RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING

SKRIPSI

DISUSUN OLEH

TRIFAHMI RIVALDO

NPM. 2209020255

PROGRAM STUDI TEKNOLOGI INFORMASI

FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI

UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA

MEDAN

2026

RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer (S.Kom) dalam Program Studi Tenologi Infromasi. pada Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara

TRIFAHMI RIVALDO
NPM. 2209020255

PROGRAM STUDI TEKNOLOGI INFROMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
MEDAN

LEMBAR PENGESAHAN

Judul Skripsi	:	RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING
Nama Mahasiswa	:	TRIFAHMI RIVALDO
NPM	:	2209020255
Program Studi	:	TEKNOLOGI INFORMASI

Menyetujui
Komisi Pembimbing

(Hevlie Winda Nazry S, S.Pd, M.Si)
NIDN. 0129079301

Ketua Program Studi

(Fatma Sari Hutagalung, S.Kom, M.Kom)
NIDN. 0117019301

Dekan

(Dr. Al-Khowarizmi, S.Kom., M.Kom.)
NIDN. 0127099201

PERNYATAAN ORISINALITAS

RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING

SKRIPSI

Saya menyatakan bahwa karya tulis ini adalah hasil karya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya.

Medan, April 2026

Yang membuat pernyataan

Trifahmi Rivaldo

NPM. 2209020255

PERNYATAAN PERSETUJUAN PUBLIKASI

KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Muhammadiyah Sumatera Utara, saya bertanda tangan dibawah ini:

Nama	: Trifahmi Rivaldo
NPM	: 2209020255
Program Studi	: Teknologi Informasi
Karya Ilmiah	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Muhammadiyah Sumatera Utara Hak Bedas Royalti Non-Eksekutif *(Non-Exclusive Royalty free Right)* atas penelitian skripsi saya yang berjudul:

RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksekutif ini, Universitas Muhammadiyah Sumatera Utara berhak menyimpan, mengalih media, memformat, mengelola dalam bentuk database, merawat dan mempublikasikan Skripsi saya ini tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis dan sebagai pemegang dan atau sebagai pemilik hak cipta.

Demikian pernyataan ini dibuat dengan sebenarnya.

Medan, April 2026

Yang membuat pernyataan

Trifahmi Rivaldo

NPM. 2209020255

RIWAYAT HIDUP

DATA PRIBADI

Nama Lengkap	: Trifahmi Rivaldo
Tempat dan Tanggal Lahir	: Medan, 28 September 2004
Alamat Rumah	: Jl. Turi Gg. Jasa No.46A
Telepon/Faks/HP	: 087738568711
E-mail	: tfrvld@gmail.com
Instansi Tempat Kerja	: -
Alamat Kantor	: -

DATA PENDIDIKAN

SD	: SDN 101789 Marindal	TAMAT: 2016
SMP	: SMPN 4 Medan	TAMAT: 2019
SMA	: SMAN 5 Medan	TAMAT: 2022

KATA PENGANTAR

Pendahuluan

Penulis tentunya berterima kasih kepada berbagai pihak dalam dukungan serta doa dalam penyelesaian skripsi. Penulis juga mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Akrim, M.Pd, Rektor Universitas Muhammadiyah Sumatera Utara (UMSU)
2. Bapak Assoc. Prof. Dr. Al - Khowarizmi, S.Kom., M.Kom Dekan Fakultas Ilmu Komputer dan Teknologi Informasi (FIKTI) UMSU.
3. Ibu Dr. Firahmi Rizky, S.Kom.,M.Kom, M.Kom. Wakil Dekan I Fakultas Ilmu Komputer dan Teknologi Informasi (FIKTI) UMSU.
4. Bapak Mhd. Basri, S.Si, M.Kom., M.Kom Wakil Dekan II Fakultas Ilmu Komputer dan Teknologi Informasi (FIKTI) UMSU.
5. Ibu Fatma Sari Hutagalung, S.Kom, M.Kom Ketua Program Studi Teknologi Informasi
6. Bapak Okvi Nugroho, S.Kom, M.Kom Sekretaris Program Studi Teknologi Informasi
7. Pembimbing Hevlie Winda Nazry S, S.Pd, M.Si
8. Orang tua yang mendukung dan mendoakan.
9. Teman-teman yang membantu.
10. Semua pihak yang terlibat langsung ataupun tidak langsung yang tidak dapat penulis ucapkan satu-persatu yang telah membantu penyelesaian skripsi ini.

RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING

ABSTRAK

Maraknya kejahatan siber melalui *Short Message Service* (SMS) atau *smishing* menimbulkan kerugian finansial dan pencurian data pribadi di masyarakat. Sistem penyaringan konvensional yang berbasis pencocokan kata kunci (*keyword matching*) sering kali gagal mendeteksi taktik penyamaran teks yang digunakan oleh pelaku kejahatan. Penelitian ini bertujuan untuk merancang dan membangun aplikasi pemfilteran SMS penipuan otomatis pada platform Android guna memberikan perlindungan kepada pengguna secara *real-time*. Pendekatan yang digunakan adalah metode *Deep Learning* dengan algoritma *1D-Convolutional Neural Network* (1D-CNN) untuk mengekstraksi fitur pola kalimat bahasa Indonesia secara otomatis. Dataset yang digunakan berjumlah 1.591 baris pesan, yang terdiri dari 920 pesan berlabel normal dan 671 pesan berlabel penipuan, dengan pembagian 80% data latih dan 20% data uji. Aplikasi dikembangkan menggunakan kerangka kerja *React Native*, di mana model kecerdasan buatan diintegrasikan langsung pada sisi perangkat pengguna (*client-side*) menggunakan pustaka *TensorFlow.js*. Untuk menjaga stabilitas aplikasi dari kendala *force close*, komputasi model dieksekusi menggunakan unit pemroses sentral (CPU). Hasil pengujian menunjukkan model 1D-CNN mencapai tingkat akurasi sebesar 96,31%, dengan nilai presisi 94% dan sensitivitas (*recall*) sebesar 96% dalam mendeteksi pesan penipuan. Pengujian fungsional memvalidasi bahwa aplikasi mampu melakukan pemindaian pesan masuk di latar belakang secara berkala melalui mekanisme *looping* setiap 5 detik, memberikan notifikasi peringatan ancaman dengan akurat, serta menyediakan fitur pemindaian teks manual dan kotak masuk secara massal. Dapat disimpulkan bahwa implementasi algoritma 1D-CNN pada aplikasi *mobile* terbukti efektif sebagai solusi pelindung perangkat dari ancaman rekayasa sosial.

Kata Kunci: SMS Penipuan; *Smishing*; *Deep Learning*; 1D-CNN; *React Native*.

DESIGN AND DEVELOPMENT OF AN AUTOMATIC FRAUDULENT SMS FILTERING APPLICATION ON THE ANDROID PLATFORM USING REACT NATIVE AND DEEP LEARNING METHODS

ABSTRACT

The prevalence of cybercrime via Short Message Service (SMS) or smishing has led to financial losses and personal data theft in society. Conventional filtering systems based on keyword matching often fail to detect text obfuscation tactics used by perpetrators. This research aims to design and develop an automatic fraudulent SMS filtering application on the Android platform to provide real-time protection for users. The approach utilized is a Deep Learning method with a 1D-Convolutional Neural Network (1D-CNN) algorithm to automatically extract features of Indonesian sentence patterns. The dataset consists of 1,591 message rows, comprising 920 normal labeled messages and 671 fraudulent labeled messages, split into 80% training and 20% testing data. The application was developed using the React Native framework, where the artificial intelligence model was integrated directly on the client-side using the TensorFlow.js library. To maintain application stability and prevent force-closing issues, the model computation was executed using the central processing unit (CPU). The evaluation results showed that the 1D-CNN model achieved an accuracy rate of 96.31%, with a precision of 94% and a sensitivity (recall) of 96% in detecting fraudulent messages. Functional testing validated that the application is capable of performing background scanning of incoming messages through a 5-second looping mechanism, providing accurate threat warning notifications, and offering manual text and mass inbox scanning features. It can be concluded that the implementation of the 1D-CNN algorithm in a mobile application is proven effective as a device protection solution against social engineering threats.

Keywords: Fraudulent SMS; Smishing; Deep Learning; 1D-CNN; React Native.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Meskipun saat ini teknologi pengiriman pesan berbasis internet seperti WhatsApp dan Messenger telah mendominasi, *Short Message Service* (SMS) tetap memegang peran vital sebagai infrastruktur utama autentikasi keamanan, seperti kode OTP dan notifikasi perbankan. Namun, kepercayaan pengguna terhadap jalur komunikasi ini justru dimanfaatkan oleh pelaku kejahatan siber melalui serangan *smishing* (SMS *Phishing*) untuk mencuri data pribadi atau menipu korban secara finansial. Urgensi ancaman ini tercermin dalam Laporan (Satgas PASTI, 2025), yang mencatat sebanyak 22.993 nomor telepon dilaporkan masyarakat terkait penipuan. Angka statistik ini diperkuat oleh fenomena empiris yang terjadi di lingkungan masyarakat, termasuk pengalaman kerabat terdekat penulis yang menjadi korban rekayasa sosial (*social engineering*) melalui pesan berhadiah palsu. Fenomena ini membuktikan bahwa kewaspadaan manusia saja tidak lagi cukup karena faktor psikologis korban sering kali dimanipulasi, sehingga dibutuhkan sistem pertahanan otomatis yang mampu memblokir pesan berbahaya sebelum terbaca.

Sayangnya, mekanisme penyaringan SMS pada perangkat mobile saat ini belum memadai karena pesan sering kali masuk begitu saja ke kotak masuk tanpa filter yang cerdas. Aplikasi filter konvensional yang beredar mayoritas masih mengandalkan pencocokan kata kunci (*keyword matching*), yang menurut (Al-Kaabi et al., 2024) terbukti tidak efektif menangani taktik *obfuscation* atau penyamaran teks oleh penipu (seperti menulis 'H4diah' atau 'B R I'). Di sisi lain,

pendekatan algoritma klasik seperti Naive Bayes yang digunakan dalam penelitian (Lumbantobing et al., 2021) juga memiliki kelemahan fundamental karena mengasumsikan antar kata bersifat saling bebas (*independent*), sehingga sistem gagal menangkap hubungan kontekstual antar kata dalam kalimat penipuan yang kompleks. Oleh karena itu, penelitian ini mengajukan solusi menggunakan metode *Deep Learning* dengan algoritma *1D-Convolutional Neural Network* (1D-CNN) yang mampu mengekstraksi fitur pola kalimat secara otomatis, yang kemudian diimplementasikan ke dalam aplikasi Android berbasis React Native untuk memberikan perlindungan *real-time* kepada pengguna.

Sebagai solusi dari permasalahan tersebut, penulis ingin menerapkan metode *Deep Learning* dengan algoritma *1D-Convolutional Neural Network* (1D-CNN). Dengan menggunakan algoritma 1D-CNN sistem memiliki keunggulan, berdasarkan jurnal (Faisal et al., 2022) pendekatan 1D-CNN mampu menghasilkan data terstruktur yang mempertahankan informasi urutan kata, semantik, dan sintaksis. Dengan kemampuan tersebut sistem dapat mempelajari urutan kata dan konteks antar kata secara semantik, sehingga pesan penipuan yang menggunakan variasi ejaan atau pola kalimat yang disamarkan tetap dapat diidentifikasi dengan akurasi yang lebih tinggi dan adaptif.

Sistem tersebut akan diterapkan pada aplikasi mobile. Maka React Native akan dipilih sebagai sistem yang akan mengembangkan aplikasi ini dengan mempertimbangkan pengembangan yang lebih efisien dan performa yang cepat. Hasil studi (Ramachandrappa, 2024) menunjukkan bahwa pengembangan menggunakan React Native menawarkan keseimbangan optimal, yaitu React Native memungkinkan pengembang untuk berbagi kode antar *platform*,

mengurangi waktu dan biaya sekaligus mempertahankan performa yang mirip dengan aplikasi native. Dengan aplikasi ini, proses penyaringan SMS akan lebih akurat dan efisien.

Berdasarkan masalah penipuan SMS dan pengembangan yang akan dilakukan dengan pendekatan *Deep Learning* untuk mengatasi maraknya penipuan lewat SMS, maka penulis akan mengajukan penelitian dengan judul **"RANCANG BANGUN APLIKASI PEMFILTERAN SMS PENIPUAN OTOMATIS PADA PLATFORM ANDROID MENGGUNAKAN REACT NATIVE DAN METODE DEEP LEARNING"**.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, maka rumusan masalah dalam penelitian ini adalah:

1. Bagaimana menerapkan metode *Deep Learning* dengan algoritma *1D-Convolutional Neural Network* (1D-CNN) untuk mendeteksi SMS penipuan?
2. Bagaimana merancang dan membangun aplikasi pemfilteran SMS otomatis pada platform Android menggunakan React Native yang terintegrasi dengan model *Deep Learning*?
3. Seberapa efektif penerapan algoritma 1D-CNN dalam mengklasifikasikan SMS penipuan dan SMS normal?

1.3. Batasan Masalah

Agar penelitian ini lebih terarah dan tidak menyimpang dari topik pembahasan, penulis menetapkan batasan masalah yang difokuskan pada pengembangan aplikasi perangkat bergerak (*mobile*) berbasis sistem operasi Android dengan menggunakan *framework* React Native. Sistem deteksi yang

dibangun dalam aplikasi ini secara spesifik menerapkan metode klasifikasi Deep Learning dengan algoritma *1D-Convolutional Neural Network* (1D-CNN). Adapun model tersebut dilatih menggunakan dataset berupa kumpulan teks SMS berbahasa Indonesia dengan jumlah data sebanyak 1611 data yang dikelompokkan ke dalam kategori penipuan (*spam/fraud*) dan normal.

1.4. Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dipaparkan, tujuan dari penelitian ini adalah:

1. Menerapkan metode *Deep Learning* dengan algoritma *1D-Convolutional Neural Network* (1D-CNN) untuk mengenali pola teks SMS penipuan berbahasa Indonesia.
2. Merancang dan membangun aplikasi pemfilteran SMS pada platform Android menggunakan React Native yang mampu mendeteksi pesan penipuan secara otomatis.
3. Mengukur tingkat akurasi, presisi, dan *recall* dari model 1D-CNN yang diimplementasikan dalam mendeteksi SMS penipuan.

1.5. Manfaat Penelitian

Penelitian ini diharapkan bisa membawa manfaat nyata bagi banyak pihak. Hasil utamanya adalah sebuah aplikasi Android yang bisa otomatis membedakan mana SMS asli dan mana SMS penipuan. Dengan begitu, masyarakat bisa lebih terlindungi dan tidak mudah tertipu atau kehilangan uang gara-gara SMS jahat. Selain itu, penelitian ini juga bisa menjadi bahan belajar baru di dunia teknologi, terutama sebagai contoh cara membuat aplikasi pintar (AI) dan aplikasi Android, sehingga bisa ditiru atau dikembangkan lagi oleh peneliti lain di masa depan.

BAB II

LANDASAN TEORI

2.1. *Short Message Service* (SMS) dan *Smishing*

2.1.1. Definisi SMS

Layanan Pesan Singkat (SMS) merupakan protokol komunikasi yang umum digunakan untuk mengirim pesan teks singkat di antara perangkat ponsel. Meskipun teknologi berkembang, SMS tetap relevan. Hal ini dipertegas oleh (Airlangga, 2024) yang menyatakan bahwa SMS (*Short Message Service*) telah menjadi platform penting untuk mengirimkan informasi, meskipun volume pesan sampah yang tidak diminta juga terus meningkat.

Secara teknis, SMS bekerja melalui jalur sinyal di jaringan *GSM*, bukan melalui jalur suara. Ini memungkinkan pengiriman dan penerimaan SMS tetap berlangsung meskipun pengguna sedang melakukan panggilan suara. Mekanisme pengiriman SMS menggunakan konsep Simpan-dan-Kirim. Ketika sebuah pesan dikirim, pesan tersebut tidak segera diterima oleh penerima, tetapi terlebih dahulu akan dialihkan ke Pusat Layanan Pesan Singkat (*SMSC*). *SMSC* bertanggung jawab untuk menyimpan pesan tersebut dan melakukan usaha untuk mengirimkannya ke nomor yang dituju. Kapasitas terbesar untuk satu pesan SMS dibatasi hingga 160 karakter (7-*bit*), yang mengharuskan penggunaan singkatan atau pengiriman pesan terpisah (SMS terhubung) jika konten melebihi batas ini.

2.1.2. Penipuan Digital (*Smishing*)

Dalam sistem komunikasi seluler, risiko keamanan tidak hanya terdiri dari pesan sampah (*spam*) yang bersifat pemasaran, melainkan telah berkembang menjadi bentuk penipuan digital yang disebut Smishing. Istilah *Smishing* adalah

kombinasi dari "SMS" dan "*Phishing*", yang diartikan sebagai usaha kriminal untuk mendapatkan informasi sensitif (seperti kata sandi, nomor kartu kredit, atau data pribadi) dengan menyamar sebagai pihak yang tepercaya melalui pesan teks. Fenomena ini dijelaskan oleh (Hikmaturokhman et al., 2022), yang mencatat bahwa pengguna telepon seluler sering kali diteror oleh pesan spam dengan konten yang berpura-pura (*pretentious content*) yang berasal dari nomor tidak dikenal dan berisi pesan atau tautan ke situs penipuan.

Berbeda dengan spam pemasaran yang hanya bersifat mengganggu, Smishing masuk dalam kategori *fraud* (penipuan) yang memiliki niat jahat (*malicious intent*). Para pelaku kejahatan siber menggunakan berbagai modus operandi untuk memperdaya korban dan menghindari filter keamanan operator seluler, antara lain:

1. Manipulasi Tautan (*Malicious Links*), penipu menyertakan tautan *URL* yang terlihat sah namun sebenarnya mengarahkan korban ke situs web palsu yang dirancang menyerupai situs resmi bank atau instansi pemerintah. Seringkali, penipu menggunakan layanan pemendek tautan (*URL shortener*) seperti bit.ly atau s.id untuk menyembunyikan alamat asli situs berbahaya tersebut.
2. Rekayasa Sosial (*Social Engineering*), serangan ini memanfaatkan psikologis korban dengan menawarkan hadiah palsu (misalnya: "Selamat Anda menang undian Rp100 Juta") atau menciptakan rasa urgensi dan ketakutan (misalnya: "Rekening Anda diblokir, segera konfirmasi"). Tujuannya adalah memancing reaksi impulsif agar korban segera mengklik tautan atau menghubungi nomor penipu tanpa berpikir panjang.

3. Penyamaran Karakter (*Text Obfuscation*), untuk menghindari deteksi sistem filter otomatis yang berbasis kata kunci (*keyword-based filtering*), penipu sengaja memodifikasi penulisan teks.

2.2. Natural Language Processing (NLP)

Natural Language Processing (NLP) merupakan area dalam kecerdasan buatan yang menghubungkan komunikasi antara bahasa manusia dan komputer, dengan tujuan mengambil makna dari kumpulan data teks yang tidak terstruktur. Karena model *Deep Learning* tidak dapat langsung menganalisis teks mentah, NLP berperan untuk mengubah input bahasa menjadi representasi vektor yang bersifat matematis. Keterkaitan NLP dengan keamanan siber ditegaskan oleh (Amin et al., 2024), yang dalam penelitiannya memanfaatkan pemrosesan teks berbasis model bahasa untuk mendeteksi spam berbahasa Indonesia dengan memahami konteks semantik agar dapat membedakan antara pesan penipuan dan pesan normal secara akurat.

2.3. Deep Learning

Deep Learning adalah subbidang dari *Machine Learning* yang mengadopsi struktur jaringan saraf tiruan berlapis (*Artificial Neural Networks*) untuk meniru mekanisme otak manusia dalam memproses informasi dan mengambil keputusan. Algoritma ini memiliki kemampuan unik untuk mempelajari representasi fitur secara hierarkis dan otomatis dari data mentah, berbeda dengan metode pembelajaran mesin konvensional yang sangat bergantung pada rekayasa fitur manual oleh manusia. Pendekatan ini memungkinkan komputer untuk menyelesaikan permasalahan non-linear yang kompleks, seperti pengenalan pola pada citra, suara, dan teks tidak terstruktur, Hal ini sejalan dengan (Moulana et al.,

2024) yang menyatakan bahwa *Deep Learning* menawarkan kemampuan superior dalam mempelajari dan mengenali pola dari dataset besar, sehingga tingkat akurasinya terus meningkat seiring dengan bertambahnya volume data yang dilatih.

Keunggulan utama *Deep Learning* dibandingkan algoritma tradisional dapat dilihat secara komprehensif pada perbandingan di bawah ini:

Tabel 2.1 Perbandingan Machine Learning dan *Deep Learning*

Aspek Komparasi	**Machine Learning Tradisional (SVM, Naive Bayes)**	***Deep Learning* (CNN, LSTM)**
Ekstraksi Fitur	Membutuhkan seleksi fitur manual yang rumit dan bergantung pada keahlian manusia (*hand-crafted features*).	Melakukan ekstraksi fitur secara otomatis dari data mentah (*automatic feature extraction*).
Kinerja pada Data Besar	Kinerja cenderung stagnan (*plateau*) meskipun jumlah data ditambah.	Kinerja terus meningkat seiring dengan bertambahnya volume data latih (*scalable*).
Penanganan Data Tidak Terstruktur	Kurang efektif menangani data kompleks seperti teks bebas, gambar, atau suara.	Sangat unggul dalam memproses data tidak terstruktur (*unstructured data*) seperti teks SMS.

Aspek Komparasi	Machine Learning Tradisional (SVM, Naive Bayes)	*Deep Learning* (CNN, LSTM)
Ketergantungan Domain	Sangat bergantung pada pengetahuan spesifik domain untuk menentukan fitur yang relevan.	Lebih fleksibel dan dapat beradaptasi dengan berbagai domain tanpa rekayasa fitur berat.
Pemahaman Konteks	Seringkali hanya melihat frekuensi kemunculan kata (seperti *Bag of Words*) tanpa memahami urutan.	Mampu menangkap hubungan semantik dan urutan kata (*sequential context*) dalam kalimat.

2.4. Text Preprocessing

Text Preprocessing adalah langkah dasar untuk membersihkan dan menyesuaikan data sebelum diproses ke dalam model pelatihan. Berdasarkan pendapat (Maugy Al Kautsar et al., 2025), langkah-langkah standar dalam pra-pemrosesan data meliputi pembersihan data, konversi huruf menjadi kecil, tokenisasi, penghapusan kata umum (*stopword*), dan *stemming*. Dalam studi ini, langkah-langkah tersebut diterapkan sebagai berikut:

1. Case Folding & Cleaning, proses dimulai dengan mengubah seluruh huruf menjadi kecil (*lowercase*) untuk menjaga konsistensi data. Selanjutnya, teknik pembersihan dilakukan menggunakan *Regular Expression* (Regex) untuk menghapus simbol non-alfabet dan melakukan substitusi token khusus. Pola URL diganti menjadi token **"[LINK]"** dan deretan angka diganti menjadi

"[ANGKA]" agar model fokus pada indikator pola penipuan, bukan pada isi spesifik tautan atau nilai nominalnya.

2. Tokenizing, kalimat yang telah bersih dipecah menjadi potongan kata individual (*token*) berdasarkan spasi.
3. Sequence & Padding, langkah terakhir adalah mengonversi setiap token kata menjadi angka integer unik berdasarkan indeks kamus kata. Karena arsitektur 1D-CNN membutuhkan dimensi input yang tetap, diterapkan teknik *Padding* dengan batas maksimum (*max length*) 50 kata. Jika urutan kata kurang dari batas tersebut, sistem akan menambahkan nilai nol (0), dan jika berlebih akan dipotong (*truncate*).

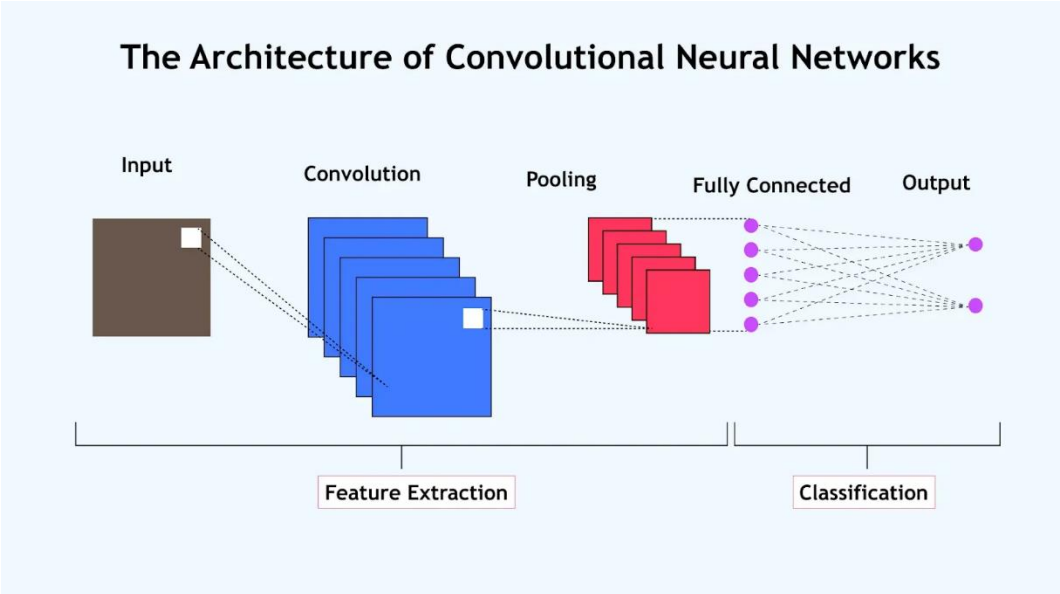
2.5. Convolutional Neural Network (CNN)

2.5.1. Konsep Dasar CNN

Convolutional Neural Network (CNN) adalah struktur *Deep Learning* yang dirancang untuk mengolah data yang memiliki format grid atau berurutan melalui penggunaan metode konvolusi matematis. Proses inti dalam CNN berlangsung pada Lapisan Konvolusi, di mana berbagai *filter* (kernel) diterapkan pada input untuk secara otomatis mengambil fitur-fitur lokal yang penting, seperti pola visual atau struktur tertentu dari data. Hasil ekstraksi ini selanjutnya diolah oleh *Pooling Layer* yang memiliki tugas untuk mengurangi dimensi data dan parameter komputasi tanpa menghapus informasi utama, sehingga model menjadi lebih efisien dan lebih tahan terhadap perubahan kecil pada input.

Setelah melewati langkah ekstraksi fitur dan pengurangan dimensi, data akan diubah menjadi vektor satu dimensi agar dapat diproses di *Fully Connected Layer*. Lapisan ini berfungsi seperti jaringan saraf tiruan tradisional yang

menghubungkan setiap neuron sepenuhnya dengan lapisan sebelumnya untuk menjalankan proses penalaran yang kompleks. Di tahap terakhir, fungsi aktivasi (seperti *Sigmoid* atau *Softmax*) digunakan untuk menghitung nilai probabilitas akhir dalam menentukan kelas keluaran yang benar. Tujuannya adalah untuk mencapai akurasi tinggi sebagaimana dijelaskan oleh (Hasti & Barmada, 2025) , yaitu membangun model yang efektif untuk Mendeteksi dan mengkategorikan pesan *smishing* SMS secara akurat dan mengatasi tantangan ketidakseimbangan data yang sering terjadi pada kasus penipuan.



Gambar 2.1 Arsitektur CNN

2.5.2. *One-Dimensional Convolutional Neural Network* (1D-CNN)

Dalam konteks pengolahan data teks, penelitian ini menerapkan varian *One-Dimensional CNN* (1D-CNN) yang memiliki perbedaan fundamental dengan 2D-CNN pada mekanisme pergerakan kernelnya. Jika 2D-CNN menggeser filter secara dua arah (matriks horizontal dan vertikal) yang umum digunakan untuk pengolahan citra, 1D-CNN hanya menggeser filter satu arah secara sekuensial melintasi sumbu urutan data.

Karakteristik pergerakan satu arah ini menjadikan 1D-CNN sangat ideal untuk *Natural Language Processing* (NLP) karena mampu menangkap hubungan antarkata yang berdekatan (*local context*) atau *n-grams* dalam sebuah kalimat secara efektif. Pendekatan ini juga menjawab tantangan yang soroti oleh (Latifah et al., 2024) , di mana traditional methods have a limitation compared to *deep learning* karena ketergantungannya pada ekstraksi fitur manual, sedangkan arsitektur seperti 1D-CNN menawarkan efisiensi komputasi yang lebih ringan sekaligus kemampuan ekstraksi fitur otomatis yang lebih adaptif terhadap variasi teks.

2.6. Platform Pengembangan Aplikasi

2.6.1. Sistem Operasi Android

Android merupakan sistem operasi yang didasarkan pada kernel Linux, yang dirancang khusus untuk perangkat bergerak dengan layar sentuh, seperti *smartphone* dan tablet. Melalui arsitektur terbuka (*open-source*), Android memberikan kebebasan kepada pengembang untuk mengakses fungsi-fungsi mendalam dari perangkat keras. Pemilihan platform Android dalam penelitian ini ditentukan oleh kebutuhan teknis utama dari aplikasi pendeteksi penipuan, yaitu kemampuan untuk mengakses data pesan yang masuk (*inbox*). Ekosistem iOS menerapkan kebijakan privasi "*sandbox*" yang ketat, yang membatasi aplikasi pihak ketiga dalam mengakses konten SMS secara langsung. Sebaliknya, Android menawarkan izin akses (*permission*) seperti *READ_SMS* yang memungkinkan aplikasi untuk mengakses dan menganalisis pesan teks secara langsung, menjadikannya satu-satunya lingkungan yang sesuai untuk menerapkan sistem deteksi *smishing* ini.

2.6.2. React Native

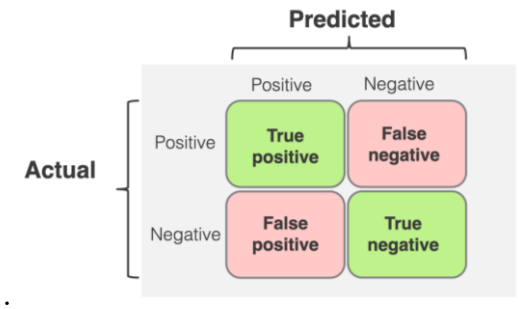
Dalam mengembangkan antarmuka dan logika aplikasi, studi ini memanfaatkan React Native, yaitu sebuah *framework* pengembangan multiplatform yang dirancang oleh Meta. Dari segi arsitektur, React Native berfungsi melalui konsep "*Bridge*" atau jembatan komunikasi yang mengaitkan kode *JavaScript* dengan modul asli (*native modules*) pada perangkat. Mekanisme jembatan ini sangat penting dalam konteks penerapan keamanan ini, karena memungkinkan logika aplikasi yang ditulis dalam *JavaScript* untuk melakukan komunikasi secara asinkron dengan sistem Android guna meminta izin akses dan mendapatkan data pesan SMS dari perangkat pengguna, sesuatu yang tidak bisa dilakukan oleh aplikasi web biasa.

Selain keterampilan teknis, pilihan untuk menggunakan React Native didorong oleh efisiensi dalam pengembangan yang berlandaskan pada filosofi "*Learn Once, Write Anywhere*". Kerangka kerja ini memungkinkan penggunaan satu basis kode untuk menciptakan aplikasi yang dapat beroperasi dengan performa yang hampir setara dengan aplikasi asli (*native*). Selanjutnya, karena React Native adalah pengembangan langsung dari pustaka React yang memakai sintaksis *JavaScript*, hal ini mengoptimalkan penggunaan keterampilan pemrograman yang telah kuasai sebelumnya. Dengan cara ini, proses pengembangan aplikasi dapat berlangsung lebih cepat karena kurva pembelajaran menjadi lebih pendek, sehingga penelitian dapat lebih difokuskan pada integrasi model kecerdasan buatan untuk mendeteksi penipuan ketimbang mempelajari bahasa pemrograman yang baru.

2.7. Pengukuran Performa Model (*Evaluation Metrics*)

Evaluasi kinerja merupakan tahapan fundamental untuk memvalidasi apakah model *Deep Learning* yang dibangun mampu melakukan klasifikasi dengan baik. Dasar dari seluruh pengukuran performa ini adalah *Confusion Matrix*, yaitu sebuah tabel representasi yang membandingkan hasil prediksi model terhadap label kebenaran dasar (*ground truth*). Hal ini dipertegas oleh (Menthe et al., 2024), yang menyatakan bahwa dalam pengembangan sistem deteksi, metrik evaluasi seperti presisi, recall, dan skor F1 memainkan peran krusial dalam menilai kinerja model deteksi spam serta memandu pemilihan pendekatan yang paling optimal untuk diterapkan.

Tabel ini memetakan empat kemungkinan hasil prediksi: *True Positive* (TP) yang berarti model dengan tepat memprediksi pesan spam sebagai spam; *True Negative* (TN) di mana model dengan tepat mengenali pesan normal sebagai normal; *False Positive* (FP) atau kesalahan tipe I, terjadi ketika pesan normal salah diprediksi sebagai spam; dan *False Negative* (FN) atau kesalahan tipe II, di mana pesan spam lolos dan diprediksi sebagai pesan normal



Gambar 2.2 Komponen-Komponen Dalam *Confusion Matrix*

Berdasarkan komponen-komponen dalam *Confusion Matrix* tersebut, dapat diturunkan berbagai metrik perhitungan, yang pertama adalah **Akurasi (*Accuracy*)**. Akurasi merepresentasikan rasio total prediksi yang benar (baik positif maupun

negatif) terhadap keseluruhan jumlah data. Metrik ini memberikan gambaran umum mengenai seberapa sering model membuat keputusan yang tepat secara global. Rumus matematis untuk menghitung akurasi dinyatakan dalam persamaan berikut:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Selain akurasi, metrik **Presisi (*Precision*)** sangat krusial dalam konteks deteksi spam untuk mengukur ketepatan prediksi positif. Presisi menghitung seberapa banyak pesan yang diprediksi sebagai spam memang benar-benar merupakan spam. Nilai presisi yang tinggi mengindikasikan bahwa model memiliki tingkat kesalahan rendah dalam memberikan "alarm palsu" (*False Positive*), yang sangat penting agar pesan penting pengguna (seperti OTP atau notifikasi bank resmi) tidak terblokir secara tidak sengaja. Rumus presisi didefinisikan sebagai:

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Metrik pelengkap lainnya adalah **Recall** atau sering disebut juga sebagai Sensitivitas. Berbeda dengan presisi yang fokus pada ketepatan prediksi, *Recall* mengukur kemampuan model dalam menemukan kembali seluruh pesan spam yang ada dalam dataset. Nilai *Recall* yang tinggi menandakan bahwa model efektif dalam menangkap ancaman dan meminimalisir jumlah spam yang lolos ke kotak masuk pengguna (*False Negative*). Persamaan untuk menghitung *Recall* adalah:

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

2.8. Ringkasan Penelitian Terdahulu

Berdasarkan pemaparan penelitian-penelitian terdahulu pada subbab sebelumnya yang memiliki keterkaitan langsung dengan topik penelitian ini, banyak pendekatan yang diadaptasi berfokus pada algoritma *Machine Learning*

konvensional seperti Regresi Logistik dan *Support Vector Machine (*SVM*)*, yang mengandalkan ekstraksi fitur yang dilakukan secara manual. Namun, karena kompleksitas pola bahasa yang digunakan oleh para pelaku spam, arah penelitian mulai berubah ke *Deep Learning*, yang memiliki kemampuan untuk melakukan ekstraksi fitur secara otomatis.

Beberapa studi sebelumnya telah berhasil menerapkan berbagai teknik klasifikasi untuk mengidentifikasi pesan *spam*, baik di platform *Email* maupun SMS. Namun, setiap teknik memiliki karakteristik kinerja yang berbeda dalam hal akurasi, waktu pengolahan, dan kebutuhan sumber daya. Berikut adalah ringkasan penelitian yang relevan dengan topik ini:

Tabel 2.2 Ringkasan Penelitian Terdahulu

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
1	Analisis Klasifikasi SMS Spam Menggunakan Logistic Regression (Reviantika et al., 2021)	Penelitian ini bertujuan mengklasifikasikan pesan spam dan *ham* menggunakan pendekatan statistik sederhana. Dengan menggunakan dataset sebanyak 1.143 pesan dan	Logistic Regression (TF-IDF)	Kelebihan: Model sangat ringan dan proses pelatihan (*training*) sangat cepat, cocok untuk sumber daya terbatas.

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
		pembagian data latih:uji 90:10, penelitian ini berhasil mencapai akurasi sebesar 95%.		Kekurangan: Kurang adaptif menangani variasi kata baru atau bahasa gaul yang tidak ada dalam kamus latih (*out-of-vocabulary*).
2	Deep Learning Algorithm Models for Spam Identification on Cellular Short Message Service (Hikmaturokhman et al., 2022)	Studi ini mengevaluasi kinerja algoritma *Deep Learning* dalam mengidentifikasi SMS penipuan (*fraudulent*) dan konten palsu pada	Deep Learning (Comparative Study)	Kelebihan: Mampu melakukan ekstraksi fitur secara otomatis (tanpa rekayasa fitur manual) dan

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
		jaringan seluler. Hasil penelitian menunjukkan bahwa *Deep Learning* mampu menangani ekstraksi fitur teks yang kompleks secara otomatis tanpa intervensi manual yang rumit.		efektif mendeteksi pola penipuan yang rumit. Kekurangan: Membutuhkan dataset dalam jumlah besar agar model dapat belajar optimal, serta memakan sumber daya komputasi tinggi.
3	Optimizing SMS Spam Detection Using Machine Learning: A	Penelitian ini melakukan analisis komparatif berbagai model ML pada dataset	Ensemble Voting Classifier, SVM, Random Forest	Kelebihan: Tingkat akurasi sangat tinggi dan

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
	Comparative Analysis (Airlangga, 2024)	standar berisi 5.572 pesan. Hasil pengujian menunjukkan SVM mencapai akurasi tertinggi (98.57%), disusul oleh *Ensemble Voting* (98.48%), membuktikan stabilitas metode ini pada dataset SMS berbahasa Inggris.		stabil untuk dataset standar. Kekurangan: Kompleksitas komputasi meningkat pada metode *Ensemble*, sehingga waktu eksekusi lebih lama dan kurang efisien untuk aplikasi *real-time* di HP.
4	Deteksi Spam Berbahasa Indonesia Berbasis Teks	Penelitian ini menerapkan model berbasis *Transformer*	BERT (*Bidirectional Encoder Representations*	Kelebihan:

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
	Menggunakan Model BERT (Amin et al., 2024)	(BERT) untuk mendeteksi spam berbahasa Indonesia. Fokus utama adalah kemampuan model memahami konteks semantik (makna kalimat) secara mendalam, bukan hanya frekuensi kemunculan kata.	*from Transformers*)	Sangat unggul dalam memahami konteks bahasa dan makna kalimat (semantik) yang ambigu. Kekurangan: Arsitektur model sangat "berat" (*resource-intensive*), sulit diimplementasikan pada perangkat *mobile* dengan spesifikasi rendah.
5	Analisis Komparasi	Penelitian ini membandingkan	CNN (*Convolutional*	Kelebihan:

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
	Kinerja LSTM dan CNN dalam Deteksi Spam Email Berbasis Deep Learning (Kautsar, Setiaji, & Rifa'i, 2025)	dua arsitektur *Deep Learning* populer, CNN dan LSTM, pada dataset 5.572 email. Hasilnya mengkonfirmasi karakteristik masing-masing model: CNN efisien dalam mengenali pola lokal (frasa spam), sementara LSTM unggul dalam memahami urutan panjang.	*Neural Network*) & LSTM	Membuktikan bahwa CNN efektif dan efisien dalam mengenali pola fitur lokal pada teks. Kekurangan: Objek penelitian adalah Email yang memiliki struktur kalimat lebih panjang/formal, berbeda karakteristik dengan SMS yang

NO	Judul dan peneliti	Pembahasan	Metode	Kelebihan atau Kekurangan
				pendek dan singkatan.

2.9. Analisis Gap

Berdasarkan kajian literatur yang ada, telah ditemukan beberapa kelemahan penting yang belum ditangani dengan baik dalam pengembangan sistem keamanan pesan. Sebagian besar penelitian yang ada masih didominasi oleh metode Pembelajaran Mesin tradisional, seperti Regresi Logistik dan Mesin Vektor Dukungan (SVM). Walaupun teknik-teknik tersebut sangat efisien dalam pemanfaatan sumber daya komputasi, kemampuan mereka sering kali terbatas dalam memahami konteks semantik yang rumit pada pola penipuan modern yang terus berkembang. Sebaliknya, penelitian yang menggunakan metode *Deep Learning* terbaru seperti BERT menghadapi tantangan terkait dengan penggunaan komputasi yang sangat besar, sehingga sulit untuk diterapkan pada perangkat mobile dengan spesifikasi yang terbatas. Dengan demikian, studi ini merekomendasikan penerapan metode 1D-CNN yang menyediakan keseimbangan yang sempurna: kemampuan dalam mengekstrak fitur secara cerdas seperti algoritma *Deep Learning* yang kompleks, tetapi dengan struktur yang jauh lebih ringan dan efisien dibandingkan dengan model yang berbasis *Transformer* atau LSTM.

Selain faktor metode, terdapat jurang yang signifikan berkaitan dengan objek data yang diteliti. Sebagian besar studi sebelumnya lebih mengutamakan

analisis pada media email yang umumnya memiliki kalimat yang formal dan panjang, atau terbatas pada penggunaan dataset dalam bahasa Inggris. Sesungguhnya, bahaya yang benar-benar dihadapi oleh masyarakat Indonesia saat ini adalah *SMS Phishing* (Smishing) yang memanfaatkan ciri khas bahasa yang sangat khusus, seperti kombinasi singkatan tidak resmi, penggunaan bahasa gaul, dan pola penulisan yang acak untuk menghindari perhatian. Variasi bahasa lokal ini belum sepenuhnya diperhatikan oleh model-model yang dilatih dengan menggunakan bahasa formal atau bahasa dari luar negeri.

Akhirnya, analisis terhadap literatur yang ada menunjukkan bahwa pengembangan model deteksi spam biasanya terhenti pada fase simulasi dan penilaian metrik dalam lingkungan komputasi yang terbatas, seperti di laptop penelitian. Hingga saat ini, belum ada langkah yang berarti untuk menerapkan model tersebut sebagai solusi praktis yang dapat diakses langsung oleh pengguna akhir. Penelitian ini bertujuan untuk mengatasi kekurangan dalam implementasi tersebut dengan tidak hanya menciptakan model teori, tetapi juga menerapkannya dalam aplikasi Android secara nyata, untuk menunjukkan bahwa model deteksi spam yang canggih dapat berfungsi dengan baik dan secara waktu nyata dalam lingkungan ponsel pintar.

BAB III

METODOLOGI PENELITIAN

3.1. Metodologi Penelitian

3.1.1. Teknik Pengumpulan Data

Pengumpulan data dalam penelitian ini dilakukan melalui metode studi dokumentasi dengan memanfaatkan sumber data sekunder dan primer untuk membentuk *dataset* yang komprehensif. Data sekunder diperoleh dengan mengunduh dataset SMS *spam* berbahasa Indonesia dari repositori publik GitHub yang tersedia dalam format *Comma Separated Values* (.CSV). Guna meningkatkan akurasi dan relevansi model terhadap tren penipuan terkini, penulis melakukan augmentasi data secara manual dengan menambahkan lebih dari 300 sampel pesan baru. Data tambahan ini dikumpulkan berdasarkan observasi langsung terhadap pola pesan penipuan (*smishing*) yang beredar di operator seluler lokal namun belum terakomodasi dalam dataset publik, sehingga total data yang digunakan diharapkan mampu merepresentasikan variasi serangan yang lebih luas dan adaptif.

3.1.2. Alur Penelitian (Research Flow)

Guna menjamin kelancaran dan keteraturan proses ilmiah, penulis merancang sebuah kerangka kerja sistematis yang memetakan seluruh tahapan penelitian dari awal hingga akhir.

Gambar 3.1 Alur Penelitian

Tahapan penelitian diawali dengan studi literatur dan identifikasi masalah terkait maraknya smishing, dilanjutkan dengan pengumpulan data menggunakan metode hibrida yang menggabungkan dataset publik GitHub dan augmentasi mandiri sebanyak lebih dari 300 sampel pesan terbaru. Data mentah tersebut kemudian melalui proses pra-pemrosesan teks yang meliputi Case Folding, Cleaning, dan Tokenizing untuk melatih arsitektur 1D-Convolutional Neural

Network (1D-CNN) menggunakan pembagian data latih dan uji. Setelah pelatihan, kinerja model dievaluasi menggunakan parameter Confusion Matrix dengan target akurasi di atas 90% serta penerapan Hyperparameter Tuning jika diperlukan. Model yang telah tervalidasi selanjutnya diintegrasikan ke dalam perancangan antarmuka dan implementasi aplikasi Android berbasis React Native, yang diakhiri dengan pengujian fungsional Black Box Testing untuk memastikan sistem berjalan real-time sesuai harapan sebelum penarikan kesimpulan akhir.

3.2. Alat dan Bahan Penelitian

3.2.1. Perangkat Keras (*Hardware*)

Untuk mendukung kelancaran proses pengembangan sistem yang membutuhkan sumber daya komputasi yang memadai, penelitian ini menggunakan spesifikasi perangkat keras sebagai berikut:

1. **Laptop ASUS TUF A15,** Berfungsi sebagai unit komputasi utama (*main station*) untuk seluruh aktivitas pemrograman, mulai dari pra-pemrosesan data hingga implementasi aplikasi. Laptop ini ditenagai oleh prosesor AMD Ryzen 7 Seri 7000 dan didukung memori (RAM) sebesar 16 GB untuk menangani *multitasking* berat, serta penyimpanan SSD 512 GB untuk kecepatan akses data yang tinggi. Sistem operasi yang digunakan adalah Windows 11.
2. **Smartphone Samsung Galaxy A06,** Digunakan sebagai perangkat uji fisik (*real device*) untuk memvalidasi performa dan tampilan aplikasi di lingkungan nyata. Perangkat ini memiliki spesifikasi RAM 6 GB dan penyimpanan internal 128 GB, yang memadai untuk menjalankan aplikasi React Native dan modul deteksi secara lancar.

3. **Android Virtual Device (Emulator),** Digunakan untuk kebutuhan simulasi awal dan proses *debugging* yang cepat tanpa perlu menghubungkan perangkat fisik secara berulang. Emulator ini dikonfigurasi dengan profil "Medium Phone" berbasis Android API Level 36.1 dan alokasi RAM virtual sebesar 4 GB.

3.2.2. Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan dalam penelitian ini mencakup alat untuk desain, pemrograman, hingga pelatihan model kecerdasan buatan, dengan rincian sebagai berikut:

1. **Figma,** Digunakan pada tahap perancangan antarmuka pengguna (*User Interface*) untuk memvisualisasikan tata letak (*wireframe*) dan desain aplikasi sebelum tahap pengkodean dimulai.
2. **Visual Studio Code,** Berfungsi sebagai editor teks (*code editor*) utama untuk menulis dan mengelola kode program aplikasi maupun skrip pra-pemrosesan data.
3. **Android Studio,** Digunakan secara spesifik untuk manajemen *Software Development Kit* (SDK), pengaturan emulator, dan kompilasi paket aplikasi Android (.apk).
4. **Google Colab,** Layanan berbasis awan (*cloud computing*) yang digunakan untuk melatih model *Deep Learning* 1D-CNN, memanfaatkan akses GPU gratis yang disediakan Google untuk mempercepat proses komputasi grafis.
5. **Bahasa Pemrograman Python,** Digunakan untuk membangun logika sisi *backend*, khususnya untuk pengolahan data teks (NLP) dan pembangunan model kecerdasan buatan menggunakan pustaka **TensorFlow** dan **Keras**.

6. **Bahasa Pemrograman JavaScript,** Digunakan sebagai bahasa utama dalam pengembangan logika aplikasi *mobile* yang berjalan di sisi pengguna (*client-side*).
7. **React Native & Expo Framework,** Kerangka kerja yang digunakan untuk membangun aplikasi Android yang responsif, efisien, dan memiliki kemampuan untuk mengakses fitur sistem *native* seperti pembacaan SMS.

3.3. Metode Pemecahan Masalah

Penelitian ini menggunakan pendekatan yang teratur untuk mengatasi masalah utama dalam analisis teks SMS yang tidak terstruktur. Strategi yang pertama ditujukan untuk memproses data mentah dengan memanfaatkan teknik *Natural Language Processing* (NLP). Mengingat bahwa pesan teks sering kali memiliki gangguan berupa simbol tidak teratur dan format yang tidak konsisten, teknik NLP digunakan untuk menstandarisasi data tersebut melalui langkah-langkah pembersihan karakter, pemisahan kalimat menjadi token kata, dan konversi teks menjadi representasi vektor angka. Tahapan ini selaras dengan pendekatan (Al-Zebari et al., 2024) yang menerapkan teknik *embedding* untuk mengubah teks menjadi format numerik untuk analisis *deep learning*, sehingga mesin memperoleh data masukan yang bersih dan terorganisir agar proses pembelajaran dapat berlangsung dengan baik.

Selanjutnya, tantangan dalam menemukan pola penipuan yang selalu berubah diatasi dengan menerapkan algoritma *Deep Learning* jenis *1D-Convolutional Neural Network* (1D-CNN). Tidak seperti metode tradisional yang mengikuti aturan yang ketat, pendekatan ini memanfaatkan kemampuan 1D-CNN untuk secara otomatis mengambil fitur pola kalimat lokal atau *n-gram* dari data

pelatihan. Hal ini didukung oleh (Al-Kabbi et al., 2024) yang mengembangkan sistem di mana ekstraksi fitur dilakukan menggunakan Convolutional Neural Network (CNN) untuk analisis tingkat kata, yang memungkinkan identifikasi pola kata yang menunjukkan adanya penipuan secara otomatis tanpa memerlukan penetapan fitur manual yang kompleks. Dengan mekanisme ini, tingkat akurasi deteksi tetap terjaga meskipun metode penipuan mengalami perubahan.

Strategi terakhir bertujuan untuk mengatasi masalah aksesibilitas agar solusi ini dapat dimanfaatkan secara luas oleh masyarakat. Penulis mengembangkan aplikasi *mobile* untuk Android dengan memanfaatkan kerangka kerja React Native sebagai sarana untuk melaksanakan model. Pendekatan ini memungkinkan sistem deteksi beroperasi langsung pada sisi perangkat pengguna (*client side*), sehingga pemindaian pesan dapat dilakukan secara langsung dan efisien. Ini menghubungkan perbedaan antara model kecerdasan buatan yang rumit dengan antarmuka pengguna yang simpel dan mudah digunakan dalam kehidupan sehari-hari.

3.3.1. Analisis Data dan Perancangan Sistem

1. Struktur Data

Basis data yang mendasari penelitian ini disimpan dalam bentuk file *Comma Separated Values* (CSV) yang berasal dari kombinasi dataset publik dan koleksi pribadi. Berbeda dengan format tabel tradisional yang membagi kolom dengan jelas, dataset ini menggunakan satu baris tunggal dengan tanda titik koma (;) berfungsi sebagai pemisah antara konten pesan dan label kategorinya. Contohnya, sebuah baris data berisi string teks pesan yang diikuti langsung oleh label klasifikasinya, seperti dalam format "isi pesan;label". Pendekatan ini dipilih karena keefisienannya dalam menyimpan data teks mentah, yang nantinya akan dipisahkan

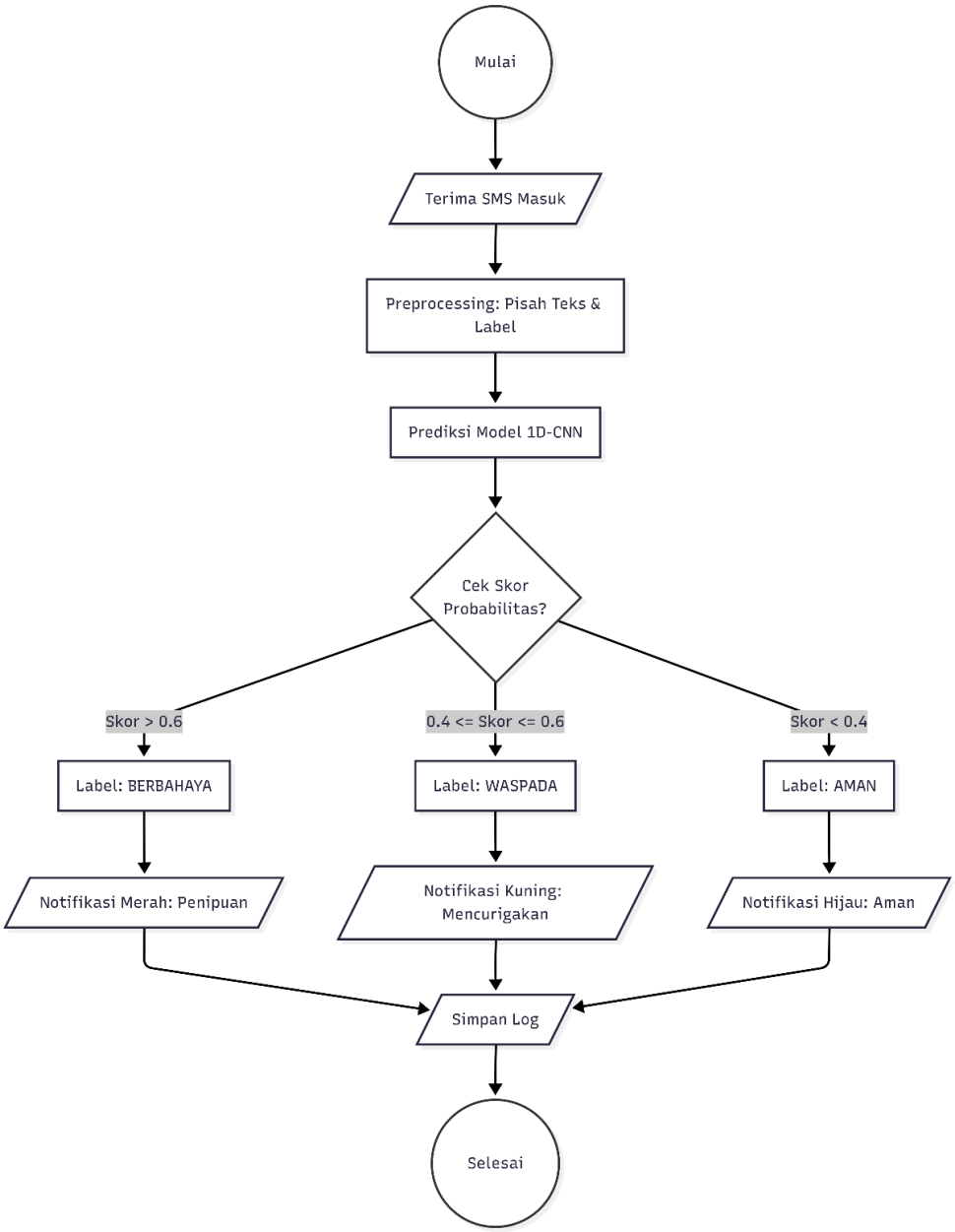
(*parsing*) secara otomatis menjadi variabel input dan variabel target oleh algoritma pada proses *preprocessing* sebelum memasuki fase *model training*.

2. Diagram UML (*Unified Modeling Language*)

Perancangan arsitektur fungsional perangkat lunak direpresentasikan menggunakan standar *Unified Modeling Language* (UML) yang meliputi Diagram *Use Case* dan *Activity Diagram*/Diagram Aktivitas.

Gambar 3.2 Diagram *Use Case* Aplikasi

Diagram *Use Case* bertujuan untuk menampilkan interaksi antara pengguna dan fitur-fitur aplikasi, di mana pengguna bisa memantau daftar kotak masuk, melihat rincian laporan pesan yang terdeteksi, serta memiliki kendali utama untuk menyalakan atau mematikan fitur perlindungan otomatis. Diagram ini membantu mendefinisikan batasan sistem serta hak akses pengguna saat menjalankan aplikasi sehari-hari.



Gambar 3.3 *Activity Diagram*

Di sisi lain, Diagram Aktivitas dibuat untuk menggambarkan jalur logika sistem yang beroperasi di latar belakang (*background service*) saat menerima pesan baru. Jalur ini menggunakan mekanisme keputusan bertingkat yang diperoleh dari skor probabilitas yang dihasilkan oleh model 1D-CNN. Proses klasifikasi dibagi menjadi tiga zona ambang batas (*threshold*) untuk meningkatkan akurasi peringatan: pesan dengan skor prediksi di bawah 0. 4 dianggap sebagai Aman (Normal), pesan dengan skor antara 0. 4 hingga 0. 6 diberi label Mencurigakan (Waspada), dan pesan dengan skor lebih dari 0. 6 dikategorikan sebagai Penipuan (Berbahaya). Jalur ini memastikan bahwa pengguna menerima informasi risiko yang sesuai untuk setiap pesan yang diterima.

3.3.2. Simulasi Perhitungan Manual

Proses pengenalan pola kalimat pada algoritma 1D-Convolutional Neural Network (1D-CNN) dirancang melalui operasi matematika konvolusi satu dimensi. Persamaan matematis untuk ekstraksi fitur pada lapisan konvolusi dirumuskan sebagai berikut:

$$Hasil = (Input_1 \times Kernel_1) + (Input_2 \times Kernel_2) + Bias \quad (1)$$

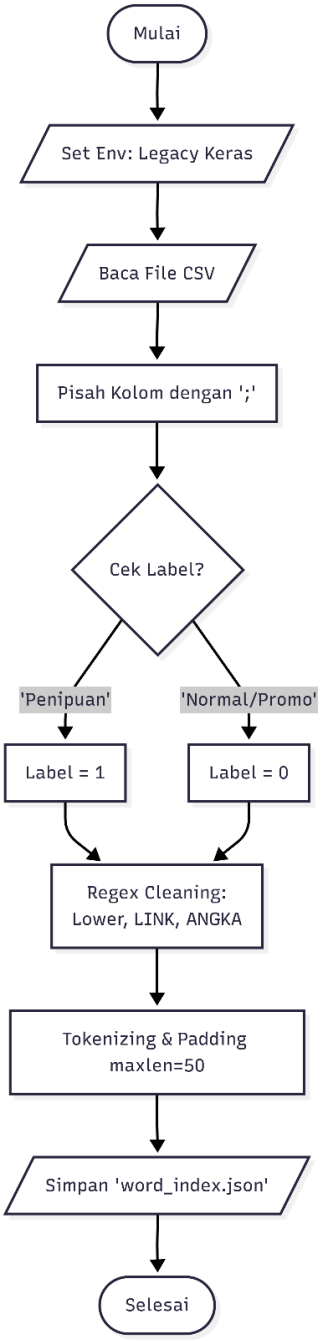
Setelah nilai konvolusi didapatkan, rancangan sistem akan memproses nilai tersebut menggunakan fungsi aktivasi *Rectified Linear Unit* (ReLU) untuk mengabaikan nilai linearitas negatif. Rumus fungsi aktivasi ReLU ditetapkan sebagai:

$$f(x) = \max(0, x)$$

Rancangan rumus-rumus matematika inilah yang akan ditanamkan ke dalam logika program untuk menghitung probabilitas apakah sebuah SMS termasuk penipuan atau normal.

3.3.3. Alur Sistem

1. Alur Sistem Preprocessing Data

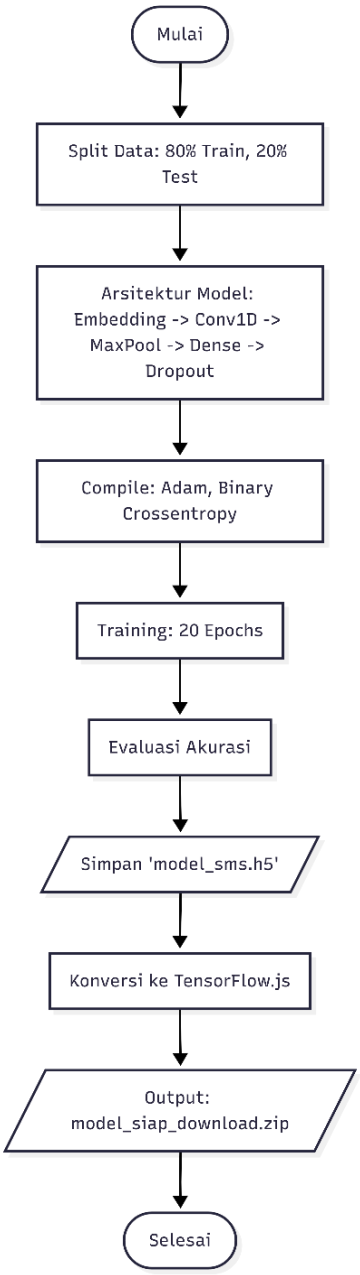


Gambar 3.4 Alur Sistem *Preprocessing*

Algoritma pra-pemrosesan dimulai dengan inisialisasi lingkungan TensorFlow ke mode Legacy dan pembacaan dataset, dilanjutkan dengan normalisasi label menjadi nilai biner (1 untuk penipuan, 0 untuk normal). Proses

pembersihan teks dilakukan menggunakan Regular Expression (Regex) yang meliputi pengubahan huruf kecil (lowercase), penggantian pola URL dan angka menjadi token khusus ([LINK]/[ANGKA]), serta penghapusan simbol non-alfanumerik. Data teks kemudian dikonversi menjadi urutan angka (sequences) melalui proses tokenisasi dengan penerapan padding hingga panjang 50 kata, diakhiri dengan penyimpanan kamus kata ke fail word_index.json.

2. Alur Pelatihan Model (1D-CNN)

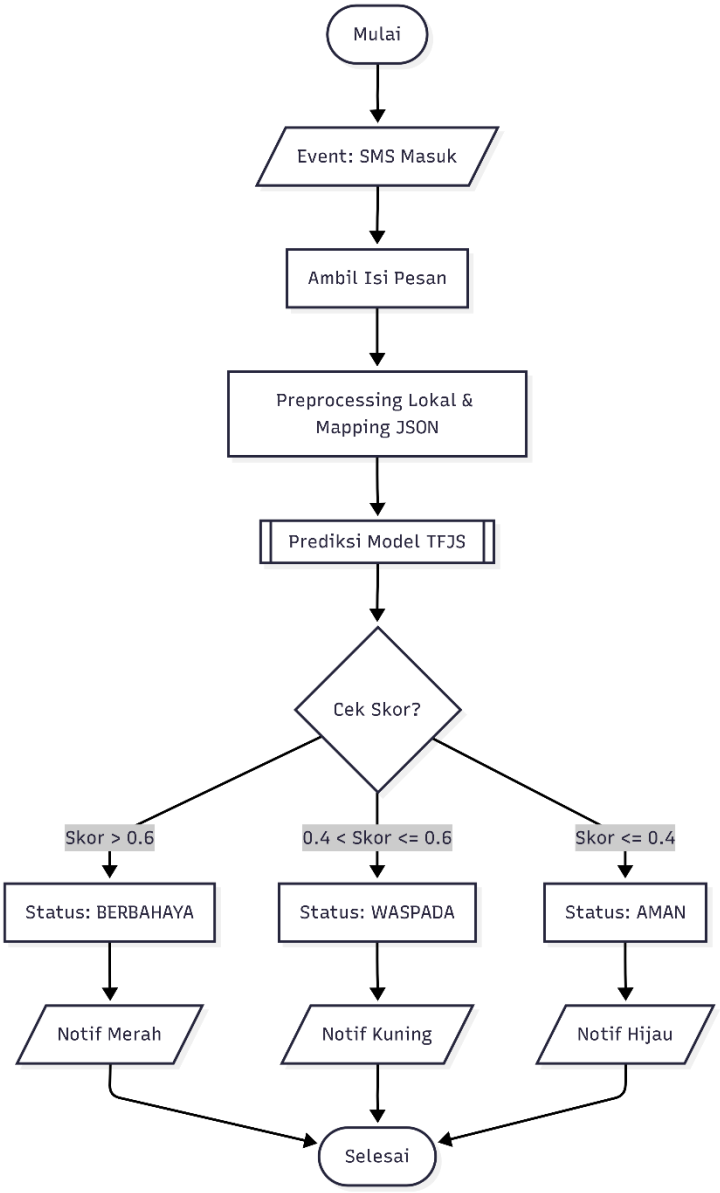


Gambar 3.5 Alur Pelatihan Model

Proses pelatihan model merupakan inti dari sistem kecerdasan buatan ini. Algoritma dimulai dengan membagi data hasil *preprocessing* menjadi data latih (80%) dan data uji (20%). Arsitektur model dibangun menggunakan metode *Sequential* dengan lapisan-lapisan spesifik seperti *Embedding*, *Conv1D* untuk

ekstraksi fitur, *GlobalMaxPooling*, dan *Dropout* untuk mencegah *overfitting*. Setelah pelatihan selesai dilakukan selama 20 *epoch*, model disimpan dalam format .h5 dan dikonversi menjadi format TensorFlow.js agar dapat dijalankan pada perangkat *mobile*.

3. Alur Sistem Aplikasi Android



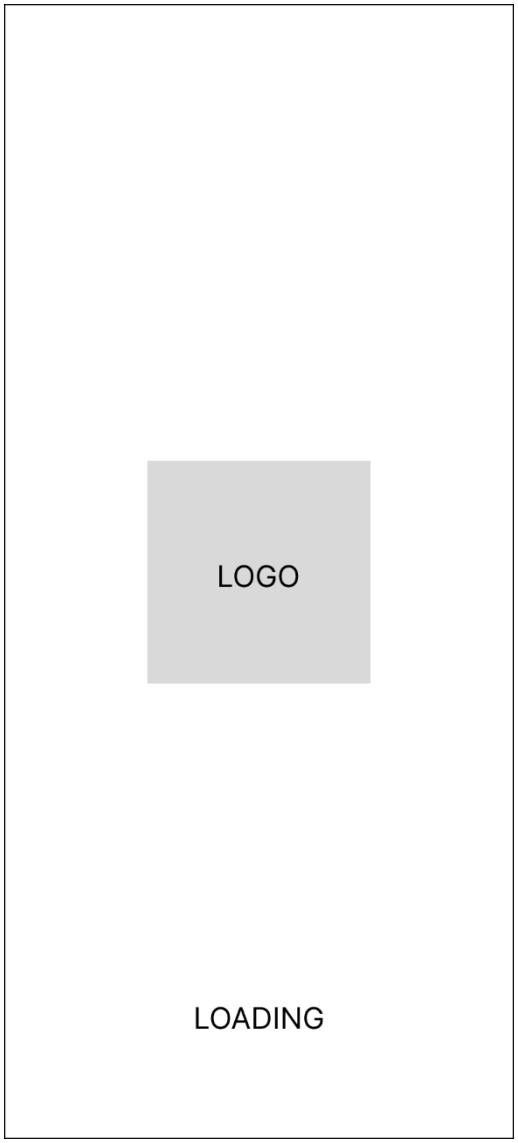
Gambar 3.6 Alur Sistem Aplikasi

Pada sisi perangkat pengguna, algoritma aplikasi dirancang untuk berjalan di latar belakang (*background service*) guna memantau pesan masuk secara *real-time*. Ketika SMS diterima, aplikasi melakukan pra-pemrosesan lokal yang serupa dengan tahap pelatihan (menggunakan word_index.json), lalu mengirimkan data tersebut ke model TensorFlow.js yang telah ditanam. Hasil prediksi berupa skor probabilitas kemudian dianalisis menggunakan logika ambang batas bertingkat (Aman, Waspada, Berbahaya) untuk menentukan jenis notifikasi yang akan ditampilkan.

3.4. Rancangan Layar (*User Interface)*

3.4.1. Tampilan *Splash Screen*

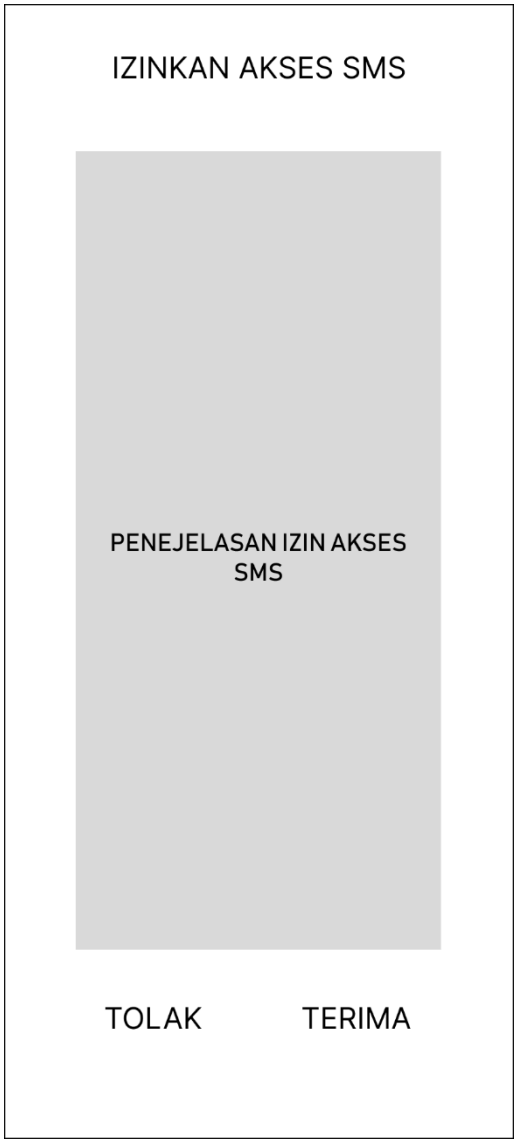
Aplikasi diawali dengan tampilan *Splash Screen* yang menyajikan identitas visual berupa logo aplikasi yang diletakkan secara sentral di tengah layar. Pada bagian bawah, terdapat indikator animasi (*loading bar*) yang menginformasikan kepada pengguna bahwa sistem sedang memuat model kecerdasan buatan dan menyiapkan sumber daya yang diperlukan sebelum masuk ke menu utama.



Gambar 3.7 Rancangan *Splash Screen*

3.4.2. Tampilan *Onboarding* & Perizinan

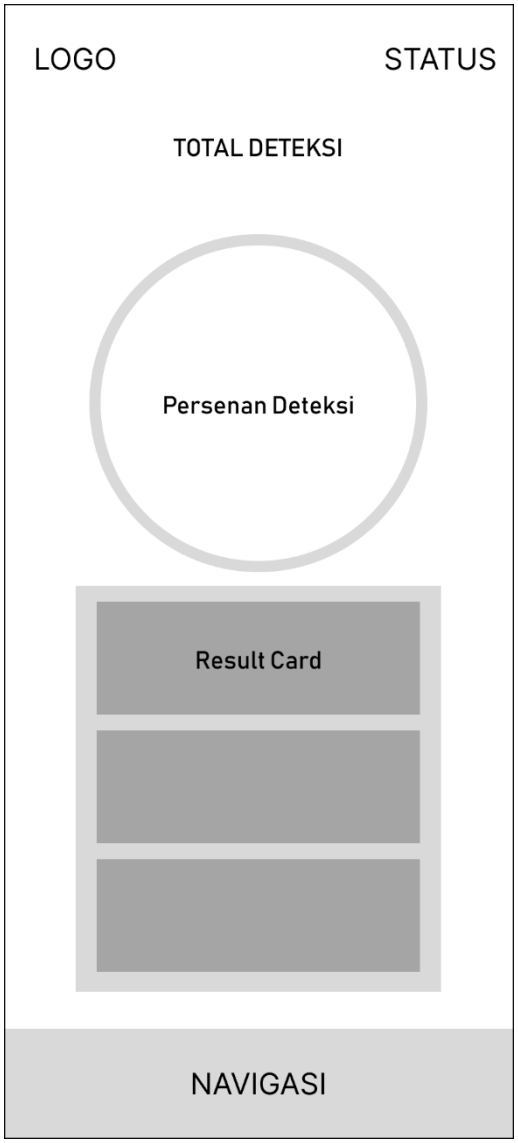
Halaman *Onboarding* dirancang untuk aspek transparansi privasi, di mana aplikasi secara eksplisit meminta izin akses pembacaan SMS (*Read SMS Permission*) kepada pengguna. Halaman ini menyertakan teks penjelasan yang edukatif mengenai alasan teknis mengapa izin tersebut dibutuhkan, memastikan pengguna memahami bahwa akses hanya digunakan untuk tujuan deteksi keamanan.



Gambar 3.8 Rancangan *Onboarding Screen*

3.4.3. Tampilan Beranda (*Home*)

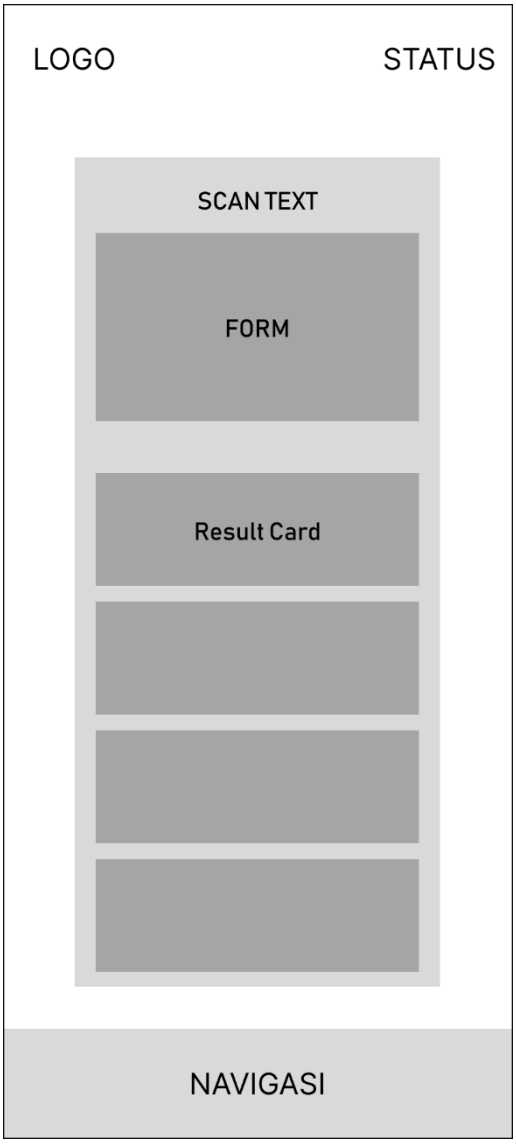
Halaman Beranda berfungsi sebagai dasbor utama yang menampilkan statistik keamanan melalui diagram lingkaran dan rincian jumlah pesan berdasarkan tiga kategori: Penipuan, Mencurigakan, dan Aman. Di bawah panel statistik, terdapat area aktivitas yang memuat daftar kartu hasil pemindaian terakhir, memberikan gambaran cepat mengenai status keamanan kotak masuk pengguna saat ini.



Gambar 3.9 Rancangan *Home Screen*

3.4.4. Tampilan Pindai Teks Manual (*Scan Text*)

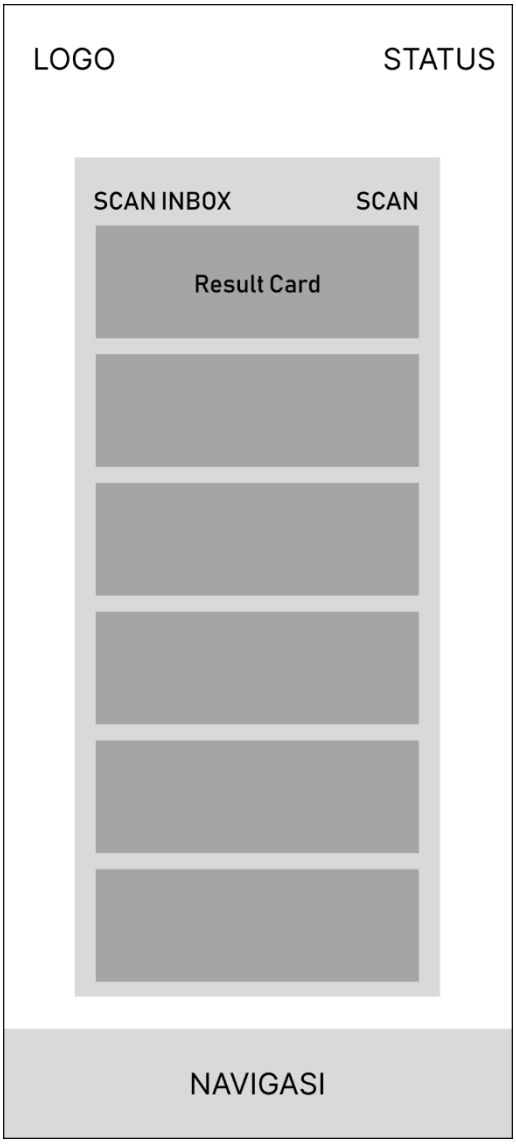
Fitur *Scan Text* memfasilitasi pengguna untuk melakukan pemeriksaan pesan secara manual dengan menyediakan kolom input teks yang luas untuk menempelkan (*paste*) pesan yang dicurigai. Setelah proses pemindaian selesai, hasil analisis akan muncul di bagian bawah dalam bentuk kartu informasi yang memuat skor probabilitas dan label kategori pesan tersebut.



Gambar 3.10 Rancangan *Scan Text Screen*

3.4.5. Tampilan Pindai Kotak Masuk (*Scan Inbox*)

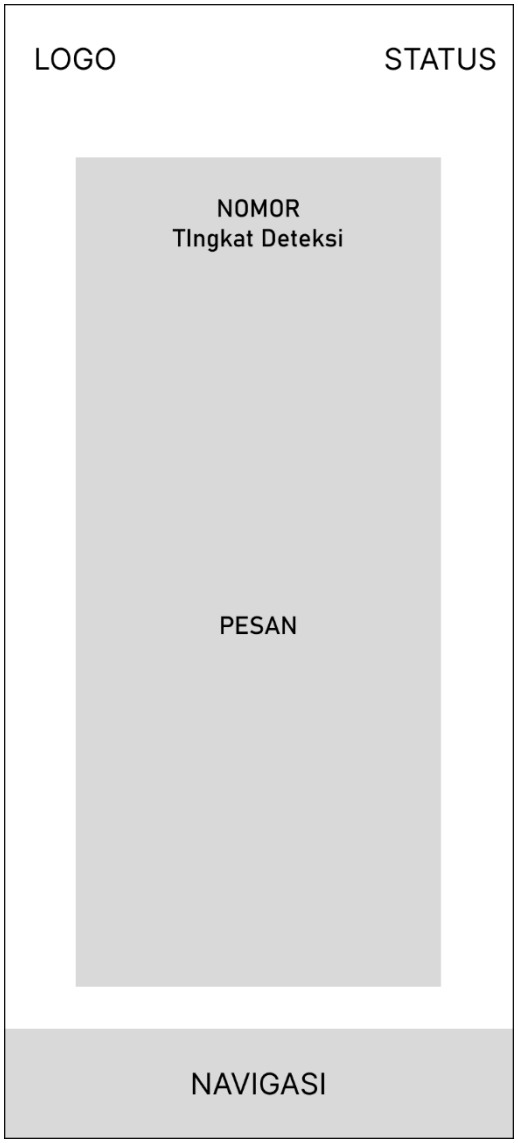
Halaman *Scan Inbox* didedikasikan untuk melakukan analisis massal terhadap seluruh pesan yang tersimpan di dalam perangkat pengguna. Halaman ini dilengkapi tombol aktivasi pemindaian yang, saat ditekan, akan menarik data pesan masuk dan menampilkannya dalam daftar kartu hasil deteksi secara berurutan sesuai tingkat risikonya.



Gambar 3.11 Rancangan *Scan Inbox Screen*

3.4.6. Tampilan Detail Hasil Deteksi (*Detail Result*)

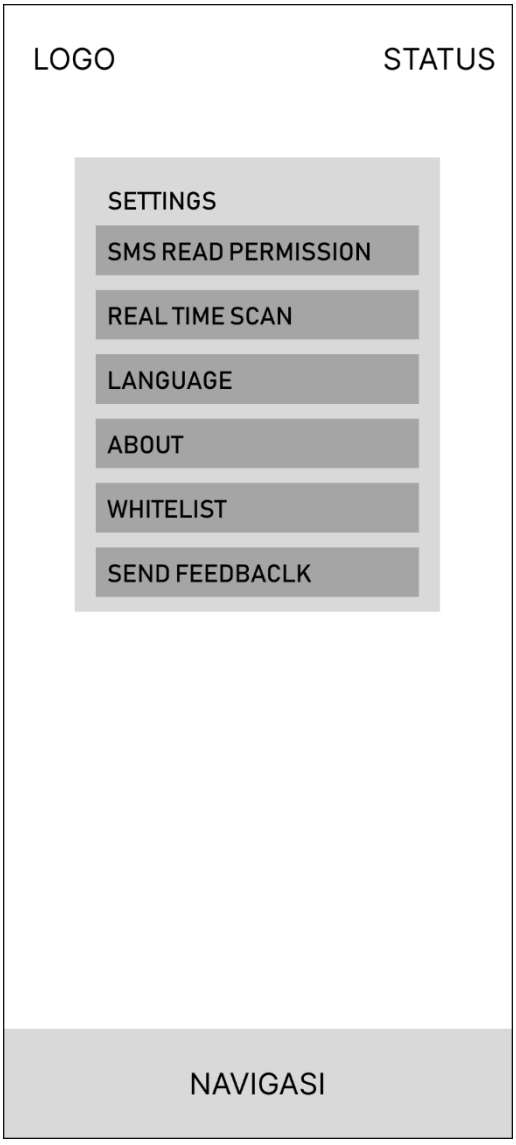
Halaman Detail Hasil dirancang untuk menyajikan transparansi analisis yang lebih mendalam terhadap pesan yang dipilih. Tampilan ini didominasi oleh sebuah kontainer informasi yang memuat indikator visual berupa batang persentase (*percentage bar*) untuk merepresentasikan tingkat probabilitas penipuan secara presisi, serta menampilkan seluruh isi teks pesan secara utuh agar pengguna dapat membaca kembali konteks kalimat lengkap yang memicu deteksi sistem.



Gambar 3.12 Rancangan *Detail Screen*

3.4.7. Tampilan Pengaturan (*Settings*)

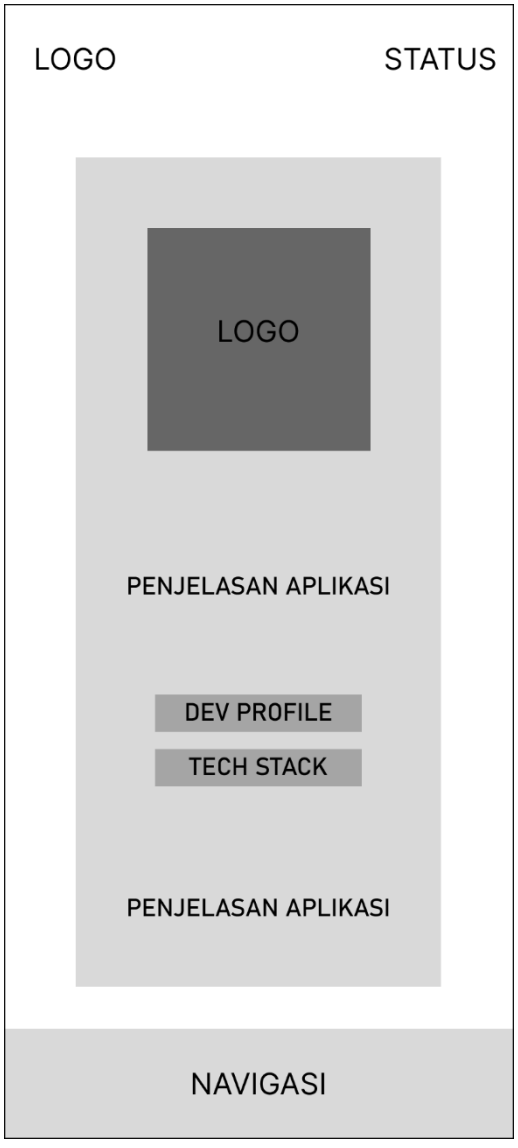
Halaman Pengaturan memberikan kendali penuh kepada pengguna untuk mengelola preferensi aplikasi, meliputi manajemen izin akses SMS, opsi bahasa, serta sakelar untuk mengaktifkan atau mematikan fitur pemindaian *real-time*. Halaman ini disusun dalam bentuk daftar menu vertikal yang juga menyertakan navigasi menuju fitur bantuan dan informasi aplikasi.



Gambar 3.13 Rancangan *Settings Screen*

3.4.8. Tampilan Tentang Aplikasi (*About*)

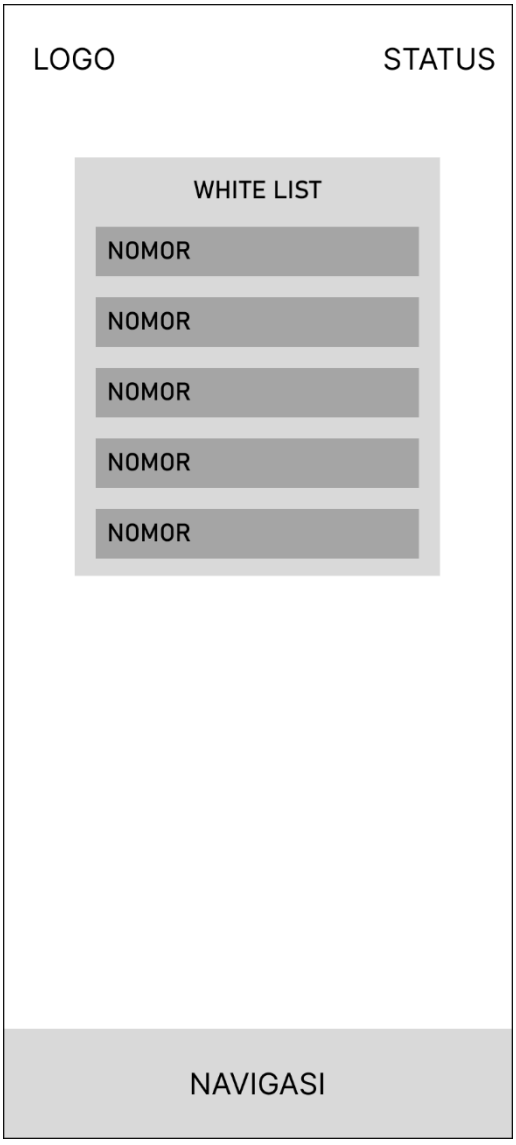
Sub-menu *About* menyajikan informasi detail mengenai versi aplikasi yang sedang berjalan serta profil pengembang sistem. Halaman ini berfungsi sebagai media transparansi dan informasi kontak, memberikan jaminan kredibilitas perangkat lunak kepada pengguna akhir.



Gambar 3.14 Rancangan *About Screen*

3.4.9. Tampilan Daftar Putih (*Whitelist*)

Fitur *Whitelist* menampilkan daftar nomor kontak atau pengirim yang telah ditandai sebagai entitas terpercaya oleh pengguna secara manual. Sistem dirancang untuk mengecualikan pesan yang berasal dari nomor-nomor yang terdaftar pada halaman ini dari pemindaian ketat, guna mencegah terjadinya kesalahan deteksi pada pesan penting.



Gambar 3.15 Rancangan *Whitelist Screen*

3.5. Perancangan Pengujian

Tahapan pengujian merupakan fase kritis dalam pengembangan sistem untuk memvalidasi bahwa perangkat lunak yang dibangun telah memenuhi spesifikasi kebutuhan, baik dari segi keandalan kecerdasan buatan maupun stabilitas fungsi aplikasi pada perangkat pengguna. Dalam penelitian ini, strategi pengujian dibagi menjadi dua metode pendekatan utama, yaitu *White Box Testing* untuk mengukur performa internal model dan *Black Box Testing* untuk memvalidasi fungsionalitas eksternal aplikasi.

3.5.1. Pengujian Kinerja Model (*White Box*)

Pengujian kinerja model bertujuan untuk mengukur sejauh mana algoritma *1D-Convolutional Neural Network* (1D-CNN) mampu mempelajari pola teks dan mengklasifikasikan pesan dengan tepat. Prosedur evaluasi dilakukan dengan membandingkan hasil prediksi model terhadap label kebenaran dasar (*ground truth*) menggunakan data uji (*test set*) yang dialokasikan sebesar 20% dari total dataset. Parameter pengukuran kinerja merujuk sepenuhnya pada metrik evaluasi yang telah dipaparkan landasan teorinya pada Bab II (Sub-bab 2.8), yang meliputi perhitungan nilai Akurasi, Presisi, dan Recall.

Penetapan standar keberhasilan model dalam penelitian ini mengacu pada ambang batas akurasi minimal 90% serta nilai presisi yang tinggi untuk meminimalkan kesalahan deteksi positif palsu (*False Positive*). Apabila hasil evaluasi awal menunjukkan performa di bawah target tersebut, maka akan dilakukan proses penyetelan ulang parameter (*Hyperparameter Tuning*) pada arsitektur model, seperti penyesuaian *learning rate* atau jumlah *epoch*, hingga

model mencapai tingkat stabilitas yang diharapkan sebelum diimplementasikan ke dalam aplikasi *mobile*.

3.5.2. Pengujian Fungsional Aplikasi (Black Box)

Pengujian fungsional atau *Black Box Testing* difokuskan pada validasi antarmuka dan logika bisnis aplikasi tanpa meninjau struktur kode internal program. Tujuannya adalah untuk memastikan bahwa seluruh fitur utama, mulai dari mekanisme izin akses, pemindaian pesan, hingga tampilan notifikasi, berjalan sesuai dengan skenario rancangan. Tabel berikut merincikan rencana pengujian pada fitur-fitur krusial sistem:

Tabel 3.1 Pengujian Fungsional Aplikasi

No	Skenario Pengujian	Kasus Uji (Input)	Hasil yang Diharapkan
1	**Deteksi Pesan Penipuan**	Sistem menerima SMS baru yang berisi kata kunci penipuan (Contoh: "Selamat Anda Menang...").	Aplikasi mampu mendeteksi konten sebagai "BERBAHAYA" dan segera memunculkan notifikasi peringatan berwarna Merah.
2	**Deteksi Pesan Normal**	Sistem menerima SMS percakapan biasa atau promosi resmi (Contoh: "Besok kita kumpul jam 9").	Aplikasi mengklasifikasikan pesan sebagai "AMAN" dan menampilkan notifikasi Hijau atau tidak mengganggu pengguna.
3	**Fitur Pindai Manual**	Pengguna menempelkan (*paste*) teks ke dalam kolom	Sistem memproses teks dan menampilkan kartu hasil

No	Skenario Pengujian	Kasus Uji (Input)	Hasil yang Diharapkan
		input pada menu *Scan Text* dan menekan tombol *Scan*.	(*result card*) yang berisi skor probabilitas serta label kategori yang sesuai.
4	**Realtime Scanning**	Sistem melakukan pemindaian Ketika pesan masuk secara langsung.	Pesan terdeteksi dan hasil scan masuk ke dalam aplikasi.
5	**Mekanisme Perizinan**	Pengguna membuka aplikasi untuk pertama kalinya setelah instalasi.	Aplikasi menampilkan *pop-up* sistem yang meminta izin akses "Read SMS" sebelum pengguna dapat masuk ke menu utama.

3.6. Jadwal Penelitian

Tabel berikut merincikan jadwal pelaksanaan penelitian yang direncanakan:

Tabel 3.2 Jadwal Penilitian

No	Kegiatan	Waktu				
		1	2	3	4	5
1	Penulisan proposal	✓				
2	Seminar dan bimbingan proposal		✓			
3	Penelitian dan tindakan			✓		
4	Analisis dan bimbingan hasil penelitian			✓	✓	
5	Ujian skripsi					✓

BAB IV

HASIL DAN PEMBAHASAN

4.1. Hasil Implementasi Sistem

Implementasi sistem merupakan tahap realisasi dari seluruh perancangan yang telah diuraikan pada bab metodologi. Pada fase ini, gagasan konseptual dan struktur algoritma diterjemahkan ke dalam bentuk produk komputasi yang nyata. Hasil akhir dari tahapan ini mencakup dua elemen utama yang terintegrasi, yakni model kecerdasan buatan yang bertugas sebagai mesin klasifikasi teks, serta aplikasi perangkat bergerak berbasis Android yang berfungsi sebagai wadah interaksi bagi pengguna akhir.

4.1.1. Implementasi Pra-pemrosesan Data

Tahap pertama dalam implementasi ini adalah penyiapan data tekstual agar dapat dipahami oleh mesin. Langkah ini sejalan dengan penelitian (Kristananda et al., 2025) yang menyatakan bahwa proses pengumpulan data melibatkan pengambilan sampel acak yang kemudian dipersiapkan dengan tahapan preprocessing yang meliputi pembersihan teks dan tokenisasi untuk memastikan keandalan masukan. Proses ini berhasil menghimpun dataset dengan total keseluruhan mencapai 1611 baris pesan teks, yang secara spesifik terdiri dari 920 pesan berlabel normal dan 671 pesan berlabel penipuan. Kumpulan data mentah tersebut kemudian diproses melalui serangkaian teknik pembersihan karakter, konversi seluruh huruf menjadi huruf kecil (*case folding*), dan pemecahan kalimat menjadi token kata. Hasil akhir dari proses penyaringan ini adalah sebuah kamus data (*dictionary*) yang disimpan dalam format berkas word_index.json. Berkas ini berperan sangat penting karena memetakan setiap kata unik ke dalam representasi

angka numerik, yang merupakan format wajib sebelum data diumpankan ke dalam model komputasi.

4.1.2. Implementasi Model 1D-CNN

Pengembangan mesin kecerdasan buatan direalisasikan melalui pelatihan arsitektur *One-Dimensional Convolutional Neural Network* (1D-CNN). Proses pembelajaran mesin ini membagi dataset yang telah bersih menjadi dua bagian utama, yaitu 80% dialokasikan sebagai data latih dan 20% sisanya sebagai data uji untuk proses validasi. Pelatihan algoritma dieksekusi sebanyak 100 siklus (*epoch*) guna memastikan model mampu mengenali dan mengekstraksi fitur pola kalimat penipuan secara maksimal. Pendekatan ini mengadopsi prinsip yang dikemukakan oleh (Ahmadi et al., 2025) bahwa pembelajaran mendalam telah menjadi kumpulan teknik yang kuat untuk mengatasi masalah rumit seperti klasifikasi spam melalui ekstraksi fitur secara otomatis. Setelah mencapai tingkat konvergensi yang optimal, model tersebut kemudian dikonversi menggunakan pustaka TensorFlow.js. Proses konversi ini menghasilkan dua berkas utama, yaitu model.json yang memuat struktur arsitektur jaringan, dan shard.bin yang menyimpan bobot parameter. Kedua berkas ini memungkinkan model ditanamkan langsung secara lokal di dalam aplikasi (*client-side*), sehingga proses deteksi dapat berjalan secara mandiri tanpa harus bergantung pada koneksi peladen eksternal.

4.1.3. Implementasi Simulasi Perhitungan Manual 1D-CNN

Untuk membuktikan rancangan rumus matematis yang telah ditetapkan pada Bab III, berikut adalah implementasi perhitungan logika kerja 1D-CNN pada sampel data sederhana. Sebagai studi kasus, sistem menerima input frasa pesan "Menang Hadiah".

Tahap pertama adalah vektorisasi, di mana teks dikonversi menjadi representasi numerik oleh sistem. Diasumsikan kata "Menang" memiliki nilai vektor 0.8 dan kata "Hadiah" bernilai 0.5, sehingga terbentuk matriks input [0.8, 0.5]. Selanjutnya, model menggunakan *filter* (kernel) dengan bobot [0.2, 0.4] dan nilai bias sebesar 0.1. Implementasi operasi konvolusinya adalah perkalian *dot* product sebagai berikut:

$$\begin{aligned} Hasil &= (Input_1 \times Kernel_1) + (Input_2 \times Kernel_2) + Bias \\ Hasil &= (0.8 \times 0.2) + (0.5 \times 0.4) + 0.1 \\ Hasil &= 0.16 + 0.20 + 0.1 = 0.46 \end{aligned} \quad (1)$$

Nilai 0.46 tersebut kemudian dilewatkan pada fungsi aktivasi ReLU. Karena 0.46>0, maka nilai dipertahankan menjadi 0.46. Nilai hasil implementasi perhitungan inilah yang menjadi fitur acuan model dalam menentukan status pesan.

4.1.4. Implementasi Arsitektur Frontend Menggunakan React Native

Implementasi antarmuka pengguna (UI) dibangun sepenuhnya menggunakan *framework* React Native. Meskipun beroperasi pada platform Android, pendekatan pemrograman React Native mengadopsi logika pengembangan berbasis komponen (*Component-Based Architecture*) yang serupa dengan pengembangan *website* modern menggunakan React.js.

Beberapa implementasi teknis utama dalam pengembangan antarmuka ini meliputi:

1. **Penggunaan *Functional Component* dan *Hooks*:** Pembuatan halaman aplikasi diimplementasikan menggunakan fungsi JavaScript. Pengelolaan status data, seperti menyimpan isi pesan SMS yang masuk atau status pemuatan (*loading*), dikelola menggunakan *hooks* useState dan useEffect.

2. **Sistem Tata Letak (Layouting) berbasis *Flexbox*:** Untuk menyusun elemen antarmuka (seperti tombol, teks, dan kartu hasil), aplikasi menggunakan StyleSheet bawaan React Native yang mengadopsi properti CSS *Flexbox*. Hal ini memastikan tampilan antarmuka selalu responsif dan proporsional di berbagai ukuran layar perangkat *smartphone* yang berbeda.
3. **Navigasi (*Routing*):** Perpindahan antar layar aplikasi (misalnya dari halaman Kotak Masuk ke halaman Detail Hasil) diimplementasikan menggunakan pustaka React Navigation, yang memberikan transisi tumpukan halaman (*stack*) yang mulus layaknya aplikasi *native*.

4.2. Implementasi Antarmuka Aplikasi

Implementasi antarmuka aplikasi merupakan tahap penerjemahan rancangan visual yang telah ditetapkan pada Bab III ke dalam bentuk produk perangkat lunak yang fungsional. Pengembangan antarmuka ini direalisasikan menggunakan kerangka kerja React Native. Bagian ini menjabarkan realisasi dari setiap halaman yang akan berinteraksi langsung dengan pengguna akhir, memastikan kemudahan navigasi dan penyampaian informasi keamanan secara komprehensif.

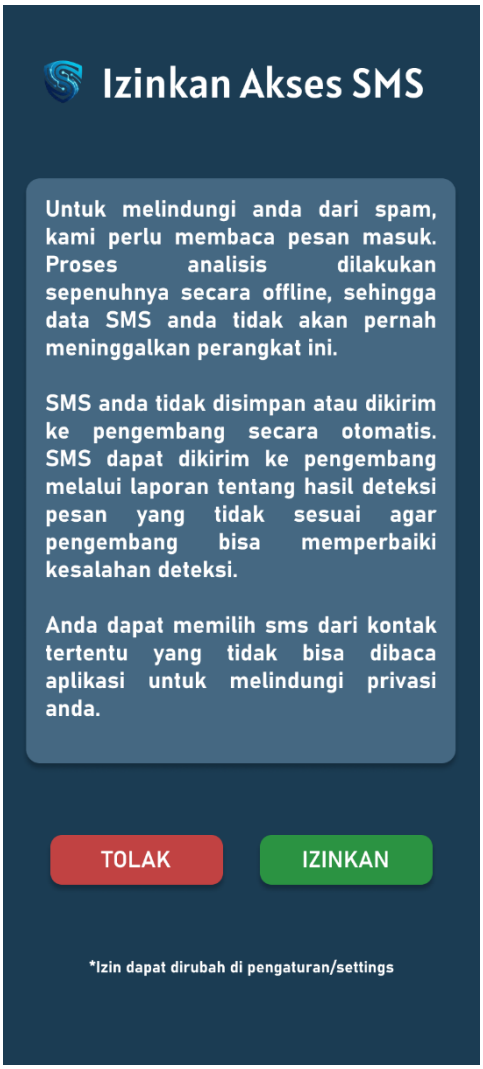
4.2.1. Implementasi Tampilan *Splash Screen*

Tahap pertama dari interaksi pengguna direalisasikan melalui implementasi antarmuka *Splash Screen*. Halaman pembuka ini berhasil menampilkan identitas visual berupa logo aplikasi yang diposisikan secara presisi di tengah layar. Selain sebagai elemen perkenalan, antarmuka ini secara fungsional memuat indikator animasi yang menandakan bahwa sistem sedang beroperasi di latar belakang untuk melakukan pemuatan model kecerdasan buatan ke dalam memori perangkat sebelum pengguna diarahkan sepenuhnya ke menu utama.

Gambar 4.1 *Splash Screen*

4.2.2. Implementasi Tampilan *Onboarding* & Perizinan

Implementasi halaman *Onboarding* direalisasikan sebagai langkah pemenuhan standar transparansi privasi dan keamanan data. Antarmuka ini dirancang secara khusus untuk meminta izin akses pembacaan pesan singkat secara eksplisit kepada pengguna. Guna membangun kepercayaan, halaman tersebut dilengkapi dengan teks penjelasan informatif yang memastikan pengguna memahami bahwa izin akses yang diberikan murni dialokasikan untuk kebutuhan proses deteksi ancaman keamanan tanpa menyalahgunakan privasi.



Gambar 4.2 *Onboarding Screen*

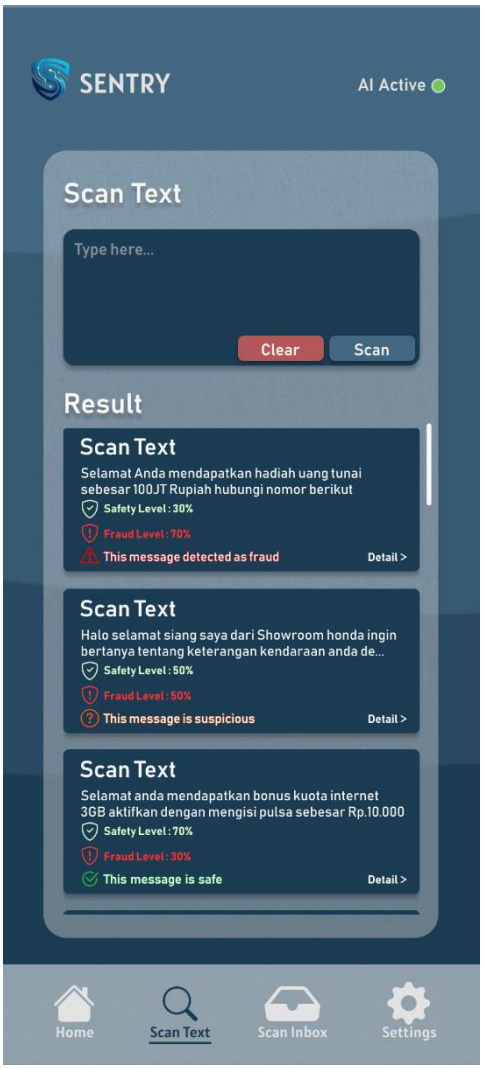
4.2.3. Implementasi Tampilan Beranda (*Home*)

Sebagai pusat kendali informasi, halaman Beranda telah berhasil diimplementasikan untuk berfungsi secara penuh sebagai dasbor utama aplikasi. Antarmuka ini menyajikan ringkasan statistik keamanan kotak masuk secara waktu nyata yang divisualisasikan melalui sebuah diagram lingkaran yang intuitif. Pada bagian bawah halaman, sistem menampilkan daftar kartu hasil pemindaian terbaru, sehingga pengguna dapat memperoleh informasi cepat dan akurat mengenai status keamanan pesan terakhir yang diterima.

Gambar 4.3 *Home Screen*

4.2.4. Implementasi Tampilan Pindai Teks Manual (*Scan Text*)

Fungsionalitas pemeriksaan pesan secara mandiri diwujudkan melalui implementasi halaman *Scan Text*. Antarmuka ini menyediakan kolom masukan teks yang cukup luas, memungkinkan pengguna untuk menyalin dan menempelkan teks pesan yang dicurigai secara langsung. Setelah pemrosesan selesai, hasil analisis komputasi langsung dimunculkan pada bagian bawah layar dalam bentuk kartu informasi terstruktur, yang memuat perincian label kategori beserta skor probabilitas tingkat ancaman dari teks tersebut.



Gambar 4.4 *Scan Text Screen*

4.2.5. Implementasi Tampilan Pindai Kotak Masuk (*Scan Inbox*)

Untuk memfasilitasi evaluasi keamanan secara menyeluruh, halaman *Scan Inbox* diimplementasikan dengan sebuah tombol aktivasi utama. Ketika instruksi eksekusi diberikan oleh pengguna, sistem secara otomatis menarik data dari kotak masuk perangkat dan melakukan analisis secara massal. Pesan-pesan yang telah diproses kemudian disajikan ke dalam antarmuka melalui bentuk daftar kartu hasil deteksi yang tersusun rapi berdasarkan tingkat risikonya, memudahkan pengguna dalam meninjau potensi ancaman.

Gambar 4.5 *Scan Inbox Screen*

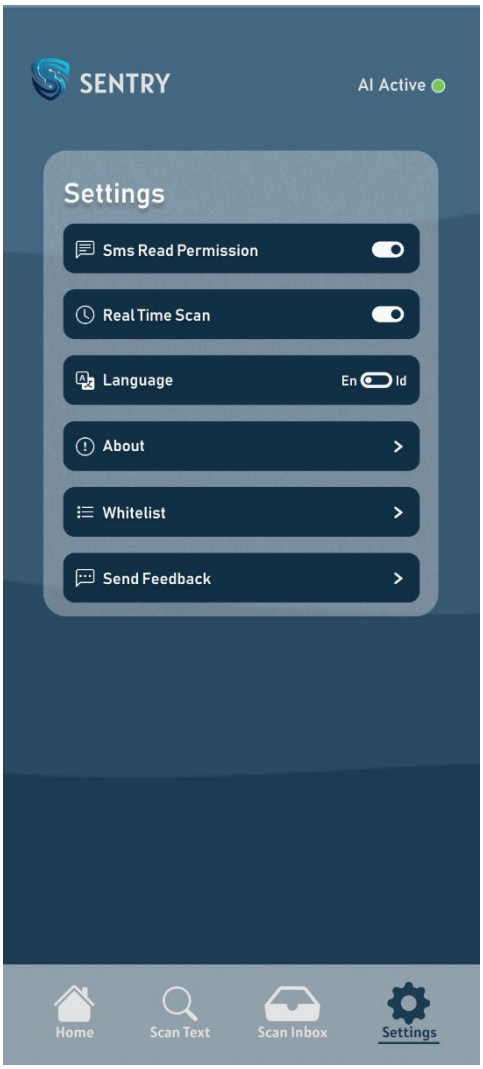
4.2.6. Implementasi Tampilan Detail Hasil Deteksi (*Detail Result*)

Prinsip transparansi analitik direalisasikan secara konkret melalui implementasi halaman *Detail Result*. Antarmuka ini dirancang untuk menyajikan rincian hasil deteksi secara lebih mendalam, salah satunya melalui indikator visual berupa batang persentase yang merepresentasikan metrik probabilitas penipuan. Selain itu, halaman ini juga menampilkan keseluruhan teks pesan secara utuh dan pengguna dapat melaporkan jika hasil pesan tidak sesuai serta menghapus hasil *scan*.

Gambar 4.6 *Detail Screen*

4.2.7. Implementasi Tampilan Pengaturan (*Settings*)

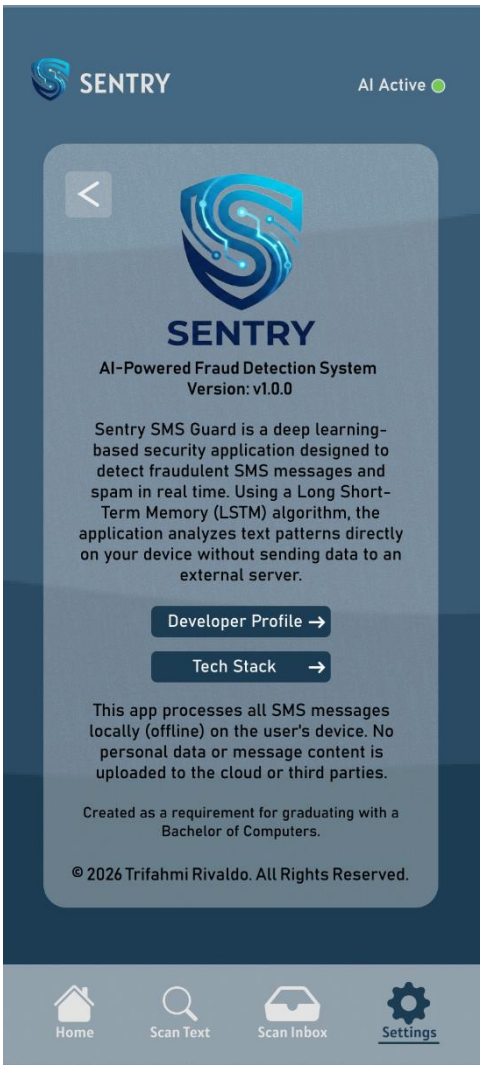
Guna memberikan keleluasaan dalam personalisasi sistem, halaman Pengaturan diimplementasikan dalam bentuk daftar menu vertikal yang terstruktur. Melalui antarmuka ini, pengguna diberikan akses penuh untuk mengelola konfigurasi krusial, seperti menyesuaikan kembali perizinan akses pesan, mengubah opsi bahasa antarmuka, serta mengendalikan sakelar fungsional untuk menyalakan atau mematikan fitur pemindaian keamanan yang berjalan secara waktu nyata.



Gambar 4.7 *Settings Screen*

4.2.8. Implementasi Tampilan Tentang Aplikasi (*About*)

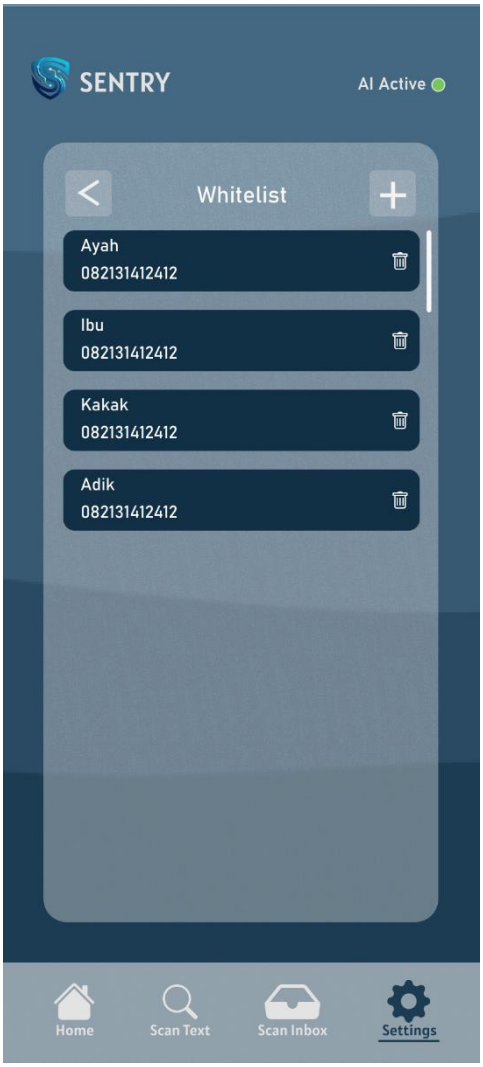
Sebagai bentuk pertanggungjawaban pengembangan perangkat lunak, antarmuka *About* diintegrasikan ke dalam sistem untuk menyajikan transparansi informasi. Halaman ini berfungsi memuat rincian spesifik mengenai versi rilis aplikasi yang sedang digunakan, serta menyediakan profil singkat dari pengembang. Implementasi halaman ini penting untuk memberikan jaminan kredibilitas dan mempermudah pengguna jika membutuhkan informasi lebih lanjut mengenai sistem yang dioperasikan.



Gambar 4.8 *About Screen*

4.2.9. Implementasi Tampilan Daftar Putih (*Whitelist*)

Fitur pengecualian keamanan berhasil diintegrasikan melalui implementasi antarmuka *Whitelist*. Halaman ini berfungsi untuk menampilkan serta mengelola daftar nomor kontak terpercaya yang telah ditentukan pengguna untuk dikecualikan dari proses pemindaian algoritma. Realisasi fitur ini memastikan bahwa pesan penting yang berasal dari kerabat dekat atau entitas resmi tidak akan salah diklasifikasikan sebagai ancaman oleh sistem deteksi.



Gambar 4.9 *Whitelist Screen*

4.3. Hasil Pengujian Sistem

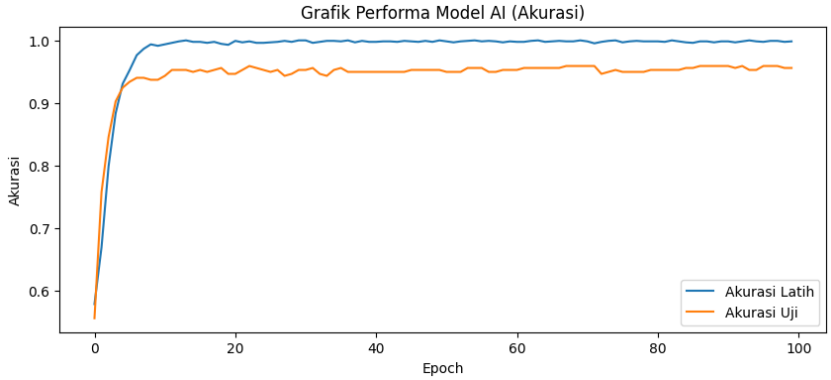
Fase pengujian ini dilaksanakan guna memvalidasi tingkat akurasi model kecerdasan buatan dalam mendeteksi ancaman penipuan, sekaligus memastikan bahwa seluruh fitur pada antarmuka aplikasi beroperasi secara optimal tanpa mengalami kegagalan fungsional. Evaluasi matriks kebingungan (*confusion matrix*) pada tahap ini sangat penting karena menurut (Yusup et al., 2026), nilai presisi dan *recall* yang dihasilkan merepresentasikan keseimbangan yang cukup baik untuk kasus klasifikasi teks dengan dataset yang tidak seimbang, yang menunjukkan bahwa model memiliki stabilitas yang memadai dalam mengenali pola kata berbahaya. Pengujian sistem ini dibagi menjadi dua aspek utama, yakni evaluasi kinerja internal model dan evaluasi fungsionalitas eksternal dari perangkat lunak yang dibangun.

4.3.1. Pengujian Kinerja Model (*White Box*)

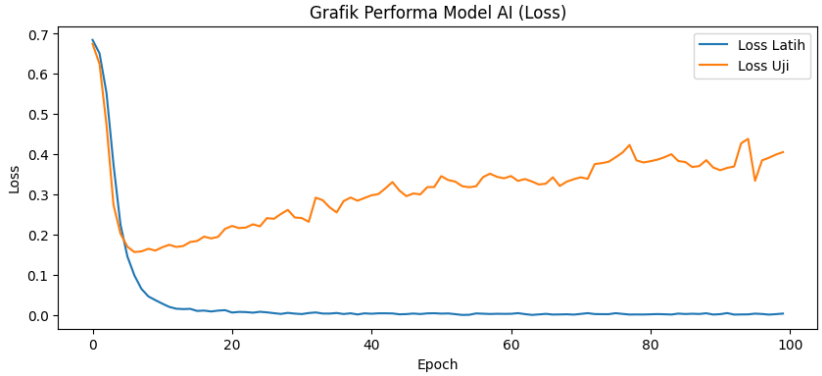
Pengujian model 1D-CNN dilakukan menggunakan platform Google Colab dengan mengalokasikan 20% data (298 baris teks) sebagai data uji. Model dilatih selama 100 *epoch*.

A. Grafik Performa Pelatihan (Akurasi dan Loss)

Berdasarkan proses *training*, model menunjukkan kurva pembelajaran yang sangat baik tanpa mengalami indikasi *overfitting* yang parah. Hal ini dibuktikan melalui grafik akurasi dan *loss* berikut:



Gambar 4.10 Grafik Performa Akurasi Latih dan Uji (100 Epoch)

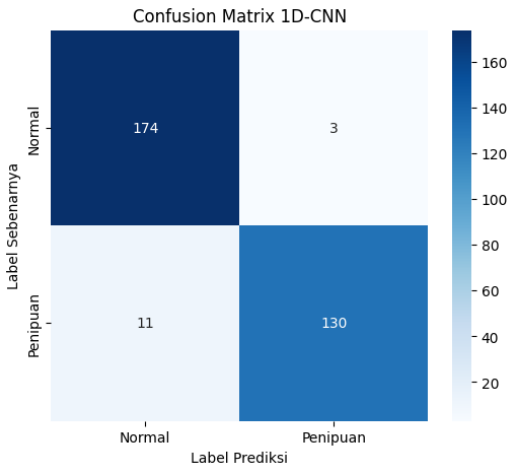


Gambar 4.11 Grafik Penurunan Loss (100 Epoch)

Pada grafik di atas, dapat dilihat bahwa garis validasi (*val_accuracy*) bergerak stabil mengimbangi akurasi latih, dan berhasil mencapai tingkat akurasi puncak sebesar **96,31%** dengan nilai validasi *loss* ditekan hingga **0.4695**.

B. Evaluasi *Confusion Matrix*

Untuk membedah rincian prediksi sistem, dilakukan pengujian *Confusion Matrix* yang menghasilkan pemetaan kelas True/False sebagai berikut:



Gambar 4.12 Visualisasi Confusion Matrix Model 1D-CNN

Dari visualisasi tersebut, model berhasil mengklasifikasikan secara tepat 169 pesan normal (True Negative) dan 117 pesan penipuan (True Positive). Kesalahan prediksi terjadi pada skala yang sangat kecil, yaitu 7 pesan (*False Positive*) dan 5 pesan (*False Negative*).

C. Laporan Klasifikasi (*Classification Report*)

Rincian angka metrik performa secara keseluruhan dijabarkan dalam tabel laporan klasifikasi yang diekstrak dari Google Colab berikut:

Tabel 4.1 Hasil Laporan Klasifikasi Model

Class	Precision	Recall	F1-Score	Support
Normal	0.94	0.98	0.96	177
Penipuan	0.98	0.92	0.95	141
Accuracy			**0.96**	**318**
Macro Avg	0.96	0.95	0.96	318
Weighted Avg	0.96	0.96	0.96	318

4.3.2. Pengujian Fungsional Aplikasi (*Black Box*)

Pengujian fungsionalitas kotak hitam dilakukan secara langsung pada perangkat fisik Android untuk memvalidasi kesesuaian antara interaksi pengguna dan respons yang diberikan oleh sistem. Berdasarkan skenario pengujian yang telah dirancang pada bab metodologi, evaluasi ini berfokus pada fitur-fitur utama aplikasi. Hasil pengujian menunjukkan bahwa seluruh fungsionalitas beroperasi dengan sangat baik tanpa adanya kegagalan sistem. Rincian hasil eksekusi pengujian tersebut dijabarkan sebagai berikut:

1. **Pengujian Deteksi Pesan Penipuan** Saat sistem menerima pesan singkat baru yang berisi kata kunci penipuan, seperti simulasi pengumuman hadiah, aplikasi terbukti mampu mendeteksi konten tersebut sebagai kategori "Penipuan". Sistem juga secara instan memunculkan notifikasi peringatan berwarna merah kepada pengguna. Hasil pengujian ini dinyatakan **Valid**.
2. **Pengujian Deteksi Pesan Normal** Ketika sistem diberikan masukan berupa SMS percakapan sehari-hari atau pesan promosi resmi yang wajar, aplikasi secara tepat mengklasifikasikannya sebagai "Aman". Notifikasi yang ditampilkan berwarna hijau dan tidak bersifat mengganggu aktivitas pengguna. Hasil pengujian ini dinyatakan **Valid**.
3. **Pengujian Fitur Pindai Manual** Pengujian dilakukan dengan menyalin dan menempelkan teks spesifik ke dalam kolom input pada menu *Scan Text*. Setelah tombol pindai ditekan, sistem berhasil memproses teks tersebut secara waktu nyata dan menampilkan kartu hasil (*result card*) yang secara akurat memuat skor probabilitas beserta label kategorinya. Hasil pengujian ini dinyatakan **Valid**.

4. **Pengujian Kontrol Fitur *Realtime Scanning*** Evaluasi ini menguji fungsionalitas sakelar pengontrol pada menu Pengaturan. Saat pengguna menonaktifkan fitur *Realtime Scanning*, layanan yang berjalan di latar belakang terbukti berhenti sepenuhnya. Sistem tidak lagi memantau kotak masuk, sehingga tidak ada notifikasi deteksi yang muncul ketika ada pesan baru diterima. Hasil pengujian ini dinyatakan **Valid**.
5. **Pengujian Mekanisme Perizinan** Pengujian dilakukan dengan menyimulasikan pembukaan aplikasi untuk pertama kalinya setelah proses instalasi (*first launch*). Sistem berhasil menampilkan jendela *pop-up* bawaan Android yang meminta izin akses pembacaan pesan ("*Read SMS*"). Pengguna diharuskan memberikan izin tersebut sebelum dapat mengakses menu utama aplikasi. Hasil pengujian ini dinyatakan **Valid**.

4.4. Pembahasan dan Analisis Kendala Teknis

Proses pengembangan dan integrasi antara model kecerdasan buatan dengan platform mobile menghadirkan berbagai tantangan teknis. Pembahasan berikut menguraikan evaluasi kelemahan model serta solusi rekayasa perangkat lunak yang diterapkan untuk mengatasi kendala operasional aplikasi.

4.4.1. Analisis Kinerja dan Kelemahan Model

Meskipun sistem memiliki tingkat akurasi yang tinggi, pengujian lapangan mengungkap adanya celah klasifikasi yang menghasilkan 7 deteksi positif palsu (False Positive) dan 5 negatif palsu (False Negative). Kasus negatif palsu umumnya terjadi pada pesan penipuan yang menerapkan teknik rekayasa sosial, di mana pesan disusun mengatasnamakan keluarga atau teman dekat tanpa menyertakan tautan

situs web. Kesamaan struktur semantik ini membuat model kehilangan konteks deteksi. Di sisi lain, kasus positif palsu kerap dipicu oleh pesan normal yang sangat singkat, seperti penggunaan satu huruf tunggal (contoh: huruf "a"), yang secara keliru diidentifikasi oleh algoritma sebagai anomali karakter yang mencurigakan.

4.4.2. Penyesuaian Eksekusi Backend TensorFlow

Kendala fatal ditemukan saat aplikasi mengalami penutupan paksa (force close) ketika sistem mencoba memuat fail model TensorFlow.js ke dalam memori perangkat. Analisis menunjukkan bahwa eksekusi bawaan yang mencoba mengakses unit pengolah grafis (GPU) perangkat seluler sering kali tidak stabil. Masalah ini berhasil diselesaikan dengan menerapkan fungsi tf.setBackend('cpu'), yang memaksa proses komputasi kecerdasan buatan agar sepenuhnya ditangani oleh unit pemroses sentral (CPU), sehingga stabilitas aplikasi tetap terjaga di berbagai jenis perangkat.

4.4.3. Modifikasi Mekanisme Pemindaian Waktu Nyata (Live Scanning)

Pustaka bawaan android-sms-listener terbukti memiliki batasan kritis karena gagal mendeteksi pesan masuk saat aplikasi diminimalkan ke latar belakang (tombol home ditekan). Untuk memastikan perlindungan berjalan terus-menerus, modul tersebut diganti menggunakan pustaka react-native-get-sms yang dikombinasikan dengan algoritma perulangan (looping). Mekanisme baru ini memicu aplikasi untuk melakukan penarikan data kotak masuk setiap 5 detik secara berkala, memastikan fitur pemindaian waktu nyata dapat beroperasi secara konsisten meskipun terdapat sedikit jeda operasional.

4.4.4. Transisi Modul Layanan Latar Belakang (Background Service)

Upaya menjalankan pemindaian secara mandiri di luar siklus hidup aplikasi utama menemui hambatan karena modul foreground-service versi lama tidak lagi mematuhi regulasi keamanan sistem operasi Android terbaru. Sistem operasi modern mewajibkan aplikasi yang berjalan di latar belakang untuk menampilkan notifikasi persisten secara eksplisit. Kendala ini diselesaikan melalui migrasi kode secara menyeluruh menggunakan pustaka react-native-background-actions yang mendukung manajemen notifikasi latar belakang sesuai dengan standar API Android terkini.

4.4.5. Penanganan Asinkronisasi Bahasa pada Notifikasi Latar Belakang

Penggunaan variabel bahasa dinamis (lokalisasi) pada notifikasi layanan latar belakang memicu kegagalan sistem yang berujung pada aplikasi tertutup otomatis. Hal ini disebabkan oleh proses rendering layanan latar belakang yang berjalan lebih cepat dibandingkan proses pemuatan berkas bahasa di siklus utama aplikasi. Solusi arsitektural yang diterapkan adalah menetapkan teks berbahasa Indonesia secara statis sebagai nilai bawaan (default) khusus untuk komponen notifikasi latar belakang, sehingga layanan dapat langsung berjalan tanpa harus menunggu proses asinkronisasi dari modul bahasa selesai.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan serangkaian proses penelitian, pengembangan, dan pengujian yang telah dipaparkan pada bab-bab sebelumnya, penelitian ini secara komprehensif telah menjawab rumusan masalah dan mencapai tujuan yang ditetapkan. Kesimpulan dari penelitian ini dapat diuraikan sebagai berikut:

1. **Penerapan Algoritma 1D-CNN untuk Deteksi SMS:** Metode *Deep Learning* dengan arsitektur *One-Dimensional Convolutional Neural Network* (1D-CNN) berhasil diterapkan secara efektif untuk mendeteksi SMS penipuan (*smishing*) berbahasa Indonesia. Melalui tahapan *Natural Language Processing* (NLP) seperti pembersihan teks dan tokenisasi, algoritma 1D-CNN terbukti mampu mengekstraksi fitur pola kalimat dan hubungan antarkata secara otomatis, sehingga model dapat mengenali variasi teks penipuan tanpa memerlukan rekayasa fitur manual yang rumit.
2. Perancangan dan Integrasi Aplikasi Mobile: Aplikasi pemfilteran SMS otomatis pada platform Android telah sukses dirancang dan dibangun menggunakan kerangka kerja React Native. Sebagaimana dijelaskan oleh Sofyan, Rahaningsih, dan (Sofyan et al., 2024), integrasi model ke dalam aplikasi memungkinkan aplikasi untuk melakukan prediksi secara otomatis saat pengguna memasukkan pesan atau menerima pesan baru di perangkat mereka. Model kecerdasan buatan berhasil diintegrasikan ke dalam ekosistem aplikasi perangkat bergerak melalui konversi ke format TensorFlow.js. Pendekatan ini memungkinkan aplikasi untuk melakukan

pemindaian secara langsung pada sisi klien (*client-side*) secara waktu nyata tanpa harus bergantung pada koneksi internet atau peladen eksternal, dengan memastikan stabilitas komputasi melalui penggunaan unit pemroses sentral (CPU).

3. **Tingkat Efektivitas dan Kinerja Model:** Penerapan algoritma 1D-CNN terbukti sangat efektif dan akurat. Berdasarkan hasil pengujian matriks kebingungan (*confusion matrix*) menggunakan 298 data uji, sistem berhasil mencapai tingkat akurasi keseluruhan sebesar 96,31 persen dengan nilai *loss* yang rendah (0,4695). Model ini juga menunjukkan sensitivitas yang sangat baik dalam menangkap pesan berbahaya, dibuktikan dengan capaian tingkat presisi sebesar 94 persen dan *recall* sebesar 96 persen untuk kelas penipuan.

5.2. Saran

Meskipun sistem telah beroperasi dengan baik dan mencapai tingkat akurasi yang tinggi, masih terdapat beberapa batasan dan kendala teknis yang ditemukan selama proses pengembangan. Mengingat pengamatan dari (Mahmud et al., 2024) bahwa sistem deteksi saat ini sering kali tidak selalu mampu mengimbangi sifat ancaman yang dinamis... sehingga studi lebih lanjut diperlukan untuk mengembangkan algoritma guna mendeteksi serangan yang semakin canggih, maka terdapat beberapa saran yang dapat dipertimbangkan untuk pengembangan dan penelitian selanjutnya:

1. **Pengayaan Dataset dan Pemahaman Konteks:** Model saat ini masih memiliki kelemahan berupa deteksi positif palsu (*False Positive*) pada pesan yang sangat singkat (misalnya satu karakter huruf) dan negatif palsu (*False Negative*) pada pesan rekayasa sosial yang tidak menyertakan tautan

URL. Pengembangan selanjutnya disarankan untuk memperbanyak variasi dataset latih yang mencakup pesan percakapan sehari-hari yang sangat pendek, serta memperkaya contoh teks manipulasi psikologis agar model memiliki pemahaman konteks semantik yang lebih utuh.

2. **Optimasi Mekanisme Layanan Latar Belakang:** Mekanisme pemindaian waktu nyata (*real-time scanning*) saat ini masih menggunakan metode penarikan data kotak masuk (*polling*) secara berkala setiap 5 detik guna menyiasati keterbatasan pustaka bawaan *React Native*. Untuk efisiensi penggunaan daya baterai dan memori yang lebih baik, penelitian mendatang disarankan untuk mengembangkan atau mengintegrasikan modul pendengar (*listener*) SMS yang ditulis langsung menggunakan bahasa *native* Android (Java atau Kotlin).
3. **Pemanfaatan Akselerasi Perangkat Keras (GPU):** Guna menghindari masalah penutupan paksa (*force close*), komputasi TensorFlow.js saat ini dipaksa berjalan sepenuhnya menggunakan CPU. Peneliti selanjutnya disarankan untuk mengeksplorasi optimasi pustaka TensorFlow.js atau memanfaatkan API *Neural Networks* bawaan Android agar proses pemindaian dapat memanfaatkan akselerasi *Graphics Processing Unit* (GPU) pada ponsel, sehingga proses deteksi menjadi jauh lebih cepat.
4. **Perbaikan Asinkronisasi Pelokalan Bahasa:** Fitur pelokalan bahasa pada komponen notifikasi yang berjalan di latar belakang saat ini harus menggunakan teks statis karena adanya kendala kecepatan proses pemuatan (*rendering*) yang tidak sinkron dengan siklus utama aplikasi. Pengembangan berikutnya diharapkan dapat merancang ulang alur

inisialisasi aplikasi agar variabel bahasa dinamis dapat dimuat secara utuh sebelum layanan latar belakang dieksekusi.

DAFTAR PUSTAKA

Ahmadi, M., Khajavi, M., Varmaghani, A., Ala, A., Danesh, K., & Javaheri, D. (2025). *Leveraging Large Language Models for Cybersecurity: Enhancing SMS Spam Detection with Robust and Context-Aware Text Classification*. http://arxiv.org/abs/2502.11014

Airlangga, G. (2024). Optimizing SMS Spam Detection Using Machine Learning: A Comparative Analysis of Ensemble and Traditional Classifiers. *Journal of Computer Networks, Architecture and High Performance Computing*, *6*(4), 1942–1951. https://doi.org/10.47709/cnahpc.v6i4.4822

Al-Kaabi, H. A., Darroudi, A., Jasim, A. K., Alaa, H., & Hussain, A.-K. (2024). Survey of SMS Spam Detection Techniques: A Taxonomy. *Alkadhim Journal for Computer Science*, *4*(2). https://doi.org/10.53523/ijoirVolxIxIDxx

Al-Kabbi, H. A., Feizi-Derakhshi, M.-R., & Pashazadeh, S. (2024). A Hierarchical Two-Level Feature Fusion Approach for SMS Spam Filtering. *Intelligent Automation & Soft Computing*, *39*(4), 665–682. https://doi.org/10.32604/iasc.2024.050452

Al-Zebari, A., Barwary, M., Omar, N., Zebari, N. A., & Zebari, D. A. (2024). Deep Learning Hybrid Approach for Accurate SMS Spam Identification. In *Journal of Information Systems Engineering and Management* (Vol. 2025, Number 10s). https://www.jisem-journal.com/

Amin, M. B. M., Hakim, G., Maulana, M. T., Alwan, M. F., Anggraheni, H. S., Naufal, M. J., & Yudistira, N. (2024). Deteksi Spam Berbahasa Indonesia Berbasis Teks Menggunakan Model Bert. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(6), 1291–1302. https://doi.org/10.25126/jtiik.2024118121

Faisal, R., Budiman, I., Abadi, F., & Turianto, D. (2022). *Applying Features Based on Word Embedding Techniques to 1D CNN for Natural Disaster Messages Classification*. IEEE.

Hasti, P., & Barmada, B. (2025). *SMS PHISHING DETECTION USING MACHINE LEARNING AND DEEP LEARNING TECHNIQUES*.

Hikmaturokhman, A., Nafi'ah, H., Larasati, S., Wahyudin, A., Ariprawira, G., & Pramono, S. (2022). Deep Learning Algorithm Models for Spam Identification on Cellular Short Message Service. *Journal of Communications*, *17*(9), 769–776. https://doi.org/10.12720/jcm.17.9.769-776

Kristananda, V., Muhaimin, A., Pembangunan, U., Veteran, N. ", & Timur, J. (2025). *IMPLEMENTASI MACHINE LEARNING UNTUK MENINGKATKAN KESADARAN MASYARAKAT DALAM DETEKSI SPAM SMS MELALUI PROGRAM MBKM MAGANG MANDIRI DI HACKTIV8 INDONESIA*. *5*(1), 1–5.

Latifah, N., Dwiyansaputra, R., & Nugraha, G. S. (2024). Multiclass Text Classification of Indonesian Short Message Service (SMS) Spam using Deep Learning Method and Easy Data Augmentation. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *23*(3), 663–676. https://doi.org/10.30812/matrik.v23i3.3835

Lumbantobing, H., Deddy, R., Melati Manalu, E., Sartika, D., Sitinjak, P., & Manurung, T. W. (2021). *Roy Deddy Hasiholan Lumban Tobing: Rancangan Aplikasi Mobile Pendeteksi Spam… Sejarah penerimaan*.

Mahmud, T., Prince, M. A. H., Ali, M. H., Hossain, M. S., & Andersson, K. (2024). Enhancing Cybersecurity: Hybrid Deep Learning Approaches to Smishing Attack Detection. *Systems*, *12*(11). https://doi.org/10.3390/systems12110490

Maugy Al Kautsar, Galet Guntoro Setiaji, & Ahmad Rifa'i. (2025). Analisis Komparasi Kinerja LSTM dan CNN dalam Deteksi Spam Email Berbasis Deep learning. *Bulletin of Computer Science Research*, *5*(4), 584–593. https://doi.org/10.47065/bulletincsr.v5i4.572

Menthe, S., Rawal, K., Hirave, M., & Patil, A. J. (2024). SMS SPAM DETECTION USING MACHINE LEARNING. *IJARCCE*, *13*(3). https://doi.org/10.17148/ijarcce.2024.13307

Moulana, C., #1, B., & Gunawan, W. (2024). *JEPIN (Jurnal Edukasi dan Penelitian Informatika) Deteksi Email Spam menggunakan Algoritma Convolutional Neural Network (CNN)*.

Ramachandrappa, N. C. (2024). A Comparative Analysis of Native vs React Native Mobile App Development. *International Journal of Computer Trends and Technology*, *72*(9), 57–62. https://doi.org/10.14445/22312803/ijctt-v72i9p110

Reviantika, F., Azhar, Y., & Marthasari, G. I. (2021). Analisis Klasifikasi SMS Spam Menggunakan Logistic Regression. *REPOSITOR*, *3*(4), 387–392. https://www.cnbcindonesia.com

Satgas PASTI. (2025). *SIARAN PERS SATGAS PASTI BLOKIR 507 AKTIVITAS DAN ENTITAS KEUANGAN ILEGAL MINTA MASYARAKAT WASPADAI PENIPUAN YANG SEMAKIN MARAK*.

Sofyan, M. A., Rahaningsih, N., & Dana, R. D. (2024). DETEKSI SMS SPAM BERBAHASA INDONESIA MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Number 3). http://bit.ly/yw_sms_spam_indonesia,

Yusup, M., Fadillah, I., Fauzanie, R. A., Pratiwi, R. L., Handayani, R. I., & Widanengsih, E. (2026). *Journal of Artificial Intelligence and Engineering Applications Spam Message Classification Using the Naïve Bayes Algorithm Based on RapidMiner* (Vol. 5, Number 2). https://ioinformatic.org/