

**TUGAS PENDAHULUAN
KONSTRUKSI PERANGKAT LUNAK**

**MODUL IX
API DESIGN & CONSTRUCTION USING SWAGGER**



Disusun Oleh:

Aorinka Anendya Chazanah / 2211104013

S1 SE-06-01

Dosen Pengampu:

Yudha Islami Sulistya, S.Kom., M.Cs

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

TUGAS PENDAHULUAN

1. IMPLEMENTASI WEB API

Dari master/main branch dan class utama, buatlah program/aplikasi web API dari spesifikasi sebagai berikut ini:

A. API yang dibuat menggunakan data dari kelas Mahasiswa.

Mahasiswa
- Nama: string
- Nim: string
+ Mahasiswa

B. API yang dibuat mempunyai lokasi sebagai berikut ‘/api/mahasiswa’, URL domain boleh dari port mana saja, misalnya <https://localhost:5001/api/mahasiswa> (port bebas)

C. Secara default, program yang dibuat memiliki array/list mahasiswa dari anggota kelompok anda (tuliskan nama anda di urutan pertama/paling atas), contohnya:

- Nama: “LeBron James”, Nim: “1302000001”
- Nama: “Stephen Curry”, Nim: “1302000002”
- dst.

D. Gunakan teknologi API sehingga program tersebut dapat menerima HTTP request sebagai berikut:

Mahasiswa	
GET	/api/Mahasiswa
POST	/api/Mahasiswa
GET	/api/Mahasiswa/{id}
DELETE	/api/Mahasiswa/{id}

- GET /api/mahasiswa: mengembalikan output berupa list/array dari semua objek mahasiswa yang tersimpan
- GET /api/mahasiswa/{index}: mengembalikan output berupa objek mahasiswa untuk index ke-‘index’
- POST /api/mahasiswa: menambahkan objek mahasiswa baru dengan menyertakan nama dan nim
- DELETE /api/mahasiswa/{index}: menghapus objek mahasiswa dengan index ke ‘index’

- E. Implementasi yang dibuat tidak menggunakan database, cukup disimpan sebagai suatu variable, dan gunakan “static” di variable tersebut yang menyimpan list/array dari objek-objek mahasiswa
- F. Dalam pembuatan program/aplikasi ini, anda dapat mengasumsikan bahwa input dari user selalu benar dan sesuai dengan tipe data yang diharapkan

JAWABAN

➤ Source code

a. File MahasiswaController.cs

```
1  using Microsoft.AspNetCore.Mvc;
2  using System.Collections.Generic;
3
4  namespace tpmodul9_2211104013.Controllers
5  {
6      [ApiController]
7      [Route("api/[controller]")]
8      public class MahasiswaController : ControllerBase
9      {
10         public class Mahasiswa
11         {
12             public string Nama { get; set; }
13             public string Nim { get; set; }
14         }
15
16         private static List<Mahasiswa> listMahasiswa = new List<Mahasiswa>
17         {
18             new Mahasiswa { Nama = "Aorinka Anendya Chazanah", Nim = "2211104013" },
19             new Mahasiswa { Nama = "Muhammad Abdul Aziz", Nim = "2211104026" },
20             new Mahasiswa { Nama = "Fadhila Agil Permana", Nim = "2211104006" },
21             new Mahasiswa { Nama = "Muhammad Luthfi Hamdani", Nim = "2211104020" },
22         };
23
24         [HttpGet]
25         public IEnumerable<Mahasiswa> Get()
26         {
27             return listMahasiswa;
28         }
29
30         [HttpGet("{id}")]
31         public ActionResult<Mahasiswa> Get(int id)
32         {
33             if (id < 0 || id >= listMahasiswa.Count)
34                 return NotFound();
35             return listMahasiswa[id];
36         }
37
38         [HttpPost]
39         public void Post([FromBody] Mahasiswa mhs)
40         {
41             listMahasiswa.Add(mhs);
42         }
43
44         [HttpDelete("{id}")]
45         public void Delete(int id)
46         {
47             if (id >= 0 && id < listMahasiswa.Count)
48             {
49                 listMahasiswa.RemoveAt(id);
50             }
51         }
52     }
53 }
54
```

b. File Mahasiswa.cs dalam folder Models

```
1 namespace tpmodul9_2211104013.Models
2 {
3     0 references
4     public class Mahasiswa
5     {
6         0 references
7         public string Nama { get; set; }
8         0 references
9         public string Nim { get; set; }
10    }
11 }
```

c. File program.cs

```
1 var builder = WebApplication.CreateBuilder(args);
2
3 // Add services to the container.
4
5 builder.Services.AddControllers();
6 // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
7 builder.Services.AddEndpointsApiExplorer();
8 builder.Services.AddSwaggerGen();
9
10 var app = builder.Build();
11
12 // Configure the HTTP request pipeline.
13 if (app.Environment.IsDevelopment())
14 {
15     app.UseSwagger();
16     app.UseSwaggerUI();
17 }
18
19 app.UseHttpsRedirection();
20
21 app.UseAuthorization();
22
23 app.MapControllers();
24
25 app.Run();
```

➤ **Hasil running program**

Swagger UI showing the API endpoints for the 'tpmodul9_2211104013' API. The UI displays two main sections: 'Mahasiswa' and 'WeatherForecast'. The 'Mahasiswa' section lists four endpoints: GET /api/Mahasiswa, POST /api/Mahasiswa, GET /api/Mahasiswa/{id}, and DELETE /api/Mahasiswa/{id}. The 'WeatherForecast' section lists one endpoint: GET /WeatherForecast. The UI also shows the API version as 1.0 and the OpenAPI specification version as 3.0.

➤ Penjelasan

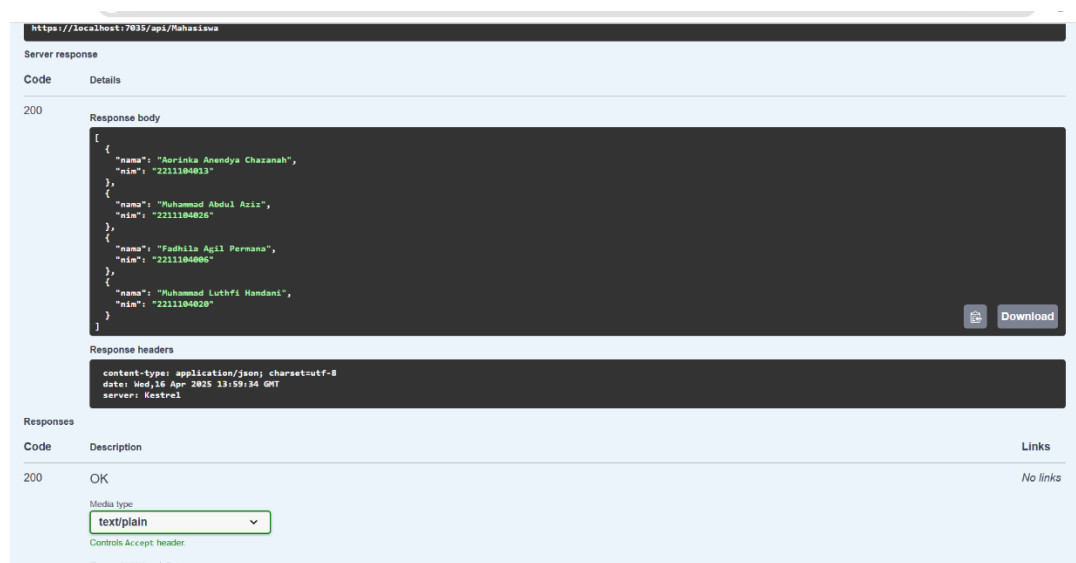
Program ini merupakan implementasi Web API sederhana dengan ASP.NET Core yang berfungsi untuk mengelola data mahasiswa. Data disimpan dalam sebuah list statis bernama `listMahasiswa`, yang berisi informasi nama dan NIM beberapa mahasiswa. Pengelolaan permintaan HTTP dilakukan melalui sebuah controller bernama `MahasiswaController`.

Controller ini menyediakan beberapa endpoint untuk melayani permintaan pengguna, seperti GET untuk menampilkan seluruh data mahasiswa atau data mahasiswa tertentu berdasarkan indeks, POST untuk menambahkan data baru, dan DELETE untuk menghapus data berdasarkan indeks. Setiap endpoint diberi atribut seperti `[HttpGet]`, `[HttpPost]`, dan `[HttpDelete]` untuk menyesuaikan dengan jenis permintaan yang diterima. Seluruh data disimpan secara sementara dalam memori aplikasi, sehingga data akan hilang ketika aplikasi dihentikan atau dijalankan ulang karena tidak menggunakan database permanen.

2. MENDEMONSTRASI WEB API

Beberapa skenario yang harus dicoba untuk memastikan jika program telah berjalan dengan baik. Buatlah dokumen yang berisi semua screenshot dari hasil uji coba skenario yang disebutkan pada list berikut ini:

A. Mencoba “GET /api/mahasiswa” saat baru dijalankan (mengeluarkan list nama mahasiswa dan nim anggota kelompok)



B. Menambahkan mahasiswa => Nama: “John Doe” dan NIM: “1302199999” dengan “POST /api/mahasiswa”

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** /api/mahasiswa
- Parameters:** No parameters
- Request body:**

```
{  "nama": "John Doe",  "nim": "1302199999"}
```
- Execute button:** A blue button labeled "Execute".
- Responses:**
 - Code:** 200
 - Response headers:**

```
content-length: 0
date: Wed, 16 Apr 2025 14:02:19 GMT
server: Kestrel
```
 - Response body:** (Empty)
 - Table:**

Code	Description	Links
200	OK	No links

C. Cek list/array dari semua mahasiswa lagi dengan “GET /api/mahasiswa”, pastikan mahasiswa yang baru ditambahkan sebelumnya ada di list mahasiswa:

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** /api/mahasiswa
- Execute button:** A blue button labeled "Execute".
- Responses:**
 - Code:** 200
 - Response body:**

```
{  "nama": "Auriska Annadya Chazannah",  "nim": "2211104013",  "nama": "Muhammad Abdul Aziz",  "nim": "2211104026",  "nama": "Tadhila Agil Permana",  "nim": "2211104006",  "nama": "Muhammad Isthfi Hamdani",  "nim": "2211104020",  "nama": "John Doe",  "nim": "1302199999"}
```
 - Response headers:**

```
content-type: application/json; charset=utf-8
date: Wed, 16 Apr 2025 14:02:19 GMT
server: Kestrel
```
 - Table:**

Code	Description	Links
200	OK	No links

D. Mencoba meminta mahasiswa dengan index 0, “GET /api/mahasiswa/0” yang seharusnya mengeluarkan nama dan nim anda:

The screenshot shows a REST client interface with a GET request to `/api/mahasiswa/{id}`. The parameter `id` is set to `0`. The response is a 200 status code with a JSON body containing student information.

Parameters

Name	Description
<code>id</code> * required integer(\$int32) (path)	0

Execute **Clear**

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7035/api/mahasiswa/0' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7035/api/mahasiswa/0
```

Server response

Code **Details**

200

Response body

```
{
  "nama": "Aorinka Azenadya Chazanah",
  "nim": "2211104013"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Wed, 16 Apr 2025 14:03:56 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

E. Menghapus objek mahasiswa dengan index ke-0 dengan “DELETE /api/mahasiswa/0”

The screenshot shows a REST client interface with a DELETE request to `/api/mahasiswa/{id}`. The parameter `id` is set to `0`. The response is a 200 status code with a response body indicating the deletion was successful.

DELETE `/api/mahasiswa/{id}`

Parameters

Name	Description
<code>id</code> * required integer(\$int32) (path)	0

Execute **Clear**

Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7035/api/mahasiswa/0' \
  -H 'accept: */*'
```

Request URL

```
https://localhost:7035/api/mahasiswa/0
```

Server response

Code **Details**

200

Response headers

```
content-length: 0
date: Wed, 16 Apr 2025 14:04:32 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	OK	No links

WeatherForecast

- F. Cek list/array dari semua mahasiswa sekali lagi dengan “GET /api/mahasiswa”, pastikan nama anda sudah tidak muncul di list tersebut:

The screenshot shows a REST client interface with the following sections:

- GET /api/mahasiswa**: The endpoint and method.
- Parameters**: A section with a "Cancel" button and the text "No parameters".
- Execute**: A blue button to execute the request.
- Clear**: A button to clear the request.
- Responses**: A section containing:
 - Curl**: A text box with the command:

```
curl -X 'GET' \
  'https://localhost:7055/api/mahasiswa' \
  -H 'accept: text/plain'
```
 - Request URL**: A text box with the URL: `https://localhost:7055/api/mahasiswa`
 - Server response**: A section with a "Code" tab and a "Details" tab.
 - 200**: The status code of the response.
 - Response body**: A text box showing the JSON response:

```
{
  "name": "Fadhila Agil Permana",
  "nia": "2211104080",
},
{
  "name": "Muhammad Luthfi Handani",
  "nia": "22111040820",
},
{
  "name": "Jahn Doe",
  "nia": "1302199999"
}
```
 - Response headers**: A text box showing the headers:

```
content-type: application/json; charset=utf-8
date: Wed, 16 Apr 2023 14:04:46 GMT
server: Kestrel
```