

Pengembangan Prototype Fullstack Pada Otomatisasi dan Digitalisasi Administrasi Pelayanan Rumah Sakit

1st Muhammad Adam Majid

Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

madammajid@student.telkomuniversity.ac.id

2nd Rita Purnamasari

Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

rustamtelu@telkomuniversity.ac.id

3rd Rustam

Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia

ritapurnamasari@telkomuniversity.ac.id

Abstrak — Penerapan program BPJS (Badan Penyelenggara Jaminan Sosial) di rumah sakit menghadapi beberapa tantangan, terutama dalam otomatisasi dan digitalisasi administrasi. Tantangan utama yang muncul adalah kesalahan dalam pencatatan kode ICD-10 (*The 10th revision of International Statistical Classification of Diseases*) oleh dokter dan hambatan verifikasi klaim INA-CBG (*Indonesia Case Base Groups*) secara manual oleh staf rekam medis. Sebagai solusi, dikembangkan sebuah sistem otomatisasi yang mengintegrasikan dokter dengan staf rekam medis, termasuk fitur pencarian kode ICD-10 melalui tiga metode yaitu pencarian berdasarkan prediksi bab penyakit menggunakan model SVC (*Support Vector Classifier*) kernel RBF (*Radial Basis Function*), pencarian dengan *stopwords* dari pustaka NLTK (*Natural Language Toolkit*), dan pencarian menggunakan NLP (*Natural Language Processing*) dengan model *en_core_web_lg* dari pustaka SpaCy. Hasil penelitian menunjukkan bahwa pencarian kode ICD-10 yang dilakukan secara manual dan melalui sistem menghasilkan hasil yang sama, namun pencarian melalui sistem memberikan lebih banyak opsi kode ICD-10. Sistem ini mendukung pencarian kode ICD-10 dalam Bahasa Indonesia dan mempermudah verifikasi klaim INA-CBG secara digital, sehingga meningkatkan efisiensi administrasi pelayanan di rumah sakit.

Kata kunci—BPJS, ICD-10, INA-CBG, Otomatisasi, Digitalisasi

I. PENDAHULUAN

Penerapan program BPJS di rumah sakit menghadapi beberapa tantangan terkait otomatisasi dan digitalisasi administrasi. BPJS Kesehatan adalah program jaminan kesehatan sebagaimana dimaksud dalam Undang-Undang Nomor 24 Tahun 2011 tentang Badan Penyelenggara Jaminan Sosial [1]. BPJS berperan penting dalam memastikan akses layanan kesehatan yang terjangkau dan merata [2]. Namun pelaksanaannya sering kali terhambat oleh masalah administratif. ICD-10 adalah Klasifikasi Statistik Internasional Tentang Penyakit dan Masalah Kesehatan Revisi ke 10 yang dikeluarkan oleh WHO (*World Health Organization*) [3]. Klaim INA-CBG adalah pembayaran yang dilakukan oleh BPJS Kesehatan kepada

rumah sakit untuk paket layanan tertentu yang ditentukan melalui pengelompokan diagnosis penyakit dan prosedur oleh staf rekam medis [4]. Kesalahan dalam pencatatan kode ICD-10 oleh dokter dapat menyebabkan ketidakakuratan dalam data medis dan klaim INA-CBG [5]. Sementara proses verifikasi klaim INA-CBG secara manual oleh staf rekam medis dapat memperlambat administrasi.

Penggunaan sistem otomatisasi untuk memperbaiki proses pencatatan dan verifikasi klaim telah diusulkan untuk masalah ini. Pengembangan sistem otomatisasi yang mengintegrasikan dokter dan staf rekam medis. Sistem ini dilengkapi dengan fitur pencarian kode ICD-10 yang menerapkan tiga metode berbeda yaitu pencarian berdasarkan prediksi bab penyakit menggunakan model SVC dengan kernel RBF, pencarian dengan *stopwords* dari pustaka NLTK, dan pencarian menggunakan NLP dengan model *en_core_web_lg* dari pustaka SpaCy. Selain itu, verifikasi klaim INA-CBG juga dilakukan secara otomatis berdasarkan kode ICD-10 dan diagnosanya.

Penelitian ini bertujuan untuk mengatasi masalah pencatatan kode ICD-10 dan verifikasi klaim INA-CBG yang sering mengalami kesalahan dan keterlambatan. Dengan mengembangkan dan menerapkan sistem otomatisasi ini, diharapkan proses pencatatan dan verifikasi dapat menjadi lebih efisien. Sehingga dapat meningkatkan kualitas administrasi pelayanan kesehatan di rumah sakit.

II. KAJIAN TEORI

A. *Support Vector Classifier* (SVC)

SVC merupakan varian dari *Support Vector Machines* (SVMs) yang digunakan untuk masalah klasifikasi. SVC bekerja dengan membagi data ke dalam kategori yang berbeda berdasarkan fitur yang diberikan [6]. Perbedaan utama SVC dengan alat klasifikasi lainnya, seperti *knowledge-based neural networks*, adalah bahwa SVC biasanya tidak memanfaatkan pengetahuan sebelumnya secara langsung dalam proses klasifikasinya [6]. Namun, SVC dapat diperluas untuk memasukkan pengetahuan sebelumnya dalam bentuk kumpulan pengetahuan poligonal

yang dapat digambarkan sebagai generalisasi dari contoh pelatihan [6].

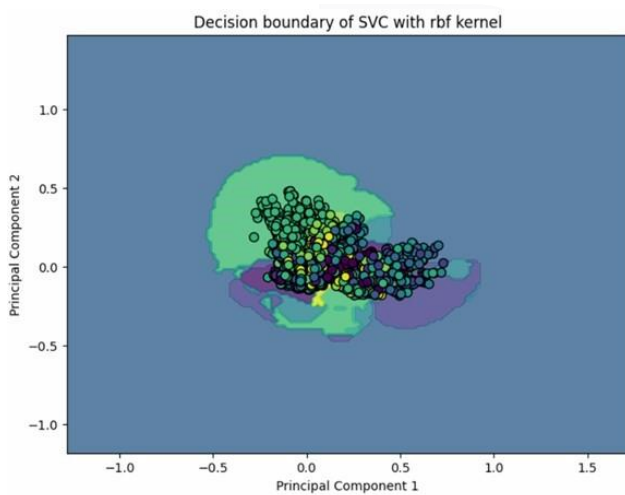
B. Kernel Radial Basis Function (RBF)

SVC kernel RBF adalah metode yang efektif untuk menangani masalah klasifikasi non-linear [7]. Fungsi ini sangat berguna karena kemampuannya dalam menangani data yang tidak dapat dipisahkan secara linear dengan memetakan data ke dimensi yang lebih tinggi [7]. Oleh karena itu, RBF sering digunakan dalam berbagai aplikasi untuk meningkatkan akurasi klasifikasi pada data yang kompleks [7].

Fungsi kernel RBF dihitung berdasarkan jarak antara n titik data dalam ruang fitur, menggunakan rumus Gaussian [7]. Misalnya, jika n titik data di ruang fitur, maka fungsi kernel RBF bisa dinyatakan sebagai:

$$K(x, x_n) = \exp(-\gamma \|x - x_n\|^2) \quad [7]$$

γ adalah parameter yang mengontrol jarak pengaruh dari sebuah titik data, dan $\|x - x_n\|^2$ adalah jarak Euclidean antara x dan x_n [7]. Nilai kernel RBF akan tinggi ketika dua titik berada dekat satu sama lain dan menurun saat jaraknya bertambah [7]. Dengan memproyeksikan data ke dalam dimensi yang lebih tinggi, RBF kernel memungkinkan SVM untuk menemukan hyperplane yang lebih optimal untuk memisahkan kelas-kelas dalam data non-linear [7].



GAMBAR 1
Pemetaan Data dengan kernel RBF

Gambar 1 menampilkan pemetaan dataset menggunakan model SVC dengan kernel RBF. Pada gambar tersebut, batas keputusan SVC dengan kernel RBF terlihat lebih kompleks dan non-linear, menciptakan ruang keputusan yang tidak hanya dipisahkan oleh garis lurus, melainkan mengikuti pola yang lebih rumit sesuai dengan distribusi data. Pemetaan ini diilustrasikan menggunakan dua komponen utama (Komponen Utama 1 dan Komponen Utama 2), yang memperlihatkan bagaimana model dengan kernel RBF mengelompokkan data dalam ruang dua dimensi.

C. Stopwords dari Pustaka Natural Language Toolkit (NLTK)

Stopwords adalah kata-kata yang sering muncul dalam teks tetapi memiliki sedikit atau tidak ada makna informatif

dalam konteks tertentu [8]. Dalam NLP dan analisis teks, *stopwords* umumnya adalah kata-kata yang tidak memberikan kontribusi signifikan terhadap pemahaman atau pemrosesan teks, seperti "the", "is", "in", "and", "of", dan sebagainya [8]. Pada tahap pra-pemrosesan data teks, *stopwords* dihapus untuk meningkatkan rasio sinyal terhadap noise, sehingga hanya kata-kata yang lebih bermakna atau relevan yang tersisa untuk analisis lebih lanjut [8].

NLTK adalah sebuah pustaka Python yang banyak digunakan dalam pemrosesan bahasa alami (NLP) [9]. Pustaka ini menyediakan berbagai alat dan sumber daya yang mendukung berbagai tugas NLP, seperti tokenisasi, stemming, tagging, parsing, dan klasifikasi teks [9]. NLTK dirancang untuk memudahkan para peneliti, pengembang, dan praktisi dalam bekerja dengan teks dan melakukan analisis bahasa alami [9].

Dalam proses pra-pemrosesan teks, *stopwords* dari pustaka NLTK sering kali dihapus untuk meningkatkan efisiensi dan efektivitas model NLP yang dikembangkan [10]. Dengan menghapus *stopwords*, analisis dapat lebih fokus pada kata-kata yang lebih bermakna dan relevan, yang dapat membantu dalam meningkatkan akurasi tugas-tugas seperti pengindeksan, klasifikasi teks, dan pemodelan topik [10].

D. NLP dengan Model en_core_web_lg dari Pustaka SpaCy

NLP adalah bidang dalam ilmu komputer yang berfokus pada interaksi antara komputer dan bahasa manusia [11]. Untuk mendukung berbagai aplikasi NLP, penelitian menggunakan model en_core_web_lg dari pustaka SpaCy. SpaCy adalah pustaka Python yang dirancang untuk pemrosesan bahasa alami yang cepat dan efisien, dan menyediakan model bahasa pra-latih yang siap digunakan untuk berbagai tugas NLP, seperti pengenalan entitas bernama (NER), analisis sintaksis, dan pemodelan dependensi [11].

Model en_core_web_lg dari SpaCy adalah salah satu model Bahasa Inggris terbesar dan dilatih pada data ekstensif untuk menghasilkan representasi vektor kata yang kaya [12]. Model ini memungkinkan pengguna melakukan tugas-tugas NLP seperti tokenisasi, pelabelan POS, dan ekstraksi fitur semantik dengan presisi tinggi [12]. Dalam pra-pemrosesan teks, en_core_web_lg berguna untuk mengidentifikasi dan menghapus *stopwords* secara otomatis, sehingga memperkuat analisis pada kata-kata yang relevan [12]. Dengan kemampuan untuk menangani teks yang kompleks, model ini menawarkan fleksibilitas dan efisiensi, menjadikannya pilihan unggul bagi peneliti dan praktisi NLP [12].

E. Python Flask API

Python Flask API adalah *framework* yang ringkas dan serbaguna untuk mengembangkan aplikasi web dan antarmuka pemrograman aplikasi (API) dengan Python [13]. Flask menawarkan struktur yang sederhana namun kuat, memungkinkan pengembang untuk membuat *endpoint* API yang dapat diakses melalui protokol HTTP [13]. Dengan pendekatan yang minimalis dan modular, Flask mempermudah integrasi dengan berbagai pustaka dan ekstensi, serta mendukung pengembangan aplikasi yang dapat diskalakan dan terorganisir dengan baik [13]. Flask API sering digunakan untuk membangun layanan *backend* yang

mendukung aplikasi *front-end*, memungkinkan manajemen data dan komunikasi antar sistem dengan arsitektur RESTful [13]. Kemampuan Flask API dalam menangani permintaan HTTP, mengelola rute, dan memproses data JSON menjadikannya pilihan yang banyak dipilih oleh pengembang untuk solusi yang cepat dan efisien dalam pembuatan dan pengelolaan API berbasis web [13].

F. Pengelolaan Sistem dengan Google Cloud Platform (GCP)

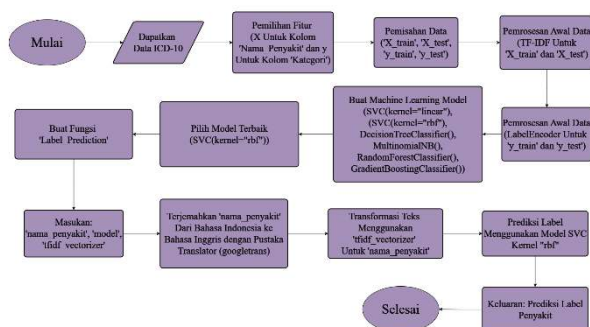
Pengelolaan database dengan SQL dari GCP merupakan pendekatan untuk mengelola, menyimpan, dan mengakses data secara efisien di lingkungan cloud [14]. GCP menyediakan berbagai layanan database yang mendukung SQL, seperti Cloud SQL, yang memungkinkan pengguna untuk menjalankan database relasional seperti MySQL, PostgreSQL, dan SQL Server di infrastruktur cloud [14]. Pengelolaan database dengan SQL di GCP tidak hanya mempermudah administrasi dan pemeliharaan database, tetapi juga meningkatkan fleksibilitas dan efisiensi dalam menangani beban kerja yang bervariasi, serta mendukung analisis data dan aplikasi yang kompleks di lingkungan cloud [14].

Cloud Run adalah platform komputasi terkelola dari Google yang dirancang untuk menjalankan aplikasi berbasis *container* dengan skala otomatis [15]. Layanan ini menyediakan lingkungan eksekusi yang sepenuhnya dikelola, sehingga pengembang dapat fokus pada pengembangan aplikasi tanpa perlu khawatir tentang manajemen infrastruktur [15]. Dengan Cloud Run, dapat menjalankan *container* di atas infrastruktur Google yang kuat, yang secara otomatis menyesuaikan kapasitas dan skala sesuai dengan kebutuhan aplikasi [15].

III. METODE

Sistem untuk otomatisasi dan digitalisasi administrasi pelayanan rumah sakit menggunakan tahapan metode yang dijelaskan melalui poin berikut:

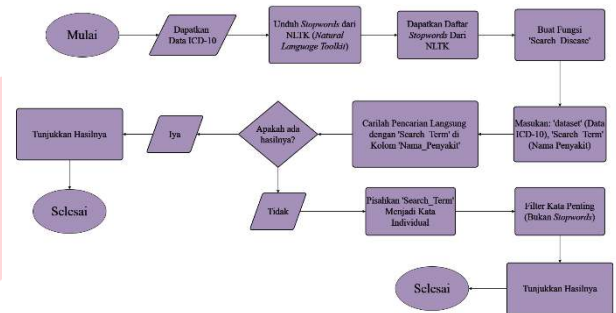
A. Desain Bagan Alir Tiga Sistem Pencarian Kode ICD-10



GAMBAR 2
Bagan Alir Fungsi Prediksi Bab Penyakit

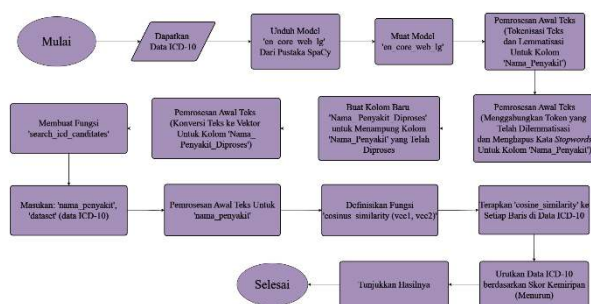
Bagan alir pada Gambar 2 menggambarkan sistem klasifikasi penyakit berbasis teks menggunakan *machine learning*. Proses dimulai dengan memuat data dari file CSV ke dalam DataFrame, membagi data menjadi set pelatihan dan pengujian dengan proporsi 80% dan 20%, serta

memastikan proporsi kategori yang konsisten. Nama penyakit diubah menjadi fitur numerik menggunakan `TfidfVectorizer` dan label kategori dikonversi dengan `LabelEncoder`. Beberapa model klasifikasi, termasuk SVC dengan kernel linear dan RBF, Decision Tree, Naive Bayes, Random Forest, dan Gradient Boosting, dilatih dan dievaluasi. Model SVM dengan kernel RBF dipilih sebagai yang terbaik dan digunakan untuk memprediksi kategori penyakit. Input nama penyakit dari pengguna diterjemahkan dari Bahasa Indonesia ke Bahasa Inggris dengan Google Translate API, kemudian diproses untuk prediksi kategori dan pencarian kode ICD-10 berdasarkan hasil prediksi.



GAMBAR 3
Bagan Alir Fungsi Pencarian dengan *Stopwords* dari Pustaka NLTK

Bagan alir pada Gambar 3 menggambarkan pencarian penyakit dalam dataset berdasarkan istilah pencarian. Proses dimulai dengan mengimpor modul NLTK dan mengunduh daftar *stopwords* untuk praproses teks. Fungsi `search_disease` mencari kecocokan langsung dengan istilah pencarian dalam kolom `Nama_Penyakit`. Jika tidak ditemukan hasil, fungsi membagi istilah pencarian menjadi kata-kata penting yang bukan *stopwords* dan mencari kecocokan kata-kata tersebut dalam kolom `Nama_Penyakit`. Hasil akhir, yang mencakup entri penyakit relevan, dikembalikan dalam bentuk `DataFrame`.

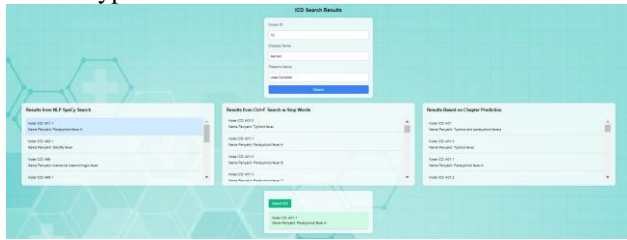


GAMBAR 4
Bagan Alir Fungsi Pencarian Menggunakan NLP

Bagan alir pada Gambar 4 menggunakan model bahasa SpaCy untuk menemukan kandidat ICD-10 yang relevan berdasarkan nama penyakit yang dimasukkan. Model SpaCy besar (`en_core_web_lg`) digunakan untuk memproses teks, dengan fungsi `preprocess_text` yang menghilangkan *stopwords* dan mengubah kata-kata menjadi bentuk lemma. Dataset diperbarui dengan kolom baru berisi nama penyakit yang telah dipraproses dan diubah menjadi vektor menggunakan model SpaCy. Fungsi `search_icd_candidates` kemudian mencari kandidat ICD-10 yang paling relevan

dengan kueri, menghitung kemiripan antara vektor kueri dan vektor dalam dataset. Hasilnya diurutkan berdasarkan kemiripan, dan 10 entri teratas dikembalikan sebagai kandidat ICD-10 yang relevan.

B. Prototype Website



GAMBAR 5
Sisi Website Untuk Dokter

Gambar 5 menunjukkan tampilan antarmuka prototipe website yang dirancang untuk dokter. Website ini memiliki beberapa fitur utama yang ditujukan untuk memudahkan dokter dalam pemilihan kode ICD-10. Pada tampilan ini, terdapat tiga bagian utama, yaitu kolom masukan dari dokter, pencarian untuk kode ICD-10, dan pemilihan kode ICD-10. Antarmuka ini dirancang dengan sederhana dengan tujuan pengembang website dapat mengubah menjadi lebih interaktif, dinamis, dan fungsional.



GAMBAR 6
Sisi Website Untuk Staf Rekam Medis

Gambar 6 menampilkan antarmuka website yang dirancang khusus untuk staf rekam medis. Pada tampilan ini, staf rekam medis dapat melihat data pasien yang telah dimasukkan oleh dokter, serta dapat memilih untuk siapa dokternya, nama pasien, dan nama penyakit. Setelah itu maka kode ICD-10 akan muncul otomatis sesuai dengan pilihan yang telah dipilih. Pada sisi kanan terdapat kolom kata pencarian yang dapat dimasukkan oleh staf rekam medis untuk pengecekan verifikasi klaim INA-CBG yang telah disesuaikan dengan bab yang ada pada dokumen tersebut. Antarmuka ini dirancang untuk mendukung efisiensi kerja staf rekam medis dalam verifikasi klaim INA-CBG. Perancangan antarmuka website sederhana ini untuk memudahkan pengembang website selanjutnya.

IV. HASIL DAN PEMBAHASAN

Pengujian halaman dokter di website mencakup input diagnosa, pencarian, dan pemilihan kode ICD-10. Dokter harus dapat memasukkan diagnosa, mencari kode ICD-10

yang sesuai, serta memilih dan memverifikasi kode ICD-10 yang relevan.

Classification Report:				
	precision	recall	f1-score	support
Chapter I	0.62	0.98	0.76	184
Chapter II	0.93	0.82	0.87	82
Chapter III	0.88	0.55	0.68	38
Chapter IV	0.89	0.86	0.87	72
Chapter IX	0.94	0.92	0.93	91
Chapter V	0.98	0.84	0.91	69
Chapter VI	0.77	0.80	0.78	79
Chapter VII	0.92	0.79	0.85	57
Chapter VIII	0.96	0.93	0.94	27
Chapter X	0.90	0.93	0.91	56
Chapter XI	0.93	0.84	0.88	94
Chapter XII	0.93	0.82	0.87	80
Chapter XIII	0.91	0.72	0.81	43
Chapter XIV	0.92	0.80	0.86	87
Chapter XIX	0.76	0.90	0.82	278
Chapter XV	0.94	0.92	0.93	92
Chapter XVI	0.99	0.94	0.96	78
Chapter XVII	0.87	0.70	0.83	141
Chapter XVIII	0.87	0.59	0.70	68
Chapter XX	0.64	0.46	0.54	74
Chapter XXI	0.97	0.87	0.92	143
Chapter XXII	1.00	0.95	0.98	22
accuracy			0.84	1955
macro avg	0.89	0.82	0.85	1955
weighted avg	0.86	0.84	0.84	1955

GAMBAR 7
Akurasi *F1-Score* Pada Model SVC Kernel RBF

Gambar 7 menunjukkan laporan klasifikasi dari model SVC dengan RBF. Laporan ini mencakup metrik *precision*, *recall*, *f1-score*, dan *support* untuk setiap chapter I hingga XXII (karena bab penyakitnya pada ICD-10 terdapat 22 bab). *Precision* mengukur ketepatan prediksi positif, sedangkan *recall* mengukur kemampuan model menangkap semua sampel positif. *F1-Score*, rata-rata harmonik dari *precision* dan *recall*, memberikan gambaran performa yang seimbang. Model ini memiliki *f1-score* rata-rata 84%, dengan chapter XX memiliki *f1-score* terendah (54%) dan chapter XXII tertinggi (98%).

result_df = search_disease(df_idc, "have a cold")			search_disease(df_idc, 'cold')		
result_df			result_df		
Kategori	Kode_ICD	Nama_Penyakit	Kategori	Kode_ICD	Nama_Penyakit
3495	Chapter X	J00 Acute nasopharyngitis [common cold]	3495	Chapter X	J00 Acute nasopharyngitis [common cold]
4410	Chapter XII	L50.2 Urticaria due to cold and heat	4410	Chapter XII	L50.2 Urticaria due to cold and heat
6089	Chapter XVI	P80.0 Cold injury syndrome	6089	Chapter XVI	P80.0 Cold injury syndrome
8234	Chapter XIX	T48.5 Anti-common-cold drugs	8234	Chapter XIX	T48.5 Anti-common-cold drugs
8869	Chapter XX	Y55.5 Anti-common-cold drugs	8869	Chapter XX	Y55.5 Anti-common-cold drugs

GAMBAR 8
Pengujian Pencarian Menggunakan *Stopwords* Dari Pustaka NLTK

Pada sisi kiri gambar 8, contoh kata input adalah "have a cold". Sistem bekerja dengan menghapus kata-kata umum seperti "have" dan "a", sehingga hanya menyisakan kata "cold". Sistem kemudian mencari kata-kata yang relevan dengan "cold", sebagaimana ditunjukkan dalam gambar. Hasil pencarian ini konsisten dengan hasil dari sistem di sisi kanan gambar, di mana input kata adalah "cold". Ini menunjukkan bahwa metode pencarian yang menghilangkan *stopwords*, seperti "have" dan "a", telah diterapkan secara efektif. Pengujian membuktikan bahwa sistem dapat mengidentifikasi dan mencocokkan hasil pencarian.

Pengujian *backend* mencakup pengujian kinerja. Pengujian kinerja meliputi pengukuran waktu respons sistem untuk dapat merespons permintaan pengguna. Berikut pada tabel 1 merupakan dokumentasi hasil pengujian *respon time* dari fungsi *backend* yang ada pada website menggunakan aplikasi Postman.

TABEL 1
Hasil Pengujian API

Fungsi	Respons Time
Pencarian Berdasarkan Prediksi Bab Penyakit	8.12 detik
Pencarian Menggunakan NLP	4.97 detik
Pencarian Menggunakan <i>Stopwords</i> Dari Pustaka NLTK	15.83 detik
Pilih Kode ICD-10	2.99 detik
Pengiriman Dokter ID	1.77 detik
Pengiriman Nama Pasien Berdasarkan Dokter ID	1.75 detik
Pengiriman Nama Penyakit Berdasarkan Dokter ID Dan Nama Pasien	1.76 detik
Pengiriman Kode ICD-10 Berdasarkan Dokter ID, Nama Pasien, Dan Nama Penyakit	1.75 detik
Pencarian Kata Pada Bab Manual Koding	2.03 detik
Pencarian Kata Pada Bab Medis	1.69 detik
Pencarian Kata Pada Bab Administrasi	1.41 detik

Hasil pengujian pada tabel 1 menunjukkan bahwa rata-rata waktu respons seluruh fungsi berada di sekitar 4 detik. Namun, ada beberapa fungsi yang mencatatkan waktu respons di atas 5 detik, seperti fungsi pencarian menggunakan *stopwords* dari NLTK dengan waktu respons 15,83 detik, serta fungsi pencarian berdasarkan prediksi bab dengan waktu respons 8,12 detik. Selain kedua fungsi tersebut, fungsi-fungsi lainnya menunjukkan performa dengan rata-rata waktu respons 2,23 detik.

V. KESIMPULAN

Pengujian terhadap sistem pencarian berdasarkan prediksi bab penyakit menunjukkan bahwa model yang digunakan memiliki performa yang baik dengan *f1-score* rata-rata sebesar 84%. Sistem pencarian yang menggunakan metode *stopwords* juga memberikan hasil yang efektif. Namun, sistem pencarian berbasis NLP dengan model yang diunduh dari pustaka SpaCy memerlukan pengujian lebih lanjut untuk menilai relevansi hasil yang ditampilkan bagi pengguna. Selain itu, semua integrasi antara dokter dan staf rekam medis berjalan dengan baik melalui API yang dihosting di Cloud Run GCP. Pengujian terhadap API juga memberikan hasil yang memuaskan, dengan rata-rata waktu respons sebesar 4 detik. Namun, perlu diperhatikan bahwa fungsi pencarian menggunakan *stopwords* dari pustaka NLTK memerlukan optimasi lebih lanjut, karena memiliki waktu respon hingga 15 detik, sehingga evaluasi lebih lanjut terhadap waktu respon ini masih diperlukan. Secara keseluruhan, prototype website fullstack ini menunjukkan potensi untuk mempermudah pengembangan website lebih lanjut.

REFERENSI

[1] “BPJS Kesehatan.” Accessed: Aug. 17, 2024. [Online]. Available:

<https://sikapiuangmu.ojk.go.id/FrontEnd/CMS/Category/62#:~:text=BPJS%20Kesehatan%20adalah%20badan%20hukum,tentang%20Badan%20Penyelenggara%20Jaminan%20Sosial>.

[2] N. Muthia, A. Pratiwi, P. Studi, K. Masyarakat, and F. Kesehatan, “PERAN BPJS KESEHATAN SEBAGAI PENYELENGGARA PROGRAM JAMINAN KESEHATAN NASIONAL (JKN) DALAM SUBSISTEM PEMBIAYAAN KESEHATAN DI INDONESIA (The Role of BPJS Kesehatan as the Organizer of the National Health Insurance (JKN) Program in the Health Financing Subsystem in Indonesia).”

[3] “ICD-10.” Accessed: Aug. 17, 2024. [Online]. Available: <https://satusehat.kemkes.go.id/platform/docs/id/terminology/icd/icd-10/#:~:text=ICD%2D10%20adalah%20Klasifikasi%20Statistik,medis%20yang%20dikeluarkan%20oleh%20WHO>.

[4] ““TRANSPARANSI DAN KUALITAS: MENGURAI REGULASI TARIF LAYANAN KESEHATAN INDONESIA.”” Accessed: Aug. 17, 2024. [Online]. Available: [https://bkpsdm.jogjakota.go.id/detail/index/30061#:~:text=Tarif%20INA%2DCBG%20\(Indonesian%2D,pengeompokan%20diagnosis%20penyakit%20dan%20prosedur](https://bkpsdm.jogjakota.go.id/detail/index/30061#:~:text=Tarif%20INA%2DCBG%20(Indonesian%2D,pengeompokan%20diagnosis%20penyakit%20dan%20prosedur).

[5] “Dalam Era BPJS, Coder Rekam Medis dan Dokter Harus Pahami Benar ICD-10.” Accessed: Aug. 17, 2024. [Online]. Available: <https://fkkmk.ugm.ac.id/dalam-era-bpjs-coder-rekam-medis-dan-dokter-harus-pahami-benar-icd-10/>

[6] G. M. Fung, O. L. Mangasarian, and J. W. Shavlik, “Knowledge-Based Support Vector Machine Classifiers.”

[7] Q. Liu, C. Chen, Y. Zhang, and Z. Hu, “Feature selection for support vector machines with RBF kernel,” *Artif Intell Rev*, vol. 36, no. 2, pp. 99–115, Aug. 2011, doi: 10.1007/s10462-011-9205-2.

[8] S. Sarica and J. Luo, “Stopwords in technical language processing,” *PLoS One*, vol. 16, no. 8 August, Aug. 2021, doi: 10.1371/journal.pone.0254937.

[9] Nitin. Hardeniya, *NLTK essentials : build cool NLP and machine learning applications using NLTK and other Python libraries*. Packt Publishing, 2015.

[10] M. Wang and F. Hu, “The application of nltk library for python natural language processing in corpus research,” *Theory and Practice in Language Studies*, vol. 11, no. 9, pp. 1041–1049, Sep. 2021, doi: 10.17507/tpls.1109.09.

[11] N. C. P. Prakash, A. P. Narasimhaiah, J. B. Nagaraj, P. K. Pareek, N. B. Maruthikumar, and R. I. Manjunath, “Implementation of NLP based automatic text summarization using spacy,” *Int J Health Sci (Qassim)*, pp. 7508–7521, Aug. 2022, doi: 10.53730/ijhs.v6ns5.10574.

[12] J. Sawicki, M. Ganzha, and M. Paprzycki, “The State of the Art of Natural Language Processing—A Systematic Automated Review of NLP Literature

- Using NLP Techniques,” *Data Intell*, vol. 5, no. 3, pp. 707–749, Jun. 2023, doi: 10.1162/dint_a_00213.
- [13] K. Relan, *Building REST APIs with Flask*. Apress, 2019. doi: 10.1007/978-1-4842-5022-8.
- [14] Praveen Borra, “A Survey of Google Cloud Platform (GCP): Features, Services, and Applications,” *International Journal of Advanced Research in Science, Communication and Technology*, pp. 191–199, Jun. 2024, doi: 10.48175/ijarsct-18922.
- [15] “Apa itu Cloud Run.” Accessed: Aug. 17, 2024. [Online]. Available: <https://cloud.google.com/run/docs/overview/what-is-cloud-run?hl=id>

