Application Overview: SmartSchools ERP System

The database schema outlines a comprehensive, multi-tenant **School Enterprise Resource Planning (ERP) System**, likely developed as an ASP.NET MVC application. It's designed to manage every facet of a modern educational institution, from a single school branch to a large company overseeing multiple schools.

The system is modular, handling student lifecycle management, complex academic scheduling, a sophisticated transportation and fleet management system, financial accounting, human resources, and even an integrated social media and communication platform. The inclusion of tables for **ASP.NET Identity** (`AspNetUsers`, `AspNetRoles`) confirms that it is a .NET web application with a robust, role-based security model.

Core Modules & Functionality

The application's functionality can be broken down into several interconnected modules:

1. School & Company Administration

This is the foundational module for setting up the organizational structure.

- **Headquarters (`Headquarters`):** Manages the parent company or organization that owns the schools.
- **School Branches (`SchoolBranches`):** Allows the headquarters to manage multiple school locations. Each school has its own physical address, capacity, and settings.
- **System Settings (`SystemSettings`, `SchoolSettings`):** Provides granular control over application-wide and school-specific parameters, such as the academic year, timetable type, financial settings, and bus scheduling logic.

2. Student Information System (SIS)

This module is the heart of the application, managing all student-related data.

- **Student Registration (`Student`, `ExternalStudent`, `LiteStudentRegistration`):** Handles the entire student lifecycle, from initial lightweight online applications to full enrollment. It captures extensive personal data, photos, and document uploads.
- **Guardian & Family Management (`Guardians`, `Families`, `StudentGuardDetails`):** Manages parent/guardian information and links them to students, forming family units. This is crucial for communication and billing.
- **Academic Placement (`StudentSchoolDetails`):** Assigns students to a specific school, class (`SchoolClasses`), and section (`Sections`).

3. Academic Management

This module covers all aspects of the educational process.

- **Curriculum & Subjects (`Curriculums`, `Subjects`):** Defines the educational programs and the subjects offered within them, including details like passing marks and sessions per week.
- **Timetabling (`Timetable`, `Sessions`, `TimetableItems`):** A powerful feature for creating and managing class schedules. The schema supports both **manual** (`ManualTimetable`) and **automatic** (`AutomaticTimetable`) schedule generation.
- **Attendance Tracking (`Attendance`):** Logs student attendance daily and per session, including reasons for absence.
- **Exams & Grading (`Exams`, `Grades`):** Manages the creation of exams, recording of student grades, and calculation of results.
- **Homework Management (`HomeWork`):** Allows teachers to assign homework with deadlines and attachments.

4. Transportation Management System

This is a highly detailed module for managing the school's fleet and student transportation.

- **Bus & Fleet Management (`BusInfo`, `BusMaintenance`, `BusFuel`):** Tracks bus details, number of seats, driver assignments, maintenance schedules, and fuel consumption.
- **Tour & Route Planning (`BusTours`, `RoutingMatrix`):** Enables the creation of bus routes (tours) for different areas. The `RoutingMatrix` table suggests an advanced feature for calculating optimal travel distances and times between points.
- **Live Bus Scheduling & Tracking (`ScheduledBusTrips`, `BusTrackNavigation`):** This is the operational core. It generates daily trip schedules for each student and bus. The `BusTrackNavigation` table is designed to store real-time GPS data (latitude, longitude, speed), indicating a live bus tracking feature for parents and administrators.

5. Financial & Billing System

This module handles all financial transactions.

- **Accounts Management (`AccountsMaster`, `AccountsDetail`):** A complete double-entry accounting system for managing the school's chart of accounts, tracking debits, credits, and balances.
- **Fee Management (`Fees`, `StudentFees`):** Defines different types of fees (tuition, transport, etc.) and assigns them to students.
- **Billing & Payments (`StudentPayments`, `GuardianContractInvoice`):** Manages guardian payments, tracks installments, and generates invoices.
- **Discounts (`Discounts`):** A flexible system for applying various types of discounts to student fees.

6. Human Resources (HR) Management

Manages all school staff, not just teachers.

- **Staff Profiles (`Staff`, `Teachers`):** Stores comprehensive information about all employees.

- **Job & Contact Details (`StaffJobDetails`, `StaffContactDetails`):** Tracks job-related information like department, designation, and contact details.
- **Salary & Payroll (`StaffSalaryDetails`, `StaffBankDetails`):** Manages salary components, allowances, and bank details for payroll processing.

7. Communication & Social Platform

This modern module enhances engagement between the school, students, and parents.

- **Notifications (`Notifications`):** A system for sending targeted alerts and messages to users.
- **Social Media (`SocialMediaPosts`, `SocialMediaLikes`, `SocialMediaComments`):** An internal social network where users can create posts, and others can like and comment on them.
- **Events & Polls (`SocialMediaEvents`, `SocialMediaPolls`):** Allows for the creation of school events and interactive polls within the social platform.

8. Security & User Management

Controls access to the system.

- **ASP.NET Identity (`AspNetUsers`, `AspNetRoles`, etc.):** Manages the core web application users (likely administrators and staff) with industry-standard authentication.
- **Custom User Management (`Users`, `UserGroup`, `Privilages`):** A custom, granular permission system that assigns users to groups and grants specific privileges (`Privilages`) to functions within the application.

Database Structure and Relationships (ERD)

The database is well-structured around a few central tables that link the different modules together. Here is a simplified, text-based representation of the key relationships:

- **`Headquarters` (The Company)**
  - Has many **`SchoolBranches`** (One-to-Many).
- **`SchoolBranches` (The School)**
  - Has many **`Staff`**.
  - Has many **`SchoolClasses`**.
  - Has many **`BusInfo`** (Buses).
  - Has many **`BusTours`** (Routes).
  - Has one **`SchoolSettings`** (One-to-One).
- **`Guardians` (The Parent/Guardian)**
  - Is linked to one or more **`Student`** records (One-to-Many).
- **`Student` (The Student)**
  - Belongs to one **`Guardian`**.
  - Has one **`StudentSchoolDetails`** record, which links the student to a specific `SchoolBranch`, `SchoolClass`, and `Section`.

    - Has many **`Attendance`** records.
    - Has many **`Grades`**.
    - Can be on a **`ScheduledBusTrips`**.
- **`Staff` (The Employee)**
    - Can be a **`Teachers`**.
    - Is assigned to a `SchoolBranch` and `Department` via **`StaffJobDetails`**.
    - Can be the `TeacherID` in a **`Timetable`** or **`HomeWork`**.
    - Can be the `DriverID` or `AttendantID` in **`BusInfo`** and **`ScheduledBusTrips`**.
- **The Academic Core Relationship:**
    - A `SchoolBranch` has many **`SchoolClasses`** (e.g., Grade 1, Grade 2).
    - A `SchoolClass` has many **`Sections`** (e.g., Section A, Section B).
    - A `Section` has many **`Students`** (linked via `StudentSchoolDetails`).
    - A `Section` has many **`Subjects`**.
    - A `Section` has a **`Timetable`**, which is made up of **`Sessions`** (time slots) where a **`Teacher`** teaches a **`Subject`**.
- **The Transportation Core Relationship:**
    - A `SchoolBranch` has many **`BusInfo`** (buses).
    - A `BusInfo` is assigned a **`DriverID`** (from `Staff`).
    - A **`BusTours`** (route) is created for an area and assigned a `BusNo`.
    - **`ScheduledBusTrips`** is the transactional table, linking a `Student` to a specific `BusNumber` for a specific `TripDate` and `Direction` (morning/afternoon), with a defined `PickupOrder`.
    - **`BusTrackNavigation`** records the real-time GPS location of a `BusNumber`.

1. Core Administration & Settings 🏢

This module contains the foundational models for setting up the school's organizational structure and operational rules.

`Headquarters`

Stores information about the parent company or educational group that owns and operates the schools.

- **`CompanyID`**: (PK, int) The unique identifier for the parent company.
- **`CompanyArabicName`** / **`CompanyEnglishName`**: (nvarchar) The name of the company in both Arabic and English.
- **`Country`** / **`City`** / **`Street`**: (int, nvarchar) The physical address of the company headquarters.
- **`ContactNo`** / **`Email`**: (nvarchar) Contact information for the company.
- **`Longitude`** / **`Latitude`**: (float) GPS coordinates for the headquarters' location.
- **`ProductKey`**: (nvarchar) A license or product key for the software installation, tied to the company.
- **`Photo`**: (image) The logo of the company.

`SchoolBranches`

Represents an individual school or branch managed by the parent company.

- **`SchoolID`**: (PK, int) The unique identifier for the school branch.
- **`CompanyID`**: (FK, int) Links the school back to its parent company in `Headquarters`.
- **`SchoolArabicName`** / **`SchoolEnglishName`**: (nvarchar) The name of the school.
- **`Country`**, **`City`**, **`Street`**, **`Longitude`**, **`Latitude`**: Address and location details for the school.
- **`SchoolCurrency`**: (int) The default currency used by the school for financial transactions.
- **`YardArea`** / **`ClassroomArea`**: (int) Physical dimensions of school facilities, likely used for capacity planning.
- **`MaxSchoolStudents`** / **`MaxStudentsinClassroom`**: (int) The maximum allowed student capacity for the school and for each class.
- **`CurrentNumberofStudents`**: (int) A running count of currently enrolled students.

`SchoolSettings`

Defines the specific operational rules and academic parameters for a single school.

- **`SchoolID`**: (PK, FK, int) Uniquely identifies the settings for a specific school in `SchoolBranches`.
- **`NumberofSessionsPerDay`**: (int) The total number of class periods in a school day.
- **`WeekStartingDay`**: (int) Defines the start of the school week (e.g., 0 for Sunday, 1 for Monday).
- **`StartingTime`** / **`FirstClassStartingTime`**: (datetime) The official start time of the school day and the first session.

- **`SessionDuration` / `BreakDuration`**: (int) The length of a class period and break time in minutes.
- **`NumberofSemesters`**: (int) The number of academic terms in a year (e.g., 2 or 3).

---

2. Student & Guardian Management 👨‍👩‍👧‍👦

These models manage the core data for students and their families.

`Student`

The central table for storing all personal and administrative information about a student.

- **`StudentID`**: (PK, nvarchar) The unique identifier for the student.
- **`StudentArabicName` / `StudentEnglishName`**: (nvarchar) The student's full name.
- **`NationalNumber` / `ResidencyNumber` / `PassportNumber`**: (nvarchar) Official identification numbers.
- **`Gender`**: (int) A code representing the student's gender (e.g., 1 for Male, 2 for Female).
- **`Nationality`**: (FK, int) A code linking to the `Countries` table.
- **`DateofBirth`**: (datetime) The student's birth date.
- **`GuardianID`**: (FK, int) Links the student to their primary guardian in the `Guardians` table.
- **`Photo_A`**: (nvarchar) Stores the filename of the student's photo.
- **`ComesWithSchoolBus` / `GoesWithSchoolBus`**: (int) Flags indicating if the student uses the bus service.

`Guardians`

Stores information about the student's parents or legal guardians.

- **`GuardianID`**: (PK, int) The unique identifier for the guardian.
- **`GuardianArabicName` / `GuardianEnglishName`**: (nvarchar) The guardian's full name.
- **`NationalNumber`**: (nvarchar) The guardian's official ID number, often used for login or identification.
- **`MobileNumber`**: (nvarchar) The primary contact number, crucial for communication and notifications.
- **`PaymentMethod` / `CreditCard...` fields**: Stores preferred payment methods and credit card details for fee payments.

`StudentSchoolDetails`

This is a critical linking table that assigns a student to their specific academic placement.

- **`StudentID`**: (PK, FK, nvarchar) Links to the `Student` table.
- **`SchoolID`**: (FK, int) The school the student is enrolled in.
- **`ClassID`**: (FK, int) The class (grade level) the student is in (links to `SchoolClasses`).
- **`SectionID`**: (FK, int) The specific section (e.g., Class 5A, 5B) of the class.
- **`Status`**: (int) The student's current enrollment status (e.g., Active, Withdrawn).

- **`ComingBusTourID` / `GoingBusTourID`**: (FK, int) Links the student to their specific morning and afternoon bus routes in `BusTours`.

---

3. Academic Management 📚

Models related to classes, subjects, schedules, and performance tracking.

`SchoolClasses`

Defines the grade levels available at a school (e.g., Grade 1, Grade 10).

- **`SchoolClassID`**: (PK, int) The unique identifier for the class.
- **`ClassID`**: (FK, int) A foreign key to a generic `Class` table, representing the grade level itself.
- **`SchoolID`**: (FK, int) The school where this class is offered.
- **`CurriculumID`**: (FK, int) The curriculum associated with this class.

`Sections`

Represents a specific classroom or group of students within a `SchoolClass`.

- **`SectionID`**: (PK, int) The unique identifier for the section.
- **`SchoolClassID`**: (FK, int) Links the section to its parent class level.
- **`ClassroomID`**: (FK, int) The physical classroom assigned to this section.
- **`SectionCode` / `SectionArabicName`**: (nvarchar) The name or code for the section (e.g., 'A', 'B').
- **`NumberofStudents`**: (int) The current number of students in this section.

`Subjects`

Stores the academic subjects taught in the school.

- **`SubjectID`**: (PK, int) The unique identifier for the subject.
- **`SchoolClassID`**: (FK, int) The class level for which this subject is intended.
- **`SubjectArabicName` / `SubjectEnglishName`**: (nvarchar) The name of the subject.
- **`MaxMark` / `FailMark`**: (int) The maximum and failing scores for the subject.
- **`NumberOfSessionsPerWeek`**: (int) The number of times the subject is taught per week, used for timetabling.

`Timetable` / `Sessions` / `TimetableItems`

A set of tables that work together to create the school's master schedule.

- **`Timetable`**: The main scheduling table.
  - **`SessionID`**: (FK, int) Links to a specific time slot in the `Sessions` table.
  - **`TimetableItemID`**: (FK, int) Links to a `TimetableItems` record, which defines *what* is being taught.

    - **`TeacherID`**: (FK, nvarchar) The teacher assigned to this session.
- **`Sessions`**: Defines the individual time slots in the schedule.
    - **`SessionID`**: (PK, int) Unique ID for the time slot.
    - **`WeekDay`**: (int) The day of the week for the session.
    - **`SessionDayOrder`**: (int) The period number (1st, 2nd, 3rd, etc.).
- **`TimetableItems`**: Defines a unique combination of a subject and teacher for a specific section.
    - **`TimetableItemID`**: (PK, int) Unique ID for this combination.
    - **`SubjectID`**: (FK, int) The subject being taught.
    - **`TeacherID`**: (FK, nvarchar) The teacher responsible for it.

`Exams` and **`Grades`**

Models for managing assessments and student scores.

- **`Exams`**:
    - **`ID`**: (PK, int) Unique ID for the exam.
    - **`SubjectID`**: (FK, int) The subject of the exam.
    - **`ExamTypeID`**: (FK, int) The type of exam (e.g., Midterm, Final).
    - **`TotalGrades`**: (decimal) The maximum possible score for the exam.
- **`Grades`**:
    - **`ID`**: (PK, int) Unique ID for the grade record.
    - **`ExamID`**: (FK, int) Links the grade to a specific exam.
    - **`StudentID`**: (FK, nvarchar) The student who received the grade.
    - **`GradeValue`**: (decimal) The actual score the student achieved.

---

4. Transportation Management System 🚌

A sophisticated set of models for managing the school's transportation fleet and operations.

`BusInfo`

Stores details for each bus in the school's fleet.

- **`BusID`**: (PK, int) The unique identifier for the bus.
- **`BusNo`**: (nvarchar) The official number or identifier of the bus.
- **`SchoolID`**: (FK, int) The school the bus belongs to.
- **`DriverID`**: (FK, nvarchar) The assigned driver from the `Staff` table.
- **`AttendantID`**: (FK, nvarchar) The assigned attendant.
- **`NumberofSeats`**: (int) The seating capacity of the bus.
- **`BusCurrentStatus`**: (int) A code indicating the bus's current state (e.g., Parked, On Tour, Maintenance).

`BusTours`

Defines a specific bus route or tour.

- **`TourID`**: (PK, int) The unique identifier for the tour.
- **`BusNo`**: (FK, nvarchar) The bus assigned to this tour.
- **`TourArabicName`** / **`TourEnglishName`**: (nvarchar) The name of the route (e.g., "City Center Route").
- **`TourDirection`**: (int) Indicates if it's a morning pickup (1) or afternoon drop-off (2).
- **`StratingTime`**: (datetime) The scheduled start time of the tour.

`ScheduledBusTrips`

The primary operational table for daily bus trips. It contains a record for every student on every trip, every day.

- **`TripPassengerID`**: (PK, bigint) The unique identifier for a single student's leg of a trip.
- **`BusNumber`**: (nvarchar) The bus used for the trip.
- **`TripDate`**: (nvarchar) The date of the trip.
- **`Direction`**: (int) The direction (pickup or drop-off).
- **`PassengerID`**: (FK, nvarchar) The `StudentID` of the passenger.
- **`PassengerOnBoard`** / **`PassengeroffBoard`**: (int) Flags (0 or 1) to indicate if the student has been picked up or dropped off.
- **`PassengeronBoardDateTime`** / **`PassengeroffBoardDateTime`**: (datetime) Timestamps for when the student boarded or alighted the bus.
- **`PickupOrder`**: (int) The sequence number for the student's stop on the route.

`BusTrackNavigation`

Designed to store real-time GPS tracking data from devices on the buses.

- **`MoveSequnce`**: (PK, bigint) Unique ID for a single GPS data point.
- **`BusNumber`**: (nvarchar) The bus sending the data.
- **`MoveDate`** / **`MoveTime`**: (nvarchar) Timestamp of the data point.
- **`Latitude`** / **`Longitude`**: (float) The real-time GPS coordinates of the bus.
- **`BusSpeed`**: (nvarchar) The speed of the bus at that moment.

---

5. User & Security Management 🔐

Models for authenticating and authorizing users within the application.

`AspNetUsers` / **`AspNetRoles`** / **`AspNetUserRoles`**

Standard tables from the **ASP.NET Identity framework**. They manage user accounts (usernames, hashed passwords), roles (e.g., "Admin", "Teacher"), and the mapping between them for the web application's primary users.

`Users`

A custom user table that seems to augment or work alongside ASP.NET Identity.

- **`UserID`**: (PK, int) A unique ID for the user.
- **`StaffID`**: (FK, nvarchar) Links the user account directly to an employee in the `Staff` table.
- **`UserName`** / **`Password`**: (nvarchar) Custom login credentials.
- **`UserType`**: (int) A code defining the type of user (e.g., Company Admin, School Admin).

`Groups` and **`Privilages`**

These tables implement a custom, granular role-based access control (RBAC) system.

- **`Groups`**: Defines user groups (e.g., "Finance Department", "Admissions Team").
- **`Privilages`**: Assigns specific permissions to a `GroupID` for a particular `FunctionID` (a feature in the application), such as Read, Write, or Delete access.