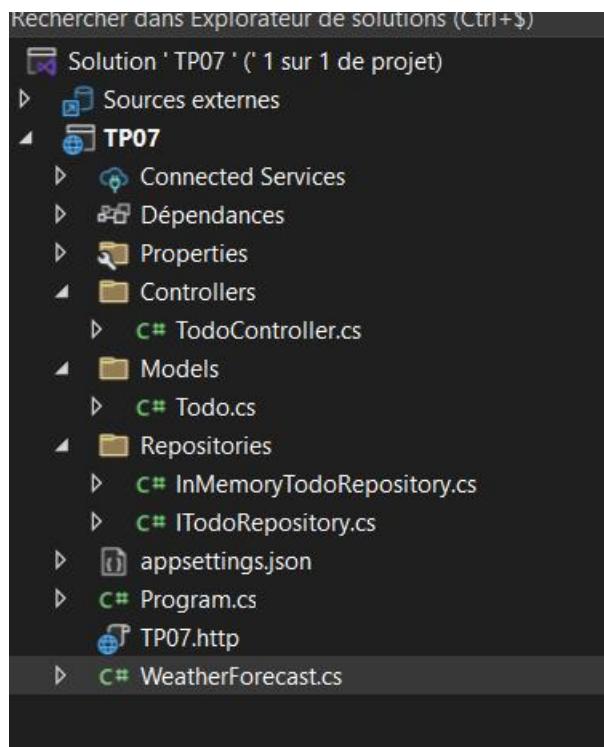


ADOGNIBO Hortice

TP7



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();
```

The screenshot shows a Swagger UI interface for a .NET application. At the top, there is a blue button labeled "GET" and the URL "/api/Todo". Below this, there are sections for "Parameters" (which says "No parameters") and "Responses". Under "Responses", there is a "Curl" section with a code block:

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

Below the curl section is a "Request URL" field containing "https://localhost:7200/api/Todo". Under "Server response", there is a table with two rows:

Code	Details
200	Response body

The "Response body" section displays a JSON array of three todo items:

```
[{"id": 1, "title": "Apprendre .NET", "isDone": false}, {"id": 2, "title": "Faire le TP de GI", "isDone": true}, {"id": 3, "title": "Réviser pour l'examen", "isDone": false}]
```

At the bottom of the "Response body" section are "Download" and "Copy" buttons. Below the response body, there is a "Response headers" section with the following content:

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

Under "Responses", there is another table:

Code	Description	Links
200	OK	No links

In the "Description" column for the 200 OK response, there is a "Media type" dropdown set to "text/plain". Below the dropdown, it says "Controls Accept header." and provides "Example Value" and "Schema" links.

POST /api/Todo

Parameters

No parameters

Request body

application/json

Example Value | Schema

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

Responses

Code	Description	Links
200	OK	No links

Media type

text/plain

Controls Accept header.

Example Value | Schema

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

POST /api/Todo

Parameters

No parameters

Request body

application/json

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

Cancel

Execute

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 <pre>{ "id": 4, "title": "string", "isdone": true }</pre> Response headers <pre>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</pre>

Responses

Code	Description	Links
200 OK	Media type <input checked="" type="button" value="text/plain"/> <input type="button" value="application/json"/> Controls Accept header. Example Value Schema <pre>{ "id": 0, "title": "string", "isdone": true }</pre>	No links

GET /api/Todo/{id}

Parameters

Name	Description
id * required integer(\$int32)	<input type="text" value="id"/> (path)

Parameters

Name	Description
id * required	integer(\$int32) 2 (path)

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 19:30:00 GMT
server: Kestrel
```

Responses

Code Description

200 OK

Links

Media type

text/plain

Controls Accept header.

[Example Value](#) | [Schema](#)

```
{
  "id": 2,
  "title": "string",
  "isDone": true
}
```

PUT /api/Todo/{id}

Parameters

[Try it out](#)

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

Request body

application/json

[Example Value](#) | [Schema](#)

```
{
  "id": 2,
  "title": "string",
  "isDone": true
}
```

Responses

Code	Description	Links
200	OK	No links

PUT /api/Todo/{id}

Parameters

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

Request body

application/json

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

Execute

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

Request body

application/json

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

Execute Clear

Responses

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 <small>Undocumented</small>	Response headers date: Sun,14 Dec 2025 19:31:53 GMT server: Kestrel

Responses

Code	Description	Links
------	-------------	-------

The screenshot shows a REST API documentation interface for a DELETE request to `/api/Todo/{id}`. The `id` parameter is required and has a value of 1. The interface includes sections for Parameters, Responses, Curl, Request URL, Server response, and a detailed Response headers section.

Parameters

Name	Description
<code>id</code> * required	integer(\$int32) 1 (path)

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**

200 OK

Links

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();
    }

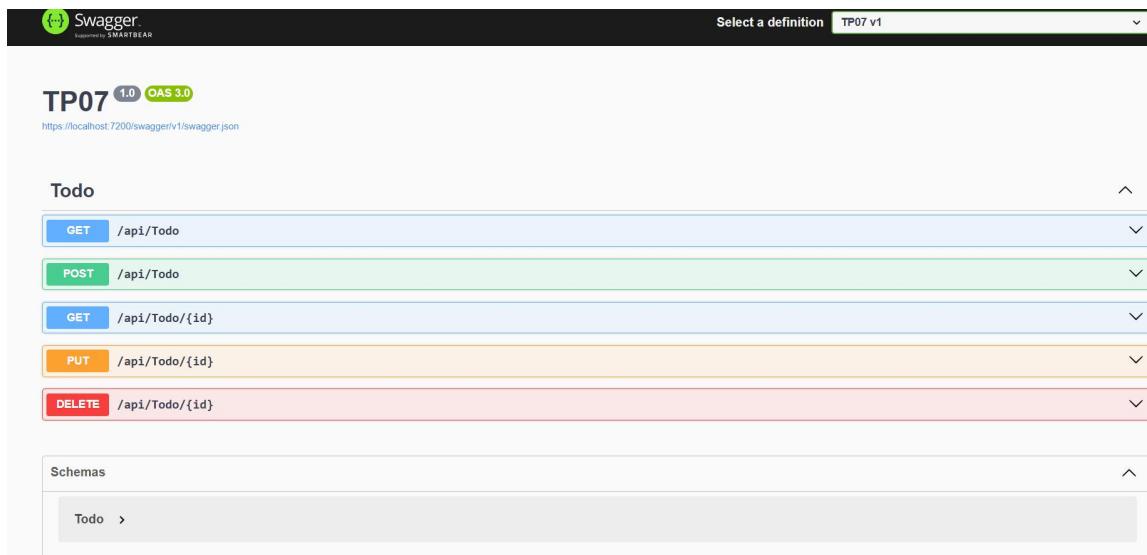
    _todoRepository.Update(id, todo);

    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();  
  
    }  
  
}  
  
}
```



The screenshot shows the Swagger UI interface for the TP07 v1 API definition. At the top, there is a header bar with the Swagger logo and the text "Select a definition" followed by "TP07 v1". Below the header, the title "TP07" is displayed with a version "1.0" and an "OAS 3.0" badge. The URL "https://localhost:7200/swagger/v1/swagger.json" is also shown. The main content area is titled "Todo" and contains a list of API endpoints:

- GET /api/Todo** (blue button)
- POST /api/Todo** (green button)
- GET /api/Todo/{id}** (light blue button)
- PUT /api/Todo/{id}** (orange button)
- DELETE /api/Todo/{id}** (red button)

Below the endpoints, there is a section titled "Schemas" which contains a single entry: "Todo >".