Final Year Project Specification: Smart E-Attendance and Virtual Classroom Platform

---

Feature 1: Offline Attendance Module

Objective:

Enable secure offline attendance using smartphone fingerprint authentication and IP subnet restriction.

Components:

- **Smartphone Fingerprint Scanner via WebAuthn**: Use Web Authentication API (WebAuthn) to access fingerprint-based biometric login on supported devices.
- **OTP Verification Fallback System**: A backend service (e.g., PHP mailer or SMS API like Twilio) to send one-time passwords for devices without fingerprint scanners.
- **IP Subnet Matching**: Logic to verify if student and teacher are connected to the same subnet (e.g., 192.168.1.x) by comparing the first three octets of their IP addresses.
- **Teacher Attendance Portal**: A web page for the teacher to initiate and close attendance sessions, setting an active time frame.
- **Student Attendance Portal**: A portal that checks IP subnet, then prompts biometric authentication or OTP before logging attendance to the MySQL database.

Workflow:

1. **Teacher opens attendance session**, generating a valid session with IP and time.
2. **Students access attendance portal**, triggering IP subnet match check.
3. If subnet matches:
    - **Student authenticates using fingerprint (WebAuthn)**
    - If fingerprint not available, OTP is sent to registered number/email
4. Upon success, attendance is recorded in database with timestamp and IP.

Feature 2: Online Virtual Class Module

Objective:

Facilitate live virtual classes with auto-attendance, chat, screen sharing, and whiteboard features.

Components:

- **Jitsi Meet API Integration**: Open-source video conferencing platform; integrate using iframe or Jitsi Meet External API in your web app.
- **Auto-attendance via Login Tracking**: Record the time and ID of students who join the class through their authenticated portal.
- **Live Chat**: Enable in-class communication using built-in Jitsi chat or an external JavaScript-based group chat (e.g., Socket.io or Firebase).
- **Screen Share**: A default feature of Jitsi Meet that allows presenters to share their screens.
- **Collaborative Whiteboard**: Integration of plugins like Excalidraw or Jitsi whiteboard for drawing and collaboration.

Workflow:

1. Teacher creates a virtual class link via Jitsi.
2. Students join using their login-authenticated portal.
3. System logs attendance when student joins.
4. During session:
    - Students & teacher use live chat
    - Teacher shares screen or uses whiteboard
5. Session ends; attendance and session metadata stored.

Feature 3: Learning Materials & Assignments Module

Objective:

Allow teachers to share resources and assignments with deadlines; students can submit assignments and access recordings.

Components:

1. **Notes Repository**:
    - Teachers can upload and publish **class notes** for students.
    - Notes could be in various formats (PDF, Word, PPT, etc.).
    - Students can **download** notes for offline study.
2. **Assignments Management**:
    - Teachers can create **assignments** with descriptions, deadlines, and submission instructions.
    - Assignments can be uploaded as file attachments (e.g., PDF, DOCX, etc.).
    - Students can **submit assignments** via the portal by uploading their work before the deadline.
3. **Assignment Submission & Grading**:
    - Students can submit assignments within a set timeframe.
    - Teachers can **grade** assignments and provide feedback.
    - Assignments are stored in the database with fields like student_id, assignment_id, submission_time, grade, etc.
4. **Recording Upload and Access**:
    - **Online class recordings** (e.g., recorded Jitsi Meet/Google Meet sessions) can be uploaded by the teacher.
    - Students can access, download, and watch the class recordings for review.
5. **Notifications**:
    - Notifications will alert students when a new note, assignment, or recording is posted.

- Reminders can be sent before assignment deadlines.
- Students will be notified once their assignments are graded.

6. **Student Dashboard**:
    - A personalized **dashboard** for each student where they can view **assigned notes**, **submitted assignments**, **grades**, and **class recordings**.

Workflow:

1. Teacher uploads class notes and creates assignments with due dates.
2. Students can:
    - Download notes
    - Submit assignments within the portal before deadline
    - Download past class recordings
3. Teacher evaluates and provides feedback/scores optionally.

Feature 4: Class Routine Management

Objective:

Enable admin to upload weekly class routines and auto-distribute them to teachers and students.

Components:

- **Routine Upload Form**: Admin interface to upload schedule via form or Excel/CSV file.
- **Routine Data Model**: MySQL tables structured with fields like Program, Section, Course, Day, Time, TeacherId.
- **Routine Display UI**: Student and teacher dashboards that filter and show relevant classes based on login data.
- **Optional Reminder System**: Use cron jobs or JavaScript to show upcoming classes.

Workflow:

1. Admin uploads routine with day-wise, time-wise class entries mapped to course and section.
2. On login, teacher and students see only their relevant schedule.
3. Upcoming classes and reminders can be shown dynamically.

Feature 5: Student Performance Analysis using ML

Objective:

Use machine learning to predict and visualize student performance based on attendance, assignment marks, and internal exam scores.

Components:

1. Data Collection Module

- Collect and store:
    - **Attendance percentage per subject**
    - **Assignment marks per subject**
    - **Internal exam/test marks per subject**

2. ML Model Development (Backend)

- Use Python with libraries like **scikit-learn**, **pandas**, **NumPy**, etc.
- Possible models:
    - **Linear Regression** (to predict final performance/marks)
    - **Decision Trees / Random Forest** (to classify performance levels: good, average, poor)
    - **K-Means Clustering** (to group students based on learning behavior)

3. Data Preprocessing

- Normalize and clean the dataset:
    - Fill missing values
    - Convert categorical data if needed

- Standardize the scale (e.g., 0–100 marks)

4. Prediction & Analysis

- Run model on the data to:
  - Predict **expected performance/final marks**.
  - Flag students with **risk of poor performance**.
  - Suggest **which factor** (low attendance, assignment, internal marks) is dragging performance down.

5. Frontend Visualization

- Show visual graphs/charts:
  - Bar charts for subject-wise marks
  - Pie chart of attendance distribution
  - Radar chart comparing attendance vs marks
- Give a **performance summary** for each student

Workflow:

1. Teacher updates attendance and scores data.
2. Backend runs Python-based ML script to:
   - Predict final performance
   - Classify performance level
   - Generate suggestions
3. Students and teachers see:
   - Subject-wise performance analysis
   - Graphical reports and prediction
   - Feedback based on data trends