Movie Database

Nicholas Van Beek
Andrew Strelke
Craig Grant

Description:

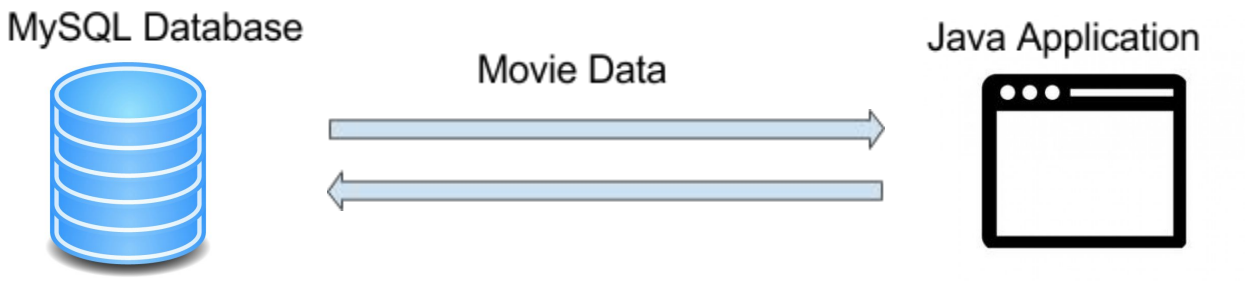
This project provides a database and GUI for storing and retrieving information about Movies. The database is capable of storing many attributes about a movie such as its Title, Duration, Genre, Director, Actors, reviews etc. Since actors and directors can be relevant to many different movies, their information is stored in separate tables and attached to each movie where necessary to avoid storing duplicate information and make it quicker to add actor/director information to a movie. There is also a table of reviews related to each movie and the GUI user can create/view reviews on the movie.

Architecture:

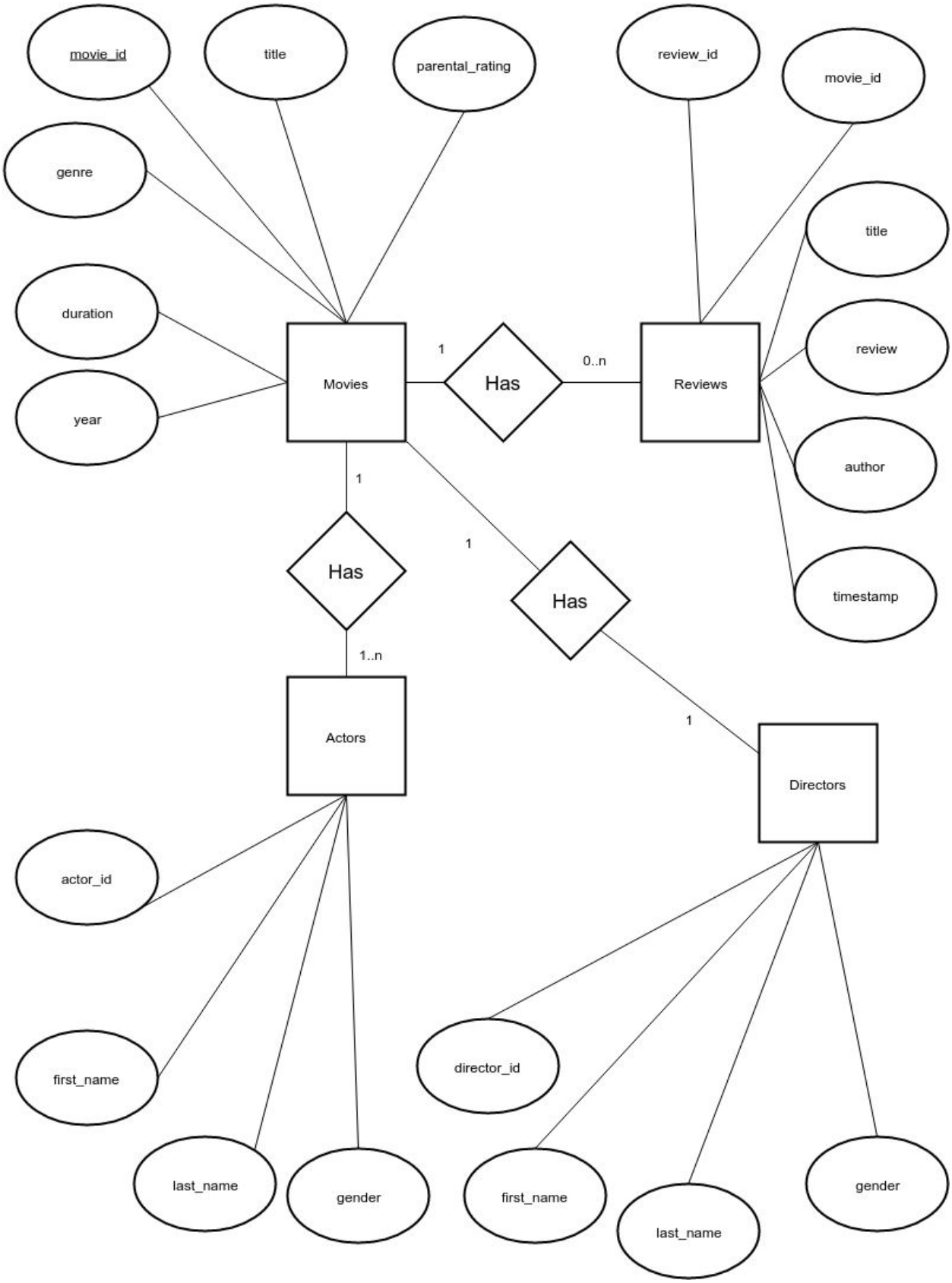
The Movies application has only two parts, the database and Java application. MySQL is the Database Management System used for the Movie database. All of the data is stored in a single database “movies”. In the “movies” database there are a series of tables such as “movies”, “actors” and “directors” where information related to the movies is stored.

Java + JDBC is used to query the database and build the GUI. We wrote a Database.java class as well as a bunch of object classes such as Movie.java and Actor.java which turn the relational data (data queried from the database) into objects that the front end can then iterate through and display in lists. The GUI was built in Java Swing making it cross platform and easy use with JDBC.

We chose this architecture to make installation, setup and portability as simple as possible.

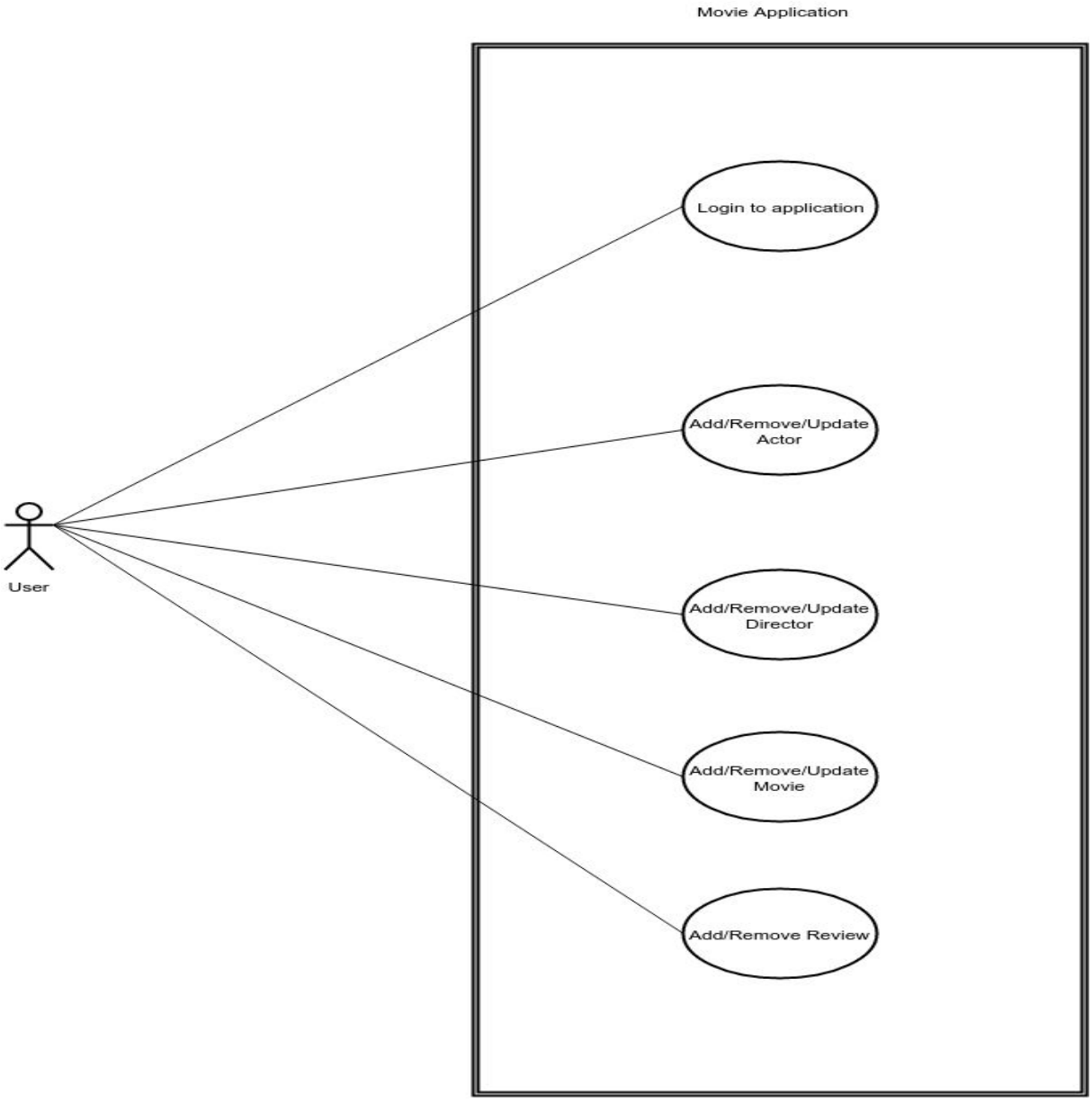


E/R Diagram:


movie_id
title
parental_rating
review_id
movie_id
genre
title
duration
review
Movies
1
Has
0..n
Reviews
year
author
1
timestamp
1
Has
Has
1..n
1
Actors
Directors
actor_id
director_id
first_name
last_name
gender
first_name
last_name
gender

Use Case Diagram:

The application is intended to be a general use application for storing information about movies. The application is only intended for one actor/user to be able to store relevant movie information and alter it for the purpose of storage. Because of this, the use case diagram is really simple since only one actor uses the software/database.



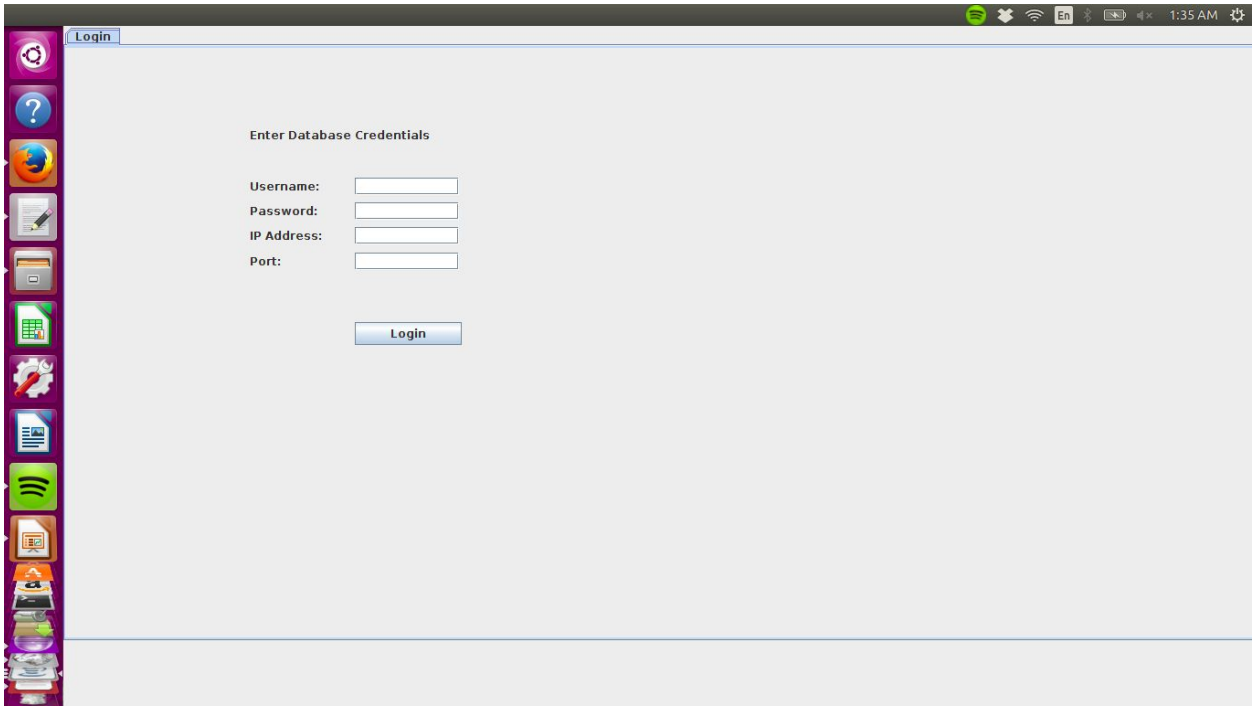
Installation Instructions:

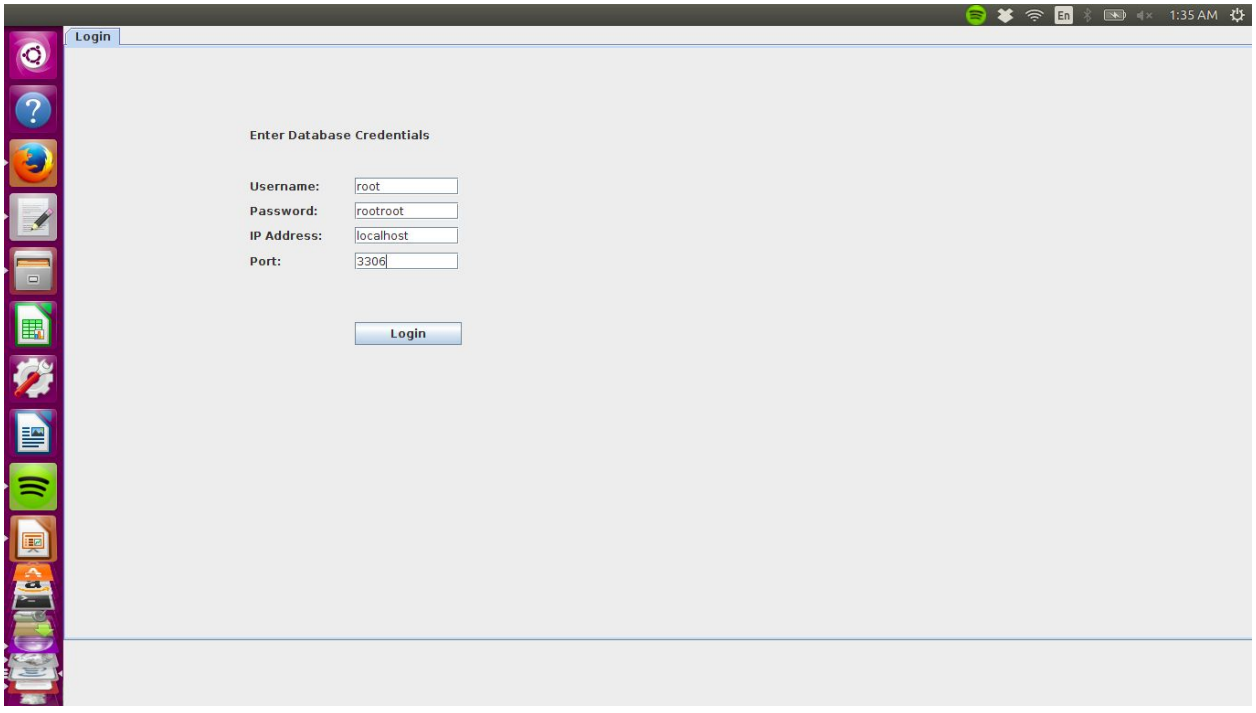
1) Import the movies.sql file into a MySQL database called “movies”

2) Run the Movies.jar file to run the application with. You can use “java -jar Movies.jar” in the terminal/command prompt.

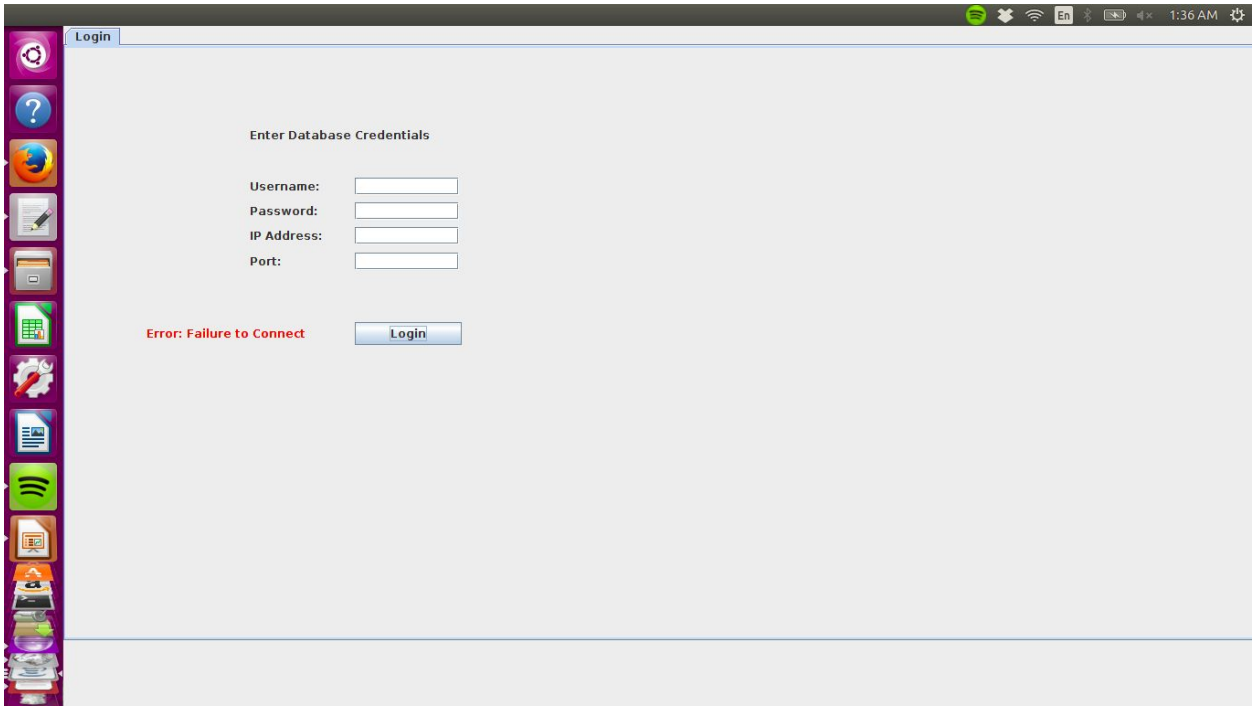
Using the Program:

First login using the ip address, port and username/password of the database.

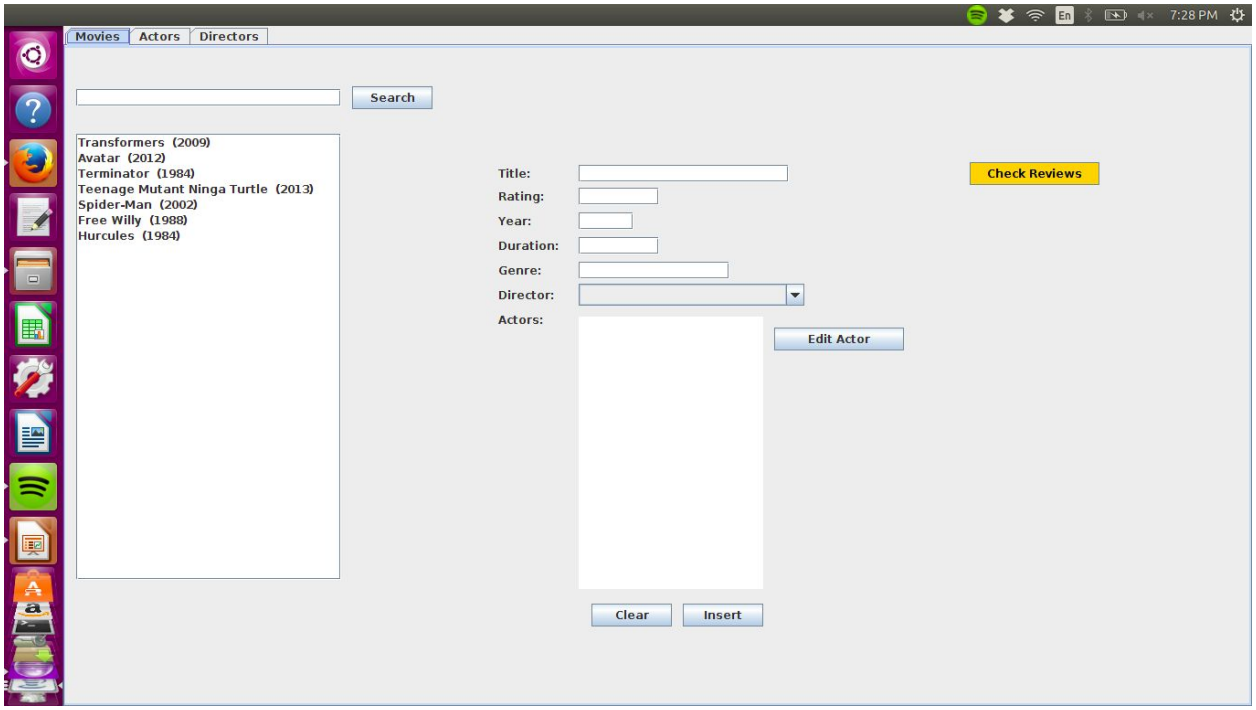




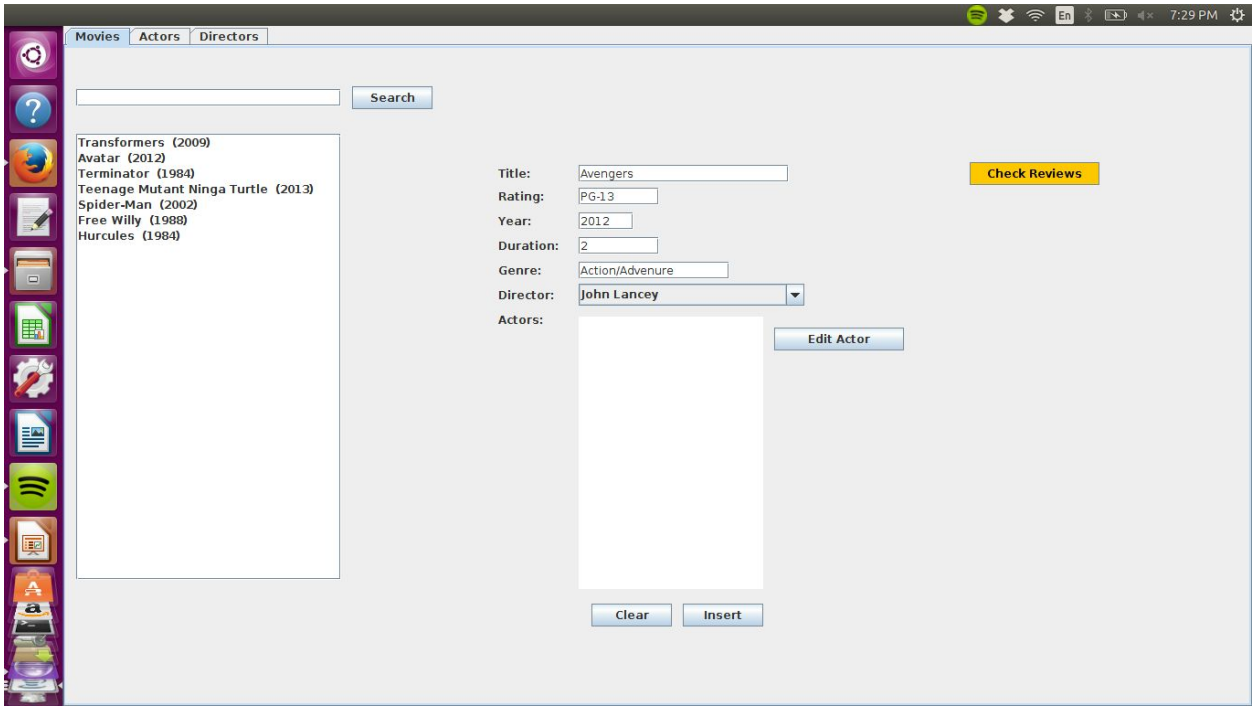
If the user's database creditentials are incorrect, a warning message will display informing the user that the connection failed.



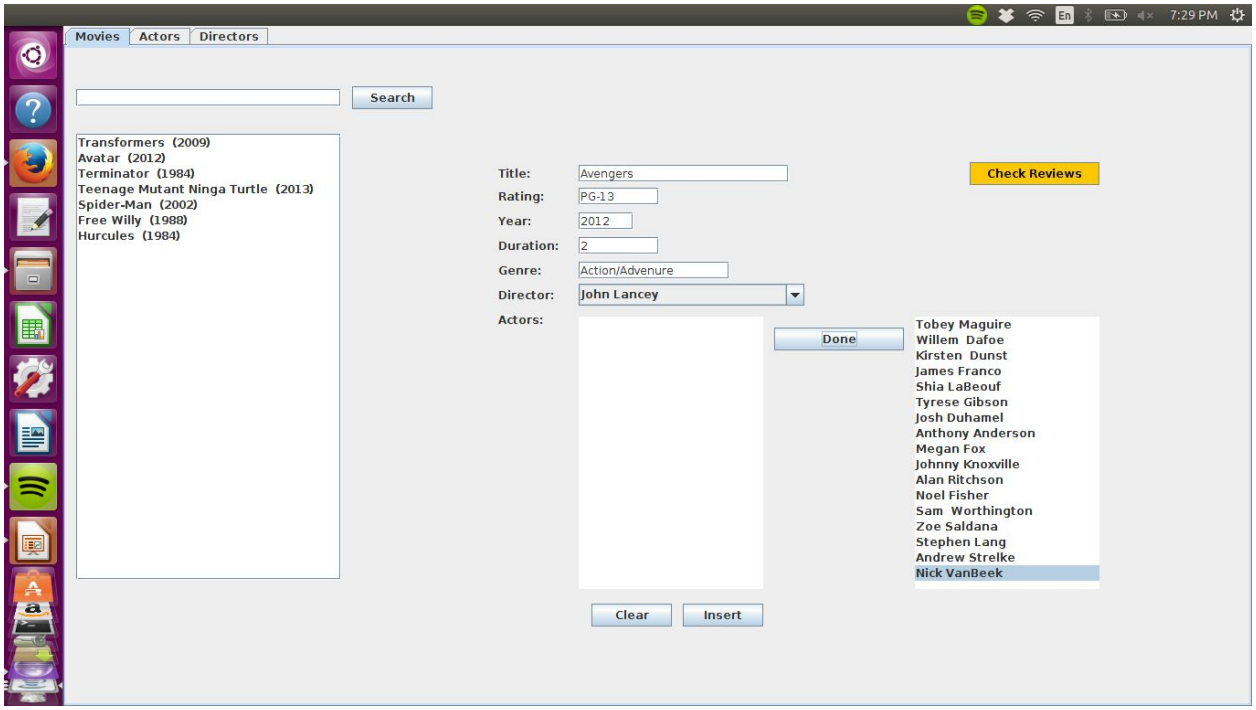
Once the user has logged in, they are brought to the movie tab where they are able to search for a pre-existing movie and view details will be displayed by category



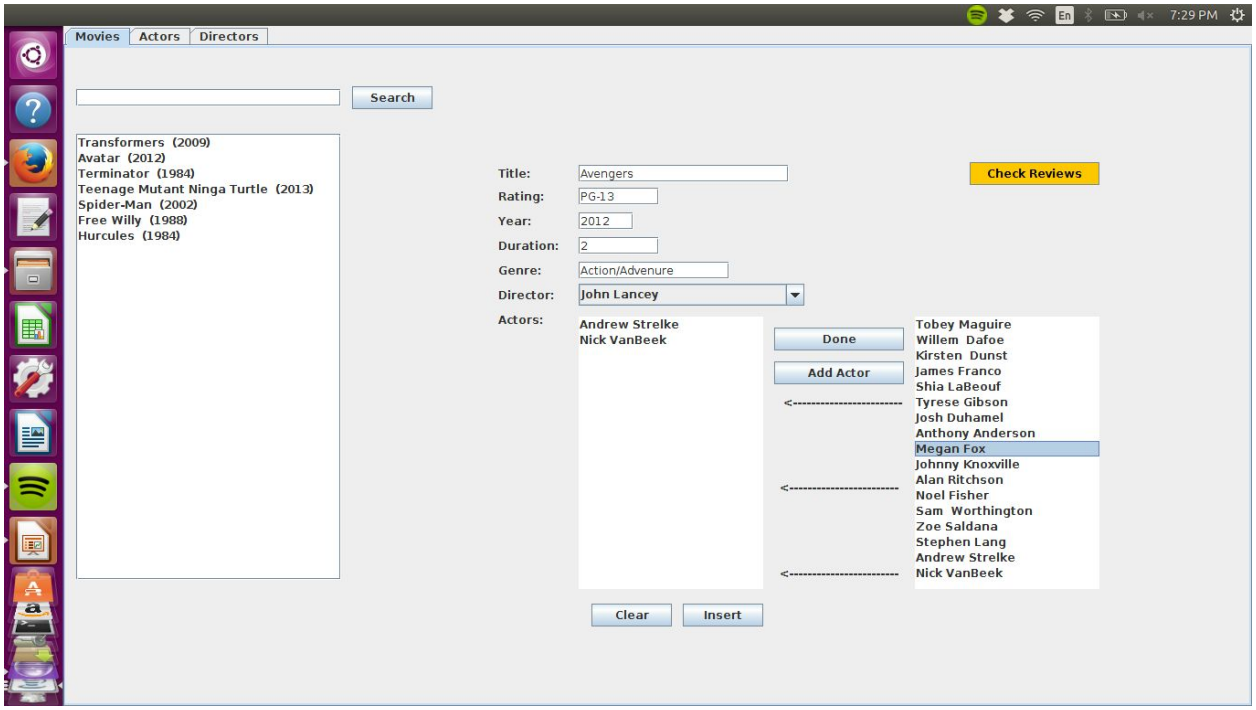
You also have the ability to add a movie into the database by inserting information into its respected text fields.



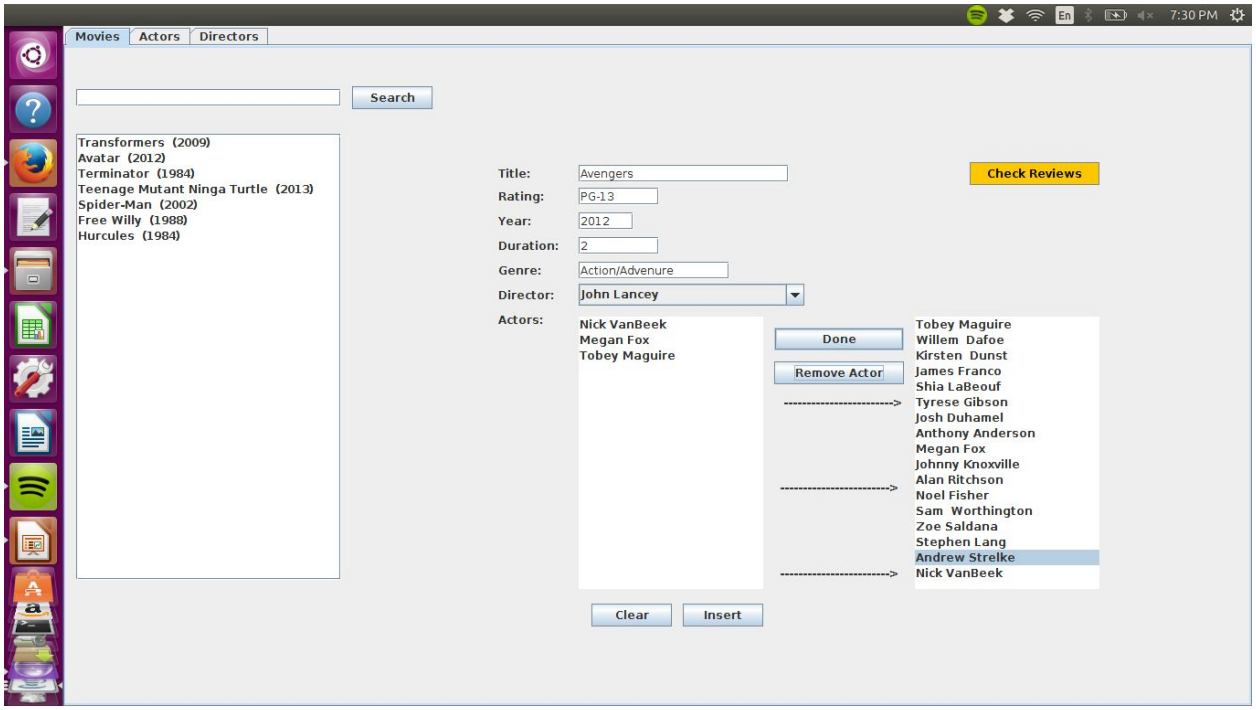
You can associate actors to a given movie by pressing “edit actor”



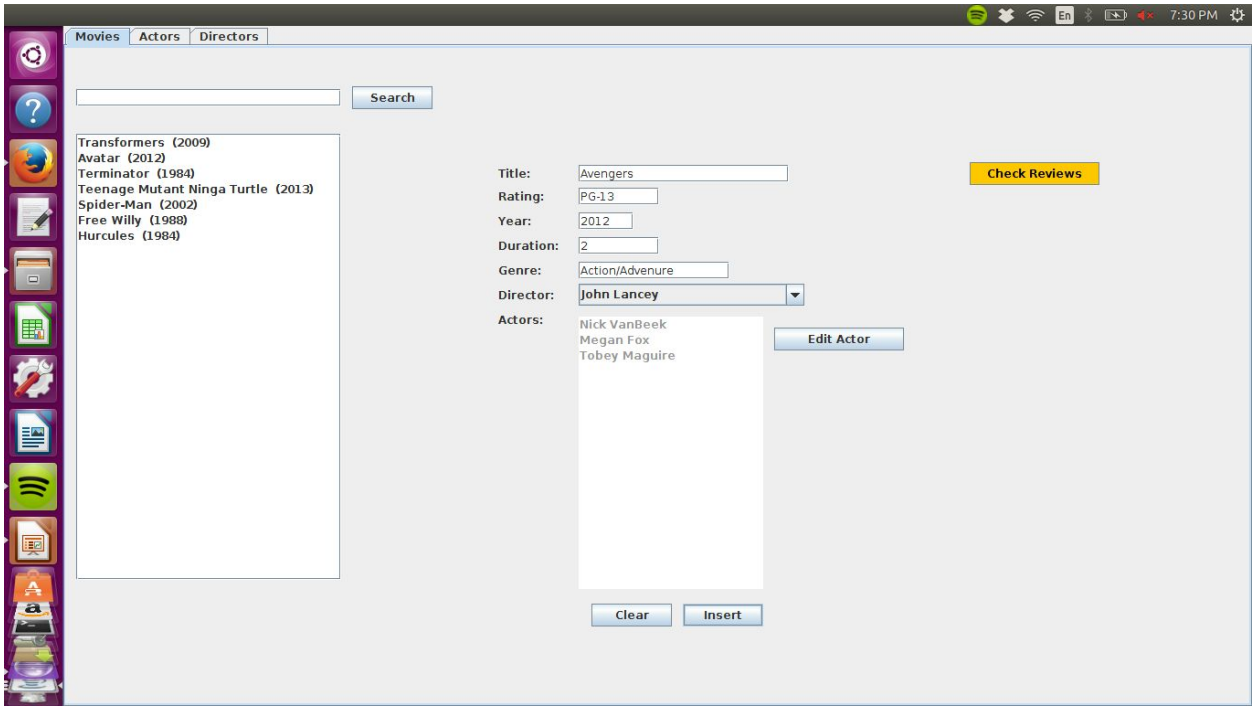
Then click each actor individually, and then add them to the description by clicking “Add Actor.”



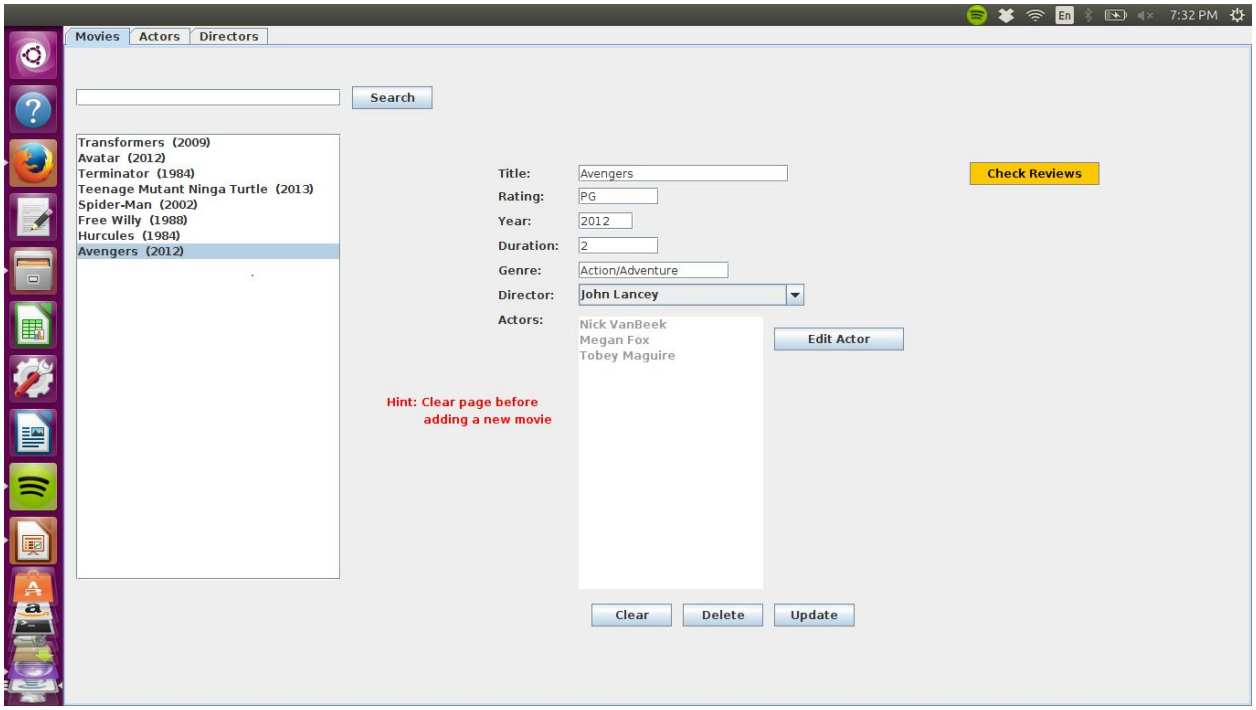
You can also remove actors by clicking on them in the description and clicking remove actor

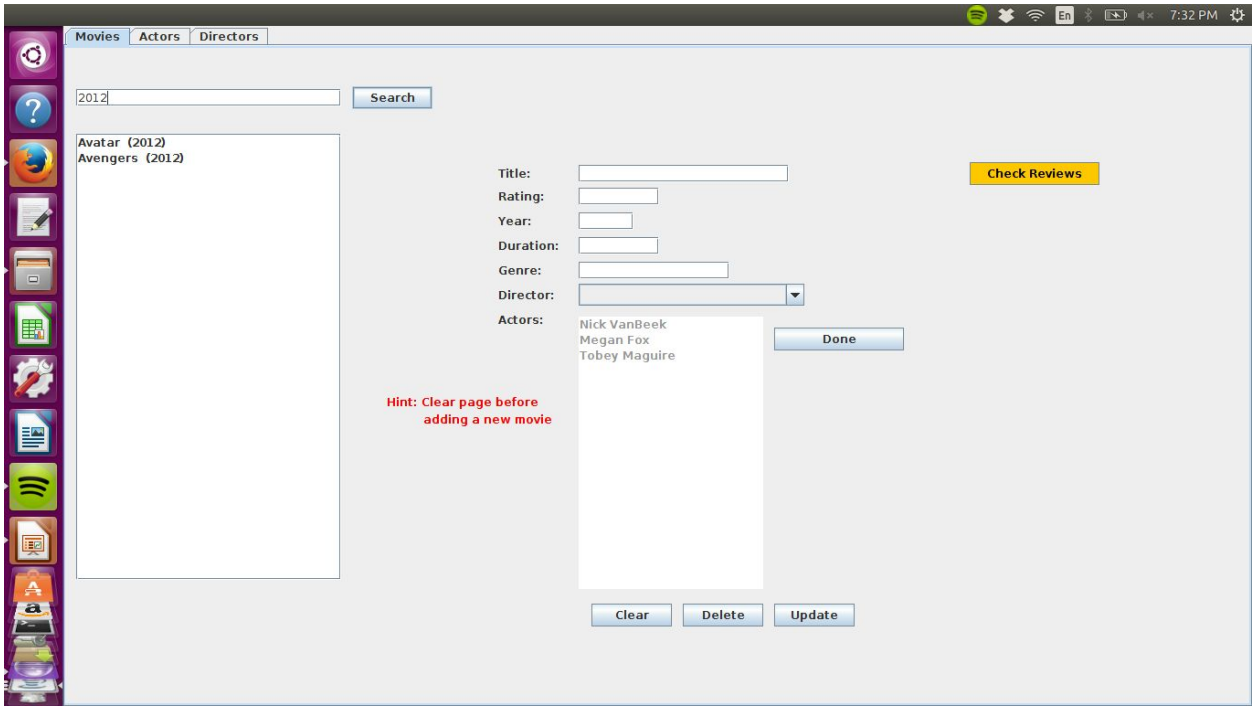


You can now “insert” this movie to the movie list

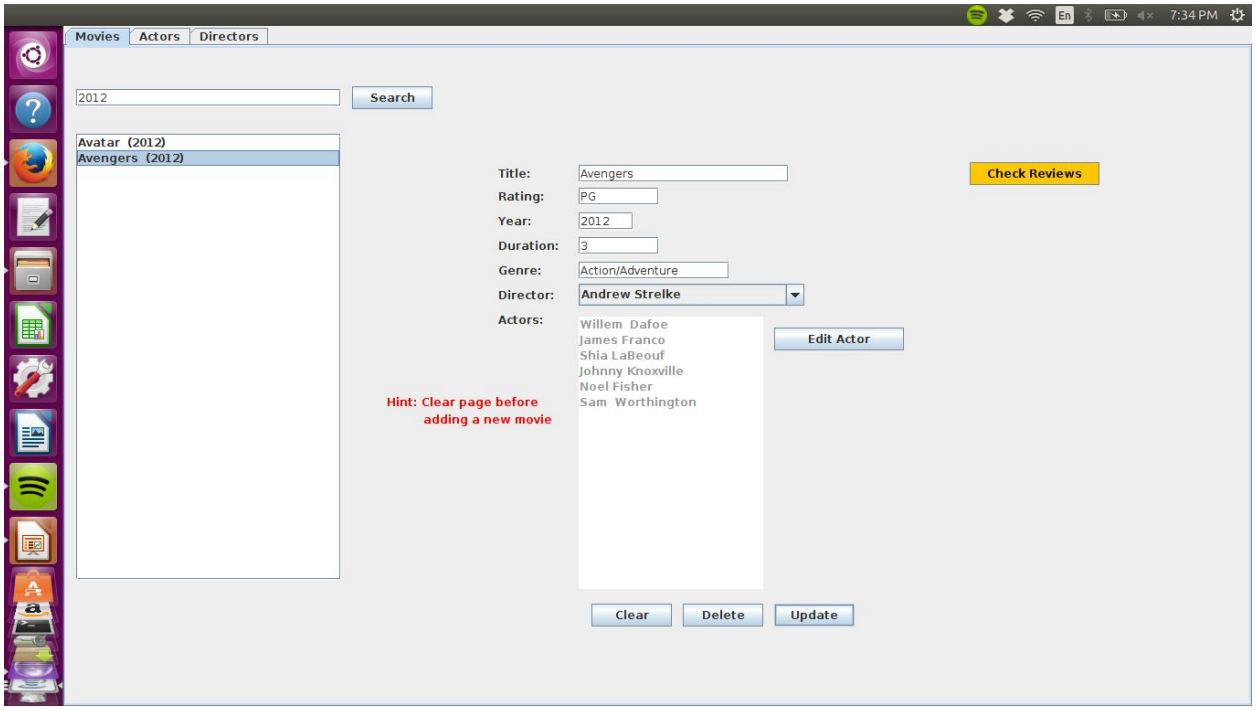


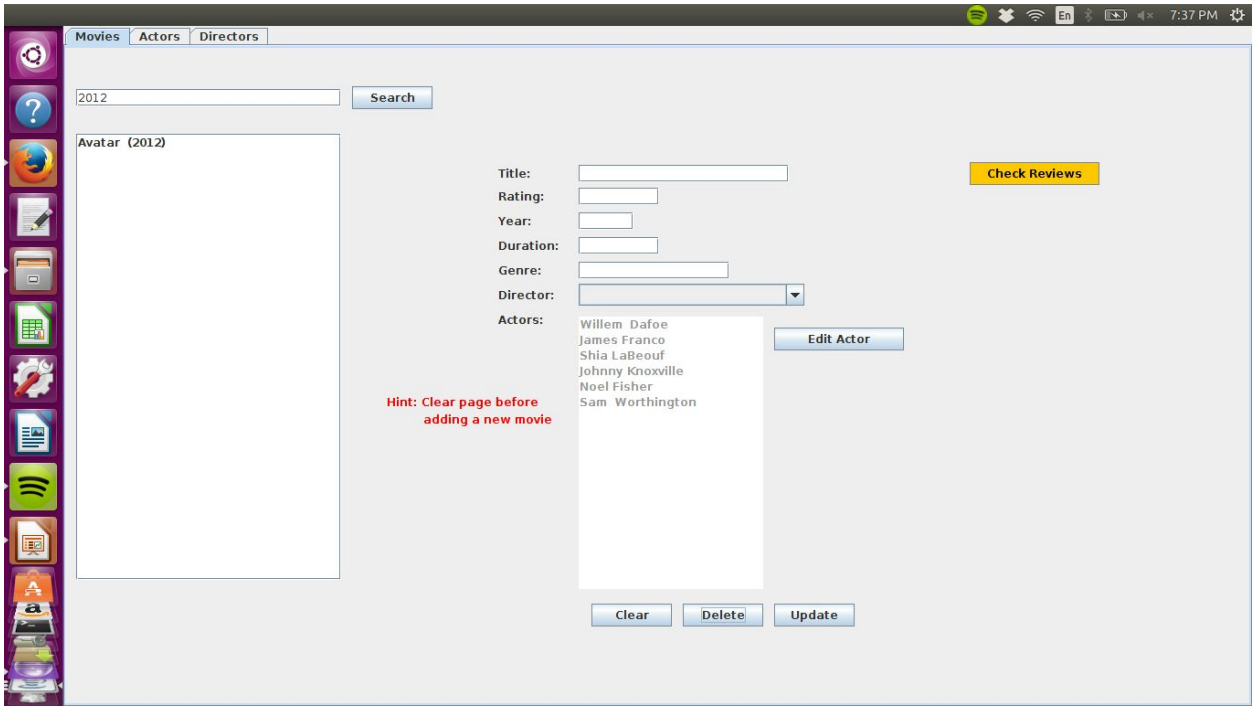
Make sure to clear the page before adding a new movie



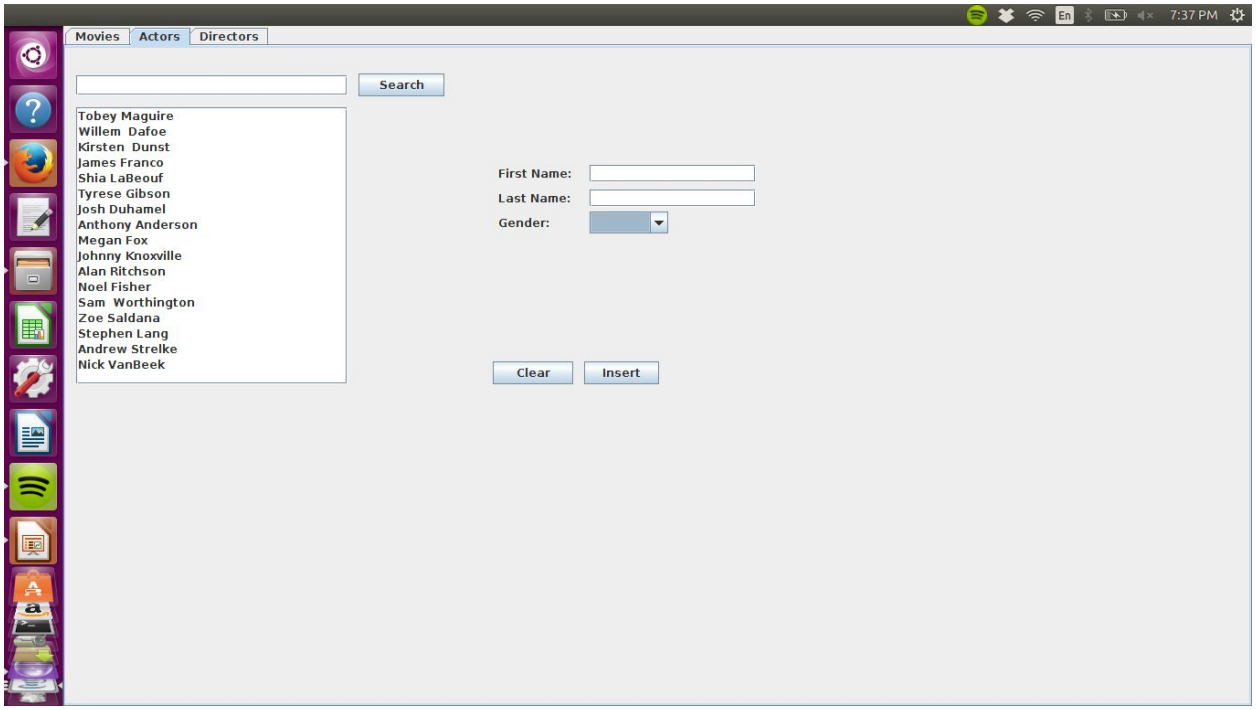


Delete a movie from the list by clicking on it and choosing delete

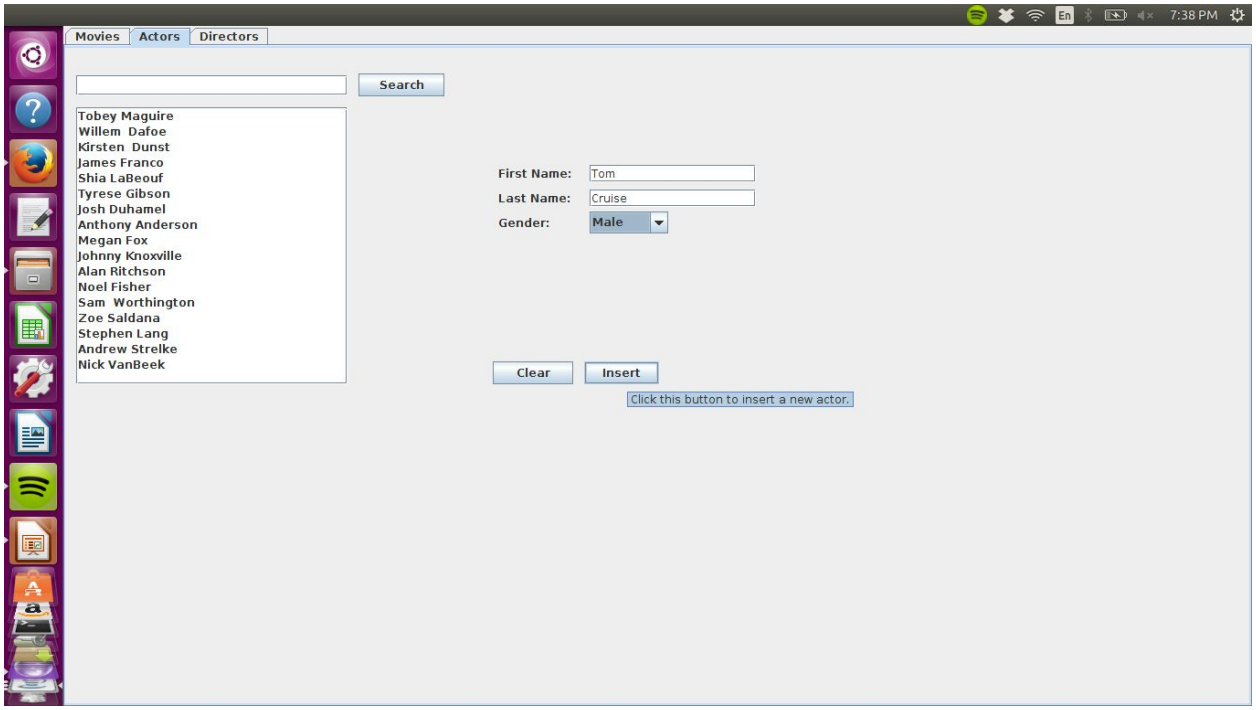




Here we have the actor list



Simply fill in the first and last name, as well as the gender. Then click insert



Your new actor will then appear in the list