ChatGPT

Fantasy Football Auction Value Model Implementation Guide

Overview

This guide outlines a step-by-step plan to build a **fantasy football auction value model** tailored for a 12-team league with flexible settings. The model will incorporate custom scoring rules, multi-year data, and various analytical inputs (projections, advanced stats, betting lines, etc.) to produce dollar values for players. We use a **Value Over Replacement Player (VORP)** framework as the foundation, while noting alternative approaches and adjustments. The plan is structured in modular steps so different components can be developed in parallel. Key formulas and potential pitfalls are highlighted throughout.

Step 1: Ingest League Settings (Scoring Rules & Budget)

Task: Begin by capturing the league's roster requirements, scoring system, and draft budget from the provided rules (e.g. the CBS Sports league PDF). These settings will drive how player statistics translate into fantasy points and how values scale with budget.

- **Parse Roster and Scoring Details:** Read all scoring categories, point values, and bonuses. For example, if passing touchdowns are 6 points with bonus points for long TDs [1], or 1 point per 25 passing yards with bonus points at certain yardage thresholds [2], the model must incorporate each rule. Construct formulas to convert raw stats into fantasy points for each position. *Example:*
- *Passing points* = (PassYards ÷ 25 * 1) + (PassTD * 6) + (LongPassTD Bonus, if applicable) – (Interceptions * 3) [1].
- *Rushing points* = (RushYards ÷ 10 * 1) + (RushTD * 6) + (LongRushTD Bonus) [3] [4], etc.

- Include all nuanced settings (receptions scoring, yardage bonuses, two-point conversions, fumbles, defensive scoring, etc.) to accurately calculate projected fantasy points under the custom system. **Pitfall:** Missing a scoring rule or bonus can skew player projections significantly (e.g. failing to account for a **points-per-reception (PPR)** or big-play bonus will undervalue receivers). Always double-check that the model's scoring outputs match a few manual examples from the rules.

- **Budget Inputs:** Likewise, input the league's draft budget per team (e.g. $200 is a typical default [5], but this should be configurable). The total auction pool is `Budget * Number of teams` (e.g. $200 * 12 = $2400). All computed player values will later be scaled to this pool. The model should be flexible to adjust if the league changes the budget (for instance, some leagues use $100 or $250 budgets). **Pitfall:** Ensure that value calculations linearly adjust with budget – if the budget doubles, player dollar values should roughly double as well, to maintain consistent relative pricing.

- **Modular Output:** This step can be encapsulated in a **League Config module** that outputs key parameters: number of teams, roster spots by position, scoring coefficients for each stat, and total budget. Downstream modules (projection processing, value calculation) will use these parameters.

By isolating league settings, the model can easily handle different leagues' rules without altering core logic.

Step 2: Compile Weekly Player Projections (2024 & 2025)

Task: Gather per-week player projection data for the 2024 and 2025 seasons, with emphasis on 2025. Using weekly granularity allows accounting for bye weeks and matchup-based variations, and summing to season totals for value calculations.

- **Data Collection:** Collect projections for all relevant players (QB, RB, WR, TE, K, DST) for each week of 2025. This could come from fantasy projections APIs, expert consensus rankings, or in-house models. Also obtain 2024 weekly projections (and actual outcomes if available) for backtesting. Each player's *season projected points* under the league's scoring is the sum of weekly projections (taking into account the scoring rules from Step 1). Ensure bye weeks are included as zero-point weeks so that total projections reflect missed games.

- **Aggregation and Emphasis on 2025:** Focus the model on the 2025 projections for the actual draft values. The 2024 data will be used for validation and tuning. If multiple projection sources are available, consider combining them (e.g. averaging or a weighted average based on past accuracy) to improve reliability [6]. For example, you might average projections from several experts, weighting each by their historical accuracy, to get a consensus forecast. This mitigates biases from any single source.

- **Example Calculation:** For a given player, convert projected stats to fantasy points using the league scoring. If a running back is projected for 1,200 rushing yards and 10 TDs over 17 games, and the league awards 0.1 points per yard and 6 per TD, his base points = 1200*0.1 + 10*6 = 120 + 60 = 180, plus any bonuses (e.g. three 100-yard game bonuses would add 9 points if 3-point each). Do this for all players.

- **Bye Week Handling:** Because weekly projections are used, a player's total already excludes their bye week (no points that week). This inherently downgrades season totals for players with byes (which is everyone in a single-bye NFL season). We will later account for replacing those bye week points from a bench player (Step 7), but at this stage, just record the raw totals. **Pitfall:** Watch out for players projected to miss multiple weeks (injuries or suspensions) – their totals will be lower, but teams can substitute backups for those weeks. We might mark these players and later consider adding replacement-level points for missed games (see Step 7) [7].

- **Output:** This step produces a dataset of all players with their projected fantasy points for 2025 (and a similar dataset for 2024 for backtesting). This can be done by a **Projections module** which reads in raw projections, applies the scoring formula from Step 1, and outputs fantasy point totals per player. This module is independent and can be developed/tested separately (e.g. verifying a known player's output against a manual calculation).

Step 3: Integrate Advanced Stats (2023–2024) for Projection Refinement

Task: Enhance the baseline projections using advanced metrics from the 2023 and 2024 seasons. These player- and team-level analytics (e.g. target share, air yards, accuracy, play pace, pressure rates) provide context that pure projections might miss, helping to fine-tune each player's expected performance.

- **Player-Level Metrics (RB/WR):** Incorporate stats like **target share, air yards, yards after catch, broken tackle rate,** etc., from recent years to adjust projections. For example, a wide receiver's **target market share** in 2024 is a strong indicator of his role in the offense and is relatively stable year-to-year [8]. If a WR had a 25% target share last year and his team's offense is largely unchanged, it's reasonable to project a similar share in 2025. Use 2024 team pass attempt projections to convert that share into raw targets. Likewise, use **air yards** data to gauge a receiver's depth of targets – a player with high air yards but low actual catch totals might be undervalued by raw projections, indicating upside if efficiency improves. Adjust receiving yardage projections upward if a player underperformed his air yards (regression to the mean could yield a jump in actual yards). For running backs, consider metrics like **yards after contact, target share out of backfield,** or **goal-line carry share** to tweak their yardage and TD projections (e.g. a RB with an unusually low TD total despite many goal-line carries in 2024 might be due for a TD increase, so you could project slightly more TDs than the baseline).

- **Team-Level Metrics:** Use 2024 team statistics such as **quarterback accuracy, offensive play-calling tendencies, and offensive line protection** to refine player outputs. For instance, if a team's QB had a very high **catchable pass rate** or accuracy rating in 2024, that benefits his receivers – they are likely to convert targets into receptions at a higher rate [9]. The model can reflect this by boosting the catch percentage or yards per target for that team's receivers in 2025 projections. Conversely, if a team allowed high pressure on the QB (poor offensive line), you might temper the QB's projected efficiency (lower yards per attempt, or slightly fewer big plays) which trickles down to WR/RB receiving stats. Also account for **play pace and run/pass split** ("play type" tendencies): a fast-paced offense with high play counts or a high pass percentage will produce more volume for fantasy players. If a new coaching staff or dramatic 2024 trend suggests a shift (e.g. team moving from run-heavy to pass-heavy), adjust player volume projections accordingly.

- **Integration Method:** Develop a **Projection Adjustment module** that takes the initial projections (from Step 2) and applies these advanced stat factors. This could be as simple as applying multipliers or offsets (for example, increase a WR's projected receptions by 5% if his 2024 target share was very high and likely sustainable), or as complex as building a regression model that predicts fantasy points using these metrics as features. In either case, clearly document the adjustments: e.g., *"Increased Player X's projected yards by +50 because his air yards in 2024 suggest he was underutilized"*. **Pitfall:** Ensure that any adjustments are grounded in data – not all advanced stats are **predictive** of future fantasy success. Some are merely descriptive of past performance [10]. Focus on metrics with proven year-to-year correlation (target share, carry share, accuracy, etc.) and be cautious with those that are more volatile. The goal is to add context, not to overfit. Test the impact by checking if these adjustments would have improved projections for 2024 actual results (as part of backtesting).

- **Outcome:** Updated 2025 projections for each player that reflect not just baseline expectations but also each player's situation and efficiency. This yields more nuanced projected fantasy points. This step is parallelizable; one team member could handle incorporating receiver metrics while another handles quarterback and team-level factors, for example.

Step 4: Incorporate Sportsbook Player Prop Data (2025)

Task: Integrate 2025 player prop betting lines (e.g. from sportsbooks' over/under totals for yards, TDs, etc.) into the projections. The betting markets provide a real-time consensus expectation for player performance, which can serve as an additional, often sharp, data point.

- **Collect Prop Lines:** Gather available **season-long props** for key statistics (passing yards, rushing yards, receiving yards, touchdowns, etc.) for as many relevant players as possible. For example, if a top running back has a Vegas line of 1150.5 rushing yards for 2025, and our current projection was 1050, we might reconsider and adjust upward. Sportsbooks continuously update these lines based on news and betting, so they encapsulate up-to-date information and wisdom of crowds. Research has shown that incorporating Vegas lines can improve projection accuracy – one study found using prop-based projections was ~3.5% more predictive than a standard source's projections [11], which can translate to a several-point weekly edge.

- **Merge with Projections:** For each player, if a prop line is available for a stat, adjust the player's projection toward that line. A straightforward approach is to **replace** the model's forecast for that stat with the prop number (assuming the prop is a fair median expectation). Another approach is to take a weighted average (e.g. blend 50% model projection, 50% prop) to mitigate small differences. The closer to the season, the more weight you might give the prop lines (since they'll reflect any preseason developments). An actual implementation example: *"Start with Sleeper's projections and as soon as any stat is available as a prop bet, replace it with that"* [12] – as the season nears, more of the projection is driven by prop data, improving accuracy. Ensure consistency: if you adjust a QB's passing yards to match a prop, also adjust his receivers' totals proportionally (so as not to double-count yards).

- **Account for Missing Props:** Not all players will have prop lines (often only prominent players have season totals, and injured/suspended or uncertain-role players may be omitted) [13]. The model should detect when a stat is unavailable and simply retain the original projection for those cases. **Pitfall:** Don't drop a player entirely or give him zero just because there's no betting line – lack of a line doesn't imply lack of production, it might just mean the player is off the bookmakers' radar (e.g. rookies or backup RBs). Also note that **season-long props tend to be conservative medians and may undervalue upside** [14]. A player with high injury risk or variability might have a modest line that represents a median outcome; our fantasy model might still project a higher ceiling if everything goes right. Be cautious about blindly lowering a high-upside player's projection because of a low Vegas line – you might incorporate the information by, say, flagging the risk rather than fully capping his projection.

- **Finalize Adjusted Projections:** After blending in the prop data, you have a set of 2025 projections that combine multiple sources: traditional projections, advanced stat-informed tweaks, and market-derived expectations. These should be more robust than any single source. Keep a log of which players were heavily adjusted due to props, as this indicates the market sees them differently than initial projections (a point of discussion for fantasy drafters). This step can be handled by a **Betting Data integration module** running in parallel to others, as it just reads props and updates the projection list.

Step 5: Determine Replacement Levels and Calculate VORP

Task: Using the projections, establish the **replacement player** baseline for each position and compute each player's *Value Over Replacement Player (VORP)*. This measures how much better a player is than what's freely available, forming the core of the value model.

- **Define Positional Baselines:** Given the league roster settings, determine what constitutes a "replacement-level" player at each position. In a 12-team league with starting requirements: 1 QB, 2 RB, 2 WR, 1 TE, 1 FLEX (RB/WR/TE), 1 K, 1 DST (assuming the flex can be 0–1 starters from RB/WR/TE) [15], a reasonable approach is:
- **QB:** 12 starters, so the replacement baseline might be around the QB13 (the best quarterback not in a starting lineup each week). You could take the projected points of the QB13 as the baseline.
- **RB:** 24 must-start (2 per team). Additionally, 12 flex spots are shared among RB/WR/TE. Likely a chunk of those flexes will be RBs (especially if RBs project higher than comparable WR/TE). As an approximation, assume about half the flex positions (6 of them) are filled by RBs. That means roughly 24 + 6 = **30th RB** could be the replacement baseline. (Alternatively, a more precise method is to sort all RB/WR/TE by projected points to see which position dominates the flex – for example, if the top 12 flex-eligible players beyond the mandatory starters include 8 RBs and 4 WRs, then RB30 and WR28 would be the "last starter" at those positions). If unsure, one could use the **RB30's** points as baseline and later adjust if needed.
- **WR:** Similarly, 24 must-start plus the remainder of flex. If we assumed ~6 RBs in flex, then ~6 of the flex could be WRs. That suggests using around the **WR30** as baseline.
- **TE:** 12 must-start (one per team) and typically few TEs make it into the flex (unless a second TE projects very high). It's safe to use **TE13** (the first TE outside the starters) as the baseline.
- **K and DST:** 12 each are started. Often in fantasy, the "replacement" K/DST is essentially the one you can pick up week-to-week (K13 or DST13). We will handle DST separately in Step 8, but initially set DST13 as baseline for calculation.

These baselines correspond roughly to the **"last starter"** in each position if every team drafts optimally. In value-based drafting terms, this is **Value Over Last Starter (VOLS)** approach [16] – it prioritizes starters by assuming anything worse than the worst starter has no value. However, we may also consider a slightly lower baseline (e.g. a bench player or true waiver wire level) to not overly inflate starters (a pure **VORP** approach uses the best available free agent as baseline [17], which devotes more value to bench depth). There is a strategic trade-off here: using last-starter baseline will give higher values to top players (leaving little budget for bench), whereas using a waiver baseline will spread value more to bench players [18]. We will later adjust between these extremes (see Step 7).

- **Compute VORP:** For each player, calculate:
  VORP = Projected Fantasy Points – Baseline Fantasy Points for that position.
  For example, if the baseline for RB is 150 points (what the RB30 is projected to score) and our player is projected for 250, his VORP = 100 points. A baseline player himself would have VORP $\approx$ 0, and players below the baseline (e.g. RBs beyond the 30th) would have negative VORP (which essentially means they are freely replaceable and should be valued at the minimum $1). Use the baseline values

determined above: e.g. subtract QB13's points from each QB's projection to get QB VORPs, and so on for other positions.

- It might be wise to **average a few players around the cutoff** to smooth the baseline. For instance, instead of exactly RB30's points, take the average of RB29–RB32 as the baseline estimate, on the assumption that the replacement level is a small tier of similar players. This avoids odd spikes if one player at the cutoff has an anomalous projection.

- For the flex positions (RB/WR/TE), ensure consistency: the baseline we chose for each (RB30, WR30, TE13 etc.) should collectively account for roughly the top 12 flex spots. If, say, our model baseline led to a situation where the WR33 actually has more projected points than RB30 (meaning a WR might actually be a better flex than the baseline RB we assumed), we should revisit and possibly use a unified approach (like create a combined list of all RB/WR/TE projections sorted, find the 12th flex player's points). A more advanced method is the **"man-games" or BEER** approach [19], which uses historical data to estimate how many total games starters + bench will account for, effectively raising the baseline for positions with more injury/byes. We will touch on this in Step 7.

- **Corner Cases:** If the league has unusual settings (e.g. 2 QBs or superflex), adjust baseline accordingly (e.g. in a superflex, 12 teams could start up to 24 QBs, so baseline QB would be QB25). In our case, standard 1 QB means a lot of QB supply on waivers, which will make QB VORPs smaller relative to RB/WR (reflecting **positional scarcity** – QBs score the most points but are least scarce, while RBs/WRs are more scarce, which VORP captures by subtracting a higher baseline for QBs). **Pitfall:** Misidentifying the replacement level will mis-rank the players. If you set the bar too low (waiver level) you might undervalue elite starters (too much value given to bench), whereas too high (last starter) might overvalue top players and assume no bench value [18]. A balanced approach or at least awareness of these extremes is important (we will likely adjust between them).

- **Output:** At this point, each player has a VORP score (which can be positive, zero, or negative). This VORP list (especially for all positive VORPs) is the basis for dollar values. This computation could be done in a **Value Calculator module**, independent of the projection calculations. One team member can take the finalized projections and handle baseline selection + VORP calculation while others are still refining projections, as long as the interface (projected points per player) is agreed upon.

Step 6: Convert VORP to Auction Dollar Values (Initial Value Assignment)

Task: Distribute the total available auction budget across players in proportion to their VORP, then calibrate those values using historical 2024 auction results. This yields preliminary dollar values for each player, which we will fine-tune for special factors next.

- **Budget Distribution Formula:** Sum up the VORP of all draft-worthy players (generally those with positive VORP, since negative or zero VORP players are essentially replacement-level who should be worth $1 or $0). Let `S =` sum of VORPs for all players *expected to be drafted*. (We might consider all players with VORP > 0, and also ensure we're including roughly the number of players that will be drafted. For a 12-team league with 16 roster spots, about 192 players will be drafted; including some negative VORP players in the last rounds for bench is fine, they'll just get $1 values.) Now the total league budget is `B = 12 teams * $200 = $2400` (assuming $200 each). We allocate money such that **$\$ per VORP point = B / S$**. For each player, **Initial Dollar Value = (VORP_player) * (B / S)**. This basic proportional method ensures the sum of all player values equals the total money available (so the model in aggregate "spends" all the money). It inherently gives larger shares to those with higher VORP (i.e. far above replacement).

- *Example:* Suppose in our projections, the sum of positive VORPs for all players comes out to 1200 points above replacement. With $2400 to allocate, that's $2 per VORP point. A player with 100 VORP points would get $200 value. A player with 10 VORP would get $20, and someone at 1 VORP would get $2. Players at or below 0 VORP are essentially $1 (the minimum bid). We'd likely floor everything at $1 for any player who comes out below that.

- Ensure to round or handle fractional dollars in a reasonable way (usually nearest dollar, and always at least $1 if a player is draftable).

- **Calibrate with 2024 Actual Auction Values:** Now incorporate the *historical realized prices from 2024* as a sanity check and calibration factor. Compare the model's initial values (had we applied it to 2024 projections) to what players actually went for in 2024 auctions (if available from the league or a similar source). Identify patterns or biases: Did the model systematically value certain positions higher or lower than the market did? For example, maybe the model says top QBs are worth $40, but last year no QB went above $25 – indicating that in practice, managers discount QBs (likely due to plentiful supply). Or the model might undervalue elite RBs if our baseline was a bit low – if the top RB went for $70 last year and our model says $60, we might adjust to capture that people pay a premium for top RBs. Use the 2024 data to compute adjustment factors:

- Calculate the total spend by position in actual 2024 drafts vs. the model. E.g., if in reality about 45% of money went to RBs, 35% to WRs, 10% to QBs, etc., but the model is allocating 38% RB, 40% WR, 15% QB, that indicates a redistribution is needed to mirror how fantasy managers actually value positions (which can be influenced by risk and position scarcity perceptions). We can apply a correction multiplier to each position's values so that the positional spend matches a target (which could be the historical distribution or an "optimal" one you believe in). This is similar to ensuring the model doesn't deviate wildly from expectations.

- On a per-player level, one could do a regression or ratio of actual price vs model VORP for 2024 to derive a conversion. However, because each year and league can differ, it's better to adjust in broad strokes (by tiers or positions) rather than chase every individual price.

- **Incorporate Premiums/Discounts:** One observed strategy (per analysis by fantasy experts) is to spend more on guaranteed starters and less on bench players. Our initial VORP method treats a point of VORP the same for any player, but in practice, **teams often pay a premium for top-tier players** and savvy managers reserve less budget for marginal improvements on the bench [20]. We can explicitly model this by tweaking the dollar allocation curve: for example, increase the cost of the top X ranked players by some percentage and decrease the cost of lower-tier players [20]. A study by the Harvard Sports Analytics group found that winning teams tend to overspend on their starting lineup and underspend on bench depth [21]. In our model, we can apply, say, a +10% price multiplier to players projected as starters (the highest ranks at each position) and a –10% to late-round bench-caliber players [20]. This effectively reallocates a bit of budget towards the elite players (who carry more week-to-week value) and away from the fringe players (who might only play during byes or if injuries occur). The result is closer to how actual auction drafts behave and aligns with the idea that bench points are less valuable than starting points.

- **Example of Tiered Adjustment:** Define “Tier 1” as players in the top ~20 overall (who will be starters for every team) – give them a slight bump in price. Define “Tier 3” as players beyond the top ~100 who are likely bench fodder – give them a slight reduction (many of these will end up $1 anyway). Ensure the total still sums to the budget after adjustments (we might need to re-normalize slightly). This mimics the market: e.g., if pure math said the #1 player is worth $50, you might bump to ~$55 knowing managers often pay a bit extra for the top stud; if it said a low-end WR3 is $3, you might drop to $1–2 because in practice those players go cheap as everyone’s saving money for bigger names.

- **Validate vs 2024:** Apply these adjusted rules to 2024 data as a backtest: feed in the 2024 projections, run the model, see if the output values more closely align with the actual 2024 auction results. If yes, the calibration is working. If not, iterate – maybe our baseline needed tweaking or the premium was too high/low. **Pitfall:** Be careful not to overfit the exact prices of last year – the goal is to capture general tendencies. Each year some players will exceed projections (e.g. a breakout rookie might have low projection but ended up costly due to hype). Our model should perhaps flag such cases (e.g. huge discrepancy between projection and price often means drafters are baking in “upside” or unknown potential). We can incorporate a note or adjust values slightly for players with high uncertainty (perhaps using variability or ceiling projections to add a few dollars to high-upside talents).

- **Outcome:** At this stage, the model outputs a list of players with initial dollar values that are grounded in projections and adjusted toward historical market behavior. This list should roughly allocate the full $2400 across all players, with top players carrying the largest price tags and bench players at $1–$2. The **Value Calculator module** can produce this list, and a **Calibration sub-module** can apply the heuristics above. These can be developed somewhat in parallel: one person can implement raw VORP-to-dollar conversion while another analyzes historical data and designs the adjustment factors, then they integrate the two. All citations for sources of heuristics and formulas (like the premium on starters or using AAV) should be documented for transparency [22].

Step 7: Adjust for Positional Scarcity, Bye Weeks, and Depth Considerations

Task: Refine the values by accounting for roster construction factors that go beyond raw projections: positional scarcity (supply and demand of positions), bye weeks and the need for bench depth, and overall team needs. These factors ensure the model’s recommendations are practical for drafting a competitive roster, not just an academic exercise.

- **Positional Scarcity & Tier Drops:** Review the output values to ensure they truly reflect scarcity at each position. Scarcity is inherently handled by VORP (since baseline subtraction makes fewer high-end options at a position more valuable), but double-check edge cases. For example, if the top tight end is far above the rest (a common scenario), the model will give him a huge VORP. That is correct, but we might still highlight that in the guide (the model could tag such players as “Tier 1 gap” players). Consider adding a **"drop-off" metric** – the point difference between a player and the next replacement at his position [23]. A large drop-off signifies a scarce position; the model might even add a dollar or two to such players because securing that positional advantage is worth a premium. Conversely, at deep positions (e.g. WR, if many are close in projection), the drop-off from one to the next is small, so the model's prices there will be flatter and that's fine. Ensure the values encourage a balanced roster spend consistent with scarcity: e.g., in a start-1 QB league, QB gets less budget share (lots of replacements) while RB/WR get more (limited reliable starters). If the calibration in Step 6 used actual spending, it likely already captured this (managers naturally pay for scarcity), but verify it aligns with intuition and league norms.

- **Depth and Bench Value:** We now explicitly factor in that bench players (depth) have diminishing value. In practice, only the points a bench player scores **when in your lineup** matter (typically during a starter's bye or injury). One way to account for this is the **"man-games" or BEER** baseline method [24]: instead of assuming each team only needs X players (just the starters) all season, assume each starting roster spot will require perhaps 1.2 or 1.3 players over the course of the season (to cover bye weeks and some injuries). This effectively raises the baseline for positions slightly, because replacement-level production will come from a combination of bench players over the year. For the model, a simpler approach is to **reserve a portion of the budget for bench players** and not allocate it via VORP. For example, you might decide that each team should save ~$20 of their $200 for bench acquisitions, leaving $180 for the starting lineup. If we redo the VORP dollar calculation with $180 per team (so $2160 total instead of $2400) for the top players, their values will increase, reflecting that we want to spend more on starters. The remaining $20 per team is essentially the pool for all the $1-$2 bench players. This approach mirrors advice from experts: *"spend most (or even all) of your cap on starters"* and only minimal on bench [25]. In our model, that was partially implemented with the tiered premium/discount in Step 6. If not already done, we can explicitly do this by reducing the budget used in the main distribution and ensuring a minimum for bench spots.

- Concretely, if each team has 7 bench spots (15 total roster, 8 starters in this league's settings [26]), one might allocate say $1–$2 for each bench spot ($10–$15 total), meaning use ~$185 of the $200 in the primary allocation. This guarantees that after spending on starters (which our model values), teams will have a small leftover to fill the bench with $1 players. The model's output should reflect something similar – perhaps the lowest-ranked draftable players are at $1, and there's a sharp cutoff. That is good and expected.

- Another angle: ensure **each roster position has enough depth**. For instance, each team will need at least one backup RB and WR for bye weeks. The values for those mid-tier RB/WR should not drop to zero if they'll be needed as bye week fill-ins. Our model already gives them some value (they have VORP > 0 since they are above baseline), but this is a reminder to check that the baseline chosen wasn't too high. If we had used a pure "last starter" baseline, bench players would all have ~0 or negative VORP, undervaluing them (that's the VOLS scenario) [27]. If we used a pure waiver baseline, bench players have more value but perhaps too much emphasis (that's VORP scenario) [28]. A moderate baseline or explicit bench budget reservation strikes a balance. **Pitfall:** Over-allocating budget to bench can leave your starting lineup underpowered (e.g. spending $5 each on a bunch of bench guys instead of using that $ to upgrade a starter). Under-allocating leaves you with no safety net if a starter busts or gets hurt. The model's values should encourage spending, say, ~80–90% on starters and ~10–20% on bench, which is a healthy mix.

- **Bye Weeks:** Because we used weekly projections, players' totals are already reduced for their bye week, but it's worth noting how to handle it in roster value. One idea (from some tools) is to **add replacement points for bye weeks or missed games** back to a player's total to represent that you can cover that week with a bench player [7]. For example, if your baseline replacement (waiver) player at RB scores 8 points in a given week, and your star RB will be on bye in Week 7, you could add ~8 points to the star's season total to reflect that you won't actually take a zero, you'll start someone else. This might slightly increase top players' values since their one missed week isn't a total loss to a fantasy team. However, since *every* player has one bye, this adjustment might just add a flat amount to all players' totals and thus not change the relative rankings much – it can be optional. It's more relevant for players who might miss multiple games (where you'd add a few games of replacement). The model can include a toggle for "Impute replacement points for missed games" [7]. By default, for fairness, we can leave bye week replacement out (since it affects players equally). Just ensure in narrative to managers that values assume each player misses their bye – which is true for all.

- **Final Checks:** At this point, confirm that the values make sense holistically: Does each team's optimal spend roughly follow the model? (e.g. If you take the top values for 8–9 starters, does it total around $200? It should if we allocated correctly.) Are there any glaring outliers (like a DST or kicker valued at $5 – if so, we need to fix that in Step 8)? Are positional values reasonable (e.g. probably the top 15 most expensive players are mostly RB/WR, with maybe a top TE and the very top QB in the mix, reflecting that typically RB/WR dominate early spending in a 1-QB league)? If something looks off (say, the 5th best QB has a higher price than the 15th RB, but usually managers would take RB there), revisit baseline or position adjustment. Possibly our calibration with 2024 data would catch this since it aligns with how people actually behave.

- **Documentation:** Include notes in the guide for these strategic considerations so users understand why the model might slightly overprice a player with a big positional drop-off or why almost all bench players are set to $1. These are features, not bugs, aligning with optimal roster construction strategy. As a parallel task, one team member can specifically focus on injecting these roster strategy adjustments (bye week filler logic, ensuring bench allocation) while another focuses on the pure math of Step 6, since these are modular tweaks.

Step 8: Handle DST Modeling Limitations and Fallback Approaches

Task: Address the special case of **Defense/Special Teams (DST)** units, which are notoriously difficult to project and often lack the depth of historical data that players do. We will flag the limitations and use a simplified, cautious approach for DST values. (Similar logic can apply to kickers, though the question highlights DST).

- **Unpredictability of DSTs:** Fantasy DST scoring is highly variable week to week and year to year. Turnovers and defensive touchdowns – which drive DST points – are often random. Moreover, advanced stats for team defenses (like pressure rates, yards allowed, DVOA, etc.) do not always translate cleanly to fantasy points. There is typically a **lack of granular projections** for DSTs compared to skill players. Because of this, any model-driven value for DSTs has high uncertainty. The top-scoring DST in one season can be a mediocre one the next due to schedule changes or randomness.

- **Baseline and Replacement for DST:** In a 12-team league with 1 starting DST each, the baseline is around DST13 (the best available on waivers). Often, streaming a DST (picking one each week based on matchup) can yield roughly top-6 DST results over a season, which means even the concept of a fixed "replacement DST" is tricky – managers can rotate and get good results without investing draft capital. Therefore, **the value of having a top DST is marginal**. Our VORP calculation might show some positive VORP for the top DSTs, but we should be careful not to overvalue it. In practice, fantasy managers tend to pay the minimum ($1) for DSTs, or at most a few dollars for the very top projected unit, if any.

- **Apply a Value Cap or Penalty:** To reflect this, implement a rule to **deflate DST values** in the output. For example, if the raw VORP math said the #1 DST is worth $5, we might cap it at $1–$2. A quantitative way: one toolkit imposed a fixed point penalty to DSTs' VORP to push them down the ranks (because pure projections overrated them relative to expert rankings) [29]. They found that applying about a –6 VORP points adjustment to all DSTs better aligned the values with reality (and similarly penalized kickers more) [29]. We can do something analogous: subtract a certain number of fantasy points from each DST before computing VORP or simply reduce their dollar values post hoc. Essentially, this **bakes in the unpredictability** and the notion that even if a DST projects well, it's not a reliable advantage. As a result, most DSTs will end up at $1, maybe the top one or two at $2 if your league mates like them. That's okay – it matches the common draft behavior where many owners stream DSTs instead of spending auction money on them.

- **Fallback Ranking Approach:** If historical data for DST is missing (say we don't have good projections or prior auction values), a pragmatic fallback is to **rank DSTs by a simple metric** (like their NFL team strength or a composite of last year's points and offseason changes) and then assign all of them the minimum value except maybe the top tier a dollar more. For instance, if the **49ers DST** is consensus top and **Patriots DST** second, you might set 49ers $2, Patriots $1, everyone else $1. The exact values aren't crucial, because the difference between DSTs is small and weekly matchups can level the field. It's more important the model doesn't erroneously suggest spending, say, $10 on a DST because of some projection quirk. By capping these, we avoid a major pitfall.

- **User Guidance:** The model's output should clearly mark DST values with an asterisk or note like "DST values are kept low due to high volatility and the viability of week-to-week streaming." This alerts anyone using the values that spending significant draft capital on a defense is not recommended. If the league has special scoring making DST more important (some leagues heavily reward defenses, though our given rules for DST are fairly standard [30] [31]), you might adjust the cap slightly upward but still be cautious.

- **Pitfall:** *Missing historical data* – often, we have less data on how DSTs were drafted because many sites or leagues don't track $ spent on DST in the same detail. If we lack 2024 auction prices for DST, rely on anecdotal evidence (usually $1 each) or expert rankings which universally put low fantasy value on DSTs relative to any starting RB/WR. Another pitfall would be if our scoring is extremely DST-friendly (e.g. huge points for shutouts or sacks), in which case one or two defenses might actually merit a few more dollars. Check the league scoring: in our case, it gives points for low points allowed and yards allowed, etc. [32] [33] – even so, historically the difference between the #1 DST and #10 DST might be only a couple points per game. So we proceed with a conservative approach.

- **Kickers:** While not explicitly asked, a similar logic can apply to kickers: they are also low-value and unpredictable, so typically all get $1. If the model shows a kicker with higher value, consider capping/penalizing similarly. This ensures focus (and budget) stays on positions that matter most.

- **Implementation:** This can be a simple post-processing step in the **Value Calculator module** – after computing all values, set DST (and K) values to min(ComputedValue, $X cap). Or subtract a fixed VORP points (effectively lowering their computed dollars). The exact method can be tested on 2024 data: for example, if the top DST in 2024 was projected 150 points and baseline 100 (50 VORP) which gave $ say 4, but in draft it went for $1, we know to impose a heavy penalty. The reference approach of ~–18 points for kickers and –6 for DST VORP was empirically derived to align with expert ranks [29]. We can adjust those numbers for our scoring if needed.

- **Outcome:** The final values will likely show almost all DSTs at $1, which is by design given their unpredictability [34]. This is a clear flag in the model output that spending more on DST is not advised, and if someone really wants a top DST, they shouldn't pay much. Our model thus provides a fallback: treat DST in a minimalistic way if robust modeling is not feasible, which is often the most practical solution.

Step 9: Modular Design and Parallel Development Notes

Task: Structure the implementation in a modular way to allow multiple developers (or processes) to work in parallel and to enable easy maintenance. Below is a recommended breakdown of components and their responsibilities:

- **League Settings Module:** Handles input of scoring rules and roster settings (Step 1). It should output a data structure with scoring coefficients (e.g. points per yard, points per TD, bonus thresholds, etc.) and league constants (teams, positions, budget). This module can be completed independently; for testing, one can input known rules and verify it calculates example fantasy points correctly.
- **Projection Processing Module:** Takes raw player projections (from external sources or initial data files) and applies the scoring rules to compute fantasy point projections (Step 2). This includes summing weekly projections to season totals. This module should also incorporate sub-components:
- *Advanced Stats Adjustment Submodule* (Step 3): Ingests historical advanced metrics and modifies the projections accordingly (could be a function that takes a player's initial projection and returns an adjusted projection based on the metrics).
- *Betting Lines Integration Submodule* (Step 4): Ingests sportsbook prop data and merges it, altering the projections where applicable.
  These submodules can be worked on in parallel (one person focuses on pulling and applying advanced metrics, another on fetching and integrating prop data). They both feed into the final adjusted projections. Clear interfaces should be defined, e.g., the projection data format (player ID, stats, initial fantasy points) that each submodule reads and writes to.
- **Value Calculator Module:** Performs the baseline determination and VORP calculations (Step 5) and then converts VORP to dollar values (Step 6). This includes:
- *Baseline & VORP Submodule:* encapsulates the logic for choosing replacement ranks and computing VORP for each player. This can be tested with sample projections (e.g. verify that with simplified input, the correct players have zero VORP at the baseline rank).

- *Dollar Allocation Submodule:* applies the budget distribution and initial dollar assignment. This is where we ensure total sums match budget and apply any tier-based multipliers or position budget constraints.
  These could be combined or sequential in one module, but logically they are distinct steps. They can be developed once the Projection module is delivering results, but a developer could start earlier using dummy projection data to simulate the pipeline.
- **Calibration & Rules Tuning Module:** Handles the adjustments based on historical data and strategic considerations (Steps 6–7). This includes reading in the 2024 actual auction values dataset and computing adjustment factors, as well as encoding rules like “bench budget = 10%” or “premium to starters = 10%” and “DST cap = $1” from Step 8. This module can run analyses on past data to justify the adjustments (for documentation and fine-tuning). It will modify the outputs of the Value Calculator before finalizing. This part benefits from close collaboration with the fantasy domain experts to choose the right values for premiums/penalties. It can be developed and tested using the 2024 projections and known outcomes (backtesting).
- **Output & Reporting Module:** Finally, generate the output guide: a list of players with their auction values, plus any notes/flags (e.g. “DST values capped due to volatility”). Also compile a summary of key insights (like how much of budget to spend on each position, which positions are scarce, etc., derived from the model). This module might simply take the final values and format them for presentation. It should also highlight any edge case notes (e.g. if a player’s value is high due to projected missed games being covered by replacement).

Each of these modules can be developed in parallel by different team members (“agents”) once the interfaces are defined. For example, one can work on scraping and formatting prop bet data while another fine-tunes the VORP baseline logic, and another sets up the scoring formula calculations. Regular integration points (where modules are combined) should be planned, but by dividing the tasks, we speed up development and play to each team member’s expertise (data gathering, analytics, fantasy domain knowledge, etc.). **Potential Pitfalls in Integration:** Ensure consistent **player identifiers** across all data sources (decide on a unique ID for each player so projections, stats, and actual values all refer to the same player). Also, double-check units (e.g. if one projection source uses 0.1 per yard vs our config expects per yard, etc.). Having a few test cases flow through the entire pipeline early can catch these issues.

Finally, maintain clear documentation within each module, especially for any hard-coded adjustments (like “why did we subtract 6 points from DST VORP?” – cite the reasoning [29] in comments). This makes the model transparent and easier to update next year with new data.

By following these steps, we will have a comprehensive auction value model that adapts to custom league settings, leverages a wide array of data for 2025 projections, and grounds its output in sound value-based drafting principles. The modular design ensures the system is maintainable and that improvements or new data can be incorporated by updating individual components without overhauling the whole model. Each step’s careful considerations and calibrations aim to produce values that are not only mathematically sound but also practically useful for dominating a 12-team fantasy auction draft [11] [18]. The end result is an actionable plan for implementation and a set of dollar values that account for **scoring nuances, player projections, advanced metrics, betting markets, and historical market behavior** – giving fantasy GMs a winning edge.

Sources: The approach draws on concepts from fantasy analytics literature and expert community insights, including value-based drafting frameworks [35] [18], research on optimal auction spending strategies [20] [21], the utility of advanced stats in projections [8] [9], and the incorporation of betting lines for improved accuracy [11]. These sources, along with the specific league's rules [1] [2], have been cited throughout to justify the model design choices and to ensure transparency in the method.

---

[1] [2] [3] [4] [5] [15] [26] [30] [31] [32] [33] Scoring.pdf
file://file-SnKevSgP7p4DdECMkgph54

[6] [7] [20] [21] [22] [23] [25] [29] [34] Fantasy Football Auction Draft Optimizer Tool
https://fantasyfootballanalytics.net/2013/06/win-your-fantasy-football-auction-draft.html

[8] [10] Fantasy Football Advanced Stats Guide: Breaking down key analytics and why they'll help win your leagues - CBSSports.com
https://www.cbssports.com/fantasy/football/news/fantasy-football-advanced-stats-guide-breaking-down-key-analytics-and-why-theyll-help-win-your-leagues/

[9] Meet the Metric - Explaining PlayerProfiler's Accuracy Rating Metric
https://www.playerprofiler.com/article/tua-tagovailoa-fantasy-football-ranking-stats-profile-meet-the-metric-accuracy-rating/

[11] [12] [13] [14] Use Vegas Player Prop Data to Dominate Your Fantasy Draft (and Season) : r/fantasyfootball
https://www.reddit.com/r/fantasyfootball/comments/1mq3m15/use_vegas_player_prop_data_to_dominate_your/

[16] [17] [18] [19] [24] [27] [28] [35] Subvertadown | Guide to understanding the different baselines in Value Based Drafting (VBD): VOLS vs. VORP vs. Man-games (and BEER +)
https://subvertadown.com/article/guide-to-understanding-the-different-baselines-in-value-based-drafting-vbd-vols-vs-vorp-vs-man-games-and-beer-