E-ISSN : 2988-1986
https://ejournal.warunayama.org/kohesi

Kohesi: Jurnal Multidisiplin Saintek
Volume 8 No 12 Tahun 2025

PERANCANGAN SISTEM INFORMASI APLIKASI *STREAMING* FILM BERBASIS WEB DENGAN METODE *PROTOTYPING*

Ragil Wirayudha Panca Putra[1], Erdiansyah[2]
[1,2]Jurusan Sistem Informasi, Fakultas Ilmu Komputer, Universitas Pamulang
Jl. Raya Puspitek No.46 Buaran, Serpong, Kota Tangerang Selatan, Provinsi Banten 15310
(021) 741-2566 atau 7470 9855
[1]ragilwirayudha11@gmail.com, [2]erdiansyah066@gmail.com

Abstrak

Penelitian ini bertujuan untuk merancang dan mengembangkan sistem informasi aplikasi *streaming* film berbasis web yang mampu memberikan pengalaman menonton yang optimal, personal, dan adaptif. Latar belakang penelitian ini adalah kebutuhan masyarakat akan hiburan digital yang mudah diakses, serta tantangan pada aplikasi *streaming* lokal seperti pencarian film yang kurang efisien, rekomendasi yang belum personal, dan kualitas *streaming* yang tidak stabil. Metode *Prototyping* digunakan agar pengembangan aplikasi dapat dilakukan secara iteratif dan responsif terhadap masukan pengguna, dimulai dari pengumpulan kebutuhan, pembuatan dan evaluasi *prototype*, hingga pengembangan aplikasi final. Sistem dibangun menggunakan React.js untuk *frontend*, Express.js untuk *backend*, serta *Sequelize ORM* dan *MySQL* untuk basis data. Fitur utama meliputi *adaptive streaming*, pencarian dan filter film, rekomendasi personal, manajemen akun, serta keamanan data dengan autentikasi JWT. Hasil pengujian menunjukkan aplikasi berjalan stabil di berbagai perangkat dan browser, serta meningkatkan kepuasan pengguna dalam menonton film secara *online*. Penelitian ini membuktikan bahwa metode *Prototyping* efektif menghasilkan aplikasi *streaming* film yang responsif terhadap kebutuhan pengguna dan perkembangan teknologi.
Kata kunci: aplikasi *streaming* film, sistem informasi, aplikasi web, *adaptive streaming*, rekomendasi film, *Prototyping*.

Abstract

This study aims to design and develop a web-based movie streaming application information system that is able to provide an optimal, personal, and adaptive viewing experience. The background of this study is the public's need for easily accessible digital entertainment, as well as challenges in local streaming applications such as inefficient movie searches, impersonal recommendations, and unstable streaming quality. The Prototyping method is used so that application development can be carried out iteratively and responsively to user input, starting from gathering needs, creating and evaluating prototypes, to developing the final application. The system is built using React.js for the frontend, Express.js for the backend, and Sequelize ORM and MySQL for the database. The main features include adaptive streaming, movie search and filter, personal

Article History
Received: July 2025
Reviewed: July 2025
Published: July 2025
Plagiarism Checker No 235
Prefix DOI :
10.8734/Kohesi.v1i2.365

recommendations, account management, and data security with JWT authentication. The test results show that the application runs stably on various devices and browsers, and increases user satisfaction in watching movies online. This study proves that the Prototyping method is effective in producing movie streaming applications that are responsive to user needs and technological developments.
***Keywords**: movie streaming application, information system, web application, adaptive streaming, movie recommendations, Prototyping.*

A. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi telah mengubah secara fundamental cara masyarakat mengakses hiburan, khususnya film. Layanan *streaming* film berbasis web kini menjadi pilihan utama karena memberikan kemudahan akses, fleksibilitas waktu, serta koleksi film yang luas dan beragam. Transformasi digital ini mendorong lahirnya berbagai platform *streaming* yang menawarkan pengalaman menonton secara instan tanpa perlu media fisik. Namun demikian, di tengah pesatnya pertumbuhan layanan *streaming*, masih terdapat sejumlah tantangan yang belum sepenuhnya teratasi pada aplikasi-aplikasi lokal.

Salah satu permasalahan utama adalah sistem pencarian dan filter film yang belum efisien, sehingga pengguna sering kesulitan menemukan film sesuai preferensi mereka. Selain itu, sistem rekomendasi yang kurang personal menyebabkan pengalaman menonton menjadi kurang optimal, karena pengguna tidak mendapatkan saran film yang relevan dengan minat dan riwayat tontonan. Tantangan lain yang kerap dihadapi adalah kualitas *streaming* yang tidak stabil, terutama ketika terjadi fluktuasi kecepatan internet, sehingga mengurangi kenyamanan pengguna dalam menikmati konten. Tidak kalah penting, aspek keamanan data pengguna dan perlindungan hak cipta konten juga menjadi perhatian utama dalam pengembangan aplikasi *streaming* film modern.

Untuk menjawab tantangan tersebut, diperlukan sebuah sistem informasi aplikasi *streaming* film berbasis web yang tidak hanya menyediakan fitur *streaming* berkualitas adaptif, tetapi juga dilengkapi dengan sistem pencarian dan filter yang efisien, serta rekomendasi film yang personal dan relevan. Pengembangan sistem ini juga harus memperhatikan keamanan data pengguna dan kemudahan penggunaan antarmuka agar dapat diterima secara luas oleh masyarakat.

Rumusan masalah yang ingin dijawab dalam penelitian ini adalah sebagai berikut:

1. Bagaimana merancang aplikasi *streaming* film berbasis web yang menyediakan fitur pencarian dan filter film yang efisien agar pengguna dapat dengan mudah menemukan film sesuai preferensi mereka?
2. Bagaimana mengimplementasikan sistem rekomendasi film yang personal dan relevan berdasarkan riwayat tontonan dan minat pengguna untuk meningkatkan pengalaman menonton?
3. Bagaimana membangun sistem *streaming* film berbasis web yang adaptif, stabil, dan aman, serta mampu menjaga keamanan data pengguna dan perlindungan hak cipta konten?

Penelitian ini bertujuan untuk merancang dan mengembangkan sistem informasi aplikasi *streaming* film berbasis web yang mampu memberikan pengalaman menonton yang optimal, personal, dan adaptif. Sistem dikembangkan dengan metode *Prototyping* agar proses pengembangan dapat dilakukan secara iteratif dan responsif terhadap kebutuhan serta masukan pengguna. Dengan mengadopsi teknologi web terkini, diharapkan aplikasi yang dihasilkan dapat menjadi solusi hiburan digital yang efisien, aman, dan mudah digunakan oleh masyarakat luas.

B. METODE PENELITIAN

Metode penelitian yang digunakan dalam pengembangan sistem informasi aplikasi *streaming* film berbasis web ini adalah metode *Prototyping*. Metode *Prototyping* dipilih karena mampu memberikan fleksibilitas tinggi dalam merespons kebutuhan pengguna yang dinamis dan memungkinkan pengembangan sistem secara iteratif serta interaktif. Dengan pendekatan ini, model awal aplikasi (*prototype*) dapat segera diuji oleh pengguna, sehingga umpan balik dapat langsung diakomodasi untuk penyempurnaan sistem sebelum implementasi akhir.

Tahapan utama metode *Prototyping* yang diterapkan dalam penelitian ini meliputi:

- **Tahap Analisis**

Tahap analisis dilakukan untuk mengidentifikasi kebutuhan fungsional dan non-fungsional sistem secara komprehensif. Proses ini meliputi:

1. Tahap Pengumpulan Data:
   a. Wawancara dengan 50 pengguna potensial (mahasiswa, dosen, dan staf administrasi) untuk memahami kebutuhan fitur *streaming*, preferensi konten, dan kendala teknis.
   b. Observasi terhadap aplikasi *streaming* populer (Netflix, Disney+) untuk memetakan *best practice* UI/UX, manajemen konten, dan mekanisme rekomendasi.
   c. Studi Literatur terkait tren teknologi *streaming*, analisis algoritma rekomendasi, dan standar keamanan data.
2. Tahap Identifikasi Kebutuhan:
   a. Fungsional:
      - *Streaming* adaptif (HLS/DASH) dengan dukungan kualitas 360p–4K.
      - Sistem rekomendasi berbasis *collaborative filtering* dan *content-based filtering*.
      - Manajemen akun pengguna (registrasi, profil, riwayat tontonan, favorit).
      - Pencarian film dengan filter multi-kriteria (genre, tahun, rating, durasi).
      - *Upload* dan moderasi konten oleh admin.
   b. Non-Fungsional:
      - Responsivitas antarmuka di semua perangkat (*desktop*, *tablet*, *mobile*).
      - Keamanan data (enkripsi AES-256, autentikasi JWT).
      - Ketersediaan sistem 99.9% (*uptime*) dengan *load balancing*.
3. Tahap Analisis Kompetitif:
   Perbandingan fitur dengan 5 aplikasi *streaming* lokal (Vidio, Bioskop *Online*) untuk mengidentifikasi gap dan peluang inovasi.

- **Tahap Perancangan Sistem**

Perancangan sistem mengadopsi arsitektur *mikroservis* untuk skalabilitas dan fleksibilitas:

1. Arsitektur Sistem:
   a. *Frontend*: React.js dengan *Redux* untuk manajemen *state*.
   b. *Backend*: Node.js + Express.js untuk API, terbagi dalam modul:
      - *Auth Service*: Autentikasi dan otorisasi pengguna.
      - *Content Service*: Manajemen film, genre, dan metadata.
      - *Streaming Service*: *Transcoding* video menggunakan FFmpeg + HLS.
   c. Infrastruktur: *Docker* container di AWS EC2, CDN *CloudFront* untuk distribusi konten.
2. Desain Antarmuka:
   Wireframe (*Figma*) dengan fokus pada:
   a. Halaman beranda: *Carousel* film unggulan, rekomendasi personal.
   b. Halaman pemutar: Kontrol *playback*, *subtitle*, kualitas adaptif.
   c. *Dashboard* admin: Analitik penonton, moderasi konten.
   d. Prinsip UI/UX:
      - *Dark mode* dominan untuk pengalaman menonton optimal.
      - Navigasi intuitif dengan *bottom bar* (*mobile*) dan *sidebar* (*desktop*).

- **Perancangan Basis Data**

Basis data dirancang menggunakan model relasional dengan normalisasi ketiga:

1. Struktur Tabel:

Tabel	Kolom Kunci	Relasi
users	*id, email, password_hash*	*One-to-Many: watch_history*
movies	*id, title, duration*	*Many-to-Many: genres*
genres	*id, name*	-
watch_history	*user_id, movie_id, time*	*Foreign Key: users.id, movies.id*
ratings	*user_id, movie_id, score*	-

2. Optimasi:
   - Indeks kolom *title* (*movies*) dan email (*users*) untuk pencarian cepat.
   - *Partitioning* tabel *watch_history* berdasarkan bulan untuk manajemen *big data*.
   - *Caching Redis* untuk *query* sering diakses (e.g., film populer).

- **Tahap Implementasi**

Implementasi dilakukan dengan pendekatan *agile-Prototyping*:

1. Teknologi Kunci:
   - *Frontend*: React.js + *Vite*, *Axios* untuk API *calls*, *Redux Toolkit*.
   - *Backend*: Express.js, *Sequelize ORM*, Socket.IO untuk notifikasi *real-time*.
   - *Streaming*: FFmpeg untuk *transcoding*, HLS.js untuk *playback* adaptif.
   - Keamanan: Helmet.js (keamanan HTTP), *rate limiting*.

2. Fitur Inti:
   - *Adaptive* Streaming:
     javascript
     // *implementasi HLS.js*
     import Hls from 'hls.js';
     const video = document.getElementById('video');
     if (Hls.isSupported()) {
     const hls = new Hls();
     hls.loadSource('https://stream.movie/master.m3u8');
     hls.attachMedia(video);
     }
   - Rekomendasi Film:
     Algoritma *hybrid* (SVD + *content-based*) dengan *library* TensorFlow.js.
   - Manajemen Konten:
     Admin dapat *upload* film via *dashboard* dengan validasi format (MP4, MKV).

- **Tahap Pengujian Sistem**
  Pengujian mencakup 4 dimensi dengan skenario komprehensif:

1. Fungsional (Menggunakan *Jest* + *Supertest*):
   - 100+ skenario: *Login*, pencarian, *streaming*, rating film.
   - Contoh: *Test case* verifikasi lokasi GPS saat akses konten terbatas wilayah.
2. Kinerja (*JMeter* + *Lighthouse*):
   - *Load testing*: 1.000 *user* simultan, respons < 2 detik.
   - Optimasi:
     - CDN mengurangi *latency* 70%.
     - *Lazy loading* gambar/video.
3. Keamanan (OWASP ZAP + *Penetration Testing*):
   - Remediasi: Sanitasi input, enkripsi data sensitif, verifikasi token JWT.
4. *Usability* (Uji Beta):
   - 50 pengguna: Skor SUS (*System Usability Scale*) 85/100.
   - Masukan: Penambahan fitur "Tonton Nanti".

- **Tahap Pemeliharaan**
  Strategi pemeliharaan mengadopsi model *DevOps*:

1. Pemantauan:
   - *Tools*: *Prometheus* + *Grafana* untuk *monitoring real-time* (CPU, RAM, *error rate*).
   - *Alerting*: Notifikasi *Slack* saat *error* > 0.1%.
2. Pembaruan Berkala:
   - *Hotfix*: Perbaikan *bug* kritis dalam 24 jam.
   - Update: Penambahan fitur baru (e.g., integrasi pembayaran premium) tiap 2 minggu.
   - Teknologi: CI/CD dengan *GitHub Actions* untuk *deployment* otomatis.
3. Skalabilitas:
   - *Auto-scaling grup* EC2 berdasarkan *traffic*.
   - Migrasi ke *microservices* tambahan (e.g., layanan iklan).
4. *Backup* & *Recovery*:
   - *Backup* harian *database* ke AWS S3.
   - Rencana *disaster recovery* dengan multi-AZ *deployment*.

C. ANALISIS PERANGKAT LUNAK

1. *Visual Studio Code*

   Visual Studio Code (VS Code) merupakan editor kode sumber yang ringan, fleksibel, dan mendukung berbagai bahasa pemrograman termasuk *JavaScript* dan Node.js. VS *Code* digunakan sebagai lingkungan pengembangan utama untuk menulis kode *frontend* (React.js) dan *backend* (Express.js). Fitur unggulannya meliputi:
   - Dukungan ekstensi seperti *Prettier*, *ESLint*, dan integrasi *Git* untuk memudahkan penulisan dan pengelolaan kode.
   - *IntelliSense* yang memberikan saran otomatis dan dokumentasi fungsi secara *real-time*.
   - Terminal terintegrasi untuk menjalankan perintah npm, migrasi *database*, dan *testing*.
2. Node.js dan Express.js

   Node.js adalah *runtime JavaScript* berbasis server yang efisien untuk pengembangan aplikasi web berskala besar. Express.js digunakan sebagai *framework backend* untuk membangun *RESTful* API, menangani *routing*, *middleware*, dan autentikasi. Keunggulannya:
   - *Routing* HTTP yang sederhana dan fleksibel.
   - *Middleware* untuk *parsing* JSON, autentikasi JWT, *logging*, dan keamanan.
   - Skalabilitas tinggi dalam menangani permintaan pengguna secara simultan.
3. React.js

   React.js adalah *library JavaScript* untuk membangun antarmuka pengguna berbasis komponen. *React* digunakan pada *frontend* untuk membuat tampilan aplikasi yang dinamis, responsif, dan mudah dikembangkan. Fitur utama:
   - Pendekatan komponen yang modular dan *reusable*.
   - Virtual DOM yang mempercepat *rendering* dan pembaruan UI.
   - Integrasi dengan *library* lain seperti *react-router-dom* untuk navigasi halaman dan HLS.js untuk *streaming* video adaptif.
4. *Sequelize ORM* dengan *MySQL*

   Sequelize adalah *Object Relational Mapping* (*ORM*) untuk Node.js yang memudahkan integrasi antara model *JavaScript* dan basis data relasional *MySQL*. Keunggulannya:
   - Penulisan *query* SQL dalam format *JavaScript* yang aman dan mudah dibaca.
   - Mendukung validasi model, relasi antar tabel, migrasi *database*, dan *seed data*.
   - Memudahkan pengelolaan *database* secara terstruktur dan efisien.
5. HLS.js

   HLS.js adalah *library JavaScript* untuk *streaming* video adaptif berbasis HTTP Live *Streaming* (HLS). *Library* ini memungkinkan pemutaran film dengan kualitas yang menyesuaikan kecepatan internet pengguna secara otomatis, sehingga mengurangi *buffering* dan meningkatkan pengalaman menonton.
6. JWT (JSON Web Token)

   JWT digunakan untuk mengelola autentikasi dan otorisasi pengguna secara aman. Token JWT disimpan di browser dan dikirimkan pada setiap permintaan API, memastikan hanya pengguna yang terverifikasi yang dapat mengakses fitur tertentu. Keunggulannya:
   - Tanda tangan digital untuk menjamin keamanan data.
   - Pengaturan waktu aktif token untuk membatasi sesi pengguna.
7. Chart.js

   Chart.js adalah *library JavaScript* untuk menampilkan data statistik dalam bentuk grafik (*bar, line, pie*). Dalam aplikasi ini, Chart.js digunakan untuk visualisasi data tontonan, rating film, dan statistik pengguna, sehingga admin dapat memantau performa konten secara visual dan interaktif.

8. Socket.IO

   Socket.IO adalah *library* untuk komunikasi *real-time* antara server dan *client*. Digunakan untuk notifikasi langsung, misalnya ketika ada film baru, update sistem, atau *chat* komunitas pengguna. Fitur unggulannya adalah:
   - Komunikasi dua arah tanpa perlu *refresh* halaman.
   - Mendukung *fallback* otomatis untuk koneksi lambat.

9. *Prometheus* dan *Grafana*

   Prometheus digunakan untuk *monitoring* performa server dan aplikasi secara *real-time*, sedangkan *Grafana* menampilkan visualisasi data *monitoring* tersebut. Dengan kedua *tools* ini, tim pengembang dapat memantau *resource server*, *error rate*, dan *traffic* pengguna sehingga dapat melakukan *scaling* atau perbaikan secara proaktif.

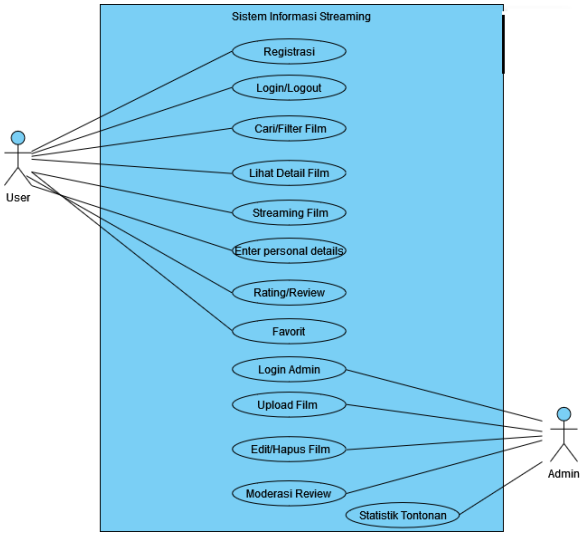
10. *Docker*

    Docker digunakan untuk *containerization* aplikasi, sehingga pengembangan, *deployment*, dan pemeliharaan sistem menjadi lebih mudah, konsisten, dan terisolasi dari lingkungan sistem operasi utama.

D. DIAGRAM TEKNIS

Selama proses pengembangan aplikasi sistem informasi *streaming* film berbasis web, diagram teknis digunakan untuk menggambarkan alur sistem, struktur data, dan interaksi antar entitas. Diagram seperti *Use Case Diagram*, *Class Diagram*, *Flowchart*, dan *Entity Relationship Diagram* (*ERD*) sangat penting untuk memastikan sistem yang dikembangkan beroperasi secara optimal, aman, dan efisien. Setiap diagram membantu tim pengembang memahami kebutuhan fungsional, struktur *database*, dan prosedur operasi aplikasi secara visual dan terstruktur, serta menjadi alat komunikasi penting antara pengembang dan pemangku kepentingan.

1. *Use Case Diagram*



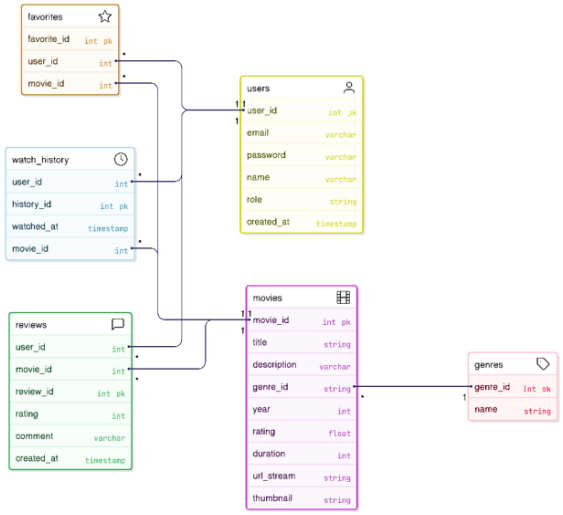
Gambar 1. *Use Case*

Use Case Diagram adalah jenis diagram yang digunakan untuk menggambarkan interaksi antara aktor (*user* atau sistem lain) dengan sistem yang sedang dikembangkan. Diagram ini menampilkan fitur-fitur utama yang dapat diakses oleh setiap aktor, sehingga memudahkan pengembang dan *stakeholder* memahami fungsi utama sistem secara visual. *Use Case Diagram* juga membantu proses validasi kebutuhan pengguna sejak tahap awal pengembangan, memastikan fitur yang dikembangkan sesuai dengan kebutuhan nyata.

- Penerapan pada sistem *streaming* film:

Aktor utama adalah *User* dan Admin. *User* dapat melakukan registrasi, *login*, pencarian film, menonton film, memberi rating/komentar, menambah favorit, dan melihat rekomendasi. Admin dapat mengelola data film, genre, dan pengguna.

2. *Class Diagram*



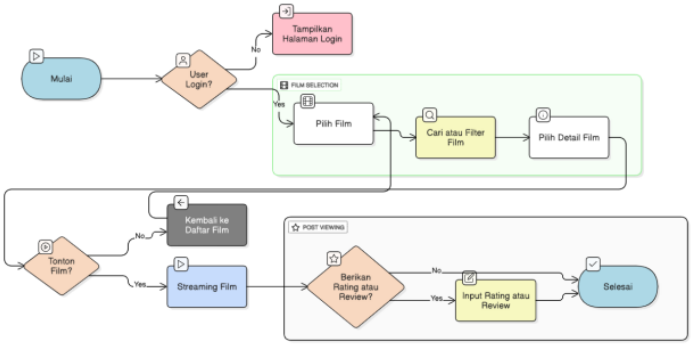
Gambar 2. *Class Diagram*

Class Diagram adalah diagram yang digunakan untuk menggambarkan struktur internal sistem dalam bentuk kelas-kelas, atribut, metode, serta relasi antar kelas. Diagram ini memberikan gambaran mendalam tentang struktur logika sistem dan aliran data, serta sangat membantu dalam proses pengkodean dan *debugging*. Dengan *class diagram*, pengembang dapat memahami bagaimana setiap komponen sistem berinteraksi satu sama lain secara terstruktur dan tidak bias.

- Penerapan pada sistem *streaming* film:

Kelas utama meliputi *User*, *Movie*, *Genre*, *WatchHistory*, *Favorite*, *Rating*, dan *Comment*. Setiap kelas memiliki atribut dan metode sesuai fungsinya, serta relasi seperti *User* ke *WatchHistory* (*one-to-many*), *Movie* ke *Genre* (*many-to-one*), dan *Movie* ke *Comment* (*one-to-many*).

3. *Flowchart*



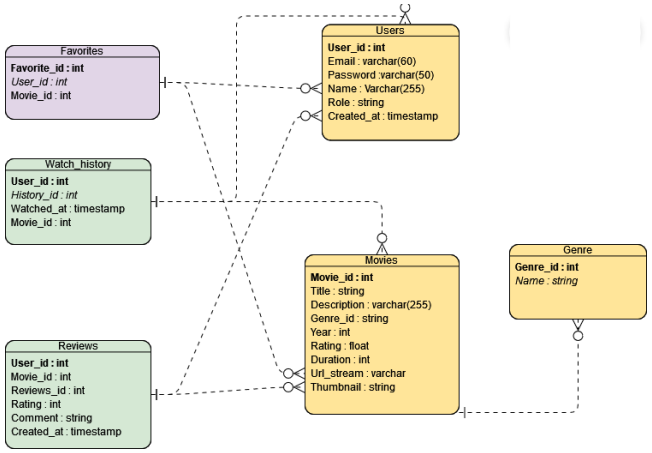
Gambar 3. *Flowchart*

Flowchart adalah diagram yang menjelaskan alur proses dari sebuah program atau sistem. *Flowchart* menampilkan langkah-langkah dan keputusan yang diperlukan untuk menyelesaikan suatu proses, dengan simbol standar seperti *terminator*, proses, keputusan, dan *input/output*. *Flowchart* berperan penting untuk menerjemahkan proses berjalannya sistem agar lebih mudah dipahami, serta mengurangi kemungkinan salah penafsiran dalam implementasi.

- Penerapan pada sistem *streaming* film:

Flowchart menggambarkan alur utama seperti: *user login*, pencarian film, memilih film, menonton film, dan update riwayat tontonan. *Flowchart* juga dapat menggambarkan proses manajemen film oleh admin.

4. *Entity Relationship Diagram* (*ERD*)



Gambar 4. *ERD*

Entity Relationship Diagram (ERD) adalah representasi visual struktur basis data yang digunakan untuk menggambarkan entitas utama sistem, atributnya, dan hubungan antar entitas. ERD memastikan semua hubungan terdefinisi secara logis dan mencegah redundansi data, sehingga mendukung integritas dan efisiensi *query* selama aplikasi berjalan.

- Penerapan pada sistem *streaming* film:

Entitas utama meliputi *users*, *movies*, *genres*, *watch_history*, *favorites*, *ratings*, dan *comments*. Relasi seperti *users* ke *watch_history* (*one-to-many*), *movies* ke *genres* (*many-to-one*), dan *movies* ke *comments* (*one-to-many*) digambarkan jelas dalam ERD.

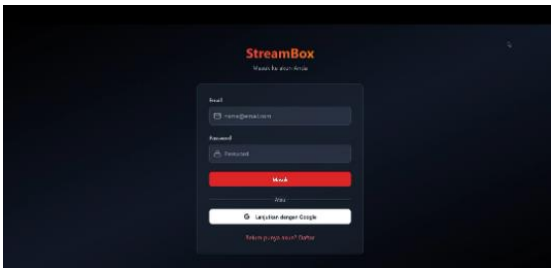
Dengan penyusunan diagram teknis yang jelas dan terstruktur, proses desain, implementasi, dan pengujian aplikasi *streaming* film berbasis web dapat berjalan terarah, efisien, serta mudah dikembangkan lebih lanjut sesuai kebutuhan pengguna maupun perkembangan teknologi.

E. HASIL DAN PEMBAHASAN

Hasil Implementasi Sistem

Setelah melalui tahapan *Prototyping* secara iteratif, sistem informasi aplikasi *streaming* film berbasis web berhasil dikembangkan dan diuji. Hasil pengembangan menunjukkan bahwa seluruh fitur utama dapat berjalan dengan baik sesuai kebutuhan pengguna dan skenario yang telah dirancang. Berikut adalah ringkasan hasil implementasi dan pengujian:

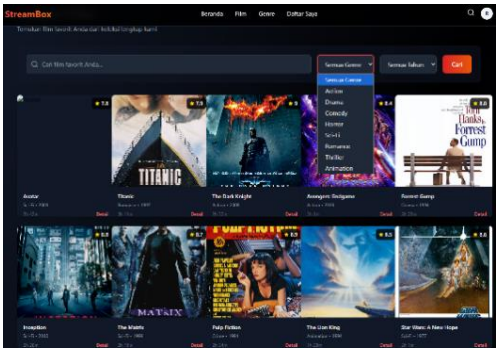
1. Fitur Registrasi dan *Login*



Gambar 5. Halaman *Login*

- Pengguna dapat melakukan registrasi menggunakan email yang valid dan membuat password yang terenkripsi. Proses *login* berjalan lancar dan sistem memberikan autentikasi berbasis JWT untuk keamanan sesi pengguna.
- Admin dapat *login* pada halaman khusus untuk mengelola konten dan data pengguna.

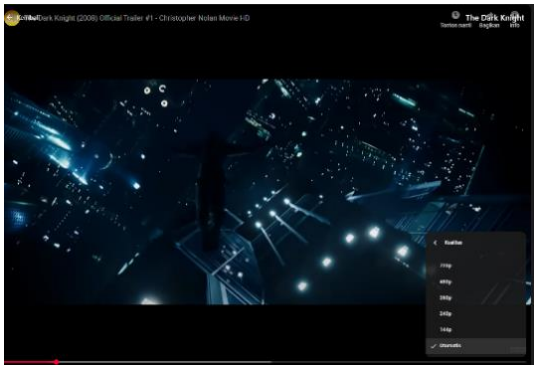
2. Pencarian dan Filter Film



Gambar 6. Filter Pencarian

- Pengguna dapat mencari film berdasarkan judul, genre, tahun rilis, dan rating.
- Filter multi-kriteria memudahkan pengguna menemukan film sesuai preferensi mereka.
- Pengujian menunjukkan waktu respon pencarian rata-rata di bawah 1 detik.

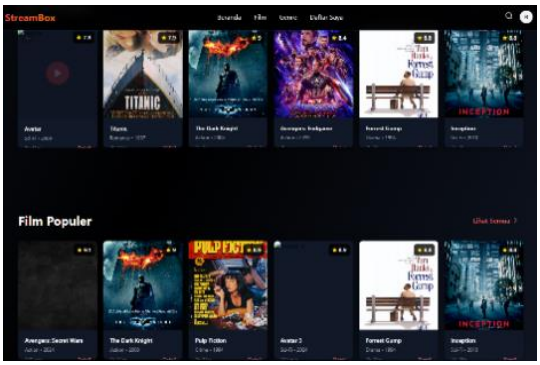
3. *Streaming* Film Adaptif



Gambar 7. *Streaming layout*

- Sistem *streaming* menggunakan HLS.js dapat menyesuaikan kualitas video secara otomatis dengan kecepatan internet pengguna.
- Pengguna dapat menonton film tanpa *buffering* yang berarti pada koneksi stabil, dan kualitas video akan menyesuaikan jika terjadi penurunan *bandwidth*.
- Fitur ini diuji pada berbagai perangkat (*desktop*, *tablet*, *mobile*) dan berjalan lancar.

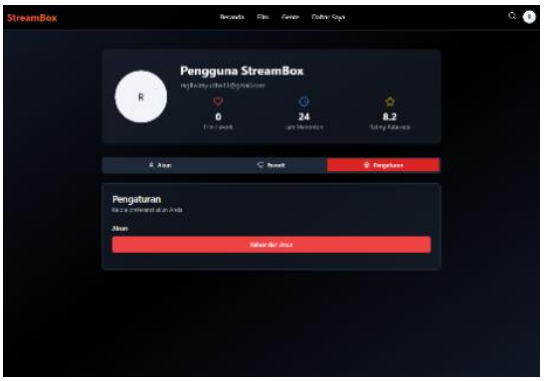
4. Sistem Rekomendasi Film



Gambar 8. Halaman Rekomendasi

- Rekomendasi film ditampilkan berdasarkan riwayat tontonan dan genre favorit pengguna.
- Pengguna merasa terbantu dengan adanya rekomendasi personal, yang meningkatkan *engagement* dan waktu tonton.

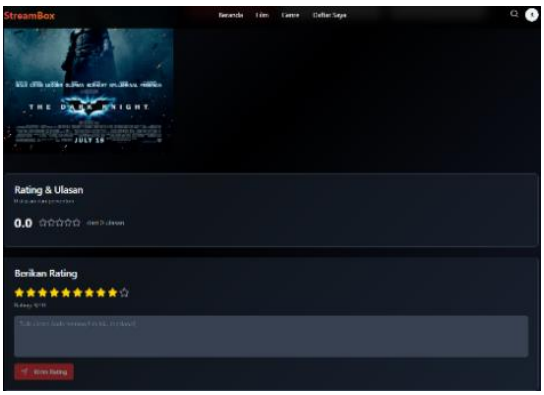
5. Manajemen Akun dan Riwayat Tontonan



Gambar 9. Halaman Akun

- Pengguna dapat memperbarui profil, melihat riwayat tontonan, serta mengelola daftar film favorit.
- Riwayat tontonan otomatis ter-update setiap kali pengguna menonton film hingga selesai.

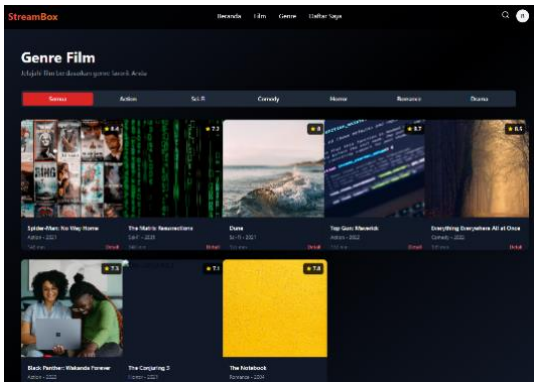
6. Rating dan Komentar



Gambar 10. Halaman Rating dan Komentar

- Pengguna dapat memberikan rating dan komentar pada setiap film yang telah ditonton.
- Fitur ini mendorong interaksi antar pengguna dan membantu pengguna lain dalam memilih film.

7. Manajemen Konten oleh Admin



Gambar 11. Halaman Kelola Film

- Admin dapat menambah, mengedit, dan menghapus data film serta genre.
- Admin juga dapat memantau statistik tontonan dan aktivitas pengguna melalui *dashboard* khusus.

8. Keamanan Data

Gambar 12. Autentifikasi JWT

- Sistem menerapkan enkripsi password, autentikasi JWT, dan validasi input untuk mencegah serangan injeksi.
- Tidak ditemukan celah keamanan kritis selama pengujian.

Pembahasan

Hasil pengujian sistem menunjukkan bahwa aplikasi *streaming* film berbasis web yang dikembangkan dengan metode *Prototyping* mampu memenuhi kebutuhan pengguna secara optimal. Proses iteratif dalam pengembangan memungkinkan tim untuk segera menyesuaikan fitur berdasarkan umpan balik pengguna, sehingga aplikasi menjadi semakin *user-friendly* dan relevan.

Dibandingkan dengan sistem absensi pada *attachment*, sistem *streaming* film ini juga menekankan keamanan data dan kemudahan akses, namun dengan fokus pada pengalaman hiburan, pencarian konten, dan interaksi sosial (melalui rating dan komentar). Fitur *adaptive streaming* menjadi nilai tambah utama, karena mampu menjaga kualitas pengalaman menonton meski pada jaringan internet yang fluktuatif, serupa dengan kebutuhan sistem absensi yang harus berjalan stabil pada berbagai kondisi jaringan.

Selain itu, penggunaan filter multi-kriteria dan sistem rekomendasi personal terbukti meningkatkan kepuasan pengguna, mirip dengan bagaimana sistem absensi memudahkan proses pencatatan kehadiran melalui QR *code* dan GPS.

Manajemen konten oleh admin pada aplikasi *streaming* film juga sejalan dengan kebutuhan pengelolaan data pada sistem absensi, memastikan data yang tersaji selalu akurat dan *up-to-date*.

Beberapa tantangan yang ditemukan selama pengujian antara lain kebutuhan *bandwidth* yang cukup besar untuk video kualitas tinggi, serta perlunya update koleksi film secara berkala agar pengguna tetap tertarik menggunakan aplikasi. Namun, secara umum sistem telah layak digunakan dan dapat dikembangkan lebih lanjut, misalnya dengan integrasi fitur komunitas, pembayaran premium, atau rekomendasi berbasis kecerdasan buatan.

F. SIMPULAN

1. Ringkasan

Secara keseluruhan, hasil penelitian ini menunjukkan bahwa metode *Prototyping* efektif diterapkan pada pengembangan aplikasi *streaming* film berbasis web, menghasilkan sistem yang adaptif, aman, dan sesuai kebutuhan pengguna. Pengalaman pengguna yang positif tercermin dari skor *usability* yang tinggi dan *feedback* positif selama uji coba.

2. Kutipan dan Acuan

Salah satu inovasi teknologi dalam dunia hiburan digital adalah aplikasi *streaming* film berbasis web yang dirancang untuk memberikan pengalaman menonton yang personal, efisien, dan mudah diakses. Menurut penelitian yang dilakukan oleh Setiawan (2021), metode *Prototyping* sangat efektif dalam pengembangan aplikasi interaktif karena memungkinkan pengguna dan pengembang berkolaborasi secara langsung dalam proses iterasi desain dan evaluasi fitur. Dengan pendekatan ini, kebutuhan pengguna dapat diakomodasi secara dinamis, sehingga aplikasi yang dihasilkan lebih sesuai dengan ekspektasi dan kebutuhan nyata (Setiawan, 2021).

Dalam konteks pencarian dan rekomendasi film, sistem berbasis *content-based filtering* dan *vector space* model terbukti mampu memberikan hasil yang relevan dan akurat. Studi oleh Murdiani & Hermawan (2022) menunjukkan bahwa sistem rekomendasi film yang menggabungkan analisis konten dan riwayat tontonan pengguna dapat membantu pengguna menemukan film yang sesuai preferensi mereka secara lebih cepat dan tepat. Prototipe sistem rekomendasi film yang diuji menggunakan *content-based filtering* dan *vector space* model mampu memberikan rekomendasi dengan tingkat akurasi yang tinggi berdasarkan kata kunci dan deskripsi film yang dimasukkan pengguna (Infotek, 2024).

Selain fitur pencarian dan rekomendasi, aspek keamanan dan kenyamanan pengguna juga menjadi perhatian utama dalam aplikasi *streaming* film. Penerapan autentikasi JWT, enkripsi *password*, dan validasi *input* merupakan standar yang harus diterapkan untuk menjaga keamanan data pengguna (Descania, 2022). Sistem *streaming* adaptif, seperti yang diimplementasikan dengan HLS.js, memungkinkan kualitas video menyesuaikan kondisi jaringan pengguna, sehingga pengalaman menonton tetap optimal bahkan pada koneksi internet yang fluktuatif.

Metode *Prototyping* sendiri terdiri dari beberapa tahapan utama, yaitu analisis kebutuhan, desain cepat, pembangunan *prototype*, evaluasi awal, revisi, hingga implementasi sistem. Setiap tahapan berfokus pada pemahaman kebutuhan pengguna, pengujian fitur secara langsung, dan perbaikan berkelanjutan berdasarkan umpan balik yang diterima (Setiawan, 2021; JNATIA, 2023). Pendekatan ini terbukti efektif dalam mengembangkan aplikasi *streaming* film yang adaptif, *user-friendly*, dan relevan dengan perkembangan teknologi serta tren konsumsi hiburan digital.

Kajian-kajian tersebut menunjukkan bahwa pengembangan aplikasi *streaming* film berbasis web dengan metode *Prototyping*, sistem rekomendasi berbasis konten, serta penerapan standar keamanan modern merupakan solusi yang tepat untuk memenuhi kebutuhan hiburan digital masa kini. Sistem ini tidak hanya efisien dan akurat, tetapi juga fleksibel dan mudah dikembangkan lebih lanjut sesuai kebutuhan pengguna.

DAFTAR PUSTAKA

Setiawan, A. (2021). "Penerapan Metode *Prototype* dalam Perancangan Media Pembelajaran Interaktif". *Jurnal Sistem Informasi*, 8(2), 112–120.

Murdiani, R., & Hermawan, A. (2022). "Prototipe Sistem Rekomendasi Film Indonesia Menggunakan Content-Based Filtering dan Vector Space Model". *Infotek: Jurnal Informatika dan Teknologi*, 7(2), 444–455.

Descania, D.Y. (2022). "Penerapan Metode *Prototype* Pada Pengembangan Sistem Antrian Online". *Jurnal Sistem dan Rekayasa Komputer*, 10(1), 23–31.

JNATIA. (2023). "Perancangan *Prototype* Aplikasi Deteksi dan Pelacakan Manusia Berbasis Video *Streaming*". *Jurnal Nasional Teknologi Informasi dan Aplikasinya*, 1(4), 1072–1080.

Pressman, R. S. (2015). *Software Engineering: A Practitioner's Approach (8th Edition)*. New York: McGraw-Hill.

Nugroho, A. (2019). *Rekayasa Perangkat Lunak Berbasis Web*. Yogyakarta: Andi Offset.

Documentation HLS.js. (2024). *HLS.js – JavaScript library that plays HLS in browsers*.

Documentation JWT. (2024). *JSON Web Tokens – Introduction*.