

# Laporan Praktikum WSE #7

Mata Kuliah : Web Service Engineering  
Dosen Pengampu : Muhayat, M.IT  
Praktikum : P7 – RESTful API Hardening & Observability  
Nama Mahasiswa : SITI MAGFIRATUN WARAHMAH  
NIM : 230104040059  
Kelas : TI23B  
Tanggal Praktikum : 17-11-2025

## A. Tujuan Praktikum

1. Menerapkan middleware keamanan dasar (Helmet, Rate Limit, CORS).
2. Menambahkan logging sistematis menggunakan morgan atau winston.
3. Menerapkan error handling global dan response konsisten.
4. Menggunakan environment variable (.env) untuk konfigurasi sensitif.
5. Memahami konsep monitoring sederhana (service health & uptime endpoint).
6. Menjaga struktur modular dari project Express.js.

## B. Lingkungan & Tools

Kategori	: Tools / Library
Runtime	: Node.js 18+
Framework	: Express.js
Keamanan	: helmet, cors, express-rate-limit
Logging	: morgan (basic), winston (optional advanced)
Env Config	: dotenv
Testing	: Postman / Thunder Client
Dokumentasi	: Markdown (README.md) atau Swagger (optional)

## C. Aktivitas Praktikum

Langkah	Aktivitas Mahasiswa	Tujuan
1	Clone project hasil UTS ke folder baru P7-Hardening-230104040059	Reuse project dasar
2	Install paket baru: npm install helmet cors express-rate-limit dotenv morgan	Setup environment
3	Buat file .env dan .env.example	Pisahkan konfigurasi (port, mode, rate-limit, dsb)
4	Tambahkan <b>Helmet</b> di app.js	Tambahkan security headers
5	Tambahkan <b>CORS</b> → izinkan hanya domain tertentu	Latih konsep origin restriction

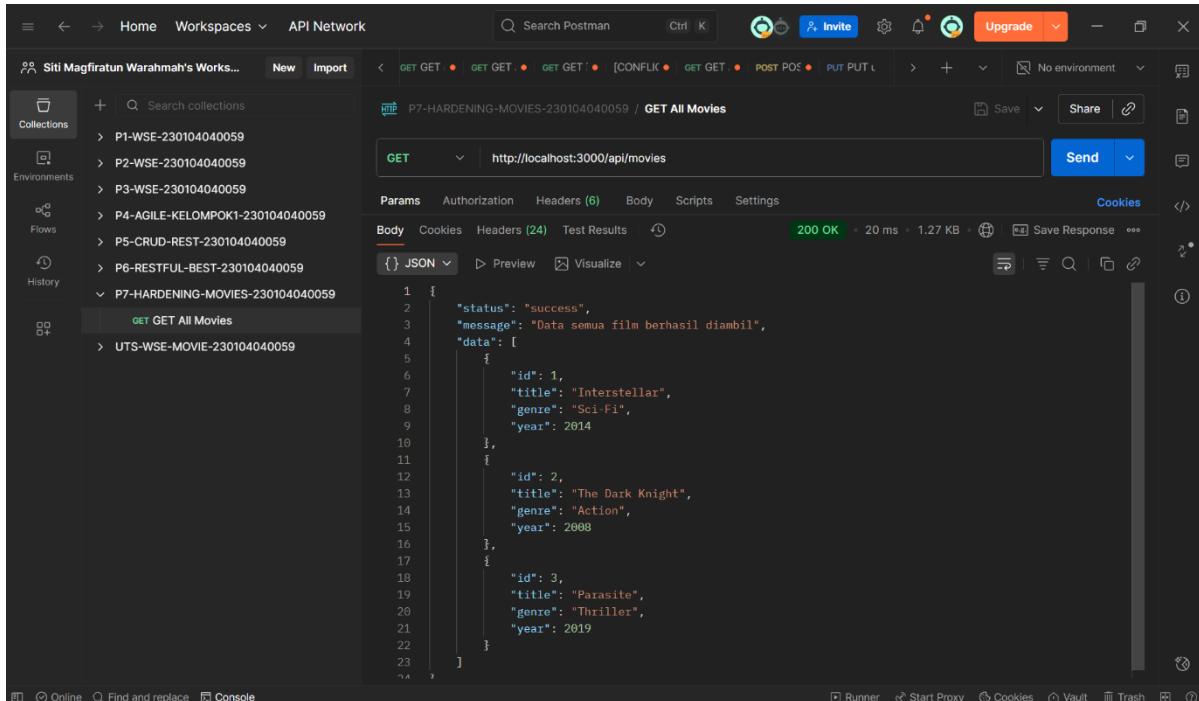
6	Tambahkan <b>Rate Limiter</b> (misal max 100 request / 15 menit)	Cegah serangan DDoS sederhana
7	Implementasikan <b>morgan</b> (atau winston) untuk logging request ke file	Latih observability
8	Tambahkan <b>Global Error Handler</b> → middleware terakhir	Tangani semua error konsisten
9	Tambahkan endpoint /api/health dan /api/metrics	Monitoring & uptime check
10	Dokumentasikan hasil (README + screenshot Postman + contoh log)	Bukti penerapan

Method	Endpoint	Deskripsi	Autentikasi	Status Code	Keterangan
GET	/api/<resource>	Mendapatkan semua data resource	Tidak	200	Resource berasal dari project UTS
GET	/api/<resource>/:id	Mendapatkan data berdasarkan ID	Tidak	200 / 404	404 jika ID tidak ditemukan
POST	/api/<resource>	Menambah data baru	Tidak	201 / 400	Validasi input wajib
PUT	/api/<resource>/:id	Mengubah data berdasarkan ID	Tidak	200 / 400 / 404	Konsisten dengan RESTful principles
DELETE	/api/<resource>/:id	Menghapus data berdasarkan ID	Tidak	204 / 404	Response kosong jika berhasil
GET	/api/info	Menampilkan informasi service	Tidak	200	Metadata API & identitas
GET	/api/health	Mengecek status API	Tidak	200	Monitoring uptime & environment
ANY	endpoint tidak dikenal	Handler 404 global	Tidak	404	Ditangani oleh middleware
ERROR	internal server error	Global Error Handler	Tidak	500	Response error JSON

## D. Hasil & Bukti

Lampirkan Screenshot Hasil Uji Endpoint di Postman

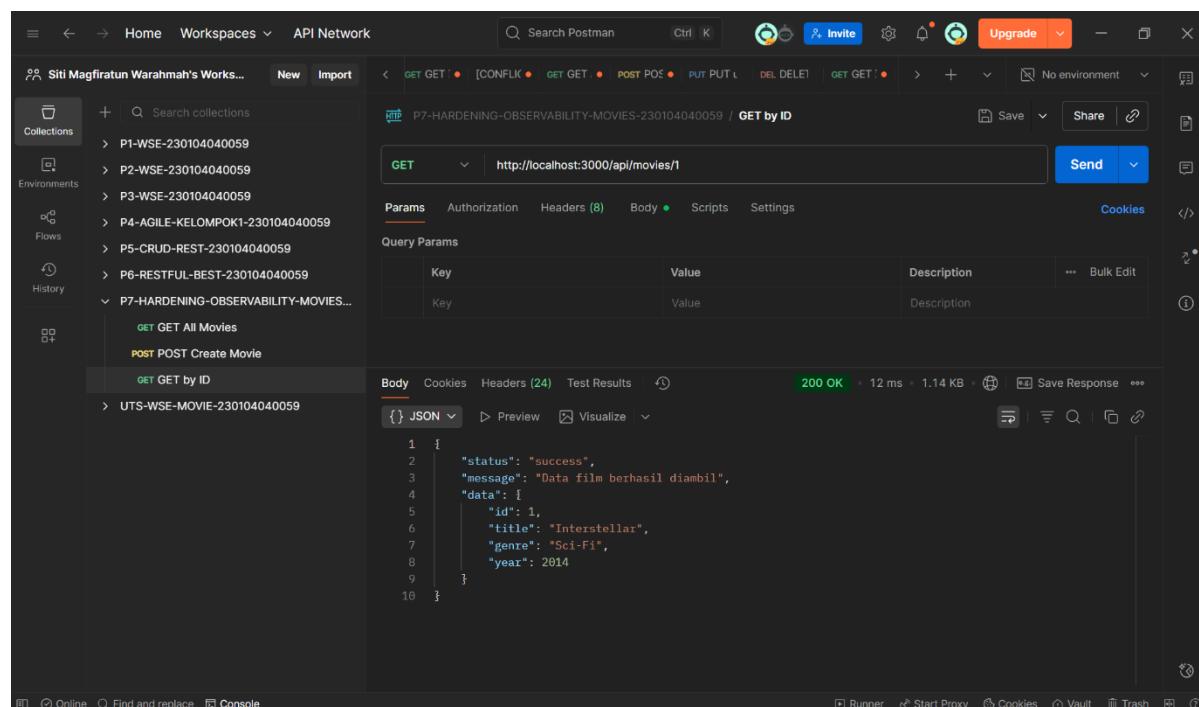
- GET /api/movies (Mendapatkan semua data movie) → 200 OK



The screenshot shows the Postman interface with a dark theme. On the left, there's a sidebar with 'Collections' containing several items like P1-WSE, P2-WSE, P3-WSE, P4-AGILE, P5-CRUD-REST, P6-RESTFUL-BEST, and P7-HARDENING-MOVIES. The main area shows a collection named 'P7-HARDENING-MOVIES-230104040059'. A specific endpoint 'GET All Movies' is selected. The request URL is 'http://localhost:3000/api/movies'. The response status is '200 OK' with a response time of '20 ms' and a size of '1.27 KB'. The response body is a JSON object:

```
1 {  
2     "status": "success",  
3     "message": "Data semua film berhasil diambil",  
4     "data": [  
5         {  
6             "id": 1,  
7             "title": "Interstellar",  
8             "genre": "Sci-Fi",  
9             "year": 2014  
10        },  
11        {  
12            "id": 2,  
13            "title": "The Dark Knight",  
14            "genre": "Action",  
15            "year": 2008  
16        },  
17        {  
18            "id": 3,  
19            "title": "Parasite",  
20            "genre": "Thriller",  
21            "year": 2019  
22        }  
23    ]  
24}
```

- GET /api/movies/1 (Mendapatkan data berdasarkan ID) → 200 OK



The screenshot shows the Postman interface with a dark theme. On the left, there's a sidebar with 'Collections' containing several items like P1-WSE, P2-WSE, P3-WSE, P4-AGILE, P5-CRUD-REST, P6-RESTFUL-BEST, and P7-HARDENING-OBSERVABILITY-MOVIES. The main area shows a collection named 'P7-HARDENING-OBSERVABILITY-MOVIES-230104040059'. An endpoint 'GET by ID' is selected. The request URL is 'http://localhost:3000/api/movies/1'. The response status is '200 OK' with a response time of '12 ms' and a size of '1.14 KB'. The response body is a JSON object:

```
1 {  
2     "status": "success",  
3     "message": "Data film berhasil diambil",  
4     "data": [  
5         {  
6             "id": 1,  
7             "title": "Interstellar",  
8             "genre": "Sci-Fi",  
9             "year": 2014  
10        }  
11    ]  
12}
```

POST /api/movies (Menambah data baru) → 201 Created

The screenshot shows the Postman interface with a collection named "Siti Magfiratur Warahmah's Works...". A POST request is made to `http://localhost:3000/api/movies`. The request body contains:

```
1 {
2   "title": "Interstellar",
3   "genre": "Sci-Fi",
4   "year": 2014
5 }
```

The response status is 201 Created, with a response body:

```
1 {
2   "status": "success",
3   "message": "Data film berhasil dibuat",
4   "data": [
5     {
6       "id": 4,
7       "title": "Interstellar",
8       "genre": "Sci-Fi",
9       "year": 2014
10    }
11 }
```

PUT /api/movies/1 (Mengubah data berdasarkan ID) → 200 OK

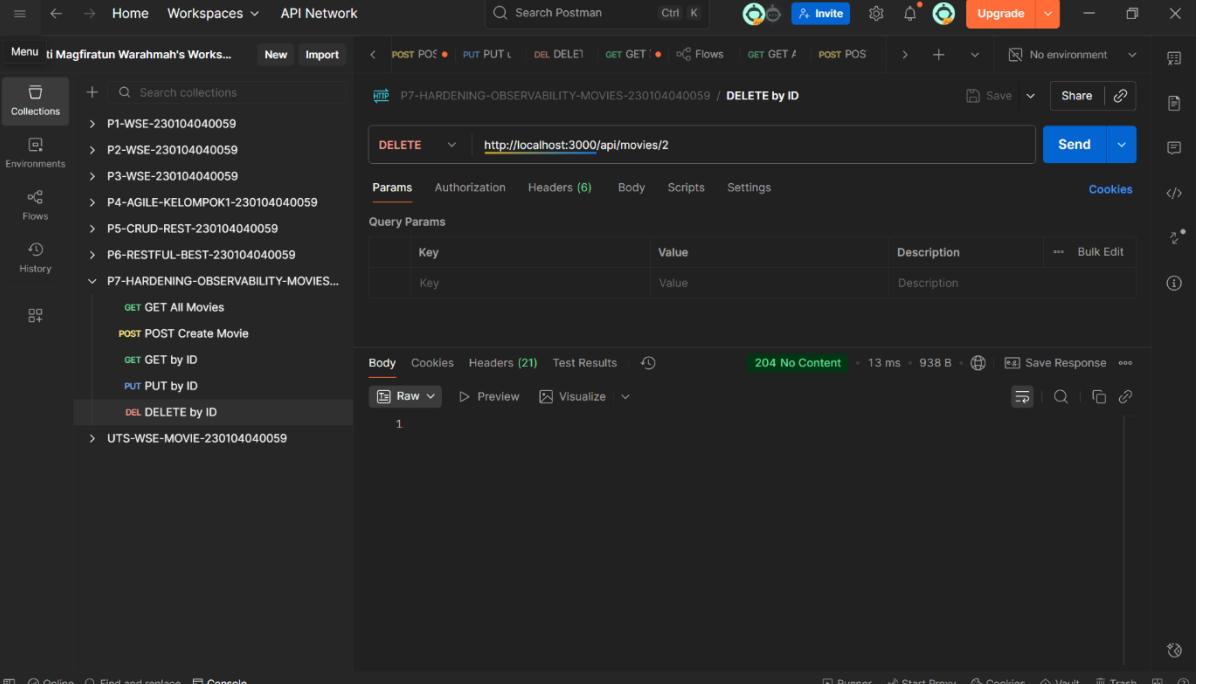
The screenshot shows the Postman interface with a collection named "Siti Magfiratur Warahmah's Works...". A PUT request is made to `http://localhost:3000/api/movies/1`. The request body contains:

```
1 {
2   "title": "Interstellar - Director's Cut",
3   "genre": "Sci-Fi",
4   "year": 2014
5 }
```

The response status is 200 OK, with a response body:

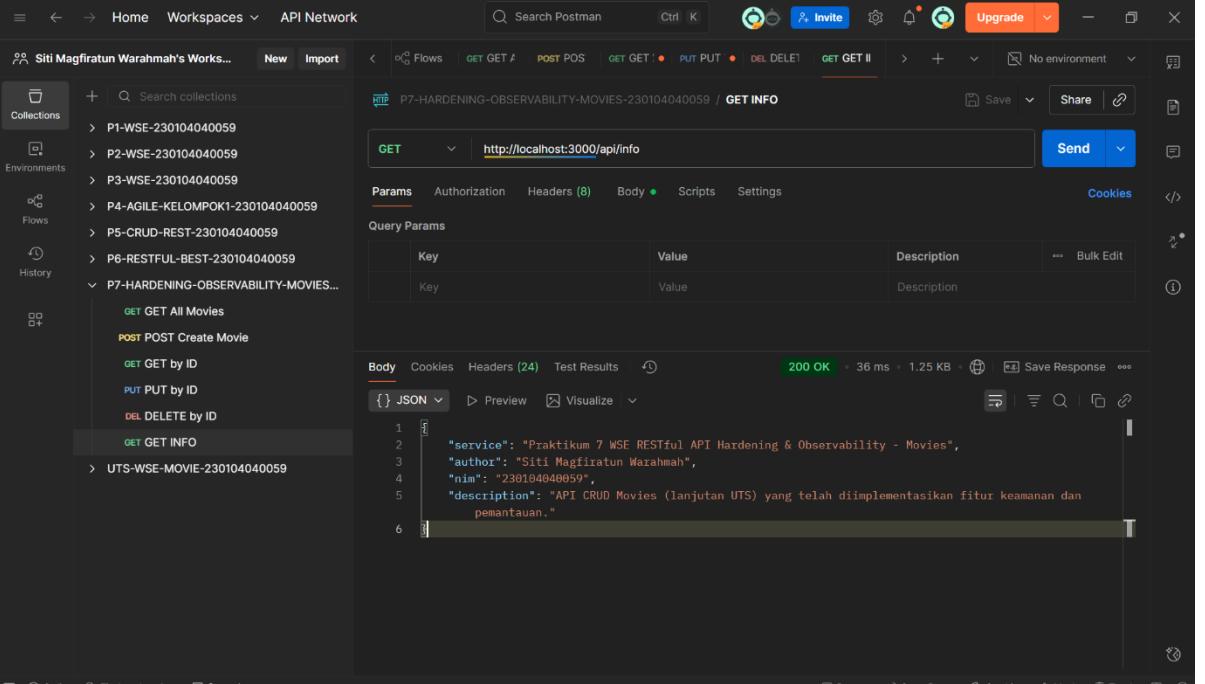
```
1 {
2   "status": "success",
3   "message": "Data film berhasil diperbarui",
4   "data": [
5     {
6       "id": 1,
7       "title": "Interstellar - Director's Cut",
8       "genre": "Sci-Fi",
9       "year": 2014
10    }
11 }
```

**DELETE /api/movies/2 (Menghapus data berdasarkan ID) → 204 No Content**



The screenshot shows the Postman interface with a collection named "P7-HARDENING-OBSERVABILITY-MOVIES-230104040059". A "DELETE by ID" request is selected, targeting the URL `http://localhost:3000/api/movies/2`. The response is a 204 No Content status with a 13 ms duration and 938 B size.

**GET /api/info (Menampilkan Informasi service) → 200 OK**



The screenshot shows the Postman interface with the same collection. A "GET INFO" request is selected, targeting the URL `http://localhost:3000/api/info`. The response is a 200 OK status with a 36 ms duration and 1.25 KB size. The response body is a JSON object:

```
1 "service": "Praktikum 7 WSE RESTful API Hardening & Observability - Movies",
2 "author": "Siti Magfiratur Warahmah",
3 "nim": "230104040059",
4 "description": "API CRUD Movies (lanjutan UTS) yang telah diimplementasikan fitur keamanan dan pemantauan."
```

GET /api/health (Mengecek status API) → 200 OK

The screenshot shows the Postman interface with the following details:

- Collection:** Siti Magfiratur Warahmah's Works...
- Request:** GET /api/health
- Response Status:** 200 OK
- Body (JSON):**

```
1 {  
2   "status": "ok",  
3   "timestamp": "2025-11-23T06:09:48.551Z"  
4 }
```

GET (ANY) /api/salah-alamat (Handler 404 global) → 404 Not Found

The screenshot shows the Postman interface with the following details:

- Collection:** Siti Magfiratur Warahmah's Works...
- Request:** GET /api/salah-alamat
- Response Status:** 404 Not Found
- Body (JSON):**

```
1 {  
2   "status": "fail",  
3   "message": "Endpoint tidak ditemukan"  
4 }
```

## ❖ GET (ERROR) /api/movies (Global error handler) → 500 Internal Server Error

The screenshot shows the Postman application interface. On the left, there's a sidebar with collections, environments, flows, and history. The main area displays a request for 'P7-HARDENING-OBSERVABILITY-MOVIES-2301040059 / ERROR internal server'. The method is set to 'GET' and the URL is 'http://localhost:3000/api/movies'. The 'Headers' tab shows '(24)' entries. The 'Body' tab is selected and shows a JSON response with the following content:

```
1 | {  
2 |   "status": "error",  
3 |   "message": "Pengujian Global Error Handler 500"  
4 | }
```

The status bar at the bottom indicates a '500 Internal Server Error' with a duration of '38 ms' and a size of '1.1 KB'.

## E. Analisis

Praktikum ini bertujuan memperkuat API (Hardening) dan mempermudah pemantauan (Observability). Dari sisi keamanan, fitur Helmet (untuk header HTTP) dan Rate Limiter (membatasi request) sudah berhasil diuji dan berfungsi melindungi API dari traffic berlebihan. Dari sisi pemantauan, middleware Morgan Logger bekerja mencatat semua aktivitas API ke file. Bagian terpenting, Global Error Handler terpusat kita berhasil menangkap error internal 500 dan bahkan error 404 pada endpoint yang salah, di mana keduanya diubah menjadi respons JSON yang konsisten, tidak lagi berupa HTML error bawaan. Ini membuktikan bahwa semua lapisan pengamanan dan pelaporan error sudah terpasang dengan baik.

## F. Kesimpulan

Secara keseluruhan, implementasi Hardening dan Observability pada API CRUD movies dinyatakan berhasil. Semua endpoint yang sudah ada (CRUD) tetap berfungsi, dan enam fitur tambahan (Helmet, CORS, Rate Limiter, Logger, Health Check, dan Global Error Handler) telah terintegrasi dengan sempurna. API kini tidak hanya dapat melakukan operasi data, tetapi juga lebih aman, lebih informatif (melalui /api/info dan log), dan lebih stabil dalam menangani segala jenis kegagalan dengan respons JSON yang rapi.