InsightX

Media Intelligence Platform – System Overview

---

1. Overview

InsightX is a cross-media intelligence platform designed to unify **movies, books, and games** into a single, coherent discovery experience.
Rather than acting as a simple catalog, InsightX focuses on **context, quality, and reasoning**—helping users understand *what* to consume, *why* it is recommended, and *how* it can be experienced.

InsightX emphasizes:

- Rich metadata aggregation
- Cinematic, media-first UI/UX
- Explainable recommendations
- Region-aware availability
- Technical capability awareness
- Progressive personalization based on user behavior

---

2. High-Level Architecture

InsightX follows a **layered, service-oriented architecture**:

Flutter Client

|

| HTTPS (JWT)

v

Spring Boot Backend (System of Record)

|

| Internal HTTP

v

FastAPI Intelligence Service

|

v

External Metadata & Content APIs

A shared **Redis Free Tier instance** is used as an auxiliary caching layer.

---

3. Technology Stack

Client Layer

- Flutter
- Dart
- Custom animations and transitions
- Media rendering (images, trailers)
- Secure local storage

Core Backend

- Spring Boot 3
- Java 17
- Spring Security (JWT-based authentication)
- Spring Data JPA
- PostgreSQL
- WebClient
- **Redis (Free Tier)**

Responsibilities

- Authentication and authorization
- User profiles and preferences
- Region management
- Watched status
- Ratings and reviews
- Persistence and transactional integrity
- API orchestration

Intelligence & Aggregation Layer

- FastAPI
- Python 3.11+
- AsyncIO with httpx
- Pydantic
- **Redis (Free Tier)**

- LLM integration for reasoning and explanations

Responsibilities

- External API aggregation
- Media normalization
- Recommendation scoring
- Cross-media mapping
- Theme extraction
- AI-assisted explanations
- Visual asset resolution (logos, posters, backdrops)

---

4. Redis Free Tier Usage Strategy

Redis is used as a **non-critical, performance-oriented layer**, ensuring InsightX remains functional even without cache availability.

Cached Data Types

- External API responses (TMDB, books, games)
- Watch provider lookups
- Recommendation result sets
- AI explanation outputs
- Frequently accessed media details

Design Principles

- Cache-aside pattern
- Short to medium TTLs
- Safe eviction behavior
- No source-of-truth data stored in Redis

Benefits

- Reduced API rate-limit pressure
- Faster UI response times
- Lower backend load
- Cost-efficient scalability using free-tier limits

---

5. Unified Media Model

InsightX models all content types using a **shared base schema**, with media-specific extensions.

Common Fields

- ID
- Title
- Media type (movie / book / game)
- Visuals (logo, poster or cover, backdrop)
- Genres
- Description
- Ratings (normalized)
- Availability
- User state (watched, rating)

The intelligence layer guarantees **best-effort completeness** and explicit fallbacks for missing data.

---

6. Media-Specific Features

🎬 **Movies**

- Logos, posters, backdrops, trailers
- Runtime, genres, language, cast & crew
- IMDb, Rotten Tomatoes, Metacritic ratings
- Streaming availability (region-aware)
- Technical capabilities (resolution, HDR, audio)
- Blu-ray / UHD Blu-ray reference links

📚 **Books**

- Cover visuals and derived backdrops
- Author, page count, publisher, categories
- Aggregated ratings
- Preview and purchase options

🎮 **Games**

- Cover art, background art, screenshots, trailers
- Platforms, genres, developers

- Metacritic and user ratings
- System requirements
- Average completion time
- Store availability

---

7. Visual Asset Strategy

Logos and images are treated as **optional enhancements**.

Fallback Priority

1. Logo (if available)
2. Styled title text
3. Poster / cover
4. Backdrop

This ensures consistent rendering regardless of asset availability.

---

8. User Interaction & State

User Actions

- Mark as watched
- Rate content
- Write reviews
- Save or bookmark

Stored Data

- Watch history
- Ratings
- Reviews
- Preferences
- Region

Derived Taste Profile

Generated from:

- Rating behavior
- Genre frequency
- Theme affinity

Used directly by the recommendation engine.

---

9. Recommendation System

Design Philosophy

- Deterministic
- Explainable
- Inspectable

Signals

- Genre overlap
- Theme similarity
- Creator overlap
- User preference weighting

Output

- Ranked recommendations
- Per-item explanation signals

---

10. AI Companion

Role

- Explain recommendations
- Assist exploration
- Answer contextual questions
- Provide cross-media reasoning

Constraints

- Operates only on structured inputs
- No direct data fetching
- No ranking authority

---

11. API Flow

Client → Spring Boot

- JWT-secured REST APIs

Spring Boot → FastAPI

- Internal HTTP
- Service-level authentication

FastAPI → External APIs

- Async fan-out
- Redis-backed caching
- Normalized responses

---

12. Data Storage

PostgreSQL

- Users
- Preferences
- Watched entries
- Ratings and reviews

Redis (Free Tier)

- Metadata caching
- Recommendation caching
- AI explanation caching

FastAPI remains stateless.

---

13. UI / UX Principles

- Media-first layouts
- Cinematic transitions
- Dark-theme dominance
- Poster-driven hierarchy
- Smooth navigation motion
- Typography-led readability

---

14. Fault Tolerance

- Cache misses fall back to live fetch
- Missing assets degrade gracefully
- Partial API failures handled via cached data

- AI failures fall back to deterministic explanations

---

15. Extensibility

InsightX supports future expansion into:

- Additional media types
- ML-driven recommendations
- Social discovery
- Advanced personalization
- New data providers

---

16. Summary

InsightX integrates:

- A **Flutter-based cinematic client**
- A **Spring Boot system of record**
- A **FastAPI intelligence engine**
- A **Redis Free Tier caching layer**

Together, these components deliver a **high-performance, explainable, and visually rich media intelligence system** across movies, books, and games.