

LAPORAN Laporan Analisis Studio Ghibli API

Disusun guna memenuhi tugas mata kuliah

PRATIKUM Interoperabilitas

Dosen Pengampu:

Sepyan Purnama Kristanto, S.Kom., M.Kom.



Oleh:

Mohammad Andy YusriL : 362458302111

Reyfta Ogy Reynata : 362458302107

Ahmad Nur Bintang A : 362458302103

I Kadek Dika D : 362458302053

KELAS 2E

PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK

POLITEKNIK NEGERI BANYUWANGI

TAHUN 2025

1. Studio Ghibli API adalah layanan RESTful API publik yang menyediakan data tentang film dan dunia Studio Ghibli dalam format JSON. API ini berfungsi untuk memberikan informasi seputar daftar film beserta detailnya seperti judul, sutradara, tahun rilis, deskripsi, dan skor. Selain itu, API ini juga menyajikan data tentang karakter, lokasi, spesies, serta kendaraan yang muncul di dalam film. Dengan adanya API ini, pengembang dapat menggunakannya untuk membuat aplikasi web, mobile, atau proyek lain yang menampilkan katalog film, informasi karakter, hingga detail dunia Studio Ghibli secara praktis dan terstruktur.

2. Analisis Sumber Daya **films**:

- Jelaskan struktur data JSON dari satu objek film. Sebutkan minimal 5 properti kunci dan tipe datanya (misal: title (String), release date (String)).

Nama	Tipe data	Deskripsi
Director	String	Untuk menunjukan nama sutradara
Description	String	Synopsis film
People	Array	Untuk menunjukan nama karakter
Location	Array	Untuk menunjuk Lokasi yang ada di film
Vehicles	Array	Daftar kendaraan yang ada di film

- Tempelkan contoh respons JSON (potongan yang relevan saja) dari GET/films/{id}

```
- const express = require('express');  
- const cors = require('cors');  
- const { error } = require('console');  
- 
```

```

- const app = express();
- const PORT = 3200;
- // // 3
- let idSeq = 3;
- // // 2
- let movies = [
-   { id: 1, title: 'parasite', director: 'bong joon-ho', year: 2019 },
-   { id: 2, title: 'The dark knight', director: 'nolan', year: 2019 },
- ];
-
- let directors = [
-   { id: 1, nama: 'kastoyo', birthyear: 950 },
-   { id: 2, nama: 'kaspri', birthyear: 1950 },
- ];
- // middlewere
- app.use(cors());
- // 3
- // middlewere express.json
- app.use(express.json());
- // route
- app.get('/status', (req, res) => {
-   res.json({
-     ok: true,
-     service: 'film-api',
-     time: new Date().toISOString()
-   });
- });
- // 2
- app.get('/movies', (req, res) => {

```

```

-   res.json(movies);
-   });
-   app.get('/movies/:id', (req, res) => {
-       const id = Number(req.params.id);
-       const movie = movies.find(m => m.id === id);
-       if (!movie) return res.status(404).json({ error: 'Movie tidak
ditemukan' });
-       res.json(movie);
-   });
-   // 3
-   //POST/movies-Membuatfilmbaru
-   app.post('/movies', (req, res) => {
-       const { title, director, year } = req.body || {};
-       if (!title || !director || !year) {
-           return res.status(400).json({ error: 'title, director, year
wajib diisi' });
-       }
-       const newMovie = { id: idSeq++, title, director, year };
-       movies.push(newMovie);
-       res.status(201).json(newMovie);
-   });
-   //PUT/movies/:id-Memperbaruidatafilm
-   app.put('/movies/:id', (req, res) => {
-       const id = Number(req.params.id);
-       const movieIndex = movies.findIndex(m => m.id === id);
-       if (movieIndex === -1) {
-           return res.status(404).json({ error: 'movie tidak
ditemukan' });
-       }

```

```

-   const { title, director, year } = req.body || {};
-   const updateMovie = { id, title, director, year };
-   movies[movieIndex] = updateMovie;
-   res.json(updateMovie);
- });
- //DELETE/movies/:id-Menghapusfilm
- app.delete('/movies/:id', (req, res) => {
-   const id = Number(req.params.id);
-   const movieIndex = movies.findIndex(m => m.id === id);
-   if (movieIndex === -1) {
-     return res.status(404).json({ error: 'movie tidak
ditemukan' });
-   }
-   movies.splice(movieIndex, 1);
-   res.status(204).send();
- });
-
- //mengamnil satu direktor
- app.get('/directors/:id', (req, res) => {
-   const id = Number(req.params.id);
-   const directorIndex = director.findIndex(d => d.id === id);
-   if (directorIndex === -1) {
-     return res.status(404).json({ error: 'director e ilang pak!'
});
-   }
-   res.json(director);
- }
- });
- // mengambil semua direktor
- app.get('/directors', (req, res) => {

```

```

-     res.json(directors);
-   });
-
-   // membuat sutradara baru
-   app.post('/directors', (req, res) => {
-     const { nama, birthyear } = req.body;
-     if (!nama || !birthyear ) {
-       return res.status(400).json({ error: 'nama, birthyear
wajib diisi' });
-     }
-     const newdirektor = { id: idSeq++, nama, birthyear };
-     directors.push(newdirektor);
-     res.status(201).json(newdirektor);
-   });
-
-   // PUT
-   app.put('/movies/:id', (req, res) => {
-     const id = Number(req.params.id);
-     const movieIndex = movies.findIndex(m => m.id === id);
-     if (movieIndex === -1) {
-       return res.status(404).json({ error: 'movie tidak
ditemukan' });
-     }
-     const { title, director, year } = req.body || {};
-     const updateMovie = { id, title, director, year };
-     movies[movieIndex] = updateMovie;
-     res.json(updateMovie);
-   });
-

```

```

- //DELETE
- app.delete('/movies/:id', (req, res) => {
-   const id = Number(req.params.id);
-   const movieIndex = movies.findIndex(m => m.id === id);
-   if (movieIndex === -1) {
-     return res.status(404).json({ error: 'movie tidak
ditemukan' });
-   }
-   movies.splice(movieIndex, 1);
-   res.status(204).send();
- });
-
-
- // middleware fallback untuk menangani rute 404 Not found
- app.use((req, res) => {
-   res.status(404).json({ error: 'Rute tidak ditemukan' });
- });
- // 4
- app.use((err, req, res, _next) => {
-   console.error('[EROR]')
- });
- // Start server
- app.listen(PORT, () => {
-   console.log(server aktif di
http://localhost:${PORT}/movies);
- });

```

des

3. Endpoint yang Dianalisis:

- GET /films: Jelaskan fungsinya.

Endpoint **GET /films** berfungsi untuk mengambil daftar semua film yang tersedia di Studio Ghibli API. Saat endpoint ini dipanggil, server akan merespons dengan data dalam format JSON berupa array berisi objek film.

Setiap objek film di dalam array berisi informasi penting seperti id, title, original_title, description, director, producer, release_date, running_time, rt_score, serta relasi ke resource lain seperti people, species, locations, dan vehicles.

- GET /films/{id}: Jelaskan fungsinya.

Endpoint GET /films/{id} digunakan untuk mengambil data detail dari satu film tertentu berdasarkan id yang diminta. Jika GET /films menampilkan daftar semua film dalam bentuk array, maka GET /films/{id} hanya menampilkan satu objek JSON yang berisi informasi lengkap tentang film tersebut. Data yang ditampilkan meliputi id, title, original_title, description, director, producer, release_date, running_time, rt_score, serta relasi ke resource lain seperti people, species, locations, dan vehicles. Dengan endpoint ini, pengguna dapat memperoleh detail spesifik dari sebuah film Studio Ghibli untuk ditampilkan dalam aplikasi atau kebutuhan lainnya.