

The Movie App



Author:

Jorge García Torralba

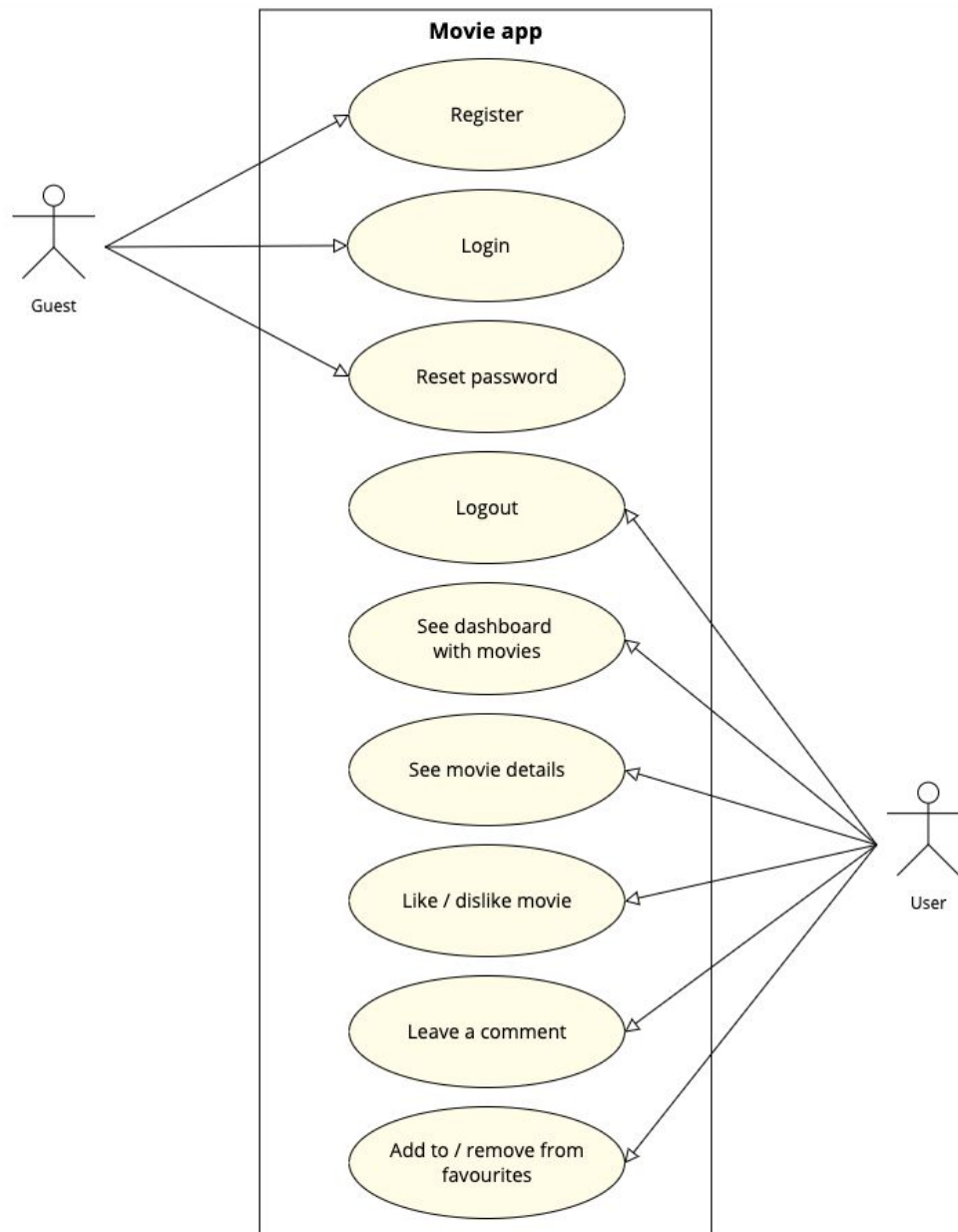
Table of content

Use case diagrams	2
Database diagram	3
Basic wireframe design	4
Login	4
Homepage	5
Movie details	6
Technologies used	7
Front-end	7
Back-end	7
Time estimation	8
Product Analysis	10
SWOT	10
Competition	10
Cost Estimation	11
Changes and unforeseen events	12
Daily log	13
Lessons learned	16
Conclusions	17

1. Use case diagrams

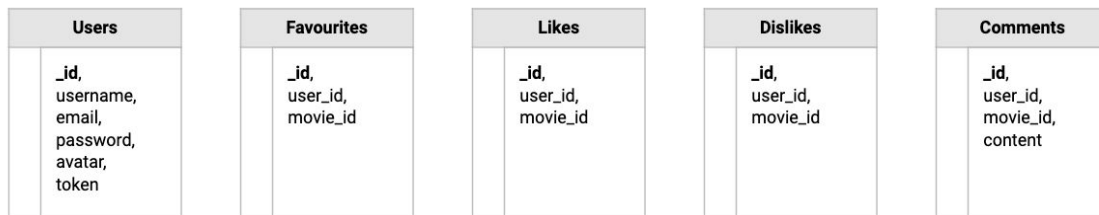
The use case diagram should represent in a graphic, simplified manner the interaction possibilities of the different participants using a given platform (like a webpage) or program.

In this project, we could identify the following:



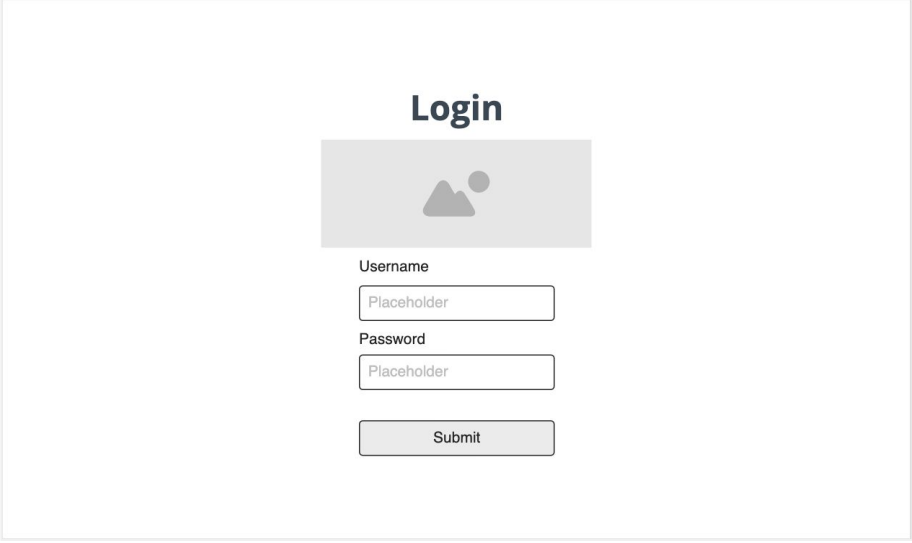
2. Database diagram

The following database design was necessary for the project, and it consisted basically in the five collections and their relations as per the screenshot below:



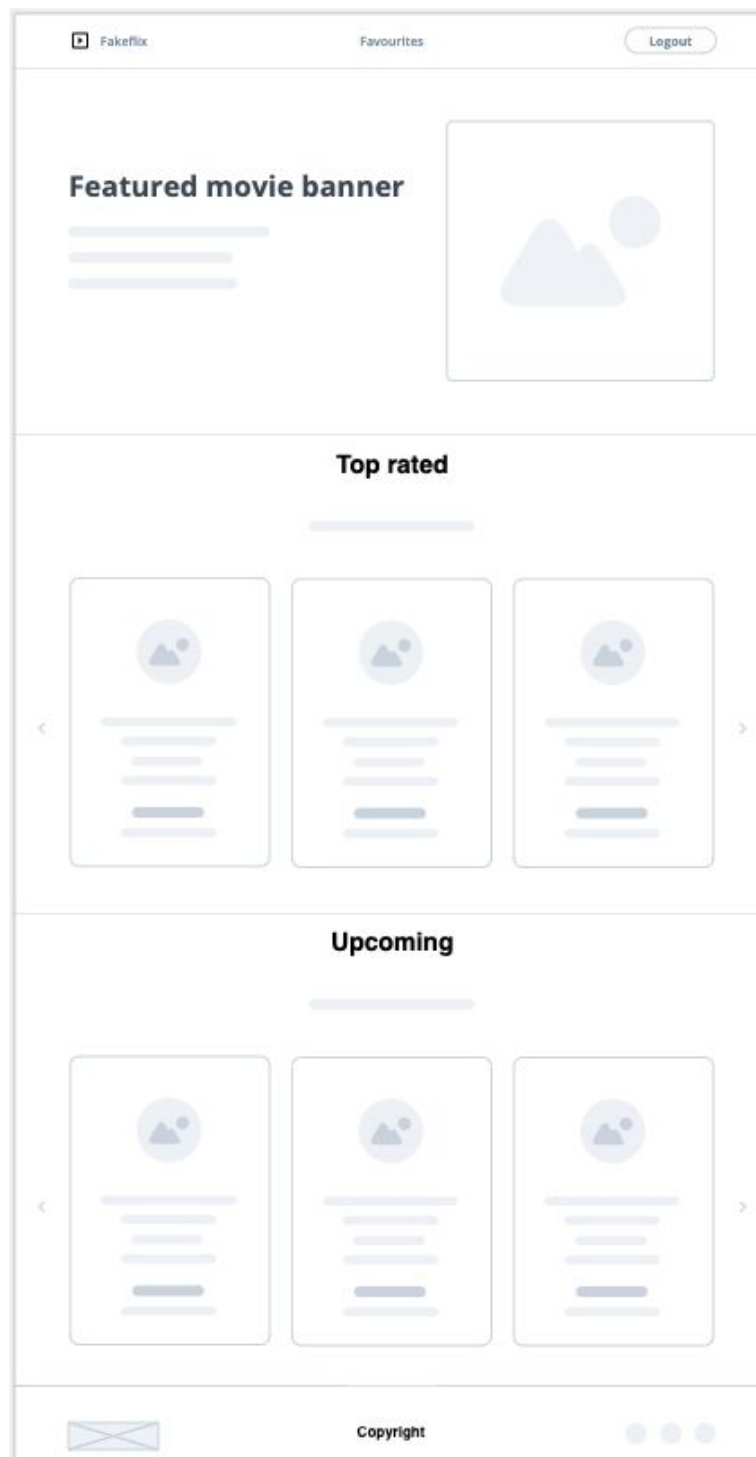
3. Basic wireframe design

3.1. Login

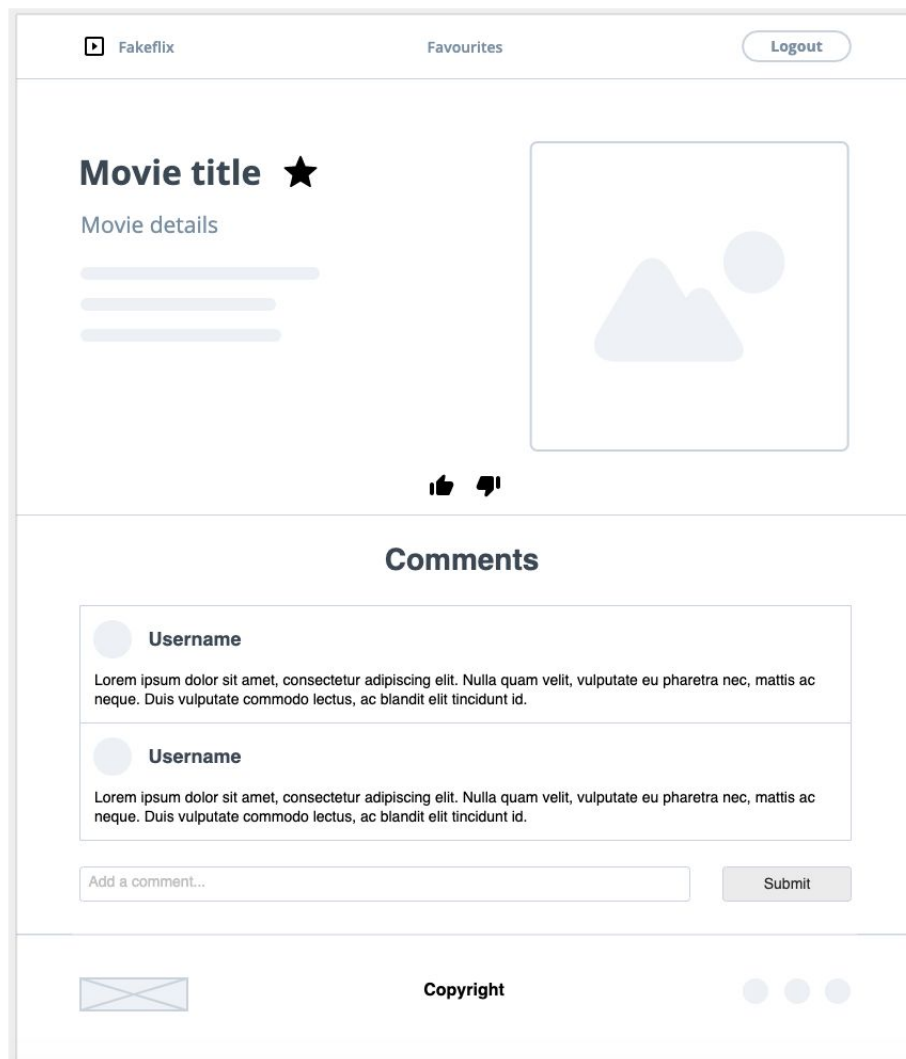


A wireframe of a login form. At the top, the word "Login" is centered in a bold font. Below it is a gray rectangular box containing a simple icon of a mountain and a circle. Underneath this box, the label "Username" is followed by a text input field with the placeholder text "Placeholder". Below that, the label "Password" is followed by another text input field with the placeholder text "Placeholder". At the bottom of the form is a gray rectangular button with the text "Submit" centered on it.

3.2. Homepage

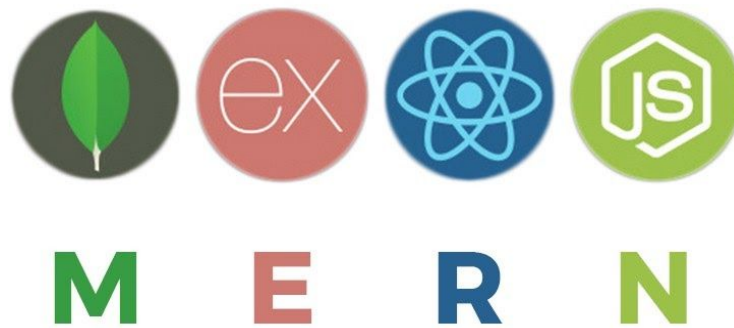


3.3. Movie details



4. Technologies used

The idea of the project was to develop a relatively simple solution, but which included a number of functionalities wide enough to get to know the MERN stack, mainly the frontend library React.



The reason for this choice was that, given the little time we could dedicate to it during the last weeks of the program, it could be an opportunity to spend the extra time we had for the final project to consolidate the knowledge we just acquired.

4.1. Front-end

- React
- Redux
- Bootstrap
- Fetch API
- SCSS/SASS
- Jest

4.2. Back-end

- node.js
- Express
- JSON Web Tokens
- MongoDB
- Mongoose
- External API (<https://www.themoviedb.org/documentation/api>)
- Axios
- Jest

5. Time estimation

Task	Est. time (days)	Calendar Week	Side
Prepare presentation with project proposal	1	0	-
Define database structure in MongoDB	1	1	Backend
Create main files 'server.js' and 'app.js' with server backbone : database connection, middleware, routers and static assets folder	2	1	Backend
Create collection models for User, Like, Dislike, Favourite and Comment with mongoose	1	1	Backend
Create file 'app-config.js' with environment constants	1	1	Backend
Create files with database connection and migration	1	2	Backend
Define user-related routes, controller methods, middlewares and "demo" http requests	2	2	Backend
Define movie-related routes, controller methods, middlewares and "demo" http requests (including likes, dislikes, favourites and comments)	2	2	Backend
Create a test suite for controllers (tested: comments, likes, favourites, movies).	2	3	Backend
Create a test suite for routers (tested: auth, movie, user).	1	3	Backend
Further research on React functional components and Redux	1	3	Frontend
Create React project , folder clean-up and	1	3	Frontend

creation of front-end routes			
Design components and pages related to login (registration, login, password-forgot and password-reset)	2	4	Frontend
Define Redux store, root-reducer and user-related types, reducer, actions and containers	3	4	Frontend
Design components and pages related to homepage , then create movie-related types, reducer, actions and containers (including: navbar, random-movie banner, favourite, top-rated, now-playing and upcoming sections)	5	5	Frontend
Design components and pages related to details page , then create movie-related types, reducer, actions and containers (including: movie details, add/remove favourite, like/dislike, media and comments)	5	6	Frontend
Create a basic test suite for separate components (to at least get a first contact with frontend testing)	5	7	Frontend
Project documentation and deployment on Heroku	5	8	Both

6. Product Analysis

6.1. SWOT

Strengths	Weaknesses
<ul style="list-style-type: none"> • Attractive visual style • Self-updated content through third-party API 	<ul style="list-style-type: none"> • Too much dependant on external content (third-party API)
Opportunities	Threats
<ul style="list-style-type: none"> • Scalable solution either towards social network or to movie-related content database, or both 	<ul style="list-style-type: none"> • Strong and well-established competition (e.g. TMDb, IMDb)

6.2. Competition

We could define here two types of competition, one based on visual styling only (Netflix) and another based on functionality (like any movie-related database on the Internet, such as TMDb or IMDb).

Netflix

This giant of the entertainment industry is well known by all of us and has nothing to be discussed about. It is the king of the Internet-based streaming services and has a solid, well-looking website and applications available on all mobile platforms, not to mention that it produces its own content and counts on its own database used for private purposes. The visuals of this project are based on it.

The Movie Database and Internet Movie Database

Both of them are websites which focus on their databases to offer information relative to movies, TV shows, actors and everything involving the entertainment industry. They have the option to rate movies but they don't really implement a "social" section where posting comments or giving likes and dislikes like in any social network. However, they do give us the possibility to add movies to a favourites section or a follow-up list, which is rarely used.

6.3. Cost Estimation

A simple but effective way to estimate the cost of the project could be the total time spent in development at a rate based on the experience of the developer or the team dedicated to its construction.

In this case, it took a time of around 5-6 weeks for a single and inexperienced Junior developer to build a basic prototype of a usable product. Of course, the final application could have had a larger set of functionalities and several improvements if handled to a more experienced team, but as a starting point we could evaluate the initial cost of ownership to something like this:

Main task	Time (hours)	Rate (EUR/hour)	Subtotal (EUR)
Development	$40 \times 4 = 160$	20	3,200
Testing	$40 \times 1.5 = 60$	20	1,200
Deployment	$40 \times 0.5 = 20$	10	200
Total (w/o taxes)			4,600 EUR

Additionally, we should sum up to that amount the costs derived from hosting the application on a server and registering a domain for the web, which would vary depending on the maintenance and other services contracted and the name chosen for the domain, but would be at least around 25€ to 50€ per month.

7. Changes and unforeseen events

- The features finally included in the project correspond quite accurately to the ones considered in the initial proposal, despite the technical difficulties found or the unexpected events that happened during the development phase.
- A number of “nice-to-have” features that were planned while working on the backend side of the project (on top of the ones presented in the initial proposal) could not be brought into the final product (e.g. “Profile” section) after reducing significantly the available time to work on the project when I started on my new job.
- During the development period, I had to prepare up to three technical tests as a requirement to continue with the selection process for several job applications. They caused me to step out of the project and lose some focus, provided that the technologies applied on them were not related to the ones used in the project.

8. Daily log

Calendar Week	Day #	Log
1	1	<ul style="list-style-type: none"> • The project should focus on learning those technologies on which we could dedicate less time during the last weeks of the program → React, Redux. • The backend is more under control and code from previous projects could be reused.
1	2	<ul style="list-style-type: none"> • Analysis of the project requirements and first questions for the weekly tutoring session → Linter & dependency injection
1	3	<ul style="list-style-type: none"> • Definition of API and key endpoints to consider for the project → themoviedb.org
1	4	
1	5	
2	1	<ul style="list-style-type: none"> • Due to the necessary preparation time for several interviews along the week, the scheduled time to spend in the project for this week is considerably lower than usual.
2	2	<ul style="list-style-type: none"> • Backbone of the backend side of the project is ready, although punctual adjustments might be needed later on to adjust to frontend needs.
2	3	
2	4	<ul style="list-style-type: none"> • Minimum test suites for the backend side completed using Jest, mainly related to database, routers and controllers.
2	5	<ul style="list-style-type: none"> • Frontend development can start, although additional research time on React + Redux is required. • Support sessions organized by the school in order to provide and comment on a project example coded using the MERN stack.
3	1	<ul style="list-style-type: none"> • Technical tests are required as a next step from different application processes. Time spent in project development this week is highly affected.
3	2	

3	3	
3	4	
3	5	<ul style="list-style-type: none"> After the preparation of the technical tests, only a few login views and components are complete. Delay of a few days in the development of the project.
4	1	<ul style="list-style-type: none"> General state for the application is defined. Redux is implemented for the login views and the homepage is the next step.
4	2	
4	3	
4	4	<ul style="list-style-type: none"> Design and state handling is completed for the homepage view.
4	5	
5	1	<ul style="list-style-type: none"> First day in the new job position. From this point onwards, the available time to dedicate to the project is less than half compared to regular workload until date.
5	2	
5	3	
5	4	<ul style="list-style-type: none"> Design and state handling is completed for the movie details view.
5	5	
6	1	<ul style="list-style-type: none"> Frontend testing phase started. Although the testing tool is the same (Jest), it turns out to be way more difficult than testing the backend due to unknown terminology and functionality.
6	2	
6	3	
6	4	
6	5	

7	1	<ul style="list-style-type: none">• This week is dedicated to deployment on Heroku, but again due to working hours and Christmas holiday season there is little time left to use in the project.
7	2	
7	3	
7	4	
7	5	
8	1	<ul style="list-style-type: none">• Project documentation, including this file and a short presentation as a wrap-up/conclusion also for the program.
8	2	
8	3	
8	4	
8	5	

9. Lessons learned

- React and Redux: I started the project with a very limited and theoretical knowledge of what these libraries were, and now I feel comfortable enough to face a code interview or a short project at work. The example repository provided by the academic team was key.
- MongoDB and Mongoose ODM: great alternative to classic relational databases. Really flexible and easy to use, accomplishing the same goals and with the same functionality as MySQL+Sequelize, for instance.
- Testing with Jest: easier for the backend than for the frontend, in my opinion. It is also a really easy-to-use tool with a lot of potential, which almost became a standard in the industry. I believe a basic knowledge is mandatory to start in any position related to web development and I felt I had to practise with it during the project.
- Deployment with Heroku: this free, cloud-based platform is widely used among the developer community mainly because of its simplicity and the high-quality documentation that can be found on the Internet. I found it an almost painless, first deployment experience that every junior developer should learn.
- General: after completing the project and having worked in its different steps on both sides (frontend and backend), I would recommend not to implement functionalities which are not certain to be used at a later stage (e.g. endpoints in the backend which were finally not consumed in the frontend due to lack of time). They suppose an extra effort which is going to be wasted that could have been used in some other key functionality of the project, more testing, better user interface...Stick to the plan and check the project status vs. the initial objectives frequently.

10. Conclusions

Thanks to this project, we learned to deal with the creation of a new product from scratch and were able to apply all the knowledge gathered in our stay at Assembler over the past months.

Even though there is always room for improvement and one might think that the project could have had this or that additional functionality, I am very satisfied with what I achieved in this time, especially considering that we had to combine working on it with other professional obligations (such as job interviews or technical tests). I can even say that most of the time I enjoyed its development.

Personally, when I look back and I see myself at the beginning of the program, I realize all the progress I have experimented -from a limited theoretical understanding of what was the web with almost zero experience in web programming, to being able to develop a fully functional project by myself based on some of the latest technologies- I feel grateful for having found Assembler and having been a part of it.

I want to end up by thanking the team -mainly academics but also management and careers hub- not only for their support at all times during this journey but also for the high quality of the program, the variety and the interest of the contents provided, and their flexibility and commitment during the difficult times of the pandemic outbreak.