

POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Gestionale
Classe L8 – Ingegneria dell'Informazione



Sviluppo di un'applicazione per ottimizzare il completamento di una rosa al Fantacalcio

Relatore *Prof. Fulvio Corno*

Candidato *Jacopo Dylan Badino*

Matr. 248726

A.A. 2019/2020

Indice

1 Titolo del progetto e riferimenti dello studente	5
2 Proposta di progetto	5
2.1. Descrizione del problema proposto	5
2.2. Descrizione della rilevanza gestionale del problema	5
2.3. Descrizione del data-set per la valutazione	5
2.4. Descrizione preliminare degli algoritmi coinvolti	6
2.5. Descrizione preliminare delle funzionalità previste per l'applicazione software	6
3 Descrizione dettagliata del problema affrontato	8
4 Descrizione del data-set	9
4.1. Tabella "Statistiche"	9
4.2. Tabella "Quotazioni"	10
5 Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati	12
5.1. L'interfaccia utente	12
5.2. La logica applicativa	12
5.3. Il database e le strutture dati	13
5.4. Algoritmi utilizzati	13
6 Diagramma delle classi delle parti principali dell'applicazione	15
7 Esempi di utilizzo dell'applicazione	16
7.1. Ricerca giocatori chiave	16
7.2. Calcolo Best MF Team	17
7.3. Calcolo Best MV Team	19
7.4. Link al video YouTube	19
8 Valutazioni sui risultati ottenuti e conclusioni	20

1. Titolo di progetto e riferimenti dello studente

Sviluppo di un'applicazione per ottimizzare il completamento di una rosa al fantacalcio.

Per realizzare l'applicazione mi sono appoggiato al sito Fantacalcio.it.

2. Proposta di progetto

2.1. Descrizione del problema proposto

Dai videogiochi alla vendita di diritti, il calcio è oggi un vero e proprio settore di mercato a sé stante.

Recentemente, in particolare, sembra essere esplosa la moda del "Fantacalcio", un gioco che può essere svolto tra amici o in rete, e nel quale ognuno di noi può diventare allenatore di una squadra di calcio, costruita acquistando giocatori tramite aste o basandosi su prezzi prefissati, a fronte ovviamente di un budget limitato di "fantamilioni", deciso precedentemente.

Dopo aver acquistato i giocatori necessari, il "fantallenatore" dovrà schierare la propria formazione di giornata in giornata, e otterrà un punteggio che varierà a seconda dei voti assegnati dalle redazioni dei quotidiani sportivi e a seconda di "bonus" e "malus" (gol, assist, ammonizioni, espulsioni...) attribuiti ai suoi 11.

Anche in questo ambito non sono tardati ad arrivare brand dedicati come "Fantacalcio.it" che ha iniziato a vendere prodotti a tema, ma anche applicazioni per gestire i risultati e studiare tattiche. La mia applicazione intende seguire questo filone, cercando di fornire consigli sui giocatori migliori e di aiutare gli utenti ad ottenere grandi risultati con la minima spesa.

2.2. Descrizione della rilevanza gestionale del problema

Per quanto si tratti in fondo soltanto di un passatempo, e per quanto nell'effettivo successo finale della squadra abbia un ruolo fondamentale anche la fortuna, questa può essere sicuramente aiutata da un mercato svolto oculatamente. Ed è in questo mercato che entra in gioco il nostro lato manageriale.

Sarà importante gestire bene i nostri "fantamilioni", individuare i giocatori che stanno facendo meglio durante la stagione in corso e il cui prezzo è ancora conveniente, bilanciare i reparti (Portiere, Difesa, Centrocampo e Attacco). In generale sarà nostro obiettivo ottimizzare le spese per la nostra rosa, riuscendo ad ottenere giocatori migliori sulla carta spendendo il minimo possibile.

2.3. Descrizione dei data-set per la valutazione

Per il data-set mi sono appoggiato al sito "Fantacalcio.it", scaricando dapprima i file riguardanti le statistiche sui giocatori (<https://www.fantacalcio.it/statistiche-serie-a>) e poi quelli sulle quotazioni (<https://www.fantacalcio.it/quotazioni-fantacalcio>) tramite le due sezioni messe a disposizione dal sito. Entrambi i file erano inizialmente in formato xlsx e ho quindi provveduto a convertirli in formato sql per poterli poi sfruttare come visto a lezione.

Si noti come i dati in questione si riferiscono alla stagione appena terminata. L'applicazione potrebbe essere maggiormente utile a stagione in corso, utilizzando quindi dati parziali e che possono essere sfruttati per il prosieguo della stagione.

2.4. Descrizione preliminare degli algoritmi coinvolti

La prima parte dell'applicazione sfrutterà semplicemente operazioni di tipo inserimento/ricerca, oltre a qualche basilare operazione svolta sui dati.

La seconda parte dell'applicazione utilizzerà invece un algoritmo di ricorsione per la ricerca della migliore squadra possibile, dopo aver settato i filtri del caso.

2.5. Descrizione preliminare delle funzionalità previste per l'applicazione software

L'applicazione sarà divisa in due parti.

La prima parte presenterà un menu a tendina dal quale sarà possibile scegliere una delle squadre iscritte al campionato in corso. Cliccando su un apposito bottone, verranno visualizzati per la squadra selezionata alcuni giocatori chiave:

- Top Player: Il giocatore con la media fantavoto più alta della rosa;
- Sorpresa: Il giocatore con il rapporto media fantavoto / quotazione iniziale più alto della rosa e con una fantamedia almeno sufficiente;
- Muro: Il difensore con la media voto pulita più alta della rosa;
- Assistan: Il giocatore con più assist all'attivo della rosa;
- Flop: Il giocatore (portieri esclusi) con la media fantavoto più bassa della rosa.

La seconda parte presenterà invece un campo di testo attraverso il quale l'utente potrà specificare i crediti a sua disposizione e dei menu attraverso i quali potrà scegliere quanti giocatori per ruolo sta cercando. Dopo aver settato i filtri della ricerca, cliccando sull'apposito bottone, l'applicazione calcolerà per lui:

- Best MF Team: la migliore rosa possibile in termini di fantavoto, intesa come l'insieme dei giocatori col ruolo richiesto che abbia la somma complessiva delle fantamedie più alta e il cui prezzo cumulato non superi il budget a disposizione;
- Best MV Team: la migliore rosa possibile in termini di voto pulito, intesa come l'insieme dei giocatori col ruolo richiesto che abbia la somma complessiva delle medie voto più alta e il cui prezzo cumulato non superi il budget a disposizione;

Si noti che tutti i giocatori presi in considerazione dal programma dovranno aver preso parte ad almeno il 50% delle partite.

3. Descrizione dettagliata del problema affrontato

Dopo aver pensato a lungo su quale argomento incentrare la mia tesi di laurea, la mia scelta è ricaduta infine su qualcosa che facesse davvero parte della mia quotidianità.

Il fantacalcio è un gioco che negli ultimi tempi ha visto la sua popolarità crescere sempre di più nel nostro paese: un gruppo di "fantallenatori", ognuno con la sua rosa, schiera la formazione prima di ogni giornata di campionato e dal fischio d'inizio della prima partita comincia a tifare, e a sperare che i propri beniamini lo portino mese dopo mese alla vittoria finale.

Ad ogni giornata viene calcolato un punteggio, sulla base dei voti assegnati dalle varie redazioni dei quotidiani sportivi e a seconda di "bonus" (punti extra assegnati per gol, assist, rigori parati, ecc....) e "malus" (punti sottratti al voto originale e dovuti a gol subiti, ammonizioni, espulsioni, ecc....) attribuiti ai suoi 11.

Il momento più importante per ogni "fantallenatore" che si rispetti è quello della scelta dei giocatori ad inizio anno: qua si decidono gran parte delle sorti dell'annata fantacalcistica, in quanto una puntata eccessiva su un giocatore sbagliato o una serie di acquisti mal pensati porteranno inevitabilmente a pessimi risultati. E se per vincere il titolo a fine anno sono necessari grandi intuizioni calcistiche e tanta tanta fortuna, è altrettanto importante saper spendere oculatamente i propri crediti, riempire al meglio gli slot per i giocatori disponibili, sempre con un occhio al portafoglio.

Questo tipo di situazione è equiparabile ad una lunga serie di problemi gestionali: dal problema dello zaino ad un semplice problema di ottimizzazione in cui abbiamo l'obiettivo di massimizzare i possibili voti dei nostri giocatori e dunque il nostro punteggio dovendo però rispettare il vincolo imposto dal budget e dalle quotazioni attuali dei giocatori stessi.

Nella seconda parte dell'applicazione in particolare viene sviluppato un algoritmo di ricorsione che risolve proprio un piccolo problema di programmazione lineare:

- Funzione obiettivo: massimizzare la media (o fantamedia) complessiva dei giocatori scelti.
- Vincoli: esattamente p portieri, d difensori, c centrocampisti e a attaccanti, spendendo al massimo x crediti.

Per non farsi trovare impreparati al momento della scelta dei giocatori, sempre più persone spendono molto tempo su siti dedicati a raccogliere statistiche sui talenti delle varie squadre. L'applicazione da me sviluppata si propone di far risparmiare, per quanto possibile, un po' di tempo a chi ha bisogno di completare la propria rosa con pochi crediti a disposizione, e in generale di dare consigli riguardo ai migliori giocatori da acquistare.

4. Descrizione del data-set


Come detto in precedenza, il data-set è stato scaricato dal sito "Fantacalcio.it", inizialmente in formato xlsx e successivamente convertito in formato sql per poter essere sfruttato all'interno dell'applicazione.

Nel data-set sono presenti due tabelle:

- la tabella delle *statistiche*;
- la tabella delle *quotazioni*.

4.1. Tabella "Statistiche"

La tabella "statistiche" contiene come chiave primaria un codice univoco per il giocatore in questione, seguito da una serie di dati riguardanti le sue prestazioni nella stagione corrente.

#	Nome	Tipo di dati
 1	Id	INT
2	R	CHAR
3	Nome	TINYTEXT
4	Squadra	TINYTEXT
5	Pg	INT
6	Mv	FLOAT
7	Mf	FLOAT
8	Gf	INT
9	Gs	INT
10	Rp	INT
11	Rc	INT
12	R+	INT
13	R-	INT
14	Ass	INT
15	Asf	INT
16	Amm	INT
17	Esp	INT
18	Au	INT

In particolare, le colonne indicano:


- Id: Un identificativo univoco per il giocatore in questione, chiave primaria della tabella;
- R: Il ruolo del giocatore, indicato da un carattere (P per Portiere, D per Difensore, C per Centrocampista, A per Attaccante);
- Nome: Il nome del giocatore;

- Squadra: Il nome della squadra di appartenenza del giocatore;
- Pg: Il numero di partite disputate dal giocatore;
- Mv: La media voto del giocatore;
- Mf: La media fantavoto del giocatore, che tiene conto di bonus e malus (gol, assist, ammonizioni, espulsioni...);
- Gf: I gol segnati dal giocatore;
- Gs: I gol subiti dal giocatore;
- Rp: I rigori parati dal giocatore;
- Rc: I rigori calciati dal giocatore;
- R+: I rigori segnati dal giocatore;
- R-: I rigori sbagliati dal giocatore;
- Ass: Gli assist del giocatore;
- Asf: Gli assist da fermo del giocatore;
- Amm: Le ammonizioni rimediate dal giocatore;
- Esp: Le espulsioni rimediate dal giocatore;
- Au: Gli autogol del giocatore.

4.2. Tabella “Quotazioni”

La tabella “quotazioni” contiene informazioni sui prezzi dei giocatori.

Essa contiene come chiave primaria un codice univoco per il giocatore in questione, seguito da una serie di dati riguardanti la sua quotazione in sede d’asta.

#	Nome	Tipo di dati
 1	Id	INT
2	R	CHAR
3	Nome	TINYTEXT
4	Squadra	TINYTEXT
5	QtA	INT
6	Qtl	INT
7	Diff	INT

In particolare, le colonne indicano:

- Id: Un identificativo univoco per il giocatore in questione, chiave primaria della tabella;
- R: Il ruolo del giocatore, indicato da un carattere (P per Portiere, D per Difensore, C per Centrocampista, A per Attaccante);
- Nome: Il nome del giocatore;

- Squadra: Il nome della squadra di appartenenza del giocatore;
- QtA: La quotazione attuale del giocatore, aggiornata dopo l'ultima giornata disputata;
- QtI: La quotazione iniziale del giocatore, ad inizio stagione;
- Diff: La differenza tra quotazione attuale ed iniziale.

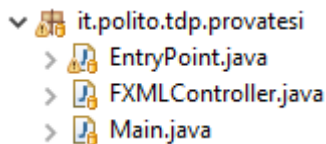
5. Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati

L'applicazione è stata realizzata con il "Pattern MVC" (Model-View-Controller) ed il "Pattern DAO" (Data-Access-Object), due tecniche di programmazione che prevedono la separazione logica delle tre componenti principali del programma:

- L'interfaccia utente;
- La logica applicativa;
- Il database e le strutture dati.

Per ognuna di queste sezioni è stato creato un package.

5.1. L'interfaccia utente



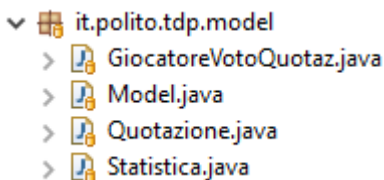
In questo package sono presenti tutte le classi utili all'avvio del programma e alla gestione dell'interfaccia per l'utente.

La classe *EntryPoint* contiene comandi puramente tecnici necessari per la creazione della scena e quindi della finestra che verrà visualizzata all'apertura dell'app. È il punto d'ingresso del programma.

La classe *FXMLController* si occupa invece di gestire l'interazione tra l'utente e le varie sezioni dell'applicazione, le risposte ai comandi in input e le stampe a video dei vari messaggi. Comunica col Model per ricevere le informazioni lavorate a livello logico e mostrarle in output.

La classe *Main* infine ha il solo scopo di consentire l'avvio dell'applicazione.

5.2. La logica applicativa



In questo package sono presenti tutte le classi utili alla gestione della logica applicativa del programma.

La classe *Quotazione* fa riferimento alla tabella "Quotazione" che troviamo nel database, presentando un attributo per ognuna delle colonne presenti. Una volta estratti i dati dal file sql le informazioni sono raccolte in una collezione di oggetti *Quotazione*, per poi essere sfruttati all'interno del *Model*.

La classe *Statistica* fa riferimento alla tabella "Statistica" che troviamo nel database, presentando un attributo per ognuna delle colonne presenti. Una volta estratti i dati dal file sql le informazioni sono raccolte in una collezione di oggetti *Statistica*, per poi essere sfruttati all'interno del *Model*.

La classe *GiocatoreVotoQuotaz* incrocia invece le due tabelle appena citate, sfruttando la chiave univoca comune relativa all'identificativo del giocatore. Questa classe è stata creata per velocizzare alcune operazioni e stampe a video presenti nel programma.

La classe *Model* infine contiene tutta la logica dell'applicazione, dai metodi per calcolare i 5 giocatori chiave di ogni squadra fino agli algoritmi ricorsivi per calcolare il "Best MF Team" e il "Best MV Team".

5.3. Il database e le strutture dati



In questo package sono presenti tutte le classi utili all'interazione del programma con il database "*fanta.sql*".

La classe *DBConnect* contiene i metodi per la configurazione e lo stabilimento della connessione tra l'IDE e il database. È necessario modificare la password una volta scaricato il progetto, aggiornandola con la propria, per far sì che il collegamento avvenga correttamente.

La classe *TesiDAO* contiene invece i metodi con cui dal database vengono prelevate le informazioni e restituite all'IDE all'interno di collezioni di oggetti Quotazione e Statistica.

La classe *TestTesiDAO* ha il solo scopo di consentire alcuni test sul corretto funzionamento dei metodi di *TesiDAO*.

5.4. Algoritmi utilizzati

Gli algoritmi utilizzati sono perlopiù semplici per la prima parte dell'applicazione, salvo poi complicarsi leggermente nella seconda parte.

In *TesiDAO* sono presenti due semplici metodi che scorrono riga per riga le due tabelle presenti nel database e restituiscono una *List* dei due oggetti in questione, per i quali è stata creata un'apposita classe: Quotazione e Statistica.

In *Model* è invece presente un maggior numero di metodi.

Viene innanzitutto effettuata una inizializzazione, con la creazione di un oggetto *TesiDAO* per prepararsi al prelievo di dati e vengono create due *HashMap* che associano gli identificativi unici dei calciatori alle rispettive voci in tabella, al fine di poter reperire più velocemente queste informazioni all'interno del programma.

Il metodo *calcolaTopPlayer* analizza giocatore per giocatore tutte le informazioni relative alle sue statistiche, assicurandosi che faccia parte della squadra indicata in input e che abbia preso parte ad almeno la metà delle partite fin qui disputate in campionato. Ogni qualvolta trovi un giocatore con le caratteristiche indicate e con una fantamedia migliore dei precedenti, aggiorna la soluzione. A fine ciclo, restituisce il *Top Player* calcolato.

Il metodo *calcolaAssistMan* analizza giocatore per giocatore tutte le informazioni relative alle sue statistiche, assicurandosi che faccia parte della squadra indicata in input e che abbia preso parte ad almeno la metà delle partite fin qui disputate in campionato. Ogni qualvolta trovi un giocatore con le caratteristiche indicate e con

la somma degli assist da fermo e di quelli in movimento maggiore dei precedenti, aggiorna la soluzione. A fine ciclo, restituisce l'*assistman* calcolato.

Il metodo *calcolaMuro* analizza giocatore per giocatore tutte le informazioni relative alle sue statistiche, assicurandosi che faccia parte della squadra indicata in input, che abbia preso parte ad almeno la metà delle partite fin qui disputate in campionato e che sia un difensore. Ogni qualvolta trovi un giocatore con le caratteristiche indicate e con la media voto migliore dei precedenti, aggiorna la soluzione. A fine ciclo, restituisce il *muro* calcolato.

Il metodo *calcolaFlop* analizza giocatore per giocatore tutte le informazioni relative alle sue statistiche, assicurandosi che faccia parte della squadra indicata in input, che abbia preso parte ad almeno la metà delle partite fin qui disputate in campionato e che non sia un portiere. Ogni qualvolta trovi un giocatore con le caratteristiche indicate e con la media fantavoto peggiore dei precedenti, aggiorna la soluzione. A fine ciclo, restituisce il *flop* calcolato.

Il metodo *calcolaSorpresa* analizza giocatore per giocatore tutte le informazioni relative alle sue statistiche e alle sue quotazioni, assicurandosi che faccia parte della squadra indicata in input, che abbia preso parte ad almeno la metà delle partite fin qui disputate in campionato e che abbia una fantamedia almeno sufficiente. Ogni qualvolta trovi un giocatore con le caratteristiche indicate e con il rapporto *media fantavoto / quotazione iniziale* maggiore dei precedenti, aggiorna la soluzione. A fine ciclo, restituisce la *sorpresa* calcolata.

Sono poi presenti due metodi più complessi, *CalcolaBestMF* e *CalcolaBestMV*, che si appoggiano su altri due metodi, *ricorsione* e *ricorsione2*, per calcolare la soluzione ottimale in termini di fantamedia e media pulita cumulate dopo aver ricevuto in input il numero di giocatori per ruolo richiesti e i crediti a disposizione.

Un algoritmo è detto ricorsivo quando contiene al suo interno una *funzione ricorsiva*, ovvero un metodo che richiama sé stesso. In questo modo, un grosso problema viene scomposto in sotto problemi più piccoli che verranno risolti sempre col metodo stesso, scomponendoli se necessario nuovamente, in modo iterativo. Per evitare che si creino cicli infiniti è opportuno stabilire una *condizione di terminazione*, ovvero una condizione che, se vera, permetta di uscire dal ciclo.

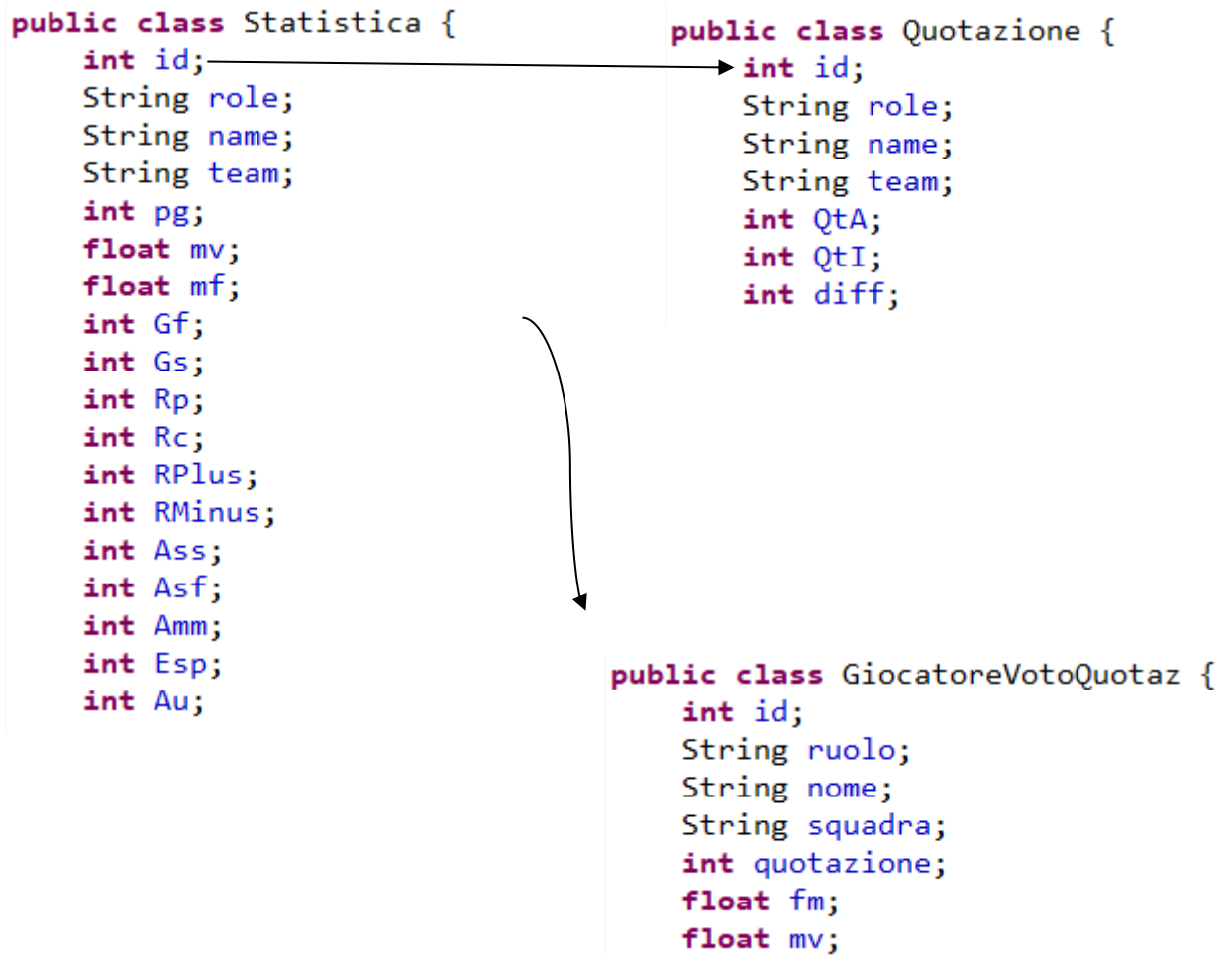
Grazie alla tecnica ricorsiva è possibile semplificare fortemente molti tipi di problemi comuni, ma spesso le soluzioni ricorsive non sono le più efficienti, in quanto l'invocazione iterativa di una funzione ha un costo rilevante.

Nel caso di questa applicazione, il programma crea una *List* di *GiocatoreVotoQuotaz* vuota e visibile a livello globale, che verrà aggiornata di volta in volta quando il calcolatore troverà una nuova soluzione migliore. Dopo aver salvato in locale i dati in input riguardo a crediti e ruoli richiesti, viene invocato un secondo metodo, quello ricorsivo, passandogli una seconda *List* inizialmente vuota, e i crediti rimasti, inizializzati al budget iniziale. A questo punto la funzione ricorsiva analizza giocatore per giocatore, e nel caso si tratti di un ruolo richiesto e il budget lo permetta, prova ad aggiungere l'oggetto alla soluzione parziale (la seconda *List*).

La *condizione di terminazione* dell'algoritmo è rappresentata dal fatto che la soluzione parziale abbia soddisfatto le richieste per ruolo fatte dall'utente. A questo punto, appoggiandosi a due metodi esterni *CalcolaTotalMedia* e *CalcolaTotalFantaMedia*, il programma calcola il punteggio cumulativo del team, e nel caso questo sia più alto di quello della soluzione finale o ancora la soluzione finale sia ancora vuota, la aggiorna.

6. Diagramma delle classi delle parti principali dell'applicazione

Le classi principali dell'applicazione sono soltanto tre: *Statistica*, *Quotazione* e *GiocatoreVotoQuotaz*.



Statistica è legata a *Quotazione* dall'identificativo univoco del giocatore.

L'attributo *Id* permette dunque, dato un codice, di ritrovare nell'applicazione sia le informazioni sulle statistiche del calciatore in questione, sia quelle riguardo alla sua quotazione.

Per semplificare alcuni metodi, è stata creata una classe *GiocatoreVotoQuotaz* che integra le informazioni principali di entrambe le tabelle, mantenendo come chiave primaria l'*Id* del giocatore.

Le tre classi sono sfruttate all'interno del *Model* e del *FXMLController* per qualsiasi operazione sui dati.

7 Esempi di utilizzo dell'applicazione

Di seguito è riportato un esempio di utilizzo dell'applicazione.

7.1. Ricerca giocatori chiave



**FantaApp
Serie A**



Seleziona una squadra: **Calcola**

Top Player:

Sorpresa:

Assistman:

Muro:

Flop:

L'applicazione presenta una prima sezione dove è possibile selezionare una squadra da un menu a tendina, che mostrerà tutte le squadre del campionato attuale in ordine alfabetico.

In caso provassimo a calcolare i giocatori chiave senza aver ancora selezionato una squadra, il sistema ci avviserà del problema.



Top Player:

Sorpresa:

Assistman:

Muro:

Flop:

Vediamo invece un esempio di output selezionando la squadra Parma:



The screenshot shows the 'FantaApp Serie A' interface. At the top, there is a logo for 'SERIE A TIM'. Below it, a dropdown menu is set to 'Parma' with a 'Calcola' button next to it. The results are displayed in a list of player statistics:

Category	Player	Value
Top Player:	CORNELIUS	MF: 7,83
Sorpresa:	KULUSEVSKI	QI: 4 credit
Assistman:	KULUSEVSKI	8 assist
Muro:	GAGLIOLO	MV: 6,00
Flop:	PEZZELLA GIU.	MF: 5,66

Il programma ci restituisce i giocatori da lui calcolati, dandoci alcuni consigli su come muoverci in sede d'acquisto.

Andreas Cornelius, ci mostra la grafica, è il giocatore con la fantamedia più alta in rosa. Con i suoi 12 gol in 26 partite, è lui il *Top Player* del Parma.

Le nomine di *Sorpresa* e di *Assistman* spettano in questo caso allo stesso giocatore. Il classe 2000 *Dejan Kulusevski* ha effettivamente sorpreso tutti alla sua prima stagione tra i grandi, per la gioia di chi ci ha puntato a inizio anno: la *quotazione iniziale* era di soli 4 crediti, e a fine stagione i numeri parlano di 36 partite giocate, una fantamedia di 7,43 condita da 9 gol e 8 assist.

Il *Muro* difensivo del Parma è stato invece *Riccardo Gagliolo*, con una media voto pulita appena sufficiente. Forse la squadra emiliana dovrebbe lavorare un po' per quanto riguarda la retroguardia: difensori del Parma sconsigliati.

Il *Flop* non a caso è stato proprio un difensore, *Giuseppe Pezzella* è giovane e ha tempo di imparare, ma per il momento con la sua fantamedia di 5,66 nella nostra rosa è meglio non averlo.

7.2. Calcolo Best Mf Team

La seconda parte dell'applicazione è invece detta "Tappabuchi" e permette di selezionare una quantità limitata di giocatori per ruolo da dei menu a tendina e di digitare una quantità di crediti disponibili.

Il "Tappabuchi"

Crediti a disposizione:

Portieri: Difensori:

Centrocampisti: Attaccanti:

L'algoritmo di calcolo della rosa è infatti piuttosto costoso a livello computazionale, e aumentando in modo eccessivo i crediti a disposizione o il numero di giocatori richiesti, il sistema potrebbe bloccarsi.

Sono stati dunque inseriti controlli nel programma, per far sì che la domanda dell'utente rispetti i seguenti vincoli:

- Non devono essere selezionati più di 4 giocatori;
- Selezionando 4 giocatori, deve essere inserito un budget non superiore ai 20 crediti;
- Selezionando 3 giocatori, deve essere inserito un budget non superiore ai 35 crediti.

Il "Tappabuchi"

Crediti a disposizione:

Portieri: Difensori:

Centrocampisti: Attaccanti:

Inserisci un massimo di 35 crediti, altrimenti abbassa il numero di giocatori!

Il "Tappabuchi"

Crediti a disposizione:

Portieri: Difensori:

Centrocampisti: Attaccanti:

Seleziona un massimo di 4 giocatori!

Andiamo ora a vedere un esempio pratico, inserendo un budget a disposizione di 30 crediti, 1 portiere e 2 centrocampisti.

Il "Tappabuchi"

Crediti a disposizione:

Portieri: Difensori:

Centrocampisti: Attaccanti:

C: PEROTTI (Roma) - QtA: 13 - FV: 7,00
 C: ALLAN (Napoli) - QtA: 5 - FV: 6,26
 P: STRAKOSHA (Lazio) - QtA: 12 - FV: 5,16

Il sistema calcola la soluzione migliore.

Dai calcoli effettuati risulta utile investire la maggior parte del budget su due giocatori: il portiere *Thomas Strakosha* della Lazio, con una fantamedia di 5,16 di tutto rispetto per un portiere, e l'esterno offensivo della Roma *Diego Perotti*, fantamedia del 7 e tanti bonus per lui.

Per completare il terzetto, *Allan* è la scelta ideale, in quanto a soli 5 crediti riusciremo a portarci a casa un giocatore dalla fantamedia di 6,26.

7.3. Calcolo Best MV Team

Vediamo invece un esempio di calcolo di Best MV Team.

Il Best MV Team calcola la rosa migliore in termini di media voto pulita. Questo tool può risultare molto utile quando si utilizza il cosiddetto *Modificatore Difesa*, una possibile aggiunta alle regole base del fantacalcio che prevede l'assegnazione di punti bonus a seconda della media dei voti presi dai giocatori della retroguardia.

Mettiamo caso che ci siano rimasti soltanto 18 crediti, e di aver bisogno ancora di due giocatori nel ruolo Portiere e di altri due giocatori nel ruolo Difensore.

The screenshot shows a web interface titled "Il 'Tappabuchi'". It features a form for calculating the Best MV Team. At the top, there is a text input field labeled "Crediti a disposizione:" with the value "18". Below this, there are four dropdown menus for selecting the number of players for each position: "Portieri:" (set to 2), "Difensori:" (set to 2), "Centrocampisti:" (set to 0), and "Attaccanti:" (set to 0). Below the dropdowns are two buttons: "Calcola Best MF Team" and "Calcola Best MV Team", with the latter being highlighted with a blue border. Below the buttons, a list of player suggestions is displayed, each with their position, name, club, and calculated MV value:

- P: BERISHA E. (SPAL) - QtA: 5 - MV: 6,31
- P: JORONEN (Brescia) - QtA: 5 - MV: 6,29
- D: KUMBULLA (Verona) - QtA: 6 - MV: 6,15
- D: CISTANA (Brescia) - QtA: 2 - MV: 5,79

Ed ecco un buon quartetto.

Se il nostro interesse è avere un portiere che, al di là dei gol subiti, possa portare a casa dei buoni voti in pagella, *Etrit Berisha* e *Jesse Joronen* sono due ottime soluzioni e ci consentiranno di spendere 10 crediti in totale. Per la difesa puntiamo tutto su *Marash Kumbulla*, il giovanissimo talento del Verona, e concludiamo il pacchetto difensivo con *Andrea Cistana*, fantamedia al di sotto della sufficienza ma tante partite giocate, utile come ultima scelta di reparto in caso di emergenza.

7.4. Link al video YouTube

Un video dimostrativo sull'utilizzo dell'applicazione è disponibile al link <https://youtu.be/1SDvpNchbGc>

8 Valutazione sui risultati ottenuti e conclusioni

Quando ho pensato a questa applicazione l'obiettivo era di fornire un aiuto a chi, giocando al fantacalcio, si ritrova a dover completare la propria rosa, cercando giocatori a prezzi vantaggiosi per riempire gli ultimi slot rimasti liberi.

Per quanto riguarda la prima parte dell'applicazione, tutto ha funzionato come avevo previsto, essendo anche gli algoritmi molto semplici. Il programma dà buoni consigli, mostrando sia i giocatori più famosi (Top Player, Assist Man), sia alcuni nomi più ricercati, che potrebbero esserci sfuggiti.

Le migliorie apportabili a questa sezione possono essere davvero tante: l'aggiunta di nuovi giocatori chiave, la possibilità di scegliere per quali giocatori vedere statistiche o ancora l'aggiunta di campi per poter personalizzare la ricerca, come per esempio vincolare il prezzo massimo di un giocatore.

La semplicità dei calcoli effettuati sui dati rende possibile davvero tante varianti, e aggiornare l'applicazione basandosi sul feedback degli utenti non sarebbe un problema.

Per quanto riguarda la seconda parte del programma invece, sapevo che un algoritmo che cercasse una soluzione ideale tra tutti i ruoli sarebbe stato abbastanza costoso in termini computazionali, ma la mia speranza era che i termini di ricerca potessero essere un po' più flessibili di quelli che si sono poi effettivamente dimostrati possibili. In ogni caso trovo che l'applicazione possa comunque essere utile: quando si crea una rosa si parte solitamente dai giocatori migliori, e quindi più costosi. Non è dunque difficile immaginare una situazione in cui le nostre necessità saranno molto limitate, al pari delle nostre disponibilità.

Anche in questo caso, seppure in modo più ristretto, si potrebbero aggiungere nuove funzionalità, come ad esempio qualche vincolo sui giocatori scelti dal *Tappabuchi*, valorizzando le presenze oppure i gol segnati.

In generale posso dirmi soddisfatto di aver lavorato a questa applicazione. Il mio percorso di studi, in particolar modo i corsi di Programmazione ad Oggetti, Basi di Dati e Tecniche di Programmazione mi hanno permesso di apprendere le basi per lo sviluppo di piccole applicazioni. Applicare questi nuovi concetti ad uno dei miei passatempi preferiti è stato sicuramente un bel modo per concludere la mia Laurea Triennale.



Quest'opera è distribuita con Licenza [Creative Commons Attribuzione - Non commerciale - Condividi allo stesso modo 4.0 Internazionale](https://creativecommons.org/licenses/by-nc-sa/4.0/)