Hide School Management System (HSMS)

1. ◆ Architecture

- Frontend (Vue.js) → Runs in the browser (User Interface)
- Backend (Python) → Handles API, database, logic
- Communication happens via HTTP requests (REST API or GraphQL), usually with JSON.

High School Management System (Plan)

Core Modules (Features)

A typical HSMS has these modules:

1. Authentication & Roles

- Admin (full access)
- Teacher (manage classes, grades, attendance)
- Student (view timetable, results, assignments)
- Parent (track student performance)

2. Student Management

- o Enroll new students
- Manage profiles (name, DOB, address, guardian, etc.)
- Class assignments

3. Teacher Management

- Add teachers
- Assign teachers to classes & subjects

4. Class & Subjects

- o Create classes (Grade 10A, 10B, ...)
- o Assign subjects (Math, Physics, History, ...)

5. Attendance

- o Teachers mark student attendance
- Students/parents can view attendance reports

6. Grades & Exams

- o Enter exam marks
- Calculate GPA / final results
- Report cards

7. Timetable & Announcements

- Daily/weekly schedules
- Notifications for students & teachers

8. Reports & Dashboard

- o Student list, performance charts
- Attendance summary

Tech Stack

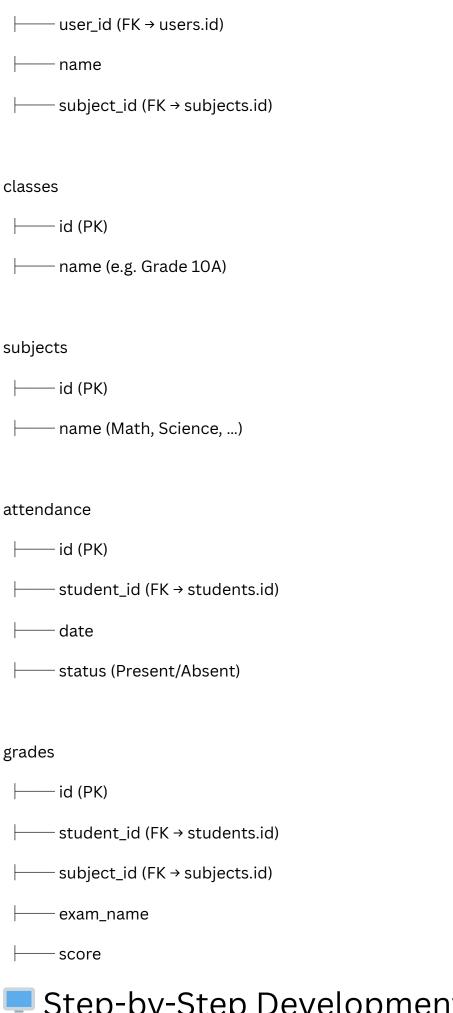
- Frontend (Vue.js 3 + Vite + TailwindCSS)
 - o For student/teacher/admin dashboards
 - Charts for reports (using chart.js or recharts)
- Backend (FastAPI Python)
 - REST API for students, teachers, classes, grades, etc.
 - Authentication (JWT tokens)
- Database (MariaDB/MySQL)
 - o Tables: students, teachers, users, classes, subjects, grades, attendance

Database Design (Basic Schema)

Here's a simplified version:
users
id (PK)
username
password (hashed)
role (admin, teacher, student, parent)
students
id (PK)
l—— user_id (FK → users.id)
—— name
—— dob
├── class_id (FK → classes.id)

teachers

— id (PK)





Step 1: Backend (FastAPI + MariaDB)

- Create models for users, students, teachers, classes, subjects, attendance, grades.
- Implement APIs:
 - POST /auth/login
 - POST /students → add student
 - GET /students → list students
 - POST /attendance → mark attendance
 - POST /grades → add exam result
 - GET /grades/student/{id} → view student grades

Step 2: Frontend (Vue.js)

- Create separate dashboards:
 - Admin Dashboard → manage users, teachers, students
 - Teacher Dashboard → mark attendance, upload grades
 - o **Student Dashboard** → view timetable, grades, attendance
 - o **Parent Dashboard** → track child performance

Step 3: Authentication

- Implement JWT authentication in FastAPI
- Store JWT in Vue (localStorage)
- Show different menus based on user role

Step 4: Extras

- Use TailwindCSS for UI
- Use **Chart.js** for graphs (attendance %, grade trends)
- Add export feature (Excel/PDF report cards)

X Example Workflow

- 1. Admin adds a new student & assigns them to a class.
- 2. **Teacher** marks daily attendance & uploads exam grades.
- 3. **Student** logs in → sees attendance & results.
- 4. Parent logs in → sees child's progress.