QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

Movie Website Manager

BNT Group

Members:

- Kiều Hiếu Thành CE140026
- Dương Trọng Nhân CE130213
- Nguyễn Tuấn Bằng CE130421

Mentor: Võ Hồng Khanh

Table of Contents

I. Problem Definition 3

II. User Requirements 4

1. Functions 4

2. System Requirements 5

III. Roles 6

IV. Schedule 7

V. Design Pattern 9

1. Database Structure 9

2. Table Structure 10

VI. Program Interface 15

VII. Maintenance and Updating 26

VIII. Check List 27

I. Problem Definition

Web movie management software is a software that specializes in providing feature films from many leading domestic and international film producers, with a variety of interesting movies and movies. With a team of professional programmers from the BNT Group that has a lot of experience. Our films are full of information such as movie name, production year, director, actor, country. Today, the technology age of 4.0 is the time when we can watch movies at home without having to go to the theater to see, our company wants to set up a movie web management application to watch movie and manage (for administrators) online. The requirement is to build a website to watch online movies that manage movies, viewers, money loaded into accounts, advertise, download movies, download movies, etc.

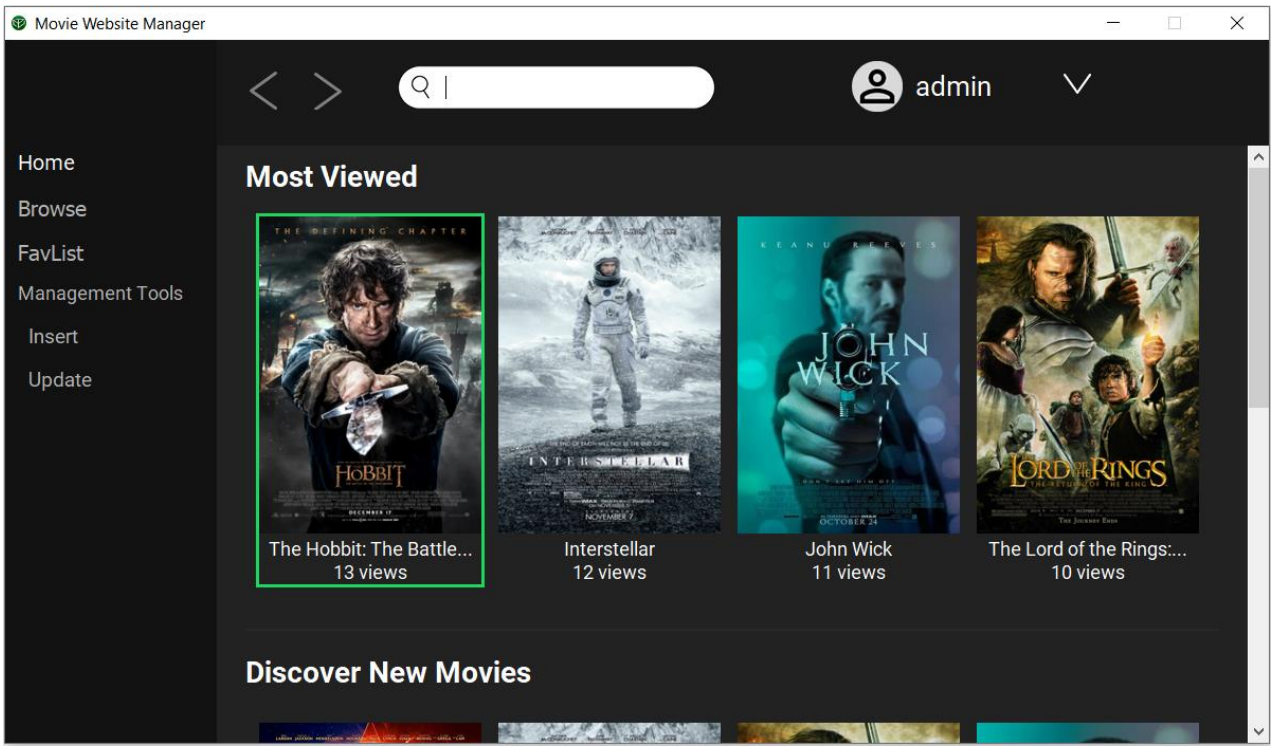


FIGURE 1 OUR MOVIE WEBSITE MANAGER APPLICATION

II. User Requirements

1. Functions

Functions in our program varies depending on the security level of user account.

Normal users (level 0) will only have basic functions like:

- Create their own account
- Update their's user profile
- Browse, Search for movies
- Create/Update favorite list

Moderators (level 1) will have additional functions:

- Create/Update Movies, Directors, Actors, Studio, Country

Administrators (level 2) can access to all functions of others level and plus these functions:

- Create accounts for other users
- Can access/update all information of non-admin account (except account's password)

2. System Requirements

Hardware Requirement (minimum)	Hardware Requirement (recommended)
+ CPU: Intel Pentium 2 266 MHz **+ RAM:** 128 MB **+ Storage:** 128 MB of free disk space	**+ CPU:** Intel Core i9 9900K 3.6 GHz **+ RAM:** 16 GB **+ Storage:** 256 GB of free disk space

Software Requirement
+ Window XP or higher + JRE 1.7 or higher + VLC media player version 3.0.0 or higher

III. Roles

Role	Member	Responsibilities
Leader	ThanhKH	Creating Schedule and Check List, Assigning assignment to team members
Secretary	NhanDT	Taking note
Analyst & Design	ThanhKH, NhanDT, BangNT	Table of content, Problem Defining, User Requirement. Interface Designing
Coder	ThanhKH, NhanDT, BangNT	Programing all functions for applications
Tester	NhanDT, ThanhKH	Testing all functions
Maintenance	NhanDT, BangNT	Checking and backup data

IV. Schedule

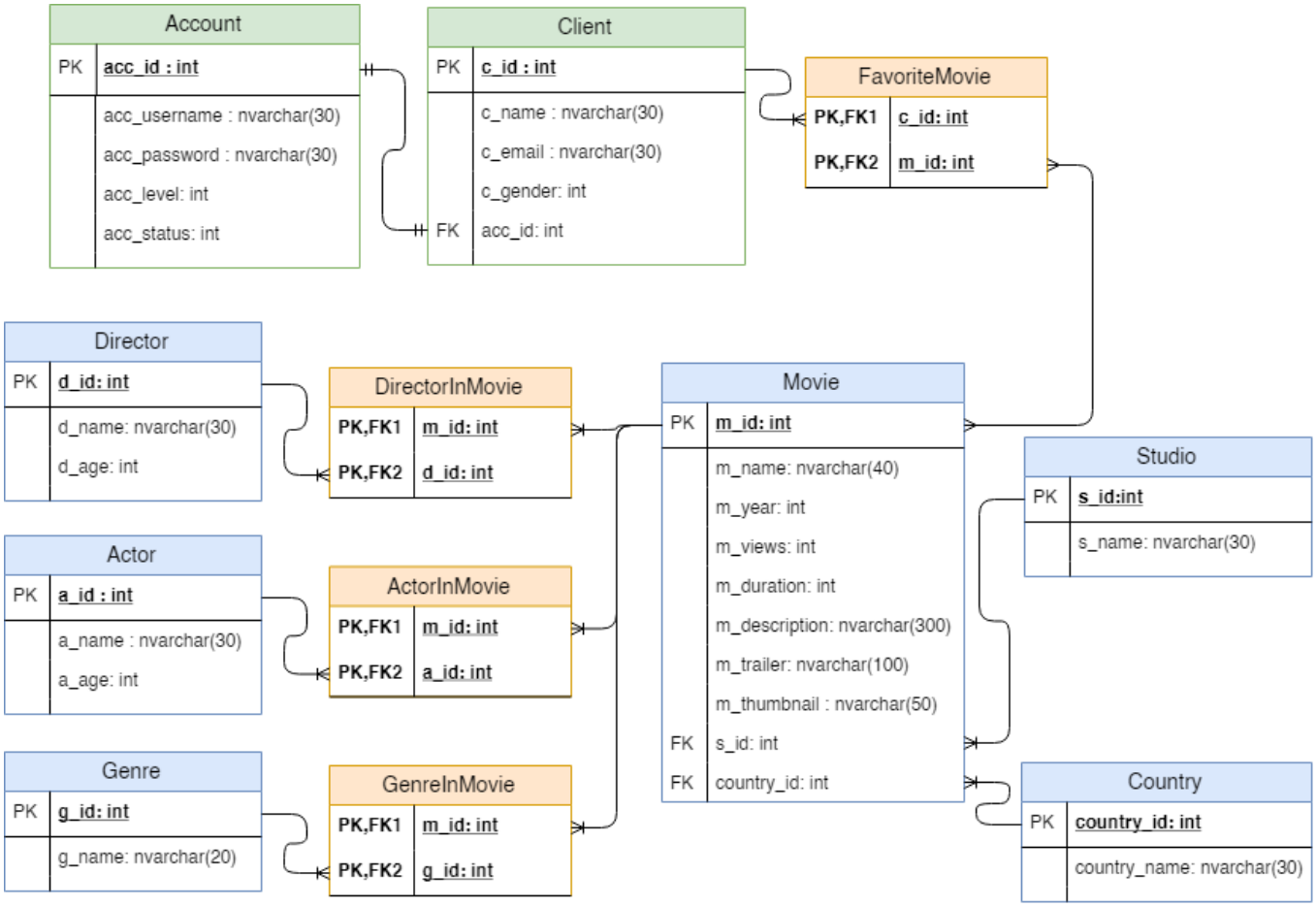
No.	Assignment	Member	Start Time	ETF[1] (hrs)	Deadline
1	Problem Definition	ThanhKH, NhanDT	28/06/2019	2	23h50 28/06/2019
2	User Requirement and System Requirement	BangNT	28/06/2019	4	23h50 28/06/2019
3	Analysis	ThanhKH	08/07/2019	12	23h50 08/07/2019
4	Database Diagram & Table Structure	ThanhKh, NhanDT	09/07/2019	24	23h50 10/07/2019
5	Basic GUI Design	ThanhKH, NhanDT, BangNT	11/07/2019	24	23h50 12/07/2019
6	Exception & Info Classes Coding	ThanhKH, NhanDT, BangNT	12/07/2019	4	23h50 12/07/2019
7	Model Classes Coding	ThanhKH, NhanDT, BangNT	13/07/2019	16	20h 14/07/2019
8	Log In & Create Account Frame	NhanDT	15/07/2019	24	23h50 17/07/2019

[1] Estimate Time to Finish

9	Main menu Frame	ThanhKH, BangNT	15/07/2019	48	23h50 17/07/2019
10	Detail GUI Design	ThanhKH, NhanDT, BangNT	18/07/2019	48	23h50 19/07/2019
11	Testing & Debugging	BangNT, NhanDT	19/07/2019	12	23h50 19/07/2019
12	Document Finishing	ThanhKH	19/07/2019	4	12h 19/07/2019
13	Presentation	BangNT	24/07/2019	2	14h15 24/07/2019
14	Implementation	BangNT, NhanDT	25/07/2019	3	17h00 25/07/2019

V. Design Pattern

1. Database Structure



2. Table Structure

Account			
Fleld Name	**Data Type**	**Contraint**	**Description**
acc_id	int	PRIMARY KEY	ID of Account
acc_username	nvarchar (30)	NOT NULL	Username of Account
acc_password	nvarchar (20)	NOT NULL	Password of Account
acc_level	int	NOT NULL	Security level of Account
acc_status	int	NOT NULL	Status of Account

Client			
Fleld Name	**Data Type**	**Contraint**	**Description**
c_id	int	PRIMARY KEY	ID of Client
c_name	nvarchar (20)	NOT NULL	Name of Client
c_email	navrchar (30)	NOT NULL	Email of Client
c_gender	int	NOT NULL	Gender of Client
acc_id	Int	FOREIGN KEY	ID of Client's Account

Movie			
Fleld Name	**Data Type**	**Contraint**	**Description**
m_id	int	PRIMARY KEY	ID of Movie
m_name	nvarchar (40)	NOT NULL	Name of Movie
m_year	int	NOT NULL	Year of Movie
m_views	double	NOT NULL	Views of Movie
m_duration	int	NOT NULL	Duration of Movie
m_description	nvarchar (300)	NOT NULL	Summary about Movie
m_trailer	nvarchar (40)	NOT NULL	Path to Trailer of Movie
m_thumbnail	nvarchar (40)	NOT NULL	Path to Thumbnail of Movie
s_id	int	FOREIGN KEY	Id of Studio
country_id	int	FOREIGN KEY	Id of Country

Studio			
Fleld Name	**Data Type**	**Contraint**	**Description**
s_id	int	PRIMARY KEY	ID of Studio
s_name	nvarchar (20)	NOT NULL	Name of Studio

Actor			
Fleld Name	**Data Type**	**Contraint**	**Description**
a_id	int	PRIMARY KEY	ID of Actor
a_name	nvarchar (20)	NOT NULL	Name of Actor
a_age	int	NOT NULL	Age of Actor

Genre			
Fleld Name	**Data Type**	**Contraint**	**Description**
g_id	int	PRIMARY KEY	ID of Genre
g_name	nvarchar (20)	NOT NULL	Name of Genre

Director			
Fleld Name	**Data Type**	**Contraint**	**Description**
d_id	int	PRIMARY KEY	ID of Director
d_name	nvarchar (20)	NOT NULL	Name of Director
d_age	int	NOT NULL	Age of Director

Country			
Fleld Name	**Data Type**	**Contraint**	**Description**
country_id	int	PRIMARY KEY	ID of Country
country_name	nvarchar (20)	NOT NULL	Name of Country

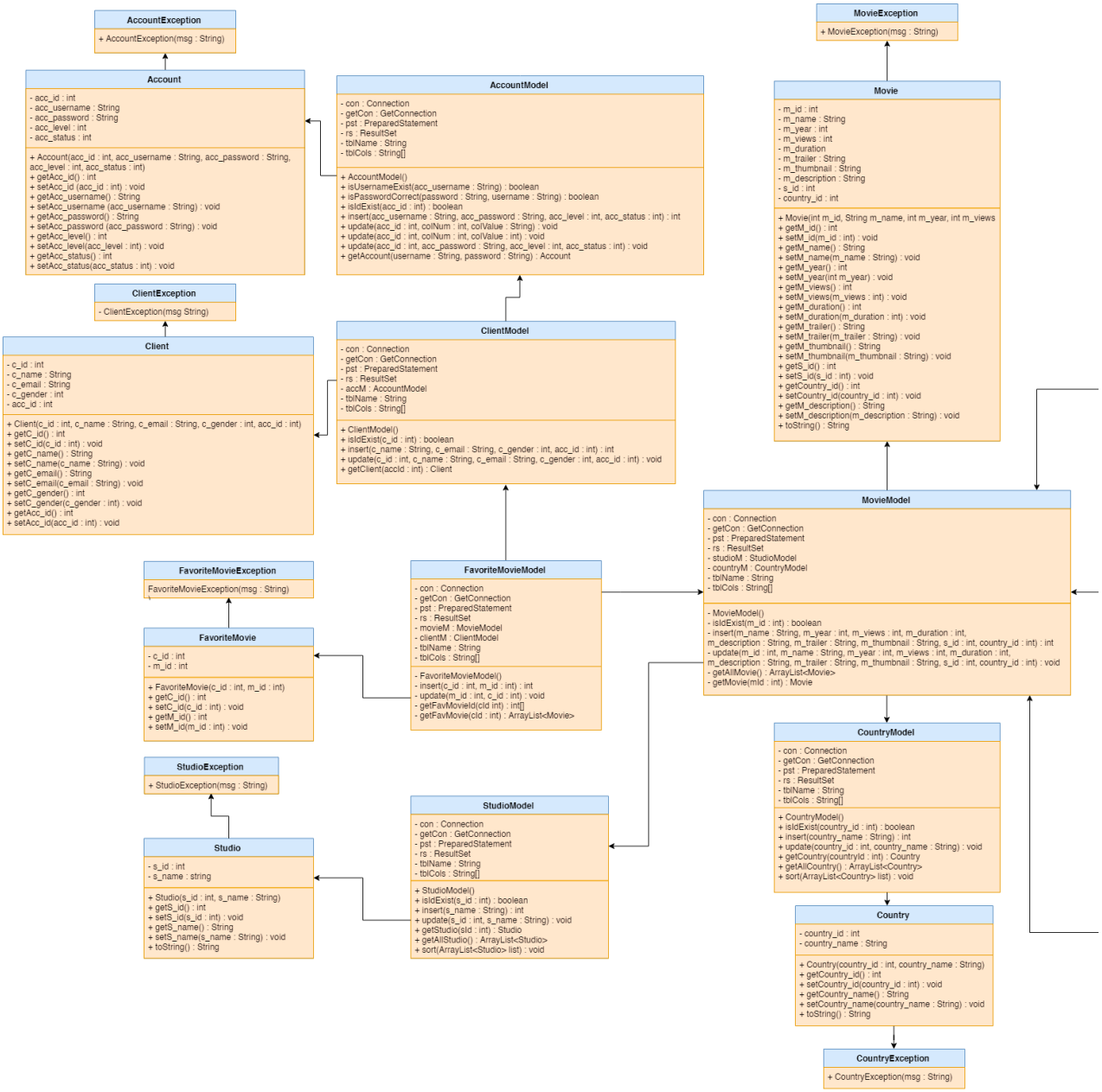
FavoriteMovie			
Fleld Name	**Data Type**	**Contraint**	**Description**
c_id	int	PRIMARY KEY, FORGEIN KEY	ID of Client
m_id	int	PRIMARY KEY, FORGEIN KEY	ID of Movie

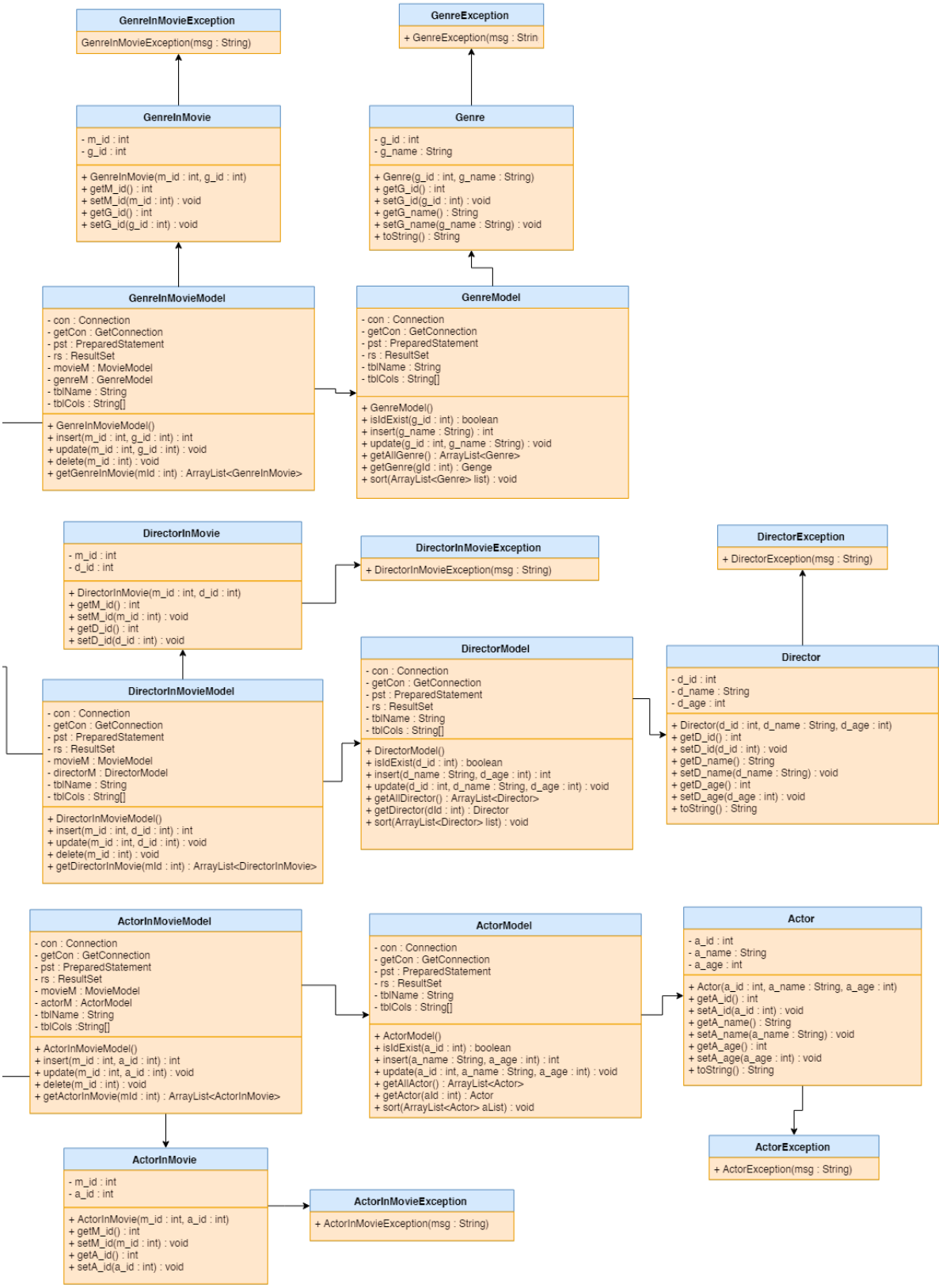
ActorInMovie			
Fleld Name	**Data Type**	**Contraint**	**Description**
m_id	int	PRIMARY KEY, FORGEIN KEY	ID of Movie
a_id	int	PRIMARY KEY, FORGEIN KEY	ID of Actor

DirectorInMovie			
Fleld Name	**Data Type**	**Contraint**	**Description**
m_id	int	PRIMARY KEY, FORGEIN KEY	ID of Movie
d_id	int	PRIMARY KEY, FORGEIN KEY	ID of Director

GenreInMovie			
Fleld Name	**Data Type**	**Contraint**	**Description**
m_id	int	PRIMARY KEY, FORGEIN KEY	ID of Movie
g_id	int	PRIMARY KEY, FORGEIN KEY	ID of Genre

3. Class diagram




GenreInMovieException
GenreInMovieException(msg : String)
GenreException
+ GenreException(msg : Strin
GenreInMovie
- m_id : int
- g_id : int
+ GenreInMovie(m_id : int, g_id : int)
+ getM_id() : int
+ setM_id(m_id : int) : void
+ getG_id() : int
+ setG_id(g_id : int) : void
Genre
- g_id : int
- g_name : String
+ Genre(g_id : int, g_name : String)
+ getG_id() : int
+ setG_id(g_id : int) : void
+ getG_name() : String
+ setG_name(g_name : String) : void
+ toString() : String
GenreInMovieModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- movieM : MovieModel
- genreM : GenreModel
- tblName : String
- tblCols : String[]
+ GenreInMovieModel()
+ insert(m_id : int, g_id : int) : int
+ update(m_id : int, g_id : int) : void
+ delete(m_id : int) : void
+ getGenreInMovie(mId : int) : ArrayList<GenreInMovie>
GenreModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- tblName : String
- tblCols : String[]
+ GenreModel()
+ isIdExist(g_id : int) : boolean
+ insert(g_name : String) : int
+ update(g_id : int, g_name : String) : void
+ getAllGenre() : ArrayList<Genre>
+ getGenre(gId : int) : Genge
+ sort(ArrayList<Genre> list) : void
DirectorInMovie
- m_id : int
- d_id : int
+ DirectorInMovie(m_id : int, d_id : int)
+ getM_id() : int
+ setM_id(m_id : int) : void
+ getD_id() : int
+ setD_id(d_id : int) : void
DirectorInMovieException
+ DirectorInMovieException(msg : String)
DirectorException
+ DirectorException(msg : String)
DirectorInMovieModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- movieM : MovieModel
- directorM : DirectorModel
- tblName : String
- tblCols : String[]
+ DirectorInMovieModel()
+ insert(m_id : int, d_id : int) : int
+ update(m_id : int, d_id : int) : void
+ delete(m_id : int) : void
+ getDirectorInMovie(mId : int) : ArrayList<DirectorInMovie>
DirectorModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- tblName : String
- tblCols : String[]
+ DirectorModel()
+ isIdExist(d_id : int) : boolean
+ insert(d_name : String, d_age : int) : int
+ update(d_id : int, d_name : String, d_age : int) : void
+ getAllDirector() : ArrayList<Director>
+ getDirector(dId : int) : Director
+ sort(ArrayList<Director> list) : void
Director
- d_id : int
- d_name : String
- d_age : int
+ Director(d_id : int, d_name : String, d_age : int)
+ getD_id() : int
+ setD_id(d_id : int) : void
+ getD_name() : String
+ setD_name(d_name : String) : void
+ getD_age() : int
+ setD_age(d_age : int) : void
+ toString() : String
ActorInMovieModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- movieM : MovieModel
- actorM : ActorModel
- tblName : String
- tblCols :String[]
+ ActorInMovieModel()
+ insert(m_id : int, a_id : int) : int
+ update(m_id : int, a_id : int) : void
+ delete(m_id : int) : void
+ getActorInMovie(mId : int) : ArrayList<ActorInMovie>
ActorModel
- con : Connection
- getCon : GetConnection
- pst : PreparedStatement
- rs : ResultSet
- tblName : String
- tblCols : String[]
+ ActorModel()
+ isIdExist(a_id : int) : boolean
+ insert(a_name : String, a_age : int) : int
+ update(a_id : int, a_name : String, a_age : int) : void
+ getAllActor() : ArrayList<Actor>
+ getActor(aId : int) : Actor
+ sort(ArrayList<Actor> aList) : void
Actor
- a_id : int
- a_name : String
- a_age : int
+ Actor(a_id : int, a_name : String, a_age : int)
+ getA_id() : int
+ setA_id(a_id : int) : void
+ getA_name() : String
+ setA_name(a_name : String) : void
+ getA_age() : int
+ setA_age(a_age : int) : void
+ toString() : String
ActorException
+ ActorException(msg : String)
ActorInMovie
- m_id : int
- a_id : int
+ ActorInMovie(m_id : int, a_id : int)
+ getM_id() : int
+ setM_id(m_id : int) : void
+ getA_id() : int
+ setA_id(a_id : int) : void
ActorInMovieException
+ ActorInMovieException(msg : String)

VI. Program Interface

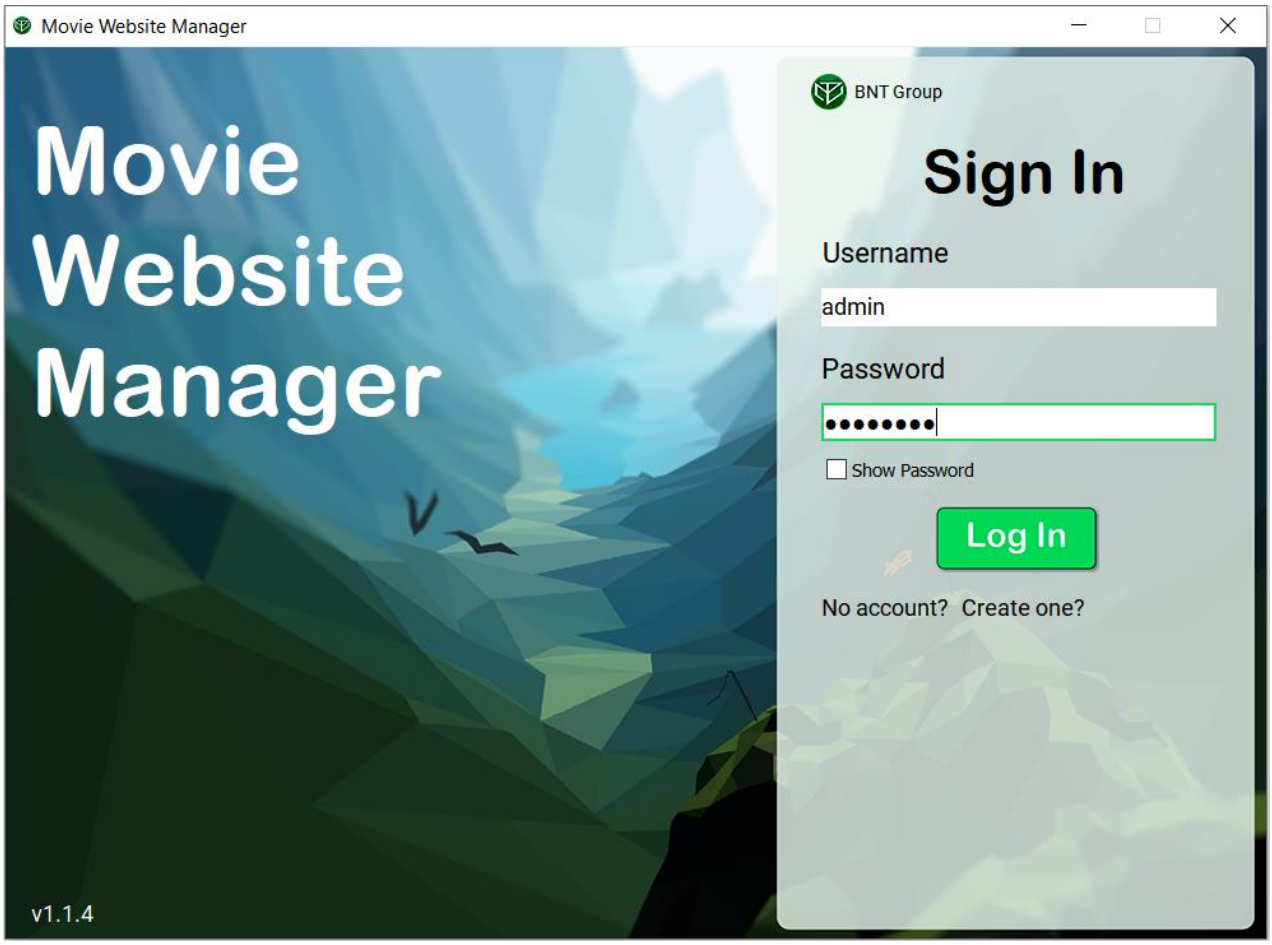


FIGURE 2 LOG IN

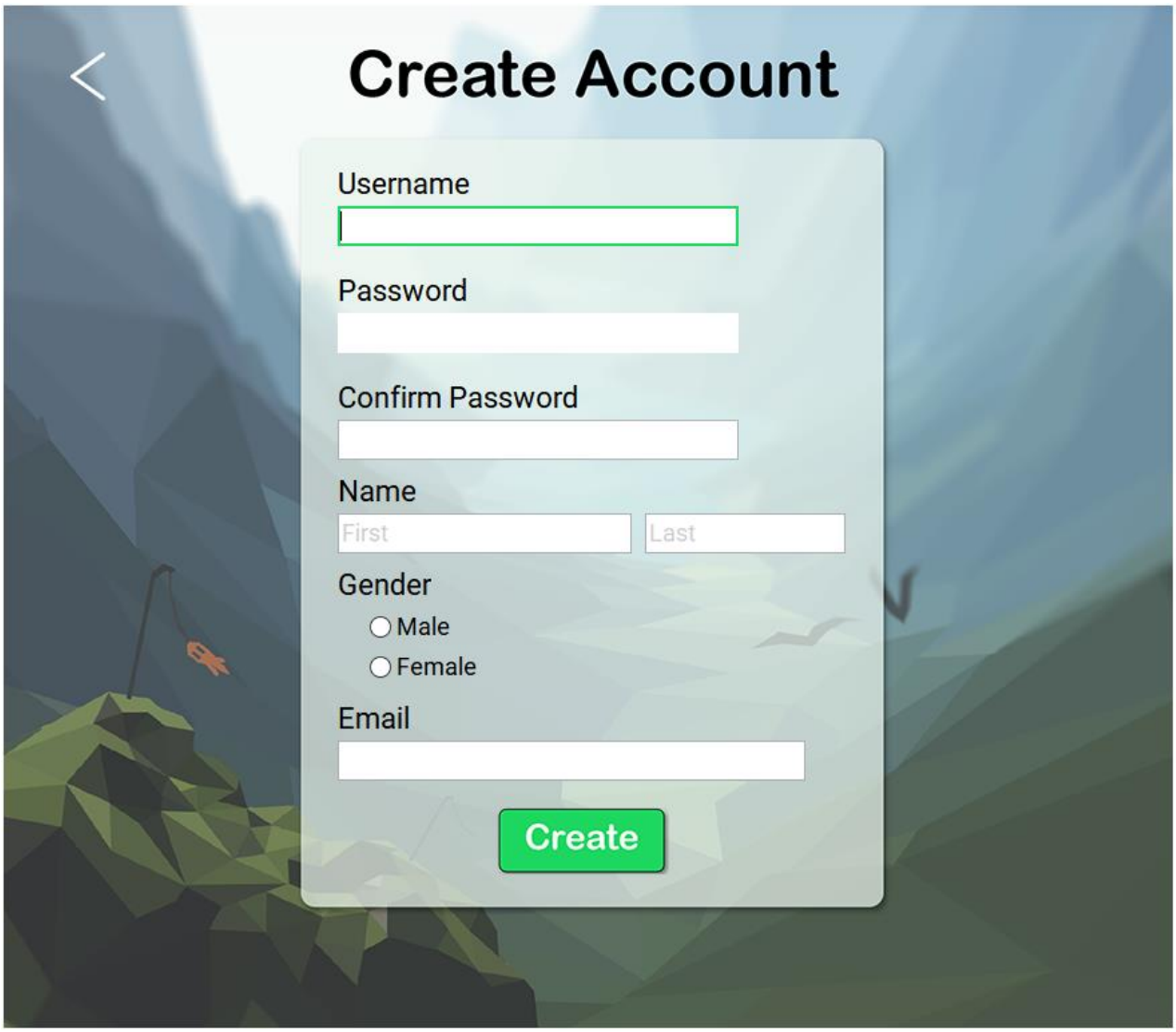


FIGURE 3 CREATE ACCOUNT

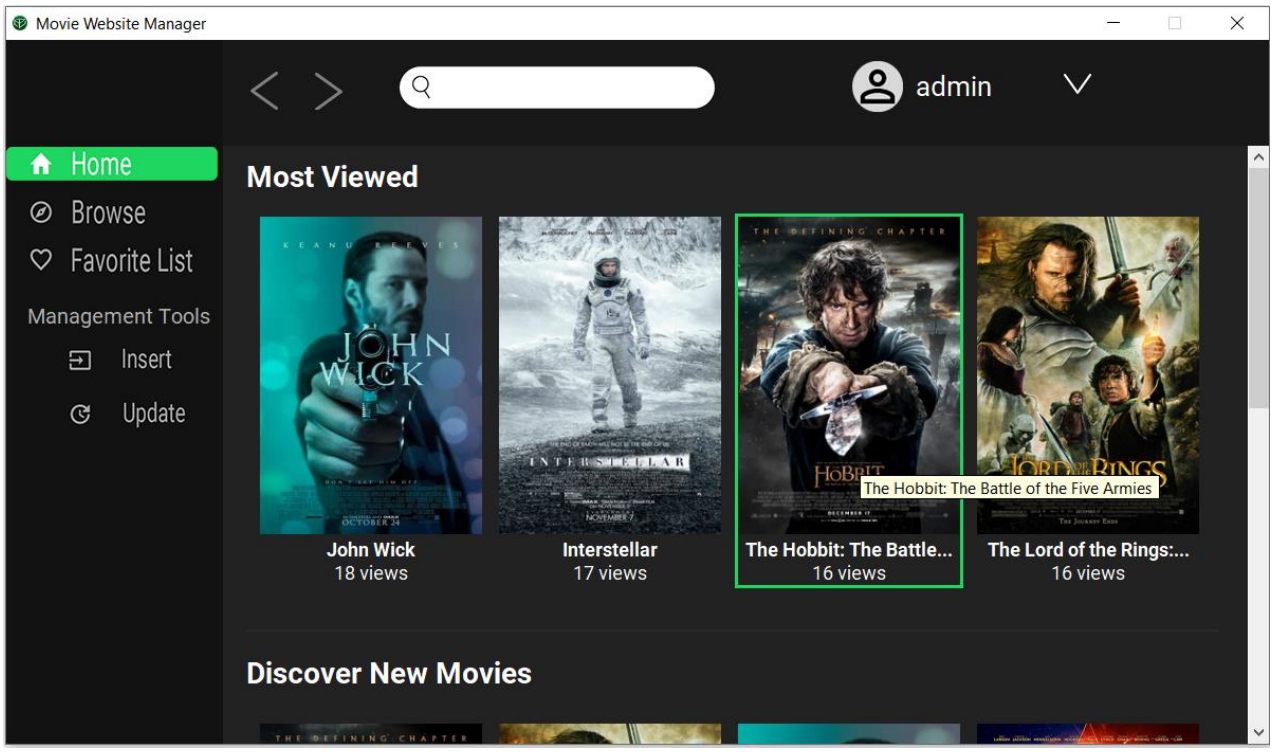


FIGURE 4 MAIN MENU

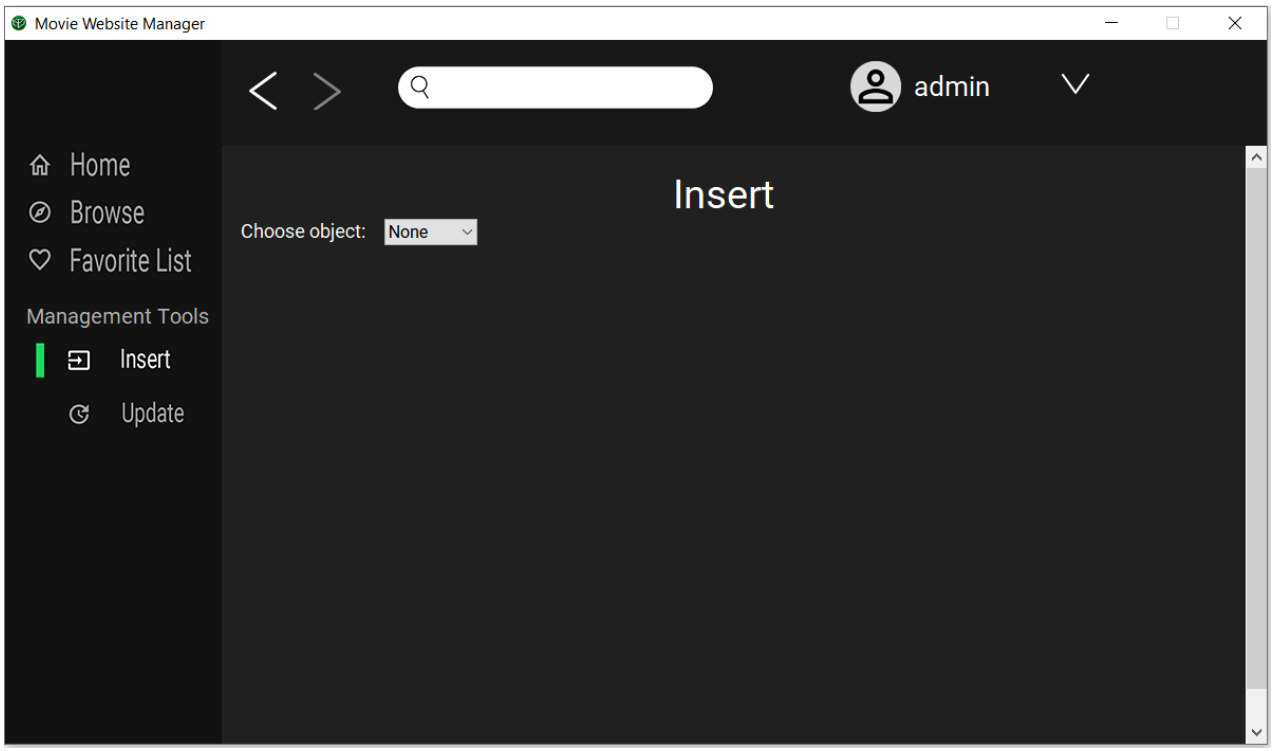


FIGURE 5 INSERT WITH NO OBJECT IS CHOOSING

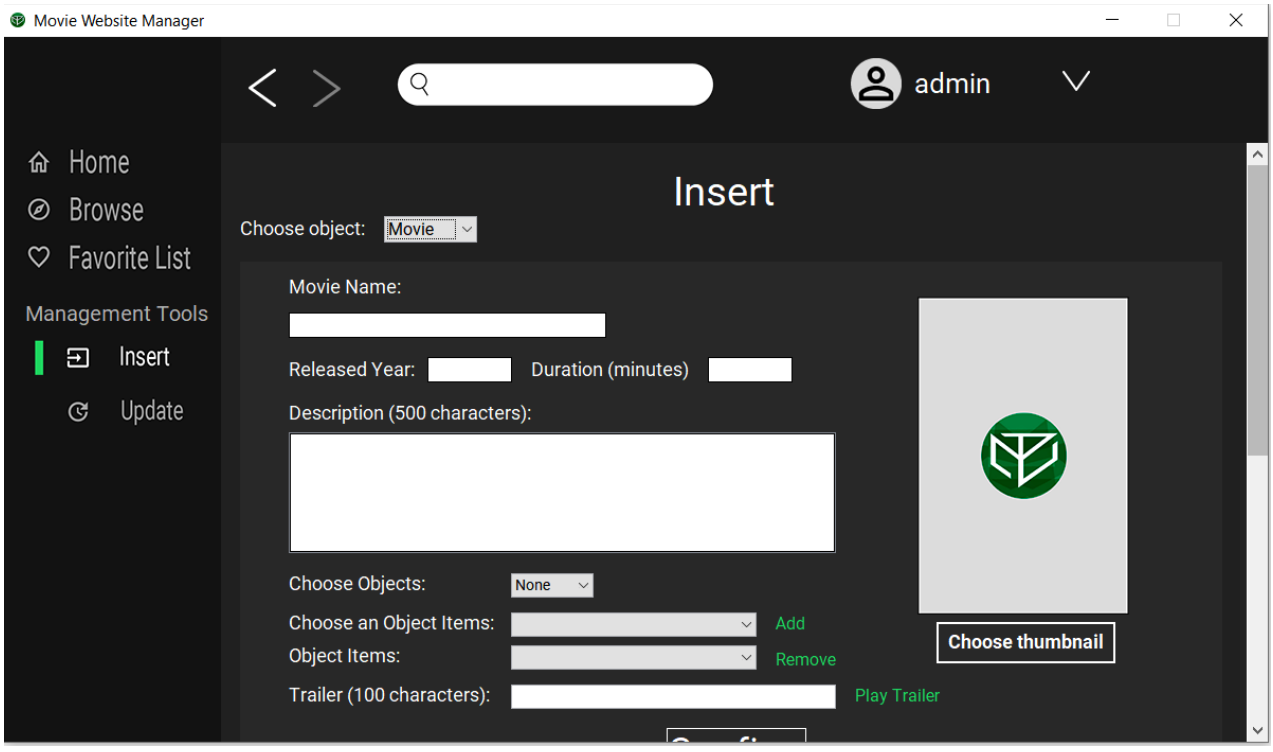


FIGURE 6 INSERT MOVIE

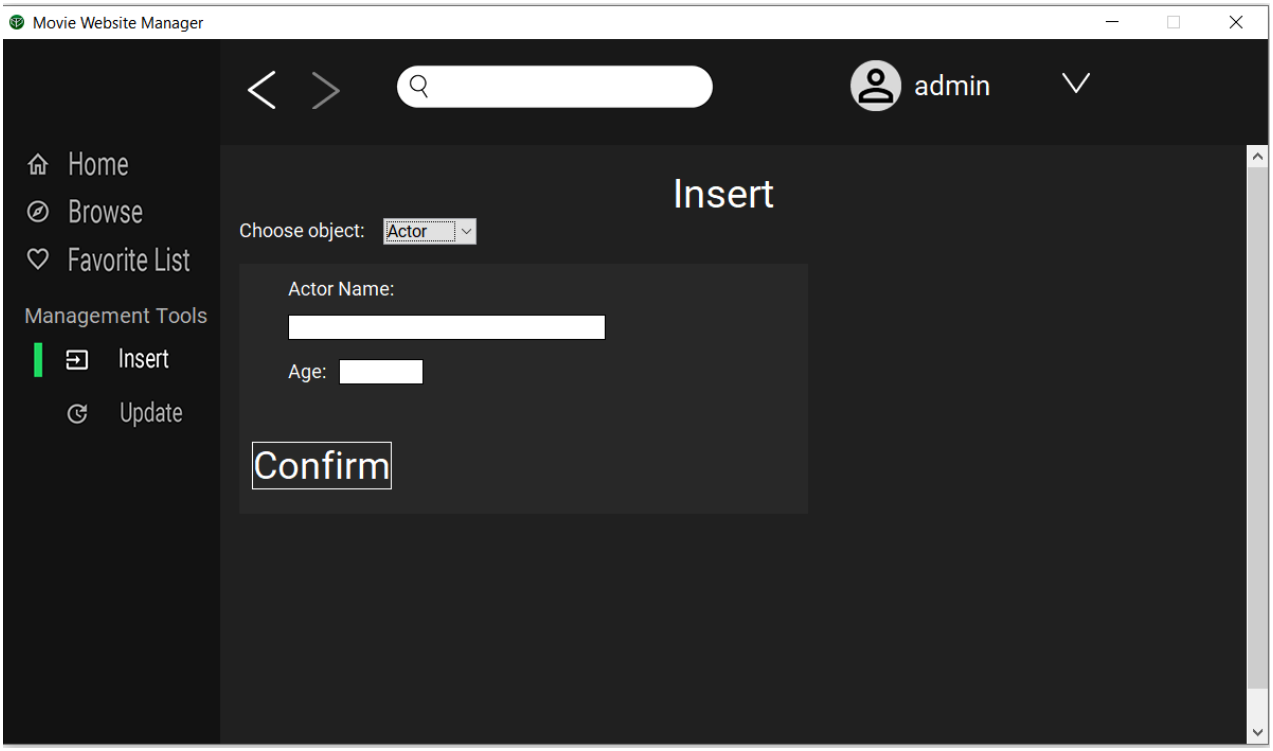


FIGURE 7 INSERT ACTOR / DIRECTOR

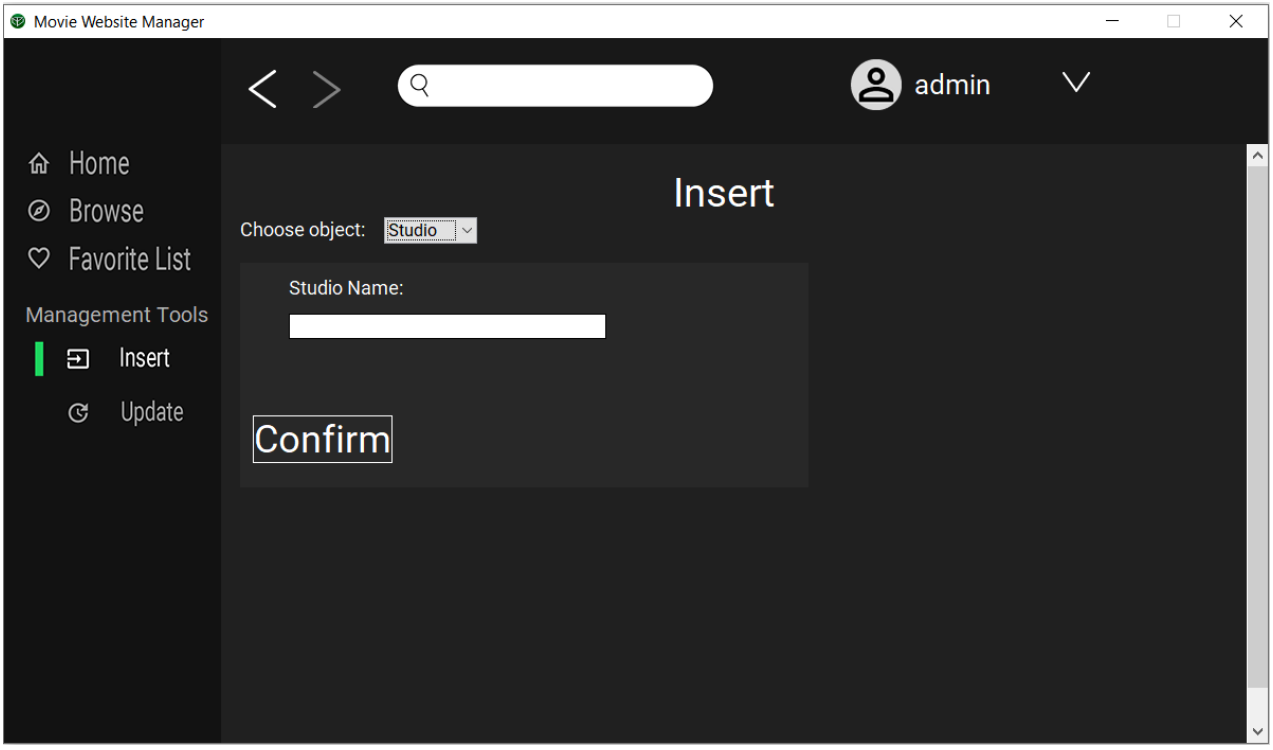


FIGURE 8 INSERT STUDIO / GENRE / COUNTRY

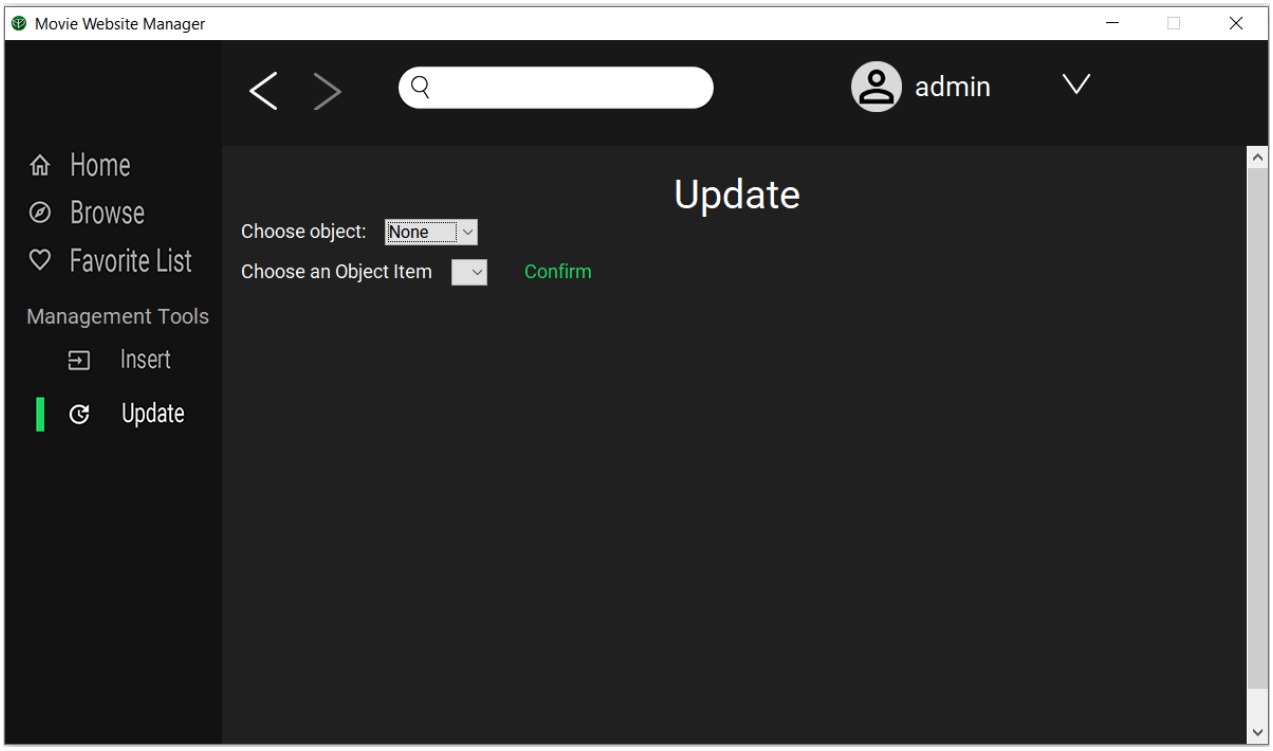


FIGURE 9 UPDATE WITH NO OBJECT IS CHOOSING

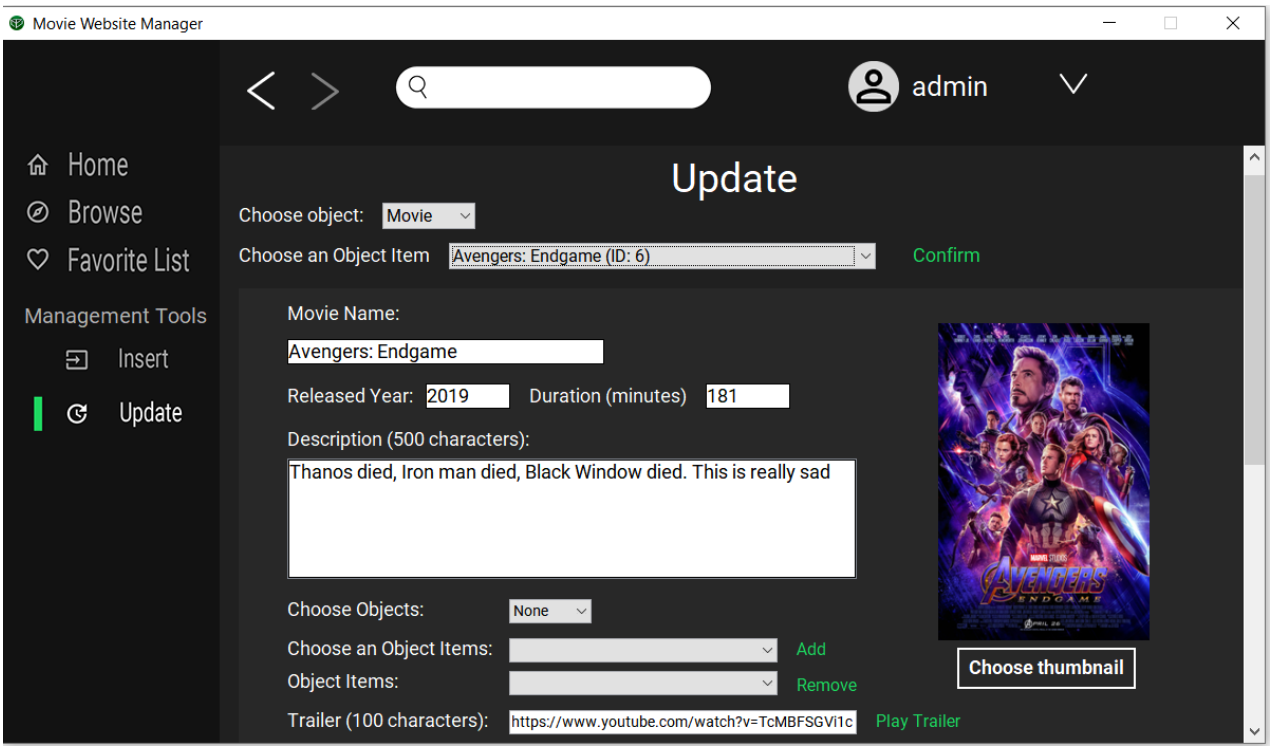


FIGURE 10 UPDATE MOVIE WITH DEMO MOVIE (1)

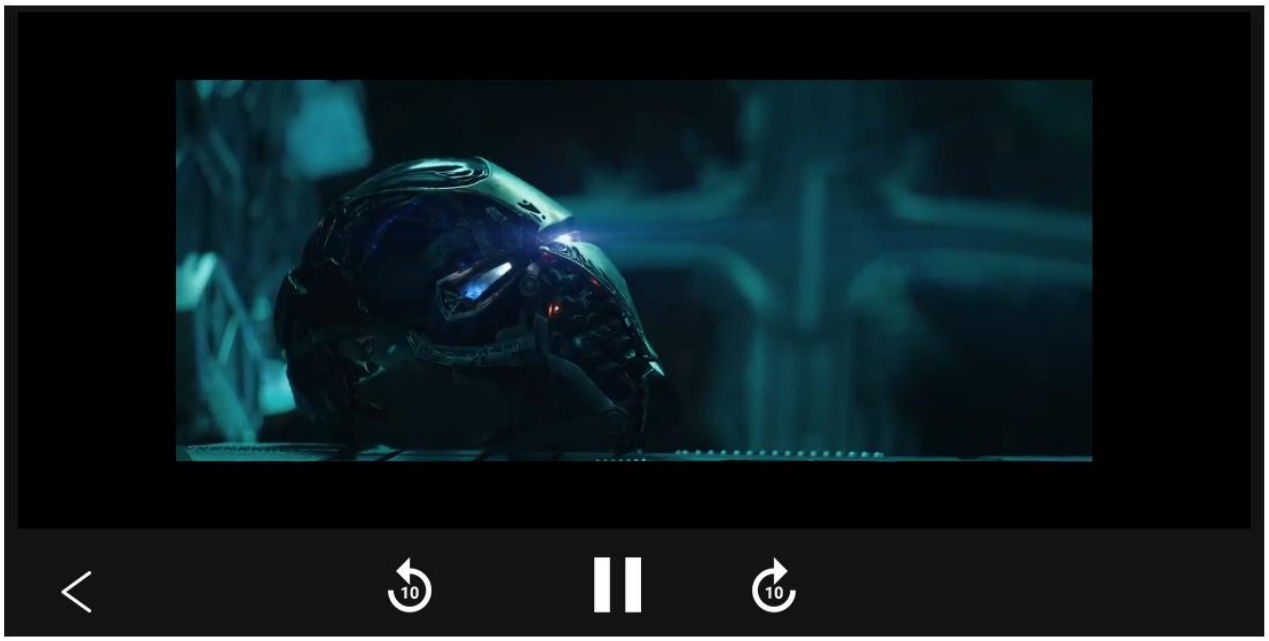


FIGURE 11 UPDATE MOVIE WITH DEMO MOVIE TRAILER(2)

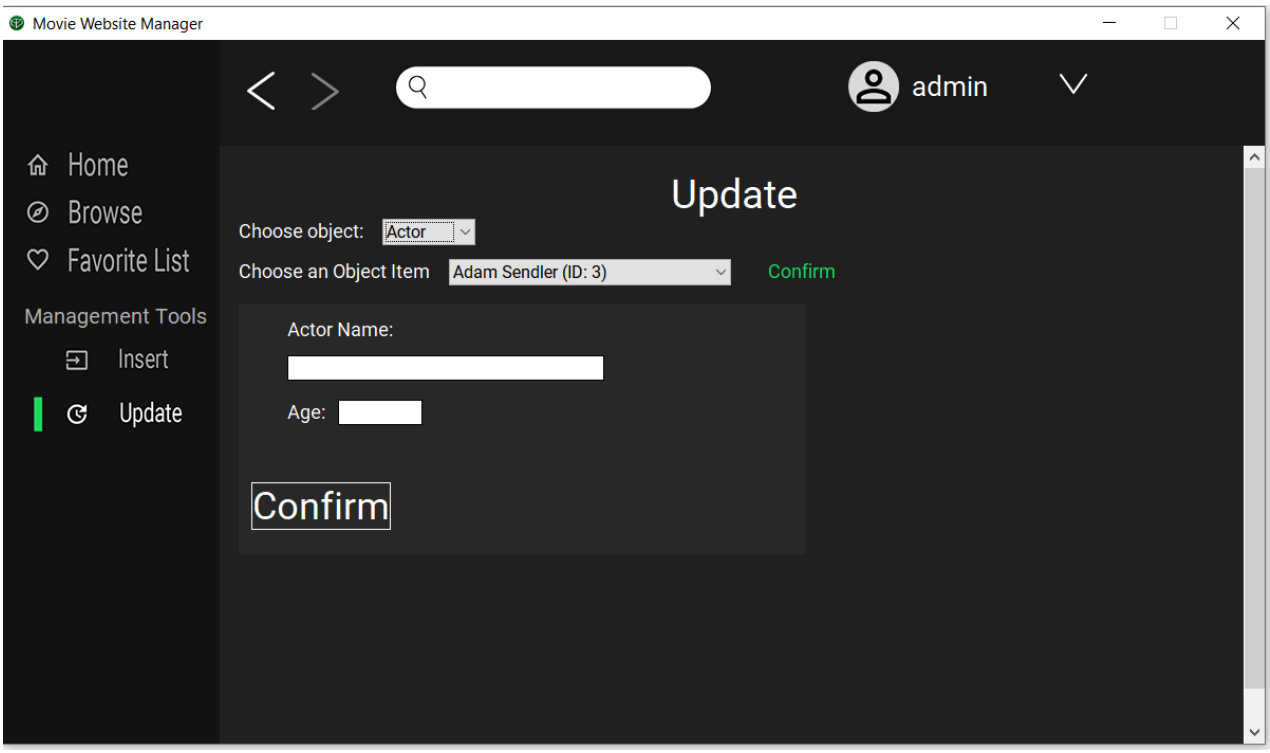


FIGURE 12 UPDATE ACTOR / DIRECTOR

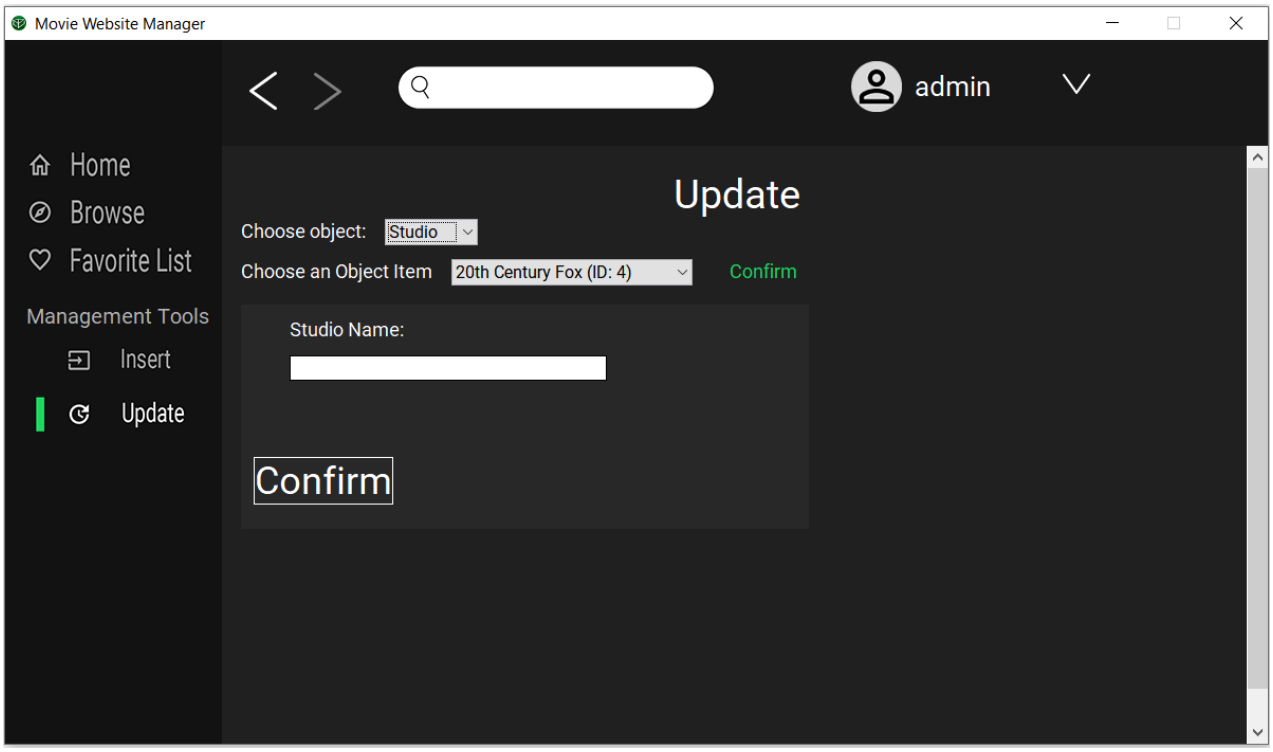


FIGURE 13 UPDATE STUDIO / GENRE / COUNTRY

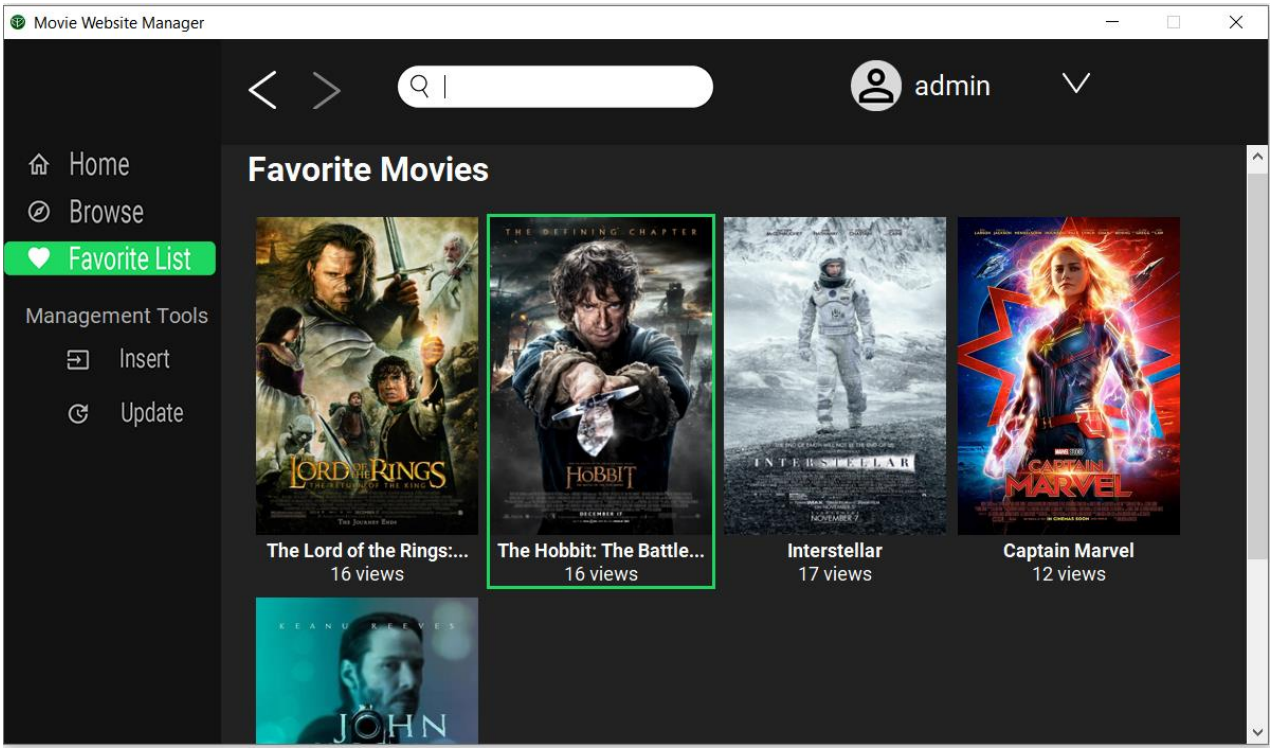


FIGURE 14 FAVORITE LIST

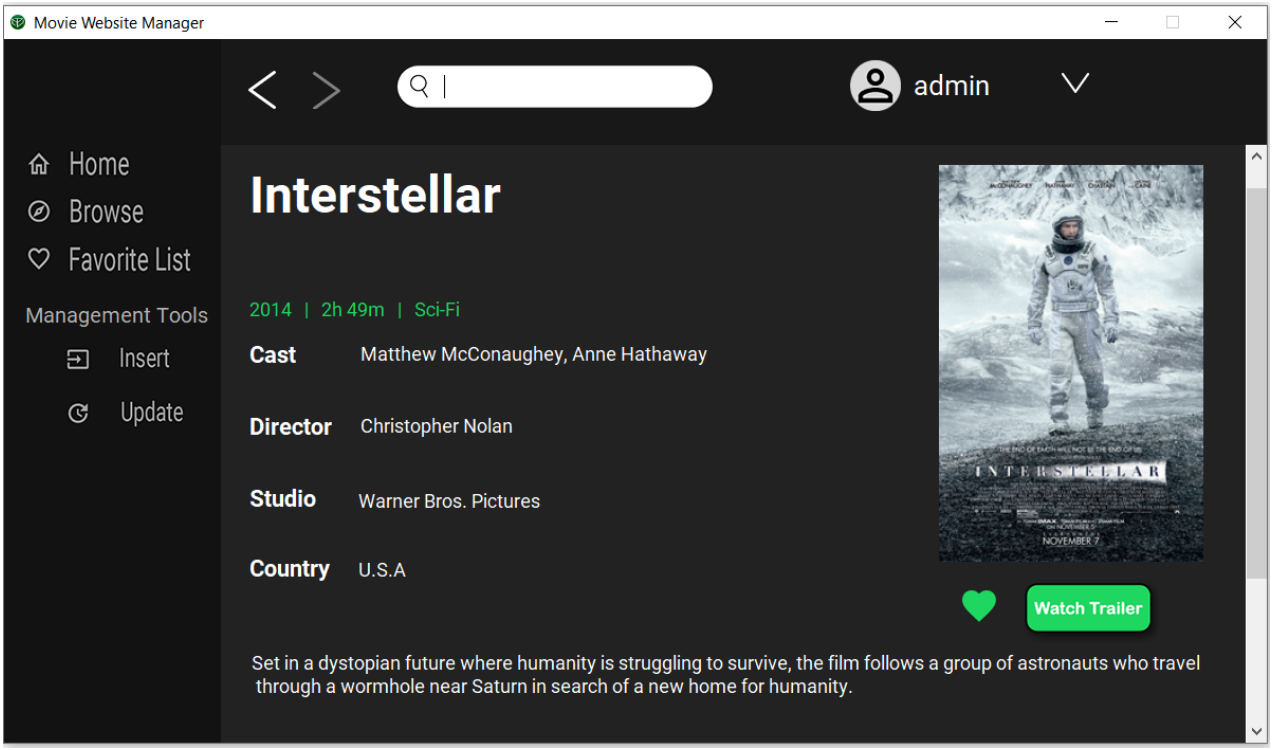


FIGURE 15 MOVIE INFORMATION

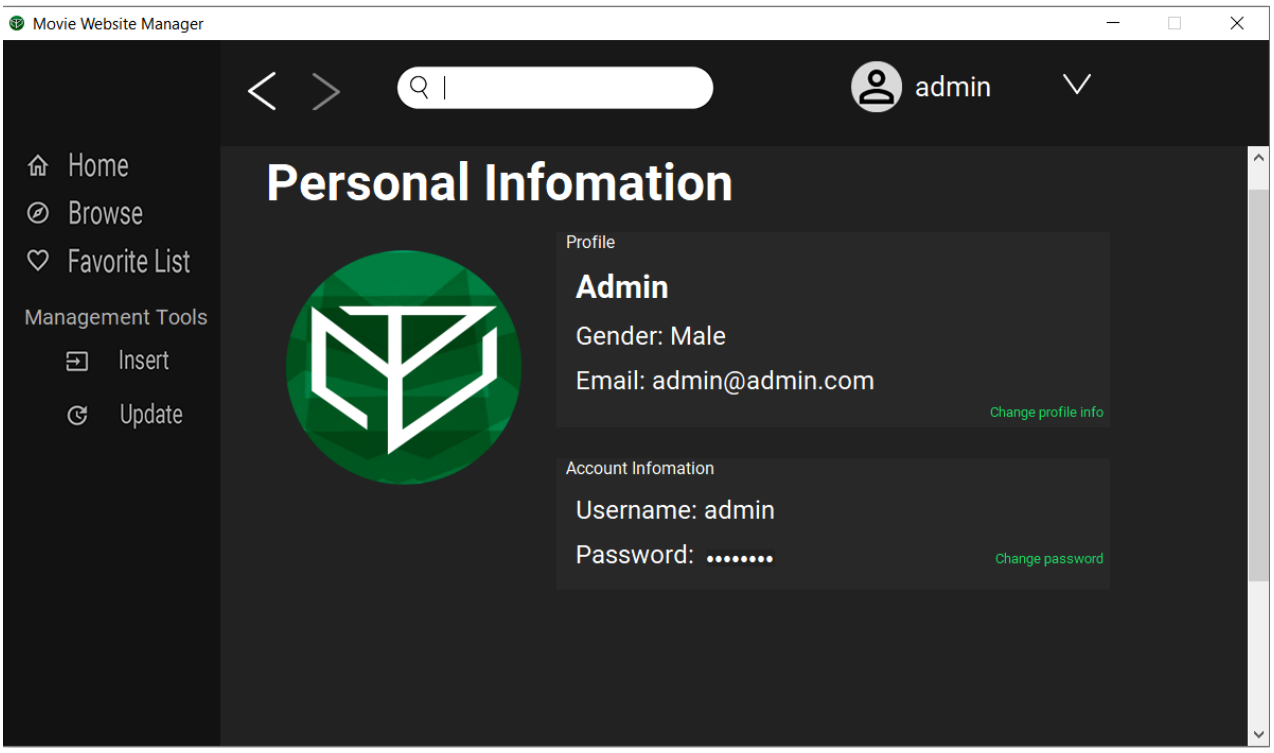


FIGURE 16 PERSONAL IINFORMATION

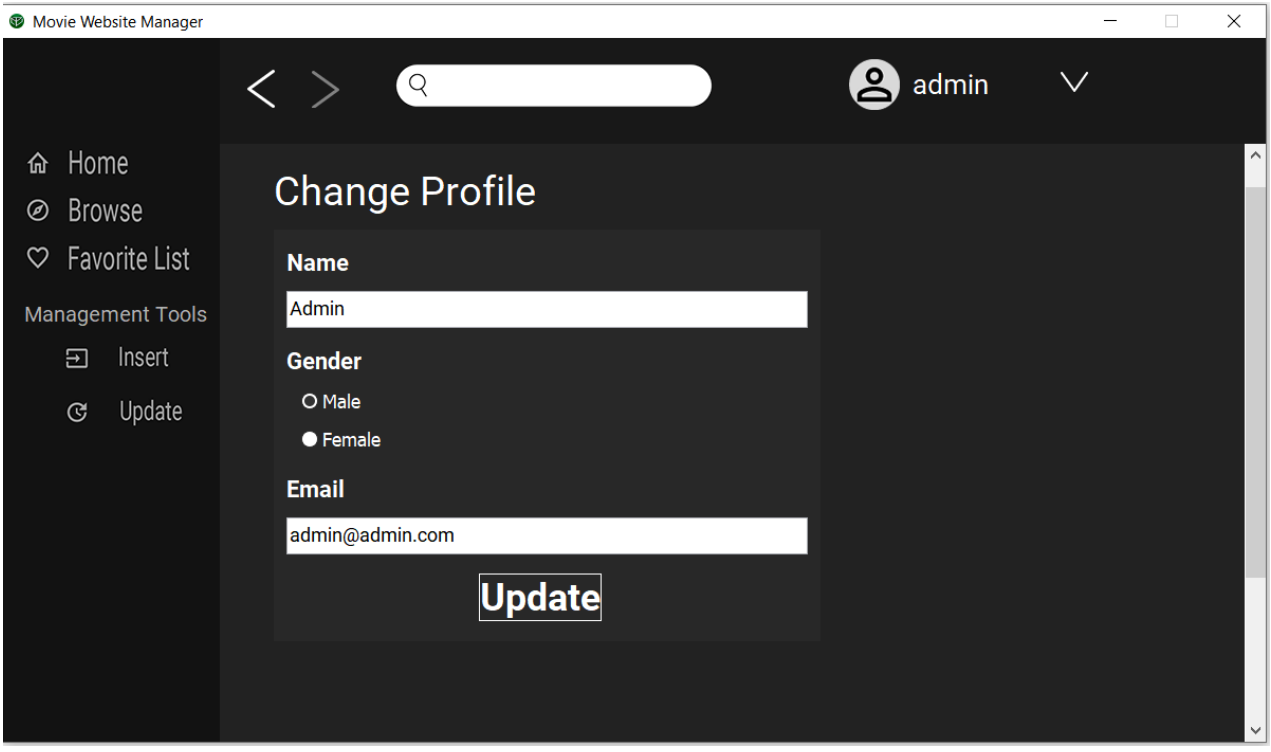


FIGURE 17 UPDATE PROFILE

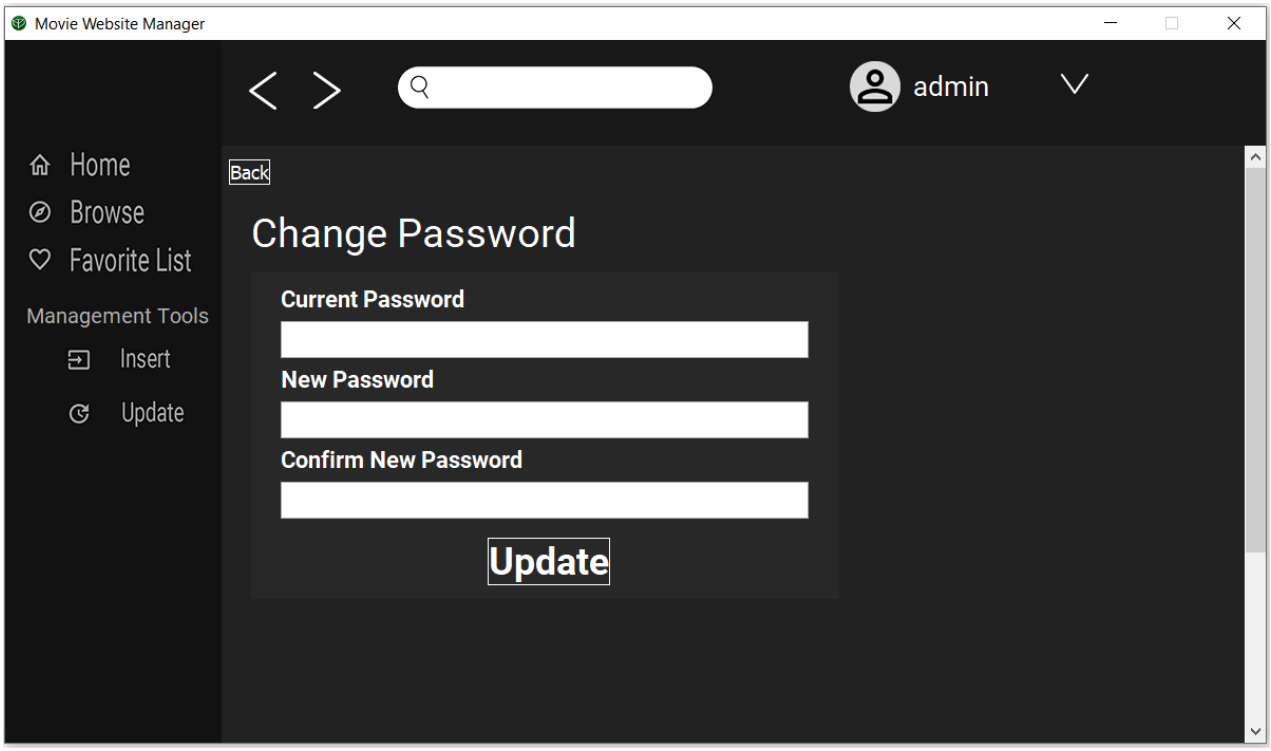


FIGURE 18 UPDATE PASSWORD

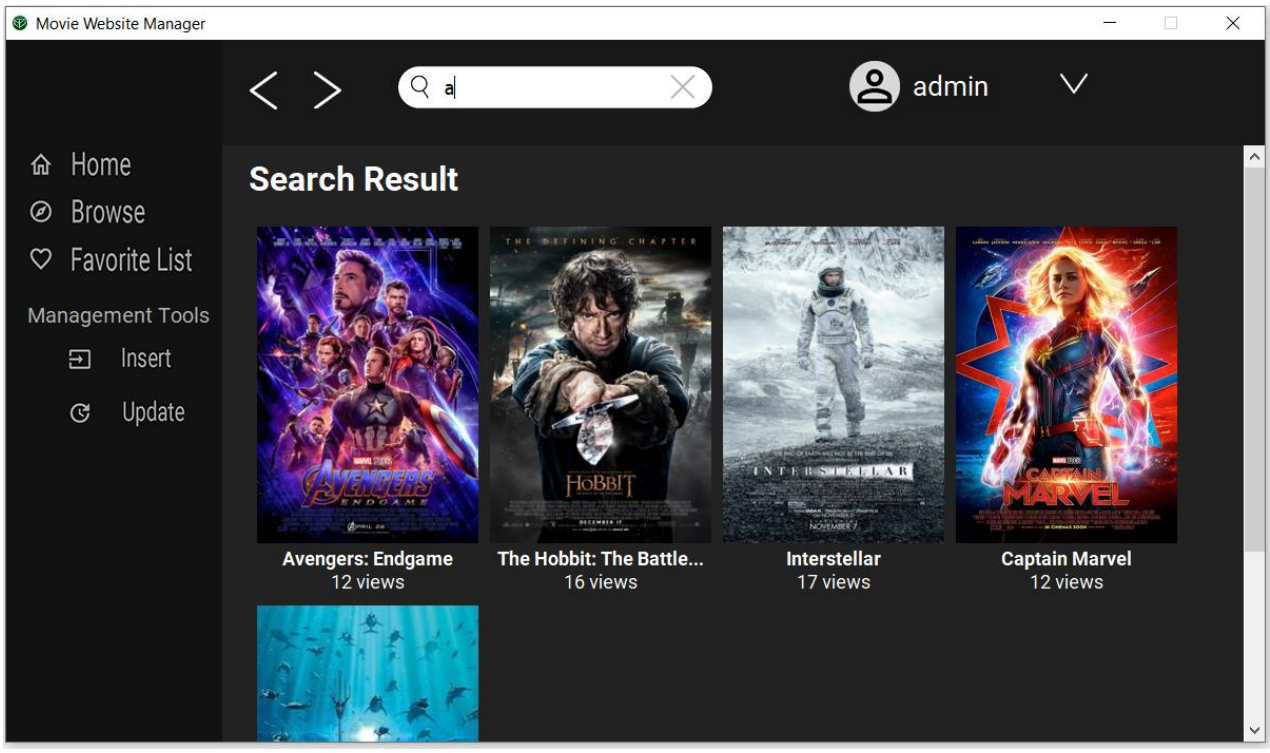


FIGURE 19 SEARCH

VII. Maintenance and Updating

Maintenance:

- Offical project will be stored mainly on our group's chat box on facebook
- We also back up our project to other cloud services like Google Drive, OneDrive every week.
- Functions that are under coding will be stored on group's members devices.

FIGURE 20 CLOUD SERVICES FOR BACKING UP

Updating:

- There are three kinds of update: Big ones, small ones and patches.
  - Big updates: new function(s) or big changes in algorithms.
  - Small ones: include UI update or patches.
  - Patches: bugs fix.
- Our offical project will be updated whenever a new version released.
- New functions will only be added to main project when it's has been tested carefully by our group's testers.
- Changes also written in a log file so that members can track all the changes

VIII. Check List

Assignment	Member	Actual Start Time	ETF[2] (hrs)	Actual End time	Work hours (hrs)	Note
Problem Definition	ThanhKH, NhanDT	8h 28/06/2019	2	9h 28/06/2019	1	Completed
User Requirement and System Requirement	BangNT	8h 28/06/2019	4	12h 28/06/2019	4	Completed
Analysis	ThanhKH	8h 08/07/2019	12	20h 08/07/2019	12	Completed
Database Diagram & Table Structure	ThanhKh, NhanDT	8h 09/07/2019	24	23h50 09/07/2019	16	Completed
Basic GUI Design	ThanhKH, NhanDT, BangNT	8h 11/07/2019	24	23h50 11/07/2019	16	Completed
Exception & Info Classes Coding	ThanhKH, NhanDT, BangNT	8h 12/07/2019	4	12h 12/07/2019	4	Completed
Model Classes Coding	ThanhKH, NhanDT, BangNT	8h 13/07/2019	16	20h 13/07/2019	12	Completed

[2] Estimate Time to Finish

Log In & Create Account Frame	NhanDT	8h 15/07/2019	24	12h 16/07/2019	20	Completed
Main menu Frame	ThanhKH, BangNT	8h 15/07/2019	48	12h 17/07/2019	40	Completed
Detail GUI Design	ThanhKH, NhanDT, BangNT	8h 18/07/2019	48	20h 19/07/2019	36	Completed
Testing & Debugging	BangNT, NhanDT	8h 19/07/2019	12	20h 19/07/2019	12	Completed
Document Finishing	ThanhKH	20h 19/07/2019	4	23h 19/07/2019	3	Completed
Presentation	BangNT	13h 24/07/2019	2	14h15 24/07/2019	1,25	Completed
Implemen-tation	BangNT, NhanDT	8h 25/07/2019	3	10h 25/07/2019	2	Completed