Università degli Studi di Salerno
Corso di Ingegneria del Software

Rated
Test Case Specification
Versione 1.0

2/01/2026

Membri del gruppo

Nome	Matricola
Francesco Rao	NF22500211
Bruno Nesticò	NF22500213
Roberto Fiorenza	NF22500212

Revision History

Data	Versione	Descrizione	Autore
2/01/2026	1.0	Prima stesura completa	Francesco Rao, Bruno Nesticò, Roberto Fiorenza

Indice

1. Introduzione

A seguito dell'approvazione delle Change Request, l'attività di manutenzione si è concentrata sull'analisi del dominio delle modifiche e sulla revisione della progettazione del sistema **Rated**. L'obiettivo primario è stato individuare le componenti software impattate dall'introduzione delle nuove funzionalità, volte ad aggiornare la piattaforma includendo funzioni che aiutino l'utente con la pianificazione dei propri film.

Nel presente documento, dopo la presentazione dell'architettura attuale del Sistema, verranno analizzate nel dettaglio le componenti impattate da ciascuna change request, descrivendo le modifiche apportate al codice sorgente per la corretta implementazione delle nuove funzionalità.

2. Architettura del Sistema attuale

Al fine di comprendere il funzionamento del Sistema, prima di discutere dell'impatto delle modifiche, sono state riportate le principali caratteristiche strutturali e funzionali.

2.1 Panoramica Generale delle Funzionalità di Sistema

Il sistema attuale prevede diverse funzionalità specifiche a seconda della tipologia di utente (**Guest**, **Recensore**, **Gestore del Catalogo** o **Moderatore**) che ha effettuato l'accesso alla piattaforma.

Nel dettaglio:

- **L'utente Guest** può navigare nel catalogo dei film, consultare le schede tecniche, leggere le recensioni e visualizzare i profili pubblici degli altri membri. Per interagire attivamente, ha la possibilità di registrarsi al sistema creando un account personale.
- **L'utente Recensore**, previa autenticazione, può gestire il proprio profilo e interagire con la community pubblicando recensioni. Può inoltre votare i contributi altrui tramite un sistema di like/dislike e segnalare eventuali contenuti inappropriati. La sua dashboard personale tiene traccia delle recensioni pubblicate e del punteggio reputazionale accumulato.
- **Il Gestore del Catalogo** ha la responsabilità di mantenere aggiornata la libreria cinematografica. Può inserire nuovi titoli, modificare le informazioni dei film esistenti o rimuoverli dal sistema, garantendo che i dati (regista, data d'uscita, genere, ecc.) siano accurati e validati.
- **Il Moderatore** opera in un'area dedicata per garantire l'integrità della community. Il suo compito è esaminare le recensioni segnalate, approvandole o eliminandole. In caso di rimozione, il sistema gestisce automaticamente l'assegnazione degli

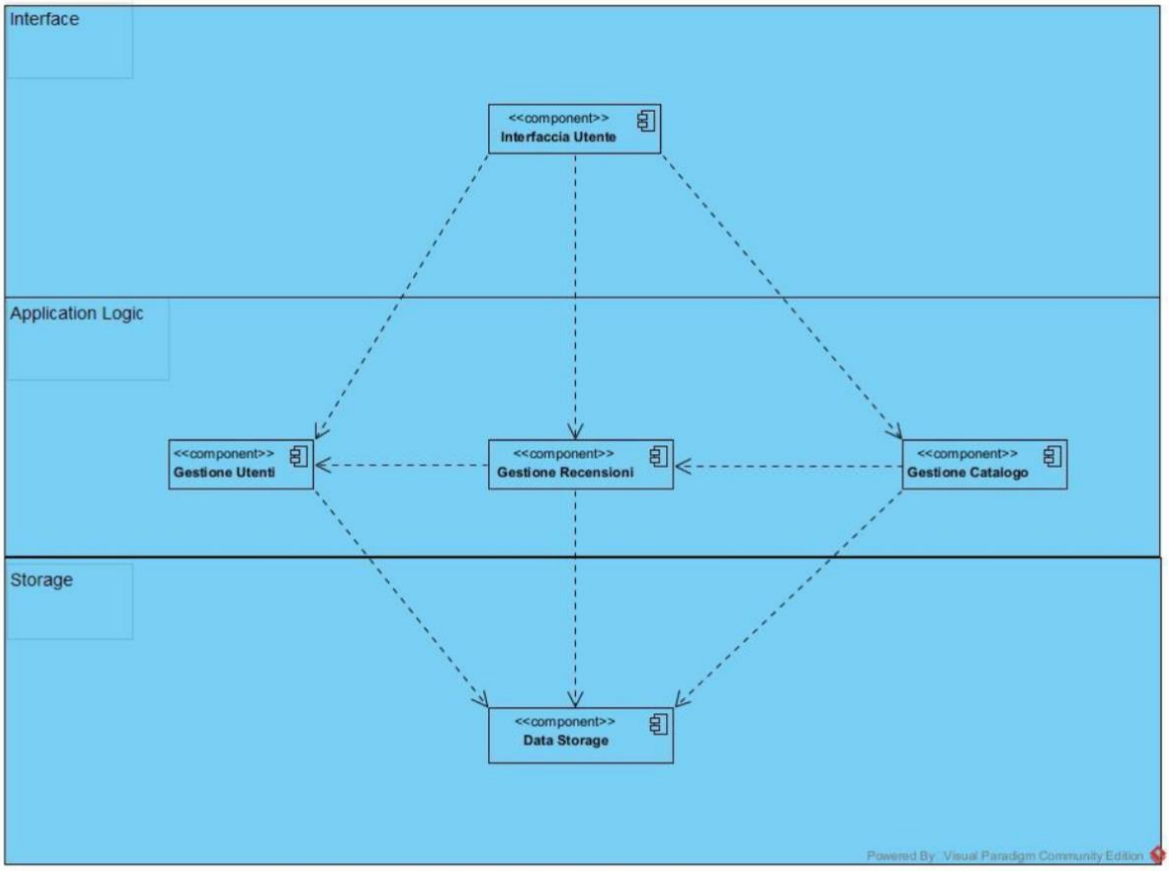
stati di *Warned* o *Limited* all'autore della recensione, limitandone le funzionalità in caso di recidiva.

Ogni tipologia di utente registrato usufruisce di una visualizzazione personalizzata e di strumenti specifici per il proprio ruolo, garantendo un ambiente sano, dinamico e partecipativo.

2.2 Architettura del Sistema

Il sistema è caratterizzato da un'architettura a tre livelli: livello Interface, livello Application Logic e livello Storage.

Component Diagram



2.2.1 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 validare i dati in ingresso, assicurando la conformità alle specifiche del sistema e restituendo eventuali messaggi di errore in caso di input errati.

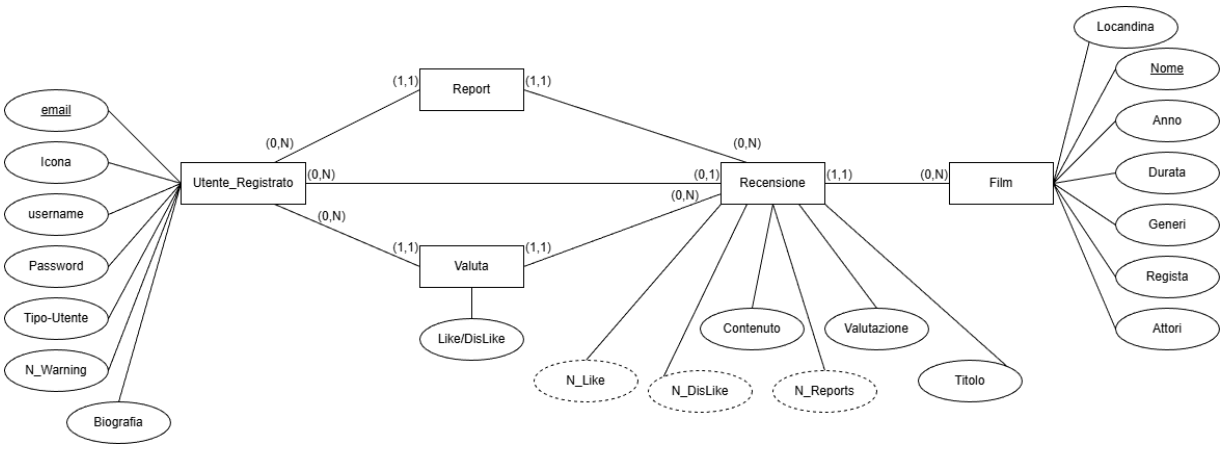
2.2.2 Application Logic

La Logica Applicativa è suddivisa in tre sottosistemi principali:

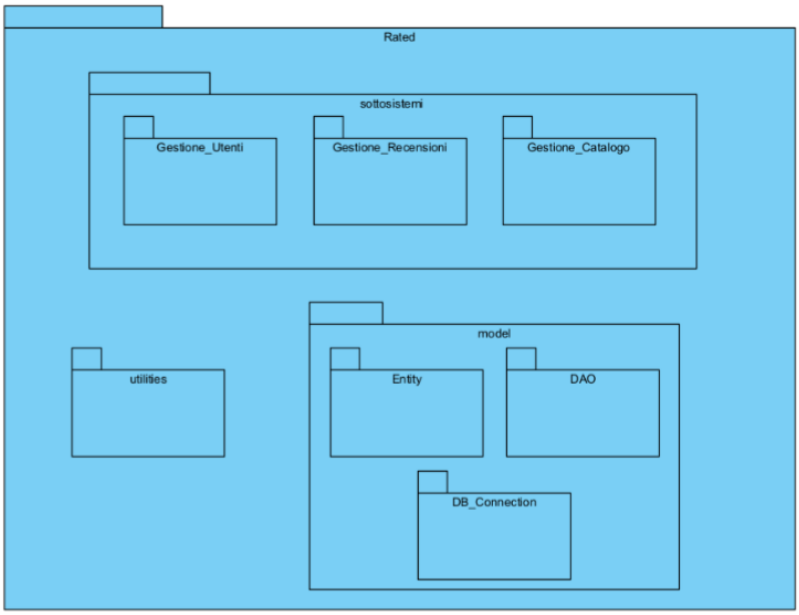
- **Gestione Utenti:** si occupa dell'autenticazione, della creazione e dell'eliminazione degli account. Garantisce la protezione delle credenziali attraverso l'adozione di protocolli sicuri per la gestione dei dati sensibili.
- **Gestione Catalogo:** consente ai Gestori di inserire, aggiornare e rimuovere 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 la segnalazione di contenuti inappropriati e l'assegnazione di punteggi che influenzano il sistema reputazionale.

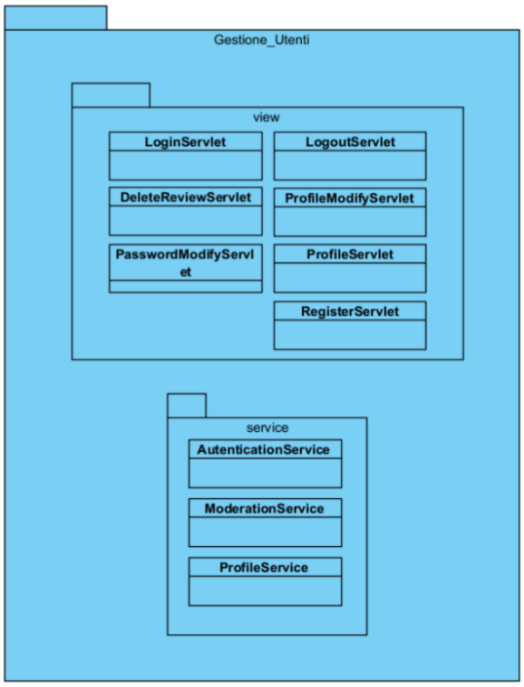
2.2.3 Storage

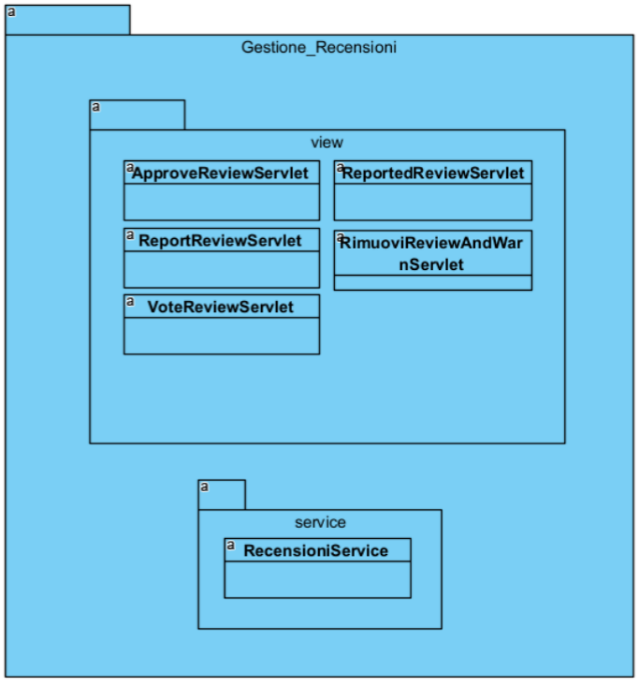
Il Data Storage rappresenta il sottosistema responsabile della memorizzazione e gestione, tramite database relazionale, degli oggetti persistenti.

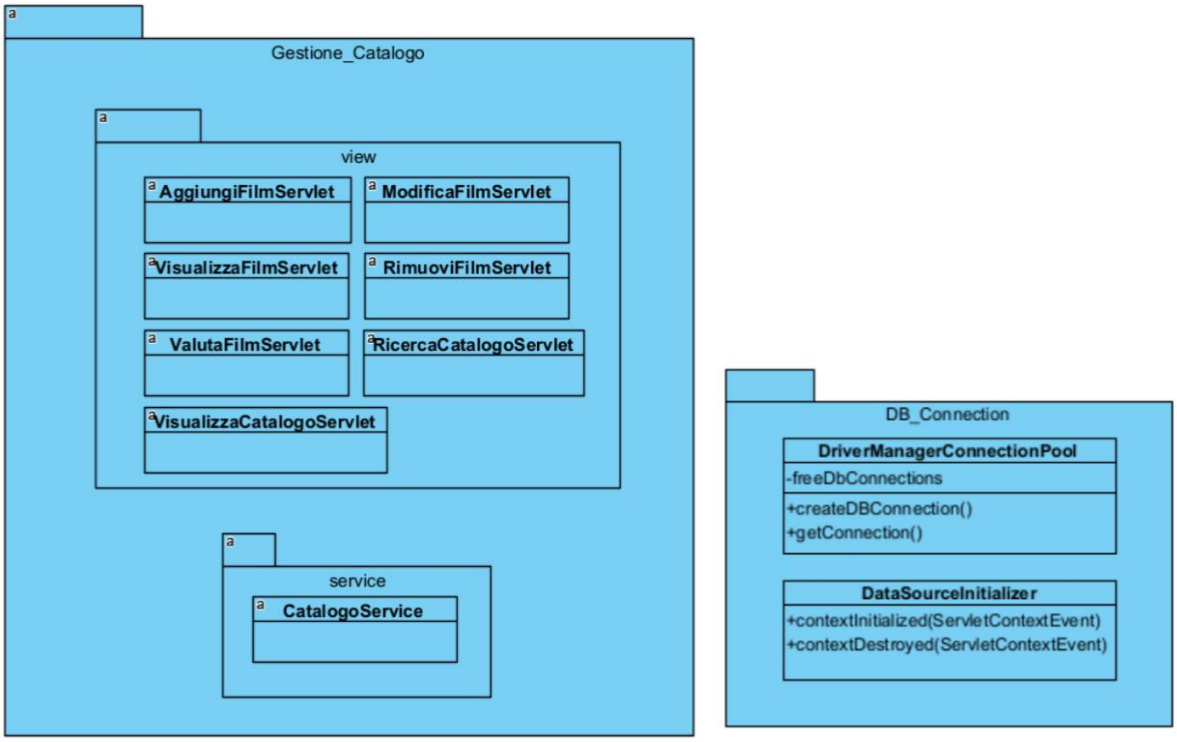


2.3 Packaging del sistema









3 Analisi delle Change Request

3.1 Change Request 1

La change request 1 consiste nell'aggiunta alla piattaforma di una sezione specifica a ciascun utente per la visualizzazione di due liste, quella relativa ai film visti (watched list) e film da vedere (watchlist). La sezione sarà accessibile dalla pagina utente e le liste potranno essere aggiornate selezionando un film dal catalogo e aggiungendolo o rimuovendolo, tramite appositi pulsanti nella pagina del film.

3.1.1 Analisi della richiesta di Modifica

La richiesta di modifica si configura come un intervento di manutenzione evolutiva, in quanto introduce nuove funzionalità di gestione dei contenuti personali che non erano precedentemente previste nel sistema. A differenza del workflow base di sola consultazione del catalogo, questa modifica richiede l'introduzione di 2 nuove classi "interesse" e "visto" tra gli utenti e i "Film", rendendo inoltre necessaria la persistenza di tali classi. Nello specifico si prevede che la classe "interesse" abbia un attributo boolean che se true indica che il film di riferimento sia in watchList.

Nello specifico, l'implementazione di queste nuove funzionalità comporterà i seguenti interventi:

- **Aggiornamento del Profilo Utente:** Realizzazione di una sezione dedicata, accessibile della pagina profilo, per la visualizzazione delle due liste (Watched e Watchlist).
- **Evoluzione dell'Interfaccia Film:** Integrazione nelle pagine di dettaglio dei film di trigger interattivi (pulsanti o icone) per permettere all'utente di inserire o rimuovere il film dalle proprie liste in tempo reale.
- **Gestione della Logica di Stato:** Implementazione dei controlli necessari a verificare se un film è già presente in una delle due liste, o tra le recensioni dell'utente, evitando duplicazioni o incongruenze.
- **Persistenza dei Dati:** Modifica della base dati per gestire la relazione "molti-a-molti" tra utenti e film, specifica per le tipologie di lista richieste.

Al fine di integrare correttamente queste funzionalità, si procederà con un'analisi degli artefatti impattati seguendo un approccio top-down: partendo dalla revisione dei casi d'uso e della documentazione, si arriverà a determinare i cambiamenti strutturali nel database e nelle classi di logica applicativa.

3.1.2 Impact Analysis

Come esplicitamente richiesto dalla change request è necessario andare ad aggiornare il database per ospitare le tabelle inerenti alla watchList e alla watchedList. Nello specifico, analizzando DB.sql e il rispettivo schema ER si è effettuato l'Impact Analysis prevedendo che le tabelle saranno aggiunte nel seguente modo:

```
CREATE TABLE Visto (
   email VARCHAR(255) NOT NULL,
   ID_Film INT NOT NULL,
   PRIMARY KEY (email, ID_Film),
   FOREIGN KEY (email) REFERENCES Utente_Registrato(email),
   FOREIGN KEY (ID_Film) REFERENCES Film(ID_Film)
);
```

```
CREATE TABLE Interesse (
   email VARCHAR(255) NOT NULL,
   ID_Film INT NOT NULL,
   interesse BOOLEAN NOT NULL,
   PRIMARY KEY (email, ID_Film),
   FOREIGN KEY (email) REFERENCES Utente_Registrato(email),
   FOREIGN KEY (ID_Film) REFERENCES Film(ID_Film),
);
```

pertanto si considera come parte delle componenti impattate **DB.sql**

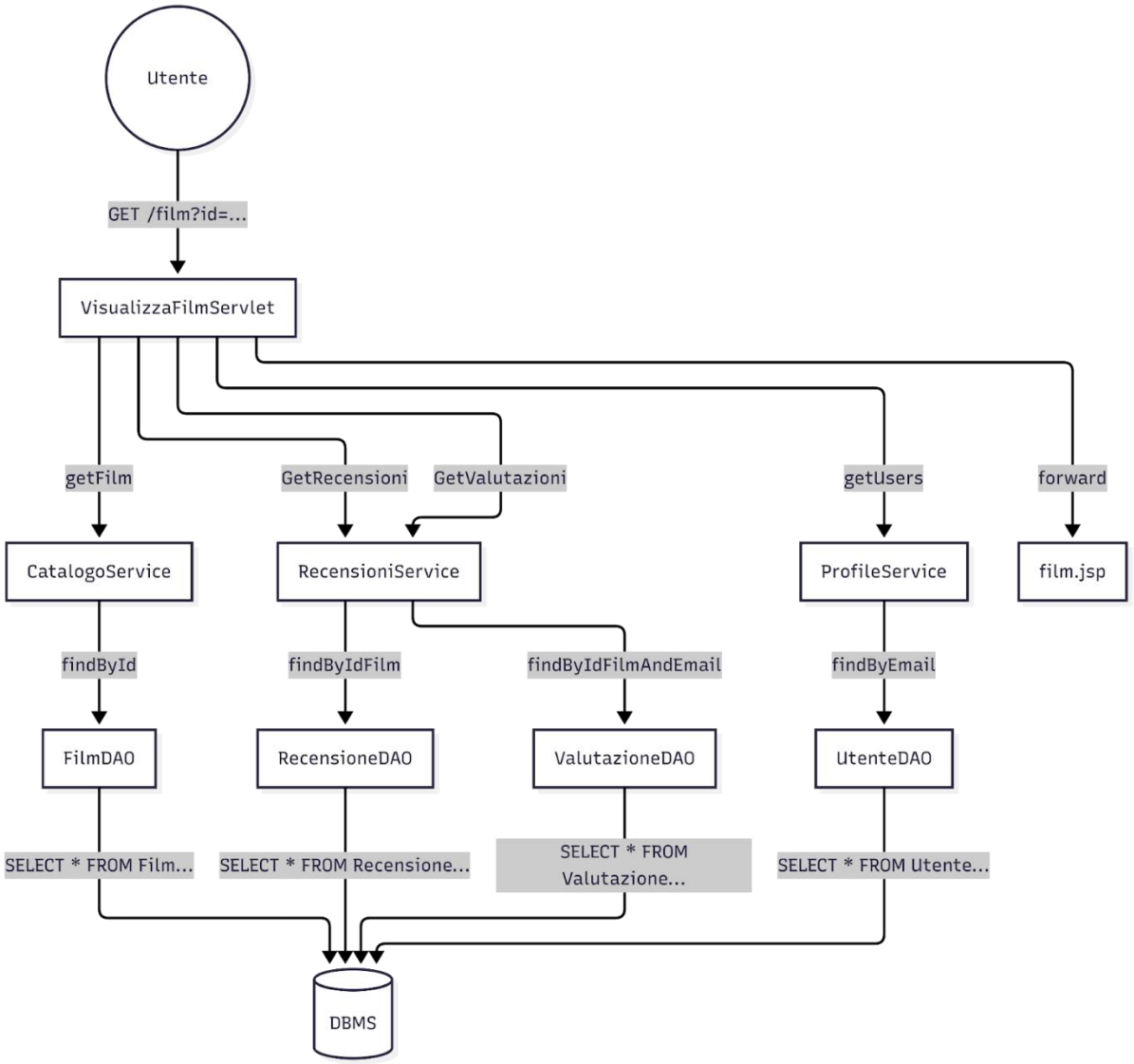
3.1.3 Determinazione dello Starting Impact Set (SIS)

Attraverso uno studio dell'architettura del sistema descritto nella documentazione, si è notato che ogni funzionalità offerta dalla piattaforma è erogata tramite una servlet di partenza. Nello specifico date le modifiche inerenti alla change request si sono identificate le seguenti servlet il quale studio dei rispettivi call graph dovrebbe esporre le componenti aventi bisogno di modifiche:

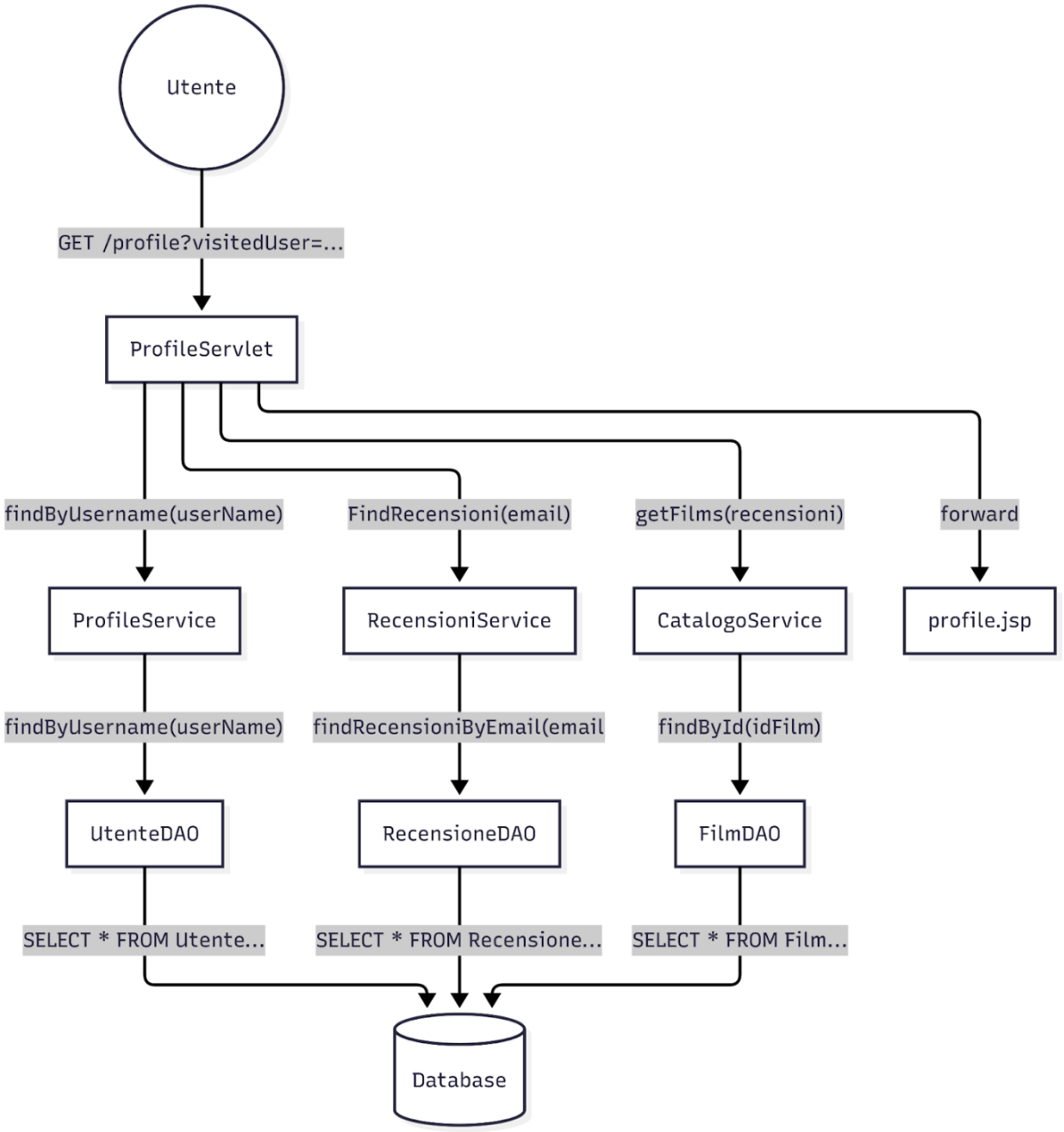
-**VisualizzaFilmServlet**
-**ProfileServlet**

Nota: *Nel sistema Rated ogni DAO invoca getter/setter sui rispettivi Bean; tuttavia, al fine di rendere i seguenti call graph più leggibili, si è scelto di non rappresentare queste chiamate. Ciò è stato tenuto in considerazione durante la fase di Impact Analysis per includere i DAO tra gli elementi impattati.*

Call graph di VisualizzaFilmServlet:



Call graph di ProfileServlet:



Dall’analisi dei call graph esposti, i componenti identificati come direttamente impattati e facenti parte del **SIS** sono i seguenti:

Livello Storage

- **DB.sql**: Aggiornamento dello schema database con l'inserimento delle tabelle dedicate alla persistenza della *watchList* e *watchedList*.
- **VistoBean / InteresseBean**: Creazione delle classi Bean per la modellazione e la gestione dei dati relativi alle liste all’interno del sistema.
- **VistoDAO / InteresseDAO**: Implementazione dei rispettivi DAO per la gestione della persistenza dei dati sul database relativi alla relazione di visto/interesse tra utenti e film del catalogo.

Livello Application Logic

- **ProfileService**: Implementazione di nuovi metodi per la gestione delle liste tramite l'interfacciamento con i rispettivi DAO (InteresseDAO, VistoDAO).
- **VisualizzaFilmServlet**: Aggiornamento della logica di controllo per la gestione dinamica dei pulsanti in film.jsp (es. switch funzionale tra "aggiungi" e "rimuovi" in base alla presenza del film nelle liste).
- **ViewUserFilmsServlet**: Creazione della servlet dedicata alla preparazione dei dati e al controllo del flusso per la visualizzazione di userFilms.jsp.

Livello Interface

- **film.jsp**: Modifica dell’interfaccia per l'integrazione dei pulsanti di aggiunta e rimozione dalle liste.
- **userFilms.jsp**: Creazione della pagina dedicata alla visualizzazione delle liste (*watchList* e *watchedList*) dell’utente.
- **Profile.jsp**: Inserimento del pulsante di navigazione per richiamare la ViewUserFilmsServlet.

3.1.4 Determinazione del Candidate Impact Set (CIS)

Un'analisi più precisa degli elementi del **SIS** ha evidenziato la necessità di modificare anche i file JS e CSS associati alle JSP i quali sono quindi stati inclusi nel **CIS**.

Elementi considerati impattati nel **Candidate Impact Set (CIS):**

Livello Storage

- **DB.sql**: Aggiornamento dello schema database con l'inserimento delle tabelle dedicate alla persistenza di *watchList* e *watchedList*.
- **VistoBean / InteresseBean**: Creazione delle classi Bean per la modellazione e la gestione dei dati relativi alle liste all'interno del sistema.
- **VistoDAO / InteresseDAO**: Implementazione dei rispettivi DAO per la gestione della persistenza dei dati sul database (operazioni CRUD).

Livello Application Logic

- **ProfileService**: Implementazione di nuovi metodi per la gestione delle liste tramite l'interfacciamento con i rispettivi DAO (InteresseDAO, VistoDAO).
- **VisualizzaFilmServlet**: Aggiornamento della logica di controllo per la gestione dinamica dei pulsanti in film.jsp (es. switch funzionale tra "aggiungi" e "rimuovi" in base alla presenza del film nelle liste).
- **ViewUserFilmsServlet**: Creazione della servlet dedicata alla preparazione dei dati e al controllo del flusso per la visualizzazione di userFilms.jsp.

Livello Interface

- **film.jsp**: Modifica dell'interfaccia per l'integrazione dei pulsanti di aggiunta e rimozione dalle liste.
- **userFilms.jsp**: Creazione della pagina dedicata alla visualizzazione delle liste (*watchList* e *watchedList*) dell'utente.
- **Profile.jsp**: Inserimento del pulsante di navigazione per richiamare la ViewUserFilmsServlet.
- **filmFunctions.js** : Aggiunta della gestione dei pulsanti per inserire/rimuovere il film nella watched/watchList e della logica di controllo *(es. se un film è già presente nella watchList deve essere visualizzato il pulsante per rimuoverlo)*

- **userFilmsFunctions.js :** Creazione della logica di corretta gestione del passaggio tra la visualizzazione di watchedList e watchList in userFilms.jsp
- **profileScript.js :** Aggiunta ipotetica della logica legata all'apertura della pagina userFilms.jsp
- **Film.css :** Modifiche relative allo stile dei pulsanti per aggiungere/rimuovere a watched/watchList
- **Profile.css :** Modifiche relative allo stile del pulsante per accedere alla sezione di watched/watchList (userFilms.jsp)
- **UserFilms.css :** Creazione del file di stile per la pagina di visualizzazione di watched/watchList (userFilms.jsp)

3.1.5 Actual Impact Set

Durante l'implementazione un'attenta analisi della pagina del profilo (profile.jsp) e della nuova pagina di visualizzazione per watched/watchList (userFilms.jsp) ha consentito di eliminare l'ipotesi di impatto alla profileScript.js indicata nel **CIS** determinando che l'aggiunta del nuovo pulsante nel profilo non richiede una componente logica JS ad esso associata.

Le componenti effettivamente impattate sono state:

Livello Storage

- **DB.sql**: Aggiornamento dello schema database con l'inserimento delle tabelle dedicate alla persistenza di *watchList* e *watchedList*.
- **VistoBean / InteresseBean**: Creazione delle classi Bean per la modellazione e la gestione dei dati relativi alle liste all'interno del sistema.
- **VistoDAO / InteresseDAO**: Implementazione dei rispettivi DAO per la gestione della persistenza dei dati sul database (operazioni CRUD).

Livello Application Logic

- **ProfileService**: Implementazione di nuovi metodi per la gestione delle liste tramite l'interfacciamento con i rispettivi DAO (InteresseDAO, VistoDAO).
- **VisualizzaFilmServlet**: Aggiornamento della logica di controllo per la gestione dinamica dei pulsanti in film.jsp (es. switch funzionale tra "aggiungi" e "rimuovi" in base alla presenza del film nelle liste).

- **ViewUserFilmsServlet**: Creazione della servlet dedicata alla preparazione dei dati e al controllo del flusso per la visualizzazione di userFilms.jsp.

Livello Interface

- **film.jsp**: Modifica dell'interfaccia per l'integrazione dei pulsanti di aggiunta e rimozione dalle liste.
- **userFilms.jsp**: Creazione della pagina dedicata alla visualizzazione delle liste (*watchList* e *watchedList*) dell'utente.
- **Profile.jsp**: Inserimento del pulsante di navigazione per richiamare la ViewUserFilmsServlet.
- **filmFunctions.js** : Aggiunta della gestione dei pulsanti per inserire/rimuovere il film nella watched/watchList e della logica di controllo *(es. se un film è già presente nella watchList deve essere visualizzato il pulsante per rimuoverlo)*
- **userFilmsFunctions.js :** Creazione della logica di corretta gestione del passaggio tra la visualizzazione di watchedList e watchList in userFilms.jsp
- **Film.css :** Modifiche relative allo stile dei pulsanti per aggiungere/rimuovere a watched/watchList
- **Profile.css :** Modifiche relative allo stile del pulsante per accedere alla sezione di watched/watchList (userFilms.jsp)
- **UserFilms.css :** Creazione del file di stile per la pagina di visualizzazione di watched/watchList (userFilms.jsp)

Tabella di confronto CIS vs AIS

Componente	**CIS**	**AIS**
DB.sql	✓	✓
VistoBean	✓	✓
InteresseBean	✓	✓
VistoDAO	✓	✓
InteresseDAO	✓	✓
ProfileService	✓	✓
VisualizzaFilmServlet	✓	✓
ViewUserFilmsServlet	✓	✓
film.jsp	✓	✓
userFilms.jsp	✓	✓
Profile.jsp	✓	✓
filmFunctions.js	✓	✓
userFilmsFunctions.js	✓	✓
Film.css	✓	✓

Profile.css	✓	✓
UserFilms.css	✓	✓
profileScript.js	✓	

3.1.6 Discovered Impact Set

Non sono emersi componenti non considerati.

3.1.7 False Positive Impact Set (FPIS)

Il False Positive Impact Set (FPIS) è definito come

$$FPIS = CIS \setminus AIS$$

ossia l'insieme dei componenti previsti come impattati (CIS) ma risultati non necessari dopo l'implementazione (AIS).

Per la CR1: FPIS = {profileScript.js}. In fase di implementazione si è stabilito che la logica di gestione della navigazione/pulsante può essere gestita direttamente nella JSP senza introdurre ulteriore script dedicato.

3.1.8 Metriche di valutazione (Precision e Recall)

Considerando i componenti unici:

|CIS| = 17,

|AIS| = 16

|CIS ∩ AIS| = 16

Le metriche **Recall** e **Precision** sono utilizzate per valutare l'efficacia del processo di impact analysis, misurando quanto il Candidate Impact Set (CIS) rappresenti correttamente gli impatti reali (AIS). In particolare, **Recall** indica la copertura degli impatti effettivi, mentre **Precision** quantifica la quota di elementi del CIS che risultano realmente impattati.

$$Recall = \frac{|CIS \cap AIS|}{|AIS|}$$

La **Recall** è uguale ad **1** nel nostro caso (**16/16**), riflette ovvero un Discovered Impact Set vuoto.

$$\text{Precision} = \frac{|CIS \cap AIS|}{|CIS|}$$

Nel nostro caso, la **Precision** risulta **≈ 0.94**, poiché **16 elementi su 17** presenti nel **CIS** sono effettivamente impattati (**AIS**) (ossia **16/17**): ciò indica che il processo di impact analysis ha prodotto un insieme candidato molto accurato, con **un solo falso positivo** (un elemento incluso nel CIS ma non realmente impattato)

3.2 Change Request 2

La change request 2 consiste nell'integrazione di un sistema di suggerimento film personalizzato, basato sulle preferenze dell'utente inoltre tale sistema dovrà tenere conto dei film già visti e/o in watchList dall'utente e prevedere la possibilità di interagire con tale film con non mi interessa e aggiungi alla watchList. La modifica prevede la ristrutturazione della HomePage attraverso l'inserimento di una sezione dedicata ai titoli consigliati, calcolati in base ai generi preferiti espressi dall'utente e allo storico dei film già visualizzati. Le preferenze dell'utente potranno essere indicate durante la fase di registrazione e visualizzate, con possibilità di modifica, nella pagina profilo.

3.2.1 Analisi della richiesta di Modifica

La richiesta di modifica si configura come un intervento di manutenzione evolutiva, in quanto introduce funzionalità precedentemente assenti nel sistema. Rispetto alla consultazione statica del catalogo, questa modifica richiede l'introduzione di una relazione specifica tra gli utenti e le categorie cinematografiche (Generi), rendendo necessaria la gestione della persistenza di tali preferenze. Nello specifico, l'implementazione di queste nuove funzionalità comporterà i seguenti interventi:

- **Gestione delle Preferenze Utente:** Modifica della pagina di registrazione e della relativa logica di controllo per consentire all'utente di selezionare i generi di interesse durante la registrazione. Inoltre dovrà essere integrata nella pagina utente la visualizzazione di tali preferenze e una sezione per la loro modifica post-registrazione.
- **Evoluzione della HomePage:** Ristrutturazione dell'interfaccia principale per gli utenti autenticati, con l'aggiunta di una sezione dinamica dedicata ai suggerimenti e l'integrazione tramite i pulsanti trigger "Aggiungi alla watchList" e "Non mi interessa" sui singoli film.

- **Gestione della Logica di Stato:** Implementazione di una query di selezione che filtri il catalogo incrociando i generi preferiti con i generi dei film non ancora visti, gestendo al contempo l'esclusione dei titoli contrassegnati come "Non di interesse" per evitare ridondanze.
- **Persistenza dei Dati:** Modifica del database e del modello applicativo per gestire la relazione "molti-a-molti" tra utenti e generi preferiti e una ristrutturazione mirata su come viene memorizzato il genere di un film andando a creare una tabella “genere” la quale sarà caricata di default con tutti i generi gestiti dal sistema ed inoltre sarà creata un’ulteriore nuova tabella “GenereFilm” per mettere in relazione un film con i propri generi, tali modifiche richiederanno la creazione dei relativi oggetti Bean e DAO necessari alla manipolazione dei dati.

Al fine di integrare correttamente queste funzionalità, si procederà con un'analisi degli artefatti impattati seguendo un approccio top-down: partendo dalla revisione dei casi d'uso e della documentazione, si arriverà a determinare i cambiamenti strutturali nel database e nelle classi di logica applicativa.

3.2.2 Impact Analysis

Come esplicitamente richiesto dalla change request è necessario andare ad aggiornare il database per ospitare le tabelle inerenti alla watchList e alla watchedList. Nello specifico, analizzando **DB.sql** e il rispettivo **schema ER** si è effettuato l’Impact Analysis prevedendo che le tabelle saranno aggiunte e/o modificate nel seguente modo:

```
CREATE TABLE Film (
  ID_Film INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
  Locandina LONGBLOB,
  Nome VARCHAR(255) NOT NULL,
  Anno YEAR NOT NULL,
  Durata INT NOT NULL,
  Regista VARCHAR(255),
  Trama VARCHAR(255),
  Valutazione INT DEFAULT 1 CHECK (Valutazione BETWEEN 1 AND 5),
  Attori TEXT
);
```

Rimozione della colonna Genere (precedentemente VARCHAR). Il genere viene ora debitamente gestito come di seguito.

```
CREATE TABLE Genere (
   Nome VARCHAR(50) NOT NULL PRIMARY KEY
);

CREATE TABLE Film_Genere (
   ID_Film INT NOT NULL,
   Nome_Genere VARCHAR(50) NOT NULL,
   PRIMARY KEY (ID_Film, Nome_Genere),
   FOREIGN KEY (ID_Film) REFERENCES Film(ID_Film)
   FOREIGN KEY (Nome_Genere) REFERENCES Genere(Nome)
);
```

Permette l'associazione di multipli film a molteplici generi.

```
CREATE TABLE Preferenza (
   email VARCHAR(255) NOT NULL,
   Nome_Genere VARCHAR(50) NOT NULL,
   PRIMARY KEY (email, Nome_Genere),
   FOREIGN KEY (email) REFERENCES Utente_Registrato(email)
   FOREIGN KEY (Nome_Genere) REFERENCES Genere(Nome)
);
```

Mappa gli interessi degli utenti registrati verso specifici generi presenti a sistema.

Si considera quindi come parte delle componenti impattate **DB.sql**

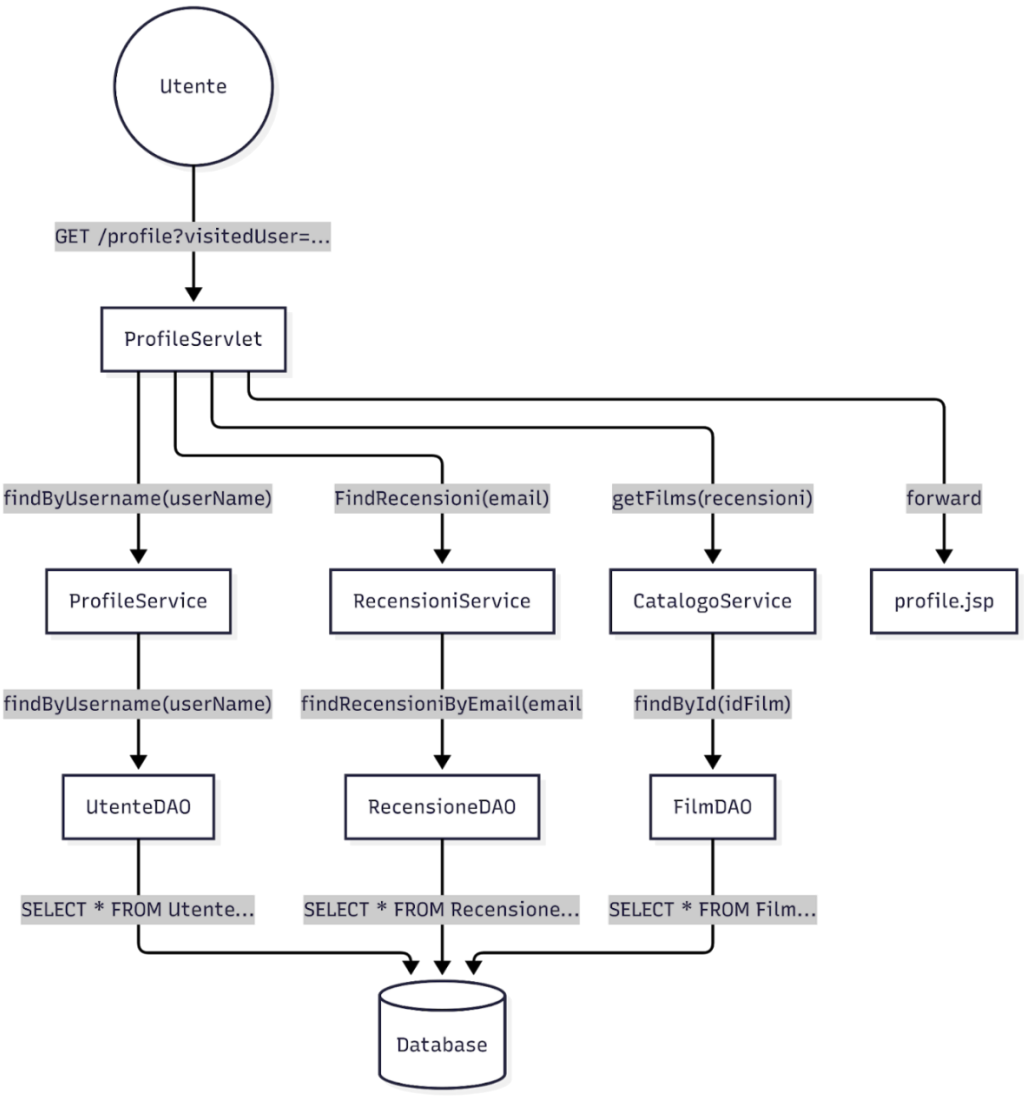
3.2.3 Determinazione dello Starting Impact Set (SIS).

Attraverso uno studio dell'architettura del sistema descritto nella documentazione, si è notato che ogni funzionalità offerta dalla piattaforma è erogata tramite una servlet di partenza. Nello specifico date le modifiche inerenti alla change request si sono identificate le seguenti servlet il quale studio dei rispettivi call graph dovrebbe esporre le componeti aventi bisogno di modifiche:

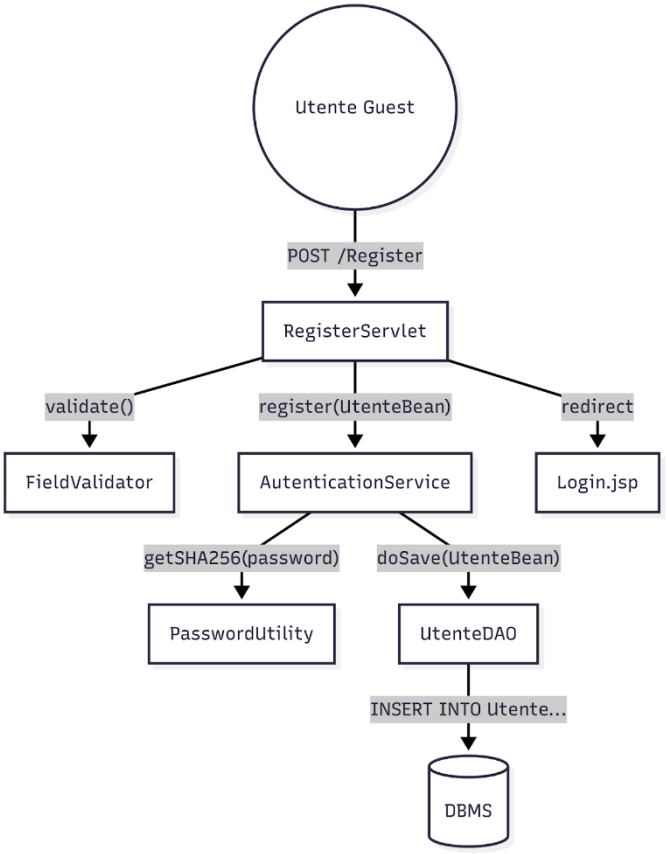
-**ProfileServlet**
-**RegisterServlet**

Nota: *Nel sistema Rated ogni DAO invoca getter/setter sui rispettivi Bean; tuttavia, al fine di rendere i seguenti call graph più leggibili, si è scelto di non rappresentare queste chiamate. Ciò è stato tenuto in considerazione durante la fase di Impact Analysis per includere i DAO tra gli elementi impattati.*

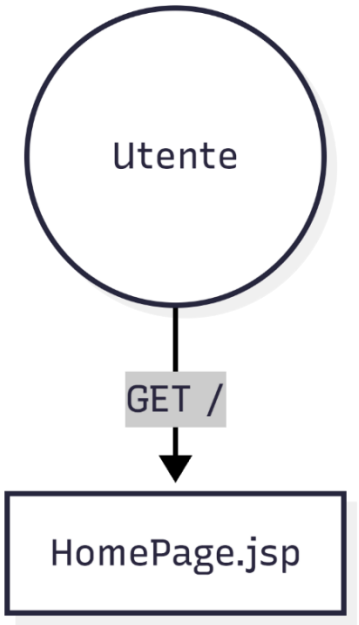
Call graph di ProfileServlet



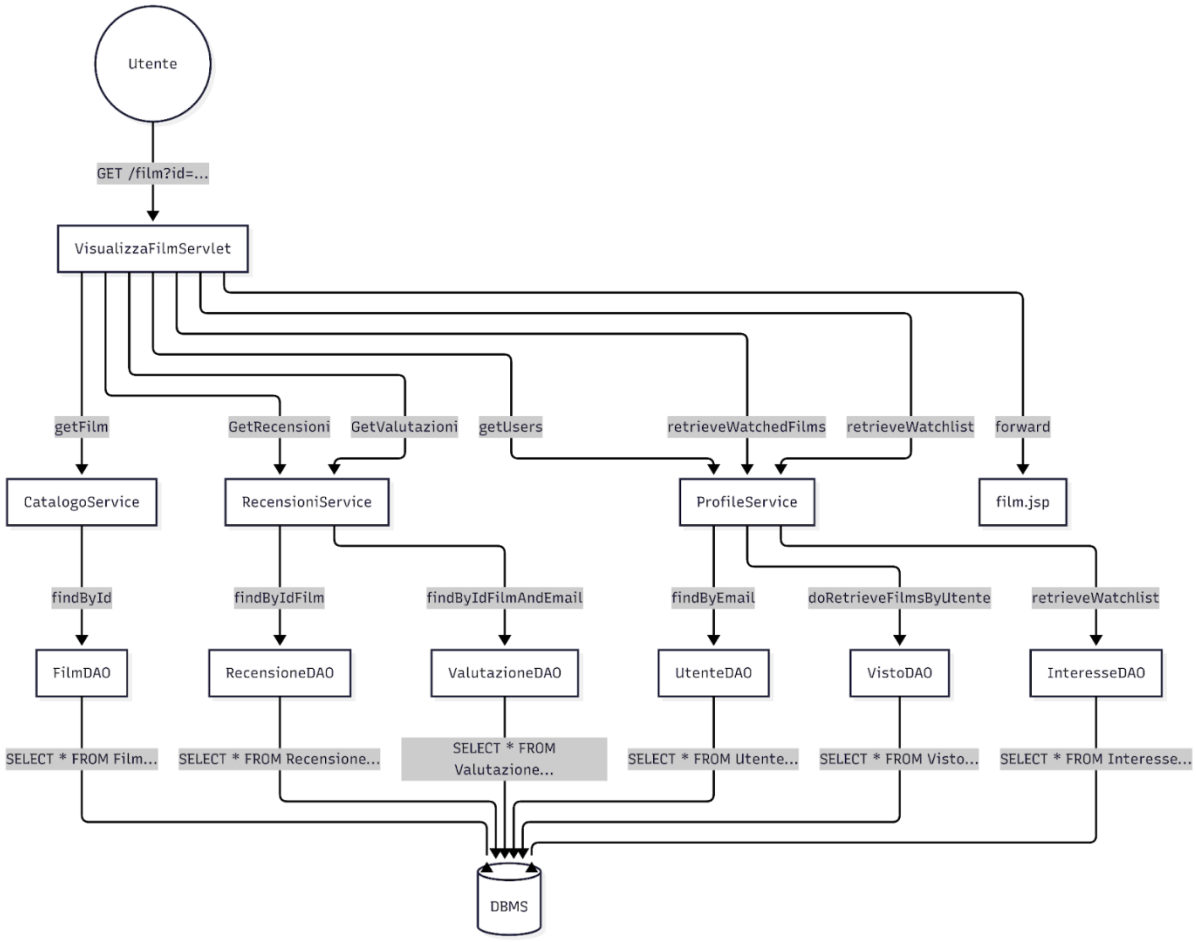
Call graph di RegisterServlet



Call graph di accesso alla HomePage



Call Graph di VisualizzaFilmServlet



Dall'analisi dei call graph esposti, i componenti identificati come direttamente impattati e facenti parte del **SIS** sono i seguenti:

Livello Storage

- **DB.sql:** Necessario per l'aggiornamento dello schema e la migrazione dei dati dei generi.
- **PreferenzaBean / GenereBean:** Creazione degli oggetti Java per mappare le nuove entità del database.
- **PreferenzaDAO / GenereDAO:** Implementazione dei metodi CRUD per gestire le preferenze degli utenti e l'associazione generi-film.
- **FilmDAO:** Aggiornamento dei metodi di recupero film per supportare il nuovo schema Film_Genere e la query di raccomandazione.

Livello Application Logic

- **RegisterServlet:** Integrazione della logica per acquisire i generi selezionati dall'utente durante la creazione dell'account e salvarli tramite PreferenzaDAO.
- **ProfileServlet:** Modifica per recuperare insieme ai dati dell'utente anche la lista dei suoi generi preferiti dal database.
- **ProfileService:** Implementazione dei metodi per consentire la modifica dei generi scelti dall'utente.
- **HomePageServlet:** Creazione della servlet per gestire il caricamento della pagina principale, includendo la logica di recupero dei "Film Consigliati" filtrati per l'utente loggato.

Livello Interface

- **register.jsp:** Inserimento di una sezione (es. checkbox list) per la selezione dei generi preferiti.
- **profile.jsp:** Aggiunta di un'interfaccia di visualizzazione e modifica per le preferenze salvate.
- **homePage.jsp:** Ristrutturazione del layout per ospitare la nuova sezione "Consigliati per te" con i relativi pulsanti di interazione.

3.2.4 Determinazione del Candidate Impact Set (CIS)

Un'analisi più precisa degli elementi del **SIS** ha evidenziato la necessità di modificare anche i file JS e CSS associati alle JSP i quali sono quindi stati inclusi nel **CIS**.

Elementi considerati nel **Candidate Impact Set (CIS):**

Livello Storage

- **DB.sql:** Necessario per l'aggiornamento dello schema e la migrazione dei dati dei generi.
- **PreferenzaBean / GenereBean:** Creazione degli oggetti Java per mappare le nuove entità del database.
- **PreferenzaDAO / GenereDAO:** Implementazione dei metodi CRUD per gestire le preferenze degli utenti e l'associazione generi-film.

- **FilmDAO:** Aggiornamento dei metodi di recupero film per supportare il nuovo schema Film_Genere e la query di raccomandazione.

Livello Application Logic

- **RegisterServlet:** Integrazione della logica per acquisire i generi selezionati dall'utente durante la creazione dell'account e salvarli tramite PreferenzaDAO.
- **ProfileServlet:** Modifica per recuperare insieme ai dati dell'utente anche la lista dei suoi generi preferiti dal database.
- **ProfileService:** Implementazione dei metodi per consentire la modifica dei generi scelti dall'utente.
- **HomePageServlet:** Creazione della servlet per gestire il caricamento della pagina principale, includendo la logica di recupero dei "Film Consigliati" filtrati per l'utente loggato.

Livello Interface

- **register.jsp:** Inserimento di una sezione (es. checkbox list) per la selezione dei generi preferiti.
- **profile.jsp:** Aggiunta di un'interfaccia di visualizzazione e modifica per le preferenze salvate.
- **homePage.jsp:** Ristrutturazione del layout per ospitare la nuova sezione "Consigliati per te" con i relativi pulsanti di interazione.
- **registerValidation.js:** Aggiornamento per validare che l'utente selezioni almeno un numero minimo di generi preferiti.
- **homePageRecs.js:** Creazione dello script per gestire via AJAX i trigger "Non mi interessa" e "Aggiungi alla Watchlist" senza ricaricare l'intera pagina.
- **profileScript.js:** Inizialmente considerato per la gestione della modifica generi nel profilo.
- **Home.css / Register.css / Profile.css:** Aggiornamento degli stili per ospitare i nuovi elementi grafici (grid dei consigliati, checkbox dei generi).

3.2.5 Actual Impact Set

In fase di implementazione si è stabilito che **profileScript.js** non necessita di modifiche in quanto la gestione della selezione generi nel profilo può essere gestita direttamente tramite form standard nella jsp.

Le componenti effettivamente impattate sono state:

Livello Storage

- **DB.sql:** Modifica strutturale tabelle.
- **PreferenzaBean / GenereBean:** Modellazione dati.
- **PreferenzaDAO / GenereDAO:** Persistenza relazioni.
- **FilmDAO:** Query di recomendazione dei film.

Livello Application Logic

- **RegisterServlet:** Persistenza preferenze in fase di iscrizione.
- **ProfileService:** Logica di business per l'aggiornamento dei gusti utente.
- **HomePageServlet:** Recupero dati per la sezione consigliati.

Livello Interface

- **register.jsp / profile.jsp / homePage.jsp:** Aggiornamenti UI per mostrare i nuovi dati.
- **registerValidation.js:** Controllo input lato client per i generi.
- **homePageRecs.js:** Gestione asincrona dei feedback sui suggerimenti (Interesse/Watchlist).
- **CSS (Home, Register, Profile):** Adeguamento stilistico per i nuovi componenti.

Tabella di confronto CIS vs AIS

Componente	CIS	AIS
DB.sql	✓	✓
PreferenzaBean	✓	✓
GenereBean	✓	✓
PreferenzaDAO	✓	✓
GenereDAO	✓	✓
FilmDAO	✓	✓
RegisterServlet	✓	✓

ProfileServlet	✓	✓
ProfileService	✓	✓
HomePageServlet	✓	✓
register.jsp	✓	✓
profile.jsp	✓	✓
homePage.jsp	✓	✓
registerValidation.js	✓	✓
homePageRecs.js	✓	✓
Home.css	✓	✓
Register.css	✓	✓
Profile.css	✓	✓
profileScript.js	✓	

3.2.6 Discovered Impact Set

Non sono emersi componenti non considerati.

3.2.7 False Positive Impact Set (FPIS)

Per la definizione del FPIS si rimanda alla **Sezione 3.1.8**.

Per la CR2: FPIS = {profileScript.js}. Durante l'implementazione si è stabilito che non sono richieste modifiche a tale componente.

3.2.8 Metriche di valutazione (Precision e Recall)

Per la definizione delle metriche e delle relative formule si rimanda alla **Sezione 3.1.8**.

Considerando i componenti unici :

|CIS| = 19,

|AIS| = 18

|CIS ∩ AIS| = 18

Ne consegue che la **Recall = 1 (18/18, DIS vuoto)**, mentre la **Precision ≈ 0.95 (18/19)**.

Questi valori indicano che l'approccio adottato è stato **efficace**: la Recall pari a 1 mostra che il CIS ha incluso tutti i componenti effettivamente impattati, evitando omissioni. La Precision elevata conferma inoltre che la validazione ha anche qui contenuto i falsi positivi, limitandoli a un solo componente incluso nel CIS ma non realmente necessario da modificare.