Database Schema:

Movies(id: int(11), title: varchar(20), genre: varchar(20), release: date, language: varchar(20), duration: int(11), summary: varchar(256))

Crew(mid: int(11), name: varchar(20), role: varchar(20))

Tags(mid: int(11), tag: varchar(20))

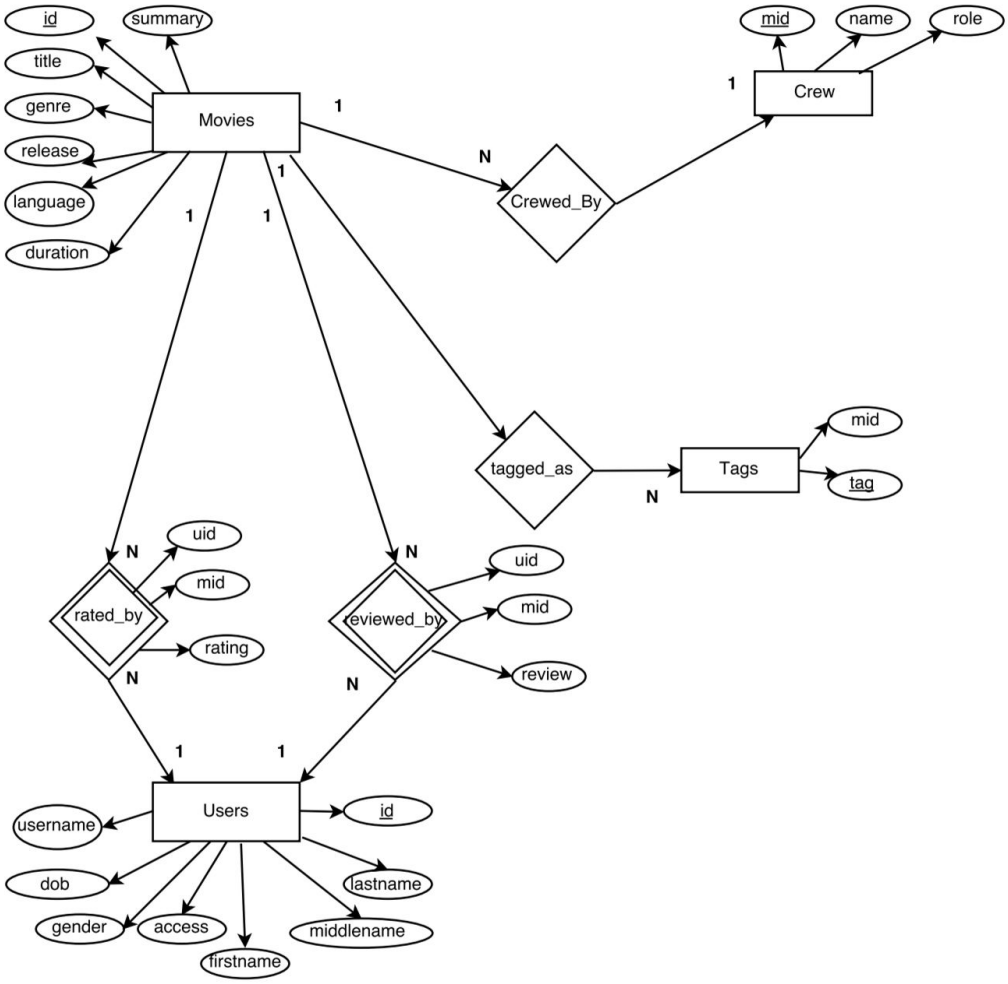
Users(id: int(20, firstname: varchar(20), middle: char(1), lastname: varchar(20), username: varchar(20), dob: date, gender: char(1), access: varchar(20))

Ratings(uid: int(11), mid: int(11), rating: int(11))

Reviews(uid: int(11), mid: int(11), review: varchar(256))

All of the relations (crew, rating, tags, and review) are 3NF because they aren’t determined by any nonprime attributes (only by candidate keys of the table)

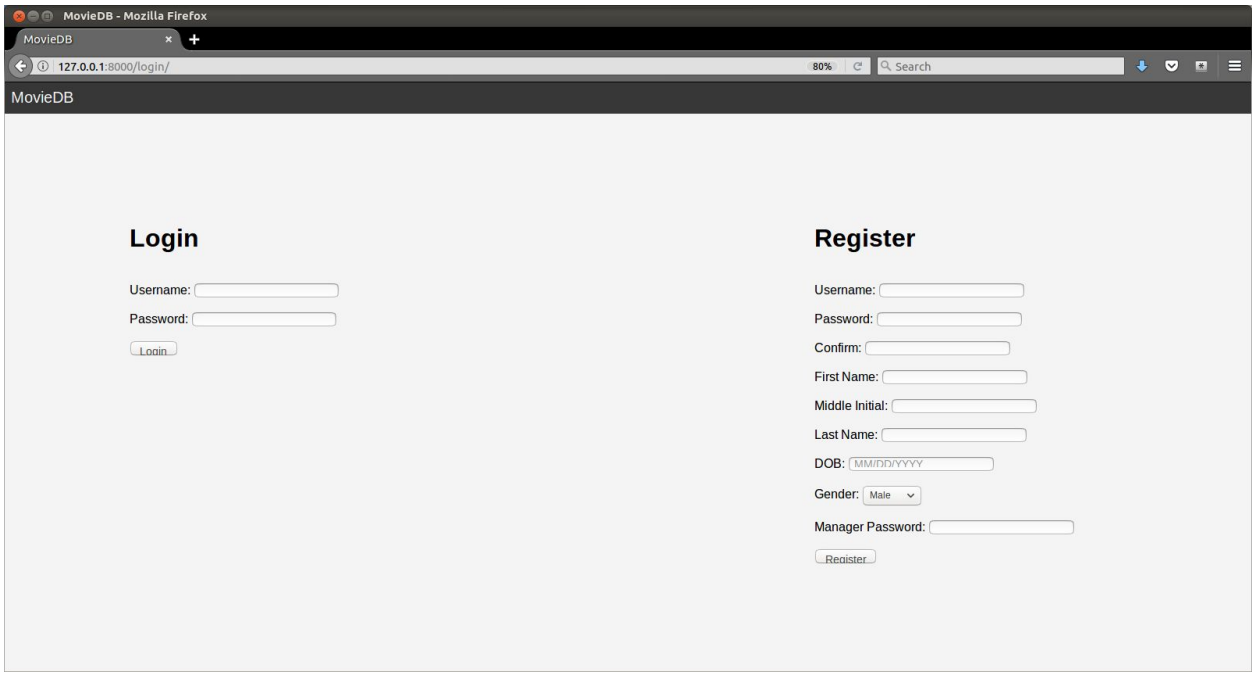
ER Diagram



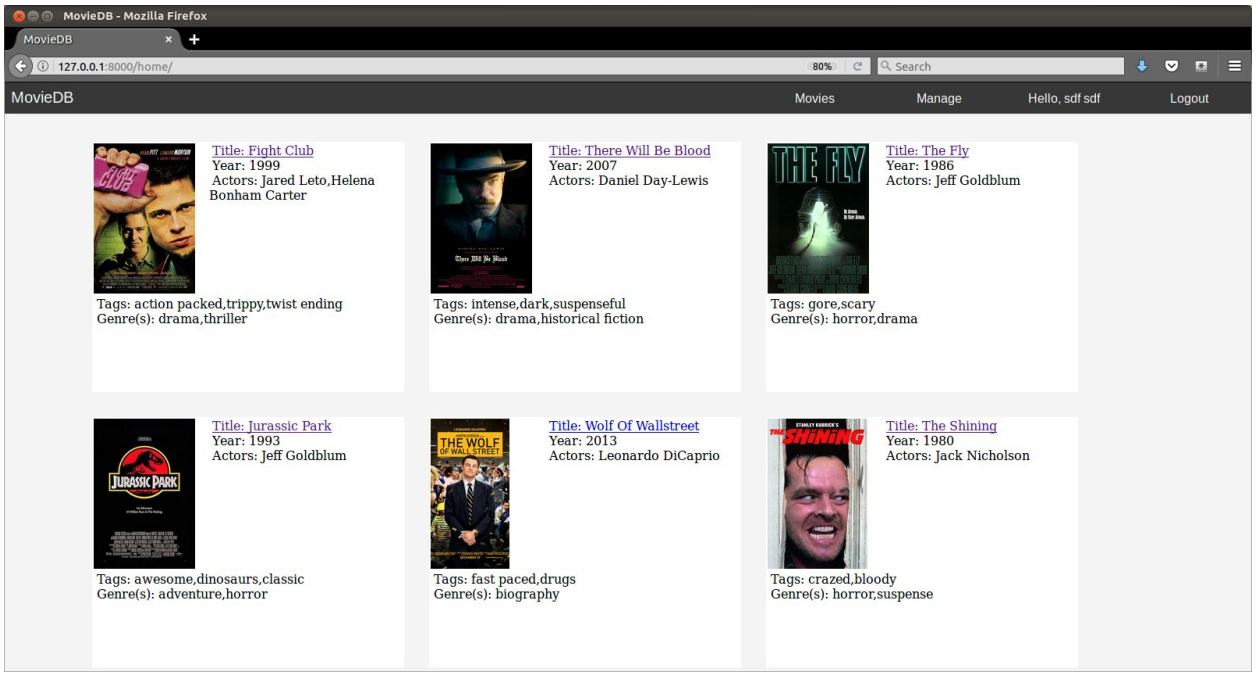
Functionalities (all users)

- Login/logout
- User Registration

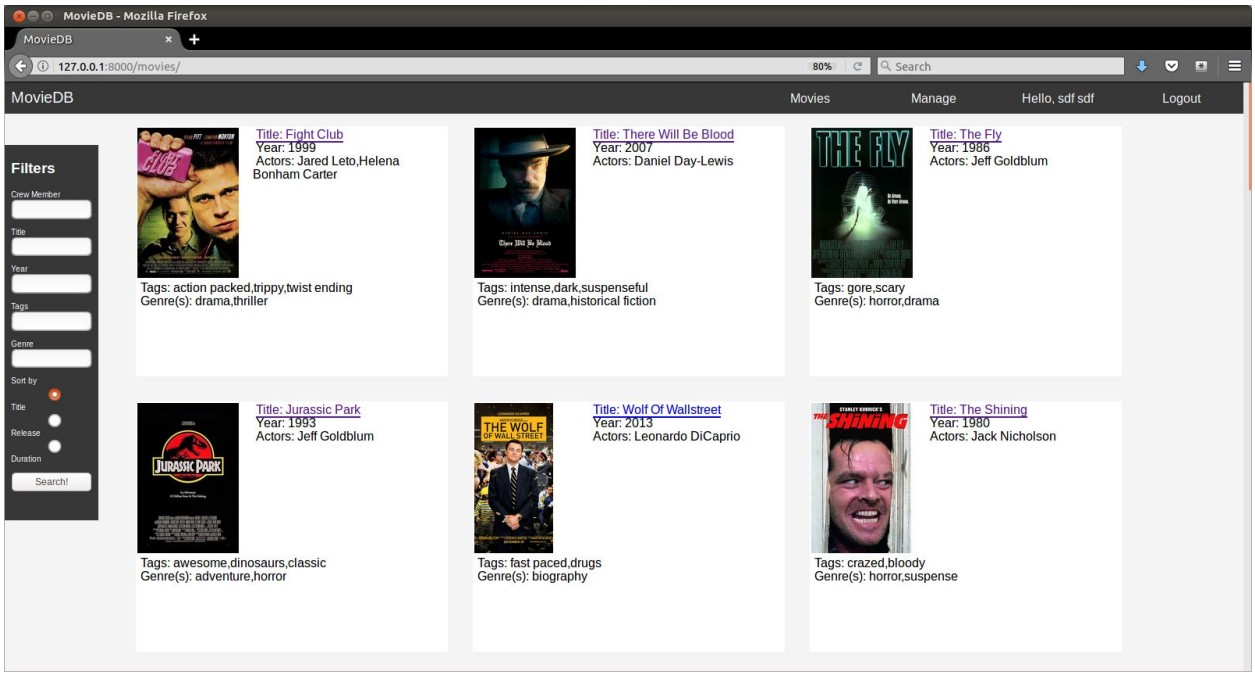
- New Manager Registration (With manager password)



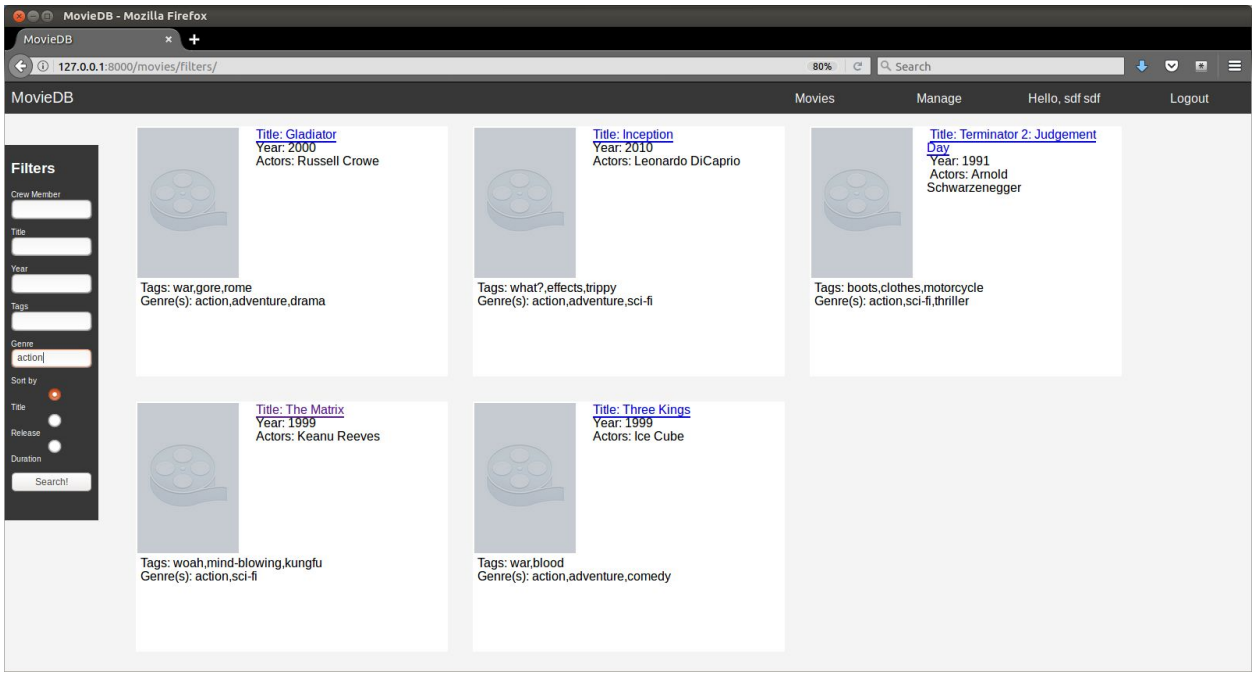
- Home Movies page visible

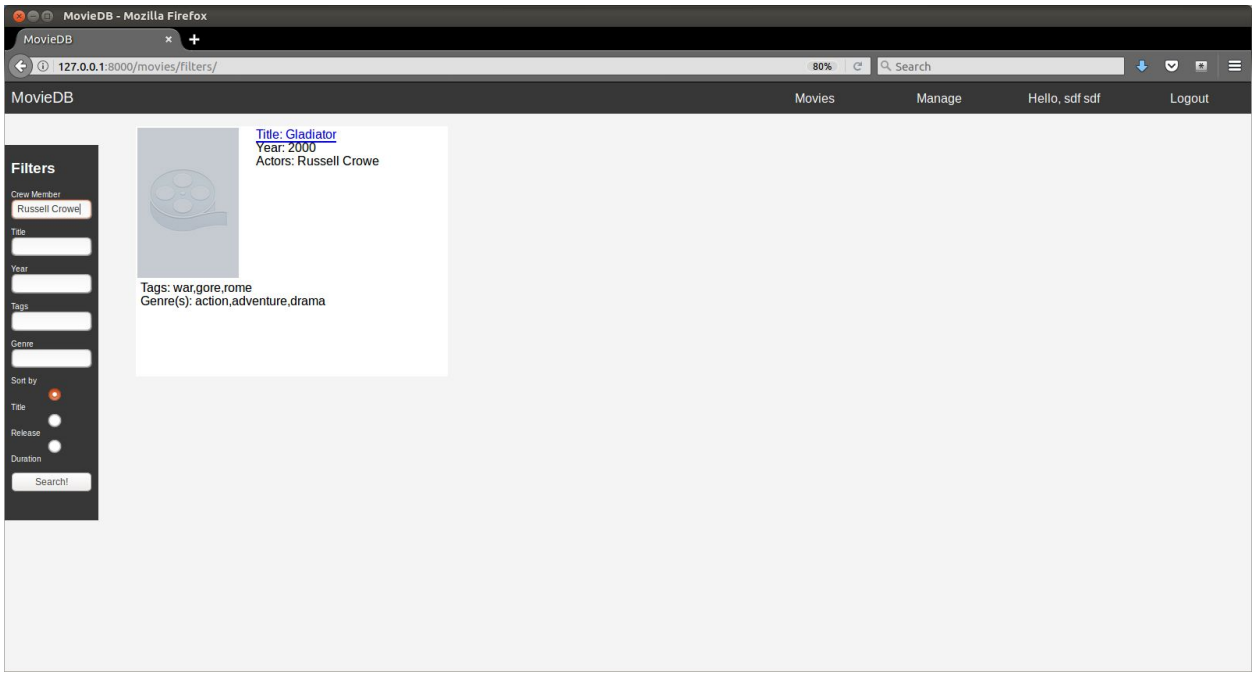


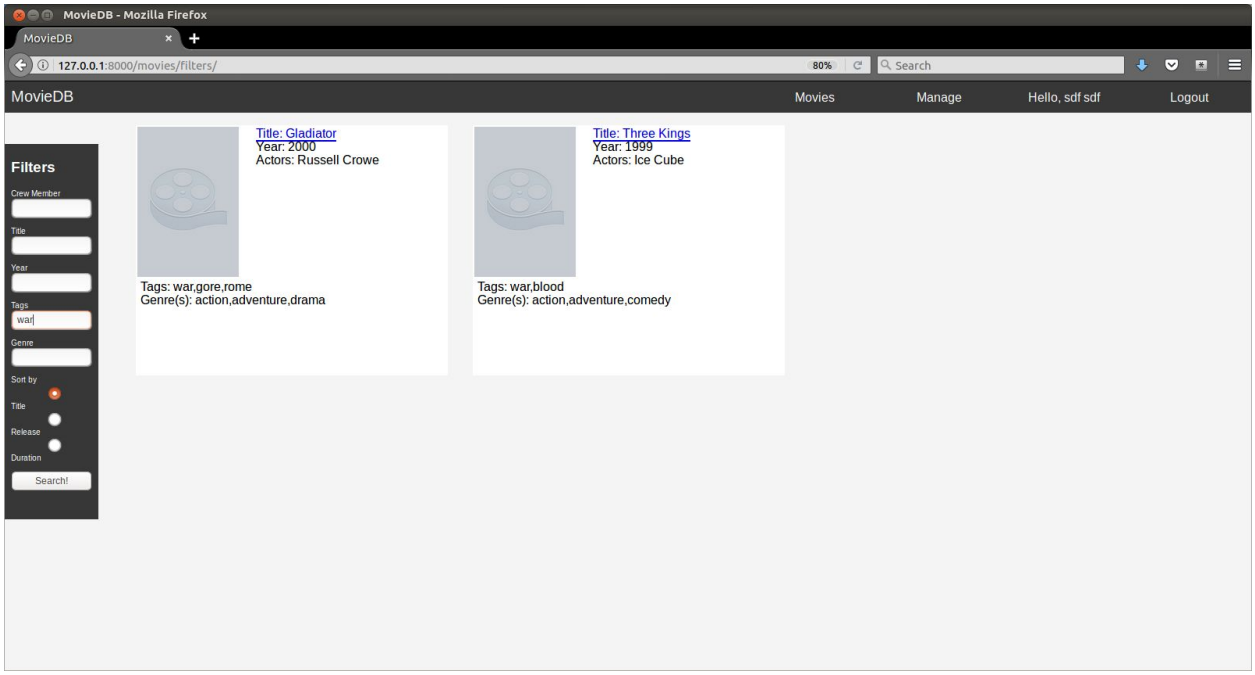
- All movies page (with filter box) visible



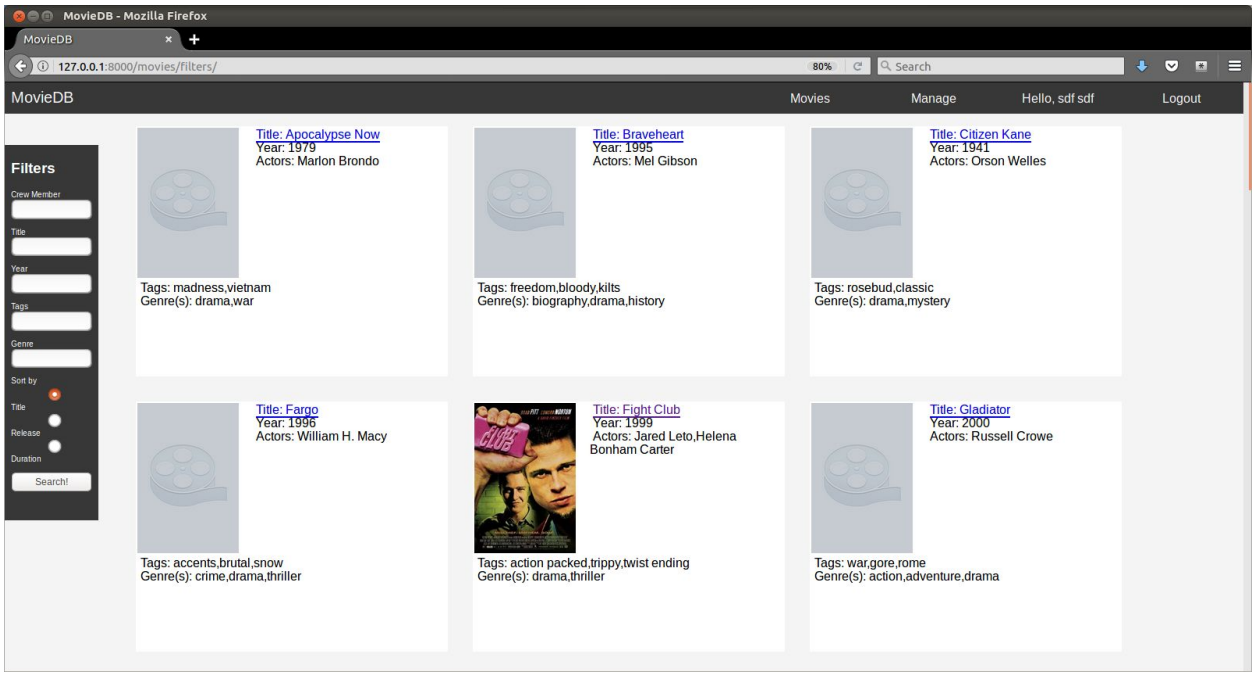
- Filter by genre, tag, film crew

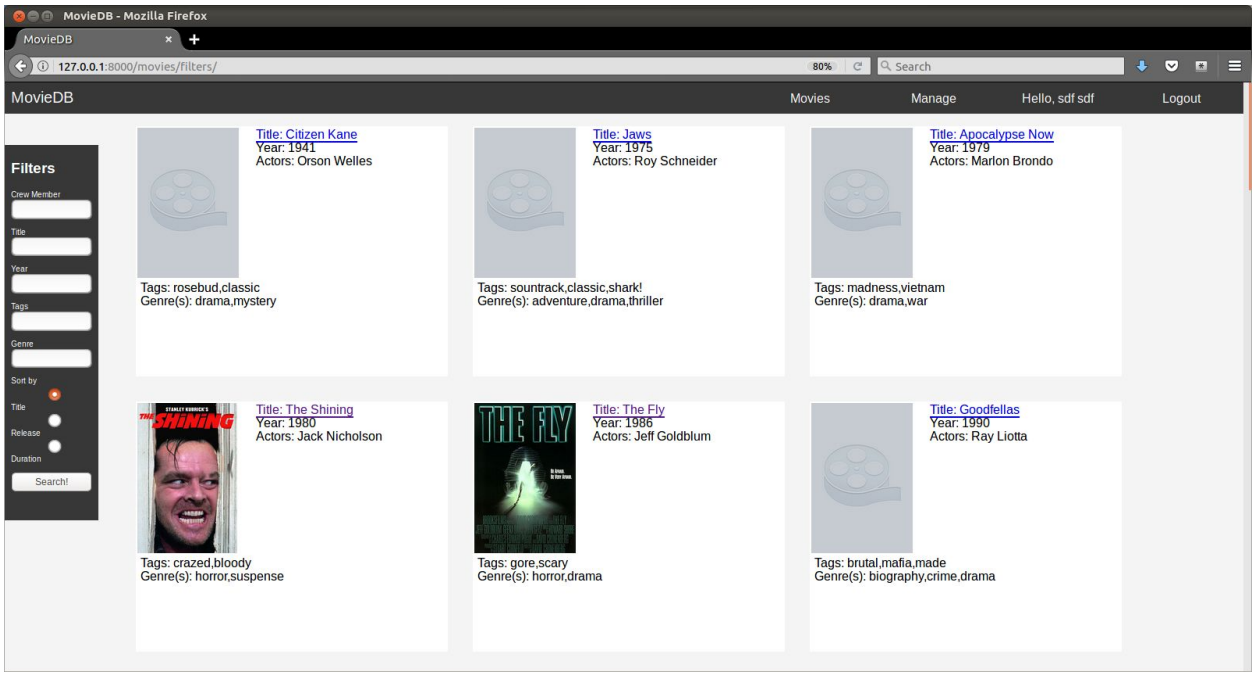


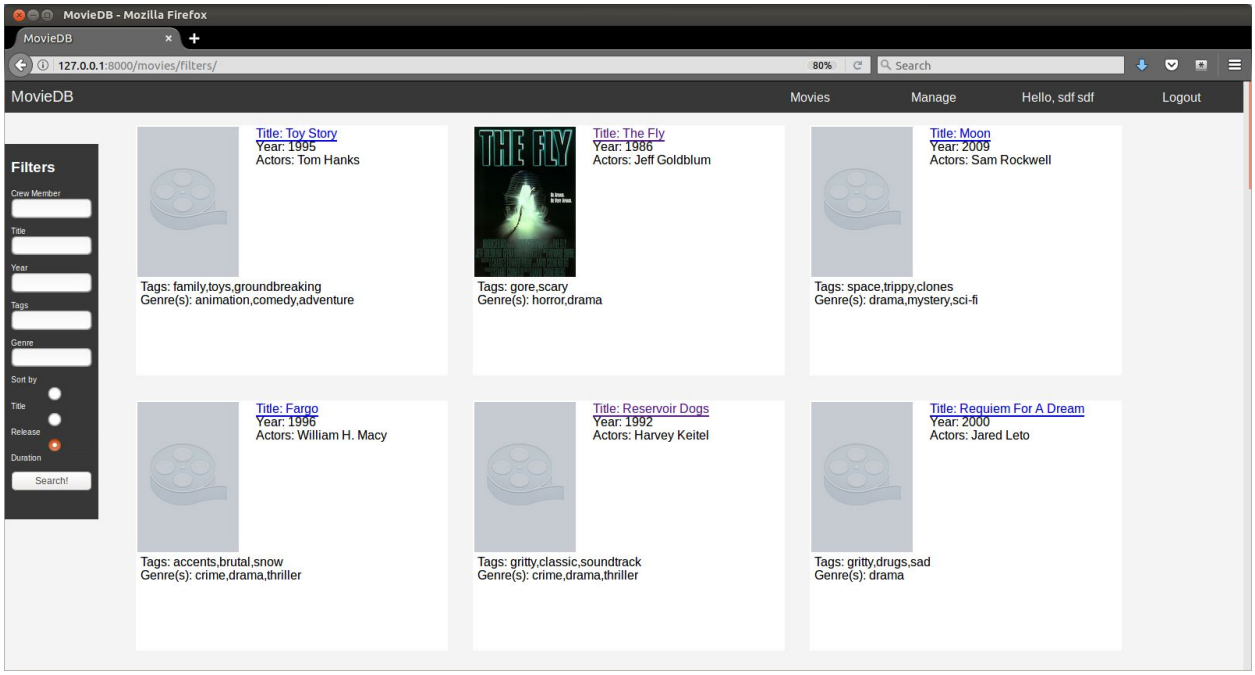

MovieDB - Mozilla Firefox
MovieDB
127.0.0.1:8000/movies/filters/
80%
Search
MovieDB
Movies
Manage
Hello, sdf sdf
Logout
Filters
Crew Member
Russell Crowe
Title
Year
Tags
Genre
Sort by
Title
Release
Duration
Search!
Title: Gladiator
Year: 2000
Actors: Russell Crowe
Tags: war,gore,rome
Genre(s): action,adventure,drama


MovieDB - Mozilla Firefox
MovieDB
127.0.0.1:8000/movies/filters/
80%
Search
MovieDB
Movies
Manage
Hello, sdf sdf
Logout
Filters
Crew Member
Title
Year
Tags
war
Genre
Sort by
Title
Release
Duration
Search!
Title: Gladiator
Year: 2000
Actors: Russell Crowe
Tags: war,gore,rome
Genre(s): action,adventure,drama
Title: Three Kings
Year: 1999
Actors: Ice Cube
Tags: war,blood
Genre(s): action,adventure,comedy

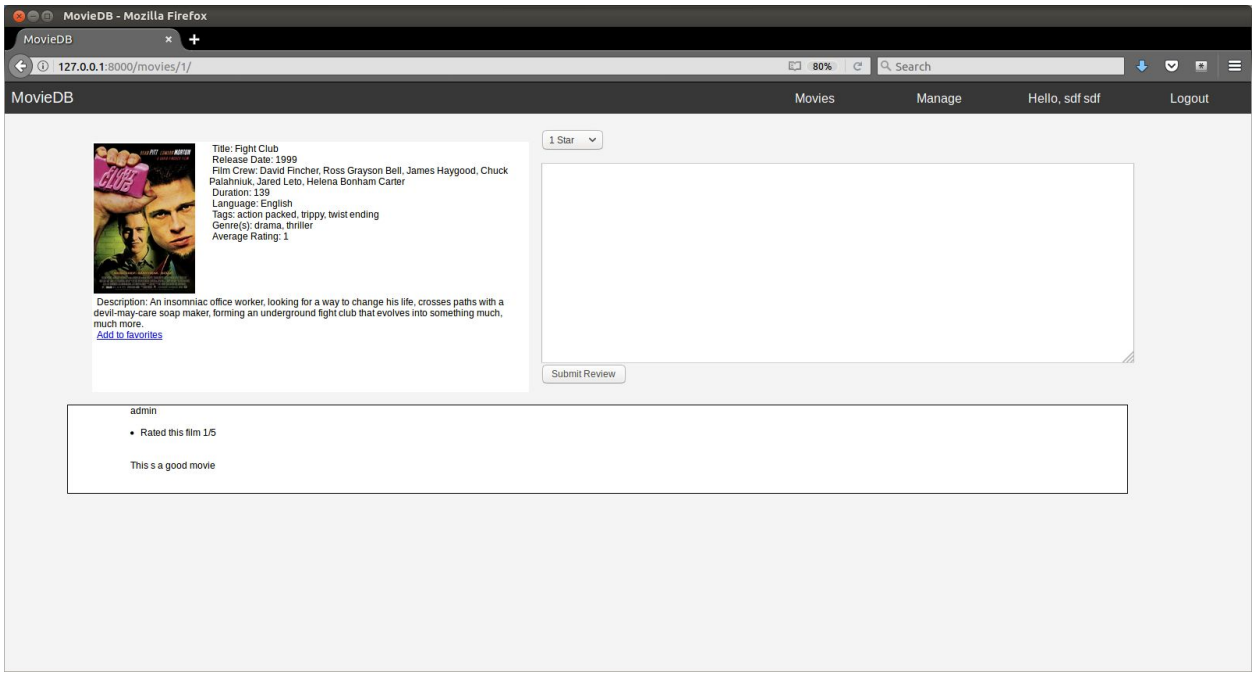
- **Sort by title, year, duration (bonus)**



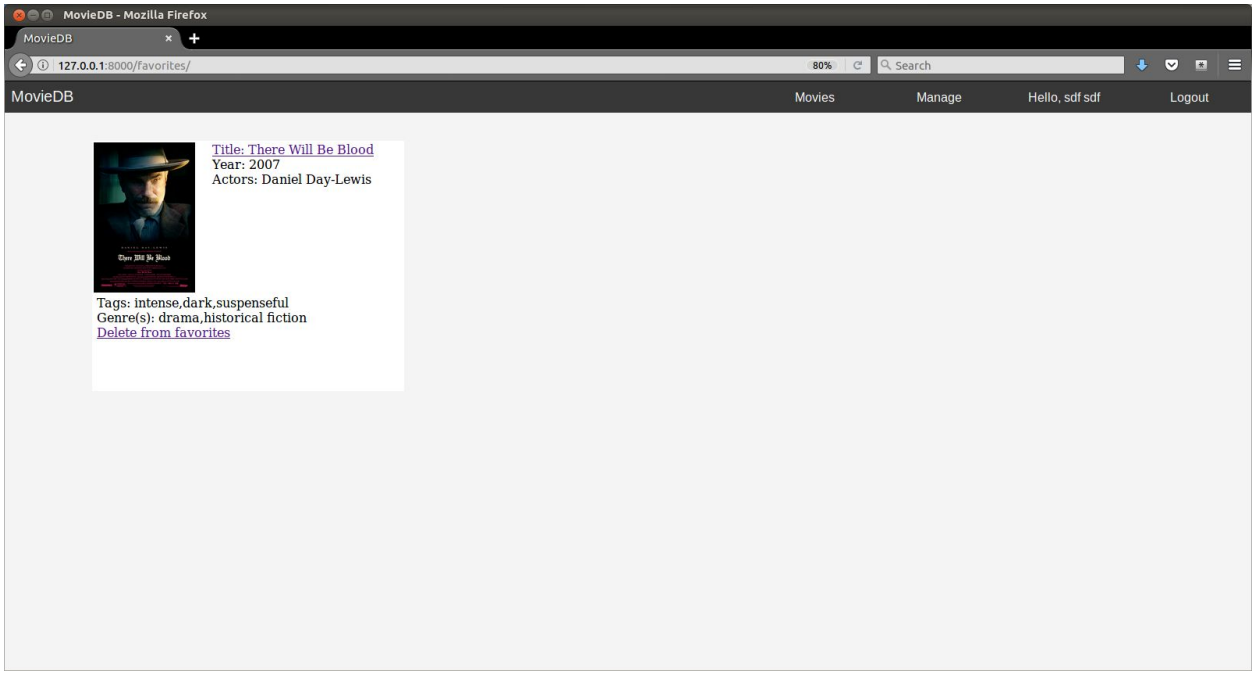

MovieDB - Mozilla Firefox
MovieDB
127.0.0.1:8000/movies/filters/
80%
Search
MovieDB
Movies
Manage
Hello, sdf sdf
Logout
Filters
Crew Member
Title
Year
Tags
Genre
Sort by
Title
Release
Duration
Search!
Title: Citizen Kane
Year: 1941
Actors: Orson Welles
Tags: rosebud,classic
Genre(s): drama,mystery
Title: Jaws
Year: 1975
Actors: Roy Schneider
Tags: sountrack,classic,shark!
Genre(s): adventure,drama,thriller
Title: Apocalypse Now
Year: 1979
Actors: Marlon Brondo
Tags: madness,vietnam
Genre(s): drama,war
Title: The Shining
Year: 1980
Actors: Jack Nicholson
Tags: crazed,bloody
Genre(s): horror,suspense
Title: The Fly
Year: 1986
Actors: Jeff Goldblum
Tags: gore,scary
Genre(s): horror,drama
Title: Goodfellas
Year: 1990
Actors: Ray Liotta
Tags: brutal,mafia,made
Genre(s): biography,crime,drama


MovieDB - Mozilla Firefox
MovieDB
127.0.0.1:8000/movies/filters/
80%
Search
MovieDB
Movies
Manage
Hello, sdf sdf
Logout
Filters
Crew Member
Title
Year
Tags
Genre
Sort by
Title
Release
Duration
Search!
Title: Toy Story
Year: 1995
Actors: Tom Hanks
Tags: family,toys,groundbreaking
Genre(s): animation,comedy,adventure
Title: The Fly
Year: 1986
Actors: Jeff Goldblum
Tags: gore,scary
Genre(s): horror,drama
Title: Moon
Year: 2009
Actors: Sam Rockwell
Tags: space,trippy,clones
Genre(s): drama,mystery,sci-fi
Title: Fargo
Year: 1996
Actors: William H. Macy
Tags: accents,brutal,snow
Genre(s): crime,drama,thriller
Title: Reservoir Dogs
Year: 1992
Actors: Harvey Keitel
Tags: gritty,classic,soundtrack
Genre(s): crime,drama,thriller
Title: Requiem For A Dream
Year: 2000
Actors: Jared Leto
Tags: gritty,drugs,sad
Genre(s): drama

- Addition of ratings + reviews

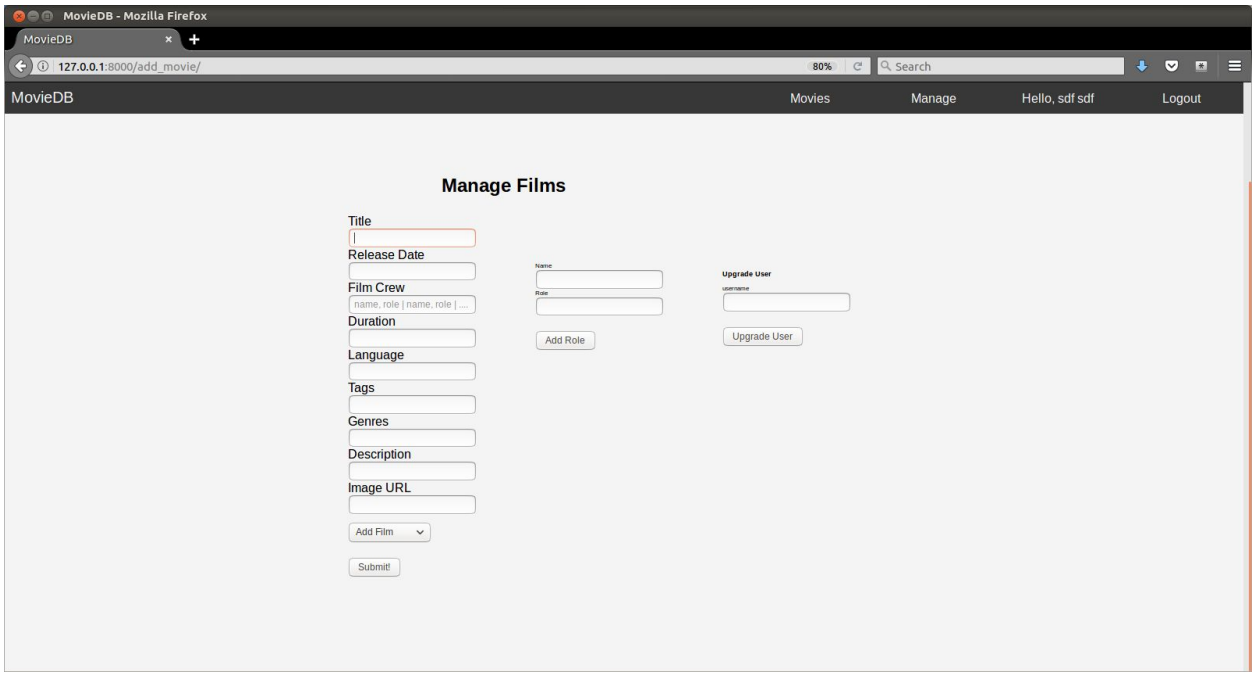


- **Favorite movies can be saved (bonus)**



Functionalities (Managers)

- Addition of Movies + movie components (tag, genre, film crew, **title**, **year made**, **duration**)



- Deletion of Movies + movie components (Dropdown menu offers new options)
- Update of Movies + movie components
- User upgrade to manager

Testing

- Login/logout
    - Tested by Making sure registered users were the only ones able to obtain access into the website
    - Used unregistered usernames and passwords for login to test security
    - Made sure when different site links were input, it would always redirect to the login page to ensure no unregistered users could enter
- User Registration
    - Tested by inputting different values into the registration boxes - ensured that new users input values for all required fields, otherwise an error would output
    - Made sure registered user could login successfully
- New Manager Registration
    - Registration process using a predefined manager key
    - Made sure registration and login were successful, made sure user has manager capabilities (New manager tab appears)
- Initial Movies page visible
    - When any user logs in they must see movies homepage with popular movie entries (picture and name)
- Movies must have name, and it may have a picture
- Clicking movie must bring user to movie page
- All movies page (with filter box) visible

    - Any user can click on the movies tab and see all movies with filter box visible
- Filter by genre, tag, film crew, **title**, **year made**, **duration**
    - Tested by inputting different (correct and incorrect) values into text fields of the filter box. Once searched, correct movies must show up
- Addition of tags
- Addition of ratings + reviews
    - Users must be able to input a new rating, and new rating must update the movie rating average
    - Users must be able to input a new review and a new review must add to old ones and be visible
- **Favorite movies can be saved**
    - Users must be able to add and delete movies to their favorites list

Functionalities (Managers)

- Addition of Movies + movie components (tag, genre, film crew, **title**, **year made**, **duration**)
    - In the manager tab, managers must see the text fields for addition of movies and movie components
    - Selection of viable additions must add movie (or attribute)
    - Putting wrong inputs into the text boxes must show an error
- Deletion of Movies + movie components
    - Deletion of correct movie title must remove movie from database
    - Deletion of incorrect movie title/attributes must show an error
- Update of movie components
    - Movie updates must change movie attribute values
    - Incorrect movie spelling (or movie that isn't available) will return an error
- User upgrade to manager
    - Once in the manager tab, several unique text fields should be present
    - Including upgrade from regular user to manager
    - Once user was upgraded, we ensured subsequent logins by that user would give them manager capabilities

README

1. Install python 3.6 (anaconda or something else, your pick here)
2. Install django - https://www.djangoproject.com/download/
3. Install mysql - https://dev.mysql.com/doc/refman/5.7/en/linux-installation.html
4. `sudo apt-get install libmysqlclient-dev`
5. `pip install mysqlclient`
6. Create mysql user “moviedb_user” with password “moviedb_user”, grant all access to it https://www.digitalocean.com/community/tutorials/how-to-create-a-new-user-and-grant-permissions-in-mysql
7. `git clone https://weiliansong@bitbucket.org/weiliansong/moviedb.git`
8. `cd moviedb/src/moviedb/`
9. `bash remigrate.py`
10. `python manage.py shell`
11. In the opened ipython session, `cd database/`
12. In the opened ipython session, `run fill_db.py`
13. In the opened ipython session, `exit`
14. `python manage.py runserver`
15. In a browser, navigate to 127.0.0.1:8000/login

Experience and Challenge

1. Django is an excellent framework, but it is hard to grasp its concept at first. Concept of an “app” within django was confusing, and the initial design was not efficient enough for rapid, flexible development
2. User input validation can get tedious, especially when it can be composite
3. Database schema design has to be clean for efficient queries, since it has to go through django’s interface with SQL
4. Keeping track of which user logged in is a bit difficult, we eventually fall back on a backend-based method
5. Setting up new machines to run website server is tedious, but the README should be able to compensate for this difficulty
6. Git workflow is new for some of our members, so there’s a small learning curve using a version control system
7. Django’s interface with SQL resulted with us not being able to use our mysql dump directly to populate our tables. We resorted to a custom script whose functionality is essentially the same. As such, the final mysql dump file (though it does contain at least 30 entries per table) does not contain all of the data represented on the website.

Contributions

Weilian Song - django developer

Henry Uradu - front end development

Nathan Serpico - front end development

Greg McIntosh - backend development