

Nama : Prajna Paramitha Wardhany  
Kelas : SE 07 01  
NIM : 2311104016

## JURNAL MODUL 9

### 1. MEMBUAT PROJECT WEB API

Berhubung cara membuat project web api berbeda-beda untuk setiap bahasa pemrograman, langkah-langkah berikut hanya berlaku apabila dilakukan dengan menggunakan .NET dan Visual Studio. Untuk IDE dan bahasa pemrograman lain, yang terpenting adalah nama project yang dibuat yaitu “modul8\_NIM”.

- A. Buka visual studio yang sudah terinstall dengan ASP.NET dan .NET 5.0 SDK atau setelahnya
- B. Pilih New Project dan kemudian pilih ASP.NET Core Web API atau API (pastikan opsi ‘Enable OpenAPI support’ tercentang).
- C. Pastikan untuk memilih .NET versi 5.0 atau yang lebih baru.
- D. Masukkan nama projek “modul9\_NIM”.
- E. Langkah-langkah yang disertai gambar dapat dilihat pada link berikut ini (cukup dilihat pada bagian “Create a Web API project”):  
<https://docs.microsoft.com/en-us/aspnet/core/tutorials/min-web-api?view=aspnetcore-6.0&tabs=visual-studio>
- F. Setelah project tersebut selesai dibuat, coba run programnya, dan tunggu sampai program selesai di-compile.

### 2. MELAKUKAN GIT COMMIT PADA PROJECT YANG DIBUAT

Task atau langkah-langkah yang perlu dikerjakan adalah sebagai berikut:

- A. Buatlah github public repository kosong (pastikan bagian “Initialize this repository with” tidak ada yang dicentang pada saat membuat repository baru) melalui <https://github.com/>
- B. Melakukan inisialisasi git repository di folder project yang dibuat.
- C. Pastikan untuk menambahkan file “.gitignore” baik manual atau dengan menggunakan visual studio/IDE. Untuk project dengan C# dapat melihat referensi file “.gitignore” pada link berikut ini:  
<https://github.com/github/gitignore/blob/main/VisualStudio.gitignore>
- D. Membuat commit untuk versi pertama dari project yang dibuat dengan pesan commit bebas.
- E. Melakukan git push ke github repo.

### 3. 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
+ Name : string
+ Nim : string
+ Course : List<string>
+ Year: integer
+ Mahasiswa()

- B. API yang dibuat mempunyai lokasi sebagai berikut **‘/api/Mahasiswa**, URL domain boleh dari port mana saja (port bebas). Dengan menggunakan swagger API tersebut dapat menerima RESTful API dengan metoda sebagai berikut (halaman swagger dapat diakses pada <https://localhost:<PORT>/swagger/index.html>):

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

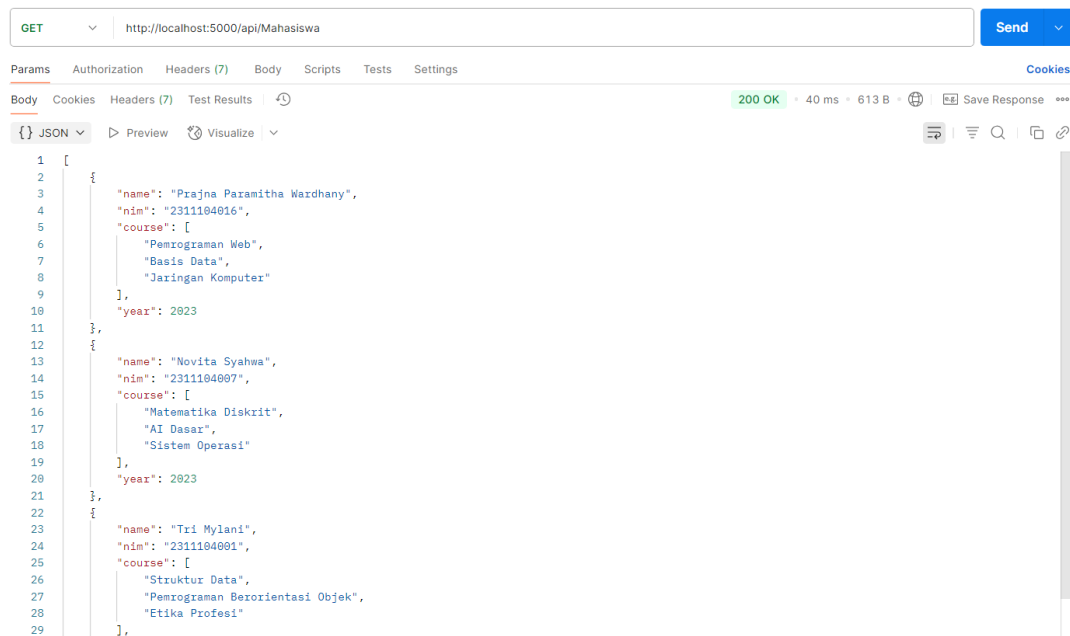
- i. GET /api/Mahasiswa: mengembalikan output berupa list/array dari semua objek Mahasiswa
  - ii. GET /api/Mahasiswa/{id}: mengembalikan output berupa objek Mahasiswa untuk index “id”
  - iii. POST /api/Mahasiswa: menambahkan objek Mahasiswa baru
  - iv. DELETE /api/Mahasiswa/{id}: menghapus objek Mahasiswa pada index “id”
- C. Secara default, program yang dibuat memiliki list Mahasiswa yang berasal dari anggota kelompok TUBES (minimal 3 data).
- D. 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.

- E. Dalam pembuatan program/aplikasi ini, anda dapat mengasumsikan bahwa input dari user selalu benar dan sesuai dengan tipe data yang diharapkan.

#### 4. MENDEMONSTRASI WEB API

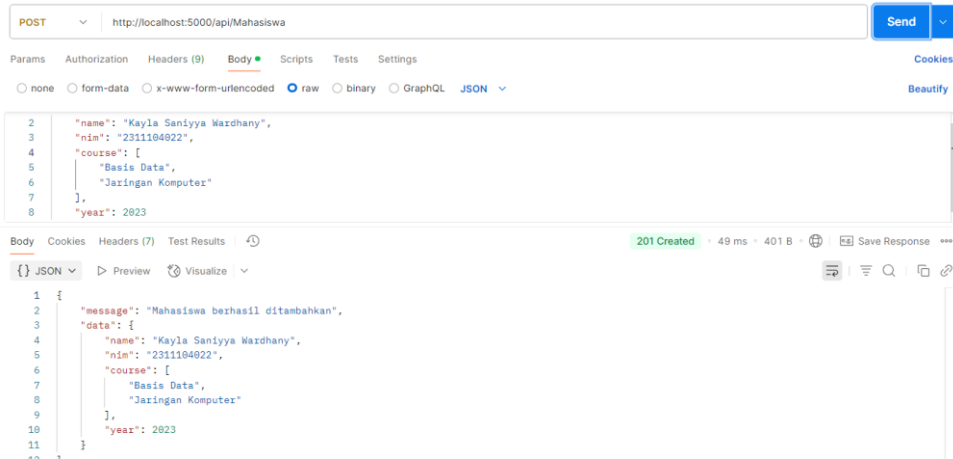
Beberapa skenario yang harus dicoba untuk memastikan jika program telah berjalan dengan baik. Buatlah dokumen yang berisi semua screenshot dari hasil uji coba scenario yang disebutkan pada list berikut ini:

- A. Mencoba “GET /api/Mahasiswa” saat baru dijalankan yang mengeluarkan list film dari TOP 3 IMDB seperti pada tampilan berikut pada saat dicoba dengan menekan tombol “Try it out” dan tombol “Execute”

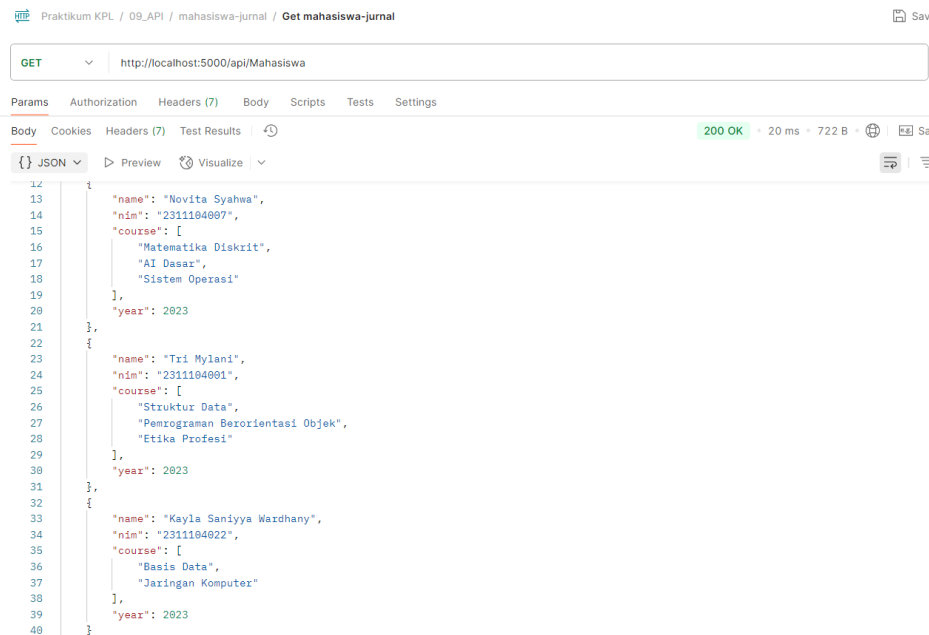


```
1  [
2    {
3      "name": "Prajna Paramitha Wardhany",
4      "nim": "2311104016",
5      "course": [
6        "Pemrograman Web",
7        "Basis Data",
8        "Jaringan Komputer"
9      ],
10     "year": 2023
11   },
12   {
13     "name": "Novita Syahwa",
14     "nim": "2311104007",
15     "course": [
16       "Matematika Diskrit",
17       "AI Dasar",
18       "Sistem Operasi"
19     ],
20     "year": 2023
21   },
22   {
23     "name": "Tri Mylani",
24     "nim": "2311104001",
25     "course": [
26       "Struktur Data",
27       "Pemrograman Berorientasi Objek",
28       "Etika Profesi"
29     ],
30     "year": 2023
31   }
32 ]
```

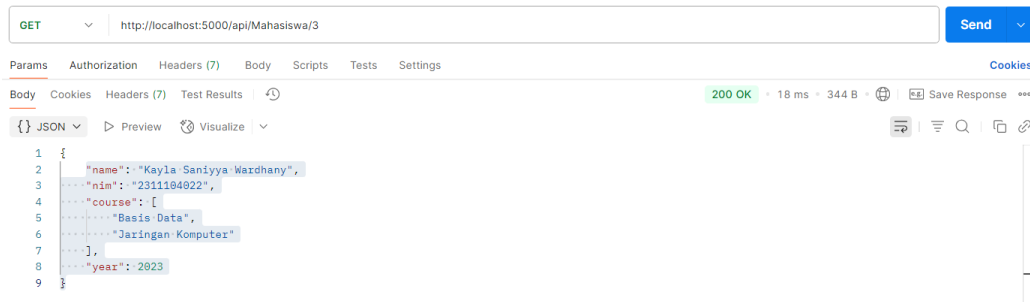
- B. Menambahkan Mahasiswa baru yaitu urutan ke-4 pada TOP IMDB list dengan memanggil API pada bagian “POST /api/Mahasiswa”



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



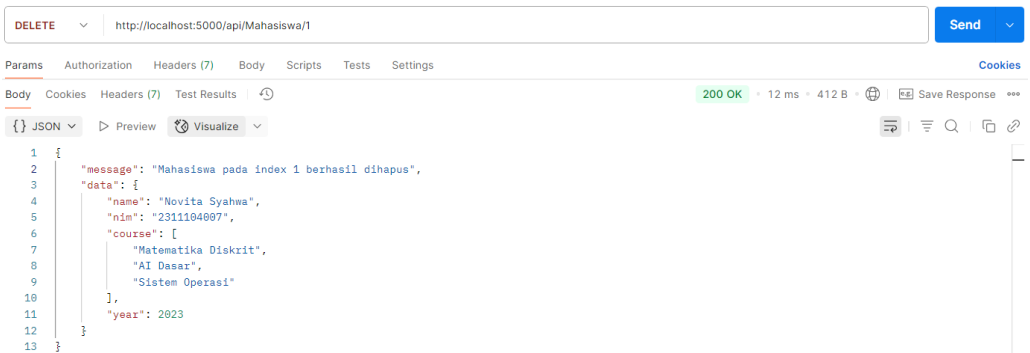
- D. Mencoba meminta Mahasiswa dengan index 3, “GET /api/Mahasiswa/3” yang seharusnya mengembalikan Mahasiswa yang baru saja ditambah:



```

1 {
2   "name": "Kayla Saniyya Wardhany",
3   "nim": "2311104022",
4   "course": [
5     "Basis Data",
6     "Jaringan Komputer"
7   ],
8   "year": 2023
9 }
  
```

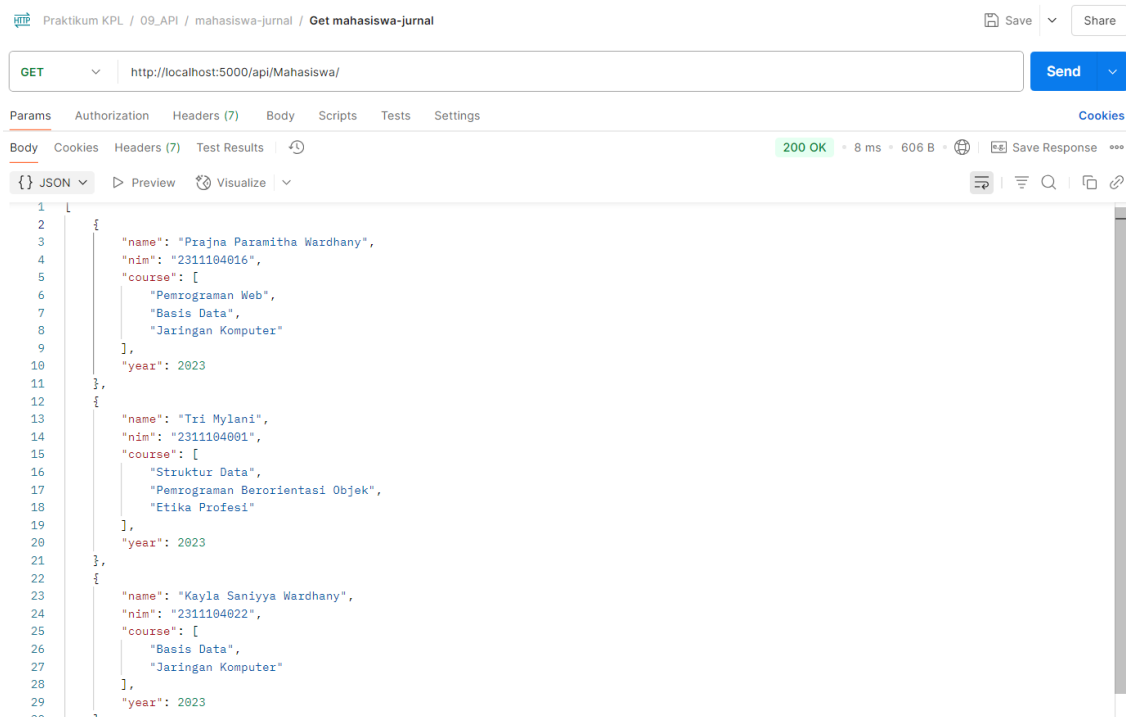
- E. Menghapus objek Mahasiswa dengan index ke-1 dengan “DELETE /api/Mahasiswa/1”



```

1 {
2   "message": "Mahasiswa pada index 1 berhasil dihapus",
3   "data": {
4     "name": "Novita Syahma",
5     "nim": "2311104007",
6     "course": [
7       "Matematika Diskrit",
8       "AI Dasar",
9       "Sistem Operasi"
10    ],
11    "year": 2023
12  }
13 }
  
```

- F. Cek list/array dari semua Mahasiswa sekali lagi dengan “GET /api/Mahasiswa”, film dengan ranking kedua “Godfather” sudah tidak ada di list:



```

1 [
2   {
3     "name": "Prajna Paramitha Wardhany",
4     "nim": "2311104016",
5     "course": [
6       "Pemrograman Web",
7       "Basis Data",
8       "Jaringan Komputer"
9     ],
10    "year": 2023
11  },
12  {
13    "name": "Tri Mylani",
14    "nim": "2311104001",
15    "course": [
16      "Struktur Data",
17      "Pemrograman Berorientasi Objek",
18      "Etika Profesi"
19    ],
20    "year": 2023
21  },
22  {
23    "name": "Kayla Saniyya Wardhany",
24    "nim": "2311104022",
25    "course": [
26      "Basis Data",
27      "Jaringan Komputer"
28    ],
29    "year": 2023
30  }
31 ]
  
```

## Kodingan json

```
[
  {
    "name": "Prajna Paramitha Wardhany",
    "nim": "2311104016",
    "course": ["Pemrograman Web", "Basis Data", "Jaringan Komputer"],
    "year": 2023
  },
  {
    "name": "Novita Syahwa",
    "nim": "2311104007",
    "course": ["Matematika Diskrit", "AI Dasar", "Sistem Operasi"],
    "year": 2023
  },
  {
    "name": "Tri Mylani",
    "nim": "2311104001",
    "course": ["Struktur Data", "Pemrograman Berorientasi Objek", "Etika Profesi"],
    "year": 2023
  }
]
```

## Kodingan index.js

```
const express = require('express'); // Menggunakan Express framework
const fs = require('fs');
const app = express();
const PORT = 5000;

app.use(express.json()); // Middleware untuk parsing request body format JSON
let mahasiswaList = JSON.parse(fs.readFileSync('./mahasiswa.json'));

// GET /api/Mahasiswa - Mengembalikan semua mahasiswa
app.get('/api/Mahasiswa', (req, res) => {
  res.json(mahasiswaList); // Kirim semua data mahasiswa
});

// GET /api/Mahasiswa/:id - Mengembalikan mahasiswa berdasarkan index
app.get('/api/Mahasiswa/:id', (req, res) => {
  const index = parseInt(req.params.id);
  if (index < 0 || index >= mahasiswaList.length) {
    return res.status(404).json({ message: 'Mahasiswa tidak ditemukan' });
  }
  res.json(mahasiswaList[index]); // Kirim data mahasiswa pada index
});

// POST /api/Mahasiswa - Menambahkan mahasiswa baru
app.post('/api/Mahasiswa', (req, res) => {
  const { name, nim, course, year } = req.body; // Ambil data dari request body
  mahasiswaList.push({ name, nim, course, year }); // Tambah ke list
  res.status(201).json({ message: 'Mahasiswa berhasil ditambahkan', data: { name, nim, course, year } });
});

// DELETE /api/Mahasiswa/:id - Menghapus mahasiswa berdasarkan index
app.delete('/api/Mahasiswa/:id', (req, res) => {
  const index = parseInt(req.params.id);
  if (index < 0 || index >= mahasiswaList.length) {
    return res.status(404).json({ message: 'Index tidak ditemukan' });
  }
  const deleted = mahasiswaList.splice(index, 1); // Hapus data dari list
  res.json({ message: `Mahasiswa pada index ${index} berhasil dihapus`, data: deleted[0] });
});

// Jalankan server & info port
app.listen(PORT, () => {
  console.log(`Server berjalan di http://localhost:${PORT}/api/Mahasiswa`);
});
```

\*Kak penjelasannya didalam kodingannya yaa hehe

---

## 5. MELAKUKAN COMMIT

Pada branch master/main:

- A. Lakukan commit dengan pesan “menambahkan API Mahasiswa”.
- B. Lakukan push ke github ke branch yang dibuat di bagian sebelumnya.

## 6. PENGUMPULAN FILE/TUGAS JURNAL

Sebelum pengumpulan, **praktikan wajib menunjukkan hasil run via share screen ke asprak**. Kumpulkan semua file berikut dalam bentuk file zip/rar/7zip:

- A. Source code dari project yang dibuat
- B. File docx/pdf yang berisi:
  - i. Link github repository kelompok
  - ii. Screenshot hasil run (hasil console output dari proses run sesuai bagian 4)

- iii. Penjelasan singkat dari kode implementasi yang dibuat (beserta screenshot dari potongan source code yang dijelaskan).

*Catatan: Tidak ada file docx/pdf (screenshot dan penjelasan) yang dikumpulkan DAN juga tidak melakukan demo/menunjukkan hasil run ke asprak maka nilai jurnal akan 0*

#### KOMPONEN PENILAIAN

- A. Penggunaan Git dan Github [10 pts]
- B. Implementasi Web API [45 pts]
- C. Demo program/aplikasi [30 pts]
- D. Laporan jurnal [15 pts]