VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

PLATFORM FOR ENJOYING MOVIES AND MANAGING YOUR WATCHLISTS

By Sparkle Candles Team
Nguyễn Khánh Hà (Leader) – ITCSIU21004
Tạ Thị Thuỳ Dương – ITCSIU21053
Bùi Văn Minh Triều – ITCSIU21241

A report submitted to the School of Computer Science and Engineering in partial fulfillment of the requirements for Web App Development Subject (IT093IU)

ACKNOWLEGMENTS

We would like to express my sincere gratitude to Associate Professor Dr. Nguyen Van Sinh for teaching the Web Application Development course. His invaluable guidance and expertise have contributed significantly to my learning journey.

We would also like to express my sincere gratitude to Mr. Nguyen Trung Nghia for his dedication and support during the lab sessions. His efforts have greatly improved my practical skills and helped me gain a deeper understanding of the subject.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Contribution Table

Name	Role	Task Done	Contribution
Nguyễn Khánh Hà	Fullstack Developer	Design the website using Figma, Building the back-end side of the platform, creating APIs, and connecting databases using Firebase and MongoDB. Connect the front-end with the back end, fixing responsive across all front-end pages, setting up reusable components to enhance scalability. Implementing business logic for adding and removing movies to/from favorite/continue/watchlists, business logic for comments functionality on the front-end, rendering user information.	100%
Bùi Văn Minh Triều	Front-end Developer	Building UI Components. Connecting the front-end with the back end. Implementing logics for sorting page, search bar, watch page and website. Integrating sign in with google and collapse menu.	85%
Tạ Thị Thuỳ Dương	Front-end Developer	Drawing Figma. Building UI Sign-in, Sign-up, description page. Comment and UI components. Connecting the front-end with the back end and process logic for pagination, movie page, tv series page and genre.	85%

ABSTRACT

The SPARKLE Candles platform is a comprehensive online application designed to enhance the movie viewing experience by integrating streaming, curation, and movie list management features. Using the MEAN stack architecture (MongoDB, Express.js, Angular, and Node.js), the platform supports seamless operations and high scalability. Key features include advanced search capabilities, personalized watchlists, and a powerful user interface optimized for responsiveness and accessibility.

Developed using an Agile methodology with Scrum, the platform adapts to dynamic user needs through iterative sprints, allowing for continuous improvement and incorporating user feedback. With an emphasis on security and scalability, features such as JSON Web Token (JWT) authentication and efficient database optimization ensure robust performance and data integrity.

The project bridges entertainment and technology, bringing cinematic experiences to a global audience. Its innovative approach positions SPARKLE Candles as a dynamic and user-friendly solution for film enthusiasts.

CHAPTER1

INTRODUCTION

1.1.Background

SPARKLE Candles is the ideal destination for movie lovers, offering thousands of attractive movies and TV series to serve the entertainment needs of all ages. With a rich content library, constantly updated with the latest blockbusters, the platform confidently conquers all interests and tastes of the audience.

The interface is exquisitely designed, optimizing the user experience, allowing you to easily search and explore a huge movie library with just a few clicks. From blockbusters to popular series, SPARKLE Candles helps you access a diverse content library by genre, year of release, or country of production. With the main language being English, this platform expands its reach to a global audience, meeting the expectations of millions of international users.

More than just a streaming website, SPARKLE Candles is truly a cinematic paradise, where you can immerse yourself in vivid stories and fully enjoy the art of cinema. The platform's highlight lies in smart features such as storing movie viewing history, creating personal playlists, helping you easily manage and save your favorite movies. With SPARKLE Candles, you don't just watch movies but also enter a world of unlimited emotions and creativity.

1.2.About us

We are SPARKLE Candles – a team of professional, dedicated programmers in the field of web application development. With the mission of bringing outstanding value to customers, we always listen and understand the needs, constantly researching to provide optimal solutions.

We are committed to designing applications that are not only beautiful, but also optimized in performance, ensuring a smooth and friendly experience for users. Our team, with a combination of youth, creativity and high expertise, not only focuses on building useful tools but also aims to improve the quality of life through technology. SPARKLE Candles – where the complete customer experience is our mission.

Team name	SPARKLE Candles
Contact	Khu phố 6, Đ. Võ Trường Toản, Phường Linh Trung, Thủ Đức, Thành phố Hồ Chí Minh
Email	spark.candles@gmail.com
Phone number	(+84)34 556 789

1.3. Development process

The Agile methodology is being adopted for our team, and for guiding project development, the Scrum framework is applied. Scrum is a perfect choice because it possesses the highest degree of flexibility, which allows the team to adapt to changes easily during development. Besides, it emphasizes improvement continuously, which can be enhanced in product efficiency by discussing and adjusting plans after each Sprint. Also, Scrum allows for the possibility of early release of some features of the product to deliver value even in the development phase to the users.

Scrum is organized around iterative cycles-the Sprints-which generally run for 2-4 weeks and are dedicated to the construction of features useful for the users. Sprint Planning will be conducted for prioritization of tasks and to clearly state the goals for implementation. This allows teams to review progress regularly and clear any obstacles in a timely manner. The end of each Sprint provides the Sprint Review, which allows showing finished features to stakeholders, followed by valuable feedback. Lastly, the Sprint Retrospective provides an opportunity for the team to reflect on the sprint that was and where they need further improvement.

Continuous improvement is at the core of our development process. We continuously enhance the user experience through refinement of the user interface and the addition of new or improved features in each Sprint, while also being responsive to feedback for product optimization to meet the ever-changing needs of users.

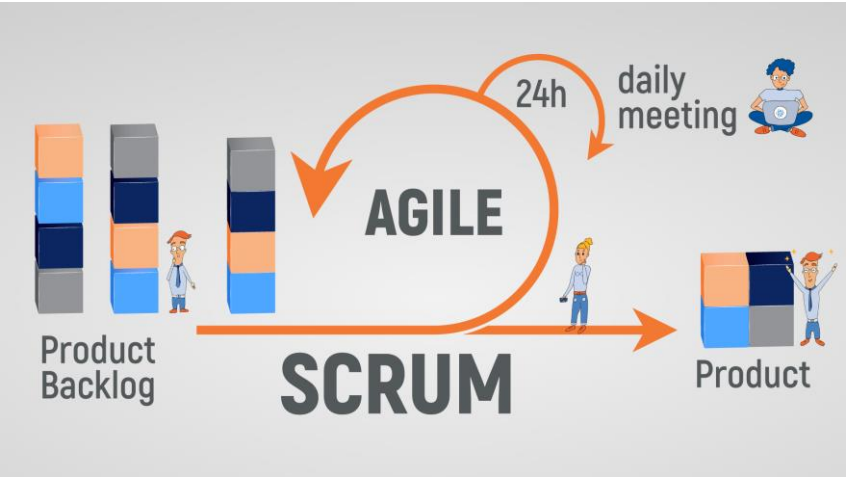

Figure 1: The Scrum Agile methodology used in software development.

1.4. Development environment

Since this is a web-based product, the web design and programming languages used in the project follow the MVC model. The following programming languages are used in our system:

Angular: Angular is a TypeScript-based development platform that enables the building of scalable web applications by providing a component-based framework, integrated libraries for routing, forms, and client-server communication, and a set of tools for development, building, testing, and updating code.

ExpressJS (Node.js): ExpressJS is used in building the server part of a web application. As a back-end platform running on Node.js, it provides support to HTTP request handling, routing, and middleware functions essential in API development.

MongoDB (NoSQL Database): NoSQL database to store website data. MongoDB offers very scalable and flexible data storage, which is appropriate for modern web applications requiring fast data retrieval and the ability to handle big volumes of data.

JWT (JSON Web Tokens): This is for added security to authenticate and authorize users in your web application.

OAuth: This is an added security that grants limited access to user resources by third-party applications without exposing the user credentials.

Firebase: Firebase is a cloud-based platform provided by Google that supports the development of mobile and web applications, including services such as real-time databases, authentication, cloud storage, and hosting. It's versatile, scalable, and streamlines development to improve the performance of an application. In our project, Firebase is used for storing images.

Figma: Figma is a powerful, cloud-based design tool that enables teams to collaborate on web and UI/UX designs in real time. It allows one to edit vector graphics, prototype, and even gives a set of collaboration tools that make it very popular among designers and developers.

The project is implemented using an MVC model:

- Model: the information stored in an external database and system
- View: the user interface.
- Control: the logic and techniques for creating a dynamic website that provides the needed functionality

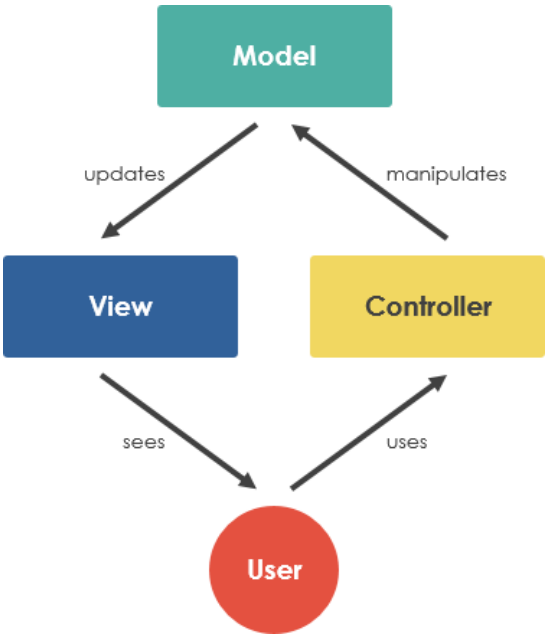

Figure 2: The Model-View-Controller (MVC) architectural pattern used in software development.

1.5.Technology Stack Overview

Furthermore, MEAN Stack Architecture is used for the project for several reasons:

MEAN Stack uses pure JavaScript in every layer: client-side and server-side and database management; this reduces complexity to improve development and maintenance speed, which can switch between a view and model with unprecedented comfort.

Ideal for modern web applications: Angular supports SPA creation, which provides smooth, interactive user experiences. MongoDB's NoSQL structure provides flexible data

storage for scalable projects. Node.js and Express.js maintain high performance for applications that require real-time responses or high concurrency.

Scalability: High scalability is supported by the MEAN Stack with MongoDB, efficiently handling large datasets, making it suitable for projects of any size.

Optimal performance: Node.js event-driven, non-blocking I/O provides very fast request handling. Along with Angular and Express.js, it reduces latency in the communication between the frontend and backend.

Powerful Open-Source Ecosystem: MEAN Stack reduces development costs by availing of a robust ecosystem of libraries and community resources for quick problem-solving.

Powerful Open-Source Ecosystem: MEAN Stack reduces development costs by availing of a robust ecosystem of libraries and community resources for quick problem-solving.

Efficient Application State Management: Angular's component-based architecture and MongoDB's native JSON storage streamline state management and interactions between the frontend and backend.

In a nutshell, MEAN Stack provides synchronized development, scalability, high performance, and seamless user experience, perfectly aligning with the project's goals.

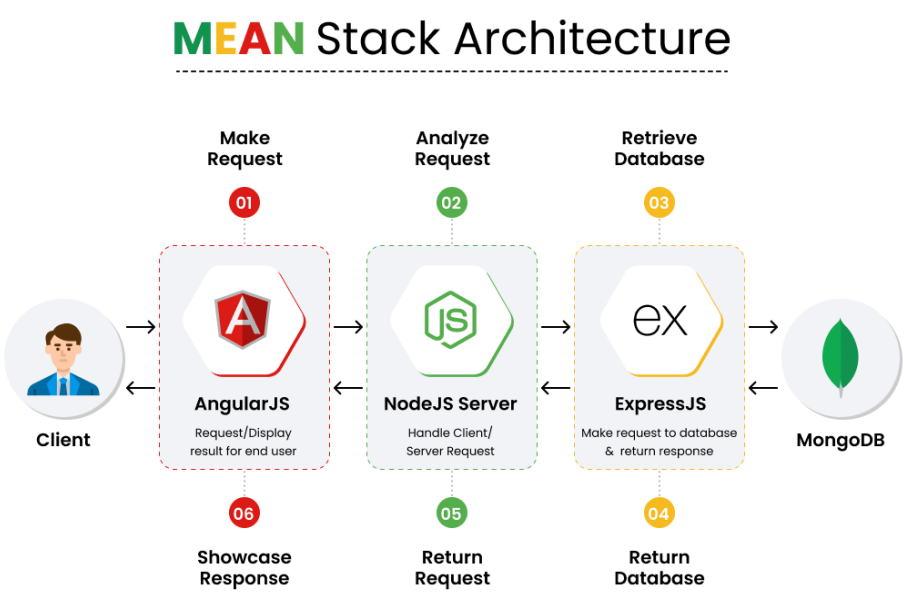

Figure 3: MEAN Stack Architecture is used for the project.

CHAPTER2

REQUIREMENT ANALYSIS AND DESIGN

2.1. REQUIREMENT ANALYSIS

2.1.1. Use case diagram

Actors:

User: The main actor who registers, logs in, searches for movies, watches content, organizes lists (favorites, watchlists, etc.), and adds comments.

Admin: The actor responsible for managing users, editing movie details, and moderating comments to maintain platform quality.

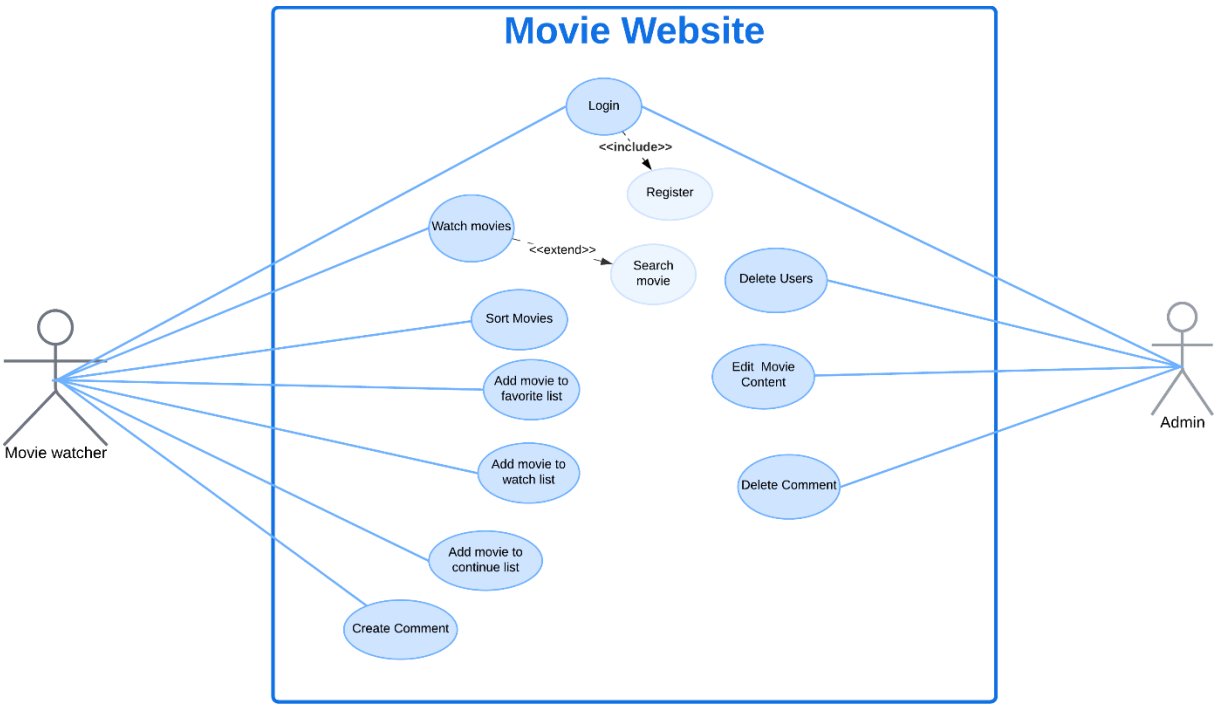

Figure 4: *Use case diagram*

Description:

- **Register:** New users can create an account by providing required details to access features like creating watchlists, favoriting movies, and leaving comments.
- **Log in:** Users and administrators can log in to access various features such as managing movie lists, editing content, and performing administrative tasks.
- **Search movie:** Users can search for movies using movie title.
- **Watch movies:** Users can stream movies directly from the website.
- **Sort movies:** Users can sort movies based on year, genre, and country for easier navigation.
- **Add movie to favorite list:** Users can mark movies as favorites for quick access to their most favorite content.
- **Add movie to watch list:** Users can add movies to watch list for future viewing, keeping track of movies they plan to watch.

- **Add movie to continue list:** Movie is added to "continue watching" list when user click to watch that specific movie, from this, user can keep track on movies that he/she watch before that.
- **Create Comment:** Users can comment on movies, share their thoughts or reviews about the content.
- **Delete Users (Admins):** Admins can manage user accounts by deleting inactive or inappropriate users.
- **Edit Movie Content (Admins):** Admins can edit movie details, including title, description, metadata, or other attributes.
- **Delete Comment (Admins):** Admins can delete inappropriate or irrelevant comments to maintain the quality of the platform and user experience.

2.1.2. User specification

Table 1: Use specification for Register

Use Case No.	**001**
Use Case Name	**Register**
Inputs	User details (e.g., name, email, password)
Outputs	[If success] Account created and confirmation message displayed. [If fail] Error message (e.g., "Email already exists").
Actor	User Open the registration page -> Display registration form. Enter details and submit -> System validates inputs. • [If successful] Create account and display confirmation. • [If fail] Display an error message.
Precondition	User is not logged in.
Postcondition	New account is created, user navigate to Login page

User story	As a new user, I want to register to the website.

Table 2: Use specification for Login

Use Case No.	**002**
Use Case Name	**Login**
Inputs	User credentials (email and password).
Outputs	[If success] User logged in and redirected to the homepage. [If fail] Error message (e.g., "Invalid email or password").
Actor	User/Admin Open the login page -> Display login form. Enter credentials and submit -> System verifies inputs. • [If successful] Grant access and redirect to homepage. • [If fail] Display an error message.
Precondition	User has a registered account.
Postcondition	User is logged into the website and navigate to Homepage
User story	As a returning user, I want to log in to access my account.

Table 3: Use specification for Watch Movies

Use Case No.	**003**
Use Case Name	**Watch Movies**

Inputs	Movie selection.
Outputs	[If success] Start movie playback. [If fail] Display error message (e.g., "Loading movie").
Actor	User Click play episode -> System streams the selected movie episode. • [If successful] Start playback. • [If fail] Display an error message.
Precondition	User is logged in and has a stable internet connection.
Postcondition	Movie playback starts.
User story	As a watcher, I want to stream movies seamlessly for entertainment

Table 4: Use specification for Search Movies

Use Case No.	**004**
Use Case Name	**Search Movies**
Inputs	Search query (title).
Outputs	[If success] Display relevant search results. [If fail] Display "No results found" message.
Actor	User Open search bar -> Enter search query. Waiting query -> System processes and fetches results. • [If successful] Display matching movies. • [If fail] Display a "No results found" message.

Precondition	User is logged in and has a stable internet connection.
Postcondition	Search results are displayed to the user.
User story	As a movie enthusiast, I want to search for specific movies to find what I want to watch.

Table 5: Use specification for Add favorites

Use Case No.	**005**
Use Case Name	**Add Movie to Favorite List**
Inputs	Movie selection.
Outputs	[If success] Movie added to the favorite list. [If fail] Display error message.
Actor	User Navigate to the movie description page -> Click "Heart" icon. The system updates the user's favorite list. • [If successful] Movie is added to favorite list in database and render on UI. • [If fail] Display an error message.
Precondition	User is logged in.
Postcondition	Movie is saved in the favorite list.
User story	As a movie enthusiast, I want save my favorite movies for easier keep track.

Table 6: Use specification for Add watch list

Use Case No.	**006**
Use Case Name	**Add Movie to Watch List**
Inputs	Movie selection.
Outputs	[If success] Movie added to the watch list. [If fail] Display error message.
Actor	User Navigate to the movie description page -> Click "+ My List" button. The system updates the user's watch list. • [If successful] Movie is added to watch list in database and render on UI. • [If fail] Display an error message.
Precondition	User is logged in.
Postcondition	Movie is saved in the watch list.
User story	As a movie enthusiast, I want to save my watch list for easier keep track.

Table 7: Use specification for Add continue list

Use Case No.	**007**
Use Case Name	**Add Movie to Continue List**
Inputs	Movie selection.

Outputs	[If success] Movie added to the continue list. [If fail] Display error message.
Actor	User Start watching a movie -> System automatically tracks progress. Resume watching -> Continue list updates with the movie. • [If successful] Display confirmation. • [If fail] Display an error message.
Precondition	User is logged in.
Postcondition	Movie is saved in the continue list, user navigate to watch page
User story	As a viewer, I want to resume movies from where I left off for a seamless experience.

Table 8: Use specification for Create comment

Use Case No.	**008**
Use Case Name	**Create Comment**
Inputs	User comment text, movie ID.
Outputs	[If success] Comment added and displayed on the watch page of specific movie [If fail] Display error message.
Actor	User Open the watch page -> Enter a comment in the comment box. Submit comment -> System saves the comment. • [If successful] Display the comment. • [If fail] Display an error message.
Precondition	User is logged in.

Postcondition	Comment is added to the movie page.
User story	As a movie fan, I want to share my opinions and reviews on movies to engage with other viewers.

Table 9: Use specification for Sort Movies

Use Case No.	**009**
Use Case Name	**Sort Movies**
Inputs	Sorting criteria (e.g., genre, release year, country).
Outputs	[If success] Display movies sorted according to the selected criteria. [If fail] Display error message (e.g., "Unable to sort movies").
Actor	User Navigate to the movie sorting -> Select a sorting option. System sorts movies based on the selected criteria. [If successful] Display sorted movie list. [If fail] Display nothing.
Precondition	User is logged in.
Postcondition	Movies are displayed in the desired order.
User story	As a viewer, I want to sort movies to easily find content that matches my preferences

Table 10: Use specification for Delete Users

Use Case No.	**010**

Use Case Name	**Delete Users**
Inputs	User account selection.
Outputs	[If success] User account is deleted, and a confirmation message is displayed. . [If fail] Display error message (e.g., "Failed to delete user").
Actor	Admin Navigate to the user management panel -> Select a user account to delete. Confirm the deletion -> System removes the user account. • [If successful] Display confirmation message. • [If fail] Display an error message.
Precondition	Admin is logged in.
Postcondition	The selected user account is deleted.
User story	As an admin, I want to remove inactive or inappropriate users to maintain a healthy platform environment.

Table 11: Use specification for Edit Movie Content

Use Case No.	**011**
Use Case Name	**Edit Movie Content**
Inputs	Movie details (e.g., title, metadata).
Outputs	[If success] Updated movie content is saved and displayed. . [If fail] Display error message (e.g., "Failed to update movie content").

Actor	Admin Navigate to the movie management panel -> Select a movie to edit. Modify details and submit -> System updates the movie information. • [If successful] Save changes and display updated content. • [If fail] Display an error message.
Precondition	Admin is logged in.
Postcondition	The movie's content is updated successfully.
User story	As an admin, I want to update movie information to ensure accurate and relevant content is available to users.

Table 12: Use specification for Delete Comment

Use Case No.	**012**
Use Case Name	**Delete Comment**
Inputs	Selected comment.
Outputs	[If success] Comment is deleted, and a confirmation message is displayed. . [If fail] Display error message (e.g., "Failed to delete comment").
Actor	Admin Navigate to the comment management panel -> Select a comment to delete. Confirm the deletion -> System removes the selected comment. • [If successful] Display confirmation message. • [If fail] Display an error message.
Precondition	Admin is logged in.

Postcondition	The selected comment is deleted.
User story	As an admin, I want to delete inappropriate or irrelevant comments to maintain a positive user experience.

2.2. Functional requirements

Movie Search: Search for your favorite movies easily using our advanced search engine. Filter results by title, actors, directors, genre, and year of release. Our intelligent recommendation system suggests movies based on your watch history and preferences so you will never run out of something new to watch.

Superior Streaming Quality: Immerse yourself in the cinematic experience with our high-definition (HD and Full HD) streaming capabilities coupled with pristine audio quality. Customize your viewing experience through options such as playback speed adjustment, video quality selection, and subtitle preferences. Appreciate the flexibility to enjoy your favorite films offline by utilizing our user-friendly download functionality.

Custom Watchlists: Organize your movie library by creating custom watchlists. Create a few lists to categorize movies by genre, emotional tone, or whatever else you see fit. Effortlessly return to your favorite films or find new ones easily.

Discussions and Reviews: Share your opinions and engage with other movie buffs through the review and critique of films. Our active community allows discussions of your favorite films, uncovers unknown masterpieces, and builds a flourishing film culture.

Registration and Access: Fast-track the account creation process to have full access to the site's features. You can also log in using your already existing Google or Gmail account for better usability. On successful authentication, your watch history, watchlists, and personal preferences will be synced on all of your devices.

2.3. Non-Functional Requirements

2.3.1. User Interface (UI) Design

Consistency: The UI should be consistent in terms of look and feel throughout the app; that is, using the same color scheme, typography, and layout, which gives a harmonious and recognizable experience to the users.

Usability: It means the UI must be intuitive and easy to navigate so as to minimize the learning curve of users. Clear and concise labeling, together with the logical placement of elements, enhance usability.

Accessibility: The UI complies with accessibility standards like WCAG for users with disabilities; in other words, providing alternative text for images and sufficient color contrast and being able to navigate the platform using a keyboard.

Responsiveness: It should be able to adjust to different screen sizes and devices easily, maintaining the same experience on each platform.

Aesthetics: The UI design should be visually appealing yet in line with the global brand identity. The right color palettes, typographic options, and imagery can elevate the experience for users in a meaningful way.

2.3.2. Content Presentation

Clarity: Content should be presented in a clear, concise, and easy-to-understand manner. Avoid jargon and complex language.

Accuracy: All content should be accurate, up-to-date, and free from errors.

Consistency: The style and tone of the content should be consistent throughout the application.

Interactivity: Content should be engaging and interactive, encouraging users to explore and learn more.

Relevance: Content should be relevant to the target audience and meet their specific needs.

2.3.3. Customization

Flexibility: Users should be able to personalize their experience by customizing settings and preferences.

Usability: Customization options should be easy to find and use, with clear instructions.

Security: Customization options should not compromise the security of the application.

Compatibility: Customized settings should not interfere with the core functionality of the application.

2.3.4. Administrative Interface

Security: The administrative interface should be protected with strong authentication mechanisms to prevent unauthorized access.

Usability: The administrative interface should be designed to be efficient and easy to use, allowing administrators to manage the system effectively.

Flexibility: The administrative interface should provide a wide range of tools and features for managing the system.

Reporting: The system should generate detailed reports on system usage and performance.

Scalability: The administrative interface should be able to handle increasing workloads as the system grows.

2.4. Design

2.4.1. Interface Requirements

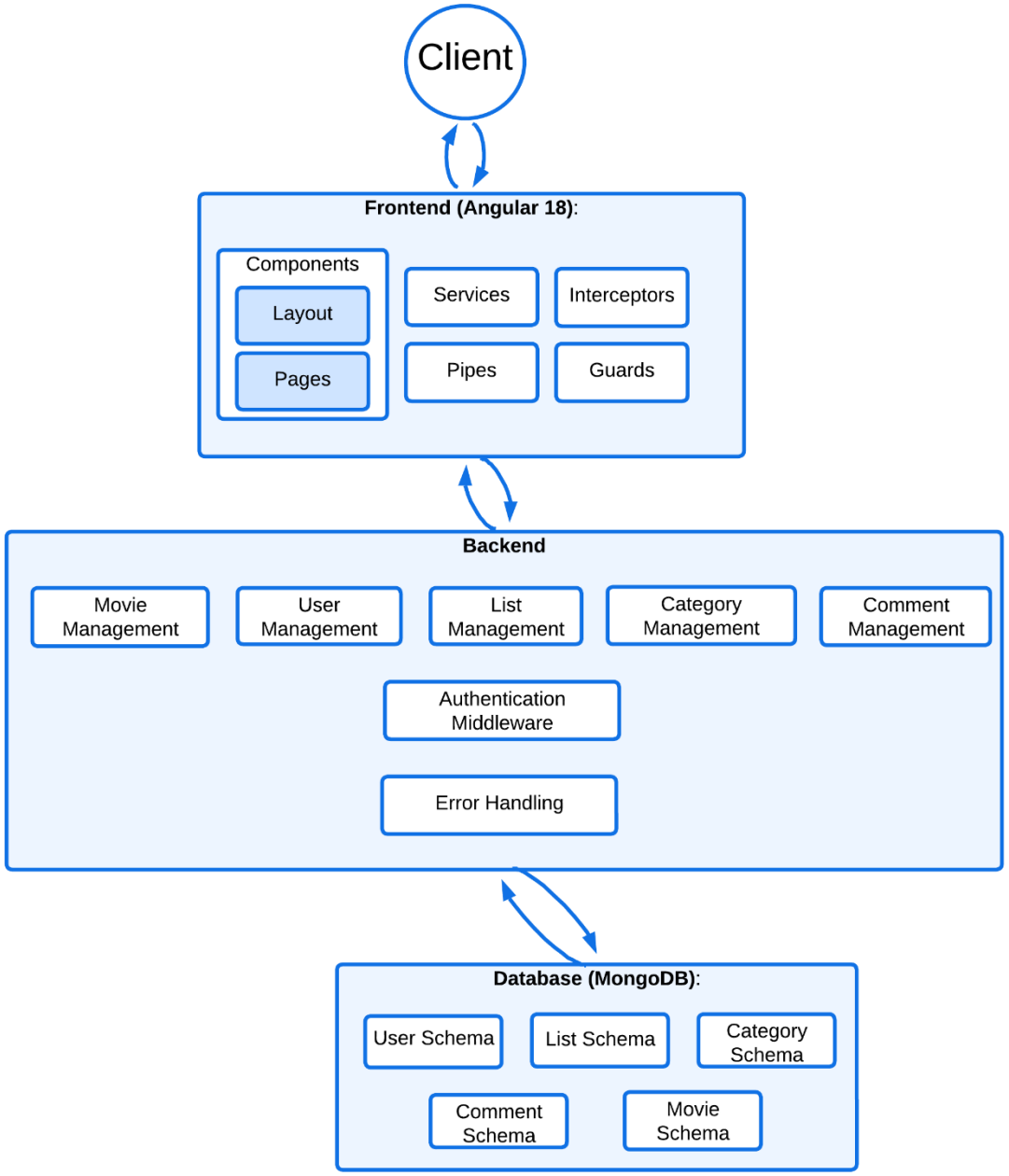

***Figure 5**: System architecture model*

The architecture of the movie platform is designed to prioritize scalability, data security, and efficient system operations. It is divided into several layers and components, each responsible for specific tasks:

Hosting

- The platform is hosted using a reliable and scalable hosting service that supports MEAN stack applications (MongoDB, Express.js, Angular, Node.js). The web application and database are deployed separately to ensure optimal performance and security.

Frontend

- Main Content: This web app has been designed with an interface using Angular, which is a powerful framework allowing easy building and maintenance of dynamic and fast user interfaces. Using the component-based architecture in Angular, the web app guarantees seamless, adaptive user experiences across various devices and display sizes.
- Service State: To effectively manage application state, Angular Services are used. This approach collects data and logic together, thus making the code reusable and maintainable. The services handle functions such as fetching data from APIs, storing user preferences, and managing authentication.
- Guards and Pipes: Guards control access to specific routes based on user role and authentication status. Pipes process and transform data for display, enhancing usability
- Angular Router: The Angular Router enables one to move fluently between multiple views and routes within an application structure. In this way, by defining routes and their respective components, it becomes possible to dynamically present relevant content depending on user action and the URL. This significantly enhances the user experience by providing a coherent transition between pages.

Backend

- Core Features: The backend is built using Node.js and Express.js, managing key functions such as User Management, Movie Management, List Management, Category Management, and Comment Management. Each module is responsible for handling data operations and business logic respectively.
- Authentication Middleware: User authentication is implemented using JSON Web Token (JWT) to secure access and determine user permissions for specific functions.
- Error Handling: The centralized error handling mechanism effectively captures and resolves application failures, ensuring system stability and reliability.

Database

- Schema: The database leverages MongoDB to store data. The NoSQL structure allows for flexible and scalable data processing, supporting a wide range of data types and relationships. Specific schemas include:
    - User schema: Stores user details and permissions.
    - Movie schema: Manages movie metadata such as title, genre, release year, and rating.
    - List schema: Handles user-created lists such as watchlists and favorites.
    - Category schema: Defines categories for movies (e.g., genre).
    - Comment schema: Stores user comments and associated metadata (e.g. timestamps).

This MEAN stack architecture enables a solid and secure system architecture to benefit the end-users with smooth operation and functions alongside maintaining an effective operation.

2.4.2. Class diagram

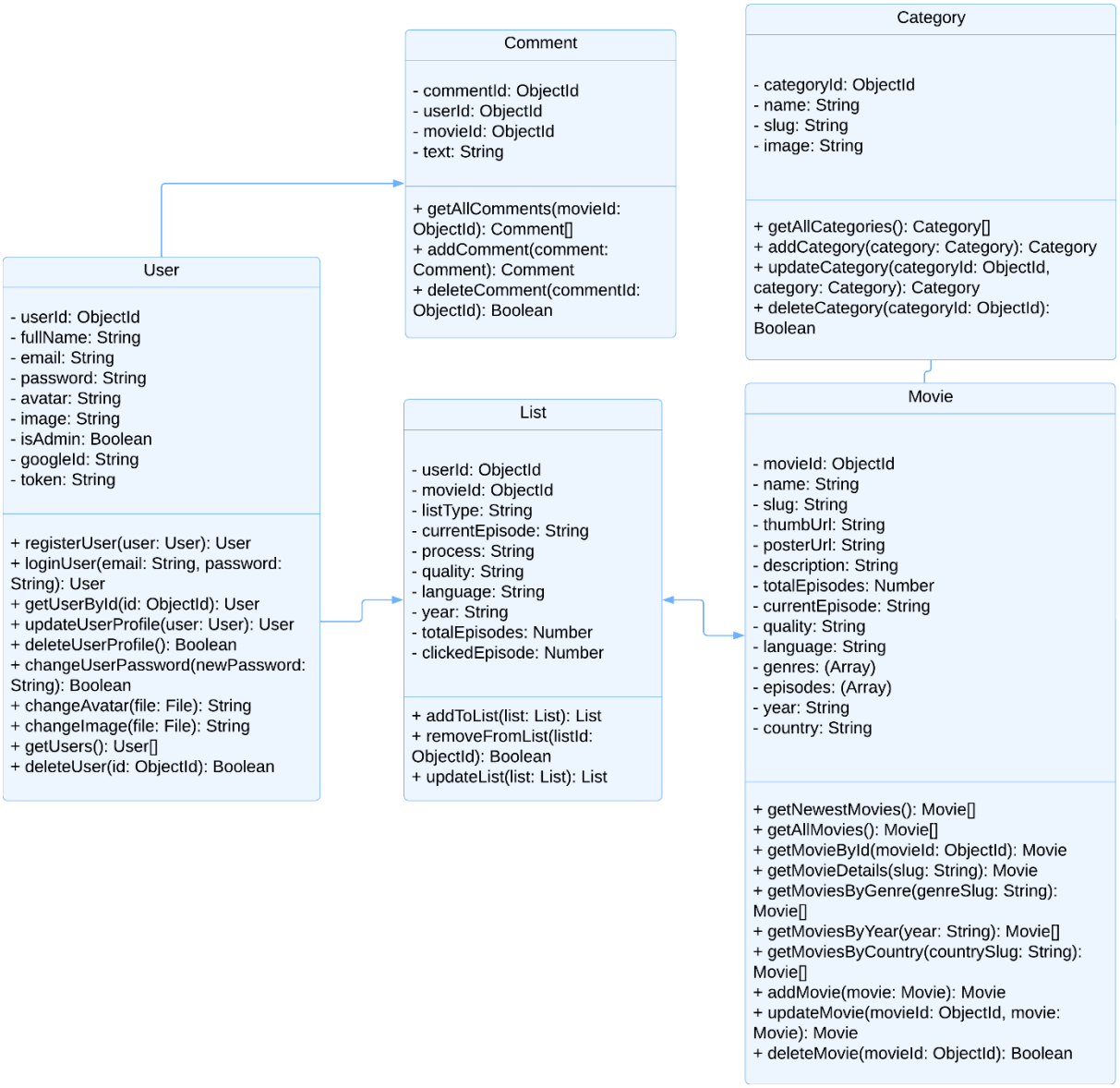

Figure 6: *Class Diagram of Movie Website*

CHAPTER3

IMPLEMENT

3.1. Sign Up page

The sign-up page allows users to create new accounts, granting them access to exclusive features and services within the system.

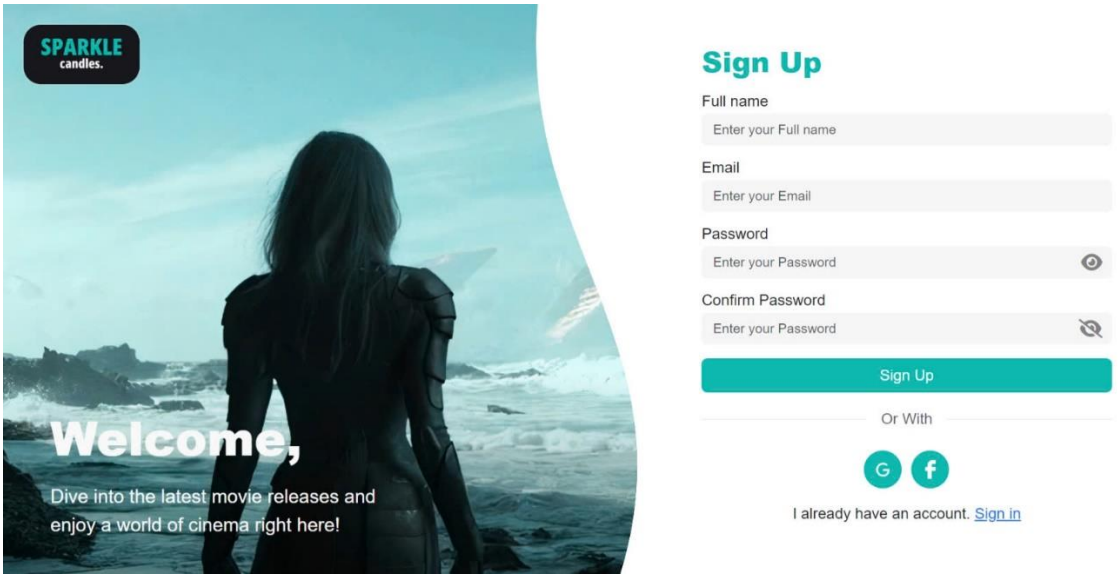

Figure 7: *Sign-Up Page*

3.1. Sign In page

The sign-in page allows users to authenticate their identity by entering their registered account information, thereby granting access to the system's features and services

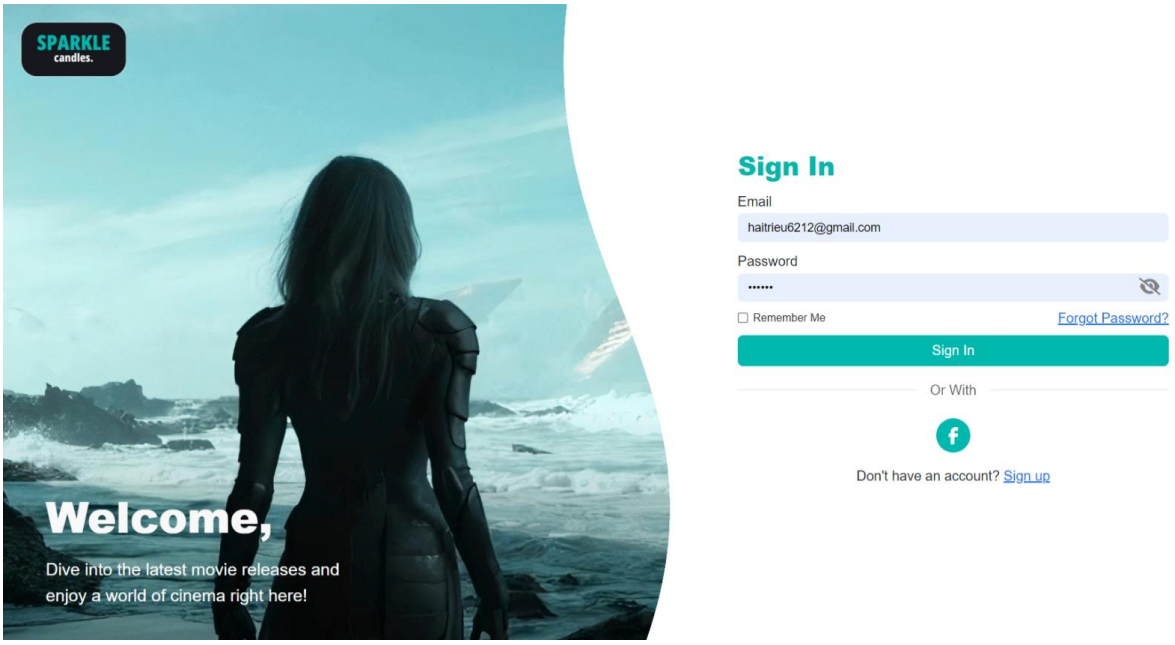

Figure 8*: Sign-In Page*

3.3. Home page

The homepage serves as the primary entry point of the application, providing an overview of key features, highlighted content, and quick access functionalities.

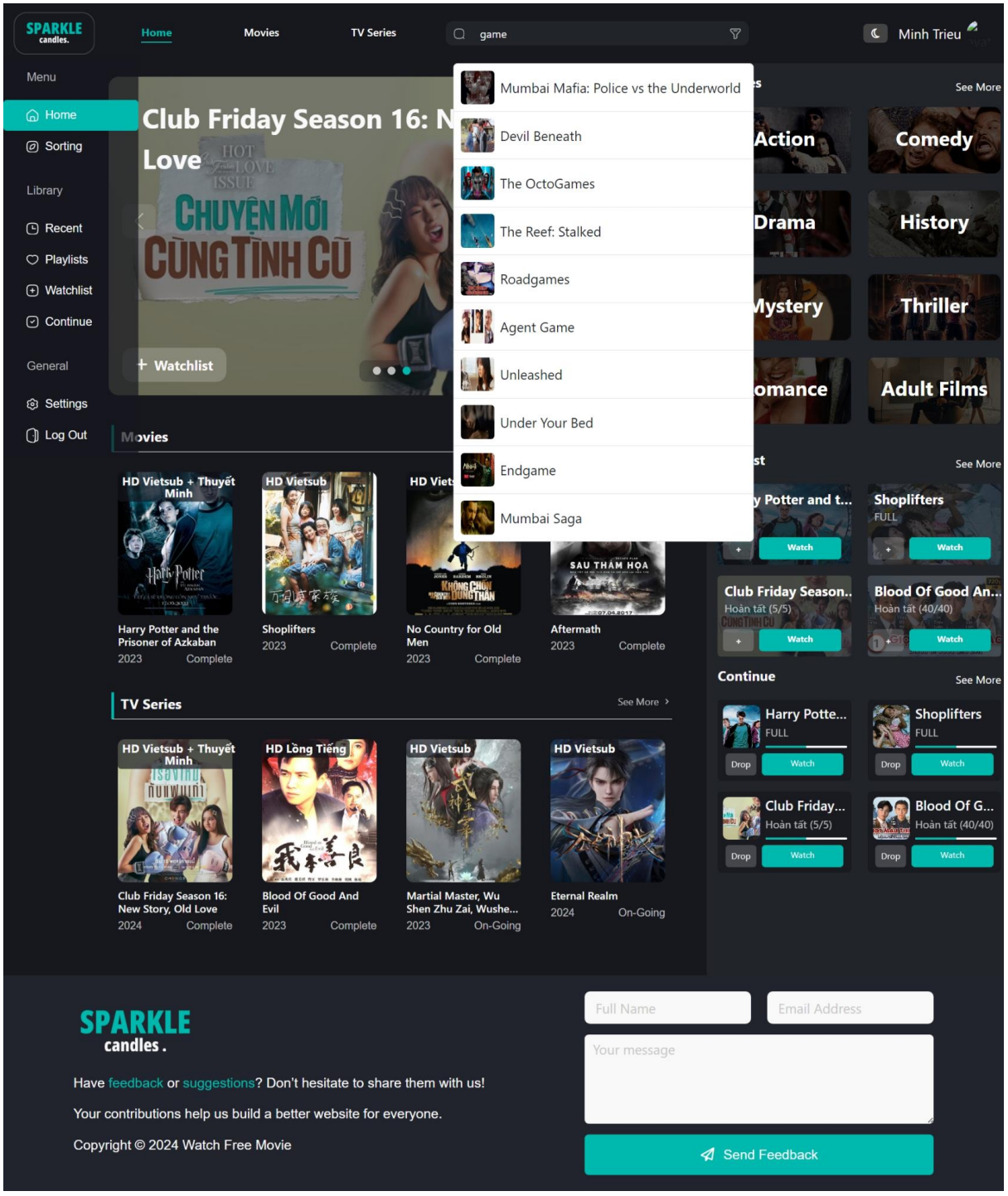

***Figure 9**: Home Page*

3.4. Sorting page

The sorting page allows users to filter and organize content based on specific criteria such as genre, country, and year, providing a customized and efficient browsing experience."

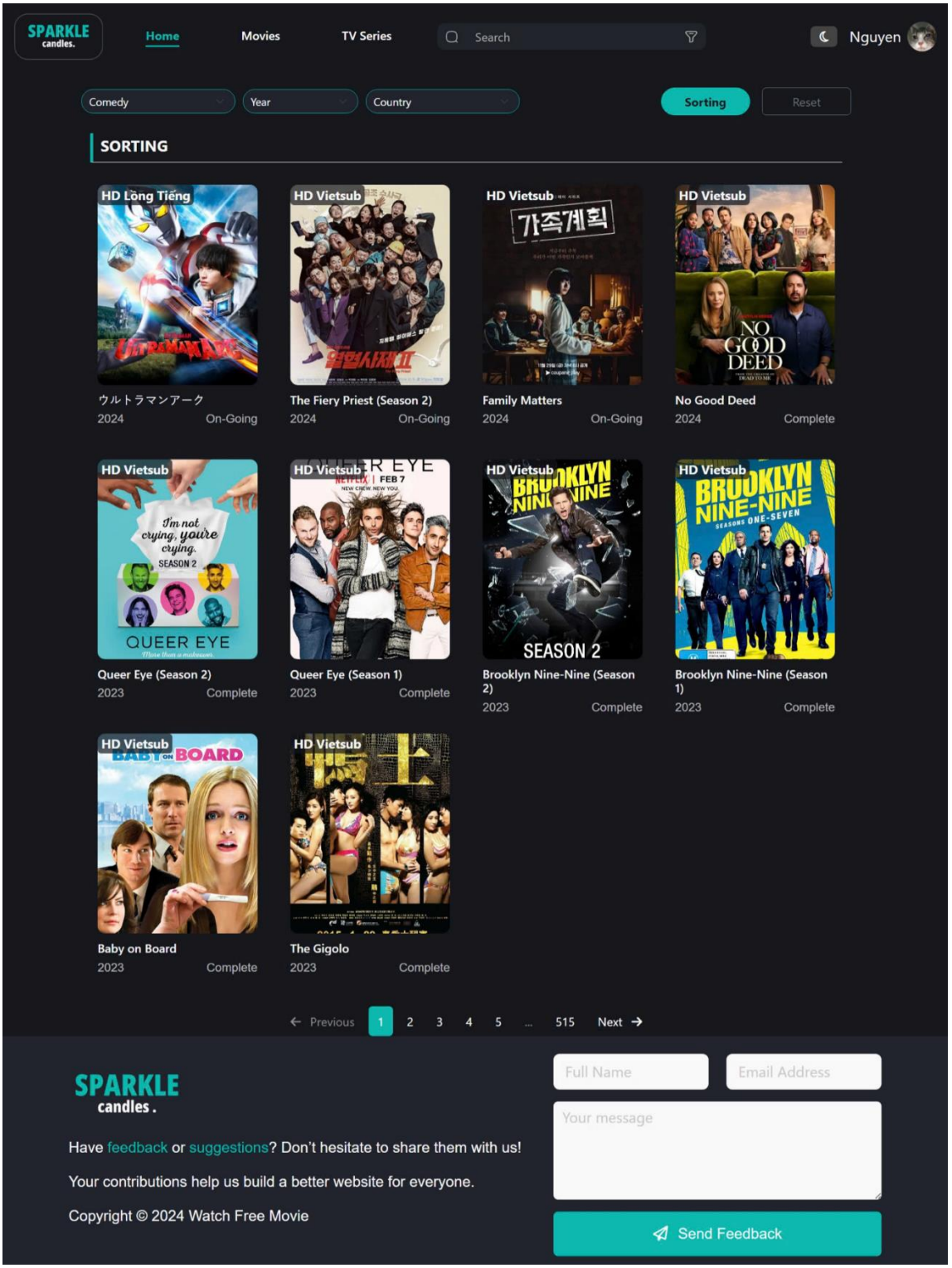

***Figure 10**: Sorting Page*

3.5. Description page

The description page provides comprehensive details about a specific film, including its plot, cast, crew, release date, and user ratings, allowing viewers to make informed decisions about whether or not to watch it.

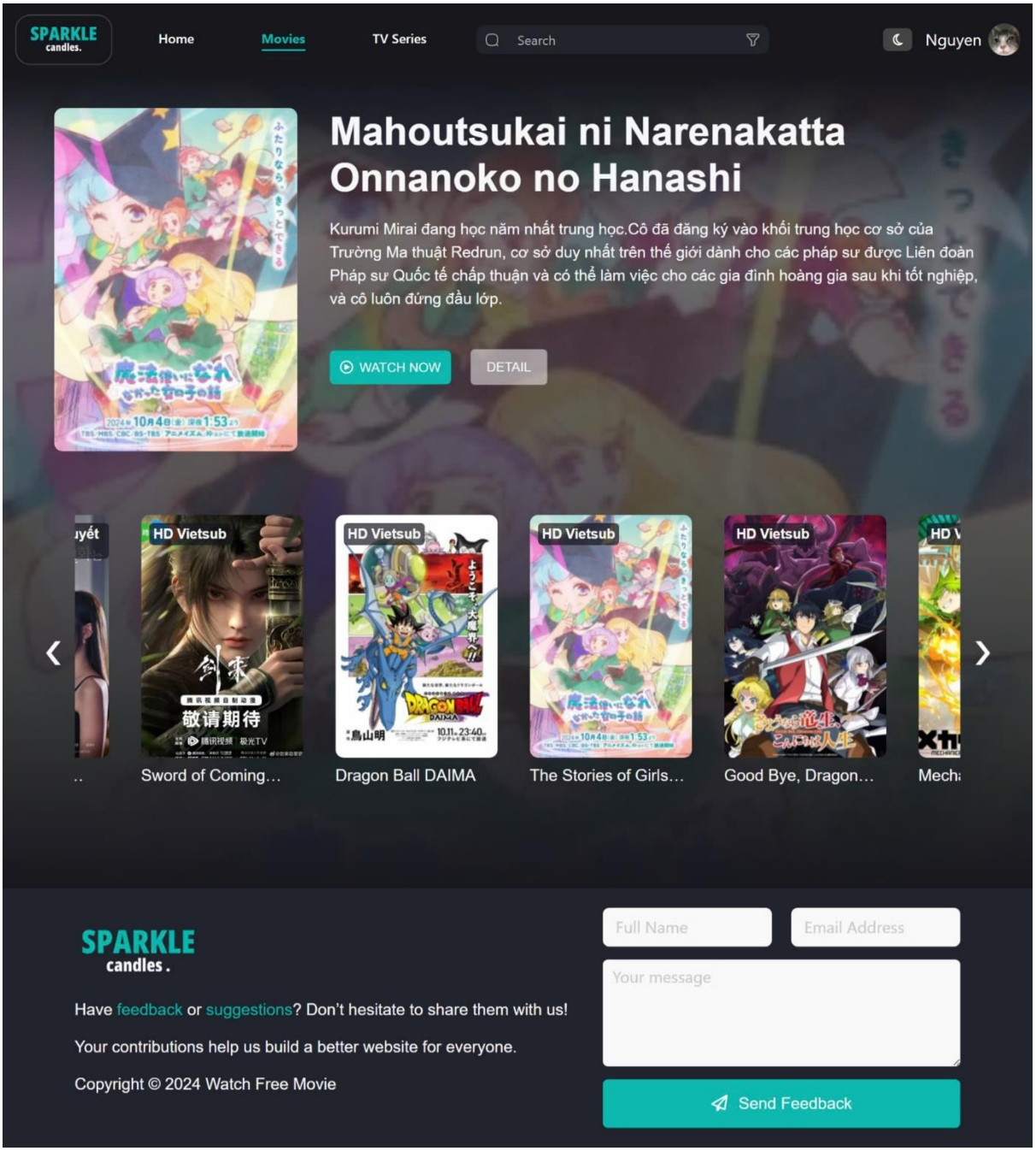

***Figure 11**: Description Page*

3.6. Watching page

The watching page is where users can stream their selected films, providing a seamless viewing experience with features such as play, pause, rewind, and adjustable playback speed.

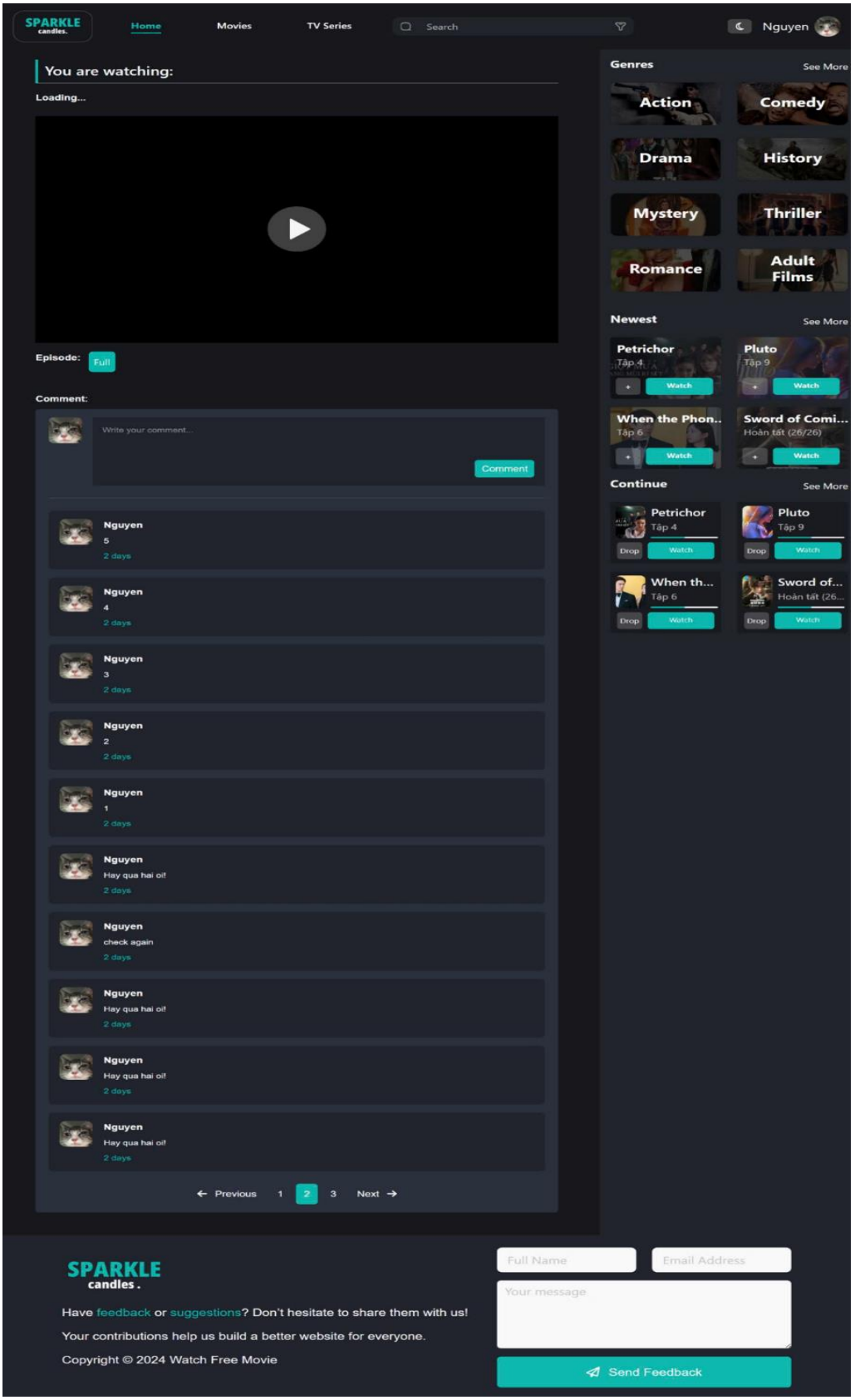

Figure 12: *Watching Page*

3.7. TV Series page

The TV series page offers a dedicated platform for users to watch their favorite shows, providing features such as episode guides, season selection, and the ability to create personalized watchlists.

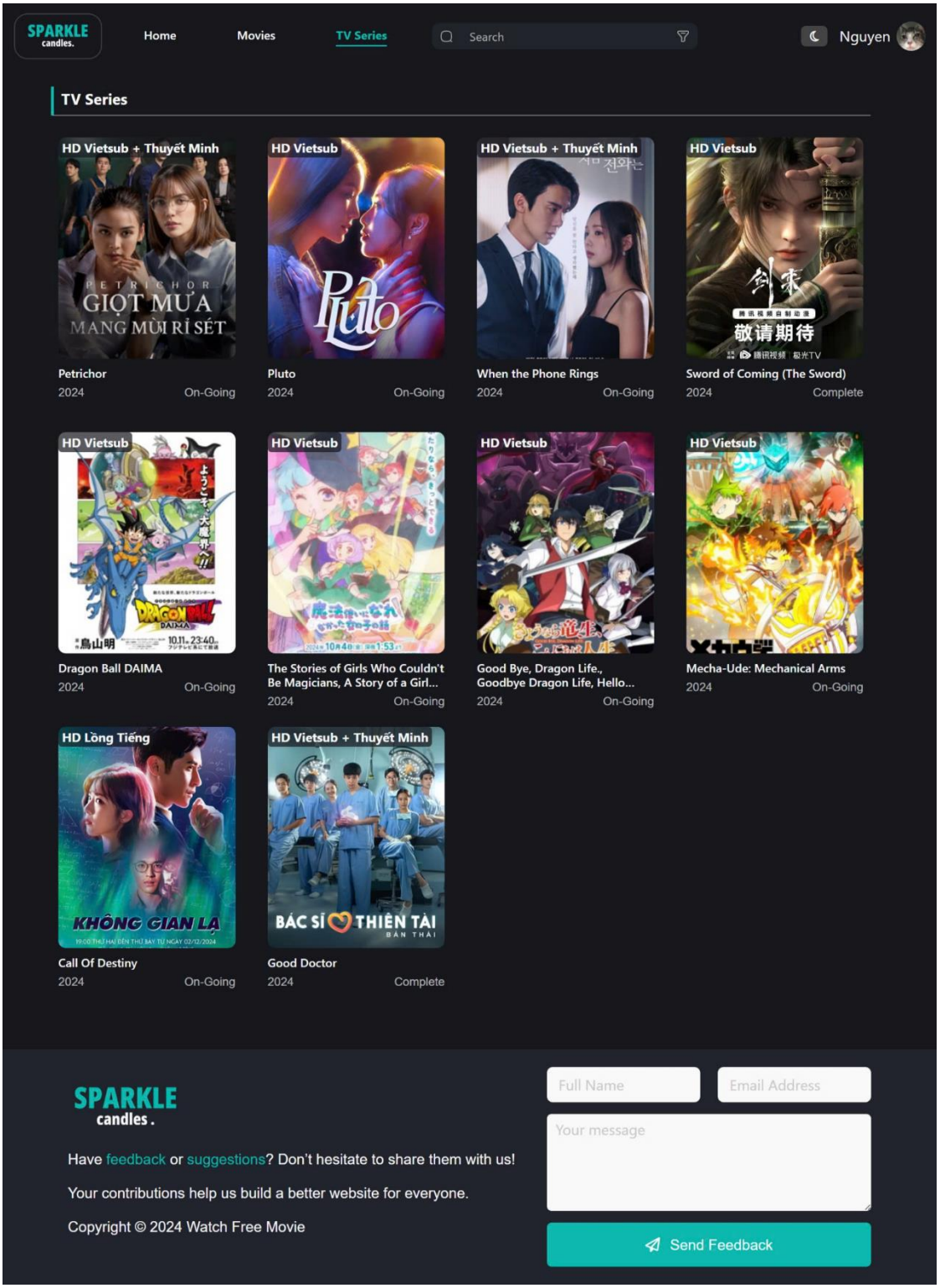

***Figure 13**: TV Series*

3.8. Setting page

The settings page allows users to manage their personal information, viewing preferences, and watch history, creating a tailored and personalized experience

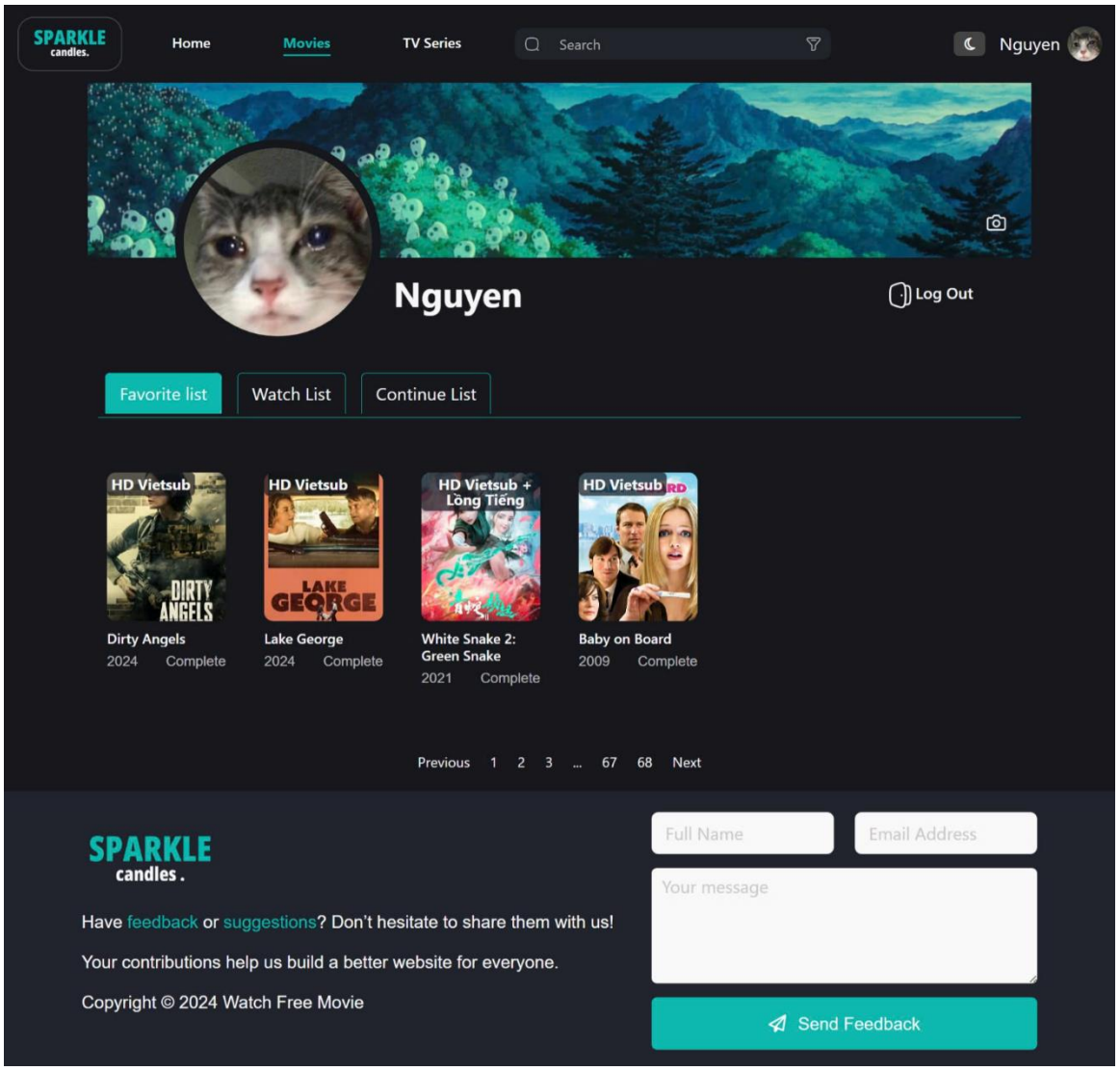

Figure 14: *Setting Page*

3.9. Movies page

The movies page serves as a curated collection of the latest cinematic releases, providing users with easy access to newly released films and upcoming titles

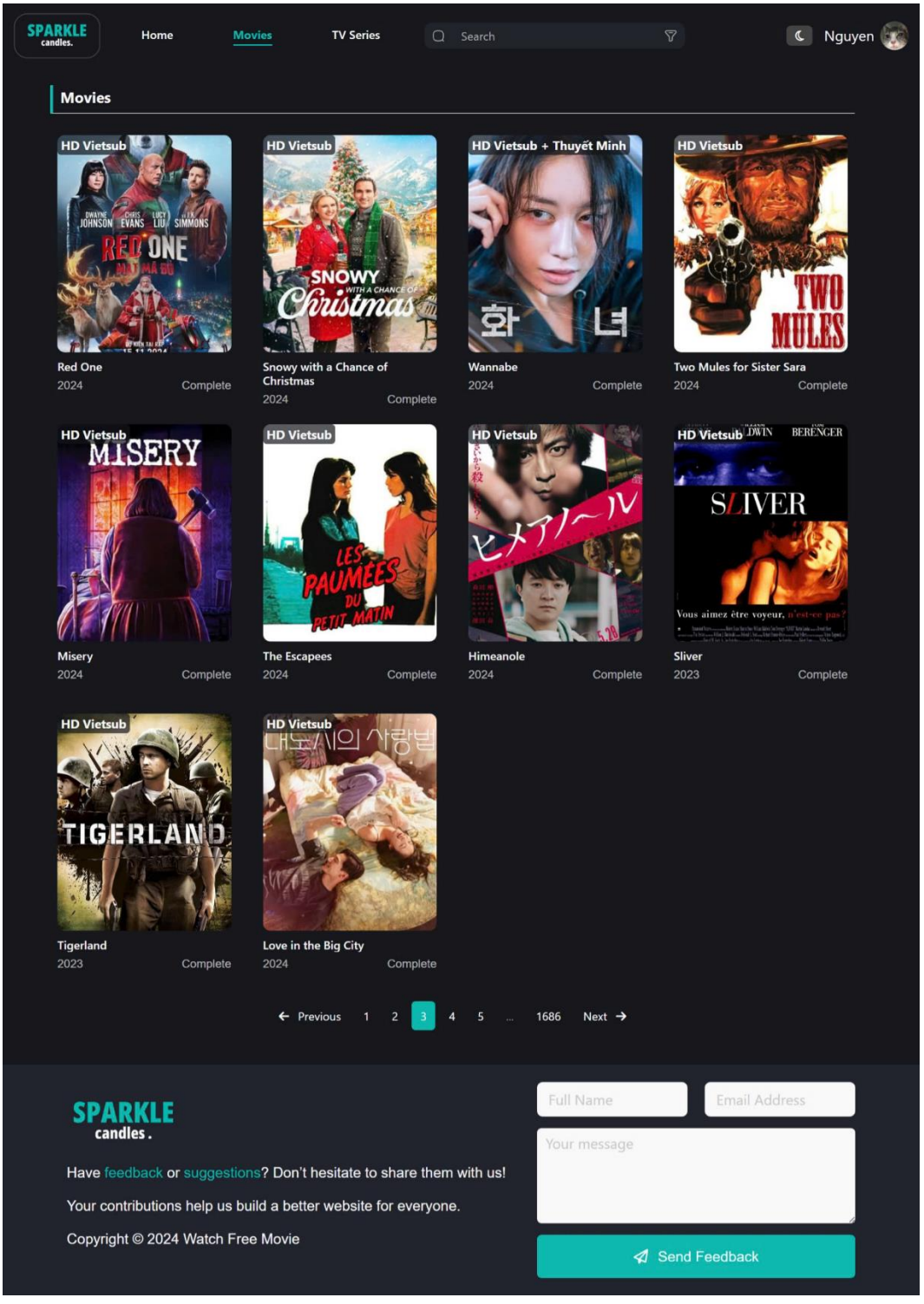

***Figure 15**: Movies Pages*

CHAPTER4

DISCUSSION AND CONCLUSION

4.1.Discussion

SPARKLE Candles is a remarkable website that provides a smooth and interactive movie viewing experience on an online platform. This online platform has been developed keeping in mind the needs of various audience groups. It will provide ease of use in the interface while integrating movie viewing, sorting, and categorizing features in a huge library of movies and TV series that is updated regularly. The development process follows Agile and Scrum methodologies, thus allowing the team to effectively incorporate user feedback and adapt the product to meet changing preferences and needs.

During the development phase, some of the challenges faced by the team were:

- User Interface Design: The main challenge was to ensure the interface is intuitive and appealing for users of all ages and technical levels. Ensuring the best user interface across different devices, from mobile to desktop, requires extensive testing and multiple design iterations.
- Content management and updates: Building a huge library of movies and TV shows across different genres while updating them regularly to meet viewer preferences is time-consuming. Moreover, maintaining quality metadata for movies, such as descriptions, genres, and appropriate subtitles, adds to the complexity.
- Scalability: As user traffic increases, ensuring that the platform's backend infrastructure can handle the large volume without compromising performance becomes a critical task. Efficient database optimization and cloud-based solutions play a vital role in delivering a smooth streaming experience.
- Security and Licensing: Protecting user accounts, payment information, and licensed content requires implementing strong security measures. Ensuring compliance with copyright laws while delivering content globally also requires careful consideration.

4.2.Conclusion

SPARKLE Candles has been one of the leading online movie portals consistent with its mission: to provide a wide diversity of movies and series in safe, comfortable, and highly adaptable environments to their customers. Combining outstanding technology and

paying attention to every tiny detail in its design, this site will make both casual viewers and those who know and love good movies feel comfortable.

In the future, SPARKLE Candles plans to add more content, introduce advanced personalization features, and further localize. This will help the platform to become an entertainment home for every family across the globe and continue with the evolution of the audience's changing preferences.

REFERENCES

O. S. Wael Zayat, "Framework Study for Agile Software Development Via Scrum and Kanban," International Journal of Innovation and Technology Management, vol. 17, no. 4, p. 2030002 (24 pages), 2020.

M. E. Hema Krishnan, "MongoDB – a comparison with NoSQLdatabases," International Journal of Scientific & Engineering Research, , vol. 7, no. 5, pp. 1035-1037, May 2016.

Ryan Boyd, "Exploring OAuth 2.0 for API Security," Pro RESTful APIs, vol. 3, no. 6, pp. 220–240, November 2020

Minko Gechev, "Exploring Angular for Enterprise-Ready Web Applications," Packt Publishing, vol. 18, no. 3, pp. 67–89, 2019.

C. S. Massimo Nardone, "JSON Web Token (JWT) Authentication," in Pro Spring Security, December 2023.

M. Antonio Rossi, "Designing Responsive Websites with Bootstrap 5," Web Development Journal, vol. 22, no. 4, pp. 56–78, April 2023.