# ****Attendance Management System - Project Overview (**[GitHub](https://github.com/amaan0810/attendence-management)**)****

## ****Project Description****

The **Attendance Management System** is a web application designed to efficiently manage and track student attendance in an educational environment. The system allows three types of users:

* **Admin**: Manages users, including students and teachers, and controls user access.
* **Teacher**: Tracks and marks attendance for students, and views their records.
* **Student**: Marks their own attendance and views their attendance history.

The application provides a simple and intuitive user interface, utilizing React for the frontend and Node.js/Express for the backend. Authentication is handled using JWT (JSON Web Tokens) to ensure secure access to protected resources.

### ****Features****

* **Admin Dashboard**: Admin users can view all users, restrict or activate users, and manage roles.
* **Teacher Dashboard**: Teachers can manage attendance for students, view student lists, and track student attendance.
* **Student Dashboard**: Students can mark their own attendance via selfies, view their attendance history, and update their profiles.
* **Profile Management**: Each user (admin, teacher, student) can view and update their profile, including password changes.
* **Authentication**: JWT-based authentication for secure login and role-based access control.

## ****File Structure****

The project follows a typical full-stack React application structure. Below is an overview of the directory layout, including both frontend and backend components:

### ****Frontend Directory Structure****

attendance-frontend/

├── public/ # Static assets like index.html and images

│ └── index.html

├── src/ # Source code

│ ├── assets/ # For static files like images or icons

│ ├── components/ # Reusable components used in different dashboards

│ │ ├── Admin/ # Admin-related components (e.g., UserTable)

│ │ │ ├── UserTable.jsx

│ │ │ └── RestrictUserModal.jsx

│ │ ├── Teacher/ # Teacher-related components (e.g., StudentList, AttendanceTable)

│ │ │ ├── StudentList.jsx

│ │ │ └── AttendanceTable.jsx

│ │ ├── Student/ # Student-related components (e.g., MarkAttendance, AttendanceHistoryTable)

│ │ │ ├── MarkAttendance.jsx

│ │ │ └── AttendanceHistoryTable.jsx

│ │ ├── Common/ # Common components (Header, Footer, etc.)

│ │ │ ├── Header.jsx

│ │ │ └── Footer.jsx

│ │ ├── ProtectedRoute.jsx # Protect routes based on login

│ ├── pages/ # Pages for different routes (e.g., Dashboard, Profile, Login)

│ │ ├── AdminDashboard.jsx

│ │ ├── TeacherDashboard.jsx

│ │ ├── StudentDashboard.jsx

│ │ ├── Profile.jsx # Displays the user's profile information

│ │ ├── UpdateProfile.jsx # Allows user to update profile details

│ │ ├── Login.jsx # Login page for authentication

│ │ ├── Register.jsx # Registration page

│ │ ├── HomePage.jsx # Landing page of the app

│ ├── utils/ # API utility functions (e.g., axios instance for API calls)

│ │ └── api.js

│ ├── App.jsx # Main application with routing

│ ├── index.jsx # Entry point to the app

│ ├── index.css # Global styles (Tailwind CSS setup)

├── tailwind.config.js # Tailwind CSS configuration

├── vite.config.js # Vite configuration for the frontend build

├── package.json # Frontend dependencies and scripts

├── .gitignore # Git ignore file to exclude unnecessary files from version control

### ****Backend Directory Structure****

attendance-backend/

├── controllers/ # Logic for handling requests

│ ├── authController.js # Handles authentication (login, register, JWT generation)

│ ├── userController.js # Handles user management (admin, teacher, student)

│ └── attendanceController.js # Handles attendance-related functionality

├── models/ # Mongoose models for MongoDB

│ ├── User.js # User schema (includes roles and account status)

│ ├── Attendance.js # Attendance schema

├── routes/ # Express routes

│ ├── auth.js # Authentication routes (login, register, password update)

│ ├── user.js # Admin routes for managing users

│ ├── attendance.js # Routes for handling attendance

├── middlewares/ # Custom middleware functions

│ └── authMiddleware.js # Middleware to verify JWT token for protected routes

├── config/ # Configuration files (e.g., for MongoDB and JWT secret)

│ ├── db.js # Database connection setup

│ └── config.js # Server and environment configurations

├── server.js # Main server file to run the Express app

├── .env # Environment variables (e.g., JWT\_SECRET, DB\_URI)

├── package.json # Backend dependencies and scripts

├── .gitignore # Git ignore file for the backend

## ****Functionality Overview****

### ****Frontend****

1. **Home Page**: The entry point of the application, which allows the user to either log in or register.
2. **Login & Registration**: Users (Admin, Teacher, Student) can log in or create new accounts.
3. **Admin Dashboard**: Admins can view, restrict, and reactivate users, and manage user roles.
4. **Teacher Dashboard**: Teachers can mark attendance for students, view student lists, and manage attendance records.
5. **Student Dashboard**: Students can mark their own attendance, view their attendance history, and update their profile.
6. **Profile Management**: Each user (Admin, Teacher, Student) has the ability to view and update their profile, including their password.

### ****Backend****

1. **Authentication**: Secure authentication using JWT (JSON Web Token).
2. **Role-Based Access Control**: Admins have access to manage all users, while teachers can manage students' attendance, and students can only view and mark their own attendance.
3. **User Management**: Admins can view all users and restrict/reactivate them.
4. **Attendance Management**: Teachers and students can track attendance. Students can mark their own attendance via selfies.

## ****Conclusion****

This Attendance Management System allows for efficient management and tracking of student attendance in an educational setting. With separate dashboards for Admins, Teachers, and Students, it provides an intuitive user interface and secure authentication mechanisms. The backend is powered by Node.js and MongoDB, with a clean and simple architecture to handle user roles, attendance data, and profile management.

Feel free to expand on this system by adding features like reports, notifications, or more detailed analytics!

Let me know if you need additional modifications or explanations for any part of this project!