Thenical doumentation: School Management

I. INTRODUCTION

Welcome to the School Management System API project! This project aims to provide an efficient and flexible API for managing various aspects of a school, including schools, classes, students, subjects, and course notes. The School Management System API is designed to facilitate the organization and administration of educational institutions. It allows users to create, read, update, and delete data related to schools, classes, students, subjects, and course notes.

II. TECHNOLOGY USED

SQLite :

SQLite is a very popular and widely used relational database management system (RDBMS). Unlike other RDBMSs, SQLite is embedded in the application itself, which means it operates as a library rather than a separate database server. This allows easy integration into various applications and systems, without requiring complex configuration or administration.

SQLite is designed to be lightweight, fast, and easy to use. It stores data in a single file, which makes it convenient for embedded applications or applications that don't need a heavier database management system. SQLite supports standard relational database features, such as ACID (Atomicity, Consistency, Isolation, Durability) transactions, advanced SQL queries, indexes, and views.

Thanks to its serverless nature, SQLite is often used in scenarios such as mobile applications, web browsers, embedded systems, light desktop tools, and ***small development projects*** where an autonomous database is needed.

ASP.NET :

ASP.NET is a web development framework created by Microsoft. It allows you to build dynamic and powerful web applications using the C# or Visual Basic .NET programming language. ASP.NET is based on the Common Language Runtime (CLR), which is the .NET runtime environment.

ASP.NET offers a wide range of features for web development, including support for design patterns such as the Model-View-Controller (MVC) pattern and the WebForms pattern. It also features event-based architecture, built-in server controls, security management, session management, error handling, and more.

Thanks to its tight integration with the CLR, ASP.NET enjoys high performance and high scalability. It is capable of handling a large number of concurrent requests and supports deployment on Windows servers with Internet Information Services (IIS) or Linux servers with .NET Core.

ASP.NET is also compatible with a wide range of technologies and services, such as SOAP and REST web services, JSON APIs, messaging services, database services (like SQLite), cloud services and more. more besides.

In summary, SQLite is a lightweight, self-contained relational database management system ideal for embedded applications or ***small projects***. ASP.NET, on the other hand, is a comprehensive web development framework for building powerful web applications with great flexibility and advanced features.

III. SHEMA

ClassR{

```
    name            string
                    nullable: true
    id              integer($int32)
    schoolId        integer($int32)
}
```

CourseNote{

```
    id              integer($int32)
    subjectId       integer($int32)
    studentId       integer($int32)
    mark            number($double)
}
```

School{

```
    name            string
                    nullable: true
    city            string
                    nullable: true
    zipCode         string
                    nullable: true
    country         string
                    nullable: true
    id              integer($int32)
    adress          string
                    nullable: true
}
```

Student{

```
    id              integer($int32)
    firstName       string
                    nullable: true
    lastName        string
                    nullable: true
    adress          string
                    nullable: true
    city            string
                    nullable: true
    zipCode         string
                    nullable: true
    country         string
                    nullable: true
}
```

Subject {

id	integer($int32)	
classId	integer($int32)	
maxPoint	integer($int32)	
name	string	
	nullable: true	

}

IV. ENDPOINTS

Note: ***for some PUT requests below you have to give a key and value dictionary to update the information (I made this choice to keep the correct schema)***

Example: upgrade grad of student base on subjectId, studentId and mark

```
{
  "studentID": 1,
  "mark": 2.3,
  "subjectId": 1
}
```

- School

`http://localhost:5261/api/SchoolControllers`

This endPoint is use to **get** liste of all school of database and return a list of School object.

`http://localhost:5261/api/SchoolControllers`

You can use also this endpoint for **Post** data to database

`http://localhost:5261/api/SchoolControllers/id?id=1`

This endpoint is use to **get** details of school filter by id

You can use also this endpoint for update school information with **PUT** request and **DELETE** request for delete particular school

- Class

`http://localhost:5261/api/ClassControllers`

This endpoint is use to **get** all class in database, you can also use it with **POST** request to register data in database.

`http://localhost:5261/api/ClassControllers/schoolId?schoolId=1`

This endpoint is use to **get** details of class filter by id

You can use also this endpoint for update class information with **PUT** request and **DELETE** request for delete particular class.

- CourseNote

```
http://localhost:5261/api/CourseNoteControllers
```

This endpoint is use to **get** all couseNote in database, you can also use it with **POST** request to register data in database or **DELETE** grade of particular student basing on subject id and student id.

```
http://localhost:5261/api/CourseNoteControllers/dag
```

DELETE all grade of student basing on subject id and student id.

- Student

```
http://localhost:5261/api/StudentControllers
```

This endpoint is use to **get** all student in database, you can also use it with **POST** request to register data in database.

```
http://localhost:5261/api/StudentControllers
```

You can use this endpoint with **PUT** request to update class of student based on studentId and classId that you want to assigned it

studentI

```
http://localhost:5261/api/StudentControllers/stusentId
```

You can use this endpoint for update student information (details) with **PUT** request

```
http://localhost:5261/api/StudentControllers/upd
```

You can use this endpoint to upgrade grade of student base on subjectId, studentId, and mark.

```
http://localhost:5261/api/StudentControllers/classId
```

You can use this endpoint to get details of student based on their class (classId)

```
http://localhost:5261/api/StudentControllers/subjectId
```

You can use this endpoint to get mark of student based on subjectId and studentId

- Subject

```
http://localhost:5261/api/SubjectControllers
```

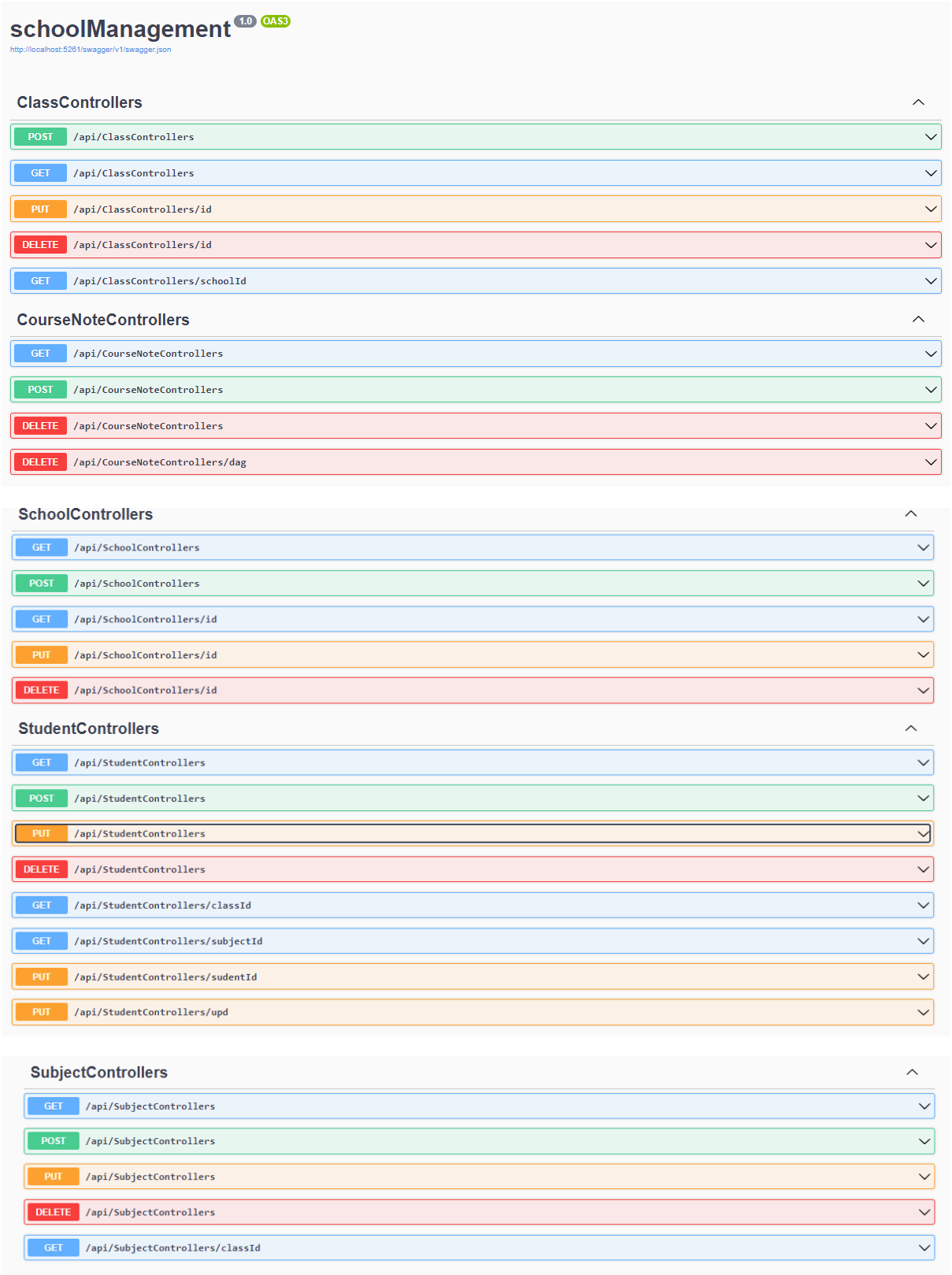
This endpoint is use to **get** all subject in database, you can also use it with **POST** request to register data in database or with **PUT** to make update.

```
http://localhost:5261/api/SubjectControllers?subjectId=2
```

you can use this endpoint to DELETE particular subject in database.

```
http://localhost:5261/api/SubjectControllers/classId?classId=1
```

You can use this endpoint to **GET** subject based on classId.


schoolManagement 1.0 OAS3
http://localhost:5261/swagger/v1/swagger.json
ClassControllers
POST /api/ClassControllers
GET /api/ClassControllers
PUT /api/ClassControllers/id
DELETE /api/ClassControllers/id
GET /api/ClassControllers/schoolId
CourseNoteControllers
GET /api/CourseNoteControllers
POST /api/CourseNoteControllers
DELETE /api/CourseNoteControllers
DELETE /api/CourseNoteControllers/dag
SchoolControllers
GET /api/SchoolControllers
POST /api/SchoolControllers
GET /api/SchoolControllers/id
PUT /api/SchoolControllers/id
DELETE /api/SchoolControllers/id
StudentControllers
GET /api/StudentControllers
POST /api/StudentControllers
PUT /api/StudentControllers
DELETE /api/StudentControllers
GET /api/StudentControllers/classId
GET /api/StudentControllers/subjectId
PUT /api/StudentControllers/sudentId
PUT /api/StudentControllers/upd
SubjectControllers
GET /api/SubjectControllers
POST /api/SubjectControllers
PUT /api/SubjectControllers
DELETE /api/SubjectControllers
GET /api/SubjectControllers/classId