

CS235 Assignment 2

Johnathan Leung (jleu075)

Link: <https://github.com/TheImaginaryOne/movies-py>

Project structure

The web server (Flask server) is split into multiple layers, which is the repository (data), services, views (with templates). Separating the application into these layers ensures a strong separation of concerns and the single responsibility principle.

The services connect to the repository layer as well as the other data storage systems, such as flask.session (to manage cookies for user login). It also exposes some functions to the view layer, to allow the views to interact with the “business logic”. This layer serves as the layer that connects the data to the view.

As of now, there are services for handling user login and searching for movies and adding reviews.

The repository contains all the data and provides (simple) methods to access and edit the data. Currently the repository is an in-memory database in a class called “MemoryRepository”, but it implements an interface called “Repository”, with methods such as get_user and add_review. If a real storage database is implemented in the future (eg. SQLite), then it will also implement the same “Repository” interface such that it can be easily swapped with a Memory Repository without breaking the program.

This implements the Dependency Inversion Principle, because the service layer depends on the Repository abstraction instead of the MemoryRepository directly, so there is some decoupling

The endpoints, which are all the http routes, act as views that the browser can request. These endpoints also connect to the Jinja templates, which is the view which the end user actually sees. The code in the endpoints are simple because the important logic is delegated to the service layer.

This view layer is split into Flask “Blueprints” so that views with related functionality are put together. Specifically, there is a “user” blueprint, which has user login and registration, as well as the “movies” blueprint (at url /movies), which contains movie search, movie display and movie reviews.

Extra feature


The website displays movie posters on the movie information page. The posters are fetched from the TMDB (The Movie Database) website. Firstly, a function (in the service layer) sends a request to the TMDB API to find the poster url for a certain movie name. Then it will inject the poster url into the template.

```
response = requests.get(f'https://api.themoviedb.org/3/search/movie',
                        params={
                            'api_key': key,
                            'query': movie.title,
                            'year': movie.release_year
                        })
contents = response.json()
poster_path = contents['results'][0]['poster_path']

return f'http://image.tmdb.org/t/p/w500/{poster_path}'
```

A screenshot is given here

Fresh Oranges [Search Movies](#) [Hello, user1234](#) [Log out](#)



Guardians of the Galaxy (2014)

Metascore	IMDB Rating
76.0 /100	8.1 /10 (757074 votes)

Director James Gunn
Actors Chris Pratt, Vin Diesel, Bradley Cooper, Zoe Saldana
Revenue \$333.13 million
Genres Action, Adventure, Sci-Fi

A group of intergalactic criminals are forced to work together to stop a fanatical warrior from taking control of the universe.

Reviews

[Add/edit your review](#)

user1234 rated it 8/10
cool movie!