# **PRAKTIKUM**

**PEMROGRAMAN BERORIENTASI OBJEK**

**LAPORAN PROYEK AKHIR**

**MOVIE REVIEW**

****

**DISUSUN OLEH :**

|  |  |  |
| --- | --- | --- |
| **ANGGOTA** | **:** | **SATYA GHIFARI A 123180051**  **LIEK ALLYANDARU 123180054** |
| **KELAS/PLUG** | **:** | **E** |
| **NAMA ASISTEN** | **:** | **MONICA AMEILIANA ANDISTA** |
|  |  | **AMIRA SALSABILA** |

### PROGRAM STUDI INFORMATIKA

### JURUSAN TEKNIK INFORMATIKA

### FAKULTAS TEKNIK INDUSTRI

### UNIVERSITAS PEMBANGUNAN NASIONAL ”VETERAN”

#### YOGYAKARTA

#### 2020

# **HALAMAN PENGESAHAN**

**LAPORAN PROYEK AKHIR**

****

Disusun oleh :

*Satya Ghifari Adipratama* 123180051

*Liek Allyandaru* 123180054

Telah Diperiksa dan Disetujui oleh Asisten Praktikum Implementasi Basis Data

Pada Tanggal : …………………

|  |  |
| --- | --- |
| **Menyetujui,**  **Asisten Praktikum**  **Monica Ameiliana Andista M**  **NIM. 124180041** | **Asisten Praktikum**  **Amira Salsabila**  **NIM. 124180067** |

|  |
| --- |
| **Mengetahui,**  **Ka. Lab. Geoinformatka**  **Budi Santosa, S.Si.,M.T.**  **NIK. 2 7009 02 0234 1** |

# **KATA PENGANTAR**

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa yang senantiasa mencurahkan rahmat dan hidayah-Nya sehingga kami dapat menyelesaikan praktikum Pemrograman Berorientasi Objek serta laporan proyek akhir praktikum yang berjudul Movie Review. Adapun laporan ini berisi tentang proyek akhir yang kami pilih dari hasil pembelajaran selama praktikum berlangsung.

Tidak lupa ucapan terima kasih kepada asisten dosen yang selalu membimbing dan mengajari kami dalam melaksanakan praktikum dan dalam menyusun laporan ini. Laporan ini masih sangat jauh dari kesempurnaan, oleh karena itu kritik serta saran yang membangun kami harapkan untuk menyempurnakan laporan akhir ini.

Atas perhatian dari semua pihak yang membantu penulisan ini, kami ucapkan terima kasih. Semoga laporan ini dapat dipergunakan seperlunya.

Yogyakarta, Mei 2020

Penyusun

# **DAFTAR ISI**

[**HALAMAN PENGESAHAN** ii](#_Toc25323565)

[**KATA PENGANTAR** iii](#_Toc25323566)

[**DAFTAR ISI** iv](#_Toc25323567)

[**BAB I PENDAHULUAN** 1](#_Toc25323568)

[1.1 Latar Belakang 1](#_Toc25323569)

[1.2 Tujuan 1](#_Toc25323570)

[**BAB II PEMBAHASAN** 2](#_Toc25323571)

[2.1 Dasar Teori 2](#_Toc25323572)

[2.2 Deskripsi Umum Proyek Akhir 3](#_Toc25323573)

[2.3 Inti Pembahasan 4](#_Toc25323574)

[**BAB III JADWAL PENGERJAAN DAN PEMBAGIAN TUGAS** 19](#_Toc25323575)

[3.1 Jadwal Pengerjaan 19](#_Toc25323576)

[3.2 Pembagian Tugas 19](#_Toc25323577)

[**BAB IV KESIMPULAN DAN SARAN** 20](#_Toc25323578)

[4.1 Kesimpulan 20](#_Toc25323579)

[4.2 Saran 20](#_Toc25323580)

[**DAFTAR PUSTAKA** 21](#_Toc25323581)

# **BAB I**

**PENDAHULUAN**

## **Latar Belakang**

Dunia perfilman mengalami kemajuan pesat selama satu dekade terakhir baik di Indonesia maupun internasional. Banyak faktor pendukungnya, salah satunya kemajuan teknologi mempermudah produksi dan distribusi film. Selain itu, adanya permintaan yang cukup tinggi dari masyarakat modern yang selalu butuh hiburan baru ditengah tempo kehidupan mereka yang sangat cepat. Oleh karena itu, selalu ada film baru yang rilis hampir setiap satu bulan sekali.

Berdasarkan fakta diatas, dunia perfilman tentunya penuh dengan persaingan. Adanya persaingan ini mempunyai dua sisi layaknya koin. Di satu sisi, positif karena adanya persaingan meramaikan dunia perfilman dan membuat produser film bisa melakukan evaluasi dari film lain. Namun, di sisi yang lain, ada negatifnya karena persaingan ini terkadang membuat produksi film harus dilakukan secepat mungkin agar waktu rilis bisa dipercepat. Hal ini terkadang bisa mempengaruhi kualitas dari film. Selain itu, terjadi monopoli studio besar yang sudah memiliki banyak penonton sehingga persaingan yang terjadi banyak terfokus pada studio besar. Padahal, banyak juga studio yang kecil sampai menengah dengan kualitas film yang boleh diadu dengan studio besar dari semua sisi baik dari tokoh, audio & visual, jalan cerita , dan lainnya.

Sisi negative tersebut pada akhirnya juga berdampak pada penonton. Banyak kasus saat orang selesai menonton film mereka merasa kecewa telah menghabiskan waktunya menonton film tersebut entah karena visual yang buruk, penokohan yang tidak tepat, atupun jalan cerita yang tidak menarik. Selain itu, ada juga kasus penonton tidak bisa menikmati film karena tidak sesuai dengan mereka. Misalnya, filmnya terlalu banyak adegan kekerasan.

Hal tersebut akhirnya mendorong adanya website atau aplikasi yang digunakan untuk melakukan ulasan (*review*) film. Pada aplikasi ini orang bisa menuturkan bagiaman film ini dan menjadi refrensi bagi orang lain. Tidak jarang yang melakukan ulasan adalah kritikus film ternama ataupun media daring yang memiliki reputasi untuk melakukan ulasan film. Kemudia aplikasi ini dapat diakses siapapun secara gratis yang membuat setiap orang yang terkoneksi internet bisa mengaksesnya.

Aplikasi seperti ini membantu para penonton untuk memilih film apa yang ingin mereka tonton. Apalagi di tengah pandemi seperti sekarang, orang butuh banyak hiburan yang bisa diakses secara daring karena hiburan yang bersifat harus keluar rumah hampir tidak ada atau bahkan dilarang. Karena itu penulis membuat aplikasi untuk melakukan ulasan film untuk membantu banyak orang memilih film hiburannya di tengah pandemi.

## **Tujuan**

Tujuan dari proyek Praktikum Pemrograman Berorientasi Objek ini adalah untuk membuat sistem *review* film untuk membantu penonton memilih film yang ingin ditonton serta untuk menyelesaikan tugas akhir dari mata kuliah Praktikum Pemrograman Berorientasi Objek.

# **BAB II**

**PEMBAHASAN**

## **2.1 Dasar Teori**

**2.1.1 Bahasa Pemrograman Java**

Java adalah bahasa pemrograman berorientasi objek. Sebagai bahasa pemrograman berorientasi objek, Java menggunakan kelas-kelas untuk mengorganisasikan kode program dalam modul-modul logikal.

**2.1.2 Pemrograman Berorientasi Objek**

Dalam bahasa pemrograman berorientasi objek, program dibagi menjadi objek-objek yang saling berinteraksi satu sama lain. Objek tersebut dapat memiliki wujud yang nyata, maupun tidak nyata (abstrak). Konsep-konsep dasar dari pemrograman berorientasi objek adalah inheritance (pewarisan), encapsulation (pengkapsulan), dan polymorphism.

**2.1.3 *Overriding***

Overriding adalah kemampuan suatu subclass untuk menulis code milik method() yang abstract milik super class, atau memodifikasi isi attribute atau isi method() yang tidak abstract, tidak private, tidak final, dan non static milik super class. Method-method yg overloading wajib berada di dalam class yg sama atau dalam satu keluarga (sebagian method()-method() tersebut di dalam superclass dan yg lain di dalam subclass).

**2.1.4 *Overloading***

Overriding adalah kemampuan suatu subclass untuk menulis code milik method() yang abstract milik super class, atau memodifikasi isi attribute atau isi method() yang tidak abstract, tidak private, tidak final, dan non static milik super class.

Modifikasi modifier akses pada attribute, method(), dan constructor() miliki super class, boleh dilakukan dengan cara memperluas daerah aksesnya.

**2.1.5 *Encapsulation***

Encapsulation atau pengkapsulan digunakan untuk menyembunyikan atribut atau method pada suatu objek, dari objek lainnya, atau dengan kata lain yakni Information hiding. Pengkapsulan dilakukan dengan memberikan modifier private pada atribut maupun method. Kemudian, untuk memberikan nilai pada atribut private, digunakan method ‘setter’, dan untuk mengambil nilai dari atribut private, digunakan method ‘getter’. Method setter dan getter ini sama seperti method biasa, hanya berbeda pada pemanfaatan yang dituliskan programmernya saja.

**2.1.6 *Inheritance***

Inheritance adalah penurunan atribut dan method dari suatu kelas induk (superclass) ke kelas anak (subclass). Pewarisan bersifat menyeluruh dan beruntun, sehingga jika A merupakan superclass dari B, B merupakan superclass dari C, maka A juga merupakan superclass dari C. Jika ada atribut atau method dari C dan B yang ingin dimodifikasi, kita hanya perlu memodifikasi atribut/method pada A. Pernyataan extends digunakan untuk melakukan pewarisan ini.

**2.1.7 *Interface***

Interface merupakan suatu kelas yang berisi method-method tanpa implementasi, namun tanpa modifier abstract. Apabila terdapat atribut, maka atribut tersebut berlaku sebagai konstanta (static final). Interface bersifat seperti kontrak yang bisa digunakan oleh setiap kelas yang nyata, dan penggunaannya bukan diwariskan (extends) melainkan diimplementasikan (implements).

**2.1.8 *Exception***

Exception merupakan suatu objek Error khusus yang dibuat ketika terjadi kesalahan pada suatu program. Setelah suatu objek exception dibuat oleh java, objek tersebut dikirimkan ke program, proses ini disebut throwing an exception. Objek exception tersebut kemudian diterima oleh program, dan penanganan exception selanjutnya terserah pada program.

Penanganan exception dalam java dibuat dengan menuliskan kode program exception-handling pada Source code. Exception-handling berguna untuk mencegah program menjadi “Hang”. Code program exception handling secara umum terdiri dari 3 bagian yaitu try, catch dan finally.

**2.1.9 SWING**

GUI merupakan suatu metode untuk antar-muka komputer berbasis grafis. GUI digunakan dalam pembuatan program aplikasi dengan mempertimbangkan dua aspek yaitu keindahan tampilan dan kemudahan penggunaan program. Swing adalah kumpulan class–class yang digunakan untuk membuat GUI. Swing adalah GUI yang lightweight, karena lebih multiplatform dan sedikit menggunakan Native class.

**2.1.10 Java *Database Connectivity***

JDBC (Java *DataBase Connectivity*) merupakan teknologi Java yang populer sebagai standar pengaksesan database. Java tidak membedakan cara pemrograman database yang satu dengan lainnya. Sebagai contoh: cara Java mengolah tabel di Oracle dan MySQL sama saja, yang berbeda hanyalah alamat URL koneksi dan driver JDBC-nya. Sehingga kita dapat membuat sebuah program yang sama di Java untuk mengolah database yang ada di MySQL, Oracle, SQL Server dan lain-lain asalkan nama dan struktur tabelnya sama.

**2.1.3 *Model View Controller***

Model View Control pertama kali diperkenalkan oleh Trygve Reenskaug, seorang pengembang perangkat lunak Smalltalk pada Xerox Palo Alto Research Center pada tahun 1979. Konsep ini memisahkan antara akses data, logika bisnis, dan antarmuka pada perangkat lunak, secara detail MVC dibagi 3, yaitu :

1. Model, merupakan bagian dari MVC yang bertugas menghubungkan langsung antara tugas-tugas dengan proses di Database (Create, Read, Update, Delete). Sebagai contoh pada Java, dibuat kelas sendiri antara koneksi dengan query akses data ke database. Jadi untuk menggunakan fungsi Insert, Update, Delete, dan Select kita tinggal memanggil method-method yang menangani hal tersebut.
2. View, merupakan bagian dari MVC yang bertugas menghubungkan antara pengguna dengan tugas perangkat lunak, dengan kata lain View menangani bagian Interface atau Antarmuka untuk mempermudah si pengguna menggunakan perangkat lunak.
3. Controller, menjadi tulang punggung perangkat lunak. Controller menghubungkan antara View dan Model yang meng-handle semua proses-proses yang terjadi didalamnya. Sebagai contoh kasus User Login, tugas Controller adalah mencocokkan data yang di-inputkan user dengan data yang ada di database

## **2.2 Deskripsi Umum Proyek Akhir**

Program ini merupakan program yang digunakan untuk membuat ulasan sebuah film agar penonton dapat dengan mudah menentukan pilihannya.

Program ini dimulai dari menu *login* untuk pengguna yang ingin mengulas sebuah film. Pada halaman *login*, pengguna dapat mengisi *username* dan *password* untuk masuk ke halaman menu. Jika belum memiliki akun, pengguna dapat melakukan registrasi terlebih dahulu dengan menekan tombol registrasi pada halaman *login*.

Halaman menu menampilkan empat pilihan yaitu input, *review* anda, *profile*, dan daftar film. Menu input digunakan untuk menginputkan sebuah film yang akan diulas, jika film belum ada, maka film otomatis akan ditambahkan dalam *database*.

Menu *review* anda digunakan untuk menampilkan *review* atau ulasan yang sudah pernah user inputkan. Tampilan data berupa table yang dapat diklik untuk meng-*update* datau menghapus data ulasan.

Menu *profile* digunakan untuk menampilkan identitas pengguna dan menampilkan tingatan ulasan yang dilihat dari jumlah ulasan yang dibuat. Selain itu terdapat tombol ubah password untuk mengubah password pengguna.

Menu Daftar film digunakan untuk menampilkan semua film yang telah diulas oleh semua pengguna. Data ditampilkan dalam bentuk table yang dapat diklik untuk melihat detail dari film tersebut.

## **2.3 Inti Pembahasan**

**2.3.1 *Listing* Program**

viewnya.b\_login.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

if (viewnya.getFusername().equals("")

|| viewnya.getFpassword().equals("")) {

JOptionPane.showMessageDialog(null, "field kosong");

} else {

String nama = viewnya.getFusername();

String pass = viewnya.getFpassword();

if (modelnya.cekLogin(nama, pass)) { // kalo nilai true, nanti pindah halaman

System.out.println("Pindah ke Main Menu");

MVCMainMenu mainMenu = new MVCMainMenu(nama);

}

viewnya.fusername.setText("");

viewnya.fpassword.setText("");

}

}

});

***Listing* Program 2.1** Controller Login

public boolean cekLogin(String username,String pass){

int cek = 0;

try {

statement = (Statement)koneksi.createStatement();

String querynya = "SELECT \* FROM `user` WHERE `username`='"+username+"' AND `password`='"+pass+"'";

ResultSet hasilnya = statement.executeQuery(querynya);

while (hasilnya.next()){

cek++;

}

if(cek>0){

JOptionPane.showMessageDialog(null,"Login Success");

//mengembalikan nilai true biar di cek di controller nantinya

return true;

}

else{

JOptionPane.showMessageDialog(null,"Login Failed");

}

// statement.executeUpdate(querynya);

} catch (Exception e) {

JOptionPane.showMessageDialog(null, e.getMessage(),"Hasil",JOptionPane.ERROR\_MESSAGE);

}

return false;

}

***Listing* Program 2.2** Model Login

public class RataRataUser implements Hitung {

private int nilai1;

private int nilai2;

private int nilai3;

public RataRataUser(int nilai1, int nilai2, int nilai3) {

this.nilai1 = nilai1;

this.nilai2 = nilai2;

this.nilai3 = nilai3;

}

public int hitung(){

return (nilai1 + nilai2 + nilai3)/3;

}

}

***Listing* Program 2.3** Proses Hitung Nilai Keseluruhan Review Pengguna

public class RataRataSemua extends RataRataUser{

private int nilai;

private int nilaiDbUser;

public RataRataSemua(int nilai, int nilai1, int nilai2, int nilai3) {

super(nilai1, nilai2, nilai3);

this.nilai = nilai;

}

public RataRataSemua(int nilai, int nilaiDbUser, int nilai1, int nilai2, int nilai3) {

super(nilai1, nilai2, nilai3);

this.nilai = nilai;

this.nilaiDbUser = nilaiDbUser;

}

@Override

public int hitung() {

return (super.hitung() + nilai)/2;

}

public int hitungDelete(){

return (nilai\*2)-nilaiDbUser;

}

}

***Listing* Program 2.4** Proses Hitung Nilai Keseluruhan Review dari Semua Pengguna

private boolean filmAda(String judul) {

int cek = 0;

try{

statement = (Statement)koneksi.createStatement();

String querynya = "SELECT \* FROM `movie` WHERE `judul`='"+judul+"'";

ResultSet hasilnya = statement.executeQuery(querynya);

while (hasilnya.next()){

cek++;

}

if (cek>0) {

return true;

}

}

catch(Exception e){

System.out.println(e.getMessage());

}

return false;

}

***Listing* Program 2.5** Pemeriksaan Judul Film

private boolean filmUdahDireview(String judul, String username) {

int cek = 0;

try{

statement = (Statement)koneksi.createStatement();

String querynya = "SELECT \* FROM review WHERE username='" + username + "' AND id\_movie=(SELECT id\_movie FROM movie WHERE judul= '"+ judul + "')";

ResultSet hasilnya = statement.executeQuery(querynya);

while (hasilnya.next()){

cek++;

}

if (cek>0) {

return true;

}

}

catch(Exception e){

System.out.println(e.getMessage());

}

return false;

}

***Listing* Program 2.6** Model Pemeriksaan Judul Film yang Sudah diinput oleh Pengguna

// ngitung rata2 baru review oleh user

RataRataUser rataBaruUser = new RataRataUser(intPlot, intKarakter, intVisualAudio);

int BaruUser = rataBaruUser.hitung();

// ambil nilai dari movie sebelumnya

statement = (Statement)koneksi.createStatement();

String query = "SELECT nilai FROM movie WHERE judul='" + judul + "'";

ResultSet hasil = statement.executeQuery(query);

while (hasil.next()){

nilaiSemua = hasil.getInt("nilai");

}

// ambil keseluruhan dari review sebelumnya

statement = (Statement)koneksi.createStatement();

query = "SELECT keseluruhan FROM review WHERE username='" + username + "' AND id\_movie=(SELECT id\_movie FROM movie WHERE judul='" + judul + "')";

hasil = statement.executeQuery(query);

while (hasil.next()){

nilaiUser = hasil.getInt("keseluruhan");

}

***Listing* Program 2.7** Model Pengambilan Nilai, Keseluruhan, dan Penghitungan Rata – Rata Baru

// nyari rata2 sebelumnya diambil dari data2 yg ada sekarang

RataRataSemua rataBaruSemua = new RataRataSemua(nilaiSemua, nilaiUser, nilaiUser, nilaiUser, nilaiUser); // 3 nilai terakhir tdk dibutuhkan

int rataSemuaSebelum = rataBaruSemua.hitungDelete();

System.out.println("Nilai rata2 semua sebelumnya " + rataSemuaSebelum);

// ngitung rata2 baru dari data sebelumnya dan data yg baru dimasukkan

rataBaruSemua = new RataRataSemua(rataSemuaSebelum, intVisualAudio, intPlot, intKarakter);

int rataSemuaBaru = rataBaruSemua.hitung();

***Listing* Program 2.8** Model Perhitungan Nilai Sebelum Pengguna Memberi Review

public boolean cekUser(String username){

int cek = 0;

try {

statement = (Statement)koneksi.createStatement();

String querynya = "SELECT \* FROM `user` WHERE `username`='"+username+"'";

ResultSet hasilnya = statement.executeQuery(querynya);

while (hasilnya.next()){

cek++;

}

if(cek>0){

JOptionPane.showMessageDialog(null,"Username sudah ada");

//mengembalikan nilai true biar di cek di controller nantinya

return true;

}

else{

return false;

}

} catch (Exception e) {

JOptionPane.showMessageDialog(null, e.getMessage(),"Hasil",JOptionPane.ERROR\_MESSAGE);

}

return false;

}

***Listing* Program 2.9** Memeriksa username sudah ada atau belum

this.registerView.b\_register.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

// buat register akun

String username = registerView.getFusername();

String password = registerView.getFpassword();

String password\_confirm = registerView.getFpasswordConfirm();

if (username.equals("") || password.equals("") || password\_confirm.equals("")) { // ngecek field kosong atau engga

JOptionPane.showMessageDialog(null, "field kosong");

}

else {

System.out.println(password + " " + password\_confirm);

if (password.equals(password\_confirm)) { // ngecek password sama atau engga

System.out.println("yey");

if (!registerModel.cekUser(username)) {

registerModel.registerUser(username,password);

}

}

else {

JOptionPane.showMessageDialog(null, "Password tidak sama");

}

}

}

});

***Listing* Program 2.10** Memeriksa password untuk registrasi sudah betul atau belum

public boolean cekPassLama(String username,String passLama){

int cek=0;

try {

statement = (Statement)koneksi.createStatement();

String querynya = "SELECT `password` FROM `user` WHERE `username`='"+username+"' AND `password`='"+passLama+"'";

ResultSet hasilnya = statement.executeQuery(querynya);

while (hasilnya.next()){

cek++;

}

if(cek>0){

//mengembalikan nilai true biar di cek di controller nantinya

return true;

}

else{

JOptionPane.showMessageDialog(null,"Password Lama Salah");

}

} catch (Exception e) {

JOptionPane.showMessageDialog(null, e.getMessage(),"Hasil",JOptionPane.ERROR\_MESSAGE);

}

return false;

}

***Listing* Program 2.11** Memeriksa password lama untuk proses ganti password

viewnya.b\_ganti.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent ae) {

if(viewnya.getTfPassLama().equals("")||viewnya.getTfPassBaru().equals("")||

viewnya.getTfPassKonfirm().equals("")){

JOptionPane.showMessageDialog(null, "field kosong");

}

else{

String PassLama = viewnya.getTfPassLama();

String PassBaru = viewnya.getTfPassBaru();

String PassKonfirm = viewnya.getTfPassKonfirm();

if (modelnya.cekPassLama(viewnya.getUsername(), PassLama)) {

if (PassBaru.equals(PassKonfirm)) {

modelnya.gantiPass(viewnya.getUsername(), PassBaru);

viewnya.tfPassBaru.setText("");

viewnya.tfPassLama.setText("");

viewnya.tfPassKonfirm.setText("");

}

else{

JOptionPane.showMessageDialog(null, "Konfirmasi ulang Password");

}

}

}

});

***Listing* Program 2.12** Memeriksa password baru sekaligus proses ganti password

viewnya.tabel.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent me) {

super.mouseClicked(me);

baris = viewnya.tabel.getSelectedRow();

dataterpilih = viewnya.tabel.getValueAt(baris, 0).toString();

JOptionPane.showMessageDialog(null, "cek Film");

MVCFilmDetail detail = new MVCFilmDetail(dataterpilih);

}

});

***Listing* Program 2.13** Proses klik judul film untuk mengetahui review orang lain dan detail film

public String[] tampilkanData(String id\_movie){

try {

int jmlData=0;

String data[] = new String[6];

String query = "SELECT \* FROM `movie` WHERE `id\_movie`= '"+id\_movie+"'";

statement = (Statement)koneksi.createStatement();

ResultSet resultSet = statement.executeQuery(query);

while (resultSet.next()){

data[0] = resultSet.getString("id\_movie");

data[1] = resultSet.getString("durasi");

data[2] = resultSet.getString("judul");

data[3] = resultSet.getString("genre");

data[4] = resultSet.getString("sutradara");

data[5] = resultSet.getString("nilai");

}

return data;

} catch (Exception errorTampil) {

JOptionPane.showMessageDialog(null, errorTampil.getMessage(),"Hasil",JOptionPane.ERROR\_MESSAGE);

return null;

}

}

***Listing* Program 2.14** Menampilkan detail film

public String[][] tampilkanReview(String id\_movie){

try {

int dataReview = 0;

String data[][] = new String[banyaknyaReview(id\_movie)][3];

String query = "Select \* from `review` WHERE `id\_movie`= '"+id\_movie+"'";

ResultSet resultSet = statement.executeQuery(query);

while (resultSet.next()){

data[dataReview][0] = resultSet.getString("username");

data[dataReview][1] = resultSet.getString("keseluruhan");

data[dataReview][2] = resultSet.getString("review");

dataReview++;

}

return data;

} catch (Exception errorTampil) {

JOptionPane.showMessageDialog(null, errorTampil.getMessage(),"Hasil",JOptionPane.ERROR\_MESSAGE);

return null;

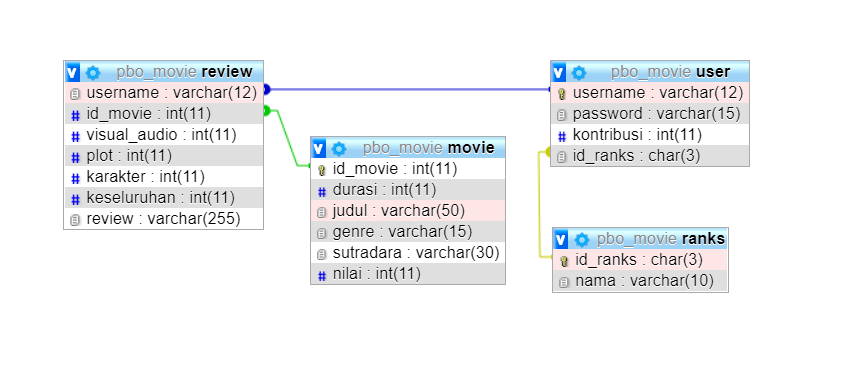
}

}

***Listing* Program 2.15** Menampilkan review user lain

**2.3.2 Penjelasan Program**

1. ***Database***



**Gambar 2.1** Rancangan *Database*

Pada program ini terdapat empat entitas yaitu *user*, *movie*, *review*, dam *ranks.* *User* digunakan untuk menyimpan data – data pengguna seperti username, password, jumlah kontribusi, dan id\_ranks yang terhubung langsung ke table *ranks*.

Table *ranks* digunakan untuk menyimpan kategori *rank* mulai dari *baby, grown up,* sampai *warrior*.

Selanjutnya ada table *movie* yang digunakan untuk menyimpan data – data film seperti judul, durasi, genre, sutradara, dan nilai untuk menyatakan nilai keseluruhan dari film tersebut.

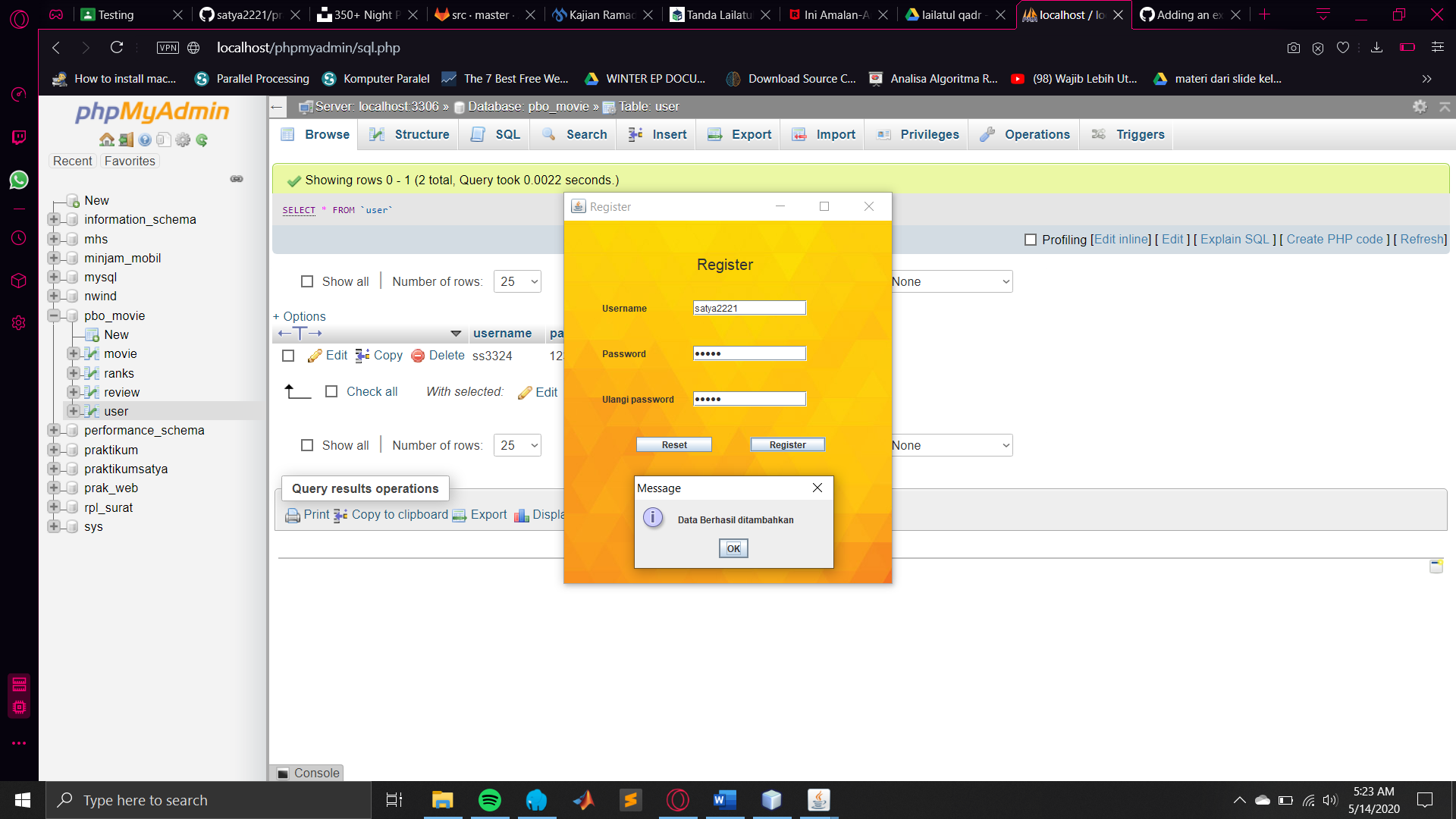
Kemudian table *review* merupakan komposit dari table user dan movie yang menyimpan data – data kriteria penilaian (visual audio, karakter, dan plot), nilai keseluruhan yang merupakan rata – rata dari kriteria penilaian, dan *review* singkat dari film.

1. **Login**

Pada login akan diminta untuk menginputkan username dan password. Kemudian data yang diinputkan akan diambil oleh controller seperti listing program 2.1. Setelah itu data yang diinputkan akan diperiksa pada login model seperti pada listing program 2.2. Jika data ada, akan dilanjutkan ke halaman menu yang memuat berbagai menu. Dan jika data tidak ada, maka akan menampilkan pesan eror.

1. **Register**

Jika, user belum memiliki akun (username dan password), maka user harus melakukan registrasi terlebih dahulu. User akan diminta untuk membuat username dan passwordnya sendiri. Password dipastikan dua kali untuk memastikan password yang dimasukkan user sudah sesuai dengan kemauan user. Proses dipastikannya password ada di controller sebagiamana listing program 2.10 Untuk username, setiap user harus memilki username nya sendiri-sendiri sehingga jika ada username sama maka akan muncul dialog yang meminta user untuk mengganti username. Proses tersebut akan dilakukan oleh model dari halaman Register seperti pada listing program 2.9. Jika dua proses diatas tidak mengalami masalah, maka sebuah dialog akan muncul seperti gambar



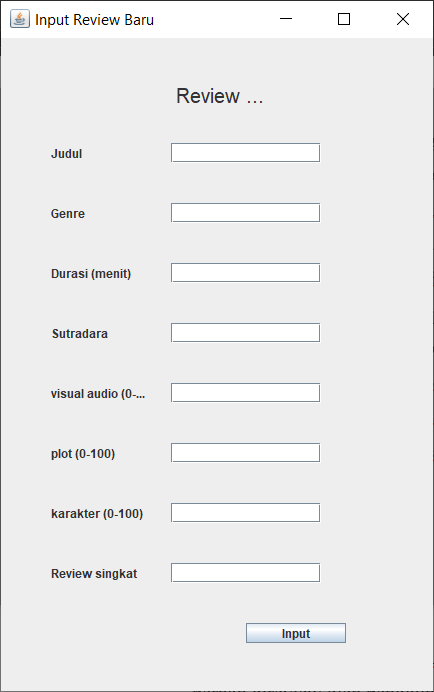
**Gambar 2.2** Muncul pesan bahwa registrasi berhasil

1. **Input *Review***

Pada menu ini pengguna akan diminta untuk menginputkan data – data seperti judul, genre, durasi, sutradara, kriteria penilaian (visual audio, plot, dan karakter), dan review singkat. Untuk kriteria penilaian hanya dapat menginputkan angka saja.

Proses yang terjadi yaitu program akan menghitung nilai yang telah diinputkan seperti listing program 2.3. Kemudian mengambil data yang telah diinputkan. Selanjutnya program akan memeriksa apakah judul yang diinputkan sudah ada atau belum seperti listing program 2.5. Jika belum ada, maka akan menginputkan data film baru ke dalam table movie kemudian menginputkan data – data ke dalam table review menggunakan data nilai yang telah diinputkan dan dihitung pada listing program 2.3.

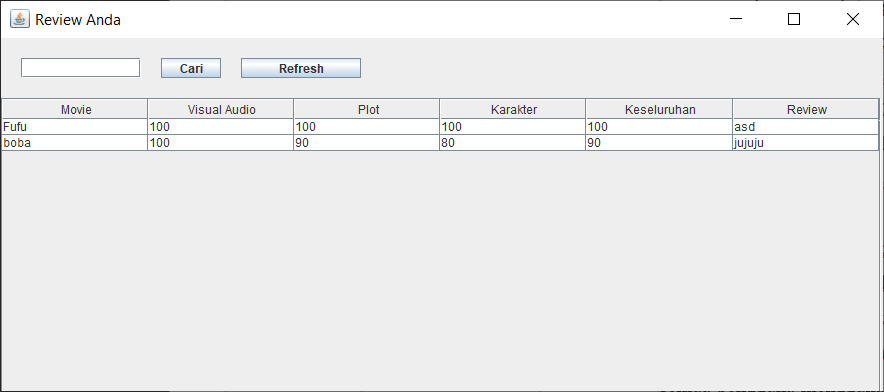
Jika film sudah ada, program akan memeriksa apakah user pernah menginputkan judul film tersebut atau belum menggunakan listing program 2.6. Jika sudah pernah, maka akan ditampilkan pesan eror, jika belum pernah maka memasukkan data ke dalam table review, kemudian akan menghitung nilai keseluruhan review dari semua pengguna menggunakan listing program 2.4, lalu setelah mendapatkan nilai baru maka program akan mengupdate nilai pada table movie serta menambahkan satu kontribusi kepada pengguna.



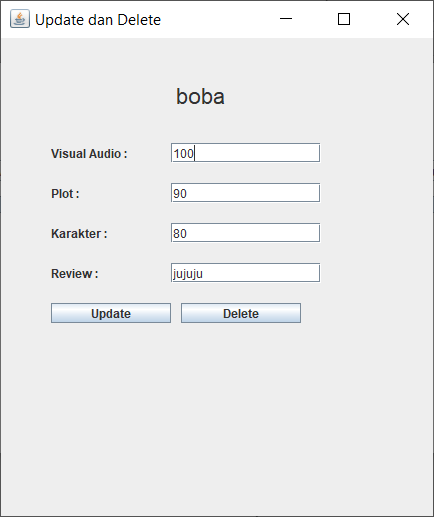
**Gambar 2.3** Tampilan Input Data Review

1. ***Review* anda**

Pada bagian ini akan ditampilkan table yang berisi data – data film yang pernah direview oleh pengguna. Pada saat table di klik, akan ada opsi untuk update ataupun delete data.



**Gambar 2.4** Tampilan Review Anda



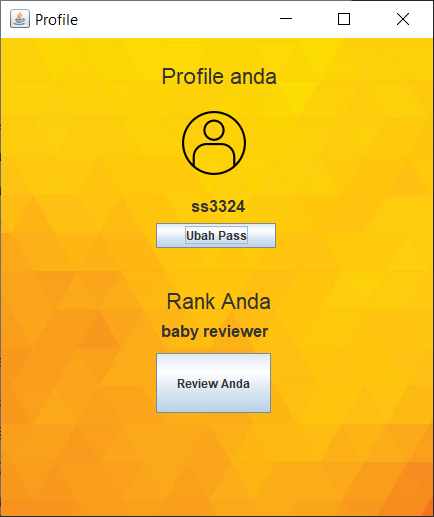
**Gambar 2.5** Tampilan Opsi Update dan Delete

Update digunakan untuk memperbarui nilai review yang pengguna inputkan sebelumnya. Pertama program akan mengambil nilai dari input user. Kemudian program akan menghitung rata – rata baru sesuai data yang diinputkan kemudian mengambil nilai dari table movie dan keseluruhan dari table review seperti pada listing program 2.7. Selanjutnya program akan mencari nilai movie sebelum pengguna menginputkan review dari film yang dipilih seperti listing program 2.8. Setelah mendapatkan nilai baru program akan meng-*update* data yang ada.

Pada Delete, proses yang digunakan hamper sama, namun program hanya melakukan perhitungan untuk mendapat nilai movie sebelum pengguna menginputkan review dari film yang dipilih seperti listing program 2.8. Setelah mendapatkan nilai tersebut, data review pengguna pada table review akan dihapus, kemudian data nilai pada table movie akan di-*update* menggunakan nilai yang didapatkan pada listing program 2.8. Lalu kontribusi dari pengguna akan dikurangi satu.

1. ***Profile***

Pada halaman ini digunakan user untuk bisa mengganti password mereka dan menuju halaman Review anda serta mengetahui ranking user. selain melalui main menu. Untuk mengubah password user perlu click tombol ganti pass sebagaiman pada gambar 2.6. Setelah di click maka window baru akan muncul. Pada window tersebut user diminta untuk memasukkan password lama, password baru, dan konfirmasi password baru. Password lama akan dicek pada model halaman ganti password seperti pada listing program 2.11. Selanjutnya, password baru akan dicek pada controller halaman ganti password sebagaimana listing 2.12.

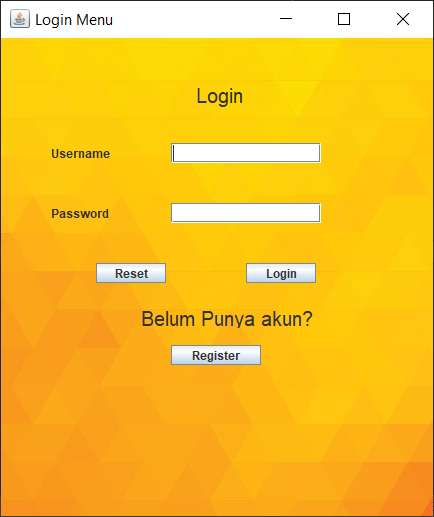


**Gambar 2.6** Tampilan Menu profile

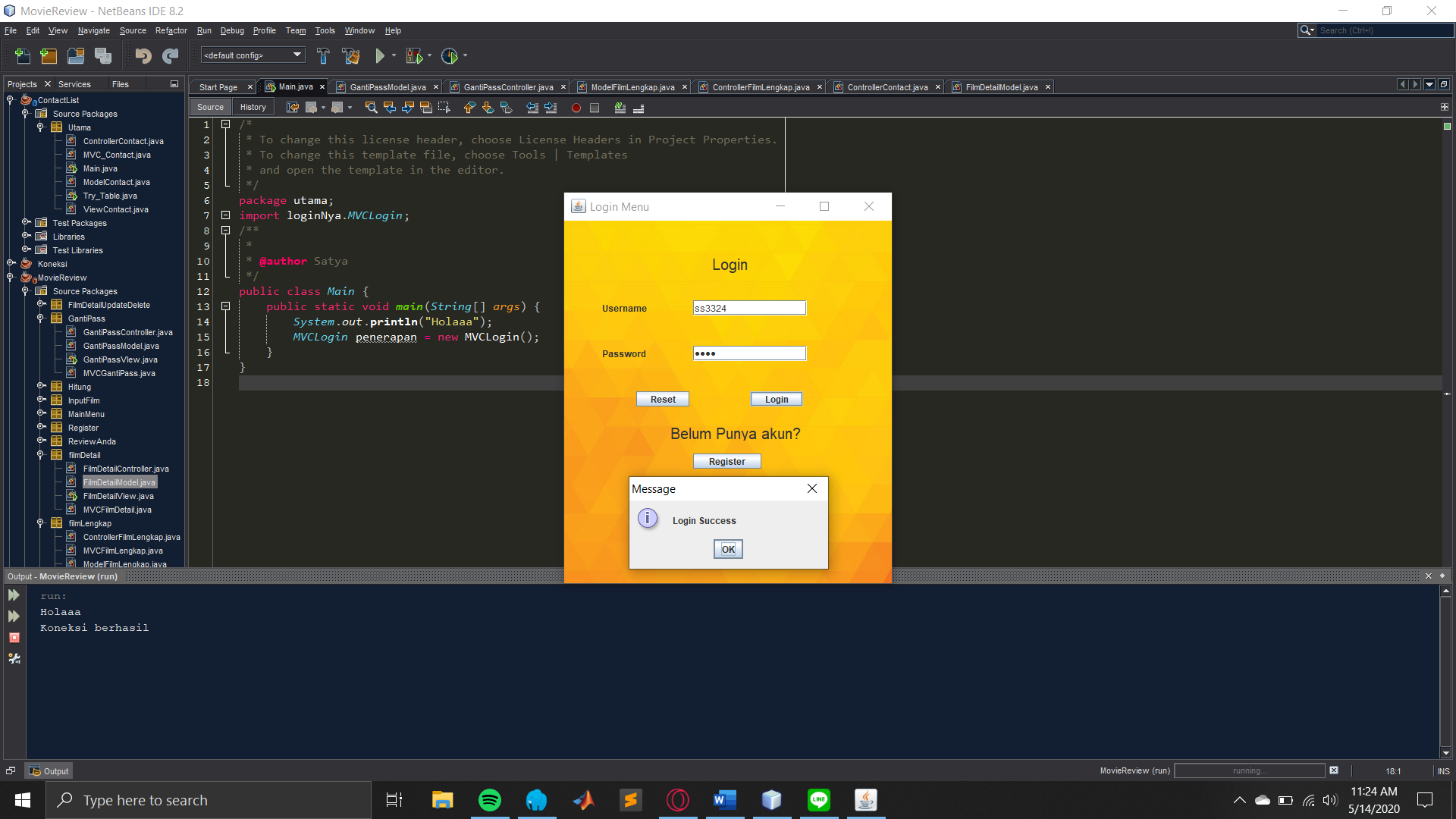
1. **Daftar Film**

Pada halaman ini user bisa melihat seluruh data film yang ada, baik yang pernah user review ataupun belum. Dengan halaman ini user diharapkan bisa mengeksplorasi film baru atau lama yang bisa jadi tontonan baru. Selain itu, user bisa melihat penilaian kebanyakan orang sehingga bisa menjadi penilaian awal sebelum menonton. Untuk mendukung hal tersebut, user bisa melihat review orang lain pada film yang diklik oleh user. Proses tersebut ada pada listing 2.13. Selain muncul review user lain pada film yang dipilih, ada juga detail tentang film tersebut yaitu genre, durasi, dan sutradara film tersebut. Proses munculnya data film dan review orang lain ada pada listing 2.14 & 2.15

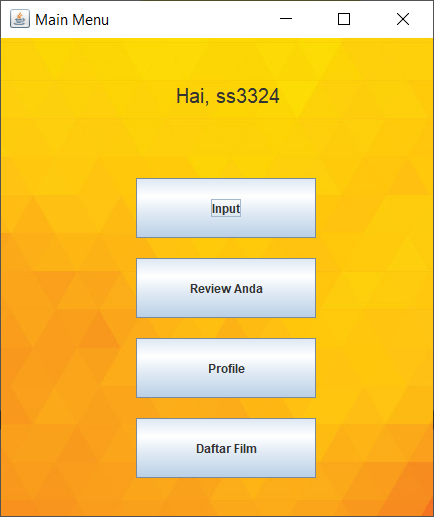
**2.3.3 Output Program**



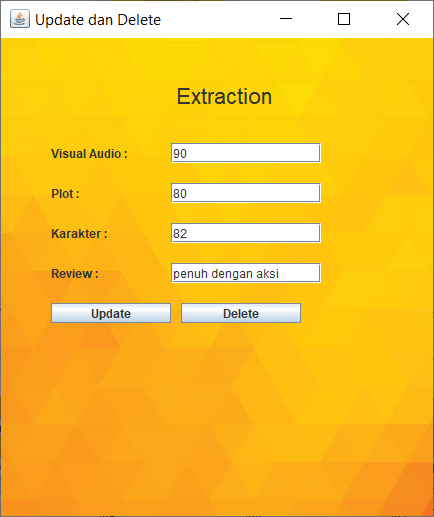
**Gambar 2.7** Tampilan halaman login yang akan pertama kali tampil setelah di run



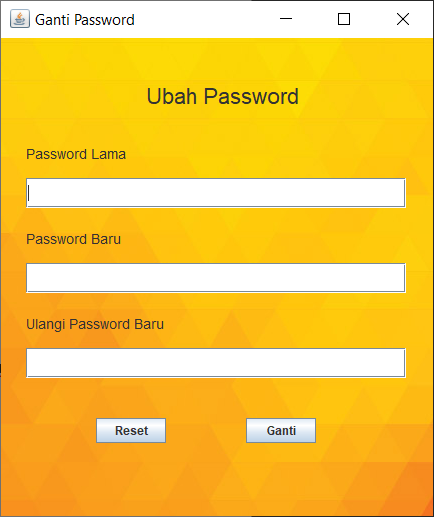
**Gambar 2.8** Jika berhasil login maka akan muncul dialog



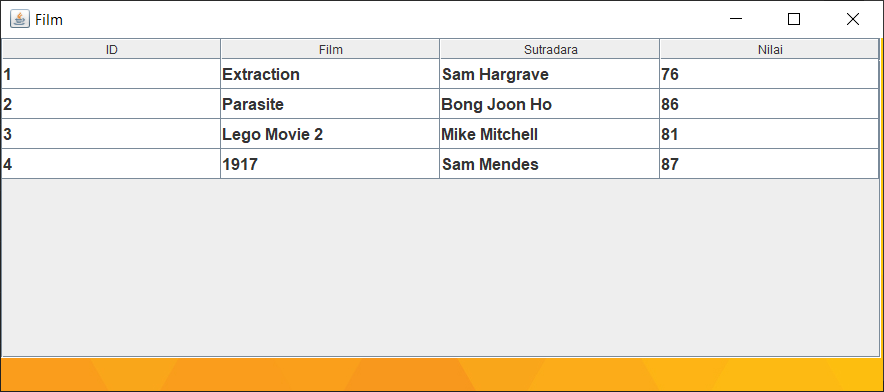
**Gambar 2.9** Tampilan menu setelah berhasil login



**Gambar 2.10** Tampilan pada menu review anda bagian untuk mengupdate penilaian atau menghapus review



**Gambar 2.11** Tampilan pada menu profile bagian untuk mengubah password



**Gambar 2.12** Tampilan menu daftar film



**Gambar 2.13** Tampilan detail & review dari user

# **BAB III**

**JADWAL PENGERJAAN DAN PEMBAGIAN TUGAS**

## **3.1 Jadwal Pengerjaan**

Jadwal pengerjaan proyek Praktikum Pemrograman Berorientasi Objek *Movie Review* ini sebagai berikut:

**Tabel 3.1** Tabel Jadwal Pengerjaan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| no | Nama Tugas | Minggu ke- | | | |
| April | | Mei | |
| 3 | 4 | 1 | 2 |
| 1 | Menentukan Judul Proyek |  |  |  |  |
| 2 | Perancangan Konsep Proyek |  |  |  |  |
| 3 | Desain Skenario Program |  |  |  |  |
| 4 | Halaman Login |  |  |  |  |
| 5 | Register Akun |  |  |  |  |
| 6 | Halaman Input Data |  |  |  |  |
| 7 | Halaman Review Anda |  |  |  |  |
| 8 | Halaman Profile |  |  |  |  |
| 9 | Halaman Daftar Film |  |  |  |  |
| 10 | Pembuatan Laporan |  |  |  |  |

## **3.2 Pembagian Tugas**

Pembagian tugas proyek Praktikum Pemrograman Berorientasi Objek *Movie Review* ini sebagai berikut:

**Tabel 3.2** Tabel Pembagian Tugas

|  |  |  |
| --- | --- | --- |
| no | aktivitas | penanggung jawab |
| 1 | Pembahasan Judul | Satya & Allyandaru |
| 2 | Perancangan Konsep | Satya & Allyandaru |
| 3 | Desain Skenario | Satya & Allyandaru |
| 4 | Halaman Login | Satya & Allyandaru |
| 5 | Halaman Register | Allyandaru |
| 6 | Halaman Input Data | Allyandaru |
| 7 | Halaman Review Anda | Allyandaru |
| 8 | Halaman Profile | Satya |
| 9 | Halaman Daftar Film | Satya |
| 10 | Pembuatan Laporan | Satya & Allyandaru |

# **BAB IV**

**KESIMPULAN DAN SARAN**

## **4.1 Kesimpulan**

Kesimpulan yang dapat diambil dari pengerjaan ini adalah proyek *Movie Review* yang telah dibuat dapat melakukan penginputan ulasan film sehingga pengguna dapat membaca ulasan yang ada sebelum menonton sebuah film untuk menentukan pilihan film yang akan ditonton pengguna. Selain itu program dapat digunakan untuk meng-*update* data ulasan jika sewaktu-waktu memerlukan perubahan, menampilak *profile* pengguna dan menampilkan semua data ulasan dari semua pengguna untuk mempermudah pengguna sebelum menonton film.

## **4.2 Saran**

Adapun saran yang dapat dipertimbangkan untuk penyempurnaan dan pengembangan program *Movie Review* lebih lanjut adalah sebagai berikut:

1. Sistem *Movie Review* diharapkan dapat bekerja dengan lebih optimal dengan menyediakan fitur-fitur maupun menu-menu yang lebih banyak serta lebih kompleks, sehingga pengguna dapat lebih nyaman dalam melakukan aktivitas.
2. Tampilan pada yang diharapkan lebih baik dan kedepannya bisa diperbarui sesuai dengan teori UI UX.

# **DAFTAR PUSTAKA**

Anonim.2019.*Modul Praktikum Pemrograman Berorientasi Objek Program Studi Teknik Infromatika Jurusan Teknik Informatika Fakultas Tekik Industri UPN “Veteran” Yogyakarta*. Yogyakarta: UPN “Veteran” Yogyakarta.