FILM VIBE

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

BTECH (Computer Engineering)

TO

RK UNIVERSITY, RAJKOT

SUBMITTED BY

Name of Student	Enrollment No.
DHWAJ SHAH	22SOECE13034
FAIZAN KHIMANI	22SOECE13021

UNDER THE GUIDANCE OF

Internal Guide
Prof. Neha Chauhan
RK University
Rajkot

SCHOOL OF ENGINEERING, RK UNIVERSITY, RAJKOT

DECLARATION

I/We hereby certify that I/We am/are the sole author(s) of this project work and that neither any part of this project work nor the whole of the project work has been submitted for a degree to any other University or Institution. I/We certify that, to the best of my/our knowledge, my/our project work does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my/our project document, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. I/We declare that this is a true copy of my/our project work, including any final revisions, as approved by my/our project review committee.

Signature of Student (S)

Khimani Faizan Yusufbhai (22SOECE13021)

Date: _____

Place: _____

Dhwaj Shah (22SOECE13034)

Date: _____

Place: _____

CERTIFICATE

This is to certify that the work which is being presented in the Project Report entitled **"FilmVibe",** in partial fulfillment of the requirement for the award of the degree of **B.Tech. (Computer Engineering)** and submitted to the School of Engineering, RK University, is an authentic record of my/our own work carried out during a period from **June 2024 to December 2024.**

The matter presented in this Project Report has not been submitted by me/us for the award of any other degree elsewhere.

Signature of Student (S)

Khimani Faizan Yusufbhai (22SOECE13021) Dhwaj Shah (22SOECE13034)

This is to certify that the above statement made by the student(s) is correct to the best of my knowledge.

Internal Guide	Head of Department
Prof. Neha Chauhan	Dr. Chetan Shingadiya
Assistant Professor,	CE / IT
RK University ,	School of Engineering,
Rajkot	RK University, Rajkot

SCHOOL OF ENGINEERING
RK UNIVERSITY

SCHOOL OF ENGINEERING, RK UNIVERSITY, RAJKOT

Acknowledgement

I would like to express my sincere gratitude to my project guide, **(Prof. Neha Chauhan)**, for their continuous support, guidance, and valuable insights throughout the development of this project. Their expertise and constructive feedback helped me overcome various challenges and improve the quality of my work.

I also extend my heartfelt thanks to the **faculty members** of the **School of Engineering, RK University, Rajkot**, for providing me with the necessary resources and a conducive learning environment.

A special mention goes to my family and friends, whose constant encouragement and motivation helped me stay focused during the project.

Finally, I would like to thank everyone who contributed directly or indirectly to the successful completion of this project. Your support has been invaluable.

Abstract

The project titled **"FileVibe"** is a dynamic movie website developed to provide users with an engaging and user-friendly platform to explore, discover, and learn about movies. Utilizing modern web technologies such as **React** for the frontend, **Tailwind CSS** for styling, and **MongoDB** for backend integration, FileVibe offers seamless navigation, real-time data updates, and an appealing user interface.

The platform enables users to search for movies, view detailed descriptions, including genres, release dates, and cast information, and explore movie reviews and ratings. The goal of this project is to create a scalable, secure, and highly responsive movie database that enhances the user experience and makes information readily available.

Through its innovative design and efficient backend, FileVibe aims to serve as a comprehensive movie platform for both casual viewers and movie enthusiasts. The system is built with scalability and future enhancements in mind, ensuring that it can adapt to changing user needs and technological advancements.

Notations, Naming Convention, and Abbreviations

Notations:

- **UI** – User Interface
- **UX** – User Experience
- **API** – Application Programming Interface

Naming Conventions:

- **Component Naming (React):**
  Components are named using **PascalCase** (e.g., MovieList, UserProfile, SearchBar).
- **CSS Classes (Tailwind CSS):**
  Tailwind CSS utility classes are used for styling, following **kebab-case** for naming (e.g., bg-gray-100, text-center, mt-4).
- **Variables and Functions:**
  Variables and functions in JavaScript follow **camelCase** (e.g., fetchMovies, userData, handleSubmit).
- **Database Naming (MongoDB):**
  Database collections and documents follow **snake_case** for easier readability and consistency (e.g., user_profiles, movie_reviews, movie_data).

Abbreviations:

- **CRUD – Create, Read, Update, Delete**
- **DB – Database**
- **JSON – JavaScript Object Notation**
- **HTTP – Hypertext Transfer Protocol**
- **JS – JavaScript**
- **MVC – Model View Controlle**

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Project Summary

FileVibe is a web-based platform that provides users with a comprehensive movie database. Users can search for films, explore genres, view movie descriptions, and check ratings and reviews. Built with React for the frontend, Tailwind CSS for styling, and MongoDB for the database, the platform focuses on a responsive, scalable design with seamless navigation.

1.2 Purpose: Goals & Objectives

FileVibe's main goal is to offer a robust movie search platform with intuitive navigation and detailed film information. Specific objectives include:

- User-friendly Interface: Ensuring easy movie searches and browsing.
- Detailed Movie Descriptions: Offering rich data including plot summaries, cast, and more.
- Scalability & Security: Using MongoDB for secure data storage and scalability.

1.3 Scope

FileVibe enables users to:

- Search and browse movies by genre or title.
- Access detailed descriptions and reviews.
- Read existing ratings, though users cannot yet contribute reviews or ratings. It does not offer movie streaming services.

1.4 Technology and Literature Review

- React: For dynamic UI components.
- Tailwind CSS: For a clean, responsive design.
- MongoDB: Provides database management, user authentication, and secure storage.

CHAPTER 2: PROJECT MANAGEMENT

2.1 Project Planning and Scheduling

- 2.1.1 Development Approach: Agile methodology ensures continuous feedback and iteration.
- 2.1.2 Project Plan: Key milestones include project initiation, database setup, frontend development, testing, and deployment. The team consists of frontend and backend developers, and testers.
- 2.1.3 Schedule Representation: A Gantt chart details the timeline and tasks.

2.2 Risk Management

- 2.2.1 Risk Identification: Key risks include performance bottlenecks, security vulnerabilities, and scalability issues.
- 2.2.2 Risk Analysis: Performance risks are deemed low, while scalability and security risks are moderate.
- 2.2.3 Risk Planning: MongoDB’s scalable architecture and built-in security measures will mitigate risks.

CHAPTER 3: SYSTEM REQUIREMENT STUDY

3.1 User Characteristics

The target audience includes general users who seek movie information and possess basic web navigation skills.

3.2 Hardware and Software Requirements

- Hardware: Standard client devices (laptops, smartphones) with internet access.
- Software: Modern web browsers, React frontend, MongoDB backend.

3.3 Constraints

- Regulatory Policies: Compliance with data privacy regulations.
- Interfaces: Integration with MongoDB for data management and storage.
- Security: Enhanced with MongoDB’s authentication protocols.

CHAPTER 4 : SYSTEM STUDY

4.1 Study of the Current System

-IMDb and Other Existing Platforms:

Platforms like IMDb and Rotten Tomatoes provide users with extensive movie information, reviews, and ratings. However, the complexity of these systems often overwhelms users, especially those looking for a simple and fast solution to discover trending movies and download them. FilmVibe intends to offer a more user-friendly, responsive interface that focuses on real-time data retrieval with minimalistic design and easy navigation.

4.2 Problems and Weaknesses of the Current System

-Complex User Interfaces:

Many existing platforms offer too many features that can complicate the user experience. Users may struggle to find basic functionalities like searching for the latest trending movies or accessing download links quickly.

- Lack of Real-time Updates:

Existing platforms do not provide real-time data updates, making it difficult for users to access the latest movies and ratings instantly. Delays in updates reduce user engagement, especially for users who want timely information.

4.3 Requirements of the New System

The new system, **Film Vibe ,** will meet the following requirements:

- User-friendly Interface:

The system will have a clean, intuitive, and responsive interface that can be easily navigated across devices.

- Real-time Data Retrieval:

Movies, ratings, and reviews will be updated in real-time to keep users engaged with the most current information.

- Secure and Scalable Architecture:

The system will incorporate MongoDB for secure and scalable database management and authentication.

4.4 Feasibility Study

- Technical Feasibility:

Film Vibe will be developed using React for the front-end and MongoDB for the back-end. React's component-based architecture allows for reusable UI components, making development and scaling easier. MongoDB provides real-time databases, making it ideal for the project's needs, along with built-in authentication for user security.

- Operational Feasibility:

By simplifying the user interface and providing real-time updates, the system is designed to enhance user experience. The focus on minimalism and performance ensures that users will have an easy and enjoyable experience while using FilmVibe.

4.5 Requirements Validation

All requirements for FilmVibe have been validated through a combination of:

- User Testing:

A group of users was asked to test early prototypes of the system to ensure the design met expectations and provided smooth interactions.

- System Testing:

Extensive testing was performed to ensure that the system works as expected in various environments and under different load conditions.

4.6: Functions of the System

4.6.1 Use Cases

The primary use cases for the system include:

- Movie Search:

Users can search for movies by title, genre, or popularity. The system will display relevant results, including movie images, ratings, and release dates.

- View Movie Details:

Users can view detailed information about any movie, including a synopsis, cast, reviews, and available download options.

- Read Reviews:

Users can read reviews and ratings of the movies. Future versions will allow user-generated reviews and ratings.

4.7: Data Modeling

4.7.1 ER Diagram (Entity Relationship Diagram)

The ER diagram depicts relationships between:

- Users: Representing individual system users.

- Movies: Representing movie records, including details like titles, genres, and ratings.

4.7.2 Data Dictionary

A detailed list of key database fields includes:

- User Table: Contains fields such as user_id, username, email, and password.

- Movies Table: Contains movie_id, title, genre, release_date, rating, etc.

- Reviews Table: Contains review_id, user_id (foreign key), movie_id (foreign key), review_text, rating, and timestamp.

4.8: Functional and Behavioral Modeling

4.8.1 Data Flow Diagram (DFD)

Level 0 DFD:

This provides a high-level overview of the system, illustrating how data flows between the user, the movie database, and the review system.

Level 1 DFD:

This offers a more detailed view, showing specific processes such as movie search, review display, and user authentication.

Chapter 5: System Design

5.1 Database Design

The database for FilmVibe is designed with the following key tables:

- Users Table stores all user-related data.

- Movies Table stores movie details such as title, genre, and rating.

- Reviews Table contains all reviews submitted by users, including the rating and review text.

5.2 System Procedural Design

Flowcharts provide an overview of:

- Movie Search Process:

Users enter a search term, and the system queries the database for relevant matches, returning results with movie posters and key details.

- Navigation Process:

Describes how users navigate between different pages such as the homepage, genre-specific pages, and movie details.

5.3 Input/Output and Interface Design

The design follows modern UX/UI principles with:

- Input: Search functionality, user login, and genre selection.

- Output: Responsive movie details, reviews, and ratings are presented clearly on mobile and desktop devices.

Chapter 6: Implementation

6.1 Implementation Environment

- Front-end Technology: React.js, offering component-based architecture.

- Back-end Technology: MongoDB, providing real-time data synchronization, user authentication, and secure storage.

6.2 Program/Module Specification

Each module, such as movie search, user authentication, and review display, is implemented as a self-contained component. This ensures consistency in development and code readability.

6.3 Security Features

The system uses MongoDB Authentication for secure user login and management. Users' data, including passwords, are encrypted, and the system is protected against common web security vulnerabilities like SQL injection and XSS.

6.4 Coding Standards

JavaScript coding standards, such as variable naming conventions, code indentation, and commenting, were strictly followed. Each function and module is thoroughly documented to ensure maintainability.

Chapter 7: Testing

7.1 Testing Plan

The system was tested in phases:

- Unit Testing: Ensured that individual components (e.g., search bar, movie card) function correctly.

- Integration Testing: Verified that different components work together seamlessly.

- User Acceptance Testing: Conducted with real users to gather feedback and ensure that the system meets their needs.

7.2 Testing Strategy

Black-box testing was used to test the system from the user's perspective, focusing on input/output behaviors rather than internal code structure.

7.3 Testing Methods

- Functional Testing: Verified that key functionalities like searching for movies and viewing details worked as intended.

- Performance Testing: Ensured that the system responds quickly even with a large volume of data.

7.4 Test Cases

Test cases include:

- Movie Search Test Case: Verifies that users can search for movies by title and genre.

- Authentication Test Case: Ensures that only registered users can log in and access their profiles.

- Data Retrieval Test Case: Confirms that the system retrieves and displays the correct movie details and reviews.

Chapter 8: Screenshots and User Manual

Screenshots include:

- Home Page: Showing trending movies and a search bar.

- Movie Details Page: Displaying movie information, reviews, and download options.

- User Login Page: Secure login interface for user authentication.

The user manual provides a step-by-step guide on how to:

- Search for movies.

- View detailed information.

- Navigate through the website.

Chapter 9: Limitations and Future Enhancements

Limitations

- No User-Generated Reviews:

  The current system does not allow users to post their own reviews or ratings.

- Limited Movie Database:

  The system currently features a limited number of movies compared to larger platforms like IMDb.

Future Enhancements

- User Reviews:

  Implement a feature that allows users to submit reviews and rate movies.

- Recommendation Engine:

  Add a recommendation feature based on user behavior and preferences.

Chapter 10: Conclusion and Discussion

The development of FilmVibe demonstrates how modern web technologies can be harnessed to build a fast, responsive, and user-friendly movie discovery platform. While the system fulfills its initial goals, such as providing real-time data and a simple interface, future versions can include more advanced features like user-generated reviews and personalized recommendations to further enhance the user experience.