# **Technical Report**

### **Posture AI: Digital Sports and Athlete Assessment Platform**

**Team:** Aditya Gaikar - 28

Sumanyu Joshi - 34

Harsh Kumar

Annish Mundhe - 49

**Course:** Full Stack Development (Sem V)  
 **Date:** October 2025

## **1. Introduction and Project Context**

### **1.1 Executive Summary**

The **Posture AI: Sports and Athlete Assessment Platform** represents a digital transformation initiative aimed at improving the efficiency, accuracy, and accessibility of athlete evaluation across India. By combining **AI-driven posture detection**, **real-time scoring**, and **data analytics**, this platform seeks to assist programs like *Khelo India* in identifying, nurturing, and tracking grassroots sports talent.

The platform provides two distinct user experiences:

* **Athletes** — who can upload their assessment videos, receive AI-based posture feedback, and track progress over time.
* **Officials and Coaches** — who can review performance data, validate AI assessments, and make informed decisions for selection and training.

This report outlines the **end-to-end technical implementation** of the system, detailing frontend, backend, database, and deployment strategies, all integrated using modern full-stack technologies (React, Node.js, MongoDB, Docker, TailwindCSS, JWT, and WebSockets).

### **1.2 Problem Domain and Justification**

The sports ecosystem in India faces several persistent challenges in athlete assessment and development:

* **Inconsistent Evaluation:** Manual evaluation introduces human bias and inconsistency across different evaluators or regions.
* **Limited Infrastructure:** Many regions lack access to standardized assessment centers or certified coaches.
* **Data Fragmentation:** There is no centralized repository for athlete data, making long-term performance tracking difficult.
* **Operational Costs:** Organizing physical assessment camps for thousands of athletes is costly and logistically complex.
* **Lack of Motivation:** Athletes often do not receive immediate feedback or gamified engagement to encourage consistent training.

The **Posture AI platform** addresses these gaps through automation, transparency, and scalability. It uses digital video submissions, AI posture tracking, and web-based analytics to eliminate manual inefficiencies while promoting inclusivity in talent identification.

### **1.3 Objectives**

The primary objectives of the project are:

1. To create a **web-based assessment platform** that allows athletes to record, upload, and receive analysis of their training posture.
2. To assist officials in **data-driven selection** by providing quantified performance metrics.
3. To implement **secure, role-based access** ensuring data privacy and integrity.
4. To develop a **scalable full-stack architecture** supporting real-time updates, continuous deployment, and user engagement features.
5. To demonstrate integration of ten critical full-stack experiments—ranging from frontend UI design to backend API security and DevOps deployment—within a cohesive real-world solution.

**2. System Architecture and Technology Stack**

The project is built upon a **Three-Tier Architecture**, leveraging the **MERN stack** (MongoDB, Express.js, React, Node.js) for its unified language environment (JavaScript/TypeScript), high performance, and robustness. This structure ensures **modularity, scalability, and maintainability** across the entire application lifecycle.

### **2.1 Architectural Overview**

The application is logically segmented into three distinct and interconnected layers:

#### **1. Presentation Tier (Frontend)**

* **Technology:** **React.js**
* **Purpose:** Responsible for rendering the user interface and handling all client-side logic. It functions as a **Single Page Application (SPA)**, communicating with the API via asynchronous requests.
* **Key Tools:**
  + **Tailwind CSS:** Used for utility-first styling to ensure rapid development of **responsive** and modern UIs.
  + **Redux Toolkit (RTK):** Manages **complex global state** (e.g., user authentication status, athlete performance data) in a centralized and predictable manner.

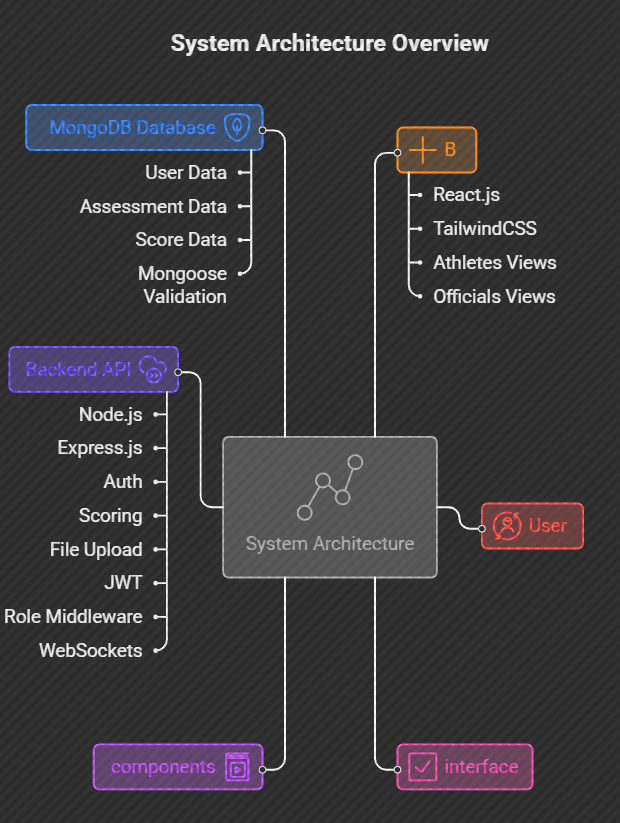
#### **2. Application Tier (Backend/API)**

* **Technology:** **Node.js** with the **Express.js** framework.
* **Purpose:** Houses the core business logic, handles routing, processes client requests, manages user authentication, and interfaces with the database. Node.js's non-blocking, event-driven nature ensures high throughput and low latency.
* **Key Tools:**
  + **JWT (JSON Web Token):** Implements **stateless authentication** and **Role-Based Access Control (RBAC)** for secure API access.
  + **Socket.io:** Provides the **WebSockets** layer for **real-time communication**, essential for live leaderboards and scoring updates.

#### **3. Data Tier (Database)**

* **Technology:** **MongoDB**
* **Purpose:** A flexible, document-based NoSQL database used to store all application data, including athlete profiles, assessment records, official credentials, and configurable event parameters.
* **Key Tools:**
  + **Mongoose:** Serves as the Object Data Modeling (ODM) layer, providing **schema enforcement** and simplified, type-safe interaction with the MongoDB instance.

### **2.2 System Architecture Diagram (Conceptual)**



## 

## **3. Implementation Details**

### **3.1 Frontend Development**

The frontend, developed using **React.js**, delivers a seamless, responsive Single Page Application (SPA) interface.

* **TailwindCSS** was used for rapid UI development, ensuring responsiveness across devices.  
   Example: The athlete dashboard automatically adapts between mobile and desktop views with simple utility classes (flex-col md:flex-row).
* **React Hooks (useEffect, useContext, custom hooks)** enabled modular data handling:  
  + useEffect() manages lifecycle tasks such as fetching user data on mount.
  + useContext() provides global access to authentication and theme data.
  + A **custom hook** useAssessmentData() centralizes logic for fetching and caching performance data.
* **Redux Toolkit** was integrated for managing global application state (auth, athletes, assessments). Each data slice ensures predictable updates, allowing synchronized dashboards for all users.

### **3.2 Backend and Database Integration**

The backend uses **Express.js** and **Node.js** to create a modular and secure REST API.

#### **Key Features**

* **RESTful Endpoints:** /api/auth, /api/athletes, /api/assessments
* **Database Models:**
  + UserSchema – defines athlete and official roles.
  + AssessmentSchema – stores score arrays, video URLs, evaluator remarks.
  + PostureAnalysisSchema – used for AI feedback metadata (accuracy, deviation angles).
* **Data Validation:** Mongoose enforces schema rules while express-validator ensures clean, secure input handling.

#### **Security Measures**

* **JWT Authentication:** Each request to protected routes verifies a signed token containing user ID and role.
* **Role-Based Access:** Officials can approve or reject athlete assessments; athletes can only modify their own data.
* **Rate Limiting & Helmet:** Prevents brute-force login attempts and sets secure HTTP headers.

### **3.3 Real-Time Features with WebSockets**

Using **Socket.io**, the system provides real-time updates:

* Officials updating scores trigger instant UI refreshes for athletes’ dashboards.
* Leaderboards auto-update without reloading the page.
* Chat channels can be added later for live event coordination.

This asynchronous architecture eliminates the need for heavy HTTP polling and provides an engaging, responsive experience.

### **3.4 API Testing and Validation**

**Postman** was employed to test all API endpoints through automated test suites:

* Verified success responses (201/200) for valid requests.
* Checked authorization handling for missing or invalid tokens.
* Tested input sanitization by attempting XSS and invalid payloads.

These validations ensured the backend was robust, secure, and production-ready.

### **3.5 Deployment and DevOps Integration**

Deployment follows a **CI/CD pipeline** integrated via **GitHub Actions**:

1. **CI Step:** Runs linting, unit tests, and build verification on every push.
2. **CD Step:** Auto-deploys to Vercel (frontend) and Render (backend) after passing all tests.
3. **Docker Integration:**
   * Each environment is encapsulated in a Docker container.
   * Multi-stage Dockerfiles reduce image size and ensure version consistency.
   * Backend container exposes ports for WebSocket and REST APIs, ensuring portability.

This automated workflow allows continuous feature rollout with minimal downtime.

## **4. AI and Posture Analysis Module**

A distinctive feature of the platform is **AI-assisted posture evaluation**, integrated through model endpoints designed to process uploaded videos.  
 The model analyzes:

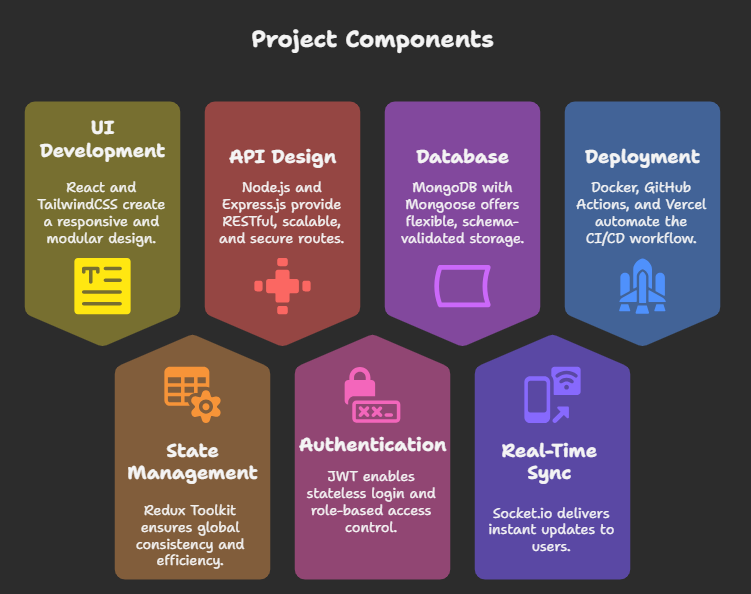
* **Joint Angles and Limb Positions** via Mediapipe or OpenPose.
* **Deviation Metrics:** Calculates posture deviation percentage for each repetition (push-ups, squats, etc.).
* **Performance Scoring:** AI-generated numeric score is combined with official’s input for a final composite rating.

This component is modularly designed, allowing integration with any external AI API or cloud-based inference system later.

## **5. Results and Outcomes**

### **5.1 Functional Achievements**

* Successful implementation of a **role-based, multi-user system**.
* Fully responsive dashboards for both athlete and official interfaces.
* Secure, production-ready API with validated endpoints.
* Real-time leaderboard synchronization using WebSockets.  
  Automated CI/CD deployment pipeline ensuring rapid iteration.

**5.2 Technical Integration Summary**

## **6. Conclusion and Future Scope**

### **6.1 Summary**

The **Posture AI Platform** successfully merges sports technology, AI analysis, and modern web architecture to offer a scalable, accessible, and transparent athlete assessment ecosystem.  
 It demonstrates a deep understanding of full-stack development concepts—covering everything from UI design to backend security, state management, and DevOps practices.

### **6.2 Future Enhancements**

* **AI Automation:** Integrate live posture detection models for real-time scoring during practice sessions.
* **Data Analytics Dashboard:** Predictive analysis for athlete performance trends using ML models.
* **Mobile App Development:** React Native app for instant access and offline training uploads.
* **Blockchain for Data Integrity:** Secure athlete history and assessment records with tamper-proof transactions.
* **Microservices Migration:** Modularize backend for independent scaling of high-traffic components like video analysis and leaderboard computation.

### **6.3 Final Remarks**

This project embodies the spirit of *Digital India* by transforming sports assessment into a transparent, data-driven, and inclusive ecosystem. By empowering athletes with AI insights and enabling officials with reliable analytics, **Posture AI** lays the foundation for a smarter, more equitable approach to talent identification and development in Indian sports.