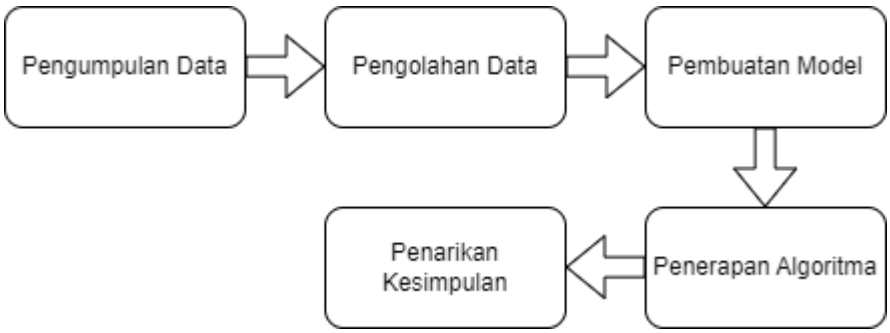
BAB III. METODOLOGI PENELITIAN

3.1. Waktu dan Tempat Penelitian

Penelitian ini akan dilakukan di lingkungan Jurusan Teknologi Informasi Politeknik Negeri Malang dengan rentang waktu pengerjaan dimulai dari bulan Januari tahun 2023 sampai dengan bulan Juli tahun 2023.

3.2. Metode Penelitian

Metode penelitian yang digunakan peneliti adalah metode penelitian yang terdiri dari lima fase sebagai berikut. Pertama adalah pengumpulan data, kedua adalah pengolahan data, ketiga adalah pembuatan model, keempat adalah penerapan algoritma, dan kelima adalah penarikan kesimpulan. Gambaran metode penelitian secara umum dapat dilihat pada gambar 3.1.



Gambar 3. 1. Tahapan Metode Penelitian

3.2.1. Pengumpulan Data

Data yang akan dikumpulkan untuk penelitian ini adalah data yang memenuhi kriteria sebagai berikut:

1. Data hanya berupa teks.
2. Data mayoritas berbahasa indonesia.
3. Data terdiri dari soal dan jawaban.
4. Data soal tidak bersifat membandingkan dua hal atau lebih.
5. Jumlah karakter pada data jawaban tidak lebih 65.500 karakter.

Satu data soal setidaknya harus memiliki lima data jawaban. Contoh dari data soal dan jawaban yang digunakan dapat dilihat pada tabel 3.1.

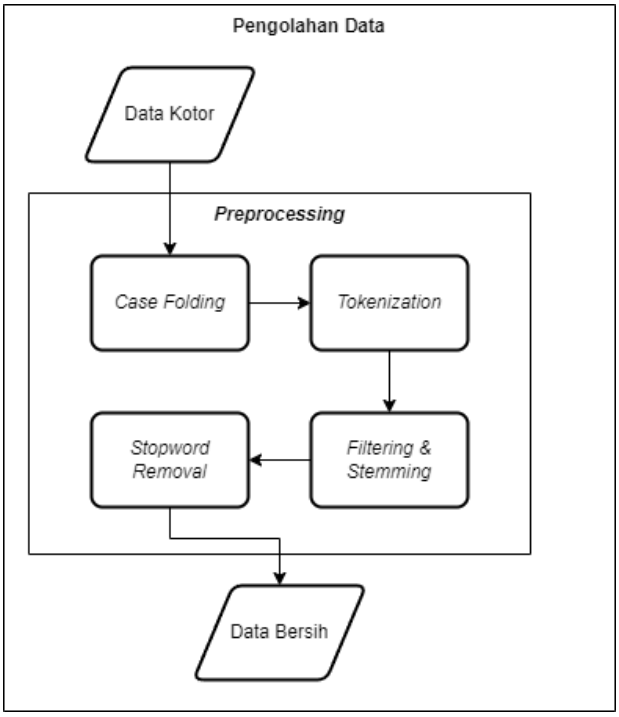
Tabel 3. 1. Contoh Data Soal dan Jawaban

Soal	**Jawaban**
Sebutkan definisi komputer.	Komputer adalah perangkat elektronik yang digunakan untuk mengolah data!
	Komputer merupakan alat bantu yang memproses informasi dengan cepat.
	Komputer dapat menjalankan program sesuai dengan instruksi yang diberikan.
	Komputer memiliki kemampuan untuk menyimpan dan mengambil data.
	Komputer terdiri dari beberapa komponen seperti CPU, RAM, dan hard disk!

Metode pengumpulan data yang dilakukan adalah metode Penelitian lapangan. Metode ini dilakukan dengan cara melakukan permintaan kepada pihak yang bersangkutan agar memperoleh data yang *valid*. Pihak yang bersangkutan adalah dosen di Jurusan Teknologi Informasi di Politeknik Negeri Malang yang bersedia memberikan datanya.

3.2.2. Pengolahan Data

Pada tahap ini peneliti menerapkan metode *preprocessing* terhadap data jawaban yang dipilih oleh *user*. Metode *preprocessing* yang digunakan menggunakan bantuan *library* yang bernama Sastrawi. Tahapan ini berguna untuk mengubah data kotor menjadi data bersih. Tahapan ini terdiri dari *case folding*, *tokenization*, *filtering* & *stemming*, dan *stopword removal*. Alur proses pengolahan data dapat dilihat pada gambar 3.2.



Gambar 3. 2. Alur Proses Pengolahan Data

1. *Case Folding*

   Case Folding merupakan metode yang digunakan untuk mengubah atau menghilangkan semua huruf kapital yang ada pada sebuah dokumen menjadi huruf kecil. Contoh hasil proses *case folding* dapat dilihat pada tabel 3.2.

Tabel 3. 2. Contoh Proses *Case Folding*

Input	*Output*
Komputer terdiri dari beberapa komponen seperti CPU, RAM, dan hard disk!	**k**omputer terdiri dari beberapa komponen seperti **cpu**, **ram**, dan hard disk!

2. *Tokenization*

   Tokenization merupakan tahapan penguraian *string* teks menjadi *term* atau kata. Tujuan dari *Tokenization* adalah memisahkan kata-kata dalam sebuah paragraf, kalimat, atau halaman ke dalam kata tunggal. Contoh hasil proses *tokenization* dapat dilihat pada tabel 3.3.

Tabel 3. 3. Contoh Proses *Tokenization*

Input	*Output*

komputer terdiri dari beberapa komponen seperti cpu, ram, dan hard disk!	['komputer', 'terdiri', 'dari', 'beberapa', 'komponen', 'seperti', 'cpu,', 'ram,', 'dan', 'hard', 'disk!']

3. *Filtering & Stemming*

Filtering dilakukan untuk menghilangkan data *string* seperti *hyperlink,* dan simbol. Sedangkan *stemming* merupakan tahapan pengubahan suatu kata menjadi kata dasarnya. Contoh hasil proses *filtering & stemming* dapat dilihat pada tabel 3.4.

Tabel 3. 4. Contoh Proses *Filtering & Stemming*

Input	*Output*
['komputer', 'terdiri', 'dari', 'beberapa', 'komponen', 'seperti', 'cpu,', 'ram,', 'dan', 'hard', 'disk!']	['komputer', '**diri**', 'dari', 'beberapa', 'komponen', 'seperti', 'cpu', 'ram', 'dan', 'hard', '**disk**']

4. *Stopword Removal*

Stopword removal, merupakan tahapan penghapusan kata-kata yang tidak relevan dalam penentuan topik sebuah dokumen dan kata yang sering muncul pada dokumen, misalnya “dan”, “atau”, “sebuah”, “adalah”, pada dokumen berbahasa Indonesia. Contoh hasil proses *stopword removal* dapat dilihat pada tabel 3.5.

Tabel 3. 5. Contoh Proses *Stopword Removal*

Input	*Output*
['komputer', 'diri', '**dari**', 'beberapa', 'komponen', '**seperti**', 'cpu', 'ram', '**dan**', 'hard', 'disk']	['komputer', 'diri', 'beberapa', 'komponen', 'cpu', 'ram', 'hard', 'disk']

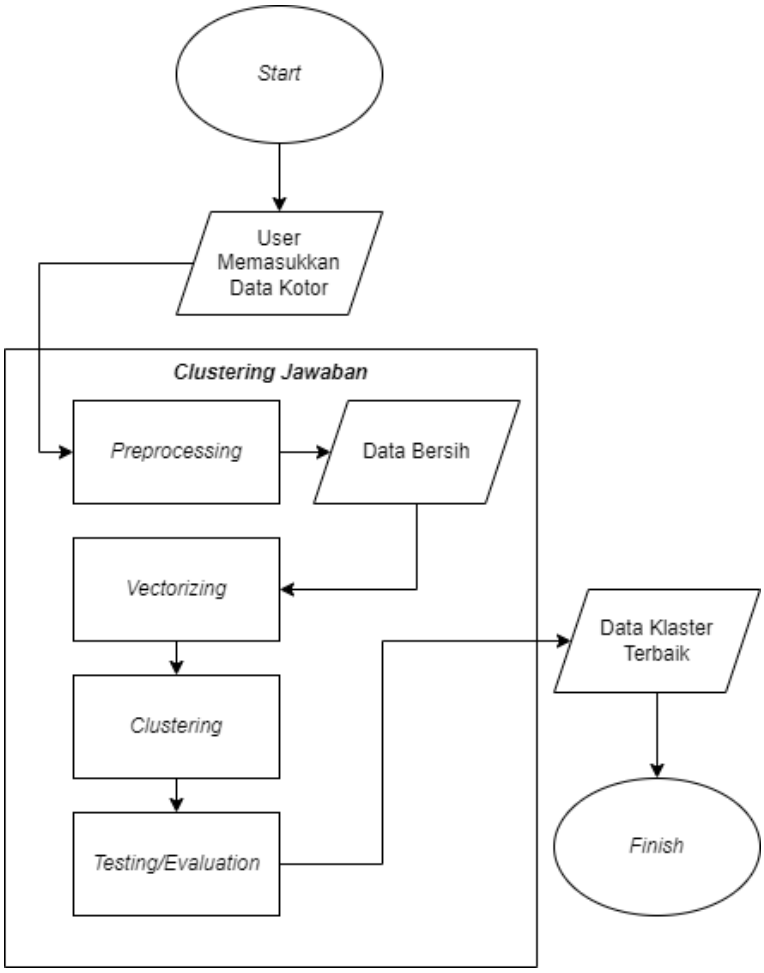
Setelah proses *stopword removal* selesai, data disatukan kembali agar menjadi sebuah kalimat. Untuk contoh hasil *preprocessing* data kotor menjadi data bersih dapat dilihat pada tabel 3.6.

Tabel 3. 6. Contoh Proses *Preprocessing*

Input (Data Kotor)	*Output* (Data Bersih)
['komputer', 'diri', 'beberapa', 'komponen', 'cpu', 'ram', 'hard', 'disk']	komputer diri beberapa komponen cpu ram hard disk

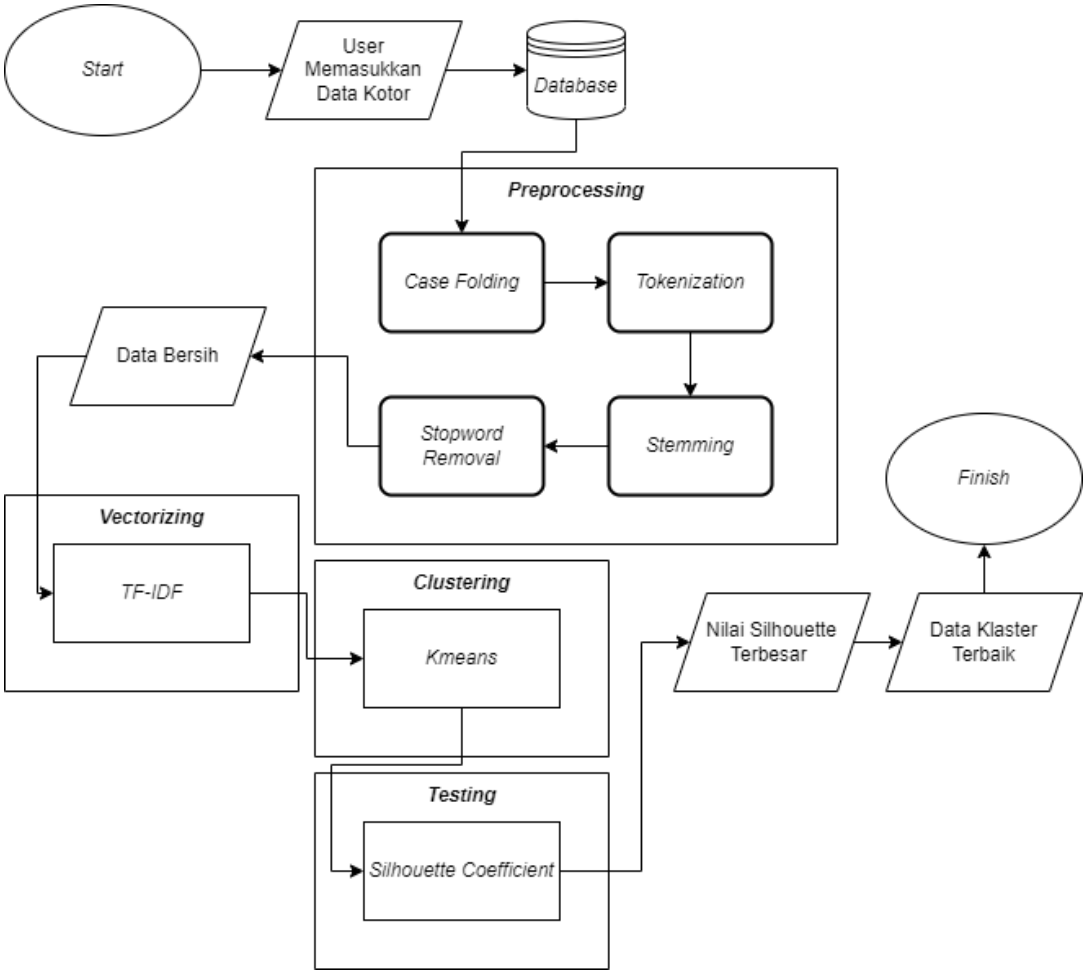
3.2.3. Pembuatan Model

Pada penelitian ini model yang dibuat adalah model untuk melakukan *clustering* menggunakan *TF-IDF* sebagai *vectorizer* dan *Kmeans* seperti yang diilustrasikan pada gambar 3.3.



Gambar 3. 3. Model Sistem

Pada gambar 3.3 digambarkan model dari sistem *Clustering* Jawaban Uraian Mahasiswa Menggunakan *TF-IDF* dan *Kmeans*. Pertama diawali dengan *user* yang memasukkan *input* berupa data kotor atau data jawaban yang ingin di proses menjadi klaster. Setelah itu sistem akan melakukan proses *clustering* jawaban dimana akan menghasilkan *output* berupa data klaster terbaik yang dapat ditampilkan oleh *user*. Untuk proses lebih lengkapnya dapat dilihat di gambar 3.4.



Gambar 3. 4. Model Detail Sistem

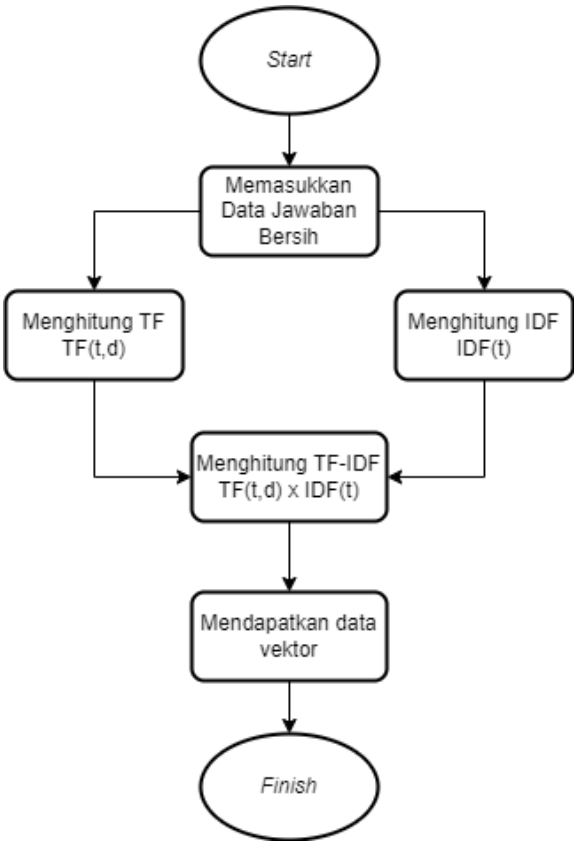
Gambar diatas merupakan detail dari proses yang dilakukan dalam sistem, sama seperti sebelumnya proses diawalai dengan *user* yang memasukkan *input* atau data kotor berupa jawaban uraian mahasiswa. Selanjunya data tersebut akan disimpan didalam *database* sebelum dilakukan *preprocessing*. Pada tahap *preprocessing* sendiri yang terdiri dari *case folding*, *tokenization*, *filtering* & *stemming*, dan *stopword removal* akan mengubah data kotor menjadi data bersih.

Selanjutnya data bersih akan masuk ke proses *vectorizing* menggunakan *TF-IDF*, setelah itu data vektor dari *TF-IDF* akan maju ke tahap *clustering* menggunakan *Kmeans*. Setelah itu data klaster akan masuk ke proses *testing* menggunakan metode *silhouette coefficient* yang akan menghasilkan data klaster yang akan menjadi *output*.

3.2.4. Penerapan Algoritma

Pada penerapan algoritma ini peneliti akan memaparkan apa saja proses dan algoritma yang digunakan untuk memproses data yang sudah diolah. Setidaknya ada tiga tahap dan empat algoritma yang akan digunakan peneliti yaitu *TF-IDF* sebagai *vectorizer*, *Kmeans* untuk tahap *clustering*, dan *silhouette coefficient* untuk tahap *testing* atau pengujian klaster.

Data yang sudah di *preprocessing* akan masuk ke tahap pembobotan menggunakan metode *TF-IDF*. Proses ini bertujuan untuk mengubah data teks menjadi data vektor karena untuk melakukan proses selanjutnya yaitu *clustering* dibutuhkan data vektor. Contoh algoritma *TF-IDF* dapat dilihat pada gambar 3.5.



Gambar 3. 5. Alur Algoritma *TF-IDF*

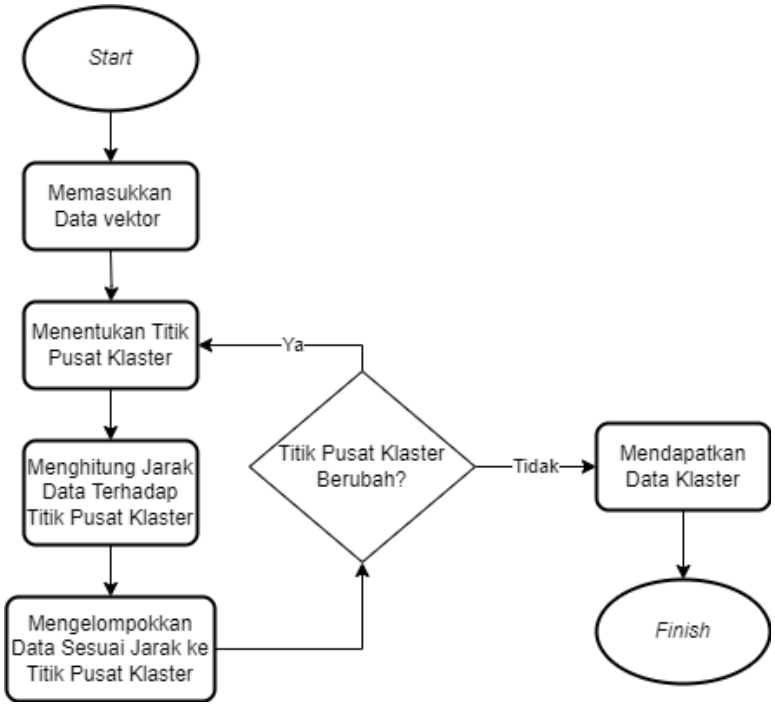
TF-IDF adalah metode ekstraksi fitur yang mengukur nilai sebuah kata dengan cara membandingkan frekuensi kata tersebut muncul dalam sebuah kalimat atau dalam kasus ini dalam sebuah jawaban dengan jumlah jawaban dimana kata tersebut muncul. Prosesnya kurang lebih sebagai berikut:

1. Setelah data bersih dimasukkan maka akan dilakukan dua proses secara bersamaan yaitu menghitung *TF(t,d)* dan *IDF(t)*. *TF* sendiri adalah

frekuensi sebuah kata muncul dalam satu jawaban contoh jawaban "*TF* singkatan dari *Term Frequency*" didapatkan nilai *TF* (singkatan) adalah berapa kali kata "singkatan" muncul pada jawaban dibagi ada berapa kata dalam jawaban tersebut. Didapat nilai *TF* (singkatan) adalah 1/5, sedangkan *IDF* adalah invers frekuensi dokumen, *IDF* ini berfungsi untuk mencari seberapa sering sebuah kata muncul dalam semua jawaban, dimana *IDF* (singkatan) = log (jumlah jawaban/jumlah jawaban yang mempunyai kata "singkatan".

2. Setelah didapat nilai *TF(t,d)* dan *IDF(t)* akan didapat nilai *TF-IDF* dengan cara mengalikan *TF* dengan *IDF* dimana data hasil perkalian ini akan menjadi data terekstrak atau data vektor.

Setelah mendapatkan data vektor maka proses selanjutnya adalah *clustering* menggunakan algoritma *Kmeans*. Proses ini bertujuan untuk mengelompokkan data berdasarkan ciri atau hasil pembobotan pada proses sebelumnya. Untuk alur proses *Kmeans clustering* ditunjukkan pada gambar 3.6.



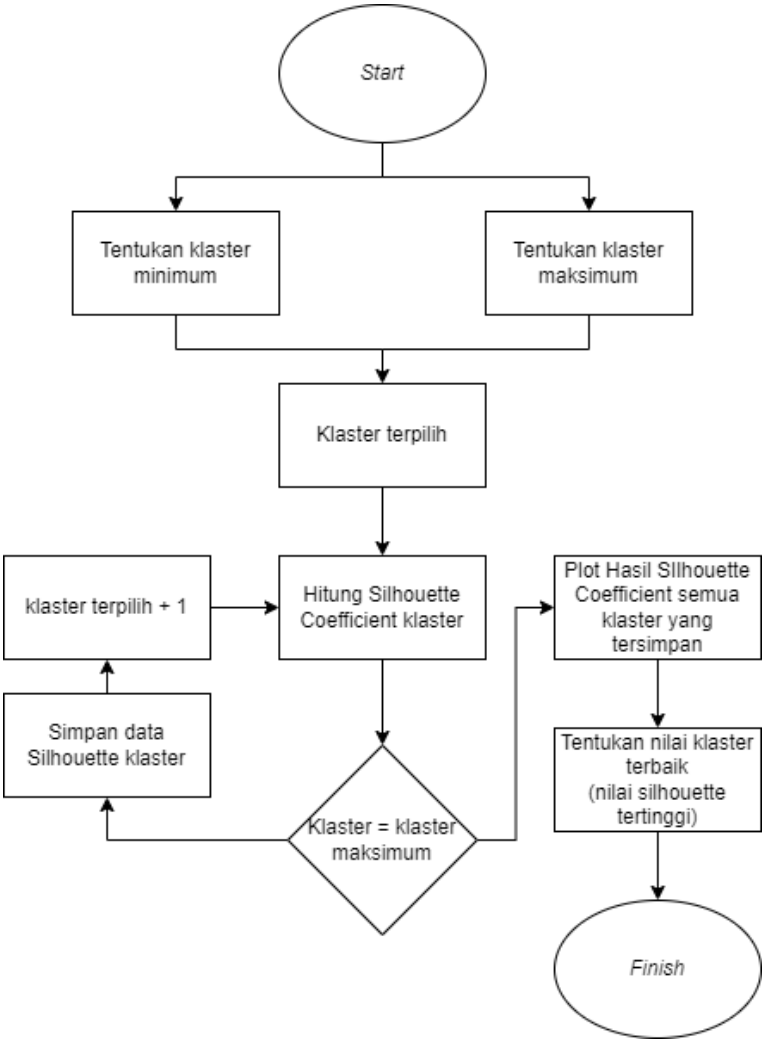
Gambar 3. 6. Alur Proses *Kmeans Clustering*

Kmeans adalah salah satu algoritma *clustering* yang cukup populer karena kecepatan dan skalabilitasnya. *Kmeans* menggunakan proses iterasi pusat dari klaster atau disebut *centroid* ke posisi rata-rata setiap data dan mengelompokkan

data tersebut ke klaster dengan *centroid* terdekat menggunakan *euclidean distance*. Proses dari *kmeans clustering* kurang lebih sebagai berikut:

1. Data vektor dimasukkan sebagai *input* dan sistem akan otomatis melakukan *looping* atau pengulangan terhadap setiap data jumlah klaster yang memungkinkan
2. Selanjutnya menentukan titik pusat klaster. Titik ini berfungsi sebagai patokan jarak yang digunakan untuk mengelompokkan data.
3. Selanjutnya menghitung jarak data terhadap titik pusat klaster.
4. Selanjutnya data yang sudah dihitung jaraknya akan dikelompokkan sesuai jarak terdekat dengan klaster.
5. Selanjutnya ditentukkan titik pusat baru dimana titik ini didapat dari nilai rata-rata data yang masuk ke klaster tersebut.
6. Setelah itu kembali dilakukan kembali langkah ke tiga sampai lima.
7. Jika titik pusat terbaru nilainya sama dengan titik pusat pada iterasi sebelumnya, maka penentuan titik pusat baru dapat dihentikan dan telah diperoleh *output* berupa hasil klaster.

Hasil dari *clustering* ini selanjutnya akan masuk ke proses *testing* menggunakan *silhouette coefficient*. Hasil *testing* ini dilakukan untuk mencari tau jumlah klaster terbaik yang akan direkomendasikan sistem untuk ditampilkan pada *user* sehingga *user* tidak perlu menentukan jumlah klaster. Alur algoritma *silhouette coefficient* dapat dilihat pada gambar 3.7.



Gambar 3. 7. Alur Proses *Silhouette Coefficient*

Pada gambar diatas dilakukan *testing* menggunakan *silhouette coefficient* yang akan menghitung jarak rata-rata data antar klaster dibagi dengan jarak rata rata terdekat data dengan klaster. Proses dari algoritma *silhouette coefficient* kurang lebih sebagai berikut.

1. Menentukan jumlah klaster minimum dan maksimum, hal ini berguna untuk proses pengulangan dimana pada proses *testing* semua jumlah klaster akan diproses.
2. Setelah didapat jumlah klaster terpilih, sistem akan menghitung nilai *silhouette*
3. Setelah itu akan ada kondisi yang memeriksa apakah jumlah klaster terpilih sama dengan klaster maksimum? Jika tidak maka data akan disimpan dandata jumlah klaster terpilih akan ditambah 1, lalu dilakukan

perhitungan *silhouette* kembali menggunakan data jumlah klaster yang baru. Jika kondisi terpenuhi maka dilakukan *plotting* atau pembuatan grafik dari semua data yang tersimpan

4. Selanjutnya berdasarkan grafik yang telah dibuat akan diambil data jumlah klaster dengan nilai *silhouette* tertinggi yang akan menjadi data klaster terbaik.

3.2.5. Penarikan Kesimpulan

Penarikan kesimpulan akan dilakukan menurut hasil dari proses uji coba. Selain uji coba klaster yang menggunakan *silhouette coefficient,* peneliti juga melakukan uji coba efektifitas sistem. Uji coba ini bertujuan untuk mencari tahu apakah *clustering* jawaban uraian mahasiswa ini dapat benar-benar membantu dosen mempermudah proses penilaian jawaban uraian mahasiswa atau tidak.

Uji coba ini akan dilakukan dengan cara membandingkan hasil pengelompokkan jawaban secara manual dan pengelompokkan jawaban menggunakan sistem yang dibuat, berikut adalah kemungkinan yang dapat timbul dari hasil pengujian efektifias

1. Tidak efektif: Jika persentase < 25% atau perbandingan durasi antara proses manual dengan sistem adalah > 1/1
2. Kurang efektif: jika persentase > 25% dan < 50% atau perbandingan durasi antara proses manual dengan sistem adalah < 1/1 dan > 1/2
3. Cukup efektif: jika persentase > 50% dan < 75% atau perbandingan durasi antara proses manual dengan sistem adalah < 1/2
4. Sangat efektif: jika persentase > 75% dan perbandingan durasi antara proses manual dengan sistem adalah < 1/2

Pada uji coba ini jumlah klaster akan ditentukan berdasarkan jumlah klaster yang memiliki nilai *silhouette coefficient* terbesar dan yang akan menjadi indikator pembanding adalah persentase ketepatan klaster dan durasi dilakukannya pengelompokkan. Untuk contoh tabel pengujian efektifitas sistem dapat dilihat pada tabel 3.7.

Tabel 3. 7. Contoh Pengujian Efektifitas

Klaster = 3 (dimisalkan)			
Soal	Jawaban	Kelompok/	Hasil

		Klaster (A-C)		(Sesuai/Tidak)
		Manual	Sistem	
Sebutkan definisi komputer.	Komputer adalah perangkat elektronik yang digunakan untuk mengolah data!	A/B/C	A/B/C	Sesuai/Tidak
	Komputer merupakan alat bantu yang memproses informasi dengan cepat.	A/B/C	A/B/C	Sesuai/Tidak
	Komputer dapat menjalankan program sesuai dengan instruksi yang diberikan.	A/B/C	A/B/C	Sesuai/Tidak
	Komputer memiliki kemampuan untuk menyimpan dan mengambil data.	A/B/C	A/B/C	Sesuai/Tidak
	Komputer terdiri dari beberapa komponen seperti CPU, RAM, dan hard disk!	A/B/C	A/B/C	Sesuai/Tidak
Durasi		menit/	menit	
Persentase				%