Full Stack Developer Assignment: Movie Review Platform

Objective

Develop a movie review platform where users can browse movies, read and write reviews, and rate films. The application should have a React frontend and a Node.js backend using Express and SQL/MongoDB.

Requirements

Frontend (React)

1. Create a responsive UI with the following pages/components:
    - Home page with featured movies and trending films
    - Movie listing page with search and filter functionality (by genre, year, rating, etc.)
    - Individual movie page with details, trailers, cast, and reviews
    - User profile page with review history and watchlist
    - Review submission form with star ratings and text reviews
    - Movie watchlist functionality
2. Implement state management (e.g., using Redux or React Context)
3. Use React Router for navigation
4. Integrate with the backend API
5. Implement error handling and loading states
6. Optional: Integrate with a movie database API (like TMDB) for movie posters and details

Backend (Node.js, Express, SQL/MongoDB)

1. Set up a RESTful API with the following endpoints:
    - GET /movies - Retrieve all movies (with pagination and filtering)
    - GET /movies/:id - Retrieve a specific movie with reviews
    - POST /movies - Add a new movie (admin only)

    - GET /movies/:id/reviews - Retrieve reviews for a specific movie
    - POST /movies/:id/reviews - Submit a new review for a movie
    - GET /users/:id - Retrieve user profile and review history
    - PUT /users/:id - Update user profile
    - GET /users/:id/watchlist - Retrieve user's watchlist
    - POST /users/:id/watchlist - Add movie to watchlist
    - DELETE /users/:id/watchlist/:movieId - Remove movie from watchlist
2. Implement data validation and error handling

3. Use SQL/MongoDB for data persistence

4. Implement user authentication (registration/login)

5. Calculate and store average ratings for movies

Database Schema Considerations

- Movies: title, genre, release year, director, cast, synopsis, poster URL, average rating
- Users: username, email, password (hashed), profile picture, join date
- Reviews: user ID, movie ID, rating (1-5 stars), review text, timestamp
- Watchlist: user ID, movie ID, date added

Evaluation Criteria

1. Code quality and organization
2. Proper use of React hooks and components
3. RESTful API design and implementation
4. Database schema design and relationships
5. Error handling and edge case management
6. User authentication and authorization
7. Documentation clarity
8. UI/UX design considerations
9. Performance optimization (pagination, caching, etc.)

Bonus Features (Optional)

- Movie recommendation system based on user ratings
- Social features (follow other users, see their reviews)
- Advanced search with multiple filters
- Movie trailer integration
- Admin dashboard for managing movies and users
- Real-time notifications for new reviews on watchlisted movies

Submission

1. Provide a GitHub repository link containing your project
2. Include a comprehensive README with:
    - Setup and installation instructions
    - API documentation
    - Database setup instructions
    - Environment variables required
    - Any additional notes or design decisions
3. (Optional) Deploy the application and provide a live demo URL

Technical Notes

- Ensure responsive design for mobile and desktop
- Implement proper error boundaries in React
- Use environment variables for sensitive configuration
- Include input validation on both frontend and backend
- Consider implementing rate limiting for API endpoints