

**LAPORAN PRAKTIKUM  
KONSTRUKSI PERANGKAT LUNAK**

**MODUL IX  
REST API NODE.JS**



**Disusun Oleh :  
Satria Ariq Adelard Dompas  
S1SE-06-02**

**Asisten Praktikum :  
Muhamad Taufiq Hidayat**

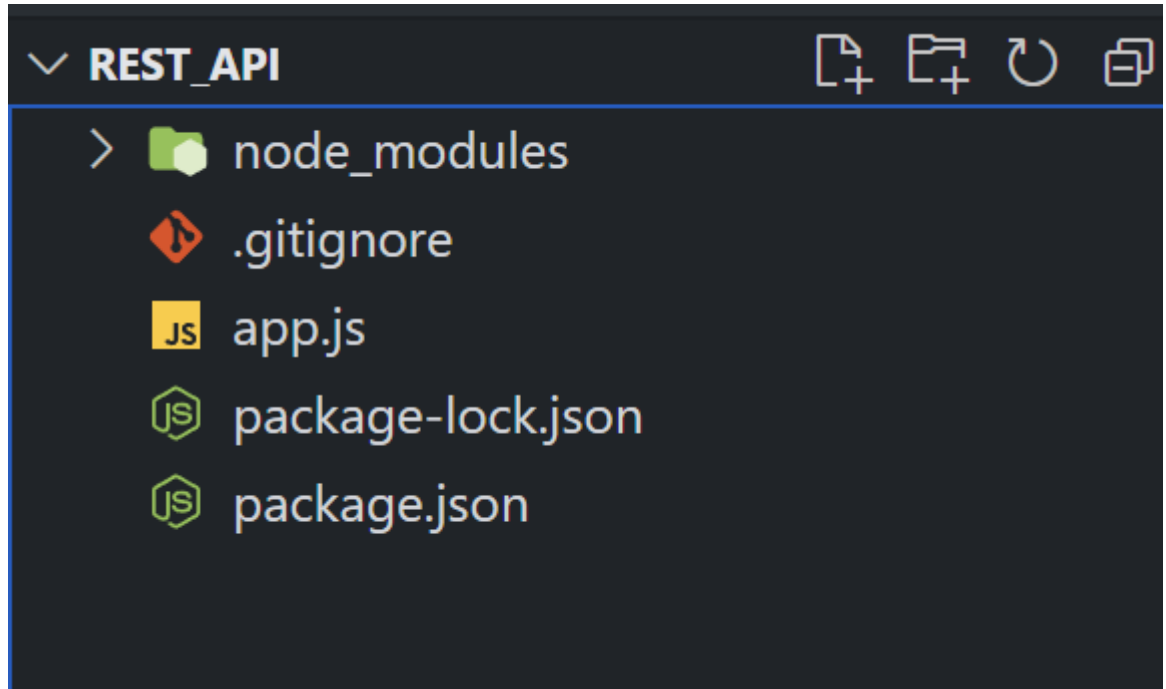
**Dosen Pengampu :  
Riyan Dwi Yulian Prakoso, S.Kom., M.Kom.**

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK  
DIREKTORAT TELKOM KAMPUS PURWOKERTO  
2025**

## BAB I GUIDED

Hasil Program:

1. Inisialisasi Struktur Folder



2. app.js



```
1  const express = require('express');
2  const app = express();
3  const port = 3000;
4
5  app.use(express.json());
6
7  // Simpan data mahasiswa di array static
8  let mahasiswa = [
9    { nama: "Muhamad Taufiq Hidayat", nim: "21102206" },
10   { nama: "Febrilia Ananda", nim: "220220106" },
11   { nama: "LeBron James", nim: "1302000003" }
12 ];
13
14 // GET semua mahasiswa
15 app.get('/api/mahasiswa', (req, res) => {
16   res.json(mahasiswa);
17 });
18
19 // GET mahasiswa berdasarkan index
20 app.get('/api/mahasiswa/:index', (req, res) => {
21   const index = parseInt(req.params.index);
22   if (index >= 0 && index < mahasiswa.length) {
23     res.json(mahasiswa[index]);
24   } else {
25     res.status(404).json({ message: 'Data tidak ditemukan' });
26   }
27 });
28
29 // POST mahasiswa baru
30 app.post('/api/mahasiswa', (req, res) => {
31   const { nama, nim } = req.body;
32   mahasiswa.push({ nama, nim });
33   res.status(201).json({ message: 'Data berhasil ditambahkan' });
34 });
35
36 // DELETE mahasiswa berdasarkan index
37 app.delete('/api/mahasiswa/:index', (req, res) => {
38   const index = parseInt(req.params.index);
39   if (index >= 0 && index < mahasiswa.length) {
40     mahasiswa.splice(index, 1);
41     res.json({ message: 'Data berhasil dihapus' });
42   } else {
43     res.status(404).json({ message: 'Data tidak ditemukan' });
44   }
45 });
46
47 app.listen(port, () => {
48   console.log(`Server berjalan di http://localhost:${port}`);
49 });
```

Contoh output:

Pretty-print ☒

```
[
  {
    "nama": "Muhamad Taufiq Hidayat",
    "nim": "21102206"
  },
  {
    "nama": "Febrilia Ananda",
    "nim": "220220106"
  },
  {
    "nama": "LeBron James",
    "nim": "1302000003"
  }
]
```

Penjelasan Kode:

Kode di atas merupakan implementasi API sederhana menggunakan Express.js untuk mengelola data mahasiswa yang disimpan dalam array statis. API ini mendukung operasi dasar seperti membaca semua data mahasiswa (GET /api/mahasiswa), membaca satu mahasiswa berdasarkan index (GET /api/mahasiswa/:index), menambah data mahasiswa baru dengan mengirim JSON lewat body (POST /api/mahasiswa), serta menghapus data mahasiswa berdasarkan index (DELETE /api/mahasiswa/:index). Middleware `express.json()` digunakan agar server dapat membaca data JSON dari body permintaan. Server berjalan pada port 3000, dan saat dijalankan akan menampilkan pesan bahwa server aktif di <http://localhost:3000>.

## BAB II

### UNGUIDED

Dokumentasi REST API dengan Postman

1.app.js

```
const express = require("express");
const app = express();
const port = 3000;

app.use(express.json());

// Data awal mahasiswa
let mahasiswa = [
  { nama: "Satria Ariq Adelard Dompas", nim: "2211104033" },
  { nama: "Dwi Candra", nim: "2211104035" },
  { nama: "Ade Fatkhul", nim: "2211104066" },
];

// GET semua mahasiswa
app.get("/api/mahasiswa", (req, res) => {
  res.json(mahasiswa);
});

// GET mahasiswa berdasarkan index
app.get("/api/mahasiswa/:index", (req, res) => {
  const index = parseInt(req.params.index);
  if (index >= 0 && index < mahasiswa.length) {
    res.json(mahasiswa[index]);
  } else {
    res.status(404).json({ message: "Data tidak ditemukan" });
  }
});

// POST mahasiswa baru
app.post("/api/mahasiswa", (req, res) => {
  const { nama, nim } = req.body;
  mahasiswa.push({ nama, nim });
  res.status(201).json({ message: "Data berhasil ditambahkan" });
});

// DELETE mahasiswa berdasarkan index
app.delete("/api/mahasiswa/:index", (req, res) => {
  const index = parseInt(req.params.index);
  if (index >= 0 && index < mahasiswa.length) {
    mahasiswa.splice(index, 1);
    res.json({ message: "Data berhasil dihapus" });
  } else {
```

```

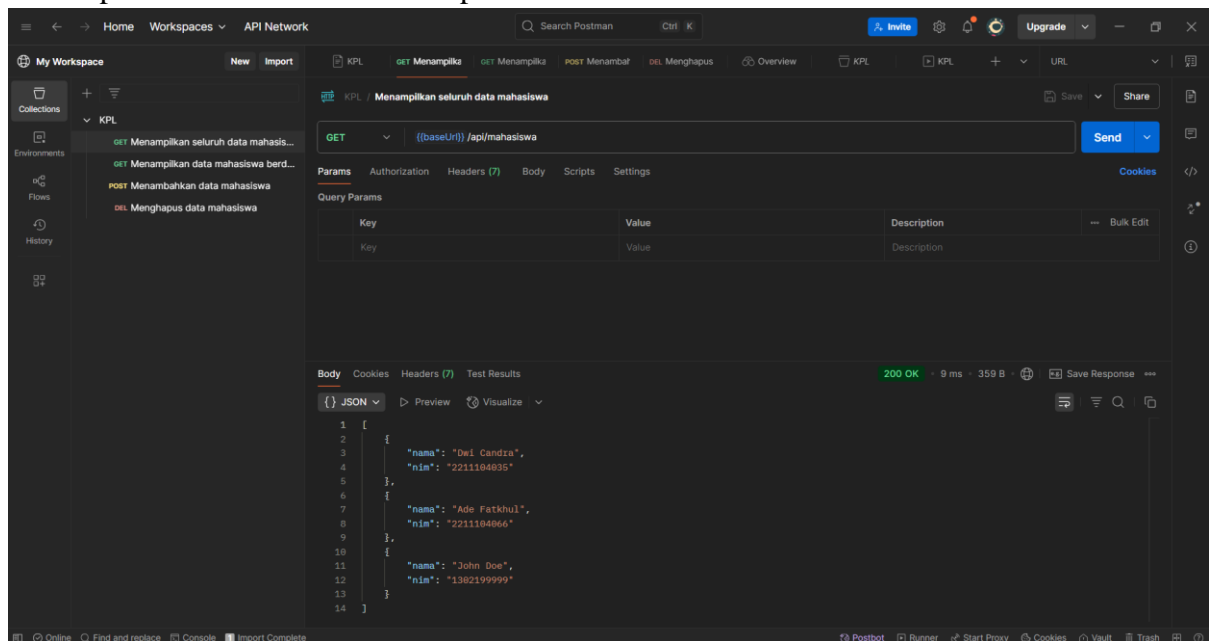
    res.status(404).json({ message: "Data tidak ditemukan" });
  }
});

app.listen(port, () => {
  console.log(`Server berjalan di http://localhost:${port}`);
});

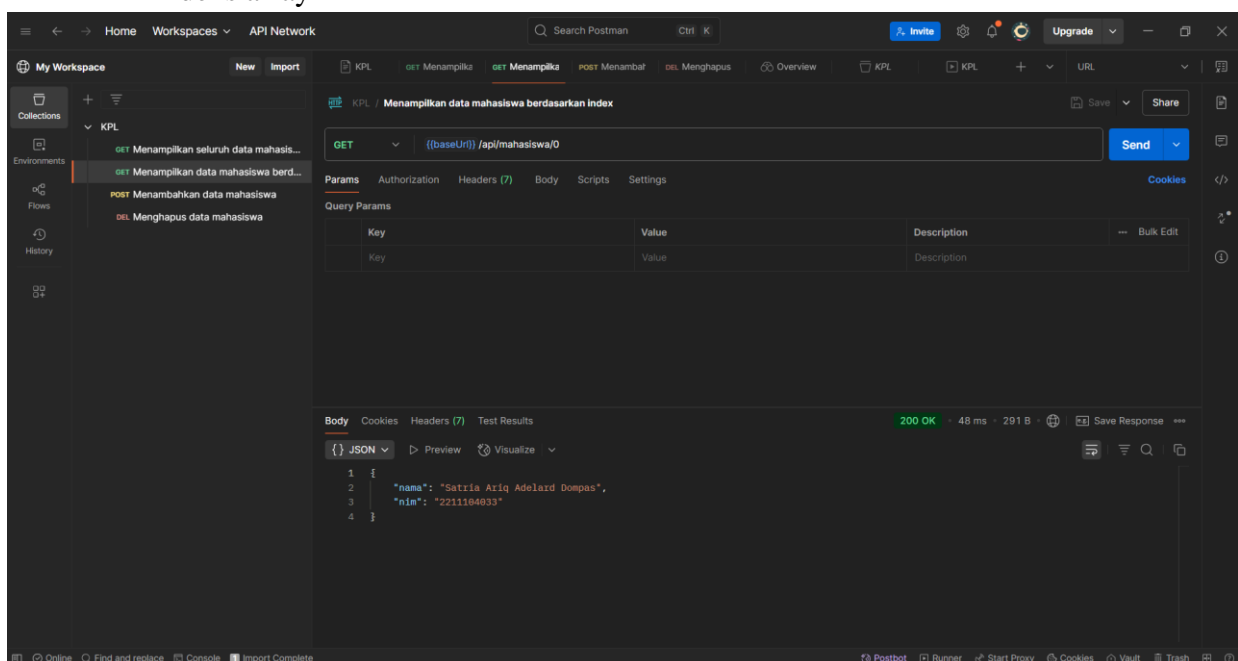
```

## 2. Dokumentasi Postman

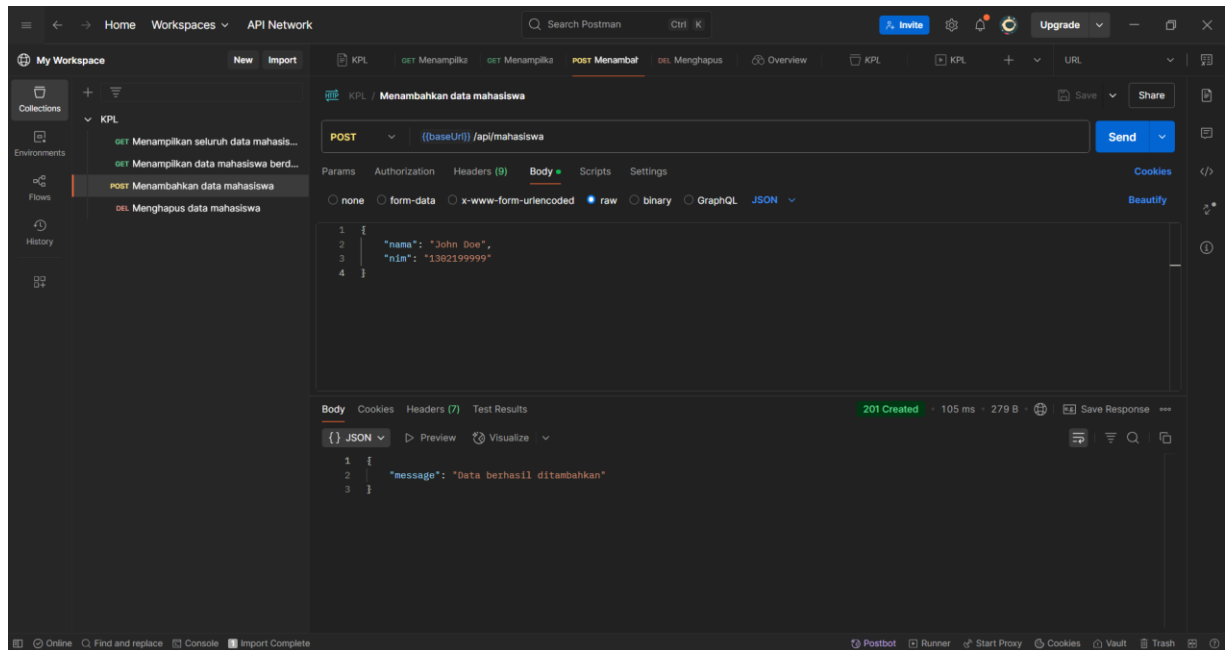
### a. GET /api/mahasiswa untuk Menampilkan seluruh data mahasiswa



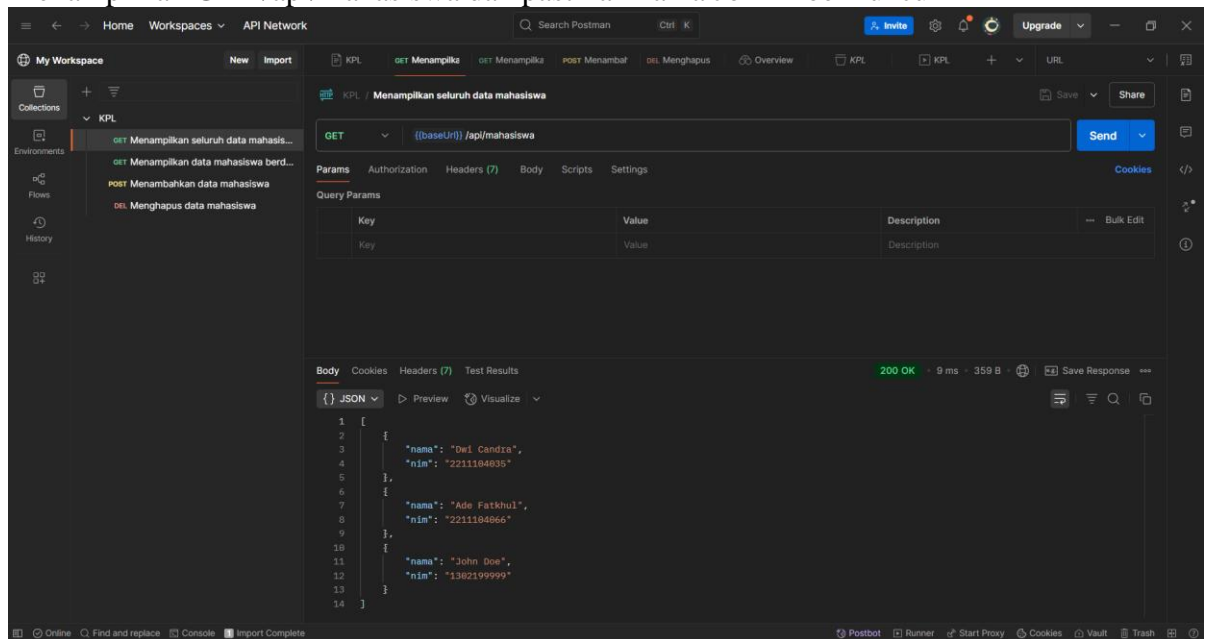
### b. GET /api/mahasiswa/:index untuk Menampilkan data mahasiswa berdasarkan indeks array



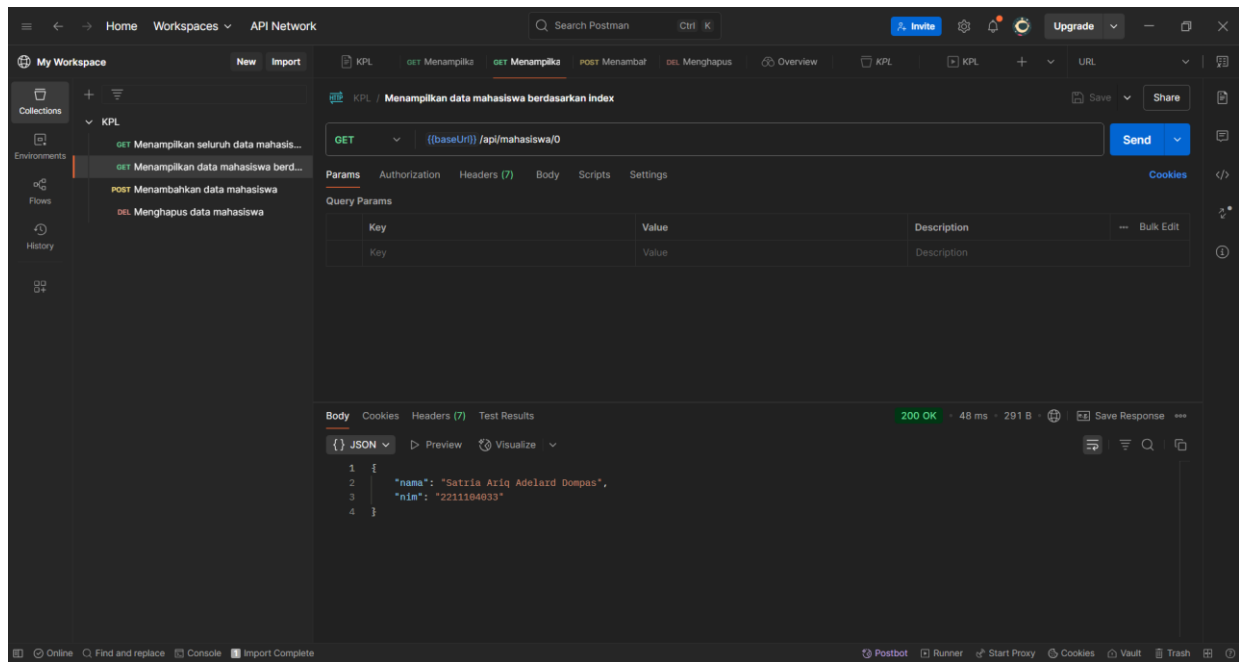
- c. Menjalankan POST /api/mahasiswa untuk menambahkan John Doe – 1302199999



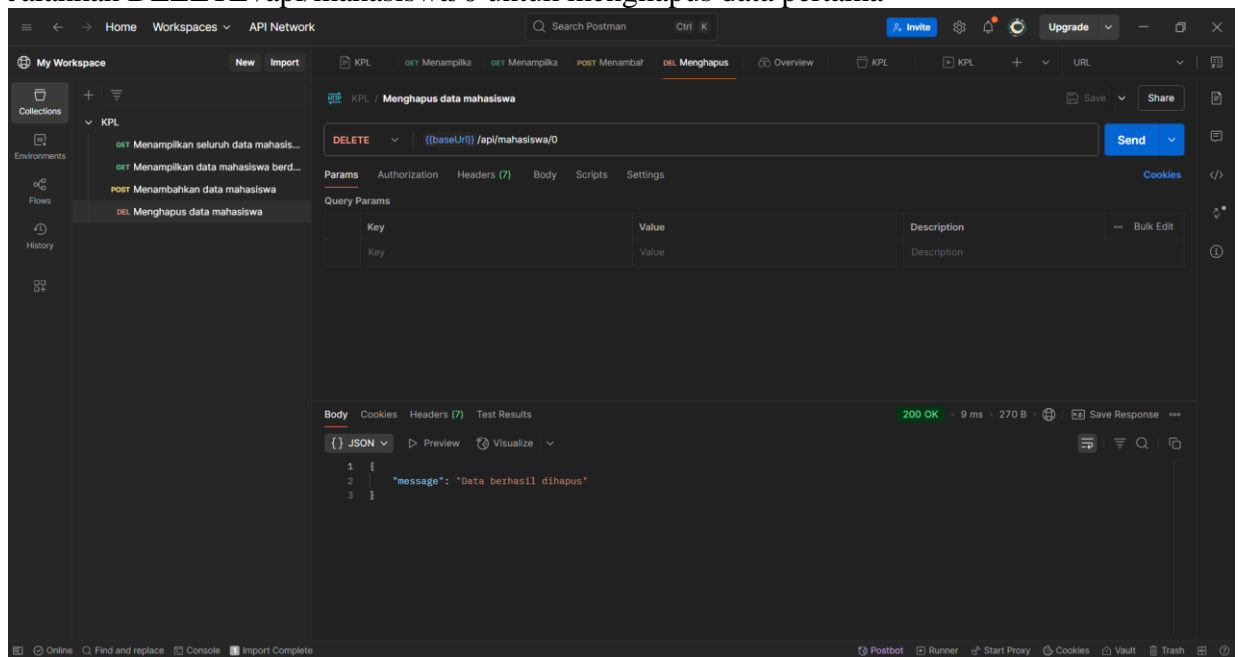
- d. Menampilkan GET /api/mahasiswa dan pastikan nama John Doe muncul



- e. Jalankan GET /api/mahasiswa/0 dan pastikan data pertama (nama Anda) muncul



f. Jalankan DELETE /api/mahasiswa/0 untuk menghapus data pertama





g. Jalankan GET /api/mahasiswa lagi dan pastikan data Anda sudah hilang

The screenshot shows the Postman interface with a GET request to `((baseUri)) /api/mahasiswa`. The response is a 200 OK status with a response time of 9 ms and a body size of 359 B. The response body is a JSON array containing three student records.

Key	Value	Description
Key	Value	Description

```
1 {
2   {
3     "nama": "Dwi Candia",
4     "nim": "2211184635"
5   },
6   {
7     "nama": "Ade Fatkhul",
8     "nim": "2211184666"
9   },
10  {
11    "nama": "John Doe",
12    "nim": "1362199999"
13  }
14 }
```