ChatGPT

NFL Data Lab Research Blueprint for Analysts and Advanced Fantasy Users

Executive summary

You asked for a deep-research, module-by-module plan that **includes all player-level and team-level modules and metrics from your text**. The problem: your message references a "listed" module inventory, but the actual list **is not present** in the text I received. Under your stated assumption, this report therefore uses a **standard module registry** (a structured template used in analytics product design) so you can map your existing modules/metrics into it with almost no rework once you paste your inventory.

The core architectural truth (and yeah, it's unsexy): **you don't have a "data lab" until you have a reproducible play-level truth layer with stable IDs, lineage, and automated validation.** That's not vibes; it's standard dimensional modeling practice: facts + dimensions, conformed keys, and quality tests to prevent silent corruption when you update. [1]

Your north-star should be a workflow where an analyst can: - pick a research question ("Is this offense sustainable on early downs?"), - slice by well-defined situations, - see both **outcomes** (EPA/WP, success) and **process** (expected models like xPass/xYAC; tracking-derived separation/coverage if available), - and drill from aggregates → games → drives/series → individual plays. [2]

Recommended priorities and roadmap

Priority	What you build	What it unlocks	Why this is the right order
P0	Play-level "truth" warehouse (IDs, joins, lineage) + automated QA	Trust, reproducibility, consistent drilldowns	Star-schema discipline prevents downstream chaos [3]
P1	Core contextual value layer: EP/EPA + WP/WPA + success rate + drive/series	Most high-signal research questions	These are the backbone metrics for efficiency + leverage [4]
P2	Expected-model enrichment: xPass/PROE + xYAC (and QB EPA attribution if needed)	Usage vs efficiency separation; "what should have happened"	Expected models reduce narrative bias from raw results [5]
P3	Research UX: hub → team/player pages → module detail pages → play evidence	Real "lab" behavior: slicing, comparing, validating	Dashboards must support at-a-glance + drill paths [6]

Priority	What you build	What it unlocks	Why this is the right order
P4	Composite indices + validation/ backtesting	“One-number” summaries that don’t lie	Composites must be decomposable + auditable [7]
P5	Tracking-derived process (if your availability includes it)	Separation/coverage/ pocket/spacing layers	NFL tracking captures 10Hz movement + derived ML stats [8]

Data-source posture (primary-source aligned)

Official-ish NFL capture includes the stadium-entered play-by-play system documented as GSIS (designed to capture play-by-play in-stadium and generate gamebook/reports). [9]
For tracking and ML-derived advanced stats, Next Gen Stats [10] is documented by NFL Football Operations as an in-venue system (UWB receivers + RFID tags), collecting player movement at 10 Hz and producing additional derived data points per play. [11]

Research intentions and measurement principles

A good lab is not “a bunch of charts.” It’s a decision system: you make it easy to test hypotheses, quantify uncertainty, and trace claims back to evidence. [12]

A practical structuring lens is the nested model from Tamara Munzner [13] (affiliated with University of British Columbia [14]), which frames visualization systems as: domain task/data → abstraction → visual encoding/interaction → algorithms. This is exactly how you keep “research intent” from degenerating into random metrics. [15]

The core question families your lab should answer

Efficiency with context: “How good is a player/team given down, distance, score, and time?”
Use expected points (EP) and expected points added (EPA). EPA is commonly defined as the difference between EP after vs before a play (value relative to scoring expectation). [16]

Leverage-aware value: “Did it matter to winning?”
Use win probability (WP) and sum changes in WP (WPA). nflfastR exposes win probability computation and documents required inputs and outputs (including spread-adjusted vs non-spread). [17]

Stability vs volatility: “Is production repeatable or spike-play driven?”
Success rate is a standard efficiency lens that thresholds gains by down and distance; Pro Football Reference’s published definition uses 40%/60%/100% needed yards across downs. [18]

Usage vs efficiency (fantasy-critical): “Is this player earning volume, or just occupying role?”
Expected models help separate what *should* happen from what did: `add_xpass()` (xPass, pass over expected) and `add_xyac()` (expected yards after catch and related EPA expectations) are explicitly documented in nflfastR. [19]

Process (if tracking exists in your availability): "How did it happen?"
NFL Football Ops describes tracking hardware and that it captures location/speed/acceleration at 10 times per second and derives advanced stats via ML. [11]
Independent NFL partner writing also describes streaming coordinates for all 22 players at 10 Hz and the ball faster (25 Hz) in the NGS system. [20]

Baseline metric stack you should standardize

If your platform enforces a "house standard" for only a handful of measures, it should be these: EP/EPA, WP/WPA, success rate, drive/series outputs, xPass/PROE, and xYAC-derived splits. These are all documented/implemented in nflfastR and/or formally defined in published references for success rate and dashboard intent. [21]

Module registry and module-by-module blueprint

Because your module list wasn't included, the best "drop-in" solution is a registry where every module declares:

- **Scope:** frame / play / series / drive / game / week / season
- **Side:** offense / defense / special teams
- **Entities:** player_id, team_id, opponent_id, game_id, play_id (as applicable)
- **Situations:** down, ydstogo, yardline, time remaining, score differential, WP bucket
- **Measures:** volume, efficiency, value, expected, composite
  This structure is consistent with how dimensional models separate measured events (facts) from descriptors (dimensions), enabling drilling and slicing without logic duplication. [22]

Module-by-module analysis table

The table below is written as a *specification template*. When you paste your actual module/metric inventory, you map each of your modules into one row (or split a row if your module is broader).

Key metric definitions and availability cues referenced here are from nflfastR (EP/WP calculators, xPass, xYAC, series conversion) and PFR's success rate definition. [23]

Module	High-value research questions	Best visualizations	Composite metrics to build	Validation checks
Team offense efficiency	Are we efficient overall, and where (early downs, red zone, neutral script)?	Rolling EPA/play line; down×distance heatmap; EPA/play distribution	Early-down EPA/play; explosive rate + success blend	EPA totals reconcile with play counts; situation filters are explicit [24]

Module	High-value research questions	Best visualizations	Composite metrics to build	Validation checks
Team defense efficiency	Do we prevent sustaining drives or just avoid explosives?	EPA allowed heatmap; success allowed by down; drive-outcome tree	Defensive consistency index = z(success allowed)+z(explosive allowed)	Opponent assignment stable; defense/offense play tagging correct
Game leverage and win impact	Which plays/ players swing win probability?	WPA leaderboard with play drill; “top 10 WPA plays” table	WPA over time; leverage-weighted EPA	WP pre/post alignment and spread/non-spread labeling correct [17]
Play calling identity	Are we more/ less pass-heavy than situation expects?	PROE trend; PROE by score state; 1st-down pass rate vs xPass	Team PROE = avg(pass_oe)	`add_xpass()` NA pre-2006; handle missingness without breaking charts [25]
QB passing value	Is QB creating value or riding YAC/scheme?	EPA/dropback trend; CPOE vs aDOT scatter; “air vs YAC value” bars	QB value split (air EPA + YAC-over-expected EPA)	If using nflfastR attribution, keep versioning (changelog notes fixes) [26]
Receiver usage + efficiency	Is target earning real? Who’s a volume-only mirage?	Usage vs EPA/ target scatter; air yards vs YAC split	Receiver Value = air_epa + (yac_epa − xyac_epa)	xYAC fields exist and are computed consistently for seasons used [27]
YAC skill and design separation	Who adds after-catch value beyond expectation?	YAC vs xYAC scatter; distribution of (YAC − xYAC)	YAC+ per target = avg(YAC − xyac_mean_yardage)	xyac_mean_yardage meaning documented; missingness tracked [28]
Rushing (team/RB)	Is rushing creating value or just avoiding negatives?	EPA/rush by box count (if available); success rate by down	Rush contribution index (volume + EPA/ rush + success)	Run/pass labeling stable; penalty handling consistent

Module	High-value research questions	Best visualizations	Composite metrics to build	Validation checks
Series/drive sustainability	Can we move chains and finish drives?	Series conversion trend; drive start field heatmap; drive result tree	Sustainability score = z(series conv)+z(early-down success)	Series definition and kneel exclusions documented in function [29]
Special teams + field position	Are we winning hidden yards?	Starting field position distributions; punt/return impact bars	Field position value via EP deltas	ST play tagging consistent; EP inputs for those plays valid [30]
Tracking-derived process (if available)	Separation, spacing, coverage, rush proximity—why outcomes happen	Route/separation timelines; pocket maps; coverage vs EPA	Process indices (separation vs expected, pressure rate vs time-to-throw)	Frame alignment, event alignment, completeness; keys gameId/playId/nflId [31]

Composite metric formulas you can implement immediately

The formulas below intentionally rely on *documented* base fields (EP/WP calculators, xPass definitions, xYAC documentation, series conversion definition). [32]

Composite	Formula	Inputs	Why it's useful	Guardrails
EPA/play	ΣEPA ÷ N plays	EPA, play filter	Core efficiency summary	Always show N; allow situation presets [16]
WPA	Σ(WP_after − WP_before)	WP pre/post	Leverage-aware value	Label spread vs non-spread WP explicitly [17]
Success rate	successful ÷ total, with 40/60/100 thresholds	down, ydstogo, yards gained	More stable than EPA	Publish the definition in UI metadata [18]
PROE	avg(pass_oe)	pass_oe	Team identity independent of raw pass rate	Handle NA pre-2006 as documented [25]
YAC-over-expected (EPA)	avg(yac_epa − xyac_epa)	yac_epa, xyac_epa	Separates created YAC from expected	Use nflfastR field semantics; show season coverage [27]

Composite	Formula	Inputs	Why it's useful	Guardrails
Sustainability score	z(series conv) + z(early-down success)	series conv, success	Drive-building stability	Keep normalization within season + context filters [33]

Practical validation checks (per module and globally)

Treat data QA as product functionality, not plumbing. dbt's built-in test taxonomy is a good baseline: `unique`, `not_null`, `accepted_values`, and `relationships` (referential integrity). [34]

Minimum test suite to ship P0: - Play keys are unique and non-null; player keys non-null where expected. [35]
- Relationship tests: every play references a valid game; every player-play references a valid play and player. [36]
- Accepted values for enumerations (play_type, side, down ranges). [35]
- Versioning/lineage fields to detect model updates and upstream changes (critical when using expected models and public packages that evolve). [37]

Dashboard layouts and interaction patterns

Dashboards are only "good" if they support monitoring and comprehension on a single screen, with the ability to drill into detail when something looks off. Perceptual Edge [38] and Stephen Few [39] explicitly define dashboards as single-screen monitoring tools and emphasize clarity over decoration. [40]

Interaction patterns to copy (because they work): - Cross-filter / cross-highlight between visuals for fast hypothesis testing. [41]
- Drillthrough to detail pages (player → game log → plays) to keep aggregates honest. [42]

Suggested dashboards

Hub page (League + research launchpad)
A single screen with: - Global filters: season, week range, situation preset (neutral script, early downs, red zone), team/opponent. - Left: sortable team table (EPA/play, success rate, PROE, series conversion). - Center: "identity DNA" panel (PROE trend + early-down success trend). - Right: distribution view (league EPA/play histogram) with selected team highlighted.
This is aligned with dashboard-as-monitoring: one screen, key signals, clear context. [40]

Team page (research-grade, not fan-page)
Top strip: team season summary with sparklines and sample sizes.
Middle: two heatmaps (offense and defense) for EPA/play or success rate by down×distance.
Bottom: drive/series outcomes and a play list (the "evidence table") filtered by any click.
Series concept and computation are well-defined in nflfastR's series conversion documentation. [29]

Player page (role → efficiency → evidence)
Header: player, position, team(s), season range, and **hard sample-size guardrails** (snaps, targets, rushes,

dropbacks).
Core visuals: - Usage vs efficiency scatter (position peer filter) - Split bars (by down, field zone, score state) - “Top plays” list by EPA and WPA (positive and negative) with drill to play detail
WP and EP computation inputs/outputs are documented, supporting reproducible play-level drilldowns. [43]

Module detail page (definition-first)
Every module page starts with: definition, included/excluded plays, data lineage (source, refresh timestamp, model version), and known limitations. This matches the nested-model emphasis on explicit domain framing and validation. [15]

Visualization option comparison table

Visual	Strength	Best for	Failure mode	Fix
Rolling line	Trend + stability	EPA/play, PROE, success rate	Hides small-sample volatility	Show N and optional confidence bands
Quadrant scatter	Role vs performance	Usage vs efficiency	Lies with tiny samples	Size by volume; min-threshold toggles
Heatmap	Situation clarity	down×distance, red zone	Over-binning creates fake patterns	Show counts; drill to play table
Distribution plot	Variance & outliers	Boom/bust players	Mixes contexts	Filters/presets per context
Outcome tree/ Sankey	Process comprehension	drive/series outcomes	Visual clutter	Limit nodes; click-path filtering

The “make it monitorable and comprehensible” orientation is consistent with Few’s dashboard principles. [40]

Data model and tagging schema

Dimensional modeling is the cleanest fit for an NFL lab because it naturally supports: conformed keys, stable drill paths, and performance for aggregate queries. Kimball Group [44] defines fact tables as measurements joined via foreign keys to descriptive dimensions in a star schema. [45]

Tagging schema (the part most sports tools screw up)

A tagging layer keeps modules composable and prevents “one-off metric pages that can’t be combined.”

Tag	Examples	Why it matters
scope	play, drive, series, game, season, frame	Determines valid aggregations and drilldowns
side	offense, defense, special_teams	Prevents attribution mixing
position_group	QB, RB, WR, TE, OL, DL, LB, DB, K, P	Enables peer comparisons
metric_family	volume, efficiency, value, expected, composite	Chooses sensible default visuals
situation_dims	down, ydstogo, yardline, time, score_diff, WP bucket	Reusable filters across modules
lineage	source, refresh_ts, model_version	Reproducibility and trust

Recommended warehouse tables and major fields

This table set supports both (a) play-by-play only and (b) tracking-enhanced modes. Tracking schemas commonly use game-play-player join keys; the NFL Big Data Bowl schema explicitly identifies `gameId`, `playId`, and `nflId` as key variables, with frame-level x/y/s/dis/dir/event fields. [46]

Dimensions - `dim_game`: game_id, season, week, home_team, away_team, kickoff, venue/weather tags (if available).
- `dim_team`: team_id, abbreviation, conference/division, branding history (optional SCD).
- `dim_player`: player_id, name, position, physicals, entry year, team history (optional SCD).
SCD techniques (including Type 2 rows for history) are a standard Kimball pattern when entity attributes change over time. [47]

Facts - `fact_play`: game_id, play_id, posteam, defteam, play_type, yards_gained, penalties, EP, EPA, WP pre/post, success flag, and situation fields required by calculators. [48]
- `fact_player_play`: play_id + player_id + role (passer/rusher/target/defender if available), opportunities and value attribution fields.
- `fact_drive` and `fact_series`: drive/series identifiers, start state, result, points, field position outcomes; series conversion definitions and kneel exclusions are documented in nflfastR. [29]
- `fact_team_game`: pre-aggregated metrics for fast dashboards (EPA/play, PROE, success, series conversion).
- `fact_tracking_frame` (optional): game_id, play_id, nflId/player_id, frame_id, x, y, s, dis, dir, event. [49]

Data quality rule map

Implement these as tests plus monitoring: - Key uniqueness / not-null (plays, games, players). [35]
- Relationships (fact → dimension) as referential integrity; dbt documents `relationships` semantics and that NULLs are excluded unless separately tested. [36]
- Accepted values for categorical columns. [35]
- Freshness checks and model version fields for expected models / package-driven logic (nflfastR changelog demonstrates that model logic can change/fix). [50]

Implementation plan with milestones, computations, and cadence options

Required computations (minimum viable lab)

These are directly supported/defined by nflfastR docs: - EP and EPA via expected points computation. [16]
- WP pre/post via win probability computation (spread and non-spread variants). [17]
- Success flags (use an explicit published definition, e.g., PFR 40/60/100). [18]
- xPass and pass_oe via expected dropback model (with NA handling pre-2006 as documented). [25]
- xYAC fields including xyac_mean_yardage and xyac_epa, enabling YAC-over-expected analysis. [28]
- Series conversion and series results (with kneel-down exclusion as documented). [29]

Update cadence options

Because you left cadence unspecified, build your pipelines so the same tables can refresh under different schedules:

- **Postgame batch:** refresh after each game window; stable “final” labels.
- **Nightly incremental:** daily refresh for research; weekly full rebuild for corrections.
- **Near-real-time (only if feeds support it):** mark as provisional, and reconcile to postgame final.

This is mainly a product-trust decision: dashboards are for monitoring, so freshness and correctness must be communicated clearly. [40]

Mermaid flowchart for data flow

```
flowchart LR
  A[Stadium-entered play by play systems e.g. GSIS exports] --> B[Raw zone
immutable]
  C[Modeled pbp + schedules + rosters] --> B
  D[Tracking + derived stats if available] --> B

  B --> E[Clean zone normalize IDs]
  E --> F[Feature zone EP EPA WP WPA success xPass xYAC]
  F --> G[Warehouse star schema facts + dimensions]

  G --> H[Query API semantic layer]
  H --> I[Hub page]
  H --> J[Team pages]
  H --> K[Player pages]
  H --> L[Module detail pages]

  G --> M[QA tests + monitoring]
  M --> N[Freshness anomalies lineage]
```

GSIS documentation supports the “capture play-by-play at each stadium” framing, and NFL Football Ops documents the tracking system and derived ML stats for NGS. [51]

Mermaid gantt for a pragmatic build

```
gantt
  title NFL Data Lab Build Plan
  dateFormat  YYYY-MM-DD
  axisFormat  %b %d

  section Foundation
  IDs + star schema + lineage                :a1, 2026-02-17, 14d
  dbt-style QA tests + monitoring            :a2, after a1, 14d

  section Core context metrics
  EP/EPA + WP/WPA + success                  :b1, after a1, 21d
  Drives + series conversion marts           :b2, after b1, 14d

  section Expected models
  xPass/PROE and xYAC integration            :c1, after b1, 14d

  section Product experience
  Hub/team/player/module pages + drillthrough :d1, after b1, 21d

  section Advanced
  Composite indices + backtesting            :e1, after d1, 21d
  Tracking-derived process layer (optional)  :e2, after e1, 28d
```

The QA milestone is grounded in documented dbt test categories and referential integrity semantics. [34]
The visualization and drilldown emphasis aligns with dashboard design principles and cross-filter/drillthrough patterns documented for BI systems. [6]

Example “visual assets” to build first

Wireframe descriptions (text-only, platform-agnostic)
- **Hub:** filters (season/week/situation) + team rank table + PROE trend + EPA distribution + “top movers.” [40]
- **Team page:** KPI strip + offense/defense heatmaps + drive outcomes + evidence play table. [52]
- **Player page:** usage/role strip + usage vs efficiency scatter + splits + top positive/negative plays by EPA/WPA. [53]
- **Module detail:** definition + primary chart + secondary chart + evidence + export. [15]

Three chart recommendations (implement early) - **Rolling line chart:** weekly rolling EPA/play (team) or EPA/opportunity (player), with sample-size display. [54]
- **Usage vs efficiency scatter:** e.g., target share vs EPA/target; dropbacks vs EPA/dropback; size by opportunities. [55]

- **Down×distance heatmap:** success rate or EPA/play by down and yards-to-go buckets; click-to-drill plays. [56]

One blunt requirement to make this “research-grade”

Every composite or ranking view must be: 1) **decomposable** into components on click, and
2) **traceable** back to the underlying plays.
That’s not optional if you want a lab instead of a leaderboard, and it’s consistent with the nested-model emphasis on validated abstraction and encoding choices that support real tasks. [57]

---

[1] [3] [22] [45] Fact Tables and Dimension Tables
https://www.kimballgroup.com/2003/01/fact-tables-and-dimension-tables/?utm_source=chatgpt.com

[2] [7] [12] [15] [57] A Nested Model for Visualization Design and Validation
https://www.cs.ubc.ca/labs/imager/tr/2009/NestedModel/NestedModel.pdf?utm_source=chatgpt.com

[4] [10] [16] [21] [23] [24] [30] [32] [43] [48] [53] [54] [55] Compute expected points — calculate_expected_points
https://nflfastr.com/reference/calculate_expected_points.html?utm_source=chatgpt.com

[5] [19] [25] [38] Add expected pass columns — add_xpass
https://nflfastr.com/reference/add_xpass.html?utm_source=chatgpt.com

[6] [14] [40] [44] Dashboard Design for Real-Time Situation Awareness
https://www.perceptualedge.com/articles/Whitepapers/Dashboard_Design.pdf?utm_source=chatgpt.com

[8] [11] [13] [31] NFL Next Gen Stats | NFL Football Operations
https://operations.nfl.com/gameday/technology/nfl-next-gen-stats/

[9] [51] nflgsis.com
https://nflgsis.com/gsis/documentation/stadiumguides/stadium_technician_s_guide.pdf

[17] Compute win probability — calculate_win_probability
https://nflfastr.com/reference/calculate_win_probability.html?utm_source=chatgpt.com

[18] [56] Success Rate Comes to Pro Football Reference!
https://www.sports-reference.com/blog/2023/09/success-rate-comes-to-pro-football-reference/?utm_source=chatgpt.com

[20] A decade of NFL Next Gen Stats innovation
https://www.amazon.science/blog/a-decade-of-nfl-next-gen-stats-innovation?utm_source=chatgpt.com

[26] [37] [50] Changelog
https://nflfastr.com/news/index.html?utm_source=chatgpt.com

[27] Get started with nflfastR
https://nflfastr.com/articles/nflfastR.html?utm_source=chatgpt.com

[28] add_xyac Add expected yards after completion (xyac) variables
https://www.rdocumentation.org/packages/nflfastR/versions/5.1.0/topics/add_xyac?utm_source=chatgpt.com

[29] [33] [52] Compute Series Conversion Information from Play by Play
https://nflfastr.com/reference/calculate_series_conversion_rates.html?utm_source=chatgpt.com

[34] [35] [39] Add data tests to your DAG | dbt Developer Hub
https://docs.getdbt.com/docs/build/data-tests?utm_source=chatgpt.com

36 About data tests property | dbt Developer Hub
https://docs.getdbt.com/reference/resource-properties/data-tests?utm_source=chatgpt.com

41 Filters and highlighting in Power BI reports
https://learn.microsoft.com/en-us/power-bi/create-reports/power-bi-reports-filters-and-highlighting?utm_source=chatgpt.com

42 Drillthrough in Power BI Reports: Navigate to Detailed ...
https://learn.microsoft.com/en-us/power-bi/create-reports/desktop-drillthrough?utm_source=chatgpt.com

46 49 raw.githubusercontent.com
https://raw.githubusercontent.com/nfl-football-ops/Big-Data-Bowl/master/schema.md

47 Slowly Changing Dimensions
https://www.kimballgroup.com/2008/08/slowly-changing-dimensions/?utm_source=chatgpt.com