## WicketWise Product Requirements Document (PRD)

**Version:** 1.0  
**Updated:** July 2025  
**Team:** Phi 1.618 Engineering  
**Status:** Active Development

## 1. Executive Summary

**Vision**  
WicketWise is a next-generation cricket analytics platform built on advanced AI, real-time data, and deep domain knowledge. It combines transformer models, knowledge graphs, and explainable GNNs to predict outcomes and reveal tactical insights in T20 cricket.

**Core Value Proposition**

* Real-time, ball-by-ball prediction
* Transformer + GNN fusion for deep reasoning
* Multi-hop knowledge graph for contextual awareness
* Explainable AI via GNNExplainer + SHAP
* Enterprise APIs and tactical visualizations

**Target Users**

* Primary: Cricket analysts, broadcasters, betting operators
* Secondary: Coaches, journalists, data scientists
* Tertiary: Fans, researchers, sports tech developers

## 2. Product Architecture Overview

**System Components**

Data Ingestion --> Match Aligner & Cleaner --> Knowledge Graph  
 \--> Feature Generator  
 \--> Multi-hop GNN Embedding Engine  
 \--> Crickformer Transformer Model  
 \--> Inference Engine --> APIs/UI/Chatbot

**Technology Stack**

* Python 3.11+, PyTorch 2.7+, PyTorch Geometric
* Streamlit, React (planned), D3.js
* Redis, Neo4j, PostgreSQL, Docker
* OpenAI GPT-4o, MC Dropout, GNNExplainer

## 3. Core Functionality

### 3.1 Data Processing

* Ingests CSV/JSON/NVPlay/Decimal formats
* Aligns and deduplicates overlapping match feeds using fuzzy + LLM matching
* Validates against schema and fills gaps using fallback rules

### 3.2 Knowledge Graph Engine

* Multi-hop heterogeneous graph using PyG/NetworkX
* Node Types: player, team, match, venue, phase, event, bowler\_type
* Edge Types: faced, dismissed\_by, plays\_for, partnered\_with, excels\_against, etc.
* Edge attributes include runs, wickets, venue, phase, date, dismissal\_type
* Role and style embeddings added to node features (opener, death bowler, etc.)
* Graph updated in real time; message passing supports 1-hop to 3-hop reasoning
* Temporal decay: Learnable embedding + Time2Vec

### 3.3 ML Models

**CrickFormer** – Transformer model for ball-by-ball prediction

* Inputs: numerical + categorical + embeddings from GNN
* Architecture: 6-layer transformer, 128D embeddings, 300-ball sequence
* Outputs: next ball (0–6, wicket), over summary, innings score, match outcome

**GNN Architectures**

* GraphSAGE – Inductive and scalable
* GCN – Stable and spectral
* GATv2 – Attention-weighted influence detection
* Embeddings (128D) exported for player/venue/phase

### 3.4 Real-time Inference Engine

* Processes live matches with sub-500ms latency
* Monte Carlo dropout for confidence
* Feeds Streamlit dashboard, WebSocket, REST APIs

### 3.5 Explainability System

* GNNExplainer: Identify influential nodes/edges per prediction
* SHAP/LIME integration with CrickFormer
* Visual overlays + downloadable reports (SVG, PNG)

## 4. User Interface Design

### 4.1 Design Principles

* Phi 1.618 visual identity (modern, intelligent, focused)
* Responsive and intuitive UI for all devices
* Glassmorphic, high-contrast theme with cricket-specific colors

### 4.2 Key Components

* PlayerCard – Form, stats, GNN/Transformer signals
* WinProbabilityBar – Updated per ball, with confidence
* OddsPanel – Market vs model overlay
* TacticalChat – AI assistant using GPT-4o
* Match Simulator – Test custom scenarios

### 4.3 Navigation

* Sidebar with key modules: Live, Simulator, Player Analytics, Admin, History
* Sticky match header and phase context

## 5. Data Requirements

### 5.1 Inputs

* Ball-by-ball feed: match\_id, innings, over, ball, batter, bowler, runs, extras, dismissal\_type
* Player metadata: name, team, role, recent form, style vector
* Match context: weather, venue, phase, competition, pitch condition

### 5.2 Quality

* Critical Fields: >99% complete
* Important Fields: >95%
* Optional Fields: >70%
* Historical update lag: <24 hrs, Live delay: <5 sec

### 5.3 Storage

* Raw: CSV/JSON in S3-like object store
* Processed: Parquet files in warehouse
* Graph: Neo4j or HeteroData (PyG)
* Cache: Redis (1-hour TTL)

## 6. API Specifications

* GET /api/v1/matches/{match\_id}/predictions
* GET /api/v1/players/{player\_id}/analysis
* POST /api/v1/models/explain
* WebSocket: wss://api.wicketwise.ai/v1/stream/matches/{match\_id}

All APIs return predictions, confidence intervals, and metadata.

## 7. Performance Requirements

* Prediction: <500ms latency
* API: <200ms for cache, <1s uncached
* Model inference: <50ms
* Graph updates: <100ms
* Throughput: 1M balls/hr, 10k+ users

## 8. Security & Compliance

* OAuth 2.0, JWTs, MFA for admins
* GDPR/CCPA compliance
* Rate limiting and role-based access
* TLS 1.3 + AES-256 encryption
* Audit logging and backup recovery (RPO <1h)

## 9. Technical Specifications

### 9.1 GNN

gnn\_config = {  
 'input\_dim': 64,  
 'hidden\_dim': 128,  
 'output\_dim': 128,  
 'layers': 3,  
 'heads': 4,  
 'dropout': 0.1,  
 'lr': 0.001,  
 'device': 'auto-detect'  
}

### 9.2 CrickFormer

CrickFormerConfig = {  
 'sequence\_length': 300,  
 'embedding\_dim': 128,  
 'attention\_heads': 8,  
 'layers': 6,  
 'dropout': 0.1,  
 'numerical\_features': 45,  
 'categorical\_vocab': {  
 'batter': 5000,  
 'bowler': 3000,  
 'venue': 100,  
 'phase': 10,  
 'event': 10  
 }  
}

## 10. Development & Deployment

* Git workflow: feature/bugfix/hotfix branches
* CI/CD: Docker, GitHub Actions, PyTest, Playwright
* Blue-green deployments with rollbacks
* TorchServe for model hosting

## 11. Testing Strategy

* Unit Tests: 70% (models, utils)
* Integration: 20% (data → prediction)
* E2E: 10% (UI + APIs)
* Load: 10k concurrent users
* Stress: edge case inference spikes

## 12. Success Metrics

* Model accuracy (ball prediction): >85%
* Inference latency: <500ms
* API uptime: 99.9%
* User engagement (MAUs): 50k target
* NPS score: >50

## 13. Risk Assessment

* **Model drift**: Drift monitoring + auto-retraining
* **Infrastructure**: Multi-region failover + autoscaling
* **Security**: Regular audits, penetration testing
* **Team risks**: Cross-trained dev team, documented onboarding

## 14. Future Roadmap

### Short-term (3–6 months)

* Weather-aware embeddings
* Batting intent model (video signals)
* GATv2 for dynamic attention

### Medium-term (6–12 months)

* Counterfactual reasoning
* Causal inference GNNs
* Player fatigue + pressure models

### Long-term (12–24 months)

* Multi-sport platform: Football, basketball
* AR-based tactical overlays
* Plugin SDK + dev ecosystem

## 15. Conclusion

WicketWise is a comprehensive, AI-first platform for cricket intelligence. With its fusion of multi-hop knowledge graphs, real-time transformers, and explainability tooling, it brings a new level of depth, speed, and trust to cricket strategy and prediction.

**Next steps:**

* Finalize core modules
* Begin pilot deployments with analysts and broadcasters
* Extend model coverage across formats and geographies
* Launch WicketWise beta