API DEVELOPMENT PATH

API DESIGN DOCUMENT TEMPLATE

v1.0

An example RESTful API design document that can be referenced during designing an API.

A downloadable resource of the Designing RESTful APIs course.

WWW.CODEWITHPRAVEEN.COM

Preface

Hi there!

I hope you are doing good.

I've created this design document template to help you with your everyday programming in API. You can **use this as a reference document** while designing an API for the requirements in hand. It shows you how a typical API design document looks in the simplest form.

Designing an API is the first step you need to do when working with APIs. **This downloadable resource is part of the Designing RESTful APIs course**, which covers the essentials of designing concepts that any API programmer **must** know. Click here to know more about the companion course.

See you in the course video!
Praveen.

Table of Contents

API for College Management System (CMS)

API Overview

Title: OpenAPI Specification for CMS
Description: API Specification document of the CMS system
Contact: Praveenkumar Bouna (http://myorganization.com/staff/praveenkumar-bouna)
Version: 1.0
API Type: Public
Server base URL: http://localhost:44333/api

Courses

GET /api/v1/courses

Content-Type: application/json
Request:
    Query parameters (optional):
     courseType:
        - Support for Filtering.
        - Filter the results by course type.
        - Example: api/courses?courseType=Engineering
     page
        - Support for Pagination.
        - Page number of the result to fetch.
        - Example: api/courses?page=1&size=4
     size
        - Support for Pagination.
        - Page size of each result.
        - Example: api/courses?page=1&size=4
     sortBy
        - Support for Sorting.
        - The field name to sort the results on.
        - Example: api/courses?sortBy=courseName
Responses:
    HTTP 200 OK
    Response Body Format:

```
[
        {
                "courseId": Unique ID of a course in the system
                "courseName": Name of the course
                "courseDuration": Duration of the course in years
                "courseType": Type of the course
        }
    ]
    Example:
    [
        {
                "courseId": 1,
                "courseName": "Computer Science",
                "courseDuration": 4,
                "courseType": "Engineering"
        },
        {
                "courseId": 2,
                "courseName": "Computer Science",
                "courseDuration": 4,
                "courseType": "Engineering"
        }
    ]
```

==

POST /api/v1/courses

Content-Type: application/json
Request:

```
    Request Body Format:
    {
        "courseName": Name of the course
        "courseDuration": Duration of the course in years
        "courseType": Type of the course
    }
    Example:
    {
        "courseId": 1,
        "courseName": "Computer Science",
        "courseDuration": 4,
        "courseType": "Engineering"
    }
```

Responses:

```
    HTTP 201 CREATED
    Response Body Format:
```

```
{
    "courseId": Unique ID of a course in the system
    "courseName": Name of the course
    "courseDuration": Duration of the course in years
    "courseType": Type of the course
}
Example:
{
    "courseId": 1,
    "courseName": "Computer Science",
    "courseDuration": 4,
    "courseType": "Engineering"
}

HTTP 400 BAD REQUEST
Response Body Format:
{
    "error":
    {
        "code": Unique error code for your project
        "message": Useful message for developers to identify the issue
    }
}

Example:
{
    "error":
    {
        "code": "INVALID_INPUT",
        "message": "One or more input arguments are invalid",
    }
}
```

==

GET /api/v1/courses/{courseId}

Content-Type: application/json
Request:

Example:

GET /api/v1/courses/1

Responses:

```
	HTTP 200 OK
	Response Body Format:
	{
		"courseId": Unique ID of a course in the system
		"courseName": Name of the course
		"courseDuration": Duration of the course in years
		"courseType": Type of the course
	}
	Example:
	{
		"courseId": 1,
		"courseName": "Computer Science",
		"courseDuration": 4,
		"courseType": "Engineering"
	}

	HTTP 404 NOT FOUND
	Response Body Format:
	{
		"error":
		{
			"code": Unique error code for your project
			"message": Useful message for developers to identify the issue
		}
	}

	Example:
	{
		"error":
		{
			"code": "INVALID_INPUT",
			"message": "One or more input arguments are invalid",
		}
	}
```

==

PUT   /api/v1/courses/{courseId}

Content-Type: application/json
Request:
	Request Body Format:

```
        {
                "courseName": Name of the course
                "courseDuration": Duration of the course in years
                "courseType": Type of the course
        }
        Example:
        PUT /api/v1/courses/1
        {
                "courseName": "Computer Science",
                "courseDuration": 3,
                "courseType": "Engineering"
        }
Responses:
        HTTP 200 OK
        Response Body Format:
        {
                "courseId": Unique ID of a course in the system
                "courseName": Name of the course
                "courseDuration": Duration of the course in years
                "courseType": Type of the course
        }
        Example:
        {
                "courseId": 1,
                "courseName": "Computer Science",
                "courseDuration": 4,
                "courseType": "Engineering"
        }

        HTTP 404 NOT FOUND
        Response Body Format:
        -- refer POST /api/courses/{courseId} --
```

===

DELETE /api/v1/courses/{courseId}

```
Content-Type: application/json
Request:
        None
Responses:
        HTTP 204 NO CONTENT
```

None

HTTP 404 NOT FOUND
Response Body Format:
-- refer POST /api/courses/{courseId} --

===

GET /api/v1/courses/{courseId}/students

Content-Type: application/json
Request:
None
Responses:
HTTP 200 OK
Response Body Format:

```
[
	{
		"studentId": Unique ID of a student in the system
		"firstName": First name of the student
		"lastName": Last name of the student
		"phoneNumber": Mobile number of the student
		“address": Permanent address of the student
	}
]
```

Example:

```
[
	{
		"studentId": 101,
		"firstName": "James",
		"lastName": "Smith",
		"phoneNumber": "555-555-1234",
		"address": "US"
	},
	{
		"studentId": 102,
		"firstName": "Robert",
		"lastName": "Smith",
		"phoneNumber": "555-555-5678",
		"address": "US"
	}
]
```

```
        HTTP 400 BAD REQUEST
        -- refer POST /api/courses --
```

==

POST /api/v1/courses/{courseId}/students

```
Content-Type: application/json
Request:
        Request Body Format:
        {
                "studentId": Unique ID of a student in the system
                "firstName": First name of the student
                "lastName": Last name of the student
                "phoneNumber": Mobile number of the student
                "address": Permanent address of the student
        }
        Example:
        {
                "studentId": 101,
                "firstName": "James",
                "lastName": "Smith",
                "phoneNumber": "555-555-1234",
                "address": "US"
        }
Responses:
        HTTP 201 CREATED
        Response Body Format:
        {
           "studentId",
           "firstName",
           "lastName",
           "phoneNumber",
           "address",
        }

        Examples:
        {
           "studentId": 101
           "firstName": "James"
           "lastName": "Smith"
```

```
    "phoneNumber": "555-555-1234"
    "address": "US"
}

HTTP 400 BAD REQUEST
-- refer POST /api/courses --
```

Thank you!

I hope this resource was helpful to you.