

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

```
1  const express = require('express');
2  const swaggerJsdoc = require('swagger-jsdoc');
3  const swaggerUi = require('swagger-ui-express');
4  const app = express();
5  const port = 8000;
6
7  // Middleware untuk parsing JSON
8  app.use(express.json());
9
10 // Data Mahasiswa default
11 let mahasiswaList = [
12   { nama: 'Aisyah Putri Ramadhani', nim: '2311104030' },
13   { nama: 'Bima Satria Nugraha', nim: '2311104031' },
14   { nama: 'Citra Ayu Lestari', nim: '2311104032' },
15   { nama: 'Daffa Arsyad Hermawan', nim: '2311104033' },
16   { nama: 'Elsa Fitriani', nim: '2311104034' },
17   { nama: 'Fajar Setiawan', nim: '2311104035' },
18 ];
19
20 // Swagger configuration
21 const options = {
22   definition: {
23     openapi: '3.0.0',
24     info: {
25       title: 'API Data Mahasiswa',
26       version: '1.0.0',
27       description: 'API untuk manajemen data mahasiswa',
28     },
29     servers: [
30       {
31         url: `http://localhost:${port}`,
32         description: 'server',
33       },
34     ],
35   },
36   apis: ['./mahasiswa.js'],
37 };
38
```

```

39 const specs = swaggerJsdoc(options);
40
41 /**
42  * @swagger
43  * tags:
44  *   name: Mahasiswa
45  *   description: Manajemen data mahasiswa
46  */
47
48 /**
49  * @swagger
50  * /api/mahasiswa:
51  *   get:
52  *     summary: Mendapatkan daftar semua mahasiswa
53  *     tags: [Mahasiswa]
54  *     responses:
55  *       200:
56  *         description: Sukses mendapatkan data mahasiswa
57  *         content:
58  *           application/json:
59  *             schema:
60  *               type: array
61  *               items:
62  *                 $ref: '#/components/schemas/Mahasiswa'
63  */
64 app.get('/api/mahasiswa', (req, res) => {
65   res.json(mahasiswaList);
66 });
67
68 /**
69  * @swagger
70  * /api/mahasiswa/{id}:
71  *   get:
72  *     summary: Mendapatkan data mahasiswa berdasarkan id
73  *     tags: [Mahasiswa]
74  *     parameters:
75  *       - in: path
76  *         name: id
77  *         required: true
78  *         schema:
79  *           type: integer
80  *         description: Id mahasiswa (dimulai dari 0)
81  *     responses:

```

```

39  const specs = swaggerJsdoc(options);
40
41  /**
42   * @swagger
43   * tags:
44   *   name: Mahasiswa
45   *   description: Manajemen data mahasiswa
46   */
47
48  /**
49   * @swagger
50   * /api/mahasiswa:
51   *   get:
52   *     summary: Mendapatkan daftar semua mahasiswa
53   *     tags: [Mahasiswa]
54   *     responses:
55   *       200:
56   *         description: Sukses mendapatkan data mahasiswa
57   *         content:
58   *           application/json:
59   *             schema:
60   *               type: array
61   *               items:
62   *                 $ref: '#/components/schemas/Mahasiswa'
63   */
64  app.get('/api/mahasiswa', (req, res) => {
65    res.json(mahasiswaList);
66  });
67
68  /**
69   * @swagger
70   * /api/mahasiswa/{id}:
71   *   get:
72   *     summary: Mendapatkan data mahasiswa berdasarkan id
73   *     tags: [Mahasiswa]
74   *     parameters:
75   *       - in: path
76   *         name: id
77   *         required: true
78   *         schema:
79   *           type: integer
80   *         description: Id mahasiswa (dimulai dari 0)
81   *     responses:

```

```

82 *      200:
83 *      description: Sukses mendapatkan data mahasiswa
84 *      content:
85 *      application/json:
86 *      schema:
87 *      $ref: '#/components/schemas/Mahasiswa'
88 *      404:
89 *      description: Mahasiswa tidak ditemukan
90 */
91 app.get('/api/mahasiswa/:id', (req, res) => {
92   const id = parseInt(req.params.id);
93
94   if (isNaN(id)) {
95     return res.status(400).json({ message: 'ID harus berupa angka' });
96   }
97
98   if (id >= 0 && id < mahasiswaList.length) {
99     res.json(mahasiswaList[id]);
100   } else {
101     res.status(404).json({ message: 'Mahasiswa tidak ditemukan' });
102   }
103 });
104
105 /**
106 * @swagger
107 * /api/mahasiswa:
108 *   post:
109 *     summary: Menambahkan data mahasiswa baru
110 *     tags: [Mahasiswa]
111 *     requestBody:
112 *       required: true
113 *       content:
114 *       application/json:
115 *       schema:
116 *       $ref: '#/components/schemas/MahasiswaInput'
117 *     responses:
118 *       201:
119 *       description: Mahasiswa berhasil ditambahkan
120 *       content:
121 *       application/json:
122 *       schema:
123 *       $ref: '#/components/schemas/Mahasiswa'
124 */

```

```

125 app.post('/api/mahasiswa', (req, res) => {
126   const { nama, nim } = req.body;
127   if (!nama || !nim) {
128     return res.status(400).json({ message: 'Nama dan NIM harus diisi' });
129   }
130   const newMahasiswa = { nama, nim };
131   mahasiswaList.push(newMahasiswa);
132   res.status(201).json(newMahasiswa);
133 });
134
135 /**
136  * @swagger
137  * /api/mahasiswa/{id}:
138  *   delete:
139  *     summary: Menghapus data mahasiswa berdasarkan id
140  *     tags: [Mahasiswa]
141  *     parameters:
142  *       - in: path
143  *         name: id
144  *         required: true
145  *         schema:
146  *           type: integer
147  *         description: Id mahasiswa (dimulai dari 0)
148  *     responses:
149  *       200:
150  *         description: Mahasiswa berhasil dihapus
151  *         content:
152  *           application/json:
153  *             schema:
154  *               $ref: '#/components/schemas/Mahasiswa'
155  *       400:
156  *         description: ID harus berupa angka
157  *       404:
158  *         description: Mahasiswa tidak ditemukan
159  */
160 app.delete('/api/mahasiswa/:id', (req, res) => {
161   const id = parseInt(req.params.id);
162
163   if (isNaN(id)) {
164     return res.status(400).json({ message: 'ID harus berupa angka' });
165   }
166

```

```

167   if (id >= 0 && id < mahasiswaList.length) {
168       const deletedMahasiswa = mahasiswaList.splice(id, 1)[0];
169       res.json({
170         message: 'Mahasiswa berhasil dihapus',
171         data: deletedMahasiswa,
172       });
173     } else {
174       res.status(404).json({ message: 'Mahasiswa tidak ditemukan' });
175     }
176   });
177
178   /**
179   * @swagger
180   * components:
181   *   schemas:
182   *     Mahasiswa:
183   *       type: object
184   *       properties:
185   *         nama:
186   *           type: string
187   *         nim:
188   *           type: string
189   *       example:
190   *         nama: Nama Mahasiswa
191   *         nim: NIM Mahasiswa
192   *     MahasiswaInput:
193   *       type: object
194   *       required:
195   *         - nama
196   *         - nim
197   *       properties:
198   *         nama:
199   *           type: string
200   *         nim:
201   *           type: string
202   *       example:
203   *         nama: Mahasiswa Baru
204   *         nim: Nim Baru
205   */

```

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.

```
1  {
2    "name": "tp_gui_2311104008",
3    "version": "1.0.0",
4    "main": "index.js",
5    "scripts": {
6      "test": "echo \"Error: no test specified\" && exit 1"
7    },
8    "author": "",
9    "license": "ISC",
10   "description": "",
11   "dependencies": {
12     "express": "^5.1.0",
13     "swagger-jsdoc": "^6.2.8",
14     "swagger-ui-express": "^5.0.1"
15   }
16 }
17
```

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.