BAB II

LANDASAN TEORI

Dalam merancang dan membangun aplikasi, sangatlah penting untuk mengetahui terlebih dahulu dasar-dasar teori yang digunakan. Dasar-dasar teori tersebut digunakan sebagai landasan berpikir dalam melakukan pembahasan lebih lanjut sehingga terbentuk suatu aplikasi yang sesuai dengan tujuan awal.

1.1 Manajemen Komplain

Dalam menerima jasa atau pelayanan sebuah perusahaan jasa ada kalanya mengalami ketidakpuasan atas layanan jasa tersebut. Ketidakpuasan tersebut dapat dinyatakan dalam bentuk pernyataan yang disebut komplain. Komplain merupakan sanggahan atau sikap menentang/menyanggah yang dinyatakan sebagai reaksi atas ketidakpuasan terhadap suatu layanan jasa. Komplain menandakan adanya perasaan kekesalan atau kekecewaan akan sesuatu yang didapatkan. Hal ini mengidentifikasikan bahwa apa yang didapat tidak sesuai dengan harapan (Kottler, 1997). Pelayanan yang diberikan oleh penyelenggara layanan, baik berupa barang atau jasa tidak semuanya mampu memberikan kepuasan bagi pelanggan. Menurut Tjiptono (2005) Manajemen Komplain adalah bentuk penanganan atau penataan, pengaturan yang dilakukan oleh suatu perusahaan dalam menyelesaikan atau mengatasi sanggahan atau reaksi ketidak puasan konsumen terhadap proses penggunaan sumberdaya organisasi,

pengoordinasian kegiatan organisasi, dan kegiatan-kegiatan fungsi manajemen yang dilakukan tidak efektif dan efisien oleh perusahaan tersebut.

Mekanisme komplain sangat diperlukan dalam penyelenggaraan pelayanan publik terutama bagi penyedia layanan untuk memperbaiki sistem pelayanan. Perbaikan sistem ini dilakukan dengan memanfaatkan respon yang diperoleh dan mengelolanya menjadi bahan pengambilan keputusan. Mekanisme komplain adalah suatu bagian dari sistem pelayanan publik untuk memfasilitasi, mengakomodasi, dan mengelola komplain konsumen atas pelayanan publik yang diterimanya. Mekanisme tersebut meliputi prosedur pengajuan, perangkat organisasi, mekanisme transparasi, media partisipasi konsumen dan perangkat pemberdayaan masyarakat (Suprapto, 2006).

Menurut Ombudsman (2010) mengemukakan bahwa suatu sistem penanganan komplain adalah cara yang terorganisasi untuk menanggapi, mencatat laporan, dan menggunakan pengaduan untuk meningkatkan layanan kepada pelanggan. Di dalamnya termasuk prosedur-prosedur bagi pelanggan untuk membuat pengaduan dan pedoman bagi karyawan untuk menyelesaikan komplain, serta menyediakan informasi kepada manajer dan staf yang dapat membantu untuk mencegah ketidakpuasan pelanggan di masa mendatang. Sistem pengolahan pengaduan yang efektif merupakan bagian penting dari sektor pelayanan publik yang berkualitas. Kebijakan dan prosedur yang baik dalam penanganan komplain mencakup informasi sebagai berikut:

a. Definisi Komplain
b. Siapa saja yang diperbolehkan mengeluh
c. Bagaimana orang bisa mengeluh

d. Penjelasan tentang proses pengaduan

e. Peninjauan ulang (baik internal maupun eksternal)

f. Kebutuhan komunikasi termasuk waktu respon

g. Keadilan dan kesetaraan persyaratan

h. Privasi dan kesetaraan persyaratan

i. Bantuan yang tersedia untuk mengajukan komplain.

Menurut Ombudsman (2010) terdapat sebelas kebijakan dan prosedur penanganan komplain efektif yang terdiri dari:

a. *Commitment*

b. *Visibility and access*

c. *Responsiveness*

d. *Fairness*

e. *Resolutions*

f. *Service improvement*

g. *Data collection*

h. *Accountability*

i. *Review*

j. *Assistance*

k. *Charges*

Sementara itu menurut Tjiptono (2008) terdapat empat aspek penting dalam menangani komplain pelanggan, yaitu empati terhadap pelanggan, kecepatan dalam penanganan komplain, kewajaran/keadilan dalam menyelesaikan komplain, serta kemudahan bagi konsumen untuk menghubungi perusahaan.

1.2 *Workflow Management System*

Menurut Chaffey (1998), *Workflow Management System* (WfMS) sebagai tipe perangkat lunak khusus yang digunakan untuk membantu kerja kolaboratif dengan komputer dan sering disebut sebagai otomatisasi *Workflow,* karena WfMS bisa mengotomatisasi tugas atau aktifitas yang dilakukan manusia atau komputer dari sebuah organisasi.

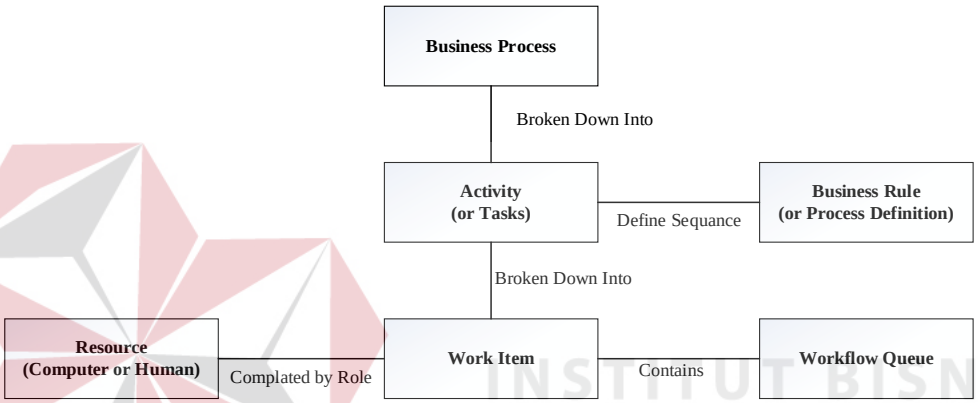
Workflow Management Coalition (WfMC) menggambarkan *workflow* sebagai fasilitas komputerisasi atau otomatisasi proses bisnis secara keseluruhan atau sebagian, sedangkan WfMS digambarkan sebagai sebuah sistem yang mendefinisikan, menciptakan, dan mengelola pelaksanaan *workflow* melalui penggunaan perangkat lunak, yang berjalan pada satu atau lebih *workflow engine*, yang mampu menafsirkan definisi proses, berinteraksi dengan peserta alur kerja dan, jika diperlukan, meminta penggunaan alat dan aplikasi TI.

Workflow dapat memberikan perbedaan yang besar pada efisiensi operasional dari proses yang ada pada sebuah bisnis. *Workflow* dapat membantu manajer dalam mengoordinasikan tugas-tugas yang dilakukan oleh staf dan memberikan informasi kepada staf untuk membantu manajer melakukan tugas-tugasnya. Keuntungan bisnis utama dari penerapan sebuah sistem *workflow* adalah waktu penyelesaian dan biaya dari proses bisnis yang ada sekarang dapat dikurangi (Chaffey, 1998).

2.2.1 Elemen Kerja Kunci dalam *Workflow System*

Sebuah *workflow* dapat digambarkan sebagai suatu hal yang terdiri atas serangkaian kegiatan, yang bersama-sama, membentuk sebuah proses bisnis. Pada Gambar 2.1 akan dijelaskan bagaimana sebuah kegiatan dipecah menjadi

workitem individu yang harus diselesaikan. Setiap *workitem* disini dilakukan oleh sebuah *resource*, baik perangkat lunak, perangkat keras, atau seorang personil yang memiliki tanggung jawab untuk melakukannya. *Work item* yang akan diselesaikan ditunjukkan pada sebuah *workflow queue*, yang adalah sebuah daftar kerja dari semua tugas yang akan diselesaikan oleh seorang individu atau sebuah tim.



Gambar 2.1 kerja kunci dalam *Workflow System* (Chaffey, 1998)

1. *Process Elements (Work Activities and Tasks)*

   Aktifitas kerja atau tugas adalah unit kerja individu yang membentuk *workflow*. Aktifitas-aktifitas ini biasanya bisa diuraikan menjadi sub tugas yang membentuk sebuah hirarki tugas. Pada saat sebuah aktifitas kerja diselesaikan, perubahan status sebuah obyek akan terjadi dan perlu dicatat oleh sistem.

2. *Resources and Their Roles*

   Resources adalah sumber daya manusia atau komputer yang melakukan aktifitas kerja yang membangun proses bisnis. *User* atau *computer resource*, yang dikenal sebagai *workflow participant* diberikan satu atau beberapa peran yang akan menentukan apakah mereka dapat melakukan tugas tertentu.

Penggunaan peran daripada individu lebih penting karena akan memudahkan untuk memindahkan tanggung jawab seseorang ke orang lain dengan peran yang sama. Pada situasi tertentu, adalah penting untuk menentukan bahwa sebuah tugas ditingkatkan pada sebuah peran yang berbeda.

3. *Dependencies and Business Rules*

   Dependencies (dependensi) menjelaskan bagaimana aktifitas yang berbeda berhubungan satu sama lain. Dependensi didefinisikan oleh *business rules* yang membangun *workflow*. Urutan dari aktifitas dapat diatur berdasarkan *pre-condition* atau *post-condition* yang harus dipenuhi sebelum mulai atau selesainya sebuah aktifitas.

4. *Workflow Queue*

   Sistem *workflow* biasanya menerapkan sebuah antrian *workflow* yang digunakan untuk menugaskan sebuah tugas ke individu. Sebuah urutan workflow akan menampung sebuah daftar tugas atau aktifitas yang harus dikerjakan dalam sebuah urutan prioritas.

5. *Case Management*

   Penggunaan dari sebuah *case* atau tiruan dari *folder* adalah sebuah hal yang umum pada sistem *workflow*. Sebuah *case* akan terdiri atas sebuah *instance* tunggal dari subyek dan obyek yang utama dari *workflow*, yaitu *customer*. Setiap *case* dapat digambarkan sebagai sebuah berkas dari sebuah lemari arsip yang menyimpan semua informasi yang berhubungan dengan *customer*.

6. *Messaging*

   Pesan tambahan dapat dikirim antara teman sekerja ketika terjadi kejadian yang tidak biasa, yang mengganggu lancarnya jalan dari sistem. Sistem mungkin

menggunakan *standard company mail system*, atau sistem *workflow* akan mengijinkan sebuah notifikasi untuk dikeluarkan atau dapat mengijinkan perubahan jalur sebuah tugas atau pencabutan sebuah tugas.

2.2.2 *Administrative Workflow System*

Administrative Workflow System adalah sebuah sistem *workflow* umum digunakan, yang memanfaatkan penggunaan form elektronik yang terhubung dengan surat elektronik. Sistem ini biasa diaplikasikan ke dalam tugas-tugas administrasi rutin seperti persetujuan pengajuan liburan, pemrosesan pemesanan pembelian, dll. The Gartner Group memperkirakan bahwa 83% dari semua dokumen bisnis di US adalah dokumen formulir dengan biaya pembelian tahunan sebesar 6-8 milyar USD dan biaya pemroresan mencapai 360 milyar USD. Formulir-formulir kertas ini menjadi target dari 1995 *Paper Reduction Act*. (Chaffey, 1998)

Manfaat yang besar dapat terjadi melalui mengotomatisasikan proses berbasis formulir. proses dapat berbalik lebih cepat menggunakan formulir elektronik dan mengurangi biaya melalui pengurangan biaya pembelian formulir dan waktu siklus yang lebih pendek. salah satu penghematan biaya terbesar adalah dalam koordinasi pengolahan formulir yang sekarang ditangani oleh logika bisnis yang dibangun ke dalam aplikasi. (Chaffey, 1998)

1.3 Perangkat Lunak (*Software*)

Perangkat lunak (*software*) adalah (1) perintah (program komputer) yang bila dieksekusi memberikan fungsi dan unjuk kerja seperti yang diinginkan. (2) Struktur data yang memungkinkan program memanipulasi informasi secara

proporsional. (3) dokumen yang menggambarkan operasi dan kegunaan program (Pressman, 2012a). Perangkat lunak tidak hanya program komputer saja, tetapi juga semua dokumentasi terkait dan data konfigurasi yang diperlukan untuk membuat program tersebut beroperasi dengan benar (Sommerville, 2001).

Menurut Pressman (2012a), perangkat lunak lebih merupakan elemen logika dan bukan merupakan elemen sistem fisik. Dengan demikian, perangkat lunak memiliki ciri yang berbeda dari perangkat keras:

1. Perangkat lunak dibangun dan dikembangkan, tidak dibuat dalam bentuk yang klasik.
2. Perangkat lunak tidak pernah usang.
3. Sebagian besar perangkat lunak dibuat secara *custom-built*, serta tidak dapat dirakit dari komponen yang sudah ada.

Terdapat 2 tipe dari produk perangkat lunak menururt Sommerville (2001), yaitu:

1. *Generic Software*

   Perangkat lunak generik adalah perangkat lunak mandiri (*stand-alone*) yang diproduksi oleh sebuah perusahaan pengembangan perangkat lunak dan dijual di pasaran secara bebas.

2. *Custom Software*

   Custom software (atau yang dipesan terlebih dahulu) adalah perangkat lunak yang dipesan oleh seorang pembeli tertentu. Perangkat lunak ini dikembangkan khusus oleh kontraktor perangkat lunak untuk pembeli tersebut.

1.4 Perangkat Keras (*Hardware*)

Perangkat keras komputer adalah alat pengolah data yang bekerja secara elektronis dan otomatis. Perangkat keras komputer dapat bekerja apabila ada unsur manusia yang mengerti tentang alat itu dan dapat bekerja menggunakan alat itu. Komputer merupakan sistem karena merupakan sekumpulan objek yang berhubungan dan bekerjasama untuk menghasilkan sesuatu yang diinginkan. (Suyanto, 2005)

Dalam buku Pengantar Teknologi untuk Bisnis, Suyanto (2005) mengelompokkan sistem perangkat keras komputer menjadi empat unsur utama dan satu unsur tambahan yaitu Unit Masukan, *Central Processing Unit* (CPU), *Storage Memory*, Unit Keluaran, dan unsur tambahan *Communication Link*. Macam-macam perangkat keras komputer yaitu:

1. Unit Masukan: *Mouse*, *Joystick*, *Trackball*, *Trackpad*, *Trackpoint*, *Touchscreen*, *Light Pen*, *Digitizing Tablet*, *Unit Remote Control*.
2. Alat Masukan Otomatisasi Data: *Optical Mark Reader* (OMR), *Optical Character Reader* (OCR), *Handheld Scanner*, *Flatbed Scanner*, *Path-through Scanner*, *Film Scanner*, *Digital Camera*, *Camcorder*, *Snappy*.
3. Unit Keluaran: *Monitor*, *Printer*, *Plotter*, *Microform*, *Microfilm*, *Microfich*, *Projector*, *Speaker*.
4. Alat Masukan Keluaran: *Poin of Sale* (POS), *Soundcard*, *MIDI Sport*, *Digital Versaitle Disc* (DVD) dan *Compact Disc* (CD).

2.5 Web

Menurut Shelly dan Vermalat (2010), *web* adalah koleksi dokumen elektronik milik semua orang di dunia yang mengaksesnya melalui internet

menggunakan *web browser*. Menurut Simamarta (2010), aplikasi *web* adalah sebuah sistem informasi yang mendukung interaksi pengguna melalui antarmuka berbasis *web*. Fitur-fitur aplikasi *web* biasanya berupa data *persistence*, mendukng transaksi dan komposisi halaman *web* dinamis yang dapat dipertimbangkan sebagai *hibridasi*, antara *hypermedia* dan sistem informasi. Aplikasi *web* adalah bagian dari *client-side* yang dapat dijalankan oleh *browser web*. *Client-side* mempunyai tanggung jawab untuk pengeksekusian proses bisnis.

Interaksi *web* menurut simamarta (2010), dibagi dalam tiga langkah utama, yaitu:

1. Permintaan

   Pengguna mengirimkan permintaan ke *server web*, biasanya *via* halaman *web* yang ditampilkan pada *browser web*.

2. Pemrosesan

   Server web menerima permintaan yang dikirimkan oleh pengguna, kemudian memproses permintaan tersebut.

3. Jawaban

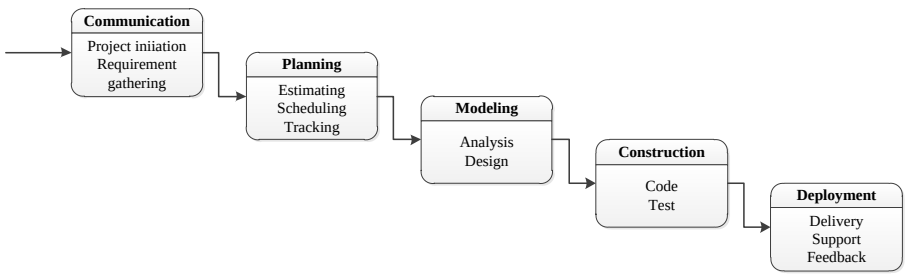
   Browser menampilkan hasil dari permintaan pada jendela *browser*.

2.6 Siklus Hidup Pengembangan Sistem

Menurut Kendall dan Kendall (2008), Siklus Hidup Pengembangan Sistem, nama lain dari *System Development Life Cycle* (SDLC) ini merupakan suatu proses pengembangan atau perubahan suatu sistem perangkat lunak. Pengembangan atau perubahan tersebut dilakukan dengan menggunakan model-model dan metodologi yang digunakan oleh banyak orang, yang telah mengembangkan sistem-sistem perangkat lunak sebelumnya. Hal tersebut tentu

berdasarkan *best practice* atau cara-cara yang telah teruji dengan baik oleh banyak orang yang menggunakannya. SDLC memiliki beberapa model dalam penerapan tahapan prosesnya. Beberapa model SDLC tersebut antara lain yaitu Model *Waterfall, Spiral, Rapid Application Development, Agile dan Prototype*. Masing-masing model memiliki kelemahan dan kelebihan, sehingga hal yang terpenting adalah mengenali tipe pelanggan dan memilih menggunakan model SDLC yang sesuai dengan karakter pelanggan dan sesuai dengan karakter pengembang perangkat lunak.

Menurut Pressman (2012b), *System Develoment Life Cycle* (SDLC) ini biasanya disebut juga dengan model *waterfall.* Menurut Pressman (2012b), nama lain dari Model *Waterfall* adalah Model Air Terjun, kadang dinamakan siklus hidup klasik (*classic life cyle*), dimana hal ini menyiratkan pendekatan yang sistematis dan berurutan (sekuensial) pada pengembangan perangkat lunak. Pengembangan perangkat lunak dimulai dari spesifikasi kebutuhan pengguna dan berlanjut melalui tahapan-tahapan perencanaan (*planning*), pemodelan (*modeling*), konstruksi (*construction*), serta penyerahan sistem perangkat lunak ke para pelanggan/pengguna (*deployment*), yang diakhiri dengan dukungan berkelanjutan pada perangkat lunak yang dihasilkan.



Gambar 2.2 Pengembangan menggunakan Model *Waterfall* (Pressman, 2012b)

Gambar 2.2 menunjukkan tahapan umum dari model proses *waterfall*. Model ini disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Akan tetapi, Pressman (2012b) memecah model ini meskipun secara garis besar sama dengan tahapan-tahapan model *waterfall* pada umumnya.

Model ini merupakan model yang paling banyak dipakai dalam *Software Engineering.* Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ketahap *Communication, Planning, Modeling, Construction, dan Deployment.*

Berikut ini adalah penjelasan dari tahap-tahap yang dilakukan di dalam Model *Waterfall* menurut Pressman (2012b):

a. *Communication*

   Langkah pertama diawali dengan komunikasi kepada konsumen/pengguna. Langkah awal ini merupakan langkah penting karena menyangkut pengumpulan informasi tentang kebutuhan konsumen/pengguna.

b. *Planning*

   Setelah proses *communication* ini, kemudian menetapkan rencana untuk pengerjaan *software* yang meliputi tugas-tugas teknis yang akan dilakukan, risiko yang mungkin terjadi, sumber yang dibutuhkan, hasil yang akan dibuat, dan jadwal pengerjaan.

c. *Modeling*

   Pada proses *modeling* ini menerjemahkan syarat kebutuhan sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat *coding*.

Proses ini berfokus pada rancangan struktur data, arsitektur *software*, representasi *interface*, dan detail (algoritma) prosedural.

d. *Construction*

Construction merupakan proses membuat kode (*code generation*). *Coding* atau pengkodean merupakan penerjemahan desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh *user*. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap sistem yang telah dibuat. Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap sistem tersebut untuk kemudian bisa diperbaiki.

e. *Deployment*

Tahapan ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan *user*. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

2.7 Testing

Menurut Romeo (2003), *testing* adalah proses pemantapan kepercayaan akan kinerja program atau sistem sebagaimana yang diharapkan. Testing *software* adalah proses mengoperasikan *software* dalam suatu kondisi yang dikendalikan untuk verifikasi, mendeteksi *error* dan validasi. Verifikasi adalah pengecekan atau pengetesan entitas-entitas, termasuk *software*, untuk pemenuhan dan konsistensi dengan melakukan evaluasi hasil terhadap kebutuhan yang telah ditetapkan. Validasi adalah melihat kebenaran sistem apakah proses yang telah ditulisan

sudah sesuai dengan apa yang dibutuhkan oleh pengguna. Deteksi *error* adalah testing yang berorientasi untuk membuat kesalahan secara intensif, untuk menentukan apakah suatu hal tersebut terjadi bilamana tidak seharusnya terjadi atau suatu hal tersebut tidak terjadi. *Test cas*e merupakan suatu tes yang dilakukan berdasarkan pada suatu inisialisasi, masukan, kondisi ataupun hasil yang telah ditentukan sebelumnya. Adapun kegunaan dari *test case* ini, adalah sebagai berikut:

1. Untuk melakukan testing kesesuaian suatu komponen terhadap desain *White Box Testing.*
2. Untuk melakukan testing kesesuaian suatu komponen terhadap spesifikasi *Black Box Testing*.

2.7.1 *White Box Testing*

Menurut Romeo (2003), *white box testing* adalah suatu metode desain *test case* yang menggunakan struktur kendali dari desain prosedural. Seringkali *white box testing* diasosiasikan dengan pengukuran cakupan tes, yang mengukur persentase jalur-jalir dari tipe yang dipilih untuk dieksekusi oleh *test cases*. *White box testing* dapat menjamin semua struktur *internal* data dapat dites untuk memastikan validitasnya.

Cakupan pernyataan, cabang dan jalur adalah suatu teknik *white box testing* yang menggunakan alur logika dari program untuk membuat *test cases* alur logika adalah cara dimana suatu bagian dari program tertentu dieksekusi saat menjalankan program. Alur logika suatu program dapat direpresentasikan dengan *flow graph*.

2.7.2 *Black Box Testing*

Menurut Romeo (2003), *black box testing* dilakukan tanpa adanya suatu pengetahuan tentang detail struktur internal dari sistem atau komponen yang dites, juga disebut sebagai *functional testing*. *Black box testing* berfokus pada kebutuhan fungsional pada *software*, berdasarkan pada spesifikasi kebutuhan dari *software*.

Dengan adanya *black box testing*, perekayasa *software* dapat menggunakan kebutuhan fungsional pada suatu program. *Black box testing* dilakukan untuk melakukan pengecekan apakah sebuah *software* telah bebas dari *error* dan fungsi-fungsi yang diperlukan telah berjalan sesuai dengan yang diharapkan.

2.8 Skala Likert

Angket atau kuisioner adalah daftar pertanyaan yang diberikan kepada orang lain yang bersedia memberikan respon, sesuai dengan permintaan pengguna. Tujuan dari menyebarkan angket adalah mencari informasi dari responden tanpa khawatir bila responden memberikan jawaban yang tidak sesuai dengan kenyataan (Riduwan, 2005).

Menurut Husein (2003), skala *likert* berhubungan dengan pernyataan seseorang terhadap sesuatu. Skor pada skala *likert* berarah *positif* dan *negatif*. Skala likert digunakan untuk mengukur sikap, pendapat, dan presepsi seseorang atau kelompok tentang kejadian atau gejala sosial.

Perhitungan skor penilaian untuk setiap pertanyaan (QS) didapatkan dari jumlah pengguna (PM) dikalikan dengan skala nilai (N). Jumlah skor tertinggi (STtot) didapatkan dari skala tertinggi (NT) dikalikan jumlah pertanyaan (Qtot) dikalikan total pengguna (Ptot). Nilai persentase akhir (Pre) diperoleh dari jumlah

skor hasil pengumpulan data (JSA) dibagi jumlah skor tertinggi (STot) dikalikan 100%. Persamaan 2.1, 2.2, dan 2.3 adalah persamaan dengan menggunakan Skala Likert.

$$QS(n) = PM \times N \quad (2.1)$$

$$STtot = NT \times Qtot \times Ptot \quad (2.2)$$

$$Pre = JSA / STtot \times 100\% \quad (2.3)$$

Keterangan:

QS(n) = Skor pertanyaan ke-n

PM = Jumlah pengguna yang menjawab

N = Skala nilai

STtot = Total skor tertinggi

NT = Skala nilai tertinggi

Qtot = Total pertanyaan

Ptot = Total pengguna

Pre = Persentase akhir (%)

JSA = Jumlah skor akhir

Analisis dilakukan dengan melihat persentase akhir dari proses perhitungan skor. Nilai persentase kemudian dicocokkan dengan kriteria interpretasi skor yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 Keterangan Nilai Skala Likert (Husein, 2003)

Nilai	Keterangan
0% – 20%	Sangat Kurang
21% – 40%	Kurang
41% – 60%	Cukup
61% – 80%	Baik
81% – 100%	Sangat Baik