Tugas Pendahuluan Modul 09

2311104008
Viona Aziz Syahputri

Link Gitbut
https://github.com/viona123/KPL_Viona-Aziz-Syahputri_2311104008_SE07-01/tree/main/09_API/TP_GUI_2311104008

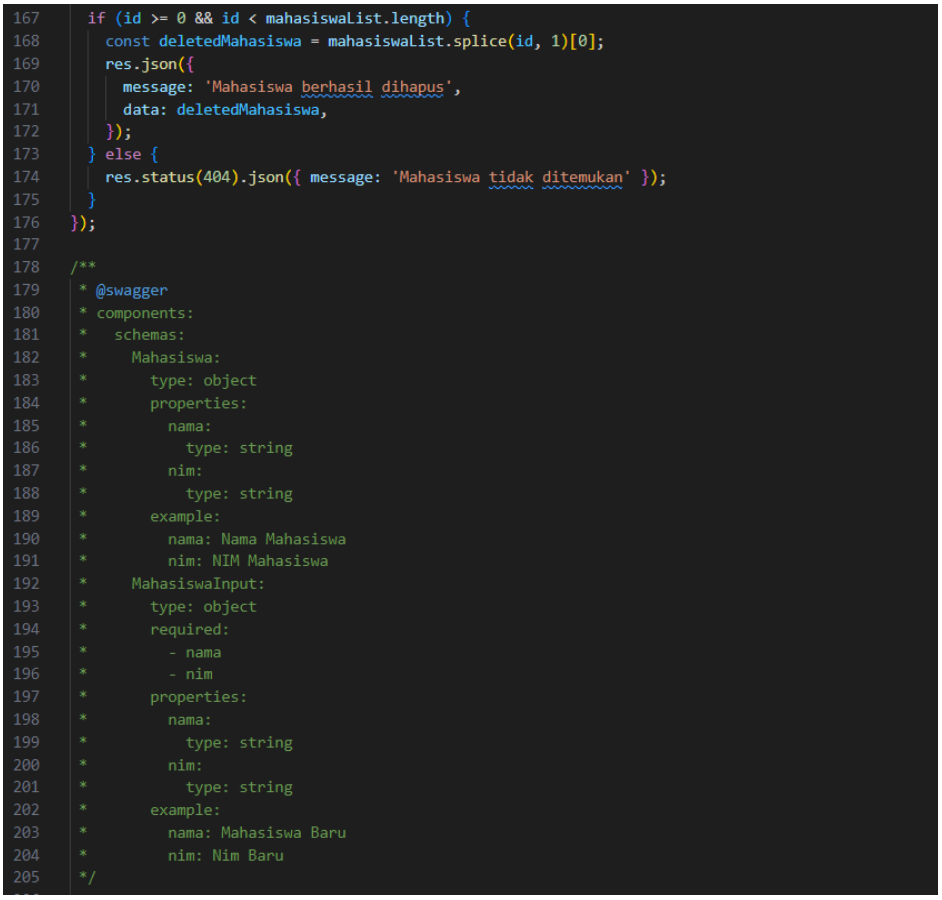
```
const express = require('express');
const swaggerJsdoc = require('swagger-jsdoc');
const swaggerUi = require('swagger-ui-express');
const app = express();
const port = 8000;

// Middleware untuk parsing JSON
app.use(express.json());

// Data Mahasiswa default
let mahasiswaList = [
  { nama: 'Aisyah Putri Ramadhani', nim: '2311104030' },
  { nama: 'Bima Satria Nugraha', nim: '2311104031' },
  { nama: 'Citra Ayu Lestari', nim: '2311104032' },
  { nama: 'Daffa Arsyad Hermawan', nim: '2311104033' },
  { nama: 'Elsa Fitriani', nim: '2311104034' },
  { nama: 'Fajar Setiawan', nim: '2311104035' },
];

// Swagger configuration
const options = {
  definition: {
    openapi: '3.0.0',
    info: {
      title: 'API Data Mahasiswa',
      version: '1.0.0',
      description: 'API untuk manajemen data mahasiswa',
    },
    servers: [
      {
        url: `http://localhost:${port}`,
        description: 'server',
      },
    ],
  },
  apis: ['./mahasiswa.js'],
};

```

```
const specs = swaggerJsdoc(options);

/**
 * @swagger
 * tags:
 *   name: Mahasiswa
 *   description: Manajemen data mahasiswa
 */

/**
 * @swagger
 * /api/mahasiswa:
 *   get:
 *     summary: Mendapatkan daftar semua mahasiswa
 *     tags: [Mahasiswa]
 *     responses:
 *       200:
 *         description: Sukses mendapatkan data mahasiswa
 *         content:
 *           application/json:
 *             schema:
 *               type: array
 *               items:
 *                 $ref: '#/components/schemas/Mahasiswa'
 */
app.get('/api/mahasiswa', (req, res) => {
  res.json(mahasiswaList);
});

/**
 * @swagger
 * /api/mahasiswa/{id}:
 *   get:
 *     summary: Mendapatkan data mahasiswa berdasarkan id
 *     tags: [Mahasiswa]
 *     parameters:
 *       - in: path
 *         name: id
 *         required: true
 *         schema:
 *           type: integer
 *         description: Id mahasiswa (dimulai dari 0)
 *     responses:
```

```
const specs = swaggerJsdoc(options);

/**
 * @swagger
 * tags:
 *   name: Mahasiswa
 *   description: Manajemen data mahasiswa
 */

/**
 * @swagger
 * /api/mahasiswa:
 *   get:
 *     summary: Mendapatkan daftar semua mahasiswa
 *     tags: [Mahasiswa]
 *     responses:
 *       200:
 *         description: Sukses mendapatkan data mahasiswa
 *         content:
 *           application/json:
 *             schema:
 *               type: array
 *               items:
 *                 $ref: '#/components/schemas/Mahasiswa'
 */
app.get('/api/mahasiswa', (req, res) => {
  res.json(mahasiswaList);
});

/**
 * @swagger
 * /api/mahasiswa/{id}:
 *   get:
 *     summary: Mendapatkan data mahasiswa berdasarkan id
 *     tags: [Mahasiswa]
 *     parameters:
 *       - in: path
 *         name: id
 *         required: true
 *         schema:
 *           type: integer
 *         description: Id mahasiswa (dimulai dari 0)
 *     responses:
```

```
 *       200:
 *         description: Sukses mendapatkan data mahasiswa
 *         content:
 *           application/json:
 *             schema:
 *               $ref: '#/components/schemas/Mahasiswa'
 *       404:
 *         description: Mahasiswa tidak ditemukan
 */
app.get('/api/mahasiswa/:id', (req, res) => {
  const id = parseInt(req.params.id);

  if (isNaN(id)) {
    return res.status(400).json({ message: 'ID harus berupa angka' });
  }

  if (id >= 0 && id < mahasiswaList.length) {
    res.json(mahasiswaList[id]);
  } else {
    res.status(404).json({ message: 'Mahasiswa tidak ditemukan' });
  }
});

/**
 * @swagger
 * /api/mahasiswa:
 *   post:
 *     summary: Menambahkan data mahasiswa baru
 *     tags: [Mahasiswa]
 *     requestBody:
 *       required: true
 *       content:
 *         application/json:
 *           schema:
 *             $ref: '#/components/schemas/MahasiswaInput'
 *     responses:
 *       201:
 *         description: Mahasiswa berhasil ditambahkan
 *         content:
 *           application/json:
 *             schema:
 *               $ref: '#/components/schemas/Mahasiswa'
 */
```

```
app.post('/api/mahasiswa', (req, res) => {
  const { nama, nim } = req.body;
  if (!nama || !nim) {
    return res.status(400).json({ message: 'Nama dan NIM harus diisi' });
  }
  const newMahasiswa = { nama, nim };
  mahasiswaList.push(newMahasiswa);
  res.status(201).json(newMahasiswa);
});

/**
 * @swagger
 * /api/mahasiswa/{id}:
 *   delete:
 *     summary: Menghapus data mahasiswa berdasarkan id
 *     tags: [Mahasiswa]
 *     parameters:
 *       - in: path
 *         name: id
 *         required: true
 *         schema:
 *           type: integer
 *         description: Id mahasiswa (dimulai dari 0)
 *     responses:
 *       200:
 *         description: Mahasiswa berhasil dihapus
 *         content:
 *           application/json:
 *             schema:
 *               $ref: '#/components/schemas/Mahasiswa'
 *       400:
 *         description: ID harus berupa angka
 *       404:
 *         description: Mahasiswa tidak ditemukan
 */
app.delete('/api/mahasiswa/:id', (req, res) => {
  const id = parseInt(req.params.id);

  if (isNaN(id)) {
    return res.status(400).json({ message: 'ID harus berupa angka' });
  }
```



```
  if (id >= 0 && id < mahasiswaList.length) {
    const deletedMahasiswa = mahasiswaList.splice(id, 1)[0];
    res.json({
      message: 'Mahasiswa berhasil dihapus',
      data: deletedMahasiswa,
    });
  } else {
    res.status(404).json({ message: 'Mahasiswa tidak ditemukan' });
  }
});

/**
 * @swagger
 * components:
 *   schemas:
 *     Mahasiswa:
 *       type: object
 *       properties:
 *         nama:
 *           type: string
 *         nim:
 *           type: string
 *       example:
 *         nama: Nama Mahasiswa
 *         nim: NIM Mahasiswa
 *     MahasiswaInput:
 *       type: object
 *       required:
 *         - nama
 *         - nim
 *       properties:
 *         nama:
 *           type: string
 *         nim:
 *           type: string
 *       example:
 *         nama: Mahasiswa Baru
 *         nim: Nim Baru
 */
```

Codingan ini sebenarnya adalah aplikasi sederhana pakai Express.js (alat bantu dari Node.js) buat ngatur data mahasiswa. Kamu bisa akses lewat alamat http://localhost:8000. Di dalamnya udah ada daftar mahasiswa lengkap dengan nama dan NIM-nya. Aplikasi ini bisa nunjukin semua data mahasiswa, nyari data mahasiswa berdasarkan ID, nambahin data baru, sampai hapus data juga bisa. Tapi datanya belum disimpan di database, masih disimpan sementara di dalam memori, jadi kalau server dimatikan, semua data akan hilang.

Nah, biar orang lain gampang ngerti cara pakai API ini, dibuatlah dokumentasi otomatis pakai Swagger. Dokumentasi ini bisa dibuka di browser lewat http://localhost:8000/api-docs, isinya nunjukin semua fitur API, cara ngaksesnya, dan contoh data yang dipakai. Intinya, program ini adalah contoh API simpel buat ngatur data mahasiswa, lengkap dengan dokumentasi biar gampang dipelajari dan dipakai.

```json
{
  "name": "tp_gui_2311104008",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "express": "^5.1.0",
    "swagger-jsdoc": "^6.2.8",
    "swagger-ui-express": "^5.0.1"
  }
}
```

file pengaturan utama dalam proyek Node.js kamu yang berfungsi untuk menyimpan informasi proyek dan daftar library yang dibutuhkan. Di dalamnya terdapat nama proyek (tp_gui_2311104008), versi (1.0.0), serta file utama yang akan dijalankan (index.js). Bagian scripts berisi perintah untuk dijalankan lewat terminal, walaupun saat ini hanya ada perintah test yang belum aktif. Selain itu, terdapat tiga dependensi penting, yaitu express untuk membuat server web, swagger-jsdoc untuk membuat dokumentasi otomatis dari komentar kode, dan swagger-ui-express untuk menampilkan dokumentasi tersebut dalam antarmuka web. File ini sangat penting karena memudahkan pengelolaan dan instalasi library hanya dengan satu perintah seperti npm install.