11/17/2025

SYSTEM REQUIREMENT & DESIGN DIAGRAM

231662 – Mamoona

231659 – Maheen Fatima

231617 – Huzaifa Yasir Khan

Table of Contents

Table of Figures

s

1. Introduction & Problem Statement

Context: In today's digital entertainment landscape viewers use multiple streaming platforms, often forgetting titles they wanted to watch, being unsure where a movie streams, and lacking a single personalized hub to manage watchlists and preferences.

Problem Statement:

- Users forget titles and lose track of movies across platforms.
- It is hard to know where a movie is streaming at any given time.
- No unified, free, highly-personalized solution exists that combines search, streaming availability, watchlist management, trailers, and parental controls in one place.

Goal: Build **CinaVerse**, a full-stack web application that integrates with TMDb (and JustWatch fallback), provides centralized watchlist & review management, streaming availability, parental & age filters, authentication, screens/subscriptions management, and logging/auditing.

2. Proposed Solution

CinaVerse will be a web application with:

- Frontend: React or Vue.js (SPA)
- Backend: Node.js or Express (RESTful services)
- Database: PostgreSQL (or SQLite for small-scale)
- External Integrations: TMDb API for movie data; JustWatch as fallback for streaming availability
- Features: Search, movie details, trailers, watchlist CRUD, reviews, user auth, admin panel, parental controls, screen purchase/plan management, logs (login/API usage)

3. Requirements

- **Functional Requirements (User Stories)**

FR-1: User Registration

- **Actor:** Visitor / User
- **User Story:** As a visitor, I want to register an account, so that I can create a personal profile and access watchlists and premium features.
- **Preconditions:** User is not logged in; registration page is accessible
- **Postconditions (Success):**
    - Account created in database
    - Confirmation email sent
    - User redirected to dashboard
- **Postconditions (Failure):**
    - Account not created
    - Error message displayed (e.g., email already exists)

FR-2: User Login / Authentication

- **Actor:** User
- **User Story:** As a user, I want to log in securely, so that my personal watchlist and account remain private.
- **Preconditions:** User has a registered account; login page accessible
- **Postconditions (Success):**
    - User session created
    - Dashboard displayed
    - Login activity logged
- **Postconditions (Failure):**

- User remains logged out
- Error message displayed (wrong credentials)

FR-3: User Logout

- **Actor:** User
- **User Story:** As a user, I want to log out to terminate my session.
- **Preconditions:** User is logged in
- **Postconditions (Success):**
  - Session terminated
  - User redirected to home page
  - Logout logged in system
- **Postconditions (Failure):**
  - Session remains active
  - Error message displayed

FR-4: Fetch Movie via API (TMDb)

- **Actor:** User
- **User Story:** As a user, I want to search movies and fetch details via TMDb API, so that I can view information about any movie.
- **Preconditions:** User logged in; TMDb API accessible
- **Postconditions (Success):**
  - Movie details displayed (title, poster, description, cast, rating, genres, trailer link)
  - API usage logged
- **Postconditions (Failure):**
  - Movie details not loaded

- Error message displayed (API unavailable or rate limit exceeded)

FR-5: Watch Movie Trailer / Demo Video

- **Actor:** User
- **User Story:** As a user, I want to watch a movie trailer or demo video via API, so I can preview the movie before adding it to my watchlist.
- **Preconditions:** Movie selected; video API or YouTube link available
- **Postconditions (Success):**
  - Video plays in embedded player
  - User can pause, resume, or stop
  - Watch activity logged
- **Postconditions (Failure):**
  - Video does not play
  - Error message displayed

Only trailers or sample videos can be streamed. Full movie streaming requires licensing.

FR-6: Add / Update / Remove Movies from Watchlist

- **Actor:** User
- **User Story:** As a user, I want to manage my watchlist, so I can track movies I want to watch.
- **Preconditions:** User logged in; movie selected
- **Postconditions (Success):**
  - Movie added, removed, or updated with status (watched/unwatched)
  - Changes logged
- **Postconditions (Failure):**
  - Operation fails

- Error message displayed

FR-7: Submit / View / Delete Reviews

- **Actor:** User
- **User Story:** As a user, I want to post reviews and see others' reviews, so I can share opinions and make decisions.
- **Preconditions:** User logged in; movie selected
- **Postconditions (Success):**
  - Review posted, edited, or deleted
  - Aggregated ratings updated
  - Actions logged
- **Postconditions (Failure):**
  - Review not saved or deleted
  - Error displayed

FR-8: View Streaming Availability

- **Actor:** User
- **User Story:** As a user, I want to see where a movie is streaming, so I know which platforms offer it.
- **Preconditions:** Movie selected; streaming availability API accessible
- **Postconditions (Success):**
  - Platforms displayed (Netflix, Prime, Disney+, rental options)
  - API call logged
- **Postconditions (Failure):**
  - Streaming platforms not displayed

    - Error message shown

FR-9: Admin Panel

- **Actor:** Admin
- **User Story:** As an admin, I want to manage users, reviews, watchlists, and analytics, so I can moderate content and monitor activity.
- **Preconditions:** Admin logged in; admin rights verified
- **Postconditions (Success):**
    - Users, reviews, and watchlists can be managed
    - Analytics accessible
    - Admin actions logged
- **Postconditions (Failure):**
    - Action blocked
    - Error message displayed

FR-10: Parent / Age Filter Panel

- **Actor:** Parent
- **User Story:** As a parent, I want to set age or genre restrictions, so children only see appropriate content.
- **Preconditions:** Parent account logged in; parental settings page accessible
- **Postconditions (Success):**
    - Restrictions applied to child accounts
    - Settings logged
- **Postconditions (Failure):**
    - Restrictions not applied

    - Warning displayed

FR-11: View Available Plans (Subscription / Screens)

- **Actor:** User
- **User Story:** As a user, I want to view available subscription plans or “screens” so that I can choose a suitable plan.
- **Preconditions:** User logged in; plans page accessible
- **Postconditions (Success):**
    - List of available plans displayed with features and pricing
    - Plans retrieved from database/API
- **Postconditions (Failure):**
    - Plans not displayed
    - Error message shown

FR-12: Purchase Plan / Screen (Subscription / Stripe API)

- **Actor:** User
- **User Story:** As a user, I want to purchase a subscription plan so I can unlock premium features such as additional watchlist capacity, parental filters, or analytics access.
- **Preconditions:** User logged in; Stripe payment gateway accessible
- **Postconditions (Success):**
    - Payment processed successfully
    - Premium features unlocked
    - Purchase logged
- **Postconditions (Failure):**
    - Payment failed

- Features not unlocked
- Error displayed

FR-13: Logging & Analytics

- **Actor:** Admin / Developer
- **User Story:** As a system admin, I want to log user activity, API calls, and watchlist/review changes, so I can audit the system and detect abuse.
- **Preconditions:** Logging service active
- **Postconditions (Success):**
  - Activities recorded securely in database
  - Logs accessible to admin
- **Postconditions (Failure):**
  - Logs not recorded
  - Admin notified

Non-Functional Requirements

NFR-1: Security

- **Description:** Protect user accounts, API keys, and payment information.
- **Metric / Success:** Password hashing, JWT sessions, Stripe PCI compliance.
- **Failure / Risk:** Data leak, unauthorized access.

NFR-2: Performance

- **Description:** System should be responsive for all user operations.
- **Metric / Success:** API calls <500ms; watchlist operations fast.
- **Failure / Risk:** Lag, slow API responses, poor UX.

NFR-3: Availability

- **Description:** The app should be accessible to users at all times.
- **Metric / Success:** 99% uptime for core features.
- **Failure / Risk:** Downtime, features inaccessible to users.

NFR-4: Scalability

- **Description:** System should support many concurrent users.
- **Metric / Success:** Modular backend enabling horizontal scaling.
- **Failure / Risk:** System slows down or crashes under high load.

NFR-5: Privacy

- **Description:** Comply with applicable privacy laws.
- **Metric / Success:** Minimal personal data stored; GDPR/CCPA compliant.
- **Failure / Risk:** Privacy breach, non-compliance penalties.

NFR-6: Usability

- **Description:** The application should be user-friendly and accessible.
- **Metric / Success:** Mobile-first responsive UI, accessible forms (a11y).
- **Failure / Risk:** Layout breaks on devices; confusing forms; poor UX.

NFR-7: Maintainability

- **Description:** The system should be easy to maintain and update.
- **Metric / Success:** Well-documented REST APIs, UML diagrams.
- **Failure / Risk:** Hard to maintain, undocumented code, slow updates.

NFR-8: Logging & Audit

- **Description:** Track and audit user and system actions.
- **Metric / Success:** Logs for login, API usage, watchlist, reviews.
- **Failure / Risk:** Missing or incomplete logs; poor audit trail.

NFR-9: Reliability

- **Description:** The system should handle errors gracefully and continue operation.
- **Metric / Success:** Errors handled properly; retries on API failures.
- **Failure / Risk:** Crashes, unhandled exceptions, data inconsistency.

4. System Design & Artifacts

- **Class Diagram**

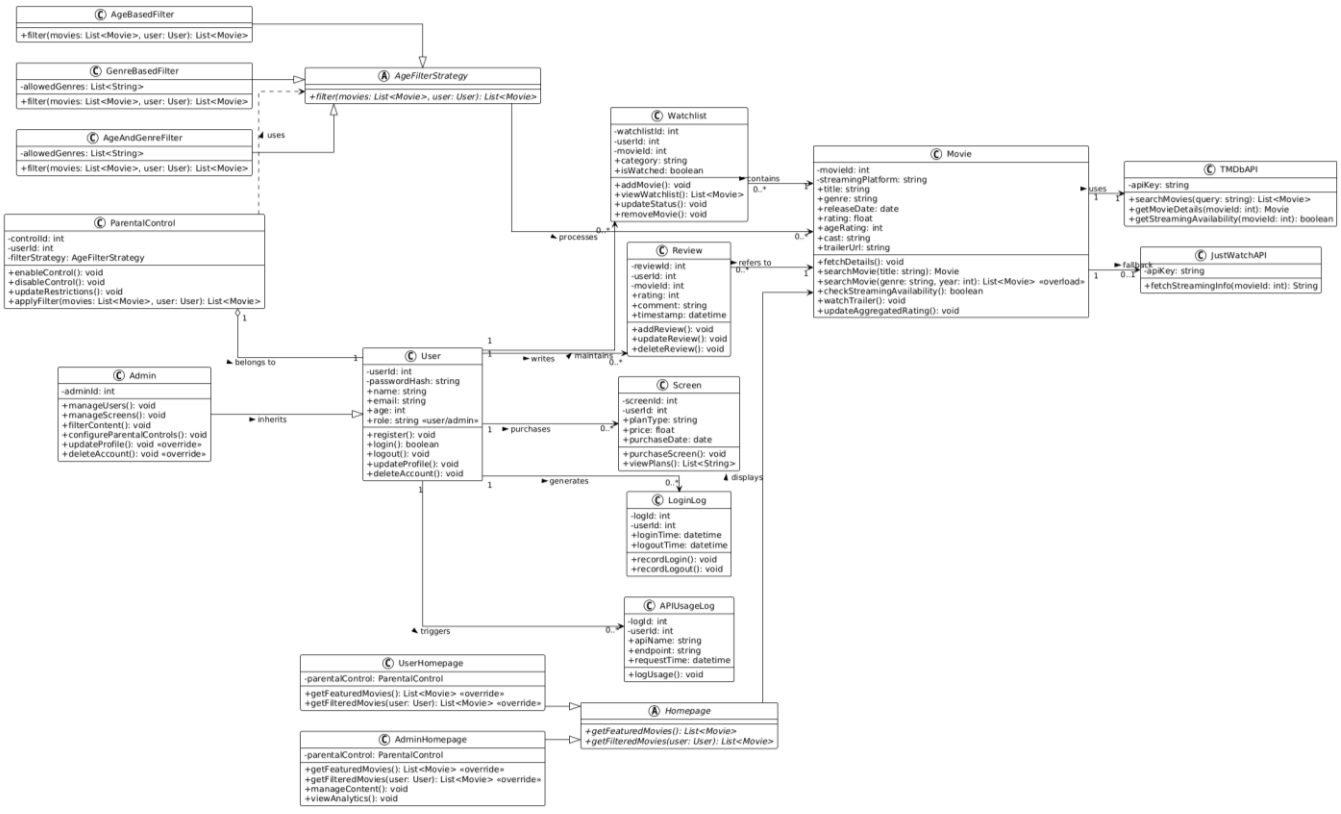

Figure 1 - Class diagram

- **Activity Diagram**

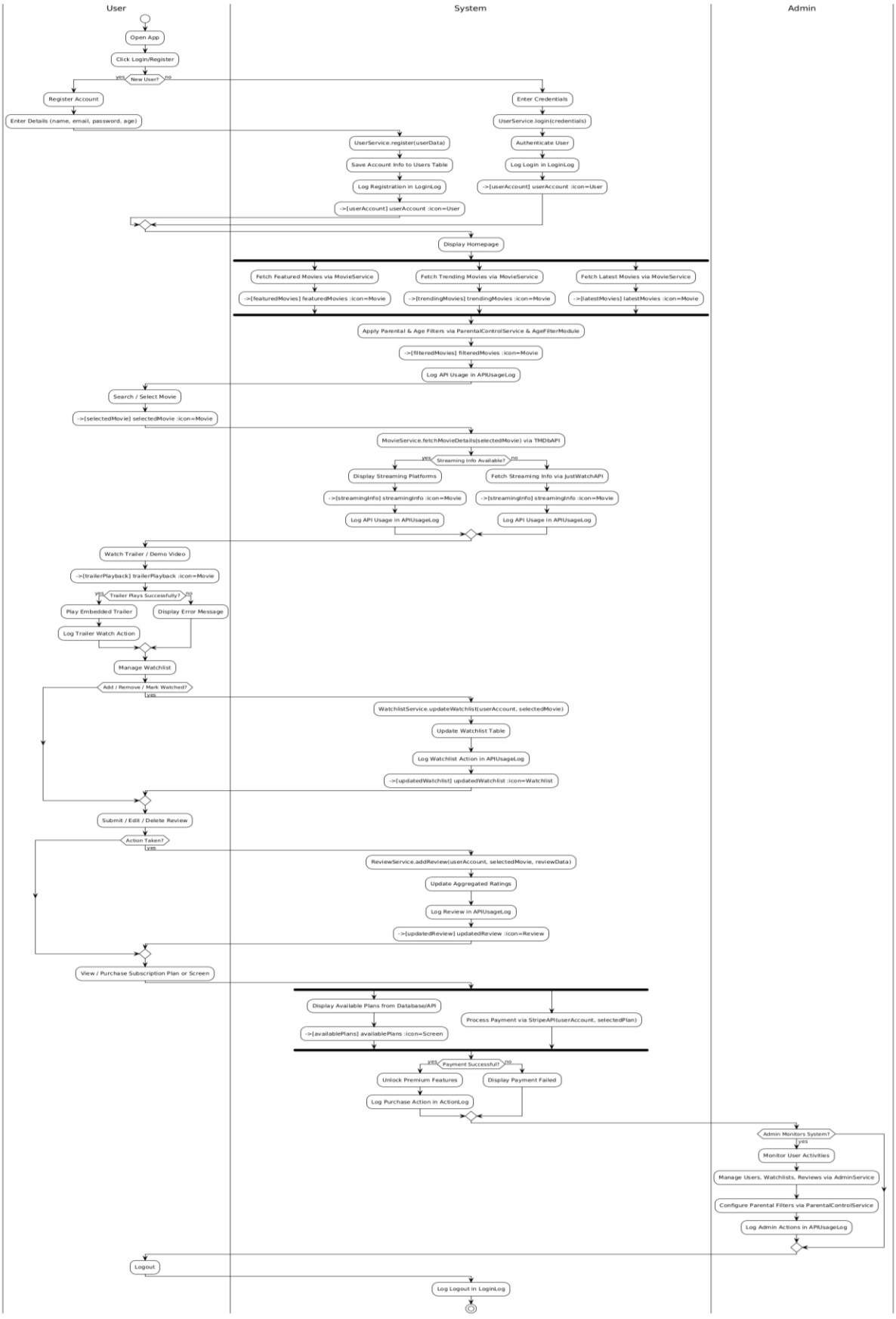

Figure 2 - activity diagram

- **Architecture Diagram**

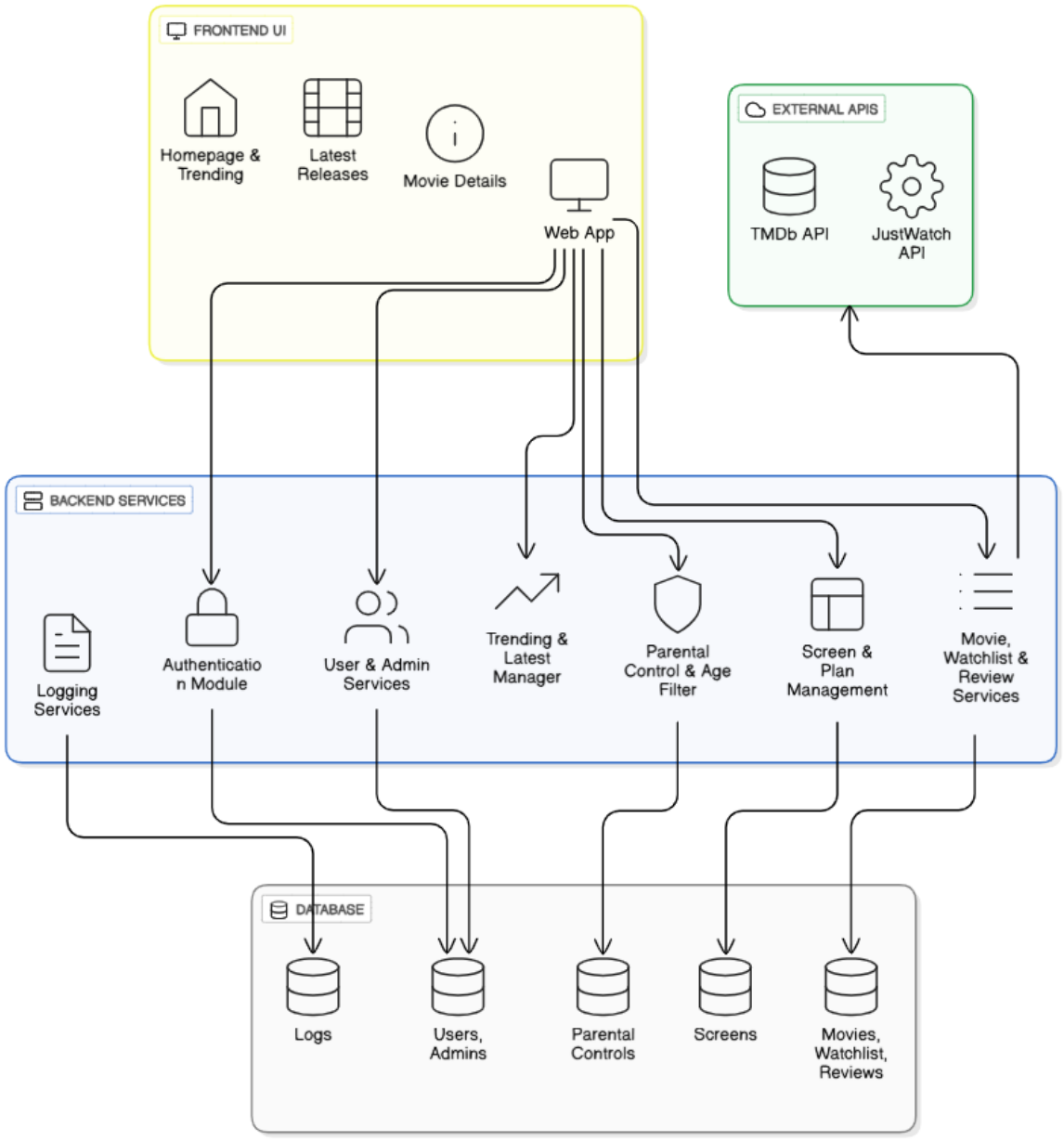

Figure 3 - architecture diagram

Microservices Architecture is used, the overall system's backend is decomposed into multiple, small, independent services (like the **Authentication Module**, **Parental Control & Age Filter**, and **Movie, Watchlist & Review Services**), each owning its distinct data store and operating independently. This structure allows for decoupled development, deployment, and scaling of individual functions. We can use the **Model-View-Controller (MVC)** pattern *within* each of these individual services and the frontend component.

- **ERD of the system.**

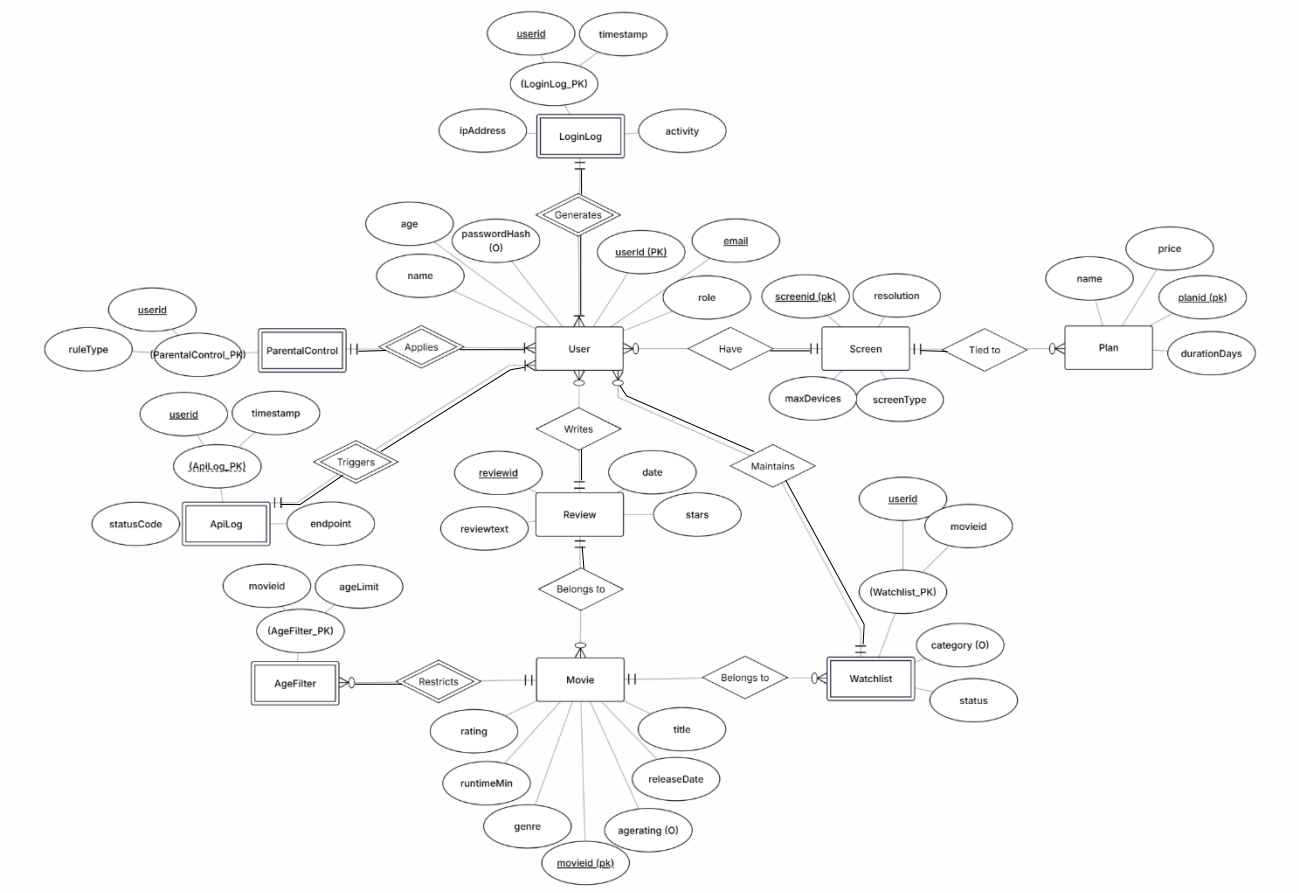

Figure 4 – ERD

- **Use Case Diagram**

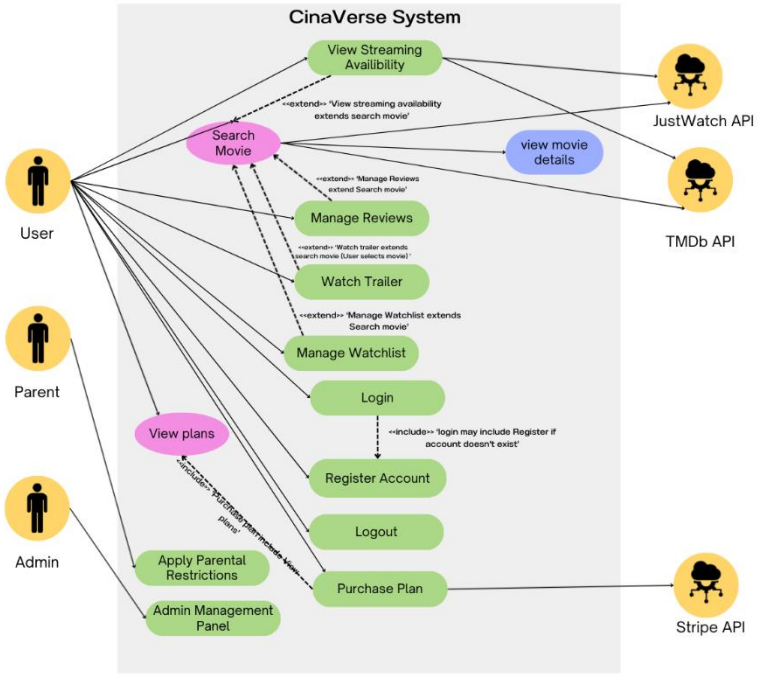

Figure 5 - usecase

UC-01: Register Account (FR-1)

Primary Actor: User

Preconditions:

- User is not already registered
- System is available

Postconditions (Success):

- New user account is created
- User data is saved in the database

Postconditions (Failure):

- Account is not created
- Error message is displayed

Main System Flow:

1. User selects the "Register" option
2. System displays the registration form
3. User enters required details

4. User submits the form
5. System validates the data
6. System creates the account
7. System shows confirmation message

Alternate Flows:

- 3a. If email already exists → System displays error message

Exceptions:

- Database connection failure
- Server timeout

UC-02: Login (FR-2)

Primary Actor: User

Preconditions:

- User account exists

Postconditions (Success):

- User is authenticated
- User session is created

Postconditions (Failure):

- Access is denied
- Error message is displayed

Main System Flow:

1. User opens the login page
2. User enters username and password
3. System verifies credentials
4. System grants access

Alternate Flows:

- 3a. Invalid credentials → System prompts retry

Exceptions:

- Account locked after multiple failed attempts

UC-03: Logout (FR-3)

Primary Actor: User

Preconditions:

- User is logged in

Postconditions (Success):

- User session is terminated

Postconditions (Failure):

- Session remains active

Main System Flow:

1. User clicks the logout button
2. System terminates the session
3. System redirects to homepage

Alternate Flows:

- 2a. Session already expired → System redirects to login

Exceptions:

- Session storage error

UC-04: Search Movie (FR-4)

Primary Actor: User

Preconditions:

- User is logged in

Postconditions (Success):

- Movie search results are displayed

Postconditions (Failure):

- No results are shown

Main System Flow:

1. User enters a movie name
2. System sends request to TMDb API
3. System receives search results

4. System displays results

Alternate Flows:

- 2a. No results found → System displays “No results found”

Exceptions:

- External API timeout

UC-05: Watch Trailer (FR-5)

Primary Actor: User

Preconditions:

- User has selected a movie

Postconditions (Success):

- Trailer video is played

Postconditions (Failure):

- Trailer does not play

Main System Flow:

1. User clicks “Watch Trailer”
2. System loads video player
3. System streams the trailer

Alternate Flows:

- 2a. Trailer unavailable → System displays message

Exceptions:

- Video streaming error

UC-06: Manage Watchlist (FR-6)

Primary Actor: User

Preconditions:

- User is logged in

Postconditions (Success):

- Watchlist is updated

Postconditions (Failure):

- Watchlist remains unchanged

Main System Flow:

1. User selects a movie
2. User chooses add/update/remove
3. System updates watchlist in database

Alternate Flows:

- 2a. Movie already exists in watchlist → System notifies user

Exceptions:

- Database write failure

UC-07: Manage Reviews (FR-7)

Primary Actor: User

Preconditions:

- User is logged in
- A movie is selected

Postconditions (Success):

- Review is saved/edited/deleted

Postconditions (Failure):

- Review is not modified

Main System Flow:

1. User opens review section
2. User writes or edits a review
3. User submits the review
4. System stores the review

Alternate Flows:

- 3a. Empty review → System displays validation error

Exceptions:

- Database error

UC-08: View Streaming Platforms (FR-8)

Primary Actor: User

Preconditions:

- A movie is selected

Postconditions (Success):

- List of streaming platforms is displayed

Postconditions (Failure):

- Platform list is not displayed

Main System Flow:

1. User clicks “View Streaming Options”
2. System sends request to JustWatch API
3. System receives data
4. System displays platforms

Alternate Flows:

- 2a. No platforms available → System displays message

Exceptions:

- API service failure

UC-09: Manage Users/Reviews/Watchlists (FR-9)

Primary Actor: Admin

Preconditions:

- Admin is logged in

Postconditions (Success):

- Selected data is successfully managed

Postconditions (Failure):

- No data changes are saved

Main System Flow:

1. Admin opens admin dashboard
2. Admin selects a management option

3. Admin performs edit/delete actions
4. System saves changes

Alternate Flows:

- 2a. Invalid selection → System shows error

Exceptions:

- Database failure

UC-10: Apply Age/Genre Restrictions (FR-10)

Primary Actor: Parent

Preconditions:

- Parent account is logged in

Postconditions (Success):

- Restrictions are saved and enforced

Postconditions (Failure):

- Restrictions are not applied

Main System Flow:

1. Parent opens parental control settings
2. Parent selects age and genre restrictions
3. System validates the settings
4. System saves the restrictions

Alternate Flows:

- 2a. Invalid age range → System shows warning

Exceptions:

- Configuration save failure

UC-11: View Plans (FR-11)

Primary Actor: User

Preconditions:

- User is logged in

Postconditions (Success):

- Subscription plans are displayed

Postconditions (Failure):

- Plans are not loaded

Main System Flow:

1. User navigates to plans page
2. System retrieves available plans
3. System shows plans

Alternate Flows:

- 2a. No plans available → System displays message

Exceptions:

- Service retrieval error

UC-12: Purchase Plan/Screens (FR-12)

Primary Actor: User

Preconditions:

- User is logged in
- A plan is selected

Postconditions (Success):

- Subscription is activated

Postconditions (Failure):

- Payment fails and subscription is not activated

Main System Flow:

1. User selects a subscription plan
2. System redirects to payment gateway
3. User completes payment
4. System confirms payment
5. System activates subscription

Alternate Flows:

- 3a. User cancels payment → System cancels process

Exceptions:

- Payment gateway error

UC-13: Log User Activity & API Calls (FR-13)

Primary Actor: Admin

Preconditions:

- System is operational

Postconditions (Success):

- Activity is recorded in logs

Postconditions (Failure):

- Log entry is not saved

Main System Flow:

1. System captures user activity
2. System stores log entry
3. Admin can view logs

Alternate Flows:

- 2a. Log storage full → System rotates logs

Exceptions:

- Log database failure

- **System Sequence Diagram**

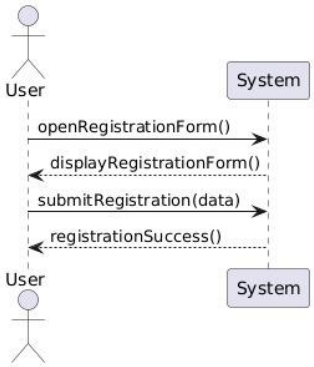

Figure 6 - register account ssd

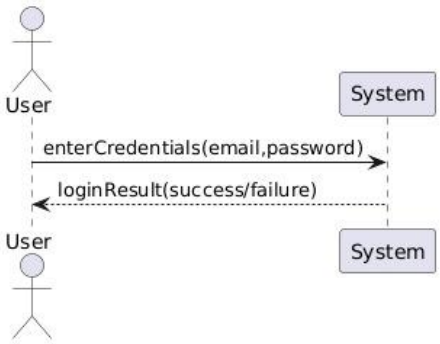

Figure 7 - login ssd

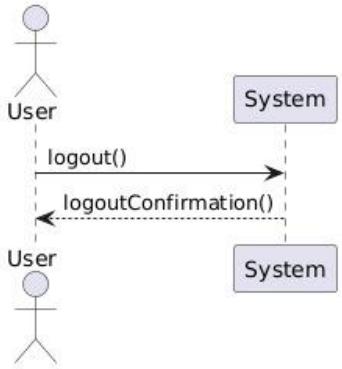

Figure 8 - logout ssd

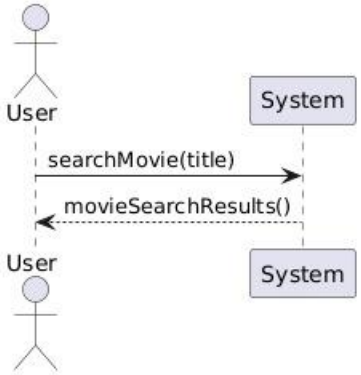

Figure 9 - searchmovie via tmdbapi ssd

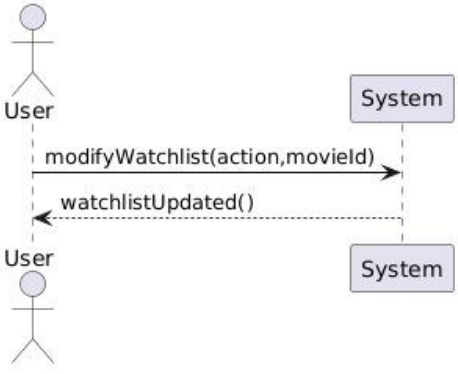

Figure 10 - add/update/remove from watchlist ssd

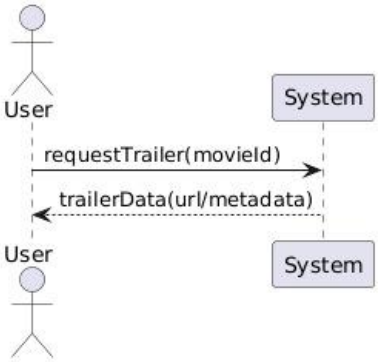

Figure 11 - watchtrailer ssd

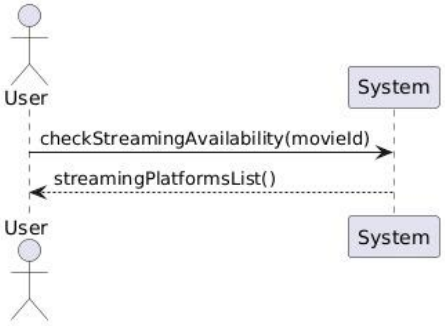

Figure 12 - view streaming platforms ssd

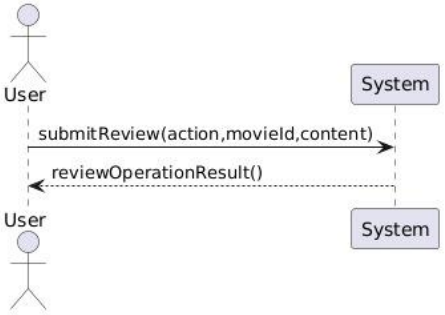

Figure 13 - submit / edit / delete review

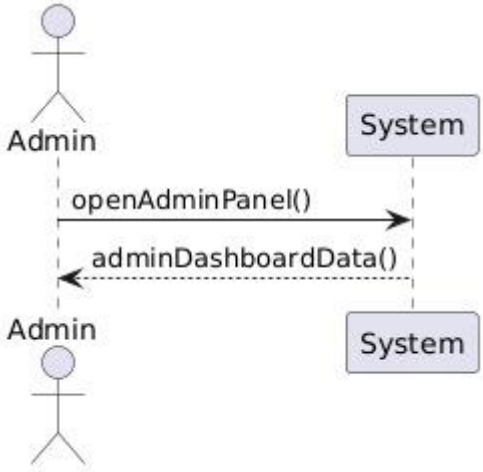

Figure 14 - manage users / reviews / watchlists

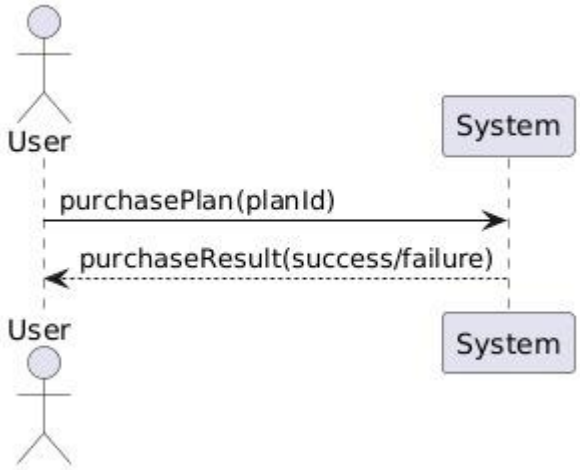

Figure 15 - purchase plans/ screens ssd

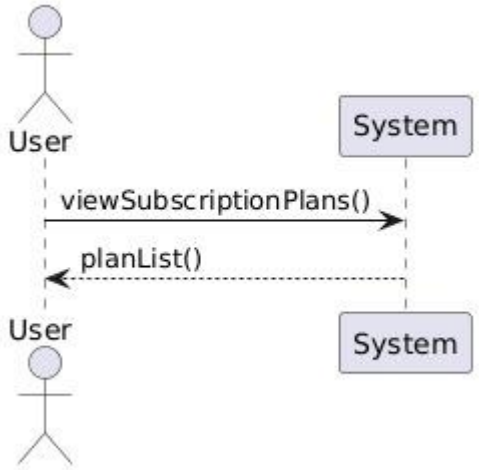

Figure 17 - view plans ssd

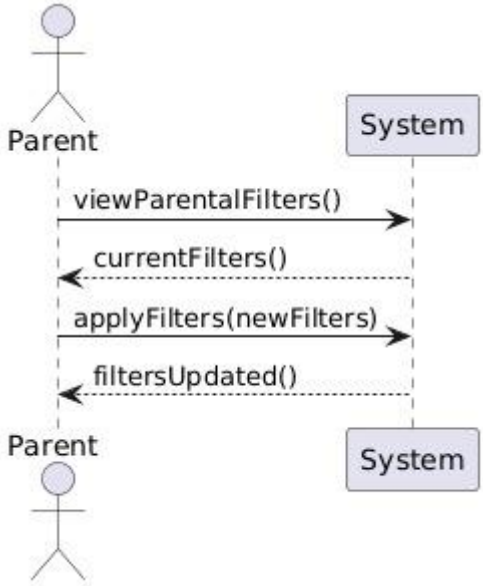

Figure 16 - apply age/ genre restrictions

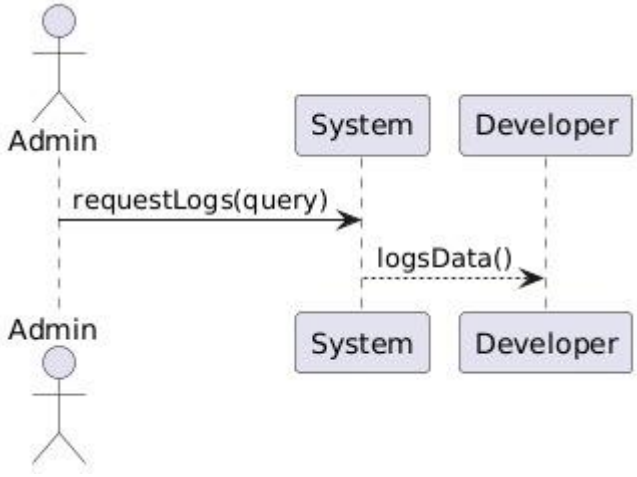

Figure 18 - log user activity and api calls ssd

- **Sequence Diagram**

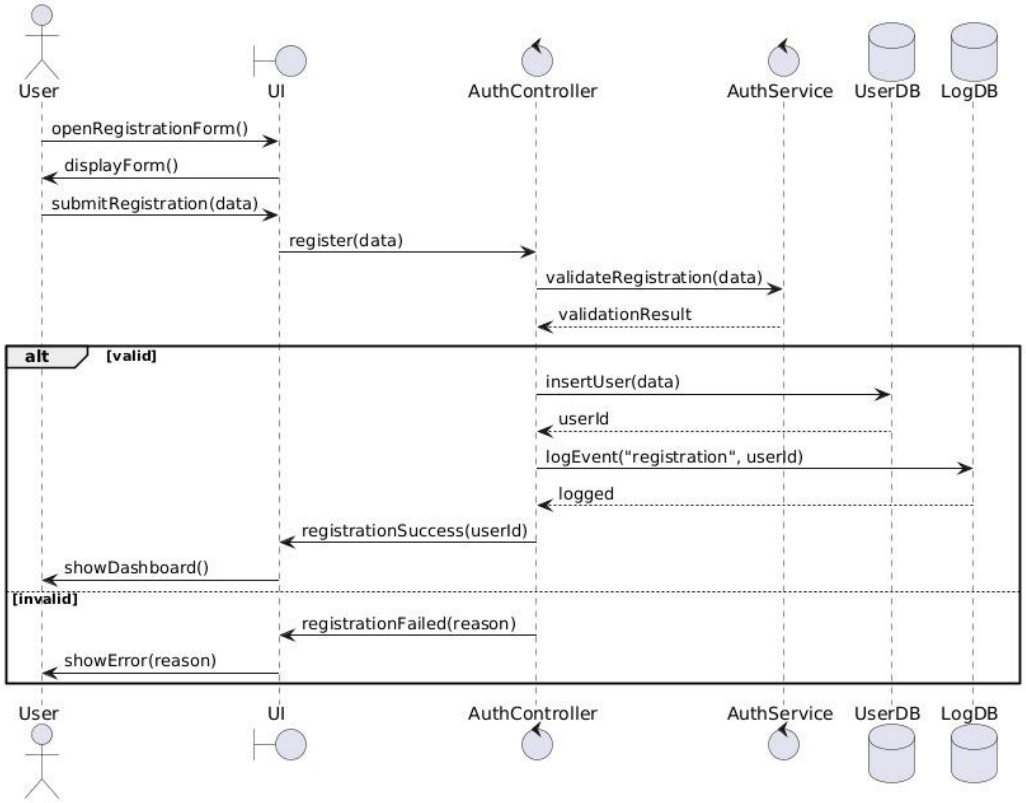

Figure 19 - register account sequence

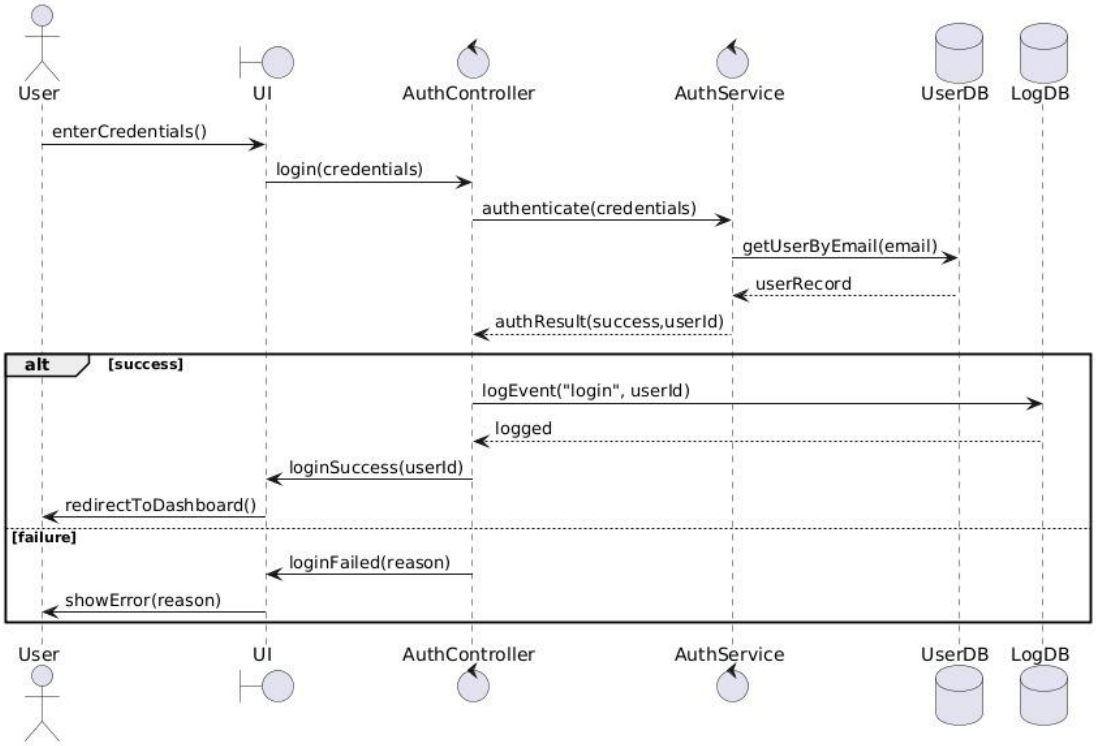

Figure 20 - login sequence

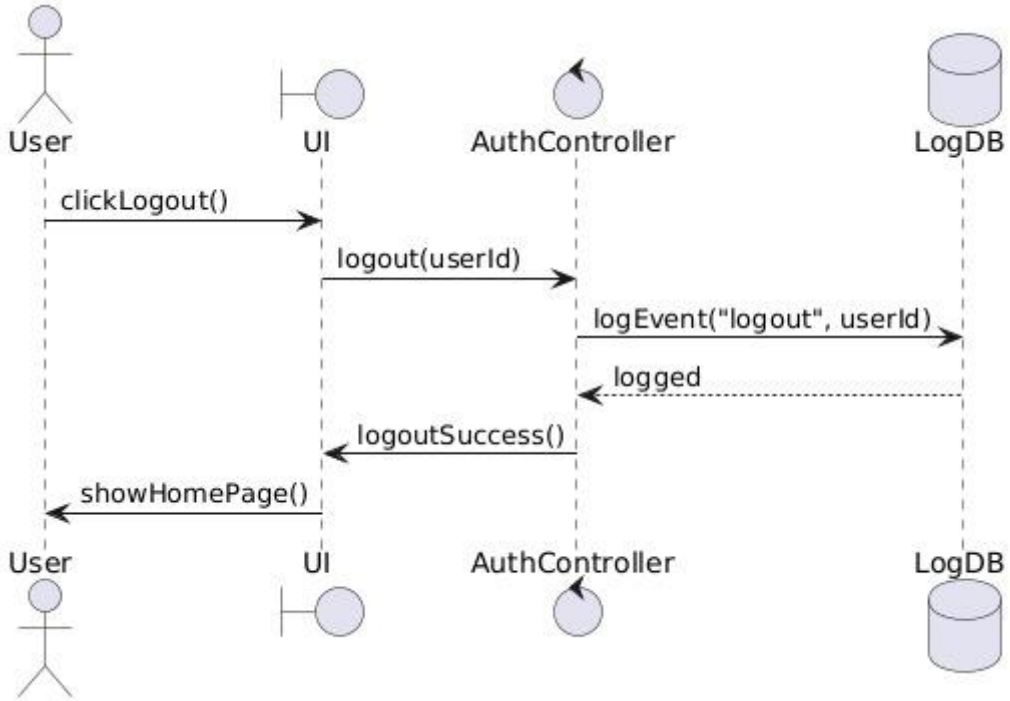

Figure 21 - logout sequence

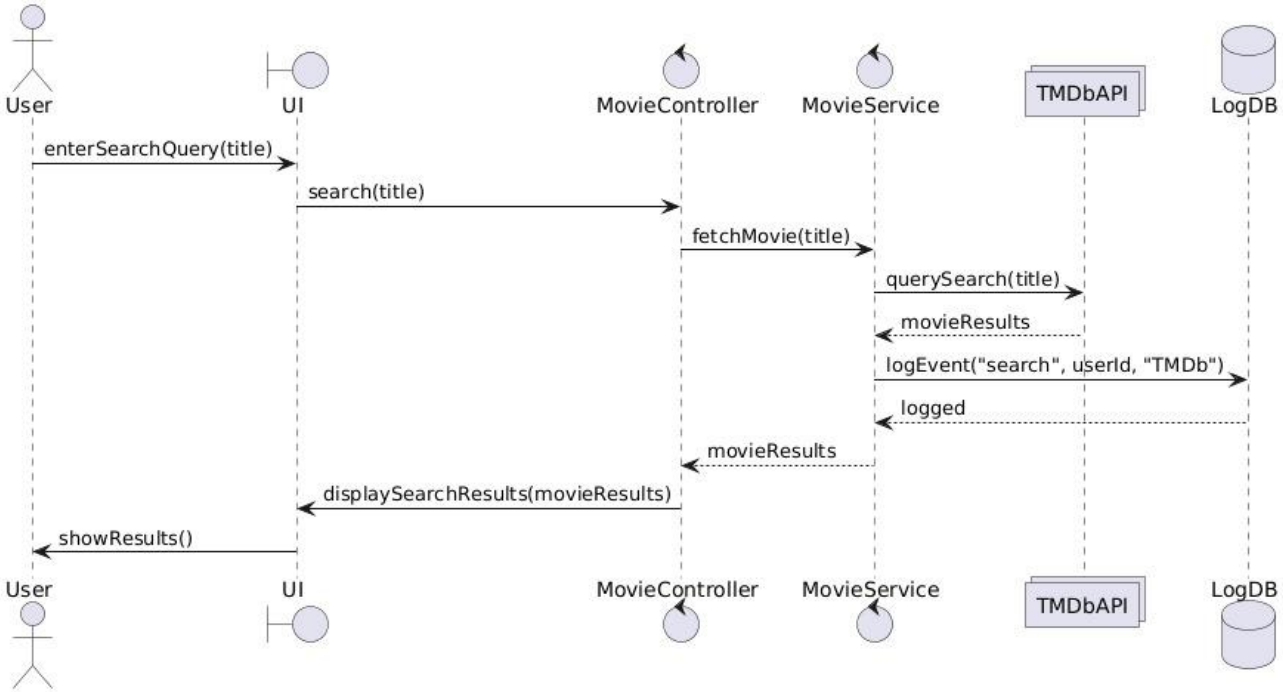

Figure 22 - search movie via tmdb api sequence

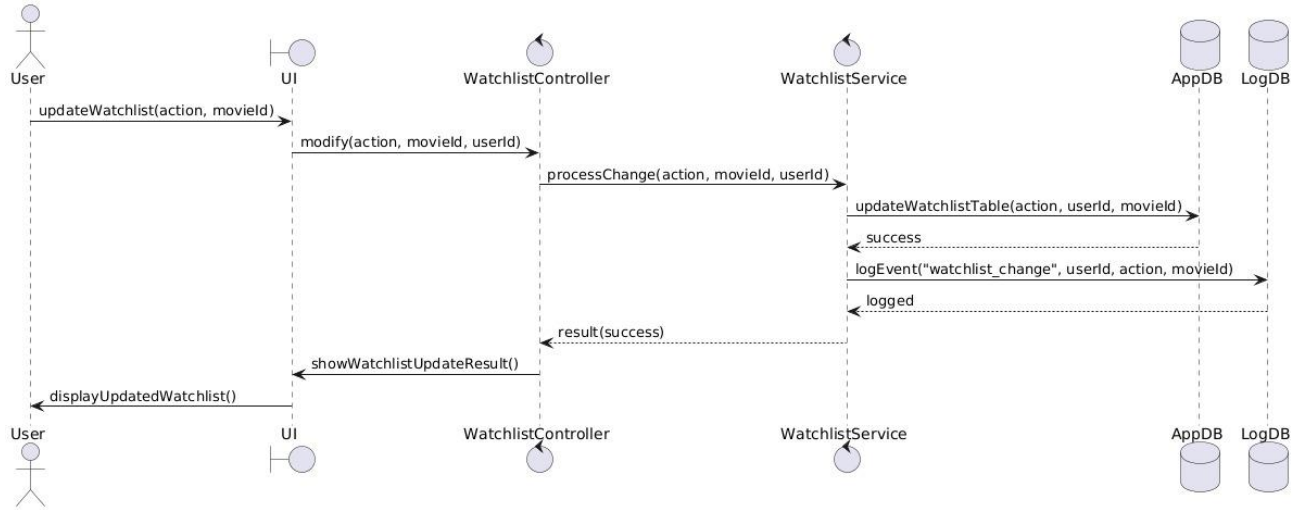

Figure 23 - add/ update / remove from watchlist sequence

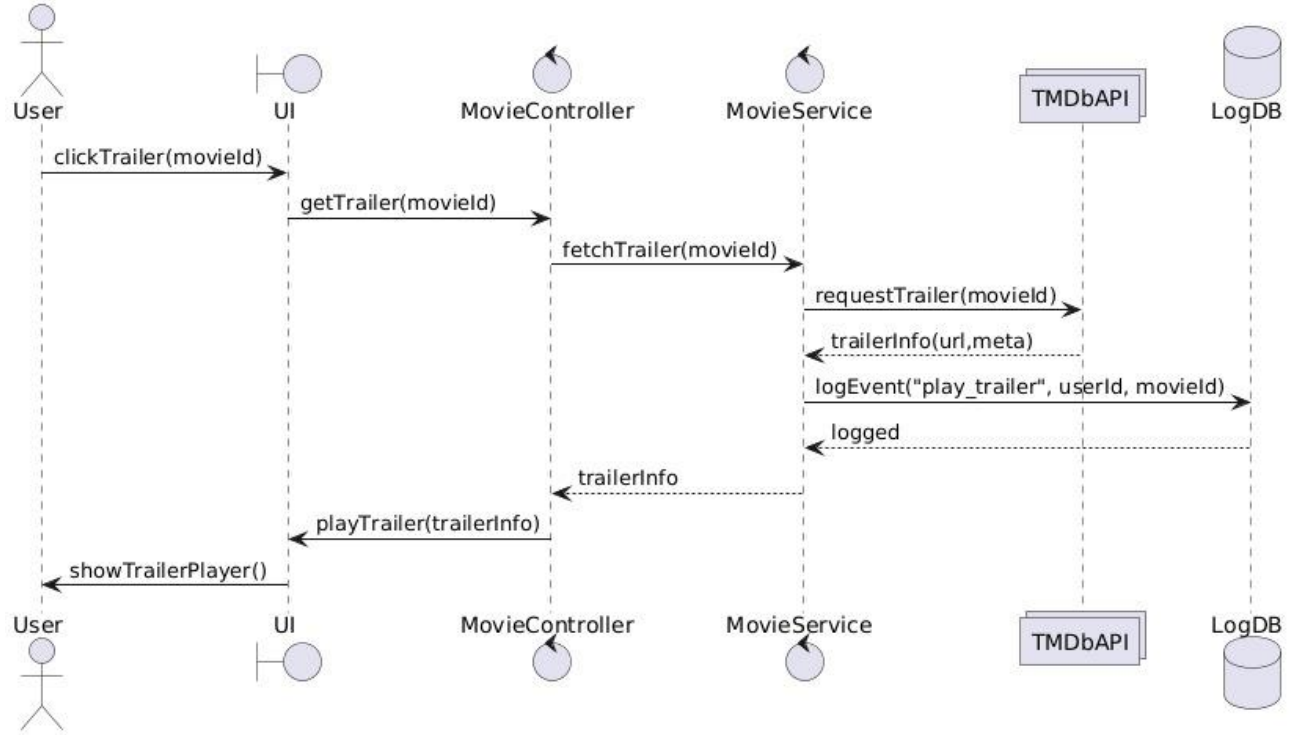

Figure 24 - watch trailer sequence

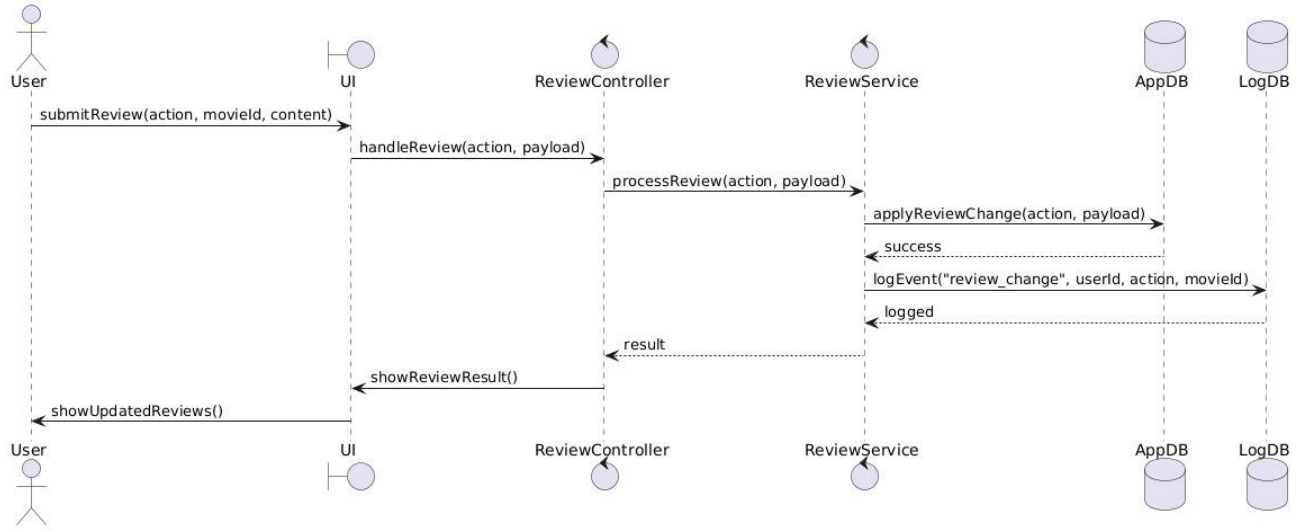

Figure 25 - submit / edit / delete review sequence

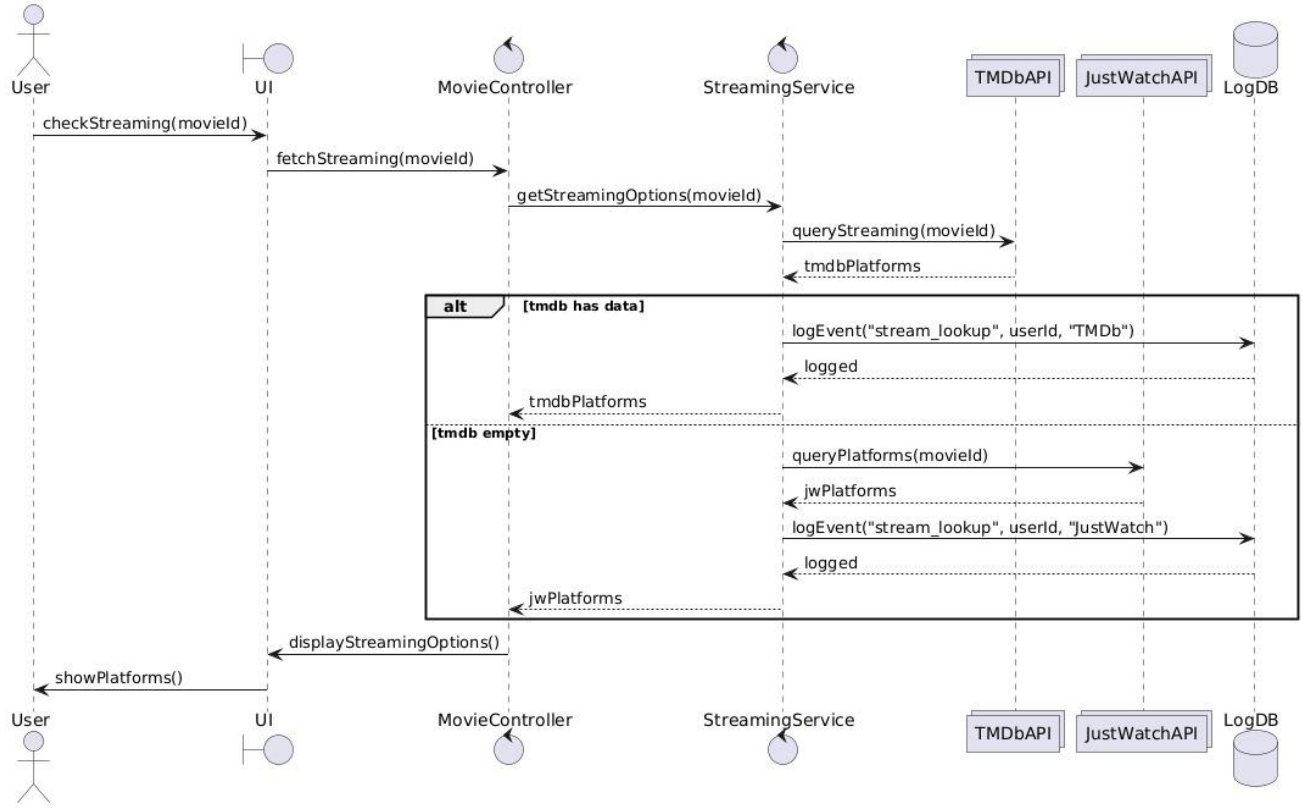

Figure 26 - view streaming platforms sequence

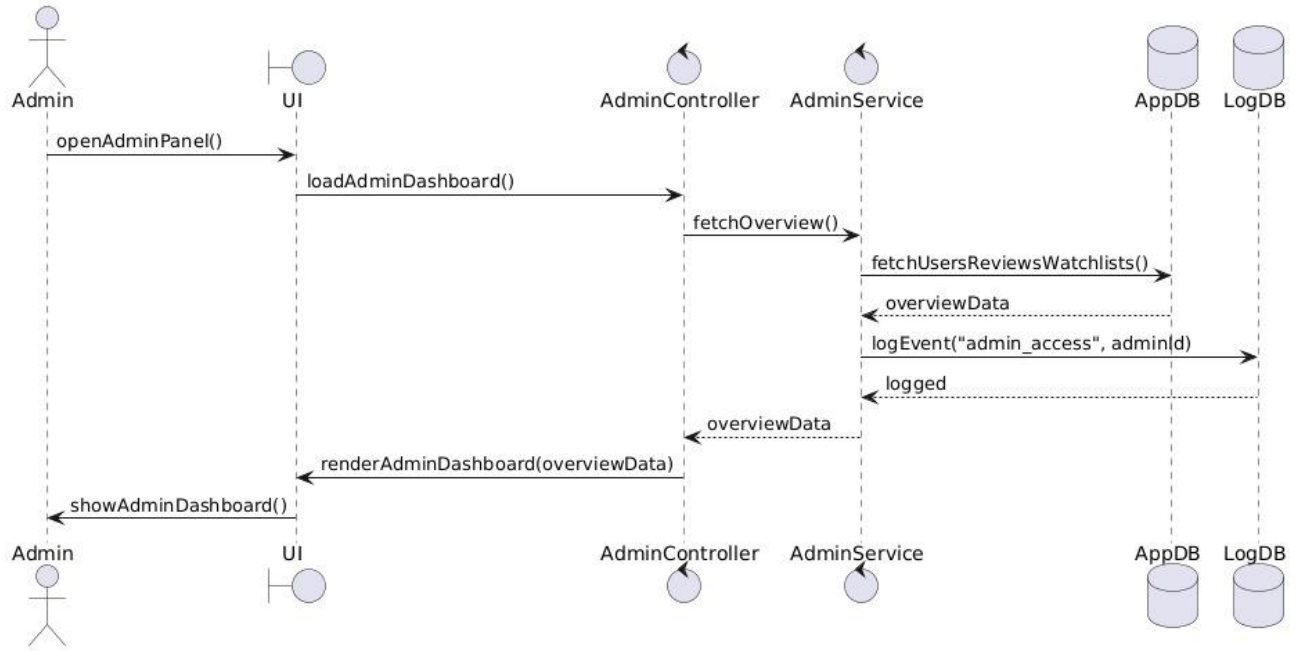

Figure 27 - manage user / review / watchlist sequence

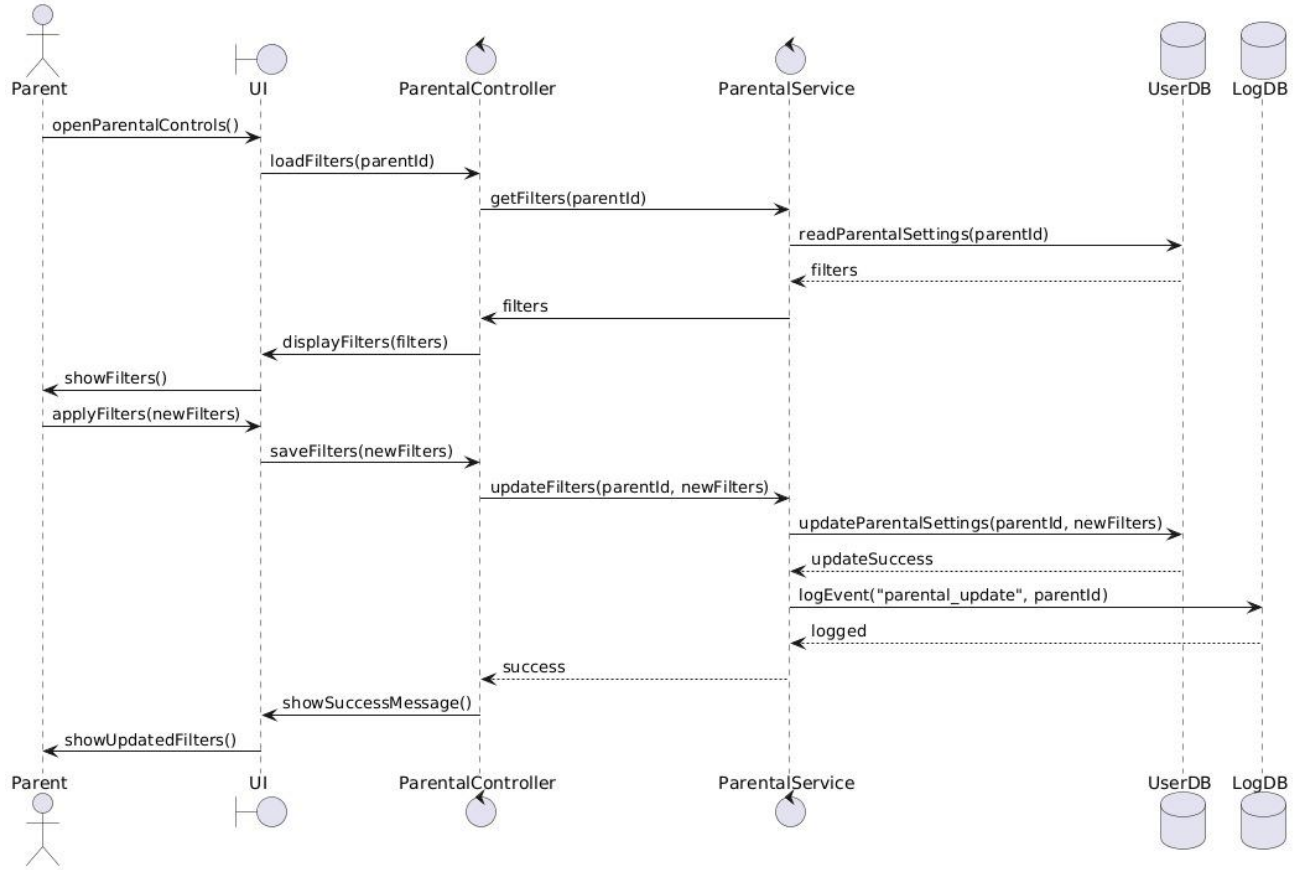

Figure 28 - apply age / genre restrictions sequence

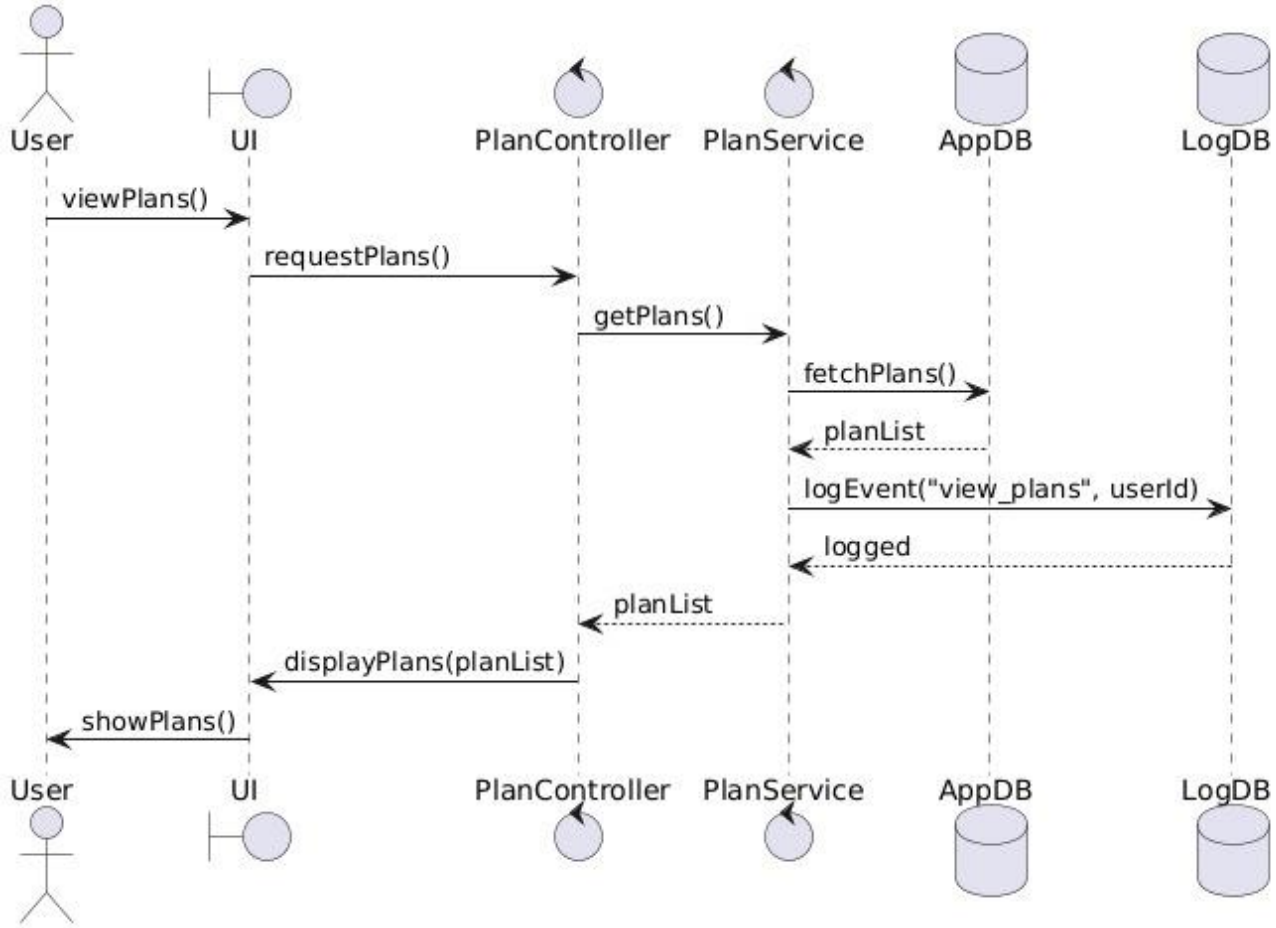

Figure 29 - view plans sequence

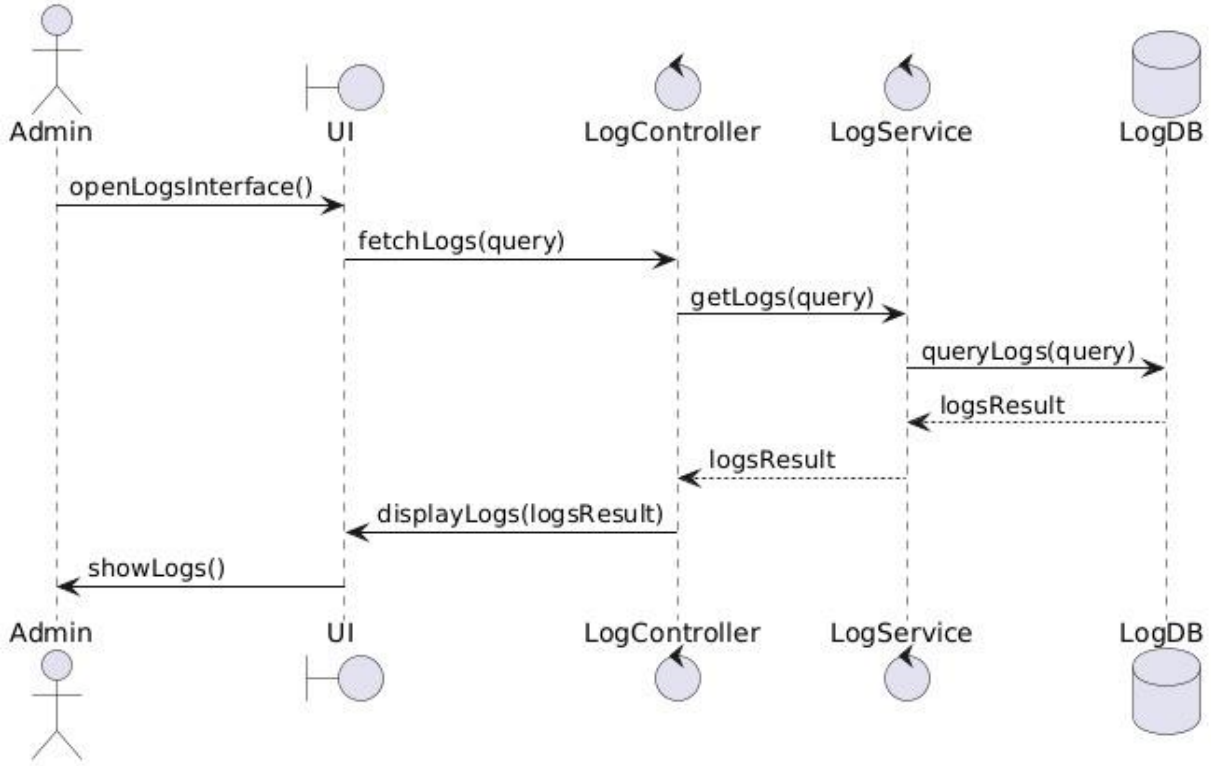

Figure 30 - log user activity and api calls sequence

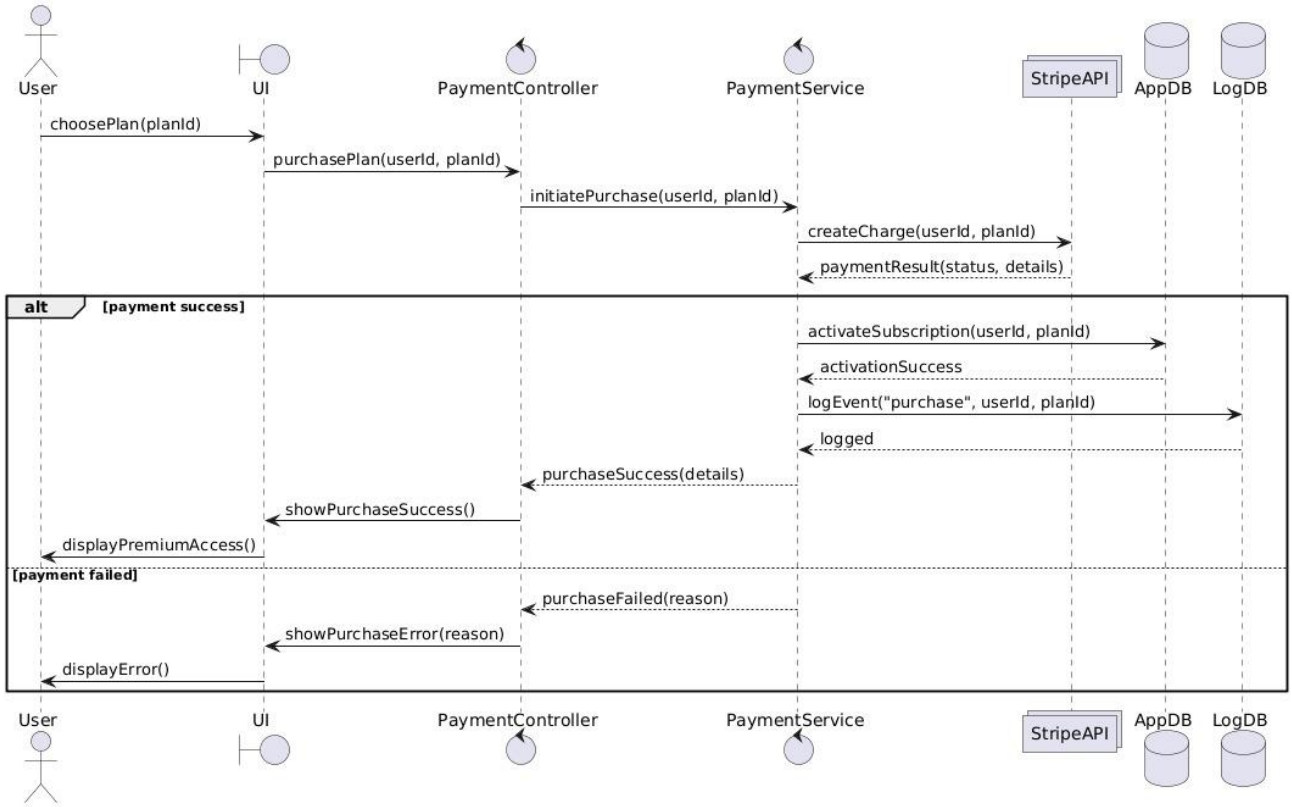

Figure 31 - purchase plan/ screen sequence

- **Domain Model**

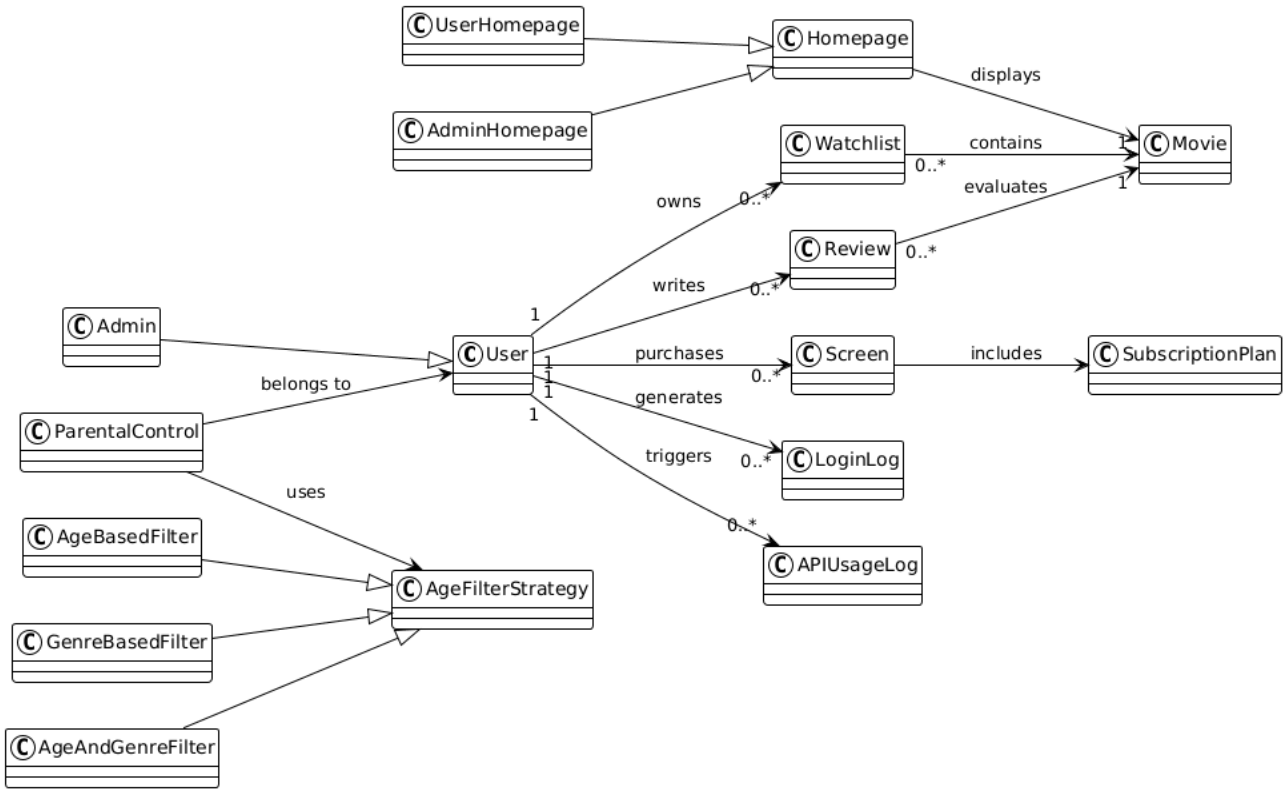

Figure 32 - domain model diagram

- **Deployment Diagram**

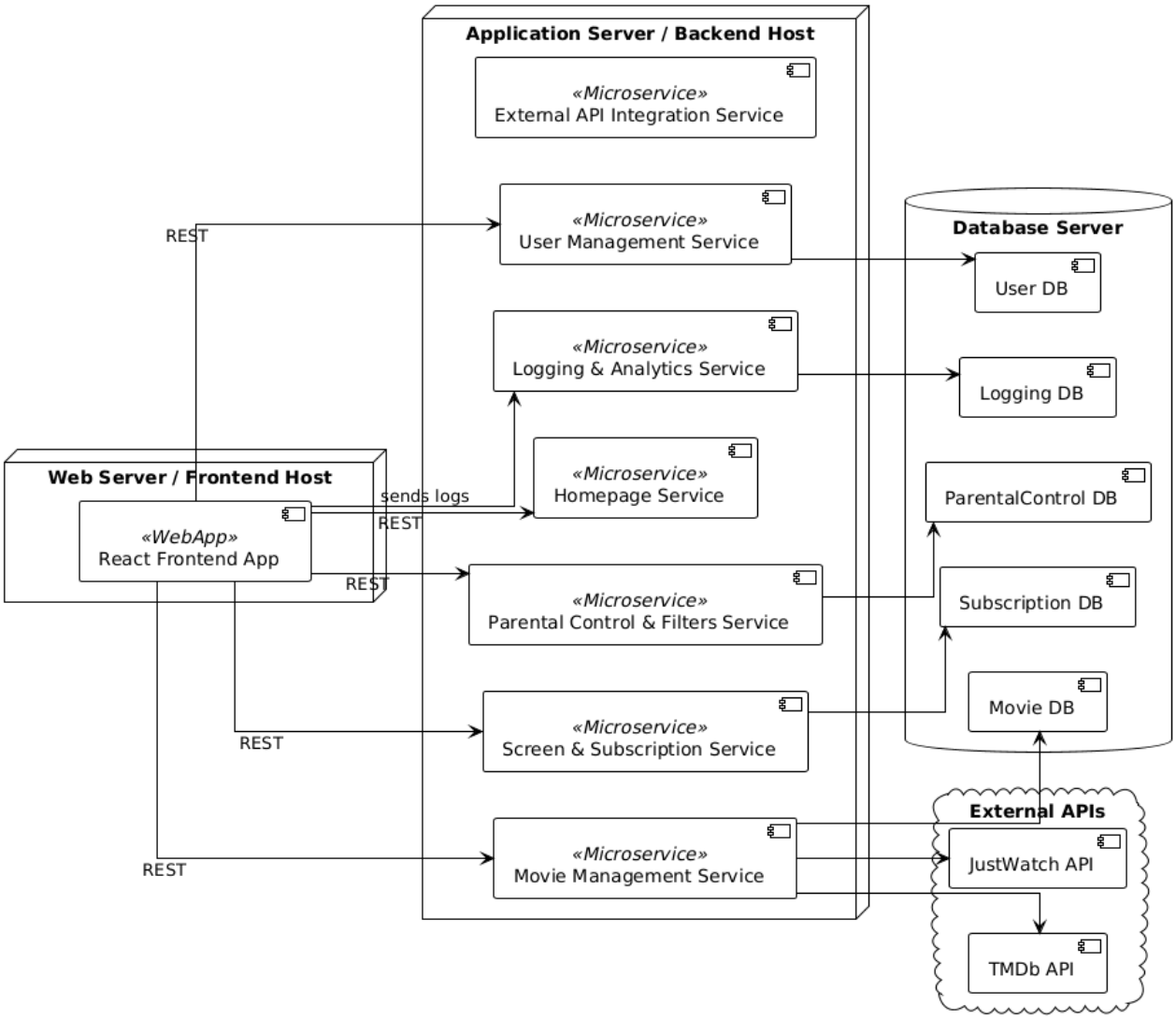

Figure 33 - Deployment diagram

- **Package Diagram**

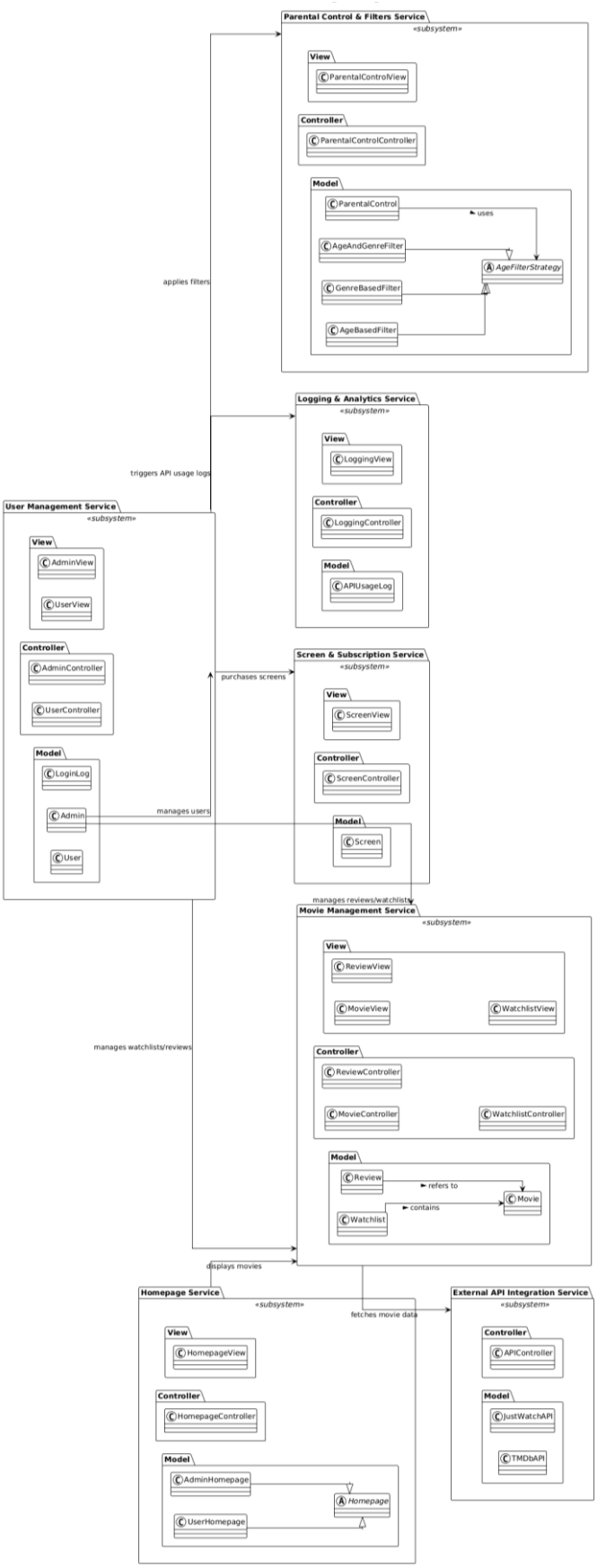

Figure 34 - Package Diagram

- **Component Diagram**

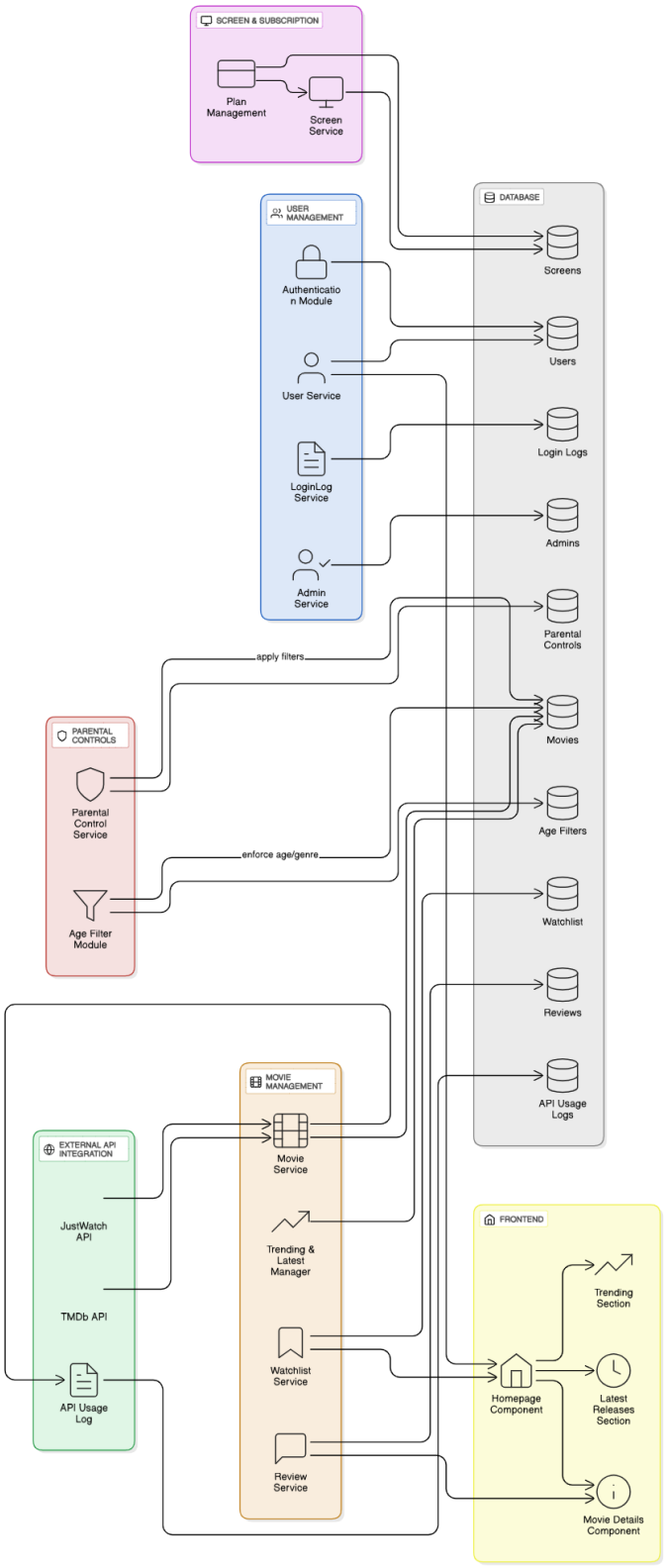

Figure 35 - component diagram