Perfetto! È il momento giusto per consolidare i progressi e aggiornare la documentazione del progetto. Un documento ben tenuto è preziosissimo.

Basandomi sui file "Fanta-asta v1.0.docx", "Fanta-asta v1.1.docx", e il file implementazione predictive.txt, e sulle implementazioni che abbiamo discusso e realizzato, ho preparato una versione aggiornata del documento tecnico.

Dato che non posso modificare direttamente i file .docx, ti fornirò il testo in formato Markdown, che potrai facilmente copiare e incollare nel tuo editor di documenti preferito (come Microsoft Word, Google Docs, LibreOffice Writer) e poi formattare come desideri (applicando stili per titoli, grassetti, elenchi, ecc.).

Ho cercato di fare un merge delle informazioni presenti nelle versioni 1.0 e 1.1 del documento, integrandole con le note da "implementazione predictive.txt" e con le funzionalità che abbiamo sviluppato, inclusi i nuovi comandi Artisan. Ho dato precedenza alla struttura e ai contenuti della versione 1.1 dove si sovrapponevano.

# **Documento Tecnico: Applicazione Laravel per Proiezioni e Strategia Asta Fantacalcio**

Versione: 1.2 (Aggiornamento basato sulle implementazioni recenti)

Data: 28 maggio 2025

## **Indice:**

1. Obiettivo del Progetto
2. Architettura di Sistema e Tecnologie
3. Gestione dei Dati 3.1. Fonte Dati Primaria (Input Utente) 3.2. Fonti Dati per Arricchimento (API Esterne) 3.3. Database Applicativo
4. Modulo di Proiezione Performance Calciatori 4.1. Dati di Input per le Proiezioni 4.2. Logica di Calcolo delle Proiezioni 4.3. Output delle Proiezioni
5. Modulo di Valutazione e Identificazione Talenti (Da Implementare) 5.1. Calcolo del "Valore d'Asta Interno" 5.2. Identificazione Giocatori Sottovalutati ("Scommesse")
6. Modulo Strategia d'Asta (Da Implementare) 6.1. Configurazione Lega Fantacalcistica Utente 6.2. Suddivisione Giocatori in Fasce (Tiering) 6.3. Pianificazione Budget per Reparto 6.4. Gestione "Coppie" Titolare/Riserva 6.5. Gestione Diversificazione/Concentrazione per Squadra 6.6. Generazione Lista d'Asta Finale
7. Struttura Applicativa Laravel (Alto Livello) 7.1. Modelli Principali (Eloquent) 7.2. Servizi Chiave (Implementati e Da Implementare) 7.3. Controller e Viste 7.4. Processi in Background (Jobs) 7.5. Comandi Artisan Personalizzati
8. Considerazioni Aggiuntive e Sviluppi Futuri

## **1. Obiettivo del Progetto**

L'obiettivo primario è sviluppare un'applicazione web basata su Laravel che assista l'utente nella preparazione e nella conduzione dell'asta del Fantacalcio (Serie A). L'applicazione fornirà proiezioni personalizzate sulle performance dei calciatori, identificherà giocatori sottovalutati e aiuterà a definire una strategia d'asta ottimale, tenendo conto delle regole specifiche della lega dell'utente e di dinamiche di mercato complesse.

## **2. Architettura di Sistema e Tecnologie**

* **Piattaforma:** Applicazione Web
* **Framework Backend:** Laravel (PHP)
* **Database:** Database relazionale (es. MySQL, PostgreSQL)
* **Frontend:** Blade templates, con possibile utilizzo di JavaScript (es. Livewire, Vue.js o Alpine.js) per interattività.
* **Librerie Chiave:**
  + Maatwebsite/Laravel-Excel per l'importazione/esportazione di file XLSX.
  + Guzzle (tramite client HTTP di Laravel) per le chiamate API esterne.

## **3. Gestione dei Dati**

### **3.1. Fonte Dati Primaria (Input Utente)**

* **File XLSX Roster Ufficiale:** L'applicazione permette l'upload di file XLSX (es. da Fantacalcio.it) contenenti:
  + Lista ufficiale dei calciatori.
  + Ruoli ufficiali (P, D, C, A) e ruoli Mantra (Rm).
  + Quotazioni iniziali (CRD).
  + Un tag/titolo dalla prima riga del foglio "Tutti" viene estratto per riferimento.
* **File XLSX Statistiche Storiche:** L'applicazione permette l'upload di file XLSX per le statistiche storiche dei giocatori (ultime N stagioni).
  + Formato atteso: Riga 1 come titolo/tag, Riga 2 con intestazioni (Id, R, Rm, Nome, Squadra, Pv, Mv, Fm, Gf, Gs, Rp, Rc, R+, R-, Ass, Amm, Esp, Au).
  + La stagione viene derivata dal nome del file.

### **3.2. Fonti Dati per Arricchimento (API Esterne)**

* Per elaborare proiezioni accurate e arricchire i dati dei giocatori, l'applicazione si integra con API esterne (es. **Football-Data.org**).
* **Dati Recuperati (Implementato):**
  + Data di nascita del giocatore (per calcolo età).
  + ID del giocatore sull'API esterna.
  + Posizione/ruolo dettagliato fornito dall'API.
* **Dati Potenziali Futuri dall'API:**
  + Nazionalità, piede preferito, altezza/peso.
  + Statistiche storiche dettagliate (per integrare o sostituire l'import da XLSX).
  + Dati storici sulle performance delle squadre (classifiche, gol) per il tiering dinamico.
  + Statistiche avanzate (xG, xA - Sviluppo Futuro).
* **Dati Qualitativi Curati (Manualmente o Futuri Servizi):**
  + Probabili rigoristi.
  + Giocatori che ricoprono ruoli tattici diversi da quelli ufficiali.
  + Informazioni su gerarchie (titolari/riserve).

### **3.3. Database Applicativo**

Il database memorizza:

* **Players (players):**
  + Dati anagrafici (ID piattaforma Fantacalcio fanta\_platform\_id, nome).
  + Ruolo ufficiale Fantacalcio (role - Classic: P,D,C,A).
  + Quotazione iniziale (initial\_quotation), attuale (current\_quotation), FVM (fvm).
  + team\_id (foreign key a teams).
  + **Campi Arricchiti (da API):** api\_football\_data\_id, date\_of\_birth, detailed\_position.
  + Supporto Soft Deletes.
* **Teams (teams):**
  + Nome ufficiale (name), nome breve (short\_name).
  + serie\_a\_team (boolean).
  + tier (integer): Fascia di forza della squadra, inizialmente da seeder, in futuro calcolato dinamicamente.
  + Opzionale: logo\_url, api\_football\_data\_team\_id.
* **HistoricalPlayerStats (historical\_player\_stats):**
  + Statistiche di un giocatore in una data stagione (player\_fanta\_platform\_id, season\_year).
  + team\_id (foreign key a teams per la squadra di quella stagione).
  + team\_name\_for\_season, role\_for\_season (Classic), mantra\_role\_for\_season.
  + Metriche: Pv, Mv, Fm, Gf, Gs, Rp, Rc, R+, R-, Ass, Amm, Esp, Au.
* **UserLeagueProfiles (user\_league\_profiles):**
  + Configurazione della lega dell'utente (nome, budget, numero giocatori per ruolo, partecipanti, scoring\_rules JSON).
* **ImportLogs (import\_logs):**
  + Traccia le importazioni dei file (nome file, tipo, stato, dettagli, conteggi righe).
* **PlayerTacticalNotes (player\_tactical\_notes):** (Da Implementare)
  + Attributi speciali: rigorista, ruolo tattico offensivo/difensivo, specialista calci piazzati.
* **TeamHistoricalStandings (team\_historical\_standings):** (Da Implementare)
  + Per memorizzare dati storici delle squadre (posizione, punti, gol) per il tiering dinamico.
* **AuctionPlans, AuctionPlanTargets:** (Da Implementare)

## **4. Modulo di Proiezione Performance Calciatori**

### **4.1. Dati di Input per le Proiezioni**

* Medie e FantaMedie storiche ponderate (da HistoricalPlayerStats).
* Età del giocatore (calcolata da date\_of\_birth recuperata via API).
* Ruolo ufficiale Fantacalcio (da Players).
* Ruolo tattico reale (da PlayerTacticalNotes o dedotto da detailed\_position API - sviluppo futuro).
* Forza/fascia della squadra di appartenenza (tier da Teams).
* Minutaggio/Presenze attese (stima interna).
* Status di rigorista/specialista (da PlayerTacticalNotes - sviluppo futuro).

### **4.2. Logica di Calcolo delle Proiezioni (Stato Attuale e Sviluppi)**

* **Recupero Dati Storici:** Caricamento delle statistiche delle ultime N stagioni per il giocatore. Se non presenti, si usano proiezioni di default basate su ruolo, tier squadra ed età.
* **Ponderazione Stagionale:** Pesi decrescenti per le stagioni più vecchie (es. 50%-33%-17%).
* **Calcolo Medie Ponderate per Partita:** Per MV, gol/partita, assist/partita, ecc.
* **Aggiustamento per Età ("Maturità Calcistica"):** (Implementato)
  + Applicazione di un ageModifier basato sull'età del giocatore (calcolata dalla date\_of\_birth fornita dall'API) e curve di rendimento definite per ruolo (fasi di crescita, picco, declino). Il modificatore influenza MV (in modo smorzato), gol/partita, assist/partita, probabilità di clean sheet e presenze attese.
* **Aggiustamento Statistiche per Contesto Squadra (Tier):** (Implementato)
  + Le statistiche offensive (gol, assist) per partita sono moltiplicate per un tierMultiplierOffensive (>1 per squadre forti, <1 per deboli).
  + Le statistiche difensive (gol subiti) per partita sono modulate da un tierMultiplierDefensive (inverso).
  + Il tier attualmente proviene dal TeamSeeder, in futuro sarà calcolato dinamicamente.
* **Stima Presenze Attese:** (Implementato)
  + Parte dalla media ponderata delle partite giocate storicamente (avg\_games\_played).
  + Viene modulata da un presenzeTierFactor (derivato dal tierMultiplierOffensive della squadra) e da un presenzeAgeFactor (derivato dall'ageModifier).
  + Limitata a un range realistico (es. 5-38 partite).
* **Proiezione Clean Sheet per Partita (Difensori/Portieri):** (Implementazione Basilare)
  + Assegnata una probabilità base di clean sheet in base al tier della squadra.
  + Modulata leggermente dall'ageModifier.
* **Calcolo FantaMedia Proiettata per Partita:** (Implementato)
  + Le statistiche medie per partita aggiustate (incluse MV, gol/partita, assist/partita, ecc., e il contributo medio atteso del clean sheet se applicabile) vengono passate al FantasyPointCalculatorService.
  + Il servizio calcola la FantaMedia per partita attesa, basata sulle regole di punteggio della lega dell'utente.
* **Calcolo Totali Stagionali Proiettati:** (Implementato)
  + La FantaMedia proiettata per partita viene moltiplicata per le presenze\_attese per ottenere i fantapunti totali stagionali.
  + Le singole statistiche medie per partita (gol, assist, ecc.) vengono moltiplicate per le presenze\_attese per ottenere i totali stagionali.
* **Sviluppi Futuri Già Identificati per la Logica di Proiezione:**
  + Affinamento proiezione clean sheet (più data-driven).
  + Modulazione più raffinata dell'impatto del tier (es. su gol subiti, già parzialmente introdotto).
  + **Proiezione Rigoristi:** Analisi e proiezione della "rigoristicità" basata su dati storici (Rc, R+) e PlayerTacticalNotes.
  + Considerazione della regressione verso la media.
  + Aggiustamento per **Ruolo Tattico Specifico** (da detailed\_position API / PlayerTacticalNotes).

### **4.3. Output delle Proiezioni**

* **FantaMedia Proiettata per Partita (fanta\_media\_proj\_per\_game).**
* **Fantapunti Totali Stagionali Proiettati (total\_fantasy\_points\_proj).**
* Media Voto Proiettata per Partita (mv\_proj\_per\_game).
* Presenze Proiettate (presenze\_proj).
* Proiezioni delle singole statistiche totali per la stagione (gol, assist, ammonizioni, ecc. in seasonal\_totals\_proj).
* Un "Breakout Score" (opzionale, futuro).

## **5. Modulo di Valutazione e Identificazione Talenti (Da Implementare)**

[Come da versione 1.1 - Sezioni 5.1, 5.2]

## **6. Modulo Strategia d'Asta (Da Implementare)**

[Come da versione 1.1 - Sezioni 6.1 a 6.6]

## **7. Struttura Applicativa Laravel (Alto Livello)**

### **7.1. Modelli Principali (Eloquent)**

* **Player**: Come descritto in Sezione 3.3.
* **Team**: Come descritto in Sezione 3.3.
* **HistoricalPlayerStat**: (Rinominato da HistoricalSeasonStat) Come descritto in Sezione 3.3.
* **UserLeagueProfile**: Come descritto in Sezione 3.3.
* **ImportLog**: Come descritto in Sezione 3.3.
* AuctionPlan: (Da implementare)
* AuctionPlanTarget: (Da implementare)
* PlayerTacticalNote: (Da implementare)
* TeamHistoricalStandings: (Da Implementare)

### **7.2. Servizi Chiave (Implementati e Da Implementare)**

* **Servizi di Importazione (Logica nei Controller/Import Classes):**
  + RosterImportService: Per roster e quotazioni (attualmente in RosterImportController e TuttiSheetImport).
  + HistoricalStatsImportService: Per statistiche storiche (attualmente in HistoricalStatsImportController e TuttiHistoricalStatsImport).
* **DataEnrichmentService**: (Implementato Parzialmente)
  + Recupera dati anagrafici (data di nascita, posizione dettagliata, ID API) da Football-Data.org.
  + Logica di matching nomi giocatori e squadre.
  + Futuro: Recupero dati storici squadre, statistiche avanzate.
* **ProjectionEngineService**: (Implementazione Iniziale Avanzata)
  + Cuore del sistema, implementa la logica di proiezione (storico ponderato, età, tier squadra).
  + Calcola medie per partita e totali stagionali.
* **FantasyPointCalculatorService**: (Implementato)
  + Converte statistiche proiettate (per partita) in FantaMedia per partita basata sulle regole di lega.
* AuctionValueCalculatorService: (Da implementare)
* PlayerTieringService: (Da implementare)
* TeamTieringService: (Da implementare) Per calcolare dinamicamente il tier delle squadre.
* AuctionStrategyBuilderService: (Da implementare)
* PairAnalyzerService: (Da implementare)
* TeamConcentrationService: (Da implementare)

### **7.3. Controller e Viste**

* **Controller Implementati:**
  + RosterImportController: Gestisce upload e importazione roster.
  + HistoricalStatsImportController: Gestisce upload e importazione statistiche storiche.
  + UserLeagueProfileController: Gestisce la creazione/modifica del profilo lega.
* **Viste Implementate:**
  + uploads/roster.blade.php
  + uploads/historical\_stats.blade.php
  + league/profile\_edit.blade.php
  + layouts/app.blade.php (layout base)
* **Futuri:** Controller e viste per visualizzazione proiezioni, costruzione piano d'asta, ecc.

### **7.4. Processi in Background (Jobs) (Da Implementare Come Asincroni)**

* ImportFantacalcioRosterJob: Attualmente la logica è sincrona nel controller. Da convertire in Job per importazioni grandi.
* ImportHistoricalStatsJob: Attualmente la logica è sincrona. Da convertire in Job.
* EnrichPlayerDataJob: (Attualmente la logica è nel comando Artisan players:enrich-data) Per arricchire i dati dei giocatori tramite API in background, gestendo rate limiting e retry.
* FetchTeamHistoricalStandingsJob: Per recuperare i dati storici delle squadre.
* RecalculateProjectionsJob: Per aggiornare le proiezioni se i dati di base cambiano.

### **7.5. Comandi Artisan Personalizzati**

* **php artisan test:projection {fanta\_platform\_id}**
  + **Scopo:** Testa il ProjectionEngineService per un giocatore specifico, utilizzando il suo fanta\_platform\_id.
  + **Utilizzo:** php artisan test:projection 4220 (per testare Zambo Anguissa).
  + **Logica:** Carica il giocatore, recupera/crea un profilo lega di default, esegue ProjectionEngineService->generatePlayerProjection() e mostra l'output JSON delle proiezioni. Utile per debug e calibrazione della logica di proiezione.
* **php artisan players:enrich-data {--player\_id=all} {--player\_name=} {--delay=6}**
  + **Scopo:** Arricchisce i dati dei giocatori (data di nascita, posizione dettagliata, ID API) interrogando l'API esterna Football-Data.org.
  + **Utilizzo:**
    - php artisan players:enrich-data: Processa tutti i giocatori nel database che hanno date\_of\_birth O detailed\_position O api\_football\_data\_id a NULL.
    - php artisan players:enrich-data --player\_name="NomeParziale": Processa i giocatori il cui nome contiene "NomeParziale".
    - php artisan players:enrich-data --player\_id=ID\_DATABASE: Processa il giocatore con l'ID specificato dalla tabella players.
    - php artisan players:enrich-data --delay=10: Imposta un ritardo di 10 secondi tra le chiamate API per giocatore (per gestire rate limit).
  + **Logica:** Utilizza il DataEnrichmentService per trovare il match del giocatore/squadra sull'API, recuperare i dati e aggiornare il database locale.

## **8. Considerazioni Aggiuntive e Sviluppi Futuri**

[Come da versione 1.1, integrando le novità]

* **Qualità dei Dati:** Fondamentale. L'accuratezza dell'arricchimento API (matching nomi) e dei dati storici importati è cruciale.
* **Manutenzione:** Aggiornamento annuale rose, dati qualitativi. Gestione promosse/retrocesse.
* **Tiering Squadre Dinamico:** Implementare il recupero dati storici squadre e il TeamTieringService per calcolare i tier in modo più oggettivo.
* **Usabilità (UX/UI):** Interfaccia intuitiva.
* **Gestione Rate Limit API:** Rendere più robusta la gestione degli errori 429 con retry e backoff nel DataEnrichmentService.
* **Machine Learning:** Potenziamento futuro per proiezioni.
* **Supporto Modalità Mantra:** Estensione.
* **Community Features:** Condivisione dati asta.

Spero che questa versione aggiornata e unificata ti sia d'aiuto! È un documento vivo, quindi potremo continuare ad aggiornarlo man mano che il progetto cresce.