Web Technologies Project Report

Group 4

28/12/2023

1 Introduction

Our group has developed a movie library that acts as an independent service in which users can get an overview of a broad range of existing movies. The movie library enables users to create a watchlist and keep an overview of the added movies. Further, the web service features a rating system allowing users to rate movies with 1-5 stars. Additionally, a public comment section is available. Thus, allowing users to make comments, which can be read by others.
The movie library differentiates it's watchlist system from existing streaming services by providing access to a wider collection of movies. Thereby, creating an online environment in which users can create a collected overview.
characters with spaces: 688 words: 112
—— guide: ——
The goal of the introduction is to let the readers (the professor and TAs) know the topic of your work and the main takeaways of it. The introduction should be broad enough to understand the document without reading it and specific enough to let the reader know: *If you are interested in this topic, you should read this work.*
In the context of the Web Technologies course, the introduction should clearly describe:

- Motivation: what problem does this work try to solve (and why is it important)
- Project: clearly describes the topic the group chose to work with
- Contributions: main takeaways that readers will get from reading this work

Remember to keep this and all other sections within the page and column limits. It is your responsibility to describe first the most important and interesting aspect of each section. That way, you can leave behind non-interesting and repeated information more easily.

Length. Half a column.

2 Frontend

In our Laravel application, we have used HTML, CSS, and JavaScript to build the front-end side of the application. Instead of directly writing HTML, we have utilized Blade templates, which is the built-in templating engine in Laravel. Blade allows us to incorporate PHP code into our HTML, which is particularly useful for displaying data from APIs, databases, or other sources.

HTML 5:
HTML 5 is the latest version of HTML, introducing additional semantic tags that enable us to create content with proper structure and meaning. By using semantic tags, we ensure that our web pages are accessible, readable for search engines, and compatible with screen readers. Moreover, semantic tags make it easier for developers to understand the page structure and the purpose of each section. For instance, we have utilized tags like <header>, <nav>, <main>, and <footer> to define the header, navigation bar, and footer of our pages.

In our project, we have, to the best of our ability, tried to make use of these semantic tags to define the different sections of the page. For example, on the page for a specific movie, we have separate sections for movie information, comments, and casts. This division into sections gives a clarity and makes it easier to understand the purpose of each section. Using the <section> tag instead of generic <div> tags provides more descriptive and meaningful markup.

CSS:
CSS is used to define the style and visuals of the HTML elements on the page, defining aspects such as layout, colors, fonts, and overall page structure. Its purpose is to enhance the visuals and can also, if used correctly, enhance user-friendliness of the page with animations, transitions and things like hover effects, which can be used to indicate that an element is clickable and interactive.

In our application we didn't want the default styles, so we used CSS to style the page and application to what we wanted. However, CSS can become challenging to navigate and extend. This is because you never know if a style is used somewhere else in the application. This can lead to unintentional changes in other parts of the application, which can be hard to debug.

To avoid this, especially since this was a group project, which can make the problem even more apparent, we created a separate CSS file for each page unless it was a component that was used on multiple pages. In order to combat this problem, we also tried to use descriptive class names together with only targeting IDs and classes that were specific to the page. This way, we could avoid unintentional changes in other parts of the application. There is a downside to doing this, however, as it can lead to a lot of duplicate code, which can also make it hard to maintain, as it is likely that at some point, there will be unused styles in those CSS files.

JavaScript:
JavaScript is a programming language that is used to make web pages interactive and dynamic like adding and removing elements from the page, changing the style of elements, and making API calls. Since Laravel is a server-side application, most of our data is fetched from the server when the page is loaded. But we also have some dynamic elements on the page, such as the comment section on a movie, which is fetched from the server when the user clicks on the comment button by making an API call to the server, the watchlist button to correctly display the current state of the movie's existence in the user's watchlist, and the rating system, which is used to rate a movie from 1 to 5 stars.

The way that you incorporate JavaScript into HTML is by using the <script> tag. This tag can be used to include JavaScript files, but it can also be used to write JavaScript directly in the HTML file. We have use the first approach, as it is easier to maintain and debug, as you can easily see which JavaScript files are included in the page. However, we have also used the latter but only for small scripts that are only used on that page.

The first task of this project is to build the front-end side of the application. In this section, we will provide a technical description of the implementation details related to HTML 5, CSS, and JavaScript.

- HTML 5: We will explain which HTML tags we have used, where we have used them, and the reasons behind our choices.

- CSS: We will discuss why and how we have utilized CSS, including interesting selectors and declarations, and how it is integrated into the application.

- JavaScript: We will explain why and how we have used JavaScript in our application, including any noteworthy behaviors and how they are incorporated into the application.

Resources. Lectures 1 to 3.

Length. 2 columns.

3 Recourse Management

The second task of this project is to build a resource management. A resource is a model of an object in your system, and it could be anything: movies, music albums, pets, etc. This section should clearly describe the management of the chosen resource: the CRUD (Create-Read-Update-Delete) operations associated with that resource:

- Technically describe the resource (ie, the model)
- Technically describe how the CRUD operations are implemented

Resources. Lectures 4 to 6.

Length. 2 columns.

4 Authentication and Authorization

To help fortify the systems overall security, it is very important that the system knows which users has acces to surtain thing and users that doesnt have access this is done by implementing authorization and authentication on our system.

Authentication: is the process of verifying the identity of a user. In our system we have two distinct user categories 'Guest' and 'User'. The 'Guest' user is the default user and can only access the home page and the login page. The 'User' user is the authenticated user and can access all the pages in the system.
the implementation of the authentication is done by verifying the user's identity, the system asks for a username and password when you login. The system then checks if the username and password match the ones stored in the database. If they match, the user is authenticated and can access the system. if the username and password does not match anything in the systems database, the user is not authenticated and can only acces a limited set of resources on the page.

Authorization: now that the user is authenticated, the system needs to check if the user is authorized to access the resource he is trying to access. In our system we have two distinct user categories 'Guest' and 'User'. The 'Guest' user is the default user and can only access the home page and the login page. The 'User' user is the authenticated user and can access all the pages in the system. the implementation of the authorization is done by checking the user's role, the system checks if the user is a 'Guest' or a 'User'. This is done within the system to ensure that the user is authorized to access the resource he is trying to access. if the guest is trying to access something that he should not be able to access, the system will redirect him to the login page. this will help fortify the security of the system and ensure that only the right users can access the right resources and as a result, the system will be more secure.

To give a better indication of the different user roles we have in our system, we have:

- **Guest:** The default user, can only access the home page and the login page.
- **User:** The authenticated user, can access all the pages in the system.

By giving our website diffrent user roles, we can ensure that the system, is safe against incomming attacks. it alos makes it so, that people can't acces infomation in the system that they should not be able to see without having created a user.
it also works the other way around, only people that are created on the system, can comment on the different movies on the website, while the unauthorized users can only see the diffrent comments. If we where to develp the system further we could add some more user roles, with differnt authorizations, we could make an admin users, this user has access to everything including the profiles of the different users. With the way we have implemented the authorization and authentication in our system, we can easily scale the diffrent users roles, without any complications.

5 Conclusion

The goal of the conclusion is similar to the introduction: it summarizes the work itself and the takeaways a reader should take when reading this work. However, it can use the information presented in the work to be more specific than the introduction.
In the context of the Web Technologies course, the conclusion should clearly describe:

- Summary: summary of the work and main takeaways. Also include a class diagram of the system (the models).
- Future Work: interesting directions on how the presented work can evolve in the future (it may be the starting point to choose individual extension topics)

Length. Half a column.