**High-Level Design Document**

**Project Name:** Attendance Tracking System  
**Version:** 1.0

**1. Introduction**

**Purpose**

This document provides a high-level design for the Project X Attendance Tracking System. The goal is to define the architecture, main components, interfaces, and key functionalities that meet the requirements outlined in the initial specification.

**Scope**  
The system will manage users (students and instructors), enable secure attendance tracking through barcode scanning, and ensure data protection using modern security standards. It will be deployed as a cloud-based solution.

**2. System Architecture**

**Overview**  
The system follows a **three-tier architecture**:

* **Presentation Layer**: Web/mobile interfaces for students, instructors, and admins.
* **Application Layer**: Backend services handling business logic, authentication, and data processing.
* **Data Layer**: Cloud database (MongoDB Atlas or MySQL) for storing user data, attendance logs, and configurations.

**3. Functional Components**

**3.1 User Management**

* Admin dashboard for:
  + Student and instructor registration
  + Uploading student photos and details
  + Removing users
  + Managing device registrations
* RBAC enforcement for access restrictions

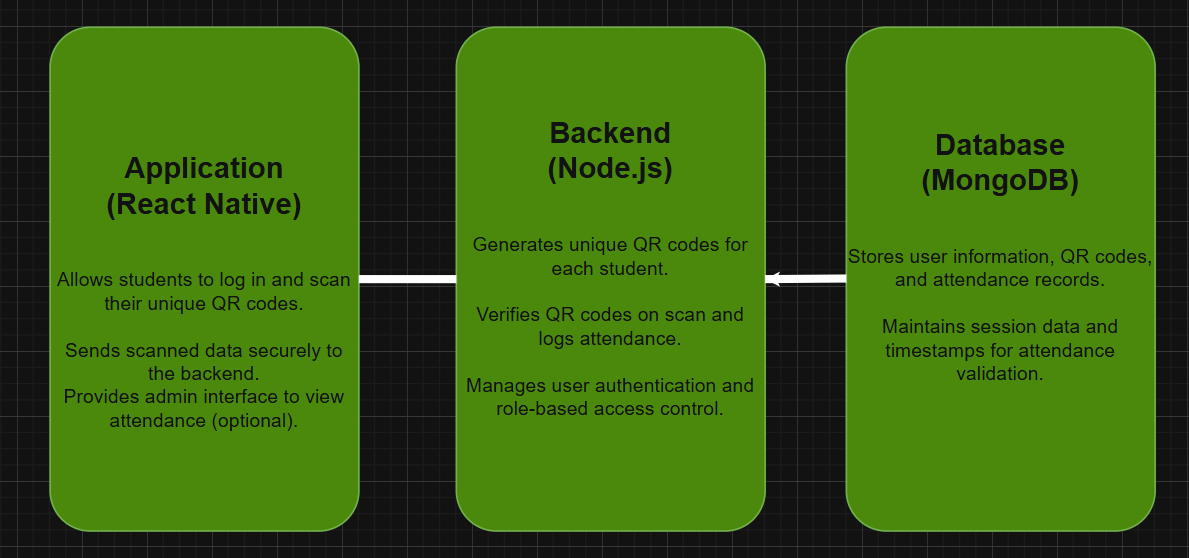
**3.2 Attendance Tracking**

* Instructor interface for barcode-based scanning using mobile/laptop cameras
* Auto-drop logic for students with 3 consecutive absences
* Student dashboard to view attendance records per course
* Attendance reporting tools for admins

**4. Security Design**

|  |  |
| --- | --- |
| **Feature** | **Desciption** |
| **RBAC** | **Define roles: Admin, Instructor, Student. Restrict feature access accordingly.** |
| **Data Encryption** | **Admin-controlled encryption for sensitive data in storage and transit.** |
| **Encryption** | **Admin-controlled encryption at rest** |
| **Access Restrictions** | **Only authorized users can access protected features** |

**5. Architecture Diagram**

****

**6. Data Flow / ER Diagrams**

**Entities:**

* User (student/admin)
* QR Code (unique per student)
* Attendance Record (student ID, session ID, timestamp, status)
* Session (attendance event with unique ID and time)

**Flow:**

1. User logs in → Backend authenticates → Role assigned.
2. Backend generates/stores unique QR code per student.
3. Student scans QR code → Data sent to backend.
4. Backend verifies QR code against student/session.
5. Attendance record created with timestamp.
6. Admin retrieves attendance data via dashboard.

**7. Technology Stack**

* **Frontend:** React Native
* **Backend:** Node.js
* **Database:** MongoDB (NoSQL for flexible attendance data)
* **QR Code:** QR code generation libraries (e.g., qrcode npm package)
* **Security:** HTTPS, RBAC for permissions

**8. Assumptions & Constraints**

* QR scanning works via camera-enabled devices
* University users will register manually via web dashboard
* System does not integrate with other university APIs (per R05.03)

**9. Data Flow Description**

1. **Registration:**  
   Authorized user registers students/instructors → system stores data + ID photo in DB
2. **Attendance:**  
   Instructor scans barcode → system validates → records timestamped attendance
3. **Viewing:**  
   Students login → view their attendance per course

**10. Timeline**

* **Initial System Check:** May 15, 2025
* **Full Deployment:** May 22, 2025