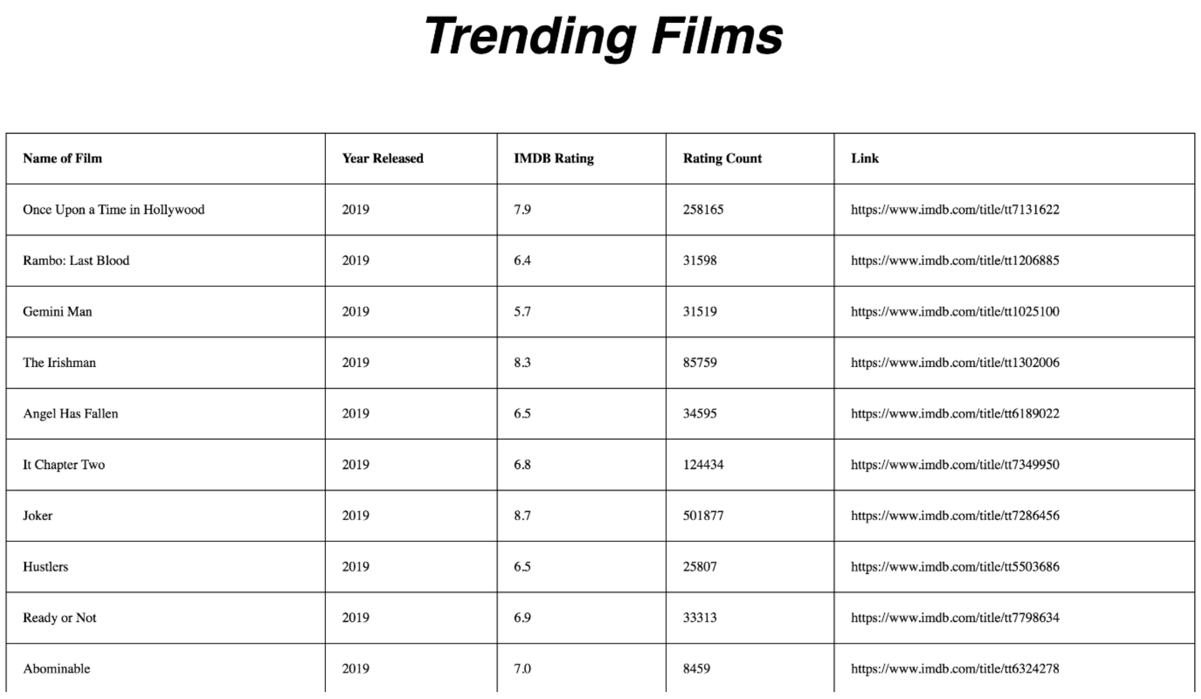
Report

<u>The differences between the initial plan and the final implementation.</u>

The initial plan for this project was to produce a web application for the top trending films on the current date. The approach described was to research different APIs for retrieving valid data for use on the webpage.

The services I chose to use were 2 recommended APIs on the RapidAPI site. To get trending information I chose to use Uflixit, which had a ‘wanted’ endpoint, which returns the top 10 trending films based on ratings by IMDb, Rotten Tomatoes and Meta Critic. For future research, I would seek to understand their algorithm for choosing the 10 and possibly look at other services offering this. However, this was the only one on the RapidAPI that has this type of endpoint, so I chose to use it due to time constraints. The second service I used was IMDb API which would return ratings for the top 10 titles described above.

On the table of contents for the trending films page, I was not able to gather all the data described based off using the API, meaning I had to reduce the number of columns in the table. This of course means that there is less information being provided the user, however with the problem that occurred I believe I adapted well regarding the time-frame, providing useful information to the user about the top trending films to the current date.



Trending Films

Name of Film	Year Released	IMDB Rating	Rating Count	Link
Once Upon a Time in Hollywood	2019	7.9	258165	https://www.imdb.com/title/tt7131622
Rambo: Last Blood	2019	6.4	31598	https://www.imdb.com/title/tt1206885
Gemini Man	2019	5.7	31519	https://www.imdb.com/title/tt1025100
The Irishman	2019	8.3	85759	https://www.imdb.com/title/tt1302006
Angel Has Fallen	2019	6.5	34595	https://www.imdb.com/title/tt6189022
It Chapter Two	2019	6.8	124434	https://www.imdb.com/title/tt7349950
Joker	2019	8.7	501877	https://www.imdb.com/title/tt7286456
Hustlers	2019	6.5	25807	https://www.imdb.com/title/tt5503686
Ready or Not	2019	6.9	33313	https://www.imdb.com/title/tt7798634
Abominable	2019	7.0	8459	https://www.imdb.com/title/tt6324278

<u>Reflect upon the challenges faced and achievements made during this assessment</u>

A problem that I ran into was actually retrieving this information due to issues with RapidAPI (a service that supplies the APIs to the customers). I took example code snippets in python from the API documentation and used them in my python code. However, for both APIs, after installing the 'requests' package upon which they depend, on execution of the script I got the error "TypeError TypeError: 'Request' object is not callable".

I tried to research the error but couldn't find support on this in time, so I decided to take the approach of using the Test Endpoint function provided by the APIs to run these manually. This required taking out a low-cost subscription.

I took the data that I managed to retrieve from the endpoints and hardcoded this into lists so I could prove my concept still. Although I wasn't able to call the endpoints on an automatic basis, I was still able to implement the flask and jinja features covered on the course in my solution.

<u>Describe any features that you would add.</u>

If I was to complete this project again, I would make sure I could test functionality of the APIs before detailing them in my plan. This way, I would find the problems early and rectify, or find an alternative source of data. However, with little time producing a project, learning python for the first time and other modules to complete, it seemed to be an unrealistic task that I set myself and would re-consider my approach to gathering data or like I said testing the APIs before coursework 1 was implemented.

My first step would be to get the endpoints working by communicating with the RapidAPI support teams. Their website welcomes feedback on people's projects and they've offered an email address and Facebook page.

> "*We'd love to hear your thoughts, feedback or even just news around your latest project.*
>
> *Send us a message at support@rapidapi.com or contact us via our <u>Facebook page</u>*"

Another feature that I would add is implementing both the Trending Music and Trending TV pages using the same table format as the trending films. This feature would be implemented in a similar way as described in coursework 1, gathering data from relevant APIs and displaying the data into a table.

If I had more time during this project, I would put more effort in the user interface instead of focusing heavily on gathering the data to implement onto the site. I would style the table of contents, so it looks more attractive. However, because the main priority of this project was to gather data from an external source using python, I didn't put as much time into creating an aesthetically pleasing table of contents.