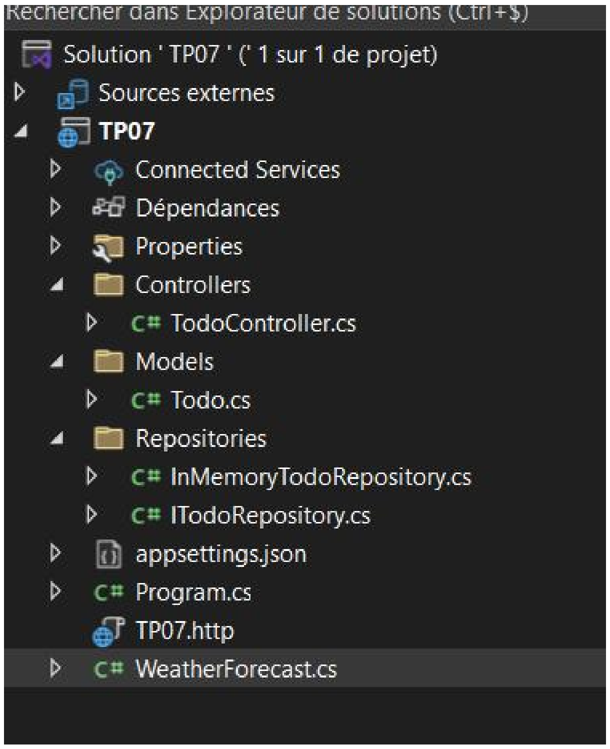
Compte rendu (TP 7)   Grpe B

Génie.Info 2

ADOGNIBO Hortice

TP7


Rechercher dans Explorateur de solutions (Ctrl+$)

```
Solution ' TP07 ' (' 1 sur 1 de projet)
  Sources externes
  TP07
    Connected Services
    Dépendances
    Properties
    Controllers
      TodoController.cs
    Models
      Todo.cs
    Repositories
      InMemoryTodoRepository.cs
      ITodoRepository.cs
    appsettings.json
    Program.cs
    TP07.http
    WeatherForecast.cs
```

Models

Todo.cs:

```
using System.ComponentModel.DataAnnotations;

namespace TP07.Models

{

    public class Todo

    {

        public int Id { get; set; }

        [Required(ErrorMessage = "Le titre est obligatoire.")]

        [MaxLength(100, ErrorMessage = "Le titre ne doit pas dépasser 100 
caractères.")]
```

```
        public string Title { get; set; }

        public bool IsDone { get; set; }

    }

}
```

Repositories:

ITodoRepository.cs:

```
using System.Collections.Generic;

using TP07.Models;


namespace TP07.Repositories

{

    public interface ITodoRepository

    {

        IEnumerable<Todo> GetAll();!;

        Todo? GetById(int id);

        Todo Add(Todo todo);

        bool Update(int id, Todo todo);

        bool Delete(int id);

    }

}
```

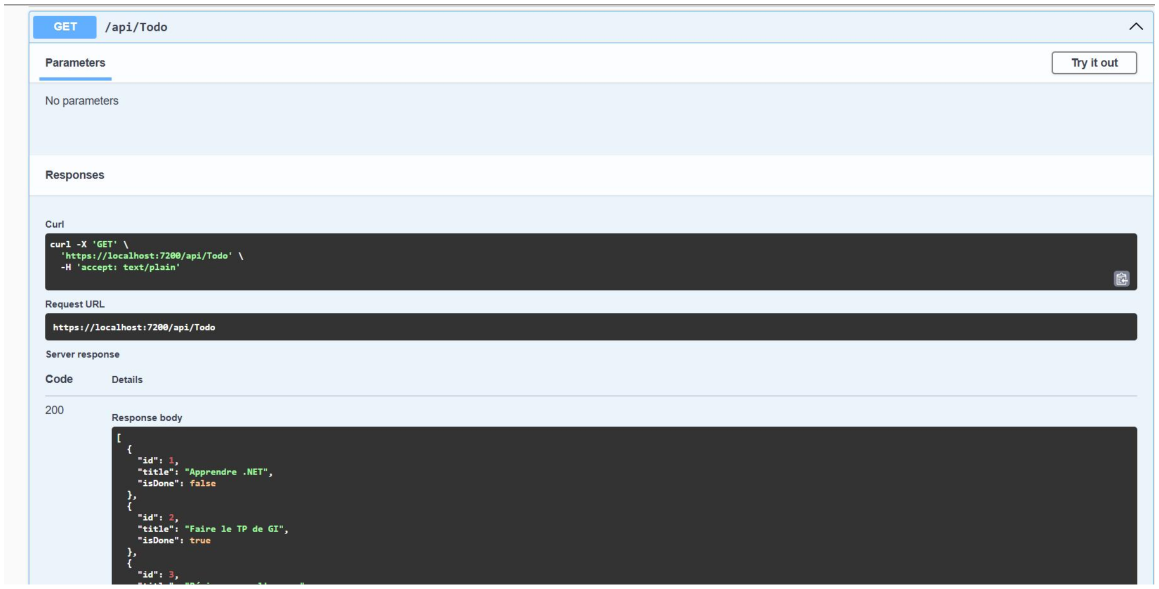
InMemoryTodoRepository.cs:

```
using System.Collections.Generic;

using System.Linq;

using System.Xml.Linq;

using TP07.Models;

using TP07.Repositories;
```

```
namespace TP07.Repositories
{
    public class InMemoryTodoRepository : ITodoRepository
    {
        private static List<Todo> _todos = new List<Todo>();

        private static int _nextId = 1;

        public InMemoryTodoRepository()
        {
            if (_todos.Count == 0)
            {
                _todos.Add(new Todo { Id = _nextId++, Title =
"Apprendre .NET", IsDone = false });
                _todos.Add(new Todo { Id = _nextId++, Title = "Faire le TP de
GI", IsDone = true });
                _todos.Add(new Todo { Id = _nextId++, Title = "Réviser pour
l'examen", IsDone = false });
            }
        }
        public IEnumerable<Todo> GetAll()
        {
            return _todos;
        }
        public Todo? GetById(int id)
        {
            return _todos.FirstOrDefault(t => t.Id == id);
        }
        }

        public Todo Add(Todo todo)
        {
            todo.Id = _nextId++;
            _todos.Add(todo);
            return todo;
        }

        public bool Update(int id, Todo updatedTodo)
        {
            var existingTodo = GetById(id);
            if (existingTodo == null)
                return false;

            existingTodo.Title = updatedTodo.Title;
            existingTodo.IsDone = updatedTodo.IsDone;

            return true;
        }

        public bool Delete(int id)
        {
            var todo = GetById(id);
            if (todo == null)
                return false;

            return _todos.Remove(todo);
        }
    }
}
```

Program.cs:

```
using TP07.Repositories;


var builder = WebApplication.CreateBuilder(args);



builder.Services.AddSingleton<ITodoRepository, InMemoryTodoRepository>();


builder.Services.AddControllers();


// Swagger

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();


var app = builder.Build();


// Dev tools

if (app.Environment.IsDevelopment())

{

    app.UseSwagger();

    app.UseSwaggerUI();

}


app.UseHttpsRedirection();


app.UseAuthorization();


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



/api/Todo

Parameters

No parameters

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7200/api/Todo' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7200/api/Todo
```

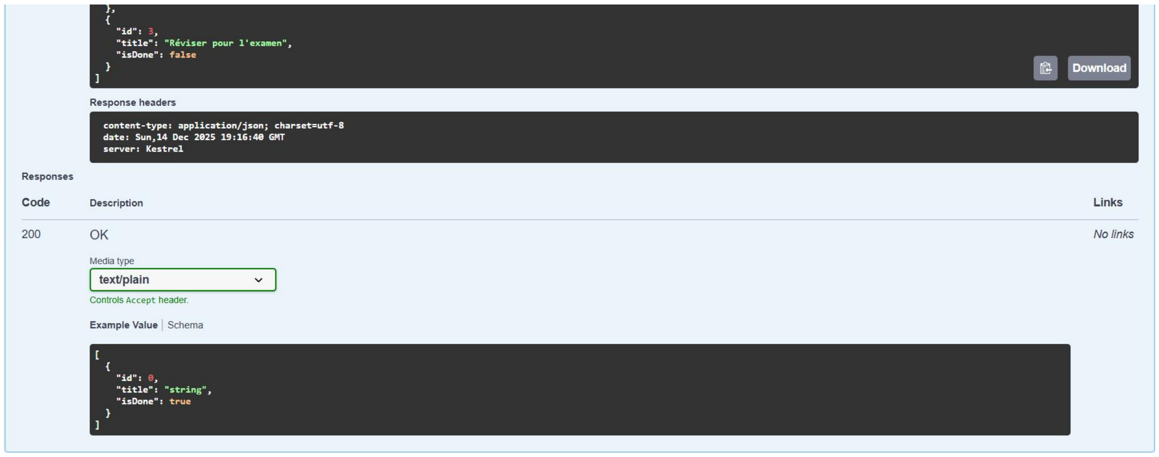
Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "title": "Apprendre .NET",
    "isDone": false
  },
  {
    "id": 2,
    "title": "Faire le TP de GI",
    "isDone": true
  },
  {
    "id": 3,
```



```
  },
  {
    "id": 3,
    "title": "Réviser pour l'examen",
    "isDone": false
  }
]
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sun,14 Dec 2025 19:16:40 GMT
server: Kestrel
```

Responses

Code Description Links

200 OK *No links*

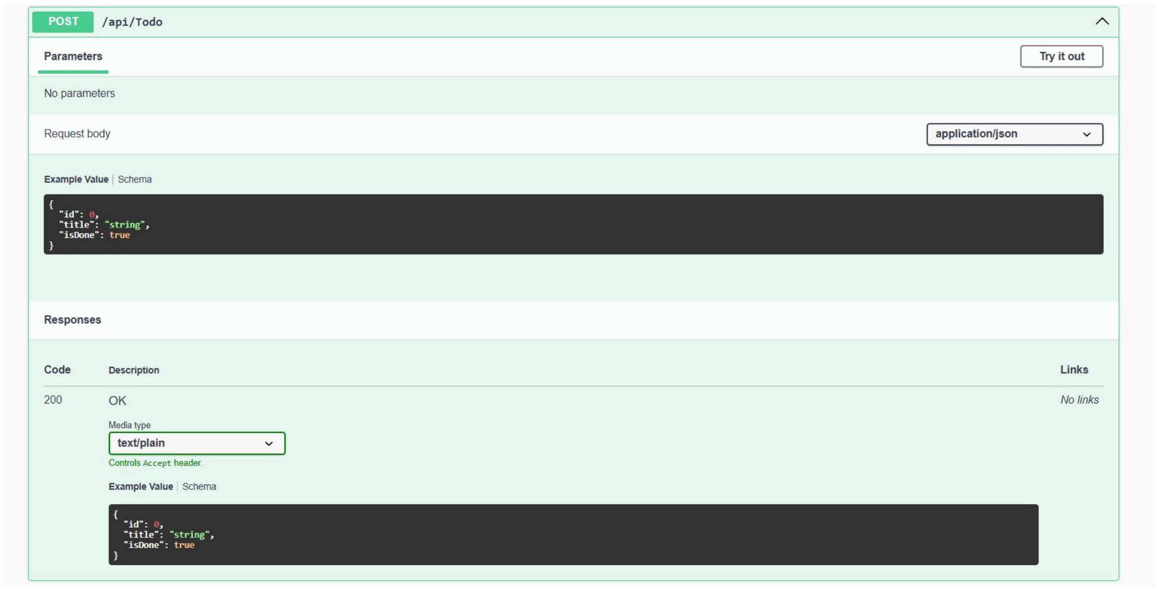
Media type

text/plain

Controls Accept header.

Example Value | Schema

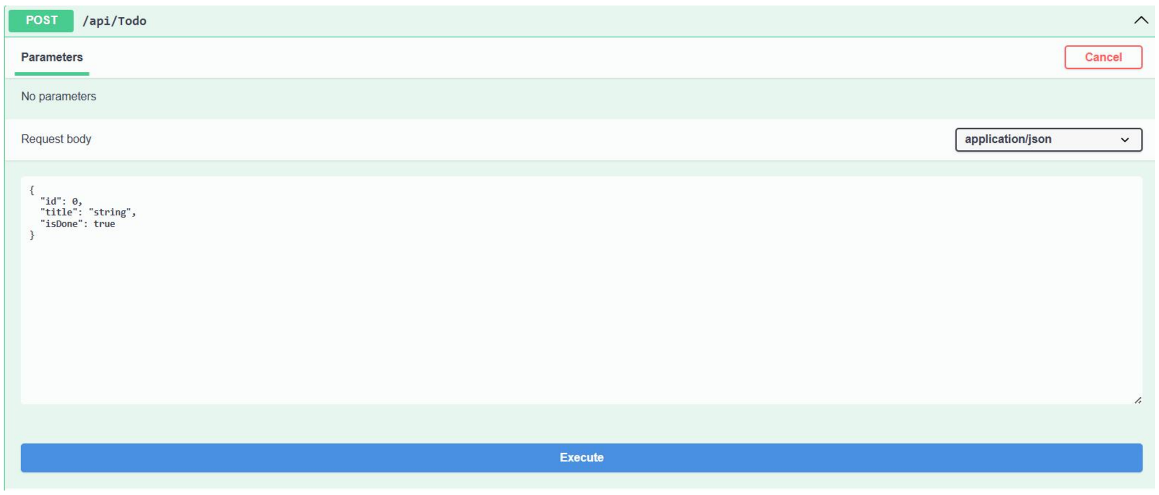
```
[
  {
    "id": 0,
    "title": "string",
    "isDone": true
  }
]
```



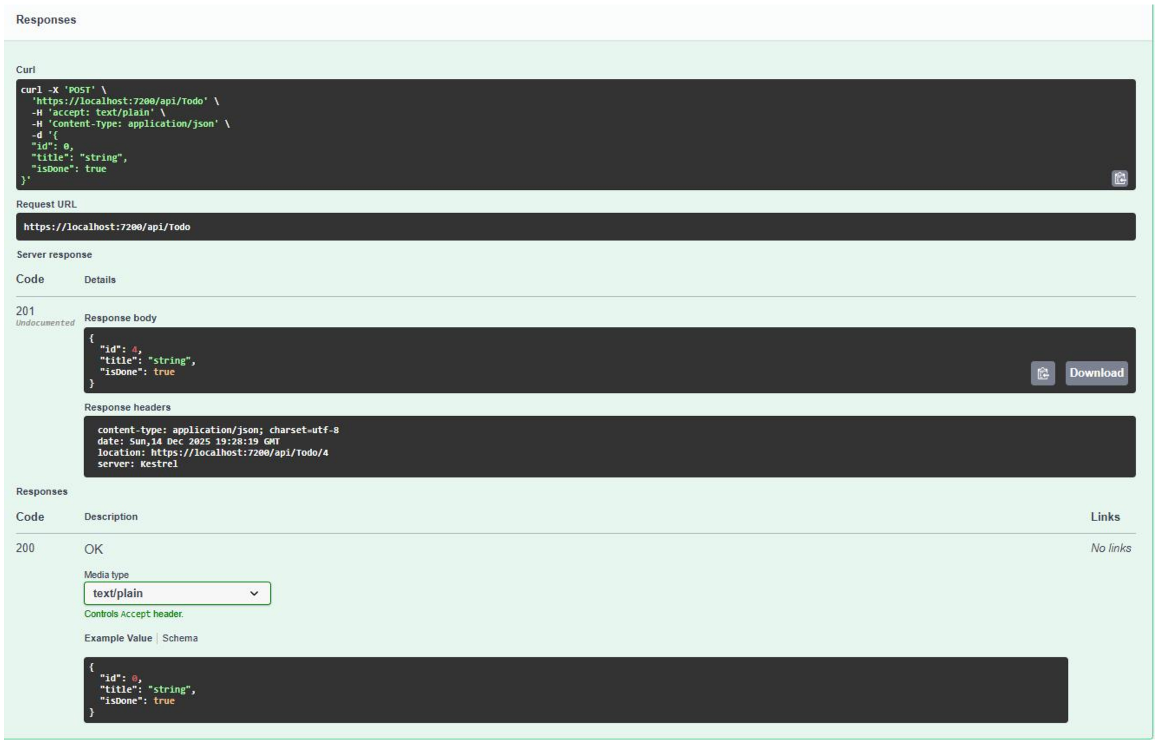
```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```

Code	Description	Links
200	OK	No links

```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```



```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```


Responses

Curl

```
curl -X 'POST' \
  'https://localhost:7200/api/Todo' \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 0,
  "title": "string",
  "isDone": true
}'
```

Request URL

```
https://localhost:7200/api/Todo
```

Server response

Code	Details
201 *Undocumented*	Response body

```
{
  "id": 4,
  "title": "string",
  "isDone": true
}
```

Response headers

```
 content-type: application/json; charset=utf-8
 date: Sun,14 Dec 2025 19:28:19 GMT
 location: https://localhost:7200/api/Todo/4
 server: Kestrel
```

Responses

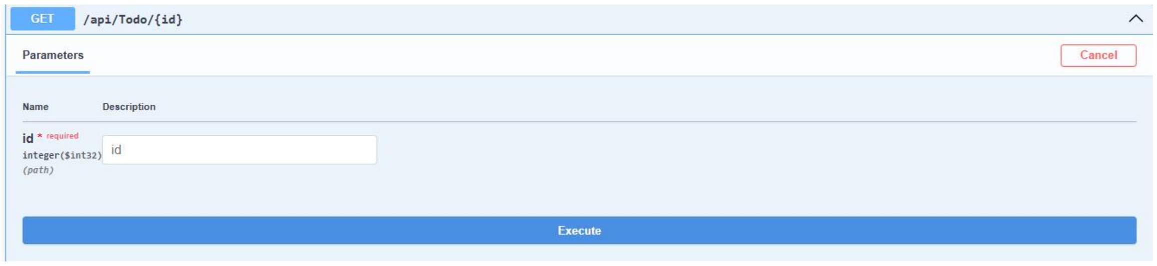
Code	Description	Links
200	OK	*No links*

Media type: text/plain

Controls Accept header.

Example Value | Schema

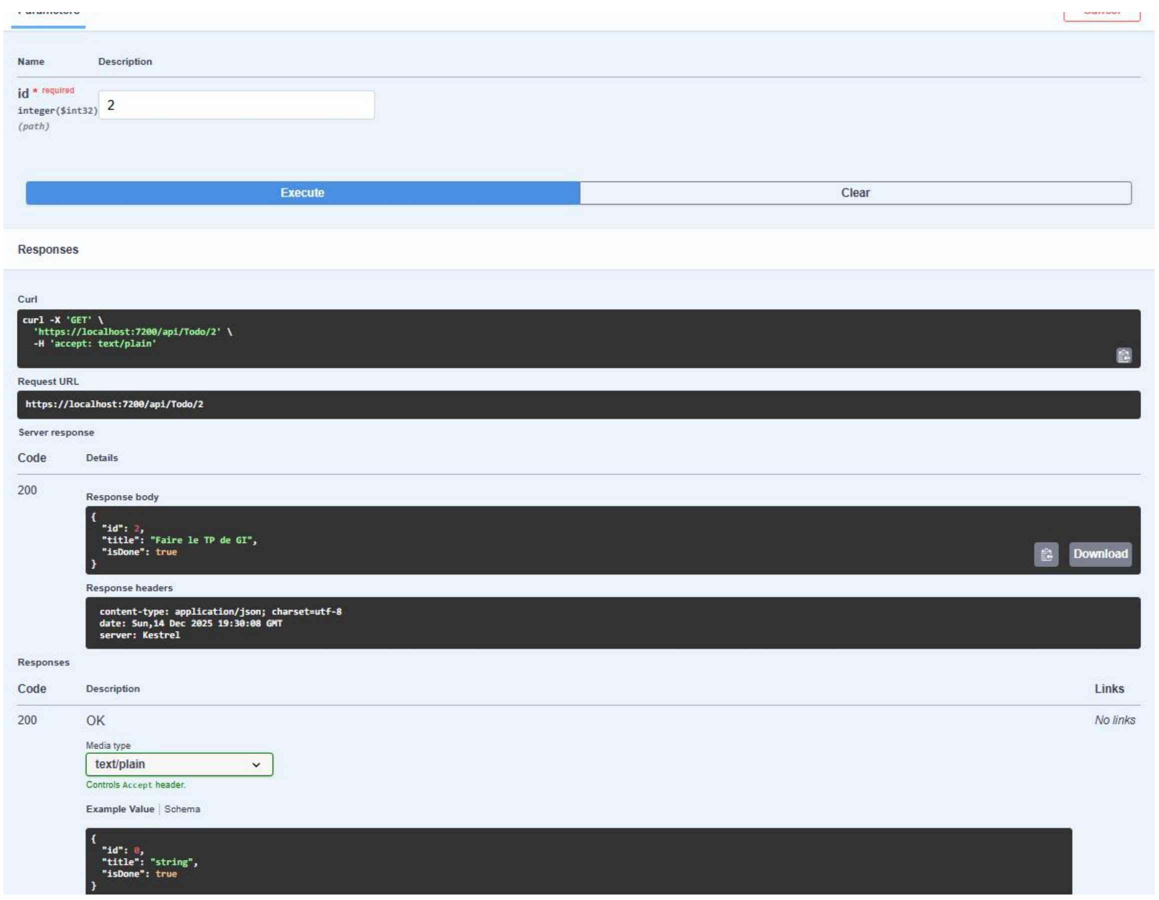
```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```



/api/Todo/{id}

Parameters

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



Name Description

id * required
integer($int32)
(path)
2

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7200/api/Todo/2' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7200/api/Todo/2
```

Server response

Code Details

200

Response body

```
{
  "id": 2,
  "title": "Faire le TP de GI",
  "isDone": true
}
```

Download

Response headers

```
 content-type: application/json; charset=utf-8
 date: Sun,14 Dec 2025 19:30:08 GMT
 server: Kestrel
```

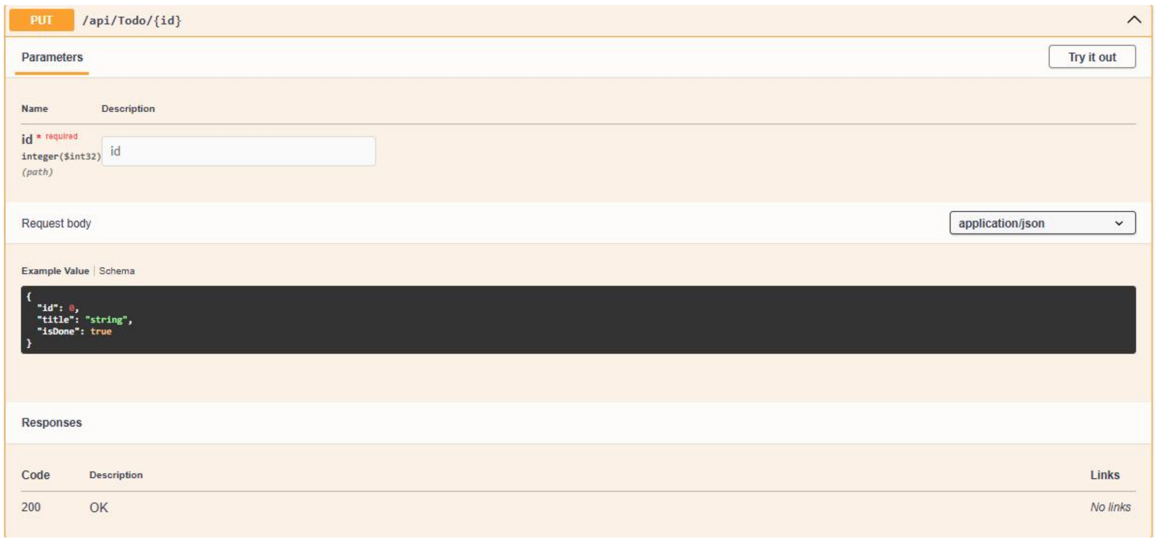
Responses

Code	Description	Links
200	OK	*No links*

Media type
text/plain
Controls `Accept` header.

Example Value | Schema

```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```



PUT /api/Todo/{id}

Parameters Try it out

Name Description

id * required
integer($int32)
(path)
id

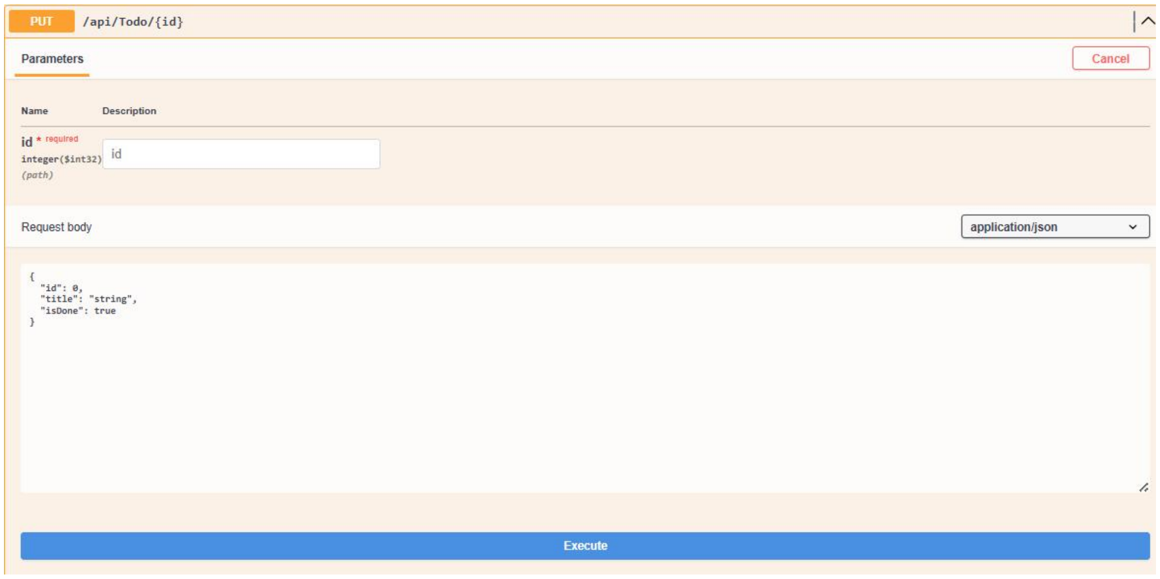
Request body application/json

Example Value | Schema

```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```

Responses

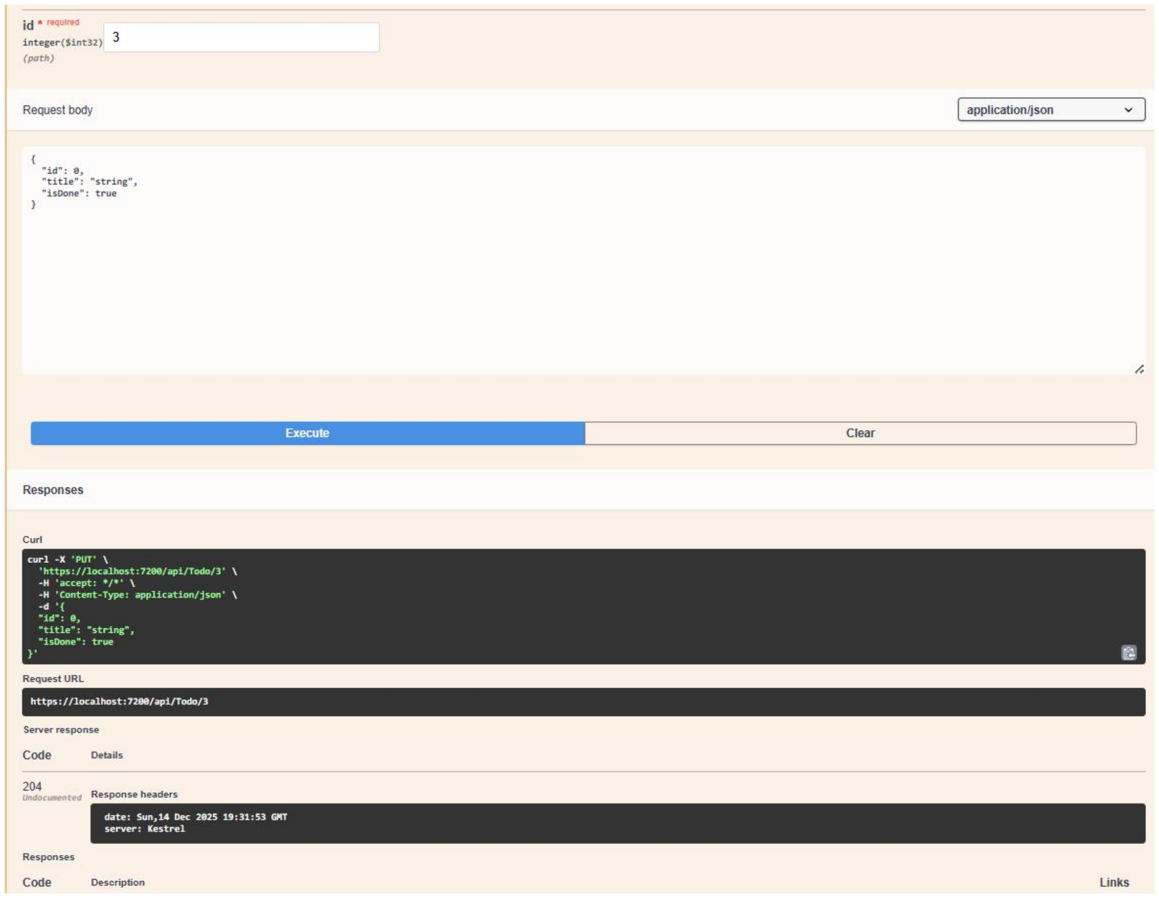
Code	Description	Links
200	OK	*No links*



id * required
integer($int32)
(path)
id

Request body application/json

```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```


id * required
integer($int32)
(path)
3

Request body application/json

```
{
  "id": 0,
  "title": "string",
  "isDone": true
}
```

Curl

```
curl -X 'PUT' \
  'https://localhost:7200/api/Todo/3' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 0,
  "title": "string",
  "isDone": true
}'
```

Request URL

```
https://localhost:7200/api/Todo/3
```

Server response

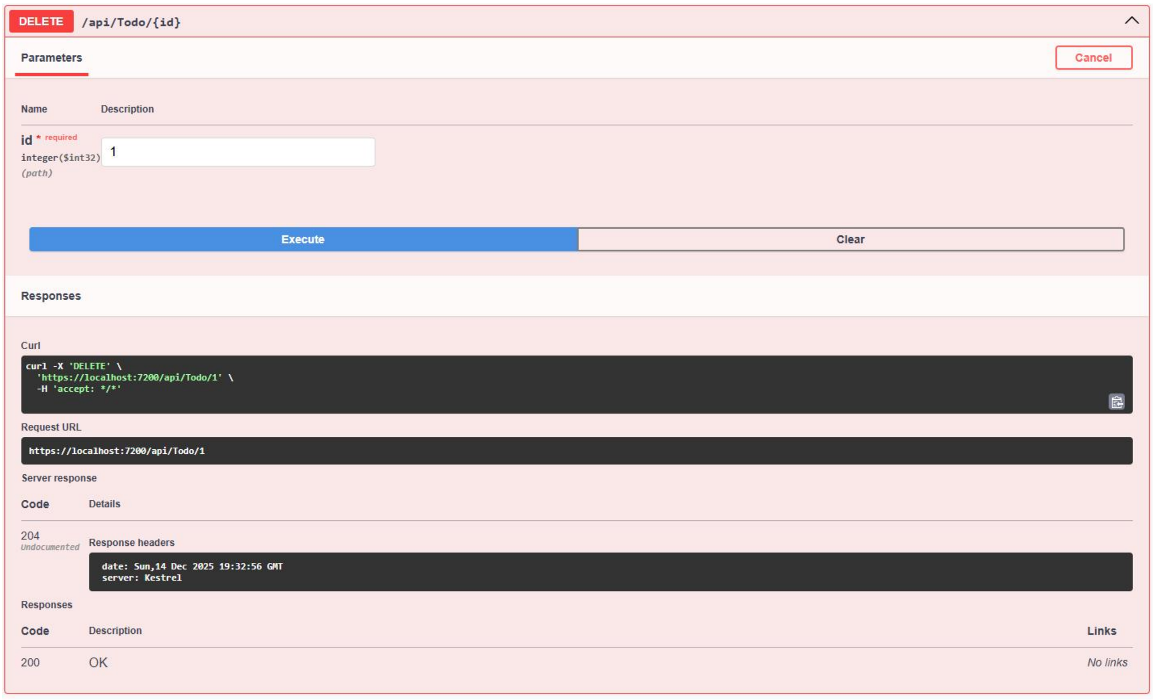
Code Details

204
Undocumented
Response headers

```
date: Sun,14 Dec 2025 19:31:53 GMT
server: Kestrel
```

Responses

Code Description Links



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

Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7200/api/Todo/1' \
  -H 'accept: */*'
```

Request URL

```
https://localhost:7200/api/Todo/1
```

Server response

Code	Details
204 Undocumented	Response headers

```
date: Sun,14 Dec 2025 19:32:56 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

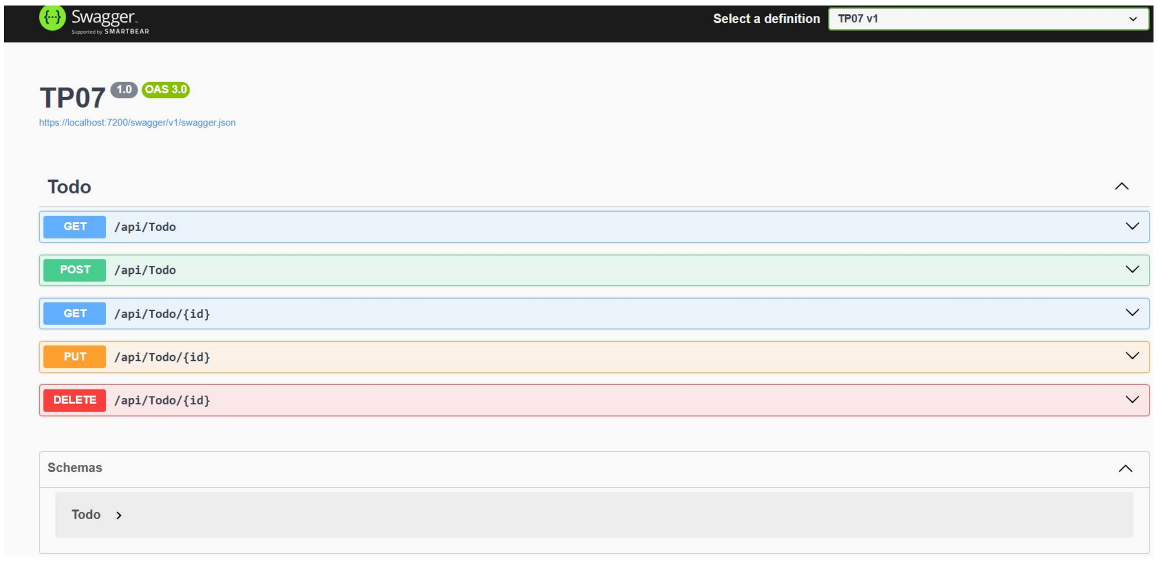
Controllers:

TodoController.cs:

```
using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

using TP07.Models;

using TP07.Repositories;


namespace TP07.Controllers

{

    [ApiController]

    [Route("api/[controller]")]

    public class TodoController : ControllerBase

    {

        private readonly ITodoRepository _todoRepository;

        public TodoController(ITodoRepository todoRepository)

        {
            _todoRepository = todoRepository;

        }

        [HttpGet]

        public IEnumerable<Todo> GetAll()

        {

            return _todoRepository.GetAll();

        }

        [HttpGet("{id}")]

        public ActionResult<Todo> GetById([FromRoute] int id)

        {

            var todo = _todoRepository.GetById(id);

            if (todo == null)

            {

                return NotFound();

            }

            return Ok(todo);

        }

        [HttpPost]

        public ActionResult<Todo> Create([FromBody] Todo todo)

        {

            if (todo == null || string.IsNullOrWhiteSpace(todo.Title))

            {

                return BadRequest("Le Todo est invalide.");

            }

            _todoRepository.Add(todo);


            return CreatedAtAction(nameof(GetById), new { id = todo.Id },
todo);

        }
```

```
[HttpPut("{id}")]
public ActionResult Update([FromRoute] int id, [FromBody] Todo todo)
{
    if (todo == null || string.IsNullOrWhiteSpace(todo.Title))
    {
        return BadRequest("Le Todo est invalide.");
    }

    var existingTodo = _todoRepository.GetById(id);
    if (existingTodo == null)
    {
        return NotFound();
    }

    _todoRepository.Update(id, todo);

    return NoContent();
}
[HttpDelete("{id}")]
public ActionResult Delete([FromRoute] int id)
{
    var existingTodo = _todoRepository.GetById(id);
    if (existingTodo == null)
    {
        return NotFound();
    }

    _todoRepository.Delete(id);
```

```
                return NoContent();
            }


        }
    }
```



TP07 1.0 OAS 3.0

https://localhost:7200/swagger/v1/swagger.json

Todo

GET /api/Todo

POST /api/Todo

GET /api/Todo/{id}

PUT /api/Todo/{id}

DELETE /api/Todo/{id}

Schemas

Todo