

# 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, or player across datasets.	3	1	3
R11	Implement user-defined alerts for news mentions and injury updates of subscribed teams. (Set groundwork for alert logic.)	8	2	3
R12	Store and display recent injury reports per team/player. (Initial database schema design.)	3	2	3
R13	Optimize DB queries and introduce Redis caching for high-traffic endpoints.	3	3	3
R14	Implement fuzzy matching logic for news articles to associate with teams and players.	5	5	3
R15	Add UFC-specific features like weight class tracking and fight round stats.	8	5	3
R16	Implement player and team comparison features in the user interface.	5	5	3
R17	Implement database backup, recovery, and migration procedures using Alembic.	5	5	3
R18	Containerize services with Docker and deploy via Docker Compose.	3	3	4
R19	Create admin dashboard to manage API keys, data sources, and logs.	5	4	4

# Requirements Stack

ID	Requirement Description	Story Points	Priority	Sprint Number
R20	Implement data deduplication and validation pipeline for ingest jobs.	5	4	4
R21	Integrate email as an additional notification channel for user alerts.	3	5	4
R22	Add logging, monitoring, and error handling across all services with Prometheus and Grafana.	5	5	4
R23	Create user profile management including preferences and subscription settings.	3	5	4
R24	Add mobile notifications for live game updates and subscribed team news.	5	4	5
R25	Design responsive UI supporting both desktop and mobile clients.	5	4	5
R26	Add authentication (JWT) for user accounts and subscriptions to teams/players.	5	5	5
R27	Build Discord bot/webhook integration for sending alerts and live updates.	3	5	5
R28	Add analytics module tracking user engagement and alert metrics.	3	5	5
R29	Add pagination, sorting, and infinite scrolling for lists in the UI.	3	5	5
R30	Implement rate limiting and API throttling to prevent abuse.	3	5	5
R31	Enable social media sharing of stats, alerts, and dashboards.	3	5	5
R32	Set up CI/CD pipeline for automated testing and deployment to cloud infrastructure.	8	5	5