**Student Management System**

## Goal:

To build a role-based system where **Admins**, **Teachers**, and **Students** can perform specific operations like registering, verifying users, marking/viewing attendance, requesting changes, and managing classes.

1. **User Roles & Functionalities:**

2.1. **Admin**

1. Can verify and update any user
2. Can assign teachers to classrooms
3. Can view system-wide attendance reports

2.2. **Teacher**

1. Can view students in their assigned classroom
2. Can mark attendance
3. Can view attendance history
4. Can handover class to another teacher
5. Can promote students to the next academic year

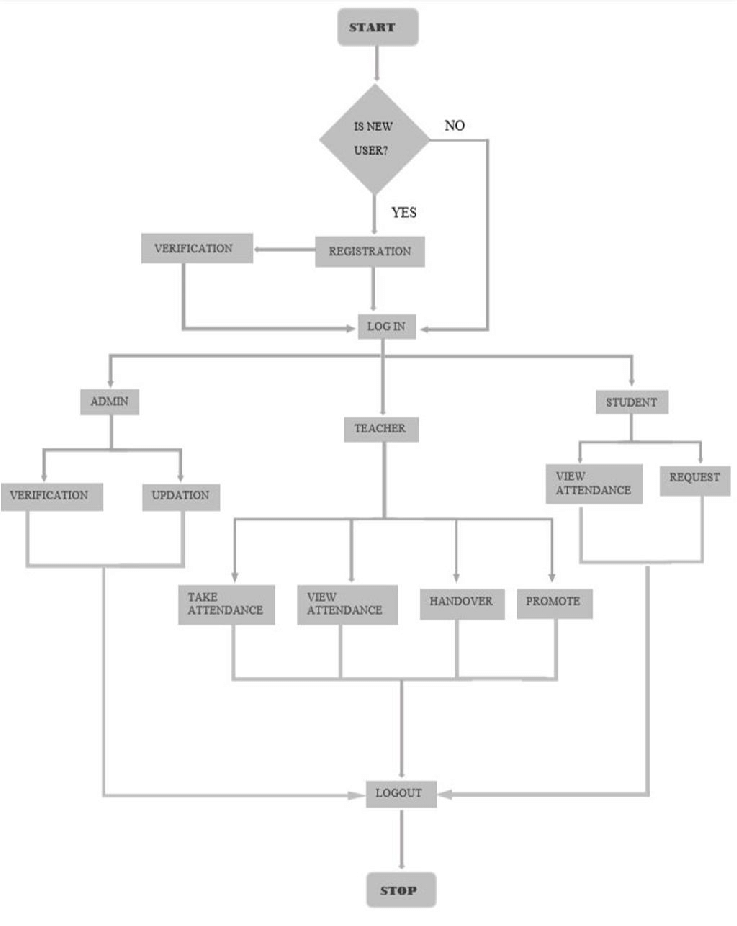
2.3 . **Student**

1. Can view own attendance
2. Can request corrections if attendance is incorrect

## 3.Tech Stack Overview

| **Layer** | **Tool** |
| --- | --- |
| Frontend | React.js, Axios, React Router |
| Backend | Django, Django REST Framework |
| Auth | JWT (SimpleJWT) |
| Database | MySQL |

**4.FLOWCHART**



## 5.Database Models (Conceptual)

| **Entity** | **Attributes** |
| --- | --- |
| **User** | id, name, email, password, role (admin/teacher/student), is\_verified |
| **Classroom** | id, name, assigned\_teacher |
| **StudentProfile** | user (FK), classroom (FK) |
| **Attendance** | student (FK), date, status, recorded\_by (FK) |
| **AttendanceRequest** | student (FK), date, reason, status (pending/approved/rejected) |
| **Handover** | classroom, from\_teacher, to\_teacher, date |
| **Promotion** | student, from\_class, to\_class, date |

1. **Users api endpoints ( http://127.0.0.1:8000 )**

| **Method** | **Endpoint** | **Description** | **Required Payload (JSON)** | **Response Example** |
| --- | --- | --- | --- | --- |
| POST | /api/register/ | Register a new user (student/teacher) | { "id": "T001", "name": "Alice", "email": "alice@example.com", "password": "pass123", "role": "teacher" } | { "message": "User registered successfully." } |
| POST | /api/login/ | Login and receive JWT token | { "id": "T001", "password": "pass123" } | { "access": "...", "refresh": "..." } |
| GET | /api/students/ | Get all students (Auth required) | **Header:** Authorization: Bearer <access\_token> | [{"id": "S001", "name": "John", "email": "john@example.com"}] |
| GET | /api/teachers/ | Get all teachers (Auth required) | **Header:** Authorization: Bearer <access\_token> | [{"id": "T001", "name": "Alice", "email": "alice@example.com"}] |
| DEL | /api/delete-user/<str:user\_id>/ | Delete Specified User  (Auth required) | **Header :**  Authorization: Bearer <access\_token> | {"detail": "User deleted successfully."  } |

1. **Attendance api endpoints**

Summary of Endpoints

| Endpoint | Method | Use Case | Required Payload |
| --- | --- | --- | --- |
| /classrooms/create/ | POST | Create a new classroom and add students from an Excel file. | {  “classroom\_name”:”TYCORE5-SP”,  “subject\_id”:”2”,  “teacher\_id”:”TOO1”,  “student\_excel”:TYCORE5.xlsx  } |
| /classrooms/ | GET | Get a list of all classrooms. |  |
| /classrooms/<int:classroom\_id>/ | GET | Get details for a specific classroom including attendance. |  |
| /classrooms/<int:classroom\_id>/update/ | PUT/PATCH | Update details of a specific classroom. | {  provide updated data} |
| /classrooms/<int:classroom\_id>/delete/ | DELETE | Delete a specific classroom and its related attendance records. |  |
| /mark/ | POST | Mark attendance for students in a classroom. | {  "classroom\_id": 1,  "date": "2025-04-01",  "attendance": [  {"student\_id": 123, "status": "present"},  {"student\_id": 124, "status": "absent"}  ]  } |
| /classrooms/<int:classroom\_id>/attendance/ | GET | Get the attendance records for a classroom. |  |
| /corrections/create/ | POST | Submit an attendance correction request. | {  "attendance\_id": 1,  "student\_id": 123,  "reason": "I was present but my attendance was marked absent."  } |
| /corrections/ | GET | View all attendance correction requests. |  |
| /corrections/<int:request\_id>/ | PUT | Approve or reject a correction request. | {  "status": "approved"  } |
| /subjects/create/ | POST | Create a new subject. | {  “code”:”21BTCS201”,  “name”:”OS”  } |
| /subjects/ | GET | Get a list of all subjects. | None |