

Implementasi Metode *Hybrid Genetic Algorithm* dan *Random Forest* untuk Deteksi Dini Tautan *Phishing*

Hafiz Alfariysi ^{1*}, Siti Sa'uda ², Fatoni ³, Edi Surya Negara ⁴

^{1,2,3,4} Program Studi Teknik Informatika, Fakultas Sains Teknologi, Universitas Bina Darma, Kota Palembang, Provinsi Sumatera Selatan, Indonesia.

ABSTRACT

The evolving nature of phishing attacks requires rapid and accurate detection. However, high-dimensional URL attributes can increase computational overhead during real-time processing. This study proposes a hybrid approach combining Genetic Algorithm (GA) and Random Forest (RF) for web-based phishing detection. Using 235,749 samples from the PhiUSIIL Phishing URL dataset, GA selects 20 features from the original 47 attributes, combining lexical URL characteristics with lightweight web content indicators. The dataset was partitioned using an 80:20 stratified split, yielding 188,599 training samples and 47,150 independent test samples. Evaluation on the test set yielded an accuracy of 99.66%, precision of 99.63%, recall of 99.78%, and F1-score of 99.70%, with 60 false negatives and 100 false positives. Stability testing using 10 different random seeds for data partitioning produced an average accuracy of 99.65%, with a standard deviation of 0.0241%. The model was integrated into a two-layer defense architecture using Flask and Vue.js, achieving a local server response time of less than 25 milliseconds.

ABSTRAK

Dinamika serangan *phishing* menuntut sistem deteksi dini yang cepat dan akurat. Namun, banyaknya atribut URL dapat meningkatkan beban komputasi dalam pemrosesan *real-time*. Penelitian ini mengusulkan pendekatan *hybrid* yang menggabungkan Genetic Algorithm (GA) dan Random Forest (RF) untuk mendeteksi *phishing* berbasis web. Dengan menggunakan 235.749 sampel dari dataset PhiUSIIL Phishing URL, GA menyeleksi 20 fitur dari 47 atribut awal yang mencakup karakteristik leksikal URL dan indikator konten web ringan. Data dibagi menggunakan *stratified split* dengan rasio 80:20, yang menghasilkan 188.599 data latih dan 47.150 data uji terisolasi. Evaluasi pada data uji menghasilkan akurasi 99,66%, presisi 99,63%, *recall* 99,78%, dan *F1-score* 99,70%, dengan 60 *False Negative* dan 100 *False Positive*. Pengujian stabilitas menggunakan 10 *random seed* pada pembagian data menghasilkan rata-rata akurasi 99,65% dengan standar deviasi 0,0241%. Model diintegrasikan ke dalam arsitektur pertahanan dua lapis (*two-layer defense*) menggunakan Flask dan Vue.js, dengan waktu respons peladen lokal kurang dari 25 milidetik. Arsitektur ini menggabungkan penyaringan URL melalui *blacklist* lokal dan klasifikasi berbasis pembelajaran mesin untuk menangani URL yang belum terdaftar. Integrasi tersebut memungkinkan sistem melakukan pemeriksaan URL secara otomatis serta mencatat riwayat pemindaian untuk mendukung pemantauan dan evaluasi hasil deteksi. Penelitian ini menghasilkan rancangan sistem deteksi *phishing* yang menggabungkan seleksi fitur, klasifikasi, dan implementasi aplikasi web dalam satu alur pemrosesan.

ARTICLE HISTORY

Received: 15 September 2026.
Revised: 17 September 2026.
Accepted: 10 October 2026.
Available Online: 30 October 2026.
Published: 1 December 2026.

KEYWORDS

Confusion Matrix; Flask; Genetic Algorithm; Phishing Detection; Random Forest.

KATA KUNCI

Confusion Matrix; Flask; Genetic Algorithm; Deteksi Phishing; Random Forest.

1. Pendahuluan

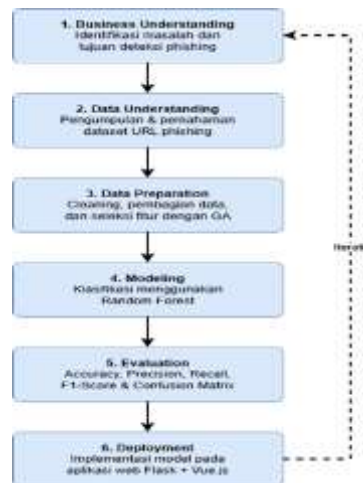
Phishing merupakan ancaman keamanan siber yang terus meningkat seiring dengan berkembangnya aktivitas digital. APWG mencatat 3,8 juta serangan global pada 2025, meningkat dari 3,76 juta pada 2024, sedangkan Laporan Tahunan Kamsiber 2024 mencatat 26,7 juta aktivitas PHISHINGSITE di Indonesia. Data tersebut menunjukkan perlunya mekanisme deteksi yang mampu beradaptasi terhadap pola serangan baru (APWG, 2026; Kominfo-CSIRT, 2025). Pendekatan konvensional berbasis blacklist dan aturan memiliki keterbatasan karena URL baru dapat mengarahkan pengguna ke situs berbahaya sebelum teridentifikasi dalam basis data ancaman (Alani & Tawfik, 2022; Mat Rani *et al.*, 2023). Oleh karena itu, diperlukan pendekatan yang tidak hanya mengandalkan daftar URL berbahaya yang telah diketahui, tetapi juga mampu mengidentifikasi karakteristik URL yang mengindikasikan potensi serangan. Pendekatan machine learning digunakan untuk mengklasifikasikan URL berdasarkan karakteristik leksikal, seperti panjang URL, informasi domain, dan penggunaan karakter khusus, sebelum halaman diakses (Al-Haimi *et al.*, 2022). Meskipun pendekatan ini dapat digunakan untuk mendeteksi pola URL yang mencurigakan, penggunaan terlalu banyak atribut dapat meningkatkan kompleksitas komputasi selama pemrosesan real-time. Oleh karena itu, seleksi fitur diperlukan untuk mengurangi atribut yang tidak relevan atau redundan sehingga model dapat menjadi lebih sederhana dan efisien tanpa mengabaikan ketepatan klasifikasi (Al-Jumah *et al.*, 2023; Gupta *et al.*, 2021). Proses ini juga dapat mengurangi noise dalam data dan membantu model berfokus pada atribut yang relevan untuk membedakan URL yang aman dan berbahaya. Selain itu, efektivitas seleksi fitur tidak hanya ditentukan oleh berkurangnya jumlah atribut, tetapi juga oleh kemampuan subset fitur yang terpilih dalam mempertahankan informasi penting untuk proses klasifikasi. Oleh karena itu, proses seleksi perlu mempertimbangkan relevansi setiap atribut terhadap tujuan deteksi agar pengurangan dimensi data tidak menghilangkan karakteristik penting yang membedakan URL phishing dan URL yang sah. Teknik penyamaran (obfuscation) yang digunakan oleh pelaku kejahatan siber dapat menghasilkan struktur URL yang kompleks dan meningkatkan tantangan dalam pemilihan atribut yang relevan. Kondisi ini berkaitan dengan permasalahan dimensionalitas fitur yang dapat memengaruhi efisiensi pemodelan (Ahammad *et al.*, 2023). Penggunaan seluruh atribut tanpa seleksi juga berpotensi memperpanjang waktu inferensi dan meningkatkan risiko overfitting, bergantung pada karakteristik dataset dan model yang digunakan (Rahman *et al.*, 2023). Oleh karena itu, teknik optimasi heuristik dapat digunakan untuk mencari kombinasi fitur yang memiliki daya pembeda tinggi dengan tetap mempertimbangkan kinerja klasifikasi (Subasi *et al.*, 2021). Pendekatan tersebut diharapkan dapat menghasilkan representasi data yang lebih ringkas sehingga proses pembelajaran dan penerapan model dapat berfokus pada atribut yang relevan.

Salah satu metode seleksi fitur yang dapat digunakan adalah Genetic Algorithm (GA). Algoritma ini mencari kombinasi atribut melalui proses evolusi untuk memperoleh subset fitur yang sesuai dengan fungsi objektif yang ditetapkan. Setelah seleksi fitur dilakukan, Random Forest (RF) digunakan sebagai algoritma klasifikasi berbasis ensemble learning untuk membedakan URL berdasarkan atribut terpilih. Penggunaan Random Forest dalam deteksi URL berbahaya telah dibahas dalam penelitian sebelumnya (Al-Sarem *et al.*, 2022; Wiratama & Karyawati, 2025). Kombinasi GA dan Random Forest dalam penelitian ini diarahkan untuk menyeleksi fitur dan melakukan klasifikasi menggunakan atribut struktural URL serta indikator konten web yang digunakan dalam dataset. Penggunaan kedua algoritma tersebut memungkinkan proses optimasi fitur dan evaluasi klasifikasi dilakukan secara terstruktur. Dengan demikian, pemilihan kombinasi fitur tidak hanya berorientasi pada penyederhanaan atribut, tetapi juga perlu mempertimbangkan kemampuan model dalam menghasilkan prediksi yang akurat dan konsisten. Selain pemilihan algoritma klasifikasi, efisiensi deteksi pada aplikasi web real-time juga bergantung pada arsitektur sistem. Eksekusi model machine learning untuk setiap permintaan URL berpotensi menambah beban komputasi peladen, terutama ketika URL yang sama diperiksa berulang kali. Mekanisme pertahanan bertingkat (multi-layer defense) yang menggabungkan penyaringan lokal dengan inferensi machine learning dapat digunakan untuk membedakan penanganan URL yang telah diketahui dan URL yang memerlukan pemeriksaan lebih lanjut (Khan *et al.*, 2024). Mekanisme Adaptive Blacklist Update juga dapat digunakan untuk memperbarui daftar ancaman berdasarkan hasil deteksi tanpa harus selalu melatih ulang model (Verma & Sharma, 2023). Integrasi mekanisme penyaringan awal dan klasifikasi berbasis model menjadi aspek penting dalam pengembangan sistem karena proses deteksi tidak hanya berkaitan dengan ketepatan klasifikasi, tetapi juga dengan alur pemeriksaan URL dan pengelolaan hasil pemindaian. Berdasarkan permasalahan tersebut, penelitian ini

mengusulkan pendekatan hybrid Genetic Algorithm dan Random Forest untuk deteksi dini tautan phishing. GA digunakan untuk menyeleksi fitur dari 47 atribut awal, sedangkan Random Forest digunakan untuk mengklasifikasikan URL berdasarkan fitur terpilih. Pendekatan ini diimplementasikan dalam aplikasi web dengan arsitektur pertahanan dua lapis, yaitu penyaringan awal menggunakan blacklist lokal dan pemeriksaan berbasis model terhadap URL yang belum terdaftar. Penelitian sebelumnya telah membahas seleksi fitur berbasis GA maupun pengembangan aplikasi deteksi phishing (Bu & Kim, 2022; Wulandari & Irawati, 2025). Penelitian ini mengintegrasikan seleksi fitur, klasifikasi, dan mekanisme penyaringan URL dalam satu aplikasi untuk mendukung deteksi ancaman secara real-time (Connolly & Atlam, 2025; Riad *et al.*, 2026). Aspek implementasi menjadi penting karena model deteksi dengan kinerja klasifikasi yang baik perlu didukung oleh mekanisme pemeriksaan yang dapat digunakan dalam alur kerja aplikasi. Oleh sebab itu, penelitian ini tidak hanya berfokus pada pengembangan model, tetapi juga pada integrasi model ke dalam sistem yang memungkinkan pengguna melakukan pemeriksaan URL melalui tahapan penyaringan dan klasifikasi. Evaluasi penelitian mencakup kinerja klasifikasi dan waktu respons aplikasi pada lingkungan pengujian yang digunakan.

2. Metodologi

Penelitian ini menggunakan desain eksperimen (experimental research) dengan pendekatan kuantitatif untuk menguji kinerja metode hybrid Genetic Algorithm (GA) dan Random Forest (RF) dalam mendeteksi tautan phishing. Kerangka kerja penelitian mengacu pada Cross-Industry Standard Process for Data Mining (CRISP-DM), yang terdiri atas enam tahapan terstruktur dan bersifat iteratif, yaitu Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, dan Deployment (Rianti *et al.*, 2023; Schröer *et al.*, 2021). Alur penelitian secara keseluruhan ditunjukkan pada Gambar 1.



Gambar 1. Alur CRISP-DM

2.1 Business Understanding

Tahap *Business Understanding* diawali dengan identifikasi keterbatasan mekanisme deteksi berbasis *blacklist*, khususnya dalam mengenali URL baru yang belum tercatat dalam basis data ancaman. Identifikasi ini menjadi dasar penetapan tujuan penelitian, yaitu mengembangkan sistem deteksi dini *phishing* berbasis web melalui kombinasi seleksi fitur menggunakan Genetic Algorithm (GA) dan klasifikasi menggunakan Random Forest (RF) (Martinez-Plumed, 2021). Penelitian ini berfokus pada pemanfaatan fitur URL dan atribut yang tersedia dalam dataset untuk membedakan tautan yang sah (*legitimate*) dan tautan *phishing*. Kinerja model dievaluasi menggunakan metrik *accuracy*, *precision*, *recall*, dan F1-score. Model kemudian diintegrasikan ke dalam aplikasi web dengan arsitektur pertahanan dua lapis (*two-layer defense*), yang terdiri atas penyaringan awal menggunakan *blacklist* lokal dan klasifikasi berbasis *machine learning*. Implementasi menggunakan Flask pada sisi peladen (*backend*) dan Vue.js pada sisi klien (*frontend*).

2.2 Data Understanding

Tahap *Data Understanding* dimulai dengan pengambilan PhiUSIIL Phishing URL Dataset dari repositori Kaggle. Dataset awal terdiri atas 235.792 sampel URL dan 56 atribut. Pemeriksaan awal data (*data profiling*) mengidentifikasi 43 baris anomali yang berkaitan dengan *missing values* dan kendala pemisahan data akibat format *delimiter*. Setelah proses pembersihan, jumlah sampel menjadi 235.749. Selanjutnya, lima atribut berbentuk teks atau identitas, yaitu FILENAME, URL, Domain, TLD, dan Title, dihapus dari proses pemodelan karena tidak digunakan sebagai masukan numerik dalam rancangan model. Tahap ini menghasilkan 50 fitur numerik dan satu variabel target biner, dengan label 0 untuk URL *legitimate* dan label 1 untuk URL *phishing*. Distribusi kelas pada dataset bersih ditunjukkan pada Tabel 1.

Tabel 1. Rekapitulasi Distribusi Kelas

Kategori	Representasi label	Jumlah sampel URL	Persentase
Total sampel bersih	–	235.749	100,00%
Aman (<i>legitimate</i>)	0	100.901	42,80%
Phishing	1	134.848	57,20%

Berdasarkan Tabel 1, kelas *legitimate* mencakup 42,80% dari keseluruhan sampel, sedangkan kelas *phishing* mencakup 57,20%. Distribusi tersebut menunjukkan adanya perbedaan proporsi antarkelas, tetapi keduanya tetap terwakili dalam jumlah sampel yang memadai untuk proses pembelajaran terawasi (*supervised learning*). Sebanyak 50 fitur numerik, termasuk URLLength, CharContinuationRate, dan IsHTTPS, selanjutnya dipersiapkan untuk proses pengolahan data dan pemodelan.

2.3 Data Preparation

Tahap *Data Preparation* mencakup pengolahan atribut, pencegahan data leakage, pembagian dataset, dan seleksi fitur menggunakan Genetic Algorithm (GA). Dari 50 fitur numerik yang diperoleh setelah pembersihan awal, tiga atribut, yaitu URLSimilarityIndex, DomainTitleMatchScore, dan URLTitleMatchScore, dihapus karena dinilai berpotensi menimbulkan target leakage. Penghapusan tersebut menyisakan 47 atribut yang digunakan sebagai kandidat fitur dalam proses seleksi. Alasan penghapusan ketiga atribut tersebut perlu didukung oleh pemeriksaan terhadap definisi atribut dan hubungannya dengan variabel target. Dataset kemudian dibagi menggunakan stratified sampling dengan rasio 80:20. Pembagian ini bertujuan mempertahankan proporsi kelas pada data latih dan data uji. Data uji diisolasi dari proses seleksi fitur dan pelatihan model agar dapat digunakan untuk mengevaluasi kinerja model pada data yang tidak dilibatkan dalam proses pembelajaran. Rincian pembagian dataset disajikan pada Tabel 2.

Tabel 2. Rincian Partisi Data.

Kategori partisi	Phishing (1)	Aman (0)	Total sampel	Persentase
Data latih (<i>training</i>)	107.878	80.721	188.599	80,00%
Data uji (<i>testing</i>)	26.970	20.180	47.150	20,00%
Total dataset	134.848	100.901	235.749	100,00%

Seleksi fitur dilakukan pada data latih menggunakan Genetic Algorithm (GA). Parameter GA yang digunakan meliputi *crossover rate* sebesar 0,8, *mutation rate* sebesar 0,03, ukuran populasi 25 individu, dan penghentian proses pada generasi ke-12. Proses seleksi menghasilkan 20 fitur terpilih dari 47 kandidat fitur awal. Fitur tersebut digunakan sebagai masukan bagi model Random Forest pada tahap pemodelan. Fitur terpilih mencakup atribut yang berkaitan dengan struktur URL dan karakteristik halaman web, seperti URLLength, NoOfDigitsInURL, HasSubmitButton, HasHiddenFields, dan NoOfEmptyRef. Pemilihan fitur bertujuan mempertahankan informasi yang relevan bagi proses klasifikasi sekaligus mengurangi jumlah atribut yang harus diproses oleh model. Interpretasi mengenai kemampuan masing-masing fitur dalam membedakan URL *legitimate* dan *phishing* perlu disesuaikan dengan hasil analisis fitur yang tersedia. Proses optimasi GA menggunakan fungsi kebugaran (*fitness function*) yang menggabungkan akurasi klasifikasi Random Forest melalui *5-fold cross-validation* dan penalti terhadap jumlah fitur. Fungsi tersebut dirumuskan sebagai berikut:

$$\text{Fitness} = \alpha \cdot \text{Accuracy} + (1 - \alpha) \cdot \left(1 - \frac{k}{N}\right) \quad (1)$$

dengan (k) merupakan jumlah fitur terpilih, (N) adalah jumlah kandidat fitur awal, yaitu 47, dan ($\alpha=0{,}9$) merupakan bobot yang diberikan pada akurasi klasifikasi. Komponen pertama mendorong pemilihan fitur yang menghasilkan akurasi tinggi, sedangkan komponen kedua memberikan nilai lebih pada subset fitur yang lebih ringkas. Kombinasi kedua komponen tersebut digunakan untuk menyeimbangkan kinerja klasifikasi dan kompleksitas fitur. Evaluasi *cross-validation* untuk fungsi kebugaran dilakukan pada data latih agar data uji tetap terpisah dari proses optimasi.

2.4 Modeling

Tahap Modeling bertujuan membangun model klasifikasi yang mempelajari pola dari fitur terpilih. Pemodelan menggunakan 188.599 sampel data latih dengan 20 atribut hasil seleksi Genetic Algorithm (GA). Algoritma klasifikasi yang digunakan adalah Random Forest Classifier, yaitu metode *ensemble learning* berbasis *bagging* yang menggabungkan prediksi dari sejumlah pohon keputusan (*decision trees*) untuk menentukan kelas URL. Model membedakan URL aman dengan label 0 dan URL phishing dengan label 1. Pelatihan model dilakukan menggunakan pustaka Scikit-learn dengan konfigurasi parameter yang disajikan pada Tabel 3. Konfigurasi tersebut digunakan sebagai pengaturan dasar (*baseline*) dalam eksperimen

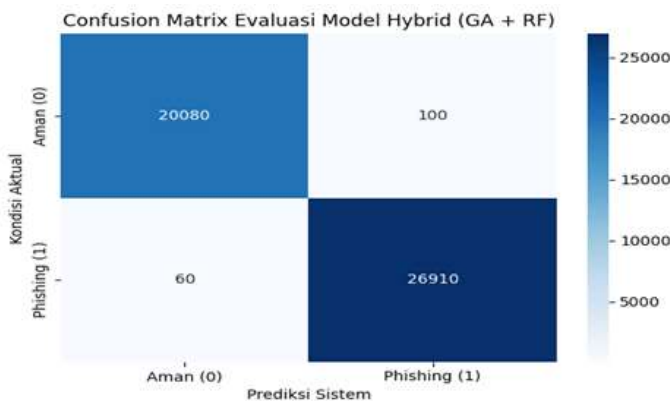
Tabel 3. Rincian Konfigurasi Random Forest

Parameter	Nilai	Deskripsi
n_estimators	100	Menggunakan 100 pohon keputusan dalam model.
criterion	gini	Menggunakan Gini Impurity sebagai kriteria pemisahan simpul.
max_depth	None	Tidak menetapkan batas kedalaman maksimum pohon secara eksplisit.
random_state	42	Menetapkan nilai awal generator bilangan acak untuk mendukung keterulangan eksperimen.
n_jobs	-1	Memanfaatkan prosesor yang tersedia untuk menjalankan komputasi secara paralel.

Dalam proses pelatihan, Random Forest membangun 100 pohon keputusan. Pada setiap simpul, algoritma mempertimbangkan subset fitur secara acak sesuai mekanisme internalnya untuk menentukan pemisahan kelas. Hasil prediksi dari pohon-pohon tersebut digabungkan untuk menghasilkan klasifikasi akhir. Penggunaan 20 fitur hasil seleksi GA membatasi masukan model pada atribut yang telah dipilih selama tahap optimasi, sedangkan mekanisme *bagging* memungkinkan Random Forest menggabungkan hasil dari beberapa pohon keputusan. Dengan pendekatan ini, model mempelajari pola klasifikasi berdasarkan fitur yang tersedia pada data latih untuk membedakan URL aman dan URL phishing. Model yang telah dilatih diserialisasi ke dalam berkas model_rf_phishing.pkl. Berkas tersebut digunakan kembali dalam tahap evaluasi dan integrasi aplikasi. Penyimpanan model memungkinkan proses inferensi dilakukan tanpa harus melatih ulang model setiap kali aplikasi menerima permintaan pemeriksaan URL.

2.5 Evaluation

Tahap *Evaluation* bertujuan mengukur kemampuan model Random Forest dalam mengklasifikasikan URL pada data yang tidak digunakan selama pelatihan. Evaluasi dilakukan menggunakan 47.150 sampel data uji yang telah dipisahkan pada tahap *Data Preparation*. Salah satu instrumen evaluasi yang digunakan adalah *confusion matrix*, yang merangkum hasil klasifikasi ke dalam empat kategori, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Melalui *confusion matrix*, hasil prediksi model dibandingkan dengan label aktual pada data uji untuk mengetahui jumlah klasifikasi yang benar maupun kesalahan prediksi. Analisis ini penting untuk menilai kemampuan model dalam membedakan URL *legitimate* dan URL *phishing*, terutama dalam mengidentifikasi tautan phishing yang berpotensi diklasifikasikan sebagai URL yang aman. Hasil evaluasi melalui *confusion matrix* ditampilkan pada Gambar 2.



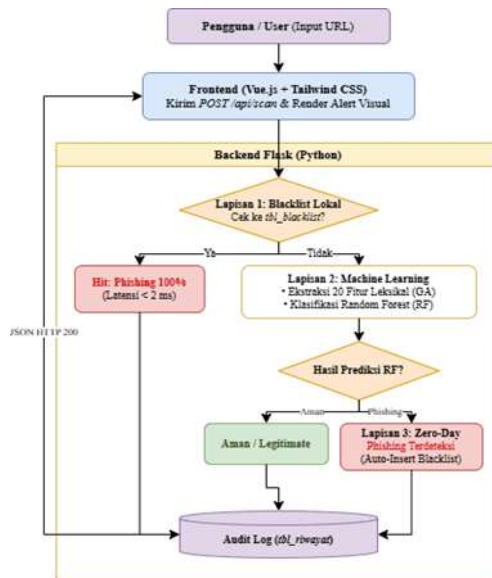
Gambar 2. Confusion Matrix Evaluasi Model

Berdasarkan hasil pengujian, model menghasilkan 60 kasus *False Negative* (FN), yaitu URL *phishing* yang diklasifikasikan sebagai URL aman. Jumlah tersebut setara dengan sekitar 0,22% dari seluruh sampel *phishing* pada data uji. Hasil ini berkaitan dengan nilai *recall* sebesar 99,78%, yang menunjukkan bahwa sebagian besar URL *phishing* dalam data uji berhasil teridentifikasi. Selain evaluasi pada data uji, pengujian stabilitas dilakukan melalui sepuluh percobaan menggunakan variasi *random seed* pada pembagian data. Pengujian menghasilkan rata-rata akurasi sebesar 99,65% dengan standar deviasi 0,0241%. Hasil tersebut menunjukkan variasi akurasi yang relatif kecil pada skenario pengujian yang dilakukan. Namun, pengujian stabilitas ini tidak dengan sendirinya membuktikan bahwa model sepenuhnya bebas dari *overfitting*. Kesimpulan mengenai generalisasi model perlu mempertimbangkan pemisahan data, proses seleksi fitur, dan prosedur validasi yang diterapkan.

2.6 Deployment

Tahap *Deployment* mengintegrasikan model ke dalam aplikasi web untuk mendukung pemeriksaan URL melalui arsitektur pertahanan dua lapis (*two-layer defense*). Implementasi memisahkan antarmuka pengguna pada sisi klien dan proses pengolahan permintaan pada sisi peladen. Antarmuka dibangun menggunakan HTML5, Tailwind CSS, dan Vue.js dengan pendekatan *single-page application* (SPA). Antarmuka digunakan untuk menampilkan status pemindaian, hasil deteksi, dan informasi fitur yang relevan. Penggunaan pendekatan SPA memungkinkan interaksi pengguna dan penyajian hasil pemeriksaan dilakukan melalui satu antarmuka aplikasi, sehingga proses pemeriksaan URL dapat disajikan secara terstruktur. Pada sisi peladen, Flask bertindak sebagai penyedia *REST API* yang menangani permintaan pemeriksaan URL, berinteraksi dengan basis data MySQL, dan menjalankan inferensi menggunakan model `model_rf_phishing.pkl`. Flask berperan sebagai penghubung antara permintaan yang dikirimkan dari sisi klien dan proses klasifikasi yang dilakukan pada sisi peladen. Pemisahan komponen tersebut memungkinkan antarmuka pengguna dan proses klasifikasi menjalankan fungsi yang berbeda, tetapi tetap terintegrasi dalam satu alur kerja sistem. Secara operasional, proses pemeriksaan dimulai ketika pengguna memasukkan URL melalui antarmuka aplikasi. Permintaan tersebut diteruskan ke peladen untuk diproses melalui API. Selanjutnya, sistem menjalankan proses inferensi menggunakan model Random Forest yang telah dilatih untuk menghasilkan prediksi terhadap URL yang diperiksa. Hasil prediksi kemudian diteruskan kembali ke antarmuka untuk ditampilkan kepada pengguna sebagai informasi hasil deteksi. Dengan alur tersebut, model pembelajaran mesin tidak hanya digunakan dalam tahap eksperimen, tetapi juga diintegrasikan ke dalam aplikasi agar dapat mendukung pemeriksaan URL secara langsung. Integrasi model ke dalam aplikasi web merupakan bagian penting dari penerapan hasil penelitian karena menghubungkan proses analisis berbasis pembelajaran mesin dengan kebutuhan pemeriksaan URL oleh pengguna. Arsitektur yang memisahkan sisi klien dan sisi peladen juga membantu memperjelas pembagian tanggung jawab setiap komponen, mulai dari penerimaan masukan, pemrosesan permintaan, pelaksanaan inferensi, hingga penyajian hasil klasifikasi. Sementara itu, basis data MySQL digunakan sebagai bagian dari komponen sistem sesuai dengan rancangan implementasi yang dikembangkan. Melalui tahap *Deployment*, penelitian ini diarahkan untuk menghasilkan sistem deteksi *phishing* yang menggabungkan model klasifikasi

dengan antarmuka aplikasi web. Implementasi tersebut menjadi bentuk penerapan model hasil pelatihan ke dalam lingkungan penggunaan yang lebih praktis. Alur integrasi antarkomponen, mulai dari antarmuka pengguna hingga proses inferensi pada sisi peladen, ditunjukkan pada Gambar 3.



Gambar 3. Alur Sistem Deteksi Phishing

Berdasarkan Gambar 3, proses pemeriksaan URL dimulai dengan penyaringan menggunakan daftar *blacklist* lokal yang disimpan pada tabel *tbl_blacklist*. Jika URL ditemukan dalam daftar tersebut, sistem langsung mengembalikan status *phishing* tanpa menjalankan model *machine learning*. Berdasarkan pengujian implementasi yang dilaporkan, jalur ini memiliki waktu respons di bawah 2 milidetik. Jika URL tidak ditemukan dalam *blacklist*, sistem melanjutkan pemeriksaan melalui model Random Forest dengan mengekstraksi fitur yang diperlukan untuk klasifikasi. URL baru yang teridentifikasi sebagai *phishing* selanjutnya ditambahkan ke tabel *tbl_blacklist* sebagai mekanisme pembaruan daftar ancaman secara adaptif. Riwayat pemeriksaan dicatat dalam tabel *tbl_riwayat* sebelum hasil dikirimkan ke antarmuka melalui respons JSON. Dengan mekanisme tersebut, sistem menggabungkan penyaringan URL yang telah diketahui dengan klasifikasi berbasis model untuk URL yang belum tercatat. Efektivitas pengurangan beban komputasi melalui mekanisme ini perlu didukung oleh hasil pengukuran waktu respons atau penggunaan sumber daya sistem. Pada proses pemeriksaan URL baru, sistem juga menggunakan pustaka *requests* dan *BeautifulSoup* untuk mengambil halaman melalui HTTP GET dan mengekstraksi atribut konten web, seperti *HasSubmitButton* dan *HasHiddenFields*. Pengambilan tersebut dilakukan secara terbatas untuk memperoleh informasi yang dibutuhkan oleh model, bukan untuk melakukan perayapan situs secara menyeluruh. Jika permintaan ke peladen target mengalami *timeout* atau ditolak, sistem memberikan nilai default 0 pada atribut konten yang gagal diperoleh dan melanjutkan pemeriksaan menggunakan fitur yang tersedia. Perlakuan terhadap nilai default tersebut perlu dipastikan konsisten dengan proses pembentukan data latih agar perbedaan kondisi ekstraksi tidak memengaruhi interpretasi hasil klasifikasi.

3. Hasil

Prapemrosesan *PhiUSIIL Phishing URL Dataset* dilakukan melalui penghapusan 43 baris data anomali yang berkaitan dengan *missing values* dan kendala pemisahan data, serta lima atribut metadata identitas. Proses tersebut menghasilkan 235.749 sampel bersih yang terdiri atas 134.848 sampel *phishing* (label 1) dan 100.901 sampel aman (*legitimate*, label 0). Dataset selanjutnya dipartisi menggunakan *stratified split* dengan rasio 80:20, yang menghasilkan 188.599 sampel data latih dan 47.150 sampel data uji. Untuk mengurangi potensi *data leakage*, tiga atribut, yaitu *URLSimilarityIndex*, *DomainTitleMatchScore*, dan

URLTitleMatchScore, dihapus dari kandidat fitur sebelum proses seleksi. Selanjutnya, Genetic Algorithm (GA) diterapkan pada data latih untuk menyeleksi atribut yang digunakan dalam pemodelan Random Forest (RF). Proses seleksi menghasilkan 20 fitur dari 47 kandidat fitur awal. Ke-20 fitur terpilih mencakup tiga kelompok informasi, yaitu karakteristik struktural URL, indikator konten web, dan fitur referensi. Karakteristik struktural URL meliputi *URLLength*, *IsDomainIP*, *CharContinuationRate*, *TLDLength*, *NoOfSubDomain*, *HasObfuscation*, *ObfuscationRatio*, *NoOfDigitsInURL*, *NoOfQMarkInURL*, *SpacialCharRatioInURL*, dan *IsHTTPS*. Indikator konten web meliputi *HasTitle*, *IsResponsive*, *NoOfPopup*, *HasExternalFormSubmit*, *HasSubmitButton*, *HasHiddenFields*, *Pay*, dan *HasCopyrightInfo*. Sementara itu, *NoOfEmptyRef* digunakan sebagai fitur referensi. Daftar fitur tersebut menjadi masukan bagi model Random Forest pada tahap pelatihan dan evaluasi.

```

# 1. Bersihkan data awal / Identifikasi yang bukan fitur otomatis
kolom_identities = ['URL', 'Domain', 'TLD', 'Title']
df_clean = df_raw.drop(columns=kolom_identities, errors='ignore').dropna()

# 2. Penuhi nilai label (memerikan nilai 0 untuk URL aman)
if 'label' in df_clean.columns:
    y = df_clean['label'].astype(str).str[0].astype(int)
    X = df_clean.drop(columns=['label'])
else:
    X = df_clean.iloc[:, :-1]
    y = df_clean.iloc[:, -1].astype(str).str[0].astype(int)

# Pastikan semua fitur X bertipe float/int numerik
X = X.apply(pd.to_numeric, errors='coerce').dropna(inplace=True)

# Tampilkan label sebagai nilai yang digunakan dalam data
df_distribusi = pd.DataFrame()
"Jumlah sampel": y.value_counts().values,
"Persentase (%)": (y.value_counts(normalize=True) * 100).round(2).values
], index=['Phishing (%)', 'Aman / legitimate (%)'])

print("==== REKAPITULASI PENGOLAHAN KLAS DATASET (RAB 1) ====")
print(df_distribusi.to_markdown())
print(f"Jumlah fitur Numerik Bersih (X): {X.shape[1]} atribut lokal.")

===== REKAPITULASI PENGOLAHAN KLAS DATASET (RAB 1) =====

```

	Jumlah Sampel	Persentase (%)
Phishing (1)	334888	55.2
Aman / legitimate (0)	309001	49.8

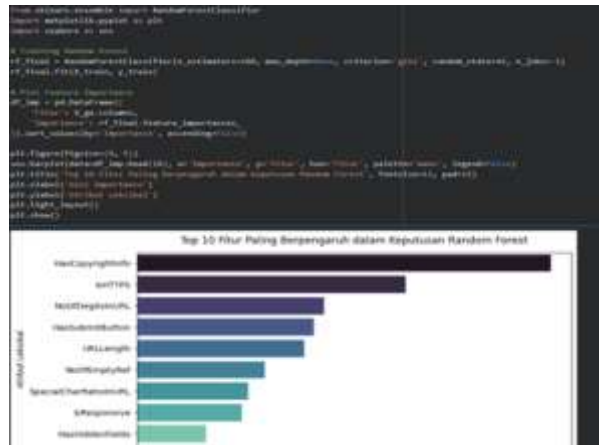
Ditambah fitur Numerik Bersih (X): 58 atribut lokal.

Gambar 4. Hasil Prapemrosesan Data dan Seleksi Fitur

Gambar 4 menunjukkan tahapan pengolahan dataset dan seleksi fitur menggunakan GA. Hasil tahap ini berupa dataset bersih dan subset 20 fitur yang digunakan dalam pemodelan. Pengurangan jumlah fitur menyederhanakan representasi masukan model; dampaknya terhadap waktu komputasi perlu ditentukan berdasarkan pengukuran kinerja yang dilakukan.

3.1 Hasil Pelatihan Model

Pelatihan model dilakukan menggunakan 188.599 sampel data latih yang memuat 20 fitur hasil seleksi GA. Sebanyak 47.150 sampel data uji dipisahkan untuk digunakan dalam evaluasi independen. Pemisahan tersebut dimaksudkan agar data uji tidak digunakan dalam proses pelatihan model. Algoritma klasifikasi yang digunakan adalah Random Forest Classifier dengan konfigurasi dasar (*baseline*), yaitu 100 pohon keputusan ($n_estimators = 100$), kriteria pemisahan *Gini Impurity* (*criterion = gini*), kedalaman pohon tanpa batas maksimum yang ditetapkan secara eksplisit ($max_depth = None$), dan nilai *random state* sebesar 42 untuk mendukung keterulangan eksperimen. Random Forest menggunakan mekanisme *Bootstrap Aggregating* (*bagging*) dengan membangun sejumlah pohon keputusan dan menggabungkan hasil prediksinya. Dalam penelitian ini, 100 pohon keputusan digunakan untuk mempelajari pola pada 20 fitur terpilih dan membedakan URL aman (label 0) dari URL *phishing* (label 1). Setelah proses pelatihan selesai, model disimpan dalam berkas *model_rf_phishing.pkl* untuk digunakan pada tahap evaluasi dan implementasi aplikasi web.

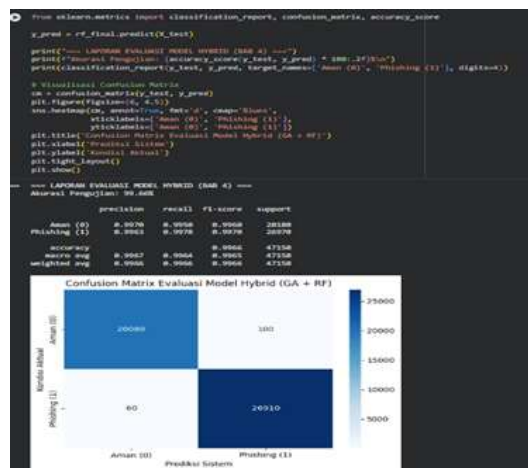


Gambar 5. Hasil Pelatihan Model Random Forest

Gambar 5 menampilkan hasil pelatihan model Random Forest, termasuk urutan fitur berdasarkan tingkat kepentingannya (*feature importance*). Visualisasi tersebut memberikan gambaran mengenai kontribusi relatif setiap fitur dalam proses klasifikasi model. Namun, tingkat kepentingan fitur tidak dengan sendirinya menunjukkan hubungan sebab-akibat maupun signifikansi statistik suatu fitur terhadap kelas *phishing*.

3.2 Hasil Evaluasi Model

Evaluasi dilakukan terhadap 47.150 sampel data uji yang tidak digunakan dalam pelatihan model. Pengujian ini bertujuan mengukur kemampuan Random Forest dalam mengklasifikasikan URL berdasarkan 20 fitur hasil seleksi GA. Hasil evaluasi dirangkum menggunakan *confusion matrix*, yang terdiri atas *True Negative* (TN), *True Positive* (TP), *False Positive* (FP), dan *False Negative* (FN). Hasil pengujian mencatat 20.080 sampel *True Negative*, 26.910 sampel *True Positive*, 100 sampel *False Positive*, dan 60 sampel *False Negative*. Berdasarkan hasil tersebut, model memperoleh akurasi sebesar 99,66%, *precision* sebesar 99,63%, *recall* sebesar 99,78%, dan F1-score sebesar 99,70% pada kelas *phishing*.



Gambar 6. Hasil Evaluasi Model

Gambar 6 menyajikan hasil evaluasi model, termasuk *confusion matrix* dan metrik klasifikasi. Dari total 134.848 sampel *phishing* pada dataset bersih, sebanyak 26.970 sampel masuk ke dalam data uji, dengan 26.910 sampel teridentifikasi dengan benar dan 60 sampel diklasifikasikan sebagai URL aman. Dengan demikian, jumlah *False Negative* pada data uji adalah sekitar 0,22% dari seluruh sampel *phishing* dalam data uji. Nilai *recall* sebesar 99,78% menunjukkan bahwa model berhasil mengidentifikasi sebagian besar sampel *phishing* yang diuji. Nilai *precision* sebesar 99,63% menunjukkan proporsi prediksi *phishing* yang

benar di antara seluruh prediksi positif model, sedangkan F1-score sebesar 99,70% merangkum keseimbangan antara *precision* dan *recall*. Secara keseluruhan, hasil pengujian menunjukkan kinerja klasifikasi yang tinggi pada dataset uji. Interpretasi tersebut berlaku pada data dan prosedur pengujian yang digunakan; generalisasi terhadap URL di luar dataset tetap memerlukan validasi tambahan.

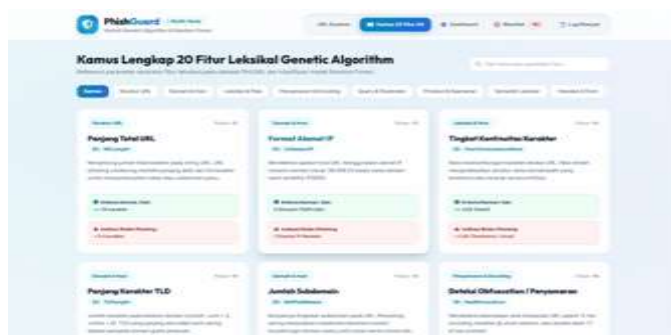
3.3 Hasil Implementasi Antarmuka Web

Model deteksi *phishing* diintegrasikan ke dalam aplikasi web PhishGuard dengan arsitektur pertahanan dua lapis (*two-layer defense*). Implementasi menggunakan Flask pada sisi peladen (*backend*) dan Vue.js pada sisi klien (*frontend*), dengan basis data MySQL untuk mendukung penyimpanan *blacklist* dan riwayat pemeriksaan. Aplikasi menyediakan antarmuka untuk memasukkan URL, menjalankan pemeriksaan, dan menampilkan hasil klasifikasi berupa status *Legitimate* atau *Phishing* serta indikator risiko (*Risk Meter*), sesuai dengan fitur yang tersedia pada implementasi.



Gambar 7. Tampilan Halaman Utama Pemindai Deteksi Dini

Gambar 7 memperlihatkan halaman utama PhishGuard yang digunakan untuk memasukkan URL dan menampilkan hasil pemeriksaan. Antarmuka ini menjadi titik interaksi pengguna dengan sistem deteksi. Pengguna dapat mengirimkan URL untuk diperiksa, kemudian melihat status keamanan yang dihasilkan oleh alur pemeriksaan aplikasi. Halaman *Dashboard* menyediakan ringkasan statistik aktivitas sistem, yang mencakup total pemindaian URL, jumlah URL *phishing* yang terdeteksi, jumlah URL aman, dan jumlah URL *phishing* baru yang tercatat dalam *blacklist*. Antarmuka ini juga menampilkan fungsi *Adaptive Blacklist Update Idle Engine* untuk simulasi pembaruan *blacklist* tanpa pelatihan ulang model (*retraining*), sesuai dengan fungsi yang dijelaskan dalam implementasi.



Gambar 8. Tampilan Halaman Kamus 20 Fitur

Halaman kamus fitur menyajikan penjelasan mengenai 20 fitur hasil seleksi GA dalam bentuk kartu (*card-based design*). Fitur pencarian (*search bar*) dan penyaringan kategori membantu pengguna menelusuri informasi fitur. Setiap kartu memuat deskripsi fitur dan informasi pendukung untuk menjelaskan karakteristik yang digunakan dalam proses klasifikasi. Penyajian ini membantu pengguna memahami masukan model, tetapi belum dengan sendirinya merupakan evaluasi formal terhadap keterjelasan keputusan (*explainable AI*).



Gambar 11. Tampilan Halaman Log Riwayat

Gambar 11 menunjukkan halaman log riwayat yang menampilkan catatan pemeriksaan URL. Pencatatan ini mendukung penelusuran hasil deteksi dan pemeriksaan aktivitas aplikasi. Rentang *Confidence Score* sebesar 84%–100% yang terlihat pada antarmuka perlu dijelaskan berdasarkan definisi dan cara perhitungannya di dalam sistem, agar tidak disamakan secara otomatis dengan probabilitas bahwa suatu URL benar-benar berbahaya.

4. Pembahasan

Arsitektur *two-layer defense* yang diusulkan menggabungkan penyaringan URL melalui *blacklist* lokal dan klasifikasi menggunakan *machine learning*. Pada lapisan pertama, URL yang telah tercatat dalam tabel *tbl_blacklist* dapat langsung dikenali tanpa menjalankan model Random Forest. Mekanisme ini dirancang untuk mengurangi kebutuhan inferensi berulang terhadap URL yang telah diketahui. Waktu respons di bawah 2 milidetik yang dilaporkan pada jalur *blacklist* menunjukkan respons yang cepat pada kondisi pengujian yang digunakan. Namun, besarnya pengurangan beban komputasi secara keseluruhan perlu dikonfirmasi melalui pengukuran perbandingan antara jalur *blacklist* dan jalur klasifikasi. Pada lapisan kedua, Genetic Algorithm (GA) digunakan untuk menyeleksi fitur dari 47 kandidat menjadi 20 fitur terpilih, atau mengurangi jumlah fitur sebesar sekitar 57,45%. Pengurangan dimensi ini menyederhanakan masukan yang digunakan oleh Random Forest (RF). Meskipun demikian, dampak reduksi fitur terhadap waktu komputasi dan risiko *overfitting* perlu dinilai berdasarkan hasil pengujian yang relevan. Sementara itu, pencegahan *target leakage* dilakukan melalui identifikasi dan penghapusan tiga atribut yang dinilai berpotensi menimbulkan kebocoran informasi target. Dengan demikian, penghapusan atribut yang berpotensi mengalami *leakage* dan seleksi fitur menggunakan GA merupakan dua langkah yang memiliki tujuan berbeda. Berdasarkan evaluasi pada 47.150 sampel data uji, model memperoleh akurasi sebesar 99,66%, *recall* sebesar 99,78%, dan *precision* sebesar 99,63%. Nilai tersebut menunjukkan bahwa model mampu mengklasifikasikan sebagian besar sampel pada data uji dengan tingkat kesalahan yang rendah. Nilai *recall* yang tinggi juga menunjukkan bahwa sebagian besar sampel *phishing* berhasil diidentifikasi. Hasil perbandingan dengan penelitian terdahulu disajikan pada Tabel 4.

Tabel 4. Perbandingan Kinerja Model dengan Penelitian Terdahulu

Peneliti/sumber	Algoritma klasifikasi	Pendekatan seleksi fitur	Akurasi
Sanjaya (2025)	Random Forest standar	Tanpa seleksi fitur; menggunakan seluruh fitur leksikal	97,40%
Tampinongkol <i>et al.</i> (2024)	SVM dan Random Forest	Tanpa reduksi fitur	95,80%
Chen Chong <i>et al.</i> (2025)	Random Forest + GA	GA difokuskan pada <i>hyperparameter tuning</i>	99,00%
Penelitian ini	Random Forest + GA	GA untuk seleksi fitur; reduksi menjadi 20 fitur	99,66%

Berdasarkan Tabel 4, penelitian ini memperoleh nilai akurasi tertinggi di antara nilai yang dilaporkan, yaitu 99,66%. Nilai tersebut lebih tinggi daripada akurasi Sanjaya (2025) sebesar 97,40%, Tampinongkol *et al.* (2024) sebesar 95,80%, dan Chen Chong *et al.* (2025) sebesar 99,00%. Perbedaan pendekatan yang ditampilkan dalam tabel terletak pada penggunaan GA untuk seleksi fitur dalam penelitian ini, sedangkan penelitian Chen Chong *et al.* (2025) dilaporkan menggunakan GA untuk *hyperparameter tuning*. Pada penelitian ini, GA digunakan untuk memilih subset fitur yang kemudian menjadi masukan bagi Random Forest. Pendekatan tersebut bertujuan mengurangi jumlah atribut sambil mempertahankan informasi yang relevan untuk klasifikasi URL. Fitur terpilih mencakup karakteristik struktural URL dan indikator konten web. Namun, kontribusi setiap fitur terhadap peningkatan akurasi perlu ditafsirkan berdasarkan hasil analisis fitur yang tersedia. Selain itu, perbandingan akurasi pada Tabel 4 perlu dibaca secara hati-hati karena perbedaan dataset, komposisi fitur, pembagian data, dan prosedur evaluasi dapat memengaruhi hasil masing-masing penelitian. Selain evaluasi menggunakan satu pembagian data uji, stabilitas model diperiksa melalui sepuluh pengujian dengan variasi *random seed* pada proses pembagian dataset. Konfigurasi Random Forest dipertahankan tetap, sedangkan partisi data diubah untuk mengamati variasi hasil akurasi. Rincian pengujian disajikan pada Tabel 5.

Tabel 5. Hasil Pengujian Stabilitas Model Sebanyak 10 Kali

Pengujian	<i>Random seed</i>	Akurasi
Uji #1	9	99,69%
Uji #2	18	99,66%
Uji #3	27	99,64%
Uji #4	36	99,62%
Uji #5	45	99,63%
Uji #6	54	99,64%
Uji #7	63	99,66%
Uji #8	72	99,69%
Uji #9	81	99,64%
Uji #10	90	99,67%
Rata-rata	–	99,65%
Standar deviasi	–	0,0241 poin persentase

Berdasarkan Tabel 5, akurasi pada sepuluh pengujian berada pada rentang 99,62% hingga 99,69%. Rata-rata akurasi sebesar 99,654% dibulatkan menjadi 99,65%, dengan standar deviasi sekitar 0,0241 poin persentase. Rentang dan standar deviasi tersebut menunjukkan bahwa variasi akurasi relatif kecil pada skenario pengujian yang dilakukan. Pengujian stabilitas ini menggunakan variasi *random_state* pada pembagian data secara *stratified split* dengan rasio 80:20, sedangkan konfigurasi inialisasi Random Forest dipertahankan tetap. Dengan demikian, hasil pengujian menggambarkan konsistensi akurasi ketika model dievaluasi menggunakan variasi partisi sampel. Akan tetapi, konsistensi tersebut tidak secara langsung membuktikan bahwa model sepenuhnya bebas dari *overfitting*. Penilaian terhadap *overfitting* tetap perlu mempertimbangkan perbandingan kinerja data latih dan data uji, prosedur validasi, serta pemisahan data yang benar selama proses seleksi fitur dan pemodelan.

5. Kesimpulan

Penelitian ini mengembangkan dan mengevaluasi pendekatan *hybrid* Genetic Algorithm (GA) dan Random Forest (RF) untuk mendeteksi tautan *phishing* melalui aplikasi web. Penerapan GA pada data latih menghasilkan reduksi dari 47 atribut kandidat menjadi 20 fitur terpilih yang digunakan sebagai masukan bagi model Random Forest. Prosedur pemisahan data dan seleksi fitur dirancang untuk mengurangi risiko *data leakage* serta menjaga independensi evaluasi. Model yang dihasilkan diintegrasikan ke dalam aplikasi web menggunakan Flask pada sisi *backend* dan Vue.js pada sisi *frontend*, dengan arsitektur pertahanan dua lapis (*two-layer defense*) yang menggabungkan penyaringan *blacklist* lokal dan klasifikasi berbasis *machine learning*. Evaluasi terhadap 47.150 sampel data uji menghasilkan

akurasi sebesar 99,66%, presisi 99,63%, *recall* 99,78%, dan F1-score 99,70%, dengan 60 kasus *False Negative* dan 100 kasus *False Positive*. Pengujian stabilitas melalui sepuluh variasi *random seed* menghasilkan rata-rata akurasi sebesar 99,65% dengan standar deviasi 0,0241 poin persentase. Hasil tersebut menunjukkan kinerja klasifikasi yang tinggi dan variasi akurasi yang relatif kecil pada skenario pengujian yang dilakukan. Pada implementasi aplikasi, penyaringan URL yang telah terdaftar dalam *tbl_blacklist* dilaporkan memiliki waktu respons di bawah 2 milidetik, sedangkan rata-rata waktu respons peladen pada pengujian lokal dilaporkan kurang dari 25 milidetik. Hasil ini menunjukkan potensi integrasi seleksi fitur, klasifikasi, dan penyaringan *blacklist* untuk mendukung pemeriksaan URL secara efisien pada lingkungan pengujian. Penelitian ini memiliki keterbatasan karena evaluasi model menggunakan PhiUSIIL Phishing URL Dataset, sedangkan pengujian aplikasi dilakukan pada lingkungan lokal berbasis Flask dan Vue.js. Oleh karena itu, hasil yang diperoleh belum secara langsung menggambarkan kinerja sistem pada seluruh kondisi jaringan, variasi perangkat, atau pola serangan *phishing* yang terus berkembang. Selain itu, kegagalan pengambilan halaman melalui HTTP request dapat membatasi ekstraksi atribut konten web tertentu. Penelitian selanjutnya disarankan mengembangkan mekanisme alternatif berbasis analisis leksikal URL ketika ekstraksi konten gagal, serta mengevaluasi penggunaan *headless browser*, seperti Puppeteer, untuk menangani halaman yang memerlukan eksekusi JavaScript. Pengujian lanjutan juga perlu dilakukan pada dataset eksternal dan lingkungan operasional yang lebih beragam untuk menilai generalisasi model, keandalan ekstraksi fitur, dan konsistensi waktu respons sistem.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada Universitas Bina Darma atas dukungan dan kesempatan yang diberikan dalam pelaksanaan penelitian ini. Ucapan terima kasih juga disampaikan kepada dosen pembimbing atas arahan, bimbingan, masukan, dan dukungan sejak tahap awal penyusunan skripsi hingga penyelesaian penelitian dan artikel jurnal ini. Penulis turut menyampaikan apresiasi kepada rekan-rekan yang telah memberikan bantuan, dukungan, dan motivasi selama proses penelitian dan penyusunan artikel. Semoga segala bantuan dan dukungan yang diberikan dapat memberikan manfaat bagi semua pihak

Referensi

- Ahammad, I., Rahman, M. S., & Islam, M. R. (2023). URL obfuscation techniques and feature selection in machine learning-based phishing detection: A review. *Journal of Information Security and Applications*, 75, Article 103480.
- Alani, M. M., & Tawfik, A. I. (2022). Rule-based and blacklist approaches in phishing detection: Limitations and future directions. *Cybersecurity*, 5(1), 12–25.
- Al-Haimi, B., Sjarif, N. N. A., & Yuhaniz, A. Z. (2022). Lexical features-based detection of phishing URLs using machine learning techniques: A systematic review. *IEEE Access*, 10, 32145–32160.
- Al-Jumah, A., Al-Khulaifi, A., & Bamasag, O. (2023). Lightweight lexical feature selection for real-time phishing URL detection using Random Forest. *Computers & Security*, 128, Article 103150.
- Al-Sarem, M., Saeed, F., Boulila, W., & Al-Mohaifer, A. (2022). Feature selection and ensemble learning algorithms for detecting phishing websites. *IEEE Access*, 10, 56342–56357.
- Anti-Phishing Working Group. (2026). *Phishing activity trends report: 4th quarter 2025*.
- Bu, S. J., & Kim, S. B. (2022). Genetic Algorithm-based feature selection for improving classification performance in phishing URL detection. *Expert Systems with Applications*, 201, Article 117100.

- Chen Chong, L., Wei, T. S., & Guan, C. Y. (2025). Genetic Algorithm-optimized Random Forest for enhanced phishing URL classification. *Journal of Cybersecurity and Data Protection*, 14(1), 45–58.
- Connolly, L., & Atlam, H. F. (2025). Hybrid cybersecurity defense systems: Integrating dynamic caching and machine learning for web security. *Journal of Information Security and Applications*, 81, Article 103720.
- Gupta, B. B., Yadav, K., & Joshi, R. C. (2021). Efficient phishing URL detection using minimal lexical feature sets and Random Forest. *International Journal of Information Security*, 20(4), 511–525.
- Khan, F., Ahmad, A., & Shah, S. A. (2024). Ultra-low latency web security architecture using multi-layer caching and machine learning classifiers. *IEEE Access*, 12, 15420–15432.
- Martínez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernández-Orallo, J., Kull, M., Lachiche, N., Ramírez-Quintana, M. J., & Flach, P. A. (2021). CRISP-DM twenty years later: From data mining processes to data science trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 33(8), 3048–3061. <https://doi.org/10.1109/TKDE.2019.2962680>
- Mat Rani, N. S., Ahmad, A., & Ismail, Z. (2023). Informative feature selection techniques for phishing URL classification using ensemble methods. *Information Security Journal: A Global Perspective*, 32(5), 310–324.
- Rahman, A., Das, S., & Hossain, M. A. (2023). Mitigating overfitting and dimensionality curse in phishing URL detection using evolutionary algorithms. *Computers & Security*, 132, Article 103350.
- Riad, A. M., El-Bakry, H. M., & Abdel-Fattah, M. A. (2026). Real-time web application defense using dual-layer filtering and machine learning. *Computers & Electrical Engineering*, 121, Article 109210.
- Rianti, E., Suhada, A., & Mulyani, S. (2023). Penerapan kerangka kerja CRISP-DM dalam pemodelan data mining untuk deteksi ancaman siber. *Jurnal Sistem Informasi dan Teknologi Informasi*, 12(1), 45–56.
- Sanjaya, R. (2025). Analisis performa algoritma Random Forest standar pada klasifikasi URL phishing tanpa reduksi dimensi. *Jurnal Teknologi Informasi dan Ilmu Komputer*, 12(1), 89–97.
- Schröer, C., Kruse, F., & Marx Gómez, J. (2021). A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, 181, 526–534.
- Subasi, A., Baloglu, E., & Karabatak, M. (2021). Phishing website detection using heuristic feature selection and ensemble learning algorithms. *Applied Intelligence*, 51(6), 3820–3836.
- Tampinongkol, M., Rindengan, Y., & Karouw, S. (2024). Komparasi metode SVM dan Random Forest untuk deteksi serangan phishing berbasis fitur URL mentah. *Jurnal Teknik Informatika*, 19(2), 145–154.
- Verma, S., & Sharma, R. (2023). Adaptive local blacklist updates for zero-day phishing protection without dynamic model retraining. *Cybersecurity*, 6(1), Article 18.
- Wiratama, A., & Karyawati, A. E. (2025). Deteksi URL berbahaya berbasis fitur struktural menggunakan algoritma Random Forest. *Jurnal Rekayasa Sistem dan Teknologi Informasi*, 9(2), 112–119.
- Wulandari, R., & Irawati, I. (2025). Pengembangan aplikasi web deteksi phishing secara otomatis dengan ekstraksi fitur URL interaktif. *Jurnal Edukasi dan Penelitian Informatika*, 11(1), 78–86.