BAB 2
TINJAUAN PUSTAKA

2.1 Profil SD Negeri Panembong 1

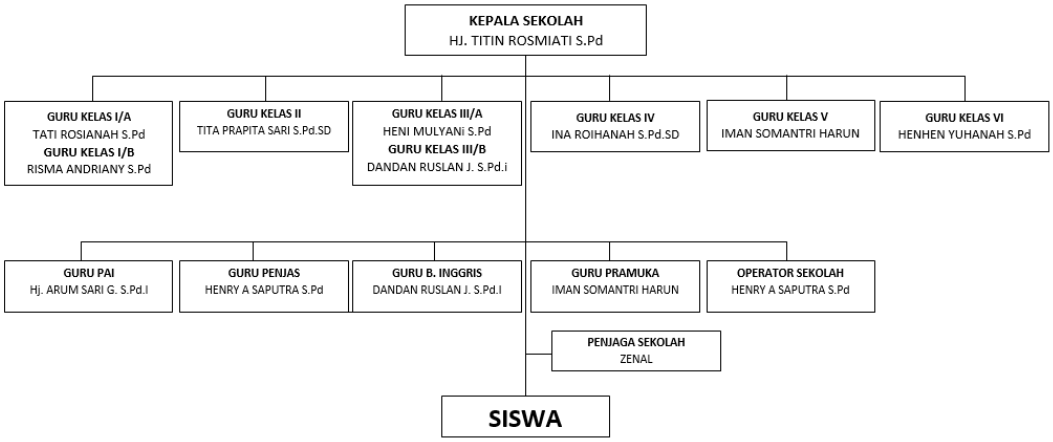
Sekolah Dasar (SD) Negeri Panembong 1 UPTD (unit pelaksana teknis daerah) Pendidikan Kecamatan Cianjur merupakan lembaga pendidikan formal tingkat awal. SD Negeri Panembong 1 memiliki 10 orang guru, 129 siswa laki-laki, 114 siswa perempuan, 7 rombongan belajar, 7 ruang kelas, dan 1 perpustakaan. Kurikulum yang digunakan SD Negeri Panembong 1 ini adalah Kurikulum 2013.

2.1.1 Lokasi SD Negeri Panembong 1

Sekolah Dasar (SD) Negeri Panembong 1 berlokasi di Jl. Desa Limbangansari RT 02/ RW 06 Kel. Limbangansari Kec. Cianjur Kab. Cianjur Prov. Jawa Barat 43217.

2.1.2 Struktur Organisasi SD Negeri Panembong 1

Stuktur Organisasi Sekolah Dasar (SD) Negeri Panembong 1 Kab. Cianjur adalah sebagai berikut:



Gambar 2.1 Struktur Organisasi SD Negeri Panembong 1

2.1.3 Visi dan Misi

Visi dan Misi SD Negeri Panembong 1 adalah sebagai berikut:

2.1.3.1 Visi

Visi SD Negeri Panembong 1 yaitu terwujudnya peserta didik yang cerdas, terampil, sehat dan berakhlakul karimah.

2.1.3.2 Misi

Misi SD Negeri Panembong 1 adalah sebagai berikut:

1. Memberikan pelayanan pembelajaran yang optimal.
2. Mengembangkan bakat, minat dan keterampilan siswa.
3. Membiasakan hidup bersih dan sehat.
4. Menerapkan sistem pembelajaran yang berlandaskan akhlak mulia.

2.1.4 Tujuan

Tujuan Pendidikan yang ingin dicapai di Sekolah Dasar Negeri Panembong 1 UPTD Pendidikan Kecamatan Cianjur adalah membentuk peserta didik yang cerdas, terampil dan sehat, serta dilandasi akhlak mulia sebagai bekal hidup mandiri, serta siap untuk melanjutkan ke jenjang Pendidikan yang lebih tinggi.

2.1.5 Logo SD Negeri Panembong 1

Berikut ini merupakan gambar logo SD Negeri Panembong 1

Gambar 2.2 Logo SD Negeri Panembong 1

2.2 Android

Basis dari sistem operasi Android adalah Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri dan digunakan oleh berbagai macam perangkat *mobile*. Pada saat perilisan perdana, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode – kode Android di bawah lisensi Apache. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD) [4].

Gambar 2.3 Logo Android

2.2.1 Arsitektur Android

Secara garis besar Arsitektur Android adalah sebagai berikut [4]:

1. *Applications dan Widgets*
2. *Application Frameworks*
3. *Libraries*
4. *Android Runtime*
5. *Linux Kernel*

2.2.1.2 *Applications dan Widgets*

Puncak dari diagram arsitektur Android adalah lapisan aplikasi dan widget. Lapisan aplikasi merupakan lapisan yang paling terlihat oleh pengguna saat program dijalankan. Pengguna hanya bakal menyaksikan program kala digunakan tanpa tahu sistem yang berjalan dibalik lapisan aplikasi [4].

2.2.1.3 *Application Frameworks*

Kerangka aplikasi menyediakan kelas-kelas yang bisa digunakan untuk mengembangkan aplikasi Android. Bagian terpenting didalam kerangka aplikasi Android adalah sebagai tersebut [4]:

1. *Activity Manager*, berfungsi untuk mengontrol siklus hidup aplikasi dan menjaga keadaan ”Backstack“ untuk navigasi penggunaan.
2. *Content Providers*, berfungsi untuk merangkum data yang memungkinkan digunakan oleh aplikasi lainnya, seperti daftar nama.
3. *Resource Manager*, untuk mengatur sumber daya yang ada dalam program. Serta menyediakan akses sumber daya diluar kode program, seperti karakter, grafik, dan file layout.
4. *Location Manager*, berfungsi untuk memberikan informasi detail mengenai lokasi perangkat Android berada.
5. *Notification Manager*, mencakup berbagai macam peringatan seperti, pesan masuk, janji, dan lain sebagainya yang akan ditampilkan pada status bar.

2.2.1.4 *Libraries*

Android menggunakan beberapa paket pustaka yang terdapat pada C/C++ dengan standar *Berkeley Software Distribution* (BSD) hanya setengah dari yang aslinya untuk tertanam pada kernel Linux. Beberapa pustaka diantaranya [4]:

1. *Media Library* untuk memutar dan merekam berbagai macam format audio dan video.
2. *Surface Manager* untuk mengatur hak akses layer dari berbagai aplikasi.
3. *Graphic Library* termasuk didalamnya SGL dan OpenGL, untuk tampilan 2D dan 3D.
4. *SQLite* untuk mengatur relasi *database* yang digunakan pada aplikasi.

5. SSl dan WebKit untuk *browser* dan keamanan internet.

Pustaka-pustaka tersebut bukanlah aplikasi yang berjalan sendiri, namun hanya dapat digunakan oleh program yang berada di level atasnya. Sejak versi Android 1.5, pengembang dapat membuat dan menggunakan pustaka sendiri menggunakan *Native Development Toolkit* (NDK).

2.2.1.5 *Android Runtime*

Android Runtime merupakan mesin virtual yang membuat aplikasi Android menjadi lebih baik dengan paket pustaka yang telah ada. Dalam Android *Runtime* terdapat 2 bagian utama, yaitu [4]:

1. Pustaka Inti, Android dikembangkan melalui bahasa pemrograman Java, tapi Android Runtime bukanlah mesin virtual Java. Pustaka inti Android menyediakan hampir semua fungsi yang terdapat pada pustaka Java serta beberapa pustaka khusus Android.
2. Mesin Virtual Dalvik, Dalvik merupakan sebuah mesin virtual yang dikembangkan oleh Dan Bornstein yang terinspirasi dari nama sebuah perkampungan yang berada di Iceland. Dalvik hanyalah interpreter mesin virtual yang mengeksekusi file dalam format Dalvik Executable (*.dex).

2.2.1.6 *Linux Kernel*

Android dibangun di atas kernel Linux 2.6. Linux merupakan sistem operasi terbuka yang handal dalam manajemen memori dan proses. Oleh karenanya pada Android hanya terdapat beberapa pelayanan yang diperlukan seperti keamanan, manajemen memori, manajemen proses, jaringan dan driver. Kernel linux menyediakan driver layar, kamera, *keypad*, WiFi, *Flash Memory*, *audio*, dan IPC (*Interprocess Communication*) untuk mengatur aplikasi dan keamanan [4].

2.2.2 Komponen Aplikasi Android

Fitur penting Android adalah bahwa satu aplikasi dapat menggunakan elemen dari aplikasi lain (untuk aplikasi yang memungkinkan). Sebagai contoh,

sebuah aplikasi memerlukan fitur scroller dan aplikasi lain telah mengembangkan fitur scroller yang baik dan memungkinkan aplikasi lain menggunakannya.

Agar fitur tersebut dapat bekerja, sistem harus dapat menjalankan aplikasi ketika setiap bagian aplikasi itu dibutuhkan, dan pemanggilan objek java untuk bagian itu. Oleh karenanya Android berbeda dari sistem-sistem lain, Android tidak memiliki satu tampilan utama program seperti fungsi *main()* pada aplikasi lain. Sebaliknya, aplikasi memiliki komponen penting yang memungkinkan sistem untuk memanggil dan menjalankan ketika dibutuhkan [4].

2.2.2.1 *Activities*

Activity merupakan bagian yang paling penting dalam sebuah aplikasi, karena *Activity* menyajikan tampilan visual program yang sedang digunakan oleh pengguna. Setiap *Activity* dideklarasikan dalam sebuah kelas yang bertugas untuk menampilkan antarmuka pengguna yang terdiri dari *Views* dan respon terhadap *Event* [4].

2.2.2.2 *Services*

Suatu *service* tidak memiliki tampilan antarmuka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat, memperbarui sumber data dan menampilkan notifikasi. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *Activity* tidak aktif atau tidak tampak [4].

2.2.2.3 Intents

Intens merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman *web*, dan lain sebagainya. Intents tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misal pesan masuk [4].

2.2.2.4 *Broadcast Receiver*

Broadcast Receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sama halnya dengan service, *Broadcast Receivers* tidak menampilkan antarmuka pengguna. Namun, *Broadcast Receivers* dapat menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna [4].

2.2.2.5 *Content Providers*

Content Providers digunakan untuk mengelola dan berbagi *database*. Data dapat disimpan dalam *file* sistem, dalam *database SQLite*, atau dengan cara lain yang pada prinsipnya sama. Dengan adanya *Content Provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya [4].

2.2.3 Tipe Aplikasi Android

Terdapat 3 kategori aplikasi android [4]:

1. *Foreground Activity* Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap efektif walaupun tidak terlihat.
2. *Background Service* Aplikasi yang memiliki interaksi terbatas dengan user, selain dari pengaturan konfigurasi, semua dari prosesnya tidak tidak tampak pada layar.
3. *Intermittent Activity* Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat efektif jika dijalankan di *background* dan jika diperlukan akan memberi tahu pengguna tentang kondisi tertentu.

2.3 *Game*

Menurut Erick Zimmerman, dan Katie Salen Game merupakan suatu sistem yang memiliki aturan-aturan tertentu, dimana pemain akan terlibat di dalam suatu permasalahan sehingga dapat menghasilkan suatu hasil yang dapat diukur yaitu menang atau kalah. *Game* merupakan sesuatu hal yang dimainkan dengan suatu

aturan tertentu yang biasa digunakan untuk tujuan kesenangan dan dapat juga digunakan sebagai sarana edukasi.

Game umumnya melibatkan stimulasi mental, fisik, atau keduanya. Banyak *game* yang dapat membantu mengembangkan keterampilan praktis yang berfungsi sebagai latihan, atau melakukan peran pendidikan, simulasional, atau psikologis [5].

2.3.1 Jenis – jenis *Game*

Menurut Expro (2010) terdapat beberapa jenis game sebagai berikut [6]:

1. *Action Games*

   Jenis *game* yang pertama yaitu permainan aksi (*Action games*), jenis game yang satu ini adalah game yang menggunakan refleks, akurasi, dan waktu yang tepat untuk menyelesaikan sebuah tantangan. Ini mungkin merupakan genre dasar dari sebuah permainan, dan yang memiliki permainan terbanyak. Dalam permainan aksi, biasanya terdapat pertempuran. Sub-genre dari permainan aksi ini yaitu seperti : *Fighting*, FPS, TPS

2. *Fighting Games*

   Fighting game adalah jenis game bertarung. Seperti dalam arcade, biasanya pada game jenis fighting kita dapat memilih karakter dengan kemampuan yang berbeda-beda kita dapat mengeluarkan jurus-jurus ampuh dalam pertarungannya. Jenis fighting biasanya one on one dalam sebuah arena yang sempit atau dibatas luas arena pertarungannya. Contoh *action game* : *Mortal kombat, Street fighter*, *Tekken*, dll

3. FPS (*First Person Shooter*)

   FPS (*First Person Shooter*) adalah jenis *game* menembak dengan tampilan pada *game* tersebut yaitu tokoh yang sedang kita mainkan, biasanya cara pandangnya hanya memperlihatkan tangan dan senjatanya saja. *Game* FPS ini memilik misi-misi untuk suatu tujuan tertentu. Ciri khas *game* ini adalah penggunaan senjata jarak jauh. Contoh *game* Fps: *Cross Fire*, *Point blank*, *Counter Strike*, dll

4. TPS (*Third Person Shooter*)

   TPS (*Third Person Shooter*) adalah *game* yg memiliki ciri khas mirip dengan FPS yaitu biasanya memiliki *gameplay* tembak menembak hanya saja sudut pandang yg digunakan dalam *game* ini adalah orang ketiga terlihat setengah badan. Contoh *game* TPS: *Ghost Recon, Hallo, Dead space* dll

5. *Strategy*

   Strategy adalah genre *game* yang memiliki *gameplay* untuk mengatur suatu unit atau pasukan untuk menyerang markas musuh dalam rangka memenangkan permainan. Biasanya di dalam game *Strategy*, kita dituntut untuk mencari gold untuk membiayai pasukan kita. *Games Strategy* dibagi 2, yaitu sebagai berikut:

   a. *Real Time Strategy* (RTS)

      Pada *game* ini, kita dapat mengendalikan pasukan secara langsung, dari mencari sumber daya, hingga menghancurkan musuh. Semua pertempuran ini dapat kita saksikan secara langsung.

   b. *Turn Based Strategy* (TBS)

      Sistemnya seperti *Turn Based* RPG, tetapi disini selain mengendalikan *character* utama, kita mengendalikan pasukan dan kota kita untuk memenangkan pertarungan. Biasanya kita memainkan game nya di atas peta.

6. *Tycoon*

   Tycoon adalah *game* yang menjadikan kita sebagai seorang *bussinesman* yang akan mengembangkan sesuatu properti untuk dikembangkan hingga laku di pasaran.

7. *Racing*

   Racing Game adalah *game* sejenis *racing* yang memungkinkan kita untuk mengendalikan sebuah kendaraan untuk memenangkan sebuah balapan.

8. *Action Adventure*

   Action Adventure adalah *game* berupa petualangan salah seorang karakter yang penuh dengan penuh aksi yang akan terus ada hingga *game* tersebut tamat. (Biasanya *Action* dimasukan kategori RPG).

9. *Arcade*

   Arcade game adalah *genre game* yang tidak terfokus pada cerita, melainkan hanya dimainkan "*just for fun*" atau untuk kejar-mengejar *poin*.

10. *Fighting Game*

   Fighting adalah genre game bertarung. Seperti dalam arcade, pemain dapat mengeluarkan jurus-jurus ampuh dalam pertarungannya. Genre *fighting* biasanya *one on one* dalam sebuah arena yang sempit.

11. *Sports*

   Adalah genre bertema permainan olahraga. Sistem permainan akan berbeda-beda tergantung jenis olahraga yang menjadi tema *game* tersebut.

12. *Casual*

   Adalah sebuah tipe *game* yang menyuguhkan *gameplay* sederhana sehingga mampu dinikmati oleh banyak kalangan. *Gameplay* yang relatif sederhana juga berimbas pada biaya produksi yang kemudian juga menjadi relatif lebih murah. Tapi kesederhanaan dari sebuah *casual game* tidak secara otomatis membuat *game* tipe ini jadi lebih mudah untuk dikembangkan, bahkan sebaliknya dalam beberapa hal menjadi lebih menantang.

2.3.2 *Game* Edukasi

Game Edukasi adalah *game* yang dibuat untuk membantu pengguna dalam mempelajari sesuatu, baik tentang konsep, pemahaman ataupun latihan. Perancangan *Education game* yang baik menurut Hurd dan Jenuings haruslah memenuhi beberapa kriteria berikut ini [7]:

1. Nilai Keseluruhan (*Overall Value*)

   Nilai keseluruhan dari suatu *game* terpusat pada desain dan panjang durasi *game*.

2. Dapat Digunakan (*Usability*)

   Mudah digunakan adalah poin penting bagi pembuat game.

3. Keakuratan (*Accuracy*)

   Keakuratan diartikan sebagai bagaimana keberhasialn model sebuah *game* dapat dituangkan ke dalam percobaan atau perancangannya.

4. Kesesuaian (*Appropriateness*)

   Kesesuaian dapat diartikan bagaimana isi dan desain *game* dapat diadaptasikan terhadap keperluan user dengan baik.

5. Relevan (*Relevance*)

   Relevan artinya dapat menyampaikan isi game ke target *user*. Sistem harus dapat menuntun user untuk mencapai tujuan pembelajaran.

6. Objektifitas (*Objectives*)

   Objektifitas dalam menentukan tujuan *user* dan karakteristik dari keberhasilan atau kegagalan.

7. Umpan Balik (*Feedback*)

   Untuk membantu pemahaman user bahwa permainan (*performance*) mereka sesuai dengan objek *game* atau tidak, *feedback* harus disediakan.

2.4 *Google Play Games*

Google Play Games adalah layanan permainan daring dan SDK yang dioperasikan oleh Google untuk sistem operasi Android. Ini memiliki kemampuan permainan multi-pemain secara real-time, penghematan awan, papan skor sosial dan publik, prestasi, dan fitur anti-pembajakan. Layanan ini memungkinkan pengembang untuk menyertakan fitur dalam permainan mereka tanpa harus mengembangkan fitur tersebut sendiri. Layanan Google Play Games diperkenalkan di Google I/O 2013 *Developer Conference*. Pada tahun 2015, ribuan permainan yang diterbitkan melalui Google Play Store mendukung layanan tersebut.

Aplikasi bergerak mandiri dari *Google Play Games* diluncurkan pada tanggal 24 Juli 2013, di sebuah acara yang disebut "Sarapan dengan Sundar Pichai" bersama-sama dengan Nexus 7 baru, Android 4.3 dan Chromecast. Layanan ini mirip dengan *Apple Game Center*, jaringan permainan sosial untuk iOS. Salah satu fitur sosial dari *Play Games* adalah mendaftarkan pengguna Google+ permainan teman di layar utamanya. Menurut Google, Google Play menerima lebih dari 100 juta pengguna baru antara Januari 2014 dan Juni 2014, membuatnya jaringan permainan bergerak tercepat sepanjang masa [8].

2.4.1 *Google Play Games Real-time Multiplayer*

Google Play games real-time multiplayer API (*Application Programming Interface*) merupakan salah satu layanan yang disediakan oleh *Google Play* yang dapat digunakan untuk menghubungkan beberapa pemain secara bersamaan ke dalam suatu permainan dan mentransfer data antara pemain yang terhubung. Layanan ini dapat membantu upaya pengembangan game karena API dapat menangani beberapa tugas sebagai berikut [9].

1. Mengelola koneksi jaringan untuk membuat dan menjaga *room* (ruangan) real-time multiplayer yang digunakan.
2. Menyediakan antarmuka untuk memilih pemain yang akan diundang kedalam permainan, mencari pemain secara acak, atau gabungan dari keduanya.
3. Menyimpan informasi ruangan dan status ruangan di server *Google Play game* selama siklus dari *real-time multiplayer* berlangsung.
4. Mengirim undangan bermain kepada pemain lain. Pemberitahuan undangan bermain akan muncul disemua perangkat tempat pemain masuk (*log in*).

Google play games sudah menyediakan beberapa fitur yang bisa langsung digunakan seperti:

1. *Create a Quick Game*

   Sistem akan mengatur dan memasukan pemain kedalam room dengan lawan secara acak (juga disebut "*automatch*"). Pengembang game dapat menetapkan jumlah minimum dan maksimum lawan dalam permainan dan *platform Google Play Game* secara otomatis menempatkan pemain di sebuah room dengan jumlah lawan yang sudah ditentukan.
2. *Create With Invitation Screen*

   Google play game akan menampilkan tampilan undangan standar kepada pengguna, di mana mereka dapat memilih teman mana yang ingin mereka ajak bermain. Layar ini juga memungkinkan pemain untuk menambah lawan otomatis, sehingga mereka bahkan dapat mencampur dan mencocokkan (misalnya, mereka dapat memilih untuk bermain dengan dua teman tertentu dan satu lawan acak).

3. *Accept From Inbox*

   Google play game akan menampilkan kotak masuk undangan kepada pengguna, yang merupakan layar Google Play Game standar yang berisi semua undangan yang tertunda yang telah diterima pengguna. Pengguna kemudian dapat menerima salah satu undangan tersebut untuk bergabung ke sebuah ruangan. Menerima undangan. Terima undangan tertentu yang ID-nya Anda kenal. Ini biasanya dilakukan sebagai tanggapan untuk menerima undangan ke ruang (kami akan membahas ini dalam detail mode nanti).

4. *Accept Invitation*

   Terima undangan tertentu yang ID-nya Anda kenal. Ini biasanya dilakukan sebagai tanggapan untuk menerima undangan ke kedalam sebuah room.

Dalam proses pengiriman data (pesan) *google play games* menyediakan beberapa jenis pengiriman data (pesan) yaitu:

1. Berdasarkan jenis data (pesan)
   a. *Reliable messages*

      Reliable messages memiliki sifat cukup cepat dan pengiriman data, integritas, dan pemesanan dijamin sampai tujuan. Pengembang *game* dapat memilih untuk diberi tahu tentang status pengiriman dengan menggunakan *callback*. *Reliable messages* cocok untuk mengirim data yang tidak sensitif terhadap waktu. Pengembang *game* juga dapat menggunakan *Reliable messages* untuk mengirim kumpulan data besar di mana data dapat dibagi menjadi segmen yang lebih kecil, dikirim melalui jaringan, dan kemudian dipasang kembali oleh klien penerima. *Reliable messages* mungkin memiliki ketepatan pengiriman yang tinggi. Ukuran maksimum pesan tepercaya yang dapat Anda kirim adalah 1400 byte.

   b. *Unreliable messaging*.

      Unreliable messaging memiliki sifat sangat cepat tetapi pesan tidak dijamin sampai tujuan. Klien *game* hanya mengirimkan data sekali tanpa jaminan pengiriman data tiba ditujuan. Namun, integritas terjamin, jadi tidak perlu menambahkan *checksum*. Pesan tidak tepercaya memiliki latensi rendah dan cocok untuk mengirim data yang peka terhadap waktu. Aplikasi

bertanggung jawab untuk memastikan bahwa permainan berperilaku dengan benar jika pesan dijatuhkan dalam transmisi atau diterima tidak berurutan. Ukuran maksimum untuk pesan tidak tepercaya yang dapat dikirim adalah 1168 byte.

2. Berdasarkan penerima data
   a. *Send to all*

      Dikirim kesemua pemain di dalam room
   b. *Send to player*

      Dikirim ke pemain tertentu di dalam room.

2.5 Metode *Linear Congruent Method* (LCM)

Linear Congruent Method (LCM) merupakan *metode* pembangkitan bilangan acak yang banyak digunakan dalam program komputer. LCM memanfaatkan model linear untuk membangkitkan bilangan acak.

Pengacakan dengan *Linear Congruent Method* (LCM) menggunakan Persamaan (1).

$$\boldsymbol{x(i) = (a * x(i-1) + c)\ mod\ m} \quad (1)$$

dengan:

$x(i)$ = bilangan acak ke i.

$a\ dan\ c$ = konstanta LCM.

m = batas maksimum bilangan acak.

Ketentuan-ketentuan pemilihan setiap parameter pada Persamaan (1) adalah sebagai berikut:

a = multiplier (pengganda), $0 < a < m$

c = increment (penambah), $0 \leq c < m$

m = modulus, $0 < m$

$x(i)$ = nilai awal, $0 \leq x(0) < m$

$c\ dan\ m$ merupakan bilangan prima relatif

$a -$ 1 dapat dibagi oleh faktor prima dari m

$a -$ 1 kelipatan 4 jika m juga kelipatan 4

Salah satu sifat dari metode ini adalah terjadi pengulangan pada periode waktu tertentu atau setelah sekian kali pembangkitan, untuk mengatasi terjadinya pengulangan tersebut maka penentuan konstanta LCM $(a, c\ dan\ m)$ sangat menentukan baik dan tidaknya bilangan acak yang diperoleh dalam arti memperoleh bilangan acak yang seakan-akan tidak terjadi pengulangan dengan melakukan beberapa kali pengujian [10].

2.6 UML (*Unified Modeling Language*)

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu Unfied Modeling Language (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks - teks pendukung. UML hanya berfungsi untuk melakukan pemodelan [11].

Menurut Braun (2001), Unified Modelling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Sedangkan Whitten (2004) menyatakan UML merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek [12].

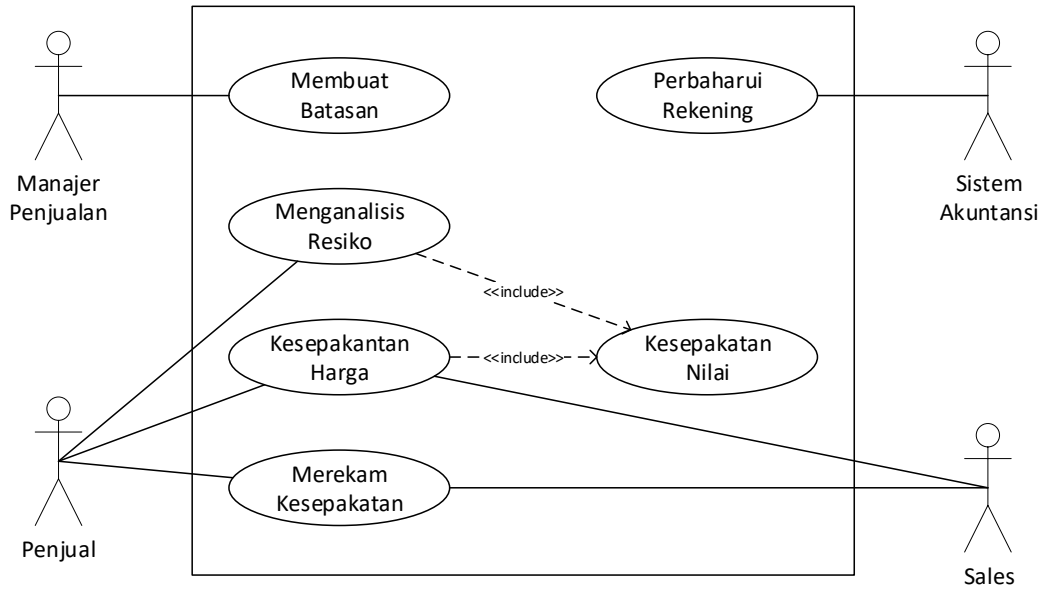
Menurut Dharwiyanti dan Wahono (2003), UML adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem [13].

Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

2.6.1 *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. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Use case diagram akan sangat menolong saat kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rencana bersama dengan klien, dan merancang test case untuk seluruh feature yang tersedia pada sistem. Sebuah *use case* juga mampu *meng-extend use case* lain bersama dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain [13].



Gambar 2.4 Contoh *Use Case Diagram*

Sumber: Buku UML Distilled Edisi 3 [14]

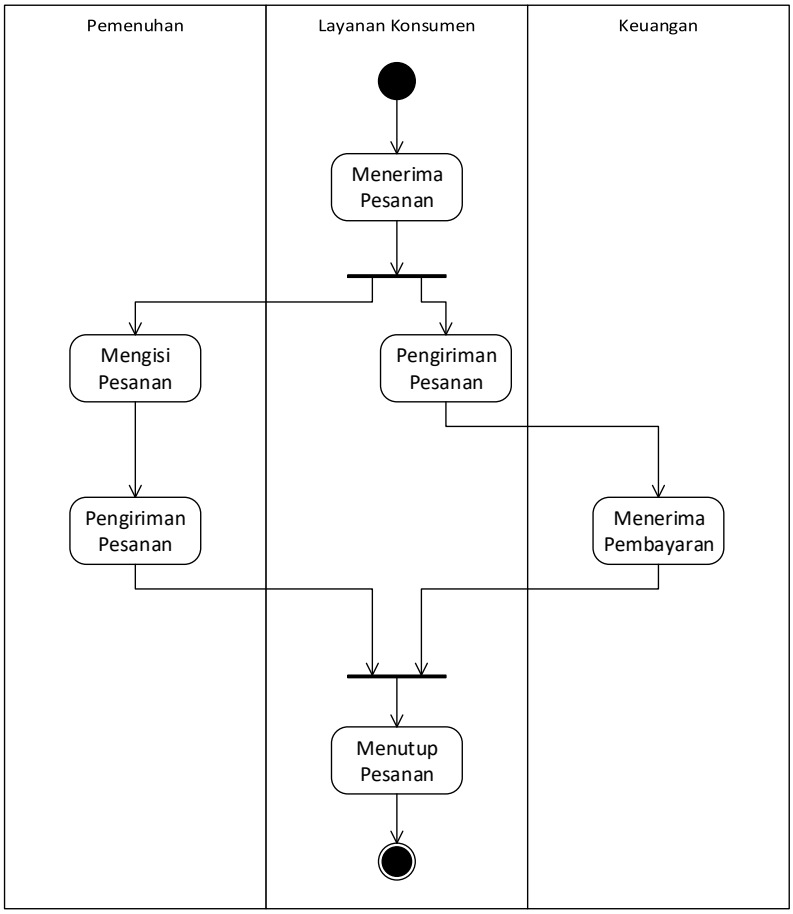
2.6.2 *Activity Diagram*

Activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang berada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa *activity diagram* menggambarkan

aktivitas bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan sistem [13].

Activity diagram juga banyak digunakan untuk mendefinisikan hal-hal berikut.

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didfinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.



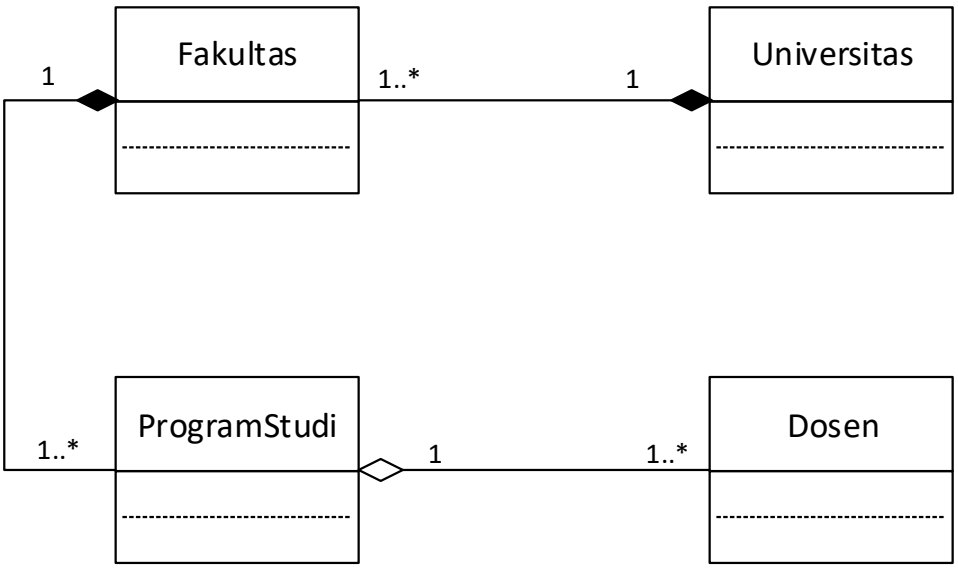
Gambar 2.5 Contoh *Activity Diagram*

Sumber: Buku UML Distilled Edisi 3 [14]

2.6.3 *Class Diagram*

Class diagram menunjang didalam memvisualisasikan susunan kelaskelas dari suatu proses dan merupakan *type* diagram yang paling banyak dipakai. Selama langkah desain, *class diagram* berperan didalam menangkap susunan dari seluruh kelas yang membentuk arsitektur proses yang dibuat.

Class adalah sebuah spesifikasi yang jikalau diinstansiasi akan membuahkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* diagram menjelaskan struktur dan uraian *class*, *package* dan objek beserta interaksi satu mirip lain seperti *containment*, pewarisan, asosiasi, dan lain-lain seperti pada gambar berikut [13].



Gambar 2.6 Contoh *Class Diagram*

Sumber: www.slideshare.net

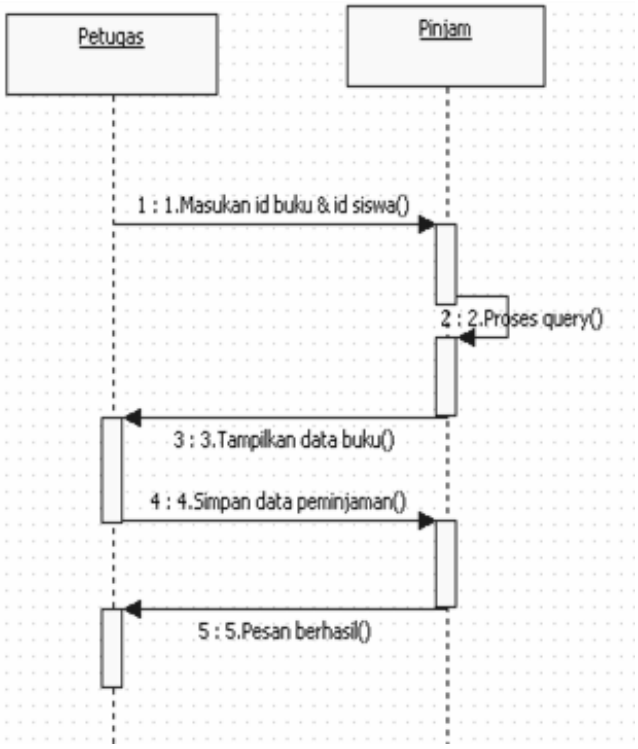
2.6.4 *Sequence Diagram*

Diagram sequence menampilkan interaksi antar objek dalam dua dimensi. Dimensi vertikal adalah waktu, dimana waktu terjadi ke arah bawah. Sedangkan dimensi horizontal merepresentasikan objek-objek individual. Tiap objek (termasuk aktor) tersebut mempunyai waktu aktif yang direpresentasikan dengan kolom vertikal yang disebut dengan *lifeline*. Pesan (*message*) direpresentasikan sebagai panah berasal dari satu lifeline ke *lifeline* yang lain. Secara mudahnya

sequence diagram adalah deskripsi tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dijalankan untuk membuat sesuatu sesuai dengan *use case diagram*.

Untuk menggambar *sequence* diagram maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki oleh kelas yang diinstansiasi menjadi objek itu. Membuat sequence diagram juga dibutuhkan untuk melihat *scenario* yang ada pada *use case*.

Banyaknya diagram *sequence* yang harus dibuat adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada *sequence diagram*, sehingga semakin banyak use case yang dibuat maka *sequence diagram* yang dibuat semakin banyak juga [13].



Gambar 2.7 Contoh *Sequence Diagram*

Sumber: saiiamilla.wordpress.com

2.7 Perangkat Lunak yang Digunakan

Berikut adalah perangkat lunak yang digunakan untuk membangun *game* ini yaitu: Unity versi 5.6.1, Adobe Photoshop CS.

2.7.1 Unity

Unity 3D adalah sebuah *game engine* yang berbasis *cross-platform*. Unity dapat digunakan untuk membuat sebuah game yang bisa digunakan pada perangkat komputer, ponsel pintar android, iPhone, PS3, dan bahkan X-BOX. Unity adalah sebuah sebuah *tool* yang terintegrasi untuk membuat *game*, arsitektur bangunan dan simulasi. Unity bisa untuk *games* PC dan *games Online*. Untuk *games online* diperlukan sebuah plugin, yaitu Unity *Web Player*, sama halnya dengan Flash *Player* pada *Browser*. Unity tidak dirancang untuk proses desain atau *modelling*, dikarenakan unity bukan *tool* untuk mendesain. Jika ingin mendesain, pergunakan 3D *editor* lain seperti 3dsmax atau Blender. Banyak hal yang bisa dilakukan dengan unity, ada fitur *audio reverb zone, particle effect*, dan *Sky Box* untuk menambahkan langit.

Gambar 2.8 Logo Unity

Fitur *scripting* yang disediakan, mendukung 3 bahasa pemrograman, JavaScript, C# dan Boo. *Flexible* dan *Easy Moving, rotating* dan *scaling objects* hanya perlu sebaris kode. Begitu juga dengan *Duplicating, removing,* dan *changing properties*. *Visual Properties Variables* yang di definisikan dengan *scripts* ditampilkan pada *Editor*. Bisa digeser, di drag and drop, bisa memilih warna dengan *color picker*. Berbasis .NET, artinya penjalanan program dilakukan dengan *Open Source* NET *platform*, Mono.

Unity menggunakan sebuah konsep yang disebut *Parenting*, ini digunakan untuk membuat sebuah *game object* menjadi anak dari *game object* yang lain. Tarik sebuah *game object* dan pindahkan tepat di atas tulisan *game object* yang akan dijadikan *parent* dalam hierarki. *Game object* yang terdapat dalam sebuah *game* object lainnya akan mengikuti perpindahan dan perputaran ketika *game object parent* mengalami perubahan posisi [15].

2.7.1.1 Fitur – Fitur Unity

Banyak kelebihan *game engine unity*, dapat dilihat dari beberapa fitur berikut ini [15].

1. *Rendering*

   Graphics engine yang digunakan adalah Direct3D (Windows, Xbox 360), OpenGL (Mac, Windows, Linux, PS3), OpenGL ES (Android, iOS), dan proprietary APIs (Wii). Ada pula kemampuan untuk *bump mapping, reflection mapping, parallax mapping, screen space ambient occlusion* (SSAO), *dynamic shadows using shadow maps, render-totexture and full-screen post-processing effects*.

2. *Scripting*

   Script game engine dibuat dengan Mono 2.6, sebuah implementasi *opensource* dari .NET Framework. Programmer dapat menggunakan UnityScript (bahasa terkustomisasi yang terinspirasi dari sintax ECMAScript, dalam bentuk JavaScript), C#, atau Boo (terinspirasi dari sintax bahasa pemrograman phyton). Dimulai dengan dirilisnya versi 3.0, Unity menyertakan versi MonoDevelop yang terkustomisasi untuk debug script.

3. *Asset Tracking*

   Unity juga menyertakan Server Unity Asset sebuah solusi terkontrol untuk *developer game asset* dan *script*. *Server* tersebut menggunakan PostgreSQL sebagai *backend*, sistem audio dibuat menggunakan FMOD library (dengan kemampuan untuk memutar Ogg Vorbis compressed audio), video *playback* menggunakan Theora codec, engine daratan dan vegetasi (dimana mendukung *tree billboarding, Occlusion Culling dengan Umbra*), *built-in lightmapping*

dan *global illumination* dengan *Beast*, *multiplayer networking* menggunakan RakNet, dan navigasi mesh pencari jalur built-in.

4. *Platforms*

   Unity *support* pengembangan ke berbagai *platform*. Di dalam *project*, *developer* memiliki kontrol untuk mengirim keperangkat *mobile*, *web browser*, *desktop*, dan *console*. Unity juga mengijinkan spesifikasi kompresi tekstur dan pengaturan resolusi di setiap platform yang didukung. Saat ini *platform* yang didukung adalah BlackBerry 10, Windows 8, Windows Phone 8, Windows, Mac, Linux, Android, iOS, Unity Web Player, Adobe Flash, PlayStation 3, Xbox 360, Wii U dan Wii. Meskipun tidak semua terkonfirmasi secara resmi, Unity juga mendukung PlayStation Vita yang dapat dilihat pada *game* Escape Plan dan Oddworld: New 'n' Tasty.

5. *Asset Store*

   Diluncurkan November 2010, Unity *Asset Store* adalah sebuah *resource* yang hadir di Unity *editor*. *Asset store* terdiri dari koleksi lebih dari 4,400 asset *packages*, beserta 3D *models*, *textures* dan *materials*, sistem *particle*, musik dan efek suara, tutorial dan *project*, *scripting package*, *editor extensions* dan *online service*.

6. *Physics*

   Unity juga memiliki *suport built-in* untuk PhysX *physics engine* (sejak Unity 3.0) dari Nvidia dengan penambahan kemampuan untuk simulasi *real-time cloth* pada *arbitrary* dan *skinned meshes*, *thick ray cast*, dan *collision layers*.

2.7.2 Adobe Photoshop CS

Adobe Photoshop merupakan suatu aplikasi yang paling banak digunakan dikalangan desainer grafis dan pengolahan citra. Sejak pertama diperkenalkan oleh Adobe corporation pada dekade 90-an, Photoshop langsung mendapatkan tempat dikalangan profesional dan praktisi imege-editing dengan segala kecanggihan fitur dan kemampuan yang maksimal serta kemudahan dalam penggunaannya.

Adobe Photoshop yang dirilis pada tahun 2012 yang lalu oleh Adobe corp., adalah Adobe Photoshop CS dengan slogan yang cukup menjanjikan The

Professional Standard in Desktop Digital Imaging merupakan pengembangan dari Adobe Photoshop generasi sebelumnya, dan pada edisi terakhir ini sudah dilengkapi dengan fasilitas *Image Ready CS*. Kelengkapan ini dimaksudkan untuk memberikan fleksibelitas, efisiensi dan kemudahan dalam penggunaan, sehingga dapat dengan mudah membuat dan menyunting *image* dengan kualitas tinggi yang siap untuk dicetak, *publisher*, ataupun untuk ditempatkan di web, yang selama ini merupakan kendala untuk menempatkan image berkualitas tinggi pada web, karena berhubungan erat dengan beban *loading* data di sisi pengguna [16].

Gambar 2.9 Logo Photohsop

2.7.2.1 Area Kerja Photoshop

Beberapa Area kerja Photoshop, diantaranya [16]:

1. *Menu Bar* Berisi perintah utama untuk membuka *file*, *save*, mengubah ukuran gambar, *filter* dan lain-lain.
2. *Option* Berisi pilihan dari *tool* yang dipilih. Misalnya dipilih kuas/*brush*, maka ukuran/diameter *brush* ada di sini.
3. Gambar Menampilkan gambar yang sedang dibuat atau diperbaiki.
4. *Pallete Well* Cara cepat untuk mengakses *palet brushes*, *tool resets* dan *Layer Comps*. Juga dapat digunakan untuk meletakkan palet yang sering digunakan.
5. *Toolbox* Berisi *tool* untuk menyeleksi dan memodifikasi gambar.
6. *Palette* Berisi jendela-jendela kecil yang di dalamnya terdapat perintah dan pilihan untuk dokumen/gambar yang sedang dikerjakan.

2.8 Pengujian *Blackbox*

Black-box testing adalah metode pengujian perangkat lunak yang tes fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja. pengetahuan khusus dari kode aplikasi / struktur internal dan pengetahuan pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun di sekitar spesifikasi dan persyaratan, yakni, aplikasi apa yang seharusnya dilakukan. Menggunakan deskripsi eksternal perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. Tes ini dapat menjadi fungsional atau non-fungsional, meskipun biasanya fungsional. Perancang uji memilih input yang valid dan tidak valid dan menentukan output yang benar. Tidak ada pengetahuan tentang struktur internal benda uji itu.

Metode uji dapat diterapkan pada semua tingkat pengujian perangkat lunak: unit, integrasi, fungsional, sistem dan penerimaan.ini biasanya terdiri dari kebanyakan jika tidak semua pengujian pada tingkat yang lebih tinggi, tetapi juga bisa mendominasi unit testing juga. Metode ujicoba blackbox memfokuskan pada keperluan fungsional dari software. Karna itu ujicoba blackbox memungkinkan pengembang software untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba blackbox berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya [17]:

a. Fungsi-fungsi yang salah atau hilang
b. Kesalahan interface
c. Kesalahan dalam struktur data atau akses database eksternal.
d. Kesalahan kinerja
e. Kesalahan Inisialisasi dan terminasi

2.9 Skala Likert

Skala data yang digunakan untuk pengukuran variabel independen adalah skala likert. skala likert adalah skala yang digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dalam penelitian, fenomena sosial ini telah ditetepkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variable penelitian. Dengan skala likert, maka

variable yang akan diukur dijabarkan menjadi indikator variable. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyususn item-item instrumen yang dapat berupa pernyataan atau pertanyaan [18]. Data yang telah terkumpul melalui angket, kemudian penulis olah kedalam bentuk kuantitatif, yaitu dengan cara menetapkan skor jawaban dapat dilihat pada Tabel 2.1

Tabel 2.1 Penilaian Skala Likert

Alternatif	Skor
Sangat setuju / selalu / sangat positif	5
Setuju / sering / positif	4
Kurang setuju / ragu-ragu / netra	3
Tidak setuju / hampir tidak pernah	2
Sangat tidak setuju / tidak pernah	1

Kemudian dengan teknik pengumpulan data angket, maka instrumen tersebut misalnya diberikan kepada 100 orang yang diambil secara random. Dari 100 orang tersebut setelah dilakukan analisis misalnya :

1. Sebanyak 25 orang menjawab sangat setuju (SS)
2. Sebanyak 40 orang menjawab setuju (S)
3. Sebanyak 5 orang menjawab kurang setuju (KS)
4. Sebanyak 20 orang menjawab tidak setuju (TS)
5. Sebanyak 10 orang menjawab sangat tidak setuju (STS)

Data interval tersebut kemudian dianalisis dengan menghitung rata-rata jawaban berdasarkan skoring setiap jawaban dariresponden. Berdasarkan skor yang telah diteteapkan dapat dihitung sebagai berikut.

Jumlah skor untuk 25 orang yang menjawab SS	= 25 x 5	= 125
Jumlah skor untuk 40 orang yang menjawab S	= 40 x 4	= 160
Jumlah skor untuk 5 orang yang menjawab KS	= 5 x 3	= 15
Jumlah skor untuk 20 orang yang menjawab TS	= 20 x 2	= 40
Jumlah skor untuk 10 orang yan menjawab STS	= 10 x 1	= 10
Jumlah Total Nilai		= 350

Jumlah skor yang diperoleh dari penelitian adalah 350. Jadi berdasarkan data itu maka tingkat persetujuannya yaitu jumlah total nilai dibagi jumlah total responden = (350 : 100) = 3,5 secara kontinum dapat dilihat seperti pada Gambar 2.3.

Gambar 2.10 Secara Kontinum

Jadi berdasarkan data yang diperoleh dari 100 responden maka rata-rata 3,5 terletak pada daerah setuju.