ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DIPARTIMENTO DI INFORMATICA - SCIENZA e INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Analisi, Progettazione e Distribuzione in Cloud di applicativo multipiattaforma per l'organizzazione di eventi condivisi e la condivisione multimediale automatica in tempo reale

Relatore:
Chiar.mo Prof.
Michele Colajanni

Presentata da:
Giacomo Romanini

Sessione Luglio 2025

Anno Accademico 2025/2026

Abstract

Lo sviluppo di un applicativo multipiattaforma diretto all'organizzazione di eventi condivisi, caratterizzato in particolare dalla condivisione multimediale in tempo reale, richiede opportune capacità di scalabilità, atte a garantire una risposta efficace anche con alti volumi di richieste, offrendo prestazioni ottimali. Le tecnologie cloud, con la loro disponibilità pressoché illimitata di risorse e la completa e continua garanzia di manutenzione, offrono l'architettura ideale per il supporto di simili progetti, anche con fondi limitati.

Tuttavia, l'integrazione tra la logica applicativa ed i molteplici servizi cloud, insieme alla gestione delle loro interazioni reciproche, comporta sfide specifiche, in particolare legate all'ottimizzazione di tutte le risorse. L'individuazione e la selezione delle soluzioni tecnologiche più adatte per ogni obiettivo, così come l'adozione delle migliori pratiche progettuali, devono procedere parallelamente con lo sviluppo del codice, al fine di sfruttare efficacemente le potenzialità offerte.

In tale prospettiva, questa tesi illustra le scelte progettuali ed implementative adottate nello sviluppo dell'applicativo in questione, evidenziando l'impatto dell'integrazione delle risorse cloud sul risultato finale.

Indice

Introduzione **1**

Organizzazione dei capitoli 3

0.1 L'analisi del problema 4

0.1.1 Analisi delle funzionalità 4

0.1.2 L'ideazione dell'architettura logica 8

Introduzione

In un contesto sociale sempre più connesso, la crescente quantità di contatti, la rapidità delle comunicazioni e l'accesso universale alle informazioni rendono la ricerca, l'organizzazione e la partecipazione ad eventi estremamente facile, ma al contempo generano un ambiente frenetico e spesso dispersivo.

Risulta infatti difficile seguire tutte le opportunità a cui si potrebbe partecipare, considerando le numerose occasioni che si presentano quotidianamente. Basti pensare, ad esempio, alle riunioni di lavoro, alle serate con amici, agli appuntamenti informali per un caffè, ma anche a eventi più strutturati come fiere, convention aziendali, concerti, partite sportive o mostre di artisti che visitano occasionalmente la città.

Questi eventi possono sovrapporsi, causando dimenticanze o conflitti di pianificazione, con il rischio di delusione o frustrazione. Quando si è invitati a un evento, può capitare di essere già impegnati, o di trovarsi in attesa di una conferma da parte di altri contatti. In questi casi, la gestione degli impegni diventa complessa: spesso si conferma la partecipazione senza considerare possibili sovrapposizioni, o dimenticandosi, per poi dover scegliere e disdire all'ultimo momento.

D'altra parte, anche quando si desidera proporre un evento, la ricerca di un'attività interessante può diventare un compito arduo, con la necessità di consultare numerosi profili social di locali e attività, senza avere inoltre la certezza che gli altri siano disponibili. Tali problemi si acuiscono ulteriormente quando si tratta di organizzare eventi di gruppo, dove bisogna allineare gli impegni di più persone.

In questo contesto, emergono la necessità e l'opportunità di sviluppare uno strumento che semplifichi la proposta e la gestione degli eventi, separando il momento della proposta da quello della conferma di partecipazione. In tal modo, gli utenti possono valutare la disponibilità degli altri prima di impegnarsi definitivamente, facilitando in conteporanea sia l'invito sia la partecipazione.

In risposta a tali richieste è stata creata Wyd, un'applicazione che permette agli utenti di organizzare i propri impegni, siano essi confermati oppure proposti. Essa permette anche di rendere più intuitiva la ricerca di eventi attraverso la creazione di uno spazio virtuale centralizzato dove gli utenti possano pubblicare e consultare tutti gli eventi disponibili, diminuendo l'eventualità di perderne qualcuno. La funzionalità chiave di questo progetto si fonda sull'idea di affiancare alla tradizionale agenda degli impegni confermati un calendario separato, che mostri tutti gli eventi a cui si potrebbe partecipare.

Una volta confermata la partecipazione a un evento, questo verrà spostato automaticamente nell'agenda personale dell'utente. Gli eventi creati potranno essere condivisi con persone o gruppi, permettendo di visualizzare le conferme di partecipazione. Considerando l'importanza della condivisione di contenuti multimediali, questo progetto prevede la possibilità di condividere foto e video con tutti i partecipanti all'evento, attraverso la generazione di link per applicazioni esterne o grazie all'ausilio di gruppi di profili. Al termine dell'evento, l'applicazione carica automaticamente le foto scattate durante l'evento, per allegarle a seguito della conferma dell'utente.

Figura 1: Il logo di Wyd

La realizzazione di un progetto come Wyd implica la risoluzione e la gestione di diverse problematiche tecniche. In primo luogo, la stabilità del programma deve essere garantita da un'infrastruttura affidabile e scalabile. La persistenza deve essere modellata per fornire alte prestazioni sia in lettura che in scrittura indipendentemente dalla quantità delle richieste, rimanendo però aggiornata e coerente. La funzionalità di condivisione degli eventi richiede inoltre l'aggiornamento in tempo reale verso tutti gli utenti coinvolti. Infine, il caricamento ed il salvataggio delle foto aggiungono la necessità di gestire richieste di archiviazione di dimensioni significative.

Organizzazione dei capitoli

Il seguente elaborato è suddiviso in cinque capitoli.
Nel primo capitolo si affronta la fase di analisi delle funzionalità, durante la quale, partendo dall'idea astratta iniziale, si definiscono i requisiti e le necessità del sistema, per poi creare la struttura generale ad alto livello dell'applicazione.
Nel secondo capitolo si affrontano le principali scelte architetturali e di sviluppo che hanno portato a definire la struttura centrale dell'applicazione.
Il terzo capitolo osserva lo studio effettuato per gestire la memoria, in quanto fattore che più incide sulle prestazioni. Particolare attenzione è stata dedicata, infatti, a determinare le tecnologie e i metodi che meglio corrispondono alle esigenze derivate dal salvataggio e dall'interazione logica degli elementi.
Il quarto capitolo si concentra sulle scelte implementative adottate per l'inserimento le funzionalità legate alla gestione delle immagini, che, oltre ad introdurre problematiche impattanti sia sulle dimensioni delle richieste sia sull'integrazione con la persistenza, richiedono l'automatizzazione del recupero delle immagini.
Infine, nel quinto capitolo, verranno analizzati e discussi i risultati ottenuti testando il sistema.

0.1 L'analisi del problema

A seguito dell'identificazione dei requisiti e dei casi d'uso, l'analisi del problema entra nel merito del comportamento dell'applicazione, evidenziandone il rapporto con le funzionalità. Determina quindi l'architettura logica, che delinea le relazioni fondamentali del sistema, individuando i componenti logici principali e le loro responsabilità.

0.1.1 Analisi delle funzionalità

Le funzionalità vengono dedotte dai casi d'uso, sintetizzando i servizi principali dell'applicazione. In particolare, le funzionalità vengono rilevate in base alla loro relazione con i casi d'uso, alla specificità del compito che assolvono e alla pertinenza reciproca.

Si riportano in tabella le funzionalità che racchiudono altri casi d'uso.
In particolare, VisualizzaEvento permette di accedere alla maggior parte delle azioni che l'utente può attuare sugli eventi. Allo stesso modo GestioneGruppi e GestioneProfili permettono di eseguire le azioni correlate al loro contesto.

Funzionalità	**Scomposizione**
EventiConfermati	VisualizzaEvento
EventiProposti	VisualizzaEvento
VisualizzaEvento	CreaEvento, ModificaEvento, ConfermaEvento, DisdiciEvento, CondividiConLink, CondividiAiGruppi, CaricaImmagini, EliminaImmagini, ConfermaImmagini
GestioneGruppi	CercaProfili, AggiungiProfiloAlGruppo, CreaGruppo
CercaProfili	AggiungiProfilo
GestioneProfili	CambiaProfilo

Tabella 1: Scomposizione delle funzionalità

Di ogni funzionalità vengono evidenziati il grado di complessità, la tipologia di azione che svolgono e i requisiti collegati. Il grado di complessità riassume la quantità e la difficoltà implementativa delle azioni che una funzionalità ricopre. La tipologia riporta in maniera generale la qualità dei servizi offerti. Infine si riportano gli identificatori dei requisiti funzionali che ogni funzionalità soddisfa.

A parte il Login, la Registrazione e ScritturaLog, il cui servizio è diretto e uniforme in tutta l'applicazione, tutte le altre funzionalità prevedono una gestione e manipolazione di più dati, a volte in strutture complicate, a volte permettendo una modifica puntuale delle informazioni interessate.

Funzionalità	**Tipo**	**Grado di complessità**	**Requisiti Collegati**
Login	Interazione esterno e lettura dati	semplice	R2F
Registrazione	Interazione esterno e memorizzazione dati	semplice	R1F
EventiConfermati	Interazione esterno e gestione dati	complessa	R3F, R8F
EventiProposti	Interazione esterno e gestione dati	complessa	R4F, R9F
GestioneGruppi	Interazione esterno e gestione dati	complessa	R15F, R16F
GestioneProfili	Interazione esterno e gestione dati	complessa	R17F, R18F, R19F
VisualizzaEvento	Interazione esterno e gestione, lettura e memorizzazione dati	complessa	R5F, R6F, R7F, R8F, R9F, R10F, R11F, R12F, R14F
AggiornaEvento	Gestione dati	complessa	R20F
RecuperaImmagini	Lettura dati	complessa	R13F
ScritturaLog	Memorizzazione dati	semplice	R21F

Tabella 2: Funzionalità

Si procede analizzando i dati che ogni funzionalità gestisce, indicandone la tipologia, la protezione richiesta e i vincoli correlati, per conoscere in maniera definitiva tutte le caratteristiche delle informazioni scambiate. L'analisi delle informazioni non viene riportata in quanto poco rilevante ai fini della tesi, ma eventuali dettagli saranno riportati quando necessario.

A seguito dell'analisi delle informazioni, si procede con l'analisi dei vincoli, in cui si chiarificano i requisiti non funzionali, evidenziandone le criticità e quali componenti ne vengono coinvolti.

Requisito	**Categorie**	**Impatto**	**Funzionalità**
Semplicità dell'interfaccia	Usabilità	Intuitività di utilizzo	Login, Registrazione, EventiConfermati, EventiProposti, GestioneGruppi, GestioneProfili, VisualizzaEvento, RecuperaImmagini
Velocità della ricerca dei dati	Tempo di Risposta	Maggiore reattività	EventiConfermati, EventiProposti, GestioneGruppi, GestioneProfili, RecuperaImmagini
Velocità di memorizzazione dei dati	Tempo di Risposta	Maggiore reattività	Registrazione, AggiornaEvento, RecuperaImmagini
Controllo Accessi	Sicurezza	Peggiorano tempo di risposta e usabilità, migliorano la privacy dei dati	EventiConfermati, EventiProposti, GestioneGruppi, GestioneProfili, VisualizzaEvento
Protezione dei Dati	Sicurezza	Peggiorano tempo di risposta, migliorano la privacy dei dati	Login, Registrazione, EventiConfermati, EventiProposti, GestioneGruppi, GestioneProfili, VisualizzaEvento, AggiornaEvento, RecuperaImmagini

Requisito	**Categorie**	**Impatto**	**Funzionalità**
Scalabilità delle richieste	Tempo di Risposta	Minor degradamento delle prestazioni	EventiConfermati, EventiProposti, AggiornaEvento, RecuperaImmagini

Tabella 3: Analisi dei vincoli

Infine si definiscono logicamente le maschere, ovvero i componenti visuali essenziali del programma. A ogni maschera corrisponderà un'interfaccia grafica attraverso la quale l'utente potrà accedere alle funzionalità. Vengono quindi associate le maschere alle funzionalità di cui permettono l'esecuzione, indicando le informazioni relative.

Maschera	**Informazioni**	**Funzionalità**
View Login	email, password	Login
View Registrazione	email, password	Registrazione
View EventiConfermati	lista eventi confermati	EventiConfermati, AggiornaEvento
View EventiProposti	lista eventi proposti	EventiProposti, AggiornaEvento
View VisualizzaEvento	Identificativo utente, titolo, descrizione, data e orario di inizio, data e orario di fine, confermato, immagini, profili associati	VisualizzaEvento, RecuperaImmagini
View GestioneGruppi	lista gruppi	GestioneGruppi
View CercaProfili	tag di ricerca, lista profili	CercaProfili
View GestioneProfili	Lista profili, Identificativo utente, Identificativo profilo corrente	GestioneProfili

Tabella 4: Maschere

0.1.2 L'ideazione dell'architettura logica

Definite le relazioni e le informazioni relative alle funzionalità, si esprimono logicamente i componenti principali del sistema e le loro relazioni. Le funzionalità vengono espresse a livello logico tramite package e diagrammi delle classi, mentre i dati vengono descritti all'interno del dominio.

Il modello del dominio individua le entità che rappresentano logicamente le dipendenze tra i dati. Ogni entità presenta i suoi dati tramite proprietà, identificate da un nome e dalla tipologia del dato. Inoltre, all'interno del modello vengono indicati i rapporti tra le entità specificando le cardinalità reciproche.

Il dominio di Wyd si concentra attorno a due entità principali: Event e Profile.
Gli Event contengono tutti i dati generali degli eventi, quali l'ora d'inizio e l'ora di fine, il titolo e la descrizione. Strettamente correlate a loro ci sono le immagini, identificate con Photo. Ogni Event può avere più Photo, ma una Photo può essere relativa da un solo Event. I Profile racchiudono i dati dei profili, che devono essere identificati da un Tag unico in tutto il programma. Group rappresenta un gruppo. Più Profile possono fare parte di un Group, ma, allo stesso modo, un Profile può appartenere a più Group.

L'Account memorizza le informazioni attraverso cui l'utente può accedere e agire in quanto User, entità che descrive l'utente. L'utente ha diverse modalità di accesso, ed è per questo motivo che più Account possono essere relativi allo stesso User. Lo stesso utente può impersonare più profili, e un profilo può essere gestito da più utenti. User e Profile sono quindi connessi in una relazione molti a molti.

La maggior parte delle operazioni avrà a che fare con le due entità principali, da cui l'importanza dell'elemento che ne descrive la relazione, ovvero ProfileEvent.
ProfileEvent ha un ruolo centrale in quanto sarà l'unità interrogata sia quando si vorranno ottenere gli eventi di un determinato profilo, sia quando bisognerà recuperare i profili relativi a un evento. Contiene tutte le informazioni relative al particolare profilo sul determinato evento, rendendolo infatti l'entità più modificata di tutto il progetto.

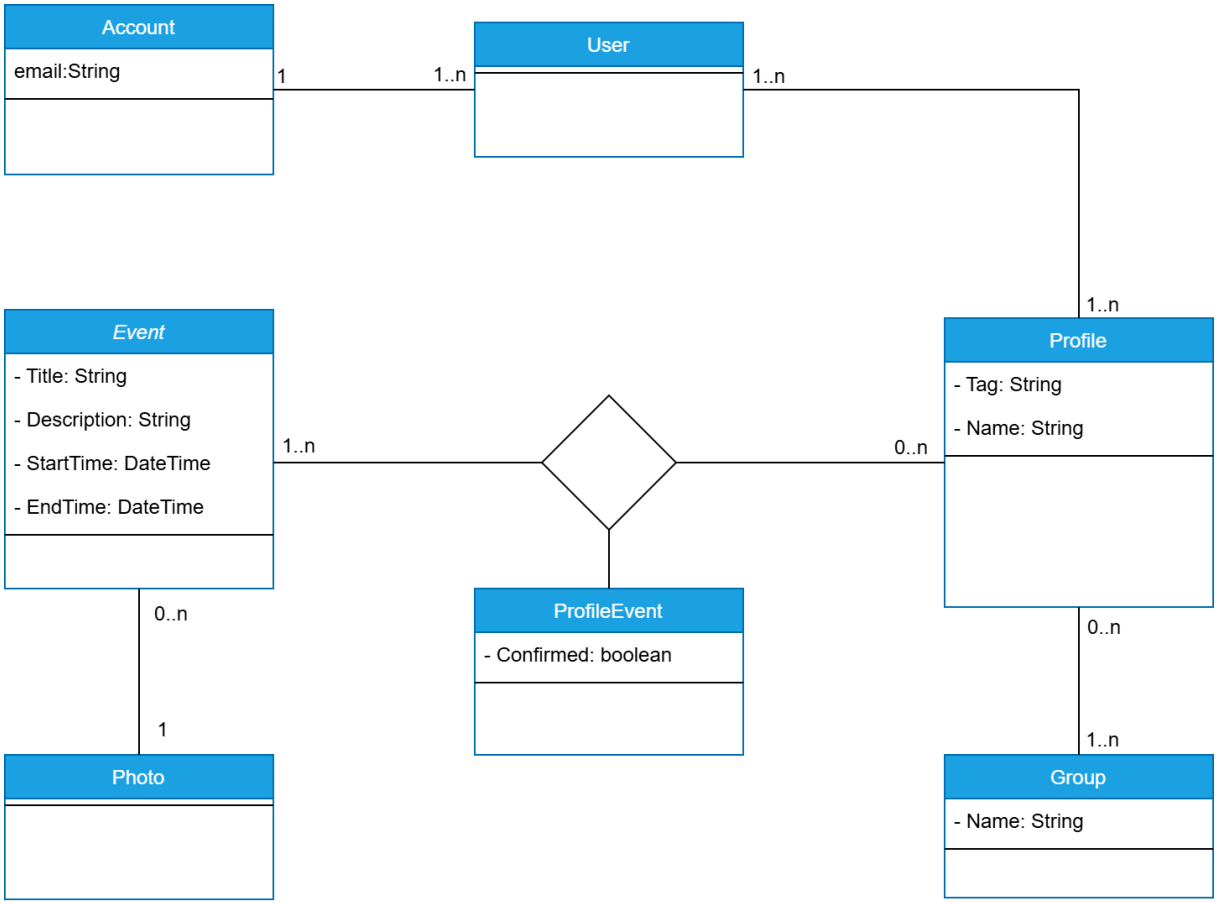

Figura 2: Modello del dominio

Il diagramma dei package descrive la divisione delle responsabilità logiche.
Ogni package rappresenta una parte di prodotto che soddisfa una determinata responsabilità. La responsabilità viene individuata in base alla peculiarità e alle dipendenze delle funzionalità che ricopre. Si possono così distinguere, ad esempio, package relativi a interfacce grafiche, logiche applicative o gestione della persistenza, in base alle caratteristiche specifiche del prodotto. Il diagramma dei package offre una prima struttura delle parti del progetto e del loro rapporto.

Distinguiamo i diversi package in base al loro scopo.
InterfacciaAccesso e InterfacciaUtente sono le parti che si occuperanno dell'interazione grafica con l'utente. InterfacciaAccesso deve presentare le schermate di login e di registrazione, e tutti i passaggi intermedi che saranno necessari. InterfacciaUtente si occuperà di mostrare le viste per interagire con il resto delle funzionalità dell'applicazione.

Il package del Dominio contiene la persistenza principale del progetto, con tutti i dati delle entità del dominio. Il package delle Immagini contiene i file multimediali, recuperabili tramite i metadati salvati sul Dominio. Il package Log riceve e salva le informazioni relative alle richieste svolte dai vari servizi.

GestioneAccesso segue la logica per autenticare gli utenti, collaborando con InterfacciaAccesso e il Dominio, per indirizzare l'utente a InterfacciaUtente in caso di login positivo. GestioneProfilo è il package che racchiude le funzionalità applicative del progetto. Si occupa quindi d'implementare tutta la logica relativa agli eventi, ai profili e alle loro interazioni. GestioneAggiornamenti si occupa infine di connettere GestioneProfilo e InterfacciaUtente per trasmettere le modifiche in tempo reale.

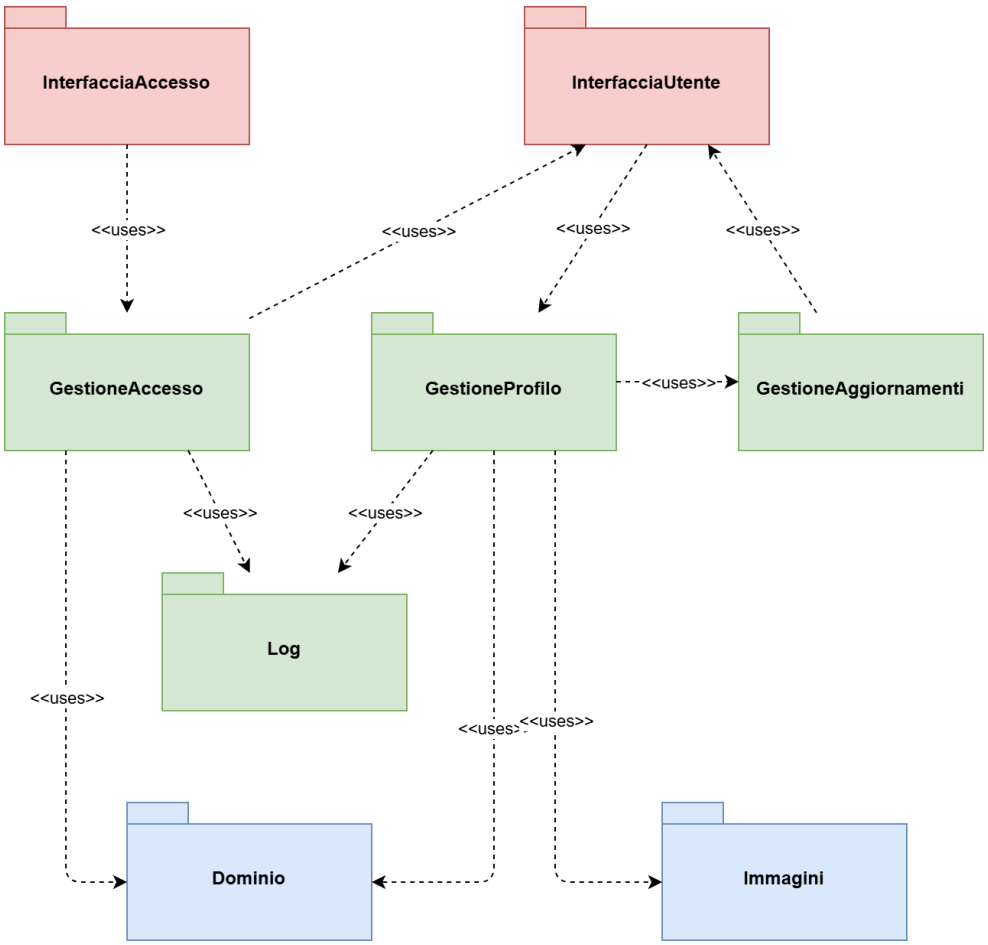

Figura 3: Diagramma dei Package

Ogni package contiene una o più classi che lo implementano.
Ogni classe rappresenta un componente logico che assume uno specifico scopo. Le classi possono presentare dei metodi, ovvero delle istanze che descrivono le funzionalità fornite, alle quali altri componenti possono fare richiesta di esecuzione. La definizione delle classi permette di creare una struttura iniziale presentando le funzionalità minime e le dipendenze tra le parti.

InterfacciaUtente ha una classe per ogni funzionalità principale. Ci sono quindi le classi di ViewEventiConfermati e ViewEventiProposti, che permettono di visualizzare i relativi impegni. Attraverso queste interfacce si può interagire con ViewVisualizzaEvento, per interagire con i dettagli dell'evento. ViewGestioneGruppi presenta la lista dei gruppi con le azioni relative e permette l'accesso a ViewCercaProfili, la schermata per trovare altri profili all'interno dell'applicazione. ViewGestioneProfili è la classe che visualizza i profili dell'utente e ne permette il cambio.

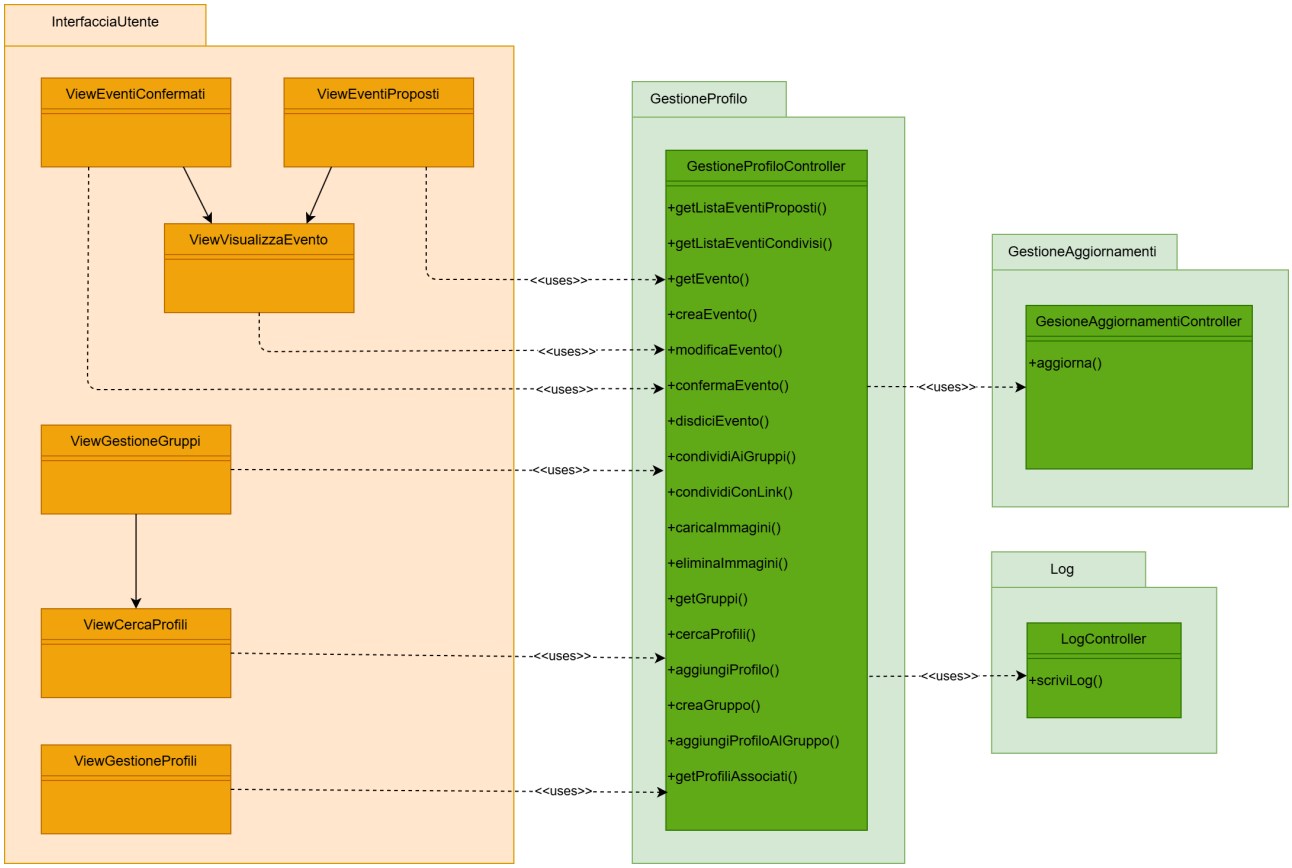

Figura 4: Diagramma delle classi: interfaccia utente, gestione profilo e aggiornamenti

GestioneProfilo è il package principale, a cui le classi di InterfacciaUtente si rivolgono soddisfare le richieste dell'utente. Prevede una sola classe, GestioneProfiloController, che risponde alle azioni a cui un profilo può avere accesso, ovvero tutte quelle previste. Contiene quindi le funzioni relative ai principali casi d'uso, quali a l'ottenimento degli impegni, la creazione o la conferma dell'evento o il caricamento delle immagini.

GestioneAggiornamenti ha il solo compito di aggiornare i dispositivi a seguito di una chiamata da GestioneProfilo, per cui contiene una classe con un metodo. InterfacciaAccesso prevede invece due classi, una per il Login e una per la Registrazione. GestioneAccesso ha una sola classe ma che presenta, seguendo la stessa logica, due metodi distinti. Il package Log, fornendo un solo metodo, contiene usa sola classe.

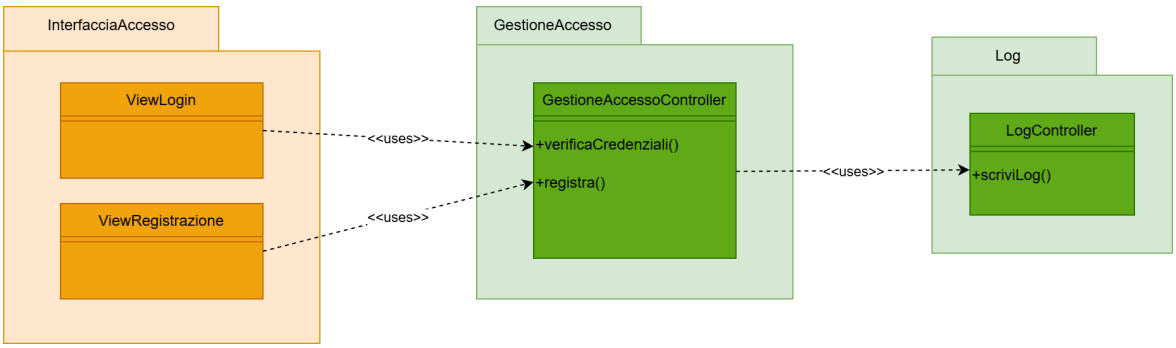

Figura 5: Diagramma delle classi: interfacca e gestione accesso