

Tugas Akhir

Pemrograman Berbasis Objek



Sistem Ticketing Bioskop

24040700110 Muhammad Alif Razaqi

Journal : Sistem Ticketing Bioskop

Sistem Pemesanan Tiket Bioskop Menggunakan Object-Oriented Programming (C++)

Proyek ini merupakan implementasi sistem pemesanan tiket bioskop berbasis C++ yang dirancang menggunakan prinsip Object-Oriented Programming (OOP). Sistem ini mensimulasikan proses pemesanan tiket, pengelolaan kursi, dan pencetakan struk transaksi. Dengan memanfaatkan konsep OOP seperti class, objek, enkapsulasi, dan komposisi, sistem ini memberikan pendekatan modular dan terstruktur untuk kebutuhan edukasi pemrograman.

Dalam pembelajaran pemrograman, siswa sering kesulitan memahami bagaimana konsep OOP diterapkan dalam kasus nyata. Proyek ini menjawab tantangan tersebut dengan menyajikan simulasi sistem bioskop yang:

- Mengelola daftar film dan jadwal tayang

- Menyediakan layout kursi yang dapat dipesan
- Mencetak struk transaksi secara otomatis
- Menunjukkan interaksi antar objek secara eksplisit

Industri hiburan, khususnya bioskop, membutuhkan sistem yang efisien untuk mengelola pemesanan tiket. Dalam praktik nyata, sistem ini biasanya berbasis aplikasi atau website. proyek ini dibuat sebagai simulasi sederhana menggunakan bahasa C++ agar dapat memahami penerapan **Object-Oriented Programming (OOP)** dalam kasus nyata.

Dengan pendekatan OOP, sistem ini:

- Memisahkan tanggung jawab ke dalam beberapa class.
- Menggunakan enkapsulasi untuk melindungi data.
- Menunjukkan interaksi antar objek (Film, Penonton, Bioskop).
- Memberikan simulasi nyata pemesanan kursi dan pencetakan struk.

Permasalahan yang Diangkat

Dalam pembelajaran pemrograman, kita sering kesulitan memahami bagaimana konsep OOP diterapkan dalam kasus nyata. Proyek ini menjawab tantangan tersebut dengan menyajikan simulasi sistem bioskop yang:

- Mengelola daftar film dan jadwal tayang
- Menyediakan layout kursi yang dapat dipesan
- Mencetak struk transaksi secara otomatis
- Menunjukkan interaksi antar objek secara eksplisit

Solusi Dan Tujuan yang Diberikan

- Desain modular dengan pemisahan tanggung jawab antar class
- Enkapsulasi data kursi, harga, dan jadwal film
- Validasi pemesanan kursi untuk mencegah double-booking
- Membuat simulasi sistem pemesanan tiket bioskop menggunakan C++.
- Visualisasi layout kursi secara real-time
- Fondasi untuk pengembangan fitur

interaktif dan penyimpanan data

- Menampilkan layout kursi secara visual agar mudah dipahami.
- Mencetak struk transaksi sebagai bukti pemesanan.

Sistem Pemesanan Tiket Bioskop

1. Komponen Utama Sistem

Sistem ini dibangun dengan **tiga class inti** yang saling berinteraksi:

Class Film

- **Peran:** Merepresentasikan sebuah film beserta kursi yang tersedia.
- **Atribut:**
 - judul → nama film.
 - jam → jam tayang.
 - harga → harga tiket.
 - kursi → matriks 2D (vector<vector<bool>>) untuk status kursi.
- **Fungsi:**
 - tampilkanKursi() → menampilkan layout kursi (O = kosong, X = terisi).

- pesan(r,c) → memesan kursi tertentu jika masih kosong.

- **Atribut:**

- daftar → kumpulan film (vector<Film>).

Class Penonton

- **Peran:** Merepresentasikan pelanggan/penonton yang membeli tiket.
- **Atribut:**
 - nama → nama penonton.
- **Fungsi:**
 - getNama() → mengembalikan nama penonton.

- **Fungsi:**

- tambahFilm() → menambahkan film ke daftar.
- tampilkanFilm() → menampilkan semua film beserta jam tayang dan harga.
- pesanTiket() → memproses pemesanan kursi, mencetak struk transaksi.

Class Bioskop

- **Peran:** Bertindak sebagai **controller utama** yang mengelola daftar film dan transaksi.

- tampilkanLayout() → menampilkan layout kursi film tertentu.

2. Hubungan Antar Class

Hubungan	Deskripsi
Bioskop → Film	Bioskop memiliki banyak film (komposisi).
Bioskop → Penonton	Bioskop menerima data penonton saat pemesanan tiket.
Film → Kursi	Film mengelola status kursi (kosong/terisi).
Penonton → Film	Penonton memesan kursi pada film tertentu.

- Sistem menampilkan kondisi kursi setelah pemesanan (O = kosong, X = terisi).

3. Alur Logika Sistem

1. Inisialisasi Sistem

- Buat objek Bioskop.
- Tambahkan beberapa film ke daftar bioskop.

2. Tampilkan Daftar Film

- Sistem menampilkan semua film dengan judul, jam tayang, dan harga.

3. Pemesanan Tiket

- Penonton memilih film dan kursi.
- Sistem memvalidasi apakah kursi tersedia.
- Jika tersedia → kursi ditandai terisi, struk dicetak.
- Jika tidak tersedia → pesan error ditampilkan.

4. Cetak Struk

- Struk berisi: nama penonton, judul film, jam tayang, kursi, harga, dan tanggal transaksi.

5. Tampilkan Layout Kursi

4. Visualisasi Sistem

Bayangkan sistem ini seperti **alur kerja nyata di bioskop**:

- **Bioskop** = manajer utama
- **Film** = daftar tayangan + kursi
- **Penonton** = pelanggan yang memesan
- **Interaksi:** Penonton → Bioskop → Film → Kursi → Struk

5. Keunggulan Sistem

- **Modular:** setiap class punya tanggung jawab jelas.
- **Enkapsulasi:** detail kursi dan harga disembunyikan dari luar.
- **Interaktif:** menampilkan layout kursi secara visual.
- **Realistik:** mencetak struk dengan timestamp.
- **Scalable:** mudah ditambah fitur (misalnya pembatalan tiket, penyimpanan data).

Contoh Implementasi

```
#include <iostream>
#include <vector>
#include <string>
#include <ctime>
using namespace std;

// -----
// Class Film
// -----
class Film {
    string judul, jam;
    int harga;
    vector<vector<bool>> kursi;
public:
    Film(string j, string t, int h, int r, int c) {
        judul = j; jam = t; harga = h;
        kursi = vector<vector<bool>>(r,
vector<bool>(c, false));
    }
    string getJudul(){ return judul; }
    string getJam(){ return jam; }
    int getHarga(){ return harga; }

    void tampilkanKursi() {
        cout << "\nLayout kursi " << judul << "
(" << jam << ")\n";
        for(int i=0;i<kursi.size();i++){
            for(int j=0;j<kursi[i].size();j++)
                cout << (kursi[i][j] ? "X " : "O ");
        }
    }
};

bool pesan(int r,int c){
    if(r>=0 && r<kursi.size() && c>=0 &&
c<kursi[0].size() && !kursi[r][c]){
        kursi[r][c]=true; return true;
    }
    return false;
};

// -----
// Class Penonton
// -----
class Penonton {
    string nama;
public:
    Penonton(string n){ nama=n; }
    string getNama(){ return nama; }
};

// -----
// Class Bioskop
// -----
class Bioskop {
    vector<Film> daftar;
public:
    void tambahFilm(Film f){
        daftar.push_back(f);
    }
};
```

```

void tampilkanFilm(){

    cout << "\n==== DAFTAR FILM ====\n";
    // -----
    // Main Program
    // -----

    for(int i=0;i<daftar.size();i++)
        cout << i+1 << ". " <<
daftar[i].getJudul()
        << " (" << daftar[i].getJam() << ")"
Rp" << daftar[i].getHarga() << endl;

}

void pesanTiket(int idx, Penonton p, int
r, int c){

    if(idx>=0 && idx<daftar.size() &&
daftar[idx].pesan(r,c)){

        time_t now=time(NULL);

        cout << "\n--- STRUK ---\n";
        cout << "Penonton : " <<
p.getNama() << endl;

        cout << "Film : " <<
daftar[idx].getJudul() << "(" <<
daftar[idx].getJam() << ")\n";

        cout << "Kursi : (" << r << "," << c
<< ")\n";

        cout << "Bayar : Rp" <<
daftar[idx].getHarga() << endl;

        cout << "Tanggal : " <<
ctime(&now);

    } else cout << "Kursi tidak
tersedia!\n";
}
}

void tampilkanLayout(int idx){

    if(idx>=0 && idx<daftar.size())
daftar[idx].tampilkanKursi();

}
};

int main(){

    Bioskop b;

    b.tambahFilm(Film("Avengers","19:00",50
000,5,6));

    b.tambahFilm(Film("Spiderman","21:00",4
5000,5,6));

    b.tampilkanFilm();

    Penonton p1("Adi");

    b.pesanTiket(0,p1,2,3);

    b.tampilkanLayout(0);

    return 0;
}

```

Contoh Output

```
==== DAFTAR FILM ====
1. Avengers (19:00) Rp50000
2. Spiderman (21:00) Rp45000

--- STRUK ---
Penonton : Adi
Film      : Avengers (19:00)
Kursi     : (2,3)
Bayar     : Rp50000
Tanggal   : Sat Jan 03 01:10:16 2020

Layout kursi Avengers (19:00)
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 X 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

Tantangan Teknis

- **Manajemen Kursi:**
Representasi kursi sebagai matriks boolean.
- **Validasi Input:** Menangani kursi yang sudah terisi.
- **Enkapsulasi:**
Menyembunyikan detail internal kursi dan harga.
- **Struktur Modular:**
Memisahkan logika film, penonton, dan bioskop.
- **Timestamp:** Menggunakan ctime() untuk mencetak waktu transaksi.

Struktur Data yang Digunakan

1. vector (Standard Template Library - STL)

Digunakan sebagai **array dinamis** untuk menyimpan daftar film dan layout kursi.

Penggunaan:

- vector<Film> → menyimpan daftar film dalam class Bioskop.
- vector<vector<bool>> → menyimpan status kursi (true = terisi, false = kosong) dalam class Film.

Keunggulan:

- Ukurannya fleksibel (bisa bertambah/dikurangi).
- Mendukung akses indeks seperti array.
- Cocok untuk layout kursi 2D dan daftar film.

2. string

Digunakan untuk menyimpan data teks seperti nama penonton, judul film, jam tayang, dan tanggal transaksi.

Penggunaan:

- string nama → nama penonton.
- string judul, jam → informasi film.

- string tanggal → waktu transaksi (dihasilkan dari ctime()).

Keunggulan:

- Mudah digunakan dan dimanipulasi.
- Mendukung operasi seperti concatenation, comparison, dan input/output.

3. bool

Digunakan untuk menandai status kursi dalam layout bioskop.

Penggunaan:

- kursi[i][j] = true → kursi sudah dipesan.
- kursi[i][j] = false → kursi masih kosong.

Keunggulan:

- Ringan dan efisien untuk status biner.
- Cocok untuk representasi layout kursi.

4. time_t dan ctime()

Digunakan untuk mencetak waktu transaksi saat pemesanan tiket.

Penggunaan:

cpp

```
time_t now = time(NULL);
```

```
cout << "Tanggal :" <<
ctime(&now);
```

Keunggulan:

- Memberikan timestamp real-time.
- Format output mudah dibaca.

5. int

Digunakan untuk menyimpan nilai numerik seperti harga tiket dan indeks kursi.

Penggunaan:

- harga → harga tiket film.
- r, c → baris dan kolom kursi.

Interaksi Antar Struktur

- vector<Film> diakses oleh Bioskop untuk menampilkan dan memesan tiket.
- vector<vector<bool>> diakses oleh Film untuk memvalidasi dan menandai kursi.
- string digunakan di semua class untuk identitas dan informasi.
- time_t digunakan saat mencetak struk transaksi.

Contoh Representasi Kursi

```
vector<vector<bool>> kursi(5,  
vector<bool>(6, false));  
  
= layout kursi 5 baris × 6 kolom,  
semua kursi awalnya kosong.
```

tetap menyediakan
akses lewat metode
publik.

4. Struktur Modular

- Memisahkan logika ke dalam class Film, Penonton, dan Bioskop.
- Tantangan: mendesain hubungan antar class agar tidak saling bergantung berlebihan.

5. Penggunaan Waktu (Timestamp)

- Menggunakan ctime() untuk mencetak tanggal transaksi.
- Tantangan: format waktu bawaan C++ kadang kurang rapi, perlu dipahami cara manipulasi string waktu.

Tantangannya

1. Manajemen Kursi

- Representasi kursi menggunakan vector<vector<bool>> menuntut pemahaman matriks 2D.
- Tantangan: memastikan indeks valid dan mencegah *out of range* error.

2. Validasi Pemesanan

- Harus dicek apakah kursi sudah terisi sebelum memesan.
- Tantangan: mencegah *double-booking* dan menampilkan pesan error yang jelas.

3. Enkapsulasi Data

- Menyembunyikan detail internal (kursi, harga) agar tidak langsung diakses dari luar.
- Tantangan: menjaga konsistensi data sambil

Pembelajarannya

- **Penerapan OOP Nyata**
Siswa belajar bagaimana konsep class, objek, dan enkapsulasi digunakan dalam kasus dunia nyata.
- **Penggunaan STL (vector, string)** Melatih pemahaman struktur data dinamis dan manipulasi teks.

- **Interaksi Antar Objek**
Menunjukkan bagaimana Penonton berinteraksi dengan Film melalui Bioskop.
- **Validasi dan Error**
Handling Pentingnya memeriksa input agar sistem tidak crash dan tetap ramah pengguna.
- **Modularitas dan Skalabilitas** Dengan desain modular, sistem mudah dikembangkan (misalnya menambah fitur pembatalan tiket, penyimpanan data, atau GUI).
- **Nilai Edukatif** Proyek ini memperkuat konsep fundamental C++:
 - Desain dan instansiasi class
 - Penggunaan vector 2D
 - Enkapsulasi dan komposisi
 - Simulasi alur bisnis sederhana

Kesimpulan dan Refleksi Edukatif

Proyek Sistem Pemesanan Tiket Bioskop berhasil mendemonstrasikan penerapan prinsip Object-Oriented Programming (OOP) dalam bahasa C++. Dengan membagi sistem ke

dalam tiga class utama (Film, Penonton, dan Bioskop), program menjadi modular, terstruktur, dan mudah dipahami.

Sistem ini mampu:

- Mengelola daftar film beserta jadwal tayang dan harga tiket.
- Menyediakan layout kursi dengan status real-time (kosong/terisi).
- Memproses pemesanan tiket dengan validasi kursi.
- Mencetak struk transaksi lengkap dengan informasi penonton, film, kursi, harga, dan tanggal.

Nilai Edukatif

Proyek ini memberikan manfaat besar bagi pembelajaran siswa:

- Memperkuat konsep OOP: siswa belajar bagaimana class, objek, enkapsulasi, dan komposisi diterapkan dalam kasus nyata.
- Melatih penggunaan struktur data STL: seperti vector untuk array dinamis dan string untuk manipulasi teks.
- Meningkatkan keterampilan problem solving: melalui validasi input, manajemen kursi, dan pencetakan struk.

- Mendorong pemikiran modular: siswa memahami pentingnya pemisahan tanggung jawab antar class agar sistem mudah dikembangkan.
- Menyiapkan fondasi untuk proyek lebih kompleks: seperti integrasi file I/O, GUI, atau database.

Kesimpulan Akhir

Secara keseluruhan, sistem ini bukan hanya simulasi sederhana, tetapi juga alat pembelajaran yang efektif untuk memperkenalkan konsep OOP kepada siswa. Dengan desain yang modular dan fleksibel, sistem ini dapat dikembangkan lebih lanjut sesuai kebutuhan, sekaligus memberikan pengalaman nyata bagaimana pemrograman dapat menyelesaikan masalah sehari-hari.