SMART PROCTORING SYSTEM

Industry-Grade Backend Architecture Documentation

1. Introduction

This document describes the complete backend architecture of the Smart Proctoring System from an industry and production-oriented perspective. Although the project is academic and simulation-based, the backend design follows real-world enterprise standards, modularity, and security practices.

2. Architectural Design Principles

- Modular & Scalable Design - Separation of Concerns (SoC) - Secure-by-Design Architecture - Auditability & Traceability - AI & Blockchain Integration Readiness

3. High-Level Backend Architecture

The backend follows a layered architecture model consisting of API, Service, AI Engine, Blockchain, and Data layers. Each layer is isolated and communicates through well-defined interfaces.

3.1 Layered Architecture Overview

Presentation Layer: Handles HTTP requests from frontend Application Layer: Business logic & workflows AI Processing Layer: Face, Voice & Stress analysis Blockchain Layer: Immutable event logging Persistence Layer: PostgreSQL database

4. Backend Technology Stack

Layer	Technology	Purpose
API Layer	Flask (Python)	RESTful API handling
Auth	JWT	Secure session management
AI Engine	OpenCV, Librosa, NumPy	Biometric & stress analysis
Blockchain	Python (SHA-256)	Immutable logging
Database	PostgreSQL	Relational data storage
Security	Hashing, RBAC	Data protection & access control

5. Detailed Backend Module Architecture

5.1 Authentication & Authorization Module

This module manages user authentication, authorization, and role-based access control. Responsibilities: - User login & token generation - Role validation (Admin / Student) - Secure route protection

5.2 Exam Management Module

Handles the lifecycle of exams including creation, scheduling, access control, and submission locking. Responsibilities: - Exam creation & configuration - Time-bound access enforcement - Submission validation

5.3 AI Proctoring Engine

The AI engine is responsible for all biometric verification and behavioral analysis. It operates on pre-collected datasets and produces structured anomaly reports. Submodules: - Face Recognition Engine - Voice Recognition Engine - Stress Detection Engine

5.4 Blockchain Logging Module

Implements a simulated blockchain to maintain immutable logs of all exam-related activities. Responsibilities: - Event hashing - Block creation & chaining - Audit trail generation

5.5 Evaluation & Reporting Module

Aggregates exam submissions, proctoring alerts, and blockchain logs to generate final reports. Responsibilities: - Result computation - Certificate issuance - Analytics & insights

6. Backend Folder Structure (Industry Style)

backend/ ■■■ app.py ■■■ config/ ■■■ database/ ■■■ models/ ■■■ repositories/ ■■■ services/ ■■■ controllers/ ■■■ ai_engine/ ■■■ blockchain/ ■■■ security/ ■■■ logs/ ■■■ utils/

7. Data Flow Architecture

1. Frontend sends request to API layer 2. Controller validates request & auth token 3. Service layer processes business logic 4. AI engine performs analysis (if required) 5. Blockchain logs the event 6. Database persists structured data 7. Response returned to frontend

8. Security Architecture

- JWT-based authentication - Encrypted biometric hashes - Blockchain-backed audit logs - Strict role-based access control - Input validation & sanitization

9. Scalability & Future Readiness

The backend architecture is designed to be cloud-ready. Modules can be independently scaled and replaced with real-time AI services, external blockchain networks, and microservices in future iterations.

10. Conclusion

This industry-grade backend architecture ensures that the Smart Proctoring System meets professional standards while remaining suitable for academic demonstration. It bridges theoretical concepts with real-world backend design practices.