

Relazione Fantaroyale

Domenico Tripepi

January 17, 2022

1 Panoramica

FantaRoyal è una piattaforma che nasce con l'intento di unire e far divertire chiunque sia appassionato al fantacalcio. Chiunque voglia registrare un account presso il sito web dedicato, potrà usufruire dei seguenti servizi:

- Sfidare centinaia di utenti partecipando a varie competizioni
- Vincere premi in denaro settimanali e annuali
- Restare aggiornato sulle news riguardanti il mondo del calcio giorno per giorno

1.1 Come si gioca?

Una volta registrato l'utente dovrà iscrivere una squadra ad una competizione, una gara che vede coinvolti diversi giocatori che lottano di giornata in giornata schierando la formazione ideale, formata dai loro calciatori preferiti, per accaparrarsi il primo posto della classifica. Ogni giornata, ogni giocatore dovrà comprare 15 CALCIATORI utilizzando il budget a disposizione. In seguito ne vanno schierati 11 e 4 (uno per ruolo) andranno in panchina. Ciascun giocatore durante le partite giocate prenderà una VALUTAZIONE in relazione alla prestazione calcistica da esso sostenuta. La somma delle valutazioni dei giocatori schierati costituisce il punteggio che la formazione di quel giocatore ha ottenuto in quella giornata. Se un calciatore degli 11 titolari non dovesse giocare, verrà sostituito dal calciatore con lo stesso ruolo che però siede in panchina. I punteggi della giornata, delle squadre iscritte ad una competizione vanno a formare la CLASSIFICA SETTIMANALE. Il punteggio ottenuto in ciascuna giornata da un giocatore inoltre, va a sommarsi di settimana in settimana per contribuire al punteggio della CLASSIFICA GENERALE.

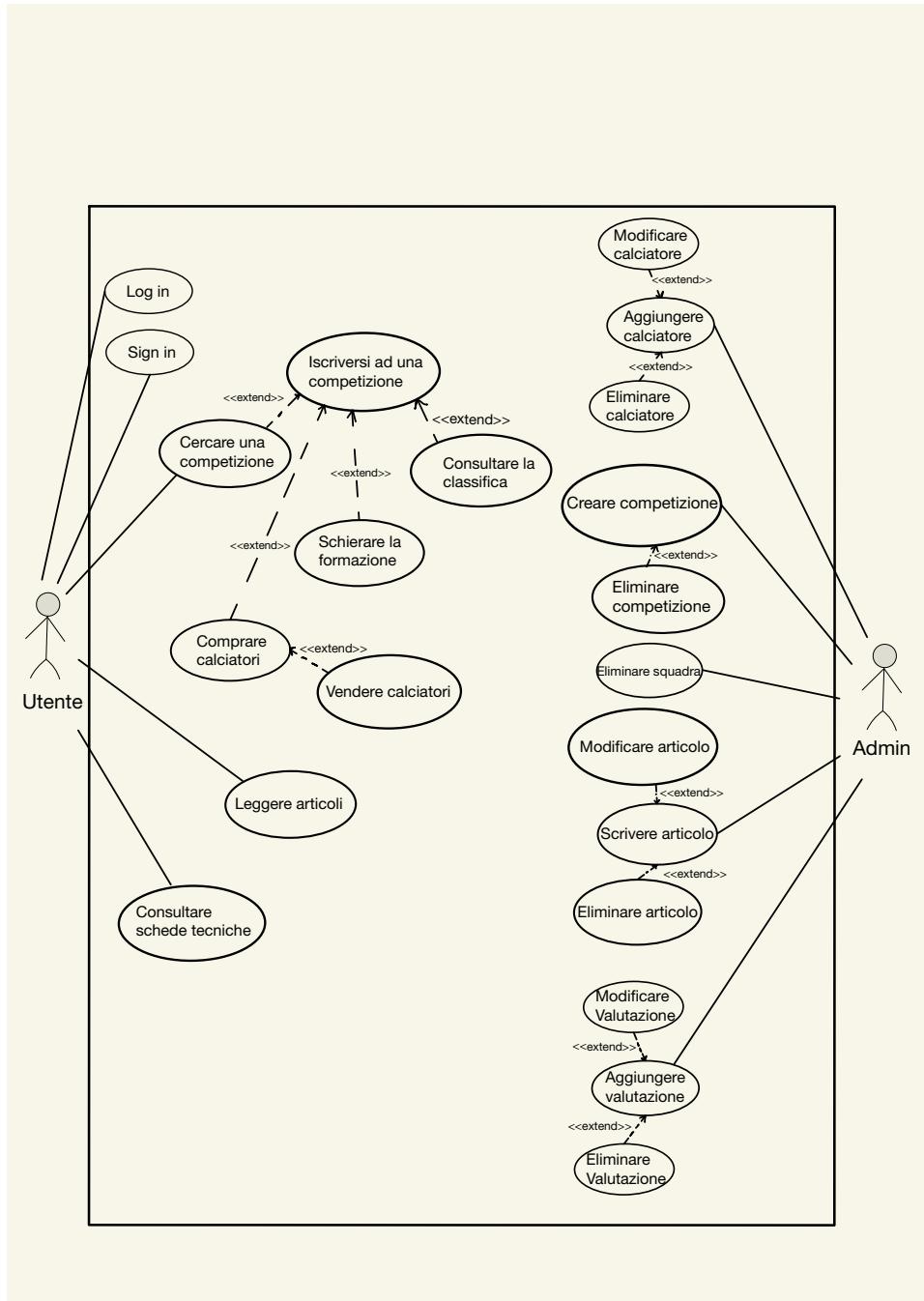
- Valutazione: la valutazione di un calciatore, oltre che dal voto attribuitogli da una testata giornalistica sportiva, è influenzato anche da bonus e malus (Gol, Assist, Ammonizioni,...) che si sommano al voto stesso formando il Fantavoto.

2 Elenco delle specifiche

- Ciascun giocatore deve poter registrare un account fornendo email, password, email collegata al proprio conto PayPal e scegliendo un nickname. Ogni utente sarà inoltre identificato univocamente da un codice.
- Ciascun giocatore può creare una squadra alla quale dovrà assegnare un nome, sarà inoltre identificata da un ID univoco. La squadra dovrà dunque essere iscritta ad una competizione.
- Un giocatore può possedere una sola squadra all'interno della stessa competizione.
- Per iscrivere la propria squadra ad una competizione sarà necessario pagare una quota di iscrizione annuale che varia di competizione in competizione.
- Le competizioni sono divise in giornate che corrispondono alle giornate del campionato italiano di Serie A.
- Per ogni competizione ci saranno due tipi di classifiche, una che riguarda la singola giornata, che vedrà vincitore solamente il primo classificato, ed una seconda classifica annuale, che vedrà premiati invece i primi tre classificati.
- L'ammontare dei premi dipende sempre dal numero di partecipanti alla competizione e dal costo dell'iscrizione.
- Ogni giornata, ogni giocatore dovrà comprare 15 CALCIATORI: 2 portieri, 4 difensori, 5 centrocampisti e 4 attaccanti. Di questi, 11 (1 portiere - 3 difensori - 4 centrocampisti - 3 attaccanti) andranno schierati, e i restanti 4 (uno per ruolo) andranno a formare la panchina. Questo per ogni squadra posseduta.
- Il punteggio conseguito da ciascun giocatore è determinato dalla somma delle valutazioni dei calciatori schierati.
- Ciascun calciatore è identificato da un ID univoco e sarà caratterizzato da un nominativo, un prezzo, il ruolo che ricopre e il club a cui appartiene.
- Ad ogni calciatore è associata una scheda tecnica che ne riporta tutte le statistiche del campionato corrente.
- Ogni giocatore avrà a disposizione una pagina dedicata al proprio profilo nella quale sarà possibile visualizzare tutte le squadre del giocatore e modificare le proprie credenziali.
- La piattaforma sarà gestita da degli amministratori.

- Gli amministratori (identificati univocamente da un codice ID) riceveranno un nominativo e una password con i quali avranno accesso alla piattaforma.
- Gli amministratori devono pubblicare articoli nell'apposita sezione, specificando titolo, tipologia e contenuto.
- Gli amministratori devono creare le competizioni, specificando nome, massimo di iscritti, prezzo di iscrizione e budget acquisti.

Use Case Diagram

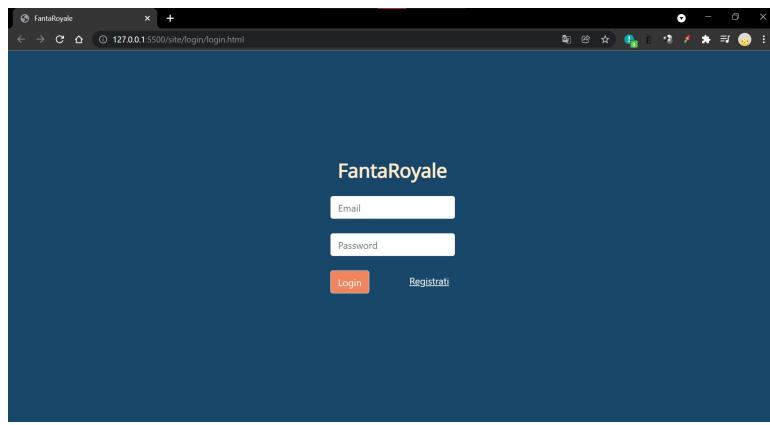


3 Interfaccia grafia

Interfaccia utente

L'utente dell'applicazione per poter usufruire dei servizi deve necessariamente effettuare il login, dunque la prima pagina che viene visualizzata è proprio quella di login.

3.0.1 login



L'utente dovrà semplicemente inserire i dati con i quali si è registrato e se sono validi, verrà reindirizzato alla home del sito.

Un nuovo utente che non possiede ancora un account potrà cliccare sul link di registrazione in basso a destra del form e provvedere all'iscrizione.

3.0.2 signin

The screenshot shows a web browser window with the URL `127.0.0.1:5500/site/signin/signin.html`. The page title is "FantaRoyale SignIn". It features a form with five input fields: "Email", "Password", "Password Confirm", "Nickname", and "Email PayPal". A red "Sign In" button is located at the bottom of the form.

In questa pagina è possibile per un nuovo utente creare un account con il quale poter navigare tra le pagine del sito.

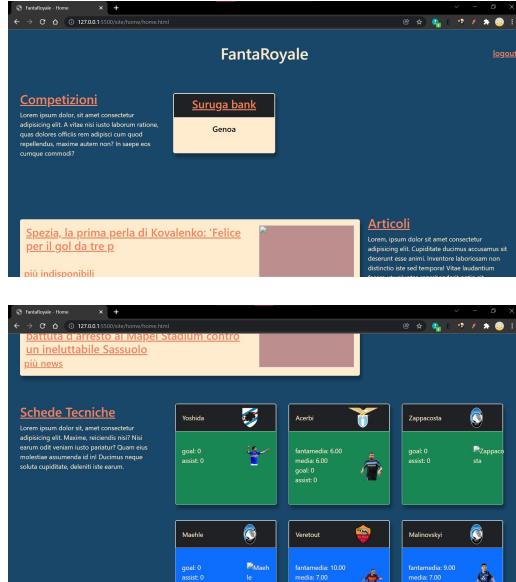
Una volta compilato il form, cliccando sul pulsante "Sign In", se tutti i campi sono stati riempiti correttamente, l'utente verrà reindirizzato alla pagina di login.

3.0.3 profilo

The screenshot shows a web browser window with the URL `127.0.0.1:5500/site/profilo/profilo.html`. The page title is "Profilo del presidente Frank". On the left, there is a sidebar titled "Lista Squadre" with four items: "Suruga bank", "Genoa", "Karl-Rappan", and "Sossi". On the right, there is a form titled "Aggiorna Profilo" with fields for "Nome" (containing "Frank"), "Email" (containing "francesco@gmail.com"), "new password", and "cicc1@paypal.it". A green "Aggiorna" button is at the bottom of the form.

Il profilo è una pagina dedicata al giocatore nella quale troviamo, sul lato sinistro, l'elenco delle squadre di cui è il presidente e sul lato destro un form che permette l'update delle proprie credenziali

3.0.4 home



Una volta effettuato il login, l'utente viene reindirizzato alla home. Questa fornisce un'anteprima dei contenuti delle singole sezioni del sito e ne permette la navigazione.

Le sezioni in questione sono Competizioni, Articoli e Schede Tecniche.

Ciascuna sezione fornisce una breve descrizione delle proprie funzionalità e in particolare:

- nella sezione delle competizioni, vengono mostrate le competizioni a cui è iscritto l'utente. Cliccando sul link "Competizioni" si viene reindirizzati alla pagina contenente tutte le competizioni esistenti, mentre cliccando sulla scheda della singola competizione viene mostrata la pagina ad essa dedicata
- nella sezione degli articoli, vengono mostrate le anteprime di 3 articoli (uno per tipologia) che come per le competizioni possono essere visitati singolarmente cliccando sui rispettivi titoli o avere una vista completa di tutti gli articoli esistenti cliccando sul link "Articoli"
- nella sezione delle schede tecniche invece viene mostrata solo un'anteprima di 9 schede tecniche (3 per ruolo esclusi i portieri) ed è possibile, come per le precedenti sezioni, accedere alla pagina cliccando su "Schede Tecniche"

3.0.5 competizioni

The screenshot shows a web browser window titled "Competizioni". On the left, there is a search bar with placeholder text "Cerca..." and a "vai!" button. Below it is a filter section with a dropdown menu set to "Selezionare un filtro..." and a "Rimuovi filtri" button. To the right, there are three cards representing different competitions:

- Suruga bank**
prezzo iscrizione: 150
numero iscritti: 1
massimo iscritti: 50
budget: 350
data creazione: 2021-08-25
data termine: 2021-06-14
- Karl-Rappan**
prezzo iscrizione: 35
numero iscritti: 2
massimo iscritti: 15
budget: 300
data creazione: 2021-08-24
data termine: 2021-11-10
- Mitropa**
prezzo iscrizione: 40
numero iscritti: 1
massimo iscritti: 10
budget: 280
data creazione: 2021-08-24
data termine: 2021-08-25

In questa pagina sulla sinistra troviamo un form di ricerca che consente all'utente di filtrare le competizioni per cercare quelle che più lo soddisfano.

Sulla destra troviamo delle schede con le informazioni di ogni competizione e un link per accedere alle relative pagine

3.0.6 singola competizione

The screenshot shows a web browser window titled "Competizione" for the "Suruga bank" competition. On the left, there is a sidebar with "Info competizione" containing participant details: 1 participant, budget: 350, start date: 2021-08-25 12:44:15, end date: 2021-06-14 23:59:00. Below it are two buttons: "Mercato" and "Classifica". The main content area is titled "Formazione Titolare" and shows a table of player statistics for "giornata 16".

Nome	voto	goal	assist	ammonizioni	espulsioni	autogoa	sbagliato	inviolata	subiti	fantavoto
Reina										
Skriniar										
Zappacosta										
Acerbi										
Kessie										
Malinovský										
Veretout										
Mæhle										
Vlahovic										
Insigne										
Zapata										
Totali										

Questa pagina viene visualizzata in maniera differente a seconda se si è iscritti o meno alla competizione:

- Se non si possiede una squadra iscritta, in basso a sinistra viene visualizzato un input field che ci permette di inserire il nome della squadra se si desidera partecipare alla competizione (il pulsante iscriviti nell'implementazione completa del sito sarebbe sostituito dal PayPal button che consentirebbe all'utente di pagare il prezzo dell'iscrizione).

- una volta completata l'iscrizione al posto dell'input field vengono visualizzati due bottoni che reindirizzano l'utente al Mercato e alla Classifica della competizione (saranno approfonditi entrambi più avanti).

Sulla destra a questo punto sarà possibile visualizzare la Formazione schierata per la giornata attuale con le valutazioni dei calciatori, in caso questa non fosse schierata verrà visualizzato il messaggio "formazione non schierata"

3.0.7 mercato

The screenshot shows a web browser window titled 'Mercato'. At the top, it displays 'Budget: 14'. On the right, there's a 'Home' link. The main area contains two tables. The first table, titled 'portieri', lists players like Handanovic, Szczesny, Musso, Maignan, Audero, and Cragno with their prices (17, 16, 15, 15, 10, 9) and clubs (Inter, Juventus, Atalanta, Milan, Sampdoria, Cagliari). Each row has a green 'acquista' button. The second table lists players like Meret, Reina, Cuadrado, Skriniar, Zappacosta, Acerbi, Chiesa, Kessie, and Malinovskiy with their prices (14, 14, 19, 16, 15, 12, 29, 26, 25) and clubs (Napoli, Roma, Juventus, Inter, Atalanta, Lazio, Juventus, Milan, Atalanta). Each row has a checkbox in the 'Schiera' column and a red 'vendi' button.

portieri			
Calciatore	Prezzo	Club	Acquista
Handanovic	17	Inter	<button>acquista</button>
Szczesny	16	Juventus	<button>acquista</button>
Musso	15	Atalanta	<button>acquista</button>
Maignan	15	Milan	<button>acquista</button>
Audero	10	Sampdoria	<button>acquista</button>
Cragno	9	Cagliari	<button>acquista</button>

Calciatore	Prezzo	Club	Schiera	Vendi
Meret	14	Napoli	<input type="checkbox"/>	<button>vendi</button>
Reina	14	Roma	<input checked="" type="checkbox"/>	<button>vendi</button>
Cuadrado	19	Juventus	<input type="checkbox"/>	<button>vendi</button>
Skriniar	16	Inter	<input checked="" type="checkbox"/>	<button>vendi</button>
Zappacosta	15	Atalanta	<input checked="" type="checkbox"/>	<button>vendi</button>
Acerbi	12	Lazio	<input checked="" type="checkbox"/>	<button>vendi</button>
Chiesa	29	Juventus	<input type="checkbox"/>	<button>vendi</button>
Kessie	26	Milan	<input checked="" type="checkbox"/>	<button>vendi</button>
Malinovskiy	25	Atalanta	<input checked="" type="checkbox"/>	<button>vendi</button>

Il mercato è la pagina dove l'utente può acquistare i calciatori e schierare la formazione.

Sul lato sinistro è presente il budget a disposizione che varierà in maniera dinamica ad ogni acquisto e vendita di ogni calciatore.

Subito sotto vi sono 4 sezioni di calciatori, raggruppati per ruolo. Quando si acquista uno di questi cliccando sull'apposito pulsante, esso scomparirà dalla tabella dei calciatori acquistabili e verrà inserito nella tabella di sinistra che rappresenta l'anteprima della formazione che si andrà a schierare.

I titolari dovranno essere schierati spuntando l'apposita checkbox e i restanti verranno messi automaticamente in panchina.

Una volta selezionati i titolari cliccando su "Schiera Formazione" la formazione sarà schierata per la giornata corrente. Se si desidera apportare dei cambiamenti è possibile farlo cliccando sul pulsante "Modifica Formazione" che sarà visualizzato al posto del precedente una volta schierata la formazione.

È possibile accedere al mercato fintanto che la giornata non è ancora iniziata.

3.0.8 classifica

Questa pagina mostra all'utente le due classifiche che caratterizzano ogni competizione:

- sulla sinistra è presente la classifica giornaliera che mostra in ordine decrescente i punteggi che ogni squadra ha ottenuto nella singola giornata.

È possibile visualizzare anche le classifiche giornaliere delle giornate precedenti.

La squadra che si classifica prima nella giornata vince un premio.

- sulla destra è presente la classifica generale che mostra in ordine decrescente la somma dei punteggi giornalieri di ciascuna squadra.

I primi tre classificati saranno coloro che vinceranno i premi a fine stagione.

3.0.9 articoli



In questa pagina sono presenti inizialmente tutti gli articoli in ordine decrescente di pubblicazione.

È possibile visualizzare un'anteprima del contenuto, l'immagine, la data di pubblicazione, l'autore, la tipologia e il titolo.

Cliccando su quest'ultimo sarà possibile visualizzarne il contenuto per intero.

In alto è presente una barra di navigazione che permette di visualizzare gli articoli raggruppati per tipologia.

3.0.10 schede tecniche

The screenshot shows a web browser window titled "Info Calciatori". On the left, there is a search bar with placeholder text "Cerca..." and a "vali" button. Below it are three dropdown menus: "Ruolo" set to "Centrocampista", "Club" set to "Selezione un club...", and "Filtro per" set to "Selezione un filtro...". A yellow "Rimuovi filtri" button is also present. On the right, there are two player cards. The first card is for "Malinovskyi" (Inter logo), showing statistics: fantamedia: 9.00, media: 7.00, goal: 0, assist: 0, ammonizioni: 0, espulsioni: 0, porta inviolata: 0, goal subiti: 0, autogoa: 0. The second card is for "Kessie" (AC Milan logo), showing similar statistics: fantamedia: 6.00, media: 6.00, goal: 0, assist: 0, ammonizioni: 0, espulsioni: 0, porta inviolata: 0, goal subiti: 0, autogoa: 0. Both cards feature a small image of the player in their respective team's kit.

La pagina fornisce informazioni sui singoli calciatori raggruppando tutti i dati necessari in delle schede.

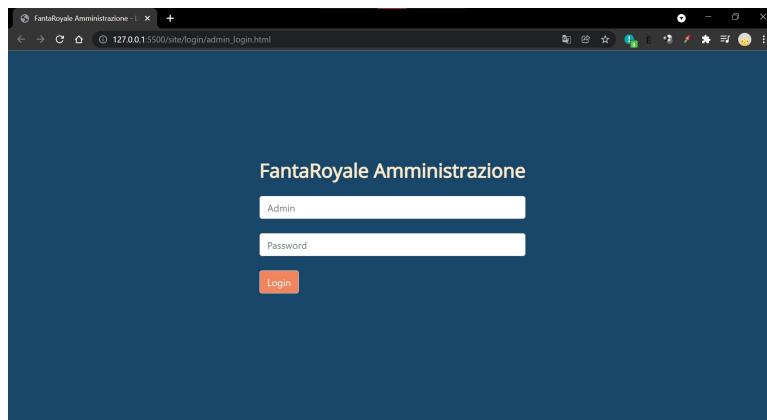
Sul lato sinistro è presente un form di ricerca che agevola la visualizzazione delle schede, sul lato destro invece vengono visualizzate le schede.

Ciascuna scheda è caratterizzata da:

- nome calciatore
- stemma del club in cui il calciatore milita
- numero di goal, assist, ammonizioni, espulsioni, clean sheet, goal subiti e autogoa
- immagine del calciatore

Interfaccia admin

3.0.11 login



Gli amministratori non hanno la possibilità di registrarsi, ricevono il loro nickname e la loro password con i quali possono accedere alla parte amministrativa tramite questa pagina di login.



Una volta effettuato l'accesso, l'admin viene reindirizzato alla home dell'amministrazione tramite la quale è possibile navigare tra le differenti sezioni.

3.0.12 competizioni

The screenshot shows a web application window titled "Amministra competizioni". On the left, there's a sidebar with buttons for "Crea una nuova competizione" (Create a new competition), a search bar, a filter dropdown set to "Selezionare un filtro...", and a "Rimuovi filtri" (Remove filters) button. The main area displays three competition entries in orange boxes:

- Suruga bank**
prezzo iscrizione: 150
numero iscritti: 1
massimo iscritti: 50
budget: 350
data creazione: 2021-08-25
12:44:15
data termine: 2021-06-14
23:59:00

[Elimina competizione](#)
- Karl-Rappan**
prezzo iscrizione: 35
numero iscritti: 2
massimo iscritti: 15
budget: 300
data creazione: 2021-08-24
17:07:03
data termine: 2021-11-10
12:27:45

[Elimina competizione](#)
- Mitropa**
prezzo iscrizione: 40
numero iscritti: 1
massimo iscritti: 10
budget: 280
data creazione: 2021-08-24
17:04:51
data termine: 2021-08-25
12:33:39

[Elimina competizione](#)

La pagina è molto simile a quella visualizzata dagli utenti, ma presenta degli elementi che permettono all'amministratore di gestire le competizioni:

- il pulsante "Crea una nuova competizione" mostra un form che consente, una volta compilato correttamente, di creare una nuova competizione
- il form di ricerca viene mantenuto per facilitare la ricerca delle competizioni da gestire
- sotto ogni scheda delle competizioni è presente un pulsante "Elimina competizione" che consente, appunto, di eliminare la competizione in questione

3.0.13 valutazioni

giornata:	Selezionare una gio	voto	goal	assist	ammonizioni	espulsioni	autogoa	rigore sbagliato	porta violata	goal subiti	fantavoto	aggiorna	elimina
Gosens		7	1	0	0	0	0	0	0	0	10	aggiorna	elimina
Handanovic		6	0	0	0	0	0	0	1	0	7	aggiorna	elimina
Hemandez		4	0	0	0	1	0	0	0	0	3	aggiorna	elimina
Cuadrado		4	0	0	0	1	0	0	0	0	3	aggiorna	elimina
Skriniar		6	0	0	0	0	0	0	0	0	6	aggiorna	elimina
Acerbi		6	0	0	0	0	0	0	0	0	6	aggiorna	elimina
Mkhitarian		7	1	1	0	0	0	0	0	0	11	aggiorna	elimina
Chiesa		6	0	0	0	0	0	0	0	0	6	aggiorna	elimina
Kessie		6	0	0	0	0	0	0	0	0	6	aggiorna	elimina
Malinovskiy		7	0	2	0	0	0	0	0	0	9	aggiorna	elimina

Questa pagina si presenta come una grande tabella contenente la valutazione di ciascun giocatore giornata per giornata che può essere selezionata dalla look-up table nell'angolo superiore sinistro della tabella.

- Per aggiornare una valutazione è necessario cliccare sul pulsante "aggiorna" e modificare il campo desiderato
- Per eliminare una valutazione, basta cliccare il rispettivo pulsante "elimina"
- In alto, vicino al titolo della pagina, è presente il pulsante "Aggiungi valutazione" che apre un form in una nuova finestra tramite il quale è possibile creare una nuova valutazione per un giocatore. (idealmente questa operazione dovrebbe essere automatizzata)

3.0.14 giocatore

id_giocatore	nickname	squadre	elimina
26	ninni	<button>get squadre</button>	<button>elimina</button>
25	ilDuro	<button>get squadre</button>	<button>elimina</button>
24	ciccone	<button>get squadre</button>	<button>elimina</button>
23	dedone	<button>get squadre</button>	<button>elimina</button>
22	meeeo	<button>get squadre</button>	<button>elimina</button>
21	Paolol	<button>get squadre</button>	<button>elimina</button>
20	filippo	<button>get squadre</button>	<button>elimina</button>
18	xxxMarjo	<button>get squadre</button>	<button>elimina</button>
17	nellino	<button>get squadre</button>	<button>elimina</button>

Questa pagina permette all'amministratore di eliminare l'account di un giocatore, vedere quali squadre esso possiede ed eventualmente eliminare.

Cliccando su "get squadre" viene aperta una nuova finestra contenente le squadre del giocatore corrispondente.

3.0.15 calciatori

The screenshot shows a web interface for managing players. On the left, there is a form with fields for 'nominativo', 'prezzo', 'Ruolo' (with a dropdown menu 'Selezione un ruolo...'), and 'Club' (with a dropdown menu 'Selezionare un club...'). Below these fields is a green button labeled 'Aggiungi calciatore'. On the right, there are two tables of players:

portieri				
Calciatore	Prezzo	Club	Elimina	Aggiorna
Handanovic	17	Inter	<button>elimina</button>	<button>aggiorna</button>
Szczesny	16	Juventus	<button>elimina</button>	<button>aggiorna</button>
Musso	15	Atalanta	<button>elimina</button>	<button>aggiorna</button>
Maignan	15	Milan	<button>elimina</button>	<button>aggiorna</button>
Meret	14	Napoli	<button>elimina</button>	<button>aggiorna</button>
Reina	14	Roma	<button>elimina</button>	<button>aggiorna</button>
Audero	10	Sampdoria	<button>elimina</button>	<button>aggiorna</button>
Cragno	9	Cagliari	<button>elimina</button>	<button>aggiorna</button>

difensori				
Calciatore	Prezzo	Club	Elimina	Aggiorna
Handanovic	17	Inter	<button>elimina</button>	<button>aggiorna</button>
Szczesny	16	Juventus	<button>elimina</button>	<button>aggiorna</button>
Musso	15	Atalanta	<button>elimina</button>	<button>aggiorna</button>
Maignan	15	Milan	<button>elimina</button>	<button>aggiorna</button>
Meret	14	Napoli	<button>elimina</button>	<button>aggiorna</button>
Reina	14	Roma	<button>elimina</button>	<button>aggiorna</button>
Audero	10	Sampdoria	<button>elimina</button>	<button>aggiorna</button>
Cragno	9	Cagliari	<button>elimina</button>	<button>aggiorna</button>

In questa pagina è possibile gestire la lista di tutti i calciatori.

Sul lato sinistro della pagina è presente un form che permette di aggiungere un nuovo calciatore alla lista.

Sul lato destro troviamo i calciatori divisi per ruoli in diverse tabelle, per ciascuno di essi sono presenti due pulsanti, uno permette di eliminare il calciatore l'altro apre una finestra con un form che permette di modificare le caratteristiche del rispettivo calciatore.

3.0.16 articoli

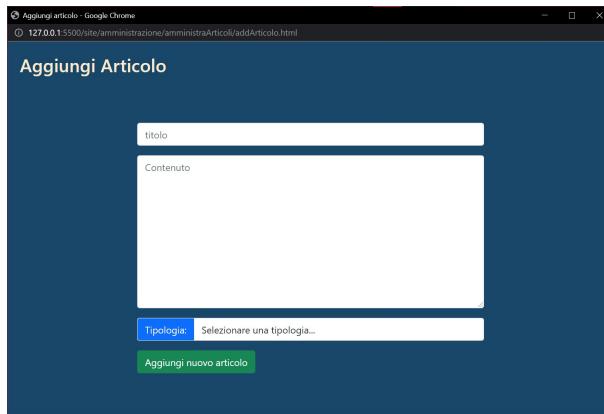


Questa pagina è destinata alla gestione degli articoli.

Il pulsante vicino al titolo permette di creare un nuovo articolo compilando il form che viene mostrato all'interno della finestra che si apre al click del pulsante.

Sotto ogni articolo è presente il pulsante "elimina" che permette di eliminare il rispettivo articolo.

Il pulsante "aggiorna" invece apre una finestra che permette di modificare i dati di un articolo.



4 Dettagli implementativi

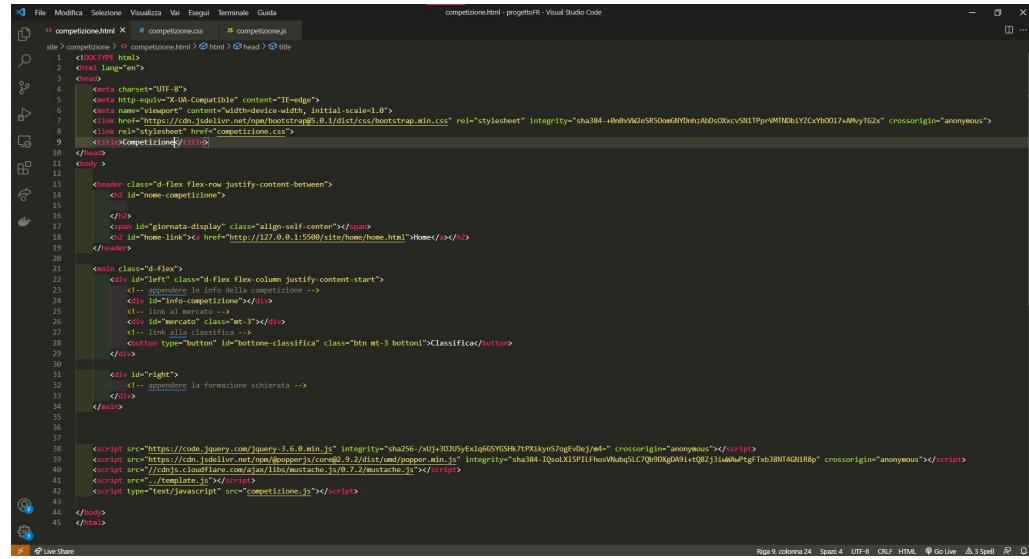
FantaRoyale è realizzata seguendo i principi del paradigma REST, viene fatto uso di un database realizzato con MySQL con il quale l'applicazione interagisce tramite delle api scritte in PHP. Lato client oltre ad HTML e CSS viene utilizzato Bootstrap 5, JavaScript con l'impiego di alcune librerie come jQuery, Mustache, jQuery validator e Moment.

Piccola panoramica sulle librerie

- jQuery è una libreria JavaScript per applicazioni web che ha l'obiettivo di semplificare la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML, nonché semplificare l'uso di funzionalità AJAX, la gestione degli eventi e la manipolazione dei CSS.
- Mustache js permette l'utilizzo di template contenenti il codice di presentazione in HTML per la formattazione dei dati contenuti nelle “viste”, ossia oggetti in javascript, sulla pagina, separando la logica dell'applicazione dal livello di presentazione.
- jQuery validator offre una gestione semplificata della validazione dei form, si crea un oggetto contenente le regole dei campi del form, i messaggi di errore da mostrare nel caso questi ultimi non vengano rispettati e la funzione da eseguire se i campi sono compilati correttamente, infine l'oggetto viene passato alla funzione validate eseguita sul form.
- Moment js semplicemente rende più semplice lavorare con le date in modo che non ci siano incongruenze di tipo con le date che saranno inserite nel database

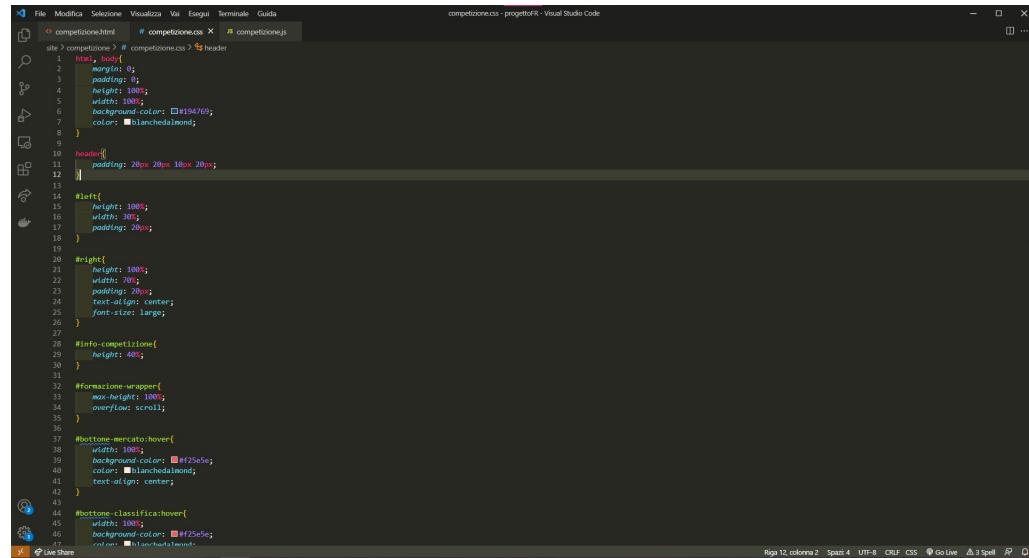
4.1 Organizzazione del front-end

4.1.1 HTML e CSS



```
<!DOCTYPE html>
<html lang="it">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" width="device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-+Rn/54tSlmPAk6ZvYDQG0NcduaTcZbL8qyJq4dI3BpWZlH7zQ8fppu0l4o" crossorigin="anonymous">
    <link rel="stylesheet" href="competizione.css">
    <title>Competizione</title>
  </head>
  <body>
    <div id="gloriosa-display" class="align-self-center">
      <a href="http://127.0.0.1:5500/site/home/home.html">Home</a>
    </div>
    <div id="left" class="d-flex flex-column justify-content-start">
      <!-- Info della competizione -->
      <div id="info-competition"></div>
      <!-- link al mercato -->
      <div id="mercato" class="mt-3"></div>
      <div id="bottoni" class="mt-3">
        <button type="button" id="bottoni-classifica" class="btn mt-3 bottoni">Classifica</button>
      </div>
    </div>
    <div id="right">
      <!-- apprendere la formazione schierata -->
    </div>
  </main>
</body>
```

La struttura del front-end è realizzata tramite l'utilizzo di HTML e CSS principalmente, con l'impiego di Bootstrap5 per il posizionamento degli elementi del documento e per lo styling dei form in particolare.



```
html, body {
  height: 100%;
  width: 100%;
  background-color: #1a1a1a;
  color: #e0e0e0;
}

header {
  padding: 20px 20px 10px 20px;
}

#titolo {
  height: 100px;
  width: 300px;
  padding: 20px;
}

#right {
  height: 100px;
  width: 700px;
  padding: 20px;
  text-align: center;
  font-size: large;
}

#info-competition {
  height: 40px;
}

#formazioni-wrapper {
  max-height: 100px;
  overflow: scroll;
}

#bottoni-mercato:hover {
  width: 100px;
  background-color: #f2f2f2;
  color: #1a1a1a;
  text-align: center;
}

#bottoni-classifica:hover {
  width: 100px;
  background-color: #f2f2f2;
  color: #1a1a1a;
}
```

4.1.2 JavaScript, Mustache e templates

Gli script JavaScript sono divisi in aree di competenza, nella prima parte del documento troviamo tutte le variabili globali e un controllo per verificare che il token sia settato.

Subito dopo abbiamo tutte le chiamate alle funzioni che vanno a recuperare le risorse dal server all'interno di `$(document).ready(() => {})`

```
1 //VARIABILI DI UTILITY
2 const baseURL = "http://localhost:80/server/api/";
3 const token = window.localStorage.getItem("token");
4 var url = new URL(window.location.href);
5 var id_competizione = url.searchParams.get("id");
6 var id_squadra = localStorage.getItem("id_squadra"); //aggiornato da checkIscrizione()
7 var giornata = localStorage.getItem("giornata"); //aggiornato da getGiornata()
8
9 //se il token non è valido il giocatore verrà reindirizzato alla pagina di login
10 ~ if(!token){
11     alert("Utente non autorizzato!");
12     window.location.href = "http://127.0.0.1:5500/site/login/login.html";
13 }
14
15 //DOM
16
17 ~ $(document).ready( () =>
18     checkIscrizione();
19     getGiornata();
20     getCompetizione();
21     getFormazione(calcoloPunteggio);
22
23 );
24 );
```

Di seguito troviamo le implementazioni delle richieste `$.ajax({...})` al server che sono così strutturate:

- type = metodo della richiesta (GET, POST, PUT, DELETE)
- url = path della risorsa sul server
- data = dati utili al server per elaborare la richiesta (presente solo se necessari)
- success = funzione da eseguire se la richiesta è andata a buon fine
- error = funzione da eseguire se la richiesta NON è andata a buon fine
- beforeSend = funzione da eseguire prima che la richiesta venga effettuata

```

28 //richiesta per iscriversi ad una gara competitiva utilizzando l'ida
29 const getCompetition = () => $.ajax({
30   type: "POST",
31   url: baseURL + "competition/competition.php/" + id_competizione,
32   success: openCompetition,
33   error: error,
34   beforeSend: before
35 });
36
37 //richiesta per verificare che un utente sia iscritto o meno alla competizione
38 const checkRegistration = () => $.ajax({
39   type: "GET",
40   url: baseURL + "competition/iscrizione.php/" + id_competizione,
41   success: appendRegistration,
42   error: appendRegistration,
43   beforeSend: before
44 });
45
46 //richiesta per creare una nuova iscrizione per l'utente nella competizione specificata
47 const newRegistration = () => $.ajax({
48   type: "POST",
49   url: baseURL + "competition/iscrizione.php",
50   data: JSON.stringify({
51     nome: $("#nome-squadra").val(),
52     cognome: id_competizione
53   }),
54   contentType: "application/json",
55   success: success,
56   error: error,
57   beforeSend: before
58 });
59
60 //richiesta per recuperare la gara attuale
61 const getGaraAttuale = () => $.ajax({
62   type: "GET",
63   url: baseURL + "giornata/giornata.php",
64   success: setGiornata,
65   error: error,
66   beforeSend: before
67 });
68
69 //richiesta per recuperare la formazione schierata per la giornata attuale
70 const getFormazione = (success) => $.ajax({
71   type: "GET",
72   url: baseURL + "formazione/formazione.php",
73   data: {
74     squadra: id_squadra,
75     giornata: giornata
76   },
77   success: success,
78   error: error,
79   beforeSend: before
80 });

```

E infine le implementazioni delle funzioni di **success**, **error** e **beforeSend**.

Le funzioni di **error** e **beforeSend** sono sempre uguali:

- **error**: si occupano di generare un **alert** che mostra all'utente il messaggio di errore definito nell'api della risorsa a cui si sta tentando di accedere.
- **beforeSend**: si occupano di inserire il token tra gli header della richiesta con l'identificativo di "Authorization"

```

const errore = (error) => {
  alert(error.responseJSON.message);
}

const before = (xhr) => {
  xhr.setRequestHeader("Authorization", token); //inserimento del token tra gli header
}

```

Le funzioni di **success** sono molto diverse tra loro, ma nella maggior parte dei casi fanno uso di **Mustache** per aggiungere al documento i dati recuperati dal server facendo uso dei template, implementati in un file apposito.

```

// FUNZIONI SUCCESS - ERROR - BEFORE

const appendCompetizione = (response) => {
    // console.log(response);

    $("#nome-competizione").innerHTML = response.nome_competizione;
    $("#info-competizione").append(Mustache.render(competizione.competizioneInfo, response));
    $("#classifica").append(Mustache.render(competizione.classificaLink, response));
    localStorage.setItem("budget", response.budget);
}

💡 Funzione che fa visualizzare un form di registrazione per la nuova squadra o il link al mercato a seconda se il giocatore è registrato o meno alla competizione
const appendMercato = (response) => {
    console.log(response)

    $("#mercato").append(Mustache.render(competizione.mercato, response.squadra));
    localStorage.setItem("id_squadra", response.squadra.id_squadra);
    $("#bottone-mercato").click(() => window.location.href = "http://127.0.0.1:5500/site/mercato/mercato.html?id=" + id_competizione);
    $("#bottone-classifica").click(() => location.href = "http://127.0.0.1:5500/site/classifica/classifica.html?id=" + id_competizione);
    getFormazione	appendFormazione();
}

const appendIscrizione = () => {
    $("#mercato").append(competizione.iscrizione);
    $("#bright").remove();
    $("#bottone-classifica").remove();

    $("#iscriviti").click(novaIscrizione);
}

const competizioni = {
    competizioneTemp: '',
    competizioneAmm: ''
}



<div class="card-header">
        <a href="http://127.0.0.1:5500/site/competizione/competizione.html?id={{id_competizione}}">{{nome_competizione}}
</div>
</div>
<div class="card card-scheda">
    <div class="card-header">
        <a href="http://127.0.0.1:5500/site/competizione/competizione.html?id={{id_competizione}}">{{nome_competizione}}



21


```

4.2 Autenticazione tramite token

Per poter navigare tra le pagine del sito, è necessario, tanto per il giocatore quanto per l'amministratore, possedere un token, assegnato a ciascun utente al momento del login.

L'autenticazione è interamente gestita dalla libreria JWT (JSON web token), uno standard che definisce un metodo per la trasmissione sicura di dati tra parti sotto forma di JSON.

Quando l'utente riesce ad effettuare l'accesso al sito inserendo i propri dati nel form di login, viene generato il "jwt", che è il risultato dell'hashing tra i dati contenuti nel token e la chiave di codifica.

```
$token = array(
    "iat" => $issued_at,
    "exp" => $expiration_time,
    "iss" => $issuer,
    "data" => array(
        "id_giocatore" => $giocatore->id,
        "nickname" => $giocatore->nickname,
        "email" => $giocatore->email,
        "email_paypal" => $giocatore->email_paypal
    )
);

// generate jwt
$jwt = JWT::encode($token, $key);

http_response_code(200);
echo json_encode(array("message" => "Login effettuato con successo!", "jwt" => $jwt));
die;
```

Da questo momento ogni qualvolta il client effettuerà una chiamata al server per reperire una risorsa, sarà necessario controllare la validità del token che sarà inserito tra gli header con l'identificativo di "Authorization".

Una volta recuperato e decodificato, sarà possibile chiaramente recuperare i dati contenuti al suo interno.

```
if($_SERVER['REQUEST_METHOD'] != 'OPTIONS'){ // le CORS-Policy richiedono che le chiamate OPTIONS non necessitino dell'Authorization header
$headers = getallheaders(); //array associativo con tutti gli header

// recuperare jwt
$jwt = isset($headers["Authorization"]) ? $headers["Authorization"] : "";

// se jwt non è vuoto
if($jwt){

    // si usa JWT::decode() per recuperare i dati dell'utente dal token
    try {
        // decode jwt
        $decoded = JWT::decode($jwt, $key, array('HS256'));
        $dataToken = (array)$decoded->data;
    }

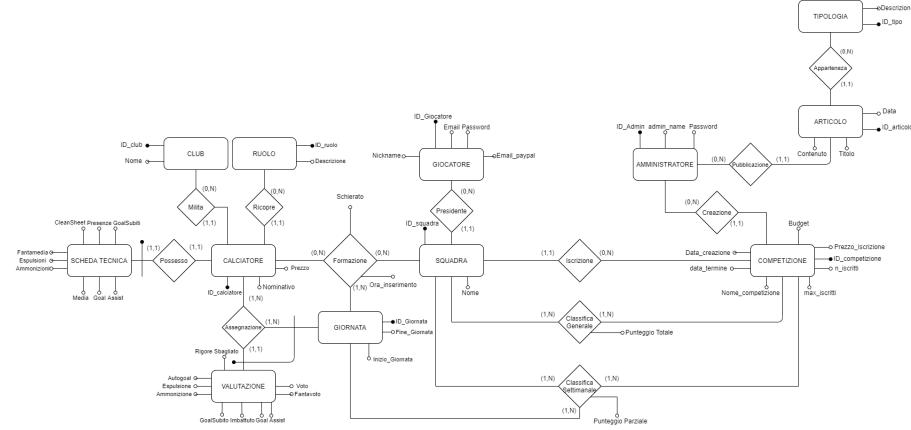
    // se JWT::decode non va a buon fine, significa che il token non è valido
    catch (Exception $e){
        // comunicare all'utente che la navigazione non è autorizzata
        http_response_code(401); //Unauthorized
        echo json_encode(array("message" => "Accesso non autorizzato!")); //messaggio d'errore
        die;
    }
}

// se jwt è vuoto comunicare all'utente che la sua navigazione non è autorizzata
else{
    http_response_code(401);
    echo json_encode(array("message" => "Accesso non autorizzato!"));
    die;
}
```

4.3 Database

In questa sezione viene mostrato il modello ER tramite il quale è stata effettuata la progettazione fisica del database, seguita dalla descrizione delle tabelle e relazioni presenti.

4.3.1 Modello Entity Relationship



4.4 Descrizione entità e relazioni

4.4.1 amministratore

Questa entità contiene i dati di accesso di un admin, ossia il suo admin name e la sua password, inoltre ogni admin è identificato univocamente da un id.

SQL

```
CREATE TABLE `amministratore` (
  `id_admin` int(10) UNSIGNED NOT NULL,
  `admin_name` varchar(64) NOT NULL,
  `passwd` varchar(64) NOT NULL
)
```

4.4.2 articolo

Questa entità è composta dagli attributi che caratterizzano un articolo, cioè il titolo, il contenuto, la data di creazione, la tipologia (che fa riferimento all'id della tipologia contenuta nell'entità tipologia) e l'autore dell'articolo (che fa riferimento all'id dell'amministratore che l'ha scritto). Ciascun articolo è identificato univocamente da un id.

SQL

```
CREATE TABLE `articolo` (
  `id_articolo` int(10) UNSIGNED NOT NULL,
  `titolo` varchar(255) NOT NULL,
  `contenuto` text NOT NULL,
  `data_creazione` timestamp NOT NULL DEFAULT current_timestamp()
    ON UPDATE current_timestamp(),
  `tipologia` int(10) UNSIGNED NOT NULL,
  `autore` int(10) UNSIGNED NOT NULL
)
```

4.4.3 calciatore

Questa entità contiene i dati relativi a ciascun calciatore che interessano al giocatore in fase di acquisto: nominativo, prezzo, ruolo (che fa riferimento all'id del ruolo contenuto nell'entità ruolo) e club (che fa riferimento all'id del club nell'entità club). Ogni calciatore è identificato univocamente da un id.

SQL

```
CREATE TABLE 'calciatore' (
    'id_calciatore' int(10) UNSIGNED NOT NULL,
    'nominativo' varchar(64) NOT NULL,
    'prezzo' int(10) UNSIGNED NOT NULL,
    'ruolo' int(10) UNSIGNED NOT NULL,
    'club' int(10) UNSIGNED NOT NULL
)
```

4.4.4 club

Questa entità contiene semplicemente il nome e l'identificativo di ciascun club presente in serie A.

SQL

```
CREATE TABLE 'club' (
    'id_club' int(10) UNSIGNED NOT NULL,
    'nome' varchar(64) NOT NULL
)
```

4.4.5 competizione

Gli attributi di questa entità definiscono le regole delle competizioni. La data di creazione coincide con l'inizio della competizione e la data di termine, chiaramente, con la fine. Sono presenti anche il prezzo di iscrizione, il numero di iscritti attuali, il massimo di iscritti consentito e il budget che ciascun giocatore avrà a disposizione per comporre la sua squadra giornata per giornata. Ciascuna competizione deve essere creata da un amministratore che è rappresentato dall'attributo creatore, inoltre possiede un nome e un identificatore univoco.

SQL

```
CREATE TABLE `competizione` (
  `id_competizione` int(10) UNSIGNED NOT NULL,
  `nome_competizione` varchar(64) NOT NULL,
  `data_creazione` datetime NOT NULL DEFAULT current_timestamp(),
  `prezzo_iscrizione` int(10) UNSIGNED NOT NULL,
  `numero_iscritti` int(10) UNSIGNED NOT NULL DEFAULT 0,
  `max_iscritti` int(10) UNSIGNED DEFAULT NULL,
  `budget` int(10) UNSIGNED NOT NULL,
  `creatore` int(10) UNSIGNED NOT NULL,
  `data_termine` datetime NOT NULL
)
```

4.4.6 formazione

Questa entità contiene tutti i calciatori acquistati da un giocatore per la rispettiva giornata. Ogni record è identificato dall'insieme di attributi squadra, calciatore e giornata, rappresentati dai loro id. Inoltre sono presenti altri 2 attributi, schierato che consente di distinguere i titolari dai panchinari e ora inserimento che rappresenta il momento in cui è stata inserita la formazione dall'utente. La formazione dunque è costituita dall'insieme dei 15 record che appartengono alla stessa squadra schierati per una determinata giornata.

SQL

```
CREATE TABLE `formazione` (
  `squadra` int(10) UNSIGNED NOT NULL,
  `calciatore` int(10) UNSIGNED NOT NULL,
  `schierato` tinyint(1) NOT NULL DEFAULT 0,
  `giornata` int(10) UNSIGNED DEFAULT NULL,
  `ora_inserimento` timestamp NOT NULL DEFAULT current_timestamp()
  ON UPDATE current_timestamp()
)
```

4.4.7 giocatore

In questa entità sono contenuti i dati di ciascun utente che si registra al sito: nickname, email, password ed email paypal. email e password saranno utilizzati per il login, la mail di paypal per i pagamenti. Ciascun utente sarà caratterizzato da un identificativo univoco. Inoltre è presente un attributo eliminato che serve per mantenere nel database i dati di un utente che ha deciso di eliminare l'account.

SQL

```
CREATE TABLE 'giocatore' (
    'id_giocatore' int(10) UNSIGNED NOT NULL,
    'nickname' varchar(64) NOT NULL,
    'email' varchar(64) NOT NULL,
    'passwd' varchar(64) NOT NULL,
    'email_paypal' varchar(64) NOT NULL,
    'eliminato' tinyint(1) DEFAULT 0
)
```

4.4.8 giornata

Questa entità rappresenta la giornata di campionato durante la quale i giocatori si sfideranno. È identificata da un id che corrisponde anche al numero stesso della giornata. Inoltre sono presenti due attributi per stabilirne l'inizio e la fine.

SQL

```
CREATE TABLE 'giornata' (
    'id_giornata' int(2) UNSIGNED NOT NULL,
    'inizio_giornata' datetime DEFAULT NULL,
    'termine_giornata' datetime DEFAULT NULL
)
```

4.4.9 ruolo

Entità che mantiene i ruoli di ciascun giocatore, ciascun record è costituito da un id e dalla descrizione del ruolo

SQL

```
CREATE TABLE 'ruolo' (
    'id_ruolo' int(10) UNSIGNED NOT NULL,
    'descrizione' varchar(64) NOT NULL
)
```

4.4.10 scheda tecnica

In questa entità sono raccolti tutti i bonus/malus che un calciatore riceve durante l'anno e che influiscono sul fantavoto e di conseguenza sulla fantamedia. I bonus sono goal, assist, e clean sheet. I malus ammonizioni, espulsioni, autogoaal e goal subito.

SQL

```
CREATE TABLE 'scheda_tecnica' (
    'calciatore' int(10) UNSIGNED NOT NULL,
    'goal' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'assist' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'clean_sheet' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'goal_subiti' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'ammonizioni' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'espulsioni' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'autogoaal' int(10) UNSIGNED NOT NULL DEFAULT 0
)
```

4.4.11 squadra

Questa entità rappresenta la squadra con cui l'utente partecipa ad una competizione, esso infatti può iscrivere una sola squadra per competizione. Ogni squadra è identificata univocamente da un id ed è caratterizzata da un nome, dall'identificativo del giocatore che la possiede (presidente) e dall'id della competizione a cui è iscritta.

SQL

```
CREATE TABLE 'squadra' (
    'id_squadra' int(10) UNSIGNED NOT NULL,
    'nome_squadra' varchar(64) NOT NULL,
    'presidente' int(10) UNSIGNED NOT NULL,
    'competizione' int(10) UNSIGNED NOT NULL
)
```

4.4.12 tipologia

Questa entità contiene le tipologie alle quali gli articoli possono appartenere. Ogni record è caratterizzato da un identificatore univoco e da una descrizione della tipologia.

SQL

```
CREATE TABLE 'tipologia' (
    'id_tipologia' int(10) UNSIGNED NOT NULL,
    'descrizione' varchar(64) NOT NULL
)
```

4.4.13 valutazione

Questa entità raccoglie tutte le valutazioni che ciascun giocatore riceve giornata per giornata. L'identificativo del calciatore insieme al numero della giornata identificano univocamente ogni record che si compone degli attributi voto, fantavoto, goal, assist, clean sheet, goal subito, ammonizioni, espulsioni, autogoal.

SQL

```
CREATE TABLE 'valutazione' (
    'calciatore' int(10) UNSIGNED NOT NULL,
    'giornata' int(10) UNSIGNED NOT NULL,
    'voto' float UNSIGNED NOT NULL DEFAULT 0,
    'fantavoto' float UNSIGNED NOT NULL DEFAULT 0,
    'goal' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'assist' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'clean_sheet' int(10) UNSIGNED DEFAULT 0,
    'goal_subiti' int(10) UNSIGNED DEFAULT 0,
    'ammonizioni' float UNSIGNED NOT NULL DEFAULT 0,
    'espulsioni' int(10) UNSIGNED NOT NULL DEFAULT 0,
    'autogoal' int(10) UNSIGNED DEFAULT 0,
    'rigore_sbagliato' int(10) UNSIGNED DEFAULT 0
)
```

4.5 Modelli ed API

Il back-end è organizzato in Modelli che consentono di applicare il *principio di singola responsabilità*, ovvero ogni classe si occupa della gestione di una singola risorsa.

Ogni classe, che dunque rappresenta il modello di ciascuna risorsa del sito, eredita da una classe padre che si chiama *DatabaseManager*.

Questa gestisce la connessione al database e fornisce l'attributo `$conn` utilizzabile in tutte le classi figlie quando necessitano di connettersi al database.

```
class DatabaseManager{
    public $conn;
    public function __construct()
    {
        $this->conn = $this->getConnection();
    }
    //metodi
    public function getConnection(){
        $hostname = "localhost";
        $dbname = "fantasyrole";
        $username = "root";
        $password = "";
        try {
            $connection = new PDO("mysql:host=$hostname", $username, $password);
            $connection->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
        } catch (PDOException $e) {
            http_response_code(500);
            die();
        }
        return $connection;
    }
}
```

4.5.1 Struttura dei modelli

Ogni modello è composto da 4 sezioni:

- **attributi:** caratterizzano la risorsa e corrispondono agli attributi della rispettiva entità nel database
- **costruttore:** semplicemente il costruttore della classe che serve a istanziarne un oggetto

```
</php>
require_once "/Users/Intel/Desktop/progettoFR/server/models/Tipologia.php";
require_once "/Users/Intel/Desktop/progettoFR/server/models/Amministratore.php";
class Articolo extends DatabaseManager{
    //Attributi
    public $id;
    public $titolo;
    public $contento;
    public $data_creazione;
    public $tipologia;
    public $autore;
    public $admin_name;
    public $descrizione;
    //costruttore
    public function __construct()
    {
        parent::__construct();
    }
}
```

- **metodi**: funzioni di utility, spesso usate per effettuare dei controlli

```
//metodo
//funzione per controllare che l'articolo sia presente nel db
public function articoloExist($id)
{
    $connection = $this->conn;

    $query = "SELECT * FROM articolo WHERE id_articolo = ?";

    $stmt = $connection->prepare($query);

    $stmt->bindParam(1, $id);

    $stmt->execute();

    $result = $stmt->fetch(PDO::FETCH_ASSOC);

    if(!$result) return false;

    return true;
}
```

- **metodi CRUD**: metodi per aggiungere, leggere, aggiornare ed eliminare records dal database

```
//CREATE
public function create()
{
    $connection = $this->conn;

    if(!($tipologia > tipologialist($this->tipologia, $connection)) return false;
    if(!($autore > autorelist($this->autore, $connection)) return false;

    //query
    $query = "INSERT INTO articolo
    SET
    titolo = '$titolo',
    contenuto = '$contenuto',
    tipologia = '$tipologia',
    autore = '$autore'";

    $stmt = $connection->prepare($query);

    //sanitize input
    $this->titolo = htmlspecialchars(strip_tags($this->titolo));
    $this->contenuto = htmlspecialchars(strip_tags($this->contenuto));
    $this->tipologia = htmlspecialchars(strip_tags($this->tipologia));
    $this->autore = htmlspecialchars(strip_tags($this->autore));

    $stmt->bindParam(':titolo', $this->titolo);
    $stmt->bindParam(':contenuto', $this->contenuto);
    $stmt->bindParam(':tipologia', $this->tipologia);
    $stmt->bindParam(':autore', $this->autore);

    //execute query
    return $stmt->execute();
}
```

4.5.2 Struttura delle APIs

Ogni api è costituita da più file, uno che funge da root e gli altri svolgono le operazioni CRUD.

Il file *root* ha il compito di:

- fornire gli header che permettono la corretta esecuzione delle chiamate HTTP, nel rispetto delle politiche CORS
- includere i modelli inerenti alle risorse da gestire
- gestire le richieste tramite uno switch case che consente di includere gli altri file di cui l'api si compone a seconda del metodo della richiesta che il server riceve

```
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
header("Access-Control-Allow-Methods: POST, GET, PUT, DELETE");
header("Access-Control-Max-Age: 3600");
header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

$baseProjectURL = '/Users/Intel/Desktop/progettoFR/';

require_once $baseProjectURL.'server/config/databaseManager.php';
require_once $baseProjectURL.'server/models/Articolo.php';
require_once $baseProjectURL.'server/api/validate_token.php';

$parts = explode('/', parse_url($_SERVER['REQUEST_URI'], PHP_URL_PATH));
if(isset($parts[3])) $id = $parts[3];

$articolo = new Articolo();
```

```

switch ($method) {
    case "POST":
        include_once "create_articolo.php";
        break;
    case "GET":
        if($method[$id]) include_once "read_articolo.php";
        include_once "read_articoli.php";
        http_response_code(400);
        break;
    case "PUT":
        if($method[$id]) include_once "update_articolo.php";
        http_response_code(400);
        break;
    case "DELETE":
        if($method[$id]) include_once "delete_articolo.php";
        http_response_code(400);
        break;
    case "OPTIONS":
        http_response_code(200);
        break;
    default:
        http_response_code(405);
        break;
}

```

I file richiamati dalla root api hanno dunque il compito di eseguire l'operazione CRUD corrispondente al metodo della richiesta che il server riceve e fornire una risposta al client con un messaggio e un codice HTTP che comunica l'esito dell'operazione.

Tale operazione è inserita all'interno di un *try and catch* che serve a rilevare eventuali problemi durante l'esecuzione delle query al database.

```

//acquisizione dei dati presenti nel body della richiesta
$data = json_decode(file_get_contents("php://input"));

//inializzazione degli oggetti con i dati precedentemente acquisiti
$articolo->titolo = $data->titolo;
$articolo->contenuto = $data->contenuto;
$articolo->tipologia = $data->tipologia;
$articolo->autore = $data->autore;

//inserimento dei dati nel database
try {
    if($articolo->create()){
        echo json_encode(array("message" => "articolo creato."));
        http_response_code(200);
        die;
    }
    echo json_encode(array("message" => "errore nella creazione dell'articolo"));
    http_response_code(400);
    die;
} catch (PDOException $e) {
    echo json_encode(array("message" => "errore interno al Server."));
    http_response_code(500);
    die;
}

```