Hai perfettamente ragione! Gli esempi pratici rendono la documentazione molto più chiara e utile.

Integriamo subito degli esempi pratici nella documentazione dei file di configurazione.

## **Documentazione Aggiornata (con Esempi Pratici):**

### **Documentazione: config/player\_age\_curves.php**

**File:** config/player\_age\_curves.php

1. Scopo Generale  
   Questo file di configurazione definisce come l'età di un giocatore influenza le proiezioni delle sue performance. Specifica le diverse fasi della carriera di un calciatore (sviluppo, picco, mantenimento, declino) per differenti ruoli o tipologie di ruolo e i modificatori quantitativi da applicare alle statistiche di base.
2. Utilizzo Principale  
   Il file player\_age\_curves.php è utilizzato principalmente dal ProjectionEngineService per:
   * Calcolare un ageModifier generale per un giocatore in base alla sua età e al suo ruolo specifico per la curva di età.
   * Applicare questo modificatore (o sue variazioni) a diverse statistiche proiettate, come la Media Voto (MV), i gol, gli assist, la probabilità di clean sheet e le presenze attese.
   * L'obiettivo è rendere le proiezioni più realistiche, tenendo conto che le prestazioni dei giocatori tendono a seguire una curva durante la loro carriera.
3. **Struttura e Configurazione Dettagliata**  
   Il file restituisce un array associativo con le seguenti chiavi principali:
   * descrizione: Una breve descrizione testuale dello scopo del file.
   * disclaimer: Un avviso che sottolinea come le curve siano generalizzazioni.
   * dati\_ruoli: Questa è la sezione chiave. È un array associativo dove ogni chiave rappresenta un raggruppamento di ruolo specifico per la curva di età (es. 'P' per portieri, 'D\_CENTRALE', 'D\_ESTERNO', 'C' per centrocampisti, 'A' per attaccanti).  
     Ogni ruolo ha la seguente struttura:
     + fasi\_carriera: Un array che definisce le età di transizione tra le fasi:
       - sviluppo\_fino\_a: Età fino alla quale il giocatore è considerato in fase di sviluppo.
       - picco\_inizio: Età in cui inizia la fase di picco.
       - picco\_fine: Età in cui termina la fase di picco.
       - mantenimento\_fino\_a: Età fino alla quale il giocatore potrebbe mantenere un buon livello prima di un declino più marcato (opzionale, la logica attuale si basa principalmente su picco e declino).
       - declino\_da: Età da cui inizia la fase di declino più evidente.
     + note\_picco\_declino: Note testuali descrittive sulla tipica evoluzione del ruolo.
     + growth\_factor: Fattore numerico (es. 0.015) che determina l'incremento percentuale annuo delle prestazioni durante la fase di sviluppo, per ogni anno prima del picco\_inizio.
     + decline\_factor: Fattore numerico (es. 0.025) che determina il decremento percentuale annuo delle prestazioni durante la fase di declino, per ogni anno dopo il picco\_fine.
     + young\_cap: Limite massimo del modificatore età per i giocatori giovani (es. 1.12 significa che un giovane non può avere un moltiplicatore superiore al +12% rispetto al suo picco teorico dovuto solo all'età).
     + old\_cap: Limite minimo del modificatore età per i giocatori anziani (es. 0.78 significa che un giocatore anziano non può avere un moltiplicatore inferiore al -22% rispetto al suo picco teorico dovuto solo all'età).
   * age\_modifier\_params: Un array che definisce come l' ageModifier generale (calcolato dalle fasi di carriera) viene specificamente applicato a diverse categorie di statistiche o aspetti della proiezione.
     + mv\_effect\_ratio: (es. 0.4) Specifica quale frazione dell'effetto dell'età (differenza da 1.0 del ageModifier) si applica alla Media Voto.
     + cs\_age\_effect\_ratio: (es. 0.2) Simile a mv\_effect\_ratio, ma per l'impatto sulla probabilità di Clean Sheet.
     + presenze\_growth\_effect\_ratio: (es. 0.4) Specifica quale frazione dell'effetto *positivo* dell'età (giocatore giovane in crescita) si applica alle presenze attese.
     + presenze\_decline\_effect\_ratio: (es. 1.1) Specifica quale frazione (o amplificazione, se >1) dell'effetto *negativo* dell'età (giocatore anziano in declino) si applica alle presenze attese.
     + presenze\_growth\_cap: (es. 1.12) Limite massimo all'incremento delle presenze dovuto al fattore età per giocatori giovani.
     + presenze\_decline\_cap: (es. 0.65) Limite minimo alla riduzione delle presenze dovuto al fattore età per giocatori anziani.

**Come Configurare/Modificare (con Esempi):**

* + **Ruoli e Fasi Carriera:**
    - **Esempio:** Supponiamo di voler definire che gli attaccanti (A) abbiano un picco più breve ma intenso.  
      PHP  
      'A' => [  
       'fasi\_carriera' => [  
       'sviluppo\_fino\_a' => 22, // Sviluppo più breve  
       'picco\_inizio' => 23, // Picco inizia prima  
       'picco\_fine' => 28, // Picco finisce prima  
       'declino\_da' => 29, // Declino inizia prima  
       ],  
       'growth\_factor' => 0.030, // Crescita più rapida  
       'decline\_factor' => 0.040, // Declino più rapido  
       'young\_cap' => 1.25,  
       'old\_cap' => 0.60,  
       // ...  
      ],
  + **Fattori e Cap:**
    - **Esempio:** Per i portieri (P), se si ritiene che l'esperienza compensi maggiormente il calo fisico, si potrebbe ridurre il decline\_factor e aumentare l'old\_cap.  
      PHP  
      'P' => [  
       // ... fasi\_carriera ...  
       'decline\_factor' => 0.010, // Declino più lento  
       'old\_cap' => 0.85, // Mantengono un livello più alto più a lungo  
       // ...  
      ],
  + **Parametri di Applicazione (age\_modifier\_params):**
    - **Esempio:** Se vuoi che l'età influenzi di più la Media Voto (MV) ma meno le presenze per i giocatori in crescita:  
      PHP  
      'age\_modifier\_params' => [  
       'mv\_effect\_ratio' => 0.6, // 60% dell'effetto età sulla MV  
       'presenze\_growth\_effect\_ratio' => 0.3, // 30% dell'effetto crescita età sulle presenze  
       // ... altri parametri ...  
      ]
    - Un ageModifier di 1.10 (giocatore giovane in crescita) con mv\_effect\_ratio: 0.6 porterebbe a un moltiplicatore MV di 1 + (0.10 \* 0.6) = 1.06 (+6%).
    - Con presenze\_growth\_effect\_ratio: 0.3, lo stesso ageModifier porterebbe a un moltiplicatore presenze di 1 + (0.10 \* 0.3) = 1.03 (+3%), limitato poi da presenze\_growth\_cap.

### **Documentazione: config/projection\_settings.php**

**File:** config/projection\_settings.php

1. Scopo Generale  
   Questo file centrale contiene una vasta gamma di parametri che governano il funzionamento del ProjectionEngineService. Definisce come vengono recuperati e pesati i dati storici, come vengono applicati gli aggiustamenti (diversi da quelli puramente legati all'età), come viene gestita la logica dei rigoristi, i valori di default e fallback, e la configurazione dei campi di output delle proiezioni.
2. Utilizzo Principale  
   Il file projection\_settings.php è il cuore della configurazione del ProjectionEngineService. Ogni aspetto del calcolo delle proiezioni, dalla selezione dei dati storici alla definizione delle statistiche finali, fa riferimento a parametri definiti in questo file. Permette una calibrazione fine del modello predittivo senza modificare direttamente il codice del servizio.
3. **Struttura e Configurazione Dettagliata (con Esempi dove rilevante)**  
   Il file restituisce un array associativo con numerose chiavi:
   * **Parametri Rigoristi:**
     + penalty\_taker\_lookback\_seasons: (es. 2) Numero di stagioni passate da analizzare per identificare un rigorista.
     + min\_penalties\_taken\_threshold: (es. 3) Numero minimo di rigori calciati nel periodo di lookback perché un giocatore sia considerato un potenziale rigorista primario.
     + league\_avg\_penalties\_awarded: (es. 0.20) Media di rigori assegnati per squadra a partita nel campionato.
     + penalty\_taker\_share: (es. 0.85) Quota percentuale dei rigori di squadra che si stima il rigorista designato calci.
     + default\_penalty\_conversion\_rate: (es. 0.75) Tasso di conversione dei rigori usato come fallback.
     + min\_penalties\_taken\_for\_reliable\_conversion\_rate: (es. 5) Numero minimo di rigori calciati storicamente dal giocatore perché il suo tasso di conversione personale sia considerato affidabile.
   * **Gestione Dati Storici e Medie:**
     + lookback\_seasons: (es. 4) Numero di stagioni storiche da considerare per calcolare le medie ponderate delle statistiche generali.
     + season\_decay\_factor: (es. 0.75) Fattore di decadimento applicato ai pesi delle stagioni più vecchie. Se per 3 stagioni, i pesi potrebbero essere (1), (1 \* 0.75), (1 \* 0.75 \* 0.75), poi normalizzati.
     + fields\_to\_project: Array di stringhe che elenca i nomi delle colonne delle statistiche da HistoricalPlayerStat che devono essere proiettate (es. 'avg\_rating', 'goals\_scored', 'assists').
     + min\_games\_for\_reliable\_avg\_rating: (es. 10) Numero minimo di partite giocate in una stagione perché la Media Voto di quella stagione sia considerata affidabile nel calcolo della media ponderata.
   * **Impatto del Tier Squadra:**
     + default\_team\_tier: (es. 3) Tier di squadra da usare come fallback se una squadra non ha un tier definito.
     + team\_tier\_multipliers\_offensive: Array associativo (Tier => Moltiplicatore) per modulare le statistiche offensive.
       - **Esempio:** '1' => 1.20 (un giocatore offensivo in una squadra di Tier 1 vedrà le sue stats offensive di base aumentate del 20%).
     + team\_tier\_multipliers\_defensive: Simile, ma per statistiche difensive.
       - **Esempio:** '1' => 0.80 (un portiere/difensore in una squadra di Tier 1 vedrà i suoi gol subiti di base ridotti del 20%, o la sua probabilità di CS aumentata).
     + team\_tier\_presence\_factor: Array associativo (Tier => Moltiplicatore) per modulare le presenze attese.
       - **Esempio:** '5' => 0.95 (un giocatore in una squadra di Tier 5 potrebbe vedere le sue presenze leggermente ridotte rispetto alla sua media storica a causa della debolezza della squadra).
     + offensive\_stats\_fields: Campi influenzati dal team\_tier\_multipliers\_offensive.
     + defensive\_stats\_fields\_goalkeeper: Campi (per portieri) influenzati dal team\_tier\_multipliers\_defensive.
   * **Clean Sheet:**
     + league\_average\_clean\_sheet\_rate\_per\_game: (es. 0.28) Tasso medio di clean sheet per partita osservato nel campionato.
     + clean\_sheet\_probabilities\_by\_tier: (Non presente nel file caricato, ma dedotto e implementato nel servizio) Array (Tier => Probabilità CS).
       - **Esempio (da aggiungere):**  
         PHP  
         'clean\_sheet\_probabilities\_by\_tier' => [  
          1 => 0.40, // Squadra Top Tier ha 40% prob. base di CS  
          2 => 0.33,  
          3 => 0.28, // Media lega  
          4 => 0.22,  
          5 => 0.18, // Squadra debole ha 18% prob. base di CS  
         ],
     + max\_clean\_sheet\_probability: (Non presente nel file caricato) Limite massimo per la probabilità di clean sheet proiettata (es. 0.8).
   * **Valori di Default e Fallback:**
     + default\_player\_age: (es. 25) Età da usare se date\_of\_birth non è disponibile.
     + fallback\_mv\_if\_no\_history: (es. 5.5) Media Voto di default se un giocatore non ha storico.
     + fallback\_fm\_if\_no\_history: (es. 5.5) FantaMedia di default se un giocatore non ha storico.
     + fallback\_gp\_if\_no\_history: (es. 0) Presenze di default se un giocatore non ha storico.
     + min\_projected\_presences / max\_projected\_presences: (Non presenti nel file caricato, ma usati nel codice, es. 5 / 38) Limiti per le presenze proiettate.
   * **Configurazione Output Proiezioni:**
     + fields\_to\_project\_output: Array associativo che mappa le chiavi dell'output finale (es. goals\_scored\_proj) a come calcolarle.
       - **Esempio:**  
         PHP  
         'goals\_scored\_proj' => ['type' => 'sum', 'source\_per\_game' => 'goals\_scored', 'default\_value' => 0],  
         Significa che output['goals\_scored\_proj'] sarà proiezione\_goals\_scored\_per\_partita \* presenze\_proiettate.

**Come Configurare/Modificare (con Esempi):**

* + **Logica Rigoristi:**
    - Se nel tuo campionato i rigori sono rari, potresti abbassare league\_avg\_penalties\_awarded.
    - Se hai osservato che il rigorista designato calcia quasi tutti i rigori, aumenta penalty\_taker\_share (es. a 0.90).
  + **Ponderazione Storico:**
    - Se vuoi dare molto più peso alla stagione più recente, aumenta season\_decay\_factor verso 1.0 per le stagioni più vecchie, oppure implementa una logica di pesi espliciti (es. [0.5, 0.3, 0.2] per 3 stagioni). L'attuale season\_decay\_factor nel file caricato è 0.75, ma la logica del servizio calculateWeightedAverageStats usa un sistema di pesi decrescenti (N, N-1, ..., 1) normalizzati. Potrebbe essere necessario allineare il nome della chiave alla logica effettiva o viceversa.
  + **Impatto Tier:**
    - Se le squadre di Tier 1 nel tuo campionato dominano segnando il 30% in più della media, imposta '1' => 1.30 in team\_tier\_multipliers\_offensive.
    - Se le squadre di Tier 5 subiscono il 25% in più di gol, imposta '5' => 1.25 in team\_tier\_multipliers\_defensive.
  + **Valori di Default per Giocatori Nuovi/Sconosciuti:**
    - Se un nuovo attaccante arriva in Serie A senza storico, il sistema userà fallback\_mv\_if\_no\_history, fallback\_fm\_if\_no\_history e i default da getDefaultStatsPerGameForRole. Assicurati che questi valori siano una stima prudente ma ragionevole.

Spero che questa documentazione aggiornata con esempi sia più chiara e ti aiuti a calibrare al meglio il tuo motore di proiezioni!