PERANCANGAN SISTEM REKOMENDASI RAKIT PC OTOMATIS BERDASARKAN KEBUTUHAN GAMING ATAU MULTITASKING SESUAI BUDGET MENGGUNAKAN NATURAL LANGUAGE PROCESSING

Fuad Fakhruddin Nur, Mochamad Alfan Rosid, Rohman Dijaya, Nuril Lutvi Azizah
Informatika, Universitas Muhammadiyah Sidoarjo
Jl. Raya Gelam No. 250, Candi, Sidoarjo
alfanrosid@umsida.ac.id

ABSTRAK

Kebutuhan akan *Personal Computer* (PC) yang sesuai dengan spesifikasi pengguna semakin meningkat. Namun, proses pemilihan komponen sering kali menjadi tantangan kompleks bagi pengguna awam akibat minimnya pengetahuan teknis mengenai kompatibilitas dan variasi harga. Hal ini berisiko menimbulkan inefisiensi biaya serta ketidakcocokan perangkat keras. Oleh karena itu, penelitian ini bertujuan merancang sistem rekomendasi rakit PC otomatis yang memproses masukan bahasa natural dari pengguna. Metode yang digunakan mencakup *Natural Language Processing* (NLP) berbasis kamus kata kunci untuk mengekstraksi kebutuhan spesifik dan batasan anggaran dari teks input. Selanjutnya, algoritma *K-Nearest Neighbors* (KNN) diterapkan untuk mencari komponen paling relevan, yang dipadukan dengan *Constraint-Based Filtering* guna memastikan kompatibilitas teknis antar komponen dan kesesuaian harga. Sistem ini dikembangkan menggunakan arsitektur *frontend* React JS dan *backend* Flask. Hasil pengujian menunjukkan bahwa sistem berhasil memberikan rekomendasi spesifikasi PC yang optimal, kompatibel, dan sesuai preferensi pengguna dengan tingkat akurasi mencapai 70%.

Kata kunci : *Sistem Rekomendasi, Rakit PC, Natural Language Processing, K-Nearest Neighbors, React JS, Flask*

1. PENDAHULUAN

Perkembangan teknologi perangkat keras komputer yang pesat dalam beberapa tahun terakhir telah menciptakan berbagai pilihan komponen PC dengan spesifikasi dan harga yang beragam. Kondisi ini memicu permasalahan berupa kompleksitas dalam proses pemilihan komponen untuk merakit *Personal Computer* (PC), terutama bagi pengguna awam yang tidak memiliki pengetahuan teknis memadai. Pengguna sering kali menghadapi kesulitan dalam menentukan kombinasi komponen yang tepat untuk memenuhi kebutuhan spesifik mereka, baik untuk keperluan *gaming* maupun *multitasking*, dengan mempertimbangkan keterbatasan anggaran yang dimiliki [1].

Kesalahan dalam memilih perangkat keras ini dapat mengakibatkan pemborosan biaya, performa yang tidak optimal, atau bahkan ketidakcocokan komponen (*incompatibility*) yang mengharuskan pengguna mengeluarkan biaya tambahan.

Upaya untuk membantu pengguna dalam merakit PC dan mengembangkan sistem rekomendasi telah dieksplorasi pada beberapa penelitian terdahulu berfokus pada perancangan *User Experience* (UX) aplikasi merakit PC dengan pendekatan *User-Centered Design* (UCD) dan menghasilkan prototipe dengan kegunaan yang baik, namun belum mengimplementasikan mesin rekomendasi otomatis [1].

Di sisi lain, mengembangkan sistem rekomendasi film menggunakan algoritma *K-Nearest Neighbors* (KNN) yang berjalan baik, namun domain tersebut tidak memiliki kendala kompatibilitas perangkat keras yang rumit seperti pada komponen PC [2].

Hal ini menunjukkan masih perlunya pengembangan sebuah sistem komprehensif yang dapat menjembatani celah tersebut.

Berangkat dari permasalahan dan tinjauan tersebut, penelitian ini bertujuan untuk merancang sistem rekomendasi rakit PC otomatis yang dapat menganalisis kebutuhan spesifik serta memberikan rekomendasi komponen yang optimal. Sebagai rencana penyelesaian masalah, penelitian ini akan memanfaatkan teknologi *Natural Language Processing* (NLP) agar sistem dapat memahami kebutuhan dan batasan anggaran pengguna yang disampaikan secara langsung dalam bahasa natural [1][3].

Sistem ini akan mengevaluasi kompatibilitas, performa, dan batasan anggaran secara otomatis. Guna menunjang operasionalnya, keseluruhan sistem diimplementasikan menggunakan arsitektur *frontend* React JS dan *backend* Flask untuk memastikan performa aplikasi yang responsif dan andal [4][5].

2. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Berbagai penelitian telah dilakukan untuk mengeksplorasi penerapan kecerdasan buatan dalam sistem rekomendasi dan otomatisasi seleksi. mengembangkan sistem *screening* CV otomatis pada aplikasi SKILLQ, yang membuktikan bahwa integrasi *pipeline* ekstraksi teks, pra-pemrosesan, dan

klasifikasi *K-Nearest Neighbors* (KNN) efektif memangkas waktu seleksi secara signifikan [6].

Pendekatan serupa dalam pemrosesan teks terstruktur pada sistem penerimaan magang, di mana penggunaan TF-IDF dan KNN berhasil mengotomasi seleksi peserta secara terstruktur [7].

Selain itu, mengevaluasi model data *chatbot* menggunakan KNN dan mencapai akurasi tinggi pada skenario tertentu, menegaskan keandalan algoritma ini dalam menangani klasifikasi teks [8].

Dalam domain sistem rekomendasi, membangun sistem rekomendasi film berbasis *Content-Based Filtering* dan KNN yang menghasilkan presisi yang baik, menunjukkan efektivitas KNN dalam memetakan preferensi pengguna[2].

Terkait pengembangan arsitektur aplikasi, *User Experience* (UX) aplikasi perakitan PC dengan pendekatan *User-Centered Design* (UCD) yang menghasilkan tingkat usabilitas tinggi [1].

Dari sisi teknologi *backend*, membuktikan bahwa implementasi API menggunakan *framework* Flask menghasilkan sistem yang aman, mudah dipelihara, dan lulus pengujian fungsional [5][9][10].

Sementara itu, pada sisi *front-end*, menyoroti keunggulan React JS sebagai *Single Page Application* (SPA) yang meningkatkan kecepatan *rendering* dan efisiensi pengembangan antarmuka [4][11][12].

2.2. Analisis Gap

Meskipun penelitian-penelitian terdahulu telah memberikan kontribusi signifikan, terdapat celah penelitian (*research gap*) yang perlu diatasi. Penelitian berhasil merancang antarmuka aplikasi rakit PC yang baik, namun studi tersebut berhenti pada tahap purwarupa (*prototype*) dan belum mengimplementasikan mesin rekomendasi otomatis yang mampu memproses input pengguna secara dinamis [1].

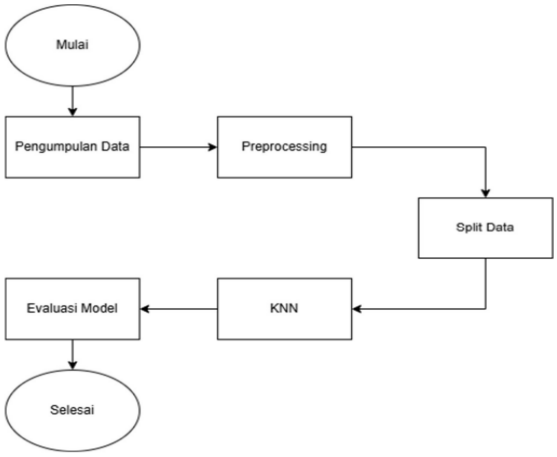
Di sisi lain, sistem rekomendasi yang dikembangkan terbukti efektif pada domain film, seleksi magang, dan, sistem rekomendasi hotel, namun kedua domain tersebut tidak memiliki kendala teknis yang kompleks seperti pada perakitan PC, yang melibatkan aturan kompatibilitas perangkat keras (*hardware compatibility*) yang ketat serta batasan anggaran (*budget constraints*) yang spesifik[2][7][13].

Penelitian ini bertujuan untuk menjembatani kesenjangan tersebut dengan mengembangkan sistem rekomendasi yang tidak hanya berfokus pada desain antarmuka atau klasifikasi teks semata, tetapi menggabungkan ketiganya. Sistem ini mengintegrasikan ekstraksi kebutuhan berbasis *Natural Language Processing* (NLP), algoritma rekomendasi KNN, dan aturan filter berbasis kendala (*constraint-based filtering*) untuk kompatibilitas dan anggaran, yang diimplementasikan pada arsitektur modern React JS dan Flask untuk menghasilkan solusi perakitan PC yang utuh dan fungsional.

3. METODE PENELITIAN

3.1. Alur Penelitian

Alur penelitian yang ditunjukkan pada Gambar 1, Penelitian ini menerapkan metode *Research and Development (R&D)* dengan pendekatan *System Development Lifecycle (SDLC)* model Waterfall. Tahapan penelitian dimulai dari studi literatur, pengumpulan data, pra-pemrosesan data, pengembangan model klasifikasi dan rekomendasi, hingga evaluasi sistem. Pendekatan ini dipilih untuk memastikan pengembangan sistem dilakukan secara terstruktur dan sistematis dari tahap analisis hingga pengujian.



Gambar 1. Alur Tahapan Penelitian

3.2. Pengumpulan Data

Data penelitian diperoleh melalui dua metode utama. Pertama, teknik web scraping digunakan untuk mengambil data komponen PC dari situs pcpartpicker. Atribut data yang dikumpulkan mencakup nama komponen, kategori (CPU, GPU, RAM, Storage, PSU, Motherboard), spesifikasi teknis (clock speed, kapasitas, socket, tipe memori, TDP), serta harga pasar terkini. Data hasil scraping dan dataset publik kemudian diselaraskan format dan atributnya untuk menghasilkan dataset final yang konsisten.

3.3. Preprocessing Data

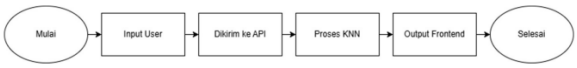
Sebelum digunakan dalam pelatihan model, dataset melalui tahap pra-pemrosesan (preprocessing) untuk memastikan kualitas data. Tahapan ini meliputi:

a. Data Cleaning: Menghapus duplikasi, menangani data kosong (*missing values*), dan membersihkan deskripsi yang tidak relevan.
b. Text Normalization: Menstandarkan penulisan teks (huruf kapital, simbol) agar mudah diproses oleh modul *Natural Language Processing* (NLP).
c. Labeling: Melabeli data berdasarkan kategori kebutuhan pengguna, yaitu "gaming" atau "multitasking".

Setelah melalui tahap ini, dataset dibagi dengan rasio 80:20, di mana 80% digunakan sebagai data latih (training data) untuk membangun pola hubungan fitur,

dan 20% sebagai data uji (testing data) untuk evaluasi objektif.

3.4. Arsitektur Sistem (NLP dan KNN)



Gambar 2. Alur penggunaan sistem

Gambar 2 menjelaskan bahwa sistem rekomendasi yang diusulkan menggabungkan pipeline NLP, algoritma *K-Nearest Neighbors* (KNN), dan *Constraint-Based Filtering*. Alur kerja sistem adalah sebagai berikut:

a. Input Pengguna: Pengguna memasukkan deskripsi kebutuhan dalam bahasa alami (teks bebas), contohnya "PC gaming 5 juta".
b. Pemrosesan NLP: *Backend* memproses input melalui tokenisasi dan ekstraksi kata kunci untuk mengidentifikasi jenis kebutuhan (gaming/multitasking) serta batasan anggaran (*budget constraint*).
c. Pencocokan KNN: Algoritma KNN menghitung kedekatan antara vektor kebutuhan pengguna dan vektor dataset komponen menggunakan metrik jarak (seperti *Euclidean Distance* atau *Cosine Similarity*) untuk mencari kandidat komponen terdekat (k tetangga).
d. Constraint Filtering: Kandidat hasil KNN disaring kembali untuk memastikan kompatibilitas perangkat keras (misal: kecocokan soket CPU dan Motherboard) serta total harga komponen tidak melebihi anggaran pengguna.
e. Output: Sistem menampilkan rekomendasi spesifikasi PC lengkap dengan estimasi harga.

3.5. Skenario Pengujian dan Evaluasi

Pengujian dilakukan menggunakan beberapa skenario variasi parameter algoritma KNN, yaitu variasi nilai k(k=3 dan k=5) serta variasi metrik jarak (Euclidean vs Manhattan Distance). Hal ini bertujuan untuk menentukan konfigurasi yang menghasilkan rekomendasi paling relevan.

Evaluasi kinerja model diukur menggunakan data uji yang belum pernah dilihat model sebelumnya. Metrik evaluasi yang digunakan meliputi:

Precision (P): Mengukur ketepatan prediksi positif.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

Recall (R): Mengukur kemampuan model menemukan kembali data yang relevan.

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

F1-Score: Rata-rata harmonic dari Precision dan Recall.

$$\frac{1}{F1} = \frac{1}{2}\left(\frac{1}{precision} + \frac{1}{recall}\right) \quad (3)$$

Accuracy: Mengukur tingkat ketepatan keseluruhan sistem.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

4. HASIL DAN PEMBAHASAN

4.1. Analisis Kebutuhan

Tahap analisis kebutuhan bertujuan untuk mengidentifikasi fungsi-fungsi utama yang harus dimiliki oleh sistem agar dapat menyelesaikan permasalahan pengguna dalam merakit PC, serta menentukan batasan teknis sistem. Berdasarkan latar belakang dan metode yang diusulkan, kebutuhan sistem dibagi menjadi dua kategori:

4.2. Kebutuhan Fungsional

a. Input Bahasa Alami: Sistem harus mampu menerima masukan berupa teks bebas dari pengguna, seperti deskripsi kebutuhan (misalnya: "PC gaming") dan nominal anggaran (misalnya: "5 juta").
b. Identifikasi Berbasis Kamus (*Keyword Mapping*): Sistem harus memiliki mekanisme pencocokan teks menggunakan kamus kata kunci (*keyword dictionary*). Sistem tidak melakukan tokenisasi kompleks, melainkan memindai input pengguna untuk menemukan kata kunci yang terdaftar dalam kamus (seperti “gaming”, “intel”. “amd”) untuk menentukan kategori penggunaan, serta mendeteksi pola angka untuk batasan anggaran (*budget*).
c. Rekomendasi Komponen (KNN): Sistem harus mampu menghitung kedekatan antara kebutuhan pengguna dengan data komponen yang tersedia menggunakan algoritma *K-Nearest Neighbors* (KNN) dengan metrik jarak seperti *Euclidean Distance*.
d. Penyaringan Kompatibilitas: Sistem wajib menerapkan aturan filter (*constraint filtering*) untuk memastikan komponen yang direkomendasikan saling kompatibel, misalnya kecocokan antara *socket* CPU dan *Motherboard*, serta memastikan total harga tidak melebihi anggaran.
e. Manajemen Data Komponen: Sistem membutuhkan basis data komponen yang mencakup atribut teknis (CPU, GPU, RAM, dll) dan harga terkini yang diperoleh melalui *web scraping*.

4.3. Kebutuhan Non-Fungsional

a. Antarmuka Pengguna (User Interface): Aplikasi harus memiliki antarmuka yang responsif dan mudah digunakan, yang diimplementasikan menggunakan teknologi *Single Page Application* (SPA) berbasis React JS.
b. Kinerja Backend: *Backend* sistem harus mampu menangani permintaan API dengan cepat dan aman menggunakan *framework* Flask, serta memproses algoritma rekomendasi dalam waktu yang wajar.
c. Akurasi Data: Data komponen harus melalui tahap *preprocessing* (pembersihan dan normalisasi) untuk menjamin kualitas rekomendasi yang dihasilkan.

4.4. Arsitektur Aplikasi

Sistem mengadopsi arsitektur client-server yang terpisah:

a. Frontend (React JS): Berfungsi sebagai antarmuka input dan visualisasi hasil. Komponen ini mengirimkan *string* input pengguna ke server dan menerima respons rekomendasi dalam format JSON.
b. Backend (Flask dan Flash): Berfungsi sebagai pusat pemrosesan logika. Di sinilah tersimpan kamus kata kunci untuk proses NLP sederhana, logika algoritma KNN, dan aturan-aturan kompatibilitas (*rule-based constraints*).

4.5. Alur Proses Sistem (NLP & KNN)

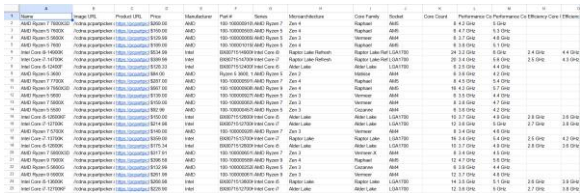
Berdasarkan alur penelitian, mekanisme kerja sistem dirancang sebagai berikut:

a. Pencocokan Kata Kunci (Keyword Matching):
Saat pengguna memasukkan teks, sistem tidak memecah kalimat menjadi token gramatikal. Sebaliknya, modul NLP di backend membandingkan string input dengan daftar kata dalam kamus kata kunci yang telah didefinisikan sebelumnya.
   - Jika ditemukan kata "game" atau "gaming", sistem menetapkan kategori kebutuhan sebagai "Gaming".
   - Jika ditemukan pola angka (misal "5000000" atau "5 juta"), sistem menetapkannya sebagai batasan anggaran (*budget constraint*)[21].

b. Pencarian Kandidat (KNN):
Parameter yang diperoleh dari hasil pencocokan kamus (Kategori & Budget) dikonversi menjadi vektor target. Algoritma KNN kemudian menghitung jarak (distance) antara vektor target ini dengan data komponen di database untuk mengambil *k* komponen yang paling mirip karakteristiknya.
c. Penyaringan Kendala (Constraint Filtering):
Kandidat komponen hasil KNN divalidasi ulang. Sistem memeriksa aturan kompatibilitas fisik (misalnya: socket LGA1700 harus dipasangkan dengan motherboard yang mendukung LGA1700) dan memverifikasi bahwa total harga paket tidak melampaui budget yang telah diekstraksi dari input.
d. Output Rekomendasi:
Paket komponen yang lolos validasi dikirimkan ke frontend untuk ditampilkan sebagai rekomendasi spesifikasi PC yang siap rakit.

4.6. Hasil Pengambilan Data

Data komponen yang menjadi basis pengetahuan (*knowledge base*) sistem diperoleh melalui teknik *web scraping* dari situs penyedia spesifikasi perangkat keras komputer. Data mentah yang dikumpulkan disimpan dalam format terstruktur (CSV) untuk memudahkan proses pembacaan oleh sistem *backend*.



Gambar 3. Hasil scraping data cpu

Gambar 3 memperlihatkan sampel dataset mentah untuk kategori *Central Processing Unit* (CPU) yang berhasil dikumpulkan. Dataset ini mencakup berbagai generasi prosesor, mulai dari arsitektur AMD Zen 3 dan Zen 4 hingga Intel Alder Lake dan Raptor Lake.

4.7. Performa Model

Evaluasi kinerja sistem rekomendasi dilakukan menggunakan metode *Confusion Matrix* untuk mengukur seberapa akurat sistem dalam mengklasifikasikan spesifikasi komponen (CPU dan GPU) ke dalam kategori "Gaming" atau "Multitasking" sesuai dengan input pengguna. Pengujian dilakukan terhadap 1.742 skenario data uji yang mencakup rentang anggaran dari Rp 3.000.000 hingga Rp 90.000.000.

Hasil evaluasi diukur berdasarkan empat parameter utama: *Accuracy*, *Precision*, *Recall*, dan *F1-Score*. Ringkasan hasil pengujian dapat dilihat pada Tabel 1. Berdasarkan Tabel 1, sistem menghasilkan akurasi keseluruhan (*overall accuracy*) sebesar **70%**. Analisis mendalam terhadap masing-masing kelas adalah sebagai berikut:

a. Analisis Kategori Gaming Kategori *Gaming* memiliki nilai Recall sempurna (1.00), yang menunjukkan bahwa sistem berhasil mengenali seluruh data yang seharusnya dikategorikan sebagai PC Gaming tanpa ada yang terlewat (tidak ada *False Negative*). Namun, nilai Precision (0.63) menunjukkan adanya tingkat *False Positive* yang cukup tinggi. Hal ini mengindikasikan bahwa sistem memiliki kecenderungan (*bias*) untuk merekomendasikan spesifikasi *Gaming* pada kondisi yang ambigu. Ketika komponen memiliki spesifikasi "Balanced" (dapat digunakan untuk kedua tujuan, misalnya pada *budget* menengah), algoritma KNN dan *Constraint Filtering* cenderung mengarahkan rekomendasi ke profil *Gaming* untuk memastikan performa grafis tetap optimal.
b. Analisis Kategori Multitasking Kategori *Multitasking* menunjukkan nilai Precision yang sangat tinggi (0.99). Artinya, ketika sistem memprediksi sebuah spesifikasi sebagai PC Multitasking, prediksi tersebut 99% benar dan sangat spesifik (misalnya PC dengan *Core Count* tinggi atau VRAM besar). Namun, nilai Recall (0.40) tergolong rendah. Hal ini disebabkan oleh ketatnya kriteria *Multitasking* dalam sistem. Banyak PC yang sebenarnya mumpuni untuk *multitasking* ringan (terutama di anggaran rendah

hingga menengah) diklasifikasikan oleh sistem sebagai PC *Gaming* karena spesifikasi komponennya yang mirip. Sistem hanya melabeli *Multitasking* jika komponen benar-benar memiliki karakteristik *workstation* yang kuat (seperti prosesor >12 core atau GPU non-gaming).

c. Kesimpulan Evaluasi Secara keseluruhan, model rekomendasi ini memiliki karakteristik "Gaming-Safe". Sistem memprioritaskan keamanan performa grafis, sehingga pengguna yang meminta PC *Multitasking* dengan *budget* terbatas mungkin akan mendapatkan rekomendasi dengan profil *Gaming*. Meskipun demikian, hal ini dapat diterima karena spesifikasi *Gaming* pada umumnya juga mampu menangani beban kerja *multitasking* standar. Sebaliknya, untuk kebutuhan *Multitasking* berat (*High-End*), sistem mampu memberikan rekomendasi yang sangat presisi sesuai kebutuhan profesional.

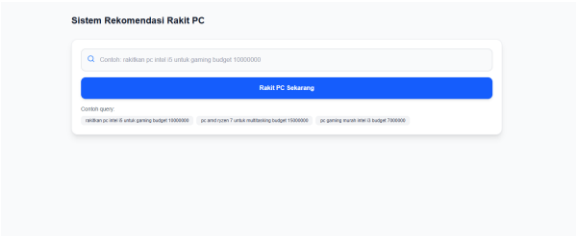
Tabel 1. Evaluasi model

Kategori	Precision	Recall	F1-Score	Support
Gaming	0.63	1.00	0.77	871
Multitasking	0.99	0.40	0.57	871
Accuracy			0.70	1742
Macro Avg	0.81	0.70	0.67	1742
Weighted Avg	0.81	0.70	0.67	1742

4.8. Implementasi Antarmuka Sistem

Implementasi antarmuka pengguna (*User Interface*) dibangun menggunakan teknologi React JS untuk memastikan interaksi yang responsif dan dinamis. Antarmuka dirancang dengan pendekatan minimalis untuk memudahkan pengguna awam dalam mengoperasikan sistem tanpa perlu menavigasi menu yang kompleks. Berikut adalah rincian tampilan antarmuka sistem:

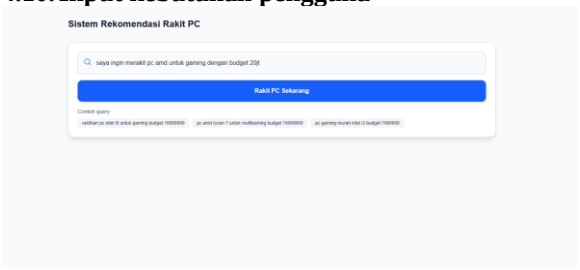
4.9. Halaman utama



Gambar 4. Halaman utama

Gambar 4 menjelaskan bahwa, Halaman utama merupakan gerbang interaksi antara pengguna dan sistem. Seperti yang terlihat pada Gambar 4, halaman ini didominasi oleh kolom pencarian (search bar) yang ditempatkan di tengah layar untuk memusatkan fokus pengguna pada input teks. Di bawah kolom pencarian, sistem menyediakan "Contoh Query" (seperti: "rakitkan pc intel i5 untuk gaming budget 10000000") untuk memberikan panduan implisit kepada pengguna mengenai format kalimat yang dapat diproses oleh sistem. Tombol "Rakit PC Sekarang" berfungsi sebagai pemicu (trigger) untuk mengirimkan request ke API backend.

4.10. Input kebutuhan pengguna



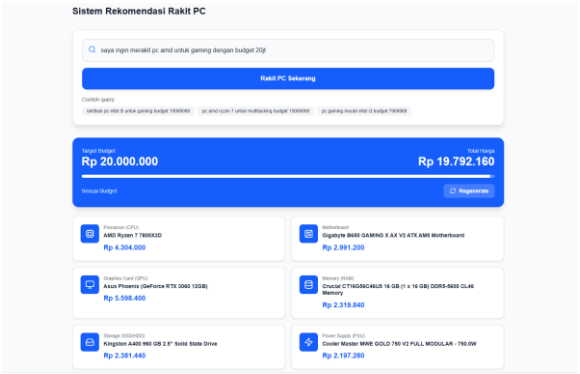
Gambar 5. Input pengguna

Terlampir paga Gambar 5, Pada tahap ini, pengguna memasukkan preferensi mereka menggunakan bahasa alami. Sistem tidak menggunakan *dropdown* atau *checkbox* yang kaku, melainkan membiarkan pengguna mengetik kebutuhan layaknya berbicara dengan asisten manusia. Sebagaimana ditunjukkan pada Gambar 5, pengguna memasukkan *query*: *"saya ingin merakit pc amd untuk gaming dengan budget 20jt"*. Berdasarkan input ini, modul NLP di backend akan mengekstrak kata kunci vital, yaitu:

a. Platform: AMD
b. Kategori: Gaming
c. Anggaran: Rp 20.000.000

Mekanisme ini membuktikan fleksibilitas sistem dalam menangani input teks bebas tanpa format yang ketat.

4.11. Hasil rekomendasi



Gambar 6. Hasil rekomendasi

Gambar 6 menjelaskan, Setelah tombol proses ditekan, sistem menampilkan halaman hasil rekomendasi seperti yang terlihat pada Gambar 6. Halaman ini dibagi menjadi dua segmen informasi utama:

a. Ringkasan Anggaran (*Budget Overview*): Bagian atas menampilkan bilah status (*progress bar*) yang membandingkan Target Budget (Rp 20.000.000) dengan Total Harga komponen yang direkomendasikan (Rp 19.792.160). Visualisasi ini memudahkan pengguna untuk melihat seberapa

efisien sistem dalam memaksimalkan anggaran tanpa terjadinya *over-budget*.

b. Rincian Komponen: Sistem menyajikan daftar komponen lengkap hasil algoritma KNN yang telah divalidasi kompatibilitasnya. Berdasarkan input pengguna (Gaming, AMD, 20 Juta), sistem merekomendasikan:
   - CPU: AMD Ryzen 7 7800X3D (Prosesor *high-end* yang sangat optimal untuk *gaming*).
   - GPU: Asus Phoenix GeForce RTX 3060 12GB (Kartu grafis yang mumpuni untuk *gaming* 1080p/1440p).
   - Motherboard: Gigabyte B650 Gaming X AX (Platform AM5 yang sesuai dengan prosesor).
   - Komponen Pendukung: RAM DDR5 16GB, SSD 960GB, dan PSU 750W.

Tampilan ini menunjukkan bahwa sistem berhasil mengintegrasikan preferensi "AMD" dan "Gaming" ke dalam satu paket rakitan yang utuh. Selain itu, terdapat tombol "Regenerate" yang memungkinkan pengguna untuk meminta variasi rekomendasi lain dengan parameter yang sama.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian dan pengujian yang telah dilakukan, dapat disimpulkan bahwa sistem rekomendasi rakit PC otomatis berhasil dibangun dengan mengintegrasikan antarmuka *frontend* React JS dan *backend* Flask yang responsif. Penerapan metode *Natural Language Processing* (NLP) berbasis kamus kata kunci mampu mengekstraksi kebutuhan pengguna dan batasan anggaran secara efektif, yang kemudian diolah oleh algoritma *K-Nearest Neighbors* (KNN) dan *Constraint-Based Filtering* untuk menghasilkan rekomendasi komponen yang kompatibel. Evaluasi kinerja menunjukkan sistem memiliki akurasi keseluruhan sebesar 70% dengan karakteristik model yang cenderung *"Gaming-Safe"*, dibuktikan dengan nilai *Recall* sempurna (1.00) pada kategori *Gaming*. Hal ini menandakan sistem sangat andal dalam menjamin performa grafis pengguna, meskipun masih memiliki bias yang menyebabkan spesifikasi *balanced* sering diklasifikasikan sebagai *Gaming* daripada *Multitasking*. Untuk pengembangan selanjutnya, disarankan agar penelitian berfokus pada penyempurnaan logika pembobotan algoritma untuk meningkatkan sensitivitas terhadap kategori *Multitasking* yang saat ini memiliki nilai *Recall* rendah (0.40), sehingga sistem dapat lebih jeli membedakan kebutuhan *workstation* murni dengan *gaming*. Selain itu, pengembangan modul NLP dapat ditingkatkan dari pendekatan berbasis kamus sederhana menjadi model *Deep Learning* (seperti BERT) agar mampu memahami konteks kalimat yang lebih kompleks dan ambigu. Perluasan dataset komponen serta integrasi fitur pembaruan harga secara *real-time* melalui API *e-commerce* juga sangat direkomendasikan untuk menjaga relevansi estimasi harga rakitan dengan kondisi pasar yang fluktuatif.

DAFTAR PUSTAKA

[1] D. M. Zidan, A. P. Kharisma, and M. T. Ananta, "Perancangan User Experience Aplikasi Merakit Komputer Pribadi," vol. 7, no. 1, pp. 131–139, 2023.

[2] A. R. Wahono, B. A. Saputra, and F. F. Rahman, "Sistem Rekomendasi Film Menggunakan Metode Content-Based Filtering dan Algoritma K-Nearest Neighbors (KNN)," pp. 1–6, 2024.

[3] R. Puspitasari, Y. Findawati, M. A. Rosid, I. S. Program, and U. M. Sidoarjo, "SENTIMENT ANALYSIS OF POST-COVID-19 INFLATION BASED ON TWITTER USING THE K-NEAREST NEIGHBOR AND SUPPORT VECTOR MACHINE ANALISIS SENTIMEN TERHADAP INFLASI PASCA COVID-19 BERDASARKAN TWITTER DENGAN METODE KLASIFIKASI K-NEAREST NEIGHBOR DAN," vol. 4, no. 4, pp. 669–679, 2023.

[4] A. Leo, "WEB MENGGUNAKAN FRAMEWORK REACT . JS," vol. 1, 2024.

[5] B. P. Putra *et al.*, "IMPLEMENTASI API MASTER STORE MENGGUNAKAN FLASK , REST DAN ORM DI PT XYZ," vol. 9, no. September, pp. 543–556, 2020.

[6] M. K. Neighbor, P. Aplikasi, R. M. Affandi, and D. Hermanto, "Screening CV Otomatis Dalam Natural Language Processing," vol. 1, no. 1, pp. 13–21, 2025.

[7] I. K. D. Nuryana, "PENERIMAAN PESERTA MAGANG MENGGUNAKAN NATURAL LANGUAGE PROCESSING & ALGORITMA K-NEAREST NEIGHBORS PADA PT . IDE JUALAN CREATIVE," vol. 04, no. 03, pp. 178–185, 2023.

[8] S. Retno, R. K. Dinata, and N. Hasdyna, "Jurnal Computer Science and Information Technology (CoSciTech)," vol. 4, no. 1, pp. 146–153, 2023.

[9] P. Studi, T. Fakultas, T. Informasi, U. Kristen, and S. Wacana, "IMPLEMENTASI FRAMEWORK FLASK PADA MODUL BETA-APP PADA APLIKASI SISTEM INFORMASI HELPDESK (SIH) STUDI," vol. 23, no. 2, pp. 243–258, 2023.

[10] J. Sistem and I. Cendekia, "Perancangan Sistem Informasi Penyewaan Thermoking di PT . Moderen Prima dengan Flask Python," vol. 1, no. 1, pp. 19–28, 2023.

[11] J. Informatika, F. Teknologi, and U. I. Indonesia, "Penerapan React JS Pada Pengembangan FrontEnd Aplikasi Startup Ubaform".

[12] V. C. Mawardi, "PEMBUATAN WEBSITE SEWA KANTOR MENGGUNAKAN," pp. 2–6.

[13] A. Muliawan, T. Badriyah, I. Syarif, C. B. Filtering, and H. Formula, "Membangun Sistem Rekomendasi Hotel dengan Content Based Filtering Menggunakan K-Nearest Neighbor dan Haversine Formula," vol. 7, no. 2, pp. 231–247, 2022.