Università degli Studi di Roma, La Sapienza
Dipartimento di informatica

Progetto: L'ingegneria della scacchiera Settembre, 2024

Ingegneria del software: L'ingegneria della scacchiera

by

D'annibale Luigi - Venturini Daniele

Sommario:
Si vuole progettare un sistema per il gioco degli scacchi.
In particolare gli utenti dovranno avere la possibilità di gestire il proprio profilo, gestire le loro preferenze riguardo la grafica, giocare partite contro altri utenti registrati oppure contro il motore di intelligenza artificiale e infine poter revisionare partite giocate.
Il sistema ha una struttura client-server, quindi gli utenti si collegheranno al sistema grazie ad un client dotato di interfaccia grafica, grazie al quale giocare tra loro, tramite la comunicazione con il server.

Indice

1 Guida all'uso

1.1 Prerequisiti

Questo progetto richiede l'installazione di alcune librerie esterne per essere compilato correttamente su Linux. Di seguito si trovano le istruzioni necessarie per l'installazione delle dipendenze e la compilazione del progetto.

1.1.1 wxWidgets

```
sudo apt install libwxgtk3.0-gtk3-dev
```

1.1.2 nlohmann json

Questa libreria è utilizzata per la gestione della comunicazione in JSON.

```
sudo apt install nlohmann-json3-dev
```

1.1.3 PostgreSQL

```
sudo apt install libpq-dev
```

1.1.4 Hiredis

```
sudo apt install libhiredis-dev
```

1.2 Compilazione

Il progetto è suddiviso in due componenti principali: Client e Server. Ciascuno di essi ha una propria directory e richiede una compilazione separata con il proprio file CMake.

1.3 Compilazione Client

Avviare la CMake nella cartella del Client e spostare l'eseguibile `chess` nella cartella `bin`.

1.4 Compilazione Server

Avviare la CMake nella cartella del Server e spostare l'eseguibile `Server` nella cartella `bin` (opzionale).

1.5 Problemi di Compilazione

Se doveste incontrare problemi con l'installazione delle librerie o la compilazione del progetto, siamo disponibili a far vedere il progetto direttamente e spiegare il suo funzionamento.

graphicx

2 Requisiti utente

2.1 Utente non registrato

Un **utente** vuole:

- Registrarsi al sistema Crea Profilo
- Accedere al profilo di un giocatore Login Profilo

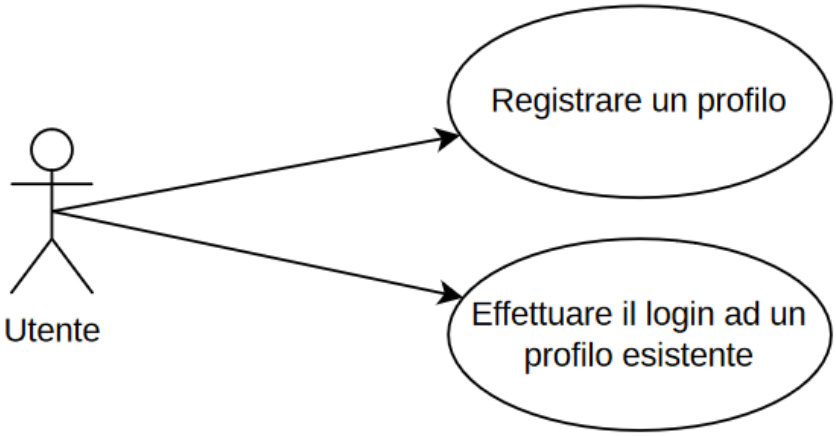


Figura 1: Use Cases per requisiti Utente

2.2 Utente registrato - Giocatore

Per questo sistema un utente registrato e loggato è considerato un **Giocatore**.

Un **giocatore** vuole:

- Visualizzare il profilo di un giocatore Visualizza Profilo
- Modificare il proprio profilo Modifica Profilo
- Giocare una partita contro il computer Partita contro il computer
- Giocare una partita con altri giocatori Partita Multigiocatore
- Visualizzare partite giocate in precedenza Esamina Partita
- Personalizzare le impostazioni del sistema Personalizzazione impostazioni

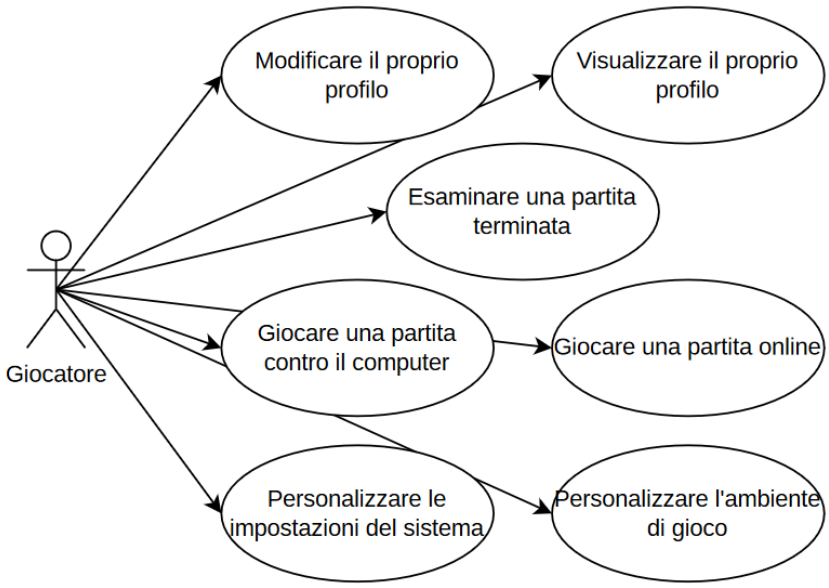


Figura 2: Use Cases per requisiti Giocatore

3 Servizi del sistema

3.1 Gestire il profilo di un giocatore

Ogni profilo può essere *creato* e in qualsiasi momento poi *modificato*. Un utente ha la possibilità di effettuare il *login* ad un profilo esistente.

3.1.1 Creazione/Login a profilo

L'utente inserisce uno username nella sezione apposita, se lo username risulta già esistente allora viene effettuato il login a quel profilo, altrimenti viene creato il profilo e l'utente viene loggato.
La scelta progettuale di fondere creazione e login viene descritta meglio nella sezione Login semplificato

3.1.2 Login semplificato

Nel contesto dello sviluppo di progetti software, l'implementazione di funzionalità quali registrazione, login e recupero password rappresentano una sfida in termini di sicurezza, sono complessi da gestire e soggetti a errori.

Per tale motivo, le moderne pratiche suggeriscono di delegare tali compiti a servizi di autenticazione esterni, chiamati "provider di identità", che possono essere gestiti internamente o esternamente

Di conseguenza, si è scelto di implementare un sistema di login semplificato. Il sistema accetta un nome utente come identificativo unico (ad esempio "Maria"), senza richiedere l'inserimento di una password. Se il nome utente esiste già nel sistema, l'utente viene autenticato; se il nome utente è nuovo, l'utente viene automaticamente registrato e autenticato.

3.1.3 Visualizzazione e modifica del proprio profilo

Un giocatore può vedere le informazioni riguardanti il suo profilo nell' area apposita, ed ha la possibilità di aggiornare alcuni dati, quali il nome utente, il nome ed il cognome; mentre altri sono gestiti interamente dal sistema quindi i punti elo del giocatore.

3.1.4 Esaminare lo storico delle proprie partite

Un giocatore può esaminare le partite da lui giocate, nella sezione dedicata basta scorrere la lista di partite giocate, e selezionando quella di interesse si verrà inseriti nella partita per effettuarne l'analisi

3.2 Giocare una partita tra due giocatori

Una partita tra due giocatori avviene, previo servizio di *ricerca casuale* offerto dal server, tramite comunicazione su canali redis tra i soli giocatori.

La **ricerca casuale** avviene in questo modo:

1. Si entra nell'apposita sezione "Play game"
2. Ogni giocatore sceglie una cadenza di gioco e avvia la ricerca
3. Il sistema accoppia 2 giocatori che scelgono la stessa cadenza di gioco, e li inserisce in una partita, determinando casualmente chi dei due debba giocare con i pezzi bianchi (e chi, di conseguenza, con i neri).

Una volta accoppiati i due giocatori, questi ultimi ricevono dal server le informazioni necessarie e stabiliscono una connessione redis tra loro, per giocare senza la necessità di risorse server; quest'ultimo verrà aggiornato solo alla fine della partita, da entrambi i giocatori, del risultato.

3.3 Giocare una partita contro il computer

Una partita contro il computer viene richiesta da un utente.

1. Si entra nell'apposita sezione "Play against computer"
2. L'utente sceglie la forza di gioco del computer, la cadenza di gioco, la scelta del colore dei pezzi (di deafult impostata casuale)
3. Il sistema inserisce l'utente in una partita contro il computer della forza scelta da lui, con la cadenza di gioco selezionata, inoltre se l'utente seleziona il colore dei sui pezzi al computer viene assegnato quello non scelto, altrimenti il sistema effettua le due assegnazioni in modo casuale

3.4 Visualizza una partita giocata

Un giocatore esamina una partita che è stata giocata.

1. Si fa click sull'ID della partita che si vuole visualizzare nella "lista"
2. Il giocatore sceglie la partita
3. Il sistema inserisce il giocatore nella partita in modalità spettatore

3.5 Personalizzare le impostazioni del sistema

Gli utenti hanno la facoltà di personalizzare i dettagli dell'interfaccia grafica nella sezione apposita.

4 Interfaccia grafica

4.1 Login

Questa è la schermata che viene visualizzata all'apertura del client. Permette di effettuare il login/sign up e indirizza alla homepage.

4.2 Homepage

Tramite questa pagina si può entrare nelle altre sezioni fondamentali della GUI del client.

4.3 Play game

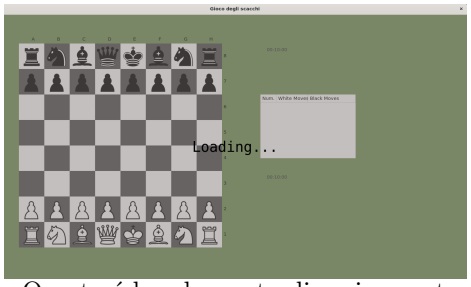
4.3.1 Play online

Questa è la sezione che viene visualizzata dopo aver premuto il bottone "Play online".

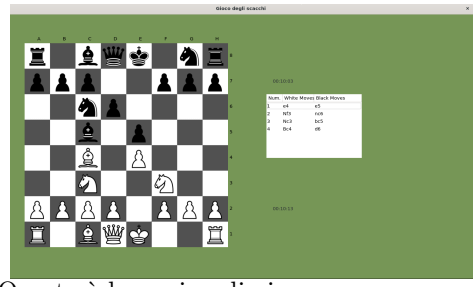
4.3.2 Play against computer

Questa è la sezione che viene visualizzata dopo aver premuto il bottone "Play against computer".

4.4 Play

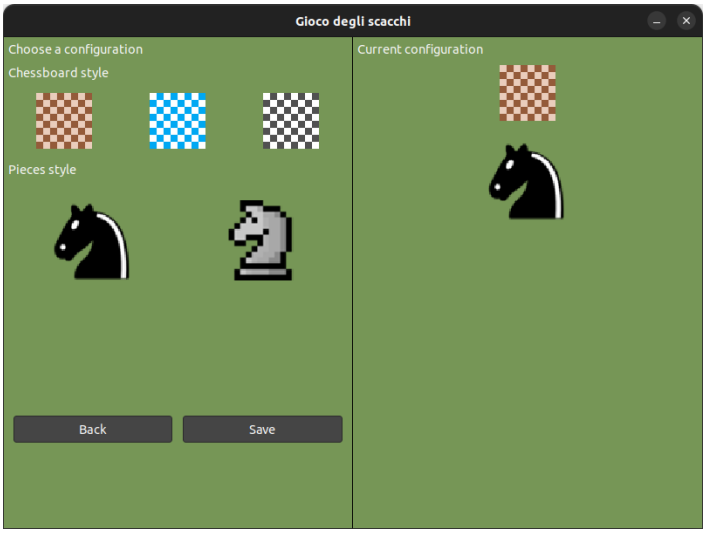


Questa é la schermata di caricamento.



Questa è la pagina di gioco.

4.5 Settings



Questa sezione serve a cambiare alcune impostazioni relative alla GUI.

4.6 View a game

Questa sezione serve a visualizzare lo storico delle partite del giocatore.

Questa è la pagina dell'analisi della partita.

4.7 Profile

Questa sezione serve a gestire il profilo del giocatore, tramite il bottone update profile è possibile cambiare i proprio dati.

5 Librerie esterne utilizzate dal sistema

5.1 Nlohmann_json

Viene utilizzata per la codifica e la decodifica (in generale per la conversione) delle informazioni da oggetto cpp a formato testualee json.

5.2 PostgreSQL (server)

Il DBMS scelto per l'implementazione della vase di dati è proprio PostgreSQL, quindi sulla macchina che fa da serrviente deve essere installata la libreria.

5.3 Hiredis

Redis viene utilizzato nella comunicazione tra client e server (come richiesto da requisiti), rendendo necessario l'utilizzato di una libreria apposita.

5.4 Wx-widgets (client)

Libreria grafica necessaria per il caricamento della **GUI**.

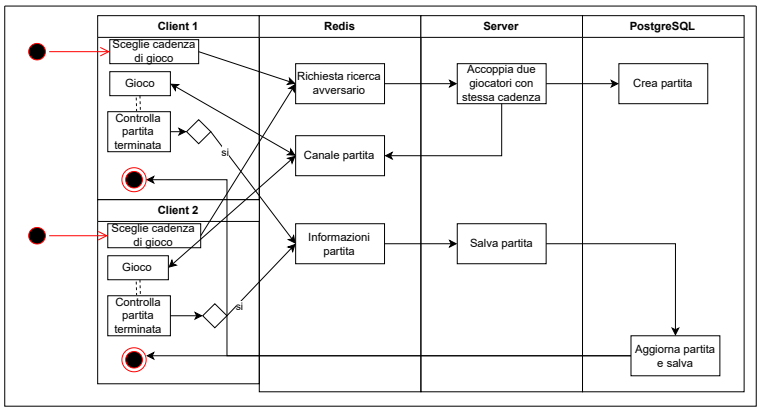
5.5 Chess (integrata nel client)

Libreria che gestisce le partite di scacchi.

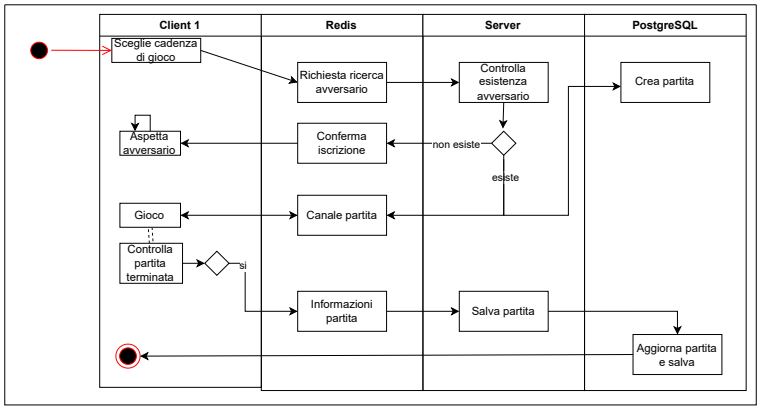
6 Activity diagrams

Di seguito sono riportati i diagrammi dell'attività di ricerca di una partita, nel primo è raffigurata l'interazione di due client paralleli che effettuano la ricerca, mentre nel secondo viene mostrato nel dettaglio il processo che ogni client effettua nella ricerca di una partita.

Interazione a due client

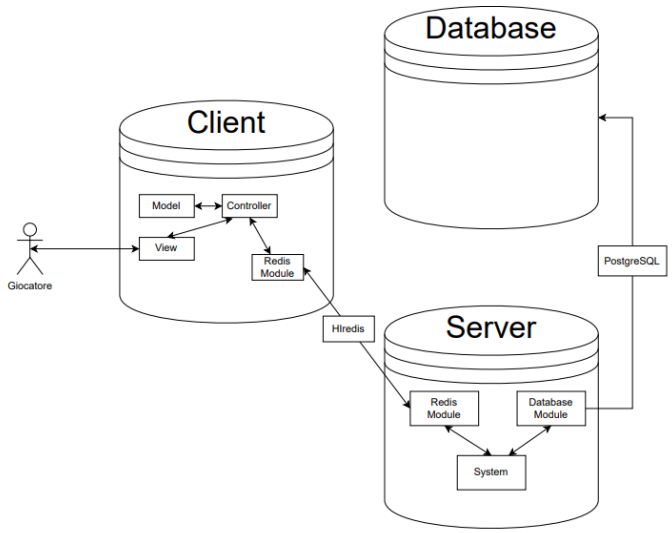


Interazione a singolo client



7 Ambiente del software

L'architettura software è suddivisa in tre componenti principali: Client, Server, e Database.



7.1 Client

È l'interfaccia con cui interagisce il giocatore (utente finale). Il client è composto da tre componenti principali: View, Model, e Controller che seguono il pattern MVC (Model-View-Controller).

- Il modulo Redis è responsabile della comunicazione con il server tramite la libreria Hiredis, la quale facilita l'interazione con il server Redis.
- La view consiste nell'interfaccia utente ed è ben discussa nella sezione dedicata
- Il modello gestisce i dati e la logica dell'applicazione, la maggior parte del lavoro viene svolta dalla libreria chess
- Il controller coordina le interazioni tra la View e il Model e la comunicazione con il Server.

7.2 Server

Si occupa dell'elaborazione delle richieste provenienti dal client. Anch'esso utilizza un modulo Redis per gestire i dati temporanei e un modulo Database per le operazioni sui dati persistenti. La parte di sistema coordina le operazioni del server, gestendo l'interazione tra i moduli Redis e Database.

7.2.1 Test

Sono stati programmati dei test che inviano al server varie richieste, formulate in modo errato, giusto con dati mancanti, scorretti e altre peculiarità per verificare il comportamento e la reattività del server. Il server riesce a superare tranquillamente tutti i test, rispondendo in modo efficace e riconoscendo gli errori, segnalandoli.

7.3 Comunicazione client-server : DGDBP

Per informazioni dettagliate può essere letta la documentazione DGDBP, in breve il dan gigi database protocol serve a definire la comunicazione tra client e server.

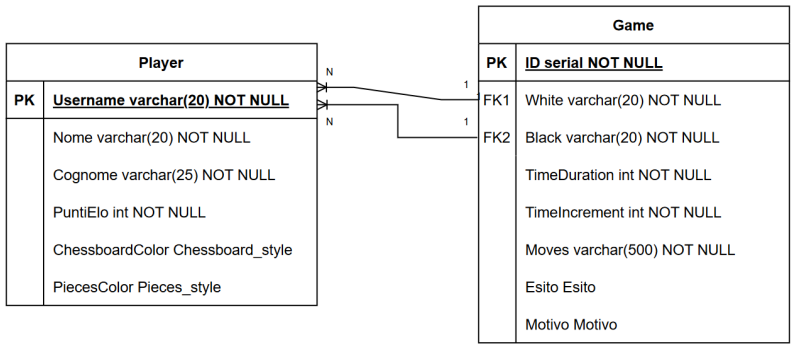
7.4 Database

È il sistema di archiviazione permanente dei dati, gestito tramite PostgreSQL. Il database interagisce direttamente con il server per la memorizzazione e il recupero dei dati.

7.4.1 Schema ER

L'immagine rappresenta il diagramma entità-relazione che illustra la struttura del database, suddiviso in due tabelle principali: Player e Game.
La tabella **Player** contiene informazioni sui giocatori con Username come chiave primaria, e include anche Nome, Cognome, PuntiElo, ChessboardColor e PiecesColor.
La tabella **Game** memorizza i dettagli delle partite, con ID come chiave primaria. Include i riferimenti ai giocatori per White e Black (collegati tramite Username nella tabella Player), oltre a TimeDuration, TimeIncrement, Moves, Esito e Motivo.

Le relazioni tra le tabelle indicano che un giocatore può partecipare a molte partite (relazione 1 a N), e ogni partita coinvolge esattamente due giocatori, uno per ciascun colore (bianco e nero).



Questa architettura risulta in un sistema distribuito dove il client interagisce con il server tramite Redis per operazioni veloci, mentre le informazioni persistenti vengono gestite attraverso il database PostgreSQL.