BAB 2
TINJAUAN PUSTAKA

2.1 Profil Toko Gawa Sejahtera

Profil Toko Gawa Sejahtera meliputi sejarah berdirinya, tempat kedudukan, bentuk dan badan hukum perusahaan serta struktur organisasi.

2.1.1 Sejarah Perusahaan

Toko Gawa Sejahtera adalah sebuah perusahaan yang bergerak di bidang penjualan bahan bangunan yang beralamat di Jl. Raya Selajambe RT006/002 dusun Cipicung Desa Padahurip Kecamatan Selajambe Kabupaten Kuningan. Perusahaan ini mulai didirikan oleh H.Rudi Salam dan Cicin Kuraesin pada tahun 1991. Pada mulanya, Toko Gawa Sejahtera merupakan toko dengan ketersediaan bahan bangunan kurang lengkap. Dalam perkembangannya, toko bangunan ini berkembang seiring permintaan dan jumlah konsumen atau pelanggan yang terus meningkat dari tahun ketahun.

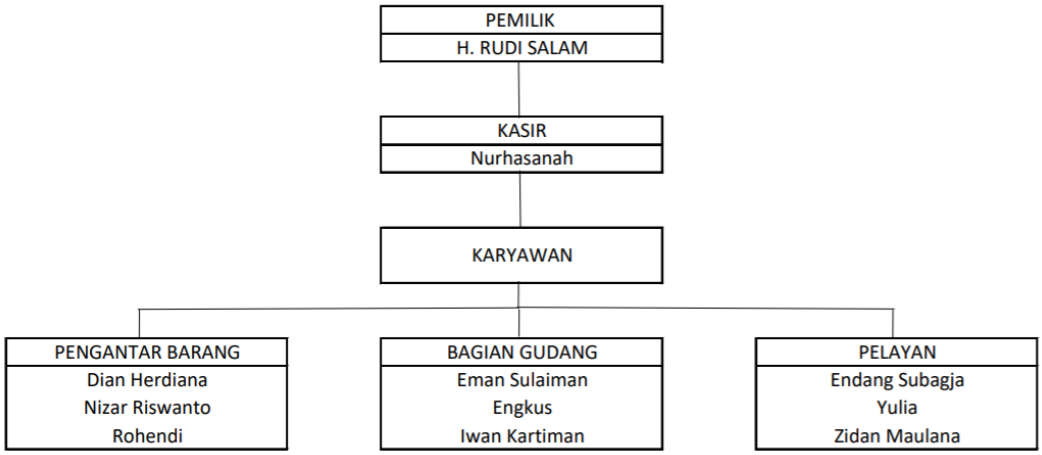
2.1.2 Bentuk dan Badan Hukum Perusahaan

Toko Gawa Sejahtera berdiri berdasarkan Ijin Pemerintah Dengan NPWP : 72.675.612.5-438.000 dan TDP : 101714900241

2.1.3 Struktur Organisasai

Dalam sebuah organisasi agar semua kegiatan berjalan dengan baik dan dapat mencapai tujuan, perlu adanya suatu struktur organisasi dan pembagian kerja (job description) yang jelas. Struktur organisasi yang baik harus menggambarkan dengan jelas wewenang dan tanggung jawab serta fungsi-fungsi dari setiap bagian yang ada dalam perusahaan, yang mana dalam hal ini merupakan salah satu syarat terciptanya suatu pengendalian internal yang memadai. Adapun struktur organisasi

diantaranya:



Gambar 2.1 Struktur Organisasi Toko Gawa Sejahtera

2.1.4 Deskripsi Kerja Bagian-bagian di Toko Gawa Sejahtera

Untuk melengkapi struktur organisasi suatu perusahaan, diperlukan uraian tugas yang akan menjelaskan tentang wewenang dan tanggung jawab dari masing-masing fungsi dalam perusahaan. Uraian tugas pada Toko Gawa Sejahtera adalah sebagai berikut :

1. Pemilik tugasnya adalah sebagai berikut :
   a) Memimpin kegiatan toko secara keseluruhan
   b) Mengelola seluruh karyawan
   c) Menerima laporan penjualan dan pembelian.
   d) Mengatur keuangan toko
   e) Pengatur gaji karyawan
2. Bagian kasir uraian tugasnya adalah sebagai berikut :
   a) Melayani pelanggan
   b) Mengelola transaksi penjualan dan pembelian.
   c) Membantu pemilik dalam membuat laporan keuangan
3. Karyawan tugasnya adalah sebagai berikut :

a) Melaksanakan tugas dari atasan sesuai bagian atau perintah dari pemilik toko / atasan
b) Pengantar barang / sopir tugasnya adalah mengantar pesanan barang ke pelanggan
c) Pengangkut barang tugasnya adalah muat bongkar barang
d) Pelayan tugasnya adalah melayani kebutuhan pelanggan dan merapihkan barang

2.2 Landasan Teori

Landasan Teori bertujuan memberikan gambaran dari teori yang terkait dalam pembangunan aplikasi. Landasan teori yang diuraikan merupakan hasil studi literatur, buku-buku, maupun situs internet.

2.2.1 Aplikasi

Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna. Aplikasi merupakan rangkaian kegiatan atau perintah untuk dieksekusi oleh komputer. Program merupakan kumpulan instruction set yang akan dijalankan oleh pemroses, yaitu berupa software. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna. Aplikasi merupakan rangkaian kegiatan atau perintah untuk dieksekusi oleh komputer. Program merupakan kumpulan instruction set yang akan dijalankan oleh pemroses, yaitu berupa software [7].

Program inilah yang mengendalikan semua aktifitas yang ada pada pemroses. Program berisi konstruksi logika yang dibuat oleh manusia, dan sudah diterjemahkan ke dalam bahasa mesin sesuai dengan format yang ada pada instructionset. Program aplikasi merupakan program siap pakai. Program yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain. Contoh-contoh aplikasiialah program pemproses kata dan Web Browser [7]. Aplikasi akan menggunakan sistemoperasi (OS) komputer dan aplikasi yang lainnya yang mendukung.Istilah ini

mulai perlahan masuk ke dalam istilah Teknologi Informasi semenjak tahun 1993, yang biasanya juga disingkat dengan app. Secara historis, aplikasi adalah software yang dikembangkan oleh sebuah perusahaan. App adalah sofware yang dibeli perusahaan dari tempat pembuatnya. Industri PC tampaknya menciptakan istilah ini untuk merefleksikan medan pertempuran persaingan yang baru, yang paralel dengan yang terjadi antar sistem operasi yang dimunculkan [7].

2.2.1 Klasifikasi Aplikasi

Aplikasi dapat digolongkan menjadi beberapa kelas, antara lain:

1. Perangkat lunak perusahaan (enterprise)
2. Perangkat lunak infrastruktur perusahaan
3. Perangkat lunak informasi kerja
4. Perangkat lunak media dan hiburan
5. Perangkat lunak pendidikan
6. Perangkat lunak pengembangan media
7. Perangkat lunak rekayasa produk

2.2.2 Android

Sejarah *Android* pada mulanya berasal dari perusahaan bernama Android, Inc. didirikan tempatnya di Palo Alto, California, pada Oktober tahun 2003 oleh Andy Rubin (pendiri Danger), Rich Miner seorang pendiri *Wildfire Communications,* Inc., Nick Sears seorang mantan VP T-Mobile, dan Chris White seorang kepala desain dan pengembangan antarmuka Web TV untuk mengembangkan mengembangkan sebuah "perangkat seluler pintar yang lebih sadar tentang lokasi dan preferensi penggunanya"[6]. Logo *Android* bisa di lihat pada gambar 2.2

Gambar 2.2 Logo Android.

Aplikasi Android ditulis dalam bahasa pemrograman Java. The Android SDK alat mengkompilasi kode-bersama dengan data dan sumber daya file-ke dalam paket Android, sebuah file arsip dengan akhiran. Semua kode dalam satu file dianggap sebagai salah satu aplikasi dan file yang Android-powered perangkat gunakan untuk menginstal aplikasi. Setelah diinstal pada perangkat, setiap aplikasi Android hidup dalam keamanan sandbox sendiri:

1. Sistem operasi Android adalah sistem multi-user Linux di mana setiap aplikasi adalah pengguna yang berbeda.
2. Secara default, sistem menugaskan setiap aplikasi Linux ID pengguna yang unik (ID hanya digunakan oleh sistem dan tidak diketahui aplikasi). Sistem ini menetapkan hak akses untuk semua file dalam aplikasi sehingga hanya user ID ditugaskan untuk aplikasi yang dapat mengaksesnya.
3. Setiap proses memiliki mesin virtual sendiri (VM), sehingga kode aplikasi berjalan secara terpisah dari aplikasi lain.
4. Secara default, setiap aplikasi berjalan dalam prosesnya sendiri Linux. Android memulai proses ketika salah satu komponen aplikasi perlu dijalankan, maka menutup proses ketika itu tidak lagi dibutuhkan atau ketika sistem harus memulihkan memori untuk aplikasi lain.

Dengan cara ini, sistem Android menerapkan prinsip paling istimewa. Artinya, setiap aplikasi, secara default, hanya memiliki akses ke komponen yang diperlukan

untuk melakukan pekerjaan dan tidak lebih. Ini menciptakan lingkungan yang sangat aman di mana aplikasi tidak dapat mengakses bagian dari sistem yang tidak diberikan izin. Adapun versi-versi *Android* yang pernah dirilis adalah sebagai berikut:

1. *Android* Versi 1.0 (*Astro*)
2. *Android* Versi 1.1 (Bender)
3. *Android* Versi 1.5 (*Cupcake)*
4. *Android* Versi 1.6 (*Donut*)
5. *Android* Versi 2.0/2.1 (*Eclair*)
6. *Android* Versi 2.2 (*Froyo*)
7. *Android* Versi 2.3 (*Gingerbread*)
8. *Android* Versi 3.0 (*Honeycomb*)
9. *Android* Versi 4.0 (*Ice Cream Sandwich*)
10. *Android* Versi 4.1 (*Jelly Bean*)
11. *Android* Versi 4.4 (*KitKat*)
12. *Android* Versi 5.0 (*Lollipop*)
13. *Android* Versi 6.0 (*Marshmallow*)
14. *Android* Versi 7.0 (*Nougat*)
15. *Android* Versi 8.0 (*Oreo*)
16. *Android* Versi 9.0 (*Pie*)

2.2.2.1 Arsitektur Android

Arsitektur *Android* Google menggambarkan *Android* seperti sebuah tumpukan *software*. Setiap lapisan dari tumpukan ini terdiri dari beberapa program yang mendukung fungsi-fungsi spesifik dari sistem operasi. Dalam paket sistem operasi *Android* tediri dari beberapa unsur seperti tampak pada gambar 2.6. Secara sederhana arsitektur *Android* merupakan sebuah *kernel Linux* dan sekumpulan pustaka C / C++ dalam suatu *Framework* yang menyediakan dan mengatur alur proses aplikasi[6].

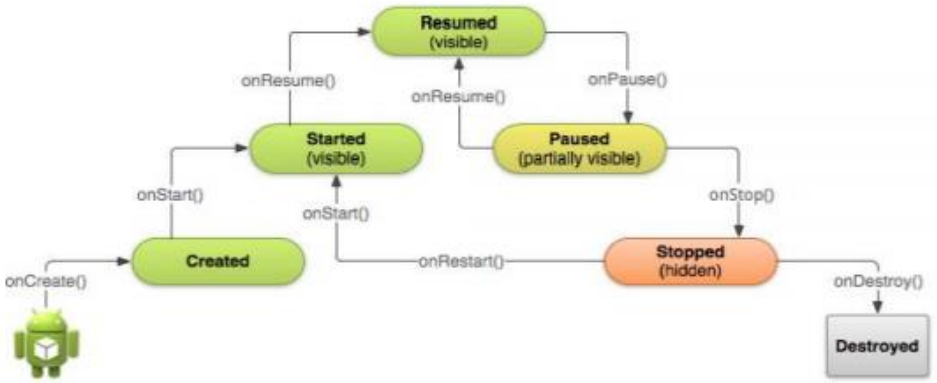
2.2.2.2 Android Life Cycle

Aplikasi *Android* terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar *file* musik, membunyikan alarm, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen *Android* yang berbeda seperti ditunjukkan pada, klasifikasi tersebut berdasarkan kelas- kelas dasar *java* yang digunakan

Gambar 2.3 Kompenen Android.

Setiap aplikasi pasti menggunakan minimal satu dari komponen tersebut akan, tetapi terdapat beberapa komponen yang mengharuskan mencantumkan *specified permission* sebelum digunakan seperti komponen Service, *Broadcast Receiver*. *Content Provider*[18].

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa di mana aplikasi yang dijalankan pada fungsi main(), sistem *Android* menjalankan kode dalam *method Activity* dengan menerapkan metode *callback* tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi *Android* memiliki siklus hidup yang berbeda dengan aplikasi desktop atau web. Hal ini dikarenakan aplikasi *mobile* memiliki tingkat interupsi proses yang lumayan tinggi seperti ketika *handling* panggilan masuk aplikasi diharuskan menghentikan proses sementara. Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna[18].



Gambar 2.4 Siklus Hidup Android.

Terdapat beberapa state dalam siklus hidup *Android* yang terjadi. Siklus Hidup *Android* akan tetapi hanya beberapa dari state tersebut yang menjadi Statis diantaranya :

1. *Resumed*

   Resumed terjadi ketika aplikasi berjalan setelah *state paused* . State ini akan menjalankan perintah program yang ditulis pada *method onResume()*[18].

2. *Paused*

   Dalam keadaan ini aktivitas yang terjadi dihentikan secara sementara tetapi masih terlihat oleh pengguna karena terdapat proses yang memiliki prioritas lebih tinggi seperti panggilan telepon. Aplikasi tidak dapat menjalankan perintah apa pun ataupun menampilkan apa pun dalam Langkah ini[18].

3. *Stopped*

   Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan *service* di *background*[18].

State lain seperti *Created* dan *Started* bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cycle callback* berikutnya. Artinya, setelah sistem *OnCreate()* dipanggil, dengan cepat sistem akan memanggil *method OnStart()*, krmudian diikuti oleh *onResume()*[18].

Pada penelitian ini versi *Android* yang digunakan adalah *Android*versi 5.0

(nougat) karena mempunyai material merupakan hasil desain terbaru sabagai spesifikasi minimal cocok utnuk pengembangan aplikasi karena sudah mempunyai hasil desain *Android* terbaru selain itu. Versi mempunyai kelebihan dalam Peningkatan pada pencarian kontekstual sehingga bisa menentukan lokasi dengan tepat ke tempat aplikasi tertentu dan akan mulai secara tepat. Secara Garis besar arsitektur *Android* dapat dijalankan dan digambarkan sebagai berikut : *Application* dan *Widgets*. Application dan Widgets adalah layer di mana *user* berhubungan dengan aplikasi saja, di mana biasanya *user* men-download aplikasi, melakukan instalasi dan menjalankan aplikasi.

1. *Framework*

Android adalah "Open Development Platform" yaitu *Android* menawarkan kepada pengembang atau member kemampuan untuk membangun aplikasi yang inovatif. Pegembang bebas untu mengakses perangkat keras, akses informasi *resource*, menjalankan *service background*, mengatur alarm, dan menambahkan status notifikasi. Komponen-komponen yang termasuk di dalam *applications frameworks*
adalah sebagai berikut:

a. *Views*
b. Content provider
c. Resource manager
d. Notification manager
e. Activity manager

2. *Libraries*

Libraries adalah layer dimana fitur-fitur *Android* berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya.

3. *Android Runtime*

Layer yang membuat aplikasi *Android* dapat djalankan dimana dalam prosesnya menggunakan Implementasi *Linux. Dalvik virtual machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi *Android*.

4. *Linux Kernel*

Linux Kernel adalah layer di mana inti dari *operating sistem* dari *Android* itu berada. Berisi *file-file sistem* yang mengatur sistem *processing, memory*, *recource*, *drivers*, dan sistem-sistem operasi *Android* lainnya[18].

2.2.3 API (Application Programming Interface)

Application Programing Interface (API) *Android* API adalah Seperangkat fungsi standar yang disediakan oleh OS atau Bahasa. Dalam Java, API dimasukkan ke dalam *package-package* yang sesuai dengan fungsinya. Berikut adalah beberapa API utama yang disediakan oleh *Android*, yaitu API untuk manipulasi *Graphical User Interface* (GUI), akses *storage*, manipulasi grafik, akses *location based service*, dan manipulasi peta.

1. *Graphical User Interface* (GUI) *Package android.view* menyediakan berbagai kelas-kelas yang akan digunakan untuk menangani *screen*, *layout*, dan interaksinya dengan pengguna.
2. Akses *Storage Android* menggunakan mekanisme storage yang berbeda dengan sistem operasi yang konvensional dimana setiap file dalam *Android* bersifat *private* terhadap aplikasi tersebut.
3. Manipulasi Grafik *Package android.graphics* menyediakan manipulasi grafik *low-level* seperti kanvas, point, pewarnaan, dan manipulasi bentuk pada *screen*.

2.2.4 RESTful Web Service

REST (*Representational State Transfer*) merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (*Hypertext Transfer Protocol*) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000.

Pada arsitektur REST, REST server menyediakan resources (sumber daya/data) dan REST client mengakses dan menampilkan resource tersebut untuk penggunaan selanjutnya. Setiap resource diidentifikasi oleh URIs (*Universal Resource Identifiers*) atau global ID. Resource tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML. Berikut adalah keuntungan dari REST :

1. Bahasa dan platform agnostic
2. Lebih sederhana/simpel untuk dikembangkan ketimbang SOAP
3. Mudah dipelajari, tidak bergantung pada tools
4. Ringkas, tidak membutuhkan layer pertukaran pesan (messaging) tambahan
5. Secara desain dan filosofi lebih dekat dengan web

Sedangkan untuk kelemahan dari REST adalah sebaga berikut ini :

1. Mengasumsi model point-to-point komunikasi - tidak dapat digunakan untuk lingkungan komputasi terdistribusi di mana pesan akan melalui satu atau lebih perantara
2. Kurangnya dukungan standar untuk keamanan, kebijakan, keandalan pesan, dll, sehingga layanan yang mempunyai persyaratan lebih canggih lebih sulit untuk dikembangkan ("dipecahkan sendiri")
3. Berkaitan dengan model transport HTTP Berikut metode HTTP yang umum digunakan dalam arsitektur berbasis REST:

1. GET, menyediakan hanya akses baca pada resource
2. PUT, digunakan untuk menciptakan resource baru
3. DELETE, digunakan untuk menghapus resource
4. POST, digunakan untuk memperbarui resource yang ada atau membuat resource baru
5. OPTIONS, digunakan untuk mendapatkan operasi yang disupport pada resource Web service adalah standar yang digunakan untuk melakukan pertukaran data antar aplikasi atau sistem, karena aplikasi yang melakukan

pertukaran data bisa ditulis dengan bahasa pemrograman yang berbeda atau berjalan pada platform yang berbeda. Contoh implementasi dari web service antara lain adalah SOAP dan REST. Web service yang berbasis arsitektur REST kemudian dikenal sebagai RESTful web services. Layanan web ini menggunakan metode HTTP untuk menerapkan konsep arsitektur REST.

2.2.4.1 Cara Kerja RESTful Web Service

Sebuah client mengirimkan sebuah data atau request melalui HTTP Request dan kemudian server merespon melalui HTTP Response. Komponen dari http request :

1. Verb, HTTP method yang digunakan misalnya GET, POST, DELETE, PUT dll.
2. Uniform Resource Identifier (URI) untuk mengidentifikasikan lokasi resource pada server.
3. HTTP Version, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
4. Request Header, berisi metadata untuk HTTP Request. Contoh, type client/browser, format yang didukung oleh client, format dari body pesan, seting cache dll.
5. Request Body, konten dari data.

Sedangkan komponen dari http response :

1. Status/Response Code, mengindikasikan status server terhadap resource yang direquest. misal : 404, artinya resource tidak ditemukan dan 200 response OK.
2. HTTP Version, menunjukkan versi dari HTTP yang digunakan.
3. Response Header, berisi metadata untuk HTTP Response. Contoh, type server, panjang content, tipe content, waktu response, dll
4. Response Body, konten dari data yang diberikan.

2.2.5 Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai computer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsistems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis Java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagi mesin virtual java (JVM).

Java merupakan bahasa pemrograman yang bersifat umum/nonspesifik (general purpose), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Java merupakan bahasa berorientasi objek (OOP) yaitu cara ampuh dalam pengorganisasian dan pengembangan perangkat lunak. Pada OOP, program computer sebagai kelompok objek yang saling berinteraksi. Deskripsi OOP secara ringkas adalah mengorganisasikan program sebagai kumpulan komponen yang disebut objek. Objek-objek ini ada secara independen, mempunyai aturan-aturan berkomunikasi dengan objek lain dan untuk memerintahkan objek lain guna meminta informasi tertentu atau meminta objek lain mengerjakan sesuatu

2.2.6 Mysql

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan DBMS (*Database Management Sistem*), database ini *multithread*, *multi-user*. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus yang bersifat khusus.

Kekuatan MySQL tidak ditopang oleh sebuah komunitas, seperti Apache, yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh pemilik masing-masing, tetapi MySQL didukung penuh oleh sebuah perusahaan profesional dan komersial, yakni MySQL AB dari Swedia.

MySQL adalah *Relational Database Management Sistem* (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Publi License*). Di mana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat closed source atau komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan mudah secara otomatis

Berikut adalah contoh sintaks sederhana pada MySQL:

1. Membuat database

```
CREATE DATABASE rental;
```

2. Melihat database

```
SHOW DATABASES;
```

3. Membuat tabel

```
CREATE TABLE customer (
        id_customer VARCHAR (8) NOT NULL,
        nama_customer VARCHAR (20) NOT NULL,
        tgl_rental DATETIME NOT NULL,
        tgl_kembali DATETIME NOT NULL,
        PRIMARY KEY (id)
);
```

4. Melihat tabel

```
SHOW TABLES;
```

5. Menghapus tabel

```
DROP TABLE customer;
```

6. Menghapus database

```
DROP DATABASE rental;
```

Dalam menggunakan *database* MySQL, setiap perintah yang diketikkan disebut *query*. Perintah MySQL dapat dikategorikan menjadi 3 sub perintah, yaitu DDL (*Data Definition* Language), DML (*Data Manipulation Language*), dan DCL (*Data Control Language*). Berikut adalah pemaparan dari setiap kategori:

1. DDL (*Data Definition Language*)

   Data Definition Language yang kalau di singkat DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur database, dalam hal ini database dan table. Beberapa perintah dasar yan termasuk DDL ini antara lain :

   a. CREATE berfungsi untuk membuat database baru, tabel baru, view baru dan kolom.
   b. ALTER berfungsi untuk mengubah struktur tabel. Seperti mengganti nama tabel, menambah kolom, mengubah kolom, menghapus kolom maupun memberikan atribut pada kolom.
   c. DROP berfungsi untuk menghapus database dan tabel.
   d. TRUNCATE berfungsi untuk menghapus semua catatan dari tabel.
   e. COMMENT berfungsi untuk menambahkan komentar pada data.
   f. RENAME berfungsi untuk mengubah nama objek.

2. DML (*Data Manipulation Language*)

   Kategori selanjutnya adalah Data Manipulation Language (DML). DML ialah perintah yang digunakan untuk mengelola/memanipulasi data dalam database. Terdapat beberapa perintah DML pada MySQL sebagai berikut :

   a. SELECT berfungsi untuk mengambil/menampilkan data dari database.
   b. INSERT berfungsi untuk memasukkan data ke dalam tabel.
   c. UPDATE berfungsi untuk memperbarui data dalam tabel.
   d. DELETE berfungsi untuk menghapus data dari tabel.
   e. CALL berfunsi untuk memanggil subprogram PL / SQL Java.

f. EXPLAIN PLAN berfungsi untuk menjelaskan jalur akses data.
g. LOCK TABLE berfungsi untuk mengunci tabel.

3. DCL (*Data Control Language*)

Kategori terakhir adalah Data Control Language (DCL). DCL ialah perintah yang digunakan untuk melakukan pengontrolan data dan server databasenya. Terdapat beberapa perintah DCL pada MySQL sebagai berikut :

a. GRANT berfungsi untuk memberikan hak akses pengguna ke database.
b. REVOKE berfungsi untuk menghilangkan hak akses yang telah diberikan dengan perintah GRANT.

2.2.7 Firebase Cloud Messaging

FCM adalah sebuah layanan yang digunakan untuk melakukan pemberitahuan (notifications) pada aplikasi berbasis Android, iOS maupun aplikasi web. Dahulunya Firebase Cloud Messaging ini bernama Google Cloud Messaging atau GCM, namun sekarang sudah berubah dan menjadi lebih besar di Firebase. Langkah utama untuk mengimplementasikan FCM di Android adalah membuat project di Firebase dan meng integrasikannya dengan aplikasi Android.

Langkah langkah yang diperlukan adalah :

1 Membuat akun atau project console di Firebase Console, Lalu Create New Project atau buatlah project baru, beri nama sesuai keperluan Anda.
2 Setelah masuk dashboard, lalu carilah tombol Add Firebase to yout Android app dan ikuti saja langkahnya (masukan nama namespace dari aplikasi anda, lalu generate dan download file configngya (google-services.json).
3 Letakan file google-services.json tersebut di folder app/ dari project Anda.
4 Jangan lupa tambahkan dependensi pada gradle, lalu sync project anda.

Sejauh ini ada dua metode cara kirim notifikasi. Metode pertama adalah paling simple, mengirim melalui halaman console firebase. Secara sederhana, kita login ke

Console Firebase, lalu kita mengirimkan pesan notif melalui fitur yang sudah tersedia disana. Metode kedua adalah dengan dengan membuat server sebagai pengirim pesan, bahasa pemrogramannya bisa menggunakan php, go lang, phyton, java.

2.2.9 Data Koefisien Bahan Material Bangunan

Koefisien adalah jumlah bahan material yang dibutuhkan dalam satuan yang telah ditentukan. Misal untuk membangun Pondasi, Sloof, Kolom, Balok, & Plat 1 M^3lantai baru di butuhkan :

a. Batu Belah 1,35 M^3
b. Pasir Pasang 0,59 M^3
c. Semen Fortland 3,4 ZAK

Angka 0,59 pada Pasir Pasang adalah koefisien yang telah di tentukan oleh badan Penelitian pekerjaan umum di indonesia begitupun bahan cat dasar dan bahan cat penutup angka ini sudah menjadi standar nasional atau (SNI) [5]. Masih banyak lagi pekerjaan lainnya, adapun data pekerjaan atau Analisa Hasil Standar Pekerjaan (AHSP) yang digunakan didalam penelitian ini adalah sebagai berikut :

1. Pekerjaan Pondasi Batu Belah 1 : 5

Tabel 2. 1 Standar Pekerjaan Pondasi Batu Belah 1 : 5.

No	Nama Bahan	Satuan	Koefisien
1	Batu Belah	M3	1,35
2	Pasir Pasang	M3	0,59
3	Semen Fortland	Zak	3,4

2. Pekerjaan Beton Pondasi, Sloof, Kolom, Balok, & Plat

Tabel 2. 2 Standar Pekerjaan Beton Pondasi, Sloof .

No	Nama Bahan	Satuan	Koefisien
1	Pasir Cor	M3	0,43

2	Koral Pecah 1/2	M3	0,8
3	Semen 40 kg	Zak	6,85

3. Pekerjaan Pemasangan batu bata

Tabel 2. 3 Standar Pekerjaan Pemasangan batu bata.

No	Nama Bahan	Satuan	Koefisien
1	Bata Merah	Bj	70
2	Pasir Pasang	M3	0,048
3	Semen 40 kg	Zak	0,2448

4. Pekerjaan Pembesian

Tabel 2. 4 Standar Pekerjaan Pembesian.

No	Nama Bahan	Satuan	Koefisien
1	Besi Polos 10 mm	Ljr	16,81818
2	Besi Polos 8 mm	Ljr	16,81818
3	Kawat Bendrat	Kg	1,7
4	Papan Randu/ begesting	Lbr	5,90625
5	Kayu 2/3 atau 3/5	Btng	18,75
6	Paku segala ukuran (1/2", 1", 1,5", & 2")	Kg	0,35

5. Pekerjaan Pengecetan

Tabel 2. 5 Standar Pekerjaan Pengecetan.

No	Nama Bahan	Satuan	Koefisien
1	Cat menie	Kg	0,20
2	Amplas	Lbr	0,2
3	Plamuur	Kg	0,15

4	Cat dasar	Kg	0,17
5	Cat penutup	Kg	0,26
6	Kuas	Bh	0,01
7	Pengencer	Kg	0,03

6. Pekerjaan Pasang Plafond

Tabel 2. 6 Standar Pekerjaan Pasang Plafond 1m2.

No	Nama Bahan	Satuan	Koefisien
1	Asbes semen	m2	1,100
2	Paku tripleks	Kg	0,010

7. Pekerjaan Plesteran, Acian, dan Benangan

Tabel 2. 7 Standar Pekerjaan Plesteran, Acian, dan Benangan.

No	Nama Bahan	Satuan	Koefisien
1	Pasir Pasang	M3	0,018
2	Semen 40 kg	Kg	4,450

8. Pekerjaan Pasang Keramik Lantai

Tabel 2. 8 Standar Pekerjaan Pasang Keramik Lantai 33cm x 33cm 1m2.

No	Nama Bahan	Satuan	Koefisien
1	Ubin keramik	Dus	1,05
2	Pasir Pasang	M3	0,045
3	Semen	Kg	8,19
4	Semen Nat / Nat Tile	Kg	0,50

9. Pekerjaan Pasang Atap Genteng Palentong Kecil

Tabel 2. 9 Standar Pekerjaan Pasang Atap 1m2.

No	Nama Bahan	Satuan	Koefisien
1	Genteng palentong	Bh	25,00

10. Pekerjaan Pasang Kaca 3 mm

Tabel 2. 10 Standar Pekerjaan Pasang Kaca 3mm 1m2.

No	Bahan	Satuan	Koefisien
1	Kaca tebal 3mm	m^2	1,10
2	Sealant	Kg	0,05

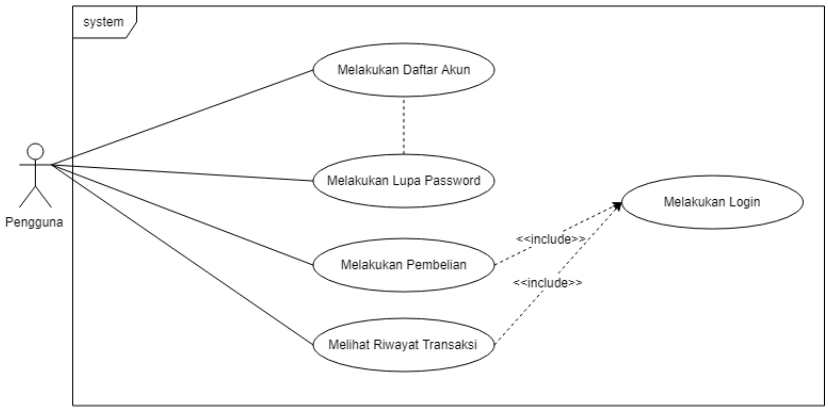
2.2.10 Unified Modeling Language (UML)

Unified Modelling Language (UML) adalah sebuah bahasa yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menggunakan class dan operation object dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa bahasa berorientasi objek [9].

2.2.10.1 Use Case Diagram

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah Use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Sebuah use case dapat meng-include fungsionalitas use case lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa use case yang di-include akan dipanggil setiap kali use case yang meng-include dieksekusi secara normal. Sebuah use case dapat di- include oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang serupa. Sebuah use case juga dapat meng-extend usecase lain dengan behaviour-nya sendiri. Sementara hubungan generalisasi antar use case menunjukkan bahwa use case yang

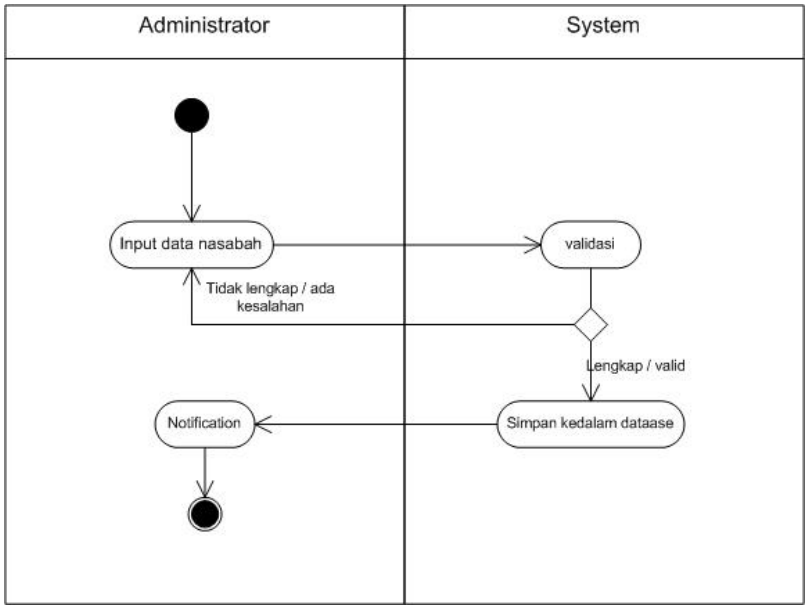
satu merupakan spesialisasi dari yang lain. Dasar menentukan sebuah use case adalah use case merupakan sesuatu yang menyediakan beberapa hasil terukur kepada pengguna atau sistem eksternal. Use case harus memiliki sangat jelas kriteria lulus / gagal. Pengembang, tester, penulis teknis, dan pengguna harus secara eksplisit tahu apakah sistem memenuhi kasus penggunaan atau tidak. Setiap bagian dari use case yang memenuhi tes sederhana ini mungkin menjadi kandidat yang baik untuk use case [10].



Gambar 2. 5 Contoh Use Case Diagram

2.2.10.2 Activity Diagram

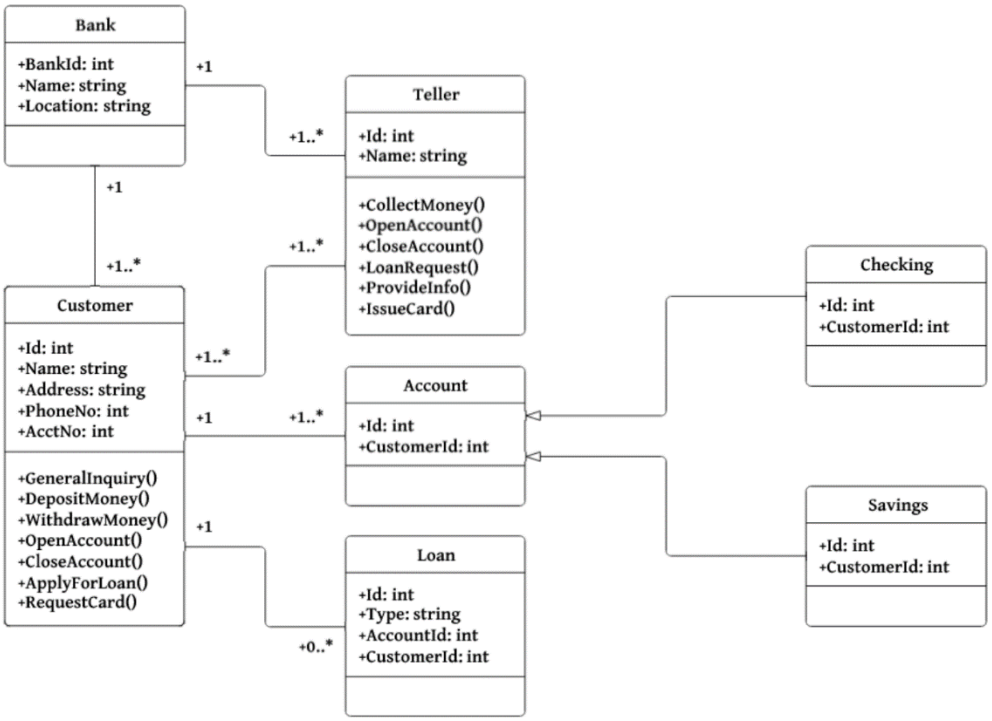
Diagram aktivitas pada umumnya adalah diagram *flowchart* yang diperluas yang menujukan aliran kendali satu aktivitas ke aktivitas lain. Diagram aktivitas berfokus pada aktivitas-aktivitas, potongan-potongan dari proses yang boleh jadi berkorespondensi dengan metode – metode atau fungsi-fungsi anggota dan pengurutan dari aktivitas-aktivitas ini. Elemen-elemen yang ada pada diagram aktivitas adalah *aktivity state* dan *action state*, transisi, dan objek. Pada penelitian ini diagram aktifitas digunakan untuk menggambarkan alur sistem dari awal hingga proses berakhir.



Gambar 2.7 *Activity diagram*.

2.2.10.2 Class Diagram

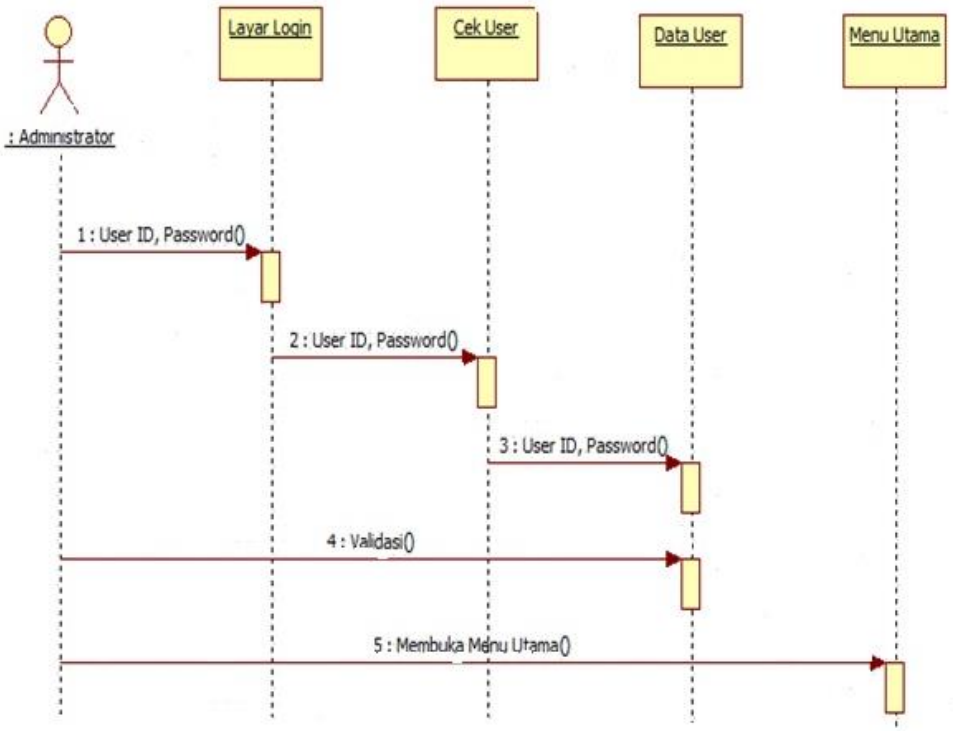
Class diagram merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. Class diagram menunjukan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi property dari kelas beserta metode yang ada pada kelas tersebut [10]. Kelas juga memiliki jenis - jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi, spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas.



Gambar 2. 6 Contoh Class diagram

2.2.10.3 Sequence Diagram

Sequence diagram adalah suatu diagram interaksi yang menekankan pada pengaturan waktu dari pesan-pesan. Diagram ini menampilkan sekumpulan peran dan pesan-pesan yang dikirim dan diterima oleh instansi yang memegang peranan tersebut. Sequence diagram menangkap objek dan class yang terlibat dalam scenario dan urut-urutan pesan yang ditukar antara objek diperlukan untuk melaksanakan fungsionalitas skenario. Sequence diagram berasosiasi dengan use case selama proses pengembangan. Dalam Unified Model Language (UML), objek dalam sequence diagram di dengan segiempat yang berisi nama objek yang diberi garis bawah. Objek dapat diberi nama dengan tiga cara : (nama objek), (nama objek dan class) atau (hanya nama class (anonymous object)).



Gambar 2. 7 Contoh Sequence Diagram

1.2.11 Model Skala Pengukuran Likert

Skala Likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang kejadian atau gejala sosial. Dengan menggunakan skala Likert, maka variabel dijabarkan menurut urutan variabel – sub variabel – indikator – deskriptor. dan deskriptor ini dapat dijadikan titik tolak untuk membuat butir instrumen berupa pernyataan atau pertanyaan yang perlu dijawab oleh responden.

Setiap jawaban dihubungkan dengan bentuk pernyataan atau dukungan sikap yang diungkapkan dengan kata – kata sebagai berikut :

Tabel 2. 11 Contoh 1 Jawaban dalam Skala Likert

Pernyataan Positif	Skor	Pernyataan Negatif	Skor
Sangat Setuju (SS)	5	Sangat Setuju (SS)	1
Setuju (S)	4	Setuju (S)	2
Netral (N)	3	Netral (N)	3
Tidak Setju (TS)	2	Tidak Setju (TS)	4
Sangat Tidak Setuju (STS)	1	Sangat Tidak Setuju (STS)	5

Terdapat 5 item untuk mengukur sikap terhadap kualitas produk yang dihasilkan oleh sebuah perusahaan X, dengan lima respon (kategori) dan dijawab oleh 10 responden, maka, setelah dibagikan kepada responden, misalnya diperoleh skor dan skor total seperti tertera pada tabel 2.14 berikut :

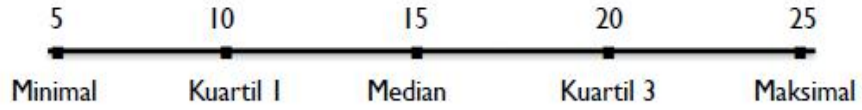
Tabel 2. 12 Hasil Skor yang diberikan responden

Responden	Butir / Item Pernyataan					Total
	1	2	3	4	5	
1	5	4	4	5	3	21
2	3	4	4	3	3	17
3	3	2	2	3	4	14
4	3	2	1	2	4	12
5	4	3	3	3	5	18
6	5	4	4	3	5	21
7	4	5	4	4	4	21
8	4	4	5	5	4	22
9	3	3	4	4	3	21
10	2	3	3	4	3	15
Total						**182**

Berdasarkan data tersebut, langkah-langkah yang dapat ditempuh untuk mengetahui bagaimana ***sikap tiap responden*** terhadap kualitas produk adalah:

1. Menentukan skor maksimal, yaitu skor jawaban terbesar di kali banyak item. 5 x 5 = 25

2. Menentukan skor minimal, yaitu skor jawaban terkecil dikali banyak item. 1 x 5 = 5
3. Menentukan nilai median, yaitu hasil penjumlahan skor maksimal dengan skor minimal dibagi dua. (25+5) : 2 = 15
4. Menentukan nilai kuartil 1, yaitu hasil penjumlahan skor minimal dengan median dibagi dua. (5+15) : 2 = 10
5. Menentukan nilai kuartil 3, yaitu hasil penjumlahan skor maksimal dengan median dibagi dua. (25+15) : 2 = 20
6. Buatlah skala yang menggambarkan skor minimal, nilai kuartil 1, median, kuartil 3 dan skor maksimal.



Gambar 2.10 Skala Skor Minimal Hingga Maksimal

1. Kategori sikap sangat positif, yaitu daerah yang dibatasi oleh kuartil 3 dan skor maksimal. (Kuartil 3 ≤ x ≤ skor maksimal.
2. Kategori sikap positif, yaitu daerah yang dibatasi oleh median dan kuartil 3. (Median ≤ x < Kuartil 3).
3. Kategori sikap negatif, yaitu daerah yang dibatasi oleh kuartil 1 dan median. (Kuartil 1 ≤ x < Median).
4. Kategori sikap sangat negatif, yaitu daerah yang dibatasi oleh skor minimal dan kuartil 1. (Skor minimal ≤ x < kuartil 1)

7. Carilah batas – batas skor untuk masing masing kategori sikap. Berdasarkan gambar skala tadi, maka skor dari keempat kategori adalah :

Tabel 2. 13 Batas Skor Kategori Sikap

Sikap Sangat Positif	Kurartil 3 ≤ x ≤ Skor Maksimal	20 –25
Sikap Positif	Median ≤ x < Kuartil 3	15 –20
Sikap Negatif	Kuartil 1 ≤ x < Median	10 –15

Sikap Sangat Negatif	Skor Minimal ≤ x < Kuartil 1	5 –10

8. Buatlah tabel distribusi frekuensi sikap tiap responden terhadap kualitas produk seperti pada tabel 2.16 berikut untuk mengetahui gambaran dari setiap sikap responden terhadap kualitas produk.

Tabel 2. 14 Tabel Distribusi Frekuensi

Kategori Sikap	Kategori Skor	Frekuensi	Persentase(%)
Sikap Sangat Positif	20 –25	5	50
Sikap Positif	15 –20	3	30
Sikap Negatif	10 –15	2	20
Sikap Sangat negatif	5 –10	-	-
		10	100

9. Berdasarkan tabel di atas tampak bahwa sikap tiap responden tentang kualitas produk tersebar pada kategori sikap yang sangat positif 50%, sikap positif 30% dan sikap negatif 20%. Persentase tersebut memberikan arti bahwa sebanyak 5 orang (50% dari keseluruhan responden) memandang produk yang dihasilkan oleh perusahaan X adalah sangat berkualitas. Lalu sebanyak 3 orang (30% dari keseluruhan responden) memandang produk yang dihasilkan oleh perusahaan X adalah berkualitas.