LAPORAN TUGAS BESAR REKAYASA PERANGKAT LUNAK
SISTEM PENCARIAN FILM
"MOVIEKU"

Dosen Pengampu:

Rizka Ardiansyah, S.Kom, M.Kom

Disusun Oleh Kelompok 2:

Annisa Nurqalbiyah	**F55124089**
Mia Islamia	**F55124114**
Ramon Pasungke	**F55124115**
Irpandi	**F55124096**

PROGRAM STUDI S1 TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

FAKULTAS TEKNIK

UNIVERSITAS TADULAKO

2025

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sistem informasi modern khususnya dalam domain konten, sangat bergantung pada API (Application Programming Interface) yang terstruktur dan aman. Proyek Movieku dikembangkan sebagai solusi yang membutuhkan pertukaran data secara *real-time* dari sumber eksternal global. Dalam konteks arsitektur perangkat lunak, adopsi prinsip RESTful API menjadi esensial untuk memastikan komunikasi yang efisien dan stateless (Pautasso, 2012). Untuk mencapai standar profesional, proyek ini menggunakan framework AdonisJS 6 dan didukung oleh database MongoDB, yang memfasilitasi implementasi API yang cepat, terdokumentasi, dan berbasis *resource*. Peningkatan signifikan dari *project* sebelumnya adalah penerapan fitur personalisasi (CRUD Film Lengkap) dan integrasi dengan TMDB API, menjadikannya platform yang komprehensif bagi pengguna.

Tuntutan fungsionalitas ini secara langsung memerlukan perhatian serius terhadap aspek keamanan. Untuk melindungi endpoint yang melibatkan data user seperti (Tambah Film atau Edit Film), Movieku mengimplementasikan mekanisme Token-based Authentication menggunakan JWT (JSON Web Token). JWT adalah standar IETF yang menyediakan cara aman dan teruji untuk mentransfer informasi antar pihak sebagai objek JSON (Hardt, 2015). Penggunaan JWT menjamin bahwa setiap request yang dilakukan oleh user telah diotentikasi dan memiliki otorisasi yang valid, sebuah praktik yang sangat penting dalam mengamankan endpoint protected dari akses yang tidak sah, sehingga secara keseluruhan, proyek ini memenuhi tuntutan akademik untuk menguasai desain API, implementasi keamanan, dan integrasi API eksternal sesuai standar industri.

1.2 Rumusan Masalah

1. Bagaimana endpoint CRUD Film Lengkap (Tambah, Edit, Hapus, Lihat) dirancang, diimplementasikan, dan didokumentasikan menggunakan standar OpenAPI/Swagger?
2. Bagaimana mekanisme keamanan JWT Authentication diterapkan pada Movieku untuk otentikasi (login & register user) dan otorisasi akses ke endpoint protected?
3. Bagaimana TMDB API dan Google Client ID diintegrasikan untuk menyediakan rekomendasi, film trending, dan potensi otentikasi pihak ketiga?
4. Bagaimana minimal 5 test case disusun dan divalidasi menggunakan Postman untuk membuktikan keamanan, fungsionalitas CRUD, dan keberhasilan integrasi API eksternal?

1.3 Tujuan Project

1. Mendokumentasikan seluruh endpoint Movieku secara otomatis dan komprehensif menggunakan Swagger API Documentation.
2. Mengimplementasikan JWT Authentication yang menyimpan token di LocalStorage dan menjamin akses ke semua endpoint CRUD yang membutuhkan token.
3. Berhasil mengintegrasikan TMDB API untuk fitur pencarian film, rekomendasi, detail film, dan film trending mingguan.
4. Menyelesaikan CRUD Film lengkap (cari, tambah, edit, hapus) yang memiliki kepemilikan user.

BAB II
LANDASAN KONSEP

2.1 Konsep Pengembangan API Movieku

Movieku merupakan aplikasi berbasis web yang berfungsi untuk mengelola informasi film. Aplikasi ini menggunakan arsitektur RESTful API yang memungkinkan pemisahan antara frontend dan backend sehingga proses komunikasi data dilakukan melalui request dan response berbasis HTTP. Backend Movieku dibangun menggunakan AdonisJS dan terhubung dengan database MongoDB untuk penyimpanan data film. Selain itu, aplikasi juga menyediakan layanan autentikasi pengguna untuk menjaga keamanan data sehingga hanya pengguna yang berhak yang dapat melakukan operasi CRUD terhadap film miliknya.

2.2 Prinsip RESTful API dalam Movieku

Pengembangan API Movieku mengikuti prinsip REST dengan pendekatan resource-oriented, di mana resource utama adalah data film. Setiap endpoint dirancang memiliki fungsi khusus yang direpresentasikan melalui metode HTTP seperti GET untuk membaca data, POST untuk menambah data, PUT untuk mengubah data, dan DELETE untuk menghapus data. Seluruh proses komunikasi bersifat stateless, yang berarti setiap permintaan harus menyertakan informasi autentikasi berupa token agar server dapat memvalidasi hak akses pengguna. Dengan struktur endpoint yang konsisten, API menjadi mudah digunakan oleh frontend maupun integrasi pihak ketiga di masa mendatang.

2.3 Keamanan API dengan JSON Web Token (JWT)

Movieku menerapkan JWT Authentication sebagai mekanisme keamanan utama. Setiap pengguna harus melakukan login untuk mendapatkan token yang kemudian digunakan pada header Authorization ketika mengakses endpoint yang dilindungi, seperti operasi CRUD film. Middleware pada backend bertugas memverifikasi token dan mencocokkannya dengan identitas pengguna. Pendekatan ini efektif untuk mencegah akses ilegal dan memastikan bahwa setiap data film hanya dapat dikelola oleh pemilik akun yang sah.

2.4 Integrasi API Eksternal TMDB

Movieku menyediakan fitur pengambilan film trending melalui integrasi dengan TMDb (The Movie Database). Backend akan melakukan request ke layanan TMDb dan menyesuaikan data yang diterima agar sesuai dengan struktur JSON Movieku. Proses ini dilakukan melalui endpoint /api/movies/tmdb/trending. Keberadaan integrasi eksternal ini meningkatkan kelengkapan informasi yang dapat digunakan oleh pengguna dalam mencari referensi film.

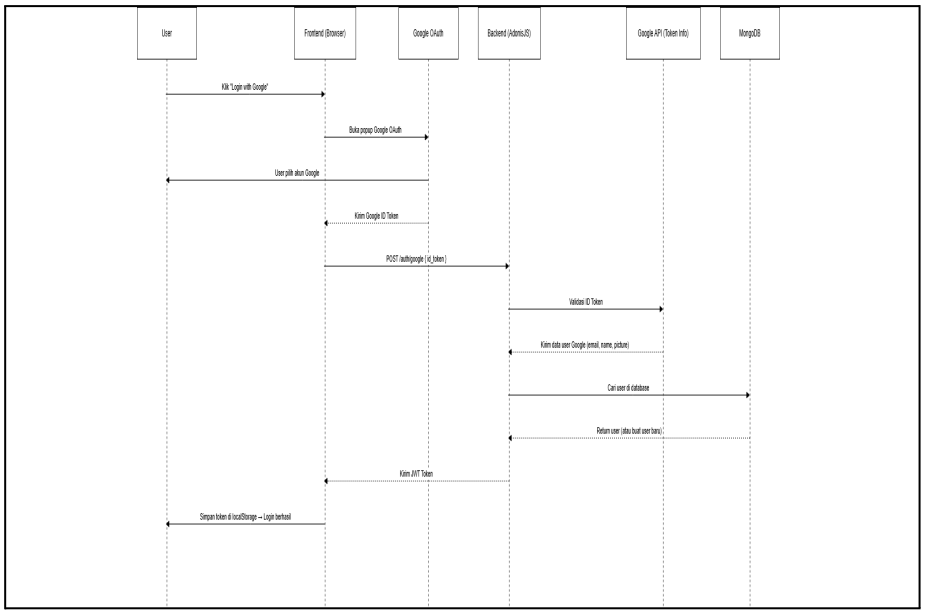
BAB III

DESAIN DAN DOKUMENTASI API

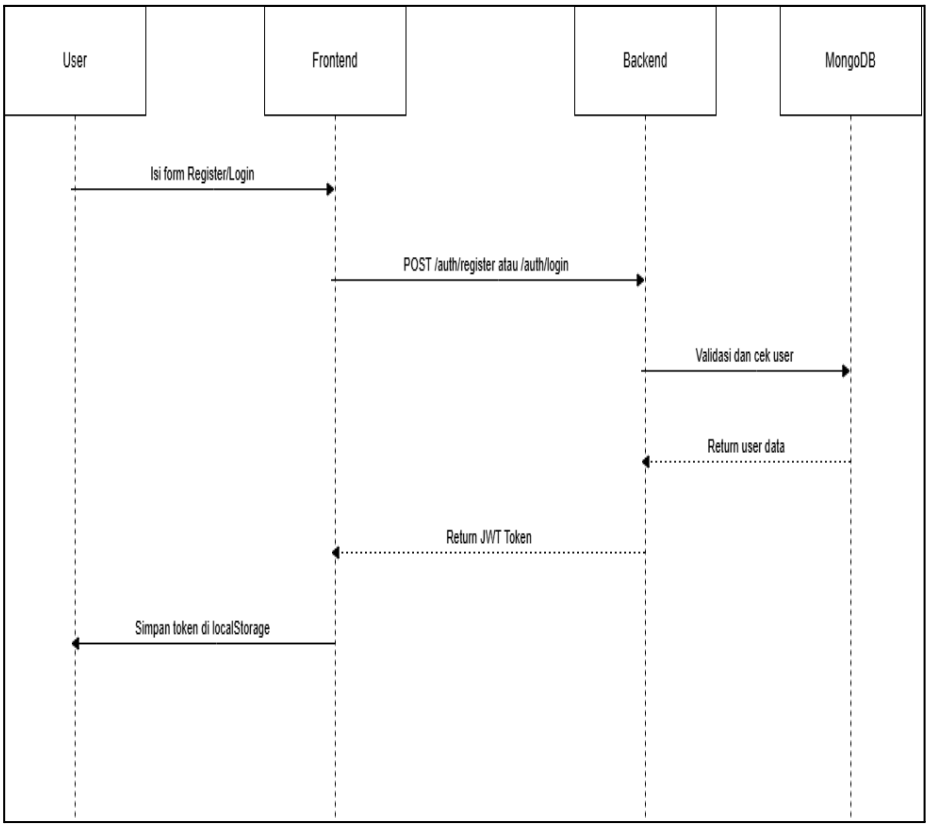
3.1 Perancangan Sistem

3.1.1 Diagram Arsitektur

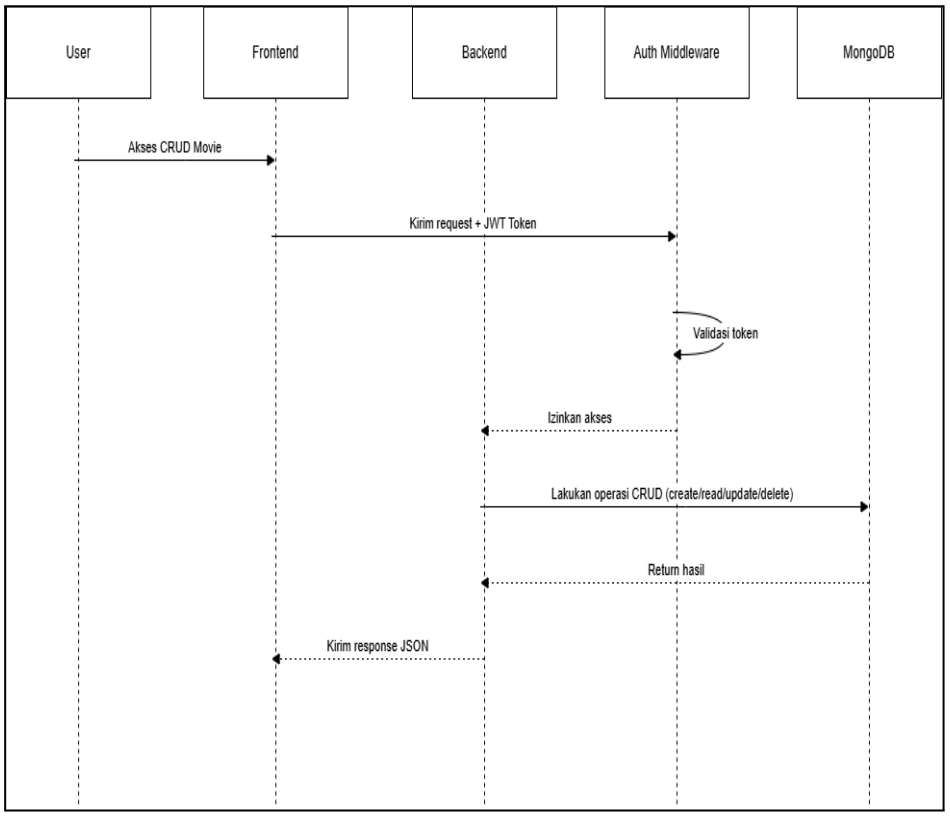
1. Diagram Alur Google Authentication (Google OAuth 2.0)



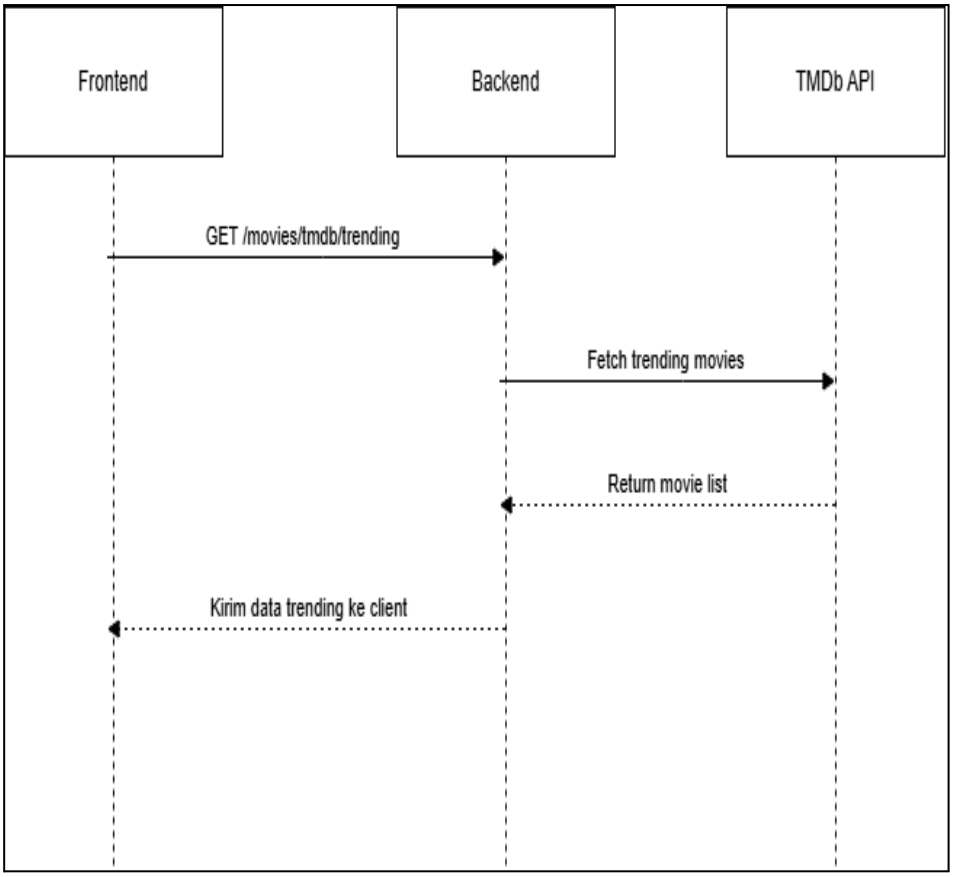
2. Diagram Alur Login/Register Biasa (Email & Password)



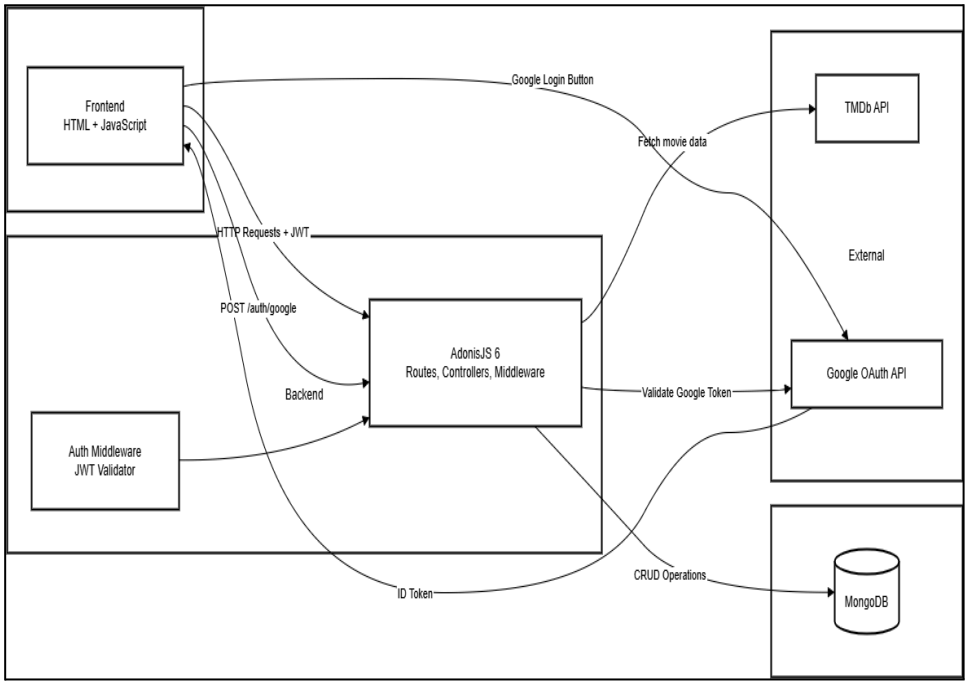
3. Diagram Alur CRUD Movie (Protected Routes)



4. Diagram Alur Integrasi TMDb API (Trending, Search, Detail)



5. Diagram Arsitektur Sistem Movieku (Architecture Overview)



6. Arsitektur API

```
Movieku/
├── app/
│   ├── controllers/
│   │   ├── AuthControllers.ts    # Auth + Google OAuth
│   │   └── MovieControllers.ts   # CRUD + TMDb Integration
│   ├── middleware/
│   │   └── auth.ts               # JWT Middleware
│   └── models/
│       ├── Movie.ts              # Movie Schema
│       └── Users.ts              # User Schema
├── frontend/
│   ├── index.html                # Homepage
│   ├── login.html                # Login Page
│   ├── register.html             # Register Page
│   ├── search.html               # Search Page
│   ├── create.html               # Create Movie
│   ├── edit.html                 # Edit Movie
│   └── delete.html               # Delete Movie
├── docs/
│   ├── openapi.yaml              # API Specification
│   └── swagger.html              # Swagger UI
├── start/
│   ├── routes.ts                 # Route Definitions
│   └── mongo.ts                  # MongoDB Connection
├── .env                          # Environment Variables
└── package.json                  # Dependencies
```

3.1.2 Model Data (Entitas)

Model data menggambarkan struktur informasi yang disimpan dalam sistem. Pada aplikasi Movieku API, terdapat dua entitas utama:

1. **Entitas User**

Nama Atribut	Tipe Data	Keterangan
user_id	ObjectId / PK	Primary key, identitas unik user
username	String	Nama akun user
email	String	Email user, bersifat unik
password	String	Password user yang sudah di-hash
created_at	Date	Tanggal ketika akun dibuat

Entitas User menyimpan data dasar setiap pengguna yang diperlukan untuk proses masuk dan pendaftaran. Data ini juga menjadi acuan bagi sistem untuk mengenali pengguna serta menghubungkannya dengan data lain.

2. **Entitas Movie**

Nama Atribut	Tipe Data	Keterangan
movie_id	ObjectId / PK	Primary key, identitas unik film
title	String	judul film
overview	String	Ringkasan atau deskripsi film
genre	String	Genre film (dapat berupa daftar genre)
release_date	Date	Tanggal rilis film

created_at	Date	Tanggal film ditambahkan ke database

Entitas Movie menyimpan informasi film yang tersedia dalam database Movieku, baik data yang ditambahkan secara manual maupun yang diambil dari TMDb melalui API.

3.2 Dokumentasi API Sesuai Standar

3.2.1 Wajib Menggunakan OpenAPI/Swagger

Untuk memenuhi standar dokumentasi modern, proyek ini menggunakan OpenAPI Specification yang disajikan melalui Swagger UI. Swagger memungkinkan dokumentasi disusun secara otomatis berdasarkan definisi endpoint yang terdaftar pada server, sehingga dokumentasi yang dihasilkan selalu konsisten dengan implementasi aktual. Dokumentasi ini bersifat interaktif dan dapat digunakan untuk melakukan pengujian setiap endpoint tanpa memerlukan aplikasi tambahan.

1. URL Dokumentasi Swagger:
   https://annn214.github.io/Movieku/swagger.html
2. Manfaat Implementasi Swagger:
   a. Visualisasi Struktur API
      Swagger menyajikan daftar endpoint, parameter, jenis request, serta contoh respons dalam bentuk antarmuka grafis yang mudah dipahami.
   b. Validasi dan Pengujian Cepat
      Pengembang dapat melakukan uji coba endpoint secara langsung melalui browser tanpa harus membuka Postman atau membaca kode sumber.
   c. Menjamin Konsistensi Kontrak Data
      Dengan Swagger, setiap perubahan pada struktur API akan diperbarui secara otomatis sehingga frontend dan backend tetap menggunakan definisi data yang sama.
   d. Meminimalkan Kesalahan Integrasi
      Dokumentasi yang jelas mengurangi potensi perbedaan format data antara client dan server.

3.2.2 Deskripsi Detail Endpoint

Seluruh endpoint API pada Movieku menggunakan awalan /api. Semua layanan dirancang mengikuti pola RESTful sehingga mudah dipahami dan

diintegrasikan. Berikut beberapa uraian detail kelompok endpoint yang tersedia dalam sistem.

1. Group Auth (Otentikasi)

   Kelompok endpoint ini digunakan untuk pengelolaan identitas pengguna, termasuk proses registrasi dan login. Endpoint ini bersifat public, namun menghasilkan token untuk mengakses endpoint lain yang bersifat protected.

   a. POST /api/auth/register

      1) Fungsi:

         Mendaftarkan pengguna baru ke dalam sistem.

      2) Input Body:

         a) username

         b) email

         c) password

      3) Keluaran:

         a) Informasi pengguna yang berhasil dibuat.

         b) Status keberhasilan registrasi.

   b. POST /api/auth/login

      1) Fungsi:

         Melakukan proses autentikasi dan menghasilkan token JWT sebagai bukti identitas untuk mengakses endpoint lainnya.

      2) Keluaran:

         a) access_token berupa JWT

         b) Informasi masa berlaku token

         c) Data pengguna terkait

2. Group Movies (Manajemen Film Protected)

   Seluruh endpoint berikut bersifat dilindungi, sehingga hanya dapat diakses apabila permintaan mencantumkan token JWT yang valid. Token harus disertakan pada header HTTP.

   a. GET /api/movies

      1) Fungsi:

         a) Menampilkan daftar film yang telah ditambahkan oleh pengguna.

         b) Menampilkan rekomendasi film atau daftar film populer yang diambil dari API TMDb.

      2) Keluaran:

a) Daftar film dalam format JSON
b) Informasi tambahan dari TMDb jika tersedia

b. GET /api/movies/:id

1) Fungsi:

Menampilkan detail sebuah film berdasarkan ID. Informasi tambahan seperti poster, rating, sinopsis, dan genre dapat diambil dari TMDb.

2) Keluaran:

a) Detail lengkap film
b) Metadata tambahan dari TMDb

c. POST /api/movies

1) Fungsi:

Menambahkan film baru ke dalam database lokal.

2) Keluaran:

a) Data film yang berhasil ditambahkan
b) Status berhasil dan ID film baru

d. PUT /api/movies/:id

1) Fungsi:

Memperbarui data film berdasarkan ID. Aksi ini hanya dapat dilakukan oleh pengguna yang menambahkan film tersebut.

2) Keluaran:

a) Data film yang telah diperbarui
b) Konfirmasi pembaruan berhasil

e. DELETE /api/movies/:id

1) Fungsi:

Menghapus film berdasarkan ID. Hanya pemilik data yang berhak menghapus.

2) Keluaran:

a) Status keberhasilan penghapusan
b) Informasi film yang dihapus

3. Group TMDb Integration

Kelompok endpoint ini digunakan untuk mengambil data film eksternal dari layanan The Movie Database (TMDb) melalui API resmi TMDb.

a. GET /api/movies/tmdb/trending

1) Fungsi:

Mengambil daftar 10 film teratas yang sedang trending secara mingguan berdasarkan data TMDb.

2) Keluaran:
   a) Daftar trending movie
   b) Poster, judul, rating, dan metadata lain dari TMDb

BAB IV
IMPLEMENTASI KEAMANAN DAN INTEGRASI

4.1 Implementasi Keamanan (Token-based Authentication)

4.1.1 Mekanisme pembuatan dan distribusi Token

Implementasi keamanan pada Movieku dimulai dengan proses otentikasi user melalui endpoint POST /api/auth/login. Setelah kredensial diverifikasi, backend AdonisJS menghasilkan JSON Web Token (JWT) yang ditandatangani. Token ini kemudian dikembalikan kepada user dan diatur kadaluarsa dalam 1 hari (JWT_EXPIRES=1d).

Penggunaan JWT didasarkan pada standar teknis IETF yang secara resmi mendefinisikan JWT. Format JWT diakui karena sifatnya yang self-contained dan stateless, yang berarti server dapat memverifikasi integritas dan validitas Token menggunakan kunci rahasia (secret key) tanpa perlu query ke database di setiap request yang dilindungi. Hal ini secara signifikan meningkatkan skalabilitas (scalability) dan performa API (Hardt, 2015).

4.1.2 Penggunaan Token untuk manajemen akses API dasar

Manajemen akses dasar dilakukan dengan mengaktifkan middleware otentikasi pada semua endpoint di bawah resource /movies. Middleware ini mewajibkan Bearer Token pada header Authorization untuk setiap request. Jika Token hilang atau tidak valid, request diblokir dengan response status 401 Unauthorized atau 403 Forbidden.

Penerapan Token-based adalah mekanisme yang efektif untuk menjaga resource API dari akses yang tidak sah. Menurut penelitian Sheth dan Gupta (2018), skema ini sangat efisien karena memisahkan proses otentikasi (memverifikasi siapa user) dari otorisasi (apa yang boleh dilakukan user). JWT, dalam kasus ini, berfungsi sebagai kunci akses yang telah diverifikasi di awal.

4.1.3 Penggunaan Token untuk manajemen akses ke fungsi tertentu (All access)

Pada Movieku, endpoint CRUD fungsional seperti PUT/api/movies/:id dan DELETE /api/movies/:id memerlukan lapisan otorisasi berlapis (sering disebut All access). Selain verifikasi validitas Token secara umum, Movieku menerapkan middleware kustom untuk memverifikasi kepemilikan data. Middleware ini mengekstrak user_id dari payload JWT dan membandingkannya dengan user_id yang terikat pada data film di MongoDB. Dengan demikian, user tidak hanya harus

terotentikasi, tetapi juga terotorisasi secara spesifik untuk memodifikasi resource tersebut, sejalan dengan prinsip least privilege dalam desain keamanan API.

4.2 Integrasi API Eksternal (Minimal 2 Public API)

4.2.1 API Eksternal 1: The Movie Database (TMDB) API

TMDB API merupakan API eksternal utama dan sumber data konten film Movieku. Integrasi ini memungkinkan Movieku menyediakan data film yang dinamis dan up-to-date tanpa perlu memelihara database konten yang besar, seperti menyediakan 10 film trending mingguan melalui endpoint GET /api/movies/tmdb/trending. Penggunaan API seperti TMDB adalah strategi yang efisien untuk memperkaya konten aplikasi, mengurangi beban pemeliharaan data backend. Detail Implementasi TMDB.

Fitur yang Diintegrasikan	Endpoint Internal Movieku	Sumber Data
Film Trending Mingguan	GET /api/movies/tmdb/trending	TMDB API
Detail Film Pelengkap	GET /api/movies/:id	TMDB API
Kunci Akses	TMDB_API_KEY	File .env

4.2.2 API Eksternal 2: Google Client ID (Otentikasi Pihak Ketiga)

Kehadiran kunci konfigurasi GOOGLE_CLIENT_ID dalam lingkungan proyek menandakan kesiapan Movieku untuk mengintegrasikan layanan Google OAuth 2.0. Integrasi ini berfungsi sebagai API eksternal kedua, yang memperluas pilihan otentikasi user melalui Social Login. Integrasi layanan otentikasi eksternal (seperti Google) adalah tren yang berkembang pesat. Sheth dan Gupta (2018) menekankan bahwa delegasi proses otentikasi kepada penyedia identitas terpercaya (seperti Google) dapat meningkatkan kemudahan penggunaan (user experience) sekaligus mengurangi beban keamanan pada backend aplikasi internal. Detail Implementasi Google Client ID.

Tujuan Integrasi	Kunci Konfigurasi	Manfaat
Otentikasi Pihak Ketiga	GOOGLE_CLIENT_ID	Memungkinkan Social Login dan meningkatkan user experience.

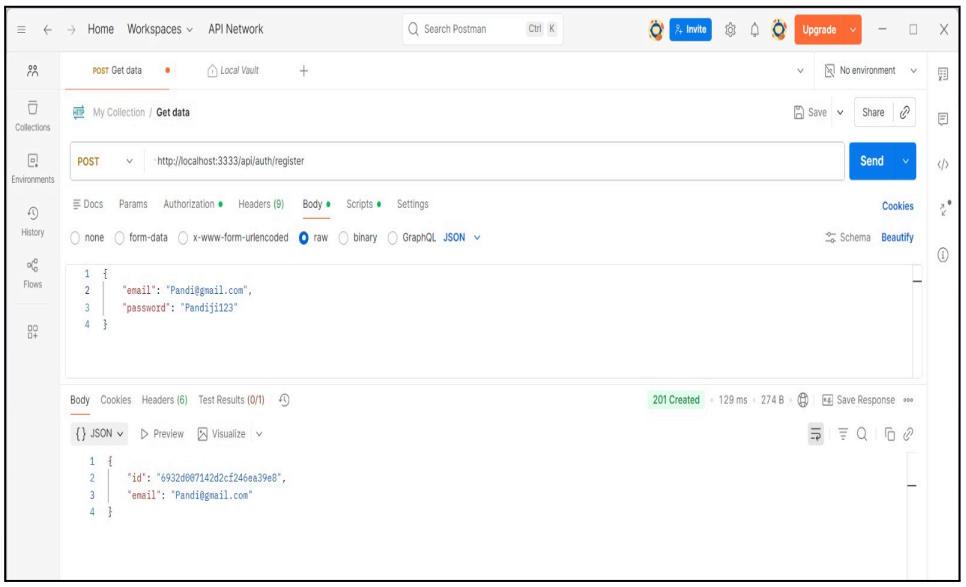
Skema Otorisasi	Google OAuth 2.0	Meringankan beban keamanan backend internal.

BAB V
PENGUJIAN DAN EVALUASI

5.1 Strategi Pengujian

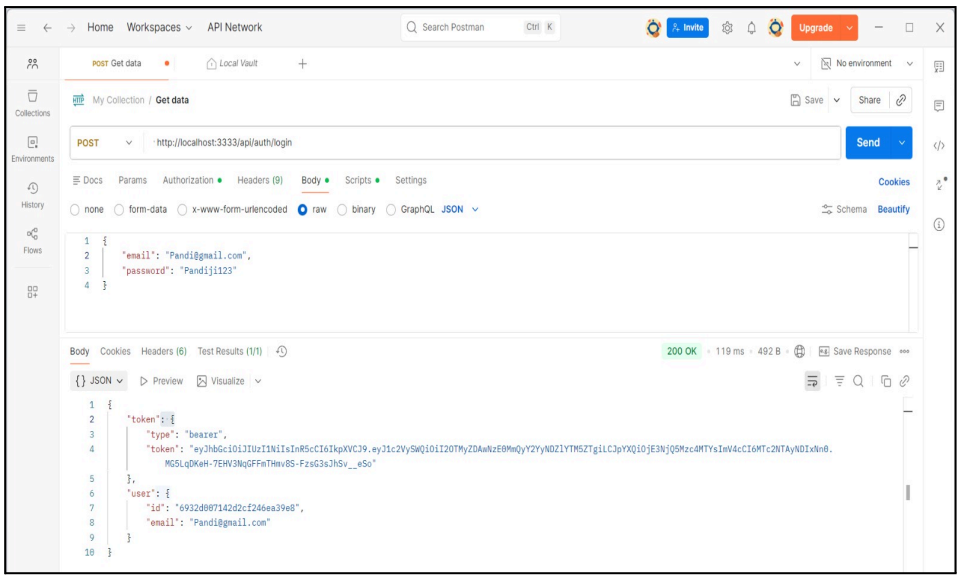
5.1.1 Alat pengujian yang digunakan: Postman

1. POST Registrasi



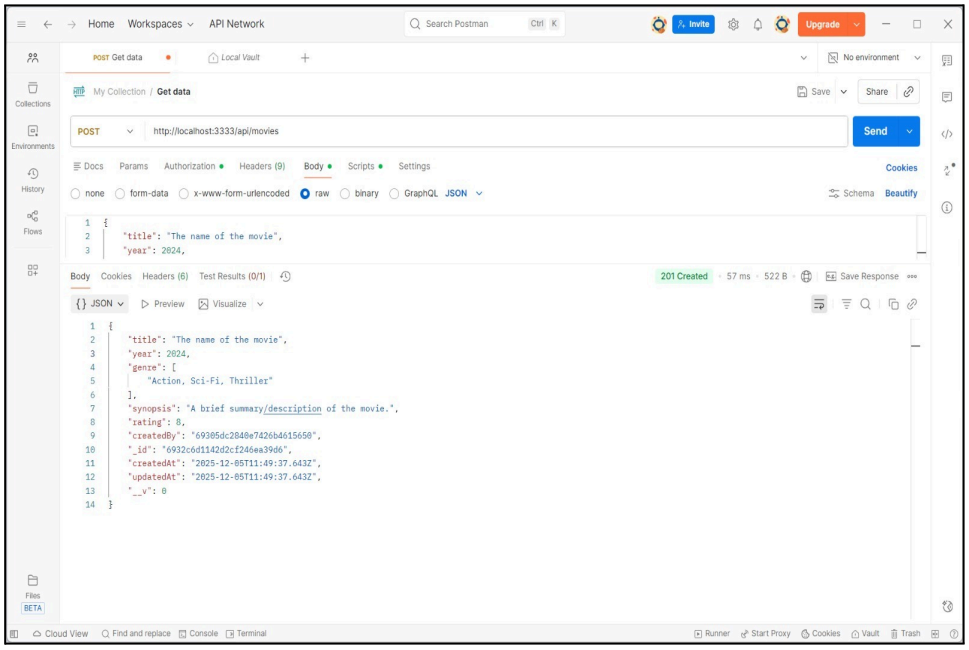
Sistem berhasil memproses pendaftaran akun baru dan menampilkan pesan bahwa registrasi sukses. Endpoint bekerja sesuai fungsi dan data pengguna tersimpan di database.

2. POST Login



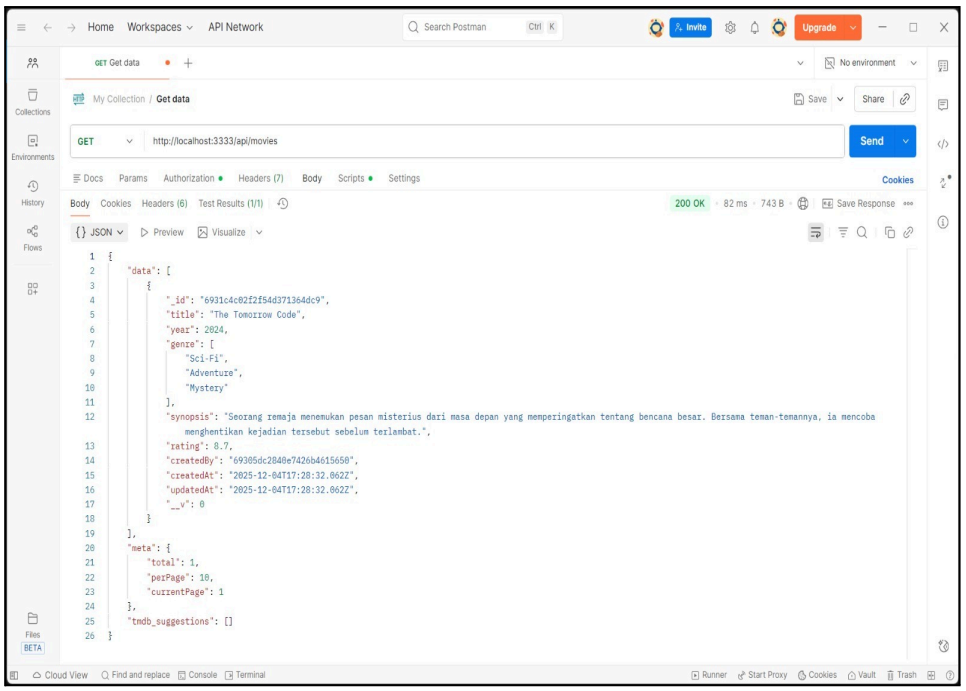
Setelah memasukkan kredensial yang benar, sistem memberikan respons sukses dan menyimpan token JWT. Token ini kemudian digunakan untuk mengakses endpoint lain yang bersifat protected.

3. POST Tambah film



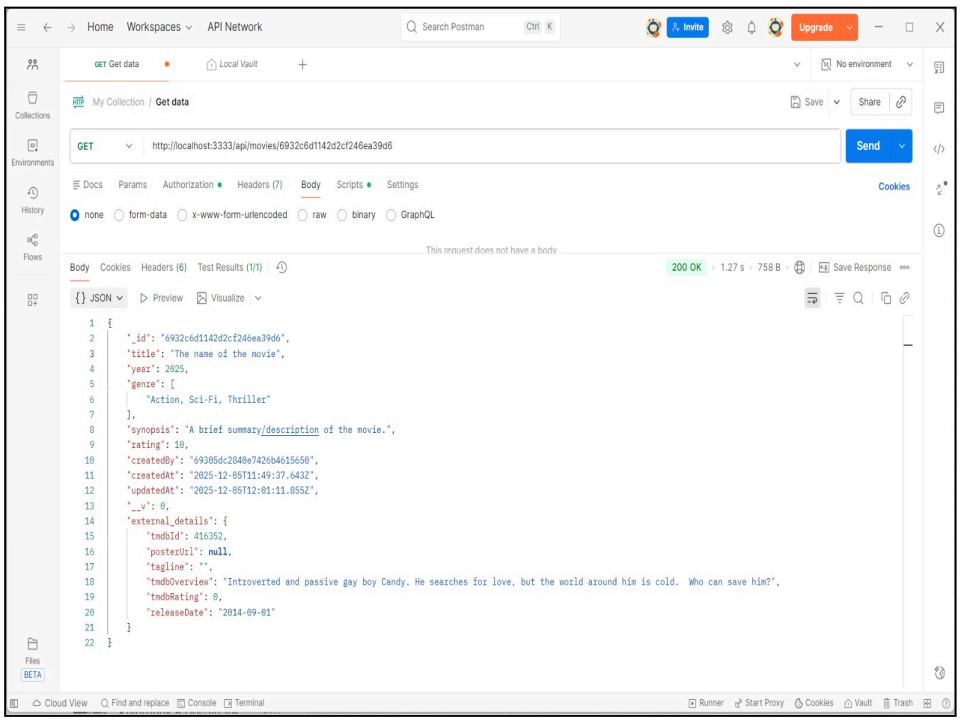
Ketika pengguna mengisi data film lengkap dan mengirimkan token valid, sistem merespons dengan status 201 Created dan data film baru tersimpan dalam database.

4. GET Pencarian film



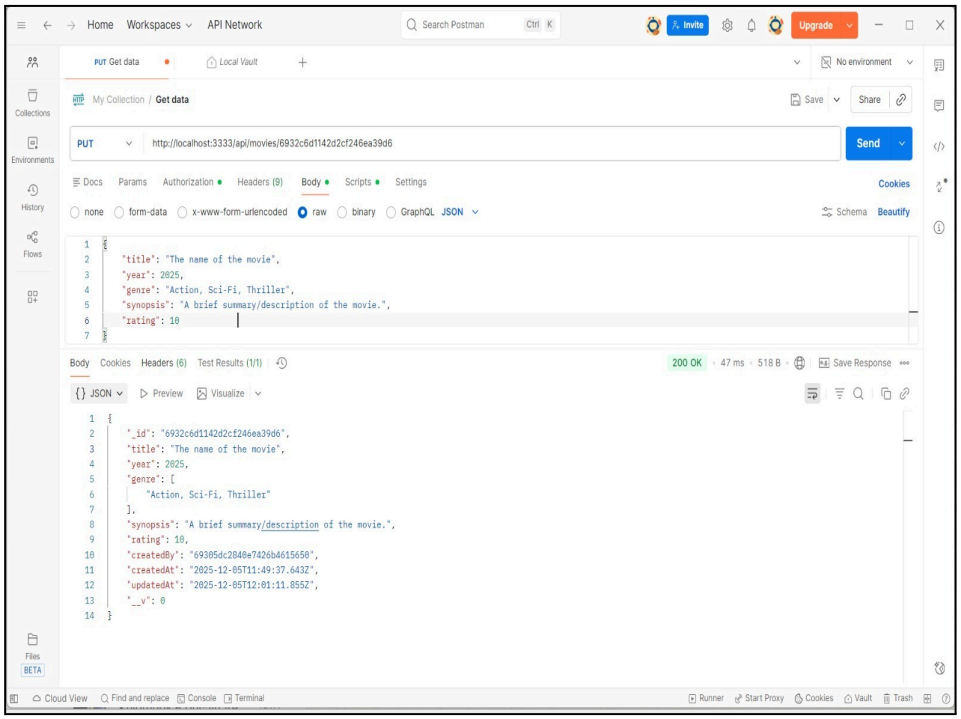
Sistem berhasil mengembalikan daftar film yang tersimpan dengan status 200 OK, termasuk informasi metadata seperti jumlah total data dan halaman aktif. Ini menandakan fitur pencarian film di database berjalan baik.

5. GET Pencarian film berdasarkan id



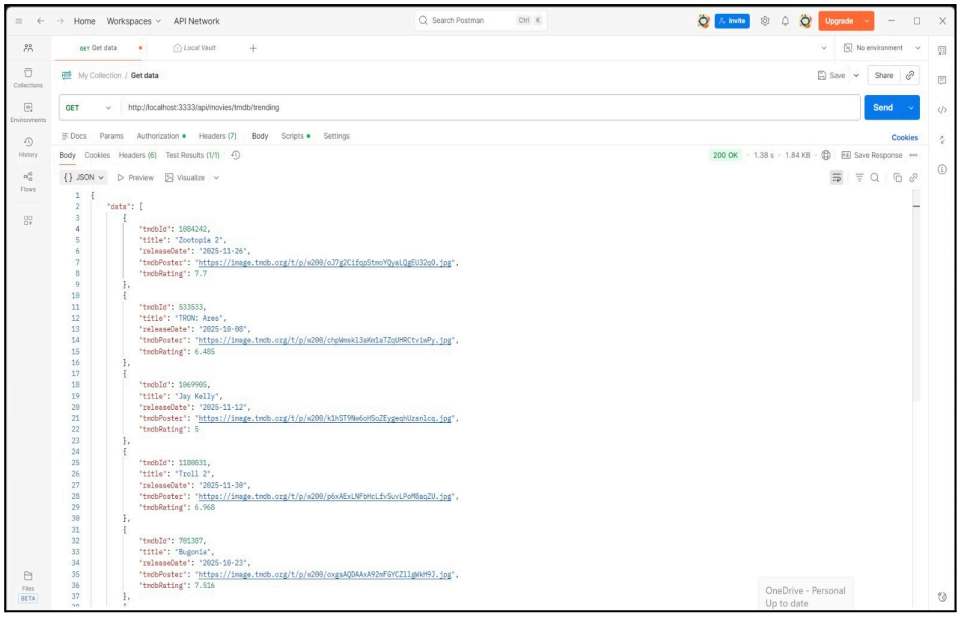
Sistem berhasil menampilkan data lengkap film sesuai ID yang diberikan, termasuk informasi tambahan dari TMDb jika ada. Artinya endpoint mampu melakukan pengambilan data spesifik secara benar.

6. PUT Perbarui film berdasarkan id



Sistem dapat memperbarui data film ketika ID valid dan token diberikan. Respons 200 OK menunjukkan pembaruan berhasil disimpan ke database.

7. GET 10 Film Trending TMDb



API berhasil mengambil dan menampilkan daftar film trending dari TMDb dengan benar. Ini membuktikan integrasi API eksternal berjalan tanpa kendala.

Dalam pengujian ini:

1. Langkah umum:
   a. Menjalankan server backend Movieku (npm run dev / node ace serve --watch)
   b. Mengatur BASE_URL di Postman, misalnya: http://localhost:3333/api.
   c. Melakukan pengujian endpoint satu per satu berdasarkan skenario test case (register → login → simpan token → CRUD film → TMDb).

5.1.2 Tipe pengujian

Tipe pengujian yang diterapkan:

1. Unit-like API Test (Endpoint-level test)
   a. Setiap endpoint diuji secara terpisah:
      1) /auth/register
      2) /auth/login
      3) /movies (GET, POST)
      4) /movies/:id (GET, PUT, DELETE)
      5) /movies/tmdb/trending
   a. Tujuannya memastikan masing-masing endpoint merespons sesuai definisi.
2. Functional / System Test pada alur utama
   a. Menguji alur nyata yang akan dilakukan user:
      1) User baru registrasi → login → mendapat token → menambah film → mengubah film → melihat daftar film → menghapus film.
   b. Memastikan hubungan antar endpoint dalam satu alur tidak bermasalah.
3. Security-related Test (JWT Protection)
   a. Menguji apakah endpoint yang seharusnya *protected* akan menolak akses jika tidak ada header Authorization: Bearer <token>.
   b. Menguji penggunaan token yang tidak valid atau kadaluarsa untuk memastikan API mengembalikan error (misalnya 401 Unauthorized / 403 Forbidden).
4. Integration Test dengan TMDb API

a. Menguji endpoint /movies/tmdb/trending untuk memastikan aplikasi berhasil mengambil data trending dari TMDb dan mengembalikan hasil ke client.

5.2 Laporan Hasil Pengujian (Minimal 5 *Test Case*)

5.1.1 Dokumentasi Hasil Pengujian

1. Test Case Registrasi Pangguna
    a. Pengguna mengisi form registrasi dengan data valid (nama, email, password).
    b. Sistem memproses permintaan dan menampilkan pesan *"Registrasi berhasil! Silakan login."*
    c. Sistem memproses permintaan dan menampilkan pesan *"Registrasi berhasil! Silakan login."*
    d. Hasil pengujian: Berhasil (LULUS) karena sesuai ekspektasi.
2. Test Case Login Pengguna
    a. Pengguna login menggunakan email dan password yang sudah terdaftar.
    b. Sistem mengirim dan menyimpan JWT token ke localStorage.
    c. Muncul pesan *"Login berhasil! Token disimpan."* dan token siap digunakan untuk mengakses endpoint protected.
    d. Hasil pengujian: Berhasil (LULUS) karena token diterima dan otentikasi sukses.
3. Test Case Tambah Film (Fungsional)
    a. Pengguna mengakses halaman tambah film dan memasukkan semua informasi film secara lengkap.
    b. Token valid ikut dikirim saat request *POST /movies*.
    c. Sistem memproses data tanpa error dan menampilkan status pemrosesan.
    d. Hasil pengujian: Berhasil (LULUS) karena data dianggap valid dan diterima oleh server.
4. Test Case Integrasi TMDb
    a. Pengguna menekan tombol *"Muat Tren TMDb"* pada halaman Cari Film.
    b. Sistem berhasil mengambil dan menampilkan 10 film trending dari API TMDb.

c. Tidak terjadi error dalam request ke API eksternal.
d. Hasil pengujian: Berhasil (LULUS) karena integrasi API eksternal berjalan baik.

5. Test Case Edit Film
   a. Pengguna mencoba mengedit film dengan memasukkan ID film yang tidak valid (bukan ObjectId).
   b. Sistem menolak request dan menampilkan pesan error seperti *"Cast to ObjectId failed…"* dan *"Cannot PUT /api/movies/"*.
   c. Tidak ada perubahan pada database.
   d. Hasil pengujian: Berhasil (LULUS) sebagai negative test, karena sistem mampu menolak input yang tidak valid.

BAB VI
PENUTUP

6.1 Kesimpulan

Proyek Movieku telah berhasil dikembangkan sebagai sistem CRUD film berbasis API yang secara signifikan melampaui project dasar sebelumnya. Pencapaian utama proyek ini adalah keberhasilan dalam mengimplementasikan persyaratan standar industri. Sistem keamanan API diimplementasikan melalui JWT Authentication, yang memastikan semua endpoint protected (CRUD Film Lengkap) diamankan dan terkait dengan kepemilikan user. Selain itu, Movieku berhasil memenuhi standar dokumentasi dengan menggunakan Swagger API Documentation, yang menyediakan kontrak API yang jelas dan otomatis untuk semua endpoint internal dan eksternal. Semua fungsionalitas inti telah divalidasi melalui

pelaksanaan minimal 5 test case menggunakan Postman, membuktikan stabilitas dan fungsionalitas sistem.

Secara teknis, Movieku terbukti mampu menangani integrasi dengan sumber data pihak ketiga. Proyek ini berhasil mengintegrasikan TMDB API untuk menyediakan konten dinamis seperti film trending mingguan dan detail film, serta menunjukkan kesiapan untuk integrasi API eksternal kedua melalui konfigurasi Google Client ID. Dengan capaian ini, proyek telah memenuhi semua tuntutan akademik, termasuk implementasi keamanan berbasis Token dan integrasi API eksternal yang disyaratkan. Oleh karena itu, Movieku dapat dikategorikan sebagai API yang siap deploy dan terdokumentasi sesuai dengan praktik terbaik dalam pengembangan perangkat lunak modern.

DAFTAR PUSTAKA

Al-Ameed, S. K., & Al-Shargabi, A. (2020). Challenges and trends in Web API integration. *Journal of Software Engineering and Applications*, *13*(9), 237–251.

Hardt, D. (2015). RFC 7519: JSON Web Token (JWT). *Internet Engineering Task Force (IETF)*. https://www.rfc-editor.org/info/rfc7519

Pautasso, C. (2012). Software architecture for RESTful web services. *Communications of the ACM*, *55*(11), 35–43.

Sheth, A., & Gupta, M. (2018). API security and authentication mechanisms: A survey. *International Journal of Computer Science and Network Security*, *18*(1), 101–107.

Zardari, S., et al. (2015). A study of Web API testing techniques and tools. *International Journal of Electrical and Computer Engineering*, *5*(2), 332–341.