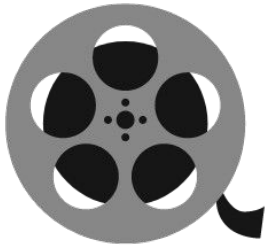


RATED

•COMMUNITY REVIEWS•

Presentazione Progetto IS 17/01/2025

Bruno Nesticò e Francesco Rao

INDICE DELLA PRESENTAZIONE

RATED
•COMMUNITY REVIEWS•

RATED
•COMMUNITY REVIEWS•

01 INTRODUZIONE

OBIETTIVO DEL PROGETTO

RATED
•COMMUNITY REVIEWS•

Il nostro team si propone di progettare e sviluppare un'applicazione basata sulle specifiche richieste di un potenziale committente.

Per garantire un processo di sviluppo efficiente, adottiamo un approccio modulare, suddividendo il sistema in componenti gestibili singolarmente. Questo metodo consente di ridurre la complessità complessiva e di ottimizzare il lavoro in ogni fase dello sviluppo.

AVVIO DEL PROGETTO

RATED
•COMMUNITY REVIEWS•

Lo sviluppo dell'applicazione inizia con l'analisi delle esigenze del committente, che presenta un problema da risolvere. Questo problema viene formalizzato attraverso il ***Problem Statement***, un documento che delinea:

- La natura del problema da affrontare,
- I vincoli di progetto da rispettare,
- I requisiti generali dell'architettura del sistema.

Una volta definito il *Problem Statement*, il team può procedere con l'analisi dettagliata e la progettazione del sistema, garantendo una soluzione efficace e aderente alle necessità del cliente.

RATED
•COMMUNITY REVIEWS•

02 PROBLEM STATEMENT

DOMINIO DEL PROBLEMA

RATED
•COMMUNITY REVIEWS•

Il panorama attuale delle recensioni cinematografiche online è frammentato e inefficiente. Gli appassionati di cinema spesso faticano a trovare recensioni di qualità, mentre i recensori non hanno un modo efficace per far emergere i loro contenuti.

Le piattaforme esistenti, come social network o siti generalisti (IMDb, Rotten Tomatoes), non offrono strumenti adeguati per valorizzare le recensioni migliori, limitandosi ad aggregare punteggi numerici senza fornire un vero **sistema di valutazione della qualità dei contributi**.

DOMINIO DEL PROBLEMA

RATED
•COMMUNITY REVIEWS•

Rated nasce per risolvere questo problema, fornendo una piattaforma **dedicata esclusivamente alle recensioni cinematografiche**, con un sistema di valutazione basato sul punteggio reputazionale. Grazie a Rated:

- Gli utenti possono **scrivere recensioni strutturate**, includendo un titolo, un testo analitico e una valutazione.
- Le recensioni possono ricevono **upvote/downvote** da parte della community, evidenziando i contributi più apprezzati.
- In un contesto pienamente social, ogni recensore può accumulare **punteggio reputazionale**, visibile nel profilo.
- Un **sistema di moderazione** garantisce un ambiente rispettoso e privo di contenuti tossici, attraverso strumenti di moderazione e segnalazioni da parte degli utenti.
- I gestori del catalogo possono mantenere **un database di film sempre aggiornato**, offrendo un'esperienza strutturata e organizzata per gli utenti.

RATED
•COMMUNITY REVIEWS•

03 REQUIREMENTS ELICITATION AND ANALYSIS

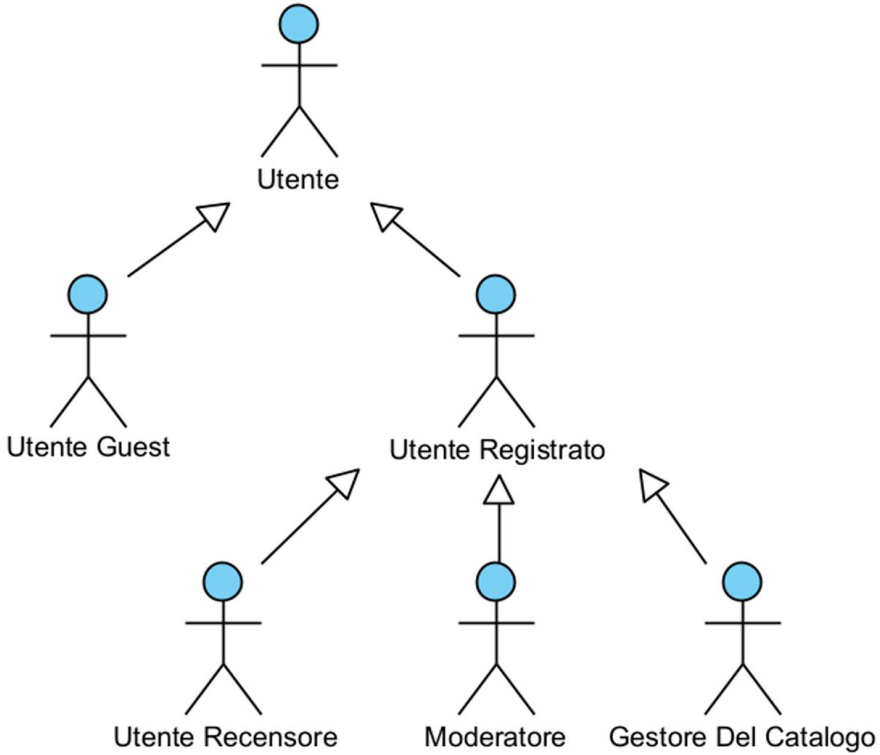
ATTIVITÀ SVOLTE IN FASE DI REQUIREMENTS ELICITATION

- Produzione del Requirement Specification
  - Produzione dello Use Case Model
    - Identificazione degli attori del Sistema
    - Identificazione degli scenari
    - Identificazione dei casi d'uso
    - Identificazione delle relazioni tra i casi d'uso

  - Identificazione dei requisiti **funzionali** e **non funzionali**

ATTORI DEL SISTEMA

RATED
•COMMUNITY REVIEWS•



La gerarchia di attori mostrata serve ad illustrare la gerarchia tra le vari tipologie di utenti.
Tuttavia gli unici utenti effettivi della piattaforma saranno: Guest, Recensori, Moderatori e Gestori del catalogo.

SCENARI DI UTILIZZO FONDAMENTALI

RATED
•COMMUNITY REVIEWS•

Pubblicazione di una recensione
• **Utenti coinvolti**: Utente Recensore
• **Descrizione:** Un utente autenticato desidera condividere una recensione su un film che ha visto. L'utente seleziona il film dal catalogo o utilizza la barra di ricerca, quindi accede alla scheda del film. Da qui, l'utente clicca su "Scrivi una recensione", inserisce il contenuto e assegna una valutazione numerica. Una volta pubblicata, la recensione sarà visibile agli altri membri della community.

Valutazione di una recensione
• **Utenti coinvolti**: Utente Recensore
• **Descrizione**: Un utente registrato legge una recensione di un altro membro della community e desidera esprimere il proprio consenso o dissenso. L'utente può cliccare sul pulsante "Vota" per aggiungere un voto positivo o negativo alla recensione. Il sistema aggiorna il punteggio cumulativo della recensione e il punteggio reputazionale dell'autore.

SCENARI DI UTILIZZO FONDAMENTALI

RATED
•COMMUNITY REVIEWS•

Moderazione delle recensioni
• **Utenti coinvolti:** Moderatore
• **Descrizione:** Un moderatore accede all'area di moderazione per verificare il contenuto delle recensioni segnalate. Il moderatore ha la possibilità di rimuovere recensioni che violano le linee guida della community oppure di approvarla rimuovendone le segnalazioni.

Gestione del catalogo film
• **Utenti coinvolti:** Gestore del catalogo
• **Descrizione:** Un gestore accede alla pagina del catalogo dei film e interagendo con gli opportuni pulsanti di funzione può aggiungere nuove schede per film non ancora presenti nel catalogo o aggiornare le informazioni esistenti.

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

Relativi a tutti gli utenti

• RF_1 **Visualizzazione Catalogo Film** (Priorità alta) Gli utenti devono poter visualizzare l'elenco dei film presenti nel catalogo.

• RF_2 **Ricerca film** (Priorità alta) Gli utenti potranno cercare un film specifico per nome tramite la barra di ricerca presente nell'header.

• RF_3 **Ordinare film catalogo** (Priorità Media) Gli utenti potranno decidere in che modo ordinare i film mostrati nel catalogo (es. rating decrescente, data d'uscita etc.)

• RF_4 **Dettagli del Film** (Priorità alta) Gli utenti devono poter accedere ai dettagli di ciascun film cliccando sul titolo. Nella scheda informativa devono essere presenti informazioni come la locandina originale, il regista, la data di uscita, il genere e le recensioni relative al film.

• RF_5 **Ordinare recensioni film** (Priorità Media) Gli utenti potranno decidere in che modo ordinare le recensioni mostrate nella scheda di un film che stanno visualizzando (valutazione film crescente/decrescente etc.)

• RF_6 **Visualizzazione pagina di un Utente** (Priorità alta) Gli utenti devono poter visualizzare la pagina utente legata ad una recensione specifica visualizzata. La pagina utente mostrerà l'username, la biografia, le recensioni pubblicate e il punteggio reputazionale dell'utente.

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

Relativi ad Utente Guest

• **RF_7 Registrazione** (Priorità alta) Gli utenti Guest devono avere la possibilità di creare un account come recensori sulla piattaforma, inserendo le opportune credenziali.

Relativi a tutti agli utenti Registrati

• **RF_8 LogIn** (Priorità alta) Gli utenti registrati devono avere la possibilità di autenticarsi tramite una pagina di LogIn per accedere al sito.

• **RF_9 LogOut** (Priorità alta) Gli utenti Registrati devono avere la possibilità di eseguire il LogOut dal sito.

• **RF_10 Modifica Password** (Priorità alta) Gli utenti Registrati devono poter modificare la password del proprio account.

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

Relativi ad Utente Recensore

• **RF_11 Visualizzazione Profilo Personale** (Priorità alta) Gli Utenti Recensori devono poter visualizzare la propria pagina utente personale, contentente le informazioni del proprio account, i pulsanti per effettuare eventuali modifiche e tutte le recensioni pubblicate.

• **RF_12 Modifica informazioni del Profilo Utente** (Priorità alta) Gli utenti Recensori devono poter modificare le informazioni del proprio account come username e biografia.

• **RF_13 Aggiunta Recensioni** (Priorità alta) Gli utenti Recensori devono poter aggiungere recensioni sui film presenti nel catalogo. Le recensioni includeranno un titolo riassuntivo, un testo descrittivo e una valutazione in stelle. Se l'utente ha ricevuto uno stato di Limited, al momento dell'inserimento di una recensione il sistema deve mostrare un alert che lo notifichi dell'impossibilità di farlo. Per ogni utente sarà possibile scrivere una sola recensione per film.

• **RF_14 Rimozione di una Recensione** (Priorità Media) Gli utenti Recensori devono poter rimuovere delle loro vecchie recensioni

...

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

• **RF_15 Visualizzazione notifica di Moderazione** (Priorità alta) Se l'utente Recensore ha ricevuto uno stato di Warned o Limited, al momento del login il sistema deve mostrare un alert che lo notifichi del suo stato.

• **RF_16 Votazione delle Recensioni** (Priorità alta) Gli utenti Recensori dovranno poter esprimere la propria approvazione (upvote) o disapprovazione (downvote) riguardo le recensioni degli altri. Ogni recensione avrà un punteggio basato sui voti ricevuti. La recensione accumulerà un punteggio reputazionale, che aumenterà con voti positivi (+1) e diminuirà con voti negativi (-1). Ogni utente potrà esprimere un solo voto per ogni recensione.

• **RF_17 Segnalazione di una Recensione** (Priorità alta) Gli utenti Recensori dovranno poter segnalare una recensione ritenuta inappropriata.

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

Relativi al Gestore del Catalogo

• **RF_18 Aggiunta Film** (Priorità alta) Gli amministratori devono poter aggiungere nuovi titoli al catalogo dei film. Il sistema validerà i dati inseriti (es. titolo, anno, genere, durata) e notificherà eventuali errori di compilazione, richiedendo correzioni prima del salvataggio.

• **RF_19 Modifica Film** (Priorità alta) Gli amministratori devono poter modificare i dettagli di un film esistente nel catalogo. Il sistema validerà le modifiche e notificherà eventuali errori, impedendo il salvataggio di dati errati.

• **RF_20 Rimozione Film** (Priorità alta) Gli amministratori devono poter rimuovere un titolo dal catalogo. Il sistema richiederà una conferma prima della cancellazione per evitare errori accidentali.

REQUISITI FUNZIONALI

RATED
•COMMUNITY REVIEWS•

Relativi al Moderatore

• **RF_21 Accesso all'area di moderazione** (Priorità alta) I Moderatori devono poter accedere ad un'area dedicata per eseguire le funzioni di moderazione

• **RF_22 Eliminare Recensione** (Priorità alta) I Moderatori devono poter eliminare una recensione segnalata da altri utenti, qualora ritengano che questa violi le linee guida della community. Quando una recensione viene rimossa, il sistema aggiorna automaticamente lo stato dell'utente che l'ha pubblicata a Warned, indicando che ha ricevuto un avvertimento. In caso di recidiva, il suo stato diventa Limited, impedendogli di aggiungere ulteriori recensioni. Tutte queste modifiche vengono gestite in modo automatico dal sistema, che invia una notifica all'utente per informarlo della decisione e delle relative conseguenze.

• **RF_23 Approvare Recensione** (Priorità alta) I Moderatori devono poter approvare una recensione eliminandone le relative segnalazioni in caso ritengano che rispetti i criteri di accettabilità della piattaforma.

REQUISITI NON FUNZIONALI

RATED
•COMMUNITY REVIEWS•

FURPS

Usability

• **NFR1:** Il sistema deve fornire messaggi di errore chiari e dettagliati in caso di input non validi.

• **NFR2:** Il sistema deve avere un design intuitivo che consenta una navigazione fluida e un rapido accesso alle principali funzionalità.

• **NFR3:** Il sistema deve essere responsivo e ottimizzato per dispositivi mobili, tablet e desktop.

REQUISITI NON FUNZIONALI

RATED
•COMMUNITY REVIEWS•

FURPS

Reliability

• **NFR4**: Il sistema deve garantire che tutte le password rispettino criteri di sicurezza, La password deve contenere tra 8 e 64 caratteri, almeno una lettera minuscola, una maiuscola, un numero e un carattere speciale (@, $, !, %, ?, &, .).

• **NFR5**: Il sistema deve rendere sicuri i campi di input presenti nelle varie pagine della piattaforma.

REQUISITI NON FUNZIONALI

RATED
•COMMUNITY REVIEWS•

FURPS

Performance
• **NFR6**: Il sistema deve garantire tempi di caricamento inferiori a 2 secondi per ogni pagina o funzione principale.

Supportability
• **NFR7**: Il codice deve essere strutturato in modo da permettere futuri aggiornamenti senza compromettere l'integrità del sistema.

DALLO SCENARIO AL CASO D'USO

Pubblicazione di una recensione

- **Utenti coinvolti:** Utente Recensore
- **Descrizione:** Un utente autenticato desidera condividere una recensione su un film che ha visto. L'utente seleziona il film dal catalogo o utilizza la barra di ricerca, quindi accede alla scheda del film. Da qui, l'utente clicca su "Scrivi una recensione", inserisce il contenuto e assegna una valutazione numerica. Una volta pubblicata, la recensione sarà visibile agli altri membri della community.

UC13-Pubblicazione di una recensione

Name	Pubblicazione Recensione
Utenti Partecipanti	Utente Recensore
Entry conditions	L'utente è autenticato nel sistema e visualizza la scheda di un film.
Flow degli eventi	1. L'utente clicca sul pulsante "Scrivi una recensione"...
…	…

ESEMPIO DI CASO D'USO COMPLETO

RATED
•COMMUNITY REVIEWS•

UC13-Pubblicazione di una recensione
Nome: Pubblicazione Recensione

Utenti Partecipanti: Utente Recensore

Entry Conditions: L'utente è autenticato nel sistema e visualizza la scheda di un film.

Flow degli eventi:
1. L'utente clicca sul pulsante "Scrivi una recensione".
2. Il sistema visualizza il form di recensione, composto da un campo per il testo, uno per il titolo e un'opzione per la valutazione numerica (es. da 1 a 5 stelle).
3. L'utente compila il campo di testo con la sua opinione e seleziona il punteggio.
4. L'utente clicca su "Pubblica".
5. Il sistema valida la recensione.
6. Il sistema memorizza la recensione e la rende visibile agli altri utenti.

Exit Conditions: L'utente ha pubblicato la recensione e rimane nella schermata del film nella quale essa è visibile.

Eccezioni/Flussi Alternativi: Al punto 5, se il sistema rileva che i dati inseriti non sono validi, avviene EUC06.

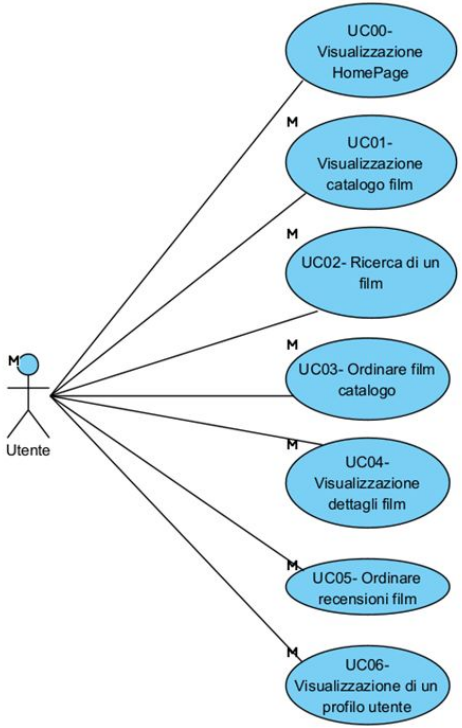
CASI D'USO INDIVIDUATI

RATED
•COMMUNITY REVIEWS•

UC00 - Visualizzazione Home Page
UC01 - Visualizzazione Catalogo Film
UC02 - Ricerca di un film
UC03 - Ordinare film catalogo
UC04 - Visualizzazione Dettagli Film
UC05 - Ordinare recensioni film
UC06 - Visualizzazione di un profilo utente
UC07 - SignUp
UC08 - LogIn
UC09 - LogOut
UC10 - Modifica Password
UC11 - Visualizzazione profilo personale
UC12 - Modifica Informazioni Utente
UC13 - Pubblicazione di una recensione
UC14 - Valutazione di una recensione
UC15 - Segnalazione di una recensione
UC16 - Rimozione di una recensione
UC17 - Accesso all'area di moderazione
UC18 - Approva recensione
UC19 - Elimina recensione
UC20 - Aggiunta di un film al catalogo
UC21 - Rimozione di un film dal catalogo
UC22 - Modifica di un film del catalogo

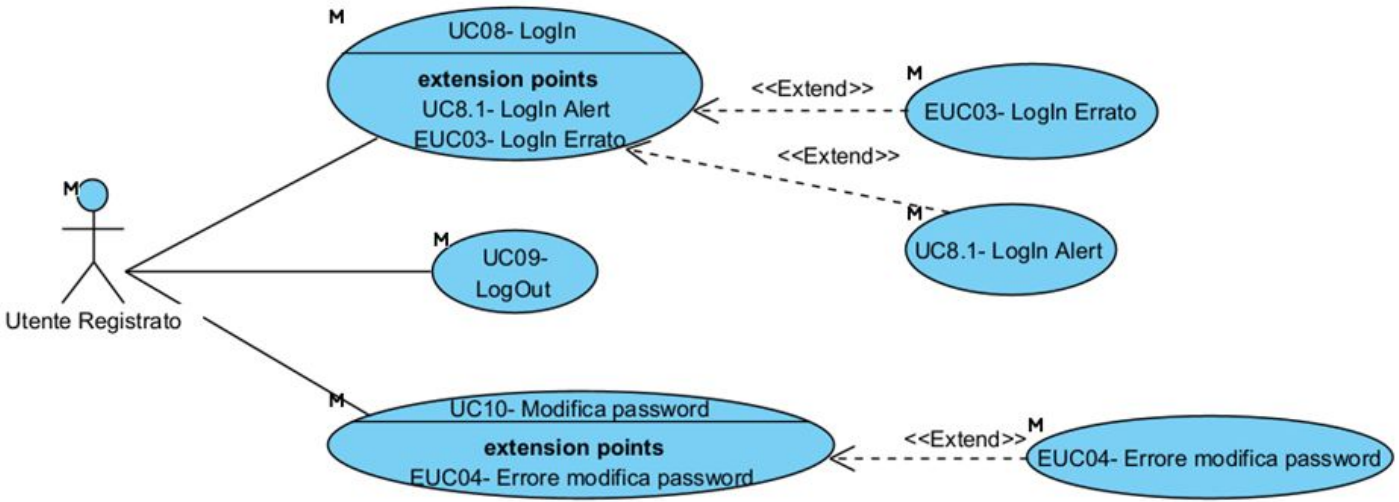
USA CASE DIAGRAM: UTENTE

RATED
•COMMUNITY REVIEWS•



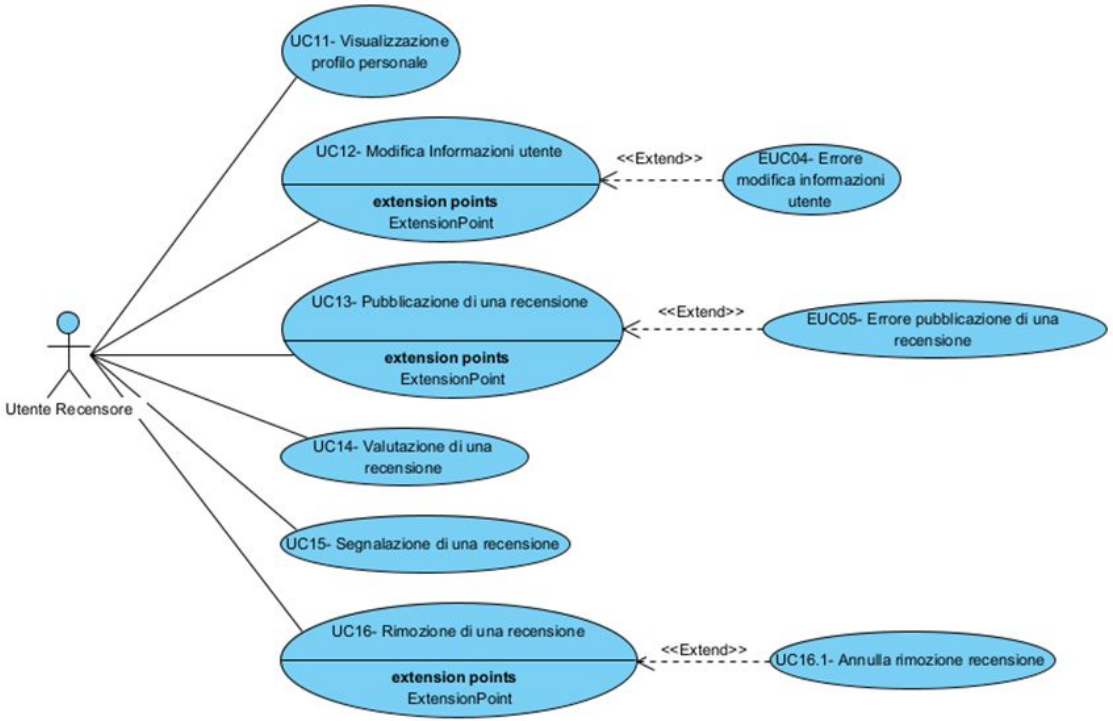
USE CASE DIAGRAM: UTENTE GUEST

USE CASE DIAGRAM: UTENTE REGISTRATO



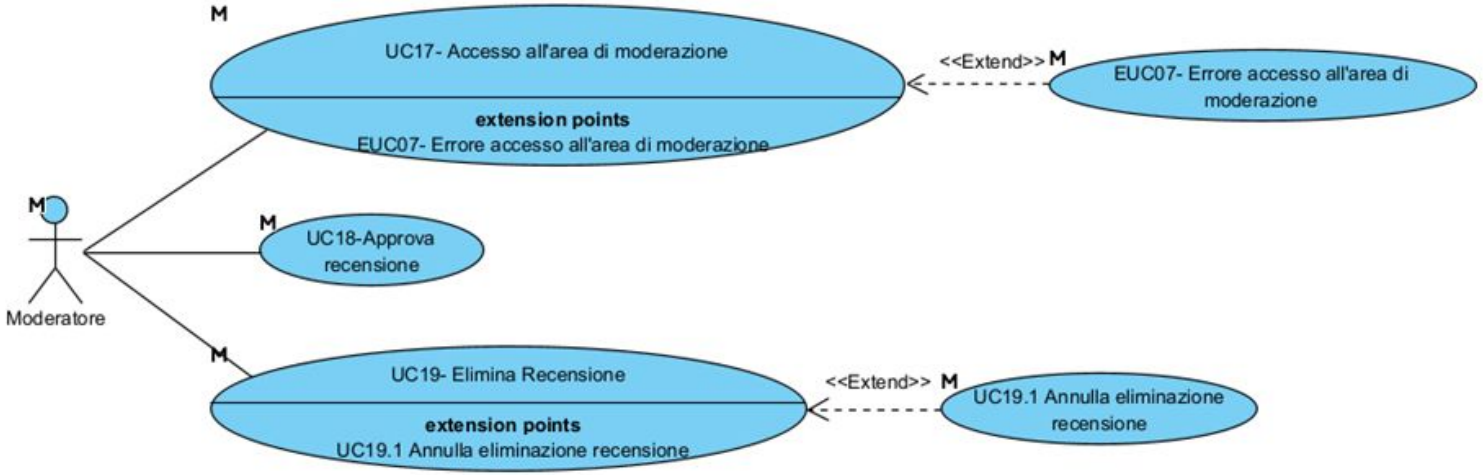
USE CASE DIAGRAM: UTENTE RECENSORE

RATED
•COMMUNITY REVIEWS•



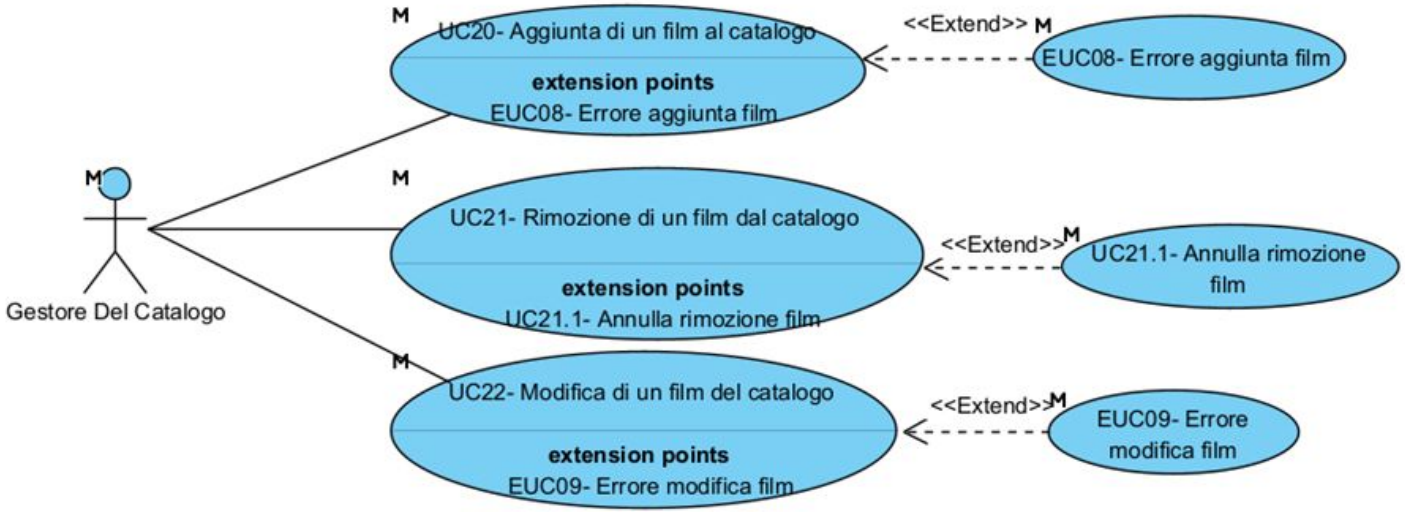
USE CASE DIAGRAM: MODERATORE

RATED
•COMMUNITY REVIEWS•



USE CASE DIAGRAM: MODERATORE

RATED
•COMMUNITY REVIEWS•



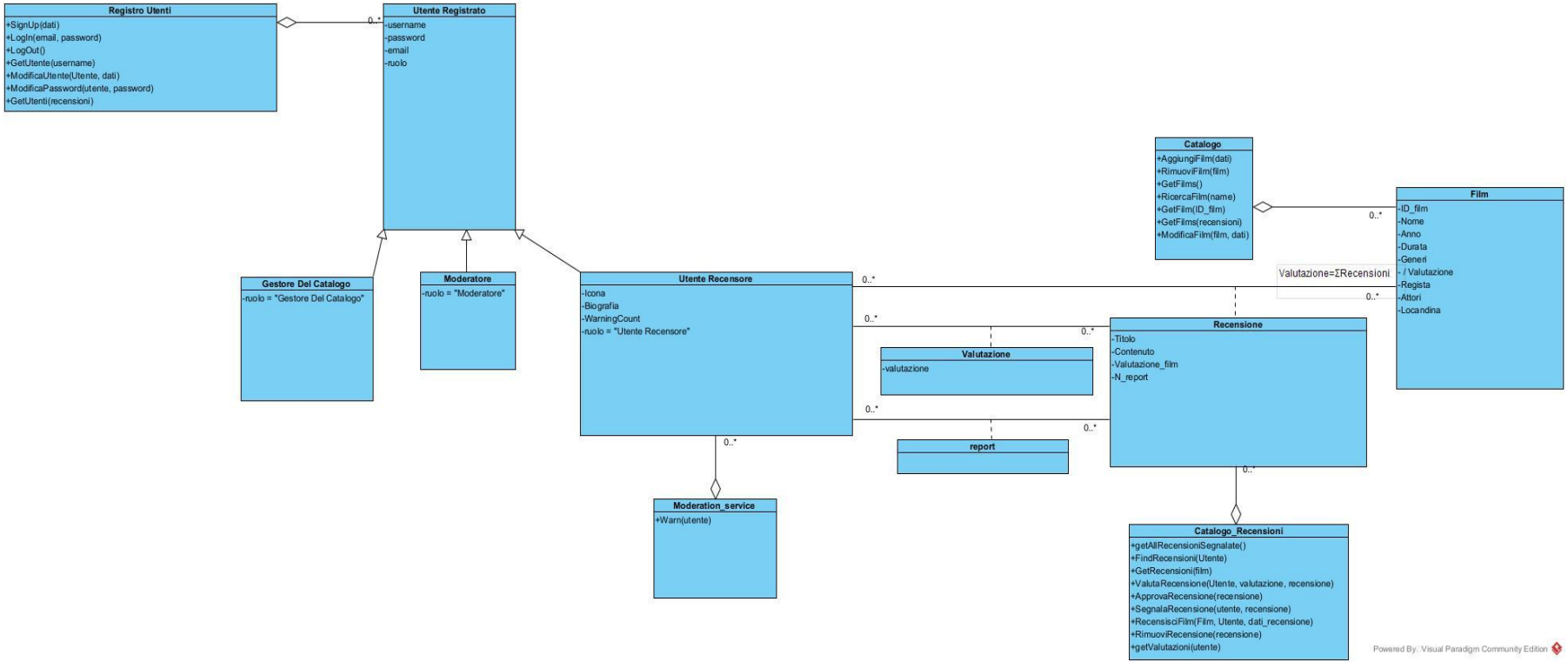
ATTIVITÀ SVOLTE PER LA REQUIREMENT ANALYSIS

RATED
•COMMUNITY REVIEWS•

- Identificazione degli Entity objects
- Identificazione dei Boundary objects
- Identificazione dei Control objects
- Produzione di una parte del modello dinamico del sistema
  - Mapping dei casi d'uso con sequence diagram.
- Produzione del modello ad oggetti di analisi:
  - Identificazione delle associazioni.
  - Identificazione degli attributi degli oggetti.
  - Modellazione delle relazioni di ereditarietà

MODELLO AD OGGETTI

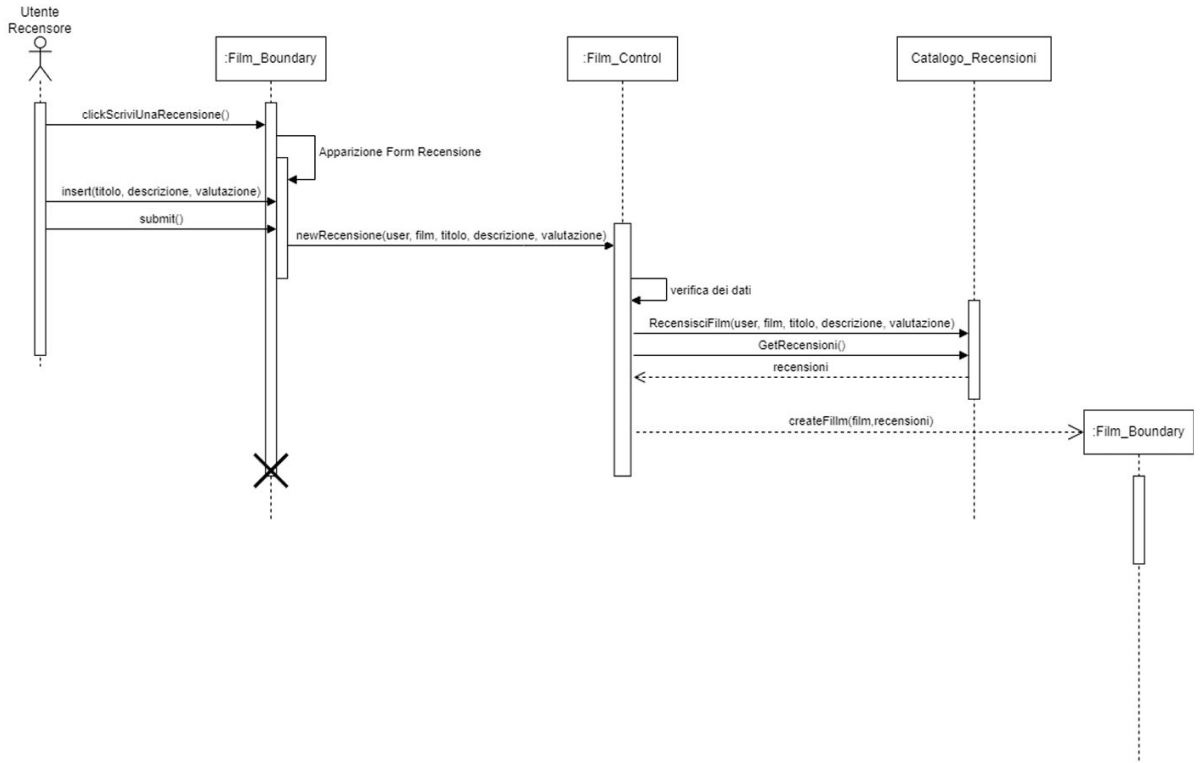
RATED
•COMMUNITY REVIEWS•



ESEMPIO DI SEQUENCE DIAGRAM COMPI

RATED
•COMMUNITY REVIEWS•

SD13- Pubblicazione di una recensione



RATED
•COMMUNITY REVIEWS•

04 SYSTEM DESIGN

ATTIVITÀ SVOLTE IN FASE DI SYSTEM DESIGN

- Identificazione dei design goals.
- Decomposizione del sistema.
  - Stile di architettura adottato
  - Individuazione dei servizi
  - Definizione della tabella degli accessi
- Mapping Hardware/Software

DESIGN GOALS: USABILITÀ

RATED
•COMMUNITY REVIEWS•

Validazione degli input: saranno implementati meccanismi per prevenire errori durante l'inserimento di dati. Messaggi di errore chiari guideranno l'utente nella correzione dei valori errati.

Design responsive: l'interfaccia sarà ottimizzata per adattarsi a diversi dispositivi (PC, tablet, smartphone), rendendo l'esperienza uniforme e accessibile.

Navigazione intuitiva: ogni pagina presenterà una barra di navigazione per facilitare l'accesso rapido alle diverse sezioni.

DESIGN GOALS: AFFIDABILITÀ

RATED
•COMMUNITY REVIEWS•

Controllo avanzato degli input: oltre alla validazione primaria, saranno effettuati ulteriori controlli per gestire scenari non previsti e prevenire errori critici.

Sicurezza dei dati: saranno adottati protocolli di crittografia per proteggere le informazioni sensibili, come le credenziali degli utenti. Le password dovranno rispettare requisiti di complessità (es. lunghezza minima e inclusione di caratteri speciali).

DESIGN GOALS: PRESTAZIONI & SUPPORTABILITÀ

RATED
•COMMUNITY REVIEWS•

Prestazioni: Il sistema deve garantire tempi di caricamento inferiori a 2 secondi per ogni pagina o funzione principale.

Supportabilità: Il codice deve essere strutturato in modo da garantire futuri aggiornamenti senza compromettere l'integrità del sistema. Per raggiungere questo obiettivo, il software sarà sviluppato suddividendo le funzionalità in sottosistemi altamente coesi e debolmente accoppiati, organizzati in strati omogenei di astrazione.

La coesione interna dei sottosistemi sarà ottimizzata raggruppando esclusivamente le classi che condividono scopi e operazioni comuni. Questo approccio di progettazione rafforza ed estende il requisito non funzionale di supportabilità, garantendo una chiara stratificazione dell'applicazione in tre livelli di responsabilità.

SUDDIVISIONE IN SOTTOSISTEMI

RATED
•COMMUNITY REVIEWS•

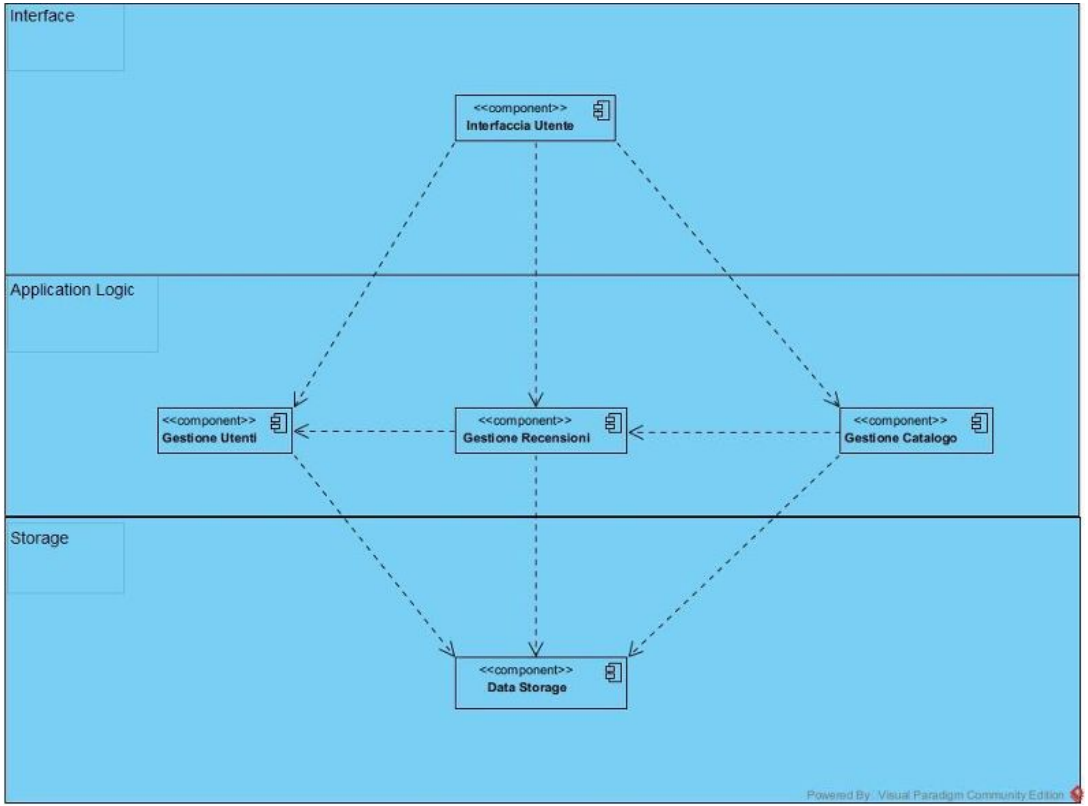
Il sistema è stato suddiviso in più sottosistemi, ciascuno caratterizzato da un'elevata concentrazione di processi con obiettivi e ambiti comuni. Questa suddivisione garantisce una chiara definizione delle responsabilità, favorendo un'architettura più organizzata e manutenibile.

Il sistema adotta un'**architettura a strati semi-chiusa**, in cui ogni livello può comunicare **sia con lo strato immediatamente inferiore che con altri componenti dello stesso livello**, se necessario. Questo approccio garantisce un equilibrio tra rigore architetturale e flessibilità. Questa configurazione favorisce:

- **Alta manutenibilità**: le modifiche a uno strato non influenzano direttamente gli altri, rendendo più semplice la gestione del codice.
- **Flessibilità**: la possibilità di comunicazione tra componenti dello stesso livello consente di ottimizzare i flussi di dati senza dover sempre passare attraverso lo strato superiore o inferiore.
- **Maggiore efficienza**: in alcuni scenari, i componenti possono interagire direttamente tra loro senza dover passare attraverso un'interfaccia comune, riducendo la latenza e migliorando le prestazioni.

ARCHITETTURA: COMPONENT DIAGRAM

RATED
•COMMUNITY REVIEWS•



PERCHÈ UTILIZZARE UN'ARCHITETTURA SEMI-CHIUSA?

Per questa ragione abbiamo deciso di adottare un'architettura semi-chiusa con l'obiettivo principale di mantenere un'elevata coesione all'interno dei nostri moduli.

1. **Gestione di Operazioni Multi-Contesto:** Alcune funzionalità del nostro sistema, come ad esempio la rimozione di una recensione che viola le linee guida e il warn dell'utente, operano su più ambiti contemporaneamente. Questa funzione, per sua natura, potrebbe appartenere sia al modulo di gestione utenti sia a quello di gestione recensioni. Un'architettura semi-chiusa consente di gestire tali operazioni in modo efficace senza compromettere la coesione di ciascun modulo.
2. **Separazione delle Responsabilità:** Adottando un'architettura semi-chiusa, riusciamo a separare chiaramente la logica delle funzioni che interagiscono con diversi domini. Questo approccio permette di isolare le responsabilità, riducendo le dipendenze incrociate tra i moduli e facilitando l'individuazione e la risoluzione di eventuali problemi.
3. **Flessibilità e Scalabilità:** Una struttura semi-chiusa offre una maggiore flessibilità nell'aggiungere nuove funzionalità che potrebbero coinvolgere più contesti. Questo rende il sistema più scalabile, permettendo di integrare facilmente nuove operazioni senza dover ristrutturare l'intera architettura.
4. **Manutenibilità Migliorata:** Con una chiara separazione delle logiche che operano su diversi ambiti, la manutenzione del codice diventa più semplice. Gli sviluppatori possono concentrarsi su specifiche aree del sistema senza il rischio di introdurre bug in altre parti non correlate.

COMPONENT DIAGRAM: INTERFACE

RATED
•COMMUNITY REVIEWS•

Interface
L'Interfaccia Utente è il sottosistema responsabile dell'interazione con l'utente finale. Fornisce le classi e le operazioni necessarie per acquisire input, visualizzare output e interagire con i servizi della piattaforma.

Si occupa inoltre di: Coordinare la logica applicativa, delegando le operazioni principali ai sottosistemi del livello sottostante. Validare i dati in ingresso, assicurando la conformità alle specifiche del sistema e restituendo eventuali messaggi di errore in caso di input errati.

COMPONENT DIAGRAM: APPLICATION LOGIC

RATED
•COMMUNITY REVIEWS•

Application Logic
La Logica Applicativa è suddivisa in tre sottosistemi principali:

Gestione Utenti: si occupa dell'autenticazione, della creazione e dell'eliminazione degli account. Garantisce la protezione delle credenziali attraverso l'adozione di protocolli sicuri per la gestione dei dati sensibili.

Gestione Catalogo: consente ai Gestori di inserire, aggiornare e rimuovere film dal catalogo, mantenendo un'offerta costantemente aggiornata e coerente con le preferenze della community.

Gestione Recensioni: supporta l'aggiunta, la valutazione e la moderazione delle recensioni. Questo include meccanismi per segnalare contenuti inappropriati e l'assegnazione di punteggi che influenzano il sistema reputazionale.

COMPONENT DIAGRAM: STORAGE

RATED
•COMMUNITY REVIEWS•

Storage
Il Data Storage rappresenta il sottosistema responsabile della memorizzazione e gestione degli oggetti persistenti, come:

Dati utente: informazioni personali, credenziali e preferenze.

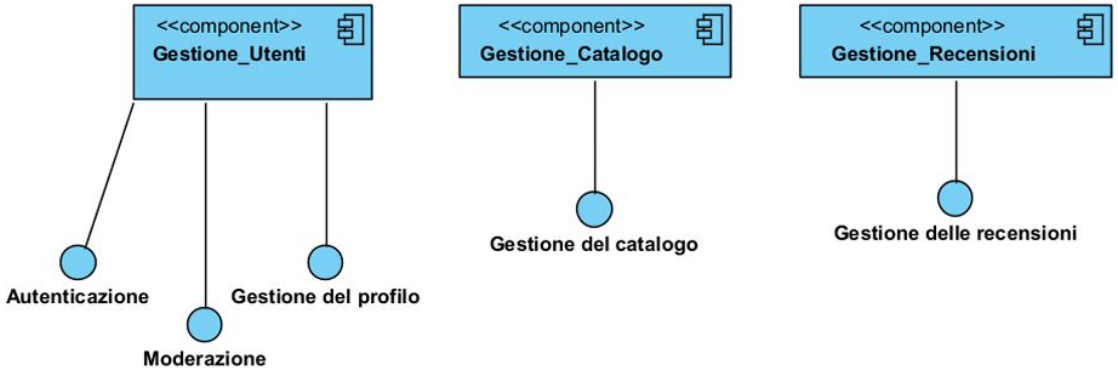
Dati di catalogo: elenco dei film disponibili con relative informazioni.

Dati delle recensioni: contenuti testuali, valutazioni e metadati associati. La persistenza dei dati è garantita attraverso l'utilizzo di un database relazionale ottimizzato per supportare operazioni frequenti e simultanee.

SERVIZI DEI SOTTOSISTEMI

RATED
•COMMUNITY REVIEWS•

Considerando i sottosistemi del livello Application Logic:
Gestione Utenti, Gestione Film e Gestione Recensioni, si elencano i servizi che offrono.



SERVIZI DEI SOTTOSISTEMI

RATED
•COMMUNITY REVIEWS•

Gestione Utenti

• Servizio di **Autenticazione**, per la registrazione dell'utente e per la verifica delle credenziali di accesso a tempo di Login. Operazioni: login(), logout(), register().

• Servizio di **Moderazione**, per gestire gli utenti che violano le linee guida. Operazioni: warn().

• Servizio di **Gestione del Profilo**, per permettere all'utente di poter aggiornare il proprio profilo. Operazioni: ProfileUpdate(), PasswordUpdate(), findByUsername(), getUtenti.

SERVIZI DEI SOTTOSISTEMI

RATED
•COMMUNITY REVIEWS•

Gestione Catalogo

• Servizio di **Gestione del Catalogo** per l'aggiunta, modifica e rimozione dei film. Operazioni: getFilms(), aggiungiFilm(), rimuoviFilm(), ricercaFilm(), getFilm(), getFilms(), addFilm(), modifyFilm(), removeFilm()

Gestione Recensioni

• Servizio di **Gestione delle recensioni**, per permettere di gestione di recensioni, valutazioni e report.

MAPPING HARDWARE/SOFTWARE

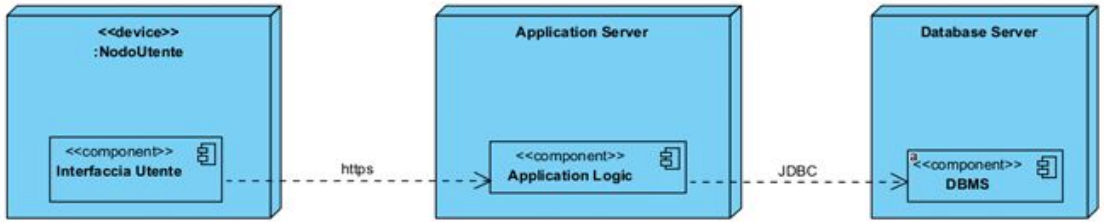
RATED
•COMMUNITY REVIEWS•

Il sistema Rated è, per sua natura, un sistema distribuito, poiché gli utenti interagiscono con esso da macchine diverse, in remoto.
Si distinguono tre tipi principali di componenti:

• Nodo Utente, che fornisce l'interfaccia utente. Questo nodo esegue il sottosistema Interface tramite un web browser.

• Application Server, che gestisce la logica applicativa e i controlli. Esegue il sottosistema Application Logic.

• Database Server, che gestisce la persistenza dei dati e i servizi offerti. Esegue il sottosistema Storage.



RATED
•COMMUNITY REVIEWS•

05 OBJECT DESIGN

ATTIVITÀ SVOLTE IN FASE DI OBJECT DESIGN

RATED
•COMMUNITY REVIEWS•

- Valutazione dei Trade-Offs di base per l'implementazione
- Design Patterns
- Packaging del sistema
- Interfacce delle classi del sistema
  - Metodi
  - PreCondizioni
  - PostCondizioni

TRADE - OFFS

RATED
•COMMUNITY REVIEWS•

Sicurezza vs. Tempi di Sviluppo:
Tenendo conto delle scadenze e dei tempi ristretti di sviluppo, non tutti i requisiti funzionali con grado di priorità media verranno considerati nell'implementazione del sistema. Saranno, inoltre, implementati solo i sistemi di sicurezza essenziali per garantire un livello adeguato di protezione. Questi includono l'autenticazione tramite email e password crittografata, la sanificazione dei campi di input dei form e una gestione degli accessi alle pagine basata sui ruoli definiti nel RAD (Guest, Recensore, Gestore del Catalogo, Moderatore). Le pagine di errore, tuttavia, saranno progettate in modo minimale e orientate esclusivamente alla funzionalità essenziale. Nella prima versione del sistema, i controlli sui form saranno limitati a verifiche di base per tutti gli utenti, inclusi quelli compilati dai Gestori del Catalogo, che si presuppone, in qualità di operatori della piattaforma, abbiano familiarità con le modalità di compilazione relative alle loro funzioni. Tuttavia, per i form critici, come quelli di LogIn e Register, saranno implementati controlli più rigorosi per garantire un adeguato livello di sicurezza. Questo approccio rappresenta un compromesso mirato a proteggere i dati sensibili e assicurare il corretto funzionamento della piattaforma, mantenendo al contempo la rapidità e la semplicità di implementazione necessarie per rispettare le scadenze.

TRADE - OFFS

RATED
•COMMUNITY REVIEWS•

Prestazioni vs. Supportabilità:
Nel contesto del progetto, è importante considerare il possibile trade-off tra prestazioni e supportabilità. Sebbene il requisito di prestazioni, definito nel RAD, richieda tempi di caricamento inferiori a 2 secondi per ogni pagina o funzione principale, si ritiene prioritario privilegiare la sopportabilità del codice. Questa scelta si fonda sull'ipotesi che l'applicativo web non presenti una complessità elevata. Di conseguenza, è ragionevole supporre che una struttura del codice orientata alla manutenibilità e agli aggiornamenti futuri non comprometta in modo significativo le prestazioni del sistema. In altre parole, adottare pratiche di sviluppo che favoriscano la supportabilità, come una progettazione modulare, codice leggibile e testabile, dovrebbe consentire di mantenere un tempo di risposta inferiore ai 2 secondi, soddisfacendo così entrambi i requisiti. Tale approccio garantirebbe un equilibrio tra efficienza immediata e sostenibilità a lungo termine del sistema, riducendo il rischio di complicazioni durante l'evoluzione dell'applicativo.

DESIGN PATTERN

RATED
•COMMUNITY REVIEWS•

Connection Pool Management Pattern

L'applicazione Rated richiede un accesso ottimizzato e centralizzato alle connessioni al database, data la natura concorrente delle operazioni effettuate dagli utenti. L'implementazione di un Connection Pool Management si rivela essenziale per migliorare l'efficienza nella gestione delle connessioni al database MySQL. L'utilizzo di questa soluzione consente di mantenere un pool di connessioni già aperte e riutilizzabili, evitando l'overhead causato dalla creazione e dalla chiusura continua di nuove connessioni.

Ciò garantisce:

- La gestione di un numero limitato di connessioni attive, prevenendo così un utilizzo inefficiente delle risorse del database.
- Una maggiore scalabilità del sistema, grazie alla possibilità di servire più richieste concorrenti.
- Un ciclo di vita chiaro e centralizzato per tutte le connessioni.

Nel progetto Rated, il **DriverConnectionPool** è responsabile della gestione di queste connessioni. Attraverso un'allocazione intelligente e il rilascio delle connessioni utilizzate, il pool di connessioni garantisce la continuità operativa dell'applicazione senza sovraccaricare il DBMS. L'accesso alle connessioni nel pool avviene tramite metodi che permettono di acquisire una connessione disponibile e di restituirla una volta terminato l'utilizzo. In questo modo si ottimizza l'utilizzo delle risorse e si previene il verificarsi di problemi di saturazione delle connessioni.

DESIGN PATTERN

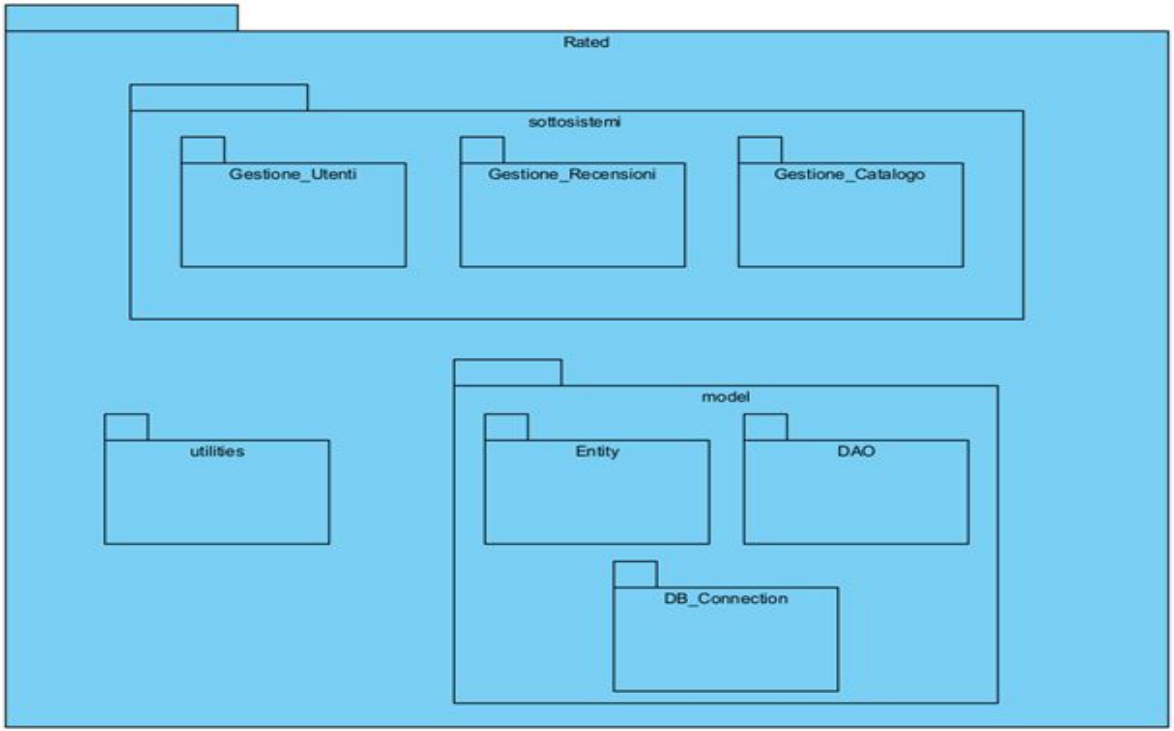
RATED
•COMMUNITY REVIEWS•

Strategy Pattern
Nel progetto, per la gestione della validazione dei campi di input, è stato adottato il Strategy Pattern. Questo design pattern consente di definire una famiglia di algoritmi di validazione (ad esempio, validazione di email, numeri, date, ecc.), incapsularli in metodi specifici all'interno di una classe e renderli intercambiabili. La classe di validazione creata funge da contenitore per tutte le funzioni di validazione necessarie, ognuna delle quali rappresenta una strategia separata per verificare uno specifico tipo di input.

Questo approccio garantisce:
• Modularità: Ogni algoritmo di validazione è implementato come un metodo distinto, facilitando la leggibilità e la manutenzione del codice.
• Flessibilità: È possibile aggiungere facilmente nuove funzioni di validazione o aggiornare quelle esistenti senza modificare il codice già scritto.
• Riutilizzabilità: Le funzioni di validazione possono essere richiamate in modo indipendente o combinate, a seconda delle necessità del sistema. Chiarezza: Separando la logica di validazione dalla logica principale dell'applicazione, si ottiene un design più chiaro e mantenibile.

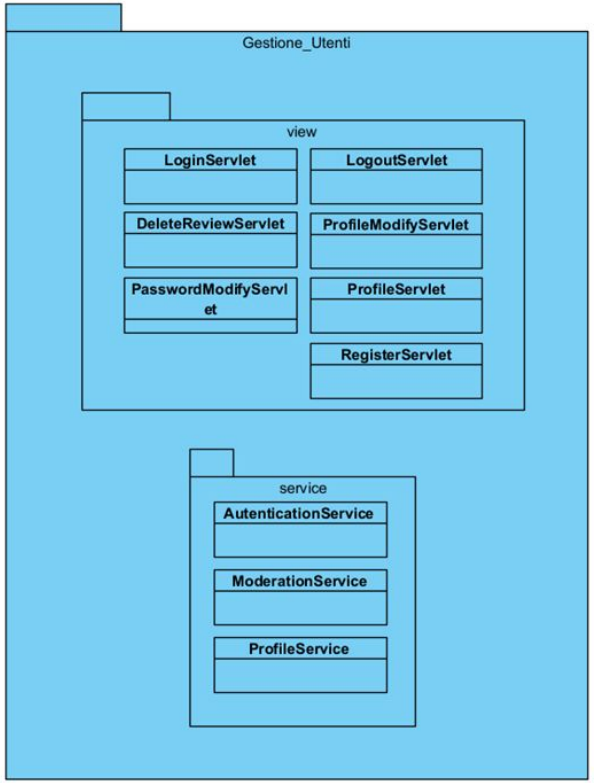
PACCHETTI

RATED
•COMMUNITY REVIEWS•



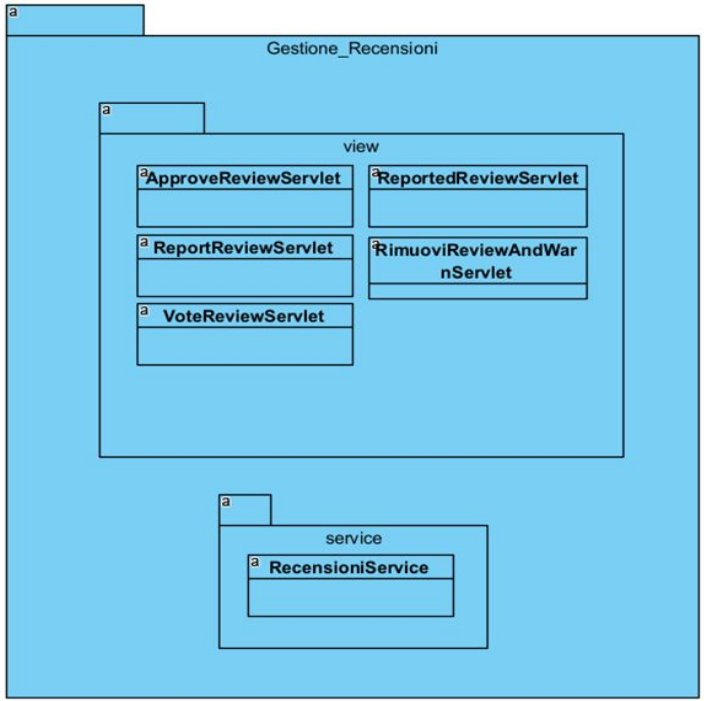
PACCHETTO GESTIONE_UTENTI

RATED
•COMMUNITY REVIEWS•



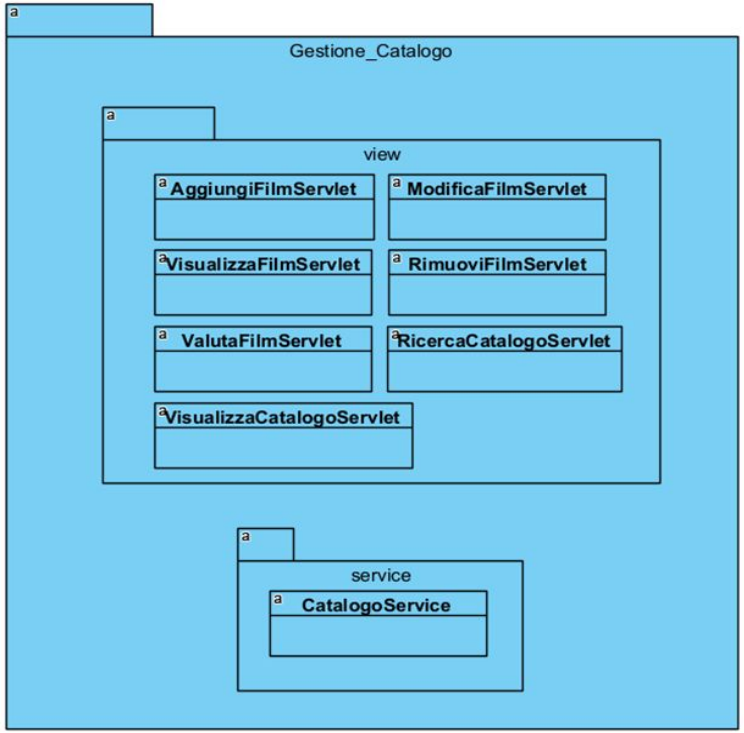
PARTIZIONAMENTO GESTIONE_RECENSIONI

RATED
•COMMUNITY REVIEWS•



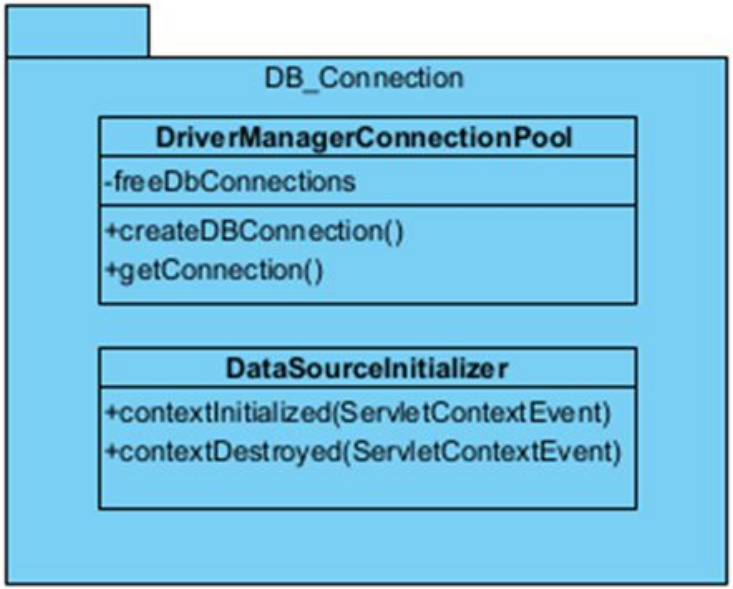
PACCHETTO GESTIONE_CATALOGO

RATED
•COMMUNITY REVIEWS•



PACCHETTO DB_CONNECTION

RATED
•COMMUNITY REVIEWS•



ESEMPIO DI DEFINIZIONE DELLE SIGNATURE ESATTE PER I METODI

RATED
•COMMUNITY REVIEWS•

Interfaccia	**AutenticationService**
Descrizione	AutenticationService fornisce il servizio relativo all'autenticazione e all'aggiornamento degli account
Metodi	+login(email: String, password: String) : UtenteBean +logout(session: HttpSession) : void +register(username: String, email: String, password: String, biografia: String, icon: byte[]) : UtenteBean
Invariante classe	//

ESEMPIO DI DEFINIZIONE DI PRE/POST CONDIZIONI DEI METODI

RATED
•COMMUNITY REVIEWS•

Nome Metodo	+login(email: String, password: String) : UtenteBean
Descrizione	Questo metodo permette a un utente di autenticarsi.
Pre-condizione	context: AutenticationService::login(email: String, password: String) pre: email <> null AND email.trim() <> "" AND password <> null AND password.trim() <> ""
Post-condizione	context: AutenticationService::login(email: String, password: String) post: (result = null AND UtenteDAO.findByEmail(email) = null) OR (result <> null AND PasswordUtility.hashPassword(password).equals(result.getPassword()))

RATED
•COMMUNITY REVIEWS•

06 TESTING

ATTIVITÀ SVOLTE IN FASE DI TESTING

RATED
•COMMUNITY REVIEWS•

- Pianificazione dei Test
- Specifiche dei casi di Test
- Esecuzione dei Test
- Incident Report
- Valutazioni finali

PIANIFICAZIONE DEI TEST

RATED
•COMMUNITY REVIEWS•

L’obiettivo principale del Testing è stato garantire che **Rated** soddisfasse i requisiti funzionali e non funzionali, identificati precedentemente.

Si sono potute testare le varie componenti del sistema tramite Test di **unità**, di **integrazione** e di **sistema**. L’attività di testing ha seguito approcci **bottom-up** (partendo dai moduli più piccoli e integrandoli progressivamente fino al sistema completo) e **black-box** (basato sull'analisi degli input e output senza conoscere il funzionamento interno del codice).

Per il testing si è deciso di usare **JUnit/Mockito.**

PIANIFICAZIONE DEI TEST

RATED
•COMMUNITY REVIEWS•

Il programma dei test per Rated è stato pianificato in funzione delle deadline di sviluppo e delle priorità di rilascio:

1. **Test di Unità** (Tempo di sviluppo)
   - Viene svolto in parallelo allo sviluppo. Ogni nuovo modulo o classe viene testato immediatamente.
   - Garantisce che le unità rispettino le specifiche fornite in fase di design.

2. **Test di Integrazione** (Tempo di sviluppo)
   - Una volta consolidati i moduli di base, si integrano progressivamente le componenti (DAO, interfacce, etc.).

3. **Test di Sistema** (2-3 giorni)
   - Al termine dell'integrazione, si eseguono test simulando l'utilizzo tipico del sistema da parte di un utente.(registrazione utente, login, pubblicazione recensioni, moderazione, gestione film, etc.).

PIANIFICAZIONE DEI TEST

RATED
•COMMUNITY REVIEWS•

PASS/FAIL CRITERIA

Sono stati definiti i seguenti criteri:

- **Test superato**: l'output del sistema corrisponde all'oracolo (risultato atteso).
- **Test fallito**: il risultato effettivo differisce da quello atteso.

SOSPENSIONE

Si è deciso di sospendere il testing solo in caso di:

- **Errori critici/bloccanti**.
- **Build instabili** dovute a modifiche errate del codice.

In caso di sospensione o fallimento:

1. Si identifica e corregge il problema (bug fix).
2. Si eseguono test di regressione per verificare le correzioni.

FUNZIONALITÀ PRINCIPALI TESTATE

RATED
•COMMUNITY REVIEWS•

Gestione Film

- Visualizzazione del catalogo e relativi dettagli
- Ricerca e ordinamento dei film
- Aggiunta, modifica e rimozione di un film (gestore)

Gestione Utente

- Registrazione (SignUp) e autenticazione (LogIn, LogOut)
- Visualizzazione e modifica del profilo personale (compresa la modifica password)
- Visualizzazione del profilo di altri utenti

Gestione Recensioni

- Pubblicazione di nuove recensioni
- Visualizzazione dettagliata di una recensione
- Valutazione della recensione (like/dislike)
- Segnalazione e rimozione di una recensione (moderazione)
- Accesso all’area di moderazione (approvazione/eliminazione)

ESEMPIO DI SPECIFICA DI UN CASO DI TEST

RATED
•COMMUNITY REVIEWS•

1.3. Ricerca di un film

Obiettivo: Verificare la corretta funzionalità della barra di ricerca per trovare i film in base a titoli (completi o parziali).

Definizione parametri (Categorie)

- **Stringa di ricerca (SR)**:
    1. Stringa esatta / parziale che matcha almeno un film
    2. Stringa che non matcha nessun film [error → nessun risultato]
    3. Stringa vuota [error → nessun risultato]

Definizione Test Frame

Test Case ID	Test Frame	Esito Atteso
TC02.1	SR1	Vengono mostrati i film corrispondenti alla stringa (≥1).
TC02.2	SR2	Messaggio "Nessun risultato" o pagina vuota di film.
TC02.3	SR3	Nessun risultato o reindirizzo a pagina catalogo.

Test Case

Test Case ID: **TC02.1**
Input:

- Stringa di ricerca = “Inception”
  Oracolo: Mostra il film “Inception” nella lista dei risultati.

Test Case ID: **TC02.2**
Input:

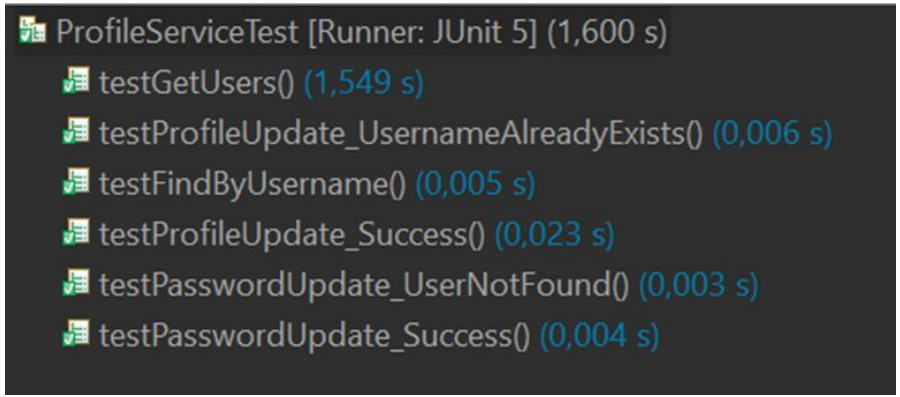
- Stringa di ricerca = “FilmCheNonEsiste123”
  Oracolo: Mostra messaggio “Nessun risultato trovato”.

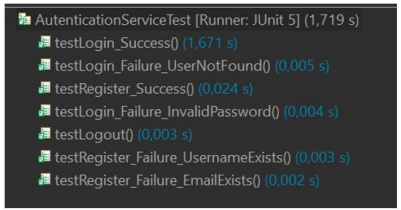
Test Case ID: **TC02.3**
Input:

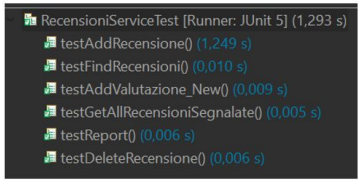
- Stringa di ricerca = “” (vuota)
  Oracolo: Viene caricato il catalogo completo.

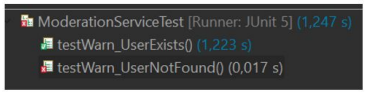
ESECUZIONE DEI TEST:TEST DI UNITA'

RATED
•COMMUNITY REVIEWS•









ESECUZIONE DEI TEST:TEST DI INTEGRAZIONE

RATED
•COMMUNITY REVIEWS•

ModerationServiceIntegrationTest [Runner: JUnit 5] (0,001 s)
- testWarn_UserNotFound() (0,001 s)

CatalogoServiceIntegrationTest [Runner: JUnit 5] (1,666 s)
- testGetFilms_ReturnsAllFilms() (1,634 s)
- testRimuoviFilm_ShouldDeleteFromDB() (0,013 s)
- testAggiungiFilm_ValidData_ShouldSaveToDatabase() (0,007 s)
- testRicercaFilm_ExistingTitle_ShouldReturnResult() (0,005 s)

ProfileServiceIntegrationTest [Runner: JUnit 5] (2,081 s)
- testGetUsers() (0,043 s)
- testProfileUpdate_UsernameAlreadyExists() (0,504 s)
- testProfileUpdate_Success() (0,511 s)
- testPasswordUpdate_UserNotFound() (0,507 s)
- testPasswordUpdate_Success() (0,509 s)

AutenticationServiceIntegrationTest [Runner: JUnit 5] (1,565 s)
- testRegister_ExistingEmail_ShouldFail() (0,041 s)
- testRegister_NewUser_ShouldSucceed() (0,503 s)
- testLogin_WrongPassword_ShouldReturnNull() (0,507 s)
- testLogin_CorrectCredentials_ShouldReturnUser() (0,509 s)

RecensioniServiceIntegrationTest [Runner: JUnit 5] (2,007 s)
- testAddRecensione_Duplicate_ShouldNotCreate() (0,134 s)
- testAddValutazione_NewLike_ShouldIncrementNLike() (0,618 s)
- testDeleteRecensione_ShouldRemoveAndUpdateFilmRating() (0,608 s)
- testAddRecensione_Valid_ShouldCreateAndUpdateFilmRating() (0,641 s)

ESECUZIONE DEI TEST:TEST DI SISTEMA

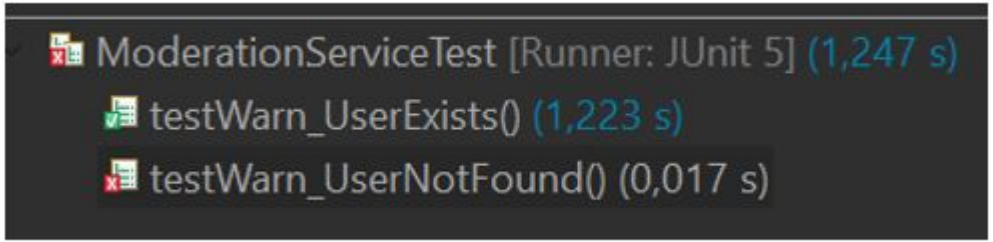
RATED
•COMMUNITY REVIEWS•

Per rispettare le scadenze del progetto sono state testate manualmente e in maniera facilmente riproducibile tutte le funzionalità di ogni tipo di utente della piattaforma, invece di usare strumenti di automazione avanzata come Selenium.

Tutti i test effettuati hanno avuto esito positivo, dimostrando la robustezza del sistema.

INCIDENT REPORT

RATED
•COMMUNITY REVIEWS•



Il fallimento della funzione Warn() in caso di utente non presente nel database non rappresenta un problema per l'applicazione. Attualmente, infatti, non sono previste funzionalità di disiscrizione, rendendo questa situazione impossibile nel contesto operativo attuale. Di conseguenza, al fine di ottimizzare i tempi di sviluppo e rispettare le scadenze, si è deciso di non implementare una gestione specifica per tale errore.

VALUTAZIONI FINALI

RATED
•COMMUNITY REVIEWS•

Dall'analisi dei risultati emerge una copertura estremamente soddisfacente dei casi di test previsti per il sistema. L'unico errore riscontrato, rispetto all'intero insieme dei test effettuati, rappresenta un esito altamente positivo sia in termini di progettazione sia di implementazione. Questo errore, come ben spiegato, non rappresenta un problema per la funzionalità del sito, è quindi garantita la piena conformità del sistema ai requisiti target.

Rated soddisfa pienamente gli obiettivi stabiliti, dimostrando una solidità funzionale coerente con le aspettative progettuali. Per rispettare le scadenze di consegna, il Test di Sistema è stato condotto manualmente, con esito positivo su tutte le funzionalità della piattaforma. Questa scelta ha assicurato un'esperienza utente corretta e fluida, rappresentando comunque un'opportunità di miglioramento per il futuro. In una revisione successiva, si potrebbe considerare l'adozione di strumenti di automazione, come Selenium, per ottimizzare ulteriormente il processo di testing. Nonostante l'attuale approccio manuale, il sistema dimostra un livello di affidabilità e funzionalità in linea con gli obiettivi progettuali, fornendo una base solida e performante per eventuali evoluzioni. Questo risultato riflette la qualità del lavoro svolto e pone le basi per ulteriori miglioramenti, mantenendo l'attenzione sull'efficienza e sull'innovazione continua.