Trentino Find

Authors
Caccavale Luca - 218724
Callegari Michele - 217838
Luongo Muzio Leonardo - 218274

Deliverable 2 - GRUPPO 14
Corso di Ingegneria del Software

Dipartimento di Ingegneria e Scienza dell'Informazione - DISI
Università di Trento
23 Aprile 2023

1 Component diagram

In questa unità andremo a presentare il Component Diagram di Trentino Find andando a specificare e descrivendo i componenti di cui si compone il sistema.

I componenti sono delle entità autonome del sistema il cui scopo principale è quello di individuare le principali strutture logiche di cui Trentino Find ha bisogno per funzionare e di cui è costituito.

Ciascun componente potrà esporre una o più interfacce (fornite o richieste), ovvero delle porte per specificare servizi o comportamenti di cui il componente dispone o di cui necessita.

1.1 Defizione dei componenti

1.1.1 Gestione database

Questo componente gestisce il servizio interno database permettendo agli altri componenti interni di interagire accedendo o modificando dati sul database.

Tipo	Nome	Descrizione
Fornita	Dati	Il componente fornisce internamente un'interfaccia per accedere ai dati sul database.
Richiesta	Acquisizione dati	Il componente richiede un'interfaccia per accedere al database.

1.1.2 Gestione autenticazione

Questo componente gestisce il servizio interno di autenticazione permettendo agli utenti ovvero il giocatore, il moderatore, l'amministratore di essere riconosciuti dal sistema coma da requisito funzionale RF1.1, RF4.1, RF5.1.

Tipo	Nome	Descrizione
Fornita	Autenticazione	Il componente fornisce internamente un'interfaccia per autenticare l'utente (ovvero il giocatore, il moderatore o l'amministratore) che permette la validazione delle credenziali inserite in fase di autenticazione.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.

1.1.3 Gestione giocatore

Questo componente gestisce il giocatore e le interazioni che ha con il sistema.

Tipo	Nome	Descrizione
Fornita	Registrazione	Il componente fornisce al giocatore un'interfaccia per registrarsi come da RF1.2.
Fornita	Autenticazione	Il componente fornisce al giocatore un'interfaccia per autenticarsi come da RF1.1. Il processo prevede l'inserimento delle credenziali del giocatore e la successiva validazione da parte del sistema.
Fornita	Informazioni personali	Il componente fornisce al giocatore un'interfaccia per la visualizzazione delle proprie informazioni personali inserite in fase di registrazione come da RF1.3.
Fornita	Modifica informazioni personali	Il componente fornisce al giocatore un'interfaccia per la modifica delle proprie informazioni personali inserite in fase di registrazione come da RF1.3.
Fornita	Attività personali	Il componente fornisce al giocatore un'interfaccia per la visualizzazione delle attività personali svolte. In particolare la visualizzazione di una cronologia degli oggetti nascosti trovati dall'utente e la visualizzazione degli oggetti nascosti registrati dall'utente come da RF1.4 e RF1.5.
Fornita	Sfida giocatore	Il componente fornisce al giocatore un'interfaccia per permettergli di sfidare altri giocatori come da RF3.2.
Fornita	ID Giocatore	Il componente fornisce agli altri componenti interni un'interfaccia per l'acquisizione di un ID per identificare in modo univoco un giocatore all'interno del sistema.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.
Richiesta	Autenticazione	Il componente richiede internamente un'interfaccia per validare le credenziali immesse dal giocatore.

1.1.4 Gestione moderatore

Questo componente gestisce il moderatore e le interazioni che ha con il sistema.

Tipo	Nome	Descrizione
Fornita	Autenticazione	Il componente fornisce al moderatore un'interfaccia per autenticarsi come da RF4.1. Il processo prevede l'inserimento delle credenziali del moderatore e la successiva validazione da parte del sistema.
Fornita	Attività giocatori	Il componente fornisce al moderatore un'interfaccia per la visualizzazione delle attività dei giocatori come l'inserimento di commenti o la registrazione di nuovi oggetti nascosti come da RF4.2.
Fornita	Elimina commento	Il componente fornisce al moderatore un'interfaccia per l'eliminazione dei commenti come da RF4.3.
Fornita	Segnala giocatori	Il componente fornisce al moderatore un'interfaccia per la segnalazione dei giocatori come da RF4.5.
Fornita	ID moderatore	Il componente fornisce agli altri componenti interni un'interfaccia per l'acquisizione di un ID per identificare in modo univoco un moderatore all'interno del sistema.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.
Richiesta	Autenticazione	Il componente richiede internamente un'interfaccia per validare le credenziali immesse dal moderatore.

1.1.5 Gestione amministratore

Questo componente gestisce l'amministratore e le interazioni che ha con il sistema.

Tipo	Nome	Descrizione
Fornita	Autenticazione	Il componente fornisce all'amministratore un'interfaccia per autenticarsi come da RF5.1. Il processo prevede l'inserimento delle credenziali dell'amministratore e la successiva validazione da parte del sistema.
Fornita	Moderatori	Il componente fornisce all'amministratore un'interfaccia per la visualizzazione dei moderatori come da RF5.3.
Fornita	Modifica moderatori	Il componente fornisce all'amministratore un'interfaccia per la modifica dei moderatori. In particolare la possibilità da parte dell'amministratore di poter aggiungere nuovi moderatori e modificare quelli già presenti come da RF5.4.
Fornita	Rimuovi giocatori	Il componente fornisce all'amministratore un'interfaccia per la rimozione dei giocatori come da RF5.2.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.
Richiesta	Autenticazione	Il componente richiede internamente un'interfaccia per validare le credenziali immesse dall'amministratore.

1.1.6 Gestione oggetto

Questo componente gestisce gli oggetti nascosti dai giocatori. In particolare gestisce le interazioni tra giocatore e oggetto nascosto come la visualizzazione in fase di gioco dei dettagli che riguardano l'oggetto nascosto come da RF2.2, l'eventuale inserimento di commenti a seguito del ritrovamento dell'oggetto come da RF3.4 ed infine la possibilità da parte del giocatore di registrare l'inserimento di nuovo oggetti come da RF2.5. Gestisce poi le interazioni tra moderatore e oggetto nascosto come l'azione di conferma di un oggetto nascosto come da RF4.4.
L'interfaccia richiesta ID giocatore permette di associare univocamente all'oggetto nascosto il giocatore che lo ha registrato.
L'interfaccia richiesta ID moderatore permette di associare univocamente all'oggetto nascosto il moderatore che ne ha confermato la registrazione.

Tipo	Nome	Descrizione
Fornita	Dettagli oggetto	Il componente fornisce un'interfaccia per permettere al giocatore la visualizzazione dei dettagli di un oggetto nascosto come da RF2.2.
Fornita	Inserisci commento	Il componente fornisce un'interfaccia per permettere al giocatore di inserire un commento a seguito del ritrovamento di un oggetto nascosto come da RF3.4.
Fornita	Registra oggetto	Il componente fornisce un'interfaccia per permettere al giocatore di registrare un oggetto nascosto come da RF2.5.
Fornita	Conferma oggetto	Il componente fornisce un'interfaccia per permettere al moderatore di confermare l'inserimento di un oggetto nascosto come da RF4.4.
Fornita	ID oggetto	Il componente fornisce agli altri componenti interni un'interfaccia per l'acquisizione di un ID per identificare in modo univoco un oggetto del sistema.
Richiesta	ID giocatore	Il componente richiede un'interfaccia per l'acquisizione dell'ID del giocatore.
Richiesta	ID moderatore	Il componente richiede un'interfaccia per l'acquisizione dell'ID del moderatore.
Richiesta	Fotocamera	Il componente richiede un'interfaccia per accedere alla fotocamera.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.
Richiesta	Coordinate geografiche	Il componente richiede un'interfaccia per accedere alle coordinate geografiche necessarie per la registrazione di un nuovo oggetto da parte di un giocatore.

1.1.7 Gestione mappa

Questo componente gestisce la mappa interna al gioco utilizzata dal giocatore.

Tipo	Nome	Descrizione
Fornita	Visualizza mappa	Il componente fornisce un'interfaccia per la visualizzazione della mappa geografica del territorio con indicate le posizioni corrispondenti ai luoghi di interesse come da RF2.1.
Fornita	Visualizza posizione	Il componente fornisce un'interfaccia per la visualizzazione della posizione attuale del giocatore sulla mappa come da RF2.1.
Richiesta	Coordinate geografiche	Il componente richiede un'interfaccia per accedere alle coordinate del giocatore.
Richiesta	Mappe	Il componente richiede un'interfaccia per accedere alle mappe geografiche del territorio.
Richiesta	Dati	Il componente richiede un'interfaccia per accedere al database.

1.1.8 Gestione Servizio Internet Geografico (GPS)

Questo componente gestisce l'interazione con il servizio esterno di Internet Geografico.

Tipo	Nome	Descrizione
Fornita	Coordinate geografiche	Il componente fornisce un'interfaccia per ottenere le coordinate geografiche.
Richiesta	Acquisizione coordinate geografiche	Il componente richiede un'interfaccia per acquisire le coordinate geografiche.

1.1.9 Gestione OpenStreetMap

Questo componente gestisce l'interazione con il servizio esterno OpenStreetMap.

Tipo	Nome	Descrizione
Fornita	Mappe	Il componente fornisce internamente un'interfaccia per accedere e visualizzare le mappe.
Richiesta	Acquisizione mappe	Il componente richiede un'interfaccia per accedere alle mappe fornite dal sistema esterno OpenStreetMap.

1.1.10 Gestione fotocamera

Questo componente gestisce l'interazione con il servizio esterno Fotocamera.

Tipo	Nome	Descrizione
Fornita	Fotocamera	Il componente fornisce internamente un'interfaccia per accedere alla fotocamera.
Richiesta	Accesso fotocamera	Il componente richiede un'interfaccia per accedere alla fotocamera.

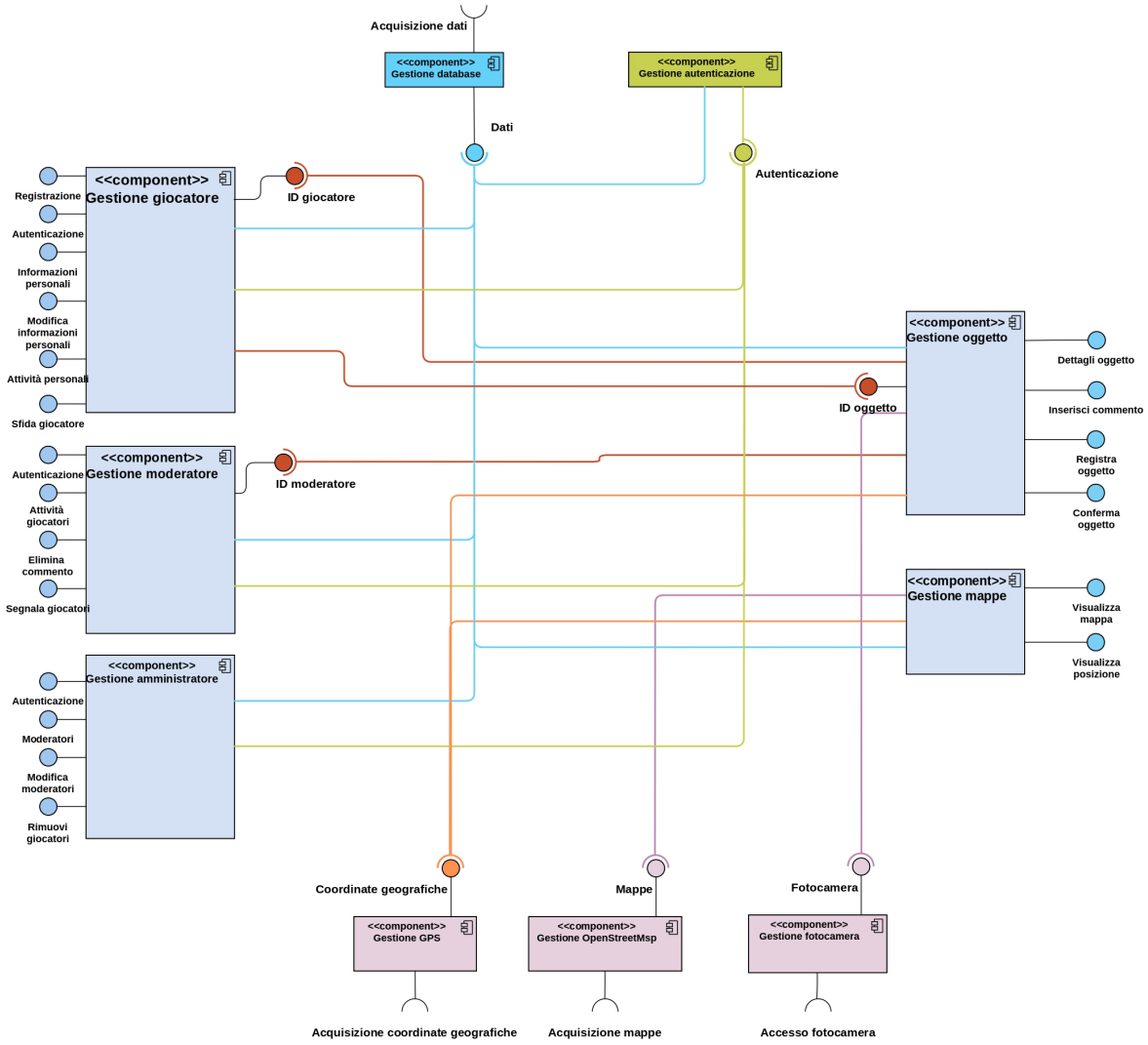

Figura 1: Component Diagram

2 Class Diagram

In questo documento viene trattato il diagramma delle classi e l'object constraint language dell'applicativo Trentino Find.
Il class diagram mostra un insieme di classi e le loro relazioni. E' un tipo di grafo dove i nodi sono le classi e le interfacce, gli archi sono le relazioni; esso può contenere anche package o sottosistemi utilizzati per raggruppare elementi.

Nel nostro sistema sono presenti le seguenti classi: Utente, Giocatore, Moderatore, Amministratore, Sfida, RankingGiocatore, Mappa, AttivitàGiocatore e Oggetto. Di seguito le varie definizioni delle classi contenute nel sistema.

2.1 Utente

La classe Utente è una superclasse che generalizza l'utilizzatore della nostra piattaforma avendo attributi e metodi comuni a tutti gli utenti; possedendo metodi astratti, la classe stessa è astratta.
Da essa derivano le seguenti classi figlie: Giocatore, Moderatore e Amministratore. Gli attributi sono: firstname, lastname, age, passwordHash, email e phone. I metodi sono: Autenticazione, Registrazione, VisuallizzaInformazioni e ModificaInformazioni.

Il metodo **Autenticazione** viene usato per registrare l'avvenuto ingresso alla piattaforma, questo metodo è astratto pertanto verrà implementato in modo specifico nelle classi figlie.

Il metodo **Registrazione** permette ai nuovi utenti di iscriversi alla piattaforma, durante la fase di registrazione, a seconda della tipologia di utente che desidera registrarsi (giocatori, moderatori e amministratori), verrà chiesto di inserire le proprie informazioni personali (nome, cognome, età, email e telefono) e la creazione di una password che deve rispettare determinati criteri di sicurezza.

Il metodo **VisualizzaInformazioni** mostra le informazioni personali dell'utente, fornite in fase di registrazione, come nome, cognome, ecc.

Il metodo **ModificaInformazioni** permetti di modificare le informazioni personali dell'utente, fornite in fase di registrazione, e la modifica della password.

2.2 Giocatore

La classe Giocatore è una classe che eredita dalla superclasse astratta Utente, che rappresenta l'utente giocatore all'interno del gioco, essa è composta dai seguenti attributi: IdPlayer, Ranking, AttivitàGiocatore; e dai seguenti metodi: VisualizzaMappa, GetPosizione, Sfida e RegistraOggetto.

L'attributo **IdPlayer** è univoco e serve per identificare il giocatore, l'attributo **Ranking** indica il livello di esperienza del giocatore e l'attributo **AttivitàGiocatore** indica la relazione con l'omonima classe AttivitàGiocatore che verrà illustrata in seguito a questo documento.

Il metodo **VisualizzaMappa** mette in relazione la classe giocatore con la classe mappa, che permette la visualizzazione da parte del giocatore di una mappa contenente le informazioni riguardanti gli oggetti.

Il metodo **GetPosizione** restituisce la posizione dell'utente appoggiandosi all'API fornite da un servizio internet geografico esempio GPS o Galileo.

Il metodo **Sfida** interagendo con la classe Sfida, permette al giocatore di creare e/o accettare la partecipazione ad una sfida con altri giocatori.

Il metodo **RegistraOggetto**, permette tramite la chiamata ad un metodo della classe AttivitàGiocatore, la registrazione di un nuovo oggetto trovato e/o un nuovo da inserire.

2.3 Moderatore

La classe Moderatore è una classe che eredita dalla classe Utente, rappresenta l'utente moderatore all'interno del gioco, essa è composta dall'attributo IdModeratore e dai metodi: VisuallizzaGiocatore, VisualizzaCommenti, EliminaCommento, SegnalaGiocatore e ConfermaObject.

Il metodo **VisualizzaGiocatore** permette al moderatore di visualizzare le informazioni riguardanti un determinato giocatore, come IdPlayer, Ranking e le informazioni contenute in AttivitàGiocatore.

Il metodo **VisualizzaCommenti** permette di visualizzare tutti i commenti riguardanti un determinato Oggetto.

Il metodo **EliminaCommento** viene utilizzato per eliminare i commenti riguardanti un determinato oggetto che non rispettano le policy della piattaforma, questo metodo è legato al metodo VisualizzaCommenti.

Il metodo **SegnalaGiocatore** serve per segnalare ad un amministratore un giocatore che ha infranto le norme della piattaforma.

Il metodo **ConfermaObject** viene utilizzato per confermare e validare un nuovo oggetto inserito nella lista degli oggetti nascosti da parte di un giocatore.

2.4 Amministratore

La classe Amministratore è una classe che eredita della superclasse Utente, rappresenta l'utente amministratore all'interno della piattaforma, essa è composta dall'attributo IdAmministratore e dai metodi: VisualizzaModeratore, ModificaModeratore e RimuoviGiocatore.

Il metodo **VisualizzaModeratore** permette all'amministratore di visualizzare le informazioni di un moderatore come IdModeratore e le sue attività tenute traccia nel database.

Il metodo **ModificaModeratore** permette la modifica, registrazione e/o eliminazione di un moderatore.

Il metodo **RimuoviGiocatore** viene usato per decidere e in caso espellere un giocatore che è stato segnalato da un moderatore.

2.5 Mappa

La classe Mappa è una classe che rappresenta la mappa del territorio in cui si trova il giocatore, con annessi punti di ricerca degli oggetti nascosti. Tale classe, ha un legame di dipendenza con la classe Oggetto, in quanto il metodo VisualizzaObject() ha come parametro un oggetto di tipo Oggetto. Tale classe è composta dai seguenti attributi: ImmagineMappa, StileMappa. E dai seguenti metodi : VisualizzaObjects, CalcolaDistanzaOG e GetMappa.

L'attributo **ImmagineMappa** è un attributo di tipo immagine, estrapolato dalla nostra applicazione per mezzo delle API, fornite dal servizio internet di geolocalizzazione.

Il metodo **GetMappa** restituisce la mappa del luogo circostanziale all'utente con annessa posizione GPS dell'utente stesso. Oltre a ciò, visualizza i punti di interesse per la ricerca degli oggetti nascosti.

Il metodo **CalcolaDistanzaOG**, interagendo con la classe Giocatore e AttivitàGiocatore, permette di calcolare la distanza che intercorre fra gli oggetti da ricercare e la posizione GPS del giocatore stesso.

2.6 AttivitàGiocatore

La classe AttivitàGiocatore è una classe aggregata alla classe giocatore, che rappresenta i progressi fatti nell'attività videoludica. Tale classe è composta dai seguenti attributi: FoundObjects, HiddenObjects, UnfoundObjects, Comments e IdPlayer. E dai seguenti metodi: VisualizzaAttivitàGiocatore, InserisciCommento, GetObjects, RegistraObject e OggettoTrovato.

Gli attributi FoundObjects, HiddenObjects, UnfoundObjects, e Comments, hanno un legame di dipendenza con la classe Oggetto in quanto raggruppano gli oggetti in array in base all'attività del giocatore.

Il metodo **VisualizzaAttivitàGiocatore** restituisce l'insieme degli oggetti trovati, nascosti e da trovare; e l'insieme dei commenti fatti da parte del giocatore, a questo metodo, oltre al giocatore può aver accesso anche il moderatore.

Il metodo **InserisciCommento** permette al giocatore di inserire un commento riguardante un oggetto, questo metodo si appoggia al metodo AggiungiCommento contenuto nella classe Oggetto.

Il metodo **GetObjects** ritorna l'insieme di tutti gli oggetti trovati, nascosti e da trovare.

Il metodo **RegistraObject** permette al giocatore di registrare un nuovo oggetto da lui nascosto.

Il metodo **OggettoTrovato** permette al giocatore di registrare il ritrovamento di un oggetto nascosto.

2.7 Oggetto

La classe Oggetto è una classe che rappresenta un oggetto presente nel gioco; esso è rappresentato con i seguenti attributi: Location, IdObject, Description, Validated e Comments e con i seguenti metodi: Registrazione, Validazione, Visualizza, AggiungiCommento, GetCommenti e RemoveCommento.

L'attributo **Location** contiene le coordinate geografiche dove l'oggetto è situato, IdObject indica l'identificativo univoco dell'oggetto.

Il metodo **Registrazione** permette la registrazione dell'avvenuto ritrovamento dell'oggetto da parte del giocatore.

Il metodo **Validazione**, è un metodo protected a cui solo il moderatore può accedervi, per confermare la corretta registrazione dell'oggetto in questione.

Il metodo **Visualizza**, permette la visualizzazione della descrizione, del commento e del nome del giocatore, che ha effettuato il ritrovamento dell'oggetto in questione, oltre alla posizione dove è avvenuto il ritrovamento .

Il metodo **AggiungiCommento**, permette l'inserimento di un eventuale commento da parte del Giocatore al momento della registrazione dell'avvenuto ritrovamento del oggetto.

Il metodo **GetCommenti**, restituisce tutti i commenti riguardanti il ritrovamento di un determinato oggetto.

Il metodo **RemoveCommenti** permette al moderatore di rimuovere i commenti, che non rispettano le policy del gioco.

2.8 RankingGiocatore

La classe RankingGiocatore è una classe che rappresenta la classifica del livello dei giocatori all'interno del gioco. Tale classe è composta dai seguenti attributi: Rank e UnlochedObject. E dai seguenti metodi: VisualizzaClassifica, NumeroOggetti e CalcoloPunti.

L'attributo **rank** rappresenta la classifica dei giocatori ordinati in base al livello raggiunto.

Il metodo **VisualizzaClassifica** permette la visualizzazione dell' id e del livello giocatore di tutti i players, in ordine decrescente, di livello.

Il metodo **NumeroOggetti** restituisce il numero di oggetti di cui il giocatore ha diritto a nascondere.

Il metodo **CalcoloPunti** viene usato dal sistema per calcolare ed aggiornare il punteggio dei giocatori, questo metodo viene chiamato ogni volta che un giocatore trova un oggetto nascosto.

2.9 Sfida

La classe Sfida è una classe che rappresenta le sfide tra due giocatori, permette ad un player di sfidarne un altro a trovare un determinato oggetto nascosto, essa è composta da i seguenti attributi: IdSfidante, IdSfidato, IdOggetto e Timer e dai seguenti metodi: Start, End e CheckObject.

Il metodo **Start** viene usato per avviare la sfida, esso si avvia quando lo sfidato accetta la sfida e fa partire il timer.

Il metodo **End** fa terminare la sfida e viene chiamato per tre diversi casi: lo sfidato si arrende, il timer è scaduto o lo sfidato ha trovato l'oggetto, con i primi due casi la sfida la vince lo sfidante, con l'ultimo caso vince lo sfidato.

Il metodo **CheckObject** viene usato dallo sfidato per validare il ritrovamento dell'oggetto nascosto dallo sfidante, se esce un esito positivo la sfida termina chiamando il metodo end.

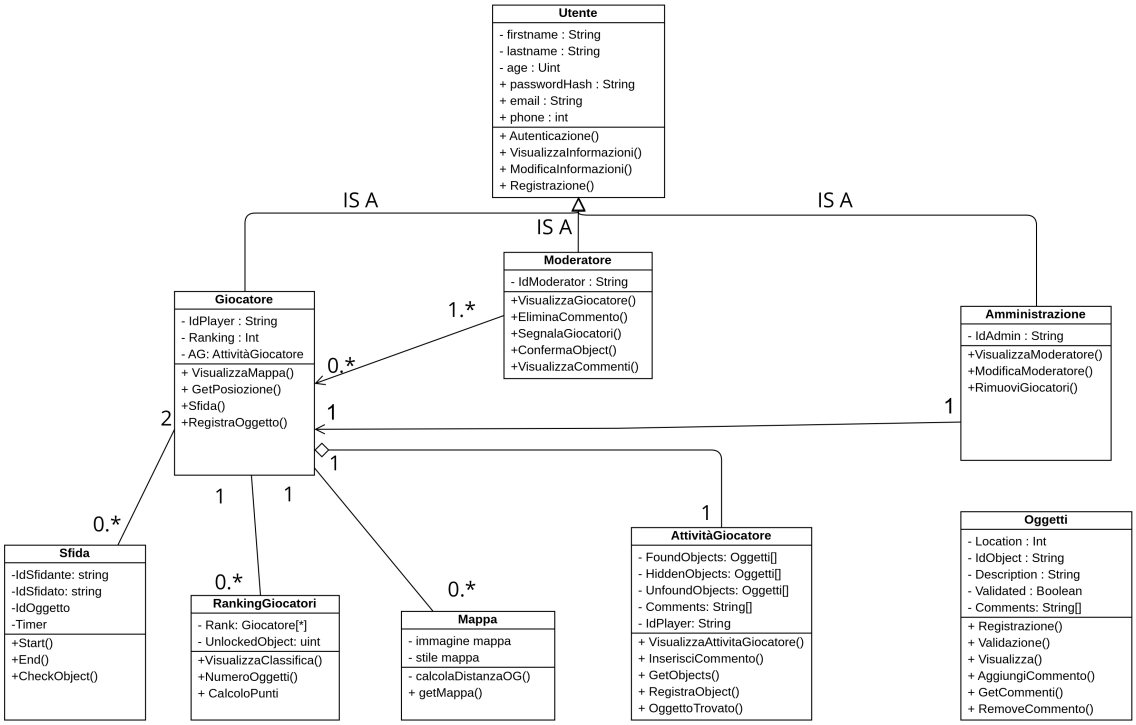

Figura 2: Class Diagram

3 Object Constraint Language

In questa parte del documento viene trattato la parte di OCL del progetto di TrentinoFind L'Object Constraint Language (OCL) è un linguaggio di specifica formale che serve per estendere il class diagram.
Di seguito i vincoli OCL usati nel progetto.

3.1 Vincoli OCL classe utente

Nei metodi di **Registrazione** e **ModificaInformazioni**: prima di salvare un utente, è necessario che i vari step della registrazione siano completati con successo.

```
context Registrazione, ModificaInformazioni :: salvaUtente(datiUtente: Utente)
pre: isStrongPassword(password: string)=true
    AND verifiedEmail(email: string)=true
    AND verifiedPhone(phone: int)=true
post: esito= true
```

Nel metodo **Autenticazione** per accedere la password e email devono essere corrette.

```
context Autenticazione :: Accesso(credenziali: email e password)
pre: isEmailCorrect(email: string)= true
    AND isPasswordCorrect(password: string)=true;
post: esito:true
```

3.2 Vincoli OCL per la classe Giocatore

Nel metodo **Sfida**: lo sfidante deve avere lo stesso livello dello sfidato.

```
context: sfida:: creaSfida(IdGiocatori: idSfidato)
pre: RankingFidante == RankingSfidato
post: creaSfida=true
```

3.3 Vincoli OCL per la classe Moderatore

Nel metodo **ConfermaObject**: l'oggetto deve essere ancora validato

```
context: ConfermaObject:: Validazion(risultato: boolean)
pre: Validated==false
post: validaOggetto(risultato)
```

3.4 Vincoli OCL per la classe Amministratore

Nel metodo **RimuoviGiocatore**: il giocatore deve essere già stato segnalato da un moderatore.

```
context: SegnalazioneGiocatore::RimuoviGiocatore(IdPlayer: int)
pre: giocatoreSegnalato==true
post: eliminaGiocatore(risultato)
```

3.5 Vincoli OCL per la classe Oggetto

Nel metodo **Validazione** come pre condizione l'oggetto deve essere registrato.

```
context: Registrazione :: Validazione()
pre: RegistrazioneOggetto()==true
post: esito == true
```

3.6 Vincoli OCL per la classe AttivitàGiocatore

Nel metodo **OggettoTrovato**: il giocatore deve essere vicino all'oggetto e l'oggetto deve essere validato.

```
context: RegistraObject::OggettoTrovato(Oggetto: oggetto, Player: player ,foto: jpg)
pre: Player.GetPosizione()==Oggetto.Location
    AND Oggetto.validated==true
post: OggettoTrovato==true
```

3.7 Vincoli OCL per la classe Sfida

Nel metodo **Start**: lo sfidato deve accettare la sfida

```
context: Player.Sfida()::Sfida.Start(accetta: boolean)
pre: accetta==true
post: timer.start()
```

Nel metodo **End**: la sfida termina in base a tre diversi casi: lo sfidato si arrende, il timer è scaduto o lo sfidato ha trovato l'oggetto.

```
context: End():end()
pre: timer==0
    OR CheckObject()==true
    OR PlayerSfidatoArreso == true
post: esito==true
```