BAB 2
TINJAUAN PUSTAKA

2.1 Analisis Sentimen

Opini dan konsep terkait seperti sentimen, evaluasi, sikap, dan emosi adalah subyek dari studi analisis sentimen dan *opinion mining*. Sejak awal tahun 2000, analisis sentimen telah berkembang menjadi salah satu daerah penelitian yang paling aktif dalam pengolahan bahasa alami (*natural language processing*). Hal tersebut juga banyak dipengaruhi oleh perkembangan studi mengenai *data mining*, *web mining*, dan *text mining*. Beberapa tahun terakhir, kegiatan industri sekitar analisis sentimen juga berkembang. Sekarang analisis sentimen hampir dapat ditemukan pada setiap bisnis dan domain sosial [3].

Analisis sentimen atau biasa disebut juga *opinion mining* adalah bidang studi yang menganalisa opini , sentimen, evaluasi, sikap, dan emosi terhadap suatu entitas [7]. Entitas tersebut dapat berupa produk, jasa, organisasi, individu, masalah, peristiwa atau sebuah topik. Aplikasi analisis sentimen telah menyebar ke hampir setiap domain mulai dari produk, konsumen, jasa, kesehatan, dan jasa keuangan untuk kegiatan sosial dan politik.

Analisis sentimen berfokus pada opini yang menyatakan atau menyiratkan sentimen positif atau negatif. Secara garis besar analisis sentimen merupakan proses klasifikasi dokumen tekstual ke dalam dua kelas, yaitu kelas sentimen positif dan kelas sentimen negatif. Analisis sentimen dilakukan untuk melihat pendapat atau opini terhadap sebuah masalah atau dapat juga digunakan untuk identifikasi kecenderungan suatu hal pada lingkup tertentu. Menurut Liu, tujuan utama dari analisis sentimen adalah untuk melakukan ekstraksi atribut pada sebuah dokumen atau teks berisi komentar untuk mengetahui ekspresi-ekspresi yang ada di dalamnya sehingga komentar-komentar tersebut dapat bisa dikategorikan positif atau negatif.

2.2 Klasifikasi

Salah satu hal yang mendasar dan penting dalam analisis sentimen saat ini adalah klasifikasi. Umumnya banyak yang menafsirkan istilah "klasifikasi" secara

luas, sehingga mencakup regresi dan *ranking*. Alasan bahwa klasifikasi sangat penting adalah bahwa banyak masalah yang menarik dapat dirumuskan sebagai penerapan klasifikasi / regresi / *ranking* ke unit tekstual [8]. Terdapat beberapa metode klasifikasi di antaranya seperti *naive bayes classifier, k-nearest neighbour, decision trees, neural network, support vecor machine* dan lain-lain [9].

2.3 Information Retrieval

Information Retrieval (IR) adalah proses untuk menemukan atau mencari suatu "*material*" (biasanya dokumen) yang bersifat tidak terstruktur (biasanya teks) yang memenuhi kebutuhan informasi dari dalam koleksi yang besar (biasanya disimpan di komputer) [8]. *Information retrieval* juga dapat mencakup jenis data lain.

Fokus dari *information retrieval* adalah bagaimana membuat representasi, mencari, dan memanipulasi koleksi dokumen digital agar pengguna mudah mencari informasi yang diperlukan. Informasi atau data yang dicari dapat berupa berupa teks, *image, audio, video* dan lain-lain. Koleksi data teks yang dapat dijadikan sumber pencarian juga dapat berupa pesan teks, seperti *e-mail*, *fax*, dan dokumen berita, bahkan dokumen yang beredar di internet. Dengan jumlah dokumen koleksi yang besar sebagai sumber pencarian, maka dibutuhkan suatu sistem yang dapat membantu *user* menemukan dokumen yang relevan dalam waktu yang singkat dan tepat.

Pada *information retrieval* terdapat tahapan *preprocessing* yang berguna untuk mengolah data sebelum dilakukan proses tertentu pada tahapan selanjutnya.

2.3.1 Preprocessing

Sebelum dilakukan proses klasifikasi, diperlukan proses *preprocessing* terhadap data mentah yang digunakan akan sebagai materi untuk klasifikasi nantinya. Tujuan *preprocessing* adalah mentransformasikan data ke dalam suatu format data yang lebih mudah untuk diproses sesuai dengan kebutuhan penelitian [10]. Langkah-langkah *preprocessing* pada penelitian ini di antaranya adalah sebagai berikut [11]:

1. Casefolding
   Langkah awal dari *preprocessing* dimulai dengan melakukan *casefolding* terhadap *tweet* yang akan diproses. *Casefolding* merupakan proses mengubah semua teks pada *tweet* menjadi huruf kecil *(lowercase)*.

2. Normalisasi Fitur
   Merupakan proses menghilangkan atau menghapus komponen-komponen khas yang terdapat pada *tweet* seperti *username, url, rt*. Hal ini dilakukan karena komponen tersebut sama sekali tidak akan berpengaruh terhadap sentimen dari suatu *tweet*. Identifikasi komponen-komponen pada normalisasi fitur akan menggunakan *regular expression*.

Tabel 2.1. Normalisasi Fitur

Krarakter					
0	8	&	`	:	?
1	9	*	~	;	/
2	!	(	[	“	
3	@	)	]	‘	
4	#	-	\	<	
5	$	_	|	>	
6	%	+	{	,	
7	^	=	}	.	

3. Convert Emoticon
   Merupakan proses konversi dari setiap *emoticon (emotion icon)* yang terdapat pada *tweet* ke dalam bentuk *string* yang merepresentasikan *emoticon* tersebut. *Emoticon* atau *emotion icon* adalah sebuah simbol atau kombinasi dari simbol-simbol yang biasanya digunakan untuk menggambarkan ekspresi wajah manusia yang mengandung emosi atau perasaan dalam bentuk pesan atau tulisan. Pada penelitian yang dilakukan oleh *Datagenetics* terhadap 96,269,892 tweets yang mengandung *emoticon*.

Emoticon yang sering digunakan oleh para pengguna *twitter* adalah sebagai berikut :

Tabel 2.2. *Emoticon* [12]

No	Emoticon	Jumlah Tweet	Persentase
1	:)	32.115.789	33.36%
2	:D	10.595.385	11.01%
3	:(	7.613.014	7.91%
4	;)	7.238.295	7.52%
5	:-)	4.254.708	4.42%
6	:P	3.588.863	3.73%
7	=)	3.564.080	3.70%
8	(:	2.720.383	2.83%
9	;-)	2.085.015	2.17%
10	:/	1.840.827	1.91%
11	XD	1.795.792	1.87%
12	=D	1.434.004	1.49%
13	:o	1.077.124	1.12%
14	=]	1.055.517	1.10%
15	D:	1.048.320	1.09%
16	;D	1.004.509	1.04%
17	:]	954.740	0.99%
18	:-(	816.170	0.85%
19	=/	809.760	0.84%
20	=(	760.600	0.79%

Emoticon yang digunakan pada penelitian ini adalah *emoticon western style*. Berikut ini adalah tabel konversi *emoticon* yang digunakan :

Tabel 2.3. Konversi *Emoticon*

Emoticon	Konversi
>:] ;) :-) :) :o) :] :3 :c) :> =] 8) =) :} :^) (: ;-)	senang
>:D :-D :D 8-D 8D x-D xD XD XD =-D =D =-3 =3 ;D	tawa
>:[:-(:(:-c :c :-< :< :-[:[:{ >.> <.< >.< =(	sedih
>:\ >:/ :-/ :-. :/ :\ =/ =\ :S :P	kesal
>:o >:O :-O “o” “O” :O o_O o.O 8-0 :o	kaget

4. Remove Stopwords

Stopwords adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna, sehingga tidak akan mempengaruhi sentimen. *Stop words* biasanya berupa kata ganti atau kata sambung.

5. Negation Handling

   Negation handling atau penanganan negasi menjadi perhatian penting dalam analisis sentimen. Hal ini karena kata negasi dapat mengubah nilai positif atau negatif dari data yang dianalisis. Jika pada *tweet* terdapat kata negasi, maka kata negasi akan digabungkan dengan kata setelahnya sehingga menjadi satu kata baru.

6. Tokenization

   Tokenization atau tokenisasi merupakan proses untuk memisah-misahkan atau memecah suatu kalimat menjadi kata-kata yang terpisah. Token adalah turunan dari urutan karakter dalam beberapa dokumen tertentu yang dikelompokkan bersama sebagai unit semantik yang berguna untuk diproses.

7. Stemming

   Stemming merupakan salah satu proses yang digunakan untuk meningkatkan performa IR dengan cara mentransformasikan kata-kata dalam sebuah dokumen teks ke dalam bentuk kata dasarnya. Algoritma untuk proses *stemming* pada penelitian ini menggunakan algoritma Nazief & Adriani. Algoritma ini memiliki keakuratan yang baik dan presisi untuk teks berbahasa Indonesia [13].

   Algoritma yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut [13]:

   1. Cari kata yang akan distem dalam kamus. Jika ditemukan maka diasumsikan bahwa kata tesebut adalah *root word*.Maka algoritma berhenti.
   2. *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dibuang. Jika berupa particles (“-lah”, “-kah”, “-tah” atau “-pun”)

maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.

3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”).Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah 3a
    a. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
    b. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hapus *Derivation Prefix*. Jika pada langkah 3 ada sufiks yang dihapus maka pergi ke langkah 4a, jika tidak pergi ke langkah 4b.
    a. Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritma berhenti, jika tidak pergi ke langkah 4b.
    b. For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika *root word* belum juga ditemukan lakukan langkah 5, jika sudah maka algoritma berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritma berhenti.
5. Melakukan *Recoding*.
6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai *root word*. Proses selesai.

2.4 Support Vector Machine

Support Vector Machine dikembangkan oleh Boser, Guyon, Vapnik dan dipertamakali dipresentasikan pada tahun 1992 di *Annual Workshop on Computational Learning Theory*. Konsep dasar SVM sebenarnya merupakan kombinasi dari teori-teori komputasi yang telah ada puluhan tahun sebelumnya. Prisip dasar SVM adalah pengklasifikasi *linear (linear classifier)*, dan selanjutnya dikembangkan agar dapat bekerja pada masalah *non-linear*, dengan memasukkan *kernel trick* pada ruang kerja berdimensi tinggi. Fungsi kernel yang biasa digunakan biasanya yaitu *linie*r, *polynomial*, *radial basis function* dan *sigmoid*. Teknik ini berusaha untuk menemukan fungsi pemisah (*classifier*) yang optimal yang bisa memisahkan dua set data dari dua kelas yang berbeda [14].

Pada kasus klasifikasi secara linier digunakan fungsi pemisah yang dapat didefinisikan sebagai berikut :

$$g(x) \coloneqq sgn\big(f(x)\big) \tag{2.1}$$
$$dengan\ f(x) = w^T x + b,$$

Sedangkan pada kasus klasifikasi secara non linear untuk menemukan hyperplane pemisah yang akurat untuk mengkasifikasikan dua kelas, adalah dengan menggunakan fungsi pemisah nonlinear (yaitu, satu yang fungsinya pemetaan ∅ adalah pemetaan nonlinear dari ruang input ke dalam beberapa ruang fitur). Didefinisikan sebagai berikut :

$$\Phi\begin{pmatrix} x \\ y \end{pmatrix} = \begin{cases} \begin{pmatrix} 6 - x_1 + (x_1 - x_2)^2 \\ 6 - x_2 + (x_1 - x_2)^2 \end{pmatrix} jika \sqrt{x_1^2 + x_2^2} \geq 2 \\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} jika\ tidak \end{cases} \tag{2.2}$$

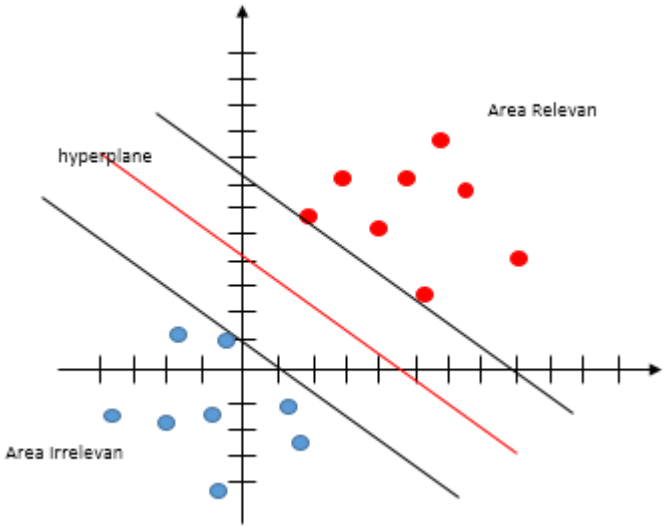
Kemudian hitung nilai untuk menemukan 3 parameter $\alpha_1, \alpha_2, dan\ \alpha_3$ didasarkan pada persamaan linear berikut

$$\begin{aligned} \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} &= +1\ (postif) \\ \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} &= +1\ (positif) \\ \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} + \alpha_1 \widetilde{S_1}.\widetilde{S_1} &= -1\ (negatif) \end{aligned} \tag{2.3}$$

Setelah mendapatkan nilai α , maka langkah selanjutnya adalah mencari hyperplane untuk memisahkan kelas positif dan kelas negatif menggunakan persamaan (II.4) [15].

$$\widetilde{W} = \sum_{i} \alpha_i \tilde{S}_i \qquad (2.4)$$

Berikut adalah gambaran dari penggunaan metode support vector machine pada pemisahan data relevan dan tidak relevan.



Gambar 2.1. Pemisah data relevan dan tidak relevan

2.5 Naive Bayes Classifier

Naive bayes classifier adalah metode klasifikasi probabilistik sederhana berdasarkan penerapan teorema *Bayes* (dari statistik *Bayesian*) dengan asumsi independen (naif) yang kuat. Sebuah istilah yang lebih deskriptif untuk model probabilitas yang digaris bawahi adalah "model fitur independen". *Naive bayes classifier* mengasumsikan bahwa kehadiran (atau ketiadaan) fitur tertentu dari suatu kelas tidak berhubungan dengan kehadiran (atau ketiadaan) fitur lainnya. Tergantung pada situasi yang tepat dari model probabilitas, naive bayes classifier dapat dilatih sangat efisien dalam *supervised learning*. Kelebihan dari *naive bayes classifier* adalah hanya memerlukan sejumlah kecil data pelatihan untuk mengestimasi parameter (rata-rata dan varian dari variabel) yang diperlukan untuk

klasifikasi. Karena variabel diasumsikan independen, hanya varian dari variabel-variabel untuk setiap kelas yang perlu ditentukan.

Metode *naive bayes classifier* menempuh dua tahap dalam proses klasifikasi teks, yaitu tahap pelatihan dan tahap klasifikasi. Pada tahap pelatihan dilakukan proses analisis terhadap sampel dokumen berupa pemilihan *vocabulary*, yaitu kata yang mungkin muncul dalam koleksi dokumen sampel yang sedapat mungkin dapat menjadi representasi dokumen. Selanjutnya adalah penentuan probabilitas prior bagi tiap kategori berdasarkan sampel dokumen. Pada tahap klasifikasi ditentukan nilai kategori dari suatu dokumen berdasarkan term yang muncul dalam dokumen yang diklasifikasi [5]. Dengan menerapkan teorema *bayes* , maka persamaan untuk klasifikasi adalah sebagai berikut :

$$Vmap = arg \max_{Vj \in V} \; P(V_j) \prod \prod P(w_k | V_j) \tag{2.5}$$

Nilai $P(V_j)$ ditentukan pada saat pelatihan, yang nilainya diperoleh dengan :

$$P(V_j) = \frac{|doc_j|}{|Contoh|} \tag{2.6}$$

dimana doc_j adalah banyaknya keyword yang memiliki kategori j dalam pelatihan, sedangkan $|Contoh|$ banyaknya keyword dalam contoh yang digunakan untuk pelatihan.

Untuk nilai $P(w_k \mid V_j)$, yaitu probabilitas kata w_k dalam kategori j ditentukan dengan :

$$P(w_k | V_j) = \frac{n_k + 1}{n + |vocabulary|} \tag{2.7}$$

dimana n_k adalah frekuensi kemunculan kata w_k dalam dokumen yang berkategori V_j, sedangkan nilai n adalah banyaknya seluruh kata dalam dokumen berkategori V_j dan $|vocabulary|$ adalah banyaknya kata dalam contoh pelatihan.

Tahapan pelatihan dan klasifikasi *naive bayes classifier* adalah sebagai berikut

Pelatihan :

1. Bentuk vocabulary
2. Untuk setiap kategori V_j hitung :
    a. Tentukan doc_j (himpunan dok dalam kategori V_j)

b. Hitung $P(V_j)$ dengan persamaan (2)

c. Hitung $P(w_k | V_j)$ dengan persamaan (3) untuk setiap w_k dalam vocabulary

Klasifikasi :

1. Hitung $P(V_j) \prod P(w_k | V_j)$ untuk setiap kategori V_j
2. Tentukan nilai maksimumnya sebagai hasil kategorisasi.

2.6 Pembobotan Kata

Metode yang digunakan untuk pembobotan kata pada penelitian ini adalah metode TF-IDF *(term frequency – inverse document frequency*). Metode TF-IDF dapat dirumuskan sebagai berikut :

$$idf = log\left(\frac{d}{df}\right) \quad (2.8)$$

dimana :

idf adalah invers dokumen *frequency*

d adalah total dokumen

df adalah jumlah dokumen yang mengandung *term* (kata)

Bobot dari setiap *term* (kata) dapat dihitung dengan rumus $w = tf * idf$

2.7 Confusion Matrix

*Confusion Matr*ix berisi informasi tentang klasifikasi aktual dan yang telah terprediksi yang dilakukan oleh sistem klasifikasi. Pada umumnya dievaluasi dengan menggunakan data matriks. Metode klasifikasi akan dievaluasi terutama pada bagian akurasi dari hasil klasifikasi. Akurasi sebuah klasifikasi berpengaruh terhadap performa dari suatu klasifikasi. Untuk melakukan analisa dapat digunakan confusion matrix, yaitu sebuah matrik dari prediksi yang akan dibandingkan dengan kelas yang asli dari data inputan [16]. Berikut ini contoh tabel yang menunjukan *confusion matrix* untuk klasifikasi dua kelas.

Tabel 2.4. *Confusion Matrix*

		Predicted Class	
		Positif	Negatif
Actual Class	Positif	True Positives	False Negatives
	Negatif	False Positives	True Negatives

Keterangan :

True Positives : Jumlah *record* positif yang diklasifikasikan sebagai positif

Flase Positives: Jumlah *record* positif yang diklasifikasikan sebagai negatif

False Negatives : Jumlah *record* negatif yang diklasifikasikan sebagai positif

True Negatives: Jumlah *record* negatif yang diklasifikasikan sebagai negatif.

Setiap kolom dari tabel confusion matrix merupakan contoh di kelas yang telah diprediksi, sedangkan setiap baris mewakili contoh di kelas yang sebenarnya. Setelah mendapatkan nilai untuk masing-masing kelas, selanjutnya adalah menghitung nilai *precision* dan akurasinya. *Precision* adalah ukuran terhadap suatu kelas yang telah diprediksi. Berikut ini adalah persamaannya :

$$Akurasi = \frac{TP - TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.9}$$

dengan TP adalah true positives, FP False positives, TN True Negatives,dan FN adalah False Negatives. [16]

2.8 Object Oriented Programming

Object Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan suatu *software* dimana dalam struktur *software* tersebut didasarkan kepada interaksi object dalam penyelesaian suatu proses tugas. Interaksi tersebut mengambil *form* dari pesan-pesan dan mengirimkannya kembali antar *object* tersebut. *Object* akan merespon pesan tersebut menjadi sebuah tindakan atau metode. Adapun konsep pada pemrograman berorientasi objek yaitu :

1. Object

   Merupakah sebuah struktur yang menggabungkan data dan prosedur untuk bekerja bersama-sama

2. Abstraction

   Ketika membangun *objects* dalam aplikasi OOP, penting untuk menggabungkan konsep *abstraction* ini.

3. Enkapsulasi

   Sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan *hidden*. Jika ingin mendapat data, harus berinteraksi dengan *object* yang bertanggung jawab atas data tersebut

4. Polymorphisms

   Kemampuan dua buah *object* yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik. Pada OOP, diterapkan tipe *polymorphism* melalui proses yang disebut *overloading*. Dapat dilakukan dalam implementasi metode yang berbeda pada sebuah *object* yang mempunyai nama yang sama.

5. Inheritance

   Penggunaan *inheritance* dalam OOP adalah untuk mengklasifikasikan objects dalam program sesuai karakteristik umum fungsinya. Hal ini akan membuat pekerjaan bersama *object* lebih mudah dan lebih intuitif. Hal ini juga membuat *programming* lebih mudah karena memungkinkan untuk mengkombinasikan karakteristik umum ke dalam *object parent* dan mewariskan karakteristik ke *child object.*

2.8.1 Unified Modelling Language (UML)

Unified Modelling Language adalah bahasa pemodelan untuk sistem atau perangkat lunak berorientasi objek. Pemodelan sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sehingga lebih mudah dipelajari dan dipahami. UML digunakan untuk melakukan analisis dan perancangan sistem/perangkat lunak. UML merupakan bahasa standar untuk merancang dan mendokumentasikan perangkat lunak yang berorientasi objek.

1. Use Case Diagram

   Use case diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. Terdiri atas diagram untuk *use case* dan *actor*. Actir merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi. *Use case* merepresentasikan operasi-operasi yang dilakukan oleh *actor*. *Use case* digambarkan berbentuk elips dengan nama operasi dituliskan didalamnya. *Actor* yang melaukan operasi dihubungkan dengan garis lurus ke *use case*.

2. Activity Diagram

   Merupakan representasi grafis dari alur kerja tahapan aktivitas. Diagram ini mendukung pilihan tindakan, iterasi dan *concurrency*. Pada pemodalan UML, *activity diagram* dapat digunakan untuk menjelaskan bisnis dan alur kerja operasional secara *step-by-sep* dari komponen suatu sistem.

3. Sequence Diagram

   Menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case*. Interaksi yang terjadi antar *class*, operasi apa yang terlibat, urutan antar operasi dan informasi yang diperlukan oleh masing-masing operasi.

4. Class Diagram

   Class Diagram merupakan diagram yang selalu ada dalam pemodelan sistem berorientasi objek. *Class* diagram menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan