BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

BAB II
DASAR TEORI

2.1. Tinjauan Pustaka

Beberapa penelitian yang relevan dilakukan oleh beberapa peneliti antara lain: penelitian dilakukan oleh Jafar Shadiq (2020) dengan judul "Sistem Informasi Kasir pada Restoran siap Saji FoodPanda Berbasis Dekstop". Penulis membuat sistem informasi yang dibutuhkan perusahaan untuk menunjang semua kegiatan bisnis agar kinerja menjadi lebih baik yaitu dengan membuat sistem informasi kasir dalam bentuk aplikasi yang bertujuan untuk mengolah data penjualan. Penulis menggunakan metode *waterfall* yang dibangun menggunakan bahasa pemrograman PHP dan *database* MySQL serta sistem ini berbasis *desktop*. Tahap pengujian menggunakan metode *blackbox*. Sistem dan informasi yang telah diproses oleh sistem akan didistribusikan oleh perusahaan agar memiliki data informasi yang valid. Sistem informasi ini masih menggunakan sistem konvensional yaitu kegiatan pemesanan menggunakan buku kertas sehingga kesalahan maupun kerusakan data bisa saja terjadi. Perancangan sistem ini membantu menangani proses bisnis perusahaan FoodPanda[3].

Penelitian sebelumnya dilakukan oleh Iskandar(2020) dengan judul "Perancangan Aplikasi Kasir *Point of Sales* Berbasis *Android* Menggunakan Metode *Rapid Application Development* untuk usaha Retail". Dalam pengujian sistemnya, penulis menggunakan metode RAD yang dibangun menggunakan bahasa pemrograman C++ dan Kotlin serta *database* MySQL dengan berbasis *android*. Tahap pengujian menggunakan metode *blackbox*. Sistem ini menghasilkan aplikasi kasir yang dapat melakukan transaksi sesuai kategori dan menghasilkan performa yang cukup baik[4].

Penelitian sebelumnya dilakukan oleh Gilang Pamungkas(2017) dengan judul "Rancang Bangun Aplikasi Kasir *Portable Android* POS (*Point Of Sale*) Yang Terintegrasi Dengan Printer Di Kafe Kantin S15 Yogyakarta". Dalam pengujian sistemnya, penulis menggunakan metode *waterfall* yang dibangun menggunakan bahasa pemrograman Java beserta *database* MySQL dengan berbasis *android*. Sistem ini menghasilkan aplikasi kasir yang dimanfaatkan untuk membantu proses transaksi penjualan dan menggantikan rekapitulasi secara manual[5].

Penelitian sebelumnya dilakukan oleh Prashant Sharma (2017) dengan judul "*RFID Based Canteen Cashier System*". Dalam pengujian sistemnya, penulis menggunakan teknologi RFID bersama dengan GUI yang dibangun menggunakan bahasa pemrograman PHP serta *database* MySQL dengan berbasis notifikasi *email*. Sistem ini memungkinkan untuk melakukan transaksi menggunakan tag RFID dan dapat melihat catatan transaksi[6].

Penelitian sebelumnya dilakukan oleh Rahma Wahdiniwaty (2020) dengan judul "*Development of cashier information system*". Dalam pengujian sistemnya, penulis menggunakan bahasa pemrograman PHP serta *database* MySQL. Sistem ini membantu kasir untuk membuat laporan penjualan dan mempermudah dalam mengetahui sistem keuangan[7].

Pada penelitian ini, penulis bermaksud membangun sistem informasi kasir berbasis *website* yang akan diterapkan di Rumah Makan Saba Mananti dengan metode *prototype* dan bahasa pemrograman PHP serta *database* MySQL. Tahap pengujian menggunakan metode *blackbox* agar dapat memberikan hasil yang lebih baik dan dapat mengetahui dari setiap langkah-langkah jalannya sistem serta berdampak baik terhadap pengembangan dan evaluasi. Pada penelitian sebelumnya sebagian berisi tentang proses transaksi, pemesanan, hingga rekapitulasi atau laporan keuangan. Pada penelitian ini sistem dapat melayani pemesanan secara *online* dan melayani pengiriman makanan. Sistem informasi ini diharapkan dapat digunakan oleh Warung Makan Sabah Mananti agar dengan mempermudah petugas kasir yang mempercepat proses pelayanan dalam bertransaksi dan memberi kemudahan dalam sistem pembukuan atau pendataan agar lebih teratur. Dengan adanya sistem informasi berbasis *website* ini, maka proses transaksi yang dilakukan lebih optimal dan pelayanan pelanggan menjadi lebih efektif dan efisien.

Tabel 2.1 Tabel Perbandingan Penelitian

Author	Judul	Metode	Permasalahan	Tujuan	Hasil
Shadiq, Jafar Wahyudin, Rayhan Lolly, Ratu (2020)	Sistem Informasi Kasir pada Restoran siap Saji Food Panda Berbasis Dekstop	*Waterfall*	Menunjang semua kegiatan bisnis agar kinerja menjadi lebih baik.	Sistem informasi ini masih menggunakan sistem konvensional yaitu kegiatan pemesanan menggunakan buku kertas sehingga kesalahan maupun kerusakan data bisa saja terjadi.	Membuat sistem informasi kasir dalam bentuk aplikasi yang bertujuan untuk mengolah data penjualan.
Iskandar, Umar Tsani Abdurrahman (2020)	Perancangan Aplikasi Kasir *Point of Sales* Berbasis Android	RAD	Melakukan transaksi penjualan produk makanan sesuai dengan kategori dan membuat laporan	Permasalahannya terjadi dalam proses rekapitulasi penjualan harus	Sistem ini menghasilkan aplikasi kasir yang dapat melakukan transaksi sesuai

Author	Judul	Metode	Permasalahan	Tujuan	Hasil
	Menggunakan *Metode Rapid Application D evelopment* Untuk usaha Retail		transaksi penjualan.	di e*ntri*-kan lagi setelah selesai penjualan hal ini akan menambah pekerjaan baru.	kategori dan menghasilkan performa yang cukup baik.
Gilang Pamungkas, Herman Yuliansyah (2017)	Rancang Bangun Aplikasi Kasir *Portable Android POS* (Point Of Sale) Yang Terintegrasi Dengan Printer Di Kafe Kantin S15 Yogyakarta	*Waterfall*	Sistem ini menghasilkan aplikasi kasir yang dimanfaatkan untuk membantu proses transaksi penjualan dan menggantikan rekapitulasi secara manual.	Kafe Kantin S15 belum memiliki sistem kasir digital atau masih menggunakan sistem manual serta rekapitulasi data transaksi penjualan dan pembelian disimpan pada file excel perhari satu file.	Hasil pengujian sistem metode unit test dari class Makanan sudah berjalan dengan lancar dan tidak ada *method* yang error sehingga dapat dinyatakan lolos dan pengujian *black box test* dapat disimpulkan bahwa aplikasi berjalan sesuai dengan apa yang telah dirancang.

Author	Judul	Metode	Permasalahan	Tujuan	Hasil
Sharma, Prashant Mathew, Bibin Nambiar, Akshay Shinde, Namrata Rai, Ratnendra (2017)	*RFID Based Canteen Cashier System*	RFID	Sistem ini dibuat dengan tujuan untuk tidak lagi bergantung pada transaksi tunai atau penanganan kupon kertas selama sebulan, melainkan memilih konsep uang digital yang dapat terstruktur dengan baik dan tersimpan dalam *database*.	Penanganan kupon selama berbulan-bulan dan mengurangi kerumunan di konter kantin.	Sistem ini mengatasi masalah penanganan kupon selama berbulan-bulan dan mengurangi kerumunan di konter kantin. Sistem ini juga akan membantu pelanggan dalam melacak makanan yang dia makan di masa lalu.
Wahdiniwati, rahma taliasih, Nurhikmah	*Development of cashier information system*	RFID	Membantu dalam perhitungan transaksi yang dilakukan sehingga dapat meminimalisir kesalahan yang terjadi.	Mengurangi antrian pelanggan dan menghemat waktu transaksi pembayaran.	Sistem informasi kasir berbasis web yang dapat membantu kasir dalam perhitungan transaksi, pembuatan catatan untuk laporan penjualan.

2.2. Landasan Teori

Dalam penelitian ini diperlukan adanya teori yang mendasar dalam menunjang proses penelitian, berikut antara lain :

2.2.1. Sistem Informasi

Sistem Informasi terdiri dari 2 bagian yaitu sistem dan informasi. Sistem adalah sekelompok komponen dan elemen untuk mencapai tujuan tertentu. Sedangkan informasi adalah sebuah data yang diolah menjadi lebih berguna dan lebih berarti bagi yang menerimanya. Sistem informasi adalah kombinasi dari *hardware, software, brainware, procedure* atau aturan yang dijalankan secara integral dalam mengelola data menjadi informasi yang bermanfaat sehingga menjadi pemecah suatu masalah ataupun dalam mengambil suatu keputusan.

Adapun pengertian sistem informasi menurut para ahli adalah sebagai berikut : Sistem informasi adalah suatu sistem yang merupakan kebutuhan pengolahan transaksi harian sehingga mendukung fungsi organisasi bersifat manajerial dalam kegiatan strategi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan laporan yang diperlukan[8].

2.2.2. Pemrograman Berorientasi Objek

Pemrograman Berorientasi Objek (OOP) adalah pemrograman yang berorientasikan kepada objek, dimana data dan fungsi dibungkus dalam *class* atau *object*. Setiap objek dapat menerima pesan, memproses data, mengirim, menyimpan dan memanipulasi data. Beberapa objek berinteraksi dengan saling memberikan informasi satu terhadap yang lainnya.

Masing-masing objek harus berisikan informasi mengenai dirinya sendiri dan dapat dihubungkan dengan objek yang lain. Pemrograman berorientasi objek berbeda dengan pemrograman prosedural yang hanya menggunakan satu halaman ke bawah untuk mengerjakan banyak perintah atau statement.

2.2.3. Basis Data

Basis data adalah kumpulan dari data yang saling terkait untuk melayani kebutuhan banyak pengguna. Penyimpanan data dengan basis data dianggap lebih baik daripada penyimpanan data dengan file karena

penyimpanan yang lebih besar, integrasi data yang membuat akses data semakin mudah.

Salah satu peran penting basis data dalam sebuah sistem informasi atau aplikasi perangkat lunak adalah sebagai sumber informasi yang mampu memenuhi kebutuhan pengguna. Sejak data relasional mendominasi *software* bisnis, perkembangan perancangan basis data mengalami kemajuan yang sangat pesat. Banyak peneliti yang menjelaskan tentang proses perbaikan basis data untuk menjaga kualitas data dan fokus pada kendala integritas. Sayangnya, masih sedikit peneliti yang membahas lebih dalam tentang penggunaan *constraint* untuk mengendalikan data yang masuk ke dalam sistem.

Constraint dapat digunakan pada semua RDBMS seperti pada MySQL, MYSQL SERVER, ORACLE, PostGreSQL. Penerapan *constraint* dalam tabel pada basis data dapat menjamin konsistensi dan integritas data karena data yang dapat masuk ke dalam tabel hanya data yang sesuai dengan batasan yang telah ditentukan. Salah satu *constraint* yang dipakai dalam basis data adalah *constraint* CHECK.

Menurut Gordon B. Davis salah satu bagian penting dalam Sistem Informasi adalah masukan (*input*), yaitu berupa data yang akan disimpan sebagai basis data. Kriteria perancangan basis data yang diperlukan untuk membuat *database* yang baik, antara lain:

1) Setiap struktur tabel harus dibuat lebih efisien dan sistematis;
2) Penyimpanan data harus dibuat efisien;
3) Ukuran tabel harus dibuat sekecil mungkin untuk mempercepat operasi di basis data;
4) Mengoptimalkan redundansi untuk meningkatkan integritas data dan meminimalisir upaya penyebaran perubahan data dari satu tabel terkait ke tabel lainnya (catatan: dalam *database* relasional, redundansi data tidak dapat dihindari);
5) Ambiguitas data di semua tabel harus dihindari.

Constraint adalah aturan dalam basis data yang tidak boleh dilanggar karena berkaitan dengan kebenaran dan konsistensi data yang menghasilkan integritas basis data. *Constraint* dalam *database* merupakan batasan nilai yang dapat memastikan hanya data yang sesuai dengan penerapan *constraint* adalah untuk menjamin integritas dan

konsistensi data dalam tabel. *Constraint* CHECK digunakan untuk meningkatkan integritas data[9].

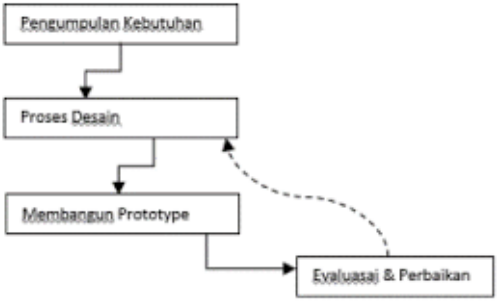
2.2.4. Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah salah satu disiplin rekayasa yang memberikan hasil abstrak. Menurut IEEE, definisi rekayasa perangkat lunak adalah aplikasi dari sebuah pendekatan disiplin dan sistematis kepada pengembangan, operasi, dan pemeliharaan perangkat lunak yaitu aplikasi dari rekayasa perangkat lunak. Model proses rekayasa perangkat lunak dipilih berdasarkan sifat aplikasi dan proyeknya, metode dan alat-alat bantu yang akan dipakai, kontrol, serta penyampaian yang dibutuhkan. Model proses untuk *software engineering* seperti model sekuensial linier atau *waterfall* model, model *prototype*, model RAD, model *incremental*, model pengembangan terpadu, model metode formal, dan model teknik generasi keempat.

2.2.5. Prototype

Metode *prototype* adalah sebuah model proses yang diterapkan saat menjalankan komunikasi dengan pelanggan untuk membuat sebuah aplikasi, *prototype* berperan penting dalam penelitian untuk memberikan gambaran aplikasi yang akurat terhadap. Pada model *prototype* ini memberikan sebuah pendekatan antara *developer* dengan pelanggan untuk terus berkomunikasi selama pembuatan aplikasi berlangsung dan *develope*r akan mendapatkan *feedback* dari pelanggan yang akan digunakan untuk memperbaiki aplikasi yang dibuat.

Tujuan dari dibuatnya *prototyping* untuk pengembangan sistem adalah mengumpulkan informasi dari pengguna sehingga pengguna dapat berinteraksi dengan model *prototype* yang dikembangkan[9]. Tahapan awal dari *prototyping* dengan pengumpulan kebutuhan yang melibatkan pengembang dan pengguna sistem menentukan tujuan, fungsi serta kebutuhan operasional sistem. Berikut adalah tahapan *prototyping* sebagai berikut:



Gambar 2.1. Metode *Prototype*

1) Tahapan pengumpulan kebutuhan yang merupakan tahap penilaian kebutuhan awal dan analisa tentang gagasan dalam mengembangkan sistem dan mengetahui komponen pada sistem yang sedang berjalan kemudian mengumpulkan informasi yang akan dikembangkan.
2) Tahapan proses desain yang tertuju pada aspek perangkat lunak Desain bertujuan mengetahui sistem akan memenuhi tujuannya dalam dibuat yang terdiri dari konsep proses dan data.
3) Tahapan membangun *prototype* membutuhkan peralatan untuk merancang sistem yang akan dibuat. Peralatan terdiri dari diagram aliran data dan diagram arus sistem.
4) Evaluasi dan Perbaikan
   Selanjutnya pada tahap ini *prototype* dievaluasi oleh pengguna pada bagian analisis desain yang digunakan untuk memenuhi kebutuhan perangkat lunak yang akan dikembangkan. *Prototype* digunakan untuk memenuhi kebutuhan pengguna serta sebagai pengembang dapat memahami lebih jelas apa saja yang perlu dilakukan.

2.2.6. Flowchart

Flowchart adalah suatu bagan yang mempunyai arus menggambarkan langkah-langkah penyelesaian suatu masalah. *Flowchart* merupakan cara penyajian dari suatu algoritma. Ada dua *flowchart* yang menggambarkan proses dengan komputer, yaitu:

a. Sistem *flowchart*

Bagan yang memperlihatkan urutan proses dalam sistem dengan menunjukkan alat media *input, output* serta jenis media penyimpanan

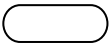
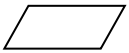
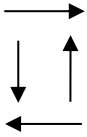
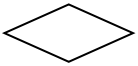
dalam proses pengolahan data.

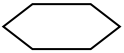
b. Program *flowchart*

Bagan yang memperlihatkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program[9].

Berikut adalah simbol-simbol yang terdapat pada *flowchart* seperti pada tabel dibawah ini:

Tabel 2.2 Simbol *Flowchart*

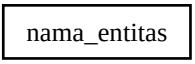
No.	Simbol	Nama	Keterangan
1.		*Terminal*	Memulai dan mengakhiri suatu program
2.		*Input/Output*	Memasukan data maupun menunjukan hasil dari suatu process tanpa tergantung dengan jenis peralatannya
3.		*Flow*	Menghubungkan antara simbol satu dengan simbol yang lain atau menyatakan jalannya arus dalam suatu proses. Simbol arus ini sering disebut juga dengan connecting line.
4.		*Decision*	Memilih proses berdasarkan kondisi yang ada.
5.		*Document*	Merupakan simbol untuk data yang terbentuk informasi.
6.		*Manual Operation*	Menunjukkan pengolahan yang tidak dilakukan oleh komputer/pc.
7.		*Punched Card*	Menyatakan input berasal dari kartu atau output ditulis ke kartu.

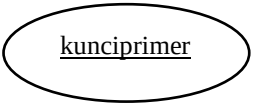
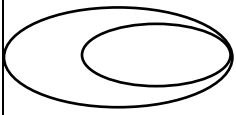
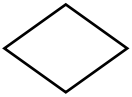
No.	Simbol	Nama	Keterangan
8.		*Connector*	Menyatakan sambungan dari suatu proses ke proses lainnya dalam halaman / lembar yang sama.
9.		*Offline Connector*	Menyatakan sambungan dari satu proses ke proses lainnya dalam halaman atau lembar yang berbeda.
10.		*Predefined Process*	Menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.

2.2.7. Entity Relationship Diagram (ERD)

ERD (*Entity Relationship Diagram*) adalah diagram yang digunakan untuk pemodelan basis data relasional. Penyimpanan basis data ketika menggunakan OODBMS (*Object-Oriented Database Management System*) maka perancangan basis data tidak perlu menggunakan ERD.

Tabel 2.3. Simbol *ERD*

No.	Simbol	Nama	Keterangan
1.		Entitas / *Entity*	Persegi panjang yang mewakili sekumpulan atau himpunan objek yang berada pada sistem.
2.		Atribut	*Field* atau kolom data yang perlu disimpan dalam entitas.

No.	Simbol	Nama	Keterangan
3.		Atribut kunci primer	*Field* atau kolom data yang disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan.
4.		Atribut multinilai	*Field* yang butuh disimpan dalam entitas yang dapat memiliki nilai lebih dari satu.
5.		Relasi	Relasi yang menghubungkan antar entitas, biasanya diawali dengan kata kerja.
6.		*Link*	Garis yang mewakili hubungan antara atribut dengan entitas dan himpunan entitas dengan entitas dan sebaliknya.

2.2.8. Prototype

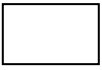
2.2.9. Unified Modelling Language (UML)

UML (*Unified Modeling Language*) adalah bahasa untuk memodelkan perangkat lunak yang akan dibuat. *Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek[10].

a) ***Usecase Diagram***

Usecase diagram digunakan untuk menggambarkan sistem dari sudut pandang pengguna sistem tersebut (*user*). Sehingga pembuatan *usecase diagram* lebih dititik beratkan pada fungsionalitas yang ada pada sistem, bukan berdasarkan alur atau urutan kejadian. Sebuah *usecase diagram* mempresentasikan sebuah interaksi antara aktor dengan sistem.

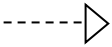
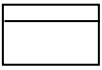
Tabel 2.4 Simbol *Usecase Diagram*

No.	Simbol	Nama	Keterangan
1.		*Use case*	Deskripsi dari urutan aksi – aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.
2.		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
3.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
4.		*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit.
5.		*Extend*	Menspesifikasikan bahwa *use case* tar*get* memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas

b) ***Class Diagram***

Class diagram adalah inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut atau properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode atau fungsi). *Class* memiliki tiga area pokok yaitu nama (*class name*), atribut, dan metode (*operation*).

Tabel 2.5 Simbol *Class Diagram*

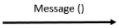
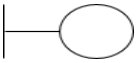
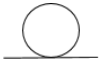
No.	Simbol	Nama	Keterangan
1.		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
2.		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
3.		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
4.		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
5.		*Realization*	Operasi yang bener-benar dilakukan oleh suatu objek.
6.		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.

c) *Sequence Diagram*

Sequence diagram merupakan penggambaran interaksi antara sejumlah objek dalam urutan waktu. Kegunaannya untuk menunjukan

rangkaian pesan yang dikirim antara objek juga interaksi antar objek yang terjadi pada titik tertentu dalam eksekusi sistem.

Tabel 2.6 Simbol *Sequence Diagram*

No.	Simbol	Nama	Keterangan
1.		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2.		*Actor*	Menggambarkan *user* atau pengguna.
3.		*Message*	Spesifikasi dari komunikasi antar *objek* yang memuat informasi – informasi tentang aktivitas yang terjadi.
4.		*Boundary*	Menggambarkan sebuah *form*.
5.		*Control Class*	Menghubungkan *boundary* dengan tabel.
6.		*Entity Class*	Menggambarkan hubungan kegiatan yang akan dilakukan.

2.2.10. Black Box

Black box adalah pengujian perangkat lunak dari segi fungsional tanpa melakukan pengujian desain dan program. Pengujian ini dilakukan untuk mengetahui apakah perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian mengacu saat memasukkan data

di *form* aplikasi dan digabungkan berdasarkan fungsi pengujian[11]. Metode ini bertujuan untuk menunjukkan suatu kesalahan pada sistem. Teknik pada pengujian *blackbox* bermacam-macam diantaranya adalah sebagai berikut :

a. *Teknik Equivalence* yaitu melakukan pembagian menjadi beberapa bagian dalam menginput data.
b. *Teknik Boundary Value Analysis* yaitu mencari kesalahan dari *software*.
c. *Teknik Fuzzing* yaitu mencari gangguan dari *software* dengan menggunakan data.
d. *Teknik Cause-Effect* yaitu menggunakan suatu grafik sebagai acuannya.
e. *Teknik Orthogonal Array Testing* yaitu untuk menginputkan data dalam skala besar.
f. *Teknik All Pair Testing* yaitu mengeksekusi data berdasarkan input parameternya.
g. *Teknik state Transition* yaitu melakukan pengujian pada mesin dengan metode grafik[2].

2.2.11. Skala Likert

Skala Likert yaitu metode perhitungan untuk keperluan riset atau jawaban seorang responden terhadap suatu pertanyaan. Setiap pertanyaan terdapat lima alternatif jawaban dengan bobot yang mengacu pada teknik Skala Likert. Skala Likert menggunakan dua bentuk pertanyaan yaitu pertanyaan positif dan pertanyaan negatif.

Pertanyaan positif diberi skor 5,4,3,2,1. Sedangkan pertanyaan negatif diberikan skor 1,2,3,4,5. Bentuk jawaban dari Skala Likert terdiri dari sangat setuju, setuju, kurang setuju, tidak setuju, dan sangat kurang setuju. Perhitungan atau hasil dari Skala Likert diperoleh dengan menentukan skor maksimal atau minimal dikalikan dengan banyaknya pertanyaan. Selanjutnya, membuat tabel distribusi frekuensi sikap setiap responden[12].