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 **Helmet** di app.js	Tambahkan security headers
5	Tambahkan **CORS** → izinkan hanya domain tertentu	Latih konsep origin restriction

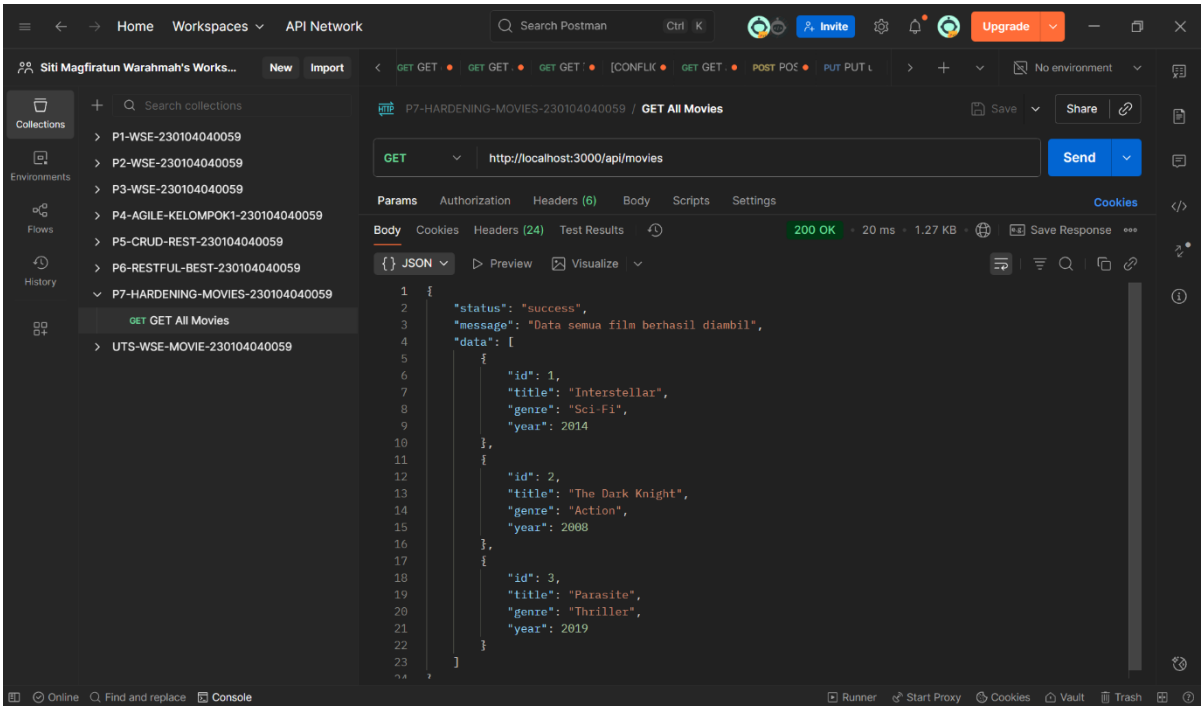
6	Tambahkan **Rate Limiter** (misal max 100 request / 15 menit)	Cegah serangan DDoS sederhana
7	Implementasikan **morgan** (atau winston) untuk logging request ke file	Latih observability
8	Tambahkan **Global Error Handler** → middleware terakhir	Tangani semua error konsisten
9	Tambahkan endpoint /api/health 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

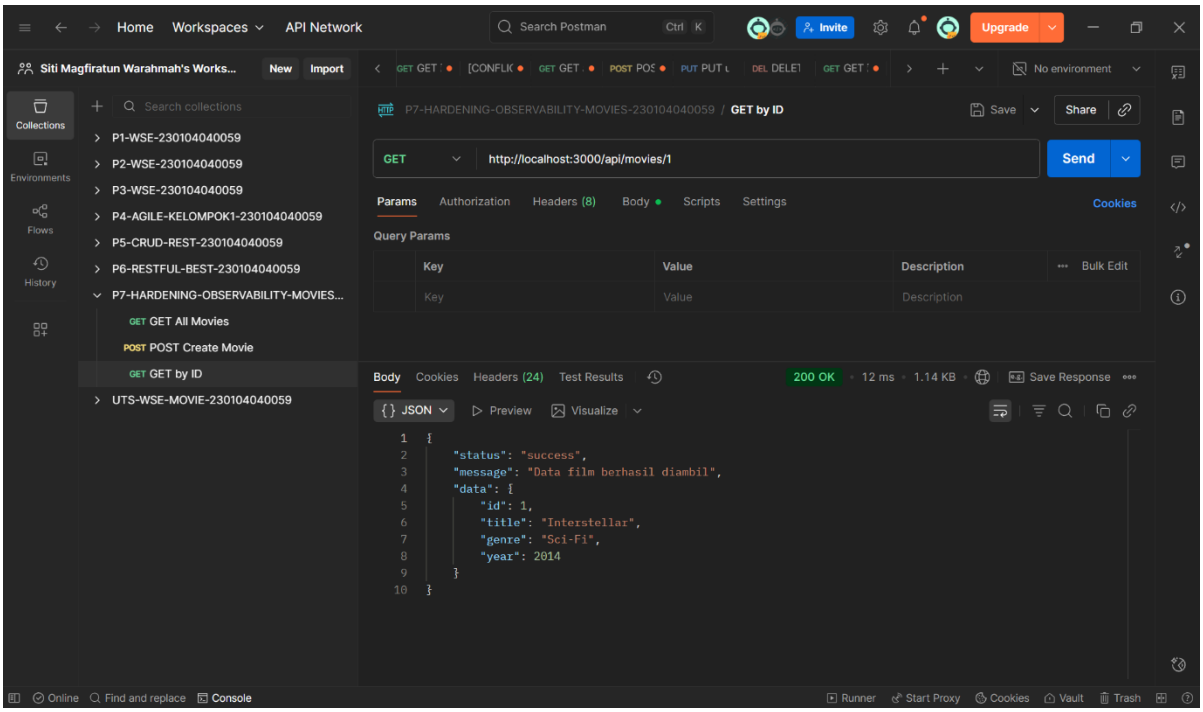
D. Hasil & Bukti

Lampirkan **Screenshot Hasil Uji** Endpoint di Postman

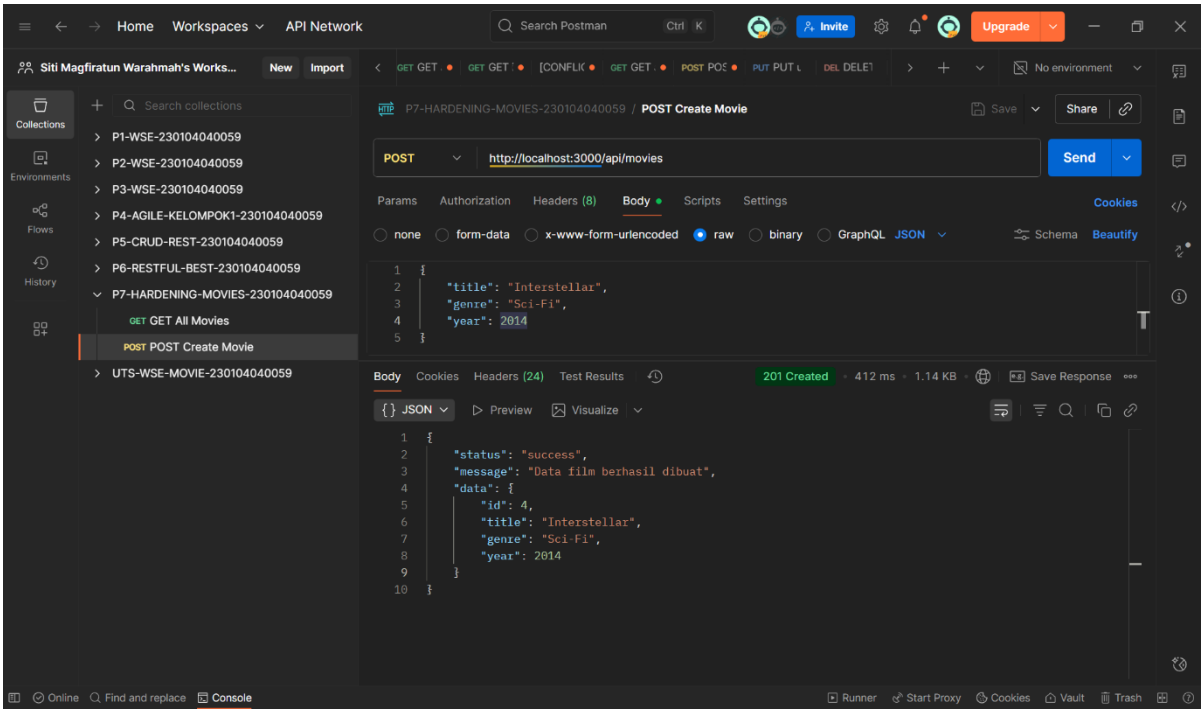
☑ GET /api/movies (Mendapatkan semua data movie) → 200 OK



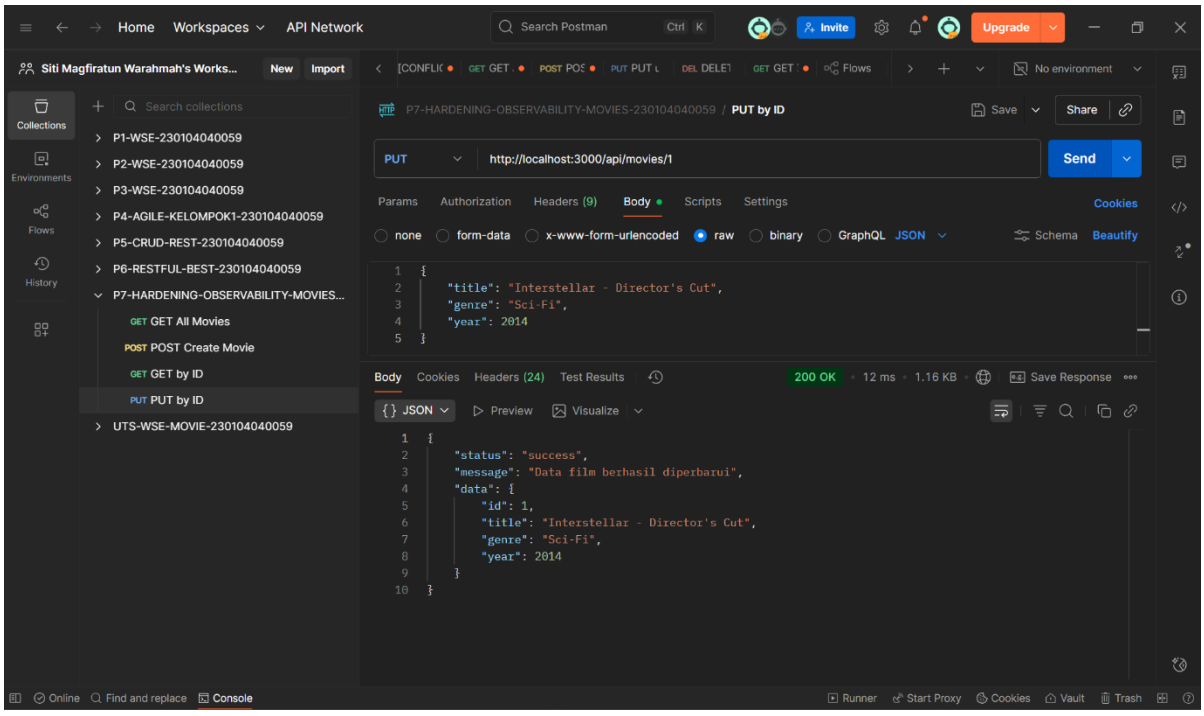
☑ GET /api/movies/1 (Mendapatkan data berdasarkan ID) → 200 OK



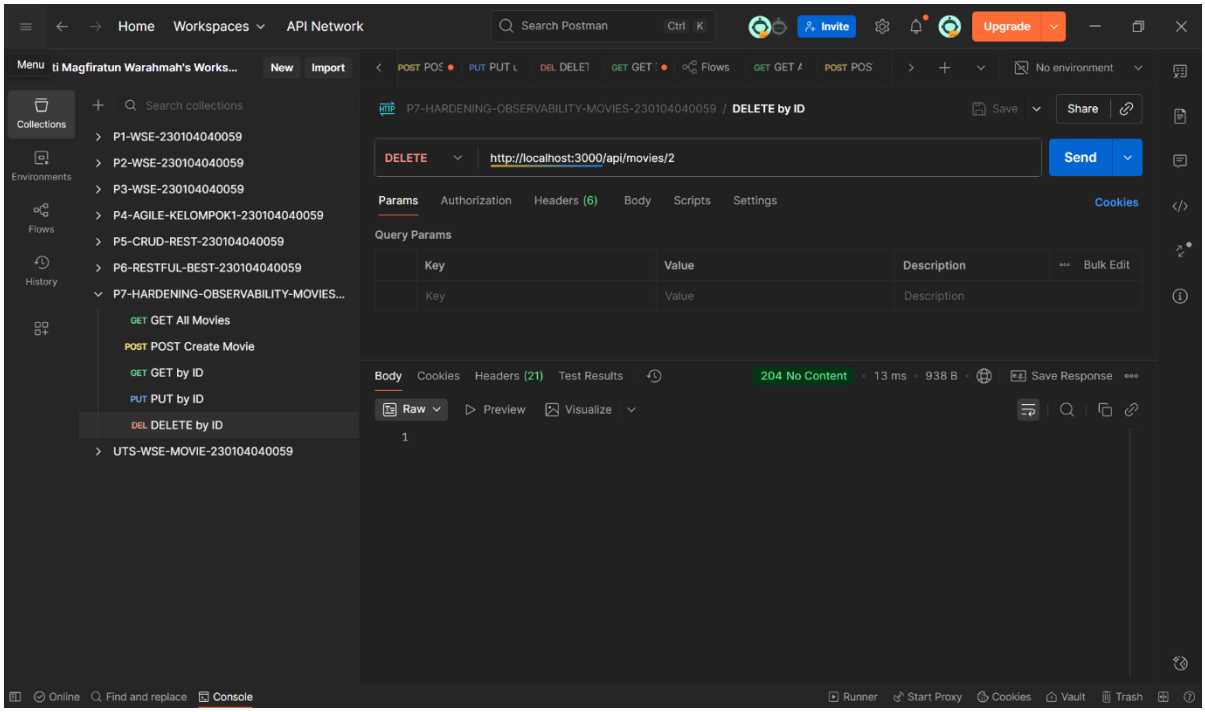
☑ POST /api/movies (Menambah data baru) → 201 Created



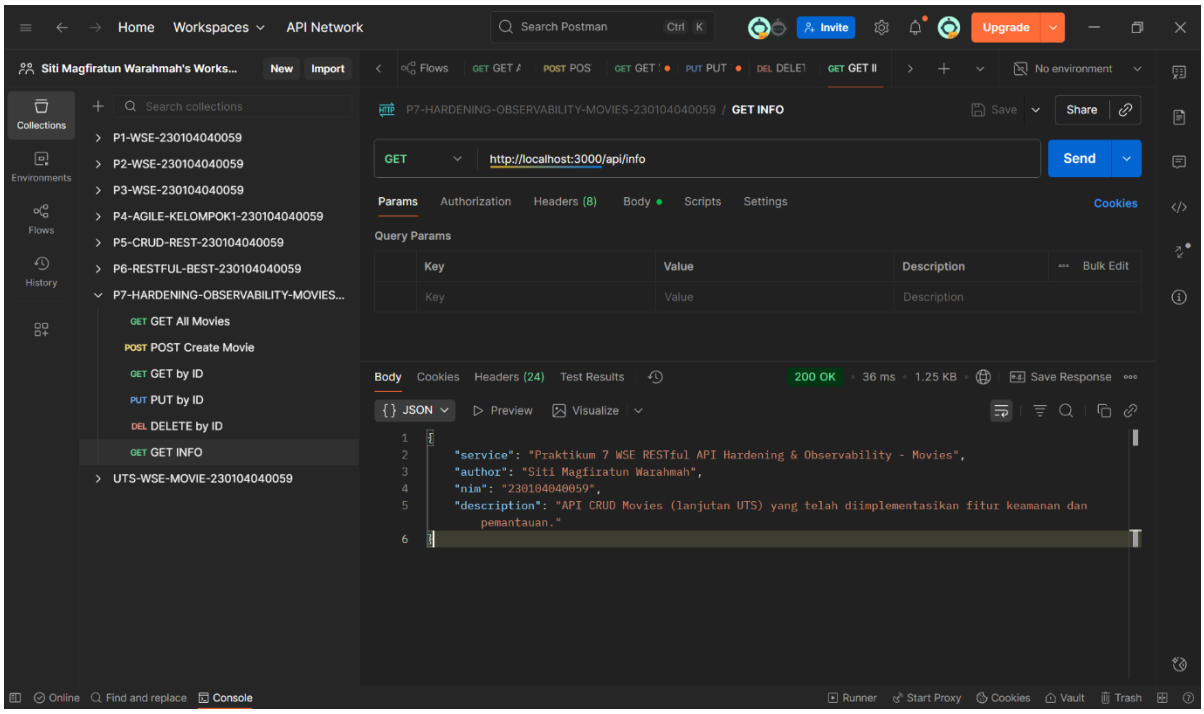
☑ PUT /api/movies/1 (Mengubah data berdasarkan ID) → 200 OK



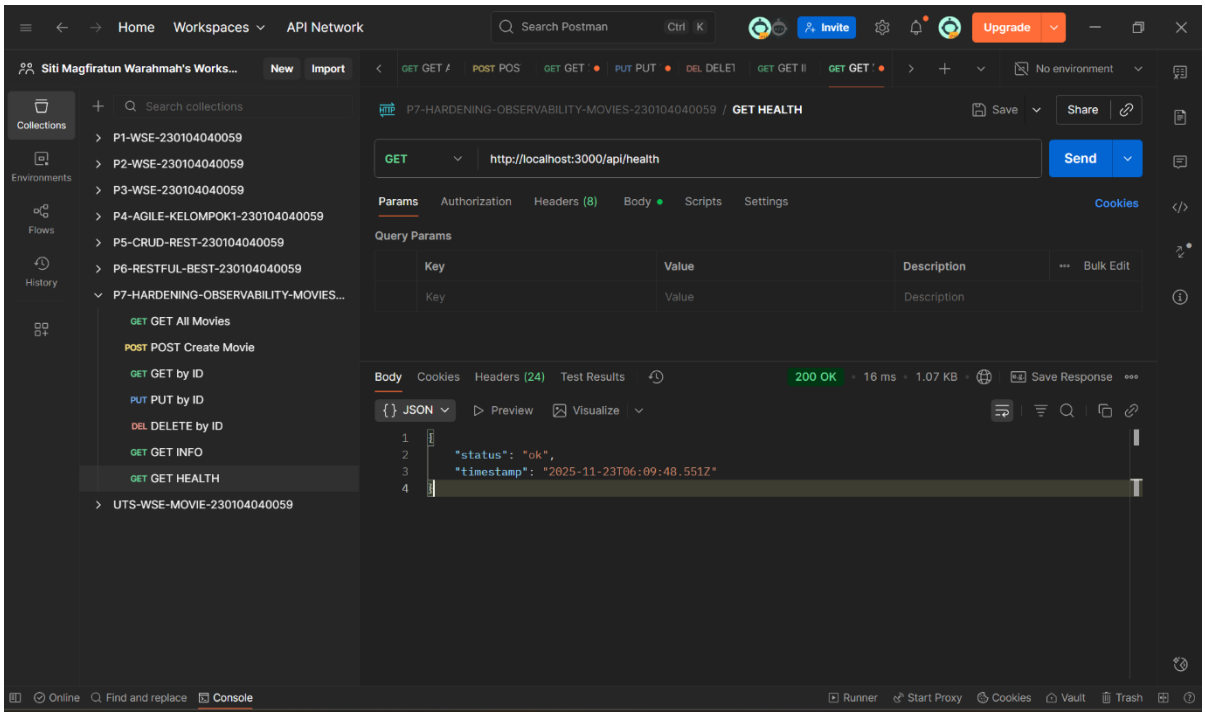
☑ DELETE /api/movies/2 (Menghapus data berdasarkan ID) → 204 No Content



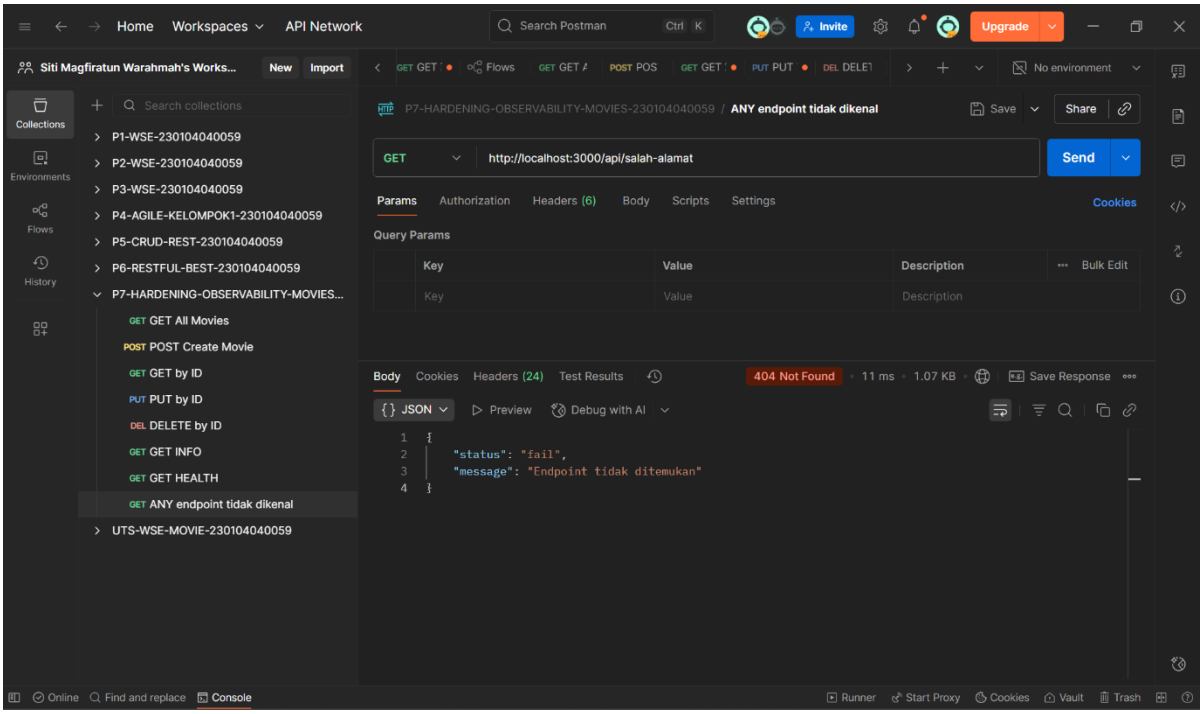
☑ GET /api/info (Menampilkan Informasi service) → 200 OK



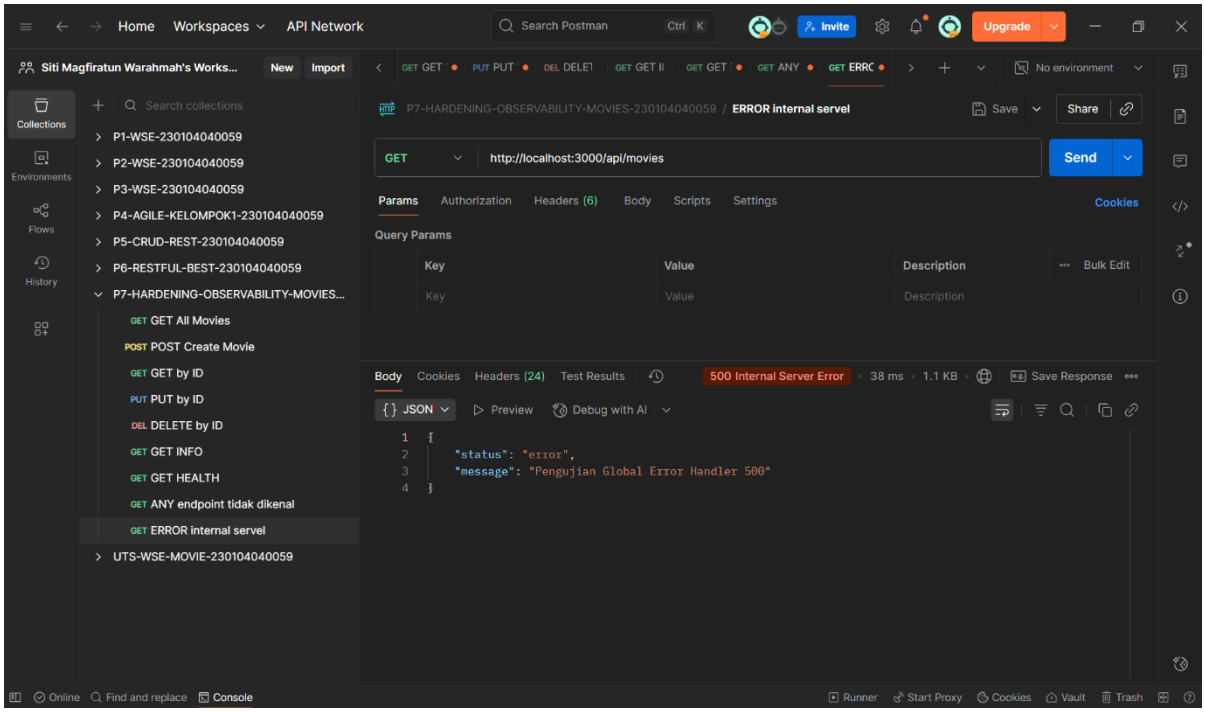
☑ GET /api/health (Mengecek status API) → 200 OK



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



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



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.