SISTEM REKOMENDASI BUKU MENGGUNAKAN METODE CONTENT BASED FILTERING

Disusun Oleh:

N a m a : Muhammad Rizqi Az Zayyad

NIM : 17523144

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2021

HALAMAN PENGESAHAN DOSEN PEMBIMBING

SISTEM REKOMENDASI BUKU MENGGUNAKAN METODE CONTENT BASED FILTERING

TUGAS AKHIR

Disusun Oleh:

N a m a : Muhammad Rizqi Az Zayyad
NIM : 17523144

Yogyakarta, 13 Juli 2021
Pembimbing,

(Arrie Kurniawardhani, S.Si., M.Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

SISTEM REKOMENDASI BUKU MENGGUNAKAN METODE CONTENT BASED FILTERING

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 13 Juli 2021

Tim Penguji

Arrie Kurniawardhani, S.Si., M.Kom. _____

Anggota 1

Septia Rani, S.T., M.Cs. _____

Anggota 2

Aridhanyati Arifin, S.T., M.Cs. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Muhammad Rizqi Az Zayyad

NIM : 17523144

Tugas akhir dengan judul:

SISTEM REKOMENDASI BUKU MENGGUNAKAN METODE CONTENT BASED FILTERING

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 13 Juli 2021

(Muhammad Rizqi Az Zayyad)

HALAMAN PERSEMBAHAN

Alhamdulillah, Segala puji bagi Allah *Subhanahu wata'ala* dengan segala nikmat yang telah diberikan, kepada-Nya kita memuji, meminta pertolongan serta meminta petunjuk. Dengan izin dari Allah *Subhanahu wata'ala* juga, kita diberikan nikmat iman dan Islam hingga saat ini. Shalawat serta salam selalu tercurahkan kepada Nabi Muhammad *Sallallahu 'alaihi wasallam* yang telah membimbing kita kepada cahaya Islam.

Terima kasih yang tak terhingga saya sampaikan kepada kedua orangtua saya yang telah memberikan kesempatan dan bimbingan yang luar biasa kepada saya untuk dapat menuntut ilmu dari awal hingga saat ini.

Terima kasih juga kepada Ibu Arrie Kurniawardhani, S.Si., M.Kom. yang telah memberikan bimbingan kepada saya untuk dapat menyelesaikan laporan tugas akhir.

Terima kasih juga kepada semua teman-teman dan orang-orang di sekitar saya yang memberikan support dan doa kepada saya dalam menyelesaikan tugas akhir ini.

HALAMAN MOTO

“Bacalah, dengan (menyebut) nama Tuhanmu yang menciptakan.” (QS. Al-Alaq: 1)

A little progress each day adds up to big results

“Belajar tanpa berpikir itu tidaklah berguna, tapi berpikir tanpa belajar itu sangatlah berbahaya!” (Ir. Soekarno)

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillahrabbil 'alamin, segala puji bagi Allah *Subhanahu wata'ala* yang telah memberikan berbagai macam nikmat, baik nikmat iman, Islam, dan juga kesehatan sehingga penulis dapat menyelesaikan tugas akhir dengan baik. Shalawat serta salam selalu tercurahkan kepada Nabi Muhammad *Sallallahu 'alaihi wasallam* yang telah membawa ajaran Islam dari zaman *jahiliyyah* menuju zaman yang penuh dengan ilmu dan cahaya Islam.

Tugas akhir ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Strata 1 (S1) pada Program Studi Informatika, Fakultas Teknologi Industri. Selama proses pengerjaan penelitian ini, penulis mendapatkan hambatan dan rintangan. Namun, dalam pengerjaanya, penulis juga mendapatkan bimbingan, support dan doa dari berbagai pihak. Oleh karena itu, penulis ingin memberikan apresiasi dan ucapan terima kasih kepada:

1. Kedua orangtua yang selalu memberikan semangat, dukungan dan doanya selama ini.
2. Dr. Raden Teguh Dirgahayu, S.T., M.Sc. selaku Ketua Program Studi Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.
3. Arrie Kurniawardhani, S.Si., M.Kom. selaku pembimbing yang telah memberikan bimbingan serta arahan selama penelitian ini berlangsung.
4. Bapak dan Ibu dosen Jurusan Informatika yang telah memberikan ilmu yang sangat bermanfaat, semoga bapak dan ibu selalu diberikan keberkahan oleh Allah SWT.
5. Teman-teman semua yang selalu memberikan support dan semangatnya.

Semoga semua pihak yang telah memberikan doa, dukungan, dan bimbingan selama ini diberikan kemudahan dan perlindungan dari Allah *Subhanahu Wata'ala*. Penulis juga memohon maaf karena dalam penelitian ini masih terdapat banyak kekurangan. Semoga penelitian ini dapat memberikan manfaat dan dapat menjadi sumber ilmu untuk menambah wawasan kita semua.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Yogyakarta, 13 Juli 2021

(Muhammad Rizqi Az Zayyad)

SARI

Buku merupakan sumber informasi dan ilmu pengetahuan yang dapat meningkatkan wawasan tentang berbagai macam hal. Tersebarnya informasi mengenai beragam jenis buku terkadang menyulitkan seseorang untuk mencari buku yang ingin dibaca berdasarkan kriteria yang disukai. Salah satu cara untuk mengelola informasi tersebut dapat dilakukan dengan menggunakan sistem rekomendasi. Sistem rekomendasi telah dimanfaatkan sebagai strategi yang efektif untuk dapat mengelola banyaknya informasi yang tersedia dan memberikan rekomendasi suatu *item* sesuai dengan keinginan pengguna. Berbagai macam industri juga telah menggunakan sistem rekomendasi untuk meningkatkan performa sistem kepada konsumen. Berbagai macam industri juga telah menggunakan sistem rekomendasi untuk mengelola banyaknya informasi yang tersebar luas. Tujuan dari penelitian ini adalah untuk membuat sebuah sistem yang dapat memberikan rekomendasi buku kepada pengguna sesuai dengan buku yang diminati sebelumnya. Data buku yang digunakan berjumlah 5143 data buku yang berasal dari situs Gramedia. Metode *content based filtering* digunakan pada penelitian ini. Untuk melakukan pembobotan dan menghitung kemiripan tiap data buku, peneliti menggunakan algoritma TF-IDF dan *cosine similarity*. Berdasarkan hasil dari pengujian sistem yang telah dibangun, sistem dapat memberikan rekomendasi berdasarkan kemiripan dari setiap buku dan menghasilkan nilai *precision* sebesar 85%.

Kata kunci – *content based filtering*, *cosine similarity*, sistem rekomendasi.

GLOSARIUM

Corpus	kumpulan teks yang tersimpan secara elektronik untuk kebutuhan penelitian.
Framework	kerangka program yang digunakan untuk membantu *developer* dalam pengembangan sistem.
Library	kumpulan program yang digunakan untuk mengembangkan sistem.
Scraping	langkah untuk mengambil data dari internet.
Preprocessing	proses pembersihan data mentah menjadi data bersih yang dapat diolah.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Membaca buku merupakan salah satu cara bagi setiap orang untuk mendapatkan ilmu pengetahuan dan wawasan baru mengenai banyak hal. Informasi mengenai beragam jenis buku saat ini telah tersebar luas di internet sehingga dapat diakses oleh banyak orang. Meski informasi mengenai beragam jenis buku telah beredar di internet, namun minat baca di Indonesia masih tergolong rendah. Menurut data riset bertajuk *World's Most Literate Nations Ranked* yang dilakukan oleh *Central Connecticut State Univesity* pada tahun 2016, Indonesia dinyatakan menduduki peringkat ke-60 dari 61 negara soal minat membaca (Devega, 2017). Hasil riset ini menunjukkan bahwa minat baca di Indonesia masih terbilang cukup rendah dibandingkan dengan negara lain termasuk dari Singapura dan Malaysia. Kondisi ini tentu akan berpengaruh terhadap pemahaman dalam ilmu pengetahuan serta teknologi untuk dapat menghasilkan produk-produk berkualitas. Selain itu, efek yang lebih besar atas keengganan untuk minat membaca pada generasi muda ini akan merugikan negara yang dapat kehilangan aset penyumbang dari segi sumber daya manusia untuk kemajuan bangsa yang berkualitas dan mempunyai produktifitas yang tinggi (Witanto, 2018).

Salah satu upaya untuk dapat meningkatkan minat baca kepada masyarakat adalah dengan menyediakan bahan bacaan yang variatif, sehingga mendukung pembelajaran dan mendorong kepada masyarakat untuk menyukai buku (Witanto, 2018). Hingga saat ini, terdapat banyak sekali media yang menyediakan layanan kepada masyarakat untuk mendapatkan bacaan buku yang variatif, seperti Gramedia, Togamas, Berdikari dan lain sebagainya. Masyarakat dapat mengakses beragam jenis buku yang tersedia pada penyedia layanan buku tersebut. Dengan banyaknya informasi buku yang tersebar luas dan untuk memudahkan pembaca buku dalam menemukan buku yang sesuai dengan kriteria yang diinginkan, penerapan sistem rekomendasi dapat dimanfaatkan. Sehingga pengguna dapat dengan mudah mendapatkan referensi baru mengenai buku yang bisa dibaca sesuai dengan kategori buku yang diminati oleh pembaca sebelumnya. Selain itu, sistem rekomendasi juga memiliki peran penting pada banyak layanan aplikasi *online*. Performa sistem rekomendasi dapat berdampak pada kesuksesan komersial pada banyak perusahaan dalam hal pendapatan dan kepuasan para pengguna (Wei, He, Chen, Zhou, & Tang, 2016).

Sistem rekomendasi memiliki tiga kategori model yang dapat digunakan, yaitu *Content Based Filtering*, *Collaborative Filtering*, dan *Hybrid Recommender System* (Zhang, Yao, Sun, & Tay, 2018). *Collaborative Filtering* digunakan untuk mengidentifikasi kesamaan antar pengguna dan memberikan rekomendasi item yang sesuai. Sistem ini merekomendasikan item yang disukai oleh kelompok dari pengguna lain yang serupa. *Collaborative Filtering* dibagi menjadi dua kategori yaitu *memory based* yang digunakan untuk menilai berdasarkan kesamaan antara pengguna atau item dan *model based* untuk memprediksi peringkat pengguna dari item yang tidak diberi peringkat. *Content Based Filtering* merupakan sistem yang memberikan rekomendasi kepada pengguna berdasarkan pada preferensi pengguna dan hubungan antar deskripsi item. Hal itu dilakukan dengan memilih item yang paling mirip dengan item yang disukai oleh pengguna. *Hybrid Recommender System* adalah kombinasi antara teknik *collaborative filtering* dan *content based*. Akurasi dari sistem ini secara umum lebih tinggi dari kedua sistem sebelumnya. Hal ini dikarenakan *collaborative filtering* memiliki kekurangan yaitu, ketergantungan pada preferensi pengguna lain dan *content based* yang memiliki permasalahan dalam memberikan rekomendasi item yang lebih kompleks (Zhang, Yao, Sun, & Tay, 2018).

Berdasarkan uraian di atas, penulis tertarik untuk membuat sebuah sistem yang dapat memberikan referensi baru kepada pengguna dalam manambah opsi pengetahuan mengenai buku-buku yang memiliki kemiripan atau kesamaan dari setiap buku. Data yang digunakan untuk membuat sistem rekomendasi ini didapatkan dari salah satu penyedia layanan untuk jual beli buku, yaitu Gramedia. Toko buku tersebut merupakan salah satu yang terbesar dan memiliki situs dengan berbagai macam buku yang disediakan untuk para pembaca. Buku yang tersedia pada situs tersebut juga sangat beragam, mulai dari buku untuk pengembangan diri, teknologi dan informasi, komik, novel, sejarah, dan lain sebagainya. Banyaknya informasi dan buku yang tersedia pada situs tersebut, membuat para pembaca memerlukan waktu untuk mencari dan menemukan buku yang diinginkan. Sebagai penyedia layanan jual beli buku seperti Gramedia, perlu memiliki sebuah sistem yang dapat memberikan kemudahan bagi para pembaca untuk mencari buku yang diminati oleh pengguna. Setiap buku yang tersedia pada situs tersebut memiliki keterkaitan antara satu buku dengan yang lainnya, baik dari segi genre, deskripsi, penulis, dan judul dari buku tersebut. Hal ini dapat dimanfaatkan untuk membuat sebuah sistem rekomendasi yang dapat memudahkan pengguna dalam mencari buku yang sesuai dengan jenis buku favoritnya.

Sistem rekomendasi menjadi salah satu solusi yang dapat menyelesaikan masalah dalam pencarian buku yang sesuai dengan keinginan para pembaca dan menambah referensi pengetahuan terhadap buku lain yang serupa. Dengan menggunakan metode *content based filtering* yang memanfaatkan *cosine similarity*, sebuah sistem dapat mengukur kemiripan pada setiap buku dengan menggunakan perhitungan antar vektor (Rymmai & JS, 2017). Penerapan metode *content based filtering* memanfaatkan konten yang terdapat pada setiap buku untuk diukur tingkat kemiripannya dengan buku-buku yang lain. Dengan mengukur kemiripan dari setiap buku, sistem dapat memberikan skor yang menjadi acuan dalam memberikan rekomendasi sesuai dengan deskripsi dari buku tersebut. Melalui sistem rekomendasi ini, pembaca buku diharapkan dapat menambah referensi baru berdasarkan buku-buku yang diminati sebelumnya.

1.2 Rumusah Masalah

Rumusan masalah dari penelitian ini adalah sebagai berikut:

a. Bagaimana membangun sebuah sistem rekomendasi berdasarkan data dari buku-buku yang tersedia.
b. Bagaimana performa metode *Content Based Filtering* dalam memberikan rekomendasi buku berdasarkan tingkat kemiripan antar item.

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah:

a. Data yang digunakan pada sistem ini berasal dari data buku Gramedia.
b. Data yang digunakan berbahasa Indonesia dan sebagian berbahasa Inggris.
c. Sistem rekomendasi diimplementasikan dengan menggunakan bahasa pemrograman Python.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah sebagai berikut:

a. Membangun sebuah sistem rekomendasi buku dengan menggunakan metode *Content Based Filtering*.
b. Mengetahui keakuratan *Content Based Filtering* dalam memberikan rekomendasi buku.

1.5 Manfaat Penelitian

Manfaat dari penilitian ini diantaranya adalah sebagai berikut:

a. Memudahkan pembaca buku dalam mencari rekomendasi buku yang diminati.
b. Meningkatkan referensi dan pengetahuan mengenai buku yang memiliki kemiripan dengan buku yang diminati oleh pembaca.
c. Meningkatkan minat baca kepada masyarakat berdasarkan buku-buku yang disukai.

1.6 Sistematika Penulisan

Dalam memberikan sebuah gambaran yang jelas mengenai pembahasan dalam pembuatan sistem rekomendasi buku menggunakan *content based filtering*, berikut ini adalah sistematika pembahasannya, yaitu:

BAB I PENDAHULUAN

Bab pendahuluan ini membahas mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, metode penelitian dan sistematika penulisan dari tugas akhir sistem rekomendasi buku menggunakan metode *content based filtering*.

BAB II LANDASAN TEORI

Bab ini akan menjadi landasan untuk mengarahkan dan mendukung proses pengembangan sistem ini. Pada bab ini juga akan menjelaskan mengenai uraian singkat hasil penelitian sebelumnya yang berkaitan dengan topik penelitian ini.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang analisis kebutuhan yang dilakukan untuk dapat menghasilkan sebuah sistem rekomendasi. Pada bab ini juga akan dibahas mengenai metode yang digunakan untuk mengambil data dan mengimplementasikannya ke dalam sistem rekomendasi yang dibuat.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang penjelasan mengenai pembahasan rancangan sistem yang dibuat dan hasil yang didapatkan setelah melalui tahapan penelitian yang dilakukan pada bab sebelumnya. Selain itu, pada bab ini juga akan membahas mengenai hasil dari pengujian sistem rekomendsasi yang telah dibuat.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi mengenai kesimpulan dan saran yang dapat diambil dari hasil penelitian atau pembahasan tugas akhir secara keseluruhan untuk diajukan pada penelitian dan pengembangan sistem kedepannya.

BAB II
LANDASAN TEORI

2.1 Dasar Teori

2.1.1 Sistem Rekomendasi

Sistem rekomendasi merupakan suatu aplikasi yang digunakan untuk memberikan rekomendasi item dalam membuat suatu keputusan yang diinginkan pengguna. Sistem rekomendasi juga disebut sebagai mesin rekomendasi yang dapat membantu dalam membuat saran kepada pelanggan berdasarkan data yang tersedia, dengan menganalisis apa yang mungkin diminati pengguna (Rymmai & JS, 2017). Terdapat beberapa faktor yang dapat mempengaruhi keputusan sistem dalam memberikan rekomendasi kepada pengguna, seperti perilaku pengguna, deskripsi item, hingga preferensi dan kebiasaan dari sekelompok pengguna yang memiliki kemiripan dalam menilai suatu item. Sistem rekomendasi telah menjadi alat yang efektif untuk menyaring informasi. Dalam arti lain, sistem rekomendasi menekankan pada karakteristik penyaringan informasi. Yaitu, menyaring produk yang mungkin menarik bagi pengguna (Shao, Li, & Bian, 2020). Sistem rekomendasi telah digunakan oleh banyak perusahaan di dunia, seperti Amazon, Netflix, hingga YouTube yang telah banyak digunakan oleh para pengguna di internet. Sistem rekomendasi membantu pengguna untuk menemukan dan memilih item (seperti buku, film, restoran) dari banyak koleksi yang tersedia di website atau dari sumber informasi elektronik lainnya. Di antara banyak dari sekumpulan item dan deskripsi, sistem rekomendasi menyediakan kepada pengguna sebuah set item yang sesuai dengan kebutuhan pengguna (G, M, C, & D, 2018).

Sistem rekomendasi terus mengalami perkembangan dan secara umum dibagi menjadi tiga kategori metode yang dapat digunakan dengan fungsi yang berbeda (Zhang, Yao, Sun, & Tay, 2018). Pertama yaitu *content based*, kedua *collaborative filtering*, dan yang ketiga yaitu *hybrid recommender system*. *Content based filtering* memberikan rekomendasi berdasarkan pada perbandingan antar item dan informasi tambahan lainnya. Beragam informasi tambahan seperti teks, gambar, dan video dapat diperhitungkan. *Collaborative filtering* merupakan sebuah sistem rekomendasi yang belajar dari interaksi historis item pengguna, baik eksplisit (misalnya peringkat dari pengguna sebelumnya) atau masukan implisit (misalnya riwayat penelusuran). Model *hybrid recommender system* merupakan gabungan dari kedua model sistem rekomendasi *content based* dan *collaborative filtering*.

2.1.2 Content Based Filtering

Sistem rekomendasi berbasis konten ini memberikan rekomendasi kepada pengguna berdasarkan pada profil preferensi pengguna dan hubungan antar deskripsi item. Hal itu dilakukan dengan membangun hubungan antar item dan propertinya dalam bentuk matriks kemudian memilih item yang paling mirip dengan item target dengan menghitung *similarity* berdasarkan fitur yang terkait dengan item yang dibandingkan dengan menggunakan berbagai fungsi matematika. *Content based filtering* memeriksa item yang disukai sebelumnya dan merekomendasikan item lain yang paling cocok. Fungsi kesamaan yang paling umum digunakan adalah *Adjusted Cosine*, *Cosine* atau *Pearson Coefficient*. Persamaan yang baik akan menghasilkan kualitas prediksi yang tinggi (Kaddam & Kumar, 2016).

2.1.3 Buku

Buku menurut Kamus Besar Bahasa Indonesia Balai Pustaka adalah lembar kertas yang berjilid, berisi tulisan atau kosong. Sedangkan menurut *Oxford Dictionary*, buku merupakan hasil karya yang ditulis atau dicetak dengan halaman-halaman yang dijilid pada satu sisi atau hasil karya yang ditujukan untuk penerbitan. Buku yang dianggap berhasil jika dapat menggugah minat dari khalayak sasaran dalam memahami isi dari buku tersebut. Untuk mendukung keberhasilan sebuah buku diperlukan sebuah desain yang dapat mencerminkan maksud dan tujuan tersebut (dosenpendidikan, 2021).

2.1.4 Web Scraping

Web Scraping adalah teknik penting yang digunakan untuk mengekstraksi data tidak terstruktur dari situs web dan mengubah data itu menjadi terstruktur. *Web scraping* juga diidentifikasi sebagai *web data extraction*, *web data scraping*, *web harvesting* atau *screen scrapping*. *Web scraping* merupakan bentuk dari *data mining*. Dasar dan tujuan penting dari proses *web scraping* adalah untuk menambang informasi dari situs web yang berbeda dan tidak terstruktur dan mengubahnya menjadi struktur yang dapat dipahami seperti *spreadsheet*, *database* atau file *comma-separated values* (CSV). Data seperti penetapan harga barang, penetapan harga saham, laporan yang berbeda, penetapan harga pasar, dan detail produk, dapat dikumpulkan melalui *web scraping*. Dengan w*eb* scrapping, informasi dapat di ekstrak dari situs web dan dapat digunakkan untuk mengambil keputusan yang efektif dalam proses bisnis (Saurkar, Pathare, & Gode, 2018).

Scraping adalah teknik yang digunakan untuk memotong informasi dari halaman web berdasarkan skrip. Halaman web merupakan dokumen yang ditulis dalam *HyperText Markup Language* (HTML), dan yang terbaru adalah *Extensible HyperText Markup Language* (XHTML) berbasis XML. Dokumen web direpresentasikan oleh struktur pohon yang disebut dengan *Document Object Model*, atau biasa disebut dengan DOM dan tujuan HTML adalah untuk menentukan format teks yang dapat ditampilkan oleh *web browser* (Saurkar, Pathare, & Gode, 2018).

2.1.5 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF merupakan implementasi dari statistik numerik yang menunjukkan relevansi kata kunci dengan beberapa dokumen tertentu, dengan menyediakan kata kunci yang tersedia, beberapa dokumen tertentu dapat diidentifikasi atau dikategorikan sesuai dengan relevansinya (Qaiser & Ali, 2018). TF-IDF merupakan gabungan dari dua kata yang berbeda, yaitu TF (*Term Frequency*) dan IDF (*Inverse Document Frequency*). *Term Frequency* digunakan untuk mengukur berapa kali suatu term muncul dalam suatu dokumen. Misalnya dalam sebuah dokumen “A7” yang berisi 1000 kata dan terdapat kata “Informasi” dalam dokumen tersebut sebanyak 10 kali. Perlu diketahui bahwa panjang total dokumen dapat bervariasi dari sangat kecil hingga besar, sehingga ada kemungkinan bahwa istilah apapun dapat muncul lebih sering dalam dokumen yang lebih besar dibandingkan dengan dokumen kecil (Qaiser & Ali, 2018). Jadi, untuk memperbaiki masalah ini, kemunculan istilah apapun dalam dokumen dibagi dengan total istilah yang ada dalam dokumen itu, untuk menemukan *term frequency*. Maka, dalam hal ini *term frequency* kata "Informasi" dalam dokumen "A7" adalah TF = 10/1000 = 0.001.

Sedangkan, untuk *Inverse Document Frequency* (IDF) adalah untuk mengukur seberapa penting kata tersebut di dalam dokumen ketika *term frequency* pada dokumen dihitung, dapat diamati bahwa algoritma memperlakukan semua kata kunci secara setara, tidak masalah jika kata itu termasuk *stopwords* seperti "dari". Semua kata kunci memiliki nilai penting yang berbeda. Katakanlah, jika kata *stopwords* "dari" muncul dalam dokumen sebanyak 2000 kali tetapi tidak terlalu memiliki arti, ditulah fungsi adanya IDF. *Inverse document frequency* memberikan bobot yang lebih rendah untuk kata-kata yang sering muncul dan memberikan bobot lebih besar untuk kata-kata yang jarang muncul (Qaiser & Ali, 2018). Contoh dari perhitungan IDF adalah, jika kita memiliki 10 dokumen dan kata "Teknologi" muncul dalam 5 dokumen tersebut, maka ukuran *inverse document frequency* (IDF) = log(10/5) = 0.3010.

Algoritma TF-IDF umum digunakan sebagai metode untuk pengukuran pada deskripsi suatu item tekstual dalam pendekatan *Content Based Filtering* (Lops, Jannach, Musto, Bogers, & Koolen, 2019). Peran dari algoritma TF-IDF sangat berguna pada banyak hal seperti kebutuhan untuk *content based filtering*, *text mining techniques*, dan *information retrieval* lainnya (Chen, 2018). Persamaan (2.1) adalah digunakan untuk melakukan penghitungan *term frequency* (Oeyliawan & Gunawan, 2017).

$$tf_{i,j} = f_{i,j} \quad (2.1)$$

TF adalah *term frequency*, dan $tf_{i,j}$ merupakan banyaknya jumlah kemunculan term t_i dalam dokumen d_j, *Term frequency* (*tf*) dihitung dengan menghitung banyaknya kemunculan term t_i dalam dokumen d_j. Selanjutnya untuk menghitung *inverse document frequency* dapat dilakukan dengan menggunakan persamaan (2.2).

$$idf_i = log\left(\frac{N}{df_i}\right) \quad (2.2)$$

Dengan idf_i adalah *inverse document frequency*, N adalah jumlah dokumen yang terambil oleh sistem, dan df_i adalah banyaknya jumlah dokumen dalam koleksi dimana term t_i muncul di dalamnya, maka perhitungan idf_i digunakan untuk mengetahui banyaknya term yang dicari (df_i) yang muncul dalam dokumen lain (Oeyliawan & Gunawan, 2017).

Perhitungan bobot dari *term* dalam sebuah dokumen menggunakan perkalian nilai *tf* dan *idf* menunjukkan bahwa deskripsi terbaik dari dokumen adalah term yang banyak muncul dalam dokumen tersebut dan sangat sedikit muncul pada dokumen yang lain. Pada persamaan (2.3) dilakukan untuk menghitung bobot *term frequency – inverse document frequency*.

$$W_{i,j} = tf_{i,j} log \frac{N}{df_i} \quad (2.3)$$

2.1.6 Cosine Similarity

Metode *Cosine Similarity* adalah mengukur kemiripan antara dua dokumen atau teks. Pada *cosine similarity* dokumen atau teks dianggap sebagai vektor (Samuel, Natan, Fitria, & Syafiqoh, 2018). Dalam pengertian lain, *cosine similarity* antara dua vektor (atau dua

dokumen pada *vector space*) merupakan pengukuran yang menghitung antara sudut kosinus di antara keduanya. Metrik ini adalah pengukuran orientasi dan bukan besarannya, dapat dilihat sebagai perbandingan antar dokumen pada ruang yang dinormalisasi karena tidak hanya besaran setiap hitungan kata (TF-IDF) yang dipertimbangkan dari setiap dokumen, tetapi sudut antar dokumen-dokumen (Rymmai & JS, 2017). Untuk membangun persamaan *cosine similarity* adalah dengan menggunakan persamaan (2.4).

$$Sim(q, d_j) = \frac{q \times d_j}{|q| \times |d_j|} = \frac{\sum_{i=1}^{n} W_{i,q} \times W_{i,j}}{\sqrt{\sum_{i=1}^{t} (W_{i,q})^2} \sqrt{\sum_{i=1}^{t} (W_{i,j})^2}} \quad (2.4)$$

Dari persamaan diatas *Sim*(q,d_j) adalah similaritas antara *query* dan dokumen. |q| adalah jarak *query*. |*dj*| adalah jarak dokumen. $W_{i,j}$ adalah bobot dokumen ke-i. $W_{i,q}$ adalah bobot *query* dokumen ke-i. Similaritas antara *query* dan dokumen atau berbanding lurus terhadap hasil perkalian jumlah bobot *query* (q) dengan bobot dokumen (d_j) dan berbanding terbalik terhadap perkalian dari akar jumlah kuadrat q(|q|) dengan akar jumlah kuadrat dokumen $|d_j|$. Hasil dari perhitungan similaritas akan menghasilkan bobot dokumen yang mendekati nilai 1 (Oeyliawan & Gunawan, 2017).

2.1.7 Precision

Precision adalah suatu perbandingan antara jumlah hasil yang didapatkan secara relevan dengan jumlah hasil keseluruhan baik yang relevan atau tidak relevan. *Precision* biasanya digunakan untuk mengukur kinerja sistem. *Precision* juga dapat dikatakan sebagai perbandingan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem.

$$Precision = \frac{TP}{FP + TP} . 100\% \quad (2.5)$$

Dimana TP adalah *True Positive* (data relevan yang terambil dalam sistem rekomendasi), FP adalah *False Positive* (data tidak relevan yang terambil dalam sistem rekomendasi). Melalui perhitungan ini, sistem rekomendasi dapat diukur performanya berdasarkan *precision*.

2.2 Penelitian Terkait

Penelitian mengenai sistem rekomendasi buku beberapa kali telah dilakukan oleh peneliti lain. (Rymmai & JS, 2017) melakukan penelitian yang berjudul Book Recommendation Using *Cosine Similarity*. Penelitian ini menerapkan metode *content based filtering* untuk merekomendasikan sebuah buku berdasarkan ringkasan plot dari buku yang telah dicari pengguna. Ringkasan plot pada dasarnya memberikan deskripsi singkat tentang buku ini. Buku yang dirujuk berdasarkan judul, penulis, genre, dan peringkat buku. Ringkasan plot dapat digunakan untuk mengukur tingkat kemiripan dari setiap buku. Makalah ini menjelaskan bagaimana sistem rekomendasi dengan menggunakan *cosine similarity* dapat dicapai dengan menghitung tingkat kemiripan antar plot berdasarkan *terms* yang digunakan.

Penelitian tentang sistem rekomendasi buku pernah dilakukan sebelumnya. (Oeyliawan & Gunawan, 2017) melakukan sebuah penelitian dengan judul Aplikasi Rekomendasi Buku pada Katalog Perpustakaan Universitas Multimedia Nusantara Menggunakan Vector Space Model. Penelitian tersebut menggunakan data buku yang berasal dari koleksi di perpustakaan UMN dengan jumlah 9.804 buku. Sistem rekomendasi yang dikembangkan menggunakan *Vector Space Model* untuk merepresentasikan dokumen dalam model vector. Rekomendasi dalam aplikasi ini didasarkan pada tingkat kemiripan antar deskripsi dari tiap buku. Pada sistem ini juga menggunakan *cosine similarity* untuk mengukur tingkat kemiripan dari tiap buku. Pengujian sistem rekomendasi berbasis konten tersebut dilakukan dengan menggunakan sampel 158 buku yang diambil. Hasil dari pengujian tersebut menunjukkan nilai F-Measure sebesar 42,3% pada sampel buku dengan dua bahasa dan 55% pada sampel buku dengan satu bahasa.

Penelitian sistem rekomendasi buku menggunakan algoritma Apriori juga pernah dilakukan. (Prabowo & Ramdani, 2020) melakukan penelitian yang berjudul Penerapan Algoritma Apriori Untuk Rekomendasi Buku Pada Amikom Resource Center. Penelitian tersebut dilakukan dengan membuat sebuah sistem rekomendasi buku berdasarkan data transaksi peminjaman buku dan menggunakan algoritma apriori untuk membentuk aturan asosiasi (*association rule*) antar buku. Hasil penelitian tersebut menunjukkan bahwa *association rule* yang terbentuk dari 562 data transaksi peminjaman buku dengan menggunakan minimal frekuensi buku 4 atau nilai minimal *support* 0.7% dan minimal *confidence* 80% menghasilkan 10 aturan asosiasi dengan keseluruhan aturan memiliki tingkat korelasi positif sehingga dapat digunakan sebagai acuan untuk pemberian rekomendasi buku.

Penelitian sistem rekomendasi buku menggunakan data dari perpustakaan daerah Provinsi Kalimantan Selatan pernah dilakukan sebelumnya. (Alkaff, Khatimi, & Eriady, 2020) melakukan penelitian yang berjudul Sistem Rekomendasi Buku Menggunakan Weighted Tree Similarity dan Content Based Filtering. Penelitian sistem rekomendasi buku ini menggunakan data dari Perpustakaan Daerah Provinsi Kalimantan Selatan yang merupakan salah satu perpustakaan dan pusat penyedia layanan informasi di Kalimantan Selatan. Penelitian ini juga dilandasi oleh pengunjung perpustakaan yang kesulitan dalam mencari buku yang berkaitan dengan buku yang dipilih sebelumnya dan juga dalam menemukan alternatif buku lain ketika buku yang diinginkan tersebut telah dipinjam. Dengan adanya rekomendasi atau saran buku-buku lain yang berhubungan diharapkan membantu dalam mendapatkan buku yang sesuai dan diinginkan pengunjung perpustakaan. Pada penelitian ini menerapkan sistem rekomendasi menggunakan metode *Content Based Filtering* dalam memberikan rekomendasi buku yang bekerja dengan melihat kemiripan item yang dianalisis dari fitur yang dikandungnya dengan *Weighted Tree Similarity*. Berdasarkan hasil pengujian yang telah dilakukan pada 5 skenario pengujian yang diujikan, maka dihasilkan nilai *precision* sebesar 88%.

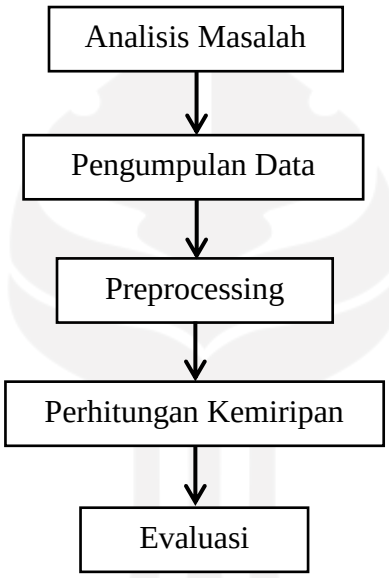
Penelitian untuk membuat sistem rekomendasi penyedia jurnal dan konferensi juga pernah dilakukan sebelumnya. (Wang, Liang, Xu, Feng, & Guan, 2018) melakukan penelitian untuk membantu penulis memutuskan di mana mereka harus mengirimkan manuskrip mereka, peneliti menyajikan *Content-Based Journals & Conferences Recommender System* tentang ilmu computer. Sistem ini merekomendasikan jurnal atau konferensi yang sesuai dengan urutan prioritas berdasarkan abstrak manuskrip. Untuk mengikuti pesatnya perkembangan ilmu dan teknologi komputer, digunakan *web crawler* untuk terus mengupdate *training set* dan *learning model*. Untuk mencapai respons online interaktif, peneliti mengusulkan model *hybrid* berdasarkan seleksi fitur *chi-square* dan *softmax regression*. Hasil pengujian menunjukkan bahwa, sistem dapat mencapai akurasi 61,37% dan menyarankan jurnal atau konferensi terbaik rata-rata dalam waktu sekitar 5 detik.

Dari hasil penelitian diatas, maka dapat diketahui bahwa metode *content based filtering* pada sistem rekomendasi memiliki beberapa teknik yang berbeda. Beberapa metode yang umum digunakan adalah dengan menggunakan teknik pembobotan TF-IDF dan *Cosine Similarity* untuk mengukur tingkat kemiripan antar dokumen. Hasil dari beberapa *review* penelitian di atas menjadi acuan peneliti untuk membuat sebuah sistem rekomendasi buku berbasis konten dengan menggunakan langkah-langkah dari penelitian sebelumnya.

BAB III
METODOLOGI PENELITIAN

3.1 Metode Analisis

Dalam membuat sebuah sistem rekomendasi buku pada penelitian ini, dibutuhkan beberapa tahapan dan proses yang perlu dilakukan, seperti analisis masalah, pengumpulan data, preprocessing, perhitungan kemiripan, dan evaluasi. Gambaran mengenai tahapan penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Langkah-langkah penelitian

3.1.1 Analisis Masalah

Penelitian ini mengangkat tema mengenai sistem rekomendasi dengan menggunakan metode *content based filtering* dan menggunakan algoritma TF-IDF serta mengukur kemiripan tiap item dengan menggunakan metode c*osine similarity*. Pada proses tersebut, sistem rekomendasi diharapkan dapat memberikan referensi baru kepada pengguna mengenai berbagai macam buku yang memiliki kemiripan dengan buku yang diminati oleh para pengguna. Selain itu, permasalahan lainnya adalah bagaimana sistem dapat menampilkan rekomendasi buku dan meminimalkan terjadinya kesalahan pada hasil rekomendasi.

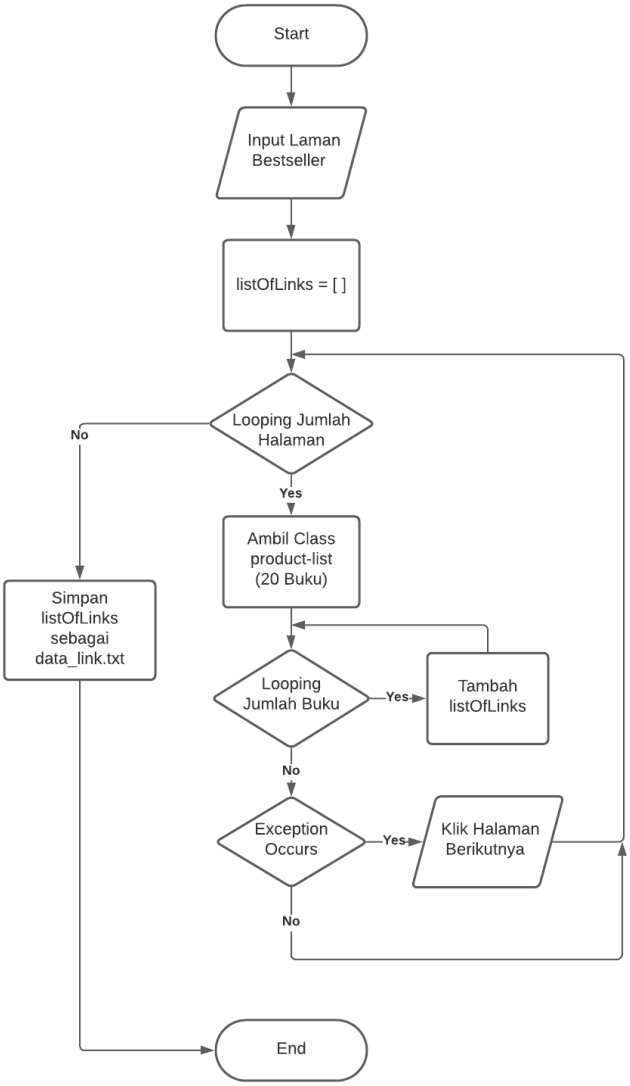
3.1.2 Pengumpulan Data

Data yang digunakan pada sistem rekomendasi buku pada penelitian kali ini merupakan buku yang didapatkan dari situs Gramedia. Penggunaan data dari situs Gramedia dikarenakan pada situs tersebut memiliki cukup banyak data buku berbahasa Indonesia yang tersedia dan Gramedia merupakan salah satu situs buku terbesar di Indonesia. Jumlah buku yang digunakan sebanyak 5143 buku dan genre buku yang diambil berasal dari beragam kategori mulai dari novel, *self improvement*, sejarah, fantasi, dan lain sebagainya.

Teknik Pengumpulan Data

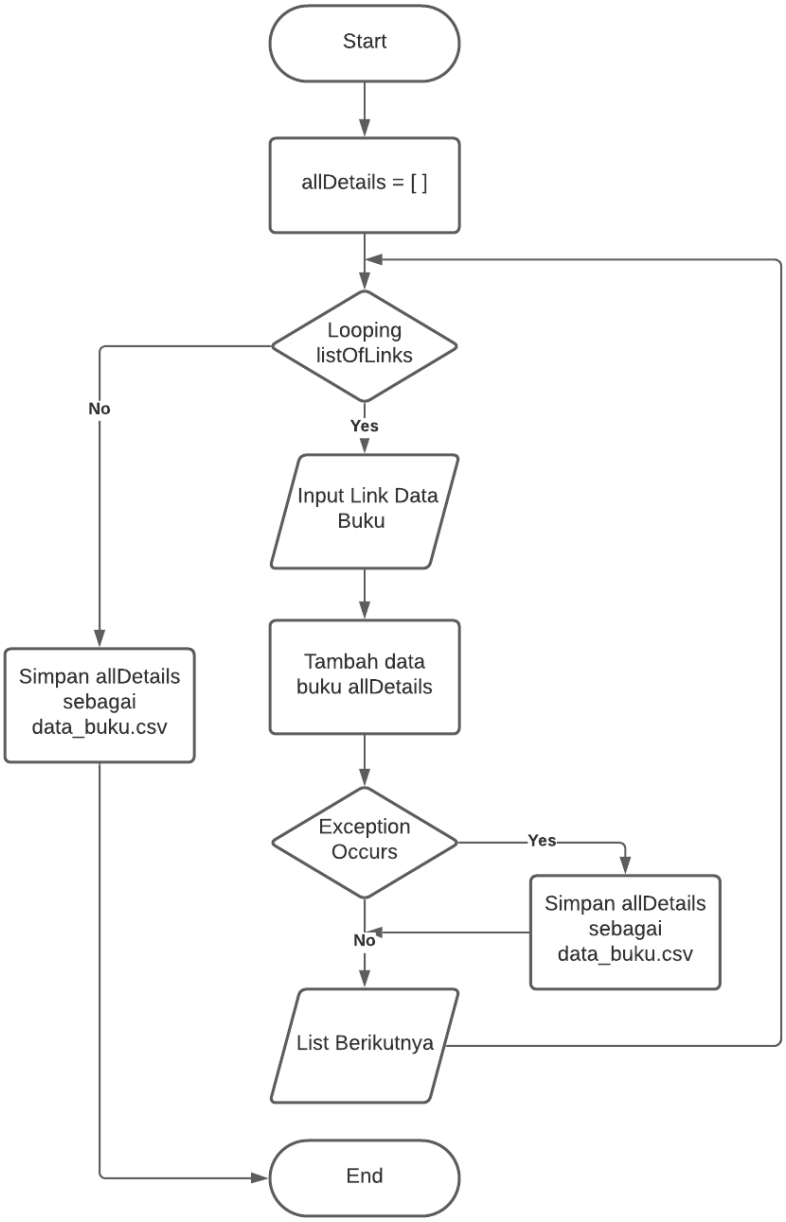
Proses pengambilan data ini dilakukan dengan menggunakan *web scraping*. *Web scraping* adalah proses pengambilan sebuah informasi dokumen yang bersifat semi struktur dari internet yang berupa halaman-halaman web yang berbentuk HTML atau XHTML. Dalam proses *scraping*, dibutuhkan beberapa indikator penting yang dibutuhkan pada sistem rekomendasi seperti situs buku, judul buku, penulis, genre, deskripsi dan *cover* dari buku tersebut. Data buku diambil sebanyak 5143 buku dari berbagai macam genre yang merupakan buku-buku populer dari situs gramedia. Pengambilan data dilakukan dengan menggunakan *jupyter notebook* dan memanfaatkan *library* Selenium sebagai *tools* untuk melakukan *scraping* pada situs gramedia. Selenium merupakan salah satu *tools auto testing* yang dapat digunakan untuk mengotomatisasi tes aplikasi web yang dilakukan pada *browser*. Hasil data yang diambil berupa judul, penulis, genre, deskripsi, *cover*, dan situs dari buku tersebut kemudian disimpan ke dalam CSV, sehingga data tersebut dapat diolah dan dimanfaatkan untuk membuat sistem rekomendasi dengan menggunakan bahasa pemrograman Python.

Scraping untuk pengambilan data buku melalui dua tahapan yang perlu dilakukan. Tahapan pertama adalah dengan mengarahkan sistem *auto testing* menuju ke halaman *bestseller* dari situs gramedia yang kemudian akan menampilkan 20 buku dari tiap halamannya. Pada tahap pertama yang perlu dilakukan adalah dengan mengambil *link* dari 20 buku yang ditampilkan pada setiap halaman di kategori *bestseller*. Tahap ini dilakukan dengan membuat sebuah *automation testing* yang berjalan secara otomatis sehingga dapat mengambil *link* dari setiap buku pada situs Gramedia. Setelah mendapatkan *link* dari setiap buku tersebut, data *link* tersebut kemudian disimpan dalam bentuk list dan disimpan dengan ekstensi txt. Berikut adalah gambaran proses tahapan pertama mengenai *scraping* yang dilakukan untuk mengambil *link* setiap buku dari situs gramedia Gambar 3.2.



Gambar 3.2 Flowchart *scraping* pengambilan link buku

Setelah berhasil mengambilan situs dari setiap buku tersebut, tahap kedua adalah dengan membuat *automation* yang berfungsi untuk mengambil bagian-bagian penting dari buku tersebut seperti judul, genre, penulis, deskripsi, dan *cover* buku. Cara yang digunakan untuk mengambil bagian penting dari HTML ini adalah dengan memanfaatkan bagian XPATH yang merupakan bahasa *query*, untuk memilih bagian dari file XML dokumen pada website tersebut. Dengan menyalin XPATH, sistem dapat mengambil bagian data buku yang ingin diambil menggunakan *library* Selenium yang telah dibuat sebelumnya. Setelah semua bagian data buku berhasil diambil, lalu data tersebut disimpan ke dalam sebuah dokumen dengan ekstensi CSV. Gambar 3.3 adalah gambaran proses tahapan kedua mengenai *scraping* yang dilakukan untuk mengambil bagian penting dari setiap buku Gramedia.

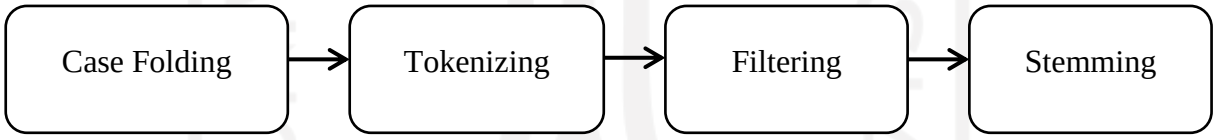


Gambar 3.3 Flowchart *scraping* pengambilan data buku

Data buku yang telah berhasil dikumpulkan melalui *scraping* dari situs Gramedia kemudian diolah dengan menggunakan bahasa pemrograman Python seperti melakukan *preprocessing*. Hal ini dilakukan untuk menghilangkan data-data yang tidak lengkap sehingga dapat mempengaruhi proses pembuatan sistem rekomendasi.

3.1.3 Preprocessing

Preprocessing merupakan langkah awal yang dilakukan untuk mengubah data mentah menjadi informasi yang lebih bersih dan bisa diolah untuk proses selanjutnya. Preprocessing pada penelitian ini dilakukan dengan memanfaatkan *Natural Language Processing* (NLP) yang merupakan cabang dari kecerdasan buatan (*artificial intelligence*) yang berkaitan dengan kemampuan untuk memberi komputer pememahaman terhadap teks dan kata-kata yang diucapkan dengan cara yang sama seperti yang dapat dilakukan manusia (Education, 2020). Pada proses ini, peneliti juga menggunakan *Natural Language ToolKit* (NLTK) yang merupakan *tools* pada *Natural Language Processing* (NLP) dengan menggunakan bahasa pemrograman Python. *Natural Language Toolkit* ini sangat mendukung proses pengolahan bahasa natural seperti *classification, tokenization, stemming, tagging, parsing,* dll (Yulio, 2019). Untuk mengolah kata pada tahapan *preprocessing* data, peneliti mengunakan korpus yang merupakan kumpulan dari teks secara tertulis atau lisan yang tersimpan untuk kebutuhan suatu penyelidikan dan penelitian, salah satunya pada *Natural Language Processing* (NLP). Tahapan yang dilakukan untuk preprocessing data melalui beberapa tahapan seperti *case folding, tokenizing, filtering* (*stopword removal*), dan *stemming*. Tahapan yang dilakukan untuk *preprocessing* data dapat dilihat pada Gambar 3.4.



Gambar 3.4 Tahapan *preprocessing*

Case folding

Tahap *case folding* dilakukan untuk mengubah kata-kata atau kalimat di dalam dokumen data buku, baik itu judul, genre, penulis, dan deskripsi buku menjadi huruf kecil. Tujuan dilakukannya proses ini adalah untuk memberikan kesamaan tiap kata pada dokumen data buku. Contoh penerapan tahap ini dapat dilihat pada Tabel 3.1.

Tabel 3.1 Proses *case folding*

Sebelum	Sesudah
Penelusuran sebuah kompleks perumahan tua terbengkalai di daerah Jakarta Timur malam itu awalnya berjalan menyenangkan.	penelusuran sebuah kompleks perumahan tua terbengkalai di daerah jakarta timur malam itu awalnya berjalan menyenangkan.
*Tentang persahabatan\n *Tentang cinta\n *tentang perpisahan\n *tentang hujan\n *tentang melupakan	*tentang persahabatan\n *tentang cinta\n *tentang perpisahan\n *tentang hujan\n *tentang melupakan
Sherlock Holmes adalah tokoh fiksi karya Sir Arthur Conan Doyle yang sangat legendaris sejak 1887 hingga sekarang.	sherlock holmes adalah tokoh fiksi karya sir arthur conan doyle yang sangat legendaris sejak 1887 hingga sekarang.
Pada masa lalu, manusia menaklukkan dunia berkat kemampuan uniknya untuk percaya pada mitos-mitos kolektif tentang dewa, uang, kesetaraan dan kebebasan—seperti dijelaskan dalam buku Sapiens.	pada masa lalu, manusia menaklukkan dunia berkat kemampuan uniknya untuk percaya pada mitos-mitos kolektif tentang dewa, uang, kesetaraan dan kebebasan—seperti dijelaskan dalam buku sapiens.
“Sebelum membangun lima perusahaan, menjadi multimiliarder, dan menulis buku-buku bestseller, saya tak punya uang, pengangguran, dan kecanduan narkoba. Selama 25 tahun saya mendengarkan keluarga dan teman-teman saya	“sebelum membangun lima perusahaan, menjadi multimiliarder, dan menulis buku-buku bestseller, saya tak punya uang, pengangguran, dan kecanduan narkoba. selama 25 tahun saya mendengarkan keluarga dan teman-teman saya

Tokenizing

Proses *tokenizing* dilakukan untuk melakukan pemotongan terhadap *string* input berdasarkan pada tiap kata. Pada proses ini, data dari tiap buku juga akan dilakukan tahap *cleaning* untuk membersihkan kalimat dari *symbol*, tanda baca, angka, *backslice*, karakter non-ASCII, url, *whitespace leading & trailing*, mengubah *multiple whitespace* menjadi *single whitespace*, *single char*. Dengan *cleaning data*, dokumen yang akan di *tokenize* menjadi lebih kecil ukurannya sehingga proses dapat dilakukan dengan lebih cepat. Contoh dari dokumen data buku yang telah melalui proses *tokenizing* dapat dilihat pada Tabel 3.2.

Tabel 3.2 Proses *tokenizing*

Sebelum	Sesudah
penelusuran sebuah kompleks perumahan tua terbengkalai di daerah jakarta timur malam itu awalnya berjalan menyenangkan.	[penelusuran, sebuah, kompleks, perumahan, tua, terbengkalai, di, daerah, jakarta, timur, malam, itu, awalnya, berjalan, menyenangkan, ...]
*tentang persahabatan\n *tentang cinta\n *tentang perpisahan\n *tentang hujan\n *tentang melupakan	[tentang, persahabatan, tentang, cinta, tentang, perpisahan, tentang, hujan, tentang, melupakan]
sherlock holmes adalah tokoh fiksi karya sir arthur conan doyle yang sangat legendaris sejak 1887 hingga sekarang.	[sherlock, holmes, adalah, tokoh, fiksi, karya, sir, arthur, conan, doyle, yang, sangat, legendaris, sejak, hingga, sekarang, ...]
pada masa lalu, manusia menaklukkan dunia berkat kemampuan uniknya untuk percaya pada mitos-mitos kolektif tentang dewa, uang, kesetaraan dan kebebasan—seperti dijelaskan dalam buku sapiens.	[pada, masa, lalu, manusia, menaklukkan, dunia, berkat, kemampuan, uniknya, untuk, percaya, pada, mitos, mitos, kolektif, tentang, dewa, uang, kesetaraan, dan, kebebasan, seperti, dijelaskan, dalam, buku, sapiens, ...]
“sebelum membangun lima perusahaan, menjadi multimiliarder, dan menulis buku-buku bestseller, saya tak punya uang, pengangguran, dan kecanduan narkoba. selama 25 tahun saya mendengarkan keluarga dan teman-teman saya	[sebelum, membangun, lima, perusahaan, menjadi, multimiliarder, dan, menulis, buku, buku, bestseller, saya, tak, punya, uang, pengangguran, dan, kecanduan, narkoba, selama, tahun, saya, mendengarkan, keluarga, dan, teman, teman, …]

Filtering

Filtering (*stopword removal*) adalah proses yang dilakukan untuk mengambil kata-kata penting yang terdapat di dalam dokumen data buku. Pada proses ini, dilakukan juga penghapusan kata-kata yang tidak deskriptif, termasuk kata penghubung, kata depan, dan kata pengganti. Misalnya seperti “di”, “yang”, “dan”, “ke” dan “dari”. *Filtering* bertujuan untuk mengurangi volume kata pada dokumen, sehingga proses dapat dilakukan dengan lebih tepat dengan menggunakan kata-kata yang penting di dalam dokumen. Contoh penerapan tahap ini dapat dilihat pada Tabel 3.3.

Tabel 3.3 Proses *filtering*

Sebelum	Sesudah
[penelusuran, sebuah, kompleks, perumahan, tua, terbengkalai, di, daerah, jakarta, timur, malam, itu, awalnya, berjalan, menyenangkan, ...]	[penelusuran, kompleks, perumahan, tua, terbengkalai, daerah, jakarta, timur, malam, berjalan, menyenangkan, ...]
[tentang, persahabatan, tentang, cinta, tentang, perpisahan, tentang, hujan, tentang, melupakan]	[persahabatan, cinta, perpisahan, hujan, melupakan]
[sherlock, holmes, adalah, tokoh, fiksi, karya, sir, arthur, conan, doyle, yang, sangat, legendaris, sejak, hingga, sekarang, ...]	[sherlock, holmes, tokoh, fiksi, karya, sir, arthur, conan, doyle, legendaris, ...]
[pada, masa, lalu, manusia, menaklukkan, dunia, berkat, kemampuan, uniknya, untuk, percaya, pada, mitos, mitos, kolektif, tentang, dewa, uang, kesetaraan, dan, kebebasan, seperti, dijelaskan, dalam, buku, sapiens, ...]	[manusia, menaklukkan, dunia, berkat, kemampuan, uniknya, percaya, mitos, mitos, kolektif, dewa, uang, kesetaraan, kebebasan, seperti, buku, sapiens, ...]
[sebelum, membangun, lima, perusahaan, menjadi, multimiliarder, dan, menulis, buku, buku, bestseller, saya, tak, punya, uang, pengangguran, dan, kecanduan, narkoba, selama, tahun, saya, mendengarkan, keluarga, dan, teman, teman, saya, …]	[membangun, perusahaan, multimiliarder, menulis, buku, buku, bestseller, uang, pengangguran, kecanduan, narkoba, mendengarkan, keluarga, teman, teman, …]

Stemming

Proses *Stemming* dilakukan untuk menemukan kata dasar dari sebuah kata yang telah mengalami *filtering*. Stemming akan menghilangkan imbuhan kata pada awal (*prefix*), akhir (*suffix*), sisipan (*infix*), dan kombinasi antara awalan dan akhiran (*confix*). Proses s*temming* dilakukan dengan menggunakan *library* dari Sastrawi yang dapat mengubah kata pada dokumen data buku yang memiliki imbuhan menjadi kata dalam bentuk dasar. Contoh penerapan *stemming* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Proses *stemming*

Sebelum	Sesudah
[penelusuran, kompleks, perumahan, tua, terbengkalai, daerah, jakarta, timur, malam, berjalan, menyenangkan, ...]	telusur kompleks rumah tua bengkalai daerah jakarta timur malam jalan senang …
[persahabatan, cinta, perpisahan, hujan, melupakan]	sahabat cinta pisah hujan lupa
[sherlock, holmes, tokoh, fiksi, karya, sir, arthur, conan, doyle, legendaris, ...]	sherlock holmes tokoh fiksi karya sir arthur conan doyle legendaris …
[manusia, menaklukkan, dunia, berkat, kemampuan, uniknya, percaya, mitos, mitos, kolektif, dewa, uang, kesetaraan, kebebasan, seperti, buku, sapiens, ...]	manusia takluk dunia berkat mampu unik percaya mitos mitos kolektif dewa uang tara bebas buku sapiens …
[membangun, perusahaan, multimiliarder, menulis, buku, buku, bestseller, uang, pengangguran, kecanduan, narkoba, mendengarkan, keluarga, teman teman, …]	bangun usaha multimiliarder tulis buku buku bestseller uang anggur candu narkoba dengar keluarga teman teman …

3.1.4 Perhitungan Kemiripan

Proses yang dilakukan untuk menghitung skor kemiripan tiap buku melalui dua tahapan. Pertama adalah dengan menggunakan algoritma TF-IDF untuk memberikan bobot pada setiap *term* pada deskripsi. Kedua adalah dengan melakukan perhitungan kemiripan tiap buku dengan menggunakan *cosine similarity*.

Pembobotan Kata TF IDF Vectorizer

Pembobotan kata dilakukan dengan menggunakan algoritma TF-IDF. Pembobotan suatu kata merupakan ukuran dari pentingnya kata tersebut dalam merepresentasikan data buku. Pada pembobotan TF-IDF, bobot akan semakin besar jika frekuensi kemunculan kata semakin tinggi, tetapi bobot akan berkurang jika kata tersebut semakin sering muncul pada buku lainnya. Misalnya terdapat tiga deskripsi yang sudah melewati tahap *preprocessing* seperti berikut:

a. Kalimat 1: “teknologi startup baru lokal”
b. Kalimat 2: “bisnis startup skala internasional”
c. Kalimat 3: “startup internasional era teknologi informasi”

Dari ketiga kalimat di atas, maka dilakukanlah perhitungan dengan menggunakan Persamaan (2.3). Melalui perhitungan dengan rumus tersebut, maka pembobotan dapat dilakukan seperti yang dapat dilihat pada Tabel 3.5.

Tabel 3.5 Pembobotan TF-IDF

No	Kata	Doc 1	Doc 2	Doc 3	df	Idf	Tf.idf		
							Doc 1	Doc 2	Doc 3
1	teknologi	1	0	1	2	Log(3/2)=0.1760	0.1760	0	0.1760
2	startup	1	1	1	3	Log(3/3)=0	0	0	0
3	baru	1	0	0	1	Log(3/1)=0.4771	0.4771	0	0
4	lokal	1	0	0	1	Log(3/1)=0.4771	0.4771	0	0
5	bisnis	0	1	0	1	Log(3/1)=0.4771	0	0.4771	0
6	skala	0	1	0	1	Log(3/1)=0.4771	0	0.4771	0
7	internasional	0	1	1	2	Log(3/2)=0.1760	0	0.1760	0.1760
8	era	0	0	1	1	Log(3/1)=0.4771	0	0	0.4771
9	informasi	0	0	1	1	Log(3/1)=0.4771	0	0	0.4771

Penerapan Algoritma Cosine Similarity

Perhitungan *cosine similarity* dilakukan untuk mengukur tingkat kemiripan antara dua vektor atau dua dokumen pada ruang vektor. Dengan pengukuran ini, maka sistem dapat menentukan kemiripan antara satu data buku dengan data buku yang lainnya. Perbandingan antara dokumen yang telah melalui *preprocessing* tidak hanya besaran dari setiap hitungan kata (TF-IDF), tetapi juga sudut antar dokumen-dokumen. Perhitungan *cosine similarity* dapat dilakukan dengan menyelesaikan Persamaan (2.4).

Sebagai contoh disini terdapat dua dokumen, yaitu A dan B yang akan digunakan untuk melakukan pengujian terhadap sistem. Dokumen tersebut sebelumnya juga telah melalui proses *preprocessing*. Nilai dari term A dan term B dapat diperoleh dari nilai kata yang unik contohnya kata “startup” pada *term* A bernilai 2 karena kata “startup” pada *term* A muncul sebanyak 2 kali, begitu juga *term* B. Kemudian, semua nilai yang diperoleh dari kata yang unik akan dimasukan dalam rumus *cosine similarity*.

Tabel 3.6 Perhitungan *cosine similarity*

No	Term	T(A)	T(B)
1	startup	2	2
2	bisnis	1	1
3	digital	1	0
4	era	1	0
5	teknologi	1	1
6	informasi	1	0

Vektor A dan vektor B pada Tabel 3.6 mewakili setiap *term* "A" dan *term* "B" untuk melihat berapa banyak nilai yang diperoleh dalam setiap kata yang unik dalam dokumen, selanjutnya akan dihitung dengan menggunakan *cosine similarity*.

a. Vektor A = (2,1,1,1,1,1)
b. Vektor B = (2,1,0,0,1,0)

Misalkan untuk mencari hasil perhitungan antara *term* A dan *term* B pada Tabel 3.6 dilakukan perhitungan kemiripan. Dengan menggunakan Persamaan pada (2.4) maka dihasilkan nilai sebagai berikut:

$$similarity\ (A,B) = \frac{(2\times 2)+(1\times 1)+(1\times 0)+(1\times 0)+(1\times 1)+(1\times 0)}{\sqrt{2^2+1^2+1^2+1^2+1^2+1^2}\times\sqrt{2^2+1^2+0^2+0^2+1^2+0^2}}$$

$$similarity\ (A,B) = \frac{4+1+0+0+1+0}{\sqrt{4+1+1+1+1+1}\times\sqrt{4+1+0+0+1+0}}$$

$$similarity\ (A,B) = \frac{6}{\sqrt{9}\times\sqrt{6}}$$

$$similarity\ (A,B) = \frac{6}{7.35}$$

$$similarity\ (A,B) = \frac{6}{7.35} = 0.82$$

Setelah berhasil melakukan perhitungan menggunakan *cosine similarity*, maka diperoleh hasil dari ukuran tingkat kemiripan antara *term* A dengan *term* B pada ruang vektor tersebut yaitu sebesar 0.82. Hasil tersebut menunjukkan bahwa *term* A dan *term* B memiliki tingkat kemiripan yang tinggi karena mendekati 1.

3.1.5 Evaluasi

Evaluasi merupakan tahapan yang dilakukan untuk mengukur kinerja dari sistem yang telah dibuat. Data yang telah diolah dan telah melalui tahapan dari *preprocessing* hingga pengukuran kemiripan menggunakan *cosine similarity* perlu dievaluasi dan diukur seberapa akurat sistem tersebut. Evaluasi dilakukan dengan menggunakan *precision* sebagai ukuran tingkat ketepatan antara informasi yang diminta oleh pengguna dan jawaban yang diberikan oleh sistem. Dengan *precision*, akurasi dari sistem rekomendasi berbasis konten yang telah dibuat dapat diukur dan dievaluasi dengan baik.

3.2 Implementasi GUI

Sistem rekomendasi yang telah berhasil dibuat selanjutnya akan diimplementasikan dalam bentuk *Graphical User Interface* (GUI) pada sebuah website. Pembuatan website dilakukan dengan menggunakan Flask sebagai *framework* untuk sistem tersebut dan di unggah pada sebuah *cloud platform* yaitu Heroku.

3.3 Komponen Perancangan Sistem

Rancangan sistem dilakukan untuk membuat sebuah sistem rekomendasi dan mengimplementasikannya pada website. Komponen dari perancangan sistem ini meliputi perangkat keras (*hardware)* dan perangkat lunak (*software*). Perangkat-perangkat yang digunakan pada penelitian ini adalah sebagai berikut:

a. Perangkat Keras
    1) Processor AMD Ryzen 3
    2) Random Access Memory (RAM) 8 GB
    3) Kapasistas SSD 250 GB dan Harddisk 1 TB
    4) Monitor 14 inch
    5) Mouse dan keyboard

b. Perangkat Lunak
    1) Sistem Operasi Windows 10
    2) Jupyter Notebook dan Google Colab
    3) Visual Studio
    4) Microsoft Excel
    5) Web Browser: Chrome dan Microsoft Edge
    6) Bahasa Pemrograman Python

BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengumpulan Data

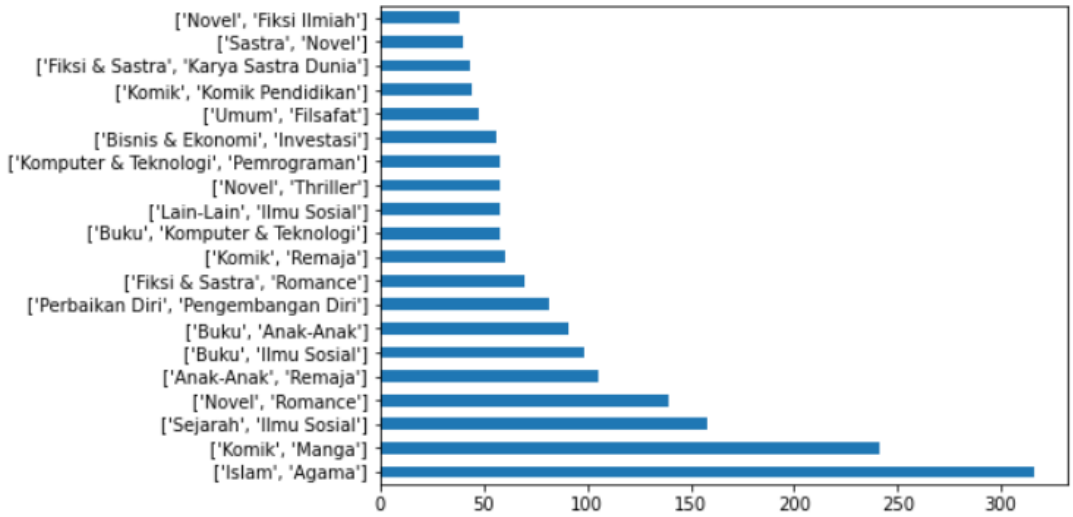
Data yang digunakan untuk penelitian ini berasal dari situs Gramedia. Pengumpulan data dilakukan dengan menggunakan *web scraping*. Proses *scraping* dapat dilakukan dengan membuat sebuah *automation system* yang dapat melakukan pengambilan data secara otomatis. Data yang diambil pada penelitian ini berjumlah 5143 data buku dan variabel penting yang dibutuhkan untuk membuat sistem rekomendasi pada penelitian ini adalah judul buku, penulis, genre, deskripsi, *cover* buku, dan situs dari setiap buku. Pengumpulan data dilakukan dengan menggunakan *web scraping*. Data yang akan digunakan untuk pembuatan sistem rekomendasi dapat dilihat pada Gambar 4.1.

	title	author	genre	description	image	site
0	Wingit	Sara Wijayanto	['Horor', 'Novel']	Penelusuran sebuah kompleks perumahan tua terb...	https://cdn.gramedia.com/uploads/items/9786230...	https://www.gramedia.com/products/wingit
1	Hujan	Tere Liye	['Fiksi & Sastra', 'Novel Grafis']	*Tentang persahabatan\n *Tentang cinta\n *tent...	https://ebooks.gramedia.com/ebook-covers/30371...	https://www.gramedia.com/products/hujan-tere-liye
2	How To Think Like Sherlock Holmes	PETER HOLLINS	['Pengembangan Diri', 'How To Think Like Sherl...	Sherlock Holmes adalah tokoh fiksi karya Sir A...	https://cdn.gramedia.com/uploads/items/9786230...	https://www.gramedia.com/products/how-to-think...
3	Homo Deus	Yuval Noah Harari	['Sejarah', 'Homo Deus']	Pada masa lalu, manusia menaklukkan dunia berk...	https://cdn.gramedia.com/uploads/items/homodeu...	https://www.gramedia.com/products/homo-deus
4	Be Obsessed or Be Average	Grant Cardone	['Perbaikan Diri', 'Pengembangan Diri']	“Sebelum membangun lima perusahaan, menjadi mu...	https://cdn.gramedia.com/uploads/items/9786020...	https://www.gramedia.com/products/be-obsessed-...
...	...	...	...	...	...	...
5138	Mendirikan Startup Yang Diburu Anggel Investor...	Hasnul Arifin	['Bisnis & Ekonomi', 'UKM & Wirausaha']	Siapa bilang orang-orang muda tidak bisa sukse...	https://cdn.gramedia.com/uploads/items/9789799...	https://www.gramedia.com/products/mendirikan-s...
5139	Startup Business Model	Hendry E Ramadhan	['Bisnis & Ekonomi', 'UKM & Wirausaha']	SALAH SATU ILMU PENTING DALAM MENGELOLA BISNIS...	https://cdn.gramedia.com/uploads/items/9786021...	https://www.gramedia.com/products/startup-busi...
5140	Legal In Startup Business	Doni Wijayanto	['Bisnis & Ekonomi', 'UKM & Wirausaha']	Buku ini membahas aturan hukum yang berlaku di...	https://cdn.gramedia.com/uploads/items/9786026...	https://www.gramedia.com/products/legal-in-sta...
5141	Eric Ries: Startup Way	Eric Ries	['Komputer & Teknologi', 'Eric Ries: Startup W...	Entrepreneur and bestselling author of The Lea...	https://cdn.gramedia.com/uploads/items/81-UORy...	https://www.gramedia.com/products/eric-ries-st...
5142	The Lean Startup	Eric Ries	['Bisnis & Ekonomi', 'Manajemen']	Lean Startup bukanlah metode yang menjadikan s...	https://cdn.gramedia.com/uploads/items/9786022...	https://www.gramedia.com/products/lean-startup...

5143 rows × 6 columns

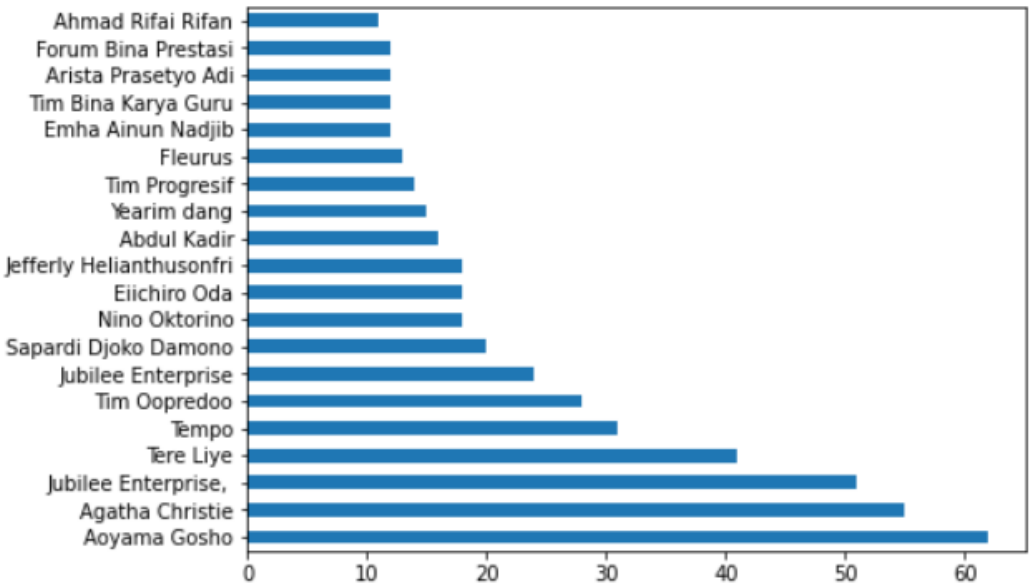
Gambar 4.1 Dataset buku

Data yang berhasil dikumpulkan dapat diketahui jumlah genre buku apa saja yang berhasil didapatkan dari hasil *scraping* di situs Gramedia. Dari 5143 data buku, diketahui 20 *genre* terbanyak dari seluruh data yang tersedia dapat dilihat pada Gambar 4.2.



Gambar 4.2 Grafik 20 genre terbanyak

Selain itu, dari data yang berhasil dikumpulkan juga dapat diketahui jumlah penulis buku terbanyak yang berhasil didapatkan. Dari 5143 data buku, 20 penulis terbanyak dapat dilihat pada Gambar 4.3.



Gambar 4.3 Grafik 20 penulis terbanyak

4.1.1 Web Scraping

Proses *scraping* yang dilakukan untuk pengambilan data dari situs Gramedia memiliki dua tahapan, yang pertama adalah dengan mengambil link buku pada tiap halaman dari situs gramedia. Setelah berhasil mengambil link dari setiap data buku yang tersedia, tahap kedua yaitu mengambil variabel-variabel buku yang dibutuhkan seperti judul, penulis, genre, deskripsi, dan *cover* dari setiap buku.

Mengambil Link Dari Situs Gramedia

Proses pengambilan link buku pada situs gramedia dilakukan dengan tujuan untuk memudahkan proses pengambilan data pada tahap selanjutnya, yaitu pengambilan variabel data buku yang dibutuhkan untuk sistem rekomendasi. Gambar 4.4 adalah kode program automasi untuk *scraping* yang dibuat pada tahapan pertama.

```
1.  import selenium
2.  import time
3.  from selenium import webdriver
4.  from webdriver_manager.chrome import ChromeDriverManager
5.
6.  driver = webdriver.Chrome(ChromeDriverManager().install())
7.  driver.get('https://www.gramedia.com/categories/buku?based_on=best-seller')
8.  listOfLinks=[]
9.
10. for i in range(3):
11.     time.sleep(3)
12.     productInfoList = driver.find_elements_by_class_name('product-list')
13.     productInfoList[0].text
14.     for perProduct in productInfoList:
15.         classPerProduct = perProduct.find_elements_by_tag_name('div')[0]
16.         linkProduct = classPerProduct.find_element_by_tag_name('a')
17.         listOfLinks.append(linkProduct.get_property('href'))
18.     try:
19.         driver.find_elements_by_class_name('ion-ios-arrow-forward')[1].click()
20.     except:
21.         continue
22.
23. with open("data_link.txt", "w") as f:
24.     for s in listOfLinks:
25.         f.write(str(s) +"\n")
```

Gambar 4.4 Kode *scraping* pengambilan link buku

Mengambil tiap elemen buku

Setelah proses pengambilan link data buku berhasil dilakukan, langkah selanjutnya adalah pengambilan variabel-variabel penting pada setiap buku yang dapat digunakan pada sistem rekomendasi buku. Gambar 4.5 adalah kode program automasi untuk pengambilan data dari setiap buku.

```
import pandas as pd
from tqdm import tqdm

allDetails = []
for i in tqdm(listOfLinks):
    try:
        driver.get(i)
        time.sleep(3)
        title = driver.find_element_by_xpath('//*[@id="title-ribbon"]').text
        author = driver.find_element_by_class_name("title-author").text
        genre1 = driver.find_element_by_xpath('//*[@id="content"]/gm-product/gm-product-view/div/gm-breadcrumb/ul/li[3]/a').text
        genre2 = driver.find_element_by_xpath('//*[@id="content"]/gm-product/gm-product-view/div/gm-breadcrumb/ul/li[4]/a').text
        genre = [genre1, genre2]
        description = driver.find_element_by_xpath('//*[@id="mat-tab-content-0-0"]/div/pre').text
        image = driver.find_element_by_class_name("image").get_attribute("src")
        tempJ = {'title':title, 'author':author, 'genre':genre,
                 'description':description, 'image':image, 'site':i
                }
        allDetails.append(tempJ)
    except:
        data_buku = pd.DataFrame(allDetails)
        data_buku.to_csv('data_buku.csv', sep='\t', encoding='utf-8')
        continue

data_buku = pd.DataFrame(allDetails)
data_buku.to_csv('data_buku.csv', sep='\t', encoding='utf-8')
```

Gambar 4.5 Kode *scraping* pengambilan data buku

4.2 Preprocessing

Tahapan *preprocessing* dilakukan untuk membersihkan kalimat deskripsi yang tidak perlu seperti, simbol, angka, kata-kata yang kurang memiliki arti dan lain sebagainya. *Preprocessing* dilakukan dengan beberapa tahapan yang akan dijelaskan secara berurutan.

4.2.1 Case Folding

Case Folding dilakukan untuk membuat seluruh teks menjadi huruf kecil, dengan begitu seluruh teks akan memiliki format penulisan yang sama dengan menggunakan huruf kecil. Kode program untuk melakukan proses ini dapat dilihat pada Gambar 4.6.

```
books['title_list'] = books['title'].str.lower()
books['author_list'] = books['author'].str.lower()
books['genre_list'] = books['genre'].str.lower()
books['description_list'] = books['description'].str.lower()
# author_list dimunculkan 3 kali untuk menambah bobot
books['author_list'] = books['author_list'].apply(lambda x: x.replace(" ", ""))
books['author_list'] = books['author_list'].apply(lambda x: [x,x,x])
```

Gambar 4.6 Kode proses *case folding*

Setelah proses *case folding* berhasil dilakukan, langkah yang dilakukan selanjutnya adalah membersihkan atau menghilangkan data genre dari yang bertuliskan judul dari buku tersebut. Kode program untuk tahap ini dapat dilihat pada Gambar 4.7.

```
# ubah string ke list
books['genre_list'] = books['genre'].apply(lambda x:x.replace("'", "").replace("[", "").replace("]", ""))
books['genre_list'] = books.genre_list.apply(lambda x: x.split(', '))

# list genre ke dua
gen = []
for genre in books['genre_list']:
    gen.append(genre[1])

# list title
tit = []
for title in books['title']:
    tit.append(title)

# hapus genre kedua jika sama dengan title
x = 0
for i in books['genre_list']:
    if i[1] == tit[x]:
        books['genre_list'][x] = books['genre_list'][x][0]
    x = x + 1

# set buku yang memiliki 1 genre menjadi list
y = 0
for i in books['genre_list']:
    if isinstance(i, str):
        books['genre_list'][y] = i.split("delimiter")
    y = y + 1

# set genre menjadi lowercase
z = 0
for i in books['genre_list']:
    for j in i:
        j = j.lower()
        n = 0
    for k in books['genre_list'][z]:
        books['genre_list'][z][n] = books['genre_list'][z][n].lower()
        n = n + 1
    z = z + 1
```

Gambar 4.7 Kode proses *cleaning* genre

Penggabungan Data

Data yang telah melalui tahapan *case folding*, kemudian digabung dalam satu kolom bernama “soup”. Kolom ini merupakan gabungan dari judul, penulis, genre, dan deskripsi buku yang telah melalui tahapan *preprocessing* sebelumnya. Kolom ini akan digunakan untuk pembobotan nilai menggunakan TF-IDF dan dilakukan proses perhitungan kemiripan dengan menggunakan *Cosine Similarity*. Gambar 4.8 adalah kode program untuk penggabungan data-data tersebut.

```
# mengubah list menjadi string
books['genre_list'] = [' '.join(map(str, l)) for l in books['genre_list']]
books['author_list'] = [' '.join(map(str, l)) for l in books['author_list']]

# genre_list dimunculkan 3 kali untuk menambah bobot
books['genre_list'] =  books['genre_list'].apply(lambda x: [x,x,x])
books['genre_list'] = [' '.join(map(str, l)) for l in books['genre_list']]

# menggabungkan title dan description ke dalam satu kolom untuk preprocessing
books['soup'] = books['description_list']  + " " + books['title_list']
```

Gambar 4.8 Kode penggabungan data

4.2.2 Tokenizing

Tokenizing dilakukan untuk melakukan pemotongan terhadap *string* atau teks menjadi potongan-potongan token sebelum dianalisis lebih lanjut. Pada tahapan ini juga dilakukan proses *cleaning* untuk menghilangkan simbol, url, angka, atau hal-hal yang kurang penting lainnya. Gambar 4.9 adalah kode program untuk melakukan tahapan *tokenizing*.

```
import string
import re
import nltk

# import word_tokenize & FreqDist dari NLTK
nltk.download('punkt')
from nltk.tokenize import word_tokenize

# menghapus tab, new line, backslice, non ASCII dan incomplete URL
books['soup'] = books['soup'].apply(lambda x: x.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\',"").replace("http://", " ").replace("https://", " "))
books['soup'] = books['soup'].apply(lambda x: x.encode('ascii', 'replace').decode('ascii'))
books['soup'] = books['soup'].apply(lambda x: ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", x).split()))

# menghapus angka
books['soup'] = books['soup'].apply(lambda x: re.sub(r"\d+", "", x))
# replace simbol with space
books['soup'] = books['soup'].apply(lambda x: x.translate({ord(c): " " for c in "!@#$%^&*()[]{};:,./<>?\|`~-=_+—"}))
# menghapus tanda baca
books['soup'] = books['soup'].apply(lambda x: x.translate(str.maketrans("","",string.punctuation)))
# menghapus whitespace leading & trailing
books['soup'] = books['soup'].apply(lambda x: x.strip())

# mengganti multiple whitespace menjadi single whitespace
books['soup'] = books['soup'].apply(lambda x: re.sub('\s+',' ', x))
# menghapus single char
books['soup'] = books['soup'].apply(lambda x: re.sub(r"\b[a-zA-Z]\b", "", x))

# NLTK word tokenize
books['soup'] = books['soup'].apply(lambda x: word_tokenize(x))
```

Gambar 4.9 Kode proses *tokenizing*

4.2.3 Filtering

Filtering (*stopword removal*) dilakukan untuk membersihkan kata-kata kurang penting dan menyisakan kata-kata penting pada dokumen data buku. Sehingga data yang diolah untuk sistem rekomendasi tidak memiliki banyak kata yang mengganggu atau kata yang kurang penting. Gambar 4.10 adalah kode program untuk melakukan proses *filtering*.

```
# mendownload daftar kata yang ada
nltk.download('stopwords')
from nltk.corpus import stopwords

# memeriksa daftar kata di stopwords
indo = stopwords.words('indonesian')
eng = stopwords.words('english')

# list stopwords bahasa indonesia
list_stopwords = indo

# menambah stopwords bahasa inggris
list_stopwords.extend(eng)

# hapus stopword pada list token
def stopwords_removal(words):
    return [word for word in words if word not in list_stopwords]

books['soup'] = books['soup'].apply(stopwords_removal)
```

Gambar 4.10 Kode proses *filtering*

4.2.4 Stemming

Stemming dilakukan untuk mengubah suatu kata dalam dokumen menjadi kata dasar. Hal ini dilakukan dengan menghilangkan semua imbuhan (*affixes*) yang terdiri dari awalan (*prefixes*), sisipan (infixes), akhiran (*suffixes*) dan kombinasi dari awalan dan akhiran (*confixes*). Gambar 4.11 adalah kode program untuk melakukan proses *stemming*.

```
# import Sastrawi package
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter

# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# stemmed
def stemmed_wrapper(term):
    return stemmer.stem(term)

term_dict = {}

```

```
for document in books['soup']:
    for term in document:
        if term not in term_dict:
            term_dict[term] = ' '

print(len(term_dict))
print("------------------------")

for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
    print(term,":" ,term_dict[term])

print(term_dict)
print("------------------------")

# apply stemmed term to dataframe
def get_stemmed_term(document):
    return [term_dict[term] for term in document]

books['soup'] = books['soup'].swifter.apply(get_stemmed_term)

# mengubah list pada kolom soup menjadi string
books['soup'] = books['soup'].agg(lambda x: ' '.join(map(str, x)))
# menggabungkan soup sebelumnya dengan author dan genre
books['soup'] = books['soup']+ " " +books[author_list']+ " " +books['genre_list']
```

Gambar 4.11 Kode proses *stemming*

4.3 Perhitungan Kemiripan

Perhitungan kemiripan merupakan tahap akhir yang dilakukan untuk menghitung similaritas dari tiap buku. Dari hasil perhitungan tersebut, maka sistem dapat memberikan rekomendasi berdasarkan kemiripan antara satu buku dengan buku yang lainnya.

4.3.1 Pembobotan Nilai TF-IDF

Pada proses pembobotan TF-IDF, sistem akan membentuk sebuah objek dari class TfidfVectorizer untuk ditampung ke dalam variabel tfidf. Kemudian, memanggil method *fit_transform* dari objek *tfidf* untuk membangun matriks TF-IDF pada data buku dengan kolom “soup”. Gambar 4.12 adalah kode program untuk pembobotan menggunakan TF-IDF.

```
# libraries for Recommendation System
from sklearn.feature_extraction.text import TfidfVectorizer

# TF-IDF Vectorizer
tfidf = TfidfVectorizer(stop_words=list_stopwords)

# membangun matriks TF-IDF
tfidf_matrix = tfidf.fit_transform(books['soup'])

# output the shape of tfidf_matrix
tfidf_matrix.shape
```

Gambar 4.12 Kode pembobotan TF-IDF

4.3.2 Cosine similarity

Cosine similarity merupakan metode yang dilakukan untuk mengukur kemiripan antara dua atau lebih dokumen teks. *linear_kernel()* digunakan untuk menghitung vektor matriks cosine similarity pada *tfidf_matrix*. Gambar 4.13 adalah kode program untuk perhitungan menggunakan *cosine similarity*.

```
# import linear_kernel
from sklearn.metrics.pairwise import linear_kernel

# menghitung matriks cosine similarity
cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)

# reset index dataframe
books = books.reset_index()
titles = books['title']

# membuat map dari index dan judul buku
indices = pd.Series(books.index, index=books['title']).drop_duplicates()
```

Gambar 4.13 Kode perhitungan *cosine similarity*

4.4 Evaluasi

Selanjutnya, sistem dapat menampilkan rekomendasi berdasarkan kemiripan buku melalui kode pada Gambar 4.14. Sistem dapat menampilkan rekomendasi buku dengan cara memasukkan judul buku. Jika judul yang dimasukkan tersedia di dalam *database*, maka sistem akan menampilkan 5 rekomendasi buku yang memiliki kemiripan dengan judul buku yang dimasukkan seperti terlihat pada Gambar 4.15. Setelah berhasil menampilkan hasil rekomendasi, maka sistem dapat di evaluasi dengan menggunakan *precision*.

```
# mengambil judul buku sebagai input dan output buku yang paling mirip
def rec_tfidf(title, cosine_sim=cosine_sim):
    recommendation = pd.DataFrame(columns = ['Book Idx', 'Title', 'Score'])
    count = 0

    idx = indices[title]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:6]
    book_indices = [i[0] for i in sim_scores]

    for i in book_indices:
        recommendation.at[count, 'Book Idx'] = book_indices[count]
        recommendation.at[count, 'Title'] = titles.iloc[book_indices[count]]
        recommendation.at[count, 'Score'] = sim_scores[count][1]
        count += 1
    return recommendation
```

Gambar 4.14 Kode untuk menampilkan hasil rekomendasi

```
[7] rec_tfidf("Arah langkah")
```

	Book Idx	Title	Score
0	248	Catatan Juang	0.433946
1	104	11.11	0.389615
2	403	Tapak Jejak	0.358445
3	23	Konspirasi Alam Semesta	0.351534
4	3260	Kelana	0.290504

```
[8] rec_tfidf("Jack Ma & Alibaba")
```

	Book Idx	Title	Score
0	1549	ALIBABA Kerajaan yang Dibangun oleh Jack Ma	0.389429
1	4078	Rich Like Jack Ma : Kalau Jack Ma Bisa Kaya, M...	0.380975
2	3261	Jack Ma: Sisi-Sisi Tak Terduga Sang Godfather ...	0.372276
3	1639	The Alibaba Way	0.350689
4	1003	Series On Disruption: #Mo-Sc	0.258189

```
[9] rec_tfidf("Be Obsessed or Be Average")
```

	Book Idx	Title	Score
0	2959	GET SMART! Berpikir dan Bertindak Seperti Oran...	0.209643
1	3011	Petunjuk Menikmati Hidup & Pekerjaan	0.197974
2	2954	The One Thing--Kekuatan Fokus Untuk Mendorong ...	0.164671
3	1061	VALUE YOUR LIFE - Lepaskan Beban Tak Kasatmata...	0.158555
4	2987	Resolve Conflicts in Your Life	0.151601

Gambar 4.15 Hasil sistem rekomendasi buku

4.4.1 Rekomendasi Berdasarkan Kata Kunci

Sistem rekomendasi yang telah dibuat sebelumnya hanya dapat memberikan rekomendasi untuk judul buku yang tersedia pada *database*. Sistem tidak dapat memberikan rekomendasi apabila buku tidak tersedia pada database. Untuk menambah fitur rekomendasi, pengguna dapat mencari rekomendasi buku berdasarkan kata kunci (*keywords*) yang dimasukkan. Gambar 4.16 adalah kode untuk pencarian rekomendasi buku berdasarkan kata kunci.

```
1. def get_recommendations(judul, jumlah):
2.     # Creates Series
3.     judul = pd.Series(judul)
4.
5.     # Cleaning
6.     judul = judul.apply(lambda x: x.replace('\\t'," ").replace('\\n'," ").replace('
   \\u'," ").replace('\\',"").replace("http://", " ").replace("https://", " "))
7.     judul = judul.apply(lambda x: x.encode('ascii', 'replace').decode('ascii'))
8.     judul = judul.apply(lambda x: ' '.join(re.sub("([@#][A-Za-z0-
   9]+)|(\w+:\/\/\S+)"," ", x).split()))
```

```
9.      judul = judul.apply(lambda x: re.sub(r"\d+", "", x))
10.     judul = judul.apply(lambda x: x.translate({ord(c): " " for c in "!@#$%^&*()[]{}
    ;:,./<>?\|`~-=_+—"}))
11.     judul = judul.apply(lambda x: x.strip())
12.     judul = judul.apply(lambda x: re.sub('\s+',' ', x))
13.     judul = judul.apply(lambda x: re.sub(r"\b[a-zA-Z]\b", "", x))
14.
15.     # Tokenizing
16.     judul = judul.apply(lambda x: word_tokenize(x))
17.
18.     # Filtering. Implementasi kode pada Gambar 4.10 dengan variabel judul
19.     # Stemming. Implementasi kode pada Gambar 4.11 dengan variabel judul
20.
21.     # Ambil kata kunci
22.     judul = judul.to_string()
23.     judul = judul[6:-1]
24.     judul = re.sub(r'[^\w\s]', '', judul)
25.
26.     # Membaca data
27.     books = pd.read_csv("/content/drive/MyDrive/data_soup.csv", sep='\t')
28.     gnr = "['keywords', {judul}]"
29.     data = [{'Unnamed: 0': 5143, 'title': judul, 'author': judul, 'genre': gnr, 'de
    scription': judul, 'image': judul, 'site': judul, 'title_list': judul, 'author_list
    ': judul, 'genre_list': judul, 'description_list': judul, 'soup': judul}]
30.
31.     # Menghapus duplikat data berdasarkan judul
32.     books.drop_duplicates(subset ="title", inplace = True)
33.
34.     # Menggabungkan database buku dengan kata kunci yang dimasukkan
35.     new = pd.DataFrame(data)
36.     frames = [books, new]
37.     result = pd.concat(frames)
38.     books = result.copy()
39.
40.     # TF-IDF Vectorizer
41.     tfidf = TfidfVectorizer(stop_words=list_stopwords)
42.     tfidf_matrix = tfidf.fit_transform(books['soup'])
43.
44.     # Menghitung matrix cosine similarity
45.     cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
46.
47.     books = books.reset_index()
48.     titles = books['title']
49.     titles = titles.str.lower()
50.     authors = books['author']
51.
52.     # Implementasi kode pada Gambar 4.7 (tanpa set genre menjadi lowercase)
53.
54.     books['genre'] = [', '.join(map(str, l)) for l in books['genre']]
55.
56.     genres = books['genre']
57.     covers = books['image'].apply(lambda x: '<img src="{}" width="100px"/>'.format(
    x) if x else '')
58.     indices = pd.Series(books.index, index=titles)
59.
60.     # Memebentuk tampilan dataframe berdasarkan judul, penulis, genre dan cover
61.     jumlah = int(jumlah) + 1
62.     rec = pd.DataFrame(columns = ['Judul', 'Penulis', 'Genre', 'Cover'])
63.     count = 0
64.
65.     idx = indices[judul]
66.     sim_scores = list(enumerate(cosine_sim[idx]))
67.     sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
68.     sim_scores = sim_scores[1:jumlah]
69.     book_indices = [i[0] for i in sim_scores]
```

```
70.
71.     for i in book_indices:
72.         rec.at[count+1, 'Judul'] = titles.iloc[book_indices[count]].title()
73.         rec.at[count+1, 'Penulis'] = authors.iloc[book_indices[count]].title()
74.         rec.at[count+1, 'Genre'] = genres.iloc[book_indices[count]].title()
75.         rec.at[count+1, 'Cover'] = covers.iloc[book_indices[count]]
76.         count += 1
77.
78.     return rec
```

Gambar 4.16 Kode untuk memberikan rekomendasi berdasarkan kata kunci

Setelah sistem rekomendasi berdasarkan pencarian kata kunci berhasil dibuat, maka sistem dapat menampilkan rekomendasi buku berdasarkan kata kunci yang dimasukkan oleh pengguna dan jumlah buku yang diinginkan seperti yang terlihat pada Gambar 4.17.

```
[8] get_recommendations("bahasa pemrograman python untuk pemula", 5)
```

	Judul	Penulis	Genre	Cover
1	Python Untuk Programmer Pemula	Jubilee Enterprise	Komputer & Teknologi, Pemrograman	<img src="https://cdn.gramedia.com/uploads/ite...
2	Belajar Singkat Pemrograman Python 3 +Cd	Budi Raharjo	Komputer & Teknologi, Pemrograman	<img src="https://cdn.gramedia.com/uploads/ite...
3	Semua Bisa Menjadi Programmer Python Basic	Ir. Yuniar Supardi	Pemrograman	<img src="https://ebooks.gramedia.com/ebook-co...
4	Semua Bisa Menjadi Programmer Python Case Study	Yuniar Supardi,	Komputer & Teknologi, Pemrograman	<img src="https://ebooks.gramedia.com/ebook-co...
5	Belajar Pemrograman Dan Hacking Menggunakan Py...	Wardana	Komputer & Teknologi, Pemrograman	<img src="https://cdn.gramedia.com/uploads/ite...

```
[10] get_recommendations("sejarah islam di indonesia", 5)
```

	Judul	Penulis	Genre	Cover
1	Islam Dalam Arus Sejarah Indonesia	Jajat Burhanudin	Sejarah, Ilmu Sosial	<img src="https://ebooks.gramedia.com/ebook-co...
2	Dinamika Sejarah Umat Islam Indonesia	Kuntowijoyo	Sejarah, Ilmu Sosial	<img src="https://cdn.gramedia.com/uploads/ite...
3	Bunga Rampai Sejarah Indonesia	Moehkardi	Sejarah, Ilmu Sosial	<img src="https://cdn.gramedia.com/uploads/ite...
4	Moderasi Islam Indonesia	Prof. Dr. Mujamil Qomar, M.Ag.	Islam, Agama	<img src="https://cdn.gramedia.com/uploads/ite...
5	Mengislamkan Indonesia	Carool Kersten	Sejarah, Ilmu Sosial	<img src="https://cdn.gramedia.com/uploads/ite...

```
[12] get_recommendations("mendirikan bisnis startup", 5)
```

	Judul	Penulis	Genre	Cover
1	Startup Lessons	Hendry E. Ramdhan	Bisnis & Ekonomi, Ukm & Wirausaha	<img src="https://cdn.gramedia.com/uploads/ite...
2	The Lean Startup	Eric Ries	Bisnis & Ekonomi	<img src="https://cdn.gramedia.com/uploads/ite...
3	101 Laws Of Successful Startup : Bagaimana Mem...	Arisatya Yogaswara	Bisnis & Ekonomi, Kepemimpinan	<img src="https://cdn.gramedia.com/uploads/ite...
4	Legal In Startup Business	Doni Wijayanto	Bisnis & Ekonomi, Ukm & Wirausaha	<img src="https://cdn.gramedia.com/uploads/ite...
5	Startup Business Model	Hendry E Ramadhan	Bisnis & Ekonomi, Ukm & Wirausaha	<img src="https://cdn.gramedia.com/uploads/ite...

Gambar 4.17 Hasil rekomendasi buku berdasarkan kata kunci

4.5 Implementasi GUI

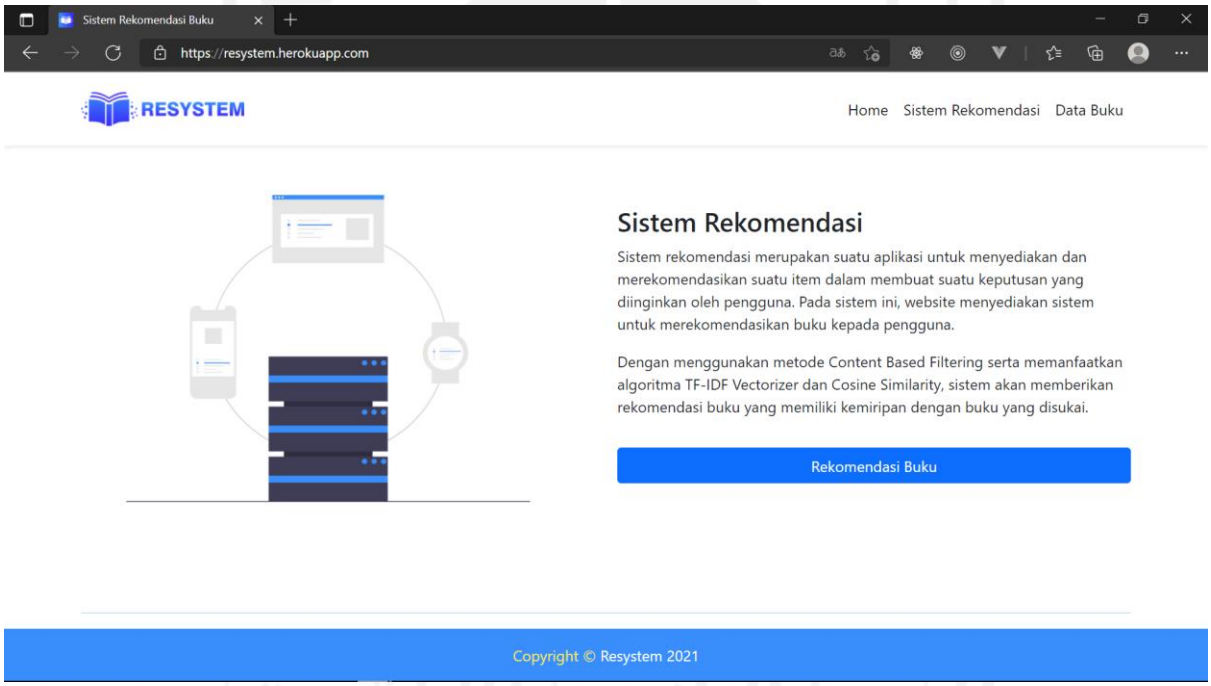
Sistem rekomendasi buku yang telah berhasil dibuat kemudian akan diimplementasikan dalam bentuk *Graphical User Interface* (GUI) pada sebuah website. Data yang digunakan pada sistem ini berupa file CSV yang berasal dari hasil pengolahan data yang telah dilakukan sebelumnya. Sehingga, kode sistem pada website hanya mengimplementasikan TF-IDF dan *Cosine Similarity* yang telah dibuat sebelumnya.

4.5.1 Perancangan Antarmuka Sistem

Sistem rekomendasi pada penelitian ini dibuat dalam bentuk website dengan menggunakan *framework* Flask dan diunggah pada *cloud platform* Heroku. Sehingga sistem rekomendasi ini juga dapat diakses oleh banyak orang untuk memberikan referensi baru terkait dengan buku yang diminati oleh pembaca buku sebelumnya.

Tampilan Halaman Pertama

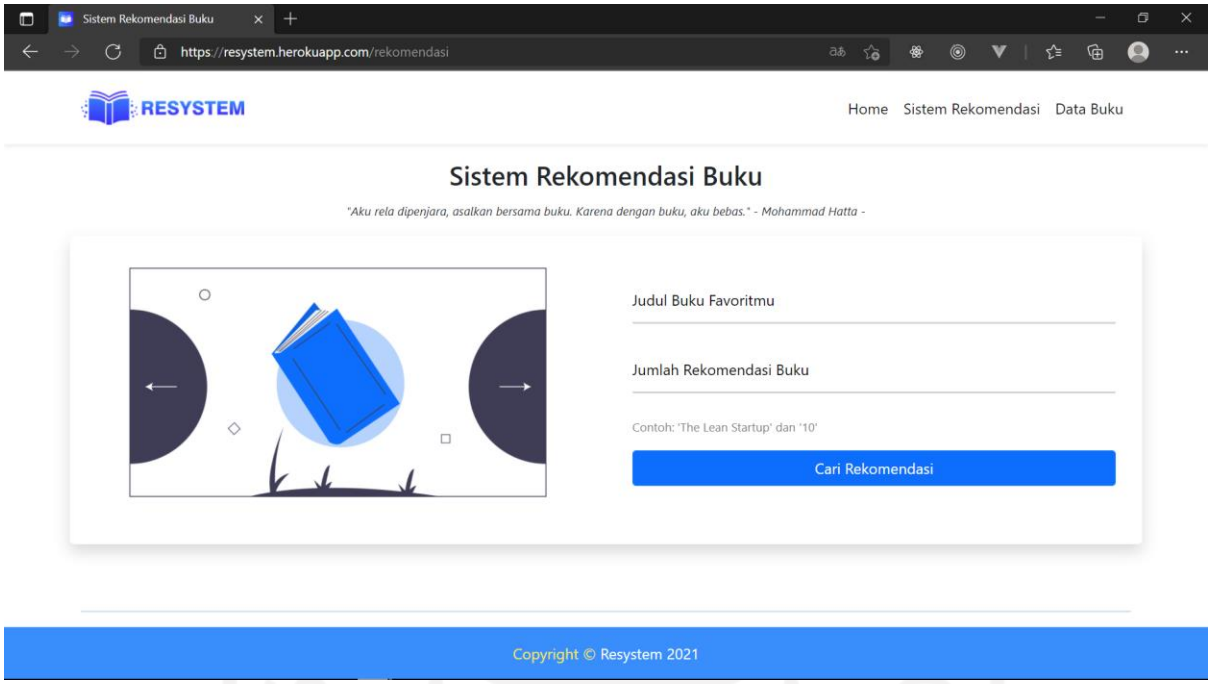
Pada tampilan awal sistem rekomendasi buku ini merupakan tampilan yang menjelaskan mengenai metode yang digunakan pada sistem rekomendasi seperti yang terlihat pada Gambar 4.18.



Gambar 4.18 Halaman pertama sistem rekomendasi

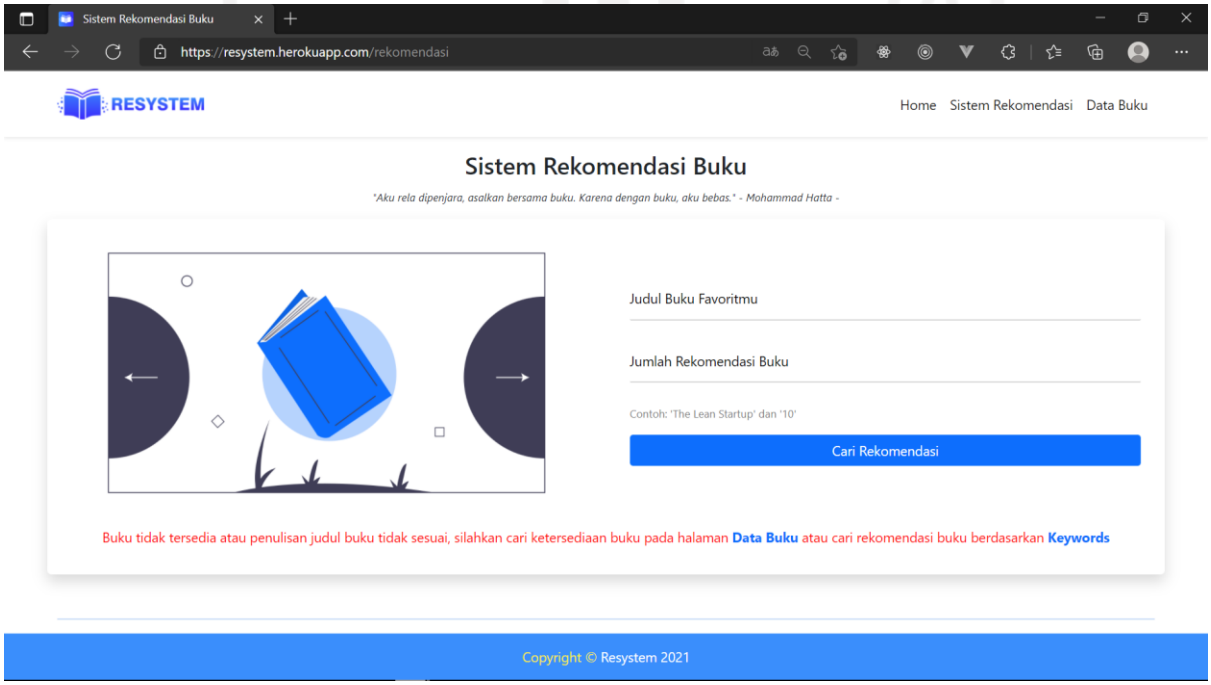
Tampilan Halaman Kedua

Halaman kedua merupakan halaman utama dari situs sistem rekomendasi buku. Dimana pada bagian ini, pengguna dapat memasukan judul buku yang diminati oleh pengguna dan dapat menentukan jumlah buku yang ingin di tampilkan sistem rekomendasi. Setelah *user* menekan tombol “Cari Rekomendasi”, maka sistem akan menampilkan rekomendasi berdasarkan tingkat kemiripan dari setiap buku, seperti yang terlihat pada Gambar 4.19.



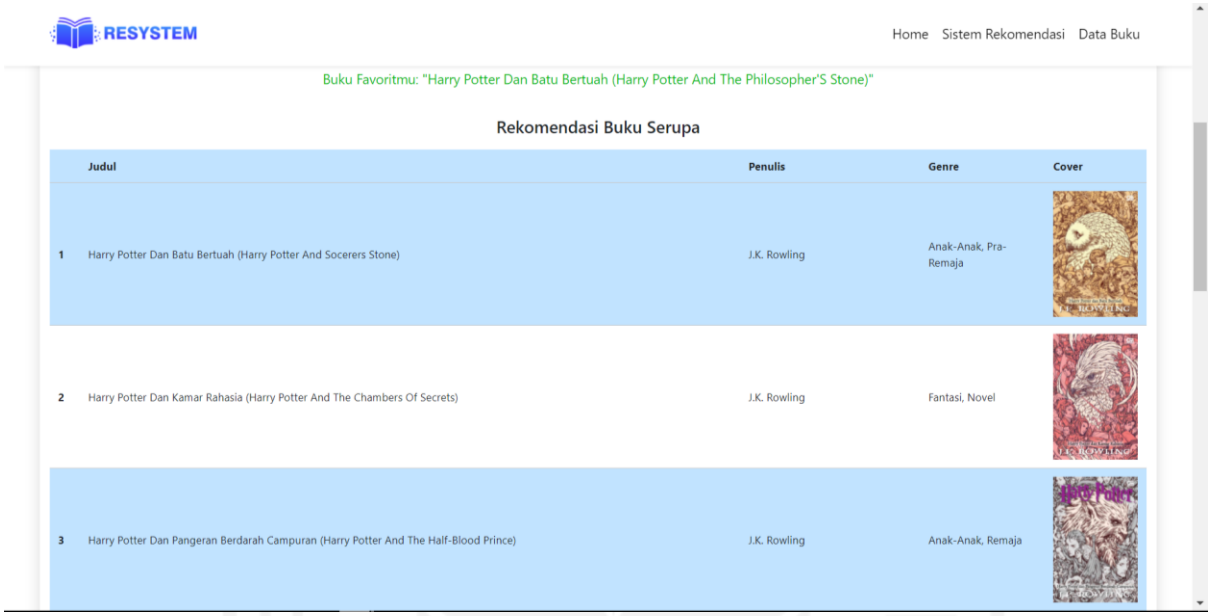
Gambar 4.19 Halaman kedua sistem rekomendasi

Apabila buku yang dimasukkan untuk sistem rekomendasi tidak tersedia di dalam *database*, maka situs akan memberikan peringatan seperti pada Gambar 4.20.



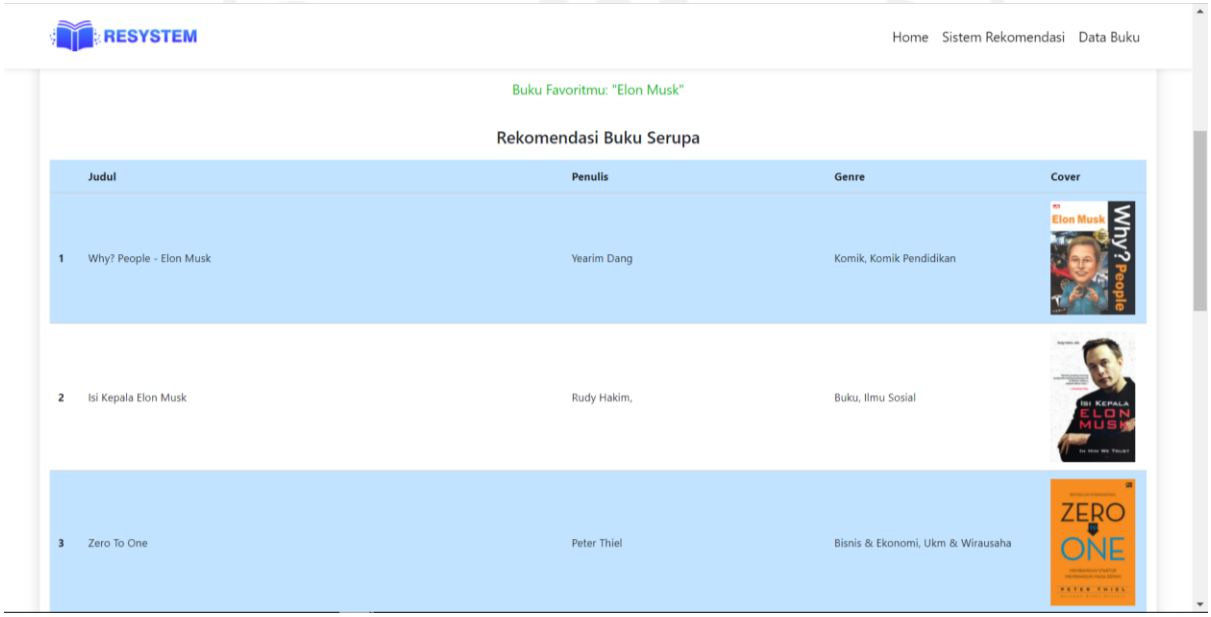
Gambar 4.20 Sistem memberikan peringatan

Apabila buku tersedia di dalam *database* maka sistem akan memberikan rekomendasi berdasarkan tingkat kemiripan yang telah dihitung menggunakan *cosine similarity*. Seperti yang terlihat pada Gambar 4.21.



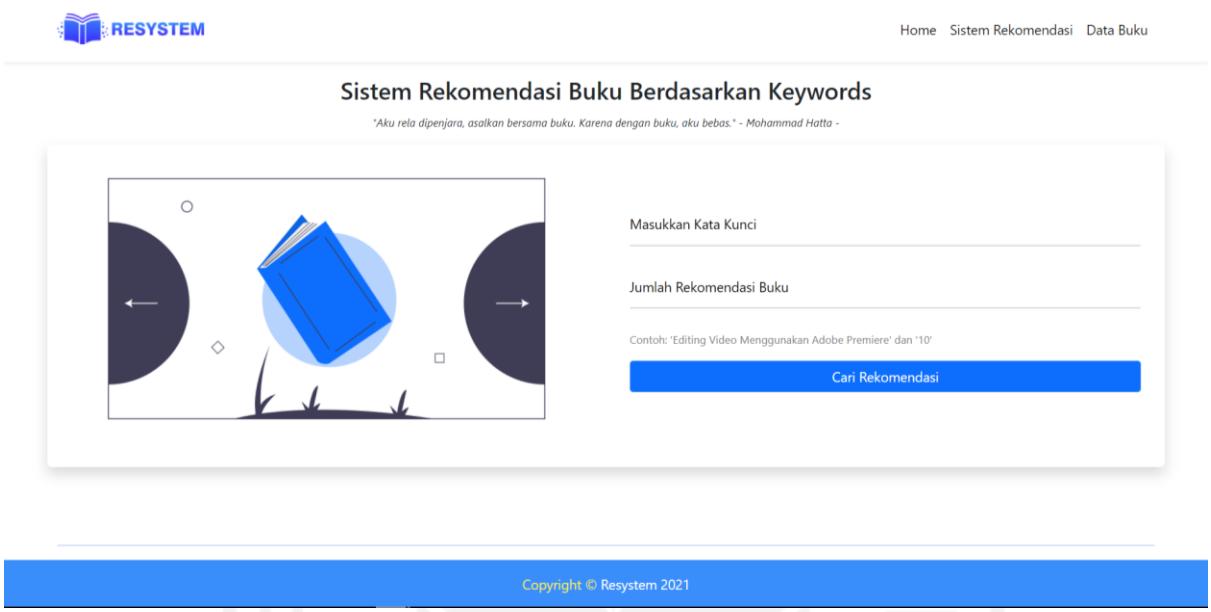
Gambar 4.21 Sistem berhasil memberikan rekomendasi

Percobaan dilakukan dengan menggunakan data buku yang lain maka akan dihasilkan rekomendasi seperti terlihat pada Gambar 4.22.



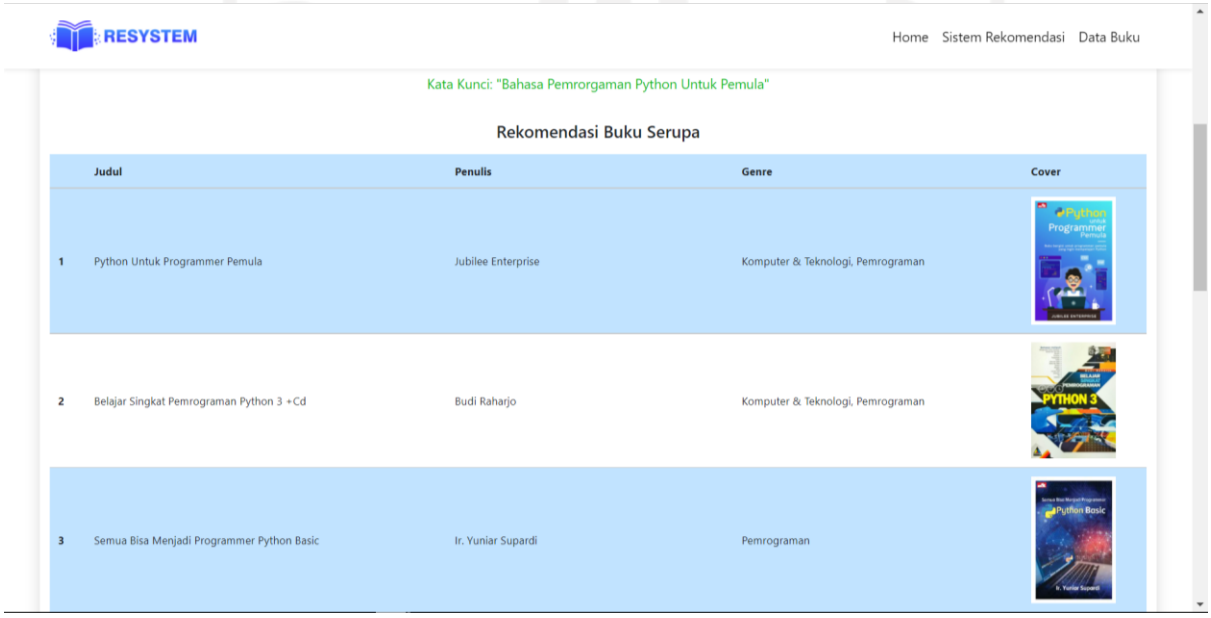
Gambar 4.22 Sistem berhasil memberikan rekomendasi

Jika pengguna ingin mencari buku berdasarkan kata kunci atau *keywords*, maka pengguna dapat klik tombol “Keywords” yang muncul seperti pada Gambar 4.20. Kemudian, tampilan halaman akan berpindah seperti yang terlihat pada Gambar 4.23.



Gambar 4.23 Halaman sistem rekomendasi berdasarkan kata kunci

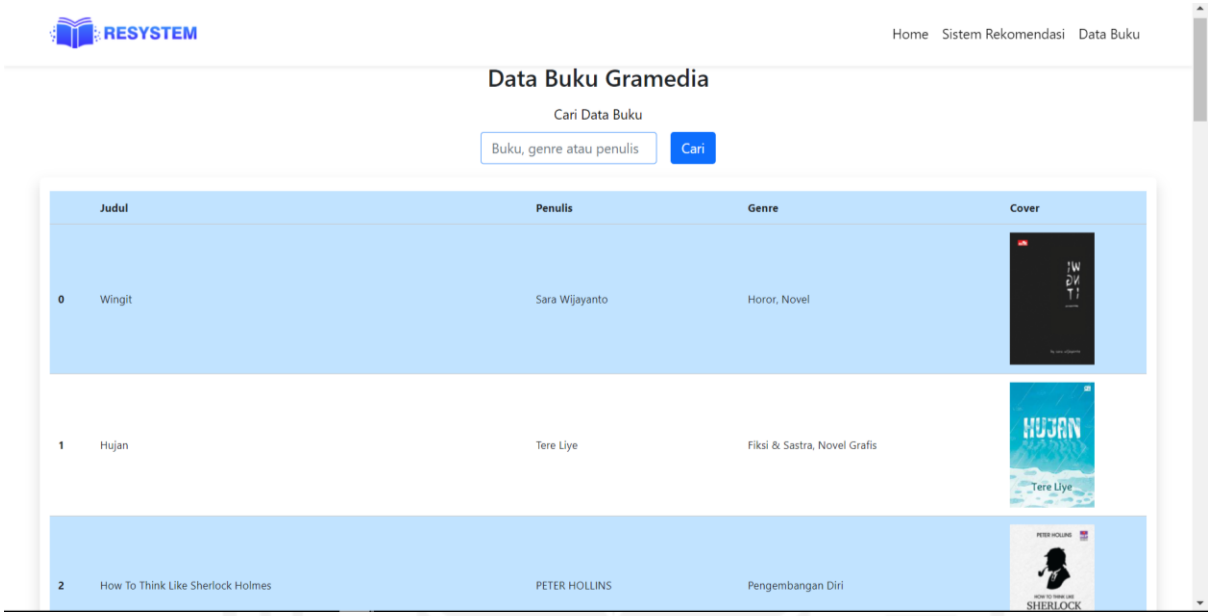
Selanjutnya, pengguna dapat menuliskan kata kunci dan jumlah rekomendasi buku yang diinginkan. Kemudian, sistem dapat memberikan rekomendasi seperti pada Gambar 4.24.



Gambar 4.24 Sistem berhasil memberikan rekomendasi berdasarkan kata kunci

Tampilan Halaman Ketiga

Halaman ketiga berfungsi untuk mengecek dan memberikan informasi mengenai data buku yang tersedia pada website, seperti yang terlihat pada Gambar 4.25.



Gambar 4.25 Sistem menampilkan data buku

4.6 Hasil Penelitian

Penelitian tugas akhir dari skripsi ini bertujuan untuk mengimplementasikan sistem rekomendasi buku berbasis *content based filtering* dengan menggunakan TF-IDF dan *cosine similarity*. Pembuatan sistem yang menggunakan data melalui situs Gramedia ini berhasil memberikan referensi dan rekomendasi baru kepada pengguna mengenai buku yang memiliki kemiripan dengan buku yang diminati oleh pengguna. Berikut adalah hasil uji dari pengembangan sistem rekomendasi yang telah dilakukan.

4.6.1 Hasil Uji Program

Pengujian dilakukan menggunakan *precision* dengan mengambil 20 sampel judul buku untuk mengukur seberapa baik akurasi dari rekomendasi buku yang diberikan oleh sistem. Pengujian dilakukan dengan menganalisis secara manual hasil dari 5 rekomendasi buku yang ditampilkan pertama. Semakin banyak buku yang memiliki kesesuaian dengan buku yang dicari pada sistem rekomendasi, maka skor *precision* semakin tinggi. Tabel 4.1 merupakan hasil dari pengujian yang dilakukan terhadap 20 buku berbeda.

Tabel 4.1 Hasil *precision* sistem rekomendasi

No	Data Uji (Judul Buku)	Jumlah Data Relevan	*Precision*
1	Think And Grow Rich	4/5	80%
2	Nebula	5/5	100%
3	Harry Potter dan Batu Bertuah (Harry Potter and The Philosopher's Stone)	5/5	100%
4	Elon Musk	5/5	100%
5	Homo Deus	3/5	60%
6	Jack Ma & Alibaba	5/5	100%
7	How To Think Like Sherlock Holmes	5/5	100%
8	Kaum Rebahan Beri Perubahan	4/5	80%
9	Soekarno Sang Guru Bangsa	5/5	100%
10	Detektif Conan 81	5/5	100%
11	Be Obsessed or Be Average	4/5	80%
12	Zero To One	2/5	40%
13	Helen dan Sukanta	3/5	60%
14	Arah Langkah	5/5	100%
15	Negeri 5 Menara	2/5	40%
16	Panduan Sukses Tes Cpns 2021	4/5	80%
17	Nasionalisme, Islamisme, Marxisme	5/5	100%
18	Atomic Habits: Perubahan Kecil yang Memberikan Hasil Luar Biasa	4/5	80%
19	Sultan Muhammad Al-Fatih	5/5	100%
20	The Lean Startup	5/5	100%
Rata-rata *precision*			85%

Hasil dari pengujian sistem rekomendasi berbasis konten dengan menggunakan algoritma TF-IDF dan *Cosine Similarity* menghasilkan rata-rata nilai *precision* sebesar 85%. Hasil tersebut menunjukkan kemampuan sistem dalam mencari atau memberikan rekomendasi berdasarkan kemiripan dari data yang dimiliki oleh tiap buku.

4.6.2 Pembahasan

Sistem rekomendasi yang berhasil dibuat dengan menggunakan beberapa variabel untuk menentukan hasil rekomendasi seperti penulis, judul, genre, dan deskripsi buku dapat mempengaruhi hasil rekomendasi yang ditampilkan. Terdapat beberapa buku yang memiliki rekomendasi cukup berbeda jika dibandingkan dengan buku favorit yang dipilih. Misalnya pada buku "Negeri 5 Menara" dari peringkat 5 besar memiliki kemiripan dengan buku "Ayo, Berlatih Silat!". Padahal, kedua buku tersebut tidak terlalu relevan. Namun, karena kedua buku tersebut ditulis oleh penulis yang sama dan pada bagian deskripsi buku terdapat beberapa kata yang sama dari kedua buku tersebut, maka kedua buku tersebut dinilai memiliki kemiripan. Variabel-variabel yang digunakan pada sistem rekomendasi seperti judul, penulis, genre, dan deskripsi akan mempengaruhi proses perhitungan kemiripan dan mempengaruhi akurasi yang dihasilkan oleh sistem rekomendasi tersebut.

Selain itu, pada buku "Zero To One" juga memiliki akurasi yang kecil, dimana hanya mendapatkan skor *precision* sebesar 40% atau hanya dapat memberikan 2 rekomendasi buku yang sesuai dari 5 rekomendasi awal yang diberikan. Hal ini terjadi karena pada saat proses pembuatan sistem rekomendasi, pembobotan terhadap genre dari buku ditambah sehingga dapat mempengaruhi hasil dari rekomendasi buku dengan judul "Zero To One". Apabila penambahan bobot pada genre tidak diberikan, maka akan terjadi pengurangan akurasi dari beberapa buku yang lain dan dapat mengurangi skor *precision* secara keseluruhan.

4.6.3 Kelebihan dan Kekurangan

Kelebihan

1. Sistem dapat memberikan jumlah rekomendasi buku sesuai dengan yang diinginkan pengguna.
2. Sistem dapat memberikan rekomendasi buku berdasarkan kata kunci (*keywords*) yang di masukkan pengguna.
3. Terdapat fitur pencarian data buku untuk mencari judul buku yang sesuai pada *database*.

Kekurangan

1. Data buku yang digunakan masih belum terlalu banyak dan perlu ditingkatkan.
2. Apabila judul buku yang dimasukkan tidak sesuai pada *database*, maka sistem tidak dapat memberikan rekomendasi.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari penelitian yang dilakukan dalam membuat sistem rekomendasi buku berbasis *content based filtering* ini, maka dapat diambil kesimpulan bahwa:

a. Sistem rekomendasi buku dengan menggunakan teknik pembobotan TF-IDF dan algoritma *cosine similarity* dapat memberikan rekomendasi kepada pengguna sesuai dengan kemiripan data buku yang tersedia pada *database*.
b. Sistem rekomendasi buku dengan menggunakan metode *content based filtering* pada penelitian ini memiliki tingkat akurasi *precision* sebesar 85%.
c. Sistem rekomendasi buku tersebut dapat diimplementasikan pada website dengan menggunakan *framework* Flask dan dapat berjalan dengan baik pada sebuah *cloud platform* Heroku.

5.2 Saran

Melalui penelitian yang telah dilakukan saat ini, terdapat beberapa saran yang dapat dilakukan untuk penelitian berikutnya, yaitu:

a. Data yang digunakan untuk sistem rekomendasi buku dapat lebih ditingkatkan dan diperbanyak jumlahnya.
b. Implementasi pada *web browser* dapat ditingkatkan dengan menambahkan beberapa fitur atau elemen tambahan seperti menyertakan deskripsi, penerbit, harga dan lain sebagainya untuk memudahkan pengguna dalam mencari informasi buku yang lebih detail.

DAFTAR PUSTAKA

Alkaff, M., Khatimi, H., & Eriady, A. (2020). Sistem Rekomendasi Buku Menggunakan Weighted Tree Similarity dan Content Based Filtering. *Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, 193-202.

Chen, S. (2018, Novmeber 21). *How TFIDF scoring in Content Based Recommender works*. Dipetik June 28, 2021, dari Medium: https://medium.com/@shengyuchen/how-tfidf-scoring-in-content-based-recommender-works-5791e36ee8da

Devega, E. (2017, October 10). *TEKNOLOGI Masyarakat Indonesia: Malas Baca Tapi Cerewet di Medsos*. Dipetik July 10, 2021, dari Kementrian Komunikasi dan Informatika: https://www.kominfo.go.id/content/detail/10862/teknologi-masyarakat-indonesia-malas-baca-tapi-cerewet-di-medsos/0/sorotan_media

dosenpendidikan. (2021, June 30). *Pengertian Buku*. Dipetik June 25, 2021, dari Dosen Pendidikan: https://www.dosenpendidikan.co.id/pengertian-buku/

Education, I. C. (2020, July 2). *What is Natural Language Processing?* Dipetik July 11, 2021, dari IBM: https://www.ibm.com/cloud/learn/natural-language-processing

G, G., M, S., C, F., & D, S. (2018). A Hybrid Approach using Collaborative filtering and Content based Filtering for Recommender System. *National Conference on Mathematical Techniques and its Applications (NCMTA 18)*, 1-7.

Kaddam, N., & Kumar, S. (2016). A review of Content and Collaborative filtering approaches on Movielens Data. *International Research Journal of Engineering and Technology (IRJET)*, 273-278.

Lops, P., Jannach, D., Musto, C., Bogers, T., & Koolen, M. (2019). Trends in content-based recommendation. *User Modeling and User-Adapted Interaction*, 239-249.

Oeyliawan, R. F., & Gunawan, D. (2017). Aplikasi Rekomendasi Buku pada Katalog Perpustakaan Universitas Multimedia Nusantara Menggunakan Vector Space Model. *ULTIMATICS*, 97-105.

Prabowo, D., & Ramdani, F. (2020). Penerapan Algoritma Apriori Untuk Rekomendasi Buku Pada Amikom Resource Center. *Information System Journal (INFOS)*, 8-12.

Qaiser, S., & Ali, R. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*, 25-29.

Rymmai, R. G., & JS, S. (2017). Book Recommendation Using Cosine Similarity. *International Journal of Advanced Research in Computer Science*, 276-281.

Samuel, R., Natan, R., Fitria, & Syafiqoh, U. (2018). Penerapan Cosine Similarity dan K-Nearest Neighbor (K-NN) pada Klasifikasi dan Pencarian Buku. *Journal of Big Data Analytic and Artificial Intelligence*, 9-14.

Saurkar, A. V., Pathare, K. G., & Gode, S. A. (2018). An Overview On Web Scraping Techniques And Tools. *International Journal on Future Revolution in Computer Science & Communication Engineering*, 363-367.

Shao, B., Li, X., & Bian, G. (2020). A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph. *Expert Systems with Applications*, 1-20.

Wang, D., Liang, Y., Xu, D., Feng, X., & Guan, R. (2018). A content-based recommender system for computer science publications. *Knowledge-Based Systems*, 1-9.

Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2016). Collaborative Filtering and Deep Learning Based Recommendation System For Cold Start Items. *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, 1-29.

Witanto, J. (2018). Minat Baca Yang Sangat Rendah. *Manajemen Kurikulum dan Pembelajaran*, 1-13.

Yulio, A. (2019, January 17). *Pengenalan dan Instalasi Python NLTK*. Dipetik July 4, 2021, dari DEVTRIK: https://devtrik.com/python/pengenalan-dan-instalasi-python-nltk/

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2018). Deep Learning based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys*, 1-35.

LAMPIRAN