MovieShop Technical Specification

Project Overview

MovieShop is an e-commerce platform for purchasing and managing movies. The project will be developed by teams of 4-5 students.

Tech Stack

- NextJS 15 with App Router
- PostgreSQL
- Prisma (ORM)
- Better Auth (Authentication)
- Tailwind CSS (Styling)
- ShadCN (UI Components)
- Zod (Data Validation)

Core Features

1. User Authentication

- Implement user registration and login functionality using Better Auth
- Create user roles: Customer and Admin (optional)
- Utilize Better Auth default schema for user data

2. Movie Management (Admin)

- CRUD operations for movies:
    - Add new movies with details (title, description, price, release date, director, actors, etc.)
    - Edit existing movie information
    - Delete movies
    - List all movies with pagination (optional pagination)
- Implement movie genre management:
    - CRUD operations for genres
    - Assign movies to multiple genres
- Manage people associated with movies:
    - Add and edit information about directors and actors

    - Associate people with movies in different roles (director, actor)

3. Shopping Experience (Customer)

- Landing page
    - Top 5 most purchased movie
    - Top 5 most recent movies
    - Top 5 Oldest Movies
    - Top 5 cheapest Movie
- Browse movies (optional filtering)
    - By genre
    - By director
    - By actor
- Search functionality for movies (basic search by title)
- View detailed movie information
- Add movies to cart (stored in cookies)
- Manage cart (add, remove, update quantities)
- Checkout process:
    - Address input
    - Payment simulation (no real payment gateway required)
    - Order confirmation

4. User Dashboard

- View order history
- Manage account information (optional)

5. Admin Dashboard (optional)

- View sales statistics
- Manage user accounts (using Better Auth features)

Database Models (Examples of potential props)

Note: These are examples of props the models could contain, not the full schema.

- All Better Auth models
- Movie:
    - title, description, price, releaseDate, imageUrl, stock, runtime
- Genre:
    - name, description
- Order:
    - userId (reference to Better Auth user), totalAmount, status, orderDate
- OrderItem:
    - orderId, movieId, quantity, priceAtPurchase

Server Actions and Data Validation

- Implement server actions for all data mutations:
    - Movie CRUD operations
    - Genre management
    - Cart operations (add, remove, update)
    - Checkout process
    - Order management
- Use Zod for server-side data validation within server actions
- Utilize Better Auth for user-related actions (registration, login, profile updates)

Cart Implementation

- Use cookies to store cart information
- Implement functions to add, remove, and update cart items in the cookie

Frontend

- Develop responsive layouts using Tailwind CSS and ShadCN components
- Create reusable React components for common UI elements
- Use React Server Components where appropriate for improved performance
- Integrate Better Auth components for user authentication UI

Additional Features (Only if there's extra time)

Note: These features should only be attempted if the core features are completed and there's additional time available.

1. Advanced Browsing and Filtering
    - Implement more complex filters (e.g., release year range, runtime, multiple genres)
2. User Reviews and Ratings
    - Allow customers to rate and review purchased movies
    - Display average ratings on movie listings
3. Wishlist
    - Enable users to add movies to a wishlist for future purchase
4. Basic Recommendation System
    - Implement a simple recommendation system based on user purchase history, favorite genres, directors, or actors
5. Movie Trailer Integration
    - Add functionality to link and display movie trailers (embedded YouTube videos)
6. Social Sharing
    - Add buttons to share movie links on social media platforms
7. Advanced Search
    - Implement full-text search for movies using PostgreSQL's full-text search capabilities
    - Include search by director, actor, and genre
8. Discount System
    - Create a simple discount system for special offers or promotional codes

Evaluation Criteria

- Functionality: All core features working as specified
- Code Quality: Clean, well-organized, and commented code
- Database Design: Proper use of Prisma and efficient database schema, including correct integration with Better Auth user schema
- UI/UX: Intuitive and responsive design using Tailwind and ShadCN
- Authentication: Secure implementation of user authentication and authorization using Better Auth
- Server Actions: Efficient and secure implementation of server actions for data mutations
- Data Validation: Proper use of Zod for server-side data validation
- Cart Implementation: Correct use of cookies for cart management
- Error Handling: Robust error handling and user feedback
- Additional Features: Successful implementation of extra features (if attempted, not required)
- Teamwork: Effective collaboration and task distribution among team members