Republic of the Philippines

DAVAO ORIENTAL STATE UNIVERSITY

Guang-guang, Dahican, City of Mati, Davao Oriental

Faculty of Computing, Data Sciences, Engineering and Technology

Information Technology Program

**ITC 130 – Applications Development in Emerging Technologies**

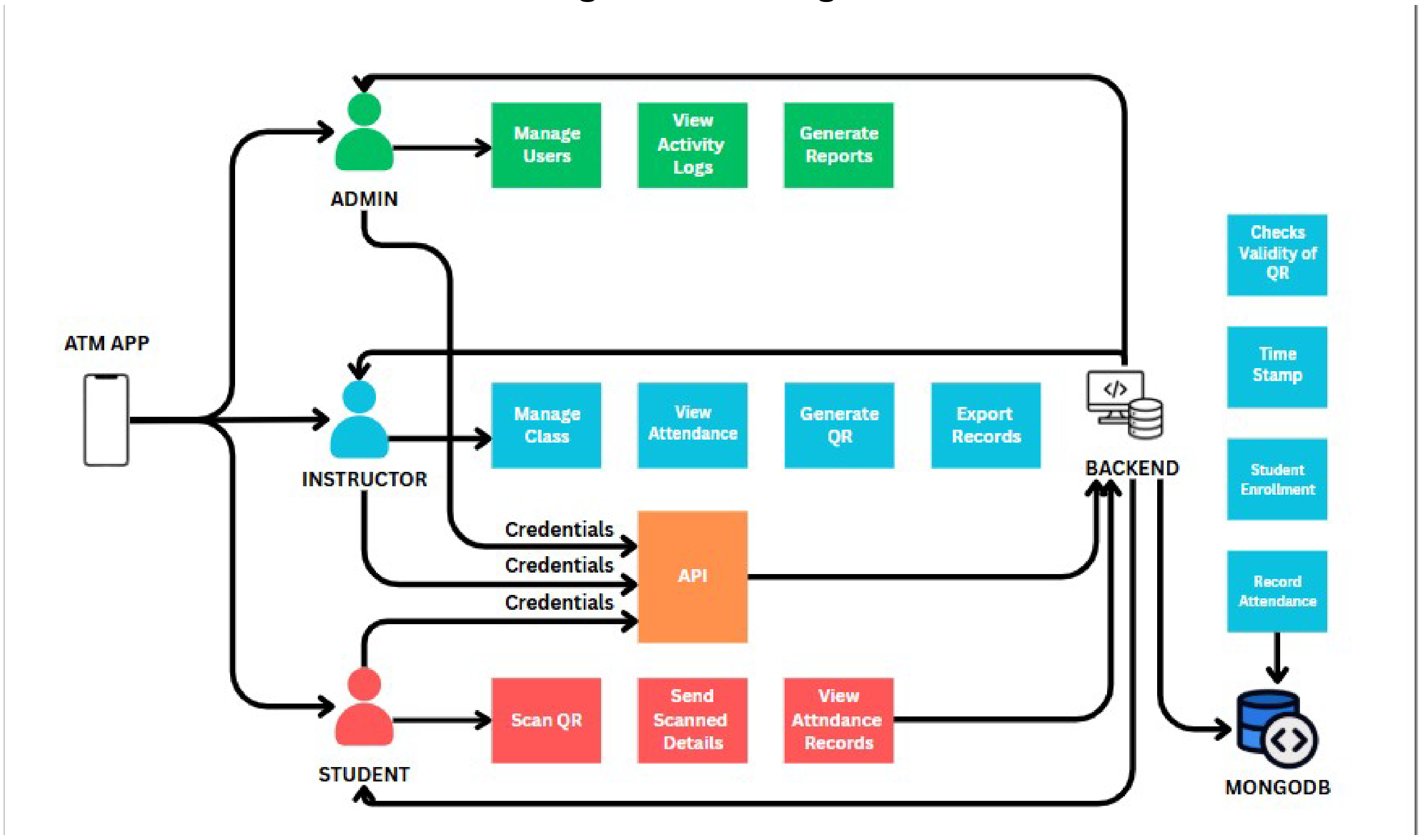
# PROJECT X: Automated Attendance Tracking System: High Level Design

Presented by:

Joseph Angelo Gardose
Hazelle Ibanez
Nofah Mae Macatabog

MAY 2025

## High Level Design



## High-Level Design (HLD) of the Automated Attendance Tracking System

### Overview

The High-Level Design (HLD) outlines the core structure and interactions of the Automated Attendance Tracking System (ATM), aligned with the system architecture shown in the diagram. The design ensures modularization, security, and scalability, with key functionalities clearly distributed among users (Admin, Instructor, Student) and coordinated via API and backend processes.

## System Architecture

The system employs a **client-server architecture** with clearly separated layers:

- **Frontend (Client Side)**
  - Built with **React Native**, represented in the diagram as the **ATM App**, which serves as the interface for Admins, Instructors, and Students.
- **Backend (Server Side)**
  - Implemented using **Node.js with Express.js**, this layer (represented by the “BACKEND” box in the image) handles business logic, QR validation, timestamping, and records processing.
- **Database Layer**
  - The **MongoDB Atlas** database (shown as "MONGODB" in the diagram) stores all structured data: student records, attendance logs, course information, and QR metadata.
- **Cloud Hosting**
  - Hosted on cloud services (e.g., Heroku, Vercel) to allow remote, scalable access across mobile and web.

---

## Major Components (Mapped to Diagram)

### A. User Management (Green Boxes: Admin Actions)

- Admins can:

  o Manage users (Manage Users) o View system-

  wide logs (View Activity Logs) o Generate

  detailed reports (Generate Reports)
- **Role-Based Access Control (RBAC)** ensures Admins have elevated privileges, consistent with their role in the system architecture.

**B. Attendance Tracking Module (Red Boxes: Student Actions, Blue Boxes: Instructor Actions)**

- **QR-Based Attendance Workflow**:
  1. **Student (Red Box: Scan QR)** generates a dynamic QR code.
  2. **Instructor (Blue Box: Generate QR / View Attendance)** scans this code via their device.
  3. The **backend** validates the QR (right-side light blue box: "Checks Validity of QR") and adds a timestamp.
  4. Attendance is stored in **MongoDB** ("Record Attendance" process).
- Credentials flow between roles is governed via the **API**, shown in orange.

**C. Course and Enrollment Management**

- Admins assign instructors and enroll students in courses.
- Instructors manage these classes (Blue Box: Manage Class) and may postpone or cancel sessions.
- Student enrollment is validated in the backend (“Student Enrollment” light blue box).

**D. Device Management**

- Instructors use **registered devices** (diagram: ATM APP device).
- Lost or unauthorized devices are tracked or deactivated by Admins.
- Only authorized devices connect to the **API layer** for attendance operations.

**E. Report Generation**

- Both Admins and Instructors can **generate and export reports** (Green and Blue Boxes: Generate Reports, Export Records).
- These reports are aggregated from MongoDB data and formatted based on roles.

**Security Design (Noted in Both Diagram & Text)**

- **HTTPS** encryption for secure communication between client and server.
- **Multi-Factor Authentication (MFA)** upon login (implied in credentials exchange).
- **JWT-based authentication** to validate and maintain session tokens.
- **RBAC** prevents unauthorized access (e.g., students cannot modify logs).
- **Device Authorization** ensures only trusted hardware interacts with sensitive functions.

**Technology Stack:**

| Layer | Technology Used |
|---|---|
| Frontend | React Native (Mobile App) |
| Backend | Node.js with Express.js |
| Database | MongoDB Atlas |
| Hosting | Cloud-based |
| Authentication | MFA |
| QR/Barcode | ZXing or similar libraries |

**Data Flow Summary (Aligned with Diagram)**

1. **Login** – Users authenticate via MFA; credentials are passed through the API (orange box).
2. **QR Code Generation** – Student generates a QR code.
3. **Scan** – Instructor scans QR via ATM App.
4. **Validation** – Backend verifies QR, enrollment, and timestamp.

5. **Logging** – Attendance is recorded in MongoDB with metadata.
6. **Reports** – Admins and Instructors generate attendance summaries from the database.

## Conclusion

This HLD, grounded in the visual architecture diagram, ensures the Automated Attendance Tracking System is modular, secure, and scalable. Each user role has clear boundaries and tools, and the system leverages modern practices in web and mobile development, ensuring robust and future-proof performance. The integrated API and database backbone allow seamless interaction between users and system services, as visualized in the design flow.