

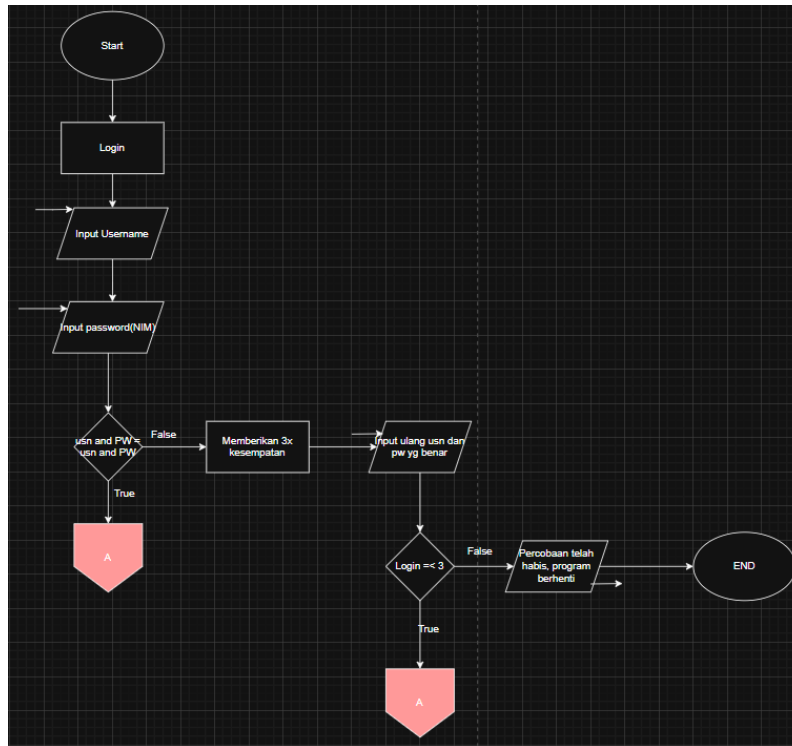
LAPORAN PRAKTIKUM
POSTTEST 2
ALGORITMA PEMROGRAMAN LANJUT



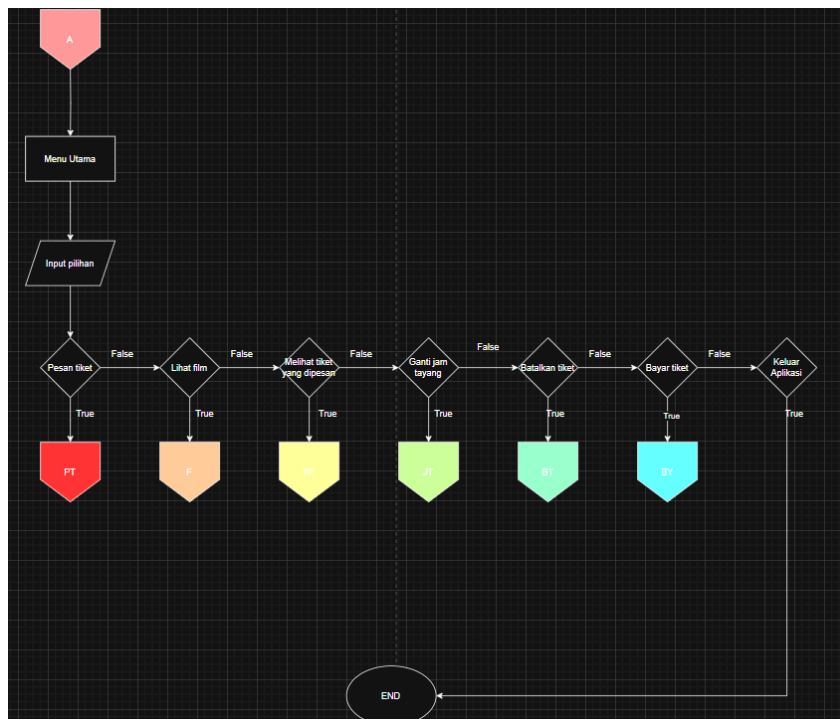
Disusun oleh:
Danendra Hazzel Putra Wahana 2409106096
Kelas C1 '24

PROGRAM STUDI INFORMATIKA
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

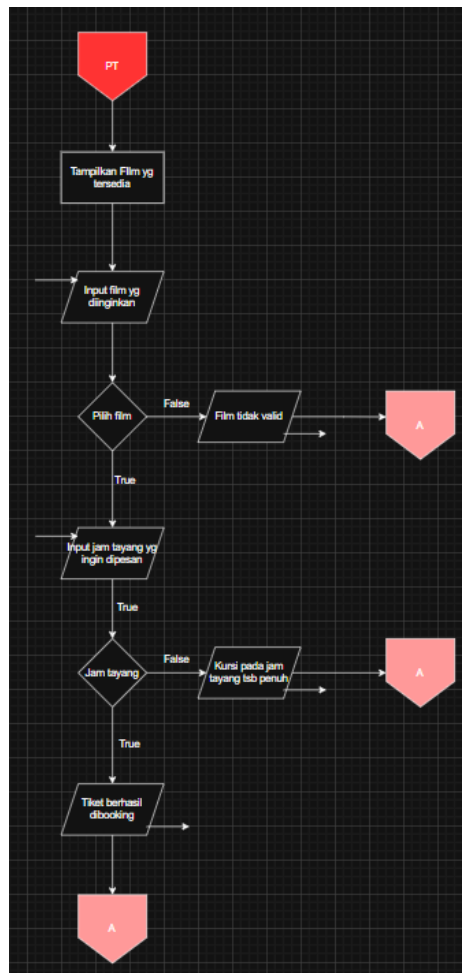
1. Flowchart



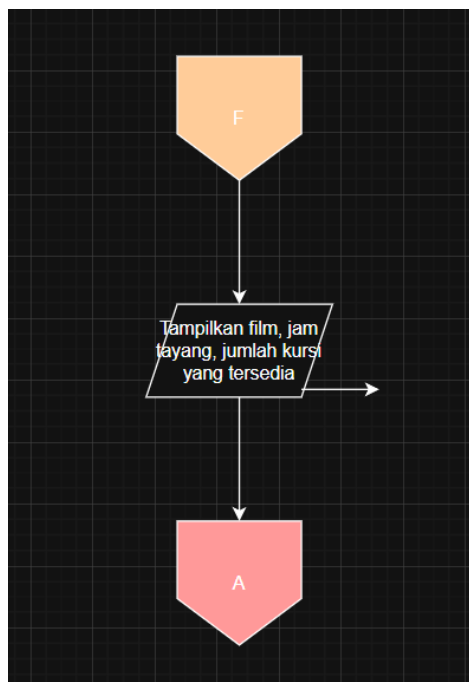
Gambar 1.1 Flowchart bagian menu login



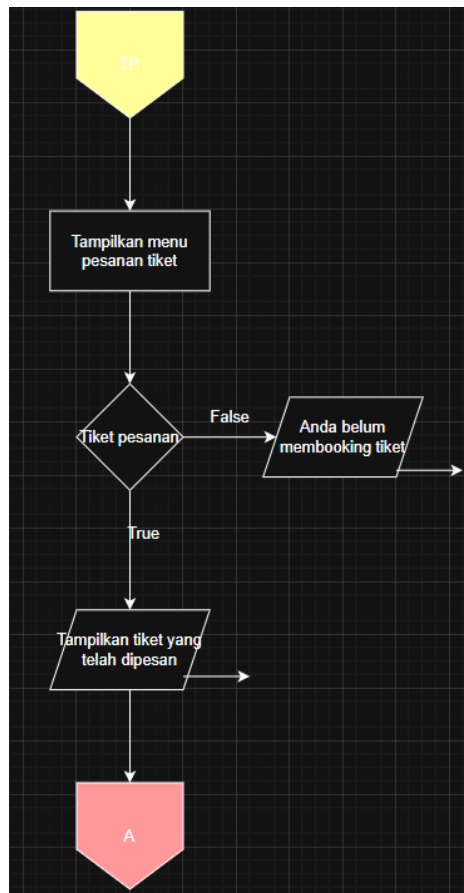
Gambar 1.2 Flowchart bagian menu utama



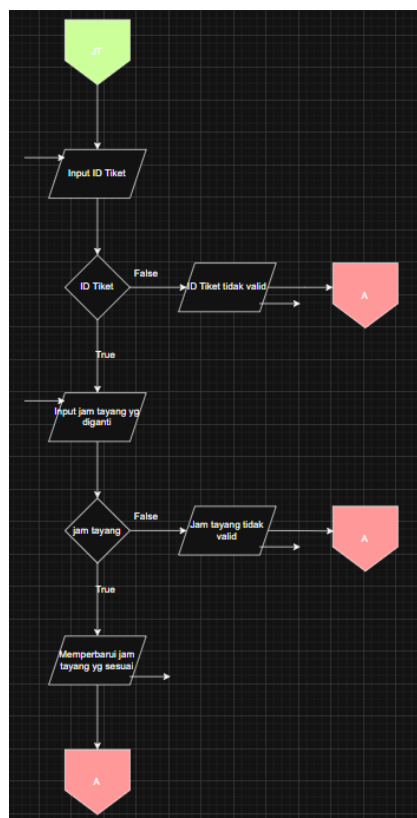
Gambar 1.3 Flowchart bagian Pemesana Tiket



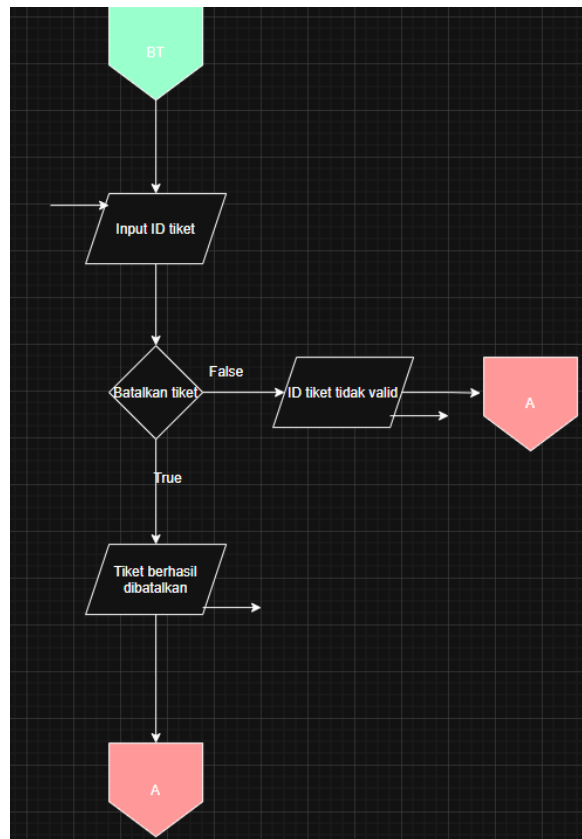
Gambar 1.4 Flowchart bagian Tampilkan film yang tersedia



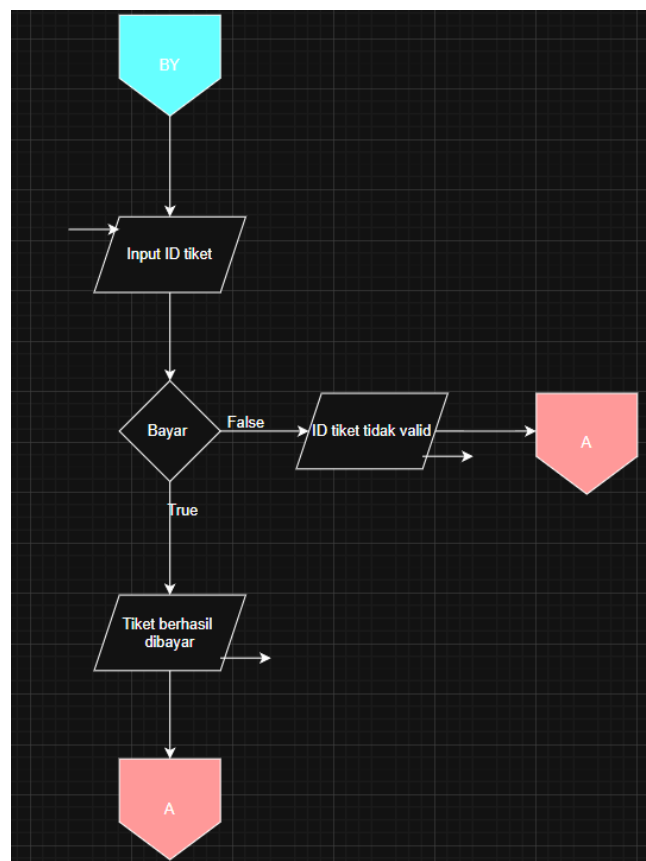
Gambar 1.5 Flowchart bagian Menu tampilan tiket yang telah dipesan



Gambar 1.6 Flowchart bagian Mengganti jam tayang



Gambar 1.7 Flowchart bagian Pembatalan tiket yang telah dipesan



Gambar 1.8 Flowchart bagian Pembayaran tiket yang dibooking

2. Analisis Program

2.1 Deskripsi Singkat Program

1. Tujuan Program:

- Program ini dibuat untuk memudahkan para user memesan ataupun membooking tiket film yang ia inginkan secara online agar tidak repot untuk datang jauh ke bioskop.

2. Fungsi/Manfaat Utama:

- Memfasilitasi User untuk dapat melihat dan memesan maupun membooking tiket secara online dari jauh dimanapun dan kapanpun.
- Dapat mengubah jam tayang yang user mau jika ada kendala maupun keadaan mendadak.
- Dapat membatalkan tiket yang dibooking jika berubah pikiran atau dalam keadaan terdesak jika tiket belum dibayar.
- Dapat membayar secara online tiket yang sudah dipesan secara praktis.

2.2 Penjelasan Alur & Algoritma

A. Login

Pada bagian ini, program meminta username dan password dalam bentuk nama panjang dan NIM user. Jika sesuai, maka login berhasil dan akan lanjut ke menu utama. Jika gagal setelah 3 percobaan, program berhenti.

```
string nama, password;
// Login system
while (loginAttempts < 3) {
    cout << "===== SELAMAT DATANG! SILAHKAN LOGIN TERLEBIH DAHULU =====\n";
    cout << "Masukkan Nama: ";
    cin.ignore();
    getline(cin, nama);
    cout << "Masukkan Password: ";
    cin >> password;

    if (nama == "Danendra Hazzel PW" && password == "2409106096") {
        cout << "Login berhasil!\n";
        cout << "Selamat Datang Diaplikasi Pemesanan Tiket Online Tuan " << nama << endl;
        break;
    } else {
        loginAttempts++;
        cout << "Login gagal, Username atau Password salah! Sisa percobaan: " << 3 - loginAttempts << endl;
    }
}

if (loginAttempts == 3) {
    cout << "Kesempatan anda telah habis, silahkan mencoba dalam beberapa saat lagi!\n";
    return 0;
}
```

B. Pesan Tiket

Pada bagian ini program berfungsi untuk memesan tiket yang ingin dipesan oleh user. Program akan menampilkan film, jam tayang dan kursi yang tersedia saat ini untuk dipesan.

```
switch (pilihan) {
    case 1: {
        // Pesan tiket
        cout << "\n===== PESAN TIKET =====\n";

        // Tampilkan film yang tersedia
        cout << "Film yang tersedia:\n";
        for (int i = 0; i < 5; i++) {
            cout << i+1 << ". " << films[i] << endl;
        }

        int filmPilihan;
        cout << "Pilih film (1-5): ";
        cin >> filmPilihan;

        if (filmPilihan < 1 || filmPilihan > 5) {
            cout << "Film tidak valid!\n";
            break;
        }

        // Tampilkan jam tayang
        cout << "Jam tayang yang tersedia:\n";
        for (int i = 0; i < 5; i++) {
            int kursiTersisa = maksKursi - kursiTerisi[filmPilihan-1][i];
            cout << i+1 << ". " << jamTayang[i] << " (Kursi tersisa: " << kursiTersisa << ")\n";
        }

        int jamPilihan;
        cout << "Pilih jam tayang (1-5): ";
        cin >> jamPilihan;

        if (jamPilihan < 1 || jamPilihan > 5) {
            cout << "Jam tayang tidak valid!\n";
            break;
        }

        // Cek apakah kursi masih tersedia
        if (kursiTerisi[filmPilihan-1][jamPilihan-1] >= maksKursi) {
            cout << "Maaf, kursi untuk film dan jam tersebut sudah penuh!\n";
            break;
        }

        // Simpan booking
        bookings[userId][0] = filmPilihan - 1;
        bookings[userId][1] = jamPilihan - 1;
        bookings[userId][2] = 0; // Belum bayar

        // Update kursi terisi
        kursiTerisi[filmPilihan-1][jamPilihan-1]++;

        cout << "Booking berhasil!\n";
        cout << "ID Booking Anda: " << userId << endl;
        cout << "Film: " << films[filmPilihan-1] << endl;
        cout << "Jam: " << jamTayang[jamPilihan-1] << endl;
        cout << "Status: Belum dibayar!\n";
    }
}
```

C. Lihat film yang tersedia

Pada bagian ini, program akan menampilkan semua film yang tersedia serta jam tayang dan kursi yang masih dapat dipesan/dibooking.

```
case 2: {
    // Lihat film yang tersedia
    cout << "\n===== FILM YANG TERSEDIA =====\n";
    for (int i = 0; i < 5; i++) {
        cout << i+1 << ". " << films[i] << endl;
        cout << "    Jam Tayang: ";
        for (int j = 0; j < 5; j++) {
            int kursiTersisa = maksKursi - kursiTerisi[i][j];
            cout << jamTayang[j] << " (" << kursiTersisa << " kursi) ";
        }
        cout << endl;
    }
    break;
}
```

D. Lihat tiket yang sudah dipesan

Pada bagian ini, program akan menampilkan tiket yang sudah anda pesan dengan informasi bahwa tiket tersebut sudah/belum dibayar. Jika anda tidak/belum memesan tiket, program akan menampilkan pemberitahuan bahwa anda tidak ada tiket yang sedang dipesan.

```
case 3: {
    // Lihat tiket yang sudah dipesan
    cout << "\n==== TIKET YANG SUDAH DIPESAN =====\n";
    bool adaTiket = false;

    for (int i = 0; i < userId; i++) {
        if (bookings[i][0] != -1) {
            adaTiket = true;
            cout << "ID Booking: " << i << endl;
            cout << "Film: " << films[bookings[i][0]] << endl;
            cout << "Jam: " << jamTayang[bookings[i][1]] << endl;
            cout << "Status: " << (bookings[i][2] == 1 ? "Sudah dibayar" : "Belum dibayar") << endl;
            cout << "-----\n";
        }
    }

    if (!adaTiket) {
        cout << "Belum ada tiket yang dipesan!\n";
    }
    break;
}
```

E. Ganti jam tayang

Pada bagian ini, program akan menampilkan tiket yang telah anda pesan dan anda dapat mengganti jam tayang yang anda inginkan jika tiket tersebut belum dibayar.

```
case 4: {
    // Ganti jam tayang
    cout << "\n==== GANTI JAM TAYANG =====\n";

    int idBooking;
    cout << "Masukkan ID Booking: ";
    cin >> idBooking;

    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {
        cout << "ID Booking tidak valid!\n";
        break;
    }

    if (bookings[idBooking][2] == 1) {
        cout << "Tidak dapat mengubah jam tayang tiket yang sudah dibayar!\n";
        break;
    }

    int filmId = bookings[idBooking][0];
    int jamLama = bookings[idBooking][1];

    // Tampilkan jam tayang yang tersedia
    cout << "Jam tayang saat ini: " << jamTayang[jamLama] << endl;
    cout << "Jam tayang baru yang tersedia:\n";

    for (int i = 0; i < 5; i++) {
        int kursiTersisa = maksKursi - kursiTerisi[filmId][i];
        cout << i+1 << ". " << jamTayang[i] << " (Kursi tersisa: " << kursiTersisa << ")\n";
    }

    int jamBaru;
    cout << "Pilih jam tayang baru (1-5): ";
    cin >> jamBaru;

    if (jamBaru < 1 || jamBaru > 5) {
        cout << "Jam tayang tidak valid!\n";
        break;
    }

    if (kursiTerisi[filmId][jamBaru-1] >= maksKursi) {
        cout << "Maaf, kursi untuk jam tersebut sudah penuh!\n";
        break;
    }

    // Update kursi terisi
    kursiTerisi[filmId][jamLama]--;
    kursiTerisi[filmId][jamBaru-1]++;

    // Update booking
    bookings[idBooking][1] = jamBaru - 1;

    cout << "Jam tayang berhasil diubah!\n";
    cout << "Film: " << films[filmId] << endl;
    cout << "Jam tayang baru: " << jamTayang[jamBaru-1] << endl;
    break;
}
```


F. Batalkan Tiket

Pada bagian ini, program akan menampilkan ID tiket dan user dapat memabatalkan tiket mana yang ingin dibatalkan selama tiket tersebut belum dibayar.

```
case 5: {  
    // Batalkan tiket  
    cout << "\n===== BATALKAN TIKET =====\n";  
  
    int idBooking;  
    cout << "Masukkan ID Booking: ";  
    cin >> idBooking;  
  
    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {  
        cout << "ID Booking tidak valid!\n";  
        break;  
    }  
  
    if (bookings[idBooking][2] == 1) {  
        cout << "Tidak dapat membatalkan tiket yang sudah dibayar!\n";  
        break;  
    }  
  
    int filmId = bookings[idBooking][0];  
    int jamId = bookings[idBooking][1];  
  
    // Update kursi terisi  
    kursiTerisi[filmId][jamId]--;  
  
    // Hapus booking  
    bookings[idBooking][0] = -1;  
    bookings[idBooking][1] = -1;  
    bookings[idBooking][2] = -1;  
  
    cout << "Tiket berhasil dibatalkan!\n";  
    break;  
}
```

G. Bayar Tiket

Pada bagian ini, program akan menampilkan menu pembayaran dan user dapat membayar tiket yang ingin dibayar.

```
case 6: {  
    // Bayar tiket  
    cout << "\n===== BAYAR TIKET =====\n";  
  
    int idBooking;  
    cout << "Masukkan ID Booking: ";  
    cin >> idBooking;  
  
    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {  
        cout << "ID Booking tidak valid!\n";  
        break;  
    }  
  
    if (bookings[idBooking][2] == 1) {  
        cout << "Tiket sudah dibayar sebelumnya!\n";  
        break;  
    }  
  
    // Update status pembayaran  
    bookings[idBooking][2] = 1;  
  
    cout << "Pembayaran berhasil!\n";  
    cout << "Film: " << films[bookings[idBooking][0]] << endl;  
    cout << "Jam: " << jamTayang[bookings[idBooking][1]] << endl;  
    cout << "Status: Sudah dibayar!\n";  
    break;  
}
```

3. Source Code

1. Login

Program akan dimulai dengan menu login. Masukkan username dan password yang berisi “Danendra Hazzel PW” dan “2409106096”. Jika login berhasil, user akan diarahkan ke menu utama program. Tetapi Jika user salah menginput sebanyak 3x maka program akan berhenti.

2. Pesan Tiket

Pada fungsi ini digunakan untuk membeli ataupun memesan sebuah tiket yang diinginkan user. Program akan menampilkan film, jam tayang, serta kursi yang tersedia pada user untuk memilih. Jika sudah, maka program akan menyimpan data pesanan tersebut dengan informasi seperti belum/sudah membayar tiket tersebut.

3. Lihat Film yang Tersedia

Pada fungsi ini, program akan menampilkan semua informasi mengenai film yang sedang tayang dari jam tayang dan jumlah kursi yang masih tersedia kepada user.

4. Lihat Tiket yang telah dipesan

Pada fungsi ini, program akan menampilkan semua tiket yang telah user pesan lengkap dengan informasi seperti sudah/belumnya membayar dan jam tayang yang dipilih oleh user. Jika user belum melakukan pemesanan, program akan menampilkan bahwa user belum melakukan pemesanan tiket apapun.

5. Ganti Jam Tayang

Pada fungsi ini, user dapat mengganti jam tayang dari tiket yang ia pesan selama ia belum membayar tiket tersebut. Jika sudah membayar, user tidak akan bisa mengganti jam tayangnya dan program akan menampilkan bahwa user tidak dapat mengganti jam tayang tiket yang telah dibayar.

6. Batalkan Tiket

Pada fungsi ini, user dapat melakukan pembatalan tiket yang telah dipesan oleh user. Jika tiket sudah dibayar, maka user tidak dapat membatalkan tiket tersebut.

7. Bayar Tiket

Pada fungsi ini, user dapat melakukan pembayaran tiket yang telah ia pesan. Tetapi jika user belum memiliki ID tiket yang valid, maka program tidak dapat membaca tiket yang ingin dibayar oleh user.

```

int main() {
    // Film data
    string Films[5] = {"Kimi No Nawa", "Godzilla : Minus one", "Avengers : secret wars", "Your Lie In April", "Cek Toko Sebelah"};

    // Jam tayang: 10:00, 12:00, 15:00, 17:00, 19:00
    string JamTayang[5] = {"10:00", "12:00", "15:00", "17:00", "19:00"};

    // Array untuk menyimpan booking
    // [user_id][0] = film_id, [user_id][1] = jam_id, [user_id][2] = status_bayar (0=belum, 1=sudah)
    int bookings[10][3];

    // Array untuk tracking apakah kursi sudah diboeking
    // [film_id][jam_id] = jumlah kursi terisi
    int kursiTerisi[5][5] = {0};

    // Kapasitas maksimum per film dan jam
    int maksKursi = 20;

    // Inisialisasi bookings dengan -1 (tidak ada booking)
    for (int i = 0; i < 10; i++) {
        for (int j = 0; j < 3; j++) {
            bookings[i][j] = -1;
        }
    }

    int userId = 0; // User ID counter
    int pilihan = 0;
    bool isRunning = true;
    int loginAttempts = 0;
    string nama, password;

    // Login system
    while (loginAttempts < 3) {
        cout << "==== LOGIN SISTEM BOOKING TIKET BIOSKOP =====\n";
        cout << "Masukkan Nama: ";
        cin.ignore(); // Membersihkan buffer input sebelum menggunakan getline
        getline(cin, nama); // Menggunakan getline untuk membaca nama dengan spasi
        cout << "Masukkan Password: ";
        cin >> password;

        if (nama == "Danendra Hazzel PW" && password == "2409106096") {
            cout << "Login berhasil!\n";
            break;
        } else {
            loginAttempts++;
            cout << "Login gagal! Sisa percobaan: " << 3 - loginAttempts << endl;
        }
    }

    if (loginAttempts == 3) {
        cout << "Anda telah mencoba login sebanyak 3 kali. Program berhenti.\n";
        return 0;
    }
}

```

```

switch (pilihan) {
    case 1: {
        // Pesan tiket
        cout << "\n===== PESAN TIKET =====\n";

        // Tampilkan film yang tersedia
        cout << "Film yang tersedia:\n";
        for (int i = 0; i < 5; i++) {
            cout << i+1 << ". " << films[i] << endl;
        }

        int filmPilihan;
        cout << "Pilih film (1-5): ";
        cin >> filmPilihan;

        if (filmPilihan < 1 || filmPilihan > 5) {
            cout << "Film tidak valid!\n";
            break;
        }

        // Tampilkan jam tayang
        cout << "Jam tayang yang tersedia:\n";
        for (int i = 0; i < 5; i++) {
            int kursiTersisa = maksKursi - kursiTerisi[filmPilihan-1][i];
            cout << i+1 << ". " << jamTayang[i] << " (Kursi tersisa: " << kursiTersisa << ")\n";
        }

        int jamPilihan;
        cout << "Pilih jam tayang (1-5): ";
        cin >> jamPilihan;

        if (jamPilihan < 1 || jamPilihan > 5) {
            cout << "Jam tayang tidak valid!\n";
            break;
        }

        // Cek apakah kursi masih tersedia
        if (kursiTerisi[filmPilihan-1][jamPilihan-1] >= maksKursi) {
            cout << "Maaf, kursi untuk film dan jam tersebut sudah penuh!\n";
            break;
        }

        // Simpan booking
        bookings[userId][0] = filmPilihan - 1;
        bookings[userId][1] = jamPilihan - 1;
        bookings[userId][2] = 0; // Belum bayar

        // Update kursi terisi
        kursiTerisi[filmPilihan-1][jamPilihan-1]++;

        cout << "Booking berhasil!\n";
        cout << "ID Booking Anda: " << userId << endl;
        cout << "Film: " << films[filmPilihan-1] << endl;
        cout << "Jam: " << jamTayang[jamPilihan-1] << endl;
        cout << "Status: Belum dibayar\n";

        userId++;
        break;
    }
    case 2: {
        // Lihat film yang tersedia
        cout << "\n===== FILM YANG TERSEDIA =====\n";
        for (int i = 0; i < 5; i++) {
            cout << i+1 << ". " << films[i] << endl;
            cout << "    Jam Tayang: ";
            for (int j = 0; j < 5; j++) {
                int kursiTersisa = maksKursi - kursiTerisi[i][j];
                cout << jamTayang[j] << " (" << kursiTersisa << " kursi) ";
            }
            cout << endl;
        }
        break;
    }
}

```

```

case 3: {
    // Lihat tiket yang sudah dipesan
    cout << "\n===== TIKET YANG SUDAH DIPESAN =====\n";
    bool adaTiket = false;

    for (int i = 0; i < userId; i++) {
        if (bookings[i][0] != -1) {
            adaTiket = true;
            cout << "ID Booking: " << i << endl;
            cout << "Film: " << films[bookings[i][0]] << endl;
            cout << "Jam: " << jamTayang[bookings[i][1]] << endl;
            cout << "Status: " << (bookings[i][2] == 1 ? "Sudah dibayar" : "Belum dibayar") << endl;
            cout << "-----\n";
        }
    }

    if (!adaTiket) {
        cout << "Belum ada tiket yang dipesan!\n";
    }
    break;
}

case 4: {
    // Ganti jam tayang
    cout << "\n===== GANTI JAM TAYANG =====\n";

    int idBooking;
    cout << "Masukkan ID Booking: ";
    cin >> idBooking;

    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {
        cout << "ID Booking tidak valid!\n";
        break;
    }

    if (bookings[idBooking][2] == 1) {
        cout << "Tidak dapat mengubah jam tayang tiket yang sudah dibayar!\n";
        break;
    }

    int filmId = bookings[idBooking][0];
    int jamLama = bookings[idBooking][1];

    // Tampilkan jam tayang yang tersedia
    cout << "Jam tayang saat ini: " << jamTayang[jamLama] << endl;
    cout << "Jam tayang baru yang tersedia:\n";

    for (int i = 0; i < 5; i++) {
        int kursiTersisa = maksKursi - kursiTerisi[filmId][i];
        cout << i+1 << ". " << jamTayang[i] << " (Kursi tersisa: " << kursiTersisa << ")\n";
    }

    int jamBaru;
    cout << "Pilih jam tayang baru (1-5): ";
    cin >> jamBaru;

    if (jamBaru < 1 || jamBaru > 5) {
        cout << "Jam tayang tidak valid!\n";
        break;
    }

    if (kursiTerisi[filmId][jamBaru-1] >= maksKursi) {
        cout << "Maaf, kursi untuk jam tersebut sudah penuh!\n";
        break;
    }

    // Update kursi terisi
    kursiTerisi[filmId][jamLama]--;
    kursiTerisi[filmId][jamBaru-1]++;

    // Update booking
    bookings[idBooking][1] = jamBaru - 1;

    cout << "Jam tayang berhasil diubah!\n";
    cout << "Film: " << films[filmId] << endl;
    cout << "Jam tayang baru: " << jamTayang[jamBaru-1] << endl;
    break;
}
}

```

```

case 5: {
    // Batalkan tiket
    cout << "\n===== BATALKAN TIKET =====\n";

    int idBooking;
    cout << "Masukkan ID Booking: ";
    cin >> idBooking;

    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {
        cout << "ID Booking tidak valid!\n";
        break;
    }

    if (bookings[idBooking][2] == 1) {
        cout << "Tidak dapat membatalkan tiket yang sudah dibayar!\n";
        break;
    }

    int filmId = bookings[idBooking][0];
    int jamId = bookings[idBooking][1];

    // Update kursi terisi
    kursiTerisi[filmId][jamId]--;

    // Hapus booking
    bookings[idBooking][0] = -1;
    bookings[idBooking][1] = -1;
    bookings[idBooking][2] = -1;

    cout << "Tiket berhasil dibatalkan!\n";
    break;
}
case 6: {
    // Bayar tiket
    cout << "\n===== BAYAR TIKET =====\n";

    int idBooking;
    cout << "Masukkan ID Booking: ";
    cin >> idBooking;

    if (idBooking < 0 || idBooking >= userId || bookings[idBooking][0] == -1) {
        cout << "ID Booking tidak valid!\n";
        break;
    }

    if (bookings[idBooking][2] == 1) {
        cout << "Tiket sudah dibayar sebelumnya!\n";
        break;
    }

    // Update status pembayaran
    bookings[idBooking][2] = 1;

    cout << "Pembayaran berhasil!\n";
    cout << "Film: " << films[bookings[idBooking][0]] << endl;
    cout << "Jam: " << jamTayang[bookings[idBooking][1]] << endl;
    cout << "Status: Sudah dibayar!\n";
    break;
}
case 7: {
    // Keluar dari program
    cout << "Terima kasih telah menggunakan aplikasi ini, enjoy the movie!\n";
    isRunning = false;
    break;
}
default: {
    cout << "Pilihan tidak valid!\n";
    break;
}
}
}

return 0;
}

```

4. Uji Coba dan Hasil Output

4.1 Uji Coba

1. Skenario 1: User ingin login untuk menjalankan program, namun user lupa username dan passwordnya. User berusaha menginput sebanyak 3x namun gagal. Maka dari itu program tidak mendeteksi user dan program tidak jalan.
2. Skenario 2: User ingin login dan memasukkan username dan password yang telah ia buat. Program berhasil mendeteksi user dan menampilkan menu utama.
3. Skenario 3 : User Membooking tiket bioskop yang ia inginkan
4. Skenario 4 : User mengganti jam tayang pada film yang ia ingin ubah
5. Skenario 5 : User membatalkan salah satu tiket yang ia booking.
6. Skenario 6 : User membayar tiket yang ia pesan
7. Skenario 7 : User ingin mengganti jam tayang tiket yang sudah dia bayar.
8. Skenario 8 : User salah menginput index pada menu

4.2 Hasil Output

```
PS C:\GITHUB\Praktikum-APL\Post-Test> cd "c:\GITHUB\Praktikum
6096-DanendraHazzelPW-PT-2 }
===== LOGIN SISTEM BOOKING TIKET BIOSKOP =====
Masukkan Nama: Danendra Hazzel PW
Masukkan Password: 240
Login gagal! Sisa percobaan: 2
===== LOGIN SISTEM BOOKING TIKET BIOSKOP =====
Masukkan Nama: DHPW
Masukkan Password: 240
Login gagal! Sisa percobaan: 1
===== LOGIN SISTEM BOOKING TIKET BIOSKOP =====
Masukkan Nama: DEHAZE
Masukkan Password: 240
Login gagal! Sisa percobaan: 0
Anda telah mencoba login sebanyak 3 kali. Program berhenti.
PS C:\GITHUB\Praktikum-APL\Post-Test\Post-Test-2> |
```

Gambar 4.1 Contoh 1

```
===== LOGIN SISTEM BOOKING TIKET BIOSKOP =====
Masukkan Nama: Danendra Hazzel PW
Masukkan Password: 2409106096
Login berhasil!

===== SISTEM BOOKING TIKET BIOSKOP =====
1. Pesan Tiket
2. Lihat Film yang Tersedia
3. Lihat Tiket yang Sudah Dipesan
4. Ganti Jam Tayang
5. Batalkan Tiket
6. Bayar Tiket
7. Keluar
Pilihan Anda: |
```

Gambar 4.2 Contoh 2

```

===== SISTEM BOOKING TIKET BIOSKOP =====
1. Pesan Tiket
2. Lihat Film yang Tersedia
3. Lihat Tiket yang Sudah Dipesan
4. Ganti Jam Tayang
5. Batalkan Tiket
6. Bayar Tiket
7. Keluar
Pilihan Anda: 1

===== PESAN TIKET =====
Film yang tersedia:
1. Kimi No Nawa
2. Godzilla : Minus one
3. Avengers : secret wars
4. Your Lie In April
5. Cek Toko Sebelah
Pilih film (1-5): 4
Jam tayang yang tersedia:
1. 10:00 (Kursi tersisa: 20)
2. 12:00 (Kursi tersisa: 20)
3. 15:00 (Kursi tersisa: 20)
4. 17:00 (Kursi tersisa: 20)
5. 19:00 (Kursi tersisa: 20)
Pilih jam tayang (1-5): 3
Booking berhasil!
ID Booking Anda: 0
Film: Your Lie In April
Jam: 15:00
Status: Belum dibayar

```

Gambar 4.3 Contoh 3

```

===== SISTEM BOOKING TIKET BIOSKOP =====
1. Pesan Tiket
2. Lihat Film yang Tersedia
3. Lihat Tiket yang Sudah Dipesan
4. Ganti Jam Tayang
5. Batalkan Tiket
6. Bayar Tiket
7. Keluar
Pilihan Anda: 4

===== GANTI JAM TAYANG =====
Masukkan ID Booking: 0
Jam tayang saat ini: 15:00
Jam tayang baru yang tersedia:
1. 10:00 (Kursi tersisa: 20)
2. 12:00 (Kursi tersisa: 20)
3. 15:00 (Kursi tersisa: 19)
4. 17:00 (Kursi tersisa: 20)
5. 19:00 (Kursi tersisa: 20)
Pilih jam tayang baru (1-5): 4
Jam tayang berhasil diubah!
Film: Your Lie In April
Jam tayang baru: 17:00

```

Gambar 4.4 Contoh 4


```
===== SISTEM BOOKING TIKET BIOSKOP =====  
1. Pesan Tiket  
2. Lihat Film yang Tersedia  
3. Lihat Tiket yang Sudah Dipesan  
4. Ganti Jam Tayang  
5. Batalkan Tiket  
6. Bayar Tiket  
7. Keluar  
Pilihan Anda: 5  
  
===== BATALKAN TIKET =====  
Masukkan ID Booking: 1  
Tiket berhasil dibatalkan!
```

Gambar 4.5 Contoh 5

```
===== SISTEM BOOKING TIKET BIOSKOP =====  
1. Pesan Tiket  
2. Lihat Film yang Tersedia  
3. Lihat Tiket yang Sudah Dipesan  
4. Ganti Jam Tayang  
5. Batalkan Tiket  
6. Bayar Tiket  
7. Keluar  
Pilihan Anda: 6  
  
===== BAYAR TIKET =====  
Masukkan ID Booking: 0  
Pembayaran berhasil!  
Film: Your Lie In April  
Jam: 17:00  
Status: Sudah dibayar
```

Gambar 4.6 Contoh 6

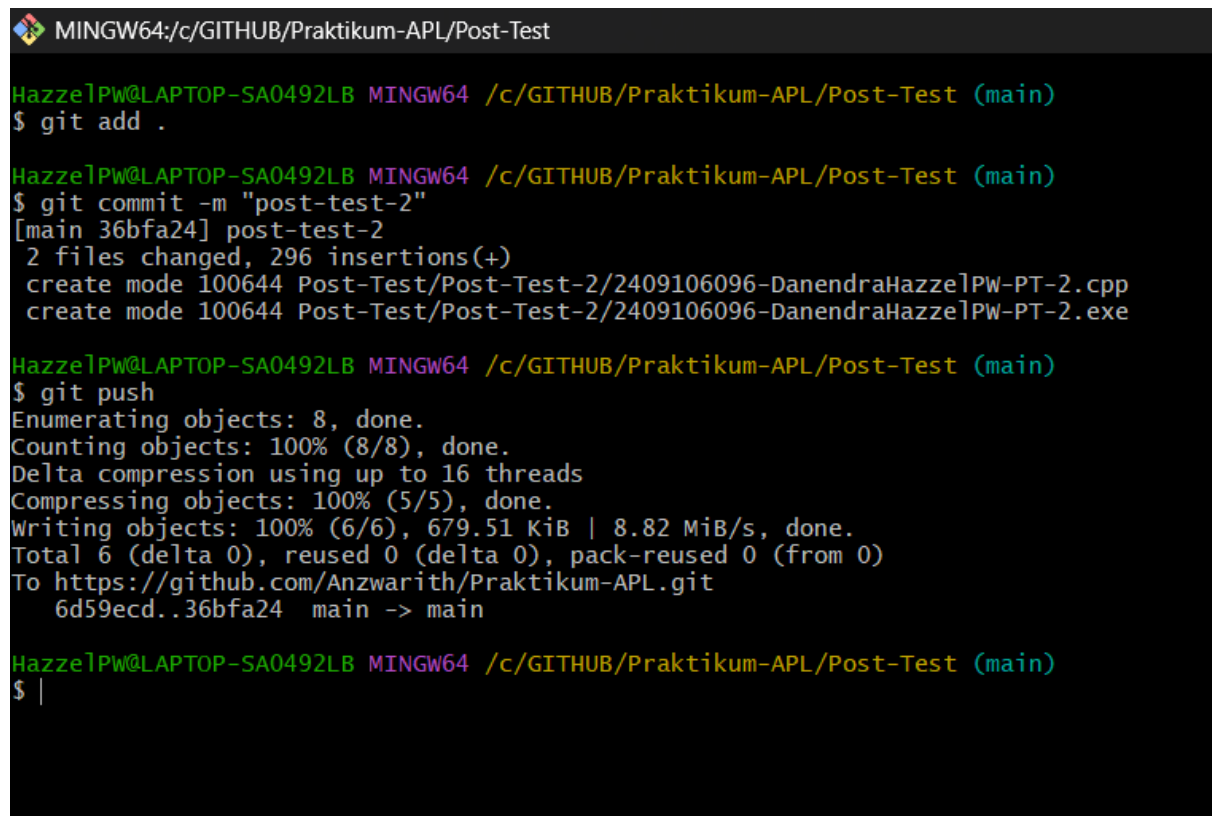
```
===== SISTEM BOOKING TIKET BIOSKOP =====  
1. Pesan Tiket  
2. Lihat Film yang Tersedia  
3. Lihat Tiket yang Sudah Dipesan  
4. Ganti Jam Tayang  
5. Batalkan Tiket  
6. Bayar Tiket  
7. Keluar  
Pilihan Anda: 4  
  
===== GANTI JAM TAYANG =====  
Masukkan ID Booking: 0  
Tidak dapat mengubah jam tayang tiket yang sudah dibayar!
```

Gambar 4.7 Contoh 7

```
===== SISTEM BOOKING TIKET BIOSKOP =====  
1. Pesan Tiket  
2. Lihat Film yang Tersedia  
3. Lihat Tiket yang Sudah Dipesan  
4. Ganti Jam Tayang  
5. Batalkan Tiket  
6. Bayar Tiket  
7. Keluar  
Pilihan Anda: 10  
Pilihan tidak valid!
```

Gambar 4.8 Contoh 8

5. Git



```
MINGW64:/c/GITHUB/Praktikum-APL/Post-Test

Hazze1PW@LAPTOP-SA0492LB MINGW64 /c/GITHUB/Praktikum-APL/Post-Test (main)
$ git add .

Hazze1PW@LAPTOP-SA0492LB MINGW64 /c/GITHUB/Praktikum-APL/Post-Test (main)
$ git commit -m "post-test-2"
[main 36bfa24] post-test-2
 2 files changed, 296 insertions(+)
 create mode 100644 Post-Test/Post-Test-2/2409106096-DanendraHazze1PW-PT-2.cpp
 create mode 100644 Post-Test/Post-Test-2/2409106096-DanendraHazze1PW-PT-2.exe

Hazze1PW@LAPTOP-SA0492LB MINGW64 /c/GITHUB/Praktikum-APL/Post-Test (main)
$ git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 679.51 KiB | 8.82 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Anzwarith/Praktikum-APL.git
   6d59ecd..36bfa24  main -> main

Hazze1PW@LAPTOP-SA0492LB MINGW64 /c/GITHUB/Praktikum-APL/Post-Test (main)
$ |
```

Gambar 5.1 Bukti pembuatan Git

Git Add berfungsi untuk memindahkan file kedalam staging area dan siap untuk melakukan langkah selanjutnya yaitu git commit. Git commit berfungsi untuk menaikkan file dari staging area ke dalam repositori pada github dengan commit message “post-test-2”. Setelah selesai, langkah selanjutnya adalah git push yang berfungsi untuk menaikkan file yang sudah di commit untuk ditampilkan ke dalam repositori github.