



Tahaluf © Copyright
2022- All Right Reserved

Harmony IT Solution



Web Application Programming Interface (API)

Tahaluf Training Center 2022





1

Overview Of HTTP Verbs

2

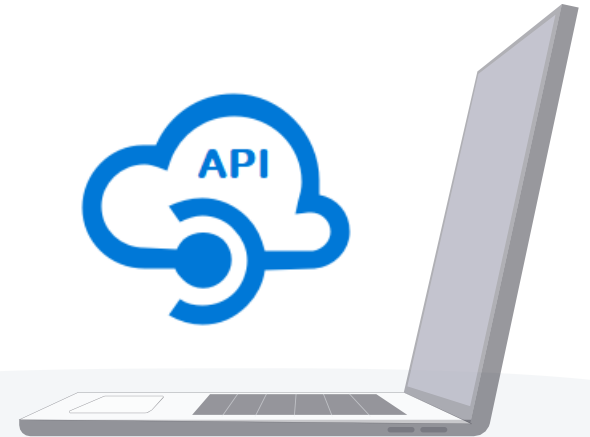
Overview Of HTTP Status Code

3

Controller

4

Upload Image





Overview Of HTTP Verbs



REST APIs enable to development of any kind of web application having all CRUD operations (Create, Retrieve, Update, Delete).

REST guidelines suggest using specific HTTP verbs on a particular type of call made to the server.





HTTP GET

Use GET requests **to retrieve resource represent information only** and not to update it in any way. As GET requests do not update the state of the resource, these are said to be **safe methods**.





HTTP POST

Use POST APIs **to create new resources**, When talking strictly in terms of REST, POST verbs are used to create a new resource into the collection of resources.





HTTP PUT

Use PUT APIs primarily **to modify existing resource** (if the resource does not exist, then API may decide to create a new resource or not).





HTTP DELETE

DELETE APIs are used **to delete resources** (identified by the Request URI).





Overview Of HTTP Status Code



HTTP STATUS CODES

5xx Server Error

- 501 Not Implemented
- 502 Bad Gateway
- 503 Service Unavailable
- 504 Gateway Timeout

4xx Client Error

- 401 Unauthorized Error
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed

3xx Redirection

- 301 Permanent Redirect
- 302 Temporary Redirect
- 304 Not Modified

2xx Success

- 200 Success / OK

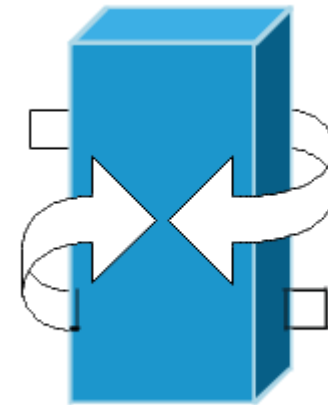
A large green rectangle with rounded corners, tilted slightly to the right. It is surrounded by a thick red outline that forms a frame around it. There are several small red circles and two small black circles scattered around the green rectangle. The word "Controller" is written in white text in the center of the green rectangle.

Controller



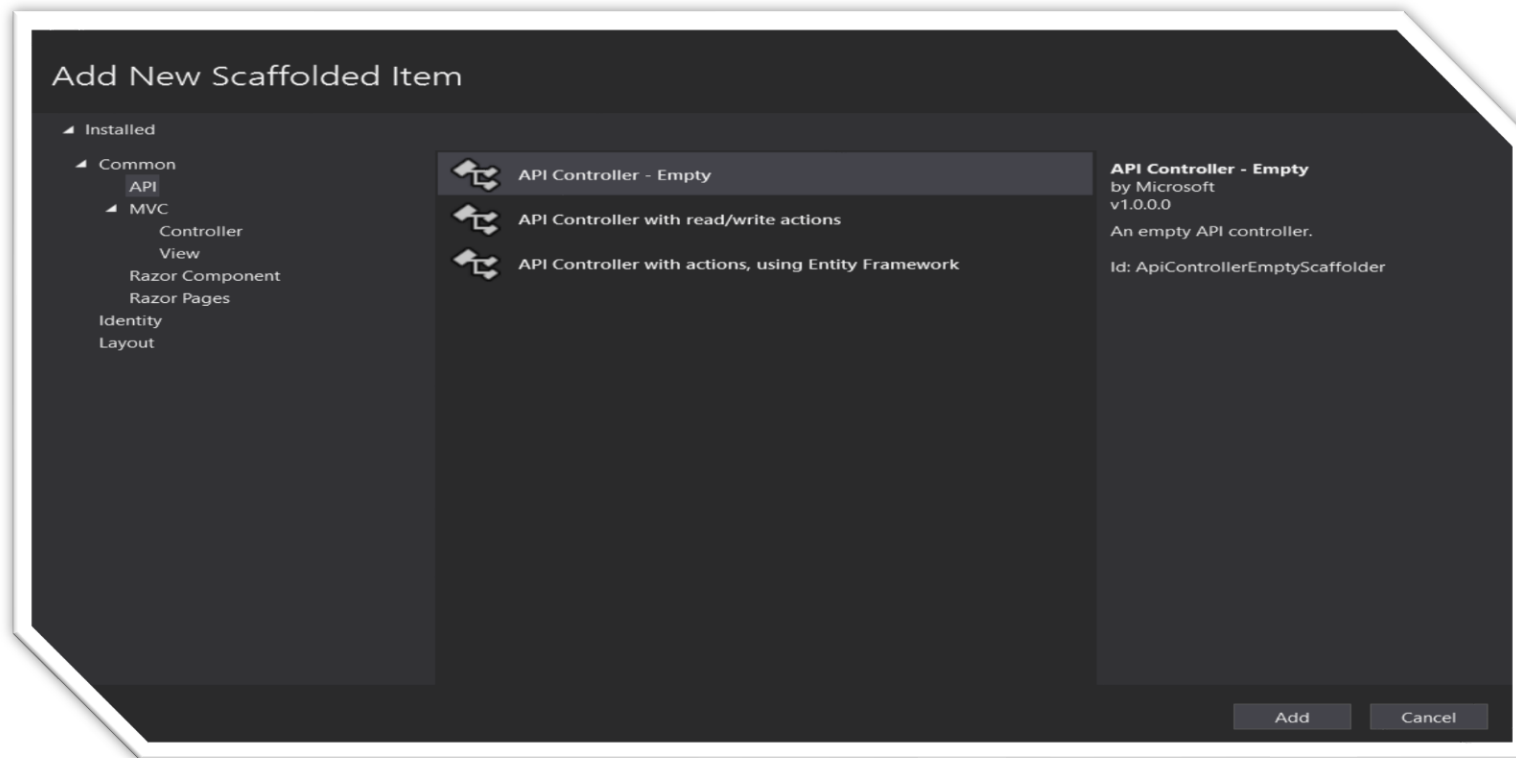
Web API Controller handles incoming HTTP methods requests and sends a response to the caller.

Web API controller class can be created in the Controllers folder or any other folder in the project's root folder.





Right Click on Controllers => Add => Controller => Choose API Controller – Empty => CourseController.





In TahalufLearn.API => Controller => CourseController add the following :

```
private readonly ICourseService courseService;  
  
    public CourseController(ICourseService  
courseService)  
    {  
        this.courseService = courseService;  
    }
```



In TahalufLearn.API => Controller => CourseController add the following :

```
[HttpGet]
    public List<Course> GetAllCourse()
    {
        return courseService.GetAllCourse();
    }
```



In Postman:

1. In URL add => `Https://LocalHost:PortNumber/Api/ControllerName/[RouteName]/[Parameters]`
2. Select The method => (Get)
3. Send the request



GET ▼ https://localhost:44379/api/Course Send ▼

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

KEY	VALUE	DESCRIPTION	...	Bulk Edit
-----	-------	-------------	-----	-----------

Body Cookies Headers (5) Test Results 🌐 Status: 200 OK Time: 1928 ms Size: 704 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ 🔍

```
1  {
2    {
3      "courseid": 1,
4      "coursename": "Angular",
5      "categoreyid": null,
6      "categorey": null,
7      "stdcourses": []
8    },
9    {
10     "courseid": 3,
11     "coursename": null,
12     "categoreyid": null,
13     "categorey": null,
14     "stdcourses": []
```



In TahalufLearn.API => Controller => CourseController add the following :

```
[HttpPost]
    public void CreateCourse(Course course)
    {
        courseService.CreateCourse(course);
    }
```



In Postman:

1. In URL add => `Https://LocalHost:PortNumber/Api/ControllerName/[RouteName]/[Parameters]`
2. Select The method => (Post)
3. Choose Body => raw => JSON => then add the course data as JSON object.
4. Send the request



https://localhost:44379/api/Course

Save

POST

https://localhost:44379/api/Course

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON

Beautify

```
1 {  
2   ...  
3   ... "coursename": "PHP",  
4   ... "categoreyid": 1  
5 }  
6  
7 }
```

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 5.67 s

Size: 135 B

Save Response

Pretty

Raw

Preview

Visualize

Text

```
1
```



In TahalufLearn.API => Controller => CourseController add the following :

```
[HttpDelete]
[Route("delete/{id}")]
public void DeleteCourse(int id)
{
    courseService.DeleteCourse(id);
}
```



In Postman:

1. In URL add => `Https://LocalHost:PortNumber/Api/ControllerName/[RouteName]/[Parameters]`
2. Select The method => (Delete)
 1. Send the request



DELETE

https://localhost:44379/api/Course/Delete/41

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 3.53 s

Size: 135 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1



In TahalufLearn.API => Controller => CourseController add the following :

```
[HttpPut]
    public void UpdateCourse(Course course)
    {
        courseService.UpdateCourse(course);
    }
```




In Postman:

1. In URL add => `Https://LocalHost:PortNumber/Api/ControllerName/[RouteName]/[Parameters]`
2. Select The method => (Put)
3. Choose Body => raw => JSON => then add the course data as JSON object.
4. Send the request



PUT

https://localhost:44379/api/Course

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

1

2

3

4

5

1

2

3

4

5

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 1776 ms

Size: 135 B

Save Response

Pretty

Raw

Preview

Visualize

Text

1



In TahalufLearn.API => Controller => CourseController add the following :

```
[HttpGet]
[Route("getByCourseId/{id}")]
public Course GetByIdCourseId(int id)
{
    return courseService.GetByIdCourseId(id);
}
```



In Postman:

1. In URL add => `Https://LocalHost:PortNumber/Api/ControllerName/[RouteName]/[Parameters]`
2. Select The method => (Get)
 1. Send the request



GET

https://localhost:44379/api/Course/getByCourseId/4

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

5

Body

Cookies

Headers (5)

Test Results

Status: 200 OK

Time: 4.18 s

Size: 267 B

Save Response

Pretty

Raw

Preview

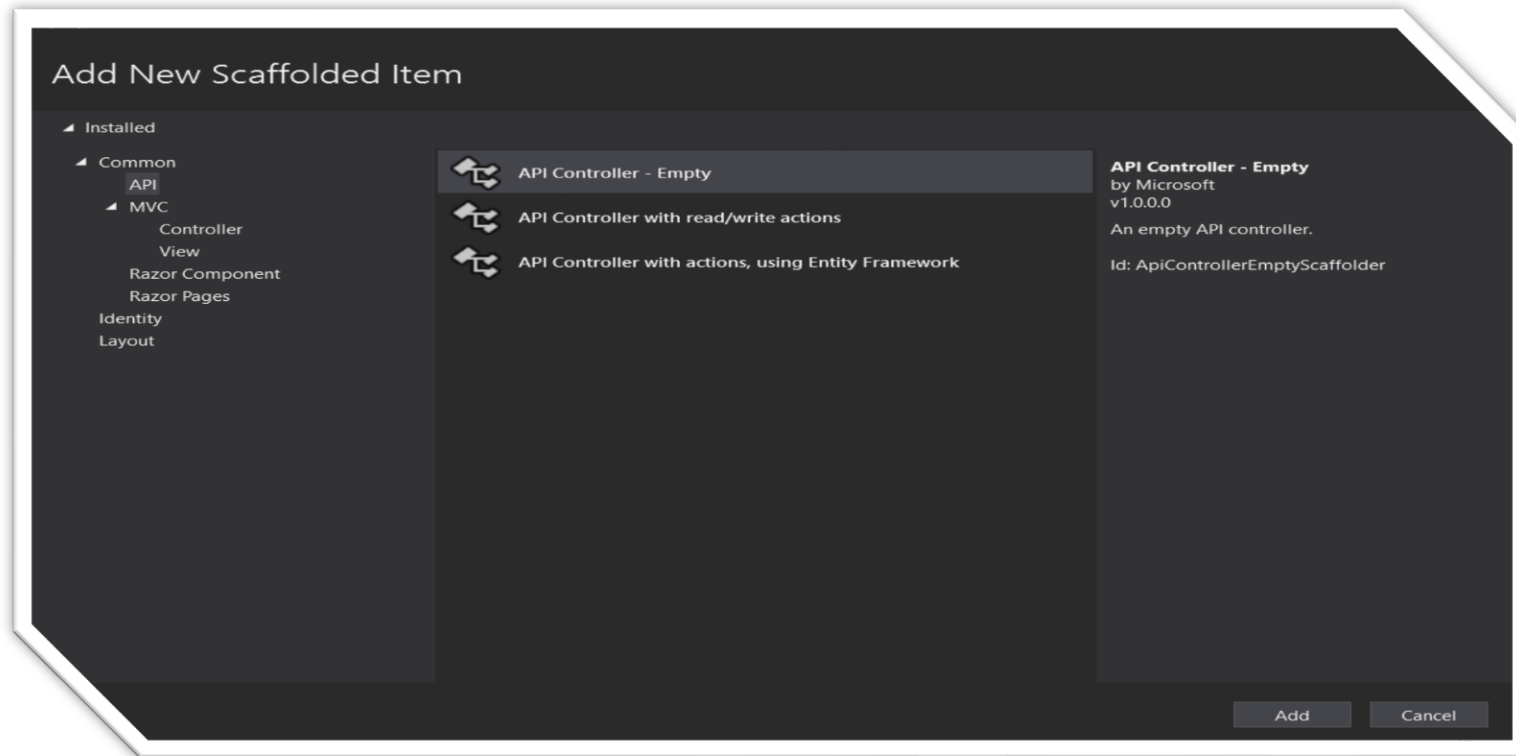
Visualize

JSON

```
1 {
2   "courseid": 4,
3   "coursename": "Oracle",
4   "categoreyid": 1,
5   "categorey": null,
6   "stdcourses": []
7 }
```



Right Click on Controllers => Add => Controller => Choose API Controller – Empty => StudentController.





In TahalufLearn.API => Controller => StudentController add the following :

```
private readonly IStudentService _studentService;  
  
public StudentController(IStudentService Istdservice)  
{  
    this._studentService = Istdservice;  
}
```



In TahalufLearn.API => Controller => StudentController add the following :

```
[HttpGet]
    public List<Student> GetAllStudent()
    {
        return _studentService.GetAllStudent();
    }
```




In TahalufLearn.API => Controller => StudentController add the following :

```
[HttpPost]
    public void CreateStudent(Student Student)
    {
        _studentService.CreateStudent(Student);
    }
```



In TahalufLearn.API => Controller => StudentController add the following :

```
[HttpPut]
    public void UpdateStudent(Student Student)
    {
        _studentService.UpdateStudent(Student);
    }
```



In TahalufLearn.API => Controller => StudentController add the following :

```
[HttpDelete]
[Route("delete/{id}")]
public void DeleteStudent(int id)
{
    _studentService.DeleteStudent(id);
}
```

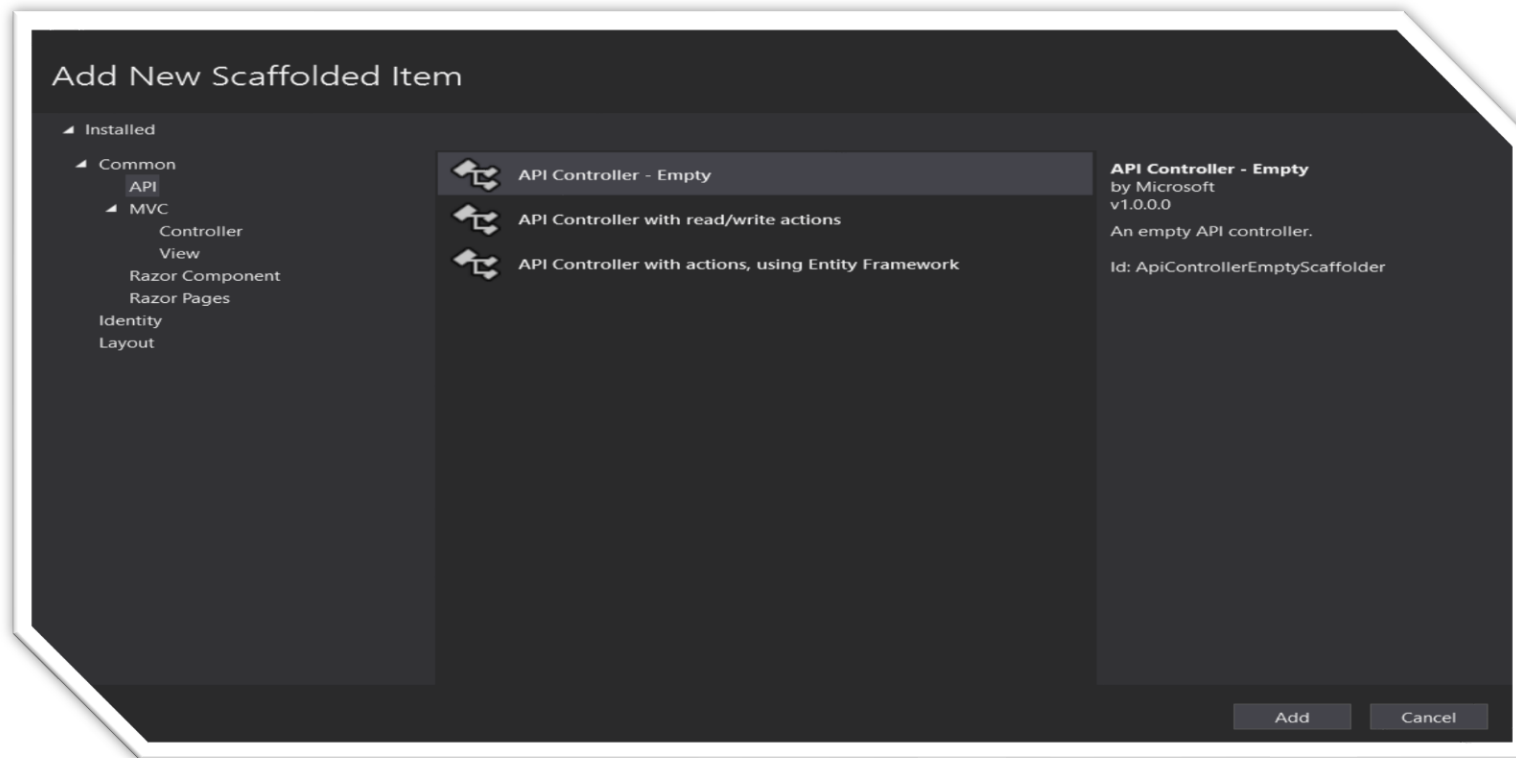


In TahalufLearn.API => Controller => StudentController add the following :

```
[HttpGet]
    [Route("getByStudentId/{id}")]
    public Student GetStudentById(int id)
    {
        return _studentService.GetStudentById(id);
    }
```



Right Click on Controllers => Add => Controller => Choose API Controller – Empty => StudentCourseController.





In TahalufLearn.API => Controller => StudentCourseController add the following :

```
private readonly IStudentCourseService _studentCourseService;  
  
public StudentCourseController(IStudentCourseService  
studentCourseService)  
{  
    _studentCourseService = studentCourseService;  
}
```



In TahalufLearn.API => Controller => StudentCourseController add the following :

```
[HttpPost]
    public void CreateStudentCourse(stdCourse studentCourse)
    {
        _studentCourseService.CreateStudentCourse(studentCourse);
    }
```



In TahalufLearn.API => Controller => StudentCourseController add the following :

```
[HttpDelete]
[Route("delete/{id}")]
public void DeleteStudentCourse(int id)
{
    _studentCourseService.DeleteStudentCourse(id);
}
```




In TahalufLearn.API => Controller => StudentCourseController add the following :

```
[HttpGet]
    public List<StdCourse> GetAllStudentCourse()
    {
        return _studentCourseService.GetAllStudentCourse();
    }
```



In TahalufLearn.API => Controller => StudentCourseController add the following :

```
[HttpGet]
    [Route("getStudentCourseById/{id}")]
    public StdCourse GetStudentCourseById(int id)
    {
        return _studentCourseService.GetStudentCourseById(id);
    }
```



In TahalufLearn.API => Controller => StudentCourseController add the following :

```
[HttpPut]
public void UpdateStudentCourse(stdCourse studentCourse)
{
    _studentCourseService.UpdateStudentCourse(studentCourse);
}
```



Exercise

- ✓ Create a function to display FirstName and LastName from table student.
- ✓ Create a function to display students by firstName.
- ✓ Create a function to display students by BirthOfDate.
- ✓ Create a function to display a student by BirthOfDate interval.
- ✓ Create a function to display the student name with the highest n(2,3,...) marks

A decorative graphic consisting of a green rounded rectangle with a red outline. The red outline is irregular, with a small circle at the bottom left and a small solid red circle at the top right. Below the green rectangle are two small black circles.

Upload Image



In TahalufLearn.API => Right Click => Add New Folder (Images):

Create upload image function in Course Controller:

```
[Route("uploadImage")]  
[HttpPost]  
public Course UploadImage()  
{  
    var file = Request.Form.Files[0];  
    var fileName = Guid.NewGuid().ToString() + "_" + file.FileName;  
    var fullPath = Path.Combine("Images", fileName);
```



Create upload image function in Course Controller:

```
using (var stream = new FileStream(fullPath, FileMode.Create))
{
    file.CopyTo(stream);
}
Course item = new Course();
item.ImageName = fileName;
return item;
}
```



```
[Route("uploadImage")]
[HttpPost]
0 references
public Course UploadImage()
{
    var file = Request.Form.Files[0];
    var fileName = Guid.NewGuid().ToString() + "_" + file.FileName;
    var fullPath = Path.Combine("Images", fileName);
    using (var stream = new FileStream(fullPath, FileMode.Create))
    {
        file.CopyTo(stream);
    }
    Course item = new Course();
    item.ImageName = fileName;
    return item;
}
```

```
}
return item;
item.ImageName = fileName;
Course item = new Course();
```




POST

https://localhost:44379/API/Course/uploadImage

Send

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettingsCookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>		<div>File<div>Select Files</div></div>			
	Key	<div>Text<div>Value</div></div>	Description		
		<div>File</div>			

Response

