Audiobook Web Application Documentation

Table of Contents

1. Introduction

This web application allows users to browse audiobooks, view details, and submit reviews and ratings. It is built using the **MERN stack (MongoDB, Express.js, React.js, and Node.js).**

Key Features

- Browse audiobooks
- Filter and sort audiobooks
- View audiobook details
- Submit reviews and ratings
- **User-friendly interface with keyboard shortcuts**
- **Save the heard books after rating**

2. System Architecture

The application follows a client-server architecture:
- Front-end: React.js
- Back-end: Node.js with Express.js
- Database: MongoDB

3. Front-end Documentation

Technologies Used

- React.js

Components

- Navbar: Functionality to search audiobooks by author and sort by genre and rating
- AudioBookList: The list of all audiobooks which can be sorted.
- AudioBookDetails: Can act as both your rated books and also the UI of single audio book
- StarRating: Useable StarRating section which can be used anywhere with variable no of stars

User Interface

The UI is divided into two main sections:
1. Left Section: Displays a list of all audiobooks
2. Right Section: Shows "Your Heard Books" (books you've rated) when no book is selected, or displays specific book details when a book is clicked

Key Features
- Search functionality by author
- Filtering by rating, and genre
- Responsive design
- Title change on book selection
- Keyboard shortcuts:
  - ESCAPE: Go back to the main view
  - ENTER: Focus on the search bar to search for authors

4. Back-end Documentation

Technologies Used
- Node.js
- Express.js
- MongoDB

Project Structure
The back-end uses a modular structure with three main components:
- Models: Define MongoDB schemas (in `models/`)
- Routes: Specify API endpoints (in `routes/`)
- Controllers: Contain request handling logic (in `controllers/`)

This structure follows the MVC pattern for improved code organisation.

Middleware
Key middleware used:
- cors: Enables Cross-Origin Resource Sharing
- dotenv: Loads environment variables
- mongoose: MongoDB object modelling

5. API Endpoints

Book Routes
Base URL: `localhost:8080/book`

METHOD	ENDPOINT	DESCRIPTION
POST	/	Create a new book
POST	/review	Create a review for a book

DELETE	/:id	Delete a book by ID
GET	/:id	Get a single book by ID
GET	/	Get all books

Filter Routes
Base URL: `localhost:8080/filter`

METHOD	ENDPOINT	DESCRIPTION
GET	/author/:name	Get books by author name
GET	/genre/:genre	Get books by genre
GET	/orderBy	Get books ordered ascending or descending

6. User Guide

Browsing Audiobooks
1. Open the application
2. Browse the list of audiobooks on the left side of the screen
3. Use the search bar to find specific authors
4. Apply filters to sort by genre or rating
5. View all the rated books and summary details when no book is selected

Viewing Book Details
1. Click on a book from the list
2. View detailed information on the right side of the screen
3. Press ESCAPE to return to the main view where you can see summary page

Submitting a Review
1. Select a book
2. Scroll to the review section
3. Enter your rating and review
4. Submit the review
5. The rated book added to you watched list

7. Deployment

Hosting

- Front-end: Github
- Back-end: Github
- Database: MongoDB Atlas

Environment Variables

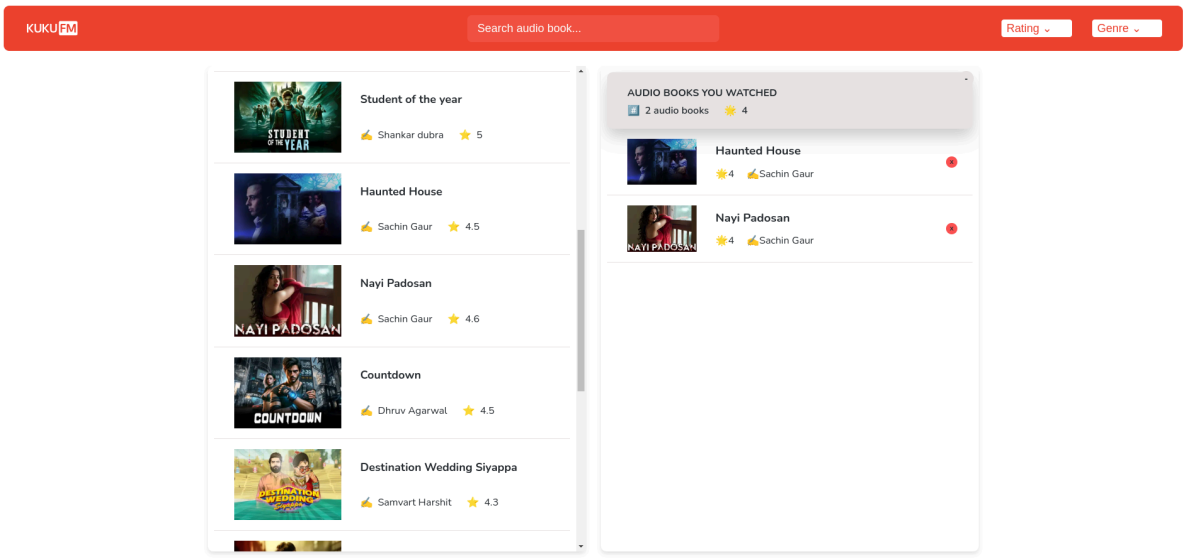
Ensure the following environment variables are set:

- `MONGODB_URL`: Connection string for your MongoDB database
- `PORT`: Port number for the server 8080 (check whether this port is busy or not lsof -i :8080 and then kill -9 <PID>)

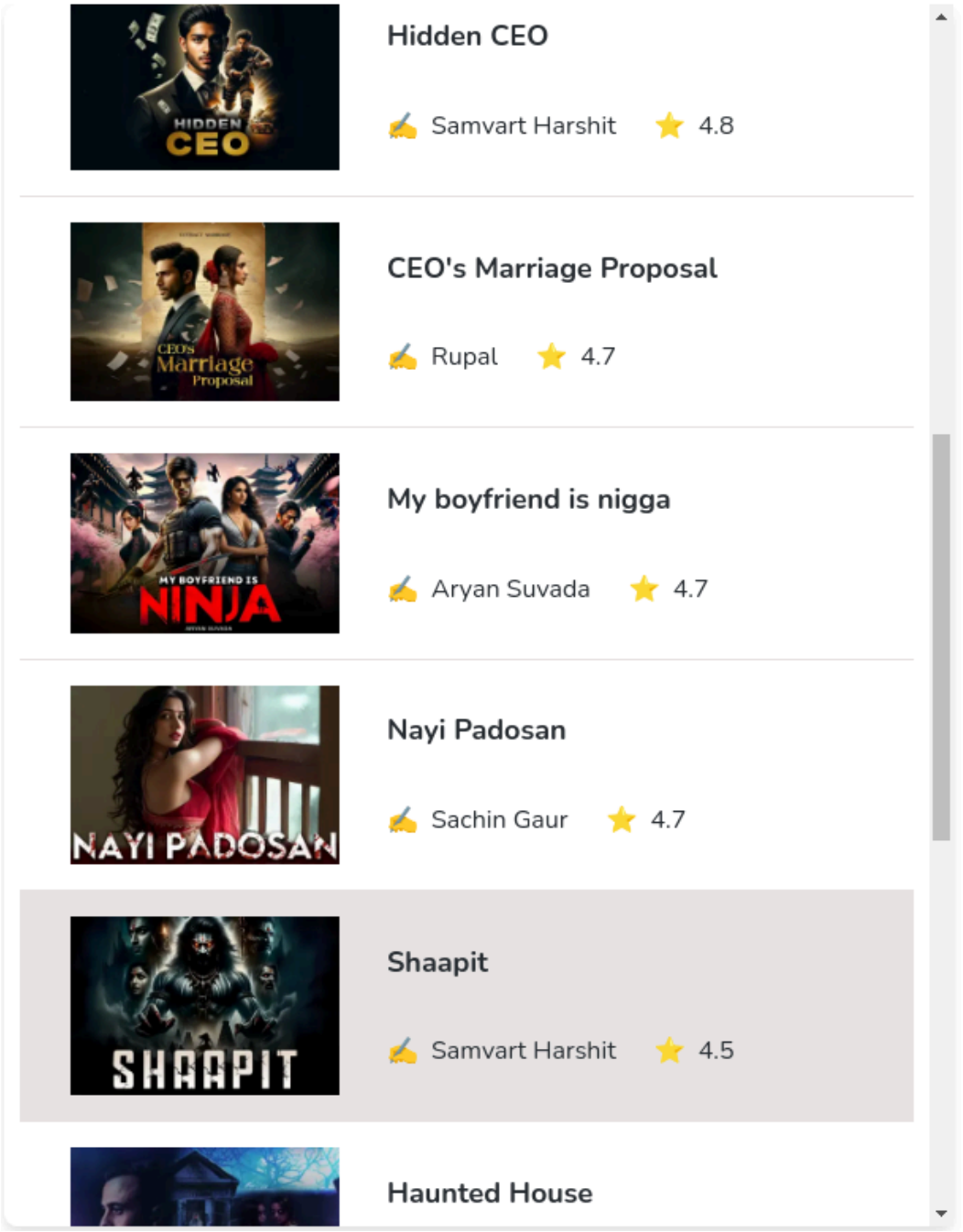
Deployment Steps

1. Clone github repo
2. **cd client** and **npm run dev** (to setup the Frontend) {http://localhost:5173/}
3. **cd api**
    a. **nodemon index.js –import** (this will import the data in your mongo database from books.json)
    b. **nodemon index.js –delete** (this will delete all the data)
    c. **Crtl + C** (Important)
    d. Now as the data is setup we can run **npm start** to start server

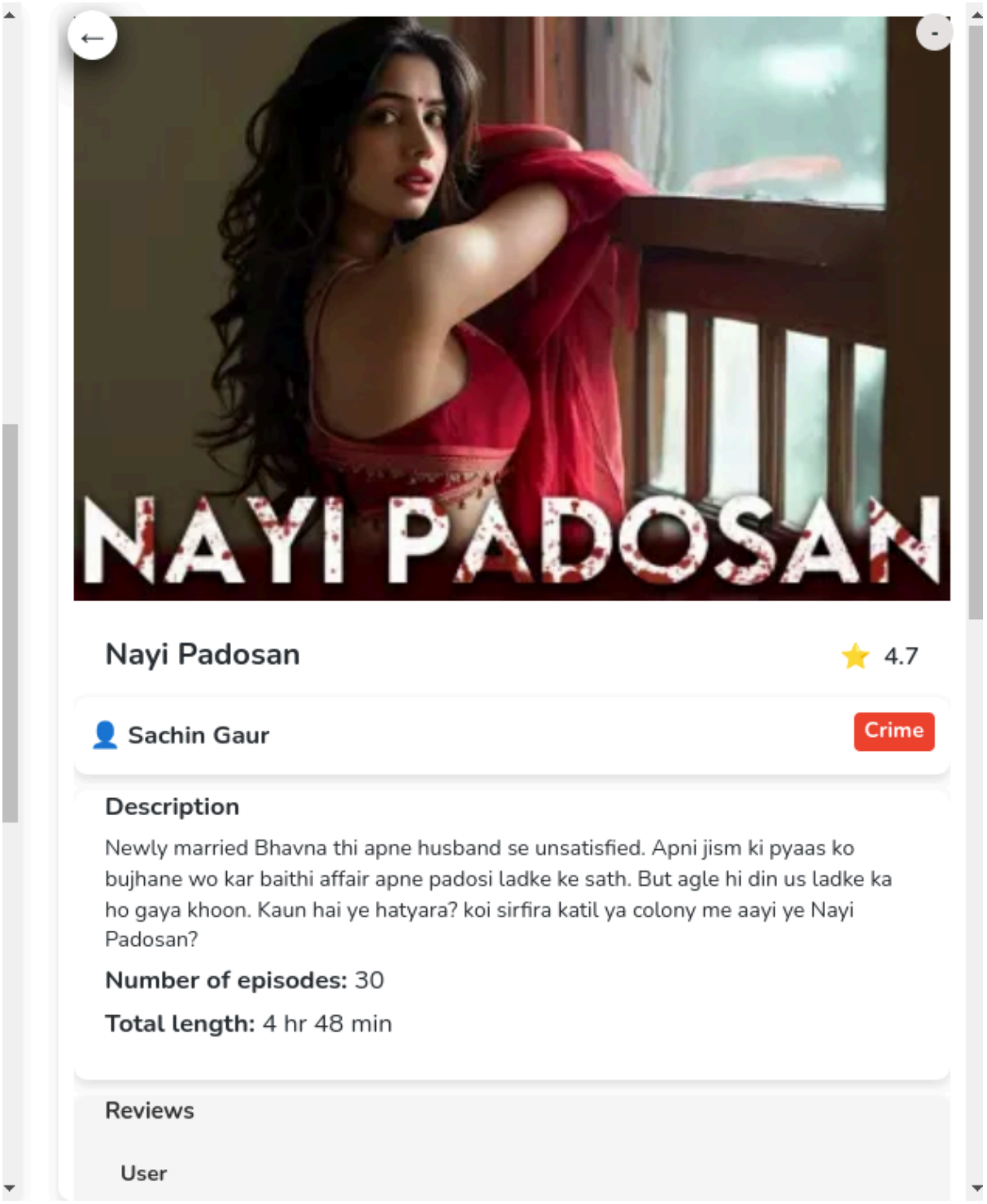
8. Pictures



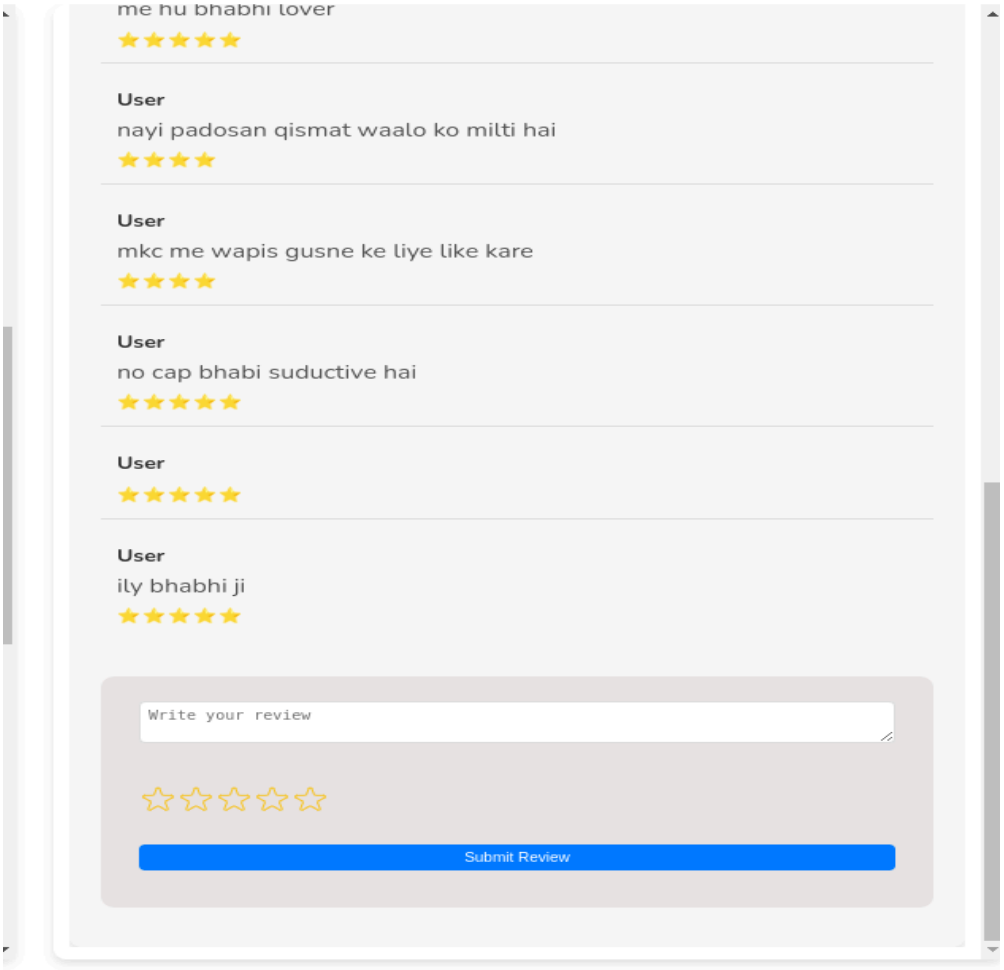
Home page where the summary is fetch from localStorage of browser and the book list is fetch by the backend



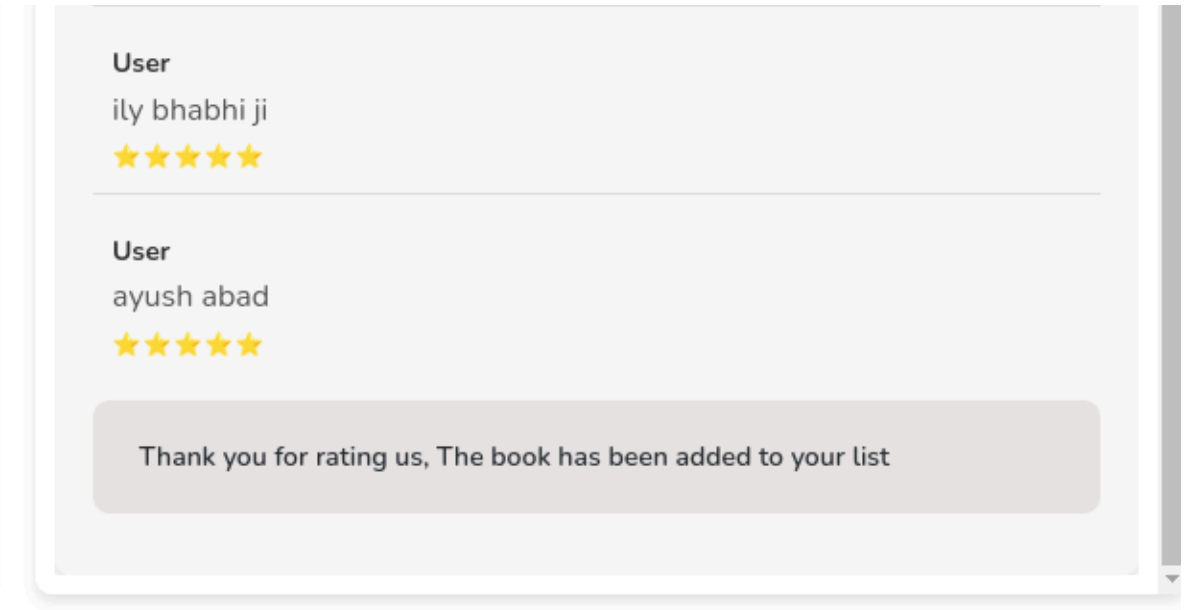
Left side of the page where the book list renders



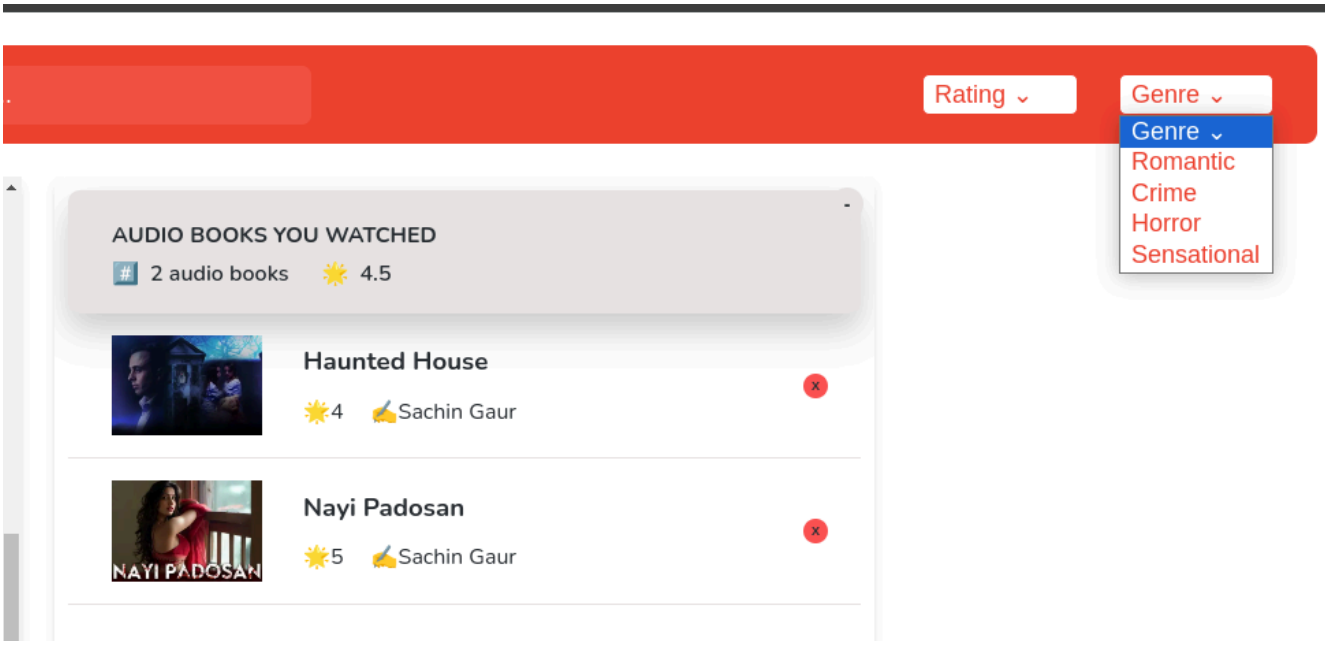
Right side of page where the Single book renders GET(/book/:id)



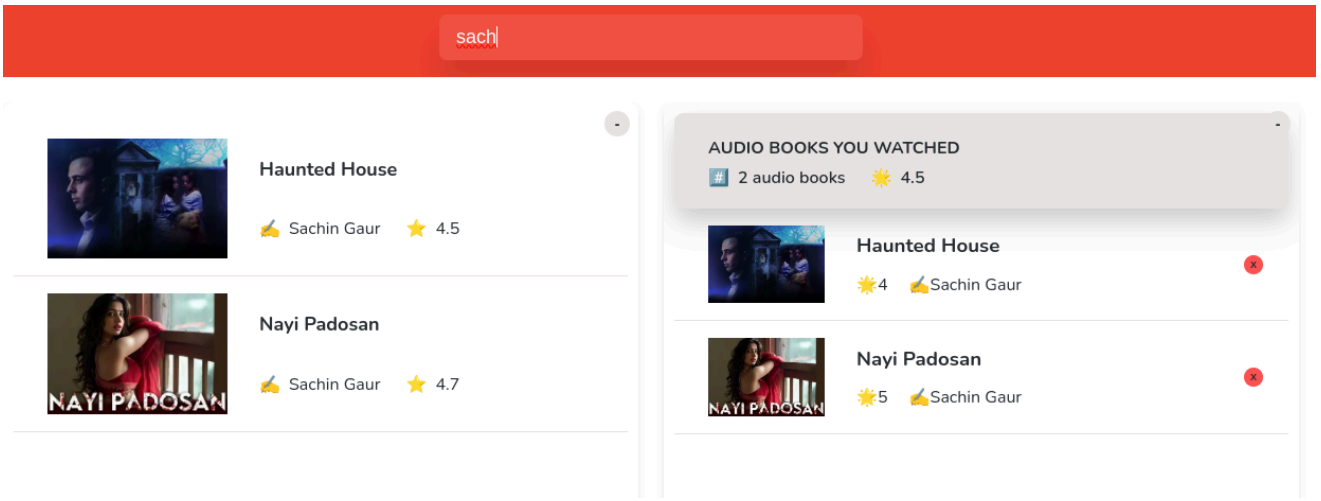
Here is the review which is updated in real time and the **avgRating is calculated based on the rating and updates on each rating changed**



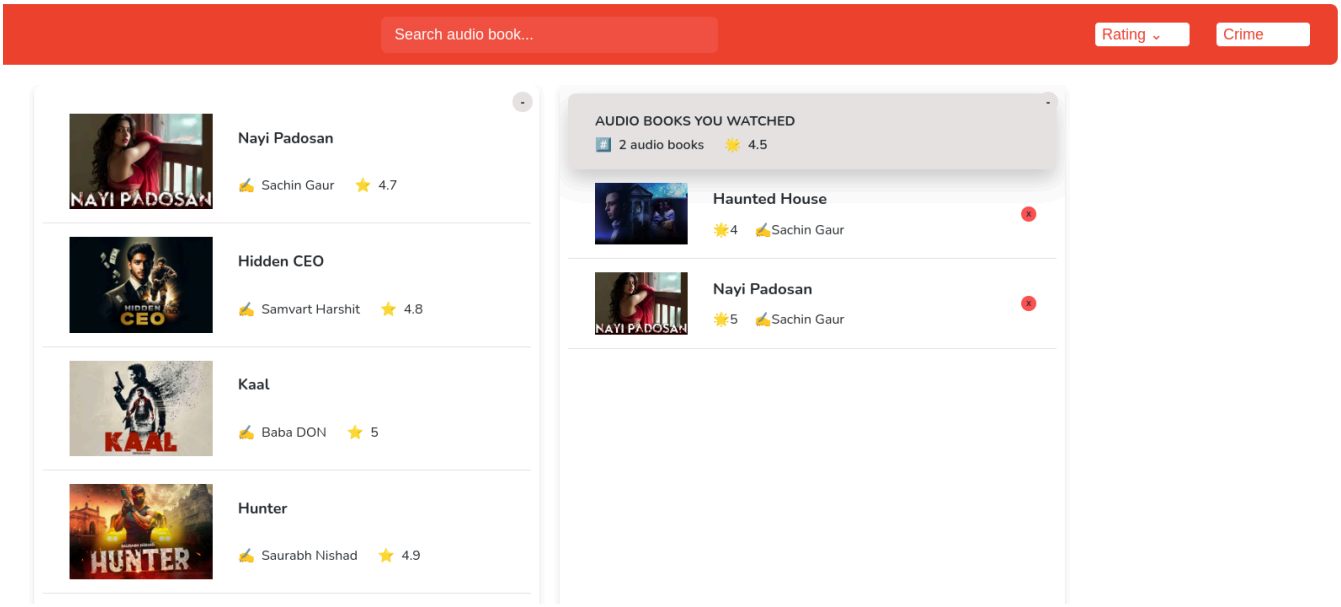
As the review is submitted the movie is added to watched list



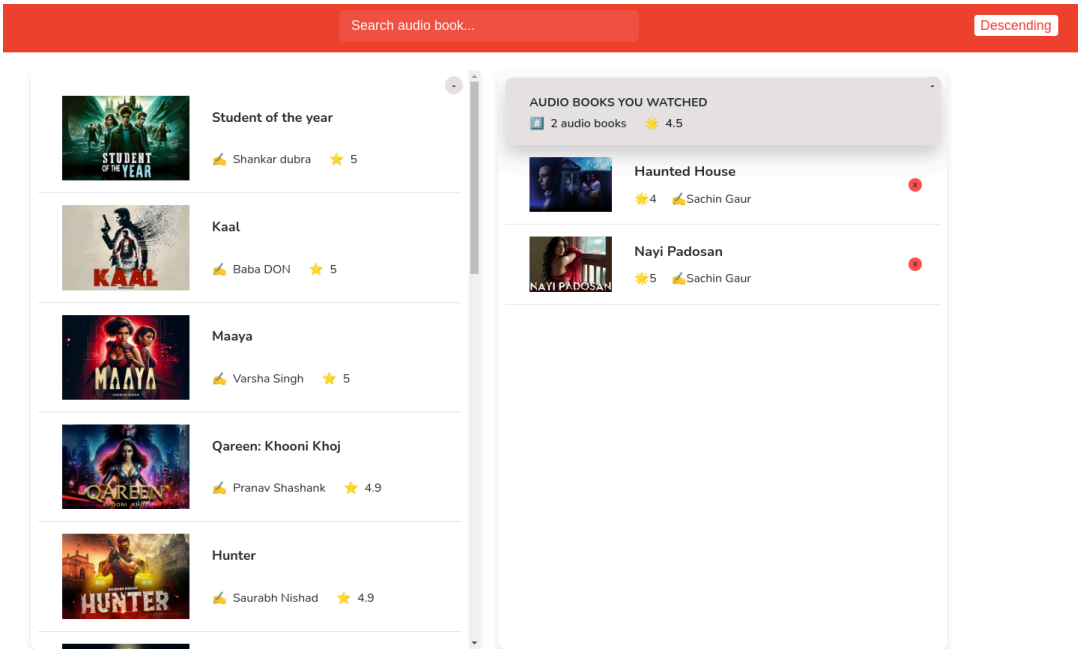
As we see the book has been added to the watch list updated summary details like no of books and rating which can be removed too.



On searching author name we can see the list updating



These are Crime genre books on updating the Genre



Reviews in Descending order