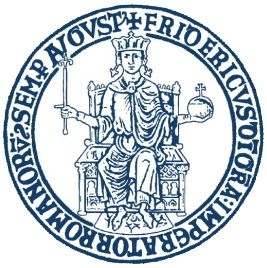


UNIVERSITÀ DEGLI STUDI
DI NAPOLI FEDERICO II

Documentazione java

Sistema informativo dei calciatori
Traccia 3

N86004535 Domenico Mautone - N86004521 Mario Russo

1 febbraio 2024

Indice

1 Traccia progetto

Si sviluppi un sistema informativo, composto da una base di dati relazionale e da un applicativo Java dotato di GUI (Swing o JavaFX), per la gestione di calciatori di tutto il mondo.

Ogni calciatore è caratterizzato da nome, cognome, data di nascita, piede (sinistro, destro o ambidestro), uno o più ruoli di gioco (portiere, difensore, centrocampista, attaccante) e una serie di feature caratteristiche (ad esempio colpo di testa, tackle, rovesciata, etc.).

Il giocatore ha una carriera durante la quale può militare in diverse squadre di calcio. La militanza in una squadra è caratterizzata da uno o più periodi di tempo nei quali il giocatore era in quella squadra. Ogni periodo di tempo ha una data di inizio ed una data di fine. Durante la militanza del giocatore nella squadra si tiene conto del numero di partite giocate, del numero di goal segnati e del numero di goal subiti (applicabile solo ai giocatori di ruolo portiere). Il giocatore può inoltre vincere dei trofei, individuali o di squadra.

Il giocatore può avere anche una data di ritiro a seguito della quale decide di non giocare più. Le squadre di calcio sono specificate dal loro nome e nazionalità.

L'amministratore del sistema si identifica con una login ed una password e ha il diritto di inserire nuovi giocatori nella base di dati, modificarne i dati, aggiungere ulteriori informazioni oppure eliminare un giocatore.

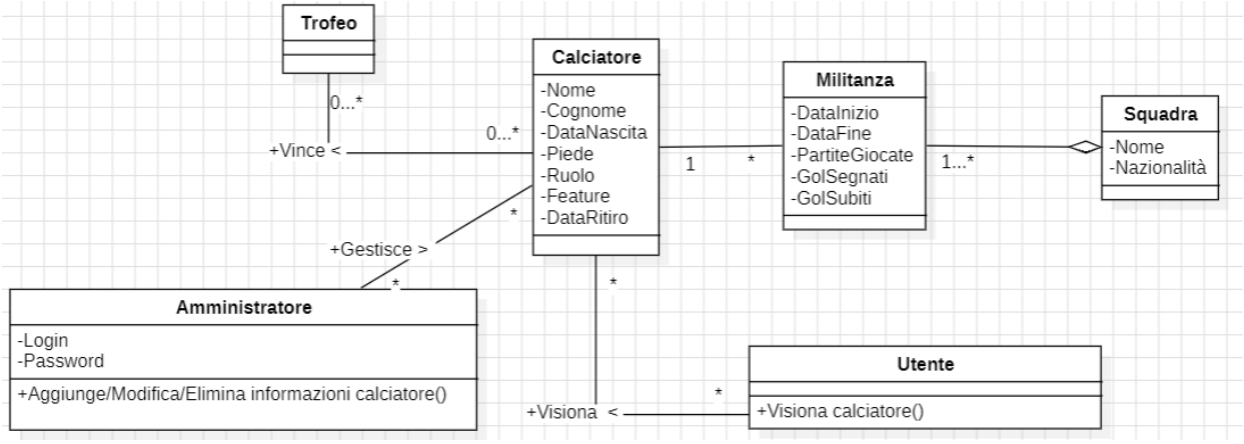
L'utente generico può vedere l'elenco dei giocatori e le loro caratteristiche e può richiedere diverse ricerche, ad esempio filtrando i giocatori per nome, per ruolo, per piede, per numero di goal segnati, per numero di goal subiti, per età, per squadre di appartenenza.

1.1 Analisi traccia

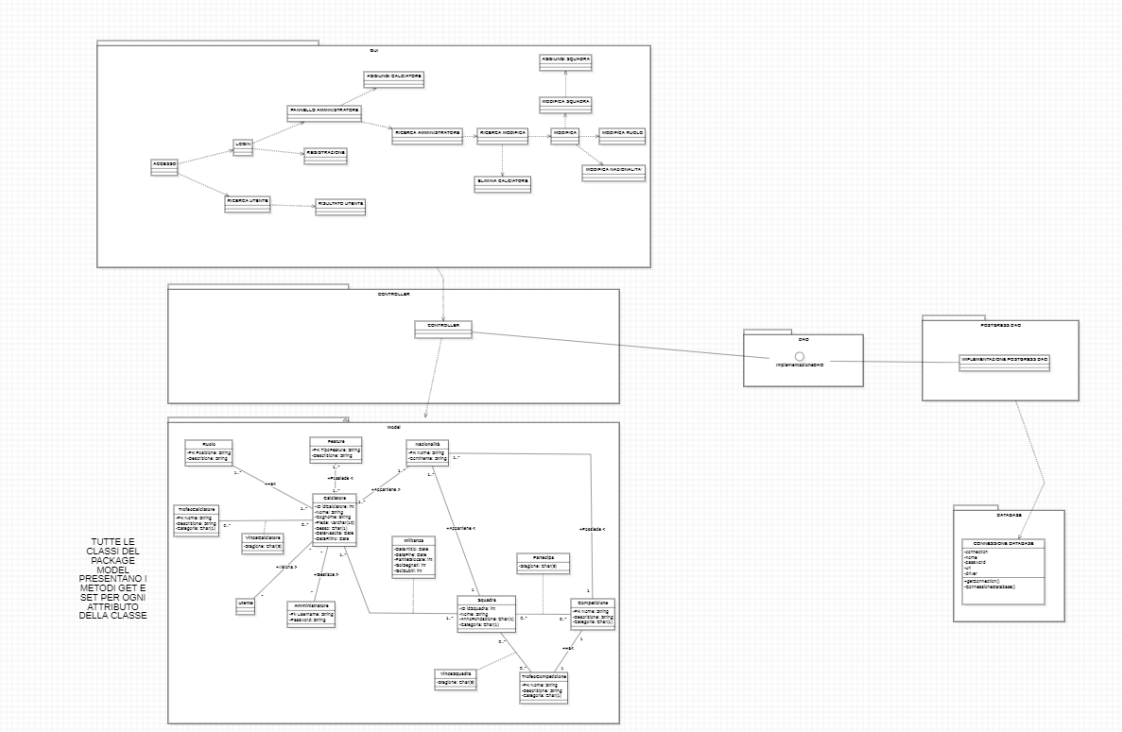
In questo problema abbiamo la classe **Calciatore** che ha come attributi: **Nome**, **Cognome**, **Data di nascita**, **Piede**(che potrà assumere solo tre valori, ovvero **sinistro**, **destro** e **ambidestro**), **Ruolo**(che può essere multiplo), **Feature** (che anch'esso può essere multiplo) e **Data ritiro**. Dopo di che abbiamo la **militanza** che è l'associazione che c'è tra calciatore e squadra e ha come attributi: **Data inizio periodo**, **Data fine periodo**, **Numero di partite giocate**, **gol segnati** e **gol subiti** (attributo considerato solo nel caso dei portieri). Poi avremmo anche una classe **trofei**. Invece la classe delle **squadre di calcio** avrà come attributi **Nome** e **Nazionalità**.

2 Modello concettuale

2.1 Modello UML del dominio del problema



2.2 Modello UML del dominio della soluzione



2.2.1 Analisi UML del dominio della soluzione

Abbiamo creato due classi **trofeo** diferenziandoli per quei trofei che vengono assegnati al **singolo** e per quei trofei che vengono assegnati alle **squadre**, ovvero quelli che si vincono in una **competizione**.

Abbiamo separato da calciatore gli attributi **Nazionalità**, **Feature** e **Ruolo** e le abbiamo rese delle **classi** dato il loro legame con il calciatore che è di tipo **molti-a-molti**.

Abbiamo infine aggiunto la classe **competizione** che rappresenta appunto le competizione di cui una squadra può far parte in una determinata **stagione** e di cui può vincere il **trofeo assegnatogli**.

3 Applicazione in java

Ci troviamo nel nostro applicativo con 6 package:

- Model
- DAO
- PostgresDAO
- Database
- Controller
- GUI

Nel **model** ritroviamo tutte le classi dell'UML con tutti i loro metodi **get**, **set** e **costruttore**, tranne per la classe **Utente** che non l'abbiamo aggiunta dato che non ha attributi e quindi è di poca rilevanza.

Nel **DAO** ritroviamo l'interfaccia **implementazioneDAO** che ha tutti i metodi che vengono poi implementati da **implementazionePostgresDAO** che risiede nel package **PostgresDAO**

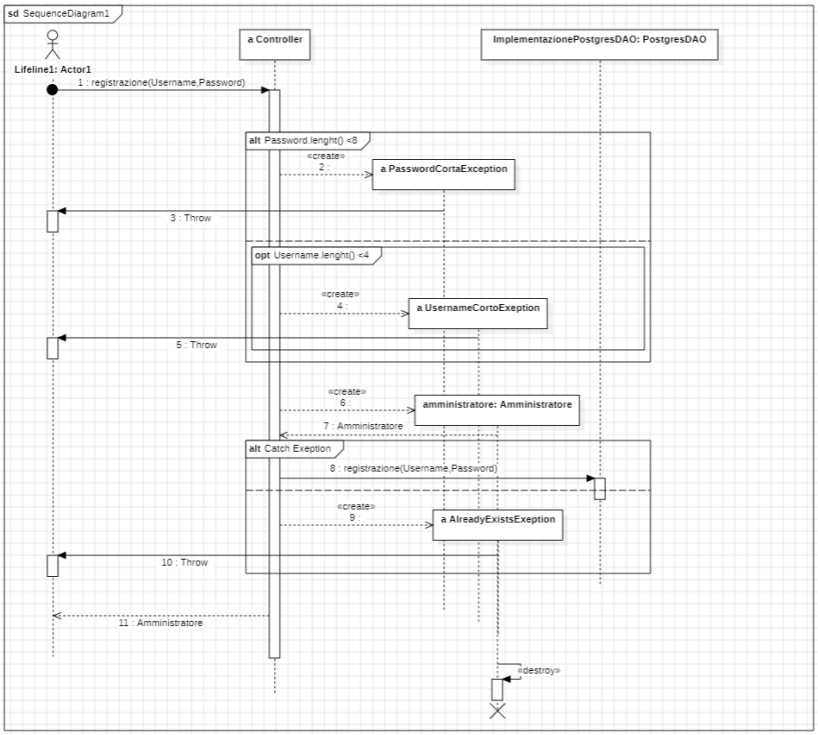
Nel **database** ritroviamo invece il necessario per collegarci col nostro database **PostgresSQL**

Nel **controller** troviamo la classe **controller** che fa da tramite tra il **database** e le **gui** prendendo e passando i dati richiesti

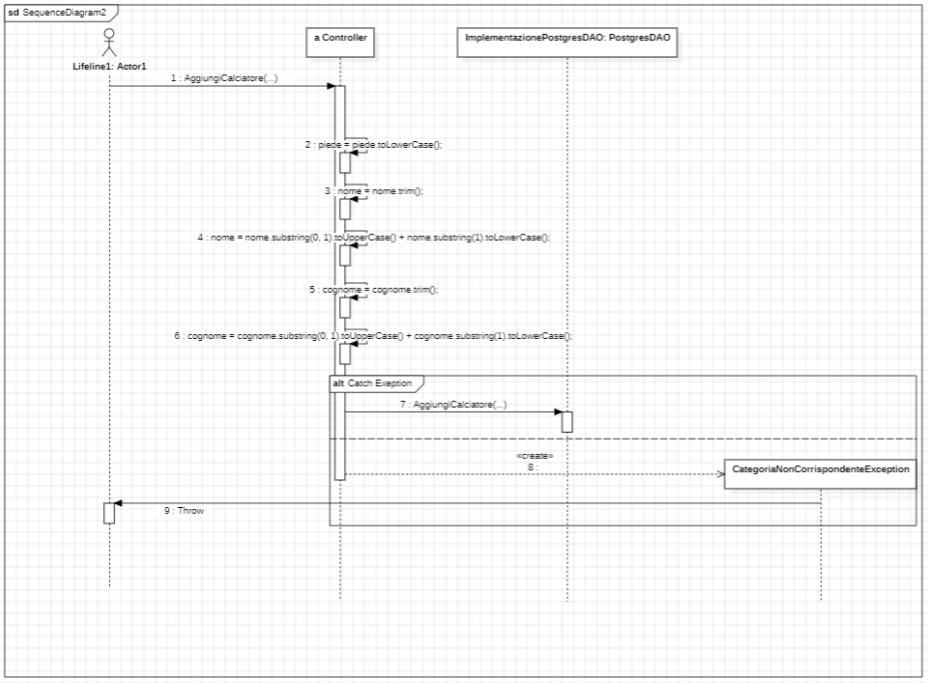
In **gui** abbiamo invece tutte le gui del nostro applicativo

4 Sequence Diagramm

4.1 Funzionalità 1



4.2 Funzionalità 2



Parametri
AggiungiCalciatore(nome,cognome,sesso,squadra,nazionalità,piede,dataNascita,ruolo,dataRitiro,dataInizio,dataFine