

Requirements Stack

ID	Requirement Description	Story Points	Priority	Sprint Number
R1	Locate RESTful FastAPI backend to serve sports data (teams, players, stats, odds) using PostgreSQL and Redis caching.	13	1	1
R2	Integrate live data feeds from APIs for football, basketball, baseball, and UFC. Normalize and store in unified schema.	13	1	1
R3	Build WebSocket interface for real-time game and stat updates to connected clients.	8	1	1
R4	Create front-end interface (React Native or PWA) to display teams, players, and stats.	8	1	1
R5	Integrate live data feeds from APIs for football, basketball, baseball, and UFC. (Begin connecting selected API endpoints and normalizing response data.)	13	1	2
R6	Build WebSocket interface for real-time game and stat updates to connected clients. (Start implementing live update mechanism.)	8	1	2
R7	Integrate betting odds API and display odds for upcoming events with bookmaker info.	5	2	2
R8	Ensure compliance with betting data policies and user age verification.	2	5	2
R9	Develop interactive visualization dashboards with charts for stats and metrics.	8	5	2
R10	Implement search and filtering by sport/team/player.	3	1	3
R11	Implement user-defined alerts for news mentions/injury updates.	8	2	3
R12	Store and display recent injury reports (initial schema).	3	2	3
R13	Optimize queries + Redis caching improvements.	3	3	3
R14	Implement fuzzy matching logic for associating news → players/teams.	5	5	3
R15	Implement ESPN Sports API directly through Vercel.	8	5	3
R16	Begin player/team comparison module with AI reasoning hooks.	5	5	3
R17	DB backup/recovery/migrations (Alembic).	5	5	3
R18	Optimize Supabase authentication flow to support stable user login sessions and faster response times.	3	3	4
R19	Enable storing and retrieving user-specific login details, preferences, and profile metadata within Supabase.	5	4	4

Requirements Stack

ID	Requirement Description	Story Points	Priority	Sprint Number
R20	Enforce login-gated access for restricted features (chatbot, advanced comparisons, personalized dashboards).	5	4	4
R21	Expand chatbot with more robust team and player comparison logic, using optimized datasets and AI reasoning.	3	5	4
R22	Add conversational chat features including persistent chat context and improved user interaction flow.	5	5	4
R23	Optimize Vercel frontend deployment (improved routing, reduced build size, faster loading).	3	5	4
R24	Optimize sports API ingestion logic, reducing latency and improving data freshness + consistency.	5	4	4
R25	Improve responsive UI across devices after Vercel optimizations (layout fixes, scaling improvements).	5	4	4
R26	Integrate improved JWT/Supabase token-based authentication for secure access control to premium features.	5	5	4
R27	Implement data integrity improvements for sports API pipeline (duplicate filtering, validation, and failover logic).	3	5	4
R28	Enhance analytics module to capture user login activity, chatbot usage, and team comparison metrics.	3	5	4
R29	Implement pagination/sorting/virtualization for faster browsing of player/team lists after API optimization.	3	5	4
R30	Add API rate-limiting and throttling aligned with new optimized data ingestion flow.	3	5	4
R31	Improve social sharing components to support updated comparison and chatbot output formats.	3	5	4
R32	Finalize CI/CD automated pipeline and ensure smooth Vercel deployment flow with updated optimizations.	8	5	4