**Software Requirements Specification (SRS)**

**Attendance Management System**

**1. Introduction**

**1.1 Purpose**

This SRS defines the requirements for an Attendance Management System (AMS) designed for Law College , Chh. Sambhaji nagar. The system enables teachers to record, edit, update, and delete attendance for their assigned subjects and sessions, based on a centralized timetable. It ensures that teachers can only manage attendance for their own classes.

**1.2 Scope**

The AMS will:

* Store and manage data about teachers, students, subjects, timetables, sessions, and attendance.
* Allow teachers to take attendance for their scheduled sessions.
* Restrict teachers from modifying attendance for sessions they are not assigned to.
* Generate sessions automatically based on a central timetable.
* Provide a foundation for future reporting features (e.g., attendance summaries).

**1.3 Definitions**

* **Teacher**: A faculty member assigned to teach specific subjects.
* **Student**: An individual enrolled in subjects whose attendance is tracked.
* **Subject**: A course or class (e.g., Mathematics).
* **Timetable**: A centralized schedule defining when subjects are taught.
* **Session**: A specific instance of a class based on the timetable.

**2. Overall Description**

**2.1 User Needs**

* **Teachers**: Need to view their scheduled sessions, record attendance, and edit/delete records for their students.
* **Admin**: Need to manage the timetable, assign subjects to teachers, and enroll students in subjects.
* **Students**: (Future scope) May view their attendance records.

**2.2 Assumptions**

* Each teacher is assigned to specific subjects.
* Sessions are generated based on the timetable and tied to specific dates.
* The system will be accessed via a web or desktop application (platform TBD).

**3. System Features**

**3.1 User Management**

* **3.1.1 Teacher Management**
  + **Description**: Store and manage teacher details.
  + **Input**: Teacher’s first name, last name, email.
  + **Output**: Unique teacher\_id assigned.
  + **Constraints**: Email must be unique.
* **3.1.2 Student Management**
  + **Description**: Store and manage student details.
  + **Input**: Student’s first name, last name, roll number.
  + **Output**: Unique student\_id assigned.
  + **Constraints**: Roll number must be unique.

**3.2 Subject Management**

* **Description**: Define subjects and assign them to teachers.
* **Input**: Subject name, teacher ID.
* **Output**: Unique subject\_id assigned.
* **Constraints**: Each subject is linked to one teacher.

**3.3 Timetable Management**

* **Description**: Create a centralized timetable for scheduling classes.
* **Input**: Subject ID, day of week, start time, end time, room (optional).
* **Output**: Unique timetable\_id assigned.
* **Constraints**: No overlapping sessions for the same subject or room (optional enforcement).

**3.4 Session Management**

* **Description**: Generate and manage specific class sessions based on the timetable.
* **Input**: Timetable ID, date.
* **Output**: Unique session\_id assigned, status (Scheduled/Completed/Cancelled).
* **Constraints**: Sessions must align with the timetable’s day and time.

**3.5 Attendance Management**

* **3.5.1 Record Attendance**
  + **Description**: Allow teachers to record attendance for their sessions.
  + **Input**: Session ID, student ID, status (Present/Absent/Late).
  + **Output**: Attendance record with attendance\_id and timestamp.
  + **Constraints**: Only the assigned teacher can record attendance.
* **3.5.2 Edit/Update Attendance**
  + **Description**: Allow teachers to modify attendance records.
  + **Input**: Attendance ID, updated status.
  + **Output**: Updated record in the database.
  + **Constraints**: Only the assigned teacher can edit their session’s records.
* **3.5.3 Delete Attendance**
  + **Description**: Allow teachers to delete attendance records.
  + **Input**: Attendance ID.
  + **Output**: Record removed from the database.
  + **Constraints**: Only the assigned teacher can delete their session’s records.

**3.6 Access Control**

* **Description**: Restrict teachers to their own subjects and sessions.
* **Input**: Teacher login credentials (e.g., email).
* **Output**: Filtered view of sessions and attendance records based on teacher\_id.
* **Constraints**: Teachers cannot access or modify data for other teachers’ sessions.

**4. Data Requirements**

**4.1 Database Schema**

The system uses a relational database with the following tables:

1. **Teachers**
   * teacher\_id (INT, PK, Auto-increment)
   * first\_name (VARCHAR(50), NOT NULL)
   * last\_name (VARCHAR(50), NOT NULL)
   * email (VARCHAR(100), UNIQUE)
2. **Students**
   * student\_id (INT, PK, Auto-increment)
   * first\_name (VARCHAR(50), NOT NULL)
   * last\_name (VARCHAR(50), NOT NULL)
   * roll\_number (VARCHAR(20), UNIQUE, NOT NULL)
3. **Subjects**
   * subject\_id (INT, PK, Auto-increment)
   * subject\_name (VARCHAR(100), NOT NULL)
   * teacher\_id (INT, FK → Teachers)
4. **Timetable**
   * timetable\_id (INT, PK, Auto-increment)
   * subject\_id (INT, FK → Subjects)
   * day\_of\_week (ENUM: Monday–Sunday, NOT NULL)
   * start\_time (TIME, NOT NULL)
   * end\_time (TIME, NOT NULL)
   * room (VARCHAR(50))
5. **Sessions**
   * session\_id (INT, PK, Auto-increment)
   * timetable\_id (INT, FK → Timetable)
   * date (DATE, NOT NULL)
   * status (ENUM: Scheduled/Completed/Cancelled, DEFAULT: Scheduled)
6. **Attendance**
   * attendance\_id (INT, PK, Auto-increment)
   * student\_id (INT, FK → Students)
   * session\_id (INT, FK → Sessions)
   * status (ENUM: Present/Absent/Late, DEFAULT: Absent)
   * timestamp (DATETIME, DEFAULT: Current timestamp)
7. **Student\_Subject** (Optional)
   * student\_id (INT, FK → Students)
   * subject\_id (INT, FK → Subjects)
   * Composite PK: (student\_id, subject\_id)

**4.2 Relationships**

* Teachers → Subjects (1-to-many)
* Subjects → Timetable (1-to-many)
* Timetable → Sessions (1-to-many)
* Students → Attendance (1-to-many)
* Sessions → Attendance (1-to-many)
* Students ↔ Subjects (many-to-many via Student\_Subject)

**5. Non-Functional Requirements**

**5.1 Performance**

* The system should handle attendance recording for up to 100 students per session without delay.
* Session generation from the timetable should occur within 5 seconds.

**5.2 Security**

* Teacher access restricted to their own data via authentication.
* Database protected against unauthorized access (e.g., SQL injection).

**5.3 Scalability**

* Should support multiple departments or colleges with minimal schema changes.

**5.4 Usability**

* Interface (TBD) should allow teachers to take attendance in fewer than 5 clicks per session.

**6. Future Scope**

* Attendance reports (e.g., by student, subject, or month).
* Student portal to view their own attendance.
* Notifications for absent students or timetable changes.

**7. Assumptions and Dependencies**

* A front-end application will enforce access control and provide a user interface.
* The database will be implemented in a relational DBMS (e.g., MySQL, PostgreSQL).
* Timetable updates require manual admin intervention.