TeamSync AI - Project Roadmap

Version: 1.0
Last Updated: November 8, 2025
Project Status: Phase 1 - MVP Development

Executive Summary

TeamSync AI is a comprehensive youth sports team management platform designed to streamline team operations, enhance communication, and provide AI-powered insights for coaches, players, and parents. This document outlines our phased development approach, starting with a focused MVP that delivers core value while establishing a foundation for future scalability and advanced features.

Table of Contents

Phase 1: MVP Features (Building Now)

1.1 Team Roster Management

Description: Comprehensive team member management system

Features:
- **Player Profiles**
- Personal information (name, age, position, jersey number)
- Contact details (email, phone)
- Emergency contact information
- Medical information and notes
- Player photo upload
- Custom fields for sport-specific data

- **Coach Profiles**
- Contact information
- Coaching certifications
- Specialization areas

- Bio and experience
- **Staff Profiles**
- Administrative staff
- Medical staff (trainers, physiotherapists)
- Parent volunteers
- Role assignments
- **Roster Views**
- Grid view with sortable columns
- Card view with photos
- Export to CSV/PDF
- Search and filter capabilities

Priority: Critical - Foundation for all other features

1.2 Event & Practice Scheduling

Description: Comprehensive calendar management system

Features:
- **Event Creation**
- Practice sessions
- Games/matches
- Team meetings
- Social events
- Custom event types

- **Event Details**
- Date and time
- Location with map integration
- Duration
- Recurrence patterns (daily, weekly, custom)
- Attendee requirements (mandatory/optional)
- Equipment needed
- Notes and instructions
- **Calendar Views**
- Monthly calendar view
- Weekly agenda view
- List view with filters
- Individual player/parent calendar view
- **Attendance Tracking**
- RSVP system (Attending/Not Attending/Maybe)
- Attendance history

- Late arrival tracking
- Absence reason collection
- **Notifications**
- Event reminders (24hr, 1hr before)
- Schedule change alerts
- Cancellation notifications

Priority: Critical - Core functionality

1.3 Announcement Board

Description: Team-wide communication hub

Features:
- **Post Creation**
- Rich text editor
- Attachment support (images, documents)
- Post categories (General, Important, Urgent)
- Scheduled posting
- Draft saving

- **Post Management**
- Edit/delete capabilities (coaches/admins only)
- Pin important announcements
- Archive old posts
- **User Interaction**
- View announcements feed
- Mark as read/unread
- Filter by category
- Search functionality
- **Notifications**
- New announcement alerts
- Urgent post push notifications
- Daily digest option

Priority: High - Essential for team communication

1.4 Analytics Dashboard with Player Position Tracking

Description: Data-driven insights for coaches and administrators with visual field position analytics

Features:
- **Attendance Analytics**
- Overall team attendance rate

- Individual player attendance trends
- Attendance by event type
- Absence patterns
- Comparative visualizations

- **Participation Stats**
- Active vs inactive members
- Event participation rates
- Engagement metrics
- **Performance Metrics**
- Custom performance indicators by sport
- Individual player progress tracking
- Team averages and benchmarks
- Performance over time graphs
- **Player Position Tracking & Heat Maps**
- **Position Time Tracking**
  - Track time spent at each position during games/practices
  - Record multiple positions per session
  - Timestamp entry and exit from positions
  - Historical position data over season
  - Position rotation patterns
- **Visual Field Heat Maps**
  - Color-coded heat map showing where player spent time
  - Intensity based on duration at each location
  - Overlay on sport-specific field layout
  - Filter by game, practice, date range, or season
  - Compare positions across multiple games
- **Position Distribution Analytics**
  - Percentage breakdown by position
  - Time spent in each field zone/area
  - Most/least played positions
  - Position versatility score
  - Visual pie charts and bar graphs
- **Parent & Player Portal**
  - Dedicated view for players and parents
  - Permission-based sharing by coach
  - Individual player position reports
  - Season summary with visual highlights
  - Download reports as PDF
  - Mobile-friendly viewing

- **Coach Controls**
  - Toggle visibility for players/parents
  - Record positions during live games/practices
  - Quick position entry interface
  - Bulk position assignment
  - Edit historical position data
  - Add notes to position changes (e.g., “Trying new position”)
- **Position Entry Methods**
  - Manual entry during/after games
  - Quick-tap interface for live tracking
  - Import from game sheets
  - Bulk upload via CSV
  - Integration with drill planning system
- **Insights & Recommendations**
  - Position balance recommendations
  - Player development suggestions based on position variety
  - Alert if player hasn’t played certain position in X games
  - Compare position distribution across team
- **Visualization Components**

- Line charts for trends
- Bar charts for comparisons
- Pie charts for distributions
- Heat maps for position tracking
- Data tables with export options

- **Reporting**

- Generate PDF reports
- Custom date range selection
- Share reports with stakeholders
- Position tracking reports for individual players
- Team-wide position distribution reports

Priority: High - Provides value differentiation and transparency

---

1.5 File Sharing System

Description: Centralized document management

Features:
- **File Upload**
- Multiple file type support (PDF, DOCX, images, videos)
- Drag-and-drop interface

- Bulk upload capability
- File size limits and validation

- **Organization**
- Folder structure
- Categories (Playbooks, Forms, Medical, Administrative)
- Tags for easy discovery

- **File Management**

- Preview functionality
- Download files
- Version control (track changes)
- Delete/archive files

- **Access Control**

- Role-based file visibility
- Private files (coaches only)
- Team-wide files
- Parent-accessible files

- **Search & Discovery**

- Full-text search
- Filter by type, category, date
- Recent files view

Priority: Medium-High - Important for organization

1.6 User Authentication & Role-Based Access

Description: Secure authentication system with granular permissions

Features:
- **Authentication**
- Email/password signup and login
- Email verification
- Password reset functionality
- Secure session management
- Remember me option

- **Role System**
- **Coach/Administrator:** Full access to all features, can manage team, create events, post announcements
- **Player:** View schedule, RSVP to events, view announcements, access shared files, view own stats
- **Parent/Guardian:** View player’s schedule, RSVP on behalf of player, view announcements, receive notifications

- **Permission Granularity**

- Create/edit/delete permissions by role
- View-only access for specific features
- Custom role creation (future enhancement)
- **Profile Management**
- Update personal information
- Change password
- Notification preferences
- Privacy settings

Priority: Critical - Security foundation

1.7 Core Integrations

Google Calendar Sync

- Two-way sync with Google Calendar
- Export events to personal calendars
- Automatic updates on changes
- Subscribe to team calendar via iCal

Stripe Payment Links

- Generate payment links for:
- Team fees
- Event registration
- Merchandise
- Tournament fees
- Payment tracking dashboard
- Receipt generation
- Payment status updates

Email Notifications

- Transactional emails (sign up, password reset)
- Event notifications
- Announcement alerts
- Weekly digest emails
- Customizable notification preferences

Priority: High - Essential for user convenience

1.8 AI Features (Phase 1)

Schedule Optimization Suggestions

- Analyze team availability patterns
- Suggest optimal practice times
- Identify scheduling conflicts
- Consider travel time between locations

- Weather-based recommendations

Performance Insights

- Identify performance trends
- Player development tracking
- Compare individual vs team averages
- Highlight areas for improvement
- Generate coaching tips

Attendance Predictions

- Predict likely attendance for upcoming events
- Identify players at risk of low attendance
- Forecast seasonality patterns
- Early warning system for engagement drops

Priority: Medium - Differentiating features

1.9 UI/UX Design

Description: Modern, responsive interface

Features:
- Responsive design (mobile, tablet, desktop)
- Intuitive navigation
- Accessible design (WCAG compliance)
- Dark mode support
- Loading states and error handling
- Toast notifications for actions
- Confirmation dialogs for destructive actions

Priority: High - User experience is critical

1.10 Visual Drill Planning & Field Configurator

Description: Interactive field layout designer for practice planning with multiple concurrent drills

Features:

Field Configuration System

- **Sport-Specific Fields**
- Pre-loaded templates for popular sports:
  - Soccer (full field, half field, small-sided grids)
  - Basketball (full court, half court, 3-point zones)
  - Baseball/Softball (full diamond, infield only, batting cages)
  - Football (full field, red zone, 7v7 configurations)
  - Hockey (full rink, zones, half ice)
  - Lacrosse, Volleyball, and more
- Accurate field dimensions based on age group and league rules

- Customizable field markings and zones
- **Practice Size Configurations**
- Full field layouts for large practices
- Half field/split field for medium groups
- Small-sided game areas (3v3, 5v5, 7v7)
- Multiple station configurations
- Custom grid creation for any practice size
- **Dynamic Field Resizing**
- Adjust field dimensions based on available space
- Scale fields proportionally
- Save custom field configurations
- Quick switch between configurations

Drill Placement & Management

- **Drag-and-Drop Interface**
- Visual drill placement on field
- Resize drill areas with handles
- Rotate drill orientation
- Snap-to-grid for precise positioning
- Collision detection (prevent overlapping drills)
- **Multi-Drill Support**
- Run multiple drills simultaneously on same field
- Assign player groups to each drill station
- Track drill duration and rotation timing
- Color-code drill areas for clarity
- Label stations (Station A, B, C, etc.)
- **Drill Library**
- Pre-built drill templates by sport
- Save custom drill configurations
- Import drills from other coaches (future)
- Tag drills by skill focus (passing, shooting, defense)
- Search and filter drill library
- Drill diagrams with player movement arrows
- **Player/Group Assignment**
- Assign players to specific drill stations
- Auto-distribute players across drills
- Balance groups by skill level
- Rotation scheduler (switch every X minutes)
- Track which players have completed which drills

Practice Plan Builder

- **Session Planning**
- Create practice plans with multiple field configurations
- Timeline view of practice structure
- Allocate time blocks to each drill
- Add warm-up, cool-down, and water break periods
- Total practice time calculator
- **Configuration Switching**
- Move between field setups during practice
- Transition planning (setup time between configurations)
- Coach notes for each configuration
- Equipment requirements per setup
- **Template Management**
- Save practice plans as templates
- Clone and modify existing plans
- Share practice plans with assistant coaches
- Season planning: Schedule practice themes

Visualization & Export

- **Interactive Field View**
- Zoom and pan controls
- Toggle field markings visibility
- Switch between 2D and isometric view
- Print-friendly layouts
- Dark mode support
- **Export Options**
- Export field layouts as images (PNG, PDF)
- Print practice plans with diagrams
- Email practice plan to coaching staff
- Mobile-friendly view for on-field reference
- **Animation & Presentation**
- Animate player movements in drills
- Step-by-step drill demonstrations
- Presentation mode for team meetings
- Video recording of animated drills (future)

Technical Implementation

- **Frontend Technology**
- Canvas-based rendering (HTML5 Canvas or SVG)
- React-based interactive components
- Drag-and-drop library (react-dnd or dnd-kit)

- Touch-friendly for tablet use
- **Data Model**
- Field configuration schema (dimensions, markings, zones)
- Drill placement coordinates and properties
- Practice plan structure (timeline, drills, assignments)
- Template definitions (reusable field layouts)
- **Performance Optimization**
- Lazy loading for drill library
- Efficient canvas rendering
- Responsive design for all devices
- Offline mode for on-field use (future)

Priority: High - Key differentiator for practice planning

1.11 Intelligent Web Scraping & Data Import System

Description: Automated data extraction from sports websites with template-based learning approach

Features:

Intelligent Web Scraping System

- **Template-Based Learning Approach**
- Analyze page structure and DOM elements
- Create reusable extraction patterns for each site
- Store templates for consistent data extraction
- Learn from successful extractions to improve patterns
- Version control for template changes
- **Multi-Site Support**
- **GameChanger (Primary: Scraping)**
    - Team rosters and player information
    - Game schedules and results
    - Player statistics and performance data
    - Game details (location, opponent, score)
- **QuickScores (Primary: API, Backup: Scraping)**
    - Use official API when available
    - Fallback to web scraping if API fails
    - Schedule and score synchronization
    - Team and player data import
- **Extensible Architecture**
    - Easy addition of new sports websites
    - Plug-and-play site adapters
    - Configuration-based site definitions

Data Extraction Capabilities

- **Teams & Rosters**
- Bulk import team member information
- Automatic profile creation for players
- Position and jersey number mapping
- Contact information extraction (when available)
- **Schedules & Events**
- Automatic calendar population
- Game and practice schedule imports
- Location and time zone handling
- Recurring event pattern detection
- **Scores & Results**
- Real-time score updates
- Historical game results
- Win/loss record tracking
- Score differential analytics
- **Player Statistics**
- Performance metrics by sport
- Individual player stats
- Season and career statistics
- Comparative analytics

Template Management Interface

- **Template Viewer**
- Browse available scraping templates
- Preview template structure and selectors
- View template effectiveness metrics
- Template version history
- **Template Editor**
- Visual selector builder
- Test templates on live pages
- Validation and error checking
- Template cloning for similar sites
- **Template Management**
- Create new templates for unsupported sites
- Update existing templates when sites change
- Delete obsolete templates
- Import/export template configurations
- Admin approval workflow for user-created templates

Technical Implementation

- **DOM Parsing & Analysis**
- Headless browser for JavaScript-rendered sites (Puppeteer/Playwright)
- HTML parser for static content (Cheerio/BeautifulSoup)
- Intelligent element detection
- Fallback strategies for layout changes
- **Selector Technology**
- CSS selectors for precise element targeting
- XPath for complex DOM queries
- Relative positioning for robust extraction
- Attribute-based selectors for reliability
- **Pattern Storage & Versioning**
- JSON-based template definitions
- Database storage for templates
- Version control with rollback capability
- Template effectiveness tracking
- A/B testing for template improvements
- **Error Handling & Resilience**
- Automatic retry with exponential backoff
- Multiple selector fallback strategies
- Site structure change detection
- Admin alerts for failing templates
- Graceful degradation to manual import

Data Import Workflow

1. **Source Selection:** Choose website or upload file
2. **Authentication:** Login to protected sites (if needed)
3. **Data Preview:** Review extracted data before import
4. **Field Mapping:** Map source fields to TeamSync fields
5. **Conflict Resolution:** Handle duplicate entries
6. **Import Execution:** Bulk import with progress tracking
7. **Verification:** Review imported data and fix errors

Compliance & Ethics

- **Respect robots.txt** directives
- **Rate limiting** to avoid overloading target sites
- **User authentication** for accessing protected content
- **Terms of service** compliance monitoring
- **Data privacy** considerations for scraped content
- **Transparent data sourcing** in UI

Priority: Critical - Core differentiator and time-saver

Phase 2+ Features (Future Development)

2.1 Real-Time Chat Functionality

Status: Deferred
Rationale: Real-time chat requires WebSocket infrastructure, increased server costs, and moderation systems. While valuable, it’s not essential for MVP launch. Phase 1 announcement board covers basic communication needs.

Planned Features:
- Team-wide chat rooms
- Direct messaging between users
- Group chats (coaches, parents, players)
- File sharing in chat
- Message reactions and threading
- Read receipts
- Online status indicators
- Chat history and search
- Message notifications

Technical Requirements:
- WebSocket server implementation
- Real-time database subscriptions
- Message queue system
- Moderation tools and content filtering
- Mobile push notification infrastructure

Estimated Timeline: Phase 2, Q2 2026

2.2 Advanced AI Strategy Automation

Status: Deferred
Rationale: Complex AI models require significant training data, infrastructure, and expertise. Phase 1 focuses on simpler AI features using external APIs. Advanced features need proven user base first.

Planned Features:
- **AI Game Strategy Assistant**
- Analyze opponent patterns
- Suggest formations and tactics
- Game plan generation

- **Automated Practice Plan Generator**
- Create custom practice plans based on team needs
- Adaptive difficulty
- Skill progression tracking

- **Player Matchup Optimizer**

- Optimal lineup suggestions
- Position recommendations
- Substitution strategy

- **Injury Risk Prediction**

- Analyze workload and fatigue
- Preventive recommendations
- Rest day suggestions

- **Recruitment Assistant**

- Player scouting insights
- Skill gap analysis
- Talent identification

Technical Requirements:
- Custom ML model training
- Large dataset collection
- GPU infrastructure for inference
- Model versioning and A/B testing
- Sport-specific model variants

Estimated Timeline: Phase 3, Q4 2026

2.3 Video Conferencing Integration

Status: Deferred
Rationale: Video conferencing is resource-intensive and adds complexity. Most teams already use Zoom/Google Meet. Integration can wait until user demand is validated.

Planned Features:
- In-app video calls
- Screen sharing
- Virtual team meetings
- Recording capabilities
- Integration with existing providers (Zoom, Google Meet)
- Virtual training sessions
- One-on-one coach-player meetings

Technical Requirements:
- WebRTC implementation or third-party SDK
- Bandwidth optimization
- Recording storage infrastructure
- Compliance with privacy regulations

Estimated Timeline: Phase 2-3, Q3 2026

2.4 Native Mobile Apps

Status: Deferred
Rationale: Mobile-responsive web app covers initial mobile needs. Native apps require separate development effort and maintenance. Wait for user growth to justify investment.

Planned Features:
- **iOS App**
- Native Swift/SwiftUI development
- Apple Push Notifications
- Face ID/Touch ID authentication
- Offline mode
- Home screen widgets

- **Android App**
- Native Kotlin development
- Firebase Cloud Messaging
- Biometric authentication
- Offline mode
- Home screen widgets

Technical Requirements:
- Mobile development team
- App store accounts and compliance
- Mobile-specific backend optimizations
- Cross-platform feature parity
- Mobile analytics and crash reporting

Estimated Timeline: Phase 3, Q1 2027

2.5 Additional Third-Party Integrations

Status: Deferred
Rationale: Focus on core integrations first. Additional integrations depend on user requests and usage patterns.

Planned Integrations:
- **Communication Platforms**
- Slack integration
- Microsoft Teams integration
- WhatsApp notifications

- **Payment Processing**
- PayPal integration
- Venmo for peer payments
- Subscription management

- **Sports-Specific Tools**

- Hudl (video analysis)
- TeamSnap import/export
- League management systems

- **Fitness & Health**

- Fitbit/Apple Health integration
- Nutrition tracking

- Sleep pattern analysis

- **Social Media**

- Instagram integration for team photos
- Twitter/X for announcements
- Facebook Events sync

Estimated Timeline: Ongoing from Phase 2+

2.6 Advanced Analytics & Reporting

Status: Deferred
Rationale: Basic analytics in Phase 1 provide foundation. Advanced features require more data and complex algorithms.

Planned Features:
- **Predictive Analytics**
- Season outcome predictions
- Player development projections
- Retention risk modeling

- **Comparative Analytics**
- Benchmark against other teams
- League-wide statistics
- Historical comparisons

- **Custom Report Builder**

- Drag-and-drop report designer
- Scheduled report generation
- White-label reports

- **Data Visualization**

- Heat maps for performance
- Interactive dashboards
- Drill-down capabilities

- **Export & API**

- Data export to BI tools
- Public API for data access
- Webhook notifications

Estimated Timeline: Phase 2-3, Q3 2026

2.7 Team-to-Team Communication Features

Status: Deferred
Rationale: Focus on single-team management first. Multi-team features require different architecture and moderation.

Planned Features:
- Inter-team messaging
- Game coordination (scheduling, location confirmation)
- Tournament organization
- League-wide announcements
- Team discovery and networking
- Scrimmage scheduling
- Resource sharing between teams
- Multi-team events

Technical Requirements:
- Multi-tenancy enhancements
- Cross-team permissions system
- Enhanced moderation tools
- League/organization tier accounts

Estimated Timeline: Phase 4, Q2 2027

2.8 Additional Future Considerations

Gamification
- Achievement badges
- Leaderboards
- Challenges and goals
- Reward system

Parent Portal Enhancements
- Carpool coordination
- Volunteer management
- Fundraising tools
- Team store integration

Medical & Health Tracking
- Injury logging
- Treatment plans
- Return-to-play protocols
- Growth and development tracking

Financial Management
- Budget tracking
- Expense reporting
- Fundraising campaign management
- Treasurer dashboard

Estimated Timeline: Phase 3-4, 2027+

Technical Architecture Decisions

3.1 Frontend: Next.js (React Framework)

Decision: Use Next.js 14+ with App Router

Rationale:
- **Server-Side Rendering (SSR):** Improves SEO and initial page load performance
- **Static Site Generation (SSG):** Pre-render pages for fast delivery
- **API Routes:** Built-in backend capabilities for simple integrations
- **React Ecosystem:** Large community, extensive component libraries
- **TypeScript Support:** Type safety reduces bugs
- **Developer Experience:** Hot reloading, excellent tooling

Migration Enablement:
- Component-based architecture allows gradual refactoring
- Can deploy to various hosting platforms (Vercel, AWS, self-hosted)
- Serverless-ready for future scaling
- Easy to add CDN for global distribution

Technology Stack:
- Next.js 14+
- TypeScript
- Tailwind CSS for styling
- Shadcn/ui for component library
- React Hook Form for form management
- Zustand or React Context for state management
- React Query for data fetching and caching

3.2 Backend: Supabase (PostgreSQL Database & Backend Services)

Decision: Use Supabase as Backend-as-a-Service (BaaS)

Rationale:
- **PostgreSQL Database:** Robust, reliable, and scalable relational database
- **Built-in Authentication:** Reduces development time, secure by default
- **Real-time Subscriptions:** Foundation for future real-time features
- **Row-Level Security (RLS):** Database-level access control
- **Storage:** File upload and management built-in
- **RESTful APIs:** Auto-generated API from database schema
- **Free Tier:** Cost-effective for MVP
- **Open Source:** Can self-host if needed

Migration Enablement:
- Standard PostgreSQL database - can migrate to self-hosted or managed Postgres
- Compatible with Prisma ORM for future flexibility
- REST API can be replaced with custom backend gradually
- Authentication can migrate to custom solution or Auth0/Clerk
- Data export capabilities for migration

Schema Design Principles:
- Normalized relational design
- Foreign key constraints for data integrity
- Indexes on frequently queried columns
- JSON columns for flexible metadata
- Audit trails (created_at, updated_at)

3.3 AI Integration: External API Approach

Decision: Use external AI APIs (OpenAI, Anthropic) via DeepAgent initially

Rationale:
- **Fast Development:** No need to train custom models
- **Cost-Effective:** Pay-per-use pricing
- **State-of-the-Art:** Access to latest AI capabilities
- **Flexibility:** Easy to switch providers or add custom models later
- **Low Maintenance:** No ML infrastructure to manage

Migration Enablement:
- API abstraction layer allows provider switching
- Can gradually introduce custom models for specific use cases
- Collect data for future model training
- Hybrid approach: external APIs + custom models

AI Architecture:
- Wrapper service for AI API calls
- Caching layer to reduce costs
- Rate limiting to control usage
- Fallback mechanisms for API failures
- Usage tracking and cost monitoring

3.4 REST API Architecture

Decision: REST-based architecture for Phase 1

Rationale:
- **Simplicity:** Easy to understand and implement
- **Compatibility:** Works with any frontend technology
- **Caching:** HTTP caching mechanisms well-established
- **Tooling:** Extensive tooling and documentation
- **Supabase Native:** Leverage Supabase auto-generated REST APIs

Migration Enablement:
- Can add GraphQL layer later for complex queries
- Transition to microservices architecture when needed
- API versioning strategy allows gradual changes
- Swagger/OpenAPI documentation for API contracts

API Design Principles:
- RESTful resource naming conventions

- Consistent error handling
- Pagination for list endpoints
- Filtering and sorting capabilities
- Rate limiting
- API versioning (v1, v2)

3.5 Deployment & Hosting

Decision: Vercel for frontend, Supabase Cloud for backend

Rationale:
- **Ease of Deployment:** Git-based deployments
- **Zero Configuration:** Automatic builds and deployments
- **Global CDN:** Fast content delivery worldwide
- **Serverless Functions:** Auto-scaling for API routes
- **Preview Deployments:** Test changes before production
- **Free Tier:** Cost-effective for MVP

Migration Enablement:
- Can move to AWS/GCP/Azure when scaling requires it
- Docker containers ready for any hosting platform
- Environment variable management for easy configuration
- CI/CD pipeline can be replicated anywhere

Future Scaling Path:
- Move to AWS ECS/EKS for containerized deployment
- Use AWS RDS or self-hosted PostgreSQL
- Add Redis for caching and session management
- Implement load balancers for high availability
- Use CloudFront or Cloudflare CDN

3.6 Web Scraping Infrastructure

Decision: Template-based scraping system with fallback strategies

Rationale:
- **Automated Data Import:** Reduce manual data entry for coaches
- **Multi-Source Support:** Import from various sports platforms
- **Template Reusability:** One template serves all users of a site
- **Maintainability:** Easy updates when sites change structure
- **Extensibility:** Community can contribute new site templates

Technology Stack:
- **Headless Browser:** Puppeteer or Playwright for JavaScript-heavy sites
- **HTML Parser:** Cheerio for static content parsing
- **Selector Engine:** CSS selectors and XPath for element targeting
- **Template Storage:** PostgreSQL with JSON columns for template definitions
- **Queue System:** Background job processing for long-running scrapes
- **Caching:** Redis for rate limiting and temporary data storage

Architecture Components:

```
┌─────────────────────────────────────────────┐
│  User Interface (Template Management)       │
├─────────────────────────────────────────────┤
│  Scraping Orchestrator                      │
│  - Job scheduling & queueing                │
│  - Site adapter selection                   │
│  - Error handling & retries                 │
├─────────────────────────────────────────────┤
│  Site Adapters (Pluggable)                  │
│  - GameChanger Adapter                      │
│  - QuickScores Adapter                      │
│  - Generic Site Adapter                     │
├─────────────────────────────────────────────┤
│  Extraction Engine                          │
│  - Template interpreter                     │
│  - Selector execution                       │
│  - Data normalization                       │
├─────────────────────────────────────────────┤
│  Template Repository                        │
│  - Template CRUD operations                 │
│  - Version management                       │
│  - Effectiveness tracking                   │
├─────────────────────────────────────────────┤
│  Database (Templates + Scraped Data)        │
└─────────────────────────────────────────────┘
```

Migration Enablement:
- Can migrate to dedicated scraping service when scale requires
- Template format is portable (JSON-based)
- Easy to add proxy rotation for large-scale scraping
- Can integrate with commercial scraping APIs (ScrapingBee, Apify)
- Modular design allows replacement of components

Security & Compliance:
- User credentials encrypted for accessing protected sites
- Rate limiting per site to be respectful
- Robots.txt compliance checking
- User-agent identification as TeamSync AI
- GDPR compliance for scraped personal data
- Audit logging for all scraping activities

3.7 Development Workflow

Version Control: Git with GitHub
Branching Strategy: Git Flow (main, develop, feature branches)
Code Review: Pull request reviews required
Testing Strategy:
- Unit tests (Jest, React Testing Library)
- Integration tests (Playwright/Cypress)
- E2E tests for critical user flows
- Scraping template validation tests

CI/CD: GitHub Actions for automated testing and deployment
Monitoring: Vercel Analytics, Supabase Dashboard, Sentry for error tracking

---

Integration Strategy

4.1 Integration Architecture

Approach: Modular integration layer with adapter pattern

Principles:
- **Abstraction:** Create interface layer for each integration
- **Loose Coupling:** Integrations should not affect core functionality
- **Error Handling:** Graceful degradation if integration fails
- **Configuration:** Environment-based settings for easy switching
- **Testing:** Mock integrations for development and testing

4.2 Google Calendar Integration

Implementation:
- Use Google Calendar API v3
- OAuth 2.0 authentication flow
- Server-side token management
- Webhook notifications for calendar changes

Features:
- Export TeamSync events to user’s Google Calendar
- Subscribe to team calendar via iCal URL
- Sync event updates bidirectionally
- Respect user’s calendar preferences (timezone, reminders)

Migration Considerations:
- Support for other calendar services (Outlook, Apple Calendar)
- Self-hosted CalDAV server option
- Webhook infrastructure for real-time sync

Error Handling:
- Retry mechanism for failed syncs
- User notification on sync failures
- Manual sync option
- Fallback to iCal export

4.3 Stripe Payment Integration

Implementation:
- Stripe Checkout for payment collection
- Stripe Payment Links for quick setup
- Webhook handling for payment events
- Dashboard for payment tracking

Features:
- Generate payment links for fees and events
- Track payment status
- Send confirmation emails
- Refund processing
- Payment history

Migration Considerations:
- Multi-gateway support (PayPal, Square)
- Subscription billing in future phases
- Invoice generation
- Accounting software integration (QuickBooks)

Security:
- PCI compliance through Stripe
- No card data stored in our database
- Secure webhook signature verification
- Environment-specific API keys

4.4 Email Notification System

Implementation:
- Email service provider: SendGrid or AWS SES
- Template-based email generation
- Scheduled email queue
- Unsubscribe management

Email Types:
- Transactional (signup, password reset)
- Event notifications (new events, reminders, changes)
- Announcements (new posts, urgent updates)
- Digests (weekly summary)
- System alerts (low attendance, inactive users)

Migration Considerations:
- Switch email providers easily
- Self-hosted email server option
- Multi-channel notifications (SMS, push) in future
- Email tracking and analytics

Best Practices:
- Plain text and HTML versions
- Mobile-responsive templates
- Clear unsubscribe links
- Personalization tokens
- A/B testing capabilities

4.5 Sports Data Integration (Web Scraping & APIs)

Implementation Strategy: Hybrid approach combining official APIs and intelligent web scraping

GameChanger Integration (Web Scraping Primary)

Approach: Template-based web scraping with intelligent pattern recognition

Implementation:
- Use Playwright/Puppeteer for dynamic content rendering
- Create and maintain GameChanger-specific scraping templates
- Handle authentication for team-specific data access
- Parse team pages, rosters, schedules, and game results
- Extract player statistics and performance data

Data Extracted:
- Team roster with player details
- Game schedules (date, time, location, opponent)
- Live and historical scores
- Player statistics (at-bats, hits, runs, etc. for baseball/softball)
- Season standings and records

Challenges & Solutions:
- **Challenge:** Site structure changes break scrapers
- **Solution:** Template versioning, multiple selector fallbacks, admin alerts
- **Challenge:** Rate limiting and bot detection
- **Solution:** Respectful scraping, delays between requests, user-agent rotation
- **Challenge:** Authentication required for team data
- **Solution:** Secure credential storage, OAuth if available, session management

Update Frequency:
- Real-time: During active games (if requested by coach)
- Scheduled: Daily sync for schedules and scores
- On-demand: Manual refresh button for immediate updates

QuickScores Integration (API Primary, Scraping Backup)

Approach: Official API first, web scraping as fallback

Implementation:
- Primary: QuickScores REST API integration
- API key management
- Standard API calls for data retrieval
- Webhook support for real-time updates (if available)
- Backup: Web scraping fallback
- Activate if API is down or unavailable
- Same template system as GameChanger
- Seamless transition between API and scraping

Data Retrieved:
- Organization and team information
- Game schedules and results
- Tournament brackets and standings
- Referee assignments
- Field/facility information

API Features:
- Official documentation compliance
- Error handling with fallback to scraping
- Rate limit management
- Cached responses to reduce API calls
- Automatic retry logic

Migration to API:
- Monitor QuickScores API availability
- A/B test API vs scraping reliability
- Gradual migration as API proves stable
- Keep scraping capability for backup

Extensible Site Support

Architecture: Plugin-based adapter system for new sites

Adding New Sports Sites:

1. **Site Analysis:**
   - Identify data structure and patterns
   - Test authentication requirements
   - Map data fields to TeamSync schema
   - Document API availability

2. **Adapter Creation:**
   - Implement site-specific adapter class
   - Define scraping templates or API integration
   - Create data transformation logic
   - Add error handling specific to site

3. **Template Definition:**
   - JSON-based template structure
   - CSS selectors or XPath expressions
   - Field mapping configuration
   - Validation rules

4. **Testing & Deployment:**
   - Test on multiple team pages
   - Validate data accuracy
   - Performance testing
   - Deploy to template repository

Example Template Structure:

```json
{
  "site": "gamechanger",
  "version": "1.0",
  "selectors": {
    "team_name": ".team-header h1",
    "roster": {
      "container": ".roster-table tbody",
      "player": "tr",
      "fields": {
        "name": ".player-name",
        "number": ".jersey-number",
        "position": ".position"
      }
    },
    "schedule": {
      "container": ".schedule-list",
      "game": ".game-item",
      "fields": {
        "date": ".game-date",
        "opponent": ".opponent-name",
        "location": ".game-location"
      }
    }
  },
  "authentication": {
    "required": true,
    "type": "login_form",
    "selectors": {
      "username": "#email",
      "password": "#password",
      "submit": "button[type='submit']"
    }
  }
}
```

Community Contributions:
- Template submission system
- Review and approval process
- Template marketplace
- Version control and rollback
- Effectiveness ratings from users

4.6 Future Integration Framework

Plugin Architecture:
- Define standard integration interface
- Registry of available integrations
- Enable/disable integrations per team
- Custom integration marketplace (future)

Integration Categories:
- Communication (Slack, Teams, Discord)
- Payments (PayPal, Venmo, Stripe)
- Calendar (Google, Outlook, Apple)
- Sports Tools (Hudl, TeamSnap, MaxPreps, SportsEngine)

- Analytics (Google Analytics, Mixpanel)
- Storage (Google Drive, Dropbox)

AI Capabilities Roadmap

5.1 Phase 1: Foundational AI Features

Timeline: Q4 2025 - Q1 2026

Schedule Optimization Engine

Technology: OpenAI GPT-4 or Claude API via DeepAgent

Capabilities:
- Analyze historical attendance data
- Consider team member availability patterns
- Factor in location and travel time
- Incorporate weather data
- Suggest optimal time slots for practices and events

Implementation:
- Collect anonymized scheduling data
- Build prompt engineering templates
- Create scoring algorithm for suggestions
- User feedback loop for improvement

Success Metrics:
- User adoption rate of AI suggestions
- Improvement in average attendance
- Time saved in scheduling

Performance Insights Generator

Technology: Statistical analysis + LLM for natural language insights

Capabilities:
- Aggregate performance data (attendance, stats, participation)
- Identify trends and patterns
- Compare individual performance to team averages
- Generate natural language summaries
- Highlight strengths and areas for improvement

Implementation:
- Define sport-specific performance metrics
- Create data aggregation pipelines
- LLM prompts for insight generation
- Visualization of insights

Success Metrics:
- Engagement with insights dashboard

- Perceived value by coaches
- Correlation with team performance

Attendance Prediction Model

Technology: Simple ML model (regression) + historical data

Capabilities:
- Predict attendance for upcoming events
- Identify at-risk players (low engagement)
- Seasonal pattern detection
- Early warning system

Implementation:
- Collect historical attendance data
- Feature engineering (day of week, time, weather, player history)
- Train simple scikit-learn model
- Update predictions regularly
- Threshold-based alerts

Success Metrics:
- Prediction accuracy
- Early intervention success rate
- Coach satisfaction with predictions

5.2 Phase 2: Enhanced AI Capabilities

Timeline: Q2 - Q3 2026

Natural Language Query Interface

- Ask questions about team data in plain English
- “Show me players with declining attendance this month”
- “What’s the optimal practice time for maximum attendance?”
- Generate custom reports via conversation

Automated Communication Assistant

- Draft announcements based on templates
- Personalize messages for different audiences
- Suggest communication timing
- Grammar and tone checking

Smart Event Recommendations

- Suggest practice drills based on team weaknesses
- Recommend game strategies based on opponent analysis
- Propose team-building activities

Intelligent Form Generation

- Auto-generate registration forms
- Create permission slips

- Medical form templates
- Custom surveys

5.3 Phase 3: Advanced AI & ML

Timeline: Q4 2026 - 2027

Custom Model Training

- Train sport-specific models on accumulated data
- Player development trajectory prediction
- Team chemistry analysis
- Win probability models

Computer Vision Integration

- Analyze game footage for insights
- Automatic highlight reel generation
- Technique analysis
- Formation recognition

Voice Interface

- Voice commands for scheduling
- Dictate announcements
- Voice-based attendance taking
- Accessibility features

Recommendation Engine

- Personalized training recommendations
- Equipment suggestions
- Nutrition and wellness tips
- Similar team connections

5.4 AI Ethics & Safety

Principles:
- **Privacy First:** No personal data used for model training without consent
- **Transparency:** Clear indication when AI is providing suggestions
- **Human Oversight:** Coaches always make final decisions
- **Bias Mitigation:** Regular audits of AI recommendations
- **Data Security:** Encrypted storage, access controls
- **Youth Protection:** COPPA compliance, parental controls

Monitoring:
- Track AI suggestion acceptance rates
- User feedback on AI features
- Regular bias audits
- Privacy impact assessments

Migration Path: DeepAgent to Scalable Infrastructure

6.1 Current State: DeepAgent Integration (Phase 1)

What is DeepAgent:
- Abacus.AI’s AI agent framework for rapid development
- Provides pre-built AI capabilities and workflows
- Simplifies integration with LLMs and AI models
- Development accelerator for MVP

Usage in Phase 1:
- AI prompt management and execution
- Integration with OpenAI/Anthropic APIs
- Quick prototyping of AI features
- Cost management and tracking

Benefits:
- Faster time to market
- Reduced AI infrastructure complexity
- Lower initial development costs
- Focus on core application features

6.2 Migration Triggers

When to migrate from DeepAgent:

1. **Scale Triggers:**
   - Exceeding 10,000 active users
   - AI API costs exceed $5,000/month
   - Response time requirements below DeepAgent capabilities
   - Need for custom model deployment

2. **Feature Triggers:**
   - Require custom ML model training
   - Need real-time AI inference with low latency
   - Want to deploy proprietary AI models
   - Require on-premise AI deployment for data privacy

3. **Cost Triggers:**
   - External AI API costs become prohibitive
   - Need to optimize for specific use cases
   - ROI justifies infrastructure investment

6.3 Migration Strategy: Phased Approach

Stage 1: Abstraction Layer (Q2 2026)

Goal: Create independence from DeepAgent without migration

Actions:
- Build AI service abstraction layer

- Create interface for AI operations
- Implement adapter pattern for DeepAgent
- Document all AI API calls
- Add switching logic for multiple providers

Benefits:
- Can switch providers without code changes
- A/B test different AI providers
- Reduce vendor lock-in
- Prepare for future migration

Code Example:

```
interface AIService {
  generateInsight(data: PerformanceData): Promise<Insight>;
  optimizeSchedule(constraints: ScheduleConstraints): Promise<Schedule>;
  predictAttendance(event: Event): Promise<AttendancePrediction>;
}

class DeepAgentService implements AIService {
  // DeepAgent implementation
}

class OpenAIService implements AIService {
  // Direct OpenAI implementation
}

class CustomMLService implements AIService {
  // Custom model implementation
}
```

Stage 2: Parallel Implementation (Q3 2026)

Goal: Run DeepAgent and custom solution side-by-side

Actions:
- Implement direct API integrations (OpenAI, Anthropic)
- Deploy custom ML models for specific features
- Route percentage of traffic to new implementation
- Compare performance, cost, and quality
- Collect metrics for decision making

Benefits:
- Risk-free testing
- Gradual rollout
- Performance comparison
- Fallback to DeepAgent if issues arise

Monitoring:
- Response time comparison
- Cost per API call
- Quality of AI outputs

- Error rates
- User satisfaction

Stage 3: Custom AI Infrastructure (Q4 2026 - Q1 2027)

Goal: Full control over AI capabilities

Actions:
- Set up ML infrastructure (AWS SageMaker, GCP AI Platform, or self-hosted)
- Deploy custom models for core features:
- Attendance prediction model
- Performance analysis model
- Schedule optimization algorithm
- Implement model versioning and A/B testing
- Create model training pipelines
- Build monitoring and alerting systems

Technology Stack:
- **Model Training:** Python, PyTorch/TensorFlow, scikit-learn
- **Model Serving:** TensorFlow Serving, TorchServe, or custom FastAPI
- **Infrastructure:** Kubernetes for orchestration, Docker containers
- **Monitoring:** Prometheus, Grafana, custom dashboards
- **Data Pipeline:** Apache Airflow for orchestration
- **Feature Store:** Feast or custom solution

Benefits:
- Lower per-request costs at scale
- Faster response times
- Custom models tailored to our data
- Full data control and privacy
- Ability to run offline/on-premise

Stage 4: Complete Migration (Q2 2027)

Goal: Independent AI infrastructure

Actions:
- Migrate all AI features to custom infrastructure
- Deprecate DeepAgent integration
- Optimize models for performance and cost
- Implement continuous learning pipelines
- Build internal AI expertise team

Long-term AI Architecture:

```
┌──────────────────────────────────────────────┐
│           TeamSync AI Platform               │
├──────────────────────────────────────────────┤
│  Frontend (Next.js) → API Layer (REST)       │
├──────────────────────────────────────────────┤
│  AI Service Layer (Abstraction)              │
├──────────────┬───────────────────────────────┤
│  LLM APIs    │  Custom ML Models             │
│  (OpenAI,    │  (Attendance, Performance,    │
│   Anthropic) │   Schedule Optimization)      │
├──────────────┴───────────────────────────────┤
│  Model Training Pipeline                     │
│  (Data Collection → Training → Deployment)   │
├──────────────────────────────────────────────┤
│  Database (PostgreSQL) + Data Warehouse      │
└──────────────────────────────────────────────┘
```

6.4 Database Migration Strategy

Current: Supabase (Managed PostgreSQL)

Future Options:

Option 1: Self-Hosted PostgreSQL

When: Need full database control, cost optimization at scale

Migration Path:
1. Export Supabase database schema and data
2. Set up self-hosted PostgreSQL (AWS RDS, GCP Cloud SQL, or EC2/GCE)
3. Implement database replication for zero-downtime migration
4. Switch connection strings
5. Monitor for issues, rollback if needed

Benefits:
- Full control over configurations
- Cost savings at scale
- Custom extensions and optimizations

Challenges:
- Need database administration expertise
- Responsible for backups and maintenance
- Need to implement own authentication if leaving Supabase

Option 2: Hybrid Approach

When: Want some benefits of both

Strategy:
- Keep Supabase for authentication and real-time features
- Move bulk data to separate PostgreSQL instance
- Use Supabase as API layer, custom database for storage

Option 3: Distributed Database

When: Global scale, millions of users

Technologies:
- CockroachDB (PostgreSQL-compatible, distributed)
- Google Cloud Spanner
- Amazon Aurora Global Database

Migration Path:
- Usually after reaching significant scale (100k+ users)
- Requires major architectural changes
- Implement database sharding strategy

6.5 Frontend & Backend Migration

Current: Next.js on Vercel, Supabase backend

Scalability Path:

Stage 1: Keep Current Architecture (0-50k users)

- Vercel scales automatically
- Supabase handles moderate load
- Cost-effective and simple

Stage 2: Optimize on Current Platform (50k-200k users)

- Implement CDN for static assets
- Add Redis caching layer
- Optimize database queries
- Use Vercel Edge Functions for performance
- Implement database read replicas

Stage 3: Hybrid Cloud (200k-500k users)

- Move API backend to custom Node.js/Go server
- Deploy on AWS/GCP with load balancers
- Keep Next.js frontend on Vercel
- Implement API gateway
- Add message queue (RabbitMQ, AWS SQS)

Stage 4: Microservices (500k+ users)

- Break monolith into services:
- User service
- Team service
- Scheduling service
- Notification service
- Analytics service
- AI service
- Use Kubernetes for orchestration
- Implement service mesh (Istio)

- Event-driven architecture
- CQRS pattern for read/write optimization

6.6 Migration Principles

Key Principles to Guide Migration:

1. **Data-Driven Decisions:** Migrate based on metrics, not assumptions
2. **Incremental Changes:** Small, reversible steps
3. **Maintain Compatibility:** No user-facing disruptions
4. **Test Extensively:** Staging environments mirror production
5. **Monitor Everything:** Metrics before, during, and after migration
6. **Plan Rollbacks:** Always have a way back
7. **Document Thoroughly:** Future team members need context
8. **Cost-Benefit Analysis:** Ensure migration provides value

6.7 Risk Mitigation

Risks and Mitigation Strategies:

Risk	Impact	Mitigation
Data loss during migration	Critical	Backups, replication, testing
Downtime affecting users	High	Blue-green deployments, off-peak migration
Performance degradation	High	Load testing, gradual rollout, monitoring
Cost overruns	Medium	Budget planning, PoC testing, monitoring
Team expertise gaps	Medium	Training, consulting, phased migration
Vendor lock-in	Low	Abstraction layers, standard technologies

Timeline & Milestones

Q4 2025: MVP Development

- **November:** Core features development (roster, scheduling, auth)
- **December:** UI/UX implementation, integrations, testing

- **January 2026:** Beta testing, bug fixes, polish

Milestone: MVP Launch with Phase 1 features

Q1 2026: MVP Launch & Iteration

- **January:** Soft launch to pilot teams
- **February:** Gather feedback, iterate on features
- **March:** Public launch marketing push

Milestone: 100 active teams using TeamSync AI

Q2 2026: Optimization & Phase 2 Planning

- **April-June:**
- Optimize performance based on usage data
- Implement AI abstraction layer
- Plan Phase 2 features
- Begin work on high-priority Phase 2 items

Milestone: 500 active teams, AI abstraction layer complete

Q3 2026: Phase 2 Development

- **July-September:**
- Advanced analytics features
- Enhanced AI capabilities
- Additional integrations based on user requests
- Parallel AI infrastructure testing

Milestone: 1,000 active teams, Phase 2 features launched

Q4 2026: Scaling & Custom AI

- **October-December:**
- Custom ML model deployment
- Performance optimizations
- Begin real-time features development
- Infrastructure scaling preparations

Milestone: 2,500 active teams, custom AI models in production

---

2027 and Beyond: Advanced Features & Scale

- Mobile app development
- Video conferencing integration

- Team-to-team features
- Full AI infrastructure migration
- International expansion
- Enterprise features for leagues and organizations

Milestone: 10,000+ active teams, sustainable revenue model

Success Metrics

Product Metrics

- Monthly Active Users (MAU)
- Daily Active Users (DAU)
- Number of active teams
- Events created per team
- Announcement engagement rate
- File downloads
- AI feature usage rate

Business Metrics

- Customer Acquisition Cost (CAC)
- Lifetime Value (LTV)
- Monthly Recurring Revenue (MRR)
- Churn rate
- Net Promoter Score (NPS)

Technical Metrics

- Page load time
- API response time
- Error rate
- Uptime percentage
- Database query performance
- AI API cost per request

Appendix

A. Technology Stack Summary

Frontend:
- Next.js 14+ (React framework)
- TypeScript
- Tailwind CSS
- Shadcn/ui components
- React Hook Form
- Zustand for state management

Backend:
- Supabase (PostgreSQL, Auth, Storage, Real-time)
- Node.js serverless functions
- REST API architecture

Web Scraping:
- Puppeteer/Playwright (headless browser for dynamic sites)
- Cheerio (HTML parsing for static sites)
- CSS Selectors & XPath (element targeting)
- Redis (rate limiting and caching)
- Background job queue (BullMQ or Supabase Edge Functions)

AI/ML:
- DeepAgent (Abacus.AI) - Phase 1
- OpenAI API
- Anthropic Claude API
- Custom ML models - Future phases

Integrations:
- Google Calendar API
- Stripe API
- SendGrid/AWS SES for emails
- GameChanger (web scraping)
- QuickScores (API + scraping fallback)

Deployment:
- Vercel (frontend)
- Supabase Cloud (backend)
- GitHub (version control)
- GitHub Actions (CI/CD)

Monitoring & Analytics:
- Vercel Analytics
- Sentry (error tracking)
- Google Analytics
- Custom logging
- Scraping success/failure tracking

B. Resource Links

Documentation:
- Next.js: https://nextjs.org/docs
- Supabase: https://supabase.io/docs
- Abacus.AI: https://abacus.ai/docs
- Stripe: https://stripe.com/docs
- Google Calendar API: https://developers.google.com/calendar

Design Resources:
- Figma design files: [TBD]
- Brand guidelines: [TBD]
- Component library: [TBD]

Project Management:
- GitHub repository: [TBD]
- Project board: [TBD]
- Slack channel: [TBD]

C. Glossary

- **MVP:** Minimum Viable Product - Initial version with core features
- **BaaS:** Backend as a Service - Cloud backend services
- **RLS:** Row Level Security - Database security mechanism
- **SSR:** Server-Side Rendering - Rendering HTML on server
- **API:** Application Programming Interface
- **REST:** Representational State Transfer
- **LLM:** Large Language Model
- **ML:** Machine Learning
- **CI/CD:** Continuous Integration/Continuous Deployment
- **COPPA:** Children's Online Privacy Protection Act

D. Contact & Ownership

Project Owner: [Your Name]
Technical Lead: [TBD]
Product Manager: [TBD]
Last Updated: November 8, 2025

Document Version History:

Version	Date	Changes	Author
1.0	Nov 8, 2025	Initial roadmap document	DeepAgent
1.1	Nov 8, 2025	Added Intelligent Web Scraping & Data Import System (Phase 1.11) - Template-based learning approach - GameChanger and QuickScores integration - Template management interface - Updated Technical Architecture (Section 3.6) - Updated Integration Strategy (Section 4.5) - Updated Technology Stack Summary	DeepAgent
1.2	Nov 8, 2025	Added Visual Drill Planning & Field Configurator (Phase 1.10) - Sport-specific field templates with dynamic sizing - Drag-and-drop drill placement with multi-drill support - Practice plan builder with configuration switching - Interactive field visualization with export options - Canvas-based rendering for tablet use	DeepAgent
1.3	Nov 8, 2025	Enhanced Analytics Dashboard with Player Position Tracking (Phase 1.4) - Visual field heat maps showing time	DeepAgent

Version	Date	Changes	Author
		spent by position - Position time tracking with timestamps and historical data - Parent & player portal with permission-based sharing - Position distribution analytics and insights - Multiple position entry methods (live tracking, CSV import) - Integration with drill planning system	

This roadmap is a living document and will be updated as the project evolves. All stakeholders should refer to this document for project direction and scope decisions.