GamesTrack

Il cuore dei tuoi dati di gioco

Introduzione

GamesTrack è un hub self-hosted personale per la sincronizzazione delle personali librerie di giochi di psn e steam permettendo la centralizzazione dei propri dati in un'unica piattaforma.
Il prodotto in questione permette un unico punto di accesso per consultare la propria intera collezione, visualizzare gli achievement e trofei sbloccati monitorando il tutto senza passaggi tra applicazioni differenti.

Contesto Applicativo

Nell'ecosistema odierno del gaming digitale, è sempre più comune per un appassionato possedere titoli distribuiti su più piattaforme — che siano console diverse, PC o servizi di cloud gaming. Tuttavia, ogni ambiente custodisce i propri dati in silos distinti: statistiche di gioco, ore complessive di utilizzo, achievement o trofei conquistati e livelli di progresso restano isolati e frammentati. Questa frammentazione rende ben più difficile il monitoraggio completo della propria attività videoludica.

Questa situazione si traduce in disagi concreti per l'utente medio, che fatica a tenere aggiornata una lista unificata dei giochi completati o in corso, e rende complicata la consultazione delle ore effettivamente dedicate a ciascun titolo. Allo stesso tempo, per i collezionisti o per chi ama archiviare e organizzare in modo meticoloso la propria collezione, l'assenza di un unico punto di aggregazione trasforma il semplice desiderio di creare elenchi ordinati e filtrabili in un'operazione laboriosa e dispersiva. Una piattaforma di "aggregazione cross-platform" risponde quindi a queste esigenze, offrendo un'interfaccia centralizzata che raccoglie, normalizza e rende confrontabili tutte le informazioni di gioco, da qualunque sistema provengano.

Soluzione Proposta

GamesTrack supera il problema della frammentazione offrendo un'unica dashboard intuitiva e sicura, attraverso la quale ogni videogiocatore può:

- Collegare i propri account Steam e PSN
    - Inserendo in modo protetto le API key (o il token NPSSO per PSN), gli utenti autorizzano GamesTrack a leggere le loro librerie e i progressi di gioco.

- Sincronizzare le librerie con metadati esaustivi
    - In pochi clic, tutte le informazioni relative ai titoli posseduti vengono importate automaticamente: copertine, descrizioni, generi, date di rilascio e dettagli sulla casa di sviluppo, grazie alle integrazioni con IGDB.

- Visualizzare statistiche aggregate
    - Ore totali di gioco, numero di trofei e achievement guadagnati vengono presentati in grafici e tabelle riassuntive: ideale per monitorare i propri traguardi e scoprire quali generi o piattaforme occupano più tempo.

- Esplorare il catalogo IGDB
    - Una ricerca avanzata permette di ordinare i titoli per data di uscita e valutazione, e di filtrarli per console e generi , consultando immediatamente tutti i dettagli dei giochi provenienti dal database IGDB, ovvero copertina, titoli, descrizione, console, generi, sviluppatore, publisher, data di uscita e valutazioni.

- Gestire wishlist personalizzate
    - Salvare e ordinare i giochi desiderati, permettendo di visualizzare titolo, copertina, descrizione, valutazioni e data di uscita.

- Pianificare sincronizzazioni regolari
    - Grazie a un job scheduler basato su ARQ e Redis, è possibile programmare aggiornamenti su richiesta o a intervalli prestabiliti, mantenendo sempre allineati i dati locali con quelli delle piattaforme esterne.

In questo modo, GamesTrack diventa il punto di riferimento per collezionisti e appassionati che vogliono avere sotto controllo ogni aspetto della propria esperienza videoludica.

Tool Utilizzati

La piattaforma è stata sviluppata sfruttando un insieme di tecnologie e librerie open-source, ognuna scelta per garantire massima efficienza, scalabilità e facilità di manutenzione.

Interfaccia utente (frontend)
Per la realizzazione dell'interfaccia grafica e' stato adottato React, un framework JavaScript moderno e modulare che ci ha permesso di creare componenti riutilizzabili e altamente

performanti. Grazie a React, gli utenti possono consultare in modo intuitivo e immediato con filtri e ordinamenti:

- Una **Dashboard** iniziale riassuntiva di vari dati aggregati dalla piattaforma
- la **lista completa dei videogiochi** presenti sulla piattaforma;
- la **wishlist**, ovvero l'elenco dei titoli che desiderano acquistare in futuro;
- la propria **libreria**, ovvero l'elenco di giochi posseduti;
- ed infine sezioni **categoriche** su console, aziende, generi e modalita' di gioco

L'interfaccia è stata progettata per offrire un'esperienza chiara e coerente su dispositivi desktop e mobile, integrando stili CSS modulari e librerie di componenti (ad esempio Tailwind CSS) per garantire coerenza visiva e tempi di caricamento ridotti.

Logica di business e API (backend)
Il cuore delle funzionalità server-side è implementato con **FastAPI**, un framework Python leggero e ad alte prestazioni che espone API RESTful veloci, documentate automaticamente tramite OpenAPI e facilmente estendibili. Tra le principali operazioni esposte al frontend troviamo:

1. **Ricerca e recupero dei giochi**: endpoint per interrogare database esterni e restituire informazioni aggiornate sui titoli.
2. **Gestione della libreria personale**: operazioni CRUD di aggiunta, rimozione e aggiornamento di giochi nella propria raccolta o nella wishlist.
3. **Gestione dei Metadata:** in caso di errore nel recupero automatico dei metadata viene anche esposta una funzionalita' per impostare manualmente i metadata corretti
4. **Sincronizzazione su richiesta**: un job che viene avviato dal frontend e successivamente eseguito in background tramite una coda implementata con ARQ e Redis, garantendo processi asincroni affidabili e scalabili.

Integrazione con servizi esterni
Per interfacciarsi con le Web API di PSN, Steam e IGDB (Internet Game Database) utilizziamo specifici wrapper Python che semplificano autenticazione, rate limiting e parsing delle risposte JSON. Queste librerie ci consentono di:

- ottenere la lista dei titoli posseduti cosi' come le statistiche di gioco e i trofei da PlayStation Network e Steam;

- scaricare metadati dettagliati (descrizioni, copertine, generi, sviluppatori, date di rilascio) da IGDB;
- scaricare dati categorici (aziende, generi, console e modalita' di gioco) da IGDB.

Persistenza dei dati
Tutti i dati vengono memorizzati in un database NoSQL MongoDB, scelto per la sua flessibilità nel gestire documenti semi-strutturati e la facilità di scalare orizzontalmente. L'interazione tra l'applicativo Python e MongoDB avviene tramite PyMongo, che ci offre

un’API semplice ed efficiente per operazioni CRUD, query complesse di aggregazione e indicizzazione avanzata.

Insieme, queste tecnologie formano un’architettura moderna, modulare e performante, capace di offrire agli appassionati di gaming una piattaforma completa per gestire, consultare e scoprire nuovi titoli in modo rapido e intuitivo.

Api Platforms

L'applicativo GamesTrack integra tre principali API esterne per fornire un sistema completo di tracciamento e sincronizzazione dei dati di gioco:

Steam Web API
La Steam Web API viene utilizzata per sincronizzare i dati dei giochi posseduti dagli utenti sulla piattaforma Steam. L'implementazione utilizza diversi endpoint dell'API Steam:

- **ISteamUser/GetPlayerSummaries/v2/**:
    - per ottenere le informazioni del profilo utente Steam
- **IPlayerService/GetOwnedGames/v1/**:
    - per recuperare la lista completa dei giochi posseduti con ore di gioco
- **ISteamUserStats/GetSchemaForGame/v2/**:
    - per ottenere il numero totale di achievement disponibili per ogni gioco
- **ISteamUserStats/GetPlayerAchievements/v1/**:
    - per recuperare gli achievement ottenuti dall'utente

La sincronizzazione Steam gestisce automaticamente la conversione delle ore di gioco da minuti a ore, calcola le percentuali di completamento degli achievement e implementa un sistema robusto di gestione degli errori SSL e rate limiting per evitare blocchi da parte dell'API.

PSNAWP (PlayStation Network API)
Per l'integrazione con PlayStation Network, l'applicativo utilizza la libreria **PSNAWP**. Questa API permette di:

- Recuperare la lista completa dei giochi PlayStation (PS3, PS4, PS5) dell'utente con annesse informazioni di gioco (numero di ore di gioco, numero di trofei guadagnati)
- Ottenere statistiche dettagliate sui trofei per ogni gioco (Platino, Oro, Argento, Bronzo)

I dati vengono elaborati e consolidati utilizzando pandas DataFrame per unire le informazioni sui giochi con quelle sui trofei, garantendo una vista completa delle statistiche PlayStation dell'utente.

IGDB API (Internet Game Database)

L'IGDB API rappresenta la fonte principale per i metadati dei videogiochi e questa integrazione fornisce:

- **Metadati** completi dei giochi: nome, descrizione, generi, piattaforme supportate, date di rilascio
- **Informazioni sulle aziende**: sviluppatori e publisher
- **Contenuti multimediali**: copertine, screenshot e artwork
- **Dati strutturati**: generi, modalità di gioco, piattaforme e valutazioni

La classe implementa un sistema di autenticazione automatica OAuth2 con Twitch (proprietaria di IGDB) che gestisce il rinnovo automatico dei token di accesso.

L'API viene utilizzata sia per arricchire i dati sincronizzati da Steam e PSN con metadati completi, sia per permettere agli utenti di cercare e aggiungere manualmente giochi alla propria libreria o wishlist attraverso l'interfaccia web.

Qui di seguito un esempio di metadata forniti da igdb:

JSON


```
[
  {
    "id": 1942,
    "name": "The Witcher 3: Wild Hunt",
    "summary": "The Witcher 3: Wild Hunt is a story-driven, next-generation
open world role-playing game set in a visually stunning fantasy universe full
of meaningful choices and impactful consequences.",
    "storyline": "The war with Nilfgaard obliterated the old order...",
    "first_release_date": 1431993600,
    "total_rating": 93.2,
    "total_rating_count": 156,
    "genres": [
      {
        "id": 12,
        "name": "Role-playing (RPG)"
      },
      {
        "id": 31,
        "name": "Adventure"
      }
    ],
    "platforms": [
      {
        "id": 6,
        "name": "PC (Microsoft Windows)",
        "abbreviation": "PC"
      },
      {
```

```
        "id": 48,
        "name": "PlayStation 4",
        "abbreviation": "PS4"
      }
    ],
    "involved_companies": [
      {
        "company": {
          "id": 908,
          "name": "CD Projekt RED"
        },
        "developer": true,
        "publisher": false
      }
    ],
    "cover": {
      "id": 82192,
      "url": "//images.igdb.com/igdb/image/upload/t_thumb/co1sf8.jpg",
      "checksum": "co1sf8",
      "width": 264,
      "height": 374
    },
    "screenshots": [
      {
        "id": 438922,
        "url": "//images.igdb.com/igdb/image/upload/t_thumb/sc9joa.jpg",
        "checksum": "sc9joa",
        "width": 1920,
        "height": 1080
      }
    ],
    "game_modes": [
      {
        "id": 1,
        "name": "Single player"
      }
    ]
  }
]
```


Metodologia Utilizzata

Approccio Metodologico
Lo sviluppo di GamesTrack ha seguito un approccio API-first e microservices-oriented, privilegiando la separazione delle responsabilità e la scalabilità orizzontale. La metodologia adottata si basa su principi di clean architecture e domain-driven design, dove ogni

componente del sistema ha una responsabilità specifica e comunica attraverso interfacce ben definite.

Architettura di Sistema
L'applicativo implementa un'architettura a tre livelli con pattern Event-Driven per la gestione asincrona delle sincronizzazioni:

1. **Livello di Presentazione (Frontend React)**
    - Single Page Application (SPA) sviluppata in React con routing client-side
    - State management centralizzato per la gestione dello stato dell'applicazione
    - Componenti modulari riutilizzabili per la visualizzazione dei dati di gioco
    - Responsive design ottimizzato per desktop e dispositivi mobili
    - Comunicazione con il backend tramite REST API e gestione asincrona delle chiamate

2. **Livello di Business Logic (Backend FastAPI)**
    - Il core applicativo è implementato come API REST utilizzando FastAPI, sfruttando le seguenti caratteristiche tecniche:
        - **Dependency Injection**: utilizzo del sistema Depends() di FastAPI per la gestione centralizzata delle connessioni database e dell'autenticazione utente
        - **Middleware personalizzato**: per CORS, logging delle richieste e gestione degli errori
        - **Authentication Bearer Token**: sistema JWT per l'autenticazione stateless degli utenti
        - **OpenAPI/Swagger**: documentazione automatica delle API con schema interattivo

3. **Livello di Persistenza e Worker Asincroni**
    - **MongoDB**: database NoSQL document-oriented per la persistenza flessibile dei dati semi-strutturati provenienti da diverse API
        - Schema dinamico: collezioni users, games, games_user, platforms_users, schedules con strutture adattabili
    - **Worker pool asincroni**: gestione dei job di sincronizzazione tramite code Redis/ARQ per elaborazione non-bloccante

Gestione Asincrona e Sincronizzazione
Il sistema implementa un message queue pattern per la gestione delle sincronizzazioni:

- **Job Scheduling**: creazione di task asincroni per ogni utente/piattaforma
- **Worker Processing**: elaborazione dei dati attraverso worker pool dedicati
- **Data Normalization**: trasformazione e pulizia dei dati prima della persistenza
- **Progress Tracking**: monitoraggio real-time dello stato delle sincronizzazioni

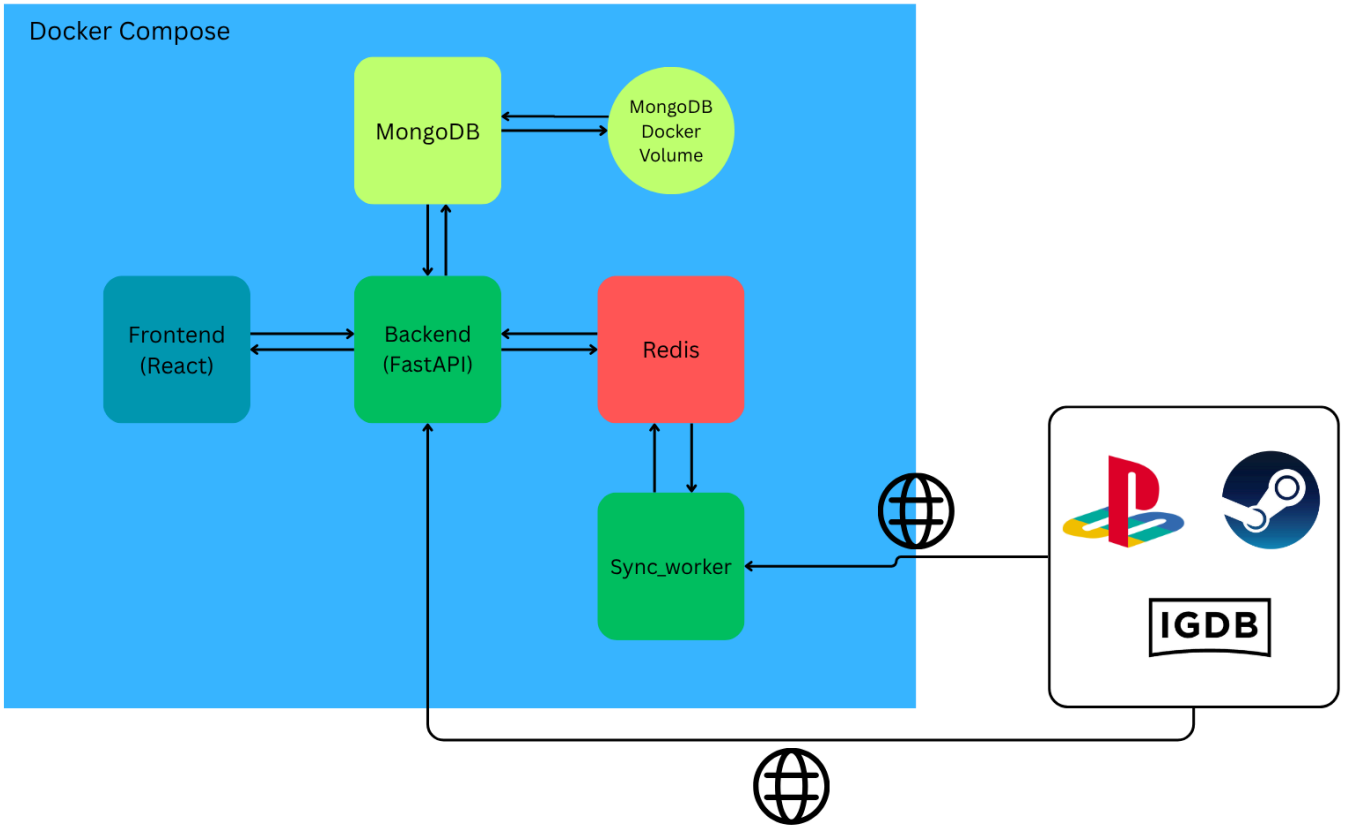
Containerizzazione e Deployment
L'intera piattaforma è containerizzata utilizzando **Docker Compose** con:

- Multi-stage builds per ottimizzazione delle immagini
- Docker Compose per orchestrazione multi-container (backend, frontend, database, worker)
- Environment-based configuration tramite variabili d'ambiente

Data Flow e Pipeline
Il flusso dei dati segue un pattern ETL (Extract, Transform, Load):

- **Extract**: recupero dati dalle API esterne (Steam, PSN, IGDB)
- **Transform**: normalizzazione, pulizia e arricchimento dei metadati
- **Load**: persistenza nel database MongoDB con relazioni cross-reference
- **Serve**: esposizione tramite API REST per il frontend

Questa architettura garantisce scalabilità, maintainability e extensibility, permettendo l'aggiunta di nuove piattaforme gaming (Xbox, Epic Games Store, Ubisoft Play, GoG, ecc.…) senza modifiche strutturali significative al core del sistema.Ar



(Architettura della piattaforma sviluppata)

Struttura del Database

GamesTrack utilizza MongoDB come database principale per gestire la complessità dei dati di gaming cross-platform, sfruttando la natura document-oriented per gestire strutture dati eterogenee provenienti da diverse API esterne.

Il database **game_tracker** è strutturato attorno a sei collezioni principali che modellano le relazioni tra utenti, giochi, piattaforme e sincronizzazioni.

La collezione **users** gestisce l'autenticazione con username e password hashate, mentre **platforms-users** crea il ponte tra gli account utente e le loro credenziali delle diverse piattaforme gaming (Steam, PSN).

Il cuore del sistema è rappresentato dalla collezione **games**, che funge da repository centrale per tutti i metadati dei videogiochi arricchiti tramite IGDB API, includendo informazioni dettagliate come generi, piattaforme supportate, sviluppatori, copertine e screenshot.

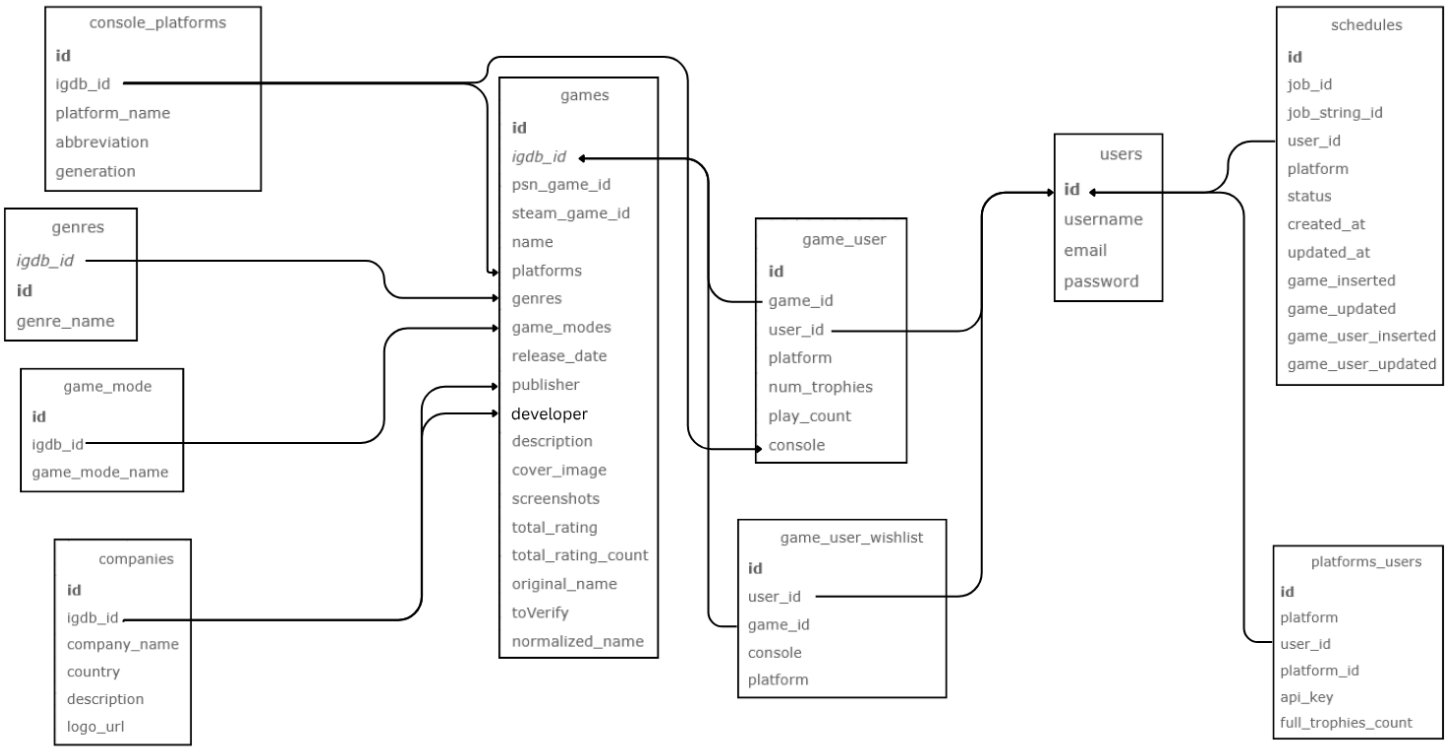
La relazione many-to-many tra utenti e giochi viene gestita attraverso **game_user**, che traccia il possesso di ogni titolo per utente e piattaforma specifica, memorizzando statistiche personali come ore giocate e trofei ottenuti.

Il sistema implementa operazioni MongoDB complesse attraverso **aggregation pipeline** per unire dati da collezioni multiple, come quando costruisce la libreria completa di un utente effettuando il join di game_user con games e console_platforms per fornire una vista arricchita con metadati completi.

Le operazioni di sincronizzazione sfruttano operazioni in bulk per ottimizzare le performance durante l'aggiornamento massivo dei dati provenienti dalle API esterne, evitando di effettuare un numero eccessivo di operazioni di inserimento o update e lasciando cosi' il DB libero per ulteriori operazioni.

La collezione **schedules** traccia tutti i job di sincronizzazione asincroni con informazioni dettagliate su stato, durata ed eventuali errori, permettendo al sistema di monitorare e debuggare efficacemente i processi di background.

L'indicizzazione strategica utilizza indici composti per ottimizzare le query più frequenti, come la ricerca per **user_id** e **platform** combinati, mentre indici unique garantiscono l'integrità referenziale evitando duplicati.



(Struttura del MongoDB creato per la piattaforma)

Questa architettura MongoDB permette a GamesTrack di gestire milioni di record mantenendo performance elevate grazie alla natura schemaless che si adatta perfettamente alla variabilità dei dati gaming e alle query complesse necessarie per aggregare statistiche cross-platform in tempo reale.

Sviluppi Futuri

Il progetto GamesTrack presenta un'architettura solida e modulare che facilita l'implementazione di nuove funzionalità per espandere ulteriormente l'ecosistema di gaming tracking. Diverse aree di sviluppo future potrebbero significativamente arricchire l'esperienza utente:

Integrazione Xbox Live
L'aggiunta del supporto per Xbox Live rappresenta un'estensione naturale della piattaforma. Utilizzando l'Xbox Live API, sarebbe possibile sincronizzare i dati degli achievement Xbox, le ore di gioco e le statistiche dei giochi da Xbox Live. L'implementazione seguirebbe il pattern già stabilito per Steam e PSN, creando un nuovo modulo che gestisca l'autenticazione OAuth2 con Microsoft e la sincronizzazione dei dati gamertag.
La struttura MongoDB esistente supporta già console multiple attraverso il campo consoles nella collezione games, rendendo l'integrazione Xbox trasparente.

Dettaglio Trofei per Gioco
Attualmente il sistema traccia solo il numero totale di trofei/achievement per gioco. Un'implementazione futura potrebbe estendere questo sistema creando una nuova collezione game_achievements che memorizzi ogni singolo trofeo con dettagli come nome, descrizione, rarità, data di ottenimento e icona. Questo permetterebbe di creare pagine dettagliate per ogni gioco nella libreria utente, mostrando progressi specifici, trofei mancanti e statistiche di completamento avanzate.

Integrazione Launcher PC Aggiuntivi
L'espansione verso altri launcher PC come Epic Games Store, GOG Galaxy, Origin/EA App e Battle.net amplierebbe significativamente la copertura della piattaforma. Ogni launcher richiederebbe un approccio specifico: Epic Games attraverso le loro Web API, GOG tramite l'integrazione Galaxy, mentre Origin e Battle.net potrebbero richiedere altre tecniche.

Tracking Prezzi per Wishlist
Una funzionalità di monitoraggio prezzi trasformerebbe la wishlist da semplice lista desideri a strumento di acquisto intelligente. Integrando API di aggregatori di prezzi come IsThereAnyDeal, SteamDB o PSPrices, il sistema potrebbe tracciare storici prezzi, notificare sconti e suggerire i migliori momenti per l'acquisto. L'implementazione richiederebbe una nuova collezione price_history e un worker dedicato per il polling periodico dei prezzi.

Statistiche Avanzate e Analytics
Il sistema potrebbe evolvere verso un vero e proprio analytics engine per gaming personale. Funzionalità come timeline delle sessioni di gioco, analisi dei pattern di gaming (generi preferiti, picchi di attività, stagionalità), confronti con altri utenti e achievement rarity ranking potrebbero essere implementate espandendo la dashboard utente.

Considerazioni Architetturali
Tutte queste implementazioni beneficerebbero dell'architettura containerizzata esistente nel docker-compose.yaml, che permette scaling orizzontale dei worker e separazione logica dei servizi. Il database MongoDB con la sua struttura flessibile basata sui documenti supporta naturalmente l'evoluzione dello schema senza migrazioni complesse, mentre il sistema di code Redis/ARQ garantisce la gestione efficiente di job di sincronizzazione anche con volumi di dati crescenti.

L'obiettivo finale sarebbe trasformare GamesTrack da semplice aggregatore di dati gaming a ecosistema completo per il gaming enthusiast, fornendo insights actionable, ottimizzazione degli acquisti e social features per condividere achievements e scoperte con la community.

Conclusioni

GamesTrack si propone come una soluzione innovativa e completa alla crescente frammentazione dell’esperienza videoludica moderna, offrendo un sistema flessibile, estensibile e orientato all’utente per la centralizzazione dei propri dati cross-platform. Attraverso un’architettura robusta, tecnologie moderne e un’integrazione profonda con le principali API di settore, la piattaforma riesce a colmare il divario tra le diverse ecosfere del

gaming, fornendo un’interfaccia unica per esplorare, organizzare e monitorare la propria collezione videoludica. Le prospettive future — dall’integrazione di nuove piattaforme all’analisi avanzata dei dati — delineano un percorso evolutivo ambizioso che mira a trasformare GamesTrack da semplice strumento di aggregazione a vero e proprio companion intelligente per ogni appassionato di videogiochi.