BAB IV
PERANCANGAN

Pada bab ini menjelaskan secara detail mengenai analisis sistem terdahulu, spesifikasi sistem kebutuhan baru, perancangan sistem, perancangan tatap muka dan langkah-langkah perancangan sistem.

4.1 Analisis Sistem Terdahulu

Analisis sistem terdahulu bertujuan untuk menganalisa cara kerja, kelebihan, kekurangan, dan kebutuhan sistem terdahulu. Pada tahap ini, peneliti dapat menemukan hal yang dibutuhkan sistem berdasarkan pengamatan. Analisis sistem terdahulu dapat memastikan perubahan sistem yang akan dikembangkan dan menutup kekurangan sistem sebelumnya.

Pada penelitian yang dilakukan oleh Rozi (2018), aplikasi analisis sentimen dibangun menggunakan bahasa pemrograman PHP dan algoritma *Naïve Bayes* untuk menentukan polaritas kalimat. Topik yang dibahas pada sistem terdahulu yaitu sentimen terhadap SAMSAT Kota Malang. Metode pengolahan data atau pra-proses data hanya menggunakan filter duplikasi tweet, *case folding*, *cleaning*, *tokenizing*, dan *stopword removal*.

4.2 Spesifikasi Kebutuhan Sistem Baru

Tahap yang dilakukan setelah melakukan analisis sistem terdahulu yaitu menentukan spesifikasi kebutuhan untuk sistem yang akan dibangun. Spesifikasi sistem diperlukan untuk mecapai tujuan penelitian ini yaitu membangun aplikasi analisis sentimen dengan metode *LSTM*. Kebutuhan sistem terdiri dari beberapa bagian yaitu sebagai berikut.

1) Spesifikasi Perangkat Lunak *(Software)*

Perangkat lunak merupakan untuk terpenting dalam melakukan pengembangan sistem. Pada penelitian ini, spesifikasi perangkat lunak ditentukan oleh peneliti dengan tujuan membantu pengembangan sistem. Berikut merupakan tabel spesifikasi perangkat lunak yang digunakan.

Tabel 4. 1 Kebutuhan Perangkat Lunak

No	Kebutuhan Perangkat Lunak
1	Web Browser
2	Pycharm Professional IDE
3	Python 3.9 atau lebih tinggi
4	Sistem Operasi Windows
5	Koneksi Internet

Tabel 4.1 merupakan spesifikasi perangkat lunak yang digunakan oleh peneliti dalam mengembangkan aplikasi dan model *LSTM*. *Web Browser* digunakan untuk menampilkan dan mengoperasikan aplikasi. Lingkungan pengembangan yang digunakan oleh peneliti adalah *IDE* dari *JetBrains* khusus untuk pemrograman Bahasa *python* yaitu *PyCharm*. Interpreter Python 3.9 dipilih karena stabil dan telah dirilis kurun waktu yang lama. Sistem operasi yang digunakan adalah *Windows* yang terhubung dengan internet.

2) Spesifikasi Perangkat Keras *(Hardware)*

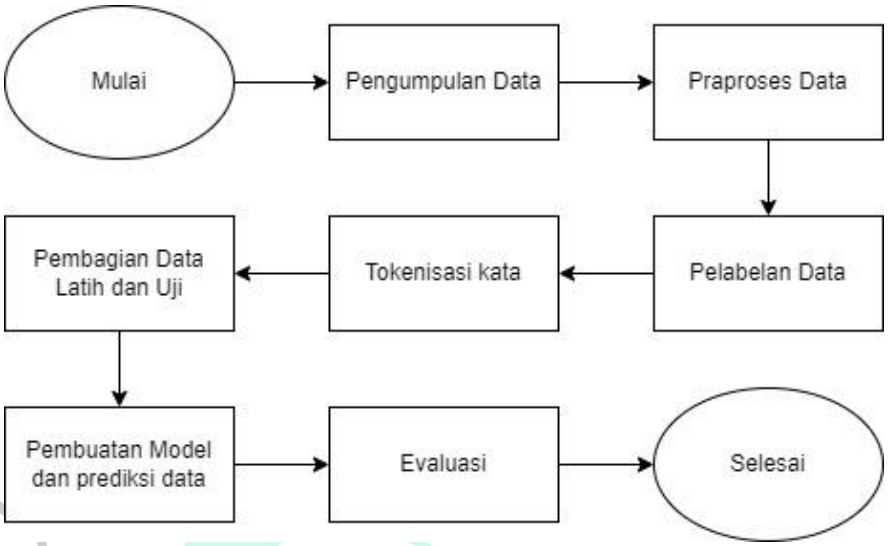
Perangkat keras diperlukan untuk mendukung pengoperasian perangkat lunak dan perangkat keras. Berikut perangkat keras yang digunakan peneliti untuk pengembangan aplikasi.

Tabel 4. 2 Kebutuhan Perangkat Keras

Perangkat Keras	Spesifikasi
RAM	16GB
Prosesor	Intel I series
GPU	GTX 1070 8GB
Penyimpanan	1TB HDD + 512GB NVME

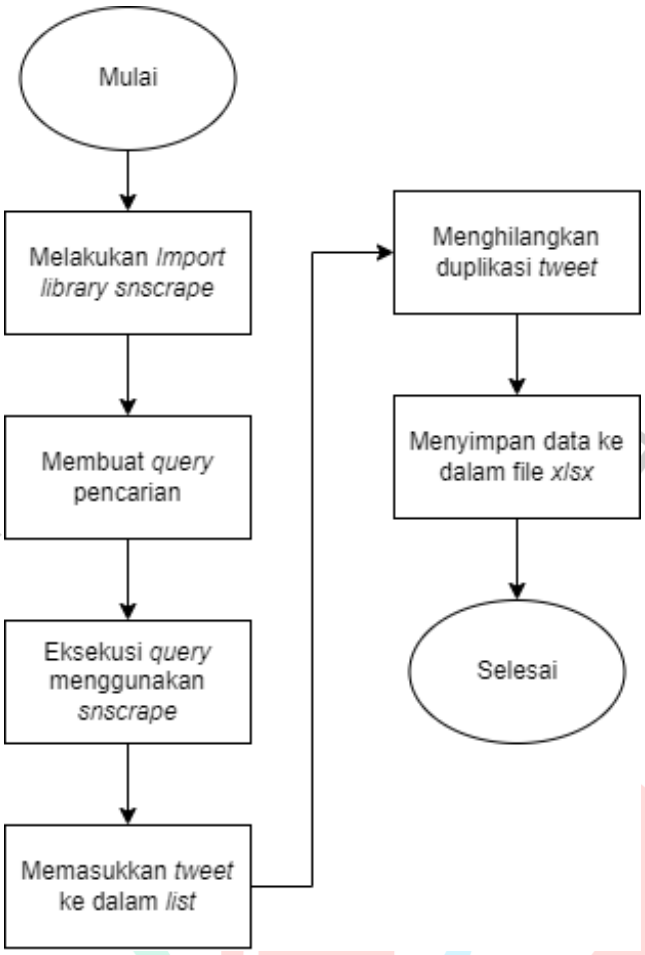
Tabel 4.2 merupakan daftar perangkat keras yang digunakan untuk menjalankan seluruh perangkat lunak. Peneliti menggunakan RAM 16GB untuk menghindari lagging saat mengolah dataset. Prosesor dan GPU yang digunakan adalah intel core I series dan GTX 1070 dalam melakukan training model *LSTM*. Seluruh data yang berhubungan dengan aplikasi dan model disimpan pada HDD dan NVME.

3) Pembuatan Model



Gambar 4. 1 *Flowchart* Pembuatan Model NLP

Gambar 4.1 merupakan *flowchart* yang digunakan untuk membuat dan menguji model NLP. Proses pembuatan model merupakan bagian utama dalam penelitian ini. Prosedur yang dilakukan dalam tahap ini yaitu pengumpulan data, praproses data, pelabelan data, pembagian data latih dan uji, tokenisasi kata, membuat model dan melakukan predikisi data menggunakan berbagai skenario pengetesan, dan melakukan evaluasi terhadap model di berbagai skenario pengujian.

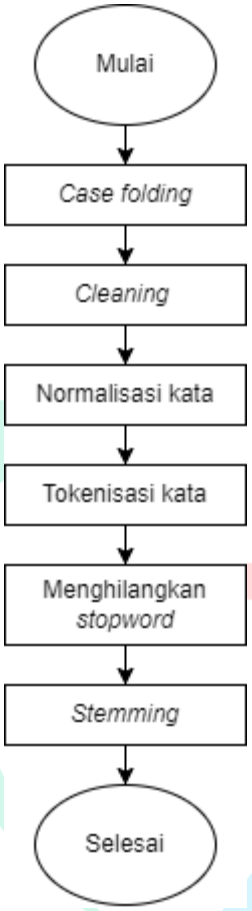


Gambar 4. 2 *Flowchart* Proses Pengumpulan Data

Gambar 4.2 merupakan *flowchart* untuk mengumpulkan data dari media sosial twitter. Hal yang dilakukan pertama kali yaitu menginstall *library snscrape*. Langkah berikutnya yaitu menyusun query yang ingin digunakan. *Query* berfungsi sebagai kumpulan data atau kata kunci yang ingin dicari dalam kolom pencarian.

Query yang disusun pada penelitian ini memiliki 2 verisi yaitu pencarian tweet Ketika e-tilang pertama kali diluncurkan dan e-tilang yang ingin diterapkan secara nasional. *Query* 1 yaitu "("e-tilang" OR "etilang" OR "etle" OR "tilang elektronik") lang:id since:2022-10-01 until:2022-12-31", sedangkan *Query* 2 yaitu "("e-tilang" OR "etilang" OR "etle" OR "tilang elektronik") lang:id since:2018-11-01 until:2019-01-31". *Query* dieksekusi menggunakan *library snscrape* untuk *scraping* atribut *tweet*. Hasil *scrape* dimasukkan ke dalam tipe data *list* dan diubah ke dalam tipe data *set*. Tujuannya yaitu mengurangi duplikasi *tweet*. Data yang

sudah difilter kemudian disimpan ke dalam tabel *DataFrame* dan disimpan ke dalam format *xlsx*.

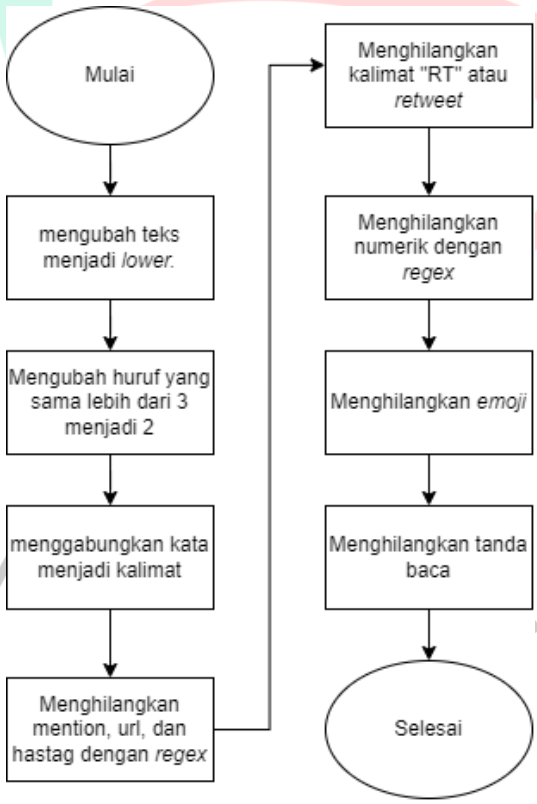


Gambar 4.3 *Flowchart* Pra-proses Data

Gambar 4.3 merupakan tahap setelah pengumpulan data yaitu praproses data. Pra-proses data adalah teknik untuk mengubah *raw data* menjadi format yang sesuai (Laurensz & Sediyono, 2021). Data yang diperoleh dari proses *crawling* tidak dapat dipakai untuk pemodelan karena data tersebut masih memiliki kata-kata dan simbol yang tidak diperlukan (Pintoko & Lhaksmana, 2018). Proses ini bertujuan untuk memperbaiki kualitas data, memudahkan pengolahan, dan analisis data di tahap selanjutnya. Data yang telah diolah akan digunakan sebagai data uji dan data latih model.

(i) *Case folding*, yaitu mengubah huruf kapital di sebuah kalimat menjadi huruf kecil atau *lowercase*. Maksud dari *case folding* yaitu menghindari kesalahan pemrosesan data akibat variasi huruf kecil dan besar. *Case folding* dapat dilakukan dengan cara memastikan tipe data variabel adalah *string*. Tipe data *string* memiliki fungsi bawaan untuk melakukan *case folding* yaitu *lower*. *Input* variabel tersebut dengan menambahkan *function lower* menjadi varibel_baru = variabel.lower().

(ii) *Cleaning*, yaitu membersihkan data dari unsur-unsur yang tidak diperlukan seperti *hashtag*, *mention*, emoji, angka, *hyperlinks*, *unicode*, simbol *retweet*, dan simbol baca lainnya selain huruf. Proses ini bertujuan untuk mengurangi *noise* pada data karena tidak diperlukan. Tujuan lain dari proses ini yaitu memperkecil ukuran *dataset* dan *corpus tokenizer* supaya proses latih model berjalan lebih cepat. Mayoritas proses *cleaning* menggunakan teknik *regex* atau *regular expression* untuk mencari pola *string* dan melakukan manipulasi.

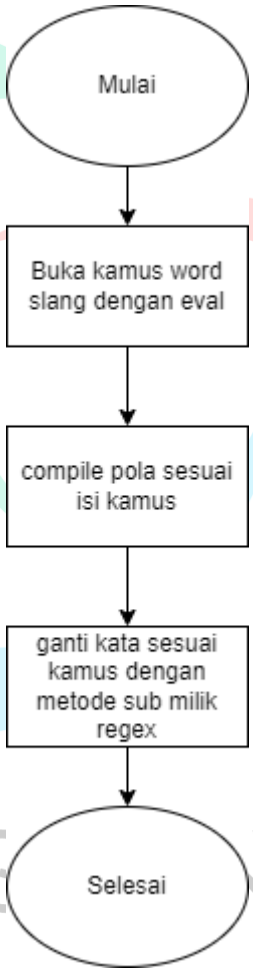


Gambar 4.4 *Flowchart* proses *Cleaning Data*

Gambar 4.4 merupakan *flowchart* untuk proses *cleaning*. Langkah awal yaitu mengubah teks menjadi lower kemudian mengubah 3 huruf atau lebih yang ditulis secara berturut-turut menjadi 2 huruf. Selanjutnya menggabungkan kata menjadi

kalimat. Kalimat tersebut kemudian dihilangkan *mention, url,* dan *hastag*. Langkah selanjutnya yaitu menghilangkan *RT* dan numerik menggunakan *regex*. Selanjutnya menghilangkan *emoji* di dalam kalimat dan menghilangkan tanda baca karena tidak diperlukan.

(iii) Normalisasi kata, yaitu memperbaiki istilah gaul atau tidak baku menjadi kata baku yang tersimpan dalam suatu file txt. Adapun sumber dari file normalisasi berasal dari *github repository* Pujangga, SentiStrenghtID, ramaprakoso, dan beberapa tambahan dari peneliti.



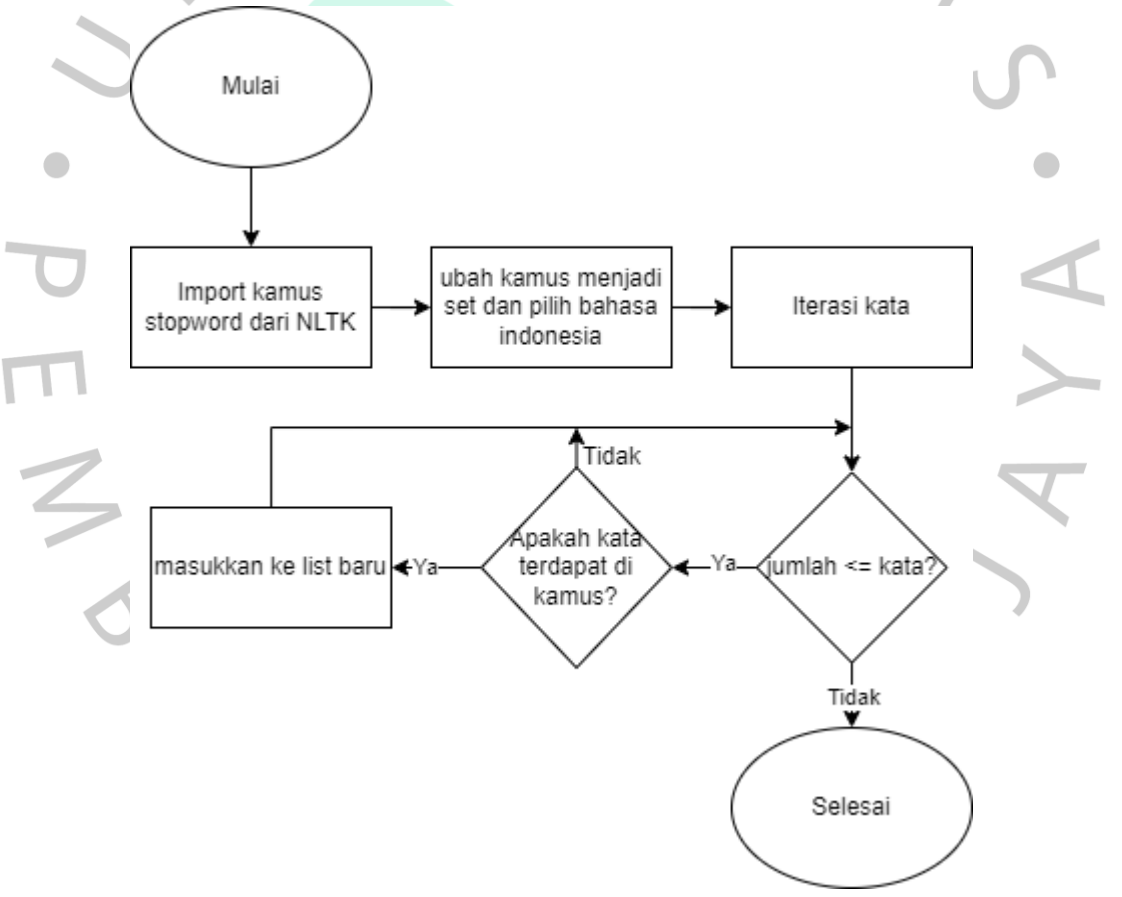
Gambar 4. 5 *Flowchart* Normalisasi Kata

Gambar 4.5 merupakan *flowchart* yang digunakan untuk melakukan normalisasi kata dalam kalimat. Adapun cara kerjanya yaitu membuka kamus normalisasi dan melakukan kompilasi pola berdasarkan kamus. Setelah itu kalimat

dimasukkan ke dalam fungsi untuk mencari kalimat yang sesuai dengan pola dan diganti dengan *substitution value* yang terdapat pada kamus normalisasi kata.

(iv) Tokenisasi, adalah proses mengubah teks menjadi urutan token atau satuan kata yang terpisah. Satu kalimat biasanya terdiri dari beberapa kata, kata tersebut dapat dipisah dan dimasukkan ke dalam *list* sebagai indeks yang berbeda.

(v) Menghilangkan *Stopword*, yaitu menghilangkan atau menghapus kata yang dianggap tidak penting pada suatu kalimat. Tahap ini juga berfungsi untuk mengurangi ukuran indeks korpus kata. Gambar 4.6 menggambarkan flowchart untuk melakukan *stopword removal*.

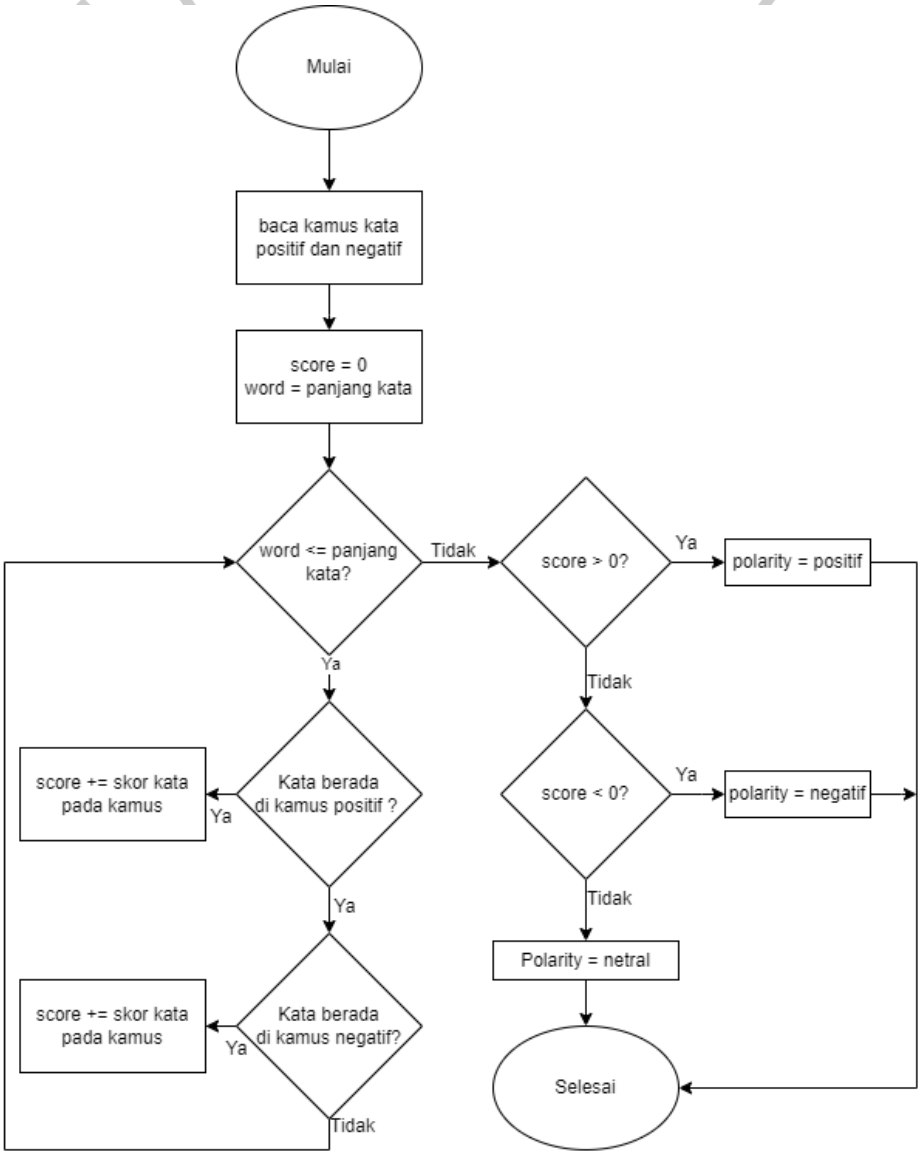


Gambar 4. 6 *Flowchart* untuk Menghilangkan *Stopword*

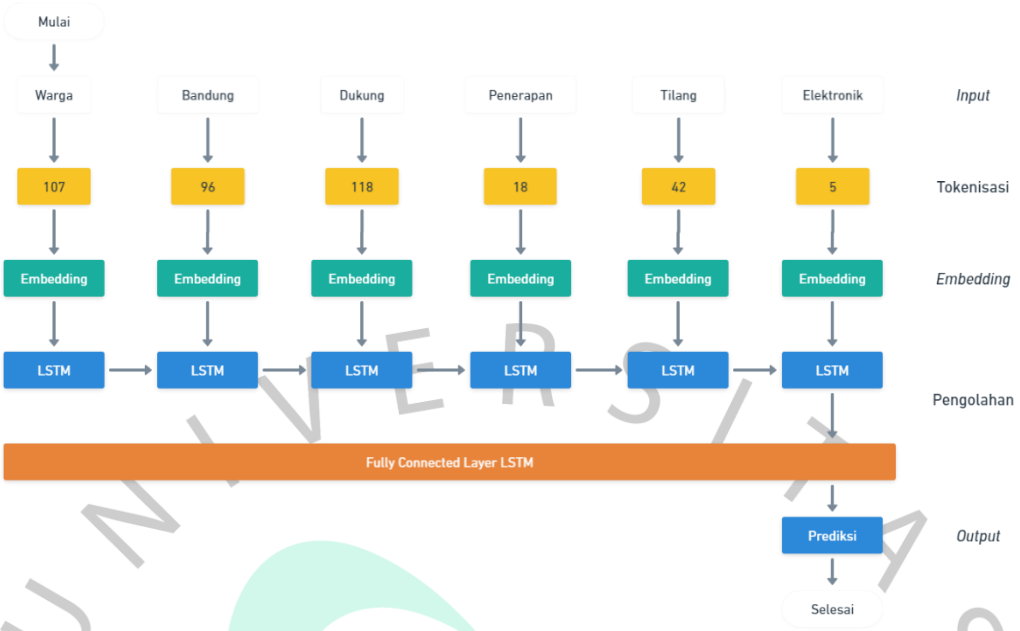
(vi) *Stemming*, yaitu mentransformasikan kata yang mengandung imbuhan seperti prefiks, sufiks, dan infiks menjadi kata dasar. Tahap ini bertujuan mengurangi indeks kata yang memiliki kata dasar yang sama. Jika fungsi *stemmer*

di *library Sastrawi* terasa lambat, maka gunakan *library swifter* untuk mempercepat proses stemming di Dataframe dengan menjalankan tugas secara paralel (Romli et al., 2021).

(vii) Pelabelan data, yaitu memberi label pada setiap data supaya model mengetahui klasifikasi setiap kalimat di dataset. Polaritas yang digunakan pada penelitian ini yaitu positif, negatif, dan netral. Pelabelan dilakukan menggunakan kamus kata negatif dan positif yang diterbitkan oleh (Koto,2017). Adapun Flowchart untuk pelabelan data otomatis dapat dilihat pada Gambar 4.7.



Gambar 4. 7 *Flowchart* Pelabelan Data otomatis



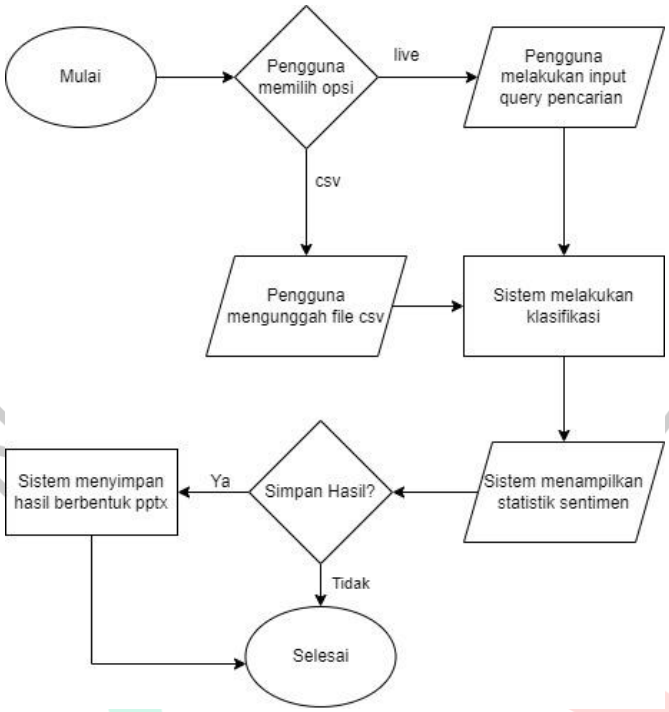
Gambar 4. 8 Arsitektur Model LSTM

Gambar 4.8 merupakan perancangan arsitektur yang digunakan untuk membangun model *LSTM*. Tahap pertama yaitu *input* berupa teks dan diubah menjadi representasi vektor berdasarkan *mapping* kata dari *tokenizer*. Kemudian vektor di-*feed* ke dalam layer *LSTM* dengan jumlah neuron yang telah ditentukan. Hasil atau output dari layer *LSTM* berupa *list* prediksi dengan jumlah anggota 3 dan memiliki nilai antara 0 hingga 2.

4.3 Perancangan Sistem

Proses perancangan sistem yaitu merencanakan atau mendesain sistem yang ingin dikembangkan. Pada sub bab ini, peneliti menjabarkan perancangan sistem pada aplikasi analisis sentimen berdasarkan pemodelan *Unifield Modeling Language* (*UML*). Model *UML* terdiri dari *Flowchart*, *Use Case diagram* dan *Activity diagram*.

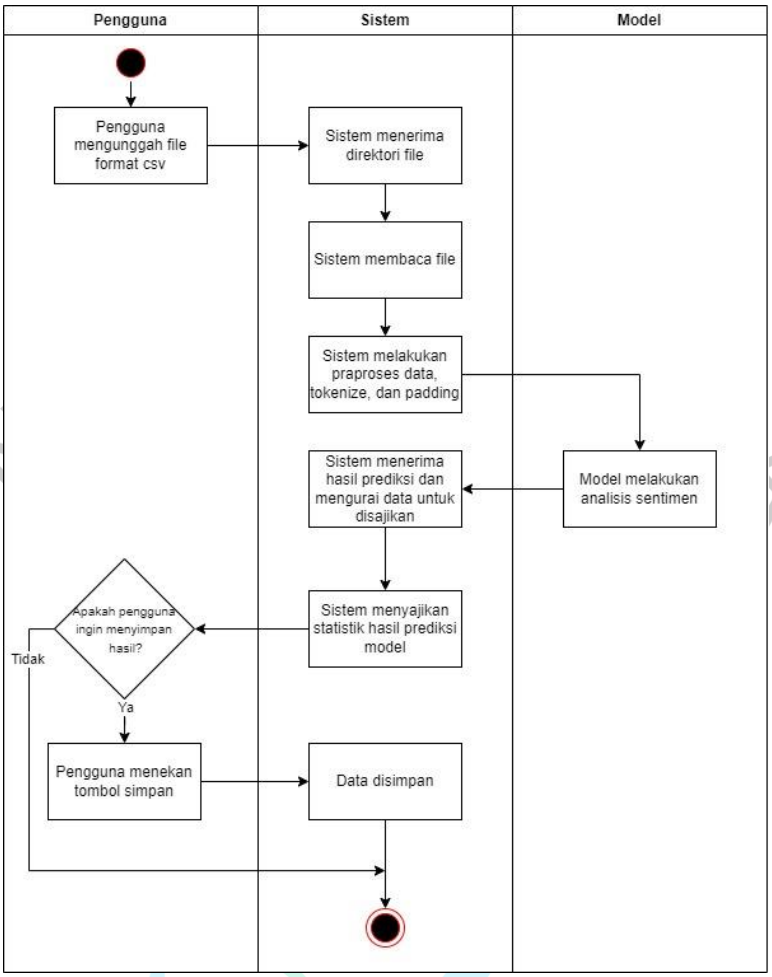
4.3.1 *Flowchart* Proses Bisnis Aplikasi



Gambar 4. 9 *Flowchart* Penggunaan Aplikasi

Gambar 4.9 merupakan *flowchart* dari sistem analisis sentimen yang dibuat. Tahap pertama dari sistem ini yaitu *user* mengakses laman, lalu mengunggah file berformat csv dengan *delimiter* titik koma (;). File yang telah terunggah kemudian dibaca dan diverifikasi oleh sistem. Jika file sesuai dengan format, maka sistem akan melakukan klasifikasi sentimen berdasarkan isi file. Hasil klasifikasi sentimen diolah oleh sistem untuk menampilkan statistik berupa total sentimen, sentimen terbanyak, diagram venn, dan diagram batang. Jika *user* ingin menyimpan statistik sentimen, maka sistem dapat menyimpannya ke dalam bentuk *power point*.

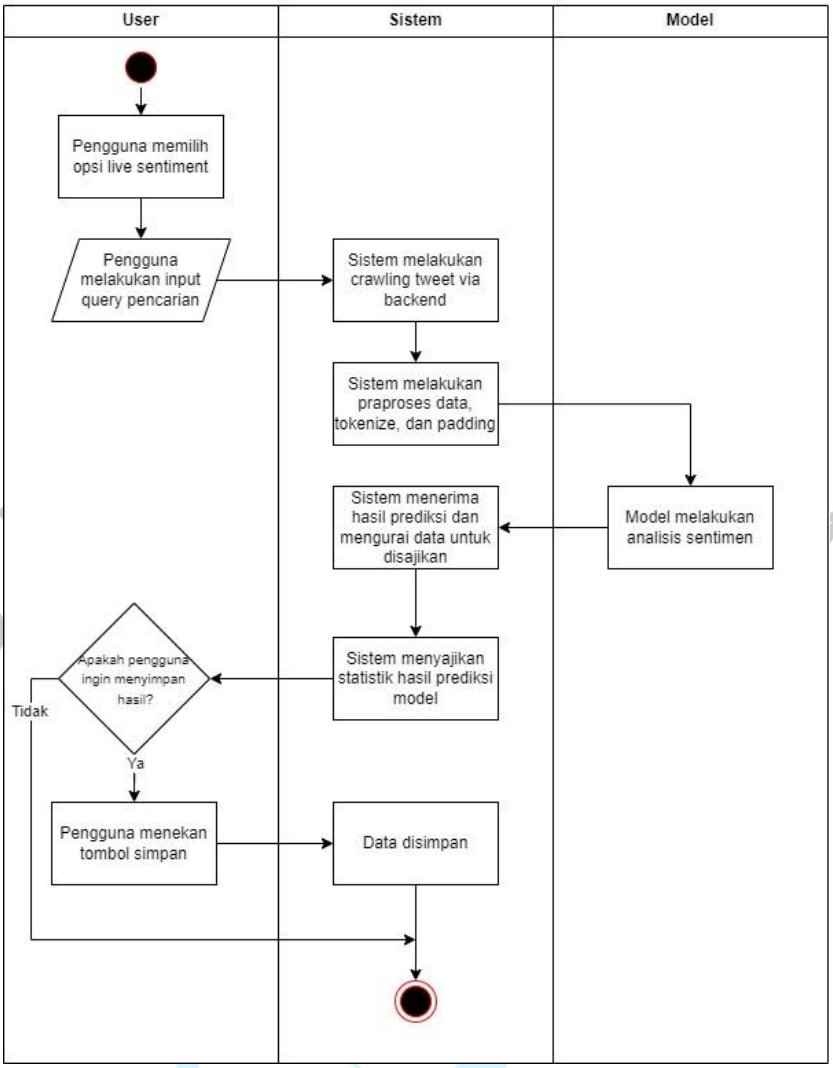
4.3.2 Activity Diagram



Gambar 4. 10 *Activity Diagram* fitur Unggah File

Gambar 4.10 merupakan rancangan diagram aktifitas yang menggambarkan alur kerja sistem. Diagram aktifitas memiliki 3 entitas yang saling terhubung yaitu *user*, sistem, dan model NLP. Aktifitas dimulai dari *user* yang mengakses aplikasi lalu mengunggah file dalam bentuk *csv* dengan *delimiter* titik koma (;).

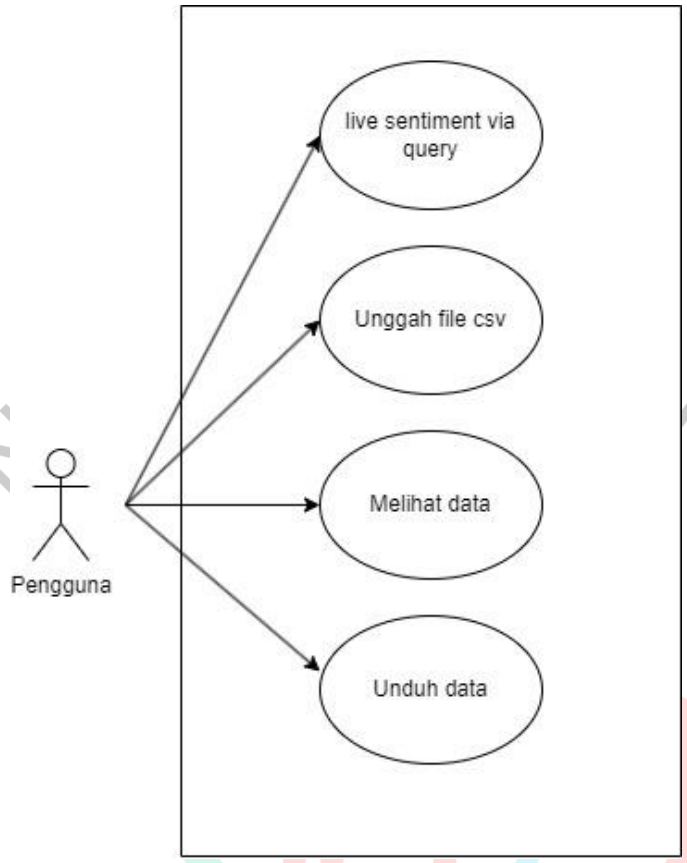
Sistem menerima file masukan dan melakukan validasi format. Setelah validasi, sistem akan membaca isi file dan melakukan praproses data seperti yang dilakukan pada Langkah pengumpulan dataset model *NLP*. Data yang telah dibersihkan kemudian dianalisis oleh model *NLP* untuk mendapatkan hasil. Data disajikan oleh sistem kepada *user* dalam bentuk statistik seperti total sentimen, sentimen terbanyak, gambar diagram lingkaran, dan gambar diagram batang.



Gambar 4. 11 *Activity Diagram* Fitur *Live Sentiment*

Gambar 4.11 merupakan *activity diagram* pada fitur *live sentiment*. *User* melakukan input *query* berupa kata kunci yang ingin dicari dan periode waktu. Sistem akan melakukan *crawling tweet* dan melakukan pembersihan data yang telah terkumpul. Data yang telah dibersihkan akan dimasukkan ke dalam model untuk menentukan polaritas setiap data. Keluaran dari model akan disajikan dalam bentuk statistik sentimen. Statistik tersebut dapat disimpan oleh *user* dalam bentuk *pptx*.

4.3.3 *Use Case Diagram*



Gambar 4. 12 *Use Case Diagram* Aplikasi

Gambar 4.12 menjelaskan bahwa *user* dari aplikasi analisis sentimen memiliki akses ke beberapa fitur. *User* dapat mengakses aplikasi analisis sentimen, kemudian melakukan unggah file dengan format yang telah ditentukan atau melakukan *live sentiment* berdasarkan *query*. *User* juga memiliki akses untuk melihat data hasil pemrosesan sistem dan menyimpannya.

4.3.4 Skenario *Use Case*

Skenario *use case* merupakan beberapa kondisi yang dilakukan oleh pengguna. Adapun beberapa skenario use case dijelaskan pada tabel berikut.

Tabel 4. 3 Skenario *Use Case* Unggah File

Nama Use Case	Unggah File
Aktor	*User*
Deskripsi	*User* melakukan unggah file ke aplikasi

Tindakan	*User* dapat menggunghah file ke aplikasi sesuai format

Tabel 4.3 merupakan table *use case* untuk melakukan unggah file. Aktor yang dapat menggunakan fitur ini adalah seluruh *user* yang mengakses aplikasi. Adapun unggahan file yang dilakukan *user* harus sesuai dengan syarat.

Tabel 4. 4 Use Case *User* Melihat Data

Nama Use Case	Melihat Data
Aktor	*User*
Deskripsi	*User* mampu melihat data yang disajikan oleh sistem
Tindakan	*User* melihat data yang disajikan

Tabel 4.4 merupakan table use case *user* untuk melihat data. Data yang diterima sistem baik dari unggah file maupun *live scraping tweet* diolah dan disajikan dalam bentuk diagram dan kalimat penjelasan.

Tabel 4. 5 Use Case *User* Menyimpan Data

Nama Use Case	Menyimpan Data
Aktor	*User*
Deskripsi	*User* dapat menyimpan data
Tindakan	*User* dapat menyimpan data secara opsional

Tabel 4.5 merupakan *use case* untuk menyimpan hasil pengolahan data oleh sistem. Aktor yang berperan adalah *user* atau user. *User* dapat menyimpan hasil olahan dan penyajian data dengan cara menekan tombol *download*.

Tabel 4. 6 Use Case Live Sentiment

Nama Use Case	Live Sentimen via Query
Aktor	*User*
Deskripsi	*User* dapat melakukan live sentiment
Tindakan	*User* melakukan input *query*, tanggal awal dan tanggal akhir.

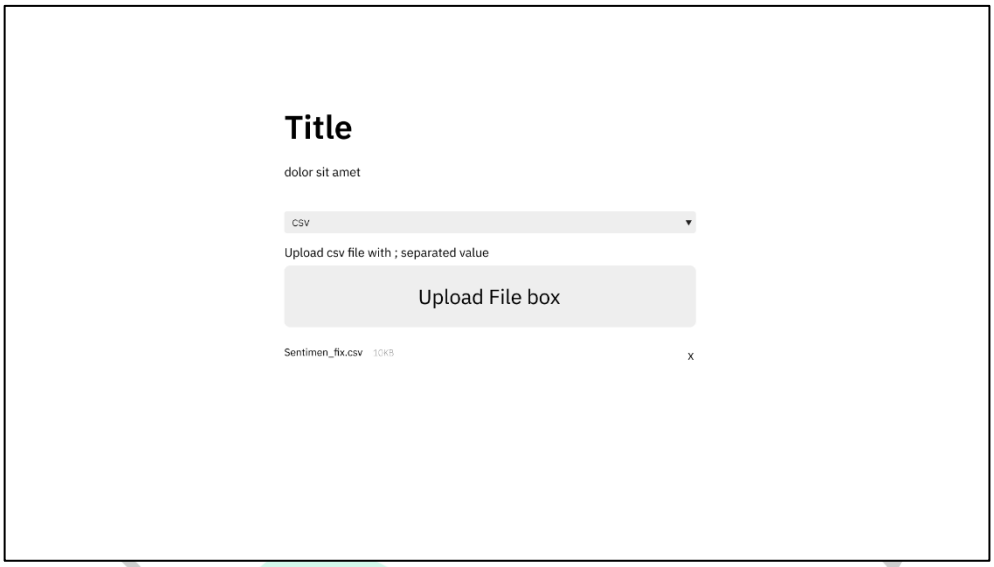
Tabel 4.6 merupakan use case untuk *live sentiment*. Aktor yang berperan pada use case ini adalah *user*. *User* dapat melakukan input *query*, tanggal awal, dan tanggal akhir sebelum menekan tombol *submit*.

4.4 Perancangan Antarmuka

Perancangan antarmuka merupakan proses merancang dan mengembangkan antarmuka *user* yang mudah digunakan dan efektif untuk sistem yang sedang dibangun. Tujuannya adalah memastikan *user* dapat berinteraksi dengan sistem dengan mudah dan dapat memahami informasi yang disajikan dengan jelas. Berikut merupakan perancangan antar muka untuk *user*.

Gambar 4. 13 Rancangan Tampilan Awal Aplikasi

Gambar 4.13 merupakan rancangan tampilkan awal atau *homepage* dari aplikasi. *User* dapat memilih opsi pengolahan data melalui unggah file (*csv*) atau fitur *live sentiment* via *dropdown menu*.



Gambar 4. 14 Rancangan Tampilan *User* Mengunggah File

Gambar 4.14 merupakan Rancangan tampilan Ketika *user* memilih opsi unggah file. *Upload* box memiliki kemampuan untuk *drag and drop* dengan ukuran file maksimal 5MB. File yang terunggah akan terlihat pada kolom bawah *upload box*.

Gambar 4. 15 Rancangan Tampilan opsi live sentiment

Gambar 4.15 merupakan rancangan tampilan untuk memasukkan *query*, tanggal mulai dan tanggal akhir. Box *query* menggunakan tipe input box, sedangkan *start date* dan *end date* menggunakan *date box* dengan format tanggal *YYYY/MM/DD*. Tombol *submit* digunakan untuk memulai pencarian *tweet* dan memprosesnya.

Gambar 4. 16 Rancangan Tampilan Hasil Pengolahan Sentimen

Gambar 4.16 merupakan rancangan untuk menampilkan hasil pengolahan sentiment. *Text* pertama berisi rincian jumlah sentiment yang diperoleh dan *text* kedua untuk menginformasikan sentiment terbanyak yang diperoleh. Diagram terbagi menjadi dua yaitu diagram batang dan diagram venn. Kedua diagram berfungsi untuk melihat sebaran sentimen.

4.5 Rancangan Pengujian

4.5.1 Rancangan Pengujian *Black Box*

Tabel 4.7 menunjukan rancangan pengujian *black box* yang terdiri dari beberapa skenario dan hasil yang diharapkan. Rancangan pengujian *black box* memiliki tujuan untuk melihat performa sistem dan meminimalisasi *bug* pada tampilan.

Tabel 4. 7 Skenario Pengujian Black Box

No	Skenario pengujian	Hasil yang diharapkan
1	*User* memilih opsi *upload csv* dan *live sentiment*.	*User* dapat memilih dan masuk ke dalam halaman *live sentiment* atau *upload csv*.
2	*User* mengisi *form* berupa *keyword* dan *tanggal* pada opsi *live sentiment*.	Sistem menampilkan isi *form* yang diisi oleh *user*.
3	*User* menekan tombol *submit* untuk memulai proses analisis sentiment pada opsi *live sentiment* dan *upload csv*.	Sistem menampilkan animasi *loading* sebagai tanda bahwa sistem sedang memproses *input* dari *user*.
4	*User* menekan tombol *browse file* untuk memilih *file* pada opsi *upload csv*.	Sistem menampilkan jendela *file explorer* untuk memilih *file*.
5	*User* melakukan *upload* file *csv* melebihi batas yaitu lebih dari 5MB.	Sistem menampilkan notifikasi bahwa file melebihi batas.
6	*User* menekan tombol *download as pptx* pada opsi *live sentiment* dan *upload csv*.	*Browser* menyimpan *file pptx* yang diberikan oleh sistem.

4.5.2 Rancangan Pengujian *White box*

Tabel 4.8 menunjukan rancangan pengujian white box yang terdiri dari beberapa skenario dan hasil yang diharapkan. Rancangan pengujian *white box* memiliki tujuan untuk melihat kesesuaian logika pemrograman berdasarkan *input* dan hasil yang diharapkan berupa *output*.

Tabel 4. 8 Rancangan Pengujian *White Box*

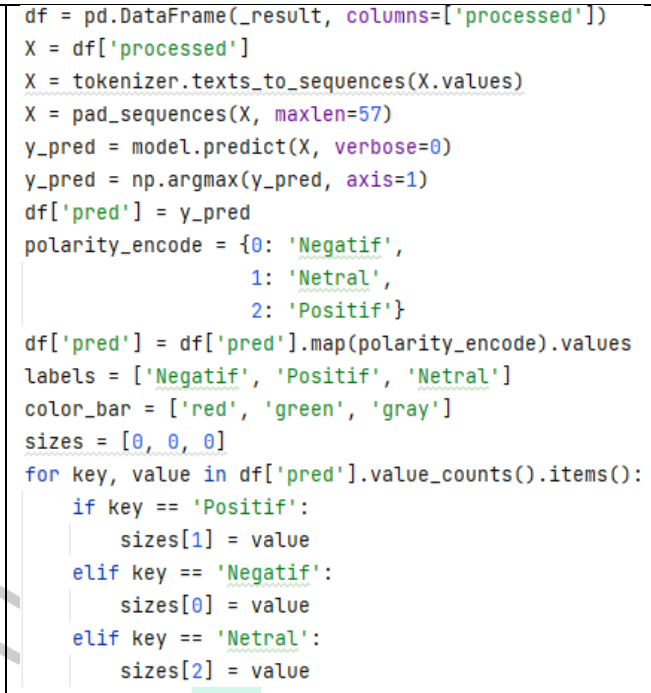
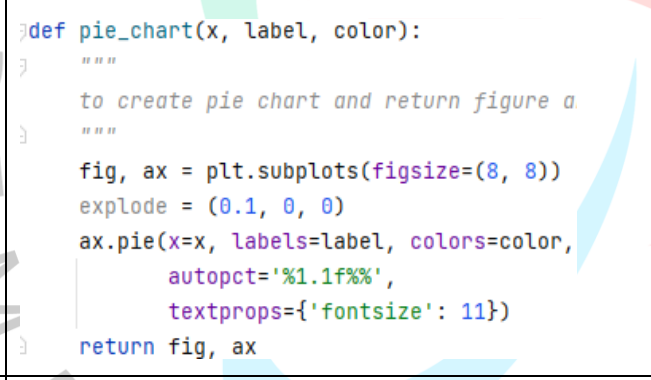
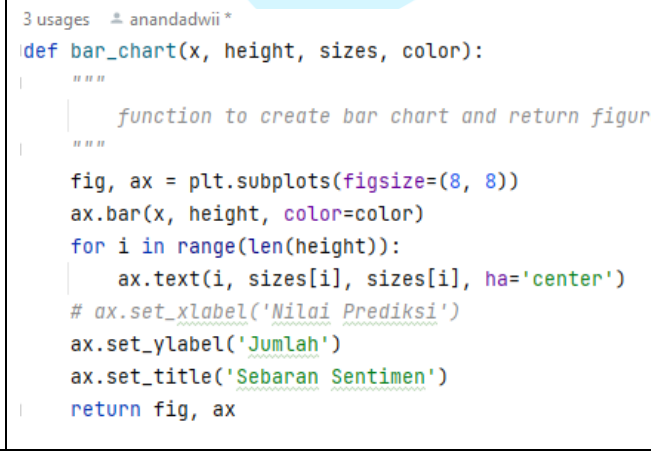
No	Skenario Pengujian	Hasil yang diharapkan
1	1 usage anandadwii * @st.cache_resource def tokenizer_loader(): """fungsi untuk load tokenizer ke dalam cache website" with open('manual_tokenizer.pickle', 'rb') as handler: tokenizer = pickle.load(handler) return tokenizer	Aplikasi memuat model dan *tokenizer* ke dalam *cache browser*.

	1 usage anandadwii *
	`@st.cache_resource`
	`def model_loader():`
	`  """ fungsi untuk load model ke dalam streamlit`
	`  return load_model('balanced_manual_model.h5')`	

2	`def remove_emoji():`
	`  """menghilangkan emoji"""`
	`  emojis = sorted(emoji.EMOJI_DATA,`
	`  key=len, reverse=True)`
	`  pattern = u'(' + u'|'.join(`
	`  re.escape(u) for u in emojis) + u')'`
	`  return re.compile(pattern)`
		Melakukan eliminasi emoji pada kalimat.

3	`def cleaning_text(value):`
	`  result = value.lower().strip()`
	`  # result = remove_three_same_char(result)`
	`  result = ' '.join(result.split())`
	`  result = re.sub(r'(@|https?)\S+|#[A-Za-z0-9_]+',`
	`  '', result).replace("&", "dan")`
	`  result = re.sub(r'RT[\s]+', '', result)`
	`  result = begone_emoji.sub(repl='', string=result)`
	`  result = re.sub(r'[0-9]+', '', result)`
	`  result = result.replace('\n', ' ')`
	`  result = result.translate(`
	`  str.maketrans('', '', string.punctuation))`
	`  return result`
		Menyamakan tipe huruf menjadi kecil, menghapus *hyperlinks*, *mention*, *hastag*, *emoji*, dan angka.

4	`def filtering_stopwords(value):`
	`  """ menghilangkan stopword pada kalimat`
	`  list_stopwords = set(`
	`  stopwords.words('indonesian'))`
	`  list_stopwords.remove("tidak")`
	`  filtered = []`
	`  for text in value:`
	`  if text not in list_stopwords:`
	`  filtered.append(text)`
	`  return filtered`
		Menghilangkan *stopword* pada kalimat.

5		*Word Embedding input*, melakukan prediksi, dan *encode output* menjadi kata.
6		Menampilkan diagram venn.
7		Menampilkan diagram batang.

8	`st.download_button(`
	`  label="Download as pptx",`
	`  data=create_ppt(dict_data, paragraph),`
	`  file_name=f"Report {str(selectbox)}.pptx")`
		Menampilkan tombol *download*.

UNIVERSITAS PEMBANGUNAN JAYA