# Day 1 – Introduction to ASP.NET Core Web API **Date:** 16-10-2025

**Objective:** To understand the structure of an ASP.NET Core Web API project, the role of each component, and how APIs communicate with clients (like React frontends or Postma

## 1. What is ASP.NET Core Web API?

- **ASP.NET Core** is Microsoft's modern, cross-platform framework for building web applications and RESTful APIs.
- A **Web API** (Application Programming Interface) exposes data and operations through HTTP endpoints (GET, POST, PUT, DELETE) that other apps can consume.
- APIs are stateless and mainly return data in **JSON** format.

### Why use ASP.NET Core Web API?

- Cross-platform (Windows, Linux, macOS)
- Built-in Dependency Injection
- High performance with Kestrel server
- Easy integration with Entity Framework Core and SQL Server
- Excellent for React / Angular / Mobile backends

## 2. Project Structure Overview

When you create a new API project in Visual Studio or VS Code (dotnet new webapi), you get these folders:

| **Folder / File** | **Purpose** |
|---|---|
| **Controllers/** | Contains controller classes that define API endpoints |
| **Models/** | Classes representing data structures or database entities |
| **Program.cs** | Entry point — configures middleware, services, routing |
| **appsettings.json** | Stores configuration (connection strings, logging) |
| **Properties/** | Contains launchSettings.json (port, profiles) |
| **wwwroot/** | Static files (optional for APIs) |

## 3. Understanding the MVC Pattern in APIs

Although APIs don't use Views, they still follow **Model–Controller** logic:

- **Model** – Represents the data (e.g., Student, Course)
- **Controller** – Handles requests and responses
- **View** – Not used here; the client (React App or Postman) acts as the "view"

## 4. How ASP.NET Core Handles a Request

1. Client sends HTTP request (GET /api/students).
2. ASP.NET Core’s **Routing Middleware** matches the URL to a controller/action.
3. The **Controller** executes logic, interacts with a database (via EF Core).
4. The **Action Method** returns a JSON response.

## 5. Core Concepts and Terms

| **Term** | **Meaning** |
|---|---|
| **Controller** | Class ending with Controller, e.g., StudentsController, defines endpoints |
| **Action Method** | Method inside a controller that handles an HTTP verb |
| **Route** | URL pattern (api/[controller]) that maps to actions |
| **[HttpGet]/[HttpPost]/[HttpPut]/[HttpDelete]** | Attributes that define which HTTP method triggers the action |
| **IActionResult / ActionResult** | Return types representing HTTP responses |
| **Dependency Injection (DI)** | Technique to provide services (like DbContext) automatically |
| **Middleware** | Components that process requests (Routing, CORS, Authentication, etc.) |

## 6. Creating Your First Web API

### Step 1 – Create the Project

```
dotnet new webapi -n StudentApi

cd StudentApi

dotnet run
```

Default endpoint: https://localhost:5001/weatherforecast

### Step 2 – Create a Model

```
namespace StudentApi.Models

{
    public class Student
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public int Age { get; set; }
        public string Grade { get; set; }
    }
}
```

**Step 3 – Add a Controller**

```
using Microsoft.AspNetCore.Mvc;
using StudentApi.Models;
using System.Collections.Generic;

namespace StudentApi.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class StudentsController : ControllerBase
    {
        private static List<Student> students = new List<Student>();

        [HttpGet]
        public IActionResult GetAllStudents()
        {
            return Ok(students);
        }

        [HttpPost]
        public IActionResult AddStudent(Student s)
        {
            students.Add(s);
            return Ok(new { message = "Student added successfully" });
        }
    }
}
```

**Step 4 – Run and Test**

```
dotnet run
```

Open **Postman** → GET https://localhost:5001/api/students → returns empty list.
Then send a POST request with JSON:

```
{
  "id": 1,
  "name": "Udaya",
  "age": 22,
  "grade": "A"
}
```

**7. Understanding appsettings.json**

Example:

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=.;Database=StudentDB;Trusted_Connection=True;"
  },
  "Logging": { "LogLevel": { "Default": "Information" } },
  "AllowedHosts": "*"
}
```

Later you'll use this connection string in Entity Framework Core to link SQL Server.

## 8. Middleware & Request Pipeline

Middleware runs sequentially for every request.
Common ones:

1. **UseRouting()** – matches URL to routes
2. **UseCors()** – allows frontend to access backend
3. **UseAuthorization()** – manages access control
4. **MapControllers()** – executes controller actions

The order matters — it defines how requests flow through the application.

### Testing Tools

- **Postman** – API testing (send GET, POST, PUT, DELETE requests)
- **Swagger (built-in)** – Visual API documentation UI
  Launch: https://localhost:5001/swagger/index.html

### Key Takeaways

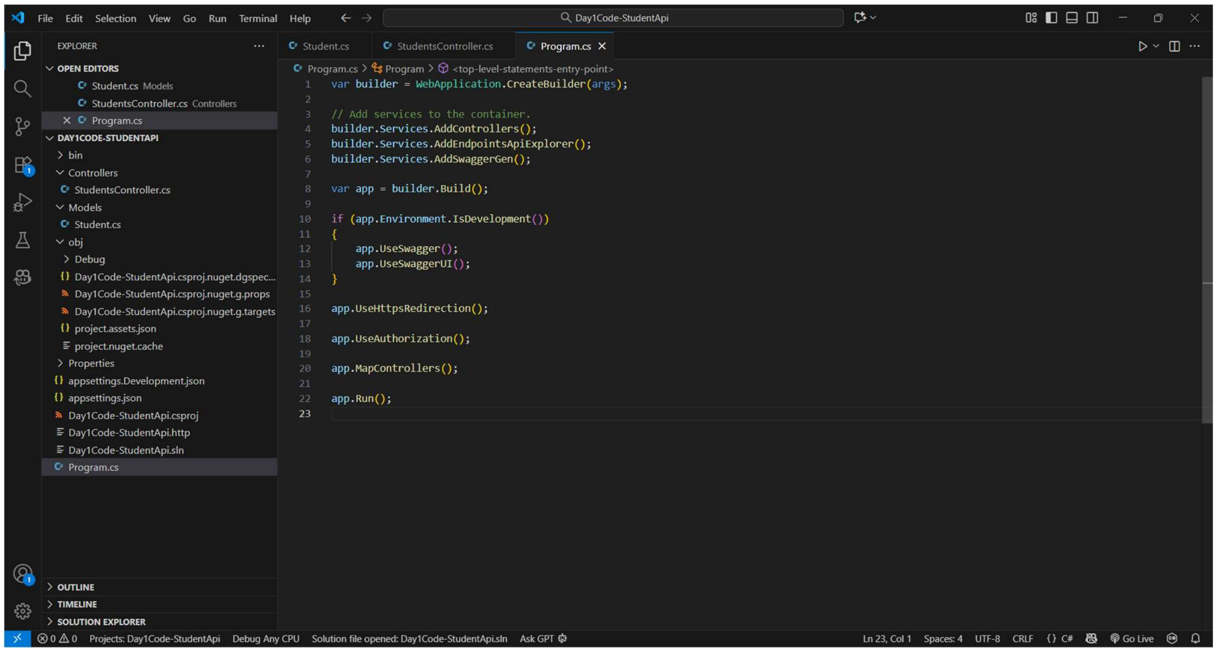
| **Concept** | **Summary** |
|---|---|
| ASP.NET Core | Framework to build REST APIs |
| Controller | Defines endpoints and logic |
| Model | Represents data structure |
| Routing | Maps URLs to controllers |
| Dependency Injection | Automatically provides services |
| Swagger & Postman | Tools for testing APIs |

### Mini Task for Day 1

Build a simple Web API with a StudentController supporting:

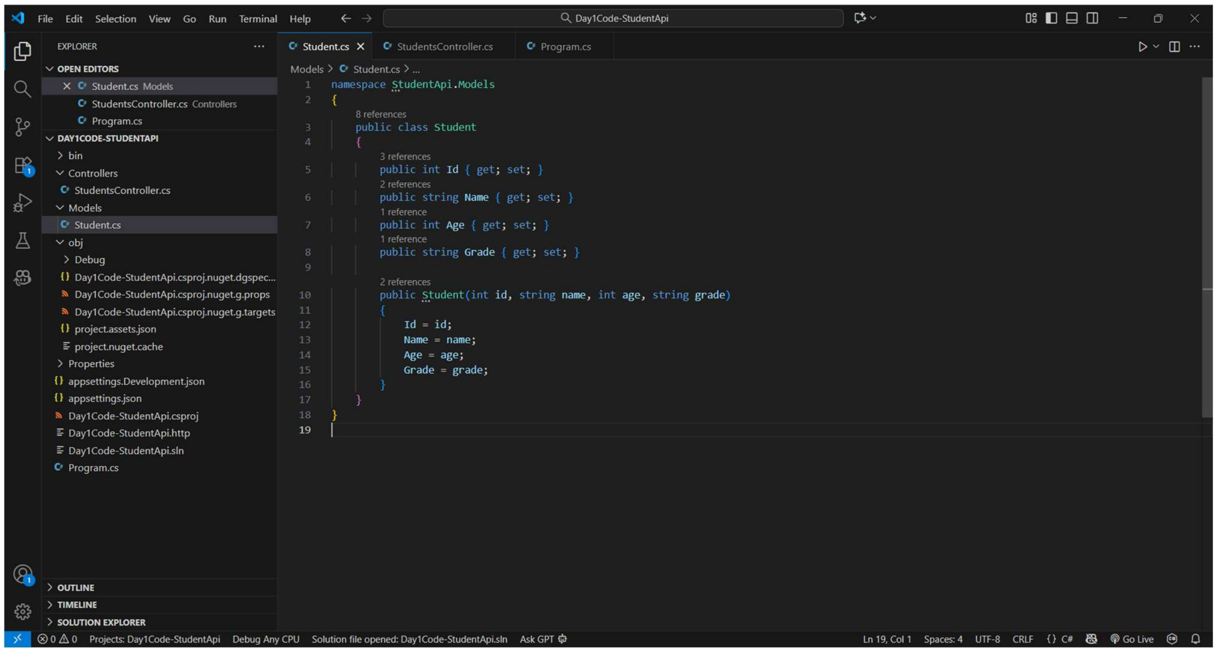
- GET – Fetch all students
- POST – Add a new student
  Return JSON responses and verify using Swagger/Postman.

## Snapshots:


File Edit Selection View Go Run Terminal Help
Day1Code-StudentApi
EXPLORER
OPEN EDITORS
Student.cs Models
StudentsController.cs Controllers
Program.cs
DAY1CODE-STUDENTAPI
bin
Controllers
StudentsController.cs
Models
Student.cs
obj
Debug
Day1Code-StudentApi.csproj.nuget.dgspec...
Day1Code-StudentApi.csproj.nuget.g.props
Day1Code-StudentApi.csproj.nuget.g.targets
project.assets.json
project.nuget.cache
Properties
appsettings.Development.json
appsettings.json
Day1Code-StudentApi.csproj
Day1Code-StudentApi.http
Day1Code-StudentApi.sln
Program.cs
OUTLINE
TIMELINE
SOLUTION EXPLORER
Program.cs > Program > <top-level-statements-entry-point>
Projects: Day1Code-StudentApi Debug Any CPU Solution file opened: Day1Code-StudentApi.sln Ask GPT
Ln 23, Col 1 Spaces: 4 UTF-8 CRLF C# Go Live

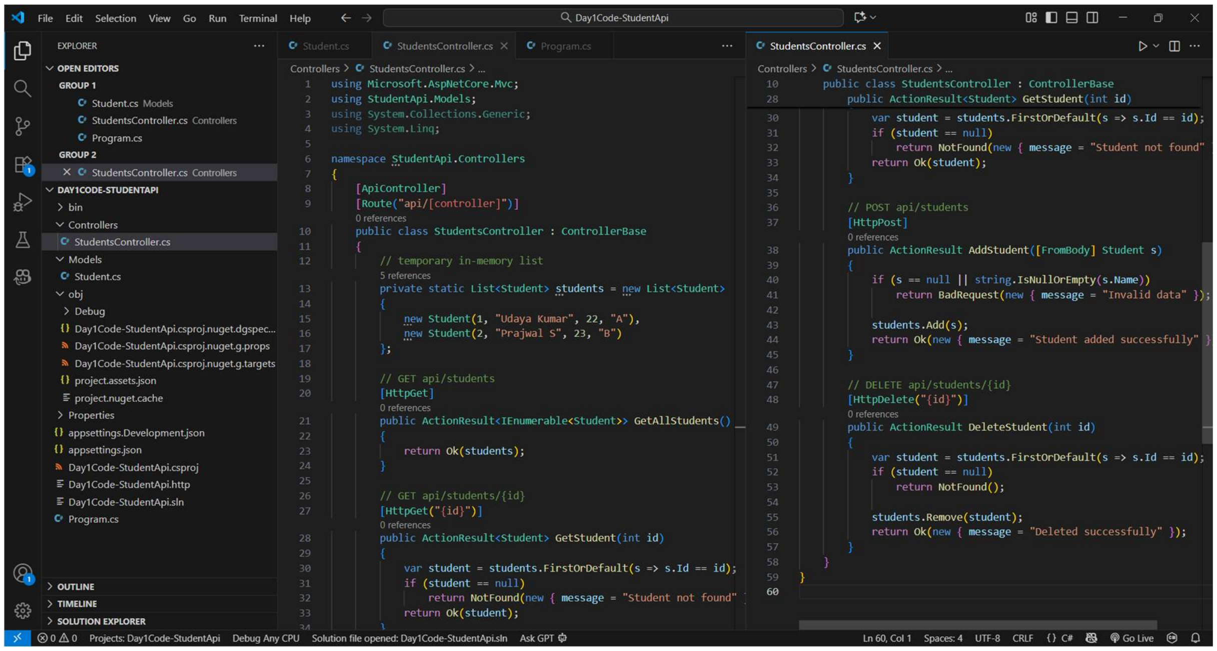
```csharp
var builder = WebApplication.CreateBuilder(args);

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

var app = builder.Build();

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

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();
```

Code : Program.cs


File Edit Selection View Go Run Terminal Help
Day1Code-StudentApi
EXPLORER
OPEN EDITORS
Student.cs Models
StudentsController.cs Controllers
Program.cs
DAY1CODE-STUDENTAPI
Models > Student.cs > ...
8 references
3 references
2 references
1 reference
1 reference
2 references
Projects: Day1Code-StudentApi Debug Any CPU Solution file opened: Day1Code-StudentApi.sln Ask GPT
Ln 19, Col 1 Spaces: 4 UTF-8 CRLF C# Go Live

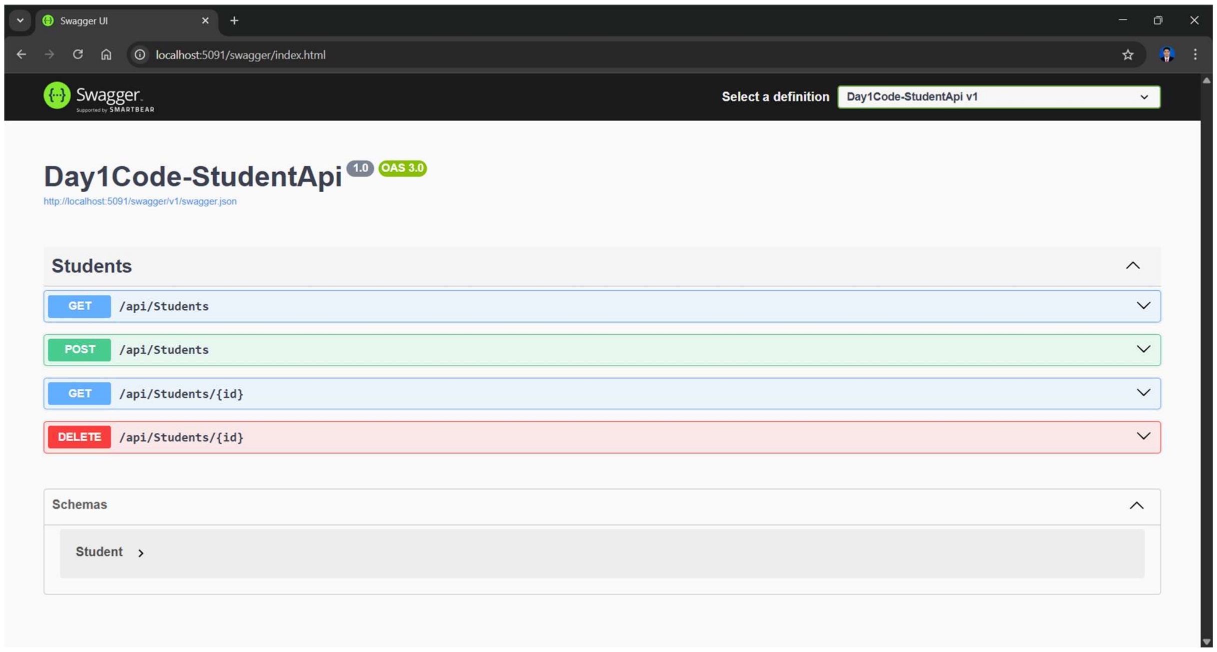
```csharp
namespace StudentApi.Models
{
    public class Student
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public int Age { get; set; }
        public string Grade { get; set; }

        public Student(int id, string name, int age, string grade)
        {
            Id = id;
            Name = name;
            Age = age;
            Grade = grade;
        }
    }
}
```

Code : Student.cs



```csharp
using Microsoft.AspNetCore.Mvc;
using StudentApi.Models;
using System.Collections.Generic;
using System.Linq;

namespace StudentApi.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class StudentsController : ControllerBase
    {
        // temporary in-memory list
        private static List<Student> students = new List<Student>
        {
            new Student(1, "Udaya Kumar", 22, "A"),
            new Student(2, "Prajwal S", 23, "B")
        };

        // GET api/students
        [HttpGet]
        public ActionResult<IEnumerable<Student>> GetAllStudents()
        {
            return Ok(students);
        }

        // GET api/students/{id}
        [HttpGet("{id}")]
        public ActionResult<Student> GetStudent(int id)
        {
            var student = students.FirstOrDefault(s => s.Id == id);
            if (student == null)
                return NotFound(new { message = "Student not found"
            return Ok(student);
        }

        // POST api/students
        [HttpPost]
        public ActionResult AddStudent([FromBody] Student s)
        {
            if (s == null || string.IsNullOrEmpty(s.Name))
                return BadRequest(new { message = "Invalid data" });

            students.Add(s);
            return Ok(new { message = "Student added successfully" }
        }

        // DELETE api/students/{id}
        [HttpDelete("{id}")]
        public ActionResult DeleteStudent(int id)
        {
            var student = students.FirstOrDefault(s => s.Id == id);
            if (student == null)
                return NotFound();

            students.Remove(student);
            return Ok(new { message = "Deleted successfully" });
        }
    }
}
```

Code : StudentController.cs



localhost:5091/swagger/index.html

Select a definition: Day1Code-StudentApi v1

Day1Code-StudentApi 1.0 OAS 3.0

http://localhost:5091/swagger/v1/swagger.json

Students

GET /api/Students

POST /api/Students

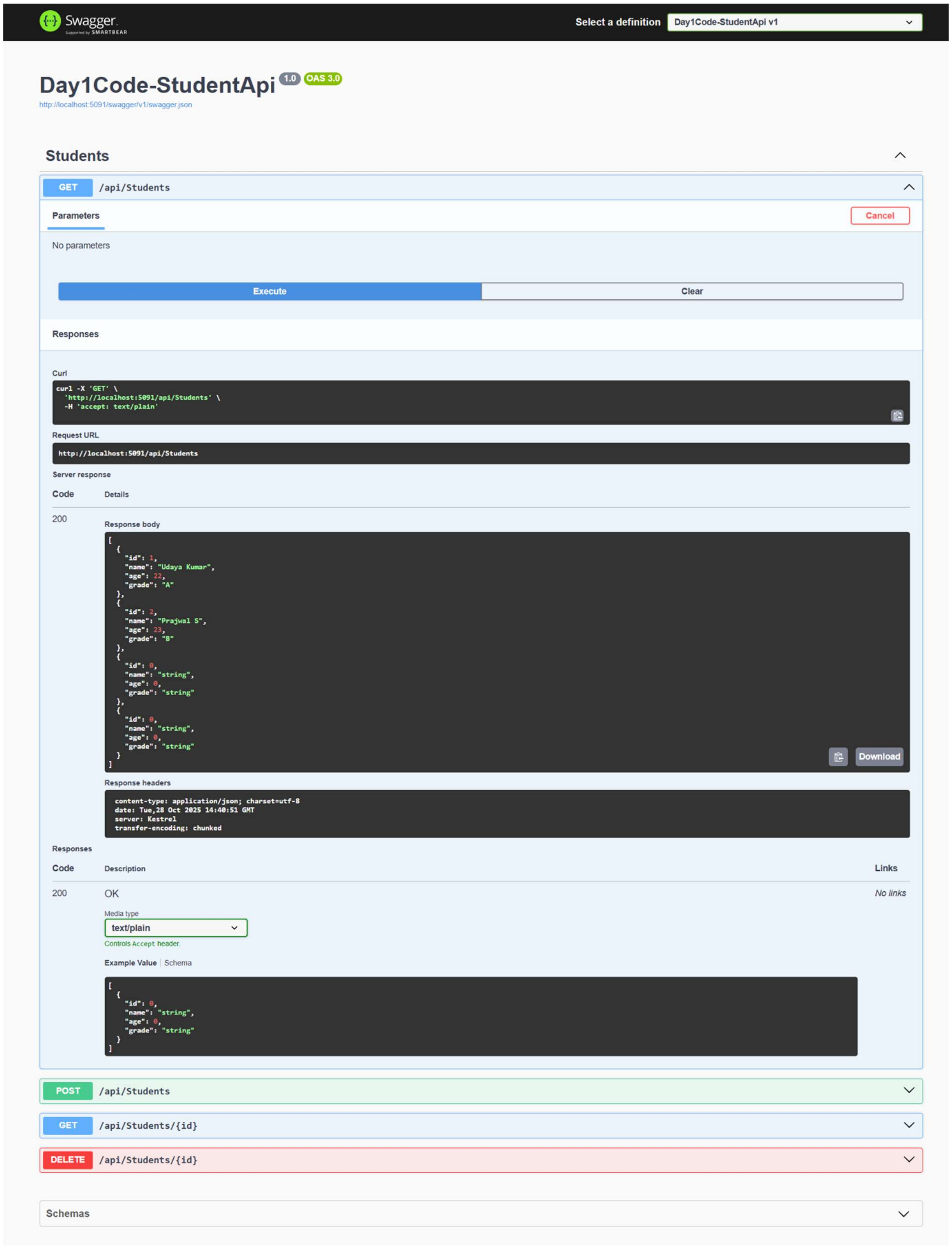
GET /api/Students/{id}

DELETE /api/Students/{id}

Schemas

Student

Output : Swagger UI



Day1Code-StudentApi 1.0 OAS 3.0

http://localhost:5091/swagger/v1/swagger.json

Students

GET /api/Students

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5091/api/Students' \
  -H 'accept: text/plain'
```

Request URL

```
http://localhost:5091/api/Students
```

Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "name": "Udaya Kumar",
    "age": 22,
    "grade": "A"
  },
  {
    "id": 2,
    "name": "Prajwal S",
    "age": 23,
    "grade": "B"
  },
  {
    "id": 0,
    "name": "string",
    "age": 0,
    "grade": "string"
  },
  {
    "id": 0,
    "name": "string",
    "age": 0,
    "grade": "string"
  }
]
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,28 Oct 2025 14:40:51 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

| Code | Description | Links |
|---|---|---|
| 200 | OK | No links |

Media type: text/plain

Controls Accept header.

Example Value | Schema

```
[
  {
    "id": 0,
    "name": "string",
    "age": 0,
    "grade": "string"
  }
]
```

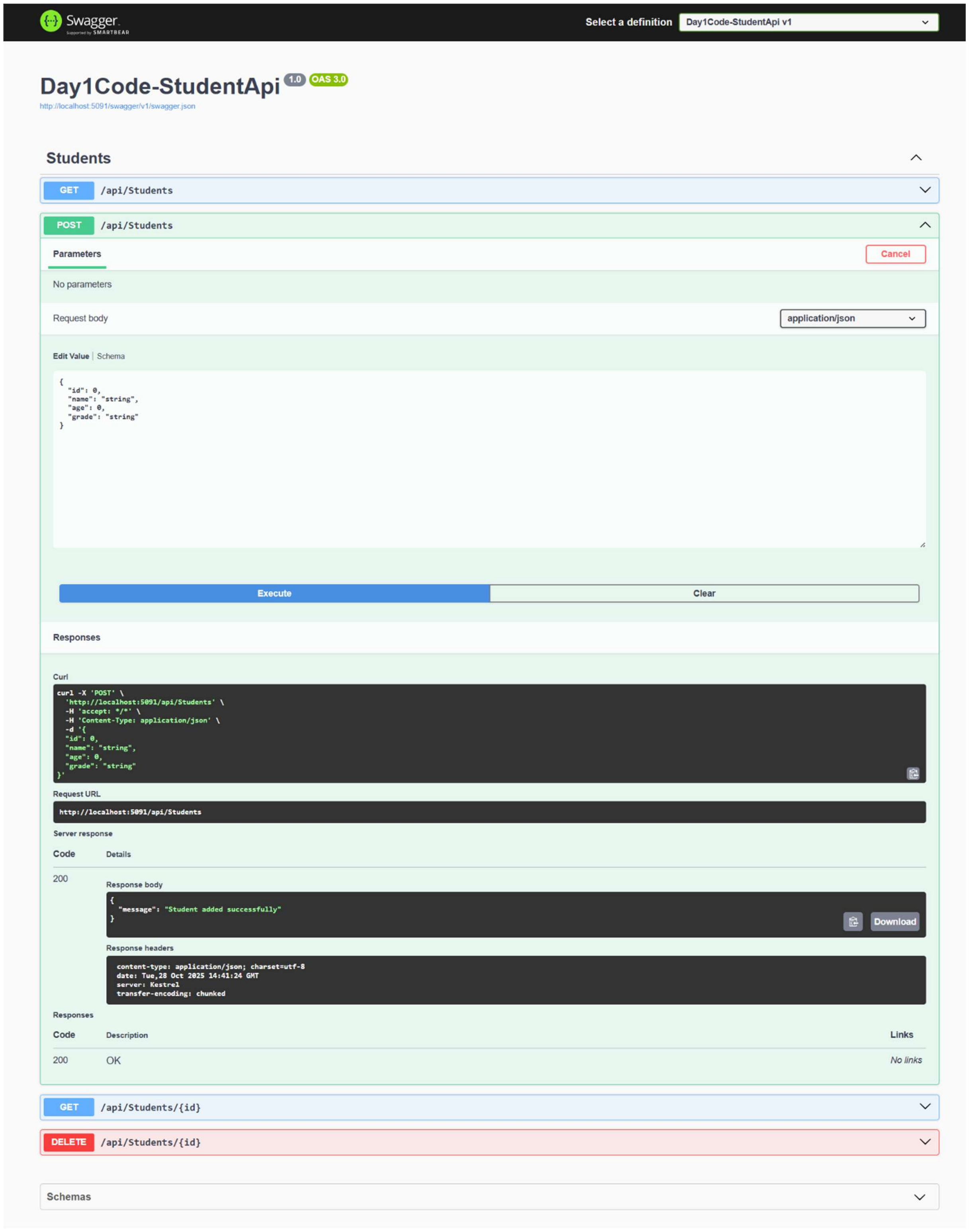
POST /api/Students

GET /api/Students/{id}

DELETE /api/Students/{id}

Schemas

Output : GET ( All data )



Day1Code-StudentApi 1.0 OAS 3.0

http://localhost:5091/swagger/v1/swagger.json

Students

GET /api/Students

POST /api/Students

Parameters

No parameters

Request body: application/json

Edit Value | Schema

```
{
  "id": 0,
  "name": "string",
  "age": 0,
  "grade": "string"
}
```

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:5091/api/Students' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 0,
  "name": "string",
  "age": 0,
  "grade": "string"
}'
```

Request URL

```
http://localhost:5091/api/Students
```

Server response

| Code | Details |
|---|---|
| 200 | Response body |

```
{
  "message": "Student added successfully"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,28 Oct 2025 14:41:24 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

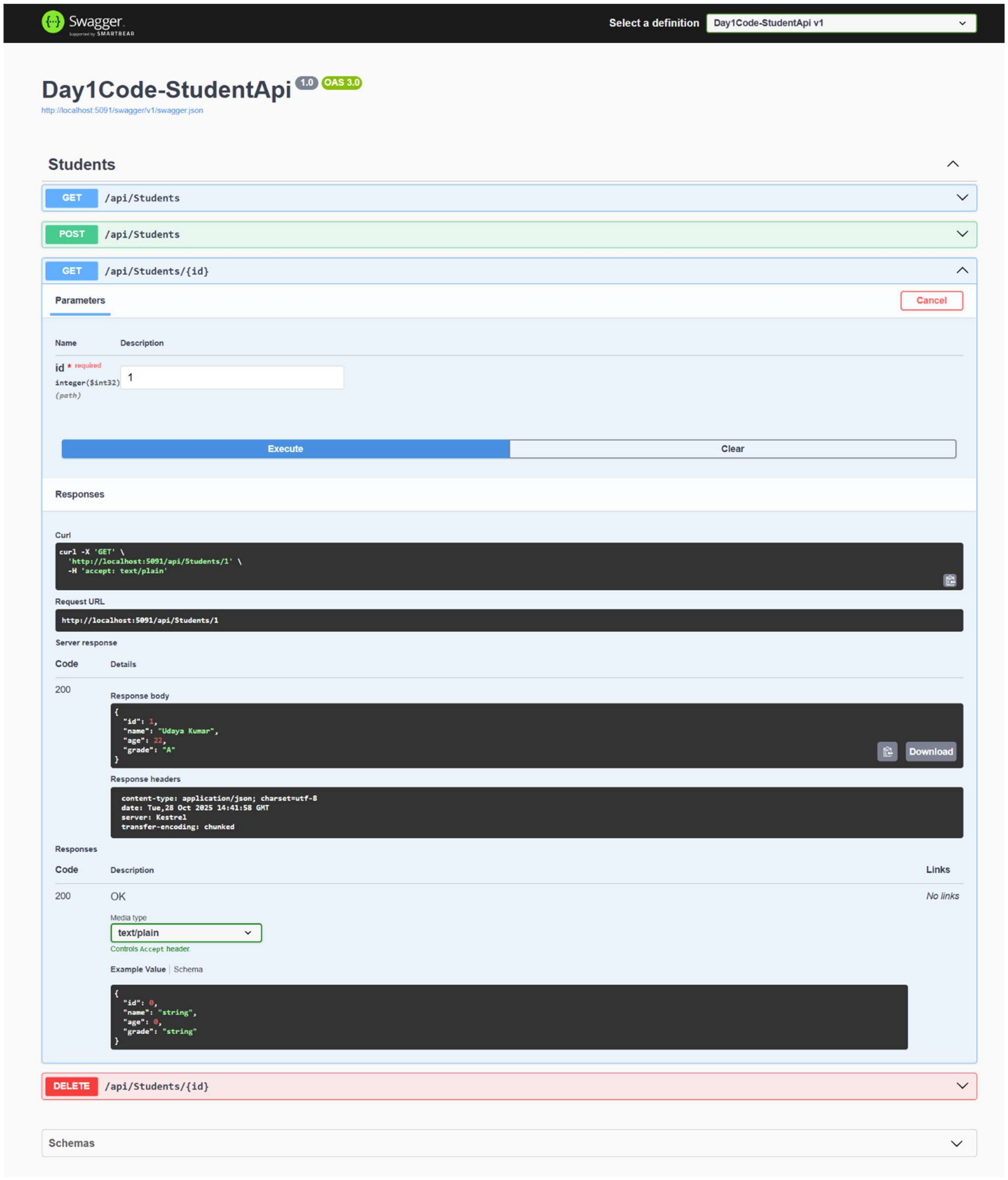
| Code | Description | Links |
|---|---|---|
| 200 | OK | No links |

GET /api/Students/{id}

DELETE /api/Students/{id}

Schemas

Output : POST



Day1Code-StudentApi 1.0 OAS 3.0

http://localhost:5091/swagger/v1/swagger.json

Students

GET /api/Students

POST /api/Students

GET /api/Students/{id}

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

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5091/api/Students/1' \
  -H 'accept: text/plain'
```

Request URL

```
http://localhost:5091/api/Students/1
```

Server response

Code | Details

200

Response body

```
{
  "id": 1,
  "name": "Udaya Kumar",
  "age": 22,
  "grade": "A"
}
```

Response headers

```
 content-type: application/json; charset=utf-8
 date: Tue,28 Oct 2025 14:41:58 GMT
 server: Kestrel
 transfer-encoding: chunked
```

Responses

| Code | Description | Links |
|---|---|---|
| 200 | OK | No links |

Media type: text/plain

Controls Accept header.

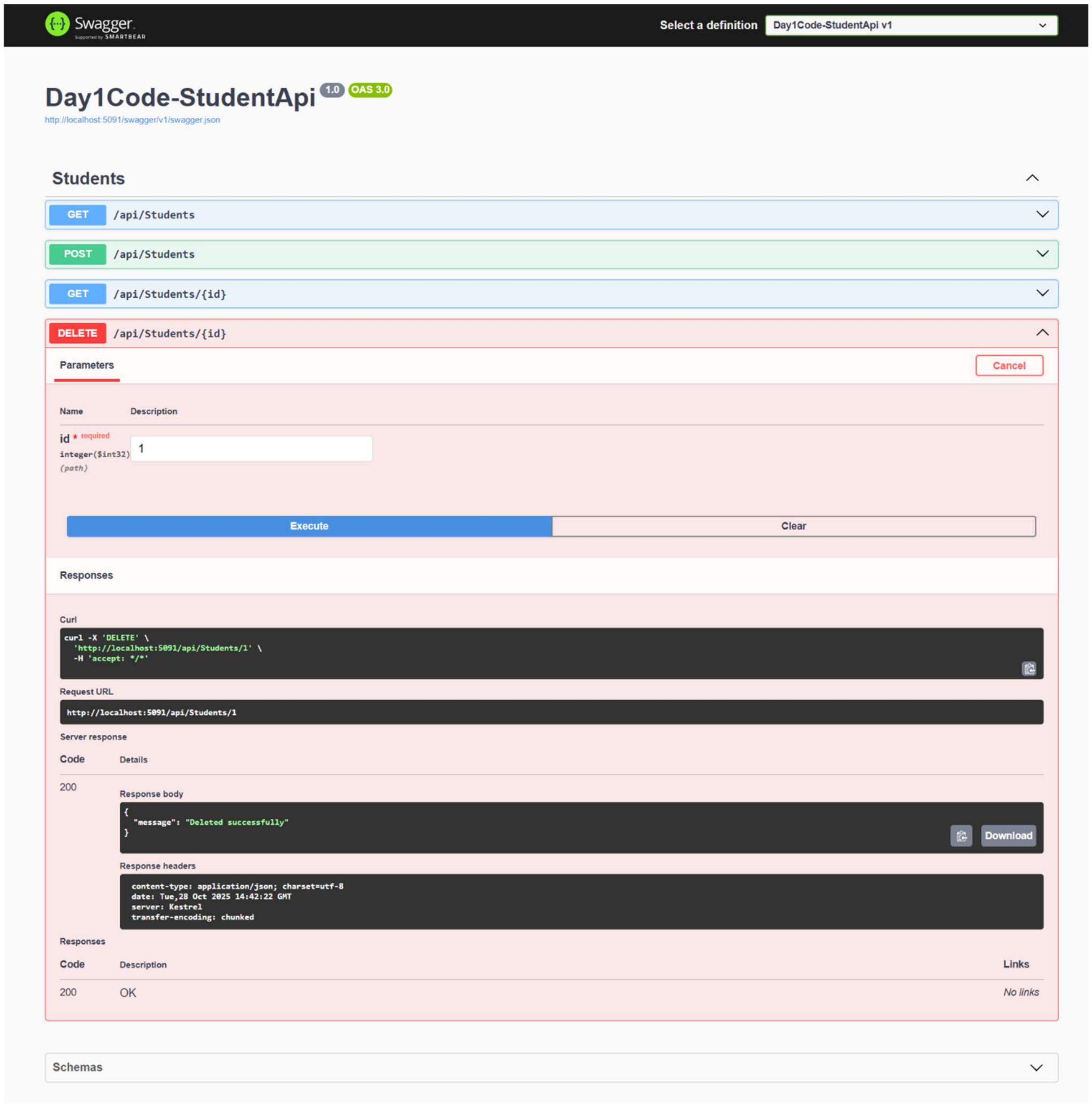
Example Value | Schema

```
{
  "id": 0,
  "name": "string",
  "age": 0,
  "grade": "string"
}
```

DELETE /api/Students/{id}

Schemas

Output : GET (Specific Data)



```
curl -X 'DELETE' \
  'http://localhost:5091/api/Students/1' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5091/api/Students/1
```

Server response

| Code | Details |
|---|---|
| 200 | Response body |

```
{
  "message": "Deleted successfully"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,28 Oct 2025 14:42:22 GMT
server: Kestrel
transfer-encoding: chunked
```

Responses

| Code | Description | Links |
|---|---|---|
| 200 | OK | No links |

Output : DELETE