BAB 2
LANDASAN TEORI

2.1 Keamanan Sistem Informasi

Keamanan sistem informasi merupakan sebuah perlindungan pribadi di media digital (*online*), termasuk aset digital dan identitas pribadi. Keamanan ini menjadi hal yang harus diperhatikan karena masih banyak orang yang menggunakan kemudahan teknologi untuk kegiatan terlarang. Ancaman keamanan sistem informasi dapat berupa *ransomware*, *man-in-the-middle*, *phising*, dan lain sebagainya.

Aspek keamanan digital terdiri dari tiga hal, yaitu *confidentiality* (kerahasiaan) yang berkaitan dengan kemampuan sistem menjaga kerahasiaan data dan informasi yang disimpan di dalamnya, *integrity* (integritas) yang merupakan kemampuan sistem menjaga keaslian data dan informasi, dan *availability* (ketersediaan) yang berkaitan dengan kemampuan sistem untuk selalu tersedia dan dapat diakses oleh pengguna terdaftar. Tujuan dari keamanan sistem informasi ini adalah untuk meminimalkan dan mencegah adanya modifikasi dan penggunaan informasi olah orang lain yang tidak berhak [7].

Dengan perkembangan teknologi yang semakin pesat saat ini, keamanan sistem informasi juga perlu dikembangkan agar fungsinya tidak melemah. Salah satu perkembangan keamanan sistem informasi yang paling banyak digunakan saat ini adalah menggunakan biometrik. Biometrik sendiri merupakan sistem atau metode autentikasi yang memanfaatkan data atau identitas biologis pengguna. Hal ini dikarenakan identitas biologis setiap orang pastinya akan berbeda dan memiliki ciri khas tersendiri sehingga metode biometrik menjadi pilihan yang tepat untuk mencegah ancaman keamanan data. Saat ini terdapat beberapa sistem biometrik yang digunakan untuk verifikasi keamanan, seperti pemindai retina, iris, sidik jari, wajah, bahkan DNA [8].

2.2 Pengenalan Wajah

Wajah manusia memiliki karakteristik unik dan berbeda antara satu dengan yang lainnya sehingga sistem pengenalan wajah menjadi salah satu sumber identifikasi yang baik. Sistem pengenalan wajah merupakan sebuah teknologi yang mampu mengidentifikasi atau memverifikasi subjek melalui gambar, video,

atau elemen audiovisual apa pun dari wajah seseorang. Studi mengenai sistem pengenalan wajah berkaitan dengan pengenalan pola, penggunaan kecerdasan buatan agar komputer dapat memperoleh pemahaman tingkat tinggi dari gambar atau video digital [9].

Sistem pengenalan wajah bekerja dengan mengambil gambar wajah dan mengubahnya menjadi representasi digital. Representasi digital ini kemudian akan dibandingkan dengan data yang telah disimpan pada *database*. Jika kedua data cocok, maka sistem dapat mengenali wajah orang tersebut dan memberikan akses atau izin yang sesuai, jika tidak maka akses akan ditolak. Sistem pengenalan wajah yang baik harus dapat membedakan antara wajah yang berbeda dengan akurasi yang tinggi. Jika sistem pengenalan wajah tidak akurat, maka akan besar kemungkinan munculnya kesalahan pemberian izin atau akses [10].

2.3 *Triangle Face*

Triangle face merupakan satu dari banyaknya metode yang dapat digunakan dalam melakukan deteksi wajah pada citra digital. Metode ini bekerja dengan mendeteksi fitur-fitur pada wajah seseorang yang membentuk segitiga sehingga disebut *triangle face*. Jarak antar fitur tersebut dihitung dan digunakan sebagai parameter untuk melakukan pengenalan wajah [11]. Fitur-fitur wajah yang digunakan meliputi mata kanan, mata kiri, hidung, mulut, tinggi wajah, dan lebar wajah. Jarak yang dicari menggunakan fitur-fitur wajah meliputi:

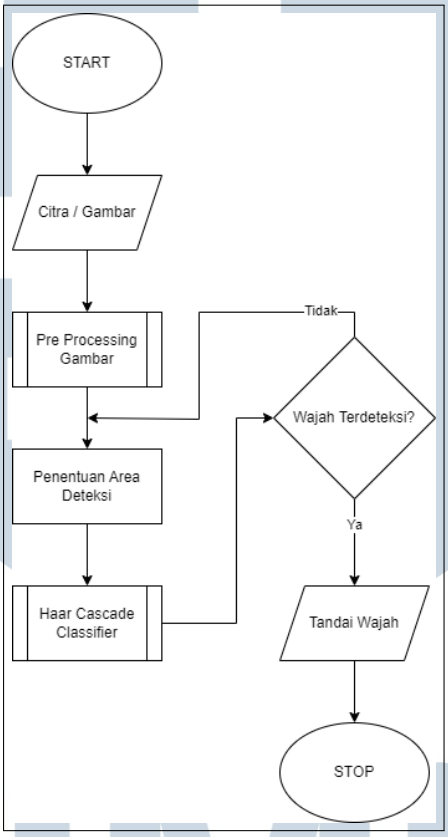
- Jarak mata kanan ke mata kiri.
- Jarak mata kanan ke mulut.
- Jarak mata kiri ke mulut.
- Jarak mata kanan ke hidung.
- Jarak mata kiri ke hidung.

Terdapat beberapa tahap pada metode ini yang harus dilakukan dalam pengenalan wajah, yaitu:

1. Deteksi Wajah

   Pengenalan wajah harus diawali dengan menentukan bagian wajah seseorang yang akan dikenali. Setelah bagian wajah ditemukan, dilakukan pemotongan area wajah untuk mempersempit ruang dalam pengolahan citra. Pada tahap ini digunakan *Haar Cascade Classifier* untuk membentuk segi empat yang mengelilingi wajah [12]. *Haar Cascade Classifier* sendiri merupakan salah

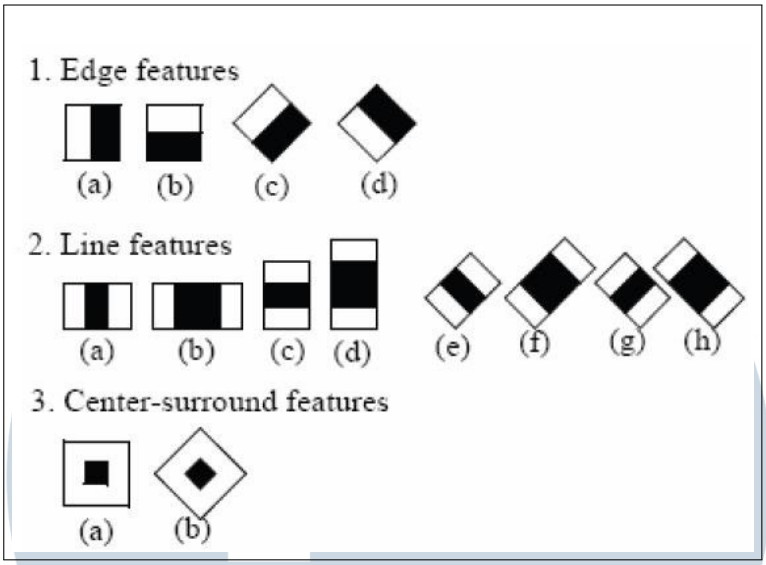
satu model *machine learning* yang biasanya digunakan sebagai dasar *object detection* (terutama deteksi wajah), dalam sebuah gambar atau video [13]. Alur proses yang dilakukan pada tahap ini dapat dilihat pada Gambar 2.1 Proses deteksi wajah menggunakan *Haar Cascade Classifier*.



Gambar 2.1. **Proses deteksi wajah menggunakan *Haar Cascade Classifier***

2. Pencarian Posisi Fitur Wajah

   Fitur-fitur pada wajah dideteksi menggunakan tepi-tepi area wajah meliputi posisi mata, hidung, dan mulut. Dalam menentukan posisi fitur wajah ini digunakan *Haar Feature*. *Haar Feature* merupakan fitur yang didasarkan pada *Wavelet Haar*, yakni gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah), atau satu terang dan satu gelap untuk dua dimensi [14].



Gambar 2.2. ***Haar Feature***

Sumber: [14]

3. Penentuan Titik Ukur dan Pengukuran Jarak

   Setelah fitur-fitur pada wajah berhasil didapatkan, maka dilakukan penentuan titik ukur yang akan digunakan untuk menentukan titik pusat koordinat pixel x dan y untuk setiap fitur wajah. Setelah titik pusat koordinar ditentukan, dilakukan pengukuran jarak antar fitur seperti yang telah disebutkan. Persamaan yang digunakan untuk mengukur jarak adalah sebagai berikut:

$$Fi = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2.1}$$

4. Normalisasi

   Pada tahap ini dilakukan normalisasi untuk menghilangkan pengaruh perbedaan ukuran citra objek ketika dilakukan pengukuran pada penentian titik ukur. Normalisasi dilakukan menggunakan persamaan berikut:

$$fi = \frac{F_i}{\sum_{i=1}^{n} F_i} \tag{2.2}$$

   Dari persamaan normalisasi di atas nilai normalisasi didapat dari hasil ukur masing-masing fitur dibagi dengan penjumlahan semua fitur yang digunakan pada citra.

2.4 *Minkowski Distance*

Minkowski distance adalah sebuah matrik pada ruang *euclidean* yang merupakan generalisasi dari jarak *euclidean* dan jarak *manhattan* [15]. Jarak yang dihitung menggunakan *minkowski distance* dapat menentukan kemiripan dari dua buah citra. Semakin mirip dua buah citra, maka akan semakin kecil jaraknya. Sebaliknya, semakin tidak mirip dua buah citra, maka akan semakin besar jaraknya [16]. *Minkowski distance* digunakan untuk menghitung kemiripan dua buah citra. Rumus dari *minkowski distance* adalah:

$$\text{dist} = \left(\sum_{k=1}^{n} |p_k - q_k|^r \right)^{\frac{1}{r}} \tag{2.3}$$

2.5 *Use Case* Diagram

Use case diagram merupakan salah satu dari banyaknya jenis diagram UML (*Unified Modelling Language*) yang menggambarkan hubungan interaksi antara sistem dan aktor. Pada diagram ini akan dijelaskan interaksi yang terjadi antara pengguna sistem dengan sistemnya. *Use case* diagram berfungsi untuk menggambarkan proses aktivitas secara urut dalam sistem, menampilkan urutan aktivitas pada sebuah proses, dan sebagai jembatan antara pembuat dengan konsumen untuk mendeskripsikan sebuah sistem.

Komponen dari *use case* diagram antara lain:

1. Sistem

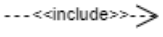
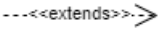
   Sebuah sistem digambarkan ke dalam bentuk persegi. Hal ini berfungsi untuk membatasi *use case* dengan interaksi dari luar sistem.

2. Aktor

   Aktor berfungsi untuk menjelaskan siapa yang berinteraksi dengan sistem. Aktor berperan untuk memberikan informasi kepada sistem, ataupun menerima informasi dari sistem. Keduanya dapat terjadi secara bersamaan. Aktor tidak memberikan kontrol terhadap sistem, namun memberikan gambaran mengenai hubungan dengan sistem.

3. *Use case Use case* merupakan komponen gambaran fungsional dalam sebuah sistem. Komponen ini dapat membantu konsumen maupun pembuat untuk saling mengenal dan mengerti alur sistem yang akan dibuat.

Simbol-simbol pada *use case* diagram:

Simbol	Nama	Keterangan
	Aktor	Mewakili orang, sistem yang lain, atau alat ketika berkomunikasi dengan use case.
	Use case	Abstraksi dan interaksi antara sistem dan aktor.
	Association	Abstraksi dari penghubung antara aktor dengan use case.
	Include	Suatu use case termasuk bagian dari use case lain.
	Extend	Suatu use case dapat diperluas dengan use case lain.
	System	Sistem yang sedang dikembangkan.

Gambar 2.3. **Simbol-simbol pada *use case* diagram**

Di atas merupakan simbol-simbol diagram *use case* dan penjelasannya yang

biasa digunakan. *Use case* diagram akan digunakan pada penelitian ini untuk menggambarkan interaksi aktor dengan sistem yang akan dibangun pada aplikasi Android.

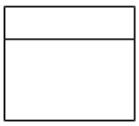
2.6 Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas, atau *activity diagram*, merupakan diagram yang dapat memodelkan proses-proses yang terjadi pada sebuah sistem. Proses dari suatu sistem akan digambarkan secara vertikal pada diagram aktivitas. Diagram ini merupakan pengembangan dari *use case* diagram yang memiliki alur aktivitas.

Activity diagram memiliki beberapa tujuan, yaitu:

1. Menjelaskan urutan aktivitas dalam suatu proses.
2. Dalam dunia bisnis biasanya digunakan untuk *modeling* (memperlihatkan urutan proses bisnis).
3. Merupakan metode perancangan yang terstruktur.
4. Mengetahui aktivitas aktor/pengguna berdasarkan diagram yang dibuat sebelumnya,

Komponen dari *activity diagram* antara lain:

Simbol	Nama	Keterangan
	Status awal	Digunakan untuk menandakan status awal, tindakan awal, atau titik awal aktivitas.
	Aktivitas	Aktivitas yang dilakukan atau sedang terjadi dalam sistem.
	State transition	Menggambarkan alur perpindahan kontrol antar state.
	Status akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
	Swimlane	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Gambar 2.4. **Komponen pada *activity diagram***

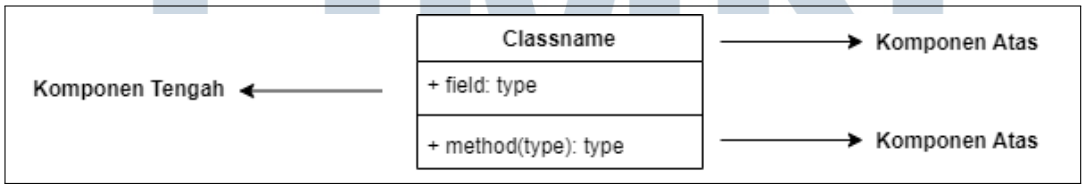
Di atas merupakan komponen yang digunakan untuk membangun suatu *activity diagram*. *Activity diagram* pada penelitian ini akan digunakan untuk menggambarkan proses-proses yang terjadi pada sebuah sistem secara vertikal.

2.7 Diagram Kelas (*Class Diagram*)

Class diagram atau diagram kelas merupakan salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi kelas, atribut, metode, dan hubungan dari setiap objek. Beberapa fungsi dari *class diagram* antara lain:

1. Menggambarkan struktur dari sebuah sistem.
2. Meningkatkan pemahaman tentang gambaran umum atau skema dari suatu program.
3. Sebagai analisis bisnis dan digunakan untuk membuat model sistem dari sisi bisnis.
4. Memberikan gambaran mengenai sistem atau perangkat lunak serta relasi-relasi yang ada di dalam sistem.

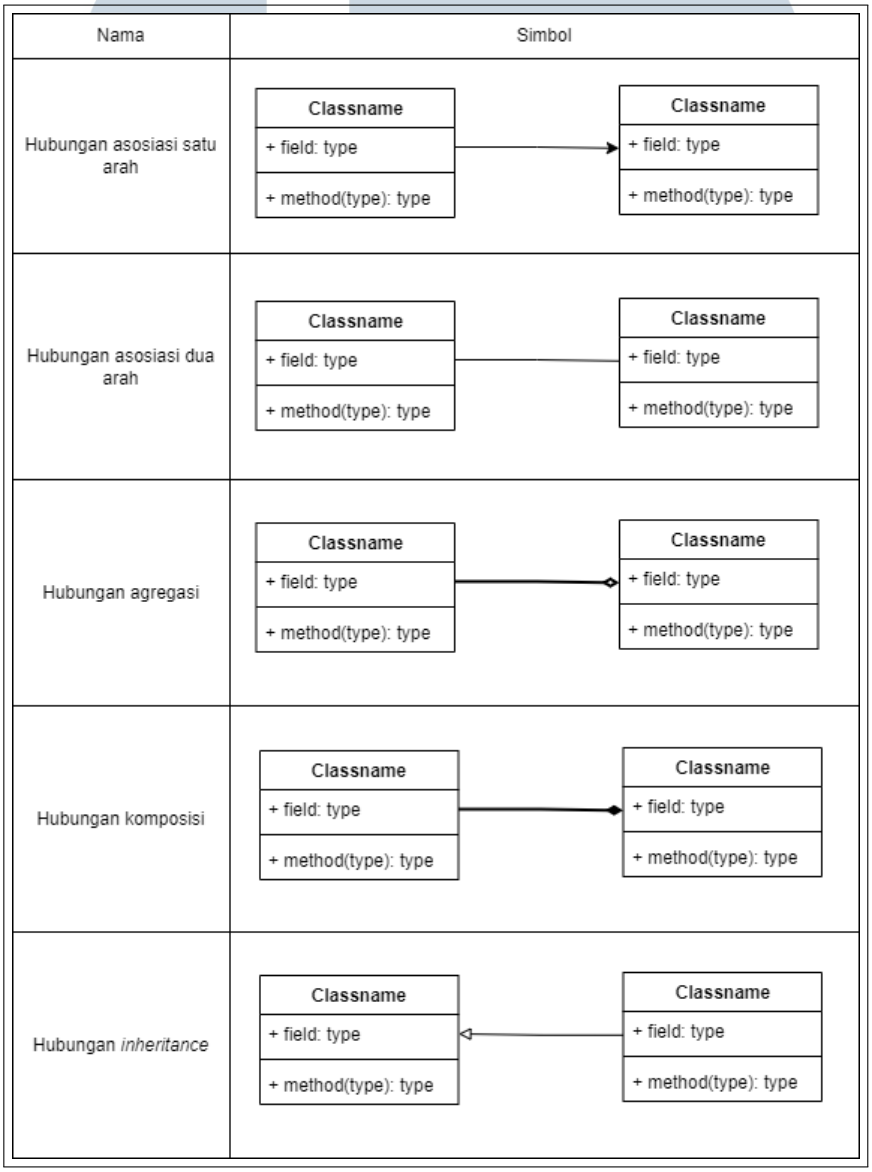
Terdapat tiga komponen penyusun dari *class diagram*, yaitu komponen atas, komponen tengah, dan komponen bawah. Komponen atas berisikan nama kelas, setiap kelas memiliki nama yang berbeda-beda. Komponen tengah berisikan atribut dari suatu kelas dan digunakan untuk menjelaskan kualitas dari suatu kelas. Pada bagian ini juga dituliskan tipe data sebagai detail suatu kelas. Komponen bawah berisi operasi yang ditampilkan dalam bentuk daftar. Operasi yang dituliskan akan menggambarkan bagaimana suatu kelas dapat berinteraksi dengan data.



Gambar 2.5. **Tiga komponen penyusun *class diagram***

Terdapat beberapa interaksi hubungan yang dapat digambarkan pada *class diagram*, yaitu hubungan asosiasi, agregasi, komposisi, dan *inheritance*. Hubungan asosiasi merupakan hubungan statis antar kelas. Biasanya digunakan untuk menggambarkan kelas yang memiliki atribut berupa kelas lain. Asosiasi dapat menggambarkan hubungan satu arah maupun dua arah. Selanjutnya terdapat hubungan agregasi, yakni hubungan antara dua kelas yang mana salah satu kelas merupakan bagian dari kelas lain. Pada hubungan ini kedua kelas dapat berdiri

masing-masing tanpa satu sama lain. Selanjutnya adalah hubungan komposisi yang menjelaskan bahwa suatu kelas merupakan bagian yang wajib dari kelas lain. Hal ini berarti keberadaan suatu kelas mempengaruhi keberadaan kelas lainnya. Dan yang terakhir adalah hubungan *inheritance*, yaitu hubungan hierarki antar kelas. Hal ini berarti suatu kelas diturunkan dari kelas lainnya dan mewarisi semua atribut dan metode kelas asalnya kemudian ditambahkan dengan fungsionalitas baru.



Gambar 2.6. **Hubungan pada *class diagram***

Dengan menggunakan *class diagram* pada penelitian ini, dapat membantu peneliti untuk menggambarkan dengan jelas struktur serta deskripsi kelas, atribut, metode, dan hubungan dari setiap objek.

2.8 Skema *Database* (*Database Schema*)

Skema *database* merupakan istilah untuk menyebut struktur atau tata letak yang menjabarkan suatu rangkaian data. Skema ini dijabarkan dalam bentuk diagram yang menjelaskan bagaimana data akan disimpan dalam *database* relasional maupun non-relasional. Skema ini menjadi landasan untuk melakukan desain DBMS (*Database Management System*) dan RDBMS (*Relational Database Management System*) seperti MySQL, PostgreSQL, dan Oracle.

Terdapat dua jenis desain skema database, yaitu:

1. Skema *Database* Fisik

   Skema ini akan menunjukkan bagaimana data diatur secara fisik dalam sistem penyimpanan. Skema *database* fisik mengubah skema logis menjadi struktur data fisik yang bisa digunakan pada DBMS tertentu. Karakteristik dari model *database* fisik antara lain:

   - Menjelaskan persyaratan data suatu proyek.
   - Semua tabel dan kolom ditentukan.
   - *Foreign key* digunakan untuk menentukan koneksi antar tabel.

2. Skema *Database* Logis

   Skema ini menjelaskan batasan atau aturan logis yang akan diterapkan pada data. Tujuan utama dari skema ini adalah memahami entitas data, termasuk relasi dan atributnya. Skema *database* logis ditampilkan dengan membuat representasi visual, yang disebut dengan diagram ER (*entity-relationship diagram*). Diagram ER secara umum menampilkan:

   - Semua entitas penting.
   - Kunci utama (*primary key*) secara khusus menjadi identitas tertentu suatu entitas.
   - Kunci tamu (*foreign key*) mendeskripsikan relasi antar entitas.

2.9 *Precision-Recall Curve*

Precision-recall curve merupakan plot *precision* sebagai fungsi *recall*. *Precision* merupakan perbandingan antara *True Positive* (TP) dengan banyaknya data yang diprediksi positif. Secara sistematis adalah sebagai berikut:

$$precision = \frac{TruePositive}{(TruePositive + FalsePositive)} \tag{2.4}$$

Sedangkan *recall* merupakan perbandingan antara *True Positive* dengan banyaknya data yang sebenarnya positif. Secara sistematis adalah sebagai berikut:

$$recall = \frac{TruePositive}{(TruePositive + FalseNegative)} \tag{2.5}$$

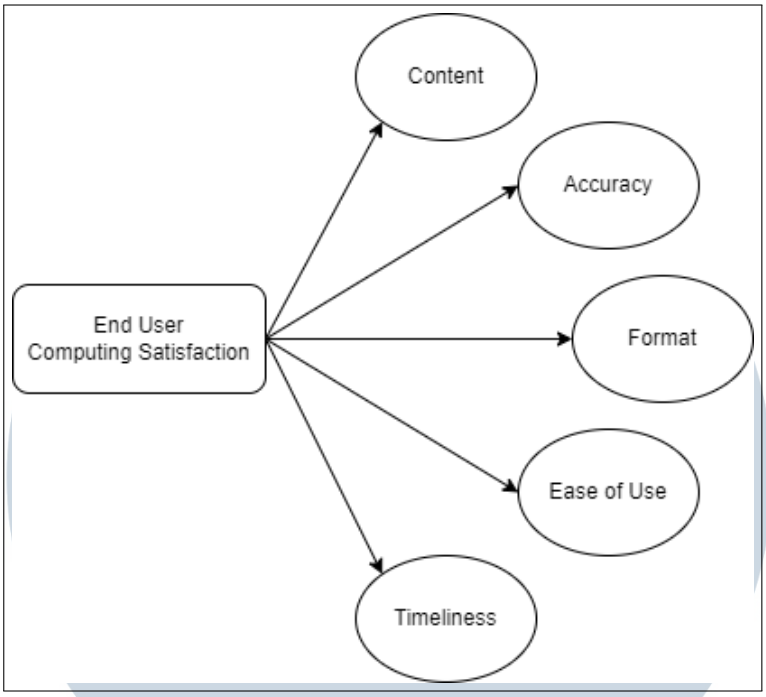
Kurva *precision-recall* menunjukkan adanya pertukaran antara dua metrik untuk berbagai nilai *confidence* pada hasil prediksi model. Hal ini berarti ketika *recall* sangat tinggi, maka *precision* akan sangat rendah, dan begitu juga sebaliknya. Selain kedua nilai tersebut, terdapat juga *f1-score* yang merupakan *harmonic mean* dari *precision* dan *recall* yang secara matematik dapat ditulis:

$$\frac{1}{F1} = \frac{1}{2}(\frac{1}{precision} + \frac{1}{recall}) \tag{2.6}$$

Secara representasi, jika *f1-score* mendapatkan nilai yang baik, maka menandakan bahwa model klasifikasi kita juga memiliki nilai *precision* dan *recall* yang baik.

2.10 EUCS (*End User Computing Satisfaction*)

EUCS merupakan metode yang digunakan untuk mengukur tingkat keputusan dari pengguna suatu sistem aplikasi dengan membandingkan antara harapan dan kenyataan dari sebuah sistem informasi. EUCS dari sebuah sistem informasi dapat didefinisikan sebagai evaluasi secara keseluruhan dari pengguna yang berdasarkan pada pengalaman mereka ketika menggunakan sistem tersebut. Evaluasi menggunakan metode ini menekankan pada kepuasan pengguna yang berdasarkan isi, keakuratan, format, waktu, dan kemudahan pengguna ketika menggunakan sistem.



Gambar 2.7. **Dimensi pada EUCS**

Pada dimensi *content*, kepuasan pengguna diukur dari sisi isi suatu sistem. Isi dari suatu sistem biasanya berupa fungsi dan modul yang dapat digunakan oleh pengguna serta informasi yang dihasilkan oleh sistem. Dimensi ini juga mengukur apakah sistem menghasilkan informasi yang sesuai dengan kebutuhan pengguna. Semakin lengkap modul dan semakin informatif sebuah sistem, maka tingkat kepuasan pengguna akan semakin tinggi.

Selanjutnya adalah dimensi *accuracy*. Dimensi ini mengukur kepuasan pengguna dari sisi keakuratan data ketika sistem menerima *input* dan mengolahnya menjadi informasi. Keakuratan sistem diukur dengan melihat seberapa sering sistem menghasilkan *output* yang salah ketika mengolah *input* dari pengguna, selain itu dapat dilihat pula seberapa sering terjadi *error* atau kesalahan dalam proses pengolahan data.

Dimensi format merupakan dimensi yang mengukur kepuasan pengguna dari sisi tampilan dan estetika antarmuka sistem. Dimensi ini menguji apakah format dari laporan atau informasi yang ditampilkan sistem memiliki antarmuka yang menarik dan memudahkan pengguna ketika menggunakan sistem. Hal ini tentunya dapat memengaruhi tingkat efektivitas pengguna.

Kemudian terdapat dimensi *ease of use* yang mengukur kepuasan pengguna dari sisi kemudahan dalam menggunakan sistem. Contohnya untuk proses

memasukkan data, mengolah data, ataupun mencari informasi yang dibutuhkan. Dimensi terakhir adalah timeliness yang mengukur kepuasan pengguna dari sisi ketepatan waktu sistem dalam menyajikan atau menyediakan data informasi yang dibutuhkan pengguna. Sistem yang tepat waktu artinya setiap permintaan atau *input* yang dilakukan oleh pengguna akan langsung diproses, selain itu *output* akan ditampilkan secara cepat tanpa harus menunggu lama.