

|  |
| --- |
| Dean Crockett, Jacob Pawlak, Nick Reaves, Connor VanMeter  4-21-2017 |

|  |
| --- |
| Movie Database |
| CS 405 Final Project Report |

Table of Contents

[(1) Database Design 2](#_Toc480578089)

[Database Assumptions 2](#_Toc480578090)

[ER Diagram 2](#_Toc480578091)

[Database Schema 3](#_Toc480578092)

[(2) Web Site 4](#_Toc480578093)

[Functionalities 4](#_Toc480578094)

[Overview 4](#_Toc480578095)

[CREATE account/Register 4](#_Toc480578096)

[SEARCH 5](#_Toc480578097)

[ADD TAGS 5](#_Toc480578098)

[RATE/REVIEW 5](#_Toc480578099)

[Add/Remove Users 5](#_Toc480578100)

[Add/Remove Movies 5](#_Toc480578101)

[Additional Functionalities 5](#_Toc480578102)

[Watch list 5](#_Toc480578103)

[Boot Strap 5](#_Toc480578104)

[POPULAR MOVIES 6](#_Toc480578105)

[(3) Testing 7](#_Toc480578106)

[(4) Contribution 7](#_Toc480578107)

[Dean Crockett: Frontend Web interface 7](#_Toc480578108)

[Jacob Pawlak: Backend 7](#_Toc480578109)

[Nick ReAves: Frontend/Backend collaboration 8](#_Toc480578110)

[Connor VanMeter: Database Design, Project Reporting 8](#_Toc480578111)

[(5) Experiences and Challenges 8](#_Toc480578112)

# (1) Database Design

Underline: primary attribute

*Italics*: multiple-value attribute

**Entity: MOVIE**

Attributes(s):

release\_date

title

description

movie\_id

duration

language

*keyword*

*genre*

Relationship(s):

RATED BY: USER

HAS: MOVIE\_CREW

**Entity: CREW**

Attribute(s):

crew\_id

first\_name

middle\_name

last\_name

Relationship(s):

HAS: MOVIE

**Entity: USER**

Attribute(s):

first\_name

middle\_name

last\_name

gender

dob (date of birth)

type

user\_id

password

Relationship(s):

RATES: MOVIE

**Relationship: ROLES**

Attribute(s):

*role*

Relates:

movie <-> crew

**Relationship: RATING**

Attribute(s):

value

review

wanted (yes/no)

Relates:

user <-> movie

## Database Assumptions

* All users may rate, review, and save more than one movie. All movies may be rated, reviewed, or saved by more than one user.
* Users and crew are different; crew *will not* be linked to users.
* Each movie may have only one language.
* Middle names may be NULL (blank).
* Value and review may be left NULL (blank).
* Duration must be a value greater than 0.
* Value is 0 or 1, where 0 is thumbs down and 1 is thumbs up.
* Wanted is whether the movie is saved to a watch-list. Value is 0 (no) by default.
* Gender: (M)ale, (F)emale, (O)ther

## ER Diagram



## Database Schema

MOVIE (movie\_id, movie\_pic, title, description, release, duration, language)

GENRE (movie\_id, genre)

TAG (movie\_id, tag)

CREW (crew\_id, first\_name, middle\_name, last\_name)

ROLES (movie\_id, crew\_id, role)

USER (user\_id, email, gender, type, dob, password, fname, mname, lname)

RATING (movie\_id, user\_id, rating, review, wanted)

WATCHLIST (movie\_id, user\_id)

1NF: All databases depend on the primary key. All values are atomic. There are no composite attributes, multivalued attributes, nested relations, or repeating groups.

2NF: All databases depend on only the primary key. Non-prime attributes do not have partial dependencies.

3NF: All databases depend on nothing but the primary key. There are no transitive dependencies.

BCNF: All databases meet this requirement.

Note: USER could have user\_id or email as the primary key since both are unique. They are together for convenience of handling the database.

# (2) Web Site

## Functionalities

The live site can be accessed at <http://www.jacobpawlak.tech/index.php>

### Overview

The home page contains a section for users to sign in/register. Users are greeted with popular movies. This list is given by the highest rated movies in the database. A logged-in user will see their watch-list on the right-hand side. A home button is in the upper left corner; a search bar is next to the home button. A banner is on the bottom that links to a Facebook page and email for contact.

### CREATE account/Register

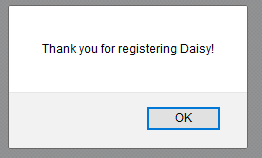
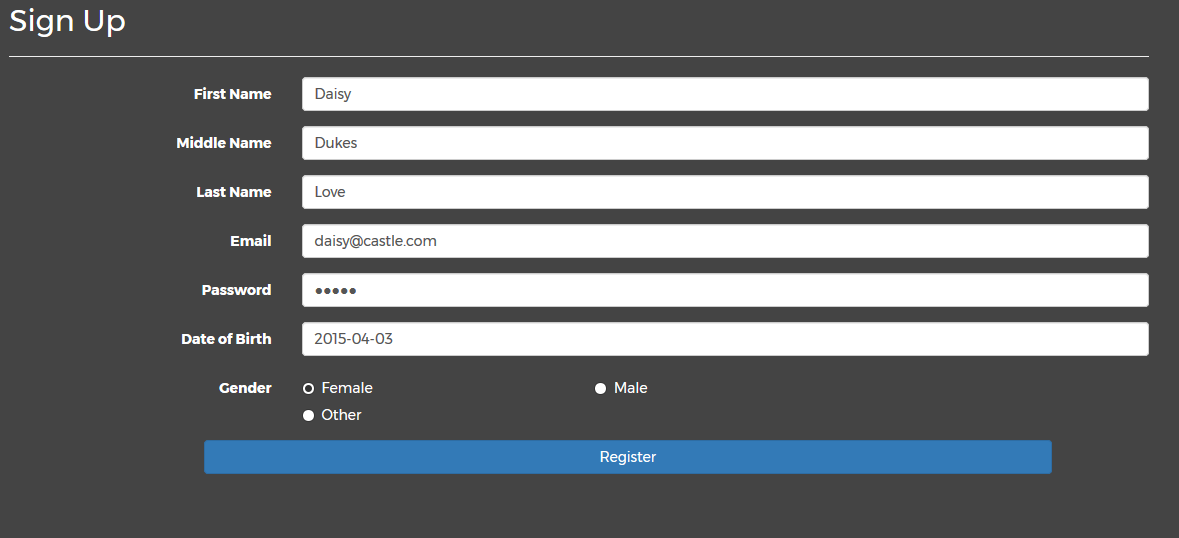
Users can register for an account. A user can hit register on the site’s header to go to registration. The user inputs their first name, middle name, last name, email, password, date of birth, and gender. An account is created with this information if the email is unique. A password confirmation field is not included. A user can sign in with the email and password provided at registration.

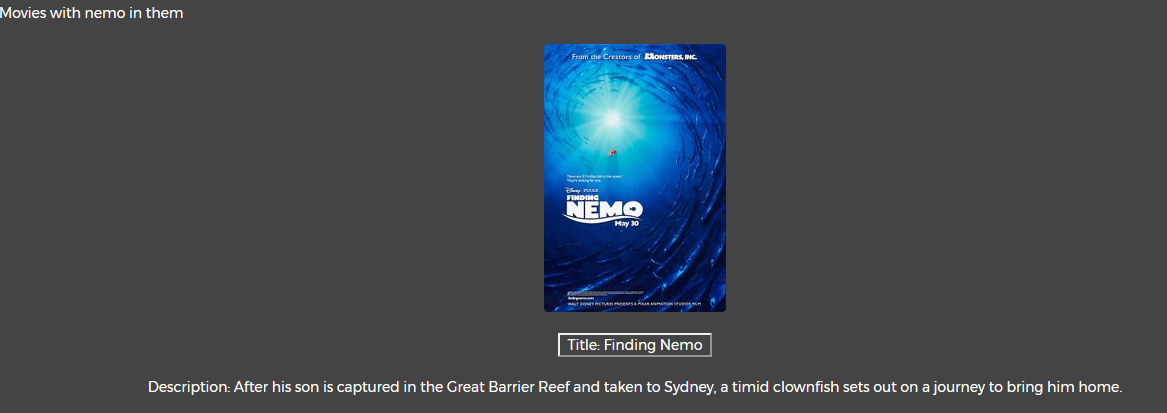
Figure . (Top) Registered users can enter their email and password to sign-in. Users can also hit register to be taken to the registration page. (Bottom) The registration requires several pieces of information from the user before an account can be created. (Right) Confirmation shown to users upon registering.



### SEARCH

Users can search for movies, crew, genres, and tags. The user is not given the option to select the individual field – search checks all fields for a match by default.

Figure . (Top Left) Empty search field. (Top Right) Search field expands when interacted. "nemo" is searched in this example. (Bottom) The "nemo" search is executed. "Finding Nemo" is found. The movie title, poster, and description are shown. The poster can be clicked to be taken to the movie page.



### ADD TAGS

All users can add tags to a movie.

### RATE/REVIEW

All users can rate and review movies. Movies are rated on a 5-star scale; the average rating is shown for each movie. Users can also leave reviews for each movie. The number of reviews and the contents of each review are shown.

### Add/Remove Users

Administrators can add/remove users and can update user roles.

### Add/Remove Movies

Administrators can add/remove movies. This includes the movie title, description

## Additional Functionalities

### Watch list

Users can add movies to their watch list. The watch list is displayed on the home page.

### Boot Strap

The webpage’s display is flexible; it will match the window size and is auto-adjusted for the device/window.

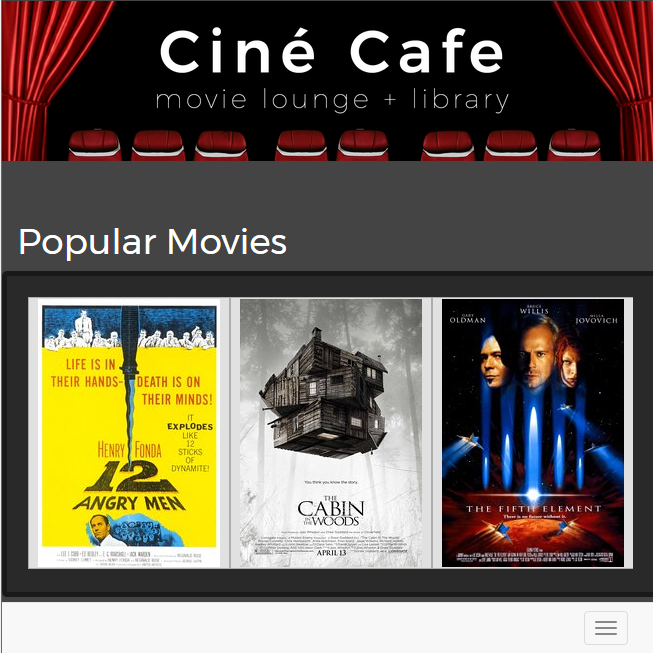


Figure . Boot strap demonstration of the website. The browser window has been shrunk; the menu has been collapsed and moved to the bottom. The number of popular movies changes depending on the size of the window.

### POPULAR MOVIES

The front page has a carasoul that display the highest rated movies in the database. Users can scroll through these options to see which movies others have highly rated.

# (3) Testing

The SQL was initially tested by manually inserting statements on the backend to make sure that the table ids autoincremented, the primary keys worked, and each table took data as intended. Testing was further expanded with the website. Each function was tested individually multiple times with different test cases to ensure that everything was working.

Search: Different movie titles, tags, genres, and crew were searched. Items that are currently not included in the database were searched as well (e.g., “Easter”).

Bootstrap: Feature was tested by visiting website on different browsers, devices, and by manually adjusting the window size.

Registration/Sign-in: Tested by varying input. Date of birth was changed to improper format (not YYYY-MM-DD) and was rejected. Registration was attempted with missing fields – registration was rejected when fields were not completed. Registered users were signed-in to ensure sign-in was working. Sign-in was also tested with unregistered emails and incorrect passwords.

Rate/Review: Ratings were tested by manually interacting with the 5-star rating system to see if the entries and average rating changed. Reviews were written and posted to see if the page was updated to show the added review and if the review count changed.

Tags: Tags were tested with normal users and admin users. Each type of user added tags. The page was refreshed to ensure the tag was added to the list. Added tags are unique – no duplicates can be added.

Add/Remove User: Admin users were tested to see if they could remove users and update user type. An admin user promoted and demoted various users. These users were then signed-in to see if the change was reflected in the UI. When removing a user, the removed user was tested to see if they could still log-in. The database was also manually inspected to ensure that all related entries were removed (i.e., USER, RATING, WATCHLIST).

Add/Remove Movie: Admin users were tested to see if they could add movies. Added movies require all fields to be complete. The movie was then searched for using the search bar – all searchable fields were checked. Movies were then deleted. The deleted movies were searched for to ensure they were gone. User watch lists were then checked to make sure they reflected the change as well. Finally, the role table was manually inspected to see that corresponding roles were removed.

# (4) Contribution

## Dean Crockett: Frontend Web interface

Dean designed the front-end web application interface using HTML and CSS. The app employed Bootstrap for proper scaling on mobile devices, as well as a modern UI. It also used JavaScript functions for actions such as displaying the user’s name and film preferences.

## Jacob Pawlak: Backend

Jacob constructed the backend database. Using both PHP and MySQL, he defined the creation statements for the foundation of our database, as well as program the more intricate functionality for our site such as aggregate functions.

## Nick ReAves: Frontend/Backend collaboration

Nick acted as liaison between Dean’s frontend web interface and Jacob’s backend code. He worked with both partners to ensure that the proposed functionality was possible with Jacob’s implementation, and assisted Jacob in building the database with Dean’s design. He also helped both partners with additional tasks, such as programming specific functions on the site.

## Connor VanMeter: Database Design, Project Reporting

Connor took the lead on constructing our database schema. With the team’s input, he designed our databases various entities, and described their attributes and relationships. He also formatted our assumptions, and drew an ER Diagram illustrating the database. Connor also performed quality control for the web-application and created the final report.

# (5) Experiences and Challenges

An initial design challenge was making sure multiple genres and tags could be listed for a single movie. This was done by creating tables dedicated to keeping track of genres and tags. The primary key for these tables is (movie\_id, genre) and (movie\_id, tag). This ensures that each entry is unique and can be tracked. Similar work was done for crew/movie roles.

One element of the site that we had hoped to implement was a ‘movies you may like’ section with user-specific recommendations. We have a clear understanding of how to implement this; we could save the tags of high-rated movies to a table with the user id as the primary key. This would allow us to query a list of the user’s favorite movie tags with the highest ratings. We made the ‘tag’ table, which has columns ‘tag’ and ‘movie\_id’, but were unable to finish the PHP code to add a high-rated movie’s tags to the table. We would have implemented this feature given more time.