SET09103

Advanced Web Technologies

Coursework One

Fergal Sexton

40092783

Introduction

The web application developed is an online catalogue of Movies. The catalogue allows the user to view movies via genres and can also view upcoming and top rated movies. The application is developed using Python, Flask (Python Micro framework), Jinja2 (Python Template Engine), HTML, CSS, JavaScript and SQLite.

Design

The website is split into two sections. The header is contained in /templates/ and consists of the navigation bar which runs across the top of the website and uses the user's width of their screen. The navigation is split into the following

Explore Movies:
- Top Rated
- Upcoming

Movie Genres:
- Action
- Adventure
- Crime
- Comedy
- Drama
- Family
- Horror
- Romance
- Science Fiction
- Superhero

Upon clicking on a links within the navigation – the user is brought to another page which consists of the movie type they requested. Each page is generated using a SQL query powered by SQLite. When a user clicks on a movie genre, the app route function in flask inserts the requested genre which in turn queries the SQLite database

```
@app.route('/movies/genres/<name>/')
def genres(name=None):
    g.db = connect_db()
    cur = g.db.execute('select id, movie_name, poster from movies where
                        category = ?', [name])
    movies = [dict(id=row[0], movie_name=row[1], poster=row[2]) for row in
              cur.fetchall()]
    g.db.close()
    return render_template('genres.html', movies=movies, name=name)
```

The URL hierarchy is represented by the genre the user is currently viewing. For example if the user is viewing the horror genre it will be displayed as http://0.0.0.0:5000/movies/genres/Horror/

The top rated page is queried by a rating between 0 – 5. 0 stands for a movie that has not been released and 5 being the best type of movie.

Upcoming movies display any movies with a 2016 release date.

The second part of the design is the main body of the website. This where all information is generated once the user clicks on a link in the navigation or one of the movies displayed in this section.

The index page is generated by randomising the items in the database. It selects 14 movies out of over 40 and displays them in a random order. This allows for the user to view movies in genres they may not have selected.

The title of the movie is displayed when the user hovers their mouse over the movie's poster and allows them to click on the poster to reveal the movie's full details such as

- Name
- Director
- Year
- Gross
- Poster
- Category
- Studio
- Rating
- Synopsis

The poster images are stored in /static/images/posters

All this information is held in the movieinfo.db database and can be displayed anywhere by querying the database using Flask, inserting the information into a dictionary and displaying it using jinja syntax.

I added error handling for the following errors:

401 – Unauthorised
400 – Bad Request
404 – Not Found
500 – Internal Server Error

I purposefully added the non-existent About page to the navigation to demonstrate the 404 error handling.

Enhancements

Within the database I created the column year to display the year the movie was released. Rather than using a TEXT column, I should have opted for a date column. This would have enabled me to display upcoming movies based on the present day rather than a set year.

I would have also liked to have developed a "My List" section where a user could save movies to their own personal list which they could view to help them choose a movie to watch in the future.

The database could have also contained more information such as the movie's actors, writers, release dates and certificates. This would give the user more information about the movie. A link to a website which contains more information would also be useful i.e. IMDB.

Allowing a user to add reviews to each movie would also help generate the rating system rather than being dependent on my biased ratings!

Personal Evaluation

Overall I found the project to be interesting and enjoyed developing the idea and overcoming the obstacles that I encountered during the development.

Once I had familiarised myself with Python and Flask, I felt I was more confident in my ability to develop something robust and a project that could eventually be used by others. This is the first time I have used a framework within a web application as I tend to develop everything myself. It has taught me to research frameworks before beginning a project and see if there has been anything suitable previously created which would assist me in future projects. As I was told in my first year in Napier University "Don't try to re-invent the wheel".

One of the biggest issues I found was inserting the variable into the SQL query. This took me the most time to solve and is one of the key features of the application. In the end I was not calling the variable correctly as I was using a method used in another programming language and not the correct way in python. Something very easily to overlook but a valuable lesson learned.

I decided to use Bootstrap for my design as it was something I had never used before but one I will most definitely be using again. The documentation is very easy to follow on their website and made the website extremely presentable without having to style the website entirely myself. I feel I have only scratched the surface with what it can achieve but plan to implement it further in future projects.

I'm looking forward to further developing my python flask skills and seeing what else I can achieve now that I have a foundation in place.

Summary

- Flask Documentation http://flask.pocoo.org/docs/0.10/
- Bootstrap http://getbootstrap.com/ (MIT License)
- iHover CSS https://github.com/gudh/ihover (MIT License)
- Star Icons http://findicons.com/icon/168914/knewstuff Designer: YellowIcon, Icon Pack: Crystal Project License: GNU/GPL