BAB IV
PERANCANGAN

Pada bab ini akan disajikan penelitian yang telah dilaksanakan oleh peneliti sebelumnya, spesifikasi untuk kebutuhan sistem baru, perancangan sistem dan antarmuka, serta langkah-langkah perancangan sistem secara rinci.

4.1 Analisis Sistem Terdahulu

Analisis mendetail terhadap sistem terdahulu dilakukan untuk mengidentifikasi kekurangan, kelemahan, serta menghindari terjadinya duplikasi dengan penelitian sebelumnya. Peneliti secara aktif mengobservasi sistem yang telah ada untuk memahami kebutuhan dalam mengembangkan sistem baru. Analisis yang dilakukan terhadap sistem sebelumnya dapat menjadi bahan pertimbangan dalam menambahkan fitur-fitur tertentu yang dapat menjadi peningkatan pada sistem yang akan dibuat.

Pada penelitian yang dilakukan oleh Nugraha dan Arrofiq, aplikasi website sederhana yang dibuat berfungsi untuk melakukan klasifikasi teks berupa analisis sentimen terhadap pesan investasi. Metode pengumpulan data yang digunakan sebagai data pelatihan diperoleh menggunakan teknik *scrapping* pada grup Telegram. Tahapan yang dilakukan dalam membangun model analisis sentimen tersebut meliputi *case folding*, *filtering*, tokenisasi, *stemming*, dan *stopwords removal*. Aplikasi klasifikasi teks yang dibangun hanya memiliki *form input* dan hasil klasifikasi yang terbagi menjadi dua kelas, yaitu legal dan ilegal.

4.2 Spesifikasi Kebutuhan Sistem Baru

Perumusan spesifikasi kebutuhan sistem baru dilakukan setelah tahapan analisis mendalam terhadap sistem terdahulu agar dapat menemukan spesifikasi pada sistem lama. Diperlukan suatu sistem baru untuk merancang dan membuat sistem klasifikasi genre film, kebutuhan tersebut dibagi menjadi dua, yaitu spesifikasi perangkat lunak (*software*) dan perangkat keras (*hardware*). Berikut adalah penjelasan masing-masing bagian tersebut.

4.2.1 Spesifikasi Perangkat Lunak (*Software*)

Pada penelitian ini, penggunaan perangkat lunak dibutuhkan sebagai media untuk membantu proses perancangan desain sistem hingga penulisan kode program. Perangkat lunak yang digunakan berupa sistem operasi dan *development environment*. Adapun rincian spesifikasi perangkat lunak tersebut, sebagai berikut.

Tabel 4. 1 Spesifikasi Kebutuhan Perangkat Lunak

No.	Perangkat Lunak	Keterangan
1.	Windows 10 (32 bit)	Batas minimal sistem operasi yang harus dipenuhi oleh pengguna
2.	*Visual Studio Code*	*Code editor* yang digunakan untuk mengembangkan aplikasi utama dalam penelitian ini
3.	*Google Colaboratory*	Platform berbasis *cloud* yang digunakan untuk pengembangan dan pelatihan model
4.	*Kaggle*	Platform yang digunakan untuk memperoleh dataset

Tabel 4.1 menunjukkan daftar perangkat lunak yang digunakan oleh peneliti untuk merancang dan mengembangkan sistem klasifikasi genre film. Berikut adalah penjelasan mengenai fungsi setiap perangkat lunak secara rinci sebagai berikut.

a) *Windows 10*

Windows 10 merupakan sistem operasi yang digunakan sebagai wadah yang memastikan kompatibilitas dan kelancaran penggunaan setiap perangkat lunak. Sistem operasi memungkinkan perangkat lunak seperti *Kaggle*, *Google Colaboratory*, *Jupyter Notebook*, *Visual Studio Code*, dan alat-alat lainnya dapat dijalankan untuk pengembangan dan pembuatan model klasifikasi genre film.

b) *Visual Studio Code*

Visual Studio Code atau disingkat *VSCode* adalah editor kode sumber yang ringan serta dilengkapi berbagai macam fitur yang dapat memudahkan proses pengembangan dan pembuatan perangkat lunak. Peneliti menggunakan kode editor ini untuk mengintegrasikan model klasifikasi genre film dengan

perangkat lunak yang terdiri dari logika program untuk dapat memproses masukkan dari pengguna, sehingga mengeluarkan luaran yang sesuai.

c) *Google Colaboratory*

Google Colaboratory digunakan sebagai platform untuk menulis dan menjalankan kode Python, khususnya dalam pengembangan model *Natural Language Processing* untuk sistem klasifikasi genre film pada penelitian ini.

d) *Kaggle*

Kaggle merupakan *platform online* yang dirancang khusus untuk mengembangkan model *machine learning*. Salah satu keunggulan platform ini adalah menyediakan berbagai macam dataset yang mencakup beragam domain *machine learning*, salah satunya yaitu dataset film yang digunakan pada penelitian ini untuk melatih model.

4.2.2 Spesifikasi Perangkat Keras (*Hardware*)

Perangkat keras berperan sebagai basis fisik untuk menjalankan perangkat lunak dalam proses perancangan sistem. Dalam penelitian ini, perangkat keras yang menjadi faktor pendukung meliputi *processor*, *storage*, dan *memory*. Tabel berikut merinci spesifikasi perangkat keras yang digunakan oleh peneliti, termasuk spesifikasi minimum yang diperlukan.

Tabel 4. 2 Spesifikasi Perangkat Keras Minimum

No.	Perangkat Keras	Keterangan
1.	*Processor*	1 Ghz atau lebih
2.	*Storage*	128 GB Hard Disk/SSD
3.	*Memory*	4 GB RAM

Tabel berikut menampilkan spesifikasi perangkat keras yang digunakan peneliti dalam merancang dan membangun sistem klasifikasi genre film.

Tabel 4. 3 Spesifikasi Perangkat Keras Peneliti

No.	Perangkat Keras	Keterangan
1.	*Processor*	2.6 Ghz
2.	*Storage*	512 GB SSD
3.	*Memory*	8 GB RAM

a) *Processor*

Processor berperan sebagai inti atau pusat pengendali dalam sebuah sistem dan terletak pada perangkat keras. Ini berfungsi untuk dapat melakukan berbagai operasi perhitungan dan pengolahan data yang diperlukan, termasuk mengeskusi instruksi-instruksi yang terdapat dalam perangkat lunak dan telah terintegrasi dengan model yang menggunakan algoritma BiLSTM.

b) *Storage*

Storage atau penyimpanan digunakan sebagai tempat untuk menyimpan data yang diperlukan oleh sistem klasifikasi genre film, seperti model BiLSTM, dataset pelatihan, serta kode sumber perangkat lunak yang telah dibuat sebelumnya.

c) *Memory*

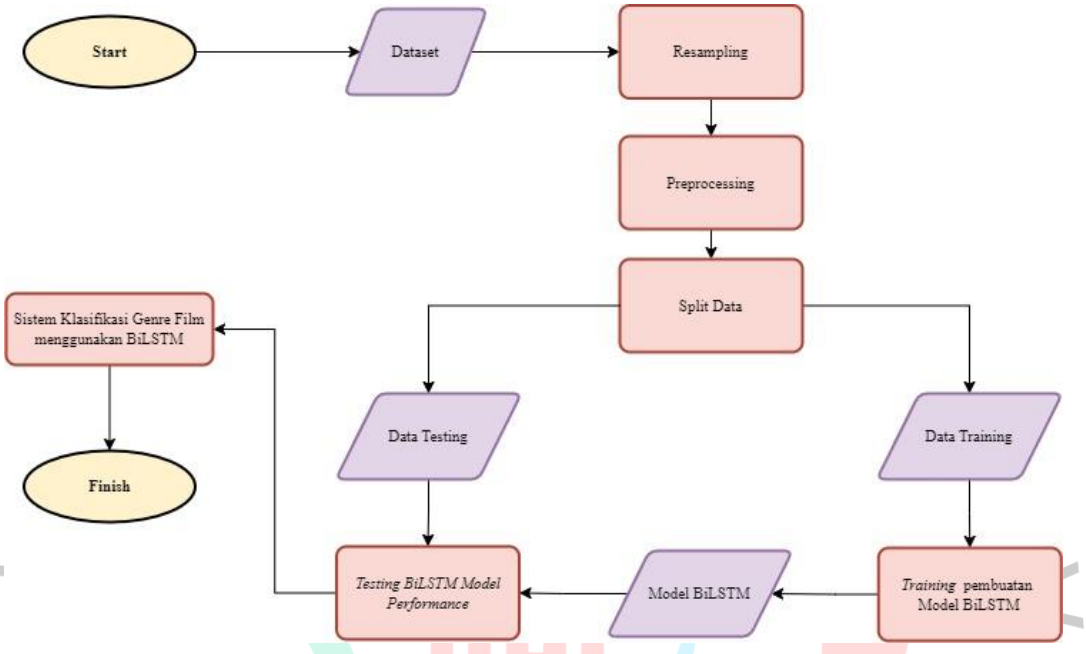
Memory memiliki peran untuk menyimpan data dan program sementara yang sedang berjalan saat ini. Perangkat keras ini juga digunakan sebagai penyimpanan model BiLSTM selama perangkat lunak sedang digunakan secara *realtime*.

4.3 Identifikasi *Keyword*

Tahap identifikasi *keyword* atau kata kunci dilakukan untuk mendukung model dalam melakukan klasifikasi genre terhadap sinopsis. Kata kunci diidentifikasi pada setiap sinopsis di setiap genre. Pengidentifikasian kata kunci dilakukan dengan cara manual dengan menemukan kata yang unik dan terkait dengan suatu genre. Kata kunci tersebut tidak boleh terkandung dalam lebih dari satu genre, agar menghindari adanya kekeliruan dalam proses klasifikasi.

4.4 Pembuatan Model

Proses perancangan sistem klasifikasi genre film, yang bertujuan untuk menentukan kategori film berdasarkan sinopsisnya, melibatkan beberapa tahapan yang diilustrasikan pada gambar di bawah ini.



Gambar 4. 1 Tahap Pembuatan Model

Gambar 4.1 merupakan tahapan proses pembuatan model pada penelitian ini untuk membuat sistem klasifikasi genre berdasarkan sinopsis menggunakan Algoritma BiLSTM. Tahapan tersebut terdiri dari mengumpulkan dataset, melakukan *training* dan *testing*, hingga sistem dapat mengklasifikasi genre film. Adapun penjelasan mengenai setiap tahapan di atas, sebagai berikut.

4.4.1 Dataset

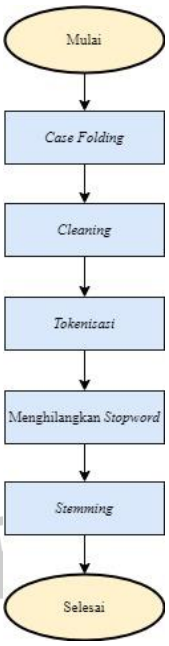
Data yang digunakan pada penelitian ini adalah data sekunder yang diperoleh dari situs *kaggle.com* dengan judul dataset yaitu "***Genre Classification Dataset IMDB***". *Dataset* ini memuat informasi seperti judul, genre, dan sinopsis setiap film yang terdaftar di *Internet Movie Database* (IMDB) hingga tahun 2020. *Dataset* ini memiliki jumlah data sebanyak 54.214 data film dan memiliki empat atribut yaitu id, *title*, genre, serta *description*. Data dalam *dataset* tersebut memiliki format *comma separated values* (CSV) yang memungkinkan untuk diolah dan dianalisis menggunakan *machine learning*.

Pada tahap awal, dilakukan proses penyaringan *dataset* untuk menyeleksi data film yang memiliki genre *action*, *adventure*, *animation*, *comedy*, *crime*, *drama*, *horror*, *romance*, *science-fiction*, dan *thriller*. Dari total 54.214 data film, proses penyaringan menghasilkan 29.267 data film yang memenuhi kriteria. Analisis genre film yang tersisa menunjukkan distribusi yang tidak seimbang, dengan genre mendominasi yaitu *drama*(46.5%), diikuti *comedy*(25.4%), *horror*(7.5%), *thriller*(5.4%), *action*(4.5%), *adventure*(2.6%), *romance*(2.3%), *science-fiction*(2.2%), *crime*(1.7%), dan *animation*(1.7%). Ketidakseimbangan data ini dikhawatirkan dapat berakibat pada hasil klasifikasi yang tidak optimal. Maka dari itu, dilakukan teknik *resampling* dengan menggunakan *library* imblearn di Python.

Metode *resampling* yang digunakan adalah *upsampling* dengan memanfaatkan teknik RandomOverSampler dari library imblearn. Teknik ini bekerja dengan memilih data film secara acak dari dataset dan menggandakannya hingga jumlahnya sama dengan frekuensi data terbanyak, yaitu genre drama. Hal ini bertujuan untuk menyeimbangkan distribusi data dan meningkatkan performa model klasifikasi dalam menangani semua genre film dengan lebih baik.

4.4.2 *Pre-processing*

Preprocessing merupakan langkah-langkah yang digunakan untuk menyeragamkan format dan membersihkan data dari segala nilai yang tidak memberikan pengaruh besar terhadap proses klasifikasi. Dalam klasifikasi teks, tahap ini memiliki peran penting agar model dapat mengolah data teks yang bersifat sekuensial atau berurutan. Tahapan *preprocessing* terdiri dari serangkaian proses yang memiliki tugasnya masing-masing, tahapannya mencakup *case folding*, *cleaning*, tokenisasi, menghilangkan *stopword*, dan *stemming*. Upaya ini bertujuan untuk mengurangi *noise* dan meningkatkan kualitas model saat tahap pelatihan. Berikut adalah diagram alir yang menunjukkan tahapan *preprocessing*.



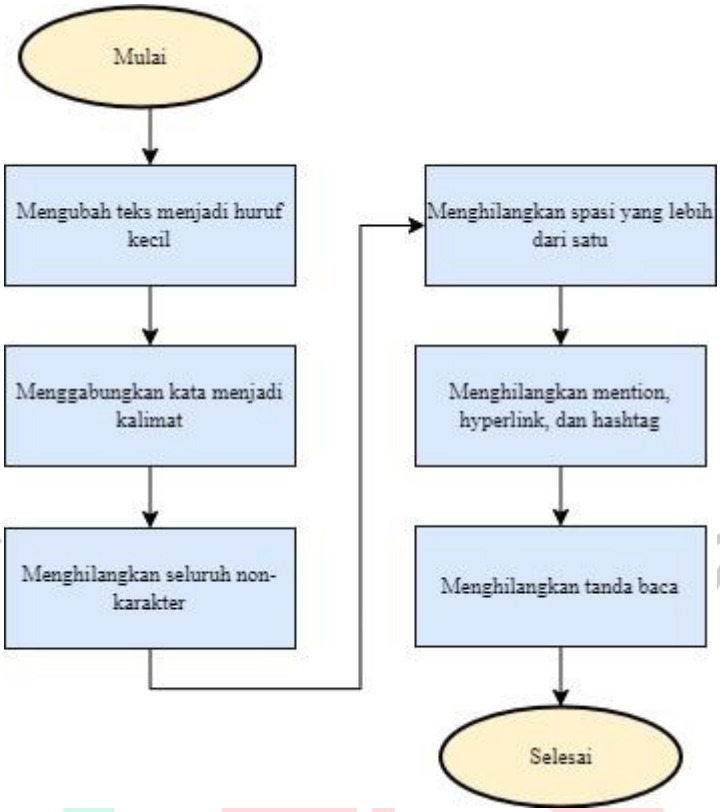
Gambar 4. 2 Flowchart *Preprocessing* Data

1) *Case Folding*

Case folding adalah tahap awal yang dilakukan pada *preprocessing* yang mentransformasi seluruh huruf besar pada kalimat menjadi huruf kecil atau *lowercase*. Tahap ini dilakukan dengan tujuan agar data memiliki format yang konsisten. Dengan begitu, data teks yang digunakan sebagai masukkan dapat diolah lebih lanjut melewati proses selanjutnya, sehingga menghasilkan analisis yang akurat dan objektif. Proses ini juga membantu dalam mengurangi variasi yang tidak perlu dalam data teks.

2) *Cleaning*

Cleaning adalah tahap membersihkan data dari elemen-elemen yang tidak diperlukan atau dapat berdampak negatif terhadap pelatihan model. Elemen-elemen tersebut mencakup *hyperlinks* atau *url*, *hashtag*, angka, simbol retweet, emoji, dan tanda baca. Tujuan dari tahap ini adalah untuk mengurangi *noise* pada data, yaitu elemen-elemen yang tidak relevan yang dapat menyamarkan pola dan makna penting dalam teks.



Gambar 4. 3 Flowchart proses *Cleaning*

Gambar 4.3 menunjukkan tahapan proses *cleaning* atau pembersihan data teks. Proses ini diawali dengan melakukan perubahan pada setiap kata teks *lowercase* atau menjadi huruf kecil. Selanjutnya, kata-kata tersebut digabungkan menjadi kalimat. Dalam kalimat tersebut, segala elemen yang menyebabkan *noise* seperti simbol atau non-karakter akan dihilangkan. Selain itu, *mention*, *hyperlink* atau *url*, dan *hashtag* juga dihilangkan, serta spasi yang berlebihan disesuaikan menjadi satu spasi. Langkah terakhir adalah menghilangkan tanda baca yang tidak diperlukan.

3) Tokenisasi

Tokenisasi adalah tahap mengubah teks yang terdiri dari kumpulan kata menjadi token yang terdiri dari satu kata. Kata-kata yang telah dijadikan token memiliki indeks yang berbeda-beda sehingga dapat digunakan sebagai kamus bahasa atau kumpulan kosakata.

4) Menghilangkan *Stopword*

Stopword adalah kata-kata umum yang sering ditemukan dalam teks, tetapi kurang memberikan informasi penting mengenai konteks teks tersebut. Contohnya adalah kata-kata yang termasuk dalam daftar *stopwords*, mencakup konjungsi atau kata penghubung, *pronouns* atau kata ganti, dan preposisi atau kata depan serta kata-kata yang sering ditemukan dan dianggap tidak relevan.

5) *Stemming*

Stemming adalah tahap dimana setiap kata dicari *root* atau kata dasarnya sehingga menghasilkan luaran berupa kata dasar tanpa imbuhan. Tujuan proses *stemming* adalah untuk mengurangi kata-kata yang memiliki bentuk berbeda tetapi dengan makna yang sama. Dengan begitu, kompleksitas dan ukuran data tidak terlalu besar, sehingga berdampak pada penghematan waktu dan memori. Pada penelitian ini, proses *stemming* diimplementasikan menggunakan bantuan salah satu *library* Python yaitu *PorterStemmer* dari NLTK.

4.4.3 Split Data

Split data adalah tahap pemisahan dataset yang telah melewati proses *preprocessing* beberapa bagian atau *subset*. Jumlah data setelah dilakukan proses *resampling* terkumpul sebanyak 136.130 data. Data tersebut terbagi menjadi 13.613 data karena terdapat 10 genre yang akan menjadi kategori klasifikasi. Secara umum, pembagian data diterapkan sehingga menghasilkan dua *set* data, yaitu data pelatihan (*training set)* dan data pengujian (*testing set*). Penjelasan kedua bagian tersebut dijelaskan sebagai berikut:

a. Data *Training* / *Training Set*

Data *training* akan digunakan oleh model sebagai bahan pelatihan agar dapat mengenali pola untuk dapat mengklasifikasi genre film berdasarkan sinopsisnya. Dalam penelitian ini, 80% dari total populasi data, yaitu sebanyak 108.904 data, digunakan sebagai data pelatihan. Data tersebut terbagi secara merata ke dalam 10 genre, dengan masing-masing genre memiliki 10.890 data.

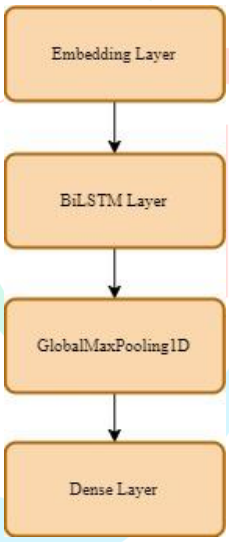
b. Data *Testing* / *Testing Set*

Data *testing* akan digunakan untuk menguji performa model yang telah melewati proses pelatihan, dengan data yang belum pernah dilihat sebelumnya.

Dalam penelitian ini, 20% dari total populasi data, yaitu sebanyak 27.226 data, digunakan sebagai data pengujian. Data tersebut terbagi secara merata ke dalam 10 genre, dengan masing-masing genre memiliki 2.722 data.

4.4.4 Rancangan Lapisan Model

Pembuatan model dilakukan dengan mendefinisikan setiap *layer* atau lapisan yang digunakan untuk dapat membangun model sistem klasifikasi genre film. Pembuatan model pada penelitian ini akan menggunakan lapisan BiLSTM sebagai salah satu lapisan utama yang akan menjadi fondasi untuk model dalam melakukan pembelajaran. Lapisan BiLSTM membantu model dalam menangkap informasi secara dua arah dari sinopsis film, sehingga dapat memahami konteks dengan lebih baik untuk klasifikasi genre.



Gambar 4. 4 Desain Layer Model Klasifikasi menggunakan BiLSTM

Gambar 4.4 menunjukkan lapisan yang digunakan peneliti dalam membuat model sistem klasifikasi genre film. Adapun penjelasan mengenai setiap lapisan tersebut, dijabarkan sebagai berikut:

a) *Embedded Layer*

Embedding layer adalah lapisan yang mengubah *input* data yang telah melewati proses tokenisasi menjadi representasi vektor dengan dimensi yang lebih rendah. Lapisan ini berfungsi untuk memetakan setiap kata (atau token) dalam teks menjadi vektor numerik yang memiliki dimensi bersifat statis atau

tetap. Tujuannya adalah untuk menangkap hubungan antar kata dan mengurangi dimensi input sehingga memudahkan pemrosesan data teks oleh model.

b) *BiLSTM Layer*

Lapisan BiLSTM akan mengklasifikasi genre film berdasarkan *input* yang diterima dari lapisan *embedded*. Lapisan ini tersusun dari dua unit LSTM, yang masing-masing memiliki tiga jenis gerbang yaitu *forget gate*, *input gate*, dan *output gate*. Unit-unit ini dibagi menjadi dua bagian, bagian pertama disebut keadaan maju, yang memproses informasi dalam urutan kronologis dari awal hingga akhir, sementara bagian kedua disebut keadaan mundur, yang memproses informasi dalam urutan terbalik dari akhir hingga awal. Sebanyak 32 unit digunakan sebagai lapisan BiLSTM. Unit-unit tersebut berfungsi sebagai ukuran atau jumlah hidden state LSTM pada lapisan tersembunyi dalam mengolah informasi.

c) *GlobalMaxPooling1D*

Lapisan Global Max Pooling 1D berperan untuk mengambil nilai maksimum dari setiap fitur dalam satu dimensi. Lapisan ini berfungsi untuk mengurangi dimensi data dengan mengambil nilai maksimum dari fitur tertentu, sehingga menghasilkan vektor dengan dimensi yang lebih rendah. Ini akan membantu model dalam mengurangi kompleksitas dan mencegah terjadinya *overfitting*. Tujuannya adalah untuk mengekstrak fitur yan paling menonjol dalam data dan engurangi ukurannnya tanpa kehilangan informasi penting.

d) *Dense Layer*

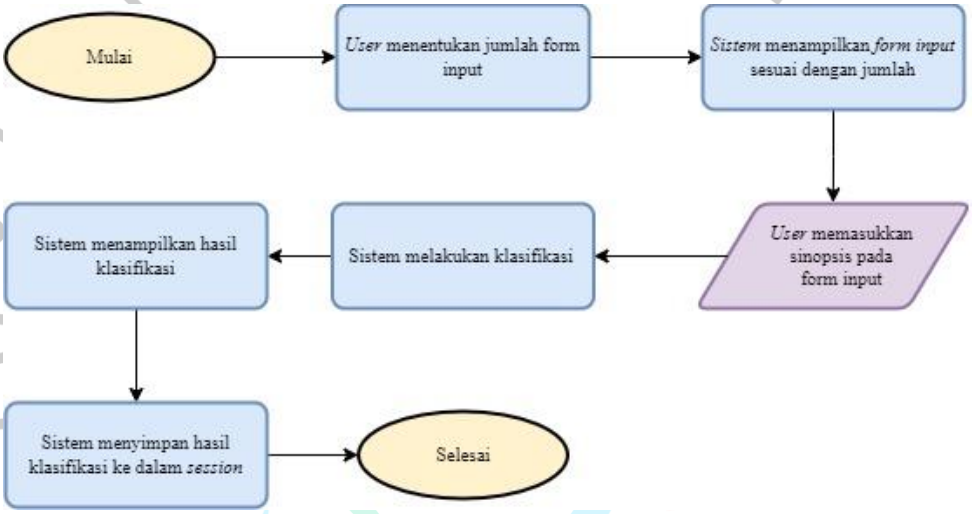
Dense Layer atau biasa disebut dengan fully connected layer yang membuat setiap unit (*neuron*) dalam lapisan tersebut terhubung dengan setiap unit di lapisan sebelumnya. Lapisan ini berfungsi untuk melakukan kombinasi linier dari input dan menerapkan fungsi aktivasi *softmax* untuk menghasilkan *output* yang digunakan pada tugas klasifikasi. Tujuannya adalah untuk melakukan transformasi non-linear pada data yang diproses serta untuk menggabungkan informasi yang telah dipelajari oleh lapisan-lapisan

sebelumnya dalam model untuk menghasilkan prediksi akhir, dalam hal ini adalah genre.

4.5 Perancangan Sistem

Perancangan sistem merupakan tahap dimana peneliti menyusun struktur dan komponen utama sistem klasifikasi genre film menggunakan pemodelan *Unified Modelling Language* (UML). Pemodelan tersebut mencakup *flowchart*, *use case diagram* , dan *activity diagram*.

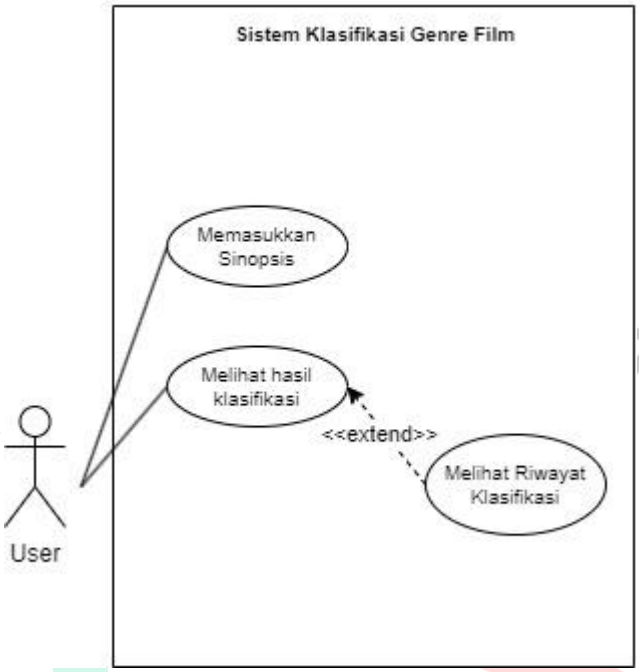
4.5.1 *Flowchart* Proses Bisnis Aplikasi



Gambar 4. 5 Flowchart Penggunaan Aplikasi

Gambar 4.5 merupakan alur sistem klasifikasi genre berdasarkan synopsis yang dibuat oleh peneliti. Tahap yang dilakukan *user* setelah mengakses halaman awal adalah menentukan banyaknya *form input* sebagai tempat untuk memasukkan sinopsis film yang ingin diklasifikasi. Setelah itu, sistem akan menampilkan *form input* sesuai dengan jumlah yang telah ditentukan, lalu *user* dapat memasukkan sinopsis film pada setiap *form input*. Tahap selanjutnya adalah sistem melakukan klasifikasi hingga memunculkan hasil klasifikasi sinopsis film tersebut dan menyimpan hasil klasifikasi ke dalam *session* untuk digunakan sebagai riwayat.

4.5.2 *Use Case Diagram*



Gambar 4. 6 Use Case Diagram Sistem Klasifikasi Genre Film

Gambar 4.6 menunjukkan rancangan *use case* diagram untuk sistem klasifikasi genre film yang dikembangkan oleh peneliti. Pada diagram tersebut *user* berperan sebagai aktor yang dapat memasukkan teks sinopsis, melihat hasil klasifikasi, dan melihat riwayat klasifikasi. Adapun penjelasan detail mengenai setiap *use case* dijabarkan pada tabel scenario di bawah ini.

Tabel 4. 4 Skenario Use Case Memasukkan Sinopsis

Nama Use Case	Memasukkan Sinopsis
Aktor	*User*
Deskripsi	*User* memasukkan teks sinopsis ke aplikasi.
Tahapan	1. *User* masuk ke tampilan awal.
	2. *User* menentukan jumlah *form input.*
	3. *User* memasukkan sejumlah teks sinopsis sesuai jumlah *form input.*

Tabel 4.4 menunjukkan tahapan *use case* ketika user memasukkan sinopsis ke dalam aplikasi. *User* dapat menentukan jumlah *input* berdasarkan jumlah sinopsis yang ingin diklasifikasi.

Tabel 4. 5 Skenario Use Case Melihat Hasil Klasifikasi

Nama Use Case	Melihat Hasil Klasifikasi
Aktor	*User*
Deskripsi	*User* dapat melihat hasil yang disajikan oleh sistem.
Tahapan	1. *User* memasukkan teks sinopsis ke dalam *form input.*
	2. User menekan tombol *submit.*
	3. Sistem melakukan klasifikasi pada sinopsis.
	4. Sistem menampilkan daftar sinopsis yang telah diklasifikasikan.
	5. User melihat hasil klasifikasi sinopsis.

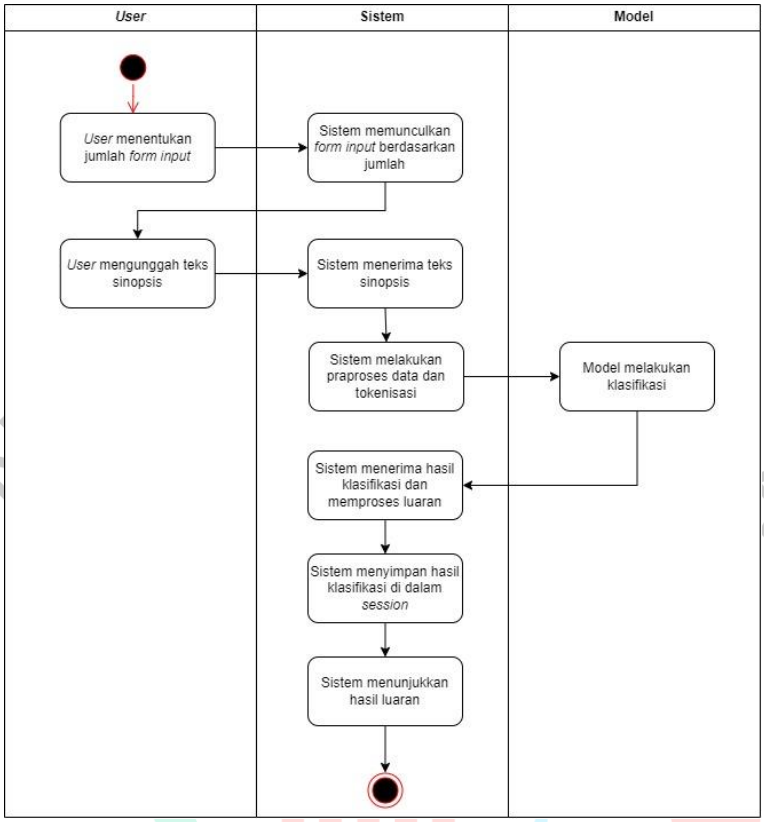
Tabel 4.5 menunjukkan scenario *use case* ketika *user* melihat hasil klasifikasi yang dihasilkan oleh aplikasi. *User* dapat melihat hasil klasifikasi berdasarkan banyaknya jumlah sinopsis yang dimasukkan ke dalam *form input*.

Tabel 4. 6 Skenario Use Case Melihat Riwayat Hasil Klasifikasi

Nama Use Case	Melihat Riwayat atau *History* Hasil Klasifikasi
Aktor	*User*
Deskripsi	*User* dapat melihat hasil yang disajikan oleh sistem.
Tahapan	*1.* *User* memasukkan teks sinopsis ke dalam *form input.*
	2. *User* menekan tombol *submit.*
	3. Sistem melakukan klasifikasi pada sinopsis.
	4. Sistem menampilkan daftar sinopsis yang telah diklasifikasikan.
	5. Sistem memasukkan hasil klasifikasi ke dalam riwayat.
	6. *User* melihat riwayat hasil klasifikasi sinopsis sebelumnya.

Tabel 4.6 menunjukkan scenario *use case* ketika *user* melihat riwayat dari hasil klasifikasi yang dihasilkan oleh aplikasi. *User* dapat melihat seluruh riwayat hasil klasifikasi yang pernah dilakukan.

4.5.3 *Activity Diagram*

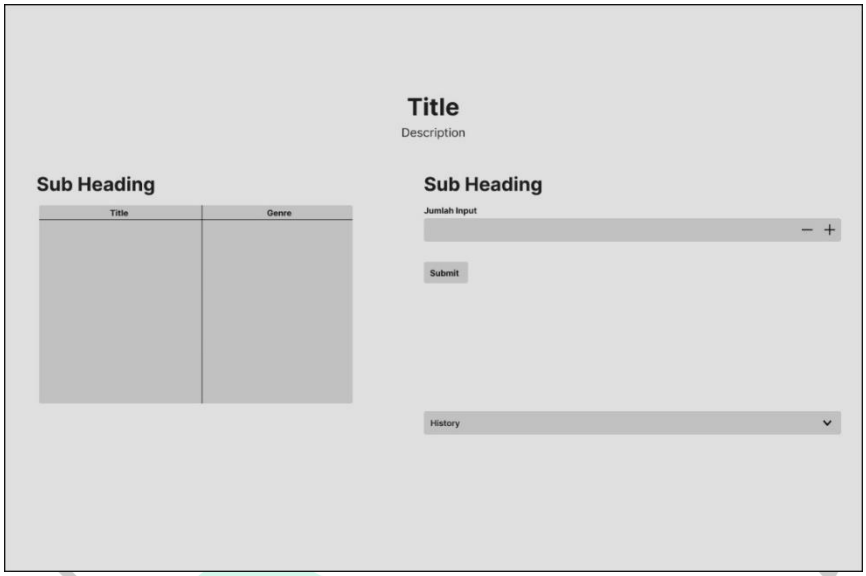


Gambar 4. 7 Activity Diagram Sistem Klasifikasi

Gambar 4.7 memperlihatkan *activity diagram* yang telah dibuat untuk mengetahui alur kerja sistem klasifikasi genre yang terdiri dari tiga entitas atau unit yaitu *user*, sistem, dan model. *User* perlu menentukan jumlah *form input* yang berfungsi sebagai tempat untuk teks sinopsis sebelum dikirimkan ke sistem. Setelah mengirimkan teks sinopsis, sistem akan melakukan praproses data dan tokenisasi untuk dimuat ke dalam model sebagai masukkan. Selanjutnya, model akan melakukan klasifikasi terhadap teks sinopsis tersebut, menyimpannya di dalam *session* untuk riwayat dan akan digunakan sebagai sistem untuk luaran.

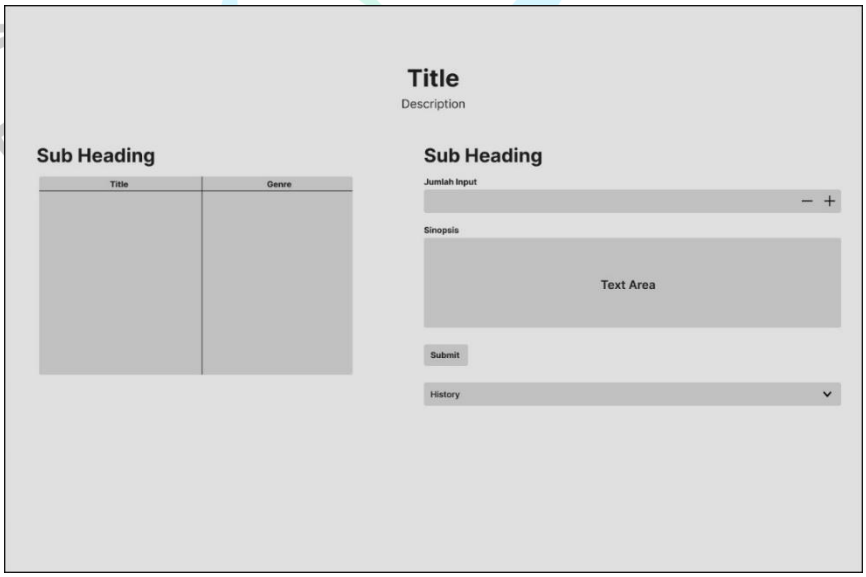
4.6 Perancangan Antarmuka

Tahap ini adalah tahap persiapan dan perancangan tampilan serta fungsi yang akan berinteraksi langsung dengan pengguna. Tahapan ini bertujuan untuk memastikan interaksi pengguna dengan sistem dapat dilakukan dengan mudah dan efektif. Adapun rancangan antarmuka, disajikan sebagai berikut.



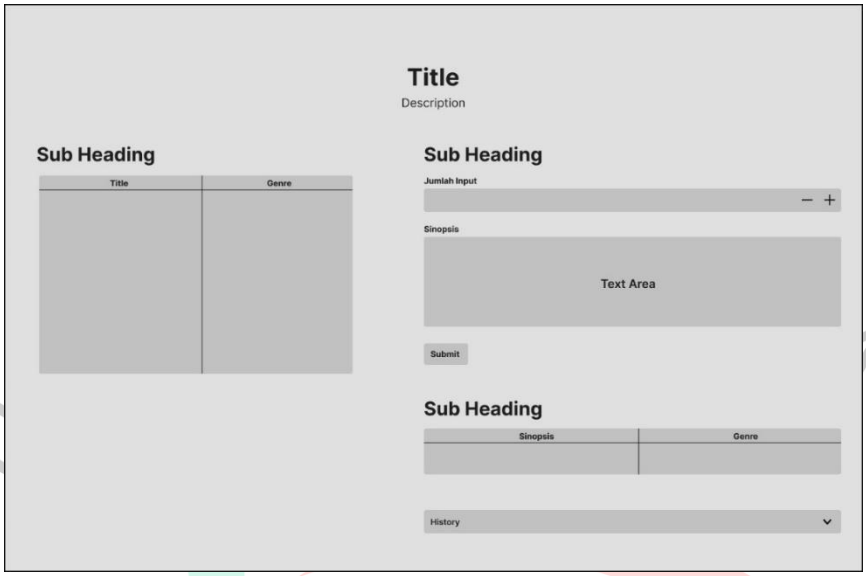
Gambar 4. 8 Rancangan Tampilan Awal Aplikasi

Gambar 4.8 menunjukkan hasil rancangan tampilan awal aplikasi atau sering disebut dengan desain *homepage*. Pada sisi kiri terdapat tabel yang menampung sampel data untuk setiap genre yang akan diidentifikasi. Tabel tersebut memiliki dua kolom yang terdiri dari judul dan genre film. Sementara itu, pada sisi kanan terdapat bagian yang berisi *form input* dimana pengguna perlu menentukan banyaknya sinopsis yang akan dimasukkan dengan menekan tombol *plus* atau tambah.



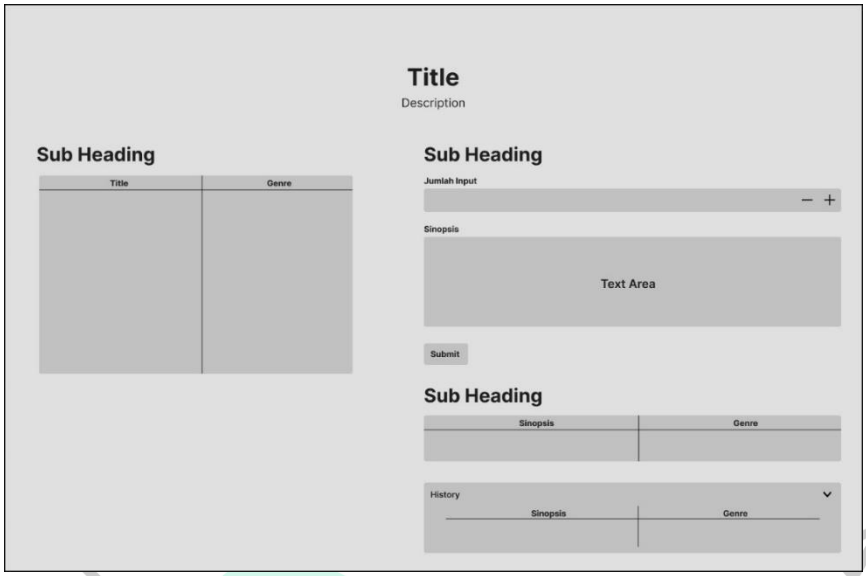
Gambar 4. 9 Rancangan Tampilan Pengguna Menentukan Jumlah Input

Gambar 4.9 menunjukkan rancangan tampilan ketika pengguna menentukan jumlah input, aplikasi secara otomatis akan menampilkan form *input* sesuai jumlah yang telah ditentukan.



Gambar 4. 10 Rancangan Tampilan Hasil Klasifikasi

Gambar 4.10 menunjukkan rancangan tampilan ketika pengguna menekan tombol *submit* dan aplikasi akan secara otomatis melakukan klasifikasi terhadap sinopsis yang telah dimasukkan. Selanjutnya, aplikasi akan menampilkan tabel berisi sinopsis yang telah dimasukkan pengguna dan genre yang merupakan hasil klasifikasi. Pada saat yang sama, aplikasi juga akan menyimpan hasil klasifikasi ke dalam *dropdown history*, sehingga pengguna dapat melihat riwayat sinopsis yang telah diklasifikasi sebelumnya.



Gambar 4. 11 Rancangan Tampilan Pengguna Menekan History

Gambar 4.11 menunjukkan rancangan tampilan ketika pengguna menekan tombol *dropdown history*. Pada bagian *history*, pengguna dapat melihat riwayat hasil klasifikasi yang telah disimpan di dalam *session* pada website.

4.7 Skenario Pengujian

Pada penelitian ini, setelah model klasifikasi genre berhasil dirancang, dibuat, dan diintegrasikan sehingga menjadi sistem yang utuh, dilakukan dua metode pengujian yang telah dijelaskan pada bab tiga. Adapun rincian mengenai setiap skenario dijelaskan sebagai berikut.

4.7.1 Skenario Pengujian Black Box

Pengujian ini bertujuan untuk mengevaluasi kinerja sistem dan meminimalisir *bug* pada tampilan saat waktu eksekusi berlangsung. Berikut adalah rancangan pengujian yang dibuat pada penelitian ini.

Tabel 4. 7 Skenario Pengujian *Black Box*

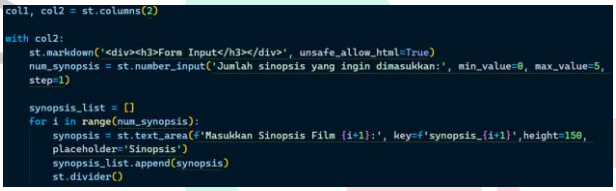
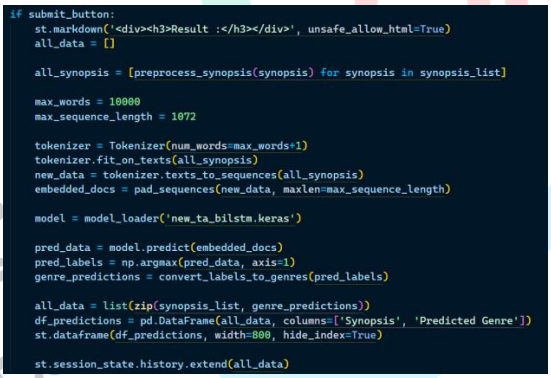
No.	Skenario Pengujian	Hasil yang diharapkan
1.	*User* mendefinisikan jumlah sinopsis yang ingin dimasukkan	Sistem menampilkan *form input* sesuai dengan jumlah yang dimasukkan
2.	*User* mengisi *form* input dengan sinopsis film	Sistem menampilkan isi *form* yang dimasukkan oleh *user*

3.	*User* menekan tombol *submit* untuk memulai proses klasifikasi genre film	Sistem menampilkan tabel yang berisi sinopsis dan hasil klasifikasi
4.	*User* menekan tombol history	Sistem menampilkan riwayat sinopsis yang telah dimasukkan sebelumnya

4.7.2 Skenario Pengujian White Box

Pengujian pada skenario ini dirancang untuk melihat kesesuaian logika pemrograman berdasarkan masukkan dan hasil yang diharapkan berupa keluaran. Adapun skenario pengujian *white box* ditunjukkan pada tabel di bawah ini.

Tabel 4. 8 Skenario Pengujian *White Box*

No.	Skenario Pengujian	Hasil yang diharapkan
1.		Sistem mampu menyajikan *form input* sesuai dengan jumlah yang dimasukkan
2.		Sistem mampu menyajikan hasil klasifikasi
3.		Sistem mampu menyajikan riwayat atau *history* hasil klasifikasi