## **Module 3 Activity 2: Basic Application REST API Diagram**

## Saurabh Kale

IFT 458/554: Middleware Programming & Database Security

Dinesh Sthapit

Sep 24th, 2023

**A basic ERD diagram that connects a student to courses and student contacts.**

**A diagram of a student

Description automatically generated**

This Entity-Relationship Diagram (ERD) serves as the foundational structure for developing a comprehensive course administration system. The primary objective of this system is to enhance the efficiency of managing student and course-related information for students, instructors, and administrators. It achieves this by facilitating seamless data access and query capabilities. The ERD forms the basis for creating several APIs, each tailored to specific aspects of the system. Below, we delve deeper into the ERD and its constituent elements:

**Entities:**

**1. Student:**

- Represents individual students, encompassing their personal details such as names, gender, nationality, and date of birth.

- Distinguished by a unique "ID" attribute serving as the primary key.

**2. Course:**

- Encompasses academic courses available within the system, providing details such as course names, teachers, descriptions, credits, and start/end dates.

- Similarly identified by the "ID" attribute, which acts as the primary key.

**3. StudentCourse:**

- Plays a pivotal role in establishing a many-to-many relationship between students and courses.

- Operates as a bridging table, facilitating the connection between students and the courses they are registered for.

- Comprises foreign keys "StudentID" and "CourseID," aligning with the Student and Course entities, respectively.

**4. StudentContact:**

- Governs student contact information, including email addresses, phone numbers, and home addresses.

- Additionally tracks attributes such as status, creation date, and update date.

- Forges a one-to-many relationship with the Student entity through the "StudentID" foreign key.

**5. Email:**

- Houses student email addresses, intricately linked with the StudentContact entity.

- Incorporates attributes like "ID," "EmailAddress," status, and timestamp data for record-keeping.

**6. Phone:**

- Tied to the StudentContact entity, responsible for storing phone numbers associated with students.

- Encompasses data fields such as "ID," "Number," phone type, status, and timestamp.

**7. HomeAddress:**

- Contains data concerning students' residential addresses, including address details, city, state, zip code, and country.

- Features properties to monitor status, creation date, and updates, mirroring the characteristics of the other contact-related entities.

**Relationships:**

**- Relationship between Students and Courses:**

- Establishes a connection allowing students to be enrolled in multiple courses, with each course accommodating multiple students.

**- Student-StudentContact Relationship:**

- Facilitates a one-to-many relationship where each student can have multiple contact entries in the StudentContact entity, accommodating various communication methods.

**- Sub-Relationships of the StudentContact Entity:**

- The StudentContact entity maintains one-to-one relationships with the Email, Phone, and HomeAddress entities, ensuring comprehensive and well-organized contact information for each student.

In summary, this ERD serves as a comprehensive roadmap for creating a course management system proficient in handling student enrollment, course administration, and thorough management of student contact information. It serves as a valuable resource for constructing and implementing APIs critical to functions such as aggregating student lists, managing course registrations, and handling contact information.

**This flow diagram provides a fundamental visual representation of the interplay and connections between the Web User Interface (UI) and its corresponding backend RESTful API. The application encompasses prominent menus pertaining to user accounts, courses, and grades.**

**A diagram of a login page

Description automatically generated**

**Show the flow and interactions between the Web UI and the REST API. For instance, how does the 'Grades' section fetch data? Which REST API endpoint does it hit?**

**A diagram of a data flow

Description automatically generated**

1. **User Initiation:** The process commences when a user, utilizing a web-based interface (Web UI), chooses to access their academic grades for enrolled courses. This initiation occurs when the user navigates to the 'Grades' section within the Web UI, signaling their intention to retrieve specific academic information.

2. **Web UI to REST API Request:** To fulfill the user's request, the Web UI establishes communication with the backend server, typically functioning as a RESTful API. In doing so, the Web UI constructs an HTTP GET request. Within this request, a distinct URL is formed, for example, /api/grades/{userId}, where {userId} is a placeholder denoting the unique identifier of the presently logged-in user. The URL's significance lies in its role in directing the request to the precise endpoint within the REST API.

3. **REST API Processing:** The REST API, tailored for managing incoming HTTP requests, receives the GET request at the /api/grades/{userId} endpoint. This specific endpoint specializes in handling grade-related requests. As part of request processing, the REST API extracts the {userId} parameter from the URL. This parameter assumes a pivotal role in identifying the user whose grades are being requested.

4. **Data Retrieval from Database:** Having identified the user through {userId}, the REST API proceeds to interact with the database. More specifically, it formulates a query directed at the 'Grades' table residing within the database. This query aims to retrieve the academic grades linked to the user in question. The database, serving as the repository for academic records, is organized to efficiently store and retrieve such data.

5. **Response from REST API:** Subsequent to concluding the data retrieval process, the REST API packages the acquired grades data into an HTTP response. This response typically adopts a structured format, such as JSON (JavaScript Object Notation), or another format well-suited for data exchange. It functions as the conduit for conveying the academic grades data from the backend to the frontend.

6. **Display in Web UI:** The Web UI, having dispatched the GET request and patiently awaited a response, ultimately receives the JSON response from the REST API. Upon receipt, the Web UI unpacks the JSON data and proceeds to present it to the user. The Web UI's presentation layer is responsible for rendering this data in a user-friendly format. This could manifest as a table, list, or any other appropriate format that enhances the user's ability to review and comprehend their academic grades.

In summary, this detailed depiction provides an all-encompassing overview of the sequence of actions encompassed within the process of a user's request for and subsequent viewing of academic grades through a web-based interface. It delineates the essential steps, from the user's initial intent to the retrieval of backend data and eventual display within the user interface.