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

Kohesi: Jurnal Multidisiplin Saintek
Volume 7 No 5 Tahun 2025

PENERAPAN METODE *CONTEN-BASED FILTERING* PADA SISTEM REKOMENDASI BUKU (STUDI KASUS: SMK TANJUNG JAKARTA BARAT)

Sulastri Gajah[1], Nenden Siti Fatonah[2]
[1,2]Program Studi Teknik Informatika, Universitas Esa Unggul
[1]Sulas8812@student.esaunggul.ac.id, [2]nenden.siti@esaunggul.ac.id

Abstract

Located in Jl. Dr. Nurdin 4 No. 1, Grogol Petamburan District, West Jakarta City, DKI Jakarta, SMK Tanjung West Jakarta is a private vocational school. Finding appropriate reading recommendations for pupils is a difficulty for SMK Tanjung West Jakarta as an educational institution. One approach that may address the issue of readers' desires and provide reference information for other like books is a book recommendation system. A system may assess the similarity of each book using a content-based filtering technique that leverages cosine similarity. Because of its potential to help readers discover books that are a good fit for their tastes, book recommendation systems have recently emerged as a hot area of academic inquiry. Book recommendation systems often use the Content-Based Filtering approach. The goal of this study is to create a system that uses the Content-Based Filtering approach to suggest books. Using this strategy, books that users have read in the past may be recommended to them based on shared characteristics. Data about books is culled from reliable sources, and the algorithm suggests reading material based on shared characteristics with user favorites. When it comes to making personalized book recommendations, the established system performs well, according to the test findings.

Keywords: *Book Recommendation System, Content-Based Filtering, TF-IDF, Cosine Similarity.*

Abstrak

Berlokasi di Jl. Dr. Nurdin 4 No. 1, Kecamatan Grogol Petamburan, Kota Jakarta Barat, DKI Jakarta, SMK Tanjung Jakarta Barat merupakan sekolah kejuruan swasta. Menemukan rekomendasi bacaan yang tepat bagi siswa merupakan kesulitan bagi SMK Tanjung Jakarta Barat sebagai lembaga pendidikan. Salah satu pendekatan yang dapat menjawab masalah keinginan pembaca dan menyediakan informasi referensi untuk buku sejenis lainnya adalah sistem rekomendasi buku. Suatu sistem dapat menilai kesamaan setiap buku menggunakan teknik penyaringan berbasis konten yang memanfaatkan kesamaan kosinus. Karena potensinya untuk membantu pembaca menemukan buku yang sesuai dengan

Article History:
Received: Maret 2025
Reviewed: Maret 2025
Published: Maret 2025

Plagirism Checker No 234
Prefix DOI :
10.8734/Kohesi.v1i2.365

selera mereka, sistem rekomendasi buku baru-baru ini muncul sebagai bidang penelitian akademis yang sedang hangat. Sistem rekomendasi buku sering kali menggunakan pendekatan Penyaringan Berbasis Konten. Tujuan dari penelitian ini adalah untuk membuat sistem yang menggunakan pendekatan Penyaringan Berbasis Konten untuk menyarankan buku. Dengan menggunakan strategi ini, buku-buku yang telah dibaca pengguna di masa lalu dapat direkomendasikan kepada mereka berdasarkan karakteristik yang sama. Data tentang buku dikumpulkan dari sumber yang dapat dipercaya, dan algoritme menyarankan bahan bacaan berdasarkan karakteristik yang sama dengan favorit pengguna. Dalam hal membuat rekomendasi buku yang dipersonalisasi, sistem yang ada bekerja dengan baik, menurut hasil pengujian.
Kata kunci: Sistem Rekomendasi Buku, *Content-Based Filtering*, TF-IDF, *Cosine Similarity*.

1. PENDAHULUAN

Membaca buku merupakan salah satu cara bagi setiap orang untuk mendapatkan ilmu pengetahuan dan wawasan baru mengenai banyak hal. Sistem Rekomendasi Buku digunakan untuk membantu pengguna atau pembaca dengan memberikan rekomendasi, serta riwayat membaca[1]. Namun, tantangan yang dihadapi oleh banyak SMK adalah keterbatasan akses terhadap buku-buku yang sesuai dengan kurikulum kejuruan yang diajarkan. Tidak semua perpustakaan SMK memiliki koleksi buku yang lengkap dan up-to-date, yang pada akhirnya dapat mempengaruhi proses belajar-mengajar dan pencapaian kompetensi siswa. Sekolah Menengah Kejuruan (SMK) Tanjung Jakarta, sebagai institusi pendidikan, menghadapi tantangan dalam menyediakan referensi bacaan yang sesuai dengan kebutuhan siswa[2].

Sistem rekomendasi buku dapat membantu siswa dan guru menemukan buku-buku yang sesuai dengan kebutuhan dan minat mereka sistem akan merekomendasikan buku yang memiliki kemiripan dengan buku yang dicari.[3]. Sistem rekomendasi buku bertujuan untuk membantu siswa menemukan buku yang sesuai dengan minat dan kebutuhan mereka, tanpa harus mencarinya secara manual. Metode yang digunakan untuk mengembangkan sistem ini adalah content-based filtering[4]. Implementasi content-based filtering di SMK Tanjung Jakarta diharapkan dapat meningkatkan efisiensi dan efektivitas dalam proses pemilihan buku. Dengan adanya sistem rekomendasi ini, diharapkan siswa dapat lebih mudah menemukan buku yang sesuai dengan mata pelajaran yang mereka pelajari, minat pribadi. Dengan demikian, penerapan metode content-based filtering pada sistem rekomendasi buku di SMK Tanjung Jakarta diharapkan dapat memberikan kontribusi positif dalam mendukung proses pembelajaran di sekolah[5].

Penulis memilih merancang bangun sistem rekomendasi buku, dengan menggunakan metode Content-Based Filtering. Hal ini bertujuan untuk dapat membantu pengguna dalam mencari buku dengan menentukan kemiripan konten berdasarkan dari judul buku, penulis, sinopsis, serta kategori, kemudian memberikan rekomendasi buku yang serupa[6]. Metode yang digunakan yaitu Content-Based Filtering, merupakan sistem yang memanfaatkan preferensi dari pengguna dan berdasarkan dari konten item untuk nantinya menghasilkan sebuah rekomendasi[7]

2. TINJAUAN PUSTAKA

Penelitian terdahulu yang akan menjadi referensi untuk membuat laporan tugas akhir, membantu mengidentifikasi terhadap pemahaman tentang topik yang akan diteliti. Ini melibatkan membahas bagaimana penelitian tersebut memperluas pengetahuan, mengidentifikasi masalah baru, memberikan arahan baru untuk penelitian lebih lanjut. Untuk mendapatkan panduan tambahan tentang penelitian terdahulu yang sesuai dengan tugas akhir, dan memberikan wawasan penting untuk penelitian yang akan di lakukan. Berikut ini adalah penelitian terdahulu yang berkaitan dengan tugas akhir penulis:

Penelitian terdahulu dilakukan oleh Muhammad Rizqi Az Zayyad, (2021) "Sistem Rekomendasi Buku Menggunakan Metode *Conten Based Filtering*" untuk mengembangkan sistem yang dapat menyediakan referensi baru bagi pelanggan melalui alternatif pengetahuan tentang buku-buku yang memiliki kesamaan atau saling terkait. Salah satu penyedia layanan jual beli buku, Gramedia, menyediakan data yang digunakan untuk membangun mesin rekomendasi ini. Penelitian dalam studi ini menggunakan teknik penyaringan berbasis konten. Peneliti menggunakan algoritma TF-IDF dan *cosine similarity* untuk menilai bobot dan kesamaan data setiap buku. Sistem bawaan dapat memberikan nilai presisi 85% dan memberikan saran tergantung pada kesamaan setiap buku, menurut hasil pengujian. Sementara penelitian selanjutnya dilakukan oleh Tegar Ridwansyah, Betty Subartini, Sisilia Sylviani (2024) "Penerapan Metode *Content Based Filtering* Pada Sistem Rekomendasi" Studi ini terutama akan berfokus pada salah satu teknik rekomendasi penyaringan berbasis konten sebagai contoh pendekatan sistem rekomendasi dalam tindakan. Untuk lebih memahami cara membangun sistem rekomendasi menggunakan penyaringan berbasis konten, aplikasi ini dirancang untuk digunakan. Hal ini digunakan untuk mencari metode *content-based filtering* dari berbagai pendekatan yang digunakan dan mengatasi keterbatasan yang ada pada metode tersebut, seperti *cold start* problem

3. METODE

Saran dalam penelitian ini dihasilkan menggunakan strategi penyaringan berbasis konten. Salah satu teknik untuk sistem rekomendasi adalah penyaringan berbasis konten, yang memberikan saran kepada pengguna berdasarkan kualitas dan fitur materi itu sendiri. Sasaran pemodelan CBF adalah memberikan nilai numerik untuk setiap buku. Untuk melakukan ini, vektor TF-IDF dihasilkan menggunakan deskripsi atau ringkasan buku. Sebagai cerminan kualitas masing-masing buku, vektor ini memberikan bobot lebih pada kata-kata yang signifikan dan tidak umum di seluruh koleksi. Dengan menyarankan buku lain yang sebanding dengan yang dicari pengguna, pendekatan penyaringan berbasis konten dapat meningkatkan keakuratan saran buku. [8].

3.1 Teknik Pengumpulan Data

Metode yang digunakan oleh peneliti untuk mengumpulkan data dalam penelitian ini, yaitu: studi pustaka, observasi, dan wawancara.

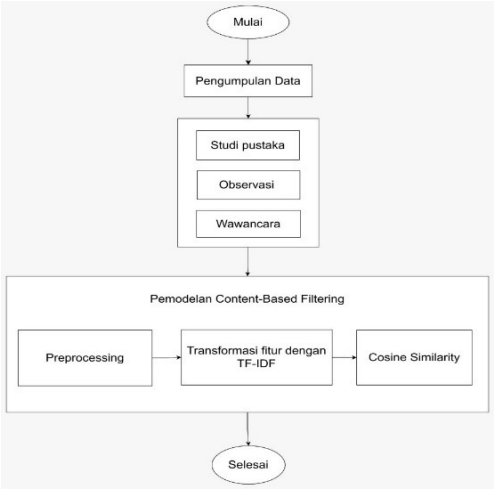
- Studi pustaka

  Digunakan untuk mengumpulkan informasi dari sumber-sumber berupa jurnal, untuk mendukung penyusunan proposal ini.
- Observasi

  Teknik ini memungkinkan peneliti mengamati secara langsung untuk memenuhi kebutuhan data yang diperlukan dalam penerapan sistem rekomendasi buku.
- Wawancara

  Wawancara digunakan untuk mengumpulkan data secara langsung dari responden untuk memperoleh informasi terkait data-data yang diperlukan.

3.2 Tahapan Penelitian



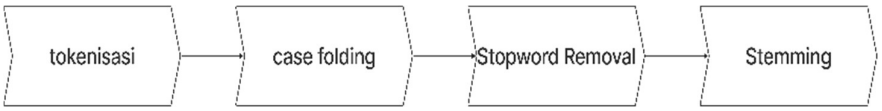
Gambar 1. Tahapan Penelitian

3.3 *Preprocessing* (Pemrosesan Data)

Pada tahap pertama pemrosesan data dilakukan dengan tiga langkah yaitu identifikasi sumber data, pembersihan data, serta pemrosesan teks.

1) Sumber data utama dalam penelitian ini adalah perpustakaan SMK Tanjung Jakarta. Data buku yang akan dikumpulkan mencakup informasi penting seperti judul buku, deskripsi atau sinopsis singkat, pengarang, kategori atau genre buku, serta informasi relevan lainnya yang dapat membantu dalam proses analisis dan rekomendasi.
2) Pembersihan data. Proses ini bertujuan untuk memastikan kualitas dan konsistensi data yang akan digunakan dalam analisis dan pemodelan. Pembersihan data meliputi penghapusan data buku yang terduplikasi, pengisian atau penghapusan data buku yang tidak lengkap (misalnya, buku tanpa sinopsis), serta penanganan karakter yang tidak perlu seperti tanda baca yang berlebihan atau karakter khusus yang dapat mengganggu proses analisis teks.

3) Pemrosesan teks terdiri dari tokenisasi, *case folding*, *stopword removal*, *stemming*.



Gambar 2. Tahapan *Preprocessing*

a. Tokenisasi adalah proses tokenisasi teks, yang melibatkan pembagian teks menjadi bagian-bagian yang lebih kecil. Kombinasi huruf, angka, simbol, atau tanda baca apa pun dapat berfungsi sebagai token.
b. *Case folding* adalah mengubah teks menjadi huruf kecil. Ketika kata-kata serupa tetapi menggunakan huruf kapital yang berbeda, ini membantu menyatukannya.
c. *Stopword* adalah kata-kata umum yang sering muncul dalam teks tetapi tidak membawa banyak makna (contoh: "ini", "dan", "yang"). Menghilangkan *stopword* dapat membantu mengurangi ukuran data dan meningkatkan efisiensi pemrosesan selanjutnya.
d. *Stemming* adalah proses yang memotong akhiran kata.

3.4 Transformasi Fitur dengan TF-IDF

TF-IDF adalah suatu pendekatan terhadap pembobotan kata yang menilai signifikansi suatu kata dalam suatu korpus (Rahmadani et al., 2025). Teknik ini menggabungkan dua metrik:

1) *Term Frequency* (TF)

Seberapa sering sebuah kata muncul dalam dokumen. Semakin sering muncul, semakin tinggi nilai TF-nya.

2) *Inverse Document Frequency* (IDF)

Frekuensi kemunculan suatu kata di seluruh korpus. Semakin besar nilai IDF, semakin jarang kemunculannya.

Nilai TF-IDF, yang merupakan hasil perkalian TF dan IDF, menunjukkan seberapa penting sebuah kata dalam konteks dokumen dan seluruh korpus. TF-IDF akan memberikan bobot lebih pada kata-kata yang umum dalam teks tetapi tidak umum dalam kumpulan data lainnya.

$$\mathrm{TF} - \mathrm{IDF} = \mathrm{TF\ X\ IDF} \qquad (1)$$

- Perhitungan TF-IDF

Ada 3 sinopsis buku (dokumen) sebagai berikut:

- Dokumen 1: "Buku ini menceritakan petualangan seorang pemuda pemberani."
- Dokumen 2: "Novel ini mengisahkan perjalanan seorang wanita mencari jati diri."
- Dokumen 3: "Buku ini membahas petualangan seorang detektif memecahkan misteri."

1. Kata-Kata Unik (*Vocabulary*)

Dari ketiga dokumen ini, berikut adalah daftar semua kata unik (tanpa kata-kata umum seperti "ini", "seorang"):

Buku
Menceritakan
Petualangan
Pemuda
Pemberani

Novel
Mengisahkan
Perjalanan
Wanita
Mencari
Jati
Diri
Membahas
Detektif
Memecahkan
Misteri

2. *Term Frequency* (TF)

Mari kita hitung frekuensi setiap kata dalam dokumen pertama dan hitung TF:

- **Dokumen 1:** "Buku ini menceritakan petualangan seorang pemuda pemberani."
  Total Kata: 7 (Buku, ini, menceritakan, petualangan, seorang, pemuda, pemberani)
  - TF untuk setiap kata:
    Buku: 1/7 = 0.143
    menceritakan: 1/7 = 0.143
    petualangan: 1/7 = 0.143
    pemuda: 1/7 = 0.143
    pemberani: 1/7 = 0.143
- **Dokumen 2:** "Novel ini mengisahkan perjalanan seorang wanita mencari jati diri."
  Total Kata: 9 (Novel, ini, mengisahkan, perjalanan, seorang, wanita, mencari, jati, diri)
  - TF untuk setiap kata:
    Novel: 1/9 ≈ 0.111
    mengisahkan: 1/9 ≈ 0.111
    perjalanan: 1/9 ≈ 0.111
    wanita: 1/9 ≈ 0.111
    mencari: 1/9 ≈ 0.111
    jati: 1/9 ≈ 0.111
    diri: 1/9 ≈ 0.111
- **Dokumen 3:** "Buku ini membahas petualangan seorang detektif memecahkan misteri."
  Total Kata: 8 (Buku, ini, membahas, petualangan, seorang, detektif, memecahkan, misteri)
  - TF untuk setiap kata:
    Buku: 1/8 = 0.125
    membahas: 1/8 = 0.125
    petualangan: 1/8 = 0.125
    detektif: 1/8 = 0.125
    memecahkan: 1/8 = 0.125
    misteri: 1/8 = 0.125

3. *Inverse Document Frequency* (IDF)

Inverse Document Frequency dihitung dengan rumus:

$$IDF(t) = \log\left(\frac{N}{df(t)}\right) \qquad (2)$$

- Di mana:
  N = Jumlah total dokumen
  df(t) = Jumlah dokumen yang mengandung kata tersebut

IDF untuk beberapa kata:

- "Buku": Ada di Dokumen 1 dan Dokumen 3, jadi df(Buku) = 2

$$IDF(Buku) = \log\left(\frac{3}{2}\right) \approx 0{,}405$$

- "petualangan": Ada di Dokumen 1 dan Dokumen 3, jadi df(petualangan) = 2

$$IDF(petualangan) = \log\left(\frac{3}{2}\right) \approx 0{,}405$$

- "menceritakan": Hanya ada di Dokumen 1, jadi df(menceritakan) = 1

$$IDF(menceritakan) = \log\left(\frac{3}{1}\right) \approx 1{,}099$$

- "mengisahkan": Hanya ada di Dokumen 2, jadi df(mengisahkan) = 1

$$IDF(mengisahkan) = \log\left(\frac{3}{1}\right) \approx 1{,}099$$

4. TF-IDF untuk Setiap Kata dalam Setiap Dokumen

Menghitung TF-IDF untuk setiap kata dengan mengalikan TF dan IDF.

- Dokumen 1:
  TF-IDF(Buku): 0.143 X 0.405 ≈ 0.058
  TF-IDF(menceritakan): 0.143 X1.099 ≈ 0.157
  TF-IDF(petualangan): 0.143 X 0.405 ≈ 0.058
  TF-IDF(pemuda): 0.143 X 1.099 ≈ 0.157
  (IDF pemuda dihitung seperti IDF menceritakan karena hanya muncul di dokumen ini)
  TF-IDF(pemberani): 0.143 X 1.099 ≈ 0.157
- Dokumen 2:
  TF-IDF(Novel): 0.111 X 1.099 ≈ 0.122
  TF-IDF(mengisahkan): 0.111 X 1.099 ≈ 0.122
  TF-IDF(perjalanan): 0.111 X 1.099 ≈ 0.122
  (perjalanan hanya ada di dokumen ini)
  TF-IDF(wanita): 0.111 X 1.099 ≈ 0.122
  TF-IDF(mencari): 0.111 X 1.099 ≈ 0.122
  TF-IDF(jati): 0.111 X 1.099 ≈ 0.122
  TF-IDF(diri): 0.111 X 1.099 ≈ 0.12

- Dokumen 3:
  TF-IDF(Buku): 0.125 X 0.405 ≈ 0.051
  TF-IDF(membahas): 0.125 X 1.099 ≈ 0.137
  (membahas hanya ada di dokumen ini)
  TF-IDF(petualangan): 0.125 X 0.405 ≈ 0.051
  TF-IDF(detektif): 0.125 X 1.099 ≈ 0.137
  TF-IDF(memecahkan): 0.125 X 1.099 ≈ 0.137
  TF-IDF(misteri): 0.125 X 1.099 ≈ 0.137

Setelah menghitung TF-IDF untuk setiap kata dalam setiap dokumen, hasilnya digunakan untuk membangun vektor TF-IDF yang bisa digunakan dalam model *machine learning*, seperti untuk menghitung kesamaan antar dokumen menggunakan *Cosine Similarity*.

3.5 Perhitungan TF-IDF dan *Cosine Similarity*

Dalam konteks rekomendasi buku, CBF menganalisis elemen-elemen seperti deskripsi, sinopsis, genre, dan kata kunci yang terkait dengan buku untuk mengidentifikasi buku-buku lain yang memiliki konten serupa. Langkah pertama dalam pemodelan CBF adalah mengubah setiap buku menjadi representasi numerik. Kemudian mengubah teks deskripsi atau sinopsis buku menjadi vektor TF-IDF. Langkah selanjutnya menghitung kemiripan antara buku-buku, dengan menggunakan *cosine similarity*. Semakin tinggi nilai *cosine similarity*, semakin mirip kedua buku tersebut dalam hal konten.

- **Penerapan**

Misalnya ada 3 buku dengan vektor TF-IDF sebagai berikut:

- Buku 1: [0.2, 0.5, 0.1, 0.3]
- Buku 2: [0.1, 0.4, 0.2, 0.1]
- Buku 3: [0.3, 0.6, 0.05, 0.2]

$$Cosine\ Similarity = \frac{A \cdot B}{||A|| \times ||B||} \qquad (3)$$

- Di mana:
  - A·B adalah hasil perkalian *dot product* dari dua vektor.
  - ‖A‖ dan ‖B‖ adalah norma (panjang) dari masing-masing vektor.

1. Hitung *dot product* dari vektor Buku 1 dan Buku 2:
   A·B=(0.2×0.1)+(0.5×0.4)+(0.1×0.2)+(0.3×0.1)=0.02+0.2+0.02+0.03=0.27
2. Hitung norma (panjang) dari masing-masing vektor:
   $\|A\| = \sqrt{(0.2)^2 + (0.5)^2 + (0.1)^2 + (0.3)^2} = \sqrt{0.04 + 0.25 + 0.01 + 0.09} = \sqrt{0.39} \approx 0.6245$
   $\|B\| = \sqrt{(0.1)^2 + (0.4)^2 + (0.2)^2 + (0.1)^2} = \sqrt{0.01 + 0.16 + 0.04 + 0.01} = \sqrt{0.22} \approx 0.469$
3. 3. Hitung *cosine similarity*:

$$Cosine\ Similarity = \frac{0.27}{0.6245 \times 0.469} = \frac{0.27}{0.2927} \approx 0.9225$$

Setelah menghitung *cosine similarity* antara setiap pasangan buku, didapatkan:

- Kemiripan (Buku 1, Buku 2) = 0.9225
- Kemiripan (Buku 1, Buku 3) = 0.9704
- Kemiripan (Buku 2, Buku 3) = 0.9116

Berdasarkan hasil ini, jika seorang pengguna mencari Buku 1, sistem CBF akan merekomendasikan Buku 3 sebagai pilihan pertama, dan diikuti oleh Buku 2.

3.6 Evaluasi Model

Evaluasi model adalah langkah penting untuk memahami seberapa baik sistem rekomendasi yang akan dibuat ini bekerja dan mengidentifikasi area yang perlu ditingkatkan. Dalam konteks rekomendasi buku, akan dipastikan bahwa sistem dapat memberikan rekomendasi yang relevan dan menarik bagi pengguna.

- *Precision*: Mengukur seberapa akurat rekomendasi yang diberikan. *Precision* tinggi berarti sebagian besar buku yang direkomendasikan relevan dengan pengguna.

$$Precision = \frac{Jumlah\ Buku\ yang\ Relevan\ dan\ Direkomendasikan}{Jum \quad Buku\ yang\ Direkomendasikan} \qquad (4)$$

- Dengan data yang diberikan:
  Jumlah buku yang relevan dan direkomendasikan = 3
  Jumlah total buku yang direkomendasikan = 5

$$Precision = \frac{3}{5} = 0.6 \text{ atau } 60\%$$

Ini memberikan gambaran menyeluruh tentang kinerja sistem rekomendasi. *Precision* menunjukkan seberapa relevan rekomendasi tersebut.

Uji *Precision* Sistem Rekomendasi Buku

Judul buku yang digunakan "Pengelolaan Arsip Digital" sebagai salah satu buku yang direkomendasikan oleh sistem, lalu menghitung *precision* berdasarkan relevansi rekomendasi sistem.

No.	Judul Buku	Relevansi (1: Relevan, 0: Tidak Relevan)
1	Pengelolaan Arsip	1
2	Manajemen Keuangan Digital	1
3	Teknik administrasi Keuangan	1
4	Panduan Administrasi Keuangan	0
5	Administrasi Modern	0
6	Optimalisasi Data dan Informasi	1

Langkah Perhitungan *Precision.*

Rumus *Precision*:

$$Precision = \frac{True\ Positif\ (TP)}{True\ Positif\ (TP) + Fals \quad Positif\ (FP)} \qquad (5)$$

- *True Positives* (TP): Jumlah buku yang direkomendasikan relevan (nilai relevansi = 1).
- *False Positives* (FP): Jumlah buku yang direkomendasikan tetapi tidak relevan (nilai relevansi = 0).

- Evaluasi Hasil Rekomendasi
  TP = 4
  1. Pengelolaan Arsip Digital,
  2. Manajemen Keuangan Digital,
  3. Teknik Administrasi Keuangan,
  4. Optimalisasi Data dan Informasi.
  FP = 2
  1. Panduan Administrasi Keuangan,
  2. Administrasi Modern.

- Perhitungan *Precision*

$$Precision = \frac{4}{4+2} = \frac{4}{6} = 0.67 \text{ atau } 67\%$$

Artinya, 67% dari buku yang direkomendasikan oleh sistem adalah relevan untuk pengguna. Dengan menggunakan judul buku tersebut, sistem berhasil memberikan hasil rekomendasi yang cukup baik, meskipun terdapat beberapa buku yang tidak relevan.

4. HASIL DAN PEMBAHASAN

4.1 Pengumpulan Data

Pengumpulan data buku pada penelitian ini menggunakan data buku langsung dari perpustakaan SMK Tanjung Jakarta. Data yang diambil meliputi judul buku, pengarang, tahun terbit, sinopsis. Proses ini akan melibatkan pendokumentasian secara rinci terhadap setiap buku yang ada di perpustakaan.

a. *Dataset* buku

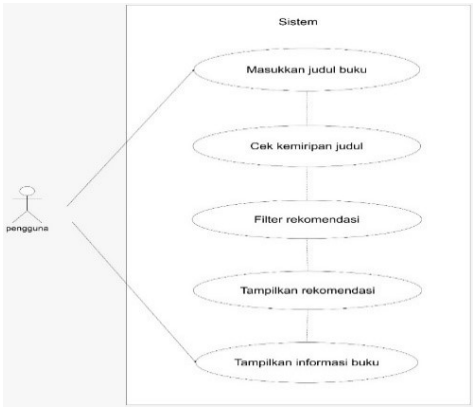
id_buku	judul_buku	deskripsi	kategori	user	penulis	rating	ulasan	tahun_terbit
1	Panduan Efektivitas Kantor melalui Otomatisasi Digital	Buku ini adalah panduan komprehensif untuk memaksimalkan produktivitas kantor melalui penerapan teknologi otomatisasi digital. Dari manajemen dokumen hingga otomatisasi alur kerja, buku ini menyediakan strategi praktis yang dapat diimplementasikan oleh perusahaan dari berbagai ukuran. Setiap bab memandu pembaca dalam memahami bagaimana otomatisasi dapat meningkatkan efisiensi, mengurangi kesalahan manual, dan menciptakan lingkungan kerja yang lebih responsif. Cocok untuk para profesional di bidang administrasi dan manajemen yang ingin meningkatkan efektivitas kerja.	OTKP	user1	Adi Santoso	8	Buku yang solid dan informatif yang menyediakan wawasan berharga. Penulis berhasil mengulas topik penting, meskipun ada beberapa bagian yang bisa dikembangkan lebih jauh. Ini adalah sumber yang baik bagi mereka yang mencari pemahaman dasar yang kuat.	2018
2	Panduan Digitalisasi Administrasi untuk Kinerja Optimal Kantor	Digitalisasi administrasi adalah langkah penting untuk meningkatkan efisiensi operasional dan kinerja kantor secara keseluruhan. Buku ini menjelaskan proses transformasi dari sistem tradisional ke sistem digital dengan bahasa yang mudah dipahami. Anda akan menemukan berbagai teknik untuk mengatur dokumen secara digital, mengelola komunikasi internal, dan menjaga data tetap aman. Buku ini ditujukan untuk para pengelola kantor dan staf administrasi yang ingin membawa kantornya menuju era digital.	OTKP	user2	Adi Santoso	7	Buku yang solid dan informatif yang menyediakan wawasan berharga. Penulis berhasil mengulas topik penting, meskipun ada beberapa bagian yang bisa dikembangkan lebih jauh. Ini adalah sumber yang baik bagi mereka yang mencari pemahaman dasar yang kuat.	2020
3	Strategi Efektivitas Kantor dengan Pengelolaan Data Terintegrasi	Pengelolaan data yang terintegrasi adalah kunci untuk meningkatkan efektivitas kantor. Buku ini memberikan panduan tentang bagaimana menghubungkan berbagai sumber data dalam satu sistem yang terpadu. Dengan strategi ini, perusahaan dapat mengakses informasi penting dengan lebih cepat dan meningkatkan koordinasi antar tim. Sinopsis ini sangat cocok bagi para manajer dan staf yang ingin memahami konsep dasar data	OTKP	user3	Adi Santoso	7	Buku yang solid dan informatif yang menyediakan wawasan berharga. Penulis berhasil mengulas topik penting, meskipun ada beberapa bagian yang bisa dikembangkan lebih jauh. Ini adalah sumber yang baik bagi mereka yang mencari pemahaman dasar yang kuat.	2018

Gambar 3. *Dataset* Buku

Data tersebut adalah data yang digunakan untuk pembuatan sistem rekomendasi

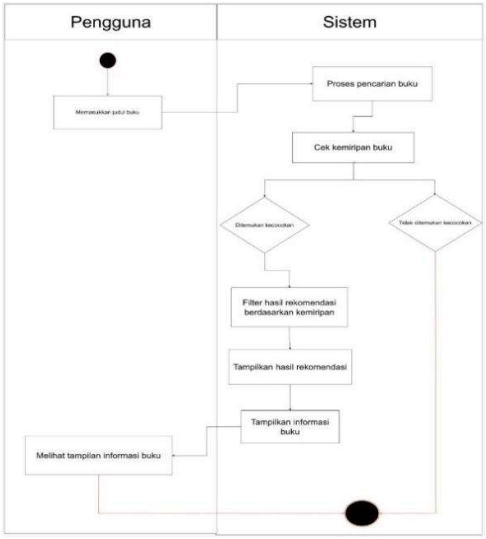
b. *Design* dan Implementasi Sistem

1) *Use Case Diagram*



Gambar 4. *Use Case Diagram.*

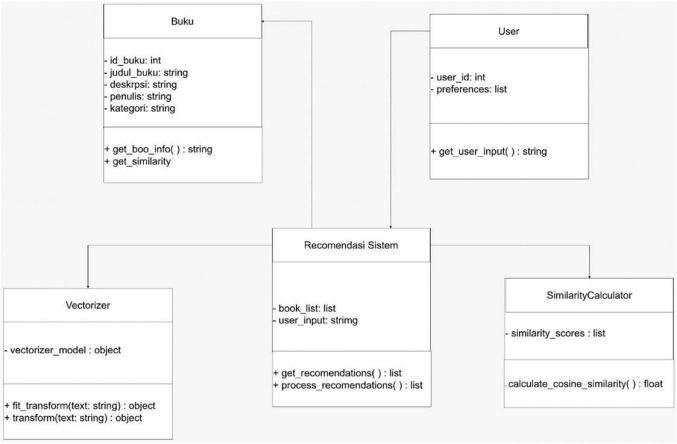
2) *Activity Diagram*



Gambar 5. *Activity* Diagram

Pada *activity* diagram sistem rekomendasi buku, setelah pengguna membuka sistem rekomendasi buku. Pengguna mengetikkan judul buku yang ingin dicari. Kemudian sistem memeriksa *dataset* untuk mencocokkan judul buku yang sesuai dengan *input* dari pengguna dan melakukan perhitungan kemiripan antara judul buku yang dimasukkan pengguna dan judul buku dalam *dataset*. Jika tidak ditemukan sistem memberi tahu pengguna bahwa tidak ada buku yang cocok, dan proses berakhir. Kemudian jika buku ditemukan maka sistem menyaring hasil rekomendasi berdasarkan skor kemiripan tertentu (misalnya, di atas 0.3). Sistem menampilkan daftar buku yang memiliki kemiripan tertinggi dengan *input* dari pengguna. Pengguna dapat melihat detail informasi buku seperti judul, deskripsi, penulis, rating, dan tahun terbit.

3) *Class Diagram*



Gambar 6. *Class Diagram*

c. Penerapan *Preprocessing*

- **Memuat model TF-IDF dari *file pickle***

```
# Load model TF-IDF dari file pickle
try:
    with open('vectorizer_judul.pkl', 'rb') as f:
        vectorizer_judul = pickle.load(f)

    with open('vectorizer_deskripsi.pkl', 'rb') as f:
        vectorizer_deskripsi = pickle.load(f)

    with open('vectorizer_penulis.pkl', 'rb') as f:
        vectorizer_penulis = pickle.load(f)

    print("Model TF-IDF berhasil dimuat.")
except FileNotFoundError as e:
    print(f"File tidak ditemukan: {e}")
    exit()
```

Bagian ini mencoba untuk memuat model TF-IDF dari *file pickle* yang sebelumnya disimpan. Jika *file* ditemukan, model TF-IDF akan dimuat ke dalam variabel *vectorizer*_judul, *vectorizer*_deskripsi, dan *vectorizer*_penulis. Pesan "Model TF-IDF berhasil dimuat" akan dicetak sebagai konfirmasi. Jika ada *file* yang tidak ditemukan (*FileNotFoundError*), pesan kesalahan akan ditampilkan, dan program akan berhenti (*exit*()), yang mengindikasikan bahwa proses tidak dapat dilanjutkan tanpa model yang diperlukan.

- ***Preprocessing***

1) *Preprocessing input* pengguna

```
user_input_processed = preprocess_text(user_input)
```

Input dari pengguna diproses menggunakan fungsi *preprocess_text* untuk mengubah teks menjadi bentuk yang lebih sederhana (huruf kecil, tanpa tanda baca, dan tanpa *stopwords*).

2) Pencarian judul buku

```
matching_row = df[df['judul_buku'].str.contains(user_input_processed,
case=False, regex=False)]

    if matching_row.empty:
        print("Judul tidak ditemukan dalam dataset.")
        return None
```

Fungsi ini mencari judul buku di *dataset* yang mengandung kata kunci dari *input* pengguna. Jika tidak ditemukan, maka fungsi akan mengembalikan pesan "Judul tidak ditemukan dalam *dataset*."

3) Mengambil data buku yang ditemukan

```
user_deskripsi = matching_row['deskripsi'].values[0]
    user_penulis = matching_row['penulis'].values[0]
    user_kategori = matching_row['kategori'].values[0]
```

Mengambil deskripsi, penulis, dan kategori dari buku yang ditemukan di *dataset*.

4) *Preprocessing* deskripsi dan penulis

```
user_deskripsi_processed = preprocess_text(user_deskripsi)
user_penulis_processed = preprocess_text(user_penulis)
```

Melakukan *preprocessing* pada deskripsi dan nama penulis buku yang ditemukan

5) Menghitung *similarity* berdasarkan judul

```
user_tfidf_judul = vectorizer_judul.transform([user_input_processed]) *
0.9
    tfidf_judul = vectorizer_judul.transform(df['judul_buku']) * 0.9
    title_similarity_scores = cosine_similarity(user_tfidf_judul,
tfidf_judul)[0]
```

Mengubah *input* judul buku pengguna menjadi representasi vektor TF-IDF. Menghitung *similarity score* antara judul *input* pengguna dengan semua judul buku dalam *dataset* menggunakan *cosine similarity*. Bobot 0.9 digunakan untuk memberikan prioritas tinggi pada judul buku saat menghitung kesamaan.

6) Mengambil kandidat awal berdasarkan judul

```
initial_candidates = sorted(range(len(title_similarity_scores)),
key=lambda i: title_similarity_scores[i], reverse=True)
```

Mengurutkan indeks buku berdasarkan *similarity score* dengan judul buku *input* pengguna.

7) Menghitung *similarity* berdasarkan deskripsi dan penulis

```
   user_tfidf_deskripsi =
vectorizer_deskripsi.transform([user_deskripsi_processed]) * 0.7
      user_tfidf_penulis =
vectorizer_penulis.transform([user_penulis_processed]) * 0.1
      user_tfidf_combined = hstack([user_tfidf_deskripsi,
user_tfidf_penulis])

      tfidf_deskripsi =
vectorizer_deskripsi.transform(df.loc[initial_candidates, 'deskripsi'])
* 0.7
      tfidf_penulis =
vectorizer_penulis.transform(df.loc[initial_candidates, 'penulis']) *
0.1
      tfidf_combined = hstack([tfidf_deskripsi, tfidf_penulis])
```

Menggabungkan representasi TF-IDF dari deskripsi dan penulis untuk buku yang dipilih sebagai kandidat awal. Bobot 0.7 diberikan untuk deskripsi dan 0.1 untuk penulis, menunjukkan bahwa deskripsi lebih berpengaruh daripada nama penulis.

8) Menghitung skor *similarity* akhir

```
similarity_scores = cosine_similarity(user_tfidf_combined,
tfidf_combined)[0]
```

Menghitung *cosine similarity* antara *input* pengguna (gabungan deskripsi dan penulis) dengan kandidat buku yang relevan.

9) Menyaring dan mengurutkan hasil rekomendasi

```
sorted_indices = sorted(range(len(similarity_scores)), key=lambda i:
similarity_scores[i], reverse=True)

    final_indices, final_scores = [], []

    # Filter hasil hanya dengan similarity_score di atas 0.3
    min_similarity_threshold = 0.3
    for idx in sorted_indices:
        book_index = initial_candidates[idx]
        book_title = df.iloc[book_index]['judul_buku']
        book_kategori = df.iloc[book_index]['kategori']

        if book_title != user_input_processed and
similarity_scores[idx] >= min_similarity_threshold:
            final_indices.append(book_index)
            final_scores.append(similarity_scores[idx])

        if len(final_indices) == top_n:
            break
```

Hanya memilih buku yang memiliki *similarity score* di atas 0.3 dan tidak sama dengan judul *input* pengguna. Mengambil hingga *top_n* rekomendasi terbaik berdasarkan kesamaan.

10) Membuat *DataFrame* rekomendasi

```
rekomendasi = df_original.iloc[final_indices][['id_buku', 'judul_buku',
'deskripsi', 'kategori', 'penulis', 'rating', 'tahun_terbit']].copy()
    rekomendasi['similarity_score'] = final_scores
    return rekomendasi
```

Mengambil informasi buku seperti judul, deskripsi, kategori, penulis, rating, dan tahun terbit untuk hasil rekomendasi akhir. Mengembalikan *DataFrame* yang berisi buku-buku yang direkomendasikan beserta *similarity score*.

- **Fungsi utama untuk interaksi**

```
# Fungsi utama untuk interaksi
def main():
    while True:
        user_input = input("\nMasukkan judul buku yang dicari ('exit'
untuk keluar): ")
        if user_input.lower() == 'exit':
            break

        recommendations = get_recom_stepwise(user_input)

        if recommendations is not None:
            print("\nRekomendasi Buku Berdasarkan Pencarian Anda:")
            print(recommendations.to_string(index=False))

# Jalankan program
if __name__ == "__main__":
    main()
```

1) Membuat *loop* interaksi pengguna

```
while True:
 user_input = input("\nMasukkan judul buku yang dicari ('exit' untuk keluar): ")
```

Program dimulai dengan *loop while True* yang berarti akan terus berjalan hingga pengguna memutuskan untuk keluar. Pengguna diminta memasukkan judul buku yang ingin dicari.

2) Menghentikan program

```
if user_input.lower() == 'exit':
 break
```

Jika pengguna mengetik *exit* (dalam bentuk huruf kecil atau besar), program akan berhenti dengan menggunakan perintah *break*.

3) Mencari rekomendasi berdasarkan *input* pengguna

```
recommendations = get_recom_stepwise(user_input)
```

Memanggil fungsi *get_recom_stepwise* dengan *input* pengguna untuk mencari rekomendasi buku. *Get_recom_stepwise* akan mengembalikan *DataFrame* yang berisi daftar rekomendasi buku atau *None* jika tidak ada buku yang cocok ditemukan.

4) Menampilkan hasil rekomendasi

```
if recommendations is not None:
 print("\nRekomendasi Buku Berdasarkan Pencarian Anda:")
 print(recommendations.to_string(index=False))
```

Jika fungsi *get_recom_stepwise* mengembalikan hasil (tidak *None*), maka hasil rekomendasi akan ditampilkan ke layar. *to_string(index=False)* digunakan untuk mencetak *DataFrame* tanpa menampilkan indeks baris, sehingga lebih rapi.

5) Memulai program

```
if __name__ == "__main__":
 main()
```

Bagian ini memastikan bahwa fungsi main hanya dijalankan jika *file* ini dieksekusi secara langsung (bukan diimpor sebagai modul di *file* lain). Kondisi if _name_ == "_main_" adalah pola umum dalam *Python* untuk mengontrol eksekusi skrip.

Pengguna:

```
Rekomendasi Buku Berdasarkan Pencarian Anda:
Masukkan judul buku yang dicari ('exit' untuk keluar): Rekayasa Perangkat Lunak
```

Output:

```
Rekomendasi Buku Berdasarkan Pencarian Anda:
 id_buku 386
judul_buku Rekayasa Perangkat Lunak Berbasis Objek
deskripsi          Buku ini mengulas rekayasa perangkat lunak berbasis objek, mencakup konsep
                   OOP, desain pattern, dan implementasi.
kategori           Rekayasa Perangkat Lunak
penulis                    Yususf Purnomo
```

```
rating              7
tahun_terbit     2016
similarity_score          1.0
```

Pengguna:

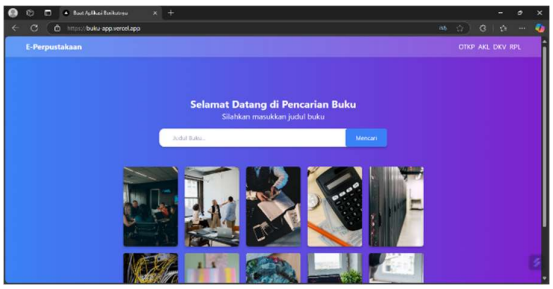
```
Masukkan judul buku yang dicari ('exit' untuk keluar): exit
```

Output:

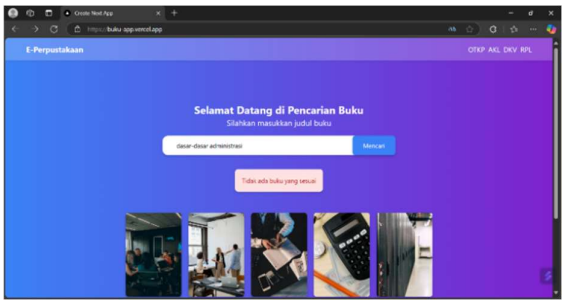
```
Program berhenti
```

Fungsi main berperan sebagai antarmuka pengguna sederhana untuk mengakses sistem rekomendasi buku. Pengguna dapat memasukkan judul buku yang ingin dicari, dan program akan menampilkan beberapa rekomendasi berdasarkan kesamaan konten dengan judul buku tersebut. Pengguna dapat terus mencari buku hingga mengetikkan *exit* untuk mengakhiri program.

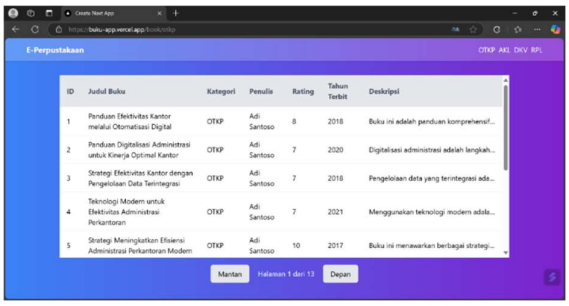
4.2 Tampilan Halaman



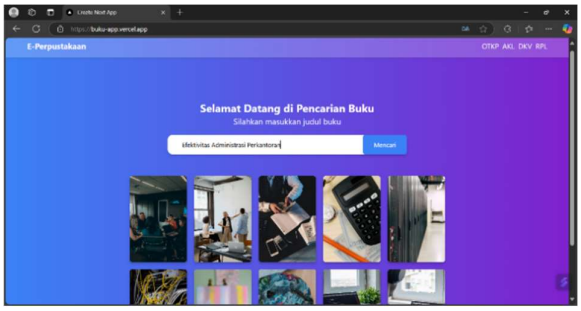
Gambar 7. Halaman Pertama Sistem Rekomendasi



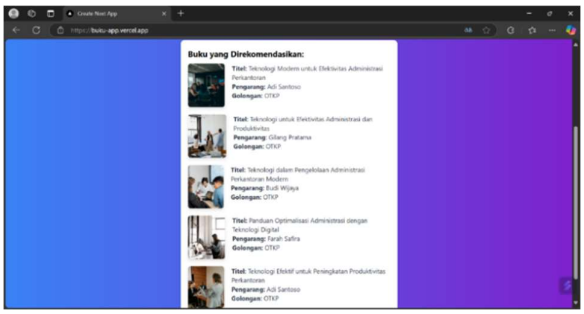
Gambar 8. Sistem Memberikan peringatan



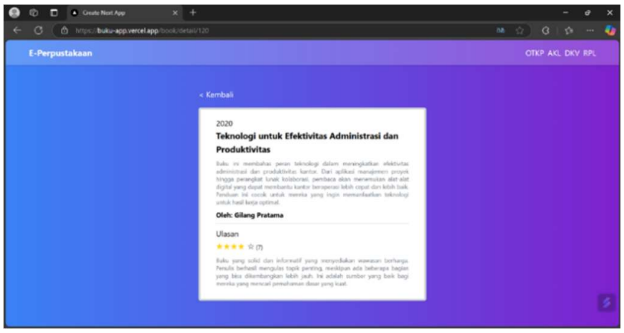
Gambar 9. Halaman Sistem Berdasarkan Jurusan



Gambar 10. Halaman Sistem Mencari Buku



Gambar 11. Halaman Sistem Merekomendasikan Buku



Gambar 12. Halaman Sistem Menampilkan Deskripsi Buku

5. KESIMPULAN

Pembuatan sistem rekomendasi dengan metode *Content-Based Filtering* yang merekomendasikan buku yang memiliki kemiripan dengan buku yang dicari. Tujuannya membangun sistem yang dapat memberikan rekomendasi buku berdasarkan kesamaan fitur konten dengan buku-buku yang telah disukai oleh pengguna sebelumnya. Maka dapat diambil kesimpulan sebagai berikut:

1) Sistem dapat memberikan rekomendasi buku yang sesuai dengan minat pengguna yang memiliki kemiripan dengan buku yang dicari yang tersedia pada *dataset.*
2) Kesamaan fitur buku, menggunakan metode seperti TF-IDF dan *cosine similarity* terbukti efektif dalam menghitung kesamaan antara buku dan memberikan rekomendasi yang akurat.

DAFTAR PUSTAKA

Zayyad, M. R. A. (2021). "Sistem Rekomendasi Buku Menggunakan Metode Content Based Filtering".

Ridwansyah, T., Subartini, B., & Sylviani, S. (2024). "Penerapan Metode Content-Based Filtering pada Sistem Rekomendasi". *Mathematical Sciences and Applications Journal*, *4*(2), 70-77.

Anggoro, T. (2023). *TA:* "Penerapan Metode Content-Based Filtering untuk Membangun Sistem Rekomendasi Buku pada Perpustakaan Universitas Dinamika". (Doctoral dissertation, Universitas Dinamika).

Rokhim, A., & Saikhu, A. (2017). "Sistem Rekomendasi Buku Pada Aplikasi Perpustakaan Menggunakan Metode Collaborative Filtering Pada Smkn 1 Bangil". *SPIRIT*, *8*(2).

Alkaff, M., Khatimi, H., & Eriadi, A. (2020). "Sistem Rekomendasi Buku Menggunakan Weighted Tree Similarity dan Content Based Filtering". *MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput*, *20*(1), 193-202.

Ardiansyah, R., Bianto, M. A., & Saputra, B. D. (2023). "Sistem Rekomendasi Buku Perpustakaan Sekolah menggunakan Metode Content-Based Filtering". *Jurnal CoSciTech (Computer Science and Information Technology)*, *4*(2), 510-518.

Ritdrix, A. H., & Wirawan, P. W. (2018). "Sistem Rekomendasi Buku Menggunakan metode item-based collaborative filtering". (Doctoral dissertation, Universitas Diponegoro).

WIDAYANTI, R., CHAKIM, M. H. R., LUKITA, C., AWALI, M. R., VALLEN, J., ANDINI, D., ... & NABILA, D. LAPORAN AKHIR PENELITIAN MANDIRI PENGEMBANGAN METODE FILTRASI KOLABORATIF UNTUK PENINGKATAN SISTEM REKOMENDASI.

Hendrayana, N., & Wibowo, JS (2024). "Sistem Rekomendasi Pencarian Buku Perpustakaan Dengan Algoritma Content Based Filtering". *Elkom: Jurnal Elektronika dan Komputer*, *17*(1), 271-278.

Badriyah, T., Fernando, R., & Syarif, I. (2018). "Sistem Rekomendasi Content Based Filtering Menggunakan Algoritma Apriori". *Konferensi Nasional Sistem Informasi (KNSI) 2018*.

Arfisko, H. H. (2022). "Sistem Rekomendasi Film Menggunakan Metode Hybrid Collaborative Filtering Dan Content-Based Filtering". *E-Proceeding of Engineering*, *9*(Vol.9, No.3(2022): Juni 2022), 2149. https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/18066

Rabbani, S., Safitri, D., Rahmadhani, N., Sani, A. A. F., & Anam, M. K. (2023). "Perbandingan Evaluasi Kernel SVM untuk Klasifikasi Sentimen dalam Analisis Kenaikan Harga BBM". *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *3*(2), 153–160. https://doi.org/10.57152/malcom.v3i2.897

Rahmadani, R., Rahim, A., Muhammadiyah, U., Timur, K., Muhammadiyah, U., Timur, K., Muhammadiyah, U., & Timur, K. (2025). "ANALISIS SENTIMEN ULASAN "OJOL THE GAME" DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA NAÏVE BAYES DAN MODEL EKSTRAKSI FITUR TF-IDF". *JITET (Jurnal Informatika Dan Teknik Elektro Terapan)*, *12*(3).

Sulaeman, A. S., Sujjada, A., & Kharisma, I. L. (2024). "Penerapan Algoritma Cerdas Bidirectional Encoder Refresentations From Transformers Dalam Menganalisis Opini Publik Terhadap Produk Yang Mengalami Boikot". *INOVTEK Polbeng - Seri Informatika*, *9*(1), 460–473. https://doi.org/10.35314/isi.v9i1.4252

Wulandari, Y. T., & Rosandy, T. (2024). "Implementasi Computer Vision Dalam Sistem Deteksi Gerakan Disiplin Kampus". *Ijccs*, *x*, *No.x*(x), 1–5.