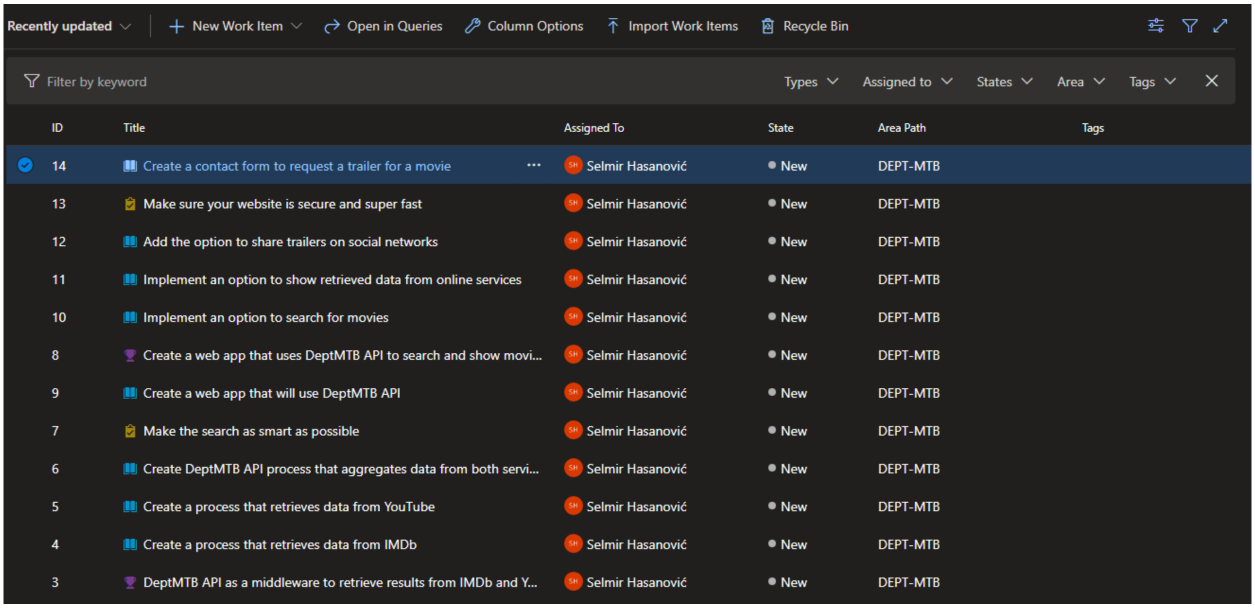
Dept Movie Trailer Browser – a project by Selmir Hasanović

Hello. :) Below I will guide you through my thinking process and implementation of a task I was assigned as a part of the interview.

First of all, I will share important links with you.

- GitHub repositories:
  - DeptMTB.API
  - DeptMTB
- Azure DevOps Project (I will invite Lana as a collaborator initially, and you may provide me with additional people to add).

My first step was to create an Azure DevOps project and organize all the parts of an assignment into features, stories, and tasks. In my work, I rely heavily on Azure DevOps (or similar tools) to help me keep track of everything related to the project, and it helps me to be organized, and complete all my tasks.

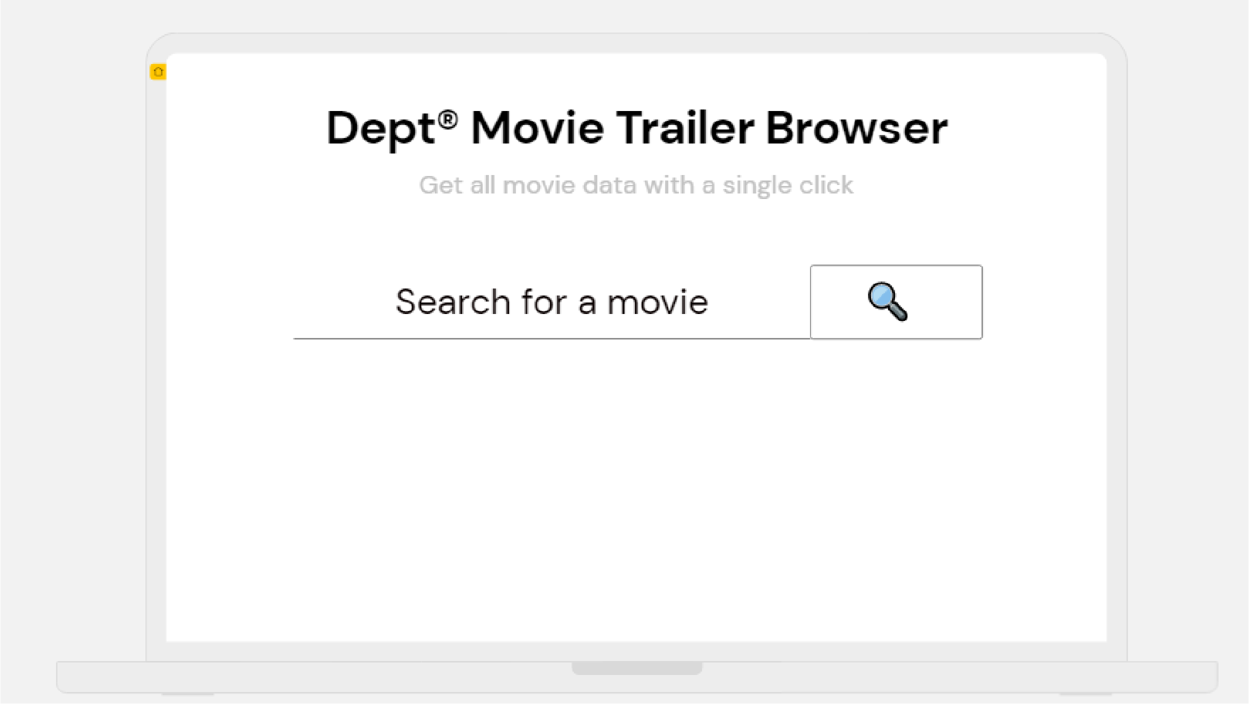


ID	Title	Assigned To	State	Area Path	Tags
14	Create a contact form to request a trailer for a movie	Selmir Hasanović	New	DEPT-MTB	
13	Make sure your website is secure and super fast	Selmir Hasanović	New	DEPT-MTB	
12	Add the option to share trailers on social networks	Selmir Hasanović	New	DEPT-MTB	
11	Implement an option to show retrieved data from online services	Selmir Hasanović	New	DEPT-MTB	
10	Implement an option to search for movies	Selmir Hasanović	New	DEPT-MTB	
8	Create a web app that uses DeptMTB API to search and show movi...	Selmir Hasanović	New	DEPT-MTB	
9	Create a web app that will use DeptMTB API	Selmir Hasanović	New	DEPT-MTB	
7	Make the search as smart as possible	Selmir Hasanović	New	DEPT-MTB	
6	Create DeptMTB API process that aggregates data from both servi...	Selmir Hasanović	New	DEPT-MTB	
5	Create a process that retrieves data from YouTube	Selmir Hasanović	New	DEPT-MTB	
4	Create a process that retrieves data from IMDb	Selmir Hasanović	New	DEPT-MTB	
3	DeptMTB API as a middleware to retrieve results from IMDb and Y...	Selmir Hasanović	New	DEPT-MTB	

Picture 1. - Initial Azure DevOps setup

After breaking down an assignment into smaller parts, and making sure I understand all requirements, the next step was to get to know IMDb and YouTube APIs by thoroughly reading documentation and choosing which API endpoints I will need since I haven't used them before. Small digression, in high school I created a small website called SMDb (which stands for Selmir Movie Database), where I kept track of all the movies I've watched. :) Hence the decision to use IMDb API.

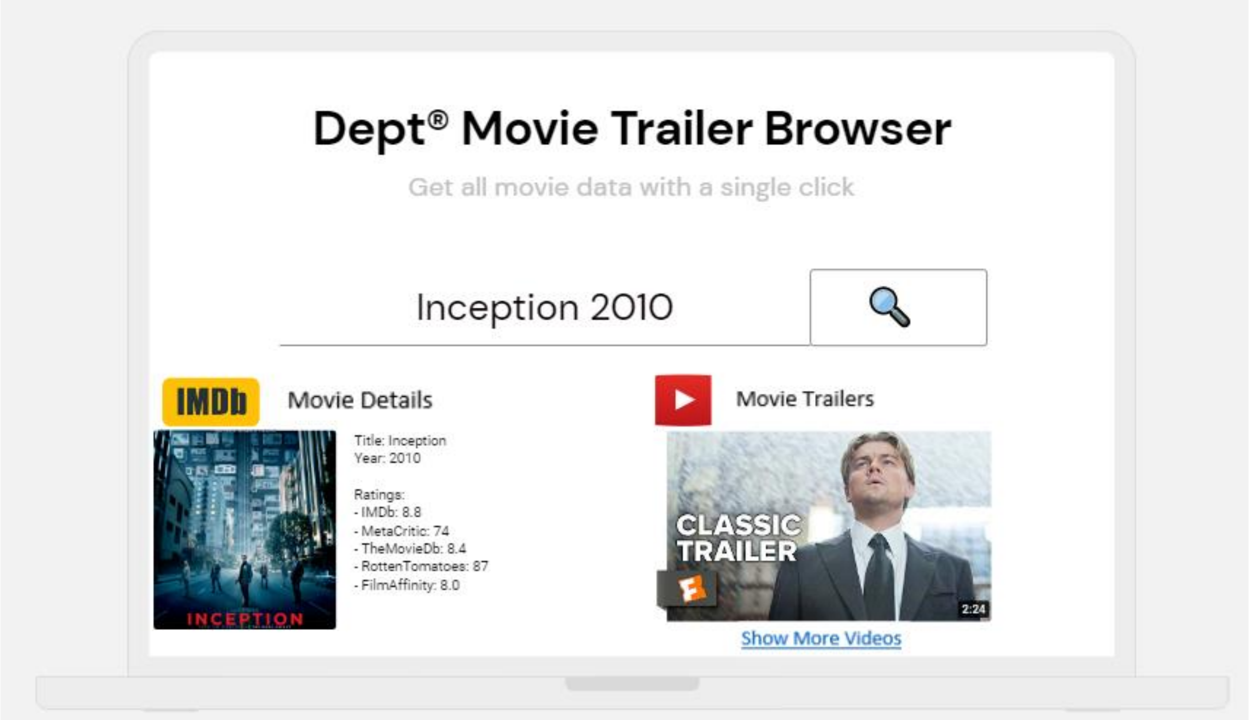
The next step was to create some mockups because I like to have a clear picture of what I need to create in the front-end. I created two simple mockups which will serve as a guideline for how my website should look.



Dept® Movie Trailer Browser

Get all movie data with a single click

Picture 2. - Mockup I



Dept® Movie Trailer Browser

Get all movie data with a single click

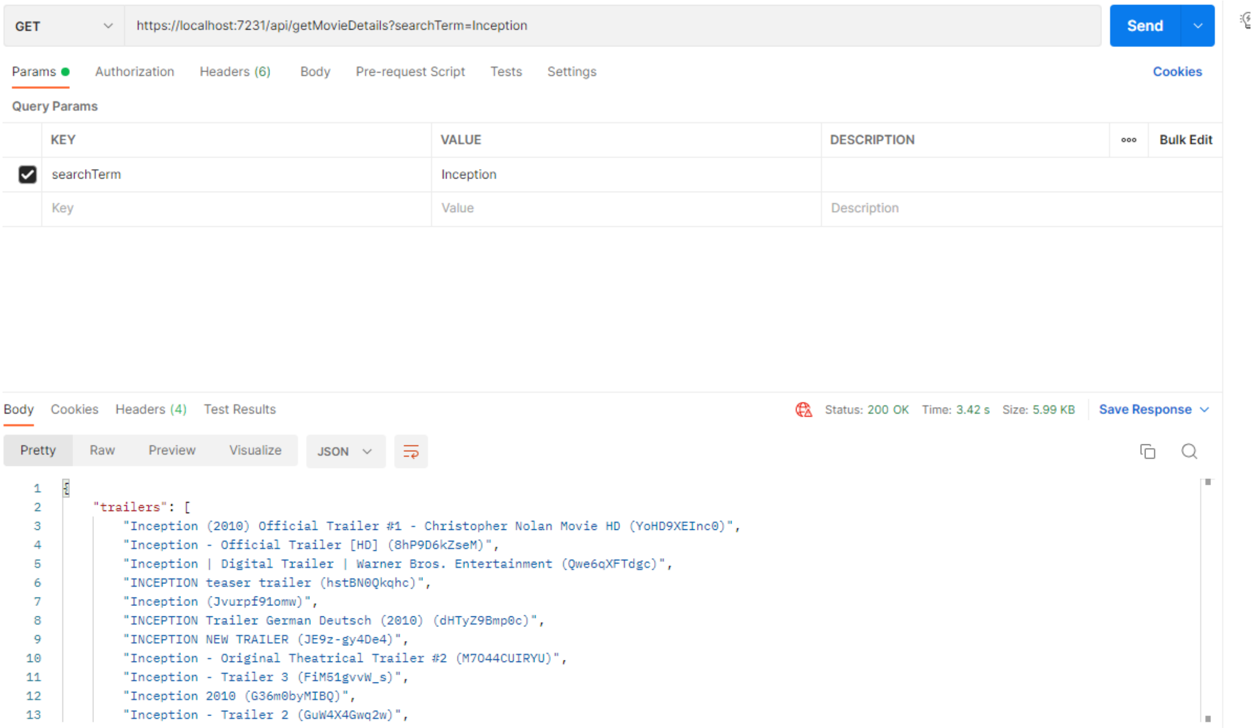
Inception 2010

Title: Inception
Year: 2010

Ratings:
- IMDb: 8.8
- MetaCritic: 74
- TheMovieDb: 8.4
- RottenTomatoes: 87
- FilmAffinity: 8.0

Picture 2. - Mockup II

To test the created API I used Postman. After implementing caching of the aggregated data, the response time of the request went from 3.42s to 15ms.



GET https://localhost:7231/api/getMovieDetails?searchTerm=Inception

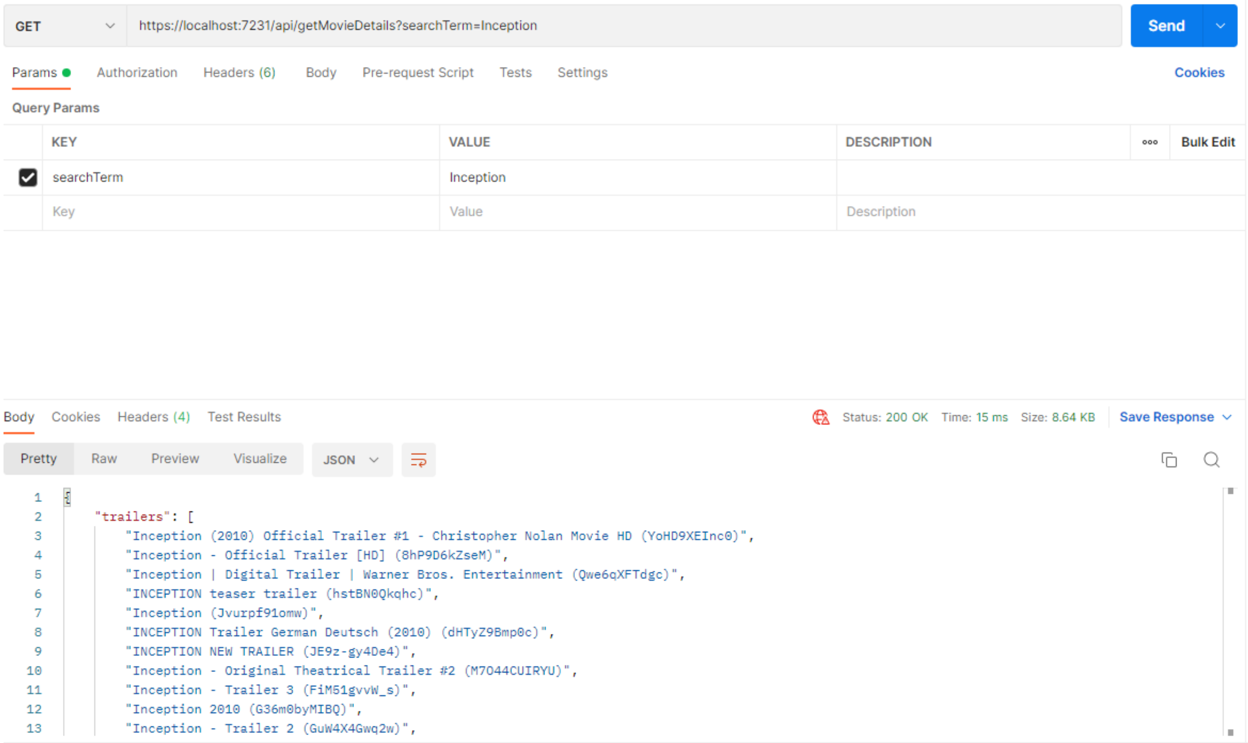
Query Params

	KEY	VALUE	DESCRIPTION
☑	searchTerm	Inception	
	Key	Value	Description

Status: 200 OK Time: 3.42 s Size: 5.99 KB

```
{
    "trailers": [
        "Inception (2010) Official Trailer #1 - Christopher Nolan Movie HD (YoHD9XEInc0)",
        "Inception - Official Trailer [HD] (8hP9D6kZseM)",
        "Inception | Digital Trailer | Warner Bros. Entertainment (Qwe6qXFTdgc)",
        "INCEPTION teaser trailer (hstBN0Qkqhc)",
        "Inception (Jvurpf91omw)",
        "INCEPTION Trailer German Deutsch (2010) (dHTyZ9Bmp0c)",
        "INCEPTION NEW TRAILER (JE9z-gy4De4)",
        "Inception - Original Theatrical Trailer #2 (M7O44CUIRYU)",
        "Inception - Trailer 3 (FiM51gvvW_s)",
        "Inception 2010 (G36m0byMIBQ)",
        "Inception - Trailer 2 (GuW4X4Gwq2w)",
```

Picture 3. - API test I



https://localhost:7231/api/getMovieDetails?searchTerm=Inception

Query Params

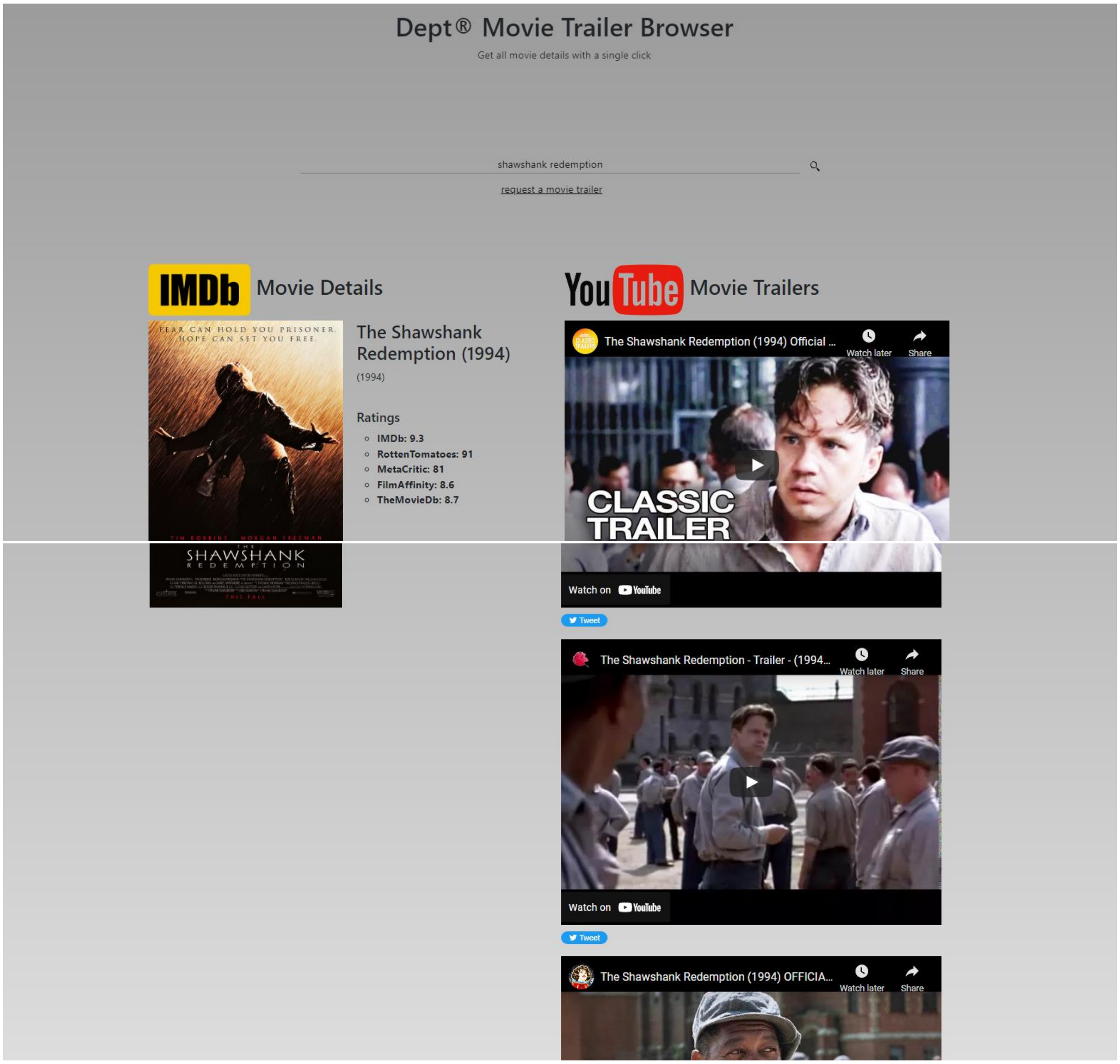
	KEY	VALUE	DESCRIPTION
☑	searchTerm	Inception	
	Key	Value	Description

Status: 200 OK Time: 15 ms Size: 8.64 KB

```
{
    "trailers": [
        "Inception (2010) Official Trailer #1 - Christopher Nolan Movie HD (YoHD9XEInc0)",
        "Inception - Official Trailer [HD] (8hP9D6kZseM)",
        "Inception | Digital Trailer | Warner Bros. Entertainment (Qwe6qXFTdgc)",
        "INCEPTION teaser trailer (hstBN0Qkqhc)",
        "Inception (Jvurpf91omw)",
        "INCEPTION Trailer German Deutsch (2010) (dHTyZ9Bmp0c)",
        "INCEPTION NEW TRAILER (JE9z-gy4De4)",
        "Inception - Original Theatrical Trailer #2 (M7O44CUIRYU)",
        "Inception - Trailer 3 (FiM51gvvW_s)",
        "Inception 2010 (G36m0byMIBQ)",
        "Inception - Trailer 2 (GuW4X4Gwq2w)",
```

Picture 4. - API test II

After final testing, this is how the website functions:



Dept® Movie Trailer Browser

Get all movie details with a single click

shawshank redemption

request a movie trailer

IMDb Movie Details

The Shawshank Redemption (1994)

(1994)

Ratings

- IMDb: 9.3
- RottenTomatoes: 91
- MetaCritic: 81
- FilmAffinity: 8.6
- TheMovieDb: 8.7

YouTube Movie Trailers

The Shawshank Redemption (1994) Official ...

The Shawshank Redemption - Trailer - (1994...

The Shawshank Redemption (1994) OFFICIA...

Picture 5. - Search result example

A brief explanation of some of my decisions during implementation:

- I used IMDb and YouTube APIs and followed their documentation to implement API calls.
- WebApp calls a single method which aggregates data from both APIs.
- I could've implemented some form of authentication, and not left my API completely open, but it doesn't seem necessary.
- I opted to implement caching on both projects, WebApp and WebApi, because it provides me with an option to reduce the number of in-app calls, and reduce response time in API.
- WebApp passes API response to two views, one for IMDb and one for YouTube videos. During the first loading, I decided to show only one video to make the website load faster, but if a user

clicks on the button 'Show more videos' then I decided to show three more videos (if available). Maybe I could've appended additional videos to those already shown, but I don't think it will make a lot of difference, and it was easier this way for me. :)

- I added options to share videos on Facebook and Twitter, as the two most important networks (can't share a video on Instagram anyway), and there is also an option in the embedded video to share it, so user has enough options. :)
- The contact form allows the user to request a trailer to be sent to his email, I implemented everything needed on the front-end, but didn't feel it's necessary to add the back-end for this feature (it would take too much time, and I don't think that's the focus of this task anyway).
- In hindsight, maybe validation could be better, I've decided to not complicate it too much, and just do basic client-side validation where necessary.
- I didn't cover myself in glory while designing UI (as you can see from mockups, and website as well :D), I like minimalistic apps that look neat and work well. Of course, I can implement more eye-pleasing websites, but for that, I would need someone more creative to provide me with mockups. :)
- I decided not to show too much data from IMDb, just some basic information. Generally, it's the same process to retrieve all the data about movies, but if I decided to list directors, cast, and similar details, it would unnecessarily take too much space and additional time.
- There are many comments in the code where I felt they are needed, to make reviewing, and eventually changing the code easier.
- I used code examples and solutions from the internet wherever possible because I don't like reinventing the wheel and trying to show off by implementing everything from scratch. :)
- I didn't surprise you with any new functionality (shame on me :D; to be honest I didn't think of anything that would be a good addition), but I tried to add some touches that make the app look good and professional. :)