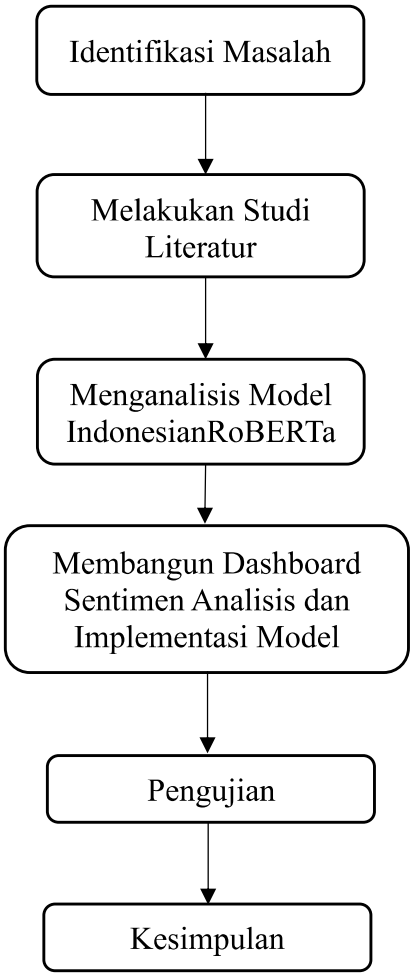
BAB III

METODOLOGI PENELITIAN

3.1. Desain Penelitian

Pada sub bab ini, penulis akan membahas mengenai desain penelitian yang menjadi kerangka utama dalam pelaksanaan penelitian ini. Desain penelitian berfungsi sebagai panduan dalam mengarahkan setiap langkah penelitian agar berjalan secara sistematis dan terencana. Mulai dari tahap awal yaitu identifikasi masalah hingga tahap akhir pengujian dan penarikan kesimpulan.



Gambar 3.1 Alur Penelitian

(**Sumber:** Penelitian 2025)

1. Identifikasi masalah

Pada tahap ini, penulis mengidentifikasi masalah terkait sentimen atau ulasan para pengguna aplikasi Instagram di *Google Play Store*. Penelitian ini Bertujuan untuk mengeksplorasi tantangan yang dihadapi oleh pengguna atau organisasi dalam memahami dan memanfaatkan data sentimen pengguna aplikasi Instagram di *Google Play Store*. Fokus utama penulis adalah pada kebutuhan akan alat yang dapat secara otomatis mengumpulkan, menganalisis, dan menyajikan sentimen pengguna secara efektif.

2. Melakukan studi literatur

Studi literatur dilakukan untuk mengumpulkan informasi dan teori yang relevan dengan penelitian ini. Ini mencakup kajian tentang teknik-teknik *Natural Language Processing* (NLP), model yang menggunakan arsitektur algoritma transformer seperti BERT dan *RoBERTa*, serta penelitian terdahulu yang telah dilakukan di bidang analisis sentimen. Studi literatur ini membantu penulis dalam memahami ruang lingkup penelitian dan membangun dasar teori yang kuat untuk pengembangan *dashboard* analisis sentimen.

3. Menganalisa model *Indonesian-RoBERTa*:

Tahap ini melibatkan analisis mendalam terhadap model *Indonesian-RoBERTa* yang akan digunakan dalam penelitian ini. Peneliti akan mempelajari arsitektur model, cara kerja, serta kekuatan dan kelemahannya dalam analisis sentimen bahasa Indonesia. Analisis ini penting untuk memastikan bahwa model yang digunakan adalah yang paling sesuai untuk kebutuhan penelitian dan dapat memberikan hasil yang optimal.

4. Pembangunan *dashboard* dan implementasi model

Pada tahap ini penulis akan membangun sebuah *dashboard* yang akan menjadi alat untuk untuk menagambil data sentimen pengguna aplikasi instagram dari *Google Play Store, dashboard* akan di bangun menggunakan *framework* laravel yang akan di integrasikan dengan program *scraping* yang berguna untuk mengambil data sentimen, model *Indonesian-RoBERTa* juga akan di implementasikan pada program *scraping* agar data yang di ambil dapat langsung di analisis. Pembangunan *dashboard* mencakup pengkodean, integrasi model, dan penyajian data yang mudah dipahami.

5. Pengujian

Setelah pengembangan *dashboard* dan implementasi model selesai, pengujian akan dilakukan untuk mengevaluasi kinerja sistem secara keseluruhan. Pengujian ini mencakup validasi akurasi model, keandalan program *scarping*, serta *User experience* dari *dashboard* yang dibangun. Hasil pengujian ini akan menentukan apakah sistem yang dikembangkan memenuhi tujuan penelitian atau masih perlu dilakukan perbaikan.

6. Kesimpulan

Tahap terakhir dalam desain penelitian adalah menarik kesimpulan berdasarkan hasil pengujian dan analisis data. Kesimpulan ini akan mencakup temuan utama dari penelitian, kontribusi yang diberikan oleh penelitian ini, serta rekomendasi untuk penelitian lebih lanjut atau pengembangan sistem di masa depan. Dokumentasi kesimpulan ini juga akan membantu dalam diseminasi hasil penelitian kepada komunitas yang lebih luas

3.2. Metode Pengumpulan data

Pengumpulan data menjadi salah satu tahapan penting dalam sebuah penelitian, karena kualitas data yang dikumpulkan akan sangat berpengaruh terhadap hasil akhir analisis. Pada penelitian ini penulis menggunakan *pre-trained model Indonesian-RoBERTa* yang dimana model tersebut merupakan model yang sudah dilatih sebelumnya dan siap digunakan, dan pada penelitian ini akan dijelaskan sumber dataset yang digunakan untuk melatih *pre-trained model Indonesian-RoBERTa* tersebut. penulis juga mengumpulkan dataset untuk melakukan *finetuning* kepada model *pre-trained model Indonesian-RoBERTa* dengan tujuan untuk mengkontekstualisasikan model *Indonesian-RoBERTa* terhadap bahasa dan gaya tulis spesifik yang khas pada platform sosial media, sehingga model dapat memahami nuansa dan ekspresi sentimen secara lebih tepat sesuai dengan domain data yang digunakan.

3.2.1. Dataset *Finetuning*

Pada penelitian ini penulis tidak menggunakan data primer, karena data yang yang dikumpulkan pada peneltian ini tidak menggunakan metode seperti wawancara, survei, ataupun obervasi melainkan menggunakan data skunder yang diunduh dari kaggle sebuah *platform* penyimpanan *public dataset*. Dataset yang digunakan merupakan kumpulaan *tweet* yang dikumpulkan dari platform twiter atau X pada masa PPKM dengan total data mencapai 23.645 *tweet*, total keselurahan data tersebut merupakan data mentah yang sudah diberi label tetapi belum dilakukan *processing* data.

Tabel 3.1 Contoh Data Mentah *Tweet* PPKM

date	*User*	*tweet*	sentiment
03-07-2021	_hayeon	Mulai besok PPKM darurat diterapkan. Mohon semua mengikuti aturan pemerintah. Jgn ada lagi yg ngeyel.	1
04-08-2021	_agus_b	Alhamdulillah, PPKM sudah level 3. Semoga kita bisa segera kembali normal.	1
05-09-2021	_budi_c	Meskipun PPKM sudah level 2, tetap patuhi protokol kesehatan ya.	1
06-10-2021	_cici_d	PPKM sudah level 1. Saya senang sekali, semoga ekonomi segera pulih.	1
07-11-2021	_dono_e	Ayo vaksin, biar PPKM segera dicabut.	1
08-12-2021	_eko_f	Meskipun sudah libur nataru, tetap jaga prokes ya. Jangan sampai PPKM lagi.	1
09-01-2022	_fafa_g	Semoga tahun ini tidak ada PPKM lagi.	1
10-02-2022	_gaga_h	Tolong dong kl PPKM, pabrik2 yg non esensial/non kritikal jgn	0

		dikasih ijin operasional. Percuma resto, mall, tmpt ibadah pd tutup, tp kl kantor2 n pabrik2 msih buka full, sama aja boong. Angka kasus pasti msih ttp tinggi.	
11-03-2022	_hadi_p	Saya sudah jenuh dengan PPKM. Kapan ini berakhir?	0
12-04-2022	_indah_s	Klo memang PPKM diperpanjang lagi, tolong pak, perhatikan nasib rakyat kecil. Jgn cuma bisa bikin aturan, tapi gak ada solusi. Rakyat lapar pak!	0
13-05-2022	_joko_w	PPKM ini bikin stress. Saya sudah di rumah terus. Kapan bisa keluar?	0
14-06-2022	_kiky_m	PPKM level 4 diperpanjang lagi. Gak ada solusi lain? Ini makin memberatkan masyarakat.	0
15-07-2022	_lisa_f	Dengan adanya PPKM level 4, banyak masyarakat yang kehilangan pekerjaan. Pemerintah harus lebih peka terhadap kondisi rakyat. Jangan biarkan rakyat menderita.	0

16-08-2022	_marwan_r	PPKM diperpanjang lagi. Saya sudah tidak punya harapan. Tolong pemerintah, jangan biarkan kami mati kelaparan.	0
17-09-2022	_nita_p	PPKM diperpanjang, kerjaan saya jadi gak ada. Tolong pemerintah, jangan diam aja.	0
18-10-2022	_ocha_s	Saya sudah jenuh dengan PPKM. Kapan ini selesai?	0
19-11-2022	_putra_b	PPKM level 4 diperpanjang. Masyarakat makin menderita. Tolong pemerintah, jangan tutup mata.	0
20-12-2022	_qiqi_r	Semoga PPKM ini tidak diperpanjang lagi. Saya sudah tidak sanggup. Ekonomi saya hancur.	0
21-01-2023	_rara_d	PPKM level 4 ini sangat menyiksa. Saya kehilangan pekerjaan. Tolong pemerintah, bantu kami.	0
22-02-2023	_santi_a	PPKM diperpanjang lagi, saya udah gak punya uang. Mau makan apa besok?	0

23-03-2023	_titi_w	PPKM ini sangat merugikan. Saya kehilangan pekerjaan. Tolong pemerintah, perhatikan rakyat.	0
24-04-2023	_udin_s	PPKM diperpanjang lagi. Gak ada solusi lain? Ini makin memberatkan masyarakat.	0
25-05-2023	_vita_n	PPKM Darurat diperpanjang, makin menderita masyarakat. Tolonglah, jangan hanya memikirkan pemerintah.	0
26-06-2023	_wahyu_k	Pak, tolong donk kebijakannya ditinjau ulang. Dengan adanya PPKM darurat ini, masyarakat kecil makin menderita. Tolong dengarkan keluhan rakyat. Jangan cuma bisa bikin aturan doang.	0
27-07-2023	_yuni_m	Tinjau ulang dong aturan PPKM darurat ini. Ini sangat merugikan bagi masyarakat, terutama sektor ekonomi. Seharusnya ada solusi lain yang bisa diterapkan. Jangan sampai masyarakat menderita.	0

28-08-2023	_zaki_a	Jadi, apakah PPKM Darurat ini hanya formalitas? Atau, ini upaya pemerintah agar ekonomi Indonesia tetap jalan? Jadi, apakah kita diminta untuk berdamai dengan keadaan?	0
29-09-2023	andikha_putra	Tolong dong kl PPKM, pabrik2 yg non esensial/non kritikal jgn dikasih ijin operasional. Percuma resto, mall, tmpt ibadah pd tutup, tp kl kantor2 n pabrik2 msih buka full, sama aja boong. Angka kasus pasti msih ttp tinggi.	0
30-10-2023	bambang_s	Dengan adanya PPKM Darurat, ekonomi makin hancur. Banyak yang di PHK, banyak yang usaha nya bangkrut.	0
31-11-2023	dian_rini	Semoga PPKM ini segera berakhir. Saya rindu hidup normal.	0
01-12-2023	faisal_a	Semoga PPKM ini segera berakhir. Saya rindu jalan-jalan. Semangat, Indonesia!	1

(Sumber: Kaggle)

3.2.2. Dataset model *Indonesian-RoBERTa*

Model *Indonesian-RoBERTa* adalah salah satu model pemrosesan bahasa alami (*Natural Language Processing* atau NLP) yang dirancang khusus untuk bahasa Indonesia. Model ini memiliki kemampuan untuk melakkukan berbagai tugas NLP salah satunya adalah analisis sentimen.

1. Spesifikasi Model

Tabel 3.2 Informasi Model *Indonesian-RoBERTa*

Model	#params	Arch.	*Training*/Validation data (*text*)
Indonesian-RoBERTa-base-sentiment-classifier	124M	*RoBERTa Base*	SmSA

(Sumber: HuggingFace)

a) Model: *Indonesian-RoBERTa-base-sentiment-classifier*

b) Jumlah Parameter (*#params*): 124juta parameter

c) Arsitektur (*Arch.*): *RoBERTa*

d) Data Latihan/Validasi (*Training/Validation* Data): SmSA

Dengan 124 juta parameter, model *Indonesian-RoBERTa* menjadi model yang memiliki peforma yang sangat baik, arsitektur transformers yang digunakan dalam model ini sama dengan yang terdapat pada model *RoBERTa* yang telah terbukti efektif dalam berbagai tugas NLP. Dataset yang digunakan untuk melatih dan memvalidasi model ini adalah dataset SmSA, dataset ini berisi kalimat-kalimat opini berbahasa Indonesia yang telah diberi label sentimen, yaitu positif, netral atau negatif (Wilie et al. 2024).

Dataset yang digunakan untuk pelatihan *pre-trained model Indonesian-RoBERTa* adalah dataset SmSA, dataset ini berisi kalimat-kalimat opini berbahasa indornesia yang telah diberi label sentimen, yaitu positif, netral, dan negatif, dataset ini biasa sering digunakan untuk mengembangkan model NLP untuk melakukan tugas sentimen analisis. SmSA dikembangkan untuk mendukung riset dan pengembangan model NLP dalam bahasa Indonesia, khususnya dalam klasifikasi sentimen, dataset ini sering digunakan dalam pelatihan dan evaluasi model pre-trained seperti *IndoBERT* atau *Indonesian-RoBERTa*, karena menyajikan data yang representatif dan relevan untuk memahami ekspresi emosi dalam bahasa Indonesia sehari-hari.

3.3. Operasional variabel

Operasional variabel adalah proses mendefinisikan variabel-variabel penelitian dengan cara yang spesifik sehingga dapat diukur dan dianalisis secara empiris. Definisi operasional ini mencakup penjelasan rinci tentang bagaimana variabel diukur, serta alat dan prosedur yang digunakan untuk pengukuran tersebut. Tujuannya adalah untuk memastikan bahwa variabel dapat diukur secara konsisten dan dapat diulang oleh peneliti lain dengan hasil yang serupa. Menurut (Sugiyono 2021), dalam bukunya "Metode Penelitian Kuantitatif, Kualitatif, dan R&D", operasional variabel adalah definisi yang diberikan kepada variabel-variabel sehingga menjadi variabel yang dapat diukur. Sugiyono menjelaskan bahwa operasionalisasi variabel harus mencakup indikator-indikator yang jelas, yang dapat digunakan untuk mengukur variabel tersebut secara akurat.

Tabel 3.3 Tabel operasional *variable*

Variabel	Definisi	Indikator	Skala Pengukuran	Sumber Data	Metode Pengumpulan Data
Sentimen	Emosi atau opini pengguna yang diekspresikan dalam teks (positif, negatif, netral).	- Positif - Negatif - Netral	Kategori	*Google Play Store*	*Web Scraping*
Ulasan Aplikasi	Komentar atau evaluasi yang diberikan oleh seseorang terkait suatu hal	- Jumlah ulasan - Panjang teks ulasan - Skor ulasan (rating)	Rasio	*Google Play Store*	*Web Scraping*

(**Sumber:** Penelitian 2025)

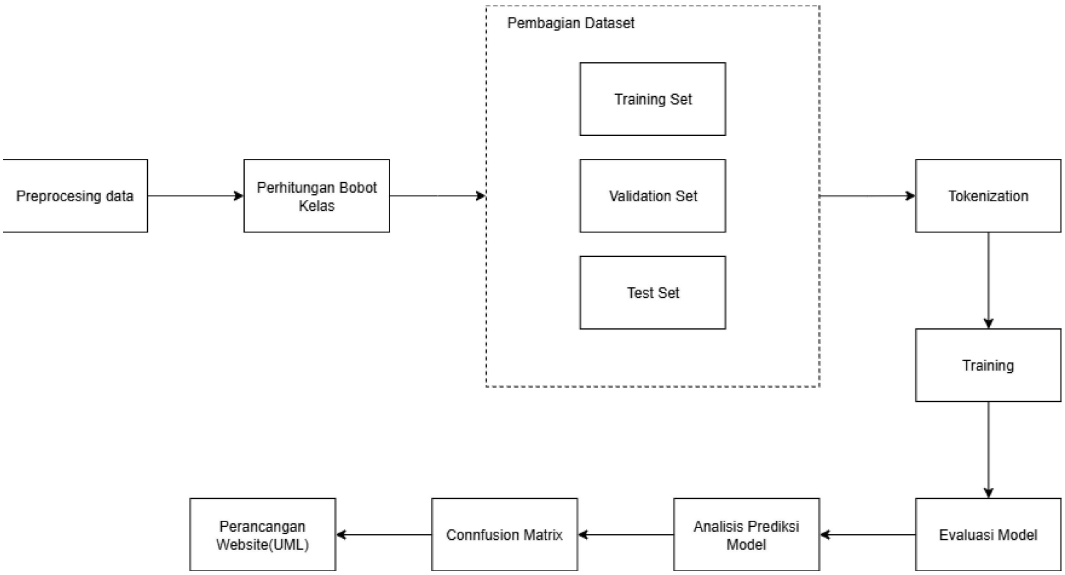
Penjelasan

1. Sentimen
    - Definisi: Sentimen mengacu pada emosi atau opini yang diekspresikan oleh pengguna dalam teks. Sentimen ini dapat berupa positif, negatif, atau netral.
    - Indikator: Kategori sentimen (positif, negatif, netral).
    - Sumber Data: *Google Play Store*
    - Metode Pengumpulan Data: web *scraping* digunakan untuk mengumpulkan teks dari ulasan aplikasi Instagram
2. Ulasan Aplikasi
    1. Definisi: Ulasan adalah komentar atau evaluasi yang diberikan oleh seseorang terkait suatu hal.
    2. Indikator: Jumlah ulasan, panjang teks ulasan, skor ulasan (rating).
    3. Sumber Data: *Google Play Store.*
    4. Metode Pengumpulan Data: Web *scraping* untuk mengumpulkan ulasan pengguna.

3.4. Metode Perancangan

Pada bab ini, akan diuraikan mengenai metode perancangan yang digunakan dalam penelitian. Metode perancangan merupakan kerangka kerja sistematis yang menjadi landasan bagi pelaksanaan penelitian, mulai dari tahap persiapan data hingga evaluasi akhir model.

Tujuan utama dari perancangan ini adalah untuk memastikan bahwa setiap langkah penelitian dilakukan secara terstruktur, valid, dan dapat direplikasi, sehingga tujuan penelitian untuk melakukan analisis sentimen pada ulasan aplikasi Instagram menggunakan model fine-tuned *Indonesian-RoBERTa* dapat tercapai secara efektif dan efisien. Metode perancangan dalam penelitian ini mencakup serangkaian tahapan yang saling berkaitan. Metode perancangan yang dilakukan akan mengacu pada kerangka kerja berikut ini.



Gambar 3.2 Metode Perancangan

(**Sumber:** Penelitian 2025)

3.4.1. *Preprocessing* data

Dataset yang digunakan pada penelitian ini adalah dataset *tweet* PPKM yang di ambil dari kaggle yang berjumlah 23.671 data, dengan pembagian untuk data sentimen yang berlabel 0 (possitif) sebanyak 1.958 data, data sentimen yang berlabel 1(netral) sebanyak 17.706 data dan data sentimen yang berlabel 2(negatif)

sebanyak 3.980 data. Sebelum data digunakan untuk *finetuning* model *Indonesian-RoBERTa*, penting dilakukan *processing* data untuk memastikan data benar benar bersih agar hasil *finetuning* dapat maksimal, proses *processing* data ini mencakup beberapa tahapan seperti menghapus tautan(URL), menghapus *User*name(@), mengubah hastag(#) menjadi teks biasa, menghapus symbol/tanda baca, menghapus spasi berlebih dan mengubah teks menjadi huruf kecil(lowercase).

Setelah data selesai di bersihkan, kemudian nama kolom *tweet* diubah menjadi *text* dan nama kolom sentimen diubah menjadi label, setelah itu baris duplikat berdasarkan kolom *text* dihapus, serta baris yang memiliki nilai kosong pada kolom *text* dan label juga di hapus.

3.4.2. Perhitungan bobot kelas (*class weights*)

Perhitungan bobot kelas setelah dibersihkan dan sebelum dilakukannya split atau pembagian data merupakan langkah yang penting dilakukan untuk menghindari ketidakseimbangan kelas dalam pembagian data dan proses pelatihan. Berdasarkan dataset yang digunakan, setelah dilakukannya tahap *processing* data jumlah kelas antara kelas positif, netral dan negatif sangat tidak seimbang dengan jumlah kelas netral sebanyak 15.579 data, negatif 3.878 data dan kelas positif hanya 1.885 data, hal tersebut dapat menimbulkan bias terhadap model pada proses pelatihan, yang akan menyebabkan model lebih cenderung akan memprediksi sentimen dengan kelas mayoritas (netral) di bandingankan positif dan negatif. Untuk menghindari hal tersebut, perhitungan dan penyeimbangan bobot kelas menjadi hal yang sangat penting untuk dilakukan, proses perhitungan bobot kelas ini akan dijelaskan dalam Langkah Langkah berikut ini:

$$w_i \frac{N}{K \cdot n_i}$$ **Rumus 3.1**

w_i =Bobot kelas ke-i

N = Jumlah sampel

n_i = Jumlah sampel kelas i

Dari rumus tersebut dapat dilakukan perhitungan pada setiap bobot kelas seperti berikut:

Tabel 3.4 Perhitungan Bobot Kelas

Kelas	Perhitugan	Bobot
0	$w_0 = \frac{21.342}{3 \cdot 1.885}$ $w_o = \frac{21.342}{5.665}$ $w_0 = 3{,}773$	3,773
1	$w_1 = \frac{21.342}{3 \cdot 15.579}$ $w_1 = \frac{21.342}{46.737}$ $w_1 = 0{,}457$	0,457
2	$w_2 = \frac{21.342}{3 \cdot 3.878}$ $w_2 = \frac{21.342}{11.634}$ $w_2 = 1{,}835$	1,835

(**Sumber:** Penelitian 2025)

3.4.3. Pembagian dataset

Setelah data berhasil di bersihakan dan penanganan bobot kelas yang tidak seimbang sudah dilakukan, langkah selanjutnya adalah pembagian dataset menjadi 3 bagian, yaitu data latih (*training* set), data validasi (*Validation Set*) dan data uji (*test set*). Dalam pembagaian data ini, data yang berjumlah 21.342 data dibagi dengan rasio 81/9/10 yang dimana 81% akan menjadi data latih, 9% data validasi dan 10% menjadi data uji. Berikut adalah penjelasan detail mengenai pembagia data ini:

1. Data latih

Ini adalah porsi terbesar dari dataset, mencakup 81% dari keseluruhan data. Data latih berfungsi sebagai "materi belajar" utama bagi model *Indonesian-RoBERTa*. Selama fase *fine-tuning*, model akan menganalisis teks dan label pada set data ini secara berulang-ulang untuk mempelajari pola, konteks, dan fitur linguistik yang membedakan sentimen positif, netral, dan negatif. Seluruh proses penyesuaian bobot internal model terjadi secara eksklusif pada data ini.

2. Data Validasi (*Validation Set*):

Sebesar 9% dari total data dialokasikan untuk set validasi, data ini berperan]sebagai "pengawas" selama proses pelatihan, pada setiap akhir epoch (satu siklus penuh pelatihan pada data latih), performa model akan diukur menggunakan data validasi. Hasil dari evaluasi ini digunakan untuk beberapa tujuan penting seperti memonitor *overfitting* dan menyetel hyperparameter.

3. Data Uji (*Test set*):

Sisa 10% dari data disimpan sebagai data uji, data uji sama sekali tidak tersentuh selama proses pelatihan dan validasi, data ini baru digunakan setelah model final telah terpilih. Tujuannya adalah untuk memberikan penilaian yang paling jujur dan tidak bias terhadap performa model di dunia nyata. Hasil evaluasi pada data uji inilah yang akan dilaporkan sebagai performa akhir dari model yang dikembangkan.

3.4.4. *Tokenization*

Tokenisasi merupakan proses yang dilakukan setelah penentuan bobot kelas dan pembagaian dataset menjadi dataset train, val dan test yang memiliki tujuan untuk memecah teks seperti kalimat menjadi bagian yang lebih kecil dan bagian tersebut disebut sebagai token, ini seperti memotong sayuran untuk dimakan karena manusia tidak bisa memakan sayuran utuh sekaligus, dalam tokenisasi, token biasanya akan berupa kata, karakter, sub-kata atau tanda baca. Proses Tokenisasi menjadi proses yang cukup penting sebelum melakukan pelatihan kepada model, karena pada dasarnya komputer tidak mengerti dengan huruf melainkan hanya mengerti angka, dengan melakukan tokenisasi teks yang sudah di pecah menjadi token, dapat diubah menjadi angka yang dimengerti oleh computer, proses mengubah token menjadi angka disebut sebagai *vectorization* atau *Embedding.* berikut ini merupakan proses tokenisasi dalam *finetuning* yang penulis lakukan:

1. Mengambil teks mentah.

Teks mentah yang sudah melewati pra-pemrosesan data akan diambil untuk pecah atau ditokenisasi, contoh teks mentah yang tokenisasi seperti berikut:

Original : "Aplikasi ini sangat berguna dan saya suka sekali!"

Setelah *preprocessing* : "aplikasi ini sangat berguna dan saya suka sekali"

2. Memecah teks menjadi token

Prsoses tokenisasi dilakukan dengan memcah kata mentah menjadi token atau menjadi bagian bagian yang lebih kecil, contoh nya pada teks "aplikasi ini sangat berguna dan saya suka sekali" akan dipecah menjadi bagian bagian kecil seperti ["aplikasi", "ini", "sangat", "berguna", "dan", "saya", "suka", "sekali"]. Ada berbagai jenis tokenizer yang untuk melakukan tokenisasi seperti WordPiece, atau BPE, model *RoBERTa* menggunakan teknik BPE dalam proses tokenisasi, begitu juga dengan model *Indonesian-RoBERTa* yang merupakan turunan langsung dari model *RoBERTa*.

3.4.5. Proses *Training* (*Fine-tuning Process*)

Proses *training* merupakan tahap di mana model machine learning belajar dari data yang telah disiapkan, data yang digunakan merupakan data yang telah melewati tahap *processing* data, pembagian dataset dan tookenisasi. Dalam penelitian ini, proses yang dilakukan bukanlah melatih sebuah model dari nol, melainkan melakukan *fine-tuning*, yang dimana model pre-trained atau model yang sudah dilatih sebelumnya dilatih kembali guna meningkatkan peforma model dan diadaptasi untuk menyelesaikan tugas yang lebih spesifik. Pendekatan ini dipilih karena lebih efisien secara komputasi dan terbukti sangat efektif untuk berbagai tugas NLP. Proses *fine-tuning* pada penelitian ini dirancang melalui beberapa langkah sebagai berikut:

1. Inisialisasi Model *Pre-trained*

Langkah pertama adalah memuat model dasar yang akan digunakan, yaitu *w11wo/Indonesian-RoBERTa-base-sentiment-classifier*. Model ini merupakan varian dari *RoBERTa* yang sudah memiliki pemahaman mendalam tentang tata bahasa, sintaksis, dan semantik Bahasa Indonesia dari pelatihan sebelumnya. Saat memuat model ini, dilakukan sebuah modifikasi *Classification Head* atau Adaptasi Kepala Klasifikasi yang dimana Kepala atau lapisan terakhir dari model *pre-trained* dikonfigurasi ulang dengan parameter *num_labels*=3. Perintah ini menginstruksikan model untuk mengganti lapisan *output* standarnya dengan lapisan baru yang dirancang khusus untuk tugas klasifikasi tiga kelas (0: positif, 1: netral, 2: negatif), sesuai dengan label pada dataset penelitian.

2. Implementasi *Weighted Loss* melalui *Custom Trainer*

Seperti yang sudah diuraikan pada sub bab sebelumnya bahwa dataset yang digunakan untuk proses *finetuning* dalam penelitian memliki ketidakseimbangan data dengan jumlah perbandingan rasio yang cukup besar, maka dilakukan pendekatan *class weighting* untuk mengatasi permasalahan tersebut dengan membuat kelas *WeightedTrainer* yang merupakan turunan dari kelas *trainer* milik *Hugging Face*. Tujuannya adalah untuk menyesuaikan perhitungan *loss* agar lebih sensitif terhadap distribusi kelas yang tidak seimbang.

Di dalam kelas tersebut, kelas *WeightedTrainer* akan menimpa fungsi *compute_loss* dan menginisialisasi fungsi kerugian *torch.nn.CrossEntropyLoss* dengan parameter *weight* yang diisi oleh tensor *class_weights*. Bobot ini sebelumnya telah dihitung berdasarkan proporsi masing-masing kelas dalam data,

dengan cara ini kesalahan prediksi pada kelas minoritas akan mendapatkan penalti yang lebih besar dibandingkan kelas mayoritas, sehingga model diharapkan mampu belajar lebih adil dan tidak bias terhadap kelas yang lebih dominan.

3. Eksekusi Proses Pelatihan

Terakhir pada pada tahap eksekusi proses pelatihan model mulai belajar dari data yang sudah disiapkan sebelumnya, pada tahap ini, model secara bertahap menyesuaikan bobot-bobot internalnya dengan mempelajari pola-pola yang ada dalam data latih. Proses ini berlangsung secara berulang dalam beberapa siklus, di mana pada setiap siklusnya model mencoba memprediksi label dari data yang diberikan, kemudian mengevaluasi seberapa jauh prediksinya dari label yang benar, dan menggunakan informasi tersebut untuk melakukan perbaikan. Selama *training*, model juga dievaluasi secara berkala menggunakan data validasi untuk memantau performa dan mencegah terjadinya *overfitting*. Inti dari tahap ini adalah proses pembelajaran yang memungkinkan model mengembangkan pemahaman terhadap data sehingga mampu melakukan prediksi yang lebih akurat pada data baru.

3.4.6. Evaluasi Model

Setelah proses pelatihan (*finetuning*) model selesai dilakukan, langkah berikutnya adalah melakukan evaluasi terhadap performa model dalam mengklasifikasikan sentimen. Evaluasi ini dilakukan dengan menggunakan data uji (*test set*) yang telah dipisahkan sebelumnya dengan tujuan untuk mengetahui seberapa baik model dapat mengenali dan mengklasifikasikan data yang belum pernah dilihat sebelumnya. Evaluasi dilakukan menggunakan metrik-metrik

klasifikasi yang umum digunakan dalam permasalahan klasifikasi multi-kelas, yaitu *precision*, *recall*, dan *F1-score*, yang dihitung untuk masing-masing kelas (per-*class*), serta dalam bentuk agregasi *macro, micro*, dan *Weighted Average*.

1. Perhitungan *Precision*, *Recall*, dan *F1-score* Per-Kelas

Didalam penelitian ini, klasifikasi dilakukan dengan tiga label sentimen, yaitu:

Label 0: *Positive*

Label *1: Neutral*

Label 2: *Negative*

Berikut adalah rumus yang digunakan:

$$Precision = \frac{TP}{TP + FP} \quad \textbf{Rumus 3.2}$$

$$Recall = \frac{TP}{TP + FN} \quad \textbf{Rumus 3.3}$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad \textbf{Rumus 3.4}$$

Berikut adalah notasi yang digunakan:

TP_j: *True Positive* untuk kelas *j*

FP_j: *False Positive* untuk kelas *j*

FN_j: *False negative* untuk kelas *j*

C_{ij}: Elemen pada *confusion matrix*, yaitu jumlah data dengan label sebenarnya i yang diprediksi sebagai *j*

a) Kelas 0 (positif):

$$TP_0 = C_{00} = 148$$

$$FP_0 = C_{10} + C_{20} = 35 + 19 = 54$$

$$FN_0 = C_{01} + C_{02} = 37 + 4 = 41$$

$$Precision_0 = \frac{TP_0}{TP_0+FP_0} = \frac{148}{148+54} = \frac{148}{202} \approx 0{,}7327$$

$$Recall_0 = \frac{TP_0}{TP_0+FN_0} = \frac{148}{148+41} = \frac{148}{189} \approx 0{,}7831$$

$$F1 - Score_0 = \frac{2.Precision_0 \cdot Recall_0}{Precision_0+Recall_0} = \frac{2 \cdot 0.7327 \cdot 0.783}{0.7327+0.783} \approx 0{,}7573$$

b) Kelas 1 (netral):

$$TP_1 = C_{11} = 1.472$$

$$FP_1 = C_{01} + C_{21} = 37 + 57 = 94$$

$$FN_1 = C_{10} + C_{12} = 35 + 51 = 86$$

$$Precision_1 = \frac{1472}{1472+94} = \frac{1472}{1566} \approx 0{,}9400$$

$$Recall_1 = \frac{1472}{1472+86} = \frac{1472}{1558} \approx 0{,}9448$$

$$F1 - Score_1 = \frac{2.Precision_1 \cdot Recall_1}{Precision_1+Recall_1} = \frac{2 \cdot 0{,}9400 \cdot 0{,}9448}{0{,}9400+0{,}9448} \approx 0{,}9424$$

c) Kelas 2 (negatif):

$$TP_2 = C_{22} = 312$$

$$FP_2 = C_{02} + C_{12} = 4 + 51 = 55$$

$$FN_2 = C_{20} + C_{21} = 19 + 57 = 76$$

$$Precision_2 = \frac{312}{312+55} = \frac{312}{367} \approx 0{,}8501$$

$$Recall_2 = \frac{312}{312+76} = \frac{312}{388} \approx 0{,}8041$$

$$F1 - Score_2 = \frac{2.Precision_2 \cdot Recall_2}{Precision_2+Recall_2} = \frac{2 \cdot 0{,}8501 \cdot 0{,}8041}{0{,}8501+0{,}8041} \approx 0{,}8264$$

2. Evaluasi Agregat: *Macro, Micro, dan Weighted Average*

Untuk mendapatkan gambaran performa model secara keseluruhan, dilakukan agregasi menggunakan tiga pendekatan, yaitu *Macro Averaging*, *Micro Averaging*, dan *Weighted Averaging*. Rumusa yang digunakan untuk menghitung evaluasi agregat ini adalah:

$$Precision_{macro} = \frac{1}{K}\sum_{i=1}^{K} Precision_i$$ **Rumus 3.5**

$$Recall_{macro} = \frac{1}{K}\sum_{i=1}^{K} Recall_i$$ **Rumus 3.6**

$$F1 - Score_{macro} = \frac{1}{K}\sum_{i=1}^{K} F1_i$$ **Rumus 3.7**

a) *Macro Averaging*

Pendekatan ini menghitung rata-rata metrik dari tiap kelas secara aritmatika tanpa mempertimbangkan proporsi jumlah data per kelas.

$$Precision_{macro} = \frac{P_0+P_1+P_2}{K} = \frac{0{,}7327+0{,}9400+0{,}8501}{3} \approx 0{,}8409$$

$$Recall_{macro} = \frac{R_0+R_1+R_2}{K} = \frac{0{,}7831+0{,}9448+0{,}8041}{3} \approx 0{,}8440$$

$$F1 - score_{macro} = \frac{F_0+F_1+F_2}{K} = \frac{0{,}7573+0{,}9424+0{,}8264}{3} \approx 0{,}8420$$

b) *Micro Averaging*

Pendekatan ini menjumlahkan semua *true positive*, *false positive*, dan *false negative* dari seluruh kelas terlebih dahulu, baru kemudian dihitung metriknya.

$$TP_{total} = 1932$$

$$FP_{total} = 203$$

$$FN_{total} = 203$$

$$Precision_{micro} = \frac{TP_{total}}{TP_{total}+FP_{total}} = \frac{1932}{2135} \approx 0{,}9049$$

$$Recall_{micro} = \frac{TP_{total}}{TP_{total}+FN_{total}} = \frac{1932}{2135} \approx 0{,}9049$$

$$F1 - Score_{micro} = \frac{2 \cdot Precision_{micro} \cdot Recall_{micro}}{Precision_{micro}+Recall_{micro}} = \frac{1.932}{2.135} \approx 0{,}9049$$

c) *Weighted Averaging*

Pendekatan ini menghitung rata-rata metrik dengan mempertimbangkan jumlah data (support) dari tiap kelas. Support untuk setiap kelas:

$$w_0 = \frac{189}{2135} \approx 0{,}0885$$

$$w_1 = \frac{1558}{2135} \approx 0{,}7296$$

$$w_2 = \frac{388}{2135} \approx 0{,}0885$$

$$Precision_{weighted} = w_0 \cdot P_0 + w_1 \cdot P_1 + w_2 \cdot P_2 \approx 0{,}9048$$

$$Recall_{weighted} = w_0 \cdot R_0 + w_1 \cdot R_1 + w_2 \cdot R_2 \approx 0{,}9002$$

$$F1 - Score_{weighted} = w_0 \cdot F1_0 + w_1 \cdot F1_1 + w_2 \cdot F1_2 \approx 0{,}9039$$

3.4.7. Analisis Prediksi Model

Setelah dilakukan evaluasi terhadap performa model menggunakan metrik-metrik klasifikasi seperti *precision*, *recall*, dan *F1-score*, langkah selanjutnya adalah melakukan analisis lebih dalam terhadap bagaimana model menghasilkan prediksi. Proses prediksi dalam model klasifikasi seperti *RoBERTa* tidak langsung menghasilkan label, melainkan melalui beberapa tahap transformasi, yaitu dari *logits* mentah ke probabilitas, dan kemudian ke prediksi akhir.

Ketika model *Indonesian-RoBERTa* menerima sebuah *Input* berupa teks, model tersebut akan menghasilkan sebuah vektor *real-valued* yang berisi 3 angka sesuai dengan jumlah kelas dalam dataset. Vektor ini disebut sebagai *logits*, dan dilambangkan sebagai:

$$z^{(1)} = [z_0^{(i)}, z_1^{(i)}, z_2^{(i)}]$$

Di mana:

$z_0^{(i)}$ = skor untuk kelas 0 (positif)

$z_1^{(i)}$ = skor untuk kelas 1 (netral)

$z_2^{(i)}$ = skor untuk kelas 2 (negatif)

Logits ini belum bisa diartikan sebagai probabilitas karena nilainya bisa negatif, tidak terbatas antara 0 dan 1 dan tidak menjamin menjumlahkan hingga 1. Untuk mengubah *logits* menjadi probabilitas yang dapat di interpretasikan, digunakan fungsi *softmax*:

$$P_j^{(i)} = \frac{e^{z_j^{(i)}}}{\sum_{k=0}^{2} e^{z_j^{(i)}}}$$

Rumus 3.8

$P_j^{(i)}$: probabilitas prediksi terhadap kelas j untuk data ke-i

$e^{z_j^{(i)}}$: eksponensial dari logit ke-j

Contoh teks yang diprediksi benar oleh model sebagai kalimat positif "Aplikasi ini sangat berguna dan saya suka sekali!" Kalimat tersebut akan menghasilkan *logits* $z^{(i)} = [6{,}3, -3{,}0, -3{,}1]$. Interpretasi dari angka yang terdapat dalam *logits* tersebut adalah:

$z_0 = 6{,}3$: Model memberikan skor tinggi untuk positif,

$z_1 = -3{,}0$: Model memberikan skor rendah untuk netral,

$z_2 = -3{,}1$: Model memberikan skor rendah untuk negatif

Logits yang sudah diperoleh tersebut akan dilakukan transformasi *softmax* untuk mengubah skor *logits* menjadi probabilitas, langkah langkah perhitungan *softmax* untuk kalimat tersebut adalah sebagai berikut:

$$e^{z_0^{(i)}} = e^{6{,}3} = 544{,}5719$$

$$e^{z_1^{(i)}} = e^{-3{,}0} = 0{,}0497$$

$$e^{z_0^{(i)}} = e^{-3{,}1} = 0{,}0450$$

$$Z = \sum_{j=0}^{2} e^{z_j^{(i)}} = 544{,}6666$$

$$P_0^{(i)} = \frac{544{,}5719}{544{,}6666} = 0{,}9998$$

$$P_1^{(i)} = \frac{0{,}0497}{544{,}6666} = 9{,}1248 \times 10^{-5}$$

$$P_2^{(i)} = \frac{0{,}0450}{544{,}6666} = 8{,}2619 \times 10^{-5}$$

Setelah model melakukan transformasi *logits* ke dalam bentuk probabilitas menggunakan fungsi *softmax*, langkah selanjutnya adalah menentukan kelas akhir yang akan menjadi *output* prediksi model. Keputusan ini ditentukan berdasarkan aturan *Maximum A Posteriori* (MAP), dimana model akan memilih kelas dengan probabilitas tertinggi sebagai prediksi akhir, dalam proses *finetuning* yang sudah dilakukan berarti:

$P_0^{(i)} = 0{,}9998 \leftarrow MAX$

$P_1^{(i)} = 0{,}000091$

$P_1^{(i)} = 0{,}000082$

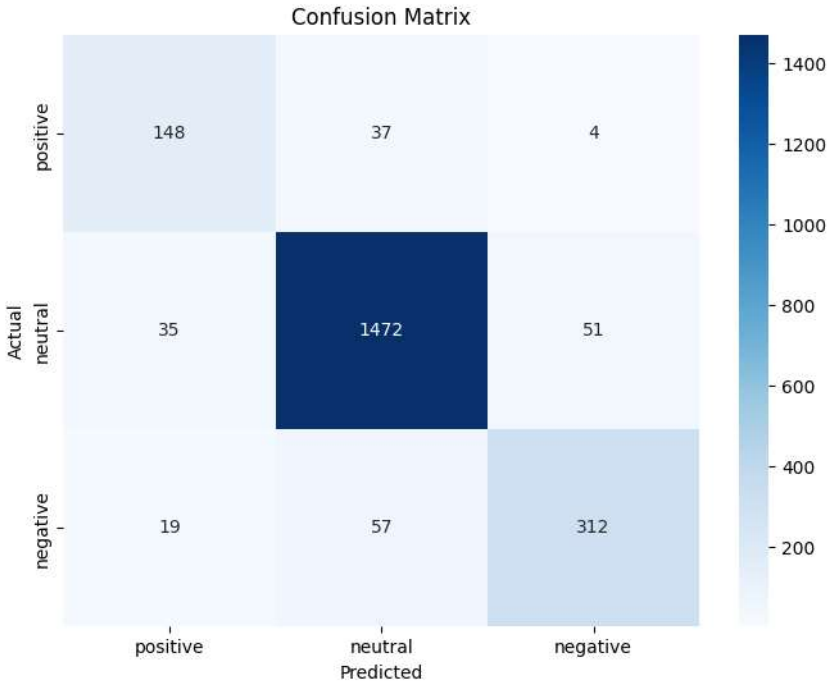
Prediksi = $argmax\{0{,}9998, 0{,}000091, 0{,}000082\} = 0$ (Positif)

Confidence = $0{,}9998$ (99.98%)

Transformasi dari *logits* ke probabilitas dan pemilihan kelas melalui pendekatan *Maximum A Posteriori* (MAP) merepresentasikan proses inferensi model yang logis, terukur, dan berbasis pada distribusi keyakinan terhadap masing-masing kelas.

3.4.8. *Confusion matrix*

Confusion matrix menjadi alat dalam klasifikasi, yang digunakan untuk memahami tipe kesalahan apa yang dilakukan oleh model, bukan hanya seberapa sering model salah atau benar secara keseluruhan. Pada *confusion matrix* terdapat informasi terkait berapa banyak prediksi benar dan kelas mana yang paling membingungkan model, informasi ini penting untuk melakukan perbaikan model.



Gambar 3.3 *Confusion matrix*

(**Sumber:** Penelitian 2025)

Pada *finetuning* yang sudah dilakukan terdapat tiga kelas (positif, netral, negatif) maka secara matematis, *confusion matrix* untuk *finetuning* ini didefinisikan sebagai matriks $C \in \mathbb{N}^{3\times3}$, dengan elemen:

$$C_{ij} = \sum_{k=1}^{n} 1\,(y^{(k)} = i \wedge \hat{y}^{(k)} = j)$$

Di mana:

C_{ij}: jumlah data yang sebenarnya kelas i, namun diprediksi sebagai kelas j

$y^{(k)}$: label aktual dari sampel ke-k

$\hat{y}^{(k)}$: label prediksi dari sampel ke-k

$1(\cdot)$: fungsi indikator, bernilai 1 jika kondisi benar, dan 0 jika salah

n: jumlah total data *test* ($n = 2.135$)

Dari hasil evaluasi model, diperoleh *confusion matrix* sebagai berikut:

$$C = \begin{bmatrix} 148 & 37 & 4 \\ 35 & 1472 & 51 \\ 19 & 57 & 312 \end{bmatrix}$$

Total *Actual Positive*: $148 + 37 + 4 = 189$

Total *Actual Neutral*: $35 + 1472 + 51 = 1.558$

Total *Actual Negative*: $19 + 57 + 312 = 388$

Total: $189 + 1558 + 388 = 2.135$

Berdasarkan Analisi prediksi model yang sudah dilakukan teks "Aplikasi ini sangat berguna dan saya suka sekali!" yang memiliki hasil prediksi dengan skor confidence sebesar 0,9998% untuk kelas positif, teks tersebut masuk kedalam elemen C_{00} yang menjadi bagian dari 148 prediksi benar pada kelas positif.

3.4.9. Perancangan Website (UML)

UML (*Unified Modelling Language*) dapat didefinisikan sebagai suatu bahasa standar visualisasi, perancangan, dan pendokumentasian sistem, atau dikenal juga sebagai bahasa standar penulisan blueprint sebuah software (Nurshadrina and Voutama 2022). Diagarm UML yang digunakan untuk menganalisis aplikasi yang dirancang adalah sebagai berikut:

1. *Use case diagram*

Use case diagram menggambarkan interaksi antara pengguna dan sistem dalam *dashboard* analisis sentimen, mencakup aktivitas seperti *Login*, *register*s, hingga melakukan analisis sentiment dari ulasan yang di ambil dari *Google Play Store*. Diagram ini menunjukkan bagaimana pengguna berperan langsung dalam setiap proses utama yang disediakan oleh sistem.

Gambar 3.4 Use Case Diagram

(Sumber: Penelitian 2025)

Skenario yang berjalan pada rancangan use case di atas adalah sebagai berikut:

Skenario *usecase*

1) Nama *Usecase* : *Login*

   Aktor : *User*

   Tujuan : Masuk ke web *dashboard*

User	Sistem Web
Masuk ke web *dashboard* dengan memasukkan email dan password	
	Memverifikasi proses *Login User*

2) Nama *Usecase* : *Register*

Aktor : *User*

Tujuan : Membuat akun untuk bisa masuk ke web *dashboard*

User	Sistem Web
Melakukan *register* akun dengan memasukkan *User*name, Email, Password dan Ulangi Password	
	Memverifikasi dan menyimpan data *register* pengguna

3) Nama *Usecase* : *Homepage*

Aktor : *User*

Tujuan : Masuk ke *homepage* web *dashboard*

User	Sistem Web
Melakukan proses *Login* dan masuk ke *homepage* web *dashboard*	
	Memverifikasi proses *Login User* dan mengizinkan *User* masuk ke *homepage* web *dashboard*

4) Nama *Usecase* : Klasifikasi Teks Positif, netral atau negatif

Aktor : *User*

Tujuan : Untuk mengetahui hasil sentiment ulasan pengguna Instagram yang di ambil dari *Google Play Store*

User	Sistem NLP
Menyalin *Package name* seperti com.instagram.android dari URL *Playstore* dan ditempel di kolom '*Package name*' pada *dashboard*.	
	Sistem mengambil ulasan dari *Google Play Store*, lalu menganalisis sentimennya menggunakan model *Indonesian-RoBERTa*

5) Nama *Usecase* : Hasil Klasifikasi

Aktor : *User*

Tujuan : Menampilkan hasil klasifikasi sentiment

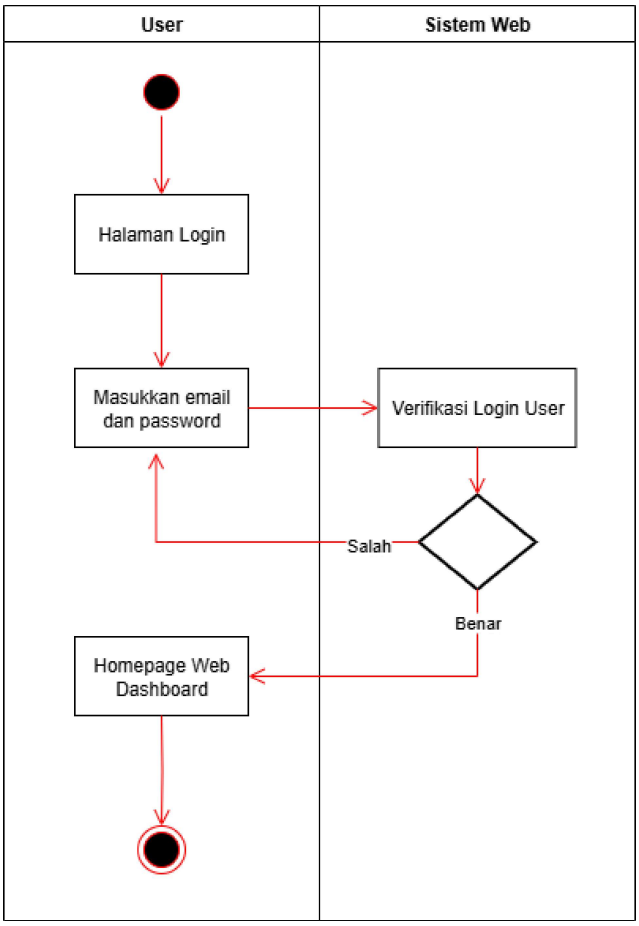
User	Sistem NLP	Sistem Web
Menunggu Sistem NLP selesai melakukan klasifikasi untuk setiap ulasan		

	Melakukan proses klasifikasi untuk setiap ulasan yang di ambil dari *Google Play Store*	
		Menampilkan hasil visualisasi dari hasil klasifikasi

2. *Activity diagram*

Activity diagram menggambarkan alur proses yang terjadi saat *User* melakukan *Login*, *register*, memasukkan *Package name* ke dalam *dashboard*, dimulai dari *Input User*, pengambilan data ulasan dari *Google Play Store*, pemrosesan data oleh model *Indonesian-RoBERTa*, klasifikasi sentimen, hingga penyajian hasil analisis dalam bentuk visual di *dashboard*.

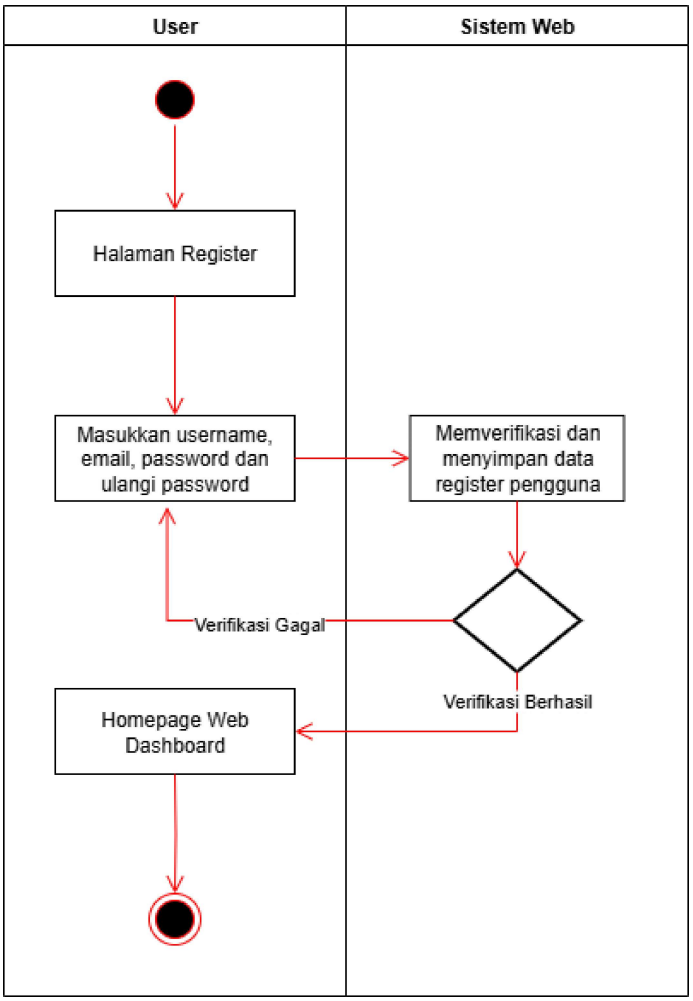
1) *Activity* diagram *Login*



Gambar 3.5 *Activity* diagram *Login*

Activity diagram ini menunjukkan alur *Login* pengguna. Proses dimulai dari halaman *Login*, dilanjutkan dengan pengisian email dan password. Sistem kemudian memverifikasi email dan password yang dimasukkan. Jika verifikasi gagal, pengguna diminta mengisi ulang data. Jika berhasil, pengguna diarahkan ke *homepage dashboard* dan proses selesai.

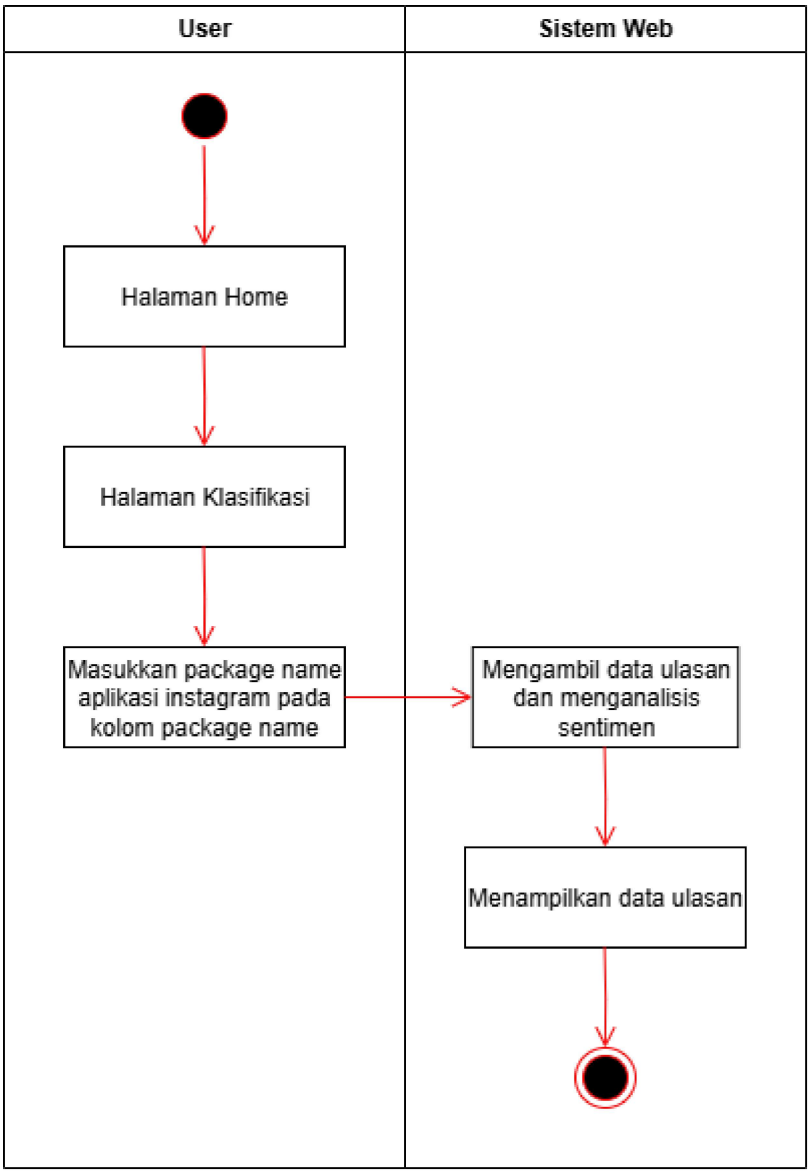
2) *Activity* diagram *register*



Gambar 3.6 *Activity* diagram *register*

Activity diagram ini menunjukkan alur registrasi pengguna pada sistem. Proses dimulai dari halaman *register*, dilanjutkan dengan pengisian *User*name, email, password, dan konfirmasi password. Sistem kemudian memverifikasi dan menyimpan data yang dimasukkan. Jika verifikasi gagal, pengguna diminta mengisi ulang data. Jika berhasil, pengguna diarahkan ke *homepage dashboard* dan proses selesai.

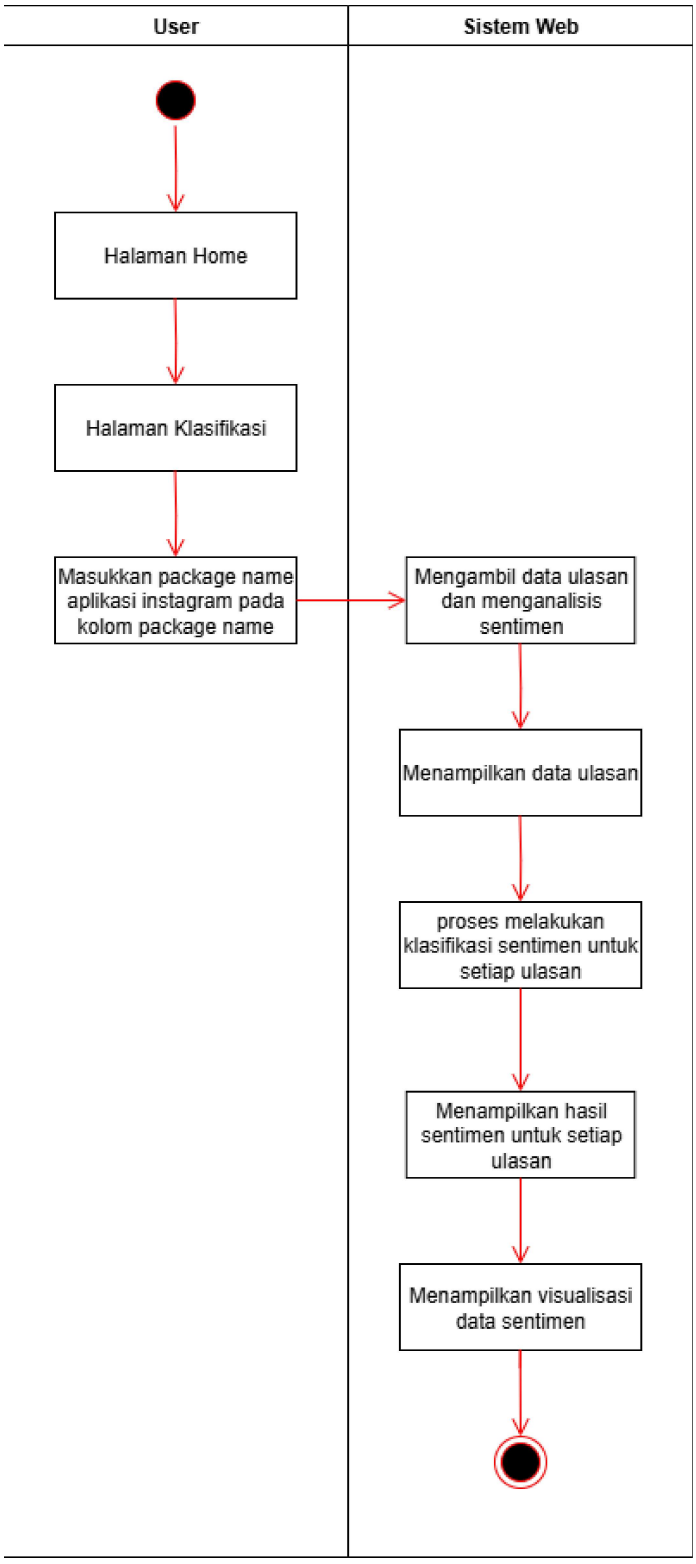
3) *Activity* diagram klasifikasi teks sentiment



Gambar 3.7 *Activity* diagram klasifikasi teks sentiment

Activity diagram ini menggambarkan alur proses klasifikasi sentimen dimulai dari pengguna mengakses halaman klasifikasi, lalu memasukkan *Package name* aplikasi, seperti com.instagram.android, sistem kemudian menjalankan proses pengambilan data ulasan dari *Google Play Store* dan menampilkannya.

4) *Activity* diagram hasil sentiment dan visualisasi data



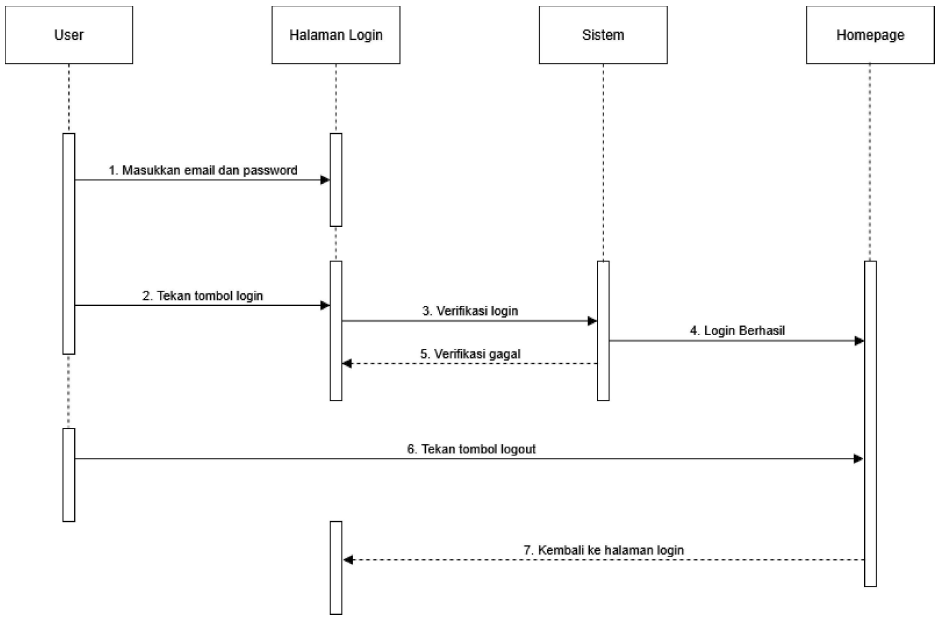
Gambar 3.8 *Activity* diagram hasil sentiment dan visualisasi data

Activity diagram ini menggambarkan alur dimana *user* menunggu sistem melakukan proses klasifikasi untuk setiap ulasan, lalu menampilkan label sentimen yang dihasilkan. Setelah itu, sistem menghitung dan menampilkan visualisasi data sentimen ke dalam *dashboard* sebagai *output* akhir. Diagram ini menunjukkan interaksi pasif dari pengguna dan aktivitas otomatis sistem dalam menghasilkan serta menyajikan informasi sentimen secara visual.

3. *Sequence Diagram*

Sequence Diagram digunakan untuk menggambarkan alur interaksi antar objek dalam sistem, diagram ini menunjukkan bagaimana objek dalam sistem saling berinteraksi, Dalam sistem analisis sentimen yang dibangun, *Sequence Diagram* digunakan untuk memvisualisasikan proses mulai dari *Input* data oleh pengguna, pemrosesan oleh sistem, hingga pengembalian hasil analisis sentimen.

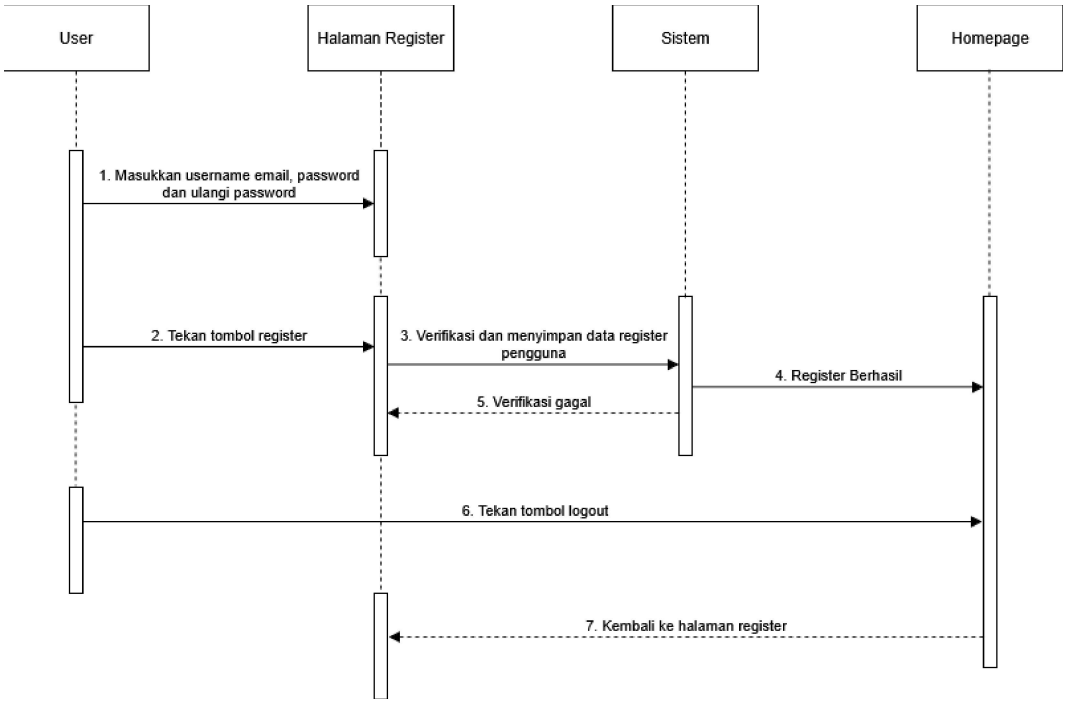
1) *Sequence Diagram Login* dan *Logout*



Gambar 3.9 *Sequence Diagram Login* dan *Logout*

Sequence Diagram pada proses *login* ini menggambarkan alur interaksi antara pengguna (*User*), halaman *login*, sistem, dan *homepage*. Proses dimulai ketika pengguna memasukkan email dan password, lalu menekan tombol *login*. Halaman *login* kemudian meneruskan permintaan ke sistem untuk melakukan verifikasi. Jika *login* berhasil, sistem mengarahkan pengguna ke *homepage*. Sebaliknya, jika verifikasi gagal, sistem memberikan respons kegagalan. Setelah *login*, pengguna dapat menekan tombol *logout* untuk keluar, dan sistem akan mengarahkan kembali ke halaman *login*. Diagram ini menjelaskan urutan proses *Login* hingga *logout* secara terstruktur dan kronologis.

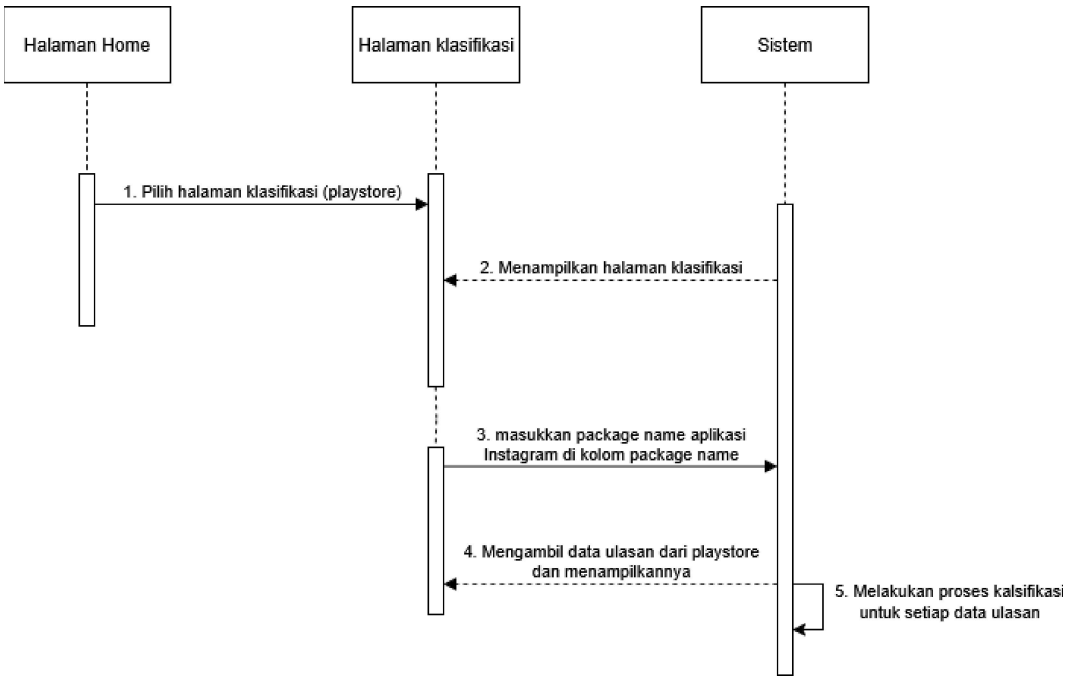
2) *Sequence Diagram Register* dan *Logout*



Gambar 3.10 *Sequence Diagram Register* dan *Login*

Sequence Diagram ini menggambarkan alur interaksi antar komponen dalam proses pendaftaran pengguna baru. Proses diawali ketika pengguna mengisi data berupa *User*name, email, password, dan konfirmasi password. Setelah menekan tombol *register*, halaman *register* akan mengirimkan data tersebut ke sistem untuk diverifikasi dan disimpan. Jika data valid dan proses penyimpanan berhasil, pengguna akan diarahkan ke halaman *homepage* dengan status *register* berhasil. Namun, jika terjadi kesalahan (seperti email sudah terdaftar), sistem akan menampilkan pesan bahwa registrasi gagal. Setelah berhasil masuk, pengguna juga dapat memilih untuk keluar dengan menekan tombol *logout* dan sistem akan mengarahkan kembali ke halaman *login*.

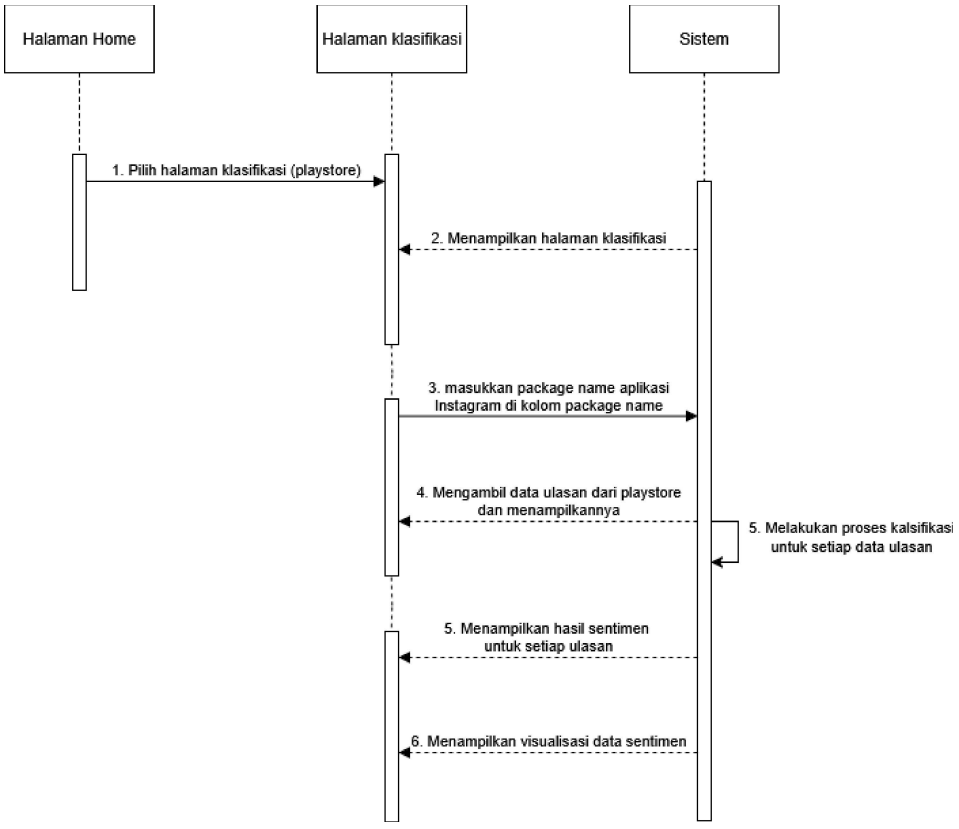
3) *Sequence Diagram* klasifikasi teks sentiment



Gambar 3.11 *Sequence Diagram* klasifikasi teks sentimen

Sequence Diagram ini menggambarkan alur proses klasifikasi ulasan pengguna instagram dari *Play Store*. Pengguna memulai dengan memilih halaman klasifikasi, kemudian sistem menampilkan halaman klasifikasi kepada pengguna. Selanjutnya, pengguna memasukkan *package name* aplikasi instagram ke dalam kolom yang tersedia, dan sistem mengambil data ulasan dari *playstore* berdasarkan *package name* tersebut. Setelah data ulasan berhasil diambil, sistem melakukan proses klasifikasi sentimen untuk menganalisis setiap data ulasan yang telah dikumpulkan. Proses ini memungkinkan sistem untuk mengkategorikan ulasan-ulasan aplikasi instagram berdasarkan sentimen positif, negatif, atau netral secara otomatis.

4) *Sequence Diagram* Hasil Klasifikasi

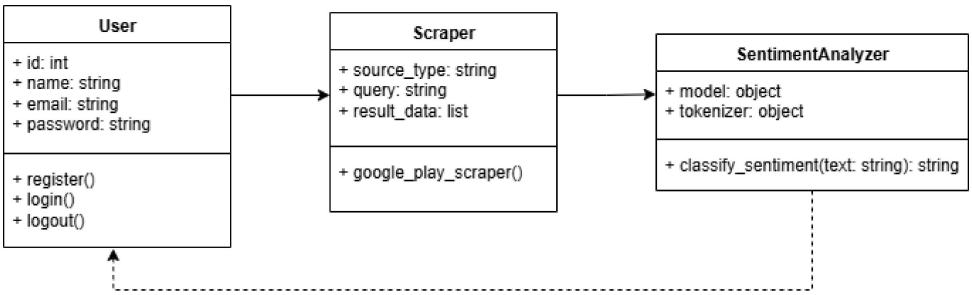


Gambar 3.12 *Sequence Diagram* Hasil Klasifikasi

Sequence Diagram ini menggambarkan alur proses pengambilan data, proses klasifikasi hingga menampilkan hasil klasifikasi sentimen dan visualisasi data. Pengguna memulai dengan memilih halaman klasifikasi, kemudian sistem menampilkan halaman klasifikasi kepada pengguna. Pengguna selanjutnya memasukkan *package name* aplikasi Instagram ke dalam kolom yang disediakan, dan sistem mengambil serta menampilkan data ulasan dari *playstore* berdasarkan *Package name* tersebut. Setelah data ulasan berhasil dikumpulkan, sistem melakukan proses klasifikasi untuk menganalisis sentimen setiap ulasan, kemudian menampilkan hasil sentimen untuk setiap data ulasan yang telah diproses. Terakhir, sistem menyajikan visualisasi data sentimen kepada pengguna, sehingga pengguna dapat melihat distribusi dan pola sentimen dari ulasan-ulasan aplikasi Instagram dalam bentuk grafik atau *chart* yang mudah dipahami

4. *Class Diagram*

Class diagram digunakan untuk memodelkan struktur statis sistem berbasis objek, diagram ini menggambarkan kelas-kelas yang ada dalam sistem beserta atribut, metode, dan relasi antar kelas. Pada sistem *dashboard* analisis sentimen yang dibangun, *class diagram* berfungsi untuk memberikan gambaran konseptual mengenai bagaimana setiap komponen sistem berinteraksi dan saling terkait dalam menjalankan fungsinya, mulai dari proses pengambilan data, analisis sentimen, hingga pengelolaan pengguna.



Gambar 3.13 *Class Diagram* Penelitian

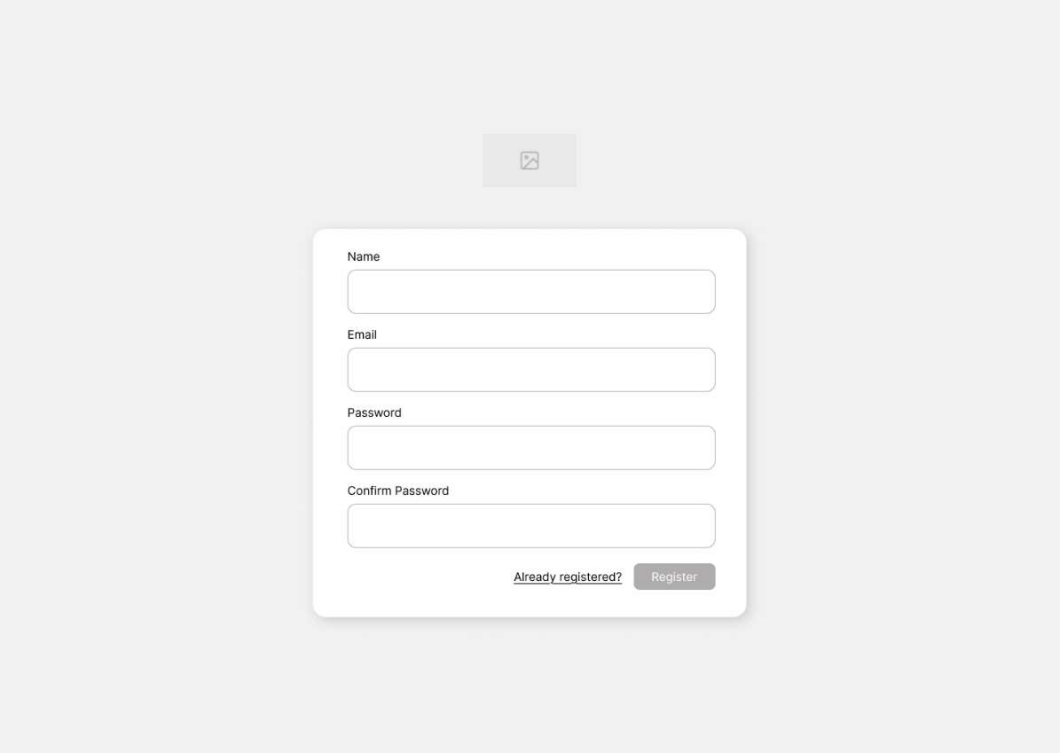
Class Diagram ini menggambarkan struktur sistem klasifikasi sentimen dengan tiga komponen utama yang saling berinteraksi. Kelas *User* berfungsi sebagai entitas pengguna dengan atribut identitas (*id, name, email, password*) dan *method* untuk autentikasi (*register*, *Login*, *logout*), yang kemudian menggunakan kelas *scraper* untuk mengambil data ulasan dari sumber tertentu melalui atribut konfigurasi (*source_type, query, result_data*) dan *method google_play_Scraper()*. Data hasil *scraping* selanjutnya diteruskan ke kelas SentimentAnalyzer yang memiliki komponen model *machine learning* (model, *tokenizer*) untuk memproses klasifikasi sentimen melalui *method classify_sentiment(),* dan hasil analisis dikembalikan kepada *user* melalui relasi *dependency*. Alur ini mencerminkan proses *end-to-end* dari *Input* pengguna hingga menghasilkan klasifikasi sentimen yang dapat divisualisasikan dalam *dashboard* web.

3.5. Desain Antarmuka

Desain antarmuka website dikembangkan dengan tujuan memberikan pengalaman pengguna yang intuitif, responsif, dan efisien dalam mengakses fitur analisis sentimen. Setiap elemen pada tampilan dirancang untuk memudahkan navigasi, memperjelas alur penggunaan, serta menyajikan informasi hasil

klasifikasi dan visualisasi data secara jelas dan informatif. Pendekatan desain difokuskan pada kesederhanaan tampilan, konsistensi elemen visual, serta keterbacaan konten agar pengguna dapat berinteraksi dengan sistem secara optimal.

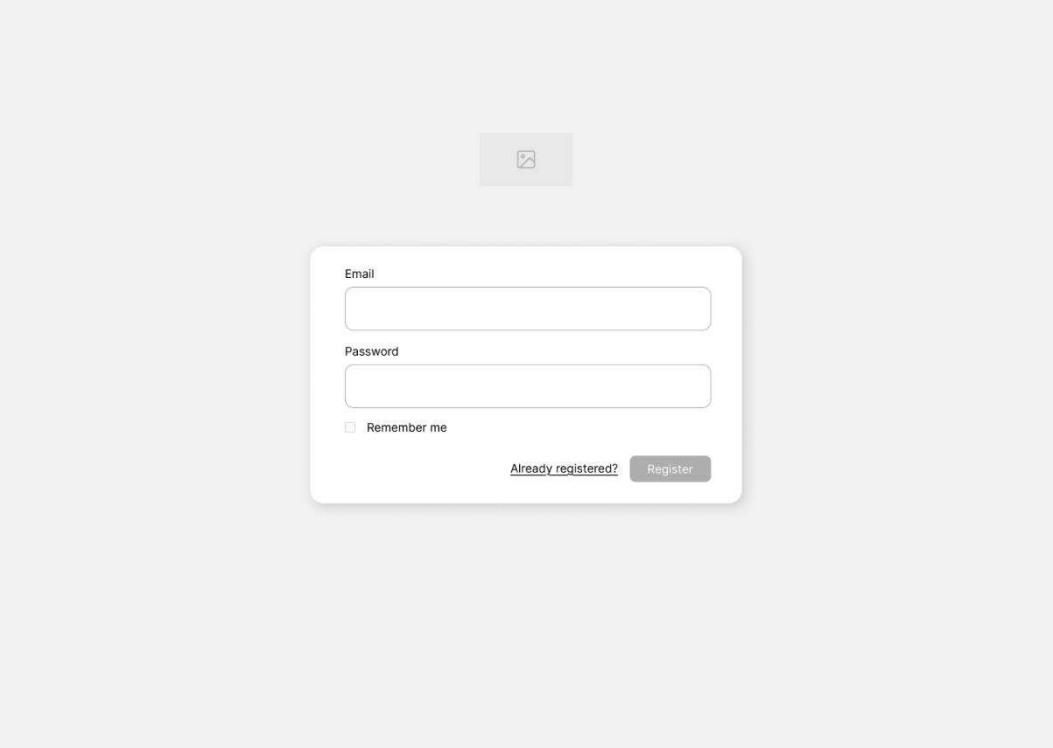
1. Halaman *Register*



Gambar 3.14 Halaman *Register*

Halaman *register* dirancang dengan tata letak yang sederhana dan minimalis untuk memudahkan pengguna dalam melakukan pendaftaran akun. Formulir pendaftaran terdiri dari empat *Input* utama seperti nama, email, password, dan konfirmasi password, yang disusun secara vertikal untuk menjaga alur pengisian tetap terstruktur dan mudah diikuti. Setiap kolom *Input* diberi label yang jelas agar pengguna dapat memahami informasi yang harus diisi tanpa kebingungan. Tombol "*Register*" diletakkan di kanan bawah sebagai aksi utama, disertai tautan alternatif "*Already register*ed?" bagi pengguna yang sudah memiliki akun.

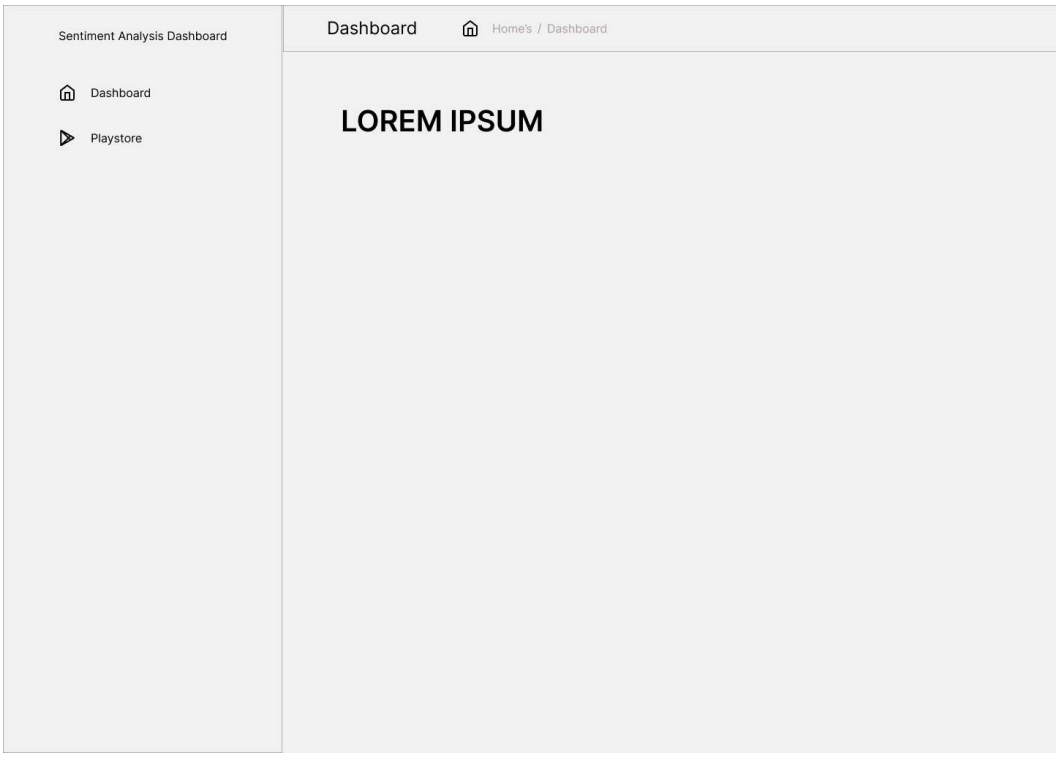
2. Halaman *Login*



Gambar 3.15 Halaman *Login*

Halaman *Login* dirancang dengan tampilan yang ederhana untuk memastikan proses masuk ke dalam sistem dapat dilakukan dengan cepat dengan hanya menyediakan 2 *Input* utama, yaitu email dan password, yang disusun secara vertikal agar mudah diisi oleh pengguna. Di bawah form tersedia opsi *"Remember me"* yang memungkinkan pengguna tetap masuk tanpa perlu *login* ulang. Tautan *"Already registered?"* dan tombol *"Register"* disediakan sebagai navigasi tambahan jika pengguna belum memiliki akun, mendukung alur autentikasi yang jelas dan terstruktur.

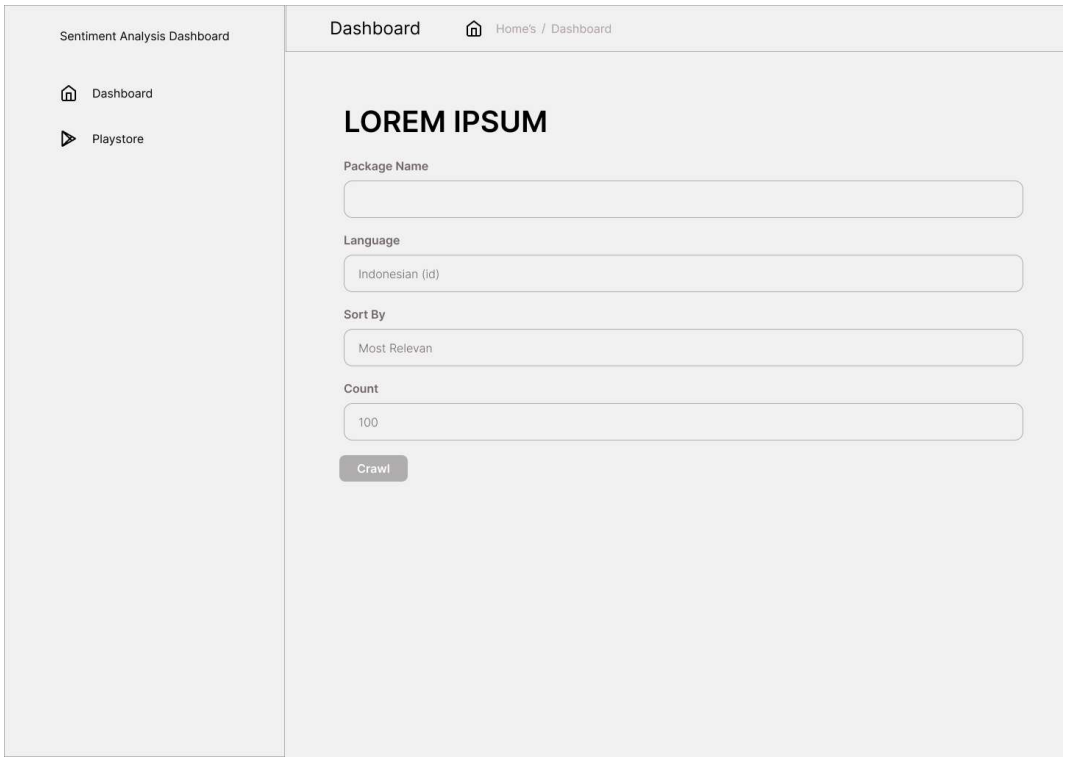
3. Halaman *Home*



Gambar 3.16 Halaman *Home*

Halaman *Home* dirancang dengan tampilan sederhana yang dimana ketika pengguna masuk hanya menampilkan 1 kalimat sapaan, kemudian di bagian kiri, terdapat *sidebar* vertikal yang menampilkan menu navigasi utama seperti *dashboard* dan *Playstore*, lengkap dengan ikon untuk memperjelas fungsi setiap menu. Bagian atas halaman menampilkan judul halaman aktif dan *breadcrumb* navigation, membantu pengguna memahami posisi mereka dalam struktur aplikasi, Warna latar yang netral dan dipilih agar tampilan tetap bersih dan fokus pengguna tertuju pada konten pendekatan desain ini memprioritaskan kemudahan akses dan kenyamanan pengguna dalam menjelajahi fitur-fitur *dashboard*

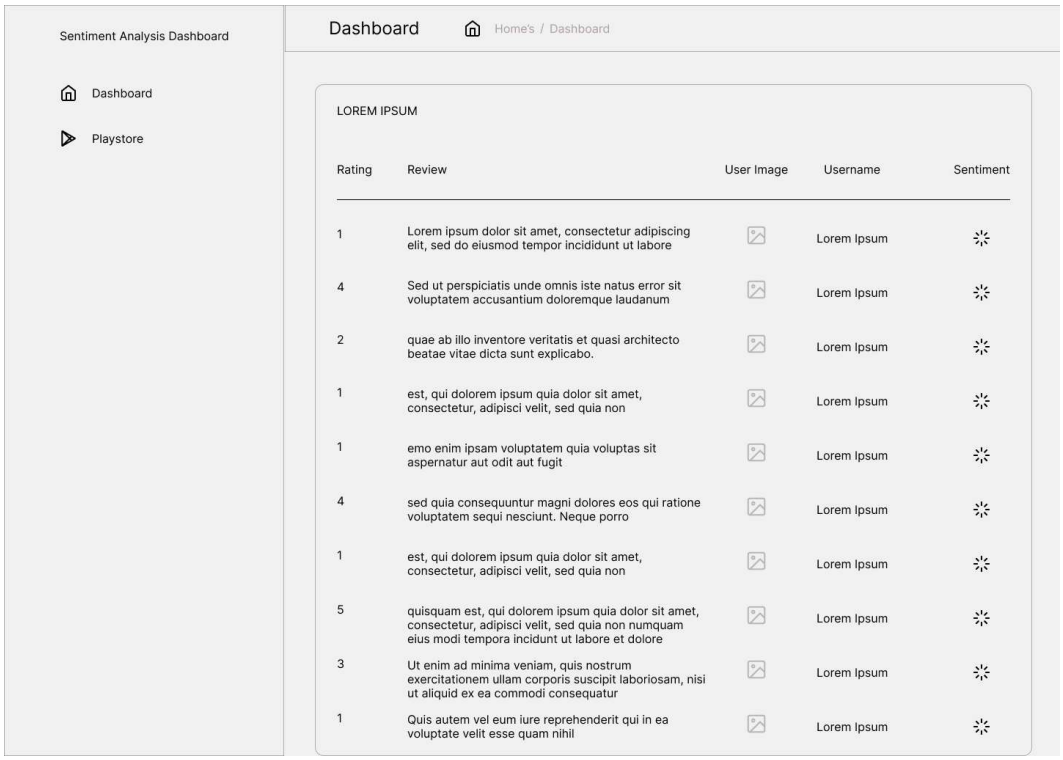
4. Halaman Klasifikasi



Gambar 3.17 Halaman Klasifikasi

Halaman klasifikasi sentimen dirancang untuk memberikan *Input* parameter sebelum sistem melakukan proses pengambilan dan analisis data ulasan dari *Google Play Store*. Halaman ini dapat diakses melalui navigasi "*Playstore*" yang tersedia di *sidebar*, komponen utama pada halaman ini terdiri atas kolom *Input Package name*, *dropdown language* untuk menentukan bahasa ulasan, opsi *Sort By* untuk mengatur urutan data berdasarkan relevansi, serta kolom *Count* untuk menentukan jumlah ulasan yang akan diambil, kemudian tombol *Scrap* digunakan untuk menjalankan proses pengambilan data dan analisis *sentiment* dari *Google Play Store*.

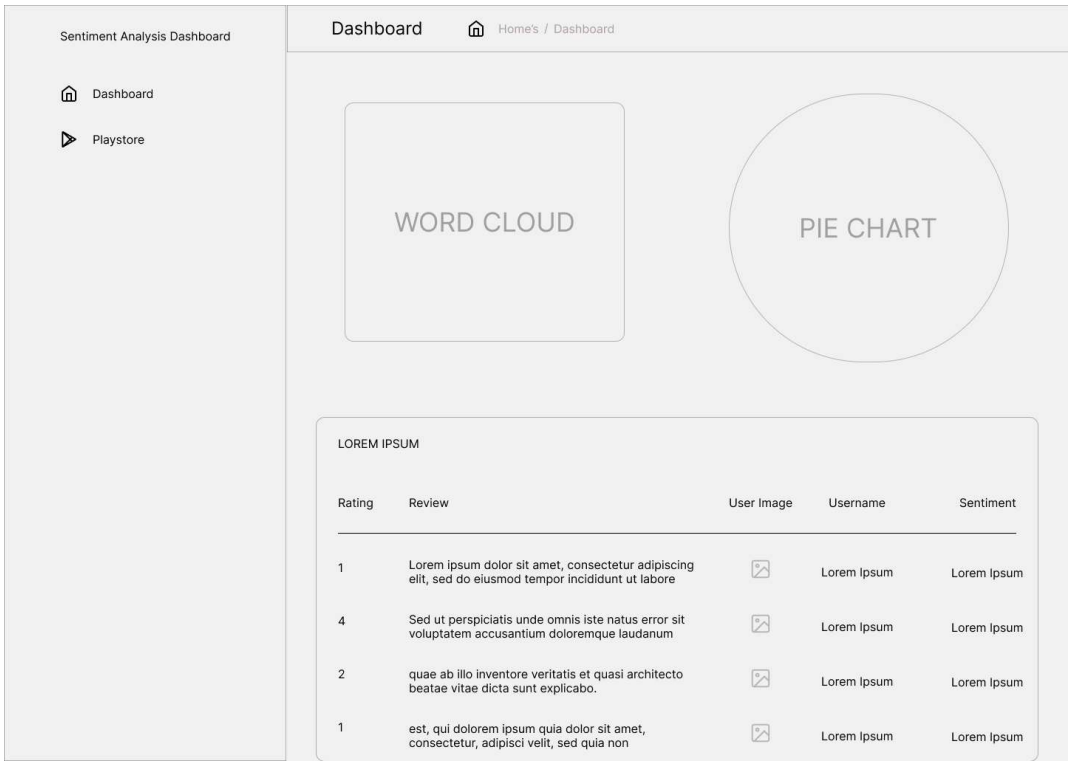
5. Ulasan yang berhasil diambil dari *Google Play Store*



Gambar 3.18 Ulasan yang berhasil diambil dari *Google Play Store*

Desain antarmuka untuk hasil data ulasan ditampilkan dalam bentuk tabel setelah pengguna menekan tombol *Scrap* pada halaman klasifikasi, tabel ini memuat beberapa kolom utama, yaitu Rating, *Review*, *User Image*, *User*name, dan *Sentiment*. Data yang ditampilkan berasal dari *Google Play Store* berdasarkan parameter *Input* yang telah dimasukkan sebelumnya. Setiap baris merepresentasikan satu ulasan pengguna aplikasi, lengkap dengan nilai rating dan isi ulasan. Kolom *Sentiment* menampilkan ikon pemrosesan sebagai indikator bahwa sistem sedang melakukan analisis sentimen terhadap masing-masing ulasan.

6. Hasil Sentimen dan visualisasi data



Gambar 3.19 Hasil Sentimen dan visualisasi data

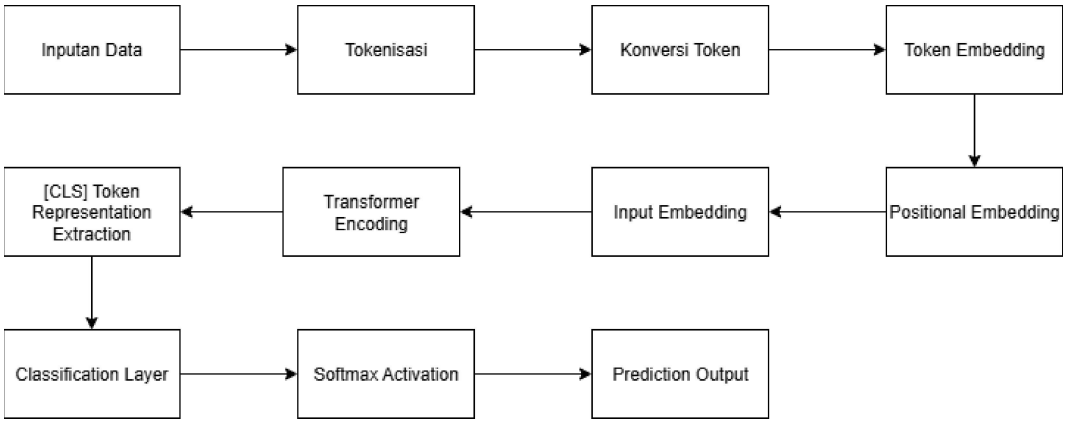
Tampilan ini merupakan tampilan setelah sistem menyelesaikan proses klasifikasi sentimen terhadap ulasan yang diambil dari *Google Play Store*, hasil analisis ditampilkan secara langsung pada halaman klasifikasi. Tampilan dibagi menjadi dua bagian utama, yaitu visualisasi dan tabel data, di atas table terdapat visualisasi data berupa *word cloud* dan *pie chart* disusun secara horizontal, di bagian bawah, hasil ulasan disajikan dalam bentuk tabel yang memuat informasi rating, isi ulasan, gambar pengguna, nama pengguna, dan hasil sentimen dari masing-masing ulasan.

3.6. Algoritma

Secara umum, algoritma dapat didefinisikan sebagai serangkaian langkah-langkah atau prosedur yang dilakukan secara berurutan untuk mencapai hasil yang diinginkan. Algoritma ini bisa sangat sederhana, seperti algoritma untuk menjumlahkan dua angka, atau sangat kompleks, seperti algoritma yang digunakan dalam *machine learning* untuk menganalisis data besar.

3.6.1. Proses *RoBERTa* dalama analsis sentimen

RoBERTa bekerja dengan cara mengolah *Input* untuk menghasilkan *output* yang diinginkan. Setiap langkah dalam algoritma harus didefinisikan dengan jelas, agar tidak menimbulkan ambiguitas selama eksekusi. Dalam analisis sentimen, algoritma yang digunakan akan menentukan bagaimana model *machine learning*, seperti *Indonesian-RoBERTa*, bekerja dalam mengklasifikasikan sentimen dari data yang dikumpulkan. Algoritma yang dipilih harus mampu menangani tantangan spesifik dari data yang ada dan memberikan hasil yang akurat dan reliabel.



Gambar 3.20 Proses *RoBERTa* untuk klasifikasi teks

(**Sumber:** Penelitian 2025)

Proses kerja *RoBERTa* dalam tugas analisis sentimen melibatkan beberapa langkah penting yang memungkinkan model untuk memahami dan menganalisis teks dengan lebih baik. Berikut adalah penjelasan detail tentang bagaimana proses kerja *RoBERTa* dalam melakukan analisis sentimen:

1. *Input* Teks

Langkah paling awal dalam proses kerja model *RoBERTa* adalah tahap penerimaan dan persiapan teks *Input*, yaitu kalimat yang akan dianalisis oleh model. Dalam konteks analisis sentimen, *Input* biasanya berupa kalimat opini, ulasan, atau pernyataan subjektif yang ingin diklasifikasikan ke dalam kategori sentimen tertentu (misalnya: positif, negatif, atau netral). Model *RoBERTa* dirancang untuk menerima *Input* dalam bentuk teks mentah (*raw text*), namun sebelum dapat diproses oleh jaringan saraf transformer, teks tersebut harus melalui sejumlah tahapan pra-pemrosesan awal agar sesuai dengan format masukan model

2. Tokenisasi

Pada tahap tokenisasi, teks yang dimasukkan dipecah menjadi potongan-potongan kecil bernama token. *RoBERTa* menggunakan metode yang memecah kata menjadi bagian-bagian lebih kecil agar bisa memahami kata-kata baru yang tidak dikenal sebelumnya. Misalnya, kata "aplikasi ini sangat berguna dan saya suka sekali" bisa dipisah menjadi ["aplikasi", "ini", "sangat", "berguna", "dan", "saya", "suka", "sekali"] potongan-potongan ini dipilih berdasarkan seberapa sering mereka muncul dalam data pelatihan. Setelah dipotong, setiap bagian diubah menjadi angka sesuai daftar yang sudah ada. Angka-angka inilah yang nantinya digunakan dalam proses selanjutnya agar model bisa memproses teks secara matematis.

3. Konvsersi Token

Setelah teks diubah menjadi token melalui proses tokenisasi, langkah selanjutnya adalah mengonversi setiap token tersebut menjadi bentuk numerik yang disebut sebagai Token ID. Dalam proses ini setiap token ditransformasikan menjadi bilangan bulat (integer) yang mengacu pada indeks token dalam *vocabulary* (daftar kosakata) milik model *RoBERTa*. Setiap token (hasil dari *tokenizer*) akan dicocokkan dengan kamus *vocabulary*, jika token ditemukan, maka ID-nya akan diambil dan digunakan. Jika tidak ditemukan (*rare*, karena menggunakan BPE), maka *tokenizer* akan memecah token lebih lanjut menjadi unit yang dikenal.

Contoh teks *Input*:

"Aplikasi ini sangat berguna"

Token hasil tokenisasi:

["<s>", "Ap", "lik", "asi", "ini", "sangat", "ber", "guna", </s>"]

Hasil konversi token ke token ID:

[0, 1348, 4513, 2098, 1012, 2031, 3981, 5023, 2]

4. Token *Embedding*

Setelah token dikonversi menjadi Token ID, langkah berikutnya adalah melakukan proses token *embedding*, yaitu mengambil representasi vektor berdimensi tetap dari setiap token berdasarkan Token ID-nya. Proses ini dilakukan dengan mengakses vektor dari matriks *embedding*, yang memiliki ukuran [*vocab_size* × *hidden_size*]. Vektor-vektor *embedding* ini telah dipelajari selama tahap *pretraining* dan mengandung informasi semantik awal dari masing-masing token. Dimensi dari token *embedding* ditentukan oleh arsitektur model, pada

RoBERTa-base, dimensi ini adalah 768. *Output* dari tahap ini adalah tensor berdimensi [*sequence_length* × *hidden_size*], yang kemudian akan dikombinasikan dengan *positional Embedding* pada tahap berikutnya untuk membentuk *Input* akhir ke *encoder* (Sy et al. 2024).

5. *Positional Embedding*

Transformer tidak memiliki arsitektur sekuensial seperti RNN, sehingga tidak memiliki kesadaran intrinsik terhadap posisi token dalam kalimat oleh karena itu, model *RoBERTa* menambahkan informasi posisi ke setiap token melalui mekanisme *positional Embedding*. *Positional Embedding* merupakan vektor berdimensi tetap yang mewakili posisi token dalam urutan *Input*, vektor ini diperoleh dari tabel *embedding* posisi yang telah dipelajari selama tahap pre*training*. Untuk setiap posisi ke-N dalam kalimat, diambil vektor posisi yang sesuai dan dijumlahkan secara element-wise dengan vektor token *embedding* (Guo, Zhu, and Han 2021). Poses singkat yang terjadi pada tahap *positional Embedding* adalah seperti berikut:

1) Setiap token pada urutan *input* diberi posisi indeks numerik**,** token pertama = 0, kedua = 1, dan seterusnya.
2) Posisi tersebut digunakan untuk mengambil vektor positional embedding dari tabel embedding posisi.
3) Vektor ini kemudian ditambahkan secara elemen-per-elemen (element-wise addition) dengan vektor token embedding yang sesuai.
4) Hasil penjumlahan tersebut adalah Input embedding akhir, yang selanjutnya diproses oleh encoder stack.

6. *Input Embedding*

Setelah memperoleh token *embedding* dan *positional embedding*, tahap berikutnya adalah membentuk representasi *Input* akhir yang disebut sebagai *Input embedding*. Proses ini dilakukan dengan cara menjumlahkan setiap vektor token *embedding* dengan vektor *positional embedding* yang sesuai berdasarkan posisi token tersebut dalam urutan *Input*. Penjumlahan dilakukan secara element-wise, sehingga dimensi hasil penjumlahan tetap sama, yaitu [*sequence_length* × *hidden_size*]. *Input embedding* ini menjadi representasi awal dari kalimat atau teks yang akan diproses oleh *encoder* dalam arsitektur Transformer. (Lai and Zhang 2023).

7. Transformers *Encoding*

Tahap Transformer *Encoding* merupakan inti dari arsitektur *RoBERTa*, di mana representasi awal dari token yang telah dibentuk melalui token *embedding* dan *positional embedding* diproses lebih lanjut untuk menangkap informasi kontekstual. Pemrosesan dilakukan oleh susunan lapisan *encoder* yang masing-masing terdiri atas dua komponen utama, yaitu multi-*head Self-attention* dan feed-*forward neural network* (FFNN).

Self-attention memungkinkan model untuk memperhatikan hubungan antar-token dalam satu urutan *Input*, sehingga representasi akhir dari setiap token mencerminkan konteks global kalimat. FFNN berfungsi mengubah representasi token secara *non-linear* agar informasi semantik lebih kaya. *Output* dari setiap lapisan *encoder* dilengkapi dengan residual connection dan *layer* normalization untuk mempercepat konvergensi dan menjaga stabilitas pelatihan.

8. [CLS] *token representation extraction*

Pada tahap akhir proses *encoding*, model *RoBERTa* menghasilkan representasi vektor untuk seluruh token dalam urutan *Input*. Namun, dalam tugas klasifikasi seperti analisis sentimen, hanya satu representasi yang dibutuhkan untuk mewakili seluruh teks. Untuk itu, digunakan token khusus <s> yang secara konvensi ditempatkan di posisi awal setiap urutan *Input*.

Token <s> akan diproses bersama token lainnya melalui semua lapisan *encoder*, dan hasil akhir dari token ini dianggap sebagai representasi keseluruhan kalimat atau dokumen. Representasi ini kemudian diambil sebagai vektor fitur utama untuk klasifikasi. Proses ini disebut sebagai [CLS] token *representation extraction*, karena mengekstraksi informasi dari token khusus yang bertugas menangkap makna global *Input*.

9. *Classification layer*

Setelah memperoleh vektor representasi kontekstual dari token <s> melalui tahapan *encoding* Transformer, proses selanjutnya adalah mengklasifikasikan vektor tersebut ke dalam label sentimen. Untuk itu, digunakan lapisan klasifikasi berupa satu *layer linear* (*fully connected*) yang memetakan dimensi vektor *Input* (misalnya 768) ke jumlah kelas yang ditentukan (misalnya 3 kelas: positif, netral, negatif).

Lapisan ini menghasilkan *logits*, yaitu skor mentah untuk masing-masing kelas. Selama tahap inferensi, *logits* ini akan dikonversi menjadi probabilitas menggunakan fungsi aktivasi *softmax*, dan label dengan probabilitas tertinggi akan diambil sebagai hasil prediksi.

Selama pelatihan, fungsi *cross-entropy loss* digunakan untuk mengukur kesalahan prediksi terhadap label yang benar, dan bobot model diperbarui melalui proses *backpropagation*. Dengan demikian, *classification layer* bertindak sebagai penghubung langsung antara representasi hasil *encoding* dan *output* akhir berupa label sentimen.

10. *Prediction output*

Setelah model menghasilkan *logits* melalui lapisan klasifikasi, tahap akhir dari sistem adalah menghasilkan prediksi kelas sentimen berdasarkan nilai *logits* tersebut, proses ini disebut sebagai *Prediction output*. Pada tahap ini, nilai *logits* yang belum terkalibrasi akan diinterpretasikan sebagai representasi kepercayaan model terhadap masing-masing kelas. Model memilih kelas dengan nilai *logits* tertinggi sebagai kelas prediksi akhir, yang kemudian dikonversi ke dalam bentuk label kategorikal seperti "Positif", "Netral", atau "Negatif".

Fungsi *argmax* digunakan untuk menentukan indeks dari nilai *logits* tertinggi, dan indeks tersebut dipetakan ke label sentimen sesuai dengan urutan kelas yang telah ditentukan. Proses ini menjadi titik akhir dalam *pipeline* klasifikasi sentimen, menghasilkan *output* yang dapat digunakan untuk analisis, pelaporan, atau visualisasi hasil.

3.6.2. Implementasi model *Indonesian-RoBERTa*

Adapun proses yang penulis lakukan dalam mengimplementasikan model hasil *finetuning Indonesian-RoBERTa* adalah sebagai berikut.

Tabel 3.5 Implementasi Model *Indonesian-RoBERTa*

Langkah	Gambar
Inisialisasi Model *Indonesian-RoBERTa*	
Model "*w11wo/Indonesian-RoBERTa-base-sentiment-classifier*" diinisialisasi menggunakan pustaka Transformers dari *Hugging Face*. Model ini dirancang khusus untuk analisis sentimen dalam bahasa Indonesia. Proses inisialisasi dilakukan melalui fungsi *pipeline*, yang secara otomatis memuat model dan *tokenizer*, serta menyiapkannya untuk memproses teks dan menghasilkan prediksi sentimen secara langsung.	

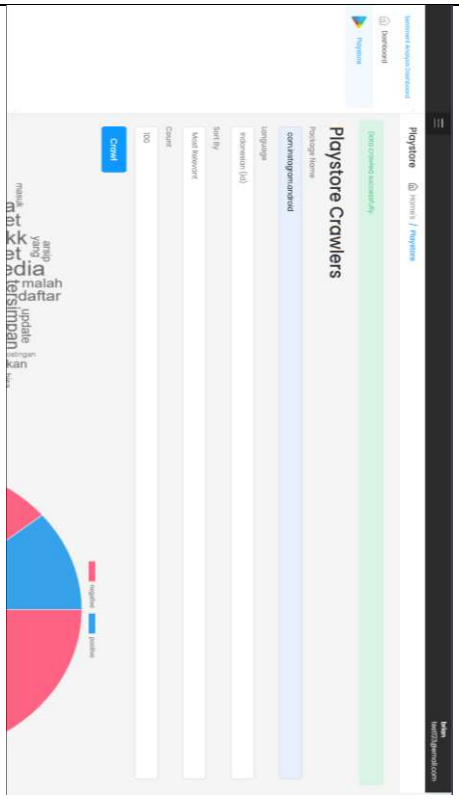
```
from transformers import pipeline
import pandas as pd

pretrained_name = "w11wo/indonesian-roberta-base-sentiment-classifier"

nlp = pipeline(
    "sentiment-analysis",
    model=pretrained_name,
    tokenizer=pretrained_name
)
```

Pembuatan API untuk Prediksi Sentimen	
Setelah model berhasil diinisialisasi, tahap selanjutnya adalah membangun antarmuka yang memungkinkan sistem menerima *Input* teks, memprosesnya dengan model, dan mengembalikan hasil klasifikasi sentimen. Untuk keperluan ini digunakan *framework* web Flask, yaitu *framework* Python yang ringan dan fleksibel untuk pengembangan aplikasi berbasis web. Sebuah API endpoint dibuat pada *rute /api/predict* dengan metode POST, yang menerima masukan berupa teks dalam format JSON. API ini bertugas menerima data dari *frontend*, menjalankan analisis sentimen menggunakan model *Indonesian-RoBERTa*, lalu mengembalikan label sentimen sebagai respons.	 ```python @app.route("/api/predict", methods=["POST"]) def prediction(): if request.method == "POST": input_data = request.get_json() texts = input_data["texts"] # Expecting a list of texts results = [nlp(text) for text in texts] return jsonify({ "status": { "code": 200, "message": "Success predicting the sentiment" }, "data": { "sentiments": results, } }), 200 else: return jsonify({ "status": { "code": 405, "message": "Method not allowed" }, "data": None }), 405 ```

<table>
<tr><td colspan="2">Integrasi Frontend dengan API</td></tr>
<tr><td>Di sisi frontend, interaksi dengan API dilakukan menggunakan JavaScript. Saat pengguna mengirimkan teks untuk dianalisis, data tersebut dikirim ke endpoint /api/predict menggunakan metode POST. Setelah API memproses Input dan menghasilkan prediksi sentimen, hasilnya dikembalikan dalam format JSON. Data hasil analisis kemudian digunakan untuk memperbarui antarmuka pengguna (User interface) secara dinamis, seperti menampilkan label sentimen pada tabel atau memperbarui elemen visual lainnya sesuai dengan hasil klasifikasi.</td><td>

```
        // Call your API endpoint with the texts for sentiment analysis
        fetch("/api/predict", {
            method: "POST",
            headers: {
                "Content-Type": "application/json",
            },
            body: JSON.stringify({ texts: reviewTexts }),
        })
            .then((response) => response.json())
            .then((data) => {
                this.updateTableWithSentiments(
                    reviewsTable,
                    data.data.sentiments
                );
                this.loading = false;
            })
            .catch((error) => {
                console.error(
                    "Error fetching and predicting:",
                    error
                );
                this.loading = false;
            });
    });
},
```

</td></tr>
</table>

Menjalankan program Web *Dashboard*	
Setelah proses integrasi antara model analisis sentimen dan antarmuka *frontend* berhasil diselesaikan, sistem dapat dijalankan untuk mengevaluasi fungsionalitas secara keseluruhan. *Dashboard* yang telah dibangun memungkinkan pengguna untuk melakukan *scraping* data ulasan atau komentar dari berbagai sumber, dan hasilnya akan langsung dianalisis oleh model *Indonesian-RoBERTa* secara asinkron.	

(Sumber: Penelitian 2025)

3.7. Waktu dan tempat penelitian

Pada sub bab ini, penulis akan menjelaskan mengenai tempat dan waktu pelaksanaan penelitian yang dilakukan. Penjelasan ini mencakup lokasi di mana seluruh proses penelitian dilaksanakan serta periode waktu yang digunakan untuk menyelesaikan setiap tahapan penelitian, mulai dari studi literatur, pengumpulan data, hingga analisis data.

3.7.1. Tempat penelitian

Penelitian ini seluruhnya dilakukan di tempat tinggal penulis, yang memberikan fleksibilitas dalam menjalankan setiap tahapan penelitian. Penggunaan tempat tinggal penulis sebagai tempat penelitian memungkinkan penulis untuk bekerja dalam lingkungan yang nyaman dan terkendali, yang sangat mendukung konsentrasi dan fokus yang diperlukan dalam proses penelitian.

Dalam penelitian ini, seluruh proses studi literatur dilakukan di rumah, termasuk membaca jurnal, artikel, serta menonton video di YouTube yang relevan dengan topik yang diteliti. Penelitian ini tidak memerlukan observasi lapangan atau interaksi langsung dengan subjek di lokasi tertentu, sehingga tempat penelitian di rumah sudah memadai.

Pengumpulan data dilakukan dengan metode *web scraping*, di mana data sentimen diambil dari *platform Google Play Store*. Seluruh proses *scraping* ini juga dilakukan di rumah menggunakan perangkat komputer dan koneksi internet yang memadai. Data-data yang dikumpulkan dari berbagai sumber ini nantinya akan dianalisis untuk mencapai tujuan penelitian.

3.7.2. Waktu peneltian

Waktu penelitian yang dilakukan cukup fleksibel, menyesuaikan dengan ketersediaan penulis. Studi literatur, pengumpulan data, hingga analisis dilakukan dalam beberapa minggu, memastikan setiap tahapan penelitian dilaksanakan dengan teliti dan sesuai dengan rencana yang telah disusun.

Dengan demikian, tempat penelitian di rumah dan waktu penelitian yang fleksibel memberikan keleluasaan dalam menjalankan penelitian ini dengan optimal, tanpa terbatas oleh faktor eksternal seperti lokasi atau waktu yang kaku. Berikut ini adalah jadwal penelitian yang penulis lakukan

Tabel 3.6 Tabel kegiatan selama penelitian

Kegiatan	Tahun 2025															
	April				Mei				Juni				Juli			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Identifikasi Masalah																
Studi Literatur																
Membangun Web *Dashboard*																
Penyusunan Bab 1																
Penyusunan Bab 2																
Penyusunan Bab 3																
Finetuning Model *Indonesian-RoBERTa*																
Implementasi model kedalam web *dashboard*																
Penyusunan Bab 4																
Penyusunan Bab 5																

(**Sumber:** Penelitian 2025)