

# MovieLand

Enrique Belenguer

[40406742@live.napier.ac.uk](mailto:40406742@live.napier.ac.uk)

Edinburgh Napier University – Advanced Web Technology (SET09103)

## Introduction

Movieland is an application which allows users to find information about Hollywood movies released between 1993 and 2008. The application allow user analyse movies, which can be divided by gender or year, and have information such as the director, producer, casting and plot. You can also access the profiles of different actors and actresses and from where you can see some information about them and the movies in which they have worked. All the dynamic content offered to the user is over different HTML templates and obtained from a static json file.

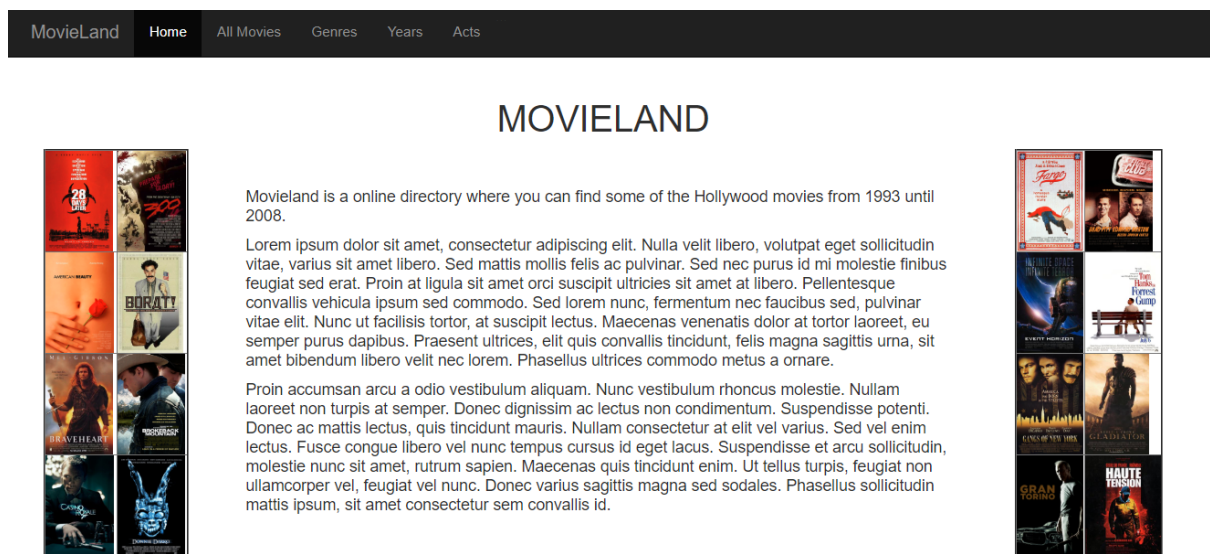


Figure 1. Home Page

## Design

The design of my website is quite simple and designed to facilitate the user's experience. In the upper part the user finds a top-menu with the different sections of the web page, with the different ways of classifying the films and a link to the page with the actors and actresses.

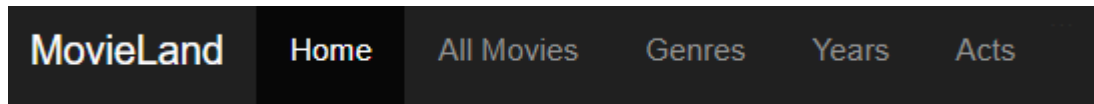
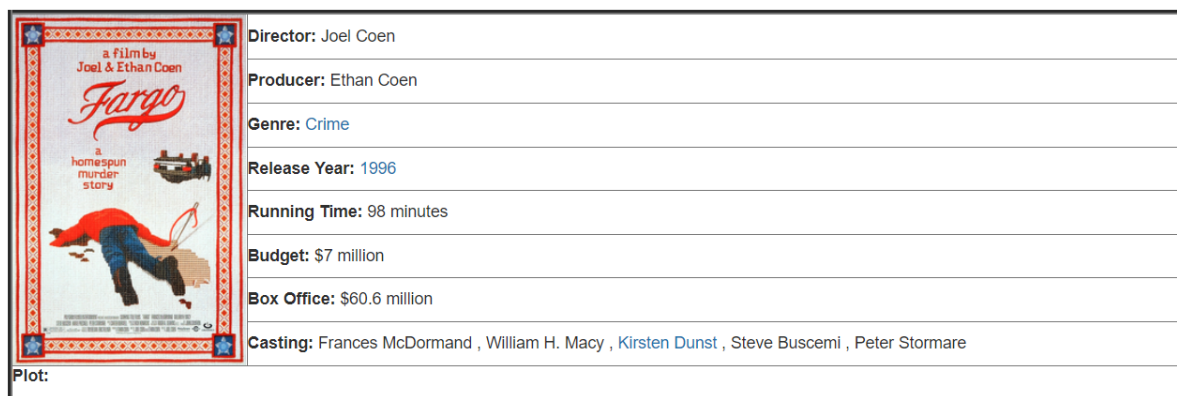


Figure 2. Top-Menu

The characteristics of each page are quite similar, and the colour scheme is the same, which helps to give consistency to the web. This helps the user to feel comfortable and not feel any distraction, which helps you to be focused on the content of this.

The same happens with the information it contains. The user can access a movie, and from there he/she can access (with its corresponding links) to all the films made that year, all the films of the same genre or even the profile of one of the actors or actresses that star on it.



<b>Director:</b> Joel Coen
<b>Producer:</b> Ethan Coen
<b>Genre:</b> <a href="#">Crime</a>
<b>Release Year:</b> <a href="#">1996</a>
<b>Running Time:</b> 98 minutes
<b>Budget:</b> \$7 million
<b>Box Office:</b> \$60.6 million
<b>Casting:</b> <a href="#">Frances McDormand</a> , <a href="#">William H. Macy</a> , <a href="#">Kirsten Dunst</a> , <a href="#">Steve Buscemi</a> , <a href="#">Peter Stormare</a>
<b>Plot:</b>

Figure 3. Movie info Page with its links

Another possible problem was to try to keep the user always well informed about their situation within the website. For this I used an easy and intuitive URL pattern together with a highlight of the top-menu.

Even so, in case the user types a bad URL, he will receive an appropriate message to communicate his error. Although in cases such as an error in writing the year or gender, the page will automatically redirect to the page with all years or all genres.

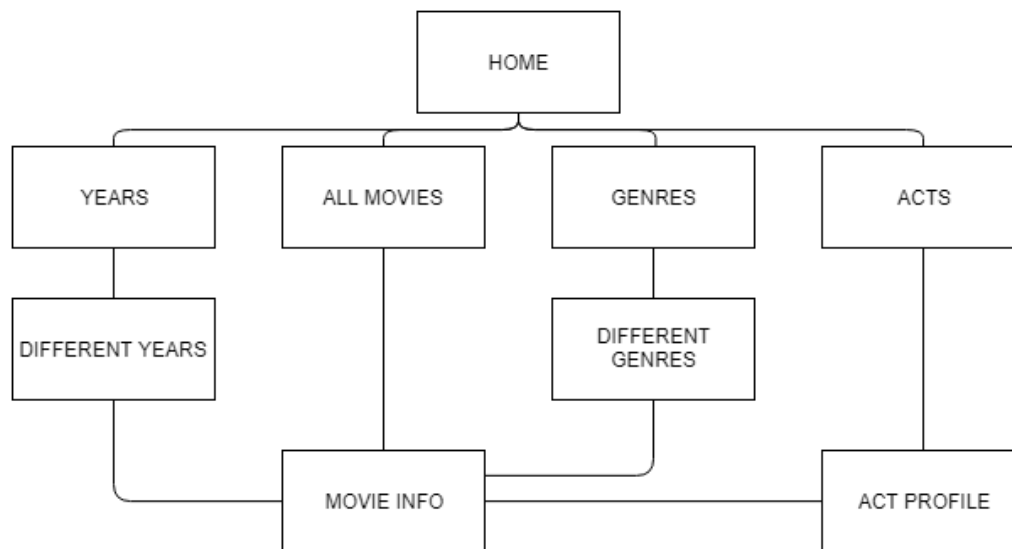


Figure 4. Navigation Map

## Enhancements

The page could have many improvements, although the most serious would be to add information from a form to a json file. During the first week I was trying to do it, and due to my ignorance of the json files treatment I did not get it. Although I have to say that I have found a way to do it and it seems to be very difficult<sup>[1]</sup><sup>[2]</sup>.

Another of the improvements that could and should have been implemented is the search bar, which would be quite useful for the user.

## Critical Evaluation

Probably one of the features that works best on the web and which I feel most proud of is the use of URL variables. This has allowed me to show a lot of dynamic content using the same template.

Also, I must say that the fact of loading the json file every time I change the page is not very professional, pr. Like the failure to show the appropriate error message in case the user makes a mistake by entering the title of the movie in the URL.

Likewise, I must say that the good use of json files has allowed me to have enough links easily getting the user to move from one page to another easily and making the page quite easy to improve in the future.

```

@app.route("/allmovies/<mov>/")
def movie(mov=None):
    INDEX = os.path.realpath(os.path.dirname(__file__))
    with open(os.path.join(INDEX, 'static', 'data.json')) as f:
        data = json.load(f)
        return render_template('movieInfo.html', data=data, mov=mov), 200

@app.route("/allmovies/act/")
def listAct(mov=None):
    INDEX = os.path.realpath(os.path.dirname(__file__))
    with open(os.path.join(INDEX, 'static', 'data.json')) as f:
        data = json.load(f)
        return render_template('listAct.html', data=data, mov=mov), 200

@app.route("/allmovies/act/<act>/")
def actInfo(act=None):
    INDEX = os.path.realpath(os.path.dirname(__file__))
    with open(os.path.join(INDEX, 'static', 'data.json')) as f:
        data = json.load(f)
        return render_template('actInfo.html', data=data, act=act), 200

```

Figure 5. URL variables

## Personal Evaluation

I have improved many of my skills during the development of the project but I'm not happy with what I have achieved.

I have used new tools like GitHub, Vim as a text editor and Bootstrap framework. I also refreshed others like Linux and HTML and I have improved well my knowledge about CSS and URL and how to treat json files. I didn't manage my time well and the last two weeks were quite stressful because of it.

Overall the application meets the coursework criteria. But after seeing some of my class mates works, I think I should have spent a little bit more of time from the beginning. That is something I will remember for the next coursework and I will try to learn from my old mistakes.

## References

[1] <http://jsbin.com/kifozy/edit?html,console,output>

[2] <https://stackoverflow.com/questions/2835559/parsing-values-from-a-json-file>