Day 7 – Final Integration and Testing (Entity Framework Core + Stored Procedures + API Testing)

Date: 22-10-2025

Objective: Learn to integrate Entity Framework Core with stored procedures and expose them via ASP.NET Core APIs. Validate functionality through structured API testing using tools like Postman or Swagger.

1. Review of Entity Framework Core Integration

Entity Framework (EF) Core is an Object-Relational Mapper (ORM) that enables .NET developers to work with a database using .NET objects, eliminating the need for most data-access code.

In this project, we have used:

- **Database First Approach** – The database is designed first, and EF Core models are generated from it.
- **DbContext Class** – It manages the database connection and mapping between classes and tables.
- **Stored Procedures with EF Core** – EF Core can execute stored procedures using FromSqlRaw() for read operations or ExecuteSqlRaw() for insert, update, and delete operations.

2. Project Components

1. **Database:**
    - StudentCourseDB containing two tables:
        - Students – Stores student details
        - Courses – Stores course information
    - Stored procedures:
        - GetAllStudents
        - sp_GetStudentById
        - sp_AddStudent
        - sp_UpdateStudent
        - sp_DeleteStudent
2. **Model Classes (in Models folder):**
    - Student.cs
    - Course.cs
    - These define the data structure in C# that maps to database tables.

3. **DbContext Class (ApplicationDbContext.cs):**
   - Contains DbSet<Student> and DbSet<Course>.
   - Manages database connections and configurations.
4. **Controller (StudentsController.cs):**
   - Contains API endpoints to perform CRUD operations using stored procedures and EF Core.

3. CRUD Operations with EF Core and Stored Procedures

A. Retrieve All Students

- **Purpose:** To fetch all student records along with their course details.
- **EF Core Code:**

```
var students = _context.Students.FromSqlRaw("EXEC GetAllStudents").ToList();
```

- **Key Point:** The stored procedure must return all columns defined in the Student model, including CourseId.

B. Retrieve Student by ID

- **Purpose:** To fetch a specific student record based on their ID.
- **EF Core Code:**

```
var student = _context.Students
    .FromSqlRaw("EXEC sp_GetStudentById @Id={0}", id)
    .AsEnumerable()
    .FirstOrDefault();
```

Validation: Return 404 if the student is not found.

C. Add a New Student

- **Purpose:** To insert a new student record into the database.
- **EF Core Code:**

```
_context.Database.ExecuteSqlRaw(
    "EXEC sp_AddStudent @Name={0}, @Age={1}, @Grade={2}, @CourseId={3}",
    student.Name, student.Age, student.Grade, student.CourseId
);
```

Validation: Check if all required fields are provided before executing.

D. Update Student Details

- **Purpose:** To update an existing student record.
- **EF Core Code:**

```
_context.Database.ExecuteSqlRaw(
    "EXEC sp_UpdateStudent @Id={0}, @Name={1}, @Age={2}, @Grade={3}, @CourseId={4}",
    id, student.Name, student.Age, student.Grade, student.CourseId
);
```

Validation: Ensure the student exists before updating.

E. Delete Student

- **Purpose:** To delete a student record from the database.
- **EF Core Code:**

```
_context.Database.ExecuteSqlRaw("EXEC sp_DeleteStudent @Id={0}", id);
```

Validation: Confirm that the record exists before deletion.

4. Testing the API

A. Using Swagger

1. Run the project (dotnet run).
2. Open Swagger at: http://localhost:<port>/swagger.
3. Test endpoints:
   - GET /api/students – Fetch all students.
   - GET /api/students/{id} – Fetch by ID.
   - POST /api/students – Add new student.
   - PUT /api/students/{id} – Update details.
   - DELETE /api/students/{id} – Delete a record.

B. Using Postman

1. Create a new collection for StudentApi.
2. Add requests for all endpoints with proper HTTP methods.
3. Use JSON body in POST and PUT requests like:

```
{
   "name": "John Doe",
   "age": 23,
   "grade": "A",
   "courseId": 2
}
```

5. Common Errors and Fixes

Error	**Cause**	**Fix**
CourseId missing in FromSql	Stored procedure not returning all columns	Include CourseId in SELECT statement
Invalid column name	Property name mismatch	Ensure property names match table columns
500 Internal Server Error	Missing validation or incorrect SQL syntax	Check stored procedure parameters and data types
Cannot connect to DB	Connection string issue	Verify appsettings.json connection string

6. Validation and Best Practices

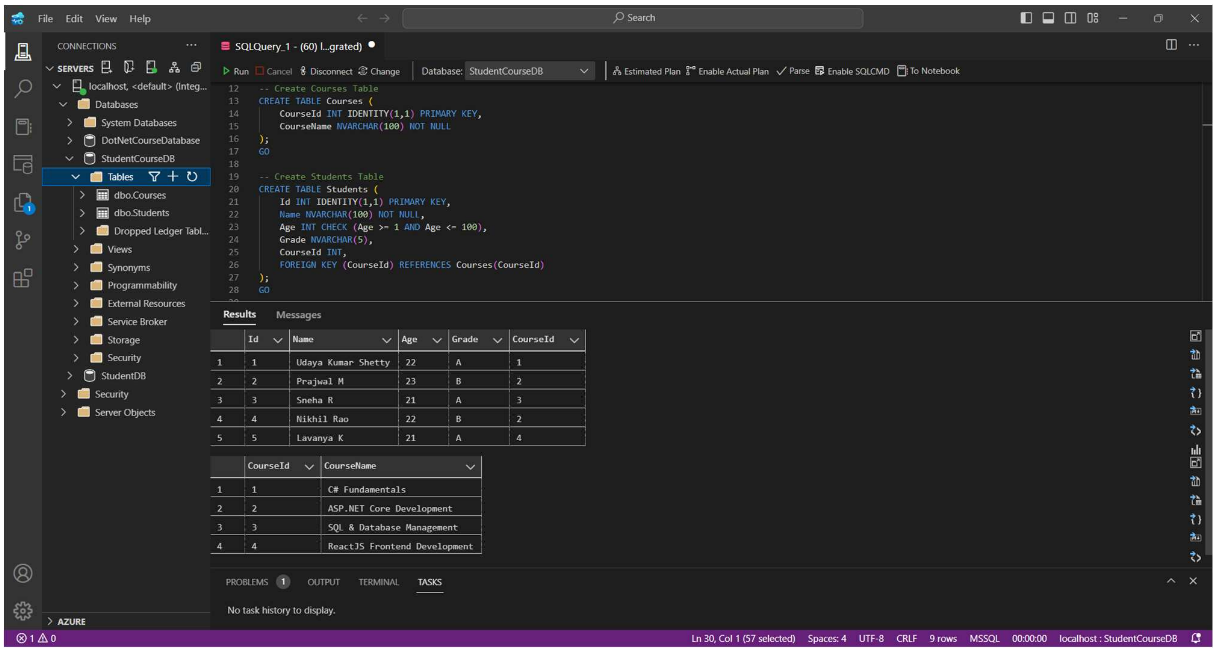
- Use **Data Annotations** in model classes for validation (e.g., [Required], [StringLength], [Range]).
- Implement **try-catch blocks** in controllers to handle exceptions gracefully.
- Ensure **stored procedures** have proper error handling using TRY-CATCH in SQL.
- Follow **RESTful conventions** for API routes and responses.
- Always **log errors** in case of failure for debugging.

8. Mini Exercise

Build and test all API endpoints:

1. Fetch all students.
2. Fetch student by ID.
3. Add a new student.
4. Update an existing student.
5. Delete a student record.

Snapshots :

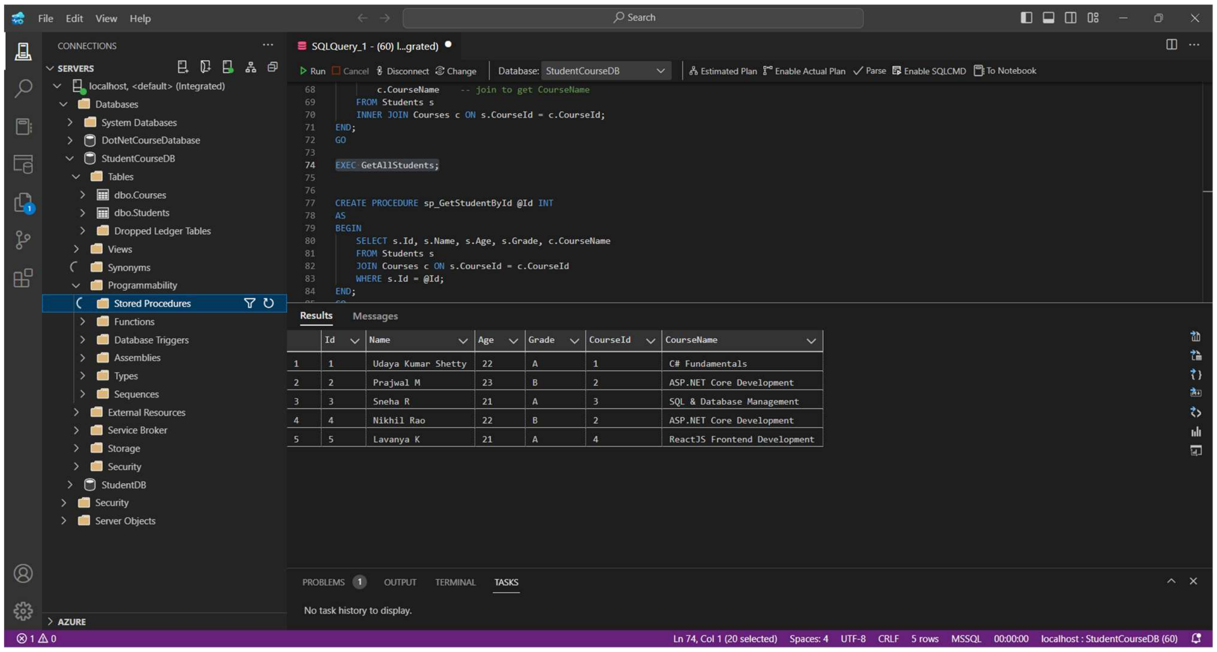


```sql
-- Create Courses Table
CREATE TABLE Courses (
    CourseId INT IDENTITY(1,1) PRIMARY KEY,
    CourseName NVARCHAR(100) NOT NULL
);
GO

-- Create Students Table
CREATE TABLE Students (
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(100) NOT NULL,
    Age INT CHECK (Age >= 1 AND Age <= 100),
    Grade NVARCHAR(5),
    CourseId INT,
    FOREIGN KEY (CourseId) REFERENCES Courses(CourseId)
);
GO
```

	Id	Name	Age	Grade	CourseId
1	1	Udaya Kumar Shetty	22	A	1
2	2	Prajwal M	23	B	2
3	3	Sneha R	21	A	3
4	4	Nikhil Rao	22	B	2
5	5	Lavanya K	21	A	4

	CourseId	CourseName
1	1	C# Fundamentals
2	2	ASP.NET Core Development
3	3	SQL & Database Management
4	4	ReactJS Frontend Development

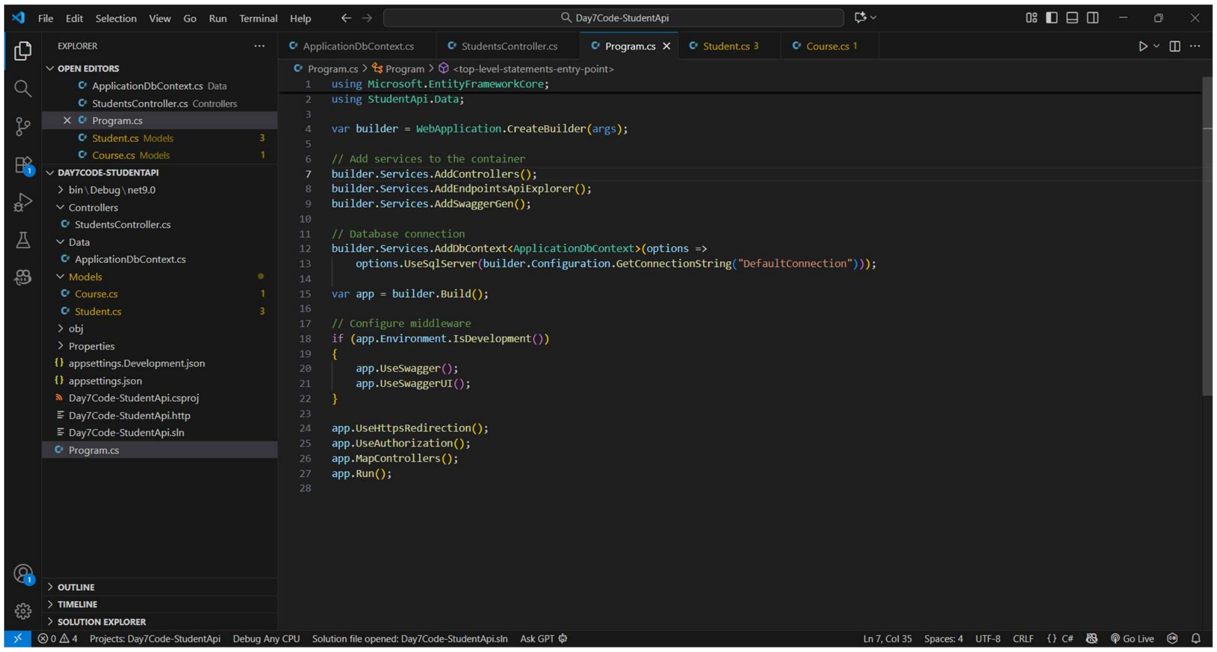
Database and tables created successfully in SSMS



```sql
            c.CourseName    -- join to get CourseName
        FROM Students s
        INNER JOIN Courses c ON s.CourseId = c.CourseId;
END;
GO

EXEC GetAllStudents;


CREATE PROCEDURE sp_GetStudentById @Id INT
AS
BEGIN
    SELECT s.Id, s.Name, s.Age, s.Grade, c.CourseName
    FROM Students s
    JOIN Courses c ON s.CourseId = c.CourseId
    WHERE s.Id = @Id;
END;
```

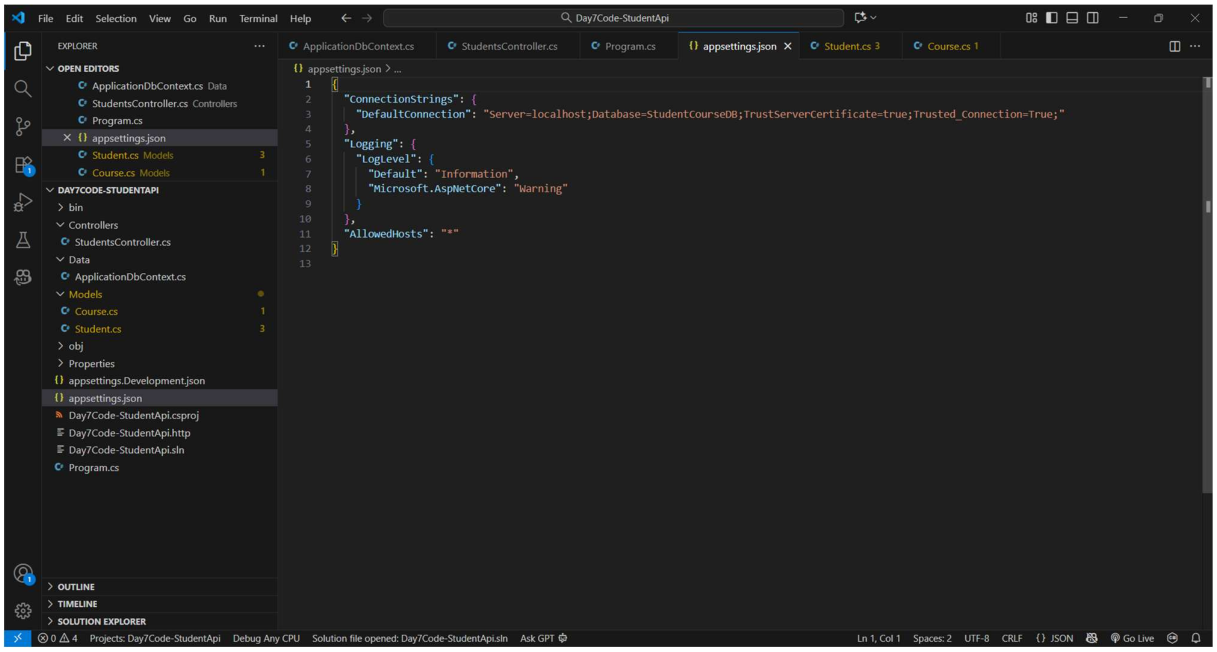
	Id	Name	Age	Grade	CourseId	CourseName
1	1	Udaya Kumar Shetty	22	A	1	C# Fundamentals
2	2	Prajwal M	23	B	2	ASP.NET Core Development
3	3	Sneha R	21	A	3	SQL & Database Management
4	4	Nikhil Rao	22	B	2	ASP.NET Core Development
5	5	Lavanya K	21	A	4	ReactJS Frontend Development

Stored procedures verified and executed in SSMS (sp_GetStudents)



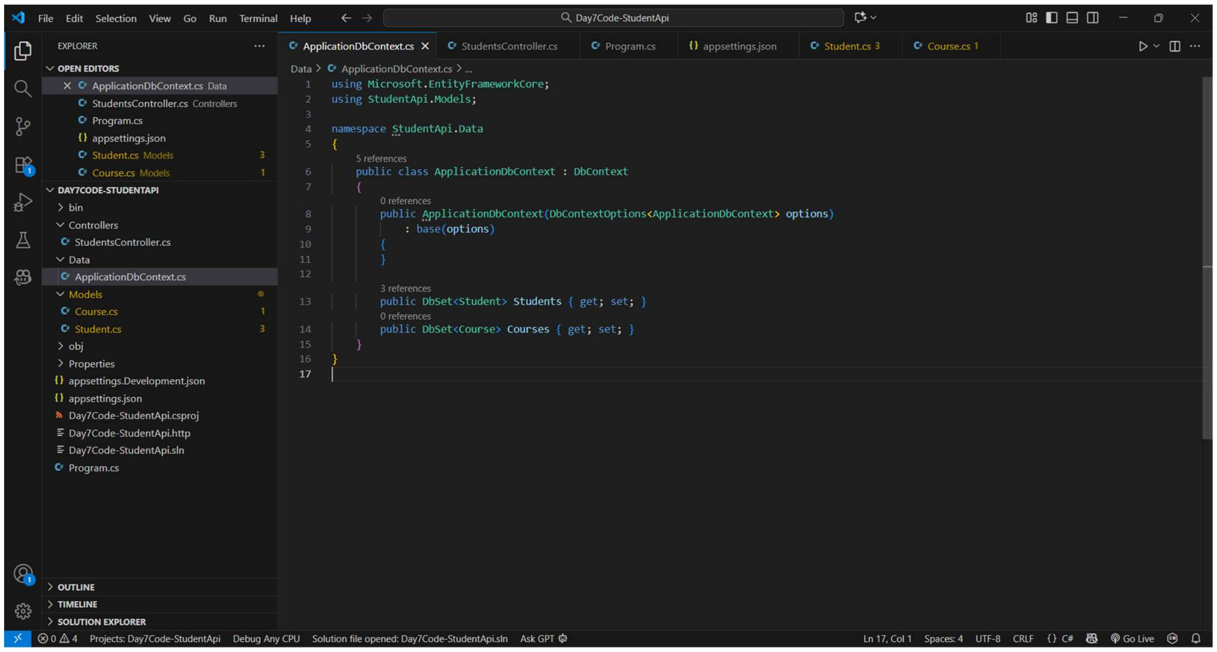
```csharp
using Microsoft.EntityFrameworkCore;
using StudentApi.Data;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// Database connection
builder.Services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();

// Configure middleware
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseHttpsRedirection();
app.UseAuthorization();
app.MapControllers();
app.Run();
```

Project structure shown in Visual Studio



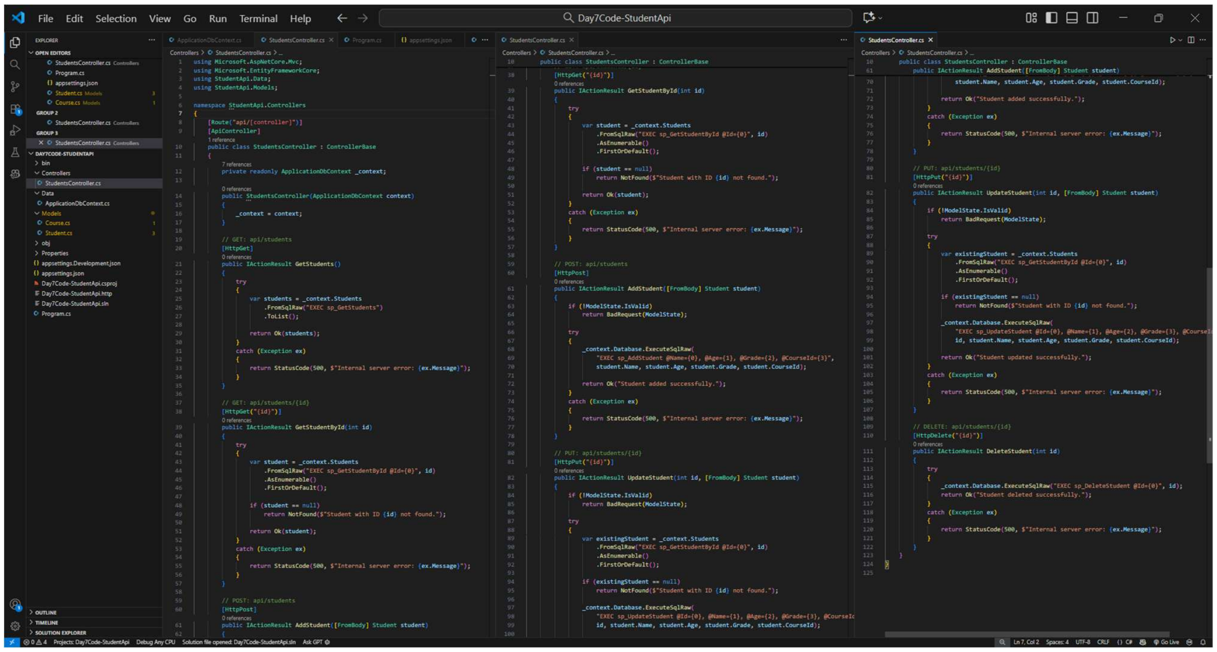
```json
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=localhost;Database=StudentCourseDB;TrustServerCertificate=true;Trusted_Connection=True;"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*"
}
```

Connection string configured in appsettings.json



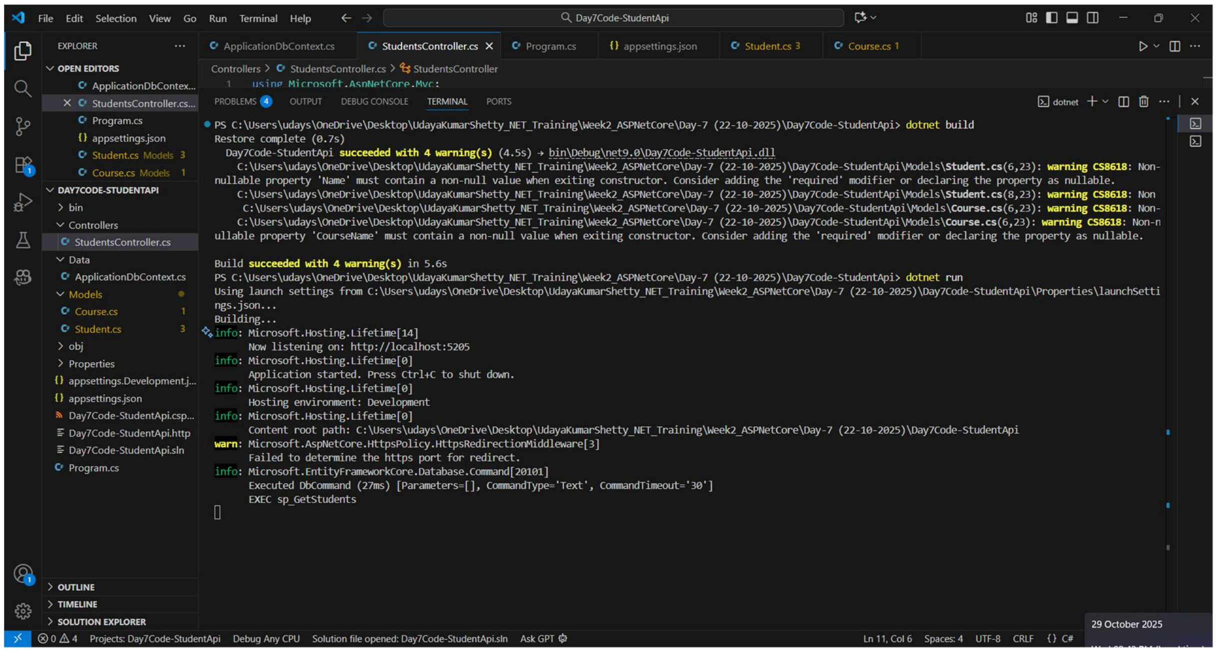
```csharp
using Microsoft.EntityFrameworkCore;
using StudentApi.Models;

namespace StudentApi.Data
{
    5 references
    public class ApplicationDbContext : DbContext
    {
        0 references
        public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
            : base(options)
        {
        }

        3 references
        public DbSet<Student> Students { get; set; }
        0 references
        public DbSet<Course> Courses { get; set; }
    }
}
```

ApplicationDbContext class with DbSets for Student and Course


Day7Code-StudentApi
StudentsController.cs
Controllers > StudentsController.cs
Solution file opened: Day7Code-StudentApi.sln
Ln 7, Col 2

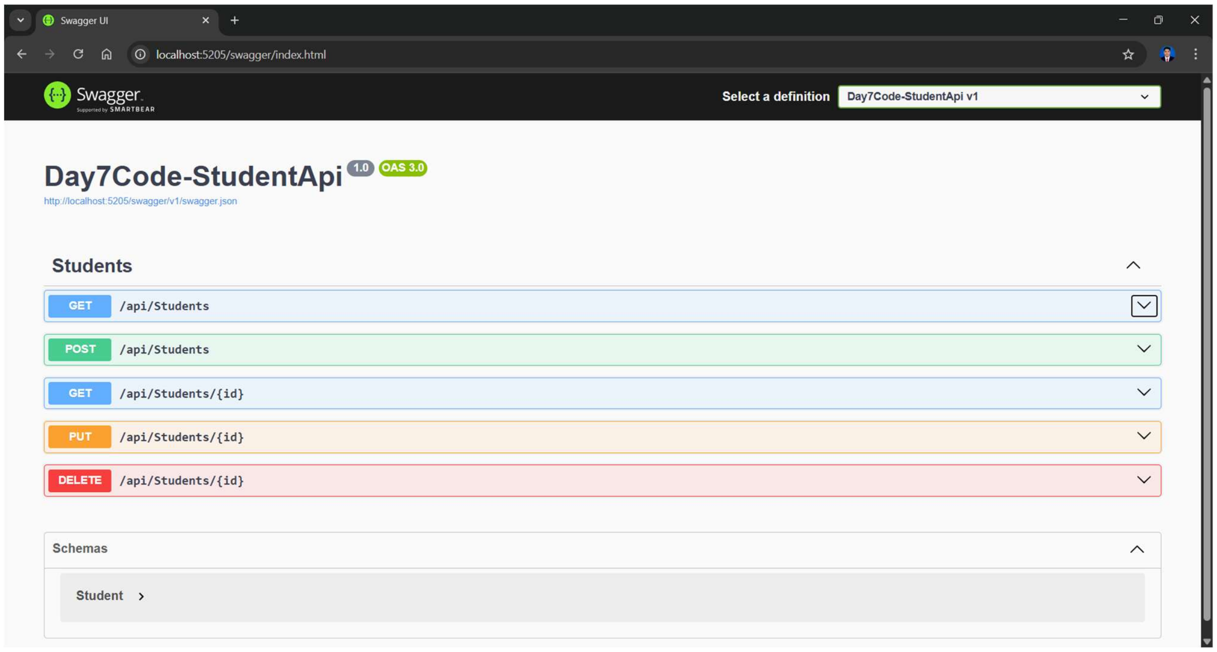
```csharp
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using StudentApi.Data;
using StudentApi.Models;

namespace StudentApi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    1 reference
    public class StudentsController : ControllerBase
    {
        7 references
        private readonly ApplicationDbContext _context;

        0 references
        public StudentsController(ApplicationDbContext context)
        {
            _context = context;
        }

        // GET: api/students
        [HttpGet]
        0 references
        public IActionResult GetStudents()
        {
            try
            {
                var students = _context.Students
                    .FromSqlRaw("EXEC sp_GetStudents")
                    .ToList();

                return Ok(students);
            }
            catch (Exception ex)
            {
                return StatusCode(500, $"Internal server error: {ex.Message}");
            }
        }

        // GET: api/students/{id}
        [HttpGet("{id}")]
        0 references
        public IActionResult GetStudentById(int id)
        {
            try
            {
                var student = _context.Students
                    .FromSqlRaw("EXEC sp_GetStudentById @Id={0}", id)
                    .AsEnumerable()
                    .FirstOrDefault();

                if (student == null)
                    return NotFound($"Student with ID {id} not found.");

                return Ok(student);
            }
            catch (Exception ex)
            {
                return StatusCode(500, $"Internal server error: {ex.Message}");
            }
        }

        // POST: api/students
        [HttpPost]
        0 references
        public IActionResult AddStudent([FromBody] Student student)
        {
            if (!ModelState.IsValid)
                return BadRequest(ModelState);

            try
            {
                _context.Database.ExecuteSqlRaw(
                    "EXEC sp_AddStudent @Name={0}, @Age={1}, @Grade={2}, @CourseId={3}",
                    student.Name, student.Age, student.Grade, student.CourseId);

                return Ok("Student added successfully.");
            }
            catch (Exception ex)
            {
                return StatusCode(500, $"Internal server error: {ex.Message}");
            }
        }

        // PUT: api/students/{id}
        [HttpPut("{id}")]
        0 references
        public IActionResult UpdateStudent(int id, [FromBody] Student student)
        {
            if (!ModelState.IsValid)
                return BadRequest(ModelState);

            try
            {
                var existingStudent = _context.Students
                    .FromSqlRaw("EXEC sp_GetStudentById @Id={0}", id)
                    .AsEnumerable()
                    .FirstOrDefault();

                if (existingStudent == null)
                    return NotFound($"Student with ID {id} not found.");

                _context.Database.ExecuteSqlRaw(
                    "EXEC sp_UpdateStudent @Id={0}, @Name={1}, @Age={2}, @Grade={3}, @CourseI
                    id, student.Name, student.Age, student.Grade, student.CourseId);

                return Ok("Student updated successfully.");
            }
            catch (Exception ex)
            {
                return StatusCode(500, $"Internal server error: {ex.Message}");
            }
        }

        // DELETE: api/students/{id}
        [HttpDelete("{id}")]
        0 references
        public IActionResult DeleteStudent(int id)
        {
            try
            {
                _context.Database.ExecuteSqlRaw("EXEC sp_DeleteStudent @Id={0}", id);
                return Ok("Student deleted successfully.");
            }
            catch (Exception ex)
            {
                return StatusCode(500, $"Internal server error: {ex.Message}");
            }
        }
    }
}
```

StudentsController implemented with CRUD endpoints



```
PS C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi> dotnet build
Restore complete (0.7s)
  Day7Code-StudentApi succeeded with 4 warning(s) (4.5s) → bin\Debug\net9.0\Day7Code-StudentApi.dll
    C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi\Models\Student.cs(6,23): warning CS8618: Non-
nullable property 'Name' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.
    C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi\Models\Student.cs(8,23): warning CS8618: Non
     C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi\Models\Course.cs(6,23): warning CS8618: Non-
    C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi\Models\Course.cs(6,23): warning CS8618: Non-n
ullable property 'CourseName' must contain a non-null value when exiting constructor. Consider adding the 'required' modifier or declaring the property as nullable.

Build succeeded with 4 warning(s) in 5.6s
PS C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi> dotnet run
Using launch settings from C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi\Properties\launchSetti
ngs.json...
Building...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5205
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\udays\OneDrive\Desktop\UdayaKumarShetty_NET_Training\Week2_ASPNetCore\Day-7 (22-10-2025)\Day7Code-StudentApi
warn: Microsoft.AspNetCore.HttpsPolicy.HttpsRedirectionMiddleware[3]
      Failed to determine the https port for redirect.
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (27ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      EXEC sp_GetStudents
```

dotnet build completed successfully in terminal



Day7Code-StudentApi 1.0 OAS 3.0

http://localhost:5205/swagger/v1/swagger.json

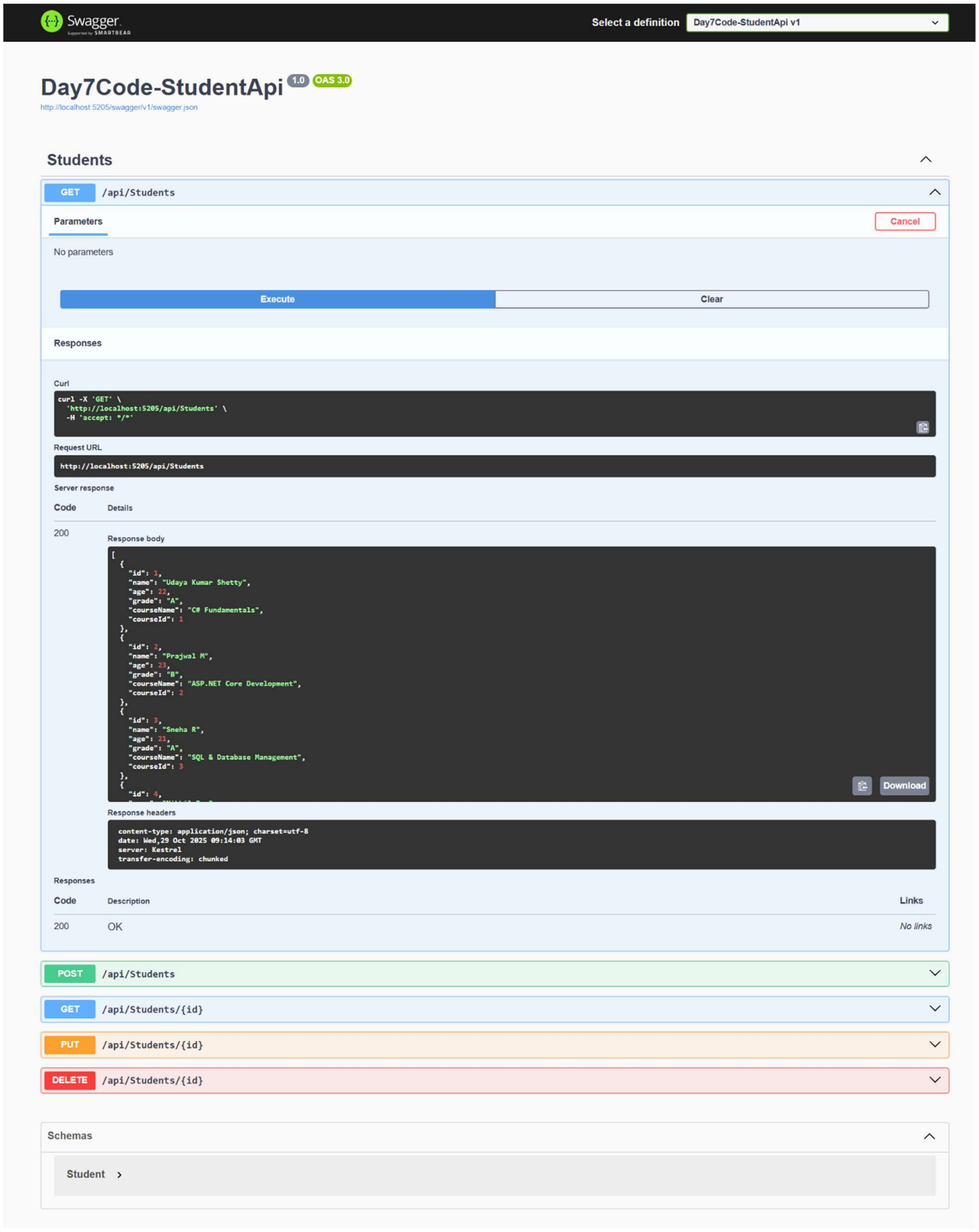
Students

GET	/api/Students
POST	/api/Students
GET	/api/Students/{id}
PUT	/api/Students/{id}
DELETE	/api/Students/{id}

Schemas

Student

Swagger UI loaded showing all API endpoints



Day7Code-StudentApi 1.0 OAS 3.0

http://localhost:5205/swagger/v1/swagger.json

Students

GET /api/Students

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5205/api/Students' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5205/api/Students
```

Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "name": "Udaya Kumar Shetty",
    "age": 22,
    "grade": "A",
    "courseName": "C# Fundamentals",
    "courseId": 1
  },
  {
    "id": 2,
    "name": "Prajwal M",
    "age": 23,
    "grade": "B",
    "courseName": "ASP.NET Core Development",
    "courseId": 2
  },
  {
    "id": 3,
    "name": "Sneha R",
    "age": 21,
    "grade": "A",
    "courseName": "SQL & Database Management",
    "courseId": 3
  },
  {
    "id": 4,
```

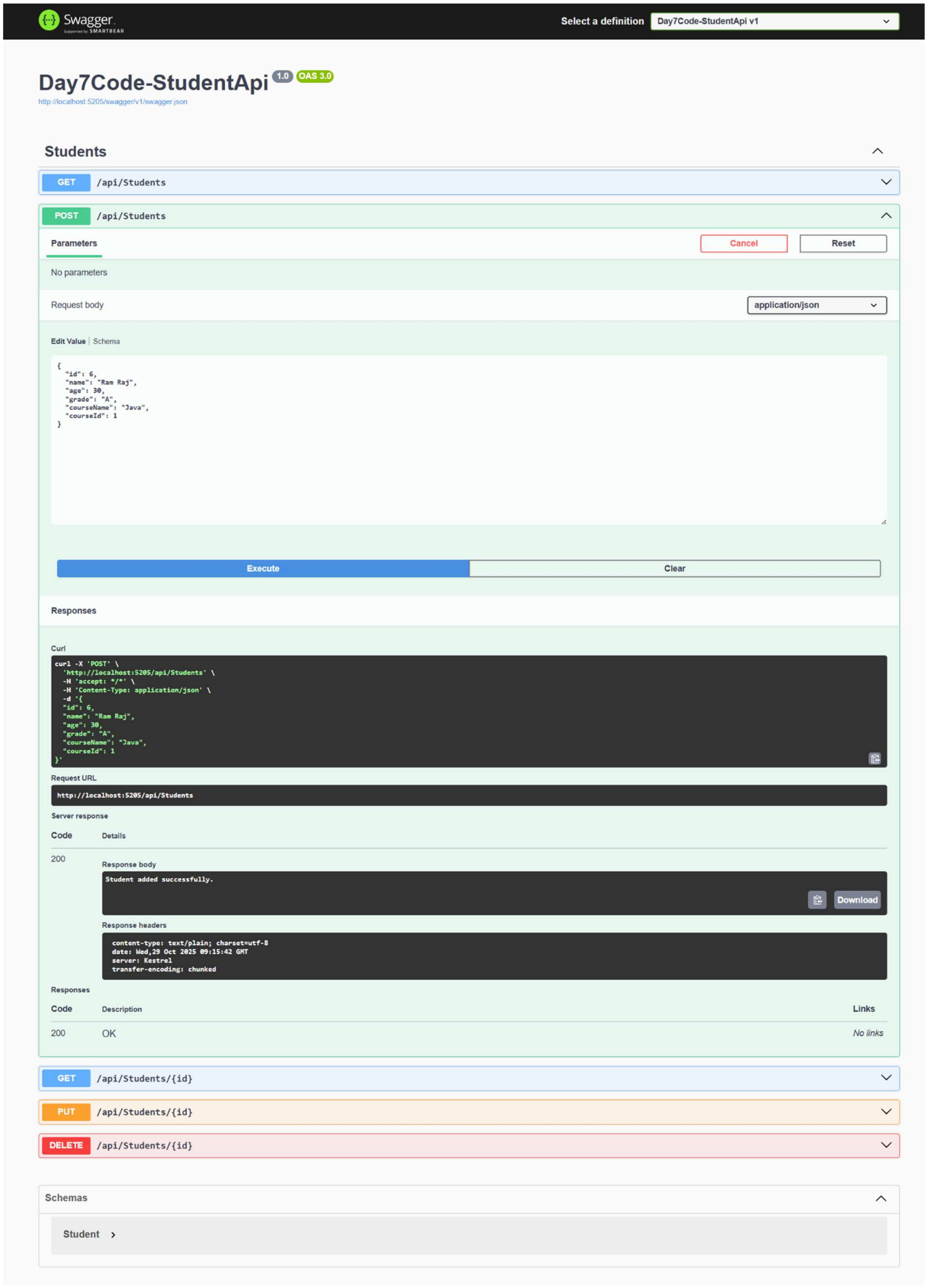
Response headers

```
content-type: application/json; charset=utf-8
date: Wed,29 Oct 2025 09:14:03 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	*No links*

GET /api/students returns full list of students



Day7Code-StudentApi 1.0 OAS 3.0

http://localhost:5205/swagger/v1/swagger.json

Students

GET /api/Students

POST /api/Students

No parameters

Request body: application/json

```
{
  "id": 6,
  "name": "Ram Raj",
  "age": 30,
  "grade": "A",
  "courseName": "Java",
  "courseId": 1
}
```

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:5205/api/Students' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 6,
  "name": "Ram Raj",
  "age": 30,
  "grade": "A",
  "courseName": "Java",
  "courseId": 1
}'
```

Request URL

```
http://localhost:5205/api/Students
```

Server response

Code: 200

Response body

```
Student added successfully.
```

Response headers

```
 content-type: text/plain; charset=utf-8
 date: Wed,29 Oct 2025 09:15:42 GMT
 server: Kestrel
 transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	*No links*

GET /api/Students/{id}

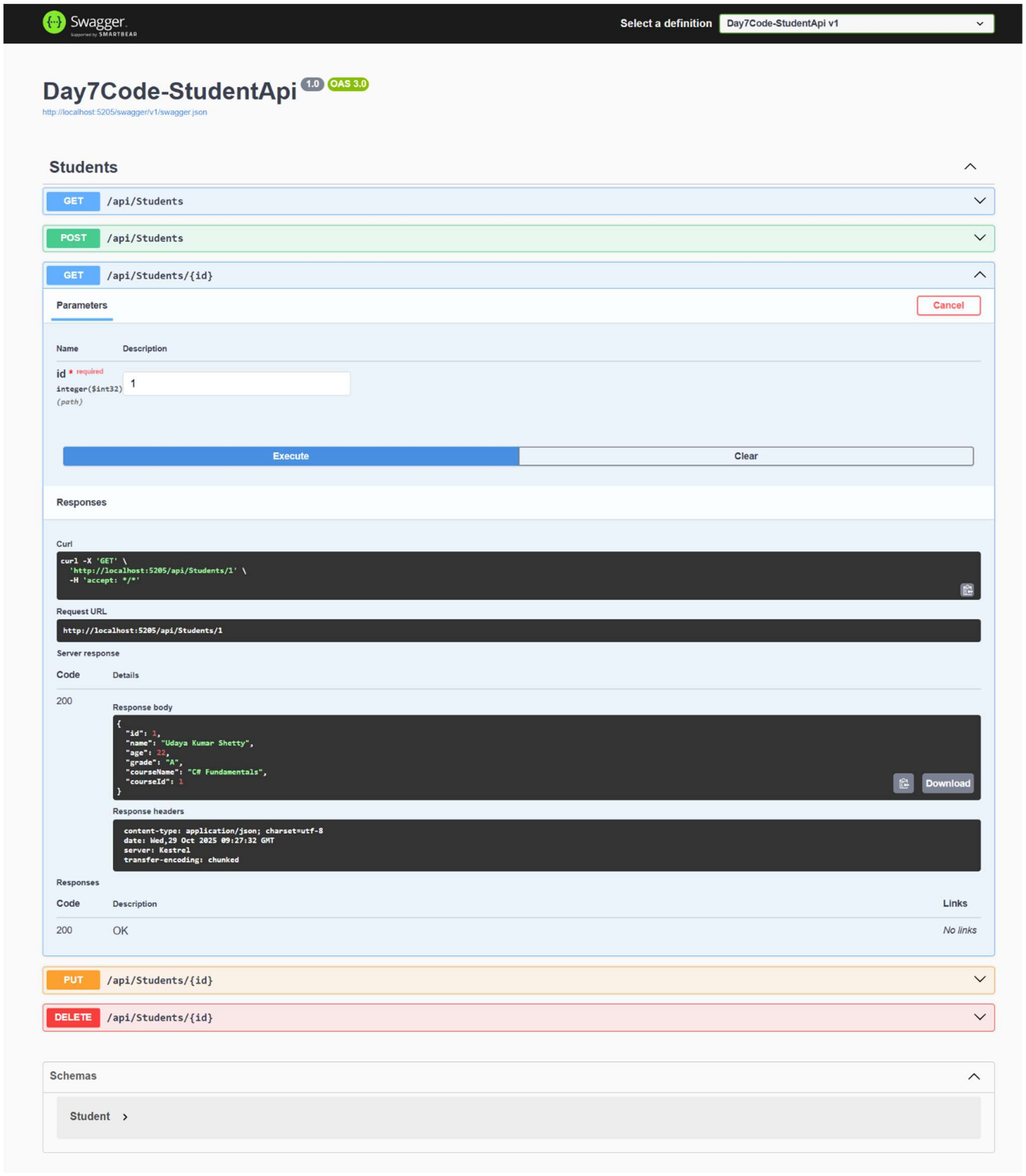
PUT /api/Students/{id}

DELETE /api/Students/{id}

Schemas

Student

POST /api/students adds new student successfully



Curl

```
curl -X 'GET' \
  'http://localhost:5205/api/Students/1' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5205/api/Students/1
```

Server response

Code	Details
200	Response body

```
{
  "id": 1,
  "name": "Udaya Kumar Shetty",
  "age": 22,
  "grade": "A",
  "courseName": "C# Fundamentals",
  "courseId": 1
}
```

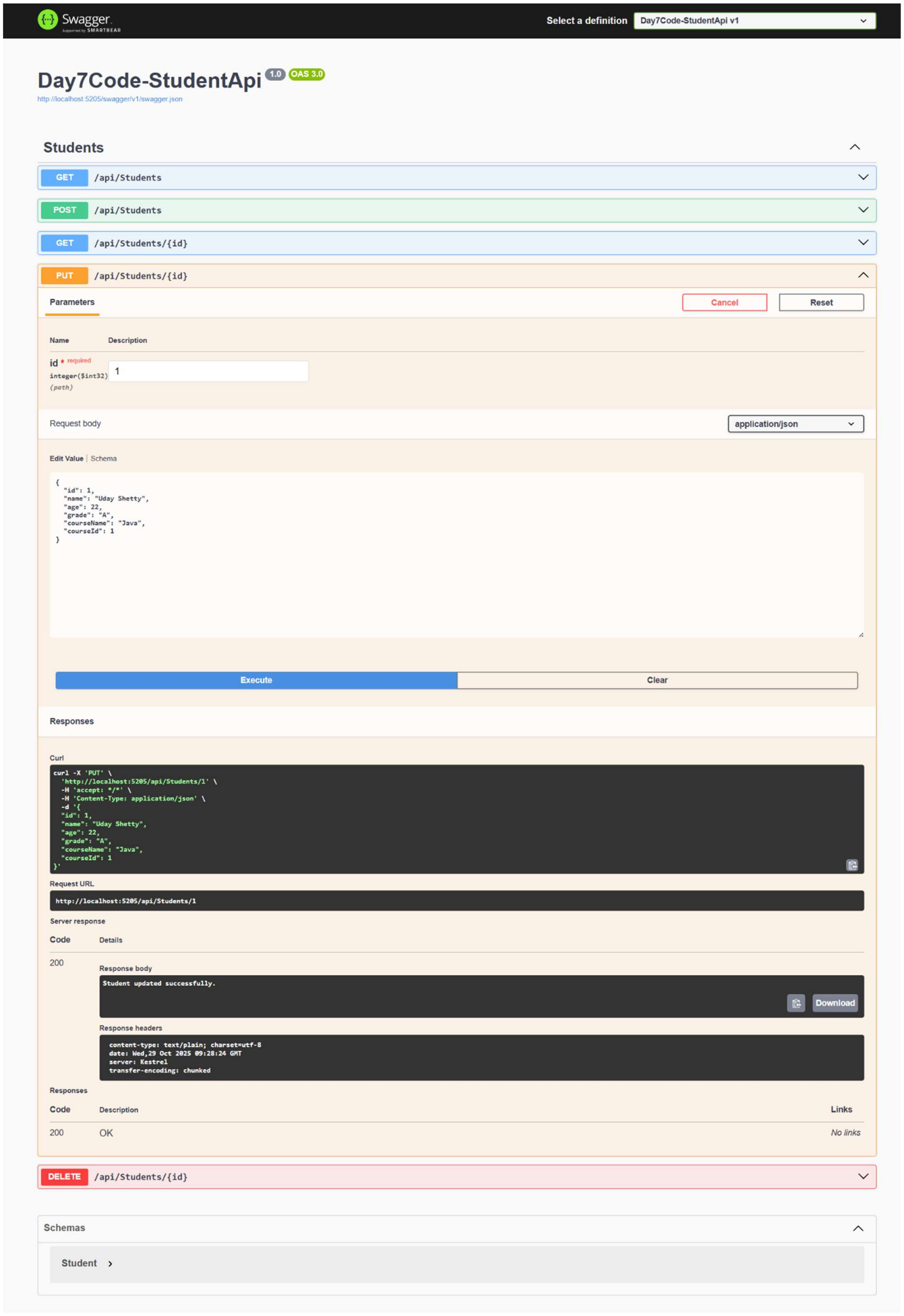
Response headers

```
 content-type: application/json; charset=utf-8
 date: Wed,29 Oct 2025 09:27:32 GMT
 server: Kestrel
 transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	*No links*

GET /api/students/{id} returns specific student



Day7Code-StudentApi

http://localhost:5205/swagger/v1/swagger.json

Students

GET /api/Students

POST /api/Students

GET /api/Students/{id}

PUT /api/Students/{id}

Parameters

Name	Description
id * required integer($int32) (path)	1

Request body

Edit Value | Schema

```
{
  "id": 1,
  "name": "Uday Shetty",
  "age": 22,
  "grade": "A",
  "courseName": "Java",
  "courseId": 1
}
```

Responses

Curl

```
curl -X 'PUT' \
  'http://localhost:5205/api/Students/1' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 1,
  "name": "Uday Shetty",
  "age": 22,
  "grade": "A",
  "courseName": "Java",
  "courseId": 1
}'
```

Request URL

```
http://localhost:5205/api/Students/1
```

Server response

Code 200

Response body

```
Student updated successfully.
```

Response headers

```
content-type: text/plain; charset=utf-8
date: Wed,29 Oct 2025 09:28:24 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

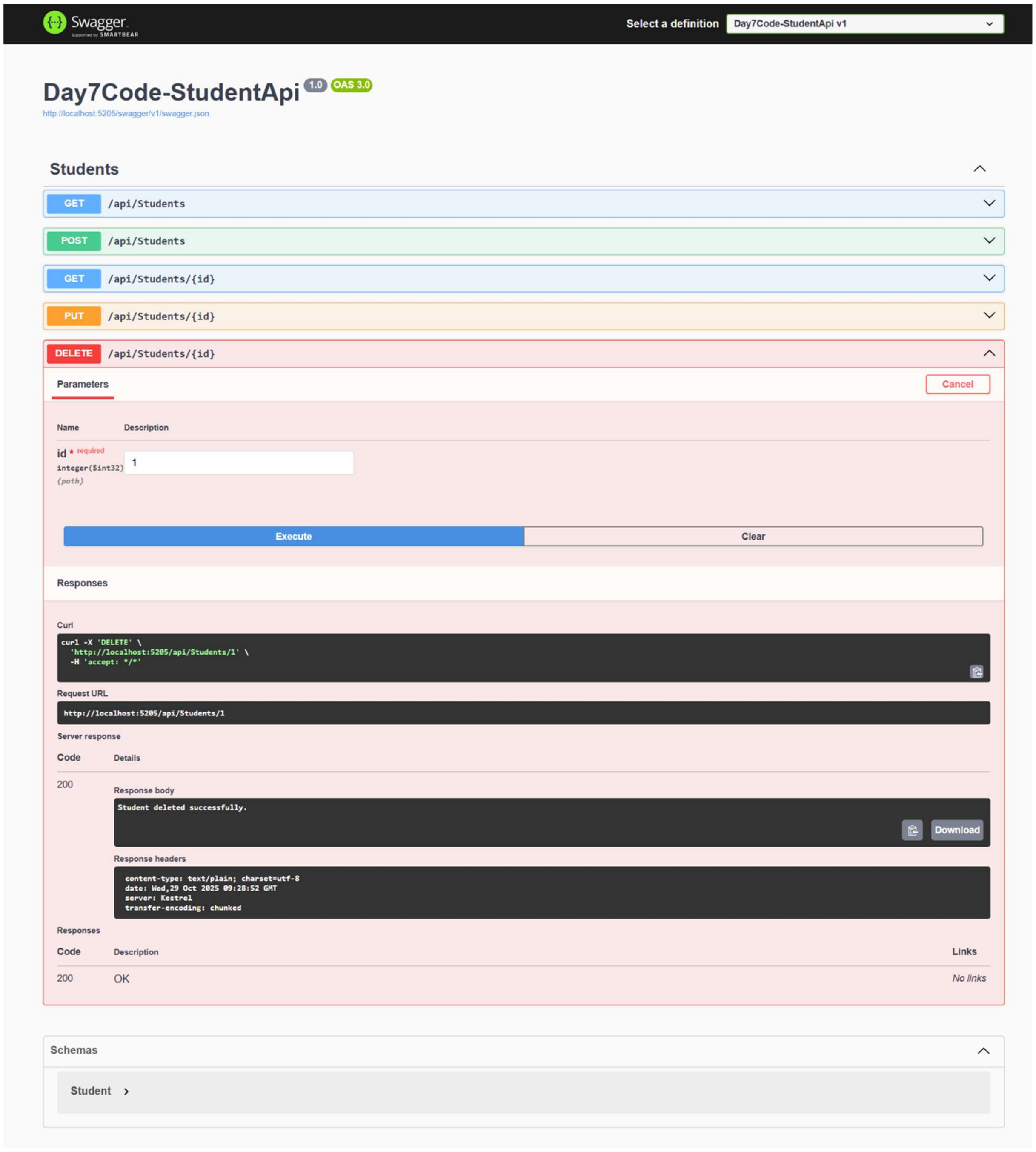
Code	Description	Links
200	OK	*No links*

DELETE /api/Students/{id}

Schemas

Student

PUT /api/students/{id} updates record details



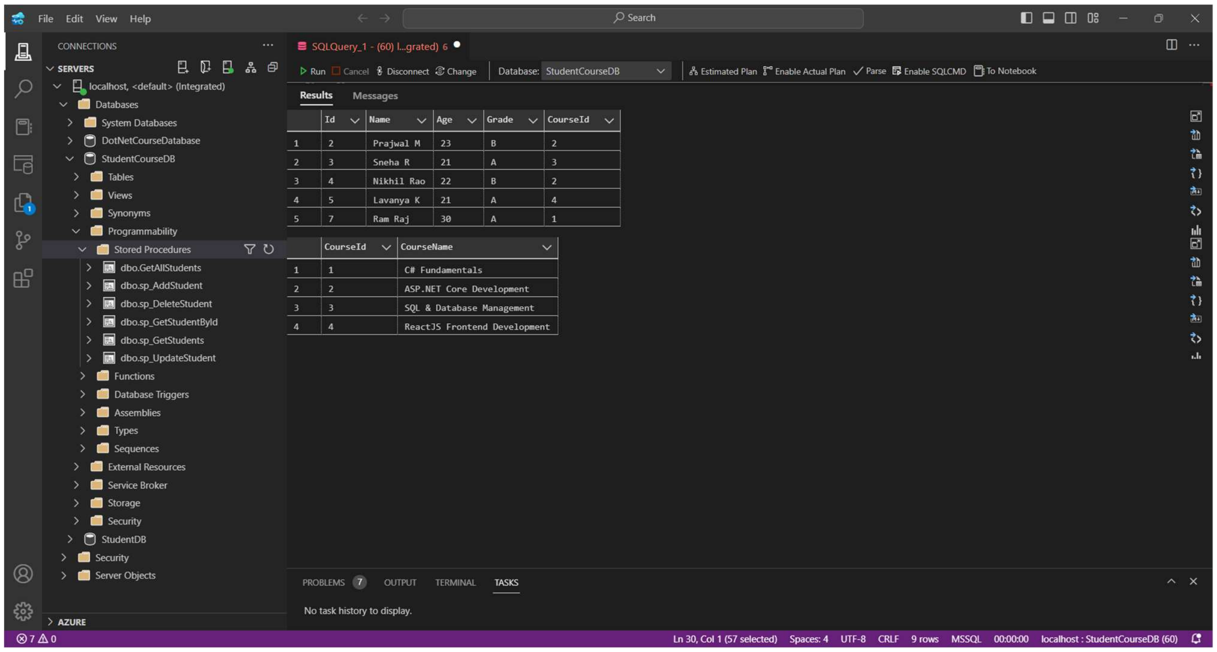
```
curl -X 'DELETE' \
  'http://localhost:5205/api/Students/1' \
  -H 'accept: */*'
```

```
http://localhost:5205/api/Students/1
```

```
Student deleted successfully.
```

```
 content-type: text/plain; charset=utf-8
 date: Wed,29 Oct 2025 09:28:52 GMT
 server: Kestrel
 transfer-encoding: chunked
```

DELETE /api/students/{id} removes record from database



	Id	Name	Age	Grade	CourseId
1	2	Prajwal M	23	B	2
2	3	Sneha R	21	A	3
3	4	Nikhil Rao	22	B	2
4	5	Lavanya K	21	A	4
5	7	Ram Raj	30	A	1

	CourseId	CourseName
1	1	C# Fundamentals
2	2	ASP.NET Core Development
3	3	SQL & Database Management
4	4	ReactJS Frontend Development

No task history to display.

SQL Server view shows updated student and course tables after CRUD operations