

MOVIE HUNTER - DESCRIPTION

Name: Fredrik Forsell

Emne: ITF11012 - DotNet.

Student Number: 162211

GENERAL DESCRIPTION

Movie Hunter is an application created with focus on having a good overview on the movies that you have access to, whether it is movies stored on your computer, or movies that you have access to through another webservice. The user of this application has access to a wide selection of functionality. For instance, the user can add new movies to a database. These movies are all displayed on one of the pages. The user can then add them to a To-Watch list, or a Watched list.

The application is set up to use the MVVM architectural pattern, however I have not used this in the app. I was planning on moving all of the code over to the viewmodel before turning it in. I realized that it required me to create bindings for every item in the xaml code. The amount of code in the app makes this a real struggle to do, but I'm hoping that it's okay, since I have the viewmodels properly configured. I feel like I have shown my competence with bindings in the other code.

REQUIREMENTS

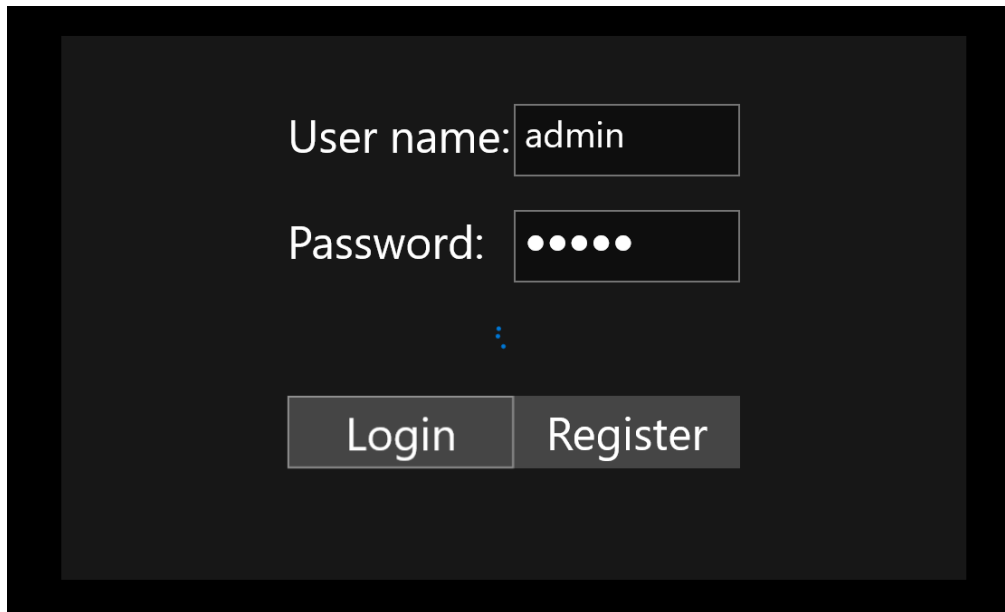
Internet access is required to be able to use this application (it has to be on the school network). You can either create a new user, or use the User with username: "admin" and password: "admin". This user already has items added to the different lists. Use this account when checking out the design and List page functionality.

FUNCTIONALITY

LOGIN

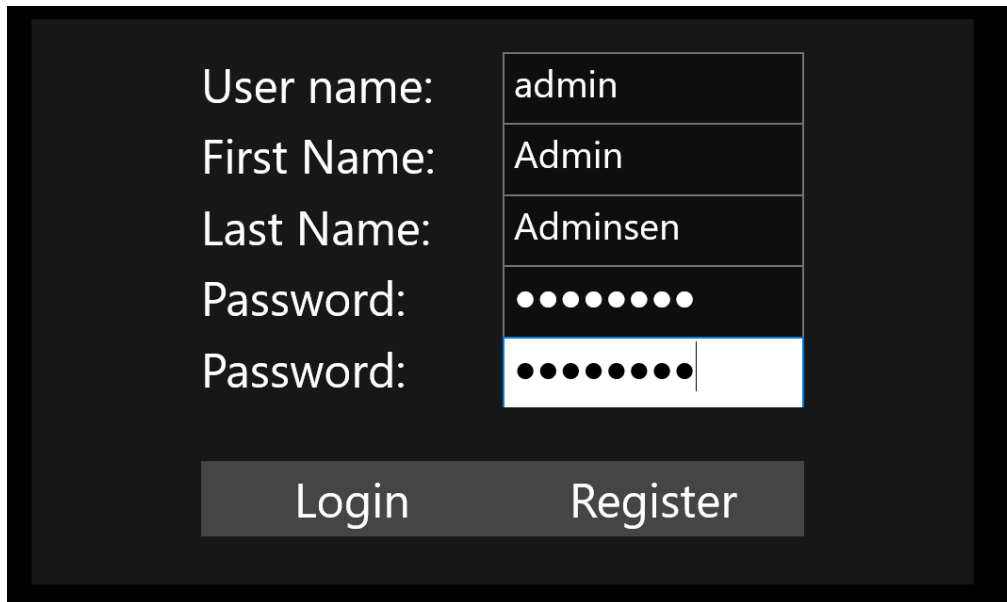
When the users open the application they are greeted with a login page as shown on the picture below. The password typed inside the password box will be converted to a salted hash value before being sent to the API for verification. This ensures that people won't be able to sniff the network traffic to get the password in plaintext. If the user clicks on the enter key (or the login button) the user will login. At this moment the user will not get notified for the reason why a login fails, However there is a spinning circle indicating that it is handling the request.

If the users clicks on the register button they will be redirected to the Registration page.

A screenshot of a login page with a dark background. It features two input fields: 'User name:' with the text 'admin' and 'Password:' with five dots. Below these fields is a small blue spinning circle. At the bottom are two buttons: 'Login' and 'Register'.

REGISTRATION

By typing information into the text boxes shown in the picture below followed by clicking enter (or clicking register), the application will create a user at the database. The password is hashed in the same way that was mentioned in the login functionality. There are currently no requirements for what information needs to be filled in. However, if the username exists in the database it will not be created. When the registration happens, the program will redirect back to the login page.



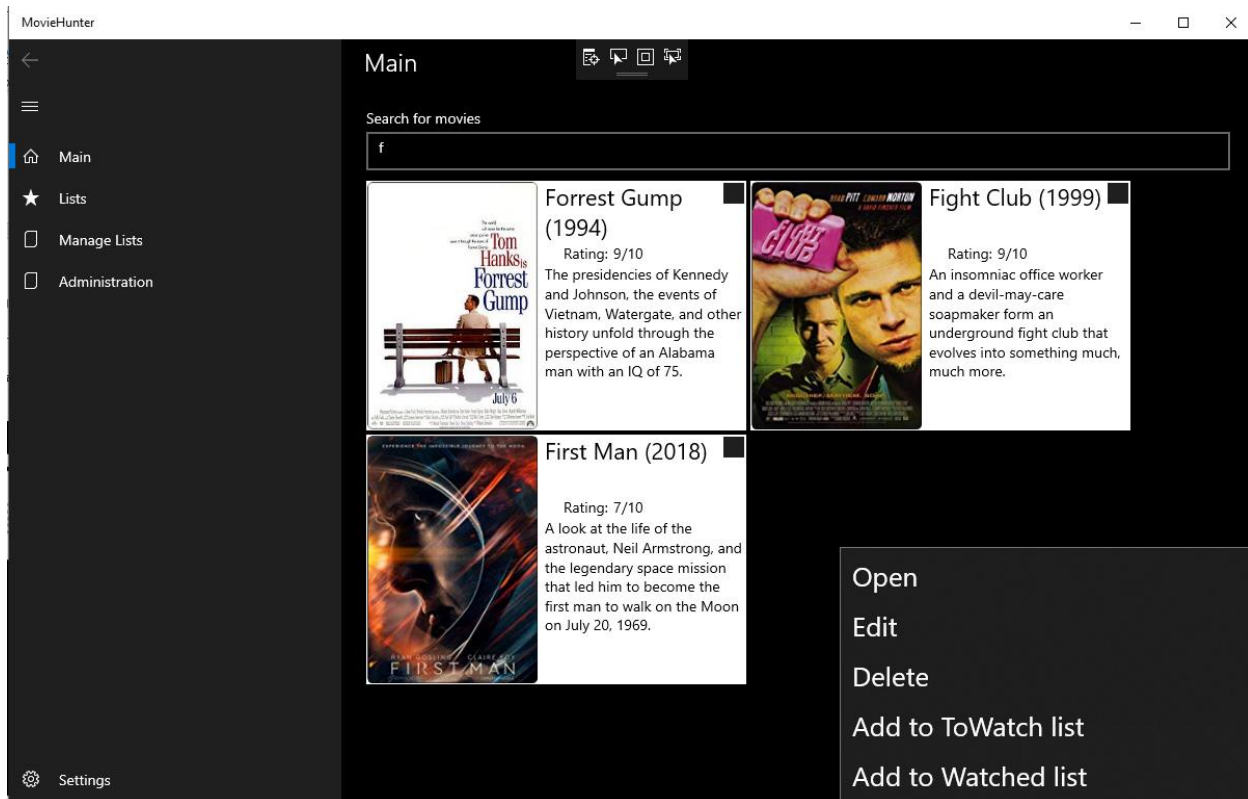
The image shows a registration form with a dark background. It contains five input fields for 'User name:', 'First Name:', 'Last Name:', 'Password:', and 'Password:'. The first three fields are filled with 'admin', 'Admin', and 'Adminsen' respectively. The two password fields are filled with masked characters (dots). Below the fields are two buttons: 'Login' and 'Register'.

User name:	admin
First Name:	Admin
Last Name:	Adminsen
Password:	●●●●●●●●
Password:	●●●●●●●●

Login Register

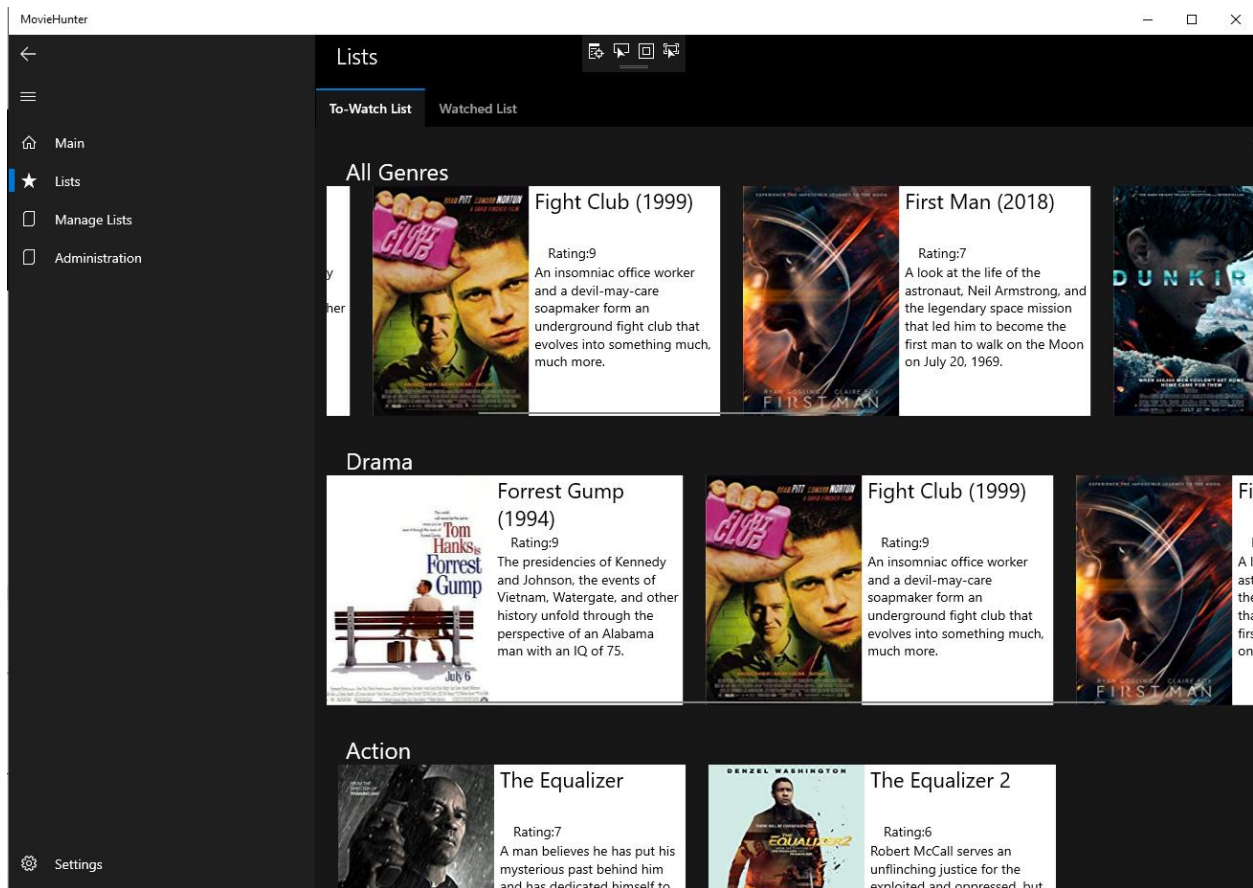
MAIN PAGE

The main page consists of all the Movie objects that are created. All movies are publicly available to all logged in users. On the top of the page there is a search box. Every time the text changes, the program calls the API and gets a new list of movies. The movies are then dynamically presented below. When the user clicks on a movie a popup window will appear. This gives the user the possibility to choose between opening the movie, or selecting more movies. In the bottom corner there is a combo box. When the user clicks on this box a selection of options appear. Open and Edit only works if one item is selected. The "Delete" and "Add to list" buttons work with multiple items. If a movie is deleted the API also deletes the movies from the lists they are saved in (ToWatch and Watched).



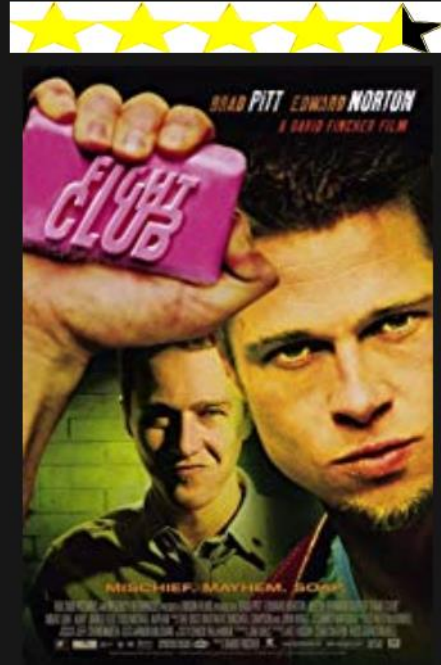
LIST PAGE

The list page consists of two tabs. One for displaying the “To Watch” page, and another for “To Watched”. Inside these tabs there are the movies are sorted into categories. This happens dynamically, so if the user adds a new movie to a new genre, a new scrollable category appears. If the user clicks on a movie they will be redirected to the movie page.



MOVIE PAGE

When a user opens a movie they are presented with the page shown below. At the top right there is an image of stars. These are relative to the movies rating. The white part is a picture which has transparent start. Behind the white image there is a yellow rectangle box. This box will get wider depending on the rating. This way it will always give a precise star rating for each movie. Since the database rating values are integers (byte) the rating will always be a rounded number. If the user scrolls down on the movie page there are two buttons ("Other button" and "Watch trailer"). The other button just shows a popup with shows the users token string. The watch trailer button opens up a youtube video. The youtube video is not linked to the movie in this version of the application.

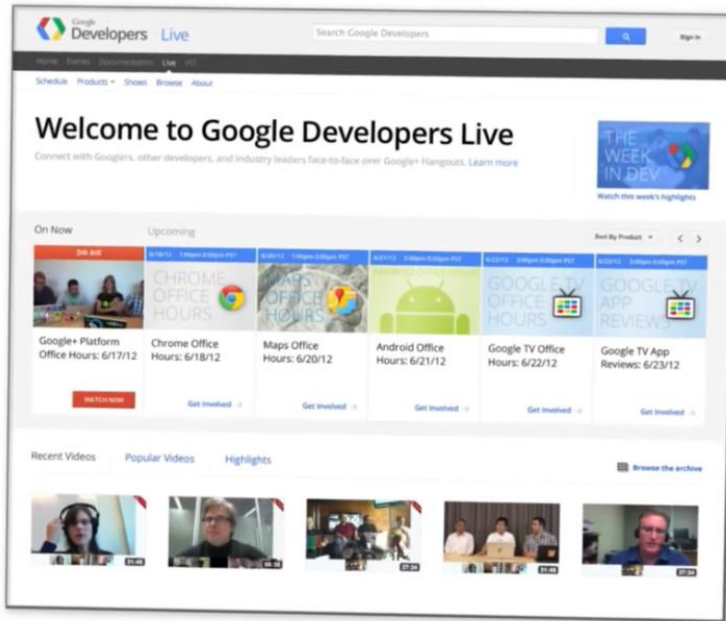
Fight Club (1999)**Category** Drama**Rating** 9**Director** David Fincher**Writers** Chuck Palahniuk**Star actor** Brad Pitt

An insomniac office worker and a devil-may-care soapmaker form an underground fight club that evolves into something much, much more.

Close popup

YouTube Developers Live: HTML5 at YouTube

Watch later Share



The screenshot displays the Google Developers Live website. At the top, there's a navigation bar with links for Home, Events, Documentation, and Live. Below this, a large heading reads "Welcome to Google Developers Live" with a subtext: "Connect with Googlers, other developers, and industry leaders face-to-face over Google+ Hangouts. Learn more". To the right of the heading is a "THE WEEK IN DEV" badge. The main content area is divided into "On Now" and "Upcoming" sections. The "Upcoming" section lists several "Office Hours" events: Google+ Platform (6/17/12), Chrome Office (6/18/12), Maps Office (6/20/12), Android Office (6/21/12), Google TV Office (6/22/12), and Google TV App Reviews (6/23/12). Each event has a "Get involved" button. Below the "Upcoming" section, there's a "Recent Videos" section with a grid of video thumbnails. A "Browse the archive" link is also present. The Google Developers logo is visible in the bottom right corner of the screenshot.

0:14 / 45:35

CC BY YouTube

ADMINISTRATION PAGE

There are three pivot “tab” items: “Add movies to the database”, “Add a person to the database” and “Add Genre to the database”

ADD MOVIES TO DATABASE

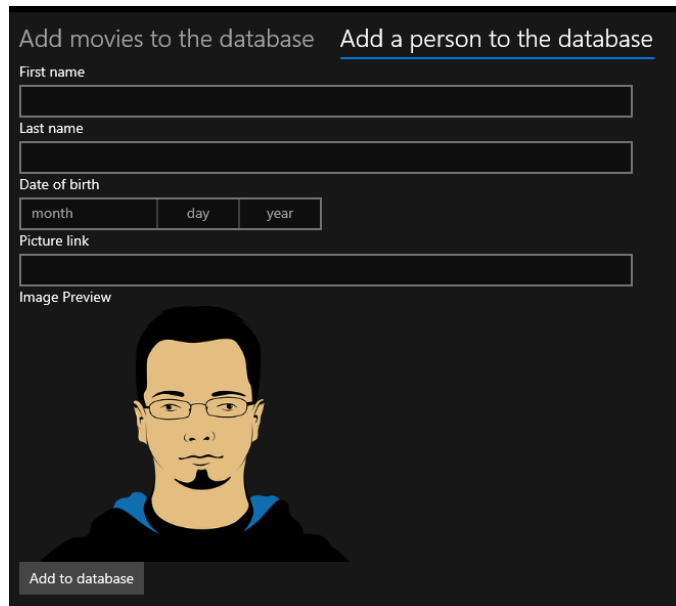
On this page users can create new Movies. The Cover image link will automatically update the image below, so that a user can see a preview on how it will look like. The genre, director, writer, and star actor box work enables searching by first name. The movie rating is chosen with help of a slider. This page has as a requirement that all fields must be filled in. If something is missing, the user will get an output message specifying what is wrong. When the movie is added, it will display on the main page.

The screenshot shows a web application interface for adding movies to a database. The page is titled 'Administration' and has three tabs: 'Add movies to the database' (selected), 'Add a person to the database', and 'Add Genre to the database'. The form contains the following fields:

- Title:** A text input field containing 'Terminator: Dark Fate (2019)'.
- Cover image link:** A text input field containing a long Amazon image URL.
- Image Preview:** A small image showing the movie poster for Terminator: Dark Fate.
- Genre:** A dropdown menu with '14: SciFi' selected.
- Summary:** A text input field containing 'Sarah Connor has returned from far away, and she's gearing up with a team of agents who will fight against T-1000.'
- Movie Rating:** A slider control set to 8.
- Director:** A dropdown menu with '13: Frank Darabont' selected.
- Writer:** A dropdown menu with '1: Fredrik Forsell' selected.
- Search:** A search bar with 'fredr' entered and a search icon.
- Add to database:** A button at the bottom of the form.

ADD A PERSON TO THE DATABASE

The functionality on this page is similar to the one on “Add movies”. The only difference is that this one uses a date picker for picking the date of birth.

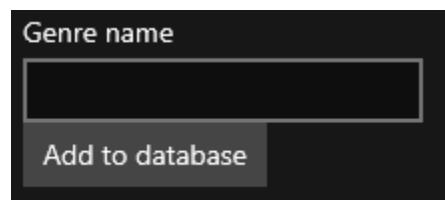


The screenshot shows a web form titled "Add movies to the database" and "Add a person to the database". The form has the following fields:

- First name: A text input field.
- Last name: A text input field.
- Date of birth: A date picker with three sub-fields for month, day, and year.
- Picture link: A text input field.
- Image Preview: A preview of a person's image, showing a man with glasses and a goatee.
- Add to database: A button at the bottom left of the form.

ADD GENRES TO THE DATABASE

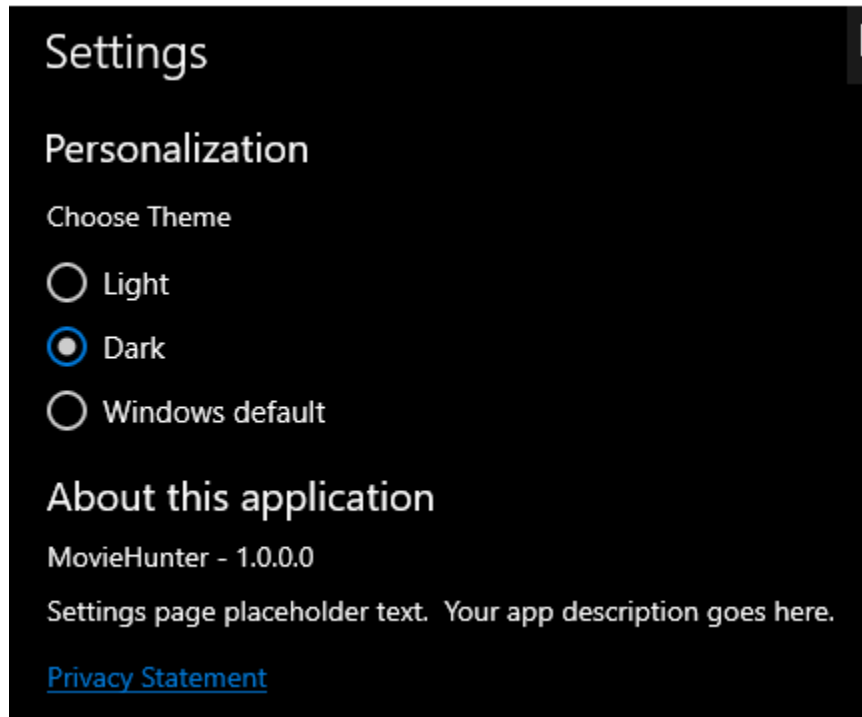
This page contain 1 text box for deciding the genre name.



The screenshot shows a web form titled "Genre name". It has a single text input field for the genre name and an "Add to database" button below it.

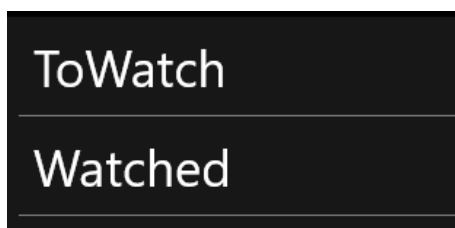
SETTINGS PAGE

The settings page was created with the template “Windows template studio”. This added the possibility to change the color theme. When designing this app, there has been focus on using variables for defining the colors. This way every page will be supporting a theme change. I a default it will be black, but the settings on the windows computer can force applications to run in dark mode if that is the windows theme.



MANAGE LIST PAGE

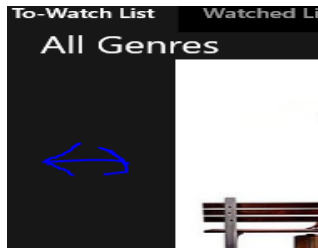
This page has a listview for all the lists the user has. When clicked on one of these items, the user will be redirected to the list page (not to be confused with the one I wrote about above). This list is displayed in the picture below (to the right). This one is not fully implemented as the buttons are not working.



Delete	Move to other list	Dunkirk (2017)
Delete	Move to other list	First Man (2018)
Delete	Move to other list	Forrest Gump (1994)
Delete	Move to other list	Fight Club (1999)
Delete	Move to other list	The Equalizer 2
Delete	Move to other list	The Equalizer

KNOWN ISSUES

- The administration page is not fully implemented, and the users has few input requirements. The same goes with the registration page. A user can theoretically be created with no username and password, but have a user id.
- At the list page where genres are sorted: When swiping through movies while also moving the mouse out of view, it can sometimes lock in the position illustrated below.



- When deleting a bunch of movies from the main page at the same time, instead of sending in a list of object to the API, it sends a delete request for every single movie.
- The watch trailer popup that occurs when pressing the button on the movie page will open in as the full size of the app minus the width of the menu on the left. If the menu is closed, it wont fill the screen.
- I have been using the Microsoft Extended design guidelines, however I have 3 errors that I haven't fixed.

	Code	Description
!		The certificate specified is not valid for signing. For more information about valid certificates, see http://go.microsoft.com
!		Certificate file 'MovieHunter_TemporaryKey.pfx' not found.
!	CS1030	#warning: 'To protect potentially sensitive information in your connection string, you should move it out of source code.

DATABASE

