# Smart Attendance System using AI Camera

## Frontend - App.js (React)

import React, { useState } from 'react';  
import './App.css';  
  
function App() {  
 const [username, setUsername] = useState('');  
 const [password, setPassword] = useState('');  
 const [role, setRole] = useState('student');  
 const [studentName, setStudentName] = useState('');  
 const [status, setStatus] = useState('');  
 const [loggedIn, setLoggedIn] = useState(false);  
 const [isRegister, setIsRegister] = useState(false);  
  
 const login = async () => {  
 const res = await fetch('/api/login', {  
 method: 'POST',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({ username, password })  
 });  
 const data = await res.json();  
 if (data.success) {  
 setLoggedIn(true);  
 setRole(data.role);  
 setStatus('Login successful');  
 } else {  
 setStatus('Invalid credentials');  
 }  
 };  
  
 const register = async () => {  
 const res = await fetch('/api/register', {  
 method: 'POST',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({ username, password, role })  
 });  
 const data = await res.json();  
 setStatus(data.message);  
 };  
  
 const markAttendance = async () => {  
 const res = await fetch('/api/mark-attendance', {  
 method: 'POST',  
 headers: { 'Content-Type': 'application/json' },  
 body: JSON.stringify({ name: studentName })  
 });  
 const data = await res.json();  
 setStatus(data.message);  
 };  
  
 return (  
 <div className="App">  
 <h1>Smart Attendance System</h1>  
 {!loggedIn ? (  
 <div>  
 <input type="text" placeholder="Username" onChange={e => setUsername(e.target.value)} />  
 <input type="password" placeholder="Password" onChange={e => setPassword(e.target.value)} />  
 {isRegister && (  
 <select onChange={e => setRole(e.target.value)}>  
 <option value="student">Student</option>  
 <option value="admin">Admin</option>  
 </select>  
 )}  
 <button onClick={isRegister ? register : login}>{isRegister ? 'Register' : 'Login'}</button>  
 <button onClick={() => setIsRegister(!isRegister)}>  
 {isRegister ? 'Switch to Login' : 'Switch to Register'}  
 </button>  
 </div>  
 ) : (  
 <div>  
 {role === 'student' && (  
 <div>  
 <input type="text" placeholder="Enter Name" onChange={e => setStudentName(e.target.value)} />  
 <button onClick={markAttendance}>Mark Attendance</button>  
 </div>  
 )}  
 {role === 'admin' && <p>Welcome Admin! Dashboard coming soon.</p>}  
 </div>  
 )}  
 <p>{status}</p>  
 </div>  
 );  
}  
  
export default App;

## Backend - Spring Boot Code Snippets

@RestController  
@RequestMapping("/api")  
public class AttendanceController {  
 @Autowired private AttendanceService attendanceService;  
 @Autowired private UserService userService;  
  
 @PostMapping("/mark-attendance")  
 public ResponseEntity<?> markAttendance(@RequestBody Map<String, String> request) {  
 String name = request.get("name");  
 boolean success = attendanceService.markAttendance(name);  
 return success  
 ? ResponseEntity.ok(Collections.singletonMap("message", "Attendance marked for " + name))  
 : ResponseEntity.status(400).body(Collections.singletonMap("message", "Failed to mark attendance"));  
 }  
  
 @PostMapping("/login")  
 public ResponseEntity<?> login(@RequestBody Map<String, String> request) {  
 String username = request.get("username");  
 String password = request.get("password");  
 Optional<User> user = userService.getUser(username, password);  
 if (user.isPresent()) {  
 Map<String, Object> response = new HashMap<>();  
 response.put("success", true);  
 response.put("role", user.get().getRole());  
 return ResponseEntity.ok(response);  
 } else {  
 return ResponseEntity.ok(Collections.singletonMap("success", false));  
 }  
 }  
  
 @PostMapping("/register")  
 public ResponseEntity<?> register(@RequestBody Map<String, String> request) {  
 String username = request.get("username");  
 String password = request.get("password");  
 String role = request.get("role");  
 boolean created = userService.createUser(username, password, role);  
 return created  
 ? ResponseEntity.ok(Collections.singletonMap("message", "User registered successfully"))  
 : ResponseEntity.status(400).body(Collections.singletonMap("message", "User registration failed"));  
 }  
}  
  
@Entity  
public class User {  
 @Id @GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;  
 private String username;  
 private String password;  
 private String role;  
}  
  
@Entity  
public class Attendance {  
 @Id @GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;  
 private String name;  
 private LocalDateTime timestamp;  
}  
  
spring.datasource.url=jdbc:mysql://localhost:3306/attendance\_db  
spring.datasource.username=root  
spring.datasource.password=yourpassword  
spring.jpa.hibernate.ddl-auto=update