RANCANG BANGUN APLIKASI KLASIFIKASI DAN DIGITALISASI DOKUMEN BERBASIS ANDROID MENGGUNAKAN ML KIT SDKS DAN ALGORITMA BERT

IMAN MUSTIKA ISMAIL

H071201050

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2024

RANCANG BANGUN APLIKASI KLASIFIKASI DAN DIGITALISASI DOKUMEN BERBASIS ANDROID MENGGUNAKAN ML KIT SDKS DAN ALGORITMA BERT

IMAN MUSTIKA ISMAIL

H071201050

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS MATEMATIKA DAN ILMU PEMNGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2024

RANCANG BANGUN APLIKASI KLASIFIKASI DAN DIGITALISASI DOKUMEN BERBASIS ANDROID MENGGUNAKAN ML KIT SDKS DAN ALGORITMA BERT

IMAN MUSTIKA ISMAIL
H071201050

Skripsi

sebagai salah satu syarat untuk mencapai gelar sarjana

Program Studi Sistem Informasi

pada

PROGRAM STUDI SISTEM INFORMASI
DEPARTEMEN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PEMNGETAHUAN ALAM
UNIVERSITAS HASANUDDIN
MAKASSAR
2024

SKRIPSI

RANCANG BANGUN APLIKASI KLASIFIKASI DAN DIGITALISASI DOKUMEN BERBASIS ANDROID MENGGUNAKAN ML KIT SDKS DAN ALGORITMA BERT

IMAN MUSTIKA ISMAIL
H071201050

Skripsi,

telah dipertahankan di depan Panitia Ujian Sarjana Sistem Informasi pada 8 November 2024 dan dinyatakan telah memenuhi syarat kelulusan

pada

Program Studi Sistem Informasi
Departemen Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Hasanuddin
Makassar

Mengesahkan:
Pembimbing Tugas Akhir,

ji, S.Si., M.Si.
24061001

Mengetahui:
Ketua Program Studi,

Prof. Dr. Jeffry Kusuma, Ph.D.
NIP. 196411121987031002

PERNYATAAN KEASLIAN SKRIPSI DAN PELIMPAHAN HAK CIPTA

Dengan ini saya menyatakan bahwa, skripsi berjudul "Rancang Bangun Aplikasi Klasifikasi dan Digitalisasi Dokumen Berbasis Android Menggunakan ML KIT SDKs dan Algoritma BERT" adalah benar karya saya dengan arahan dari pembimbing Edy Saputra Rusdy, S.Si., M.Si. sebagai Pembimbing Utama. Karya ilmiah ini belum diajukan dan tidak sedang diajukan dalam bentuk apa pun kepada perguruan tinggi mana pun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka skripsi ini. Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini adalah karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut berdasarkan aturan yang berlaku.

Dengan ini saya melimpahkan hak cipta (hak ekonomis) dari karya tulis saya berupa skripsi ini kepada Universitas Hasanuddin.

Makassar, 8-November-2024

IMAN MUSTIKA ISMAIL
NIM H071201050

UCAPAN TERIMA KASIH

Puji dan syukur penulis panjatkan kepada Allah Subhanahu Wa Ta'ala atas segala pertolongan, rahmat, dan kasih sayang-Nya yang senantiasa tercurah kepada penulis hingga saat ini. Shalawat serta salam semoga selalu tercurah kepada Nabi Muhammad Shallallahu 'Alaihi Wa Salam. Alhamdulillah, berkat nikmat dan pertolongan dari-Nya, penulis dapat menyelesaikan skripsi ini yang berjudul **"Rancang Bangun Aplikasi Klasifikasi dan Digitalisasi Dokumen Berbasis Android Menggunakan ML KIT SDKs dan Algoritma BERT"**.

Dalam menyusun skripsi ini, penulis tidak luput dari berbagai kesulitan dan hambatan, namun atas bantuan dan dorongan dari berbagai pihak akhirnya penulisan skripsi ini dapat terselesaikan.

Untuk itu, pada kesempatan ini penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya dan penghargaan yang setinggi-tingginya kepada semua pihak yang telah membantu serta mendukung penulis dalam menyusun dan menyelesaikan skripsi ini, yaitu kepada:

1. Rektor Universitas Hasanuddin, Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc.**, beserta jajarannya.
2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Bapak **Dr. Eng. Amiruddin, S.Si., M.Si.**, beserta jajarannya.
3. Ketua Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam, Bapak **Dr. Firman, S.Si., M.Si.**, beserta jajarannya.
4. Ketua Program Studi Sistem Informasi, Bapak **Prof. Dr. Jeffry Kusuma, Ph.D.**, beserta jajarannya.
5. Dosen pembimbing utama, Bapak **Edy Saputra Rusdy, S.Si., M.Si.**, yang telah memberikan arahan, bimbingan, dan panduan selama proses penyusunan skripsi.
6. Dosen penguji pertama, Ibu **Rozalina Amran, S.T., M.Eng.**, yang telah bersedia meluangkan waktu untuk menguji, memberikan kritik dan saran yang bermanfaat dalam memperbaiki skripsi ini.
7. Dosen penguji kedua, Ibu **Riskawati, S.Si., M.Si.**, yang telah bersedia meluangkan waktu untuk menguji, memberikan kritik dan saran yang bermanfaat dalam memperbaiki skripsi ini.
8. Keluarga penulis, yaitu Ayahanda **Muh. Yusuf** dan Ibunda **Kastiani Tigkas** yang telah memberikan dukungan baik secara psikis maupun materiil, bimbingan dan motivasi selama proses penyusunan skripsi ini.
9. Kakak **Iska Mustika Ismail** yang telah memberikan dukungan motivasi selama proses penyusunan skripsi ini.
10. Tim *developer* sekaligus sahabat seperjuangan masa kuliah **Galaxy** aitu Haerul, Ojan, Mamet, Azhar dan Faizah yang telah menjadi gi informasi, teman belajar, hingga tempat keluh kesah selama

n, mentor, dan tim **Bangkit Academy 2023 Batch 1** yang telah kesempatan bagi penulis untuk bisa berkembang serta banyak pengetahuan penting yang berguna untuk karir penulis

12. Mentor dan teman-teman magang **Arekmabig** selama magang di **Bejana.IO**, yaitu mas Rizky, mas Anam, mas Bintang, mas Umam, Galuh, Bintang, Yusron, Muham, Afif, Zandy, Rizki, Riska, Maria, Gloria dan Muti, yang telah memberikan pengalaman dan pengetahuan berharga yang berguna untuk karir penulis kedepannya.
13. **Laboratorium Sistem Informasi** sebagai tempat berbagi ilmu dan informasi yang berharga selama masa perkuliahan.
14. Teman-teman **KKNT 111 Digitalisasi UMKM Gowa Desa Jonjo**, yaitu Alqi, Alif, Atiqah, Clief, Gav, Gede, Maya, Ibrahim, Kifli, Bagus, Rika, dan Kamila atas kerja samanya sehingga dapat menyelesaikan program ini sebagai salah satu syarat kelulusan.
15. Teman-teman **KKN Internasional 112 Fukuoka Jepang**, yang telah memberikan partisipasi serta pengalaman berharga kepada penulis selama berada di Fukuoka, yang sangat berguna untuk rencana studi lanjutan penulis di sana.
16. Seluruh teman-teman **Sistem Informasi 2020** yang telah menjadi teman seperjuangan dari masa kuliah daring hingga akhir perkuliahan di Universitas Hasanuddin.

Penulis menyadari bahwa skripsi ini jauh dari kata sempurna, oleh karena itu dengan segala kerendahan hati, penulis ingin meminta maaf dan berikan kritik dan saran yang membangun. Akhir kata, penulis berharap skripsi ini dapat memberikan manfaat bagi penulis dan pembaca, Aamiin.

Penulis,

Iman Mustika Ismail

ABSTRAK

IMAN MUSTIKA ISMAIL. **Rancang Bangun Aplikasi Klasifikasi dan Digitalisasi Dokumen Berbasis Android Menggunakan ML KIT SDKs dan Algoritma BERT** (dibimbing oleh Edy Saputra Rusdi).

Latar Belakang. Di era digital, pengelolaan dokumen masih manual dan rentan kesalahan. *Artificial Intelligence* (AI) melalui *Natural Language Processing* (NLP) dan *text recognition* memberikan solusi otomatis untuk klasifikasi dan digitalisasi dokumen. Algoritma BERT telah terbukti efektif untuk klasifikasi teks, sementara ML Kit SDK dari Google memungkinkan pengenalan teks yang mudah diintegrasikan ke aplikasi *mobile*. Aplikasi berbasis Android dengan integrasi AI ini menjadi sangat penting untuk mengelola dokumen secara efisien. **Tujuan**. Penelitian ini bertujuan merancang aplikasi Android yang memanfaatkan ML Kit SDKs dan BERT untuk mengklasifikasikan dan mendigitalkan dokumen seperti KTP dan SIM secara otomatis. **Metode**. Pengembangan aplikasi menggunakan metode *waterfall* yang melibatkan analisis, desain, pengembangan, integrasi, dan pengujian. Model *machine learning* BERT dilatih dengan data gambar dokumen, diintegrasikan ke aplikasi Android, dan dihubungkan ke *server* menggunakan FastAPI dan Ngrok untuk akses API publik. **Hasil**. Aplikasi berhasil dikembangkan dan mampu mengklasifikasikan dokumen KTP dan SIM secara otomatis dengan tingkat akurasi yang tinggi. Pengujian menunjukkan bahwa integrasi ML Kit SDKs mampu melakukan pengenalan teks dengan baik pada gambar dokumen, sementara model BERT yang di-*fine-tune* memberikan hasil klasifikasi yang akurat. Sistem ini dapat memproses dokumen dengan cepat dan menampilkan hasil digitalisasi serta klasifikasi secara *real-time*. **Kesimpulan**. Aplikasi yang dikembangkan berhasil menerapkan algoritma BERT untuk klasifikasi dokumen dan ML Kit SDKs untuk pengenalan teks secara efektif. Aplikasi ini dapat memberikan solusi yang efisien untuk pengelolaan dokumen, terutama dalam hal klasifikasi dan digitalisasi dokumen KTP dan SIM. Implementasi teknologi ini dapat meningkatkan efisiensi dalam pengelolaan dokumen dan memungkinkan pengguna untuk mengakses dokumen dengan lebih mudah.

Kata Kunci: Android, BERT, ML KIT, Klasifikasi Dokumen, Digitalisasi Dokumen

ABSTRACT

IMAN MUSTIKA ISMAIL. **Design and Development of an Android-Based Document Classification and Digitization Application Using ML Kit SDKs and BERT Algorithm** (supervised by Edy Saputra Rusdi).

Background. In the digital era, document management is still manual and error-prone. Artificial Intelligence (AI) through Natural Language Processing (NLP) and text recognition provides an automated solution for document classification and digitization. The BERT algorithm has proven effective for text classification, while Google's ML Kit SDK allows text recognition to be easily integrated into mobile applications. Android-based applications with this AI integration are becoming very important for managing documents efficiently. **Aim**. This research aims to design an Android application that utilizes ML Kit SDKs and BERT to classify and digitize documents such as KTP and SIM automatically. **Methods**. The application development used the waterfall method involving analysis, design, development, integration, and testing. The BERT machine learning model was trained with document image data, integrated into the Android application, and connected to the server using FastAPI and Ngrok for public API access. **Results**. The application was successfully developed and was able to classify KTP and SIM documents automatically with a high level of accuracy. Tests show that the integration of ML Kit SDKs is able to perform good text recognition on document images, while the fine-tuned BERT model provides accurate classification results. The system can process documents quickly and display digitization and classification results in real-time. **Conclusion**. The developed application successfully applies BERT algorithm for document classification and ML Kit SDKs for text recognition effectively. This application can provide an efficient solution for document management, especially in terms of classification and digitization of KTP and SIM documents. The implementation of this technology can improve efficiency in document management and allow users to access documents more easily.

Keywords: Android, BERT, ML KIT, Document Classification, Document Digitization

DAFTAR ISI

DAFTAR TABEL

Nomor Urut Halaman

DAFTAR GAMBAR

Nomor Urut Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Di era digital saat ini, dokumen dan informasi menjadi salah satu aset yang paling berharga bagi individu maupun organisasi. Keberhasilan dalam mengelola dan mengakses dokumen dengan efisien dapat memberikan keunggulan kompetitif yang signifikan. Namun, sering kali proses pengelolaan dokumen masih sangat manual dan rentan terhadap kesalahan manusia, yang dapat menghambat produktivitas dan efisiensi. Banyaknya informasi pada dokumen mendorong manusia untuk mencari cara untuk mendapatkan informasi pada sebuah dokumen dengan tepat dan dengan waktu yang singkat.

Pada saat yang sama, kemajuan teknologi dan perkembangan kecerdasan buatan telah membuka peluang baru untuk mengotomatisasi dan mengoptimalkan pengelolaan dokumen. Teknik-teknik *Natural Language Processing* (NLP) dan *text recognition* telah memungkinkan pengembangan sistem yang mampu mengenali, mengklasifikasi, dan mendigitalisasikan dokumen dengan tingkat akurasi yang sangat tinggi.

Salah satu teknologi yang menjadi sangat relevan dalam bidang ini adalah algoritma *Bidirectional Encoder Representations from Transformers* (BERT). Algoritma BERT telah berhasil mencapai performa terbaik pada masalah klasifikasi teks, oleh karena itu algoritma ini telah terbukti sangat efektif dalam pemahaman konteks teks dan digunakan dalam berbagai aplikasi NLP (Shah et al., 2023). Sedangkan ML Kit sendiri merupakan *Software Development Kit* (SDK) yang disediakan oleh Google untuk memanfaatkan *machine learning* ke semua aplikasi *mobile* dalam paket yang sangat mudah digunakan, sehingga menyempurnakan aplikasi dengan banyak fitur dinamis (Modekurti et al., 2020).

Selain itu, dengan peningkatan penetrasi perangkat Android di seluruh dunia, perangkat *mobile* menjadi salah satu alat utama yang digunakan untuk mengakses dan berinteraksi dengan berbagai jenis dokumen. Oleh karena itu pengembangan aplikasi berbasis Android yang menggunakan kecerdasan buatan untuk klasifikasi dan digitalisasi dokumen menjadi semakin penting.

Kecerdasan buatan telah banyak dimanfaatkan dalam pengelolaan dokumen, terutama melalui *Natural Language Processing* (NLP) untuk otomatisasi klasifikasi teks. Studi sebelumnya telah menerapkan algoritma BERT untuk klasifikasi teks dokumen, seperti yang dilakukan oleh Shah et al. (2023), yang menggunakan BERT

sifikasi dokumen. Permana dan Budayawan (2020) juga telah klasifikasi semantik teks pada ringkasan karya ilmiah rFlow Lite. Penelitian lain oleh Aulia dan Kurniawati (2022) del klasifikasi dokumen artikel teks berita olahraga dan non-sa Indonesia dengan algoritma *Support Vector Machine* (SVM). et al. (2024) membandingkan beberapa model seperti BERT, *cision Trees* (DT) dalam klasifikasi teks yang efektif. Terakhir,

Nguyen (2024) menunjukkan bagaimana Google's *Machine Learning Kit* dapat dimanfaatkan dalam pengembangan aplikasi Android, yang mendukung implementasi AI di perangkat *mobile*. Penelitian-penelitian ini menunjukkan potensi AI dalam mengotomatisasi dan meningkatkan efisiensi pengelolaan dokumen pada berbagai platform, termasuk aplikasi *mobile*.

Penelitian ini bertujuan untuk merancang dan mengembangkan sebuah aplikasi berbasis Android yang memanfaatkan *text recognition* pada ML Kit SDKs dan algoritma BERT untuk mengklasifikasikan dokumen secara otomatis dan mendigitalkan kontennya. Aplikasi ini diharapkan dapat memberikan manfaat signifikan dalam hal efisiensi pengelolaan dokumen, memungkinkan pengguna untuk dengan mudah mengakses, mencari, dan mengelola konten pada sebuah dokumen di perangkat *mobile*. Dengan adanya kemajuan teknologi dan kecerdasan buatan, pengembangan aplikasi ini diharapkan dapat memberikan kontribusi yang berarti dalam meningkatkan produktivitas dan efisiensi dalam pengelolaan dokumen, serta menghadirkan solusi yang lebih baik untuk kebutuhan pengguna di era digital saat ini.

1.2 Tujuan dan Manfaat Penelitian

1.2.1 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka dapat dirumuskan permasalahan dari penelitian ini, yaitu:

a. Bagaimana merancang dan membangun aplikasi klasifikasi dan digitalisasi dokumen berbasis Android?
b. Bagaimana cara mengaplikasikan *text recognition* yang ada pada ML Kit SDKs pada aplikasi klasifikasi dan digitalisasi dokumen berbasis Android?
c. Bagaimana cara menerapkan algoritma BERT dalam proses klasifikasi dokumen di aplikasi berbasis Android?
d. Bagaimana kinerja dari model algoritma BERT pada aplikasi klasifikasi dan digitalisasi dokumen berbasis Android?

1.2.2 Batasan Masalah

Adapun batasan masalah pada penelitian ini, diantaranya:

a. Aplikasi yang dirancang dan dibangun hanya pada platform Android.
b. Aplikasi menggunakan *text recognition v2* yang ada pada ML Kit SDKs.

ng digunakan adalah *Bidirectional Encoder Representations mers* (BERT).
en yang dapat di klasifikasikan berupa Kartu Tanda Penduduk ırat Izin Mengemudi (SIM).

1.2.3 Tujuan

Tujuan dari penelitian ini adalah:

a. Merancang dan membangun aplikasi klasifikasi dan digitalisasi dokumen berbasis Android.
b. Mengimplementasikan teknologi *text recognition* dari ML Kit SDKs untuk meningkatkan kecepatan dan efisiensi proses digitalisasi dokumen dalam aplikasi.
c. Menerapkan algoritma BERT dalam klasifikasi dokumen guna meningkatkan akurasi pengenalan konten dalam aplikasi.
d. Mengetahui kinerja model algoritma BERT dalam konteks aplikasi klasifikasi dan digitalisasi dokumen berbasis Android.

1.2.4 Manfaat

Hasil dari penelitian ini diharapkan dapat memberikan manfaat berikut:

a. Peningkatan Efisiensi: Aplikasi yang dihasilkan diharapkan dapat membantu pengguna dalam mengelola dokumen mereka dengan lebih efisien, mengurangi waktu yang dibutuhkan untuk klasifikasi dan digitalisasi.
b. Kemudahan Akses: Dokumen yang telah didigitalkan diharapkan dapat dengan mudah diakses melalui perangkat Android, memungkinkan pengguna untuk mencari dan membaca isi dokumen dengan mudah.
c. Kontribusi Teknologi: Penelitian ini diharapkan dapat memberikan kontribusi pada pengembangan aplikasi cerdas berbasis Android dan memanfaatkan kecerdasan buatan dalam pemrosesan dokumen.
d. Referensi untuk Penelitian Selanjutnya: Hasil dari penelitian ini diharapkan dapat menjadi referensi bagi penelitian lanjutan dalam bidang pengelolaan dokumen dan aplikasi cerdas berbasis Android.

1.3 Landasan Teori

1.3.1 Rancang Bangun

Perancangan atau rancang merupakan serangkaian prosedur untuk menerjemahkan hasil analisis dan sebuah sistem ke dalam bahasa pemrograman untuk mendeskripsikan dengan detail bagaimana komponen-komponen sistem di implementasikan. Sedangkan pengertian pembangunan atau bangun sistem adalah

n sistem baru maupun mengganti atau memperbaiki sistem ecara keseluruhan maupun sebagian (Pressman, 2010).

sis Android

gram komputer atau perangkat lunak yang dirancang dan menjalankan tugas tertentu atau fungsi-fungsi tertentu pada

komputer, perangkat seluler, atau perangkat elektronik lainnya. Aplikasi dapat memiliki berbagai tujuan dan cakupan, tergantung pada kebutuhan dan spesifikasi pengembangnya.

Android adalah *Operating System* (OS) yang dikembangkan oleh Google, yang digunakan pada berbagai perangkat elektronik, terutama perangkat seluler seperti *smartphone* dan tablet. Android didasarkan pada *kernel* Linux dan dirancang untuk menyediakan lingkungan perangkat lunak yang kuat, fleksibel, dan terbuka untuk pengembangan aplikasi. Sistem operasi ini telah menjadi salah satu platform perangkat seluler paling populer di dunia, memberikan akses ke berbagai aplikasi, layanan, dan fitur yang memungkinkan pengguna untuk berkomunikasi, bekerja, bermain, dan menjalankan berbagai tugas lainnya dengan perangkat seluler mereka. Android juga memungkinkan berbagai produsen perangkat untuk mengadaptasi dan mengkustomisasi sistem operasi ini sesuai dengan kebutuhan dan preferensi mereka, menciptakan beragam pengalaman pengguna di berbagai perangkat Android. Android OS menjadi platform yang sangat diminati untuk perangkat *mobile*, menguasai sekitar 74,5% pangsa pasar. Akibatnya, terjadi peningkatan yang signifikan dalam upaya pengembangan aplikasi pihak ketiga oleh baik pengembang individu maupun perusahaan sebagai respons terhadap perubahan dinamika pasar ini. Salah satu aspek yang paling menarik dari Android adalah fleksibilitas dalam penyebaran aplikasi yang bersifat *open source* dan terbuka, yang memungkinkan pengembang mandiri untuk membuat dan mendistribusikan aplikasi mereka sendiri. Di samping itu, platform Android juga menyediakan *Application Programming Interface* (API) yang luas, memberikan aplikasi akses yang besar ke sumber daya perangkat keras dan sistem (Almomani & Khayer, 2020).

Aplikasi berbasis Android adalah aplikasi yang menggunakan Android sebagai OS yang berjalan pada aplikasi tersebut. Pada umumnya, para pengembang aplikasi Android menggunakan Java dan Kotlin sebagai bahasa pemrograman dan Android Studio sebagai *Integrated Development Environment* (IDE) selama pengembangan aplikasi.

1.3.3 ML Kit SDKs

Salah satu layanan Google Firebase adalah ML Kit. *Machine Learning* (ML) Kit adalah *mobile Software Development Kit* (SDK) yang menghadirkan kompetensi Google di bidang *machine learning* ke semua jenis aplikasi *mobile* OS dalam paket yang sangat efisien dan mudah digunakan, sehingga meningkatkan aplikasi dengan banyak fitur dinamis. Pengembang Android kini tidak perlu memiliki pengalaman di

ning dan dapat dengan mudah mengimplementasikan fungsi nya dalam beberapa baris kode. ML Kit SDK menyediakan i lainnya termasuk identifikasi bahasa, mendeteksi wajah, gambar, mengenali *landmark*. SDK ini dapat digunakan di) atau di *cloud* (Modekurti et al., 2020).

NLP merupakan subjek yang mendalam, sebagai pengembang untuk memahami setiap algoritma dan mengevaluasi hasil

algoritma serta mempelajari hasil ini untuk digunakan dalam Aplikasi Android. Namun hal ini dapat dipermudah dengan ML Kit. ML Kit memberi kita banyak API yang dapat digunakan oleh pengembang Android dengan mengimpor dependensi yang diperlukan. Oleh karena itu, ini adalah pendekatan terbaik untuk membuat aplikasi yang lebih cepat, aman, dan lebih baik dalam mengimplementasikan konsep *Machine Learning* (Modekurti et al., 2020).

1.3.4 *Text Recognition*

Text recognition atau pengenalan teks adalah proses identifikasi dan ekstraksi teks dari berbagai jenis gambar, dokumen, atau media visual. Tujuan utama dari *text recognition* adalah mengubah teks yang terdapat dalam format gambar menjadi teks yang dapat diolah oleh komputer. Hal ini memungkinkan teks yang tercetak atau ditulis tangan dalam gambar yang tidak dapat diedit menjadi data teks yang dapat dicari, disunting, dianalisis, atau diintegrasikan ke dalam berbagai aplikasi dan sistem. Teknik *text recognition* melibatkan sejumlah langkah dan konsep, termasuk pemrosesan citra, segmentasi teks, pengenalan karakter atau *Optical Character Recognition* (OCR), dan sering kali pemahaman bahasa untuk memahami konteks teks.

OCR adalah konversi elektronik atau mekanis dari gambar teks yang diketik, ditulis tangan, atau dicetak menjadi teks yang dikodekan mesin, baik dari dokumen yang dipindai, foto dokumen, foto pemandangan, atau dari teks *subtitle* yang ditumpangkan pada gambar. Biasanya, sistem OCR mencakup dua modul utama: modul deteksi teks dan modul pengenalan teks. Deteksi teks bertujuan untuk melokalisasi semua blok teks di dalam gambar teks, baik di tingkat kata maupun tingkat baris teks. Sedangkan pengenalan teks bertujuan untuk memahami isi gambar teks dan mentranskripsikan sinyal visual ke dalam *token* bahasa alami (Minghao, et al., 2023).

1.3.5 BERT (*Bidirectional Encoder Representations from Transformers*)

Menurut Rothman (2021), BERT adalah sebuah model *Deep Learning* yang digunakan untuk merepresentasikan kata-kata secara kontekstual dalam pra-pelatihan *Natural Language Processing* (NLP). Model ini dikembangkan oleh Google dan diterbitkan pada tahun 2018. Dalam pelatihan, kata-kata disesuaikan dengan menggunakan *Masked Language Model* (MLM) dan *Transformers* dua arah (*bidirectional*). Sesuai dengan namanya, BERT fokus pada proses *encoding* dan menghasilkan sebuah model bahasa. Model BERT terdiri dari struktur *encoder* rah dengan beberapa lapisan. BERT hanya menggunakan alam *Transformers* dan tidak menggunakan tumpukan *decoder*

eberapa keunggulan dibandingkan dengan metode lainnya,

a. Pemahaman Kontekstual yang Lebih Baik: BERT mengungguli metode lain dengan memanfaatkan *encoder* dan prinsip *attention*, serta memproses teks secara keseluruhan sebagai *input*, bukan hanya mengikuti urutan sekuensial. Ini memungkinkan BERT untuk memahami hubungan kontekstual setiap *token* dengan lebih baik. Selain itu, BERT juga mampu mengatasi teks yang panjang lebih efisien dibandingkan dengan *Recurrent Neural Network* (RNN), berkat penggunaan *positional encoding* saat membaca *input* (Husin, 2023).
b. Cocok untuk *Dataset* Berukuran Kecil: BERT sangat berguna untuk *dataset* berukuran kecil karena sudah di-*pre-train*, sehingga hanya perlu *fine-tuning* untuk tugas tertentu (Husin, 2023).
c. Efisien dengan Jumlah Parameter yang Lebih Sedikit: BERT menggunakan struktur *Transformers* dan memiliki jumlah parameter yang lebih sedikit daripada model *Convolutional Neural Network* (CNN). Hal ini memungkinkan BERT untuk mencapai kinerja yang lebih baik dalam waktu yang lebih efisien (Husin, 2023).

1.3.6 Klasifikasi Dokumen

Klasifikasi adalah proses pengelompokan objek yang memiliki karakteristik atau ciri yang sama ke dalam beberapa kelas. Pada umumnya klasifikasi dokumen dilakukan dengan menentukan ciri-ciri atau fitur-fitur yang diwakili oleh kalimat-kalimat penting (Indriani et al., 2017).

Sedangkan menurut penulis, klasifikasi dokumen merupakan sebuah proses pengelompokan dokumen ke dalam kategori atau kelas tertentu berdasarkan isi kontennya atau karakteristik tertentu yang relevan. Tujuan utama dari klasifikasi dokumen adalah untuk mengorganisir, mengidentifikasi, atau mengelompokkan dokumen agar dapat dengan mudah diakses, dicari, atau dianalisis. Proses ini dapat dilakukan secara otomatis dengan menggunakan algoritma pada *machine learning*, seperti algoritma BERT yang digunakan penulis pada penelitian kali ini.

1.3.7 Digitalisasi Dokumen

Digitalisasi adalah proses alih media dari bentuk tercetak, audio, maupun video menjadi bentuk digital. Sistem digitalisasi dokumen ini dalam pengalihan dokumen fisik ke dokumen digital mengatur beberapa hal seperti *scanning*, pengindeksan dokumen elektronik, pencarian dokumen dan proses cetak media elektronik di kembalikan ke media kertas (Saifudin & Widrani, 2021).

ıen yang dimaksud oleh penulis adalah proses mengonversi dokumen fisik menjadi format teks digital. Konten yang telah digital tersebut kemudian ditampilkan sesuai maksud konten digitalisasi dokumen ini agar isi konten pada dokumen dapat ngan cepat dan lebih efisien. Proses ini dapat dilakukan secara

otomatis dengan menggunakan algoritma pada *machine learning*. Pada penelitian ini penulis menggunakan *text recognition* yang telah disediakan oleh ML Kit SDKs.

1.3.8 API (*Application Programming Interface*)

Application Programming Interface (API) adalah antarmuka yang dibangun oleh pengembang sistem sehingga beberapa atau seluruh fungsi sistem dapat diakses secara terprogram (Afriansyah et al., 2023). API juga biasa dianggap sebagai suatu kumpulan teknik yang jelas untuk menciptakan suatu komunikasi antara perangkat lunak yang berbeda beda komponen. Fungsi API adalah untuk memudahkan penggunaan teknologi tertentu ketika membangun perangkat lunak atau aplikasi bagi pengembang (Akmal & Dasaprawira, 2022).

1.3.9 Kotlin

Kotlin adalah bahasa pemrograman modern, muncul pada tahun 2011, yang merupakan alternatif dari Java, yang dapat bekerja berdampingan dengan mulus. Banyak bukti tersedia dalam literatur yang menggarisbawahi bahwa Kotlin semakin populer di kalangan pengembang perangkat lunak Android (Ardito et al., 2020). Kotlin adalah bahasa pemrograman pragmatis untuk JVM dan Android yang menggabungkan fitur-fitur berorientasi objek dan fungsional serta berfokus pada *interoperability*, *safety*, *clarity*, dan *tooling support*. Sebagai bahasa serba guna, Kotlin bekerja di mana saja di mana Java bekerja: aplikasi sisi server, aplikasi seluler (Android), dan aplikasi desktop. Kotlin bekerja dengan semua alat dan layanan utama seperti IntelliJ IDEA, Android Studio, Eclipse, Maven, Gradle, Ant, Spring Boot, GitHub, dan lainnya. Salah satu fokus utama Kotlin adalah interoperabilitas dan mendukung proyek-proyek campuran Java dan Kotlin, membuat adopsi menjadi lebih mudah yang mengarah pada berkurangnya kode sama yang berulang (*boilerplate code*) dan keamanan tipe yang lebih baik. Selain itu, Kotlin memiliki pustaka standar yang luas yang membuat tugas sehari-hari menjadi mudah dan lancar sambil menjaga jejak *bytecode* tetap rendah. Tentu saja, *library* Java apa pun dapat digunakan di Kotlin juga, begitu pun sebaliknya (Breslav, 2016).

1.3.10 Python

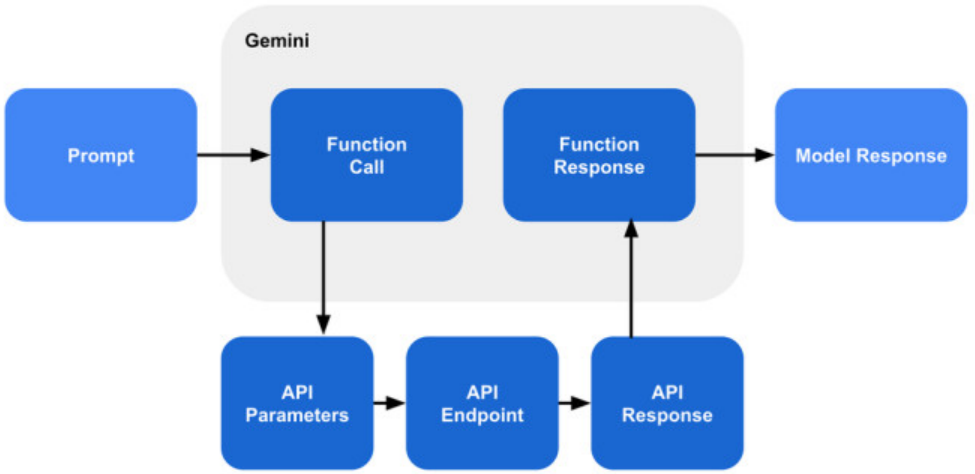
Python adalah bahasa pemrograman tingkat tinggi yang dikembangkan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Python adalah bahasa pemrograman tingkat tinggi, dinamis, berorientasi objek, dan serbaguna yang reter dan dapat digunakan dalam beragam aplikasi. Python ah dimengerti dan digunakan. Python disebut sebagai bahasa pengguna dan ramah pemula belakangan ini. Python telah aritas sebagai bahasa yang ramah bagi pemula, dan telah ebagai bahasa pengantar yang paling populer.

ang diketik secara dinamis, Python sangat fleksibel. Selain itu, mudah memaafkan kesalahan, sehingga Anda masih bisa

mengompilasi dan menjalankan program Anda sampai Anda menemukan bagian yang bermasalah. Python adalah bahasa pemrograman yang fleksibel dan sederhana (Srinath, 2017). Python telah muncul sebagai bahasa yang paling banyak digunakan oleh para *Data Scientist* dan praktisi ML, dan semakin populer di berbagai disiplin ilmu (Molner et al., 2024).

1.3.11 Gemini API

Gemini API adalah antarmuka terprogram yang diekspos oleh Google AI yang memungkinkan pengembang untuk membangun aplikasi atau fitur bertenaga AI menggunakan model Gemini Google. Dengan API ini, pengembang dapat memanfaatkan model Gemini untuk fungsi-fungsi seperti pembuatan konten, agen dialog, rangkuman, sistem klasifikasi, pembangunan mesin pengolah gambar, dan banyak lagi. Hal ini memberikan kesempatan yang signifikan bagi para pengembang untuk mengintegrasikan kemampuan AI ke dalam aplikasi mereka tanpa perlu mengembangkan model khusus mereka sendiri (Adi, 2024).



Gambar 1. Alur kerja Gemini API *(Overholt, 2024)*

Gemini API terdiri dari dua elemen utama: sumber daya dan tipe data. Sumber daya didefinisikan sebagai entitas yang dapat berinteraksi dengan klien API. Contoh sumber daya termasuk model, operasi, dan model yang disesuaikan. Tipe data berfungsi untuk mendeskripsikan format standar data yang dipertukarkan antara n demikian memastikan bahwa format data yang dikirim dan diterima sesuai dengan format yang diharapkan (Adi, 2024).
ng didukung oleh Gemini API meliputi metode *list*, yang semua model yang tersedia; metode *get*, yang mengambil entu; metode *generateContent*, yang menghasilkan konten yang diberikan; metode *countTokens*, yang menghitung jumlah

token; dan metode *embedContent*, yang menghasilkan penyematan konten. Selain itu, metode *streamGenerateContent* memungkinkan pengumpulan respons dalam beberapa bagian. Setiap sumber daya model dalam API ini memiliki properti, termasuk nama model, versi model, batas *token input* dan *output*, dan suhu, yang menentukan tingkat keacakan respons model. Dengan pemahaman yang komprehensif tentang struktur dan fungsi API Gemini, pengembang dapat merancang *input* mereka secara lebih efektif untuk mendapatkan respons yang diinginkan dari model AI (Adi, 2024).

1.3.12 FastAPI

FastAPI merupakan salah *web framework* berbasis Python yang digunakan untuk membangun sebuah *Application Programming Interface* (API) (Suresh Babu et al., 2024). FastAPI menggunakan konsep *Asynchronous Server Gateway Interface* (ASGI) yang memungkinkan *endpoint* API pada FastAPI untuk dipanggil dan mengeksekusi perintah secara asinkron. FastAPI sendiri merupakan sebuah *framework* API yang sesuai untuk *workload machine learning*, terutama untuk menghasilkan sebuah platform *Machine Learning as a Service* (MLaaS) (Pamungkas et al., 2022).

FastAPI adalah kerangka kerja web modern yang cepat dan berkinerja tinggi, dirancang untuk membangun API menggunakan Python sesuai dengan standar Python (Ramírez, 2018). Fitur-fitur utamanya meliputi:

a. Cepat: Memiliki performa sangat tinggi, setara dengan NodeJS dan Go (berkat penggunaan Starlette dan Pydantic), menjadikannya salah satu *framework* Python tercepat saat ini.
b. Efisien dalam penulisan kode: Meningkatkan produktivitas pengembangan hingga 200% hingga 300%.
c. Mengurangi *bug*: Mengurangi sekitar 40% kesalahan yang disebabkan oleh pengembang.
d. Intuitif: Mendukung *editor* dengan fitur penyelesaian otomatis di mana-mana, sehingga mempercepat proses *debug*.
e. Mudah digunakan: Dirancang untuk mudah dipelajari dan digunakan, meminimalkan waktu yang dibutuhkan untuk membaca dokumentasi.
f. Ringkas: Meminimalkan duplikasi kode dengan berbagai fitur yang terintegrasi dalam setiap deklarasi parameter, sehingga mengurangi potensi *bug*.
g. Kuat: Menghasilkan kode yang siap untuk produksi, dengan dokumentasi natis.
ndar: Sepenuhnya kompatibel dengan standar terbuka untuk OpenAPI (sebelumnya dikenal sebagai Swagger) dan *JSON*

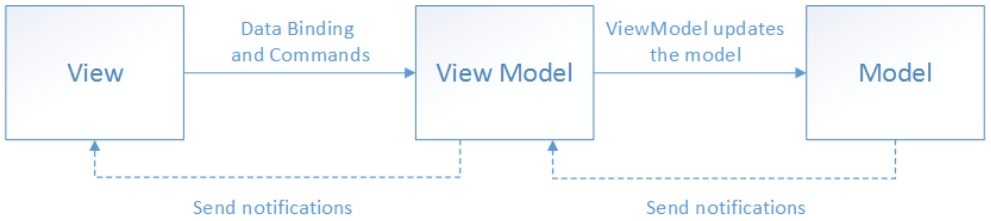
1.3.13 Ngrok

Ngrok adalah layanan *reverse proxy* global yang memungkinkan aplikasi dan layanan jaringan diakses dari mana saja tanpa perlu konfigurasi jaringan yang rumit. Ngrok bekerja dengan menerima lalu lintas dari internet ke "*endpoint*" yang telah ditentukan dan mentransmisikannya ke layanan *upstream* melalui agen perangkat lunak ringan yang terhubung dengan aman menggunakan koneksi TLS persisten. Dengan arsitektur ini, ngrok memungkinkan layanan dijalankan di berbagai *platform* seperti AWS, Azure, Heroku, atau bahkan di perangkat lokal seperti Raspberry Pi dan laptop tanpa perlu konfigurasi DNS, IP, sertifikat, atau *port* (Ngrok, 2024).

Ngrok memberikan berbagai manfaat seperti kemudahan dalam pengujian *webhook* dan demo situs lokal, serta memungkinkan pengembang menghubungkan API lokal mereka ke jaringan eksternal dengan aman. Di lingkungan produksi, ngrok dapat digunakan sebagai gerbang API, *load balancer*, atau *ingress controller* di Kubernetes, yang dilengkapi dengan fitur keamanan seperti OAuth, SAML, dan OpenID Connect. Dengan cara ini, ngrok tidak hanya memfasilitasi koneksi yang aman dan mudah, tetapi juga melindungi layanan dari serangan dan memastikan autentikasi yang kuat (Ngrok, 2024).

1.3.14 *Model-View-ViewModel* (MVVM)

Model-View-ViewModel (MVVM) adalah sebuah arsitektur perangkat lunak yang dirancang untuk memisahkan logika bisnis dan presentasi dari antarmuka pengguna (UI) aplikasi, sehingga memudahkan pengujian, pemeliharaan, dan pengembangan aplikasi. MVVM terdiri dari tiga komponen utama: *Model*, *View*, dan *ViewModel* (Microsoft, 2024).



Gambar 2. Komponen MVVM *(Microsoft, 2024)*

a. *View* bertanggung jawab untuk menampilkan *User Interface* (UI) aplikasi. Dalam MVVM, *View* dirancang agar lebih ramah bagi desainer sehingga bisa
sikan dengan mudah oleh desainer tanpa perlu melibatkan
kode.
kili data aplikasi, *Model* mengelola data dan logika bisnis tanpa
langsung dengan UI.
engelola *state* dan logika *View*. *ViewModel* berfungsi untuk
data dan operasi ke *View* serta mengatur perilaku dan interaksi

View dengan data, memastikan pemisahan logika presentasi dari UI (Anderson, 2012).

Manfaat utama MVVM adalah fleksibilitas dan kemudahan pengujian. *ViewModel* dapat berfungsi sebagai adaptor untuk *Model* yang ada, sehingga meminimalkan perubahan besar pada kode *Model*. Pengembang juga dapat membuat *unit test* untuk *ViewModel* dan *Model* tanpa melibatkan *View*. Selain itu, UI aplikasi dapat didesain ulang tanpa mengubah kode *ViewModel* dan *Model*, asalkan *View* diimplementasikan dengan XAML atau C#. Dengan demikian, desainer dapat fokus pada pembuatan *View*, sementara pengembang mengerjakan *ViewModel* dan *Model*, memungkinkan pekerjaan berjalan secara paralel (Microsoft, 2024).

1.3.15 *Black Box Testing*

Black box testing, juga dikenal sebagai pengujian fungsional, adalah teknik pengujian yang merancang kasus uji berdasarkan informasi dari spesifikasi. Dalam pengujian ini, penguji perangkat lunak tidak memiliki akses ke kode sumber internal dan tidak peduli dengan mekanisme internal sistem. Fokusnya hanya pada *output* yang dihasilkan sebagai respons terhadap *input* dan kondisi eksekusi yang dipilih (Nidhra & Dondeti, 2012).

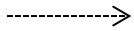
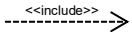
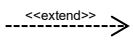
Penguji menganggap sistem sebagai "kotak hitam" yang tidak dapat dilihat isinya. Penguji hanya mengetahui bahwa informasi dapat dimasukkan ke dalam kotak hitam, dan kotak hitam akan mengirimkan sesuatu sebagai balasan. Pengujian ini dilakukan berdasarkan pengetahuan dari spesifikasi kebutuhan, di mana penguji memastikan bahwa *output* yang dihasilkan sesuai dengan yang diharapkan (Nidhra & Dondeti, 2012).

1.3.16 *Use Case Diagram*

Use case diagram adalah satu dari berbagai jenis diagram *Unified Modelling Language* (UML) yang menggambarkan hubungan interaksi antara sistem dan aktor. *Use Case* dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya (Juliarto, 2021). *Use case diagram* sendiri terdiri dari beberapa komponen.

Tabel 1. Komponen *use case diagram*

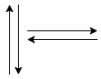
Simbol	Nama	Keterangan
	*Actor*	Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	Use Case	Abstraksi dan interaksi antara sistem dan aktor
	Association	Hubungan yang menunjukkan bagaimana aktor berinteraksi dengan *use case*

Simbol	Nama	Keterangan
	*Dependency*	Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan *use case*
	*Include*	Relasi *include* digunakan ketika sebuah *use case* selalu menyertakan *use case* lain
	*Extend*	Relasi extend digunakan ketika sebuah *use case* dapat memperluas fungsionalitas *use case* lain dalam kondisi tertentu

1.3.17 *Activity Diagram*

Activity diagram pada dasarnya menggambarkan macam-macam alir aktifitas yang akan dirancang dalam sebuah sistem. Dimana masing-masing alir memiliki awal, *decision* yang mungkin terjadi pada sistem, dan akhir dalam sistem tersebut. *Activity diagram* pada dasarnya memiliki struktur yang hampir mirip dengan *flowchart* atau diagram alir dalam perancangan sistem secara terstruktur. *Activity diagram* ini dibuat berdasarkan sebuah *use case* atau beberapa *use case* dalam *use case diagram* (Rahardjo, 2018).

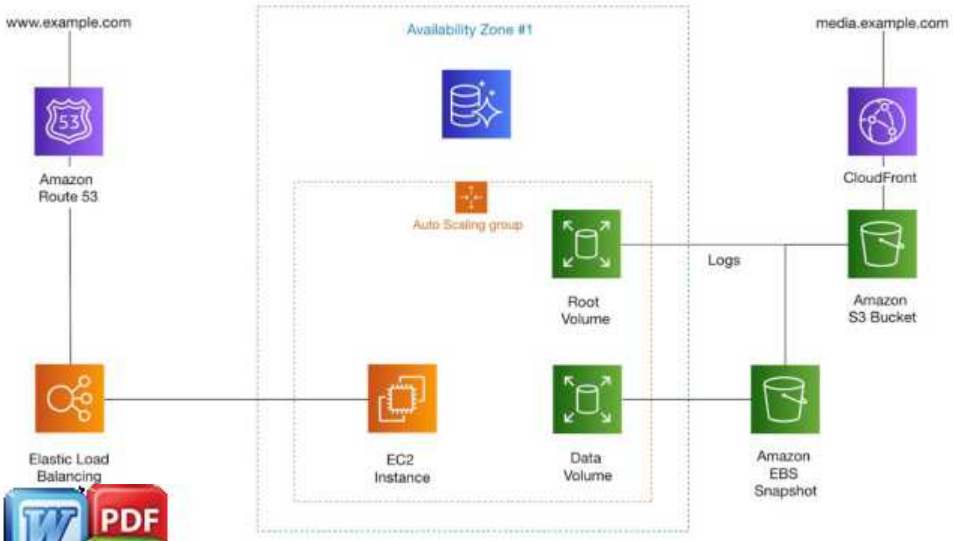
Tabel 2. Komponen *activity diagram*

Simbol	Nama	Keterangan
	*Initial Node*	Sebuah *activity diagram* memiliki sebuah status awal
	*Activity*	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
	*Connector*	Sebuah penghubung yang menghubungkan satu simbol dengan simbol lainnya
	*Decision*	Percabangan dimana ada pilihan aktivitas yang lebih dari satu
	End Node	Status akhir yang dilakukan sistem, sebuah *activity diagram* memiliki sebuah status akhir

1.3.18 *Architecture Diagram*

Diagram arsitektur adalah proses pembuatan representasi visual dari komponen sistem perangkat lunak. Dalam sistem perangkat lunak, istilah arsitektur mengacu pada beragam fungsi, implementasi, serta interaksinya satu sama lain. Karena perangkat lunak bersifat abstrak, diagram arsitektur secara visual menggambarkan berbagai pergerakan data di dalam sistem. Diagram arsitektur juga menyoroti cara perangkat lunak berinteraksi dengan lingkungan di sekitarnya (Amazon, 2023).

Diagram arsitektur memberikan berbagai manfaat, termasuk meningkatkan kolaborasi, mengurangi risiko, meningkatkan efisiensi, dan mendukung skalabilitas. Dengan memfasilitasi komunikasi antara pengembang dan desainer, diagram ini menciptakan pemahaman bersama tentang fungsionalitas sistem dan membantu tim mengembangkan komponen perangkat lunak yang efektif. Selain itu, diagram arsitektur membantu mengidentifikasi potensi risiko dalam pengembangan, memungkinkan tim untuk menangani masalah lebih awal dan mengurangi dampak masalah yang mungkin muncul. Diagram ini juga memberikan gambaran jelas tentang komponen sistem, sehingga pemangku kepentingan dapat dengan cepat mengidentifikasi dan mengatasi masalah, serta mendukung proses pemeliharaan dan pengembangan berkelanjutan. Diagram arsitektur memudahkan identifikasi cara untuk menskalakan sistem dengan efisien, membantu pemangku kepentingan mengenali hambatan dan mencari solusi yang tepat (Amazon, 2023).



rchitecture diagram (Amazon, 2023)

s diagram arsitektur yang digunakan untuk menggambarkan atu sistem. Pertama, diagram komponen memperlihatkan ri sistem serta hubungan antar komponen tersebut. Kedua,

diagram *deployment* menggambarkan konfigurasi fisik dari komponen perangkat keras dan perangkat lunak dalam sistem. Ketiga, diagram aliran data menunjukkan bagaimana data mengalir di dalam sistem dan antar komponen, yang membantu dalam memahami proses pengolahan dan pengiriman data.

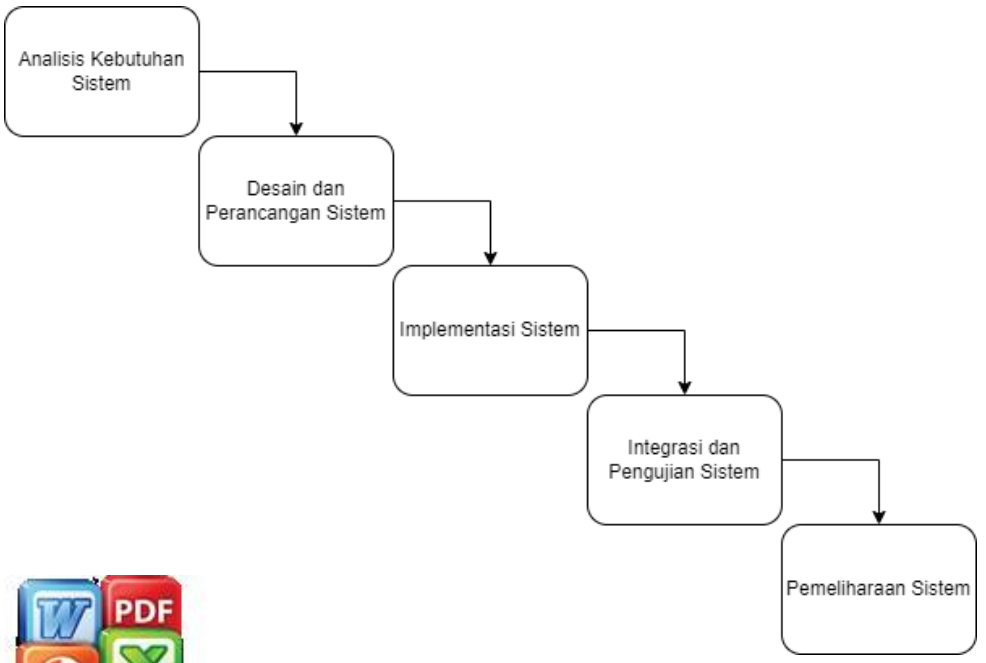
BAB II
METODOLOGI PENELITIAN

2.1 Tempat Penelitian

Studi ini sepenuhnya dilaksanakan secara independen di Lab Rekayasa Perangkat Lunak Universitas Hasanuddin Kota Makassar. Data yang diperlukan mencakup sampel dokumen seperti KTP dan SIM Republik Indonesia, sementara pembuatan model dilakukan di Google Colaboratory. Pengembangan sistem *backend* menggunakan Visual Studio Code. Selanjutnya, pengembangan aplikasi Android dilakukan dengan memanfaatkan Figma sebagai alat desain dan Android Studio untuk membangun aplikasi.

2.2 Metode Pengembangan

Metode pengembangan yang digunakan pada penelitian ini menggunakan pengembangan model Air Terjun (*waterfall*). Menurut Rosa dan Shalahuddin (2013) Model SDLC *waterfall* disebut sebagai model sekuensial linier atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*) (Dermawan & Hartini, 2017).



Vaterfall

*v*ali dengan tahap analisis kebutuhan sistem. Pada tahap ini, *u*an mendalam tentang kebutuhan pengguna dan tujuan proyek. *p* identifikasi masalah yang akan diselesaikan oleh perangkat

lunak, fungsi-fungsi yang diperlukan dan batasan-batasan proyek. Analisis kebutuhan merupakan tahap awal dalam pengembangan perangkat lunak di mana pengembang berusaha memahami apa yang harus dicapai oleh perangkat lunak (Pressman, 2010).

Kemudian, dilakukan desain dan perancangan sistem berdasarkan analisis tersebut. Pada tahap ini, perhatian diberikan kepada pengembangan kerangka kerja perangkat lunak sesuai dengan hasil analisis kebutuhan. Ini melibatkan perencanaan rinci mengenai cara perangkat lunak akan dibentuk, termasuk komponen seperti struktur data, algoritma, serta tampilan antarmuka pengguna. Penelitian ini meliputi desain dan perancangan yang mencakup *use case diagram*, *activity diagram*, *architecture diagram*, desain tampilan antarmuka dalam bentuk *wireframe*, perancangan *backend* aplikasi, serta implementasi *machine learning*.

Tahap berikutnya adalah proses pengimplementasian. Pada tahap ini, dilibatkan dalam proses pembuatan perangkat lunak berdasarkan desain yang telah dirancang sebelumnya. Tahap ini merupakan saat di mana ide dan perencanaan diubah menjadi kode yang dapat dieksekusi oleh komputer. Dalam penelitian ini dilakukan pembuatan model *machine learning*, pembuatan sistem *backend*, dan pembuatan aplikasi Android.

Setelah itu dilakukan integrasi dan pengujian sistem. Pada tahap ini, dilakukan pengujian perangkat lunak guna memverifikasi bahwa kinerjanya sesuai dengan spesifikasi yang telah ditetapkan. Proses ini melibatkan pengidentifikasian dan perbaikan masalah (*bug*) yang mungkin muncul, serta memastikan bahwa perangkat lunak memenuhi kebutuhan yang telah ditetapkan. Dalam penelitian ini, metode pengujian yang dilakukan adalah *black box testing*.

Kemudian pengembangan aplikasi diakhiri dengan *deployment* dan pemeliharaan sistem. Tahap ini berfokus pada pemeliharaan perangkat lunak setelah perilisannya. Ini termasuk perbaikan *bug*, peningkatan fitur, dan dukungan pelanggan. Pemeliharaan adalah tahap berkelanjutan yang diperlukan untuk memastikan perangkat lunak tetap relevan dan berfungsi setelah perilisannya (Pressman, 2010).

2.3 Waktu dan Jadwal Penelitian

Penelitian yang bertujuan mengembangkan aplikasi berlangsung selama periode lima bulan. Informasi mengenai jadwal penelitian tersebut tersedia dalam tabel di bawah ini.

Tabel 3. Jadwal penelitian

Mei	Juni	Juli	Agustus	September

Kegiatan	Mei	Juni	Juli	Agustus	September
Desain dan Perancangan Sistem					
Implementasi Sistem					
Integrasi dan Pengujian Sistem					
Pemeliharaan Sistem					

2.4 Analisis Kebutuhan Sistem

2.4.1 Analisis Kebutuhan Fungsional

Pada penelitian ini, penulis melakukan analisis kebutuhan fungsional untuk mengetahui fungsi-fungsi yang ada pada perangkat lunak yang dirancang. Adapun analisis kebutuhan fungsional pada penelitian ini disajikan pada tabel berikut.

Tabel 4. Kebutuhan fungsional

No.	Fungsional	Deskripsi Fungsional
1.	Aplikasi mampu menangkap gambar	Aplikasi menggunakan sumber daya Android untuk mengaktifkan kamera dan menangkap gambar tersebut untuk diolah. Pengguna juga dapat memilih gambar melalui media galeri.
2.	Aplikasi mampu memotong (*cropping*) gambar	Aplikasi dapat melakukan *cropping* pada gambar yang telah diambil atau dipilih, agar dapat lebih mudah diproses.
	pu membaca a gambar	Menggunakan fitur *Text Recognition* dari ML Kit, aplikasi mampu mengubah setiap teks yang ada pada gambar hasil tangkapan menjadi teks *string*.
	pu mengirim dan pons ke API	Dengan menggunakan FastAPI yang diintegrasikan dengan Ngrok agar API

No.	Fungsional	Deskripsi Fungsional
		dapat diakses secara publik, aplikasi dapat mengirim hasil ekstraksi teks ke API dan menerima respons dalam skema JSON.
5.	Aplikasi mampu mengklasifikasikan jenis dokumen	Dengan menggunakan model yang telah dibuat menggunakan algoritma BERT dan diintegrasikan ke dalam API, aplikasi dapat mengenali jenis dokumen berdasarkan hasil ekstraksi teks.
6.	Aplikasi mampu menampilkan hasil prediksi dan isi dokumen dalam bentuk teks pada sebuah *field text input*	Hasil prediksi menampilkan jenis dokumen beserta kontennya, yang diperoleh melalui inferensi dari respons API dalam skema JSON. Konten dokumen tersebut kemudian ditampilkan dalam bentuk teks pada sebuah *field input* teks.
7.	Aplikasi mampu menyalin isi konten dokumen	Hasil prediksi dan isi konten dokumen berupa teks dapat disalin oleh pengguna dengan menekan ikon salin pada setiap *field input* teks, atau menyalin keseluruhan teks sekaligus dengan menekan satu tombol.

2.4.2 Analisis Kebutuhan Perangkat Lunak

Penelitian ini menggunakan beberapa perangkat lunak yang terdiri dari berbagai pustaka dan IDE untuk keperluan pembuatan model *machine learning* hingga desain dan pengembangan aplikasi.

Tabel 5. Kebutuhan perangkat lunak

Nama Perangkat Lunak	Tentang Perangkat Lunak
Android Studio	*Integrated Development Environment* (IDE) resmi untuk pengembangan aplikasi Android. Berdasarkan pada editor kode yang kuat dan alat pengembang dari IntelliJ IDEA.
Figma	Alat desain yang digunakan untuk membuat *mockup*/*prototype user interface*, *website*, dan aplikasi *mobile*.
y	Digunakan sebagai media platform pemrosesan data dan *training* model.

Nama Perangkat Lunak	Tentang Perangkat Lunak
Visual Studio Code	*Code editor* yang digunakan dalam membangun *Application Programming Interface* (API).
ML Kit	*Software Development Kit* (SDK) seluler yang menghadirkan keahlian *machine learning* Google di perangkat ke aplikasi Android dan iOS. Menggunakan *Application Programming Interface* (API) *Text Recognition v2* yang disediakan ML Kit. ML Kit *Text Recognition v2* API dapat mengenali teks dalam himpunan karakter China, Devanagari, Jepang, Korea, dan Latin. API juga dapat digunakan untuk mengotomatiskan tugas entri data seperti memproses kartu kredit, tanda terima, dan kartu nama.
PyTorch	PyTorch adalah pustaka pembelajaran mesin yang berbasis pada *library* Torch, digunakan untuk aplikasi seperti *computer vision* dan *natural language processing*.
Gemini API	*Application Programming Interface* (API) dari Google yang menggunakan Gemini AI, digunakan dalam melakukan *prompting* untuk mendapatkan konteks *output* sesuai kebutuhan.
FastAPI	*Framework* Python yang digunakan dalam membangun *Application Programming Interface* (API).
Ngrok	*Proxy server* untuk membuat atau membuka jaringan *private* melalui NAT atau *firewall*, lalu Menghubungkan *localhost* ke internet dengan *tunnel* yang aman.
Kotlin	Bahasa pemrograman yang diketik secara statis yang menargetkan *Java Virtual Machine* (JVM), Android, JavaScript, dan Native.
Python	Bahasa Pemrograman yang serbaguna dan banyak digunakan dalam aplikasi web, pengembangan perangkat lunak, *data science*, dan *Machine Learning* (ML).

2.4.3 Analisis Kebutuhan Perangkat Keras

Komputer. Perangkat keras yang digunakan berperan sebagai media pengembangan aplikasi, model kecerdasan buatan, dan server dari API yang akan dibuat.

Tabel 6. Spesifikasi komputer yang digunakan

Prosesor	11th Gen Intel® Core™ i7-11600H @ 2.90GHz 2.92 GHz
Graphic Card	NVIDIA GeForce RTX 3050
RAM	16 GB
Penyimpanan	512 GB SSD
Sistem Operasi	Windows 11 Home version 22H2

Smartphone. Perangkat seluler yang digunakan bertindak sebagai media tujuan utama aplikasi. Aplikasi yang telah selesai akan diuji dan digunakan pada perangkat seluler tersebut. Detail mengenai spesifikasi perangkat Android yang digunakan adalah sebagai berikut:

Tabel 7. Spesifikasi *smartphone* yang digunakan

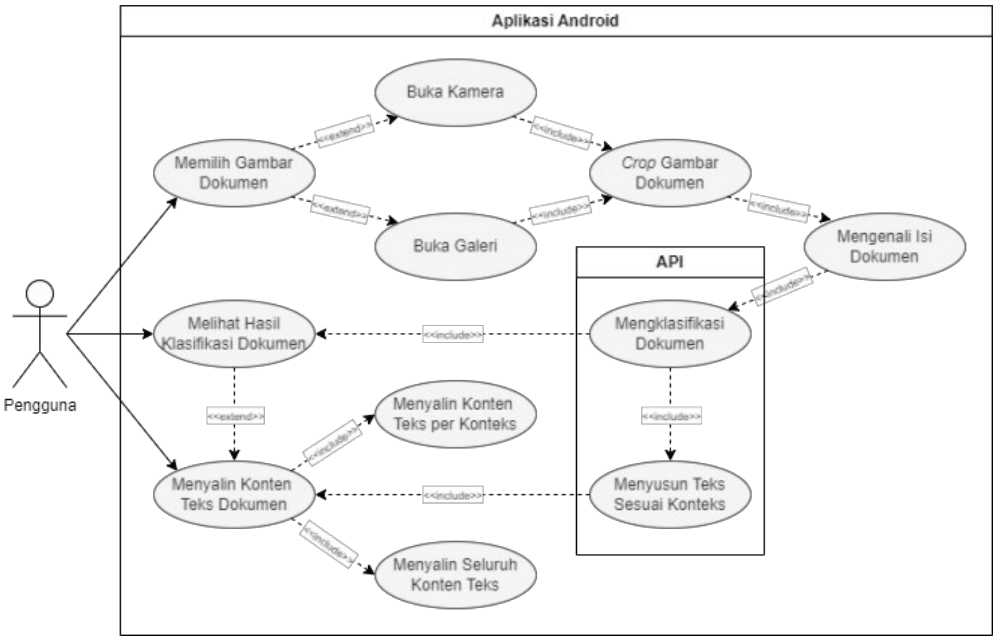
Merk/Model	Vivo V21 5G (V2050)
Versi Android	Android 13
Prosesor	MTK Dimensity 800U (2,4 GHz *Octa-core*)
RAM	8 GB + 4 GB
Penyimpanan Internal	128 GB
Layar	6.44" 2404×1080 (FHD+) *Capacitive multi-touch*, AMOLED
	Kamera Depan: 44MP OIS, f/2.0

	Kamera Belakang: 64MP OIS f/1.79 + 8MP f/2.2 (Super *wide-angle*) + 2MP f/2.4 (*Super macro*)

2.5 Desain dan Perancangan Sistem

2.5.1 *Use Case Diagram*

Untuk menggambarkan interaksi antara pengguna dan aplikasi "Klasifikasi dan Digitalisasi Dokumen" yang penulis rancang, penulis menggunakan *Use Case* Diagram. *Use Case* Diagram ini membantu penulis memahami skenario penggunaan utama dan fungsi utama aplikasi.

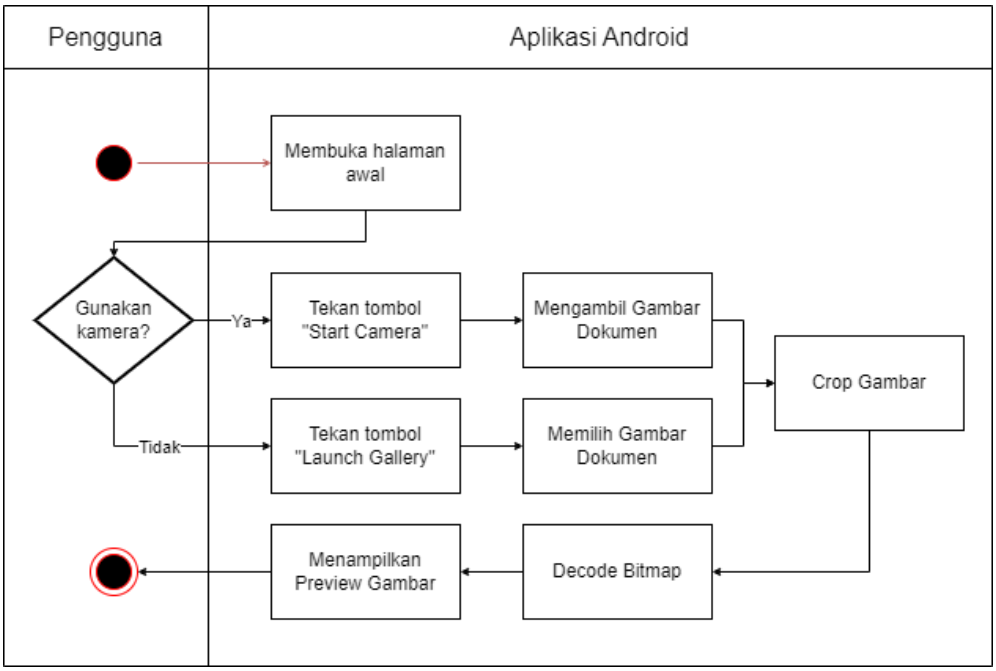


Gambar 5. *Use Case diagram*

Aktor, yang merupakan pengguna, dapat memilih gambar dokumen dengan menggunakan kamera atau galeri. Setelah pengambilan atau pemilihan gambar, pengguna dapat melakukan proses pemotongan (*cropping*) pada gambar dokumen. Setelah [illegible]guna dapat memulai pemrosesan dengan mengenali [illegible] dari dokumen tersebut. Hasil ekstraksi kemudian dikirim ke API [illegible]asi dokumen menggunakan model BERT, serta penyusunan [illegible]ya melalui *prompting* ke Gemini API. Respons dari API akan [illegible]si sehingga pengguna dapat melihat hasil klasifikasi dan konten [illegible]. Setelah itu, pengguna bisa menyalin konten per konteks atau [illegible]umen.

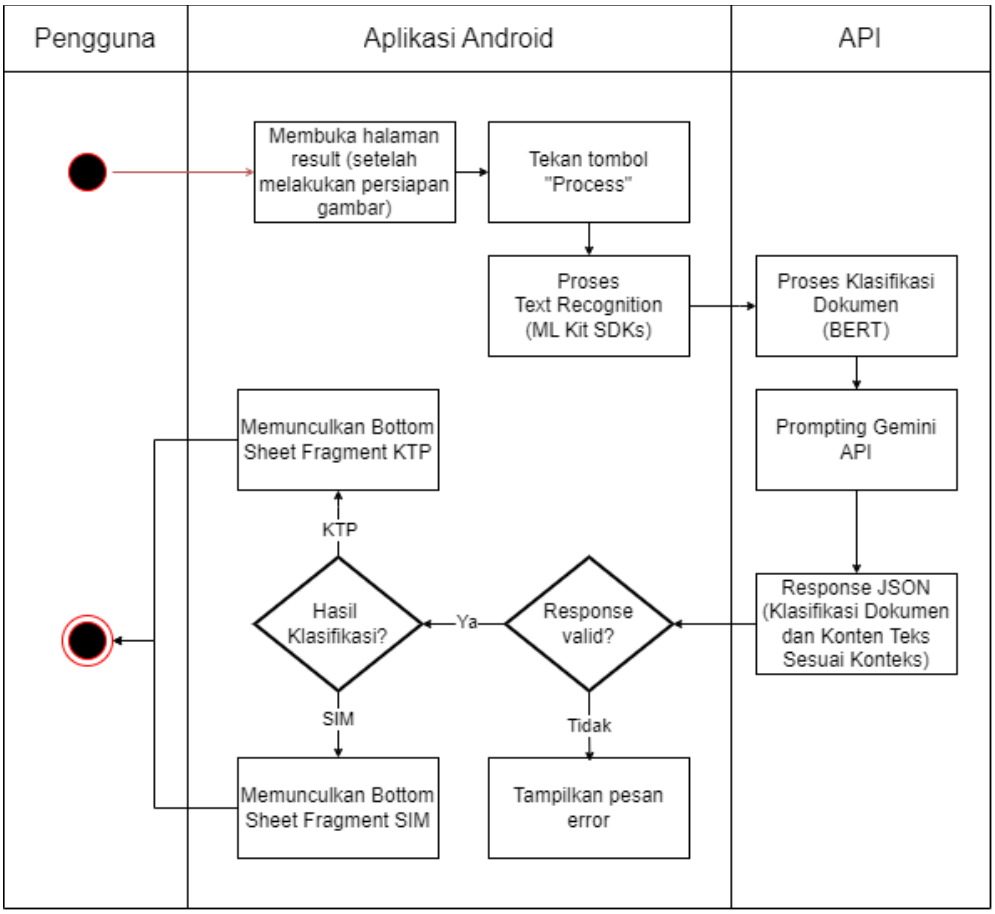
Hubungan antar *use case* digambarkan dengan notasi <<*extend*>>, yang menunjukkan perluasan fungsionalitas fitur pada kondisi tertentu, seperti pengguna harus membuka kamera atau galeri sebelum memilih gambar. Notasi <<*include*>> menjelaskan dependensi fitur, di mana pengguna harus memilih gambar dokumen terlebih dahulu sebelum dapat melakukan pemotongan (*cropping*) dan proses-proses selanjutnya. Diagram ini memberikan gambaran umum alur penggunaan aplikasi dan bagaimana fitur-fitur saling terkait.

2.5.2 *Activity Diagram*



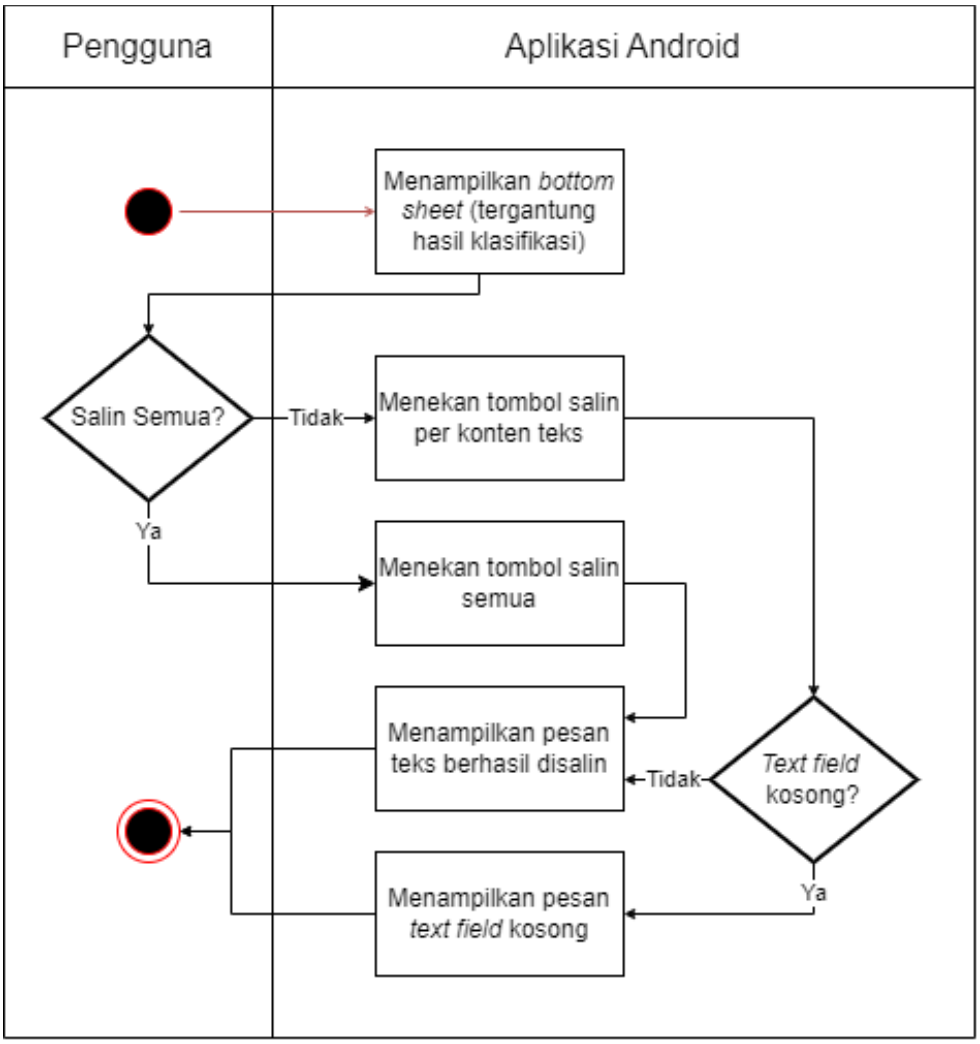
Gambar 6. *Activity diagram* mempersiapkan gambar dokumen

***Activity diagram* mempersiapkan gambar dokumen.** Pada *use case* ini, pengguna dapat mengambil gambar dokumen menggunakan kamera atau memilih gambar dokumen dari galeri perangkat kemudian pengguna dapat melakukan pemotongan pada gambar dokumen. Gambar yang telah dipotong tadi di-*decode* agar dapat ditampilkan kepada pengguna.



Gambar 7. *Activity diagram* proses klasifikasi dokumen

***Activity diagram* proses klasifikasi dokumen.** Pada *use case* ini pengguna dapat melakukan proses klasifikasi dokumen dengan menekan tombol untuk mulai melakukan proses pengklasifikasian. Selanjutnya, gambar dokumen diproses menggunakan *text recognition v2* pada ML Kit SDKs untuk mengekstraksi teks pada dokumen. Hasil ekstraksi teks tersebut dikirim ke API untuk diklasifikasi menggunakan model BERT dan untuk melakukan penyusunan konteks konten teks dokumen dengan melakukan *prompting* menggunakan Gemini API. Respons dari API dikembalikan ke aplikasi dalam bentuk JSON. Jika respons dari API tidak valid, maka sistem menampilkan pesan *error* berupa *toast*. Jika respons dari API valid, maka sistem menampilkan *bottom sheet fragment* sesuai hasil klasifikasi

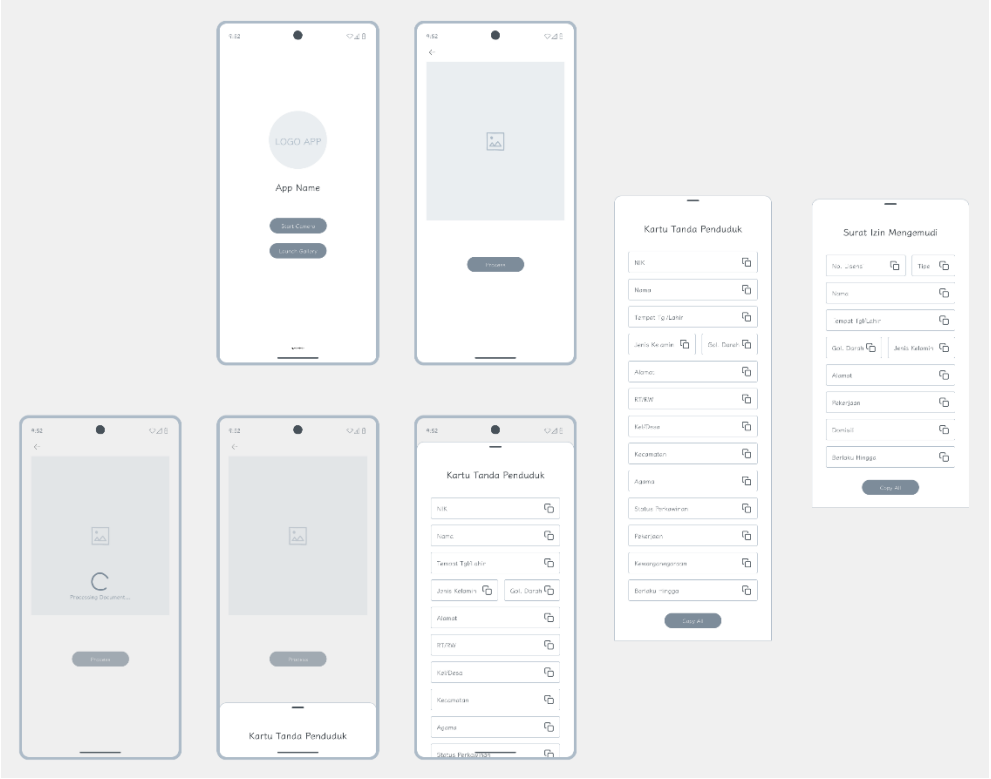


Gambar 8. *Activity diagram* menyalin teks dokumen

***Activity diagram* menyalin teks dokumen.** Pada *use case* ini pengguna dapat menyalin isi konten pada dokumen dengan menekan tombol salin semua untuk menyalin seluruh isi konten dokumen. Apabila pengguna hanya ingin menyalin bagian konten tertentu saja, pengguna dapat menekan tombol salin di samping tiap masing-masing bagian, akan tetapi jika *text field* kosong, maka sistem menampilkan pesan *error* berupa *toast*.

2.5.3 Perancangan Tampilan Antarmuka Aplikasi

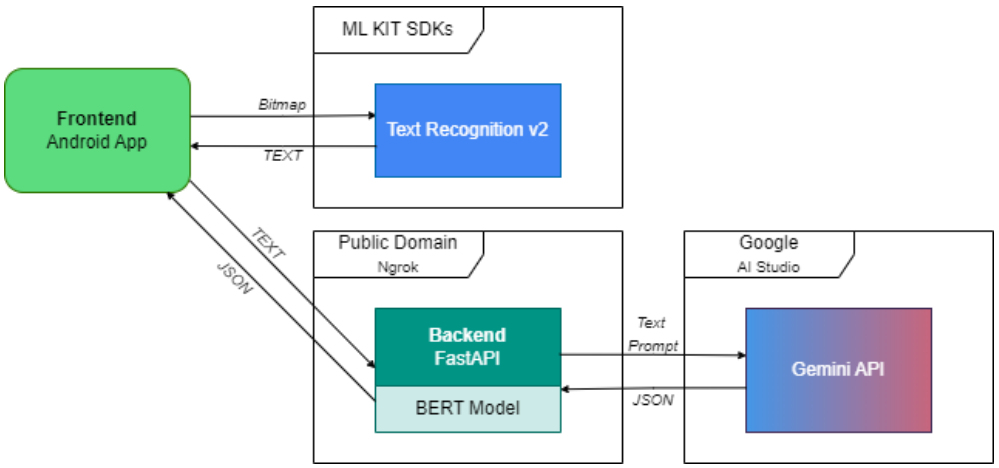
Sebelum memulai pengembangan aplikasi, penulis merancang *User Interface* (UI) melalui situs Figma. Proses perancangan dimulai dengan pembuatan *wireframe* aplikasi yang menjadi dasar desain dalam hal tata letak dan alur penggunaan.



Gambar 9. *Wireframe* aplikasi

Wireframe ini memberikan gambaran visual mengenai tata letak dan tampilan aplikasi yang dikembangkan. *Wireframe* mencakup elemen-elemen seperti halaman beranda (*home*), halaman hasil (*result*), serta *bottom sheet fragment* yang berbeda untuk setiap jenis dokumen. *Wireframe* berfungsi untuk membantu menentukan penempatan elemen-elemen UI di setiap halaman aplikasi.

2.5.4 Diagram Arsitektur Aplikasi



Gambar 10. *Architecture diagram* aplikasi

Aplikasi ini terdiri dari empat *environment*, yaitu *client*, ML KIT SDKs, server lokal dengan domain publik, dan Google AI Studio. *Client* berupa aplikasi Android, dengan *Text Recognition v2* pada ML KIT SDKs yang berfungsi untuk melakukan ekstraksi teks dari dokumen. Model *machine learning* diintegrasikan dengan FastAPI dan dihubungkan melalui Ngrok untuk mendapatkan domain publik. Gemini API pada Google AI Studio digunakan untuk melakukan *prompting* berdasarkan hasil klasifikasi dan ekstraksi teks dokumen.

Aplikasi Android bertindak sebagai *client* atau antarmuka yang digunakan oleh pengguna, sementara komponen lainnya berjalan di latar belakang dan tidak terlihat oleh pengguna. Koneksi antara *client* dan *backend* menggunakan metode *Request* dan *Response* JSON, di mana aplikasi dapat mengirimkan dan menerima data dalam bentuk objek JSON. *Backend* dijalankan di server lokal yang dihubungkan dengan Ngrok yang dapat diakses oleh *client* menggunakan domain publik dari Ngrok. Karena aplikasi masih dalam tahap pengembangan, *backend* dijalankan secara lokal. Namun, setelah aplikasi dirilis, seluruh arsitektur di-*deploy* agar dapat diakses secara publik tanpa perlu menjalankan *backend* secara lokal.

FastAPI yang dijalankan secara lokal pada komputer akan menampung model BERT. Peran utama *backend* adalah untuk melakukan inferensi dari model *machine learning* BERT, yang kemudian diakses oleh *client* (aplikasi Android) melalui server lokal yang terhubung ke Ngrok.

Model *Machine Learning*

an adalah BERT (*Bidirectional Encoder Representations from* satu algoritma dalam kategori *transformer*. Algoritma BERT mahami konten teks yang diekstraksi dari gambar dokumen. ks ini kemudian digunakan untuk mengklasifikasikan dokumen

ke dalam kategori yang sesuai. Model BERT dilatih menggunakan teknik *fine-tuning*. *Fine-tuning* adalah teknik yang digunakan dalam *deep learning* dan *transfer learning*, di mana bobot dari model yang telah dilatih sebelumnya (*pre-trained model*) dilatih kembali menggunakan data baru untuk menjalankan tugas kedua yang serupa (*fine-tuned model*). Proses ini melibatkan pengambilan model yang sudah dilatih untuk satu tugas tertentu, kemudian menyesuaikan model tersebut agar dapat menjalankan tugas kedua yang serupa.

Pre-trained model yang digunakan adalah "*bert-base-multilingual-cased*" yang dikembangkan oleh Devlin et al. pada tahun 2018 dan diambil melalui situs Hugging Face. Selanjutnya, model ini dilatih lebih lanjut menggunakan *dataset* sampel dokumen KTP dan SIM, yang berisi gambar dokumen *dummy* SIM dan KTP yang dibuat di Figma.

Data yang digunakan dalam penelitian ini merupakan kumpulan gambar dokumen dari berbagai jenis, seperti Kartu Tanda Penduduk (KTP) dan Surat Izin Mengemudi (SIM). Karena data yang digunakan merupakan data sensitif, penulis membuat kumpulan sampel gambar *dummy* dokumen tersebut di aplikasi Figma.

Setelah melakukan *fine-tuning model*, langkah selanjutnya adalah evaluasi kinerja model. Pengevaluasian model dilakukan menggunakan *confusion matrix*, dan hasil pelatihan model dievaluasi dengan metrik *f1-score*. Model kemudian diintegrasikan ke dalam server lokal menggunakan *library* FastAPI, yang dihubungkan dengan Ngrok. API ini diakses pada sistem Android menggunakan *library* Retrofit.