## Front End Engineering-II

Project Report

Semester-III (Batch-2023)

GloryBoxd

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

Dr. Shivam Singh Namir, 2310992077 (G-23)

**Department of Computer Science and Engineering**

## Chitkara University Institute of Engineering & Technology,

## Chitkara University, Punjab

**Abstract**

This project report describes how the "GloryBoxd" came into being as a web application inspired by “letterboxd.com” a film-focused platform. GloryBoxd was designed using HTML, CSS, JavaScript, and React.js to be a web application that provides for an immersive digital experience tailored specifically for cinephiles. The project has the bedrock of static HTML and CSS in order to provide the fundamental structure and styling. The interactivity and dynamic elements derive from JavaScript and React.js. This project ultimately aims at creating a visually attractive, user-friendly platform through which film lovers can find each other to review and share their cinematic journeys.

The developer aimed to develop a platform "for the cinephiles, by the cinephiles" to address needs and interests of the film-loving community. GloryBoxd reproduces the magnetic and exciting feel of a cinema space because it integrates film culture-inspired design elements and typography to create an authentic atmosphere for users. It was also conceptualized to evoke interactive behavior among users through the website's structure and features while using the features on movie reviews, ratings, and discussion. Each phase, from layout design to user interface, has been optimized for accessibility as well as intuitive navigation so users can smoothly sail through the site and interact with the platform.

This project report traces the development lifecycle of GloryBoxd, from the initial design methodology to final implementation and testing. Some of the major points involved in the process include responsive design, component modularity, and performance optimization within the React-based environment. Results: The integration of modern web development technologies with a user-centered design approach showcases the potential of GloryBoxd as a digital haven for cinephiles who would like to celebrate, critique, and explore art in cinema together.

**Table of Contents**

1. **Introduction**
   * 1.1 Background
   * 1.2 Objectives
   * 1.3 Significance
2. **Problem Definition and Requirements**
   * 2.1 Problem Statement
   * 2.2 Software Requirements
   * 2.3 Hardware Requirements
3. **Proposed Design / Methodology**
   * 3.1 Schematic Diagram
   * 3.2 File Structure
   * 3.3 Algorithms Used
4. **Results**
   * 4.1 Screenshots
   * 4.2 Metrics References

**1. Introduction**

**1.1 Background**

With the growth of digital media and the increasing popularity of online platforms, cinephiles worldwide have turned to dedicated movie review sites to discuss, rate, and catalog their favorite films. Among these platforms, Letterboxd has distinguished itself as a prominent social network for film lovers, allowing users to document their cinematic journeys, share reviews, and connect over a shared passion for movies. Letterboxd’s minimalist and visually engaging design offers a clean, easy-to-navigate interface that keeps users focused on movie content. This combination of simplicity and aesthetic appeal has inspired numerous developers to create similar user experiences. Our project aims to replicate this interface, providing a static and responsive web page to emulate the feel of Letterboxd while offering hands-on practice in fundamental front-end development.

The purpose of this project is not only to recreate the Letterboxd design but also to build a solid foundation in web development by focusing on two core technologies: HTML, CSS, JavaScript and React JS. HTML is used for structuring the web page's content, while CSS is applied to style and format the elements, giving them a refined, appealing layout. Together, these technologies allow us to create a visually engaging and static website that resembles Letterboxd, offering a simplified yet functional version of a movie review platform. The project emphasizes modular design, allowing for a clean separation of structure and style, which is a fundamental concept in web development.

This project serves as an opportunity to practice essential skills and understand the intricacies of web design. By mimicking a well-established platform like Letterboxd, this project challenges us to consider user experience and aesthetic consistency. This replication process not only strengthens proficiency in HTML, CSS, JavaScript and React JS but also provides insights into the careful planning and execution required in professional web development. The project is, therefore, both an educational exercise and a tribute to one of the most beloved platforms in the cinephile community.

**1.2 Objectives**

The primary objectives are as follows:-

* To understand and apply HTML for webpage structure: This objective involves understanding HTML to structure the webpage’s layout effectively. By mastering HTML tags and elements, we can create an organized, user-friendly framework for displaying content.
* To utilize CSS for styling and layout: This objective focuses on using CSS to enhance the visual appeal and layout of the webpage. Through styling techniques like color schemes, typography, and responsive design, CSS allows us to create an engaging and aesthetically cohesive user interface.
* To develop a responsive web design using JavaScript: This objective emphasizes using JavaScript to create a responsive web design that adapts smoothly to different screen sizes and devices. By dynamically adjusting elements and adding interactive features, JavaScript enhances the usability and engagement of the webpage across various platforms.
* To understand and implement ReactJS in the project: This objective focuses on learning and applying ReactJS to build a more dynamic and efficient user interface. By using ReactJS’s component-based structure, the project aims to improve code reusability and streamline the development of interactive, responsive elements.

**1.3 Significance**

The significance of GloryBoxd lies in its ability to provide an immersive, cinephile-focused platform while showcasing the versatility and functionality of modern web technologies. Here are some key points highlighting its importance:

- User-Centric Experience: GloryBoxd aims to create a specialized space for cinephiles by offering an interface tailored to film enthusiasts, enhancing the enjoyment of cataloguing, rating, and discussing films.

- HTML Structure: HTML serves as the backbone of the application, providing a structured layout for the web pages and ensuring semantic organization. This supports accessibility and facilitates easy navigation.

- Aesthetic and Layout with CSS: CSS is leveraged to style and bring visual cohesion to the platform. It plays a crucial role in creating a visually appealing, intuitive, and responsive layout, ensuring compatibility across different devices and screen sizes.

- Dynamic Interactions with JavaScript: JavaScript adds interactivity to GloryBoxd, allowing for user-driven features such as form submissions, animations, and dynamic page elements, making the platform feel more engaging and responsive.

- Component-Based Design with ReactJS: ReactJS is integral in developing a modular, scalable structure. By building reusable components, the project achieves a smooth, efficient, and highly interactive user interface, which also enhances development efficiency and code maintainability.

Overall, GloryBoxd demonstrates the potential of web technologies to craft engaging, tailored user experiences for niche communities.

**2. Problem Definition and Requirements**

**2.1 Problem Statement**

The problem addressed by the "GloryBoxd" project is the need for an immersive, user-friendly film review platform specifically designed for cinephiles, offering a space to discover, rate, and discuss films. Existing platforms, like Letterboxd, often lack a community-driven, cinematic atmosphere that many film enthusiasts desire. The project aims to create a visually appealing and responsive web application using HTML, CSS, JavaScript, and React.js that combines engaging design with interactive features such as movie reviews and ratings. By focusing on user experience and fostering a sense of belonging within the cinephile community, "GloryBoxd" seeks to provide a platform that enhances both personal and shared film exploration.

**2.2 Software Requirements**

For Front-End Development:

* HTML5 - For structuring the content and elements of the web pages.
* CSS3 - For styling and layout design, ensuring the website is visually appealing and responsive.
* Bootstrap - For implementing responsive design elements and components efficiently.
* JavaScript - For adding interactivity and dynamic content to the website.
* ReactJS - For building a modular, scalable, and maintainable front-end architecture.

Tools and Environment used for Development:

* Code editor (Visual Studio Code) - To write and edit code.
* GitHub - For repository hosting and project management.
* Web browser (Chrome) - To check the final result and see the preview of the website.

**2.3 Hardware Requirements**

* A computer with internet access
* Intel i5 or equivalent processor
* 8GB (Minimum)
* 256GB SSD (Minimum)
* Full HD resolution (1920x1080) (Minimum)

**3. Proposed Design / Methodology**

**3.1 Schematic Diagram**

The website will consist of several primary sections/pages:

* Landing Page
* Home Page
* Movie Details Page
* User Profile Page
* Review & Rating Section
* Search and Filter Functionality
* Navigation Bar
* Footer

**3.2 Algorithms Used**

1. HTML Structure Algorithm

* Purpose: HTML (Hypertext Markup Language) is used to structure the content on the webpage.
* Method: HTML elements are organized in a hierarchical manner to create the layout and structure of the website. These elements include headings, paragraphs, divs, images, forms, and links, each serving a specific role in the webpage’s structure.
* Implementation:
  + Use semantic tags such as <header>, <main>, <footer>, <section>, and <article> to define the main sections of the webpage.
  + Input forms for reviews and movie search functionality are included using <input>, <textarea>, and <button> tags.
  + Proper nesting and organization of elements ensure accessibility and correct rendering across different devices and browsers.
* Efficiency: HTML is not typically evaluated in terms of computational efficiency, but semantic HTML ensures better performance in terms of accessibility, SEO, and cross-browser compatibility.

2. CSS Styling Algorithm

* Purpose: CSS (Cascading Style Sheets) is used to visually style the webpage, ensuring that it looks attractive and provides a good user experience.
* Method: CSS rules are applied to HTML elements based on their class, id, or type selectors. Flexbox or CSS Grid is used for responsive layouts, ensuring the design adapts to different screen sizes (desktop, tablet, mobile).
* Implementation:
  + Responsive Design: Media queries are used to adjust layout, font size, and other properties based on screen size (@media screen and (max-width: 768px) for mobile responsiveness).
  + Flexbox/Grid: Used for creating flexible and grid-based layouts. For example, movie thumbnails might be displayed in a responsive grid.
  + Animations: CSS animations (e.g., fading, sliding) can be used for elements like buttons or hover effects to improve interactivity.
* Efficiency: CSS is inherently optimized for styling, and techniques like Flexbox and Grid make layout adjustments efficient. Using variables and reusable classes ensures minimal code repetition.

3. JavaScript Dynamic Interaction Algorithm

* Purpose: JavaScript provides interactivity by enabling the manipulation of HTML and CSS dynamically, such as handling user input, fetching data, and updating the UI without reloading the page.
* Method: JavaScript is used to handle events like button clicks, form submissions, and page navigation. It allows dynamic updates of content (e.g., filtering movies based on user input, submitting reviews).
* Implementation:
  + Event Listeners: JavaScript listens for events like click, submit, and input to trigger functions (e.g., submitting a review, filtering movies).
  + DOM Manipulation: JavaScript updates the DOM (Document Object Model) dynamically. For example, when the user types into the search bar, the list of movie thumbnails can be updated instantly.
  + AJAX/Fetch API: JavaScript can fetch movie data or user reviews from an API or server asynchronously, providing a smooth experience without page reloads.
* Efficiency: JavaScript operates in real-time, ensuring that actions such as clicking, typing, or submitting are processed instantly, enhancing user experience. For large-scale data manipulation, using frameworks like React optimizes the efficiency of DOM updates.

4. ReactJS Component-Based Architecture

* Purpose: ReactJS enables the creation of reusable components, helping organize and manage complex UIs with dynamic content.
* Method: React breaks down the UI into individual components that handle their own state and rendering logic. Components are combined to build the full page (e.g., Navbar, MovieDetail, ReviewSection).
* Implementation:
  + State Management: React’s useState and useEffect hooks allow components to manage dynamic data. For example, the movie list in the Home page is stored in state and updated when the user interacts with search filters.
  + Component Rendering: React efficiently updates the DOM based on changes in component state, ensuring that only the necessary parts of the page are re-rendered when data changes.
  + React Router: Used for handling navigation between different pages without refreshing the page. For example, when a user clicks on a movie thumbnail, React Router dynamically loads the movie details page.
* Efficiency: React uses a virtual DOM that minimizes actual DOM manipulation, making the rendering process faster and more efficient. The state-driven architecture allows for smooth updates, even with complex user interactions.

**4. Results**

**4.1 Screenshots**

****

Fig: 4.1

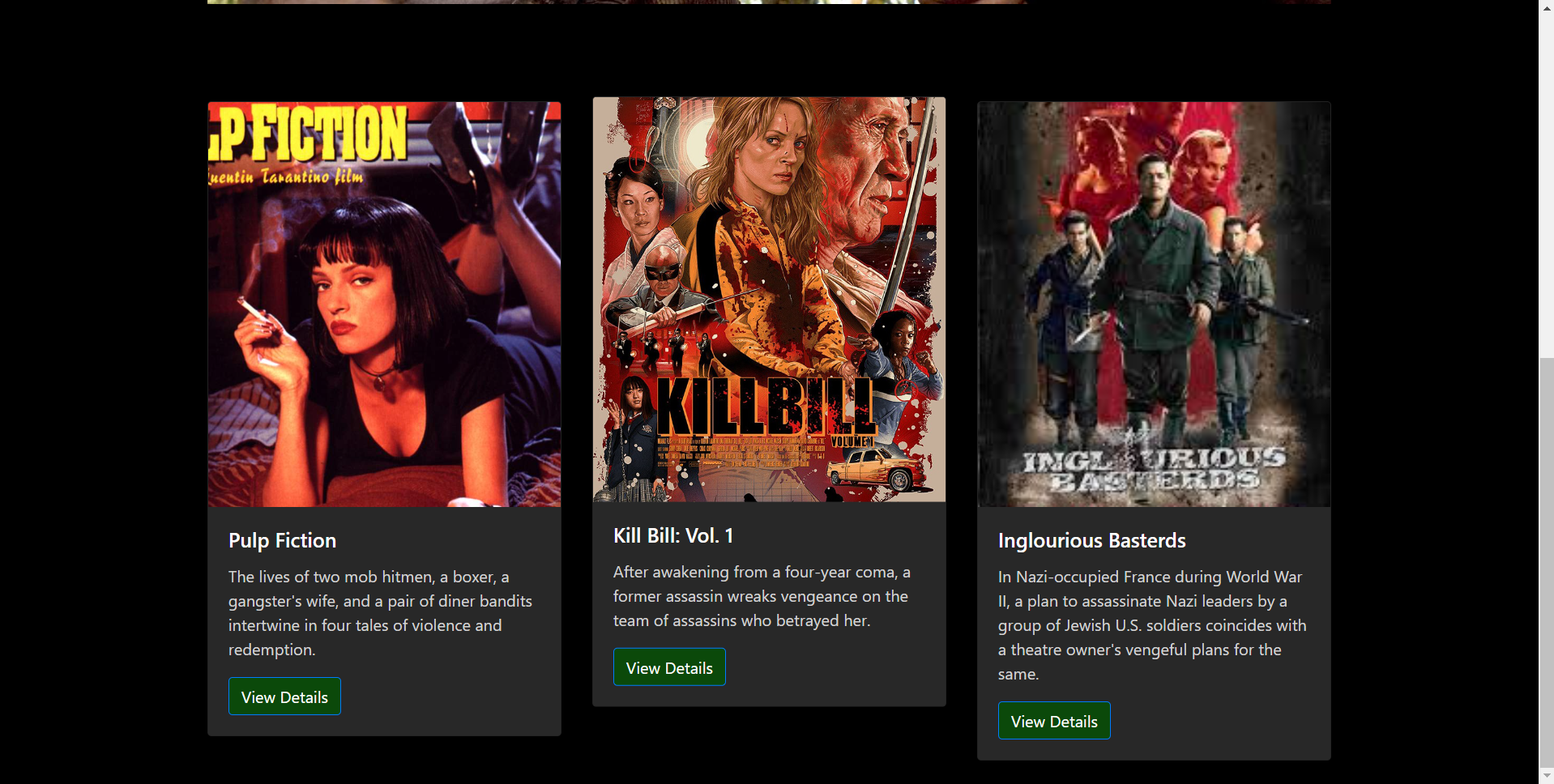
****

Fig: 4.2



Fig: 4.3

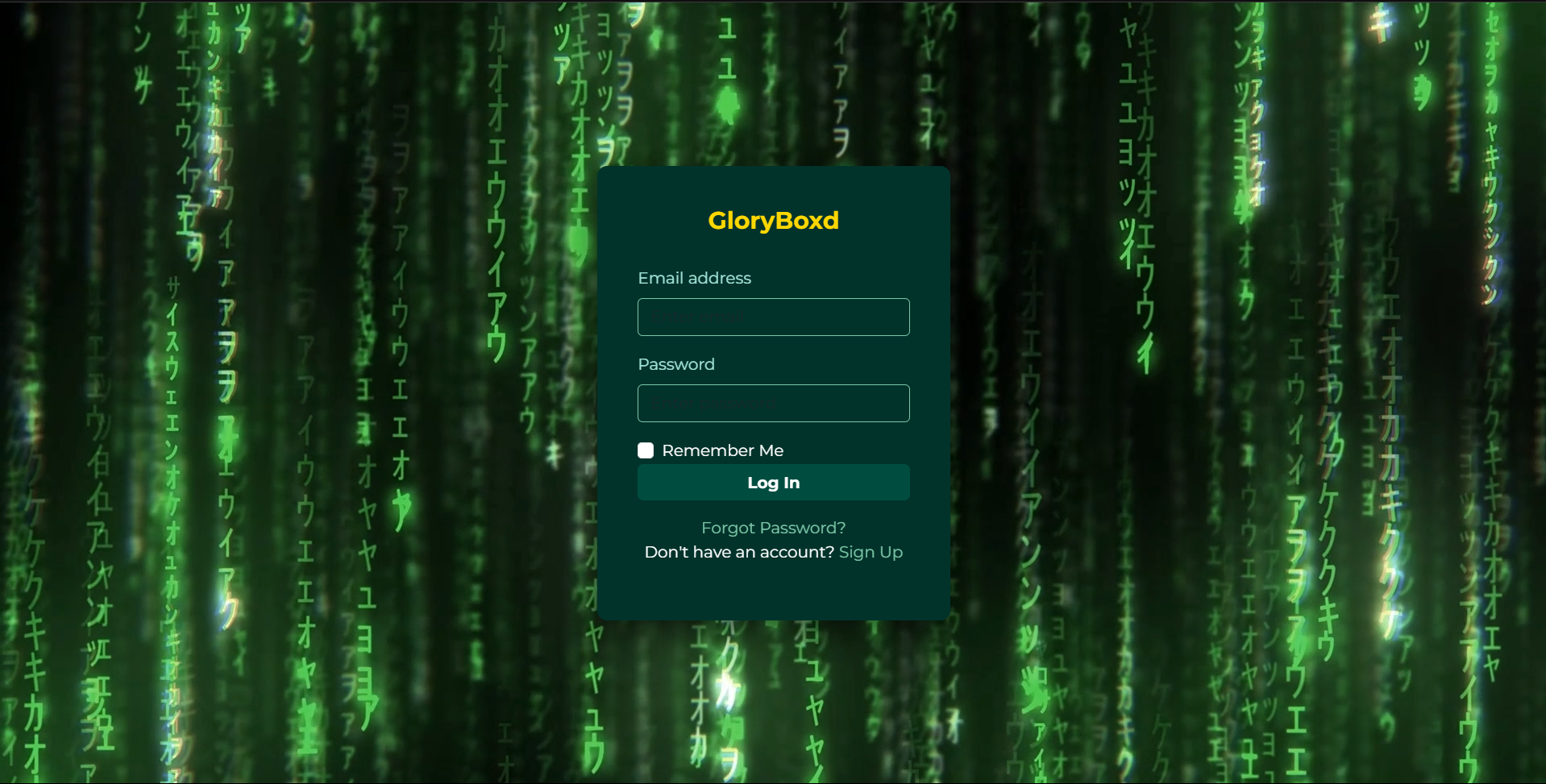


Fig: 4.4



Fig: 4.5

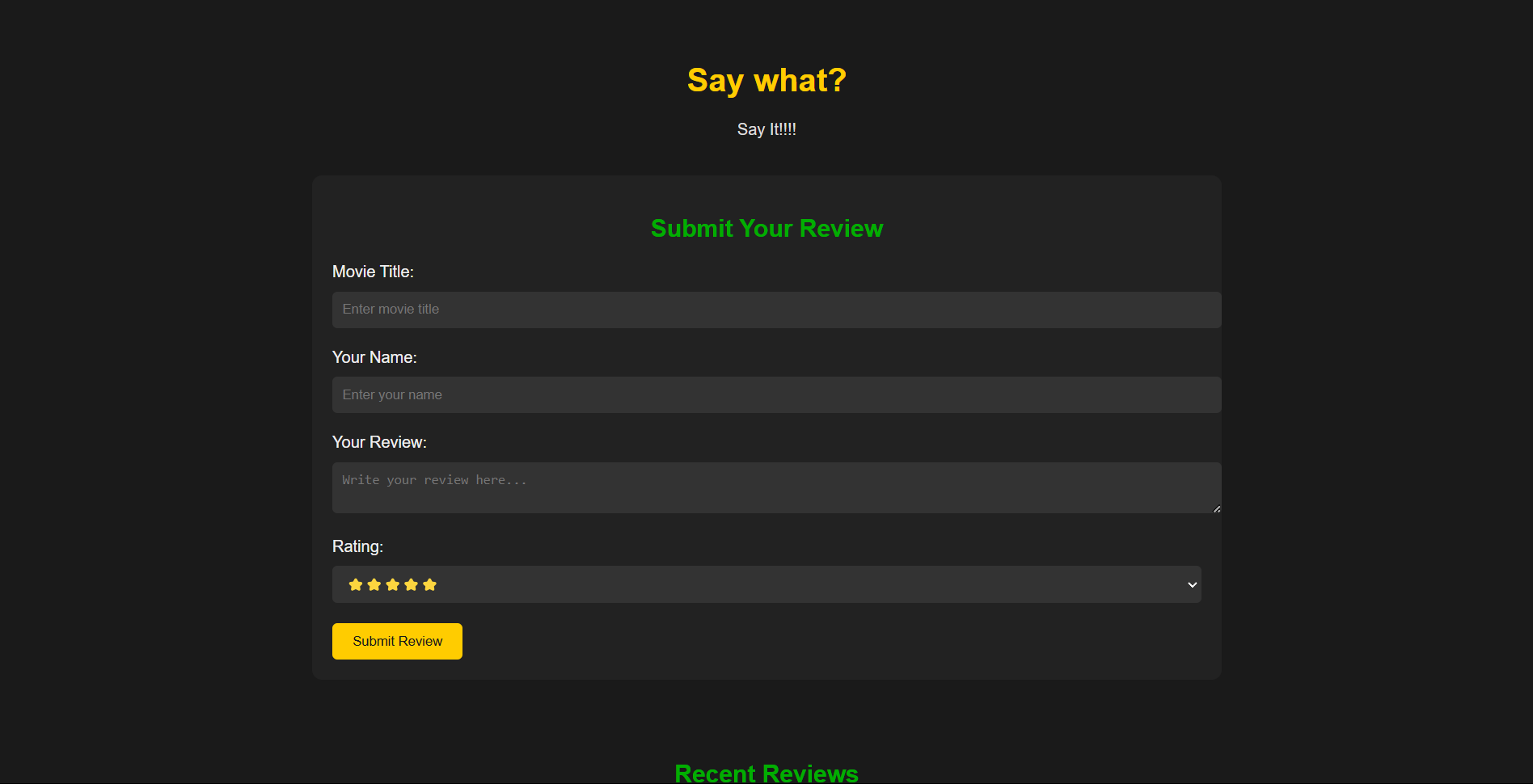


Fig: 4.6

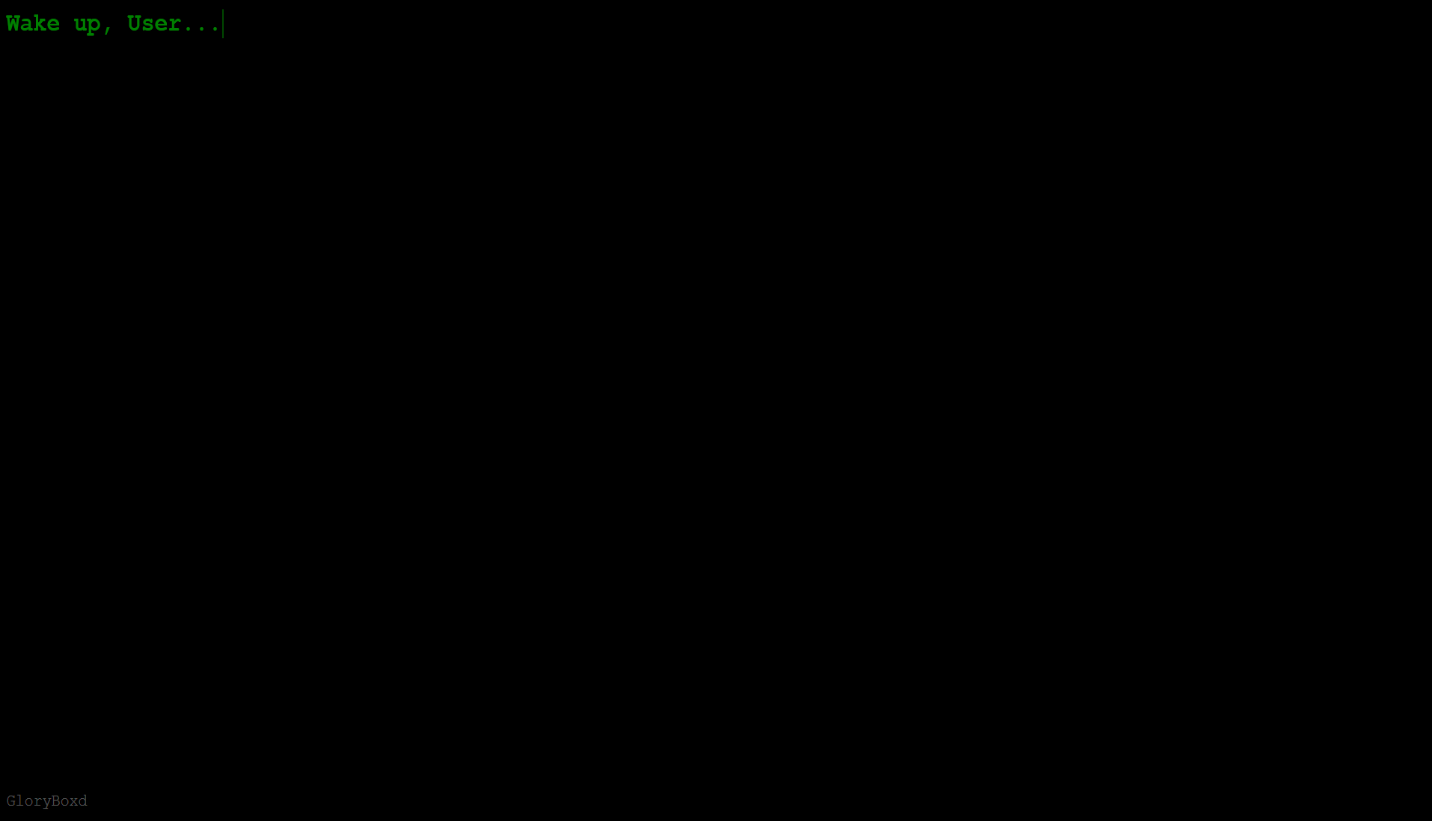


Fig: 4.7

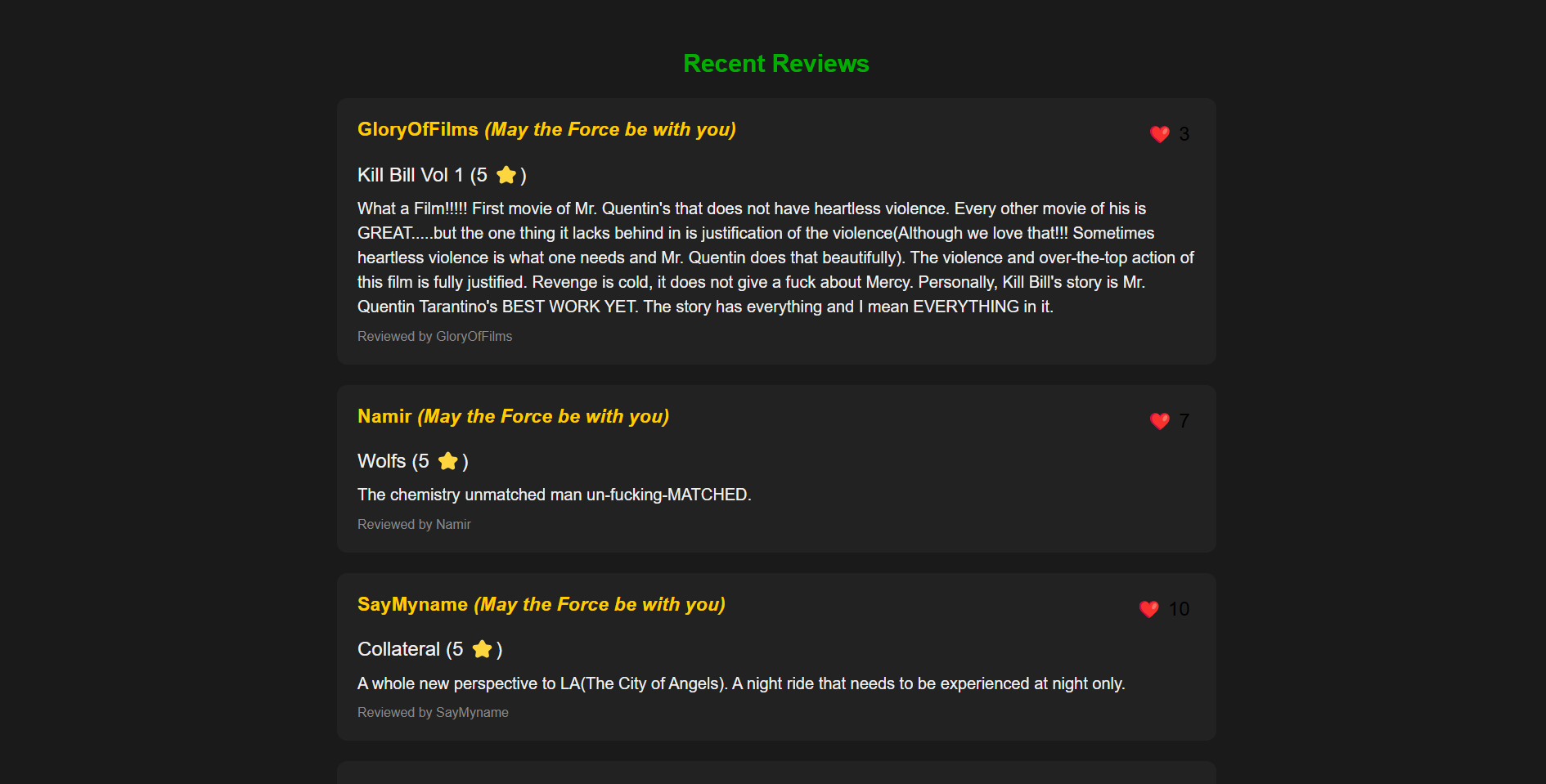


Fig: 4.8

**References**

## **Took inspiration from Letterboxd,**

* IMDB was used for reference,
* As this website is about movies, for movies by a Cinephile a lot pop-culture references have been made.
* The movie “The Matrix” has been referenced by using its signature green colour.
* Signature Font of Quentin Tarantino “Benguait Bold” was used.