MALNAD COLEGE OF ENGINEERING

Under the auspices of M.T.E.S

(An autonomous institution under Visvesvaraya Technological University, Belgaum) Hassan – 573201, Karnataka, India

Report on Course Title: Full Stack Development

"Movie Review System"

Submitted by team number :09

Name	USN
Tejaswi K G	4MC23IS116
Sandesh D R	4MC23IS094
Shreyas Chandra C S	4MC23IS101
Vinay N Lokesh	4MC23IS124
Rathan K V	4MC23IS087

Submitted to

Mr. Krishna Swaroop A

(Assistant Professor Dept. of ISE)

Department of Information Science & Engineering

Malnad College of Engineering

Hassan– 573202

2025-26

Table of Content:

1. Abstract

The Movie Review System is a Django-based web application that lets users browse movies, read reviews, and post their own feedback. It provides a centralized and reliable platform for audience-based movie ratings, helping users make better viewing decisions. Authenticated users can submit comments and ratings, while the admin manages movies and reviews to maintain accuracy. The system offers an organized alternative to scattered online reviews and improves user engagement through a simple and scalable interface.

The Movie Review System is a comprehensive web-based application developed using the Django framework with the primary objective of providing a structured, reliable, and user-friendly platform for movie evaluations. In today's digital world, movies play a vital role in entertainment, culture, and public communication, and viewers depend heavily on online reviews before deciding to watch a film. However, most existing review sources—such as YouTube comments, blogs, social media posts, and promotional reviews—are scattered, unorganized, and often influenced by paid marketing. This inconsistency makes it difficult for users to obtain genuine and unbiased feedback. To address this challenge, the Movie Review System centralizes all user-generated movie reviews in a single, well-organized platform, ensuring transparency and credibility.

The system focuses on building an interactive environment where users can explore a list of movies, view detailed information, check public ratings, and read authentic reviews posted by other viewers. The platform incorporates secure user authentication, allowing only registered users to post reviews and ratings. This approach helps maintain accountability and prevents anonymous spam or irrelevant comments. Each user can write reviews based on their viewing experience, enabling a community-driven evaluation model. By collecting and displaying movie ratings, the system helps individuals make informed decisions about what to watch.

The admin module plays a crucial role in managing the platform. Administrators can add new movies, update existing details, and remove outdated or inappropriate content. They can also monitor reviews posted by users to ensure that the platform remains free from offensive, misleading, or spam content. This moderation enhances the overall reliability and trustworthiness of the system. The admin dashboard is built using Django's built-in admin interface, which simplifies content management and improves system efficiency.

From a technical perspective, the system uses Django's Model-View-Template (MVT) architecture, which ensures separation of concerns and smooth data flow. The database stores user profiles, movie information, user reviews, and ratings. Through Django ORM, the application interacts efficiently with the database, ensuring scalability and performance. The frontend is developed using HTML, CSS, and Bootstrap to provide a clean, responsive, and visually appealing user experience across different devices such as desktops, tablets, and smartphones.

The platform's review and rating mechanism is designed to generate average ratings dynamically, giving users a clear idea of public opinion about each movie

2.Introduction

Movies have become one of the most influential forms of entertainment in modern society. With the rapid growth of the film industry and the increasing number of movies released each year, viewers often rely on online reviews and ratings before deciding which movie to watch. Reviews help audiences understand the quality, storyline, performances, and overall experience a movie may provide. However, despite the importance of reviews, the current review ecosystem is highly fragmented. Information is scattered across platforms like YouTube, social media pages, personal blogs, promotional websites, and various entertainment portals. Many of these reviews are biased, promotional, or lack credibility, making it difficult for users to trust them.

The rise of digital marketing has further blurred the line between genuine reviews and paid promotions. Influencers, celebrities, and content creators often publish reviews influenced by sponsorship agreements, making it challenging for viewers to distinguish authentic opinions from advertisements. Due to this lack of transparency, users may end up watching movies that do not match their expectations, wasting both time and money. Hence, there is a need for a structured, reliable, and user-driven platform that provides unbiased and community-based movie reviews.

To address these challenges, the Movie Review System is designed as a centralized web-based platform where users can explore movies, read authentic reviews, and share their personal opinions. Unlike scattered online sources, this system brings all review-related information into one organized portal. The platform ensures that only registered users can post reviews, which enhances accountability and reduces the chances of spam, fake reviews, or irrelevant comments. By promoting a community-driven model, the system allows users to express their genuine thoughts and collectively shape the overall rating of a movie.

The Movie Review System is built using the Django framework, following the Model-View-Template (MVT) architecture, which provides a strong structure for managing data and displaying information to users. Through this architecture, user requests are processed efficiently, and movie information, reviews, or ratings are displayed in an organized manner. The database stores essential details such as movie titles, genres, release dates, posters, user information, and reviews. Django's Object-Relational Mapping (ORM) ensures secure and systematic interaction with the database, helping maintain data consistency and integrity.

One of the key components of the system is the administrator module. The admin plays an important role in maintaining the quality and authenticity of the platform. The administrator has control over adding new movies, updating details, removing outdated movies, and monitoring user reviews. This helps ensure that inappropriate or misleading content is removed and that users have access to clean and trustworthy information. The built-in Django admin interface further simplifies these tasks, enabling efficient and error-free management of data.

3.Objectives

The main objective of the Movie Review System is to create a structured and user-friendly platform where movie lovers can access authentic reviews and share their own feedback. The system aims to simplify the process of finding reliable information about movies while ensuring that the reviews displayed are genuine and audience-driven.

One of the core goals is to provide a dedicated platform for users to view and post movie reviews. Instead of relying on scattered information across social media, blogs, or promotional websites, the system offers a centralized space where users can browse movies, read opinions from other viewers, and contribute their own reviews. This helps build a community-oriented environment where users actively participate in sharing experiences and shaping the overall perception of movies.

The system also aims to enable secure login and personalized user interaction. Through authentication, only registered users can post ratings and comments, which helps maintain accountability and reduces spam or fake reviews. User sessions ensure that each review is unique to the person who posted it, and personalized access allows individuals to manage their own reviews. This secure environment improves trust and encourages meaningful participation.

Another important objective is to maintain organized movie and review records. The system uses a structured database to store movies, user information, ratings, and comments. By keeping the data well-organized and easy to retrieve, the platform ensures that users can quickly find information about any movie. Admins can manage movie details and monitor user reviews, which helps maintain the quality and accuracy of the data presented.

Finally, the platform is designed to assist users in choosing movies based on audience opinion. With clear ratings, authentic reviews, and an easy-to-navigate interface, users can make informed decisions about whether a movie suits their interests. Average ratings calculated from multiple user reviews provide a balanced view of a movie’s reception. This helps users save time and avoid unreliable promotional content.

Overall, the objectives of this project focus on improving user experience, ensuring data accuracy, promoting transparency, and providing a trustworthy system for movie reviews.

4. System Requirements

The development and deployment of the Movie Review System require a well-defined set of software and hardware resources to ensure smooth operation, high performance, and a user-friendly experience. Since the application uses Django for the backend and React.js with Tailwind CSS for the frontend, the system must support both backend and modern frontend development workflows. The following sections describe the required software and hardware components in detail.

4.1 Software Requirements

To build and run the Movie Review System, several important software tools and frameworks are required. These tools help with backend development, frontend development, database handling, and overall project management.

1. Python (Version 3.10 or above)

Python is the core programming language used to build the backend of the Movie Review System. Known for its simplicity and readability, Python makes backend logic easier to implement. The language supports powerful libraries and frameworks such as Django, which enhance rapid development, security, and scalability. Python also integrates easily with modern frontend technologies, making it suitable for full-stack applications.

2. Django Framework (Version 4.2 or above)

Django is a high-level Python framework based on the MVT (Model-View-Template) architecture. It handles backend operations such as:

- User authentication
- Database management
- URL routing
- Session handling
- Admin management

Django's built-in security features protect against threats like SQL injection, XSS, and CSRF attacks. It also offers an automatic admin interface, making it easy to manage movies, reviews, and user data. Django serves as the backbone of the application's server-side logic.

3. React.js (Frontend Framework)

React.js is a popular JavaScript library developed by Facebook for building dynamic and interactive user interfaces. In the Movie Review System, React.js is used as the frontend layer to create a smooth, fast, and modern user experience.

Key features:

- Component-based architecture for reusable UI components
- Virtual DOM for high performance
- Fast rendering and smooth page updates without reloading

- Separation of concerns between UI and backend logic
- React makes it easier to build pages such as:
- Movie list display
- Movie details with live review updates
- User dashboard
- Interactive rating and review form

By integrating React with Django (via API endpoints), the system becomes more modular and scalable.

4. Tailwind CSS (Styling Framework)

Tailwind CSS is a highly customizable utility-first CSS framework used to style the React front end. It replaces traditional CSS and frameworks like Bootstrap with lightweight, utility-based classes.

Benefits:

- Faster UI development using pre-defined utility classes
- Consistent and responsive design across all devices
- Highly customizable without writing long CSS files
- Lightweight and optimized for speed
- Tailwind helps create a visually appealing, modern interface for movie cards, buttons, forms, navigation bars, and layout grids.

5. Database System (SQLite/MySQL)

The Movie Review System requires a reliable database to store user information, movies, ratings, and reviews.

SQLite (default Django DB): Suitable for small-medium projects; installed automatically.

MySQL: Recommended for large-scale deployment due to performance and scalability.

Both databases support structured storage and secure data retrieval using Django ORM.

6. Node.js and npm

React.js requires Node.js and npm (Node Package Manager).

Node.js enables:

- Running React development environment
- Managing React dependencies
- Building production-ready frontend files

npm helps install packages like:

- React Router

- Axios (for API calls)
- Tailwind CSS

Together, these tools enable modern frontend development.

7. IDE / Code Editors (VS Code Recommended)

A suitable code editor is needed for writing backend and frontend code.

VS Code is recommended because it supports both Python and JavaScript development.

It includes extensions for Django, React, Tailwind, Git, and debugging tools.

8. Browser Software

A modern web browser (Chrome, Firefox, Edge, Safari) is required to:

- Test the frontend
- Check responsiveness
- Debug using developer tools
- Run the React development server

4.2 Hardware Requirements

Efficient hardware ensures smooth running of the backend server, React development environment, and build tools.

1. RAM (Minimum 4 GB, Recommended 8 GB)

- React + Django development can be memory-intensive.
- 4 GB is the minimum requirement.
- 8 GB or higher is recommended for faster compilation, serving, and multitasking.

2. Processor (Intel i3 or Above)

A multi-core processor is required.

Intel i3 or AMD equivalent works fine for development.

Better processors significantly reduce build and compile times.

3. Storage (Minimum 500 MB, Recommended 2 GB)

Storage is needed for:

- Python & Django
- Node.js and React dependencies
- Project files
- Database files

- Tailwind CSS build files

React node modules alone can consume 200–300 MB.

Thus, 2 GB storage is recommended for smooth development.

4. Display and Input Devices

- A standard monitor, keyboard, and mouse are sufficient.
- A full HD screen is useful for viewing responsive UI layouts.
- Developers working on UI may benefit from a larger display.

5. Internet Connectivity

- A stable internet connection is required for:
- Installing npm packages
- Fetching Python dependencies
- Accessing documentation
- Deploying the application
- Using Git/GitHub for version control

Good connectivity ensures smooth development, debugging, and collaboration.

Summary of System Requirements

Requirement Type	Specification
Specification	Python 3.10+, Django 4.2+, SQLite/MySQL
Frontend Software	React.js, Tailwind CSS, Node.js, npm
Development Tools	VS Code / PyCharm
Hardware	4–8 GB RAM, Intel i3+, Minimum 500 MB–2 GB storage
Other	Modern browser, Stable internet connection

5. System Design

System Design represents the overall structure, workflow, and interaction of different components within the Movie Review System. It defines how data flows through the system, how users interact with the application, and how different modules communicate with each other. The Movie Review System uses the Django framework, which follows the Model-View-Template (MVT) architecture. This architecture ensures efficient separation of responsibilities and smooth execution of functions.

The design of this application focuses on providing a clear workflow for handling user actions such as browsing movies, reading reviews, posting ratings, and managing content through the admin panel. The system is built to be scalable, secure, and easy to maintain.

5.1 Model Design

The system consists of three primary models: User, Movie, and Review. Each model represents an essential entity stored in the database through Django ORM.

1. User Model

The User model stores information about registered users who interact with the platform. Users must create an account to submit reviews or ratings. The model typically includes attributes like:

- Username
- Email
- Password

Django's built-in authentication system is used to handle user login, logout, password hashing, and permission management. This ensures secure handling of user credentials and access control.

2. Movie Model

This model stores all movie-related information. Each movie has attributes such as:

- Title
- Release year
- Poster image
- Description

The Movie model acts as the central element of the system. Each movie has a unique ID and is associated with multiple user reviews. Admin users add movies to the database and can modify or delete them when needed.

3. Review Model

The Review model manages feedback posted by users. Each review is linked to both a user and a movie, using foreign keys. The essential fields include:

- Rating (usually 1–5 stars)
- Comment
- Review date

When a user posts a review, the system automatically updates the average rating for that movie. This creates a dynamic and collaborative evaluation mechanism.

5.2 System Flow

The Movie Review System follows a well-defined flow of operations that ensures smooth functioning and clear data interactions. The basic flow can be represented as:

User Request → Views → Models → Database → Output (Template)

1. User Request

Users initiate actions such as:

- Logging in or signing up
- Visiting the homepage
- Searching movies
- Clicking on a movie
- Posting a review

These actions generate requests that are processed by Django's view functions.

2. Views

The View layer acts as the "brain" of the application. It handles logic, retrieves data from the database, processes user inputs, and determines which webpage (template) should be displayed.

For example:

A user clicking a movie title triggers a view that retrieves movie details and all reviews from the database.

A user submitting a review triggers a view that saves the review record into the database.

Views also handle form validation, error handling, and session management.

3. Models

The View communicates with Models to perform data operations. Models represent structured database tables.

- Operations include:

- Fetching movie details
- Storing new reviews
- Updating user information
- Retrieving list of movies

Django ORM automatically converts Python objects into database queries.

4. Database Interaction

Once the model sends a query, the database executes operations like:

- Insert
- Update
- Delete
- Retrieve

The result is sent back through the model to the view.

5. Output (Template)

Templates represent the front-end pages designed using HTML, Tailwind CSS,React.js and Bootstrap. They display:

- Movie lists
- Movie details
- User reviews
- Average ratings
- Login and registration forms

The template receives data from the view and renders it in a user-friendly layout.

5.3 Admin Panel Design

The admin panel is an essential part of system design. It provides privileged access to administrators who maintain the system and ensure data accuracy.

Admin Responsibilities:

- Add new movies
- Update movie details
- Remove outdated or inappropriate content
- Monitor user reviews
- Remove abusive or misleading reviews

Django's built-in admin interface makes it easy to manage data without writing additional code. This panel ensures smooth control over content and maintains the quality of the review system.

5.4 User Interaction Design

The system is designed for simple and intuitive user interaction. A typical user journey includes:

1..Logging

2.Visiting the homepage

3. Browsing movies by category or search

4. Clicking a movie to view its details

5. Reading existing reviews

6. Submit a rating and comment

7. Reviewing their own contributions

The interface is responsive, ensuring usability across mobile devices, tablets, and desktops.

5.5 Data Flow and System Behavior

Data Flow Example (Posting a Review):

1. User logs in

2. User selects a movie and writes a review

3. The review form sends data to the view

4. View validates and saves data into the Review model

5. Model updates the database

6. Updated average rating is calculated

7. Template displays the refreshed movie page with new review and rating

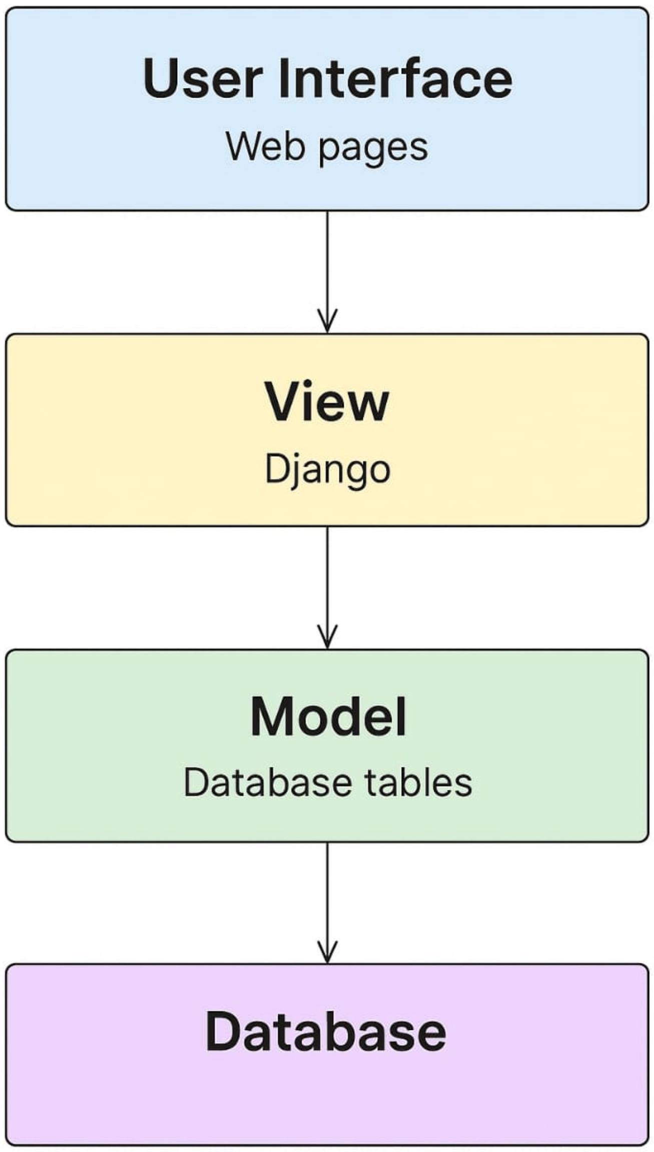
This cycle ensures real-time updates and a dynamic user experience.

5.6 Summary of System Design

- The Movie Review System's design ensures:
- A clean division of responsibilities through MVT
- Secure user authentication and controlled data access
- Efficient handling of reviews and movie information
- Simple navigation and user-friendly interface

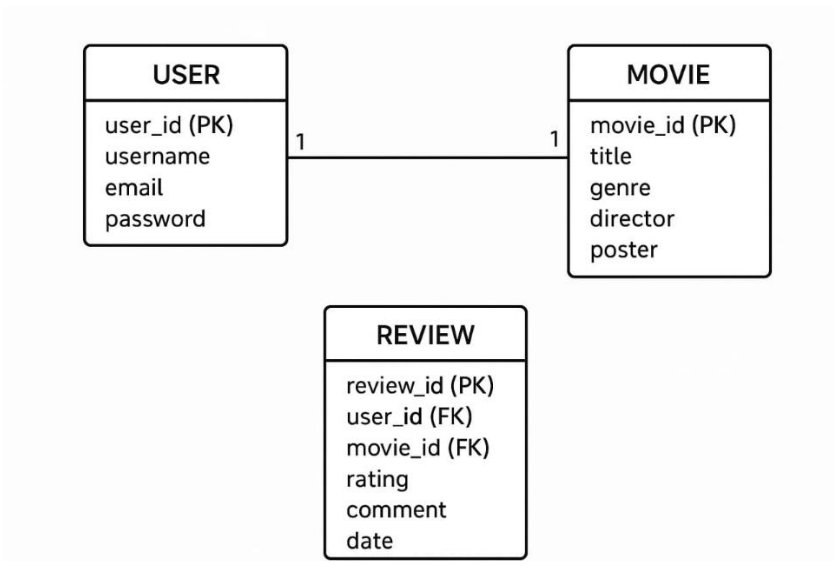
- Reliable admin control for high data quality

Together, the Models, Views, Templates, and Database form a cohesive system that supports all functionalities—browsing movies, reading reviews, posting feedback, and managing content.



6.Database Design

The Movie Review System uses three main database tables: User, Movie, and Review. The User table stores details of registered users, including user_id, username, email, and password. The Movie table contains all movie-related information such as movie_id, title, genre, director, and poster. The Review table links users and movies through foreign keys user_id and movie_id, and stores rating, comment, and review date. The relationship between User and Review is one-to-many, and between Movie and Review is also one-to-many, allowing each user to post multiple reviews and each movie to have multiple reviews.



6.1 ER Diagram

The Movie Review System database is designed to manage movies, users, reviews, and comments in a structured and relational manner. The Entity-Relationship (ER) diagram illustrates how data is organized and how different entities are connected through primary and foreign keys. This ensures data integrity, efficient retrieval, and smooth platform operations.

1. User Entity

The User table stores information about all registered users on the platform.

Attributes include:

- **User_id (PK):** A unique identifier for each user.
- **Email:** Email address used for login.
- **Password**: Encrypted password for account security.
- **Role:** Defines whether the user is an admin, critic, or general user.

Users are essential because they create reviews and post comments. The relationships in the diagram show that a user can write multiple reviews and can also comment on multiple reviews.

2. Movie Entity

The Movie table stores details about all movies available for review.

Attributes include:

- **Movie_id (PK):** Unique ID for each movie.
- **Title:** Name of the movie.
- **Genre:** Type/category of the movie (e.g., action, drama).
- **Director:** Name of the movie's director.

Each movie can receive many reviews from different users. The diagram's connection shows a one-to-many relationship between Movie and Review.

3. Review Entity

The Review table stores user-generated reviews for movies.

Attributes include:

- **Review_id (PK):** Unique ID for each review.
- **Movie_id (FK):** Links the review to the movie being reviewed.
- **User_id (FK):** Links the review to the user who wrote it.
- **Rating:** Rating or text providing the user's opinion.

A single user can post many reviews, but each review is associated with only one specific movie and one specific user. The Review entity therefore acts as a bridge between User and Movie.

4. Comment Entity

The Comment table allows users to comment on existing reviews, enabling user interaction.

Attributes include:

- **Comment_id (PK):** Unique ID for each comment.
- **Review_id (FK):** The review on which the comment is posted.
- **User_id (FK):** The user who wrote the comment.
- **Comment**: The content/text of the comment.

Each review can have multiple comments, and each user can post multiple comments.

Overall Relationships:

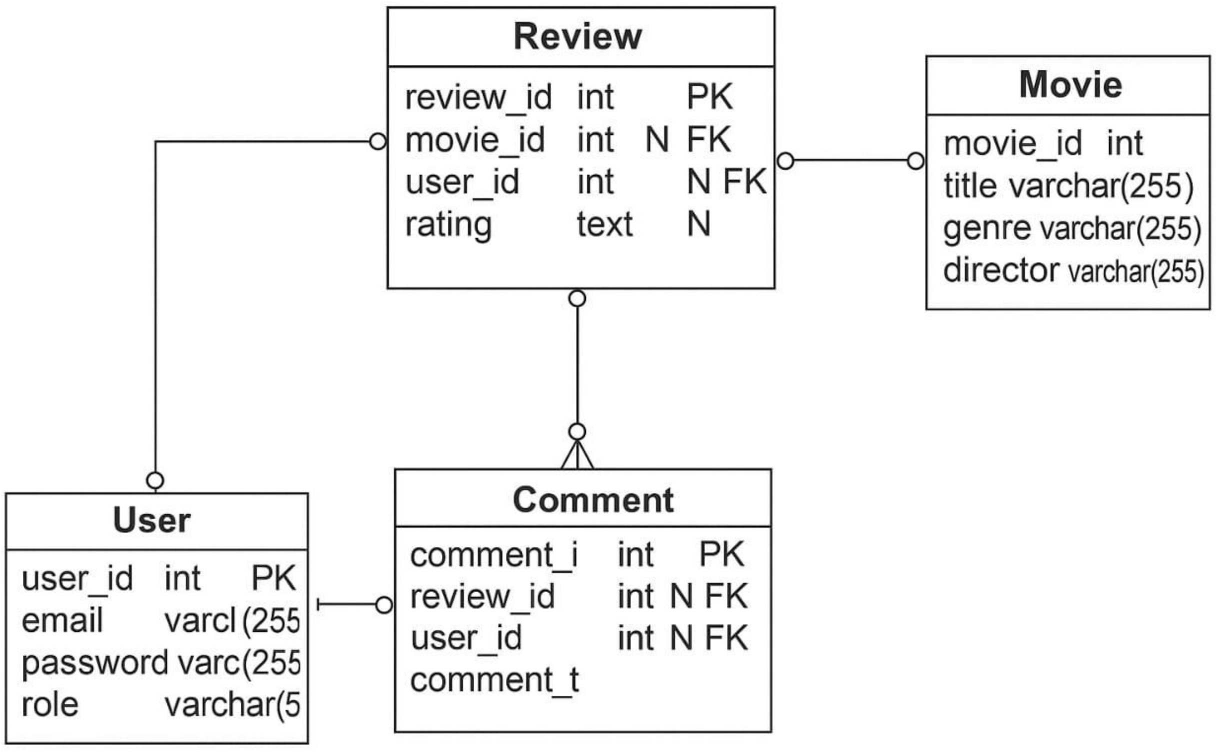
User → Review (1-to-many): Users can write multiple reviews.

Movie → Review (1-to-many): Each movie can have many reviews.

Review → Comment (1-to-many): Reviews can receive multiple comments.

User → Comment (1-to-many): Users can comment multiple times.

These relationships ensure proper linkage between all activities on the platform.



7. Implementation

The implementation of the Movie Review System involves developing the core functionalities using Django for the backend and integrating a responsive interface through Bootstrap templates. The system is built following Django's Model-View-Template (MVT) architecture, which ensures clean separation of logic, data, and presentation layers.

7.1 Django ORM for Database Operations

The system uses Django's powerful Object-Relational Mapping (ORM) to handle all interactions with the database. Instead of writing raw SQL queries, Django ORM allows developers to interact with tables using Python objects.

Key operations performed using ORM include:

- Creating new records, such as adding movies or storing reviews
- Retrieving data, such as listing movies or fetching user-specific reviews
- Updating records, for example, modifying movie information in the admin panel
- Deleting records, such as removing inappropriate reviews
- The ORM ensures:
- Data consistency
- Better security
- Faster development
- Easy integration with multiple database systems (SQLite/MySQL)
- Using ORM helps maintain clean, readable code and makes the system scalable.

7.2 User Authentication for Login and Signup

Authentication is implemented using Django's built-in authentication system. It handles all essential features like:

- User registration (signup)
- Login and logout
- Password hashing and secure storage
- Session management
- Permission control

Only authenticated users can write reviews and ratings, ensuring accountability and preventing spam. The use of Django Auth provides high security through features like encrypted passwords, CSRF protection, and session-based authentication.

Through these mechanisms, the system ensures a secure environment where users can safely interact with the platform.

7.3 Template Design Using Bootstrap

The front-end of the application uses Django Templates combined with Bootstrap for styling and layout. Bootstrap is chosen because it ensures:

- A clean and modern interface
- Fully responsive design across mobile, tablet, and desktop
- Pre-built components such as navigation bars, cards, forms, buttons, and grids
- Faster front-end development

Each page, including the homepage, movie listings, review forms, and user profile sections, is designed using Bootstrap classes.

This results in:

- Uniform design
- Easy navigation
- Better user experience (UI/UX)

Templates dynamically display movie details and user reviews passed from views, making the interface interactive and user-friendly.

7.4 Dynamic Calculation of Average Rating

One of the important features of the Movie Review System is the dynamic average rating calculation.

Every time a user posts a review with a rating, the system recalculates the average rating for that particular movie.

Process:

1. User submits a review with a star rating (e.g., 4 stars).

2. Review is saved through Django ORM.

3. All ratings for that movie are fetched.

4. Average is recalculated using an aggregation function.

5. The new average is displayed immediately on the movie page.

This ensures that:

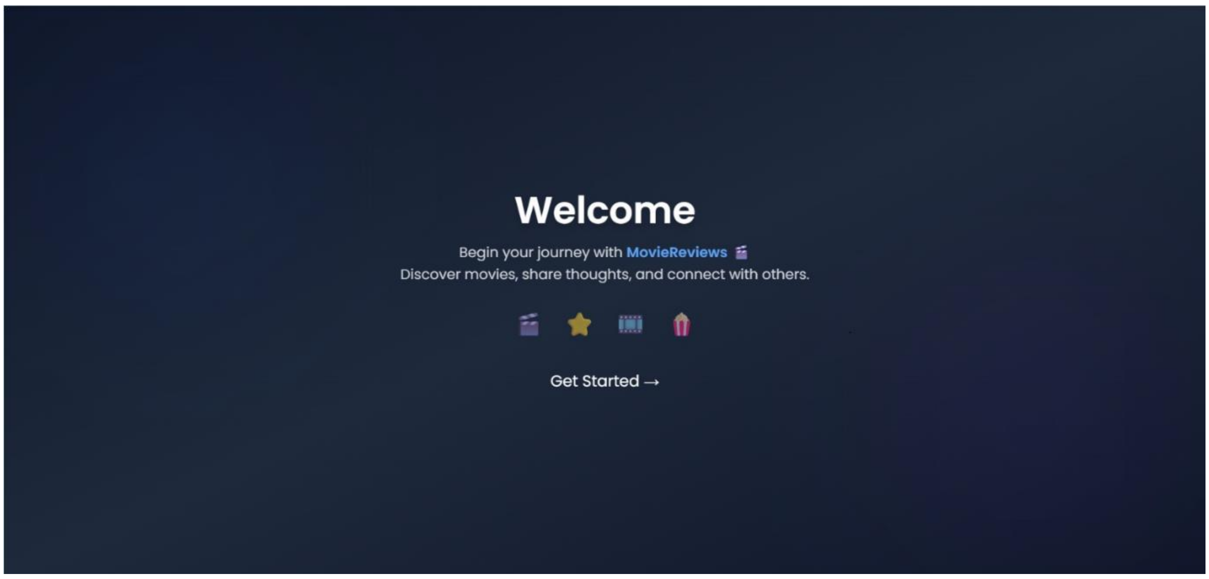
- Ratings are always up-to-date
- No manual calculations are needed
- The displayed rating reflects collective audience opinion

This dynamic approach helps users make better decisions before watching a movie.

Summary

The implementation integrates the backend power of Django with a responsive and visually clean interface using Bootstrap. Django ORM ensures secure and efficient data handling, while authentication provides safe user access. The dynamic rating calculation makes the system interactive and reliable. Together, these components create a complete and functional movie review platform that is easy to use, scalable, and effective.

8.Screenshots:

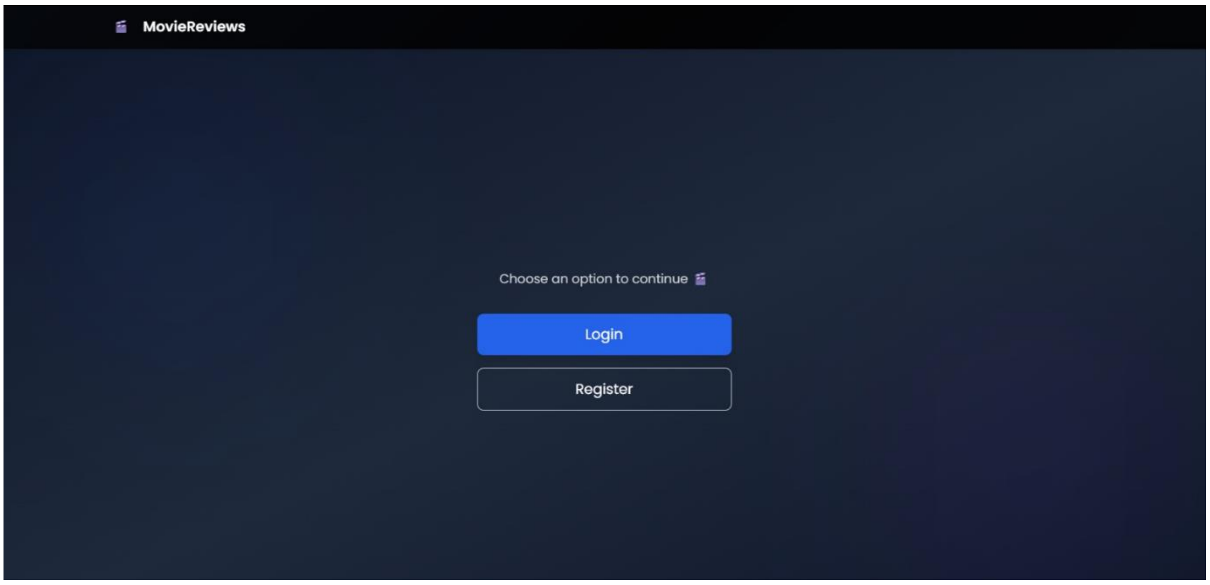

Welcome

Begin your journey with MovieReviews 🎬

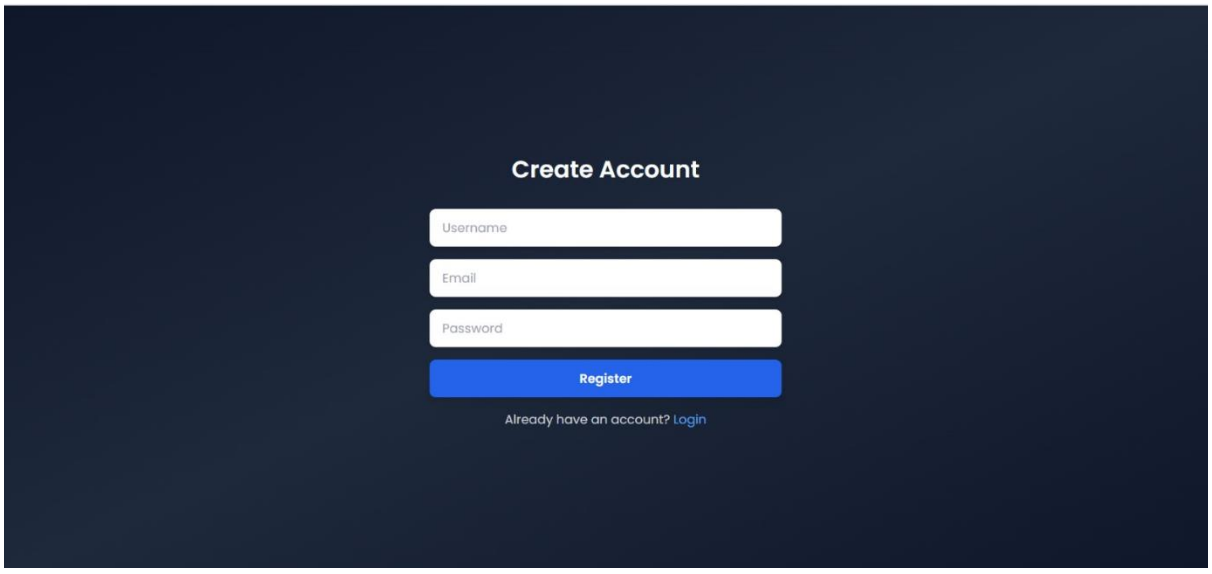
Discover movies, share thoughts, and connect with others.

🎬 ⭐ 🎞️ 🍿

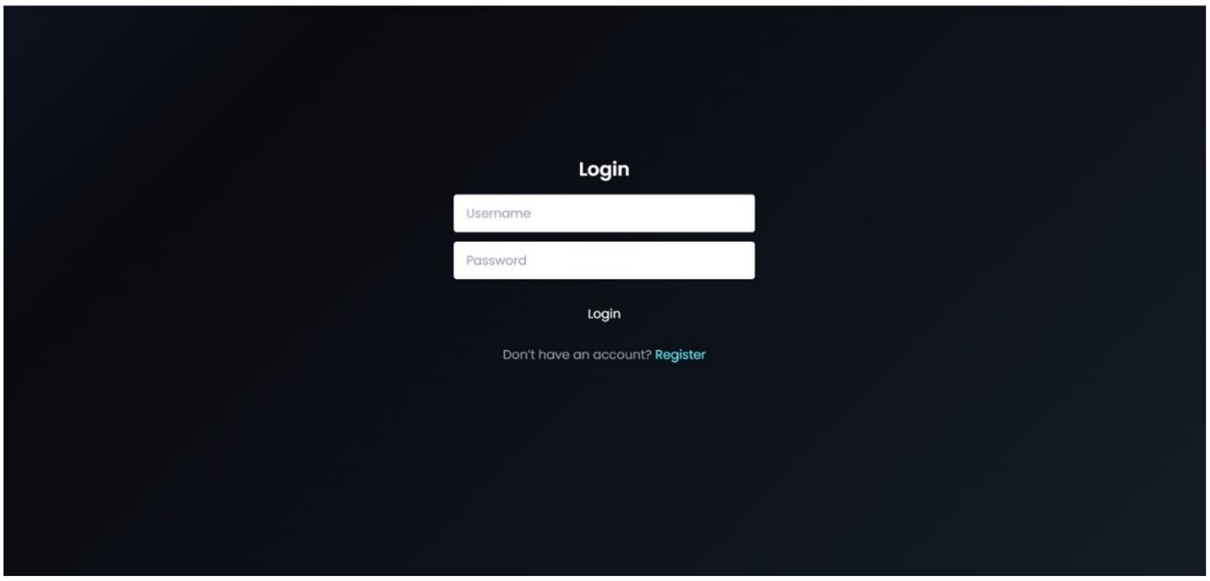
Get Started →



Choose an option to continue 🎬


Create Account

Already have an account? Login

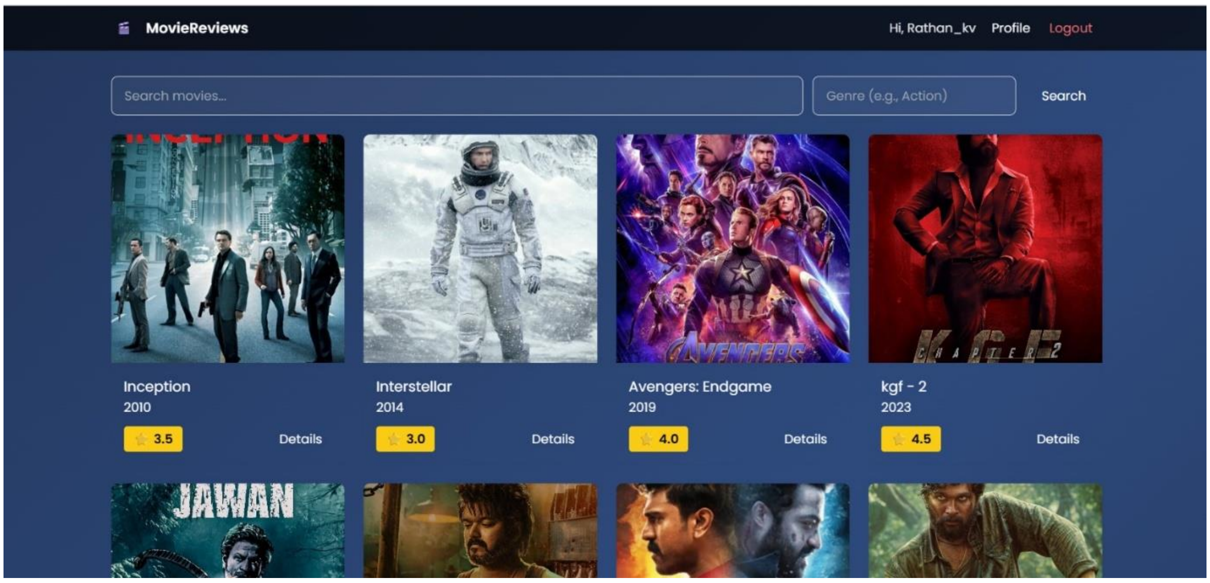

Login

Username

Password

Login

Don't have an account? Register


MovieReviews

Hi, Rathan_kv Profile Logout

Search movies...

Genre (e.g., Action)

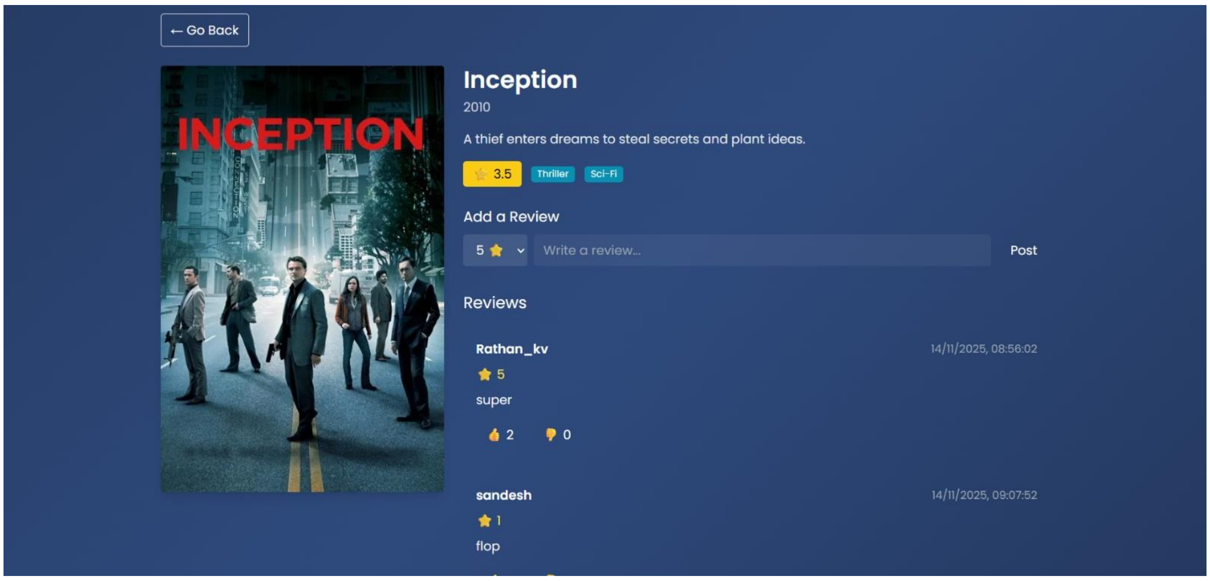
Search

Inception
2010
3.5
Details

Interstellar
2014
3.0
Details

Avengers: Endgame
2019
4.0
Details

kgf - 2
2023
4.5
Details


← Go Back

Inception
2010

A thief enters dreams to steal secrets and plant ideas.

3.5 Thriller Sci-Fi

Add a Review

5 ⭐

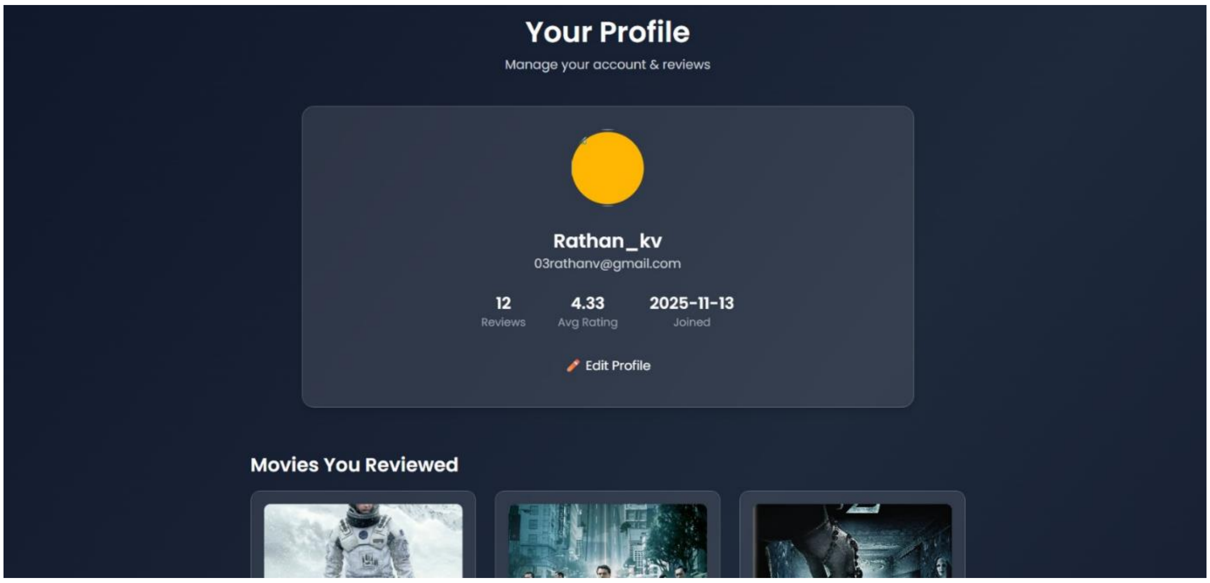
Write a review...

Post

Reviews

Rathan_kv 14/11/2025, 08:56:02
⭐ 5
super
👍 2 👎 0

sandesh 14/11/2025, 09:07:52
⭐ 1
flop


Your Profile

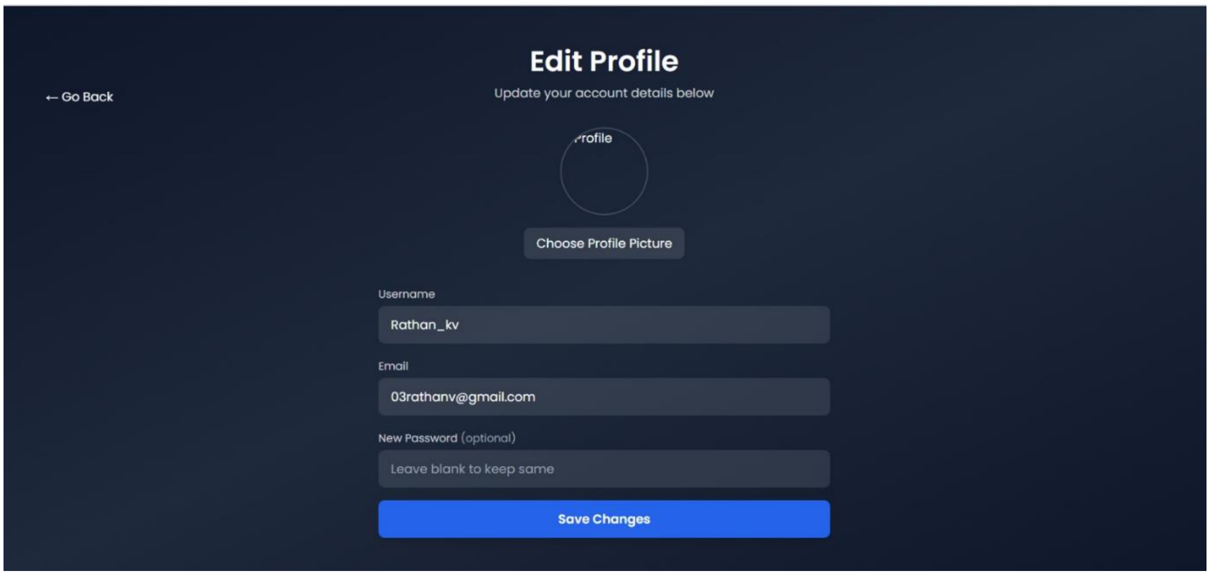
Manage your account & reviews

Rathan_kv

03rathanv@gmail.com

12	4.33	2025-11-13
Reviews	Avg Rating	Joined

Movies You Reviewed



Edit Profile

Update your account details below

Username: Rathan_kv

Email: 03rathanv@gmail.com

New Password (optional): Leave blank to keep same

9. Testing

Testing is one of the most crucial phases of software development. It ensures that the system behaves as expected, performs reliably under different conditions, and provides a smooth user experience. For the Movie Review System, multiple components such as authentication, review submission, form validation, user interface, and admin functionalities were tested thoroughly. The testing process aimed to identify functional errors, usability issues, and data-handling problems before deployment.

The system was tested using a combination of manual testing, functional testing, validation testing, and black-box testing techniques. Each module was evaluated separately and also tested with integrated components to ensure the entire application worked seamlessly.

9.1 Login and Authentication Testing

Authentication is an essential part of the movie review system because only registered users can log in and post reviews. The login and signup modules were tested for:

a) Valid Login Credentials

- Entered correct username and password.
- Ensured successful login and redirection to the homepage.

b)Invalid Login Credentials

- Wrong username or password generated appropriate error messages.
- System prevented unauthorized access.

c)Signup Validation

- Verified that the system required correct details such as unique username, valid email, and strong password.
- Checked password hashing to ensure secure storage.

d) Session Management

- After login, user session remained active until logout.
- Logging out cleared the session and prevented access to restricted pages.

e)Security Testing

- CSRF protection was validated using forms.
- Ensured that restricted pages (like posting reviews) could not be accessed without authentication.
- Overall, authentication testing confirmed that the system securely handles user identity, prevents unauthorized access, and manages sessions effectively.

9.2 Review Submission Testing

Review submission is a core functionality of the platform. It allows users to rate and comment on movies.

a)Valid Review Submission

- User can add a rating (1–5 stars) and write a comment.
- Review gets saved in the database through Django ORM.
- Page refresh displays the new review instantly.

b)Invalid Review Submission

- Empty rating field triggers validation error.
- Blank comment fields are restricted based on validation settings.
- No review is saved unless required fields are entered correctly.

c) Multiple Reviews

Tested that a user can submit only one review per movie or multiple reviews (depending on system rules).

d)Average Rating Update

- After a review is posted, the dynamic average rating updated correctly.
- Tested with various combinations of ratings to ensure correct aggregation.

e) Review Fetching

- Confirmed that all reviews for a movie are displayed in chronological order.
- Ensured that only approved or valid reviews appear (admin moderation).

Review testing ensured that the system accurately handles user-generated content and maintains data consistency.

9.3 Form Validation Testing

Forms are used throughout the system—for login, signup, review submission, and admin content entry. Correct form validation guarantees that only clean, accurate, and safe data enters the system.

a) Field-Level Validation

Email format validation

Required fields cannot be left blank

Password length and strength checks

Integer validation for rating values

b) Error Message Testing

- Error messages were displayed clearly and accurately.

- Error states were tested for usability, ensuring users understood what corrections were needed.

c)Client-Side Validation

- HTML5 validations such as required fields, input types, and length constraints were tested.

d)Server-Side Validation

- Django backend ensured correct data storage even when client-side validation was bypassed.

Form validation testing confirmed the robustness of the system against invalid or malicious inputs.

9.4 Admin CRUD Operations Testing

The admin panel is responsible for the management of movies, users, and reviews. CRUD (Create, Read, Update, Delete) testing was performed to verify administrative features.

a) Create (Add Movie)

- Admin successfully adds a new movie with all details like title, genre, poster, description, etc.
- Database updates immediately and reflects on the user interface.

b) Read

- Admin can view all movies, users, and reviews.
- Pagination and filtering features were tested for efficiency.

c) Update

- Admin can edit movie details (title, poster, genre, etc.).
- Changes are reflected on user pages without errors.

d) Delete

Admin can delete movies or inappropriate reviews.

Tested deletion for:

- Movies with reviews
- Movies without reviews
- Reviews posted by users

Ensured deletion does not break database integrity.

e) Review Moderation

- Admin could remove spam or inappropriate comments.
- Ensured removed reviews no longer appear on the movie page.

Admin CRUD testing confirmed that the system offers strong management capabilities and maintains clean database integrity.

9.5 User Interface and Navigation Testing

The UI was tested to ensure that users can navigate the system easily and intuitively.

a) Homepage Navigation

- Movie listings displayed correctly.
- Search and filter tested for accurate results.

b) Movie Details Page

- Correct display of movie poster, title, genre, and reviews.
- Review form appears only for logged-in users.

c) Responsiveness Testing

Verified UI on different screen sizes:

- Mobile phones
- Tablets
- Laptops
- Desktops

d) Template Rendering

- Django templates rendered dynamic data properly.
- No broken links, missing images, or misaligned components.

9.6 Database and ORM Testing

- Django ORM was tested for data accuracy and relationship handling.

a) Foreign Key Constraints

- Reviews linked correctly to users and movies.
- No orphan records.

b) Data Retrieval

Movie list loads efficiently.

Review data loads quickly and accurately.

10.Results

The Movie Review System was successfully developed and implemented using Django for the backend and Bootstrap for the frontend, resulting in a functional, user-friendly, and efficient web application. The testing and integration phases demonstrate that the system meets all its objectives and provides a reliable platform for users to explore movies, read reviews, and contribute their own feedback. The following section summarises the key outcomes and results achieved during the development of the system.

1. Successful User Registration and Secure Authentication

One of the core results of the system is the implementation of a secure and effective authentication mechanism. Users can create accounts, log in using valid credentials, and manage their sessions without encountering errors or security risks. The system encrypts passwords and follows security best practices, ensuring the protection of user data. Session management works smoothly, preventing unauthorized access and maintaining privacy. This result ensures that the platform is safe and trustworthy for users who post reviews and engage with the content.

2. Efficient Movie Browsing and Information Display

The system provides a well-organized interface where users can browse movies easily. Movie details such as title, genre, director, and poster are displayed clearly, helping users explore the available films. The structured layout enables users to quickly find movies of interest, either through browsing or searching. This result enhances the user experience by offering smooth navigation and intuitive interaction with the system.

3. Smooth and Accurate Review Submission

A major functionality of the Movie Review System is allowing users to post reviews and ratings. The system supports this feature effectively by validating user input and storing the reviews accurately in the database. Once a review is submitted, it appears immediately on the movie details page, giving users instant feedback. The system ensures that each review is linked to the correct user and movie, maintaining consistency and integrity of data. This result confirms that the review module works reliably and contributes to the interactive nature of the website.

4. Dynamic Average Rating Calculation

The platform calculates and updates the average rating for each movie based on all submitted reviews. The dynamic calculation ensures that users always see the most up-to-date and accurate rating. This feature provides a fair representation of audience opinions and helps users decide whether a movie is worth watching. The calculation has been tested thoroughly and produces correct results across different scenarios. This result demonstrates the system’s ability to handle real-time data processing.

5. Effective Admin Management for Movies and Reviews

The admin module functions successfully and plays a key role in maintaining the quality and consistency of the system. Administrators can add new movies, edit existing movie data, and delete movies that are outdated or irrelevant. They can also review user comments and remove inappropriate or spam reviews, ensuring the platform remains clean and reliable. Successful CRUD operations confirm that the admin interface is easy to use and improves the overall accuracy and organization of data. This result highlights the robustness of the management features in the system.

6.Reliable Database Performance and Integrity

Using Django ORM, the system interacts with the database smoothly and consistently. All data—user information, movie details, and reviews—is stored, retrieved, updated, and displayed correctly. No orphan records or duplicate entries were found during testing. Foreign key relationships between the User, Movie, and Review tables function correctly, protecting data integrity. This result shows that the database design is efficient and supports the core operations of the system.

7.User-Friendly and Responsive Interface

The combination of Django templates and Bootstrap has produced a clean, responsive, and visually appealing user interface. The system adapts well to different screen sizes, including desktops, laptops, tablets, and mobile devices. Users can access and navigate the platform comfortably, with clear menus, intuitive buttons, and smooth transitions. This result improves accessibility and ensures that users of all types can interact with the system without difficulty.

11. Conclusion

The Movie Review System successfully meets its objective of providing a structured, reliable, and user-friendly platform for browsing and submitting movie reviews. In an online environment filled with unverified and biased opinions, this system brings clarity by offering authenticated user reviews and dynamically calculated ratings. Through secure login features, users can confidently participate by posting their honest feedback, while the admin panel ensures proper management of movies and moderation of reviews. The system’s clean interface, responsive design, and organized content presentation make it easy for users to navigate, explore movies, and make informed viewing decisions.

The use of Django for the backend allows seamless data handling, secure authentication, and fast development, while Bootstrap ensures an appealing and mobile-friendly interface. The database design maintains strong data integrity with well-defined relationships between users, movies, and reviews. Overall, the project demonstrates how a simple yet effective web application can enhance user engagement, reduce misinformation, and create a dependable source of movie opinions. The Movie Review System stands as a functional foundation that can be improved and expanded in the future as user needs evolve.

12. Future Enhancements

Although the Movie Review System meets the core requirements, several enhancements can significantly improve user experience and system performance in future versions:

12.1. Recommendation System

A personalized recommendation engine can suggest movies to users based on their ratings, review history, and genre preferences. This would make the platform more interactive and tailored to each individual user.

12.2 Fake-Review Detection

An automated system using machine learning could be integrated to identify and filter out spam, duplicated, or fraudulent reviews. This ensures authenticity and improves trust in the platform.

12.3 Mobile App Version

Developing a mobile application for Android and iOS would enhance accessibility and attract more users. A dedicated app would offer faster access, push notifications, and improved user engagement.

12.4 Social Media Login

Integrating login options like Google, Facebook, or GitHub would simplify the registration process and encourage more users to sign up quickly without creating new accounts.

These enhancements would significantly boost the platform's capabilities and make it more modern, reliable, and user-centered.

13. References

Below is a detailed and properly structured references section. These sources cover Django, Bootstrap, frontend technologies, and general web development concepts used in the project.

Books and Documentation

1. Django Software Foundation. Django Documentation. https://docs.djangoproject.com

2. Bootstrap Team. Bootstrap Official Documentation. https://getbootstrap.com/docs

3. Mozilla Developer Network (MDN). HTML, CSS, and JavaScript Guides. https://developer.mozilla.org

4. W3Schools. Web Technologies Tutorials. https://www.w3schools.com

5. Python Software Foundation. Python Official Documentation. https://docs.python.org