

DOKUMENTASI API

Source Code:

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

app.use(express.json());

// Simpan data mahasiswa di array static
let mahasiswa = [
  { nama: "Salman Alfa Rizzi", nim: "2211104056" },
  { nama: "Arip", nim: "220220106" },
  { nama: "wiwin", nim: "1302000003" }
];

// 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}`);
});
```

Output PostMan:

Get api/mahasiswa

The screenshot shows the Postman interface for a GET request to `http://localhost:3000/api/mahasiswa`. The response is a 200 OK status with a 5 ms response time and 355 B of data. The response body is displayed in JSON format, showing an array of three student objects.

Key	Value	Description
Key	Value	Description

```

1  [
2    {
3      "nama": "Salman Alfa Rizzi",
4      "nim": "2211104056"
5    },
6    {
7      "nama": "Arip",
8      "nim": "220220106"
9    },
10   {
11     "nama": "wiwin",
12     "nim": "1302000003"
13   }
14 ]
```

Post api/mahasiswa

The screenshot shows a REST client interface with a dark theme. At the top, the request method is **POST** and the URL is `http://localhost:3000/api/mahasiswa`. The request body is a JSON object: `{ "nama": "arip", "nim": "2211104057" }`. The response status is **201 Created** with a response time of 36 ms and a body size of 279 B. The response body is a JSON object: `{ "message": "Data berhasil ditambahkan" }`.

POST `http://localhost:3000/api/mahasiswa` **Send**

Params Authorization Headers (9) **Body** Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   "nama": "arip",
3   "nim": "2211104057"
4 }
```

Body Cookies Headers (7) Test Results **201 Created** • 36 ms • 279 B • •

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "message": "Data berhasil ditambahkan"
3 }
```

Lalu Get lagi untuk cek udah masuk atau belum datanya

The screenshot shows a REST client interface with a GET request to `http://localhost:3000/api/mahasiswa`. The request is configured with the method `GET` and the URL `http://localhost:3000/api/mahasiswa`. The response is a `200 OK` status with a response time of `8 ms` and a body size of `390 B`. The response body is displayed in JSON format, showing an array of four student objects.

```
1 {
2   "nama": "arip",
3   "nim": "2211104057"
4 }
```

```
1 [
2   {
3     "nama": "Salman Alfa Rizzi",
4     "nim": "2211104056"
5   },
6   {
7     "nama": "Arip",
8     "nim": "220220106"
9   },
10  {
11    "nama": "wiwin",
12    "nim": "1302000003"
13  },
14  {
15    "nama": "arip",
16    "nim": "2211104057"
17  }
18 ]
```

Cek index 0 dengan menambahkan /0

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

- Request:** GET `http://localhost:3000/api/mahasiswa/0`
- Response:** 200 OK, 6 ms, 282 B
- Body (JSON):**

```
1 {
2   "nama": "Salman Alfa Rizzi",
3   "nim": "2211104056"
4 }
```

Lalu coba delete 1 data dengan pake DELETE

The screenshot shows an API client interface with a dark theme. At the top, the URL bar displays `DEL http://localhost:3000/api/`. Below it, the request path is `http://localhost:3000/api/mahasiswa/3`. The HTTP method is set to **DELETE**. The request body is a JSON object:

```
{  "nama": "arip",  "nim": "2211104057"}
```

. The interface includes tabs for Params, Authorization, Headers (9), Body, Scripts, and Settings. The Body tab is active, showing the raw JSON. Below the request, the response is displayed with a status of **200 OK**, a time of 8 ms, and a size of 270 B. The response body is a JSON object:

```
{  "message": "Data berhasil dihapus"}
```

. The response is shown in the 'Pretty' format.

DEL `http://localhost:3000/api/`

`http://localhost:3000/api/mahasiswa/3`

DELETE `http://localhost:3000/api/mahasiswa/3` **Send**

Params Authorization Headers (9) **Body** Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

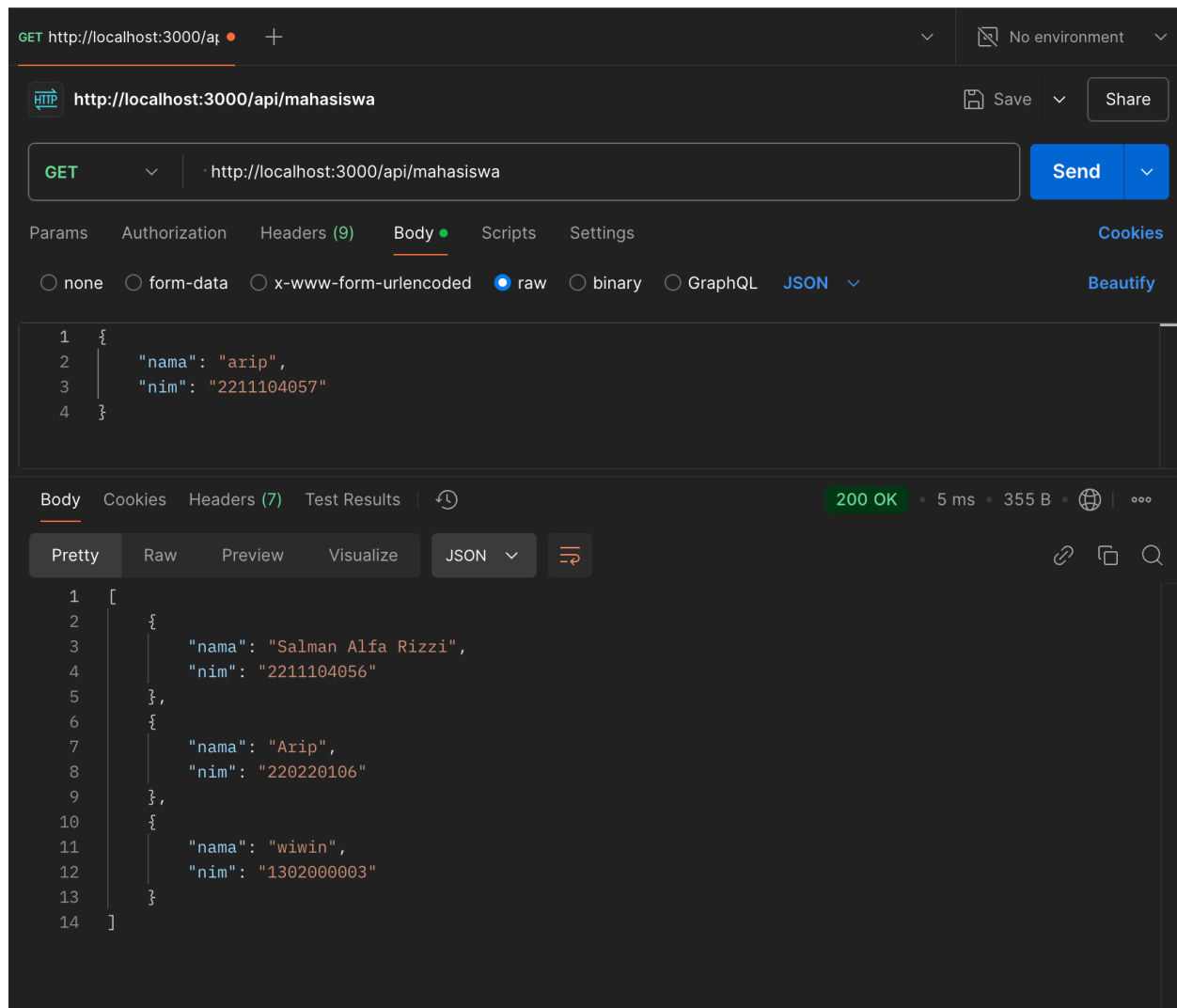
```
1 {
2   "nama": "arip",
3   "nim": "2211104057"
4 }
```

Body Cookies Headers (7) Test Results **200 OK** • 8 ms • 270 B •

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "message": "Data berhasil dihapus"
3 }
```

Cek apakah sudah kehapus pake GET



Penjelasan Program:

Program ini adalah sebuah RESTful API sederhana yang dibangun menggunakan Node.js dan Express.js untuk mengelola data mahasiswa. API ini menyediakan beberapa endpoint, seperti GET `/api/mahasiswa` untuk menampilkan seluruh data mahasiswa, POST `/api/mahasiswa` untuk menambahkan data baru, GET `/api/mahasiswa/:id` untuk melihat data mahasiswa berdasarkan indeks, serta DELETE `/api/mahasiswa/:id` untuk menghapus data berdasarkan indeks. Data mahasiswa disimpan sementara di dalam array, sehingga setiap perubahan hanya bersifat sementara (tidak tersimpan permanen). API ini dapat diakses dan diuji menggunakan Postman, dan sangat cocok digunakan untuk pembelajaran dasar pemrograman backend dan pengujian dengan metode HTTP.