Cricket with AI — Product Development Document

1. Product Summary & Success Metrics

Name: Cricket with AI

Tagline: Real-time cricket scoring + AI insights — schedule, score, analyze.

Primary users: Match scorers, tournament organizers, coaches, club managers, casual players.

Key value props:

- Accurate ball-by-ball scoring + shareable scorecards.

- Tournament & fixture management with automatic scheduling & NRR.

- AI features: prompt-based match scheduling, smart suggestions, predictive insights, natural-language match summaries.

- Real-time live updates, offline scoring, and exports.

Success metrics:

- DAU/MAU for live matches.

- Time-to-schedule tournament via AI (seconds).

- Scorecard accuracy.

- Retention for tournament organizers.

- API latency < 200ms for score updates.

2. Feature List (Core + Advanced AI)

Core features:

- Ball-by-ball scoring.

- Complete scorecard.

- Real-time updates.

- Tournament management.

- Offline mode.

- Export/share.

- User accounts.

- Player profiles & stats.

Advanced / Nice-to-have:

- Auto-schedule generator.

- Boundary tracker, wagon wheel, graphs.

- Live commentary synthesis.

- Role-based dashboards.

- Multi-format support.

AI Features:

- Prompt-based match scheduling.

- Smart fixture optimizer.

- Automatic match summary generator.

- Ball-by-ball commentary assistant.

- Player insights & predictions.

- Question-answering over match/tournaments.

- Natural-language match creation.

3. High-level Architecture

Frontend: React + Material UI.

Backend: Node.js + Express + Socket.io.

Database: MongoDB (Atlas).

AI Layer: Gemini API via LangChain microservice.

Storage: S3-compatible.

Deployment: Docker + GitHub Actions + Cloud Run.

4. Data Model

(users, players, matches, tournaments, embeddings_index)

[Omitted for brevity — see full plan in chat.]

5. API & Realtime Endpoints

Core REST + Socket.io events (score_update, match_end).

6. Frontend Structure

Pages: Home, Match, Score/Create, Tournament, AI/Scheduler, Profile.

Components: ScoreBoard, BallEntryPad, OverTimeline, LiveChart, AISchedulePrompt.

7. AI Integration (Gemini + LangChain)

Python microservice via LangChain, with scheduling, summary, QA tools.

Security: server-side Gemini API key, rate-limited endpoints.

8. Dev & Infra Stack

React + MUI + Node + MongoDB + LangChain + Gemini + Docker + CI/CD.

9. Testing & Deployment

Unit + Integration + E2E + AI prompt tests.

Deployment via GitHub Actions + Cloud Run + MongoDB Atlas.

10. Step-by-Step Plan (12 weeks)

Sprint 0–11 from setup → MVP → AI → QA → launch.

11. Deliverables

- Repo scaffold

- API contract

- UI mockups

- AI microservice

12. Next Actions

Generate project scaffold or LangChain microservice.