School Management System - Documentation

1. Project Overview

Project Name: School Management System

Purpose: To digitalize and simplify the daily administrative and academic processes of a school such as managing students, teachers, classes, fees, results, and attendance.

2. Tech Stack

Frontend: HTML, CSS, Bootstrap, JavaScript, Razor (for .NET)

Backend: ASP.NET Core MVC / PHP / Django

Database: SQL Server / MySQL

Others: jQuery, AJAX, Entity Framework, Identity/Auth

3. Modules / Features

Admin Panel:
- Add/Edit/Delete Students, Teachers
- Assign Teachers to Classes
- Create Subjects and Timetables
- Manage Attendance
- View Reports (attendance, performance)
- Manage Fees

Student Panel:
- View Profile
- View Attendance
- Download Results
- Pay Fees
- View Timetable

Teacher Panel:
- Mark Attendance
- Enter Marks
- View Assigned Classes
- Upload Homework

4. Project Structure

/SchoolManagementSystem
|-- /Controllers
| |-- AdminController.cs
| |-- StudentController.cs
| |-- TeacherController.cs
|-- /Models

```
|   |-- Student.cs
|   |-- Teacher.cs
|   |-- Class.cs
|   |-- Subject.cs
|   |-- Attendance.cs
|   |-- Fee.cs
|-- /Views
|   |-- /Admin
|   |-- /Student
|   |-- /Teacher
|-- /wwwroot (CSS, JS, Images)
|-- appsettings.json
|-- Startup.cs
```

5. Installation Guide

Prerequisites:

- Visual Studio / VS Code

- SQL Server / MySQL

- .NET Core SDK / PHP / Python

Setup:

1. Clone or download the repo

2. Open project in Visual Studio

3. Configure appsettings.json with your DB connection

4. Run migrations: dotnet ef database update

5. Run the project: dotnet run

6. Authentication Roles

Admin: Full control over system

Teacher: Class management, mark attendance, enter marks

Student: View info, pay fees, download results

7. Database Tables

Students: Stores student profile info

Teachers: Stores teacher info

Classes: Class-wise data

Subjects: List of subjects

Attendance: Daily student attendance

Fees: Fee status records

Results: Marks per subject per student

8. Sample API Endpoints

GET /api/students - Get all students

POST /api/student/create - Add new student

GET /api/attendance/{id} - Get attendance by class id

POST /api/fee/pay - Fee payment logic

9. Testing

Unit tested controllers with xUnit/NUnit

Manual testing for:

- Student registration

- Attendance marking

- Fee payments

10. Screenshots

Add some UI screenshots like:

- Admin dashboard

- Student profile

- Fee payment page

- Attendance marking screen

11. FAQ

Q: Default login credentials?

A: admin@school.com / Password123

Q: How to reset password?

A: Click 'Forgot Password' on login screen.

12. Contribution Guide

1. Fork the repo

2. Create feature branches

3. Raise pull requests

4. Follow coding guidelines from CONTRIBUTING.md