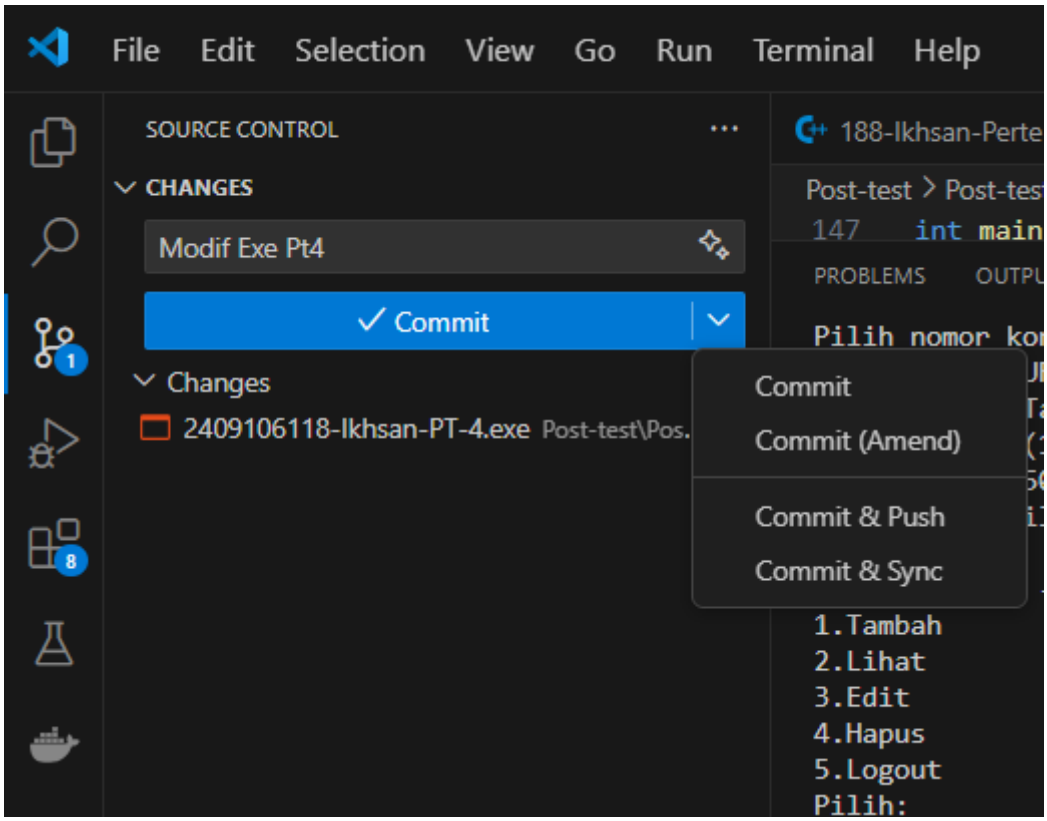
```
-- Menu Admin --
1.Tambah
2.Lihat
3.Edit
4.Hapus
5.Logout
Pilih:
        2
+----+----------------------+--------------+-----------------+----------+
| No | Judul                | Lokasi       | Tanggal         | Harga    |
+----+----------------------+--------------+-----------------+----------+
|  1 | Midnigth Serenade    | Samarinda    | 11-Oktober-2024 |   200000 |
|  2 | Arctic Monkey        | Balikpapan   | 12-Oktober-2024 |   500000 |
|  3 | FUR                  | Tanah Grogot | 17-Agustus-2026 |   150000 |
+----+----------------------+--------------+-----------------+----------+

-- Menu Admin --
1.Tambah
2.Lihat
3.Edit
4.Hapus
5.Logout
Pilih:
3
+----+----------------------+--------------+-----------------+----------+
| No | Judul                | Lokasi       | Tanggal         | Harga    |
+----+----------------------+--------------+-----------------+----------+
|  1 | Midnigth Serenade    | Samarinda    | 11-Oktober-2024 |   200000 |
|  2 | Arctic Monkey        | Balikpapan   | 12-Oktober-2024 |   500000 |
|  3 | FUR                  | Tanah Grogot | 17-Agustus-2026 |   150000 |
+----+----------------------+--------------+-----------------+----------+
Pilih nomor konser: 3
Edit Judul (FUR): Wali
Edit Lokasi (Tanah Grogot): Banjarmasin
Edit Tanggal (17-Agustus-2026): 17-Agustus-2026
Edit Harga (150000): 190000
Konser berhasil diubah.
```

Gambar 4.4 Output Lihat Konser dan Edit

```
-- Menu Admin --
1.Tambah
2.Lihat
3.Edit
4.Hapus
5.Logout
Pilih:
4
+----+----------------------+--------------+-----------------+----------+
| No | Judul                | Lokasi       | Tanggal         | Harga    |
+----+----------------------+--------------+-----------------+----------+
|  1 | Midnigth Serenade    | Samarinda    | 11-Oktober-2024 |   200000 |
|  2 | Arctic Monkey        | Balikpapan   | 12-Oktober-2024 |   500000 |
+----+----------------------+--------------+-----------------+----------+
Pilih nomor konser: 2
Konser berhasil dihapus.

-- Menu Admin --
1.Tambah
2.Lihat
3.Edit
4.Hapus
5.Logout
Pilih:
5

=== Pemesanan Tiket Konser ===
1. Register
2. Login
3. Exit
Pilih opsi:
3

Simulasi: Jumlah pengguna ditambah satu secara manual.
Total pengguna sekarang: 3

Menampilkan jumlah konser dengan pointer:
Jumlah konser saat ini: 1
PS D:\GITHUB\Praktikum-Apl\Post-test\Post-test-5>
```

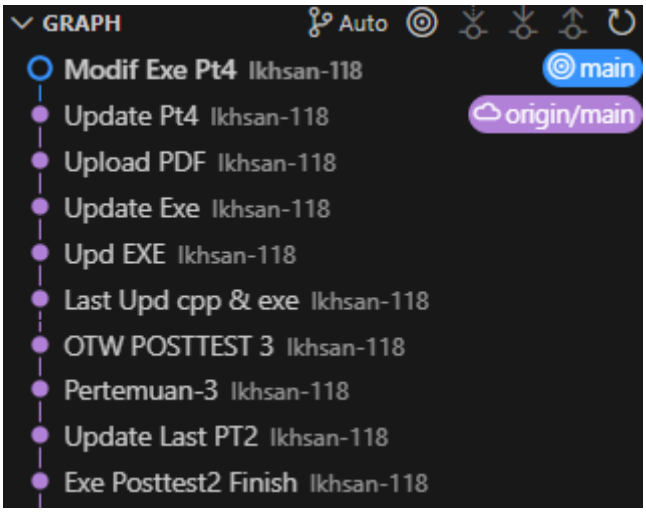
Gambar 4.5 Output Hapus hingga Exit

5. Langkah-Langkah Git pada VSCode



Gambar 5.1 Source Control Git Visual Code Studio

Pertama buka source control pada bagian sidebar di aplikasi Visual Code Studio, lalu input pesan yang ingin dipasang di message commit cintihnya seperti punya saya “Modif Exe Pt4”, lalu pencet tanda panah kebawah untuk mencari opsi “Commit & Push” agar programnya terupdate dan tersimpan di Github Cloud.



Gambar 5.2 Hasil apabila berhasil masuk Cloud